بسمه تعالی

گزارش کار سوم آزمایشگاه طراحی سیستم‌های دیجیتال

توصیف جریان داده

استاد:

دکتر علیرضا اجلالی

نویسندگان:

امیررضا آذری ۹۹۱۰۱۰۸۷

غزل طحان ۹۹۱۰۶۳۷۴

بزرگمهر ضیا ۹۹۱۰۰۴۲۲

دانشگاه صنعتی شریف

تابستان ۱۴۰۲

فهرست

هدف

هدف از این آزمایش آشنایی با توصیف جریان داده است. این توصیف با دستور assign در وریلاگ شناخته شده و جریان یک سیم به وسیله آن مشخص می‌شود. برای آشنایی با این دستور، ابتدا یک مقایسه کننده یک بیتی (Cascadable 1-bit comparator) ساخته و سپس با اتصال ۴ تا از این مقایسه‌کننده‌ها یک مقایسه کننده ۴ بیتی می‌سازیم. سپس یک ماژول مقایسه‌کننده سریال می‌سازیم که یک مدار ترتیبی است که از دو ورودی خود بیت‌های دو عددی که باید مقایسه شوند را بیت به بیت دریافت نموده و در هر پالس ساعت حاصل مقایسه را تا جایی که مقایسه کرده (تا بیتی که مقایسه انجام شده) در خروجی سریال خود تحویل می‌دهد. نکته مهم این است که در ساخت مدارهای مذکور، تنها مجاز به استفاده از توصیف جریان داده هستیم.

توجه شود که پیاده سازی این آزمایش روی برد با آزمایش ۴ است.

بخش اول _ Cascadable 1-bit comparator

طراحی ماژول:

در ابتدا می‌خواهیم یک مقایسه‌کننده یک بیتی طراحی کنیم که چهار ورودی input_greater_than، input_equal، x و y را می‌گیرد و دو خروجی output_greater_than و output_equal را می‌دهد. ورودی‌های input_greater_than و input_equal مشخص می‌کنند که نتیجه مقایسه بیت‌های قبلی به ترتیب بزرگتر و مساوی بوده است. x و y نیز ورودی‌های اصلی مدار هستند. خروجی‌های output_greater_than و output_equal نیز به ترتیب بزرگتر بودن (x > y) و تساوی ورودی‌ها (x = y) را نشان می‌دهند. طبیعتا ۰ بودن هردو خروجی به معنای کوچکتر بودن پاسخ (x < y) است.

خروجی output_greater_than زمانی برابر با ۱ است که یا نتیجه مقایسه بیت‌های قبلی برابر با بزرگتر باشد، یا این نتیجه برابر با تساوی بوده و ورودی x از y بزرگتر باشد که در این صورت برابر با مقدار زیر می شود.

```
output_greater_than = input_greater_than | (input_equal & (x > y))
```

خروجی output_equal زمانی برابر با ۱ است که نتیجه مقایسه بیت‌های قبلی برابر با تساوی باشد و x = y باشد که برابر با مقدار زیر می‌شود.

```
output_equal = input_equal & (x == y)
```

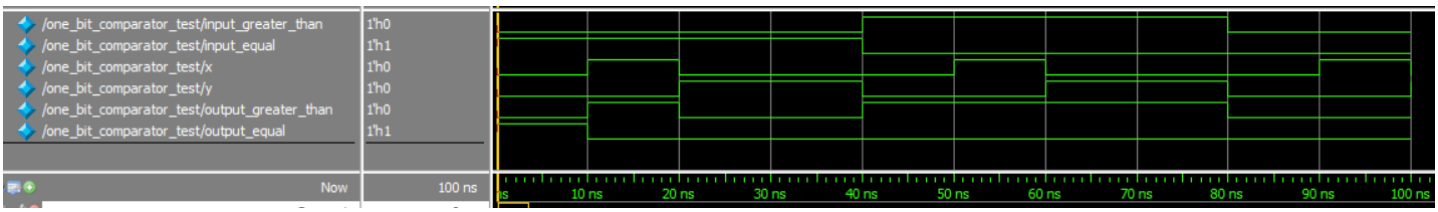
در نهایت کد وریلاگ این ماژول به صورت زیر در می‌آید:

```verilog
module one_bit_comparator (
    input input_greater_than,
    input input_equal,
    input x,
    input y,
    output output_greater_than,
    output output_equal
);
    assign output_greater_than = input_greater_than | (input_equal & (x > y));
    assign output_equal = input_equal & (x == y);
endmodule
```

