Implementation and Comparison of various types of 16 bit adders on FPGA board

For

Mini Project – 2B: FPGA Design Project (ECM501)
(REV- 2019 'C' Scheme) of Third Year (Semester-VI)

Bachelors in Engineering

By

Aishvarya Birambole

Isha Chavan

Risa Samanta

Saakshi Karkera

Supervisor

Mrs. Neeta Chavan

Department of Electronics and Telecommunication

Vivekanand Education Society's Institute of Technology

Mumbai University

2021-2022

Mini Project-2B Approval

Project entitled by Isha Chavan, Aishvarya Birambole, Risa Samanta and Saaskshi Karkera under the guidance of Mrs. Neeta Chavan is approved for degree of Bachelor of Engineering.

Examiners

1. ___________________
2. ___________________

Supervisors

1. ____________________
2. ____________________

Date:
Place:

Certificate

This is to certify that Isha Chavan, Saakshi Karkera, Aishvarya Birambole, Risa Samanta have completed the project report on the topic Implementation and Comparison of various types of 16 bit adders on FPGA board satisfactorily in partial fulfillment of the requirements for the award of Mini Project 2B (REV- 2019 ‘C’ Scheme) of Third Year, (Semester-VI) in Electronics and Telecommunication under the guidance of Mrs. Neeta Chavan during the year 2021-2022 as prescribed by University of Mumbai.

Supervisor

Head of Department
Dr. Chandansingh Rawat

Principal
Mrs. Jayalakshmi Nair

Examiner 1

Examiner 2

Declaration

We declare that this written submission represents our ideas in our words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misinterpreted or fabricated or falsified any idea/data/fact/source in my submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Isha Chavan)

(Saakshi Karkera)

(Aishvarya Birambole)

(Risa Samanta)

Date:

Table of Contents

Abstract

VLSI designers are constantly working towards the optimization of speed, power, and area of circuits, but practically it is difficult to optimize all at the same time. This project presents a comparative study of the designs of various adders- Carry Select Adder, CarrySkip Skip Adder, Ripple Carry Adder and Carry Look Ahead Adder, which have been designed using Xilinx ISE 14.7 Design Suite and synthesized for Spartan 3A FPGA. All the adders have been designed for 16-bit operands and a comparison of delay performance and area utilization has been made as per the data obtained from the synthesis results. The performance of speed and area of adder designs has been analyzed, and it has been observed that both the parameters cannot be optimized at the same time. If parallelism of an adder increases in order to increase the speed of operation, then it will result in large area occupancy; and if area is to be optimized then we have to adjust with the slow speed of the system.

List of figures

List of Tables

Chapter 1

Introduction

1. Introduction

With the advancements in VLSI technology, the circuit designs are getting miniature in size, consuming lesser power for performing their intended operation and becoming faster in operation. We all know that area, power and speed are the major constraints in VLSI design, and the designers are taking enormous efforts to improve their designs relative to these constraints, but all of these cannot be improved simultaneously.

In this modern era of technological advancements, everything is becoming fast-paced and heading towards completely digital processes. Hence, there is an immense need of developing faster processors which would operate on digital signals, but as we head towards improving any one of the design parameters, the other parameters are also affected, and so with the improvement in speed of operation of any circuit, its area occupancy also increases.

In circuits like digital signal processor (DSP), microprocessor, or arithmetic and logic unit (ALU) of any processor, the unit performing arithmetic operations is very important when considered with respect to the design constraints mentioned above. Most arithmetic circuits consist of adder, subtractor, multiplier, divider, etc.; the adder unit being the most basic unit among all the other units.

1. Carry Select Adder: The carry-select adder comprises two ripple carry adders and a multiplexer. Addition of two n-bit numbers with a carry-select adder is computed with two adders, therefore we use two ripple carry adders. In order to perform the calculation twice, one time by assuming the carry-in being zero and the other time assuming it will be one. After both the results are calculated, the correct sum, and the correct carry-out, is selected by the multiplexer once when the correct carry-in is known.

2. Carry Skip Adder: Carry skip adder enhances the delay of Ripple Carry Adder with less effort when compared with other adders. In the worst case, the improvement in delay is achieved by making use of several carry skipSkip adders to form a block-carry-skip-adder. The performance of this carry skip adder can be enhanced with only a few combinations of input bits.

3. Ripple Carry Adder: The ripple carry adder computes the prefixes for 2 bit groups. These prefixes are used to find the prefixes for the 4 bit groups, which in turn are used to compute the prefixes for 8 bit groups and so on. These prefixes are then used to compute the carry out of the particular bit stage. These carries will be used along with the Group Propagate of the next stage to compute the Sum bit of that stage. Brent Kung Tree will be using 2log2N - 1 stages

4. Carry Look Ahead Adder: - Carry Look Ahead Adders are also called as fast adders as it reduces the time compared to other adders by propagating carry before the sum output, leading to great performance. In our project, a 16-Bit Carry Look Ahead Adder is implemented using xilinx tool and their simulation is done. Using the logical expressions for the carry propagation (P) and carry generation (G) the carry and sum of the adder (1-Bit) is generated.

Chapter 2

Review of Literature

2.1 Amala Maria Alex, Nidhish Antony[1]: In this paper the performance of Carry Select adder using two methods is compared. In Carry Select Adder the possible values of input carry are 0 and 1. So in advance, the result can be calculated. Further we have the multiplexer stage, for calculating the result in its advanced stage. The conventional design is the use of dual Ripple Carry Adders (RCAs) and then there is a multiplexer stage. Here, one RCA (Cin=0) is replaced by Brent kung adder. As, RCA (for Cin=1) and Brent Kung adder (for Cin=0) consume more chip area, so an add-one scheme i.e Binary to Excess-l converter is introduced. By using parallel prefix adder, delay and power consumption of different adder architectures is reduced. As, parallel prefix adders derive fast results therefore Brent kung adder is used.

2.2 Maroju SaiKumar, Dr. P. Samundiswary[2]: In this paper, the design of various adders such as Ripple Carry Adder, Carry Skip Adder, Carry Increment Adder, Carry Look Ahead Adder, Carry Save Adder, Carry Select Adder, Carry Skip Adder are discussed and the performance parameters of adders such as area and delay are determined and compared. Various adders are designed using Verilog HDL. Then, they are simulated and synthesized using Xilinx ISE 13.2 for Virtex-6 family devices with speed grade -2. It is observed that Ripple Carry Adder, Carry Increment Adder and Carry Select Adder are having better performance in terms of area (LUT's and Slices). Carry Increment Adder , Carry Save Adder, Ripple Carry Adder achieve better results in terms of delay.

2.3 Shrikanth K. Shirakol, Ajaykumar S. Kulkarni, A.F.Akash, Nikhil N. Amminabhavi and Aditya Parvati [3] : In this paper, the adder circuits which perform superior performance are compared. The carry-skip adder proposed here reduces the time needed to propagate the carry by skipping over groups of consecutive adder stages, and is known to be comparable in speed to the carry look-ahead technique while it uses less logic area and less power . Carry Skip Adder (CSA) is simulated for different structures such as 2, 4 and 8-blocks. Simulation results show that CSA is faster than RCA. Furthermore 8-block CSA is faster than 4-block CSA & 4-block CSA is faster than 2-block CSA.

2.4 Sajesh Kumar , Mohamed Salih [4] : In this paper the design of carry select adder is implemented with Kogge Stone tree using two different approaches. Both the adders match in terms of their performance and are much better than simple Kogge Stone adders. The new configuration is then implemented on a Spartan 3E XC3S1600E FPGA device and the performance is compared. Results show that CSA implemented with Kogge Stone shows better performance in terms of delay, even though its area is somewhat more than that implemented with RCA.

Chapter 3

Project Description

3.1 Problem Definition

Adders are one of the widely used digital components in digital integrated circuit design. It has special significance in VLSI design and is used in computers and many other processors. In the rapidly growing mobile industry, faster units are not the only concern but also smaller areas and less power become major concerns for the design of digital circuits. There are many types of adder designs available so finding which would be best for which conditions is a task. That’s why there is a need to compare the available adders.

3.2 Steps involved

1. Open Xilinx 14.7 for windows 10.
2. Create a new project for each adder and write their respective Verilog codes.
3. Synthesize the codes.
4. After that create test benches for those adders and save them.
5. Go to simulation, click on Behavioral Check syntax and then click on Simulate Behavioral Model to observe the waveforms.
6. After simulation, we will be able to observe the waveforms for given inputs and their respective output.

3.3 Block Diagram of proposed project:

1. Carry Select Adder:

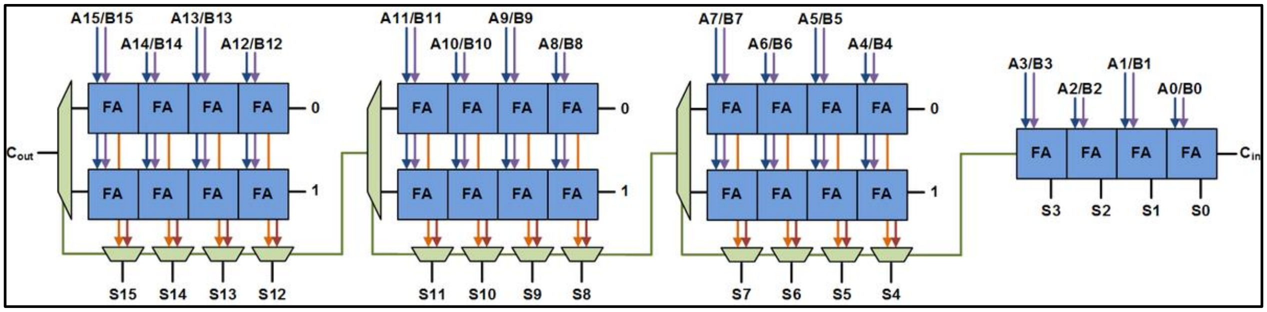


Fig1: Carry Select Adder

2. Carry Skip Adder:

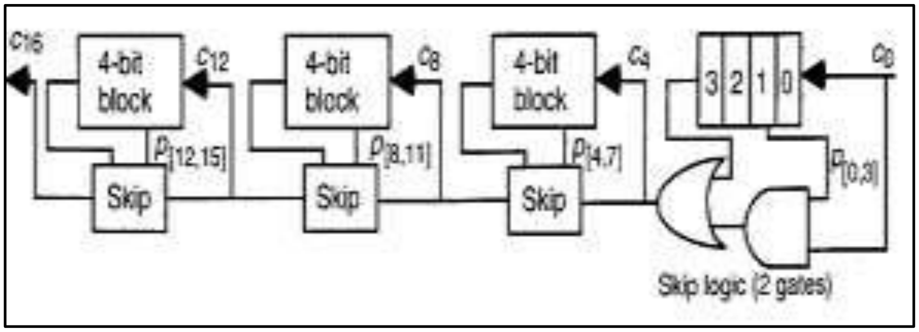


Fig2: Carry Skip Adder

3. Ripple Carry Adder:

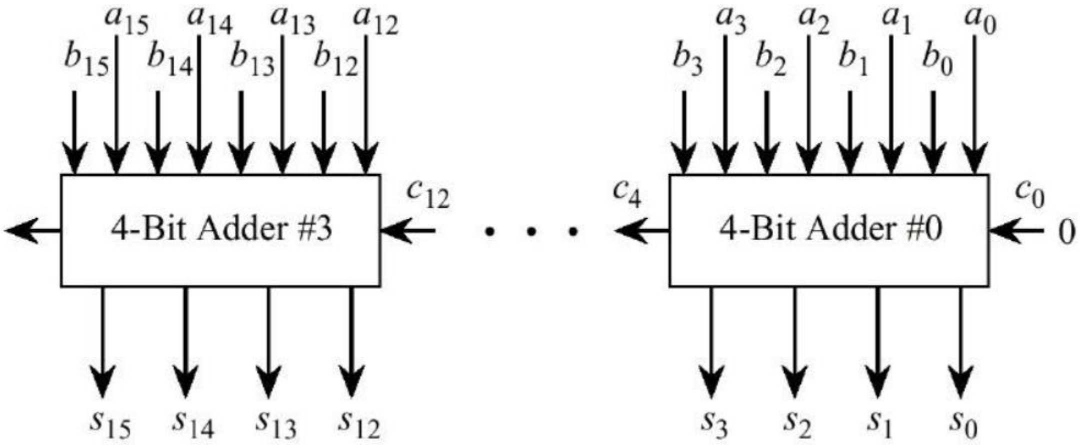


Fig3: Ripple Carry Adder

4. Carry Lookahead Adder

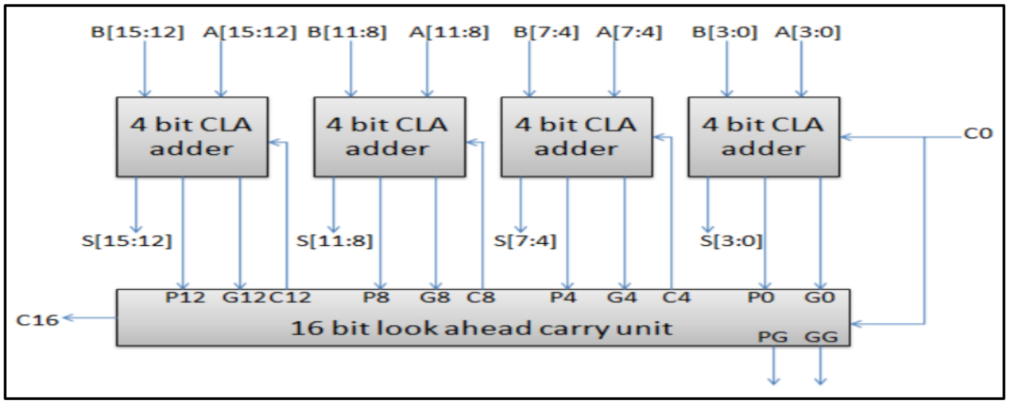


Fig4: Carry Lookahead Adder

3.4 Component Description

3.4.1 Hardware - Elbert V2 Spartan 3A FPGA Development Board

Elbert V2 is an easy to use FPGA Development board featuring Xilinx Spartan-3A FPGA. Elbert V2 is specially designed for experimenting and learning system design with FPGAs. This development board features Xilinx XC3S50A TQG144 FPGA. The USB 2.0 interface provides fast and easy configuration download to the on-board SPI flash. You don’t need a programmer or special downloader cable to download the bit stream to the board.

Board features :

- FPGA: Spartan XC3S50A in TQG144 package
- Flash memory: 16 Mb SPI flash memory (M25P16)
- USB 2.0 interface for On-board flash programming
- FPGA configuration via JTAG and USB
- 8 LEDs, Six Push Buttons and 8 way DIP switch for user defined purposes
- One VGA Connector
- One Stereo Jack
- One Micro SD Card Adapter
- Three Seven Segment Displays
- 39 IOs for user defined purposes
- On-board voltage regulators for single power rail operation

Applications:

- Product Prototype Development
- Home Networking
- Signal Processing
- Wired and Wireless Communications
- Educational tool for schools and universities

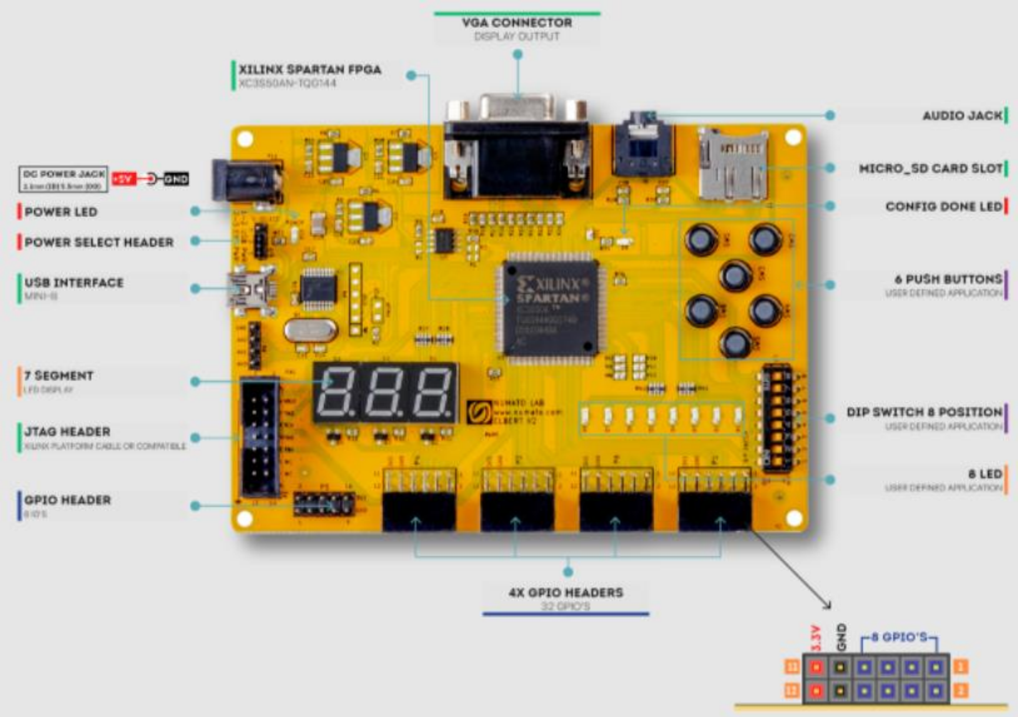


Fig5: Elbert V2 Spartan 3A

3.4.3 LED

3.4.4 DIP Switches

3.4.5 Resistors

3.4.6 Software - Xilinx ISE WebPACK

Xilinx ISE WebPACK design software is the industry's only FREE, fully featured front-to-back FPGA design solution for Linux, Windows XP, and Windows 10. ISE WebPACK is the ideal downloadable solution for FPGA and CPLD design offering HDL synthesis and simulation, implementation, device fitting, and JTAG programming. ISE WebPACK delivers a complete, front-to-back design flow providing instant access to the ISE features and functionality at no cost. Xilinx has created a solution that allows convenient productivity by providing a design solution that is always up to date with error-free downloading and single file installation.

