Journal of Engineering Sciences
Vol 10, Issue 8, Aug /2019
ISSN NO:0377-9254

HIGH-SPEED AND LOW-POWER VLSI-ARCHITECTURE FOR INEXACT SPECULATIVE ADDER

#1SYED SHANAWAZ , #2V PARTHA SARATHI REDDY
1,2Assistant Professor
DEPT OF ECE
Dr.K V SUBBA REDDY INSTITUTE OF TECHNOLOGY, KURNOOL

ABSTRACT: This project presents carry-look ahead adder (CLA) based design of the contemporary inexact-speculative adder (ISA) which is fine grain pipelined to include few logic gates along its critical path and thereby, enhancing the frequency of operation. Additionally, various stages of the proposed ISA architecture has been clock gated to reduce the power consumed by this design. Functional verification and hardware implementation for various configurations of the suggested ISA is carried out on field-programmable gate-array (FPGA) platform. It could operate at a maximum clock frequency of 324 MHz which is 52% better then the conventional ISA. Thereafter, the synthesis and post-layout simulation of 32-bit proposed ISA is carried out using 90 nm complementary metal oxide semiconductors (CMOS) technology node for power and area analysis. Our design occupied 5.11 mm2 of chip area and consumed 9.68 mW of total power at 400 MHz clock frequency. The proposed ISA burns 52.8% lesser power than the state-of-the-art work

I. INTRODUCTION

1.1 MOTIVATION

As the scale of integration keeps growing, more and more sophisticated signal processing systems are being implemented on a VLSI chip. These signal processing applications not only demand great computation capacity but also consume considerable amount of energy. While performance and Area remain to be the two major design tolls, power consumption has become a critical concern in today's VLSI system design[]. The need for low-power VLSI system arises from two main forces. First, with the steady growth of operating frequency and processing capacity per chip, large currents have to be delivered and the heat due to large power consumption must be removed by proper cooling techniques. Second, battery life in portable electronic devices is limited. Low power design directly leads to prolonged operation time in these portable devices.

Addition usually impacts widely the overall performance of digital systems and a crucial arithmetic function. In electronic applications adders are most widely used. Applications where these are used are multipliers, DSP to execute various algorithms like FFT, FIR and IIR. Wherever concept of multiplication comes adders come in to the picture. As we know millions of instructions per second are performed in microprocessors. So, speed of operation is the most important constraint to be considered while designing multipliers. Due to device portability miniaturization of device should be high and power consumption should be low. Devices like Mobile, Laptops etc. require more battery backup.

So, a VLSI designer has to optimize these three parameters in a design. These constraints are very difficult to achieve so depending on demand or application some compromise between constraints has to be made. Ripple carry adders exhibits the most compact design but the slowest in speed. Whereas carry look ahead is the fastest one but consumes more area. Carry select adders act as a compromise between the two adders. In 2002, a new concept of hybrid adders is presented to speed up addition process by Wang et al. that gives hybrid carry look-ahead/carry select adders design. In 2008, low power multipliers based on new hybrid full adders is presented.

DESIGN of area- and power-efficient high-speed data path logic systems are one of the most substantial areas of research in VLSI system design. In digital adders, the speed of addition is limited by the time required to propagate a carry through the adder. The sum for each bit position in an elementary adder is generated sequentially only after the previous bit position has been summed and a carry propagated into the next position.

The CSLA is used in many computational systems to alleviate the problem of carry propagation delay by independently generating multiple carries and then select a carry to generate the sum.

1.2 NEED FOR LOW POWER DESIGN

The design of portable devices requires consideration for peak power consumption to ensure reliability and proper operation. However, the time averaged power is often more critical as it is linearly related to the battery life. There are four sources of power dissipation in digital CMOS circuits: switching power, short-circuit power, leakage power and static power. The following equation describes these four components of power:

$$P_{avg} = P_{switching} + P_{short-circuit} + P_{leakage} + P_{static} \quad (2.1)$$

$$= \alpha C_L V_{dd} V_s f_{ck} + I_{sc} V_{dd} + I_{leakage} V_{dd} + I_{static} V_{dd} \quad (2.2)$$

Pswitching is the switching power. For a properly designed CMOS circuit, this power component usually dominates, and may account for more than 90% of the total power. α Denotes the transition activity factor, which is defined as the average number of power consuming transitions that is made at a node in one clock period. Vs is the voltage swing, where in most cases it is the same as the supply voltage, Vdd. CL is the node capacitance. It can be broken into three components, the gate capacitance, the diffusion capacitance, and the interconnect capacitance. The interconnect capacitance is in general a function of the placement and routing. fck is the frequency of clock. The switching power for static CMOS is derived as follows.

During the low to high output transition, the path from Vdd to the output node is con-ducting to charge CL. Hence, the energy provided by the supply source is

$$E = \int_0^\infty V_{dd} I(t) dt \quad (2.3)$$

Where $I(t) = \frac{V_s}{R} e^{-t/RC_L}$ is the current drawn from the supply? Here, R is the resistance of the path between the Vdd and the output node. Therefore, the energy can be rewritten as

$$E = C_L V_{dd} V_s \quad (2.4)$$

During the high to low transition, no energy is supplied by the source. Hence, the average power consumed during one clock cycle is

$$P = \frac{E_{percycle}}{T} = C_L V_{dd} V_s f_{ck} \quad (2.5)$$

Eq. (2.4) and Eq. (2.5) estimate the energy and the power of a single gate only. From a system point of view, α is used to account for the actual number of gates switching at a point in time.

