Investigating Energy and Security Trade-offs in the Classroom With the Atom LEAP Testbed

Peter A. H. Peterson, Digvijay Singh, William J. Kaiser, Peter L. Reiher
{pahp, reiher}@cs.ucla.edu
{digvijay@ucla.edu, kaiser@ee.ucla.edu
University of California, Los Angeles

Abstract

We recently used the Atom LEAP (a new energy measurement testbed developed at UCLA) as the foundation for CS 188, an undergraduate research seminar investigating potential trade-offs between security and energy consumption. Twenty-three students, in five groups, researched the energy costs of full disk encryption, network cryptography, and sandboxing techniques, as well as the potential savings from two concepts: offloading security computation, and enabling user-level applications to modulate their security behavior based on battery capacity and environmental security. The Atom LEAP is an exciting and powerful tool. A self-contained energy measurement platform, it can generate 10,000 component-level power samples per second during runtime. The Atom LEAP synchronizes individual samples to the time stamp counter of the Intel Atom CPU, allowing us to measure small code segments in the kernel or in user space. The success of CS 188 was possible because of the Atom LEAP's unique capabilities and ease of use. Following the success of the class, we are working to improve the hardware and software tools, in the hope that the Atom LEAP might someday become a widespread tool for energy research and education.

1 Introduction

Mobile and embedded devices face difficult trade-offs between security, functionality, and power use. Reducing the energy consumption of computers has become a key issue in the fields of computer science and electrical engineering. In order to meet the needs of research and education in these areas, we need suitable tools and courses. To that end, we present the Atom LEAP (Low-Energy Aware Platform) testbed, a self-contained energy measurement and education platform; and we describe our experiences using the testbed in an undergraduate research course examining trade-offs between security and energy consumption.

The Atom LEAP is an inexpensive, component-level energy-measurement platform capable of acquiring 10,000 samples per second, synchronized to the execution of code in kernel and/or user space. To our knowledge, the Atom LEAP is a first in many categories, the most important being that it can synchronize high-resolution sample data with the Atom CPU's nanosecond granularity timestamp counter (TSC). Based on a low-power, mini-ITX motherboard and a USB-enabled sampler, it is self-contained and portable, weighing only about ten pounds. It is relatively inexpensive for a research tool, with a total equipment cost of approximately $1,500. The Atom LEAP workbench software includes a Linux-based open-source software stack with both low-level and high-level user-friendly tools for data collection, workload execution, hardware sharing with multiple users and local or remote data analysis.

In the winter of 2011, Intel supported UCLA CS 188, an undergraduate research seminar which used the Atom LEAP to explore trade-offs between security and energy consumption in the context of a fictional pad device. Five student groups investigated different high-level trade-offs between security functionality and energy cost in the areas of disk encryption, network encryption, virtualization and isolation, offloading computation, and security-sensitive applications. Our experience was very positive, providing interesting results from both a security and efficiency perspective, introducing a number of students to performance analysis, and validating the Atom LEAP as an educational tool.

2 Atom LEAP

2.1 Introduction to the Atom LEAP

LEAP (Low-Energy Aware Platform) originally started as an embedded network power manager and monitor[9], but it has grown into a full-fledged system power experi-

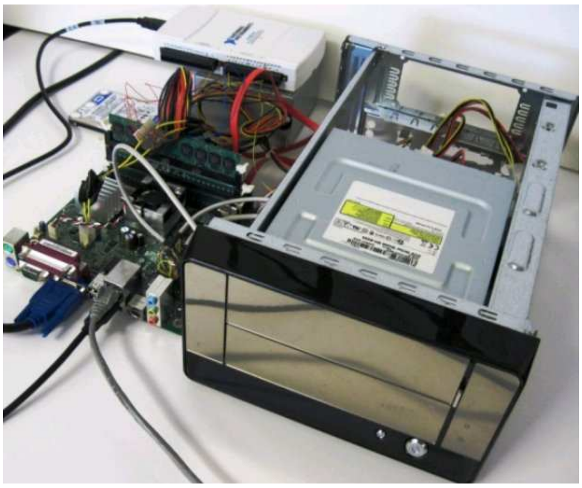

Figure 1: Atom LEAP next to case, with DAQ behind.

mentation platform[12, 13]. The current generation tool, the Atom LEAP, is pictured in Figure 1. It is built around an Intel Atom mini-ITX motherboard and an external, USB National Instruments (NI) Digital Acquisition Device (DAQ).