شکل ۱. کد وریلاگ ماژول مقایسه کننده یک بیتی

تست ماژول :

برای تست صحت کارکرد ماژول، تست بنچ در فایل one_bit_comparator_test نوشته شد که در آن تعدادی از حالات بررسی شدند. شکل موج این تست‌ها در شکل زیر قابل مشاهده است.



شکل ۲. شکل موج تست بنچ ماژول مقایسه کننده یک بیتی

برای مثال، از زمان ۱۰ تا ۲۰، مقادیر ورودی به صورت زیر هستند:

input_greater_than = 0

input_equal = 1

x = 1

y = 0

که به این معناست که نتیجه مقایسه بیت‌های قبلی برابر تساوی بوده و نتیجه مقایسه بیت‌های جدید نشان‌دهنده x > y است که بیت خروجی output_greater_than باید برابر با ۱ شود که در شکل موج نیز همین‌طور است. باقی تست-کیس‌ها نیز به همین ترتیب درستی مدار را نشان می‌دهند.

بخش دوم _ مقایسه کننده ۴ بیتی

طراحی ماژول:

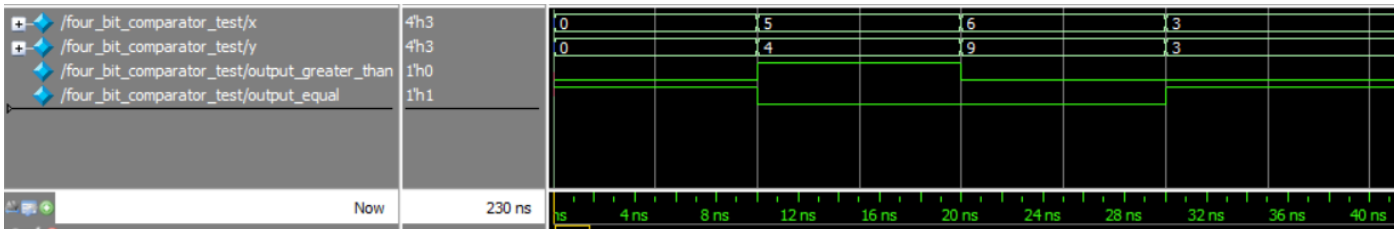
در این بخش با اتصال ماژول‌های بخش قبل سعی داریم یک مقایسه‌کننده ۴ بیتی را طراحی کنیم. این مدار دو ورودی سه بیتی x و y و دو خروجی output_greater_than و output_equal دارد و عملکرد آن مشابه قسمت قبل است. برای طراحی آن ۴ ماژول مقایسه‌کننده یک بیتی را به هم متصل می‌کنیم. به این صورت که خروجی‌های output_greater_than و output_equal هر ماژول را به ترتیب به ورودی‌های input_greater_than و input_equal ماژول بعدی متصل می‌کنیم. خروجی output_greater_than و output_equal ماژول نهایی نیز به خروجی output_greater_than و output_equal مدار وصل می‌شود. همچنین به اولین ماژول ورودی‌های input_greater_than = 0 و input_equal = 1 می‌دهیم، زیرا در غیر این صورت مقایسه بلافاصله تمام می‌شود. در نهایت کد وریلاگ این ماژول به صورت زیر در می‌آید:

```
1   `include "one_bit_comparator.v"
2
3   module four_bit_comparator (
4       input[3:0] x,
5       input[3:0] y,
6       output output_greater_than,
7       output output_equal
8   );
9       wire c1_greater_than, c1_equal, c2_greater_than, c2_equal, c3_greater_than, c3_equal;
10      one_bit_comparator c1(1'b0,           1'b1,     x[3], y[3], c1_greater_than,     c1_equal);
11      one_bit_comparator c2(c1_greater_than, c1_equal, x[2], y[2], c2_greater_than,     c2_equal);
12      one_bit_comparator c3(c2_greater_than, c2_equal, x[1], y[1], c3_greater_than,     c3_equal);
13      one_bit_comparator c4(c3_greater_than, c3_equal, x[0], y[0], output_greater_than, output_equal);
14  endmodule
```

