6.4 Data Types

Table 6-1 lists the size, representation, and range of each scalar data type for the C28x compiler. Many of the range values are available as standard macros in the header file limits.h.

Table 6-1. TMS320C28x C/C++ Data Types

			Range	
Type	**Size**	**Representation**	**Minimum**	**Maximum**
char, signed char	16 bits	ASCII	-32 768	32 767
unsigned char, _Bool	16 bits	ASCII	0	65 535
short	16 bits	2s complement	-32 768	32 767
unsigned short	16 bits	Binary	0	65 535
int, signed int	16 bits	2s complement	-32 768	32 767
unsigned int	16 bits	Binary	0	65 535
long, signed long	32 bits	2s complement	-2 147 483 648	2 147 483 647
unsigned long	32 bits	Binary	0	4 294 967 295
long long, signed long long	64 bits	2s complement	-9 223 372 036 854 775 808	9 223 372 036 854 775 807
unsigned long long	64 bits	Binary	0	18 446 744 073 709 551 615
enum	16 bits	2s complement	-32 768	32 767
float	32 bits	IEEE 32-bit	1.19 209 290e-38(1)	3.40 282 35e+38
double	32 bits	IEEE 32-bit	1.19 209 290e-38(1)	3.40 282 35e+38
long double	64 bits	IEEE 64-bit	2.22 507 385e-308(1)	1.79 769 313e+308
pointers	16 bits	Binary	0	0xFFFF
far pointers	22 bits	Binary	0	0x3FFFFF

(1) Figures are minimum precision.

NOTE: TMS320C28x Byte is 16 Bits

By ANSI/ISO C definition, the sizeof operator yields the number of bytes required to store an object. ANSI/ISO further stipulates that when sizeof is applied to char, the result is 1. Since the TMS320C28x char is 16 bits (to make it separately addressable), a byte is also 16 bits. This yields results you may not expect; for example, size of (int) = = 1 (not 2). TMS320C28x bytes and words are equivalent (16 bits). To access data in increments of 8 bits, use the __byte() and __mov_byte() intrinsics described in Section 7.4.6.

6.4.1 Support for 64-Bit Integers

The TMS320C28x compiler supports the long long and unsigned long long data types. The range values are available as standard macros in the header file limits.h.

The long long data types are stored in register pairs. In memory they are stored as 64-bit objects at word (32-bit) aligned addresses.

A long long integer constant can have an ll or LL suffix. Without the suffix the value of the constant determines the type of the constant.

The formatting rules for long long in C I/O require ll in the format string. For example:

```
printf("%lld", 0x0011223344556677);
printf("%llx", 0x0011223344556677);
```

The run-time-support library functions, llabs(), strtoll() and strtoull(), are added.