Engineering Universe for Scientific Research and Management
ISSN (Online): 2319-3069
Vol. XVI Issue V
May 2024

Implementation of ALU for Reconfigurable Logic

Pratibha Daheriya[1], Abhishek Agwekar[2]
Department of Electronics and Communication Engineering, TCST Bhopal[1,2]
daheriyapratibha@gmail.com[2]

Abstract

In this work we have implemented an arithmetic and logic unit for reconfigurable logic. Arithmetic and logic unit proposed in this design supports 8 operations including 4 arithmetic and 4 logic operations. Further we have proposed one more ALU supporting 15 instructions with the inclusion of BCD operations and rotate and shift operations. The target device is KINTEX 7, 28nm technology.

Keywords: *dynamic power consumption, FPGA, arithmetic and logic unit, reconfigurable logic.*

1. Introduction

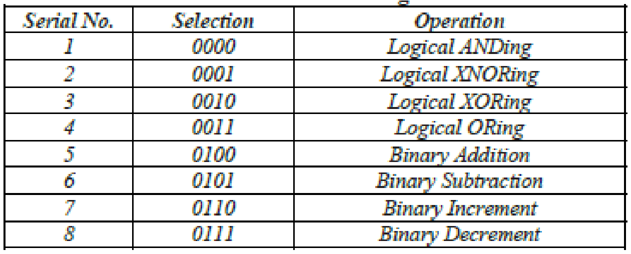
This is an era of hand held devices and equipments, most of these devices runs on battery, this puts a constraint on standby time, to increase standby time more and more battery life is needed, one way of solving this issue is to reduce power consumption of device or equipment. These days almost every device is intelligent, this intelligence came from using processors, and in forthcoming years this trend is likely to be increase. But these processors consume lot of the power of device as lot of switching activity is going inside. ALU (Arithmetic and Logic Unit) is the heart of any processor; this also consumes most of the processor power.

2. Conventional Eight Operation Design

The operations supported by proposed design are listed in table 1 and the block diagram of reference design is shown in figure 1. As shown in table 1 that the proposed design can perform 8 logical and arithmetic and logical operations. Eight units are used to implement the eight operations. At any given time instant only one of the eight units will be in use to perform one of the eight operations but clock signal CLK is assigned to all the eight units at all times, this increases the dynamic power consumption of the design.

Figure 2 shows the internal structure of logical AND operation. The AND operation is implemented by an array of AND gates followed by a D flip flop.

Table 1 Operations Supported by 8 Operations Conventional Design



Serial No.	*Selection*	*Operation*
1	*0000*	*Logical ANDing*
2	*0001*	*Logical XNORing*
3	*0010*	*Logical XORing*
4	*0011*	*Logical ORing*
5	*0100*	*Binary Addition*
6	*0101*	*Binary Subtraction*
7	*0110*	*Binary Increment*
8	*0111*	*Binary Decrement*

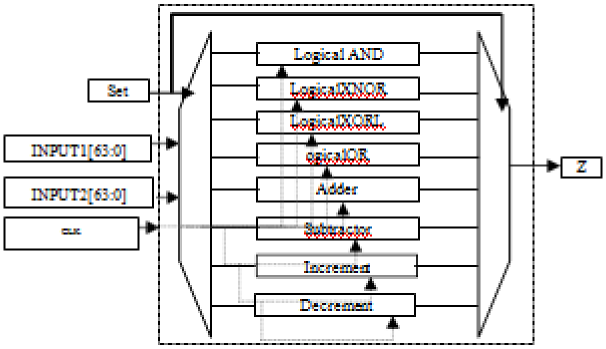


Fig. 1 Conventional 8 Operation Design - Internal Architecture

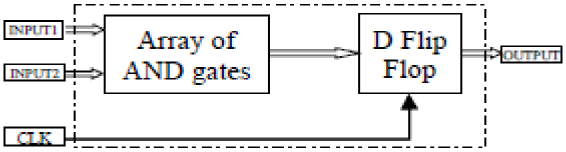


Fig 2: Internal Structure - Logical AND

Similarly internal structures of all the units are shown in figure 2 to figure 9.

