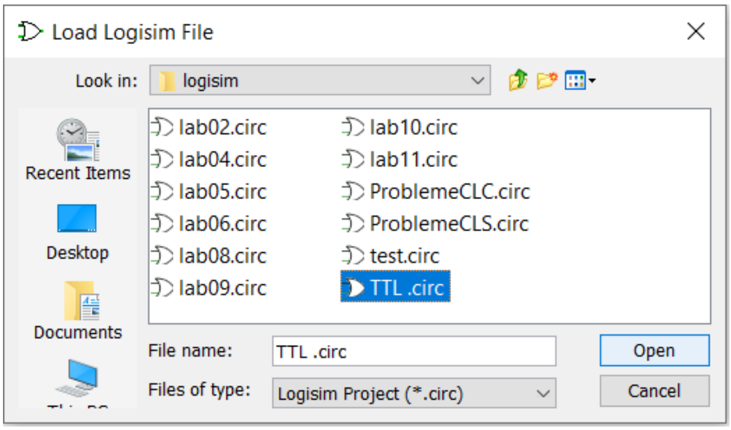
4 Design and simulation of digital circuits using computer aided design software (II)

4.1 Objectives

The management of the libraries is described as means to hierarchy design in Logisim. An example of hierarchy design is detailed. The facilities offered by Logisim for the combinational analysis of a circuit are enumerated.

4.2 Library management in Logisim

An implicit library is attached by default to a Logisim project. The library may contain several circuits represented by their logic diagram. The file created by Logisim – when saving the project – contains the project library with its logic diagrams, as well as the project dependencies. This enables the possibility to load the project library into another project; its circuits will be available for reuse. The integration of circuits from current library or from other libraries is called *hierarchy design*. A library can be loaded with command Load Library > Logisim-library… from the Project menu. Logisim will request the associated project file, as in Figure 4. 1.



