**УДК 681.5**

# ПРОЦЕДУРА ТРАНСЛЯЦІЇ ТАБЛИЧНОГО ОПИСУ ЦИФРОВИХ ПРИСТРОЇВ У ПРОГРАМИ НА МОВАХ ОПИСУ АПАРАТУРИ

**Малиновський М. Л., Коніщева А. П.**

*Харьковский национальный технический университет сельского хозяйства имени Петра Василенко*

*Запропоновано процедуру трансляції опису цифрових пристроїв, зробленого за допомогою табличної мови THDL, в опис текстовою мовою Verilog.*

**Постановка проблеми.** Як відомо, існуючі середовища проектування цифрових пристроїв (ISE, Quartus і ін.) підтримують інструментальні засоби, які можна розділити на текстові та графічні. До текстових належать мови опису апаратури VHDL, Verilog та ін., До графічних – засоби структурного опису (у вигляді блокових діаграм) і поведінкового опису (у вигляді діаграм станів).

Графічні засоби наочні та компактні. Їх використання сприяє спрощенню взаємодії та покращенню взаєморозуміння в колективі розробників. Програми, підготовлені в графічних редакторах, краще піддаються аналізу та корекції.

Тим не менш, більшість розробників в якості основного використовують текстовий інструментарій. Такий стан справ пояснюється як суб'єктивними, так і об'єктивними факторами, в тому числі тим, що 1) текст не прив'язаний до редактора, 2) текстові засоби володіють великими можливостями і гнучкістю при описі пристроїв, що вимагають використання вкладених конструкцій, параметрізуємих пристроїв з регулярною архітектурою і т.д.

Таким чином, існує суперечність між наочністю одних засобів та універсальністю інших. На думку авторів, табличні засоби здатні в значній мірі вирішити дане протиріччя.

Тим не менш, універсальні табличні мови, які дозволили б повною мірою використовувати всі переваги таблиць, до теперішнього часу не розвивалися.

У зв'язку з цим актуальною є проблема розробки табличних інструментальних засобів опису цифрових пристроїв і трансляції табличного опису в текстові формати, які підтримуються існуючими компіляторами.

**Аналіз публікацій.** Опис табличних мов проектування наведено в [1], [2], [4] опис мови Verilog в [3]. Аналіз публікацій підтверджує наявність передумов застосування табличних мов і актуальність завдання розробки транслятора з табличної мови в текстову мову, що компілюється.

**Метою дослідження**, проведеного авторами, є забезпечення можливості практичного застосування гнучкої та універсальної табличної мови для опису цифрових пристроїв.

**Основні матеріали дослідження**

**1. Загальні відомості про THDL**

THDL (Tabular Hardware Description Language) – таблична мова опису цифрових пристроїв.

У програмі на THDL (як і в інших HDL-програмах) виділяються дві частини:

1) інтерфейсна – оголошення імені компонента і опис сигналів;

2) логічна – опис логіки роботи пристрою.

Основними табличними конструкціями першої частини є COMPONENT і BINARY SIGNALS. На рисунку 1 наведено приклад використання даних конструкцій. Зліва за допомогою мови THDL оголошений компонент test_thdl і описані його сигнали різних типів і розрядності. Справа – ті ж дані, але описані за допомогою мови Verilog.

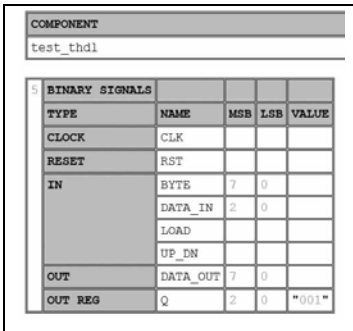
В логічній частині використовуються конструкції, що описують дії (ACTION, FUNCTION) та умови, при яких виконуються ці дії (CONDITION, PRIORITY CONDITION, ARGUMENT, PRIORITY ARGUMENT).

