

جزوه درس مدار منطقی
علیرضا سلطانی نشان
دانشجوی نرم افزار
ترم سوم

فهرست مطالب

۴	۱ اعداد - مبنایها - مکمل ها - کدها
۵	۱.۱ تصاعد هندسی
۶	۲.۱ تبدیل مبنا از ۲ به ۱۰ یا برعکس
۶	۳.۱ تبدیل مبنا از ۲ به ۱۰
۶	۴.۱ تبدیل مبنا از ۱۰ به دو
۷	۵.۱ تبدیل مبنا از ۱۰ به هشت و برعکس
۷	۶.۱ تبدیل اعداد صحیح از مبنای ۱۰ به ۱۶ و برعکس
۷	۷.۱ تبدیل اعشاری دهدهی به دودویی و برعکس
۹	۸.۱ تبدیل اعداد مبنای دو به مبنای هشت و برعکس
۱۰	۹.۱ تبدیل مبنا ۲ به ۱۶ و برعکس
۱۱	۱۰.۱ تبدیل اعداد مبنای ۸ به ۱۶ و برعکس
۱۲	۱۱.۱ متمم ها یا (Complements)
۱۴	۱۲.۱ اعداد علامت دار
۱۶	۱۳.۱ عملیات روی اعداد بدون علامت در مبنایهای مختلف
۱۷	۱۴.۱ تقریب دو عدد بی علامت در مبنای ۲ به روش مستقیم (قرض گرفتن)
۱۷	۱۵.۱ جمع و تفریق اعداد علامت دار
۱۸	۱.۱۵.۱ جمع دو عدد علامت دار در سیستم مکمل ۲
۱۹	۱۶.۱ Overflow سرریز
۱۹	۱۷.۱ بازه مختلف ساخت اعداد در سیستم های مختلف با کمک n بیت
۱۹	۱۸.۱ تشخیص سرریز دو عدد بدون علامت
۱۹	۱۹.۱ تشخیص سرریز در اعداد علامت دار مکمل ۲
۲۱	۲۰.۱ تقریب دو عدد علامت دار در سیستم مکمل ۲
۲۲	۲۱.۱ سیستم D2B
۲۲	۱.۲۱.۱ کد کردن اعداد دهدهی
۲۴	۲.۲۱.۱ جمع دو عدد BCD
۲۴	۲۲.۱ کد گری یا کد انعکاسی

۲۵	۱.۲۲.۱ تبدیل باینری ۴ بیتی به گری ۴ بیت
۲۵	۲.۲۲.۱ تبدیل کد گری به کد باینری
۲۶	۲۳.۱ ASCII Codes
۲۸	۲۴.۱ کدهای تشخیص و تصحیح خطا
۲۸	۱.۲۴.۱ کدهای تشخیص خطا
۲۹	۲۵.۱ Overlapping Parity or Block Parity
۲۹	۲۶.۱ Parity Checksum
۳۰	۱.۲۶.۱ Single-precision checksum
۳۰	۲.۲۶.۱ Double-precision checksum
۳۰	۳.۲۶.۱ Residue checksum
۳۰	۴.۲۶.۱ HoneyWell checksum
۳۰	۲۷.۱ کد همینگ یا Hamming Code
۳۱	۱.۲۷.۱ فاصله همینگ
۳۱	۲.۲۷.۱ نحوه بدست همینگ کد و تشخیص و تصحیح خطا
۳۳	۳.۲۷.۱ پیدا کردن عدد غلط و تصحیح آن
۳۴	۴.۲۷.۱ نحوه محاسبه تعداد پرتی های همینگ

۲ جبر بول - ساده سازی - EPI PI،

۳۵	۱.۲ Not Or And truth table
۳۵	۲.۲ استفاده از نمودار ون برای نمایش منطقی
۳۵	۱.۲.۲ Single Variable
۳۵	۲.۲.۲ Double Variable
۳۶	۳.۲.۲ Three Variable
۳۷	۳.۲ on-set and off-set
۳۸	۴.۲ شمایل گیت های تاکنون گفته شده
۳۹	۵.۲ XNOR Xor NOR NAND truth table
۴۰	۶.۲ چند مورد از اصطلاحات
۴۰	۱.۶.۲ Term
۴۰	۲.۶.۲ Product Term
۴۰	۳.۶.۲ Sum Term
۴۰	۴.۶.۲ Min Term
۴۱	۵.۶.۲ Max Term
۴۱	۶.۶.۲ SoP Sum of Product
۴۱	۷.۶.۲ PoS Product of Sum

۴۲	۸.۶.۲	۱ خودت را امتحان کن
۴۲	۹.۶.۲	۲ خودت را امتحان کن
۴۳	۷.۲	Dual Low
۴۳	۱.۷.۲	Self Dual
۴۴	۲.۷.۲	Duality اصل
۴۴	۸.۲	خاصیت های گزاره ها
۴۵	۹.۲	توابع با خروجی چندتایی
۴۶	۱۰.۲	Gates
۵۲	۱۱.۲	Intermediate Functions
۵۳	۱۲.۲	تمارین مین ترم ها
۵۸	۱۳.۲	Min term diagram
۶۰	۱۴.۲	تمارین ماکس ترم ها
۶۲	۱۵.۲	Max term diagram
۶۵	۱۶.۲	گسترش و ترکیب مسئله
۶۷	۱۷.۲	بررسی گسترش مسئله در جمع ها و ضرب ها
۶۹	۱۸.۲	Converting to Canonical Sum تبدیل به جمع استاندارد
۷۰	۱۹.۲	بررسی و آنالیز مدار
۷۰	۱.۱۹.۲	مسئله ۱
۷۱	۲.۱۹.۲	مسئله ۲
۷۱	۲۰.۲	سناریو اول
۷۲	۲۱.۲	سناریو دوم
۷۲	۲۲.۲	سطوح مدار
۷۴	۲۳.۲	بهینه کردن مدار ها با تغییر جزئی در آنها
۷۴	۱.۲۳.۲	AND-OR/NAND-NAND in Sum of Product
۷۴	۲.۲۳.۲	OR-AND/NOR-NOR in Product of Sum
۷۵	۲۴.۲	Karnough Map
۷۵	۱.۲۴.۲	سنتز مدار
۷۷	۲۵.۲	نحوه ساده سازی به کمک کارنو مپ
۷۸	۲۶.۲	گروه بندی اعضا در جدول کارنو
۸۱	۲۷.۲	بدست آوردن PoS با SoP
۸۲	۱.۲۷.۲	نکته بسیار مهم
۸۶	۲۸.۲	کارنو با ورودی های زیاد

۸۸	۳ مدارات ترکیبی
۸۸	۱.۳ مدار سکویی مجوریتی یا Majority Circuit
۹۰	۲.۳ مقایسه کننده ها Comparator
۹۰	۱.۲.۳ مقایسه کننده تک بیتی
۹۱	۲.۲.۳ مقایسه کننده دوبیتی
۹۲	۳.۳ Half Adder
۹۳	۴.۳ Full Adder
۹۴	۵.۳ 4 Bit Ripple Carry Adder

۱ اعداد - مبنایها - مکمل ها - کدها

کلا اگه ما بخوایم هر عددی را نمایش بدهیم یا آنرا بنویسیم، این اعداد میتواند مبنایهای مختلفی داشته باشد. در حالت کلی عددی مثل a را میتوانیم اینگونه نمایش دهیم:

$$a = a_{n-1}a_{n-2}...a_2a_1a_0.a_{-1}a_{-2}a_{-3}...a_{-m} \bullet$$

در حقیقت هر کدام از این اعداد دارای ارزش مکانی هستند، که در زیر به درستی نشان داده شده است :

$$a = a_{n-1}^{r^{n-1}}, a_{n-2}^{r^{n-2}}, ..., a_2^{r^2}, a_1^{r^1}, a_0^{r^0}, a_{-1}^{r^{-1}}, a_{-2}^{r^{-2}}, a_{-m}^{r^{-m}} \bullet$$

که بعد از اعشار (۰) دارای n رقم عدد صحیح و قبل آن n رقم عدد اعشاری داریم. نکته مهمی که در اینجا بایستی یادآوری شود آنست که هر عدد به توان منفی برابر است با:

$$r^{-m} = \frac{1}{r^m} \bullet$$

معرفی انواع مبنایها:

۱. ۰...۱ = مبنای دو Binary

۲. ۰...۹ = مبنای ۱۰ Decimal

۳. ۰...۷ = مبنای هشت Octal

۴. ۰...۹, A, B, C, D, E, F = مبنای ۱۶ Decimal Hexa

نکته: زمانی که در مبناها، به مبناهای بزرگی میرسیم، ما از ۰ تا ۹ را به رسمیت میدانیم و برای نمایش اعداد بعد از آن از حروف انگلیسی استفاده میکنیم.

اگر عددی در مبنا r باشد و بخواهیم آنرا به مبنا ۱۰ تبدیل کنیم کافیه هر رقم را در ارزش مکانی خودش ضرب کرده و حاصل را باهم جمع کنیم.

۱.۱ تصاعد هندسی

$$(r-1)_{r^1} (r-1)_{r^0} \cdot (r-1)_{r^{-1}} (r-1)_{r^{-m}} = (r-1) \frac{r^n-1}{r-1} = r^n - 1$$

بطوری که r مبنا و n تعداد ارقام است، بطور مثال، در مبنا r دسیمال که سیستم دهدهی است، برای بدست آوردن بزرگترین عدد مجموعه از $r-1$ که میشود ۹، یعنی ۹ بزرگترین عدد مجموعه مبنا r دسیمال است، و اگر به تعداد دو رقم آنرا در نظر بگیریم، یعنی ۹۹، در جایگاه مربوطه بررسی خواهیم کرد، یعنی $99 = (10^1 * 9) + (10^0 * 9)$ و در مقابل تصاعد هندسی داریم، $10^2 - 1 = 99$.

تمرین: تبدیل مبناهای زیر را انجام دهید:

در هنگام انجام تبدیل مبناهای زیر، قصد یادآوری مطالب گذشته است، اما در بین آنها مطالبی هم ذکر شده است که نکاتی را در بر دارند و بایستی به عنوان مطلب جدید آنرا در نظر داشت:

نکات

برای تبدیل مبنا از ۱۰ به هر مبنا دیگری باید ۱۰ را در آن مبنا پی در پی تقسیم کنیم که در نهایت به با استفاده از باقی مانده ها و مقسوم علیه آخر تقسیم از راست به چپ نتایج تقسیم را کنار هم بگذاریم.

در هنگام تبدیل مبنا، جلوترین و آخرین ارزش مکانی عدد را MSB یا $MostSignificantBit$ و کمترین و آخرین ارزش مکانی در عدد را LSB یا $LeastSignificantBit$ گفته میشود. مانند: $(327.2)_{10}$ که عدد ۳ MSB و ۷ LSB .

۲.۱ تبدیل مبنا از ۲ به ۱۰ یا بالعکس

در تبدیل مبنا از ۲ به ۱۰ میبایست به صورت عادی با استفاده از ارزش مکانی هر عدد 2^0 و 2^1 ... یا به صورت خودمان ۱، ۲، ۴، ۸، ۱۶، ۳۲، ۶۴، ... و در نهایت هر قسمتی که بیت روشن یا ۱ داشت را نتیجه را با هم جمع میکنیم.

۳.۱ تبدیل مبنا از ۲ به ۱۰

تمرین:

$$(11001)_{2 \rightarrow 10} = (1 * 1) + (1 * 8) + (1 * 16) = 1 + 8 + 16 = 25 \bullet$$

$$(1111001)_{2 \rightarrow 10} = (1 * 1) + (1 * 8) + (1 * 16) + (1 * 32) + (1 * 64) = \bullet$$

$$1 + 8 + 16 + 32 + 64 = (121)_{10}$$

$$(100011111)_{2 \rightarrow 10} = (1*1)+(1*2)+(1*4)+(1*8)+1*16+(1*256) = (281)_{10} \bullet$$

۴.۱ تبدیل مبنا از ۱۰ به دو

همانطور که در بالاتر گفته شد در تبدیل مبنا از ۱۰ به ۲ یا به هر مبنای دیگر میتوانیم هب راحتی از تقسیم پی در پی ۱۰ در مبنای مورد نظر با استفاده از باقی مانده ها به جواب نهایی

برسیم.

تمرین:

$$(34)_{10 \rightarrow 2} = 100010 \quad (34)_{10 \rightarrow 2} = \bullet$$

Calculate	مانده باقی
$34 / 2 = 17$.
$17 / 2 = 8$	۱
$8 / 2 = 4$.
$4 / 2 = 2$.
$2 / 2 = 1$.
$1 / 2 =$	۱

۵.۱ تبدیل مبنا از ۱۰ به هشت و برعکس

$$\bullet (972)_{10 \rightarrow 8} =$$

$$(972)_{10 \rightarrow 8} = (1714)_8$$

Calculate	مانده باقی
$972 / 8 = 115$	۷
$115 / 8 = 14$	۳
$14 / 8 = 1$	۶
$1 / 8 = 0$	۱

$$\bullet (1637)_{8 \rightarrow 10} = (7 * 8^0) + (3 * 8^1) + (6 * 8^2) + (1 * 8^3) = (927)_{10}$$

۶.۱ تبدیل اعداد صحیح از مبنای ۱۰ به ۱۶ و برعکس

برای تبدیل مبنا از ۱۰ به ۱۶ درست مثل قبل از تقسیم پیاپی ۱۰ در ۱۶ استفاده خواهیم کرد:

$$\bullet (954)_{10 \rightarrow 16} = (3BA)_{16} \quad (954)_{10 \rightarrow 16}$$

Calculate	مانده باقی
$954 / 16 = 59$	$10 \rightarrow A$
$59 / 16 = 3$	$11 \rightarrow B$
$3 / 16 = 0$	۳

$$\bullet (3BA)_{16 \rightarrow 10} = (10 * 16^0) + (11 * 16^1) + (3 * 16^2) = 954$$

۷.۱ تبدیل اعشاری دهدهی به دودویی و برعکس

مهم ترین بخش تبدیل مبناها زمانیست که شما با مبنایی اعشاری رو به رو میشوید، در حالت تبدیل مبنا از دودویی به سیستم دهدهی، قسمتی که به صورت صحیح است را طبق معمول محاسبه می کنیم، و آن قسمتی که بعد از اعشار قرار دارد، به صورت $2^{-1}, 2^{-2}, 2^{-3} \dots$ محاسبه خواهیم کرد و در نهایت نتیجه قسمت صحیح را به نتیجه قسمت اعشاری قرار خواهیم داد:

$$(1110.01)_{2 \rightarrow 10} = .۱$$

$$(1110)_{2 \rightarrow 10} = 14$$

$$(.01)_{2 \rightarrow 10} = (0 * 2^{-1}) + (1 * 2^{-2}) = \frac{1}{4}$$

$$\rightarrow 14 + \frac{1}{4} = \frac{56}{4} + \frac{1}{4} = \frac{57}{4} = 14.25$$

$$(10111001.0101)_{2 \rightarrow 10} = .۲$$

$$(10111001)_{2 \rightarrow 10} = 185$$

$$(0.0101)_{2 \rightarrow 10} = \frac{1}{4} + \frac{1}{16} = \frac{4}{16} + \frac{1}{16} = \frac{5}{16} = 0.3125$$

$$\rightarrow 185 + 0.3125 = 185.3125$$

حالا نوبت به انجام تبدیل مبنا ۱۰ به ۲ میرسد، در این بخش، طبق معمول آن قسمتی که سمت صحیح عدد قرار دارد را به صورت عادی از ۱۰ به ۲ تبدیل میکنیم، اما آن بخشی که به صورت اعشاری است را باید کمی توجه کنیم، قسمت اعشاری را ضرب ۲ میکنیم، و حاصل آن را بررسی میکنیم و قسمت صحیح حاصل را به عنوان نتیجه تقسیم اول در نظر میگیریم، و آن بخش اعشاری را (فقط اعشاری) دو باره ضرب در دو میکنیم، انقدر این ضرب را انجام میدهیم که قسمت اعشاری به صفر برسد، ممکن است در طی این عملیات تعداد ضرب کردن به ۲ زیاد شود، در این مواقع تا هشت مرحله ضرب کردن کافی است (قاعده پرشدن حافظه ۸ بیت). لازم به ذکر است که در تمامی تبدیل های سیستم ددهی به هر مبنای دیگری، مانند باینری یا اکتال یا سه سه ای، چهارچهاری و غیره دقیقا همان عملیاتی که گفته شد صورت میگیرد، هم بخش تقسیم پیاپی در قسمت صحیح، هم ضرب متناوب در قسمت اعشاری.

تمرین:

$$(12.25)_{10 \rightarrow 2} = .۱$$

$$(12)_{10 \rightarrow 2} = (1100)_2$$

$$(0.25)_{10 \rightarrow 2} =$$

۲۵.۰ * ۲ = ۵۰.۰	۰
۵۰.۰ * ۲ = ۰.۱	۱

بعد از اینکه عدد اعشار حاصل دومین ضرب برابر با صفر شد دیگر نیازی به ضرب کردن نیست و از همان دو عدد نتیجه ۰ و ۱ استفاده میکنیم و در نتیجه خواهیم داشت:

$$\rightarrow (1100.01)$$

$$(15.361)_{10 \rightarrow 2} = .۲$$

$$(15)_{10 \rightarrow 2} = (1111)_2$$

$$(0.361)_{10 \rightarrow 2} =$$

$۳۶۱.۰ * ۲ = ۷۲۲.۰$	۰
$۷۲۲.۰ * ۲ = ۴۴۴.۱$	۱
$۴۴۴.۰ * ۲ = ۸۸۸.۰$	۰
$۸۸۸.۰ * ۲ = ۷۷۶.۱$	۱
$۷۷۶.۰ * ۲ = ۵۵۲.۱$	۱
$۵۵۲.۰ * ۲ = ۱۰۴.۱$	۱
$۱۰۴.۰ * ۲ = ۲۰۸.۰$	۰
$۲۰۸.۰ * ۲ = ۴۱۶.۰$	۰

→ (1111.01011100)

۸.۱ تبدیل اعداد مبنای دو به مبنای هشت و برعکس

برای تبدیل مبنای دو به هشت فقط کافی است از سمت راست به چپ سه بیت سه بیت جدا کنیم و براساس ریتم ۱، ۲، ۴ آنها را باهم جمع کنیم، نکته ای که در این میان باید توجه کنیم آن است که اگر تعداد ارقام مضربی از سه نباشد، بایستی از سمت چپ، عدد صفر اضافه کنیم.

