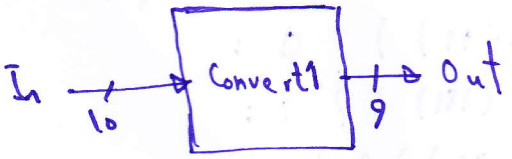
حل سؤال ۵ - (این تمرین کاملاً در سطح تمرین‌های امتحان است)

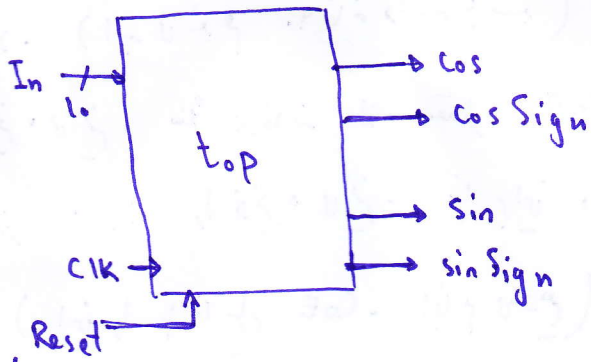
قبل از هر چیز دقت می‌کنیم که عدد ورودی هر چه باشد، مثلاً ما لازم است مقدار Sin و Cos را برای عددی بین ۰ تا ۳۶۰ درجه حساب کنیم.

بنابراین یک ماژول می‌نویسیم که ورودی آن عدد ۱۰ بیتی و خروجی آن یک عدد بین ۰ تا ۳۶۰ درجه است.



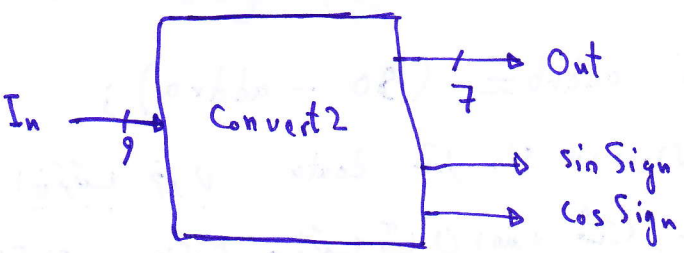

```
assign Out = ( (In >= 0) && (In < 360) ) ? In :
             ( (In > 359) && (In < 720) ) ? (In - 360) :
             (In - 720) ;
```

خوب! حالا دقت می‌کنیم که برای محاسبه Sin و Cos لازم نیست کل ورود ۰ تا 360 درجه را پوشش دهیم و اگر فقط ۰ تا ۹۰ درجه را پوشش دهیم کافیست. برای همین Range، فقط علامت Sin و Cos عوض خواهد شد. پس ماژول ما (top level) این شکلی است:



این که عوض Cos و Sin چقدر باشد بری گردد به خانه‌ای که داخل ماژول استفاده می‌کنیم برای Look up کردن مقدار Sin و Cos.

پس یک ماژول می‌خواهیم که از روی عدد ورودی که مقداری بین ۰ تا 360 دارد عدد خروجی را تولید کند که مقدار بین صفر تا ۹۰ دارد. همین ماژول باید سیگنال‌های SinSign و CosSign را هم تولید کند.



```
assign Out = ( (In >= 0) && ( In < 91) ) ? In :
             ( (In > 90) && ( In <= 180) ) ? (180 - In) :
             ( (In > 180) && ( In < 271) ) ? (270 - In) :
                                             (360 - In) ;

// sinSign will be 1 whenever the output value should be negative
assign sinSign = ( (In >= 0) && ( In < 91) ) ? 0 :
                 ( (In > 90) && ( In < 181) ) ? 0 :
                 ( (In > 180) && ( In < 271) ) ? 1 : 1 ;
// Synthesizer will optimize the above statement into two conditions
assign cosSign = ( (In >= 270) && (In < 91) ) ? 0 : 1 ;
```

خوب! این مدول هم تمام شد. حالا وارد اصلی را می‌سازیم که ترکیبات خروجی آن مقدار sin و cos را شد.