During workload execution, the NI-DAQ acquires up to 10,000 samples per second of component-level power consumption data for multiple devices including, but not limited to: CPU, RAM, hard disk, northbridge, USB, and power supply. The LEAP is capable of measuring very small segments of code. While this granularity is unnecessary for normal applications or benchmarks, it is essential for capturing the energy use of microbenchmarks like system calls. The Atom LEAP attributes power to specific events by use of a novel synchronization scheme linking a sync signal to readings of the Atom's nanosecond timestamp counter (TSC).

Instrumentation within the workload reads the TSC value at the start and stop of target tasks. We call these TSC values "energy calipers" because they measure the specific elapsed time between the start and end of a given task. For C code, energy calipers are created by a small amount of inline assembly which executes very quickly; for interpreted code, we read the TSC with a command line utility. Data is buffered continually by the DAQ, and read periodically by the Atom into a RAM disk. Following a complete test, Atom LEAP tools create basic statistical summaries of the data. This ensures minimal workload interference while also providing extremely accurate sampling information.

2.2 Atom LEAP as Research Testbed

The Atom LEAP is a powerful tool for researchers interested in low-level power consumption data. Component-level measurement is important because of the potential for inter-device interactions[6]. Another reason for the necessity of component-level power measurement relates to the various power modes of computer hardware. Modern hardware often includes explicit power modes, such as dynamic voltage frequency scaling (DVFS) for CPUs. These modes enable direct power versus performance trade-offs. However, in addition to these explicit modes, most computer hardware demonstrates a range of energy consumption based on factors such as its current explicit mode and the rate of requests being issued to it. For example, CPU, RAM, and hard disk energy consumption can fluctuate wildly, within a single operational mode.

The Atom LEAP, via energy calipers, can measure code segments in both kernel and user space. This can provide significant insight into the power consumption of computer hardware at all levels of the software stack. For example, we can use energy calipers to measure the component-level cost of different file system modules in the kernel. This unique perspective can provide insight into hardware and software behavior, visually illuminating design issues as well as potential avenues for future power savings.

The Atom LEAP's design is simple enough that it can be easily applied to any hardware with a reliable clock and a synchronization signal source. Current work includes a battery-powered Mobile LEAP and a DSP LEAP device. While instrumenting the hardware of a new platform is a delicate and time-consuming task, having this common set of techniques and software means that researchers do not need to completely "reinvent the wheel" every time they wish to instrument a new system.

3 Atom LEAP in the Classroom

Performance measurement is an important (yet neglected) issue for education in our field, especially as the demand for energy efficiency continues to increase. In the past, runtime was *the* key measure of utility, and could be measured with simple utilities like `time`. However, investigating the energy issue adds multiple dimensions to the problem, including component-level costs and other issues such as constant power dissipation, static leakage, heat, and so on. This is a richer problem space and provides an opportunity to "dig deeper" into hardware and software, as well as to teach valid system performance measurement techniques. But direct energy measurement requires hardware.

The simplicity, accessibility, and hands-on nature of Atom LEAP also makes it an ideal tool for investigating these issues. While other direct and indirect methods of energy measurement and modeling are valuable, the simplicity of Atom LEAP eliminates unnecessary layers of abstraction and adds a strong dimension of realism to the education experience.

While it is simple and easy to use, Atom LEAP is not a toy. Its sampler is industrial strength, its computer hardware is already widely deployed in netbooks, and it is beginning to compete in the tablet space. Yet, the simplicity and familiarity of the LEAP environment leads to a system with few barriers to entry. Its low cost means that even small labs and departments can afford to construct one of their own. While its intuitive interface and ease of use make it easy for students and researchers to quickly scratch the surface of a problem in minutes, its accuracy and granularity also provide enough depth to make great inroads into energy consumption problems.

