به نام خدا

آز معماری – دکتر سربازی آزاد

دانشکده مهندسی کامپیوتر

دانشگاه صنعتی شریف

گزارش آزمایش ۳

اعضای گروه:

میترا قلی‌پور-۴۰۱۱۰۶۳۶۳

نیکا قادری-۴۰۱۱۰۶۳۲۸

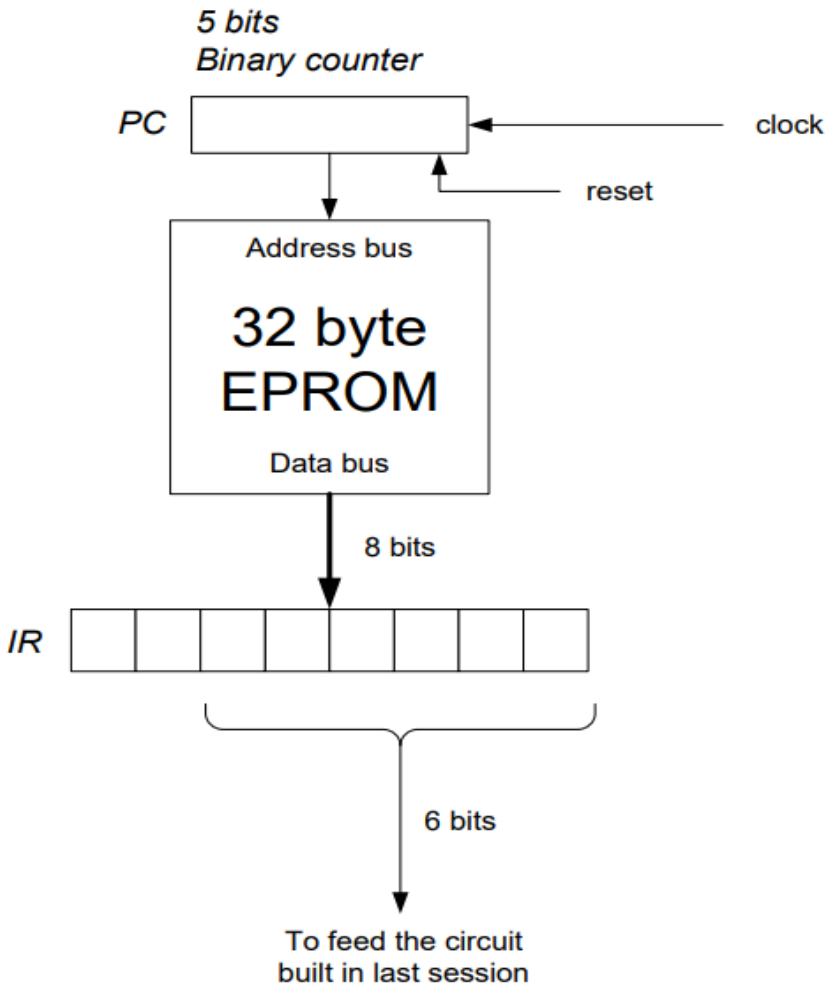
ملیکا علیزاده-۴۰۱۱۰۶۲۵۵

هدف و نتیجه مورد انتظار

در این آزمایش به متصل کردن بخش محاسبات طراحی شده در بخش قبل به یک حافظه برنامه‌پذیر می‌پردازیم و هدف از این آزمایش آشنایی با نحوه واکشی دستورات در پردازنده است تا در نهایت بتوانیم عدد دهم دنباله فیبوناچی را محاسبه کنیم.

