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

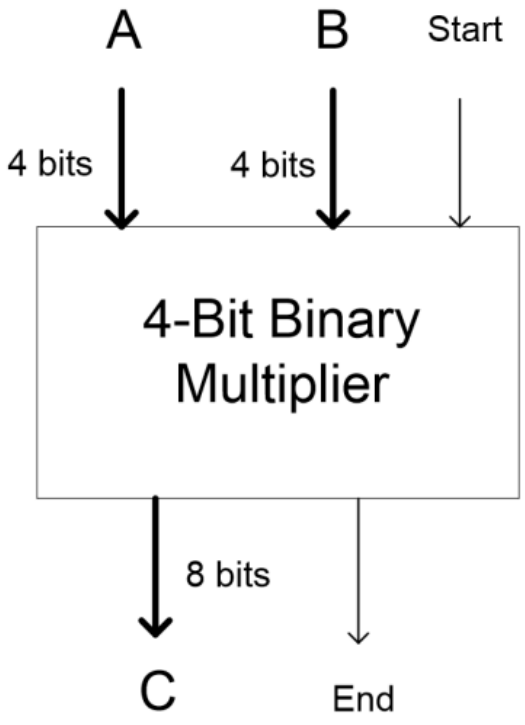
ملیکا علیزاده-۴۰۱۱۰۶۲۵۵

هدف و نتیجه مورد انتظار

در این آزمایش به طراحی و ساخت مدار ضرب‌کننده ۴بیتی ممیز ثابت می‌پردازیم. این ضرب‌کننده دو عدد ۴بیتی و یک سیگنال شروع به عنوان ورودی دارد و خروجی آن یک عدد ۸بیتی و یک سیگنال پایان است که پس از فعال شدن سیگنال شروع، حاصل ضرب تا حداکثر ۷ کلاک بدست می‌آید و سیگنال پایان فعال می‌شود.

شرح آزمایش

با فعال شدن سیگنالStart ، ضرب‌کننده شروع به کار کرده و حاصل‌ضرب دو عدد ورودی چهار بیتی A و B را به روش shift & add محاسبه می‌کند. پس از اتمام عملیات، حاصل‌ضرب ۸ بیتی را روی خط C قرار داده و با فعال کردن سیگنال End پایان عملیات را اعلام می‌کند.

