ISSN 2250-3153

Reduced Area & Improved Delay Module Design of 16-Bit Hamming Codec using HSPICE 22nm Technology based on GDI Technique

[1]Raj Kumar Mistri , [2]Md Arman Ansari, [3]Md Mustak Ali, [4]Manju Kumari, [5]Manoj Prabhakar, [6]Manju Kumari

[1] Assistant Professor, Department of Electronics & Communication Engg., RTCIT, Ranchi, Jharkhand, India.
[2,3,4,5,6] B.Tech Scholar, Department of Electronics & Communication Engg., RTCIT, Ranchi, Jharkhand, India.

Abstract- Gate diffusion input (GDI)—A new technique of low-power digital combinatorial circuit design is described. This technique allows reducing power consumption, propagation delay, and area of digital circuits while maintaining low complexity of logic design. Performance comparison with traditional CMOS and various pass-transistor logic design techniques is presented. The different methods are compared with respect to the layout area, number of devices, delay, and power dissipation.

Index Terms- GDI, VLSI, CMOS, SOI, Pass Transistors, Transmission Gate.

I. INTRODUCTION

Most of the VLSI applications, such as digital signal processing, image and video processing, and microprocessors, extensively use arithmetic operations. Addition, subtraction, and multiplication are examples of the most commonly used operations. Recently, building low-power VLSI systems has emerged as highly in demand because of the fast growing technologies in mobile communication and computation. The goal of this paper is designing a low-voltage and so low-power 16-bit hamming codec cell with the GDI technique. This technique that was recently developed and presented in [1], proposes an efficient alternative for logic design in standard CMOS and SOI technologies.

Hamming codec: Hamming codec includes mainly two sections, first one is hamming encoder and second one is hamming decoder.Hamming codes are used by hamming encoder to encode the input data as well as by hamming decoder to decode the encoded data. Hamming code is one of the most common error detecting and correcting codes used in Random access memory. In hamming code, k parity bit is added to an n-bit data word forming a new word of nth bit. The bit positions are numbered in sequence from 1 to nth. Those positions numbered as a power of 2 are reserved for the parity bits. The remaining bits are the data bits. There is relationship between data length (n) and the number of parity that must be added is as follows,

$$2^k - k - 1 \geq n \quad \text{.....1.1}$$

For example for 16 bit data, inequality goes to 2^k-1-k≥16, so at k=5, above inequality satisfy, hence for 16 bit data number of parity bit will be 5. Similarly for 128 bit data number of parity bit required is 8.

GDI Technique: GDI cell contains three inputs – G (common gate input of NMOS and PMOS), P (input to the source/drain of PMOS), and N (input to the source/drain of NMOS).

Bulks of both NMOS and PMOS are connected to N or P (respectively), so it can be arbitrarily biased at contrast with CMOS inverter. It must be remarked, that not all the functions are possible in standard P-Well CMOS process, but can be successfully implemented in Twin-Well CMOS or SOI technologies. GDI Technique allows improvements in design complexity level, transistor counts, static power dissipation and logic level swing.

TG Technology :Transmission gate logic circuit is a special kind of pass-transistor logic circuit. It is built by connecting a PMOS transistor and an NMOS transistor in parallel, which are controlled by complementary control signals. Both the PMOS and NMOS transistors will provide the path to the input logic "1" or "0", respectively when they are turned on simultaneously. Thus, there is no voltage drop problem whether the "1" or "0" is passed through it. It contains the 20 transistors [2]

II. COMPARISONS WITH OTHER LOGIC STYL**ERROR! REFERENCE SOURCE NOT FOUND.**ES

Circuits were designed at the transistor level in a 0.35m twin-well CMOS process technology (V, V). Each set includes a logic cell implemented in three different techniques: GDI, CMOS and transmission gate. Cells were designed for a minimal number of transistors in each technique as shown in Table I. Most circuits where implemented with ratio of three to achieve the best power-delay performance Same transitions of logic values were supplied to the inputs of the test circuits in each technique. Measured values apply to transitions in inputs connected to gate of transistors, in order to achieve a consistent comparison. Measurements were performed on test circuits that were placed between two blocks, which contain circuits similar to the device under test (DUT). The measured power is that of the DUT, including the power consumed by driving the next stage, thus accounting for the input power consumption and not just the power directly consumed from supply. This allows more

realistic environment conditions for test circuit, instead of the ideal input transitions of the simulator's voltage sources [3].

