1-پیاده سازی محاسبات به روشpipeline

با استفاده از FPGA خانوادۀ Spartan3-400 ،

الف -ماژولی به زبان Verilog بنویسید که دو عدد ۳۲ بیت بدون علامت را بصورت pipeline در یکدیگر ضرب نماید (به کمک ضرب کننده های ۱۶ بیتی .)

ب -با استفاده از ماژول قسمت الف، ماژولی به زبان Verilog بنویسید که دو عدد ۳۲ بیت با علامت(با نمایش مکمل)۲ را بصورت pipelineدر یکدیگر ضرب نماید .

پ -در هر قسمت، سرعت clock خوردن و حجم منابع اشغال شده بر روی سخت افزار را استخراج و با یکدیگر مقایسه کنید .

ت -در صورتی که محاسبۀ حاصلضرب را به اجزای کوچکتر شکسته شود، آیا تأثیری بر روی سرعت clock خوردن و حجم منابع اشغال شده دارد یا خیر؟ چرا؟

2- پیاده سازی محاسبات به روشhardware-reuse

با استفاده از FPGA خانوادۀ Spartan3-400 ،

الف -ماژولی به زبان Verilog بنویسید که دو عدد ۳۲ بیت بدون علامت را به کمک hardware-reuse در یکدیگر ضرب نماید(به کمک تنها یک عدد ضربکننده های ۱۸بیتی .)

ب -با استفاده از ماژول قسمت الف، ماژولی به زبان Verilog بنویسید که دو عدد ۳۲ بیت با علامت(با نمایش مکمل)۲ را بصورت hardware-reuse در یکدیگر ضرب نماید .

پ -در هر قسمت، سرعت clock خوردن و حجم منابع اشغال شده بر روی سخت افزار را استخراج و با یکدیگر و با نتایج تمرین سری اول مقایسه کنید .

ت -در صورتی که محاسبۀ حاصلضرب را به اجزای کوچکتر شکسته شود، آیا تأثیری بر روی سرعتclock خوردن و حجم منابع اشغال شده دارد یا خیر؟ چرا؟

3- روش های بهبود زمان بندی

ایده های Retiming ، Repiplining و C-Slow Retining را در طراحی های دیجیتال توضیح دهید.

4- Retiming

کد زیر را در نظر بگیرید:

```
module Adder4to1(
  input clk, reset,
  input [9:0] A1, A2, A3, A4,
  output reg [11:0] Sum
);
reg [9:0] A1L, A2L, A3L, A4L;
reg [11:0] P1;
always@ (posedge clk) begin
  if (reset) begin
    A1L <= 0;
    A2L <= 0;
    A3L <= 0;
    A4L <= 0;
    P1 <= 0;
```

```
    Sum <= 0;
  end
  else begin
    A1L <= A1;
    A2L <= A2;
    A3L <= A3;
    A4L <= A4;
    P1 <= A1L+A2L+A3L+A4L;
    Sum <= P1;
  end
end
endmodule
```

با استفاده از تکنیک جابجایی رجیسترهای درونی (retiming) و بدون در نظر گرفتن قابلیت ابزار سنتز برای استفاده خودکار از این تکنیک، کد را به شکلی بازنویسی کنید که حداکثر فرکانس ماژول افزایش یابد .حداکثر فرکانس را در دو حالت روی Spartan3 XC3S400 FPGA بررسی کنید.

5- ضرب کننده

بر روی بسیاری از FPGA های شرکت Xilinx ، ضرب کننده های اختصاصی ۱۸×۱۸ بیت وجود دارد .میخواهیم به کمک این ضرب کننده ها عملیات ضرب ۳۶×۳۶ بیت را پیاده سازی کنیم .می دانیم که اگر هر دو عدد بدون علامت باشند، عملیات ضرب به شکل زیر قابل شکستن است:

$$A[35:0] \times B[35:0] = (A[17:0] \times B[17:0]) + ((A[17:0] \times B[35:18])<<18) + ((A[35:18] \times B[17:0])<<18) + ((A[35:18] \times B[35:18])<<36)$$

الف)ساختار توصیف شده را به شکل Pipeline دو طبقه و با استفاده از ۴ ضربکننده ۱۸×۱۸ پیاده سازی کنید.

طبقه اول Pipeline شامل ضرب کننده ها و طبقه دوم شامل یک جمع کننده ۴ به ۱ است.

ب)ساختار توصیف شده را به شکل Pipeline سه طبقه و با استفاده از ۴ ضربکننده ۱۸×۱۸ پیاده سازی کنید.

طبقه اول Pipeline شامل ضرب کننده ها و طبقه دوم شامل دو جمع کننده ۲ به ۱ و طبقه سوم شامل دو جمع کننده ۲به ۱ است.

