Custom chip design using Altera Quartus II

.

By Kuntal Patel.

This project report is an introduction to the emerging concept of FPGA custom chip design, layout planning and related tools. I have designed 4-bit Adder using Altera Quartus II. Quartus II software delivers the highest productivity for programmable logic and structured ASIC designs. The Time Quest timing analyzer is the first timing analysis tool from an FPGA vendor to provide comprehensive native support for the industry-standard Synopsys Design Constraint (SDC) timing format. This ASIC strength tool enables users to create, manage, and analyze designs with complex timing constraints, such as clock multiplexed designs and source synchronous interfaces, and to quickly perform advanced timing verification. In addition, it includes an expanded team-based design environment with a project manager interface for managing resource and timing budgets at the top-level design.

1. FPGA: WHAT IS IT AND HOW IT IS DIFFERENT THAN OTHER TECHNOLOGIES?

Field Programmable Gate Arrays (FPGAs) are semiconductor devices that are based around a matrix of configurable logic blocks (CLBs) connected via programmable interconnects. FPGAs can be reprogrammed to desired application or functionality requirements after manufacturing. This feature distinguishes FPGAs from Application Specific Integrated Circuits (ASICs), which are custom manufactured for specific design tasks. Although one-time programmable (OTP) FPGAs are available, the dominant types are SRAM based which can be reprogrammed as the design evolves.

ASIC and FPGAs have different value propositions, and they must be carefully evaluated before choosing any one over the other. Information abounds that compares the two technologies. While FPGAs used to be selected for lower speed/complexity/volume designs in the past, today's FPGAs easily push the 500 MHz performance barrier. With unprecedented logic density increases and a host of other features, such as embedded processors, DSP blocks, clocking, and high-speed serial at ever lower price points, FPGAs are a compelling proposition for almost any type of design.

Due to their programmable nature, FPGAs are an ideal fit for many different markets. As the industry leader, Xilinx provides comprehensive solutions consisting of FPGA devices, advanced software, and configurable, ready-to-use IP cores for markets and applications such as:

- Aerospace & Defense
- ASIC Prototyping
- Audio
- Automotive
- Broadcast & Pro AV
- Consumer Electronics
- High Performance Computing and Data Storage
- Industrial
- Medical
- Video & Image Processing

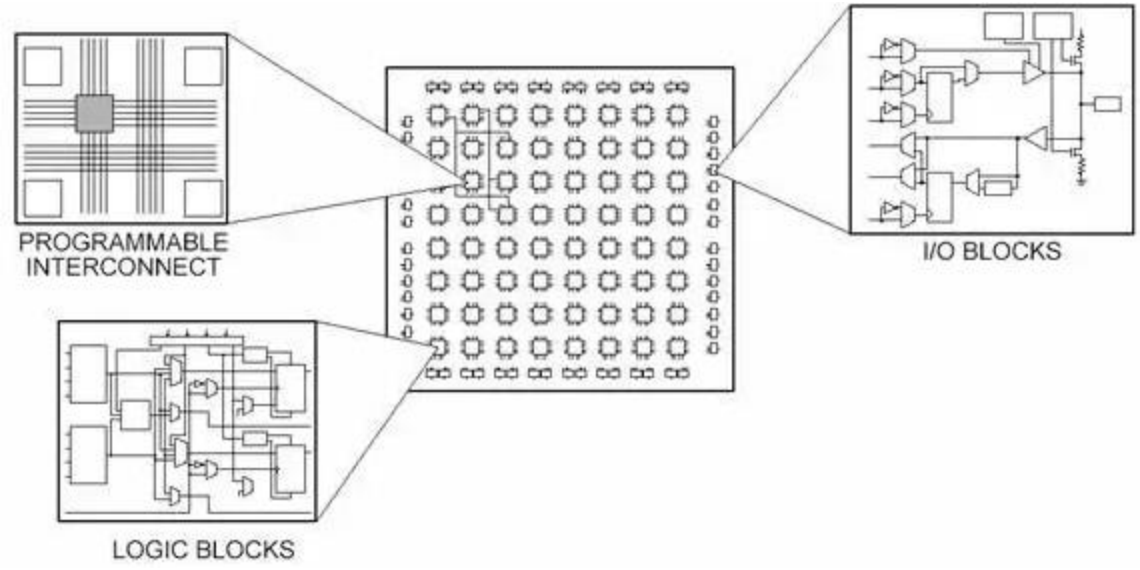


FIGURE 1.

2. CHIP DESIGN AND AVAILABLE TOOLS

Today's ASIC Chips are pretty complex packed with larger chunk of transistors targeted to a specific manufacturing process for fabricating the integrated circuits, in a sub nanometer regime, involving many of challenges, like knowledge of various protocols, architectures, models, formats, standards, knowledge about CMOS logic, Digital Design concepts, taming the EDA tool for the various design requirements like area, timing, power, thermal, noise, routability, lithography aware, knowledge about various variabilities like channel length, Vt, line width variations, lens aberrations, IR drop effects, inter-die, intra die-variations, effects, and various noise-effects like Package noise, EMI noise, power grid noise, cross-talk noise, and ability to test and validate and know to model and characterize all these effects upfront in the design-phase, steps to increase yield to increase profitability curve, with short span of time-to market to minimize the risk and maximize the predictability and an modular approach to Success. Now let's delve in to the "Art of Chip Designing".

The metric for today's ASIC Design is on one side of the coin is the maximum number of devices integrated, reduced die-size, optimal power, speed thermal performance, addressing signal integrity, addressing reliability, enhanced yield techniques, reducing PLL jitters for reliable functionality, testability, integration of analog and digital in single SoC, Lithography friendly DRC, Functionality met, high speed interfaces to memories, the IO fabric, IO buffer analysis and selection, implementation to validate the silicon, ability to in-corporate last minute spec changes/functional & timing bugs Engineering Change Orders, Optimized package feasible, development phases involving multi-geographical multi-site, complex database handling, various models/abstractions/standards/formats/Protocols, advanced process, library characterization modelling silicon, higher degree of EDA tools, design re-use standards, building designs robust enough to deal with EMI noise/package noise/power-ground noise/cross-talk noise/substrate-noise/clock-jitter/process uncertainties/IR-drop/On-chip variation and on other side of the coin, how to address all these issues right from the design stage is the challenge of today's Chip Design Industry.

This tight bonding of integration across the domains/abstractions/tools/designers/process/protocols/standards/design re-use/decision in optimal trade-offs/design know-how's/cross-culture design community needs a modular uniform approach, to bring first pass silicon success. Now let's deal with the Implementation challenges and steps to achieve it in the design-phase.