The performance evaluation is made with respect to the switching delay, transistor count and power consumed by Mod-GDI and CMOS logic. In CMOS the number of transistors used to realize a function is twice that of GDI. It is observed that GDI logic style has low area, low power and low delay when compared to TG and CMOS logic style.[4]

Among the forceful investigation in the field of low power, high speed digital applications due to the growing demand of systems like phones, laptop, palmtop computers, cellular phones, wireless modems and portable multimedia applications etc has directed the VLSI technology to scale down to nano-regimes, allowing additional functionality to be incorporated on a single chip[5].

Fig.1 introduce the design methodology of 2-input AND, OR & XOR gate by three different technologies (CMOS, TG & GDI). For 2-input AND gate & OR gate CMOS & TG technique uses 6T in each, however GDI tech. uses only 2T in each. For 2-input XOR gate, CMOS tech. uses 12T & TG tech. uses 8T, however GDI tech. uses only 4T. Since GDI tech. uses fewer transistors (T) in comparison to other technology, which cause to save the chip area as well as reduction of propagation delay.

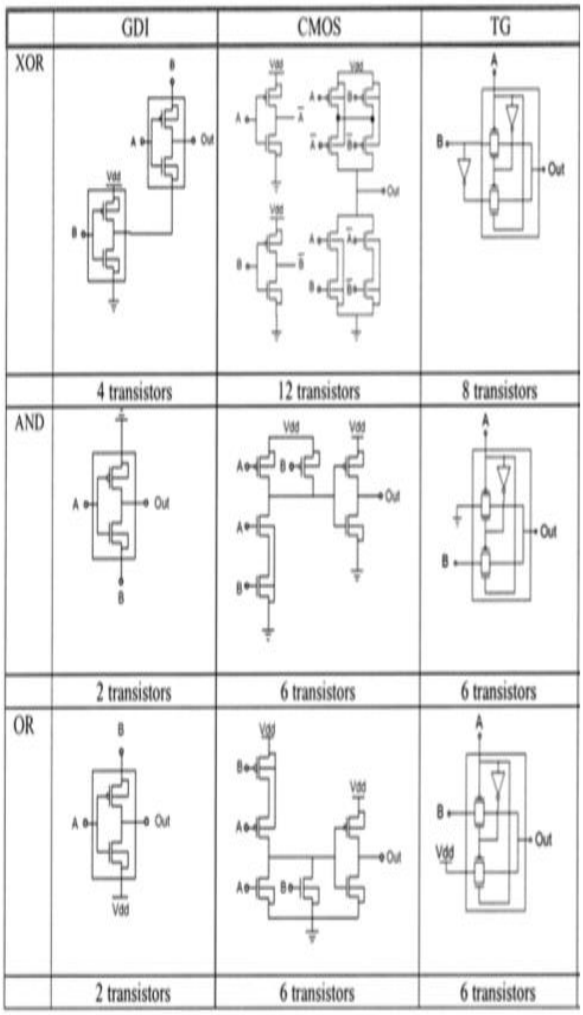

Fig.1 AND, OR and XOR module using GDI, CMOS, and TG design techniques.

III. HAMMING ENCODER

Hamming encoder encode the input data by adding the parity bit. For 16 bit data number of parity bit can be determined 2^k-1-k≥16, so at k=5 above inequality satisfy hence total number of parity bit added should be 5. Suppose IN1, IN2……IN16 are the input data and P1, P2, P4, P8, and P16 are the parity bits. Since parity bit always present at power of 2 positions. Since there is 5 parity bits so its positions will be 2^0,2^1,2^2, $2^3$and 2^4 that is 1st , 2nd , 4th , 8th and 16th position. Hence encoded bit & its position can be arranged as

P1, P2, IN1, P4, IN2, IN3, IN4, P8 IN5, IN6, IN7, IN8, IN9, IN10, IN11, P16, IN12, IN13, IN14, IN15, IN16.