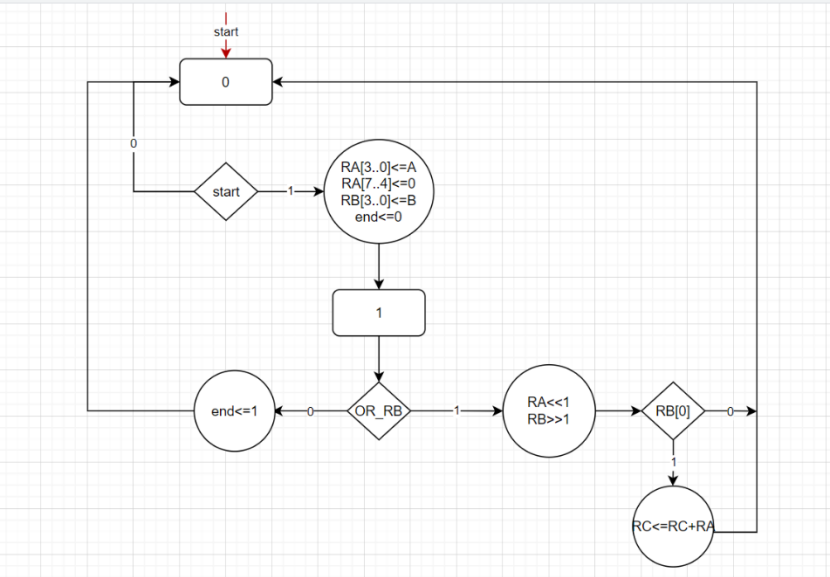


بلوک ضرب‌کننده چهار بیتی

بخش اول: طراحی کلی مدار ضرب‌کننده ۴بیتی و طراحی ASM

ابتدا باید نیازهای مدار را بررسی کنیم. دو عدد ۴ بیتی(۲بیت برای قسمت صحیح و ۲بیت برای اعشار در نظر گرفته شده است.) به عنوان ورودی که بدون علامت هستند و یک خروجی ۸بیتی(۴بیت برای قسمت صحیح و ۴بیت برای اعشار در نظر گرفته شده است.) داریم. حال چون ضرب‌کننده با الگوریتم shift & add ساخته شده است باید از شیفت رجیستر استفاده کنیم تا بتوانیم عوامل ضرب را شیفت دهیم که این رجیسترها دارای قابلیت بارگذاری موازی و شیفت هستند. همچنین یک سیگنال ورودی(start) برای اعلام شروع ضرب داریم و یک سیگنال(end) نیز برای اعلام اتمام عملیات ضرب نیاز است. سیگنال شروع به صورت push button گذاشته شده است پس فقط در لحظه کلیک کردن یک می‌شود و پس از آن تا زمانی که دوباره کلیک شود صفر خواهد بود.

حال برای اینکه طبق چارت زیر مدار را پیاده‌سازی کنیم باید اجزای آن را در نظر بگیریم:



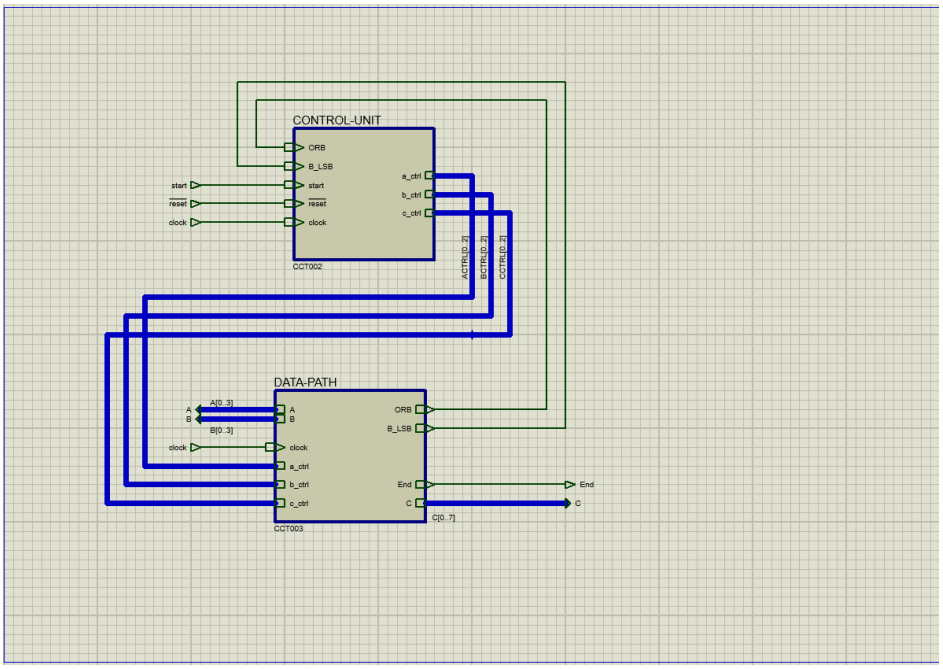
چارت طراحی شده

طبق این چارت زمانی که سیگنال شروع یک شود، مقادیر اولیه ورودی‌ها در رجیسترها لود می‌شود پس باید ساخت مدار از شیفت رجیستر ۴ بیتی استفاده کنیم و چون عامل اول باید بدون از دست دادن اطلاعات قابلیت شیفت داشته باشد باید از شیفت رجیستر ۸ بیتی استفاده کنیم پس ورودی اول را در ۴ بیت کم ارزش آن لود می‌کنیم و باقی بیت‌ها را صفر می‌کنیم. همچنین باید سیگنال پایان را صفر کنیم. نتیجه نهایی را نیز در یک شیفت رجیستر به نام C قرار میدهیم و همواره در این رجیستر جمع مقدار آن با مقدار شیفت داده شده رجیستر اول قرار می‌گیرد. در مرحله بعد چند اتفاق به صورت همزمان انجام می‌َشود. اگر تمام بیت‌های عامل دوم صفر شود یعنی عملیات به پایان رسیده و مقدار سیگنال پایان یک می‌شود و نتیجه حاصل می‌شود. و اگر عملیات به

پایان نرسید عامل اول به چپ و عامل دوم به راست شیفت می‌خورد. و در صورت یک بودن بیت آخر عامل دوم باید مقدار نهایی را با مقدار شیفت داده شده رجیستر اول جمع کنیم.

