Digital System Design using HDL Lab Report

Experiment 7a

Synthesis of Adders

Submitted by

Madhu Krishnan A P

M.Tech VLSI and Embedded Systems

Department of Electronics

Cochin University of Science and Technology

Contents

1 Ripple carry adder

A Ripple Carry Adder (RCA) is a simple digital circuit used to perform binary addition. It consists of a series of full adders connected in a chain-like fashion, where the carry output from each full adder is passed as an input to the next. Each full adder in the chain computes the sum of two corresponding bits from the operands and the carry bit from the previous stage. The final sum is produced after all the adders in the chain compute their respective sums and propagate the carry through the entire sequence. Hence the name "ripple carry."

While the Ripple Carry Adder is easy to design and understand, it suffers from a major limitation: the carry propagation delay. Since each full adder must wait for the carry from the previous one, the overall delay increases linearly with the number of bits being added. For larger numbers of bits, this can lead to slower performance, especially in high-speed computing systems. As a result, alternative adder designs like the Carry Lookahead Adder (CLA) are often preferred when speed is critical, as they reduce the carry propagation delay.

1.1 4-Bit Ripple carry adder

In this experiment, the Ripple Carry Adder (RCA) was implemented in Verilog and synthesized using the Cadence Genus tool, as shown in Figure 1.

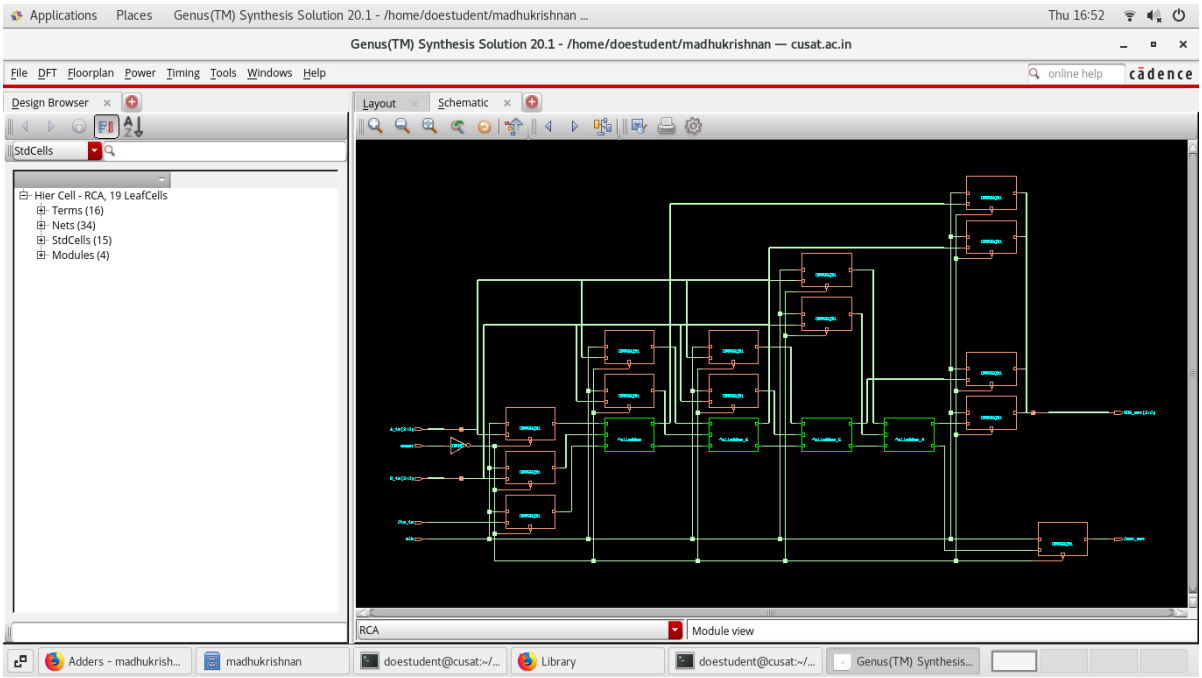

Figure 1: 4-bit Ripple carry adder: Schematic

Several key metrics were evaluated to assess the performance and efficiency of the circuit:

- Area: The area of the synthesized circuit was measured to determine how much physical space the RCA occupies on the chip. This provides insight into the effi-

ciency of the design in terms of resource usage. The area results from the synthesis process highlight the trade-off between the number of gates used and the circuit's functionality.

- Zero Slack Condition: The timing analysis ensured that the RCA met the required timing constraints, focusing on achieving zero slack. Zero slack indicates that the signal propagation time from input to output exactly matches the clock period, avoiding timing violations. This condition ensures the RCA operates within the desired performance parameters without timing delays that could lead to incorrect results.

- Power Consumption: The power consumption of the RCA was analyzed to assess its energy efficiency. Power analysis provided insights into dynamic power (due to switching activity) and static power (due to leakage currents). The power consumption results help evaluate how much energy the circuit uses during operation, which is a critical factor in the design of low-power digital systems.

1.1.1 Area analysis

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:00:10 pm
  Module:                 RCA
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
====================================================================

Instance    Module    Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------
RCA                         23    377.693     0.000      377.693 <none> (D)
  adder0 fulladder           1     19.679     0.000       19.679 <none> (D)
  adder1 fulladder_6         1     21.193     0.000       21.193 <none> (D)
  adder2 fulladder_5         3     24.221     0.000       24.221 <none> (D)
  adder3 fulladder_4         3     24.221     0.000       24.221 <none> (D)
  (D) = wireload is default in technology library
```

Figure 2: 4-bit Ripple carry adder: Area analysis

The synthesis report for the RCA (Ripple Carry Adder) module shows that the design uses 23 cells, resulting in a total area of 377.693 units. The module consists of different full adder instances (e.g., fulladder, fulladder_6, fulladder_5, fulladder_4), each with varying areas. The wireload model is set to the default "none," the synthesis was done under slow (balanced_tree) operating conditions, prioritizing balanced timing. The report highlights the overall area usage without additional wireload considerations for interconnects.

1.1.2 Power analysis

The power consumption analysis for the RCA (Ripple Carry Adder) module reveals that the majority of the power is consumed by the registers, which account for a significant 88.37 percent of the total power usage, amounting to 2.21429e-04 W. This is primarily due to the internal, leakage, and switching power within the register components, highlighting their importance in overall power consumption. Logic gates contribute 6.26 percent of the total power, or 1.56849e-05 W, indicating the combinational logic's role in the design. Clock components also consume a notable amount of power, representing 5.37 percent (1.34612e-05 W), which is expected due to signal states' constant toggling and driving. The remaining categories, such as memory, latches, bounding boxes, and pads, contribute negligible power, essentially having zero impact on the total consumption. The total power consumed by the entire RCA module is 2.50575e-04 W, with registers being the dominant consumer, followed by the logic and clock components. This distribution emphasizes the need for careful power management in the register and logic sections of the design, as they are the primary contributors to the overall power usage.

```
Instance: /RCA
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     1.40633e-06  2.15781e-04  4.24230e-06  2.21429e-04  88.37%
        latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        logic     5.54170e-07  1.20378e-05  3.09287e-06  1.56849e-05   6.26%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  1.34612e-05  1.34612e-05   5.37%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     1.96050e-06  2.27819e-04  2.07963e-05  2.50575e-04 100.00%
  Percentage           0.78%       90.92%        8.30%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 3: 4-bit Ripple carry adder: Power

1.1.3 Timing analysis

The timing analysis report for the 4-bit Ripple Carry Adder (RCA) synthesized using Cadence Genus provides key insights into the setup check and critical timing path. The analysis focuses on the path from the A_reg[0]/CK flip-flop to the SUM_out_reg[3]/D flip-flop, with the setup check having zero slack (0 ps), indicating that the design is operating right at the timing limit. The arrival time at SUM_out_reg[3]/D is 1246 ps, which matches the required time, confirming no timing violations and leaving no margin for error. The data path delay, which includes delays from various gates (e.g., ADDFX1, ADDFHX1, XOR2XL), is carefully tracked, and the report ensures that the

signal propagation meets the required timing conditions under the specified operating conditions (slow, balanced_tree). This analysis highlights that the design operates at the boundary of its timing constraints, and adjustments would be necessary if the timing were further optimized or improved.

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:00:10 pm
  Module:                 RCA
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
====================================================================


Path 1: MET (0 ps) Setup Check with Pin SUM_out_reg[3]/CK->D
          Group: clk
     Startpoint: (R) A_reg[0]/CK
          Clock: (R) clk
       Endpoint: (R) SUM_out_reg[3]/D
          Clock: (R) clk

                     Capture       Launch
        Clock Edge:+    1390            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1390            0

             Setup:-     134
       Uncertainty:-      10
     Required Time:=    1246
      Launch Clock:-       0
         Data Path:-    1246
             Slack:=       0

#-------------------------------------------------------------------------------------------------
#  Timing Point    Flags    Arc    Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                        (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------------
  A_reg[0]/CK       -       -      R     (arrival)     14    -   100     0       0    (-,-)
  A_reg[0]/Q        -       CK->Q  F     DFFRHQX1       1  6.7    69   379     379    (-,-)
  adder0/g47/CO     -       A->CO  F     ADDFX1         1  6.3   106   265     644    (-,-)
  adder1/g45/CO     -       CI->CO F     ADDFHX1        2  3.4    75   210     854    (-,-)
  adder2/g41/Y      -       A1->Y  F     AO22X1         2  3.5    65   186    1040    (-,-)
  adder3/g45/Y      -       A->Y   R     XOR2XL         1  2.3    66   206    1246    (-,-)
  SUM_out_reg[3]/D <<<      -      R     DFFRHQX1       1    -     -     0    1246    (-,-)
#-------------------------------------------------------------------------------------------------
```

Figure 4: 4-bit Ripple carry adder: Timing analysis

1.2 8-bit Ripple carry adder

An 8-bit Ripple Carry Adder adds two 8-bit binary numbers using the same approach as the 4-bit RCA. The carry bit is propagated sequentially from one bit to the next, causing the addition process to take longer as the number of bits increases. The speed of the 8-bit RCA is slower compared to more advanced adders like Carry Lookahead

Adders (CLA) because of the carry propagation delay. Despite this, it remains a simple and cost-effective solution for basic addition tasks in digital circuits.

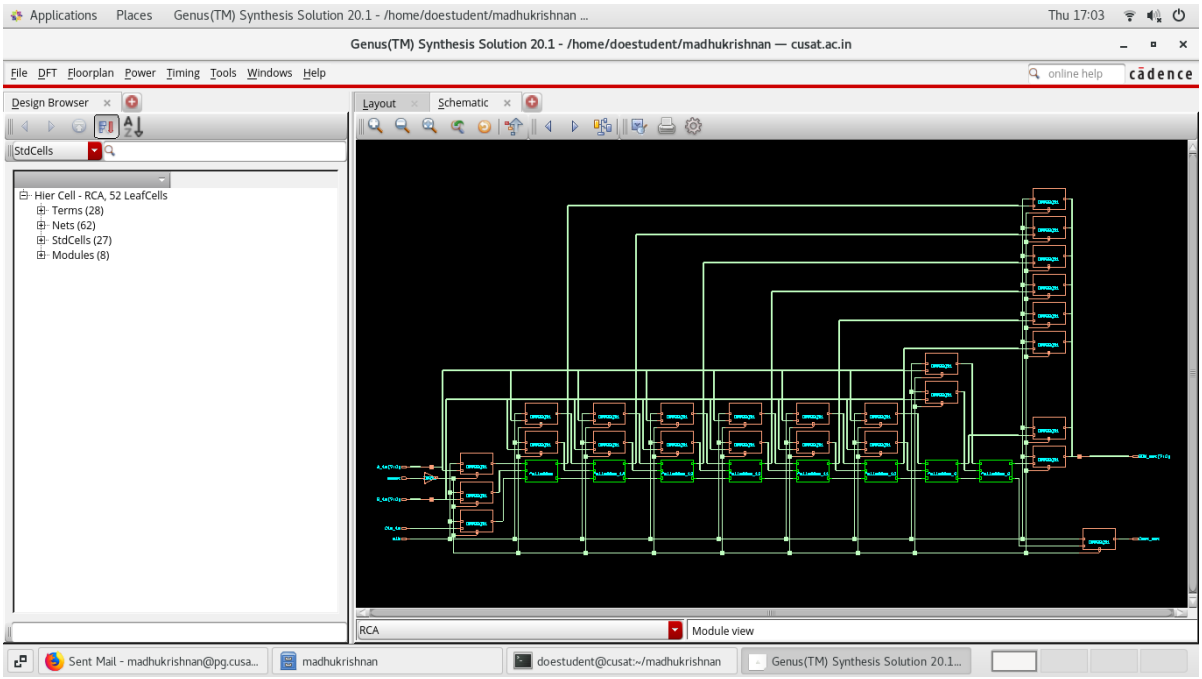

Figure 5: 8-bit Ripple carry adder: Schematic

1.2.1 Area analysis

The synthesis report for the RCA (Ripple Carry Adder) module reveals 67 cells used in the design, with a total area of 781.121 units. The RCA module consists of several full adder instances, each with varying areas. For example, adder0 uses six cells with a total area of 28.005 units, while adder1 has seven cells and occupies 41.630 units. Other adders, such as adder2, adder3, and adder4, also vary in size, with areas ranging from 25.735 to 27.248 units. The module utilizes a timing library for area mode and operates under slow (balanced_tree) conditions. The wireload model is set to the default "none," implying that no additional wireload model has been applied for interconnects. The RCA design utilizes multiple full adders with varying areas to achieve the required functionality.

1.2.2 Power analysis

The power consumption breakdown for the RCA (Ripple Carry Adder) module shows that the majority of the power (86.05 percent) is consumed by the registers, with a total of 3.70349e-04 W. This is followed by the logic gates, which account for 8.98 percent of the total power, consuming 3.86515e-05 W. Clock components contribute 4.97 percent of the total power, amounting to 2.13681e-05 W. The remaining categories consume negligible power, including memory, latches, bounding boxes, pads, and PM. The module's total power consumption is 4.30369e-04 W, with registers being the dominant power consumer.

```
=============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:06:05 pm
  Module:                 RCA
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
=============================================================

Instance    Module     Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------
RCA                         67    781.121     0.000      781.121 <none> (D)
  adder0 fulladder           6     28.005     0.000       28.005 <none> (D)
  adder1 fulladder_14        7     41.630     0.000       41.630 <none> (D)
  adder2 fulladder_13        4     25.735     0.000       25.735 <none> (D)
  adder3 fulladder_12        4     25.735     0.000       25.735 <none> (D)
  adder4 fulladder_11        5     27.248     0.000       27.248 <none> (D)
  adder5 fulladder_10        5     27.248     0.000       27.248 <none> (D)
  adder6 fulladder_9         4     25.735     0.000       25.735 <none> (D)
  adder7 fulladder_8         5     24.221     0.000       24.221 <none> (D)
  (D) = wireload is default in technology library
```

Figure 6: 8-bit Ripple carry adder: Area

```
Instance: /RCA
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     2.88923e-06  3.61303e-04  6.15699e-06  3.70349e-04  86.05%
        latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        logic     1.40658e-06  2.74225e-05  9.82241e-06  3.86515e-05   8.98%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  2.13681e-05  2.13681e-05   4.97%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     4.29582e-06  3.88725e-04  3.73475e-05  4.30369e-04 100.00%
  Percentage           1.00%       90.32%        8.68%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 7: 8-bit Ripple carry adder: Power analysis

1.2.3 Timing analysis

The timing analysis for the 8-bit Ripple Carry Adder (RCA) design, synthesized using Cadence Genus, shows that the setup checks for the critical path between the registers and the sum output was successful with zero slack, indicating that the circuit meets the timing requirements.