Key Features:

- A free, downloadable PLD design environment for both Microsoft Windows and Linux!

- Embedded processing design support for the Zynq-7000 SoC family the Z-7010, Z-7020, and Z-7030
- The industry's fastest timing closure with Xilinx SmartCompile technology
- Complete, front-to-back design environment, including the Xilinx CORE Generator™ system and the full PlanAhead design and analysis tool — with new RTL to Bitstream design flow for Logic Designers!
- Integrated HDL verification with the Lite version of the ISE Simulator (ISim)
- The easiest, lowest cost way to get started with the industry leader for productivity, performance, and power
- Easily upgradeable to any of the ISE Design Suite Editions from the Xilinx Online Store.

3.5 Working of proposed project

We have implemented 16 bit adders on Elbert V2 FPGA Board and compared their performances using a synthesis report.

3.5.1 Carry Select Adder

We have created a full adder module with two half adder modules. Then created a 4-bit ripple carry adder module with 4 full adder modules. We have used two ripple carry adders. In order to perform the calculation twice, one time by assuming the carry-in being zero and the other time assuming it will be one. After both the results are calculated, the correct sum, and the correct carry-out, is selected by the multiplexer once when the correct carry-in is known. Here we have created 2:1 MUX with parameter width = 16.

3.5.2 Carry Skip Adder

We have created a full adder module with two half adder modules. Then created a 4-bit ripple carry adder module with 4 full adder modules. Then we have created a generated file to get all propagated bits.

Carry Skip adder enhances the delay of Ripple Carry Adder with less effort when compared with other adders. In the worst case, the improvement in delay is achieved by making use of several carry skip adders to form a block-carry-skip-adder. The performance of this carry skip adder can be enhanced with only a few combinations of input bits.

3.5.3 Ripple Carry Adder

We have created a full adder module with two half adder modules. Then created a 4-bit ripple carry adder module with 4 full adder modules. Create a 16 bit ripple carry adder module with four 4-bit ripple carry adder modules and one 4-bit ripple carry adder module.

3.5.3 Carry Lookahead Adder

We have created a full adder module with two half adder modules. Then created a 4-bit ripple carry adder module with 4 full adder modules. We have used two ripple carry adders. In order to perform the calculation twice, one time by assuming the carry-in being zero and the other time assuming it will be one. After both the results are calculated, the correct sum, and the correct carry-out, is selected by the multiplexer once when the correct carry-in is known. Here we have created 2:1 MUX with parameter width = 16.

Chapter 4

Implementation

4.1 Hardware

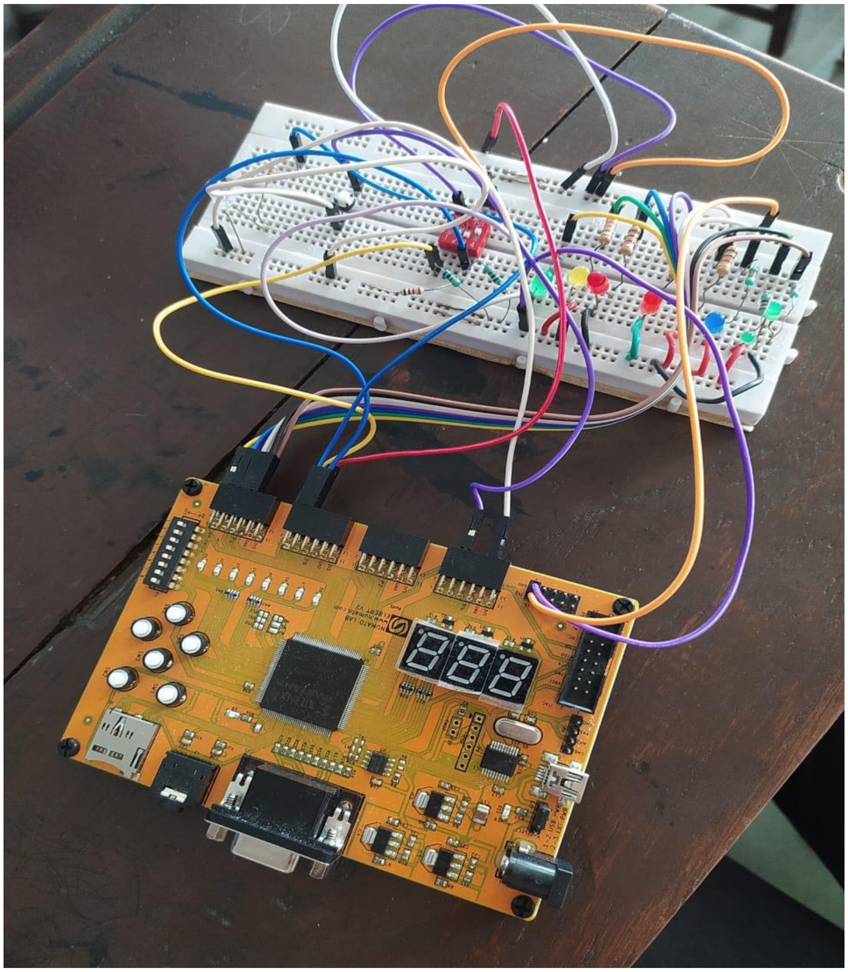


Fig6: Hardware

4.2 Software (Flowchart/Algorithms)

1.Carry Select Adder(16 bit):

Step 1: Create a Half adder module.

Step 2: Create a full adder module with two half adder modules.

Step 3: Create a 4-bit ripple carry adder module with 4 full adder modules.

Step 4: Create 2:1 MUX module with parameter width=16.

Step 5: Create a 4-bit carry select adder slice module with two 4 bit ripple carry adder and 2: 1 MUX modules.

Step 6: Create a 16 bit carry select adder module with three 4-bit carry select adder modules and one 4-bit ripple carry adder module.

2. Carry Skip Adder(16 bit):

Step 1: Create a Half adder module.

Step 2: Create a full adder module with two half adder modules.

Step 3: Create a 4-bit ripple carry adder module with 4 full adder modules.

Step 4: Create a generate file to get all propagate bits.

Step 5: Create 2:1 MUX module.

Step 6: Create a 4-bit carry select adder slice module with one 4 bit ripple carry adder and 2: 1 MUX modules.

Step 7: Create a 16 bit carry skip adder module with four 4-bit carry skip adder modules.

3. Ripple Carry Adder(16 bit):

Step 1: Create a Half adder module.

Step 2: Create a full adder module with two half adder modules.

Step 3: Create a 4-bit ripple carry adder module with 4 full adder modules.

Step 4: Create a 16 bit ripple carry adder module with four 4-bit ripple carry adder modules and one 4-bit ripple carry adder module.

4. Carry Lookahead Adder(16 bit):

Step 1: Create a Half adder module.

Step 2: Create a full adder module with two half adder modules.

Step 3: Create a 4-bit carry lookahead adder module with 4 full adder modules.

Step 4:Create a 4-bit carry lookahead adder slice module.

Step 5: Create a 16 bit carry lookahead adder module with four 4-bit carry lookahead adder modules.

Chapter 5

Results

5.1 Xillinx Simulation Results

1. Carry Select Adder

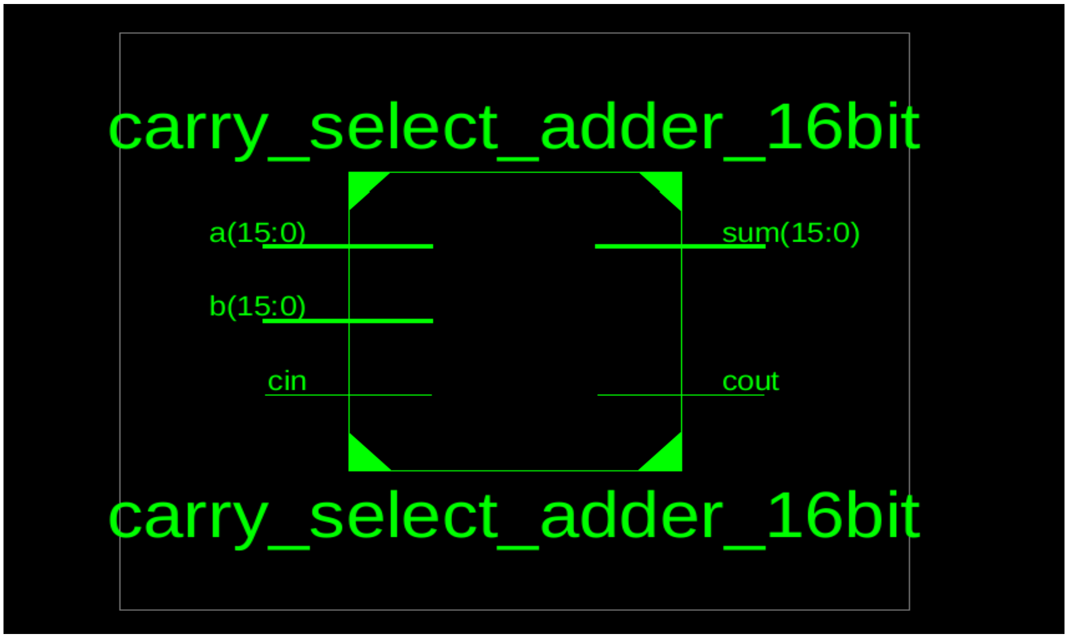


Fig7: RTL Schematic

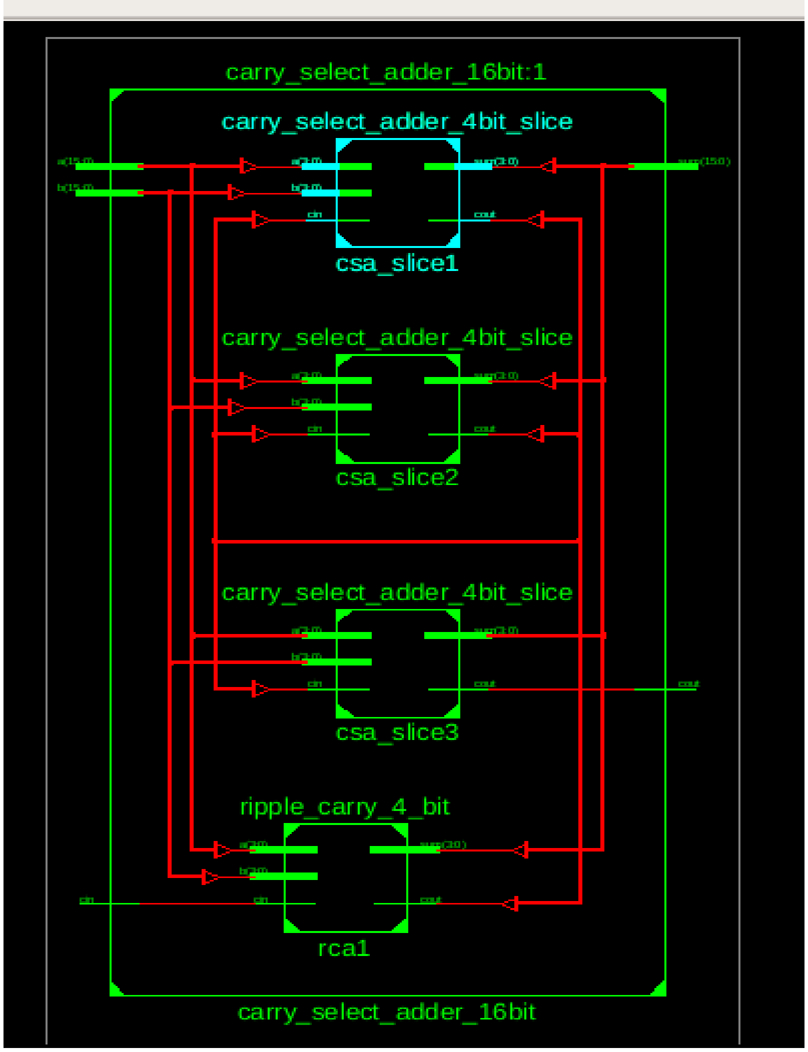


Fig 7.1: RTL Schematic

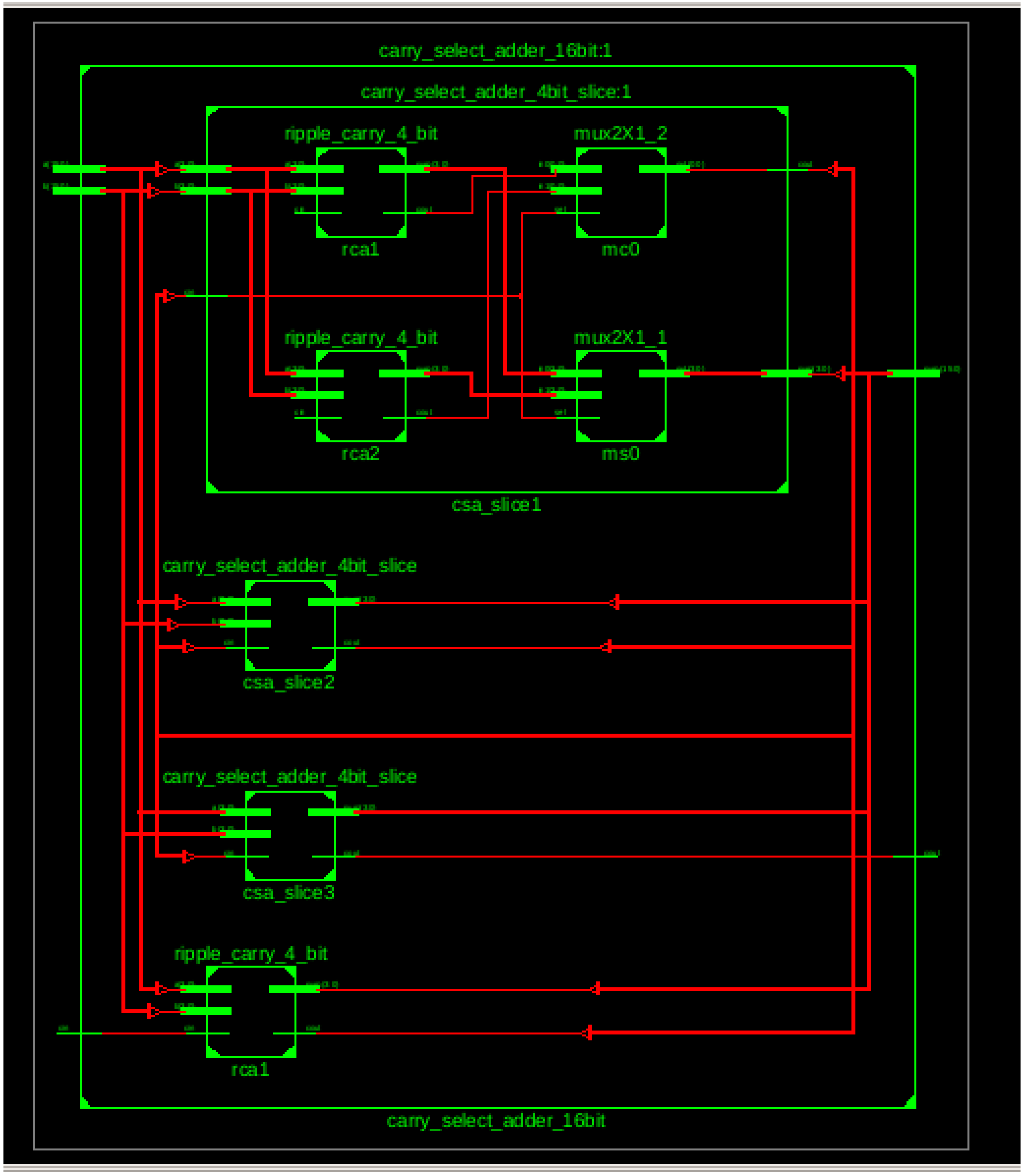
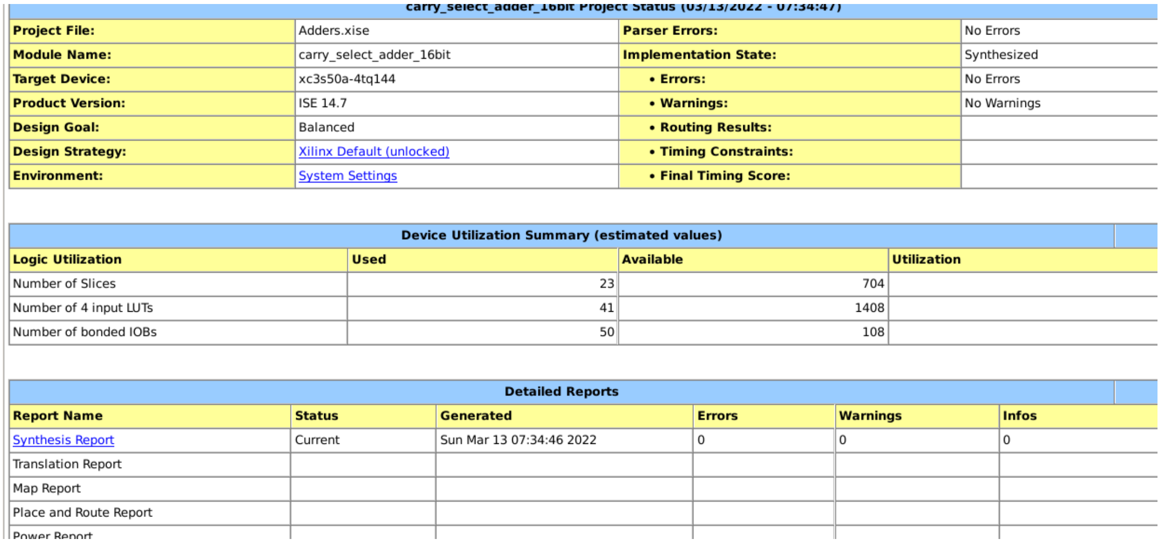


Fig8: Tech Schematic



carry_select_adder_16bit Project Status (03/13/2022 - 07:34:47)

