HCTL Open International Journal of Technology Innovations and Research (IJTIR)
http://ijtir.hctl.org
Volume 14, April 2015
e-ISSN: 2321-1814, ISBN (Print): 978-1-62951-946-3

Conversion of MATLAB code in VHDL using HDL Coder & Implementation of Code on FPGA

Harshit Pant[1], Himanshu Bourai[2] and Gaurav Singh Rana[3], S. C. Yadav[4]

[1,2,3]Student, Dept. of Electronics & Communication, Graphic Era University, Dehradun
[4]Asst. Professor, Dept. of Electronics & Communication, Graphic Era University, Dehradun

harshitpant123@gmail.com

Abstract

The advancement of technology in these days give rises to many tools where implementation of an electronics circuit, algorithms and different control or communication systems is possible. MATLAB is one of the tools which are able design and test systems and algorithm. To check systems performance on a practical system there is a need to implement it on a target device or FPGA. HDL coder in MATLAB is the one of the tool by which convert MATLAB code (floating point design) to VHDL code (bit stream) so that the code may run in FPGA kit. This paper consists of how to convert our MATLAB code to VHDL code by HDL coder and what is the source utility of the converted code.

Keywords

MATLAB, HDL coder, VHDL.

Introduction

MATLAB means matrix laboratory. It is a powerful tool use to design any mathematical model very easily and check there result. MATLAB has several specifications but by using MATLAB the analysis of hardware design is not possible it gives the software analysis and results using its calculation in matrix form. To analyze hardware, VHDL codes and VERILOG codes are used by which implementation of that code can be made possible on target device. Now a major question rises is MATLAB code able to convert its software design in any hardware descriptive language and the answer comes itself by using HDL coder in MATLAB. HDL coder is work as a bridge which connects two different software MATLAB and HDL software. HDL coder one can convert .m code to .vhd code or it can change the MATLAB code to VHDL code so that it can implement on FPGA kit. Main advantage of HDL coder is that it gives the elapsed time to execute the program. To convert code, initially it is compulsory to convert floating point design (MATLAB code) to fixed point design in which every variable length is fixed to a particular no of bits for

e-ISSN: 2321-1814, ISBN (Print): 978-1-62951-946-3

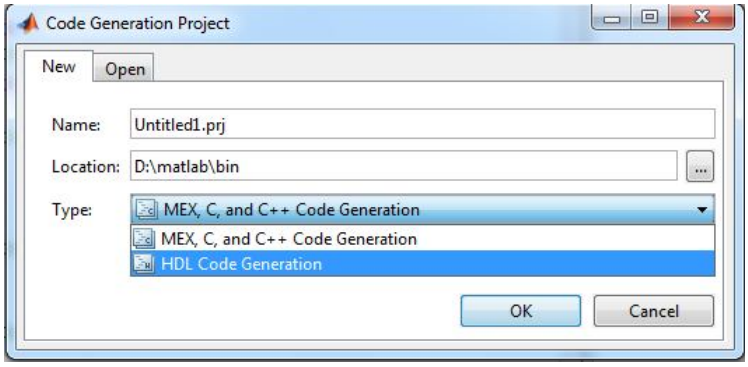
MATLAB® Version 7.3 (R2006b) added support for 64-bit indexing. With 64-bit indexing, we can create variables with up to (2^48) - 1 elements on 64-bit platforms [1].

Floating and Fixed Point Variables

Digital signal processing can be divided into two categories fixed and floating points. This refers to format used to store and manipulate no. within devices. Fixed point DSPs are usually representing no. in bit form of different length (16 bit or more) [2]. Floating point DSP refers to the values where variable size is not fixed. MATLAB code is a floating point design and HDL code is a fixed point design.

HDL Coder

HDL coder is a coder in MATLAB which is used to covert .m code into .vhd code so that the code can run for FPGA and ASIC design. It is target independent for Xilinx and Altera FPGA's. This coder supports only MATLAB functions it provides an advisor which can generate VHDL codes. HDL coder is open using coder command. Those commands ask to open earlier project or to create new project.



