فصل ۲: بیتها، انواع داده ها و عملیات

۲-۱- بیتها و انواع داده ها

۲-۱-۱- بیت: واحد اطلاعات

همانطور که در فصل ۱ بیان شد، رایانه سیستمی با چندین مرحله تبدیل اطلاعات میباشد. برای مثال کار با زبانهای طبیعی مانند زبان انگلیسی در چندین مرحله توسط الکترونیک رایانه، در چندین مرحله، به حرکت الکترونها تبدیل شده است. بمعنای دیگر برای کار با داده ها باید حرکت الکترونها بررسی شود.در رایانه ها میلیونها قطعات کوچک ولی خیلی سریع وجود دارند که حرکت الکترونها را، توسط ولتاژ در مدار، کنترل میکنند. در عین اینکه این قطعات میتوانند به مقدار ولتاژ در مدار، عکس العمل نشان دهند ولی در عمل بسیار ساده تر است که به وجود یا عدم وجود ولتاژ عکس العمل نشان دهند. این بدان علت است که عکس العمل نشان دادن به وجود و یا عدم وجود ولتاژ بین دو نقطه بسیار ساده تر از اندازه گیری دقیق ولتاژ بین آن دو نقطه میباشد.
برای درک بهتر این موضوع، یک پریز برق را در نظر بگیرید. شما میتوانید ولتاژ دقیق آنرا اندازه گیری نمایید، مثلاً ۲۱۰، ۲۰۱ و یا ۲۲۵ ولت اندازه گیری خواهد شد. ولیکن بسیار ساده تر است که ببینیم آیا ولتاژ در پریز هست یا خیر، مثلاً انگشت خود را داخل پریز قرار دهید!!!!

اگر بخواهیم دقیق تر باشیم، باید بگوئیم که مدارهای الکترونیکی رایانه مقدار مطلق ولتاژ صفر بعنوان صفر در نظر نمی گیرند. بلکه مقادیر نزدیک به صفر را بعنوان صفر در نظر میگیرد (مقادیر کمتر از ۰,۵ ولت). به همین ترتیب، وجود ولتاژ کامل را بعنوان یک در نظر نمی گیرد. بلکه مقادیر نزدیک به ولتاژ کامل را بعنوان یک در نظر می گیرد (هر ولتاژی بالاتر از ۲,٤ ولت تا ۲,۹ ولت یک در نظر گرفته میشود).

برای سادگی نمایش از "۱" بعنوان وجود ولتاژ و از "۰" بعنوان عدم وجود ولتاژ استفاده میشود. ما به هر کدام از ۰ و یا ۱، یک "بیت" میگوییم که مخفف کلمه عدد باینری (دوتائی) میباشد[1]. همانند ارقام در مبنای ۱۰ که ۰ تا ۹ میباشند و ارقام دهگانه خوانده میشوند، در باینری دو رقم صفر و یک وجود دارند که ارقام باینری خوانده میشوند.

برای آنکه بتوان کار سودمندی توسط رایانه انجام داد لازم است که بتوان اعداد منحصر بفرد زیادی را توسط آن نمایش داد. لیکن ولتاژ سر سیم فقط دو مقدار منحصر بفرد را میتواند نمایش دهد، یکی را با ۰ و دیگری را با ۱. لذا برای نمایش منحصر بفرد تعداد زیادی عدد، نیاز به ترکیب چندین بیت میباشد. برای مثال، با هشت بیت، که مربوط به ولتاژ سر هشت سیم میباشد، مثلاً میتوان مقدار ۰۱۰۰۱۱۱۰ و یا ۱۱۱۰۱۱۰۱ را نمایش داد. در واقع با هشت بیت میتوان ۲٥٦ ($۲^۸$) مقدار مختلف را نمایش داد. اگر k بیت داشته باشیم میتوان ۲^k مقدار مختلف را نمایش داد. هر کدام از نمایشهای k بیتی کدی میباشد که مربوط به یک مقدار منحصر بفرد میباشد.

۲-۱-۲- انواع داده ها

برای نمایش یک مقدار روشهای مختلفی وجود دارد. برای مثال عدد ٥ را میتوان به روش نمایش دسیمال (دهتایی) بصورت ٥ نشان داد. همچنین اگر فردی دست خود را بالا بگیرد و انگشتان خود را باز نشان دهد میتواند نمایشی از عدد ٥ باشد. به بیان دیگر آن فرد میگوید "عددی که من به

---

۱ bit: binary digit

شما نشان میدهم میتواند با شمردن انگشتان مـن مشـخص گـردد". نوشـته شـده ایـن روش ۱۱۱۱۱ میباشد که نمایش باز بودن ۵ انگشت است. این روش به تکگانه۲ معروف است. در زبان یونانی، عدد ۵ را با V نمایش میدهند که روش دیگـری از نمـایش داده هـا و اعـداد اسـت. روش دیگـر، روش نمایش باینری میباشد که در رایانه ها استفاده میگردد و عدد پنج با ۰۰۰۰۰۱۰۱ نمایش داده میشود.

باید توجه داشت که این کافی نیست که فقط اعداد را نمایش دهیم. لازم است که بتوان بر روی آنهـا عملیات انجام دهیم. به دادههایی که میتوان عملیاتی بر روی آنها انجـام داد، نـوع داده یـا data type گفته میشود. در رایانهها به هر مجموعه انواع دادهها و دستورات مربوط بـه آنهـا سـاختار مجموعـه دستورالعمل۳ گفته میشود. در این کتاب، ما بطور عمده از دو نوع داده استفاده میکنیم: اعداد مکمل ۲ برای نمایش اعداد مثبت و منفی به منظور انجام عملیات ریاضی و کـدهای اسـکی (ASCII codes) برای نمایش حرفهایی که توسط کی بورد به رایانه وارد کرده و یا بـر روی صـفحه تصـویر کـامپیوتر نشان میدهیم. هر دوی این نوع داده ها را بزودی توضیح خواهیم داد.

انواع داده های دیگری هم وجود دارد که در رایانه ها مورد استفاده قرار گرفته و در بسیاری از آنها موجود میباشند. مثلاً "نمایش علمی" اعداد را بیاد آورید که در آن عـدد دسـیمال ٦٢١ بصـورت $٦,٢١ \times ١٠^{٢}$ نمایش داده میشود. رایانه هایی وجود دارند که اعـداد را بـه اینصـورت نمـایش داده و عملیاتی را فراهم می آورند که بر روی این اعداد عمل میکنند. این نمایش علمـی اعـداد بـه نمـایش ممیز شناور٤ معروف میباشد. این روش نمایش در بخش ۲-٦ توضیح داده خواهد شد.

۲ unary

۳ ISA: Instruction Set Architecture

۴ floating point

۲-۲- اعداد صحیح

۲-۲-۱- اعداد صحیح بدون علامت

اولین نوع داده، به بیان دیگر روش نمایش اطلاعات، که باید مورد بررسی قرار گیرد اعداد صحیح بدون علامت میباشند. این نوع داده ها مصارف بسیار زیادی در رایانه ها دارند. اگر میخواهیم یک کار را به تعداد مشخص تکرار نماییم، اعداد بدون علامت این امکان را میدهد که دفعاتی را که آن کار انجام شده را بشماریم. اینکار را میتوان به سادگی با شمارش تعداد دفعات با اضافه کردن یک عدد بدون علامت انجام داد. همچنین اعداد بدون علامت را میتوان برای مشخص کردن خانه های حافظه، به همان صورتی که خانه‌ها در یک کوچه شماره گذاری شده اند مثلاً شماره ۲۹ کوچه موحدی و شماره ۳۲ کوچه موحدی، و تمایز بین خانه‌های حافظه استفاده میگردند.

برای نمایش اعداد صحیح بدون علامت از رشته ای از بیت‌ها، ارقام باینری، استفاده میشود. بدین منظور، همانند سیستم اعداد دسیمال، محل هر بیت دارای مقدار مشخص است. بعنوان مثال شما با عدد دسیمال ۳۲۹ آشنا هستید که محل هر رقم معنا و نماد خودش را داراست. ۳ در۳۲۹ از عدد ۹ دارای ارزش بیشتری است در حالیکه مقدار مطلق ۹ از ۳ بیشتر است. این بدان خاطر است که ۳ معادل با ۳۰۰ (3×10^2) بوده در حالیکه ۹ معادل 9×10^0 (نه ضرب در ۱۰ بتوان صفر) میباشد. به بیان دیگر ۳ در محل صدگان بوده در حالیکه ۹ در محل یکان است.

نمایش باینری اعداد صحیح بدون علامت نیز شبیه به روش دسیمال می باشد، با این تفاوت که ارقام استفاده شده بیتهای ۰ و ۱ هستند و مبنای عدد ۲ است بجای ۱۰. برای مثال اگر ما ۵ بیت برای نمایش داریم، عدد ٦ را میتوان به صورت ۰۰۱۱۰ نمایش داد که معادل زیر می باشد.

$$0\times2^4+0\times2^3+1\times2^2+1\times2^1+0\times2^0$$

با داشتن k بیت میتوان 2^k عدد صحیح را بین ۰ و 2^k-1 نمایش داد. در مورد مثال بالا، یک عدد

باینری پنج بیتی میتواند اعداد صحیح بدون علامت بین ۰ تا ۳۱ را نمایش دهد.

۲-۲-۲- اعداد صحیح علامت دار

بسیاری از کارهای محاسباتی نیاز به کار با اعداد منفی هم دارند. لذا میتوان مجموعه ۲k مقادیر قابل نمایش با k بیت را به دو نیمه تقسیم کرده، نیمی را برای نمایش اعداد منفی و نیم دیگر را برای نمایش اعداد مثبت استفاده نمود. بدین ترتیب، عدد پنج بیتی مثال بالا که برای نمایش اعداد بین ۰ تا ۳۱ استفاده شده را میتوان برای نمایش اعداد بین ۱ تا ۱۵ و ۱- تا ۱۵- استفاده کرد، یعنی ۳۰ عدد صحیح که ۱۵ عدد منفی و ۱۵ عدد مثبت میباشند. از آنجا که ۲۵ برابر ۳۲ می شود، در نتیجه هنوز دو ترکیب ۵ بیتی به هیچ عددی اختصاص داده نشده اند. یکی از آنها ۰۰۰۰۰ است که قاعدتاً به صفر اختصاص داده خواهد شد بدین ترتیب مجموعه اعداد بین ۱۵- و ۱۵ کامل میگردد. بدین ترتیب یک عدد ۵ بیتی باقی میماند که به روشهای مختلف میتوان آنرا به عددی نسبت داد که در ادامه بدان اشاره خواهد شد.

مسئله دیگر اینست که چگونه می توان هر کد ۵ بیتی، ترکیبی از صفرها و یک ها، را به یک عدد بین ۱۵- و ۱۵ اختصاص داد. میتوان اعداد مثبت را سمت راست، همانند روش قبل، بر مبنای محل اختصاص داد. چون k بیت موجود است و قصد داریم که نصف ۲k کد را برای نمایش اعداد از ۰ تا ۲ k-۱ استفاده کنیم، تمامی اعداد مثبت صحیح یک، ۰ در سمت چپ خود خواهند داشت. در مثال بالا (k=۵) بزرگترین عدد، یعنی ۱۵، معادل ۰۱۱۱۱ خواهد بود.

