**بسمه تعالی**

**تمرین سری 4 معماری**

**طراحی پردازنده با روش میکرو پروگرمینگ**

**محمد صادق خراسانی**

**95108008**

**فرضیات مورد نیاز**

قبل از هر کاری برای پیشروی کار میبایست برای میکرو اپریشن ها نام انتخاب کرد و فرمت میکرو اینستراکشن ها را تعیین کرد.

به همین دلیل من برای هر میکرو اینستراکشن (یا ورد ( contrl rom3فیلد برای حداکثر 3 میکرو اپریشن همزمان انتخاب کردم و سپس مانند کتاب 2 بیت برای conditionو2بیت برای branchدر نظر گرفتم.در 7 بیت کم ارزش ورد رام هم ادرس محل بعدی که باید از رام خوانده شود قرار دارد.

همانطور که تخمین زده شده بود و همانطور که در نهایت دیده شد نیازی به رام با بیشتر از 7 بیت ادرس نبود جرا که با 7 بیت میتوانیم 128 خانه را ادرس دهی کنیم در حالی که بیشتر از 90 خانه برای دستورات داده شده نیازی نبود.

به همین دلیل رام با ورد 23 بیت و ادرس دهی 7 بیتی برای کنترل یونیت انخاب کرده ام.

سپس تمام دستورات را با نام میکرو اپریشن ها در اکسل وارد کرده و کد وریلاگ مربوط به آن را در فرمولی در اکسل که بسته به فیلد های نام میکرو اپریشن هاست گرفته ام تا قابلیت تغییر دادن میکرو اینستراکشن ها را داشته باشم.

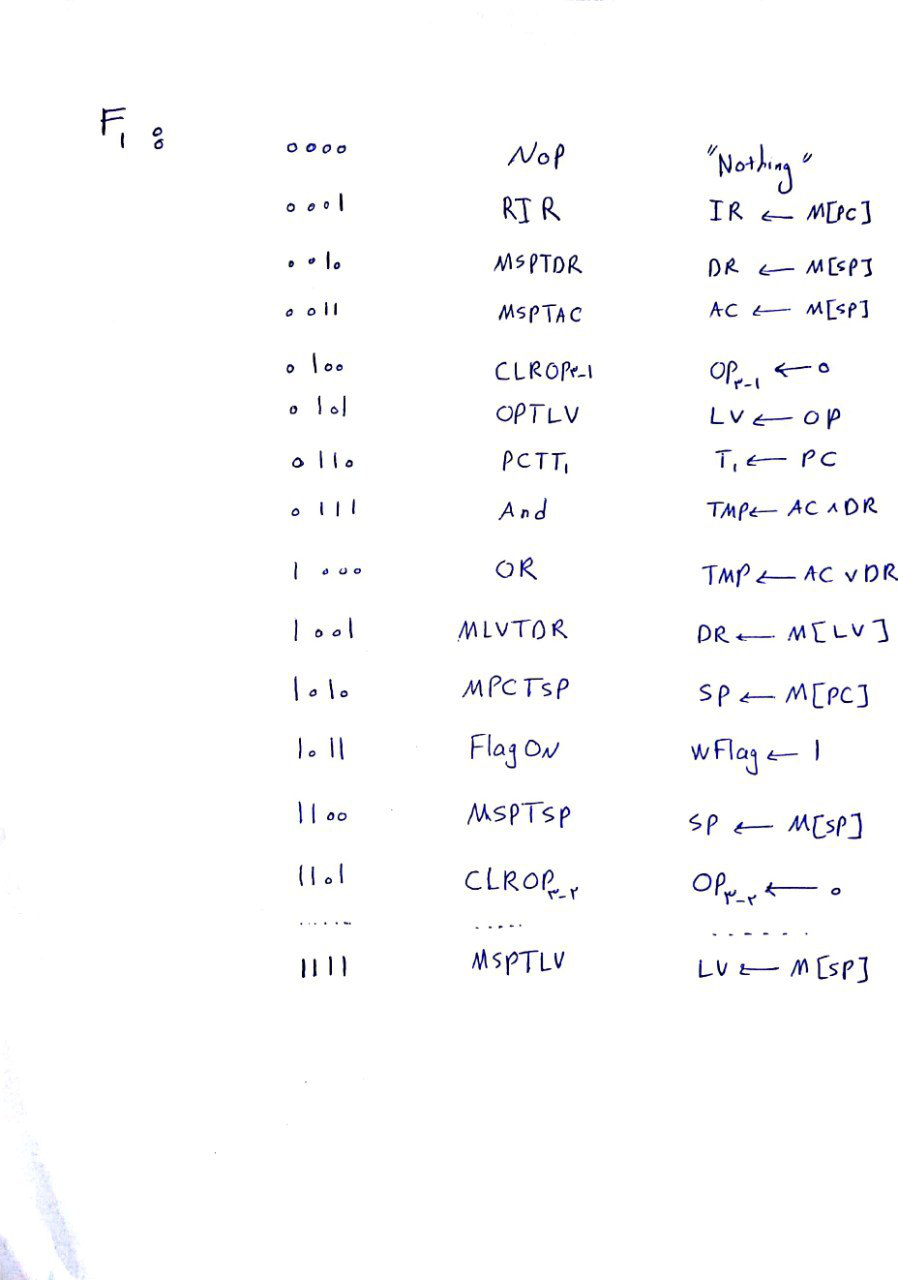
فایل مربوط به اکسل پیوست نیز شده است.