"Figure 1: Project creation for code conversion"

After opening earlier project or creating new project that HDL coder it is required to create MATLAB test bench and MATLAB function. After creating MATLAB function and test bench the compilation of the code is required that is completed by MEX compiler.

MEX COMPILER - A MEX compiler is a compiler which create a binary mex file of the source code .MEX-files generated using Microsoft Windows. Software Development Kit (SDK) require that Microsoft Visual Studio 2010 run-time libraries be available on the computer they are run on[1]. Before the conversion of the code to VHDL initially a compilation has to take place on system by a command –

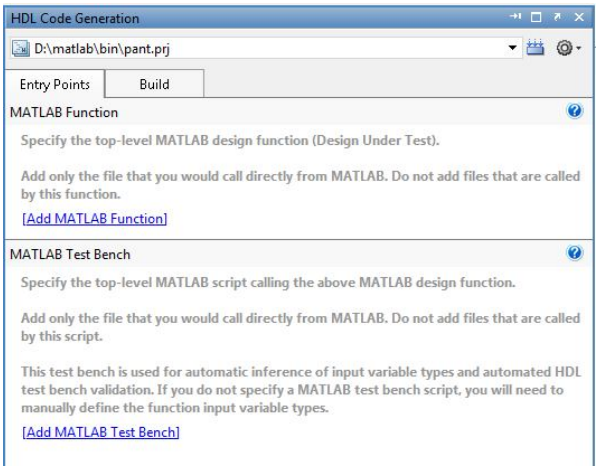
Mex –setup

The workflow of the command Mex-setup is shown in the dialog box below:

```
>> mex –setup

Welcome to mex -setup.  This utility will help you set up
a default compiler.  For a list of supported compilers, see
http://www.mathworks.com/support/compilers/R2012a/win32.html

Please choose your compiler for building MEX-files:
```

```
Would you like mex to locate installed compilers [y]/n? y

Select a compiler:
[1] Lcc-win32 C 2.4.1 in D:\matlab\sys\lcc

[0] None

Compiler: 1

Please verify your choices:

Compiler: Lcc-win32 C 2.4.1
Location: D:\matlab\sys\lcc

Are these correct [y]/n? y

Trying to update options file:
C:\Users\abc\AppData\Roaming\MathWorks\MATLAB\R2012a\mexopts.bat
From template:          D:\matlab\bin\win32\mexopts\lccopts.bat

Done . . .
```

After completion of mex compilation HDL coder asks to add MATLAB function and test bench.

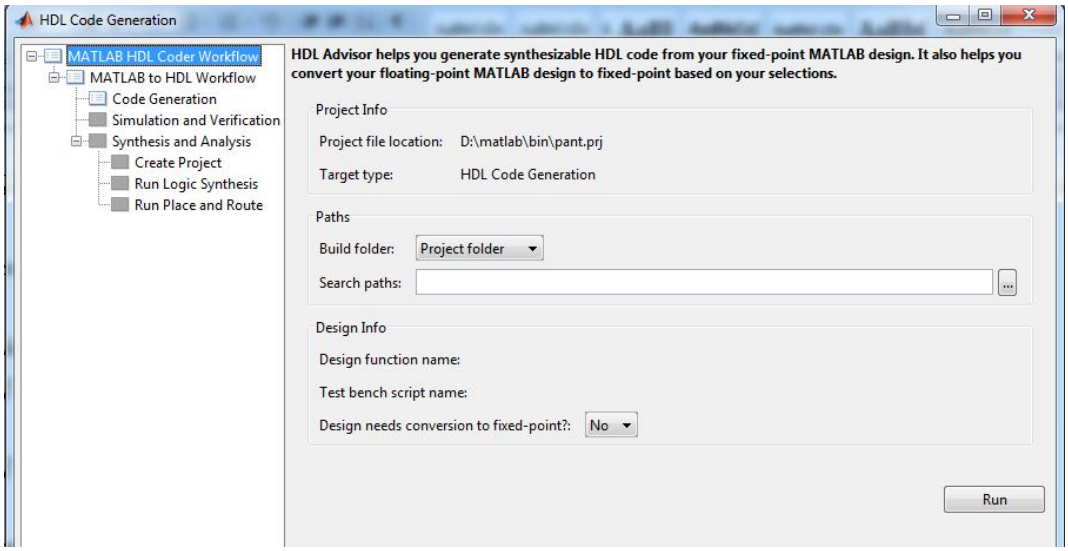


"Figure 2: Entry points of HDL coder"

