به نام خدا

دانشگاه صنعتی شریف
دانشکده مهندسی برق

طراحی سیستم های مبتنی بر ASIC/FPGA

تمرین سوم

امیرحسین یاری
۹۹۱۰۲۵۰۷

۲۱ فروردین ۱۴۰۳

فهرست مطالب

پیاده‌سازی با استفاده از IP Core

الف

Q1_Module.v کد Verilog ماژول ALU (واحد پردازشی عملیاتی) را با استفاده از IP Core های ضرب و جمع ممیز شناور زایلینکس پیاده سازی می کند. ALU این قابلیت را دارد که دو عدد ممیز شناور را جمع یا تفریق کند، و همچنین عملیات ضرب را انجام دهد. در اینجا، برای پیاده سازی عملیات ضرب و جمع، از یک ضرب کننده و یک جمع کننده استفاده شده است.

ماژول ALU دارای ورودی هایی مانند دو عدد ۳۲ بیتی a و b که به عنوان ورودی های عملیاتی استفاده می شوند، سیگنال کلاک (clk)، ورودی عملیاتی که تعیین می کند کدام عملیات (جمع، تفریق یا ضرب) انجام شود و همچنین چندین خروجی از جمله نتیجه، overflow و underflow برای ممیز شناور و invalid_op برای نشان دادن عملیات نامعتبر می باشد.

سپس، Core IP های ضرب و جمع ممیز شناور موجود در زایلینکس به عنوان ضرب کننده و جمع کننده استفاده شده اند. این IP ها عملیات ضرب و جمع را انجام می دهند و نتایج به عنوان خروجی های میانی (intermediate result) تولید می کنند.

سپس، با استفاده از یک بلاک always که به صورت همیشگی فعال است، عملیات مورد نیاز انتخاب و اعمال می شود. در این بلاک، با توجه به مقدار ورودی عملیاتی، نتیجه و مقادیر overflow، underflow و invalid_op بر اساس نتایج میانی ضرب و جمع انتخاب و تعیین می شوند.

پس از آن، هر عملیات به صورت موازی انجام می شود و نتایج به عنوان خروجی های ALU ارائه می شود. این طراحی بهینه است زیرا تنها یک ضرب کننده و یک جمع کننده استفاده می شود، که منابع سخت افزاری را کاهش می دهد و همچنین سرعت عملیات را افزایش می دهد.

منطق کنترل ALU را در تصویر زیر مشاهده می‌کنید.

```
// Control logic for selecting operation
always @* begin
    case(operation)
        6'b000000: begin // Addition
            result = add_sub_result;
                overflow = add_sub_overflow;
                underflow = add_sub_underflow;
                invalid_op = add_sub_invalid_op;
        end
        6'b000001: begin // Subtraction
            result = add_sub_result;
                overflow = add_sub_overflow;
                underflow = add_sub_underflow;
                invalid_op = add_sub_invalid_op;
        end
        6'b000010: begin // Multiplication
                result = multiply_result;
                underflow = multiply_underflow;
                overflow = multiply_overflow;
                invalid_op = multiply_invalid_op;
        end
        default: begin
            result = 32'b0; // Default result for invalid operation
            invalid_op = 1;
        end
    endcase
end
```

ب

این تست بنچ برای ماژول Q1_Module طراحی شده است تا عملکرد آن را در شرایط مختلف از جمله جمع، تفریق و ضرب مورد آزمون قرار دهد.

در این تست بنچ، ابتدا ورودی های ماژول شامل دو عدد ممیز شناور ۳۲ بیتی (a و b) و نوع عملیات مورد نظر (operation) تعیین می‌شود. همچنین سیگنال کلاک نیز تولید می‌شود.

سپس عملکرد ماژول در شرایط مختلف تست می‌شود. برای هر عملیات (جمع، تفریق و ضرب)، مقادیر مورد انتظار برای نتایج محاسبه شده و وضعیت‌های underflow، overflow و invalid_op برای هر عملیات بررسی می‌شوند.