جدول ۲-۱- سه روش نمایش اعداد صحیح علامت دار

نمایش	مقدار نمایش داده شده		
	علامت و مقدار	مکمل ۱	مکمل ۲
٠٠٠٠٠	٠	٠	٠
٠٠٠٠١	١	١	١
٠٠٠١٠	٢	٢	٢
٠٠٠١١	٣	٣	٣
٠٠١٠٠	٤	٤	٤
٠٠١٠١	٥	٥	٥
٠٠١١٠	٦	٦	٦
٠٠١١١	٧	٧	٧
٠١٠٠٠	٨	٨	٨
٠١٠٠١	٩	٩	٩
٠١٠١٠	١٠	١٠	١٠
٠١٠١١	١١	١١	١١
٠١١٠٠	١٢	١٢	١٢
٠١١٠١	١٣	١٣	١٣
٠١١١٠	١٤	١٤	١٤
٠١١١١	١٥	١٥	١٥
١٠٠٠٠	-٠	-١٥	-١٦
١٠٠٠١	-١	-١٤	-١٥
١٠٠١٠	-٢	-١٣	-١٤
١٠٠١١	-٣	-١٢	-١٣
١٠١٠٠	-٤	-١١	-١٢
١٠١٠١	-٥	-١٠	-١١
١٠١١٠	-٦	-٩	-١٠
١٠١١١	-٧	-٨	-٩
١١٠٠٠	-٨	-٧	-٨
١١٠٠١	-٩	-٦	-٧
١١٠١٠	-١٠	-٥	-٦
١١٠١١	-١١	-٤	-٥
١١١٠٠	-١٢	-٣	-٤
١١١٠١	-١٣	-٢	-٣
١١١١٠	-١٤	-١	-٢
١١١١١	-١٥	-٠	-١

توجه نمایید که در تمامی روشهای نمایش نشان داده شده درجدول ۲-۱ صفر و اعداد صحیح مثبت، همگی با صفر سمت چپ شروع میشوند. در مورد اعداد صحیح منفی در مثال ۵ بیتی ما اعداد ۱- تا ۱۵- چگونه است؟ اولین فکری که به ذهن میآید اینست که چون اعداد مثبت با صفر سمت چپ شروع میشوند، چطور است که اعداد صحیح منفی را با یک سمت چپ شروع نماییم؟

این نوع نمایش همان روش علامت و مقدار است که در جدول ۲-۱ نشان داده شده است. روش دوم، که در تعدادی از کامپیوترهای اولیه، همچون CDC٦٦٠٠ استفاده شده است، به ترتیب زیر است:

یک عدد منفی با معکوس کردن تمامی بیتهای یک عدد مثبت حاصل میشود. مثلاً عدد ۵- از معکوس کردن تمامی بیتهای عدد ۵، یعنی ۰۰۱۰۱، حاصل میشود و معادل ۱۱۰۱۰ میباشد. این روش نمایش اعداد منفی به مکمل یک معروف است که در شکل ۲-۱ نشان داده شده است.

با توجه با مباحث بالا شما ممکن است فکر کنید که طراح یک رایانه میتواند هر رشته بیتی را به هر عدد صحیحی که خود میخواهد اختصاص دهد. شما درست فکر کردهاید. لیکن این روش میتواند مشکلات زیادی در طراحی و ساخت یک مدارمنطقی برای جمع دو عدد صحیح ایجاد نماید. به همین علت است که روشهای مقدار علامت و روش مکمل یک هر دو مناسب برای عمل جمع نبوده و طراحی مدارات منطقی جمع برای ایندو مشکل میباشد. بدین علت، طراحان کامپیوتر، که از قبل میدانستند که چگونه مدار منطقی جمع دو عدد را طراحی کنند، بدنبال نمایشی از اعداد بودند که بتوان جمع اعداد را با ساده سازی مدار منطقی انجام داد. لذا روش مکمل دو بوجود آمد که اکنون در تمامی رایانه های تولید شده استفاده میشود.

۲-۳- اعداد صحیح مکمل دو

در جدول ۲-۱ نمایش اعداد ١٦- تا ١٥ به روش مکمل دو نشان داده شده است. سوال اینست که چرا این نمایش انتخاب شده است؟ همانطور که قبلا نشان داده شد، اعداد صحیح مثبت را میتوان

بسادگی بر مبنای موقعیت مکانی بیتها نشان داد. با داشتن ۵ بیت، ما دقیقا نصف ۲۵ نمایش ممکن را برای نمایش ۰ و اعداد صحیح مثبت بین ۱ تا $۲^{٤}-۱$ استفاده میکنیم.

انتخاب روش نمایش اعداد منفی بر مبنای این ایده است که تا حد امکان مدارهای منطقی را ساده نگاه داشت، به بیان دیگر سعی شود که نمایش مطابق با روش کنونی جمع باشد که تقریبا در تمامی رایانه های موجود شبیه میباشد. واحدی که در ریزپردازنده رایانه مسئولیت جمع را دارد بنام واحد عملیات ریاضی و منطقی (ALU) خوانده میشود. در فصل ۳ و ٤ به توضیح بیشتر ساختار این واحد خواهیم پرداخت. نکته مورد نظر در اینجا این است که این واحد دارای ۲ ورودی و یک خروجی میباشد. هدف آن اینست که دو ورودی را گرفته و عمل جمع را بر روی رشته بیت های ورودی انجام داده و خروجی را، که نتیجه جمع دو ورودی است، بصورت یک رشته بیت نمایش دهیم.

برای مثال اگر ورودی به ALU دو رشته ۵ بیتی باشد، که یکی ۰۰۱۱۰ و دیگری ۰۰۱۰۱ باشد، خروجی ALU که نتیجه جمع دو ورودی است ۰۱۰۱۱ خواهد بود. عمل جمع به شرح زیر است:

$$\begin{array}{r} ۰۰۱۱۰ \\ ۰۰۱۰۱ \\ \hline ۰۱۰۱۱ \end{array}$$

جمع دو عدد باینری همانند جمع دو عدد دسیمال با جمع از راست به چپ و ستون به ستون انجام میگیرد. اگر جمع دو مقدار باعث ایجاد رقم انتقالی گردد، آن رقم به ستون بعدی سمت چپ اضافه میگردد.

نکته قابل توجه اینست که ALU نمی داند که دو رشته بیتهایی که جمع میکند چه مقادیری را نشان میدهند. ALU فقط ایندو را جمع میکند بدون توجه به مقادیر نمایش داده شده توسط ایندو رشته بیت. لذا بسیار مفید خواهد بود اگر نمایش عدد بگونه ای باشد که نتیجه جمع توسط ALU

خود نمایش صحیح عدد باشد. مثلاً در یک نمایش، وقتی یک عدد با منفی خود جمع گردد نتیجه عمل صفر می شود. به بیان دیگر اگر ورودی به ALU اعداد A و A- باشند، خروجی ALU باید ۰۰۰۰۰ باشد.

بدین منظور روش نمایش مکمل ۲ طراحی گردیده بگونه ای که اگر یک عدد با منفی خود جمع گردد، عدد خروجی معادل صفر در نمایش مکمل دو میباشد. مثلاً چون ۰۰۱۰۱، نمایش عدد ۵+ است، ۱۱۰۱۱ نمایش عدد ۵- میباشد و جمع ایندو معادل صفر است. آنچه که بسیار حائز اهمیت می باشد این است که تا زمانیکه نتیجه بزرگتر از ۱۵+ و کوچکتر از۱٦- بدست نیامده، این اضافه کردن ALU به درستی انجام خواهد گرفت.

توجه دقیق به نمایش ۱- و ۰، ۱۱۱۱۱ و ۰۰۰۰۰، ضروریست. هنگامیکه عدد۱، با نمایش مکمل دو ۰۰۰۰۱ ،را با ۱-،با نمایش مکمل دو ۱۱۱۱۱، جمع میکنیم، حاصل ۰۰۰۰۰ میباشد. ولیکن یک بیت انتقالی هم نتیجه این عمل در انتها، خواهد بود. این بیت انتقالی نتیجه ای در جمع ندارد، یعنی جمع ۱ و ۱- باید ۰۰۰۰۰ شود نه ۱۰۰۰۰۰. در نتیجه بیت انتقالی در نظر گرفته نمیشود. در واقع بیت انتقالی سمت چپ در مکمل دو بی تاثیر بوده و همواره کنار گذاشته می شود.

نکته: یک راه میانبر برای اینکه نمایش A- (A≠۰) را بدست آوریم اینست که اگر نمایش A را میدانیم، تمامی بیتهای آنرا معکوس کرده، به بیانی مکمل آنرا بدست آورده، سپس نمایش حاصل را با ۱ جمع نماییم. نمایش بدست آمده نمایش مکمل دو A- میباشد. اگر A را با معکوس شده A، مکمل A، جمع نماییم نتیجه ۱۱۱۱ خواهد بود. مثلاً جمع ۱، یعنی ۰۰۰۱، و معکوس شده ۱، یعنی ۱۱۱۰، معادل ۱۱۱۱ میشود. لذا نتیجه جمع آن با ۰۰۰۱، عدد صفر، یعنی ۰۰۰۰، حاصل میشود. بدین جهت نمایش مکمل دو A- در نتیجه جمع ۱ با معکوس شده A بدست میاید.

مثال۲-۱ : نمایش مکمل دو عدد ۱۳- چیست؟

۱. فرض کنید A مساوی ۱۳+ است. لذا نمایش باینری آن معادل ۰۱۱۰۱ میباشد.
۲. معکوس شده، یا مکملA مساوی ۱۰۰۱۰ میباشد.

۳. با اضافه کردن ۱ مکمل دو آن بدست میاید یعنی ۱۰۰۱۱

برای اثبات صحت عمل، میتوان ۱۳ را با ۱۳- جمع نمود:

۰۱۱۰۱

۱۰۰۱۱

۰۰۰۰۰

ممکن است که متوجه شده باشید که هنگام جمع ۰۱۱۰۱ و ۱۰۰۱۱، خروجی ۰۰۰۰۰ و ALU یک بیت انتقالی ایجاد میکند که بیشتر از ۵ بیت میشود. به بیان دیگر جمع ۰۱۱۰۱ و ۱۰۰۱۱ در واقع معادل ۱۰۰۰۰۰ میباشد. ولیکن، همانطور که قبلا هم بیان شد، در مکمل ۲، این بیت انتقالی اضافی در نظر گرفته نمیشود.

تا این مرحله، ما با ۵ بیت ۱۵ عدد مثبت و ۱۵ عدد منفی را نمایش داده ایم. همچنین یک نمایش برای ۰ داریم. از آنجا که تعداد بیتها ۵ میباشد، یعنی k=۵، باید بتوانیم ۳۲ مقدار متفاوت را نمایش دهیم در حالیکه تا اینجا ما فقط ۳۱ (۱۵+۱۵+۱) عدد را نمایش داده ایم. نمایش باقیمانده ۱۰۰۰۰۰ است. بنظر شما چه عددی را به این نمایش اختصاص دهیم؟

ما متوجه شده ایم که ۱- معادل ۱۱۱۱۱، ۲- معادل ۱۱۱۱۰ ، ۳- معادل ۱۱۱۰۱، ... و ۱۵- معادل ۱۰۰۰۱ میباشد. توجه کنید، همانند نمایش اعداد مثبت، هنگامیکه از ۱- تا ۱۵- میرویم، انگار ALU ۰۰۰۰۱ را از نمایش عدد کم میکند. لذا منطقی است که ۱۰۰۰۰ را معادل ۱۵-۱- (۱۰۰۰۱-۰۰۰۰۱)، یعنی ١٦- بگیریم.

در فصل ۵ کامپیوتری را به شما معرفی میکنیم که آنرا LC-3 (مخفف Little Computer ۳) نام گذاری کرده ایم. LC-3 بر روی اعداد ١٦ بیتی کار میکند. لذا اعداد صحیح مکمل ۲ در آن میتوانند بین ۳۲۷٦۸- و ۳۲۷٦۷ را نمایش دهند.

۲-٤- تبدیل اعداد باینری به دسیمال و بالعکس

معمولا تبدیل اعداد از باینری به دسیمال ویا دسیمال به باینری، نمایشی است که ما روزمره آنرا استفاده میکنیم.

۲-٤-۱- تبدیل اعداد باینری به دسیمال

جهت سهولت نمایش، اعداد ۸ بیتی مکمل ۲ را در نظر میگیریم که نمایش اعداد ۱۲۷ تا ۱۲۸- دسیمال میباشند. همانطور که قبلا بیان شد، یک عدد مکمل ۲ هشت بیت را میتوان بصورت زیر نمایش داد

$$a_7a_6a_5a_4a_3a_2a_1a_0$$

که هر بیت a_i میتواند ۰ یا ۱ باشد. تبدیل اعداد باینری به دسیمال به روش زیر انجام میگیرد:

۱. بیت آخر (سمت چپ و یا a_7) را بررسی کن.
   a. اگر ۰ بود عدد مثبت است و میتوان مقدار آنرا را بررسی کرد.
   b. اگر بیت آخر ۱ بود آنگاه عدد منفی میباشد. اول باید مکمل دو آنرا حساب نمود که معادل مثبت مقدار عدد را بما بدهد.

۲. مقدار عدد بسادگی از فرمول زیر بدست میاید:

$$a_6.2^6 + a_5.\ 2^5 + a_4.\ 2^4 + a_3.\ 2^3 + a_2.\ 2^2 + a_1.\ 2^1 + a_0.\ 2^0$$

که این بسادگی با جمع توان ۲ ها که ضریب ۱ دارند حاصل میگردد.

