Xilinx FPGA Implementation OfArithmetic Logic Shift Unit (ALSU)

Rag had Z. Tawfiq*

Received on: 26 /10/2005
Accepted on: 2 /3 / 2006

Abstract

A Field Programmable Gate Array (FPGA) is a digital integrated circuit that can be programmed to do any type of digital function. This paper represents how to map the arithmetic logic shift unit (ALSU) to the Xilinx FPGA Chip.

The ALSU architecture was captured through the use of the VHDL hardware description language. Xilinx Integrated Software Environment ISE 6.i (project navigator) CAD software package was used to synthesize and implement the architecture of the ALSU to the FPGA chip. The architecture was functionally and performance validated through the package software simulation via post-synthesis and post-implementation software simulations.

Keywords: Arithmetic Logic Shift Unit (ALSU), Field Programmable Gate arrays (FPGA), VHDL

الخلاصة

أن رقاقة مصفوفات بوابات المجال المبرمج (FPGA) عبارة عن دائرة منطقية متكاملة يمكن ان تبرمج لتقوم بأداء أي دالة منطقية.أن هذا البحث يبين كيفية تمثيل وحدة الحساب و المنطق(ALSU) في رقاقة مصفوفات بوابات المجال المبرمج (FPGA) لشركة (Xilinx) للالكترونيات الدقيقة.

تم تصميم و اختبار وحدة الحساب و المنطق باستخدام لغة وصف التصميم باستخدام الدوائر الفائقة السرعة (VHDL) و ذلك بأستخدام برنامج (Active VHDL) 3.5 في حين تمت عملية التركيب و التنفيذ باستخدام برنامج(Xilinx Integrated Software Environment ISE 6.I (project navigator). تمت عملية محاكاة (Simulations) للتصميم بأستخدام نفس البرنامج وذلك لأختباره قبل تنفيذه .

* Dept. of Control and System Engineering, University Of Technology, Baghdad, Iraq.

1- Introduction

The hardware of the computer is usually divided into three major parts, the central processing unit (CPU) contains an arithmetic and logic unit for manipulating data, a number of registers for storing data, and a control circuits for fetching and executing instructions.

The arithmetic and logic unit (ALU) is the part of the computer that actually performs arithmetic and logic operations on data. All of the other elements of the computer system: control unit, register, memory, I/O are there mainly to bring data into the ALU for it to process and then to take the results back out [1].

The arithmetic logic unit is a combinational circuits, so that the entire register transfer operation from the source register through the arithmetic logic unit and into the destination register can be performed during one clock pulse period. The ALSU provides arithmetic, logic and shift operations [2].

2- Introduction To Vhdl

VHDL is a hardware description language for modeling digital circuits that can range from the simple connection of gates to complex systems. VHDL is an acronym of VHSIC Hardware Description Language, and VHSIC in turn is an acronym for Very High Speed Integrated Circuits [3, 4].

VHDL is designed to fill a number of needs in the design process.
Firstly, it allows description of the structure of a design that is how it is decomposed into sub-designs, and how those sub-designs are interconnected. Secondly, it allows the specification of the function of designs using familiar programming language forms. Thirdly, as a result, it allows a design to be simulated before being manufactured, so that designers can quickly compare alternatives and test for correctness without the delay and expense of hardware prototyping [5].

VHDL, in many respects, is similar to a regular computer programming language, such as C. for example, it has constructs for variable assignments, conditional statements, loops, and functions. In a computer program language, a compiler is used to translate the high-level source code to machine code. In VHDL, however, a synthesizer is used to translate the source code to a description of the actual hardware circuit that implements the code. Accurate functional and timing simulation of the code is also possible in order to test the correctness of the circuit [6].

3- Arithmatic Logic Shift Unit Design

In this section it is required to design an 8-bit ALSU that perform addition, subtraction, logic operations (Complement, AND, OR, XOR) and shift operation (shift to the right, and shift to the left).

