به نام خدا

آموزش VHDL

(قسمت اول)

Very High Speed Integrated Circuit Hardware Description Language
(Part 1)

نحوه تعریف ورودی و خروجی

با یاد و نام خدا اولین درس از VHDL را آغاز می کنیم قبل از شروع درس توجه به نکات زیر الزامی است:

شما چه هدفی را از آموختن این زبان دنبال می کنید؟ آیا تفریحی آن را یاد میگیرید؟ یا نه شاید اهداف مهمتری را دنبال می کنید؟

به نظر من این زبان ساده ترین زبان برای برنامه نویسی سخت افزاری می باشد.با وجود تکنولوژی جدید FPGA شما بیشتر به این نوع زبان احتیاج خواهید داشت . FPGA چیست؟ تراشه هایی هستند شبیه به IC با پایه هایی که چهار طرف آن را گرفته است.شما می توانید برنامه های پیچیده ای را روی FPGA قرار داده و حد اکثر استفاده را با هزینه ای کم و صرف وقت کم بدست آورید.ابتدا باید زبان VHDL را خوب بیاموزیم تا بتوانیم وارد وادی FPGA شویم تا به نحو احسن از آن سود ببریم.(ناگفته نماند که زبانهای دیگری هم برای Program کردن تراشه های FPGA وجود دارد که من VHDL را بهتر می پسندم!)

در آینده ای نزدیک قصد دارم مقاله ای توصیفی در مورد تراشه های قابل برنامه ریزی ارائه دهم.

به امید خدا.

توجه:این بخش آموزشي بر اساس کامپایلر و محیط نرم افزار Active-VHDL ارائه شده است.

Entity-۱

اکنون وقت آن رسیده که با اولین دستور آشنا شوید : **Entity**

همیشه هر سیستمی در طراحی VHDL با Entity آغاز خواهد شد.

مرکز اصلی برنامه شما یا به عبارت دیگر همان Main شما بین دو عبارت Entity و end Entity; تعریف می شود.نام سیستمی که شما قصد طراحی آن را دارید مقابل کلمه ی Entity قرار می گیرد مثلا اگر می خواهید یک Processor طراحی کنید باید بنویسید:

```
Entity Processor is
--این قسمت محل تعریف ورودی ها و خروجی های برنامه شماست
End entity;
```

حالا باید ببینیم منظور از ورودی و خروجی که بعد از Entity تعریف می شود چیست؟

۱-پارامتر ها : مانند پهنای یک گذر گاه یا Bus برای یک پردازنده یا ماکزمیم مقدار فرکانس برای clock سیستم.

۲-راه های ارتباطی از خارج به سیستم و از داخل سیستم به خارج که وظیفه انتقال داده ها را بر عهده دارند.

با مطا لعه ی مثال زیر که یک ثبات ۸ بیتی را توصیف می کند دو مطلب بالا را بهتر درک خواهید کرد:


```
Entity eight_bit_register is
پارامترها:
طول = ۸
ماکسیمم فرکانس=۵۰ مگاهرتز
راه های ارتباطی با ثبات:
D_in ۸بیت ورودی
D_out ۸بیت خروجی
CLK یک بیت ورودی
End entity eight_bit_register;
```

نمونه دیگر :

توجه:پارامترها و را هها‌ی ارتباطی با ثبات گفته شده در این مثال فقط جنبه ی سمبولیک دارند و صرفا برای درک بیشتر آورده شده اند و نمونه کد VHDL نیستند.

```
Entity eight_bit_register is
Generic ( length=8
          Fmax =50 MHz
        );
Port     ( D_IN    eight-bit input
           D_OUT eight-bit output
           CLK      one-bit input
         );
End entity eight_bit_register;
```

پارامتر ها در زبان VHDL بعد از کلمه ی Generic می آیند و ورودی خروجی ها بعد از کلمه ی Port خواهند آمد که در ادامه بیشتر در مورد این دو قسمت بحث می کنیم قبلا کمی در مورد Architecture صحبت می کنیم:

Architecture-۲

مثال:وسایل پیشرفته الکترونیکی مانند تلویزیون گاهی کمتر استفاده میشوند زیرا اگر وسیله ای برای کنترل کردن و ارتباط برقرار کردن با آنها در اختیار نداشته باشیم.این وسیله قسمت مهمی در یک طراحی VHDL می باشد مثلا در TV همان Remote Control را فرض می کنیم Architecture هم دقیقا کار آن را انجام می دهد که به صورت زیر تعریف می شود:

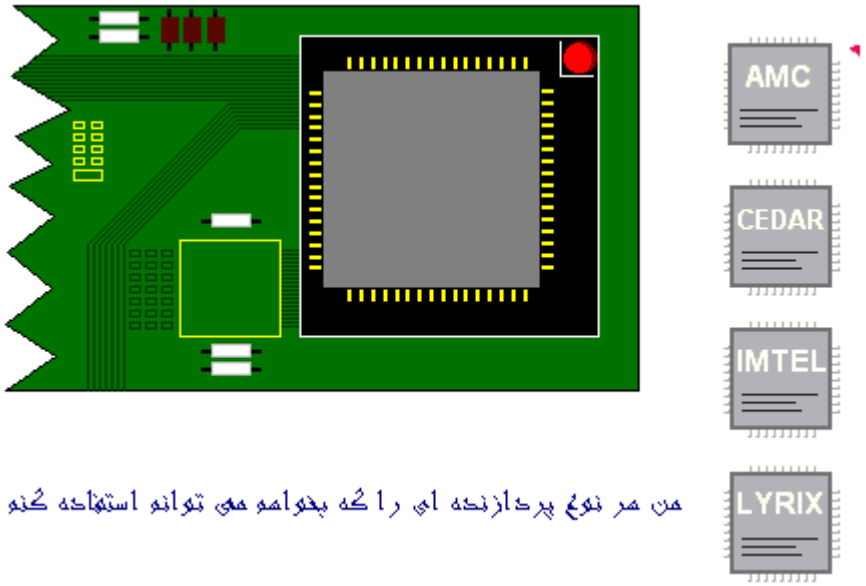
```
Entity Tvset is

...

End entity Tvset;

Architecture TV2000 of Tvset is

...

End Architecture TV2000;
```

دستوراتی که ما می توانیم در قسمت Architecture قرار دهیم ۲ نوع هستند:

1-**Functionality (Behavioral)** رفتاری
2-**Structural** متنی و ساختمانی

منظور از رفتاری این است که دستورات ما فقط رفتار سیستم را توصیف کنند و به انواع گیتهای به کار گرفته شده و ریزه کاري ها در سیستم کاری نداریم. سنتز کردن اینگونه برنامه ها توسط نرم افزارهای سنتز کننده دشوار خواهد بود ولی درک و فهم برنامه ای که ما به این روش نوشته ایم بسیار آسان است.
منظور از ساختمانی یعنی دستورات ما از نظر مدار های منطقی کامل باشند و تمام گیتهای موجود در سیستم را توصیف می کنیم که به این یک برنامه نویسی کامل گفته می شود.سنتز کردن اینگونه برنامه ها آسان می باشد ولی درک انها توسط ما کاری دشوار خواهد بود.

هر سیستمی که ما قصد طراحی آن را داریم فقط یک Entity دارد با چند Architecture .
عکس این مطلب درست نیست یعنی برای یک Architecture نمی توانیم تعدادی Entity داشته باشیم.
به مثال زیر توجه کنید:
اگر مینبوردی(MainBoard) که با پردازنده های مختلف کار میکند را به عنوان یک برنامه VHDL در نظر بگیرید خواهید دید که مینبورد ما همان قسمت Entity برنامه است چون فقط یک مینبورد داریم پس فقط یک Entity هم داریم و پردازنده های مختلف همان Architecture های برنامه ما می باشند مثلا برای CPU های AMD,Intel,Cyrix خوهیم داشت:




```
Entity pentium is
...
End entity pentium;
Architecture Amd of pentium is
..
End Amd;
Architecture Intel of pentium is
..
End Intel;
Architecture Cyrix of pentium is
..
End Cyrix;
```

با داشتن یک چنین برنامه ای هر زمان به هر نوع پردازنده ای که احتیاج داشتیم می توانیم آن را روی مینبورد وصل کنیم.
همه دستوراتی که می توان در Architecture استفاده کرد در کتابخانه ای به نام IEEE وجود دارد با مراجعه به آن مي توان ابهامات احتمالی را رفع کرد.اگر به لفظی غیر استاندارد و نا مفهوم برخوردید ابتدا Library را صدا زده سپس در کتابهای موجود در آن به دنبال کلمه مورد نظر بگردید.بسته بندی ها یا Packages که در Library وجود دارند به سه دسته تقسیم می شوند:

۱-Standard
این کتابخانه به صورت Default در نظر گرفته می شود و اگر در ابتدای برنامه نام کتابخانه مورد نظر ذکر نشد کامپایلر به طور پیش فرض آن را Standard در نظر می گیرد که شامل تمام انواع استاندارد عملگرها و اشیا می باشد.

۲-Textio
این Library فقط به منظور شبیه سازی و مدل سازی با VHDL به کار می رود و به عنوان برنامه ای که بتواند روی مدارات مجتمع پیاده سازی شود نیست و بدین صورت در ابتدای برنامه نوشته می شود:

```
Library Std;
Use Std.TextIO.all;
```

۳-STD_LOGIC_1164
این مهمترین و پرکاربرد ترین کتابخانه است و همیشه به عنوان مرجع استفاده می شود نمایش آن بدین صورت است:

```
Library IEEE;
Use IEEE.Std_Logic_1164.all;
```

اگر دقیقا پس از خواندن مطالب گفته شده در بالا ، درس بعدی را بخوانید خیلی بهتر است...

درس بعد راجع به طرز تعریف سیگنالها و پورتها خواهد بود.

پایان قسمت اول

نگارنده : فرشید سفیدگران
کارشناسی کامپیوتر سخت افزار
خرداد ۱۳۸۲
Sefidgaran@gmail.com
http://Sefidgaran.blogfa.com