شرح آزمایش

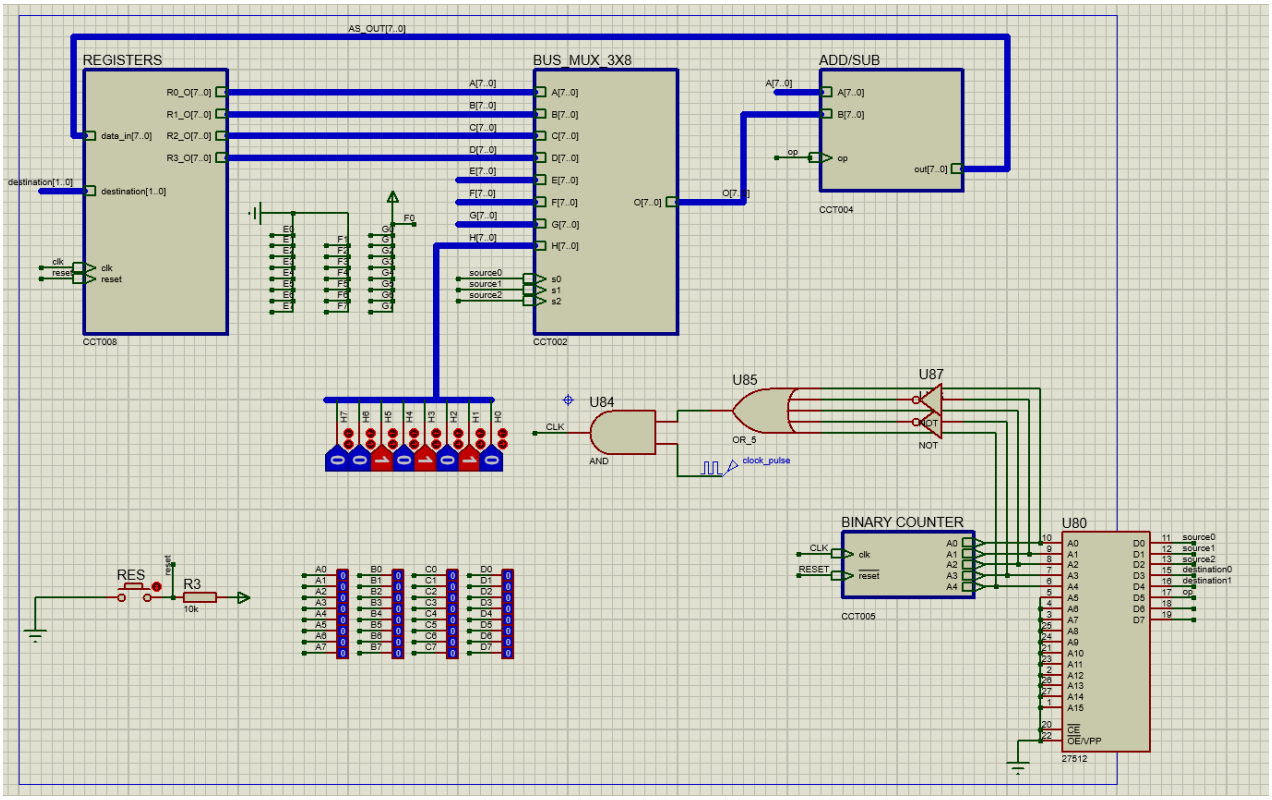
در این آزمایش، فرمان‌های مورد نیاز برای کنترل مدار آزمایش ششم از برنامه‌ای که در حافظه EPROM ذخیره شده است، گرفته می‌شود. این فرمان‌ها به ترتیب توسط یک شمارنده برنامه (PC) آدرس‌دهی شده و پس از واکشی از حافظه، اجرا می‌شوند. برای این منظور، باید مدارهای لازم به مدار آزمایش ششم افزوده شوند. شکل ۱ بلوک دیاگرام سیستم را نشان می‌دهد. پس از افزودن بخش‌های مورد نیاز به مدار آزمایش ششم، برنامه‌ای برای تولید ۱۰ جمله‌ی اول سری فیبوناچی را کدنویسی کرده و در حافظه EPROMذخیره می‌کنیم، سپس با استفاده از معماری پیاده‌سازی شده آن را اجرا می‌کنیم.



