گزارش طراحی پروژه درس معماری کامپیوتر(ALU)

نام استاد: دکتر سمیه کاشی

اعضای گروه:فاطمه رزاقی ، محمد صادق همتی

مقدمه

یک واحد حساب و منطق (ALU) طراحی شده است که مثل یک ماشین حساب کوچک داخل پردازنده عمل می کند. این مدار دو عدد هشت بیتی با علامت را به‌عنوان ورودی می‌گیرد و بر اساس یک کد چهار بیتی، عمل مورد نظر را انتخاب می کند و نتیجه شانزده بیتی آن را بلافاصله روی خروجی می گذارد . مجموعه ی عملیاتی که این ALU اجرا می کند فقط به جمع و تفریق محدود نمی شود و ضرب، تقسیم، باقی‌مانده، توان با نماهای کوچک ، جذر عدد ، لگاریتم با پایه ی دلخواه و همچنین عملگرهای بیتی ANDو OR را هم در بر می‌گیرد. منطق مدار با زبان VHDL نوشته شده و در محیط شبیه‌سازی Active-HDL نسخه 13.0 کامپایل و اجرا شده است. برای اطمینان از درستی عملکرد، یک تست بنچ همه ی کدهای عملیاتی را امتحان کرده و نشان داده است که خروجی در هر حالت با پاسخ نظری مطابقت دارد.

1. کتابخانه‌های استفاده شده و دلیل انتخاب

- **ieee.std_logic_1164**

پرکاربردترین بسته VHDL است و تمام انواع پایه ای منطق (std_logic) و (std_logic_vector) و عملگرهای AND ، ORو ... را تعریف می کند. بدون این کتابخانه نمی توان سیگنال های دیجیتال را توصیف کرد.

- **ieee.numeric_std**

نوع های عددی علامت دار (signed) و بدون علامت (unsigned) و توابع کمکی مانند resize، to_integer و to_signed را فراهم می‌کند. همه ی عملیات جمع، تفریق، ضرب و تقسیم روی بردارهای دودویی با این بسته انجام می‌شود.

- **Integer-only algorithms**

جذر، لگاریتم و توان با روش های bitwise و حلقه ای مستقیما درون پروسس اصلی پیاده شده اند.

2. رابط ماژول (Entity) و انتخاب طول بیت ها

- ورودی های **A**و **B**هرکدام 8 بیتی علامت دار هستند و بازه ی عددی –128 تا +127 را می پوشانند.
- سیگنال **Operation**چهار بیت دارد و امکان 16 کد عملیاتی را فراهم می‌کند (10 کد استفاده شده است).
- خروجی **Result**شانزده بیتی است تا نتایج بزرگ تر یا منفی (مثل ضرب دو عدد 8 بیتی)در آن جا شود.
- خروجی **Carryout** برای تشخیص سرریز جمع 9 بیتی استفاده می شود.

3. معماری و فرایند اصلی

کل منطق داخل یک فرایند ترکیبی process(A, B, Operation)قرار دارد؛ هر تغییری روی ورودی ها بلافاصله خروجی را به روزرسانی می کند و نیازی به سیگنال کلاک نیست.

4. انتخاب عمل با دستور case

– **0000جمع :**هر ورودی با تابع resizeاز 8 به 16 بیت گسترده می‌شود تا بیت علامت حفظ شود؛ سپس جمع انجام می‌گیرد.

– **0001تفریق :**دقیقاً مانند جمع است ولی با عملگر منفی.

– **0010ضرب :**ابتدا هر ورودی به 16 بیت تبدیل می شود؛ حاصل ضرب ممکن است بزرگ تر از 16 بیت شود و با resizeبریده می شود تا فقط 16 بیت پایانی باقی بماند.

– **0011توان :(A^B)** مقدار B با یک حلقهٔ ضرب متوالی محاسبه می شود.

AND – 0100بیتی :عمل AND مستقیماً روی بیت های دو بردار انجام می‌شود و سپس نتیجه به نوع signedتبدیل می شود تا خطا ندهد.

OR – 0101بیتی :مشابه AND ولی با عملگر OR

– 0110جذر صحیح :اگر A منفی نباشد، جذر عدد صحیح A به‌صورت integer-only و با الگوریتم بیت‌به‌بیت محاسبه می‌شود و تنها بخش صحیح آن به خروجی اختصاص می‌یابد.

– 0111لگاریتم پایهٔ دلخواه :((log_B(A)) شرط A>0 و B>1 بررسی می شود؛ در یک حلقه A بارها بر B تقسیم می شود تا کوچک تر از پایه شود، و تعداد تقسیم ها کف لگاریتم است.

– 1000تقسیم صحیح :اگر B صفر نباشد، Aبر B تقسیم شده و خروجی می شود؛ در غیر این صورت صفر برگردانده می‌شود.

– 1001باقیمانده :(A mod B) همان بررسی تقسیم بر صفر انجام می شود، سپس عملگر modنتیجه باقی‌مانده را تولید می‌کند.

– othersکد نامعتبر :برای هر کد تعریف نشده خروجی صفر 16 بیتی تولید می شود.

6.کنترل سرریز و خطا

قبل از انجام عمل جمع و ضرب، ورودی ها از 8 به 16 بیت گسترش می‌یابند تا بیت علامت حفظ شود و فضای کافی برای رشد نتیجه ایجاد گردد.

تقسیم و باقیمانده ابتدا مقدار صفر بودن Bرا چک می‌کنند و در صورت صفر بودن، نتیجه را صفر می سازند.

برای جذر عدد منفی و لگاریتم با ورودی یا پایه ی نامعتبر، خروجی صفر داده می شود.