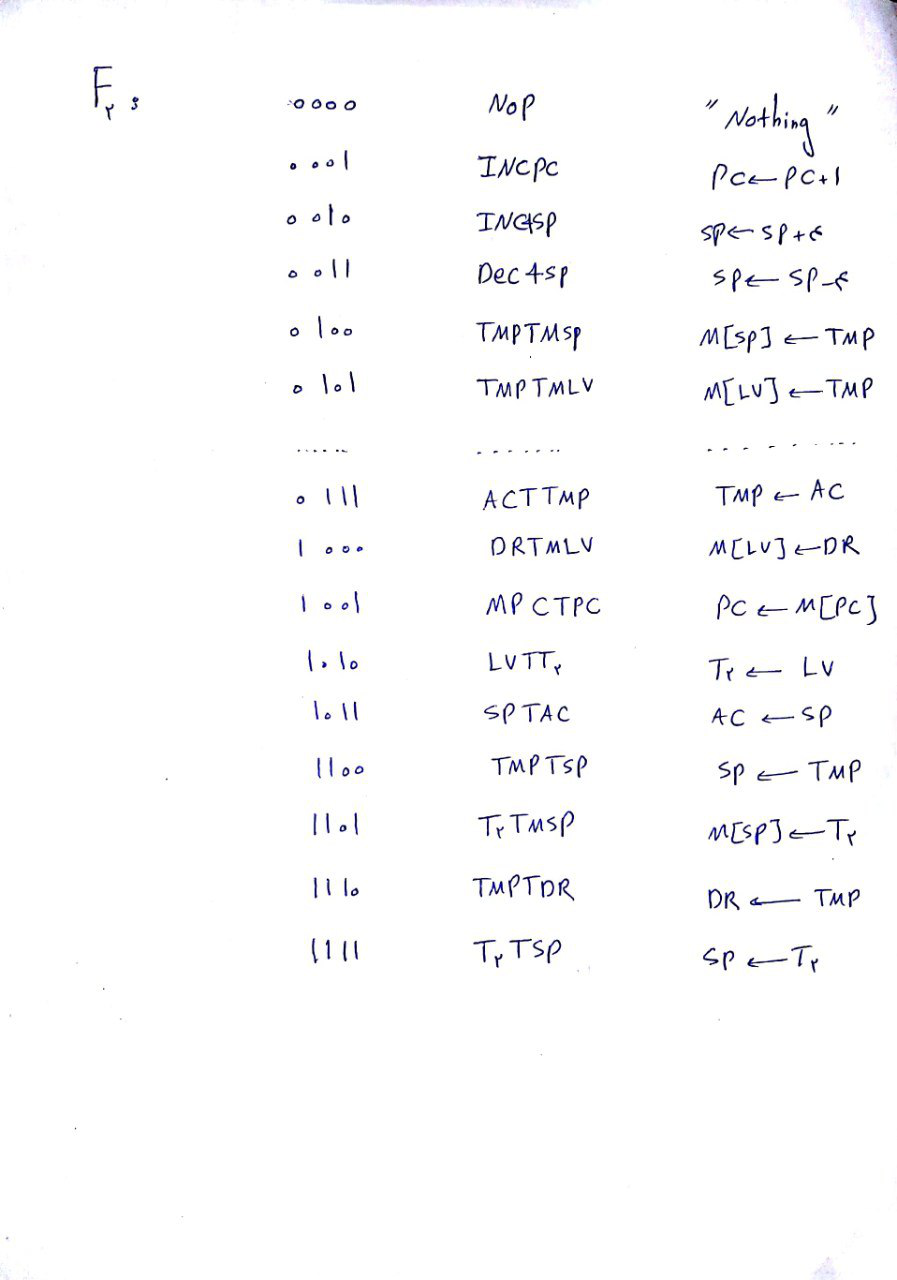
قبل از نوشتن میکرو اینستراکشن ها طبیعتا نیاز بود برای هر کدام از فیلد های F1وf2وf3میکرو اپریشن ها را تعینن کنیم و برای هر کدام نامی بگذاریم .