Figure (1) shows a 1-bit arithmetic logic shift unit, inputs A_i , B_i are applied to both the arithmetic and logic units. A particular micro-operation is selected with inputs S_1 , S_0 . A 4×1 multiplexer at the output chooses between an arithmetic output in D_i and a logic output in E_i . The data in the multiplexer are selected with the inputs S_3 and S_2. The

other two data inputs to the multiplexer receive inputs A_{i-1} for the shift right operation and A_{i+1} for the shift left operation. The circuit of this figure must be repeated n times for an n-bit ALSU. The output carry C_{i+1} of a given arithmetic stage must be connected to the input carry C_i of the next stage in sequence. The input carry to the first stage is the input carry C_m which provides a selection variable for the arithmetic operations.

Table (1) lists the 14 operations of the ALSU. The first eight are arithmetic operation and are selected with S_3 S_2=00. The next four are logic operations and are selected with S_3S_2=01.The last two operations are shift operations and are selected with S_3 S_2=10 and 11.

Eight bit ALSU can be constructed using eight unit of the 1-bit ALSU shown in figure (1) by connecting the carry out from the first stage to the carry in of the second stage and defined the required signal to connect the shift right and shift left to each unit. Figure (2) show an 8-bit ALSU with two 8-bit inputs($A1$, $A2$.$A3$, $A4$, $A5$, $A6$, $A7$, $A8$, $B1$, $B2$, $B3$, $B4$, $B5$, $B6$, $B7$, $B8$) and four select lines (, S_0, S_1, S_2, S_3) and carry in input signal (Cin), shift left input (shl),shift right input (shr) and with 8-bit output (f1, f2, f3, f4, f5, f6, f7, f8) and the carry out signal (Co).

4-Xilinx Integrated Software Enviroment ISE 6.I (Project Navigator) Cad Software Package with Vhdl Program

The Field Programmable Gate Array (FPGA) is a type of programmable logic device. Programmable logic devices are a class of general purpose chips that can be configured for a wide verity of applications [7, 8].

Xilinx (Project Navigator) CAD software package is used to synthesize and implement the architecture of the ALSU to the FPGA chip.

4.1- The Design Step

1- Design Entry.
2- Design Simulation.
3- Design Synthesis.
4- Design Implementation

4.1.1- Design Entary

The design entry in the proposed work has been done using VHDL programming language. So, VHDL is used to create an entity that performs the required operation. When describing a circuit at this level, you would write basically the same thing as in the arithmetic description, except that you have to use the correct syntax required by the VHDL.

The description of a simple part of the program represent 1-bit of the Arithmetic Logic Shift Unit architecture using VHDL is shown bellow:

```
------ 1-bit alu -------LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY alubit IS PORT (
A111, B111, sl, sr, cinl, s01, s11, s21,
s31: IN
STD_LOGIC;
Tyco: OUT STD_LOGIC);
END alubit;
Architecture STRUCTURAL of alubit IS
COMPONENT and2gate IS PORT (
```

```
i1,i2: IN STD_LOGIC;
o: OUT STD_LOGIC);
END COMPONENT;
COMPONENT or2gate IS PORT(
i1, i2: IN STD_LOGIC;
o: OUT STD_LOGIC);
END COMPONENT;
COMPONENT notgate IS PORT(
i: IN STD_LOGIC;
o: OUT STD_LOGIC);
END COMPONENT;
COMPONENT xor2gate IS PORT(
i1, i2: IN STD_LOGIC;
o: OUT STD_LOGIC);
END COMPONENT;
COMPONENT multiplexer4 IS PORT(
d0,d1,d2,d3,s0,s1: IN STD_LOGIC;
y: OUT STD_LOGIC);
END COMPONENT;
COMPONENT fulladder IS PORT(
A,B,cin: IN STD_LOGIC;
sum,cout: OUT STD_LOGIC);
END COMPONENT;
Signal          e1,e2,e3,e4,e5,e6,e7,e8:
STD_LOGIC;
begin
u11:notgate portmap(B111,e1);
u12:and2gate portmap(A111,B111,e3);
u13:or2gate portmap(A111,B111,e4);
u14:xor2gate portmap(A111,B111,e5);
u15:notgate portmap(A111,e6);
u16:multiplexer4
portmap('0',B111,e1,'1',s01,s11,e2);
u17:multiplexer4
portmap(e3,e4,e5,e6,s01,s11,e8);
u18:fulladder      portmap(A111,e2,cin1
,e7,co);
u19:multiplexer4
portmap(e7,e8,sl,sr,s21,s31,yy);
END Structural;
```

