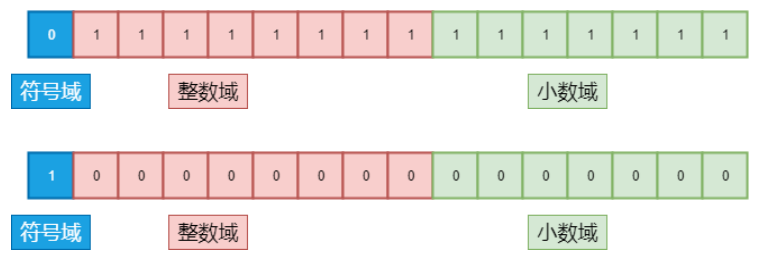
#### 数据标幺化及定点计算

**Q格式：**

Q1.31格式中，数值由1bit符号位和31bit分数位构成，数值范围-1（0x80000000）到（0X7FFFFFFF）;

Q1.15格式中，数值由1bit符号位和15bit分数位构成，数值范围-1（0x8000）到（0X7FFF）;

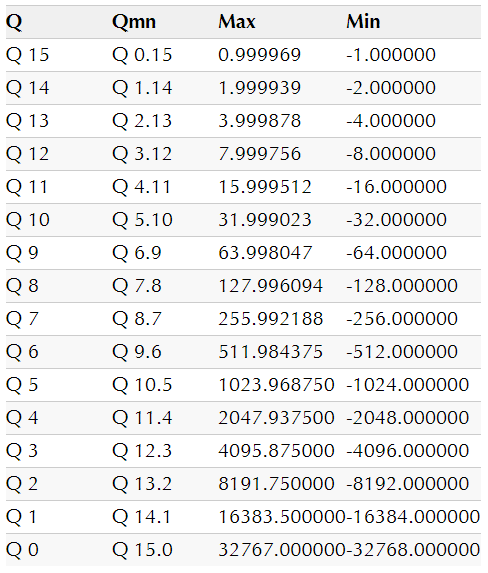


Q1.15转float：

判断最高位（16bit）是否为1，再将低15位/32768.

float转Q1.15：

判断float正负，小数位乘32768.



**数据标幺化：**

三角函数计算时，为加快计算速度，先对标幺化，再转为定点计算。

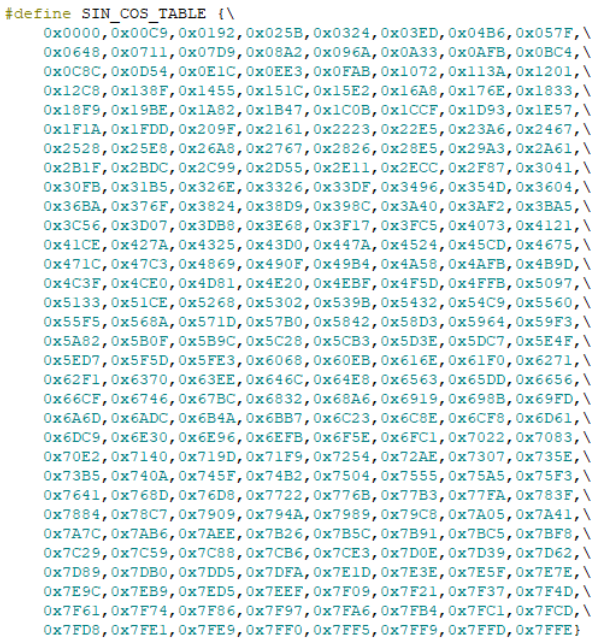
角度计算时，以为基值对[-, ]中的数标幺化，得到标幺值范围[-1,1]。例如，有角度，它的标幺值为。

