# A Novel, High Performance and Power Efficient Implementation of Decimal to BCD Converter using 90nm Hybrid PTL/CMOS Technique

Navya Rajput, Manan Sethi, Karna Sharma,Paanshul Dobriyal,Geetanjali Sharma

Abstract— Computing and electronic systems today process data majorly in decimal format. Applications involving data processing in decimal format are proliferating day by day. The cornucopia of demands for decimal data processing have intern pressed the issue concerning the hardware requirements for the binary to BCD conversion which lays down the basics for decimal multipliers. This paper presents four different designs for a binary to BCD converter using two different concepts with each concept implemented using 90nm hybrid PTL/CMOS and 90nm CMOS technology. The power consumption of all four designs were evaluated on Tanner EDA tool 13.0. The power consumption was calculated for all the four designs at 100Mhz and 50Mhz for three voltages and at three different temperatures. The 90nm Hybrid PTL/CMOS implementation of proposed design outperforms other three implementations not only in terms of power consumption but also in terms of number of transistors used. The number of transistors used for 90nm Hybrid PTL/CMOS implementation of proposed design were reduced to 674 as compared to 90 nm CMOS implementation of conventional design which required 1880 transistors and it is 68.11% more power efficient as compared to conventional design.

**Index Terms**— BCD, CMOS ,decimal, delay,Hybrid PTL/CMOS ,PDP, Power Consumption.

## 1 Introduction

The intense use of decimal arithmetic in the analog and digital electronics ,business and internet applications necessitated the evolution of efficient hardware support in this direction. Most of the operations involving decimal numbers are not only complex and sluggish their implementation also requires gargantuan transistors. Binary to decimal converter's massive use in today's advance and dynamic electronics industry along with their slow and high power consuming circuits has led to the motivation behind improving BCD architectures, to enable speedy arithmetic operation.

This paper is divided into six sections. Section 1 gives an overview of the current use of decimal to BCD converters. Section two describes the basics of Binary coded decimal (BCD). Section 3 includes the explanation of earlier implemented and currently used converters and block diagram of a conventional decimal to BCD converter. Section 4 contains the introduction of proposed design, its implementation and algorithm used and its block diagram. This section also includes categorical explanation of each component incorporated in the proposed design. Section 5 includes the comparison results for all the four designs and quantitative description of the comparisons. Section 6 has the conclusion and lastly we included the references helped and motivated us to introduces this power and hardware efficient converter design.

## 2 Binary Coded Decimal

BCD is a form of binary representation of decimal numbers where each decimal digit is represented by a fixed number of bits, usually four or eight, sometimes special bit patterns are used for a sign or for other indications. For example, the number $(6236)_{10}$ =$(0110\ \ 0010\ \ 0011\ \ 0110)_{BCD}$ . In this example it is clear that each number in decimal is coded in four bit binary format and then combined to form a BCD number. The range of BCD numbers is [0, 9] so the multiplication of two BCD numbers can yield number in range [0, 81]. The representation of 81 in decimal is $(1010001)_2$ so we need 7 bits for the representation of 81 in decimal format. 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 € ($X_i$, $Y_j$) = $10B_i + C_j$ needs fast and efficient BCD converters. The binary to BCD conversion is generally inefficient if the binary number is very large. Hence the conversion can be done in parallel for every 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. Another alternative would be to compress the partial products of all binary terms in parallel and then convert them to BCD as done in [8]. Figure1 shows the multiplication of two BCD number which results into partial products that are converter into BCD format using binary to BCD converter.