Приклад використання конструкцій логічної частини в мовах THDL та Verilog наведений на рисунку 2:

а) Конструкція ACTION – інкрементується значення регістрового сигналу Q і присвоюється значення комбінаційному сигналу DATA_OUTP;

б) Конструкція ACTION доповнена умовою з пріоритетом PRIORITY CONDITION. Якщо отримано сигнал LOAD, виконується дія Q <= DATA_IN. Якщо сигнал LOAD не отримано, а отримано UP_DN, сигнал Q інкрементується. При неотриманні жодного з перерахованих сигналів значення Q декрементується;

в) Конструкції PRIORITY ARGUMENT і FUNCTION. При отриманні сигналу addr_ld перевіряється значення аргументу addr_in. За результатами перевірки функціям w_match і r_match присвоюються відповідні значення.



COMPONENT

test_thdl

BINARY SIGNALS

| TYPE | NAME | MSB | LSB | VALUE |
|---|---|---|---|---|
| CLOCK | CLK | | | |
| RESET | RST | | | |
| IN | BYTE | 7 | 0 | |
| | DATA_IN | 2 | 0 | |
| | LOAD | | | |
| | UP_DN | | | |
| OUT | DATA_OUT | 7 | 0 | |
| OUT REG | Q | 2 | 0 | "001" |

```
module test_thdl (CLK,
RST, BYTE, DATA_IN,
LOAD, UP_DN,
DATA_OUT, Q);

input CLK, RST, LOAD,
UP_DN;
input [7:0] BYTE;
input [2:0] DATA_IN;

output [7:0] DATA_OUT;
output [2:0] Q;
wire CLK;
wire RST;
wire [7:0] BYTE;
wire [2:0] DATA_IN;
wire LOAD;
wire UP_DN;

reg [7:0] DATA_OUT;
reg [2:0] Q;
```

Рисунок 1 – Опис інтерфейсної частини пристрою на THDL та Verilog

а)

| 3 | ACTION |
|---|---|
| | Q = Q + 1 |
| | DATA_OUTP = !BYTE |

```
always @(*)
 begin
  DATA_OUTP <= !BYTE;
 end
always @(posedge CLK)
 begin
  Q <= Q + 1;
 end
```

б)

| 2 | PRIORITY CONDITION | ACTION |
|---|---|---|
| | LOAD | Q = DATA_I |
| | UP_DN | Q = Q + 1 |
| | ELSE | Q = Q - 1 |

```
always @(posedge CLK)
 begin
  if (LOAD)   begin
    Q <= DATA_IN;
   end
  else if (UP_DN)
   begin
    Q <= Q + 1;
   end
  else  begin
    Q <= Q - 1;
  end
 end
```

```
always @(posedge CLK)
 begin
  if (addr_ld)   begin
   if (addr_in == "1010")
    begin
     w_match <= 1;
     r_match <= 0;
    end
   else if (addr_in == "1011")
    begin
     w_match <= 0;
     r_match <= 1;
    end
   else  begin
     w_match <= 0;
     r_match <= 0;
    end
   else  begin
    w_match <= 0;
     r_match <= 0;
  end  end
```

в)

| PRIORITY ARGUMENT | | FUNCTION | |
|---|---|---|---|
| addr_ld | addr_in | w_match | r_match |
| 1 | "1010" | 1 | 0 |
| | "1011" | 0 | 1 |
| | else | 0 | 0 |
| ELSE | | 0 | 0 |

Рисунок 2 – Елементи опису логічної частини пристрою на THDL и Verilog

## 2. Алгоритм трансляції

Розроблений транслятор забезпечує можливість введення опису в табличному форматі на мові THDL і трансляцію цього опису в мову Verilog.

Алгоритм трансляції розглянемо на прикладі реалізації лічильника за модулем 6 (рис. 3).

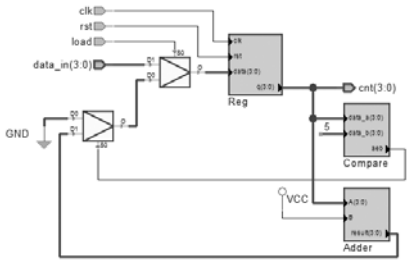



Рисунок 3 – RTL-схема лічильника за модулем 6

Дану схему можна описати за допомогою табличних та текстових конструкцій (рис. 4). Трансляція полягає в перетворенні табличних конструкцій в текстові, при цьому схемна реалізація пристрою повинна залишатися ідентичною.