Pshort_circuit is the short-circuit power. It is a type of dynamic power and is typically much smaller than Pswitching. Isc is known as the direct-path short circuit current. It refers to the conducting current from power supply directly to ground when both the NMOS and PMOS transistors are simultaneously active during switching.

Pleakage is the leakage power. Ileakage refers to the leakage current. It is primarily determined by fabrication technology considerations and originates from two sources. The first is the reverse leakage current of the parasitic drain-/source-substrate diodes. This current is in the order of a few femtoamperes per diode, which translates into a few microwatts of power for a million transistors. The second source is the sub threshold current of MOSFETs, which is in the order of a few nanoamperes. For a million transistors, the total sub threshold leakage current results in a few mill watts of power.

Pstatic is the static power and Istatic is static current. This current arises from circuits that have a constant source of current between the power supplies such as bias circuitries, pseudo-NMOS logic families. For CMOS logic family, power is dissipated only when the circuits switch, with no static power consumption.

Energy is independent of the clock frequency. Reducing the frequency will lower the power consumption but will not change the energy required to perform a given operation, as depicted by Eq. (2.4) and Eq. (2.5). It is important to note that the battery life is determined by energy consumption, whereas the heat dissipation considerations are related to the power consumption.

There are four factors that influence the power dissipation of CMOS circuits. They are technology, circuit design style, architecture, and algorithm. The challenge of meeting the contradicting goals of high performance and low power system operation has motivated the development of low power process technologies and the scaling of device feature sizes.

Design considerations for low power should be carried out in all steps in the design hierarchy, namely 1) Fundamental, 2) material, 3) device, 4) circuit, and 5) system.

1.2.1 LOW VOLTAGE

Power consumption is linearly proportional to voltage swing (Vs) and supply voltage (Vdd) as indicated in Eq. (2.5). For most CMOS logic families, the swing is typically rail-to-rail. Hence, power consumption is also said to be proportional to the square of the supply voltage, Vdd. Therefore, lowering the Vdd is an efficient approach to reduce both energy and power, presuming that the signal voltage swing can be freely chosen. This is, however, at the expense of the delay of circuits. The delay, td, can be shown to be proportional to $V_{dd}/(V_{dd} - V_T)^\gamma$. The exponent γ is between 1 and 2. It tends to be closer to 1 for MOS transistors that are in deep sub-micrometer region, where carrier velocity saturation may occur. γ Increases toward 2 for longer channel transistors.

The current technology trends are to reduce feature size and lower supply voltage. Lowering Vdd leads to increased circuit delays and therefore lower functional throughput. Smaller feature size, however, reduces gate delay, as it is inversely proportional to the square of the effective channel length of the devices. In addition, thinner gate

oxides impose voltage limitation for reliability reasons. Hence, the supply voltage must be lowered for smaller geometries. The net effect is that circuit performance improves as CMOS technologies scale down, despite of the Vdd reduction. Therefore, the new technology has made it possible to fulfill the contradicting requirements of low-power and high throughput.

The various techniques that are currently used to scale the supply voltage include optimizing the technology and device reliability, trading off area for low power in architecture driven approach, and exploiting the concurrency possibility in algorithmic transformations. Hence, the voltage scaling is limited by the threshold voltage Vth.

In applications such as digital processing, where the throughput is of more concern than the speed, architecture can be designed to reduce the supply voltage at the expense of speed without throughput degradation. Hence, the performance of the system can be maintained.

1.3 LANGUAGE AND TOOLS USED

- Verilog (HDL)
- Modelsim6.4b
- Xilinx ISE 10.1

1.4 ADVANATGES

- Low power consumption
- Less area (less complexity)
- More speed compare regular CSLA

1.5 APPLICATIONS

- Arithmetic logic units
- High Speed multiplications
- Advanced microprocessor design Digital signal process

II. LITERATURE REVIEW

2.1 TYPES OF ADDERS

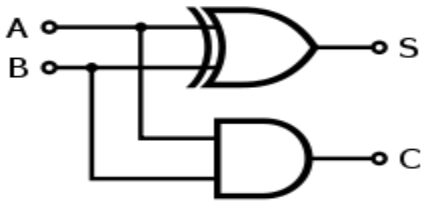
Addition is the most common and often used arithmetic operation on microprocessor, digital signal processor, especially digital computers. Also, it serves as a building block for synthesis all other arithmetic operations. Therefore, regarding the efficient implementation of an arithmetic unit, the binary adder structures become a very critical hardware unit. In any book on computer arithmetic, someone looks that there exists a large number of different circuit architectures with different performance characteristics and widely used in the practice. Although many researches dealing with the binary adder structures have been done, the studies based on their comparative performance analysis are only a few.

This is about a digital circuit. For an electronic circuit that handles analog signals see mixer. In electronics, an **adder** or **summer** is a digital circuit that performs addition of numbers. In many computers and other kinds of processors, adders are used not only in the arithmetic logic unit(s), but also in other parts of the processor, where they are used to calculate addresses, table indices, and similar.