۳. در نهایت، اگر عدد اولیه منفی بود، یک علامت منفی به عدد اضافه میگردد.

مثال ۲-۲- : عدد مکمل ۲ی ۱۱۰۰۰۱۱۱ را به مقدار معادل دسیمال آن تبدیل کنید:

چون بیت سمت چپ ۱ است، عدد منفی میباشد. لذا اول مکمل دو آنرا بدست میآوریم، یعنی معکوس کرده و با یک جمع میکنیم (۰۰۱۱۱۰۰۱).
مقدار عدد را بروش زیر محاسبه میکنیم

$$0.\ 2^6+1.\ 2^5 +1.\ 2^4 +1.\ 2^3 +0.\ 2^2 +0.\ 2^1 +1.\ 2^0$$

یا

۳۲+۱۶+۸+۱

عدد دسیمال معادل عدد باینری ۱۱۰۰۰۱۱۱ عدد ۵۷- میباشد.

۲-٤-۲- تبدیل دسیمال به باینری

تبدیل عدد دسیمال به باینری کمی سخت تر از تبدیل باینری به دسیمال میباشد. نکته کلیدی اینست که اگر عدد مثبت باینری فرد است، آخرین بیت سمت راست آن یک میباشد و اگر زوج است، آخرین بیت سمت راست صفر خواهد بود.
مجددا نمایش ۸ بیتی را، برای محاسبه معادل دسیمال، در نظر بگیرید:

$$a_7.2^7+ a_6.\ 2^6+ a_5.\ 2^5 +a_4.\ 2^4 +a_3.\ 2^3 +a_2.\ 2^2 +a_1.\ 2^1 + a_0.\ 2^0$$

برای توضیح بهتر روش، اجازه دهید از یک مثال استفاده کنیم. فرض کنید میخواهیم عدد ۱۰۵+ را به مکمل ۲ آن تبدیل کنیم. توجه فرمایید که ۱۰۵+ عدد مثبت است. اول a_iها را برای محاسبه مقدار مکمل دو بدست میاوریم. از آنجایی که عدد مثبت است، بعد از محاسبه مقدار بیت a_7 را، که میدانیم ۰ است، به مقدار اضافه میکنیم.

قدم اول اینست که a_i هایی را بدست آوریم که در معادله زیر درست باشند:

$$105= a_6.\ 2^6+ a_5.\ 2^5 +a_4.\ 2^4 +a_3.\ 2^3 +a_2.\ 2^2 +a_1.\ 2^1 + a_0.\ 2^0$$

چون ۱۰۵ عدد فرد است، a_0 مساوی یک خواهد بود. با کم کردن ۱ از معادله بالا، به معادله زیر میرسیم:

$$104= a_6.2^6+ a_5.2^5 +a_4.2^4 +a_3.2^3 +a_2.2^2 +a_1.2^1$$

بعد هر دو سمت معادله بالا را تقسیم بر ۲ مینماییم که به معادله زیر ختم میگردد:

$$52= a_6.\ 2^5 +a_5.\ 2^4 +a_4.\ 2^3 +a_3.\ 2^2 +a_2.\ 2^1 + a_1.\ 2^0$$

۵۲ عدد زوج است و در نتیجه a_1 ، تنها ضریبی که با عددی بتوان ۲ ضرب نشده است، مساوی صفر خواهد بود.
حال این روش را ادامه داده و هر بار عدد سمت راست را از هر طرف معادله کم نموده و سپس بر دو تقسیم مینماییم. در انتها ببینید که آیا عدد دسیمال سمت چپ زوج است یا فرد. برای مثال، ادامه مثال بالا بصورت زیر خواهد بود:

$$52= a_6.\ 2^5 +a_5.\ 2^4 +a_4.\ 2^3 +a_3.\ 2^2 +a_2.\ 2^1$$

تقسیم بر دو معادله بالا به معادله زیر منتهی میگردد:

$$26= a_6.\ 2^4 + a_5.\ 2^3 +a_4.\ 2^2 +a_3.2^1 +a_2.2^0$$

لذا $a_2=$۰ و قدم بعدی:

$$13= a_6.2^3 +a_5.2^2 +a_4.2^1 +a_3.2^0$$

لذا $a_3=1$ و قدم بعدی:

$$6= a_6.2^2 +a_5.2^1 +a_4.2^0$$

لذا $a_4=$۰ و قدم بعدی:

$$3= a_6.2^1 +a_5.2^0$$

لذا $a_5=1$ و قدم بعدی:

$$1= a_6.2^0$$

لذا $a_6=$۰ و عمل تبدیل کامل شده است. نمایش باینری، مکمل دو، معادل عدد ۱۰۵+ برابر ۰۱۱۰۱۰۰۱ میباشد.

بهتر است روش تبدیل را خلاصه کرده، یا به بیان دیگری نشان دهیم. اگر یک عدد صحیح دسیمال N داده شده باشد، عدد باینری مکمل ۲ آن به روش زیر ساخته میشود:

ابتدا نمایش باینری مقدار N را از معادله ساخته شده زیر بدست می آوریم.

$$N= a_6.2^6+a_5.2^5+a_4.2^4+a_3.2^3+a_2.2^2+a_1.2^1+a_0.2^0$$

۱. m=N
۲. m را تقسیم بر دو کنید.
۳. باقی مانده تقسیم را به مجموعه بیتها از سمت چپ اضافه کنید.
٤. m را مساوی خارج قسمت قرار دهید.
۵. اگر m مخالف صفر بود به مرحله ۳ بروید.
٦. صفر به آخرین بیت سمت چپ اضافه کنید.
۷. اگر N مثبت بود، عدد بدست آمده و به مرحله ۱۰ بروید.

۸. اگر N منفی بود، مکمل دو عدد را بدست آورید (با معکوس کردن تمامی بیتها و اضافه کردن یک به آن).
۹. پایان.

۲-۵- عملیات بر روی بیتها- قسمت اول: عملیات ریاضی

۲-۵-۱- جمع وتفریق

عملیات ریاضی بر روی اعداد باینری مکمل ۲ خیلی شبیه به عملیات ریاضی بر روی اعداد دسیمال است که شما به آن عادت دارید.

جمع دو عدد از سمت راست به چپ انجام میگیرد که هر بار دو رقم با هم جمع میشوند. در هر مرحله یک نتیجه جمع و یک رقم انتقالی تولید میگردد. در اعداد باینری، بجای ایجاد رقم انتقالی بعد از ۹ (چرا که ۹ بزرگترین رقم دسیمال است)، رقم انتقالی بعد از ۱ تولید میگردد (۱ بزرگترین رقم باینری میباشد).

مثال ۲-۳: بر مبنای نمایش ۵ بیتی، جمع اعداد ۱۱ و ۳ چیست؟

نمایش مکمل دو عدد ۱۱: ۰۱۰۱۱

نمایش مکمل دو عدد ۳: ۰۰۰۱۱

جمع دو عدد، که مساوی ۱٤ است ۰۱۱۱۰

تفریق دو عدد همان جمع یکی با منفی عددی است که کم میشود. به بیان دیگر A-B مساوی A+(-B) میباشد.

مثال ۲,٤: نتیجه ۹-۱٤ چیست؟

نمایش مکمل ۲ عدد ۱٤: ۰۱۱۱۰

نمایش مکمل ۲ عدد ۹: ۰۱۰۰۱

اول باید مکمل دو ۹- را بدست آورد: ۱۰۱۱۱

جمع ۱٤ با ۹- میشود: ۰۱۱۱۰

۱۰۱۱۱

----- ---

که مساوی ۵ میباشد. ۰۰۱۰۱

توجه کنید که رقم انتقالی انتهایی در نظر گرفته نمیشود.

مثال ۲-۵ : نتیجه جمع یک عدد با خودش چیست (مثلاً x+x)؟

فرض کنید مثال زیر نمایش باینری عدد هشت بیتی است که به ما اجـازه میدهـد از ۱۲۸- تـا ۱۲۷ را نمایش دهیم. فرض کنید x مساوی ۵۹ است که بـا ۰۰۱۱۱۰۱۱ نمـایش داده میشـود. اگـر ۵۹ را بـا خودش جمع نماییم، معادل ۰۱۱۱۰۱۱۰ میشود. همانطوریکه مشاهده میکنید، نتیجه جمع همان عـدد اولیه است با این تفاوت که تمامی بیتها یک رقم به سمت چپ شیفت کرده اند. آیا این تصادفی است و یا اگر جمع دو عدد از حد قابل نمایش، مثلاً ۱۲۷ در نمایش ۸ بیتی، بیشتر نشود اتفاق می افتد؟

اگر نمایش مکانی را استفاده کنیم، عدد ۵۹ بصورت زیر خواهد بود:

$$0.2^6+ 1.2^5 +1.2^4 +1.2^3 +0.2^2 +1.2^1 + 1.2^0$$

جمع ۵۹+۵۹ هم مساوی ۵۹×۲ میباشد که در نمایش ما معادل:

$$2.(0.2^6+ 1.2^5 +1.2^4 +1.2^3 +0.2^2 +1.2^1 + 1.2^0)$$

میباشد. به بیان دیگر ۵۹+۵۹ معادل

$$0.2^7+1.2^6+1.2^5+1.2^4+0.2^3+1.2^2+1.2^1$$

میباشد که هر بیت را به سمت چپ شیفت می نماید. لذا جمع یک عدد با خودش (اگر تعداد مورد نیاز بیت برای نمایش آن وجود داشته باشد) برابر با انتقال دادن بیتهای عدد یکبار به سمت چپ است.

۲-۵-۲- بسط علامت

در بسیاری از موارد مناسب است که یک عدد کوچک را با بیتهای کمتری نمایش داد. برای مثال، بجای نمایش دادن عدد ۵ با ۰۰۰۰۰۰۰۰۰۰۰۰۰۱۰۱، مناسب است که آنرا با ٦ بیت بصورت ۰۰۰۱۰۱ نمایش داد. همانطور که در اعداد دسیمال اضافه کردن صفر در سمت چپ عدد تاثیری در مقدار آن ندارد و مقدارها یکی هستند (مثلاً ۱۲٤۵ تومان با ۰۰۰۰۱۲٤۵ تومان یکی میباشند) صفرهای سمت چپ در نمایش باینری نیز تاثیری در مقدار عدد ندارند.

در مورد اعداد منفی چطور؟ برای بدست آوردن اعداد منفی، اعداد مثبت آنها را گرفته و مکمل (معکوس) آنها را حساب کرده سپس نتیجه مکمل را با عدد ۱ جمع می نماییم. مثلاً نمایش ۵- بر مبنای نمایش ۵ که ۰۰۰۱۰۱ میباشد معادل ۱۱۱۰۱۱ است. لذا اگر ۵ بصورت ۰۰۰۰۰۰۰۰۰۰۰۰۰۱۰۱ نمایش داده شود، ۵- با ۱۱۱۱۱۱۱۱۱۱۱۱۱۰۱۱ نمایش داده میشود. همانگونه که صفرهای سمت چپ در عدد مثبت تاثیری نداشته و مقدار آن را تغییر نمیدهد، یکهای اعداد منفی نیز در مقدار عدد منفی تاثیری ندارد.

حال اگر بخواهیم دو عدد را با هم جمع نماییم، لازم است دو عدد با تعداد بیتهای یکسان نمایش داده شوند. مثلاً برای جمع عدد ۱۳ با ۵-، که ۱۳ با ۰۰۰۰۰۰۰۰۰۰۰۰۱۱۰۱ و عدد ۵- با ۱۱۱۰۱۱، نمایش

داده شده اند، جمع دو عدد که با تعداد بیت یکسان نمایش داده نشده اند بصورت زیر خواهد بود:

۰۰۰۰۰۰۰۰۰۰۰۰۱۱۰۱

\+ ۱۱۱۰۱۱

———————————

اگر بخواهیم این دو را جمع کنیم، با بیتهای نمایش داده نشده در عدد ۵- چه باید بکنیم؟ اگر نبود بیت ها را مساوی ۰ بگیریم، آنگاه دیگر عدد ۵- را با ۱۳ جمع نمیکنیم. به بیان دیگر اگر بیتهای خالی را با ۰ پر کنیم، عدد حاصل دیگر ۵- نخواهد بود، بلکه نمایش حاصل ۰۰۰۰۰۰۰۰۰۰۱۱۱۰۱۱ میشود که مساوی عدد ۵۹+ است. طبیعی است که نتیجه عمل جمع هم مساوی ۷۲ شده و جمع اشتباه است. ولیکن، اگر دقت کنیم که نمایش ٦ بیتی ۵- و نمایش ١٦ بیتی آن فقط در تعداد یکهای سمت چپ تفاوت کنند، برای انجام جمع، ابتدا ۵- را به ١٦ بسط میدهیم. لذا خواهیم داشت:

۰۰۰۰۰۰۰۰۰۰۰۰۱۱۰۱

+۱۱۱۱۱۱۱۱۱۱۱۱۱۰۱۱

———————————

۰۰۰۰۰۰۰۰۰۰۰۰۱۰۰۰

و نتیجه عمل همانگونه که انتظار می رفت ۸+ است.