Atom LEAP provides accurate, component-level data at an extremely high granularity. As opposed to mathematical models or simulators, LEAP provides physical, hands-on instrumentation that students can see, use, and modify. Because of this realism, if students wish to run accurate experiments, they must learn and use good experimental practices (such as avoiding caching effects or interfering workloads). The LEAP software facilitates good practice by generating time-stamped archives of test data including tab-separated results, and tools for automating repetitions, without reducing the process to pushing a button.

A complete Atom LEAP is inexpensive, open-source and robust. With a few minutes of training, students work directly with the hardware, enabling them to take ownership of the experimentation process. Because the Atom LEAP runs a virtually unmodified version of Linux, the world of open source tools and packages is readily available for experimentation. Measuring discrete events within a program currently requires instrumenting source code; however Atom LEAP can also sample energy for closed-source, binary workloads at the task level.

While the LEAP hardware is easy to use in person, it can also be shared across a network. Linux is above all a networked operating system, so students can remotely perform experiments that don't require hardware changes. When combined with Atom LEAP's serial console and a networked power switch, students can even perform risky kernel experiments remotely, confident that they can recover the system without assistance.

4 Case Study

4.1 CS 188: Energy and Security Tradeoffs

The simplicity and flexibility of the Atom LEAP make it an exceptional testbed for energy-related education. We support this claim by describing highlights from UCLA CS 188, an upper-level undergraduate research course, supported by Intel.

CS 188 explored trade-offs between energy use and security functionality within the context of [the fictional] Banana Computer's new zPad[11]. We defined the following five project areas in terms of a research and development project investigating proposed features for the zPad. Four to five students were assigned to each project based on their preferences, and we met weekly for progress reports and discussion.

4.2 Full Disk Encryption

4.2.1 Project Description

The CryptoDisk team's broad charge was to investigate the energy cost of providing full disk encryption. While we originally hoped to compare hardware and software encryption costs, our encrypted disks did not arrive in time. Instead, the members of the group chose to investigate the use of compressed file systems to reduce the energy cost of software full disk encryption in Linux.

The main question the CryptoDisk group addressed was whether compressing data before encrypting it was a power win and whether the compression algorithm and filesystem used made a significant difference. They compared Btrfs[1] and SquashFS[2] with a writable overlay file system, using LZO, gzip, and XZ compression. (SquashFS with the overlay was more efficient than Btrfs, so henceforth we will only describe SquashFS results.) LZO is fast, but does not have the best compression ratio. XZ compresses very well, but typically takes longer than the others. gzip represents a compromise between speed and compression ratio.

4.2.2 Results

Figure 2 shows the energy used by Atom components to compress and encrypt (also decrypt and decompress) a large number of files in the process of searching for a string in the Linux kernel source. A variety of workloads showed generally similar results, except for video playback, probably due to video's low compressibility. The energy consumed by the bridge is not shown; it is similar to the other components, but is so much greater that it dwarfs the consumption of the other components.

Figure 2 shows that for this group's tests, it is much better to compress data before encryption. All tested compression algorithms perform significantly better than the control. But a compression algorithm that sacrifices speed for compression ratio (e.g., XZ) is much inferior to algorithms that favor speed over compression. However, speed is not unequivocally better. gzip, which offers a compromise between speed and compression, is the winner overall. The measured workload under gzip actually took less time to run than LZO (even though LZO compression is *much* faster than gzip), because the workload run time includes both the compression and I/O costs. This result could not be predicted without testing. Only

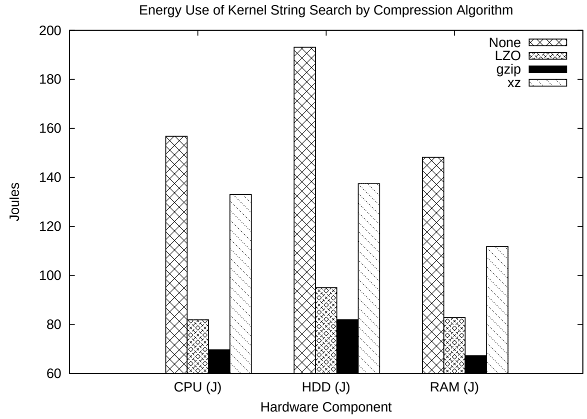

Figure 2: Compression saves energy for FDE. SquashFS with gzip uses less energy than LZO or xz.

by performing significant tests was the group able to tell that the compromise of gzip was best.