بخش دوم: ساخت مدار

برای ساخت مدار، ما آن را به دو بخش اصلی تقسیم کردیم یعنی data path و control unit. در بخش data path بخش‌های اصلی مدار قرار دارد و در بخش control unit سیگنال‌های کنترلی بخش اصلی وجود دارد که در نهایت به data path می‌روند.



درون ضرب‌کننده ۴بیتی

مسیرداده

برای ساخت این بخش باید از تراشه‌های زیر استفاده کنیم:

۷۴۱۹۴ که یک شیفت رجیستر ۴بیتی است که ۴ عمل را با توجه به مقدار سلکت آن به صورت زیر انجام می‌دهد:

s1s0

00→NOP

01→shift left

10→shift right

11→load

همچنین این تراشه دارای یک ریست آسنکرون و یک بیت برای ورودی شیفت که صفر گذاشتیم.

۷۴۱۹۸ که یک شیفت رجیستر ۸ بیتی است و ۴ عمل را با توجه به مقدار سلکت آن به صورت زیر انجام می‌دهد:

s1s0

00→NOP

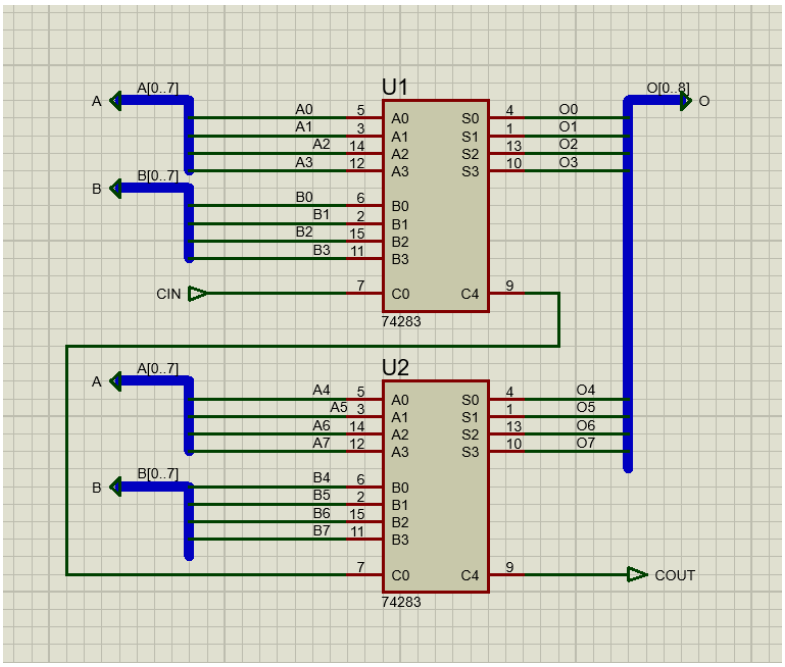
01→shift left

10→shift right

11→load

همچنین این تراشه دارای یک ریست آسنکرون و یک بیت برای ورودی شیفت که صفر گذاشتیم.

۷۴۲۸۳ که یک جمع کننده ۴ بیتی است که برای ساخت یک جمع‌کننده ۸ بیتی از دوتای آنها استفاده کردیم.