Encoded bit position provides the information of position of data bit & parity bit at the output side of hamming encoder. 2 to the power i (2^i) indicate the parity bit position at the output of encoder, where i=0,1,2,……k and k indicate the total no. of parity bit added in it. The bit position other than 2^i at the output of encoder indicates the data bit position. For n-bit data no. of parity bit required is k, where k satisfies following inequality $2^k - k - 1 \geq n$. In this inequality for 16-bit data, at k=5 inequality satisfy, so no. of parity bit required is 5 and its position at output of encoder must be 1st , 2nd , 4th , 8th & 16th .
Position of data bit at output of encoder can be evaluated by $2^k - k > p$ where 'p' & indicate the data bit position at input side of encoder & 'k+p' indicate the data bit position at output side of encoder. Suppose we would like to know the position of 5th input data, in this case value of p will 5 & at k=4 inequality $2^k - k > p$ satisfy, hence its position at output side of encoder will be 'p+k' that is 9th .

Fig.2 provide the design information of 2-input AND, OR & XOR gate by three different technologies (CMOS, TG & GDI). For 2-input AND gate & OR gate CMOS & TG technique uses 6T in each, however GDI tech. uses only 2T in each.
For 2-input XOR gate, CMOS tech. uses 12T & TG tech. uses 8T, however GDI tech. uses only 4T. Since GDI tech. uses fewer transistors (T) in comparison to other technology, which cause to save the chip area as well as reduction of propagation delay.

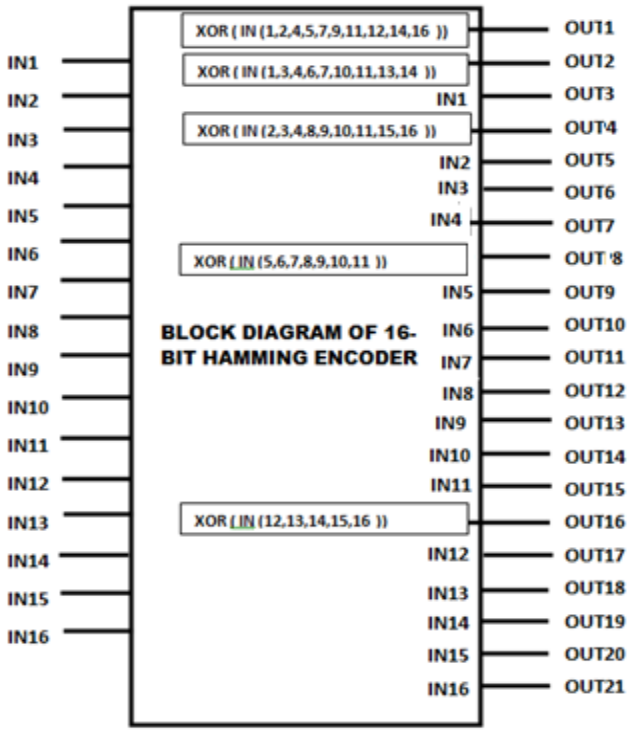

Figure 2: Block diagram of 16-bit hamming Encoder.

III.(A) Evaluation of parity bit

As it is clear that output of encoder is always in terms of input data bits and parity bits added to it. Now it is important to evaluate the parity bits. For this we developed a c-program based algorithm which is capable to evaluate parity bits as well as the position of input data bit at the output of encoder. Steps required to evaluate the parity bits are as follows

Step-1. Firstly n-bits input data of encoder is stored in array, suppose n-bit input data can be stored in array enco_inp[n]

Step-2. In this step no. of parity bit is evaluated, suppose k is integer variable then below code evaluate the no. of parity bit for n-bit data & its value will be stored in k.

```
for(int k=1;k<=100;k++){ if((pow(2,k)-k-1)>=n)  break; }
```

Step-3. In this step position of input bit at output of encoder is decided and this has been stored at the appropriate position at its output. Array enco_op[n+k] stored the input data at its appropriate positions.

```
int a=1;  int enco_op[n+k+1];
for(int j=1;j<=k;j++){ for(int i=a; i<=n; i++){
if((i+j)%(int(pow(2,j)))==0){
a++; break;}} ecno_op[i+a]=enco_inp[i];}
```

Step-4. In this step parity bit is evaluated and placed it at appropriate position of output of encoder.

```
int c[k][n];
int a1,b1=1;
for(int j=1;j<k;j++){a1=1; for(int i=1; i<=n+k; i++){
if(int(i/(int(pow(2,j))))!=0  &&  (i%(int(pow(2,j)))>=int(pow(2,j-1))) && (i%(int(pow(2,j)))<int(pow(2,j))) {
c[b1][a1]=enco_op[i];
a1++}}b1++;}
```

above code stored the values of bits which can be XORed to evaluate parity bits in 2-D array c[k][n]. c[1][], c[2][], c[3][], c[4][], c[5][] stores the values of bits for 1^{st} , 2^{nd} , 4^{th} , 8^{th} & 16^{th} position of parity bit respectively.

