J Inf Process Syst, Vol.15, No.6, pp.1406~1421, December 2019
https://doi.org/10.3745/JIPS.03.0129

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Selecting a Synthesizable RISC-V Processor Core for Low-cost Hardware Devices

Dennis Agyemanh Nana Gookyi* and Kwangki Ryoo*

Abstract

The Internet-of-Things (IoT) has been deployed in almost every facet of our day to day activities. This is made possible because sensing and data collection devices have been given computing and communication capabilities. The devices implement System-on-Chips (SoCs) that incorporate a lot of functionalities, yet they are severely constrained in terms of memory capacitance, hardware area, and power consumption. With the increase in the functionalities of sensing devices, there is a need for low-cost synthesizable processors to handle control, interfacing, and error processing. The first step in selecting a synthesizable processor core for low-cost devices is to examine the hardware resource utilization to make sure that it fulfills the requirements of the device. This paper gives an analysis of the hardware resource usage of ten synthesizable processors that implement the Reduced Instruction Set Computer Five (RISC-V) Instruction Set Architecture (ISA). All the ten processors are synthesized using Vivado v2018.02. The maximum frequency, area, and power reports are extracted and a comparison is made to determine which processor is ideal for low-cost hardware devices.

Keywords

Hardware Resources, IoT, Low-Cost Hardware Devices, RISC-V, SoC, Synthesizable Processors

1. Introduction

The Internet-of-Things (IoT) has become a powerful and ubiquitous platform for the interconnection of devices used in our everyday life. The lowest abstraction of IoT platforms consists of low-cost hardware devices for data sensing and collection [1]. IoT has revolutionized sensors and data collection devices. The devices used to be stand-alone and needed a manual data logger system where a person record reading taken by the sensors. With the advent of IoT, these devices are given computing and communication capabilities to read, process, and transmit data through wireless channels [2]. To perform the function of reading, processing, and transmitting real-time data, low-cost hardware devices have to integrate several components ranging from sensors, radio frequency (RF) communication modules, and security modules for secured data storage and transmission. To achieve these functionalities, a synthesizable processor [3] is vital to coordinate and control the flow of algorithms execution, error processing, and easy integration of new components. Choosing a synthesizable processor core for low-cost devices is a challenge because of the low hardware area of the devices [4].

Manuscript received December 28, 2018; first revision May 9, 2019; second revision August 19, 2019; accepted October 14, 2019.
Corresponding Author: Kwangki Ryoo (kkryoo@gmail.com)
* Dept. of Information and Communication Engineering, Hanbat National University, Daejeon, Korea (dennisgookyi@gmail.com, kkryoo@gmail.com)

This paper examines ten synthesizable processors that implement the RISC-V ISA [5]. The processors include Roa Logic RV12 [6], ORCA [7], SiFive E31 [8], SCR1 [9], Rocket [10], BOOM [11], MRISCV [12], PICORV32 [13], Shakti-E [14], and Hummingbird [15]. The register transfer level (RTL) codes of the processors are downloaded from their respective repository. The most important step in selecting a processor core for low-cost hardware devices is to evaluate its hardware resource consumption. This is done to make sure that the processor meets the resource limitation of the hardware device. For that matter, the synthesizable processors' codes are synthesized using Vivado v2018.02 [16] which is a software for synthesizing, simulating, and analyzing hardware description languages (HDLs). The maximum frequency, area, and maximum dynamic power are extracted from the synthesis reports. The reports are analyzed to identify the ideal processor core for a low-cost hardware device.

The main contributions of this work are as follows:

- The requirements/specifications of a low-cost hardware device for IoT applications are establish ed and an ideal field programmable gate array (FPGA) board is selected.
- The advantages of using open source synthesizable processors over standard processors are state d after which the ten processors are examined.
- The synthesis reports of the processors are examined and the ideal processor for low-cost hardw are devices is identified.
- The ideal RISC-V processor core is compared to other processor cores that implement different ISAs.

The remaining of this paper is as follows: Section 2 elaborates some related works, Section 3 states the basic requirements of low-cost hardware devices used in IoT platforms, Section 4 gives a summary of the ten synthesizable processors and the advantages of using open-source processors, Section 5 examines and discusses the synthesis reports of the synthesizable processors and identifies the ideal processor for low-cost hardware devices, Section 6 compares the ideal RISC-V processor core to other cores that implement different ISAs, the paper ends with the conclusion discussed in Section 7.

2. Related Work

Since the inception of the RISC-V ISA in 2010 at the University of California in Berkeley, there have been hundreds of processor cores that implement the ISA. There is however no comprehensive study that compares and contrasts the various RISC-V processors. This section, therefore, examines related works involving processor cores that implement other ISAs. Christensson and Mattsson [17] evaluated and discussed three synthesizable processors which included LEON2 (from Gaisler Research), MicroBlaze (from Xilinx), and OpenRISC 1200 (from opencores). Various processor benchmarks, as well as performance per clock cycle, were used to compare the processors. The paper concluded that the LEON2 processor gave the best performance. Westerlund [18] studied and compared two processor cores which include LEON2 and NIOS II (from Altera). The performance metrics were the same as that for [17]. The research also concluded that LEON2 showed better results as compared to NIOS II. Jia et al. [19] investigated a number of opensource processors which include LEON3, LEON2, OR1200, S1, AltOR32, LatticeMicro32, and Amber. The processors are implemented on FPGA devices and their results are compared and discussed. This work concluded that LEON3 gave the best results in terms of performance (more discussion on this work is done in Section 6).

There exist a number of issues with the related works in this section. Most of the processor cores examined in the works are not designed for low-cost IoT devices and for that matter an important metric like hardware area utilization is not considered. Most of the processors are for commercial purposes and need expensive licenses before they can be used. Some of the processors are delivered as black-boxes and for that matter, designers have no access to the internal workings of the cores. This work, however, examines ten RISC-V processors with their source code available at no expense. All the processors are geared toward low-cost devices so this work evaluates and selects the ideal core for IoT devices. This work is important because the most important step in selecting a processor core for low-cost hardware devices is to evaluate its hardware resource consumption to analyze the feasibility of the core fitting inside the hardware.

3. Requirements Low-cost Hardware Devices

Examining wireless communication devices like the CC2530 [20] and nRF9160 SiP [21] manufactured by Texas Instruments and Nordic Semiconductor, respectively, a general architecture of low-cost hardware devices for sensing, data storage, and secure data transmission consists of similar hardware components. These devices can be used in applications such as smart cities, smart agriculture, item tracking, medical, and smart wearables.

