INTERNATIONAL JOURNAL FOR TRENDS IN ENGINEERING & TECHNOLOGY
VOLUME 12 ISSUE 1 – APRIL 2016 - ISSN: 2349 - 9303

Design of Low Power Speculative Han-Carlson Adder

S.Sangeetha
II ME - VLSI Design, Akshaya College of Engineering and Technology, Coimbatore
sangeethasoctober@gmail.com

S.Kamatchi
Assistant Professor, Department of ECE, Akshaya College of Engineering and Technology, Coimbatore
kamatchi.be@gmail.com

Abstract—In this paper we have proposed speculative Han-Carlson adder. The proposed adder Employs speculation: the exact mathematic function is replaced with an approximated one that is faster and gives the correct result most of the time, but not for all time. The approximated adder is augmented with an error detection network that asserts an error signal when speculation fails. The speculative adder to reduce delay and power consumption compared to non-speculative adder and simulated using ModelSim 6.3f. Delay and power consumption of non-speculative and speculative Han-Carlson adder were analyzed using Xilinx ISE 8.1i.

***Index Terms*—**Addition, mathematic, non-speculative adder, speculative adder, delay, power consumption.

1. INTRODUCTION

Adders are fundamental functional units in computer mathematic. Binary adders are used in microprocessor for addition and subtraction functions as well as for floating point multiplication and division. Therefore adders are fundamental mechanisms and improving their performance is one of the most important challenges in digital designs.

High speed adders are based on parallel-prefix architectures [1], [2], including Brent-Kung [3], Kogge-Stone [4], Sklansky [5], Han-Carlson [6], Ladner-Fischer [7], Knowles [8]. These standard architectures operate with fixed latency. Better results can be attained by using Speculative adders, that have been recently proposed in literature [9]. Proposed adder employs speculation: the exact mathematic function is substituted with an approximated one that is faster and gives the correct result large amount of the time, but not for all time. The approximated adder is augmented by means of an error detection network that asserts an error signal when speculation fails. In this case, another clock cycle is required to obtain the correct result with the help of a correction stage.

2. LITERATURE REVIEW

2.1 Kogge-Stone Adder

Kogge-Stone Adder (KSA) is a parallel-prefix form of carry look ahead adder. Fig. 1. shows the 16-bit KSA was developed by [4] P. M. Kogge and H. S. Stone which they published in1973. KSA is a high-speed adder design as it generate carry signal in O(log2 n) time and has the best performance in industry. In KSA, carry is computed fast by calculating it in parallel at the cost of increased area. One of the main limitations of the work is that wiring increases with respect to number of bits thus increasing the chip area.

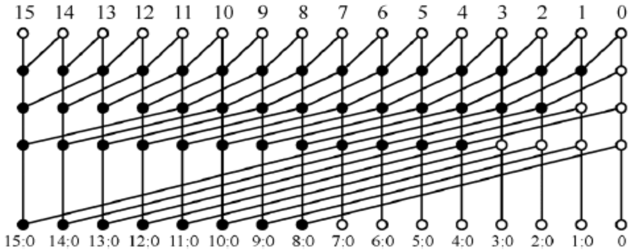


Fig. 1. 16-bit Kogge-Stone Adder

2.2 Brent-Kung Adder

Brent-Kung Adder (BKA) was developed by [3] R. P. Brent and H. T. Kung which they published in 1982. BKA has maximum logic depth and avoid explosion of wires. The work is mainly focused on area minimization. The BKA performs odd computation first and then even. Fig. 2. shows buffers used to minimize the fan-out and loading on the gates, but the buffers are generally misplaced. The basic blocks used in this case are gray and black cells.