Although adders can be constructed for many numerical representations, such as binary-coded decimal or excess-3, the most common adders operate on binary numbers. In cases where two's complement or ones' complement is being used to represent negative numbers, it is trivial to modify an adder into an adder–subtractor. Other signed number representations require a more complex adder. Different type of adders as follows

2.1.1 HALF ADDER

The **half adder** adds two one-bit binary numbers *A* and *B*. It has two outputs, *S* and *C* (the value theoretically carried on to the next addition); the final sum is $2C + S$. The simplest half-adder design, pictured on the right, incorporates an XOR gate for *S* and an AND gate for *C*. With the addition of an OR gate to combine their carry outputs, two half adders can be combined to make a full adder.



A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Figure 1 Half Adder

2.1.2 FULL ADDER

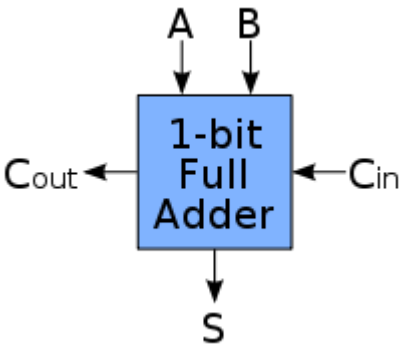


Figure 2 Full Adder

Schematic symbol for a 1-bit full adder with C_{in} and C_{out} drawn on sides of block to emphasize their use in a multi-bit adder. A **full adder** adds binary numbers and accounts for values carried in as well as out. A one-bit full adder adds three one-bit numbers, often written as *A*, *B*, and C_{in}; *A* and *B* are the operands, and C_{in} is a bit carried in from the next less significant stage.[2] The full-adder is usually a component in a cascade of adders, which add 8, 16, 32, etc. binary numbers. The circuit produces a two-bit output sum typically represented by the signals C_{out} and *S*, where $\text{sum} = 2 \times C_{out} + S$. The one-bit full adder's truth table is:

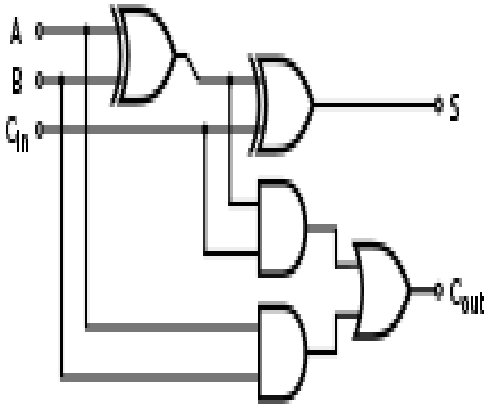


Figure 3 Full Adder-Gate Levels

A full adder can be implemented in many different ways such as with a custom transistor-level circuit or composed of other gates. One example implementation is with $S = A \oplus B \oplus C_{in}$ and $C_{out} = (A \cdot B) + (C_{in} \cdot (A \oplus B))$.

In this implementation, the final OR gate before the carry-out output may be replaced by an XOR gate without altering the resulting logic. Using only two types of gates is convenient if the circuit is being implemented using simple IC chips which contain only one gate type per chip. In this light, C_{out} can be implemented as

$$C_{out} = (A \cdot B) \oplus (C_{in} \cdot (A \oplus B))$$

A full adder can be constructed from two half adders by connecting *A* and *B* to the input of one half adder, connecting the sum from that to an input to the second adder, connecting C_i to the other input and OR the two carry outputs. Equivalently, *S* could be made the three-bit XOR of *A*, *B*, and C_i, and C_{out} could be made the three-bit majority function of *A*, *B*, and C_i.

III. PROPOSED DESIGN

3.1 Introduction

High-speed adders are highly desirable in the present day scenario, though power (or energy) and silicon area are equally vital. Spectrum sensors used in intelligent cognitive-radio environment [1], [2] as well as internet of everything (IoE) [3] devices focused on physical interfaces are largely-explored research areas in the recent time. Hardware for the algorithms of such applications is basically focused on sensing and actuating where the response time is key component to be optimized for real-time interfaces. Thereby, the design of highly optimized adders in terms of speed play significant role in the present era and hence this paper focuses in the design of same. With tolerable degradation in accuracy and performance, it is feasible to conceive high-speed, low power and area efficient design using inexact and approximate circuit technique [4]. Accuracy of such circuits can be traded off to improve the power and speed by speculation.

Thereby, such adders are referred as inexact speculative adder (ISA). Various optimized versions of such ISA have been reported in literature [5]-[9] and these works concentrated mostly on enhancing the accuracy of their results. However, there is space to further improve the speed of such adders by retaining the accuracy with minimum error. Thereby, our contributions in this work are as follows: design and analysis of the carry looks ahead adder (CLA) based ISA has been carried out. Thereafter, this adder is fine grain pipelined to reduce the critical path delay that further enhances the operating speed.