4.3 Network Encryption

4.3.1 CryptoFlex Description

The CryptoFlex group investigated how cryptography could be modulated to save power. Cryptography always costs energy, but the cost varies by cipher and parameters such as keysize. The students investigated and compared the costs of available algorithms and estimated how much energy could be saved if the best (potentially more expensive cryptography with large keys) was used for mission-critical communications, while using more efficient (but still widely used) ciphers for everything else.

4.3.2 Results

Generally, the costs of encrypting even a moderate amount of data (such as 1 Mbyte) are not very high, but the CryptoFlex results nonetheless show that different ciphers use very different amounts of power. To measure this in a network communication context, the group used the energy calipers of Atom LEAP to isolate the encryption and decryption routines in SCP. Figure 3 shows the costs for the CPU, disk, and RAM for the ciphers 3DES, AES-128, and Blowfish. As the figure shows, 3DES is the most power-hungry, while Blowfish uses less energy on average than its competitors. Only for the RAM do these preliminary tests suggest statistical significance for the difference, but that element shows a 20% power improvement over AES.

The primary differences in the power costs for different cryptographic algorithms on the Atom are tightly tied to their runtimes, illustrating clearly that running less code burns less energy. It could be interesting to see whether various explicit power modes affect energy use,

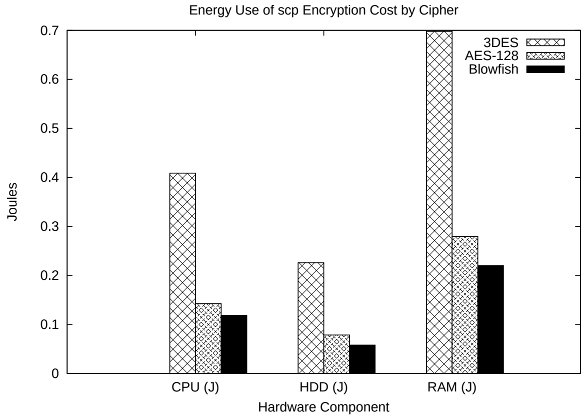

Figure 3: Blowfish is the most efficient cipher.

but our particular Atom board lacks frequency scaling and has only two C-states — run and stop.

4.4 Energy- and Security-Aware Applications

4.4.1 PowerSecZone Description

The PowerSecZone group was asked to investigate how security-sensitive applications might be able to save energy if they could query power levels, security posture, and environmental conditions. For example, if a system has a low battery but is working on non-sensitive data in a secure environment, how much power could be saved without unduly affecting security?

The group built a general framework for applications to receive power, security, and environmental status updates. They then altered three applications to make use of the framework: a web browser, an instant message client, and a virus scanner. All produced interesting results, but we will focus on results from the virus scanner here.

Virus scanning is critical to security, but can be quite costly as it tends to access many files. In addition to pattern matching, modern virus scanners run potentially expensive algorithms to determine if malware is hiding in a file. The group altered ClamAV to use different virus scanning options based on the power state of the system, and then measured the power benefits of reducing the amount or complexity of scanning.

Among the options they examined were scanning the whole disk, scanning just the user's home directory and below, skipping particular types of files (e.g., pictures), and turning off algorithmic detection. The tests were performed on a disk containing a typical OpenSUSE GNU/Linux installation, plus an actual home directory containing a large number of files. Among those files were a few very large compressed archives and a large number of smaller images of a particular format. The total size of the archives was roughly equal to the total size

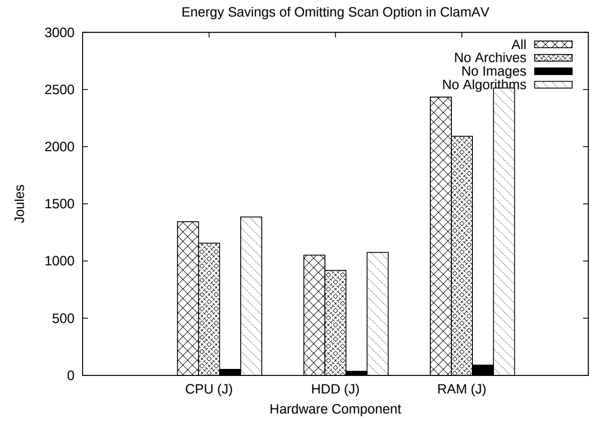

Figure 4: Omitting images saves power because of the number of files, not the number of bytes.

of the images.