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:06:05 pm
  Module:                 RCA
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


Path 1: MET (0 ps) Setup Check with Pin SUM_out_reg[7]/CK->D
          Group: clk
     Startpoint: (R) A_reg[0]/CK
          Clock: (R) clk
       Endpoint: (R) SUM_out_reg[7]/D
          Clock: (R) clk

                     Capture       Launch
        Clock Edge:+    1630            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1630            0

             Setup:-     133
       Uncertainty:-      10
     Required Time:=    1487
      Launch Clock:-       0
         Data Path:-    1486
             Slack:=       0
```

```
#-------------------------------------------------------------------------------------------------
#  Timing Point    Flags    Arc    Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                        (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------------
  A_reg[0]/CK       -       -      R     (arrival)     26    -   100     0       0    (-,-)
  A_reg[0]/Q        -       CK->Q  R     DFFRHQX1       3  6.0    85   368     368    (-,-)
  adder0/g87/Y      -       A->Y   F     NAND2XL        1  2.8    93    90     458    (-,-)
  adder0/g83/Y      -       B0->Y  R     OAI21X1        2  3.3    96    65     523    (-,-)
  adder1/g44/Y      -       A1N->Y R     OAI2BB1X1      2  4.6    72   162     685    (-,-)
  adder2/g46/Y      -       B->Y   F     NAND2X1        1  2.7    64    66     751    (-,-)
  adder2/g49/Y      -       B0->Y  R     OAI2BB1X1      2  4.6    72    66     816    (-,-)
  adder3/g52/Y      -       B->Y   F     NAND2X1        1  2.7    64    66     882    (-,-)
  adder3/g64/Y      -       B0->Y  R     OAI2BB1X1      2  4.6    72    66     948    (-,-)
  adder4/g49/Y      -       B->Y   F     NAND2X1        1  2.8    65    67    1015    (-,-)
  adder4/g2/Y       -       B->Y   R     NAND2X1        2  4.6    65    63    1078    (-,-)
  adder5/g49/Y      -       B->Y   F     NAND2X1        1  2.8    65    65    1143    (-,-)
  adder5/g2/Y       -       B->Y   R     NAND2X1        2  4.6    65    63    1206    (-,-)
  adder6/g46/Y      -       B->Y   F     NAND2X1        1  2.7    64    64    1270    (-,-)
  adder6/g52/Y      -       B0->Y  R     OAI2BB1X1      2  4.4    71    64    1335    (-,-)
  adder7/g45/Y      -       A->Y   F     INVX1          2  3.3    42    44    1379    (-,-)
  adder7/g42/Y      -       B->Y   R     MXI2XL         1  2.2   115   107    1486    (-,-)
  SUM_out_reg[7]/D <<<      -      R     DFFRHQX8       1    -     -     0    1486    (-,-)
#-------------------------------------------------------------------------------------------------
```

Figure 8: 8-bit Ripple carry adder: Timing analysis

The setup time for the final sum output (SUM_out_reg[7]) was 133 ps, with an arrival time of 1630 ps and a required time of 1487 ps, resulting in no timing viola-

tions. The path analysis includes various delay contributions from individual gates like NAND2XL and OAI2BB1X1, with the final delay reaching 1486 ps for the sum output. This confirms that the circuit operates within the required timing constraints, ensuring reliable functionality.

1.3 16-bit Ripple carry adder

A 16-bit Ripple Carry Adder extends the concept of the 8-bit RCA to handle 16-bit binary numbers. Like its smaller counterparts, it adds corresponding bits of the two numbers starting from the least significant bit and propagates the carry bit through each bit in the sequence. As the bit-width increases, the carry propagation delay becomes more significant, leading to slower performance in high-speed applications. Despite the delay, a 16-bit RCA is still a basic, straightforward design suitable for low-complexity systems. A 16-bit ripple carry adder was synthesized as shown in Figure 9.

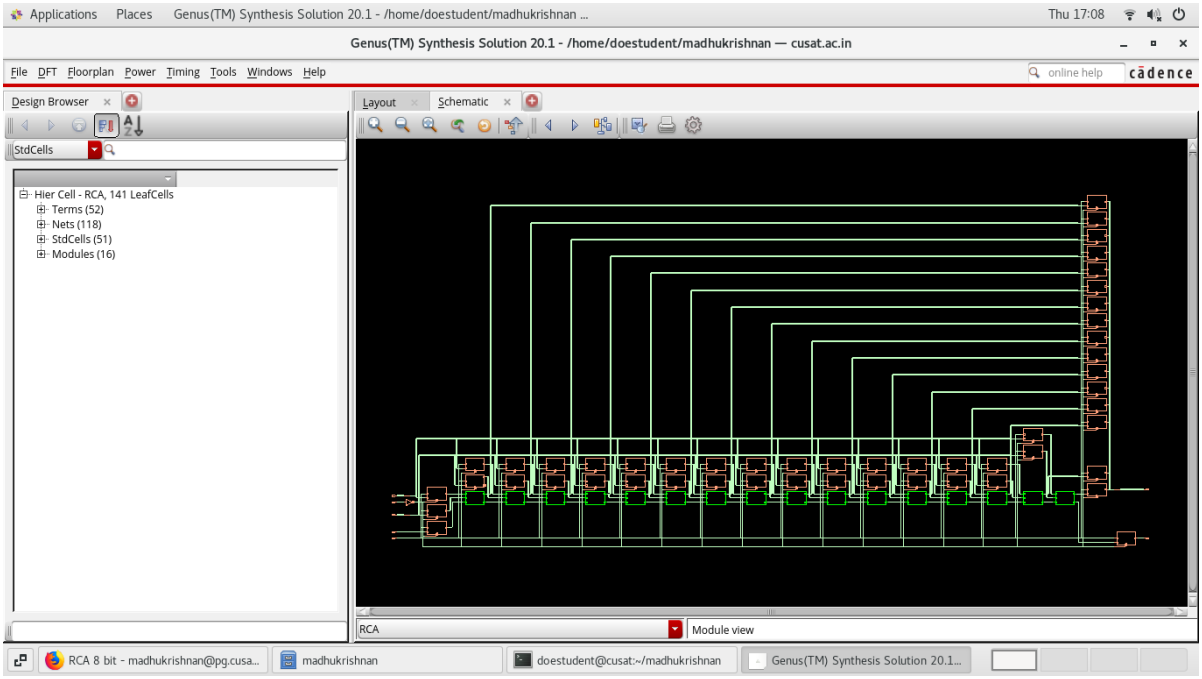

Figure 9: 16-bit Ripple carry adder: Schematic

1.3.1 Area analysis

The RCA module, generated using the Genus Synthesis Solution, contains 142 cells with a total area of 1597.816 units. Each of the 16 full adder instances, labelled adder0 to adder15, has varying cell counts and areas, ranging from 5 to 8 cells and from 24.978 to 60.552 units in the area. The total net area for each instance is 0, as the report indicates, and no wireload is specified, as the technology library's default wireload model is used. The synthesis process was carried out under "slow" operating conditions with a balanced tree configuration, and the wireload mode was "enclosed."

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:11:04 pm
  Module:                 RCA
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

 Instance    Module     Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------
RCA                          142   1597.816     0.000     1597.816 <none> (D)
  adder0  fulladder            6     37.088     0.000       37.088 <none> (D)
  adder1  fulladder_30         8     46.171     0.000       46.171 <none> (D)
  adder2  fulladder_29         5     31.790     0.000       31.790 <none> (D)
  adder3  fulladder_28         5     30.276     0.000       30.276 <none> (D)
  adder4  fulladder_27         8     60.552     0.000       60.552 <none> (D)
  adder5  fulladder_26         5     31.790     0.000       31.790 <none> (D)
  adder6  fulladder_25         5     31.790     0.000       31.790 <none> (D)
  adder7  fulladder_24         5     38.602     0.000       38.602 <none> (D)
  adder8  fulladder_23         6     43.143     0.000       43.143 <none> (D)
  adder9  fulladder_22         5     30.276     0.000       30.276 <none> (D)
  adder10 fulladder_21         6     29.519     0.000       29.519 <none> (D)
  adder11 fulladder_20         6     29.519     0.000       29.519 <none> (D)
  adder12 fulladder_19         6     29.519     0.000       29.519 <none> (D)
  adder13 fulladder_18         5     31.790     0.000       31.790 <none> (D)
  adder14 fulladder_17         5     27.248     0.000       27.248 <none> (D)
  adder15 fulladder_16         5     24.978     0.000       24.978 <none> (D)
  (D) = wireload is default in technology library
```

Figure 10: 16-bit Ripple carry adder: Area

1.3.2 Power analysis

The RCA instance's power analysis shows power consumption distribution across different categories. The total power is 5.34178e-04 W, with leakage power contributing 8.80905e-06 W (1.65 percent), internal power at 4.72157e-04 W (88.39 percent), and switching power at 5.32119e-05 W (9.96 percent). Most power comes from the "register" category, which accounts for 82.94 percent of the total power, followed by the "logic" category at 12.20 percent. The "clock" category contributes 4.87 percent, while other categories such as "memory," "latch," "bbox," "pad," and "pm" have no significant power consumption. The power breakdown is mainly internal, with a small portion of switching power.

1.3.3 Timing analysis

The timing analysis of the 16-bit Ripple Carry Adder (RCA) shows that the setup check for the critical path between the flip-flops and the output register is met with zero slack, indicating no timing violations. The path starts at the B_reg[0]/CK flip-flop, where the clock edge arrives at 2580 ps, and ends at the SUM_out_reg[15]/D flip-flop, where the data is captured. The data path delay is 2484 ps, with a required setup time of 86 ps and an uncertainty of 10 ps. The data propagation through various logic gates, including

```
Instance: /RCA
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total     Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     5.32414e-06  4.28236e-04  9.47512e-06  4.43036e-04  82.94%
       latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       logic     3.48491e-06  4.39208e-05  1.77414e-05  6.51472e-05  12.20%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     0.00000e+00  0.00000e+00  2.59953e-05  2.59953e-05   4.87%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     8.80905e-06  4.72157e-04  5.32119e-05  5.34178e-04 100.01%
  Percentage           1.65%       88.39%        9.96%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 11: 16-bit Ripple carry adder: Power

NAND, NOR, and XOR gates, is detailed in the report, showing the time taken for each gate's operation. The analysis confirms that the timing constraints are satisfied, with a slack value of 0 ps, indicating optimal performance for the 16-bit RCA design.

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:11:04 pm
  Module:                 RCA
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


Path 1: MET (0 ps) Setup Check with Pin SUM_out_reg[15]/CK->D
          Group: clk
     Startpoint: (R) B_reg[0]/CK
          Clock: (R) clk
       Endpoint: (F) SUM_out_reg[15]/D
          Clock: (R) clk

                     Capture       Launch
       Clock Edge:+    2580            0
       Src Latency:+      0            0
       Net Latency:+      0 (I)        0 (I)
           Arrival:=    2580            0

             Setup:-      86
       Uncertainty:-      10
     Required Time:=    2484
      Launch Clock:-       0
         Data Path:-    2484
             Slack:=       0
```

Timing Point	Flags	Arc	Edge	Cell	Fanout	Load (fF)	Trans (ps)	Delay (ps)	Arrival (ps)	Instance Location
B_reg[0]/CK	-	-	R	(arrival)	50	-	100	0	0	(-,-)
B_reg[0]/Q	-	CK->Q	F	DFFRX2	3	14.5	84	396	396	(-,-)
adder0/g86/Y	-	B->Y	R	NOR2X2	1	5.3	73	77	473	(-,-)
adder0/g83/Y	-	A1->Y	F	OAI21X2	2	4.4	65	68	541	(-,-)
adder1/g46/Y	-	B->Y	R	NAND2X1	1	5.3	75	67	608	(-,-)
adder1/g44/Y	-	B->Y	F	NAND2X2	2	7.1	75	73	681	(-,-)
adder2/g46/Y	-	B->Y	R	NAND2X2	1	5.3	55	53	734	(-,-)
adder2/g44/Y	-	B->Y	F	NAND2X2	2	4.6	61	57	791	(-,-)
adder3/g46/Y	-	B->Y	R	NAND2X1	1	2.8	52	51	842	(-,-)
adder3/g49/Y	-	B->Y	F	NAND2X1	2	6.9	113	98	940	(-,-)
adder4/g57/Y	-	B->Y	R	NAND2X2	1	5.3	54	61	1002	(-,-)
adder4/g2/Y	-	B->Y	F	NAND2X2	2	7.1	70	68	1069	(-,-)
adder5/g46/Y	-	B->Y	R	NAND2X2	1	5.3	55	52	1121	(-,-)
adder5/g44/Y	-	B->Y	F	NAND2X2	2	7.1	75	68	1189	(-,-)
adder6/g46/Y	-	B->Y	R	NAND2X2	1	5.3	55	53	1242	(-,-)
adder6/g44/Y	-	B->Y	F	NAND2X2	2	7.1	75	68	1310	(-,-)
adder7/g49/Y	-	B->Y	R	NAND2X2	1	5.3	55	53	1363	(-,-)
adder7/g2/Y	-	B->Y	F	NAND2X2	2	7.0	70	68	1431	(-,-)
adder8/g49/Y	-	B->Y	R	NAND2X2	1	5.3	52	52	1482	(-,-)
adder8/g2/Y	-	B->Y	F	NAND2X2	2	4.6	54	56	1538	(-,-)
adder9/g49/Y	-	B->Y	R	NAND2X1	1	2.8	52	49	1587	(-,-)
adder9/g52/Y	-	B->Y	F	NAND2X1	2	4.4	80	76	1663	(-,-)
adder10/g49/Y	-	B->Y	R	NAND2X1	1	2.8	52	57	1720	(-,-)
adder10/g2/Y	-	B->Y	F	NAND2X1	2	4.4	81	76	1796	(-,-)
adder11/g49/Y	-	B->Y	R	NAND2X1	1	2.8	52	57	1852	(-,-)
adder11/g2/Y	-	B->Y	F	NAND2X1	2	4.4	81	76	1928	(-,-)
adder12/g49/Y	-	B->Y	R	NAND2X1	1	2.8	52	57	1985	(-,-)
adder12/g2/Y	-	B->Y	F	NAND2X1	2	7.1	116	100	2085	(-,-)
adder13/g49/Y	-	B->Y	R	NAND2X2	1	5.3	55	62	2147	(-,-)
adder13/g2/Y	-	B->Y	F	NAND2X2	2	4.6	61	57	2204	(-,-)
adder14/g46/Y	-	B->Y	R	NAND2X1	1	2.8	52	51	2254	(-,-)
adder14/g2/Y	-	B->Y	F	NAND2X1	2	5.3	93	84	2338	(-,-)
adder15/g45/Y	-	A->Y	R	INVX1	2	5.4	64	73	2412	(-,-)
adder15/g42/Y	-	B->Y	F	MXI2X1	1	2.2	76	72	2484	(-,-)
SUM_out_reg[15]/D	<<<	-	F	DFFRHQX8	1	-	-	0	2484	(-,-)

Figure 12: 16-bit Ripple carry adder: Timing analysis

1.4 32-bit Ripple carry adder

A 32-bit Ripple Carry Adder similarly adds two 32-bit binary numbers to the 16-bit RCA by propagating the carry bit through each of the 32 bits. This results in an even longer delay due to the increased number of bits, making the adder slower than other more advanced addition circuits like Carry Lookahead Adders (CLA) or Carry Select Adders (CSA). While the 32-bit RCA is simple and easy to implement, it is not ideal for applications requiring high-speed arithmetic operations, as the carry propagation time can significantly impact performance.

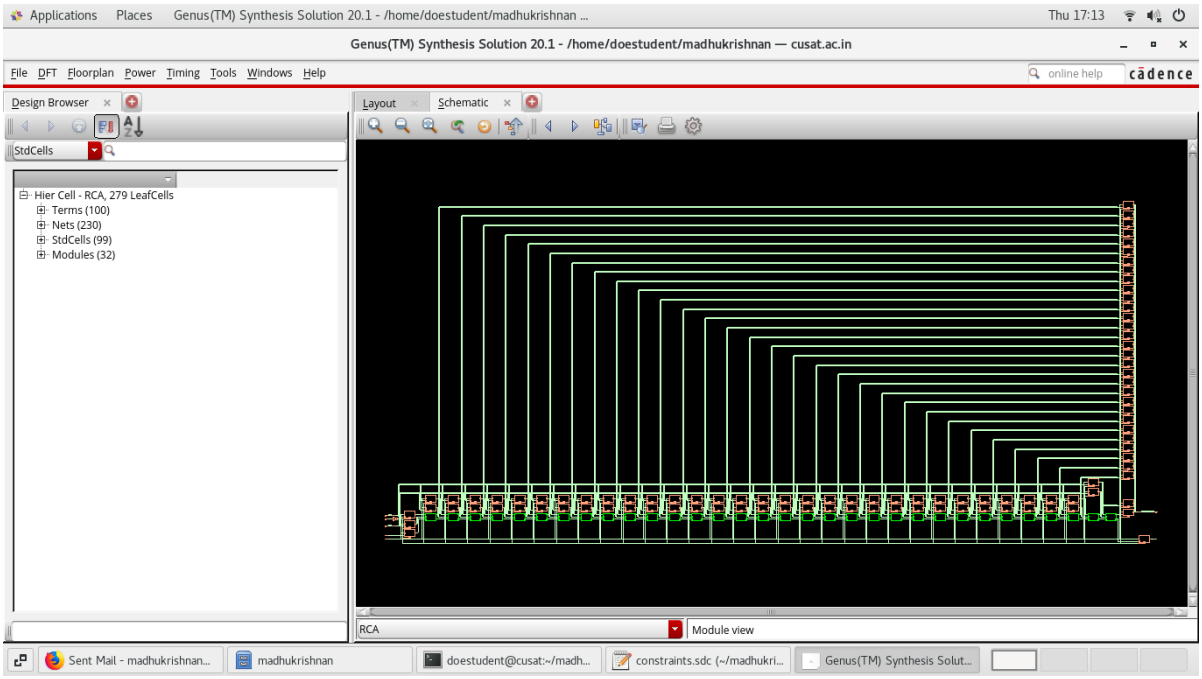

Figure 13: 32-bit Ripple carry adder: Schematic

1.4.1 Area analysis

The generated report provides a detailed analysis of an RCA (Ripple Carry Adder) module synthesized using the Genus(TM) Synthesis Solution. The report outlines various parameters, including the instance, module, and cell count for each full adder instance within the design.

The RCA module contains 229 cells, with a total area of 2848.215. The cells are distributed across various full adder instances, each with different areas, ranging from 24.221 to 40.116 units. All the cells have a wireload mode set to default in the technology library, which is noted in the report as "<none> (D)".

The operating conditions are labelled "slow" using a balanced tree approach for timing optimization. The wireload mode is set to "enclosed," suggesting that the layout design ensures that the wire capacitances do not exceed the cell area limits. The area mode indicates that timing constraints were applied using the technology's timing library.

This synthesis report overviews the module's structure and optimization parameters for physical and timing characteristics.

