DOI: 10.22184/1992-4178.2017.166.6.100.106

РЕАЛИЗАЦИЯ Verilog-ПРОЕКТОВ В БАЗИСЕ ПЛИС INTEL FPGA С ИСПОЛЬЗОВАНИЕМ ИНСТРУМЕНТА СИНТЕЗА Yosys

А.Строгонов, д.т.н[1]., П.Городков[2]

УДК 621.3.049
ВАК 05.27.00

Инструмент синтеза с открытым исходным кодом Yosys (Yosys Open SYnthesis Suite) широко используется при проектировании заказных БИС и ПЛИС. В пятом номере журнала "ЭЛЕКТРОНИКА: Наука, Технология, Бизнес" за 2017 год мы рассказали об особенностях применения Yosys при реализации Verilog-проектов в базисе заказных БИС и ПЛИС Xilinx. Продолжим изучение возможностей этого популярного в академических университетских центрах программного инструмента. Рассмотрим использование Yosys для синтеза Verilog-проекта в ПЛИС Intel FPGA (Altera) серии Cyclone IV и покажем, как посредством формирования VQM-файла без использования мегафункций САПР Quartus Prime можно реализовывать цифровые устройства в базисе современных ПЛИС.

VQM-файл обеспечивает связь между инструментом синтеза Yosys и САПР Quartus Prime. В рамках статьи под термином "синтез цифровых устройств" понимается автоматическое преобразование проекта с уровня регистровых передач (RTL) на уровень вентилей технологического базиса ПЛИС.

VQM-файл – это представление RTL-проекта на "атомном" уровне с использованием сетевых примитивов технологического списка межсоединений (netlist) и мегафункций конкретной серии ПЛИС.

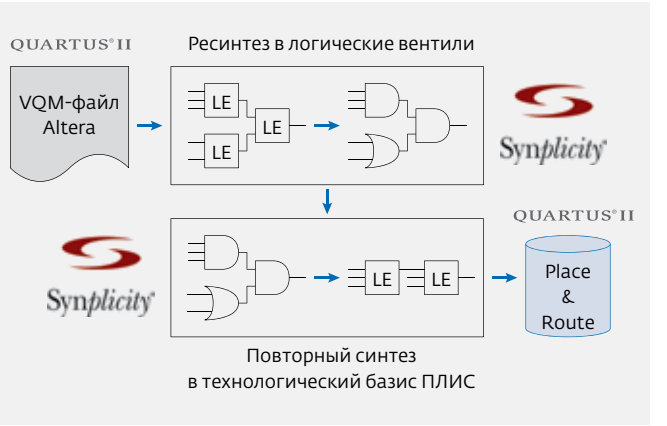

Рис.1. Ресинтез технологического отображения в логические вентили из VQM-файла проекта и последующий синтез логики в базисе ПЛИС с получением VQM-файла

1 Воронежский государственный технический университет, профессор кафедры полупроводниковой электроники и наноэлектроники, тел. (4732) 437695, andreistrogonov@mail.ru.

2 Воронежский государственный технический университет, аспирант кафедры полупроводниковой электроники и наноэлектроники, тел. (4732) 437695, gorodkoff@gmail.com.