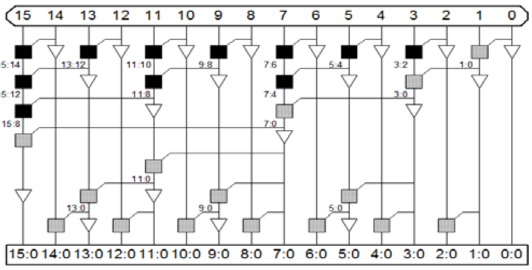


Fig. 2. 16-bit Brent-Kung Adder

2.3 Ladner-Fischer Adder

Ladner- Fischer Adder (LFA) was developed by R. E. Ladner and M. J. Fischer in 1980. In LFA prefix tree is a structure that assembles between Brent-Kung and Sklansky prefix tree. The LFA [7] has minimum logic depth but it has large fan-out. Ladner- Fischer adder has carry operator nodes. Fig. 3. shows the 16-bit LFA.

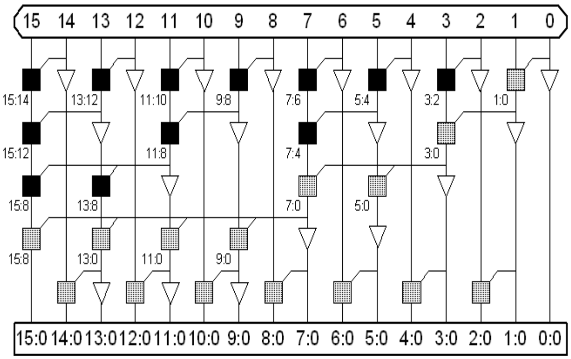


Fig. 3. 16-bit Ladner-Fischer Adder

2.4 Sklansky Adder

Sklansky Adder (SA) [5] was developed by J. Sklansky in 1960. SA's structure is the simplest along with the prefix adders. In SA , binary trees of cells produce all the carry input bits simultaneously. Fig. 4. shows the 16-bit SA.

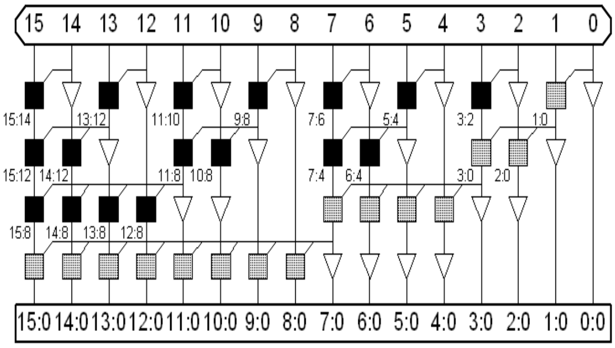


Fig. 4. 16-bit Sklansky Adder

The Sklansky or divide-and conquer tree reduces the delay to log2n stages by computing intermediate prefixes. This comes at the expense of fan-outs that double at each level. These high fan-out cause poor performance on wide adders unless the high fan-out gates are suitably sized, or critical signals are buffered before being used for intermediate prefixes. Transistor sizing can cut into the regularity of the design because various sizes of each cell are required although the larger gates can spread into adjacent columns.

2.5 Approximation Circuits

Approximate circuits was developed by S.-L. Lu in 2004. This work is mainly focused in achieving reduction in delay. Approximate adders [9] are used in the microprocessors that developed shorter carry chains; hence the speed can be increased. Handshaking-overhead caused during the completion of execution at each stage is a major limitation.

3. EXISTING WORK

This work 16-bit non-speculative Han-Carlson adder (HCA) [6] is presented. It can be subdivided in three stages: pre-processing, prefix-processing and post-processing. Fig. 5. shows the 16-bit non-speculative HCA.

Pre-processing stage

In this stage A & B are inputs. The propagate and generate signals are given by the equations (1) & (2).

$$P_i = A_i \text{ xor } B_i \qquad (1)$$
$$G_i = A_i . B_i \qquad (2)$$

Prefix-processing stage

In this stage propagate and generate are computed by using equations (3) & (4).

$$P_{(i:k)} = P_{(i:j)} \, . \, P_{(j-1:k)} \qquad (3)$$
$$G_{(i:k)} = G_{(i:j)} + (G_{(j-1:k)} \, . \, P_{(i:j)}) \qquad (4)$$

Post-processing stage

In this stage carry and sum are computed by using equations (5) & (6).

$$C_i = G_{[i:0]} + P_{[i:0]}C_{-1} \qquad (5)$$
$$S_i = P_i \text{ xor } C_{i-1} \qquad (6)$$

The Han-Carlson trees are the family of networks between Kogge-Stone and Brent-Kung. Han-Carlson adder can be viewed as a sparse version of Kogge-Stone adder. This scheme is different from Kogge-Stone scheme in the sense that these performs carry-merge functions on even bits and generate/propagate operation on odd bits. At the end, these odd bits recombine with even bits carry signals to produce the true carry bits.