مثال/تمرین:

$$(11001)_{2 \rightarrow 8} = .۱$$

$$(011'001)_{2 \rightarrow 8} = (31)_8$$

$$(1110001)_{2 \rightarrow 8} = .۲$$

$$(001'110'001)_{2 \rightarrow 8} = (161)_8$$

$$(100111)_{2 \rightarrow 8} = .۳$$

$$(100'111)_{2 \rightarrow 8} = (47)_8$$

تبدیل مبنا اعشاری از دو به هشت:

این نوع تبدیل مبناهم بایستی در نظر داشته باشیم که بهتر به دو قسمت صحیح و اعشاری تقسیم می شود، در قسمت اعشاری اگر تعداد ارقام مضربی از ۳ بود میتوان به صورت سه تا سه از راست به چپ از جدایی را انجام داد، در غیر این صورت باید از سمت چپ به تعداد لازم ۰ اضافه کنیم. در قسمت اعشاری هم همین قاعده صادق است، با این تفاوت که اگر تعداد ارقام سمت اعشار مضربی از ۳ نبود این بار بایستی از سمت راست به تعداد لازم صفر وارد

کنیم.

$$\begin{aligned}(10011.1101)_{2 \rightarrow 8} &= .1 \\ (010'011)_{2 \rightarrow 8} &= (23)_8 \\ (0.110'100)_{2 \rightarrow 8} &= (64)_8 \\ &\rightarrow (23.64)_8\end{aligned}$$

تبدیل برعکس ۸ به ۲ هم دقیقاً به همین صورت است.

$$\begin{aligned}(10101.0111)_{2 \rightarrow 8} &= .1 \\ (010'101)_{2 \rightarrow 8} &= (25)_8 \\ (011'100)_{2 \rightarrow 8} &= (34)_8 \\ &\rightarrow (25.34)_8\end{aligned}$$

$$\begin{aligned}(25.34)_{2 \rightarrow 8} &= .2 \\ (010'101)_{2 \rightarrow 8} &= (25)_8 \\ (011'100)_{2 \rightarrow 8} &= (34)_8 \\ &\rightarrow (10101.0111)_2\end{aligned}$$

$$\begin{aligned}(55.67)_{2 \rightarrow 8} &= .3 \\ (101'101)_{2 \rightarrow 8} &= (55)_8 \\ (110'111)_{2 \rightarrow 8} &= (67)_8 \\ &\rightarrow (101101.110111)_2\end{aligned}$$

۹.۱ تبدیل مبنا ۲ به ۱۶ و برعکس

در تبدیل مبنا از ۲ به ۱۶ هم دقیقاً مانند مبنای ۸ است که، بطوری که باید تعداد ارقام ضربی از ۴ باشند و در هنگام اعشاری شدن هم باید برای جبران کسری صفر در سمت چپ برای عدد صحیح و در سمت راست برای عدد اعشاری.

$$\begin{aligned}(1111101)_{2 \rightarrow 16} &= .1 \\ (0111'1101)_{2 \rightarrow 16} &= (7D)_{16} \\ (1011101100)_{2 \rightarrow 16} &= .2 \\ (0010'1110'1100)_{2 \rightarrow 16} &= (2EC)_{16}\end{aligned}$$

$$(1111101.0110)_{2 \rightarrow 16} = 16 \text{ to } 2 \text{ Decimal } 3$$

$$(0111'1101)_{2 \rightarrow 16} = (7D)_{16}$$

$$(.0110)_{2 \rightarrow 16} = 6$$

$$\rightarrow (7D.6)_{16}$$

$$(F25.03)_{16 \rightarrow 2} = 2 \text{ to } 16 \text{ } 4$$

$$(F25)_{16 \rightarrow 2} = (1111'0010'0101)_2$$

$$(.03)_{16 \rightarrow 2} = (0000'0011)_2$$

$$\rightarrow (111100100101.00000011)_2$$

۱۰.۱ تبدیل اعداد مبنای ۸ به ۱۶ و برعکس

$$(A36)_{16 \rightarrow 8} = 1$$

$$(1010'0011'0110)_{16 \rightarrow 2} = (101000110110)_2$$

$$(101'000'110'110)_{2 \rightarrow 8} = (5066)_8$$

$$(753)_{8 \rightarrow 16} = 2$$

$$(111'101'011)_{8 \rightarrow 2} = (111101011)_2$$

$$(0001'1110'1011)_{2 \rightarrow 16} = (1EB)_{16}$$

Extra

ممکن است بخواهیم از عددی که به صورت دسیمال است متوجه بشویم که این عدد در مبنای دو چند بیتی است، برای این کار، چون که در مبنای دو هستیم، پس بر پایه توانی از ۲ پیش می‌رویم، به همین خاطر عددی که بدست خواهید آورد بایستی از عدد دسیمال گفته شده بزرگ تر مساوری باشد. مانند:

$$(768)_{10 \rightarrow 2}$$

$$2^8 = 256 - 1 > = 768$$

$$2^9 = 512 - 1 > = 768$$

$$2^{10} = 1024 - 1 > = 768$$

پس متوجه خواهید شد که عدد ۷۶۸ وقتی به بانتری تبدیل میشود ۱۰ بیتی خواهد بود.

۱۱.۱ متمم‌ها یا (Complements)

متمم‌ها در کامپیوترهای دیجیتال برای ساده کردن عمل تفریق و یا عملیات منطقی به کار میروند. ساده سازی عملیات منجر به پیاده سازی مدارات ساده تر میگردد. در هر مبنایی مانند r ، دو نوع متمم وجود دارد: یکی متمم مبنای و دیگری متمم کاهش یافته. فرم اول به نام متمم r و دومی به متمم $r - 1$ مرسوم است. وقتی که مقدار مبنای (یا پایه) را جایگزین کنیم، برای اعداد دودویی، متمم‌های ۲ و ۱ و برای دهدهی، متمم‌های ۱۰ و ۹ را خواهیم داشت.

دو نوع متمم برای هر عدد در مبنای r وجود دارد:

- متمم مبنای یا مکمل r
 - متمم مبنای کاهش یافته یا مکمل $r - 1$
- اگر عدد N در مبنای r شامل n رقم باشد:

- متمم مبنای $r^n - N$
- متمم مبنای کاهش یافته $(r^n - 1) - N$

نکته بسیار مهم: r^n در هر مبنایی برابر است با $1 + n(0)$

برای مثال در 16^3 درست است که برابر با 4096 میشود اما این عدد بدست آمده در واقع در مبنای ۱۰ است نه در مبنای ۱۶، بهمین خاطر بایستی عدد حاصله را در مبنای ۱۶ تبدیل کنیم، که با توجه به قاعده بالا ما خواهیم داشت یک عدد ۱ به همراه تعداد ارقام صفر یعنی ۱۰۰۰

$$2^3 = 1000 \ 10^3 = 1000_{10} \ 8^3 = 1000_8 \ 16^3 = (1000)_{16}$$

اما، باید توجه داشته باشیم که هر کدام از اعداد بدست آمده بالا در حقیقت آنچیزی که نشان میدهد، نیست، فقط عدد ۱۰۰۰ در مبنای ۱۰ است، که حقیقتاً برابر این مقدار است، بقیه حالت‌ها در حالت ماکسیموم خود قرار دارند، یعنی ۱۰۰۰ در مبنای ۸ برابر با ۷۷۷ است.

و به یاد داشته باشید که در بدست آوردن مبنایی مانند ۸ و هر مبنایی به غیر از ۲ و ۱۰ (برای مثال در اینجا هشت آورده شده)، مقدار ۷۷۷ درواقع مقدار متمم کاهش یافته را به شما بر میگرداند.

و همچنین در این مبنای یعنی مبنای ۸، همانطور که گفته شد با ۷۷۷ شما مکمل کاهش یافته را بدست خواهید آورد، برای بدست آوردن مکمل یا متمم ۸ بایستی در نظر داشته باشید

اگر عدد مبنای هشت شما در انتها دارای صفر بود، مانند ۱۲۰، ۴۰۳۰۰، یا هر عددی که به تعداد صفر ختم شده باشد، در جواب این مسئله با تعداد عدد ۷ در آخر عدد مواجه میشوید، برای داشتن متمم مبنای ۸ همین تعداد ۷ پایانی را کافیسیت برابر با صفر قرار دهیم و یک عدد به عدد بعد از آن اضافه کنیم

لازم به ذکر است که در هر مبنایی به غیر از ۱۰ و دو در بدست آوردن متمم ها، شما همیشه متمم کاهش یافته را بدست می آورید!

مثال های مهم برای این نکته:

$$(123)_8 = 8^3 - 123 = 777 - 123 = 654 \text{ reduce. } 654 + 1 = 655 \text{ radix}$$

$$(1230)_8 = 8^3 - 1230 = 7777 - 1230 = 6547 \text{ reduce. } 6547.6540 + (4 + 1) \rightarrow 6550$$

این مورد تنها در مبنای ۸ نیست بلکه در مبناهای طبیعی دیگر، مانند ۳، ۴، ۵، ... نیز وجود دارد.

نتیجه گیری نهایی اگر، مبنای ۱۰ دادن برای مکمل و مکمل کاهش، از قاعده بومی $r^n - n$ و $(r^n - 1) - n$ استفاده میکنیم که مبنا به توان تعداد ارقام برابر با همان ۱ با تعدادی صفر در جلوش است. در مبنای دو هم که طبق تعریف نیازی به دوباره کاری نیست، و فقط به قاعده ای که در چند خط بالا گفته شد دقت کنید.

متمم اعداد

متمم ۱۰ عدد ۵۴۶۷۰۰ برابر است با

$$10^6 - 546700 = 453300$$

متمم ۹ عدد ۵۴۶۷۰۰ برابر است با

$$(10^6 - 1) - 546700 = 453299$$

روش سریع دیگر آن است که اگر به ما مبنا را دادند میتوانیم با یکی کم کردن از آن به مبنای کاهش یافته آن برسیم، و برعکس، اگر متمم مبنای کاهش یافته را به ما دهند میتوانیم با یکی اضافه کردن به آن به متمم مبنا برسیم!

متمم ۲ عدد ۱۱۰۱۱۰۰ برابر است با:

$$108 = 1101100$$

$$2^7 - 108 = (20)_{10} = (10100)_2$$

متمم ۱ عدد ۱۱۰۱۱۰۰ برابر است با:

$$(2^7 - 1) - 108 = (19)_{10 \rightarrow 2} = (0010011)_2$$

متمم اعداد در مبنای ۲ به سریع ترین روش

نکات

متمم ۱: تمام ارقام NOT میشود. متمم ۲: تمام صفر های سمت راست تا زمانی که به اولین یک از سمت راست برسیم همان طور به شکل اول خود باقی می ماند، اما از آن به بعد بیت های بعدی NOT میشود.

متمم ۱ عدد ۰۰۱۱۰۱۱ برابر است با: ۱۱۰۰۱۰۰

متمم ۲ عدد ۱۱۰۱۱۰۰ برابر است با: ۰۰۱۰۱۰۰

این همه گفتیم، پس در سیستم ۱۶ تایی چگونه؟

برای بدست آوردن مکمل ۱۶ و مکمل کاهشی آن (۱۵) بایستی بدانیم که ماکسیموم برای n رقم مثل $16^3 = (1000)_{16}$ آخرین درجه برای مقدار ۱۰۰۰، ۱۵ ۱۵ ۱۵ است که میبایستی این عدد را از مبنای ۱۶ کم کنیم تا آنگاه مبنای کاهش یافته را بدست بیاوریم.

مثال

$$(A86)_{16} = 16^3 = 1000_{16} - A86 = 151515 - 1086 = 579 \text{ Reduce... } 579(9 + 1 = 10 = A) = 5710 \text{ or } 57A$$

تمرین

۱. مبنا ۱۰ و مکمل ۹ عدد $(256.73)_{10}$ را بنویسید.

۲. مبنا ۱۰ و مکمل ۹ عدد $(325.12)_{10}$ را بنویسید.

۳. هر دو مقدار ۹ و ۱۰ $(256.73)_{10}$

۴. هر دو مقدار ۹ و ۱۰ $(325.12)_{10}$

۵. هر دو مقدار ۷ و ۸ $(276.35)_8$

۶. هر دو مقدار ۹ و ۱۰ $(9300)_{10}$

۷. $(304000)_8$

۸. $(11110100)_2 = 00001100$

۱۲.۱ اعداد علامت دار

اعداد را میتوان به ۴ روش نمایش داد:

- روش بدون علامت:
عدد را به صورت عادی ارزش گذاری میکنیم و در نهایت نتیجه را مینویسیم
- روش مقدار علامت:
سمت چپ ترین بیت نشان دهنده علامت عدد است.
- نمایش عدد به صورت مکمل یک
- نمایش عدد بصورت مکمل ۲

نکته

- در روش مقدار علامت، بیت صفر نشان دهنده مثبت بودن و بیت یک نشان دهنده منفی بودن است.
- در مکمل یک عدد را به صورت مکمل یک نمایش میدهیم.
- در مکمل ۲ عدد را بصورت مکمل دو نمایش میدهیم.

مثال

عدد ۸ بیتی $(10010100)_2$ را ب به مبنای دهدهی تبدیل کنید با فرض اینکه:

۱. این عدد در سیستم بی علامت باشد. $(10010100)_2 = 4 + 16 + 128 = 148$

۲. این عدد در سیستم علامت مقدار باشد. $(10010100)_2 = -(4 + 16) = -20$

۳. این عدد در سیستم مکمل ۱ باشد.

$$(10010100)_2 = -(01101011) = -(1 + 2 + 8 + 32 + 64) = -107 =$$

۴. این عدد در سیستم مکمل ۲ باشد.

$$(10010100)_2 = -(01101100) = -(4 + 8 + 32 + 64) = -108$$

مثال

عدد ۸ بیتی $(01000101)_2$ را ب به مبنای دهدهی تبدیل کنید با فرض اینکه:

۱. این عدد در سیستم بی علامت باشد. $(01000101)_2 = 69$

۲. این عدد در سیستم علامت مقدار باشد. $(01000101)_2 = +69$

۳. این عدد در سیستم مکمل ۱ باشد.

$$(01000101)_2 = +(10111010) = 2 + 8 + 16 + 32 + 128 = +186$$

۴. این عدد در سیستم مکمل ۲ باشد. $(01000101)_2 = +(10111011) = +187$

نکته مهم

گاهی ممکن است بخواهیم عددی را منفی اش را به صورت دودویی بنویسیم اما باید یک نکته مهمی را مورد نظر داشته باشیم، برای مثال اگر از ما بخواهند که عدد ۹ را نمایش دهیم آن را به صورت $(1001)_2$ نشان میدهیم، اما اگر از ما بخواهند که منفی این عدد را نمایش بدهیم نمی توان آنرا به این صورت $(11001)_2$ نشان داد، در حقیقت بازهم به جواب ۹- خواهیم رسید اما در ۵ بیت معنایی ندارد، چرا که کامپیوتر همه اعداد در در تعدادی از توان های دو نمایش میدهد مانند ۴ بیت، ۸ بیت، ۱۶، ۳۲، ۶۴، ۱۲۸ و غیره. پس برای نشان دادن عدد ۹- باید به این صورت بنویسیم: $(1000, 1001)_2 = (-9)_{10}$

۱۳.۱ عملیات روی اعداد بدون علامت در مبنای مختلف

انسان برای شمارش و انجام عملیات ریاضی (جمع و تفریق و ضرب و تقسیم) از مبنای ۱۰ استفاده میکند، دلیل این انتخاب توسط انسان، تعداد انگشت های دست او بود. جدول ضرب هم بر اساس مبنای ۱۰ نوشته شده است. اما اگر ما ۸ انگشت داشتیم مجبور بودیم از مبنای ۸ استفاده کنیم. در این صورت دیگر جمع و تفریق ما بر اساس مبنای ۸ است که جواب هایی که از مبنای ۱۰ در محاسبات ریاضی بدست می آوریم در مبنای ۸ بسیار متفاوت است. یعنی $۷ \times ۶ = ۴۲$ ، $۱۲ - ۵ = ۷$ ، $۷ + ۱ = ۸$. پس میفهمیم که محاسبات در مبنای غیر ۱۰ برای ما بسیار سخت و دشوار است. زیرا ما وقتی عملیات ساده مبنای غیر از ۱۰ را حفظ نیستیم پس میتوانیم مسائل را به زبان خودمان یعنی مبنای ۱۰ ترجمه کنیم و در همین مبنای محاسبات را انجام دهیم و سپس به مبنای خواسته شده توسط مسئله تبدیل میکنیم. در مورد عملیات جمع و تفریق، میتوانیم از مبنای ۱۰ استفاده نکنیم و از جدول کمک بگیریم.

جمع اعداد در مبنای غیر ۱۰

تمرین:

$$(101101)_2 + (010111)_2 = \bullet$$

$$(276)_8 + (357)_8 = \bullet$$

$$(276)_{10} + (357)_{10} = \bullet$$

$$(2A58)_{16} + (71D0)_{16} = \bullet$$

$$(2F2C)_{16} + (2FAA)_{16} = \bullet$$

جمع دو عدد در مبنای ۲

$$(111101)_2 + (10111)_2 = (1010100)_2$$

در هنگام جمع دو عدد باینری باید توجه داشت که اگر جمع اول با دومی بیشتر از ۱ شد، در حقیقت عدد بدست آمده در مبنای ۱۰ است، به همین خاطر این عدد را بایستی به مبنای دو سریعاً تبدیل کنیم، و با بخش بعدی به جمع بپردازیم. بعد از اینکه حاصل خود را به صورت عدد دودویی بدست آوردیم، برای بررسی آن میتوانیم، صورت اول جمع را به سیستم دهدهی و صورت دوم هم همینطور تبدیل کرده و سپس باهم جمع کنیم، و در نهایت حاصل را به مبنای ده برده و در آخر بررسی برابر خود را انجام میدهیم، این بررسی نشان دهنده آن است که حاصل بدست آمده در مبنای دو چقدر امکان خطا دارد، در ادامه صحبت خواهیم کرد.