4.4.2 Results

Figure 4 shows the results of omitting ClamAV options. The data series labeled normal is a scan of the user's home directory. The full root scan was vastly more expensive, and is not shown. The other series each exclude a different element each (e.g., algorithmic detection). Intuitively, algorithm detection seems like it might be costly; however, its energy cost is statistically insignificant. (It appears lower on this graph, but statistical measures of confidence at the 95% level show no significant differences in the two options.)

The graph clearly shows that the largest determinant of scan cost is the *number of files*, not the number of bytes or the complexity of the scanning algorithm. The "no image" option is much cheaper than the other options, even though it scans around the same number of bytes as the more expensive "no archive" option, because it scans many fewer files. This suggests that methods of prioritizing scan order could offer a promising trade-off between security achieved and power spent.

4.5 Isolation and Virtualization

4.5.1 ElectricSandbox Desription

For machines without power limitations, one popular method of improving the security of downloaded code is to run it in a "sandbox" environment, limiting the amount of damage it could do if it proves malicious. There are different types of sandboxing technologies with different requirements, providing different levels of protection. We expected sandboxing to require more energy, but how much? And what was provided in return?

The ElectricSandbox project examined the costs of running application code in Linux and in different types of sandbox environments. The group tested Linux (as a control), chroot jails, and VirtualBox[3] with various user workloads. While they originally hoped to test User-Mode Linux[4] as a middle ground, they were unable to get it working in time. The students also had problems getting all tests to work in all three environments, but they still completed the project.

4.5.2 Results

In the end, the group's results were not surprising. VirtualBox is *vastly* more expensive in energy use than the other options, in part due to our Atom CPU lacking hardware virtualization extensions. The *best* case was a penalty of 60% more energy to run inside VirtualBox. Costs of double or even five or six times the energy were more typical. Given the number of high-profile information leakage issues facing applications on mobile devices, perhaps better virtualization an isolation facilities are in order.

Chroot jails naturally had a much smaller performance penalty, about 4% on average. At the very least, chroot jails may be an economical way to provide a little more isolation between processes. The group was disappointed that they were not able to get User-Mode Linux working in time, as it seemed to be a nice "middle ground" between the VirtualBox and chroot.

4.6 Offloading Security Operations

4.6.1 Offloading Description

The Offloading group looked at whether security-related tasks could be performed remotely with equal security and less power, or whether the increased communication costs would eliminate the savings. The example we highlight from their work is offloading cryptographic signatures.

The scenario is that the user of the pad computer wants to send a cryptographically signed message using her certificate. She has access to a trusted server via an untrusted wireless network. Thus, she must encrypt the transmission to the server. The server receives the data, creates the email message, signs, encrypts, and sends the message. The group used scp to encrypt the network transfer of the message, testing various message sizes and encryption options. Signatures were always created using public key cryptography via X.509 certificates, SHA-1 hashing, 1024 bit public keys, and RSA for cryptographic signing. In Figure 5, we show one sample result, where the group sent a 256-byte email message (not counting headers) using 3DES to encrypt it.

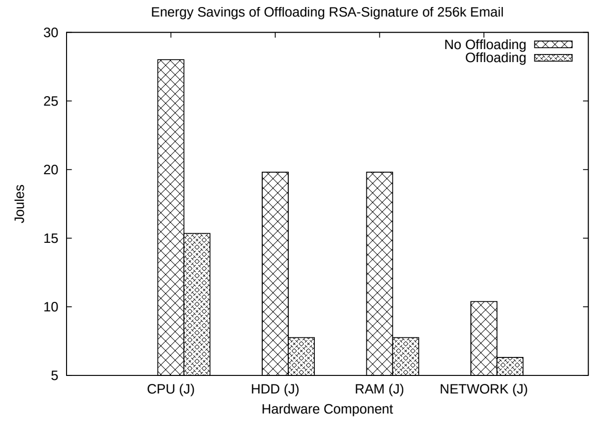

Figure 5: Offloading work can result in a power savings.