A low-cost hardware device for sensing, data storage, and transmission should consist mainly of RF Front End, a battery regulator, and System-on-Chip (SoC). The RF Front End has the responsibility of transmitting and receiving data to and from other devices, respectively. Some prominent technologies used at the RF Front End include low power solutions such as Bluetooth, Zigbee, and Cellular IoT [22]. Other technologies like long-term evolution (LTE) could be used but this is usually infeasible because of high power consumption. A power source usually made up of batteries with an average life of two years is used to power the devices. A technique like clock-gating is used to ensure that the device consumes less power when idle. The brain of the low-cost hardware device is the SoC. The SoC consist of a processor, memory, bus, peripherals, and other modules on a single chip. The processor is the heart of the SoC. It is responsible for error processing, interfacing, and scheduling of the different components available on the device. Security modules in SoC have become very important for the fact that low-cost devices store and transmit sensitive data that could be harmful in the wrong hands. The security modules include lightweight encryption/decryption algorithms [23], lightweight authentication algorithms [24], and lightweight key exchange algorithms [25]. This paper focuses on selecting an ideal processor for the low-cost hardware devices but it is important to first examine the platform device (IoT development board) in which the processors will be implemented.

3.1 IoT FPGA Development Board

The first and important step in deploying IoT based application is the selection of a platform that meets design goals such as performance, hardware area, and power consumption. These platforms could be interfacing sensors with microcontrollers, custom application specific integrated circuit (ASIC) or commercial-off-the-shelf (COTS) FPGAs. The microcontroller platform offers the advantage of being the fastest in terms of product time-to-market but poses significant problems when it comes to the design of

low power applications and the interfacing of different sensors. ASIC design offers the advantage of lower overall cost per device in mass production and also flexible device size, performance, and power consumption. The disadvantage of ASIC designs is that it takes a longer time-to-market and also not reconfigurable. Though FPGA designs are limited in terms of performance, device size, and energy consumption, it has recently become a desirable option for IoT based applications. This is because it has a faster time-to-market as opposed to ASIC designs, different techniques such as clock gating could be used for low power designs as opposed to microcontroller designs, and it is flexible for interfacing different types of sensors. Another major advantage of FPGA is its re-programmability which gives it unlimited flexibility.

A typical IoT application will be a face recognition system. This type of system identifies video frames and detects faces within the frames. The scalability of such a system is dependent on both the initial cost and the ongoing operational cost. A system with cameras that transfer video data to the cloud for computation will cost a lot in terms of the bandwidth used for uploading and computation in the cloud. On the other hand, a localized computation can reduce the cost of bandwidth. To design an IoT face recognition system, the low-cost criteria are directly linked with the kind of hardware used for integrating sensor networks. Xilinx and Altera are the two major companies that are focused on the manufacturing of FPGA chips. This work examines the Xilinx Parallella Board [26] because it is an ideal board for IoT based designs [27]. The Parallella board is equipped with the Zynq-7000 family of FPGA devices. The device is ideal for low-cost IoT designs because its available resources are equivalent to most IoT sensing and data collection devices. The device consists of 53200 Look-up Table (LUT) elements, 17500 LUTRAM, 106400 Flip-flops (FF), 140 Block RAMs (BRAMs), and 220 DSPs. For that matter, a low-cost application should implement algorithms that use less than 53200 LUTs, less than 17500 LUTRAM, less than 106400 FFs, less than 140 BRAMs, and less than 220 DSPs. Gao et al. [28] proposed an object detection framework that improved object identification performance in IoT devices. The framework was implemented on the Parallella FPGA board which costs less than $200. The experimental results indicated that the proposed system processed 1920×1080 image resolution in 1.7 frames per second which is 7.8 times higher than a standard object identification system.

4. RISC-V Synthesizable Processor Cores

In the development of IoT that sense, store, and transmit sensitive information, the hardware architecture should implement functionalities such as trusted intellectual property (IP) and secured communication. To achieve these functionalities, the best solution is to design the application using processor cores that implement open ISAs like RISC-V rather than the standard ARM or x86-based processors. An open ISA gives a designer the advantage of implementing an architecture that is best for a specific application. This makes the designer innovative in implementing designs with the lowest power consumption. An additional advantage of using open ISAs is that designers can obtain processor source codes from vendors. With the source code available, a deep inspection can be done to avoid the insertion of malicious codes by some vendors.

RISC-V is a relatively recent ISA that was primarily designed for academic purposes. It has quickly become an open-source ISA for both research and industry implementation. The main goal for designing

the RISC-V ISA was for it to be freely available for both academic (using small base integer ISA) and industry (using higher base integer ISA). The ISAs are named using the format of RV32/64/128IMAFDC. The naming convention is explained as follows: "RV" is an abbreviation for RISC-V, 32/64/128 is the size of the address space used by a particular ISA, "I" stands for Integer, "M" stands for Multiply/Divide, "A" stands for Atomic operations, "F" stands for single-precision floating-point, "D" stands for double-precision floating-point, and "C" stands for compressed ISA. The base ISA is the RV32/64/128I which is compulsory for any implementation. Outside the base ISA, other extensions such as "MAFDC" can be added depending on the type of application in which the designer wants to implement.

From the inception of the RISC-V ISA, hundreds of processor cores have been developed by individuals and corporations. Each processor core claims superiority over another and designed to meet a specific goal. Four main factors were considered when selecting the ten processor cores for this work which include the availability of source code, free license, stable developmental stage, and availability of processor documentation. The source code in the form of RTL design files should be available to the designer for easy inspection and modification to suit a specific application. The inspection of the source code is done to prevent the insertion of malicious codes that could leak information or degrade the performance of the system. One of the most important factors in choosing the processors was the availability of a free license. The license should give designers the ability to copy and modify the processor code. The developmental stage of the processor core should be stable. A stable processor core means that the source code is free of bugs and meets the functionality of the design specification. Lastly, the processor documentation should be available. This is necessary because the designer uses less effort to understand the processor when a detailed design document is available. These factors led to the selection of the ten processors for this work.

Here, a summary of the ten RISC-V processor cores (Roa Logic RV12, ORCA, SiFive E31, SCR1, Rocket, BOOM, MRISCV, PICORV32, Shakti-E, and Hummingbird) is given. The focus is primarily on only the processor cores. Other components attached to the processors such as memory, bus, and other peripherals are left out. A summary of all the processor cores is shown in Table 1.