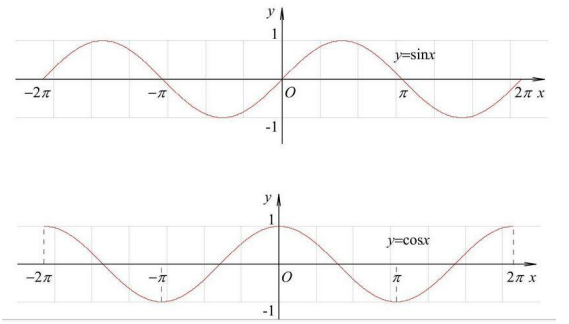
**定点计算：**

定点数有整数也有小数。一般来说，嵌入式处理器处理浮点数的速度是慢于整数计算的，因此可以将待计算数值标幺处理后，使用Q1.15格式（都是整数）的定点计算。

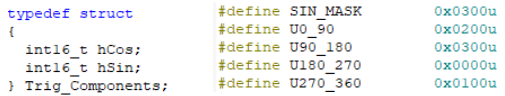


**以ST电机库三角函数定点算法为例(查表法)：**





SIN\_COS\_TABLE为q1.15格式数据。0x0000到0x7FFE，表示十进制0至1；SIN\_COS\_TABLE共有256个数据，其表示将长度分割成256个单元。





解释一下几个点：

（1）



[-Π，Π]对应[-32768,32767]，Q1.15格式的角度hAngle+32768，使得数据整体上移，此时[-Π，Π]对应[0,65535]。

（2）

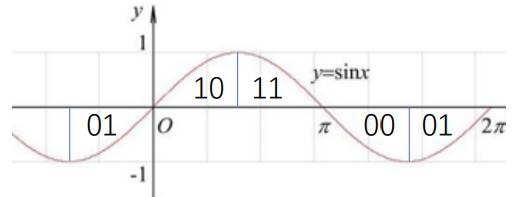


无符号16位数据取值范围[0,65535]，刚好和（1）中上移后得到的[-Π，Π]对应范围[0,65535]相吻合。

（3）



为什么要右移6位？先来看一张图。



由前面可知，定点角度在x轴[-Π,Π]定点为[-32768,32767]分了=65536份（65536数值可以理解成数据位置索引，下面同理），而SIN\_COS\_TABLE又将[0，Π/2]分了=256份，即[-Π，Π]共分了1024份（含义同上）。

经过（1）和（2）的操作后，定点角度在x轴[0,2Π]定点为[0,65535]分了=65536份（65536数值可以理解成数据位置索引，下面同理），而SIN\_COS\_TABLE又将[0，Π/2]分了=256份，即[0，2Π]共分了1024份（含义同上）。

从采样的角度来理解，TABLE表在=65536个数据中按一定间隔顺序采样了=1024个数，采样点数据精度最大为。

因此uhindex=uhindex>>6，将原来属于[0,65535]的uhindex按采样精度映射到[0,1024]，所以映射后的uhindex属于[0,]， 对应x轴[0,2Π]。

又因为[0,Π/2]对应[0,]，所以查看10bit的uhindex最高两位，就可以知道当前uhindex位于哪个编号区域。

实际计算例子：

1）



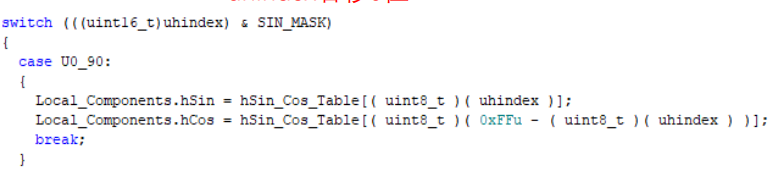
Angle属于10扇区，对应[0，Π/2]。

2）



上面存在溢出的情况，低十六位后，右移6位后，可知uhindex属于00扇区。

SIN\_MASK=0x0300u，是判断的扇区掩码，还是以举例，对应uhindex=1010101010。



，对应U0\_90，此时

sin=SIN\_COS\_TABLE[(uint8\_t)(uhindex)];cos和sin在[0,Π/2]是互补的，所以有

Cos=SIN\_COS\_TABLE[(uint8\_t)(0xFFu-(uint8\_t)(uhindex))];

其余角度计算方式同理。

**溢出检测：**

定点计算会出现溢出现象，根据实际计算情况必须要做溢出处理。

在park与反park变换计算中，park变换计算的时候已经做了溢出处理，所以在反park变换的计算时不用再做溢出处理。