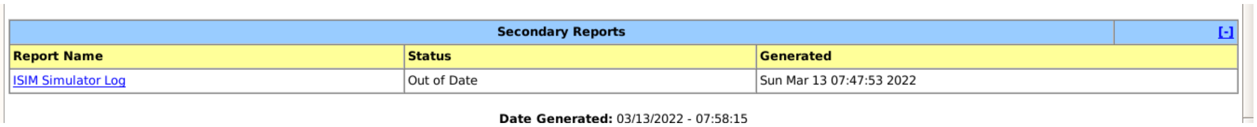
Project File:	Adders.xise	**Parser Errors:**	No Errors
Module Name:	carry_select_adder_16bit	**Implementation State:**	Synthesized
Target Device:	xc3s50a-4tq144	• **Errors:**	No Errors
Product Version:	ISE 14.7	• **Warnings:**	No Warnings
Design Goal:	Balanced	• **Routing Results:**	
Design Strategy:	Xilinx Default (unlocked)	• **Timing Constraints:**	
Environment:	System Settings	• **Final Timing Score:**	

Device Utilization Summary (estimated values)

Logic Utilization	Used	Available	Utilization
Number of Slices	23	704	
Number of 4 input LUTs	41	1408	
Number of bonded IOBs	50	108	

Detailed Reports

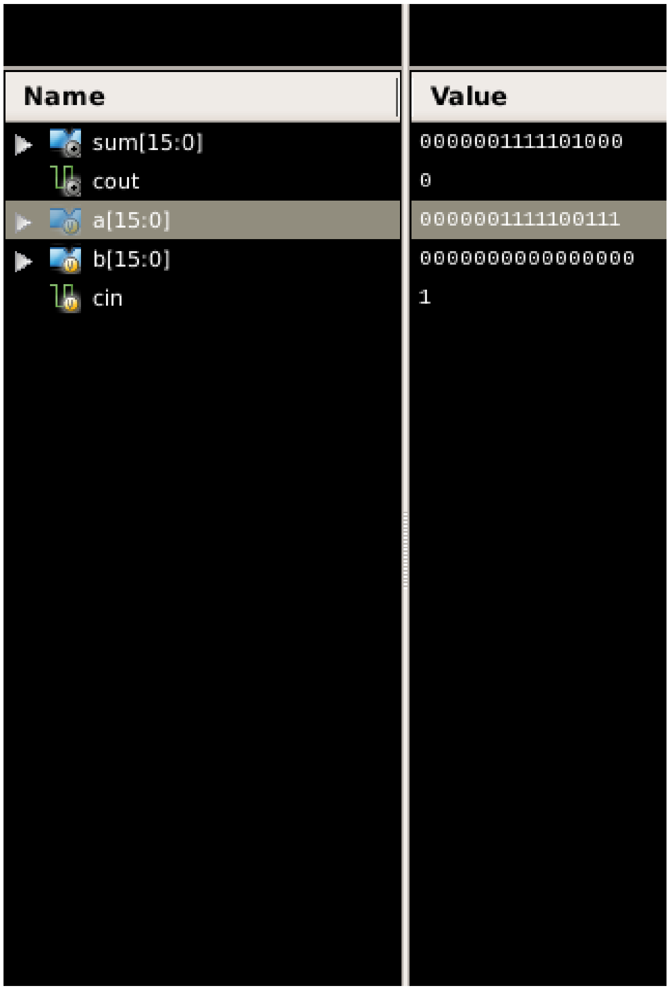
Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	Sun Mar 13 07:34:46 2022	0	0	0
Translation Report					
Map Report					
Place and Route Report					
Power Report					



Secondary Reports		[-]
Report Name	**Status**	**Generated**
ISIM Simulator Log	Out of Date	Sun Mar 13 07:47:53 2022

Date Generated: 03/13/2022 - 07:58:15

Fig9: Synthesize summary



Name	Value
sum[15:0]	0000001111101000
cout	0
a[15:0]	0000001111100111
b[15:0]	0000000000000000
cin	1

Fig10: Simulation

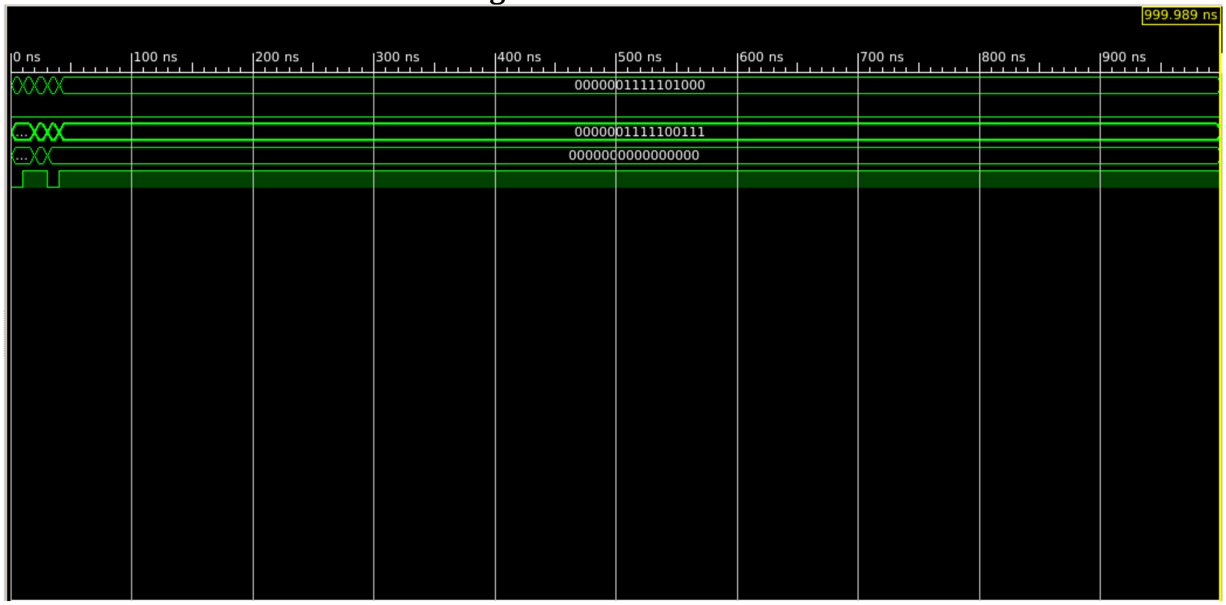


Fig10.1: Simulation

2. Carry Skip Adder

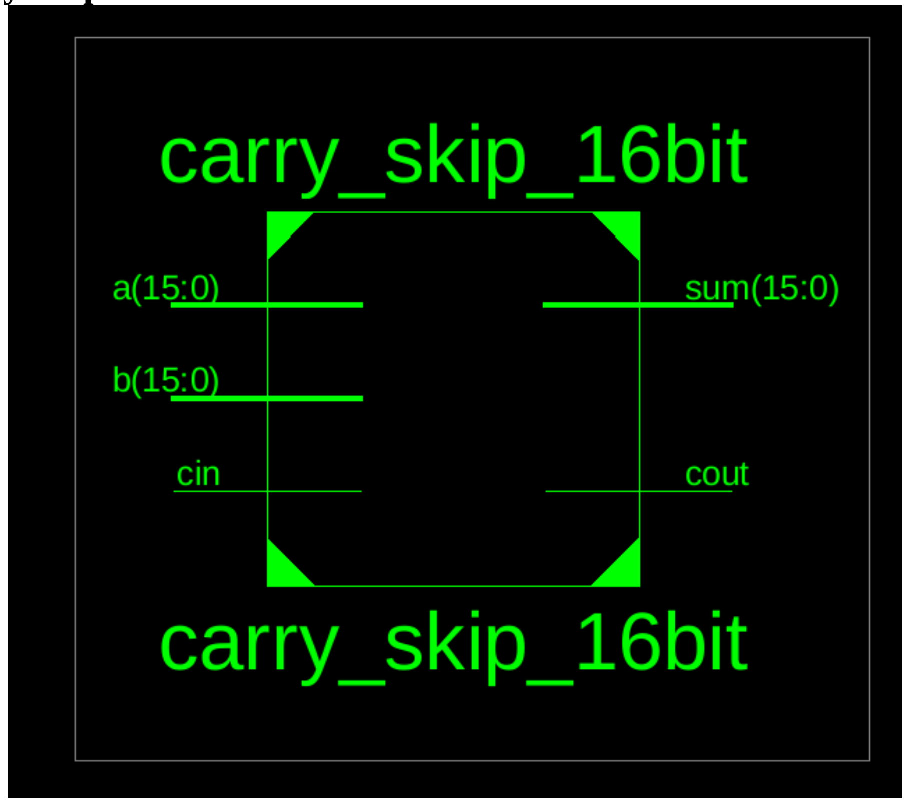


fig11: RTL Schematic

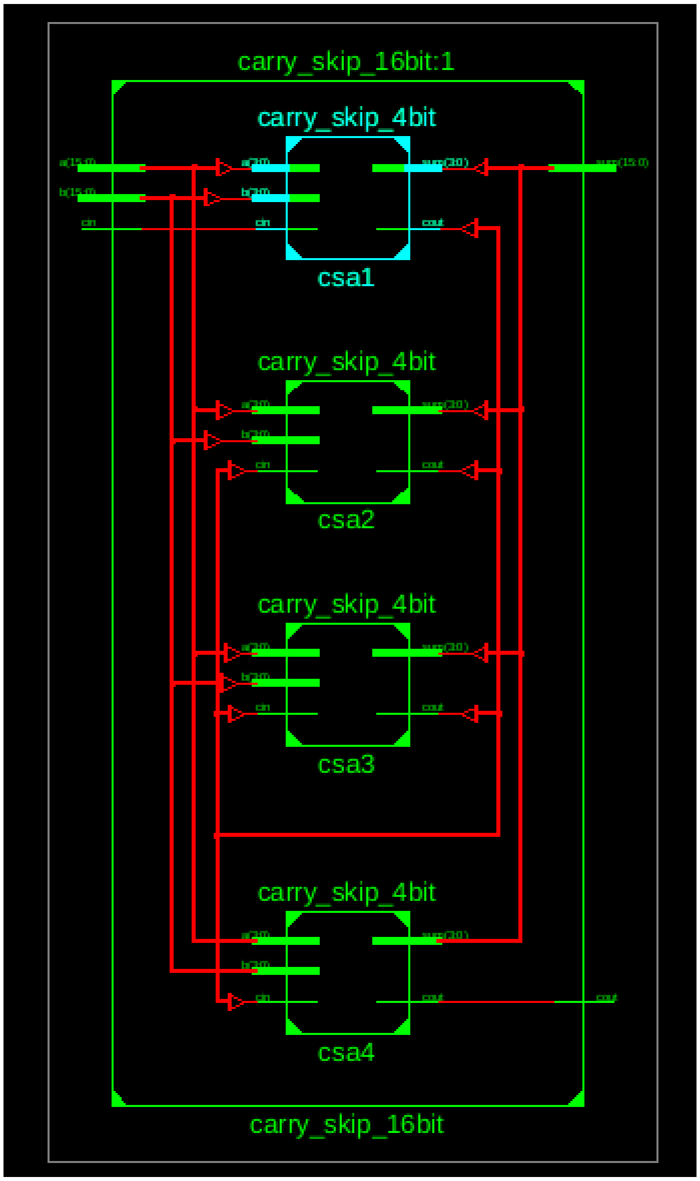


Fig11.1: RTL Schematic

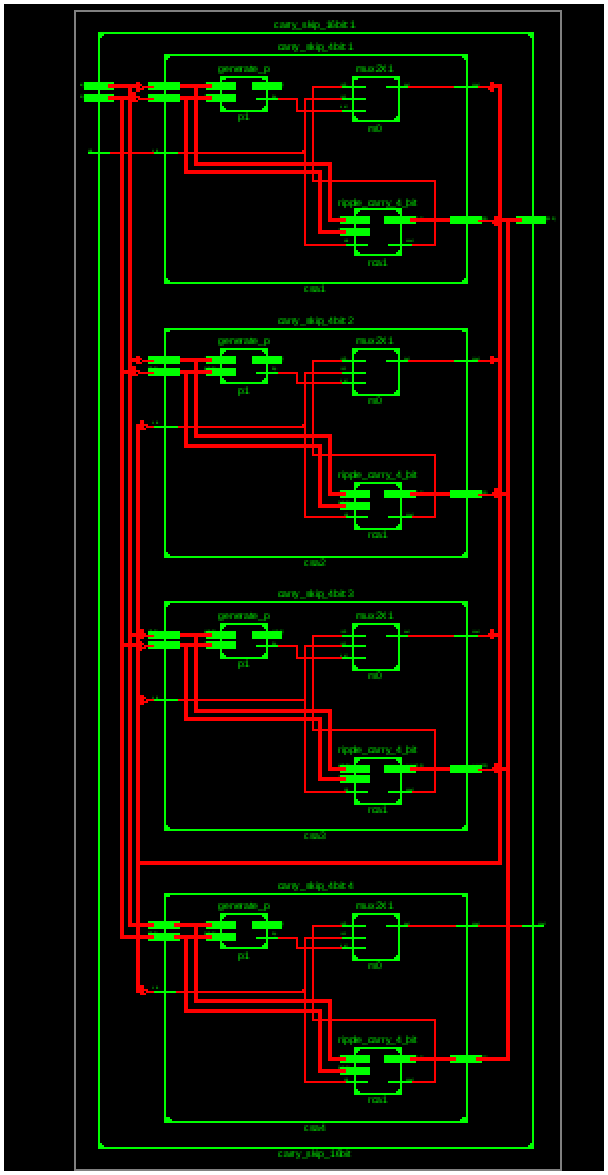
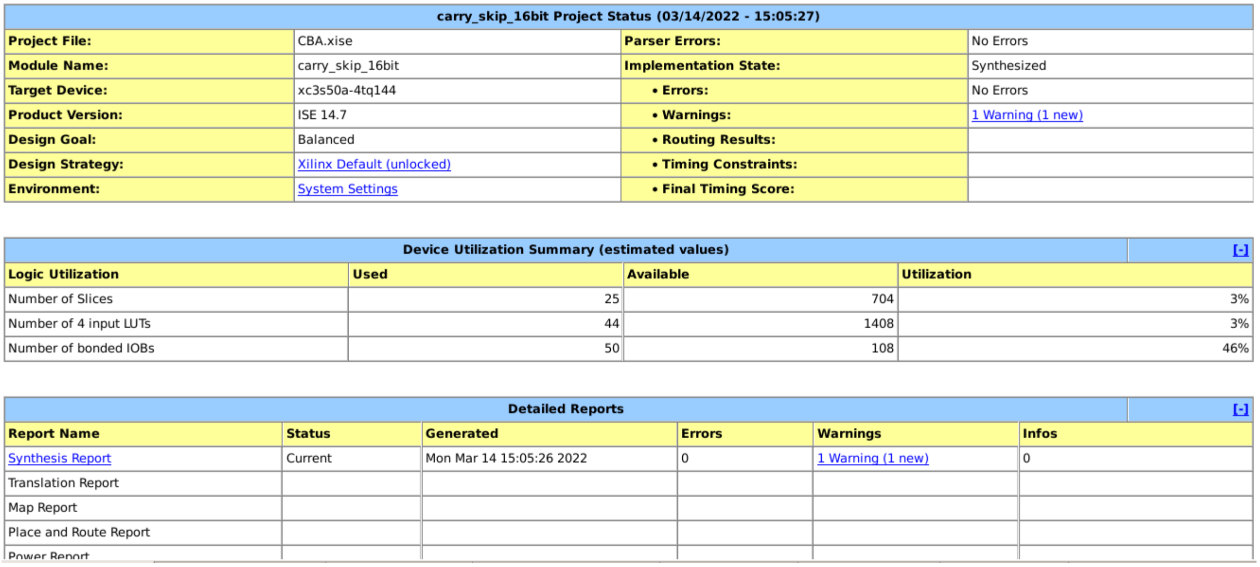


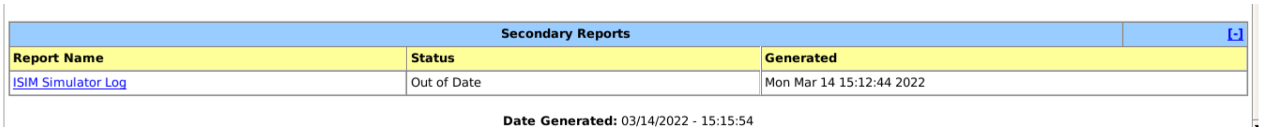
Fig12: Tech Schematic



carry_skip_16bit Project Status (03/14/2022 - 15:05:27)			
Project File:	CBA.xise	**Parser Errors:**	No Errors
Module Name:	carry_skip_16bit	**Implementation State:**	Synthesized
Target Device:	xc3s50a-4tq144	**• Errors:**	No Errors
Product Version:	ISE 14.7	**• Warnings:**	1 Warning (1 new)
Design Goal:	Balanced	**• Routing Results:**	
Design Strategy:	Xilinx Default (unlocked)	**• Timing Constraints:**	
Environment:	System Settings	**• Final Timing Score:**	

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	25	704	3%
Number of 4 input LUTs	44	1408	3%
Number of bonded IOBs	50	108	46%

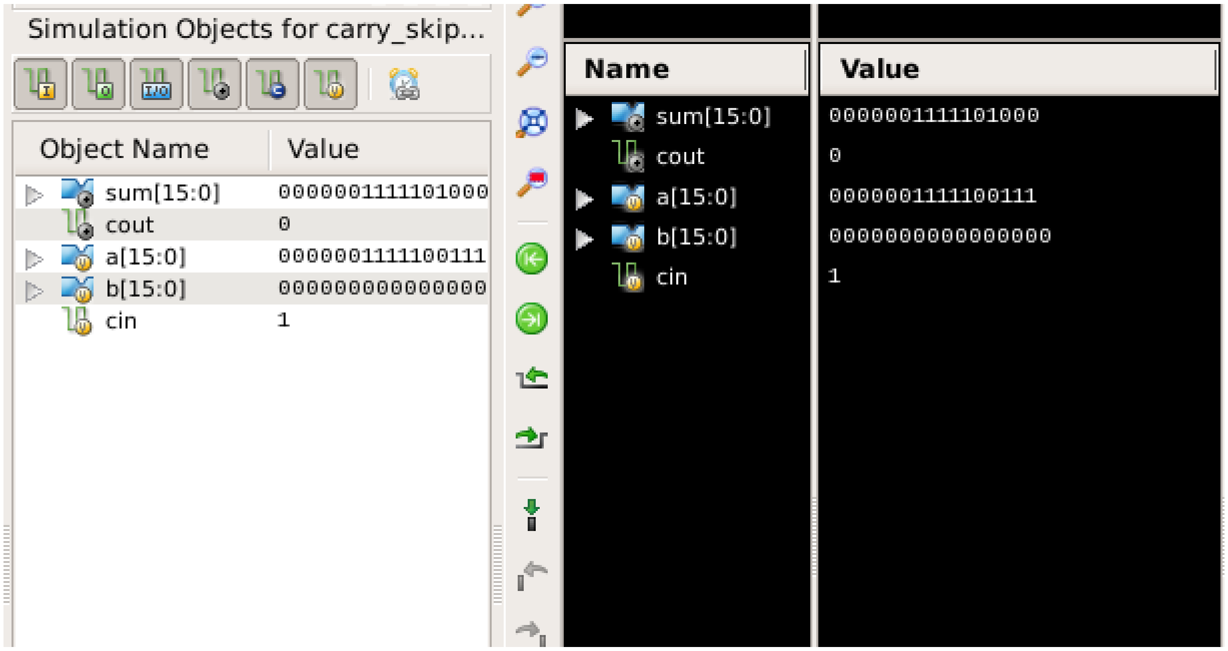
Detailed Reports					[-]
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	Mon Mar 14 15:05:26 2022	0	1 Warning (1 new)	0
Translation Report					
Map Report					
Place and Route Report					
Power Report					



Secondary Reports		[-]
Report Name	**Status**	**Generated**
ISIM Simulator Log	Out of Date	Mon Mar 14 15:12:44 2022

Date Generated: 03/14/2022 - 15:15:54

Fig13: Synthesize summary



Simulation Objects for carry_skip...