- Roa Logic RV12 RISC-V Processor Core: Roa Logic RV12 RISC-V processor is developed and maintained by Roa Logic. RV12 can be configured to use RV32I or RV64I RISC-V ISAs which is suitable for embedded applications. It implements the Harvard architecture to enable access to instruction and data memories simultaneously. It is optimized to use a 6-stage pipeline with an optional data cache, instruction cache, branch prediction, a debug unit, and multiplier/division units.
- ORCA RISC-V Processor: ORCA implements a subset of the RISC-V ISA. It is designed to target FPGAs and can be programmed as an RV32I or RV32IM processor. It is designed and maintained by Vectorblox and was originally to serve as a host to Vectorbox's high-speed Matrix processor [29]. It can currently operate as a standalone processor core with instruction and data caches. It can be configured to use interconnections such as Intel's Avalon [30], WISHBONE [31] or Xilinx's Local Memory Bus [32].
- SiFive E31 RISC-V Processor Core: So far, SiFive E31 processor is the most deployed RISC-V core which is developed and maintained by SiFive. It takes full advantage of the RISC-V ISA which makes it suitable for IoT devices in terms of power and area. The E31 core supports Atomic, Multiply, and compressed RISC-V ISA (RV32IMAC). It is a 32-bit core which has a very

high-performance single-issue in-order execution pipeline. It achieves an execution rate of one instruction per clock cycle. The overall core consists of local interrupts, execution pipeline, data, and instruction memory units.

- SCR1 RISC-V Processor Core: SCR1 is an open-source processor designed and maintained by Synatocore. The ISA can be configured as RV32I with optional Multiply instructions. It has the Harvard architecture and a pipeline implementation that can be set to either two or four stages. It also consists of a high-speed multiply/divide unit. It is highly optimized for area and power efficiency and has an optional debug controller.
- Rocket RV12 RISC-V Processor Core: Rocket is an in-order five-stage pipeline processor core that implements RV32IMAFD and RV64IMAFD RISC-V ISA. It is developed and maintained by SiFive and the University of California Architecture Laboratory. The processor core consists of a memory management unit (MMU) that supports page-based virtual memory and non-blocking data cache access. Rocket is currently a library of processor core components. Almost all modules designed for Rocket are used by other processors and functional units. The Rocket processor is generated using the Rocket-chip SoC generator which instantiates an entire SoC design including the processor core, cache, bus, and other peripherals.
- BOOM RISC-V Processor Core: Berkeley Out-of-Order Machine (BOOM) was heavily influenced by the design of the Alpha 21264 processor [33]. BOOM is developed and maintained by Esperanto and the University of California Architecture Laboratory. It implements the RV64IMAFD RISC-V ISA. It has six pipeline stages. The BOOM processor core is also instantiated using the Rocket-chip SoC generator.
- MRISCV RISC-V Processor Core: MRISCV is a RISC-V processor core that is designed and maintained by OnChipUIS. It is designed to target RV32I ISA with optional Multiply instructions. MRISC core consists of an arithmetic logic unit (ALU), past and future memory units, interrupt unit, instruction decoder, registers, and Pico Co-Processor Interface for fast multiplications. The core is designed to be used for research and the development of IoT.
- PICORV32 RISC-V Processor Core: PICORV32 is a RISC-V processor core that implements the RV32IMC ISA. It is developed and maintained by Clifford Wolf. The processor is designed to serve as an auxiliary core in FPGA designs. It has a high frequency and can be integrated into designs without clocking issues. The registers can be configured as dual-port (provides better performance) or single port (smaller hardware area) register files. It has a selectable memory interface and optional interrupt and co-processor units.
- Shakti-E RISC-V Processor Core: The Shakti processor project is responsible for the development of a variety of processor cores based on the RISC-V ISA. This project is maintained by the Indian Institute of Technology (IIT) Madras. Shakti-E is a low area variant of the Shakti processors. It implements the RV32I RISC-V ISA. It is a three-stage in-order processor core aimed at low-frequency applications. It has an optional memory protection unit and consumes very little power.
- Hummingbird RISC-V Processor Core: Hummingbird RISC-V processor core is the first open-source processor core from China. It is developed and maintained by Bob Hu. It implements the RV32IMAC RISC-V ISA. It is a two-stage pipeline with a low area and power implementation. It has performance and area benchmarks that make it good for IoT ultra-low power applications. This processor is specifically designed to be used for research purposes.

Table 1. Summary of RISC-V processor cores

Processor core	ISA	Architecture	Data width (Bits)	Pipeline depth	Multiplier	Divider
Roa Logic	RV32I/RV64I	Harvard	32/64	6	Available	Available
ORCA	RV32I/RV64I	Harvard	32/64	5	Available	Available
SiFive E31	RV32IMAC	Harvard	32	5	Not available	Not available
Hummingbird	RV32IMAC	Von-Newman	32	2	Not available	Not available
SCR1	RV32I/RV32E	Harvard	32	4	Available	Available
Rocket	RV32IMAFD/ RV64IMAFD	Harvard	32/64	5	Available	Available
BOOM	RV64IMAFD	Harvard	64	6	Available	Available
MRISCV	RV32I	Von-Newman	32	No pipeline	Available	Not available
PICORV32	RV32IMC	Von-Newman	32	3	Available	Available
SHAKTI-E	RV32I	Harvard	32	3	Available	Available

5. Synthesis Results and Discussion

FPGA devices have become a preferable option when it comes to the implementation of digital systems. This is because it is low-cost, flexible, and most importantly has a short time-to-market. Current FPGAs have numerous resources that lead to the implementation of SoC designs that consume millions of hardware gates. Vivado v2018.02 is used to synthesize the RISC-V processor cores to obtain the hardware resources used for the implementation of each core. Vivado is a computer-aided design (CAD) software tool that automatically converts high-level description languages like HDLs into gate level. The tool generates area, power, and performance estimates that are important in implementing a design in the real world. These estimates are usually very fast and less expensive to produce, yet they accurately depict the real implementation of a design. It is important to note that processor benchmarking programs such as Dhrystone [34], Stanford [35], Paranoia [36], and EEMBC [37] are good for measuring and comparing the execution time of processors. However, in low-cost designs, low hardware area and low power cores are preferred over high-speed modules. Therefore, the first step in selecting a suitable synthesizable processor for low-cost devices is to measure the hardware resources used by the processors. It is important to understand the general RTL design flow up to the synthesis level [38]. The flow starts with the design or product requirement which is translated into a specification document. The RTL is then modeled to meet the specification using high-level description languages like Verilog or VHDL. The results are verified using testbenches also written using high-level languages. The design is then synthesized using a logic synthesizer. The function of the logic synthesizer is to translate the RTL code to logic gates and registers. The logic synthesizer takes the RTL code together with the technology library and design constraints and generates the gate-level netlist. The FPGA device used for the synthesis is the XC7Z020 device. This device is from the Zynq-7000 family of FPGAs with package clg400 and a speed grade of -2. The device is ideal for low-cost IoT designs because its available resources are equivalent to most IoT sensing and data collection devices. The device consists of 53200 LUT elements, 17500 LUTRAM, 106400 Flip-flops, 140 Block RAMs (BRAMs), and 220 DSP blocks.