تصویر زیر خروجی تست بنچ را نشان می‌دهد.

```
Addition Test:
a = 01000001001000000000000000000000, b = 01000000101000000000000000000000, operation = 000000
Result = 01000001011100000000000000000000, Underflow = 0, Overflow = 0, Invalid Operation = 0

Subtraction Test:
a = 01000001001000000000000000000000, b = 01000000101000000000000000000000, operation = 000001
Result = 01000000101000000000000000000000, Underflow = 0, Overflow = 0, Invalid Operation = 0

Multiplication Test:
a = 01000001001000000000000000000000, b = 01000000101000000000000000000000, operation = 000010
Result = 01000010010010000000000000000000, Underflow = 0, Overflow = 0, Invalid Operation = 0
```

با بررسی نتایج تست‌بنچ با این سایت صحت عملکرد مدار را تایید می‌کنم.

ج

با اضافه کردن کد زیر در تست‌بنچ

```
initial begin
    $dumpfile("withIPPower.vcd");
    $dumpvars(1, Q1_Module_TB.uut);
end
```

و به کمک ابزار xpower توان ایستا و پویای مدار را محاسبه می‌کنیم که مقادیر آن را در تصویر زیر مشاهده می‌کنید.

	Total	Dynamic	Quiescent
Supply Power (W)	0.083	0.001	0.082

د

داخل فایل IP_Core_Report.pdf می‌توانید گزارش کاملی از منابع مصرفی را مشاهده کنید.

ه

با ایجاد Constraint مربوط به کلاک، بیشترین فرکانس کلاک را 256MHz بدست آوردیم.

```
NET "clk" TNM_NET = clk;
TIMESPEC TS_clk = PERIOD "clk" 3.9 ns HIGH 50%;
```

$$\frac{1}{3.9n} \cong 256MHz$$

پیاده‌سازی بدون استفاده از IP Core

الف

ALU.v ماژول یک واحد پردازش ALU می باشد.
ورودی‌های این ماژول شامل کلاک مدار (clk) برای همگام‌سازی عملیات، دو عدد ۳۲ بیتی به نام‌های a_operand و b_operand که عددهایی هستند که قرار است عملیات حسابی بر روی آن‌ها انجام شود، و یک ورودی ۴ بیتی به نام Operation که کد عملیات مورد نظر (مانند جمع، تفریق و ضرب) را مشخص می‌کند.
خروجی‌های این ماژول شامل یک عدد ۳۲ بیتی به نام ALU_Output که نتیجه عملیات حسابی را نشان می‌دهد، و سه خروجی تک بیتی دیگر به نام‌های Exception، Overflow و Underflow می‌باشد.
در این ماژول، عملیات جمع و تفریق با استفاده از یک واحد Addition_Subtraction و عملیات ضرب با استفاده از یک واحد Multiplication انجام می‌شود. ورودی‌ها به این دو واحد با توجه به Operation مشخص می‌شود و نتایج آن‌ها به‌ترتیب در متغیرهای Add_Sub_Output_reg و Mul_Output_reg ذخیره می‌شود.
سپس با توجه به مقدار Operation، نتایج محاسبه شده در مراحل قبل به‌صورت چندگانه در متغیرهای خروجی مربوطه (ALU_Output، Exception، Overflow، Underflow) قرار داده می‌شوند تا به‌عنوان خروجی‌های نهایی ماژول تولید شوند.

ب

ALU_TB.v یک محیط تست برای ماژول ALU فراهم می‌کند تا عملکرد آن را در شرایط مختلف تست کند. این محیط تست شامل تغییر ورودی‌های ماژول و مشاهده خروجی‌های آن در طول زمان است.
ابتدا مقادیر ورودی‌ها از جمله clk ، a_operand و b_operand (دو عدد ورودی)، و Operation (عملیات مورد نظر) به ماژول ALU اعمال می‌شود.
سپس با استفاده از یک بلاک initial، کلاک ابتدا با مقدار صفر مقداردهی می‌شود، سپس هر ۵ واحد زمانی (با توجه به timescale تعیین شده)، مقدار کلاک تغییر می‌کند. این فرآیند برای همیشه ادامه دارد، بنابراین کلاک با فرکانس مشخصی پالس می‌دهد.
در بلاک initial دیگر، مقادیر ورودی‌ها و عملیات‌ها برای تست‌های مختلف مشخص می‌شوند. برای هر تست، مقادیر ورودی تنظیم می‌شوند، سپس یک دوره زمانی انتظار داده می‌شود تا اعمال و تحلیل نتیجه انجام شود. سپس با استفاده از display، جزئیات هر تست از جمله مقادیر ورودی و خروجی‌های ماژول چاپ می‌شود.

