International Journal of Engineering & Technology, 7 (1.1) (2018) 710-713

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Development of graphical user interface for open source VLSI digital synthesis tool Qflow

K. Sripath Roy [1], K. Abhiram[2], M. Arun Sumanth[2], Jaishree Jaishankar[2*], P. Abhishek[2], B. Nikhil Prabhat[2] , L. Gnana Teja[2]

[1]Asst Professor, Department of Electronics and Communication Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur, Andhra Pradesh, India.
[2]Student, Department of Computer Science & Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur, Andhra Pradesh, India.
**Corresponding author E-mail: jaishree.jayashankar@gmail.com*

Abstract

There are many tools that are used for simulation in the domain of VLSI technology but none of them are easily accessible. There is a need for Free and open source tools in this stream so as to make them accessible to everyone. There are efficient tools that already exist in open source in VLSI stream but are not used widely because of their command line user interface. Hence, creating a user friendly interface will help many developers and users to work easily. This paper deals with the idea to solve the above issue by creating a Graphical User Interface for the open source VLSI tool called QFlow. Qflow is a tool used in synthesizing a VLSI circuit from the Verilog source code. There are multiple tools integrated with this tool to assure the simulation process. It is a combination of many dependencies that are used for synthesis, placement, layout viewing and routing in a fabrication process. All the independent tools used for the Verilog code simulation are integrated onto a single platform. Qt is used for creating the cross-stage application.

Keywords: *VLSI,Qflow ,Qt, Verilog*

1. Introduction

Qflow is a command line tool i.e. we use this tool using terminal commands specific to that tool. Because of not having GUI, so many people cannot use this tool as it does not have a user friendly interface. This disadvantage prevents its usage by the larger community of developers. Hence, to solve this issue we are working towards designing the Graphical User Interface for QFlow. To get in par with the commercial tools, there is a need to develop GUI for this tool. Hence, our area of work revolves around contributing in the VLSI stream [1].

It is a combination of many dependencies viz. Yosys, Graywolf, Magic, QRouter for synthesis placement, layout viewing, routing respectively in a fabrication process. These dependencies work independently and will give their respective output of their own [2]. In this Graphical user interface we are trying to integrate all the dependencies into a single platform. Using Qt we will use the output of the inner dependency tool to the other and vice versa.

2. Available VLSI tools

Qflow

Qflow is a tool chain for VLSI digital synthesis flow starting from verilog code and ending in a layout for a specific application. In the present world of VLSI, we have many proprietary tools like Xilinx design suite, Vivado and also EDA tools like Cadence, Synopsys etc. But these are commercial tools and so many small companies, startups and normal people cannot afford for these tools. Qflow is an open source tool which is free to use and is released under GPL license [3-5]. Many OpenCores are tested and fabricated using this tool and it is also an efficient tool to work with. Qflow mainly runs ASIC design and also some of Xilinx FPGA designs. This tool itself is a framework and has a number of tools in it. The VLSI design flow consists of the following steps.

- Verilog Synthesis
- Placement
- Routing
- Layout Generation

These tasks are done by different tools like Verilog Synthesis is done using Yosys tool, Placement by Graywolf, Routing by Qrouter and Layout by Magic. The workflow of these tools is described in the rest of the document [6]. Qflow has many dependencies which will perform many independent functions.

