An Optimised Diminished One Modulo 2^n+1 Low Power Static And Dynamic Adder Using Circular Carry Selection

Suganya devi.R[1], Prabhu.B.M[2]
PG scholar, Embedded System Technologies, Angel College of Engineering & Technology, Tirupur[1].
Asst. Prof. Embedded System Technologies, Angel College of Engineering & Technology, Tirupur[2].

Abstract: The diminished-one modulo 2^n+1 addition is an important arithmetic operation for a high-performance residue number system. In this paper, we propose a new circular-carry-selection (CCS) technique for modulo 2^n+1 addition in the diminshed-one number domain. The architecture design of CCS is technique for modulo 2^n+1 addition in the diminished-one number domain. The architecture design of CCS modular adder is simple and regular for various bit-width inputs. Low power static and dynamic adder technique is used for actual VLSI implementation; the proposed modular adder can demonstrate its superiority of savings up to 39.5% in Areax Time and 46.3% in Timex Power performances over those of the previous existing solutions under 180-nm CMOS technology. Finally, the chip area and the clock rate of CCS diminished-one modulo $2^{16}+1$ adder are 26746µm2 and 476MHz, respectively.

Keywords: circular carry selection (CCS) modulo 2^n+1 adder, residue number system (RNS), VLSI design.

I. Introduction

Residue number system (RNS) is a non-weighted number system which exhibits a parallel carry-free arithmetic feature in digital signal processing (DSP). RNS is based on a - moduli set (P1,P2,.,.,PN) where all moduli Pi are pair-wise relatively prime. The binary number X can be converted into a residue representation (x1,x2.,.,xN) by forward conversion where xi = X modulo Pi (denoted by <X> Pi), In RNS, the arithmetic operation of X and Y is defined by zi = <xi◊yi>Pi for i=1,2.,.,N where ◊ indicates addition, subtraction or multiplication ,For example, assume two 5-bit binary numbers X= 1310 = 011012 and y= 1710=100012 For 3-moduli set (P1,P2,P3) = (3,5,7) we can obtain the residue representations X = (1,3,6) and Y = (2,2,3) Compared with binary number system, the residue number in each modular channel has the smaller bit-width which is only 2- or 3-bit wide. An RNS addition of X and Y is given as follows:

(z1,z2,z3) = (<1+2>3,<3+2>5,<6+3>7) = (0,0,2)

The result (0,0,2) is the residue representation of the sum value x=1310 .It can be found that the computations of z1,z2, and z3 are independently obtained by three modular additions in parallel. This indicates the carry-free feature of the residue arithmetic. Many moduli sets such as (2^n-1,2^n,2^n+1) (2^n-1,2^n,2^n+1,2^{2n}+1) and (2^n-1,2^n,2^n+1, 2^{2n}+1+1) etc, are frequently utilized for designing successful RNS-based DSP applications. Among these moduli sets, the arithmetic in modulo 2^n-1 type or 2^n type channel only handles bit operands and the corresponding modulo operation is easy to design, On the contrary the arithmetic in modulo 2^{2n}+1 type channel computes (n+1) bit operands and its modulo operation is more complex to implement, such that it mainly dominates the performance of the whole RNS system in terms of area, delay and power. Therefore, the 2^n+1 type modulus is the significant and complicated modular element in many moduli sets. In this paper we focus on the design subject of an efficient modulo 2^n+1 addition. Given two (n+1) bit inputs A and B in the range [0,2n] the modulo 2^n+1 addition is defined by <A+B>2n+1. The diminished-one number arithmetic was adopted to design an efficient modulo 2^n+1 adder. For a diminished-one modulo 2^n+1 adder the inputs A and B are decreased by one to obtain diminished-one data A* = A -1 and B* =B-1 which have n-bit width. Therefore, the diminished –one modulo 2^n+1 addition can be designed by n-bit adder and modulo function. This leads to the resulting modular adder be suitable for constructing a high-speed RNS addition. Several hardware designs of diminished -one modulo 2^n+1 adder. Although these modular adder architectures are fast especially for the fastest parallel-prefix modulo 2^n+1 adder their circuit costs are sill heavy. The latest design is the select-prefix modulo 2^n+1 adder exhibits an improved performance in the area-delay space.