1.4.2 Power analysis

The power analysis for the RCA instance reveals that the majority of power consumption (84.78 percent) is due to the register category, with a total power consumption of 4.06106e-04 W. The logic category contributes 10.22 percent (4.89384e-05 W), primarily from internal and switching power. The clock category accounts for 5.00 percent

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:18:04 pm
  Module:                 RCA
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


 Instance    Module     Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------
RCA                          229   2848.215     0.000     2848.215 <none> (D)
  adder0  fulladder            6     27.248     0.000       27.248 <none> (D)
  adder1  fulladder_62         5     40.116     0.000       40.116 <none> (D)
  adder2  fulladder_61         5     29.519     0.000       29.519 <none> (D)
  adder3  fulladder_60         4     28.005     0.000       28.005 <none> (D)


  adder28 fulladder_35         4     24.978     0.000       24.978 <none> (D)
  adder29 fulladder_34         3     24.221     0.000       24.221 <none> (D)
  adder30 fulladder_33         4     24.978     0.000       24.978 <none> (D)
  adder31 fulladder_32         5     24.221     0.000       24.221 <none> (D)
  (D) = wireload is default in technology library
```

Figure 14: 32-bit Ripple carry adder: Area

```
Instance: /RCA
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total     Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     9.84430e-06  3.89797e-04  6.46519e-06  4.06106e-04  84.78%
       latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       logic     5.71867e-06  3.15494e-05  1.16703e-05  4.89384e-05  10.22%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     0.00000e+00  0.00000e+00  2.39446e-05  2.39446e-05   5.00%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     1.55630e-05  4.21346e-04  4.20801e-05  4.78989e-04 100.00%
  Percentage           3.25%       87.97%        8.79%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 15: 32-bit Ripple carry adder: Power

(2.39446e-05 W), while the memory, latch, bbox, pad, and pm categories contribute no significant power. The overall power distribution shows that internal power is the dominant factor, contributing 87.97 percent of the total power consumption.

1.4.3 Timing analysis

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:18:04 pm
  Module:                 RCA
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
====================================================================


Path 1: MET (0 ps) Setup Check with Pin SUM_out_reg[31]/CK->D
          Group: clk
     Startpoint: (R) B_reg[1]/CK
          Clock: (R) clk
       Endpoint: (R) SUM_out_reg[31]/D
          Clock: (R) clk

                     Capture       Launch
        Clock Edge:+    5470            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    5470            0

             Setup:-     134
       Uncertainty:-      10
     Required Time:=    5326
      Launch Clock:-       0
         Data Path:-    5326
             Slack:=       0


#-------------------------------------------------------------------------------------------------
#   Timing Point    Flags    Arc    Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                     (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------------
  B_reg[1]/CK       -        -      R     (arrival)     98    -   100     0       0    (-,-)
  B_reg[1]/Q        -        CK->Q  R     DFFRHQX1       2  3.5    66   348     348    (-,-)
  adder1/g2/Y       -        A->Y   F     CLKXOR2X1      2  4.5    74   230     578    (-,-)
  adder1/g46/Y      -        A->Y   R     NAND2X1        1  2.7    50    58     636    (-,-)
  adder1/g44/Y      -        B0->Y  F     OAI2BB1X1      2  4.6    90    81     717    (-,-)
  adder2/g2/Y       -        B->Y   R     NAND2X1        1  5.3    68    75     792    (-,-)
  adder2/g44/Y      -        B->Y   F     NAND2X2        2  6.5    72    68     861    (-,-)

  adder28/g43/Y     -        B->Y   R     NAND2XL        1  2.7    66    71    4820    (-,-)
  adder28/g41/Y     -        B0->Y  F     OAI2BB1X1      2  3.4    75    73    4893    (-,-)
  adder29/g2/Y      -        B1->Y  F     AO22X1         2  3.5    65   164    5058    (-,-)
  adder30/g43/Y     -        B->Y   R     NAND2XL        1  2.7    66    67    5124    (-,-)
  adder30/g41/Y     -        B0->Y  F     OAI2BB1X1      2  4.4    87    82    5207    (-,-)
  adder31/g42/Y     -        A->Y   R     MXI2XL         1  2.3   118   119    5326    (-,-)
  SUM_out_reg[31]/D <<<      -      R     DFFRHQX1       1    -     -     0    5326    (-,-)
#-------------------------------------------------------------------------------------------------
```

Figure 16: 32-bit Ripple carry adder: Timing analysis

The timing analysis for the RCA module indicates that the setup checks for Path 1 between the flip-flop stages, from the launch at B_reg[1]/CK to the capture at the SUM_out_reg[31]/D, shows no slack (0 ps). The required time for the data path is 5326 ps, with an arrival time of 5470 ps, resulting in a setup slack of 0 ps. The detailed timing breakdown includes various logic gates, such as NAND2X1, OAI2BB1X1, and

AO22X1, with the path traversing through several stages, including adders and registers, with increasing delays (ranging from 48 ps to 162 ps). The final capture of data at SUM_out_reg[31]/D completes the path, with no timing violations observed.

1.5 64-bit Ripple carry adder

A 64-bit Ripple Carry Adder adds two 64-bit binary numbers using the same principle as smaller bit-width RCAs. The carry bit propagates from one bit to the next, which results in a substantial delay for large bit-width numbers. As the bit-width increases, the delay associated with carry propagation becomes even more pronounced, making the 64-bit RCA impractical for high-speed operations in modern systems. While simple and easy to implement, the 64-bit RCA is rarely used in performance-critical applications due to its slow addition time, particularly in processors and large-scale digital systems.

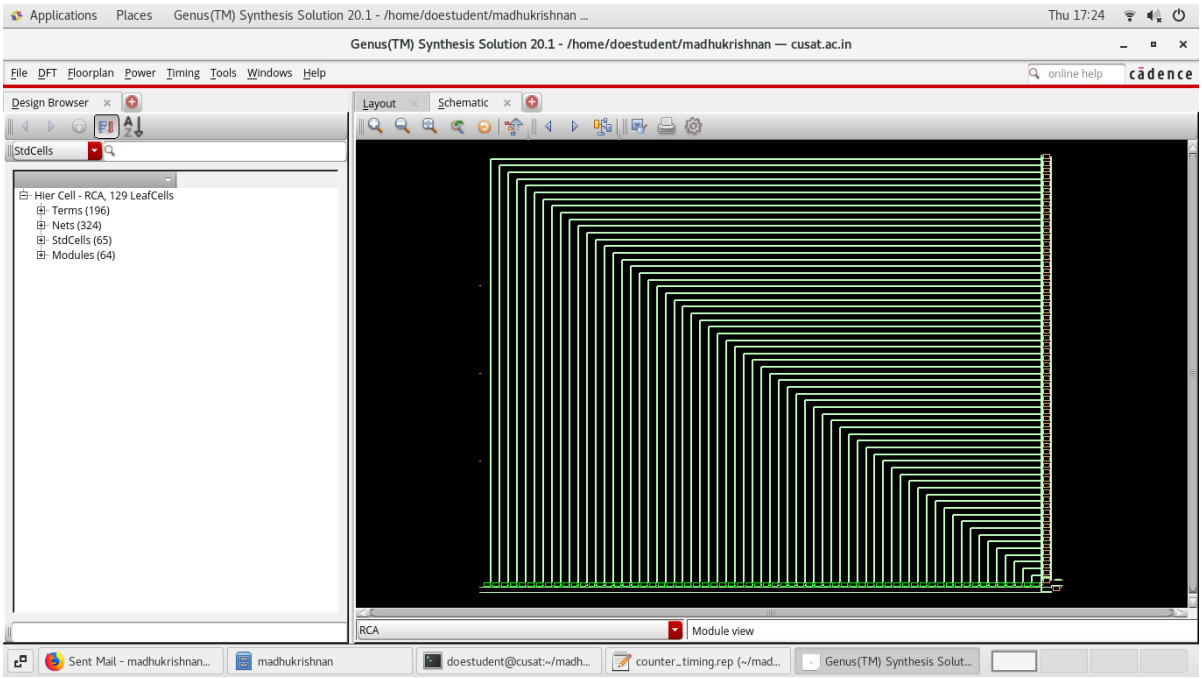

Figure 17: 64-bit Ripple carry adder: Schematic

1.5.1 Area analysis

The synthesis report for the 64-bit Ripple Carry Adder (RCA) provides a detailed breakdown of its design, showing a total cell area of 5765.307 units with 474 cells contributing to the structure. Each stage of the RCA consists of a full adder, with individual cell areas varying between 24.221 and 43.900 units. This variation reflects optimization efforts to balance timing delays and manage fanout across different adder stages. Larger cell areas in certain stages likely correspond to critical paths with more stringent timing constraints. The design is compact and adheres to the operating conditions specified for a balanced tree architecture, ensuring that the wireload effects are minimized. Despite the inherent delay associated with carry propagation in Ripple carry adders, this implementation demonstrates efficient area utilization and a structured approach to managing

timing and load across all 64 stages, making it suitable for applications where area and power are prioritized over speed.

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:36:37 pm
  Module:                 RCA
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

 Instance     Module     Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------------
RCA                          474   5765.307     0.000     5765.307 <none> (D)
  adder0  fulladder            6     27.248     0.000       27.248 <none> (D)
  adder1  fulladder_126        5     39.359     0.000       39.359 <none> (D)
  adder2  fulladder_125        5     42.386     0.000       42.386 <none> (D)
  adder60 fulladder_67         4     25.735     0.000       25.735 <none> (D)
  adder61 fulladder_66         4     24.978     0.000       24.978 <none> (D)
  adder62 fulladder_65         4     24.978     0.000       24.978 <none> (D)
  adder63 fulladder_64         3     24.221     0.000       24.221 <none> (D)
  (D) = wireload is default in technology library
```

Figure 18: 64-bit Ripple carry adder: Area

1.5.2 Power analysis

```
Instance: /RCA
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category          Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     1.94877e-05  4.13560e-04  6.86197e-06  4.39910e-04  83.46%
        latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        logic     1.18897e-05  3.60099e-05  1.38700e-05  6.17696e-05  11.72%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  2.54197e-05  2.54197e-05   4.82%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     3.13774e-05  4.49570e-04  4.61517e-05  5.27099e-04 100.00%
  Percentage           5.95%       85.29%        8.76%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 19: 64-bit Ripple carry adder: Power

The 64-bit Ripple Carry Adder (RCA) power analysis reveals a total power consumption of 5.27099e-04 W, with the majority contributed by registers, accounting for 83.46 percent of the total power. Registers exhibit a significant 85.29 percent of internal power usage, alongside minimal contributions from leakage (5.95 percent) and switching power (8.76 percent). Logic gates contribute a smaller share, making up 11.72 percent of the total power, primarily from internal and switching activities. The clock network adds a modest 4.82 percent to the overall power due to switching activity. Other components, such as memory, latches, bounding boxes, pads, and power management, show negligible or no power usage. This breakdown highlights the dominance of register power consumption in the design and underscores the importance of optimizing internal and switching power for energy efficiency.

1.5.3 Timing analysis

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:36:37 pm
  Module:                 RCA
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


Path 1: MET (0 ps) Setup Check with Pin SUM_out_reg[63]/CK->D
          Group: clk
     Startpoint: (R) B_reg[1]/CK
          Clock: (R) clk
       Endpoint: (R) SUM_out_reg[63]/D
          Clock: (R) clk

                    Capture       Launch
       Clock Edge:+   10200            0
       Src Latency:+      0            0
       Net Latency:+      0 (I)        0 (I)
           Arrival:=  10200            0

             Setup:-    134
       Uncertainty:-     10
     Required Time:=  10056
      Launch Clock:-      0
         Data Path:-  10056
             Slack:=      0

#-------------------------------------------------------------------------------------------------
#   Timing Point    Flags    Arc    Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                          (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------------
  B_reg[1]/CK       -        -      R     (arrival)    194    -   100     0       0    (-,-)
  B_reg[1]/Q        -       CK->Q   R     DFFRHQX1       2  3.5    66   348     348    (-,-)
  adder1/g2/Y       -       A->Y    F     CLKXOR2X1      2  3.5    66   222     570    (-,-)
  adder1/g44/Y      -       A0N->Y  F     OAI2BB1X1      2  4.4    87   148     718    (-,-)
  adder61/g43/Y     -       B->Y    R     NAND2XL        1  2.7    66    71    9627    (-,-)
  adder61/g41/Y     -       B0->Y   F     OAI2BB1X1      2  3.5    76    74    9701    (-,-)
  adder62/g43/Y     -       B->Y    R     NAND2XL        1  2.7    66    71    9772    (-,-)
  adder62/g41/Y     -       B0->Y   F     OAI2BB1X1      2  3.5    76    74    9846    (-,-)
  adder63/g2/Y      -       A->Y    R     XOR2XL         1  2.3    66   210   10056    (-,-)
  SUM_out_reg[63]/D <<<     -       R     DFFRHQX1       1    -     -     0   10056    (-,-)
#-------------------------------------------------------------------------------------------------
```

Figure 20: 64-bit Ripple carry adder: Timing analysis

The report generated by the Genus Synthesis Solution provides an analysis of the timing and area characteristics of the RCA module under the specified operating conditions. The setup check for the path between the start point, B_reg[1]/CK, and the endpoint, SUM_out_reg[63]/D, demonstrates a slack of 0 ps, indicating that the timing requirements are met. The analysis includes details on the clock edges, latencies, and arrival times for the capture and launch clocks. The data path delay is calculated as 10,056 ps, which aligns with the required time. The path's propagation is detailed through multiple instances of logic gates and flip-flops, including DFFRHQX1, CLKXOR2X1, NAND2X1, and OAI2BB1X1, among others. Each stage specifies the fanout, load, transition times, and delays, ensuring clarity in understanding the contributions to the overall delay. The timing points reflect a consistent and balanced propagation across the components, maintaining the integrity of the timing analysis. This comprehensive breakdown confirms the circuit meets its design constraints while providing insights into potential optimization opportunities.

2 Carry look-ahead adder

A Carry Lookahead Adder (CLA) is a high-speed adder designed to overcome the delay caused by the sequential carry propagation in traditional adders like the Ripple Carry Adder (RCA). Using the concepts of generate (G) and propagate (P) signals, the CLA precomputes carry signals for each bit position, enabling simultaneous calculation of all carries. This reduces the dependency on sequential carry propagation and significantly enhances performance, especially for large bit-width adders. The CLA achieves this efficiency by leveraging hierarchical carry generation logic, often implemented using AND, OR, and XOR gates. It is an ideal choice for high-performance arithmetic operations in digital systems.

2.1 4-bit Carry lookahead adder

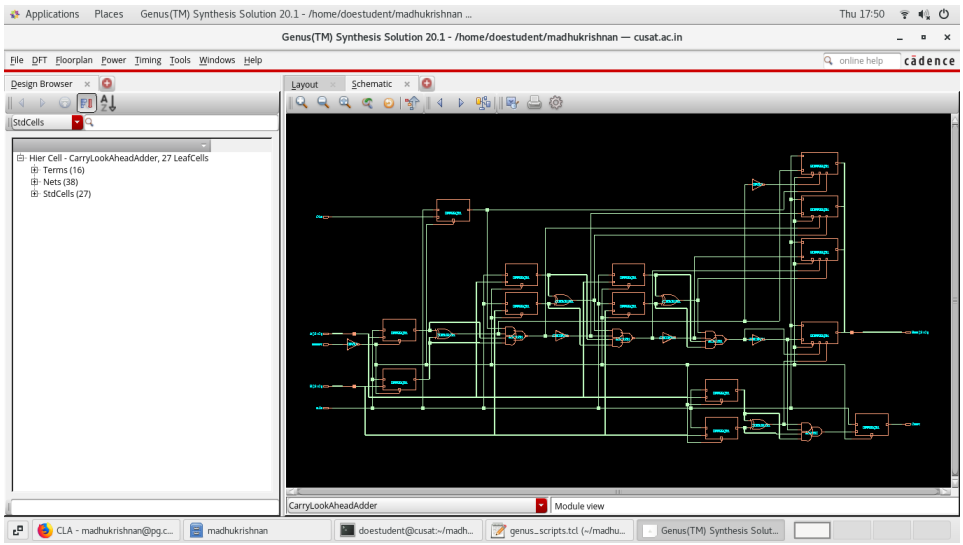

Figure 21: 4-bit Carry lookahead adder: Schematic

A 4-bit Carry Lookahead Adder (CLA) is a digital circuit that adds two 4-bit binary numbers. It improves upon the traditional ripple carry adder by reducing the propagation delay of carry bits. In a CLA, carry signals are generated quickly using generate and propagate functions, allowing the carry to be computed in parallel rather than sequentially. This results in faster addition compared to ripple carry adders, as it avoids the need for each bit to wait for the carry from the previous bit.

2.1.1 Area analysis

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:55:17 pm
  Module:                 CarryLookAheadAdder
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

      Instance      Module  Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------------
CarryLookAheadAdder                37    435.974     0.000      435.974 <none> (D)
  (D) = wireload is default in technology library
```

Figure 22: 4-bit Carry lookahead adder: Area

The synthesis report for the CarryLookAheadAdder module, generated by Genus Synthesis Solution on December 12, 2024, provides details about the module's design. It contains 37 cells, with a total cell area of 435.974 units. The net area is 0.000, indicating that no additional space was allocated for interconnects or routing. The total area of the module is 435.974 units, and no specific wireload model was applied, as the default wireload from the technology library is used. The module is synthesized under slow operating conditions with a balanced tree approach and uses the enclosed wireload mode for design estimation.

2.1.2 Power analysis

The power analysis for the CarryLookAheadAdder module provides a breakdown of power consumption across different categories. The total power consumption is 4.11005e-04 W, with leakage power contributing 2.48091e-06 W, internal power 3.70312e-04 W, and switching power 3.82123e-05 W. The registers consume most of the power, accounting for 85.57 percent of the total, followed by logic components at 9.83 percent. The clock power consumption is 4.60 percent, while memory, latches, pads, and other components contribute negligible power. The power consumption is predominantly internal, with a small proportion of switching power.

2.1.3 Timing analysis

The timing analysis for the CarryLookAheadAdder module shows a setup check for Path 1, with a clock period of 2 ps and a setup slack of 2 ps, indicating the design

```
Instance: /CarryLookAheadAdder
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     1.86163e-06  3.43119e-04  6.71257e-06  3.51694e-04  85.57%
        latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        logic     6.19279e-07  2.71926e-05  1.25730e-05  4.03849e-05   9.83%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  1.89267e-05  1.89267e-05   4.60%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     2.48091e-06  3.70312e-04  3.82123e-05  4.11005e-04 100.00%
  Percentage           0.60%       90.10%        9.30%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 23: 4-bit Carry lookahead adder: Power

meets the required timing constraints. The setup check involves transitioning from the Cin_reg_reg/CK to Sum_reg[3]/D, with the launch clock arriving at 0 ps and the required time being 880 ps. The path consists of various logic gates, including NAND2X1, OAI21X2, NOR2X2, CLKINVX2, and MXI2X1, with their respective delays and load values detailed in the table. The timing analysis confirms that the data path has a sufficient slack of 2 ps, ensuring the design functions correctly under the given conditions.