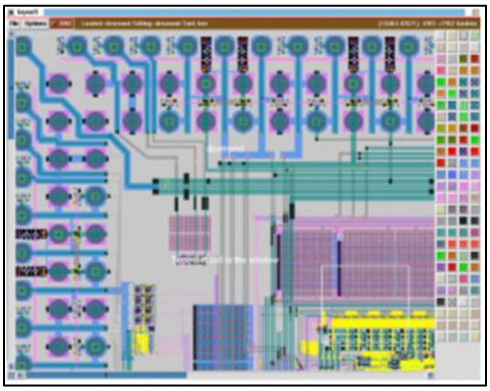


Fig. 1.11: Sample output using magic

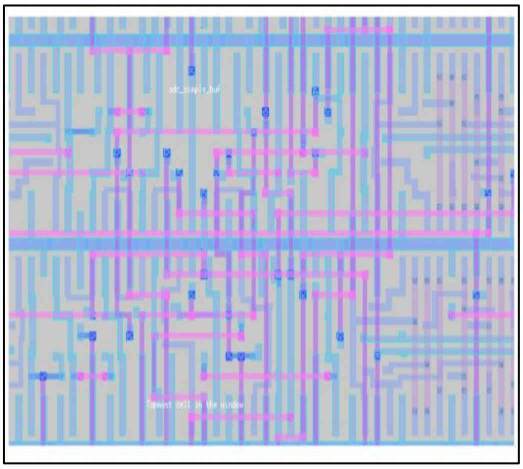


Fig. 1.12: Visualization

Magic

Magic is a VLSI layout tool, configured in the 1980s in Berkeley by John Oosterhout. Magic is an intuitive framework for making and changing VLSI circuit layouts. With Magic, you utilize a shading illustrations show and graphics tablet to outline essential cells and to join them progressively into expansive structures. Magic is not quite the same as other format editors you may have utilized. The most critical contrast is that Magic is more than only a shading painting tool: it sees a considerable amount about the idea of circuits and uses this data to furnish you with extra operations [7].

Magic installation needs plenty of disk space and high bandwidth internet connection. We first install Cygwin after installing Cygwin. We will be having a terminal on desktop and then the installation starts as per instructions given in the dialogue box. At present the magic version is 8.2.Magic uses two windows: one for text and a separate window for displaying layouts. Magic provides many features for its users such as plowing, which allows layouts to stretch [8-10].

When we talk about Qflow, Qflow will give the final layout in DEF format. Magic is the tool which can understand DEF file formats and display the layout using ASIC standard cells. So Qflow uses magic for layout display.

Qrouter

Qrouter is a tool to produce metallic courses and also to physically interface a netlist in a VLSI fabricating innovation. It is a maze router, otherwise called an "over the cell" router or an "sea of gate" router. That is, not at all like a channel router, it is typically bundled at the very least separating and places metal courses over the standard cells.

Qrouter utilizes the standard open arrangements LEF and DEF as info and yield records. Take the cell meanings

of a LEF record and investigate the geometry of every cell to decide contact focuses and course checks. Next, it peruses the area of the cell, the area of the pins and the rundown of associations from a DEF document, plays out the point by point way and composes a DEF record commented on as yield.

Yosys

Yosys is a free verilog synthesis suite. Many digital design layout is using hardware languages, VHDL and HDL. Yosys is first free software for verilog synthesis which have many similar verilog features. It supports for many Fpga's synthesis and it also supports many standard cells i.e ,ASIC synthesis. Here is the few list of features of yoys:

- It supports verilog-2005 features
- Supports EDIF, BLIF,BTOR formats
- It also gives RTL verilog file after synthesis

In the below figure yosys synthesis flow is described.

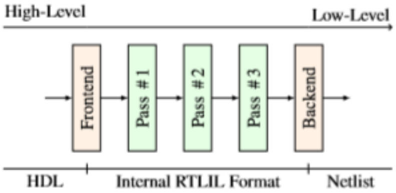


Fig. 1.13: Yosys data flow

So Yosys is the best open source synthesis tool as of today. So Qflow uses Yosys for verilog synthesis.

Iverilog

Iverilog is popularly known as Icarus Verilog. It works both as simulator and synthesizer tool. It works as compiler, assembling source code written in Verilog to other required format. It comes under the GNU General Public License. For group simulation, the compiler can produce a middle frame called vvp get together. This transitional frame is executed by the ``vvp" charge. Icarus keeps on showing signs of improvement and better. Icarus is being utilized for genuine plan work by organizations now as a test system, and is beginning to be helpful as a synthesizer for a Xilinx FPGA stream too. This center compiler, after it is done with parsing and semantic investigation, utilizes loadable code generators to produce code for bolstered targets. The tgt-*/registries contains the hotspot for the objective code generators that are packaged with Icarus Verilog

Qt

Qt is a cross-stage application system that is utilized to create application programming that can keep running on different programming and equipment stages with next to zero change to the hidden code base, while as yet being a local application with local abilities and speed.

3. Development plan

Linux is the platform chosen to design the GUI for QFlow as it is also a Free and Open Source operating system and it has a large community to back off for it's development and bug correction. Support of the community and availability of API can be used to solve real time problems. After a proper search for an efficient open source tool for synthesis of the Verilog source code, Qflow was observed to fit the requirements of an efficient synthesis tool.

Initially the layout for the widget and the action for every button in the widget was designed manually. Then python scripts to create widget operation was programmed. Later a script to connect the GUI code to the widget was scripted in python.

Later an installer was designed. The function of the installer is to install all the dependencies required for the Qflow without opening the terminal. This installer is a shell script that installs Qt, Qflow, Iverilog, Qrouter and the rest dependencies.

A double click on the installer will install the py-qt, iverilog, Qflow, GTK wave and also an optional directory for the python scripts to be placed. In the application directory Qflow is added as a widget, hence the Qflow logo is visible when 'Qflow' is searched for. When Qflow icon is clicked on, then it's Graphical User Interface comes into shape. It has buttons for project creation. When the corresponding button is clicked a new directory/folder for a new project is created. A folder/directory for layout, synthesis, source are created correspondingly. With a click on the 'New file' button a new file in the source directory will be created.

To perform these tasks there is a scripting file in the back end to execute the functions in a relay format i.e. in a continuous manner based on the previous event. The text file created is to be saved with a .v extension. Using the 'Run' button the entire simulation process can be executed. To exit from the setup created the GUI is configured with a 'quit' button

Fig. 3.1: Qflow after installation

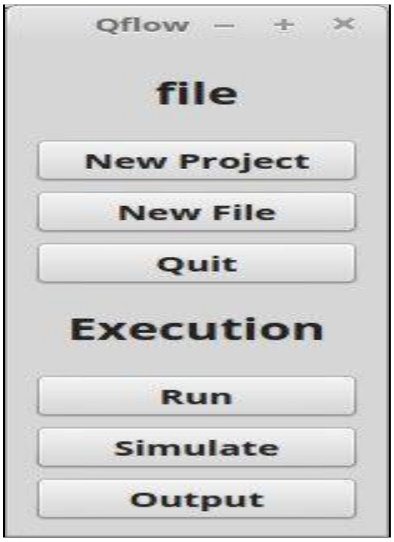


Fig. 3.2: Qflow Widget

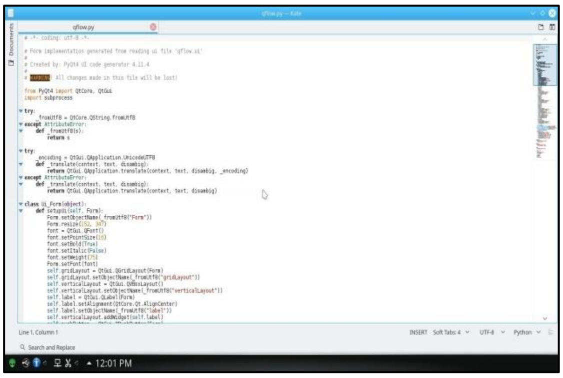
4. Conclusion and future scope

There are many open source tools to synthesize a VLSI circuit and one such efficient tool is QFlow. But though it is efficient it is found to have only a Command line interface. To solve this issue we have designed the Graphical user interface for QFlow. Developing front end for the tool is equally important as the back end as the front end makes it user friendly and universal to use. Our project is basic GUI for QFlow that can be further developed. The future of technology is free and open source software. Hence, we are in an urge to develop free and open source developmental tools in Electronics and Communication Domain. Though there are many open source tools in Communication field, there is dearth in the free and open source tools for VLSI. This turns out to be the responsibility of the community to develop such tools that help in defining the Open Cores which is the free and open s ource alternative for IP Cores. Hence, there is a lot of scope in VLSI stream where we need to develop the Open Cores. This GUI for QFlow is the basic step to contribute to the community that strives to develop the Open Cores.

5. Acknowledgment

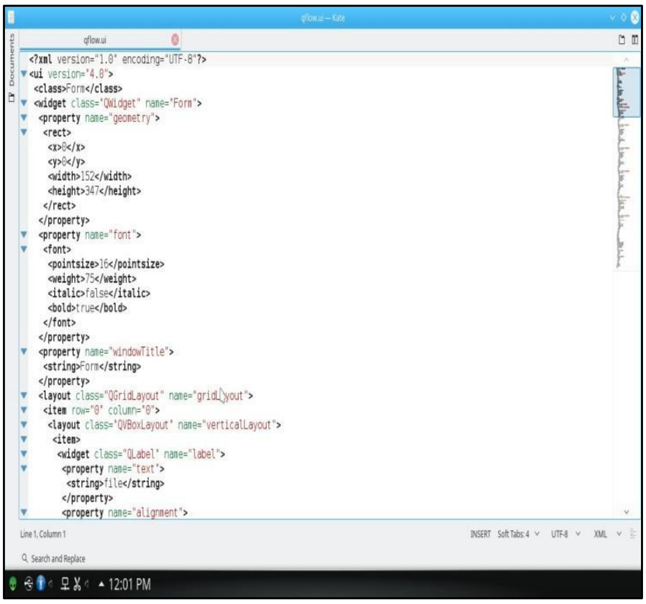
We thank Tim from United States who is currently working on Qflow development for helping us while working on Qflow functioning and installation problems

6. Results



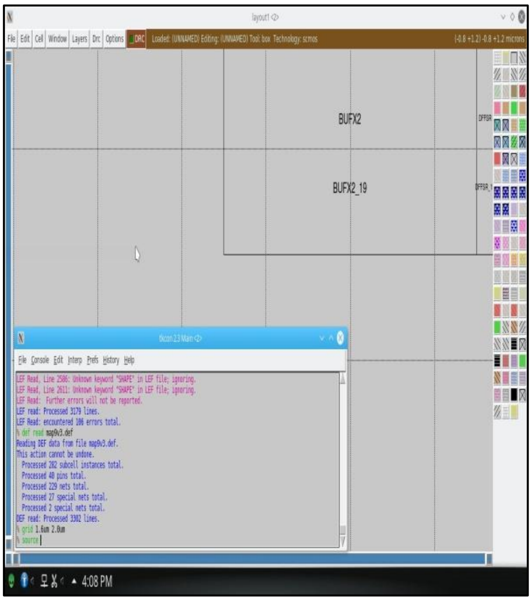
```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'qflow.ui'
#
# Created by: PyQt4 UI code generator 4.11.4
#
# WARNING! All changes made in this file will be lost!

from PyQt4 import QtCore, QtGui
import subprocess

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:
    _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)

class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName(_fromUtf8("Form"))
        Form.resize(152, 347)
        font = QtGui.QFont()
        font.setPointSize(16)
        font.setBold(True)
        font.setItalic(False)
        font.setWeight(75)
        Form.setFont(font)
        self.gridLayout = QtGui.QGridLayout(Form)
        self.gridLayout.setObjectName(_fromUtf8("gridLayout"))
        self.verticalLayout = QtGui.QVBoxLayout()
        self.verticalLayout.setObjectName(_fromUtf8("verticalLayout"))
        self.label = QtGui.QLabel(Form)
        self.label.setAlignment(QtCore.Qt.AlignCenter)
        self.label.setObjectName(_fromUtf8("label"))
        self.verticalLayout.addWidget(self.label)
```

Fig. 6.1: User interface file for Qflow



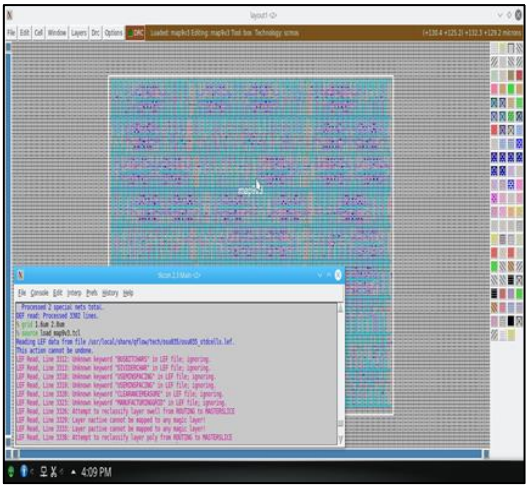
```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>Form</class>
 <widget class="QWidget" name="Form">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>152</width>
    <height>347</height>
   </rect>
  </property>
  <property name="font">
   <font>
    <pointsize>16</pointsize>
    <weight>75</weight>
    <italic>false</italic>
    <bold>true</bold>
   </font>
  </property>
  <property name="windowTitle">
   <string>Form</string>
  </property>
  <layout class="QGridLayout" name="gridLayout">
   <item row="0" column="0">
    <layout class="QVBoxLayout" name="verticalLayout">
     <item>
      <widget class="QLabel" name="label">
       <property name="text">
        <string>file</string>
       </property>
       <property name="alignment">
```

Fig. 6.2: Python file for Qflow


layout1
File Edit Cell Window Layers Drc Options
DRC
Loaded: (UNNAMED) Editing: (UNNAMED) Tool: box Technology: scmos
BUFX2
BUFX2_19
DFFSR
tkcon 2.3 Main
File Console Edit Interp Prefs History Help
4:08 PM

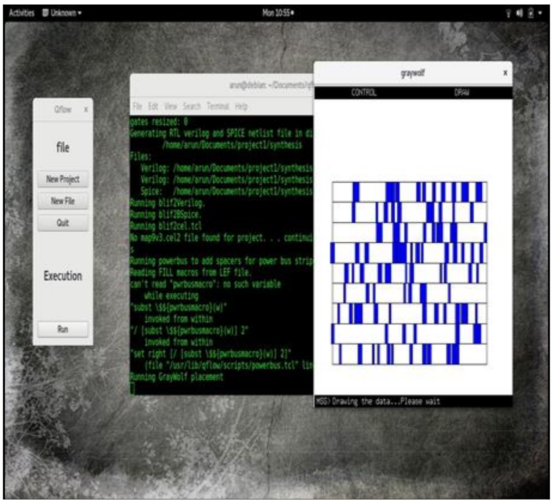
```
LEF Read, Line 2586: Unknown keyword "SHAPE" in LEF file; ignoring.
LEF Read, Line 2611: Unknown keyword "SHAPE" in LEF file; ignoring.
LEF Read:  Further errors will not be reported.
LEF read: Processed 3179 lines.
LEF Read: encountered 106 errors total.
% def read map9v3.def
Reading DEF data from file map9v3.def.
This action cannot be undone.
  Processed 282 subcell instances total.
  Processed 40 pins total.
  Processed 229 nets total.
  Processed 27 special nets total.
  Processed 2 special nets total.
DEF read: Processed 3302 lines.
% grid 1.6um 2.0um
% source
```

Fig. 6.3: Final layout for verilog file


layout1
File Edit Cell Window Layers Drc Options
DRC
Loaded: map9v3 Editing: map9v3 Tool: box Technology: scmos
map9v3
tkcon 2.3 Main
File Console Edit Interp Prefs History Help
4:09 PM

```
  Processed 2 special nets total.
DEF read: Processed 3302 lines.
% grid 1.6um 2.0um
% source load map9v3.tcl
Reading LEF data from file /usr/local/share/qflow/tech/osu035/osu035_stdcells.lef.
This action cannot be undone.
LEF Read, Line 3312: Unknown keyword "BUSBITCHARS" in LEF file; ignoring.
LEF Read, Line 3313: Unknown keyword "DIVIDERCHAR" in LEF file; ignoring.
LEF Read, Line 3318: Unknown keyword "USEMINSPACING" in LEF file; ignoring.
LEF Read, Line 3319: Unknown keyword "USEMINSPACING" in LEF file; ignoring.
LEF Read, Line 3320: Unknown keyword "CLEARANCEMEASURE" in LEF file; ignoring.
LEF Read, Line 3321: Unknown keyword "MANUFACTURINGGRID" in LEF file; ignoring.
LEF Read, Line 3326: Attempt to reclassify layer nwell from ROUTING to MASTERSLICE
LEF Read, Line 3329: Layer nactive cannot be mapped to any magic layer!
LEF Read, Line 3333: Layer pactive cannot be mapped to any magic layer!
LEF Read, Line 3336: Attempt to reclassify layer poly from ROUTING to MASTERSLICE
```

Fig. 6.4: Qflow execution using GUI widget



```
gates resized: 0
Generating RTL verilog and SPICE netlist file in di
        /home/arun/Documents/project1/synthesis
Files:
   Verilog: /home/arun/Documents/project1/synthesis
   Verilog: /home/arun/Documents/project1/synthesis
   Spice:   /home/arun/Documents/project1/synthesis
Running blif2Verilog.
Running blif2BSpice.
Running blif2cel.tcl
No map9v3.cel2 file found for project. . . continui
s
Running powerbus to add spacers for power bus strip
Reading FILL macros from LEF file.
can't read "pwrbusmacro": no such variable
    while executing
"subst \$${pwrbusmacro}(w)"
    invoked from within
"/ [subst \$${pwrbusmacro}(w)] 2"
    invoked from within
"set right [/ [subst \$${pwrbusmacro}(w)] 2]"
    (file "/usr/lib/qflow/scripts/powerbus.tcl" lin
Running GrayWolf placement
```

```
MSG> Drawing the data...Please wait
```

Fig. 6.5: Installer execution for Qflow

References

[1] Azzini I, Muresano R & Ratto M, “Dragonfly: a multi-platform parallel toolbox for MATLAB/Octave”, *Computer Languages, Systems & Structures*, Vol.52, (2018), pp.21-42.

[2] Zabołotny WM, “Version control friendly project management system for FPGA designs”, *Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments*, (2016).

[3] Scholz A & Juang JN, “Toward open source CubeSat design”, *Acta astronáutica*, Vol.115, (2015), pp.384-392.

[4] Montero MA & Salvadeo PA, “OpenCores® a robust complex network”, *7th Argentine School of Micro-Nanoelectronics, Technology and Applications (EAMTA)*, 2013, 15-21.

[5] Abid F, Izeboudjen N, Sahli L, Lazib D, Titri S, Louiz F & Bakiri M, “Opencores based Embedded System on Chip for Network Applications”, *Proceedings of the International Conference on Circuits, Systems, Signals*, pp.191-196.

[6] Bolado M, Posadas H, Castillo J, Huerta P, Sanchez P, Sánchez C, Fouren H & Blasco F, “Platform based on open-source cores for industrial applications”, *Proceedings Design, Automation and Test in Europe Conference and Exhibition*, 2004, pp.1014-1019.

[7] Murali A & Hari Kishore K, “Efficient and high speed key-independent AES-based authenticated encryption architecture using FPGAs”, *International Journal of Engineering & Technology*, (2017).

[8] Koch D, Beckhoff C, Wold A & Torresen J, “EasyPR-An easy usable open-source PR system”, *International Conference on Field-Programmable Technology*, (2013), pp.414-417.

[9] Beagam KSH, Jayashree R & Khan MA, “A new DC power flow model for Q flow analysis for use in reactive power market”, *Engineering Science and Technology, an International Journal*, Vol.20, No.2,(2017), pp.721-729.

[10] Sinha V, Hautvast G, Sonnemans J, de Bliek H, Jalba A & Breeuwer M, “Fast and easy visualization of blood flow patterns in 4D Qflow MRI”, *Journal of Cardiovascular Magnetic Resonance*, Vol.14, No.1,(2012).