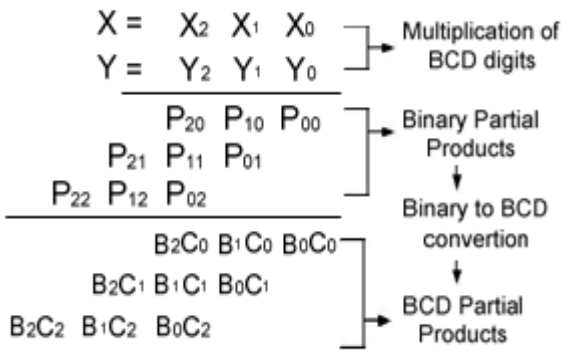


Fig. 1: Illustration of BCD conversion in BCD

dition. G. Jaberipur and A. Kaivani in [9] discuss the technique for reduction and partial product generation .There are several different n bit binary to BCD conversion schemes [10,11,12] that were introduced earlier but all of these suffered from common drawbacks of high latency, hardware requirements and power consumption. A series of BCD multipliers have been proposed recently which used fixed bit binary to BCD conversion. As shown in figure 1 the underlying principle to convert the 7 bit partial product $P_{ij} = P_0P_1P_2P_3P_4P_5P_6$ to its corresponding 8 bit BCD number $B_iC_i$ where Ci represents the lower BCD digits and Bi represents the higher BCD digit, has been done by various novel conversion architectures[7, 8] .

## 3 Hybrid PTL/CMOS Implementation of Conventional Binary to BCD Converter

### 3.1 Hybrid PTL/CMOS Technique

Pass transistor is the logic design in which the primary inputs drive the gate terminals and source-drain terminals in contrast to static CMOS where primary inputs drive gate terminals. Source side of logic transistor networks is connected to some input signals instead of the power lines. One of the main advantages of this logic style is that only one pass transistor network (either NMOS or PMOS) is sufficient to perform the logic function thereby using PTL one can save on the hardware requirements for the implementation of any circuit. Inverters are usually attached to the gate output to provide acceptable output driving capabilities.

Though PTL logic style reduces the number of transistors used for the implementation of any logic function it does not give full swing as in the case of cmos , every PTL logic design must realize a multiplexer structure in addition to these two drawbacks layout designing of PTL logic style is not straight forward and efficient. To overcome these pitfalls in the PTL logic style instead of using purely PTL logic style we use hybrid PTL/cmos logic which incorporates the advantages of both CMOS and PTL logic styles.

### 3.2 Hybrid PTL/CMOS Technique

Profound studies and work have already been done in the field of decimal to BCD. Schulte et al. [2, 3] have demonstrated different architectures for BCD multiplication. Techniques for efficient partial product generation using a recording scheme for signed magnitude partial products were introduced by Schwarz and Schulte [2]. Data independent optimization techniques which help in reducing the average latency of implementing arithmetic were proposed by Erle and Schulte [3] . The methods using semi parallel, fully parallel and serial decimal multiplication are available in lectures[4,9]. In order to use radix 10 based multiplication scheme, one needs to generate BCD partial products followed by BCD multi-operand ad-

Fig 2 shows the block diagram of a conventional binary to BCD converter. The diagram requires 6 BCD adders(brown) and a half adder(orange), the inputs stated 0 in the block diagram are ground connections. The input to the block diagram is a 7 bit binary representation of a decimal number shown as $p_0p_1p_2p_3p_4p_5p_6$ and a 8 bit output is obtained from two BCD adders a shown in the block diagram. For example if we input $(1111110)_2$ that is $63_{10}$ we will get the output as $(0110\text{-}0011)_{BCD}$ , in other words this converter converts the two digits of the decimal number individually into 4 bit binary number . BCD adder adds $6_{10}$ that is $(0110)_2$ when the input digit is greater than or equal to $9_{10}$ that is $(1001)_2$.