After adding function and test bench a advisor is available in the HDL coder which is used to run MATLAB code and generate VHDL code. If advisor does not run successfully then error has to be removed from the .m code. And if it runs successfully then 3 files are generated:

1. pkg.vhd (package file of VHDL code.)
2. cod.vhd (VHDL code.)
3. tb.vhd (test bench of code.)

The advisor in HDL coder is shown below.



"Figure 3: Advisor"

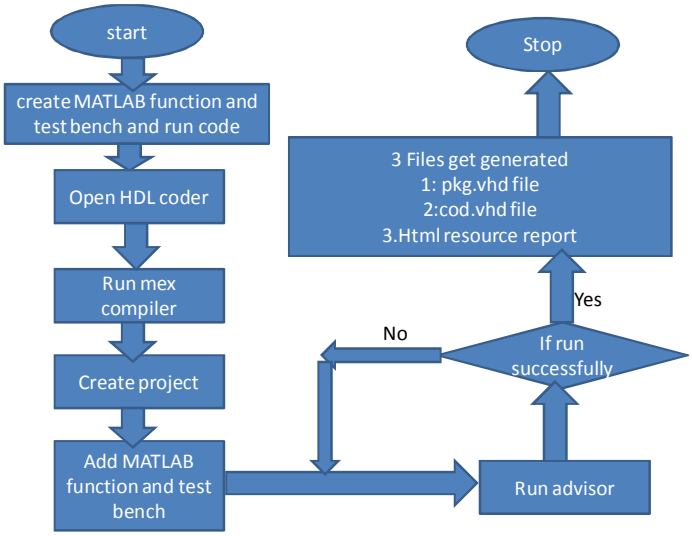
After these steps HDL code conversion of MATLAB code successfully take place.
After that completion one more report is generated which is an .html file. That file shows the resource utilization of generated VHDL code or it told how many adders, subtractors, multipliers, and multiplexers are required to implement that design in hardware [3].
The generated code is now synthesized in the software of front end designing i.e. XILINX. After all these models a synthesize report [4] is generated which tells summary of the whole design. The major reports generated in synthesizing are:

1: Synthesis Options Summary
2: HDL Parsing
3: HDL Elaboration
4: HDL Synthesis
    4.1: HDL Synthesis Report
5: Advanced HDL Synthesis
    5.1: Advanced HDL Synthesis Report
6: Low Level Synthesis
7: Partition Report
8: Design Summary
    8.1: Primitive and Black Box Usage
    8.2: Device utilization summary
    8.3: Partition Resource Summary
    8.4: Timing Report
        8.4.1: Clock Information
        8.4.2: Asynchronous Control Signals Information
        8.4.3: Timing Summary
        8.4.4: Timing Details

After all the above report finally converted code successfully runs in XILINX [5]. For the better understanding of hardware we can check different model of our converted code.

Major models which are used to check the hardware are:
1: RTL model.
2: Technology schematic model.
Above model finally gives us the hardware look of the converted MATLAB code and our problem of defining hardware from MATLAB code got resolved.