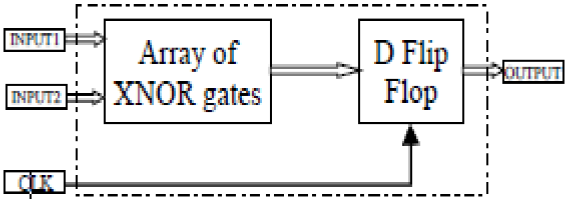


Fig 3: Internal Structure - Logical XNOR

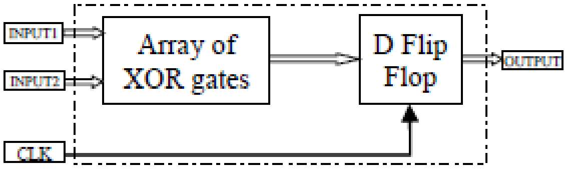


Fig 4: Internal Structure - Logical XOR

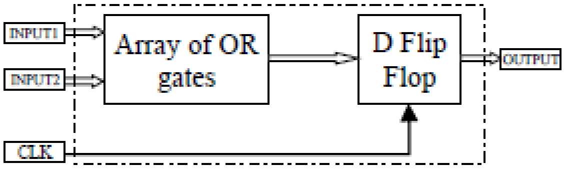


Fig 5: Internal Structure - Logical OR

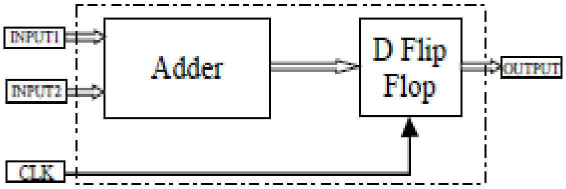


Fig 6: Internal Structure - Adder

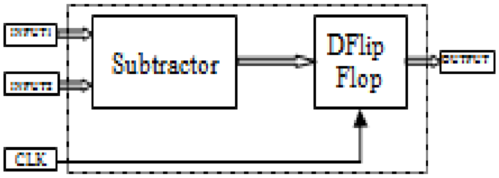


Fig 7: Internal Structure - Subtractor

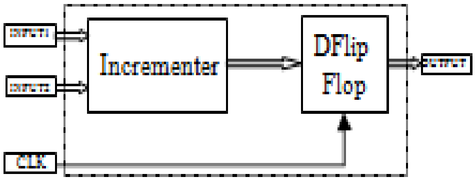


Fig 8: Internal Structure - Incrementer

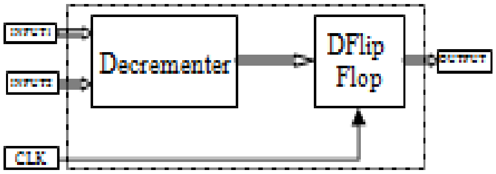


Fig 9: Internal Structure -Decrementer

3. Conventional Fifteen Operation Design

This section discusses the conventional ALU supporting fifteen operations, this ALU is based on the same design as the reference ALU supporting eight operations i.e. clock is available to all the units at all time and different units are used for performing addition, subtraction, increment and decrement. Here in this design we have incorporated 7 more instructions to perform shift, rotate and BCD operations. Table 2 shows the supported operations by this design.

Table 2 Reference ALU – 15 Instructions

Serial No.	*Selection*	*Operation*
1	*0000*	*Logical ANDing*
2	*0001*	*Logical XNORing*
3	*0010*	*Logical XORing*
4	*0011*	*Logical ORing*
5	*0100*	*Binary Addition*
6	*0101*	*Binary Subtraction*
7	*0110*	*Binary Increment*
8	*0111*	*Binary Decrement*
9	*1000*	*Rotate Right*
10	*1001*	*Rotate Left*
11	*1010*	*Shift Right*
12	*1011*	*Shift Left*
13	*1100*	*BCD Addition*
14	*1101*	*BCD Subtraction*
15	*1110*	*BCD Multiplication*
16	*1111*	*NOP*

Figure 110 shows the internal architecture of reference ALU –15 operations, no clock gating logic is used and separate blocks are used to perform addition, subtraction, increment and decrement.