4.6.2 Results

Offloading this security task — over a wireless network — even while paying the cost for cryptography across that network, proves to be a major energy win. For some system components, nearly half of the power is saved by making this choice. In part, this is due to run time. Because the server is more powerful, the overall operation takes less time — 6.18 seconds on average versus 10.67 for the non-offloading case. However, in addition, some components of the Atom motherboard used fewer watts for the offloading option than the local option, particularly disk and memory. These differences were substantial — on average 40-50% lower for the offloading option. The WiFi network power cost in watts was slightly higher for offloading, but the improvements for memory and the hard drive easily overcame that difference. So offloading secure email is not only quicker, but burns less power per second. Results for other message sizes and cryptographic options differed only in magnitude. Offloading this operation was always cheaper.

5 Lessons Learned

5.1 Course Design

The Atom LEAP is a good platform for research courses. Students in this course had the flexibility to take their projects in unexpected directions, highlighting the platform's suitability for more open-ended courses. The Atom LEAP provided a valuable opportunity for students to gain legitimate research experience.

Undergraduate computer science research courses can be dynamic and challenging. Students seemed to find using real systems engaging, exciting, and challenging. We tried to ensure that each student performed at least some experiments, and we feel that all groups made interesting discoveries (described in Section 4). The investigative process required students to dig deeper into certain areas than they probably would have in a more structured class, and students seemed to enjoy that. However, this "depth over breadth" could also be seen as a potential drawback, since student experience depended heavily on the evolution of their project. Finally, teaching a research class also required a smaller investment of time in terms of course development, but required much more instructor time throughout the quarter for meetings and other student support.

Identifying and assigning project groups worked well. We picked our five high-level areas, and assigned students based on their subject matter preferences. While assigning topics limited student freedom, it saved class time and ensured that the course would include a range of compelling research areas.

5.2 Managing Expectations

Students and instructors should collaborate to develop a realistic plan that will be successful even if some components are not feasible. With traditional homework, only the answers are unknown — and then only to the students. With research, a good deal is unknown to both students and instructors. Performing research means dealing with the unpredictability of the real world. Instructors should help students plan for both the "known unknowns" and the "unknown unknowns," so that the success of the overall project is not left to chance.

Projects should depend on all group members as equally as possible. Every student group in CS 188 encountered tasks that seemed simple on paper, such as installing a piece of software, but that took many times longer than anyone expected, due to version incompatibilities, lack of experience, poor documentation, hardware issues, underestimation, or combinations thereof. Every additional piece of software that must be installed increases the chance of obstacles to progress, and cutting edge software even more so. While some groups were able to overcome these issues through the extreme dedication of one or more talented students, other groups did not have that luxury. Whenever possible, the success of a quarter-long project should not rely on the extraordinary effort or preexisting experience of a few individuals.

5.3 Test Wisely

Prioritize tests, run "throwaway tests" early, and improve or expand them once they work. Tests should be prioritized, and everything less than critical priority should be run only after all critical tests are complete. Putting an emphasis on getting things working — even if imperfect to start — helped keep students from going too far down any particular "rabbit hole." Experiencing the work and kind of unpredictable obstacles that appear in real research helped students get a sense for the length of time running a single set of tests would take. Once simple tests ran properly, we helped them estimate the time required for a real testing session. This perspective helped keep students motivated and making wise time management decisions.

Script all workloads and don't run final tests until you're *really* ready. In our own research experience, it is a common testing mistake to optimistically spend time running a complete set of tests "while it's fresh in your mind," only to find later that you have forgotten exactly how the tests were run, that when you try it again the results are inexplicably different, or that some necessary change has invalidated the data you just spent hours painstakingly collecting. Encouraging students to run all their carefully scripted workloads at once helps to avoid this frustrating experience.