FPGA implementation of 8, 16 and 32 bit versions of the proposed ISA has been carried. Obtained post place- &-route results of these adders are compared with reported no pipelined ISAs. Subsequently, clock signal fed to various stages of the deep pipelined ISA-architecture has been gated to reduce the power consumption. Eventually, ASIC synthesis and post-layout simulation of the proposed 32-bit ISA has been

performed in 90 nm-CMOS technology node and is compared with the state-of-the-art ISA adder.

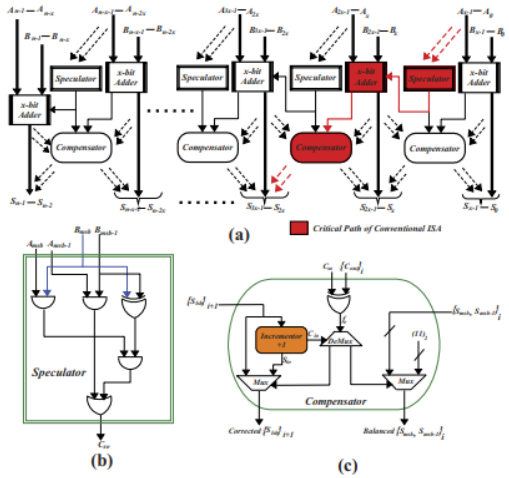


Fig. 4: (a) Basic block diagram of n-bit conventional inexact speculation adder (ISA). (b) Gate-level circuit representation of speculator block. (c) Digital architecture of compensator block.

3.2. PROPOSED VLSI ARCHITECTURE OF ISA

Block diagram and data flow of conventional ISA for n-bit addition is shown in Fig. 1. In the proposed architecture, we have segregated the n-bit input into 4-bit blocks (i.e., the value of x = 4 in Fig. 1) and each of these blocks is fed as operands to the x-bit adder. Unlike the conventional ISA architecture, the adder unit has been replaced with 4-bit CLA to further enhance the speed of operation. Comprehensive explanation with circuit details of various sub blocks of this adder are presented as follows: 1) Speculator and Adder Blocks: Prior delving into the circuit details, it is necessary to understand the notations used in this paper. Two n-bit operands for addition are represented as A = {$A0$, $A1$, ...$An-1$} and B = {$B0$, $B1$, ...$Bn-1$}; whereas, the sum, carry input and carry output are expressed as S = {$S0$, $S1$, ...$Sn-1$}, Cin and $Cout$ respectively. Gate-level circuit diagram of the speculator used in our adder design is presented in Fig. 1(b).

This block is based on CLA logic to speculate the output carry for each 4-bit adder block. Speculation is carried out for 'r' msb bits of each block where r is less than the size of block, (i.e., r < x = 4). Subsequently, the input carry for each speculator block is 0 (or 1) which introduces positive (or negative) errors respectively. The output carry, which is denoted as Cso, from each speculator block is fed as an input carry for the adder block succeeding it, as shown in Fig. 1. Now, each 4-bit adder block need not wait for the input carry from the preceding 4- bit adder block. Instead, all such adder blocks perform simultaneous additions on receiving input carries from the concerned speculator blocks. Speculator block computes carry based on the equation shown below:

$$Pi = Ai \oplus Bi;\ Gi = Ai \cdot Bi;\ Ci{+}1 = Gi + (Pi \cdot Ci); \quad (1)$$

Where (i+1)th carry bit is computed using the propagate (Pi), generate (Gi) and carry (Ci) of i th bit. This block is situated along the critical path of ISA architecture; however, it doesn't produce much delay as it computes the carry for few bits. On the other hand, adder block performs addition of 4-bit input blocks using CLA logic based on the equation below.

$$Si = Pi \oplus Ci. \quad (2)$$

This work employs CLA in the suggested architecture, as it has smaller propagation delay compared to the conventional ones. Here, the local sum obtained in parallel from each adder block is not the exact output because the addition has been performed using speculated carry inputs. Correction or Balancing of such sum value is carried out by the compensator block, as illustrated in Fig. 1. Adder and compensator blocks are the ones which consume maximum delay along the critical path of the architecture. Thereby, we tend to reduce this delay using the concept of pipelining which will be discussed in the later sections of this paper.

2) Compensator Block: Fig. 1(c) shows the digital architecture for compensator block used in the proposed ISA adder. This block compares the output carry from each 4-bit adder block with the corresponding speculated carry using a XOR gate. Thereafter, the output from XOR gate generates an error flag (fe) that triggers the activation of one of the two compensation techniques: error correction and reduction. If the XOR-gate output is '0' then the local sum is directly passed to the final output. Similarly, if the XOR gate gives '1' then this indicates that an error has occurred which can be either positive or negative. A positive error indicates a speculation of '0' instead of '1' and, hence, induces too low sum. Albeit a negative error indicates a speculation of '1' instead of '0' which induces too high sum. The compensation block performs an unsigned increment or decrement to the group of LSBs in the direction of this potential error (too high error is solved by a '-1' and too low by a '+1'). This correction is possible only if it does not result overflow. In case of overflow, the compensation block balances a group of MSBs of the preceding sub-adder in the opposite direction of the error. Balancing is performed based on the following observation:

$$2n > \{2n + 2n{-}1 + 2n{-}2 + ... + 20\}. \quad (3)$$

When $2n$ error is detected in the sum, it could be compensated by intentionally causing the LSB errors for the sum in the opposite direction (for