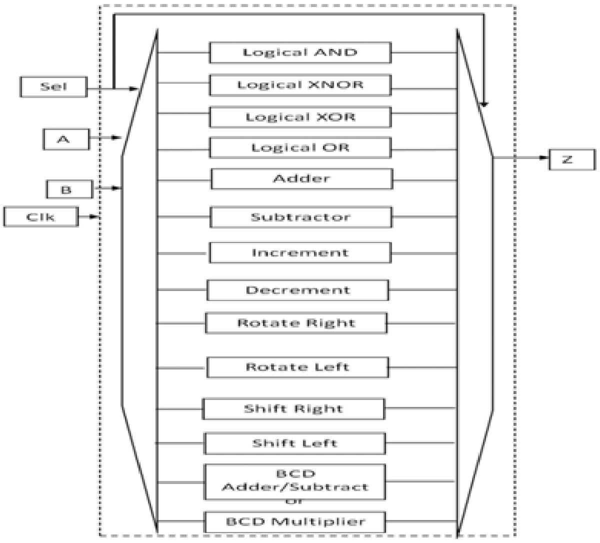


Fig. 10 Conventional 15 operation ALU – Internal Architecture

The rotate right operation is shown in figure 11, it consists of connections with input and out ports and 64 D flip flops (FD) as shown in figure. The first input signal I0 is connected to output signal O63, I1 to O0, I2 to O1 and so on. Similarly architectures of rotate and shift operations are shown in figure 12 – 14.

