به نام خدا

آزمایش سوم

آزمایشگاه طراحی سیستم‌های دیجیتال

دانشکده مهندسی کامپیوتر

دانشگاه صنعتی شریف

نویسندگان:

رادین چراغی ۴۰۱۱۰۵۸۱۵

امیرمحمد محفوظی ۴۰۱۱۰۶۴۶۹

سیدعلی جعفری ۴۰۰۱۰۴۸۸۹

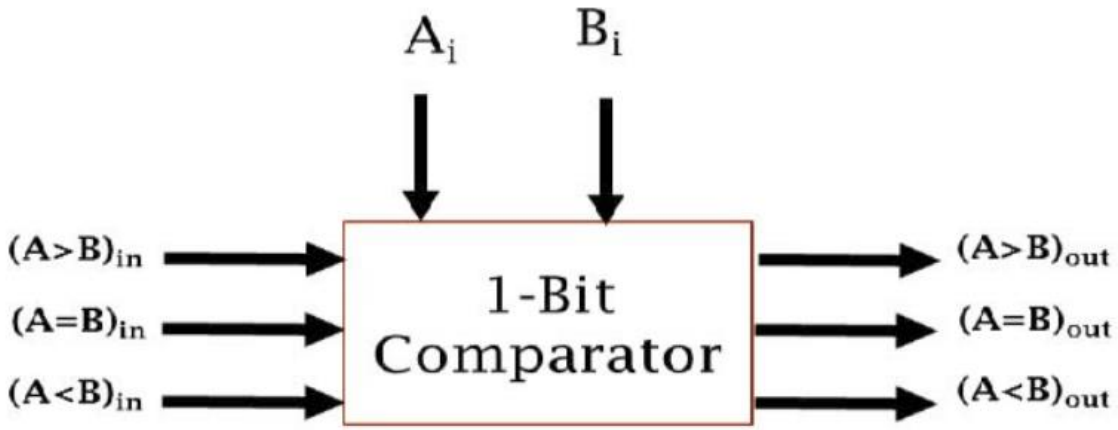
تاریخ ارائه تکلیف:

۱۴۰۳/۰۴/۲۵

بخش اول

در این بخش هدف آزمایش طراحی یک مقایسه‌کننده ۴ بیتی به صورت سلسله مراتبی و با استفاده از مقایسه‌کننده تک‌بیتی می‌باشد. روش کلی طراحی این مقایسه‌کننده به این‌صورت است که ابتدا یک Cascadable 1-bit comparator طراحی کرده و سپس با اتصال چهار عدد از این مقایسه‌کننده‌ها به یکدیگر یک مقایسه‌کننده ۴ بیتی می‌سازیم. مدار طراحی شده در این بخش ترکیبی می‌باشد.

ابتدا Cascadable 1-bit comparator را طراحی می‌کنیم. شمای کلی مدار به صورت زیر می‌باشد.



در این ماژول دو ورودی a و b بیت‌های ورودی‌ای می‌باشند که قرار است مقایسه شوند. همچنین از آن‌جایی که این مقایسه‌کننده Cascadable می‌باشد این ماژول سه ورودی lt_in، gt_in و eq_in را دارد که به ترتیب نشان‌دهنده کوچکتر بودن، بزرگتر بودن و مساوی بودن بیت a مقایسه کننده طبقه قبلی می‌باشند. در نهایت سه سیگنال lt، gt و eq خروجی‌های این ماژول می‌باشند که به ترتیب نشان‌دهنده کوچکتر بودن، بزرگتر بودن و مساوی بودن بیت a نسبت به بیت b می‌باشند.

```
module OneBitComparator (
    input wire a,
    input wire b,
    input wire lt_in,
    input wire gt_in,
    input wire eq_in,
    output wire lt,
    output wire gt,
    output wire eq
);
    assign lt = (eq_in & ~a & b) | lt_in;
    assign gt = (eq_in & a & ~b) | gt_in;
    assign eq = eq_in & (a == b);
endmodule
```

حال به دلیل دریافت ورودی‌های lt_in، gt_in و eq_in می‌پردازیم. از آن‌جایی که در مقایسه اعداد، بیت‌های پرارزشتر از اهمیت بیشتری برخوردارند، در مقایسه‌کننده یک‌بیتی ورودی‌های مذکور را دریافت میکنیم تا وضعیت مقایسه اعداد در بیت‌های پرارزشتر آشکار شوند و اگر بزرگتر یا کوچکتر بودن عدد در ارقام پرارزشتر مشخص شده بود خروجی این مقایسه‌کننده نیز به همین ترتیب تکرار میشود.

