گزارش کار تمرین کامپیوتری ۱ CAD

کیارش خراسانی زواره

بابک حسینی محتشم

توضیح دیتاپس

همانطور که در فایل طرح ها قابل مشاهده است دیتاپس ارائه شده دو مدل است یکی مدل اولیه و دیگری مدل ثانویه که همراه با کد ها بارگزاری شده است.

مدلی که کار کرد مدل دوم بود به همین دلیل اون رو توضیح میدم و فقط مشکل مدل اول رو میگم.

مدل اول در بررسی ورودی هایی که در ۸ بیت پر اهمیتشان ۱ نداشتند (مثل عدد ۲) به مشکل خورد و توانایی برسیشون رو نداشت چون از دیدن اولین بعد از ۱ شروع به شیفت دادن میکرد و دقیق ۸ بار میخواست شیفت بده ولی اگه تعداد بیت های باقیمانده کمتر از ۸ بود این شیفت دادن مقدار عدد رو زیاد میکرد.

در مدل دوم دو نوع حافظه رم داریم یکی برای خواندن یکی برای نوشتن. دلیل این تفاوت قاعل شده این است که اولا حافظه رم اول فقط توانایی خواندن و load کردن را دارد و میتواند همزمان دو خانه را به عنوان خروجی بدهد و بخواند ولی حافظه دوم برای نوشتن است و در clock در یک خانه مینویسد.

۲ شیفت رجیستر برای ذخیره ورودی ها داریم که به طبع به اندازه ورودی یعنی ۱۶ هستند. دلیل استفاده از شیفت رجیستر این است که قصد داریم تا ۸ بار بیت را شیفت بدهیم تا به اولین یک برسیم و از اولین ۱(یا اگر یک ندیدیم بعد از ۸ بار) ۸ بیت بعدی را به عنوان ورودی به ضرب کننده میدهیم.

۱ ضرب کننده داریم که به هشت بیت پر اهمیت ورودی ها وصل است و ما زمانی که آماده ورود به مرحله ذخیره خروجی برسیم انتظار داریم ضرب کننده کار خود را انجام داده باشد.

یک شیفت رجیستر ۳۲ بیتی برای ذخیره خروجی داریم که ورودی آن ۱۶ بیت خارج شده از ضرب کننده به عنوان بیت های ۰ تا ۱۵ و ۱۶ بیت دیگر صفر برای بیت های پر اهمیت است. دلیل استفاده از شیفت رجیستر آن است که به تعداد صفر هایی که جلوی اعداد ورودی بود و برای ضرب خوانده نشد نتیجه ضرب را شیفت بدهیم که به نتیجه درست برسیم.(درست عین قرار دادن ۰ جلوی ضرب عددی در ۱۰ در سیستم ده دهی)

یک counter سه بیتی برای این داریم که کنترلر بداند از ۸ جفت عددی که قرار است در حافظه بخواند چند تا از آنها خوانده شده است.

یک counter سه بیتی دیگر برای این داریم که ورودی های خوانده شده را تا دیدن اولین ۱ شیفت بدهیم.
دلیل اینکه برای دو شیفت رجیستر یک counter استفاده شده این است که هدف آن است که بدانیم چند بار وارد چرخه شیفت شده ایم و باتوجه به بیت پر اهمیت هر شیفت رجیستر (و البته cout برای این counter) نتیجه میگیریم که آیا آن شیفت رجیستر باید شیفت پیدا کند یا نه.

یک counter ۴ بیتی برای شیفت نتیجه استفاده شده که با دو سیگنال ورودی میتواند بشمارد یک سیگنال shl1 (شیفت رجیستر شماره یک) و دیگریshl2 (شیفت رجیستر شماره دو) دلیل این انتخاب را باید به صورت زیر توضیح بدم

۳۲ بیت خروجی داریم که x بیت برای ورودی رجیستر ۱ و x2 بیت برای ورودی رجیستر ۲ شیفت انجام داده ایم و ۱۶ بیت از خروجی از ضرب کننده دریافت میشود.

تعداد بیت که باید به عنوان صفر جلوی عدد اضافه شود را y فرض میکنیم واضح است که x1+x2+y=16 (۱۶ بیت دیگر خروجی). حال ما برای counter به اندازه x1 + x2 میشماریم پس اگر بعد از آن به اندازه ای که cout برای این این شمارنده ای که ۴ بیت دارد (و یعنی تا ۱۶ میتواند بشمارد) صفر شود بشماریم در واقع به اندازه y شمرده ایم. از همین منطق در کنترلر هم استفاده میکنیم و با این counter تعداد صفر هایی که باید جلوی عدد قرار گیرد را محاسبه میکنیم.

