You can further initialize numeric and string fields using normal assignment statements.

You can further initialize WAVE, NVAR, SVAR, DFREF, and FUNCREF fields using the same syntax as used for the corresponding non-structure variables. See **Runtime Lookup of Globals** on page IV-65 and **Function References** on page IV-107.

You can also use the **StructFill** operation to automatically initialize NVAR, SVAR, and WAVE fields.

Using Structures

To use ("instantiate") a structure in a function, you must allocate a STRUCT variable using:

```
STRUCT sName name
```

where *sName* is the name of an existing structure and *name* is the local structure variable name.

To access a member of a structure, specify the STRUCT variable name followed by a "." and the member name:

```
STRUCT Point pt
pt.v= 100
```

When a member is defined as an array:

```
Structure mystruct
   Variable var1
   Variable var2[10]
   …
EndStructure
```

you must specify `[index]` to use a given element in the array:

```
STRUCT mystruct ms
ms.var2[n]= 22
```

Structure and field names must be literal and can not use `$str` notation.

The *index* value can be a variable calculated at runtime or a literal number.

If the field is itself a STRUCT, continue to append "." and field names as needed.

You can define an array of structures as a field in a structure:

```
Structure mystruct
   STRUCT Point pt[100]    // Allowed as a sub-structure
EndStructure
```

However, you can not define an array of structures as a local variable in a function:

```
STRUCT Point pt[100]       // Not allowed as a function local variable
```

Structures can be passed to functions **only** by reference, which allows them to be both input and output parameters (see **Pass-By-Reference** on page IV-59). The syntax is:

```
STRUCT sName  &varName
```

In a user function you define the input parameter:

```
Function myFunc(s)
   STRUCT mystruct &s
   …
End
```

Char and uchar arrays can be treated as zero-terminated strings by leaving off the brackets. Because the Igor compiler knows the size, the entire array can be used with no zero termination. Like normal string variables, concatenation using += is allowed but substring assignment using `[p1,p2]= subStr` is not supported.