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:55:17 pm
  Module:                 CarryLookAheadAdder
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


Path 1: MET (2 ps) Setup Check with Pin Sum_reg[3]/CK->D
          Group: clk
     Startpoint: (R) Cin_reg_reg/CK
          Clock: (R) clk
       Endpoint: (R) Sum_reg[3]/D
          Clock: (R) clk

                    Capture       Launch
        Clock Edge:+    1010            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1010            0
```

```
             Setup:-     120
       Uncertainty:-      10
     Required Time:=     880
      Launch Clock:-       0
         Data Path:-     878
             Slack:=       2

#-------------------------------------------------------------------------------------------
# Timing Point   Flags   Arc   Edge   Cell      Fanout Load Trans Delay Arrival Instance
#                                                      (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------
  Cin_reg_reg/CK -       -     R     (arrival)     14    -   100     0       0    (-,-)
  Cin_reg_reg/Q  -       CK->Q R     DFFRHQX4       3 10.4    64   378     378    (-,-)
  g2424/Y        -       B->Y  F     NAND2X1        2  6.9   114   101     479    (-,-)
  g2412/Y        -       A1->Y R     OAI21X2        1  5.6    96   102     582    (-,-)
  g2410/Y        -       A->Y  F     NOR2X2         3  9.7    63    66     648    (-,-)
  g2407/Y        -       A1->Y R     OAI21X2        2  7.8   106   100     748    (-,-)
  g2406/Y        -       A->Y  F     CLKINVX2       2  5.4    46    43     791    (-,-)
  g2404/Y        -       B->Y  R     MXI2X1         1  2.2    82    86     878    (-,-)
  Sum_reg[3]/D   <<<     -     R     DFFRHQX8       1    -     -     0     878    (-,-)
#-------------------------------------------------------------------------------------------
```

Figure 24: 4-bit Carry lookahead adder: Timing

2.2 8-bit Carry lookahead adder

An 8-bit CLA extends the principles of a 4-bit CLA to handle 8-bit binary numbers. It still uses the generate and propagate functions to compute the carry bits more efficiently, reducing the delay associated with carry propagation. By adding more logic levels to handle the increased number of bits, an 8-bit CLA can perform addition in a shorter time than a ripple carry adder, making it suitable for applications where speed is crucial.

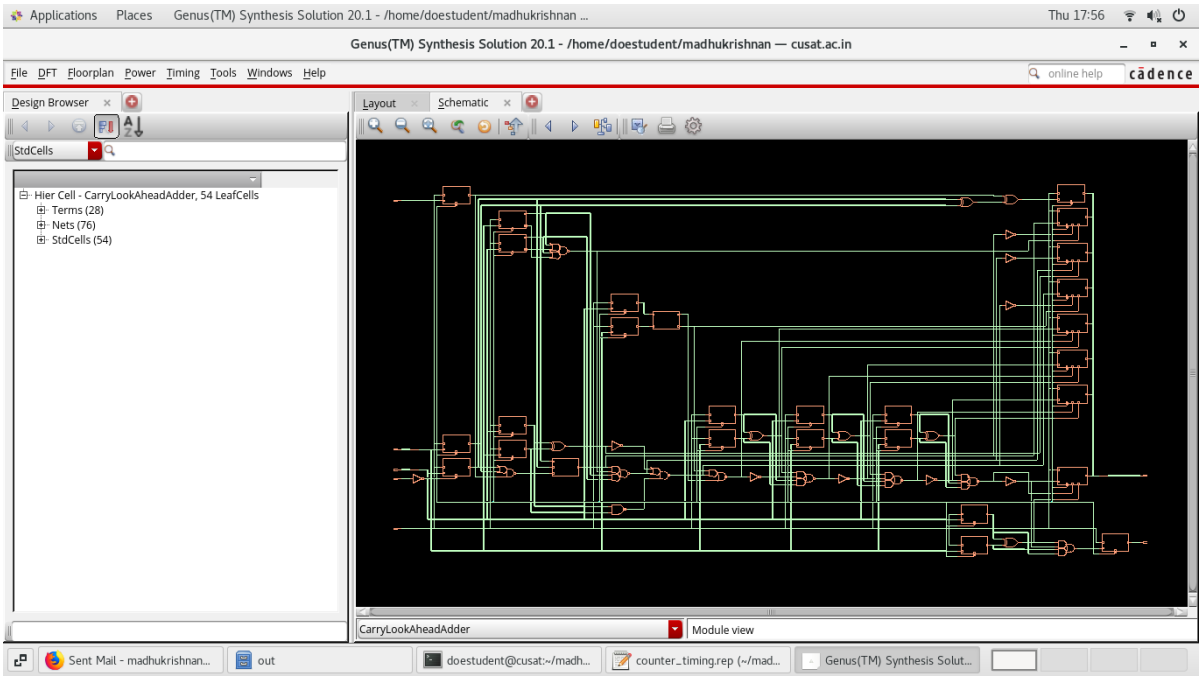

Figure 25: 8-bit Carry lookahead adder

2.2.1 Area analysis

The output represents the synthesis results of a digital design module called "CarryLookAheadAdder," generated by the Genus Synthesis Solution tool. The synthesis was performed under slow operating conditions with a balanced tree design strategy. The design contains 124 cells, which have a total area of 1016.517 units, with no additional wireload specified in the technology library. The net area is 0, indicating that there is no specific allocation for routing or wire area, and the total area remains the same as the cell area. The wireload is set to the default value for the given technology library, and the overall synthesis environment seems to prioritize timing optimization.

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:59:55 pm
  Module:                 CarryLookAheadAdder
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

      Instance      Module  Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------------
CarryLookAheadAdder               124   1016.517     0.000     1016.517 <none> (D)
  (D) = wireload is default in technology library
```

Figure 26: 8-bit Carry lookahead adder: Area

2.2.2 Power analysis

```
Instance: /CarryLookAheadAdder
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     3.53570e-06  4.88611e-04  1.38029e-05  5.05949e-04  77.85%
        latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        logic     1.74340e-06  7.42453e-05  3.91309e-05  1.15120e-04  17.71%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  2.88439e-05  2.88439e-05   4.44%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     5.27910e-06  5.62856e-04  8.17777e-05  6.49913e-04 100.00%
  Percentage           0.81%       86.60%       12.58%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 27: 8-bit Carry lookahead adder: Power

The power consumption breakdown for the "CarryLookAheadAdder" instance shows the contribution of various components to the total power usage in watts. The largest share of power is consumed by the registers, accounting for 77.85 percent of the total power. Within this, internal power (4.88611e-04 W) dominates, while switching power (1.38029e-05 W) is relatively small. The logic components contribute 17.71 percent to the power usage, with internal power (7.42453e-05 W) being the primary contributor. Clock power accounts for 4.44 percent of the total power (2.88439e-05 W), while other categories, such as memory, latch, bbox, pad, and pm, consume negligible power. The total power consumption sums up to 6.49913e-04 W, with internal power being the dominant factor (86.60 percent), followed by switching power (12.58 percent).

2.2.3 Timing analysis

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  05:59:55 pm
  Module:                 CarryLookAheadAdder
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


Path 1: MET (0 ps) Setup Check with Pin Sum_reg[6]/CK->D
          Group: clk
     Startpoint: (R) A_reg_reg[6]/CK
          Clock: (R) clk
       Endpoint: (R) Sum_reg[6]/D
          Clock: (R) clk

                    Capture       Launch
        Clock Edge:+    1230            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1230            0

             Setup:-     122
       Uncertainty:-      10
     Required Time:=    1098
      Launch Clock:-       0
         Data Path:-    1098
             Slack:=       0

#-------------------------------------------------------------------------------------------
#  Timing Point   Flags   Arc   Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                  (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------
  A_reg_reg[6]/CK -       -     R     (arrival)     26    -   100     0       0    (-,-)
  A_reg_reg[6]/Q  -       CK->Q F     DFFRHQX1       2  5.3    61   371     371    (-,-)
  g17009/Y        -       A->Y  R     NOR2X1         2  5.4   112   108     479    (-,-)
  g17000/Y        -       A->Y  F     CLKINVX1       2  5.4    64    64     543    (-,-)
  g16994/Y        -       B->Y  R     NAND2X1        2  6.9   100    76     619    (-,-)
  g16989/Y        -       A->Y  F     CLKINVX2       5 11.5    61    62     682    (-,-)
  g16987/Y        -       B->Y  R     NOR2XL         3  8.0   261   204     886    (-,-)
  g16934/Y        -       A0->Y F     AOI221X1       1  1.5   131   120    1006    (-,-)
  g16926/Y        -       A->Y  R     NAND3XL        1  2.2    80    91    1098    (-,-)
  Sum_reg[6]/D    <<<     -     R     DFFRHQX8       1    -     -     0    1098    (-,-)
#-------------------------------------------------------------------------------------------
```

Figure 28: 8-bit Carry lookahead adder: Timing analysis

The timing analysis for the "CarryLookAheadAdder" module shows a successful setup check for Path 1, with the data being correctly captured by the register Sum_reg[6]/D at the clock edge of clk. The launch edge occurs at 0 ps, while the capture edge happens at 1230 ps, with the required setup time being 1098 ps. The slack is 0 ps, indicating that the design meets the timing constraints without any violations. The path includes delays from various logic gates and registers, such as NOR2X1, NAND2X1, and AOI221X1, all of which contribute to the overall timing without causing any issues.

2.3 16-bit Carry lookahead adder

A 16-bit CLA performs the addition of two 16-bit binary numbers using the same carry lookahead concept but scaled for a larger bit width. It further optimizes carry generation by using multiple levels of lookahead logic to ensure that carry signals are calculated quickly. This results in faster computation times, especially for more complex arithmetic operations in systems requiring 16-bit precision. The design becomes more complex as the bit width increases, but the speed benefits make it ideal for high-performance digital systems.

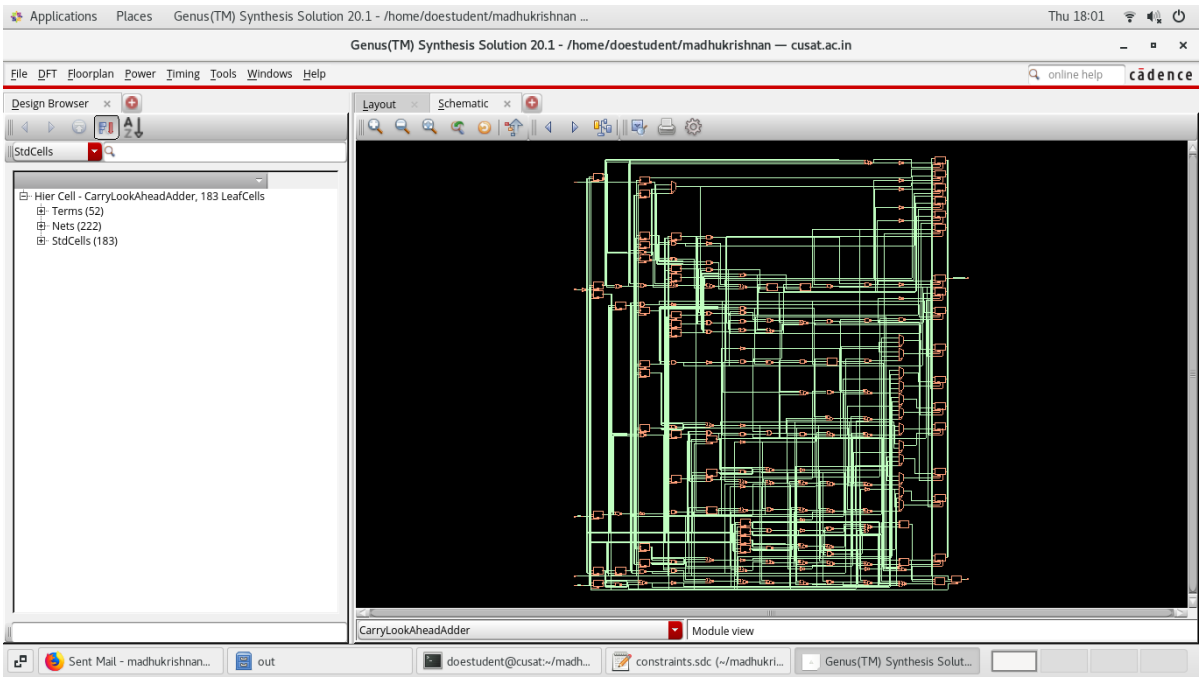

Figure 29: 16-bit Carry lookahead adder: Schematic

2.3.1 Area analysis

The synthesis results for the "CarryLookAheadAdder" module, generated by Genus Synthesis Solution, indicate that the design consists of 268 cells, with a total area of 2069.365 units. The wireload is set to the default value in the technology library, and no specific wireload has been applied. The net area is 0, suggesting no additional allocation for routing or wire area, meaning the cell area solely determines the total area. The synthe-

sis was performed under slow operating conditions using a balanced tree design strategy focusing on timing optimization.

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  06:04:18 pm
  Module:                 CarryLookAheadAdder
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

      Instance      Module  Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------------
CarryLookAheadAdder               268   2069.365     0.000     2069.365 <none> (D)
  (D) = wireload is default in technology library
```

Figure 30: 16-bit Carry lookahead adder: area

2.3.2 Power analysis

The power consumption breakdown for the "CarryLookAheadAdder" module shows a total power usage of 1.08992e-03 W, with the largest contribution coming from the registers, which account for 73.04 percent of the total power (7.96037e-04 W). The logic components contribute 22.76 percent (2.48022e-04 W), and the clock components account for 4.21 percent (4.58635e-05 W). Other categories, such as memory, latch, bbox, pad, and pm, consume negligible power. The internal power dominates the total power consumption (84.95 percent), with switching power making up 14.12 percent.

```
Instance: /CarryLookAheadAdder
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
      memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     6.03202e-06  7.68239e-04  2.17660e-05  7.96037e-04  73.04%
       latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       logic     4.14695e-06  1.57647e-04  8.62281e-05  2.48022e-04  22.76%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     0.00000e+00  0.00000e+00  4.58635e-05  4.58635e-05   4.21%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     1.01790e-05  9.25886e-04  1.53858e-04  1.08992e-03 100.01%
  Percentage           0.93%       84.95%       14.12%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 31: 16-bit Carry lookahead adder: Power

2.3.3 Timing analysis

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 12 2024  06:04:17 pm
  Module:                 CarryLookAheadAdder
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
====================================================================


Path 1: MET (0 ps) Setup Check with Pin Sum_reg[9]/CK->D
          Group: clk
     Startpoint: (R) B_reg_reg[7]/CK
          Clock: (R) clk
       Endpoint: (R) Sum_reg[9]/D
          Clock: (R) clk

                     Capture       Launch
        Clock Edge:+    1480            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1480            0

             Setup:-     234
       Uncertainty:-      10
     Required Time:=    1236
      Launch Clock:-       0
         Data Path:-    1236
             Slack:=       0