example, $Sn{-}1 = 1 \longrightarrow Sn{-}1 = 0$). In the best case where all the LSBs can be balanced in the opposite direction, the total error is reduced to 1, as follows:

$$\{2n - 2n{-}1 - 2n{-}2 - \ldots\ldots - 20\} = 1. \quad (4)$$

In general, if the number of bits used for correction is p then the first computed 'p' LSBs from the 4-bit adder block are passed on to the compensation block where it is checked whether any overflow occurs or not. It does then the right compensation technique which is balancing will be selected. All this is carried out prior the 4-bit adder block finishes computing the sum of all other bits. Preferably the value of p is 1 for the optimum results. Thus, a significant feature of this adder is that neither the pre-computing of error correction nor the compensation choice lies in the critical path of the ISA adder.

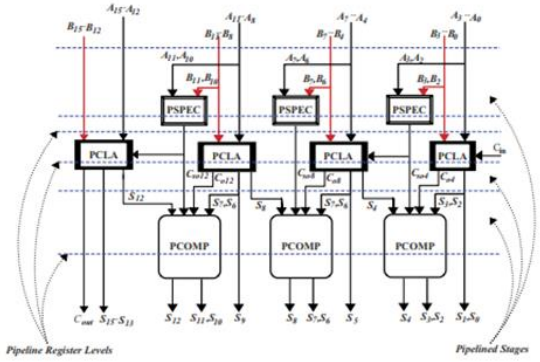


Fig.5: Deep-pipelined VLSI architecture of the proposed ISA for n = 16 bits and x = 4 bits, with five pipeline stages, for high speed applications.

The components of compensation block which are involved in the overall critical path of ISA are the XOR gate, de-multiplexer and multiplexer.

3.3. Fine-Grain Pipelined Architecture

In the conventional ISA architecture, let us assume that the combinational delay of 4-bit adder, speculator and compensator blocks to be $\partial 4b{-}adder$, $\partial spec$ and $\partial comp$ respectively. In this architecture, carrying is speculated for each 4-bit adder block and based on this; adder block calculates the local sum. Thereafter, the faulty speculation is detected by comparing speculated carry-in and prior carry-out from 4-bit adder. Subsequently, compensator block performs the correction and balancing operation. Thus, the critical path of the conventional ISA architecture includes delays of speculator of the $i\ th$ instant and the 4-bit adder plus compensator delays of $(i{+}1)th$ instant, as shown in Fig. 1 (with colored lines and blocks) and expressed as follows:

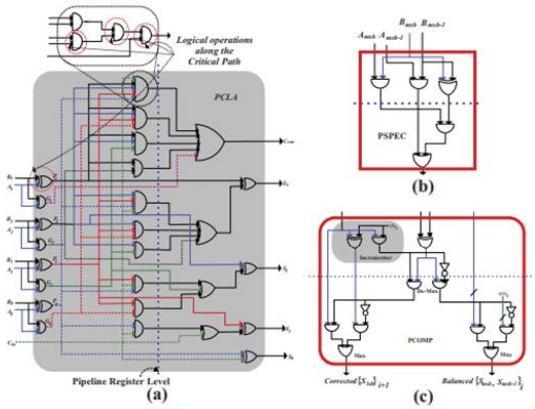


Fig. 6: Gate-level circuit of (a) Four-bit pipelined-carry look-ahead adder (PCLA) (b) Pipelined speculator (PSPEC) (c) Pipelined compensator (PCOMP) used in the proposed ISA VLSI architecture

$$\partial critical = (\partial 4b{-}adder)i + (\partial spec)i{+}1 + (\partial comp)i{+}1. \quad (5)$$

Thereby, based on the internal architectures of speculator and compensator blocks, the detailed version of critical path delay is given as:

$$\partial critical = (\partial 4b{-}adder)i + (2{\times}\partial xor + \partial and)i{+}1 + (\partial xor + \partial demux + \partial mux)i{+}1. \quad (6)$$

where ∂xor, ∂and, $\partial demux$ and ∂mux are the combinational delays of the logical XOR, logical AND, de-multiplexer and multiplexer respectively.

It is to be noted that the speculator, compensator, 4-bit adder and the overall ISA design is feed-forward VLSI architecture. Thereby, carefully analyzing these circuits and on fine-grain pipelining this ISA architecture, we can shorten the critical path delay and hence realize fast VLSI-architecture for this adder. For the ease of understanding, pipelining process of this work has been explained using n = 16 bit ISA architecture. Even though the value of n increases, critical path delay is unaffected because the value of x is always 4 bit (as discussed earlier) and the adder, speculator as well as compensator architectures remain unchanged. The proposed 16-bit ISA VLSI-architecture is shown in Fig. 2 where the conventional blocks has been replaced by the pipelined speculator (PSPEC), pipelined compensator (PCOMP) and pipelined 4-bit CLA (PCLA) units. Sub blocks PSPEC, PCLA and PCOMP contain two pipelined stages.