همانطور که مقدار اعداد مثبت با اضافه کردن هر تعداد بیت علامت، یعنی همان ۰، تغییر نمیکند به همین ترتیب مقدار اعداد منفی با اضافه کردن هر تعداد بیت علامت، یعنی همان ۱، به سمت چپ آن تغییر نخواهد کرد. چون در هر دو مورد، بیت علامت بسط داده شده است، به این

عمل بسط علامت[۵] میگوئیم. بسط علامت جهت انجام عملیات بر روی رشته بیتها با سایز مختلف انجام میگیرد و مقدار اعداد نمایش داده شده را تغییر نمیدهد.

۲-۵-۳- سرریز کردن (Overflow)

تا اینجا ما همواره با این فرض بودیم که جمع دو عدد صحیح،به اندازه کافی کوچک است که قابل نمایش با تعداد بیت موجود باشد. لیکن این همواره صحیح نیست و جمع دو عدد ممکن است بیشتر از مقدار قابل نمایش باشد. در اینصورت چه اتفاقی خواهد افتاد؟

بدون شک شما با کیلومتر شمار ماشین آشنا هستید. کیلومتر شمار کل مسافت طی شده توسط ماشین را تا یک حد مشخص نگهداری میکند. در دوران گذشته، کیلومتر شمارها میتوانستند تا ۹۹۹۹۹ کیلومتر را ثبت نمایند. حال اگر شما با ماشینتان ۱۰۰۰۹۲ رانندگی میکردید، آنگاه کیلومتر شمار مقدار ۰۰۰۹۲ کیلومتر را نشان میداد گویی ماشین شما کاملا نو است! مشکل محدودیت کیلومتر شمار در نمایش مسافت طی شده بود. لذا عدد ۱۰۰۰۹۲ بصورت ۰۰۰۹۲ نمایش داده میشد. در واقع رقم انتقالی صد هزارتایی از دست میرفت.

در این حالت گفته میشود که کیلومتر شمار سرریز کرده است[۶]. سر ریز[۷] همان رقم انتقالی از بزرگترین رقم میباشد. واضح است که نمایش ۱۰۰۰۹۲ بصورت ۰۰۰۹۲ قابل قبول نیست. بدین سبب، از آنجایی که ماشینهای بیشتری از مرز ۹۹۹۹۹ کیلومتر گذشتند، شرکتهای ماشین سازی رقم دیگری به کیلومترشمار اضافه کردند. این روزها ماشینها بعد از ۱۰۰۰۰۰۰ کیلومتر سرریز میکنند بجای ۱۰۰۰۰۰ کیلومتر.

کیلومتر شماری مثالی از عملیات ریاضی بر روی اعداد بدون علامت میباشد. کیلومترهایی که ماشین راه رفته است همواره مثبت میباشد. لذا اگر کیلومتر شمار ۰۰۰۱۲۹ را نمایش دهد وشما ۵۰ کیلومتر

---

۵ Sign-EXTension یا بطور مختصر SEXT

۶ Overflowed

۷ overflow

رانندگی کنید، کیلومتر شمار ۰۰۰۱۷۹ را نمایش خواهد داد.

در مورد اعداد علامت دار، مخصوصا اعداد صحیح مکمل ۲، سرریز کردن مقداری مهمتر مـی باشـد. حال به همان مثال ۵ بیتی خود، که امکان نمایش ۱۵– تا ۱۵ منفی را به مـا میدهـد، بـاز مـی گـردیم. فرض کنید میخواهیم اعداد ۹+ و ۱۱+ را جمع نماییم. عمل جمع بصورت زیر خواهد بود:

۰۱۰۰۱

۰۱۰۱۱

____

۱۰۱۰۰

توجه نمایید که نتیجه جمع از ۱۵ بزرگتر است لـذا قابـل نمـایش توسـط روش مکمـل ۲ی ۵ بیتـی نمیباشد. اینکه عدد بزرگتر از حد قابل نمایش است بدان معناست که عدد از ۰۱۱۱۱، بزرگترین عـدد قابل نمایش در روش مکمل ۲ی ۵ بیتی، بزرگتر است. توجه نمایید که چون عـدد مثبـت حاصـل از جمع، بزرگتر از ۱۵+ است، رقم انتقالی ایجاد شده به بیت منتهی الیه سمت چپ میرود. . ولـی ایـن بیت اختصاص به علامت عدد دارد. بدین ترتیب به راحتی میتوان تشخیص داد که چه هنگام سـرریز اتفاق میافتد. چون ما دو عدد مثبت را جمع کردیم، نتیجه باید یک عدد مثبت باشد. چون ALU یـک عدد منفی تولید کرد، میفهمیم که مشکلی وجود دارد و آن اینست که نتیجه جمع دو عدد بیش از حد قابل نمایش بوده و سرریز اتفاق افتاده است.

حال فرض کنید که دو عدد منفی را بـا یکـدیگر جمـع نمـاییم، مـثلاً ۱۲– و ٦–. عمـل جمـع بصورت زیر انجام میگیرد:

۱۰۱۰۰

۱۱۰۱۰

۰۱۱۱۰

اینجا هم سرریز اتفاق میافتد چرا که جمع ۱۲- و ٦- مساوی ۱۸- است که از حد توان نمایش مکمل ۲ی ۵ بیتی، یعنی ١٦-، بیشتر است. در نتیجه ALU یک عـدد مثبـت تولیـد میکنـد. مجـدداً براحتـی میتوان سرریز را تشخیص داد چرا که جمع دو عدد منفی نباید مثبت گردد.

توجه کنید که جمع دو عدد منفی و مثبت نمیتواند سرریز ایجاد نماید، چرا؟ این مسئله در تمرین ۲-۲۵ به شما واگذار شده است.

۲-٦- عملیات بر روی بیتها- قسمت دوم: عملیات منطقی

تا کنون دیدیم که میتوان اعمال ریاضی (همانند جمع و تفریق) را بر روی رشته های بـاینری اعمال نمود. مجموعه دیگری از عملیات مورد استفاده بر روی رشته هـای بـاینری، عملیـات منطقـی میباشند.

عملیات منطقی بر روی متغیرهای منطقی اعمال میشوند. یک متغیر منطقی میتواند مقادیر ۰ یا ۱ را بـه خود بگیرد. واژه *منطقی* تاریخچه خود را دارد و از آنجا آمده است که مقادیر ۰ و ۱ میتواننـد نمـایش دو حالت منطقی غلط و صحیح باشند. لیکن استفاده از عملیـات منطقـی از حـد ایـن تعریـف اولیـه گذشته و کاربردهای بسیاری دارد.

تعدادی توابع منطقی وجود دارند و عمده ALUها تمامی این توابع را اجرا می نمایند.

۲-۶-۱- تابع " و" (AND)

AND یک تابع منطقی میباشد بدان معنا که دو ورودی و یک خروجی خواهد داشت. به بیان دیگر، AND نیاز به دو ورودی جهت انجام عملیات دارد. هر ورودی یک متغیر منطقی میباشد که میتواند صفر یا یک باشد. خروجی AND فقط در حالتی یک است که دو ورودی آن یک باشند. در غیر اینصورت خروجی AND صفر خواهد بود.

یک مکانیزم راحت برای نشان دادن عملیات منطقی، جدول صحت میباشد. یک جدول صحت دارای n+۱ ستون و ۲^n سطر میباشد. n ستون اول جدول صحت اختصاص به n ورودی دارد. چون هر ورودی یک تابع منطقی فقط میتواند دو مقدار ۰ یا ۱ را بگیرد، پس این n ورودی میتوانند ۲^n مقدار منحصر بفرد بخود بگیرد. هر کدام از این ۲^n مقدار منحصر بفرد، که گاهاً یک ترکیب ورودی خوانده میشوند، در یک سطر جدول صحت نمایش داده میشود. ستون آخر جدول صحت اختصاص به خروجی ناشی از هر ترکیب ورودی دارد.

در مورد تابع AND با دو ورودی، جدول صحت دو ستون برای ورودی ها، و ٤ سطر ($۲^۲$) برای ترکیبهای ورودی ها دارد.

A	B	AND
۰	۰	۰
۰	۱	۰
۱	۰	۰
۱	۱	۱

عملیات منطقی AND را روی هر جفت ورودی m بیتی می توان اعمال کرد. به عنوان مثال (مثال۲–۶) اگر a و b هرکدام نمونه های ۱۶ بیتی باشند و c نتیجه AND کردن a و b باشد، به این عملیات AND کردن بیتی[8] گویند.

مثال ۲–۶ :

اگر c نتیجه AND کردن a و b باشد و a=0011101001101001، b=0101100100100001 مقدار c چه خواهد بود؟

AND دو ورودی a و b را بصورت عمل بر روی بیتهای ورودی انجام میدهیم. به بیان دیگر هر جفت بیت ورودی a_i و b_i را با یکدیگر AND کرده تا بیت c_i را بسازیم. برای مثال، چون $a_0 =1$ و $b_0=1$ در نتیجه c_0 که نتیجه AND کردن a_0 و b_0 است مساوی ۱ خواهد بود. نتیجه کلی بصورت زیر خواهد بود.

```
a: 0011101001101001
b: 0101100100100001
   ------------------------
c: 0001100000100001
```

مثال ۲–۷: فرض کنید یک رشته ۸ بیتی، که با A نمایش داده شده است، داریم که دو بیت سمت راست آن دارای اهمیت خاصی میباشند. رایانه باید بر مبنای وضعیت این دو بیت یکی از چهار عمل مشخص شده را انجام دهد. آیا ما میتوانیم ایندو بیت را مجزا کنیم تا بتوان برای تعیین عمل مورد نیاز استفاده گردند؟

بله ما میتوانیم اینکار را با استفاده از یک الگو[9] () انجام دهیم. یک الگوی بیتی[10] الگوی باینری است که اجازه میدهد تا بیتهای A به دو دسته تقسیم گردند: بیتهایی که برای ما اهمیت دارند و بیتهایی که

---

[8] bitwise

[9] mask

[10] bit mask

اهمیت ندارند. در مورد این مثال، اگر الگوی بیتی ۰۰۰۰۰۰۱۱ با A AND گردد، بیتهای ۲ تا ۷ را تبدیل به ۰ کرده و مقادیر بیتهای ۱ و ۰ را معادل بیتهای ۱ و ۰ A میگذارد. در اینحالت گفته میشود که الگوی بیتی بیتهای ۲ تا ۷ را پوشانده است.

اگر A مساوی ۰۱۰۱۰۱۱۰ باشد و آنرا با الگوی بیتی ۰۰۰۰۰۰۱۱ AND نماییم، مقدار ۰۰۰۰۰۰۱۰ حاصل میگردد. اگر A مساوی ۱۱۱۱۱۱۰۰ باشد و با همان الگوی بیتی AND نماییم، نتیجه ۰۰۰۰۰۰۰۰ خواهد بود.

به بیان دیگر نتیجه AND کردن هر رشته ۸ بیتی با الگوی بیتی ۰۰۰۰۰۰۱۱، یکی از چهار رشته ۰۰۰۰۰۰۰۰، ۰۰۰۰۰۰۰۱، ۰۰۰۰۰۰۱۰ و ۰۰۰۰۰۰۱۱ خواهد بود. نتیجه AND کردن با الگوی بیتی مشخص نمودن دو بیت مورد نظر میباشد.

۲-۶-۲- تابع یا (OR)

OR هم یک تابع منطقی باینری میباشد. این تابع نیاز به دو ورودی دارد که هر کدام یک متغیر منطقی میباشد. خروجی OR، ۱ است اگر یکی از دو ورودی یک و صفر است اگر هر دو ورودی صفر باشند. جدول صحت تابع OR را با دو ورودی در زیر نشان داده شده است:

A	B	OR
0	0	0
0	1	1
1	0	1
1	1	1

بهمان صورت که عمل منطقی AND را بر روی دو رشته m بیتی انجام میدهیم، میتوانیم عمل OR را

بر روی دو رشته ورودی m بیتی انجام دهیم.