#-------------------------------------------------------------------------------------------------
#  Timing Point   Flags   Arc   Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                      (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------------
  B_reg_reg[7]/CK -       -     R     (arrival)     50    -   100     0       0    (-,-)
  B_reg_reg[7]/Q  -       CK->Q F     DFFRHQX2       2  8.3    59   379     379    (-,-)
  g10336/Y        -       A->Y  R     NAND2X2        8 21.0   103    97     477    (-,-)
  g10303/Y        -       AN->Y R     NOR2BX2        6 12.9   131   208     684    (-,-)
  g10295/Y        -       A->Y  F     CLKINVX1       3  9.0    89    89     773    (-,-)
  g10207/Y        -       A0->Y R     OAI21X2        2  6.5    95   107     880    (-,-)
  g10189/Y        -       B->Y  F     NAND2X1        3  6.3   113   104     983    (-,-)
  g10158/Y        -       B->Y  R     NAND2XL        1  3.7    90    92    1075    (-,-)
  g10148/Y        -       S0->Y R     MXI2XL         1  2.8   133   160    1236    (-,-)
  Sum_reg[9]/D    <<<     -     R     SDFFRHQX1      1    -     -     0    1236    (-,-)
#-------------------------------------------------------------------------------------------------
```

Figure 32: 16-bit Carry lookahead adder: Timing

The timing analysis for the "CarryLookAheadAdder" module confirms a successful setup check for Path 1, ensuring data transfer from B_reg_reg[7]/CK to Sum_reg[9]/D. The clock signal, clk, has a capture edge at 1480 ps and a launch edge at 0 ps. The required setup time is 1236 ps, with data arriving exactly at this point, resulting in a slack of 0 ps, indicating no timing violations. The path involves delays from registers and logic gates, including NAND2X2, NOR2BX2, and MXI2XL, with their respective fanout, load, and transition characteristics contributing to the total delay. Overall, the design satisfies the timing constraints for this path.

2.4 32-bit Carry lookahead adder

A 32-bit CLA operates on two 32-bit binary numbers and leverages the same carry lookahead mechanism as smaller bit-width CLAs, but with more advanced hierarchical structures. It involves more levels of logic to propagate the carry quickly across all 32 bits, which significantly reduces the addition time compared to traditional methods like ripple carry adders. The increased bit width means more complex circuitry, but it is crucial in systems like microprocessors and digital signal processors where large binary numbers are frequently processed.

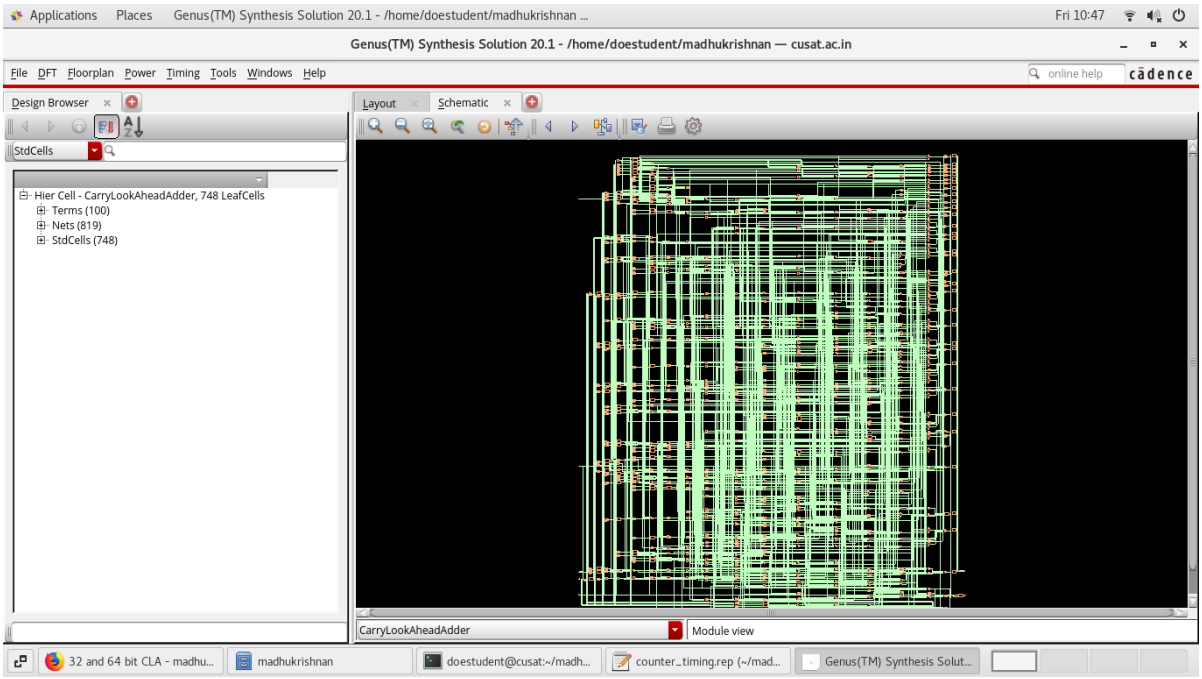

Figure 33: 32-bit Carry lookahead adder

2.4.1 Area analysis

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  10:48:43 am
  Module:                 CarryLookAheadAdder
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

      Instance      Module  Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------------
CarryLookAheadAdder               790   4646.609     0.000     4646.609 <none> (D)
  (D) = wireload is default in technology library
```

Figure 34: 32-bit Carry lookahead adder: Area

The synthesis results for the "CarryLookAheadAdder" module indicate a design consisting of 790 cells with a total area of 4646.609 units. The cell area accounts for the entire total area, as the net area is 0, suggesting no additional allocation for routing or wiring. The wireload is set to the default value specified in the technology library. The synthesis was performed under slow operating conditions with a balanced tree design strategy, emphasizing timing optimization.

2.4.2 Power analysis

```
Instance: /CarryLookAheadAdder
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     9.76451e-06  8.44209e-04  1.26164e-05  8.66590e-04  57.22%
        latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        logic     1.30353e-05  3.79471e-04  1.88442e-04  5.80948e-04  38.36%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  6.69273e-05  6.69273e-05   4.42%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     2.27999e-05  1.22368e-03  2.67985e-04  1.51446e-03 100.00%
  Percentage           1.51%       80.80%       17.70%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 35: 32-bit Carry lookahead adder: Power analysis

The power analysis for the "CarryLookAheadAdder" module indicates a total power consumption of 1.51446e-03 W, with registers being the primary contributors at 57.22 percent (8.66590e-04 W). The logic components account for 38.36 percent (5.80948e-04 W), while the clock contributes 4.42 percent (6.69273e-05 W). Other categories, including memory, latch, bbox, pad, and pm, show negligible power usage. Internal power dominates at 80.80 percent of the total, followed by switching power at 17.70 percent. Leakage power contributes a minimal 1.51 percent to the overall power consumption.

2.4.3 Timing analysis

The timing analysis for the "CarryLookAheadAdder" module's critical path indicates a setup check at pin Sum_reg[31]/CK->D, with a setup slack of 0 ps, confirming the path meets timing requirements. The clock operates at a period of 1980 ps, and the required data path time is 1793 ps. Key elements in the data path include flip-flops and logic gates such as AND2X2, NOR2X2, NAND2X1, and multiplexers, with varying delays contributing to the total data path delay. The clock and net latencies are 0, and the arrival time matches the required time, ensuring the path operates within specified timing constraints.

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  10:48:43 am
  Module:                 CarryLookAheadAdder
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


Path 1: MET (0 ps) Setup Check with Pin Sum_reg[31]/CK->D
          Group: clk
     Startpoint: (R) B_reg_reg[21]/CK
          Clock: (R) clk
       Endpoint: (R) Sum_reg[31]/D
          Clock: (R) clk

                     Capture       Launch
        Clock Edge:+    1980            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1980            0

             Setup:-     177
       Uncertainty:-      10
     Required Time:=    1793
      Launch Clock:-       0
         Data Path:-    1793
             Slack:=       0

#-------------------------------------------------------------------------------------------
#  Timing Point    Flags   Arc   Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                       (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------
  B_reg_reg[21]/CK -       -     R     (arrival)     98    -   100     0       0    (-,-)
  B_reg_reg[21]/Q  -       CK->Q F     DFFQX1         2  3.5    52   336     336    (-,-)
  g38550/Y         -       A->Y  F     AND2X2         4 11.2    68   149     484    (-,-)
  g38483/Y         -       B->Y  R     NOR2X2         6 14.4   138   122     606    (-,-)
  g38379/Y         -       A->Y  F     CLKINVX2       7 12.7    76    73     679    (-,-)
  g38363/Y         -       B->Y  R     NOR2XL         2  4.4   157   136     815    (-,-)
  g38810/Y         -       B->Y  F     NOR2BX1        2  3.8    69    63     878    (-,-)
  g38234/Y         -       B->Y  F     CLKAND2X2      3  5.0    50   132    1010    (-,-)
  g38179/Y         -       A1->Y F     OA22X1         2  4.5    78   186    1197    (-,-)
  g38132/Y         -       B->Y  R     NAND2X1        5 11.6   126   107    1303    (-,-)
  g38091/Y         -       B->Y  F     NOR2XL         1  2.7    68    65    1368    (-,-)
  g38037/Y         -       A1->Y R     OAI21X1        2  4.5   119   109    1478    (-,-)
  g38008/Y         -       B->Y  F     NOR2XL         1  2.8    66    65    1543    (-,-)
  g37990/Y         -       B0->Y R     AOI21X1        1  2.8    83    83    1626    (-,-)
  g37957/Y         -       B->Y  F     NAND3X1        1  1.7   141    91    1717    (-,-)
  g37928/Y         -       B0->Y R     OAI21XL        1  1.6   127    76    1793    (-,-)
  Sum_reg[31]/D    <<<     -     R     DFFQXL         1    -     -     0    1793    (-,-)
#-------------------------------------------------------------------------------------------
```

Figure 36: 32-bit Carry lookahead adder: Timing analysis

2.5 64-bit Carry lookahead adder

A 64-bit CLA is designed to handle the addition of two 64-bit binary numbers, providing the same speed benefits through the carry lookahead technique but for much larger data sets. As the bit width increases, the complexity of the adder grows, involving even more

sophisticated carry generation and propagation logic. However, the primary advantage remains the reduction in carry propagation delay, making 64-bit CLAs essential in high-performance computing, such as in modern processors and high-speed digital systems where 64-bit operations are common.

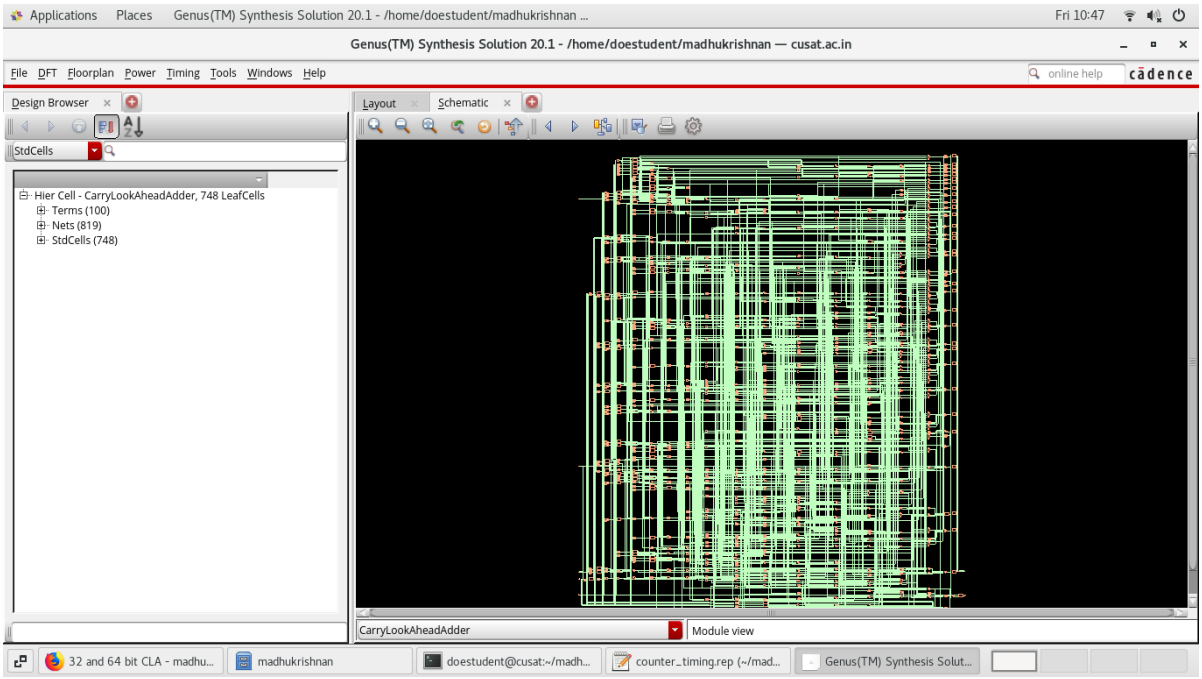

Figure 37: 64-bit Carry lookahead adder: Schematic

2.5.1 Area analysis

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  10:57:20 am
  Module:                 CarryLookAheadAdder
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

      Instance       Module  Cell Count  Cell Area  Net Area   Total Area   Wireload
-----------------------------------------------------------------------------------------
CarryLookAheadAdder              1926  10880.438     0.000    10880.438 <none> (D)
  (D) = wireload is default in technology library
```

Figure 38: 64-bit Carry lookahead adder: Area

The "CarryLookAheadAdder" module, synthesized under slow (balanced_tree) operating conditions, comprises 1926 cells with a total cell area of 10880.438 units. The net area is 0.000, indicating no additional wire area is considered in this calculation. The

total area matches the cell area, with the wireload mode set to default in the technology library.

2.5.2 Power analysis

```
Instance: /CarryLookAheadAdder
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     2.01437e-05  1.74753e-03  3.85357e-05  1.80621e-03  52.93%
        latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        logic     3.46163e-05  9.58537e-04  4.83119e-04  1.47627e-03  43.27%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  1.29681e-04  1.29681e-04   3.80%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     5.47601e-05  2.70607e-03  6.51336e-04  3.41216e-03 100.00%
  Percentage           1.60%       79.31%       19.09%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 39: 64-bit Carry lookahead adder: Power

The power analysis of the "CarryLookAheadAdder" instance reveals a total power consumption of 3.41216e-03 W, with the majority attributed to internal power (79.31 percent), followed by switching power (19.09 percent), and a negligible leakage power (1.60 percent). Registers account for 52.93 percent of the total power, consuming 1.80621e-03 W, while logic contributes 43.27 percent, consuming 1.47627e-03 W. The clock accounts for a minor 3.80 percent, with 1.29681e-04 W. Other categories like memory, latch, bbox, pad, and pm show no power contribution.

2.5.3 Timing analysis

The timing analysis for the "CarryLookAheadAdder" module examines Path 1, which begins at B_reg_reg[30]/CK and ends at Sum_reg[55]/D. This path meets the setup timing requirement with a slack of 0 ps, indicating no timing violations. The capture clock edge is at 2000 ps, with a required data arrival time of 1911 ps and an actual data path delay of 1910 ps, demonstrating precise timing compliance. The critical path involves a sequence of logic gates, including NAND2X2, NAND2BX4, NOR3XL, CLKINVX1, OA22XL, and others, each contributing to the cumulative delay with individual contributions ranging from 54 ps to 180 ps. These components work together to propagate the signal through various transitions and fanouts, ensuring accurate timing. The total path concludes at the endpoint Sum_reg[55]/D with a total data path delay of 1910 ps, effectively utilizing the available timing budget while adhering to the setup

constraints. This analysis reflects the robust timing performance of the design under the specified operating conditions.

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  10:57:19 am
  Module:                 CarryLookAheadAdder
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


Path 1: MET (0 ps) Setup Check with Pin Sum_reg[55]/CK->D
          Group: clk
     Startpoint: (R) B_reg_reg[30]/CK
          Clock: (R) clk
       Endpoint: (F) Sum_reg[55]/D
          Clock: (R) clk

                    Capture       Launch
        Clock Edge:+    2000            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    2000            0

             Setup:-      79
       Uncertainty:-      10
     Required Time:=    1911
      Launch Clock:-       0
         Data Path:-    1910
             Slack:=       0

#-------------------------------------------------------------------------------------------------
#  Timing Point    Flags   Arc   Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                      (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------------
  B_reg_reg[30]/CK -       -     R     (arrival)    194    -   100     0       0    (-,-)
  B_reg_reg[30]/Q  -       CK->Q F     DFFQX2         2  8.3    57   352     352    (-,-)
  g101097/Y        -       A->Y  R     NAND2X2        8 26.6   126   112     465    (-,-)
  g101015/Y        -       B->Y  F     NAND2BX4       7 27.9   125   117     581    (-,-)
  g100672/Y        -       C->Y  R     NOR3XL         1  2.7   180   162     743    (-,-)
  g100605/Y        -       A->Y  F     CLKINVX1       4  9.0   102    99     841    (-,-)
  g100331/Y        -       A1->Y F     OA22XL         1  1.7    60   166    1008    (-,-)
  g100255/Y        -       B->Y  R     NAND2XL        1  2.7    62    65    1073    (-,-)
  g100216/Y        -       A->Y  F     CLKINVX1       4 10.0    79    75    1148    (-,-)
  g100112/Y        -       B->Y  F     OR2X1          1  2.8    54   144    1292    (-,-)
  g100003/Y        -       A1->Y R     AOI21X1        2  4.6   125   106    1398    (-,-)
  g99942/Y         -       AN->Y R     NAND2BX1       2  4.5    84   127    1525    (-,-)
  g99877/Y         -       B->Y  R     OR2XL          1  1.6    42    89    1614    (-,-)
  g99772/Y         -       A0->Y F     OAI21XL        1  1.7    84    80    1695    (-,-)
  g99722/Y         -       B->Y  R     NOR2XL         1  2.8   111   107    1802    (-,-)
  g99523/Y         -       A1->Y F     OAI211X1       1  2.2   111   109    1910    (-,-)
  Sum_reg[55]/D    <<<     -     F     DFFHQX1        1    -     -     0    1910    (-,-)
#-------------------------------------------------------------------------------------------------
```

Figure 40: 64-bit Carry lookahead adder: Timing

3 Prefix adder

A prefix adder is a type of parallel adder used in digital circuits to perform fast addition of binary numbers. It is based on the concept of prefix computation, where intermediate carry values are precomputed and propagated through a network of logic gates. The prefix adder minimizes the propagation delay of carry bits, making it faster than traditional ripple-carry adders, where the carry bit must propagate through each bit sequentially. The most common types of prefix adders are Kogge-Stone, Brent-Kung, and Sklansky adders, each offering a different balance of speed, area, and complexity. Prefix adders are widely used in applications requiring high-speed arithmetic operations, such as in processors and digital signal processing (DSP) systems, where the reduction in carry propagation time significantly enhances overall performance. The key advantage of a prefix adder is its ability to compute the sum and carry outputs in parallel, thereby improving the overall throughput of the adder.

3.1 4-bit Prefix adder

A 4-bit Prefix Adder is a type of adder that utilizes a prefix operation to reduce the carry propagation delay in binary addition. It uses a combination of generate and propagate signals, which allow carry bits to be calculated in parallel, speeding up the addition process compared to ripple carry adders. For a 4-bit prefix adder, the carry calculation involves using a tree structure to compute the carries more efficiently. While it offers faster performance than traditional adders like the Ripple Carry Adder (RCA), it remains relatively simple and is suitable for small bit-width applications.

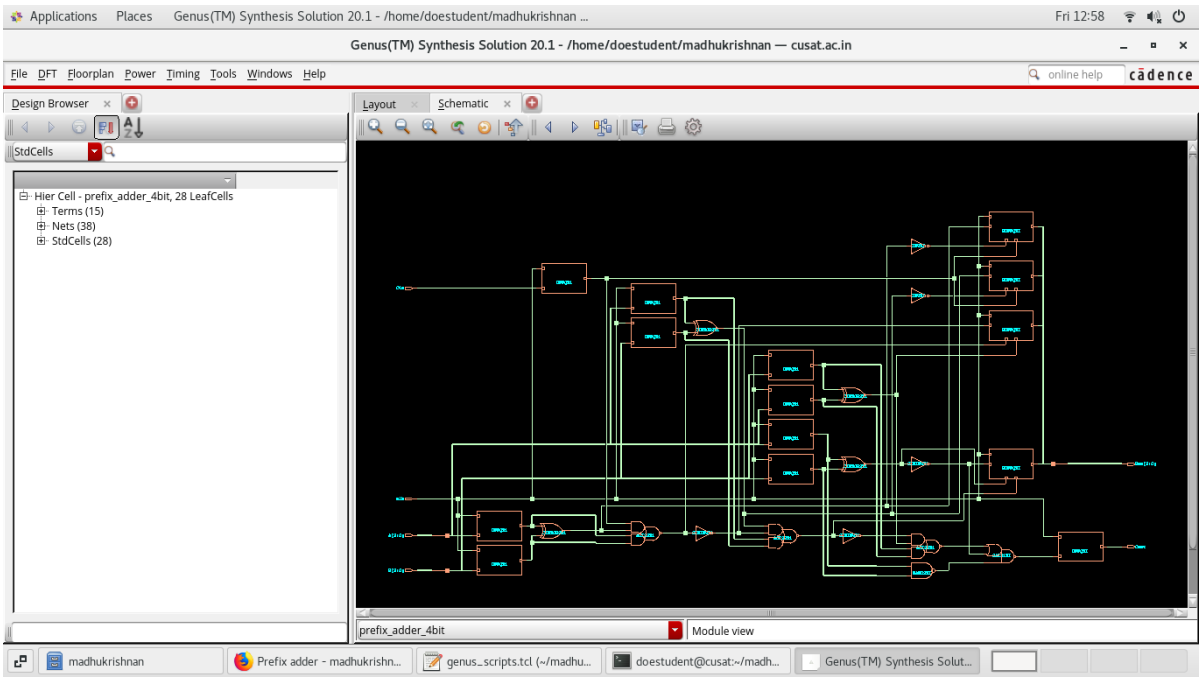

Figure 41: 4-bit Prefix adder: Schematic

3.1.1 Area analysis

The prefix_adder_4bit module has 35 cells and a total area of 336.064 units. It operates under slow conditions (balanced tree) and uses an enclosed wireload model. The area is based on the timing library, and the wireload model is the default from the technology library. This adder is likely part of a design focusing on efficient and fast addition for 4-bit inputs using the prefix addition method, which reduces carry propagation delays compared to traditional adders.

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  12:59:39 pm
  Module:                 prefix_adder_4bit
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

     Instance     Module  Cell Count  Cell Area  Net Area   Total Area   Wireload
-----------------------------------------------------------------------------------
prefix_adder_4bit                35    336.064     0.000      336.064 <none> (D)
  (D) = wireload is default in technology library
```

