بسم الله الرحمن الرحیم

**گزارش پروژه طراحی پردازنده SAYEH**

*طراح و پیاده ساز : مریم ابراهیم زاده شماره دانشجویی: 9431052*

در این پروژه به طراحی یک کامپیوتر پایه پرداختم که از سه قسمت controller ,datapath , memory تشکیل شده بود .

کلیت کار این پردازنده خواندن دستورات ازحافظه و اجرای آن ها است.



در ابتدا datapath رسم شده د رتعریف پروژه را طراحی پیدا سازی کردم .به این صورت که اول module هایی که داخل ان هست را طراحی کردم سپس ان ها در datapath به هم وصل کردم.

مختصری از module های datapath ذیل امده است.

*Register file :*

این قسمت دارای کلاک است و یک ورودی 16 بیت دارد که من اگر کلاک پایین رونده بیایید و سیگنال ها ینوشتن این قسمت فعال باشد ورودی این ماژول را در RDمینویسم.

و باهر تغییر کلاک مقدار RS & RD را که از ادرس ان ها را گرفته ام میخوانم و روی خروجی ها میگذارم.

*ALU :*

قسمت مغز اصلی این پردازنده است و در ان من توابع مورد نیاز را پیاده سازی کردم علاوه بر آن ها ضرب و مکمل دوم و شیفت را هم پیاده سازی کرده ام.

این ماژول کلاک ندارد.

*WP:*

این ماژول تنها یک ریست و این که بتواند داده قبلی را با ورودی اش جمع کند.

*AdressUnit:*

این ماژول برای کنترل ادرس که مموری میدهیم است واز address logic و program counter تشکیل شده است

*IR:*

دستور که از حافظه می اید اول وارد این رجیستر میشود.

*Flagregister :*

برای کنترل کردن فلگ های z & cاست.

حال بعد از ساخت این ها با استفاده از سیم های address,databus,opndbus,..

به هم وصل میشوند.

ورودی وخروجی های این ماژول بدین صورت است.

entity datapath is

port(

clk ,

ResetPC, PCplusI, PCplus1, RplusI, Rplus0:in std\_logic;

RS\_On\_addressUnitRside,Rd\_On\_addressUnitRside,

enablePc : in std\_logic;

b15to0 , aandb, aorb ,notb ,aaddb ,asubb ,axorb ,acompb,shrB ,shlB,twicecompb,amulb : in std\_logic;

RFLwrite,RFHwrite,

wpadd,wpreset,

irload,

srload,

address\_on\_databus,alu\_on\_databus,

ir\_on\_lopndbus,ir\_on\_hopndbus,rfright\_on\_opndbus,

cset,creset,zset,zreset,shiftir,

shadow: in std\_logic;