مثـــال ۲-۸ : اگـــر c نتیجـــه OR کـــردن a و b باشـــد (a=0011101001101001، b=0101100100100001) ، مقدار c چه خواهد بود؟

OR دو ورودی a و b را بصورت عمل بر روی بیتهای ورودی انجام میدهیم. به بیان دیگر هر جفت بیت ورودی a_i و b_i را با یکدیگر OR کرده تا بیت c_i را بسازیم. برای مثال، چون $a_0=1$ و $b_0=1$ در نتیجه c_0 که نتیجه OR کردن a_0 و b_0 است مساوی ۱ خواهد بود. بهمین ترتیب نتیجه OR کردن $a_6=1$ و $b_6=0$ مساوی ۱ خواهد بود.نتیجه کلی بصورت زیر خواهد بود.

a: 0011101001101001

b: 0101100100100001

----

c: 0111101101101001

در بعضی مواقع به عمل OR عمل OR فراگیر (inclusive-OR) گفته میشود تا بتـوان آنـرا از عمـل منطقی OR انحصاری (exclusive-OR)، که بزودی آنرا بحث خواهیم کرد، متمایز نمود.

۲-۶-۳- تابع "نه" (NOT)

NOT یک تابع منطقی یکانی[۱۱] است. این بدان معناست که تابع فقط یـک ورودی دارد. عمـل NOT همچنین به عمل مکمل[۱۲] معروف است. بعضی مواقع میگوییم که خروجی با معکـوس کـردن ورودی حاصل میگردد. اگر ورودی ۱ است، خروجی صفر خواهـد بـود و اگـر ورودی صـفر اسـت خروجی یک خواهد بود. جدول صحت تابع NOT بصورت زیر میباشد:

[۱۱] unary

[۱۲] complement

A	NOT
0	1
1	0

به همان روشی که عملیات منطقی AND و OR را بر روی دو رشته m بیتـی اعمـال میکنـیم، عمـل NOT را هم روی یک رشته m بیتی اجرا میکنیم. اگر a همان مقدار بالا باشد، c نتیجه NOT کـردن a بصورت زیر است.

a: 0011101001101001

----

c: 1100010110010110

۲-۶-۴- تابع انحصاری "یا" (Exclusive-OR)

تابع انحصاری OR، که با XOR نمایش داده میشود، یک تابع منطقی باینری میباشد. ایـن تـابع هم نیاز به دو ورودی دارد که هر دو ورودی متغیرهای منطقی میباشند. خروجی این تابع ۱ است اگر دو ورودی با یکدیگر متفاوت باشند. اگر ورودی ها مشابه باشند آنگاه خروجی صـفر اسـت. جـدول صحت این تابع بصورت زیر است:

A	B	XOR
۰	۰	۰
۰	۱	1
۱	۰	1

۱	۱	0

همانگونه که عملیات منطقی AND و OR را بر روی دو رشته m بیتی اعمال میکنیم، میتوانیم عمل XOR را بر دو رشته m بیتی اعمال کنیم.

مثال ۲-۹ : اگر a و b همان دو رشته بیت مثال ۲,٦ باشند آنگاه مقدار c که نتیجه XOR کردن a و b میباشد بصورت زیر خواهد بود.

a: 0011101001101001

b: 0101100100100001

----

c: 0110001101001000

توجه به تفاوت جدول صحت XOR و OR نمایید. در مورد XOR اگر دو ورودی ۱ باشند، خروجی صفر است. به بیان دیگر اگر ورودی اول ۱، دومی ۰ و یا اولی ۰، دومی ۱باشد، آنگاه خروجی یک است. لغت انحصاری[۱۳] بدین دلیل استفاده شده است که خروجی در صورتی ۱ است که فقط یکی از ورودیها ۱باشد. این در حالیست که در OR خروجی ۱ است اگر یکی از دو ورودی یا هر دو ۱ باشند. به همین علت است که لغت در بر گیرنده[۱٤] استفاده شده است.

مثال ۲-۱۰ :

فرض کنید که میخواهیم ببینیم که آیا دو رشته بیت یکی هستند یا خیر.از آنجا که تابع XOR فقط در حالتی ۰ تولید میکند که ورودیها یکی باشند ، میتوان گفت اگر XOR کردن دو رشته مساوی ۰

---

[۱۳] exclusive

[۱۴] inclusive

بود، آنگاه دو رشته یکی هستند. در غیر اینصورت دو ورودی متفاوت میباشند.

۲-۷- نمایشهای دیگر

چهار نمایش اطلاعات دیگر وجود دارند که در کار ما مورد استفاده واقع میشوند که عبارتند از بردار بیتها، اعداد شناور، کد اسکی ASCII و نمایش شانزده تایی (هگزادسیمال).

۲-۷-۱- بردار بیت

بردار بیت عمدتاً برای نمایش سیستمهایی که از چندین واحد تشکیل شده اند، و هر کدام به تنهایی و جدا از دیگران مشغول یا در دسترسند ، استفاده میشود. این سیستم میتواند مانند کارخانه ای باشد که هر واحد آن یک ماشین مخصوص است یا شبکه تاکسیرانی باشد که هر واحد آن یک تاکسی است. در هر کدام از اینها مهم این است که بتوان تعیین کرد که کدام واحد مشغول است و کدام واحد آزاد تا بتوان در صورت نیاز کار به آن اختصاص داد.

حال فرض کنید n واحد به این سبک داریم. برای بررسی این n واحد از یک رشته بیت n تایی، که بردار بیت گفته میشود، استفاده می نماییم. چنانچه واحدی آزاد باشد بیت متناظر آن برابر ۱ و در صورت مشغول بودن برابر صفر می باشد.

مثال ۲-۱۱ : فرض کنید که ۸ ماشین تراش در یک کارخانه داریم و میخواهیم که آنها را بر مبنای آزاد یا خالی بودن تحت نظر داشته باشیم. ما میتوانیم آنها را با یک بردار ۸ بیتی، به نام بردار بیتی اشتغال، تحت نظر داشته باشیم. در این بردار،۱ نمایش آزاد بودن و ۰ نمایش مشغول بودن ماشین میباشد. بیتها از سمت راست به چپ با ۰ تا ۷ مشخص شده اند.

بردار بیتی اشتغال ۱۱۰۰۰۰۱۰ نشان از شرایطی است که ماشین تراشهای ۱و ٦و ۷ آزاد هستند و در نتیجه میتوان به آنها کار اختصاص داد. فرض کنید کاری به ماشین تراش ۷ اختصاص داده شود. در

نتیجه بردار بیتی اشتغال باید با AND کردن با الگوی بیتی ۰۱۱۱۱۱۱۱ به روز گردد. هدف الگوی بیتی آنست که بیت ۷ را از بردار بیت اشتغال پاک کند که در نتیجه برداربیتی اشتغال معادل ۰۱۰۰۰۰۱۰ میشود.

توجه داشته باشید که ما قبلاً با مفهوم الگوی بیتی در مثال ۲–۷ برخورد داشتیم. بیاد آورید الگوی بیتی قادر است تعدادی از بیتها را برای بررسی انتخاب کرده و بقیه را در نظر نگیرد. در این مثال، بیت ۷ به صفر تبدیل شده و بقیه بیتها، بیتهای ٦ تا ۰، در نظر گرفته نشده اند.

فرض کنید واحد ۵ کار خود را تمام کرده و در حالت بی کار قرار میگیرد. ما میتوانیم بردار بیتی اشتغال را به روز کرده با انجام عمل OR منطقی با الگوی بیت ۰۰۱۰۰۰۰۰، نتیجه عمل ۰۱۱۰۰۰۱۰ خواهد بود.

۲-۷-۲- اعداد شناور (floating point)

بیشتر اعمال ریاضی که ما در این کتاب انجام میدهیم بر روی اعداد صحیح میباشند. برای مثال، LC-3 داده های مکمل ۲ی ١٦ بیتی را استفاده میکند. با ١٦ بیت، میتوان اعداد بین ۳۲۷٦۸– تا ۳۲۷٦۷ ($-۲^{۱۵}$ تا $۲^{۱۵}-۱$) را بر مبنای مکمل دو نمایش داد. در این حالت میگوییم که دقت نمایش ما به اندازه ۱۵ بیت میباشد و محدوده دامنه آن $۲^{۱۵}$ است. همانطور که در درس فیزیک یا شیمی یاد گرفته اید، در بعضی موارد، ما نیاز به نمایش اعدادی بسیار بزرگتر داریم در حالیکه این تعداد رقم دقت نیاز نداریم. برای مثال عدد 6.023×10^{23} را بخاطر آورید که ممکن بود نیاز به حفظ کردن آن میداشتید. این عدد بسیار بزرگتر از $۲^{۱۵}$ است که با ١٦ بیت اعداد صحیح مکمل ۲ قابل نمایش باشد. از جهت دیگر ۱۵ بیت موجود در نمایش اعداد صحیح مکمل ۲ی ١٦ بیتی بیش از دقت مورد نیازمیباشد. به بیان دیگر ما فقط نیاز به تعداد بیت کافی برای نمایش مهمترین چهار رقم دسیمال داریم، یعنی ٦۰۲۳.

لذا ما در اینجا مسئله زیر را داریم. تعداد بیت موجود بیش از نیاز به دقت در اعداد میباشد در

حالیکه تعداد بیتهای مورد نیازبرای نمایش دامنه اعداد کم میباشد. روش نمایش اعداد شناور راه حلی برای این مسئله میباشد. بجای استفاده از تمامی بیتها (بجز بیت علامت) برای نمایش مقدار عـدد، در نمایش شناور اعداد، تعدادی از بیتها برای نمایش دامنه (یعنی بزرگی یا کـوچکی عـدد) اعـداد بکـار گرفته میشوند. بقیه بیتها برای دقت استفاده خواهند شد.

اغلب ساختارهای دستورالعمل ها (ISA) در رایانه های کنونی بیش از یک نوع عـدد شـناور تعریـف میکنند. یکی از این تعاریف که معمولا float خوانده میشود، از۳۲ بیـت تشـکیل شـده اسـت و بیتهـا بصورت زیر اختصاص داده شده اند:

۱ بیت برای علامت (مثبت یا منفی)

۸ بیت برای دامنه (توان)

۲۳ بیت برای دقت (قسمت اعشار)

1	← 8 bits →	← 23 →
S	exponent	fraction

$$N=(-1)^{s} \ x \ 1.fraction \ x \ 2^{exponent-127}$$

$$1 <= exponent <= 254$$

اغلب تولید کنندگان کنونی رایانه فرمول نشان داده شده در شکل ۲-۲ را برای محاسبه اعـداد شـناور ۳۲ بیت استفاده میکنند. این فرمول بخشی از استاندارد IEEE برای اعداد شناور میباشد.

همانطور که قبلا بیان شد، نمایش اعداد شناور خیلی شبیه به نمایش علمی اعداد، مثلاً $٦,٠٢٣ \times ١٠^{٢٣}$، است که در دبیرستان یاد گرفتید. این عدد سه قسمت دارد: علامت، که مثبت اسـت، ارقـام مهـم کـه ٦,٠٢٣ است و توان که ٢٣ میباشد. ما ارقام مهم را بعنوان اعشار (fraction) میخوانیم. توجه نمایید که قسمت اعشار بهنجار[۱۵] شده است، یعنی فقط یک رقم سمت چپ، قسمت اعشار میباشد.

فرمول و نمایش اعداد شناور نشان داده شده در شکل ۲-۲ هم شامل همین سه قسمت میباشد.

[۱۵] Normalized

قسمت اعشار متشکل از ۲۳ رقم باینری است. توجه کنید که قسمت اعشار بهنجار شده است و فقط یک رقم غیر صفر سمت چپ قسمت اعشار میباشد. چون عدد ۱ تنها عدد غیر صفر در نمایش باینری میباشد پس نیازی به نمایش آن نمیباشد. به همین علت فرمول شکل ۲-۲، ۲٤ بیت برای دقت عدد دارد که ۲۳ تای آن در سمت راست برای نمایش اعشار است و یک بیت سمت چپ یک است که نیاز به نمایش آن نیست.