Як згадувалося вище, будь-яка програма складається з двох частин - інтерфейсної і частини, яка описує логіку роботи пристрою. Алгоритм трансляції інтерфейсної частини залежить від синтаксису мови і не викликає складнощів. Розглянемо алгоритм трансляції частини, яка описує логіку роботи.

| COMPONENT |
|---|
| mod_6_counter |

| 1 | BINARY SIGNALS | | | | |
|---|---|---|---|---|---|
| | TYPE | NAME | MSB | LSB | VALUE |
| | CLOCK | clk | | | |
| | RESET | rst | | | |
| | IN | data_in | 3 | 0 | |
| | | load | | | |
| | OUT REG | cnt | 3 | 0 | |

| 2 | PRIORITY CONDITION | ACTION |
|---|---|---|
| | load == 1 | cnt = data_in |
| | cnt == 5 | cnt = 4'b0000 |
| | ELSE | cnt = cnt + 1'b1 |

| END |
|---|
| mod_6_counter |

```
Module
mod_6_counter       (
input wire clk,
 input wire rst,
 input wire load,
 input              wire
[3:0]data_in,
 output reg [3:0]cnt);
always  @  (posedge
clk)
 begin  if (rst)
   cnt <= 4'b0000;
  else   begin
    if(load)
     cnt <= data_in;
    else
     if (cnt == 5)
      cnt <= 4'b0000;
     else
      cnt <= cnt + 1'b1;
   end end  endmodule
```

Рисунок 4 – Табличний та текстовий описи лічильника за модулем 6

Оскільки табличні конструкції цієї частини описують тільки логіку і не враховують тип сигналу (комбінаційний або регістровий), зручно розбити схему на два блоки – комбінаційний і регістровий (рис. 5) і далі для кожної з цих частин формувати опис у текстовому форматі.

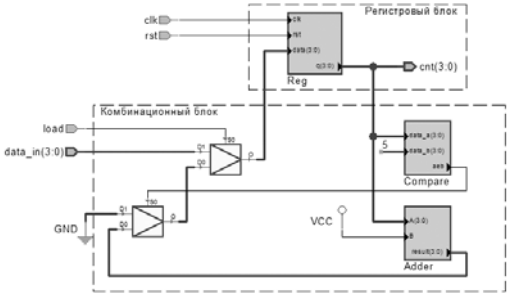



Рисунок 5 – Регістровий та комбінаційний блоки RTL-схеми

Всі обчислення і присвоювання будемо проводити в комбінаційному блоці, а збереження результатів - в регістровому. Для цього для кожного регістрового сигналу створимо допоміжний сигнал такої ж розрядності. Надамо йому ім'я основного сигналу з приставкою next_. Таким чином, для регістрових сигналів з'явиться два значення – збережене (використовується в умовах і в правій частині виразів присвоювання) і поточне – з приставкою next_ (використовується в регістровому блоці в лівій частині виразів присвоювання). В комбінаційному блоці кожну табличну конструкцію будемо транслювати наступним чином:

1) CONDITION, ARGUMENT → if (condition_1)…if (condition_n)…if (!condition_1 and…and !condition_n

2) PRIORITY CONDITION, PRIORITY ARGUMENT → if (condition_1)…else if (condition_n)…else

Для збереження результатів в регістровому блоці регістровим сигналам необхідно присвоїти значення, визначене в комбінаційному блоці (сигнали з приста-

вкою next_). Т. ч., послідовність заповнення шаблону текстового опису відповідає рисунку 6.

