A Hierarchical Design of 128 Bit Carry Lookahead Adder in 65 nm CMOS Technology

Kishore Prabhala, Prabhandhakam Sangameswara Raju

Abstract: As One Giga Hertz microprocessors power mobiles, laptops, tablets and personal computers in last few years, there is a massive need to reduce the number cycles to do addition which plays a significant role in Arithmetic Logic Unit (ALU) or Digital Signal Processing (DSP). The complexity of carry propagation is the critical variable once the designs requires addition over 32 bits. A hierarchical design has been developed to find Carry out at 16-bit stage from Propagate and Generate techniques from a 4-bit stage of Carry Lookahead Adder (CLA), so called Carry Lookahead Logic (CLL). Four blocks of CLL have been used to create another CLL block at a 16-bit level and similarly at 64 bit level and 128 bit level. A 65 nm CMOS technology library from Microwind used to simulate from logic to circuit level for the hierarchical design of 128 bit CLA and compared with 90 nm technology.

Keywords: 65 nm CMOS, Carry Lookahead Adder, Hierarchical, Microwind, and VLSI.

I. INTRODUCTION

A Nano second (ns) cycle time is common in many mobiles, laptops, tablets and personal computers which are sold close to three billion in last four years. This translates to a Giga Hertz (GHz) Microprocessor, but over 10 GHz microprocessors are powering end devices in last two year but addition which plays a key role in Arithmetic Logic Unit (ALU) or Digital Signal Processing (DSP). An adder has to be designed, verified, simulated at circuit level and layout at 65 nm to 22 nm for 128 bit data processing to meet the growing demand. Even technology moving towards 14 nm to 10 nm with 256 bit data processing is inevitable in next five years in all computation devices.

Addition has been done in two different ways. First bit addition or Least Significant Bit (LSB) has only two bits for inputs A and B from 4 to 128 bit. This is done by “Half Adder” (HA), next bit addition will be done by three bits, two bits from inputs A and B while the third bit comes as carry in from the carry out at LSBL as shown in table 1. This has been called “Full Adder” and truth table has been shown in table 2. Carry is generated when both bit are high in HA from truth table in table 1 or in FA at table 2. But in FA, Carry is generated for any combination of all three input A, B, Carry in (Cin) or A and B or A and Cin or Cin being High or “1” as shown in table 2 (Mano, 2002). The logic diagram has been shown in figure 2. Any design has to go through Karnaugh Map or K-Map, a standard process anyone can’t skip, then get logical equations and finally define logic such CMOS - Complimentary Metal Oxide Silicon or ECL – Emitter Coupled Logic or TTL – Transistor Transistor Logic (Weste, 1993). The Logic process has been done by Register Transistor Logic (RTL) in Electronic Design Automation (EDA) Tools offered by Microwind or Mentor or Cadence or Synopsys (Uyemura, 2006).

Table 1: a) Truth table, b) K-Map for Half-Adder

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Sum</th>
<th>Co</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

K- Map For Sum

```
A  B  Sum  Co
0  1  0   0
1  0  0   0
```

K- Map For Carry Out

```
A  B  Carry Out
0  1  1
1  0  0
```

The equation for Sum of Products for HA from K-Map

\[ S = AB' + A'B \]  
\[ S = AB + A'B + A'B \]  
\[ S = A + B \]  
\[ A = S \]  
\[ B = S \]  
\[ C = S \]

The symbol ⊕ stands for Exclusive OR (ExOR). A’ is the complement of A. The equation for Carry Out of a HA from K-Map

\[ \text{Carry Out} = AB \]  

Once the equations are obtained the logic logic design need to be done. Sum is generated by ExOR and Carry out generated by an AND.

Table 2: a) Truth table & b) K-Map for Full Adder (FA)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Cin</th>
<th>Sum</th>
<th>Co</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

b) K- Map For Sum of Full Adder

Revised Manuscript Received on January 05, 2020.