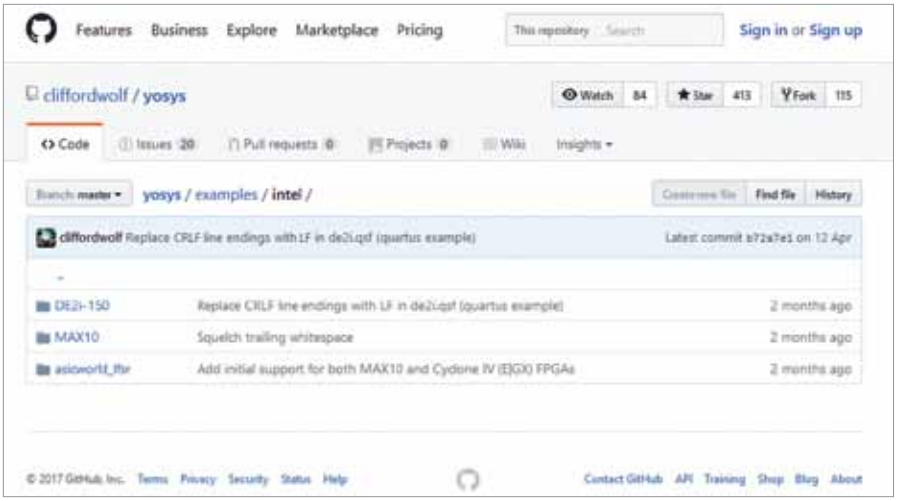

Рис.2. Страница сайта github.com с примерами для реализации проектов в базисе ПЛИС Intel FPGA серий MAX10 и Cyclone IV (папка asicworld_fsr)

Например, наиболее важные примитивы ПЛИС серии Stratix IV: lcell_comb (таблицы перекодировок, LUT и сумматоры), dffeas (триггер, тактируемый фронтом синхросигнала), mac_mult (умножитель MAC-блока), mac_out (аккумулятор MAC-блока), ram_block (секция ОЗУ). В САПР Quartus II VQM-файл можно извлечь как на этапе анализа и синтеза, так и после полной компиляции с учетом размещения и трассировки.

Широко используемый инструмент синтеза Synplify Pro от Synplicity позволяет с помощью VQM-файлов технологически отображать VHDL- и Verilog-проекты (с применением мегафункций в виде созданных в Quartus II "черных ящиков") в базис ПЛИС компании Altera. Synplify Pro может также выполнять ресинтез технологического отображения с использованием специального стиля языка Verilog – Technology Independent Coding Styles – в вентили и последующий синтез логики в базисе ПЛИС с получением VQM-файла на основе своих алгоритмов. Это позволяет в некоторых случаях уменьшить количество используемых логических ресурсов (рис.1). Подключение инструмента синтеза Synplify Pro в САПР Quartus II выполняется в меню Assigments/Settings/EDA Tool Settings/Design Entry Synthesis.

Рассмотрим реализацию Verilog-проекта в САПР Quartus Prime версии 16.1 Lite с использованием ПЛИС серии Cyclone IV GX (EP4CGX150DF31C7). На сайте github.com по адресу https://github.com/cliffordwolf/yosys/tree/master/examples/intel представлены примеры проектов для ПЛИС Intel FPGA серий MAX10 и Cyclone IV (рис.2). Исходный код проекта взят с сайта www.asic-world.com (http://www.asic-world.com/examples/verilog/lfsr.html).

```
'default_nettype none
module lfsr_updown (
clk       ,   // Clock input
reset     ,   // Reset input
enable    ,   // Enable input
up_down   ,   // Up Down input
count     ,   // Count output
overflow      // Overflow output
);
 input clk;
 input reset;
 input enable;
 input up_down;
 output [7 : 0] count;
 output overflow;
 reg [7 : 0] count;
 assign overflow = (up_down) ? (count == {{7{1'b0}}, 1'b1}) :
                   (count == {1'b1, {7{1'b0}}}) ;
 always @(posedge clk)
 if (reset)
    count <= {7{1'b0}};
 else if (enable) begin
    if (up_down) begin
      count <= {~(^(count & 8'b01100011)),count[7:1]};
    end else begin
      count <= {count[5:0],~(^(count &  8'b10110001))};
    end
 end
endmodule
```

Рис.3. Verilog-код LFSR-регистра

```
module tb();
 reg clk;
 reg reset;
 reg enable;
 reg up_down;
 wire [7 : 0] count;
 wire overflow;
initial begin
  $monitor("rst %b en %b updown %b cnt %b overflow %b",
     reset,enable,up_down,count, overflow);
  clk = 0;
  reset = 1;
  enable = 0;
  up_down = 0;
  #10 reset = 0;
  #1 enable = 1;
  #20 up_down = 1;
  #30 $finish;
end
always #1 clk = ~clk;
lfsr_updown U(
.clk      ( clk      ),
.reset    ( reset    ),
.enable   ( enable   ),
.up_down  ( up_down  ),
.count    ( count    ),
.overflow ( overflow )
);
endmodule
```

