

به نام خداوند بخشنده مهربان

امیرحسین کامرانی - ۹۵۳۱۰۷۰

اشکان گهرفر - ۹۵۳۱۴۲۷

گزارش فاز ۱_ پروژه طراحی سیستم‌های دیجیتال برنامه‌پذیر

ماژول ۱:

ساختار داده‌ی ورودی و خروجی

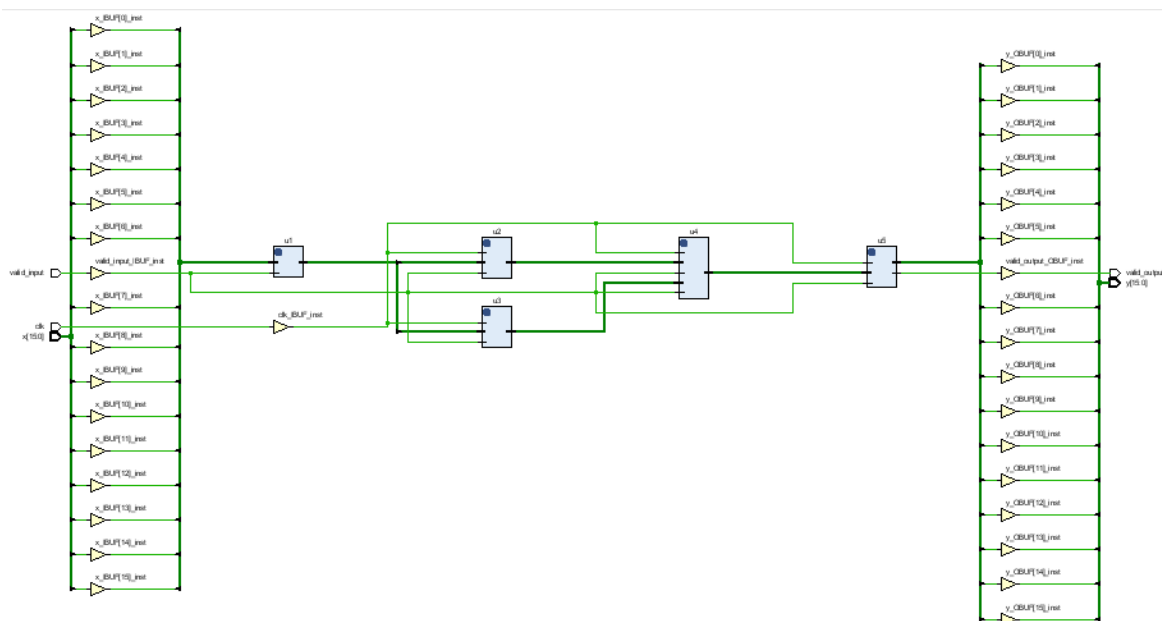
نوع داده ۱۶ بیتی ممیز ثابت است. ورودی مدار در هسته‌ی پردازشی CORDIC مورد استفاده قرار گرفته شده است. بنابراین ساختار ورودی ۱۳_۱۶ fix است. یعنی ۳ بیت برای بخش صحیح و ۱۳ بیت برای بخش اعشاری در نظر گرفته شده است.






نحوه پیاده سازی

با توجه به اینکه خوده هسته CORDIC قادر به انجام محاسبه \tanh نمی باشد با تقسیم $\sin H$ بر $\cos H$ به دست آمده از هسته کوردیک و با استفاده از تقسیم موجود در هسته Floating Point مقدار تانژانت هیپربولیک را به دست آوردیم.

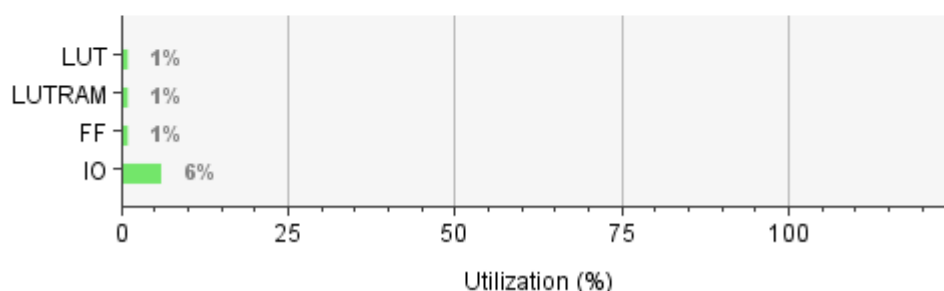
لازم به ذکر می‌باشد که برای استفاده از هسته Floating Point باید داده هایی که به صورت ممیز ثابت بودند را با استفاده از خوده هسته Floating Point به ممیز شناور تبدیل می‌کردیم و سپس بعد از انجام تقسیم با استفاده از همان هسته دوباره از ممیز شناور به ممیز ثابت تبدیل میکردیم.