Object Name	Value
sum[15:0]	0000001111101000
cout	0
a[15:0]	0000001111100111
b[15:0]	0000000000000000
cin	1

Name	Value
sum[15:0]	0000001111101000
cout	0
a[15:0]	0000001111100111
b[15:0]	0000000000000000
cin	1

Fig14: Simulation

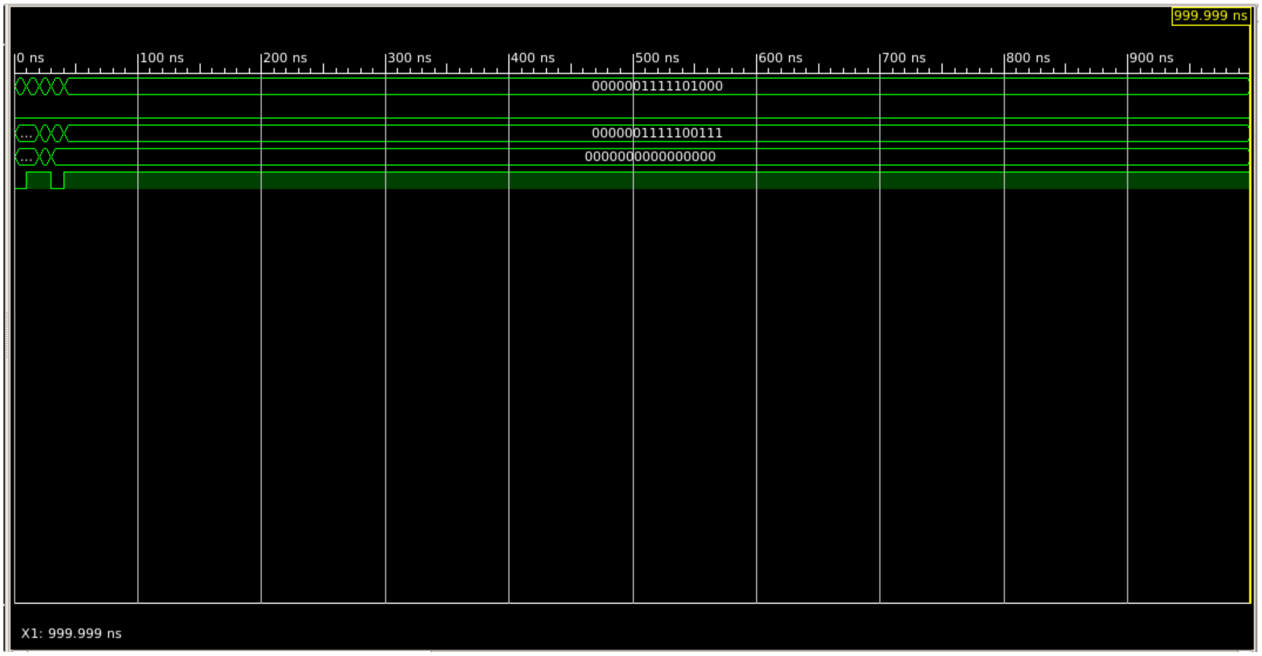


Fig14.1: Simulation

3. Ripple Carry Adder

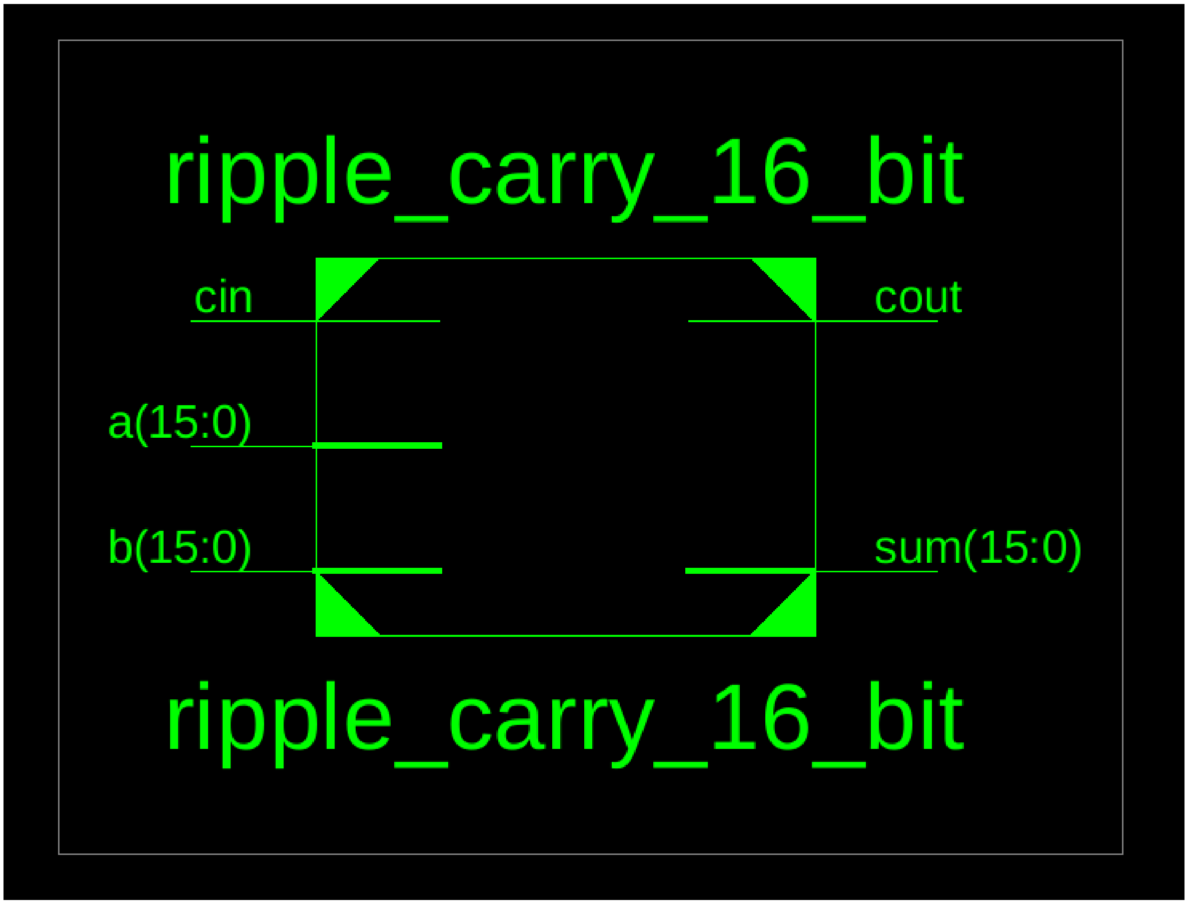


Fig15: RTL Schematic

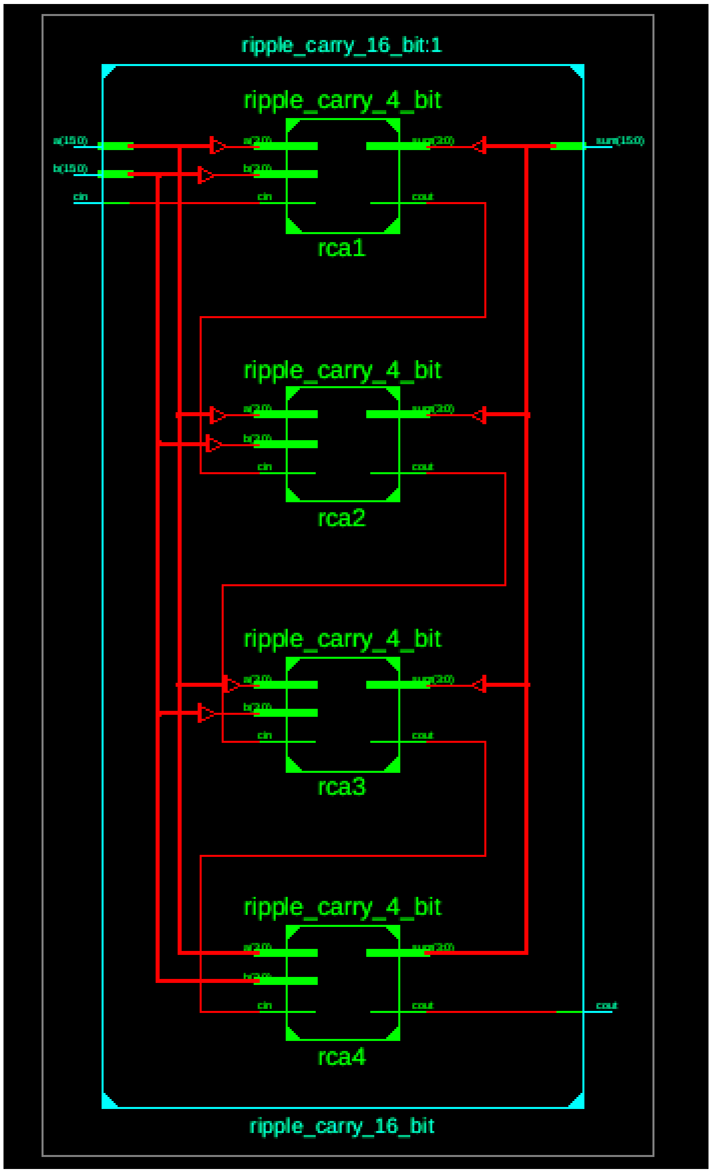


Fig15.1: RTL Schematic

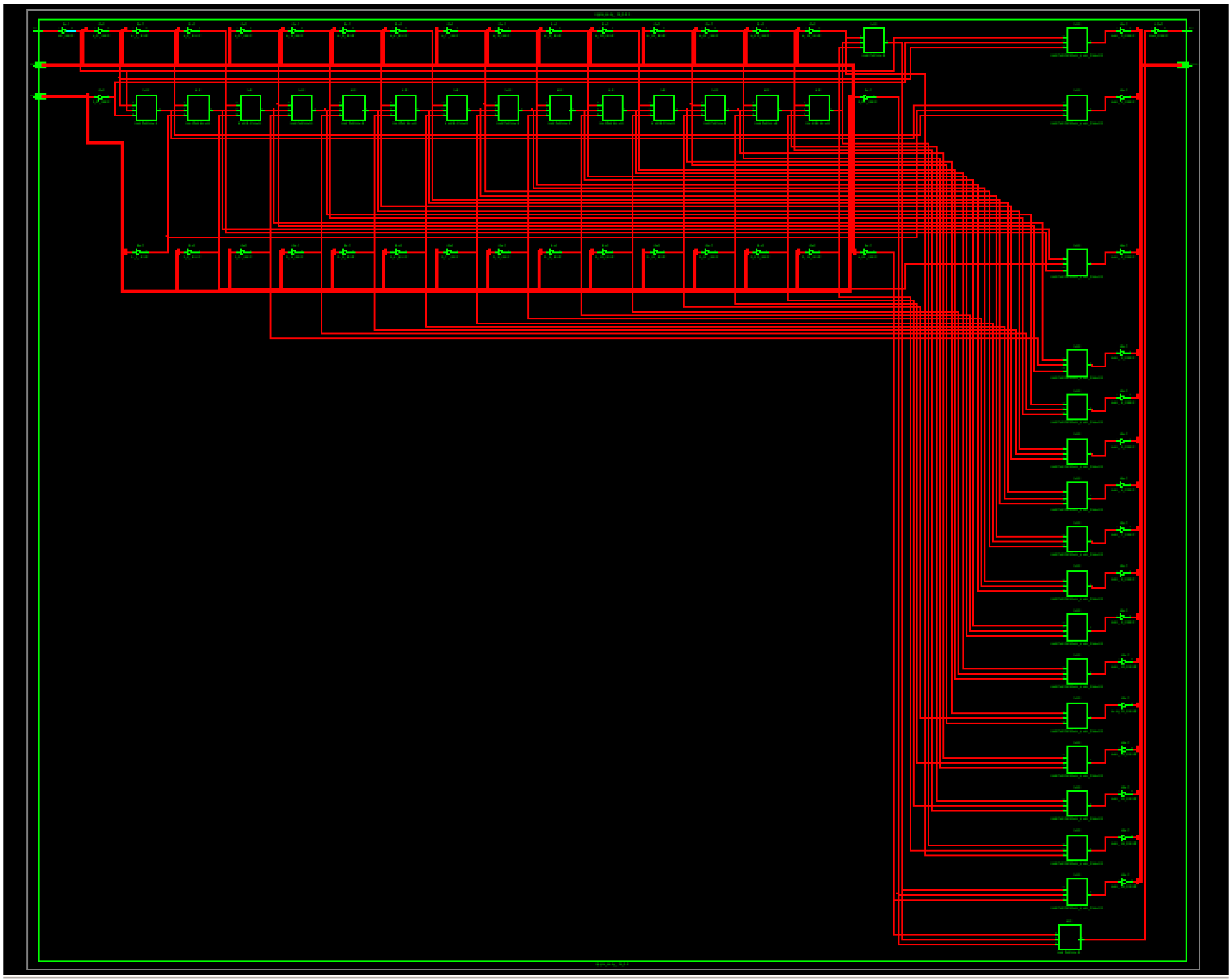
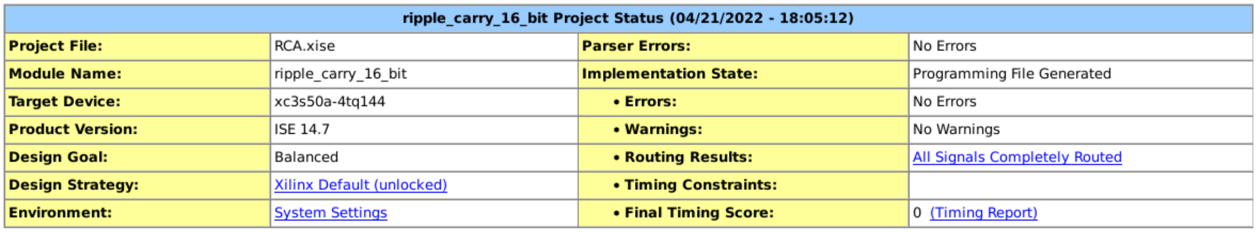
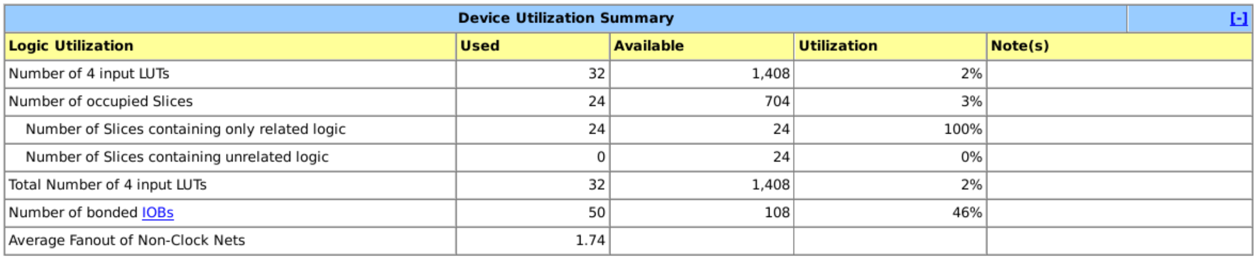


