

# Computer Architecture LAB Report

امیرحسین عالیان

4021120017

امیرمهدی عزیزی

4021120019

## آزمایش دوم

تاریخ انجام آزمایش: ۱۹ مهر ۱۴۰۴

تاریخ تحویل گزارش: ۱۹ آبان ۱۴۰۴

## فهرست مطالب

۳	۱ چکیده
۳	۱.۱ هدف آزمایش
۳	۲.۱ قطعات و ابزار ها
۳	۳.۱ جدول نتایج چند تست
۳	۴.۱ پاسخ سوال ۱
۳	۵.۱ پاسخ سوال ۲
۴	۶.۱ پاسخ سوال ۳
۴	۷.۱ پاسخ سوال ۴
۵	۸.۱ پاسخ سوال ۵
۶	۲ توضیح IC ۷۴۱۸۱
۷	۳ اتصال قطعات و شماتیک مدار
۷	۱.۳ شکل اولیه
۷	۲.۳ شکل ثانویه (پاسخ سوال ۶ - ALU ۸ بیتی)

۸	۴	نتایج شبیه سازی - <b>ALU</b> ۴ بیتی
۸	۱.۴	عمل منطقی
۹	۲.۴	عمل حسابی
۱۰	۵	نتیجه شبیه سازی - <b>ALU</b> ۸ بیتی
۱۰	۱.۵	عمل حسابی
۱۲	۶	نتایج آزمایش عملی

# ۱ چکیده

## ۱.۱ هدف آزمایش

هدف این آزمایش، آشنایی و تجربه عملی با تراشه 74181 بود که در واقع یک واحد حساب و منطق یا همان ALU است. این واحد نیز بخشی از پیاده‌سازی مسیر داده (Datapath) است که در ادامه آزمایش اول (طراحی گذرگاه - BUS) دنبال میشود.

## ۲.۱ قطعات و ابزار ها

Component	Function	Quantity
IC 74181	4-bit ALU/Function Generator	1
White LED	$\overline{C_{n+4}}$ (Carry) Indicator	1
Green LED	4-bit $F_3F_2F_1F_0$ (Output) Indicator	4
Breadboard	Component Placement	1
DC Power Supply	Provides $V_{CC}$ and GND	1

Table 1: لیست قطعات مورد استفاده در این آزمایش

## ۳.۱ جدول نتایج چند تست

$S_3S_2S_1S_0$	$A_3A_2A_1A_0$	$B_3B_2B_1B_0$	M	$\overline{C_n}$	$\overline{C_{n+4}}$	$F_3F_2F_1F_0$	A=B	P	G
0 0 0 1	1 0 1 1	1 0 0 0	1	0	1	0 1 0 0	0	0	0
0 0 1 1	1 0 1 1	0 0 1 0	0	0	0	0 0 0 0	0	0	1
0 1 1 0	0 1 1 1	0 1 1 1	0	0	0	0 0 0 0	0	0	1
1 0 1 0	1 1 0 0	0 1 0 1	0	1	0	0 0 1 0	0	1	1

Table 2: جدول مقادیر خروجی

## ۴.۱ پاسخ سوال ۱

برای انجام عمل خواسته شده باید عدد 4 بیتی A را با رشته باینری 1000، XOR کنیم.

## ۵.۱ پاسخ سوال ۲

این پایه برای مقایسه دو عدد A و B مورد استفاده قرار میگیرد. برای این منظور باید آنرا در حالت تفریق قرار داد یعنی ورودی Select باید برابر با 0110 و مقدار  $\overline{C_n} = 0$  باشد. در صورتی که دو عدد

برابر باشند انتظار میرود که این خط خروجی 1 بدهد. با کمک خطوط  $\overline{C_n}$  و  $\overline{C_{n+4}}$  میتوان تفسیر دقیق تری درباره اندازه اعداد نسبت به هم (بزرگ و کوچک بودن) ارائه داد.

### ۶.۱ پاسخ سوال ۳

در منطق **Active-Low** و **Active-High** یک سری تفاوت ها وجود دارد. اولین تفاوت قابل ذکر، تفاوت در توابع منطقی و حسابی است به طوری که برخی توابع صرفا در منطق **Active-Low** و برخی دیگر در منطق **Active-High** قابل استفاده هستند البته برخی توابع بسیار پر کاربرد مانند **A MINUS B** یا **ZERO** در هر دو منطق قابل استفاده هستند.