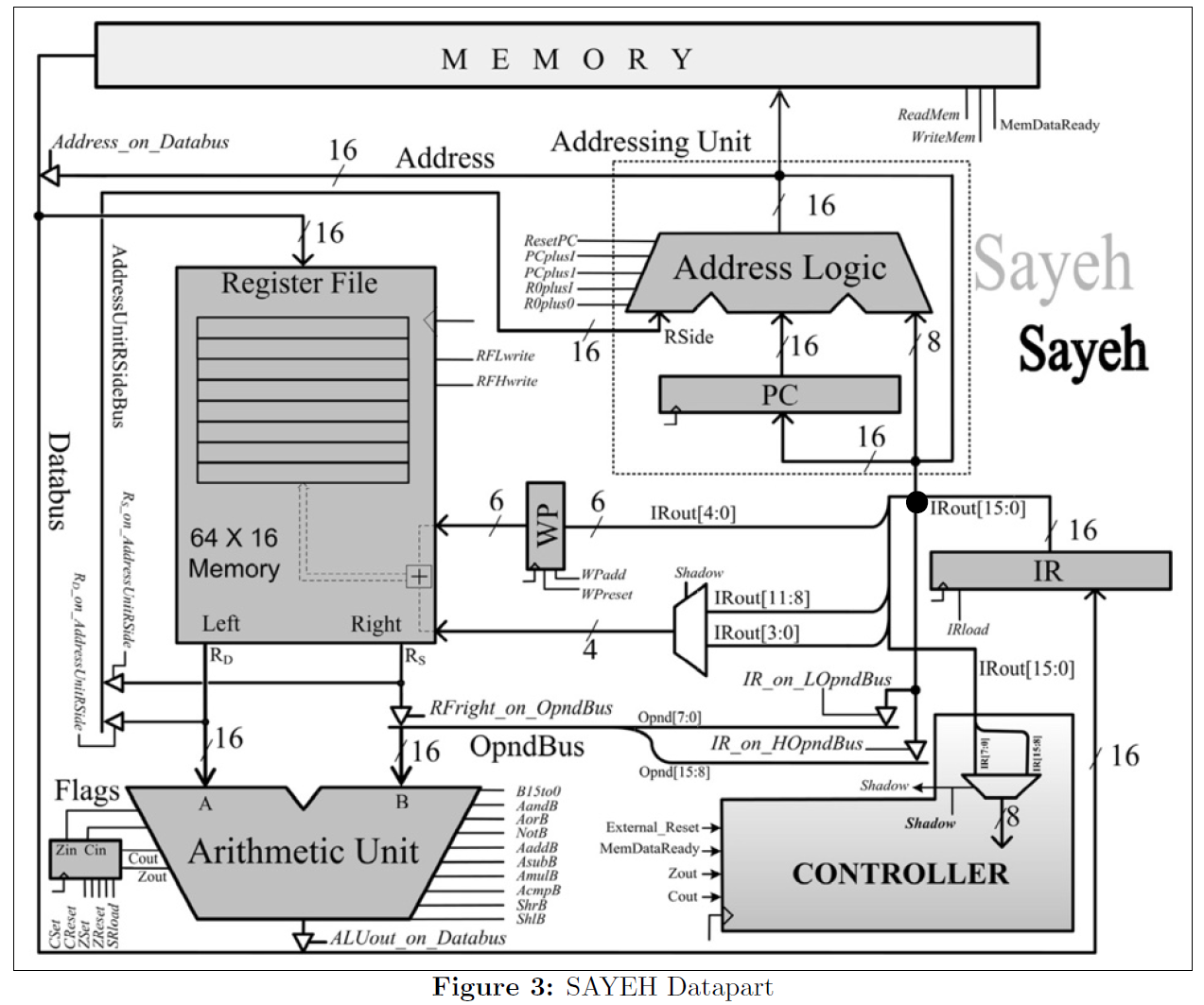
databus: inout std\_logic\_vector (15 downto 0);

addressbus,instruction: out std\_logic\_vector(15 downto 0);

cout,zout :out std\_logic

);

end datapath;



کلا 29 دستور داریم که باید ان هارا توسط کنترلردیکد کنیم و برای اجرا به datapathبفرستیم.

حال باید controller را طراحی کنیم.

در واقع خروجی های controllerمیشود ورودی های datapath زیرا باید آن را کنترل کند.

درکنترلر من 7 state یا حالت دارم:

**TYPE State\_type IS (reset, halt, fetch, fetch2, fetch3, decode, exec2); -- Define the states**

**Reset:**

برای شروع دوباره است و program counterرا یک میکند.

**Halt:**

اگر وارد این حالت بشود به اصلاح گیر میکند تا ریست بیاید و ان را از این حالت خارج کند.

**Fetch:**

برا یاین است که سیگنال readmemیک شود که از مموری بخوانیم.

**fetch2:**

در این حالت دستور وارد IRشده است ما irload را یک میکینم تا داده وارد کنترلر شود.

**fetch3:**

این حالت برای چک کردن این که ایا داریم دستو shadow را اجرا میکنیم یا دستور اصلی را است.

**Decode:**

برای این است که بفهمیم الان باید چه دستوری را اجرا کنیم.قسمتی از کار اجرا در این جا اجرا میشود.

**exec2:**

برای دستور هایی که بیا اجرا به بیش از یک کلاک احتیاج دارند.

