CENG 491

DIGIMOD
REQUIREMENT ANALYSYS
REPORT

GROUP NAME: **MANAS YAZILIM**

GROUP MEMBERS: **Gökhan ÇAPAR**

Hüseyin B. AYYILDIZ

A. Serhat DEMİR

Seltchouk AHMED

1. INTRODUCTION

1.1 Problem Definition and Project Scope

Logic is one of the oldest methodologies in human life; the foundation of logic is human mind. With development and evolution of today's popular matter, computer science, logic has become a fundamental discipline in science.

Since computers entered in to human life, they have facilitated a lot of aspect of life. One of the powerful and beneficial technologies, which computer exposed, is simulations. By the aid of simulators, we do not need to waste time and money for tests and possible failures. For example, with simulators, before using real jets, we educate fighter pilots without taking any risk. On the other hand, we can simulate a production process and detect our errors without having products with faulty. Briefly, simulators save people's lives, money and time.

Logic has become more complicated with evaluation of technology and as a result it became dependent to computers in several manners. Hence, simulating logic in computer becomes a must if we think about sophisticated chips and circuits with thousands of gates.

Our aim is to develop a Digital Circuit Simulator program that simulates behaviors of digital circuits. There are a lot of FPGA (Field Programmable Gate Array) CAD tools, however, from our point of view they are incomplete or hard to use. In this project we will try to combine positive aspects of these programs and produce a new one with all the required features to design a digital circuit and test it. Of course with a user friendly, not too sophisticated interface.

If it is necessary to mention about what will be the features of our product, we can summarize them as follows.

- It's usage will be easy and intuitive
- Having a fascinating GUI
- Supports user defined circuit components. (Macros)
- Capable of indicating errors in the circuit.
- Cut / copy / paste feature
- Zoom in /zoom out feature
- Drag / drop feature
- Import .LGF files
- Two alternative of printing (logic symbols or circuit diagram)
- Control circuits with scripting support.

2. DEFINITION of CURRENT SYSTEM SURVEY

After determination of which project we will develop, we have examined many FPGA CAD tools. There are a lot of programs which are open-source, freeware, shareware or commercial on the web. You can find brief information abut principal a few ones of them in the following pages. We are already familiar with DIGLOG since we used it in the CENG232 Logic Design course.

We tried to understand how they work, how they used and which features they have. While doing this, first we tried to use programs only looking at GUI and understood how user friendly they are. After that we have looked in more details and read help files, of course when available. This survey helped us to determine which features we will include in our project. What must be in the program and what must not.

In addition to existing finished projects, we found a lot of libraries for various programming languages that we can use in our project. The most common ones are Hardware Description Languages' for different programming languages. Since we decided to use Java we have looked in details to JHDL. The only bottleneck for using JHDL, it does not support asynchronous digital circuits for now. On the other hand we will need graphic libraries to develop a circuit design program. Since our program will support only circuits that can be drawn in two dimensions, Java 2D seems to be enough.

2.1 Literature Survey and Technical Analysis

Multisim

Muiltisim is a very developed and high priced program that supports both analog and digital circuits. Most interesting feature of this program is macros. You can define a circuit as a component so that you can use it in more complex circuits. Since it is a big program, it is difficult to learn and use the program without help. User interface should be simpler. On the other hand it has a well designed simulator. This program seems to be one of our important opponents.

Design Works

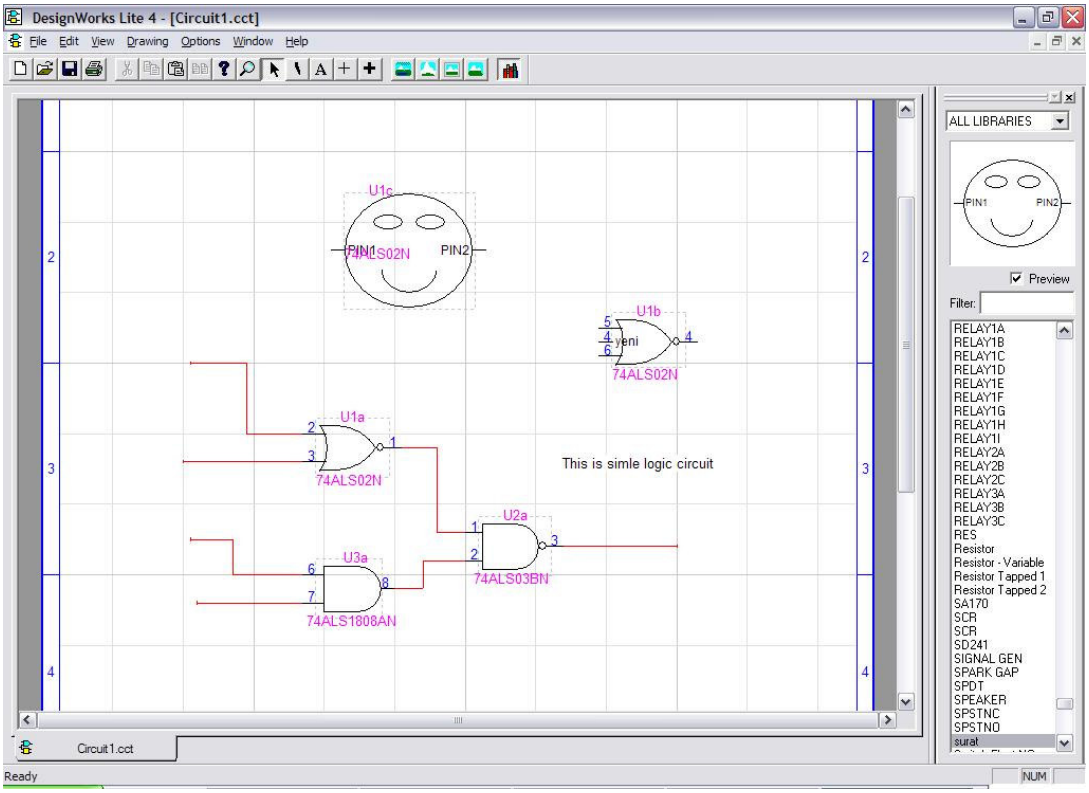
This program does not overlap with DIGLOG for several manners and attributes but seems similar systematically so we decided to examine this program will be helpful. Design Works Lite 4 has a simple interface to design except it is logic and circuit elements choices. The list of elements is on the right side o the interface and it seems complicated and redundant to put the full list.

General appearance of the program is very simple as we mentioned before has a bad GUI. It has standard features like save, copy, cut, import...etc.

One of the good features is you can create a component and add it to library and read it later from the library of course.

Zoom in/out is another advantage of this program.

The most conspicuous disadvantage of the program is that it does not have a simulation part.