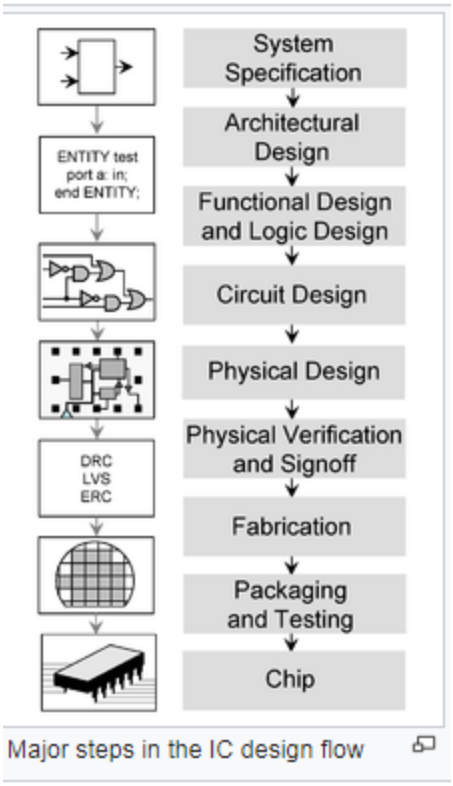

System Specification
Architectural Design
Functional Design and Logic Design
Circuit Design
Physical Design
Physical Verification and Signoff
Fabrication
Packaging and Testing
Chip
ENTITY test
port a: in;
end ENTITY;
DRC
LVS
ERC
Major steps in the IC design flow

FIGURE 2.

Few decades there can be severe negative impact on the environment. This makes the concept of smart cities a necessity. The creation of smart cities is a natural strategy to mitigate the problems emerging by rapid urbanization and urban population growth. Smart cities, in spite of the costs associated, once implemented can reduce energy consumption, water consumption, carbon emissions, transportation requirements, and city waste.
Smart cities around the globe are quite diverse in terms of their characteristics, requirements, and components. In general, standards established by organizations such as the International Organization for Standardization (ISO), provide globally understood specifications to drive growth while ensuring quality, efficiency, and safety. Standards can play an important role in the development and construction of the smart city. Standards can also provide requirements for monitoring the technical and functional performance of the smart cities. Standards can also help tackle climate change, address security and transportation issues, while ensuring the quality of water services. Standards take into account various factors such as business practices and resource management, while helping to monitor the smart city's performance and thus reduce its environmental impact. IEEE has been developing standards for smart cities for its different components including smart grids, IoT, eHealth, and intelligent transportation systems (ITS). A specific example of such a standard is ISO 37120 which defines 100 city performance indicators which include 46 core and 54 supporting indicators. Some selected indicators are the following: economy, education, energy, and environment, which can be used by city civic bodies to benchmark their service performance, learn best practices from other cities as well as compare their city against other cities.

VLSI design has become an important course at most of the electrical and computer engineering programs. However, buying licenses for commercial VLSI CAD is usually costly and requires high performance workstations which many academic institutes may not be able to afford. This paper provided some insight on the most popular open-source CAD tools that can be used in the academic field. Electic tool shows the best compatibility with different platforms, and also have plenty of design and check functions in a friendly user interface. Magic tool, which has a long history in VLSI design, has the best technical support and documentation, and it provides both stable and development versions for different use. It is very attractive to the academic institute users. Alliance tool, which can only run in Unix/Linux platforms, has the best usage stability and good balance in functions. But its popularity is limited by its strict operation system requirement. Even in Linux, it requires a re-compile with specific compiler. It also increases the complexity of the installation. In conclusion, Electric tool is highly recommended for Windows users. Yet, in Unix/Linux systems, Electric doesn't have significant advantage as it does in Windows. Magic and Alliance are also competitive for their own features. Magic may be more suitable for education that focuses on software development and Alliance may be good for education that focuses on design skill. This comparison is hopefully useful for those who are looking for a suitable VLSI CAD tool for academic purposes.

	Electric	Magic	Alliance
Format Supported	Uses *.elib format. Compatible with up to 17 formats including most common formats	Uses *.mag format. Compatible with CIF, Calma and IRSIM	Compatible with EDIF, VHDL, CIF, GDS2, SPICE
Automatic Functions	PLA, Cell, Pad and ROM generator, and 5 routers	Real-time design rule checking	Automatic optimization
Method of Model Building	Visualized GUI VHDL	Script Visualized GUI	Script Visualized GUI
Output Analysis	Multiple layout and schematic check functions and simulation faster than the others	Multiple layout and schematic check functions and simulation	Multiple layout and schematic check functions and simulation
Support	Acceptable availability of the documentation and help files.	Very detailed technology files, help files and other resources available on the website	Detailed documentation, help file and resources available on the website

Operation Difficulties	Easy to use	Easy to use, but need emulation to run in windows which sometimes cause problems	Easy to use
Installation Difficulties	Very easy, only one file and independent with any other files	Depend on Cygwin when using in windows, complicated installation procedure	Can only run in Unix and need to be re-compiled when using in various version of linux
System Requirement	Java based, cross platform, multiple OS supported also C++ binaries available	Linux/Unix, OS/2, DOS	Linux/Unix
License	GNU GPL	Berkeley Open-Source License	GNU GPL
Stability	Good with Java version but poor with C++ version	Not very good in non-Unix platform	Good
Version and updates	V8.09 for Java version and C++ version discontinued	Stable V7.5, development V8.0 About 6 month between updates	Version 5, about two month between updates

FIGURE 4.

3. WHAT IS ALTERA QUARTUS II?

The Altera Quartus II software, the industry's number one software in performance and productivity for CPLD, FPGA, and Hardcopy ASIC designs. Quartus II software delivers support for the latest 28-nm devices – the Arria V and Cyclone V devices – as well as enhancements to the Stratix V device support. Plan for success with System Console, a unique debug and monitoring solution. Other features include some enhancements to Qsys and a look at how software and hardware co-design will be enabled with the ARM-based system-on-a-chip (SoC) devices.

The Altera Quartus II design software provides a complete, multiplatform design environment that easily adapts to your specific design needs. It is a comprehensive environment for system-on-a-programmable-chip (SOPC) design. The Quartus II software includes solutions for all phases of FPGA and CPLD design. In addition, the Quartus II software allows you to use the Quartus II graphical user interface and command-line interface for each phase of the design flow. You can use one of these interfaces for the entire flow, or you can use different options at different phases.

Featuring System Console is scalable, hardware debug and monitoring tool in Quartus II software. It provides an easy way to build GUI elements and connect them to hardware so that you can monitor performance, debug designs, and demonstrate design functionality. You can also debug your design anytime – in the lab, during simulation, or after the product has been shipped.

Simulation and system-level tools integrated with Quartus II software design flow are as follows:

- ModelSim software

- DSP Builder (requires additional license)
- Qsys

The Qsys system integration tool saves significant time and effort in the FPGA design process by automatically generating interconnect logic to connect intellectual property (IP) functions and subsystems. Qsys is the next-generation SOPC Builder tool powered by a new FPGA-optimized network-on-a-chip (NoC) technology, delivering higher performance, improved design reuse, and faster verification compared to SOPC Builder.