Table 2 gives the compilation of all the results from synthesizing the ten processors using the XC7Z020 FPGA device. The results include the maximum attainable operational frequency (Freq), the total number of LUTs, the total number of FFs, the total number of BRAMs and DSPs. The percentages indicate resource utilization. The codes of the processor core are available in five high-level languages which include VHDL, Verilog, SystemVerilog (SystemV), Bluespec SystemVerilog (BSV), and Chisel [39]. VHDL, Verilog, and SystemVerilog are HDLs that can be synthesized directly by the Vivado software. BSV and Chisel are higher-level languages that are converted to Verilog before synthesis is done.

Table 2. Summary of synthesis results of RISC-V processor cores

Processor core	HDL	Freq (MHz)	LUT	LUTRAM	FF	Block RAM	DSP	Power (W)
Roa Logic	SystemV	142.8	3615 (6.8)	96 (0.55)	2246 (2.11)	0.5 (0.36)	0 (0)	0.037
ORCA	VHDL	185.2	1354 (2.55)	0 (0)	746 (0.7)	1 (0.71)	0 (0)	0.069
SiFive E31	Verilog	100	3614 (6.79)	48 (0.28)	1642 (1.54)	0 (0)	4 (1.82)	0.040
Hummingbird	Verilog	41.7	4153 (7.8)	0 (0)	2490 (2.34)	0 (0)	0 (0)	0.049
SCR1	SystemV	40	4337 (8.05)	1 (0.01)	2143 (2.01)	0 (0)	4 (1.82)	0.041
Rocket	Chisel	90.9	5073 (9.54)	88 (0.51)	2008 (1.89)	0 (0)	4 (1.82)	0.092
BOOM	Chisel	34.5	49865 (93.7)	947 (5.44)	25205 (23.7)	6 (4.29)	40 (18.2)	0.188
MRISCV	Verilog	111.1	1893 (3.56)	0 (0)	923 (0.87)	1 (0.71)	0 (0)	0.032
PICORV32	Verilog	200	904 (1.7)	24 (0.14)	566 (0.53)	0 (0)	0 (0)	0.013
SHAKTI-E	BSV	12.7	7948 (14.9)	64 (0.37)	1813 (1.7)	0 (0)	4 (1.82)	0.100

The values in brackets are presented as percentages (%).

5.1 Maximum Frequency

The maximum frequency of a digital design is a very important parameter that determines the behavior of the circuit. Xilinx Integrated Synthesis Environment (ISE) which has been discontinued since 2012 in favor of Vivado gave an estimation of the maximum frequency after the synthesis of a design. Vivado does not specifically state the maximum frequency after synthesis. The designer usually provides a timing constraint file and estimates the frequency the design should operate. The designer then checks the timing report to spot if there have been timing violations such as setup time and hold time violations. If violations occur, the frequency in the timing constraint file is reduced until the setup time and hold time are not violated. This is the approach taken by this work in finding the maximum frequency of each RISC-V processor core.

A design with high maximum frequency is ideal because it can be implemented in both low frequency and high-frequency designs with no timing issues. When a design with low frequency is implemented in the high-frequency application, there are clocking issues and the design usually gives undesirable results. Fig. 1 gives a graphical representation of the maximum frequencies of each processor core. From the figure, Rocket, Hummingbird, SCR1, BOOM, and Shakti-E recorded frequency that is below 100 MHz with Shakti-E recording as low as 13 MHz. SiFive E31, MRISCV, Roa Logic RV12, ORCA, and PICORV32 all recorded frequencies above 100 MHz with PICORV32 recording the highest frequency at 200 MHz. In terms of maximum frequency, PICORV32 outperforms the other processor cores and can be ideal for both low frequency and high-frequency IoT applications.

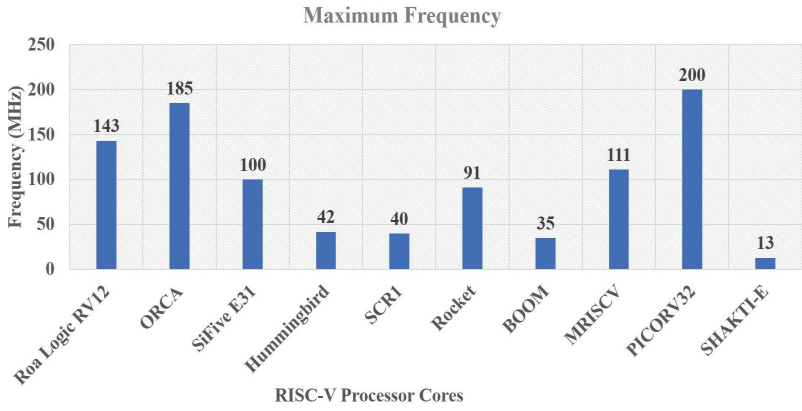


Fig. 1. Maximum frequency comparisons.

5.2 LUT Utilization

LUTs together with flip-flops and adders make up the fabric of a configurable logic block (CLB). CLBs gives FPGA its flexibility in terms of re-programmability. The XC7Z020 consists of eight LUTs per CLB that is used for the implementation of random logic. The internal matrix of a LUT is just like that of Static RAM (RAM) which is made of transistors. The LUTs in XC7Z020 can be configured as either a 6-input to 1-output or a two 5-input with separate outputs. The outputs of LUTs are usually registered by attaching a flip-flop. XC7Z020 has a maximum limit of 53200 LUTs for the implementation of a design.

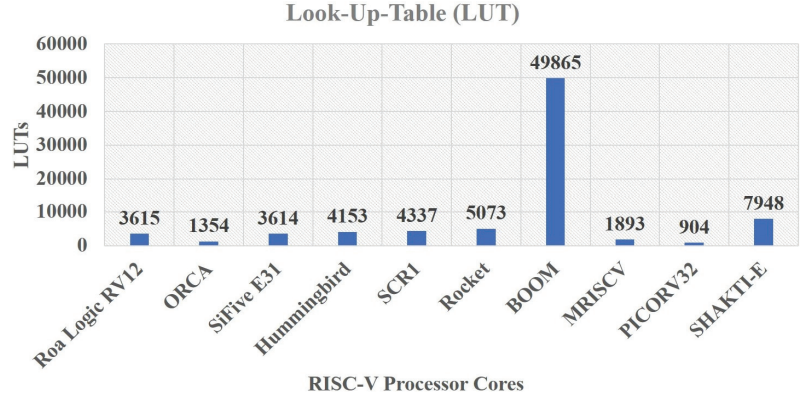


Fig. 2. LUT utilization comparisons.