In this paper, a new circular-carry-selection (CCS) technique is presented to design an efficient diminished-one modulo 2^n+1 adder. The proposed CCS modular adder simply consists of dual-sum carry look-ahead (DS-CLA) adder, circular-carry generator (CCG) and multiplexer (MUX). The DS-CLA adder is designed to generate two different sums in parallel. The carry-out bit computed by CCG is then used to circularly control the MUX for obtaining the correct modulo result. Based on UMC 180-nm CMOS design kit, the experimental results illustrate that the proposed CCS modular adder has reduced both area- time (AT) and time-power (TP) products.

The rest of this paper is organized as follows. In Section II, the architecture design of the proposed CCS modular adder is presented. Section III provides the performance comparison with the previous works and shows an efficient VLSI implementation for CCS diminished-one modulo $2^{16}+1$ adder. The conclusion is made in section IV.

II. Proposed CCS Diminished-One Modulo Adder

Assume that two n-bit diminished-one operands are A* = A -1 = a*n-1 . . . a*0 and B* =B-1 = b*n-1. . . b*0. The sum S*=s*n-1. . . s*0 derived by performing modulo 2^n+1 addition of A* and B* can be changed into the un complicated function with performing modulo 2n addition as the following expression:

$$S^* = < A^* + B^* + cn\text{-}1 > 2^n \qquad (1)$$

Where cn-1 is regarded as an original carry-out bit of (A*+ B*). Denote the carry generate term and the carry propagate term as g*i = a*i • b* i and p*i = a*i % b*i where stands for XOR function. According to CLA function. The carry term of c*i is derived by c*i = g*i + $\sum_{j=0}^{i-1}(\prod_{k=j+1}^{i} p^*{}_k) \, g^*{}_j + c^*{}_{-1} \prod_{k=0}^{i} p^*{}_k$ for i = 0,….,n-1, where c*-1 is the carry-in bit. Based on CCS technique, we set c*-1= cn-1. The Boolean function of each sum bit in (1) can be expressed as follows:

$$\begin{aligned} s_i^* &= c_{i-1}^* \oplus p_i^* \\ &= \left(g_{i-1}^* + \sum_{j=0}^{i-2} \left(\prod_{k=j+1}^{i-1} p_k^* \right) g_j^* + \overline{c_{n-1}} \prod_{k=0}^{i-1} p_k^* \right) \oplus p_i^* \quad (2) \end{aligned}$$

where

$$c_{n-1} = g_{n-1}^* + \sum_{j=0}^{n-2} \left(\prod_{k=j+1}^{n-1} p_k^* \right) g_j^*. \qquad (3)$$

Since $c_{n-1} \in \{0, 1\}$, we have

$$s_i^* = \begin{cases} s_{i,1}^* = \left(g_{i-1}^* + \sum_{j=0}^{i-2} \left(\prod_{k=j+1}^{i-1} p_k^* \right) g_j^* \right) \oplus p_i^* & \\ \quad + \prod_{k=0}^{i-1} p_k^* \oplus p_i^*, & \text{if } c_{n-1} = 0\,. \\ s_{i,0}^* = \left(g_{i-1}^* + \sum_{j=0}^{i-2} \left(\prod_{k=j+1}^{i-1} p_k^* \right) g_j^* \right) \oplus p_i^*, & \text{if } c_{n-1} = 1 \end{cases} \qquad (4)$$

In (4), we can easily design a DS-CLA adder to produce two sums si*,1 and si*,0 since they have the same term $(g^*_{i-1} + \sum_{j=0}^{i-2}(\prod_{k=j+1}^{i-1} p_k{}^*) \, g_i{}^*) \oplus p_i{}^*$ .. In other words, they can share the circuit from the view point of hardware design. At the same time, cn-1 generated by the CLA function of (3) is circularly used to control MUX for getting the correct outputs si*,s. The block diagram of CCS diminished-one modulo 2n+1 adder is shown in Fig. 1, which is simple and regular. For the sake of clarity, Fig. 2 shows the detailed logic design for CCS diminished-one modulo 2n+1 adder. Next, in order to speed up the CCS modular adder for the large dimension of n we partition the n-bit CCS modular adder into m r – bit CCS addition blocks and a fast CCG where n = m x r Fig. 3 illustrates the general (m x r) – bit CCS modular adder.