Below code evaluate the parity bits. P[1], p[2], p3], p[4] & p[5] gives the value of parity bits .

```
int p[]={0,0,0,0,0,0};
for(int j=1;j<=k;j++){
for(int i=1; i<=n; i++){p[j]=p[j]^c[j][i];}
enco_op[int(pow(2,j-1)]=p[j];}
```

Hence parity bit can calculated as,

P1=p[1]=XOR (IN1, IN2, IN4, IN5, IN7, IN9, IN11, IN12, IN14, IN16)
P2=p[2]=XOR (IN3, IN6, IN7, IN10, IN11, IN14, IN15, IN18, IN19)
P4=p3]=XOR (IN5, IN6, IN7, IN12, IN13, IN14, IN15, IN20, IN21)
P8=p[4]=XOR (IN9, IN10, IN11, IN12, IN13, IN14, IN15)
P16=p[5]=XOR (IN17, IN18, IN19, IN20, IN21)

IV. HAMMING DECODER

Hamming decoder is one that decode the encoded data. If m-bits are inputs to the hamming decoder then the no. check bit to detect & correct 1-bit error can be evaluated by inequality $2^{ck} - 1 \geq m$, where 'ck' indicate the total no. of check bit required. For example if total no of input to encoder is 21-bit then at ck=5 above inequality satisfy so, no. of check bit required is 5. Check bit evaluation by c-programming based algorithms includes following steps

Step-1: For n-bit input data to decoder, no of check can be evaluated as by c-programming based algorithms

```
for(int ck=1;ck<=100;ck++){
if((pow(2,ck=ck)-1)>=n)  break; }
```

In above code 'ck' stores the value of total no. of check bit required.

Step-2: In this step values of check bits can be evaluated. Below code evaluate the values of check bits, which are stored in ckb[1], ckb[2], ckb[3], ckb[4] & ckb[5].

```
int cc[k][n];
int a1,b1=1;
for(int j=1;j<ck;j++){a1=1; for(int i=1; i<=n; i++){
if((i%(int(pow(2,j)))>=int(pow(2,j-1)))  &&
(i%(int(pow(2,j)))<int(pow(2,j))) {
cc[b1][a1]=enco_op[i]; a1++}} b1++;}

int ckb[]={0,0,0,0,0,0};
for(int j=1;j<=ck;j++){for(int i=1; i<=n; i++){
```

ckb[j]=ckb[j]^c[j][i];}}

Suppose check bits are C1, C2, C4, C8 & C16. These check bits are evaluated as.

C1= ckb[1]=XOR(1,3,5,7,9,11,13,15,17,19,21)
C2= ckb[2]=XOR(2,3,6,7,10,11,14,15,18,19)
C4= ckb[3]=XOR(4,5,6,7,12,13,14,15,20,21)
C8= ckb[4]=XOR(8,9,10,11,12,13,14,15)
C16= ckb[5]=XOR(16.17,18,19,20,21)

Fig.3 show the block diagram of 16-bit hamming decoder which has mainly consists one 11-input xor gate, two 10-input xor gate, One 8-input xor gate ,one 6-input xor gate, sixteen 2-input xor gate & one bit corrector circuit.

One bit corrector circuit is nothing but a 5 X 32 decoder which have capable to generate only min-terms like.

∑m(3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17, 18, 19, 20, 21).