شکل ۳. کد وریلاگ ماژول مقایسه کننده ۴ بیتی

تست ماژول:

برای تست صحت کارکرد ماژول، تست بنچ در فایل four_bit_comparator_test نوشته شد که در آن تعدادی از حالات بررسی شدند. شکل موج این تست‌ها در شکل زیر قابل مشاهده است.



شکل ۴. شکل موج تست بنچ ماژول مقایسه کننده چهار بیتی

مطابق شکل، ۴ تست‌کیس داده شد که نتیجه آن‌ها قابل مشاهده است که به ترتیب تساوی، بزرگتر، کوچکتر و مجددا تساوی هستند و درستی عملکرد مدار را نشان می‌دهند.

بخش سوم _ مقایسه کننده سریال

طراحی ماژول:

در این بخش قصد داریم یک ماژول مقایسه‌کننده سریال بسازیم. این مقایسه‌کننده یک مدار ترتیبی است که با استفاده از ورودی reset در اول کار reset می‌شود و پس از آن از دو ورودی خود بیت‌های دو عددی که باید مقایسه شوند را بیت به بیت دریافت نموده و در هر پالس ساعت حاصل مقایسه را تا جایی که مقایسه کرده (تا بیتی که مقایسه انجام شده) در خروجی سریال خود تحویل می‌دهد. ورودی‌های این مدار x, y, clk, reset و خروجی‌های آن output_greater_than و output_less_than هستند که به ترتیب نشان‌دهنده x > y و x < y هستند. ۰ بودن هر دو خروجی به معنای x = y است.

برای طراحی این ماژول، ۴ wire جدید به نام‌های output_greater_than_not, output_less_than_not, input_greater_than, input_less_than تعریف می‌کنیم. سیگنال‌های input_greater_than, input_less_than نشانگر ورودی مقایسه بیت‌های قبلی مدار برای کلاک بعدی هستند. سیگنال‌های output_greater_than_not, output_less_than_not نشانگر مکمل خروجی‌های output_greater_than و output_less_than هستند.

سیگنال output_greater_than زمانی ۱ است که یا output_greater_than_not = 0 باشد یا input_greater_than = 1 باشد و کلاک نیز آمده باشد (clk = 1). به طور مشابه سیگنال output_greater_than_not زمانی ۱ است که یا output_greater_than = 0 باشد یا input_greater_than = 0 باشد و کلاک نیز آمده باشد (clk = 1). همین حالات نیز به طور مشابه برای سیگنال‌های output_less_than_not و output_less_than برقرار هستند.

سیگنال input_greater_than زمانی ۱ است که هر دو شرط زیر اتفاق بیفتند:

- reset = 0 باشد.
- یکی از دو شرط زیر اتفاق بیفتد:
  - output_greater_than = 1 باشد که به این معناست که تا الان نتیجه مقایسه بزرگتر بوده است.
  - input_less_than = 0 (به این معناست که تا الان نتیجه مقایسه کوچکتر نبوده است) و x > y باشد.

همین حالات به طور مشابه برای سیگنال input_less_than برقرار است.

