به نام خدا

گزارش کار آزمایش دهم آزمایشگاه معماری کامپیوتر

عرفانه خانمحمدی-یاسمن گودرزی

9931100-9931067

به طور کلی اگر بخواهیم به روند کلی طراحی کامپیوتر پایه بپردازیم باید بگوییم: برای طراحی کامپیوتر پایه به طراحی data path و control unit نیاز داریم به طوری که در طراحی مسیر جریان لازم است که ارتباط میان رجیستر ها و bus را مشخص کنیم و توجه داشته باشیم که توسط طراحی مدار و استفاده از ،mux در هر لحظه تنها یک رجیستر به bus دسترسی پیدا کند. همچنین به حافظه نیاز داریم تا پس از اجرای دستورات مشخص شده بر اطلاعات موجود در حافظه؛ نتیجه را در حافظه ذخیره کند.

با توجه به کد داده شده در فایل داده شده، حافظه ای با ارتفاع 256 که هر کلمه آن 16 بیت نیاز داریم، یعنی 256 ردیف با ظرفیت هر ردیف، 16 بیت. پس تمام رجیستر هایی که آدرس نگه می دارند 8 بیتی و رجیستر های نگه دارنده داده 16 بیتی هستند . با توجه به کد داده شده، این مقادیر به صورت genericدر entity تعریف شده اند. همچنین نکته قابل توجه درباره رجیستر های مورد استفاده این است که به طور کلی، رجیستر ها شامل دو نوع عام منظوره (از جمله رجیستر AC) و خاص منظوره هستند، اما در این کد صرفا از رجیستر های خاص منظوره استفاده شده است.

اگر بخواهیم به توضیح رجیستر ها بپردازیم:

PC برای ذخیره کردن آدرس دستوری که در نوبت بعدی اجرا است.

IRبرای ذخیره کردن دستوری که نوبت اجرای آن است.

AC برای ذخیره حاصل جمع مقدار خوانده شده از حافظه و مقدار قبلی AC رجیستر MAR برای فرستادن آدرس به حافظه رجیستر MDR برای ذخیره داده برای گرفتن از یا فرستادن به حافظه.

در ادامه کد رجیستر ها و سیگنال دستور نوشتن در حافظه و همچنین حافظه با ابعاد خواسته شده به صورت یک آرایه دو بعدی تعریف شده اند.

همچنین در این کد؛ دستوراتی داریم که طبق الگوریتم "فون نیومن" عمل میکنند؛ به طوری که: مسیر داده انتقال داده بین رجیستر ها را ممکن می کند و واحد کنترل کننده با مقدار دهی به پایه های رجیستر ها مثل ... , reset, Load و همچین کنترل استفاده رجیستر ها از مسیر داده ، الگوریتم فون نیومن را اجرا می کند.

اگر برای پیاده سازی از این نوع الگوریتم استفاده شود: حالت reset برای تنظیمات اولیه ، fetch برای گرفتن دستور ، decode برای فهمیدن دستور و انواع execute برای اجرای دستورات. در بین دستورات ، دستور jump, load, add را می توان در یک کالک اجرا کرد ، اما برای اجرای دستورات store و jneg به دو کالک نیاز است ، به همین دلیل برای این 2 دستور ، 2 حالت در نظر گرفته شده است.

توضیح process های موجود در کد:

1) برای خواندن و نوشتن از حافظه پیاده سازی شده است . به طوری که با آمدن لبه بالا رونده کلاک اگر سیگنال write_memory منطق یک داشته باشد ، مقدار موجود در ثبات AC در خانه ای از حافظه که آدرس آن در رجیستر MAR قرار دارد ، ذخیره می شود و اگر این سیگنال منطق صفر داشته باشد ، مقدار موجود در خانه ای از حافظه که آدرس آن در MAR قرار دارد ، در رجیستر ریخته می شود .
2) نحوه تغییر حالت سیستم توصیف شده است . در این process ابتدا سیگنال ورودی reset چک می شود ، اگر 1 باشد ، سیستم به حالت اولیه می رود و اگر صفر باشد ، حالت بعدی سیستم بر اساس حالت فعلی و با در نظر گرفتن الگوریتم فون نیومن تعیین می شود.

در توضیح قالب بندی انجام شده نیز میتوان گفت:

از 16 بیت، 8 بیت مختص opcode است تا نوع دستور را مشخص کند و 8 بیت کم ارزش به منظور ذخیره ادرس خانه ای از حافظه استفاده می شود که در آن خانه از حافظه، عملوند مورد نیاز ذخیره شده است.

از جمله دستوراتی که اجرا میشود:

1) عملیات add مقداری که از حافظه در ثبات MDR ذخیره شده است با مقدار موجود در ثبات AC جمع و حاصل در خود ثبات AC قرار می گیرد و سیستم به حالت fetch می رود.
2) عملیات store ابتدا در کالک اول سیگنال write_memory یک می شود و در ثبات MAR آدرس خانه مورد نظر برای ذخیره و در ثبات MDR مقدار مورد نظر برای ذخیره در حافظه قرار می گیرد و در کالک دوم این مقدار در حافظه ذخیره شده و سیستم به حالت fetch می رود.
3) عملیات load مقدار موجود در ثبات MDR به ثبات AC ریخته می شود که این مقدار از خانه ای از حافظه که آدرس آن در 8 بیت سمت راست دستور المعل نوشته شده است به ثبات MDR ریخته شده است و در انتها سیستم به حالت fetch می رود.
4) عملیات jump آدرس موجود در 8 بیت سمت راست دستورالعمل در ثبات PC ریخته می شود و سیستم به حالت fetch می رود.
5) عملیات jneg در کالک اول مقدار موجود در ثبات AC بررسی می شود اگر بیشتر از صفر باشد ، در کالک بعدی کاری انجام نخواهد شد) تعریف سیگنال ثبات AC به صورت singed انجام شده است (و اگر این مقدار کمتر از صفر باشد ، در کالک آدرس موجود در 8 بیت سمت راست دستور المعل در ثبات PC ریخته شده و سیستم به حالت fetch می رود.

در اخر نیز؛ باید اشاره کنیم به مقدار دهی های انجام شده به طوری که: سیستم فقط در حالت store_execute مقدار سیگنال write_memory را یک می کند و در سایر حالت مقدار صفر را برای این سیگنال در نظر می گیرد.