The graphical representation of the utilization of the LUTs after the synthesis of each processor can be seen in Fig. 2. From the figure, Rocket, Shakti-E, and BOOM used more than 5000 LUTs with BOOM recording the highest LUTs at 49865. This represents 93.7% of all the available LUTs in the device. For IoT applications that require different functionalities, implementing BOOM would leave fewer LUTs for implementing other functions. SCR1, Hummingbird, Roa Logic RV12, SiFive E31, MRISCV, ORCA, and PICORV32 all used less than 5000 LUTs which is appropriate for low-cost IoT devices. PICORV32 used only 904 LUTs accounting for 1.7% of the total available LUTs in XC7Z020.

5.3 Flip-flops Utilization

Flip-flops are the basic building blocks of a synchronous circuit. Flip-flops output only changes at the edge of a clock signal. The data (D) flip-flop is the most common and frequently used. The XC7Z020 has a total of 106400 flip-flops. Fig. 3 gives a graphical representation of the usage of flip-flops by the RISC-V processor cores. From the figure, BOOM used 25205 flip-flops which accounted for 26.69% of the available flip-flops. All the other cores used less than 3000 flip-flops with PICORV32 using only 566 which makes it ideal for low-cost devices.

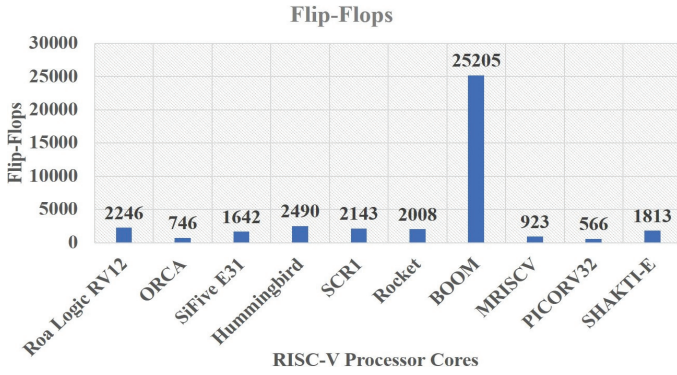


Fig.3. Flip-flop utilization comparisons.

5.4 LUTRAM Utilization

A LUTRAM is simply the use of LUTs to implement memory. In XC7Z020 devices, 25%–50% of LUTs are used to implement memory. The device consists of a total of 17400 LUTRAMs. Memory LUTs can be configured as either 64×1 RAM or 32×2 RAM. In XC7Z020, the criteria for using LUTRAM is that the memory should be less than 512 bits. An advantage LUTRAM has over BRAM is that the latter has a latency of one clock cycle while the former has no latency. A graphical representation of the utilization of LUTRAMs can be seen in Fig. 4. From the figure, BOOM used the most LUTRAMs (947) which represents 5.44% of the total available in the device. ORCA, Hummingbird, and MRISCV do not use LUTRAMs. Here all devices used less than 6% of the available LUTRAMs which is good for low-cost IoT end devices.

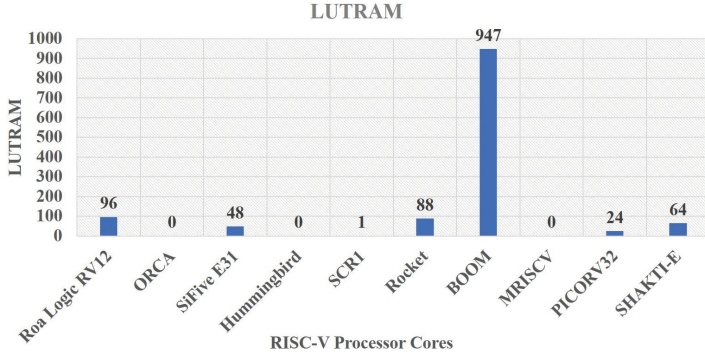


Fig. 4. LUTRAM utilization comparisons.

5.5 BRAMs and DSPs Utilization

The requirement to use a BRAM is when the memory has more than 512-bit of data. XC7Z020 consists of 36 kb of BRAM with a data width of up to 72. It is true dual-port with a programmable first-in-first-out (FIFO) logic. There is a total of 140 BRAMs available in the XC7Z020 device. Applications that implement DSP require the use of many multipliers and is best designed with a dedicated DSP block. Each DSP block consists of a 25×8-bit multiplier and a 48-bit accumulator. DSP blocks have capabilities that improve the speed and efficiency of many functions that go beyond DSP applications. Functionalities like wide multiplexers, memory address generators, bus shifters, and memory-map input/output register files can all be implemented using DSP blocks. The accumulator in the DSP block can also serve as a synchronous counter. XC7Z020 FPGA device has a total of 220 DSP blocks used for implementing DSP applications and other functionalities.

Fig. 5 shows graphical representations of the number of BRAMs and DSPs used by each RISC-V processor core. From Fig. 5, Roa Logic RV12, ORCA, BOOM, and MRISC make use of BRAMs with BOOM using the highest (6 BRAMs) representing 4.29% of the total. The rest of the processor cores do not use BRAMs. SiFive E31, SCR1, Rocket, BOOM, and Shakti-E make use of DSP blocks with BOOM using the most blocks (40 DSPs) which represent 18.18% of the total. The rest of the processors do not use DSP blocks. Hummingbird and PICORV32 do not use either BRAMs or DSPs which is ideal for low-cost IoT end devices.

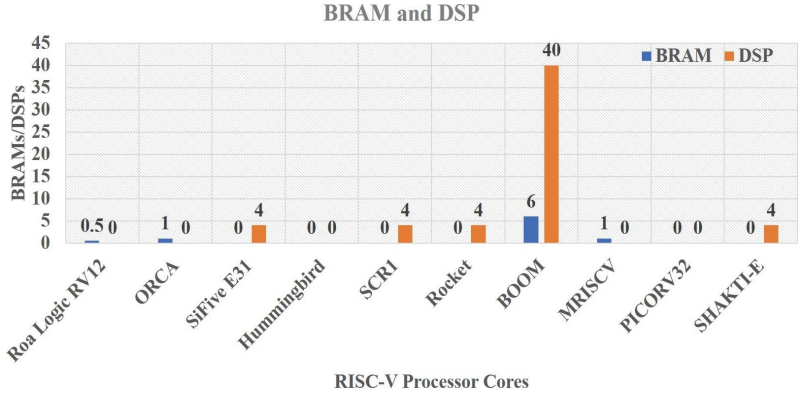


Fig. 5. BRAM and DSP utilization comparisons.