5.4 Statistics and System Effects

Make expectations about experimental practice and analysis explicit. There are only a few ways to run an experiment correctly — and an almost infinite number of ways to run it incorrectly. Instructors of courses involving performance measurement should provide all students — regardless of familiarity with hardware, operating systems, and statistics — with explicit requirements and guidance regarding expectations in these areas. Simple examples include requiring confidence intervals for all data and detailed workload documentation so instructors can validate experiments during grading. Discussing these issues critically could also be integrated into the course. Group meetings might include open dissection of the groups workload scripts Or, instructors might be able to "build in" time for "green teaming" — where one group deconstructs the energy testing methodology of another. In any case, some intense scrutiny — before final experiments are run — is a good idea.

6 Related Work and Alternatives

The Atom LEAP directly measures energy consumption at the component level. While it incorporates several novel features, it is not the only example of component-level current sensing and sampling used for energy measurement[5]. A recent example is the workbench that Microsoft used to compare the power consumption of popular web browsers[10]. Older examples in the literature are often limited to measuring a single component at a time with a digital multimeter. The Atom LEAP's major contribution to this area is the use of TSC-based "energy calipers" to measure discrete code segments, which can be used in both kernel and user code.

There are other approaches for power measurement and modeling. The simplest is to use a socket power measurement device, such as the Kill-A-Watt power socket (which has no computer interface), or the ACPI battery interface on laptops [14]. The ACPI battery interface allows virtually any laptop the ability to modulate behavior based on some knowledge of local battery levels, and a number of influential papers have used this information source. Not only does this result in a single value for the entire system, but the accuracy and temporal granularity of these devices is known to be poor.

Mathematical models and simulators are also used as proxies for empirical measurement. These methods must first identify the energy consumption of particular operations of interest (such as a disk read or writing to a register) through painstaking experimentation or calculation. Architecture-based power simulators, such as Wattch[2], exist based on this low-level approach. Counting higher-level events with hardware performance counters (HPCs) is another approach[1].

However, these are models of specific hardware configurations, and it is non-trivial to expand them. The number of events that can be simultaneously measured is limited due to the small number of HPCs in a given CPU. This limitation requires running repetitions of the same workload while changing the event types counted to estimate values for other devices. Not only is this a painstaking process, but it is not practical for uncontrolled, real-world environments.

Atom LEAP (and current sensing in general) is not limited like these previous systems. While measurements taken with an individual Atom LEAP are dependent on that configuration, the only work necessary to measure new hardware is updating the voltage of the device and the resistance of the sensing leads in the configuration file (if either has changed). This allows, among other things, easy reconfiguration of the hardware without affecting the accuracy of the results.

7 Future Work

It is our sincere hope that LEAP technology will find use beyond our labs, so we are in the process of preparing our work for a broader release. Eventually, we will release documentation for building and using LEAP technology, along with our software stack. We hope to develop an active and collaborative LEAP community and website hosting the Atom LEAP materials.

A new contribution we hope to make to this community is a step-by-step, pedagogical curriculum covering energy efficiency and performance measurement. As an open-ended, upper-level research class, CS 188 was a great success. We also think a set of directed materials for novices would be a boon to the community.

Another task to be performed before we can broadly release the Atom LEAP is to choose hardware for the flagship Atom LEAP. The current Atom motherboard[7] has certain limitations, such as a lack of DVFS, a single RAM slot, and a power-hungry northbridge[3]. These limitations make the current hardware less than ideal. Accordingly, we are investigating other boards in the hopes that we can find an appropriate and maximally flexible set of hardware for the LEAP community to use.

But above all, we must quantify the performance and power effects of the measurement and instrumentation. The current Atom LEAP hosts its own DAQ — in other words, it measures itself. While this is extremely convenient and adds to the appeal of the Atom LEAP, it raises accuracy concerns. However, we believe this overhead is basically static and fairly low-impact. We are actively working to quantify, reduce and, above all, document the energy and performance costs of the current system so that the tool can be confidently used for high-quality research. Nonetheless, we will include instructions for hosting the LEAP on a different device, such as an attached laptop or single-board computer (SBC) such as a Gumstix device[8].