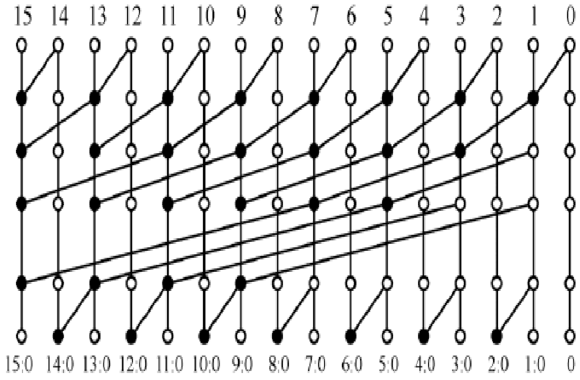


Fig. 5. 16-bit Non-Speculative Han-Carlson Adder

This adder has five stages in which the middle three stages are resembles with the Kogge-Stone structure. The advantage of this adder is that it uses much less cells and its shorter span wires than the Kogge-Stone adder and thus there is reduction in complexity at the cost of an additional stage for carry-merge path. The main drawback of this adder is that increased power consumption.

4. PROPOSED WORK

This work 16-bit speculative Han-Carlson adder (HCA) is proposed. Speculative prefix adders can be subdivided in five stages: pre-processing, speculative prefix-processing, post-processing, error detection and error correction. The error correction stage is off the critical path, as it has two clock cycles to obtain the exact sum when speculation fails.

4.1 Pre-processing stage

The propagate and generate signals are computed as in equations (1) & (2).

4.2 Speculative Prefix-processing stage

The last Kogge-Stone row of the n = 16 bit graph is shortened, resulting in a speculative prefix-processing stage with K = 8 is shown in Fig. 6. The length of the propagate chains is K = 8 only for i = 9,11,13,15, while for i = 10,12,14 the propagate chain length is K+1 = 9. The computed propagate and generate signals for the speculative Han-Carlson architecture are:

$$\begin{cases} (G, P)_{[i:0]} & \text{for } i \leq K \\ (G,P)_{[i:i-K+1]} & \text{for } i > K, \text{ i odd} \\ (G, P)_{[i:i-K]} & \text{for } i > K, \text{ i even} \end{cases} \qquad (7)$$

Propagate and generate signals are computed as in equations (3) & (4).

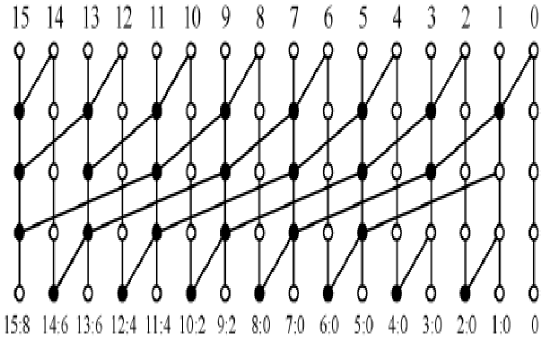


Fig. 6. Han-Carlson speculative prefix-processing stage

4.3 Post-processing stage

If no error occur means post-processing is performed then carry and sum are computed using equation (5) and (6). Otherwise error detection and correction are needed.

4.4 Error detection stage

The conditions in which at least one of the approximate carries is wrong are signaled by the error detection stage which consists of AND and OR gates as shown in Fig. 7. If error occurs OR gate indicates error signal as '1' otherwise zero.

$$EHC=p[9:2]g[1:0]+p[11:4]g[3:2]+....+p[15:8]g[7:6] \qquad (8)$$

4.5 Error correction stage

If error occurs then it will be corrected in error correction stage which is composed of black cells. Finally sum and carry are obtained at post-processing stage that is equivalent to the non-speculative Han-Carlson adder.

