



A High Performance Binary to BCD Converter

A. Hari Priya¹

Assistant Professor, Dept. of ECE, Indur Institute of Engineering, and Technology, Siddipet, Medak, India¹

ABSTRACT: Decimal data processing applications have grown exponentially in recent years thereby increasing the need to have hardware support for decimal arithmetic. Binary to BCD conversion forms the basic building block of decimal digit multipliers. This paper presents novel high speed low power architecture for fixed bit binary to BCD conversion which is at least 28% better in terms of power-delay product than the existing designs.

KEYWORDS:Decimal Arithmetic, Binary to BCD Converter

I.INTRODUCTION

In commercial business and internet based applications, decimal arithmetic is receiving significant importance. Decimal arithmetic is important in many applications such as banking, tax calculation, insurance, accounting and many more. Even though binary arithmetic is used widely, decimal computation is essential. The decimal arithmetic is not only required when numbers are presented for inspection by humans, but also it is a necessity when fractions are being used. Rational numbers whose denominator is a power of ten are decimal fractions and most them cannot be represented by binary fractions. For example, the value 0.01 may require an infinitely recurring binary number. Even though the arithmetic is correct, but if binary approximation is used instead of an exact decimal fraction, results can be wrong.

This extensive use of decimal data indicates that it is important to analyse how the data can be used and how the decimal arithmetic can be defined. However, the current general purpose computers perform decimal computations using binary arithmetic. But the problem is that decimal numbers such as 0.3 cannot be represented precisely in binary. The errors that result on conversion between decimal and binary formats cannot be tolerated as long as precision is concerned. Since decimal arithmetic has gained wide importance in financial analysis, banking, tax calculations. Insurance, telephone billing, such errors cannot be tolerated. This in turn can be overcome by using a decimal Arithmetic and logic unit. The operations related to decimal arithmetic are typically slower, more complex and occupy more area and this leads to more power and less speed when implemented in hardware. Hence, enhancing speed and reducing area is the major consideration while implementing decimal arithmetic. In this paper we will be estimating the delay and power. Decimal Arithmetic is getting significant attention in commercial business and internet based applications, providing hardware support in this direction is henceforth necessary. Recognizing the prerequisite for decimal arithmetic has led to specifications for decimal floating point arithmetic to be added in the draft revision of the IEEE-P754 standard. Decimal arithmetic operations are usually sluggish and complex, its hardware occupies more area. They are characteristically implemented using iterative methods or lookup table based reduction schemes. This has led to the motivation behind improving BCD architectures, to enable faster and compact arithmetic. BCD is a decimal representation of a number directly coded in binary, digit by digit. For instance, the number $(9321)_{10} = (1001\ 0011\ 0010\ 0001)_{BCD}$. It is observed that each digit of the decimal number is coded in binary and then concatenated to form the BCD representation of the decimal number.

As mentioned that any BCD digit lies between [0, 9] or [0000, 1001], multiplying two BCD digits can outcome in numbers between [0, 81]. All the promising combinations can be represented in a 7-bit binary number when multiplied, $(81)_{10}$ or $(1010001)_2$ being the highest. In BCD multiplication where 4-bit binary multipliers are used to multiply two BCD numbers X and Y with digits, X_i and Y_j , respectively, a partial product P_{ij} is generated of the form $(p_6p_5p_4p_3p_2p_1p_0)_2$. Conversion of P_{ij} from binary to a BCD number B_iC_j where $\pi(X_i, Y_j) = 10B_i + C_j$ wants fast and also efficient BCD converters. The binary to BCD conversion is generally ineffective if the binary number is seems to be

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

very large enough. Hence the conversion can be ended in parallel for each partial product after each BCD digit is multiplied as shown in Figure.1 and the resulting BCD numbers after conversion can be added using BCD adders. Additional alternativewould be to compress the partial products of all binary terms in parallel and then convert them to BCD as done.

In this paper we introduce a new architecture for binary to BCD conversion of partial products which forms the core of decimal multiplication algorithms such as [7] and [8].