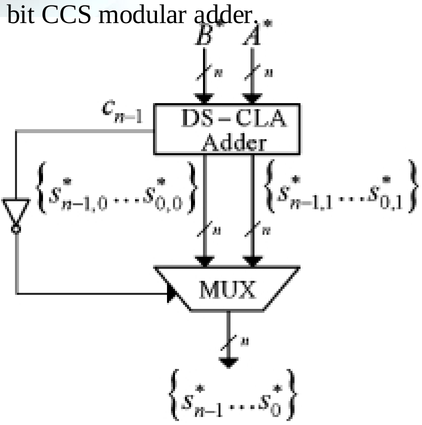


Fig. 1 Block diagram of CCS diminished-one modulo adder

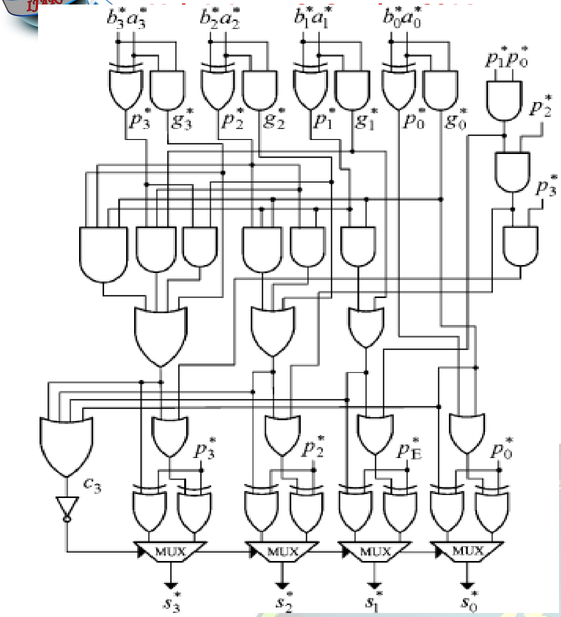


Fig. 2 Logic circuit of CCS diminished-one modulo2^4+1 adder

Both input data are divided into block inputs: $A^* = \{A^*_{m-1} \ldots.\, A^*_0\}$ and $B^* = \{ B^*_{m-1}. . . B^*_0$ where $Ai^* = a^*_{(t+1)r-1}. . . .a^*_{tr+1}a^*_{tr}$ and $B_i^* = b^*_{(t+1)r-1.} . . .b^*_{tr+1}b^*_{tr}$ for $t = 0,. . . .(m-1)$. The block sum $s^*_t = s^*_{(t+1)r-1}. . . . s^*_{tr+1}\ s^*_{tr}$ is derived by $A^*_t + B^*_t + K^*_{t-1}$ where K^*_{t-1} represents the carry-out bit of the $(t-1)^{th}$ addition block. In each 4 bit CCS addition block, the DS-CLA adder generates two block sums $s^*_{t,0} = s^*_t$ for $K^*_{t-1} = 0$ and $s^*_{t,1} = s^*_t$ for $K^*_{t-1} = 1$ in parallel. Likeeise , the carry –out bit K^*_{t-1} is used to select the correct block sum. When t = 0 K^*_{-1} is viewed as the carry-in input of the 0^{th} addition block and we can set $K^*_{-1} = c_{n-1}$