در آخر با کنار هم قرار دادن ۴ مقایسه کننده یک‌بیتی یک مقایسه کننده ۴ بیتی می‌سازیم.

```
module FourBitComparator (
    input wire [3:0] A,
    input wire [3:0] B,
    output wire lt,
    output wire gt,
    output wire eq
);
    wire [3:0] lt_internal;
    wire [3:0] gt_internal;
    wire [3:0] eq_internal;

    // Instantiate the 1-bit comparators
    OneBitComparator comp3 (
        .a(A[3]), .b(B[3]), .lt_in(1'b0), .gt_in(1'b0), .eq_in(1'b1),
        .lt(lt_internal[3]), .gt(gt_internal[3]), .eq(eq_internal[3])
    );
    OneBitComparator comp2 (
        .a(A[2]), .b(B[2]), .lt_in(lt_internal[3]), .gt_in(gt_internal[3]), .eq_in(eq_internal[3]),
        .lt(lt_internal[2]), .gt(gt_internal[2]), .eq(eq_internal[2])
    );
    OneBitComparator comp1 (
        .a(A[1]), .b(B[1]), .lt_in(lt_internal[2]), .gt_in(gt_internal[2]), .eq_in(eq_internal[2]),
        .lt(lt_internal[1]), .gt(gt_internal[1]), .eq(eq_internal[1])
    );
    OneBitComparator comp0 (
        .a(A[0]), .b(B[0]), .lt_in(lt_internal[1]), .gt_in(gt_internal[1]), .eq_in(eq_internal[1]),
        .lt(lt_internal[0]), .gt(gt_internal[0]), .eq(eq_internal[0])
    );

    // The final lt, gt, and eq outputs
    assign lt = lt_internal[0];
    assign gt = gt_internal[0];
    assign eq = eq_internal[0];
endmodule
```

تست‌بنچ مربوط به این بخش را به صورت زیر طراحی می‌کنیم. در این ماژول تمامی ترکیب‌های مختلف a و b تست شده است.

تصویر زیر نشان‌دهنده خطوط ابتدایی ماژول می‌باشد.

```
module tb;
    reg [3:0] A;
    reg [3:0] B;
    wire lt, gt, eq;

    // Instantiate the 4-bit comparator
    FourBitComparator uut (
        .A(A),
        .B(B),
        .lt(lt),
        .gt(gt),
        .eq(eq)
    );
```

تصاویر زیر نشان‌دهنده تست‌های مختلف در ماژول tb می‌باشد.