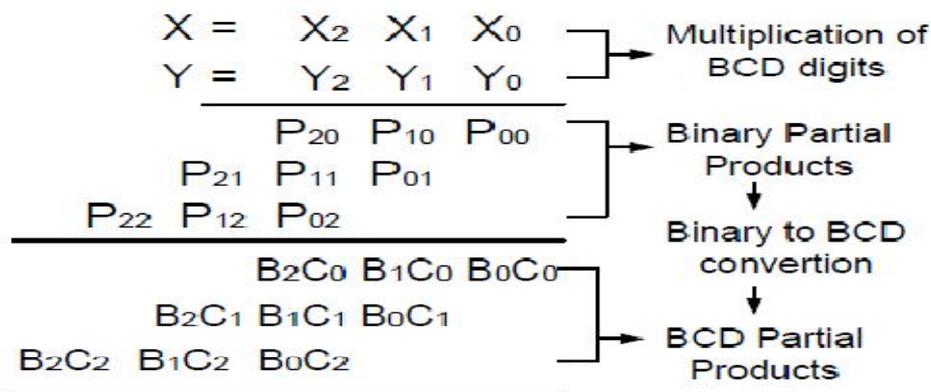


Figure 1: Illustration of BCD conversion in BCD

II.LITERATURE SURVEY

As pronounced in [1] BCD-digit multiplier can assist as the key building block of a decimal multiplier, regardless of the degree of parallelism. A BCD-digit multiplier creates a two-BCD digit product from two input BCD digits. We provide a novel design for the concluding;viewing some benefits in BCD multiplier implementations.

As described in [2] describes several methods of performing fast, efficient, binary-to-decimal conversion. With a modest amount of circuitry, an order of magnitude speed enhancement is achieved. This attainmentdeals a matchlessbenefit to general-purpose computers needing special hardware to interpret between binary and decimal numbering systems.

As described in [3], Couleur described a serial binary/ decimal conversion algorithm, the BIDEDEC method. It was a two-step methodconnecting a shift shadowed by a parallel modification of the data being converted. Now with the assistance of the integrated-circuit J-K flip-flop, the implementation of this two-step process needs an extreme amount of control logic. This paper presents a one-step conversion algorithm that is appropriate for binary-to-decimal and decimal-to-binary conversion. Anall-purpose design procedure for expressing the conversion registers as a present-state/next-state counter problem is given, along with numeroussamples of the application of the one-step algorithm. The advantages of this novel algorithm include low cost, faster operation, and also hardware modularity.

As described in [4] the risingappreciation of decimal computer arithmetic in scientific, commercial, financial and Internet-based applications, the only hardware realization of decimal arithmetic algorithms is in advance more importance. Thus Hardware decimal arithmetic units now help as an essential part of some newly commercialized general purpose processors, on the other hand complex decimal arithmetic operations, such as multiplication, have been recognized by somewhat slow iterative hardware algorithms. Nevertheless, with the fastimprovements in very large scale integration (VLSI) technology, semi- and fully parallel hardware decimal multiplication units are predictable to changeshortly. The foremost representation for decimal digits is the binary-coded decimal (BCD) encoding. The BCD-digit multiplier can help as the significant building block of a decimal multiplier, regardless of the degree of parallelism. A BCD-digit multiplier yields a two-BCD digit product from two input BCD digits. We make available a novel design for the former, showing some benefits in BCD multiplier implementations.



International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

As defined in [5] the, decimal multiplication is avital part ofthe financial, commercial, as well as internet-based computations. Here elementary building block of a decimal multiplier is a single digit multiplier. It takes two Binary Coded Decimal (BCD) inputs and provides a product that is in the range of [0, 81] characterized by only two BCD digits. A unique design for single digit decimal multiplication that role is toreduce the critical path delay and area is proposed in this investigation. On view of the among possible 256 combinations for the 8-bit input, only hundred combinations of them are seem to be valid BCD inputs. In the hundred valid combinations only four combinations need 4 times 4 multiplication, 64 combinations need 3 times 3 multiplication, and the remaining 32 combinations use either 3 times 4 or 4 times 3 multiplication. The proposed design makes use of this property. This design leads to more regular VLSI implementation, and does not have need of special registers for putting away easy multiples. So it is a fully parallel multiplier consuming only combinational logic, and is extended to a Hex/Decimal multiplier that provides either a decimal output or a binary output. The gathering of partial products produced using single digit multipliers is done by an array of multi-operand BCD adders for an (n-digit time’s n-digit) multiplication.