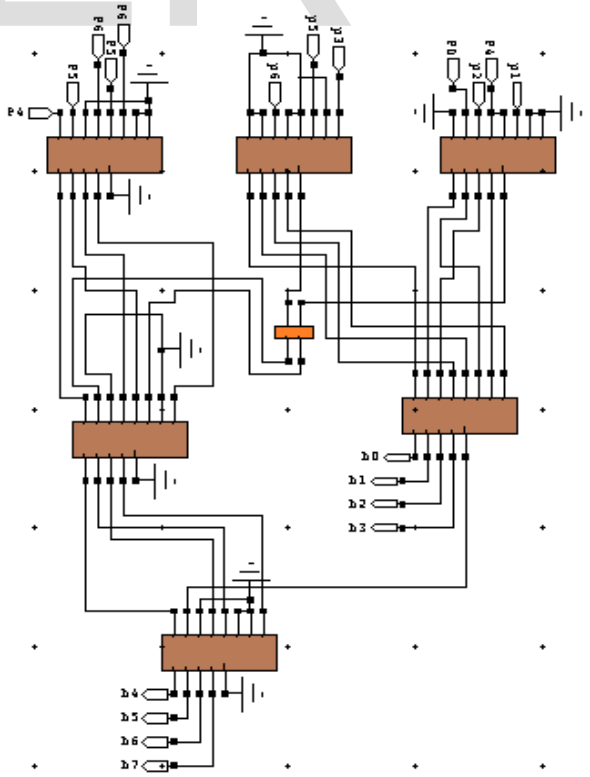

Fig 2: Block Diagram of conventional converter

## 4 HYBRID PTL/CMOS IMPLEMENTATION OF PROPOSED DESIGN

The main objective of the proposed design is to reduce the power consumption of the conventional binary to BCD converter and to reduce the hardware requirements as the proposed design incorporates considerable large number of transistors.

Let p be a 7 bit input ($p_0p_1p_2p_3p_4p_5p_6$) in binary format which is to be converted into BCD format. This input can be seen as a combination of lower significant bits(LSBs) and high significant bits(HSBs). Lower significant bits are $p_0$,$p_1$,$p_2$ and p3 and higher significant bits are $p_4$, $p_5$ and $p_6$ the weight of lower significant bits is same as that of a BCD digit and can be used to represent a BCD digit. Correction in BCD arithmetic is carried out whenever LSBs that is $p_0p_1p_2p_3$ exceeds $(1001)_2$ . The carry obtained from this procedure is added to the higher significant BCD digit calculated from the HSBs of the original binary number. HSBs contribute towards both lower significant BCD digits and higher significant BCD digits. After the BCD correction this contribution of higher significant bits towards lower significant bits is added. The resulting sum is then checked for the case $(1001)_2$ and correction is done if needed to obtain the final lower significant BCD digit. A possible carry from the above operation is added to the higher significant digit resulting in the final higher significant BCD digit.

For the multiplication of two BCD digits 6 combinations of $p_6$, $p_5$ and $p_4$ are possible which are binary representation of decimal values from 0 to 7 respectively. Each of these combinations have a different contribution towards the lower and higher significant BCD digits. By evaluating the weights of the patterns $p_6x2^7 + p_5x2^6 + p_4x2^5$ the contribution of these combinations towards LSBs and HSBs can be calculated. These contributions are stated in table 1.

Table 1
Contribution of HSBs

| Higher Significant Bits (HSBs) | BCD Weight | |
|---|---|---|
| | 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 |

Fig 4 describes the proposed binary to BCD converter which incorporates a contribution generator(pink), carry generator(yellow),two one bit adders(green),BCD adder and multiplexer(brown) and two inverters(black) .The proposed architecture is based on the fact that only limited and small numbers of outcomes are possible for conversion. The diagram categorically explains the components requirements of the design which includes two BCD correction circuits ,one contribution generator, two 2 bit adders , one carry generator and a multiplexer circuit.