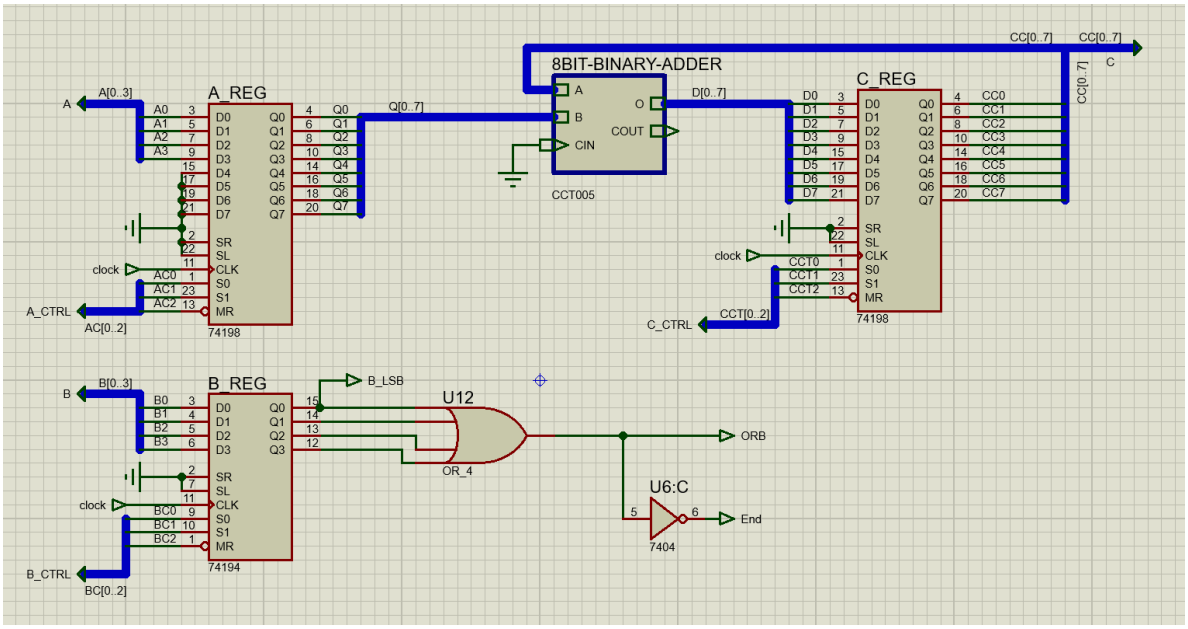


جمع‌کننده ۸بیتی

همچنین از گیت‌های not و or با ۴ ورودی استفاده شده است.

حال با استفاده از این گیت‌ها و تراشه‌ها در این بخش داریم:A یک شیفت رجیستر ۸ بیتی برای ورودی اول که ۴ بیت آن را در ۴ بیت کم‌ارزش این شیفت رجیستر قرار می‌دهیم و ۴ بیت پرارزش را صفر می‌گذاریم. B یک شیفت رجیستر ۴ بیتی برای ورودی دوم که با استفاده از بیت کم‌ارزش آن(سیگنال کنترلی برای جمع) و or تمام

بیت‌های آن(سیگنال کنترلی برای چک کردن پایان عملیات) و در نهایت با not کردن آن(سیگنال کنترلی پایان عملیات) میتوان به سیگنال‌های کنترلی دست یافت.



تصویر مسیرداده

واحد کنترل

برای این بخش از گیت‌های and و or و not و تراشه ۷۴۷۴ که یک dff است استفاده کردیم تا بتوانیم سیگنال‌های کنترلی را برای مسیر داده تولید کنیم. از dff برای نگه‌داری وضعیت فعلی مدار استفاده می‌کنیم. از سیگنال‌های کنترلی ساخته شده در این بخش به عنوان سلکت برای شیفت رجیسترهای A,B,C استفاده می‌کنیم. همچنین یک سیگنال ریست نیز برای dff نیاز است. در نهایت حالت‌های زیر بدست می‌آید:

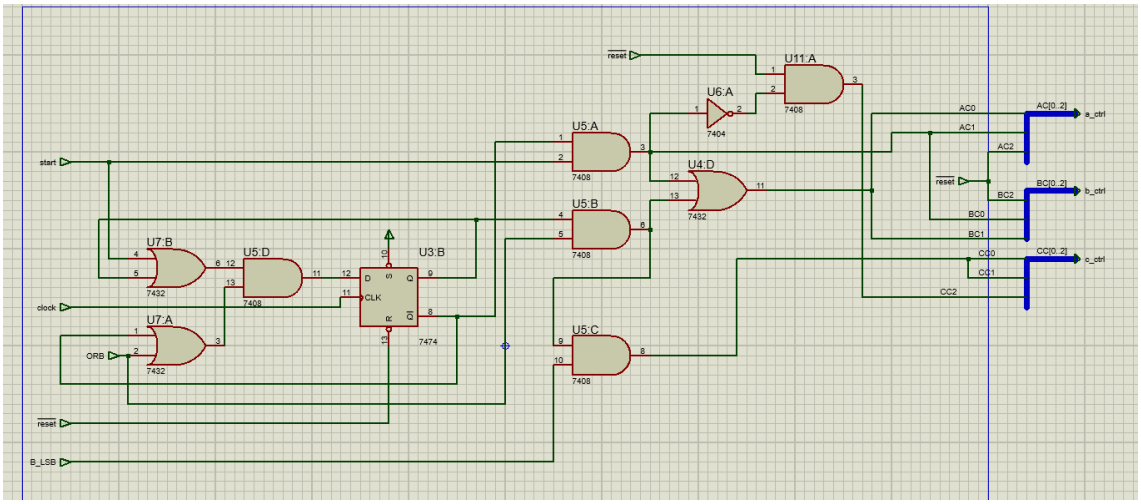
$$loadA = loadB = clearC = \bar{Q} \times S$$
$$loadC = Q \times OR(B) \times B[0]$$
$$shiftC = 0 \rightarrow never\ be\ shifted$$
$$shiftA = shiftB = Q \times OR(B)$$
$$clearA = clearB = 1 \rightarrow never\ be\ cleared$$