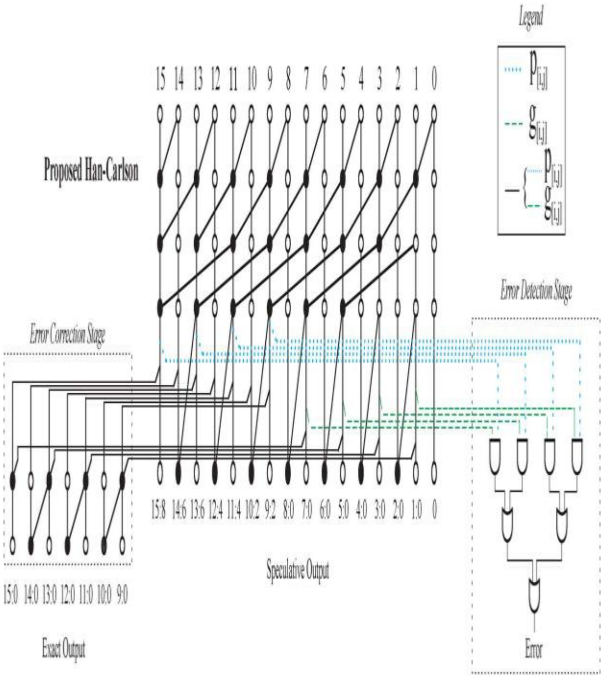
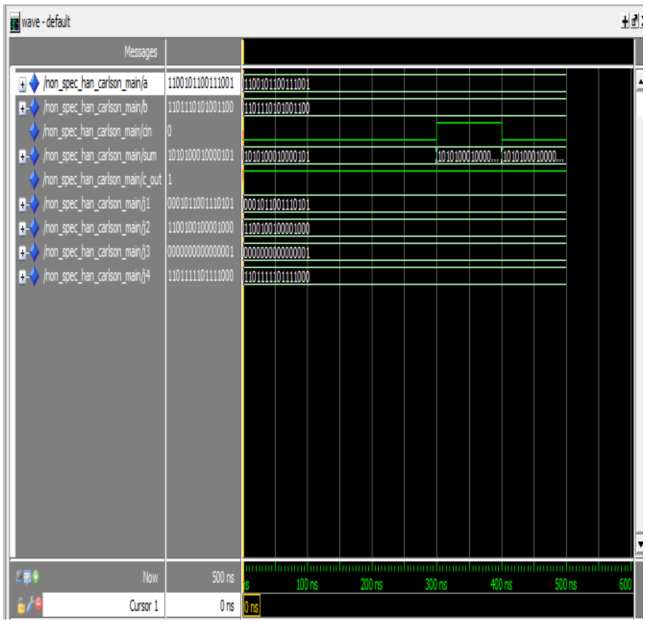


Fig. 7. 16-bit Speculative Han-Carlson Adder with error detection and correction

Speculative HCA uses smaller number of prefix operations by changing the number of stages amongst KSA and BKA. The main advantage is that reduction in delay compared to non-speculative HCA.

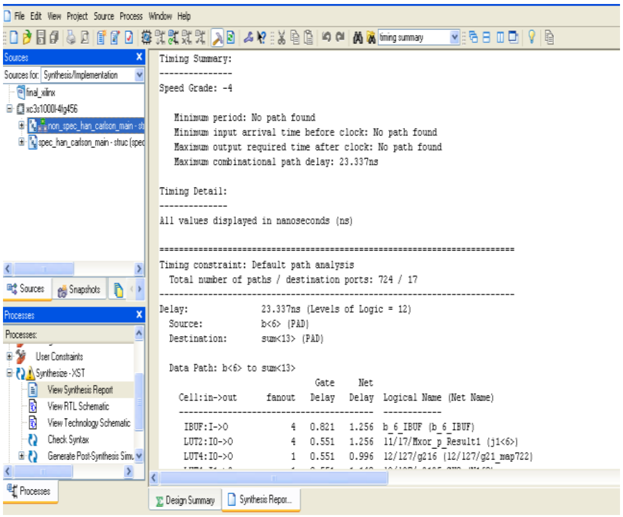
5. SIMULATION RESULTS

The simulation results for the comparison of 16-bit non-speculative and speculative HCA were simulated using ModelSim as shown in figure. Power analyzed using Xilinx 8.1i as shown in Table I.