شکل ۱:بلوک دیاگرام سیستم

بخش اول: طراحی مدار

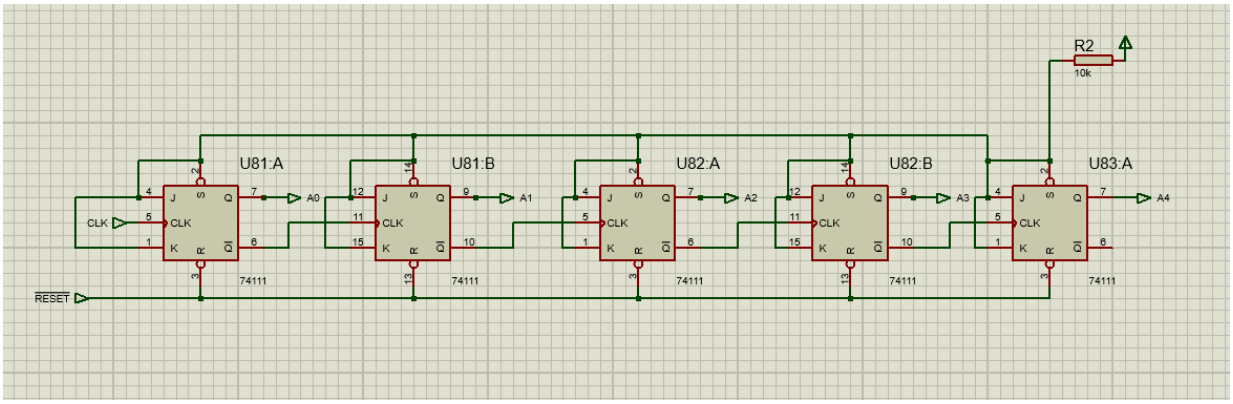
برای طراحی در این آزمایش از همان مدار آزمایش ششم استفاده کردیم و فقط نیاز بود که دو بخش جدید شمارنده ۵بیتی به عنوان pc و استفاده از تراشه ۲۷۵۱۲ به عنوان EPROM را اضافه کنیم و در نهایت برنامه مورد نظر خود را به آن بدهیم. شمای کلی مدار در شکل ۲ آمده است.



شکل ۲: مدار اصلی

شمارنده ۵ بیتی

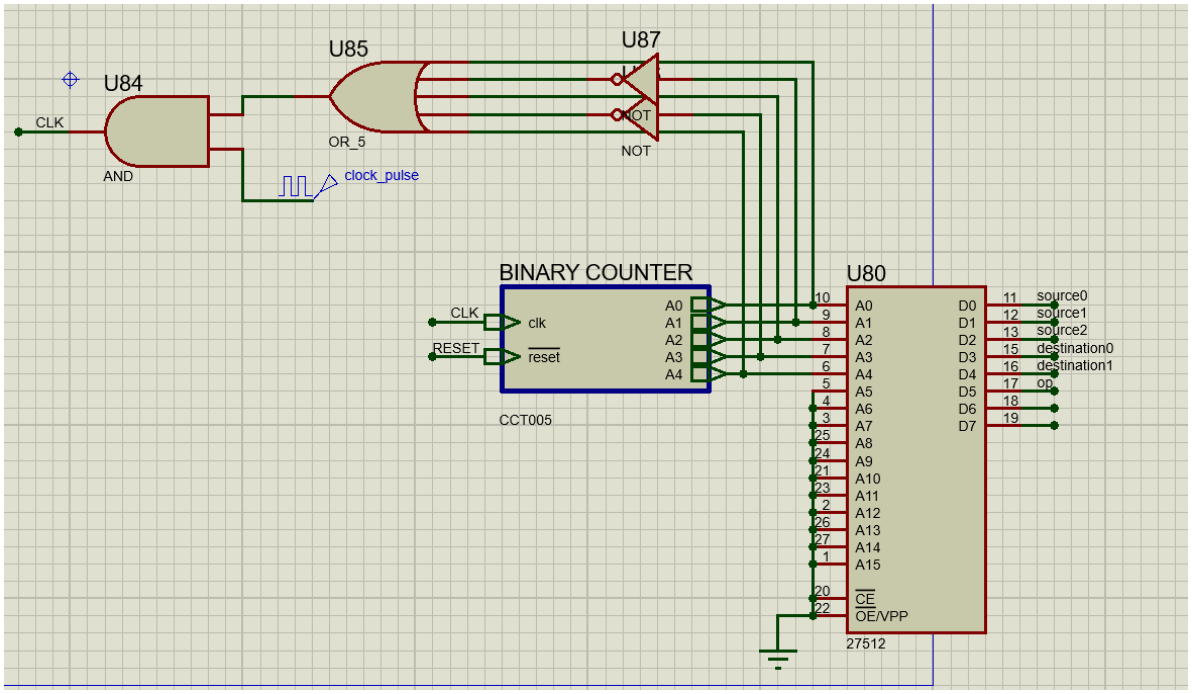
برای ساخت یک شمارنده ۵بیتی آسنکرون باید از پنج JK FF استفاده کنیم. از آنجایی که این شمارنده دهدهی است زمانی که شمارش به ده برسد مدار متوقف می‌شود. این شمارنده چون آسنکرون است در هر لبه پالس کلاک به سمت بالا شمارش می‌کند و شمارش از 0000 شروع و تا 1001 می‌رود. در نهایت با توجه به جدول درستی و روش ساخت این شمارنده که در درس مدارمنطقی با آن آشنا شدیم، مدار شکل ۳ را می‌سازیم. همچنین برای این بخش ورودی‌های کلاک و ریست را نیاز داریم.



