ASEE AMERICAN SOCIETY FOR ENGINEERING EDUCATION

Paper ID #35038

Introducing Open Source Hardware in Computer Engineering courses

Mr. Sharan Kalwani, IEEE

Sharan Kalwani is an HPC architect well versed in using deploying & managing simulation applications in several industries: automotive, design engineering, IT, bioinformatics, industrial & university research, academic computing, machine learning and the data sciences.

Dr. Subramaniam Ganesan, Oakland University

Dr. Subramaniam Ganesan, is a Professor in the department of Electrical and Computer Engineering, Oakland University, Rochester, MI 48309, USA. He has over 30 years of teaching and research experience in Digital Computer systems. He was the chair of the CSE department from1991 to 98. He has published over 100 journal papers, more than 200 papers in conference proceedings, and 3 books. He published a book on Java in 2003. He developed a custom DSP board with software for his DSP book. He is a senior member of IEEE, IEEE Computer Society Distinguished Visiting Speaker, IEEE Region 4 technical activities member and Fellow of ISPE. He received Life time Achievement award from ISAM, Lloyd L. Withrow Distinguished Speaker award from SAE, Best Teacher award from ASEE, and Oakland University. He has organized many international conferences. He is the editor in chief of an International Journal of Embedded system and Computer Engineering and International journal of Sensors and applications. He is the session organizer on ”Systems engineering” at SAE world congress for the past 15 years. His research interests are in Real time system, parallel architectures and computer systems, Automotive embedded systems security and signal processing.

Introducing Open Source Hardware in Computer Engineering courses

Abstract

In this paper we describe the ways to introduce into computer engineering courses the topic of open source hardware. The term open source refers to hardware objects which the people can modify and share since its design is publicly accessible. We will describe a few open source hardware products and identify in which courses they can be used. Senior design courses in Computer Engineering are also ideal to utilize Open source hardware and we will show some examples. We describe the design course project selection, hardware selection technical challenges, design, testing, challenges in using Open source hardware and documentation. The Open source RISC-V processors can be used in Computer Organization courses. RISC-V are open source processors which are freely available as soft core and they can be used in FPGA based Embedded Systems courses. The microprocessor board designs can be taught using the Open source hardware such as Arduino. An Embedded systems course outline is given in appendix showing the use of partially Open source hardware. This paper concludes with the advantages of using Open source hardware in computer engineering courses.

Introduction - what is Open Source?

The term open source originated in the software development domain to show a new approach to writing and implementing computer programs. Today, however, "open source" designates a new way - that we call "the open source philosophy". Open source projects, products, or initiatives embrace and follow principles of open exchange, collaborative work, joint development participation, rapid prototyping, transparency, software or hardware community-oriented development and volunteer driven help in learning, sharing tips and solutions.

Open source software is software with source code that anyone can view, download, modify, improve and use. Its authors make its source code available to others who would like to view that code, copy it, learn from it, alter it, or share it. LibreOffice, the GNU Image Manipulation Program (GIMP) and most famous of all – Linux, are all examples of open source software. Just like Open source software (OSS), Open source hardware uses licenses. A majority of these licenses are based on existing OSS licenses. Some of the widely used licenses for Open source hardware include the TAPR Open Hardware License, Balloon Open Hardware License and the Hardware Design Public License [1].

There are a number of Open source hardware design available and many are not known to students. Hardware design of microprocessor boards like Arduino, Beagle board, Amber, OpenPower, OpenSPARC, ZPU (targeting FPGA) and RISC-V designs are examples [2]. Open source hardware is mostly developed in a collaborative way. Many users make improvements, test and post them for the public.

Free and open source hardware (OSH) or FOSH, means that the hardware design (electronic schematics, Mechanical details, PCB design details, layout, HDL source code) are made freely available. When improvements are made by the user of Open source hardware, they are made freely available for everyone. Of course, there is no return on this investment, but knowledge is shared. It is not a simple open document, but includes all the codes necessary to communicate with the hardware and various devices. The licensing model is often based on the GNU Public License, which may often conflict with patent law, but the benefits of an Open source hardware approach outweigh those of restrictive patent law. There are partially Open source and fully Open source hardware.