Рис.4. Испытательный стенд (test bench) для LFSR-регистра

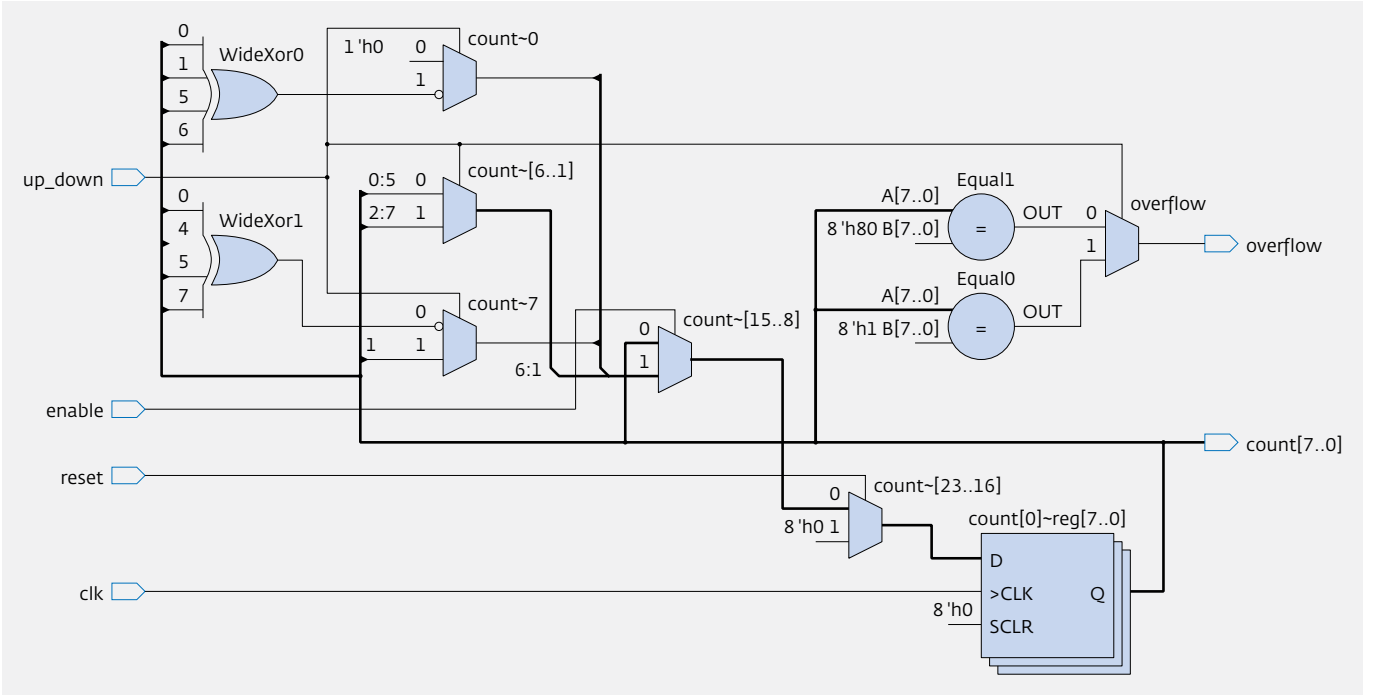

Рис.5. RTL-представление Verilog-кода LFSR-регистра на этапе анализа и синтеза

В качестве примера рассмотрим реализацию регистра сдвига с линейной обратной связью (Llinear Feedback Shift Register – LFSR), который представляет собой регистр сдвига битовых слов, у которого значение входного (вдвигаемого) бита равно линейной булевой функции от значений остальных битов регистра до сдвига. LFSR-регистр применяется для генерации псевдослучайных последовательностей битов. На рис.3 представлен Verilog-код LFSR-регистра, а на рис.4 – испытательный стенд (test bench) для него. На рис.5 показано RTL-представление LFSR-регистра на этапе анализа и синтеза, а на рис.6 – после полной компиляции на этапе размещения и трассировки в базисе ПЛИС EP4CGX150DF31C7.