* Correspondence Author

Kishore Prabhala*, Research Scholar, EEE PhD, Rayalaseema University, Kurnool, Senior Member IEE, Principal, PLNM Degree College, Opposite Acharya Nagarjuna University Mens Hostel, Nagarjuna Nagar – 522 510, Guntur Dist., AP, India.,

Prof. Prabhandhakam Sangameswara Raju, EEE, SVU Engineering College, Sri Venkateswara University, Tirupati – 517 502, AP.
The equation for Sum of Products from K-Map has been taken with individual terms for a value of “1”

\[ S = \overline{A'B'Cin'} + A'B'Cin + A'BCin' + ABCin \]

\[ = (A + B)'Cin' + (A' + B)Cin \]

\[ = A \oplus B \oplus Cin \]

----- eq. 1.3

the symbol \( \oplus \) stands for Exclusive OR, logically when inputs are not equal output will be ONE.

Equation for Carry out is

\[ C_o = AB + ACin + BCin \text{ or } AB + (AB)Cin \]

----- eq. 1.4

II. LOGIC DESIGN OF 4-BIT CLA

From 2\text{nd} bit onwards, there will be three bits needed to added and carry ripple through each adder stage. A propagation delay could have 2n gates if this design technique has to be used as Sum is generated from two “ExOR gates”, shown in figure 1.

Carry Lookahead Adder (CLA) has the fastest to process addition and the design technique has been based on the principle of calculating “Carry Out” from four bit with Propagate or Generate terms (Mano, 2002). These techniques reduces the propagation delay of carry with more logic as defined by equations 2.2 to 2.7 but the area or number logic gates will be increased.

“Generate” (G) will calculate Carry out if both A and B are “High” and “Propagate” (P) will calculate Carry at each stage. The generation values and propagation values are computed based on the carry out if A or B high but only if Carry in “High” as seen from equation 2.1.

\[ \text{Sum} = A \oplus B \oplus \text{Cin}, \text{Cout} = AB + \text{Cin}(A+B) \text{ eq. 2.1} \]

Generate, \( G = AB \)

----- eq. 2.2

Propagate, \( P = A + B \)

----- eq. 2.3

For a 4 bit CLA, the Generate and Propagate terms are

\[ G_0 = A_0B_0 \text{ and } P_0 = A_0 + B_0 \]
\[ G_1 = A_1B_1 \text{ and } P_1 = A_1 + B_1 \]
\[ G_2 = A_2B_2 \text{ and } P_2 = A_2 + B_2 \]
\[ G_3 = A_3B_3 \text{ and } P_3 = A_3 + B_3 \]

Now the carry out at each stage of 4 bit CLA is

\[ C[1]_0 = G_0 + P_0 \text{ Cin} \]
\[ C[2]_1 = G_1 + P_1 + G_0 + P_0 \text{ Cin} \]
\[ C[3]_2 = G_2 + P_2 + G_1 + P_1 \text{ Cin} + G_0 + P_0 \text{ Cin} \]
\[ \text{Cout} = G_3 + P_3 + G_2 + P_2 \text{ Cin} + G_1 + P_1 \text{ Cin} + G_0 \text{ Cin} \]

----- eq. 2.7

Carry generation from each stage has been calculated from the above equations and finally Carry out at the end of 4 bits will be developed as Carry Lookahead Logic (CLL) shown in the figure 2 from equation 2.7 (Mano, 2002)(Zhuang, 1992).

For a 4 bit adder for CLA

\[ P_0, G_0 \text{ ty } P_1, G_1 \text{ and } A_P, G_2 \text{ (CLA)} P_3 G_3 \text{ parts}. \]

One is addition and other is Carry Out logic. Full adder has been designed with four NAND gates as EXOR in two stages with \( \text{Cin rate} (G) \) coming from first NAND (it is G’ or G Bar) and Propagate (P) coming from first EXOR output, A and B as inputs. This is shown in figure 3. The input carry-in will feed the second EXOR gate designed after four NANDs in the first stage shown in the bottom of the figure 3.