Let $p_0p_1p_2p_3p_4p_5p_6$ the decimal input to the circuit where $p_0p_1p_2$ $p_3$ constitute the LSBs and $p_4p_5p_6$ contribute the HSBs of the input. The input bit $p_0$ is fed directly to the output forming $z_0$ and hence no operation is done on $p_0$. {$p_3$, $p_2$ and $p_1$} are u++-sed to check whether the LSBs are greater than $(1001)_2$ or not using equation (1) and are sent to the BCD Correction block.

$$C_1 = p_3.(p_2 + p_1) \quad -(1)$$

Whenever $C_1$ is high, BCD Correction block adds 011 to the input bits. Figure 5 shows the implementation of BCD Correction block.

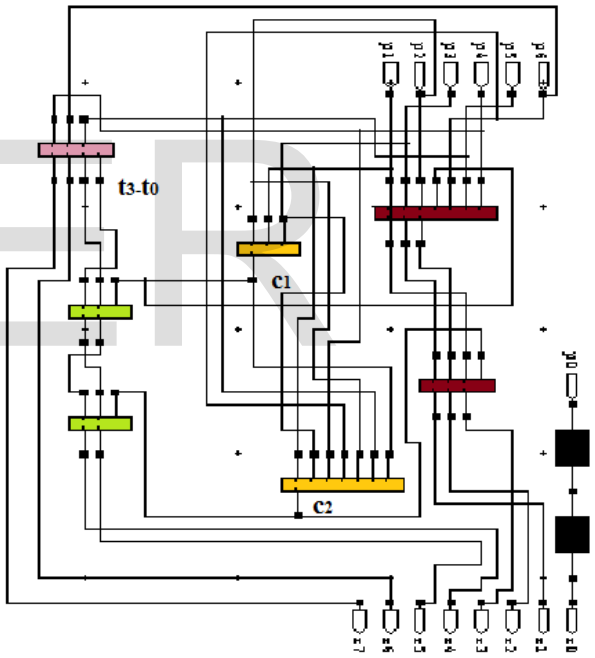


Fig 4: Block diagram for proposed design

In parallel, HSBs along with $p_3$ are fed to a simple logic block known as Contribution Generator which produces the higher significant BCD digits. The implementation of Contribution generator is shown is explained through following equation.

$$t_3 = p_6p_4 \quad (2)$$

$$t_2 = p_5(p_4+p_3)+p_6p_4 \quad (3)$$

$$t_1 = (p_5+p6)p4 \quad (4)$$

$$t_0 = p_6p5p_4+p_5p_4 \quad (5)$$

$C_1$ is the carry from the lower significant digit, so it is added to the higher significant digit $t_3t_2t_1t_0$. It is found that very few cases lead to the propagation of the incoming carry from $t_1$ to

$t_2$. Hence, we take advantage of this situation and implement {t3, t2} in combinational logic thus removing the need to add $C_1$ to these terms, thus saving hardware and complexity. 2-bit One Adder, is used to add $C_1$ to t0 and t1. There is a possibility 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 beforehand 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 $C_2 = C_1 (p_4 (p_3+p_2) + p_3p_5) + p_6p_3 + p_4p_3p_1$ . C2 is also added to result of the first 2-bit One Adder and the first higher significant digit is obtained. {$t_3$ and $t_2$} are equal to $i_7$ and $i_6$ respectively and are directly available from the Contribution Generator block.

Contribution of HSBs towards lower significant BCD digit is fixed and unique and is known once HSBs are known. We have implemented four distinct adder units which add only specified values to the inputs in parallel according to the contributions in Table 1. The different adder blocks, +1, +2, +3 and +4 are shown in Fig 5where *pink blocks* represent the ports, *black blocks* represent the inverters, *brown blocks* represent the multiplexers, *blue box* represents Nand gate and *purple box* represents Or gate.