۱۴.۱ تفریق دو عدد بی علامت در مبنای ۲ به روش مستقیم (قرض گرفتن)

در این نوع محاسبه تفریق همانند تفریق سیستم دهدهی عمل میکنیم، همان طور که در سیستم دهدهی هرگاه به عدد ۰ میرسیم که بر روی عددی غیر ۰ تفریق کنیم به صفر، ده عدد قرض میدادیم و از خانه بعدی صورت یکی کم میکردیم، در تفریق مبنای دو هم، دقیقاً همچنین اتفاقی رخ می دهد، شما زمانی که در صورت به عدد صفر میرسید که در عدد پایینی عددی غیر صفر قرار دارد، تا سقف دو، به عدد صفر قرض میدهید و یکی از خانه بعدی عدد کم میکنید و عمل تفریق در مبنای دو هم به آسانی صورت خواهد گرفت

مثال

$$(1001101)_2 - (10111)_2 = (0110110)_2$$

		1			2				
	0	2	2	0	0	2			
	1	0	0	1	1	0	1	= 77	
-			1	0	1	1	1	= 23	
	0	1	1	0	1	1	0	= 54	

شکل ۱: مثالی از تفریق در مبنای دو

۱۵.۱ جمع و تفریق اعداد علامت دار

به روش های متداول قبلی قابل انجام است.
معمولاً در کامپیوتر در سیستم مکمل ۲ انجام می شود.
بایستی همه عملوند ها را به سیستم مکمل ۲ ببریم
جواب نهایی در سیستم مکمل ۲ بدست خواهد آمد، باید برای خواندن آن دقت کنیم.

۱.۱۵.۱ جمع دو عدد علامت دار در سیستم مکمل ۲

جمع در سیستم مکمل ۲، بدون توجه به مثبت یا منفی بودن اعداد، آنها را زیر هم نوشته و جمع میکنیم، از رقم نقلی خروجی صرف نظر میکنیم یعنی آنرا حذف می کنیم.

مثال: حاصل عملیات جمع را در ۴ بیت در سیستم مکمل دو حساب کنید:

نکته ای که در این مسئله باید به آن توجه داشته باشید، آن است که در هنگام جمع به صورت بی علامت پیش میرویم، برای بررسی هر کدام از صورت جمع ها، آن عددی که بیت علامتش ۰ است که به صورت دهدهی عادی تبدیل میشود، در غیر این صورت بایستی اول به صورت مکمل ۲ نوشته شده و بعد به مبنای ۱۰ تبدیل شود.

$$1. \quad (0001)_{2(+1)} + (1001)_{2(-7)} = (1010)_{2-(6)}$$

$$2. \quad (0010)_{2(+2)} + (0111)_{2(7)_{2+7=9}} = (1001)_{2-(0111 \rightarrow -7)}$$

در مثال بالا در حقیقت Overflow رخ داده که جواب اشتبا بدست آمده است. و فلگ کری ۱ خواهد شد چرا که یک بیت اضافی دارد.

$$3. \quad (0011)_{2(+3)} + (0100)_{2(+4)_{3+4=7}} = (0111)_{2+(7)}$$

$$4. \quad (1001)_{2(-7)} + (0111)_{2(+7)_{7-7=0}} = ([1]0000)_{2(0)}$$

در مثال بالا، [۱] به این خاطر حذف شده چرا که حاصل بدست آمده بیشتر از ۴ بیت میشد.

نکته:

در هنگام جمع دو عدد در مبنای دو، اگر برای مثال صورت اول و صورت دوم هر دو مثبت باشند و در نهایت در نتیجه عددی منفی را داشته باشیم، اورفلو رخ خواهد داد، برای تصحیح این مشکل فقط کافیست که از روش Sign Extended استفاده کنیم، که اگر ۴ بیتی باشد می شود، هشت بیت، اگر هشت بیتی باشد میشود ۱۶ بیت الی آخر. این طور می توان به جواب درست دست پیدا کرد (تکرار عدد آخر).

برای مثال، در تمرین دوم بالا، ما اورفلو داریم چرا که نتیجه بدست آمده با جمع کلی درست نیست، یا اینطور جمع دو عدد مثبت باید مثبت شود اما نتیجه منفی شده است، برای بدست آوردن نتیجه درست از Sign Extended استفاده کنیم:

$$(0000, 0010)_2 + (0000, 0111)_2 = (0000, 1001)_2$$

در بالا بیت علامت صفر، اورو فلو صفر، فلگ صفر، صفر و کری هم صفر خواهد بود، و نتیجه بدست آمده هم مثبت.

۱۶.۱ Overflow سرریز

اعداد در کامپیوتر با طول محدود و تعداد بیت های مشخص به کار برده می شوند. اگر نتیجه محاسبات خارج از این محدوده شود و بیت های بیشتر در دسترس نباشد، این بیت های اضافی حذف خواهند شد و نتیجه بدست آمده صحیح نخواهد بود. در این حالت می گوییم در انجام محاسبه سرریز اتفاق افتاده است.

فلگ v یا of (*Overflow flag*): این فلگ وقتی ۱ میشود که از نتیجه محاسبات در بازه مجاز تعداد بیت نباشد، در این حالت میگوییم اورفلو یا سرریز رخ داده است. در ادامه مطالب حتما مثال هایی در این خصوص زده خواهد شد.

۱۷.۱ بازه مختلف ساخت اعداد در سیستم های مختلف با کمک n بیت

در سیستم بی علامت:

$$0 - (111...1)_2 = 2^n - 1$$

در سیستم علامت و مقدار:

$$(111...1)_2 = -(2^{n-1} - 1) - (111...1)_2 = (2^{n-1} - 1)$$

سیستم مکمل یک:

$$(111...1)_2 = -(2^{n-1} - 1) - (111...1)_2 = (2^{n-1} - 1)$$

سیستم مکمل دو:

$$(111...1)_2 = -(2^{n-1}) - (111...1)_2 = (2^{n-1} - 1)$$

۱۸.۱ تشخیص سرریز دو عدد بدون علامت

در جمع اعداد در سیستم بدون علامت، اگر پس از جمع دو علامت رقم، رقم آخر (نقلی و نهایی) یک شود، سرریز اتفاق خواهد افتاد.

$$(1101)_2 + (1100)_2 = ([1](1001))_2$$

۱۹.۱ تشخیص سرریز در اعداد علامت دار مکمل ۲

نکته:

اگر جمع دو عدد منفی، مثبت شود یا جمع دو عدد مثبت منفی شود، سرریز رخ میدهد. دقت

داشته باشید، جمع دو عدد مثبت و منفی باهم، سرریز ندارد.

اگر دو عدد A و B در سیستم مکمل ۱ و ۲ باشند، آنگاه $A+B$ در صورتی سرریز خواهد بود که:

۱. اگر A و B هر دو مثبت باشند و نتیجه منفی را بدهند.

۲. اگر A و B منفی باشند و نتیجه مثبت را بدهند.

نتیجه:

اگر دو عدد با علامت های مختلفی داشته باشیم هیچگاه اورفلو رخ نخواهد داد.

اگر دو عدد A و B در سیستم مکمل ۱ یا ۲ باشند، آنگاه $A-B$ در صورتی سرریز خواهد بود که:

۱. A مثبت باشد و B منفی باشد، و حاصل آنها منفی شود.

درستش:

$$(A_+) - (B_-) = (+)$$

۲. A منفی باشد و B عددی مثبت، حاصل آن مثبت شود

درستش:

$$(A_-) - (B_+) = (-)$$

دیگر نکته بر روی دیگر فلگ های سادست که مختصرا بیان میکنیم:

فلگ zero زمانی ۱ میشود که حاصل بدست آمده برابر با صفر باشد در غیر این صورت ۰ خواهد بود.

فلگ carry زمانی ۱ میشود که بیت اضافی وجود داشته باشد.

فلگ sign زمانی یک میشود که بیت علامت ما منفی باشد.

فلگ اورفلو زمانی یک میشود که حاصل بدست آمده با جمع منطقی ما (اگر عدد صورت اول منفی بود به مکمل ۲ میبریم در غیر این صورت بصورت ساده جمع را انجام میدهیم) برابر نباشد. (با توجه به قوانینی که در بالاتر توضیح داده شد)

حل چند تمرین:

$$۱. (1001)_{2c(-7)} + (0111)_{27} = ([1]0000)_{2(0)}$$

$$\text{carry} = ۱, \text{Zero} = ۱, \text{Of} = ۰, \text{sign} = ۰$$

$$۲. (1101)_{2c(-3)} + (1110)_{2c(-2)} = ([1]1011)_{2c(-5)}$$

$$\text{carry} = ۱, \text{Zero} = ۰, \text{Of} = ۰, \text{sign} = ۱$$

$$۳. (0101)_{2(5)} + (0100)_{24} = (1001)_{2c(0111=>-7)}$$

$$\text{carry} = ۰, \text{Zero} = ۰, \text{Of} = ۱, \text{sign} = ۱$$

$$۴. (1001)_{2c(-7)} + (1010)_{2c-6} = ([1]0011)_{2(3)}$$

$$\text{carry} = ۱, \text{Zero} = ۰, \text{Of} = ۱, \text{sign} = ۰$$

+ 6	00000110	- 6	11111010
+13	00001101	+13	00001101
+19	00010011	+ 7	00000111
+ 6	00000110	-6	11111010
-13	11110011	-13	11110011
- 7	11111001	-19	11101101

شکل ۲: تمرین از جمع اعداد در سیستم مکمل ۲

۲۰.۱ تفريق دو عدد علامت دار در سیستم مکمل ۲

در هنگام تفريق دو عدد در مکمل ۲، ميتوانيم عدد A را با مکمل عدد B جمع کنيم. و اين جمع مانند جمعی است که در بالاتر توضيح داده شد. **حل چند تمرين:**

$$۱. (1101)_2 - (1001)_2 \rightarrow (1101)_{2c(-3)} + (0111)_7 = ([1]0100)_2$$

$$\text{carry} = ۱, \text{Zero} = ۰, \text{Of} = ۰, \text{sign} = ۰$$

$$۲. (1100)_2 - (1010)_2 \rightarrow (1100)_{2c(-4)} + (0110)_6 = ([1]0010)_2$$

$$\text{carry} = ۱, \text{Zero} = ۰, \text{Of} = ۰, \text{sign} = ۰$$

$$۳. (1001)_2 - (0100)_2 \rightarrow (1001)_{2c(-7)} + (1100)_{2c(-4)} = ([1]0101)_2$$

$$\text{carry} = ۱, \text{Zero} = ۰, \text{Of} = ۱, \text{sign} = ۰$$

مثال:

با فرض دو عدد دودویی $x = 1010100$ و $y = 1000011$ تقریق های زیر را انجام دهید.

$$X - Y \quad ۱.$$

حل:

$$(1010100)_2 - (1000011)_2 \rightarrow (1010100)_{2c(-44)} + (0111101)_{261} = ([1]0010001)_{217}$$

$\bullet = \text{sign}$, $\circ = \text{Of}$, $\circ = \text{Zero}$, $\wedge = \text{carry}$

$$Y - X \quad ۲.$$

$$(1000011)_2 - (1010100)_2 \rightarrow (1000011)_{2c(-61)} + (0101100)_{244} = (1101111)_{2-17}$$

$\wedge = \text{sign}$, $\circ = \text{Of}$, $\circ = \text{Zero}$, $\circ = \text{carry}$

پس در هنگام تقریق دو عدد $A - B$ خواهیم داشت:

اگر $A > B$ پس عددی مثبت خواهیم داشت، و فلگ سر ریز با فلگ علامت صفر خواهد بود.
اگر $A < B$ ، پس عددی منفی خواهیم داشت که هم فلگ علامت و هم فلگ اورفلو ۱ خواهد بود.

اگر $A = B$ ، در این صورت هر دو یکسان هستند پس جواب صفر را خواهیم داشت که فلگ Zero روشن خواهد شد.

۲۱.۱ سیستم D۲B

این نوع اعداد به زبان خودمانی، فقط در نقش مبنای دو ظاهر می شوند وگرنه از نظر ماهیتی همان اعداد Decimal هستند، این اعداد زمانی مورد استفاده قرار میگیرند که برای تبدیل اعداد Decimal به آنها جایگاه اعداد برایمان مهم نیست، بلکه معنا و مفهوم آن عدد برایمان مهم است، مثلاً شما شماره دانشجویی یا یک شماره تلفن یک منطقه را در نظر بگیرید، که هر کدام از اعداد نشان دهنده و به معنای خاصی هست، این اعداد هیچ لزومی ندارد که صرفاً دودویی باشند، برای تبدیل اعداد Decimal به اعداد شبه باینری به چهار بیت به ازای هر عدد نیاز است، مانند تبدیل اعداد سیستم Hex. این اعداد قابل درک هستند، نسبت به اعداد باینری فضای بیشتری را میگیرند و اصلاً برای محاسبات جمع و تقریق مناسب نیستند!

۱.۲۱.۱ کد کردن اعداد دهدهی

- کدهای باید به صورت دودویی باشند، زیرا در کامپیوتر همه چیز صفر و یک است.

- کدها فقط نماد یا سمبل نمایش اطلاعات را عوض میکنند و نه مفهوم آن ها را.
 - یک کد دودویی n بیت، 2^n ترکیب ممکن از یک ها و صفرها را داراست.
- کد ها به دو دسته تقسیم می شوند:
- کدهای وزن دار: به هر مکان یک وزن اختصاص داده می شود مثل کد BCD که دارای وزن 8421 است.
- کد های بدون وزن: مثل کد افزودنی ۳، که میتوانیم با اضافه کردن ۳ عدد به کد BCD به کد ۳-Excess رسید.

Decimal Digit	BCD 8421	Excess-3	84-2-1	2*421	Biquinary 5043210
0	0000	0011	0000	0000	0100001
1	0001	0100	0111	0001	0100010
2	0010	0101	0110	0010	0100100
3	0011	0110	0101	0011	0101000
4	0100	0111	0100	0100	0110000
5	0101	1000	1011	1011	1000001
6	0110	1001	1010	1100	1000010
7	0111	1010	1001	1101	1000100
8	1000	1011	1000	1110	1001000
9	1001	1100	1111	1111	1010000

شکل ۳: سایر کدهای Decimal

در سیستم کد، $\bar{1} \bar{2} 4 8$ شما می توانید طبق وزن ها از کد BCD به کد مورد نظر خود برسید.

در سیستم کد، ۲۴۲۱ از صفر تا ۴، وزنمان را از سمت راست تعیین میکنیم، اما از خود پنج به بعد از سمت چپ به عنوان تعیین وزن استفاده خواهیم کرد.

مثال

عدد $(82)_{10}$ را به صورت، مبنای ۲، BCD، ۳-Excess، $\bar{1} \bar{2} 4 8$ ، ۲۴۲۱، کد کنید.

۱. $(82)_{10 \rightarrow 2} = (1010010)_2$

۲. $(82)_{10} = (1000, 0010)_{BCD}$

۳. $(82)_{10} = (1000, 0010)_{BCD \rightarrow (excess-3)+3} = (1011, 0101)_{Excess-3}$

۴. $(82)_{10} = (1000, 0010)_{BCD \rightarrow 8421} = (1000, 0110)_{8421}$

$$(82)_{10} = (1000, 0010)_{BCD \rightarrow 2421} = (1110, 0010)_{2421} \quad ۵.$$

عدد $(10111001)_2$ را به صورت، مبنای ۱۰، BCD، Excess-۳، 8421 ، 2421 ، کد کنید.

$$(10111001)_{2 \rightarrow 10} = (185)_{10} \quad ۱.$$

$$(185)_{10} = (0001, 1000, 0101)_{BCD} \quad ۲.$$

$$(185)_{10} = (0001, 1000, 0101)_{BCD \rightarrow (excess-3)+3} = \quad ۳.$$

$$(0100, 1011, 1000)_{Excess-3}$$

$$(185)_{10} = (0111, 1000, 1011)_{8421} \quad ۴.$$

$$(185)_{10} = (0001, 1110, 1011)_{2421} \quad ۵.$$

عدد دودویی $(10111001)_2$ را به صورت BCD بازنویسی کنید.

$$(10111001)_2 = (185)_{10} = (0001, 1000, 0101)_{BCD}$$

۲.۲۱.۱ جمع دو عدد BCD

برای جمع اعداد BCD فقط کافیست که معادل Decimal آنها را با هم جمع کنیم و بعد از آن حاصل را به BCD تبدیل خواهیم کرد.

۲۲.۱ کد گری یا کد انعکاسی

زمانی پیش می آید که عدد ورودی ما مثلاً 0111 که در مبنای ۱۰ برابر با هفت است به عنوان ورودی وارد شده و مثلاً میخواد بشود ۸ یا 1000 در طی این تبدیل ممکن است تاخیر ها و Delay های پیش آمده باعث شود به این تبدیل چندین بیت با هم تفاوت و فاصله ایجاد شود، این باعث میشود که CPU زحمت زیادی بکشد در فرایند های تبدیل، به همین خاطر سیستم عددی به نام Gray به وجود آمد که در اثر بوجود آمدن این Delay ها فاصله بین هر عدد تنها یک بیت باشد، کد گری میتواند یک ست ۲ بیتی، ۳ بیتی، ۴ بیتی، ۵ بیتی و غیره باشد، که همه این ها با هم تنها یک بیت فاصله دارند، منظور از یک بیت فاصله یعنی بعد از تبدیل عدد اصلی از حالت بومی خودش به کد گری فقط و فقط یک بیت تفاوت است.