تفاوت دیگر در نحوه عملکرد و خروجی دادن مدار هست برای مثال در منطق **Active-High** خروجی های  $F_3F_2F_1F_0$  از نوع **Active-High** خواهند بود و انتظار داریم به ازای بیت های روشن، لامپ ها نیز روشن شوند در حالی که در منطق **Active-Low** دقیقا برعکس است. جدول ۳ در زیر به جزئیات تفاوت بین ها در دو منطق اشاره میکند:

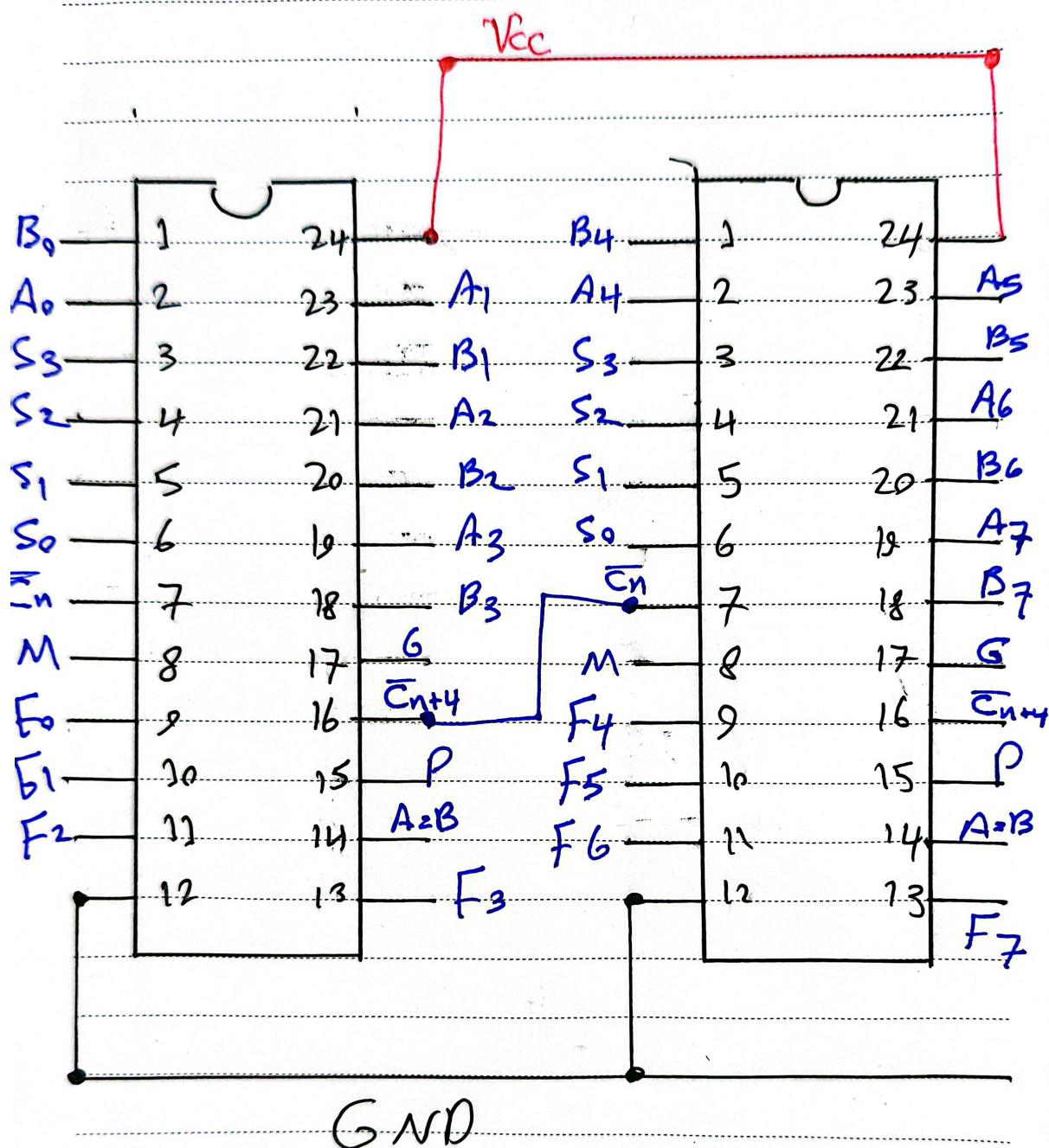
PIN	2	1	23	22	21	20	19	18	9	10	11	13	7	16	15	17
Active-low	$\overline{A_0}$	$\overline{B_0}$	$\overline{A_1}$	$\overline{B_1}$	$\overline{A_2}$	$\overline{B_2}$	$\overline{A_3}$	$\overline{B_3}$	$\overline{F_0}$	$\overline{F_1}$	$\overline{F_2}$	$\overline{F_3}$	$\overline{C_n}$	$\overline{C_{n+4}}$	$\overline{P}$	$\overline{G}$
Active-high	$A_0$	$B_0$	$A_1$	$B_1$	$A_2$	$B_2$	$A_3$	$B_3$	$F_0$	$F_1$	$F_2$	$F_3$	$C_n$	$C_{n+4}$	$X$	$Y$