A related accuracy concern is the variance in energy use between ostensibly "identical" LEAPs. We have several units at UCLA, built from the same components. We can use these devices to measure how much the power consumption varies across the set, so that we can identify (and minimize) the sources of this variance. This will help us to provide the LEAP community with a means of comparing results more effectively.

Finally, we are looking forward to building the next generation of LEAP systems. This winter we also used the Atom LEAP in Electrical Engineering 202C at UCLA, a graduate-level course in embedded network design. In that course, students focused on expanding and improving the Atom LEAP platform in new directions. Projects included a battery-powered Mobile LEAP Testbed, which was used to evaluate the potential power savings of opportunistically choosing between 3G and WiFi when available, and a DSP LEAP, using LEAP technology to measure the energy use of a Texas Instruments digital signal processor.

8 Conclusion

Our collection of five Atom LEAPs served as a great mini-testbed for our course investigating trade-offs between security functionality and energy consumption, and the course itself served as an excellent opportunity to teach careful systems performance analysis to undergraduates. The Atom LEAP's simple, flexible, and powerful design allowed students to create high-resolution, component-level energy measurements of both user and kernel code. This yielded a number of interesting results, including the demonstration of some counter-intuitive energy saving techniques. The Atom LEAP's hardware, software, and documentation is under heavy development. In the future, we plan to make this powerful research and education tool available to the community.

References

[1] Bellosa, F. The benefits of event: driven energy accounting in power-sensitive systems. In *Proc. of the 9th workshop on ACM SIGOPS European workshop* (2000).

[2] Brooks, D., Tiwari, V., and Martonosi, M. Wattch: A Framework for Architectural-Level Power Analysis and Optimizations. In *In Proceedings of the 27th Annual International Symposium on Computer Architecture* (2000).

[3] Dawson-Haggerty, S., Krioukov, A., and Culler, D. E. Power Optimization – a reality check. Tech. Rep. UCB/EECS-2009-140, 2009.

[4] Dike, J. In *Proc. 2000 Linux Showcase on Conference* (2000).

[5] Flinn, J., and Satyanarayanan, M. Powerscope: A tool for profiling the energy usage of mobile applications, 1999.

[6] Gurumurthi, S., Sivasubramaniam, A., Vijaykrishnan, M. J., Vijaykrishnan, I. N., Kandemir, M., Li, T., and John, L. K. Using complete machine simulation for software power estimation: The softwatt approach. Tech. rep., 2001.

[7] Intel Corp. The Intel Desktop Board D945GCLF2 Product Guide. `http://downloadmirror.intel.com/16960/eng/D945GCLF2_ProductGuide02_English.pdf`, 2010.

[8] Matarix, M., Koenig, N., and Feil-Seifer, D. Materials for enabling hands-on robotics and STEM education.

[9] Mclntire, D., Ho, K., Yip, B., Singh, A., Wu, W., and Kaiser, W. The low power energy aware processing (LEAP) embedded networked sensor system. In *The Fifth International Conference on Information Processing in Sensor Networks* (2005).

[10] Microsoft Corporation. Browser Power ConsumptionLeading the Industry with Internet Explorer 9. `http://blogs.msdn.com/b/ie/archive/2011/03/28/browser-power-consumption-leading-the-industry-with-in aspx`, 2011.

[11] Peterson, P., and Reiher, P. zPad Power Aware Security Devteam. `http://lasr.cs.ucla.edu/~pedro/docs/zpad-design.pdf`, 2011.

[12] Ryffel, S. *LEA2P - The Linux Energy Attribution and Accounting Platform.* PhD thesis, UCLA/ETH, 2009.

[13] Singh, D., Peterson, P. A. H., Reiher, P. L., and Kaiser, W. J. The Atom LEAP Platform For Energy-Efficient Embedded Computing: Architecture, Operation, and System Implementation. `http://lasr.cs.ucla.edu/~pedro/docs/leapwhitepaper.pdf`, 2010.

[14] Zeng, H., Ellis, C. S., Lebeck, A. R., and Vahdat, A. ECOSystem: Managing Energy as a First Class Operating System Resource.

Notes

[1] http://lwn.net/Articles/342892/
[2] http://squashfs.sourceforge.net/
[3] http://www.virtualbox.org/