This chapter describes how you can use PowerPC assembly-language instructions to perform the conversions required by the IEEE standard (described in Chapter 5, "Conversions"). The assembler provides instructions that perform many of these conversions. The conversion instructions have two operands, both of which are floating-point registers. They are of the form

instr DST, *SRC*

and are interpreted as

DST ← *op SRC*

where *SRC* and *DST* are floating-point registers and *op* is some operation.

For each type of conversion, this chapter lists the assembly-language instructions you can use to perform that conversion and gives an example of how to use the instructions.

Conversions From Integer to Floating-Point Formats

No single instruction is available to convert an integer to floating-point format. However, you can perform this operation using the algorithm in Listing 13-1. First, define the following constant:

```
kmagic: word 0x43300000,0x80000000
```

This constant must have an exponent of 52 after subtracting the bias for the double format (1023), and the lower half of the constant (bit 33) must begin with a 1. In the constant `kmagic` above, the first word (eight hexadecimal digits) corresponds to the exponent part and the last word corresponds to the integer part.

When you have an integer you want to convert, invert its sign, append the exponent part of the constant to the integer to be converted, and then load it into a floating-point register with the new exponent appended. Finally, subtract the floating-point constant from the newly formed floating-point integer. The assembly code in Listing 13-1 shows how this is done. The code fragment assumes that general-purpose register GPR0 contains the value 0 and that register GPR3 contains the value to be converted.

Listing 13-1 Converting a number from integer format to floating-point format

```
addis r1,r0,0x4330   # r1 contains 0x4330000
stw   r1,20000(r0)   # store exponent part for integer
xoris r3,r3,0x8000   # invert sign of integer
stw   r3,20004(r0)   # store fraction part for integer
                     # now all parts are in memory
lfd   f0,20000(r0)   # load integer in double format into f0
lfd   f1,kmagic(r0)  # load constant into f1
fsub  f0,f0,f1       # f0 contains converted integer
```

Conversions From Floating-Point to Integer Formats

To convert numbers in floating-point format to integer format, use one of two instructions:

`fctiw`	Convert and round in current direction.
`fctiwz`	Convert and round toward zero (truncate).

To convert double-format numbers to 32-bit integers, perform the following sequence of instructions:

```
lfd   f1,d(r2)    # load double float input into f1
fctiw f2,f1       # f2 is fixed 32-bit integer version of input
stfd  f2,d(r1)    # store f2 at location d + (r1)
lwz   r3,d+4(r1)  # r3 is fixed 32-bit integer version of input
```

To convert single-format numbers to 32-bit integers, perform the following sequence of instructions:

```
lfs   f1,d(r2)    # load single float input into f1
                  # input automatically converted to double format
fctiw f2,f1       # f2 is fixed 32-bit integer version of input
stfs  f2,d(r1)    # store f2 at location d + (r1)
lwz   r3,d+4(r1)  # r3 is fixed 32-bit integer version of input
```

To truncate double- or single-format numbers, replace `fctiw` in the above examples with `fctiwz`.

Note
The conversion instructions might raise floating-point exceptions. For more information, see the Motorola *PowerPC 601 RISC Microprocessor User's Manual*. ◆

Conversions From Single to Double Format

To convert a single floating-point number to double format, you simply load a single-format number into a floating-point register; the conversion takes place automatically. The following load instructions automatically convert single format to double. These instructions raise no floating-point exceptions and treat zeros, NaNs, and Infinities like any other value.

`lfs`	Load single format.
`lfsu`	Load single format and update.
`lfsux`	Load single format and update indexed.
`lfsx`	Load single format indexed.

For more information on the load instructions, see Chapter 11, "Introduction to Assembly-Language Numerics."

Conversions From Double to Single Format

To convert a double floating-point number to single format, either store the double number in single format (listed here and described in Chapter 11) or use the `frsp` instruction.

`frsp`	Convert double to single format.
`stfs`	Store in single format.
`stfsu`	Store in single format and update.
`stfsux`	Store in single format and update indexed.
`stfsx`	Store in single format indexed.

For store instructions, the conversion takes place automatically. The store instructions raise no floating-point exceptions and treat zeros, NaNs, and Infinities like any other value. The `frsp` instruction converts a double-format number to single format and then places it in the last half of a floating-point register. Use the `frsp` instruction immediately before using the single form of any arithmetic instruction. The following example performs single-precision addition on a number that has been converted to single format using the `frsp` instruction.

```
lfs   f1,d(r1)    #load single format number into f1
                  #conversion to double format is automatic
frsp  f1,f1       #f1 is now in single format
fadds f0,f1,f1    #so that it can be added as single format number
```

Note
The `frsp` instruction might raise floating-point exceptions. See the Motorola *PowerPC 601 RISC Microprocessor User's Manual* for more information. ◆