Quartus II software delivers superior synthesis and placement and routing, resulting in compilation time advantages and compilation time reduction using following features:

- Multiprocessor support
- Rapid Recompile
- Incremental compile

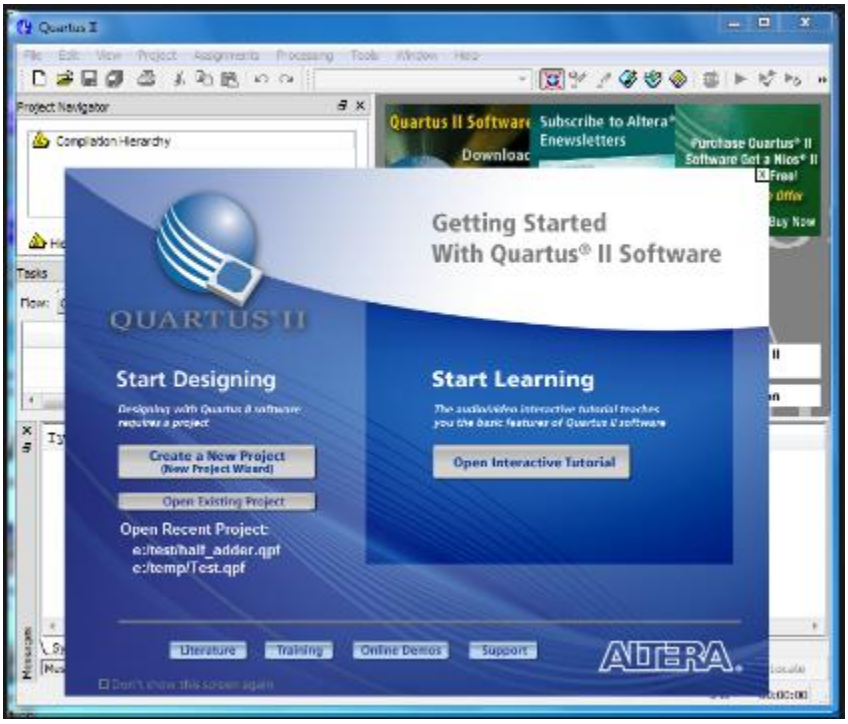


FIGURE 5.

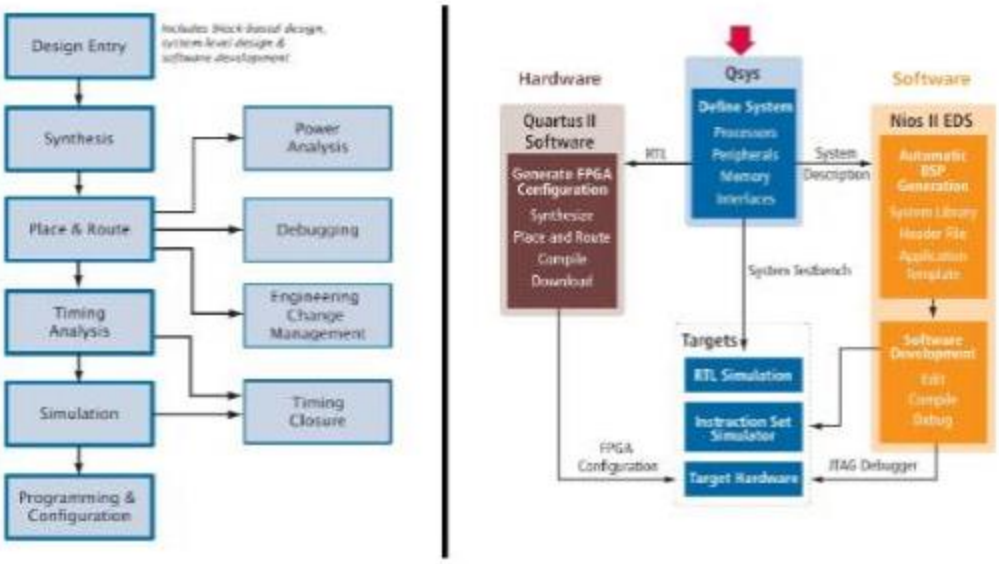


FIGURE 6.

Quartus II software is tailored to several different design methodologies, including incremental compilation design flows and block-based design flows. In a team-based incremental compilation flow, the design is divided into partitions. Each team member can functionally verify a partition independently, and then simply provide the source code for the partition to a project lead for integration. Using placeholders, the project lead can compile the larger design even if the source code is not yet complete for a partition. Compiling all design partitions in a single Quartus II project ensures that the design is compiled with a consistent set of assignments. Incremental design encompasses the following: • Synthesis • Design partitioning • Full compilation • Simulation, verification, and analysis, followed by changes to the design • Recompilation of only changed elements of the design hierarchy

Quartus Integrated Synthesis and the Fitter Quartus II software includes a comprehensive integrated synthesis solution and advanced integration with leading third-party synthesis software vendors. Quartus II integrated synthesis fully supports the Verilog HDL and VHDL languages, and includes algorithms to minimize gate count, remove redundant logic, and use device architecture efficiently. Quartus II integrated synthesis includes advanced synthesis options and compiler directives to guide the synthesis process to achieve optimal results. The Quartus II Fitter places and routes your design for the target device. Using the database created by Quartus II integrated synthesis, the Fitter matches the logic and timing requirements of the project with the available resources of the target device. It assigns each logic function to the best logic cell location for routing and timing, and selects appropriate interconnection paths and pin assignments.

Assignments allow you to specify various options and settings for the logic in your design. When you make resource assignments in your design, Quartus II software attempts to match those resource assignments with the resources on the device, meet any other constraints you have set, and then optimize the remaining logic in the design. The Assignment Editor and Pin Planner are interfaces for creating and editing pin, node, and entity-level assignments in Quartus II software. The Pin Planner allows you to make assignments to individual pins and also groups of pins. It includes a package view of the device with different colors and symbols that represent the different types of pins and additional symbols that represent I/O banks. The symbols used in the Pin Planner are very similar to the symbols used in device family data sheets.

4. DIFFERENT ALTERA FPGA PRODUCTS

Stratix series: Stratix FPGAs are typically programmed in hardware description languages such as VHDL or Verilog, using the Altera Quartus computer software.