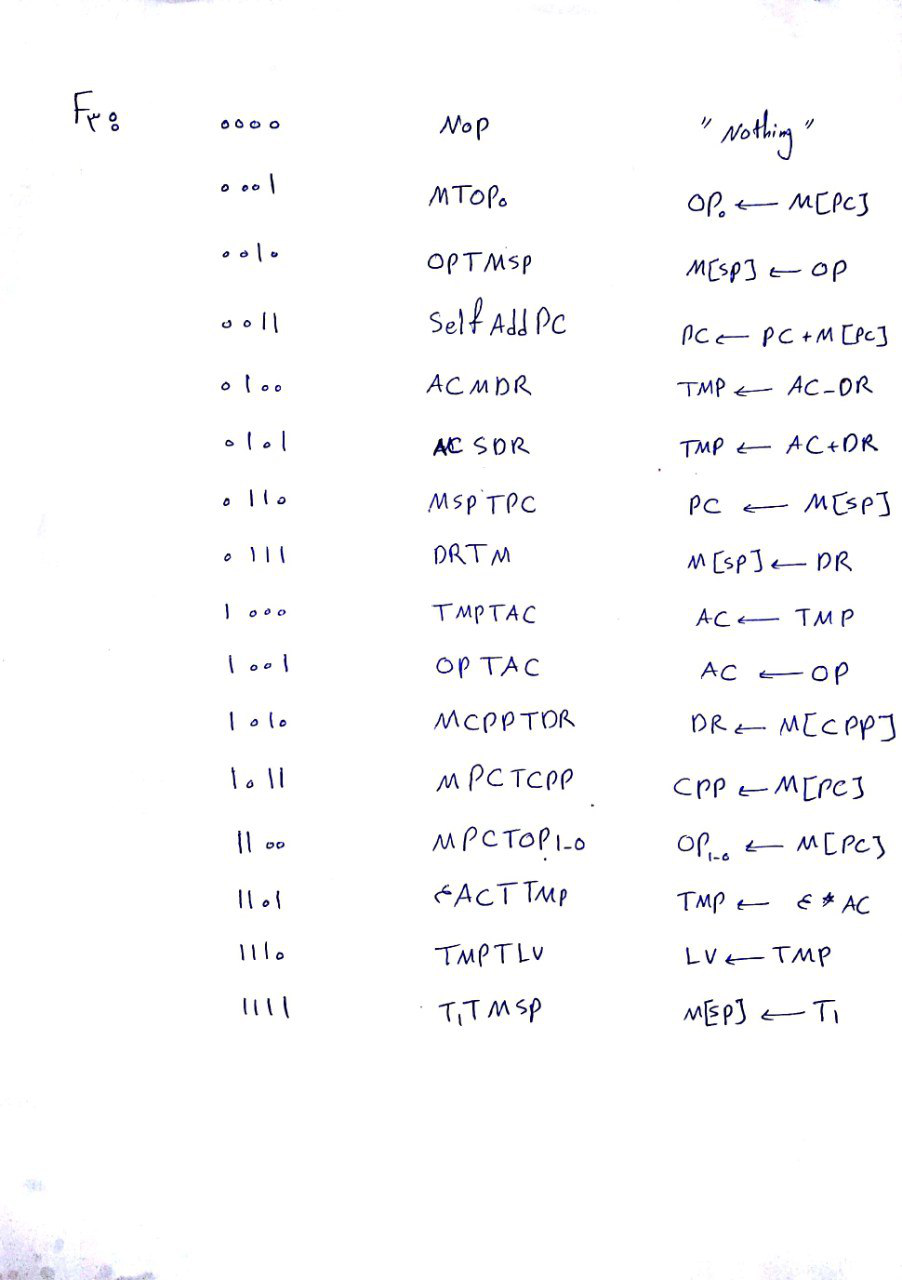
در زیر تقسیم بندی تمام میکرو اپریشن های استفاده شده را در 3 فیلد جدا میبینید .

