C NUMERIC FORMATS

Overview

The DSP supports the 32-bit single-precision floating-point data format defined in the IEEE Standard 754/854. In addition, the DSP supports an extended-precision version of the same format with eight additional bits in the mantissa (40 bits total). The DSP also supports 32-bit fixed-point formats—fractional and integer—which can be signed (twos-complement) or unsigned.

IEEE Single-Precision Floating-point Data Format

IEEE Standard 754/854 specifies a 32-bit single-precision floating-point format, shown in Figure C-1. A number in this format consists of a sign bit s, a 24-bit significand, and an 8-bit unsigned-magnitude exponent e.

For normalized numbers, the significand consists of a 23-bit fraction f and a "hidden" bit of 1 that is implicitly presumed to precede f_{22} in the significand. The binary point is presumed to lie between this hidden bit and f_{22}. The least significant bit (LSB) of the fraction is f_0; the LSB of the exponent is e_0.

The hidden bit effectively increases the precision of the floating-point significand to 24 bits from the 23 bits actually stored in the data format. It also insures that the significand of any number in the IEEE normalized number format is always greater than or equal to 1 and less than 2.

The unsigned exponent e can range between $1 \leq e \leq 254$ for normal numbers in the single-precision format. This exponent is biased by +127 (254¸ 2). To calculate the true unbiased exponent, 127 must be subtracted from e.

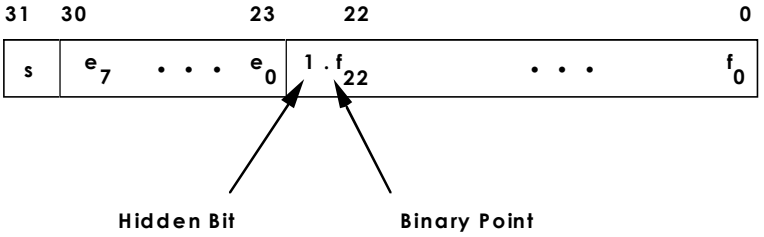

Figure C-1. IEEE 32-Bit Single-Precision Floating-Point Format

The IEEE Standard also provides for several special data types in the single-precision floating-point format:

- An exponent value of 255 (all ones) with a nonzero fraction is a Not-A-Number (NAN). NANs are usually used as flags for data flow control, for the values of uninitialized variables, and for the results of invalid operations such as $0 * \infty$.
- Infinity is represented as an exponent of 255 and a zero fraction. Note that because the fraction is signed, both positive and negative Infinity can be represented.
- Zero is represented by a zero exponent and a zero fraction. As with Infinity, both positive Zero and negative Zero can be represented.

The IEEE single-precision floating-point data types supported by the DSP and their interpretations are summarized in Table C-1.

Table C-1. IEEE Single-Precision Floating-Point Data Types

Type	Exponent	Fraction	Value
NAN	255	Nonzero	Undefined
Infinity	255	0	$(-1)^s$ Infinity
Normal	$1 \le e \le 254$	Any	$(-1)^s (1.f_{22-0}) \, 2^{e-127}$
Zero	0	0 $(-1)^s$ Zero	

Extended Precision Floating-point Format

The extended precision floating-point format is 40 bits wide, with the same 8-bit exponent as in the standard format but a 32-bit significand. This format is shown in Figure C-2. In all other respects, the extended floating-point format is the same as the IEEE standard format.

Short Word Floating-point Format

The DSP supports a 16-bit floating-point data type and provides conversion instructions for it. The short float data format has an 11-bit mantissa with a four-bit exponent plus sign bit, as shown in Figure C-3. The 16-bit floating-point numbers reside in the lower 16 bits of the 32-bit floating-point field.