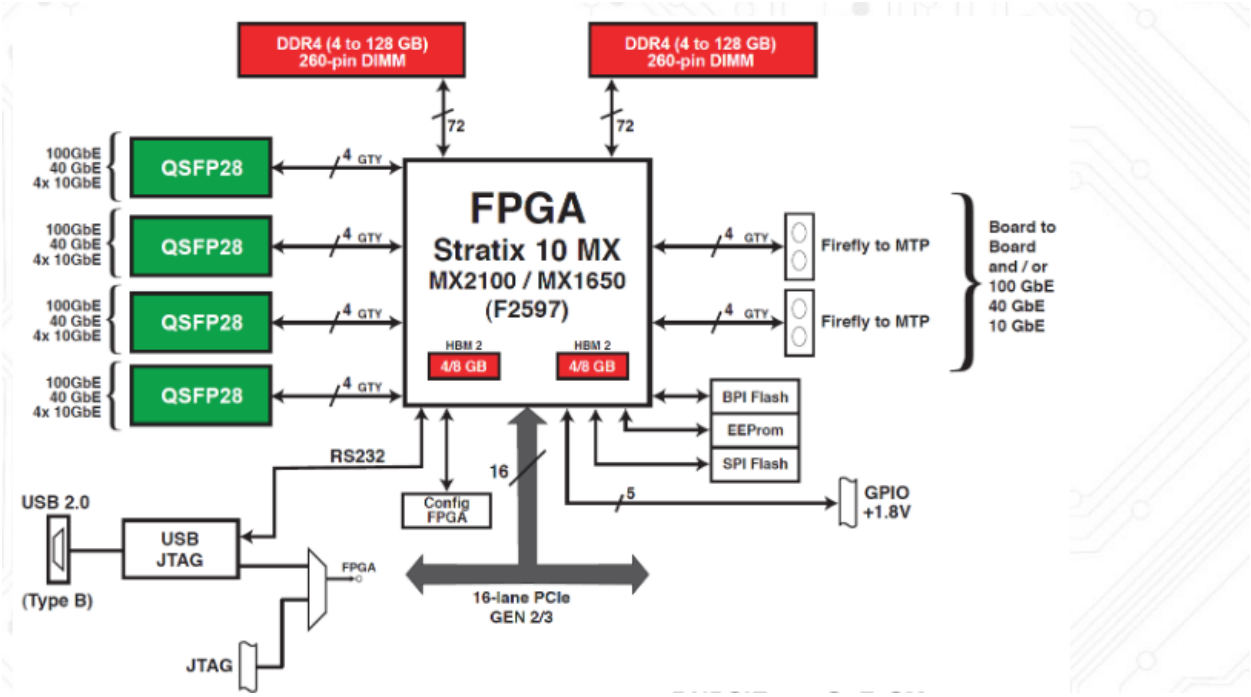


FIGURE 7.

Cyclone series: Intel Cyclone® Family FPGAs are built to meet your low-power, cost-sensitive design needs, enabling you to get to market faster. Each generation of Cyclone FPGAs solves the technical challenges of increased integration, increased performance, lower power, and faster time to market while meeting cost-sensitive requirements.

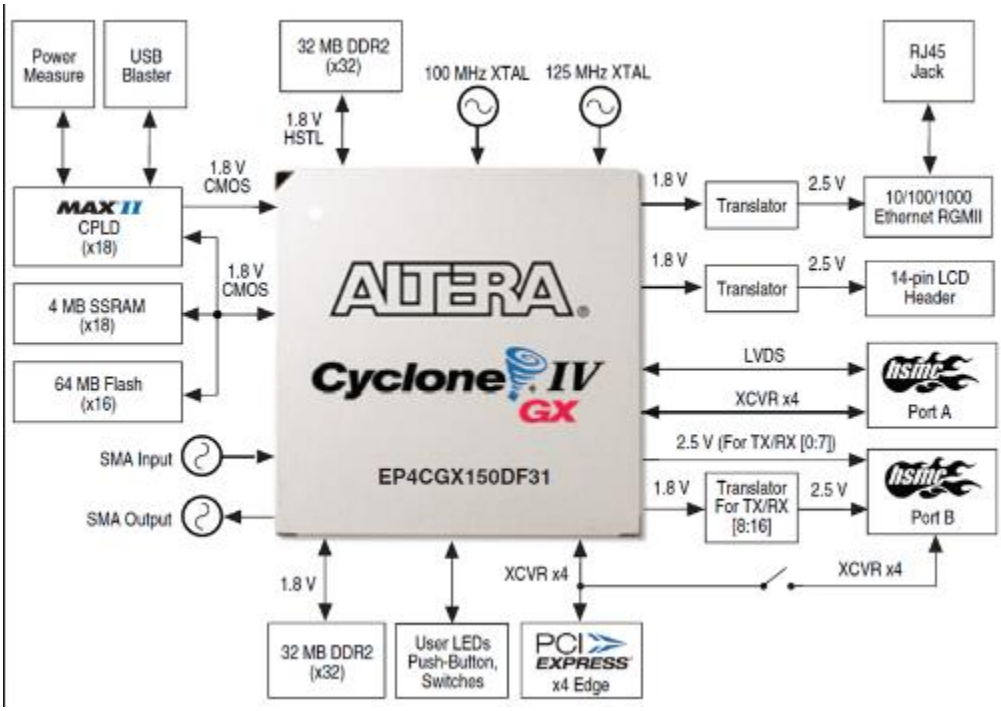


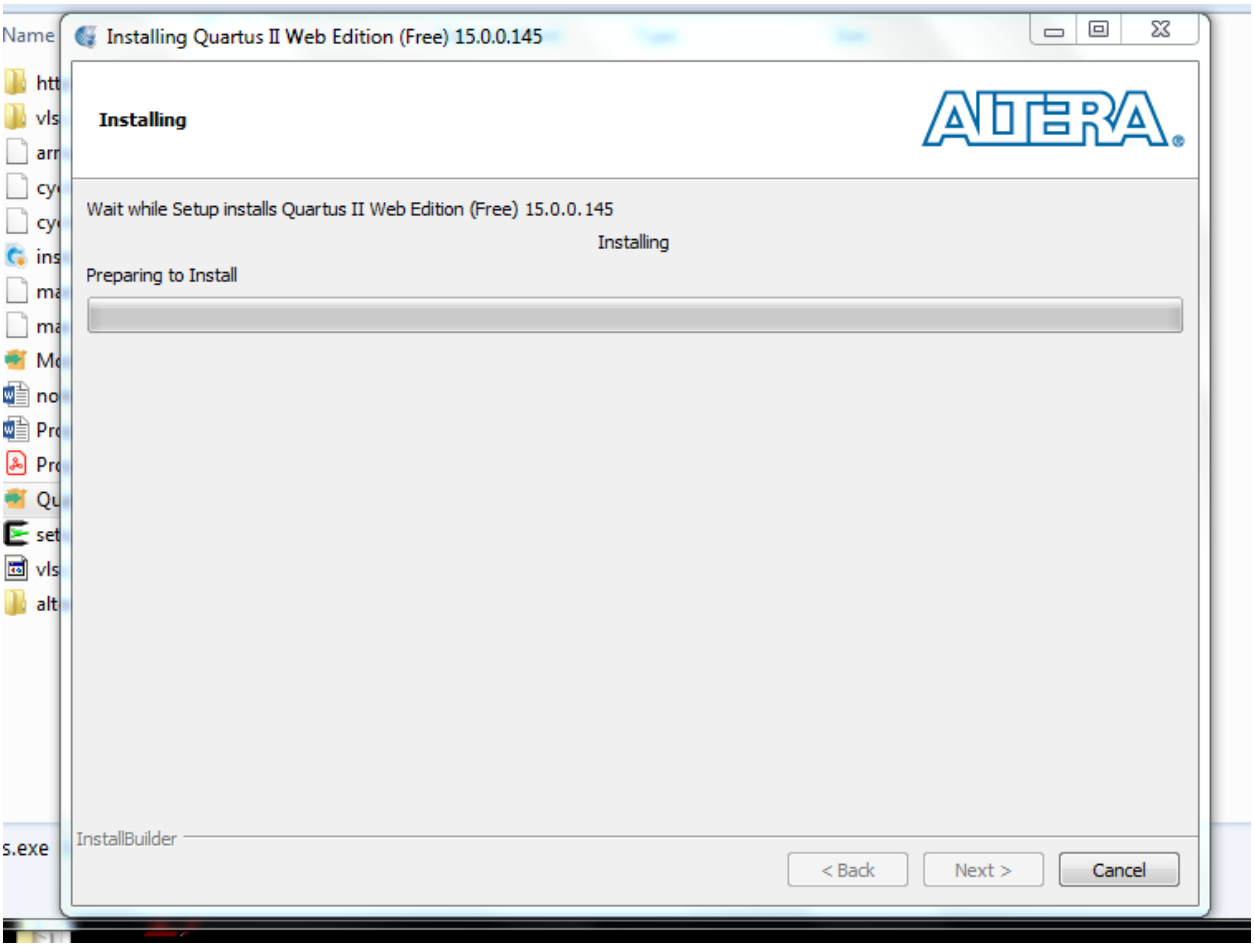
FIGURE 8.