III.PROPOSED ALGORITHM

As we discussed earlier the main motto of the proposed algorithm in the paper is to achieve the more effective fixed bit binary to BCD conversion have been considered by relating with power., delay and area.as discussed in earlier work most of author worked on 7-bit binary to 8-bit/2-digit BCD converters, so we designed our proposed algorithm keeping in the mind above considerations to have extra benefit with the converters. Here we treated $p_6p_5p_4p_3p_2p_1p_0$ be seven binary bits that are to be converted into corresponding two parts; among them first one comprises the lower significant bits (LSBs) p_3, p_2, p_1 and p_0 while the second part contains the remaining higher significant bits (HSBs) p_6, p_5 and p_4 .Here (LSBs) has the similar weight as that of a BCD digit and that could be directly used to denote a BCD digit. Therefore only exclusion attains when $p_3p_2p_1p_0$ exceeds $(1001)_2$ or $(9)_{10}$. So finally we convert the LSBs into a valid BCD number we check whether $p_3p_2p_1p_0$ exceeds $(1001)_2$, and if it see to, we then add $(0110)_2$ to it.

So now we define this kind of adding $(0110)_2$ whenever the particular number exceeds $(1001)_2$ is named correction in BCD arithmetic. So obtained carry from this process is added immediately to the higher significant BCD digit calculated from the HSBs of the original binary number. Generally HSBs not only helps to the particular higher significant BCD digit rather to the lower significant BCD digit also. So these kind of major contributions of HSBs towards the lower significant digit are added subsequently BCD correction. The obtained resulting sum is then checked for the instance $(1001)_2$ and correction is done only if wanted to get the final lower significant BCD digit.

At this timeobtained carry from the above process is added sonly to the higher significant digit causing in the final higher significant BCD digit. We also gets the case when two BCD digits are multiplied only six combinations of p_6, p_5 and p_4 (HSBs) are possible, that are likely to be 000, 001, 010, 011, 100 and 101. At this situation each of these combinations have a different involvement towards the lower and higher significant BCD digits. So finally we can say that influence can be easily calculated by assessing the weights of the patterns which are $p_6 \times 2^7 + p_5 \times 2^6 + p_4 \times 2^5$. Contribution of each of these patterns towards the lower and higher BCD digits is shown in table.

Table 2: Contribution of HSBs

Higher Significant Bits (HSBs)	Higher Significant BCD Digit	Lower Significant BCD Digit
000	0000	0000
001	0001	0110
010	0011	0010
011	0100	1000
100	0110	0100
101	1000	0000

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

The below figure clearly shows an example of the algorithm for number $(111111)_2$ or $(63)_{10}$ or $(0110\ 0011)$ BCD.

By above discussions the maximum utilization of the process is limited and small numbers of consequences are feasible for conversion has been done while designing the given architecture to reduce delay, power as well as area. Figure.3 shows the proposed architecture.