Microprocessor System design course

A typical microprocessor system design course introduces the architecture of the processor, the instruction architecture (ISA), assembly language, C programming concepts, interfacing peripherals using serial and parallel ports, use of timer and other special functional units in the processor. Open source hardware like Arduino board (shown in Figure 1) is an ideal low-cost board to use in this course. A number of projects that can be done using Arduino type boards are plentiful and can be easily found on the internet [3]. Many manufacturers are making boards compatible to Arduino Open source hardware, for example: NXP FRDM KL25z board as shown in Figure 2. The KL25Z has an on-board accelerometer, which is a sensor that measures the amount of acceleration applied to it. The Xtrinsic MEMS sensors board features three types of Xtrinsic sensors from NXP, including MPL3115A2, MAG3110 and MMA8491Q. Included in the kit are the software drivers and code that enables users to easily evaluate and demonstrate the performance of the sensors to measure altitude, pressure, as well as detection of magnetic fields and physical position [4]. Advantage of using Open source Arduino board compatible boards is low cost and availability of many sensors on these boards

These boards have many add on interface boards for connecting to peripherals like (multiple channel ADC). A CAN-bus (Controller Area Network) port (Shield) can be added to Arduino compatible boards [5] including the KL25z using the shield shown in Figure 3. There are multiple sites which share codes for interface (for example, github.com). A typical code for CAN communication using the KL25z board is given in [6]. Students can collaborate in the software development using open source boards. This introduces them to use of open source hardware and software and teaches them how to improve them for specific applications. Since most of the hardware and software information is available on the web, the system design is faster and testing is easier. They also learn to collaborate with senior engineers through the websites and user groups. The mentorship provided and suggestions provided by these mentors are very useful to students for the present course and also for their life time career.

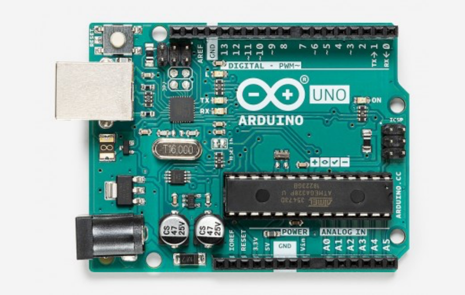


Figure 1 Arduino Uno board - an OSH board

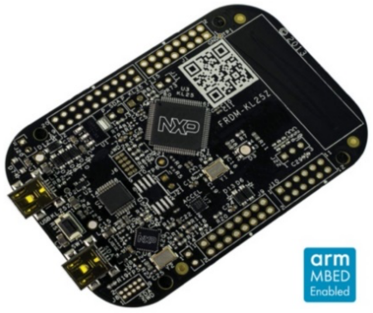


Figure 2 FRDM-KL25Z development board by NXP. This is an open source hardware board with Arduino compatibility

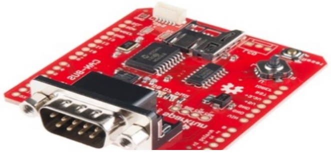


Figure 3 CAN bus (Shield) for Arduino type boards

OSH example boards and RISC-V processor

RISC is an acronym for reduced instruction set computer and the instructions are efficiently implemented inside the processor to execute faster. RISC-V is an open source hardware instruction set architecture (ISA) based on established reduced instruction set computer (RISC) principles. The project began in 2010 at the University of California, Berkeley, but many contributors are volunteers who are not affiliated with the university. FII RISC-V FPGA boards are ready-to-use development platform designed around the Field Programmable Gate Array (FPGA) from Xilinx, Altera/Intel and others. They are designed to cover all aspects of FPGA Development and Experiment. There is also the RISC-V SOC (System On a Chip). The main application areas are smart home, wearable, sensor fusion, IOT, industrial control and many others.