برای نمایش توان، ۸ بیت در فرمول نمایش داده شده در شکل ۲-۲ بکار گرفته شده است. مبنای پایه هم در فرمول شکل ۲-۲ باینری (عدد ۲) میباشد. با ۸ بیت، میتوان ۲٥٦ توان را نمایش داد. توجه کنید که فرمول فقط مقادیر ۱ تا ۲٥٤ را نشان میدهد. اگر توان ۰۰۰۰۰۰۰۰ (یعنی ۰) و یا ۱۱۱۱۱۱۱۱ (یعنی ۲٥٥) باشد این فرمول کمکی به تفسیر بیتها نمی کند.

توان واقعی عدد شناور با تفریق عدد ۱۲۷ از عدد بدون علامت نشان داده شده توسط قسمت توان بدست میاید. بعنوان مثال اگر توان واقعی ۸+ است، آنگاه عدد نمایش داده شده در قسمت توان ۱۰۰۰۰۱۱۱ میباشد که معادل ۱۳٥ است. به بیان دیگر ۸=۱۲۷-۱۳٥. به همین ترتیب، اگر توان واقعی ۱۲٥- باشد، آنگاه قسمت توان ۰۰۰۰۰۰۱۰ باید باشد که معادل عدد بدون علامت ۲ میباشد (۱۲٥- = ۲-۱۲۷).

قسمت سوم بیت علامت است: ۰ برای اعداد مثبت و ۱ برای اعداد منفی. فرمول این قسمت را می توان با $(-1)^s$ نشان داد که مساوی ۱+ است اگر s=۰ و مساوی ۱- است اگر s=۱.

مثال ۲-۱۲ : عدد ٦ ٥/٨- را بصورت شناور نشان دهید.

اول عدد ٦ ٥/٨- بصورت باینری نشان میدهیم: ۱۱۰,۱۰۱-.

$$-(1.2^2+1.2^1+0.2^0+1.2^{-1}+0.2^{-2}+1.2^{-3})$$

سپس این عدد را بهنجار میکنیم که حاصل $-1.10101 \,.\, 2^2$ می شود. چون بیت علامت ۱ است که نشان دهنده منفی بودن عدد ٦ ٥/٨- می باشد. قسمت توان ۱۰۰۰۰۰۰۱ است که معادل عدد بدون علامت ۱۲۹ میباشد و توان واقعی را میتوان از تفاضل ۱۲۷ از ۱۲۹ (۲+= ۱۲۷ –

۱۲۹) بدست آورد.. در نهایت قسمت اعشار ۱۰۱۰۱۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰ باید باشد که مقدار سمت راست عدد بهنجار شده میباشد. پس معادل عدد شناور ۶ ۵/۸- بصورت زیر است:

1 10000001 10101000000000000000000

مثال ۲-۱۳ : عدد شناور زیر چه مقدار دسیمالی را نشان میدهد؟

0 01111011 00000000000000000000000

چون بیت سمت چپ، بیت علامت، صفر است، عدد باید مثبت باشد. ۸ بیت بعدی، عدد بدون علامت ۱۲۳ را نشان میدهد. اگر ۱۲۷ را از آن کم کنیم، توان واقعی ٤- حاصل میشود. ۲۳ بیت بعدی، که قسمت اعشار میباشد، صفر است. پس عدد شناور نشان داده شده معادل 2^{-4} . 1.00000000000000000000000+ و یا ۱/۱٦ میباشد.

همانطور که گفته شد، فرمول بیان شده در حالاتی صادق است که توان نه ۰۰۰۰۰۰۰۰ و نه ۱۱۱۱۱۱۱۱ باشد. استاندارد IEEE برای این حالات هم تعاریف مناسب را در نظر گرفته است و معادل آنها را بیان میکند. در این قسمت چند حالت خاص بررسی میگردند.

۱. اگر قسمت توان معادل ۰۰۰۰۰۰۰۰ باشد، عدد معادل نمایش شناور از فرمول زیر حساب میگردد:

$$N=(-1)^s \times 0.fraction \times 2^{-126}$$

بدین ترتیب میتوان اعداد خیلی کوچک را هم نشان داد. برای مثال، عدد شناور

0 00000000 00001000000000000000000

را میتوان بصورت زیر ترجمه کرد. بیت سمت چپ نشان دهنده مثبت بودن عدد است. ۸ بیت بعدی

صفر میباشند که نشان میدهند توان واقعی معادل ۱۲۶– است. ۲۳ بیت سمت راست نشان دهنده ۰,۰۰۰۰۱۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰ که معادل ۵–۲ میباشد. در نهایت عدد معادل عدد شناور بالا معادل:

$$2^{-5}.2^{-126}=2^{-131}$$

خواهد بود.

۲. اگر قسمت توان معادل ۱۱۱۱۱۱۱۱ (یعنی ۲۵۵) باشد و قسمت اعشار صفر باشد آنگاه عدد نمایش مثبت یا منفی بی نهایت میباشد. (اگر بیت علامت صفر باشد مثبت بی نهایت و اگر یک باشد منفی بی نهایت)

۳. اگر قسمت توان معادل ۱۱۱۱۱۱۱۱ (یعنی ۲۵۵) باشد و قسمت اعشار مخالف صفر باشد، یعنی این یک عدد نیست.

٤. اگر قسمت توان معادل ۰۰۰۰۰۰۰۰ (یعنی ۰) باشد و قسمت اعشار هم صفر باشد، آنگاه عدد نمایش صفر میباشد.

مثال ۲,۱٤: چهار مثال زیر نمونه های بیشتری از تفسیر و ترجمه نمایش اعداد شناورمطابق با استاندارد IEEE را نشان می دهند.

0 10000011 00101000000000000000000

معادل $1{,}00101{,}2^{4}$ است که ۱۸,۵ دسیمال میباشد. قسمت توان، عدد بدون علامت ۱۳۱ را نشان میدهد. توان واقعی از تفریق ۱۲۷ از ۱۳۱ یعنی معادل ٤ بدست میاید. با اضافه کردن ۱ در سمت چپ قسمت اعشار، عدد باینری ۱,۰۰۱۰۱ خواهد بود. با شیفت عدد باندازه ٤ بیت به سمت چپ عدد ۱۰۰۱۰,۱ حاصل میگردد که مساوی ۱۸,۵ میباشد.

1 10000010 00101000000000000000000

معادل $۲^{۳}$,۱,۰۰۱۰۱– و یا ۹,۲۵– میباشد. بیت علامت ۱ است که نشان میدهد عدد منفی است. هشت بیت بعدی نمایش قسمت توان، عدد بدون علامت ۱۳۰ را نشان می دهد. بدین ترتیب توان واقعی از تفاضل ۱۲۷ از ۱۳۰ حاصل میگردد که معادل ۳ میباشد. با گذاشتن ۱ سمت چپ قسمت اعشار، عدد باینری ۱,۰۰۱۰۱ بدست میاید. با سه بیت شیفت دادن این عدد به سمت چپ عدد باینری ۱۰۰۱,۰۱ حاصل میگردد که معادل عدد ۹,۲۵– است.

0 11111110 11111111111111111111111

که تقریبا مساوی $۲^{۱۲۸}$ میباشد. بیت سمت چپ، یعنی بیت علامت، صفر است که نشان دهنده مثبت بودن عدد است. توان مساوی ۲٥٤ است که اگر ۱۲۷ را از آن کم کنیم، توان واقعی ۱۲۷ حاصل میگردد. با قرار دادن ۱ در سمت چپ قسمت اعشار، عدد باینری ۱,۱۱۱۱۱۱۱۱۱۱۱۱۱۱۱۱۱۱۱۱۱۱ که تقریبا مساوی ۲ است. در نتیجه عدد تقریباً معادل $۲^{۱۲۸}$ میباشد.

100000000 00000000000000000000001

معادل $۲^{-۱٤۹}$– است. بیت علامت ۱ است که نشانه منفی بودن عدد است. چون توان معادل صفر است، توان واقعی معادل ۱۲٦– میباشد. همچنین قسمت اعشار با قرار دادن ۰ سمت چپ قسمت اعشار بدست میاید. به بیان دیگر

0.00000000000000000000001

که معادل $۲^{-۲۳}$ میباشد. در نتیجه عدد نمایش داده شده با نمایش شناور بالا معادل $-2^{-23}.2^{-126}$ و یا -2^{-149} خواهد بود.

بحث کامل ریاضیات اعداد شناور مطابق با استاندارد IEEE بیش از حد این کتاب میباشد. هدف از بحث نمایش اعداد بصورت شناور آن بود که به شما نشان دهیم که بعلاوه روش نمایش مکمل ۲، روش های دیگری هم وجود دارد که بتوان اعداد خیلی بزرگ و یا خیلی کوچک را نمایش داد.

۲-۷-۳- کدهای اسکی (ASCII Codes)

نمایش دیگری از اطلاعات که تقریباً تمامی تولید کنندگان رایانه بر آن توافق کرده اند و آنرا برای انتقال اطلاعات بین مرکز محاسبات رایانه و ورودی خروجی های آن استفاده میکنند، کد اسکی میباشد. این کد که یک کد ۸ بیتی است، مخفف ASCII: American Standard Code for Information Exchange میباشد. این کد به شدت ارتباط بین صفحه کلید رایانه (Keyboard)، را که توسط یک کارخانه ساخته شده است، و خود رایانه که توسط شرکت دیگری تولید شده، و صفحه نمایش آن (یعنی مانیتور) که توسط شرکت سومی ساخته شده است را آسان میسازد.

هر دکمه بر روی صفحه کلید با یک کد منحصر بفرد اسکی مشخص شده است. برای مثال نمایش عدد ۳ در کد اسکی ۰۰۱۱۰۰۱۱، و نمایش ۲ ۰۰۱۱۰۰۱۰ ، نمایش حرف کوچک e معادل ۰۱۱۰۰۱۰۱ و نمایش کلید Return ۰۰۰۰۱۱۰۱ میباشد. لیست کامل جدول اسکی در شکل E.3 در ضمیمه ۵ آمده است. هنگامیکه شما یک کلید را روی صفحه کلید فشار میدهید، کد ۸ بیتی متناظر آن ذخیره شده و به رایانه داده میشود. اینکه این کد چگونه نگهداری شده و چگونه به رایانه داده میشود در فصل ۸ بررسی خواهد شد.

هر کلید در صفحه کلید به بیشتر از یک کد مرتبط شده است. مثلاً کد اسکی برای حرف E، ۰۱۰۰۰۱۰۱، و کد اسکی برای حرف e، ۰۱۱۰۰۱۰۱ بوده که هر دو به یک کلید وصل شده اند ولی یکی با گرفتن کلید shift و دیگری بدون گرفتن کلید shift ارسال می گردد.

برای نمایش حرف مخصوصی بر روی صفحه تصویر، رایانه لازم است که کد اسکی مربوط به آن را به قسمت الکترونیکی مربوط به صفحه تصویر بفرستد. جزئیات این مسئله هم در فصل ۸بحث خواهد شد.

۲-۷-۴- نمایش مبنای ۱۶ (Hexadecimal)

تا اینجا دیدیم که اطلاعات به صورتهای اعداد مکمل ۲، بردار بیتها، اعداد شناور یا کدهای

اسکی نمایش داده میشوند. نمایشهای دیگری هم وجود دارد که از حوصله این کتاب خارج است. ولی قبل از ترک این مبحث، اجازه دهید نمایش دیگری را معرفی کنیم که بیشتر برای سهولت کار انسان استفاده میگردد تا نمایشی برای کارها ومحاسبات داخل رایانه. این نمایش به نمایش مبنای ١٦ یا هگزادسیمال معروف است. این نمایش به زیبایی از روش نمایش مبنای ٢ حاصل میگردد و کار با رشته های طولانی اعداد باینری را ساده تر کرده و از خطا جلوگیری میکند.

بطور مشخص این روش نمایش برای کار با شبیه ساز LC-3 کاربرد دارد که در آن با رشته های باینری ١٦ بیتی سرو کار داریم. برای مثال رشته باینری زیر یک رشته ١٦ بیتی میباشد.

٠٠١١١١٠١٠١١٠١١١٠

اجازه دهید که آزمایشی انجام دهیم. با یک دست این ١٦ بیت را پوشانده و سعی کنید تا آنرا مجدداً با استفاده از حافظه خود بنویسید. نتیجه چه بود؟ آیا موفق بودید؟ روش نمایش مبنای ١٦ برای آنست که اینکار را بدون خطا بتوان انجام داد.