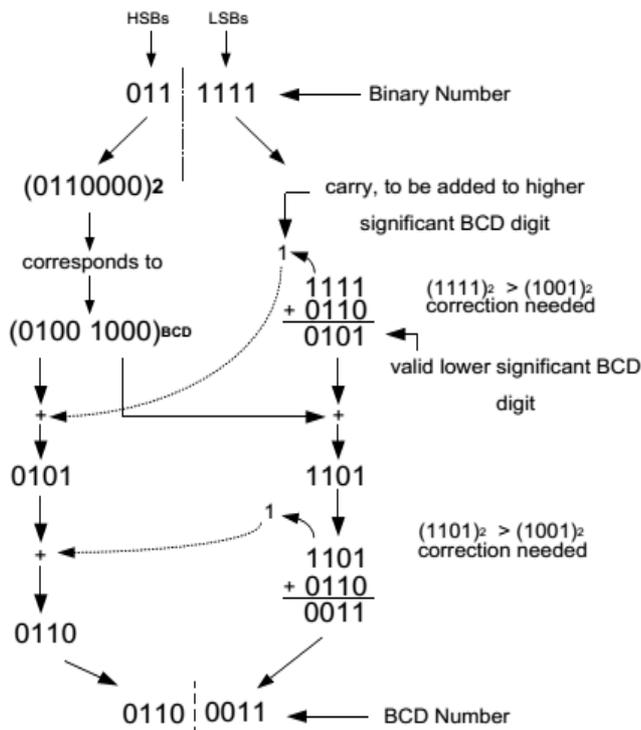


Figure 2: Proposed Algorithm for BCD conversion

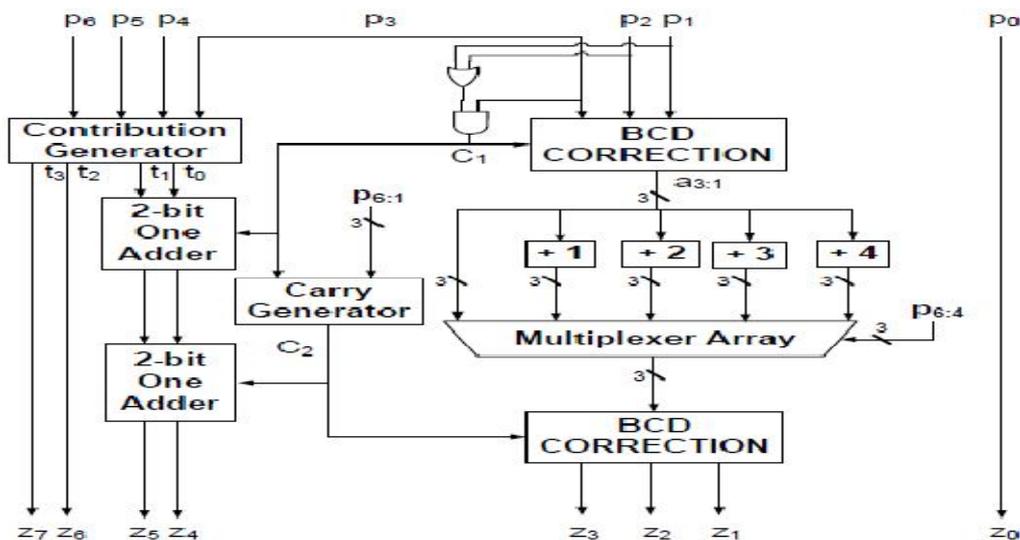


Figure 3: Proposed Architecture

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

$p_6 p_5 p_4 p_3 p_2 p_1 p_0$ are the binary bits that have to convert into corresponding BCD bits $z_7 z_6 z_5 z_4 z_3 z_2 z_1 z_0$. At the same time p_6 , p_5 and p_4 are the HSBs while p_3 , p_2 , p_1 and p_0 are the LSBs. z_0 is same as p_0 and hence therefore no operation is performed on p_0 . $\{p_3, p_2$ and $p_1\}$ are used to check whether the present LSBs are greater than $(1001)_2$ or not using equation (1) and are directed to the BCD Correction block.

$$C_1 = p_3 \cdot (p_2 + p_1) \dots\dots\dots (1)$$

Whenever C_1 is high, BCD Correction block immediately adds 011 to the input bits. Figure.4 shows the implementation of BCD Correction block.

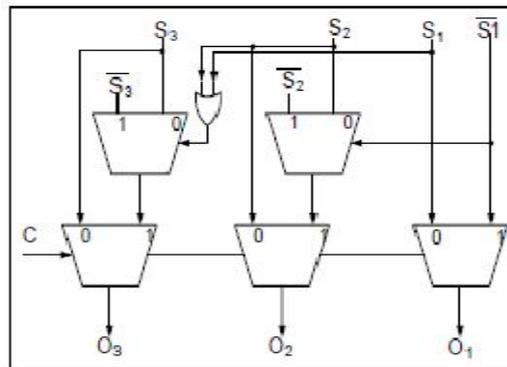


Figure.4: BCD Correction

In analogous, HSBs beside with p_3 are fed to a given simple logic block recognized as Contribution Generator which yields the higher significant BCD digits. The logic implemented by the Contribution Generator is as follows