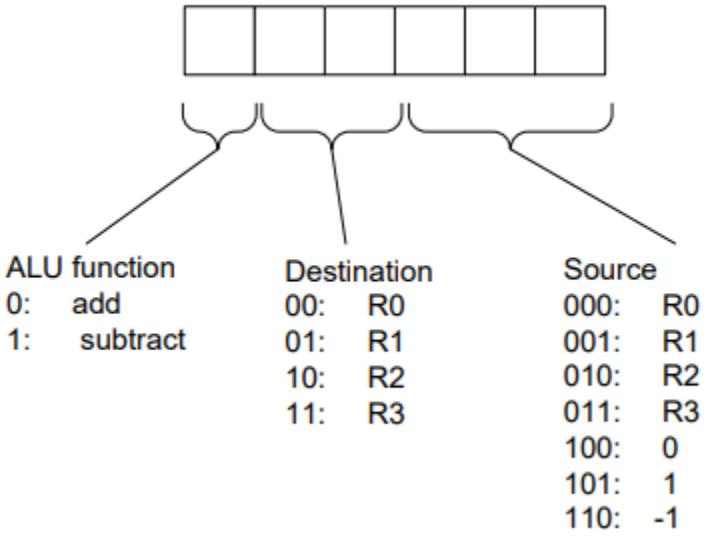
شکل ۳: شمارنده ۵بیتی

اتصال به EPROM

۵ خروجی بخش شمارنده را به ۵ ورودی اول EPROM و بقیه ورودی‌های آن را به صفر وصل می‌کنیم. خروجی‌های این حافظه به عنوان خطوط کنترلی واحد محاسبات که در آزمایش قبل ساخته بودیم استفاده می‌کنیم. شکل ۵ قالب این دستورات را نشان می‌دهد. همچنین برای راحتی کار و اینکه نیازی نباشد تا برای هر دستور دکمه کلاک را فشار دهیم، از این ساز و کار استفاده می‌کنیم که با استفاده از یک گیت or معکوس عدد ۱۰(شماره برنامه‌ای که دیگر نباید اجرا شود) را با سیگنال کلاک and می‌کنیم تا پس از آن کلاک به سیستم اعمال نشود و عملا سیستم در حالت idle قرار بگیرد. شکل ۴ نشان‌دهنده این بخش مدار است.



شکل ۴: مدار حافظه



شکل ۵: قالب دستورات

بخش دوم: نوشتن برنامه

در این بخش با توجه به قالب دستورات شکل ۵ کد تولید ده جمله اول فیبوناچی را به این صورت می‌نویسیم و سپس آن را به زبان ماشین ترجمه می‌کنیم و در نهایت آن را به صورت هگزادسیمال بدست می‌آوریم. قطعه کد زیر ده جمله اول سری فیبوناچی را در ثبات‌های R0 و R1 تولید می‌کند.