شماتیک مدار



Name	Slice LUTs (303600)	Slice Registers (607200)	F7 Muxes (151800)	F8 Muxes (75900)	Bonded IOB (600)
▼ N tangent_h	2163	1813	10	5	35
>  u1 (cordic_1)	1143	0	0	0	0
>  u2 (fix2float)	63	139	4	2	0
>  u3 (fix2float)	63	139	4	2	0
>  u4 (floating_point_0)	795	1386	0	0	0
>  u5 (float2fix)	99	149	2	1	0

Resource	Utilization	Available	Utilization %
LUT	2163	303600	0.71
LUTRAM	48	130800	0.04
FF	1813	607200	0.30
IO	35	600	5.83



مقایسه مقادیر حاصل با مقادیر Golden

قادر خروجی حاصل از ورودی هایی که نزدیک به صفر هستند به صورت دقیق درست درمیاد اما برای مقادیر بزرگتر مثل ۲,۱۲۵ یا ۱,۵- مقدار خروجی حدود ۰,۱ از مقدار golden کمتر است.

Input	Output	Golden
0	-0.0001220703125	0
0.5	0.4620361328125	0.46211715726
0.75	0.6351318359375	0.635148952387
1	0.7615966796875	0.761594155956
2.125	0.806884765625	0.971872745914
- 1.5	-0.8070068359375	- 0.905148253645

ماژول ۲:

برای پیاده سازی این ماژول با توجه به اینکه ماژول تانژانت هیپربولیک را ساخته بودیم میتوانیم از کامپوننت آن استفاده کرده و با ایجاد یک آرایه دو بعدی از طریق تعریف `type` در پکیج ساخته شده به عنوان ورودی و خروجی استفاده کنیم و برای `port map` کردن به آن نیز میتوان از دو حلقه `for` ساخته شده توسط `generic` به صورت تو در تو استفاده کرد.

ماژول ۳:

ساختار داده‌ی ورودی و خروجی

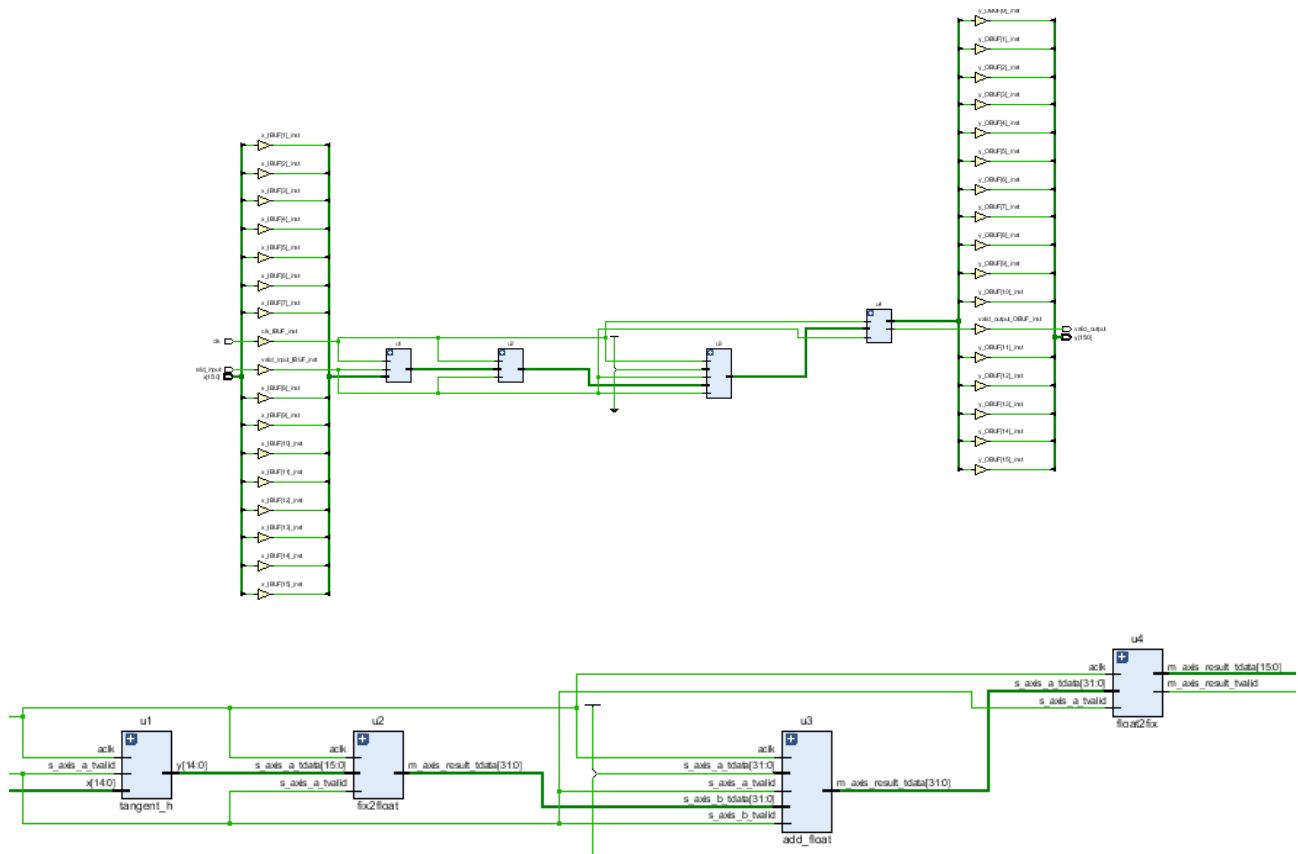
نوع داده ۱۶ بیتی ممیز ثابت است ورودی و خروجی `fix_۱۶_۱۳` است. یعنی ۳ بیت برای بخش صحیح و ۱۳ بیت برای بخش اعشاری در نظر گرفته شده است.