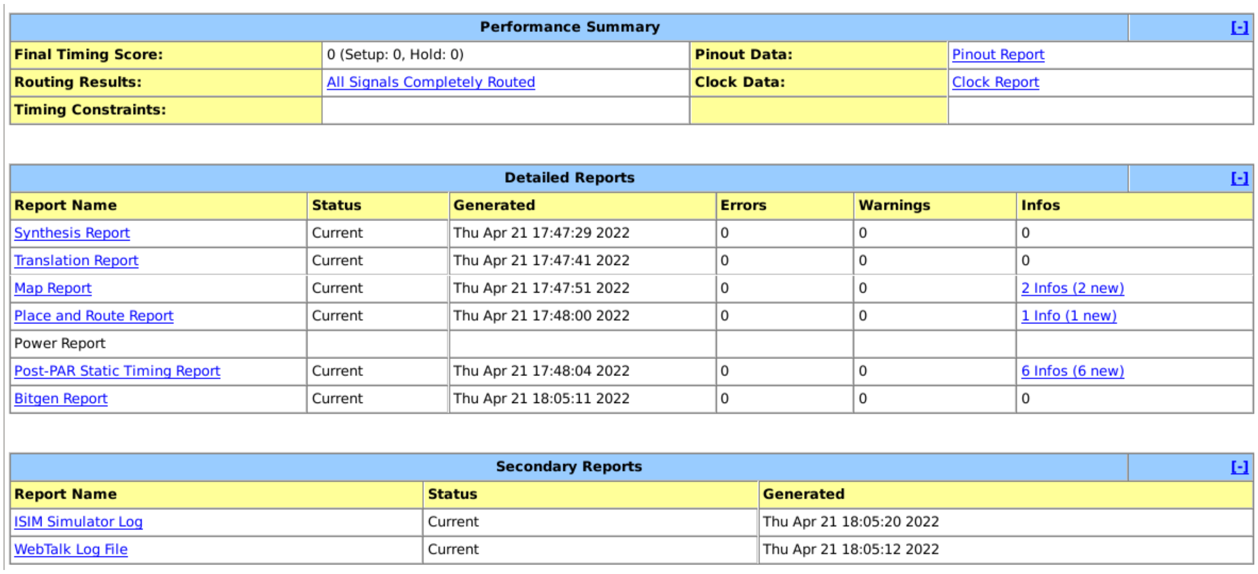
Fig16: Technology Schematic



ripple_carry_16_bit Project Status (04/21/2022 - 18:05:12)			
Project File:	RCA.xise	**Parser Errors:**	No Errors
Module Name:	ripple_carry_16_bit	**Implementation State:**	Programming File Generated
Target Device:	xc3s50a-4tq144	**• Errors:**	No Errors
Product Version:	ISE 14.7	**• Warnings:**	No Warnings
Design Goal:	Balanced	**• Routing Results:**	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	**• Timing Constraints:**	
Environment:	System Settings	**• Final Timing Score:**	0 (Timing Report)



Device Utilization Summary				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of 4 input LUTs	32	1,408	2%	
Number of occupied Slices	24	704	3%	
Number of Slices containing only related logic	24	24	100%	
Number of Slices containing unrelated logic	0	24	0%	
Total Number of 4 input LUTs	32	1,408	2%	
Number of bonded IOBs	50	108	46%	
Average Fanout of Non-Clock Nets	1.74			



Performance Summary [-]

Final Timing Score:	0 (Setup: 0, Hold: 0)	**Pinout Data:**	Pinout Report
Routing Results:	All Signals Completely Routed	**Clock Data:**	Clock Report
Timing Constraints:			

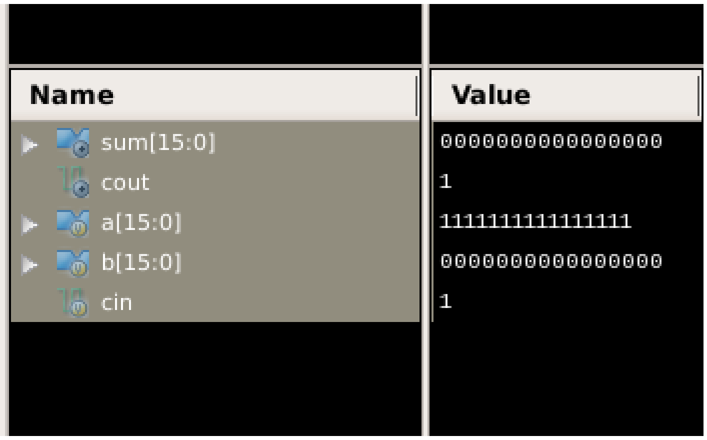
Detailed Reports [-]

Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	Thu Apr 21 17:47:29 2022	0	0	0
Translation Report	Current	Thu Apr 21 17:47:41 2022	0	0	0
Map Report	Current	Thu Apr 21 17:47:51 2022	0	0	2 Infos (2 new)
Place and Route Report	Current	Thu Apr 21 17:48:00 2022	0	0	1 Info (1 new)
Power Report					
Post-PAR Static Timing Report	Current	Thu Apr 21 17:48:04 2022	0	0	6 Infos (6 new)
Bitgen Report	Current	Thu Apr 21 18:05:11 2022	0	0	0

Secondary Reports [-]

Report Name	Status	Generated
ISIM Simulator Log	Current	Thu Apr 21 18:05:20 2022
WebTalk Log File	Current	Thu Apr 21 18:05:12 2022

Fig17: Synthesize summary



Name	Value
sum[15:0]	0000000000000000
cout	1
a[15:0]	1111111111111111
b[15:0]	0000000000000000
cin	1

Fig18: Simulation

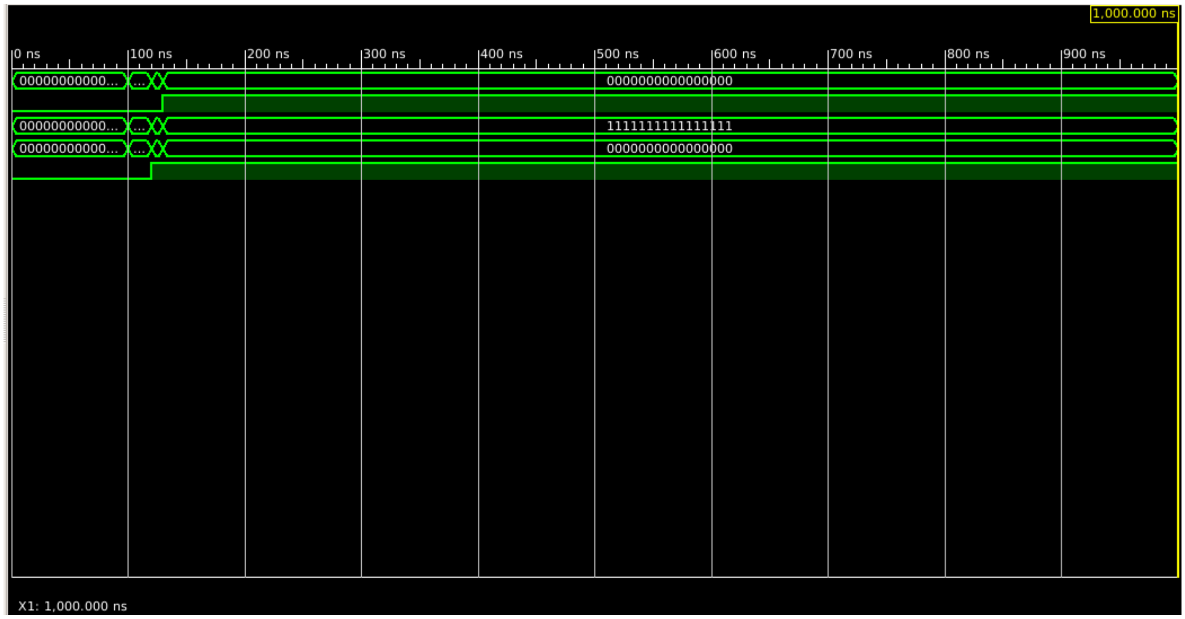


Fig18.1: Simulation

4. Carry Look Ahead Adder

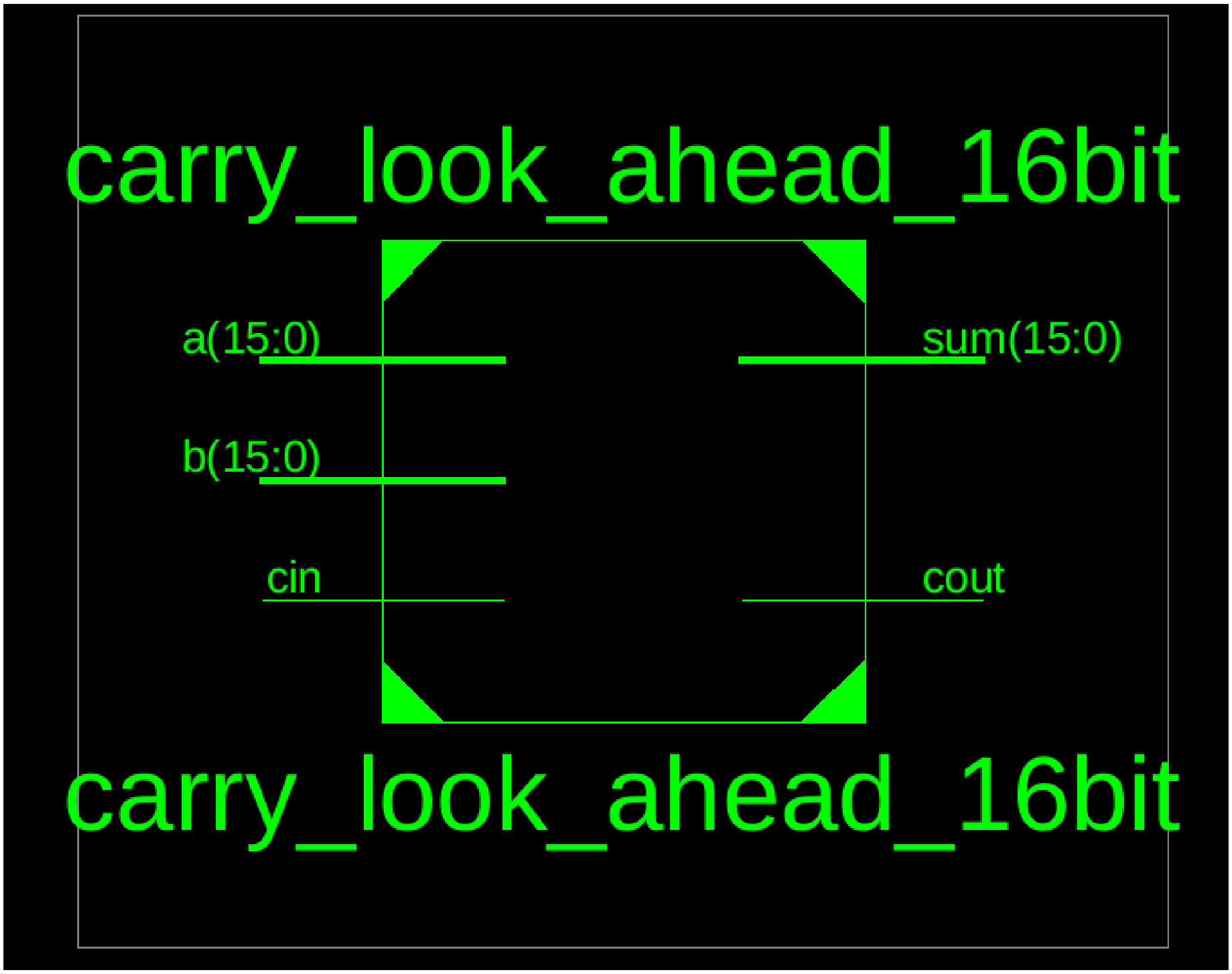


Fig19: RTL Schematic

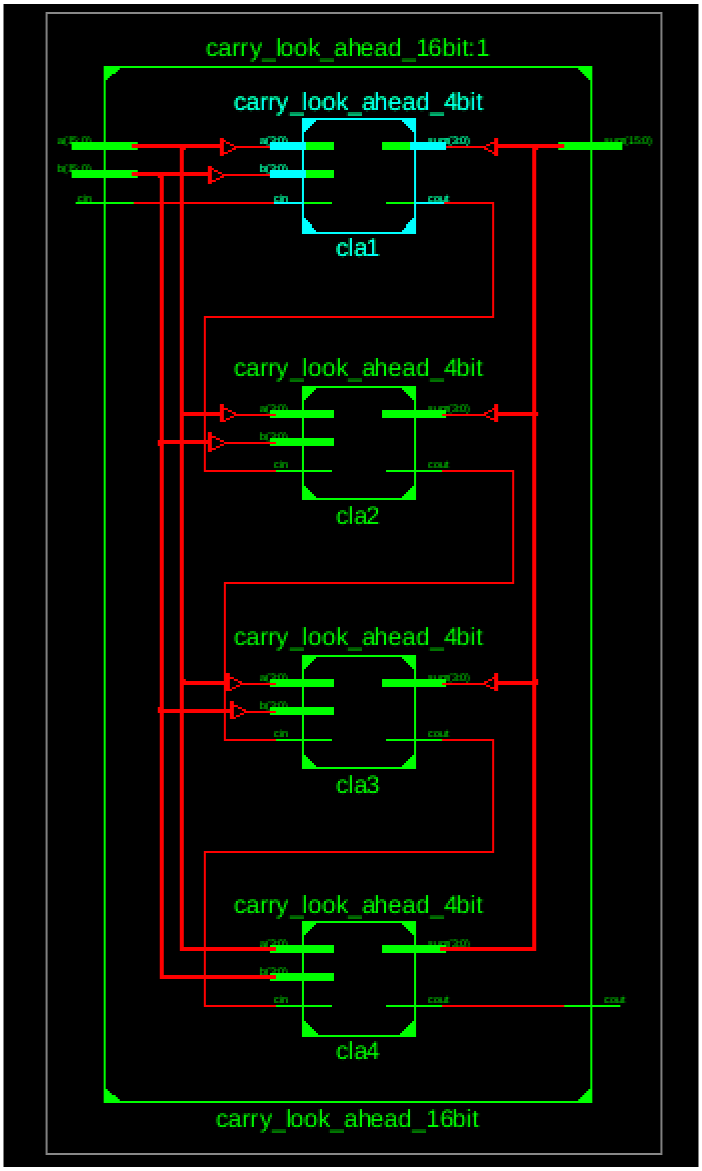


Fig19.1: RTL Schematic

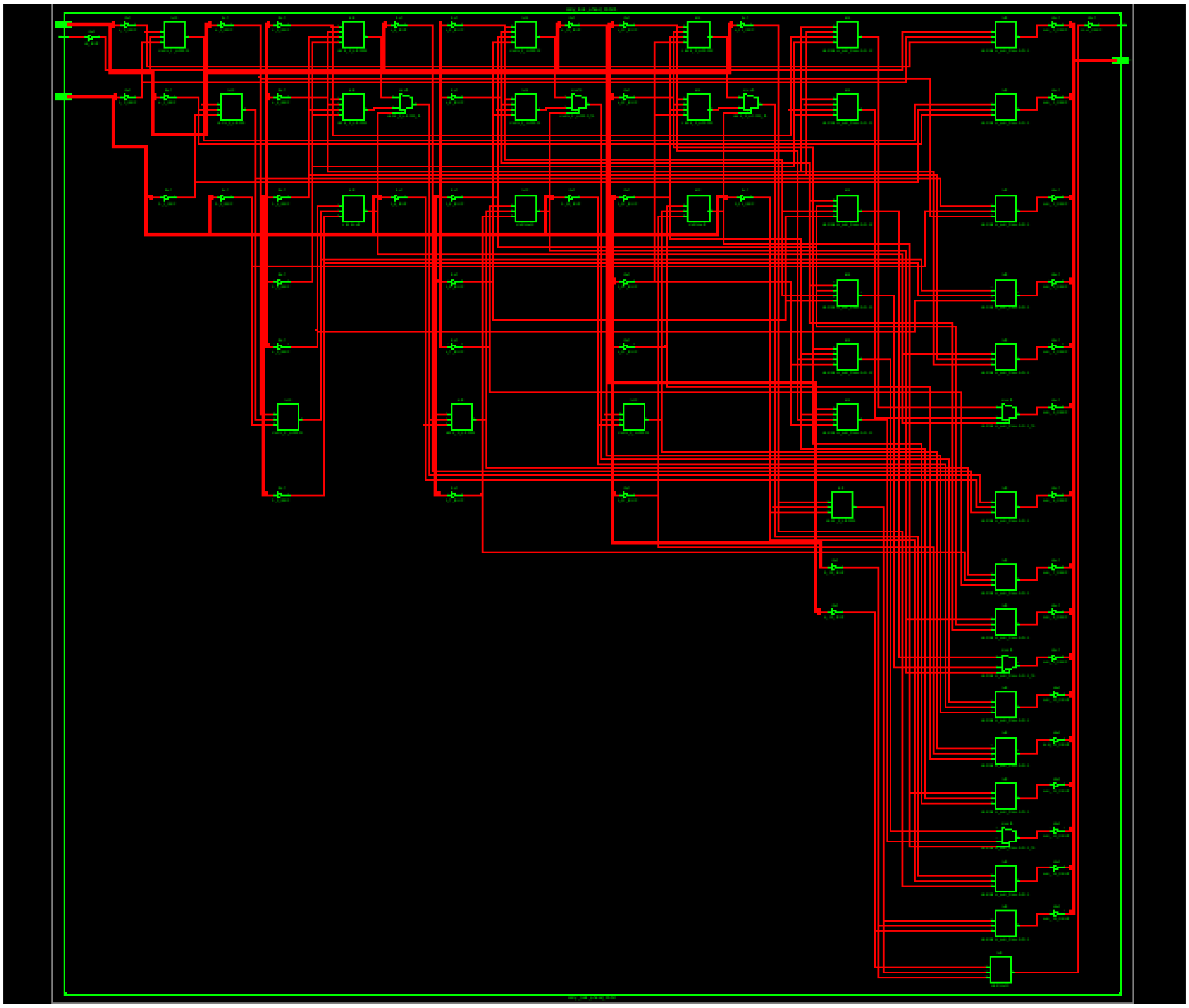
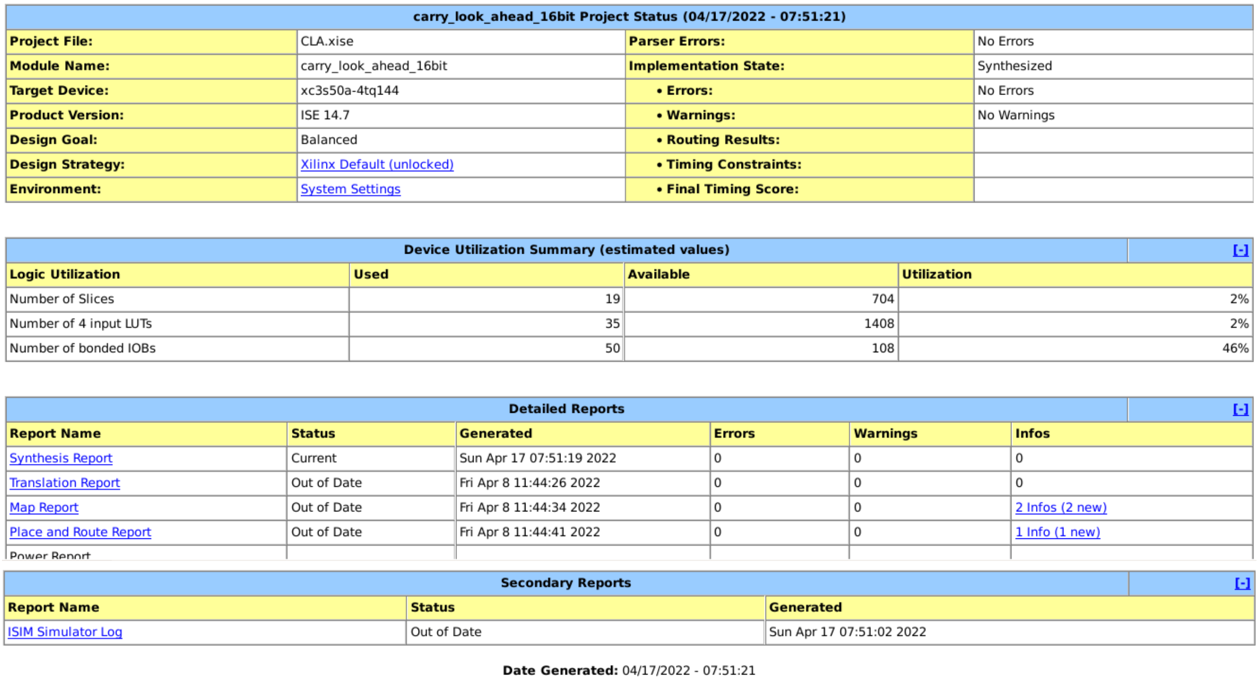