در نهایت با جدول حالت می‌توانیم به حالت‌های زیر برای خطوط سلکت هر رجیستر برسیم:

$$A: s0 = loadA + shiftA$$
$$A: s1 = loadA$$
$$B: s0 = loadB = As1$$
$$B: s1 = loadB + shiftB = As0$$
$$C: s0 = loadC$$

$$C: s1 = loadC$$

در نهایت با توجه به این روابط مدار بخش کنترل به این صورت ساخته می‌شود:



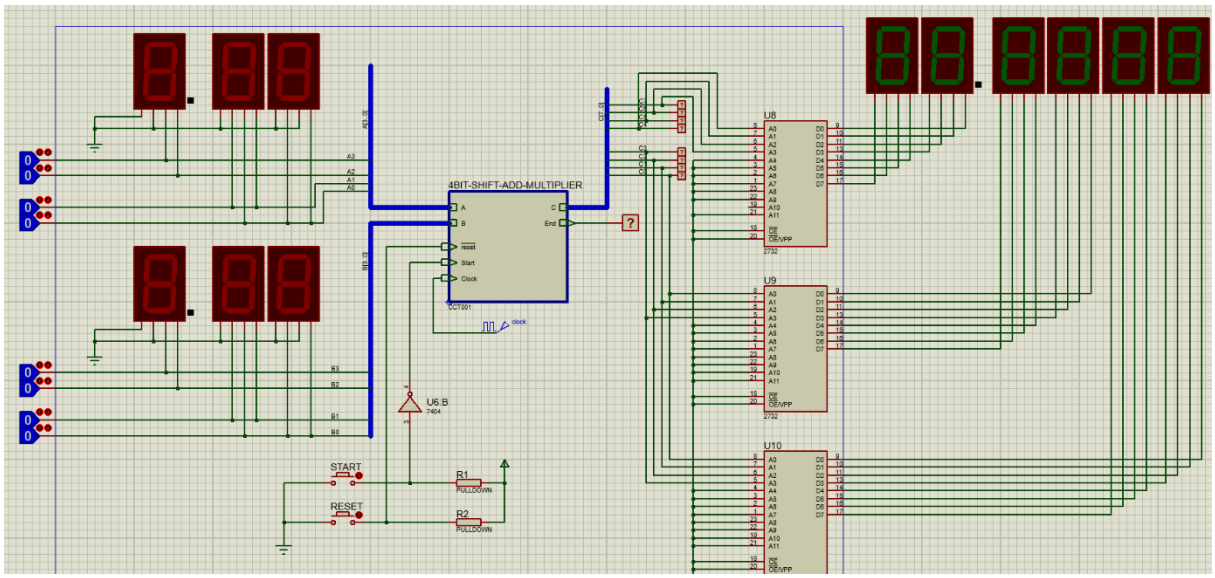
مدار واحد کنترل

مدار اصلی ضرب‌کننده

حال با استفاده از این دو بخش کنترل و مسیر داده می‌توانیم ماژول اصلی را به این صورت بسازیم. برای نمایش ورودی‌ها و خروجی‌ها از نمایشگرهای 7seg استفاده می‌کنیم. برای اعداد ورودی قسمت غیراعشاری به صورت مستقیم متصل شده اند. قسمت اعشاری نیز به این صورت وصل شده اند که اگر کم ارزش ترین بیت روشن باشد ۰.۲۵ را خواهیم داشت و اگر بیت دوم روشن باشن ۰.۵ را خواهیم داشت و اگر هر دو ۰.۷۵ را خواهیم داشت. برای نمایش خروجی ها از سهEPROM (تراشه۲۷۳۲) استفاده کرده‌ایم که اینجا مثل یک دیکودر عمل می‌کنند. دوتای آنها برای دیکود کردن ۴ بیت اعشار و دیگری برای دیکود کردن قسمت صحیح جواب است. برای اینکار ابتدا تمام حالت‌های ممکن را برای اعداد بدست می‌آوریم و سپس با تبدیل این حالت‌ها به فایل intel hex و دادن این فایل به عنوان ورودی به EPROMها خروجی مورد نظر را نمایش می‌دهیم.