بطور کلی یک رشته باینری ١٦ بیتی صورت زیر را بخود میگیرد:

$$a_{15}a_{14}a_{13}a_{12}a_{11}a_{10}a_9a_8a_7a_6a_5a_4a_3a_2a_1a_0$$

که در آن هر بیت a_i یا ٠ است و یا ١. اگر این رشته بیت باینری را بصورت یک عدد بدون علامت در نظر بگیریم، مقدار آن را میتوان از فرمول زیر حساب کرد.

$$a_{15}.2^{15} + a_{14}.2^{14} +a_{13}.2^{13} +a_{12}.2^{12} +$$
$$a_{11}.2^{11} +a_{10}.2^{10} +a_9.2^9 +a_8.2^8 +$$
$$a_7.2^7 +a_6.2^6 +a_5.2^5 +a_4.2^4 +$$
$$a_3.2^3 +a_2.2^2 +a_1.2^1 +a_0.2^0$$

میتوان $۲^{۱۲}$ را از چهار بخش اول، $۲^{۸}$ را از چهار بخش دوم، $۲^{٤}$ را از چهار بخش سوم و $۲^{۰}$ را از چهار بخش چهارم بعنوان ضریب جدا کرد که نتیجه زیر حاصل میگردد:

$$2^{12}\,[a_{15}\,.2^3 + a_{14}.2^2 + a_{13}.2^1 + a_{12}.2^0\,] +$$
$$2^{8}\,[a_{11}.2^3 + a_{10}.2^2 + a_{9}.2^1 + a_{8}.2^1\,] +$$
$$2^{4}\,[a_{7}.2^3 + a_{6}.2^2 + a_{5}.2^1 + a_{4}.2^0\,] +$$
$$2^{0}\,[a_{3}.2^3 + a_{2}.2^2 + a_{1}.2^1 + a_{0}.2^0]$$

توجه کنید که بزگترین عدد قابل نمایش در داخل گیومه ها ۱۵ میباشد که در صورت ۱ بودن تمامی ضرایب حاصل میشود. اگر داخل گیومه ها را با نمایشی بین ۰ تا ۱۵ جایگزین نماییم، ضرایب $۲^{۱۲}$، $۲^{۸}$، $۲^{٤}$، $۲^{۰}$ را میتوان با $۱٦^{۳}$، $۱٦^{۲}$، $۱٦^{۱}$ و $۱٦^{۰}$ جایگزین کرد که حاصل آن:

$$h_3.16^3 + h_2.16^2 + h_1.16^1 + h_0.16^0$$

برای مثال h_3 نمایشی است که مقدار زیر را نشان میدهد

$$a_{15}\,.2^3 + a_{14}.2^2 + a_{13}.2^1 + a_{12}.2^0$$

از انجا که این نماد باید مقداری بین ۰ تا ۱۵ را نشان دهد، ما علامتهای زیر را به آن اختصاص میدهیم:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

بدین ترتیب، ۰۰۰۰ را با ۰، ۰۰۰۱ را با ۱، ...، ۱۰۰۱ را با ۹، ۱۰۱۰ را با A، ۱۰۱۱ را با B، ... و ۱۱۱۱ را با F نمایش میدهیم. نمایش حاصل، نمایش مبنای ۱٦ یا هگزادسیمال میباشد. برای مثال، E92F یک عدد صحیح مکمل ۲ی ۱٦ بیتی را بصورت هگزادسیمان (۱٦تایی) نمایش دهد. البته در این

نمایش نمیتوان براحتی گفت عدد مثبت است یا منفی. شما چگونه خواهید فهمید؟

حال سوال اینست که نمایش مبنای ١٦ به چه کار می آید؟ مبنای ١٦ همانند دیگرنمایشهاست بدون اینکه خاصیتی داشته باشد. برای مشخص شدن خاصیت مبنای ١٦ اجازه دهید به مثال نوشتن یک رشته بیت ١٦ بیتی ازطریق حافظه بازگردیم. در آن مثال، هدف این بود که رشته بیت زیر را بدون نگاه کردن به آن به کمک حافظه بنویسیم.

0011110101101110

این رشته را میتوان به رشته بیت های ٤ تایی تقسیم نمود.

0011 1101 0110 1110

حال هر چهار بیت را میتوان به معادل مبنای ١٦ آن تبدیل نمود و نمایش زیر حاصل خواهد شد.

3 D 6 E

حال نوشتن این نمایش از حافظه مشکل نبوده و به سادگی امکان پذیر است.

بطور خلاصه نمایش مبنای ١٦ برای راحتی انسان استفاده میشود. این نمایش برای نشان دادن رشته های باینری که عدد صحیح، عدد شناور، کد اسکی و یا بردار بیتها هستند استفاده میشود. کار این نمایش کاهش تعداد ارقام باینری به ١/٤ آن میباشد با جایگزین کردن هر ٤ بیت با معادل آن بین ٠ تا F. نتیجه این جایگزینی کاهش فاحش خطا در کپی کردن میباشد که در حالت نمایش با ٠ و ١ اتفاق میافتد.

تمرینها

۲-۱- با n بیت داده شده چه تعداد ترکیب مشخص می توان ایجاد کرد؟

۲-۲- الفبای انگلیسی شامل ۲٦ حرف می باشد، حداقل چه تعداد بیت مورد نیاز است که بتوان هرحرف را به یک بیت اختصاص داد؟ چه تعداد بیت به منظور تمایز قائل شده بین حروف بزرگ و حروف کوچک همه ۲٦ حرف مورد نیاز می باشد؟

۲-۳-

۱. فرض کنید ٤۰۰ دانش آموز در یک کلاس وجود دارد، اگر هر دانش آموز به یک الگوی بیتی تخصیص داده شود کمترین تعداد بیت مورد نیاز چقدر است؟

۲. چه تعداد دانش آموز اضافه تر می تواند در کلاس پذیرفته شود بدون نیاز داشتن به بیت اضافی جهت تخصیص به هر دانش آموز.

۲-٤- با n بیت چه تعداد اعداد صحیح بدون علامت می توان نمایش داد؟ ترتیب این اعداد چیست؟

۲-٥- با استفاده از ٥ بیتی ، اعداد ۷ و ۷- را به صورت اعداد صحیح مکمل ۱، مکمل ۲ و اعداد علامت دار نمایش دهید.

۲-٦- نمایش مکمل ۲ی ٦ بیتی از ۳۲- را نشان دهید.

۲-۷- جدولی رسم نمایید که مقادیر دسیمال همه اعداد مکمل ۲ی ٤ بیتی را نشان دهد.

۲-۸-

- بزرگترین عدد مثبتی که می تواند کد مکمل ۲ ۸ بیتی را نمایش دهد چیست؟ پاسخ را به صورت باینری و دسیمال بنویسید.
- بزرگترین عدد منفی که می تواند کد مکمل ۲ی ۸ بیتی را نمایش دهد چیست؟ پاسخ را به صورت باینری و دسیمال بنویسید.

- بزرگترین عدد مثبتی که می تواند کد مکمل ۲ی n بیتی را نمایش دهد چیست؟ پاسخ را به صورت باینری و دسیمال بنویسید.
- بزرگترین عدد منفی که می تواند کد مکمل ۲ی n بیتی را نمایش دهد چیست؟ پاسخ را به صورت باینری و دسیمال بنویسید.

۲-۹- چه تعداد بیت جهت نمایش عدد آووگادرو (٦,٠٢×$١٠^{٢٣}$) به صورت نمایش باینری مکمل ۲ مورد نیاز است؟

۲-۱۰- اعداد باینری مکمل ۲ی زیر را به دسیمال تبدیل نمایید.

- ۱۰۱۰
- ۰۱۰۱۱۰۱۰
- ۱۱۱۱۱۱۱۰
- ۰۰۱۱۱۰۰۱۱۱۰۱۰۰۱۱

۲-۱۱- اعداد دسیمال زیر را به اعداد باینری مکمل ۲ی ۸ بیتی تبدیل نمایید.

- ۱۰۲
- ٦٤
- ۳۳
- -۱۲۸
- ۱۲۷

۲-۱۲- اگر رقم آخر یک عدد باینری مکمل ۲ صفر باشد در نتیجه عدد زوج است. اگر دو رقم آخر عدد باینری مکمل ۲ صفر باشند (مانند عدد باینری ۰۱۱۰۰) چه نتیجه ای می توان گرفت؟

۲–۱۳– اعداد باینری مکمل ۲ی زیر را بدون تغییر در مقادیرشان به اعداد مکمل ۲ ۸ بیتی تبدیل نمایید.

- ۱۰۱۰
- ۰۱۱۰۰۱
- ۱۱۱۱۱۱۱۰۰۰
- ۰۱

۲–۱٤– عمل ADD کردن را در الگوهای بیتی زیر را انجام داده و به شکل باینری در آورید.

- ۱۰۱۱+۰۰۰۱
- ۰۰۰۰+۱۰۱۰
- ۱۱۰۰+۰۰۱۱
- ۰۱۰۱+۰۱۱۰
- ۱۱۱۱+۰۰۰۱

۲–۱۵– همانگونه که در مثال ۲–۵ نشان داده شد شیفت یک بیت عدد باینری به چپ معادل با حاصلضرب عدد در ۲ می باشد. چه عملیاتی انجام خواهد شد اگر یک بیت عدد باینری به راست تغییر مکان دهد؟

۲–۱٦– عملِ ADD کردن را در هریک از موارد زیر با توجه به روش استاندارد بحث شده در بخش ۲–۵–۱– انجام دهید.

- ADD کردن نمایش مکمل ۱ عدد ۷ با نمایش مکمل ۱ عدد ۷–
- ADD کردن نمایش مقدار علامت دار عدد ۷ با نمایش مقدار علامت دار ۷–
- ADD کردن نمایش مکمل ۲ی عدد ۷ با نمایش مکمل ۲ عدد ۷–

۲–۱۷– اعداد باینری مکمل ۲ی زیر را ADD نموده و پاسخ را به صورت دسیمال بیان نمایید.

- ۰۱+۱۰۱۱
- ۱۱+۰۱۰۱۰۱۰۱
- ۰۱۰۱+۱۱۰
- ۰۱+۱۰

۲–۱۸– عمل ADD کردن را در اعداد باینری بدون علامت زیر انجام داده و پاسخ را به صورت دسیمال بیان نمایید.

- ۰۱+۱۰۱۱
- ۱۱+۰۱۰۱۰۱۰۱
- ۰۱۰۱+۱۱۰
- ۰۱+۱۰

۲–۱۹– مقدار منفی عدد ۲۷– را به صورت عدد صحیح مکمل ۲ با استفاده از ۸ بیت بیان نمایید. این عمل را مجدداً با ۳۲ بیت انجام دهید. این عمل چه چیز را نشان خواهد داد باتوجه به ویژگیهای بسط علامت مرتبط با نمایش مکمل ۲.

۲–۲۰– اعداد زیر شامل اعداد باینری مکمل ۲ی ٤ بیتی هستند. کدامیک از عملیات زیر سرریز ایجاد می کند؟ پاسخ خود را با ترجمه عملوندها توجیه نموده و نتایج را به صورت دسیمال در آورید.

- ۱۱۰۰+۰۰۱۱
- ۱۱۰۰+۰۱۰۰
- ۰۱۱۱+۰۰۰۱

- ۱۰۰۰-۰۰۰۱
- ۰۱۱۱+۱۰۰۱

۲-۲۱- توصیف دهید چه شرایطی باعث می شود سرریز اتفاق بیفتد وقتی که دو عدد مکمل ۲ با هم جمع می شوند.

۲-۲۲- دو عدد صحیح مکمل ۲ ۱٦ بیتی را که جمعشان باعث overflow می شود را ایجاد نمایید.

۲-۲۳- توصیف دهید چه شرایطی باعث می شود سرریز اتفاق بیفتد وقتی که دو عدد بی علامت با هم جمع می شوند.

۲-۲٤- دو عدد صحیح بی علامت ۱٦ بیتی را که جمعشان باعث سرریز می شود را ایجاد نمایید.

۲-۲۵- چرا جمع یک عدد مکمل ۲ی منفی و یک عدد مکمل ۲ مثبت تولید سرریز نمی کنند؟

۲-۲٦- فرض کنید می خواهید عدد ٦٤- را به صورت عدد مکمل ۲ بیان نمایید.