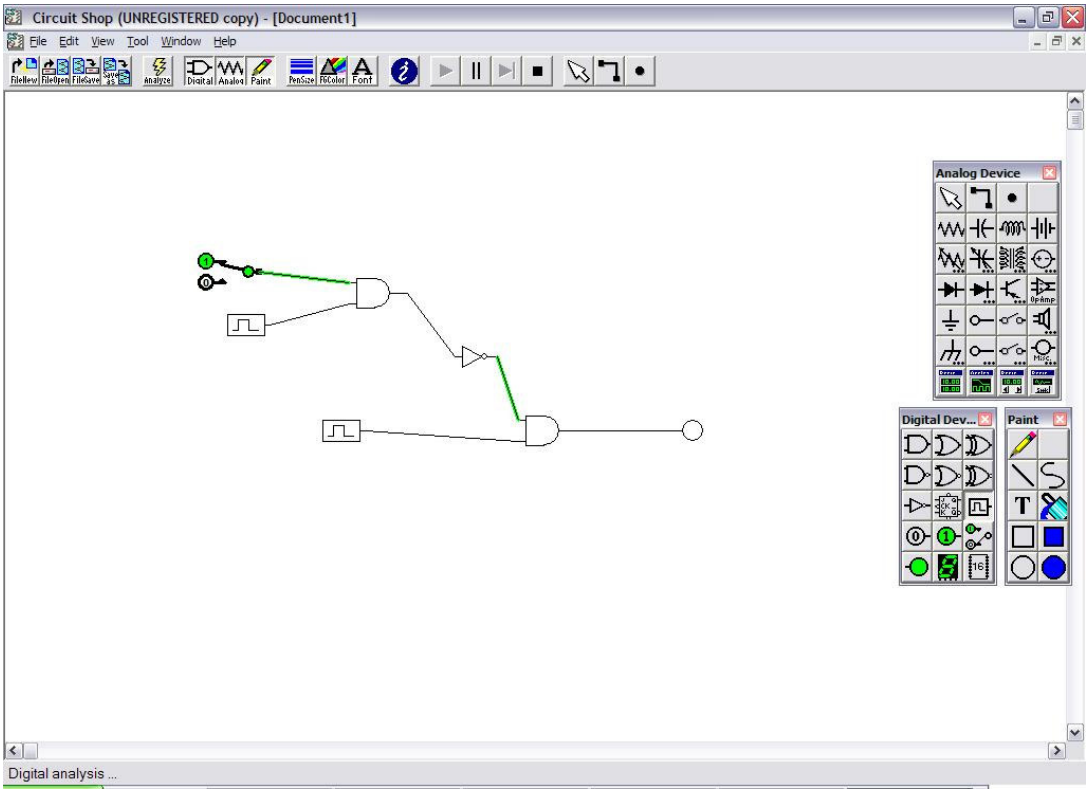
Circuit Shop

This program has an extremely simple GUI which consists of the white layer and some elements to be used in design. It has basic features like print and print preview. It lets you to create elements; we also want to implement this feature, and editing the interface.

Program exports to .bmp file which is a widely used format and easily configurable. Furthermore this program is only 1.8 Mb, which is very small indeed. Program also has an analysis attribute which are Digital Analysis and Analog Analysis which sounds good.

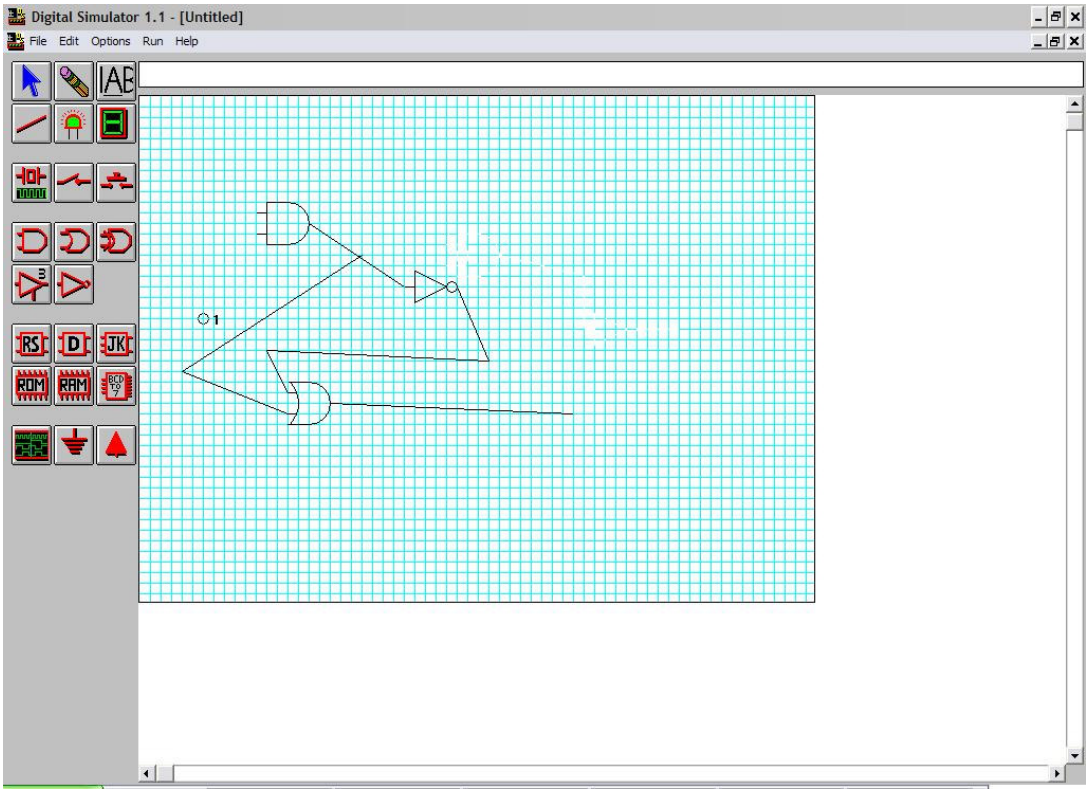
On the other hand this program is not a freeware and you have to loose money or time in order to use this program properly for a long time.

Unfortunately this program has no zoom in/out feature which will put you in a bad situation if you are in a big project.



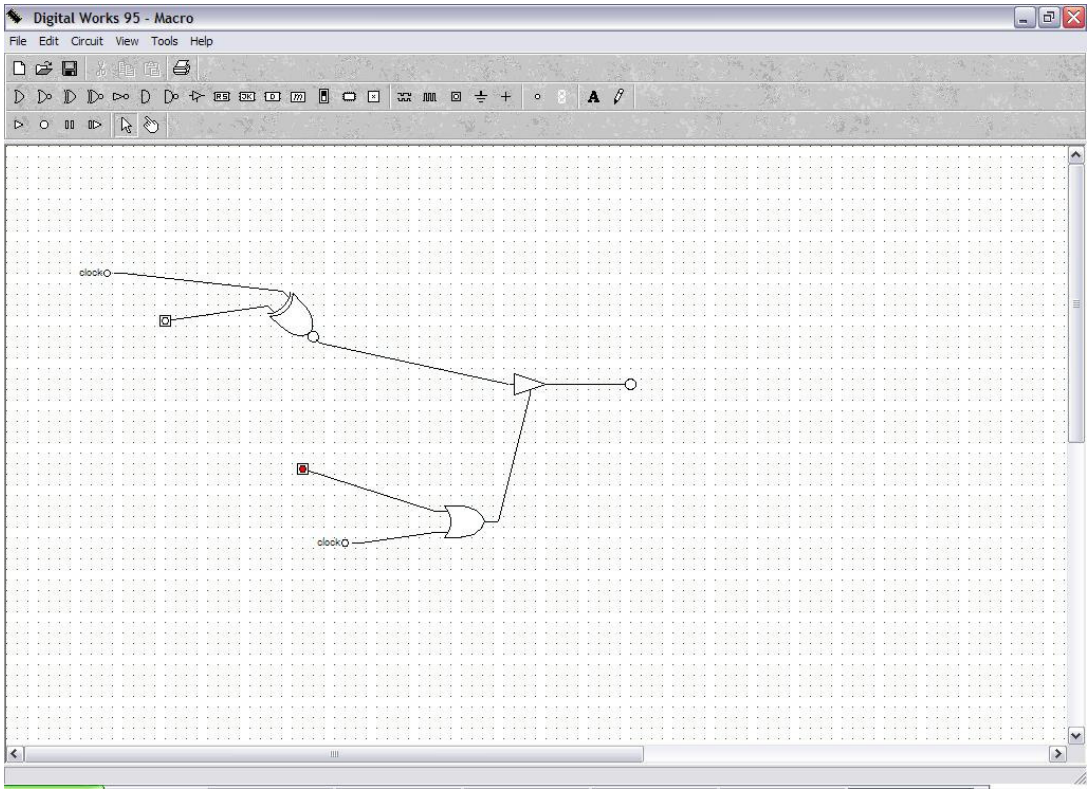
Digital Simulator

This is the simplest program that we have investigated. It has a really bad GUI. Only a few digital components are available. When you erase an element also grids are erased and you have to press redraw button to make grids look normal again. It is waste of time to use this program as a design tool.



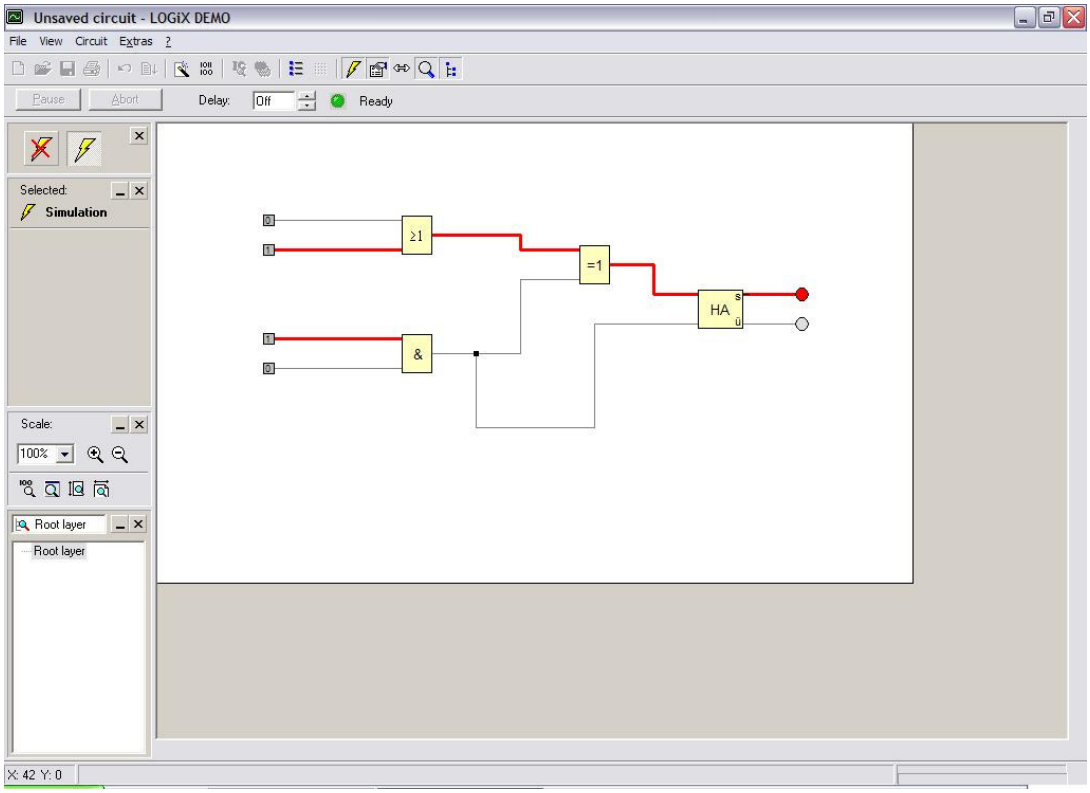
Digital Works

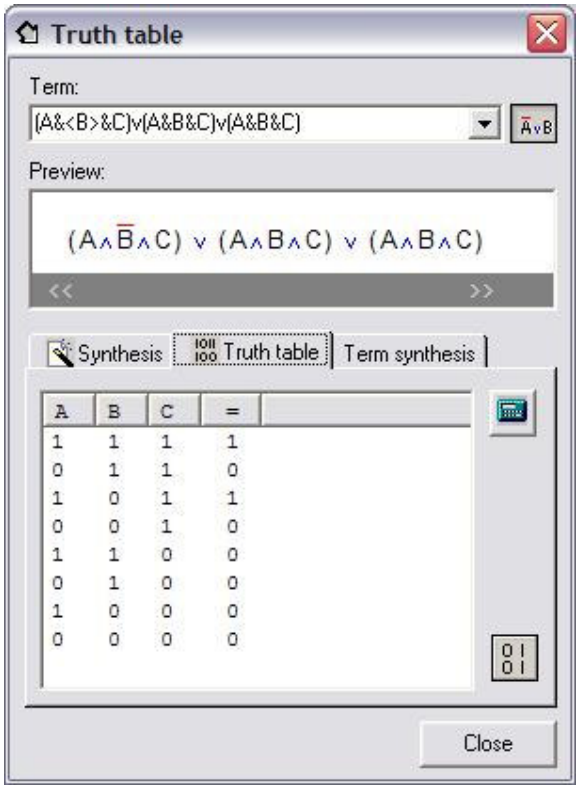
Program has a simple GUI. It has a small library, however it allows creating and saving new circuit components as macros. It can detect faults such as race conditions and bus contention. It has multiple input supports up to four inputs for logical gates. The worst thing about program is you can not zoom to your drawing and moving elements is very limited.

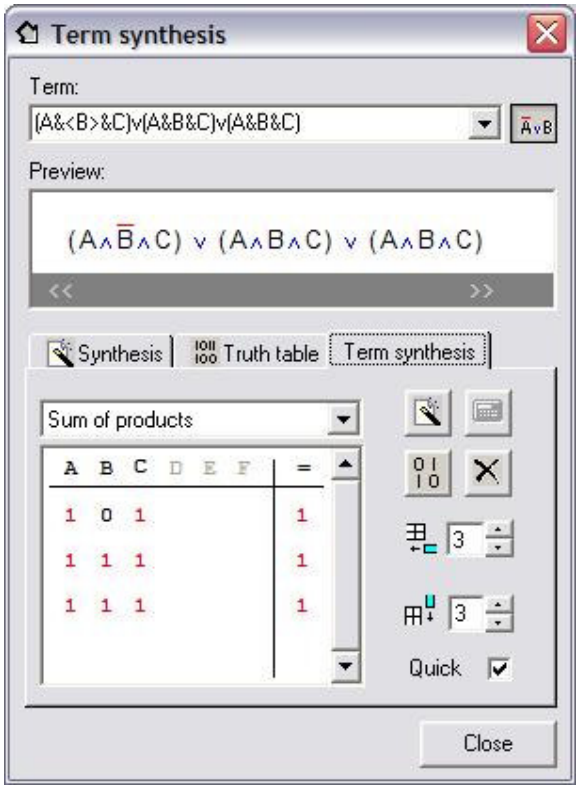


Logix

User interface is too simple. This is due to lack in library support. Program does not have macros. It does not contain import/export support also. There are some positive features that can be very useful. First of them is IC element support. The program dos not have extendable library but can import IC element library. Another characteristic is generation circuit from Boolean term. Truth table calculation and creating Boolean terms are extra properties for circuit generation and verify the correctness of circuit. Furthermore, the program has very good zoom properties.


Unsaved circuit - LOGiX DEMO
File View Circuit Extras ?
Pause Abort Delay: Off Ready
Selected:
Simulation
Scale:
100%
Root layer
Root layer
≥1
=1
&
HA
s
ü
X: 42 Y: 0


Truth table
Term:
(A&<B>&C)v(A&B&C)v(A&B&C)
Preview:
(A∧B̄∧C) ∨ (A∧B∧C) ∨ (A∧B∧C)
Synthesis
Truth table
Term synthesis
A B C =
1 1 1 1
0 1 1 0
1 0 1 1
0 0 1 0
1 1 0 0
0 1 0 0
1 0 0 0
0 0 0 0
Close

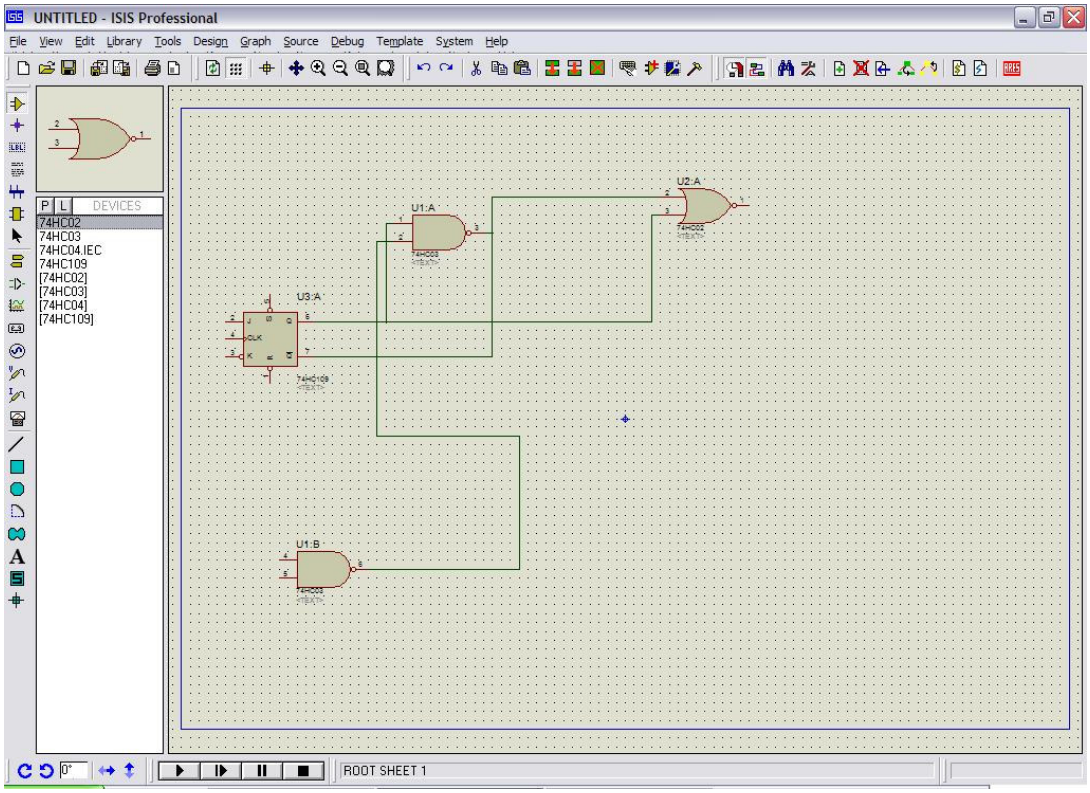


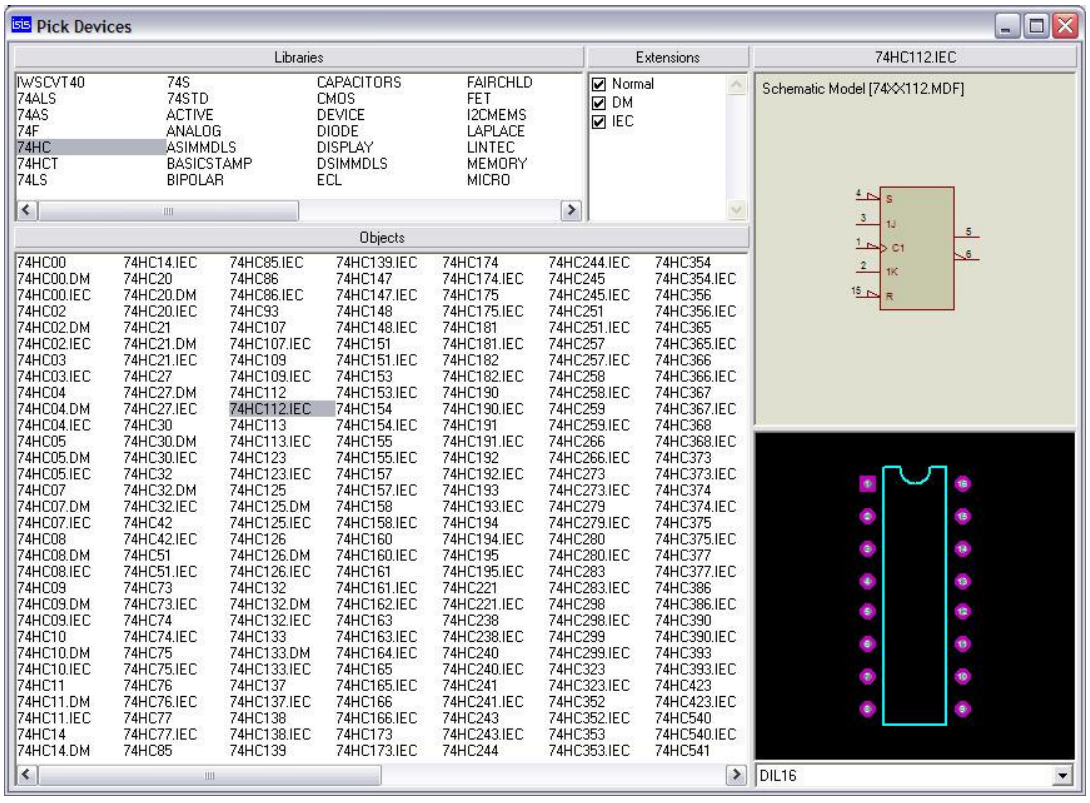
Proteus Pro

The best feature of this program is very large support of library. Making new device, editing, creating library and variety operations on elements and libraries… Certainly this wide variety of options among libraries leads to some confusions. Graphical user interface is quite complicated.

There are many supported features. Some of them are macros, error detection, zoom (delete, move, rotate are also quite simple), print support, import/export support.

The program is not very easy to use and this is the only and the most important disadvantage. However, apart from this we saw that it is one of the best in this market. Moreover, we think that the program most possible have script support, but we saw it accidentally. There is no direct way on the interface that shows it.


UNTITLED - ISIS Professional
File View Edit Library Tools Design Graph Source Debug Template System Help
P L DEVICES
74HC02
74HC03
74HC04.IEC
74HC109
[74HC02]
[74HC03]
[74HC04]
[74HC109]
U1:A
U2:A
U3:A
U1:B
ROOT SHEET 1


Pick Devices
Libraries
Extensions
74HC112.IEC
IWSCVT40 74S CAPACITORS FAIRCHLD
74ALS 74STD CMOS FET
74AS ACTIVE DEVICE I2CMEMS
74F ANALOG DIODE LAPLACE
74HC ASIMMDLS DISPLAY LINTEC
74HCT BASICSTAMP DSIMMDLS MEMORY
74LS BIPOLAR ECL MICRO
Normal
DM
IEC
Schematic Model [74XX112.MDF]
Objects
74HC00 74HC14.IEC 74HC85.IEC 74HC139.IEC 74HC174 74HC244.IEC 74HC354
74HC00.DM 74HC20 74HC86 74HC147 74HC174.IEC 74HC245 74HC354.IEC
74HC00.IEC 74HC20.DM 74HC86.IEC 74HC147.IEC 74HC175 74HC245.IEC 74HC356
74HC02 74HC20.IEC 74HC93 74HC148 74HC175.IEC 74HC251 74HC356.IEC
74HC02.DM 74HC21 74HC107 74HC148.IEC 74HC181 74HC251.IEC 74HC365
74HC02.IEC 74HC21.DM 74HC107.IEC 74HC151 74HC181.IEC 74HC257 74HC365.IEC
74HC03 74HC21.IEC 74HC109 74HC151.IEC 74HC182 74HC257.IEC 74HC366
74HC03.IEC 74HC27 74HC109.IEC 74HC153 74HC182.IEC 74HC258 74HC366.IEC
74HC04 74HC27.DM 74HC112 74HC153.IEC 74HC190 74HC258.IEC 74HC367
74HC04.DM 74HC27.IEC 74HC112.IEC 74HC154 74HC190.IEC 74HC259 74HC367.IEC
74HC04.IEC 74HC30 74HC113 74HC154.IEC 74HC191 74HC259.IEC 74HC368
74HC05 74HC30.DM 74HC113.IEC 74HC155 74HC191.IEC 74HC266 74HC368.IEC
74HC05.DM 74HC30.IEC 74HC123 74HC155.IEC 74HC192 74HC266.IEC 74HC373
74HC05.IEC 74HC32 74HC123.IEC 74HC157 74HC192.IEC 74HC273 74HC373.IEC
74HC07 74HC32.DM 74HC125 74HC157.IEC 74HC193 74HC273.IEC 74HC374
74HC07.DM 74HC32.IEC 74HC125.DM 74HC158 74HC193.IEC 74HC279 74HC374.IEC
74HC07.IEC 74HC42 74HC125.IEC 74HC158.IEC 74HC194 74HC279.IEC 74HC375
74HC08 74HC42.IEC 74HC126 74HC160 74HC194.IEC 74HC280 74HC375.IEC
74HC08.DM 74HC51 74HC126.DM 74HC160.IEC 74HC195 74HC280.IEC 74HC377
74HC08.IEC 74HC51.IEC 74HC126.IEC 74HC161 74HC195.IEC 74HC283 74HC377.IEC
74HC09 74HC73 74HC132 74HC161.IEC 74HC221 74HC283.IEC 74HC386
74HC09.DM 74HC73.IEC 74HC132.DM 74HC162.IEC 74HC221.IEC 74HC298 74HC386.IEC
74HC09.IEC 74HC74 74HC132.IEC 74HC163 74HC238 74HC298.IEC 74HC390
74HC10 74HC74.IEC 74HC133 74HC163.IEC 74HC238.IEC 74HC299 74HC390.IEC
74HC10.DM 74HC75 74HC133.DM 74HC164.IEC 74HC240 74HC299.IEC 74HC393
74HC10.IEC 74HC75.IEC 74HC133.IEC 74HC165 74HC240.IEC 74HC323 74HC393.IEC
74HC11 74HC76 74HC137 74HC165.IEC 74HC241 74HC323.IEC 74HC423
74HC11.DM 74HC76.IEC 74HC137.IEC 74HC166 74HC241.IEC 74HC352 74HC423.IEC
74HC11.IEC 74HC77 74HC138 74HC166.IEC 74HC243 74HC352.IEC 74HC540
74HC14 74HC77.IEC 74HC138.IEC 74HC173 74HC243.IEC 74HC353 74HC540.IEC
74HC14.DM 74HC85 74HC139 74HC173.IEC 74HC244 74HC353.IEC 74HC541
DIL16

2.2 Comparison Table

	Digital Works	**Multisim**	**Design Works**	**Circuit Shop**	**Digital Simulator**	**Diglog**	**Logix**	**Proteus Pro**
User Interface	Good	Fine	Fine	Fine	Bad	Bad	Good	Fine
Live Components	+	-	+	+	-	-	-	+
Macros	+	+	+	-	-	-	-	+
Error detection	+	+	+	-	-	+	-	+
Zoom	-	+	+	+	-	+	+	+
Print support	+	+	+	+	-	+	+	+
Virtual wiring	+	+	-	-	-	+	-	+
Scripting support	-	-	-	-	-	-	-	+
Import/Export	-	-	+	+	-	+	-	+
Library	Small	Large	Large	Medium	Small	Large	Small	Very large

2.3. Customer Meeting

Why do we need a Customer Meeting?

Digital circuit design tools are mostly used in computer science and electronics engineering area. Since such tools are used often, CS and EEE students should know about advantages and disadvantages, positive and negative aspects of these tools. What they like and do not like in such tools is, there are certain features that they want to see in the tool. Because of this opinion, we decide to have an interview with some student of computer science and electrical and electronics engineering department. We believe that such kind of information will be very helpful during the design stage of the project.

Interviewing with the customers face to face is helpful for getting any information more efficiently. You can ask whatever you want directly to the customer and can get his/her needs. However, this way has a disadvantage that in such a way you cannot speak a lot of people. Because of this problem we decided to have a meeting with several people at the same time. This gave us a platform for exchanging ideas.

Interview with the Customers

Due to our limited time interval we could only have a meeting with our friends in computer science and electrical and electronics engineering department. During our meeting we ask about what kind of circuit simulator tools they use and what kind of opinions they have about them. In general speaking they said that they were currently using Electronic Workbench and Proteus Pro. Both of them are pretty big programs that are used in design and simulation of analog circuits also. Since our project is based on digital circuits we do not have such an operational feature on analog circuits.

These students told us about problems of such kind of programs. We realized there are many problems even in so big and widely used programs. Engineering students complained about some export features and printing features of the programs. The most important idea that hey liked was scripting support. They said that this characteristic will ease their job to construct and test the circuits they develop.

The second feature they liked was library support. To compact designed circuits and add them to the library to use another time is very good support they said. This reusability of plotted circuit may be very helpful for large projects to reduce conflicts.

Another feature we will integrate to the program is waveform viewer that allows users to see the wire status at specific time. During the conversation with the student they said that this is very critical for designing circuits.

In conclusion, customer satisfaction is our first priority. To provide this demand we had a meeting with our potential customers and made exchange of ideas. This was very helpful for us since we form our plans and ideas to specify our goal. After getting basic structure of the deficiency we were able to focus on what is wanted from us.

2.4. Project Features

After examining similar programs and interviewing with users who use these programs, we were sure about features that are missing in other tools. We categorized these features in eleven parts:

1.Graphical user Interface: For easy using and easy learning this is very important. We will make clear and understandable user interface.

2.Live Components: After constructing circuit, edit stage comes usually. Then you may need to move or rotate the elements. This feature became important at that stage. If there is no such an option it will be quite difficult for user. This feature force element stay connected to wires wherever they are moved or rotated.

3.Macros: It is extremely efficient if you want to reuse your developed circuit as a circuit element in another circuit simulation. Our program will allow users to add their own circuits to the library as a circuit component. Further, there will be a circuit component editor to design such small components that will be supported by our scripting feature.

4.Error Detection: This feature will help developers to find errors and easier their job.

5.Zoom: This characteristic is important for going to details and viewing the general circuit during development.

6.Cut/Copy/Paste: This is necessary feature. After all we use this very often.

7.Print Support: Our program will be able to print projects in three categories. First one is printing the entire project exactly as it is. The second one is pin layout which groups gates and prints them as integrated circuits. This is helpful during designing circuits in real world. Because there are no And gates, there are integrated circuits. The third part is PDF printing. This will support previous two.

8.Virtual Wiring: To avoid complexity and confusion in circuits this is an elegant solution. There is a "from" box that sends signal and a "to" box that receives. Users will use them to connect elements instead of using wires.

9.Scripting Support: Using scripts will make users job easier. First of all, since our program will design digital circuits, there is no drawback to give a logic formula and make the program to plot circuit itself. By using scripts this will be possible. Secondly, in very complex circuits if you want to test them this will cause some problems. Without knowing anything about the circuit you can give array of inputs and get their outputs easily. Besides these you will be able to intervene the circuit during simulation.

10. Import/Export: Our program will support diglog .lgf files. It will be able to import them in the projects and export the projects to .lgf format. Moreover, we will add support for exporting as PDF format.

11. Library: This is one of the most important features in the program. Our tool will have an extendable library. In this library users will be able to add circuit elements, categorize them, edit, and delete them In addition, there will be a macro support that allows users to use their own circuits as components.

3. PROJECT SCHEDULE

3.1 Gantt Chart

TASKS	10/10-16/10	17/10-23/10	24/10-30/10	31/10-06/11	28/11-04/12	16/12-22/12
Project Proposal	■					
Field Survey	■					
User Research		■				
HDL Research		■	■			
LGF Research		■	■	■		
Script Research		■	■			
Print Support Research		■				
Platform Independency Research			■	■		
Requirement Analysis Report				■		
GUI Design						
Element Design						
Initial Design Report					■	
Prototype Demo						■
HDL Implementation						
GUI Implementation						
Editor Implementation						
Library Implementation						
Simulator Implementation						
Script Implementation						
Print Support Implementation						
LGF Support Implementation						
Project Completion						

4. PROJECT REQUIREMENTS

4.1. Functional Requirements

In order to determine the needs of the project, functional requirements must be clearly identified. With these specifications and obeying the borders, project will be pure during the process.

With DIGIMOD user will have the opportunity to use following modules:

4.1.1 Save/Open files

Current modifications and new files can be saved in to our file system and put in to the hard drive segment which is specified by user. Files in the drives can be opened with open facility and it will b able t continue to work on that file.

4.1.2 Import/Export LGF files

DIGLOG is one of the most popular “free” digital logic simulator in computer science and widely used in computer science departments of universities, which also holds for our department, computer engineering at METU.

In order to make DIGIMOD a widely used program, we want to import LGF files in to our system. Modifying files and saving it in to LFG files is another facility to help this mission.

4.1.3 Script support

Script support will ease users’ life and save time. User can test the system by giving inputs and script will take these inputs to system and return the outputs to user.

Another good characteristic is that script can convert Boolean functions in to circuits.

4.1.4 Macro creating and using

Often used circuits can be saved in to a macro in order to use it in other projects. Macros will be stored in library and exported from library as a regular gate.

4.1.5 Error detection

DIGIMOD will detect errors in circuit and warn users by changing colors of lines and gates.

4.1.6 Editor Features

Editor will have the zoom in and zoom out features in order to handle huge circuits or small circuits.

Gates, lines, macros, full or part of circuits can be cut/copy/paste with the editor. Mouse and keyboard may be used to make these operations.

4.1.7 Print support

DIGIMOD have several printing choices. User can convert the circuit to PS or PDF files as he/she wishes. Another choice is that user can directly send current project in editor to printer. Finally pin layouts of circuits can be sent to printer.

4.1.8 Virtual wiring

Labeling the wires is a good way to simplify large projects. “From” and “to” boxes are used in order to manage this feature. From and to boxes have the name of wires so putting complicated and long lines will be unnecessary.

4.1.9 Extendable library

User can add and remove elements to library.

4.2. Non-Functional Requirements

4.2.1. Usability & Interface

We think that, beyond everything, if a software product is not user friendly, it is a failure. So our modules should be easy to understand and use. We experienced that lots of digital logic simulators are not user friendly, contrary to these programs, our users will feel comfortable while using DIGIMOD. Color choice must be made carefully in order not to exhaust users' eyes and buttons must be located effectively to make the interface simpler and easy to manage. Live components facility which moves extensions of a gate or macro will help to rotate copy/paste and move gates. To shorten the adaptation and problem solving time we want to put a tutorial and a FAQ part in DIGIMOD.

4.2.2. Reliability

We want DIGIMOD to be used by students and academicians in logic courses as well as amateur users who are interested in logic. Hence our product must be as bug-free as possible because a corruption in the product which yields data loss will dramatically affect users. Stability of the product will encourage potential users to be DIGIMOD users.

4.2.3. Portability

DIGIMOD will work in both Windows and Linux platforms.

4.3. Software Requirements

We decided to develop our project with Java therefore Borland Java Builder will be our development environment. HDL, Jython, Microsoft Visio, MS Project are other software we are using or willing to use.

DIGIMOD will be platform independent software; as a result it will run in both MS Windows 98/2000/XP or Linux operating systems.

4.4 Hardware Requirements

An IBM compatible PC which has 500 MHz processor, 128 Mb memory, 500 Mb free disk space will be enough to execute DIGIMOD.

5. RISK MANAGEMENT PLAN

In software projects there are always risks that may occur. The question is "what we will do when risks arise?" Not "will any risk occur?". We can only handle this with a good risk management plan. Therefore, for the safety and feasibility of the project, extensive risk management will be carried out through the project life cycle. As an umbrella activity, risk management will evolve and update itself within the project.

5.1 Team Management

- Lack of roles and responsibility: Team members need to understand their own roles and responsibilities and those of others. So duplication of the same work and deficiencies prohibited.
- Misunderstanding or Lack of Standards: Our group consists of 4 people. We will work in parallel on different parts of the project. These parts must communicate each other. Thus standardization is very important.
- Incomplete Project Objectives: We must determine very well which work will be done and who will do clearly.

5.2 Project Parameters

- Changing Requirements: If we do not determine what we need clearly, than lots of code modification may be required.
- Lack of time: Since our time is limited, obeying the schedule is important. Of course we can not employ any programmers if we didn't finish work on time.
- Design Difficulty: Our project requirements are well defined, so our design will not be too difficult.
- Implementation: Our team is experienced coders. Only a few sophisticated algorithms may force us.

5.3 Project Team

- Unavailability of member(s): Since team members are in a hot pursue with their current semester courses, unavailability is a major risk. Especially in the implementation part all members are required to finish the project on time.

5.4 Technology

- Inappropriate Technology: Technology that will be used must be chosen very carefully. Because if it is deficient to meet all needs, than the team may have no chance to change the used technology due to time limitations.

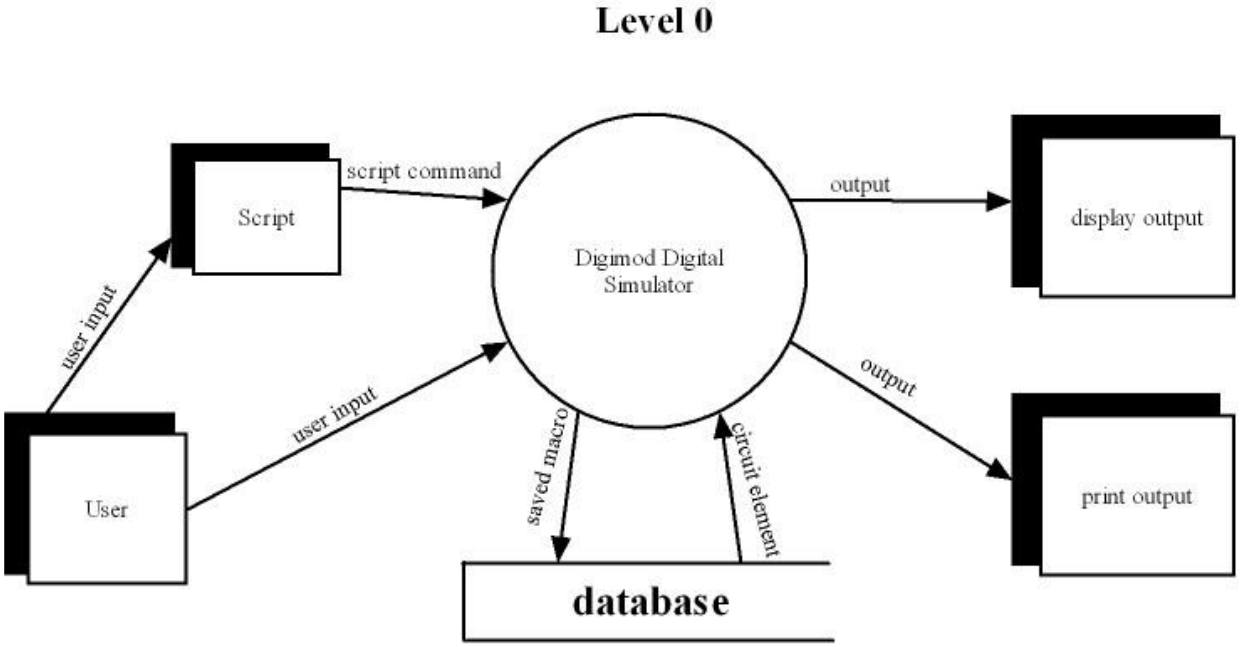
5.5 Risk Table

Risk	Probability (%)	Impact
Lack of roles and responsibility	20	Negligible
Misunderstanding or Lack of Standards	40	Marginal
Incomplete Project Objectives	10	Critical
Changing Requirements	10	Marginal
Lack of time	50	Critical
Design Difficulty	10	Negligible
Implementation Difficulty	15	Negligible
Unavailability of member(s)	70	Critical
Inappropriate Technology	5	Catastrophic

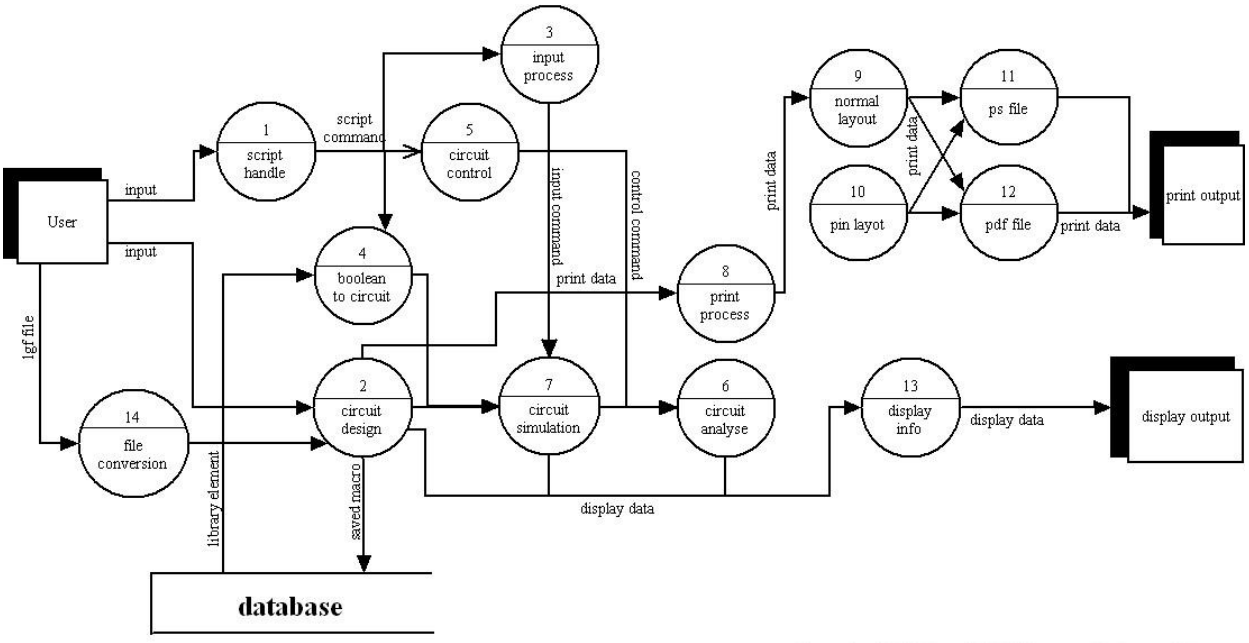
6. MODELING

6.1 Functional Model

6.1.1. Level 0 DFD



6.1.2. Level 1 DFD



6.2. Component Description

PSPECS – The Process Specifications

PSPEC: script handle

In our program, users should be able to control the program using a script. Our script will handle three different situations. It will take inputs and send it to the program to display output of the circuit, it will accept a Boolean function and convert it to a designed circuit, and it will control the program during simulation and analysis of a circuit.

PSPEC: circuit design

Since our program is a circuit simulator. It will also allow users to design their circuits by hand, using the built in elements in the library. After a new circuit is designed, it can be added as a new element to the library so that it will avoid complexity of the bigger designs and it will be available for future use.

PSPEC: input process

This is an important feature for our program. Program will accept inputs with the help of the script and users will access outputs without running the program. This will save a lot of time during processing of large number of files.

PSPEC: circuit control

During simulation and analysis of a circuit, users may want to change inputs or state of the circuit. This will be also handled in our program so that users can modify and see circuit behavior while the program is running.

PSPEC: Boolean to circuit

Our program will be able to read Boolean functions and turn them into logic circuits.

PSPEC: circuit simulation

After designing or importing a circuit, users may simulate their designs to see if it works correctly.

PSPEC: circuit analysis

This feature will help users to find their errors and also see the behavior of the circuit. We will monitor what is going on during simulation using graphs.

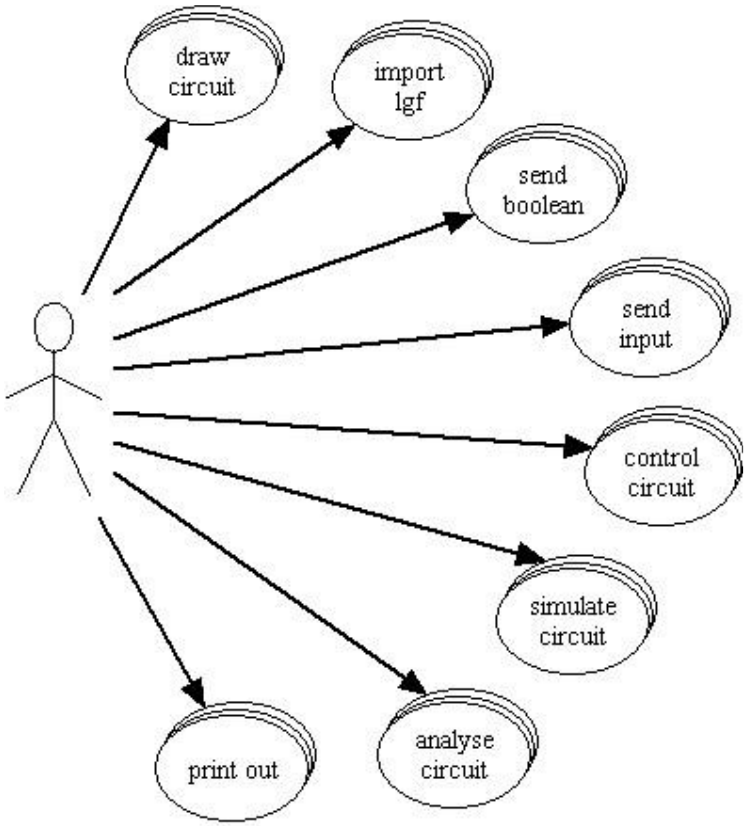
PSPEC: print process

We will have two main types of print support. First is normal layout and the second one is pin layout that is printing the circuit using actual chips with their pin layouts. Users can have both PS and PDF file formats available for print outs.

PSPEC: file conversion

Our program will support also diglog file format. It will import LGF files and process on their circuits.

6.3 Use case diagram



7. SOFTWARE QUALITY ASSURANCE

DIGIMOD will be huge software compared to our other course projects with several manners. There will be thousand lines of codes, hundreds of items to create circuits and several large table and graphics. In order not to swerve from our basic project we have to practice a software quality assurance plan.

We determined several criteria to provide this assurance process. After each milestones or important updates, we will control aspects of both code and program according to requirement specifications.

We will use Concurrent Version System (CVS) to have full control in code and allow developers to collaborate.

Severe modifications and important changes will have to be accepted by majority of the team members in order to apply to project.