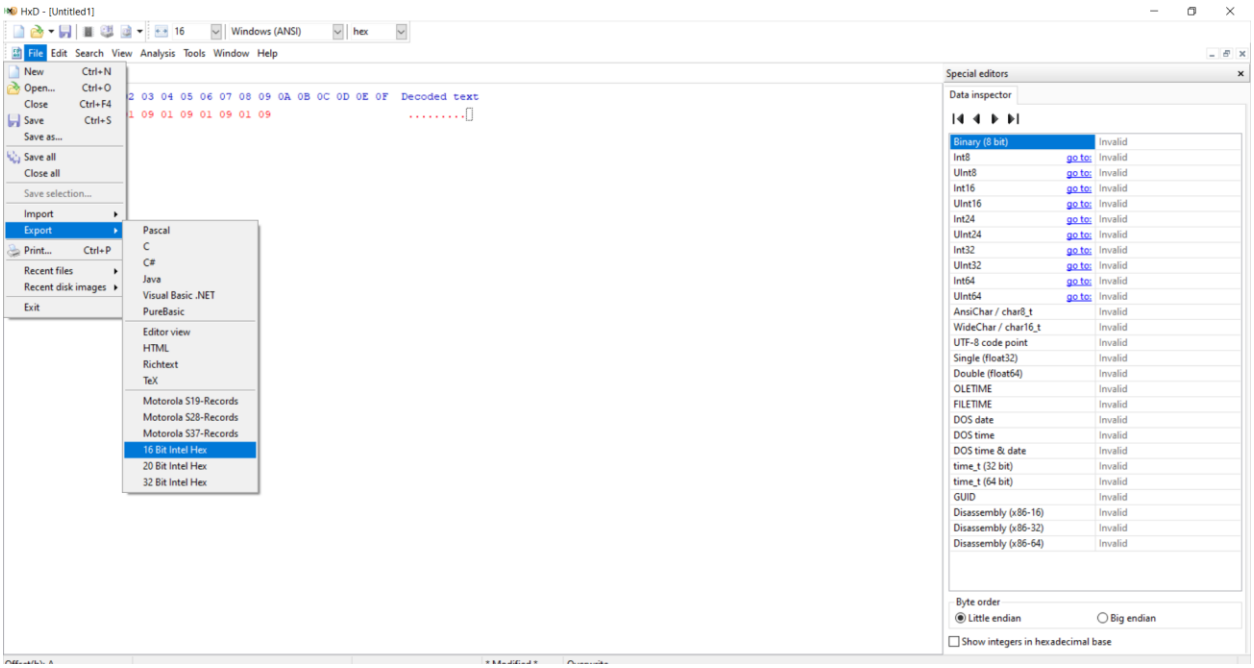
Address	Code	Instruction	Comment	Comment
00000	00 1 00 000=20	Sub R0, R0	جمله اول در R0	0→R0
00001	00 0 01 101=0D	Add R1, 1	جمله دوم در R1	1→R1
00010	00 0 00 001=01	Add R0, R1	جمله سوم در R0	1→R0
00011	00 0 01 001=09	Add R1, R0	جمله چهارم در R1	2→R1
00100	00 0 00 001=01	Add R0, R1	جمله پنجم در R0	3→R0
00101	00 0 01 001=09	Add R1, R0	جمله ششم در R1	5→R1
00110	00 0 00 001=01	Add R0, R1	جمله هفتم در R0	8→R0
00111	00 0 01 001=09	Add R1, R0	جمله هشتم در R1	13→R1
01000	00 0 00 001=01	Add R0, R1	جمله نهم در R0	21→R0
01001	00 0 01 001=09	Add R1, R0	جمله دهم در R1	34→R1

حال اگر در برنامه HxD ماشین کد دستورات را به صورت هگز وارد کنیم و سپس آن را به صورت یک فایل intel hex ۱۶ بیتی خروجی می‌گیریم. این فایل یک نوع فایل متنی است که در ابتدای هر خط به فرمت خاصی عبارات نوشته می‌شوند و قابل استفاده برای شبیه‌سازی EPROM در پروتئوس خواهد بود. این فایل دارای محتویات زیر است که کد هگز ما در آن مشخص است:

:0A000000200D0109010901090109A1

:00000001FF

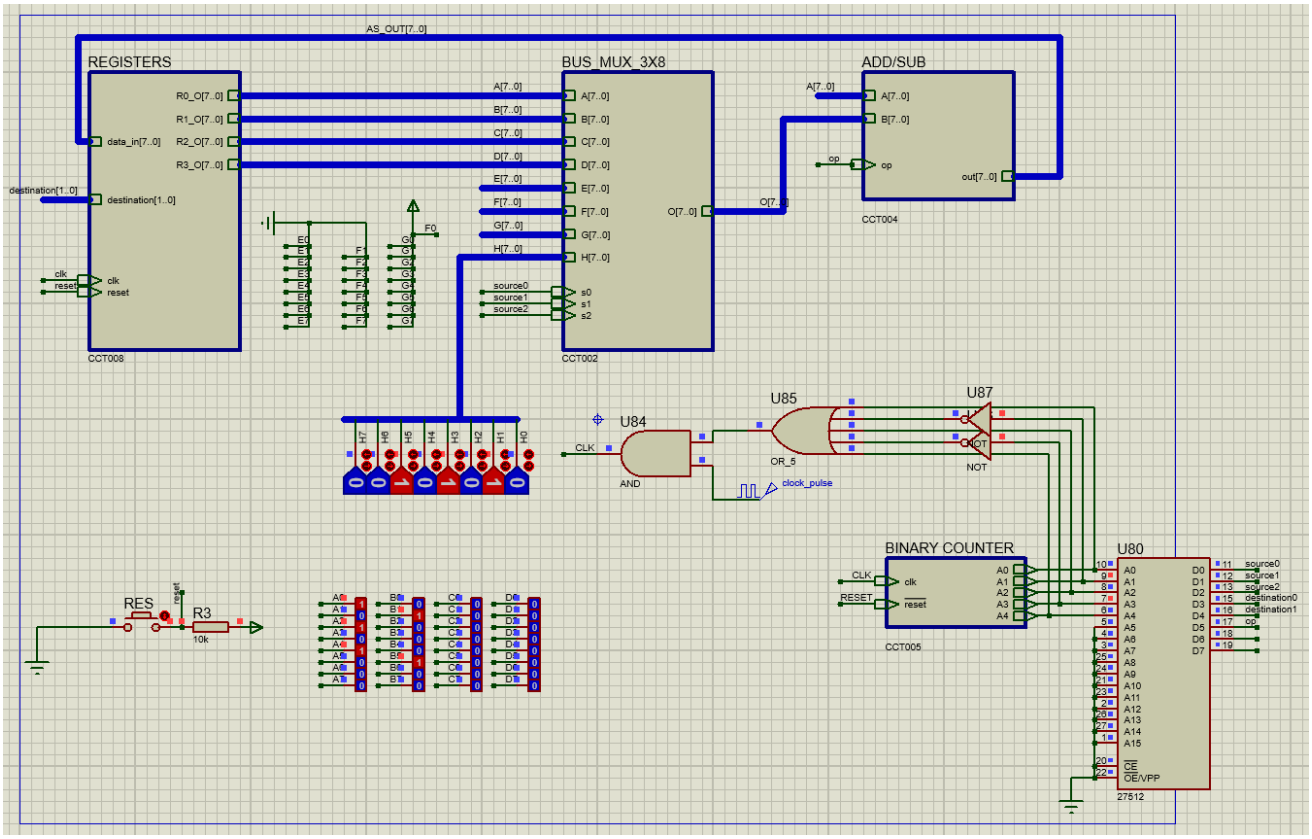
همچنین در شکل ۶ محیط کار برنامه HxD آمده است.



شکل ۶:محیط کار برنامه HxD

بخش سوم: تست

نتیجه اجرای برنامه در شکل ۷ آمده است و همانطور که مشخص است جمله دهم سری فیبوناچی یعنی عدد ۳۴ در رجیستر ۱ وجود دارد و رجیستر ۰ نیز عدد جمله نهم یعنی ۲۱ را دارد.



شکل ۷خروجی مدار