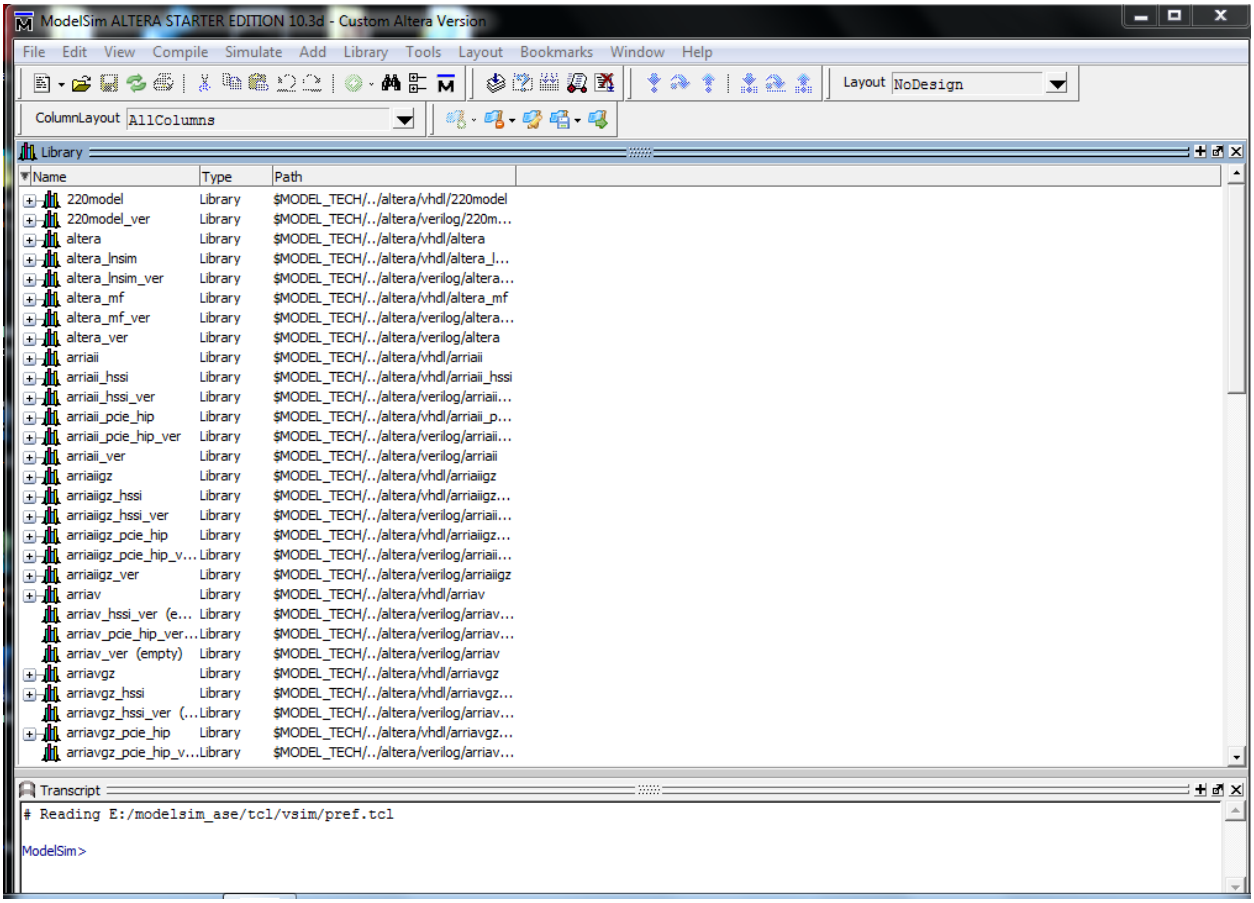
5. IMPLEMENTATION DETAILS

The purpose of the implementation is to create a 4-bit adder in Quartus II. A 4-bit Adder is a simple model of a calculator. It takes in two numbers of 4 bits each, allowing us to take numbers 0-15, but we will be using numbers 0-9. The numbers are then added together. The circuit is made in Quartus II and then is programmed onto a Field Programmable Gate Array (FPGA) which allows the circuit to be used.

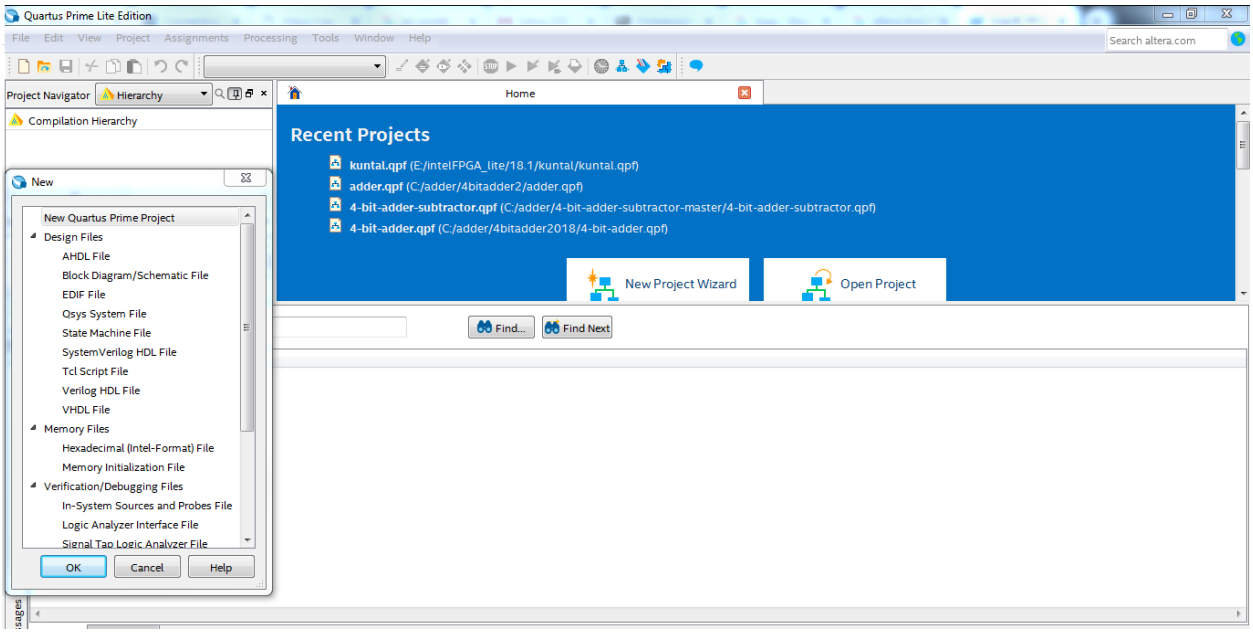
Web edition installation

I have tried to install web edition but had some issues and hence later installed Lite edition.