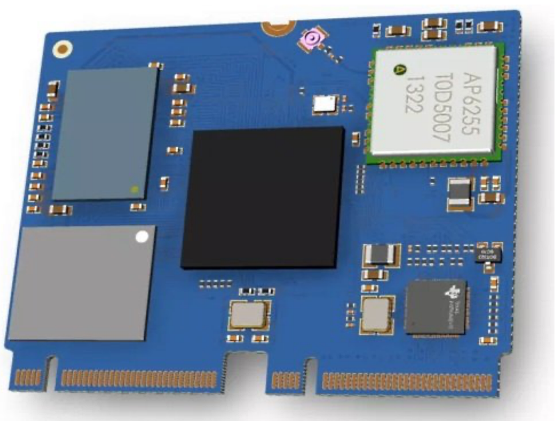


Figure 4 Single board computer using RISC-V

OSH hardware is low-cost devices as shown in Figure 4 which is a Linux running single board computer using RISC-V, which costs less than $13 [7].

Intel/Altera and Xilinx have FPGA boards with a softcore RISC-V processor. A hardcore FPGA is called SOC – FPGA (system on a chip FPGA). There are tutorials on how to build RISC-V on an FPGA board [8]. If we implement using software, then it is called softcore processor. If these processors are already implemented inside the FPGA by the manufacturer, then it is called SOC-FPGA. By integrating hard-core RISC-V CPUs with its latest FPGAs, Microsemi (a Microchip company), has further bolstered its support of RISC-V [9]. This is the same approach that Intel/Altera and Xilinx have done with Arm cores and their system-on-chip SOC FPGA.

Computer Architecture courses

RISC-V can be used as a learning tool in Computer Architecture course. It is a free and open Instruction Set Architecture. It has many industry backers like Google, Qualcomm, IBM, Samsung, Huawei and many others. It has both 32-bit and 64-bit flavors. Instruction sets are fully defined and described in the open documents. Open sourcing allows everyone to verify the cleanliness of an implementation. Moreover, RISC-V was designed to be highly scalable. It can run on a low-powered 32-bit embedded systems with 16 registers to a high performance 128-bit super computer. But more importantly, RISC-V was designed based on failures and lessons learned in the past. Designing an ISA isn't easy like many people thought. It's a highly complex task that needs expertise in a broad range of fields like electronics, Operating Systems, compiler and application software [10].

The Computer Architecture courses will teach how the RISC-V was designed to address efficient CPU implementation. The current RISC-V has 32 numbers of 32 bit registers and supports three types of instructions: register to register; memory to register; control flow. It supports math operations like multiply, add, right and left shift, logical and other arithmetic operations. Teaching these concepts are easier in the Computer Architecture courses. The RISC-V privilege level supports paged virtual memory with a 32-bit address space divided into 4KB pages. A 32-bit virtual address is separated into a 20-bit virtual page number and a 12-bit page offset. Two additional virtual memory configurations are defined for the RISC-V 64-bit environment. These concepts can be easily taught using the RISC-V Open source processor [Figure 5].

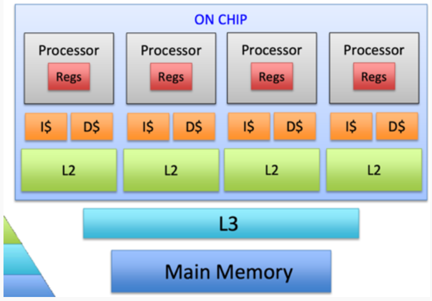
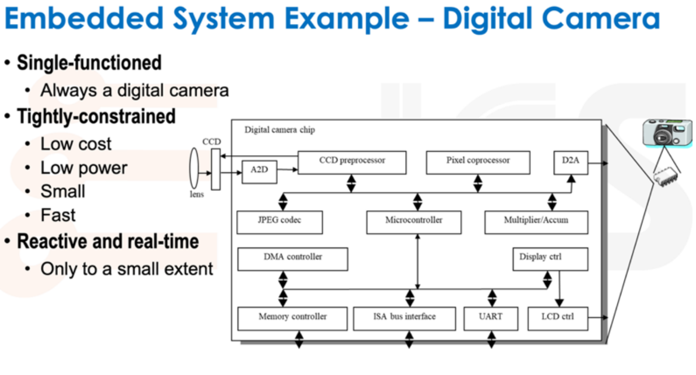