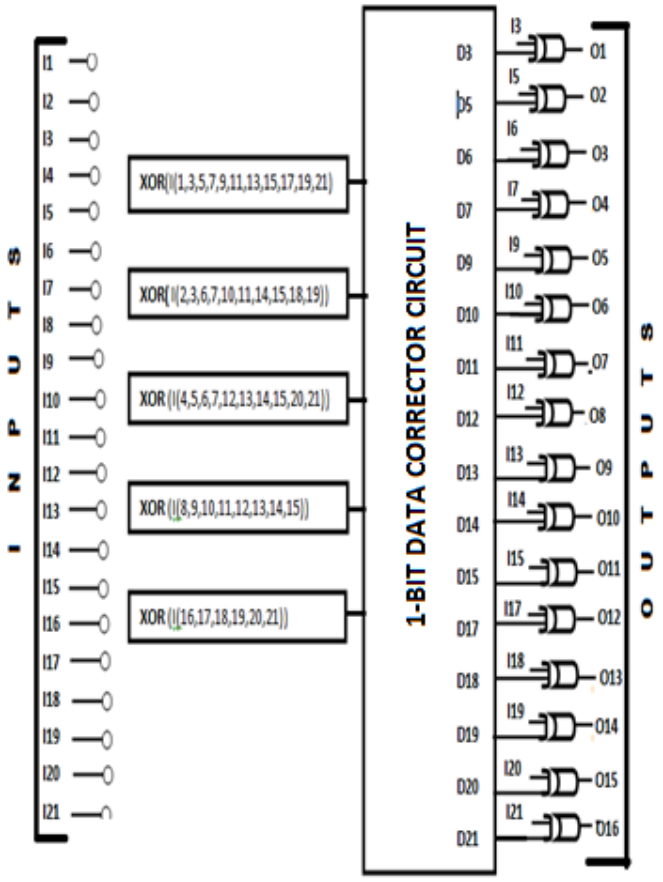

Figure 3: Block diagram of hamming Decoder.

V. SIMULATION RESULT

Simulation results consists of input waveform to the encoder, output waveform from encoder, input waveform to decoder & output from hamming decoder.

Input waveform for hamming encoder is shown in Fig.4, in this waveform 4, 16-bit data is given to the input of encoder and the data which are given to its inputs are 0101011011001001, 10000010010001101, 0111001011100000 & 1010000110100100.

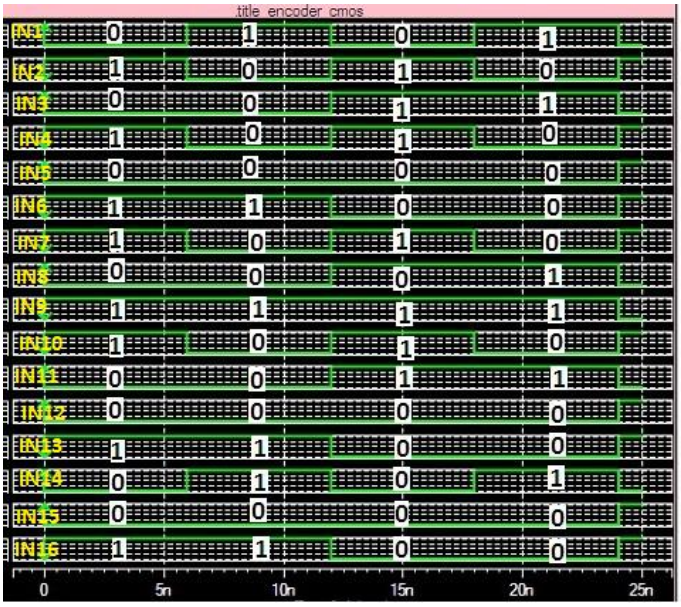

Figure.4 : Input waveform given to 16-bit hamming Encoder.

As we know that for n-input data of hamming encoder, no. of parity bits added to its output is k & its value can be evaluated when inequality $2^k - k - 1 \geq n$ satisfy. For 16-bit input data, at k=5, inequality satisfy that is why no of parity bit required is 5. Hence the no. of bits at its output side is 21. From its output waveform (Fig.5) output bits are 110110100110110001001, 001000000100100101101, 110111110010111000000, and 001001010001101100100.

Figure.5 : output waveform of 16-bit hamming Encoder.

Fig.6 show the input waveform which is given to the hamming decoder. From waveform it is clear that four 21-bit data are given as a input & these are

111110100110110001001, 001000000110100101101, 110111110000111000000, 000001010001101100100.

Bits inside the red circle indicate the error bit present at input side of decoder. Error bits are present at 3rd ,11th , 11th & 3rd positions respectively. As we know that 3rd bit-position of decoder input indicate the 1st bit input & 11th bit-position of decoder input indicate the 7th bit input of encoder. Since hamming codec has capacity to correct 1-bit data so, these bits must be corrected at output of decoder.