	A	B	C	D	E	F	G	H	I
1	0	0	0	0	0	00	00	0	00
2	0	0	0	1	0.0625	06	25	1	01
3	0	0	1	0	0.125	12	50	2	02
4	0	0	1	1	0.1875	18	75	3	03
5	0	1	0	0	0.25	25	00	4	04
6	0	1	0	1	0.3125	31	25	5	05
7	0	1	1	0	0.375	37	50	6	06
8	0	1	1	1	0.4375	43	75	7	07
9	1	0	0	0	0.5	50	00	8	08
10	1	0	0	1	0.5625	56	25	9	09
11	1	0	1	0	0.625	62	50	10	10
12	1	0	1	1	0.6875	68	75	11	11
13	1	1	0	0	0.75	75	00	12	12
14	1	1	0	1	0.8125	81	25	13	13
15	1	1	1	0	0.875	87	50	14	14

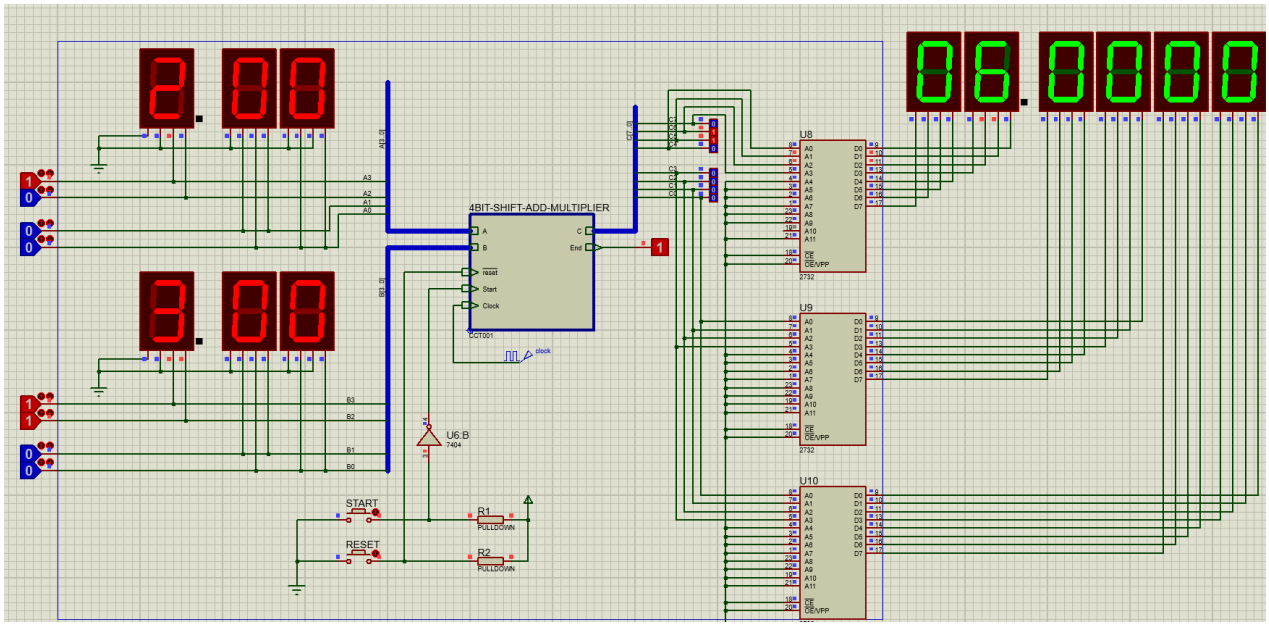
حالت‌های جواب



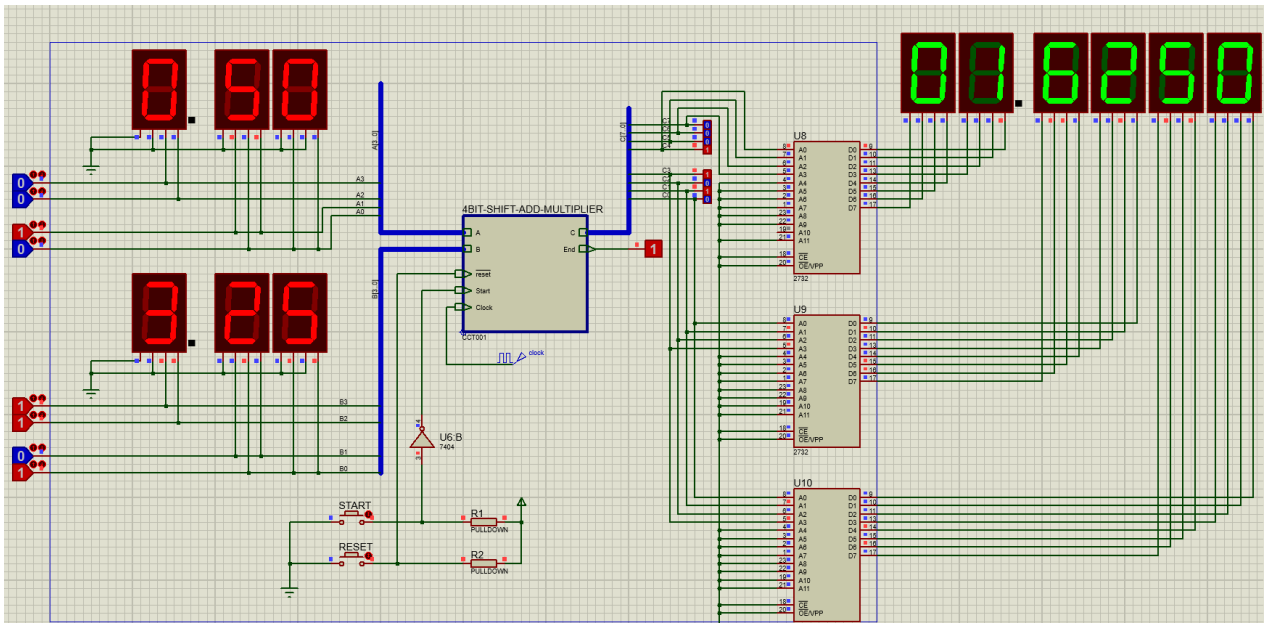
ماژول اصلی ضرب‌کننده ۴بیتی

بخش سوم: تست

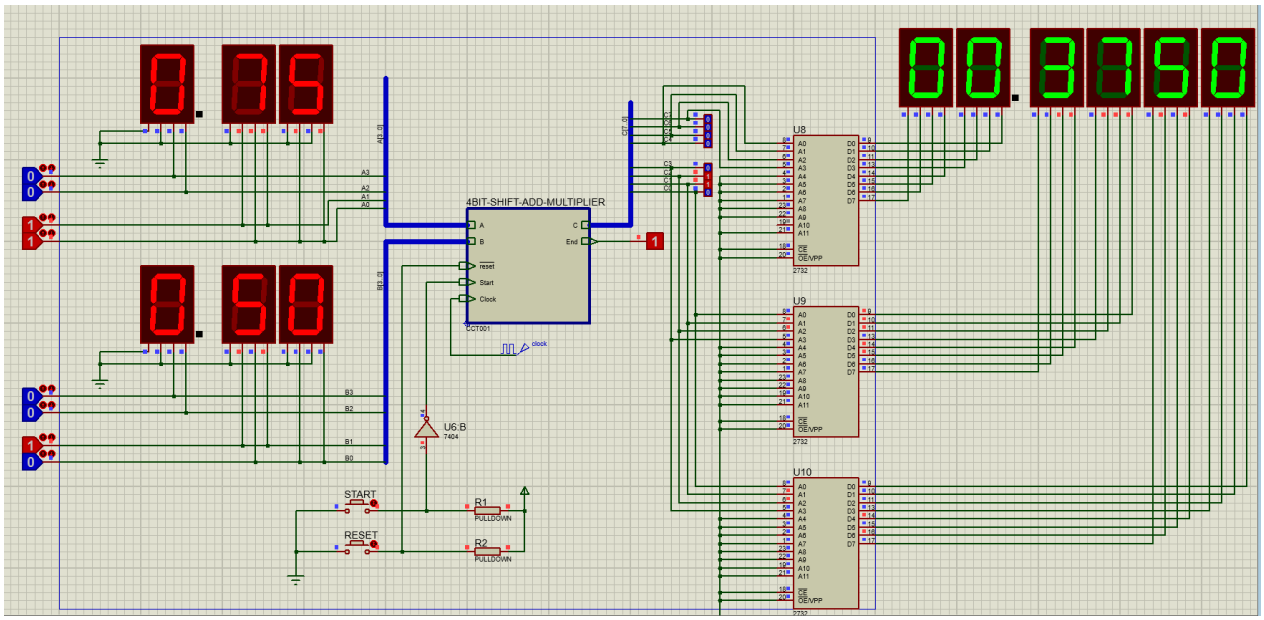
حال چند تست را برای نمایش نتیجه انجام میدهیم.



