LOW-POWER EMBEDDED SYSTEM DESIGN APPLICATIONS USING FPGAS

Sreemathy T G

Lecturer in Electronics,
Govt Polytechnic College, Kunnamkulam.

ABSTRACT

Field-programmable gate array is referred to as FPGA. An integrated circuit uses hardware to implement code so that it can run a thousand times quicker than it can on a CPU. These circuits, sometimes known as arrays, are made up of memory, additional components, or customisable logic blocks (CLBs). FPGA embedded systems are getting more and more well-liked for a variety of uses in numerous sectors. Compared to conventional microcontroller-based systems, these systems offer a number of benefits, such as excellent performance, adaptability, and quick prototype times. We present a general reconfigurable embedded system design of convolution neural networks (CNN) based on FPGA and soft-core CPU to meet the demands of mobile computing and low-power application situations. Convolution, pooling, and active layers are among the hardware circuits that house the fundamental computer modules. The soft-core CPU runs many serial data processing units and controlling logic. Low power embedded systems and Internet of Things (IoT) products may include some advanced on-chip capabilities that help reduce power consumption, or they may have a variety of power management techniques applied at the design level. Power management and low power consumption are also made possible by complex algorithms in embedded systems; any low power system may require a combination of tactics to avoid using more battery power than is necessary. FPGAs are frequently used in research and development and restricted (low) production of customised goods, when the greater cost of individual FPGAs is not as significant and where designing and producing a custom circuit would not be practical.

Keywords:- Energy-Efficient, Convolutional Neural Network, Embedded System, Computation Architecture, FPGA.

INTRODUCTION

Integrated circuits (ICs) that can be programmed to implement any digital circuit are known as field-programmable gate arrays, or FPGAs. The primary distinction between FPGAs and traditional fixed logic implementations, like Application Specific Integrated Circuits (ASICs), is that the former allows for on-site programming by the designer, eliminates non-recurring engineering (NRE) expenses, and drastically shortens time-to-market. Comparing FPGAs to ASICs, they are about 20 times bigger, 3 times slower, and 12 times less power efficient [1]. because the configuration memory-controlled programmable switches take up a lot of space and add a lot of parasitic capacitance and resistance to the logic and routing resources. The researchers' primary focus has been on faster and more area-efficient programmable routing resources. The existing clustering, placement, and routing algorithms are improved upon by the Versatile Place and Route (VPR) tool, which is described in [2] and results in significant performance enhancements. Rationale to-memory planning apparatuses, depicted in [3]-[5], shows improvement in the space proficiency of FPGAs with implanted recollections wherein parts of the application are pressed into unused recollections prior to planning the remainder of the application into rationale components. The reduction of power consumption, which is a crucial factor in determining product size, weight, and efficiency, has taken center stage in recent years. The benefits of FPGAs high power utilization and region are the balanced as a rule. The steadily developing interest for low-power compact correspondences and PC frameworks is propelling new low power methods, particularly for FPGAs, which scatter altogether more power than fixed-rationale executions. In point of fact, the International Technology Roadmap for Semiconductors (ITRS) identifies low-power design methods as an essential technology requirement [5]. FPGAs, like all integrated circuits, also lose two kinds of power—static power and dynamic power. Transistor leakage consumes static power and dissipates it when current leaks from the power supply to ground through “off-state” transistors for one of three reasons: gate-induced drain leakage, sub-threshold leakage (from the source to the drain), and gate direct tunneling leakage. When capacitances are charged and discharged during the operation of the circuit and consumed during switching events in the core or I/O of the FPGA, dynamic power is dissipated. Toggling nodes consume the majority of dynamic power. Because they use a lot of transistors per logic function to program the device, FPGAs use a lot more power than their ASIC counterparts. Each logic element and the programmable routing that connects logic elements in an FPGA contain a large number of configuration bits. The FPGA's static and dynamic power dissipation are both affected by this additional circuitry, which provides flexibility. There are mainly three ways to cut down on the amount of power used by an FPGA: system, designer, and operation level of the chip/architecture. There are three primary approaches to reducing 3 FPGA power consumption, as stated by the authors in [8]. At the system level, changes can be made first (such as simplifying the algorithms used). Second, a designer can alter the logic partitioning, mapping, placement, and routing if the FPGA's architecture is established. At long last, assuming no progressions at all are conceivable, improving working states of the gadget might be as yet encouraging (this remembers changes for the capacitance, the stockpile voltage, and the clock recurrence). For the FPGA based circuits, 80% of the all out power is consumed at the degree of framework plan and RTL and rest 20% of the power is at Entryway and semiconductor level. Here it is intended to design and implement the digital system on an FPGA, then optimize it for the lowest possible power consumption without compromising its performance by connecting Xilinx and MATLab with the Simulink HDL coder and making use of an available optimization tool. These methods are superior to those used in the past to reduce power consumption at the design level of a FPGA-based system.