Figure 42: 4-bit Prefix adder: Area

3.1.2 Power analysis

```
Instance: /prefix_adder_4bit
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     1.52081e-06  2.44658e-04  6.20733e-06  2.52386e-04  84.22%
        latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        logic     4.33268e-07  1.86100e-05  9.15648e-06  2.81998e-05   9.41%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  1.90871e-05  1.90871e-05   6.37%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     1.95408e-06  2.63268e-04  3.44509e-05  2.99673e-04 100.00%
  Percentage           0.65%       87.85%       11.50%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 43: 4-bit Prefix adder: Power

The prefix_adder_4bit module consumes a total power of 2.99673e-04 W, with the majority (87.85 percent) allocated to internal power (2.63268e-04 W). Switching power

contributes 11.50 percent (3.44509e-05 W), while leakage power is minimal at 0.65 percent (1.95408e-06 W). The clock power is 1.90871e-05 W, representing 6.37 percent of the total power consumption. The design is heavily reliant on internal power, with switching and leakage power playing a smaller role.

3.1.3 Timing analysis

```
=============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  12:59:39 pm
  Module:                 prefix_adder_4bit
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
=============================================================


Path 1: MET (12 ps) Setup Check with Pin Cout_reg/CK->D
          Group: clk
     Startpoint: (R) A_ff_reg[2]/CK
          Clock: (R) clk
       Endpoint: (R) Cout_reg/D
          Clock: (R) clk

                    Capture       Launch
        Clock Edge:+    1010            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1010            0

             Setup:-     145
       Uncertainty:-      10
     Required Time:=     855
      Launch Clock:-       0
         Data Path:-     842
             Slack:=      12
#-------------------------------------------------------------------------------------------
# Timing Point   Flags   Arc   Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                 (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------
  A_ff_reg[2]/CK -       -     R     (arrival)     14    -   100     0       0    (-,-)
  A_ff_reg[2]/Q  -       CK->Q F     DFFQX1         2  4.4    58   342     342    (-,-)
  g2/Y           -       A->Y  R     CLKXOR2X1      2  5.3    81   232     573    (-,-)
  g2831/Y        -       A->Y  F     NAND2XL        2  5.4   164   130     704    (-,-)
  g2820/Y        -       A1->Y R     OAI221X1       1  2.2   145   139     842    (-,-)
  Cout_reg/D     <<<     -     R     DFFHQX1        1    -     -     0     842    (-,-)
#-------------------------------------------------------------------------------------------
```

Figure 44: 4-bit Prefix adder: Timing

The timing analysis for the prefix_adder_4bit module indicates that the setup checks for the path from A_ff_reg[2]/CK to Cout_reg/D shows a required time of 855 ps with an arrival time of 842 ps, resulting in a slack of 12 ps. The clock edges for the capture and launch are 1010 ps and 0 ps, respectively. The path involves several components, including a D flip-flop (DFFQX1) and logic gates such as CLKXOR2X1, NAND2XL, and OAI221X1, with the final timing point being Cout_reg/D, where the required data

is captured. The net latency and setup times are also factored into the calculation, ensuring the data path meets timing requirements with a small positive slack of 12 ps.

3.2 8-bit Prefix adder

An 8-bit Prefix Adder works on the same principles as the 4-bit version but is scaled up to handle 8-bit binary numbers. It uses a more complex prefix tree to calculate carry bits in parallel, which reduces the time taken to propagate the carry compared to simpler adders. The 8-bit prefix adder is typically implemented using a series of logic gates arranged in a hierarchical structure, such as the Brent-Kung or Kogge-Stone adder. This allows for faster addition operations, making it useful in systems where speed is more critical than simplicity.

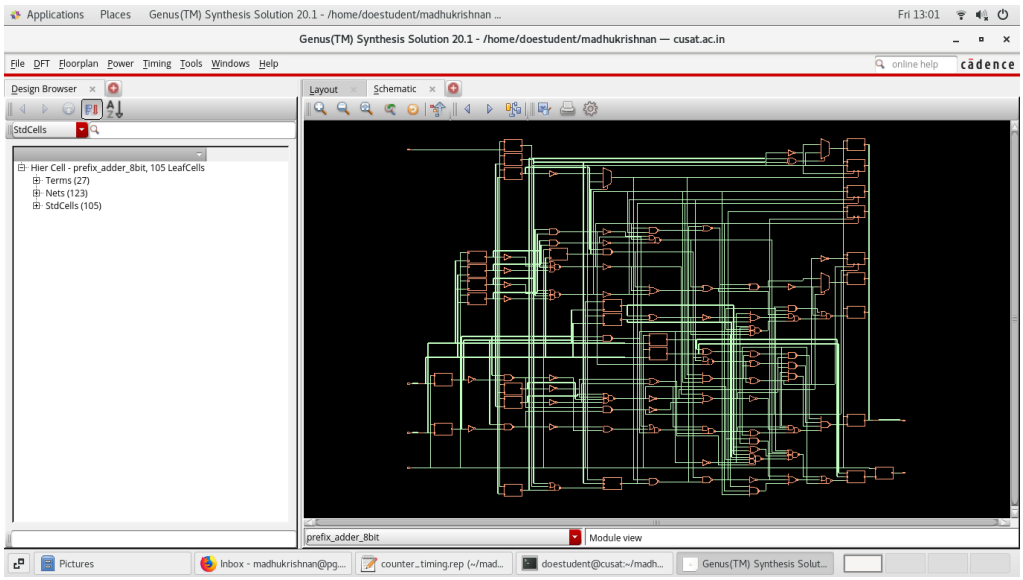

Figure 45: 8-bit Prefix adder: Schematic

3.2.1 Area analysis

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  01:01:11 pm
  Module:                 prefix_adder_8bit
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

     Instance     Module  Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------------
prefix_adder_8bit               105    784.148     0.000      784.148 <none> (D)
  (D) = wireload is default in technology library
```

Figure 46: 8-bit Prefix adder: Area

The prefix_adder_8bit module consists of 105 cells, with a total area of 784.148 units. The module was synthesized under the slow (balanced_tree) operating conditions, and the wireload model used is the default from the technology library. This configuration focuses on optimizing timing performance while maintaining an efficient use of area.

3.2.2 Power analysis

The prefix_adder_8bit module's power consumption breakdown is as follows: the registers category accounts for the largest portion, with 76.19 percent of the total power consumption, amounting to 4.10955e-04 W. The logic category follows with 18.43 percent, consuming 9.93868e-05 W. The clock category uses 5.38 percent, which is 2.90250e-05 W. The remaining categories, such as memory, latch, and pad, contribute 0 percent. The overall power consumption totals 5.39367e-04 W, with 85.75 percent of the power being internal, 13.41 percent due to switching, and a small fraction due to leakage.

```
Instance: /prefix_adder_8bit
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     2.99926e-06  3.96501e-04  1.14547e-05  4.10955e-04  76.19%
        latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        logic     1.53366e-06  6.60208e-05  3.18323e-05  9.93868e-05  18.43%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  2.90250e-05  2.90250e-05   5.38%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     4.53292e-06  4.62522e-04  7.23120e-05  5.39367e-04 100.00%
  Percentage          0.84%       85.75%       13.41%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 47: 8-bit Prefix adder: Power

3.2.3 Timing analysis

The timing analysis of the prefix_adder_8bit module highlights the critical path between the A_ff_reg[4]/CK flip-flop and the Sum_reg[7]/D flip-flop. The analysis begins with the arrival of the clock edge at A_ff_reg[4]/CK, with a launch edge at 0 ps. The path then propagates through several combinational logic gates, such as CLKINVX3, NAND2X2, NOR2X1, and OAI21X2, before reaching the endpoint at Sum_reg[7]/D. The setup time required for the path is 192 ps, and the uncertainty in timing is 10 ps. The required time for the data to arrive at the endpoint is 998 ps, which is matched by the total delay through the data path, resulting in 0 ps slack. This indicates that the design is operating at the edge of its timing constraints, with no margin for error. The flip-flops involved in the path are DFFQX4 for A_ff_reg[4]/Q and DFFQXL for Sum_reg[7]/D,

both contributing to the total delay. The analysis ensures that the timing requirements for the prefix_adder_8bit module are met, with the signal propagation completed in exactly the required time.

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  01:01:11 pm
  Module:                 prefix_adder_8bit
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
====================================================================


Path 1: MET (0 ps) Setup Check with Pin Sum_reg[7]/CK->D
          Group: clk
     Startpoint: (R) A_ff_reg[4]/CK
          Clock: (R) clk
       Endpoint: (R) Sum_reg[7]/D
          Clock: (R) clk

                     Capture       Launch
        Clock Edge:+    1200            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1200            0

             Setup:-     192
       Uncertainty:-      10
     Required Time:=     998
      Launch Clock:-       0
         Data Path:-     998
             Slack:=       0
#-------------------------------------------------------------------------------------------------
# Timing Point   Flags   Arc   Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                 (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------------
  A_ff_reg[4]/CK -       -     R     (arrival)     26    -   100     0       0    (-,-)
  A_ff_reg[4]/Q  -       CK->Q F     DFFQX4         3 16.0    57   373     373    (-,-)
  g10541/Y       -       A->Y  R     CLKINVX3       1  5.3    31    38     412    (-,-)
  g10534/Y       -       B->Y  F     NAND2X2        2  8.0    75    67     478    (-,-)
  g10530/Y       -       A->Y  R     CLKINVX1       1  2.6    42    50     528    (-,-)
  g10519/Y       -       A->Y  F     NOR2X1         1  5.3    53    53     581    (-,-)
  g10498/Y       -       A1->Y R     OAI21X2        1  2.8    91    70     651    (-,-)
  g10490/Y       -       A1->Y F     AOI21X1        3  7.9   161   131     782    (-,-)
  g10476/Y       -       A->Y  R     NOR2X1         1  2.8    91   100     882    (-,-)
  g10473/Y       -       B0->Y F     AOI31X1        1  1.7   114    42     923    (-,-)
  g10466/Y       -       D->Y  R     NAND4XL        1  1.6    70    74     998    (-,-)
  Sum_reg[7]/D   <<<     -     R     DFFQXL         1    -     -     0     998    (-,-)
#-------------------------------------------------------------------------------------------------
```

Figure 48: 8-bit Prefix adder: Timing

3.3 16-bit Prefix adder

A 16-bit Prefix Adder extends the concept of the 8-bit version, handling 16-bit binary numbers. As the bit width increases, the complexity of the prefix tree also increases,

but the benefit remains the same: reducing carry propagation delay by performing the carry calculations in parallel. The 16-bit prefix adder is often implemented using more advanced prefix-based structures, such as the Kogge-Stone or Han-Carlson adder, which balance speed and area efficiency. This makes it suitable for medium to high-performance applications, such as digital signal processors and microprocessors.

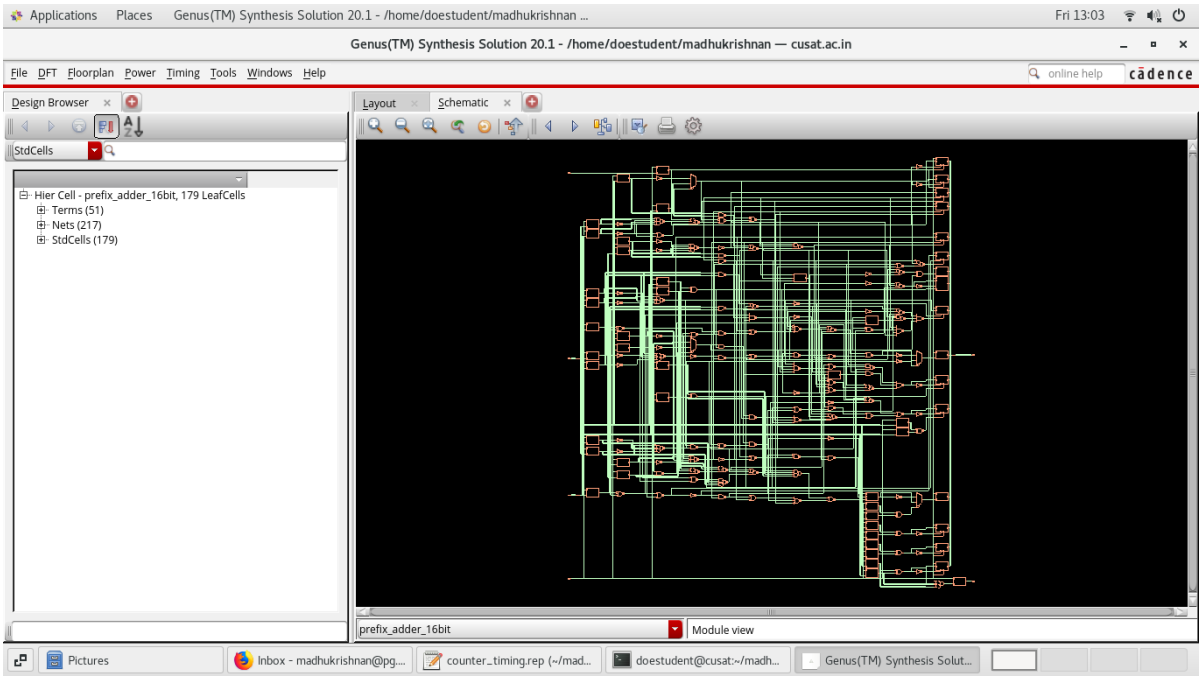

Figure 49: 16-bit Prefix adder: Schematic

3.3.1 Area analysis

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  01:02:55 pm
  Module:                 prefix_adder_16bit
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

     Instance      Module  Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------------
prefix_adder_16bit               179   1455.519     0.000     1455.519 <none> (D)
  (D) = wireload is default in technology library
```

Figure 50: 16-bit Prefix adder: Area

The prefix_adder_16bit module is synthesized with a total of 179 cells, and its total area is 1455.519 units. The cell area is 1455.519 units, with no additional net area, indicating that all of the area is used by the cells themselves. The wireload model

used is enclosed, with no specific wireload parameters defined in the technology library (marked as <none> (D)). This configuration ensures that the adder is optimized for performance while maintaining a reasonable area utilization.

3.3.2 Power analysis

```
Instance: /prefix_adder_16bit
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     5.62406e-06  6.58274e-04  1.65752e-05  6.80473e-04  77.83%
        latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        logic     2.84461e-06  9.59259e-05  4.57545e-05  1.44525e-04  16.53%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  4.92551e-05  4.92551e-05   5.63%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     8.46867e-06  7.54200e-04  1.11585e-04  8.74254e-04  99.99%
  Percentage           0.97%       86.27%       12.76%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 51: 16-bit Prefix adder: Power

The prefix_adder_16bit module's power consumption is predominantly driven by internal power, which accounts for 86.27 percent of the total, equating to 7.54200e-04 W. The register components contribute significantly, making up 77.83 percent of the power usage. Switching power, at 12.76 percent, contributes 1.11585e-04 W, while clock power consumes 5.63 percent of the total, or 4.92551e-05 W. Leakage power represents a smaller fraction, at 0.97 percent, amounting to 8.46867e-06 W. The overall power breakdown highlights the dominant role of registers and internal power in the module's energy dissipation.