Look in: logisim

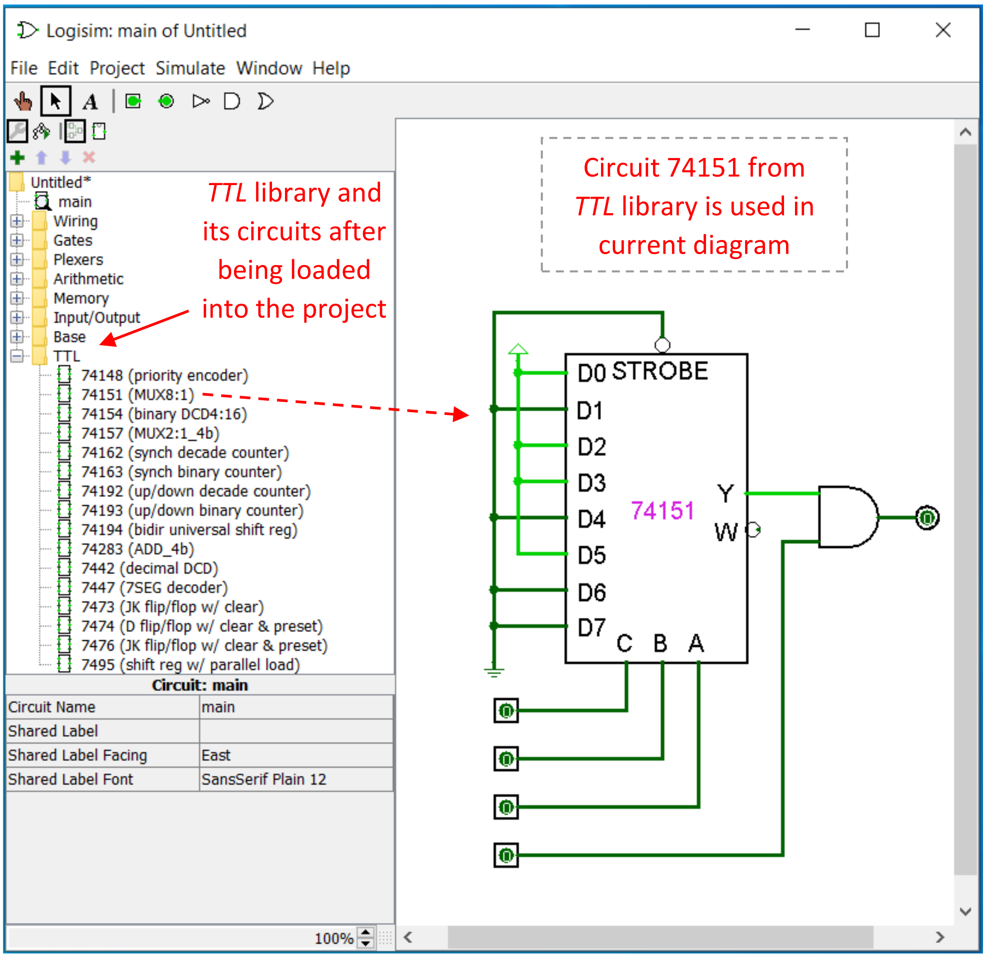
File name: TTL .circ

Files of type: Logisim Project (*.circ)

Figure 4. 1 When loading a library, Logisim asks the location of the project file implementing the library

The libraries loaded into the project appear in the Toolbox area, and their circuits – represented by symbols – are available for reuse. Figure 4. 2 highlights the access to circuits from TTL library. A project can load as many libraries as necessary, which allows the possibility to integrate components from multiple sources.

A library can be removed from the project by right-clicking its name and selecting Unload Library from the context menu. Selecting Reload Library will update the library.



Logisim: main of Untitled
File Edit Project Simulate Window Help
D0 STROBE
D1
D2
D3
D4
D5
D6
D7
C B A
74151
Y
W
100%

TTL library and its circuits after being loaded into the project

Circuit 74151 from *TTL* library is used in current diagram

- Untitled*
  - main
  - Wiring
  - Gates
  - Plexers
  - Arithmetic
  - Memory
  - Input/Output
  - Base
  - TTL
    - 74148 (priority encoder)
    - 74151 (MUX8:1)
    - 74154 (binary DCD4:16)
    - 74157 (MUX2:1_4b)
    - 74162 (synch decade counter)
    - 74163 (synch binary counter)
    - 74192 (up/down decade counter)
    - 74193 (up/down binary counter)
    - 74194 (bidir universal shift reg)
    - 74283 (ADD_4b)
    - 7442 (decimal DCD)
    - 7447 (7SEG decoder)
    - 7473 (JK flip/flop w/ clear)
    - 7474 (D flip/flop w/ clear & preset)
    - 7476 (JK flip/flop w/ clear & preset)
    - 7495 (shift reg w/ parallel load)

Circuit: main

Circuit Name	main
Shared Label	
Shared Label Facing	East
Shared Label Font	SansSerif Plain 12

Figure 4. 2 *TTL* library appears in the Toolbox after being loaded. Its circuits can be accessed via their symbols

4.3 Circuit analysis

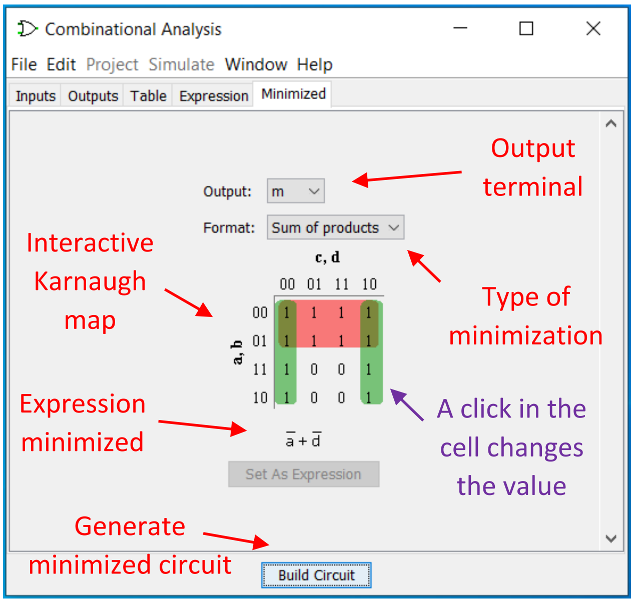
A logic diagram supports the following types of analysis:

- Circuit analysis;
- Circuit statistics.