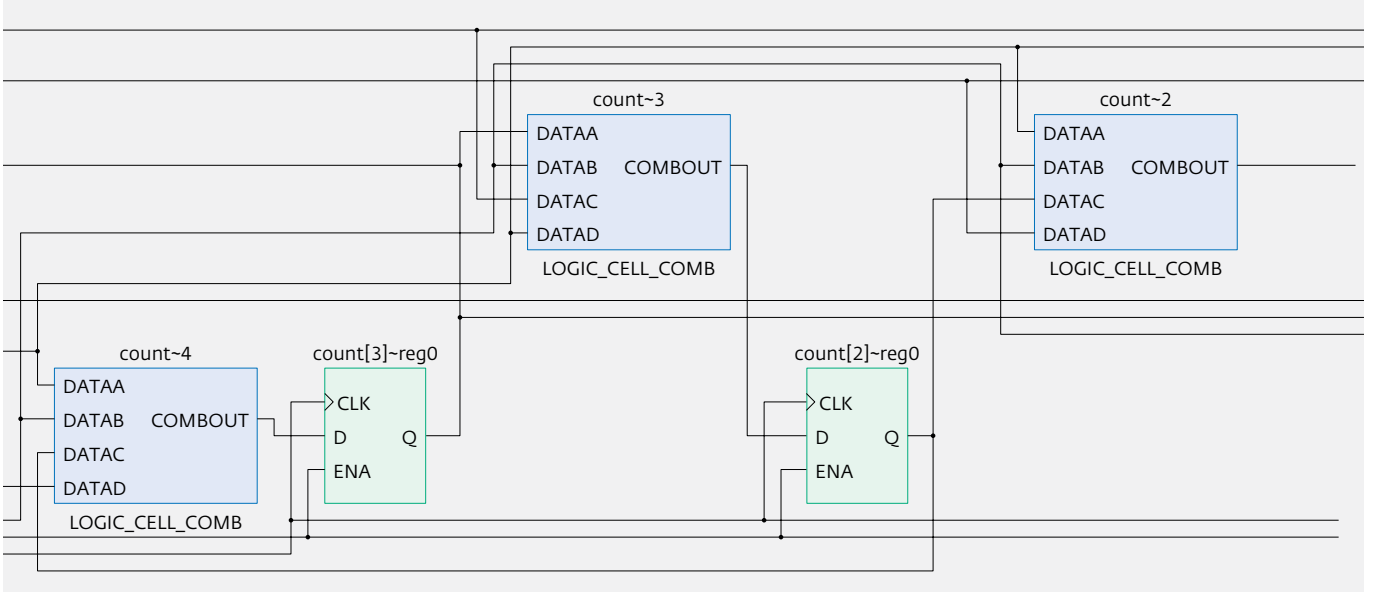

Рис.6. RTL-представление LFSR-регистра после размещения и трассировки Verilog-кода в базисе ПЛИС EP4CGX150DF31C7 (фрагмент)

```
#!/bin/bash
iverilog -o presynth lfsr_updown_tb.v lfsr_updown.v &&\
vvp -N presynth
```

Рис.7. Скрипт runme_presynth для моделирования работы LFSR-регистра

```
tricut@tricut-R59P-R60P-R61P ~/yosys/examples/intel/
asicworld_lfsr $ bash runme_presynth
rst 1 en 0 updown 0 cnt xxxxxxxx overflow x
rst 1 en 0 updown 0 cnt 00000000 overflow 0
rst 0 en 0 updown 0 cnt 00000000 overflow 0
.....
rst 0 en 1 updown 1 cnt 11100110 overflow 0
rst 0 en 1 updown 1 cnt 01110011 overflow 0
rst 0 en 1 updown 1 cnt 10111001 overflow 0
```