Overall architecture of the suggested ISA adder has been designed with five pipelined stages and there are six levels of registers included in this design, as shown in Fig. 2. This is a scalable architecture because the number of pipeline stages remains constant on increasing the bit widths of the operands, retaining the same critical path delay. The deep pipelined architectures of

sub block have been illustrated in Fig. 3. It shows the gate-level designs of PSPEC, PCOMP, PCLA and their respective pipelined stages. On observing the proposed VLSI architectures from Fig. 2 and 3, it can been seen that the critical path of suggested architecture lies in PCLA and it includes only four two-input gate delays (one XOR and three AND gate delays). Thereby, the expression of critical path delay that decides the maximum clock frequency of the proposed ISA is given as

$\partial crt{-}prop = \partial clk{-}Q(ff) + \partial xor + 3{\times}\partial and + \partial setup(ff)$ (7)

where $\partial clk{-}Q(ff)$ and $\partial setup(ff)$ represent clock-to-Q delay and setup time of launch and capture flip flops, respectively, in the design. Eventually, the maximum clock frequency that can be achieved by suggested ISA architecture in real world scenario is

$Fmax \leq 1/\{\partial crt{-}prop - \partial skew\}$ (8)

where $\partial skew$ represents the clock skew which may occur in such scenario.

3.4 FPGA Implementation

In this work, the proposed ISA adder-architecture has been coded in hardware descriptive language (HDL) and then simulated as well as synthesized in ISE 14.3 design suite. We have synthesized this architecture for three different configurations: n = 8-bit, n = 16- bit and n = 32-bit. After the successful syntax check and synthesis of the design, the generated net-lists are placed and routed (P&R) on Spartan-3E version of Xilinx FPGA board. Therefore, post P&R simulated waveform of the proposed 32-bit ISA is shown in Fig. 2 (other waveforms are excluded due lack of space in the paper). On the other side, the resource utilization and the timing information for all the configurations of suggested adder is listed in Table I. Additionally; it includes the resource and timing information of the reported non-pipelined (NPL) adder architectures. It can be observed that the proposed adder can operate at a maximum clock frequency of 324 MHz.

Table 3.1: Comparison of area consumed by the proposed 32-bit pipelined adder (PLA) with respect to 32-bit non-pipelined adder (NPLA)

ISA Architectures	32-bit PLA	32-bit NPLA
Technology (nm)	90	90
Supply Voltage (V)	0.9	0.9
Sequential Std. Cells	224 (3.25 mm^2)	98 (1.33 mm^2)
Inverter Std. Cells	95 (0.72 mm^2)	33 (0.57 mm^2)
Buffer Std. Cells	1 (3.12 μm^2)	–
Logic Std. Cells	190 (1.15 mm^2)	88 (1.12 mm^2)
Total Number of Std. Cells	510	219
Total Area (mm^2)	5.11	3.03

This value is precisely 42.15%, 48.48% and 52% better than the clock frequencies achieved by 8-bit, 16-bit and 32-bit NPL adders, respectively, reported in [9]. The comparison of exact area occupied and power consumed by these adders are possible by synthesizing as well as laying out ASIC for each of such adders.

Such analysis on area and power is also carried out in the next subsection. However, Table I clearly shows some degradation in the FPGA resources utilization of the proposed ISAs. Nevertheless, the precise degradation will be demonstrated in subsequent subsection.

3.5 Power and Area Analysis

In the digital circuit design, pipelining is the process of shortening the critical path at the cost of area which is predominated by the registers used to create pipeline stages in the design. Therefore, the suggested ISA architecture that is deep pipelined definitely requires extra registers in comparison with the conventional ones.

Table 3.2: Comparison of post P&R results obtained from FPGA implementations of 8, 16, 32-bit pipelined and non-pipelined ISA designs.

ISA Configurations	NPL$^{\Phi}$ (n=8b)	PL‡ (n=8b)	NPL$^{\Phi}$ (n=16b)	PL‡ (n=16b)	NPL$^{\Phi}$ (n=32b) [9]	PL‡ (n=32b)
FPGA Family	Spartan3E	Spartan3E	Spartan3E	Spartan3E	Spartan3E	Spartan3E
FPGA Device	xc3s500e	xc3s500e	xc3s500e	xc3s500e	xc3s500e	xc3s500e
Slices	21	33	41	66	85	134
4-Input LUTs	27	51	52	105	115	213
IOBs	27	27	51	51	99	99
Crit. Path Delay (nS)	5.396	3.081	5.980	3.081	6.419	3.081
Max. Clk. Freq. (MHz)	187.76	324.57	167.22	324.57	155.79	324.57

On the other side, we have segregated the suggested ISA architecture into different stages by pipelining it. Now, this makes our architecture suitable for clock gating. In the proposed design, we have gated the clock signal that is fed into every stage. On doing this, the ideal stages of our architecture can be deferred from the clock switching which significantly reduces the power consumption. Such gating is valid only during the beginning and ending sessions of the addition process. On the starting of addition, later pipeline stages (towards the output side) of the design are ideal and these stages can be clock gated. Unlike towards the end of addition process, earlier stages (near the input side) of the design seem to be ideal and are clock gated.

For example: pipeline stages five, four, three and two are ideal while the process is being carried out in the first stage when the addition begins. Similarly, first stage will be ideal while rest keeps processing data when addition is towards the completion. However, while the adder is in-between the process of adding continuous stream of data then there is no point of gating the clock because all the stages are busy performing the operations.