برای تبدیل کد باینری به کد گری خیلی راحت میتوانیم از طریق شباهت و تفاوت پیش برویم، عددی که بیشترین ارزش را دارد (اولین عدد از سمت چپ) یا عدد MSB را بدون

هیچ دستکاری می نویسیم، بعد از آن خود آن عدد را با عدد کناری خودش بررسی میکنیم، اگر بایکدیگر مشابه بودند، عدد صفر را مینویسیم، اگر باهم تفاوت داشتند عدد یک را مینویسیم. همین فرایند را تا انتها پیش میرویم. برای مثال:

$$Same = 0$$

$$Difference = 1$$

$$(110111)_2 \rightarrow Gray = (101100)_{Gray}$$

$$(111)_2 \rightarrow Gray = (100)_{Gray}$$

$$(10101101)_2 \rightarrow Gray = (11111011)_{Gray}$$

۱.۲۲.۱ تبدیل باینری ۴ بیتی به گری ۴ بیت

Binary	Gray
۰۰۰۰	۰۰۰۰
۰۰۰۱	۰۰۰۱
۰۰۱۰	۰۰۱۱
۰۰۱۱	۰۰۱۰
۰۱۰۰	۰۱۱۰
۰۱۰۱	۰۱۱۱
۰۱۱۰	۰۱۰۱
۰۱۱۱	۰۱۰۰
۱۰۰۰	۱۱۰۰
۱۰۰۱	۱۱۰۱
۱۰۱۰	۱۱۱۱
۱۰۱۱	۱۱۱۰
۱۱۰۰	۱۰۱۰
۱۱۰۱	۱۰۱۱
۱۱۱۰	۱۰۰۱
۱۱۱۱	۱۰۰۰

۲.۲۲.۱ تبدیل کد گری به کد باینری

برای تبدیل کد گری به باینری میبایست عدد MSB را نوشته (همانطوری) و بعد از همان به بعد با اعداد بعدی مقایسه میشود

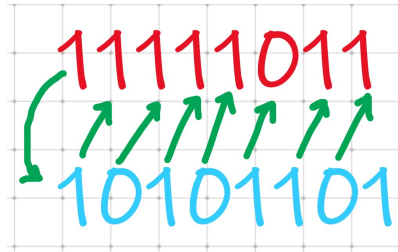
$$Same = 0$$

$$Difference = 1$$

$$(101100)_{Gray \rightarrow 2} = (110111)_2$$

$$(100)_{Gray \rightarrow 2} = (111)_2$$

$$(11111011)_{Gray \rightarrow 2} = (10101101)_2$$



شکل ۴: Gray ۲ Binary

۲۳.۱ ASCII Codes

از کدهای اسکلی زمانی استفاده میشود که ما بخواهیم نماد ها، کارکتر های مختلف، اعداد، نماد های زبان های مختلف و غیره را در سیستم خود نمایش بدهیم. این سیستم کدینگ در حالت کلی ۸ بیتی میباشد. در ۵۰ سال گذشته این سیستم ۶ بیتی بوده و توانایی نمایش حروف فقط بزرگ انگلیسی را داشت، اما بعد از آنکه این سیستم ۸ بیتی شد توانایی هایی که در بالاتر توضیح داده شد را دارا میباشد، در این سیستم کدینگ ۳ بیت به عنوان ستون ها و ۴ بیت به عنوان سطر ها مورد استفاده قرار گرفته، و بیت آخر که بیت مشخص کنند ماهیت کارکتر میباشد، اگر وضعیت این بیت ۰ باشد توانایی نمایش اعداد، کارکتر های کنترلی و یکسری علائم و نماد های مختلف مانند (*BackText*)، ؛، "، را دارا میباشد که به این دسته ها Standard ASCII گفته میشود، که تعداد کارکتر های آن ۱۲۸ تا می باشد. اما اگر این بیت هشتم ۱ باشد علاوه بر نمایش حروف انگلیسی میتواند حروف اضافه بر زبان انگلیسی را نمایش دهد مانند در زبان ترکی و آلمانی (اوملات) و غیره. که این دسته در حقیقت ASCII Extended نامیده میشود، که این دسته هم دارای ۱۲۸ تا کارکتر میباشد.

دسته بندی های کدهای الفبایی یا بطور کلی *Textual*:

Alphabet: a...z and Z ... A

Digits: ۰...۹

Special: Symbols '، "، ؛، .، .

non-printable: NULL DEL،

مهم ترین بخش جدول کدهای اسکلی مربوط به ستون اول و دوم و ابتدای ستون سوم،

مربوط به کارکتر های کنترلی میشود، حالا این کاراکتر های کنترلی چیست این مهم است:

ستون اعداد				سطر اعداد				
نوع	7	6	5	4	3	2	1	0

شکل ۵: بیت های کد اسکی

	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	,	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	\$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(8	H	X	h	x
1001	HT	EM)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[k	{
1100	FF	FS	,	<	L	\	l	
1101	CR	GS	=	=	M]	m	}
1110	O	RS	>	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

شکل ۶: ASCII Codes Table

در گذشته ماشین های تله تایپی وجود داشتند که نهایت سرعت آنها ۱۰ بیت در ثانیه بود و کلا یکسری دستگاه های مکانیکالی بودند. این دستگاه های برای ارتباط با سیستم های دیگر از یکسری کرکتر ها استفاده میکردند تا بعضی از جریان ها مورد بررسی قرار بگیرد، برای مثال وقتی به این سیستم ها قرار بود یکسری کلمات و جملات را بنویسند ارتباط آنها مانند ارتباط ما با کامپیوتر های امروزی نبود و با تاخیری کلمات چاپ میشدند، این سیستم ها برای اینکه ببینند آیا سیستم مکانیکی توانسته کراکتر های قبلی را بنویسد که کارکتر های بعدی را ارسال کند، یا اینکه دست نگهدارد که آن سیستم مکانیکی کار قبلی اش را تمام و سپس بروی کار جدید برود، در این میان از مجموعه ای از کارکتر های کنترلی استفاده میکردند که کنترل جریان را هدایت می نمودند. بطور کلی حفظ کردن نام برخی از آنها دشوار میباشد به همین دلیل در سیستم های امروزی از کنترل ترکیبی بجای استفاده از آن کرکتر ها استفاده کردند، برای مثال کرکتر ESC معادل $ctrl +$ است.

برای بدست آوردن نام و کلا تایپ این نوع کرکتر ها باید دقت داشته باشیم که از سمت چپ اعداد خود را قرار بدهیم، **اولین دسته عدد که سه بیتی** است ستون را منظور دارد، **دومین دسته که چهار بیتی** است بعد از سه بیت ستون نوشته شده و قرار میگیرد و در انتها برای اینکه بررسی کنیم آیا این عدد یک Extended است یا Standard از سمت چپ بیت MSB را یکی اکستند میکنیم که تا نوع آن مشخص و ۸ بیتی شود. و بعد در انتها میتوانیم کد بانری بدست آمده را به دسیمال تبدیل کرده و معادل آن عدد را در کد اسکی بررسی کنیم. مانند:

$$1100111 \rightarrow (110, 111)_{(2) \rightarrow 10} = 103_g$$

	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	p	
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	\$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(8	H	X	h	x
1001	HT	EM)	9	I	Y	i	y
1010	LF	SUB	+	:	J	Z	j	z
1011	VT	ESC	;	<	K	[k	{
1100	FF	FS	,	=	L	\	l	
1101	CR	GS	-	>	M]	m	}
1110	O	RS	.	?	N	^	n	~
1111	SI	US	/		O	_	o	DEL

شکل ۷: ASCII Codes Sample

یا اینکه میتوانیم بطور کلی اینگونه پیش برویم که ۳ بیت اول از ستون و ۴ بیت دوم از سطر جدول را نقطه یابی کنیم که معادل چه حرفی میشود. تصویر بالا

۲۴.۱ کدهای تشخیص و تصحیح خطا

هنگام انتقال داده ها ممکن است در آنها خطایی به علت تداخل های الکترومغناطیسی، حرارت زیاد و غیره به وجود آید. میتوان کدهایی طراحی کرد که خطا را تشخیص و تصحیح کنند. یکی از ساده ترین روش های تشخیص خطا استفاده از بیت توازن یا **Parity** است. میتوان به هر کلمه یک بیت اضافه کرد به طوری که تعداد بیت های یک آن مثلا فرد شود

۱.۲۴.۱ کدهای تشخیص خطا

سیستم توازن، یکی از ساده ترین روش ها برای تشخیص خطاست، که یک بیت توازن به اطلاعات اضافه می شود.

در حالت کلی دو نوع توازن وجود دارد، توازن زوج و توازن فرد.

توازن زوج Even Parity

یک بیت به اطلاعات اضافه میشود تا تعداد کل ۱ ها کد زوج جود.

$$0110010 = [1]0110010$$

$$0100100 = [0]0100100$$

توازن فرد Odd Parity

$$0110011 = [1]0110011$$

$$0100101 = [0]0100101$$

یک بیت به اطلاعات اضافه میشود تا تعداد کل ۱ های کد فرد شود. تا تعداد صفر ها و یک ها به تعادل و برابری برسد.

نوع دیگری از Parity وجود دارد که به تعداد یک ها توجه می شود، یعنی اگر سیستم

پربیتی زوج بودیم تعداد یک ها بایستی زوج باشند، اما اگر در سیستم پربیتی فرد بودیم باید تعداد یک ها یک دسته فرد باشند، برای اینکار در سیستم پربیتی زوج تعداد یک ها اگر فرد بود یک، ۱ را اضافه خواهیم کرد اما اگر زوج بود ۰ را وارد میکنیم. در فرد هم همینگونست اگر سیستم فرد تا ۱ داشت ۰ اگر سیستم زوج تا ۱ داشت یک، ۱ اضافه میکنیم تا نظم زوجی یک ها را بهم بزنیم و آنها را فرد کنیم!

EvenParity : $100011 = 1 \rightarrow 1000111$

OddParity : $100011 = 0 \rightarrow 1000110$

۲۵.۱ Overlapping Parity or Block Parity

یک مجموعه ای از دیتا ها که بایستی به دو صورت افقی و عمودی از آنها پربیتی گرفته شود.

000	111	Even parity
101	011	0
110	000	0
000	111	1
111	111	0
100	100	0

شکل ۸: Block Parity

در صورتی که در یکی از بیت های مجموعه داده های بالا خطایی ایجاد شود میتوان براحتی دریافت که در آن قسمت، هم به صورت افقی و هم به صورت عمودی نتیجه پربیتی با نتیجه قبلی برابر نخواهد بود. و برای نشان داده اشتباه در آن متقطه باید به صورت یک آرایه دو بعدی طور عمل کنیم که بعد اول در مورد سطر و بعد دوم در مورد ستون میباشد، مانند اشکال در $M_{[5][3]}$.

۲۶.۱ Parity Checksum

در این نوع پربیتی ما سه نوع داریم:

Single-precision checksum ۱.۲۶.۱

در این نوع از checksum دیتا های دریافتی را با هم جمع میکنیم، و حاصل بدست آمده اگر همراه با Carry بود از آن صرف نظر خواهیم کرد.

Double-precision checksum ۲.۲۶.۱

در این نوع، علاوه بر اینکه کری را به همراه حاصل مینویسیم بایستی بررسی کنیم که وجود کری آیا دسته بیت ها را در توانی از دو قرار داده است یا خیر، اگر نبود خودمان دستی این کار را انجام میدهیم

Residue checksum ۳.۲۶.۱

در این نوع از checksum، مانند Single-precision عمل میکنیم با این تفاوت که کری بدست آمده را در حاصل جمع میکنیم

HoneyWell checksum ۴.۲۶.۱

در این در صورتی که ۴ عدد دیتا را داشته باشیم دو به دو دسته ها را بهم می چسبانیم.

0000	0000	0000	
0101	0101	0101	
1111	1111	1111	00000101
0010	0010	0010	11110010
<hr/>			
0110	00010110	0111	11110111
Single-Precision	Double-Precision	Residue	Honeywell

شکل ۹: Parity checksum

۲۷.۱ کد همینگ یا Hamming Code

بطور کلی، کد همینگ، کد تشخیص و تصحیح خطا می باشد. که در مخابرات مورد استفاده قرار میگیرد. منظور از مخابرات یعنی ارسال و دریافت اطلاعات بین دو قسمت مبدا و مقصد است. که برای اولین بار به افتخار ریچارد همینگ معرفی شد. در مفهوم، در کد همینگ برای

شناسایی و تصحیح خطا دسته ای از کدها در اطلاعات ارسال میشود، که به این کدها کد همینگ می گویند.

۱.۲۷.۱ فاصله همینگ

زمانی که ما کد تصحیح شده را در کنار کدی با بیت های غلط قرار می دهیم به اختلاف بین این دو عدد، فاصله همینگ گفته می شود.

$$A = 111001 \rightarrow \text{Correct}$$

$$A_{\text{uncorrect}} = 010110 \rightarrow \text{uncorrect} \rightarrow \text{distance} = 5$$

نکته:

- کدی که فاصله d دارد میتواند $d - 1$ خطا را تشخیص دهد.
- کدی که فاصله d دارد میتواند $\lfloor \frac{d-1}{2} \rfloor$ را تصحیح کند.

۲.۲۷.۱ نحوه بدست همینگ کد و تشخیص و تصحیح خطا

برای بدست آوردن کد همینگ یک عدد میبایستی به شکل زیر عمل کنیم:

- اول باید بررسی کنیم که اعداد دریافت شده چند بیتی است، (تشخیص از تعداد ارقام عدد ارسالی)

- بعد از بدست آوردن تعداد ارقام عدد ارسالی، از سمت چپ باید کدهایی تحت عنوان Code Parity را در نظر داشته باشیم، هر کدام از این کدهای Parity توانی از دو خواهند بود. یعنی

$$2^0, 2^1, 2^2 \text{ Or } 1, 2, 4, 8, 16, 32, \text{ etc}$$

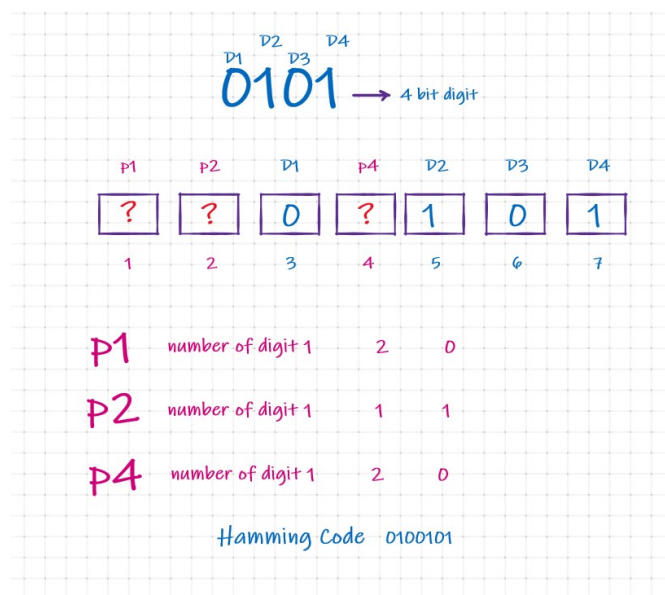
- باتوجه به قاعده بالا، از سمت چپ جایگاه های مثلا ۱ و ۲ و ۴ و ۸ و غیره را برای کد های Parity نگه میداریم، و اعداد داده اصلی را لا به لای این اعداد Parity خواهیم نوشت!

با یک مثال نحوه بدست آوردن کد همینگ را متوجه خواهید شد:

کد همینگ برای عدد ۰۱۰۱ به چه شکل خواهد بود؟

توضیح حل مثال بالا:

در این مثال ما چهار بیت داریم، و کدهای parity ۱ و ۲ و ۴ خواهیم داشت، با توجه به هر



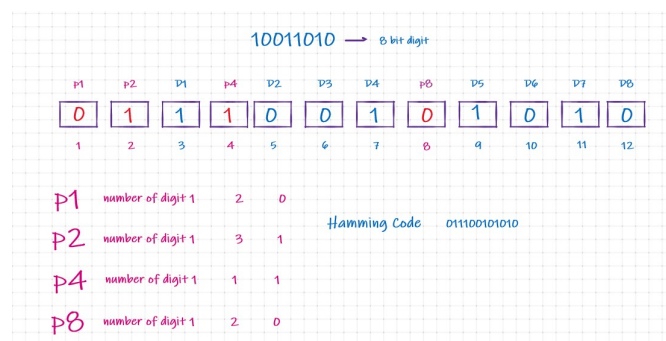
شکل ۱۰: مثال بدست آوردن کد همینگ

عدد بلاک های خالی برای قسمت های ۱ و ۲ و ۴ خواهیم گذاشت تا با روشی مناسب اعداد مناسب آنها را بدست بیاوریم و در قسمت های خالی یا بین این پرتی ها دیتای اصلی خود را خواهیم گذاشت، بعد از تنظیم تمام جایگاه ها و قرار دادن اعداد اصلی در جایگاه مناسب خودشان، با روشی ساده میتوان اعداد پرتی را بدست آورد، اینکه شما میتوانید از جایگاه اول که ۱ است یک در میان اعداد ۱ و ۰ را در نظر بگیرید، صفرها مهم نیستند بلکه باید تعداد یک ها را داشته باشیم، اگر تعداد یک ها زوج بود آن پرتی ۰ خواهد بود اگر فرد بود ۱ می باشد، بعد از جایگاه اول به جایگاه دوم یعنی ۲ میریم، که میبایست همان کار اول را از دو، با تفاوت ۲ در میان انجام دهیم تا مقدار صفر یا یک آن را بدست بیاوریم. بعد به جایگاه چهارم میرویم و بایستی از عدد جایگاه ۴، ۴ در میان تعداد یک ها را مورد بررسی قرار دهیم.

نکته: گاهی ممکن است به ۸ در میان یا بیشتر از آن، هم برخورد کنید، اگر در انتها به اندازه مناسب n در میان، عددی وجود نداشته باشد، همان تعدادی که وجود دارند را میتوان به عنوان دسته آخر مورد بررسی قرار داد.

مثال:

کد همینگ برای عدد ۱۰۰۱۱۰۱۰ به چه شکل خواهد بود؟



شکل ۱۱: مثال بدست آوردن کد همینگ

۳.۲۷.۱ پیدا کردن عدد غلط و تصحیح آن

نکته دیگری هم هست!