Рис.8. Фрагмент работы испытательного стенда (test bench) LFSR-регистра

```
#!/bin/env bash
yosys -p "synth_intel -family cycloneiv -top lfsr_updown
-vout top.vqm" lfsr_updown.v
```

Рис.9. Скрипт run_cycloneiv

```
1. Executing Verilog-2005 frontend.
Parsing Verilog input from 'lfsr_updown.v' to AST
representation.
Generating RTLIL representation for module '\lfsr_updown'.
Successfully finished Verilog frontend.
-- Running command 'synth_intel -family cycloneiv -top
lfsr_updown -vout lfsr_updown.vqm' --
2. Executing SYNTH_INTEL pass.
2.1. Executing Verilog-2005 frontend.
Parsing Verilog input from '/usr/local/bin/../share/yosys/
altera_intel/cycloneiv/cells_comb_cycloneiv.v' to AST
representation.
Generating RTLIL representation for module '\VCC'.
Generating RTLIL representation for module '\GND'.
Generating RTLIL representation for module '\
cycloneiv_io_ibuf'.
Generating RTLIL representation for module '\
cycloneiv_io_obuf'.
Generating RTLIL representation for module '\
cycloneiv_lcell_comb'.
Generating RTLIL representation for module '\dffeas'.
Successfully finished Verilog frontend.
```

Рис.10. Фрагмент отработки скрипта run_cycloneiv, выводимый в консоль

Для того чтобы реализовывать Verilog-проекты в ПЛИС серий Cyclone IV GX и MAX10, требуется новая версия Yosys 0.7+194, поэтому вначале следует обновить Yosys до версии 0.7+194 с сайта github.com/cliffordwolf/yosys (см. рис.2). В папке ~/yosys/examples/intel/asicworld_lfsr нужно запустить скрипт runme_presynth, который состоит из двух команд (рис.7).

```
/* Generated by Yosys 0.7+194 (git sha1 0290b68, clang
3.4-1ubuntu3 -fPIC -Os) */
/* top =  1  */
/* src = "lfsr_updown.v:2" */
module lfsr_updown(clk, reset, enable, up_down, count,
overflow);
  /* src = "lfsr_updown.v:24" */
  wire [7:0] yosys__00_;
  wire yosys__01_;
  wire yosys__02_;
  wire yosys__03_;
  wire yosys__04_;
  wire yosys__05_;
  wire yosys__06_;
  wire yosys__07_;
...
  cycloneiv_lcell_comb yosys__19_ (
    .combout(yosys__01_),
    .dataa(yosys__14_[2]),
    .datab(yosys__14_[3]),
    .datac(yosys__14_[4]),
    .datad(1'b1)
  );
  defparam yosys__19_.lut_mask = 16'b0000000100000001;
  defparam yosys__19_.sum_lutc_input = "datac";
  /* src = "/usr/local/bin/../share/yosys/altera_intel/
cycloneiv/cells_map_cycloneiv.v:55" */
  cycloneiv_lcell_comb yosys__20_ (
    .combout(yosys__00_[0]),
    .dataa(yosys__15_),
    .datab(yosys__14_[0]),
    .datac(yosys__02_),
    .datad(yosys__17_)
  );
  defparam yosys__20_.lut_mask = 16'b0000000011110100;
  defparam yosys__20_.sum_lutc_input = "datac";
  /* src =
```

Рис.11. Фрагмент VQM-файла, сгенерированного инструментом синтеза Yosys для ПЛИС серии Cyclone IV

На рис.8 отображается функциональное моделирование с использованием симулятора компилятора Icarus-Verilog (устанавливается дополнительно с сайта http://iverilog.icarus.com/). По команде iverilog -o Icarus-Verilog транслирует Verilog-код в файл специального формата для симуляции проекта. После выполнения этой команды создается файл presynth, который будет использоваться для симуляции по команде vpp.