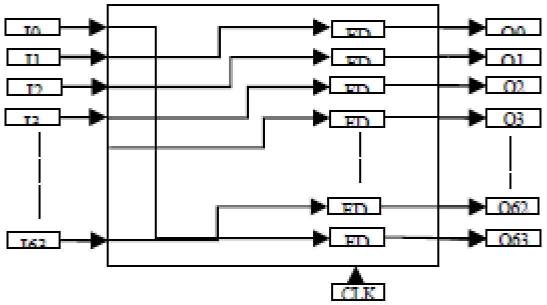


Fig. 11 Rotate Right – Internal Structure

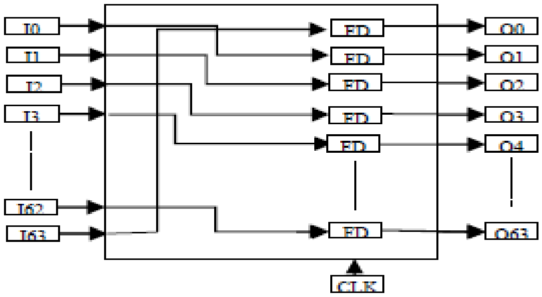


Fig. 12 Rotate Left – Internal Structure

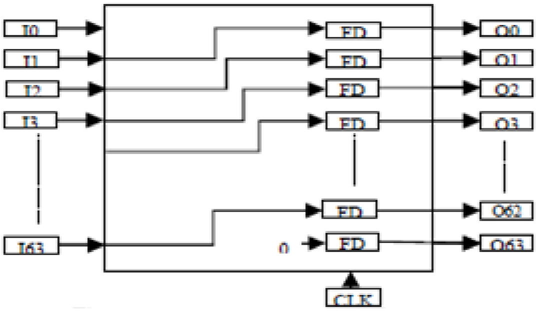


Fig.13 Shift Left – Internal Structure

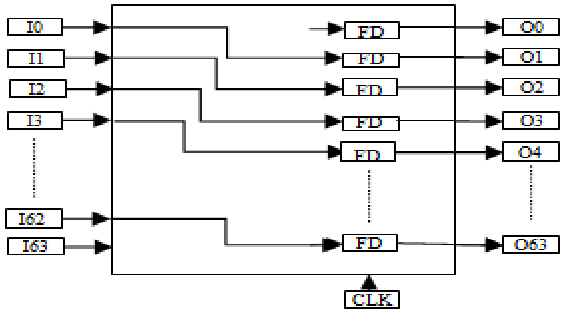


Fig.14 Shift Right – Internal Structure

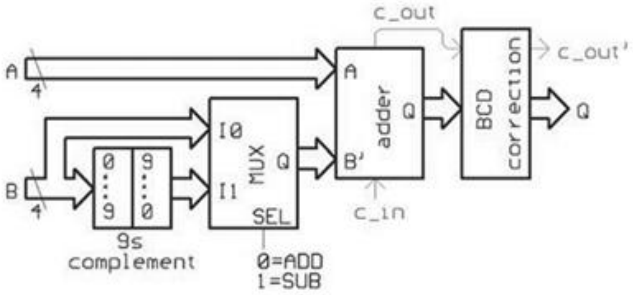


Fig.15 BCD Adder/Subtractor

The BCD adder/Subtractor unit shown in figure 15 consists of a binary adder with a BCD correction unit to perform the addition, to perform the subtraction also a 9'complement unit is also incorporated shown in figure 18. Here in figure 15 a multiplexer is also incorporated to select between the BCD addition and BCD subtraction. To perform BCD addition the data path is the simply the multiplexer, then the binary adder and then the BCD correction unit. To perform the subtraction. first the 9's complement of the second number is calculated using the 9'complement unit and the multiplexer passes this complemented number to the binary adder to perform the subtraction, the final result of BCD subtraction is obtained after the BCD correction has been performed.

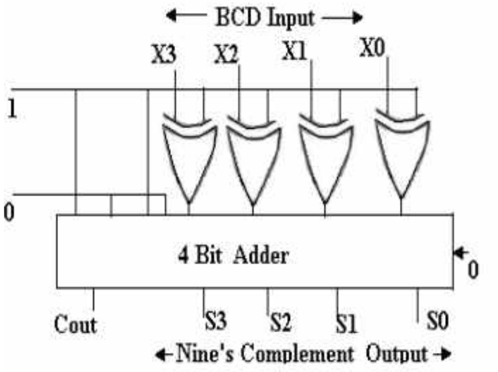


Fig.16 9's Complement Unit

The 9's complement is performed using 4 xor gates, which XORes the input number with 1 and a binary adder to perform the addition of 0110 with the xored number. Figure 17 represents an area optimized BCD digit multiplier. This multiplier produces the result of multiplication in binary and we need a binary to BCD converter shown in figure 18. The B is the higher nibble of the multiplication and C is the lower nibble of multiplication. Figure 19 shows the parallel multiplication process of a 4 x 4 BCD multiplier. Xi and yi are single digit BCD numbers. These numbers are multiplied using the single digit BCD multiplier shown in figures. pyixiH and pyixiL are higher and lower nibble of multiplication respectively. Figure 20 depicts the 4 x 4 multiplier architecture to implement the algorithm shown figure. In the process of floating point multiplication this 4x4 multiplier is extended to implement 16 x 16 multiplier.

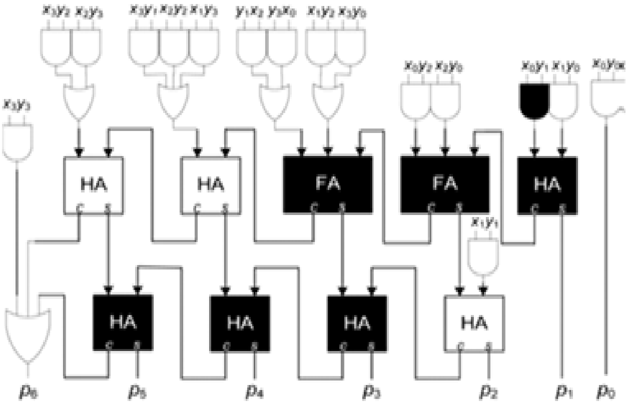


Fig.17 Area optimized – Single digit BCD Multiplier

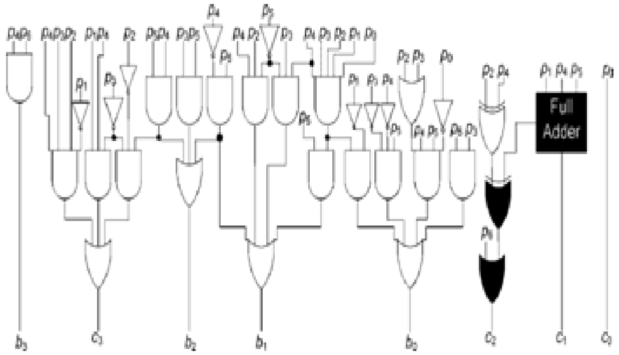


Fig.18 Binary to BCD Converter

				x3	x2	x1	x0
				y3	y2	y1	y0
				P0003L	P0002L	P0001L	p0000L
			P0003H	P0002H	P0001H	P0000H	
			P0103L	P0102L	P0101L	P0100L	
		P0103H	P0102H	P0101H	P0100H		
		P0203L	P0202L	P0201L	P0200L		
	P0203H	P0202H	P0201H	P0200H			
	P0303L	P0302L	P0301L	P0300L			
P0303H	P0302H	P0301H	P0300H				
P7	P6	P5	P4	P3	P2	P1	P0