جدول مقادیر خروجی: Table 3

### ۷.۱ پاسخ سوال ۴

پایه  $\overline{C_{n+4}}$  رقم نقلی تولید شده در جمع دو بیت آخر (**MSB**) را به ما می دهد. از این پایه برای گسترش واحد محاسبه و منطق (**ALU**) به تعداد بیت های بیشتر استفاده می کنیم، در واقع اگر بخواهیم یک **ALU** ۵ بیتی یا بیشتر از آن را بسازیم باید از چند تراشه **74181** استفاده کنیم که این پایه به عنوان ورودی رقم نقلی به پایه  $\overline{C_n}$  تراشه های دیگر داده میشود. پایه های  $\overline{P}$  و  $\overline{G}$  برای حالتی کاربرد دارند که بخواهیم محاسبات را به کمک جمع کننده از نوع **CLA** - **Carry Look Ahead** انجام دهیم. در این حالت از یک یا چند تراشه **74182** کمکی استفاده میکنیم تا با ترکیب دو تراشه با هم بتوان یک واحد محاسبه و منطق با تاخیر حداقلی بسازیم.

## ۸.۱ پاسخ سوال ۵

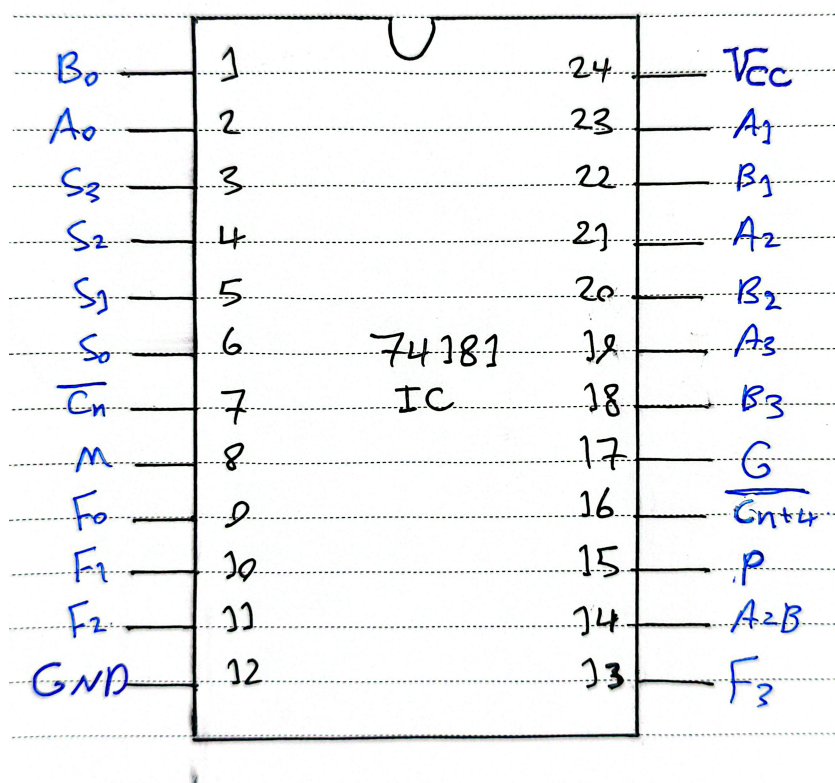


شکل ۱: بلاک دیاگرام برای ساخت ALU ۸ بیتی

اصلی ترین نکته این است که رقم نقلی C<sub>4</sub> تراشه اول را باید به عنوان رقم نقلی ورودی به تراشه دوم بدهیم. خطوط انتخاب (Select) ها باید به هر دو تراشه به صورت یکسان متصل شوند، همچنین خط M نیز باید میان هر دو مشترک باشد.

## ۲ توضیح IC ۷۴۱۸۱

## « 4-bit ALU/Function Generator »



شکل ۲: نمایی از پایه های IC ۷۴۱۸۱

در واقع این IC یک واحد محاسبه و منطق (ALU) است به این معنا که قادر است تعدادی عمل منطقی و نیز تعدادی عمل حسابی رو انجام دهد.

در صورتی که پایه  $M = 1$  باشد یعنی قصد داریم که یک عمل منطقی را انجام دهیم و در صورتی که  $M = 0$  باشد یعنی میخواهیم یک عمل حسابی را انجام بدهیم.

