تمرین اول طراحی سیستم‌های دیجیتال

دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف

آرین احدی نیا

شماره دانشجویی: ۹۸۱۰۳۸۷۸

استاد درس: جناب آقای دکتر بیات سرمدی

پاییز ۱۴۰۰

فهرست عناوین

سوال ۱

با توجه به طول متغیرها، مقادیر باینری آنها برابر خواهد بود با

i	0000 0000 0000 0000 0100 1010 0110 1100
a	1010
b	1011
c	0101
d	1111
e	00 1101

i[3+:6] برابر i[8:3] خواهد بود.

حال به تشریح و محاسبه هر یک از مقادیر می‌پردازیم.

اپراتورهای خط اول، اپراتورهای دوتایی بیتی (bitwise) هستند. به این صورت که & برابر bitwise AND است و بیت‌های دو بردار ورودی را نظیر به نظیر AND کرده و در خروجی قرار می‌دهد. همچنین اپراتور ^~ در اینجا به عنوان یک اپراتور دوتایی بیتی استفاده شده است به این صورت که بیت‌های دو بردار ورودی را XNOR کرده و در خروجی قرار می‌دهد. با این استدلال، حاصل a&c برابر 0000 و حاصل b~^d برابر 1011 خواهد بود. توجه کنید که با اینکه ترتیب اندیس‌گذاری در a و c و همچنین b و d متفاوت است، اما اندیس تاثیری در عملکرد این دو اپراتور ندارد و اولین بیت کم‌ارزش با اولین بیت کم‌ارزش، دومین بیت کم‌ارزش با دومین بیت کم‌ارزش و ... جفت میشوند و اپراتور بر آنها اعمال می‌گردد.

اپراتور {} مربوط به concatenation است به این صورت که بیت‌های بردارهای ورودی را با حفظ ترتیب به همدیگر می‌چسباند. در نتیجه حاصل {2'b10, b} برابر 101011 خواهد شد. اپراتور | برای bitwise OR است به این صورت که بیت‌های دو بردار ورودی را نظیر به نظیر OR کرده و در خروجی قرار می‌دهد. بنابرین {2'b10, b}|e برابر 101011|001101 است که برابر 101111 می‌شود.

در نهایت توجه کنید که اپراتور && یک اپراتور دوتایی منطقی است. به این صورت که هر یک از ورودی‌ها را به عنوان یک مقدار منطقی در نظر می‌گیرد و آنها را با هم AND می‌کند. توجه کنید که هر مقدار غیر از صفر به عنوان True یا تک بیت ۱ ارزیابی خواهد شد بنابرین a&&b برابر 1&&1 و در نهایت برابر 1 خواهد بود.

بنابرین خروجی این کد برابر خواهد بود با

```
0000 1011
101111 1
```

سوال ۲

خط ۱: چون N به صورت پارامتر است، تعریف اندازه ورودی b برحسب N حتما باید در خطوط بعدی انجام شود. بنابرین خط زیر به زیر خط ۴ اضافه می‌شود.

```
input [N-1:0] b;
```

و خط ۱ به شکل زیر تغییر می‌کند.

```
module q2 (q, a, b, lda, ldb, clk);
```

خط ۴: پس از تمامی دستورات Verilog، باید ";" بیاید. در انتهای این دستور ";" نیامده است.

خط ۱۰: درگیت‌های primitive، ابتدا خروجی‌ها و سپس ورودی‌ها به instance پاس داده میشوند. بنابرین این خط باید به شکل زیر تغییر کند.

```
and (out_two, b[0], lda);
```

تا به اینجا، کد تبدیل به کد کامپایل‌پذیر و شبیه‌سازی‌پذیر شده است. اما طبق فرمایش دستیاران محترم آموزشی، خروجی ـ که q است — باید مقدار داشته باشد. حقیقتا به دست آوردن ضابطه خروجی از روی صورت سوال غیر ممکن است بنابرین با توجه به یک فرض پیش می‌رویم.