$$t_3 = p_6 p_4 \dots\dots\dots (2)$$

$$t_2 = p_5 (p_4 + p_3) + p_6 p_4 \dots\dots\dots (3)$$

$$t_1 = (p_5 + p_6) p_4 \dots\dots\dots (4)$$

$$t_0 = p_6 p_5 p_4 + p_5 p_4 \dots\dots\dots (5)$$

C_1 is the carry from the lower significant digit, so it is added to the higher significant digit $t_3 t_2 t_1 t_0$. It is observed that very few cases lead to the propagation of the incoming carry from t_1 to t_2 . Hence, we take benefit of this situation and also device $\{t_3, t_2\}$ in combinational logic thus eliminating the need to add C_1 to these terms, thus saving hardware and complexity thus 2-bit One Adder, as presented in Figure.5, that is used to add C_1 to t_0 and t_1 . Now there is an option of a carry generation, when the contributions of HSBs are added to the corrected LSBs (a_3, a_2 and a_1). This carry is calculated ahead of time by a Carry Generator block using C_1 and input bits p_6 to p_1 . The logic implemented by Carry Generator is given by the equation below

$$C_2 = C_1 (p_4 (p_3 + p_2) + p_3 p_5) + p_6 p_3 + p_4 p_3 p_1 \dots\dots\dots (6)$$

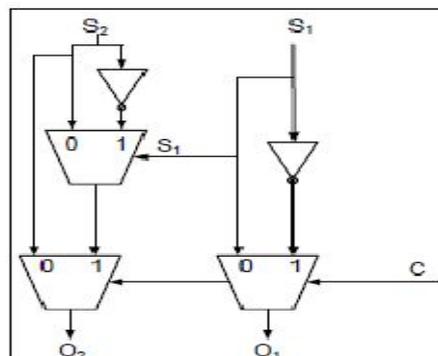


Figure.5 : 2-bit One Adder

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

C_2 is similarly added to result of the principal 2-bit One Adder using alternative 2-bit One Adder and the final higher significant digit is Higher Significant obtained. $\{t_3$ and $t_2\}$ are equivalent to z_7 and z_6 respectively and are directly accessible from the above Contribution Generator block.

Now contribution of HSBs towards lower significant BCD digit is static and distinctive and is known once HSBs are known. We have implemented four distinct adder units which add only stated values to the inputs in parallel according to the contributions in Table. On the other hand different adder blocks, +1, +2, +3 and +4 (shown in Figure.6) add 001, 010, 011 and 100 to the input bits correspondingly. So the Adder blocks take the corrected LSBs (a_3, a_2, a_1) as inputs and add particular numbers to them. The suitable result is then acquired through a multiplexer whose selection bits are p_6, p_5 and p_4 (HSBs). Then finally result from the multiplexer is then fed to BCD Correction block which takes C_2 as input then decides whether correction has to be done or not. Now the results obtained from the BCD Correction block are z_3, z_2 and z_1 which, along with z_0 , form the final lower significant BCD digit.

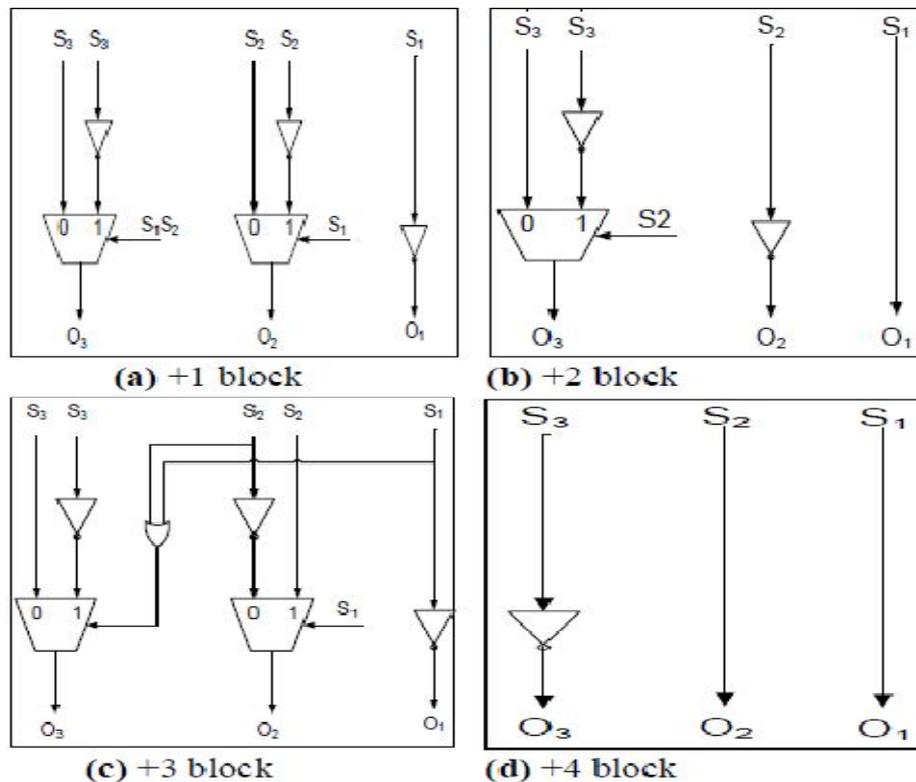


Figure.6: +1, +2, +3, +4 adder blocks

IV . RESULT AND DISCUSSION

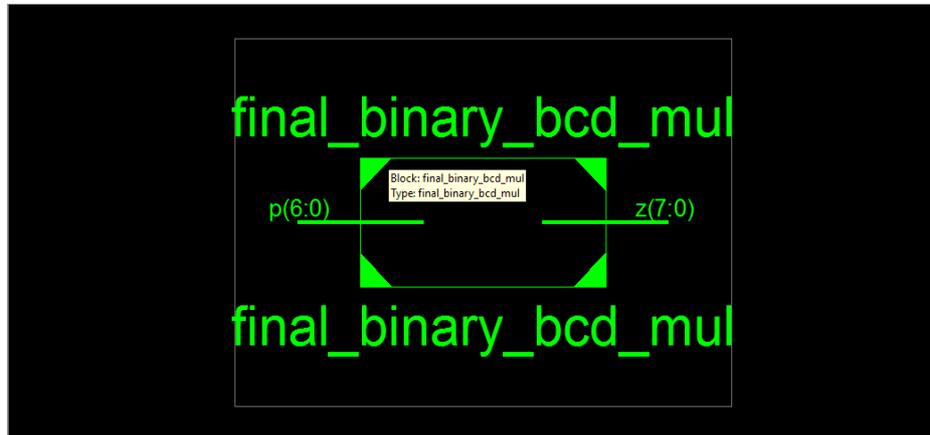
The results of the above method have been simulated with the help of Xilinx tool and analysed it and implemented its RTL Schematic using corresponding part number and shows that with the specialized converter overwhelming features that led better conversion in case of area, power and delay.

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

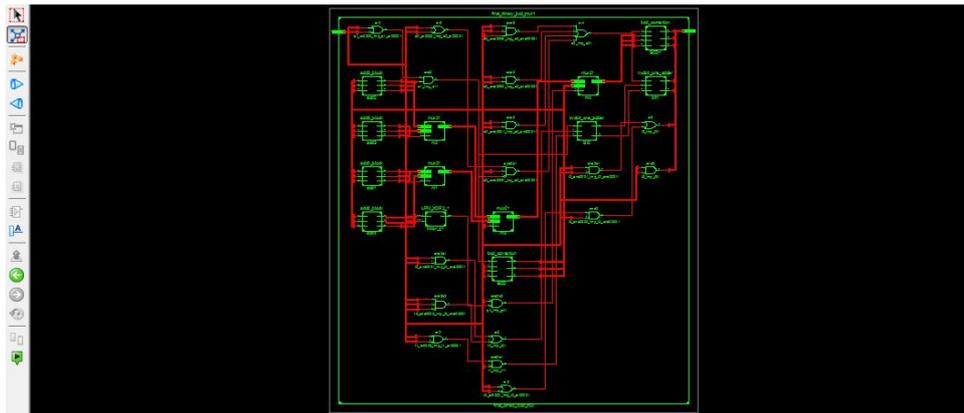
(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