In order to quantify the area occupied and power consumed by the proposed ISA, this work

includes the ASIC synthesis and post layout simulation results of 32-bit ISA architectures. Additionally, this process is carried out for non-pipelined 32-bit ISA as well for the purpose of comparison. To begin with, HDL coded architectures of 32-bit ISAs are synthesized with realistic design constraints using the standard cell library of UMC 90 nm-CMOS technology node. This process continues iteratively until a timing violation free gate level net list is generated. Thereafter, it is instantiated along with the input-output (IO) cells using the LEF (five metal layers) file of 90 nm CMOS process in the SoC Encounter platform of Cadence tool.

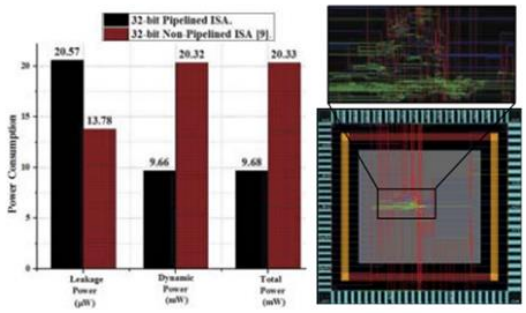


Fig7: Comparison plot of leakage, dynamic and total power consumed

These standard cells are place, power cum signal routed; clock tree synthesized and time analyzed, couple of times during the backend (physical) design process. Then, the chip-layout is design rule check (DRC), layout versus schematic (LVS), density checked and finally the switch activity file (SAF) and the net list is generated. Eventually, the final net list along with the test bench, SAF, standard parasitic exchange-format (SPEF) file are post-layout simulated to check the valid output of the layout and then calculate the total power consumed.

By 32-bit pipelined and non-pipelined ISA, when implemented in 90 nm-CMOS process and operated at 400 MHz clock frequency and chip layout of proposed 32-bit pipelined ISA with core area of 5111 μm2 when implemented in UMC 90 nm-CMOS technology node.

The proposed architectures could operate up to maximum clock frequency of 444.64 MHz with a positive slack of 251 pS. At the supply voltage of 0.9 V and a clock frequency of 400 MHz, the power consumed (including static plus dynamic powers) by suggested 32-bit pipelined and non-pipelined [9] ISAs are illustrated in Fig. 4. The pipelined ISA consumes total power of 9.68 mW at 400 MHz and it can be observed that this pipelined ISA consumes lesser power by 52.38% with respect to its peer 32-bit non-pipelined ISA, as a result of the clock gating technique incorporated in the design. On the other side, Table II shows the total area consumed as well as the standard cell counts (with their respective area consumption) of the 32-bit suggested ISA and reported non-pipelined ISA. It can be observed that there is a surge in the number of sequential and inverter cells of the proposed adder due to additional register levels used in the design. Thereby, the pipelined adder presented in this work has a area degradation of 40.7% in comparison with the non-pipelined adder. Eventually, the chip layout of proposed 32-bit pipelined ISA has been shown in Fig.

IV. SIMULATIONS AND RESULTS

Table 5.1.comparasion b/w Existing and Proposed adders

Method	area	power	delay
Existing	48%	142.44mw	14.89ns
proposed	12%	65.14mw	4.528ns

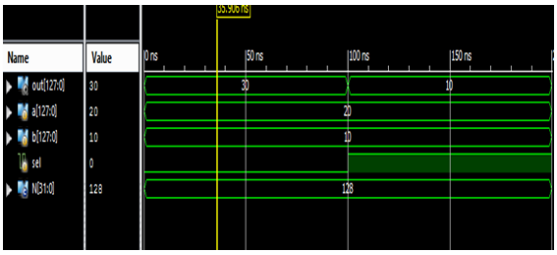


Fig. 8: simulation waveform for 128-bit adder/subtraction

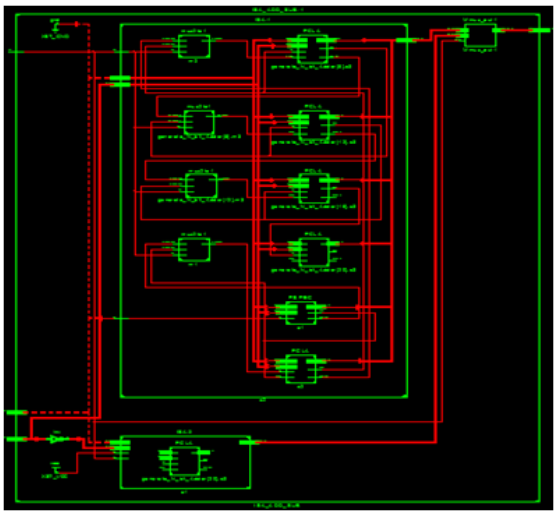


Fig. 9: RTL schematic

On-Chip	Power (W)	Used	Available	Utilization (%)
Logic	0.000	40	17600	0
Signals	0.000	89	--	--
IOs	0.000	62	230	27
Leakage	0.065			
Total	0.065			

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	4.0	84.7	25.3

Supply Summary	Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	1.000	0.005	0.000	0.005
Vccaux	1.800	0.006	0.000	0.006
Vcco18	1.800	0.001	0.000	0.001
Vccbram	1.000	0.000	0.000	0.000
Vccpint	1.000	0.020	0.000	0.020
Vccpaux	1.800	0.013	0.000	0.013
Vcco_ddr	1.500	0.002	0.000	0.002

Supply	Total	Dynamic	Quiescent
Power (W)	0.065	0.000	0.065

Fig. 10: power report

On-Chip Power Summary

On-Chip	Power (mW)	Used	Available	Utilization (%)
Clocks	0.00	0	---	---
Logic	0.00	40	17600	0
Signals	0.00	89	---	---
IOs	0.00	62	230	27
Static Power	65.18			
Total	65.18			

Fig. 11: On-Chip Power Summary

Cell:in->out	fanout	Delay	Delay	Logical Name (Net Name)
IBUF:I->O	5	0.000	0.561	a_1_IBUF (a_1_IBUF)
LUT6:I1->O	3	0.043	0.353	s1/s1/c3/cout1 (s1/s1/c_t<2>)
LUT5:I3->O	3	0.043	0.353	s1/s2/c1/cout1 (s1/s2/c_t<0>)
LUT5:I3->O	3	0.043	0.353	s1/s2/c3/cout1 (s1/s2/c_t<2>)
LUT6:I4->O	3	0.043	0.353	s1/generate_N_bit_Adder[8].s3/c1/cout1
LUT5:I3->O	3	0.043	0.353	s1/generate_N_bit_Adder[8].s3/c3/cout1
LUT6:I4->O	3	0.043	0.353	s1/generate_N_bit_Adder[12].s3/c1/cout1
LUT5:I3->O	3	0.043	0.353	s1/generate_N_bit_Adder[12].s3/c3/cout1
LUT6:I4->O	3	0.043	0.353	s1/generate_N_bit_Adder[16].s3/c1/cout1
LUT5:I3->O	2	0.043	0.433	s1/generate_N_bit_Adder[16].s3/c3/cout1
LUT5:I2->O	1	0.043	0.279	Mmux_out131 (out_20_OBUF)
OBUF:I->O		0.000		out_20_OBUF (out<20>)
Total		4.528ns	(0.430ns logic, 4.098ns route)	
			(9.5% logic, 90.5% route)	

Fig. 12: On-Chip delay Summary

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice LUTs	48	17600	0%
Number of fully used LUT-FF pairs	0	48	0%
Number of bonded IOBs	62	100	62%

Fig. 13: device utilization Summary

V. CONCLUSION

In this paper, we presented high-speed and low-power version of the contemporary ISA design. This architecture has been fine grain pipelined and clock gated to escalate speed and alleviate power consumption respectively. Experimental results showed that the suggested ISA could operate at 324.57 MHz and 444.64 MHz of maximum clock frequency in FPGA and 90 nm-CMOS ASIC platforms respectively. Subsequently at this technology node, it occupied 5111 μm2 of area and consumed 9.68 mW of total power at 400 MHz. Therefore, the proposed ISA can operate at 52% higher speed, needs 52.38% lower power and occupies 40.7% more area than the state-of-the-art ISA design. Thereby, such design would definitely play significant role in the design of contemporary as well as future electronic devices for IoE and many other contemporary applications. However, the area issue can be resolved to some extent by using lower technology nodes in the design process.

FUTURE SCOPE

This work has been designed for 64-bit and n-bit word size and results are evaluated for parameters like area, delay and power. This work can be further extended for higher number of bits. New architectures can be designed in order to reduce the power, area and delay of the circuits. Steps may be taken to optimize the other parameters like frequency, number of gate clocks, length etc.

BIBLOGRAPHY

1. Kildee Rawat, Tarek Darwish and Magdy Bayoumi, "A low power and reduced area Carry Select Adder", 45th Midwest Symposium on Circuits and Systems, vol.1, pp. 467-470, March 2002.
2. Y. Kim and L.-S. Kim, "64-bit carry-select adder with reduced area," Electron. Lett. vol. 37, no. 10, pp. 614- 615, May 2001.
3. J. M. Rabaey, Digtal Integrated Circuits-A Design Perspective.Upper Saddle River, NJ: Prentice-Hall,2001.
4. Cadence, "Encounter user guide," Version 6.2.4, March 2008.
5. R. Priya and J. Senthil Kumar, "Enhanced area efficient architecture for 128 bit Modified CSLA", International Conference on Circuits, Power and Computing Technologies, 2013.
6. Shivani Parmer and Kirat pal Singh,"Design of high speed hybrid carry select adder", IEEE, 2012.
7. I-Chyn Wey, Cheng-Chen Ho, Yi-Sheng Lin, and Chien-Chang Peng," An Area-Efficient Carry Select Adder Design by Sharing the Common Boolean Logic Term", Proceedings of the International MultiConference of Engineers and Computer Scientist 2012 Vol II,IMCES 2012,HongKong,March 14-16 2012.
8. B. Ramkumar and Harish M Kittur," Low-Power and Area-Efficient Carry Select Adder", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, VOL. 20, NO. 2, February 2012.
9. [8] Ms. S.Manjui, Mr. V. Sornagopae," An Efficient SQRT Architecture of Carry Select Adder Design by Common Boolean Logic",IEEE, 2013.