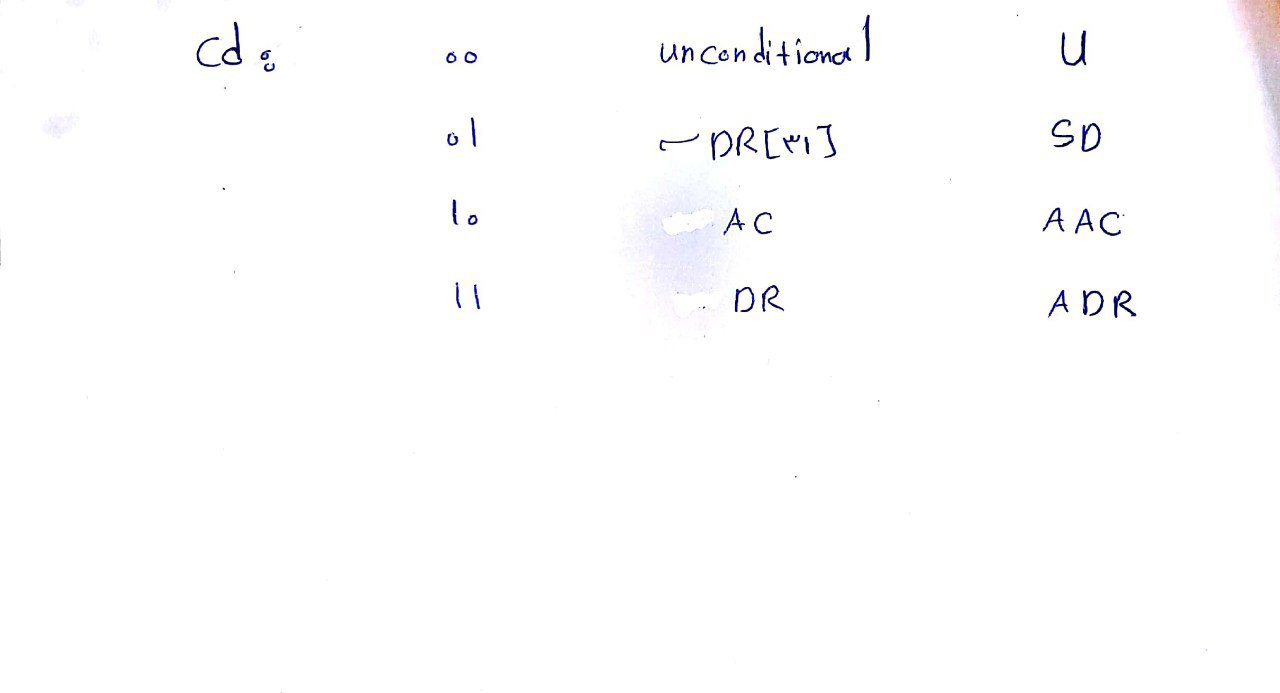
در این تقسیم بندی سعی شده است از حداکثر موازات بهره گرفته شود.