ج

با اضافه کردن کد زیر در تست‌بنچ

```
initial begin
        $dumpfile("withIPPower.vcd");
        $dumpvars(1, ALU_tb.ALU_inst);
 end
```

و به کمک ابزار xpower توان ایستا و پویای مدار را محاسبه می‌کنیم که مقادیر آن را در تصویر زیر مشاهده می‌کنید.

	Total	Dynamic	Quiescent
Supply Power (W)	0.085	0.003	0.082

د

داخل فایل Without_IP_Core_Report.pdf می‌توانید گزارش کاملی از منابع مصرفی را مشاهده کنید.

ه

با ایجاد Constraint مربوط به کلاک، بیشترین فرکانس کلاک را 175 بدست آوردیم.

```
NET "clk" TNM_NET = clk;
TIMESPEC TS_clk = PERIOD "clk" 5.7 ns HIGH 50%;
```

$$\frac{1}{5.7n} \cong 175MHz$$

مقایسه دو پیاده‌سازی انجام شده

همانطور که انتظار داشتیم، پیاده‌سازی با IP Core بهینه‌تر می‌باشد. باتوجه به نتایج بدست آمده می‌توان گفت گفت که پیاده‌سازی با IP Core کلاک بالاتری دارد و همچنین توان کمتری مصرف می‌کند.
باتوجه به مقایسه انجام شده بین منابع مصرفی این دو پیاده‌سازی، پیاده‌سازی با IP Core تعداد کمتری register و LUT دارد.

مزایای استفاده از IP Core

استفاده ازIP Core های ISE یک سری از مزایای قابل توجه در طراحی و توسعه FPGA دارد. در ادامه به برخی از این مزایا اشاره می‌کنم:

۱. افزایش سرعت توسعه: IP Coreهای ISE شامل مجموعه‌ای از ماژول‌های آماده است که از پیش توسعه داده شده‌اند. این ماژول‌ها عموماً وظایف متداولی را انجام می‌دهند مانند UART یا SPI، کنترل PWM، و غیره. با استفاده از این IP Coreها، زمان نیاز برای پیاده‌سازی و توسعه کاهش می‌یابد.

۲. کاهش خطاها: IP Coreهای ISE به‌عنوان تکه‌های نرم‌افزاری آماده‌ای ارائه می‌شوند که توسط توسعه‌دهندگان معتبر طراحی و تست شده‌اند. این به توسعه‌دهندگان اجازه می‌دهد تا از خطاهای مرتبط با پیاده‌سازی و توسعه ماژول‌های خاص خود جلوگیری کنند.

۳. افزایش قابلیت استفاده و قابلیت استفاده مجدد: IP Coreهای ISE می‌توانند به عنوان ماژول‌های آماده در طراحی‌های بعدی مورد استفاده قرار گیرند. این به توسعه‌دهندگان امکان مجدد استفاده از کدهای توسعه داده شده را فراهم می‌کند و زمان و هزینه توسعه را کاهش می‌دهد.

۴. بهبود عملکرد و بهره‌وری: با استفاده ازIP Core های ISE که به‌طور بهینه برای پلتفرم FPGA طراحی شده‌اند، می‌توان بهبود عملکرد و بهره‌وری سیستم را به‌دست آورد. اینIP Core ها به‌طور کامل با معماری FPGA سازگار هستند و بهینه‌سازی شده‌اند تا بر روی آنها به بهترین نحو عمل کنند.

۵. پشتیبانی و توسعه مستمر: شرکت‌هایی که IP Coreهای ISE را ارائه می‌دهند، معمولاً پشتیبانی و به‌روزرسانی مستمر برای محصولات خود فراهم می‌کنند. این به توسعه‌دهندگان امکان می‌دهد که به سرعت به مشکلات رایج پاسخ دهند و از امکانات جدید بهره‌مند شوند.