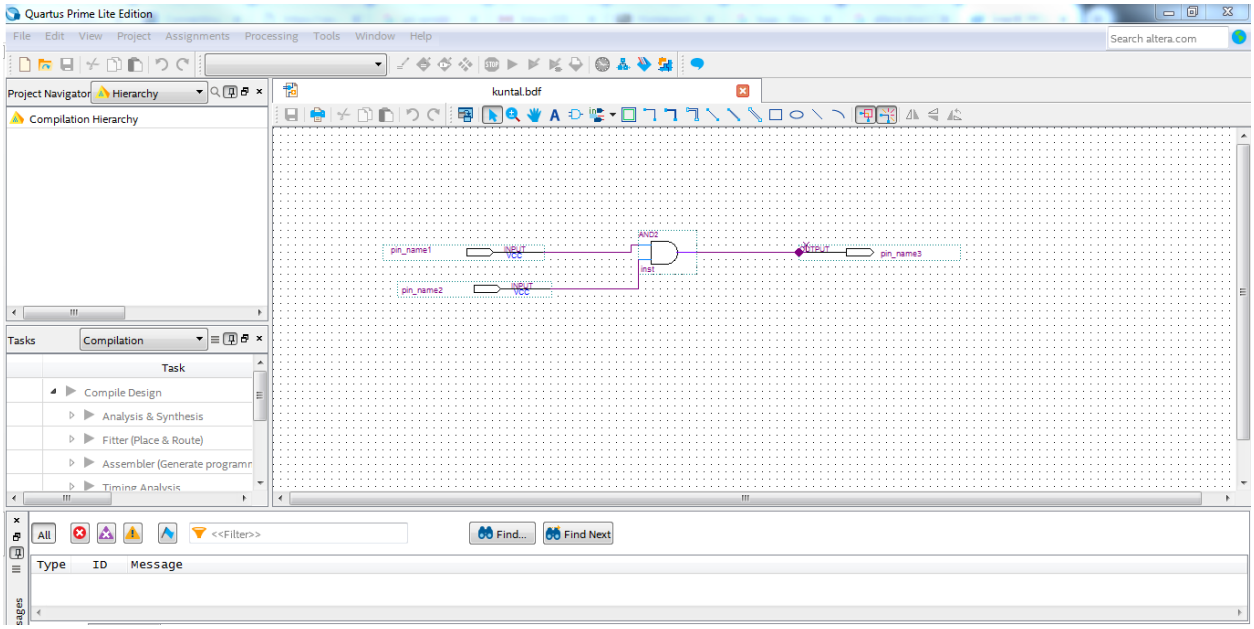




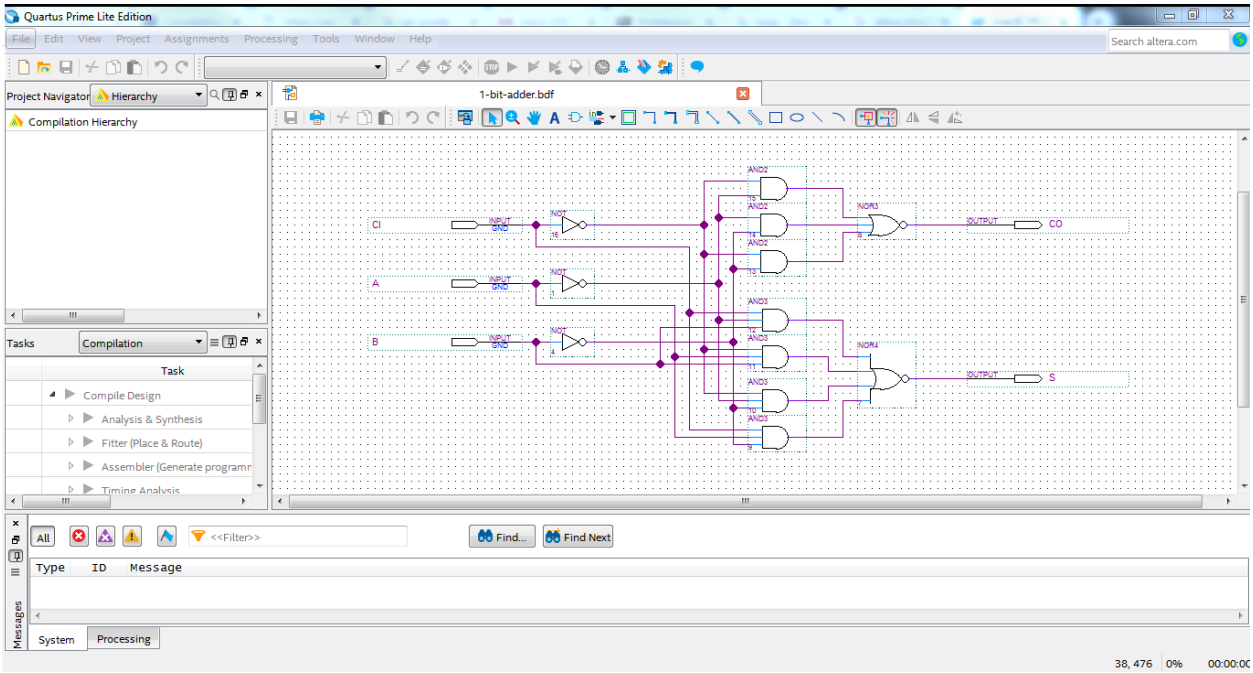
Lite edition installation:



Simple design schematic: To check the functionality and overall design flow I have simulates AND gate first.