Fig20: Technology Schematic



carry_look_ahead_16bit Project Status (04/17/2022 - 07:51:21)			
Project File:	CLA.xise	**Parser Errors:**	No Errors
Module Name:	carry_look_ahead_16bit	**Implementation State:**	Synthesized
Target Device:	xc3s50a-4tq144	• **Errors:**	No Errors
Product Version:	ISE 14.7	• **Warnings:**	No Warnings
Design Goal:	Balanced	• **Routing Results:**	
Design Strategy:	Xilinx Default (unlocked)	• **Timing Constraints:**	
Environment:	System Settings	• **Final Timing Score:**	

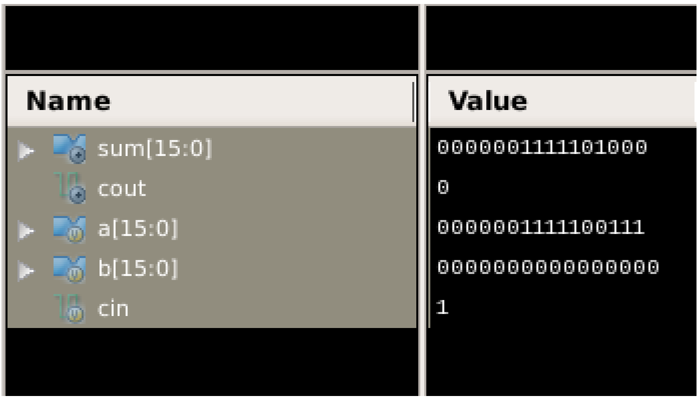
Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	19	704	2%
Number of 4 input LUTs	35	1408	2%
Number of bonded IOBs	50	108	46%

Detailed Reports					[-]
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	Sun Apr 17 07:51:19 2022	0	0	0
Translation Report	Out of Date	Fri Apr 8 11:44:26 2022	0	0	0
Map Report	Out of Date	Fri Apr 8 11:44:34 2022	0	0	2 Infos (2 new)
Place and Route Report	Out of Date	Fri Apr 8 11:44:41 2022	0	0	1 Info (1 new)
Power Report					

Secondary Reports		[-]
Report Name	**Status**	**Generated**
ISIM Simulator Log	Out of Date	Sun Apr 17 07:51:02 2022

Date Generated: 04/17/2022 - 07:51:21

Fig21: Synthesize summary



Name	Value
sum[15:0]	0000001111101000
cout	0
a[15:0]	0000001111100111
b[15:0]	0000000000000000
cin	1

Fig22: Simulation

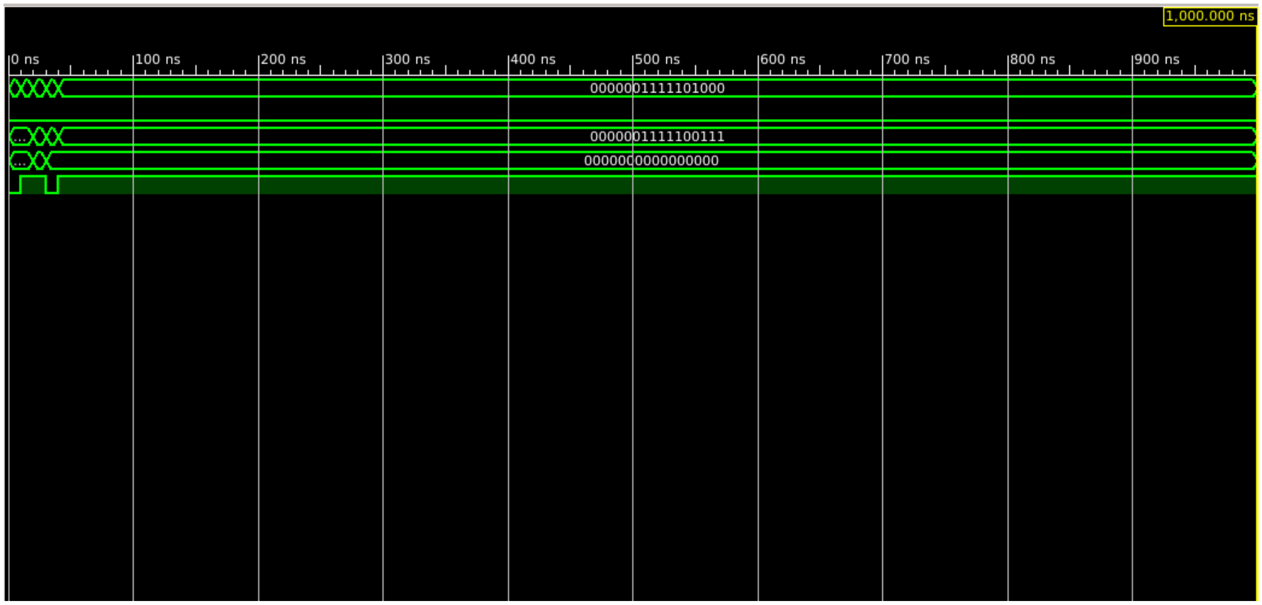
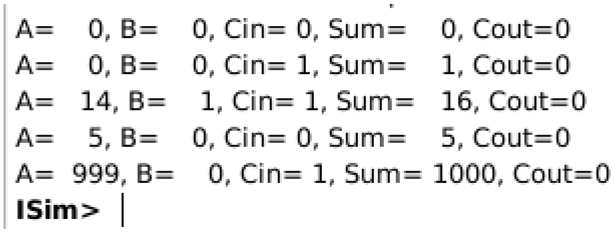


Fig23.1: Simulation



```
A=    0, B=    0, Cin= 0, Sum=    0, Cout=0
A=    0, B=    0, Cin= 1, Sum=    1, Cout=0
A=   14, B=    1, Cin= 1, Sum=   16, Cout=0
A=    5, B=    0, Cin= 0, Sum=    5, Cout=0
A=  999, B=    0, Cin= 1, Sum= 1000, Cout=0
ISim>
```

Fig23: Result

5.2 Hardware Result

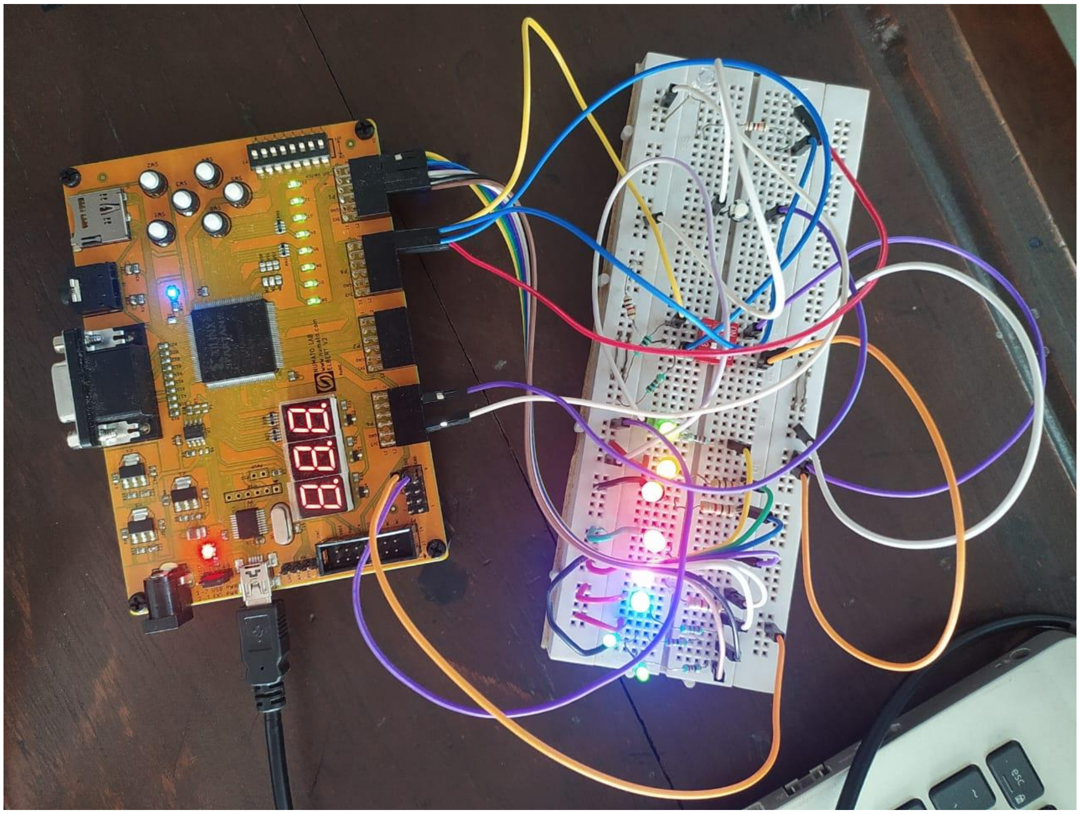


Fig24: Hardware

Case 1. a=1111111111111111

b=0000000000000000

cout=0

sum=1111111111111111

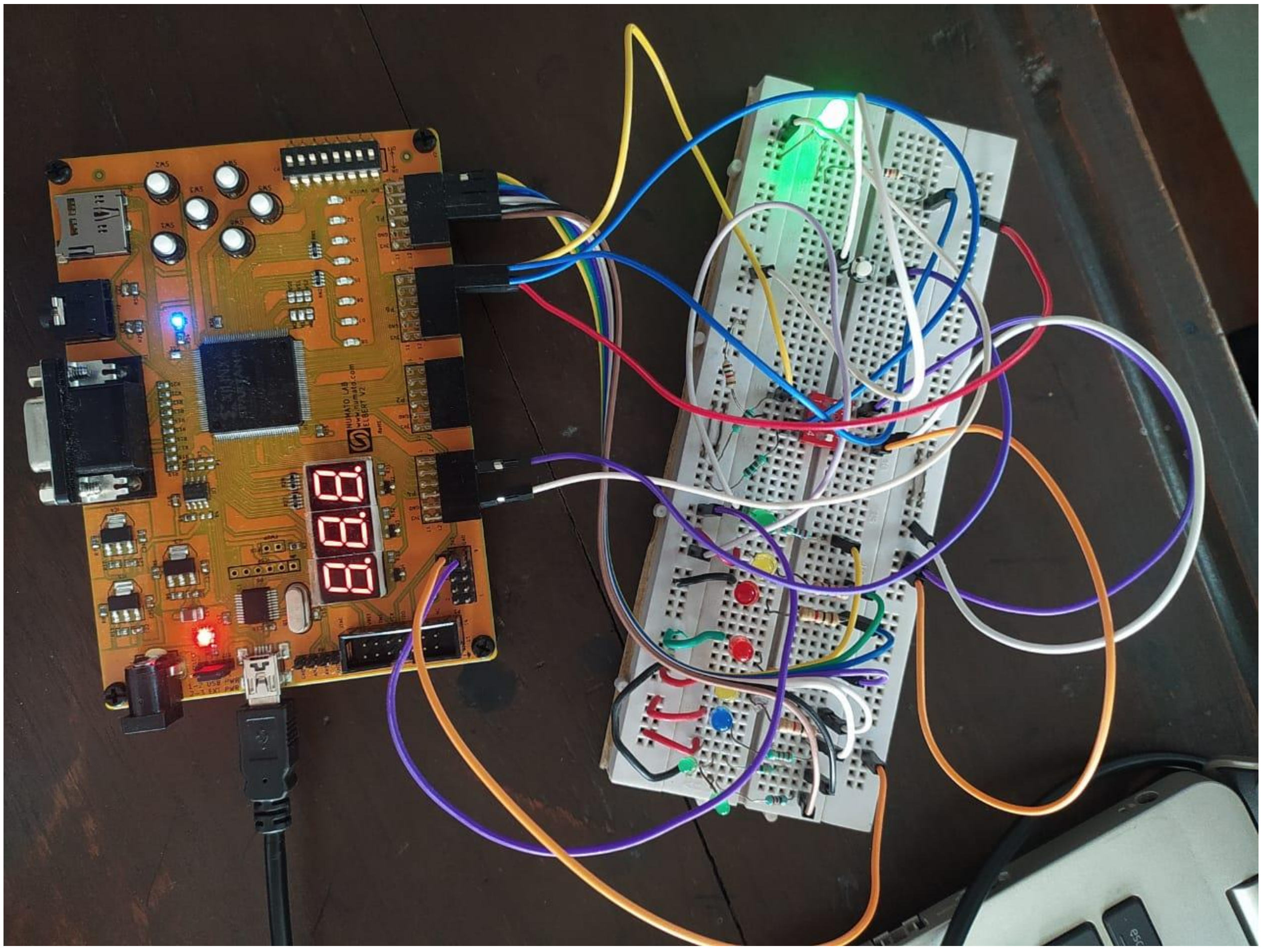


Fig25: Hardware

Case 2. a=1111111111111111

b=0000000000000001

cout=1

sum=0000000000000000

Chapter 6

Conclusion

S.No.	Design	Area (LUT's)	Area (Slices)	Delay (ns)
1.	Carry Select Adder	41	23	17.796ns
2.	Carry Skip Adder	44	25	29.313ns
3.	Ripple Carry Adder	32	18	24.577ns
4.	Carry Look Ahead Adder	35	19	21.375ns

Table 1: Comparison table

References

[1] Amala Maria Alex and Nidhish Antony, “Design of Modified Low Power and High Speed Carry Select Adder Using Brent Kung Adder”, International Journal of Scientific Development and Research, vol. 1, no. 8, pp. 196-204, 2017.

[2] Maroju SaiKumar and Dr. P. Samundiswary, “Design and Performance Analysis of Various Adders using Verilog ”, International Journal of Computer Science and Mobile Computing, vol. 2, no. 22, pp. 128-138, 2017.

[3] Shrikanth K. Shirakol, Ajaykumar S. Kulkarni, A.F.Akash, Nikhil N. Amminabhavi and Aditya Parvati, “Design and Implementation of 16-Bit Carry Bypass Adder using Efficient Low Power High Performance Full Adders”, Second International Conference on “Emerging Research in Computing Information, Communications and Applications”, vol. 61, no. 2, pp. 30–35, 2018.

[4] Sajesh Kumar, Mohamed Salih, “Efficient Carry Select Adder Design for FPGA Implementation”, International Conference on Communication Technology and System Design, vol. 67, no. 11, pp. 654–658, 2017.

[5] Yu Pang, Junchao Wang and Shaoquan Wang, “A 16-bit Carry Skip Adder Designed by Reversible Logic”, 5th International Conference on BioMedical Engineering and Informatics, pp. 1332-1335, 2016.