At bit 1, the carry out is generated from G0’ from A0 and B0 stage (First NAND output), then Cin and P0 has taken from the fifth NAND and a NAND on top this two ExOrs have been shown with connection from G0’ and

Figure 1: Logic Diagram for Sum and Carry out, Full Adder

A Very Large Scale Integrate (VLSI) design over thousands of Integrated Circuits in Metal Oxide Semiconductor (MOS) requires automation of logic with Electronic Design Automation (EDA) tools with Mentor or Cadence or Synopsys or Microwind. Generally, logic is defined as a behavioural model with respect to truth table as defined in table 2 for this case “Full Adder”. Verilog is a open source logic accepted by any EDA tool vendor and after RTL, a logic diagram will be developed (Weste, 1993). A Verilog code for a Full adder is

```verilog
module fulladder(a,b,c,sum,carry);
    input a,b,c;
    output sum,carry;
    assign sum = a \& b \& c;
    assign carry = (a \& b) | (b \& c) | (c \& a);
endmodule
```

Figure 2: Carry Lookahead Logic (CLL) for Carry Out for a 4 bit adder for CLA

P_0, G_0 ty [P_1, G_1 zed A_P, G_2 (CLA)] P_3 G_3 parts. One is addition and other is Carry Out logic. Full adder has been designed with four NAND gates as EXOR in two stages with Cin rate (G) coming from first NAND (it is G’ or G Bar) and Propagate (P) coming from first EXOR output, A and B as inputs. This is shown in figure 3. The input carry-in will feed the second EXOR gate designed after four NANDs in the first stage shown in the bottom of the figure 3.

At bit 1, the carry out is generated from G0’ from A0 and B0 stage (First NAND output), then Cin and P0 has taken from the fifth NAND and a NAND on top this two ExOrs have been shown with connection from G0’ and
null
A Hierarchical Design of 128 Bit Carry Lookahead Adder in 65 nm CMOS Technology

It has sixteen inputs from A_{15} to A_0, B_{15} to B_0 and Cin along with sixteen outputs for Sum, \textit{Sum}_{15 to 0} and a Generate term and a Propagate term as shown in figure 4.

In a Hierarchical CLA Design, ‘Propagate’ term has been redefined from equation 2.9 as
\[
P_{0,3} = P_3 P_2 P_1 P_0
\]
The “Generate” term from equation 3.0 has been redesigned as
\[
G_{0,3} = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0
\]
Now the Generate as output term at 16-bit stage
\[
\text{Generate}_{16} = G_{15} + G_{14} + G_{13} + G_{12}
\]
Propagate \text{Propagate}_p = P_{15} P_{14} P_{13} P_{12}
Generate \text{Generate}_g = G_15 + G_14 + G_13 + G_12

b) Design of 64-bit Hierarchical CLA

A Hierarchical design of 64-bit CLA takes “four 16-bit CLA with CLL” as shown in figure 5, designed from 4-bit CLAs with inputs Cin, A_{15} to A_0, B_{15} to B_0 driving first CLA and A_31 to A_16 and B_31 to B_16 driving second CLA, with inputs A_47 to A_32 and B_47 to B_32 driving third CLA, and with inputs A_63 to A_48 and B_63 to B_48 driving the last and fourth CLA as shown in figure 6. Each 64-bit CLA has a CLL block creating three outputs, a Generate and a Propagate term besides Carry out. A Carry Out and 64 outputs for \text{Sum}_{63 to 0} have been created.

In all eight terms namely P_15, G_15, P_31, G_31, P_47, G_47, P_63, G_63 have been created with equations 4.1 and 4.2.

A block diagram for a 16-bit CLA with CLL has been shown in figure 5 which will used to design for 64 bit CLA with CLL.

At first stage:
- Propagate \text{Propagate}_1 = P_3 P_2 P_1 P_0
- Generate \text{Generate}_1 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0

At second stage:
- Propagate \text{Propagate}_2 = P_3 P_2 P_1 P_0
- Generate \text{Generate}_2 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0

At third stage:
- Propagate \text{Propagate}_3 = P_3 P_2 P_1 P_0
- Generate \text{Generate}_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0

At fourth stage:

\[
\text{Propagate}_{16} = P_3 P_2 P_1 P_0
\]
\[
\text{Generate}_{16} = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0
\]
\[
\text{Propagate}_p = P_{15} P_{14} P_{13} P_{12}
\]
\[
\text{Generate}_g = G_{15} + G_{14} + G_{13} + G_{12}
\]

Figure 4: Hierarchical CLA for 16 bits from FOUR

Figure 5: Block Diagram for Hierarchical 16-bit CLA

Generate Sum terms and then propagate with generate terms with four 4-bit CLA logic as shown in figure 5 from figure 4.