Fig.19 Array BCD Multiplication Process

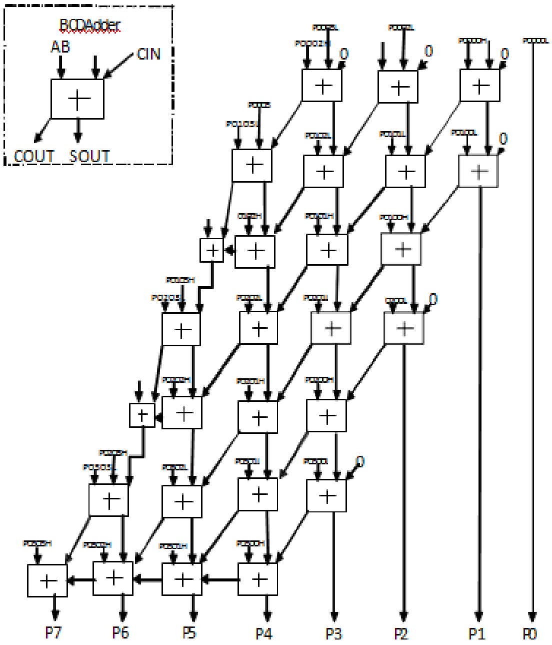


Fig.20 4 digit BCD Multiplier

4. Performance Analysis

Table 3 shows the resource and power consumption of reference ALU and area and power efficient ALU. The base paper design and the conventional design are same, we have implemented the base paper design [1] and named it conventional design.

Table 3 Performance analysis – 8 Operation designs

On Chip	Base Paper Design		Conventional	
	Power (mW)	Resource	Power (mW)	Resource
Clock	0.16	-	02	-
Logic	0.76	496	02	511
Signal	6.63	752	04	764
IOS	35.98	197	37	196
Static Power	45.32	-	45	-
Dynamic Power	43.53	-	44	-
Total	88.85	-	90	-

Table 4 shows the resource and power consumption of conventional ALU. The base paper design and the conventional design are same, we have implemented the base paper design and named it conventional design.

5. Conclusion

In this work we have successfully implemented an arithmetic and logic unit for reconfigurable logic. In this work two designs are implemented, first design is the conventional ALU supporting 8 operations. The second design is conventional design supporting 15 instructions, in this design we have increased the functionality of the base paper design by employing 7 more instructions making the design to support a total of 15 operations. The new 7 operations employs rotate right, rotate left, shift right, shift left, BCD addition, BCD subtraction and BCD multiplication.

References

[1] Shruti Murgai, “Energy Efficient And High Performance 64-bit Arithmetic Logic Unit Using 28nm Technology”, IEEE 2015.

[2] J. Shinde, and S. S. Salankar, “Clock gating-A power optimizing technique for VLSI circuits” Annual IEEE India Conference (INDICON), pp. 1-4, 2011.

[3] J. Castro, P. Parra, and A. J. Acosta, “Optimization of clock-gating structures for low-leakage high-performance applications”, Proceedings of IEEE International Symposium on Efficient Embedded Computing, pp. 3220-3223, 2010.

[4] V. Khorasani, B. V. Vahdat, and M. Mortazavi, ”Design and implementation of floating point ALU on a FPGA processor”, IEEE International Conference on Computing, Electronics and Electrical Technologies (ICCEET), pp.772-776, 2012.

[5] S. Cisneros, J. J. Panduro, J. Muro, and E. Boemo, “Rapid prototyping of a self-timed ALU with FPGAs”, International Conference on Reconfigurable Computing and FPGAs, pp. 26-33, 2012.

[6] B. S. Ryu, J. S. Yi, K. Y. Lee and T. W. Cho, ”A design of low power 16- bit ALU”, Proceedings of the IEEE TENCON Conference, pp.868- 871, 1999.