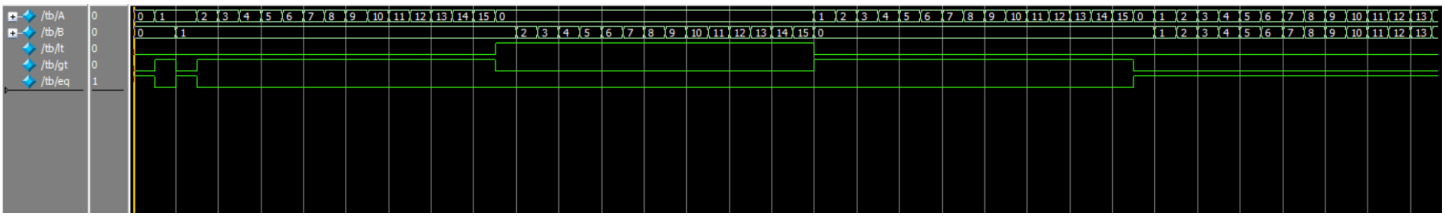
```
// Test all possible combinations
A = 4'b0000; B = 4'b0000; #10;
A = 4'b0001; B = 4'b0000; #10;
A = 4'b0001; B = 4'b0001; #10;
A = 4'b0010; B = 4'b0001; #10;
A = 4'b0011; B = 4'b0001; #10;
A = 4'b0100; B = 4'b0001; #10;
A = 4'b0101; B = 4'b0001; #10;
A = 4'b0110; B = 4'b0001; #10;
A = 4'b0111; B = 4'b0001; #10;
A = 4'b1000; B = 4'b0001; #10;
A = 4'b1001; B = 4'b0001; #10;
A = 4'b1010; B = 4'b0001; #10;
A = 4'b1011; B = 4'b0001; #10;
A = 4'b1100; B = 4'b0001; #10;
A = 4'b1101; B = 4'b0001; #10;
A = 4'b1110; B = 4'b0001; #10;
A = 4'b1111; B = 4'b0001; #10;
```

```
// Test all combinations where A is less than B
A = 4'b0000; B = 4'b0001; #10;
A = 4'b0000; B = 4'b0010; #10;
A = 4'b0000; B = 4'b0011; #10;
A = 4'b0000; B = 4'b0100; #10;
A = 4'b0000; B = 4'b0101; #10;
A = 4'b0000; B = 4'b0110; #10;
A = 4'b0000; B = 4'b0111; #10;
A = 4'b0000; B = 4'b1000; #10;
A = 4'b0000; B = 4'b1001; #10;
A = 4'b0000; B = 4'b1010; #10;
A = 4'b0000; B = 4'b1011; #10;
A = 4'b0000; B = 4'b1100; #10;
A = 4'b0000; B = 4'b1101; #10;
A = 4'b0000; B = 4'b1110; #10;
A = 4'b0000; B = 4'b1111; #10;
```

```
// Test all combinations where A is greater than B
A = 4'b0001; B = 4'b0000; #10;
A = 4'b0010; B = 4'b0000; #10;
A = 4'b0011; B = 4'b0000; #10;
A = 4'b0100; B = 4'b0000; #10;
A = 4'b0101; B = 4'b0000; #10;
A = 4'b0110; B = 4'b0000; #10;
A = 4'b0111; B = 4'b0000; #10;
A = 4'b1000; B = 4'b0000; #10;
A = 4'b1001; B = 4'b0000; #10;
A = 4'b1010; B = 4'b0000; #10;
A = 4'b1011; B = 4'b0000; #10;
A = 4'b1100; B = 4'b0000; #10;
A = 4'b1101; B = 4'b0000; #10;
A = 4'b1110; B = 4'b0000; #10;
A = 4'b1111; B = 4'b0000; #10;
```

```
// Test all combinations where A is equal to B
A = 4'b0000; B = 4'b0000; #10;
A = 4'b0001; B = 4'b0001; #10;
A = 4'b0010; B = 4'b0010; #10;
A = 4'b0011; B = 4'b0011; #10;
A = 4'b0100; B = 4'b0100; #10;
A = 4'b0101; B = 4'b0101; #10;
A = 4'b0110; B = 4'b0110; #10;
A = 4'b0111; B = 4'b0111; #10;
A = 4'b1000; B = 4'b1000; #10;
A = 4'b1001; B = 4'b1001; #10;
A = 4'b1010; B = 4'b1010; #10;
A = 4'b1011; B = 4'b1011; #10;
A = 4'b1100; B = 4'b1100; #10;
A = 4'b1101; B = 4'b1101; #10;
A = 4'b1110; B = 4'b1110; #10;
A = 4'b1111; B = 4'b1111; #10;
```

مدار را در نرم‌افزار ModelSim شبیه‌سازی می‌کنیم. بخشی از خروجی waveform و transcript در تصاویر زیر قابل مشاهده می‌باشد.



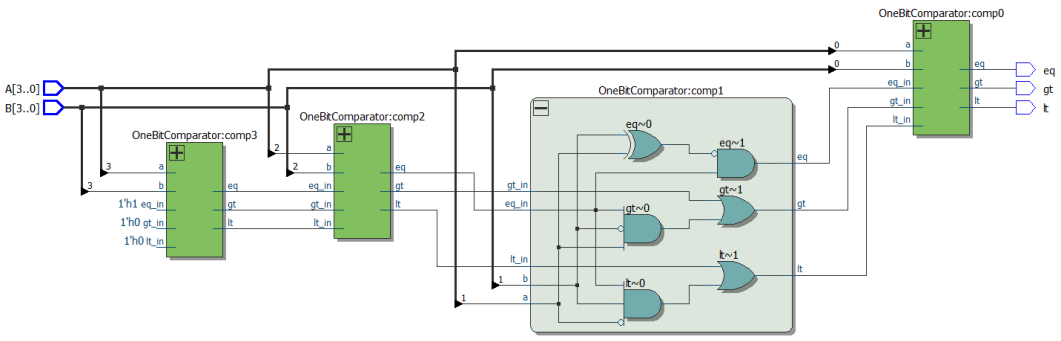
```
# A = 0010, B = 0000, lt = 0, gt = 1, eq = 0
# A = 0011, B = 0000, lt = 0, gt = 1, eq = 0
# A = 0100, B = 0000, lt = 0, gt = 1, eq = 0
# A = 0101, B = 0000, lt = 0, gt = 1, eq = 0
# A = 0110, B = 0000, lt = 0, gt = 1, eq = 0
# A = 0111, B = 0000, lt = 0, gt = 1, eq = 0
# A = 1000, B = 0000, lt = 0, gt = 1, eq = 0
# A = 1001, B = 0000, lt = 0, gt = 1, eq = 0
# A = 1010, B = 0000, lt = 0, gt = 1, eq = 0
# A = 1011, B = 0000, lt = 0, gt = 1, eq = 0
# A = 1100, B = 0000, lt = 0, gt = 1, eq = 0
# A = 1101, B = 0000, lt = 0, gt = 1, eq = 0
# A = 1110, B = 0000, lt = 0, gt = 1, eq = 0
# A = 1111, B = 0000, lt = 0, gt = 1, eq = 0
# A = 0000, B = 0000, lt = 0, gt = 0, eq = 1
# A = 0001, B = 0001, lt = 0, gt = 0, eq = 1
# A = 0010, B = 0010, lt = 0, gt = 0, eq = 1
# A = 0011, B = 0011, lt = 0, gt = 0, eq = 1
# A = 0100, B = 0100, lt = 0, gt = 0, eq = 1
# A = 0101, B = 0101, lt = 0, gt = 0, eq = 1
# A = 0110, B = 0110, lt = 0, gt = 0, eq = 1
# A = 0111, B = 0111, lt = 0, gt = 0, eq = 1
# A = 1000, B = 1000, lt = 0, gt = 0, eq = 1
# A = 1001, B = 1001, lt = 0, gt = 0, eq = 1
```