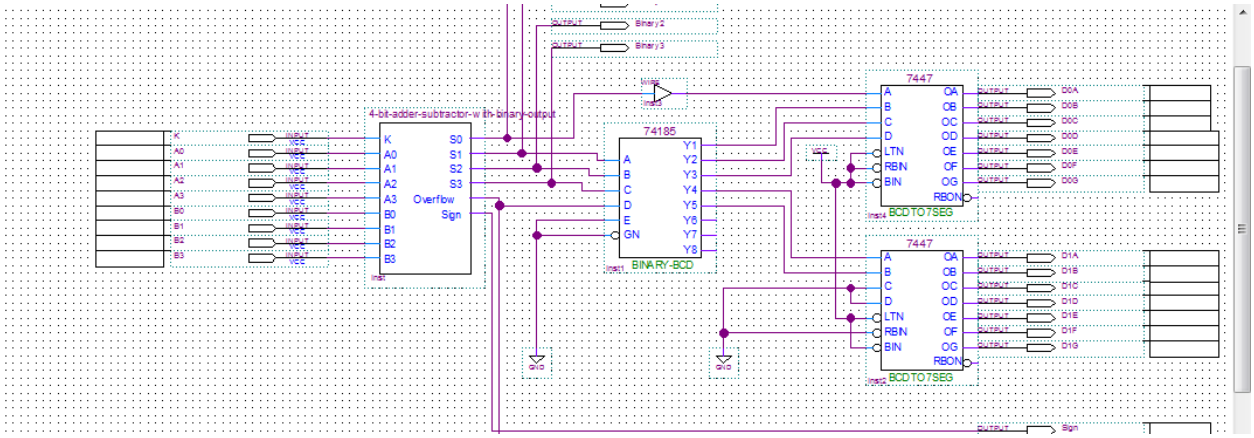


1-bit adder

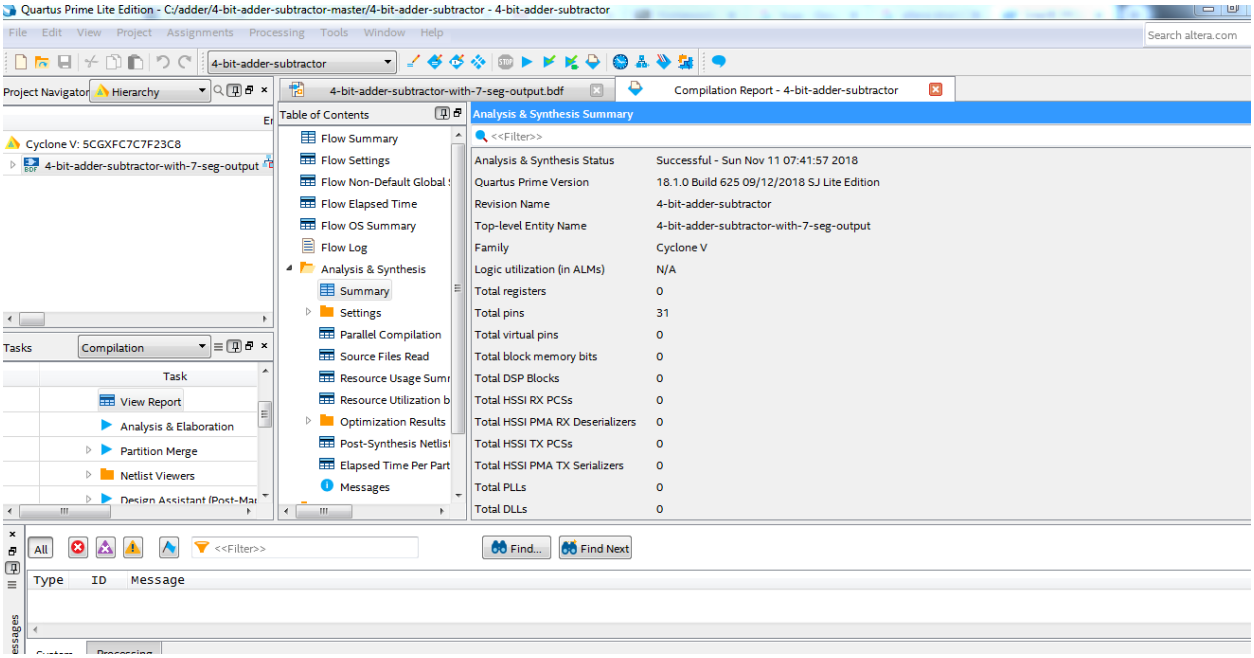


After implementing 1-bit adder I have converted it into bdf files which are block diagram schematic and uses it as an block in the final circuit.

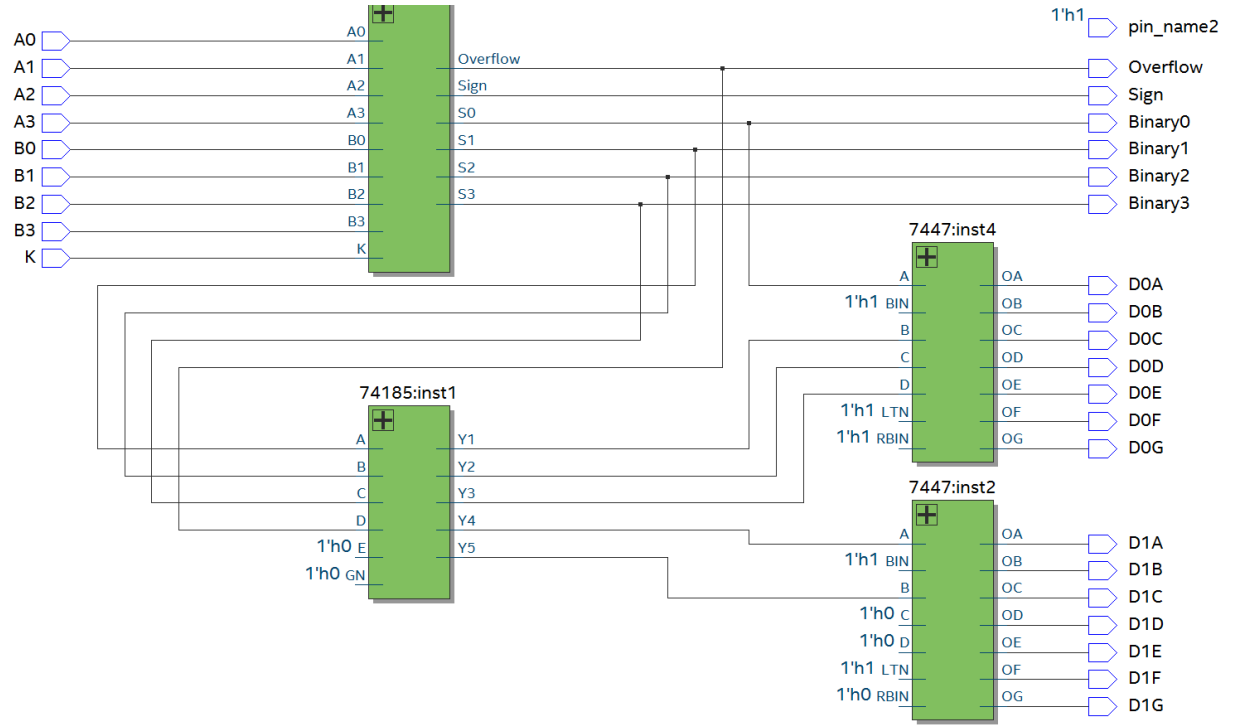
Final circuit: This circuit is made up of 4 1-bit adders and multiplexer with clock generator components.