ممکن است عددی را به شما دهند که یا اشتباه است یا درست، که شما بایستی با عمل همینگ درستی یا نادرستی آن عدد ارسالی را بررسی کنید.

مثال:

در عدد ۰۱۱۰۱۰۱ مشخص کنید که کجا خطا رخ داده است؟
در این مثال دیگر کاری به اضافه کردن جایگاه نداشته باشید با این فرض پیش بروید که تمام جایگاه ها همانی است که میدانید:

$$0_{p1}1_{p2}10_4101$$

$$p1 : \text{number of } 1 : 3 \rightarrow 1$$

$$p2 : \text{number of } 1 : 2 \rightarrow 0$$

$$p4 : \text{number of } 1 : 2 \rightarrow 0$$

با توجه فرایندی که در بالا صورت گرفت میتوان نتیجه گرفت که پرتی های بدست آمده با عدد نوشته شده مسئله در جایگاه های مناسب، یا درست است یا غلط، که مشخص کردیم که جایگاه اول و دوم غلط نوشته شده، این موضوع غلط بودن این عدد را نشان نمیدهد، برای بدست آوردن اشکال در عدد بایستی جایگاه اعداد غلط مانند جایگاه یک و دو را باهم جمع کنیم که می شود ۳، آنگاه از این نتیجه میتوان دریافت که در جایگاه سوم عدد درستی درج نشده، اگر صفر است آنرا یک میکنیم، اگر یک است آنرا صفر میکنیم. نکته ای که لازم است در انتها به آن اشاره کنم آن است که در هنگام شمارش تعداد یک های، اگر جایگاهی از خودش یک داشت آنرا حساب نخواهیم کرد.

عدد درست، ۰۱۰۰۱۰۱

مثال:

عدد ۰۱۱۱۰۰۱۰۱۱۱۰ درست دریافت شده یا غلط؟ در صورت تشخیص بیت اشتباه آنرا مشخص کنید:

$0_{p1}1_{p2}11_{p4}0010_{p8}1110$

$p1 : \text{numberof}1 : 4 \rightarrow 0 \text{Correct}$

$p2 : \text{numberof}1 : 4 \rightarrow 0 \text{Uncorrect}$

$p4 : \text{numberof}1 : 1 \rightarrow 1 \text{Correct}$

$p8 : \text{numberof}1 : 3 \rightarrow 1 \text{Uncorrect}$

جایگاه های غلط را باهم جمع میکنیم تا جایگاهی که واقعا بیت اشتباهی دارد را تشخیص دهیم، جایگاه ۲ با ۸ که میشود ۱۰:

$0111001011_{\text{uncorrectbit}}10$

عدد درست ۰۱۱۱۰۰۱۰۱۰۱۰

۴.۲۷.۱ نحوه محاسبه تعداد پرتی های همینگ

یک فرمول ساده در این رابطه وجود دارد $2^r - 1 \geq (n) \text{bitnumber} + r$ که در آن رقم توان، تعداد پرتی ها و در مقابل تعداد ارقام عدد دریافتی + آن تعداد پرتی

برای مثال عدد دودویی داریم که ۱۱ بیت است برای پیدا کردن این که این رقم چند تا جایگاه پرتی خواهد داشت به صورت زیر خواهد بود:

$$2^3 - 1 \geq 11 + 3 \text{false}$$

$$2^4 - 1 \geq 11 + 4 \text{true}$$

پس در ۱۱ بیت ۴ بیت پرتی خواهیم داشت.

۲ جبر بول - ساده سازی - EPI PI،

اساس کار مدار منطقی، جبر بول یا جبر مجموعه ها یا جبر سویچ است. جبر بول به خاطر آقای George Boole نام گذاری شده است، که برای بیان منطق انسان از آن استفاده نمود. بعد از مدتی Shannonن جبر کلیدی یا Switching Algebra را برای نمایش مدارهای کلیدی دو حالتی معرفی کرد یعنی جبر دو ارزشی.

۱.۲ Not Or And truth table

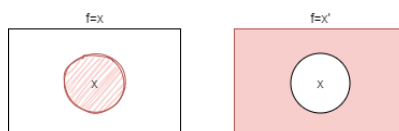
A	B	$A \cdot B$	$A + B$	$\neg A$
۰	۰	۰	۰	۱
۱	۰	۰	۱	۰
۰	۱	۰	۱	۱
۱	۱	۱	۱	۰

۲.۲ استفاده از نمودار ون برای نمایش منطقی

ما میتوانیم از نمودار ون برای نمایش وضعیت های true و false یا ۱ و ۰ استفاده کنیم.

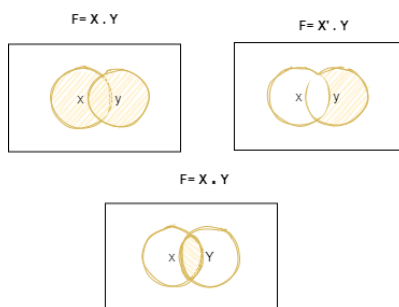
۱.۲.۲ Single Variable

Single variable



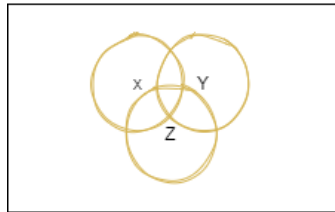
۲.۲.۲ Double Variable

Double variable



Three Variable ۳.۲.۲

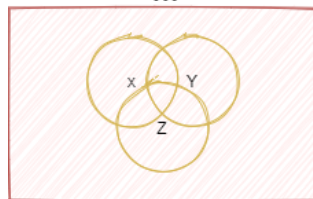
Three variable



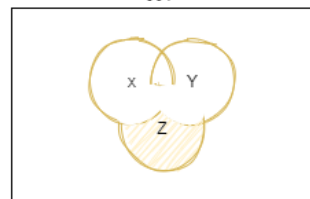
جدول زیر نشان دهنده وضعیت سه متغیر، (X, Y, Z) است، هر کدام از وضعیت ها را در ون نمایش دهید.

X	Y	Z
۰	۰	۰
۰	۰	۱
۰	۱	۰
۰	۱	۱
۱	۰	۰
۱	۰	۱
۱	۱	۰
۱	۱	۱

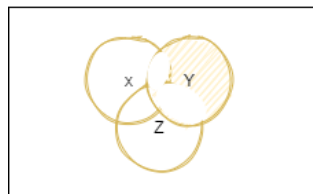
000



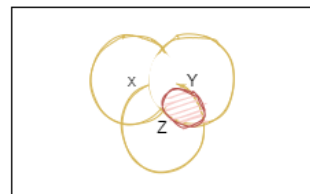
001

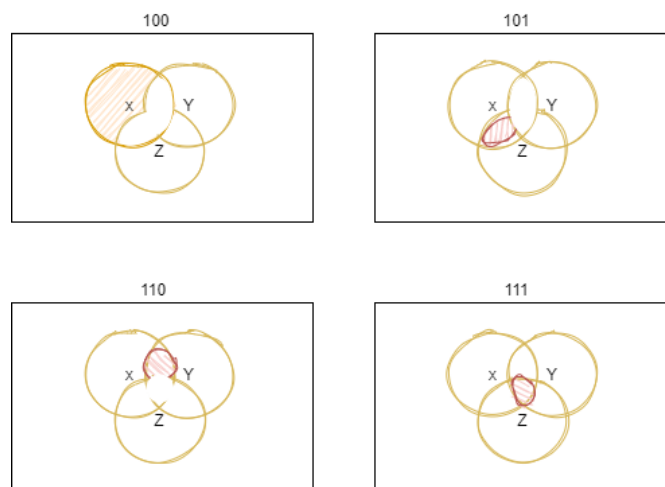


010



011





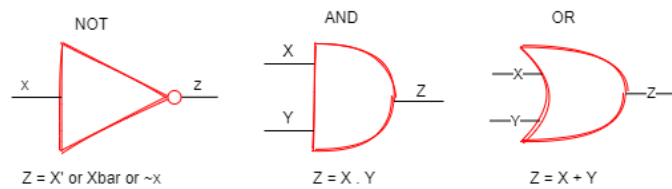
۳.۲ on-set and off-set

به جدول زیر توجه کنید:

Bit Pattern	X	Y	Z	Prime no.	status
۰	۰	۰	۰	۰	off-set
۱	۰	۰	۱	۰	off-set
۲	۰	۱	۰	۱	on-set
۳	۰	۱	۱	۱	on-set
۴	۱	۰	۰	۰	off-set
۵	۱	۰	۱	۱	on-set
۶	۱	۱	۰	۰	off-set
۷	۱	۱	۱	۱	on-set

جدول بالا در قسمت نتیجه، مشخص میکند عدد در هر سطر اول است یا نه، به هر سطری که مقدار عدد اول بودن آن، برابر با صفر شده است off-set و به هر سطری که اول بودن آن یک شده است on-set گفته میشود.

۴.۲ شمایل گیت های تانکون گفته شده



XNOR Xor NOR NAND truth table ۵.۲

A	B	NAND \uparrow	NOR \downarrow	XOR \oplus	XNOR \odot
۰	۰	۱	۱	۰	۱
۱	۰	۱	۰	۱	۰
۰	۱	۱	۰	۱	۰
۱	۱	۰	۱	۰	۱

راهی دیگر برای اثبات **Xor** یا **Xnor** وجود دارد:

$$XOR = (\bar{A}.B) + (\bar{B}.A) \equiv (\sim A \wedge B) \vee (A \wedge \sim B)$$

$$XNOR = (A.B) + (\bar{A}.\bar{B}) \equiv (A \wedge B) \vee (\sim A \wedge \sim B)$$

۶.۲ چند مورد از اصطلاحات

Term ۱.۶.۲

به هر یک از Variable ها، Term گفته میشود.

Product Term ۲.۶.۲

به هر کدام از ضرب بین عملوند ها یا and بین عملوند ها **Products Term** گفته میشود.
مانند:

$X'Y$ or XY

Uncorrect:

$(XY)'$

$(XY) + Z$

Sum Term ۳.۶.۲

به هر کدام از جمع بین عملوند ها یا or بین آنها **Sum Term** گفته میشود.

$(X + Y)$

Uncorrect:

$(X + Y)'$

$Z'W + Y$

Min Term ۴.۶.۲

به هر **Product Term** گفته میشود که حداقل یکی از آنها True یا ۱ باشند.

Max Term ۵.۶.۲

به **Sum Term** هایی گفته میشود که حداقل در یک جا تابع خروجی صفر را داشته باشد، و هر جایی که ورودی در آن برابر یک بود باید به صورت not نشان داده شود و آن ورودی که صفر است را بایستی به صورت عادی در بولین فرم نمایش دهیم.

SoP Sum of Product ۶.۶.۲

به **Product Term** هایی که بین آنها عمل جمع یا or اتفاق افتاده است، **Sum of Product** یا **SoP** گفته می شود.

$$\overbrace{(\bar{A}.B)_{ProductsTerm} + (\bar{B}.A)}^{SumofProducts} \quad (1)$$

Correct Form of SoP

$$[\bar{X}.\bar{Y}.Z + W + \bar{a}.b.c] \quad (2)$$

$$[W + \bar{X}.Z.Y + \bar{y}.z] \quad (3)$$

Uncorrect Form of SoP

$$[\bar{X}.Y.Z + W.(A + B)] \quad (4)$$

$$[X.Y + \bar{Y}.Z] \quad (5)$$

PoS Product of Sum ۷.۶.۲

به هر **Sum Term** که بین آنها عمل ضرب یا And صورت گرفته بر عکس SoP در حقیقت PoS یا **Products of Sum** گفته می شود

$$\overbrace{(\bar{A} + B)_{SumTerm} . (\bar{B} + A)}^{ProductsofSum} \quad (6)$$

Correct Form PoS

$$[(X + \bar{Y} + Z).(\bar{W} + Z).(A)] \quad (7)$$

$$[\bar{Z} \cdot (X + Y) \cdot (\bar{W} + Y)] \quad (۸)$$

Uncorrect Form PoS

$$[(\bar{X} + Y \cdot (X + W))] \quad (۹)$$

$$[(X + Y) \cdot (X + Z)] \quad (۱۰)$$

نکته:

Term ها در این حالت آنها به صورت Literal هستند که هر دوی Sum Term و Product را شامل می شوند.

۸.۶.۲ ۱ خودت را امتحان کن

Expression	Sum Term / Product Term / Both / Neither
$W * X * (Y * Z)'$	Neither
$W * X * (Y * Z)$	Product Term
$W + XY + Z$	Neither
$W + Y$	Sum Term
$(W + Y + Z)'$	Neither
W	Both

۹.۶.۲ ۲ خودت را امتحان کن

Expression	SoP / PoS / Both / Neither
$W \cdot X \cdot (\bar{Y} \cdot Z) + X \bar{Y} Z + W$	Neither
$XY + XZ + (\bar{W}YZ)$	SoP
$(X + \bar{Y} + Z)(W + X)$	PoS
$(W + Y)X(\bar{W} + Z)$	PoS
$\bar{W}Y + \bar{W}Y + X\bar{Y}$	SoP
$[W + X + Y]$	Both

Dual Low ۷.۲

در قانون دوئال در Product Term ها چند چیز تغییر خواهد کرد:

$$A.B \leftrightarrow A + B$$

$$0 \leftrightarrow 1$$

برای مثال:

$$f(a.b) \rightarrow f^* = a + b$$

$$f = a.1 + \bar{b}.c \rightarrow f^* = (a + 0).(\bar{b} + c)$$

$$f() = A \oplus B = (\bar{A}.B) + (A.\bar{B}) \rightarrow f^*() = (\bar{A} + B).(A + \bar{B})$$

Self Dual ۱.۷.۲

در قانون Self Dual به چیزی اشاره میکند که صحت دوئال آن با حالت عادی آن برابر است برای بررسی تعداد دفعاتی که بایستی true و false یا ۱ و ۰ بگذاریم، میتوانیم به تعداد عملوندها توجه کنیم و به عنوان توانی از پایه دو استفاده کنیم که ببین چند حالت میتواند این تابع داشته باشد

$$2^{number of Operand} \quad (۱۱)$$

مثال:

ثابت کنید که تابع $f = a.b + a.c + b.c$ با هم خود دگان یا سلف دوئال است:

$$f^*() = (a + b).(a + c).(b + c)$$

a	b	c	$f()$	$f^*()$
۰	۰	۰	۰	۰
۰	۰	۱	۰	۰
۰	۱	۰	۰	۰
۰	۱	۱	۱	۱
۱	۰	۰	۰	۰
۱	۰	۱	۱	۱
۱	۱	۰	۱	۱
۱	۱	۱	۱	۱

۲.۷.۲ اصل Duality

اگر دو تابع با هم سلف دوئال باشند، اگر خاصیتی برای حالت عادی تابع صدق کند برای حالت دوئال آن هم صادق می‌باشد.

۸.۲ خاصیت‌های گزاره‌ها

• خاصیت خودتوانی:

$$A.A = A$$

$$A + A = A$$

• خاصیت جذبی:

$$A.(A + B) = A$$

$$A + (A.B) = A$$

• خاصیت جذبی:

$$A.(A + B) = A$$

$$A + (A.B) = A$$

• خاصیت جابجایی:

$$A.B = B.A$$

$$A + B = B + A$$

• خاصیت شرکت پذیری:

$$A.(B.C) = (A.B).C$$

$$A + (B + C) = (A + B) + C$$

• خاصیت توزیع پذیری:

$$A.(B + C) = (A.B) + (A.C)$$

$$A + (B.C) = (A + B).(A + C)$$

• خاصیت نقیض نقیض:

$$\overline{\overline{A}} = A$$

• خاصیت متمم:

$$\overline{A}.A = 0$$

$$\overline{A} + A = 1$$

• خاصیت همانی:

$$A.1 = A|A + 1 = 1$$

$$A.0 = 0|A + 0 = A$$

• خاصیت دمورگان:

$$\overline{(A.B)} = (\overline{A} + \overline{B})$$

$$\overline{(A + B)} = (\overline{A}. \overline{B})$$

۹.۲ توابع با خروجی چندتایی

برخی توابع در این رابطه ممکن است که بیش‌تر از یک خروجی داشته باشند، که به دو صورت مورد بررسی قرار میگیرند:

• بر اساس تعداد یک‌ها

• بر اساس ایدی اولین ۱ که آمده است

در مورد دوم بیشتر روی قسمت عدد MSB توجه میکنیم، به محض اینکه این عدد یک شد ایدی مکان آنرا به صورت عادی در نظر گرفته و در قسمت باینری آنرا مینویسیم.

X	Y	Z	R_1	R_2	X	Y	Z	R_1	R_2
۰	۰	۰	۰	۰	۰	۰	۰	۰	۰
۰	۰	۱	۰	۱	۰	۰	۱	۰	۱
۰	۱	۰	۰	۱	۰	۱	۰	۱	۰
۰	۱	۱	۱	۰	۰	۱	۱	۱	۰
۱	۰	۰	۰	۱	۱	۰	۰	۱	۱
۱	۰	۱	۱	۰	۱	۰	۱	۱	۱
۱	۱	۰	۱	۰	۱	۱	۰	۱	۱
۱	۱	۱	۱	۱	۱	۱	۱	۱	۱

Counter of ۱s

Max on input id ۱ (MSB)

سوال:

خروجی تابع زیر را بر اساس ایدی MSB ها بررسی کنید

W	X	Y	Z	R_1	R_2	R_3
•	•	•	•	•	•	•
•	•	•	۱	•	•	۱
•	•	۱	•	•	۱	•
•	•	۱	۱	•	۱	•
•	۱	•	•	•	۱	۱
•	۱	•	۱	•	۱	۱
•	۱	۱	•	•	۱	۱
•	۱	۱	۱	•	۱	۱
۱	•	•	•	۱	•	•
۱	•	•	۱	۱	•	•
۱	•	۱	•	۱	•	•
۱	•	۱	۱	۱	•	•
۱	۱	•	•	۱	•	•
۱	۱	•	•	۱	•	•
۱	۱	۱	•	۱	•	•
۱	۱	۱	۱	۱	•	•

Gates ۱۰.۲

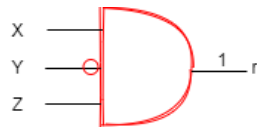
گیت ها میتوانند بیشتر از دو ورودی داشته باشند اما توابع در همه آنها مشابه است.
 در عملگر And: خروجی ها همه یک هستند در صورتی که ورودی ها ۱ باشند.
 در عملگر OR: خروجی ۱ است در صورتی که یکی از ورودی ها ۱ باشد.