5.6 Power Consumption

Vivado tool has a power analysis capability that estimates the power consumption of a design after synthesis and beyond. The power report consists of both dynamic power and static power consumption. Static power is power consumed by a design when it is at a steady-state. This is a result of transistor leakage and it mostly depends on temperature, device process, and voltage. If these parameters are constant, static power is the same for all logic designs, therefore, this work concentrates on dynamic power. Dynamic power is the power consumed when a design is operational and switching activity occurs. This power is directly proportional to the frequency of a design. This work sets the frequency to 50 MHz to analyze the power consumed by each processor core.

The graphical representation of the dynamic power consumption by each processor core is shown in

Fig. 6. From the figure Shakti-E and BOOM consume power above 0.9 W with BOOM consuming the most power at 0.188 W. All other processor cores consume power less than 0.1 W with PICORV32 consuming the least power at 0.013 W which is ideal for low-cost and low power devices.

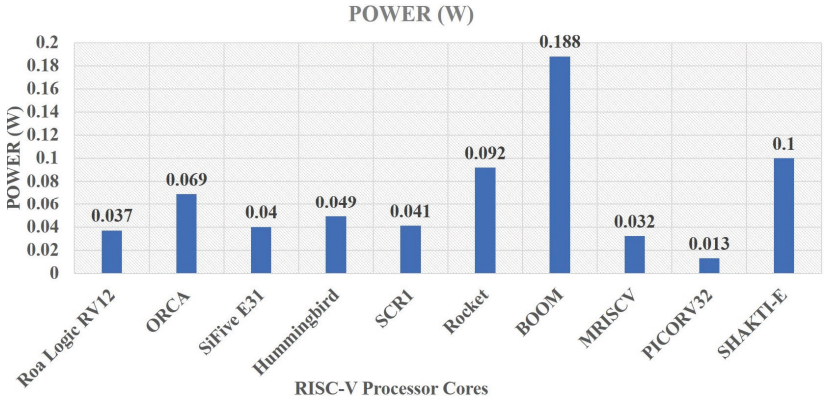


Fig. 6. Power consumption comparisons.

6. Ideal RISC-V Processor for Low-Cost Devices

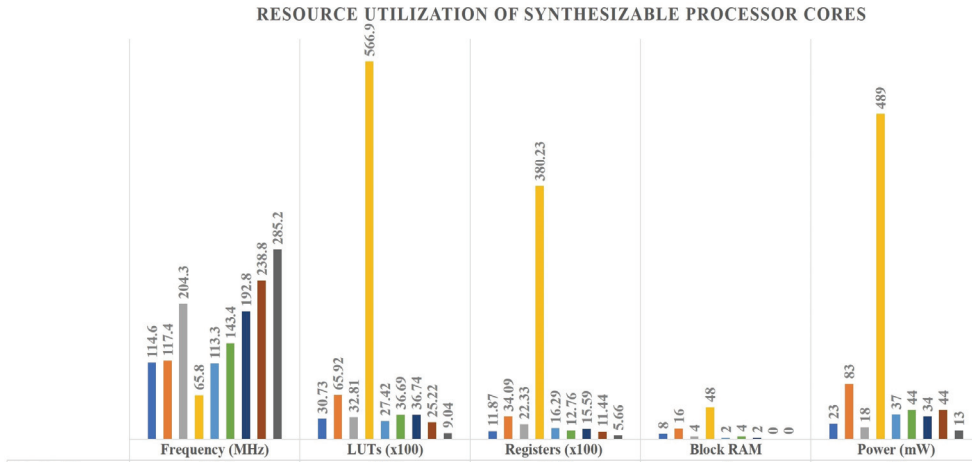
Over 80% of the SoC devices in the IoT market consist of components such as processors, bus, power management unit, security protocols, and other peripherals. The processor core usually takes up less than 5% of the hardware resources. The number of LUTs utilized and the power consumed are the two most important metrics for FPGA designs. From the synthesis reports, PICORV32, ORCA and MRISCV processors consumed 1.7%, 2.55%, and 3.56% of the available LUTs respectively. In terms of area, the three processors are suitable for low-cost devices. In terms of power consumption, MRISCV and ORCA consumed 2 times and 5 times that of PICORV32, respectively. This implies that PICORV32 is much more suitable for low area and low power applications than the other processors. It is evident that the BOOM processor core uses the most hardware resources and consumes the most dynamic power as compared to the other processor cores and therefore not suitable for low-cost devices. To get the bigger picture, it is fair to compare the synthesis results of PICORV32 to other open-source synthesizable processor cores that implement other ISAs.

Jia et al. [19] performed the synthesis of seven open-source synthesizable processor cores that implement a variety of ISAs. The processor cores include LEON3 [40], LEON2 [41], OR1200 [42], S1 [43], AltOR32 [44], LatticeMicro32 [45], and Amber [46]. The processors implement ISAs such as ARMv2, LatticeMacro32, SPARCv8, SPARCv9, and OpenRISC. The address space of all the processors is 32-bit in width. The processors were synthesized using the Virtex-7 FPGA device. From the synthesis reports, the LUTs utilization, registers utilization, BRAM utilization, maximum frequency, and dynamic power consumption are extracted and compared. Here, a comparison is made between RISC-V PICORV32 and the processors implementing other ISAs.

Fig. 7 shows a graphical representation of the resources used by open-source processor synthesized by Jia et al. [19], together with RISC-V PICORV32 (synthesized by this work). The synthesis results by Rui et al shows that LEON3 consumes the least hardware resources such as LUTs, registers, and block RAM

as compared to the other processors while S1 consumes the most hardware resources. LEON3 again has the highest maximum frequency while S1 records the lowest maximum frequency.

When LEON3 synthesis results are compared to that of PICORV32 as can be seen in Fig. 7, it is evident that the PICORV32 performs better than LEON3. The maximum frequency is fairly close with PICORV32 using a little over 46 MHz above that of LEON3. LEON3 consumes almost 3 times the LUTs and 2 times the registers consumed by PICORV32. Lastly, LEON3 consumes 3 times the dynamic power consumed by PICORV32.



	Frequency (MHz)	LUTs (x100)	Registers (x100)	Block RAM	Power (mW)
Amber23	114.6	30.73	11.87	8	23
Amber25	117.4	65.92	34.09	16	83
LatticeMicrom32	204.3	32.81	22.33	4	18
S1	65.8	566.9	380.23	48	489
AltOR32	113.3	27.42	16.29	2	37
OR1200	143.4	36.69	12.76	4	44
LEON2	192.8	36.74	15.59	2	34
LEON3	238.8	25.22	11.44	0	44
PICORV32	285.2	9.04	5.66	0	13

Fig. 7. Hardware resource utilization of open source processors.