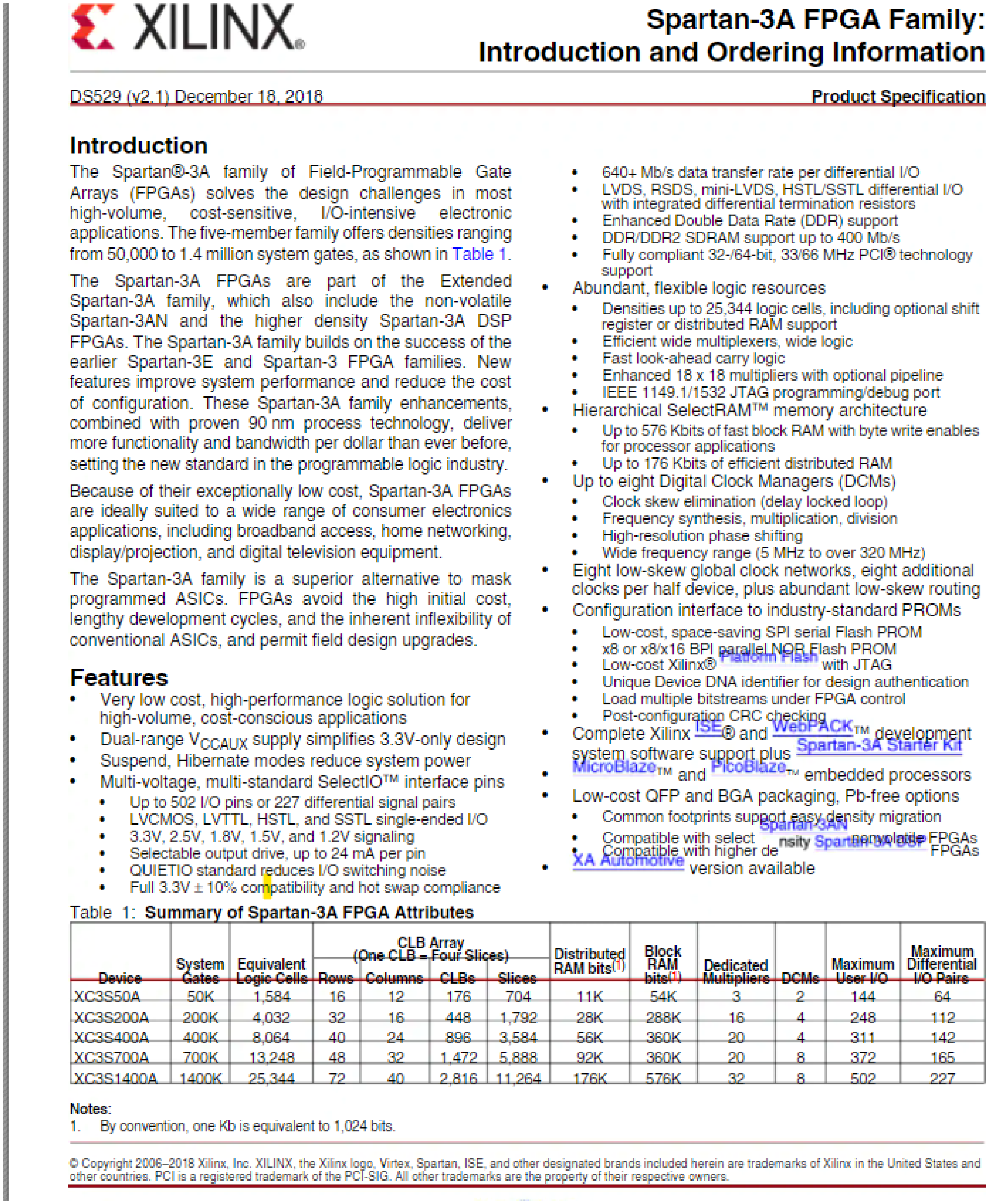
[6] P.Prashanti, Dr. B.Rajendra Naik,” Design and Implementation of High Speed Carry Select Adder”; International Journal of Engineering Trends and Technology (IJETT) – Vol. 4, pp.453-459, 2016

[7] Shivani Parmar and Kirat Pal Singh," Design of High Speed Hybrid Carry Select Adder", IEEE's 3rd International Advance Computing Conference (IACC) Ghaziabad, pp. 22-23, 2017.

[8] B. Ramkumar and Harish M Kittur, “Low-Power and Area-Efficient Carry Select Adder”, IEEE Trans. on very large scale integration systems , vol 4, no.15 pp. 55-59, 2015.

[9] Arora A. and Niranjan V, “A new 16-bit high speed and variable stage carry skip adder”, 3rd International Conference on Computational Intelligence & Communication Technology, pp. 14-19, 2017.

[10] Singh I. and Dhingra M. , “Design and Implementation of 32-Bits Carry Skip Adder using CMOS Logic in International Journal of VLSI System Design and Communication Systems”, vol. 03, pp. 1116-1112, 2018.



XILINX

Spartan-3A FPGA Family: Introduction and Ordering Information

DS529 (v2.1) December 18, 2018 **Product Specification**

Introduction

The Spartan®-3A family of Field-Programmable Gate Arrays (FPGAs) solves the design challenges in most high-volume, cost-sensitive, I/O-intensive electronic applications. The five-member family offers densities ranging from 50,000 to 1.4 million system gates, as shown in Table 1.

The Spartan-3A FPGAs are part of the Extended Spartan-3A family, which also include the non-volatile Spartan-3AN and the higher density Spartan-3A DSP FPGAs. The Spartan-3A family builds on the success of the earlier Spartan-3E and Spartan-3 FPGA families. New features improve system performance and reduce the cost of configuration. These Spartan-3A family enhancements, combined with proven 90 nm process technology, deliver more functionality and bandwidth per dollar than ever before, setting the new standard in the programmable logic industry.

Because of their exceptionally low cost, Spartan-3A FPGAs are ideally suited to a wide range of consumer electronics applications, including broadband access, home networking, display/projection, and digital television equipment.

The Spartan-3A family is a superior alternative to mask programmed ASICs. FPGAs avoid the high initial cost, lengthy development cycles, and the inherent inflexibility of conventional ASICs, and permit field design upgrades.

Features

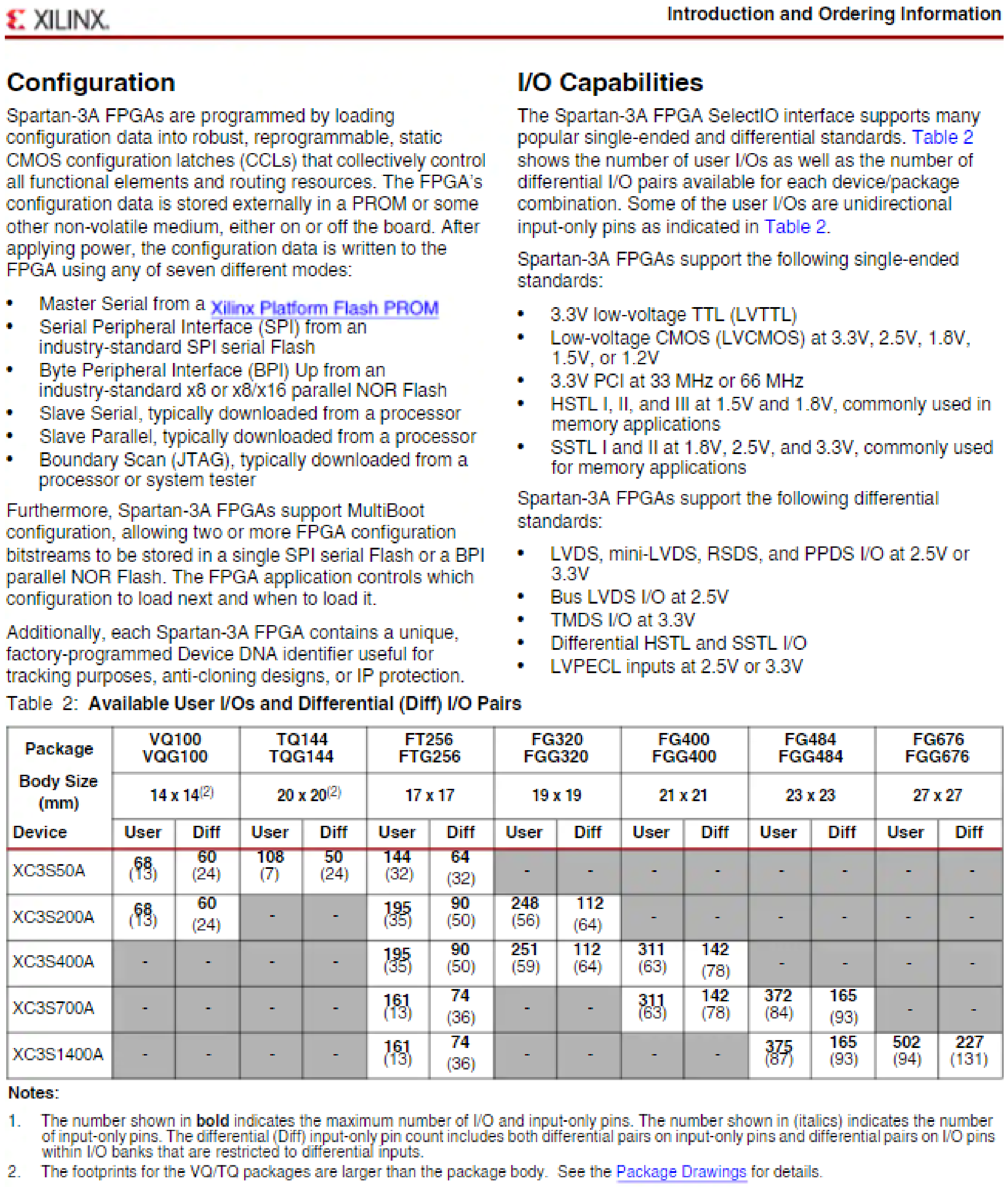
- Very low cost, high-performance logic solution for high-volume, cost-conscious applications
- Dual-range V_{CCAUX} supply simplifies 3.3V-only design
- Suspend, Hibernate modes reduce system power
- Multi-voltage, multi-standard SelectIO™ interface pins
  - Up to 502 I/O pins or 227 differential signal pairs
  - LVCMOS, LVTTL, HSTL, and SSTL single-ended I/O
  - 3.3V, 2.5V, 1.8V, 1.5V, and 1.2V signaling
  - Selectable output drive, up to 24 mA per pin
  - QUIETIO standard reduces I/O switching noise
  - Full 3.3V ± 10% compatibility and hot swap compliance
  - 640+ Mb/s data transfer rate per differential I/O
  - LVDS, RSDS, mini-LVDS, HSTL/SSTL differential I/O with integrated differential termination resistors
  - Enhanced Double Data Rate (DDR) support
  - DDR/DDR2 SDRAM support up to 400 Mb/s
  - Fully compliant 32-/64-bit, 33/66 MHz PCI® technology support
- Abundant, flexible logic resources
  - Densities up to 25,344 logic cells, including optional shift register or distributed RAM support
  - Efficient wide multiplexers, wide logic
  - Fast look-ahead carry logic
  - Enhanced 18 x 18 multipliers with optional pipeline
  - IEEE 1149.1/1532 JTAG programming/debug port
- Hierarchical SelectRAM™ memory architecture
  - Up to 576 Kbits of fast block RAM with byte write enables for processor applications
  - Up to 176 Kbits of efficient distributed RAM
- Up to eight Digital Clock Managers (DCMs)
  - Clock skew elimination (delay locked loop)
  - Frequency synthesis, multiplication, division
  - High-resolution phase shifting
  - Wide frequency range (5 MHz to over 320 MHz)
- Eight low-skew global clock networks, eight additional clocks per half device, plus abundant low-skew routing
- Configuration interface to industry-standard PROMs
  - Low-cost, space-saving SPI serial Flash PROM
  - x8 or x8/x16 BPI parallel NOR Flash PROM
  - Low-cost Xilinx® Platform Flash with JTAG
  - Unique Device DNA identifier for design authentication
  - Load multiple bitstreams under FPGA control
  - Post-configuration CRC checking
- Complete Xilinx ISE® and WebPACK™ development system software support plus Spartan-3A Starter Kit
- MicroBlaze™ and PicoBlaze™ embedded processors
- Low-cost QFP and BGA packaging, Pb-free options
  - Common footprints support easy density migration
  - Compatible with select Spartan-3AN nonvolatile FPGAs
  - Compatible with higher density Spartan-3A DSP FPGAs
- XA Automotive version available

Table 1: **Summary of Spartan-3A FPGA Attributes**

Device	System Gates	Equivalent Logic Cells	CLB Array (One CLB = Four Slices)				Distributed RAM bits[1]	Block RAM bits[1]	Dedicated Multipliers	DCMs	Maximum User I/O	Maximum Differential I/O Pairs
			Rows	Columns	CLBs	Slices						
XC3S50A	50K	1,584	16	12	176	704	11K	54K	3	2	144	64
XC3S200A	200K	4,032	32	16	448	1,792	28K	288K	16	4	248	112
XC3S400A	400K	8,064	40	24	896	3,584	56K	360K	20	4	311	142
XC3S700A	700K	13,248	48	32	1,472	5,888	92K	360K	20	8	372	165
XC3S1400A	1400K	25,344	72	40	2,816	11,264	176K	576K	32	8	502	227

Notes:

1. By convention, one Kb is equivalent to 1,024 bits.



Configuration

Spartan-3A FPGAs are programmed by loading configuration data into robust, reprogrammable, static CMOS configuration latches (CCLs) that collectively control all functional elements and routing resources. The FPGA's configuration data is stored externally in a PROM or some other non-volatile medium, either on or off the board. After applying power, the configuration data is written to the FPGA using any of seven different modes:

- Master Serial from a Xilinx Platform Flash PROM
- Serial Peripheral Interface (SPI) from an industry-standard SPI serial Flash
- Byte Peripheral Interface (BPI) Up from an industry-standard x8 or x8/x16 parallel NOR Flash
- Slave Serial, typically downloaded from a processor
- Slave Parallel, typically downloaded from a processor
- Boundary Scan (JTAG), typically downloaded from a processor or system tester

Furthermore, Spartan-3A FPGAs support MultiBoot configuration, allowing two or more FPGA configuration bitstreams to be stored in a single SPI serial Flash or a BPI parallel NOR Flash. The FPGA application controls which configuration to load next and when to load it.

Additionally, each Spartan-3A FPGA contains a unique, factory-programmed Device DNA identifier useful for tracking purposes, anti-cloning designs, or IP protection.

I/O Capabilities

The Spartan-3A FPGA SelectIO interface supports many popular single-ended and differential standards. Table 2 shows the number of user I/Os as well as the number of differential I/O pairs available for each device/package combination. Some of the user I/Os are unidirectional input-only pins as indicated in Table 2.

Spartan-3A FPGAs support the following single-ended standards:

- 3.3V low-voltage TTL (LVTTL)
- Low-voltage CMOS (LVCMOS) at 3.3V, 2.5V, 1.8V, 1.5V, or 1.2V
- 3.3V PCI at 33 MHz or 66 MHz
- HSTL I, II, and III at 1.5V and 1.8V, commonly used in memory applications
- SSTL I and II at 1.8V, 2.5V, and 3.3V, commonly used for memory applications

Spartan-3A FPGAs support the following differential standards:

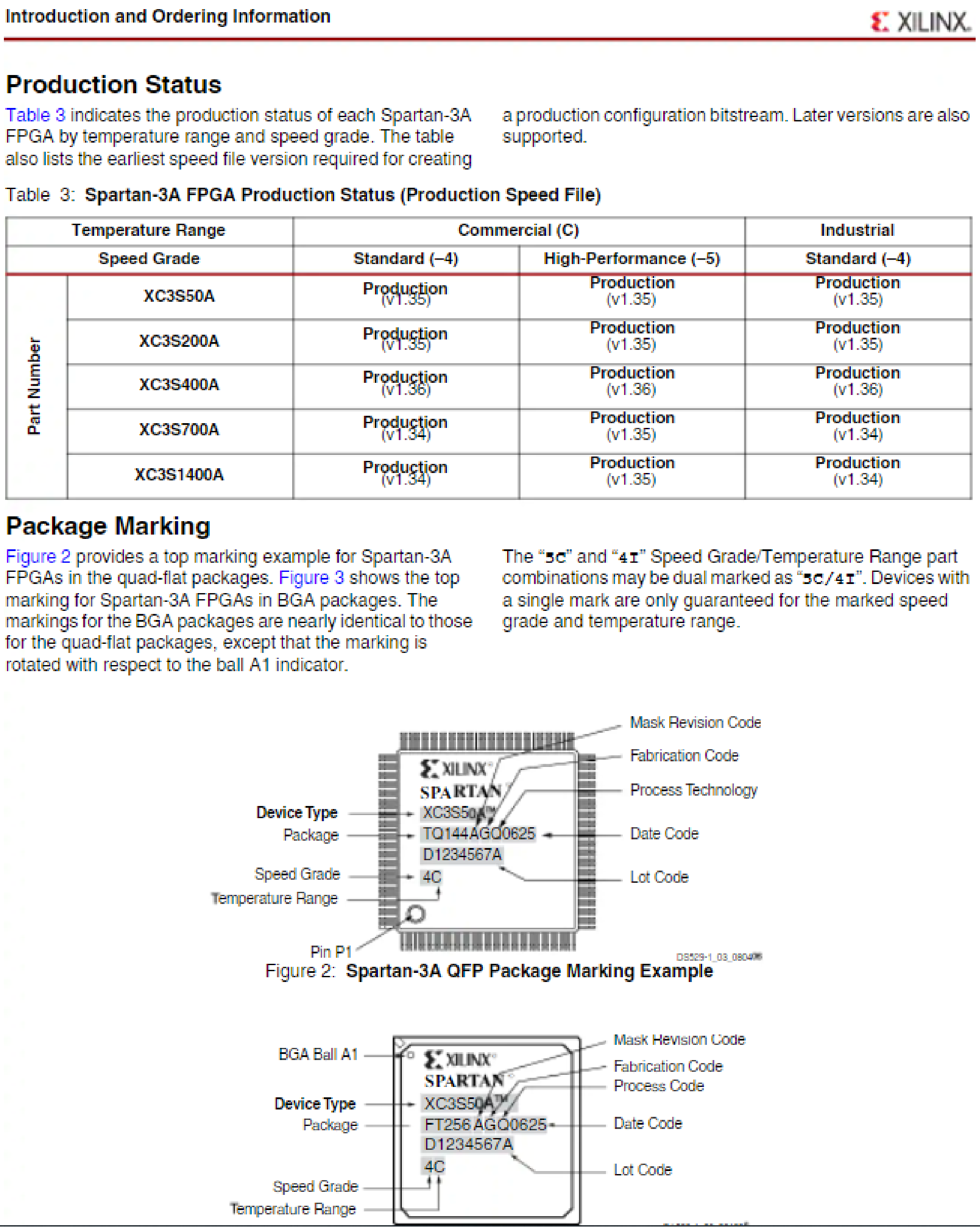
- LVDS, mini-LVDS, RSDS, and PPDS I/O at 2.5V or 3.3V
- Bus LVDS I/O at 2.5V
- TMDS I/O at 3.3V
- Differential HSTL and SSTL I/O
- LVPECL inputs at 2.5V or 3.3V

Table 2: **Available User I/Os and Differential (Diff) I/O Pairs**

Package	VQ100 VQG100		TQ144 TQG144		FT256 FTG256		FG320 FGG320		FG400 FGG400		FG484 FGG484		FG676 FGG676	
Body Size (mm)	14 x 14[2]		20 x 20[2]		17 x 17		19 x 19		21 x 21		23 x 23		27 x 27	
Device	User	Diff	User	Diff	User	Diff	User	Diff	User	Diff	User	Diff	User	Diff
XC3S50A	**68** *(13)*	**60** *(24)*	**108** *(7)*	**50** *(24)*	**144** *(32)*	**64** *(32)*	-	-	-	-	-	-	-	-
XC3S200A	**68** *(13)*	**60** *(24)*	-	-	**195** *(35)*	**90** *(50)*	**248** *(56)*	**112** *(64)*	-	-	-	-	-	-
XC3S400A	-	-	-	-	**195** *(35)*	**90** *(50)*	**251** *(59)*	**112** *(64)*	**311** *(63)*	**142** *(78)*	-	-	-	-
XC3S700A	-	-	-	-	**161** *(13)*	**74** *(36)*	-	-	**311** *(63)*	**142** *(78)*	**372** *(84)*	**165** *(93)*	-	-
XC3S1400A	-	-	-	-	**161** *(13)*	**74** *(36)*	-	-	-	-	**375** *(87)*	**165** *(93)*	**502** *(94)*	**227** *(131)*

Notes:

1. The number shown in **bold** indicates the maximum number of I/O and input-only pins. The number shown in (italics) indicates the number of input-only pins. The differential (Diff) input-only pin count includes both differential pairs on input-only pins and differential pairs on I/O pins within I/O banks that are restricted to differential inputs.
2. The footprints for the VQ/TQ packages are larger than the package body. See the Package Drawings for details.