4.1.2- Design Simulation

Xilinx (Project Navigator) CAD software package simulator allows you to simulate the design before being manufactured, so it is easy to compare alternatives and test for correctness without the delay and expensive of hardware prototype.

This simulation is an ideal time to catch design faults such as incorrect module annotation problems in the data flow and incomplete design descriptions.

The designed circuit is simulated by simulating the inputs of the ALSU ($A1$, $A2$, $A3$, $A4$, $A5$, $A6$, $A7$, $A8$, $B1$, $B2$, $B3$, $B4$, $B5$, $B6$, $B7$, $B8$, Cin, shl, shr, S_0, S_1, S_2 and S_3)with a known values and examine the outputs of the ALSU (f1, f2, f3, f4, f5, f6, f7, f8 and Co).

Figure (3) illustrate the simulation of the 8-bit ALSU with the following values in the input:

($A8$-$A1$)= $(10111111)_2$

($B8$-$B1$)= $(10111010)_2$

Shr (shift left input) = 1

Shl (shift right input) = 1

When S_3 S_2 S_1 S_0=0000 the input from B are neglected and instead, all 0's are inserted to the input of the full adder. The output becomes f= A +0+Cin. This gives f= A when Cin=0 and f= A +1,when Cin=1. In the first case we have a direct transfer from input A to output f. In the second case, the value of A is incremented by 1,for the given example;

f= A = (10111111) & Co=0

f= A +1= (11000000) & Co=0

When S_3 S_2 S_1 S_0=0001 the value of B is applied to the input of the full adder, if Cin=0, then the output f= $A+B$ .If Cin=1, output f= $A+B+1$. In the first case we have an addition operation, when in the second case it is an addition with carry.

f= $A+B$ = (01111001) & Co=1

f= $A+B+1$= (01111010) & Co=1

When S_3 S_2 S_1 S_0=0010 the complement of B is applied to the input of the full adder. If Cin=0, then f= $A+\overline{B}$. This is equivalent to a subtract with borrow, that is, $A-B-1$. If Cin=1, then f= $A+\overline{B}+1$. This produces A plus the 2's complement of B, which equivalent to a subtraction of $A-B$.

f= $A+\overline{B}$ = (00000100) & Co=1

f= $A+\overline{B}+1$= (00000101) &Co=1

When S_3 S_2 S_1 S_0=0011, all 1's are inserted into the inputs of the full adder to produce the decrement operation f= $A-1$ when Cin=0. This is because a number with all 1's is equal to the 2's complement of 1. Adding a number A to the 2's complement of 1 produces f= A+2's complement of 1 = A -1. When Cin=1, then f= $A-1+1=A$, which causes a direct transfer from input A to output f.

f= $A-1$= (10111110) &Co=1

f= $A-1+1$= (10111111) & Co=1

In the above operation the select line S_3 S_2 = 00 select the arithmetic circuit output to be input to the multiplexer and the select lines S_1 S_0 will select the eight arithmetic operation in this circuit.