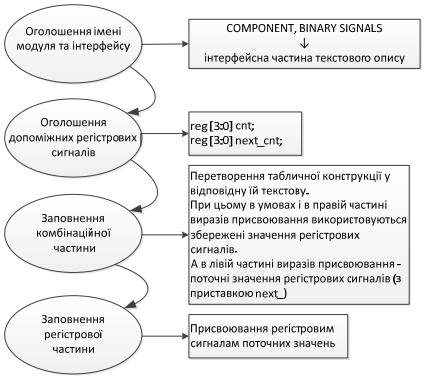



Рисунок 6 – Послідовність дій при трансляції табличного опису в текстовий

**3. Табличний редактор проектування цифрових пристроїв**

Для проектування цифрових пристроїв з використанням табличних конструкцій створено редактор EditorTHDL (рис. 7). Починати роботу необхідно зі створення проекту (а) і компонента в даному проекті (б). Далі зі списку існуючих табличних конструкцій вибирають необхідну (в). Програма являтиме собою перелік певних табличних конструкцій (г). Після створення табличного опису розробник запускає трансляцію і отримує Verilog-код (д).

**Висновки**

1. Розроблена таблична мова THDL є універсальним засобом проектування цифрових пристроїв і дозволяє підвищити якість і знизити трудовитрати на створення програмного забезпечення.

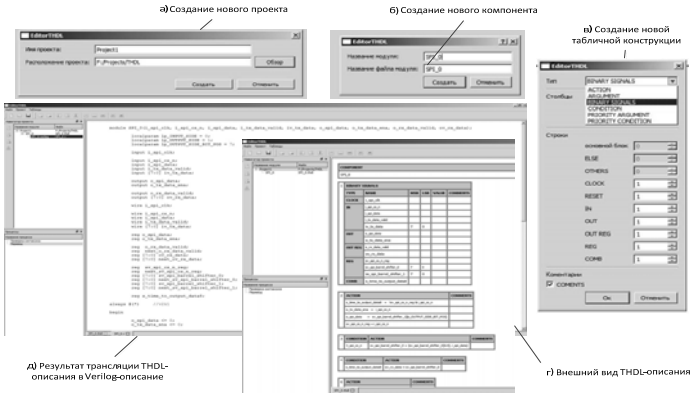



Рисунок 7 – Інтерфейс редактора EditorTHDL

2. Розроблена процедура трансляції та табличний редактор EditorTHDL дає можливість використовувати мову THDL при вирішенні практичних завдань, пов'язаних з проектуванням цифрових пристроїв на основі замовних і програмованих інтегральних схем.

**Список використаних джерел**

1. *Малиновський М. Л.* Методи і засоби табличного опису апаратури цифрових пристроїв / М. Л. Малиновський, А. П. Коніщева, А.В. Сидоренко // Інформаційно-керуючі системи на залізничному транспорті – УкрДАЗТ – 2011. – №4. – С. 69–72

2. *Languages* and General Software Aspects for Telecommunication Systems. The Tree and Tabular Combined Notation. v.3 07/2001 ITU [Internet resource]. – Access mode http://www.itu.int/ITU-T/studygroups/com10/languages/Z.140_0701_pre.pdf

3. *IEEE Standard Verilog*. 03/2001 [Internet resource]. – Access mode http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?reload=true&punumber=10779

4. *Малиновський М. Л.* Концепція створення табличних мов опису апаратури / М. Л. Малиновський, І. А. Фурман, А. Ю. Аллашев, А. В. Святобатько // Радіоелектронні і комп'ютерні системи. – НАУ ХАІ, 2010. – №6. – С. 289–291

**Аннотация**

**ПРОЦЕДУРА ТРАНСЛЯЦИИ ТАБЛИЧНОГО ОПИСАНИЯ УСТРОЙСТВ В ПРОГРАММЫ НА ЯЗЫКАХ ОПИСАНИЯ АППАРАТУРЫ**

Малиновский М. Л., Конищева А. П.

*Предложена процедура трансляции описания цифровых устройств, выполненного при помощи табличного языка THDL, в текстовое описание на языке Verilog.*

**Abstract**

**THE TRANSLATION PROCEDURE OF TABULAR DESCRIPTION HARDWARE TO PROGRAM ON TEXT SINTESIZED LANGUAGE**

M. Malynovskyi, A. Konishcheva

*Translation procedure from tabular description hardware (THDL) to text description (Verilog) is offered.*