7. Conclusion and Future Work

In this paper, ten RISC-V processors are analyzed to determine the ideal processor core for low-cost and low power IoT sensing and data collection devices. The processors include Roa Logic RV12, ORCA, SiFive E31, SCR1, Rocket, BOOM, MRISCV, PICORV32, Shakti-E, and Hummingbird. The work starts by examining the limitations of low-cost hardware IoT devices in terms of memory requirement, hardware area, and power consumption. The characteristics of the low-cost devices are established after which the processor cores are examined. Xilinx Vivado v2018.02 Integrated Software Environment is used to synthesize the processor cores using the Zynq-7000 XC7Z020 FPGA device.

From the synthesis, the maximum frequency, LUTs, LUTRAM, Flip-flops, BRAM, DSP, and power consumption reports are extracted and analyzed. The PICORV32 processor consumed 904 LUTs, 24 LUTRAMs, and 566 flip-flops. It achieved a maximum frequency of 200MHz and dynamic power of 0.013W. PICORV32 processor consumed fewer hardware resources as compared to the other processor in terms of the hardware resource, frequency, and power while the BOOM processor utilized the most hardware resources. This indicates that among the ten RISC-V processors, PICORV32 will be ideal for

low-cost hardware devices while BOOM will not be suitable.

The PICORV32 synthesis results are compared to that of other processor cores implementing different ISAs such as ARM, LatticeMicro, OpenRISC, and SPARC. A promising processor core from the other ISAs is the LEON3 core. Comparing PICORV32 to that of LEON3, LEON3 consumed 3 times more LUTs and 3 times more dynamic power as PICORV32. This implies that PICORV32 is the ideal opensource synthesizable processor core for low-cost devices.

This work is vital because the first step in selecting a processor for a low-cost SoC device is to estimate the hardware resources used by the processor core. A processor with high application coverage and high performance but utilizes a high chip area and consumes a high amount of power cannot meet the requirement of a low-cost hardware device.

In the future, processor benchmarks such as Dhrystone, Stanford, Paranoia, and EEMBC will be used to measure the execution time of the PICORV32 RISC-V processor core. The benchmarks will be executed by implementing the processor core in a lightweight SoC application. This is done in other to determine whether the processor will be ideal for a high throughput application.

References

[1] J. Guth, U. Breitenbucher, M. Falkenthal, P. Fremantle, O. Kopp, F. Leymann, and L. Reinfurt, "A detailed analysis of IoT platform architectures: concepts, similarities, and differences," in *Internet of Everything*. Singapore: Springer, 2017, pp. 81-101.

[2] Y. B. Lin, L. K. Chen, M. Z. Shieh, Y. W. Lin, and T. H. Yen, "CampusTalk: IoT devices and their interesting features on campus," *IEEE Access*, vol. 6, pp. 26036-26046, 2018.

[3] P. Schleuniger, S. A. McKee, and S. Karlsson, "Design principles for synthesizable processor cores," in *Architecture of Computing Systems-ARCS 2012*. Heidelberg: Springer, 2012, pp. 111-122.

[4] F. Armknecht, M. Hamann, and V. Mikhalev, "Lightweight authentication protocols on ultra-constrained RFIDs: myths and facts," in *Radio Frequency Identification: Security and Privacy Issues*. Cham: Springer, 2014, pp. 1-18.

[5] A. Waterman, Y. Lee, D. A. Patterson, and K. Asanovic, "The RISC-V instruction set manual, Volume I: base user-level ISA," 2011 [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2011/EECS-2011-62.html.

[6] Roa Logic, "RV12 RISC-V 32/64-bit CPU Core Datasheet v1.3," 2018 [Online]. Available: https://github.com/RoaLogic/RV12/blob/master/docs/RoaLogic_RV12_RISCV_Datasheet.pdf.

[7] V. Melikyan, E. Babayan, A. Melikyan, D. Babayan, P. Petrosyan, and E. Mkrtchyan, "Clock gating and multi-VTH power design methods based on 32/28 nm ORCA processor," in *Proceedings of IEEE East-West Design and Test Symposium (EWDTS)*, Batumi, Georgia, 2015, pp. 1-4.

[8] SiFive, "SiFive E31 Coreplex Manual v1p0," 2016 [Online]. Available: https://static.dev.sifive.com/E31-Coreplex.pdf.

[9] Syntacore, "SCR1 external architecture specification version 1.2.4," 2019 [Online]. Available: https://github.com/syntacore/scr1/blob/master/docs/scr1_eas.pdf.

[10] K. Asanovic, R. Avizienis, J. Bachrach, S. Beamer, C. Celio, H. Cook, et al., "The Rocket Chip generator," 2016 [Online]. Available: https://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-17.pdf.

[11] C. Celio, P. F. Chiu, B. Nikolic, D. A. Patterson, and K. Asanovic, "BOOM v2: an open-source out-of-order RISC-V core," 2017 [Online]. Available: https://www2.eecs.berkeley.edu/Pubs/TechRpts/2017/EECS-2017-157.pdf.

[12] C. Duran, L. Rueda, G. Castillo, A. Agudelo, C. Rojas, L. Chaparro, H. Hurtado, et al., "A 32-bit RISC-V AXI4-lite bus-based microcontroller with 10-bit SAR ADC," in *Proceedings of 7th IEEE Latin American Symposium on Circuits and Systems (LASCAS)*, Florianopolis, Brazil, 2016, pp. 315-318.

[13] C. Wolf, "PicoRV32 – a size-optimized RISC-V CPU," [Online]. Available: https://github.com/cliffordwolf/picorv32.

[14] N. Gala, A. Menon, R. Bodduna, G. S. Madhusudan, and V. Kamakoti, "SHAKTI processors: an open-source hardware initiative," in *Proceedings of 29th IEEE International Conference on VLSI Design and 15th IEEE International Conference on Embedded Systems (VLSID)*, Kolkata, India, 2016, pp. 7-8.

[15] B. Hu, "Hummingbird E203 open source introduction," [Online]. Available: https://github.com/SI-RISCV/e200_opensource/blob/master/doc.

[16] Xilinx, "Vivado design suite user guide: high-level synthesis," 2018 [Online]. Available: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2018_2/ug902-vivado-high-level-synthesis.pdf.

[17] M. Christensson and D. Mattsson, "Evaluation of synthesizable CPU cores," 2004 [Online]. Available: https://www.gaisler.com/doc/Evaluation_of_synthesizable_CPU_cores.pdf.

[18] K. Westerlund, "Comparison of synthesizable processor cores," 2005 [Online]. Available: https://www.gaisler.com/doc/leon2_vs_nios2.pdf.

[19] R. Jia, C. Y. Lin, Z. Guo, R. Chen, F. Wang, T. Gao, and H. Yang, "A survey of open source processors for FPGA," in *Proceedings of IEEE International Conference on Field Programmable Logic and Applications (FPL)*, Munich, Germany, 2014, pp. 1-6.

