

# Tips for Encouraging and Completing Undergraduate Research

Jason Vroustouris  
Department of Computer Science  
University of Illinois at Chicago  
851 S. Morgan St., MC 152, 1120 SEO  
Chicago, IL 60607  
jasonv@jasonv.com

Mitchell D. Theys  
Department of Computer Science  
University of Illinois at Chicago  
851 S. Morgan St., MC 152, 1120 SEO  
Chicago, IL 60607  
312-413-9267  
mtheys@uic.edu

## ABSTRACT

In this paper, we describe our experiences working on an undergraduate research project within the Department of Computer Science at the University of Illinois at Chicago. The chosen project was to develop a micro-controlled CPU simulator (called MythSim) to replace and improve the current outdated simulator in use in the classroom. This paper will benefit undergraduate students and potential undergraduate research advisors who want to develop new software or modify existing software for classroom use. The paper presents general tips for advisors and students who want to become active in undergraduate research project, with anecdotes from our experience developing the MythSim CPU simulator.

## Categories and Subject Descriptors

<http://www.acm.org/class/1998/overview.html>

**General Terms** Design, Human Factors

## Keywords

Index Terms - computer architecture, computer science, curriculum, java, microcode, simulator

## 1. INTRODUCTION

Undergraduate research can be a great way to expose students to a research environment and give advisors a chance to prepare students for graduate level positions. Unfortunately, opportunities are often missed because advisors are unsure how to encourage first time researchers and students have trouble completing these projects. To address these concerns, the co-authors share their successful research experience and provide a series of tips for advisors and students outlining the benefits that can result. The paper begins by introducing the MythSim simulator and how the project started. Then tips for the advisors and students are presented. The paper concludes with a brief discussion of some of the benefits obtained through the experience.

## 2. MYTHSIM: A CPU SIMULATOR

The MythSim CPU simulator is a central part of the undergraduate computer architecture curriculum in the Computer Science and Electrical and Computer Engineering departments. In the Computer Science Department the simulator is used in a two-semester sequence that covers digital logic, datapath design and assembly programming. As a final project the students'

implement a simple assembly language in microcode. By writing their own control code they gain a fundamental understanding of how the computer works. [1]

The original MythSim [2] was an in-house tool used to teach basic computer architecture principles. Problems with the user interface and portability of the code led faculty and students to consider updating the simulator. Jason was an undergraduate student who just completed the course that used the simulator. He came to office hours to ask about the possibility of porting the simulator for undergraduate research credit. This began a research and development that led to a completely redesigned simulator, based on the needs of the faculty and the students. A screen shot [Figure 1] of the simulator shows the improved design.

From the users' perspective, the result of the research performed was an improved portable simulator that assists the undergraduate learning experience.

## 3. TIPS FOR ADVISORS

Many research projects are only of interest to the Advisor and their Students. To remedy this situation one should consider developing software for use in the classroom. Is there some software that would have helped you convey a particular topic in a course? Was there a concept that would have been easier understood if a graphical simulation with tunable parameters was available? Such projects are typically of more interest to a wider circle of people and benefit students of the future. Professors in the department who periodically teach the class that would use the software being designed, and students that are in the class or expect to take it in the futures are the focus group and can provide useful information that will help direct your design process.

The remainder of this section provides tips for advisors who want to encourage undergraduates to perform research. These tips were developed while the co-authors were working on an undergraduate research project that is currently used in the undergraduate curriculum in the Computer Science Department at UIC.

### 3.1 Engaging Undergraduate Research

All educators have at one time or another thought about some simulation or demonstration that would make their courses more interesting and capture the students' interest. Due to time constraints educators rarely have time to complete these projects and so they eventually fall from our minds until the next time one teaches that course.

Enter the undergraduate research project. This project is a perfect opportunity for undergraduates to get their hands involved in research, and for educators to get all the nifty simulations and demonstrations for their courses. This arrangement becomes a win-win situation, with the student performing the work learning about managing a project, working with real situations, documentation, etc.; the instructor now having a new learning tool to help the learning process; and future students benefiting from using the new system to assist in their learning.