At first stage:
- Propagate \text{Propagate}_1 = P_3 P_2 P_1 P_0
- Generate \text{Generate}_1 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0

At second stage:
- Propagate \text{Propagate}_2 = P_3 P_2 P_1 P_0
- Generate \text{Generate}_2 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0

At third stage:
- Propagate \text{Propagate}_3 = P_3 P_2 P_1 P_0
- Generate \text{Generate}_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0

At fourth stage:

\[
\text{Propagate}_{16} = P_3 P_2 P_1 P_0
\]
\[
\text{Generate}_{16} = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0
\]
\[
\text{Propagate}_p = P_{15} P_{14} P_{13} P_{12}
\]
\[
\text{Generate}_g = G_{15} + G_{14} + G_{13} + G_{12}
\]

Figure 6: Block Diagram for Hierarchical 64-bit CLA

Figure 7: Block Diagram for Hierarchical 128-bit CLA

Published By:
Blue Eyes Intelligence Engineering & Sciences Publication

Acknowledgments

©BEIESP

DOI: 10.35940/ijitee.C8757.019320

Retrieval Number: C8757019320/2020©BEIESP
Finally there are two Propagate and Generate terms and would generate Carry out as
\[ \text{Carry Out} = G_{127,64} + G_{63,0} \times P_{127,64} + P_{127,64} \times P_{63,0} \times \text{Cin}, \]
this is final Carry Out of 128 bit Hierarchical CLA designed in CMOS.

The final Carry out has been implemented with a 2 input NAND.

V. RESULTS AND DISCUSSION

The delay for Carry out can be in a logic form as One NAND (2 input) for P or G at 1-bit stage, two NANDs (4 input) at 4-bit stage. Sum is generated from six NAND gates (2 inputs) but carry in comes from previous stage using carry out equation G+PCin.

At 4-bit stage the terms Propagate \( P = P_{3P2P1P0} \) or Generate \( G = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0 \) are designed with two NANDs (4 input) and similarly at 64-bit stage and two NANDs (3 input) at final stage with overall delay of 24 nMOS transistors, for a 2 input NAND has two nMOS transistors in series. A four bit CLA with NAND gates has been implemented in CMOS along with CLA logic as shown in figure 8. L and W of 90 nm and 65 nm have been used for nMOS transistors and for pMOS transistor width of 2.5 times used. Simulation has been done at room temperature, 26°C.

A four input NAND has a delay of 1.0 ns at 90 nm and 0.7ns at 65nm technologies while 1.8 ns have taken to generate Sum at 90 nm and 1.5 ns at 65 nm technology as shown in figure 8.

Table 3: Analysis of Sum at 4 bit vs 4 input NAND (CMOS)

<table>
<thead>
<tr>
<th>CMOS Technology</th>
<th>Sum Delay for 4 bit CLA</th>
<th>4 input NAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>90 nm</td>
<td>1.8 ns</td>
<td>1.0ns</td>
</tr>
<tr>
<td>65nm</td>
<td>1.5 ns</td>
<td>0.7ns</td>
</tr>
</tbody>
</table>

In this hierarchical design a 4 input NAND generates G and P terms at 4-bit for carry out, another at 16 bit, another 64 bit and only 2 input NAND at 128 bit as shown from figure 6. So top part addition from A127 to A63 dictated by the carry generation of three 4-input NANDs and a 2 input NAND as carrying drives the final summation stage.

Figure 8: 4-bit CLA with NAND gates used in Hierarchical Design for 128 bits

A 128-bit Hierarchical CLA designed used to simulate with 90nm and 65 nm CMOS technology has been shown in figure 10 using Microwind EDA tools.