توضیح کنترلر

تعداد استیت های کنترلر زیاد است ولی به طور کل اگر قرار باشد توضیح بدهم.

IDLE: ماشین منتظر است که سیگنال شروع به کار را دریافت کند.

Wait: منتظر است که سیگنال شروع صفر شود.

Input: ورودی را از حافظه میخواند و در رجیستر قرار میدهد پس به طبع سیگنال load برای دو رجیستر ورودی باید بدهد.

SHR1SHR2: به طور هماهنگ تا زمانی که بیت پر اهمیت حداقل یکی از رجیستر ها ۱ شود یا ۸ بار شیفت داده باشیم عملیات شیفت را انجام میدهد. با توجه به اینکه آیا ۸ بار شیفت دادیم یا بیت پر اهمیت هر دو رجیستر ۱ بود(که یعنی باید وارد مرحله ذخیره روی حافظه شویم) یا اینکه بیت پر اهمیت رجیستر ورودی اول یک شده(باید بیت های ورودی های دوم را شیفت دهیم) یا بیت پر اهمیت رجیستر ورودی دوم یک شده باشد(که یعنی باید بیت های ورودی اول را شیفت دهیم.)تصمیم میگیرم که وارد کدوم استیت شویم.

SHR1: بیت های رجیستر اول را شیفت میدهیم تا به اولین ۱ برسیم یا ۸ بیت خوانده باشیم.

SHR2: بیت های رجیستر دوم را شیفت میدهیم تا به اولین ۱ برسیم یا ۸ بیت خوانده باشیم.

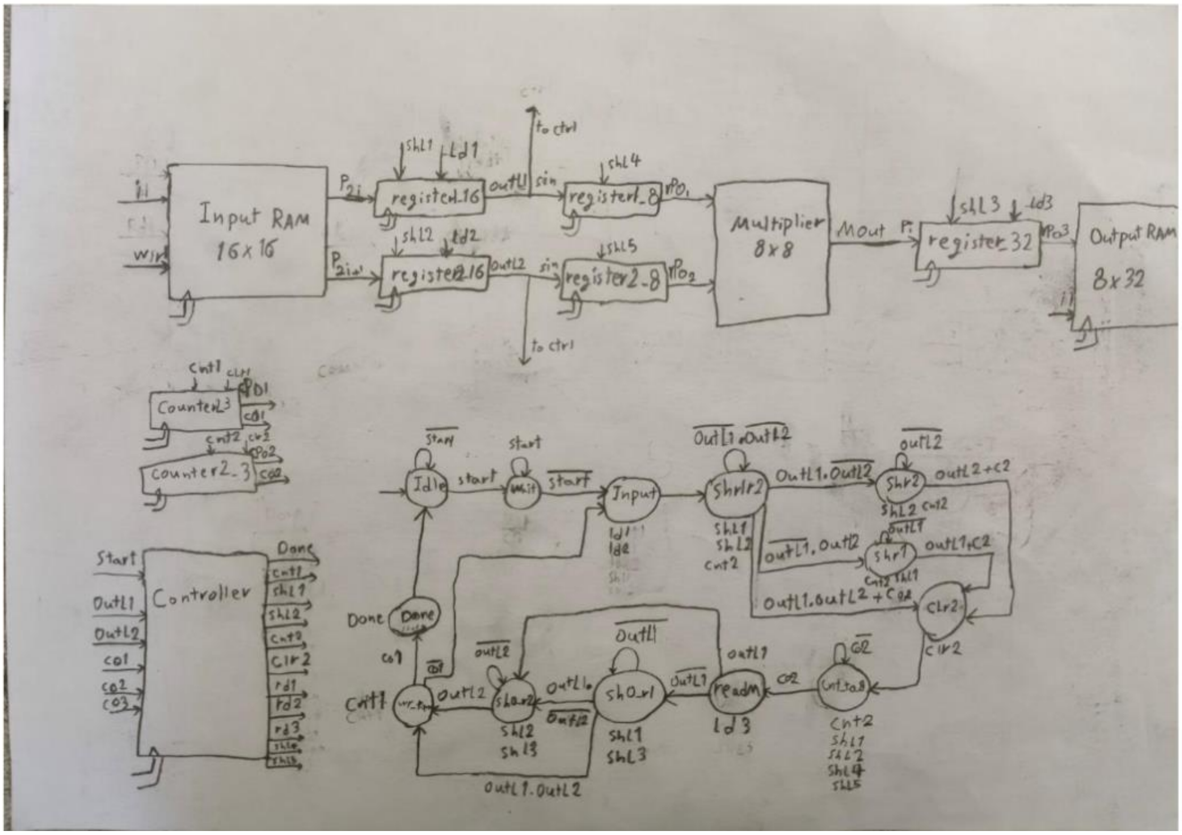
ReadM: هدف آماده سازی برای خواندن ورودی روی خروجی است. پس چک میکنیم آیا بیتی skip شده که باید ضرب میشده یا نه(co3 برابر صفر است یا خیر) اگر بیتی بود که skip شده وارد استیت SH0 میشویم و در غیر این صورت وارد استیت WRTORAM میشویم.