Figure 5: RISC V Memory Hierarchy with multiple level caches.

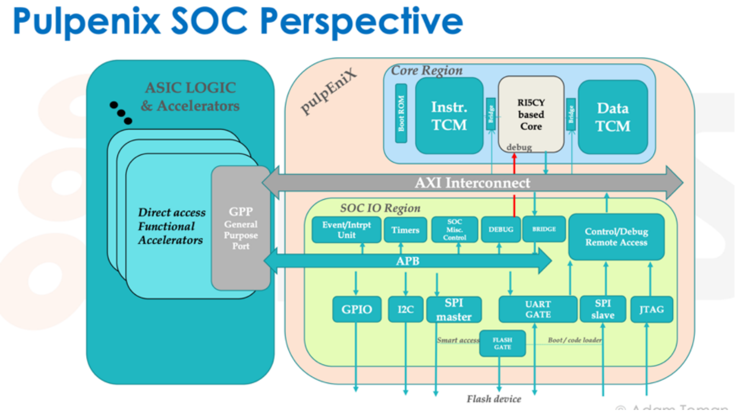
Memory Hierarchy parameters, size of Caches at different level and Cache coherency behavior can be studied with the above design, simulation and testing. The design is implemented on FPGA and tested. These are important concepts to be taught in the Computer Architecture course [11-12].



Embedded System Example – Digital Camera

- **Single-functioned**
  - Always a digital camera
- **Tightly-constrained**
  - Low cost
  - Low power
  - Small
  - Fast
- **Reactive and real-time**
  - Only to a small extent

Figure 5 Digital camera interface to a microcontroller board



Pulpenix SOC Perspective

Figure 6 System on a Chip block diagram with a Microcontroller or RISC-V

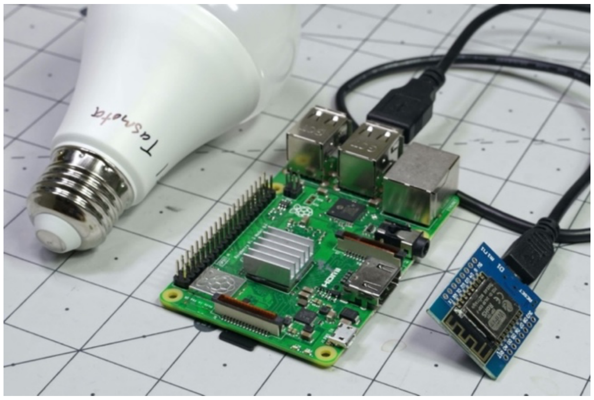


Figure 7 Home automation using Raspberry Pi

Integrate Open source hardware in senior design course projects

Undergraduate students do a senior design project before graduation. Figures 6, 7 and 8 show example senior design projects. Designing and developing projects like Home automation with a Raspberry Pi, digital camera interface to a SoC device and many more are possible with open-source hardware.

The students like the low-cost of the Open source hardware. They are happy to see the free device driver software and mentorship available on the web for Open source hardware. Support needed for the design projects are Printed circuit making tools, CAD tools, simulation and testing tools, oscilloscope, variable power supply and pulse generator. The complete circuit design of the Open source boards is available in the web. Instructor presents and reviews the Open source hardware and also the advantages of the Open source hardware. Students present their project selection, description and goal. The instructor and the students together discuss the project goals and options for the hardware and software for the projects. Instructor presents the supplements with necessary suggestions to improve the boards for the specific project. For example, the Open source hardware Arduino boards do not have automotive ethernet interface or Flex ray port or interface for driving a high torque motor. Sometimes the interface cards are available for purchase. Otherwise, students do the interface design and test it. The students do not face any technical challenges with the Open source hardware.