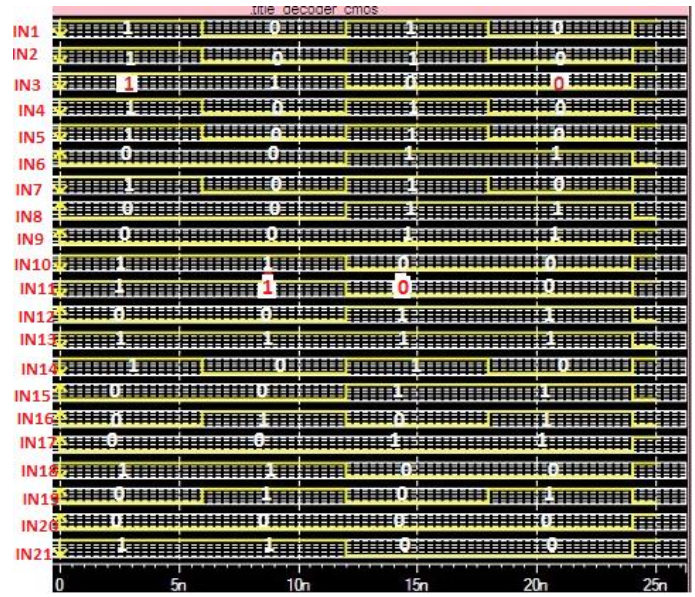

Figure.6 : Input waveform given to 16-bit hamming decoder.

Fig.7 indicate the output waveform of hamming decoder, in which error bit present at the input to decoder is corrected and data become original data which has given to the input of encoder. The original data from fig.7 are

0101011011001001, 1000001001000110 1, 0111001011100000 & 1010000110100100.

Bits inside the green bubble indicate the corrected bit by hamming decoder at output side of it.

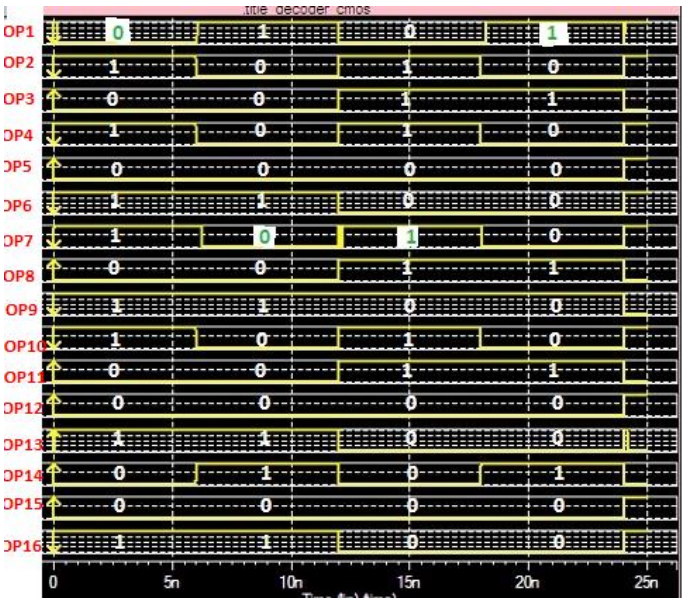

Figure.7 : output waveform of 16-bit hamming decoder.

VI. EXPERIMENTAL RESULT

Experimental result provide the information about no. of transistor usage by different technologies (CMOS,TG,GDI) as well as the simulation time taken by CPU by different technologies, which indicate the propagation delay.

Table.1 which is given below tells about the simulation time taken by CPU for both encoder as well as decoder by different technologies.

Table.1 Delay Table

	SIMULATION TIME IN SECOND TAKEN BY CPU		
	GDI	CMOS	TG
ENCODER	2.2	4.98	3.21
DECODER	5.62	12.76	8.37
TOTAL	7.82	17.74	11.58

From delay table it is clear that GDI tech. save 55.82% time over CMOS & 31.46% time over TG tech. in case of encoder. In case of decoder, GDI tech. save 55.95% time over CMOS and 32.85% over TG technology.

Table.2 confirms that GDI tech. save 66.67% chip area over CMOS & 50% over TG tech. in case of hamming encoder. In case of decoder GDI tech save 54.23% chip area over CMOS & 44.75% over TG technique.

Table.2 Transistor usage by module in three different technique.