EMBEDDED SYSTEMS DESIGN WITH FPGA

Low-End FPGAs

Designed for low power consumption, low logic density, and low complexity per chip. Examples include:

- Spartan family from Xilinx
- Cyclone family from Intel

- Mach XO/ICE40 from Lattice Semiconductor
- Fusion Family from Microsemi.

Mid-Range FPGAs

Designed for optimum solution between the low-end and high-end FPGAs, balancing performance and cost. Examples include:

- Arria from Intel
- IGL002 from Microsemi
- Artix-7/Kintex-7 series from Xilinx
- ECP3 and ECP5 series from Lattice semiconductor.

High-End FPGAs

Designed for logic density and high performance. Examples include:

- Virtex family from Xilinx
- Speedster 22i family from Achronix
- Stratix family from Intel
- ProASIC3 family from Microsemi

DESCRIPTION OF BROAD AREA

With the development of field programmable devices (FPDs), the manufacturing process of digital hardware has undergone significant transformations in recent years. Due to their adaptability, reprogrammability, and high performance, FPGAs are now an appealing architecture option for any design of digital systems. In many applications, FPGAs are replacing ASICs because the cost of developing a single chip skyrockets with each new technology iteration, while the relative costs of FPGA architectures are becoming increasingly attractive. In many current ASIC applications, the more viable alternative is the FPGA because of its field reprogrammability and many other advantages, including prototyping. In many cases, the large area and high power consumption of FPGAs outweigh these benefits. Kuon and Rose have demonstrated the way that FPGAs can depend on multiple times bigger and consume multiple times more unique power than its identical ASIC execution [5] Because of the fast development of battery fueled gadgets and compact computerized applications; the more people are paying attention to power minimization and how to optimize it. For FPGA to enter mainstream low-power applications, low power consumption is a constraint. The proposed expansive region of my examination would be FPGA based ideal plan of an exhaustive computerized framework subsequent to considering the different FPGA based low power plan methods. From the review plainly practically 80% of the complete power is at framework and RTL level though balance 20% is at door and semiconductors level. Our focal point of exploration will be on planning any thorough FPGA based advanced framework and afterward to investigate the enhancement methods to decrease its ability to have an ideal plan of that computerized framework. The coding of the RTL description needs to be done correctly in order to get the most out of any design that uses a FPGA. It is proposed to compose VHDL code for a thorough 32 digit drifting point Number juggling Unit that plays out every one of the four math tasks for example expansion, deduction, duplication and division. After that, the VHDL design code would be put into action on an FPGA platform, and the parameters would be looked at mostly in terms of power and area. The VHDL code so executed would then be changed over on to MAT lab stage utilizing Simulink model for check of VHDL code in Modelsim, where the equivalent will be advanced on power utilizing enhancement strategies and the improved code so acquired will again be switched back over completely to FPGA stage, carried out and examined.[4]

OBJECTIVE OF STUDY

Because of their adaptability, field programmability, and re-configurability, FPGAs are gaining ground in digital system design over general purpose processors and application-specific integrated circuits. However, its use in digital systems with low power requirements is limited due to its high power consumption in comparison to its competitors[6]. The primary objective is to investigate the various low-power design strategies utilized for digital systems based on FPGAs. The goal will be to create a comprehensive digital system by writing code in VHDL and putting it into action on the Xilinx FPGA platform. After that, the goal will be to investigate optimization methods so that the digital system can be designed with the least amount of power possible. The principal objective of the examination would be:

- To investigate a variety of low-power design methods utilized in FPGA-based digital systems;
- To evaluate FPGA performance;
- To construct an FPGA-based low-power optimal digital system

LITERATURE REVIEW

This section reviews and discusses the work done and the various design methods used to reduce FPGA power. Tuan and Lai in [1] analyzed spillage in the Xilinx Simple 3 FPGA, a 90nm business FPGA. The breakdown of leakage in a Spartan-3 CLB, which is comparable to the Virtex-4 CLB, is shown in Table 1. Interconnects, configuration SRAM cells, and LUTs are more susceptible to leakage. Consolidated, these designs represent 88% of complete spillage.

According to Taun and Lai [9], only during the FPGA's configuration phase can the contents of its configuration SRAM cells change. At power-up, configuration is typically done once. As a result, the operating speed of the FPGA-implemented circuit is unaffected by the speed performance of the SRAM configuration cells of an FPGA. Previously published low leakage memory techniques or the use of memory cells with high-VTH or long channel transistors can slow down the SRAM cells and reduce or eliminate their leakage. Spartan-3's design did not place a lot of emphasis on leakage. Spartan-3 interconnects and LUTs would account for 55% and 26% of total leakage, respectively, if SRAM configuration leakage were eliminated entirely.