ابتدا ماشین د رحالت ریست است سپش وارد حالت fetch میشود و در این حالت فلگ مخصوص خواندن ا زحافظه را یک میکند.سپس وارد حالت 2 fetch میشود و در این جا دستور وارد رجیستر IRمیشود سپس وارد حالت fetch3میشود و سپس چک میکند دارد دستور shadow را اجرا میکند یا نه بعد از این وارد حالت دیکد کردن میشود و قسمتی از اجرا را در همین حالت انجام میدهم. اگردستوری به دو کلاک نیاز داشته باشد وارد exec2میشود وگرنه دوباره وارد fetchمیشود.

قبل از وارد شدن به دیکد shdow را یک میکنیم.در حالت دیکد چک میکنیم کدام دستور است.اگر دستور جزو ان هایی باشد که immediate ندارند چک میکنیم :

IF shadowsig = '1' THEN

nexts <= fetch;

PCplus1 <= '1';

EnablePC <= '1';

ELSE

shadowsig <= '1';

nexts <= fetch3;

END IF;

این شرط میگوید اگر shadow یک بود یعنی الان این دستوری که اجرا کردی دستور shadow بوده است.وگرنه shadow

را یک میکنیم تا برود و دستور shadowرا اجرا کند.

دستور هایی که بیشتر از یک کلاک لازم داشته اند این شرط را در قسمت دوم اجرایشان دارند.

دستور هایی که بیشتر از یک کلاک لازم دارند در ذیل ذکر شده اند:

**and | or | not | shl | shr | add | sub | mul | cmp |oup|inp|mvr|lda|sta**

خلاصه ای از دستور ها:

Nop:

که برای یک کلاک کاری انجام نمی دهد.

Halt:

ماشین را به حالت haltمیبرد.

Szf,czf,scf,ccf,cwp:

در این دستور ها تنها فلگ مودنظر در کنترلر را یک میکنیم.

Mvr:

نیاز به دو کلاک دارد ولی کلا باید این ها یک شود

Rfright\_on\_opndbus<=1

B0to15<=1

Alu\_on\_databus<=1

Rflwrite<=1

Rfhwrite<=1

Lda:

نیاز به دو کلاک دارد ولی کلا باید این ها یک شود

Rs\_on\_addressunitRside<=1

Rplus0<=1

readMemory<=1

rfhwrite<=1

rflwrite<=1

Sta:

نیاز به دو کلاک دارد ولی کلا باید این ها یک شود

Rf\_on\_opndbus<=1

B0to15<=1

Alu\_on\_databus<=1

Rd\_on\_addressunitRside<=1

Rplus0<=1

writeMemory<=1

and | or | not | shl | shr | add | sub | mul | cmp:

دستور های مربوط به ALUهمه در دو کلاک انجام میشوند.

مثلا and:

RFright\_on\_OpndBus <= '1';

aandb <= '1';

ALU\_on\_Databus <= '1';

RFLwrite <= '1';

RFHwrite <= '1';

SRload <= '1';

Mil:

Ir\_on\_lopndbus<=1

B0to15<=1

Alu\_on\_databus<=1

Rflwrite<=1

Mih:

Ir\_on\_Hopndbus<=1

B0to15<=1

Alu\_on\_databus<=1

Rfhwrite<=1

Spc:

pcplusI<=1

address\_on\_databus<=1

rflwrite<=1

rfhwrite<=1

jpa:

Rd\_on\_addressunitrside<=1

rplusI<=1

Jpr:

PcplusI<=1

و بقیه دستور ها شبیه به دستور اخر هستند که ذکر شد.

مشکلاتی که من در طراحی و پیاده سازی این پروژه با آن مواجهه شدم:

در ابتدا من باید سیم databusرا مقدار Zمیدادم که چون نداده بودم مقدار ان مقدارینمیشد که مموری روی ان نوشته بود و U میشد.

سپس مشکل دیگر که داشتم اگر به ALUهم زمانی که داده وارد ان میشد سیگنال انجام عملیانموردنظر میدادم مقدار خروجی Uبود و بای همین من مجبور شدم تعدا حالت هایم را زیاد کنم.