Simulation Results



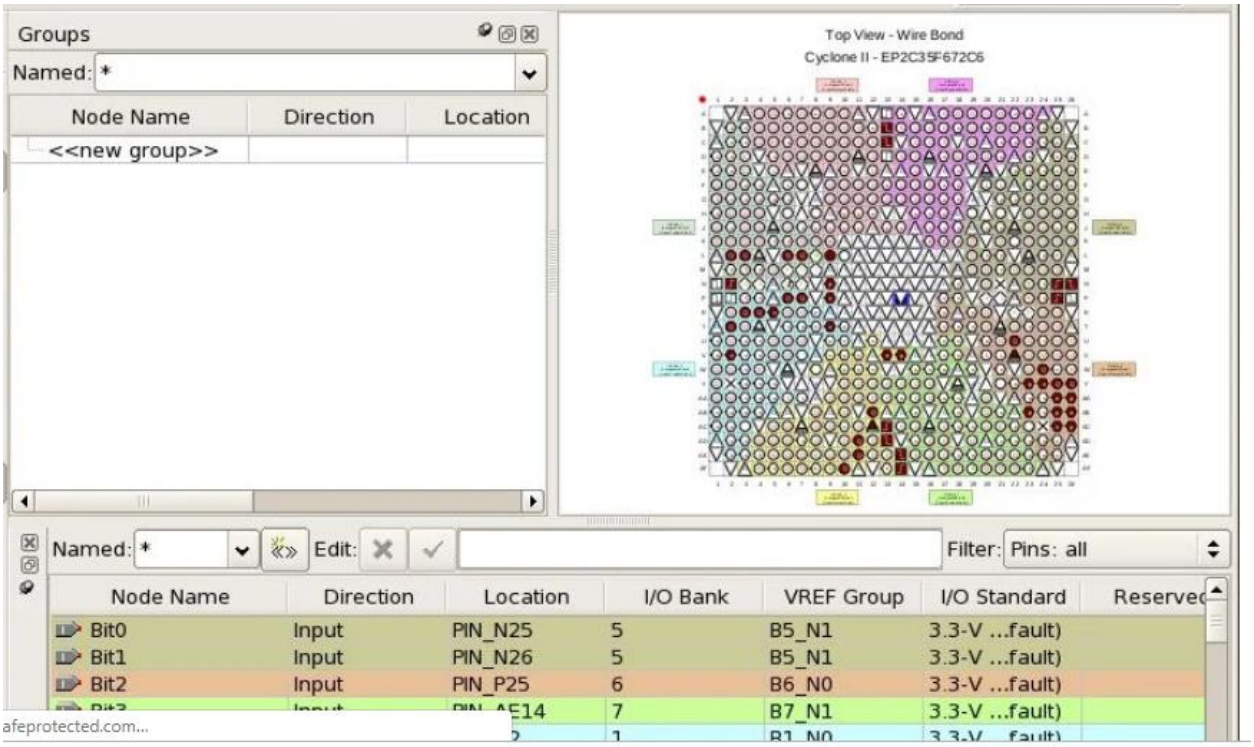
Netlist viewers



Pin assignments

```
io_4iomodule_c5_index: 77gpio_index: 2
io_4iomodule_c5_index: 60gpio_index: 476
io_4iomodule_c5_index: 62gpio_index: 6
io_4iomodule_c5_index: 26gpio_index: 472
io_4iomodule_c5_index: 20gpio_index: 10
io_4iomodule_c5_index: 12gpio_index: 468
io_4iomodule_c5_index: 27gpio_index: 14
io_4iomodule_c5_index: 71gpio_index: 464
io_4iomodule_c5_index: 56gpio_index: 19
io_4iomodule_c5_index: 14gpio_index: 460
io_4iomodule_c5_index: 22gpio_index: 22
io_4iomodule_c5_index: 10gpio_index: 456
io_4iomodule_c5_index: 11gpio_index: 27
io_4iomodule_c5_index: 73gpio_index: 452
io_4iomodule_c5_index: 74gpio_index: 30
io_4iomodule_c5_index: 76gpio_index: 448
io_4iomodule_c5_index: 2gpio_index: 35
io_4iomodule_c5_index: 78gpio_index: 444
io_4iomodule_c5_index: 9gpio_index: 38
io_4iomodule_c5_index: 36gpio_index: 440
io_4iomodule_c5_index: 51gpio_index: 43
io_4iomodule_c5_index: 23gpio_index: 436
io_4iomodule_c5_index: 53gpio_index: 46
io_4iomodule_c5_index: 50gpio_index: 432
io_4iomodule_c5_index: 0gpio_index: 51
io_4iomodule_c5_index: 43gpio_index: 428
io_4iomodule_c5_index: 67gpio_index: 54
io_4iomodule_c5_index: 16gpio_index: 424
io_4iomodule_c5_index: 44gpio_index: 59
io_4iomodule_c5_index: 29gpio_index: 420
io_4iomodule_c5_index: 1gpio_index: 62
io_4iomodule_c5_index: 8gpio_index: 416
```

Final pin layout



6. CONCLUSIONS

During this project I have faced main difficulties in installation and multi-processor simulation. This tool requires very powerful processor because for incremental design support it requires lots of calculations backend. Also I have faces difficulty to create VHDL files by Model slim tool because of by default pin assignments in multiplexer component available in Altera Quartus library. Overall simulation time and layout design is almost same as other EDA tools.

REFERENCES

1. https://www.xilinx.com/products/silicon-devices/fpga/what-is-an-fpga.html

2.https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/pr/quartusii_60_faq.pdf

3. https://www.google.com/search?q=fpga&safe=active&rlz=2C1OPRB_enUS0537US0537&source=lnms&tbm=isch&sa=X&ved=0ahUKEwj98IO15fLeAhXKJt8KHTk9BI4Q_AUIECgD&biw=1366&bih=577#imgrc=DgF0yUtkt9eHeM:

4. P. K. Chan, M. D. F. Schlag, "Architectural Tradeoffs in Field-Programmable-Device-Based Computing Systems", IEEE Workshop on FPCAs for Custom Computing Machines, pp. 152-161, 1993.

5. S. Hauck, G. Borriello, Pin Assignment for Multi-FPGA Systems, April 1994.

6. Xilinx Development System Reference Guide and the Programmable Logic Data Book, CA, San Jose: Xilinx, Inc., 1993.

7. Seung-Min, Baek Tae, Yong Kuc, "An adaptive PID learning control of DC motors", *International Conference on Systems Man and Cybernetics*, vol. 3, pp. 2877, Oct 1997.

8. Y A Sirin AKKA, Onur Akbati, Haluk Görgün, "Multiple closed loop system control with digital PID controller using FPGA", *2014 IEEE International Conference on Control Decision and Information Technologies.*.

9. http://www.altera.com.

10. https://en.wikipedia.org/wiki/Integrated_circuit_design

11. http://eecatalog.com/chipdesign/

12 http://ilin.asee.org/Conference2010/Papers/A1_Liu_Anan.pdf

13. https://www.element14.com/community/docs/DOC-40098/l/altera-introduction-to-the-quartus-ii-software

14. http://www.ee.ic.ac.uk/pcheung/teaching/ee2_digital/R2_3%20quartus2_introduction.pdf

15 https://www.dinigroup.com/web/altera.php

16. https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/manual/quartus_install.pdf

17. M. Ristović HDL Code generation using MATLAB/Simulink, INFOTEH-JAHORINA, Vol. 11, March 2012, pp 1035-1038
18. MATLAB (ver. R2011a), documentation: Simulink HDL Coder/Demos/ Computation, Mathworks, Inc. 2011,

19. Cyclone II Device Handbook, Chapter 2. Cyclone II Architecture (ver 3.1, Feb 2007), Chapter 3. Configuration & Testing (ver 2.2, Feb 2007) and Chapter 10. Selectable I/O Standards in Syclone II Devices (ver 2.4, Feb 2008), http://www.altera.com/literature/lit-cyc2.jsp

20. https://www.microsemi.com/document-portal/doc_view/129895-ac265-clock-generation-and-distribution-design-example-app-note