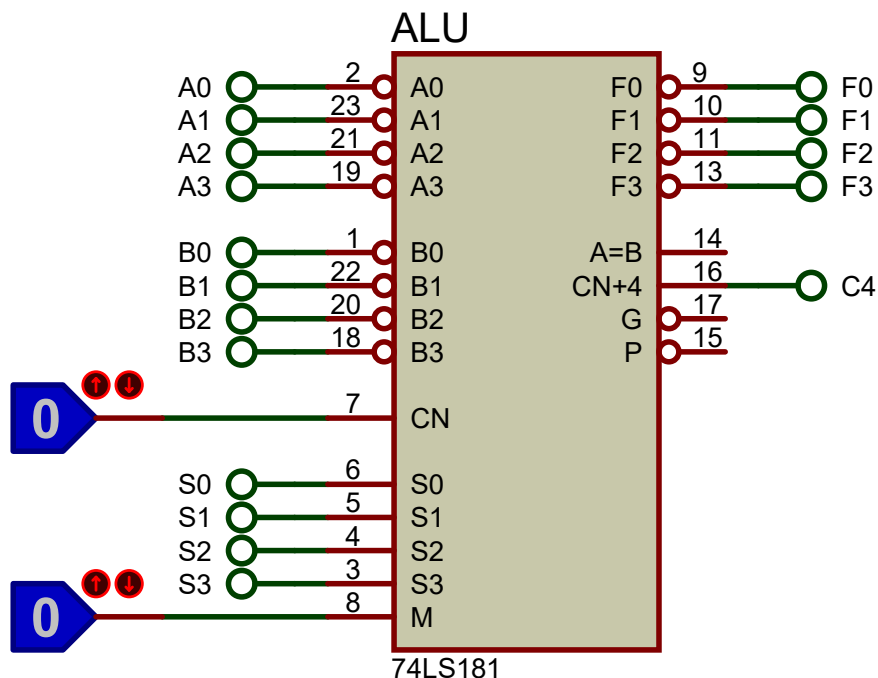
در حالتی که  $M = 0$  باشد می دانیم که با یک عمل حسابی مواجه هستیم اما برای پی بردن به تابع دقیق خواسته شده باید ورودی  $\overline{C_n}$  را نیز مورد بررسی قرار دهیم.

این IC را می توان به صورت منطق **Active-high** و نیز منطق **Active-low** بکار برد.

در این گزارش تمامی توضیحات و آزمایش ها در منطق **Active-high** انجام شده است.