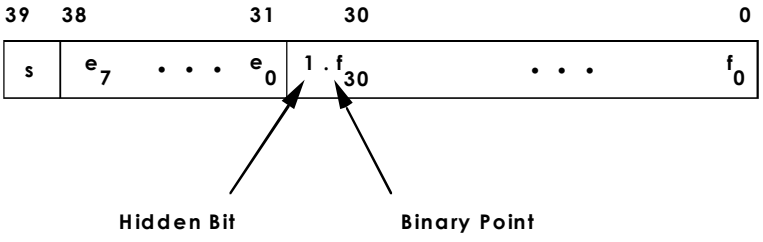

Figure C-2. 40-Bit Extended-Precision Floating-Point Format

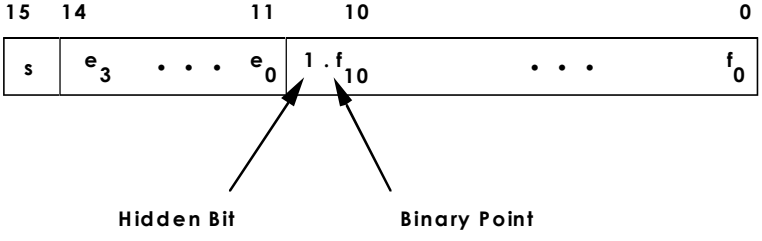

Figure C-3. 16-Bit Floating-Point Format

Packing For Floating-point Data

Two shifter instructions, FPACK and FUNPACK, perform the packing and unpacking conversions between 32-bit floating-point words and 16-bit floating-point words. The FPACK instruction converts a 32-bit IEEE floating-point number to a 16-bit floating-point number. FUNPACK converts the 16-bit floating-point numbers back to 32-bit IEEE floating-point. Each instruction executes in a single cycle. The results of

the FPACK and FUNPACK operations appear in Table C-2 and Table C-3.

Table C-2. FPACK Operations

Condition	Result
135 < exp	Largest magnitude representation.
120 < exp ≤ 135	Exponent is MSB of source exponent concatenated with the three LSBs of source exponent. The packed fraction is the rounded upper 11 bits of the source fraction.
109 < exp ≤ 120	Exponent=0. Packed fraction is the upper bits (source exponent – 110) of the source fraction prefixed by zeros and the "hidden" 1. The packed fraction is rounded.
exp < 110	Packed word is all zeros.
exp = source exponent **sign bit remains the same in all cases**	

Table C-3. FUNPACK Operations

Condition	Result
0 < exp ≤ 15	Exponent is the 3 LSBs of the source exponent prefixed by the MSB of the source exponent and four copies of the complement of the MSB. The unpacked fraction is the source fraction with 12 zeros appended.
exp = 0	Exponent is (120 – N) where N is the number of leading zeros in the source fraction. The unpacked fraction is the remainder of the source fraction with zeros appended to pad it and the "hidden" 1 stripped away.
exp = source exponent **sign bit remains the same in all cases**	

The short float type supports gradual underflow. This method sacrifices precision for dynamic range. When packing a number which would have

underflowed, the exponent is set to zero and the mantissa (including "hidden" 1) is right-shifted the appropriate amount. The packed result is a denormal which can be unpacked into a normal IEEE floating-point number.

During the FPACK operation, an overflow will set the SV condition and non-overflow will clear it. During the FUNPACK operation, the SV condition will be cleared. The SZ and SS conditions are cleared by both instructions.

Fixed-point Formats

The DSP supports two 32-bit fixed-point formats: fractional and integer. In both formats, numbers can be signed (twos-complement) or unsigned. The four possible combinations are shown in Figure C-4. In the fractional format, there is an implied binary point to the left of the most significant magnitude bit. In integer format, the binary point is understood to be to the right of the LSB. Note that the sign bit is negatively weighted in a twos-complement format.

ALU outputs always have the same width and data format as the inputs. The multiplier, however, produces a 64-bit product from two 32-bit inputs. If both operands are unsigned integers, the result is a 64-bit unsigned integer. If both operands are unsigned fractions, the result is a 64-bit unsigned fraction. These formats are shown in Figure C-5.

If one operand is signed and the other unsigned, the result is signed. If both inputs are signed, the result is signed and automatically shifted left one bit. The LSB becomes zero and bit 62 moves into the sign bit position. Normally bit 63 and bit 62 are identical when both operands are signed. (The only exception is full-scale negative multiplied by itself.) Thus, the left shift normally removes a redundant sign bit, increasing the precision of the most significant product. Also, if the data format is fractional, a single-bit left shift renormalizes the MSP to a fractional format.

The signed formats with and without left shifting are shown in Figure C-6.

The multiplier has an 80-bit accumulator to allow the accumulation of 64- bit products. For more information on the multiplier and accumulator, see "Multiply—Accumulator (Multiplier)" on page 2-13.