Circuit analysis is available with Analyze Circuit command from Project menu. The Combinational Analysis window that appears when launching this utility is organized on several Tabs as follows (Figure 4. 3):

- Inputs – contains the list of input terminals detected on the diagram. They are identified by their label or receive one automatically, if the label is missing.
- Outputs – contains the list of detected output terminals.
- Table – highlights the truth table of the circuit outputs. A click in the cells of the output columns will toggle the values between 0, 1, and x.
- Expression – lists the boolean expressions of the outputs in Minimal Disjunctive Normal Form or in Minimal Conjunctive Normal Form.
- Minimized – lists the functions of the outputs in a Minimal Normal Form – sum of products (disjunctive form) or product of sums (conjunctive form) – as set

by the Format choice. If the diagram contains up to 4 inputs, this section will highlight the corresponding Karnaugh map. The map has an interactive behavior: any click inside the grid will toggle the value in the corresponding cell between 0, 1 and x. Consequently, the minimization adjusts automatically, with effects on the grouped cells, as well as on the resulting minimal expression. If enabled, the Set As Expression button should be pressed, to set the minimized form as primary expression of the Combinational Analysis tool.


Combinational Analysis
File Edit Project Simulate Window Help
Inputs Outputs Table Expression Minimized

Output: m

Format: Sum of products

a, b \ c, d	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	1	0	0	1
10	1	0	0	1

$\bar{a} + \bar{d}$

Set As Expression
Build Circuit

Output terminal

Type of minimization

Interactive Karnaugh map

Expression minimized

A click in the cell changes the value

Generate minimized circuit

Figure 4. 3 Combinational analysis of the circuit from Figure 4. 4

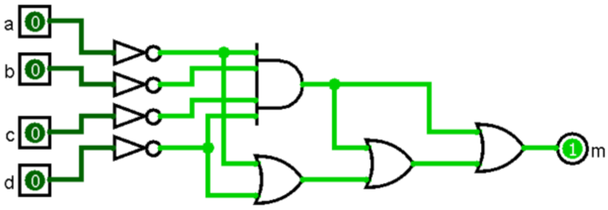


Figure 4. 4 A logic diagram with complex structure that can be simplified using the Combinational Analysis tool

Note: Circuit analysis works only for combinational circuits: which have their outputs as a function of the inputs.

The Build Circuit button is available on all tabs of the Combinational Analysis window. Pressing the button will generate a new circuit that implements the current expressions in the Combinational Analysis tool. The tool will require the name of the new circuit. The circuit will be attached to the project library. For example, the minimized version of the circuit in Figure 4. 4 is highlighted in Figure 4. 5.

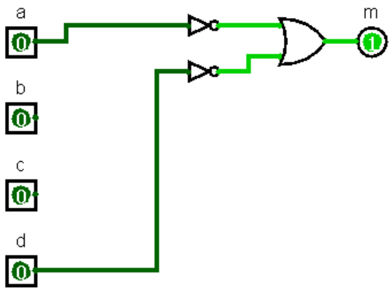
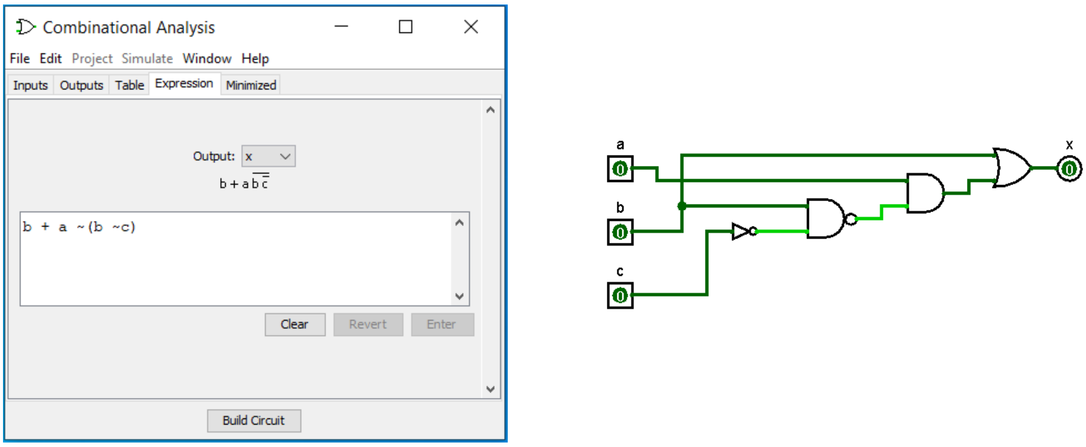


Figure 4. 5 Logic diagram after minimizing the circuit from Figure 4. 4

The Combinational Analysis tool may also be used to generate a new circuit from its analytical expression, truth table or Karnaugh map. When the diagram is empty the tool can be launched with the command Combinational Analysis from the Window menu. The input terminals may be added in the Inputs tab, and the output terminals in the Outputs tab. The logic functions corresponding to outputs can be defined analytically in the Expression tab, by truth tables in the Table tab or through the Karnaugh maps in the Minimized tab. **Note**: Karnaugh maps are available if the number of inputs is less than 5. After the output functions are defined, the circuit can be generated by pressing the Build Circuit button.