3.3.3 Timing analysis

The detailed timing analysis of the prefix_adder_16bit module highlights that the setup check for Path 1 involves multiple stages, starting with the A_ff_reg[2]/CK flip-flop and ending at the Sum_reg[9]/D flip-flop. The capture clock edge is 1360 ps, and the launch clock edge is 0 ps, resulting in an overall setup time of 133 ps with an uncertainty of 10 ps. The required time for the setup is 1217 ps, and the data path delay is exactly 1217 ps, leaving 0 ps slack, which indicates that this is a critical path. The analysis also includes detailed breakdowns of the individual gates and components, such as DFFQX2, CLKINVX1, OAI22X2, NOR2XL, MXI2XL, and others, each contributing specific delays. For instance, the DFFQX2 flip-flop has a delay of 357 ps, CLKINVX1 contributes delays of 57 ps and 63 ps, and the OAI22X2 gate adds 112 ps to the total

delay. This information is crucial for understanding the timing constraints and performance of the prefix_adder_16bit design, ensuring that the circuit meets the required timing specifications for reliable operation.

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  01:02:54 pm
  Module:                 prefix_adder_16bit
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
====================================================================


Path 1: MET (0 ps) Setup Check with Pin Sum_reg[9]/CK->D
          Group: clk
     Startpoint: (R) A_ff_reg[2]/CK
          Clock: (R) clk
       Endpoint: (R) Sum_reg[9]/D
          Clock: (R) clk

                     Capture       Launch
        Clock Edge:+    1360            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1360            0

             Setup:-     133
       Uncertainty:-      10
     Required Time:=    1217
      Launch Clock:-       0
         Data Path:-    1217
             Slack:=       0
#-------------------------------------------------------------------------------------------
# Timing Point   Flags   Arc   Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                    (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------
  A_ff_reg[2]/CK -       -     R     (arrival)     50    -   100     0       0    (-,-)
  A_ff_reg[2]/Q  -       CK->Q F     DFFQX2         3  9.9    62   357     357    (-,-)
  g4275/Y        -       A->Y  R     CLKINVX1       1  5.3    57    63     420    (-,-)
  g4238/Y        -       A1->Y F     OAI22X2        5 11.2   126   112     532    (-,-)
  g4220/Y        -       A->Y  R     CLKINVX1       2  7.1    84    92     625    (-,-)
  g4198/Y        -       A1->Y F     OAI21X2        4  9.9   109    95     720    (-,-)
  g4144/Y        -       B0->Y R     AOI21X1        4 10.8   209   182     902    (-,-)
  g4137/Y        -       B->Y  F     NOR2XL         1  2.8    91    75     976    (-,-)
  g4134/Y        -       B->Y  R     NOR2X1         2  4.3    98    98    1074    (-,-)
  g4131/Y        -       A->Y  F     CLKINVX1       1  1.7    40    37    1111    (-,-)
  g4122/Y        -       A->Y  R     MXI2XL         1  2.2   114   106    1217    (-,-)
  Sum_reg[9]/D   <<<     -     R     DFFHQX1        1    -     -     0    1217    (-,-)
#-------------------------------------------------------------------------------------------
```

Figure 52: 16-bit Prefix adder: Timing

3.4 32-bit Prefix adder

A 32-bit Prefix Adder handles larger binary numbers by employing a more extensive hierarchical structure for carry generation and propagation. As the bit width increases,

the prefix adder's complexity grows, but the carry computation is still parallelized, allowing for significantly faster addition compared to simpler adders like the Ripple Carry Adder (RCA). A 32-bit prefix adder typically uses optimized adder designs such as the Kogge-Stone or the Ladner-Fischer adder, which offer fast computation at the cost of increased hardware complexity. This makes it ideal for high-speed processors and large-scale digital systems where fast addition is crucial.

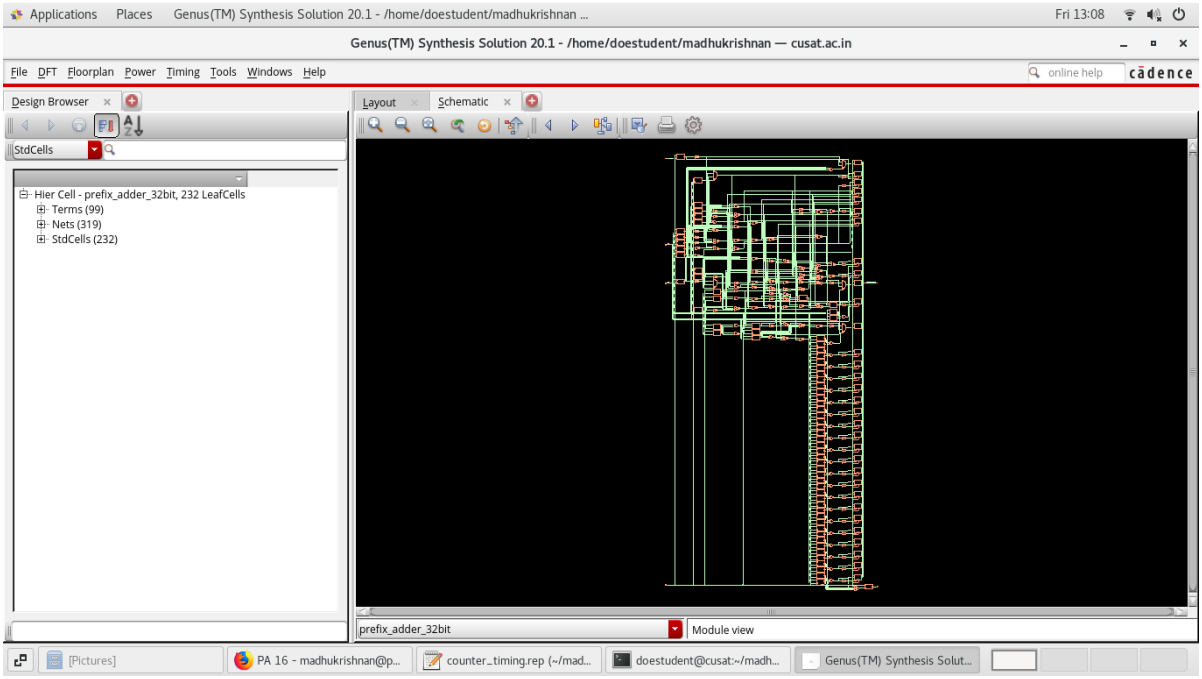

Figure 53: 32-bit Prefix adder: Schematic

3.4.1 Area analysis

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  01:07:45 pm
  Module:                 prefix_adder_32bit
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

     Instance      Module  Cell Count  Cell Area  Net Area   Total Area   Wireload
-------------------------------------------------------------------------------------
prefix_adder_32bit                232   2319.142     0.000     2319.142 <none> (D)
  (D) = wireload is default in technology library
```

Figure 54: 32-bit Prefix adder: Area

The prefix_adder_32bit module, generated using Genus Synthesis Solution, consists of 232 cells and occupies a total area of 2319.142 units. This design is characterized by

a slow operating condition, using a balanced tree approach for wireload modelling. The area calculation does not include the wireload, as indicated by the default setting in the technology library. This module represents a significant step in scaling the adder design to 32 bits, increasing both the complexity and area compared to smaller bit-width adders like 16-bit and 8-bit.

3.4.2 Power analysis

```
Instance: /prefix_adder_32bit
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     1.06008e-05  1.22657e-03  2.20606e-05  1.25923e-03  84.38%
        latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        logic     3.62205e-06  9.14136e-05  4.02153e-05  1.35251e-04   9.06%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  9.79147e-05  9.79147e-05   6.56%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     1.42229e-05  1.31798e-03  1.60191e-04  1.49240e-03 100.00%
  Percentage           0.95%       88.31%       10.73%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 55: 32-bit Prefix adder: Power

The power consumption breakdown for the prefix_adder_32bit module shows that registers account for 1.25923e-03 W, which is 84.38 percent of the total power usage. Logic components consume 1.35251e-04 W, or 9.06 percent, while the clock consumes 9.79147e-05 W, making up 6.56 percent of the total power. Leakage power is minimal at 1.42229e-05 W, representing 0.95 percent. The total power consumption for the module is 1.49240e-03 W.

3.4.3 Timing analysis

The prefix_adder_32bit module's path analysis further elaborates on the specific components and delays in the data path. The setup check is triggered from A_ff_reg[0]/CK to Sum_reg[9]/D, both driven by the same clock signal. The path traverses through various logic gates, starting with a DFFQX1 flip-flop and passing through a series of logic gates such as NOR2X1, OAI21X2, NAND3X1, and MXI2XL. These gates introduce varying delays, such as 110 ps, 98 ps, and 132 ps, before the final arrival at Sum_reg[9]/D at 1217 ps. The total path delay ensures that the data arrives at the register with sufficient timing, as indicated by a slack value of 0 ps, meaning the timing requirements are exactly met, with no margin for improvement.

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  01:07:45 pm
  Module:                 prefix_adder_32bit
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
====================================================================


Path 1: MET (0 ps) Setup Check with Pin Sum_reg[9]/CK->D
          Group: clk
     Startpoint: (R) A_ff_reg[0]/CK
          Clock: (R) clk
       Endpoint: (R) Sum_reg[9]/D
          Clock: (R) clk

                    Capture       Launch
        Clock Edge:+    1360            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1360            0

             Setup:-     133
       Uncertainty:-      10
     Required Time:=    1217
      Launch Clock:-       0
         Data Path:-    1217
             Slack:=       0
#-------------------------------------------------------------------------------------------
# Timing Point   Flags   Arc   Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                    (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------
  A_ff_reg[0]/CK -       -     R     (arrival)     98    -   100     0       0    (-,-)
  A_ff_reg[0]/Q  -       CK->Q F     DFFQX1         3  7.1    75   357     357    (-,-)
  g3090/Y        -       A->Y  R     NOR2X1         1  5.3   110   110     467    (-,-)
  g3020/Y        -       A1->Y F     OAI21X2        3  9.4    97    98     565    (-,-)
  g2978/Y        -       B->Y  R     NAND3X1        1  5.3    93    85     650    (-,-)
  g2975/Y        -       C->Y  F     NAND3X2        3  7.3   124   108     759    (-,-)
  g2970/Y        -       B->Y  R     NAND2X1        2  4.5    69    79     838    (-,-)
  g2964/Y        -       B->Y  F     NAND2X1        3  8.1   132   113     951    (-,-)
  g2962/Y        -       A->Y  R     CLKINVX1       1  2.7    57    62    1013    (-,-)
  g2952/Y        -       A1->Y F     OAI21X1        2  4.3    95    84    1097    (-,-)
  g2942/Y        -       B->Y  R     MXI2XL         1  2.2   116   119    1217    (-,-)
  Sum_reg[9]/D   <<<     -     R     DFFHQX1        1    -     -     0    1217    (-,-)
#-------------------------------------------------------------------------------------------
```

Figure 56: 32-bit Prefix adder: Timing

3.5 64-bit Prefix adder

A 64-bit Prefix Adder extends the prefix addition technique to handle 64-bit binary numbers. At this size, the carry computation involves a complex multi-level prefix tree that reduces carry propagation delay to a minimum, ensuring high-speed addition even for large data sizes. The 64-bit prefix adder is typically implemented using advanced designs like the Kogge-Stone adder or the Sklansky adder, which provide excellent speed performance but come with increased hardware overhead. This type of adder is commonly used in modern processors, high-performance computing systems, and applications that

require fast arithmetic operations on large data sets.

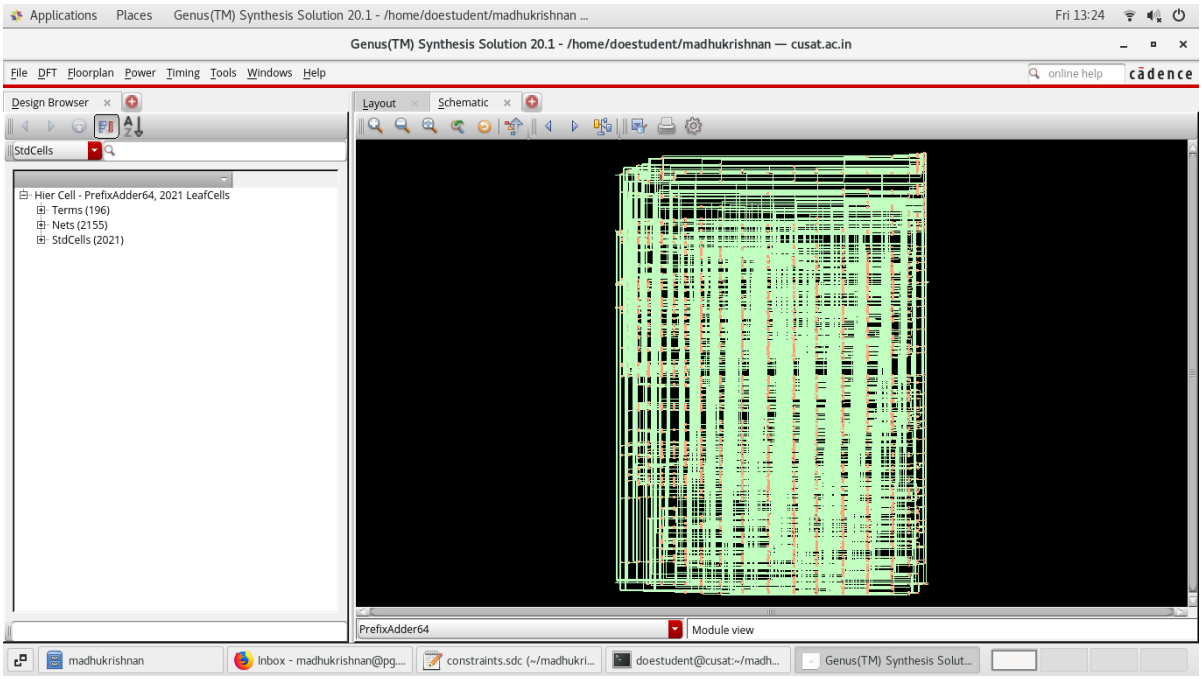

Figure 57: 64-bit Prefix adder: Schematic

3.5.1 Area analysis

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  01:23:37 pm
  Module:                 PrefixAdder64
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

   Instance    Module  Cell Count  Cell Area  Net Area   Total Area   Wireload
-----------------------------------------------------------------------------------
PrefixAdder64              2021  12195.930     0.000    12195.930 <none> (D)
  (D) = wireload is default in technology library
```

Figure 58: 64-bit Prefix adder: Area

The PrefixAdder64 module consists of 2021 cells with a total area of 12195.930, derived from the cell area of 12195.930 and no additional net area. The wireload mode is set to enclosed, and the area mode is based on the timing library. No wireload value is specified, as it is assumed to be the default for the technology library.

3.5.2 Power analysis

The PrefixAdder64 instance has a total power consumption of 4.05509e-03 W, with 81.43 percent (3.30210e-03 W) allocated to internal power, 17.10 percent (6.93602e-04 W) to switching power, and 1.46 percent (5.93889e-05 W) to leakage power. The majority of the power is consumed by registers, accounting for 61.73 percent, followed by logic components at 34.91 percent, and the clock at 3.36 percent.

```
Instance: /PrefixAdder64
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     2.43568e-05  2.41965e-03  5.93045e-05  2.50331e-03  61.73%
       latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       logic     3.50321e-05  8.82450e-04  4.98134e-04  1.41562e-03  34.91%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     0.00000e+00  0.00000e+00  1.36163e-04  1.36163e-04   3.36%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     5.93889e-05  3.30210e-03  6.93602e-04  4.05509e-03 100.00%
  Percentage           1.46%       81.43%       17.10%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 59: 64-bit Prefix adder: Power

3.5.3 Timing analysis

The timing analysis focuses on the path from A_reg_reg[48]/CK to Sum_reg[61]/SI, where the setup check shows the signal arrival time at 1950 ps, with a required setup time of 1712 ps. The clock capture edge occurs at 1950 ps, and the launch edge is at 0 ps. The total slack for this path is 0 ps, indicating that the timing is tight and meets the required specifications without any delay margin.

The analysis also provides a detailed breakdown of the delays across various logic gates and flip-flops in the data path. The signal starts at A_reg_reg[48]/CK and passes through several gates, including NOR2X1, NAND2BX4, CLKAND2X2, among others, which contribute to the total delay. It ultimately reaches the endpoint Sum_reg[61]/SI, where it is captured by the SDFFRHQX2 flip-flop. This critical timing path analysis underscores the performance of the logic gates and flip-flops in the PrefixAdder64 design, ensuring that the signal arrives within the required timing constraints with no violations.