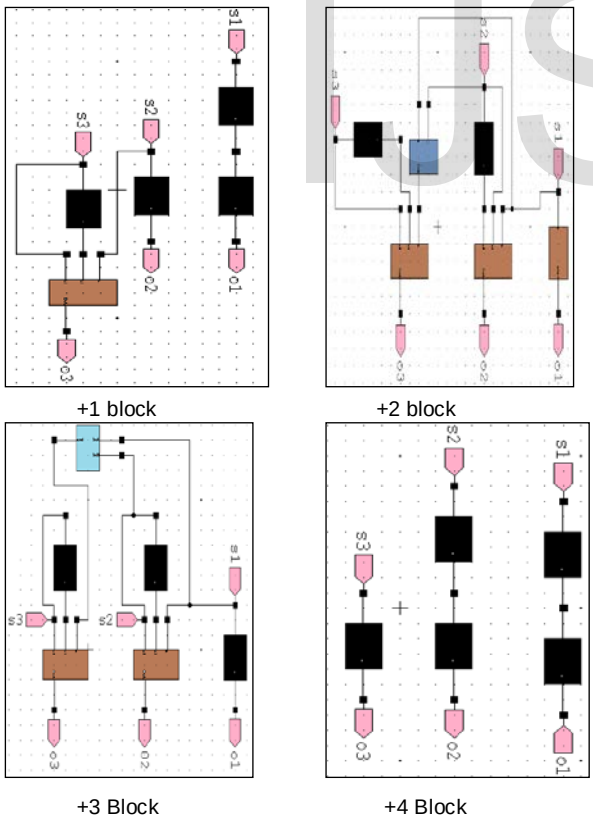


**Fig 5: Adders**

Adder blocks take the corrected LSBs ($a_3$, $a_2$, $a_1$) as inputs and add specific numbers to them. The appropriate result is then obtained through a multiplexer whose selection bits are $p_6$, $p_5$ and $p_4$ (HSBs). The result from the multiplexer is then fed to BCD Correction block which takes $C_2$ as input to decide whether correction has to be done or not. The results obtained from the BCD Correction block are $i_3$, $i_2$ and $i_1$ which, along with $i_0$, form the final lower significant BCD digit.

## 5. Simulation and Results

All the designs were implemented on Tanner EDA Tool 13.0. Power calculations for all four implementations were carried out at three different voltages(0.8v ,1v and 2v), two frequencies(100mhz and 50mhz) and three different temperatures (10°c , 20°c and 30°c). At 100Mhz, 10°c and 0.8v 90nm CMOS implementation of conventional design is 25.53% more power efficient as compared to its 90nm Hybrid PTL/CMOS implementation, 90nm CMOS implementation of proposed design is 63.18% more power efficient as compared to 90nm CMOS implementation of conventional design further 90nm Hybrid PTL/CMOS implementation of proposed design is 13.38% more power efficient than its 90nm CMOS implementation. Comparing the most power efficient implementations of both the designs that is 90nm CMOS implementation of conventional design and 90nm Hybrid PTL/CMOS implementation of proposed design at 100Mhz and 10°c we observed that 90nm Hybrid PTL/CMOS implementation of proposed design is 68.11% more power efficient than 90nm CMOS implementation of conventional design. At 50 Mhz,1v and 20°c 90nm CMOS implementation of conventional design is 22.84% more power efficient as compared to its 90nm Hybrid PTL/CMOS implementation, 90nm CMOS implementation of proposed design is 60.50% more power efficient as compared to 90nm CMOS implementation of conventional design further 90nm Hybrid PTL/CMOS implementation of proposed design is 12.54% more power efficient than its 90nm CMOS implementation. The comparisons for all the calculations are explicitly stated in the following charts. Table2 contains the values obtained at 100Mhz ad Table 1 show the values obtained at 50 Mhz.

Fig 6 shows comparison results for all four implementations at 100Mhz, 10°c for 0.8v,1v and 2v. Figure 7 shows comparison results for all four implementation at 100Mhz and 20°c, Figure 8 states the comparison results for all four designs at 100Mhz and 30°c. Similarly the results for all four designs at 200Mhz and at 10°c, 20°c and 30°c are indicated in Figure 9, Figure 10 and Figure 11 respectively.

