محاسبات در مکمل۲

ویژگي مهم نمایش مکمل۲ سادگي پیاده سازي عملیات محاسباتي است.

جمع
تفریق
ضرب و تقسیم

جمع

یکی از دلایلی که روش مکمل۲ برای ذخیره اعداد صحیح علامتدار متداول است راحتی پیاده سازی عملیات جمع و تفریق در سخت افزار است. مزیت روش مکمل۲ این است که قوائد جمع و تفریق برای اعداد علامتدار دقیقا مشابه یکدیگر هستند

جمع دو عدد مکمل۲ به سادگی مانند جمع دو عدد باینری بدون علامت صورت می گیرد. وقتی عمل جمع در مکمل۲ انجام می شود سمت چپ ترین بیت ممکن است یک رقم نقلی تولید کند. این رقم نقلی استفاده نمی شود. بخاطر داشته باشید که کلیه داده ها در کامپیوتر دارای اندازه ثابتی هستند. جمع دو بایت همیشه یک بایت را می دهد(به همین ترتیب از جمع دو کلمه یک کلمه حاصل می شود). این خاصیت در نمایش مکمل دو مهم است .

مثال۱.

Decimal	Hex	
		111
+2567	0A07	0000 1010 0000 0111
+ 467	+ 01D3	+ 0000 0001 1101 0011
3034	0BDA	0000 1011 1101 1010

مثال۲.

Decimal	Hex	
		1 1111 11
+518	0206	0000 0010 0000 0110
+(-80)	+ FFB0	+ 1111 1111 1011 0000
438	c 01B6	c 0000 0001 1011 0110

مثال۳.

Decimal	Hex	
		1 1111 1111 11 11
-25	FFE7	1111 1111 1110 0111
+(-10)	+ FFF6	+ 1111 1111 1111 0110
-35	c FFDD	c 1111 1111 1101 1101

مثال٤.

Decimal	Hex	
		1 1111 11
+18495	483F	0100 1000 0011 1111
+ 25690	+ 645A	+ 0110 0100 0101 1010
44185	AC99	1010 1100 1001 1001

مثال٥.

Decimal	Hex	
		1 1 11 111
-5633	E9FF	1110 1001 1111 1111
+ (-29456)	+ 8CF0	+ 1000 1100 1111 0000
-35089	c 76EF	c 0111 0110 1110 1111

c نشان دهنده رقم نقلی است که در نتیجه ذخیره نمی شود و برای تشخیص آن از فلگ Carry استفاده می شود. در مثال های ۲ و ۳ مجموع از ۱۶ بیت بزرگتر است و از بیت اضافی سمت چپ صرفنظر می شود. آخرین ٤ بیت همیشه جمع درست را نمی دهد. . در مثال ٤ هیچ رقم نقلی در آخرین بیت وجود ندارد اما جواب صحیح نیست زیرا AC99 نمایش عدد ۲۱۳۵۱- است نه ٤٤۱۸۵. این مشکل به دلیل بیشتر شدن حاصل جمع از بزرگترین عدد قابل نمایش در ۱٦ بیت (+۳۲۷٦۷) است.

در مثال آخر نیز پاسخ صحیح نیست 76EF نمایش عدد مثبت ۳۰٤٤۷ است. این مشکل به دلیل کمتر شدن حاصل جمع از کوچکترین عدد قابل نمایش در ۱٦ بیت (۳۲۷٦۸-) است.

در مثال های ٤ و ٥ پدیده سرریزی رخ داده است. سخت افزار کامپیوتر قادر به تشخیص پدیده سرریز ی هنگام عمل جمع نیست. کامپیوتر به طور عادی عمل جمع را انجام می دهد و پردازش بیت ها از راست به چپ می پردازد. هنگام جمع بیت ها گاهی یک رقم نقلی به ستون سمت چپ تولید می شود. این رقم نقلی به جمع دو بیت ستون اضافه می شود. ستون خاص آخرین ستون سمت چپ یا بیت علامت است که ممکن است یک رقم نقلی به آن وارد (Carry In)یا یک رقم نقلی از آن خارج (Carry Out)شود .

جدول زیر به طور خلاصه بیان می کند که چه هنگام پدیده سرریزی رخ می دهد :

Overflow?	Carry Out of Sign bit	Carry In to Sign bit
No	No	No
Yes	Yes	No
Yes	No	Yes
No	Yes	Yes

تفریق

تفریق دو عدد به فرم مکمل۲ از طریق تفریق دو عدد به صورت اعداد بدون علامت انجام می گیرد. اگر عدد دوم از اولی بزرگتر باشد یک واحد به عدد اول اضافه می شود که به آن رقم قرضی می گویند. پدیده سرریزی برای عمل تفریق نیز وجود دارد و هنگامی رخ می دهد که حاصل تفریق از محدوده قابل نمایش خارج می شود. برای تشخیص آن کامپیوتر مسئله تفریق را توسط عمل جمع انجام می دهد یعنی مکمل۲ عدد دوم را گرفته سپس با عدد اول جمع می کند. اگر در حاصل جمع پدیده سرریزی رخ دهد یعنی در تفریق هم رخ داده است .

مثل.

```
195          1 00C3        00C3
- 618        - 026A   =>  + FD96
-----        -------      -------
-423         FE59         FE59
```

ضرب و تقسیم

عملیات ضرب و تقسیم برای اعداد بدون علامت و علامت دار متفاوت از هم هستند. فرض کنید عدد FF را در خودش ضرب کنید، در ضرب بدون علامت عدد ۲۵۵ در ۲۵۵ عدد ٦٥۰۲٥ را نتیجه می دهد یا FE01 ولی در ضرب علامت دار در واقع عدد ۱- در ۱- ضرب می شود و حاصل برابر با ۱ می شود.

CPU هیچ نظری ندارد که یک بایت یا کلمه خاص چی می خواهد نمایش بدهد. اسمبلی نظری درباره نوع هایی که زبان سطح بالا دارد ندارد. این که داده چگونه تفسیر می شود بستگی به دستورالعملی دارد که روی داده کار می کند. اینکه عدد FF برای نمایش عدد علامت دار ۱- بکار رفته یا ۲۵۵ بستگی به برنامه نویس دارد. زبان C نوع های صحیح علامت دار و بدون علامت را تعریف می کند و به کامپایلر اجازه می دهد تعیین کند دستورالعملی که با داده کار می کند درست است.