		NO. OF TRANSISTORS USAGE		
		GDI	CMOS	TG
16-BIT HAMMING CODEC	ENCODER	160	480	320
	DECODER	432	944	782
TOTAL		592	1424	1102

Data given in table.1 & table.2 are graphically represented by a bar graph in fig.7, which show the simulation time delay as well as transistors counts in both hamming encoder as well as hamming decoder comparatively.

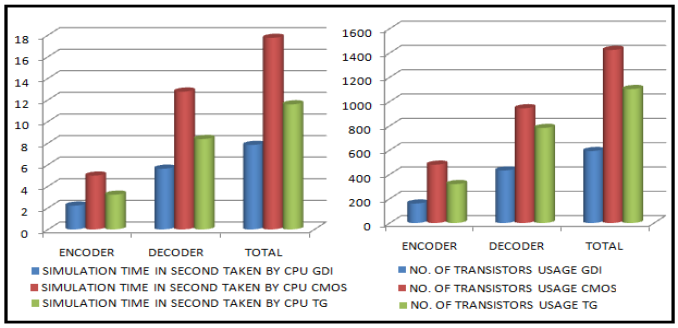

Figure.8 Graphical comparison [delay & area] of hamming codec by different techniques

VII. CONCLUSION

A GDI technique for low-power design was presented. An 16-bit Hamming Codec was designed using GDI. Numerous logic gates and high level digital circuits are implemented in various methods and process technologies, and their simulation results are discussed. Comparisons with existing TG and CMOS techniques were carried out, showing an up to 58.42% reduction of chip area in complete module using GDI over CMOS and up to 46.27% reduction of chip area using GDI over TG. In this cell, the GDI technique has been used for generating of intermediate functions of XOR and AND.

REFERENCES

[1] A. Morgenshtein, A. Fish, I. A. Wagner, "Gate Diffusion Input (GDI) – A Novel Power Efficient Method for Digital Circuits: A Design Methodology," 14th ASIC/SOC Conference, Washington D.C., USA, September 2001.

[2] C. H. Chang, J. Gu and M. Zhang, "A review of 0.18um full adder performance for tree structured arithmetic circuits", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 13, No. 6, pp.686-695, June 2005.

[3] K.Bernstein,L.M.Carrig, C.M.Durham, and P.A. Hansen, High Speed CMOS Design Styles. Norwell, MA: Kluwer Academic, 1998.

[4] A. Morgenshtein, I. Shwartz, A. Fish, "Gate Diffusion Input (GDI) Logic in Standard CMOS Nanoscale Process", 2010 IEEE 26th Convention of Electrical and Engineers in Israel.

[5] R.Uma and P. Dhavachelvan,"Modified Gate Diffusion Input Technique: A New Technique for Enhancing Performance in Full Adder Circuits" 2nd International Conference on Communication, Computing & Security [ICCCS-2012]

AUTHORS

First Author – M r. Raj Kumar Mistri born in 1983, completed his B.Tech. in Electronics and Instrumentation Engineering, NIST Berhampur Orissa India & also completed his M.Tech.in Electronics and Communication Engineering (VLSI Design & Embedded System) , NIT Jamshedpur Jharkhand India . He is currently Assistant Professor in the department of Electronics & Communication Engineering at RTC Institute of Technology Anandi, Ranchi, India. His research interests include VLSI Design, Digital Signal Processing, Computational Geometry & Pattern Recognition.

Second Author – Mr. Md. Arman Ansari born in 1989, pursuing his B.Tech.in Electronics and Communication Engineering, RTC institute Technolgy, Anandi . His research interests include Digital Hardware Designing.

Third Author – Mr. Md. Mustak Ali born in 1991, pursuing his B.Tech.in Electronics and Communication Engineering , RTC institute Technolgy, Anandi. His research interests include Digital Signal Processing.

Fourth Author – Miss Manju Kumari born in 1990, pursuing her B.Tech.in Electronics and Communication Engineering , RTC institute Technolgy, Anandi . Her research interests include Digital Signal Processing.

Fifth Author – Mr Manoj Prabahakar born in 1991, pursuing his B.Tech.in Electronics and Communication Engineering, RTC institute Technolgy, Anandi . Her research interests include Digital Hardware Designing.

Sixth Author – Miss Manju Kumari born in 1993, pursuing her B.Tech.in Electronics and Communication Engineering , RTC institute Technolgy, Anandi. Her research interests include Digital Signal Processing.