خروجی flow summary مدار در تصویر زیر قابل مشاهده است.

Flow Summary

Flow Status	Successful - Mon Jul 15 16:30:10 2024
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	Comparator
Top-level Entity Name	FourBitComparator
Family	Cyclone IV GX
Total logic elements	9 / 14,400 (< 1 %)
Total combinational functions	9 / 14,400 (< 1 %)
Dedicated logic registers	0 / 14,400 (0 %)
Total registers	0
Total pins	11 / 81 (14 %)
Total virtual pins	0
Total memory bits	0 / 552,960 (0 %)
Embedded Multiplier 9-bit elements	0
Total GXB Receiver Channel PCS	0 / 2 (0 %)
Total GXB Receiver Channel PMA	0 / 2 (0 %)
Total GXB Transmitter Channel PCS	0 / 2 (0 %)
Total GXB Transmitter Channel PMA	0 / 2 (0 %)
Total PLLs	0 / 3 (0 %)
Device	EP4CGX15BF14C6
Timing Models	Final

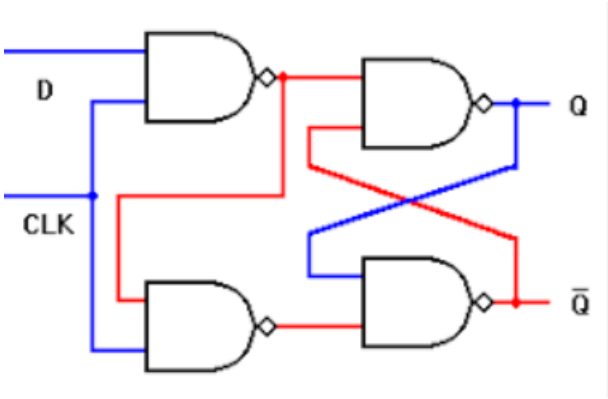
خروجی rtl_viewer مدار در تصویر زیر قابل مشاهده است. فایل pdf آن نیز در پیوست آورده شده است.



بخش دوم

در این بخش هدف آزمایش طراحی یک مقایسه‌کننده ۴ بیتی سریال می‌باشد. این مقایسه‌کننده یک مدار ترتیبی است که با استفاده از ورودی reset در اول کار reset شده و پس از آن از دو ورودی خود بیت های 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 می دهد. لازم به ذکر است توصیف صورت گرفته برای طراحی مدار باید فقط توصیف جریان داده باشد.

حال ماژول مقایسه‌کننده سریال را طراحی می‌کنیم. در این بخش فرض می‌کنیم که این ماژول بیت‌های اعداد را از بیت پرارزش به بیت کم‌ارزش دریافت می‌کند. روش طراحی ما برای این مدار به این‌صورت است که برای هر یک از خروجی‌های مدار یعنی gt، lt و eq یک latch جداگانه در نظر می‌گیریم که خروجی این latchها همان خروجی‌های مدار می‌باشند. ساختار کلی D-latch در تصویر زیر آمده است و ما نیز با توجه به همین ساختار مدار را طراحی کرده‌ایم.