Rahman proposed four distinct approaches to address leakage in FPGA interconnects, applying well-known leakage reduction methods to interconnect multiplexers [2]. In the beginning, multiple OFF transistors on unselected multiplexer paths were made possible by introducing extra configuration SRAM cells. The objective is to make use of the "stack effect." A subsequent methodology portrayed by the creator is to format piece of the multiplexer in discrete wells, permitting body-predisposition procedures to be utilized to raise the VTH of multiplexer semiconductors that are not piece of the chose signal way. Negatively biasing the gate terminals of OFF multiplexer transistors was their third strategy. Sub-threshold leakage decreases significantly as a result of the negative gate bias. Finally, the authors suggested employing dual-VTH methods, in which some multiplexer transistors have a high VTH (slow/low leakage) and others have a low VTH (fast/leaky). The double VTH thought, influences FPGA switch intricacy, as the switch should relegate delay-basic signs to low-VTH multiplexer ways. In a more recent paper, Ciccarelli applies dual-VTH methods to multiplexers and routing switch buffers [3].

In recent times, FPGA manufacturers have begun to expand the size of embedded memory cells in their FPGAs. In low-cost FPGAs, the trend toward increasing the ratio of memory bits to logic cells can be observed. Furthermore, it can be deduced that reducing logic cell leakage power is preferable to doing so in the FPGA embedded memory. In [4] Meng and others proposed a computer aided design procedure to decrease spillage power dispersal in FPGA implanted memory bits by adding way crossing and area task strategies in the implanted memory planning. The authors assume that all embedded memory cells can connect to the two supply voltages VDDH and VDDL, which are a high and low supply voltage, in order to support the drowsy mode. The cell actually holds the put away information even while the memory bit is working at the low inventory voltage however the digit will consume less spillage power as spillage power is relative to the stock voltage. For memory bits, this scheme is known as drowsy memory.

The drowsy scheme has been used in a variety of ways to reduce memory leakage power, according to the literature. nonetheless, they chiefly designated store recollections in processors [5]. Cache memories have variable latencies and dynamic data placement have distinct characteristics, whereas FPGA embedded memories access are statically scheduled and the data is stored statically. As a result, solving the problem of minimizing leakage power in FPGA embedded memories aids in selecting the optimal static layout for the data stored there in order to maximize leakage savings when the memories are in a low-leakage state. While used memory entries are put into a sleep mode when they are not accessed and are woken up when needed, the unused memory entries are not to be brought out of that mode. Meng and others offered three distinct modes in [4]: modes of sleep, drowsiness, and activity By cutting off the supply voltage to the bits of memory that are not being used, the sleep mode is used to clear out unused memory entries. The authors of the study demonstrated that, without employing any method for dynamically waking up (or putting the used memory entries to sleep), one can save an average of 36% of the memory leakage power by simply placing the unused memory entries in the sleep mode (used-active). In addition, by using the minimum number of memory entries and turning off entries that are not in use (min-entry), embedded memories can save approximately 75% of leakage power on average. It has been noticed that the drowsy-long plan saves 10% more leakage power than the min-entry plan [4]. Additionally, the pathplace algorithm typically reduces leakage power by 95%. As a result, the min-entry and path place techniques are the two most effective memory layout methods. The min-passage plot offers excellent spillage power reserve funds as far as both computational time and additional hardware required by the FPGA since it just backings dynamic and off modes. However, there are three memory modes supported by the path-place scheme: active, off, and low leakage with data retention [4].

Both a homogeneous and a heterogeneous architecture were proposed by Kumar and Anis [6]. The homogeneous design utilizes inside the bunch sub blocks of various VTH, while the heterogeneous engineering utilizes interleaved two sorts of groups, where one of the groups is made out of low VTH rationale cells and the other comprises of low and high VTH rationale cells. A CAD framework that began by assigning the entire design to high 8 VTH logic cells was proposed by the authors. Then, at that point, the calculation begins allotting the rationale cells into low VTH cells as long as the cell has positive leeway and the new way slack doesn't become negative. The calculation bunches the rationale cells into the groups that relate to the design being utilized in the following stage. Finally, the clustered designs are integrated into the FPGA architecture through constrained placement. It was discovered that the delay penalties for the homogeneous and heterogeneous architectures are almost identical and result in very close power savings from leakage.

THE ADVANTAGES OF FPGA-BASED EMBEDDED SYSTEMS

FPGA's function in embedded system design. We will concentrate on the benefits of this approach since they provide the strongest evidence for how FPGA enables the creation of effective and efficient end products.

Low Energy Consumption

FPGA-based installed frameworks are substantially more energy effective than their partners in light of different arrangements. The typical static power consumption of a flash-based FPGA is 75 W. Certainly, the simplest microcontrollers may use less energy, starting at 40 W, but this all depends on the type of MCU used; some may still match FPGAs but offer less efficiency. FPGAs, on the other hand, are without a doubt the winners when compared to CPUs or GPUs.