پ)ساختار توصیف شده را به کمک روش HardwareReuse با استفاده از تنها ۱ ضربکننده ۱۸×۱۸ پیاده سازی کنید . همانطور که میدانید برای این کار نیاز به طراحی یک ماشین حالت متناهی کنترل کننده هست .توجه داشته باشید که شکل کلی پیاده سازی همچنان به شکل Pipeline باقی می ماند اما پیاده سازی هر طبقه با توجه به شرایط جدید تغییر میکند.

ت)سه پیاده سازی قبلی خود را از نظر حجم منابع اشغال شده و حداکثر فرکانس روی Spartan3 XC3S400 FPGA مقایسه کنید .دقت داشته باشید که ضرب کننده های ۱۸×۱۸ روی منابع اختصاصی خود پیاده سازی شده باشند و نه به شکل distributed ، در غیر این صورت تغییرات مورد نیاز را انجام دهید .همچنین توجه داشته باشید که حداکثر فرکانس بر اساس منابع combinational میان دو رجیستر محاسبه میشود .برای مقایسه حداکثر فرکانس یک ماژول سطح بالاتر بر روی ماژول خود بنویسد که ورودی ماژول را پس از ذخیره در یک رجیستر(ازخروجی رجیستر)به آن تحویل دهد.

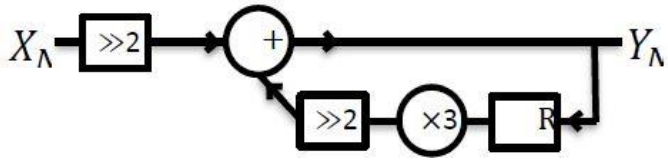
ث)اگر اعداد ورودی به شکل مکمل دو با علامت باشند، کافی است ابتدا اعداد منفی را به مثبت تبدیل کرد و پس از پردازش آنها مشابه حالت بدون علامت، در صورت نیاز حاصل را منفی کرد.

۶- روابط بازگشتی

فرمول بازگشتی زیر را در نظر بگیرید:

$$Yn=(1/4)Xn+(3/4)Yn-1 \quad n\geq 1 \quad , \quad Y0=0$$

بلوک دیاگرام زیر یک شکل پیاده سازی این فرمول را نشان می دهد (R نماد رجیستر و بنابرین معادل یک واحد زمانی تاخیر، و ≫ نماد شیفت به راست است)



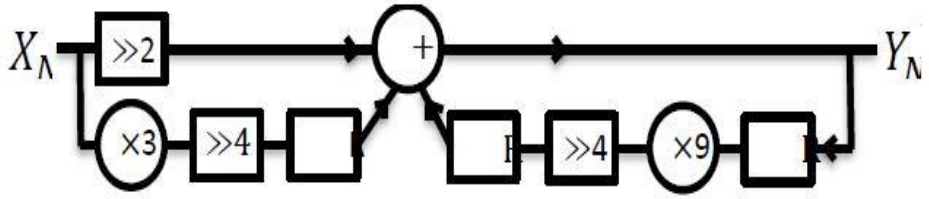
این شکل توسط کد زیر پیاده سازی شده است:

```
module RecursiveFormula(
  input clk, reset,
  input [5:0] Xn,
  output [9:0] Yn
);
reg [9:0] Ynm1 = 0; //initial condition
always @(posedge clk) begin
  if(reset) begin
    Ynm1 <= 0; //set to initial condition
  end
  else begin
    Ynm1 <= Yn;
  end
end
assign Yn = (Xn>>2)+((Ynm1*3)>>2);
endmodule
```

با جانشانی $YN-1$ ، فرمول را بر اساس $YN-2$ بازنویسی میکنیم:

$$YN=(1/4)Xn+(3/16)Xn-1+(9/16)Yn-2 \quad n\geq 1 \quad , \quad X0=Y0=Y-1=0$$

و بلوک دیاگرام زیر را برای آن در نظر میگیریم:



کد این بلوک دیاگرام پیاده سازی کنید (توجه داشته باشید که ورودی ها و خروجی نسبت به ماژول قبل تغییری ندارند.)

به منظور مقایسه حداکثر فرکانس این دو شکل پیاده سازی، یک ماژول سطح بالاتر بنویسید که ورودی ماژول RecursiveFormula را پس از ذخیره در یک رجیستر(از خروجی رجیستر)به آن تحویل دهد و خروجی ماژول را

هم در یک رجیستر ذخیره کند(توجه داشته باشید علت این کار آن است که حداکثر فرکانس بر اساس منابع combinational میان دو رجیستر محاسبه میشود)، سپس حداکثر فرکانس دو شکل پیادهسازی را روی Spartan3 XC3S400 FPGA بررسی کنید.