When S_3 S_2= 01, this select the logic circuit output to be input to the multiplexer and the select lines S_1 S_0 will select the four logic operation in this circuit.

f= $A \wedge B$ = (10111010)

f= $A \vee B$ = (10111111)

f= $A \oplus B$ = (00000101)

f= $\overline{A}$ = (01000000)

When S_3 S_2= 10, this will cause a 1-bit shift to the right of number A .The bit transferred to the end position depend on the value of the (shr) input line.

A= 10111111 & shr =1 after executing the shift operation the output of the ALSU will equal to;

f= shr A = (11011111)

When S_3 S_2= 11, this will cause a 1-bit shift to the left of number A .The bit transferred to the end position depend on the value of the (shl) input line.

A= 10111111 & shl =1 after executing the shift operation the output of the ALSU will equal to;

f= shl A = (01111111)

As another example, figure (4) illustrates the simulation of the 8-bit ALSU with the following values in the input:

(A8- A1)= (1 1 0 0 0 0 1 1)$_2$

(B8- B1)= (1 1 0 0 1 1 0 0)$_2$

Shr (shift left input) = 0

Shl (shift right input) = 0

When $S_3\ S_2\ S_1\ S_0$=0000 & Cin=0 then
f= A = (11000011) & Co=0
When $S_3\ S_2\ S_1\ S_0$=0000 & Cin=1 then
f= A +1= (11000100) & Co=0
When $S_3\ S_2\ S_1\ S_0$=0001 & Cin=0 then
f= $A + B$ = (10001111) & Co=1
When $S_3\ S_2\ S_1\ S_0$=0001 & Cin=1 then
f= $A + B$ +1=(10010000)& Co=1
When $S_3\ S_2\ S_1\ S_0$=0010 & Cin=0 then
f= $A + \overline{B}$ = (11110110) & Co=0
When $S_3\ S_2\ S_1\ S_0$=0010 & Cin=1 then
f= $A + \overline{B}$ +1= (11110111) &Co=0
When $S_3\ S_2\ S_1\ S_0$=0011 & Cin=0 then
f= A -1= (11000010) &Co=1
When $S_3\ S_2\ S_1\ S_0$=0011 & Cin=1 then
f= A -1+1= (11000011) & Co=1
When $S_3\ S_2\ S_1\ S_0$=0100 then
f= $A \wedge B$ = (11000000)
When $S_3\ S_2\ S_1\ S_0$=0101 then
f= $A \vee B$ = (11001111)
When $S_3\ S_2\ S_1\ S_0$=0110 then
f= $A \oplus B$ = (00001111)
When $S_3\ S_2\ S_1\ S_0$=0111 then
f= $\overline{A}$ = (00111100)
When $S_3\ S_2\ S_1\ S_0$=10 ×× & shr=0 then
f= shr A = (01100001)
When $S_3\ S_2\ S_1\ S_0$=11 ×× & shl=0 then
f= shl A = (10000110)

For another example, figure (5) illustrates the simulation of the 8-bit ALSU with the following values in the input:

(A8-A1)= $(0\ 0\ 0\ 0\ 1\ 1\ 1\ 1)_2$

(B8-B1)= $(1\ 1\ 1\ 1\ 0\ 0\ 0\ 0)_2$

Shr (shift left input) = 1

Shl (shift right input) = 0

The above three examples examined for all operations that the ALSU can perform according to the values of the four selected lines. The selected lined formatted so that to select all the operations shown in table (1).

4.1.3- Design Synthesis

In this step, the estion now is how the program that describes the operation of the ALSU actually gets converted to the physical circuit. The synthesizer is used to convert the VHDL description into a set of primitives or components that can be assembled in FPGA technology.

The design has been synthesized; a physical netlist of the design. Thus, design was targeted to Xilinx (virtex-I I) FPGA type (XC2V40). The resulting netlist is then used to produce a bitstream which can be downloaded to the FPGA's configuration RAM.

4.1.4-Design Implementation

In this step the netlist from the output of the synthesizer can be used directly to implement the actual circuit in a field programmable gate array (FPGA). With this final step the creation of a digital circuit that is implemented fully in integrated circuits can be done directly.