The code conversion can be shown in a flow graph:



"Figure 4: Flow Chart of HDL Coder Workflow"

Example of Code Conversion

Now let us take an example of adder:
Let A and B are two inputs and c is their output:
So their MATLAB inputs are:
A=10, b=10
C=A+B;
C=20
For performing that task a MATLAB function and test bench is required

```
MATLAB function:
function [d]=add(a,b);

d=a+b;
```

```
MATLAB test bench:
a=10;

b=10;

c=add(a,b)
```

After creating function and test bench HDL coder opens and after that MEX compilation takes place. When MEX compilation completed then MATLAB coder asks for adding function and test bench of coder. For code of addition two entry points are:

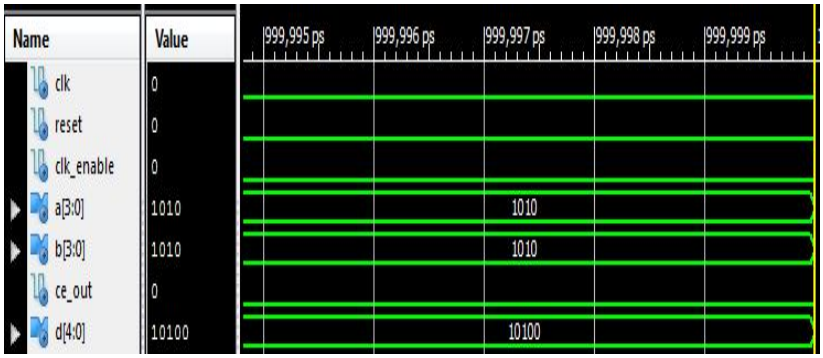
After filling entry points the advisor will run and if code generation option in advisor shows green tick then code generation generate a report and that report shows code is converted:
Report generated in code generation option of advisor is:

```
### Begin VHDL Code Generation
### Working on add_FixPt as add_FixPt.vhd
### Generating Resource Utilization Report
resource_report.html
### Begin test bench generation.
### Collect Test Bench Stimulus and Response
### Begin Simulation to log data

c =

20

### Simulation to log data completed in 0.2068 sec(s)
### Collecting data...
### Begin HDL test bench file generation with logged
samples
### Generating test bench: add_FixPt_tb.vhd
### Creating stimulus vectors...
### Elapsed Time: 29.6241 sec(s)
```

For an adder circuit package is not required so HDL coder does not generated package file. It only generates VHDL code and test bench [6]. The time taken by code to convert the code is 29.6241 sec. According to source utilization report resource used are:

Multiplier	0
Adder / Subtractor	1
Registers	0
Ram's	0
Multiplexer	0

Now analysis of adder in VHDL [7] [8] is required and its VHDL output in XILINX is shown below:



"Figure 5: VHDL OUTPUTS"

For hardware analysis we have RTL and technological schematic modal these two models are follows:

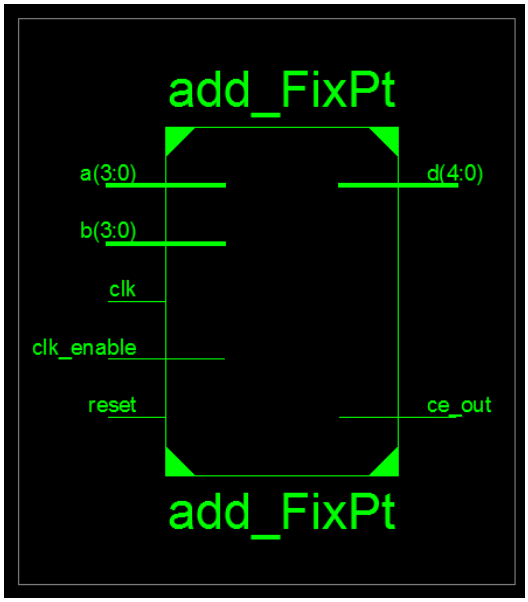
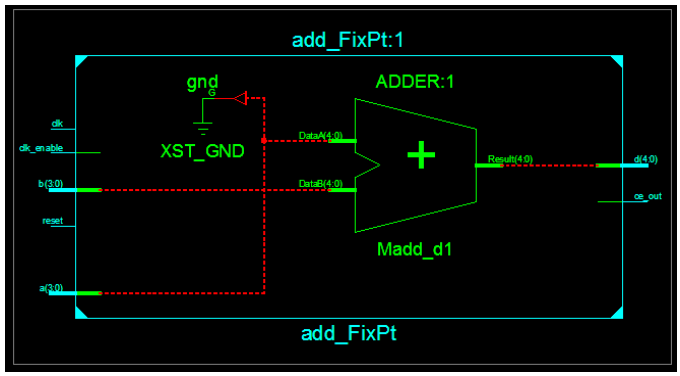
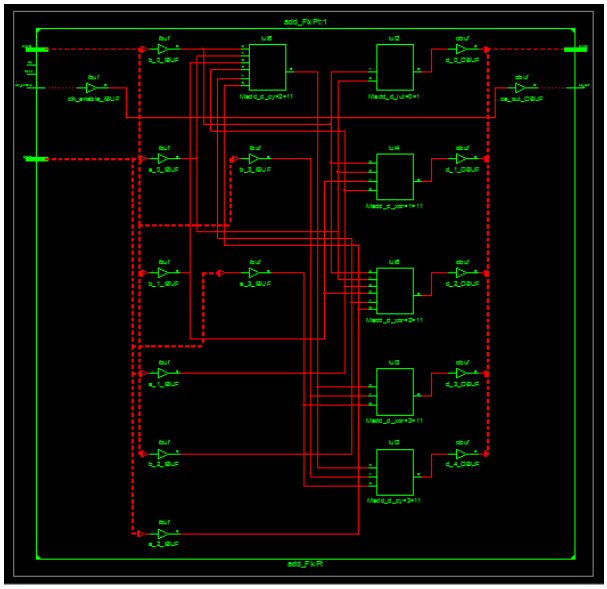