### 3.2 Finding the Student

This subsection can also be called “What to do when a student tracks you down.” Undergraduate students are not normally thinking about doing a research project, although many more should be. The time at which the students should be completing such a project, close to their senior year, is already filled with senior level electives, job searches, and the light at the end of the tunnel. But this is the very time where working on a research project can tie up those loose ends in the students’ minds and solidify many of the concepts they have been learning during their undergraduate studies. In addition, these projects have the added bonus of capturing students’ attentions and introducing the idea of graduate school which is often misunderstood by undergraduate students.

Choosing students that you have interacted with in classes that you teach is a start. Finding a student who has just completed the course that would use the project is even better, as the concepts and problems with the course are still close at hand.

### 3.3 Being Open

The nature of a project is affected by how it begins. A student might approach a professor with an idea for a new tool. A professor might approach a student to repair bugs in an existing application. Two or more students might have met and then pitch an idea to a professor. A group of professors could recognize the need for a tool and post a request on a list serve.

Be sure to be open about what you expect to get out of this project. Try and understand the situation for all involved. If either party is unhappy with something, it should be addressed quickly. Everything can be made easier if the student does the bulk of the work before registering for credit. This puts no pressure on the student to finish, nor on the advisor to accept less than a completed project. In addition, it allows for unexpected events that may cause the student to not finish the project.

### 3.4 Matching Projects and Students

Once you have some projects in mind some care must be taken in matching students and projects. An inventory of the student’s current skills and which skills they would like to learn should be created. This facilitates a better matching of skills required to skills possessed. The advisor should be prepared for students’ not always being able to complete or learn certain skills in a timely fashion. This should not hinder the project, instead a different approach to solve the problem should be examined or the piece of the system that requires the skill should be redesigned or possibly removed

### 3.5 Managing Undergraduates

Unlike a graduate student an undergraduate student has a much higher course load. This does affect their productivity and without careful planning can lead to weeks where no progress is made.

This should not be looked upon as a negative. If the student truly enjoys the project and wants to see it to fruition, the time spent away from the project is just as painful for them as you. When time permits the student will again continue working on the project and the advisor should continue to inquire about progress on a regular basis to encourage progress.

Because of the undergraduate students’ more restrictive time constraints a multi-semester project may be the solution. Multi-semester projects are not easier or harder, just longer for completion. This opportunity should not be viewed as a means to make the system more complicated, but instead as a process that allows for continued feedback and improvements based on user comments.

### 3.6 Planning for the Future

To encourage more educators and students to use the software created, the results have to be made available, documented, and freely distributed. With the advent of open source software this becomes much easier if the system created is released with an appropriate open source license [3, 4]. Releasing the code to the open source community means that after the student completes the project and moves on, there is still the potential for improvements, modifications, and newer releases by members of the open source community. In addition, the posting of the project on a site like SourceForge [5] creates a community of users and a central location to find updates and revisions.

## 4. TIPS FOR STUDENT RESEARCHERS

This section gives hints and tips from the students’ perspective. Jason was an undergraduate with the Department of Computer Science when the work was completed. He is currently a graduate student with the same department.

### 4.1 Start Unofficially

An undergraduate research project can be a major challenge and very time consuming. A good way to avoid problems is to meet with a professor in your second or third year and begin working on the project unofficially. By not officially registering you can determine if your schedule allows working on a project and quit if your circumstances change. Your advisor understands that research does not always work out. The project may never materialize, but your advisor can help you discover your abilities and may be able to direct you to a project that is better suited for you.

In our case, by starting unofficially, there was no pressure to finish the project quickly. It ended up taking about a year and a half to complete the project. Jason took the credit for the research during his last semester with most of the coding already completed. This reduced his overall workload and was like a reward for completing the project.

### 4.2 Form A Research Partnership

Think of your advisor as less of a boss and more of a partner in research. Ask yourself if this person will be able to provide the support such as periodic meeting and feedback. Not all professors are created equal. Read though the professor’s website and see what their research interests are. Make sure that this is someone you can work with for an extended period of time.

When we first met to talk about MythSim our individual roles became clear. My advisor was an expert in how the simulator

should run, but was not an expert in Java. I had just taken a class in Java and was not a hardware expert.