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 20.11-s111_1
  Generated on:           Dec 13 2024  01:23:36 pm
  Module:                 PrefixAdder64
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


Path 1: MET (0 ps) Setup Check with Pin Sum_reg[61]/CK->SI
          Group: clk
     Startpoint: (R) A_reg_reg[48]/CK
          Clock: (R) clk
       Endpoint: (R) Sum_reg[61]/SI
          Clock: (R) clk

                     Capture       Launch
        Clock Edge:+    1950            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1950            0

             Setup:-     228
       Uncertainty:-      10
     Required Time:=    1712
      Launch Clock:-       0
         Data Path:-    1712
             Slack:=       0

#-------------------------------------------------------------------------------------------
#  Timing Point    Flags   Arc    Edge   Cell      Fanout Load Trans Delay Arrival Instance
#                                                        (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------
  A_reg_reg[48]/CK -       -      R     (arrival)    194    -   100     0       0    (-,-)
  A_reg_reg[48]/Q  -       CK->Q  R     DFFRHQX1       2  8.1   101   383     383    (-,-)
  g109599/Y        -       A->Y   F     NOR2X1         1  2.9    54    55     438    (-,-)
  g109540/Y        -       AN->Y  F     NAND2BX4       7 17.1    90   174     612    (-,-)
  g109280/Y        -       B->Y   R     NOR2X1         3  5.5   116   111     723    (-,-)
  g109133/Y        -       B->Y   R     CLKAND2X2      2  4.9    49   128     851    (-,-)
  g108976/Y        -       B->Y   R     CLKAND2X2      5 14.2    83   145     996    (-,-)
  g108907/Y        -       B->Y   F     NAND2X2        6 19.2   158   129    1124    (-,-)
  g108684/Y        -       A2->Y  R     OAI32X1        2  3.3   203   175    1300    (-,-)
  g108558/Y        -       B0->Y  F     AOI21XL        1  2.6   108    76    1376    (-,-)
  g108332/Y        -       A->Y   R     NAND3X1        1  2.8    66    76    1452    (-,-)
  g108214/Y        -       A1->Y  F     AOI21X1        1  3.6   114    86    1538    (-,-)
  g108137/Y        -       B->Y   R     CLKXOR2X1      1  2.7    65   174    1712    (-,-)
  Sum_reg[61]/SI   <<<     -      R     SDFFRHQX2      1    -     -     0    1712    (-,-)
#-------------------------------------------------------------------------------------------
```

Figure 60: 64-bit Prefix adder: Timing

4 Comparison of results

4.1 Comparison of area

Bit Width	RCA Area	CLA Area	Prefix Adder Area
4-bit	377.693	435.974	336.064
8-bit	781.121	1016.517	784.148
16-bit	1597.816	2069.365	1455.519
32-bit	2848.215	4646.609	2319.142
64-bit	5765.307	10880.438	12195.930

Table 1: Comparison of Area: 4-bit, 8-bit, 16-bit, 32-bit, and 64-bit Adders

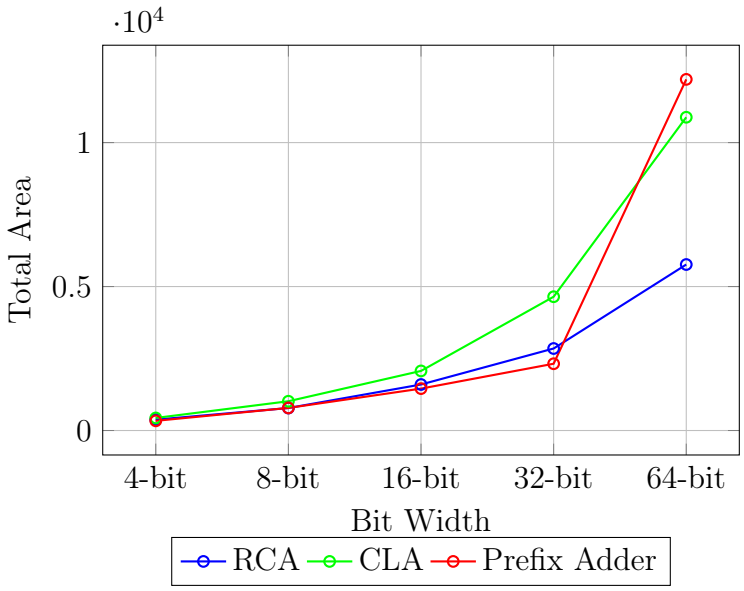

Figure 61: Comparison of Area: 4-bit, 8-bit, 16-bit, 32-bit, and 64-bit Adders

From the line graph depicting the total area of the 4-bit, 8-bit, 16-bit, 32-bit, and 64-bit Ripple Carry Adder (RCA), Carry Lookahead Adder (CLA), and Prefix Adder, the following key observations can be made:

- Increase in Area with Bit Width: For all three adder types (RCA, CLA, and Prefix Adder), the total area increases as the bit width increases. This is expected because the complexity of the adder grows with the number of bits it needs to handle, leading to an increase in the required logic gates and interconnections.
- The growth in the area is not linear but rather exponential, reflecting the increasing complexity as the bit width doubles.

4.2 Comparison of power

4.2.1 Internal Power

Adder Type	4-bit	8-bit	16-bit	32-bit	64-bit
RCA	227.819	388.725	472.157	421.346	449.570
CLA	370.312	562.856	925.886	1,223.680	2,706.070
Prefix Adder	263.268	462.522	754.200	1,317.980	3,302.100

Table 2: Internal Power Consumption in μW.

4.2.2 Leakage Power

Adder Type	4-bit	8-bit	16-bit	32-bit	64-bit
RCA	1.9605	4.2958	8.8091	15.563	31.3774
CLA	2.4809	5.2791	10.179	22.7999	54.7601
Prefix Adder	1.9541	4.5329	8.4687	14.2229	59.3889

Table 3: Leakage Power Consumption in μW.

4.2.3 Switching Power

Adder Type	4-bit	8-bit	16-bit	32-bit	64-bit
RCA	20.7963	37.3475	53.2119	42.0801	46.1517
CLA	38.2123	81.7777	153.858	267.985	651.336
Prefix Adder	34.4509	72.3120	111.585	160.191	693.602

Table 4: Switching Power Consumption in μW.

4.2.4 Total Power

Adder Type	4-bit	8-bit	16-bit	32-bit	64-bit
RCA	250.575	430.369	534.178	478.989	527.099
CLA	411.005	649.913	1.08992	1.51446	3.41216
Prefix Adder	299.673	539.367	874.254	1.49240	4.05509

Table 5: Total Power Consumption in μW for 4-bit to 64-bit adders. Values in milliwatts (mW) are indicated where applicable.

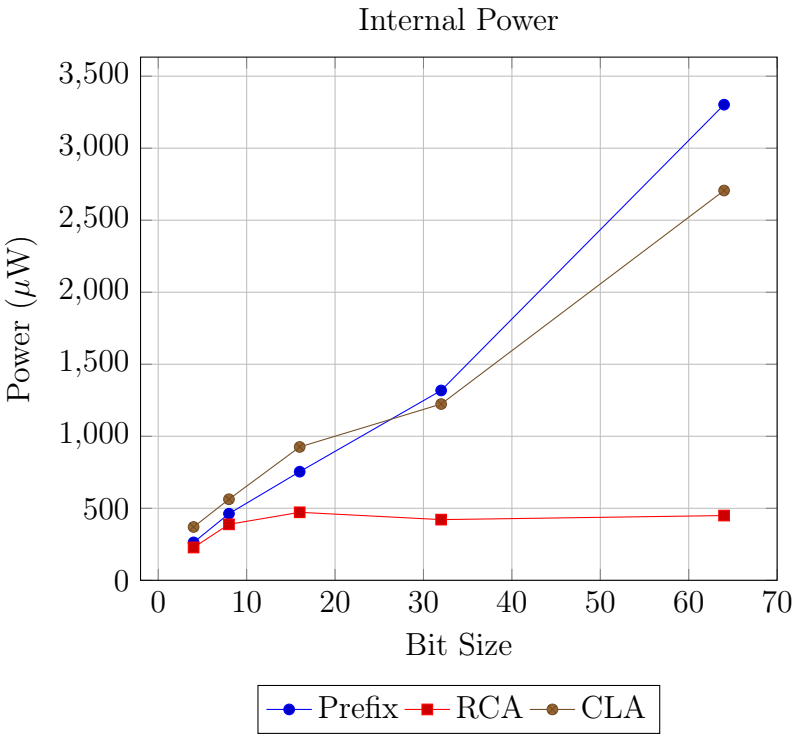

Figure 62: Internal Power

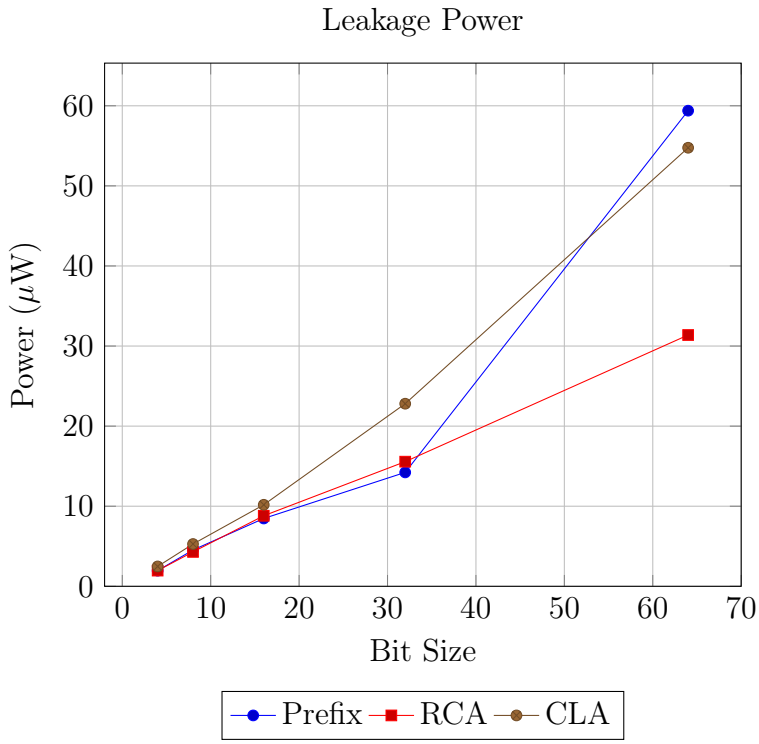

Figure 63: Leakage Power

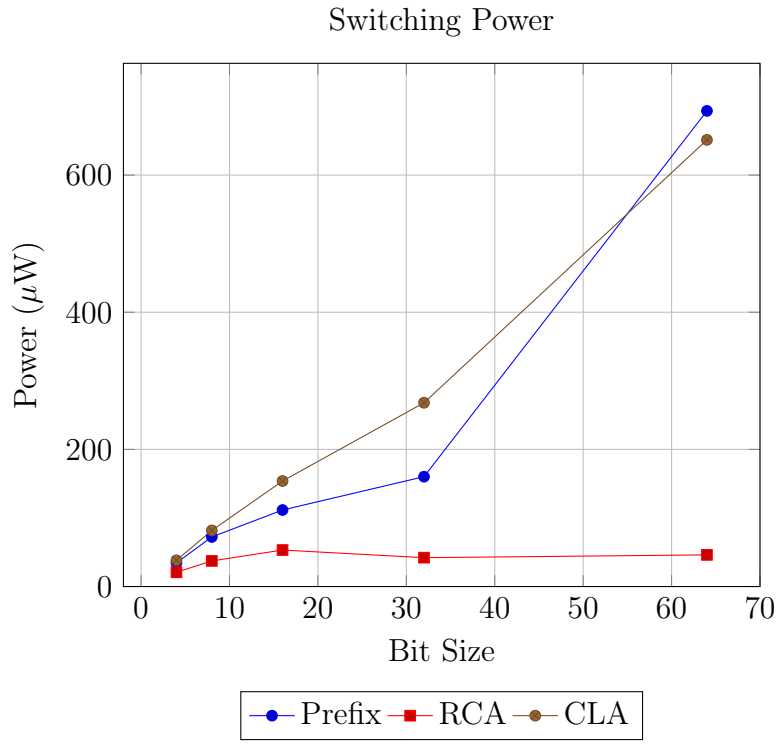

Figure 64: Switching Power

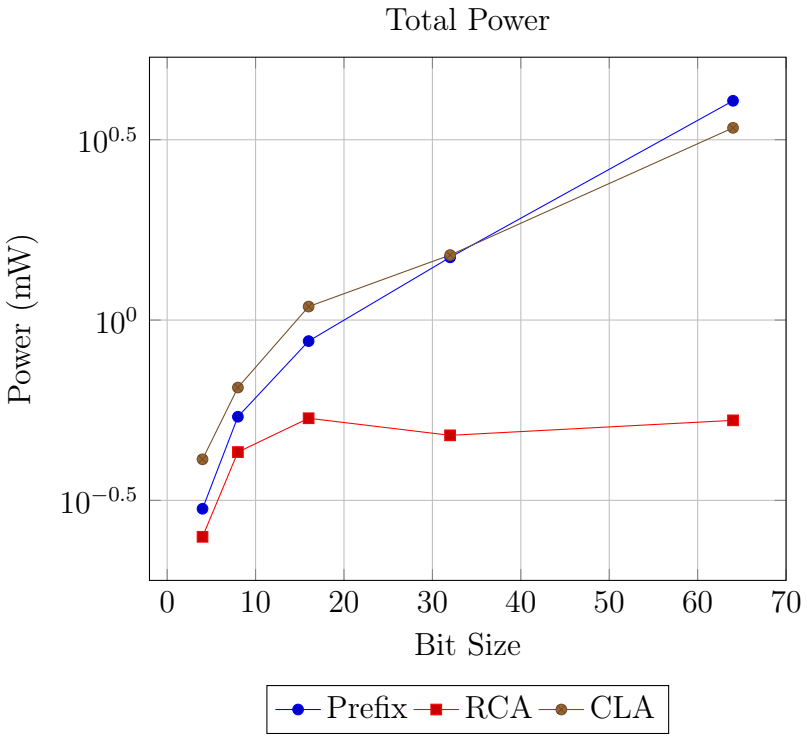

Figure 65: Total Power

The power consumption analysis of various adders, including Ripple Carry Adder (RCA), Carry Lookahead Adder (CLA), and Prefix Adders, reveals distinct trade-offs in internal, leakage, switching, and total power. RCA generally exhibits the lowest total power consumption, making it energy-efficient for small-scale operations. However, its performance degrades as bit-width increases due to slower carry propagation. CLA, while consuming more power across all categories, offers faster computation due to parallel carry generation, making it suitable for high-speed applications despite higher energy demands. Prefix Adders provide a balanced approach, consuming moderate power with scalable performance benefits, particularly for larger bit-widths. This trade-off between power efficiency and computational speed highlights the importance of selecting the appropriate adder design based on application-specific requirements.

4.3 Comparison of Timing

Bit Width	RCA (ps/ns)	CLA (ps/ns)	Prefix Adder (ps/ns)
4-bit	1390 ps / 1.39 ns	1010 ps / 1.01 ns	1010 ps / 1.01 ns
8-bit	1630 ps / 1.63 ns	1230 ps / 1.23 ns	1200 ps / 1.20 ns
16-bit	2580 ps / 2.58 ns	1480 ps / 1.48 ns	1360 ps / 1.36 ns
32-bit	5470 ps / 5.47 ns	1980 ps / 1.98 ns	1360 ps / 1.36 ns
64-bit	10200 ps / 10.2 ns	2000 ps / 2.00 ns	1950 ps / 1.95 ns

Table 6: Clock Edge for Different Adders at Zero Slack (ps/ns)

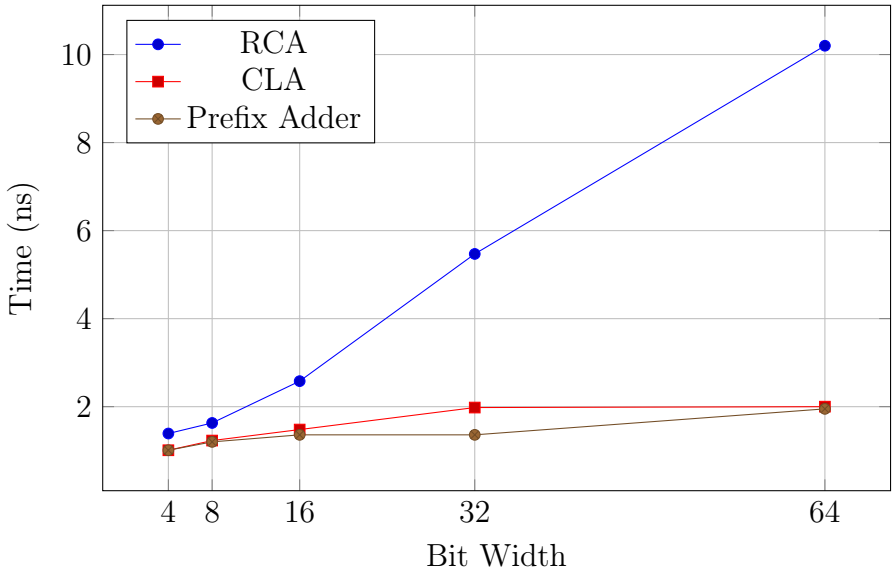

Figure 66: Timing Comparison of RCA, CLA, and Prefix Adder (in ns)

The timing analysis reveals the performance differences between RCA, CLA, and Prefix Adder designs across varying bit widths. RCA demonstrates the highest delay, with

its timing increasing significantly as bit width grows, highlighting its limited scalability. CLA shows moderate performance, with better timing than RCA due to its optimized carry propagation, though its complexity becomes evident at higher bit widths. The Prefix Adder consistently exhibits the lowest delay, showcasing its efficiency in managing carry propagation through a parallel structure. These results emphasize the suitability of Prefix Adders for high-speed applications, while RCA remains practical for simpler, low-bit-width designs.

5 Conclusion

The area, power, and timing analysis of various adders—Ripple Carry Adder (RCA), Carry Lookahead Adder (CLA), and Prefix Adders—demonstrates distinct trade-offs, making each suitable for specific applications. RCA is the most area-efficient and power-efficient for smaller bit-widths but suffers from significant timing delays due to its sequential carry propagation, limiting its scalability. CLA, with its parallel carry generation, offers superior timing performance, especially for larger bit-widths, but at the cost of increased area and power consumption. Prefix Adders strike a balance between timing and scalability, providing efficient performance for high-bit-width operations, though they require slightly more area and power than RCA. The analysis underscores the importance of selecting the appropriate adder based on the specific requirements of power, area, and speed, as no single design is universally optimal.