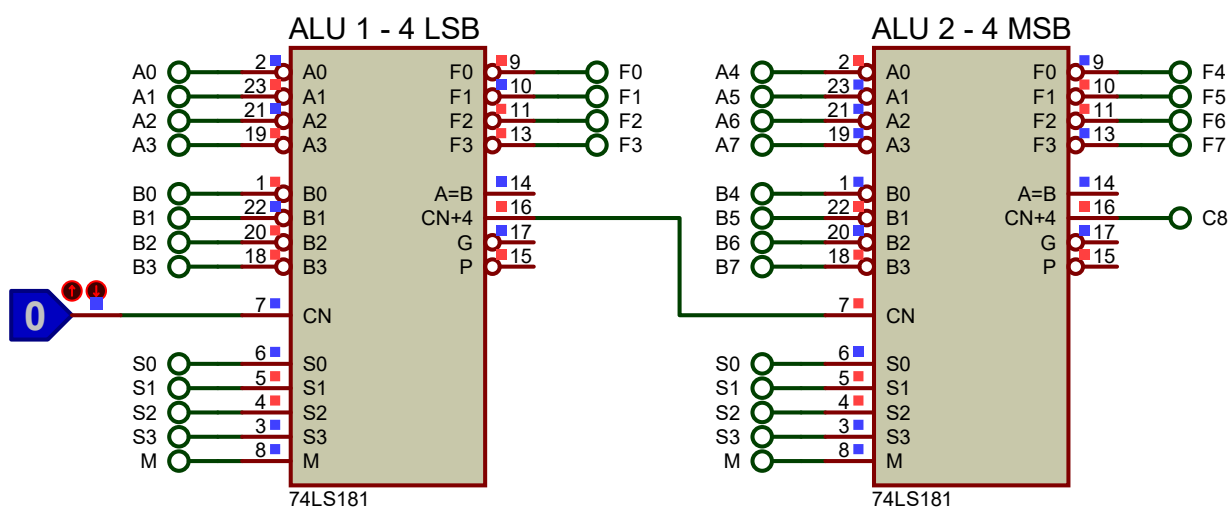
### ۳ اتصال قطعات و شماتیک مدار

#### ۱.۳ شکل اولیه



شکل ۳: شماتیک مدار در پروتئوس - ALU ۴ بیتی

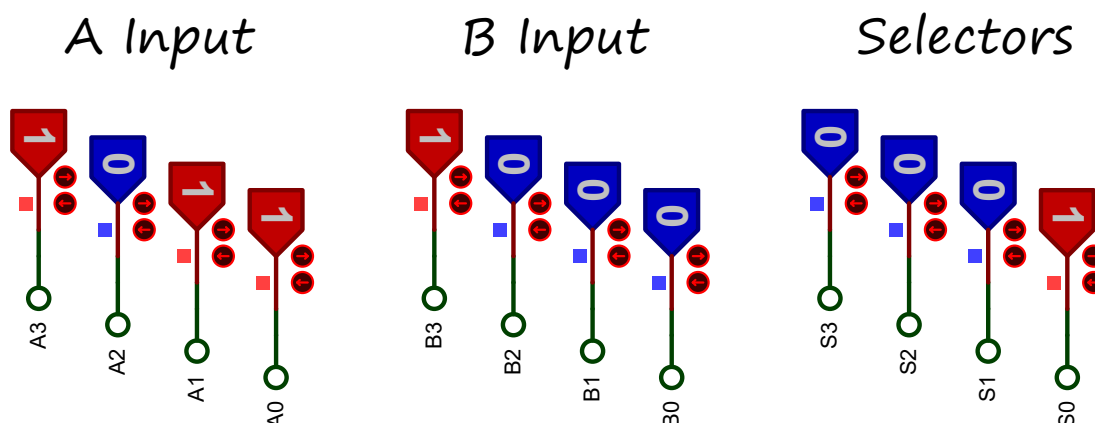
#### ۲.۳ شکل ثانویه (پاسخ سوال ۶ - ALU ۸ بیتی)



شکل ۴: شماتیک مدار در پروتئوس - ALU ۸ بیتی

## ۴ نتایج شبیه سازی - $ALU$ ۴ بیتی

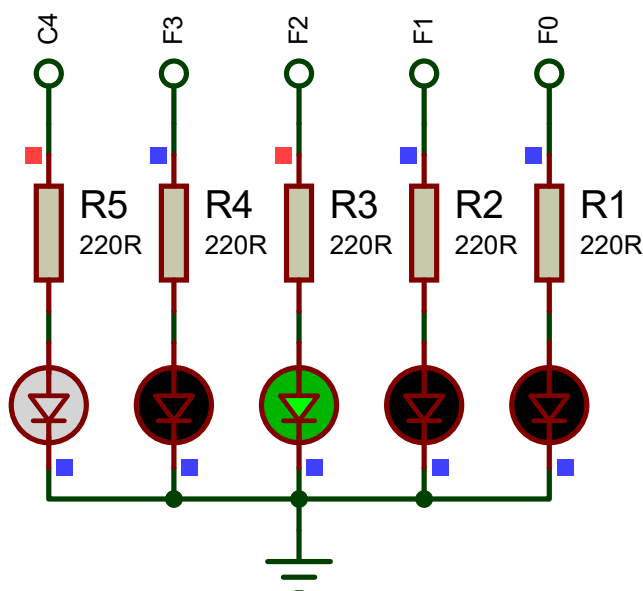
### ۱.۴ عمل منطقی



شکل ۵: ورودی شبیه سازی به ازای  $M = 1, \overline{C_n} = 0$

در این مثال  $A = 11$  و  $B = 8$  است، همچنین مقدار ورودی  $Selector$  ها برابر با عدد ۱ است که به ازای  $M = 1$  برابر با  $F = A \text{ NOR } B$  میشود. (در حالتی که  $M = 1$  است مقدار  $\overline{C_n}$  در تعیین تابع نقشی ندارد)

باید دقت داشت که در اعمال منطقی خروجی  $\overline{C_{n+4}}$  نیز اهمیتی ندارد. خروجی مورد انتظار عدد ۴ است که مطابق با نمایش باینری  $LED$  های روشن شده در زیر است:

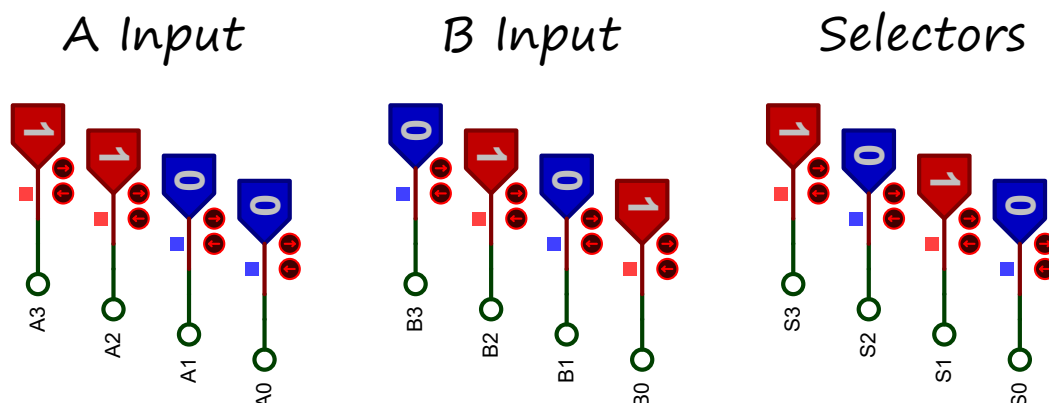


## Outputs

شکل ۶: خروجی شبیه سازی در حالی که انتظار داریم ۴ را دریافت کنیم

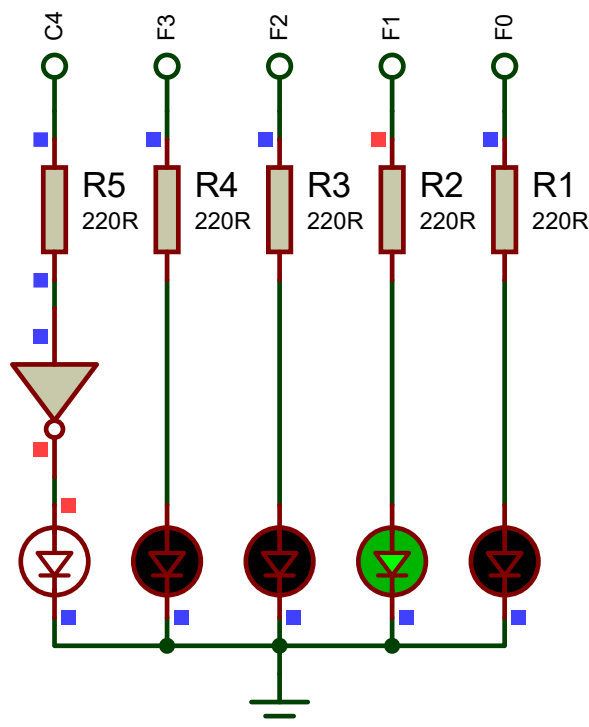


## ۲.۴ عمل حسابی



شکل ۷: ورودی شبیه‌سازی به‌ازای  $M = 0$ ,  $\overline{C_n} = 1$

در این مثال  $A = 12$  و  $B = 5$  است، همچنین مقدار ورودی Selector ها برابر با عدد 10 است که به ازای  $M = 0$  و  $\overline{C_n} = 1$  برابر با  $F = (A + B) \text{ PLUS } AB$  میشود. باید دقت کرد که خروجی خط  $\overline{C_n}$  از نوع Active-Low است به همین دلیل برای مشاهده مقدار آن در منطق Active-High در سر راه آن یک گیت NOT قرار داده ایم. نمایش باینری خروجی مورد انتظار در زیر است:

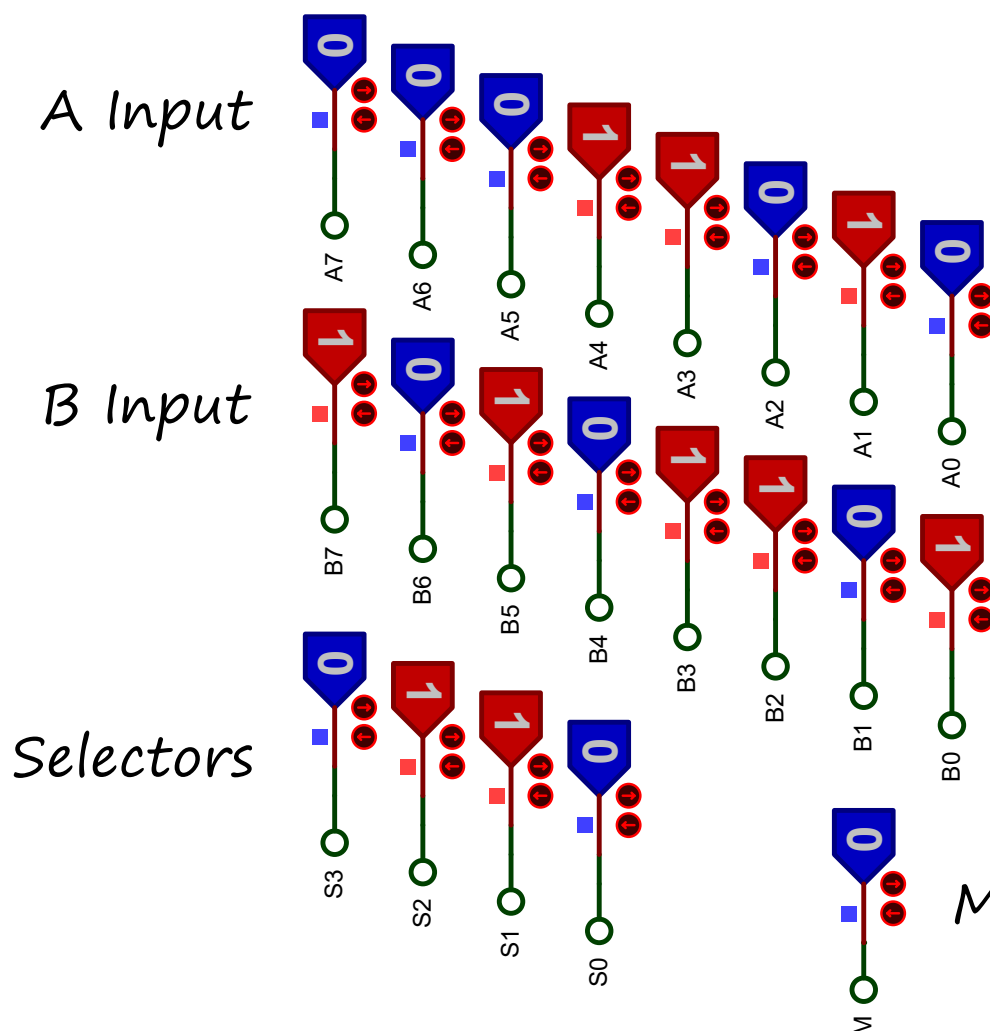


Outputs

شکل ۸: خروجی شبیه‌سازی در حالی که انتظار داریم 18 را دریافت کنیم

## ۵ نتیجه شبیه سازی - ALU ۸ بیتی

۱.۵ عمل حسابی

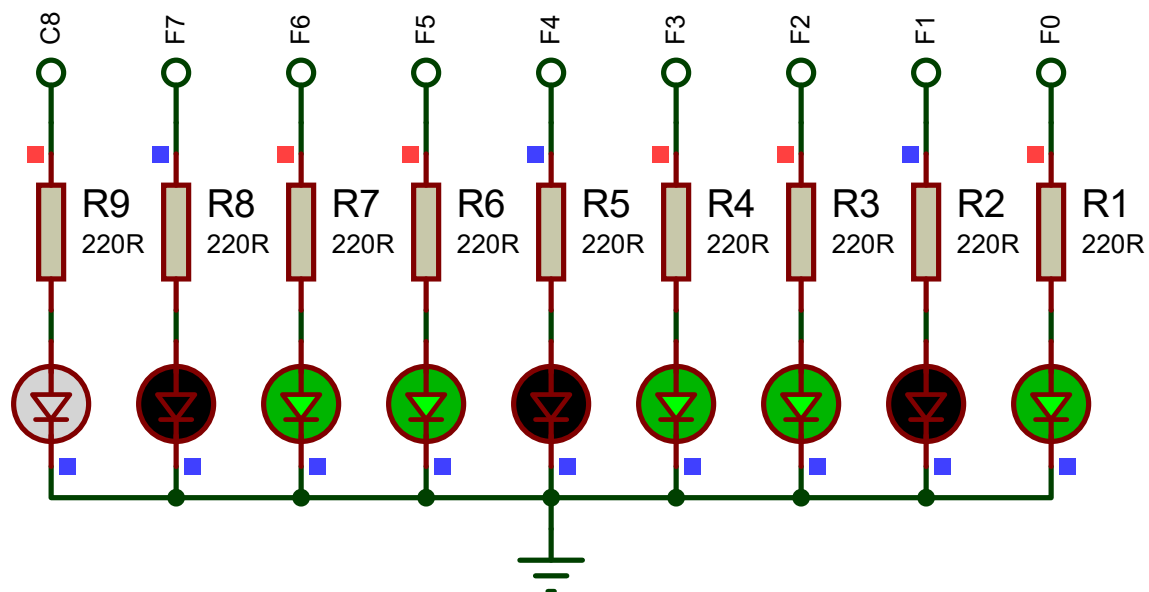


شکل ۹: ورودی شبیه سازی به ازای  $M = 0$ ,  $\overline{C_n} = 1$

در این مثال  $A = 26$  و  $B = 173$  است، همچنین مقدار ورودی Selector ها برابر با عدد 6 است که به ازای  $M = 0$  و  $\overline{C_n} = 1$  برابر با  $F = A \text{ MINUS } B$  میشود.

باید دقت داشت که در تفريق کردن چنانچه حاصل منفی شود، خط C8 اطلاعات درستی به ما نمی دهد و ما در تفسیر نتیجه آنرا نادیده میگیریم. برای تفسیر دقیق تر نتیجه عملیات حسابی و یافتن علامت عدد حاصل شده باید این کار را به کمک خطوط  $\overline{C_n}$ ،  $\overline{C_{n+4}}$  و  $A=B$  انجام دهیم.