X	Y	Z	F	X	Y	Z	F
•	•	•	•	•	•	•	•
•	•	۱	•	•	•	۱	۱
•	۱	•	•	•	۱	•	۱
•	۱	۱	•	•	۱	۱	۱
۱	•	•	•	۱	•	•	۱
۱	•	۱	•	۱	•	۱	۱
۱	۱	•	•	۱	۱	•	۱
۱	۱	۱	۱	۱	۱	۱	۱

AND

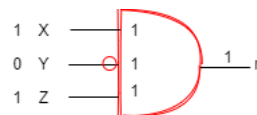
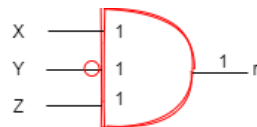
OR

ممکن است در مسئله ای از شما بخواهند که گیتی را با رسم جدول درستی، بررسی یا Decode / Checkers کنید، به همین خاطر مانند زیر عمل میکنیم.



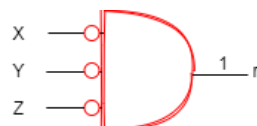
گیت بالا در ورودی دوم، هر مقداری وارد شود نقیض خواهد شد، و از آنجایی که میدانیم، گیت بالا فقط در زمانی جواب ۱ را میدهد که تمام ورودی ها یک شوند. به همین خاطر در درون گیت AND هر سه ورودی را یک میکنیم، اما در ورودی های خارج از گیت را باتوجه به نقاط یا حباب هایی که بر روی ورودی گیت کشیده شده، مقادیر مورد نظر را خواهیم نوشت.

X	Y	Z	F
۰	۰	۰	۰
۰	۰	۱	۰
۰	۱	۰	۰
۰	۱	۱	۰
۱	۰	۰	۰
۱	۰	۱	۱
۱	۱	۰	۰
۱	۱	۱	۰

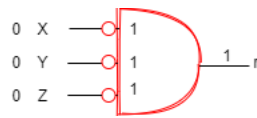


پس بخاطر نقیض بودن ابتدا ورودی اول، برای رسیدن به عدد ۱، عدد ۰ را قرار دادیم که نقیض صفر یک خواهد شد و گیت AND ما در نقطه ۱۰۱ برابر با یک خواهد بود.

مثال دوم:



جواب:



X	Y	Z	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

سه راه برای نمایش یک تابع وجود دارد:

- Equation
- Schematic
- Truth table \rightarrow unique

که حالت جدول درستی معیار است چرا که تنها یک معنا دارد و از آن میتوان به روش های دیگر رسید و آنها را پیاده سازی نمود.

مسئله:

ماشینی داریم که خودکار است، اگر در ورودی آن و چراغ های جلو روشن باشد، بوق میزند یا اگر در ورودی آن باز باشد و سویچ (کلید) روی حالت on باشد باز بوق میزند، تابع این مسئله را بنویسید.^۱

^۱ ساده سازی بین هر کدام از Term ها مانند فاکتورگیری در ریاضیات در این قسمت Adjacency گفته میشود

H	D	K	F		H	D	K	F
.
.	.	۱
.	۱	.	.		.	۱	.	.
.	۱	۱	۱		.	۱	۱	۱
۱	.	.	.		۱	۱	.	۱
۱	.	۱	.					
۱	۱	.	۱					
۱	۱	۱	.					

$$H.D + D.K \text{ or } H(D + K) \quad (۱۲)$$

بیلی دوست داره که فقط پیزی با قیمت ۵ دلار بخورد! یک پیزای ۵ دلاری میتواند فقط یکی از موارد (سوسیس، قارچ، پیرونی) را داشته باشد، اما امروز یه تخفیفی فروشگاه گذاشته که بیلی میتونه پیزای سوسیس و قارچ رو باهم در قیمت ۵ دلار داشته باشه، تابع و نمودار درستی این مسئله را بنویسید.

S	M	P	Results
.	.	.	۱
.	.	۱	۱
.	۱	.	۱
.	۱	۱	.
۱	.	.	۱
۱	.	۱	.
۱	۱	.	۱
۱	۱	۱	.

از آنجایی که معلوم است:

S	M	P	Results	S	M	P	Boolean Form	Results
.	.	.	۱	.	.	.	$\bar{S}\bar{M}\bar{P}$	۱
.	.	۱	۱	.	.	۱	$\bar{S}\bar{M}P$	۱
.	۱	.	۱	.	۱	.	$\bar{S}M\bar{P}$	۱
۱	.	.	۱	۱	.	.	$S\bar{M}\bar{P}$	۱
۱	۱	.	۱	۱	۱	.	$SM\bar{P}$	۱

$$W = (\bar{S}\bar{M}\bar{P}) + (\bar{S}\bar{M}P) + (\bar{S}M\bar{P}) + (S\bar{M}\bar{P}) + (SM\bar{P}) \quad (۱۳)$$

تابع ساده شده

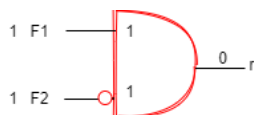
$$W = P + M + S + (SM) \quad (۱۴)$$

۱۱.۲ Intermediate Functions

گاهی اوقات برای بدست آوردن یک نتیجه میتوانیم از یکسری تابع تحت عنوان توابع واسطه^۲ استفاده کنیم، برای مثال، در مسئله زیر دو تابع واسطه بین تابع نهایی G آمده است که ما میتوانیم به کمک آن به نتیجه G برسیم:

X	Y	Z	F _۱	F _۲	G
۰	۰	۰	۰	۰	۰
۰	۰	۱	۱	۰	۱
۰	۱	۰	۱	۰	۱
۰	۱	۱	۱	۰	۱
۱	۰	۰	۱	۰	۱
۱	۰	۱	۱	۰	۱
۱	۱	۰	۱	۰	۱
۱	۱	۱	۱	۱	۰

بطور کلی تابع اول، OR بین تمام عملوند ها میباشد، و تابع دوم AND بین آنها، در تابع G در ابتدا شباهت ۰ را در دو تابع داریم اما زمانی که دو تابع به عدد ۱ رسیده اند، نتیجه G برابر با ۰ شده است، برای بدست آوردن تابع G میتوانیم به این صورت عمل کنیم:



۱۲.۲ تمارین مین ترم ها

با توجه به نتایج تابع G با استفاده از توابع Min Term تابع جدول را بدست آورید.^۳

X	Y	Z	G
۰	۰	۰	۱
۰	۰	۱	۱
۰	۱	۰	۰
۰	۱	۱	۰
۱	۰	۰	۱
۱	۰	۱	۰
۱	۱	۰	۱
۱	۱	۱	۰

^۳ در حقیقت Min Term هایی که در نظر داریم بایستی **Boolean Form** آن ورودی ها را اول بنویسیم و در قسمت هایی که تابع داده شده حداقل یک بار یک بوده است Mid Term ها بدست خواهد آمد.

جواب:

X	Y	Z	Min Terms ^f	G
•	•	•	$\bar{X}\bar{Y}\bar{Z}$	۱
•	•	۱	$\bar{X}\bar{Y}Z$	۱
•	۱	•	$\bar{X}Y\bar{Z}$	•
•	۱	۱	$\bar{X}YZ$	•
۱	•	•	$X\bar{Y}\bar{Z}$	۱
۱	•	۱	$X\bar{Y}Z$	•
۱	۱	•	$XY\bar{Z}$	۱
۱	۱	۱	XYZ	•

X	Y	Z	Min Terms	G
•	•	•	$\bar{X}\bar{Y}\bar{Z}$	۱
•	•	۱	$\bar{X}\bar{Y}Z$	۱
۱	•	•	$\bar{X}Y\bar{Z}$	۱
۱	۱	•	$XY\bar{Z}$	۱

$$W = (\bar{X}\bar{Y}\bar{Z}) + (\bar{X}\bar{Y}Z) + (X\bar{Y}\bar{Z}) + (XY\bar{Z}) \quad (۱۵)$$

مثال دوم:

X	Y	Z	G
۰	۰	۰	۰
۰	۰	۱	۰
۰	۱	۰	۱
۰	۱	۱	۱
۱	۰	۰	۰
۱	۰	۱	۰
۱	۱	۰	۱
۱	۱	۱	۰

جواب:

X	Y	Z	Min Terms	G
۰	۰	۰	$\bar{X}\bar{Y}\bar{Z}$	۰
۰	۰	۱	$\bar{X}\bar{Y}Z$	۰
۰	۱	۰	$\bar{X}Y\bar{Z}$	۱
۰	۱	۱	$\bar{X}YZ$	۱
۱	۰	۰	$X\bar{Y}\bar{Z}$	۰
۱	۰	۱	$X\bar{Y}Z$	۰
۱	۱	۰	$XY\bar{Z}$	۱
۱	۱	۱	XYZ	۰

جا هایی که تابع G ، یک شده است: ^۵

X	Y	Z	Min Terms	G
۰	۱	۰	$\bar{X}Y\bar{Z}$	۱
۰	۱	۱	$\bar{X}YZ$	۱
۱	۱	۰	$XY\bar{Z}$	۱

$$W = (\bar{X}Y\bar{Z}) + (\bar{X}YZ) + (XY\bar{Z}) \quad (۱۶)$$

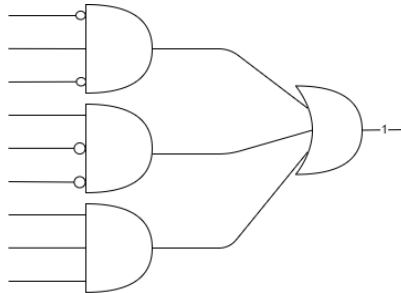
^۵ جا هایی که تابع داده شده یک شده است در حقیقت مشخص کننده Mid Term هامون هستند که Form Boolean آنها Product Term بین آنهاست

مثال سوم:

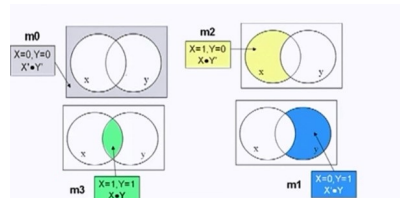
X	Y	Z	G		X	Y	Z	Min Terms	G
•	•	•	•		•	•	•		
•	•	۱	•		•	۱	•	$\bar{X}Y\bar{Z}$	۱
•	۱	•	۱		۱	•	•	$X\bar{Y}\bar{Z}$	۱
•	۱	۱	•		۱	۱	•	XYZ	۱
۱	•	•	۱						
۱	•	۱	•						
۱	۱	•	•						
۱	۱	۱	۱						

$$G = (\bar{X}Y\bar{Z}) + (X\bar{Y}\bar{Z}) + (XYZ) \quad (۱۷)$$

حالا با استفاده از گیت رسم شده بالا میتوانیم هر ورودی جدول درستی را در این گیت بگذاریم و به نتیجه تابع داده شده برسیم.



Min term diagram ۱۳.۲



برای جدول زیر نمودار ون و تابع را بنویسید

X	Y	F
۰	۰	۰
۰	۱	۱
۱	۰	۱
۱	۱	۰

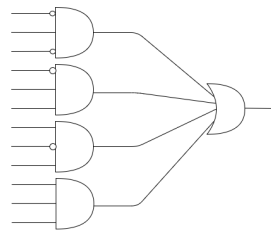
$$F = (\bar{X}Y) + (X\bar{Y}) \quad (۱۸)$$



مثال پنجم:

X	Y	Z	F		X	Y	Z	Min Terms	F
•	•	•	•		•	•	•	$\bar{X}\bar{Y}\bar{Z}$	•
•	•	۱	•		•	•	۱	$\bar{X}YZ$	•
•	۱	•	۱		•	۱	•	$X\bar{Y}\bar{Z}$	•
•	۱	۱	۱		•	۱	۱	$X\bar{Y}Z$	•
۱	•	•	•		۱	•	•	XYZ	•
۱	•	۱	۱		۱	•	۱		•
۱	۱	•	•						
۱	۱	۱	۱						

$$F = (\bar{X}\bar{Y}\bar{Z}) + (\bar{X}YZ) + (X\bar{Y}\bar{Z}) + (XYZ) \quad (۱۹)$$

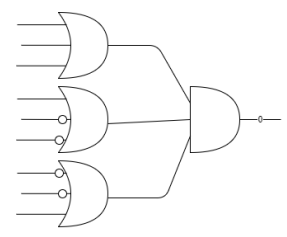


۱۴.۲ تمارین ماکس ترم ها

در ماکسترم ها همانطور که در صفحات قبلی توضیح داده شد، اگر مسئله به ما تابعی داده بود، توسط آن تابع به دنبال خروجی هایی میگردیم که برابر با صفر شده باشد، به طور کلی ماکس ترم به سام ترم هایی گفته می شود که حداقل در یک حالت برابر با صفر میباشند.

X	Y	Z	F		X	Y	Z	Max Term BF
.	$X + Y + Z$
.	.	1	1		.	1	1	$X + \bar{Y} + \bar{Z}$
.	1	.	1		1	1	.	$\bar{X} + \bar{Y} + Z$
.	1	1	.					
1	.	.	1					
1	.	1	1					
1	1	.	.					
1	1	1	1					

$$F = (X + Y + Z)(X + \bar{Y} + \bar{Z})(\bar{X} + \bar{Y} + Z) \quad (۲۰)$$



در تعریف ماکسترم باید توجه کنیم که زمانی یک ماکسترم، ماکسترم است که، بین متغیر ها جمع یا OR باشد، فرقی ندارد که به صورت not یا حالت عادی باشد، اما بایستی بررسی ها به تعداد ورودی ها صورت گیرد نه اینکه نقصی در آنها باشد.

$$F(XYZ) = >: \quad (21)$$

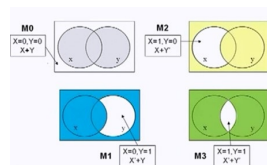
۱. $F(X + Y + Z)$

۲. $F(X + \bar{Y} + \bar{Z})$

۳. Is not max term $F(X + \bar{Y})$

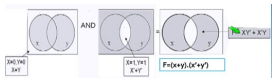
۴. $F(\bar{X}Y\bar{Z})$

Max term diagram ۱۵.۲



X	Y	F	X	Y	Max term BF	F
0	0	0	0	0	$X + Y$	0
0	1	1	1	1	$\bar{X} + \bar{Y}$	0
1	0	1				
1	1	0				

$$F = (X + Y)(\bar{X} + \bar{Y})$$
(۲۲)

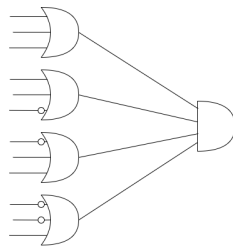


مثال سوم:

X	Y	Z	F
•	•	•	•
•	•	۱	•
•	۱	•	۱
•	۱	۱	۱
۱	•	•	•
۱	•	۱	۱
۱	۱	•	•
۱	۱	۱	۱

X	Y	Z	Max Terms BF	F
•	•	•	$X + Y + Z$	•
•	•	۱	$X + Y + \bar{Z}$	•
۱	•	•	$\bar{X} + Y + Z$	•
۱	۱	•	$\bar{X} + \bar{Y} + Z$	•

$$F = (X + Y + Z).(X + Y + \bar{Z}).(\bar{X} + Y + Z).(\bar{X} + \bar{Y} + Z) \quad (۲۳)$$



یک فرم استاندارد برای نمایش این Max term ها و min term ها وجود دارد، که چون در مین ترم ها به صورت جمع ضرب ها عمل میکنیم میتوانیم از علامت σ استفاده کنیم و در ماکسترم ها چون از ضرب جمع ها استفاده خواهیم کرد با علامت Π پیش میرویم، یعنی اگر بخواهیم مین ترم ها و مکسترم های مثال بالا را به صورت استاندارد نشان دهیم به صورت زیر خواهیم نوشت.

$$\sum_{XYZ}(2, 3, 5, 7) \mid \prod_{XYZ}(0, 1, 4, 6)$$

۱۶.۲ گسترش و ترکیب مسئله

در برخی از مسائل به ازای ورودی ها، یک تابعی قرار میدهند، که باید براساس آن عبارات را باهم ترکیب کنیم.

$$F = \bar{W}X + W\bar{X}Y \quad (۲۴)$$

این عبارت دو قسمت اصلی دارد، یک قسمت اول سمت چپ جمع، یک قسمت سمت راست جمع. برای هر کدام از قسمت ها بررسی میکنیم که اگر متغیری نات شده بود یعنی false یا ۰ است، و آن متغیری که حالت عادی دارد ۱ است.

$$F = \bar{W}X \quad (۲۵)$$

$$F = W\bar{X}Y \quad (۲۶)$$

در قسمت اول هر جایی که W برابر ۰ و X برابر ۱ باشد یعنی برابر با ۰۱ باشد را در نظر میگیریم، (مطابق جدول) و بولین فرم آن را خواهیم نوشت

W	X	Y	Z	F
۰	۰	۰	۰	۰
۰	۰	۰	۱	۰
۰	۰	۱	۰	۰
۰	۰	۱	۱	۰
۰	۱	۰	۰	۱
۰	۱	۰	۱	۱
۰	۱	۱	۰	۱
۰	۱	۱	۱	۱
۱	۰	۰	۰	۰
۱	۰	۰	۱	۰
۱	۰	۱	۰	۱
۱	۰	۱	۱	۱
۱	۱	۰	۰	۰
۱	۱	۰	۰	۰
۱	۱	۱	۰	۰
۱	۱	۱	۱	۰

بعد از این که حالات را در جدول مشخص کردیم حالا نوبت به نوشتن فرم بولین آن است:

قسمت اول:

$$F_1 = (\bar{W}X\bar{Y}\bar{Z}) + (\bar{W}X\bar{Y}Z) + (\bar{W}XY\bar{Z}) + (\bar{W}XYZ) \quad (۲۷)$$

قسمت دوم:

$$F_2 = (W\bar{X}Y\bar{Z}) + (W\bar{X}YZ) \quad (۲۸)$$

در نهایت:

$$F = (\bar{W}X\bar{Y}\bar{Z}) + (\bar{W}X\bar{Y}Z) + (\bar{W}XY\bar{Z}) + (\bar{W}XYZ) + (W\bar{X}Y\bar{Z}) + (W\bar{X}YZ) \quad (۲۹)$$

مسئله:

اگر میخواهی در درس هایت موفق باشی، باید سر موقع در کلاس هایت حاضر باشی و تکالیف را کامل و درست انجام بدی:

$$G = C.H \quad (30)$$

حالا به نظر شما دوست داشتن دانشجو نسبت به پیزا میتواند بستگی داشته باشد یا نه؟ پاسخ، خیر است چون واقعا ربطی به موفقیت دانشجو از نظر کلی ندارد و بررسی آن در مسئله بود و نبودش فرقی نمیکند! اما با این حال خواهیم داشت:

$$G = C.H.P + C.H.\bar{P} \quad (31)$$

۱۷.۲ بررسی گسترش مسئله در جمع ها و ضرب ها

اگر براساس ورودی های مسئله ما چند ورودی کم داشته باشیم، به ازای ورودی های گم شده یا نانوشته ۲ به توان آن تعداد حالت داریم که آن ورودی هایی که حضور دارند در آن حالات ثابت اند و باقی حالات نشان دهنده حالت های ورودی های نانوشته است. برای مثال:

$$-\bar{X}.Z \quad (32)$$

$$(\bar{X}.Z) * (4)form \quad (33)$$

SoP

$$(W.\bar{X}.Y.Z) + (\bar{W}.\bar{X}.Y.Z) + (W.\bar{X}.\bar{Y}.Z) + (\bar{W}.\bar{X}.\bar{Y}.Z) \quad (34)$$

PoS

$$\bar{X} + Z \quad (35)$$

$$(W + \bar{X} + Y + Z)(\bar{W}.\bar{X} + Y + Z)(W + \bar{X} + \bar{Y} + Z)(\bar{W} + \bar{X} + \bar{Y} + Z) \quad (36)$$

$$F = \bar{Z} + \bar{W}XY \tag{۳۷}$$

W	X	Y	Z
.	.	.	.
.	.	.	\
.	.	\	.
.	.	\	\
.	\	.	.
.	\	.	\
.	\	\	.
.	\	\	\
\	.	.	.
\	.	.	\
\	.	\	.
\	.	\	\
\	\	.	.
\	\	.	.
\	\	\	.
\	\	\	\

$$F = \bar{Z} \rightarrow \Sigma(0, 2, 4, 6, 8, 9, 11, 12, 13) \tag{۳۸}$$

$$F = \bar{W}XY \rightarrow \Sigma(6, 7) \tag{۳۹}$$

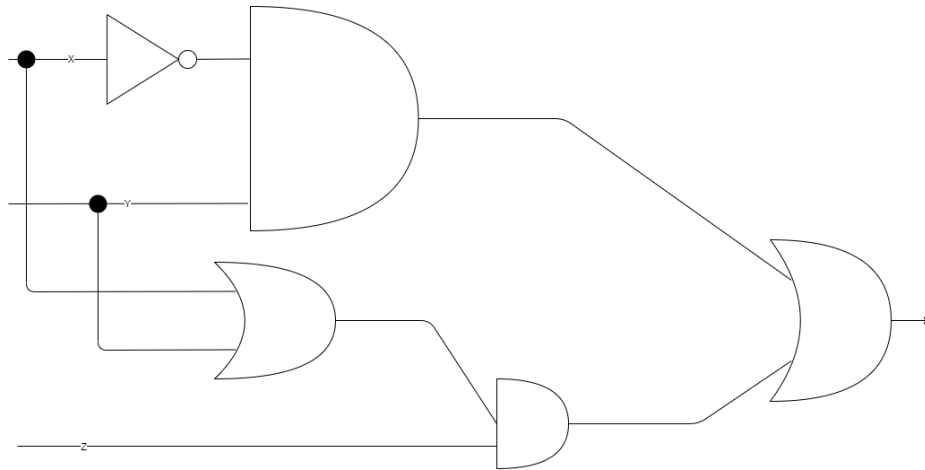
$$F = \Sigma(0, 2, 4, 6, 8, 9, 11, 12, 13) + \Sigma(6, 7) \tag{۴۰}$$

۱۸.۲ Converting to Canonical Sum تبدیل به جمع استاندارد

۱۹.۲ بررسی و آنالیز مدار

۱.۱۹.۲ مسئله ۱

مدار زیر را بررسی کنید.

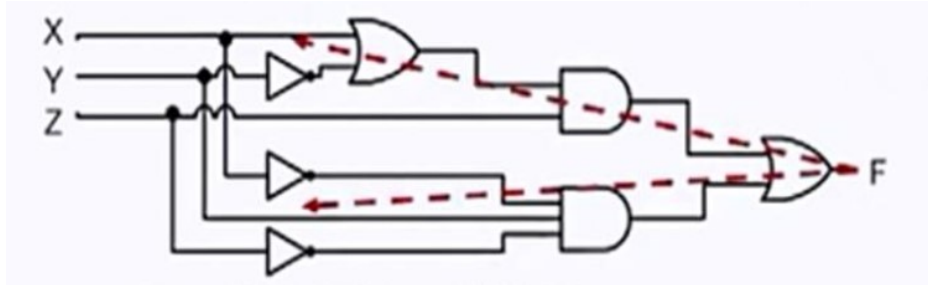


$$F = (\bar{X}Y) + Z(XY) \quad (41)$$

$$SoP = (\bar{X}Y) + (ZX) + (ZY) \quad (42)$$

$$PoS = (\bar{X} + Z)(Y + Z)(X + Y) \quad (43)$$

۲.۱۹.۲ مسئله ۲



$$F = (X + \bar{Y})Z + (\bar{X}Y\bar{Z}) \quad (۴۴)$$

$$SoP = XZ + \bar{Y}Z + (\bar{X}Y\bar{Z}) \quad (۴۵)$$

$$PoS = (Z + X)(Z + \bar{Y})(\bar{X} + Y + \bar{Z}) \quad (۴۶)$$

۲۰.۲ سناریو اول

تام با سه دوستش، آلیس، باب و چارلی به یک پارتی دعوت شده اند، اما تام زمانی میتواند به پارتی برود که فقط یکی از آنها با تام به پارتی بیایند. کدام یک از فرمول های زیر بیانگر این مسئلهست؟

- $T = A + B + C$
- $T = A.B.C$
- $T = A\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}\bar{B}C$
- $T = \bar{B}\bar{C} + \bar{A}\bar{C} + \bar{A}\bar{B}$

پاسخ درست گزینه سوم است چرا که مسئله فقط یکی از دوستان تام را گفته که اجبار به آمدن دارند.

۲۱.۲ سناریو دوم

جیل هم مانند تام به مهمانی دعوت شده که حداقل سه تا آن دوستان باید به مهمانی بیایند، کدام حالت زیر صحیح است؟

- $T = \bar{A} + \bar{B} + \bar{C}$
- $T = \bar{A}.\bar{B}.\bar{C}$
- $T = A\bar{B}C + \bar{A}BC + ABC$
- $T = BC + AC + AB$

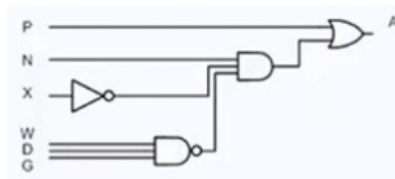
حالت ۴ صحیح است چرا که فقط دو نفر موجودند که با جیل به پارتی بیایند.

۲۲.۲ سطوح مدار

مدار هشدار را با ورودی وحشت اضافی در نظر بگیرید:
آلارم زمانی به صدا در میاید که دکمه وحشت روشن باشد یا اگر سیستم روشن باشد و شما هیجان زده نبودید و خانه امن نبود. در نظر داشته باشید که زمانی خانه امن نیست که درها و پنجره ها و پارکینگ قفل نباشند.

$$A = P + N\bar{X}(WDG) \quad (۴۷)$$

$$A_{\text{gettingDemorgan}} = P + N\bar{X}(\bar{W} + \bar{D} + \bar{G}) \quad (۴۸)$$

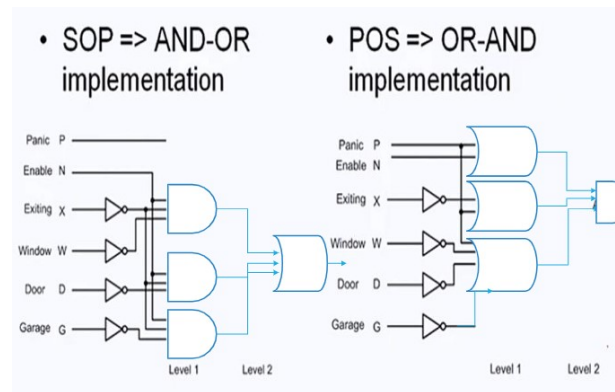


این مدار دارای ۳ سطح می باشد اما در نظر داشته باشیم که هر چقدر تعداد سطوح (براساس تعداد گیت ها) بیشتر باشد سرعت انجام فرایند ها در آن نیز کم میشود. مدار بالا دارای سه سطح است که اگر بتوانیم آن را به دو سطح برسانیم سرعت آن را نیز افزایش داده ایم. پس مدار بالا را به PoS و SoP تبدیل می کنیم:

$$SoP = P + (N\bar{X}\bar{W}) + (N\bar{X}\bar{D}) + (N\bar{X}\bar{G}) \quad (49)$$

$$PoS = (P + N)(P + \bar{X})(P + \bar{W} + \bar{D} + \bar{G}) \quad (50)$$

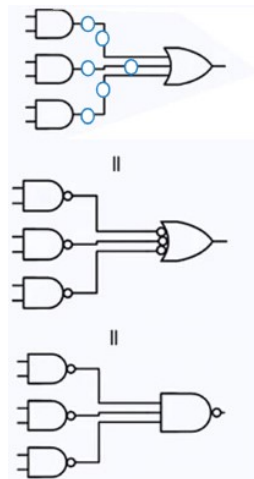
کدام یک از مدارات زیر سریع تر است؟



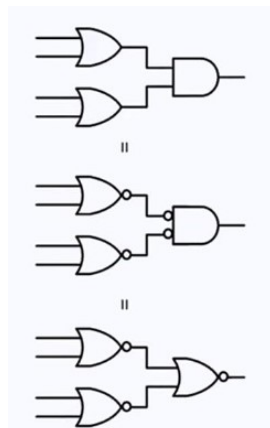
مدار سمت راستی زیرا از پیچیدگی کم تر و ورودی های کمتر و بهینه تر بهره میبرد.

۲۳.۲ بهینه کردن مدارها با تغییر جزئی در آنها

AND-OR/NAND-NAND in Sum of Product ۱.۲۳.۲



OR-AND/NOR-NOR in Product of Sum ۲.۲۳.۲



Karnough Map ۲۴.۲

کارنو یک ابزار مناسب برای ساده سازی مدار مورد استفاده قرار میگیرد. خروجی کارنو به صورت SoP و PoS میباشد. یکی از مهمترین مزیت های جدول کارنو این است که مدار را در دوسطح پیاده سازی میکند که این باعث میشود تا با ورودی ها و سخت افزار های کم به پرفورمنس بهینه دست یابیم.

۱.۲۴.۲ سنتز مدار

در این جا تا حدودی با تعریف سنتز مدار آشنا میشویم فقط که یعنی تبدیل فرم منطقی مدار به شکلی که بشود آنرا پیاده سازی کرد

جدول کارنو شکل خاصی ندارند و میتوان آنرا به شکل های مختلفی کشید (چه بصورت عمودی چه به صورت افقی)
تعداد اعضای خانه های کارنو به تعداد ورودی ها بستگی دارد که حتما توانی از دو است، یعنی اگر ۳ ورودی داشته باشیم ۸ عضو در خانه ها کارنو مپ خواهیم گذاشت. اعضای کارنو به صورت صفر ها و یک ها میباشد که مشخص کننده MinTerm و MaxTerm هستند.
شماره گذاری کارنو مپ از سیستم کد گری تبعیت میکند چرا که در کد گری قصد ما فاصله یک بیت بین هر عدد بود و در اینجا هم این فاصله یک بیت حائز اهمیت است چرا که به ما کمک می کند تا ساده سازی بهتری داشته باشیم و راحت تر بتوانیم عمل Adjacent را انجام دهیم.
برای مثال:

$$\Sigma_{XYZ}(1, 4, 5, 6) \quad (51)$$

XY \ Z	00	01	11	10
0	0 0	2 0	6 1	4 1
1	1 1	3 0	7 0	5 1

یا میتوان به فرم دیگر نمایش داد:

$$\Sigma_{XYZ}(1, 4, 5, 6) \quad (52)$$

X		0	1
YZ	00	0 ₀	1 ₁
	01	2 ₀	3 ₀
	11	6 ₁	7 ₀
	10	4 ₁	5 ₁

همچنین نمایش موارد زیر در کارنومپ

$$\Sigma_{WXYZ}(1, 2, 3, 5, 6, 7, 9, 10, 11, 14, 15)$$

(۵۳)

WX		00	01	11	10
YZ	00	0 ₁	4 ₀	12 ₀	8 ₀
	10	1 ₁	5 ₁	13 ₀	9 ₁
	11	2 ₁	6 ₁	14 ₁	10 ₁
	10	3 ₁	7 ₁	15 ₁	11 ₁

۲۵.۲ نحوه ساده سازی به کمک کارنو مپ

برای ساده سازی چند نکته حائز اهمیت هستند:

- فقط خانه های مجاور هم میتوانند باهم ساده شوند.
- تعداد اعضایی که برای ساده سازی انتخاب میکنیم باید توانی از ۲ باشد.

برای مثال:

$$\Sigma_{WXYZ}(0, 2, 4, 5, 6) \quad (۵۴)$$

XY \ Z	00	01	11	10
0	0 1	2 1	6 1	4 1
1	1 0	3 0	7 0	5 1

ساده سازی:

XY \ Z	00	01	11	10
0	0 1	2 1	6 1	4 1
1	1 0	3 0	7 0	5 1

$$(\bar{X}\bar{Y}\bar{Z}) + (\bar{X}Y\bar{Z}) + (XY\bar{Z}) + (X\bar{Y}\bar{Z}) \quad (۵۵)$$

$$(X\bar{Y}\bar{Z}) + (X\bar{Y}Z) \quad (۵۶)$$

ساده سازی اول:

$$(\bar{Z}(\bar{X}\bar{Y} + \bar{X}Y + XY + X\bar{Y})) \quad (۵۷)$$

$$(\bar{Z}(\bar{X}(\bar{Y} + Y) + X(Y + \bar{Y}))) \quad (58)$$

$$(\bar{Z}(\bar{X} + X)) \quad (59)$$

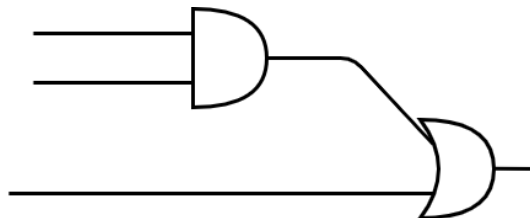
$$(\bar{Z}) \quad (60)$$

ساده سازی دوم:

$$(X\bar{Y}\bar{Z}) + (X\bar{Y}Z) \quad (61)$$

$$(X\bar{Y}(\bar{Z} + Z)) \quad (62)$$

$$\bar{Z} + (X\bar{Y}) \quad (63)$$



۲۶.۲ گروه بندی اعضا در جدول کارنو

برخی از نکات گروه بندی در کارنو مپ:

- برای گروه بندی حتما اعضا باید مجاور باشند
- برای گروه بندی باید تعداد اعضا توانی از دو باشند
- در گروه بندی تعداد دسته بندی ها و گروه ها مشکلی ندارد که کم باشد اما میتواند اعضای بیشتر را داشته باشد

- هم پوشانی بین اعضا در جدول کارنو مهم نیست و میتواند به عنوان گروهی دیگر در نظر گرفته شود.
- اگر کل جدول یک باشد تمام اعضا در نهایت یک یا true خواهد بود.
- هنگام گروه بندی کردن امکان گروه بندی به صورت مورب وجود ندارد
- هنگام گروه بندی اعضا با یکدیگر اگر اعضا در پایین ترین قسمت کارنو بودند میتوانند با قسمت بالایی کارنو گروه بندی شوند.

نکته: برای اینکه بدانیم در هر دسته چند متغیر ساده خواهند شد از فرمول زیر استفاده

میکنیم:

$$\log_2^{varqty} \quad (۶۴)$$

برای مثلا اگر در دسته ۴ تایی بودیم باتوجه به فرمول فقط دو متغیر ساده میشد، یا مثلا اگر در یک دسته ۱۶ تایی قرار بگیریم در صورتی که این دسته کوچکتر از مقدار کل باشد ۴ متغیر ساده خواهد شد.

نکته، اگر در دسته ای مانند شکل زیر قرار گرفتید بدانید نمیتوان به صورت ضربدری یا مورب ساده سازی را انجام داد، شرط مجاورت با یکدیگر به جا است.

z \ xy	xy			
	00	01	11	10
0	0	1	0	0
1	1	0	0	0

به گروه بندی کارنوی زیر دقت کنید، همان طور که معلوم است در دو قسمت یک بسته ۴ تایی وجود دارد که در با یکدیگر این دویسته هم پوشانی دارند، و همچنین در خانه ۱۱ با خانه ۸ میتواند در یک گروه قرار گیرد.