This step has several operations like completes the hardware design, translates the gate level design into hardware primitives available in (XC2V40) FPGA, assigns the design to physical locations on the chip and routes the connections between them, timing information about the design, and determines the

configuration bits to implement the design. Figure (6) illustrates the XC2V40 implementation process.

After the completion of implementation step the Floor Planner can be used to get more information about the design connectivity and resource requirements, and the design mapping via location constraints, as shown in figure (7).

At the end, Programming step is began to generate a programming file, which specifies the operation of the FPGA device. The data of this file is stored in an E^2PROM with 64K Bytes. When the power is applied to the FPGA chip, the storage cells are loaded automatically from the memory.

5- Conclusions

In this paper, an ALSU has been designed and implemented using FPGA technology. The design implemented on Xilinx (virtex-I I) FPGA type (XC2V40), Using Xilinx Integrated Software Environment ISE 6.i (project navigator) CAD software package. This design simulated on many data examples for each operation, three examples has been tested successfully for the 14 different ALSU operations and this clear from the simulation result shown in figures 3, figure 4, and figure 5. Using FPGA technology in this design enabled us to replace the use of a conventional design using logic gets (Multiplexer, AND, OR, NOT, XOR) and also the use of available ALU (for example 74181) since when we use FPGA technology it is easy to insert more function and operation that can be implemented in the designed ALSU by converting the program to implements the required functions, since FPGA's provide the possibility of implementing logic circuits by programming the required function, offer the benefits of low costs, short manufacturing turnaround time and easy design changes.

As a result, most prototypes and many production designs are now implemented on FPGA's, making hardware implementation economically feasible even for those applications which were previously restricted to software implementation.

6- References

[1] M. Morris Mano, Computer System Architecture, Prentice Hall PTR, third edition, 1993.

[2] w. stallings, Computer Organization And Architecture, Prentice Hall PTR, sixth edition, 2003.
At:www.williamstallings.com

[3] J. Bhasker, A VHDL Primer, Prentice Hall PTR, fourth edition, 2002.

[4] M. Zwolinski, Digital System Design With VHDL, Prentice Hall Inc., 2000.

[5]Peter J. Ashenden, Designers Guide to VHDL, Mrgan Kaufmann, Second Edition ,2001.
At:www.booksmatter.com

[6] J. F. Wakerly, Field-Programmable Gate Array, 1999.

[7] Virtual Computer Corporation, *FPGA: An Enabling Technology*, Product Overview.
At:www.vcc.com,2001.