از آنجایی که نوع ورودی مقادیر sin و cos را در خروجی تعریف کنیم از یک حافظه dual port

در حال ROM استفاده می‌کنیم. از آنجایی که $\sin\alpha = \cos\left(\frac{\pi}{2} - \alpha\right)$ یا در داخل

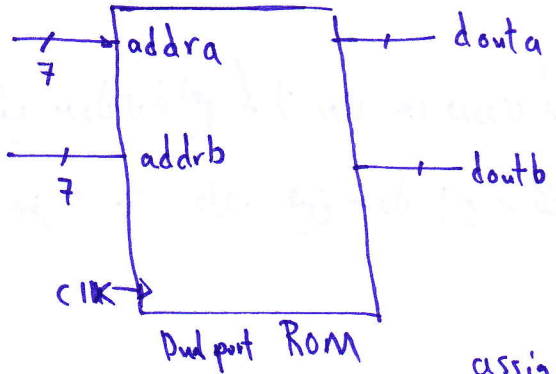
حافظه Dual port فقط مقادیر sin را ذخیره می‌کنیم. حافظه dualport یک که 91 خانه

دارد. (البته می‌شود 90 خانه داشته باشد) و در خانه‌ها به ترتیب مقدار sin را به ازای آن درجه

ذخیره می‌کنیم. مثلاً در خانه 0 مقدار $\sin 0$ را و در خانه 1 مقدار $\sin 1$ را و در خانه 90

مقدار $\sin 90$ را ذخیره می‌کنیم. بنابراین ورودی آدرس حافظه همان مقدار ورودی است.

(بعضی فرق بیشتر بهتر - دقت)



(این را با فایل CoE. انجام دادیم)

خروجی out واحد محاسبه را به addra وصل می‌کنیم. برای addrb داریم:

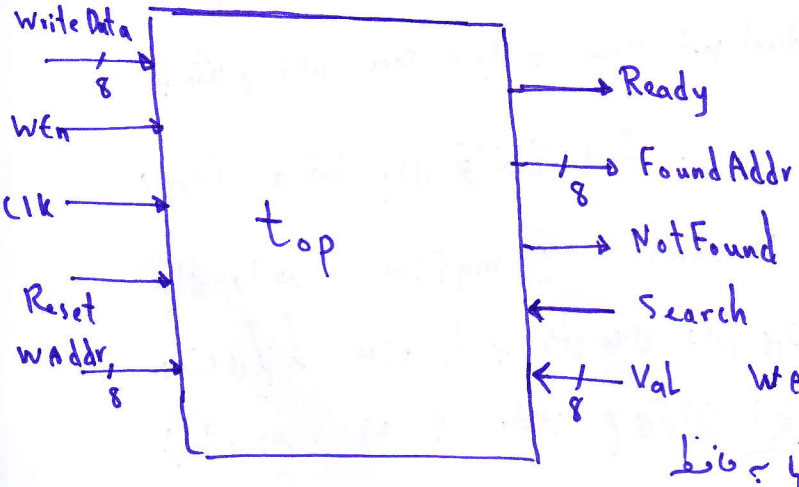
```
assign addrb = (90 - addra);
```

به این ترتیب خروجی douta مقدار sin عدد ورودی - خروجی addrb مقدار cos عدد ورودی و

sinSign و cosSign به ترتیب نشان دهنده مثبت و یا منفی بودن مقادیر sin و cos خواهند بود.

حل تمرین ۲ : (این تمرین کاملاً در سطح تمرین های امتحان است)

خوب ابتدا باید مشخص کنیم که به چه پورت هایی برای این که این واحد به درستی کار کند نیاز داریم.