[7] T. Lam, X. Yang, W. C. Tang and Y. L. Wu; , "On applying erroneous clock gating conditions to further cut down power," Design Automation Conference (ASP-DAC), 2011 16th Asia and South Pacific , vol., no., pp.509-514, 25-28 Jan. 2011.

[8] B. Pandey and M. Pattanaik, “Clock Gating Awar Low Power ALU Design and Implementation on FPGA”, 2nd International Conference on Network and Computer Science (ICNCS), Singapore, April 1-2, 2013.

[9] E. Arbel, C. Eisner and O. Rokhlenko, "Resurrecting infeasible clockgating functions," Design Automation Conference, 2009. DAC '09. 46th ACM/IEEE , vol., no., pp.160-165, 26-31 July 2009.

[10] Thomas D. Burd, “Energy-Efficient Processor System Design”, Ph.D Thesis, University of California, Berkeley, 2001.

[11] Thomas D. Burd and Robert W. Brodersen, “Design Issues for Dynamic Voltage Scaling”, ISLPED 2000, Rapallo, Italy.

[12] Pouwelse, J., Langendoen, K., and Sips, H., “Energy priority scheduling for variable voltage processors”, ISLPED 2001, Huntington Beach, CA, USA.

[13] C. Lee, J. Lee, T. Hwang, and S. Tsai., “Compiler Optimization on Instruction Scheduling for Low Power”, 13th International Symposium on System Synthesis, ACM, September 2000.

[14] Parik A, Kandemir M, Vijaykrishnan N and Irwin M.J, “Instruction Scheduling Base on Energy and Performance Constraints”, Proceedings IEEE Computer Society Workshop VLSI, 27-28 April 2000.

[15] S. Cisneros, J. J. Panduro, J. Muro, and E. Boemo, “Rapid prototyping of a self-timed ALU with FPGAs,” in Proc. International Conference on Reconfigurable Computing and FPGAs, pp. 26-33, 2012.

[16] B. S. Ryu, J. S. Yi, K. Y. Lee, and T. W. Cho, ”A design of low power 16- bit ALU,” in Proceedings of the IEEE TENCON Conference, pp.868-871, 1999.

[17] J. Monteiro, J. Rinderknecht, S. Devadas and A. Ghosh, "Optimization of combinational and sequential logic circuits for low power using precomputation," Advanced Research in VLSI, 1995. Proceedings., Sixteenth Conference on , vol., no., pp.430-444, 27-29 Mar 1995.

[18] Frank Emnett, Mark Biegel, Power Reduction Through RTL Clock Gating, SNUG San Jose, 2000.

[19] Gary K. Yeap, Practical Low-Power Digital VLSI Design, Power , EE Times India, January 2008.

[20] John F. Wakerly, Digital Design Principles and Practices, Prentice Hall, 2005.

[21] Hubert Kaeslin, ETH Zurich, Digital Integrated Circuit Design from VLSI Architectures to CMOS Fabrication, Cambridge University Press, 2008.

[22] P.J. Shoenmakers, J.F.M. Theeuwen, Clock Gating on RT- Level VHDL , Proc. of the int. Workshop on logic synthesis, Tahoe City, CA, pp. 387- 391, June 7-10,1998.

[23] L. Benini, G. De Micheli, E. Macii, M. Poncino, and R. Scarsi, Symbolic Synthesis of Clock-Gating Logic for Power Optimization of Synchronous Controllers, ACM Trans. Des. Autom. Electron, Oct. 1999.

[24] Safeen Huda, Muntasir Mallick, Jason H. Anderson, Clock Gating Architectures For FPGA Power Reduction , FPL 2009.

[25] Vojin G. Oklobdzjja, Vladlmlr M. Stojanovic, Dejan M. Markovic, Nikola M. Nedovic, DIGITA L SYSTEM CLOCKING High- Performance and Low-Power Aspects, Wiley Interscience, U.S., 2003.

[26] Vishwanadh Tirumalashetty, Hamid Mahmoodi, Clock Gating and Negative Edge Triggering for Energy Recovery Clock, ISCAS 2007, New Orleans, LA, pp. 1141-1144, 2007.

[27] Bishwajeet Pandey, Jyotsana Yadav, M Pattanaik, Nitish Rajoria “Clock Gating Based Energy Efficient ALU Design and Implementation on FPGA” 2014 IEEE.