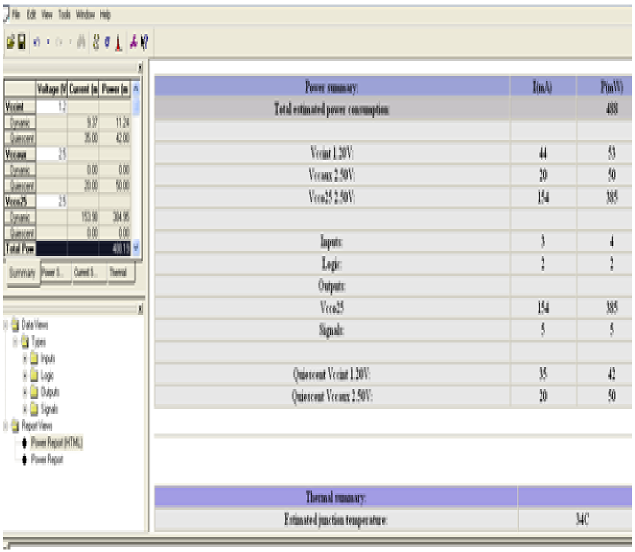
```
/non_spec_han_carlson_main/a      1100101100111001   1100101100111001
/non_spec_han_carlson_main/b      1101110101001100   1101110101001100
/non_spec_han_carlson_main/cin    0
/non_spec_han_carlson_main/sum    1010100010000101   1010100010000101   1010100010000...   1010100010000...
/non_spec_han_carlson_main/c_out  1
/non_spec_han_carlson_main/j1     0001011001110101   0001011001110101
/non_spec_han_carlson_main/j2     1100100100001000   1100100100001000
/non_spec_han_carlson_main/j3     0000000000000001   0000000000000001
/non_spec_han_carlson_main/j4     1101111101111000   1101111101111000
Now        500 ns
Cursor 1   0 ns    0 ns
```

Fig. 8. Simulation result of 16-bit non-speculative Han-Carlson Adder



```
Timing Summary:
---------------
Speed Grade: -4

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 23.337ns

Timing Detail:
--------------
All values displayed in nanoseconds (ns)

=========================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 724 / 17
-------------------------------------------------------------------------
Delay:               23.337ns (Levels of Logic = 12)
  Source:            b<6> (PAD)
  Destination:       sum<13> (PAD)

  Data Path: b<6> to sum<13>
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             4   0.821   1.256  b_6_IBUF (b_6_IBUF)
     LUT2:I0->O            4   0.551   1.256  11/17/Mxor_p_Result1 (j1<6>)
     LUT4:I0->O            1   0.551   0.996  12/127/g216 (12/127/g21_map722)
```