علاوه بر این چون هر کدام از فیلد ها f1,f2,f3 4 بیتی میباشند ینی جمعا 48 دستور را میتوانند ساپورت کنند که در تقسیم بندی بنده 46 جایگاه پر شده است.









لازم است وضیحاتی راجع به دوبیت CDبیان کنم.

Branch بعدی بسته به شراطی انجام میشود که cdاعلام میدارد.

در طراحی من:

مورد اول یعنی بدون هیچ شرطی branchانجام بشود.

مورد دوم یعنی اگه بیت 31 ام DRصفر بود branchانجام شود

مورد سوم یعنی اگر ACغیر صفر بود برنچ انجام شود

مورد چهارم یعنی اگر DRغیر صفر بود برنچ انجام شود.

دوبیتBRهم حالات jumpوmapوcallوretرا مشخص میکنند

البته در اینجا چون از callوretدر رام استفاده نشده است از دو حالت دو بیت brاستفاده ایی نمیشود اما برای نبستن امکان اضافه کردن حالات جدید در نظر گرفته شده اند

تمامی دستورات مورد نیاز در اکسل وارد شده و رام اماده شده ی خود را در سمت راست آن میبینیم(در شکل زیر)

فایل اکسل با کیفیت بهتر به شکل پی دی اف در فایل های پروژه پیوست شده است.







در ادامه هم به ساختن سخت افزار مورد نیاز پرداخته شده است.

نکته ی مهم و قابل توضیح نحوه ی هندل دستور wide میباشد.

با امدن این دستور فلیپ فلاپی به نامflagیک میشود.

رجیستر هایی که قرار است oprandدوبایتی دریافت کنند از flag ورودی میگیرند و در صورت یک بودن آن 2 بایت از خروجی مموری لود میکنند و در غیر این صورت یک بایت لود میکنند.

علاوه بر این پس از این اتفاق (یعنی بعد از لود شدن دوبایتی)مقدار flagصفر میشود و pcیکی زیاد میشود.

این مورد را در سخت افزارپیاده سازی کرده ام.

**توضیحاتی در مورد قسمت های مختلف سخت افزار**:

نکته ی اول اینکه دیتاپس من نیاز به کمی تغییر داشت چرا که در تمرین قبل نیاز به این نبود که از رجیستر sp و pcروی باس اصلی خروجی گرفته شود. اما در اینجا نیاز به این قضیه در دستور envoke ایجاد شد و نتیجتا دیتا پس کمی میبایست تغییر میکرد.