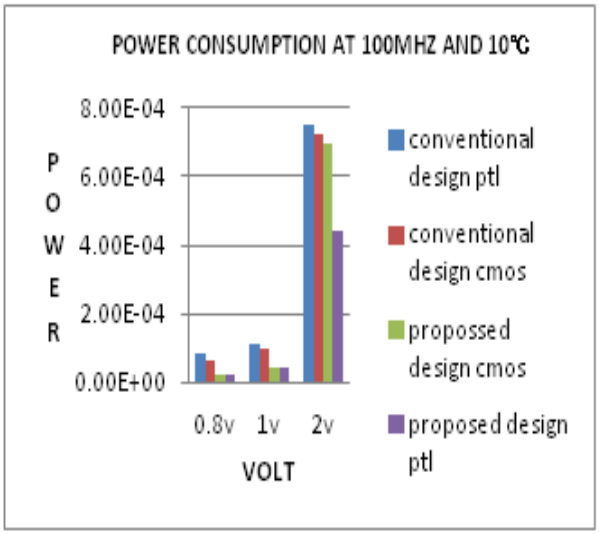


Fig 6: Comparison at 100Mhz and 10°c

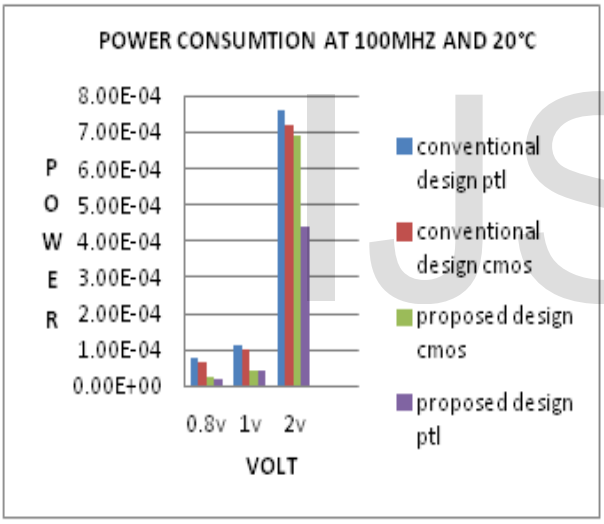


Fig 7: Comparison at 100Mhz and 20°c

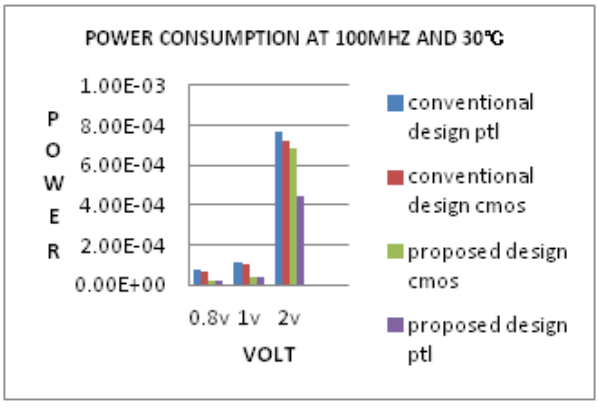


Fig 8: Comparison at 100Mhz and 30°c

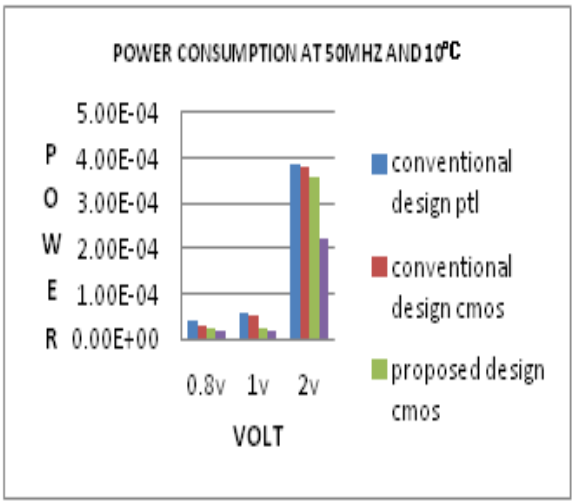


Fig 9:Comparison at 50Mhz and 10°

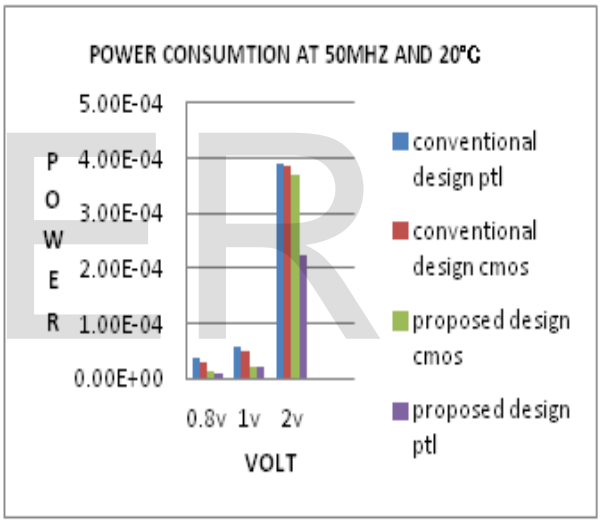


Fig 10: Comparison at 50Mhz and 20°c

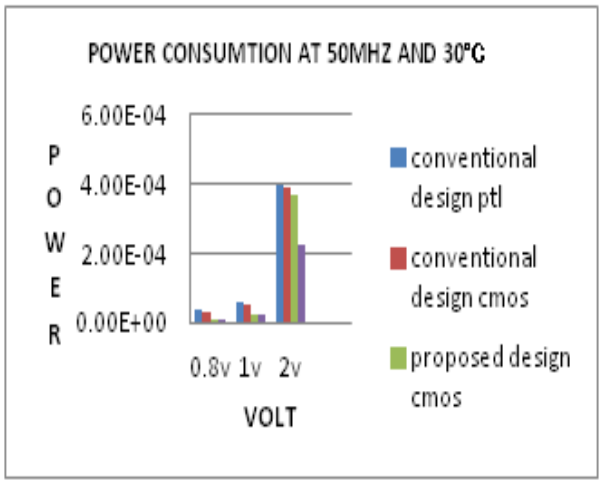


Fig 11: Comparison at 50Mhz and 30°c

Table 2
Calculations at 100MHZ

| | **100Mhz(e-005)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0.8v | | | 1v | | | 2v | | |
| **Design name** | 10 | 20 | 30 | 10 | 20 | 30 | 10 | 20 | 30 |
| Conventional design CMOS | 6.62 | 6.32 | 6.38 | 9.99 | 10.06 | 10.19 | 72.19 | 72.20 | 72.38 |
| Proposed design CMOS | 2.30 | 2.33 | 3.37 | 4.31 | 4.37 | 4.42 | 69.74 | 68.88 | 68.90 |
| Conventional design Hybrid PTL/CMOS | 8.41 | 7.64 | 7.62 | 11.16 | 11.23 | 11.36 | 75.13 | 75.75 | 76.53 |
| Proposed design Hybrid PTL/CMOS | 1.99 | 2.03 | 2.08 | 4.06 | 4.17 | 4.28 | 43.82 | 43.97 | 44.18 |