- چه تعداد بیت مورد نیاز است(مینیمم عدد)؟
- با این تعداد بیت بزرگترین عدد مثبتی که می توان نمایش داد کدامست؟(پاسخ خود را به صورت باینری و دسیمال دهید)
- با این تعداد بیت بزرگترین عدد بی علامتی که می توان نمایش داد کدامست؟(پاسخ خود را به صورت باینری و دسیمال دهید)

۲-۲۷- یک ماشین ۱٦ بیتی – LC-3- دو عدد مکمل ۲ (۰۱۰۱۰۱۰۱۰۱۰۱۰۱۰۱ و ۰۰۱۱۱۰۰۱۱۱۰۰۱۱۱۱) را با هم جمع نموده، حاصل ۱۰۰۰۱۱۱۱۰۰۱۰۰۱۰۰ می شود. آیا در این حالت مشکلی وجود دارد؟ اگر بله چرا؟ اگر نه چرا؟

۲-۲۸- چه زمانی خروجی عمل ADD کردن برابر ۱ می شود؟

۲-۲۹- جدول صحت زیر را به منظور عمل ADD کردن یک بیتی تکمیل نمایید.

X	Y	X AND Y
۰	۰	
۰	۱	
۱	۰	
۱	۱	

۲-۳۰- عملیات زیر را محاسبه نموده و پاسخ خود را به شکل باینری در آورید.

- ۰۱۰۱۰۱۱۱ AND ۱۱۰۱۰۱۱
- ۱۰۱ AND ۱۱۰
- ۱۱۱۰۰۰۰۰ AND ۱۰۱۱۰۱۰۰
- ۰۰۰۱۱۱۱۱ AND ۱۰۱۱۰۱۰۰
- (۰۰۱۱ AND ۰۱۱۰) AND ۱۱۰۱
- ۰۰۱۱ AND (۰۰۱۱AND۰۱۱۰)

۲-۳۱- چه زمانی خروجی عملیات OR برابر ۱ است؟

۲-۳۲- عملیات OR را در جدول صحت زیر برای یک بیتی تکمیل نمایید.

X	Y	X OR Y
٠	٠	
٠	١	
١	٠	
١	١	

٢-٣٣- محاسبات زیر را انجام دهید:

- ١١٠١٠١١١ OR١١٠١٠١١١
- ١٠١ OR١١٠
- ١١١٠٠٠٠٠ OR١٠١١٠١٠٠
- ٠٠٠١١١١١ OR١٠١١٠١٠٠
- (٠١٠١OR١١٠٠)OR١١٠١
- ٠١٠١OR(١١٠٠OR١١٠١)

٢-٣٤- محاسبات زیر را انجام دهید:

- NOT(١٠١١) OR NOT(١١٠٠)
- NOT(١٠٠٠ AND (٠٠١١ OR ٠١٠١))
- NOT(NOT(١١٠١))
- (٠١١٠ OR٠٠٠٠) AND ١١١١

۲-۳۵- علت استفاده از الگو در مثال ۲-۱۱ چیست؟

۲-۳٦- جهت پاسخدهی به سوالات زیر به مثال ۲-۱۱ رجوع کنید.

چه میزان الگو و چه عملیاتی بکار می رود تا مشغول بودن ماشین ۲ را نشان دهد؟

چه میزان الگو و چه عملیاتی بکار می رود تا نشان دهد ماشین ۲ و ٦ خیلی طولانی مشغول نیستند؟(توجه: با تنها یک عملیات قابل انجام است)

چه میزان الگو و چه عملیاتی بکار می رود تا مشغول بودن همه ماشینها را نشان دهد ؟

چه میزان الگو و چه عملیاتی بکار می رود تا بر آزاد بودن همه ماشینها دلالت نماید؟

روشی را پیاده کنید که بیت ماشین ۲ به صورت یک sign bit جداشود، برای مثال اگر الگوی مشغول بودن ۰۱۰۱۱۱۰۰ است، پس خروجی این روش ۱۰۰۰۰۰۰۰ و اگر الگوی مشغول بودن ۰۱۱۱۰۰۱۱ باشد در نتیجه خروجی ۰۰۰۰۰۰۰۰ است. در مجموع اگر الگوی مشغول بودن به شکل زیر باشد:

b۷	b٦	b٥	b٤	b۳	b۲	b۱	b۰

در نتیجه خروجی به صورت زیر خواهد بود:

b۲	۰	۰	۰	۰	۰	۰	۰

راهنمایی:

چه اتفاقی خواهد افتاد اگر یک الگوی بیتی را با خودش ADD کنید؟

۲–۳۷– اگر m و n دو عدد مکمل ۲ی ٤ بیتی باشند و s چهار بیتی حاصل از جمع کردن این دو باشد، چگونه می توان با استفاده ازعملیات منطقی توضیح داده شده در قسمت ۲–٦ تعیین کرد که آیا در طول add کردن سرریز اتفاق افتاده است؟ برای این منظور روشی را پیاده کنید. بدین گونه که ورودیهای آن m و n و s و خروجی آن یک الگوی بیتی تمامی صفر (۰۰۰۰) اگر سرریز اتفاق نیافتاده باشد و ۱۰۰۰ اگر سرریز اتفاق افتاده باشد.

۲–۳۸– اگر m و n دو عدد بی علامت ۲ی ٤ بیتی باشند و s چهار بیتی حاصل از جمع کردن این دو باشد، چگونه می توان با استفاده ازعملیات منطقی توضیح داده شده در قسمت ۲–٦ تعیین کرد که آیا در طول add کردن سرریز اتفاق افتاده است؟ برای این منظور روشی را پیاده کنید. بدین گونه که ورودیهای آن m و n و s و خروجی آن یک الگوی بیتی تمامی صفر (۰۰۰۰) اگر سرریز اتفاق نیافتاده باشد و ۱۰۰۰ اگر سرریز اتفاق افتاده باشد.

۲–۳۹– نمایش ممیز شناور IEEE اعداد دسیمال زیر را بنویسید.

- ۳,۷۵
- –۵۵
- ۳,۱٤۱۵۹۲۷
- ٦٤,۰۰۰

۲–٤۰– معادل دسیمال اعداد ممیز شناورIEEE زیر را بنویسید.

- ۰ ۱۰۰۰۰۰۰۰ ۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰
- ۱ ۱۰۰۰۰۰۱۱ ۰۰۰۱۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰
- ۰ ۱۱۱۱۱۱۱۱ ۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰
- ۱ ۱۰۰۰۰۰۰۰ ۱۰۰۱۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰

۲-٤۱-

- بزگترین توان استاندارد IEEE که یک عدد ممیز شناور ۳۲ بیتی را نتیجه می دهد را بنویسید.
- کوچکترین توان استاندارد IEEE که یک عدد ممیز شناور ۳۲ بیتی را نتیجه می دهد را بنویسید.

۲-٤۲- شخص برنامه نویسی، برنامه ADD کردن دو عدد را می نویسد. پس از اجرای برنامه مشاهده کرد که زمانی که عدد ۵ با ۸ ADD می شود، نتیجه m خواهد بود. توضیح دهید چرا برنامه با خطا عمل می کند؟

۲-٤۳- کدهای ASCII زیر را از طریق ترجمه هر گروه ۸ بیتی به شکل یک کاراکتر ASCII به رشته حرفهایی ترجمه نمایید.

- x48656c6c6f21
- X68454c4c4f21
- X436f6d70757465727321
- X4c432d32

۲-٤٤- چه عملیات یا عملیاتهایی می تواند استفاده بشود جهت تبدیل نمایش باینری ۳(۰۰۱۱ ۰۰۰۰ ۰۰۰۰) به نمایش ASCII ۳ (۰۰۱۱ ۰۰۱۱)؟ این عملیات را برای تبدیل عدد باینری ٤ به ASCII نیز انجام دهید. در خصوص عدد دیگر چگونه خواهد بود؟

۲-٤۵- اعداد باینری بی علامت زیر را به هگزادسیمال تبدیل نمایید.

- ۱۱۰۱ ۰۰۰۱ ۱۰۱۰ ۱۱۱۱
- ۰۰۱ ۱۱۱۱
- ۱
- ۱۱۱۰ ۱۱۰۱ ۱۰۱۱ ۰۰۱۰

۲-٤٦- اعداد هگزادسیمال زیر را به باینری تبدیل نمایید.

- X10

- X801
- xF731
- x0F1E2D
- Xbcad

۲-٤٧- نمایشهای هگزادسیمال اعداد باینری مکمل ۲ی زیر را به اعداد دسیمال تبدیل نمایید.

- xF0
- x7FF
- x16
- x8000

۲-٤٨- اعداد دسیمال زیر را به نمایشهای هگزادسیمال اعداد مکمل ۲ تبدیل نمایید.

- ٢٥٦
- ١١١
- ١٢٣،٤٥٦،٧٨٩
- -٤٤

۲-٤٩- عملیات ADD کردن موارد زیر را انجام دهید. پاسخ را به صورت هگزا دسیمال در آورید.

- X025B + x26DE
- X7D96 + xF0A0
- xA397 + xA35D
- x7D96 + x7412

در خصوص قسمتهای cو d چه میتوان گفت؟

۲-٥٠- عملیات منطقی زیر را انجام داده و پاسخهای خود را به صورت هگزادسیمال در آورید.

- X5478 AND xFDEA
- xABCD OR x1234
- NOT((NOT(xDEFA)) AND (NOT(xFFFF)))
- X00FF XOR x325C

۲-٥١- نمایش هگزادسیمال اعداد زیر را بنویسید.

- ٢٥،٦٧٥
- ٦٧٥,٦٢٥ که ٦٧٥ $\frac{5}{8}$ می باشد، در ممیز شناور ٧٥٤ استاندارد IEEE
- رشته حروف ASCII: سلام

۲-۵۲- دو عدد هگزادسیمال x434F4D50 و x55544552 را بررسی نموده، چه مقادیری را برای هر ۵ نوع داده نشان داده شده در جدول زیر نمایش می دهند؟

	x434F4D50	x55544552
باینری بی علامت		
مکمل ۱		
مکمل ۲		
ممیز شناور IEEE ۷۵٤		
رشته حروف ASCII		

- ۲-۵۳- جدول صحت زیر را برای معادلهای داده شده تکمیل نمایید. ردیف اول به عنوان نمونه انجام شده است.

- Q_1= NOT(A AND B)
- Q_2= NOT(NOT(A) AND NOT(B))

A	B	Q_1	Q_2
0	0	1	0

Q2 را به روش دیگر بیان نمایید.

۲-٥٤- جدول صحت زیر را برای معادلهای داده شده تکمیل نمایید. ردیف اول به عنوان نمونه انجام شده است.

- Q_1= NOT(NOT(X) OR(X AND Y AND Z))
- Q_2= NOT((Y OR Z) AND (X AND Y AND Z))

X	Y	Z	Q_1	Q_2
0	0	0	0	1

۲-٥٥- قبلاً اعداد در مبنای ۲ (باینری) و در مبنای ١٦(هگزا) نمایش داده شدند. حال به اعداد در مبنای ٤ بی علامت می پردازیم که به آنها اعداد چهارتایی[16] گویند که این اعداد میتواناند ۰، ۱،۲ و یا ۳ باشند.

۱. بیشترین مقدار دسیمال بی علامت که می تواند ارقام چهارتایی ۳ را نمایش دهد چیست؟

۲. بیشترین مقدار دسیمال بی علامت که می تواند ارقام چهارتاییn را نمایش دهد چیست؟ (راهنمایی: پاسخ باید تابعی از n باشد)

۳. دو عدد چهارتایی بدون علامت را با هم ADD کنید. (۲۲۱ و ۰۲۳)

[16] quad

٤. نمایش چهارتایی عدد دسیمال ٤٢ چیست؟

٥. نمایش باینری عدد چهارتایی بی علامت 123.3 کدام است؟

٦. عدد چهارتایی بی علامت ١٢٣,٣ را به فرمت ممیز شناور IEEE بیان نمایید.

٧. یک جعبه سیاه داده شده که ارقام چهارتایی m را بعنوان ورودی دریافت کرده و یک رقم چهارتایی به عنوان خروجی ایجاد می نماید، ماکزیمم عدد توابع واحدی که این باکس می تواند پیاده سازد کدام است؟

٢-٥٦- یک ممیز شناور ٨ بیتی جدیدی را تعریف کنید با یک بیت علامت ، ٤ بیت توان و ٣ بیت از اعشار. اگر xE5 برای یک عدد در این ممیز شناور ٨ بیتی یک الگوی بیتی باشد ، مقدار آن چه خواهد بود؟ (به صورت دسیمال بیان کنید).