Fig. 9. Timing summary of 16-bit non-speculative Han-Carlson Adder

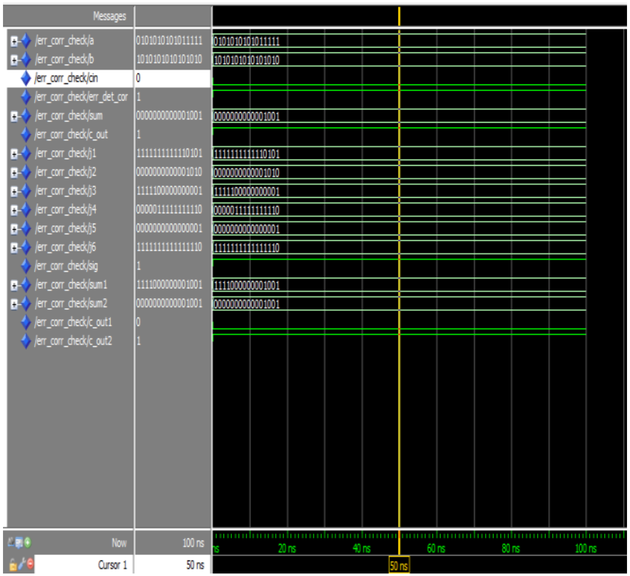


	Voltage (V)	Current (mA)	Power (mW)
Vccint	1.2		
Dynamic		9.37	11.24
Quiescent		35.00	42.00
Vccaux	2.5		
Dynamic		0.00	0.00
Quiescent		20.00	50.00
Vcco25	2.5		
Dynamic		153.98	384.95
Quiescent		0.00	0.00
Total Power			488.19

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		488
Vccint 1.20V:	44	53
Vccaux 2.50V:	20	50
Vcco25 2.50V:	154	385
Inputs:	3	4
Logic:	2	2
Outputs:		
Vcco25	154	385
Signals:	5	5
Quiescent Vccint 1.20V:	35	42
Quiescent Vccaux 2.50V:	20	50

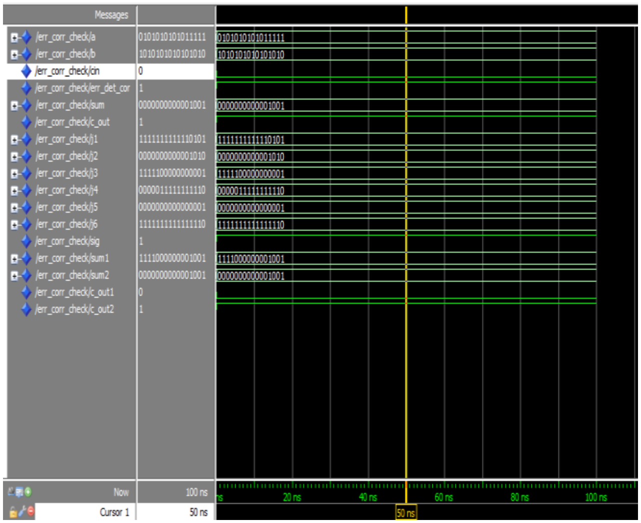
Thermal summary:	
Estimated junction temperature:	34C

Fig. 10. Power summary of 16-bit non-speculative Han-Carlson Adder



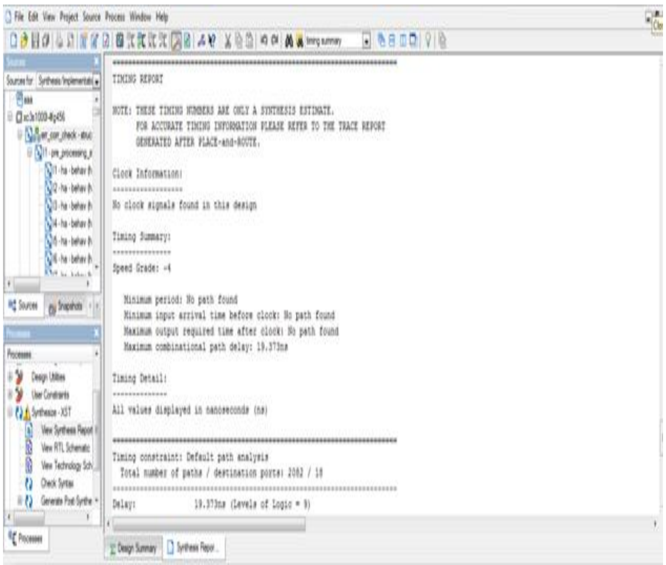
```
Messages
/err_corr_check/a            0101010101011111
/err_corr_check/b            1010101010101010
/err_corr_check/cin          0
/err_corr_check/err_det_cor  1
/err_corr_check/sum          0000000000001001
/err_corr_check/c_out        1
/err_corr_check/j1           1111111111110101
/err_corr_check/j2           0000000000001010
/err_corr_check/j3           1111100000000001
/err_corr_check/j4           0000011111111110
/err_corr_check/j5           0000000000000001
/err_corr_check/j6           1111111111111110
/err_corr_check/sig          1
/err_corr_check/sum1         1111000000001001
/err_corr_check/sum2         0000000000001001
/err_corr_check/c_out1       0
/err_corr_check/c_out2       1
```

Fig. 11. Simulation result of 16-bit speculative Han-Carlson Adder without error



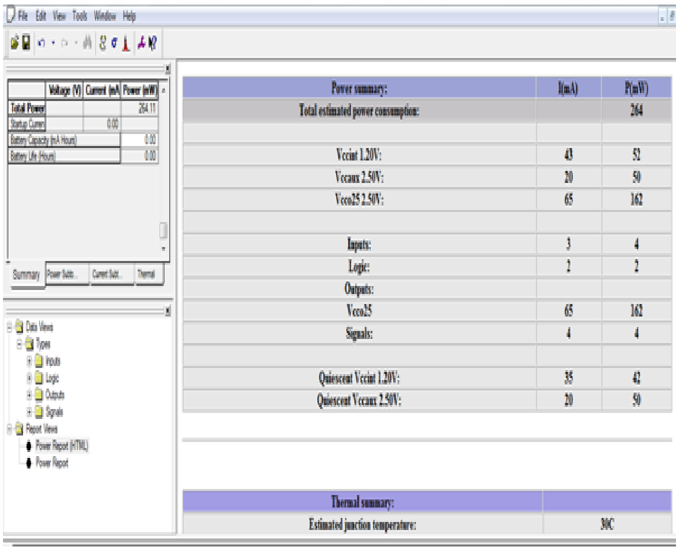
```
Messages
/err_corr_check/a            0101010101011111
/err_corr_check/b            1010101010101010
/err_corr_check/cin          0
/err_corr_check/err_det_cor  1
/err_corr_check/sum          0000000000001001
/err_corr_check/c_out        1
/err_corr_check/j1           1111111111110101
/err_corr_check/j2           0000000000001010
/err_corr_check/j3           1111100000000001
/err_corr_check/j4           0000011111111110
/err_corr_check/j5           0000000000000001
/err_corr_check/j6           1111111111111110
/err_corr_check/sig          1
/err_corr_check/sum1         1111000000001001
/err_corr_check/sum2         0000000000001001
/err_corr_check/c_out1       0
/err_corr_check/c_out2       1
```

Fig. 12. Simulation result of 16-bit speculative Han-Carlson Adder with error



```
*************************************************************************
TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
      GENERATED AFTER PLACE-and-ROUTE.

Clock Information:
------------------
No clock signals found in this design

Timing Summary:
---------------
Speed Grade: -4

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 19.373ns

Timing Detail:
--------------
All values displayed in nanoseconds (ns)

=========================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 2082 / 18
-------------------------------------------------------------------------
Delay:               19.373ns (Levels of Logic = 9)
```

Fig. 13. Timing summary of 16-bit speculative Han-Carlson Adder



	Voltage (V)	Current (mA)	Power (mW)
Total Power			264.11
Startup Current		0.00	
Battery Capacity (mA Hours)			0.00
Battery Life (Hours)			0.00

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		264
Vccint 1.20V:	43	52
Vccaux 2.50V:	20	50
Vcco25 2.50V:	65	162
Inputs:	3	4
Logic:	2	2
Outputs:		
Vcco25	65	162
Signals:	4	4
Quiescent Vccint 1.20V:	35	42
Quiescent Vccaux 2.50V:	20	50

Thermal summary:	
Estimated junction temperature:	30C

Fig. 14. Power summary of 16-bit speculative Han-Carlson Adder

TABLE I.
COMPARISION OF NON-SPECULATIVE AND SPECULATIVE HCA

Adder	Delay (ns)	Power consumption (mW)

16-bit non-speculative HCA	23.337	19.373
16-bit speculative HCA	488	264

CONCLUSION

In this paper comparison of 16-bit non-speculative and speculative Han-Carlson adder were analyzed based on parallel-prefix topology. The main advantage of this proposed speculative HCA is that reduction in delay and power.