Table 3
Calculations at 50MHZ

| | **50Mhz(e-005)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0.8v | | | 1v | | | 2v | | |
| **Design name** | 10 | 20 | 30 | 10 | 20 | 30 | 10 | 20 | 30 |
| Conventional design CMOS | 3.15 | 2.97 | 3.23 | 5.04 | 5.10 | 5.16 | 38.14 | 38.85 | 40.99 |
| Proposed design CMOS | 1.15 | 1.17 | 1.19 | 2.16 | 2.18 | 2.21 | 37.15 | 38.28 | 38.99 |
| Conventional design Hybrid PTL/CMOS | 4.24 | 3.85 | 3.86 | 5.62 | 5.67 | 5.73 | 38.51 | 38.99 | 39.57 |
| Proposed design Hybrid PTL/CMOS | 1.00 | 1.02 | 1.05 | 2.04 | 2.14 | 2.16 | 22.3 | 22.54 | 22.73 |

## 6 . CONCLUSION

In this paper we presented four different designs of a decimal to BCD converter and proved that the Hybrid PTL/CMOS implementation of proposed design is superior in terms of power consumption to other three designs. Irrespective of the method used to produce the partial products in the multiplication process these converters can be incorporated in any multiplication circuits. We have considerably reduced the number of transistors required in 90nm Hybrid PTL/CMOS implementation of proposed design which produced least power to 674 as compared to 90 nm CMOS implementation of conventional design which required 1880 transistors. Comparing the most power efficient implementations of both the designs that is 90nm CMOS implementation of conventional design and 90nm Hybrid PTL/CMOS implementation of proposed design at 100Mhz ,10°c and 0.8v we observed that 90nm Hybrid PTL/CMOS implementation of proposed design is 68.11% more power efficient than 90nm CMOS implementation of conventional design.

## 7. REFERENCES

[1] IEEE standard floating-point arithmetic . IEEE SC, Oct. 2006[Online].Available:http://754r.ucbtest.org/drafts/archive/2006-10-04.pdf .

[2] M. A. Erle, E. M. Schwarz and M. J. Schulte.( 2005) "Decimal multi plication with efficient partial product generation," *17th IEEE Symposium on Computer Arithmetic*, pp. 21-28.

[3] A. Vazquez, E. Antelo and P. Montuschi, "A New Family of High-Performance Parallel Decimal Multipliers" *18th IEEE Symposium on Computer Arithmetic*, 2007, pp. 195-204.

[4] Sreehari Veeramachaneni, M. Keerthi Krishna , L. Avinesh, P Sreekanth Reddy and M.B. Srinivas, "Novel High-Speed 16-Digit BCD Adders Conforming to IEEE 754r Format", *IEEE Computer Society Annual Symposium on VLSI*, 2007, pp. 343-350.

[5] R. K. James, T. K. Shahana, K. P. Jacob and S. Sasi, " Decimal multiplication using compact BCD multiplier" *International Conference on Electronic Design*, 2008, pp.1 – 6.

[6] G. Jaberipur and A. Kaivani, "Binary-coded decimal digit multipliers" *IET Computers and Digital Techniques*, Volume 1, Issue 4, July 2007, pp. 377 - 381.

[7] Srihari Veeramachaneni and M. B. Srinivas, "Novel High-Speed Architecture for 32-Bit Binary Coded Decimal (BCD) Multiplier" *International Symposium on Communications and Information Technologies*, 2008, pp. 543 – 546.

[8] G. Jaberipur and A. Kaivani, "Improving the Speed of Parallel Decimal Multiplication" *IEEE Transactions on Computers*, vol. 58, issue 11, Nov. 2009, pp. 1539 - 1552.

[10] M. Schmookler, "High-speed binary-to-decimal conversion" *IEEE Transactions on Computers*, 1968, vol. 17, issue 5, pp. 506–508.

[11] V. T. Rhyne, "Serial binary-to-decimal and decimal-to-binary conversion" *IEEE Transactions on Computers*, 1970, vol. 19, Issue 9, pp. 808–812.

[12] B. Arazi and D. Naccache, "Binary-to-decimal conversion based on the 28 2 1 by 5" *Electronics Letter*, 1992, vol. 28, issue 23, pp. 2151–2152.