Production Status

Table 3 indicates the production status of each Spartan-3A FPGA by temperature range and speed grade. The table also lists the earliest speed file version required for creating a production configuration bitstream. Later versions are also supported.

Table 3: **Spartan-3A FPGA Production Status (Production Speed File)**

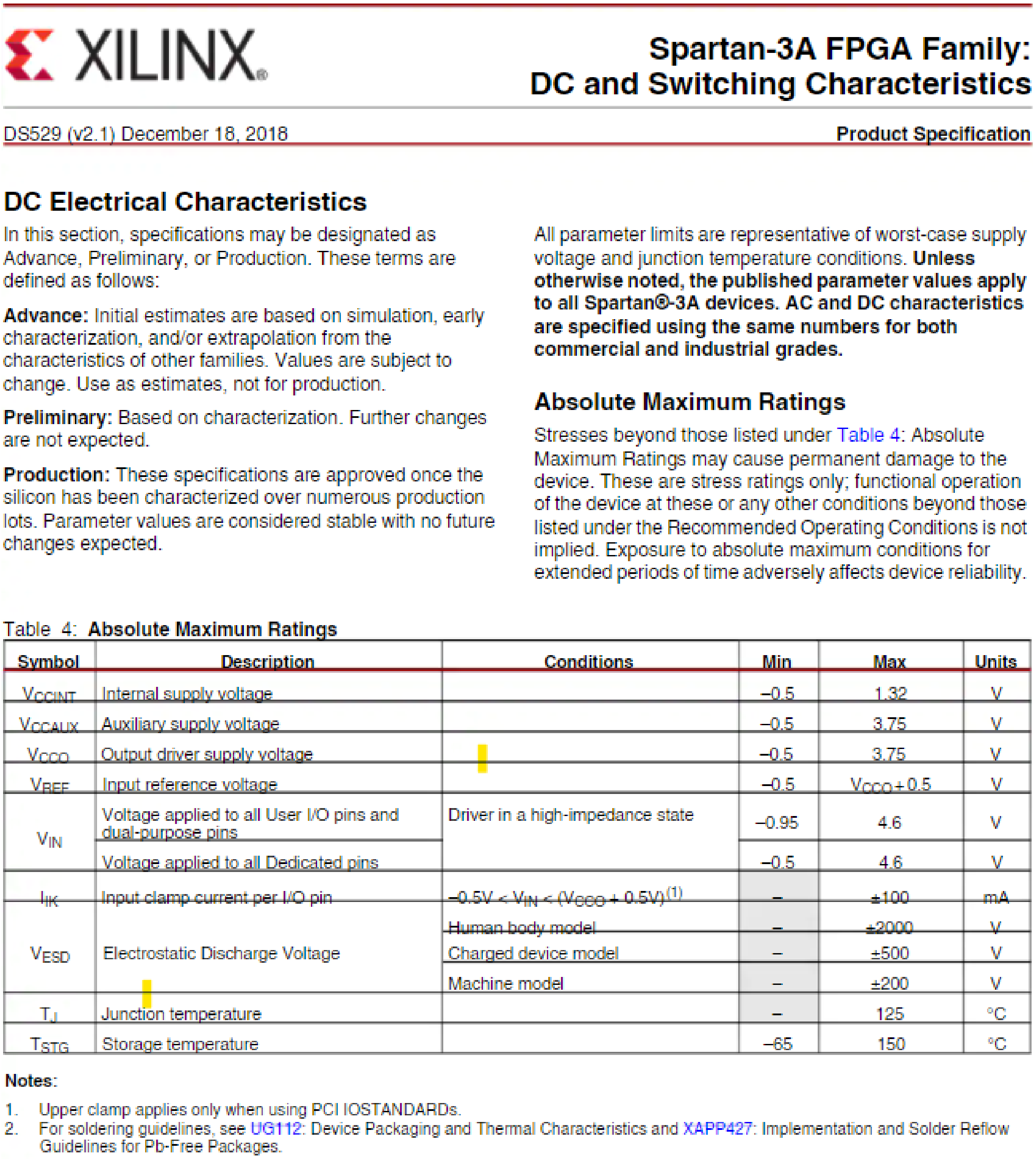
Temperature Range		Commercial (C)		Industrial
Speed Grade		Standard (–4)	High-Performance (–5)	Standard (–4)
Part Number	XC3S50A	Production (v1.35)	Production (v1.35)	Production (v1.35)
	XC3S200A	Production (v1.35)	Production (v1.35)	Production (v1.35)
	XC3S400A	Production (v1.36)	Production (v1.36)	Production (v1.36)
	XC3S700A	Production (v1.34)	Production (v1.35)	Production (v1.34)
	XC3S1400A	Production (v1.34)	Production (v1.35)	Production (v1.34)

Package Marking

Figure 2 provides a top marking example for Spartan-3A FPGAs in the quad-flat packages. Figure 3 shows the top marking for Spartan-3A FPGAs in BGA packages. The markings for the BGA packages are nearly identical to those for the quad-flat packages, except that the marking is rotated with respect to the ball A1 indicator.

The "5C" and "4I" Speed Grade/Temperature Range part combinations may be dual marked as "5C/4I". Devices with a single mark are only guaranteed for the marked speed grade and temperature range.

Figure 2: **Spartan-3A QFP Package Marking Example**



Spartan-3A FPGA Family: DC and Switching Characteristics

DS529 (v2.1) December 18, 2018

Product Specification

DC Electrical Characteristics

In this section, specifications may be designated as Advance, Preliminary, or Production. These terms are defined as follows:

Advance: Initial estimates are based on simulation, early characterization, and/or extrapolation from the characteristics of other families. Values are subject to change. Use as estimates, not for production.

Preliminary: Based on characterization. Further changes are not expected.

Production: These specifications are approved once the silicon has been characterized over numerous production lots. Parameter values are considered stable with no future changes expected.

All parameter limits are representative of worst-case supply voltage and junction temperature conditions. **Unless otherwise noted, the published parameter values apply to all Spartan®-3A devices. AC and DC characteristics are specified using the same numbers for both commercial and industrial grades.**

Absolute Maximum Ratings

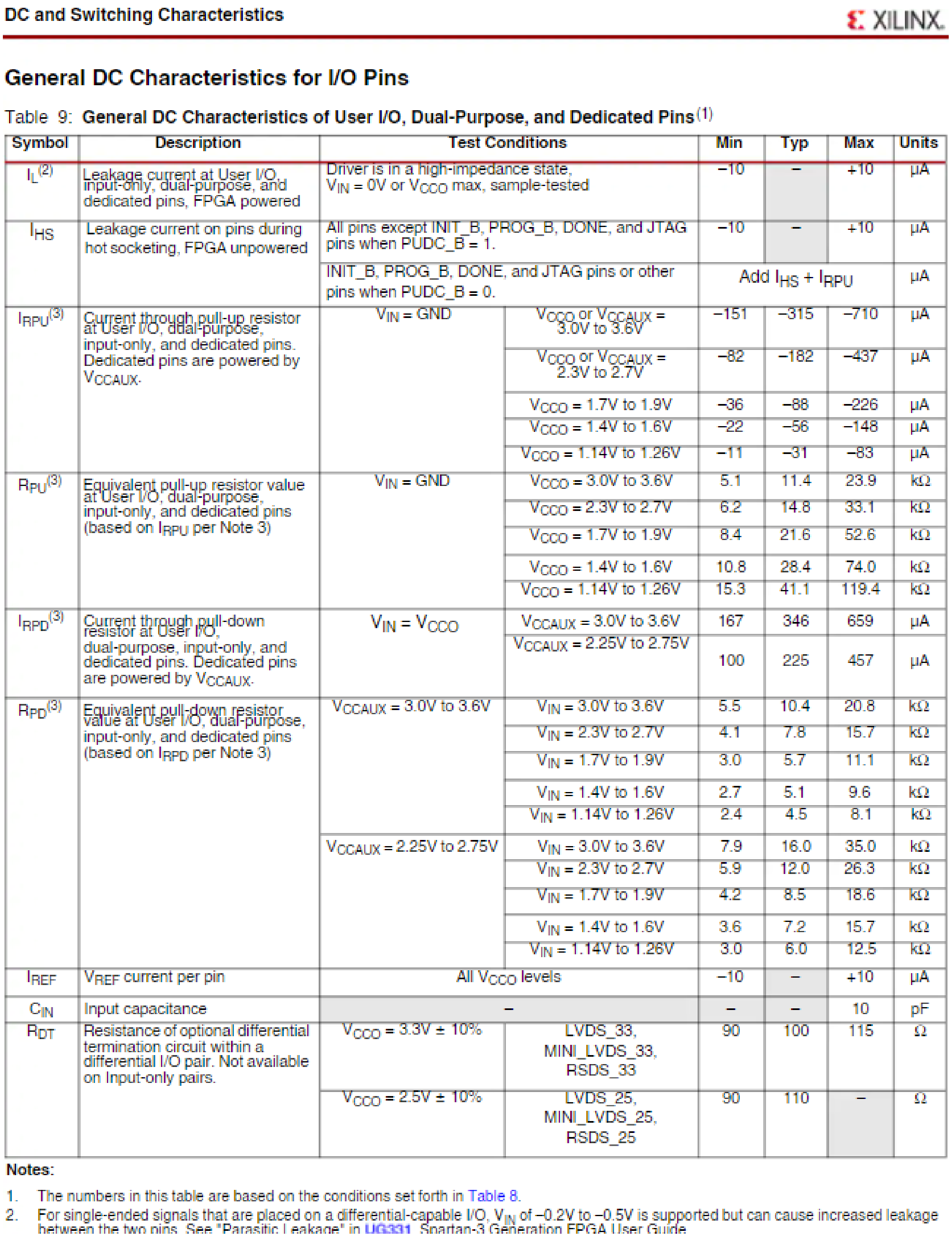
Stresses beyond those listed under Table 4: Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only; functional operation of the device at these or any other conditions beyond those listed under the Recommended Operating Conditions is not implied. Exposure to absolute maximum conditions for extended periods of time adversely affects device reliability.

Table 4: **Absolute Maximum Ratings**

Symbol	Description	Conditions	Min	Max	Units
V_{CCINT}	Internal supply voltage		–0.5	1.32	V
V_{CCAUX}	Auxiliary supply voltage		–0.5	3.75	V
V_{CCO}	Output driver supply voltage		–0.5	3.75	V
V_{REF}	Input reference voltage		–0.5	$V_{CCO}+0.5$	V
V_{IN}	Voltage applied to all User I/O pins and dual-purpose pins	Driver in a high-impedance state	–0.95	4.6	V
	Voltage applied to all Dedicated pins		–0.5	4.6	V
I_{IK}	Input clamp current per I/O pin	$-0.5V < V_{IN} < (V_{CCO} + 0.5V)$(1)	–	±100	mA
V_{ESD}	Electrostatic Discharge Voltage	Human body model	–	±2000	V
		Charged device model	–	±500	V
		Machine model	–	±200	V
T_J	Junction temperature		–	125	°C
T_{STG}	Storage temperature		–65	150	°C

Notes:

1. Upper clamp applies only when using PCI IOSTANDARDs.
2. For soldering guidelines, see UG112: Device Packaging and Thermal Characteristics and XAPP427: Implementation and Solder Reflow Guidelines for Pb-Free Packages.



DC and Switching Characteristics

General DC Characteristics for I/O Pins

Table 9: **General DC Characteristics of User I/O, Dual-Purpose, and Dedicated Pins**[1]

Symbol	Description	Test Conditions		Min	Typ	Max	Units
I_L[2]	Leakage current at User I/O, input-only, dual-purpose, and dedicated pins, FPGA powered	Driver is in a high-impedance state, V_{IN} = 0V or V_{CCO} max, sample-tested		−10	–	+10	µA
I_{HS}	Leakage current on pins during hot socketing, FPGA unpowered	All pins except INIT_B, PROG_B, DONE, and JTAG pins when PUDC_B = 1.		−10	–	+10	µA
		INIT_B, PROG_B, DONE, and JTAG pins or other pins when PUDC_B = 0.		Add I_{HS} + I_{RPU}			µA
I_{RPU}[3]	Current through pull-up resistor at User I/O, dual-purpose, input-only, and dedicated pins. Dedicated pins are powered by V_{CCAUX}.	V_{IN} = GND	V_{CCO} or V_{CCAUX} = 3.0V to 3.6V	−151	−315	−710	µA
			V_{CCO} or V_{CCAUX} = 2.3V to 2.7V	−82	−182	−437	µA
			V_{CCO} = 1.7V to 1.9V	−36	−88	−226	µA
			V_{CCO} = 1.4V to 1.6V	−22	−56	−148	µA
			V_{CCO} = 1.14V to 1.26V	−11	−31	−83	µA
R_{PU}[3]	Equivalent pull-up resistor value at User I/O, dual-purpose, input-only, and dedicated pins (based on I_{RPU} per Note 3)	V_{IN} = GND	V_{CCO} = 3.0V to 3.6V	5.1	11.4	23.9	kΩ
			V_{CCO} = 2.3V to 2.7V	6.2	14.8	33.1	kΩ
			V_{CCO} = 1.7V to 1.9V	8.4	21.6	52.6	kΩ
			V_{CCO} = 1.4V to 1.6V	10.8	28.4	74.0	kΩ
			V_{CCO} = 1.14V to 1.26V	15.3	41.1	119.4	kΩ
I_{RPD}[3]	Current through pull-down resistor at User I/O, dual-purpose, input-only, and dedicated pins. Dedicated pins are powered by V_{CCAUX}.	V_{IN} = V_{CCO}	V_{CCAUX} = 3.0V to 3.6V	167	346	659	µA
			V_{CCAUX} = 2.25V to 2.75V	100	225	457	µA
R_{PD}[3]	Equivalent pull-down resistor value at User I/O, dual-purpose, input-only, and dedicated pins (based on I_{RPD} per Note 3)	V_{CCAUX} = 3.0V to 3.6V	V_{IN} = 3.0V to 3.6V	5.5	10.4	20.8	kΩ
			V_{IN} = 2.3V to 2.7V	4.1	7.8	15.7	kΩ
			V_{IN} = 1.7V to 1.9V	3.0	5.7	11.1	kΩ
			V_{IN} = 1.4V to 1.6V	2.7	5.1	9.6	kΩ
			V_{IN} = 1.14V to 1.26V	2.4	4.5	8.1	kΩ
		V_{CCAUX} = 2.25V to 2.75V	V_{IN} = 3.0V to 3.6V	7.9	16.0	35.0	kΩ
			V_{IN} = 2.3V to 2.7V	5.9	12.0	26.3	kΩ
			V_{IN} = 1.7V to 1.9V	4.2	8.5	18.6	kΩ
			V_{IN} = 1.4V to 1.6V	3.6	7.2	15.7	kΩ
			V_{IN} = 1.14V to 1.26V	3.0	6.0	12.5	kΩ
I_{REF}	V_{REF} current per pin	All V_{CCO} levels		−10	–	+10	µA
C_{IN}	Input capacitance	–		–	–	10	pF
R_{DT}	Resistance of optional differential termination circuit within a differential I/O pair. Not available on Input-only pairs.	V_{CCO} = 3.3V ± 10%	LVDS_33, MINI_LVDS_33, RSDS_33	90	100	115	Ω
		V_{CCO} = 2.5V ± 10%	LVDS_25, MINI_LVDS_25, RSDS_25	90	110	–	Ω

Notes:

1. The numbers in this table are based on the conditions set forth in Table 8.
2. For single-ended signals that are placed on a differential-capable I/O, V_{IN} of −0.2V to −0.5V is supported but can cause increased leakage between the two pins. See "Parasitic Leakage" in UG331, Spartan-3 Generation FPGA User Guide.