WX \ YZ	00	01	11	10
00	0 0	4 0	12 1	8 1
01	1 1	5 1	13 1	9 0
11	2 1	6 1	14 1	10 0
10	3 0	7 0	15 0	11 1

با توجه به دو گروه چهار تایی و با توجه به فرمول گفته شده میتوان دریافت که این دو گروه فقط ۲ متغیر را ساده میکنند. حل این قسمت:

$$(\bar{W}\bar{X}\bar{Y}Z) + (\bar{W}\bar{X}YZ) + (\bar{W}X\bar{Y}Z) + (\bar{W}XYZ) \quad (65)$$

$$((\bar{W}\bar{X}(\bar{Y}Z + YZ)) \rightarrow (\bar{W}\bar{X})Z(\bar{Y} + Y)) = (\bar{W}\bar{X}Z) \quad (66)$$

$$(XZ(\bar{W}\bar{Y} + \bar{W}Y)) \rightarrow (XZ(\bar{W}(\bar{Y} + Y))) = (XZ\bar{W}) \quad (67)$$

$$(\bar{W}\bar{X}Z) + (\bar{W}XZ) \rightarrow (Z(\bar{W}\bar{X} + X\bar{W})) = (Z(\bar{W}(Z + \bar{Z}))) = \bar{W}Z \quad (68)$$

۲۷.۲ بدست آوردن PoS با SoP

وقتی با کارنو می‌رو به رو می‌شویم که درون آن صفرها و یک‌هایی وجود دارد، ما معمولاً برای بدست آوردن Sum of Product از یک‌ها استفاده می‌کنیم که در واقع MinTerm مورد نظر بدست بیاید. در ادامه همین جریان می‌توان مین ترم را روی صفرها هم در نظر بگیریم یعنی اینکه صفرها را از روش مین ترم بدست بیاوریم یا به طور کلی از روش Sum of Product مانند کانونی زیر:

$$F = \Sigma(2, 3, 5, 7) \quad (۶۹)$$

bc		00	01	11	10
a	0	0 0	1 0	3 1	2 1
	1	4 0	5 1	7 1	6 0

گروه بندی کردن صفرها و یک‌ها:

bc		00	01	11	10
a	0	0 0	1 0	3 1	2 1
	1	4 0	5 1	7 1	6 0

بدست آوردن مین ترم‌های یک:

$$F_{1's} = (\bar{a}bc) + (\bar{a}b\bar{c}) = (\bar{a}b(\bar{c} + c)) = \bar{a}b \quad (۷۰)$$

$$F_{1's} = (\bar{a}\bar{b}c) + (abc) = ac(\bar{b} + b) = ac \quad (۷۱)$$

$$F_{1's final} = \bar{a}b + ac \quad (۷۲)$$

بدست آوردن مین ترم‌های صفر:

$$F_{0's} = (\bar{a}\bar{b}\bar{c}) + (\bar{a}b\bar{c}) = \bar{a}\bar{b}(\bar{c} + c) = \bar{a}\bar{b} \quad (۷۳)$$

$$F'_{0's} = (\bar{a}\bar{b}\bar{c}) + (ab\bar{c}) = a\bar{c}(\bar{b} + b) = a\bar{c} \quad (74)$$

$$F'_{0'sfinal} = \bar{a}\bar{b} + a\bar{c} \quad (75)$$

۱.۲۷.۲ نکته بسیار مهم

اگر بخواهیم هم PoS و SoP صفرها و یک را بدست بیاوریم؛ یا PoS و SoP از هر دو صفرها و یک ها میگیریم که در نتیجه آنها را ساده میکنیم و به نتیجه خواهیم رسید یا از میانبر زیر استفاده میکنیم.

بدست آوردن PoS از طریق SoP:

یک میانبر راحت در این بین وجود دارد و آن هم این است که برای بدست آوردن PoS یک ها از تابع مین ترم صفرها استفاده میکنیم و آن را به صورت نقیض شده در نظر خواهیم گرفت.

برای بدست آوردن PoS صفرها از تابع مین ترم یک ها استفاده میکنیم و به صورت نقیض شده آنها حل میکنیم.

$$F_{SoP} = \bar{a}b + ac \quad (76)$$

$$F'_{SoP} = \bar{a}\bar{b} + a\bar{c} \quad (77)$$

$$F_{PoS} = (a + b)(\bar{a} + c) \quad (78)$$

$$F'_{PoS} = (a + \bar{b})(\bar{a}\bar{c}) \quad (79)$$

مسئله:

حالات مختلف چهار ورودی را در نظر بگیرید و SoP و PoS آنها را بنویسید.

$$\prod(3, 7, 10, 11, 15) \quad (۸۰)$$

cd \ ab				
	00	01	11	10
00	0 ₁	1 ₁	3 ₀	2 ₁
01	4 ₁	5 ₁	7 ₀	6 ₁
11	12 ₁	13 ₁	15 ₀	14 ₁
10	8 ₁	9 ₁	11 ₀	10 ₀

cd \ ab				
	00	01	11	10
00	0 ₁	1 ₁	3 ₀	2 ₁
01	4 ₁	5 ₁	7 ₀	6 ₁
11	12 ₁	13 ₁	15 ₀	14 ₁
10	8 ₁	9 ₁	11 ₀	10 ₀

$$\bar{c} \rightarrow 1(n) = 8 \text{ with } 1 \text{ shared between} \quad (۸۱)$$

cd \ ab				
	00	01	11	10
00	0 ₁	1 ₁	3 ₀	2 ₁
01	4 ₁	5 ₁	7 ₀	6 ₁
11	12 ₁	13 ₁	15 ₀	14 ₁
10	8 ₁	9 ₁	11 ₀	10 ₀

$$1 : [(\bar{a}\bar{b}c\bar{d}) + (\bar{a}bcd\bar{d})] + [(\bar{a}\bar{b}c\bar{d}) + (\bar{a}bcd\bar{d})] \quad (\text{A2})$$

		cd			
ab		00	01	11	10
	00	0 1	1 1	3 0	2 1
	01	4 1	5 1	7 0	6 1
	11	12 1	13 1	15 0	14 1
	10	8 1	9 1	11 0	10 0

$$2 : [(\bar{a}\bar{b}c\bar{d}) + (ab\bar{c}\bar{d})] + [(\bar{a}bcd\bar{d}) + (abcd\bar{d})] \quad (\text{A3})$$

$$F_{(SoP)} = [\bar{c} + (\bar{a}\bar{d}) + (b\bar{d})] \quad (\text{A4})$$

		cd			
ab		00	01	11	10
	00	0 1	1 1	3 0	2 1
	01	4 1	5 1	7 0	6 1
	11	12 1	13 1	15 0	14 1
	10	8 1	9 1	11 0	10 0

$$F'_{(SoP)}$$

$$[(\bar{a}\bar{b}cd) + (\bar{a}bcd)] + [(abcd) + (a\bar{b}cd)] + [(a\bar{b}cd) + (a\bar{b}c\bar{d})] = (\bar{a}cd + acd) + (a\bar{b}c) \quad (۸۵)$$

که در نهایت :

$$F_{(SoP)} = \bar{c} + (\bar{a}\bar{d}) + (b\bar{d}) \quad (۸۶)$$

$$F'_{(SoP)} = cd + (a\bar{b}c) \quad (۸۷)$$

$$F_{(PoS)} = (\bar{c} + \bar{d})(\bar{a} + b + \bar{c}) \quad (۸۸)$$

$$F'_{(PoS)} = (c)(a + d)(\bar{b} + d) \quad (۸۹)$$

۲۸.۲ کارنو با ورودی های زیاد

استفاده راحت از کارنو مپ تا زمانی است که تعداد ورودی های ما در حد و اندازه ۳ تا ۴ متغیر باشد اما اگر ۵ متغیر داشته باشیم تعداد خانه های کارنو ما زیاد و فرم و شکل آن متفاوت خواهد بود. چرا که برای مثال در ۵ ورودی که ۳۲ خانه داریم بایستی از دو کانوی ۱۶ تایی استفاده کنیم، لازم به ذکر است که این دو کارنو با یکدیگر همپوشانی دارند، یعنی اینکه ممکن است در وسط کارنو اول دارای چهار مقدار ۱ باشیم و دقیقا در همان منطقه در کارنوی دوم آن وضعیت را داشته باشیم به همین خاطر میتوانند با یکدیگر مورد بررسی قرار گیرند، منتها در بررسی همچنین کارنو هایی باید بسیار دقت داشت که به مشکل بر نخوریم چرا که تعداد محاسبه و نوع شکل و قرارگیری کمی پیچیده و پر پیچ و خم است.

مثال:

$$\Sigma(1, 4, 9, 11, 13, 15, 17, 19, 22, 25, 27, 29, 30, 31) + d(3, 12, 20) \quad (90)$$

de \ bc	00	01	11	10
00	0 ₀	1 ₁	<u>3</u>	2 ₀
01	4 ₁	5 ₀	7 ₀	6 ₀
11	<u>12</u>	13 ₁	15 ₁	14 ₀
10	8 ₀	9 ₁	11 ₁	10 ₀

de \ bc	00	01	11	10
00	16 ₀	17 ₁	19 ₁	18 ₀
01	<u>20</u>	21 ₀	23 ₀	22 ₁
11	28 ₀	29 ₁	31 ₁	30 ₁
10	24 ₀	25 ₁	27 ₁	26 ₀

خطوط سیاهی که در کارنو ها میبینید به نام Don't Care هستند که ماهیت خاصی ندارند و تاثیرشان به تنهایی فایده ای ندارد، اما اگر در کنار گروهی باشند که وجود یک یا صفر در بین آنها مهم است میتوانند در شرایطی هم به صورت یک مورد استفاده قرار گیرند یا به صورت صفر!

de \ bc	00	01	11	10
00	0 ₀	1 ₁	<u>3</u>	2 ₀
01	4 ₁	5 ₀	7 ₀	6 ₀
11	<u>12</u>	13 ₁	15 ₁	14 ₀
10	8 ₀	9 ₁	11 ₁	10 ₀

de \ bc	00	01	11	10
00	16 ₀	17 ₁	19 ₁	18 ₀
01	<u>20</u>	21 ₀	23 ₀	22 ₁
11	28 ₀	29 ₁	31 ₁	30 ₁
10	24 ₀	25 ₁	27 ₁	26 ₀

$$1(n) = 4 : [(bc\bar{d}e) + (bcde)] + [(b\bar{c}d\bar{e}) + (b\bar{c}de)] = (dce) + (d\bar{c}e) = be \quad (91)$$

$$(92)$$

$$1(n) = 4 : topDown \rightarrow [(b\bar{c}d\bar{e}) + (b\bar{c}de)] + [(\bar{b}cd\bar{e}) + (\bar{b}cde)] = (b\bar{c}e) + (\bar{b}c\bar{e}) = \bar{c}e$$

$$(93)$$

$$1(n) = 4 : SideBar \rightarrow [(bcd\bar{e}) + (\bar{b}cd\bar{e})] + [(bcd\bar{e}) + (\bar{b}cd\bar{e})] = (cd\bar{e}) + (cd\bar{e}) = cd$$

$$F_{(SoP)} = (be) + (\bar{c}e) + (cd) \quad (94)$$

۳ مدارات ترکیبی

۱.۳ مدار سکویی مجوریتی یا Majority Circuit

مدار مجوریتی فقط دارای سه ورودی می باشد که خروجی آن براساس تعداد اکثریت می باشد، مثلا اگر در این سه ورودی دو ورودی صفر بود نتیجه این مدار صفر خواهد بود.

X	Y	Z	F
۰	۰	۰	۰
۰	۰	۱	۰
۰	۱	۰	۰
۰	۱	۱	۱
۱	۰	۰	۰
۱	۰	۱	۱
۱	۱	۰	۱
۱	۱	۱	۱

x \ yz	00	01	11	10
0	0	0	1	0
1	0	1	1	1

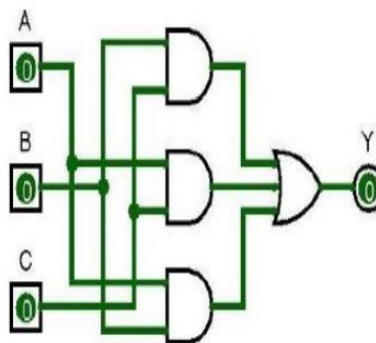
بدست آوردن تابع مجورتی سادست، اما از طریق جدول کارنو میتوان به این نتیجه رسید:

x \ yz	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$$F = [(x\bar{y}z) + (xyz)] + [(xyz) + (xy\bar{z})] + [(\bar{x}yz) + (xyz)] \quad (95)$$

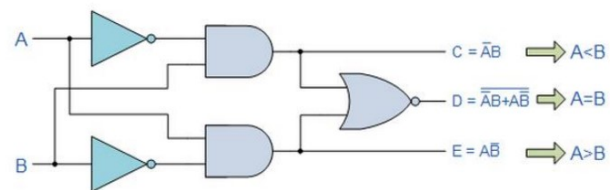
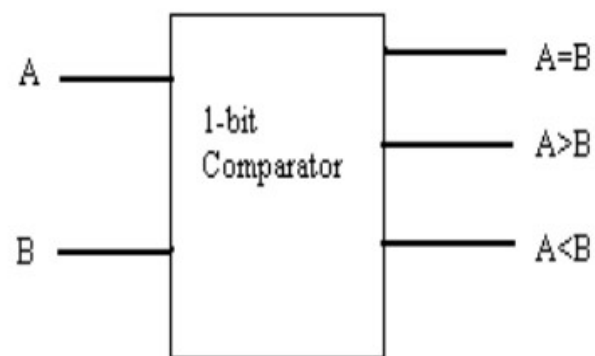
$$F = xy + xz + yz \quad (96)$$

به همین خاطر از طریق تابع بالا میتوان به مدار زیر رسید:

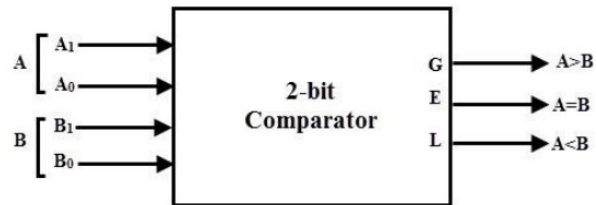


۲.۳ مقایسه کننده ها Comparator

۱.۲.۳ مقایسه کننده تک بیتی

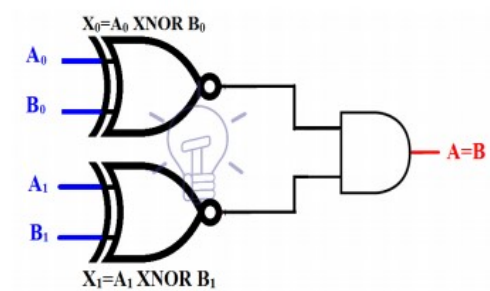


۲.۲.۳ مقایسه کننده دوبیتی

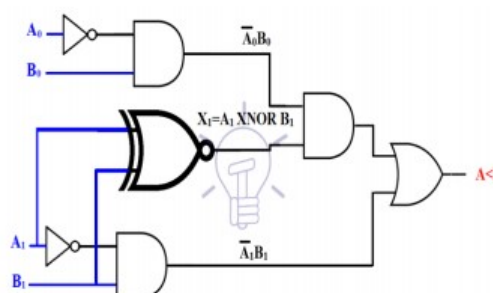


شرط برابری را معمولا با اکس نور بررسی خواهیم کرد.

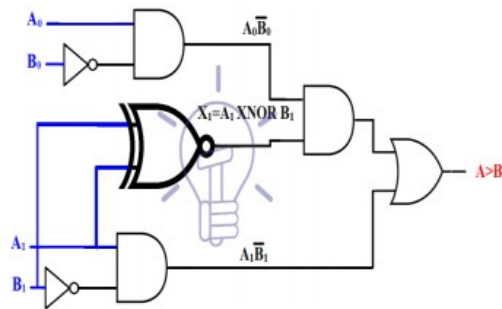
$$(ab).(\bar{a}\bar{b}) \quad (97)$$



$$A_0 < B_0 + A_1 < B_1 \rightarrow F = (\bar{A}_0.B_0) + (\bar{A}_1.B_1).(A_1.B_1) \quad (98)$$



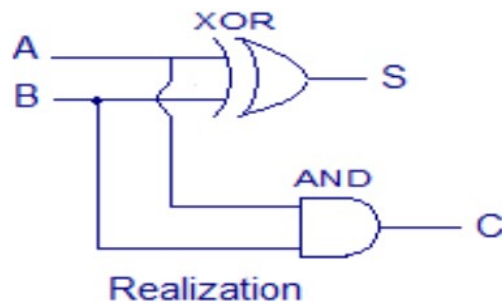
$$A_0 > B_0 + A_1 > B_1 \rightarrow (A_0.\bar{B}_0) + (A_1.\bar{B}_1).(A_1.B_1) \quad (99)$$



Half Adder ۳.۳

X	Y	Sum	Carry
۰	۰	۰	۰
۰	۱	۱	۰
۱	۰	۱	۰
۱	۱	۰	۱

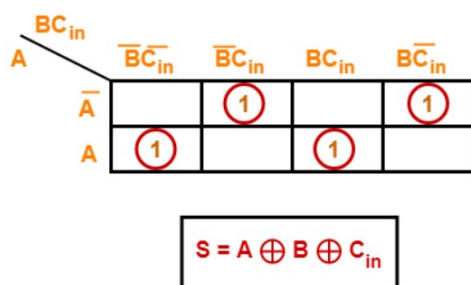
بدین معنا که در جمع دو ورودی از الگوی اکس اور استفاده شده و در کری از الگوی And بکار گرفته شده است.



۴.۳ Full Adder

X	Y	Cin	Sum	Cout
۰	۰	۰	۰	۰
۰	۰	۱	۱	۰
۰	۱	۰	۱	۰
۰	۱	۱	۰	۱
۱	۰	۰	۱	۰
۱	۰	۱	۰	۱
۱	۱	۰	۰	۱
۱	۱	۱	۱	۱

در جمع Full Adder از الگوی اکس اور و در کری آن از الگوی مجورتی استفاده شده است.



۵.۳ 4 Bit Ripple Carry Adder

جمع کننده ۴ بیتی کری ها به صورت موج مانند دیاگرام زیر است:

