

Listing 1—Bit-accurate-versus-traditional-C-code example

Traditional C Code

```
#define SCALE1 3      /* 3 extra LSB's used in previous calculations */
/*
 * Round off 3 extra LSB's of precision anyway C wants.
 */
void round1 (int *roundData)
{
    int cnt;
    int divisor = 1 << SCALE1;

    for (cnt=0; cnt<8; cnt++)
    {
        if (roundData[cnt] & 0x00200000) /* if negative */
            roundData[cnt] |= 0xFFC00000; /* extend the sign bit */
        roundData[cnt] = roundData[cnt] / divisor; /* let C scale and round */
        roundData[cnt] = roundData[cnt] & 0x003FFFFFF; /* remove sign extension */
    }
    return;
}
```

Bit-accurate C Code

```
/*
 * Round off 3 extra LSB's of precision just like the hardware.
 */
void round1 (int *roundData)
{
    int cnt;
    int one = 1 << SCALE1;
    int half = one >> 1;
    int mask = one - 1;

    for (cnt=0; cnt<8; cnt++)
    {
        if (roundData[cnt] & half)
        {
            if (roundData[cnt] & 0x00200000) /* if negative */
                if ((roundData[cnt] & mask) > half)
                    roundData[cnt] += one;
            else
                roundData[cnt] -= one;
        }
        roundData[cnt] >>= SCALE1;
    }
    return;
}
```