FUTURE WORK

Future work includes speculative HCA with concurrent subtraction technique to minimize more power for performing binary addition.

REFERENCES

[1] I. Koren, *Computer Arithmetic Algorithms*. Natick, MA, USA: A K Peters, 2002.

[2] R. Zimmermann, “Binary adder architectures for cell-based VLSI andtheir synthesis,” Ph.D. thesis, Swiss Federal Institute of Technology, (ETH) Zurich, Zurich, Switzerland, 1998, Hartung-Gorre Verlag.

[3] R. P. Brent and H. T. Kung, “A regular layout for parallel adders,” *IEEE Trans. Comput.*, vol. C-31, no. 3, pp. 260–264, Mar. 1982.

[4] P. M. Kogge and H. S. Stone, “A parallel algorithm for the efficient solution of a general class of recurrence equations,” *IEEE Trans. Comput.*, vol. C-22, no. 8, pp. 786–793, Aug. 1973.

[5] J. Sklansky, “Conditional-sum addition logic,” *IRE Trans. Electron. Comput.*, vol. EC-9, pp. 226–231, Jun. 1960.

[6] T. Han and D. A. Carlson, “Fast area-efficient VLSI adders,” in *Proc. IEEE 8th Symp. Comput. Arith. (ARITH)*, May 18–21, 1987, pp. 49–56.

[7] R. E. Ladner and M. J. Fischer, “Parallel prefix computation,” *J. ACM*, vol. 27, no. 4, pp. 831–838, Oct. 1980.

[8] S. Knowles, “A Family of Adders,” in *Proc. 14th IEEE Symp. Comput. Arith.*, Vail, CO, USA, Jun. 2001, pp. 277–281.

[9] S.-L. Lu, “Speeding up processing with approximation circuits,” *Computer*, vol. 37, no. 3, pp. 67–73, Mar. 2004.

[10] T. Liu and S.-L. Lu, “Performance improvement with circuit-level speculation,” in *Proc. 33rd Annu. IEEE/ACM Int. Symp. Microarchit. (MICRO-33)*, 2000, pp. 348–355.

[11] N. Zhu, W.-L. Goh, and K.-S. Yeo, “An enhanced low-power high speed Adder For Error-Tolerant application,” in *Proc. 2009 12th Int.Symp. Integr. Circuits (ISIC '09)*, Dec.14–16, 2009, pp. 69–72.

[12] S. M. Nowick, “Design of a low-latency asynchronous adder using speculative completion,” *IEE Proc. Comput. Digit. Tech.*, vol. 143, no.5, pp. 301–307, Sep. 1996.

[13] A. K. Verma, P. Brisk, and P. Ienne, “Variable Latency Speculative Addition: A New Paradigm for Arithmetic Circuit Design,” in *Proc. Design, Autom., Test Eur. (DATE '08)*, Mar. 2008, pp. 1250–1255.

[14] A. Cilardo, “A new speculative addition architecture suitable for two's complement operations,” in *Proc. Design, Autom., Test Eur. Conf. Exhib. (DATE '09)*, Apr. 2009, pp. 664–669.

[15] K. Du, P. Varman, and K. Mohanram, “High performance reliable variable latency carry select addition,” in *Proc. Design, Autom., Test Eur. Conf. Exhib. (DATE '12)*, Mar. 2012, pp. 1257–1262.

[16] S. K. Mathew, R. K. Krishnamurthy, M. A. Anders, R. Rios, K. R.Mistry, and K. Soumyanath, “Sub-500-ps 64-b ALUs in 0.18- m SOI/ bulk CMOS: Design and scaling trends,” *IEEE J. Solid-State Circuits*, vol. 36, no. 11, pp. 1636–1646, Nov. 2001.

[17] B. Parhami, Computer Arithmetic: Algorithms and Hardware Design. New York: Oxford Univ. Press, 2000.