```
int64Wave = expression  // Compiles signed 64-bit integer expression

uint64Wave = expression // Compiles unsigned 64-bit integer expression

tWave = expression      // Compiles text expression
```

See also **Integer Expressions in Wave Assignment Statements** on page IV-39.

The compiler is sometimes picky about the congruence between two declarations of wave reference variables of the same name. For example:

```
WAVE aWave
if (!WaveExists(aWave))
   Make/D aWave
endif
```

This generates a compile error complaining about inconsistent types for a wave reference. Because Make automatically creates a wave reference, this is equivalent to:

```
WAVE aWave
if (!WaveExists(aWave))
   Make/D aWave
   WAVE/D aWave
endif
```

This creates two wave references with the same name but different types. To fix this, change the explicit wave reference declaration to:

```
WAVE/D aWave
```

WAVE Reference Type Flags

The **WAVE** reference (see page V-1069) along with certain operations such as Duplicate can accept the following flags identifying the type of WAVE reference:

/B	8-bit signed integer destination waves, unsigned with /U.
/C	Complex destination waves.
/D	Double precision destination waves.
/I	32-bit signed integer destination waves, unsigned with /U.
/L	64-bit signed integer destination waves, unsigned with /U. Requires Igor Pro 7.00 or later.
/S	Single precision destination waves.
/T	Text destination waves.
/U	Unsigned destination waves.
/W	16-bit signed integer destination waves, unsigned with /U.
/DF	Wave holds data folder references.
/WAVE	Wave holds wave references.

These are the same flags used by the **Make**. In the case of WAVE declarations and Duplicate, they do not affect the actual wave but rather tell Igor what kind of wave is expected at runtime. The compiler uses this information to determine what kind of code to compile if the wave is used as the destination of a wave assignment statement later in the function. For example:

```
Function DupIt(wv)
   WAVE/C wv                  // complex wave
```