SH0: در این استیت به دلیلی که در بخش دیتاپس گفته شد تا جایی که co برای counter برابر ۱ شود خروجی ای که در رجیستر است را شیفت میدهیم و بعد از اتمام کار وارد WRTORAM میشویم.

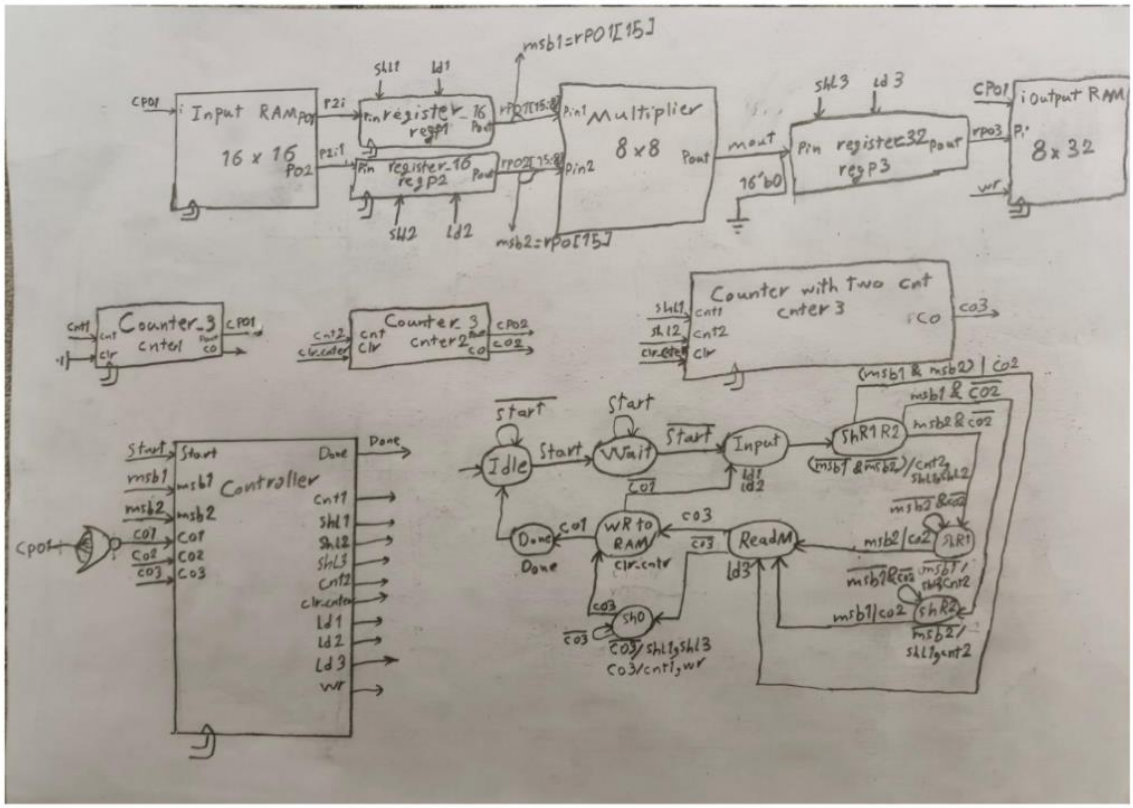
WRTORAM: در این استیت عددی که در رجیستر خروجی است را روی حافظه میریزیم و اگر هر هشت جفت ورودی خوانده شده بود وارد استیت DONE برای پایان کار می شویم و اگر تمام اعداد خوانده نشده بود دوباره به مرحله Input برمیگردیم.

DONE: سیگنال done را که نشان دهنده پایان کار است را یک clock cycle روشن کرده و دوباره وارد استیت IDLE میشویم.

طرح اولیه


Input RAM
16x16
register1_16
register2_16
register1_8
register2_8
to ctrl
to ctrl
Multiplier
8x8
Mout
register_32
Output RAM
8x32
Counter1_3
counter2_3
Start
Controller
Done
Idle
Input
readm
ld3

طرح نهایی


Input RAM
16 x 16
register_16 reg P1
register_16 reg P2
Multiplier
8 x 8
register_32 reg P3
Output RAM
8 x 32
Counter_3 cnter1
Counter_3 cnter2
Counter with two cnt cnter3
Controller
Start
Done
Idle
Wait
Input
WR to RAM
ReadM
ShR1 R2
ShR1
ShR2
ShL0