The project was my first experience researching and it was difficult to give up total control of the project. In the beginning, I used my programming skills to build the core of the simulator with my advisor's guidance. As the project progressed, we worked together to design a new user interface component. At the end, we co-wrote a conference paper about the simulator and this paper about the experience.

### 4.3 Get Feedback

Your advisor is a busy person and can't be bothered with all the little problems that arise. However, it is important to interact during the development, especially when you need feedback. After you make a major new innovation, make an appointment to show it to your advisor. Before you make your innovation a fully functional piece of your program, be sure that your advisor feels that is worth adding. Remember that your advisor has experience in the classroom. They know a thing or two about the realities of the classroom and the environment where the software will be used.

Feedback was very important for me when developing MythSim. During our meetings, we would run the latest version and I would explain the new features, and how they were implemented. Some of my ideas turned out to be too much work with not enough benefit for the students. By the end of the meeting, we would agree on what goals I should reach before making another appointment.

### 4.4 Visit the Classroom

As a software developer it is important to know the audience who is using your software. For the students in the class, especially the more advanced students, it can be a rare treat to meet the person who developed the software they use.

For MythSim, I visited the classroom twice a semester. The first time, I introduced myself, gave a little history about the software and then showed the students the interface. This usually lasted for about one class period where the professor and I highlighted different parts of the simulator that would be important for the students to be aware of. The second time was usually the day after the students turned in their final project with MythSim. I would ask the students about their experience with the simulator and spirited discussion typically ensued.

### 4.5 Track Your Hours

Undergraduate research classes usually have a minimum number of hours that should be worked. Advisors often say that they are not concerned with how much time you put, just with the goal that you set forth to work on. However, at some point you may feel that you have put in more than your fair share of time. By having a log of the time you put in and a summary of what you were doing, you can justify changing the goal.

For the first semester coding MythSim I used a spreadsheet to record a start time, end time, and what I did. It was helpful to look back later, and see how the software evolved. After that I stopped keeping track; however it would have been nice to know how many hours the whole project took.

### 4.6 Keep All Versions of Your Source Code

For every day you work on your code, make a copy of the code. If you make a major change make a copy before you do. Keep the revisions until you get credit for the class. If you want to adjust your project's goals, the different versions will be a simple way to show the work you did.

For MythSim, a very basic naming convention was used. The code was kept in a dated folder. At the beginning of a coding session, it was copied and renamed with the current date. If another version was desired for the same day, a letter was added to the end of the date. By using the format "YYYY-MM-DD" the folders sorted nicely. This proved to be a very simple and effective way for one person to work on the code base.

When we decided to release the simulator as open source, we did away with the dated folder system, and used the CVS system on the SourceForge website [5]. This system has a steep learning curve but proved to be a much more robust and a critical piece for facilitating multi-user development.

## 5. BENEFITS

The undergraduate research experience is often overlooked by educators and students because of the commitment required by a student and the unstructured nature of research. However, by attempting research a student can develop a better understanding of what it means to "perform research" and be a "researcher."

When advisors and undergrads work together, the project can be a contribution to the greater learning community. If the software is robust enough, it may be appropriate for use in the classroom. If steps are taken to package and document the software, other schools may be able to use it. If the software is innovative, it may be a worthy subject for a conference or journal paper.

Whether a student is interested in a career in business or academia developing an application for classroom use has many benefits. An undergraduate development project can help build relationships with staff and faculty that may lead to job opportunities or support in a graduate program. Fellow students may be excited to be using a new software tool that makes learning easier and get even more excited when they learn an undergraduate student worked on the project. If the project is well received in the classroom, it may be worthy of publication.

The faculty and staff at a university know that the work that they do contributes to providing a learning community. When undergraduates and faculty work together to develop software for the classroom they give back to that community. As a result, the benefits for the developers, users and the school can be realized for years to come.

## 6. ACKNOWLEDGEMENTS

Special thanks to Pat Troy and Dale Read for being the first instructors to beta test our software in their classes.