Убедившись в правильности функционирования LFSR-регистра, запускаем скрипт run_cycloneiv (рис.9). На рис.10 показан фрагмент отработки этого скрипта. При этом создается VQM-файл с именем top (top.vqm) (рис.11). Для корректной работы в САПР Quartus этот файл необходимо переименовать в lfsr_updown (lfsr_updown.vqm) или отредактировать скрипт (имя файла должно совпадать с названием модуля).

На рис.12а показано RTL-представление, построенное с использованием VQM-файла, сгенерированного инструментом Yosys в Quartus на стадии анализа и синтеза, а на рис.12б – RTL-представление на стадии размещения и трассировки (после полной компиляции) в ПЛИС EP4CGX150DF31C7. В частности, на рисунке видно, что сетевой примитив CYCLONEIV_LCELL_COMB (условный номер yosys_30_) списка соединений VQM-файла, сгенерированного Yosys, пре-

Общие сведения о количестве задействованных ресурсов ПЛИС Altera EP4CGX150DF31C7 и временные параметры модели slow-model TimeQuest-анализа после полной компиляции

Проект	Общее количество логических элементов	Число триггеров логических элементов	Максимальная рабочая частота, МГц	Рабочая частота в худшем случае, МГц
По Verilog-коду (инструмент синтеза QIS Quartus Prime)	15	8	571,43	250
С использованием VQM-файла, сгенерированного инструментом синтеза Yosys	21	8	508,65	250