[8] Xilinx Co., The FPGA Story.
At:www.xilinx.com, 2001

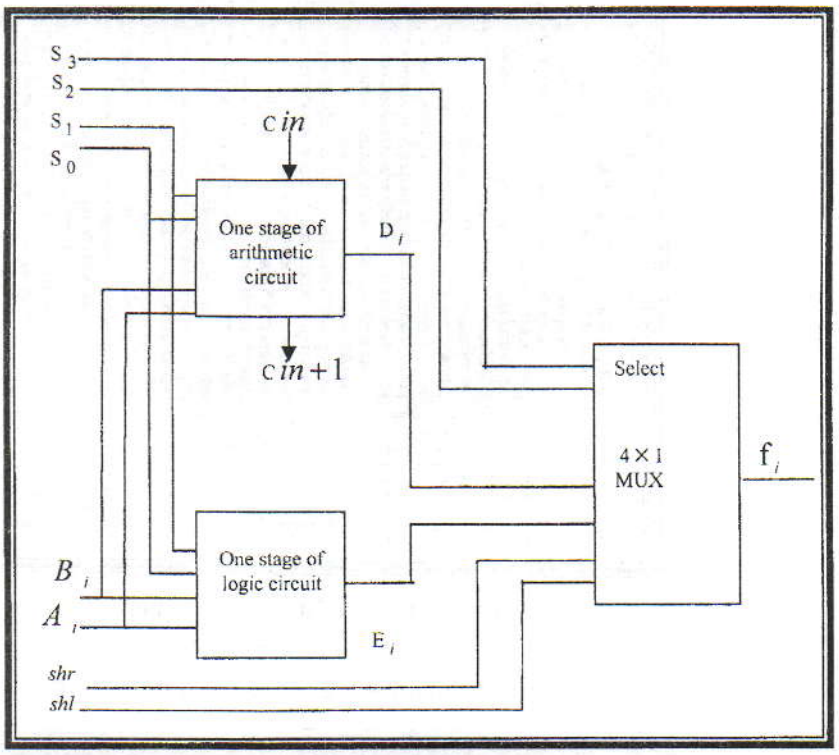

Fig. (1) One stage of Arithmetic Logic Shift Unit (ALSU).

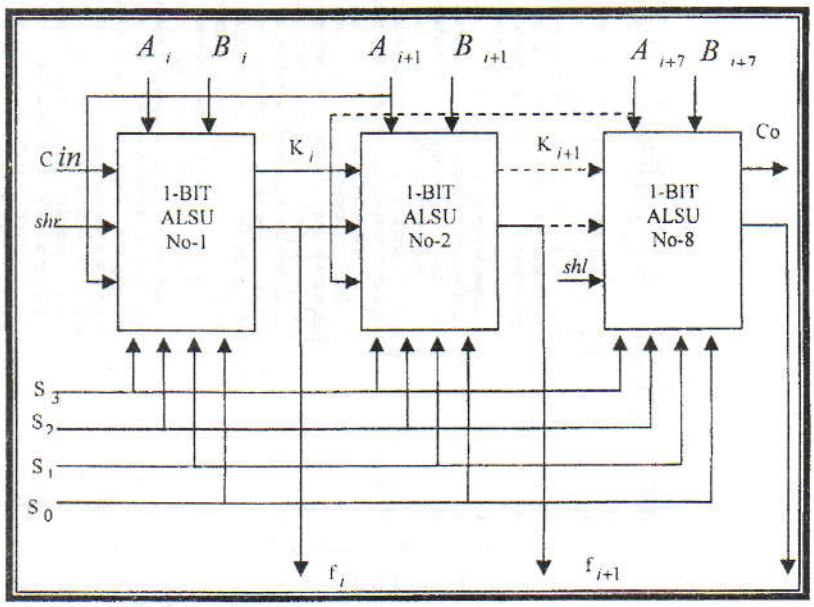

Fig. (2) Eight Bit Arithmetic Logic Shift Unit (ALSU).

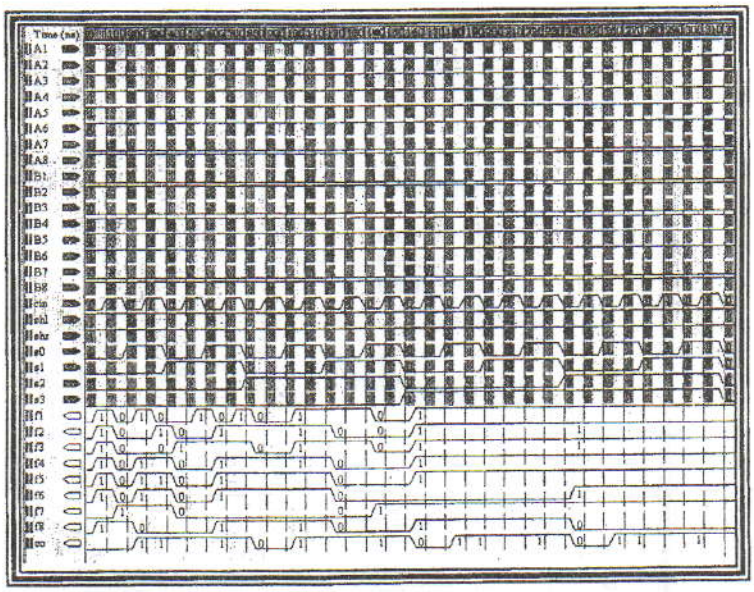

Fig. (3) Simulation of the ALSU when
$(A8\text{-}A1)=(10111111)_2$,$(B8\text{-}B1)=(10111010)_2$,
Shr (shift left input) = 1, and Shl (shift right input) = 1.