## 7. REFERENCES

- [1] Jason Vroustouris and Mitchell D. Theys, "MythSim: The Mythical Simulator for Real Students", *Frontiers in Education* 2004, October 2004
- [2] MythSim Website - <http://www.mythsim.org/>

- [3] GNU General Public License - <http://www.gnu.org/licenses/gpl.txt>
- [4] Open Source Initiative OSI – Licensing, <http://www.opensource.org/licenses/>

- [5] Source Forge: the world's largest Open Source software development website - <http://www.sourceforge.net/>

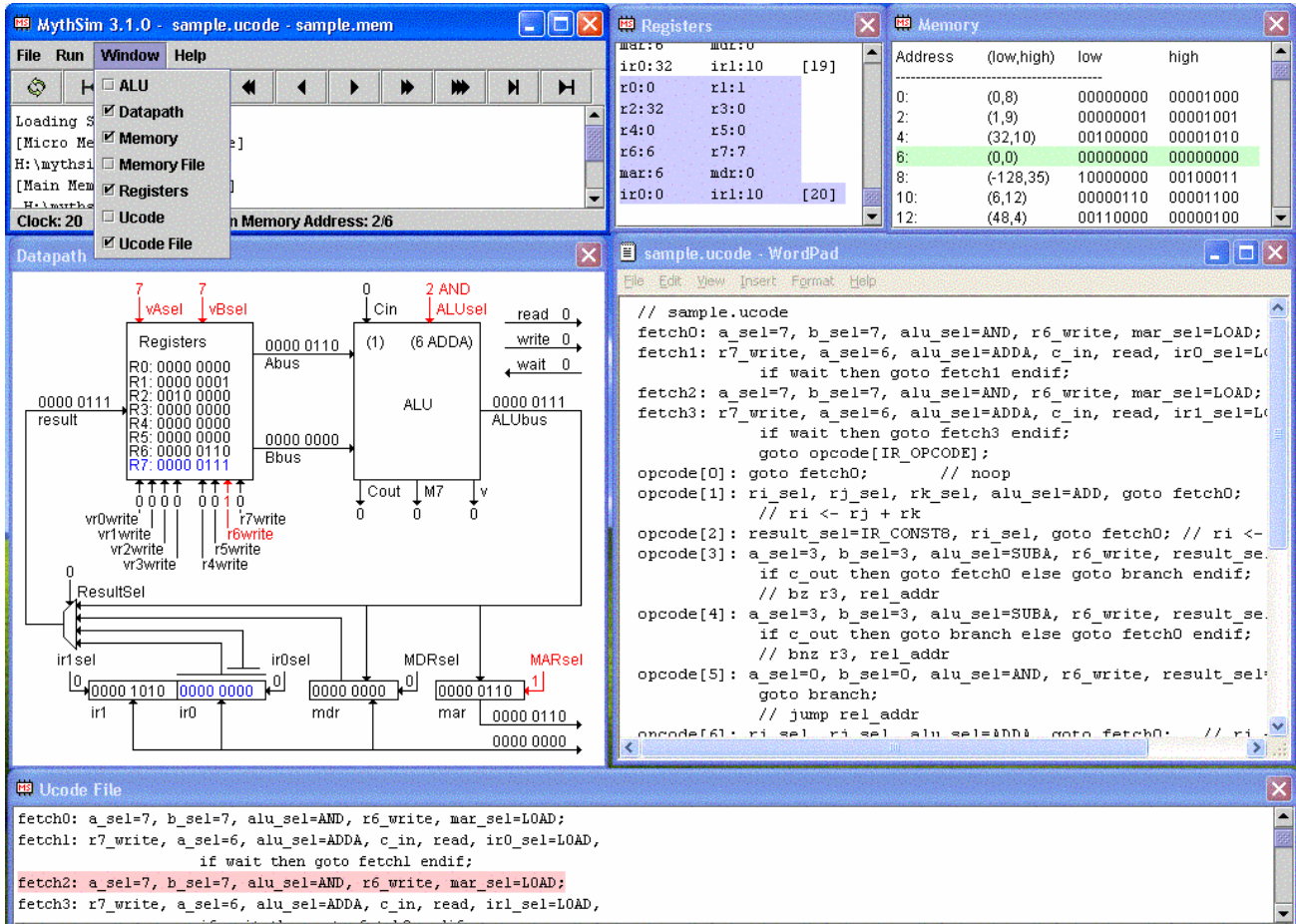


FIGURE 1  
Debugging With MythSim and WordPad on Windows