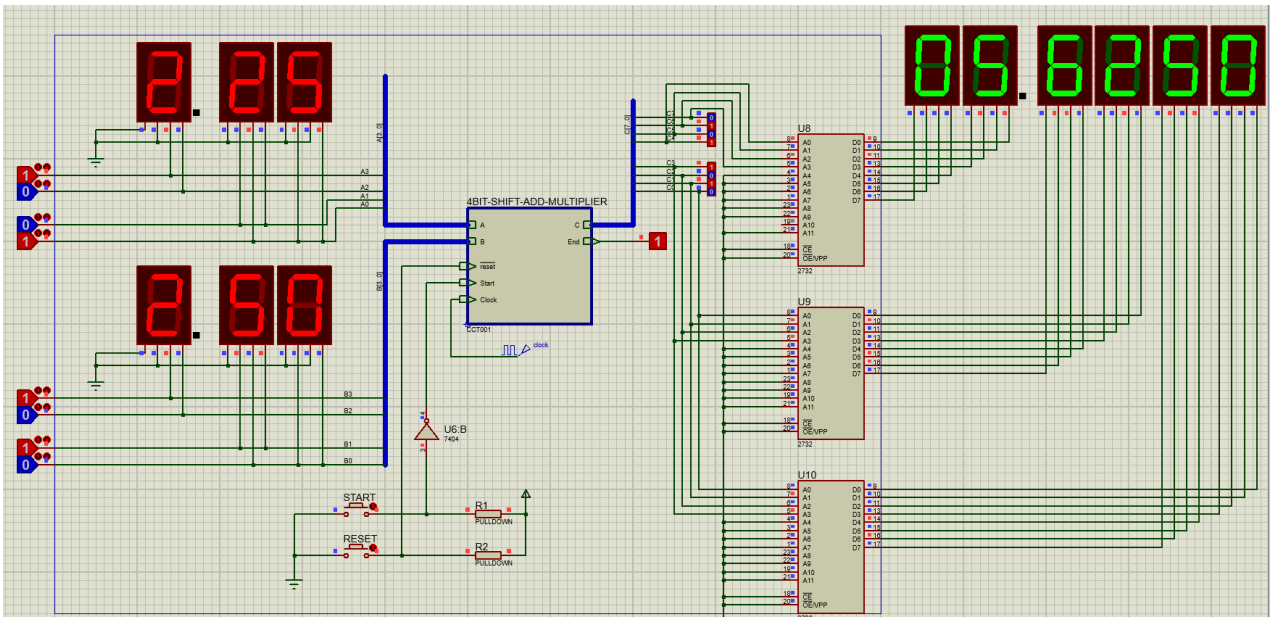
تست۱



تست۲


0.75
0.50
00.3750
4BIT-SHIFT-ADD-MULTIPLIER
A
B
C
End
reset
Start
Clock
CCT001
clock
U8
U9
U10
2732
U6:B
7404
START
RESET
R1
R2
PULLDOWN

تست۳


2.75
2.50
09.6250
4BIT-SHIFT-ADD-MULTIPLIER
A
B
C
End
reset
Start
Clock
CCT001
clock
U8
U9
U10
2732
U6:B
7404
START
RESET
R1
R2
PULLDOWN

تست۴