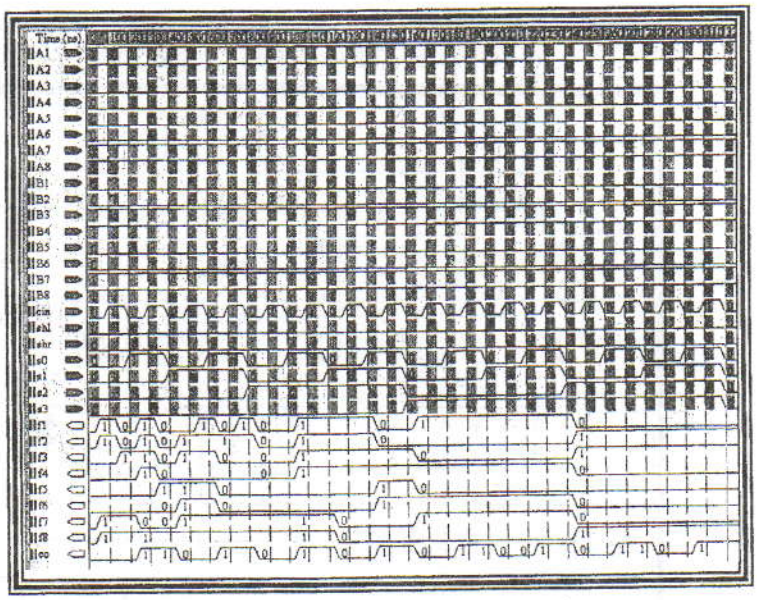

Fig. (4) Simulation of the ALSU when
$(A8\text{-}A1)=(11000011)_2$,$(B8\text{-}B1)=(11001100)_2$,
Shr (shift left input) = 0, and Shl (shift right input) = 0.

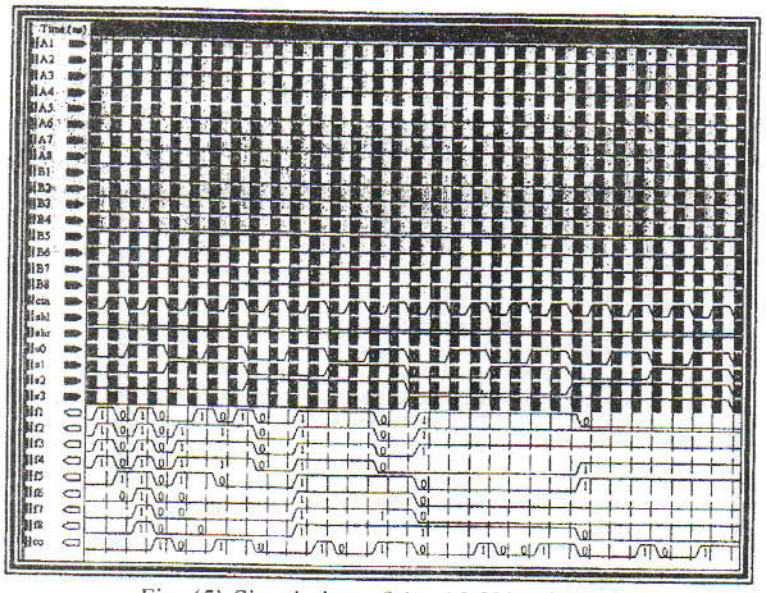

Fig. (5) Simulation of the ALSU when $(A8\text{-}A1)=(11110000)_2$, $(B8\text{-}B1)=(00001111)_2$, Shr (shift left input) = 1, and Shl (shift right input) = 0.