Reprogrammability

The fact that field-programmable gate arrays (FPGAs) can be reprogrammed is yet another significant advantage of using FPGAs to design embedded systems. Engineers can add or change installed logic functions and elements even after they have been installed. This gives you more flexibility during the design phase as well as after the launch. Eventually, this means reserve funds, likewise with FPGAs, you really want less assets to plan models and test your items, and any potential blunders can be amended rapidly.

High Data Processing Speed

FPGA-based embedded systems speed up data processing due to their parallel functionality. The ability of various components of the field programmable gate array to analyze distinct data chunks speeds up the process.

This is especially important for devices that need to work with data that changes in real time or very quickly. FPGAs are frequently the only viable option because microprocessors and microcontrollers are simply unable to handle information at such a rapid rate.

High-Level Code Synthesis

Standard programming languages like C and C++ are supported by modern FPGAs. Subsequently, specialists can plan code in a more conventional climate and afterward just exchange it to a FPGA. The entire development process is made to run much more quickly and at lower costs as a result of this.

Lower Need for Peripherals

Since the I/O connection points of inserted frameworks in view of FPGAs are arranged in the product, architects can try not to incorporate extra I/O equipment. Accordingly, the general expenses of the entire framework drop, and it's feasible to finish and send off gadgets all the more rapidly, as you don't need to request or plan additional equipment.

CONCLUSION

There are benefits and drawbacks to each approach to lowering FPGA power consumption at the system, device, circuit, and architecture levels. A result of a low-power, high-performance FPGA-based design of a 32-bit floating point arithmetic unit that performs all four operations—adding, subtracting, multiplying, and dividing two 32-bit floating point numbers—with the lowest possible power consumption and without sacrificing any other design parameters. In order to achieve the research's goal, the proposed method must be adaptable to any digital system design utilizing an FPGA while consuming the least amount of power possible. FPGAs are incredibly adaptable. After the board is built, they enable developers to test any number of variables. New configuration files that enable new functionality are downloaded onto the device whenever modifications are required. OEMs are able to ship systems earlier in the design process because of this flexibility.

REFERENCES

[1]. T. Tuan and B. Lai. “Leakage Power Analysis of a 90nm FPGA". In: IEEE Custom Integrated Circuits Conference, pp. 57-60, San Jose, CA, 2003.

[2]. Rahman and V. Polavarapuv. “Evaluation of Low-Leakage Design Techniques for FieldProgrammable Gate Arrays". In: ACM/SIGDA International Symposium on Field Programmable Gate Arrays, pp. 23-30, Monterey, CA, 2004.

[3]. L. Ciccarelli, A. Lodi, and R. Canegallo. “Low Leakage Circuit Design for FPGAs". In: IEEE Custom Integrated Circuits Conference, pp. 715-718, Orlando, FL, 2004.

[4]. Meng Y, Sherwood T, Kastner R. Leakage power reduction of embedded memories on FPGAs through location assignment. In: Proceedings of Design Automation Conference (DAC). 2006:612–617

[5]. Kim NS, Flautner K, Blaauw D, et al. Circuit and micro architectural techniques for reducing cache leakage power. IEEE Trans Very Large Scale Integration (VLSI) Syst. 2004;12: 167–184.

[6]. Kumar A, Anis M. Dual-threshold CAD framework for sub threshold leakage power aware FPGAs. IEEE Trans Computer Aided Des Integrated Circ Syst. 2007;26:53–66.

[7]. Khabbazan, B., & Mirzakuchaki, S. (2019, August). Design and implementation of a low-power, embedded CNN accelerator on a low-end FPGA. In 2019 22nd Euromicro Conference on Digital System Design (DSD) (pp. 647-650). IEEE.

[8]. Chéour, R., Khriji, S., El Houssaini, D., Baklouti, M., Abid, M., & Kanoun, O. (2019). Recent trends of FPGA used for low-power wireless sensor network. IEEE Aerospace and Electronic Systems Magazine, 34(10), 28-38.

[9]. Pinyotrakool, K., & Supmonchai, B. (2020, March). Design of a low power processor for embedded system applications. In 2020 8th International Electrical Engineering Congress (iEECON) (pp. 1-4). IEEE.

[10]. Verma, G., Khare, V., & Kumar, M. (2019). More precise FPGA power estimation and validation tool (FPEV_Tool) for low power applications. Wireless Personal Communications, 106, 2237-2246.

[11]. Branco, S., Ferreira, A. G., & Cabral, J. (2019). Machine learning in resource-scarce embedded systems, FPGAs, and end-devices: A survey. Electronics, 8(11), 1289.