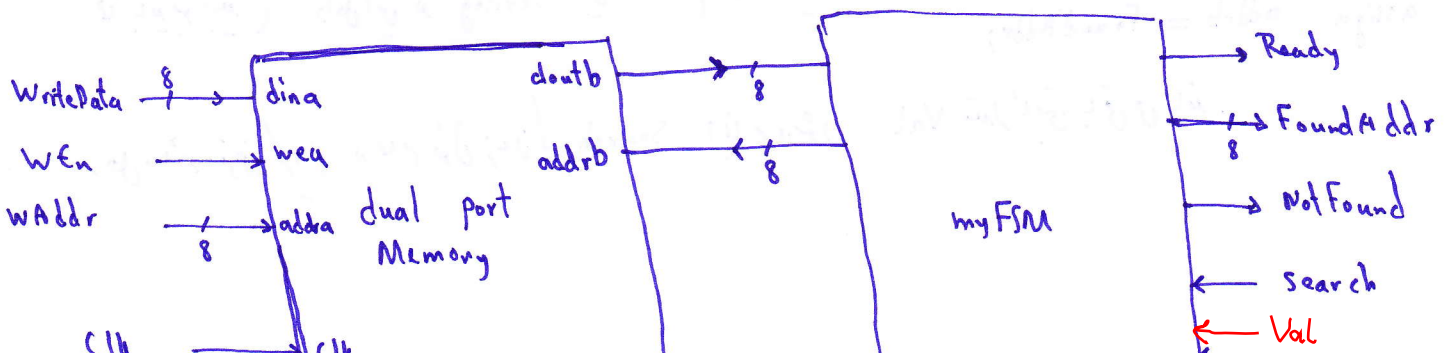
شما عملاً در پروژه ها خودتان باید تعیین کنید که ورودی شما چه ورودی هایی باید داشته باشد اما در امتحان لازم نیست.

سیگنال های WriteData ، WEn ، WAddr سیگنال هایی هستند که داده هایش را با آنها به حافظه می نویسید. فعلاً فرض کنید که حافظه دارای ۲۵۶ خانه که هرکدام به عرض ۸ بیت است.

- Ready — این سیگنال را وقتی که کل Search کامل انجام شده ۱ می کنیم. این سیگنال هنگامیکه داریم حافظه را جستجو می کنیم باید صفر باشد.
- FoundAddr — اگر جستجو به دنبال Val در حافظه با موفقیت انجام شده روی خروجی FoundAddr آدرس خانه ای که Val را در آن پیدا کرده ایم باز می گردانیم. (و Ready را ۱ می کنیم)
- NotFound — اگر تا به حافظه را گشتیم ولی Val را پیدا نکردیم تنها این سیگنال را فعال یا Ready ۱ می کنیم.
- Search — دستور آغاز جستجو — هنگامیکه می خواهیم برای یک مقداری داخل حافظه جستجو کنیم آن مقدار را روی Val قرار می دهیم و برای ۱ سیکل ساعت Search را فعال می کنیم.

حال با توجه به این ورودی / خروجی ها ، طراحی را انجام می دهیم:

ابتدا بلوک دیاگرام کل طرح را رسم می کنیم. سپس به طراحی تک تک اجزا می پردازیم.