تصویر صفحه بعد طراحی ماژول را با استفاده از زبان Verilog نشان می‌دهد.

```
module SerialComparator (
    input wire clk,
    input wire reset,
    input wire a,
    input wire b,
    output wire lt, // less than
    output wire gt, // greater than
    output wire eq  // equal
);
    // Internal signals
    wire gt_in, lt_in, eq_in;

    //output inverted signals
    wire not_gt, not_eq, not_lt;

    assign lt_in = ((eq & ~a & b) | lt) & (~reset);
    assign gt_in = ((eq & a & ~b) | gt) & (~reset);
    assign eq_in = ((eq & (a == b)) & (~reset)) | (reset);

    assign lt = ~(not_lt & ~(clk & lt_in));
    assign not_lt = ~(lt & ~(clk & ~(lt_in)));

    assign gt = ~(not_gt & ~(clk & gt_in));
    assign not_gt = ~(gt & ~(clk & ~(gt_in)));

    assign eq = ~(not_eq & ~(clk & eq_in));
    assign not_eq = ~(eq & ~(clk & ~(eq_in)));

endmodule
```

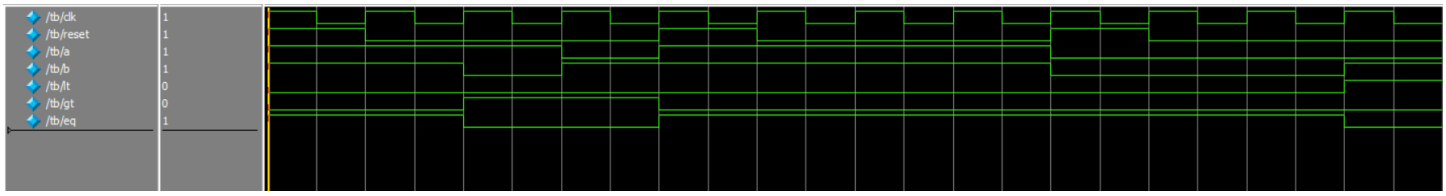
حال ماژول testbench را طراحی می‌کنیم. تصاویر زیر تست‌ها موجود در ماژول را نشان می‌دهند. فایل کلی ماژول نیز در مانند سایر ماژول‌ها در پیوست موجود است.

```
// Test1: a = 110 b = 101
reset = 1;
clk = 1;
a = 1;
b = 1;
#5
clk = 0;
#5
reset = 0;
clk = 1;
#5
clk = 0;
#5
a = 1;
b = 0;
clk = 1;
#5
clk = 0;
#5
a = 0;
b = 1;
clk = 1;
#5
clk = 0;
#5
```

```
// Test2: a = 111 b = 111
reset = 1;
clk = 1;
a = 1;
b = 1;
#5
clk = 0;
#5
reset = 0;
clk = 1;
#5
clk = 0;
#5
a = 1;
b = 1;
clk = 1;
#5
clk = 0;
#5
a = 1;
b = 1;
clk = 1;
#5
clk = 0;
#5
```

```
// Test3: a = 000 b = 001
reset = 1;
clk = 1;
a = 0;
b = 0;
#5
clk = 0;
#5
reset = 0;
clk = 1;
#5
clk = 0;
#5
a = 0;
b = 0;
clk = 1;
#5
clk = 0;
#5
a = 0;
b = 1;
clk = 1;
#5
clk = 0;
#5
```

تصویر زیر خروجی waveform را برای این مدار نشان می‌دهند.



خروجی flow summary مدار در تصویر زیر قابل مشاهده است.

Flow Summary	
Flow Status	Successful - Mon Jul 15 16:39:21 2024
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	serial_comparator
Top-level Entity Name	SerialComparator
Family	Cyclone IV GX
Total logic elements	7 / 14,400 (< 1 %)
Total combinational functions	7 / 14,400 (< 1 %)
Dedicated logic registers	0 / 14,400 (0 %)
Total registers	0
Total pins	7 / 81 (9 %)
Total virtual pins	0
Total memory bits	0 / 552,960 (0 %)
Embedded Multiplier 9-bit elements	0
Total GXB Receiver Channel PCS	0 / 2 (0 %)
Total GXB Receiver Channel PMA	0 / 2 (0 %)
Total GXB Transmitter Channel PCS	0 / 2 (0 %)
Total GXB Transmitter Channel PMA	0 / 2 (0 %)
Total PLLs	0 / 3 (0 %)
Device	EP4CGX15BF14C6
Timing Models	Final

خروجی rtl_viewer مدار در تصویر زیر قابل مشاهده است. فایل pdf آن نیز در پیوست آمده است.