[20] Texas Instruments, "A true system-on-chip solution for 2.4-GHz IEEE 802.15.4 and ZigBee applications," 2009 [Online]. Available: www.ti.com/lit/ds/symlink/cc2530.pdf.

[21] Nordic Semiconductor, "nRF91 low power cellular IoT," [Online]. Available: https://www.nordicsemi.com/eng/Products/Low-Power-Cellular-IoT/nRF9160-SiP.

[22] G. Pau, C. Chaudet, D. Zhao, and M. Collotta, "Next generation wireless technologies for Internet of Things," *Sensors*, vol. 18, no. 1, pp. 1-5, 2018.

[23] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe, "PRESENT: an ultra-lightweight block cipher," in *Cryptographic Hardware and Embedded Systems-CHES 2007*. Heidelberg: Springer, 2007, pp. 450-466.

[24] D. Paolo and A. D. Santis, "On ultralightweight RFID authentication protocols," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 4, pp. 548-563, 2011.

[25] R. Alvarez, C. Caballero-Gil, J. Santonja, and A. Zamora, "Algorithms for lightweight key exchange," *Sensors*, vol. 17, no. 7, pp. 1-14, 2017.

[26] Parallela, "Parallela-1.x Reference Manual Version 14.09.09," [Online]. Available: https://www.parallella.org/board.

[27] D. Chen, J. Cong, S. Gurumani, W. Hwu, K. Rupnow, and Z. Zhang, "Platform choices and design demands for IoT platforms: cost, power and performance tradeoffs," *IET Cyber-Physical Systems: Theory and Applications*, vol. 1, no. 1, pp. 70-77, 2016.

[28] F. Gao, Z. Huang, Z. Wang, and S. Wang, "An object detection acceleration framework based on low-power heterogeneous manycore architecture," in *Proceedings of IEEE 3rd World Forum on Internet of Things (WF-IoT)*, Reston, VA, 2016, pp. 597-602.

[29] VectorBlox, "VectorBlox MXP Programming Reference," 2018 [Online]. Available: https://github.com/VectorBlox/ mxp/blob/master/docs/mxp_reference.pdf.

[30] Intel, "Avalon interface specifications: updated for Intel Quartus prime design suite," 2019 [Online]. Available: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/manual/mnl_avalon_spec.pdf.

[31] Silicore, "Specification for WISHBONE System-on-Chip (SoC) interconnection architecture for portable IP cores," 2002 [Online]. Available: https://cdn.opencores.org/downloads/wbspec_b3.pdf.

[32] Xilinx, "Local memory bus (LMB) v3.0: LogiCORE IP product guide," 2016 [Online]. Available: https://www.xilinx.com/support/documentation/ip_documentation/lmb_v10/v3_0/pg113-lmb-v10.pdf.

[33] R. E. Kessler, "The Alpha 21264 microprocessor," *IEEE Micro*, vol. 19, no. 2, pp. 24-36, 1999.
[34] R. P. Weicker, "Dhrystone: a synthetic systems programming benchmark," *Communications of the ACM*, vol. 27, no. 10, pp. 1013-1030, 1984.
[35] C. Coleman, D. Narayanan, D. Kang, T. Zhao, J. Zhang, L. Nardi, P. Bailis, K. Olukotun, C. Re, and M. Zaharia, "DAWNBench: an end-to-end deep learning benchmark and competition," [Online]. Available: https://github.com/stanford-futuredata/dawn-bench-entries.
[36] R. Karpinski, "Paranoia: A floating-point benchmark," *Byte Magazine*, vol. 10, no. 2, pp. 223-235, 1985.
[37] J. A. Poovey, T. M. Conte, M. Levy, and S. Gal-On "A benchmark characterization of the EEMBC benchmark suite," *IEEE Micro*, vol. 29, no. 5, pp. 18-29, 2009.
[38] M. B. Lin, *Digital System Designs and Practices: Using Verilog HDL and FPGAs*. Singapore: Wiley, 2008.
[39] J. Bachrach, H. Vo, B. Richards, Y. Lee, A. Waterman, R. Avizienis, and J. Wawrzynek, "Chisel: constructing hardware in a Scala embedded language," in *Proceedings of IEEE Design Automation Conference (DAC)*, San Francisco, CA, 2012, pp. 1212-1221.
[40] AEROFLEX GAISLER, "SPARC V8 32-bit processor LEON3/LEON2-FT companion core datasheet," 2010 [Online]. Available: http://www.actel.com/ipdocs/LEON3_DS.pdf.
[41] GAISLER RESEARCH, "LEON2 processor user's manual," 2004 [Online]. Available: https://www.cse.wustl.edu/~roger/465M/leon2-1.0.23-xst.pdf.
[42] D. Lampret, "OpenRISC 1200 IP core specification," 2001 [Online]. Available: https://www.isy.liu.se/edu/kurs/TSEA44/OpenRISC/or1200_spec.pdf.
[43] SRISC, "Simply RISC S1 Core," [Online]. Available: https://github.com/freecores/s1_core/tree/master/docs.
[44] OpenCores, "AltOR32 – Alternative Lightweight OpenRISC CPU," 2015 [Online]. Available: https://opencores.org /projects/altor32.
[45] Lattice Semiconductor, "LatticeMico32 processor reference manual," 2012 [Online]. Available: https://www.latticesemi.com/view_document?document_id=52077.
[46] Amber, "Amber 2 core specification," 2019 [Online]. Available: https://opencores.org/projects/amber.

DennisAgyemanh Nana Gookyi https://orcid.org/0000-0002-6622-0538

He received a B.Sc. degree in Computer Engineering from Kwame Nkrumah University of Science and Technology, Ghana, in 2013 and M.ENG. degree in Information and Communication Engineering from Hanbat National University, South Korea in 2017 where he is currently a PhD candidate. His research interests include SoC design and verification platforms, lightweight cryptography, and SoC design for security.

Kwangki Ryoo https://orcid.org/0000-0001-8574-7418

He received B.S., M.S., and Ph.D. degrees in Electronic Engineering from Hanyang University, Korea in 1986, 1988, and 2000, respectively. From 1991 to 1994, he was an Assistant Professor at the Korea Military Academy (KMA) in South Korea. From 2000 to 2002, he worked as a Senior Researcher at the Electronics and Telecommunications Research Institute (ETRI), Korea. From 2010 to 2011, he was a Visiting Scholar at the University of Texas in Dallas. Since 2003, he has been a Professor at Hanbat National University, Daejeon Korea. His research interests include engineering education, SoC platform design and verification, image signal processing and multimedia codec design, and SoC design for security.