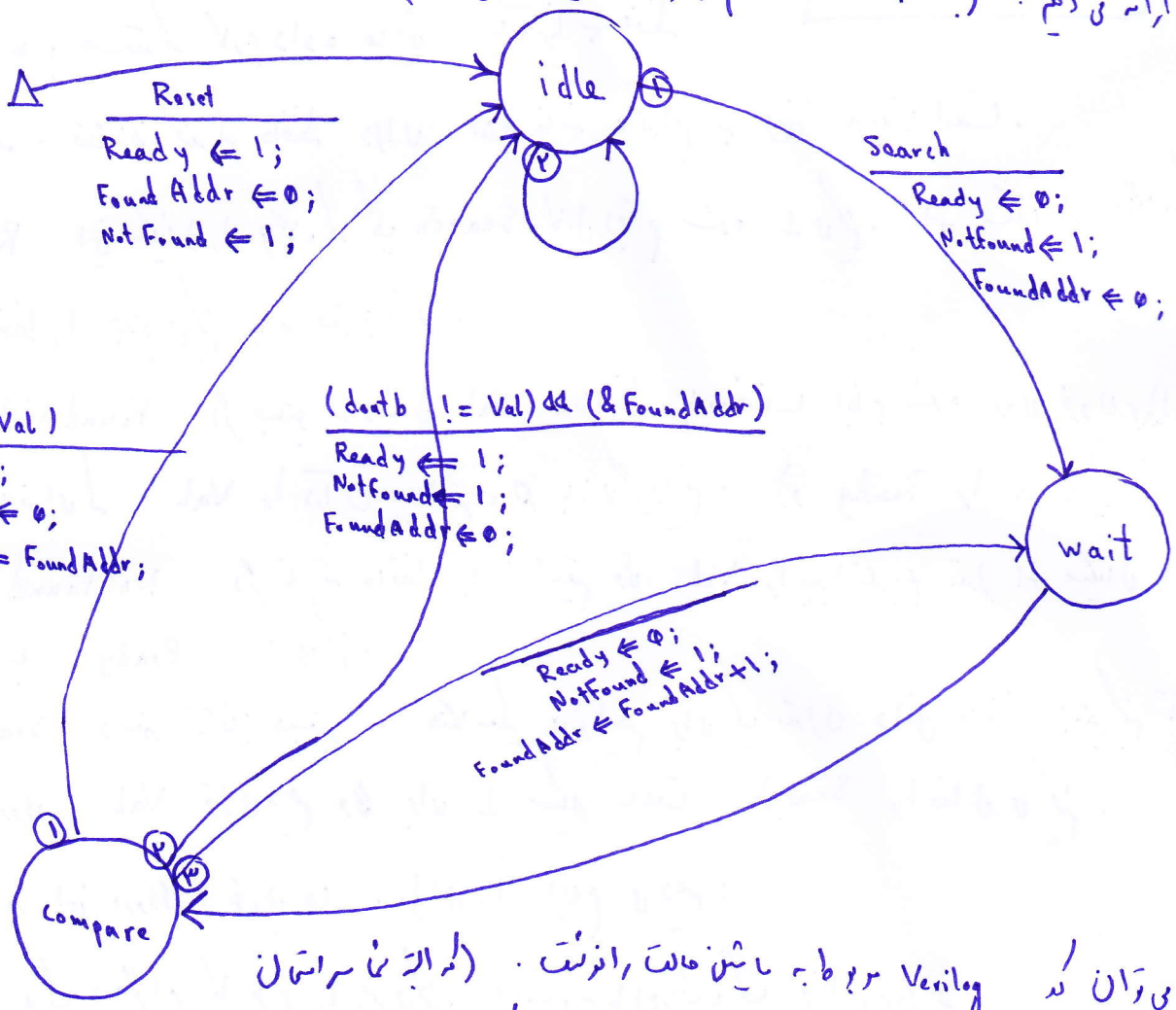


بلوک دیاگرام top level module را رسم کردیم. همانطور که می‌بینید تنها کار ما طراحی واحد myFSM است.

به هنگام تولید core مربوط به dual port mem، تنظیم کرده‌ایم که پورت b فقط برای نوشتن و پورت a فقط برای خواندن باشد.

برای واحد myFSM یک State Machine می‌کشیم.

دقت کنید که FSM را به روش‌های مختلفی می‌توان کشید. ما تنها یک روش را (با ۳ state) ارائه می‌دهیم. (با ۲ state هم می‌توان این را حل نمود)



حال می‌توان کد Verilog مربوط به ماشین حالت را نوشت. (البته تنها سرانگشتان قدم بایستی بنویسید) داخل این کد Verilog این خط را هم اضافه کنید:

```
assign  addrb = FoundAddr;
```

در حل ما فرض کردیم در تمام طول زمانی که Search انجام می‌شود Val مقدار ثابتی باقی می‌ماند.