Operation select					Operation	Function
S_3	S_2	S_1	S_0	C_{in}		
0	0	0	0	0	F= A	Transfer A
0	0	0	0	1	F= A + 1	Increment A
0	0	0	1	0	F= $A + B$	Addition
0	0	0	1	1	F= $A + B + 1$	Add with carry
0	0	1	0	0	F= $A + \overline{B}$	Subtract with borrow
0	0	1	0	1	F= $A + \overline{B} + 1$	Subtraction
0	0	1	1	0	F= $A - 1$	Decrement A
0	0	1	1	1	F= A	Transfer A
0	1	0	0	×	F= $A \wedge B$	AND
0	1	0	1	×	F= $A \vee B$	OR
0	1	1	0	×	F= $A \oplus B$	XOR
0	1	1	1	×	F= $\overline{A}$	Complement A
1	0	×	×	×	F= shr A	Shift right A into F
1	1	×	×	×	F= shl A	Shift left A into F

Table (1) Function table for Arithmetic Logic Shift Unit (ALSU).

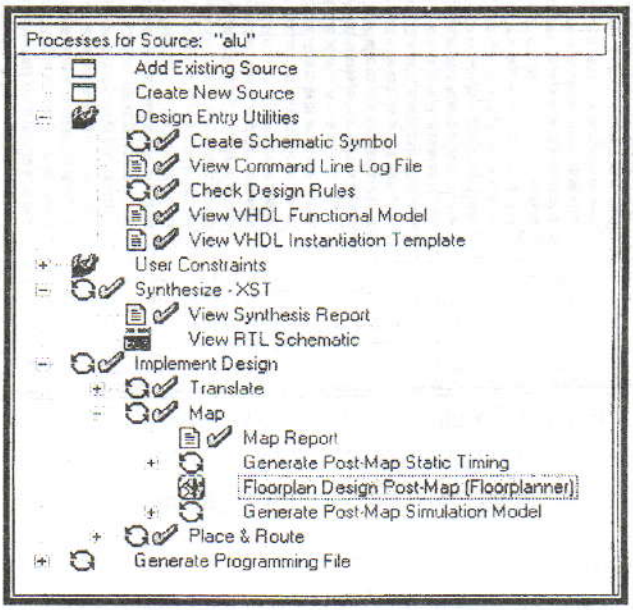

Fig. (6)The Process screen illustrates the XC2V40 implementation process.

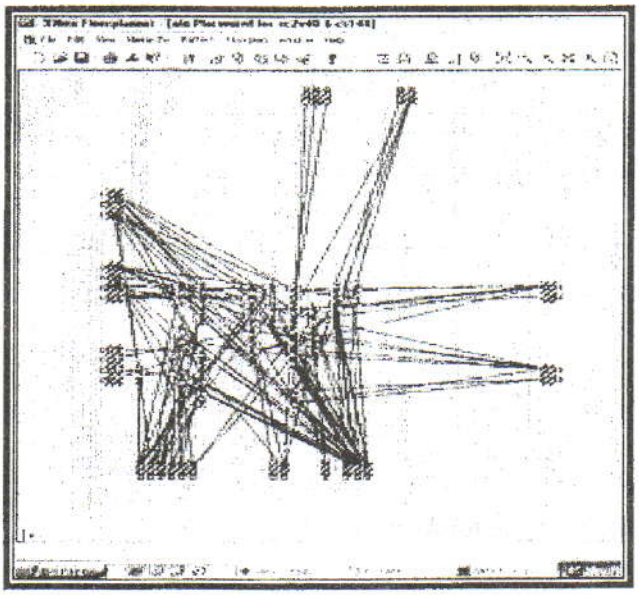

Fig. (7) The Floor Planner.