Note: The analytical expression defined in the Expression tab accepts the following operators: NOT, AND, OR and XOR. Operator NOT is represented by character ~, AND is represented by a space, OR can be defined with + and XOR with ^. Brackets may also be used if necessary. After finalizing the expression, the Enter button should be pressed, if available (Figure 4. 6).


Combinational Analysis
File Edit Project Simulate Window Help
Inputs Outputs Table Expression Minimized
Clear
Revert
Enter
Build Circuit
a
b
c
x
0
0
0
0

Output: x

$b + a\,\overline{b\,\overline{c}}$

```
b + a ~(b ~c)
```

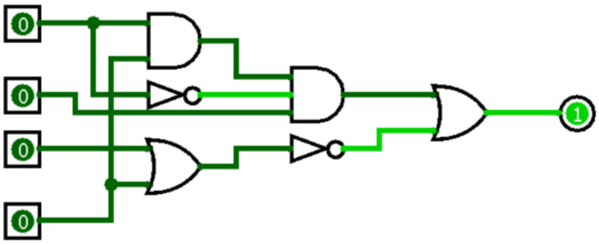
Figure 4. 6 Boolean expression of the output function defined in the Combinational Analysis tool (left). The corresponding circuit generated for this expression (right)

Circuit statistics can be visualized using the command Get Circuit Statistics from the Project menu. A new window will appear containing statistical data related to elements found in the current logic diagram.

4.4 Assignments

1. Create a new project in Logisim and load the *TTL* library. Analyze the list of circuits inside the library.

2. Create a new diagram for the circuit in the next figure and use the Combinational Analysis tool to generate the equivalent circuit obtained after minimization using the Minimized tab.



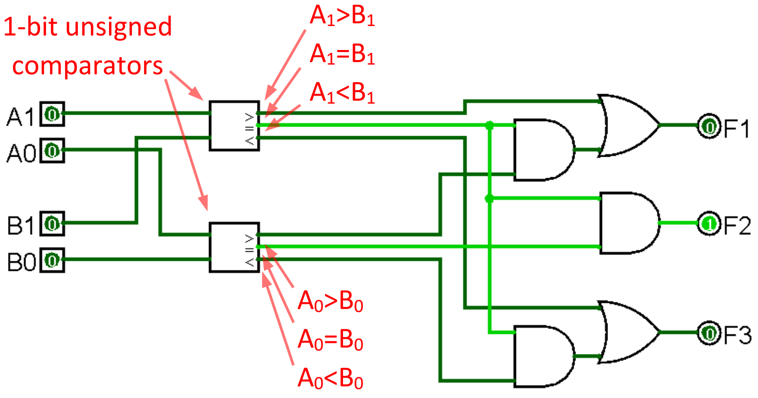
3. Use tabs Inputs, Outputs and Expression from the Combinational Analysis tool to implement the following logic functions in a single logic diagram.

f1 = ~a + b + ~c   f2 = ~(a + b) (a + c)   f3 = ~a + ~(~b c)

4. Implement the logic diagram in the next figure, which represents the design of an unsigned 2-bit comparator unit, using 1-bit unsigned comparators (with attribute `Numeric Type=Unsigned`) from the *Arithmetic* library, and additional logic gates. Test the 2-bit comparator in the simulator for all input combinations.

The diagram explained: The 2-bit numbers which are compared are A_1A_0 and B_1B_0. The diagram implements the next rules for the 3 cases possible:

- $F_1 = A_1A_0>B_1B_0$: $F_1=1$ if $A_1>B_1$ **OR** ($A_1=B_1$ **AND** $A_0>B_0$);
- $F_2 = A_1A_0=B_1B_0$: $F_2=1$ if $A_1=B_1$ **AND** $A_0=B_0$;
- $F_3 = A_1A_0<B_1B_0$: $F_3=1$ if $A_1<B_1$ **OR** ($A_1=B_1$ **AND** $A_0<B_0$).


1-bit unsigned comparators
A1>B1
A1=B1
A1<B1
A1
A0
B1
B0
F1
F2
F3
A0>B0
A0=B0
A0<B0