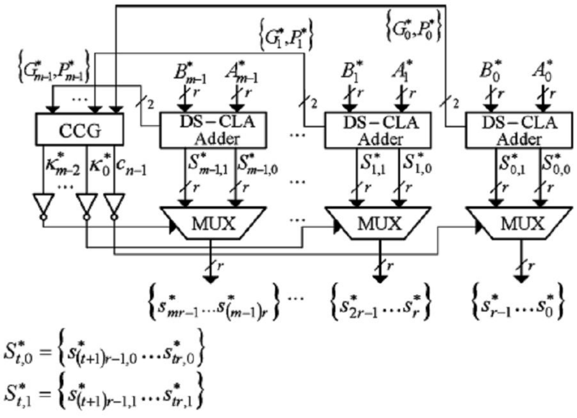


Fig. 3 The (m x r) partitioned CCS modular adder

b asked on CCS technique. Each carry-out signal K^*_{t-1} for t = 1. . . M-1 can be generated by CCG as follows:

$$\kappa^*_{t-1} = G^*_{t-1} + \sum_{j=0}^{t-2}\left(\prod_{l=j+1}^{t-1} P^*_l\right) G^*_j + \overline{c_{n-1}} \prod_{l=0}^{t-1} P^*_l$$
$$= \begin{cases} \kappa^*_{t-1,1} = G^*_{t-1} + \sum_{j=0}^{t-2}\left(\prod_{l=j+1}^{t-1} P^*_l\right) G^*_j \\ \qquad + \prod_{l=0}^{t-1} P^*_l, & \text{if } c_{n-1} = 0\,. \\ \kappa^*_{t-1,0} = G^*_{t-1} + \sum_{j=0}^{t-2}\left(\prod_{l=j+1}^{t-1} P^*_l\right) G^*_j, & \text{if } c_{n-1} = 1 \end{cases} \quad (5)$$

In (5), the block generate term $G^*_t = g^*_{tr}+(r-1) + \sum_{j=tr}^{tr+(r-2)} (\prod_{k=j+1}^{tr+(r-1)} p^*_k)\ g^*_j$ and the block propagate term $P^*t = \prod_{k=tr}^{tr+(r-1)} p^*_k$ are provided by the t^{th} CCS addition block. Besides, according to the expressions of G*i and P*i the original carry-out bit cn-1 in (3) can be also produced by CCG as follows:

$$c_{n-1} = G^*_{m-1} + \sum_{j=0}^{m-2} (\prod_{t=j+1}^{m-1} p^*_i)\ G^*_j \qquad (6)$$

After comparing (5) and (6), the carry signals of $K^*_{t-1},1$ and $K^*_{t-1},0$ can be extracted from the Boolean function of computing the carry-out bit cn-1 simultaneously. By using MUX for selection, the carry signal K^*_{t-1} in (5) is generated quickly. Fig. 6 depicts the CCG logic circuit for the 4 x 4 partitioned CCS modular adder.

III.Static And Dynamic Ripple Carry Adder

The most basic and intuitive BFA is an SRC adder. This type of adder has the benefits of simplicity and a synchronicity. A synchronicity means that the output of the adder can be accessed at any point during a clock cycle. This allows the adder to be used in two main styles of processors: 1) those that read/ calculates data on the rising clock edge and write data on falling clock edge and 2) those that read/ calculate data during one or more full clock cycles and write data during one or more subsequent clock cycles. AOI (And- Or-Invert) logic is a technique of using equivalent Boolean logic expressions to reduce the number of gates required for a particular expressions. This, in turn, reduces capacitance and consequently propagation times.

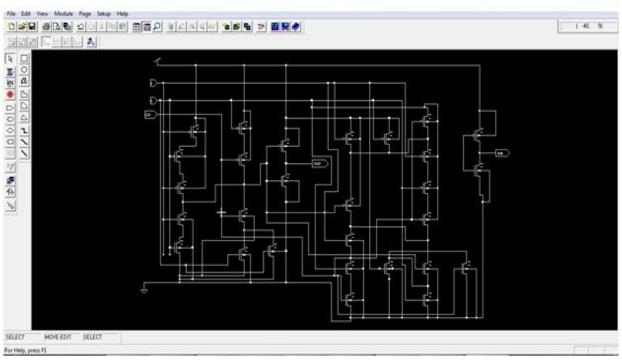


Fig. 4.1 bit static Ripple carry adder

The DRC adder is an advanced version of the SRC. Utilizing a clock allows the adder to take advantage of a technique known as recharging. This involves the charging the sum and carry bits to an intermediate value (usually VDD/ 2). This reduces the rise and fall time when logic low or high is computed. The downside to this approach, however, is that the adder result is only available when the clock signal is high. Consequently, a latch is generally used to hold the data for the remainder of the clock cycle. Power consumption of the adder is also increased due to the recharging.

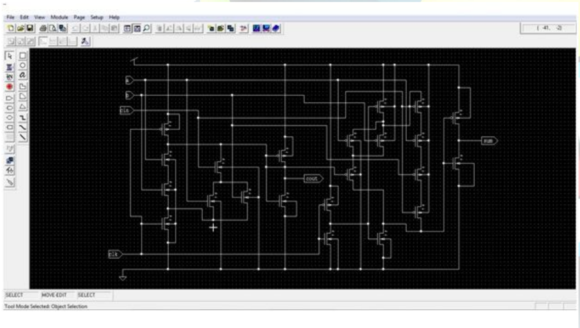

File Edit View Module Page Setup Help
cout
SELECT
MOVE EDIT
SELECT
Tool Mode Selected: Object Selection

Fig. 5 1 bit dynamic Ripple carry adder

A processor designer has a few choices when choosing a clock to work with this type of adder. Since the result can only be calculated when the clock is high, the clock period must be at least twice as long as the adder propagation time. Depending upon the needs of the processor, anywhere from (1) to n number of bits could be computed in one clock cycle.

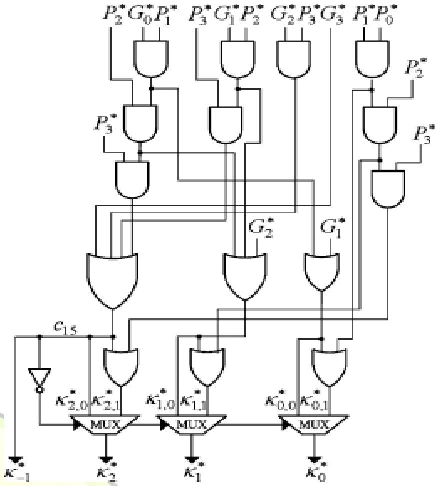
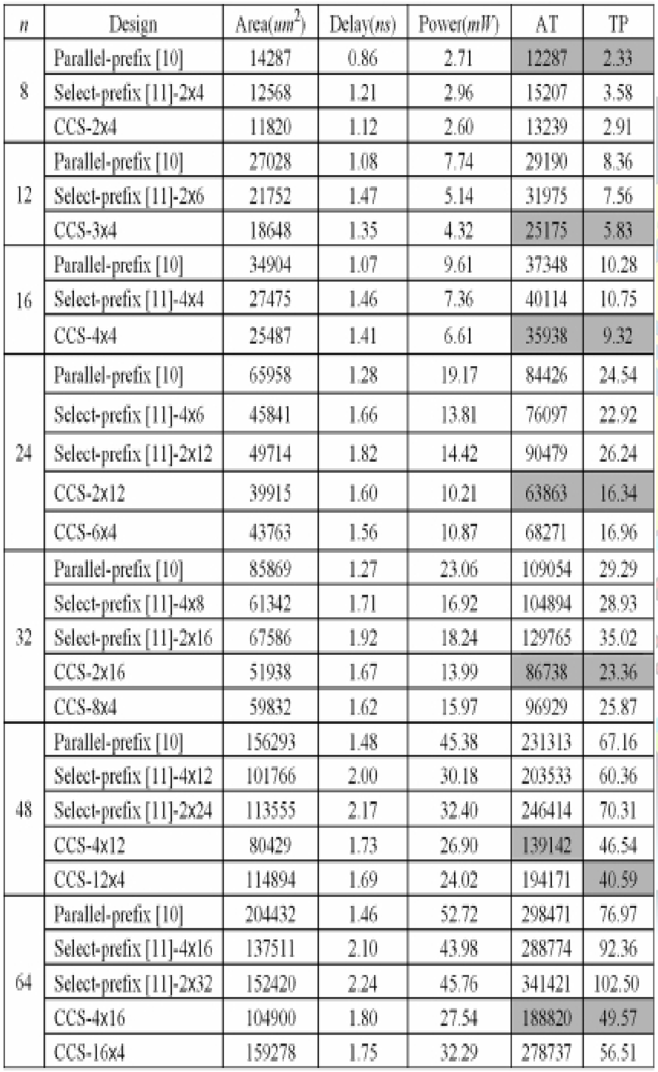