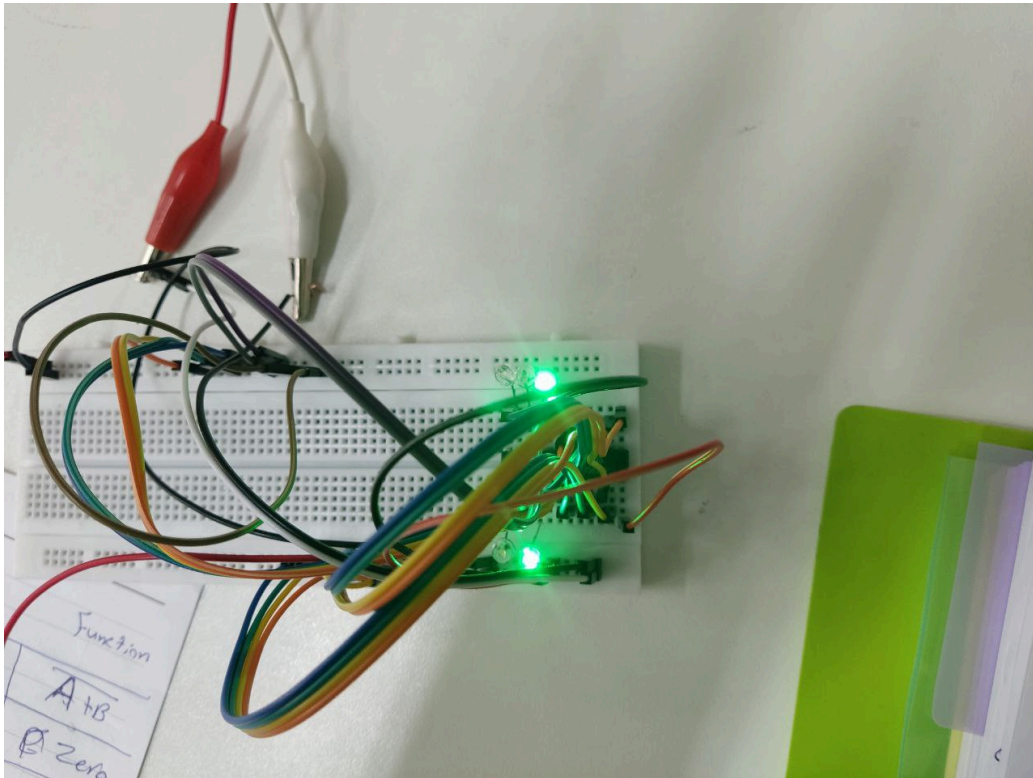
خروجی مورد انتظار عدد 147- است که مطابق با نمایش باینری LED های روشن شده در زیر است:



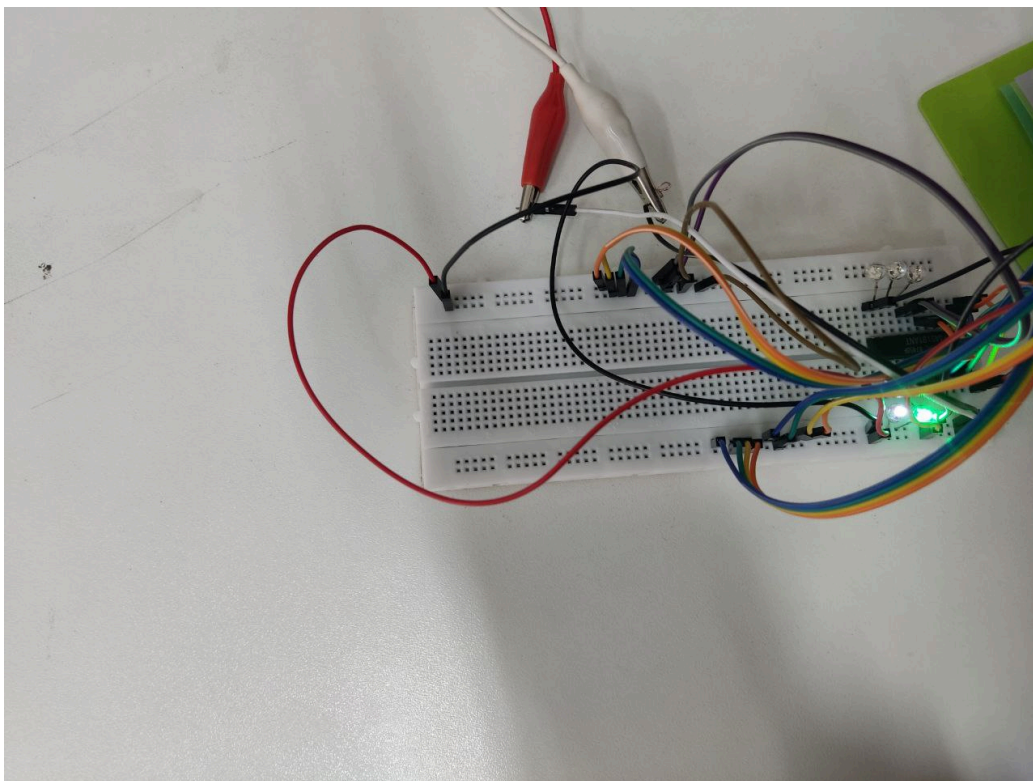
*Outputs*

شکل ۱۰: خروجی شبیه سازی در حالی که انتظار داریم 147- را دریافت کنیم

## ۶ نتایج آزمایش عملی



شکل ۱۱: نتیجه آزمایش عملی به ازای  $\overline{C_n} = 0, M = 1, B = 4, A = 7, S_3 S_2 S_1 S_0 = 0001$



شکل ۱۲: نتیجه آزمایش عملی به ازای  $\overline{C_n} = 1, M = 0, B = 1, A = 11, S_3 S_2 S_1 S_0 = 0110$