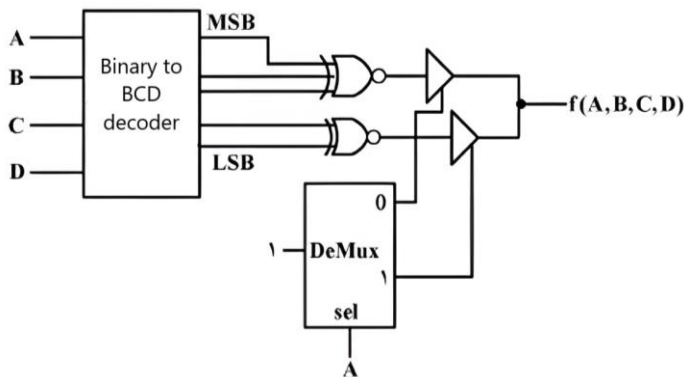




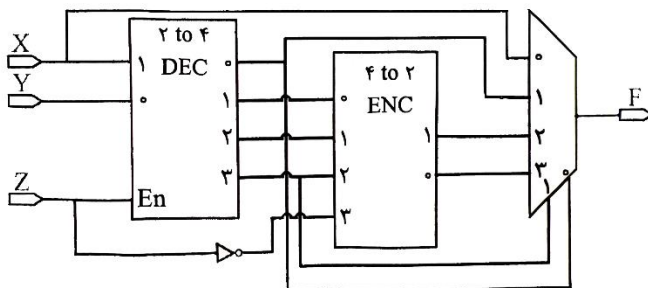
به موارد زیر توجه کنید:

- ۱- حتما نام و شماره دانشجویی خود را روی پاسخنامه بنویسید.
- ۲- در حل سوالات به نوشتن جواب آخر اکتفا نکنید. همه مراحل میانی را هم بنویسید.
- ۳- کل پاسخ تمرینات را در قالب یک فایل pdf با شماره دانشجویی خود نام گذاری کرده در سامانه CW بارگذاری کنید.
- ۴- در صورت مشاهده هر گونه مشابهت نامتعارف هر دو (یا چند) نفر کل نمره این تمرین را از دست خواهند داد.

۱- (۲ نمره) مشخص کنید مدار زیر بر حسب مینترمها چه تابعی را پیاده سازی می کند. توجه کنید در تبدیل کننده باینری به BCD، ورودی A با ارزش ترین و ورودی D کم ارزش ترین هستند. همچنین خروجی MSB با ارزش ترین و خروجی LSB کم ارزش ترین هستند.



۲- (۲ نمره) خروجی تابع F را به دست آورید.



۳- (۲ نمره) تابع  $F(a, b, c) = \bar{a}\bar{b} + \bar{a}c$  را با هر کدام از موارد زیر پیاده سازی کنید. در صورت نیاز می توانید از گیت های پایه استفاده کنید.

الف- Mux  $8 \times 1$

ب- Mux  $4 \times 1$  با خطوط انتخاب ab

ج- Mux  $4 \times 1$  با خطوط انتخاب bc

د- Mux  $2 \times 1$  با خط انتخاب c

۴- (۲ نمره) بررسی کنید که هر یک از واحدهای زیر یک منطق کامل هست یا خیر.

الف- نیم تفریق کننده

ب- مقایسه کننده دوبیتی

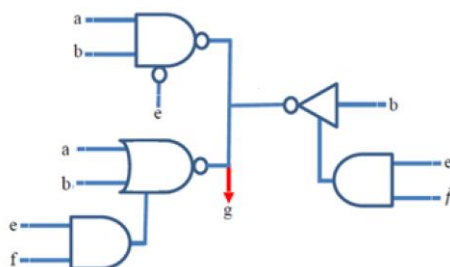
۵- (۲ نمره) با حداقل تعداد مالتی پلکسر  $2 \times 1$ ، یک مالتی پلکسر  $8 \times 1$  بسازید. سپس بررسی کنید برای ساخت یک مالتی پلکسر  $64 \times 1$  به چه تعداد مالتی پلکسر  $2 \times 1$  نیاز است.

۶- (۲ نمره) به کمک یک دیکودر ۳ به ۸ با خروجی های active-low و کمترین گیت اضافه با کمترین تعداد ورودی مداری بسازید که یک عدد ۳ بیتی دریافت کند و حاصل جمع مقسوم علیه های آن را تولید کند.

۷- (۲ نمره) تابع زیر را به کمک یک مالتی پلکسر  $4 \times 1$  و حداقل گیت ممکن پیاده سازی کنید. (هر یک از گیت های AND، OR، NAND، NOR، XOR و NOT را بدون توجه به تعداد ورودی یک گیت محسوب کنید). راهنمایی: متغیرهای ارزش بالا را به ورودی های انتخاب مالتی پلکسر بدهید.

$$f(w_1, w_2, w_3, w_4, w_5) = \overline{w_1} \overline{w_2} \overline{w_4} \overline{w_5} + w_1 w_2 + w_1 w_3 + w_1 w_4 + w_2 w_3 w_5$$

۸- (۳ نمره) تابع g مدار زیر را به شکل ساده شده SOP بنویسید.



۱- (۳ نمره) با استفاده از تعداد کافی مالتی پلکسر  $2 \times 1$  یک جمع کننده کامل (full adder) بسازید. فرض کنید ورودی های صفر و یک در دسترس هستند، اما هیچ گیت منطقی ندارید.