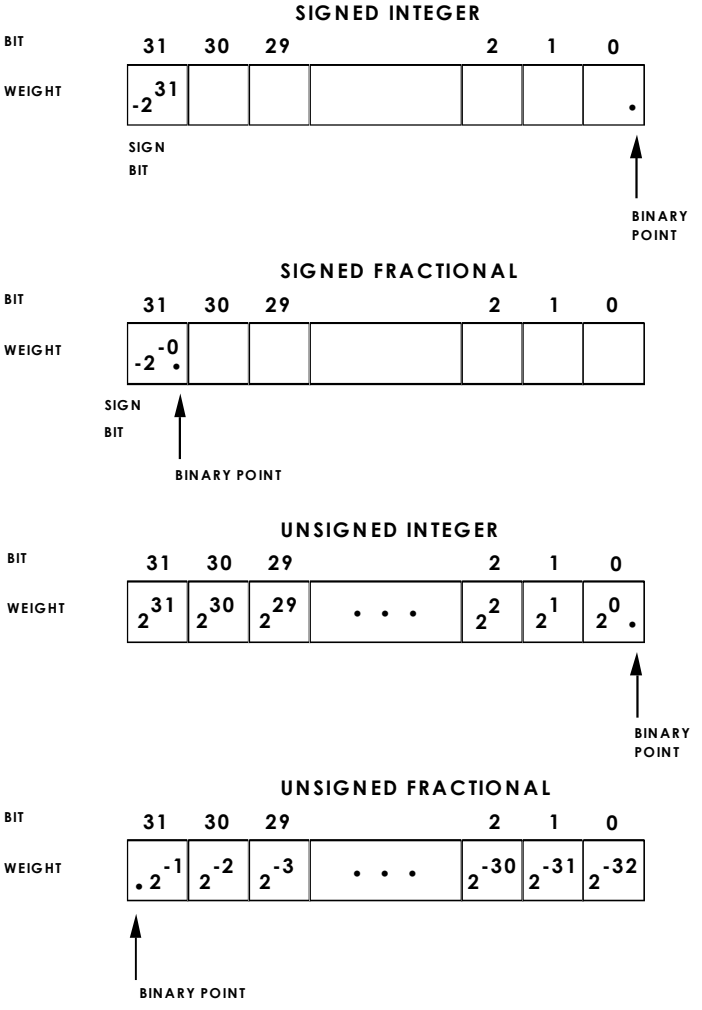

Figure C-4. 32-Bit Fixed-Point Formats

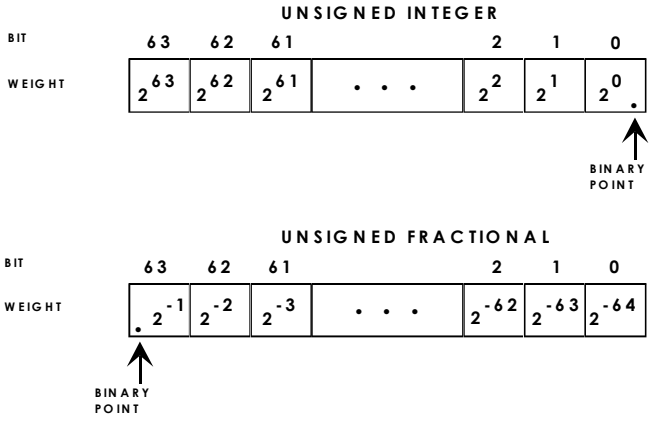

Figure C-5. 64-Bit Unsigned Fixed-Point Product

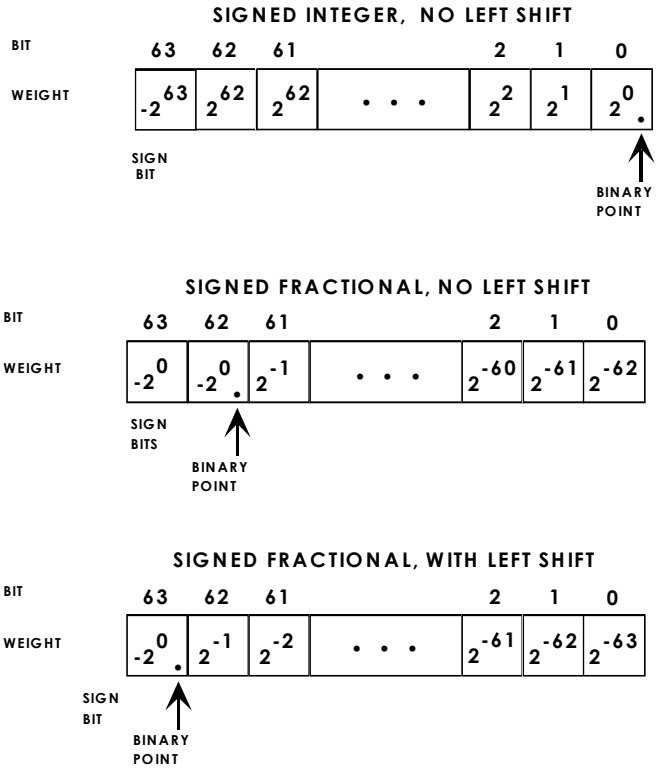

Figure C-6. 64-Bit Signed Fixed-Point Product