فرض: فرض می‌کنیم که دو بیت خروجی برابر مقدار هر یک از سیم‌ها می‌شود.

در این صورت اصولا نیازی به تعریف wire نبود و میتوانستیم مستقیما از q[0] و q[N-1] به جای آنها استفاده کنیم. در این صورت دو خط مربوط به تعریف wireها حذف و گیت‌ها به صورت زیر در می‌آمدند.

```
xor (q[0], a[N-1], ldb);
and (q[N-1], b[0], lda);
```

اما راهکار دیگر این است که از buffer برای وصل کردن wireهای تعریف شده به خروجی استفاده کنیم. در این صورت دو خط به صورت زیر به انتهای کد اضافه می‌شد.

```
buf (q[0], out_one);
buf (q[N-1], out_two);
```

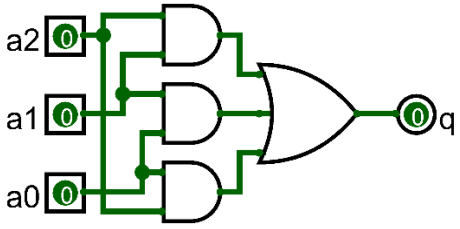
سوال ۳

(الف) با استفاده از جدول کارنو و ساده سازی POS، ضابطه خروجی مدار بر حسب ورودی‌ها به شکل زیر خواهد بود.

a_0 \ a_2a_1	00	01	11	10
0	**0**	**0**	**1**	**0**
1	**0**	**1**	**1**	**1**

$$q = a_0a_1 + a_1a_2 + a_2a_0$$

در نتیجه شماتیک مدار به شکل زیر در خواهد آمد.



(ب) ماژول مورد نظر را می‌توانیم به شکل زیر در زبان Verilog در سطح گیت تولید کنیم.

```
module Hamming_weight (a, q);
    output q;
    input [2:0]a;

    wire a1, a2, a3;

    and (a1, a[0], a[1]);
    and (a2, a[1], a[2]);
    and (a3, a[2], a[0]);

    or (q, a1, a2, a3);
endmodule
```

سوال ۴

(الف) ساده‌سازی اولیه را بر مبنای جدول کارنو و ساده‌سازی POS انجام می‌دهیم. سپس با انجام عملیات‌های جبر منطقی تلاش برای ساده‌سازی هر چه تمام‌تر و حذف گیت‌های NOT می‌کنیم. در نهایت توجه فرمایید که با توجه به کامل بودن منطقی NAND و یا NOR گیت NOT را می‌توانیم به صورت‌های

$$a' = NAND(a, a)$$

$$a' = NOR(a, a)$$

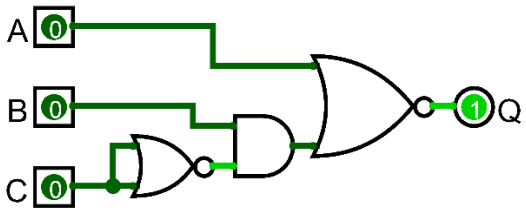
بسازیم.

A \ CB	00	01	11	10
0	**1**	**0**	**1**	**1**
1	**0**	**0**	**0**	**0**

$$q = A'(B' + C)$$

$$q = A'(B' + C) = \Big(\big(A'(B' + C)\big)'\Big)' = (A + BC')' = NOR(A, BC') = NOR(A, NOR(C, C)B)$$

بنابرین شماتیک مدار به صورت زیر در خواهد آمد.



(ب) ماژول مورد نظر را می‌توانیم به شکل زیر در زبان Verilog در سطح گیت تولید کنیم.

```
module truth_table (A, B, C, Q);
    output Q;
    input A, B, C;

    wire C_not, and_res;

    nor (C_not, C, C);
    and (and_res, C_not, B);
    nor (Q, A, and_res);
endmodule
```