Students and faculty do an iterative, creative, decision-making process to come up with the Open source-hardware based solution. They develop requirements, do analysis and look at multiple Open source solutions, evaluate solutions against requirements, consider the risks, and make trade-offs to obtain a correct solution. They also consider the legal considerations, maintainability, manufacturability, marketability, and standards during the project. Student teams keep a log of initial concept, requirement/ goal for the project, system design, simulation, iterative redesign and simulation to meet the goals, and implementation details on hardware with software. The whole process, design tradeoffs and history are kept in the final project document folder. In addition to intellectual property /commercialization of the final product, the integration of all the concepts learnt in the curriculum is considered throughout the capstone design course. The projects are evaluated by a number of multidisciplinary faculty, fellow students and the instructors. The student outcomes and design criteria are considered in the evaluation. Later the design and software are made public as Open source. Full documentation for the interface is also submitted in the final project report.

Some of the recent projects are: Smart home with connected home appliances, CAN protocol-based sensor network, and driver assist to notify the presence of obstacles in fog conditions. These are low-cost systems with commercial potential. The algorithms developed with the hardware can be patented. A real time system with a unique algorithm to remove fog from the front road scene is developed. This is also a patentable idea.

Advantages and disadvantages of using Open source hardware and software

The Open source products are free. Open source boards are cheaper than commercially marketed products. They are created by skillful and talented people, highly reliable and there are user community for trouble shooting. There are some minor disadvantages such as absence of friendly and extensive support like a commercial product. One may have to modify the Open source hardware for a specific project and this not only introduces delay but also reliability problem.

Conclusion

The term Open source refers to hardware product which the people can modify and share since its design is publicly accessible. In this paper we described the ways to use in computer engineering courses the Open source hardware. We identified a few Open source hardware products that can be used in Computer Engineering courses. We showed also that senior capstone design course can utilize Open source hardware. Use of Open source hardware in the courses provide real-world experiences to students.

Appendix A

A few fully open-source processors/CPUs and microcontrollers are listed here.

> Freeduino – an open-source physical computing platform based on a simple I/O board and a development environment that implements the open source Processing / *Wiring* language. Also clones of this platform including Freeduino.Tinkerforge – a platform co. Amber is an ARM-compatible 32-bit RISC processor. Amber implements the ARMv2 instruction set. LEON, a 32-bit, SPARC-like CPU created by the European Space Agency. OpenPOWER, based on IBM's POWER8 and newer multicore processor designs. OpenSPARC, a series of open-source microprocessors based on the UltraSPARC T1 and UltraSPARC T2 multicore processor designs. Parallax P8X32A Propeller is a multicore microcontroller with an emphasis on general-purpose use. ZPU, a small, portable CPU core with a GCC toolchain. It is designed to be compiled targeting FPGA· Zet (hardware), x86 implementation on programmable logic. OpenRISC 1200, an implementation of the open source OpenRISC 1000 RISC architecture.

Appendix B

Course Title: **Microprocessor based systems design (4 credits)**
TEXTBOOK:
The RISC-V Reader: An Open Architecture Atlas, David Patterson & Andrew Waterman

Course Description:
This course deals with advanced concepts in the programming and the interfacing of modern microprocessors/microcontrollers to the outside world as demonstrated by a variety of application examples. It covers the advanced architecture of modern processors and the many I/O peripherals now commonly found on-board. Laboratory activities provide the student with experience in developing the hardware and software required to incorporate microprocessors into systems that solve real-world interfacing problems.