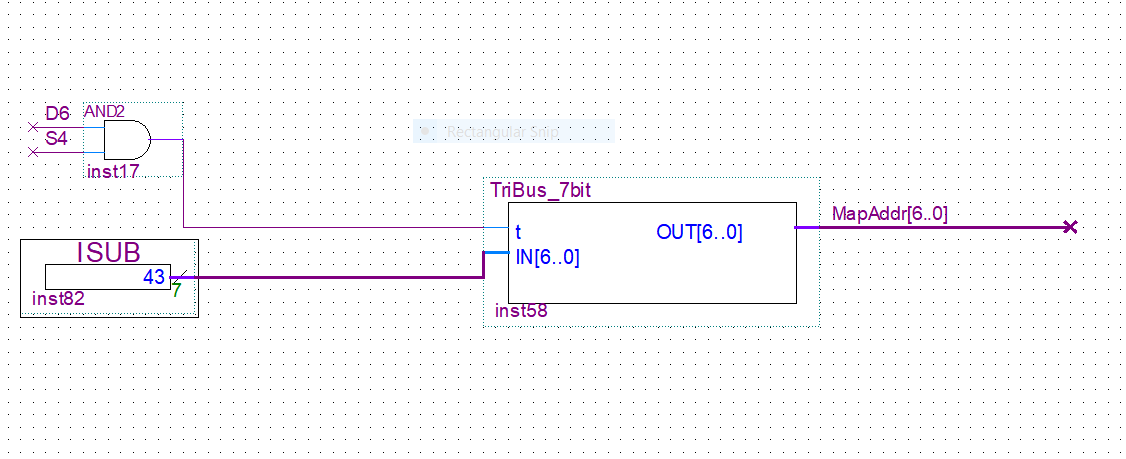
در مورد کنترل یونیت لازم است مطالبی گفته شود.

یک اینکه در کنترل یونیت از یک رام و یک رجیستر ادرس برای رام و دو مالتی پلکسر برای انتخاب ادرس بعدی رام و واحد mapو واحد incrementorو واحد genSignalاستفاده شده است .

واحد mapدستورات را از رجیستر IRگرفته و به عنوان خروجی ادرس نقطه ی شروع دستور مورد نظر در رام را تولید میکند.

در ساخت این واحد از CONSTANTهای کوارتیوس ((LMPاستفاده شده است که به را حتی قابلیت تغییر ادرس شروع دستور دلخواه را به ما میدهد.

شیوه ی کار به این صورت است که مانند دیکود کردن هر دستور در تمرین قبل اگر بیت مربوط به یک دستور یک شد کنترل بافری که مربوط به این دستور هست یک میشود در نتیجه مقدار کانستنت مربوط به این دستور که همان نقطه ی شروع دستور مورد نظر در رام میباشد بر روی باس خروجی میرود.(مثلا در شکل زیر اگر andمورد نظر که نشان دهنده ی امدن دستور isubاست یک شود مقدار کانستن "43"به باس خروجی متصل میشود )



در نتیجه خروجی واحد mapperادرس شروع اولین اینستراکشن از دستور مورد نظر در رام خواهد بود.

در مورد واحد gen signalباید کفت که این واخد وظیفه ی تولید سیگنال های ترجمه شده برای دیتاپس را خواهد داشت .

در واقع خروجی هر ورد "کنترل رام" یا به عبارتی (f1,f2,f3)به این واحد داده میشود تا سیگنال های لود و...رجیستر ها یا مالتی پلکسر ها را در دیتاپس تولید کند.

در این واحد 3 دیکودر قرار داده شده است که نام میکرو اپریشن مربوط به خودشان را تولید میکنند .

و هر سیگنال تولید شده بسته به این که با میکرو اپریشن ها اشتراک داشته باشد یا نه حاصل orبعضی از ان ها خواهد بود که به طور کامل همگی در نظر گرفته شده اند.

واحد دیگری به نام input logic نیز وجود دارد که در آن انتخاب وردی ادرس بعدی رام انجام میشود.

کنترل زمان هایی که نیاز به برنچ میباشد و انتخاب مالتی پلکسری که ادرس بعدی را مشخص میکند در این واحد با گیت های منطقی انجام شده است.

فایل های مربوط به تمامی قسمت ها در فایل فرستاده شده پیوست شده است.

در این تمرین نیز مانند تمرین قبلی برای کنترل مموری از یک فلیپ فلاپ استفاده شده است که دقیقا مانند تمرین قبل پیاده سازی شده است و در مواقعی باید منتظر خروجی مموری بماند این کار انجام میشود.