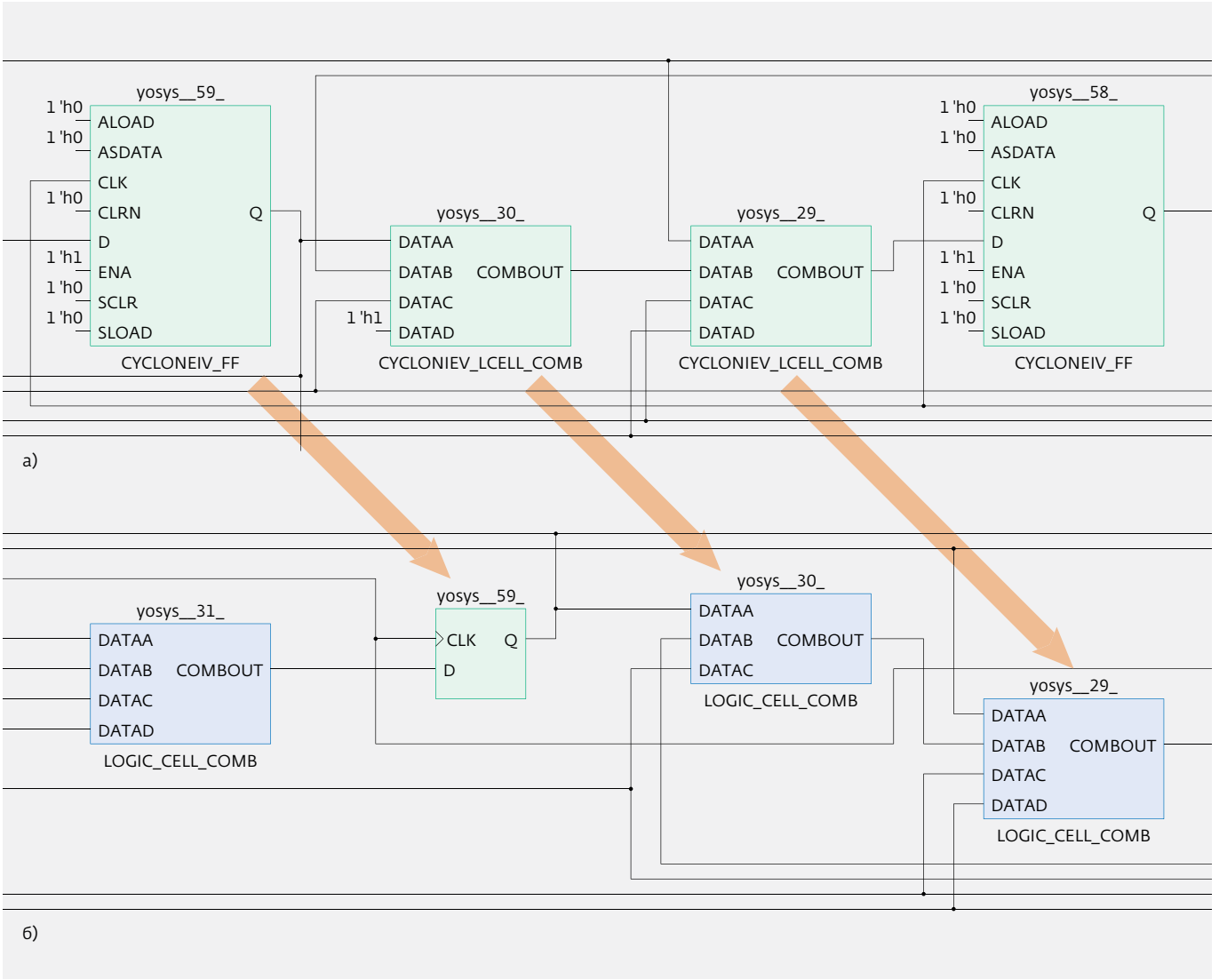

Рис.12. Преобразование сетевых примитивов списка соединений VQM-файла, сгенерированного инструментом Yosys в САПР Quartus: от стадии анализа и синтеза (а) к стадии размещения и трассировки (б) в базисе ПЛИС EP4CGX150DF31C7