Course Objectives:

- Describe the architecture of modern microprocessors and microcontrollers
- Develop embedded software in C given a set of design requirements and constrains
- Interface to various analog and digital I/O devices
- Learn about embedded operating system to develop multitasking systems
- Learn about CAN-based communication systems

Work in a team environment to design a microprocessor-based system and communicate the results in a written report and/or an oral presentation

Course details:

1. **Introduction**
   Need of advanced microprocessors, Difference between RISC (Reduced Instruction Set Computing) and CISC (Complex Instruction Set Computing), RISC Design philosophy, ARM (Acorn/Advanced RISC Machine) Design Philosophy, History of RISC-V microprocessor, RISC-V processor family, Development of RISC-V architecture. Open source hardware and software concepts.

2. **The RISC-V Architecture and Programmers Model**
   The RISC Machine, RISC-V Core data flow model, Architectural inheritance, The RISC-V programmer's model: General purpose registers, RISC-V memory map, data format, load and store architecture, Architecture revisions, RISC-V development tools

3. **RISC-V Instruction set**
   Data processing instructions, Arithmetic and logical instructions, Rotate and barrel shifter, Branch instructions, Load and store instructions, Software interrupt instructions, Program status register instructions, Conditional execution, Multiple register load and store instructions, Stack instructions, Thumb instruction set, advantage of thumb instructions, Assembler rules and directives, Assembly language programs for shifting of data, factorial calculation, swapping register contents, moving values between integer and floating point registers.

4. **C Programming for RISC-V**
   Overview of C compiler and optimization, Basic C data types, C Looping structures, Register allocations, function calls, pointer aliasing, structure arrangement, bit-fields, unaligned data and Endianness, Division, floating point, Inline functions and inline assembly, Portability issues. General purpose I/O, general purpose timer, PWM Modulator, UART, I2C Interface, SPI Interface, ADC, DAC. CAN Interface.

5. **Learn about basics of RTOS.** Use of FreeRTOS on Arduino boards**.**

6. **Advanced Microprocessor Bus Architecture (AMBA) Bus System**
   User peripherals, Exception handling in RISC-V, RISC-V optimization techniques (speed, power, energy, memory requirements).

References

1. Open hardware License, https://tapr.org/the-tapr-open-hardware-license/
2. Arduino Projects, https://create.arduino.cc/projecthub
3. The Arduino Uno board is a good board to get started, https://store.arduino.cc/usa/arduino-uno-rev3, *retrieved December 20, 2020*
4. Manufacturer: NXP, Mfg Part No: XTRINSIC-SENSORS-EVK, Price: $18.25, https://www.element14.com/community/docs/DOC-54360/l/mems-sensors-evaluation-kit-with-nxp-frdm-kl25z-freedom-platform, *retrieved December 20, 2020*
5. CAN bus Shield that can be used with Arduino compatible boards to add CAN ports, https://www.sparkfun.com/products/13262, *retrieved December 20, 2020*
6. https://github.com/John579/FRDM_KL25Z_CAN/tree/b5a61c8b8ce18ccf7dc6cfd80fb6c9ead307cc03, *retrieved December 20, 2020*
7. https://blog.adafruit.com/2020/11/13/the-east-west-trade-split-draws-china-to-risc-v-riscv-aallan-hacksterio/, *retrieved December 20, 2020*
8. https://www.symmathics.com/index.php/building-risc-v-tutorial-page/, *retrieved December 20, 2020*
9. https://www.microsemi.com/product-directory/fpga-soc/5210-mi-v-embedded-ecosystem, *retrieved December 20, 2020*
10. https://www.quora.com/Why-is-RISC-V-having-so-much-impact-if-there-are-already-open-architectures-such-as-SPARC, *retrieved December 20, 2020*
11. https://www.symmathics.com/index.php/building-risc-v-tutorial-page/, retrieved *December 20, 2020*
12. http://www.eng.biu.ac.il/temanad/files/2019/11/RISC-V-Intro-for-Hackathon.pdf, *retrieved December 20, 2020*