Figure 6: RTL Schematic and Technological schematic of Adder



"Figure 7: Internal View of RTL Schematic of Adder"



"Figure 8: Internal Structure of Technological Schematic"

Now it is seen that the converted code is running successfully in XILINX and the hardware analysis of code is clearly seen by RTL and technological schematic. After the code run in VHDL the analysis takes place for FPGA of Spartan 3A family. it is used to analyze code in XC3S50A target device and according to synthesize report of target device the resources used are:
1: # Adders/Subtractors: 1

2: 5-bit adder: 1

Conclusion

In this paper it is proposed a efficient way for the conversion of MATLAB code in VHDL code by using HDL coder of MATLAB and it is seen that by using HDL coder .m file is successfully converted in .vhd file and by converting that .m file in .vhd file it is simple to analyze the hardware structure of generated module or it can be say that using HDL coder hardware analysis and FPGA implementation can be successfully take place by generating a simple source code of MATLAB. It is also shown that source utility report generated by HDL coder and FPGA synthesis are equal both of them requires equal no. of components. So it can be concluded that the HDL coder generate an efficient way to convert floating point design of MATLAB code into fixed point design of VHDL code.

References

[1] "HDL coder™ 1.5 users guide", The Mathworks, Inc.2009.

[2] Subhash Chandra Yadav, Krishan Raj, R. G. Varshney,"Error In 2d DCT Using Cosine Function Getting By' C' And 'VHDL' And 'IEEE 754' Representation of DCT Values" International journal of electronics amd computer science Engg,PP.252-257,march2012

[3] Malay Haldar, Anshuman Nayak, NagrajShenoy, Alok Choudhary, and Prith Banerjeee. "FPGA Hardware Synthesis from MATLAB", Fourteenth International Conference on VLSI Design, 2001.

[4] Gerstlauer A, Haubelt C, Pimentel A D, Stefanov T P, Gajski D D, Teich J. Electronic System-Level Synthesis Methodologies. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 2009; 28(10)1517-1530.

[5] Xilinx user guide. http://www.xilinx.com/products/software/sysgen/app_docs/user_guide.htm.

[6] V. A. Pedroni, "circuit design with VHDL" MIT press, 2004.

[7] Holmes, C., VHDL Language Course, Rutherford Appleton Laboratory, Microelectronics Support Centre, Chilton, Didcot, 23-25 May 1995.

[8] Riesgo T, Torroja Y, Torre E. Design methodologies based on hardware description languages. IEEE Transactions on Industrial Electronics 1999; 46(1) 3-12.