Fig. 6 Logic design of CCG for 4 x 4 partitioned CCS modular adder

We compare the CCS diminished-one modulo 2^n + 1 adder against two previous design of parallel- prefix modular adder and select-prefix modular adder, which are regarded as the faster and the most AT efficient designs among the existing solutions. In order to make an accurate comparison, we use UMC 180-nm design kit with cadence's PKS and Silicon Ensemble tools to implement the designs of and our CCS modular adder. The above modular adder implementations include a real-zero indicator which is referred to deal with special zero representation in diminished-one number domain.

Table I shows the comparison in terms of area, delay time, power consumption, AT and TP products with various dimensions of n =, 12, 16, 24, 32, 48 and 64, which are commonly used for RNS- based DSP applications. Two designs of CCS and select-prefix modular adders are realized under the block portioning of m x n for the optimal performance. The shaded parts in the table indicate the best results for the specific dimension of n. we can see that, for n > 8 the CCS modular adder has less AT and TP products. Fig. 7 illustrates the AT and TP gains of the proposed CCS modular adder against the designs. From Fig. 7, our proposed CCS modular adder is up to the AT and TP gains of 39.5% and 39.6% more efficient than the parallel-prefix modular adder while the gains of 34.6% and 46.3% than the select-prefix modular adder, respectively. Overall, our approach can achieve the average AT gains of 18.8% and 20.6%, and the average TP gains of 21.2% and 26.0%. This leads CCS modular adder to be profitable for many real applications when requiring a good compromise in area, delay and power. Finally, we implement the chip of CCS diminished-one modulo 2^{16} + 1 adder and the corresponding layout is shown in Fig.6. The chip area is about responding layout is shown in Fig. 7. The chip area is about 26746 µm2. Considering the parasitic effects of wire loading and I/O pad, the power consumption of the chip is measured at 11.2 mW under a 1.8-V power supply. The working frequency can achieve 476 MHz.

TABLE 1
COMPARISON OF THE SYNTHESIZED ADDERS



n	Design	Area(um^2)	Delay(ns)	Power(mW)	AT	TP
8	Parallel-prefix [10]	14287	0.86	2.71	12287	2.33
	Select-prefix [11]-2x4	12568	1.21	2.96	15207	3.58
	CCS-2x4	11820	1.12	2.60	13239	2.91
12	Parallel-prefix [10]	27028	1.08	7.74	29190	8.36
	Select-prefix [11]-2x6	21752	1.47	5.14	31975	7.56
	CCS-3x4	18648	1.35	4.32	25175	5.83
16	Parallel-prefix [10]	34904	1.07	9.61	37348	10.28
	Select-prefix [11]-4x4	27475	1.46	7.36	40114	10.75
	CCS-4x4	25487	1.41	6.61	35938	9.32
24	Parallel-prefix [10]	65958	1.28	19.17	84426	24.54
	Select-prefix [11]-4x6	45841	1.66	13.81	76097	22.92
	Select-prefix [11]-2x12	49714	1.82	14.42	90479	26.24
	CCS-2x12	39915	1.60	10.21	63863	16.34
	CCS-6x4	43763	1.56	10.87	68271	16.96
32	Parallel-prefix [10]	85869	1.27	23.06	109054	29.29
	Select-prefix [11]-4x8	61342	1.71	16.92	104894	28.93
	Select-prefix [11]-2x16	67586	1.92	18.24	129765	35.02
	CCS-2x16	51938	1.67	13.99	86738	23.36
	CCS-8x4	59832	1.62	15.97	96929	25.87
48	Parallel-prefix [10]	156293	1.48	45.38	231313	67.16
	Select-prefix [11]-4x12	101766	2.00	30.18	203533	60.36
	Select-prefix [11]-2x24	113555	2.17	32.40	246414	70.31
	CCS-4x12	80429	1.73	26.90	139142	46.54
	CCS-12x4	114894	1.69	24.02	194171	40.59
64	Parallel-prefix [10]	204432	1.46	52.72	298471	76.97
	Select-prefix [11]-4x16	137511	2.10	43.98	288774	92.36
	Select-prefix [11]-2x32	152420	2.24	45.76	341421	102.50
	CCS-4x16	104900	1.80	27.54	188820	49.57
	CCS-16x4	159278	1.75	32.29	278737	56.51