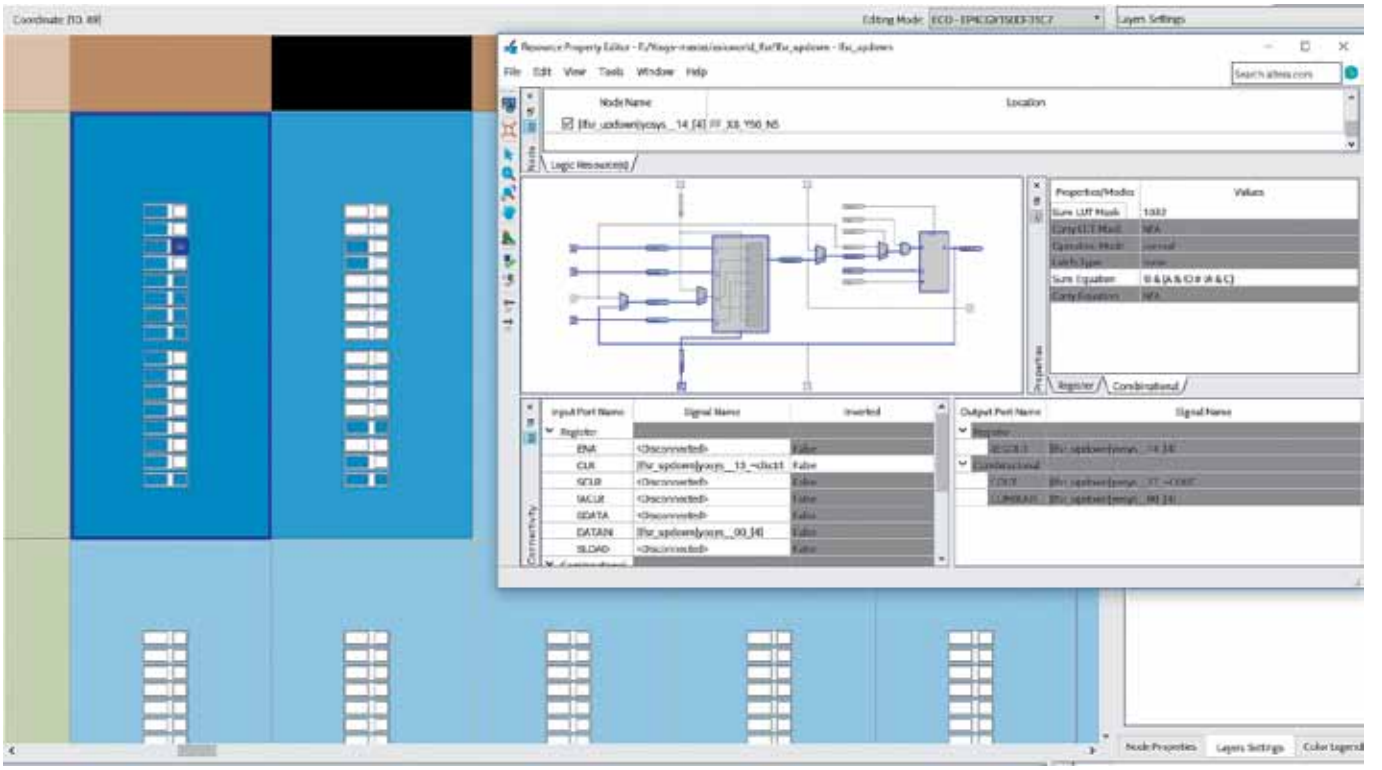

Рис.13. Задействованные логические ресурсы ПЛИС EP4CGX150DF31C7 при реализации проекта, представленного VQM-файлом, сгенерированным инструментом Yosys

образуется в примитив LOGIC_CELL_COMB списка соединений на стадии размещения и трассировки. Рис.13 отражает задействованные логические ресурсы ПЛИС EP4CGX150DF31C7, упакованные в кластеры из адаптивных логических модулей с указанием месторасположения на кристалле (выделены синим).

В таблице приведены общие сведения по числу задействованных ресурсов ПЛИС Altera EP4CGX150DF31C7 и временные параметры модели slow-model TimeQuest-анализа (напряжение питания ядра 1200 мВ, температура 85°С) без использования файла временных ограничений, заданных пользователем. Сравнивая значения, представленные в таблице, можно сделать вывод, что инструмент синтеза QIS САПР Quartus Prime более эффективно упаковывает комбинационную логику проекта, а число триггеров в обоих случаях остается одинаковым, что объясняется дополнительной оптимизацией, которую выполняет САПР Quartus.

В заключение остается отметить, что с помощью программного инструмента для Verilog-синтеза с открытым программным кодом (в данном случае инструмента синтеза Yosys новой версии 0.7+194) можно реализовывать цифровые устройства и в базис самых современных индустриальных ПЛИС Intel FPGA серии MAX 10 посредством формирования VQM-файла. Для отладки проектов можно использовать плату DE2i-150.

ЛИТЕРАТУРА

1. **Строгонов А., Цыбин С., Городков П.** Использование синтезатора Synplicity Synplify для разработки проектов цифровых устройств в Altera Quartus II // Компоненты и технологии, 2016, № 5. С. 96–100.
2. **Строгонов А., Городков П.** Реализация Verilog-проектов в базисе заказных БИС и ПЛИС с использованием инструмента синтеза Yosys // ЭЛЕКТРОНИКА: Наука, Технология, Бизнес. 2017. № 5. С. 98–109.
3. **Строгонов А., Городков П.** Реализация Verilog-проектов в базисе индустриальных ПЛИС Xilinx с применением синтезатора Yosys // Компоненты и технологии. 2017. № 6. С. 104–107.
4. **Строгонов А., Городков П.** Реализация Verilog-проектов в базисе академических ПЛИС с применением САПР VTR7.0 // Компоненты и технологии. 2017. № 5. С. 12–17.
5. **Строгонов А., Городков П.** Программные средства с открытым исходным кодом для проектирования цифровых устройств в базисах БИС и ПЛИС // Компоненты и технологии. 2017. № 3. С. 88–97.
6. http://www.clifford.at/yosys/
7. https://github.com/cliffordwolf/yosys