Figure 9: Propagation delay of simulation for 4-bit CLA with NAND gates used in Hierarchical CLA Design for 128 bits

Table 3: Analysis of Sum at 4 bit vs 4 input NAND (CMOS)

Figure 10: 128-bit Hierarchical CLA

Only inputs A3 to A0 and B3 to B0 have been changed in the simulation to find propagation delay because the “Carry out calculation has to begin from the Least Significant Bits (LSB) for worst case delay. Figure 9 shows the beginning of x-axis scale of 3.4 ns with each sub-division with 0.1 ns for final carry called “Carry 128” for full cycle of inputs. Inputs A0 to A127 have been kept at “High” and B0 has been toggled from “Low” to “High” while all other inputs of B have been kept at “Low” to have a carry propagated through all 128 bits. The final carry has a delay of 4.6 ns at 90 nm and 4.0 ns at 65nm technologies in Microwind simulation at using 90 nm and 65 nm libraries as shown in figure 11.
A Hierarchical Design of 128 Bit Carry Lookahead Adder in 65 nm CMOS Technology

Figure 11: Propagation delay of simulation for Hierarchical CLA Design for 128 bits

Previous work of Abdulkajadeed (2015) shows a delay 5.62 ns for a 64 bit hierarchical CLA with FPGA Virtex7 family. Even at 128 bits, this result is faster by 20 percent. Mangaraar (2014) did a 128-bit Carry Select Adder with a delay of 36.52 ns at 90nm CMOS technology [10]. Balasubramanian (2016) did only 32 bit adder design with delay of 3.53 ns [2]. Bharathi A, et (2014) did a 128 bit CLA with a Domino Logic using HSPICE but through a graph could not read the propagation value for sum [3].

Table 4: Analysis of Propagation Delay of Sum at 128-bit Hierarchical CLA at 26°C

<table>
<thead>
<tr>
<th>CMOS Technology</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>90 nm</td>
<td>4.6 ns</td>
</tr>
<tr>
<td>65nm</td>
<td>4.0 ns</td>
</tr>
<tr>
<td>Previous work of Abdulkajadeed (2015) at 130 nm</td>
<td>5.62 ns</td>
</tr>
</tbody>
</table>

VI. CONCLUSION

The analysis from the simulation of Microwind for a Hierarchical CLA design at 128 bits shows carry out of 4.6 ns at 90 nm and 4.0 ns at 65nm CMOS technology. The difference is 13 percent. Throughout our literature review there were only less than half dozen papers identified having research done on 128 bit addition with various architectures of adder. Only [1] Abdulkajadeed (2015) did similar work with 64 bits having a delay of 5.62 ns. 4 ns is critical point of technology as microprocessors already have gone ahead with over 10 GHz which translates a cycle time of 100 ps. At this speed even a 128 bit adder will take 40 to 46 cycles to complete an addition. This need be improved as 50 to 100 GHz microprocessors have been experimented in the latest designs of smart phones as of 2019. This is need for the shrinking technology node of Length of a transistor to 45 nm and 22 nm.

REFERENCES


AUTHORS PROFILE

Kishore Prabhala profile Kishore Prabhala is a research Scholar in EEE PhD, Rayalaseema University, Kurnool, AP and also Senior Member IEEE. He published six papers in CMOS VLSI design in India and guided over 8 projects for M.Tech. students in VLSI. He left USA in 1994 after working at Motorola, MMI and National Semiconductor from 1981. Currently, director of VLSI Design Centre, PSK Research Foundation, Opp ANU Mens Hostle, Nagarjuna Nagar – 522 510, and also Principal, PLNM Degree College, Opposite Acharya Nagarjuna University Mens Hostel, Nagarjuna Nagar – 522 510, Guntur Dist., AP, India. He received a MSIE from Georgia Institute of Technology, GA, USA in 1989 and BSEE from Purdue University, W.Lafayette, IN, USA in 1981, Cell: 9177408565

Prof. Prabandhakam Sangameswarra Raju is a Professor in Dept. of Electrical and Electronics Engineering, SVU Engineering College, Sri Venkateswara University, Tirupati – 517 502, Chitoor Dist., AP. He received M.Tech. and Ph.D. from SVU Engineering College. He has been teaching PG course for last 25 years and guided over 60 projects. He published over 80 papers. Currently there are 8 students pursuing Ph.D.