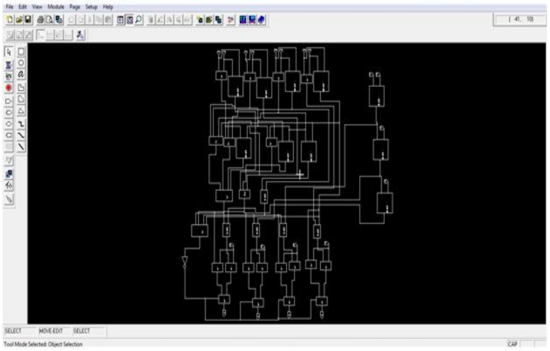


Fig. 7 Chip layout for CCS diminished-one modulo 2^{4+1} adder

IV. CONCLUSION

A new CCS diminished-one modulo $2^n + 1$ adder has been introduced and developed to derive the most compromising design in terms of area, delay and power. For a large bit- width requirement, our CCS modular adder is realized by the combination of CCS addition blocks, CCG and MUX to lead into the simple and regular circuit structure. Based on static UMC 180-nm CMOS technology, the VLSI implementation of CCS modular adder indeed has better area-delay and delay-power performances over those of the previous designs.

ACKNOWLEDGEMENT

I thank my friends everyone who gave me support in all the ways for designing my project with various ideas and also I thank my guide Mr.B.M.Prabhu who has been with me in all my steps in this project.

REFERENCES

[1]. N. S. Szabo and R. I. Tanaka, Residue Arithmetic and Its Applications to Computer Technology.New York: McGraw Hill, 1967.

[2]. M. A. Sonderstrand et al., Residue Number System Arithmetic: Modern Applications in Digital Signal Processing.New York: IEEE Press, 1986.

[3]. A. B. Premkumar, E. L. Ang, and E. M.-K. Lai, “Improved memory- less RNS forward \converter based on the periodicity of residues,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 53, no. 2, pp. 133–137, Feb.2007.

[4]. Y. Wang, X. Song, M. Aboulhamid, and H. Shen, “Adder-based residue to binary number converters for (2^n-1, 2^n, 2^n+1),” IEEE Trans. Signal Process., vol. 50, no. 7, pp. 1772– 1779, Jul. 2002.

[5]. B. Cao, C. H. Chang, and T. Srikanthan, "An efficient reverse converter for the 4-moduli set (2^n-1, 2^n, 2^n+1, 2^{2n}+1) based on the new chinese remainder theorem," IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol. 50, no. 10, pp. 1296–1303, Oct.2003.

[6]. P. V. Ananda Mohan and A. B. Premkumar, "RNS-to-binary converters for two four- moduli sets(2^n-1, 2^n, 2^n+1, 2^n+1-1) and (2^n-1, 2^n, 2^n+1, 2^n+1+1) ,"IEEE Trans. Circuits Syst. I, Reg. Pape rs,vol. 54, no. 6, pp. 1245–1254, Jun. 2007.

[7]. L. M. Leibowitz, "A simplified binary arithmetic for the fermat number transform," IEEE Trans. Acous., Speech, Signal Process., vol. 24, pp. 356–359, 1976.

[8]. R.Zimmermann,"Binary modulo bit Adder Architecture for Cell-Based VLSI and their Synthesis," Ph.D dissertation, Swiss Federal Inst. of Technology, Zurich, Switzerland, 1997.

[9]. R.Zimmermann,"Efficient optimized VLSI implementation of Modulo 2^n±1 addition and multiplication," in Proc.14th IEEE SympComputer Arithmetic, Apr. 1999, pp. 158–167.