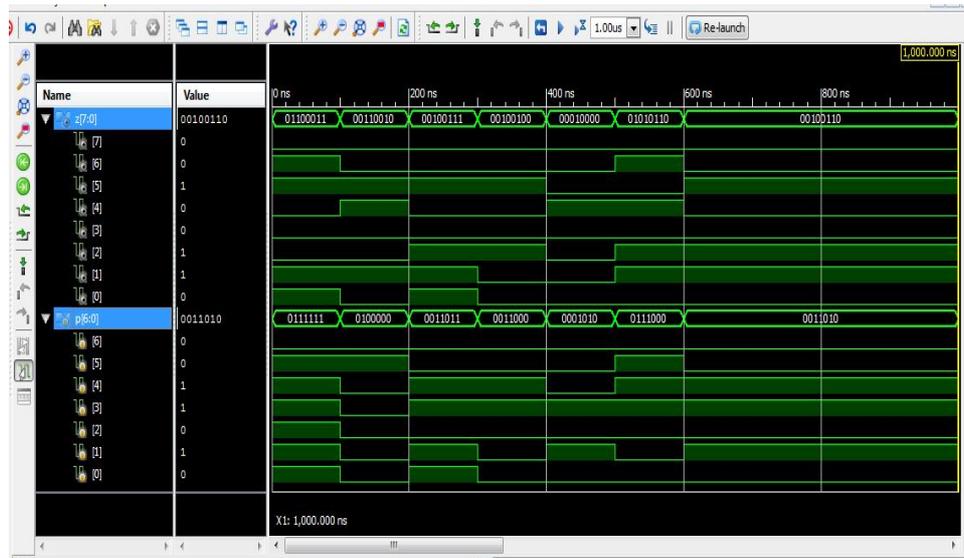
RTL Schematic:



Internal Schematic:



Simulation:





International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

VI.CONCLUSION

We discussed that the architectures have been designated using Verilog HDL. Corresponding Delay, power and area values for the designs are acquired by synthesizing the Verilog HDL description language. The proposed converter is very flexible that can be easily plugged into any homogeneous multiplication architectures to accomplish better performance regardless of the method employed to generate binary partial products.

REFERENCES

1. Arazi, B., and Naccache, D.: „Binary-to-decimal conversion based on the 28 2 1 by 5”, Electron. Lett, 1992.
2. Schmookler, M.: „High-speed binary-to-decimal conversion”, IEEE Trans. Comput., 1968.
3. Rhyne, V.T.: „Serial binary-to-decimal and decimal-to-binary conversion”, IEEE Trans. Comput., 1990.
4. Binary-coded decimal digit multipliers Jaberipur, G.; Kaivani A. Computers and Digital Techniques, IET Volume 1, Issue 4, July 2007.H. Khalife, N. Malouch, S. Fdida, “Multihop cognitive radio networks: to route or not to route,” IEEE Network, vol. 23, no. 4, pp. 20-25, 2009.
5. Decimal multiplication using compact BCD multiplier James, R.K.; Shahana, T.K.; Jacob, K.P.; Sasi, S.; Electronic Design, 2008. ICED 2008. International Conference on 1-3 Dec. 2008.
6. Decimal multiplication using compact BCD multiplier James, R.K.; Shahana, T.K.; Jacob, K.P.; Sasi, S.; Electronic Design, 2008. ICED 2008. International Conference on 1-3 Dec. 2008 Page(s):1 – 6.
7. Binary-coded decimal digit multipliers Jaberipur, G.; Kaivani, A. Computers and Digital Techniques, IET Volume 1, Issue 4, July 2007 Page(s):377 - 381.
8. Novel High-Speed Architecture for 32-Bit Binary Coded Decimal (BCD) Multiplier Veeramachaneni, S.; Srinivas, M.B. Communications and Information Technologies, 2008. ISCIT 2008. International Symposium on 21-23 Oct. 2008 Page(s):543 – 546.

BIOGRAPHY



A. Hari Priya received BTech (ECE) in Malla Reddy College of Engineering and Techn Maisammaguda, Dhulapally, Kompally, Medchal in the year 2009. And MTech (VLSI System Design) in Hyderabad Institute of Tech. and Mangement at Gowdevally (v), Medchal (M), in the year 2011. She is currently working as Assistant Professor, Dept. of ECE, Indur Institute of Engineering. And Technology, Siddipet, Medak (D) India.