نحوه پیاده سازی

با توجه به اینکه در ماژول ۱ توانستیم مقدار `tanh` را محاسبه کنیم اکنون میتوانیم با استفاده از فرمول ذیل برای محاسبه `sigmoid` آن را با استفاده از شیفت به راست ریاضی و استفاده از هسته `Floating Point` برای جمع کردن محاسبه کنیم:

$$sigmoid(x) = \frac{1}{2} + \frac{1}{2} \times \tanh\left(\frac{x}{2}\right)$$

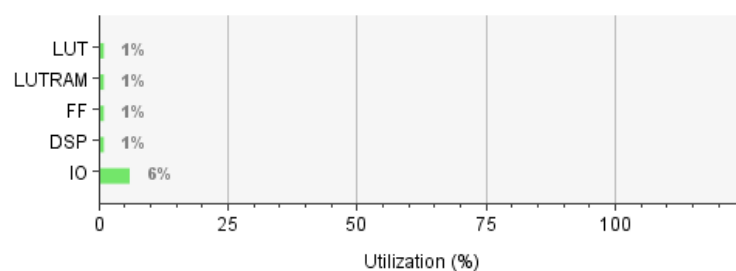
برای پیاده سازی این فرمول قبلا ماژول محاسبه تانژانت هیپربولیک را به دست آورده بودیم و فقط کافیست مقدار ورودی آن را با حفظ علامت یک شیفت به راست ریاضی بدهیم تا تقسیم بر دو شود و سپس جواب حاصل از کامپوننت تانژانت هیپربولیک را نیز یک بار دیگر با حفظ علامت یک بیت شیفت به راست ریاضیاتی بدهیم و سپس باید خروجی حاصل از این عملیات ها را با عدد ۰,۵ جمع کنیم و با توجه به اینکه برای نوع داده حاصله که ممیز ثابت می باشد هیچ عملگر جمع کننده ای با استفاده از هسته پردازشی نداشتیم پس باید آن را با استفاده از هسته `Floating Point` به نوع داده ممیز شناور با دقت `single` تبدیل میکردیم و سپس با استفاده از همین هسته به اضافه عدد ۰,۵ میکردیم و در نهایت داده حاصله را تبدیل به ممیز ثابت ۱۶ بیتی می کردیم.



گزارش منابع استفاده شده

Name	Slice LUTs (303600)	Slice Registers (607200)	F7 Muxes (151800)	F8 Muxes (75900)	DSP s (2800)	Bonded IOB (600)
▼ N sigmoid	2520	2416	16	8	2	34
> u1 (tangent_h)	2163	1813	10	5	0	0
> u2 (fix2float)	63	139	4	2	0	0
> u3 (add_float)	195	315	0	0	2	0
> u4 (float2fix)	99	149	2	1	0	0

Resource	Utilization	Available	Utilization %
LUT	2520	303600	0.83
LUTRAM	69	130800	0.05
FF	2416	607200	0.40
DSP	2	2800	0.07
IO	34	600	5.67



مقایسه مقادیر حاصل با مقادیر Golden

قادر خروجی حاصل از ورودی هایی که نزدیک به صفر هستند به صورت دقیق درست درمیاد اما برای مقادیر بزرگتر مثل ۲,۱۲۵ یا ۱,۵- مقدار خروجی حدود ۰,۲ از مقدار golden تفاوت دارد.

Input	Output	Golden
0	0.5	0.5
0.5	0.561279296875	0.622459331
0.75	0.589599609375	0.679178699
1	0.615478515625	0.731058579
2.125	0.6966552734375	0.893309406
-1.5	0.3411865234375	0.182425524

ماژول ۴:

برای پیاده سازی این ماژول با توجه به اینکه ماژول sigmoid را ساخته بودیم میتوانیم از کامپوننت آن استفاده کرده و با ایجاد یک آرایه دو بعدی از طریق تعریف type در پکیج ساخته شده به عنوان ورودی و خروجی استفاده کنیم و برای port map کردن به آن نیز میتوان از دو حلقه for ساخته شده توسط generic به صورت nested استفاده کرد.