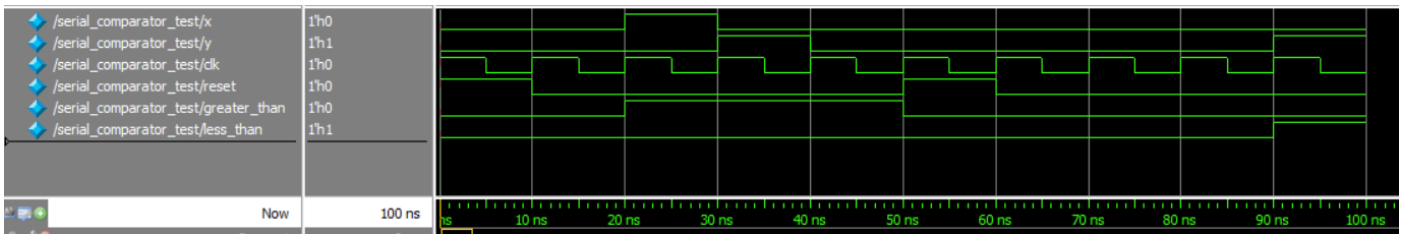
در نهایت کد وریلاگ این ماژول به صورت زیر در می‌آید:

```verilog
module serial_comparator (
    input x,
    input y,
    input reset,
    input clk,
    output output_greater_than,
    output output_less_than
);
    wire output_greater_than_not, output_less_than_not, input_greater_than, input_less_than;
    // Assign outputs
    assign output_greater_than = (~output_greater_than_not) | (input_greater_than & clk);
    assign output_greater_than_not = (~output_greater_than) | (~input_greater_than & clk);
    assign output_less_than = (~output_less_than_not) | (input_less_than & clk);
    assign output_less_than_not = (~output_less_than) | (~input_less_than & clk);
    // Assign inputs
    assign input_greater_than = (~reset) & (output_greater_than | ((~input_less_than) & (x > y)));
    assign input_less_than = (~reset) & (output_less_than | ((~input_greater_than) & (x < y)));
endmodule
```

شکل ۵. کد وریلاگ ماژول مقایسه کننده سریال

تست ماژول:

برای تست صحت کارکرد ماژول، تست بنچ در فایل serial_bit_comparator_test نوشته شد که در آن تعدادی از حالات بررسی شدند. شکل موج این تست‌ها در شکل زیر قابل مشاهده است.



شکل ۶. شکل موج تست بنچ ماژول مقایسه کننده سریال

در تست کیس اول (از زمان ۰ تا ۵۰)، ابتدا سیگنال ریست ۱ داده شد که مطابق شکل و به درستی هر دو خروجی برابر ۰ هستند. سپس ریست ۰ شده و ورودی‌های x = 0100 و y = 0010 بیت به بیت در هر پالس ساعت به مدار داده شدند که از بیت دوم به بعد سیگنال greater_than = 1 شد که درست است.

در تست کیس دوم (از زمان ۰ تا ۵۰)، پس از ریست کردن مدار ورودی‌های x = 0000 و y = 0001 بیت به بیت در هر پالس ساعت به مدار داده شدند که از بیت آخر به بعد سیگنال less_than = 1 شد که درست است.

نتیجه‌گیری:

در این آزمایش با توصیف جریان داده آشنا شدیم و مدارهای مقایسه کننده یک بیتی (Cascadable 1-bit comparator)، مقایسه کننده ۴ بیتی و مقایسه‌کننده سریال را به وسیله وریلاگ و تنها با استفاده از توصیف جریان داده طراحی و تست کردیم.

منابع و مراجع:

- S. Palnitkar. Verilog HDL: A Guide to Digital Design and Synthesis. 2nd Edition, Prentice Hall, 2003.
- ACEX 1K Programmable Logic Family Data Sheet. Available at www.altera.com.
- ModelSim User's Manual. Available at www.actel.com.
- Introduction to the Quartus II Software. Available at www.altera.com