Creating a User Defined Conversion

A user defined conversion is coded as a stored procedure subroutine (coded in **SYSPROCS**) that is passed 4 parameters. The name of the user defined conversion is the name of your subroutine. The syntax and arguments are described below.

Syntax

Subroutine Conversion_Name(*type, value, branch, return_value*)

Conversion_Name is the name of the user-defined subroutine. When calling a user-defined conversion in a BASIC+ program, pass the name of this name as the conversion argument. For example, to call a conversion subroutine called **ROMAN**, that converts Latin numerals to Roman numerals, you might use the following statement:

```
x = OConv( Data, "[ROMAN]")
```

> When calling a subroutine using the IConv() or OConv() functions, the subroutine name must be in uppercase, square brackets, enclosed in quotation marks, as illustrated above. When calling a subroutine from the Validation/Conversion display of the Form Designer, or from the Table Designer, do not enclose the name in quotation marks.

Parameters

Parameter	Description
Type	Type is used to specify whether this is an input conversion (**ICONV**) or an output conversion (**OCONV**). Pass a variable containing the literal string '**ICONV**' or '**OCONV**'.
Value	Value is used to pass the data to be converted. The conversion subroutine can manipulate value in any way. Whatever changes are made to value is reflected in the program that called the subroutine. It is wise to copy value to return_value at the beginning of the procedure, and then changing return_value as required for the conversion.
Branch	Branch, an optional argument, can be used for special processing. If a branch is used, the extra argument is specified in the [] call in the function itself, as follows: x = OConv(Data, "[ROMAN,3]") This calls the ROMAN subroutine, passing "3" to the Branch argument. Branch, an optional argument, can be used for special processing. If a branch is used, the extra argument is specified in the **[]** call in the function itself, as follows: `x = Oconv( Data, "[MYCONV, PARAM1*PARAM2*PARAM3]")` This calls the **ROMAN** subroutine, passing **"3"** to the Branch argument. *Branch* is a single argument value. However, a delimited string could be used to pass multiple values. The conversion program would then parse the string, using the Field() function with the specified delimiter. The output conversion call to a subroutine called **MYCONV**, passing 3 parameters (**PARAM1**, **PARAM2**, and **PARAM3**) would be coded as: `x = Oconv( Data, "[MYCONV, PARAM1*PARAM2*PARAM3]")`
Return_value	The variable return_value contains the result of the conversion after the subroutine processes the value parameter.

<table>
<tr>
<td>The Status() Function</td>
<td>
The conversion can use the <b>Status()</b> function to report whether a conversion has been successful.

The conventional values for this function are:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td><b>Status() = 0</b> ;* conversion OK</td></tr>
<tr><td>1</td><td><b>Status() = 1</b> ;* data cannot be converted as requested</td></tr>
<tr><td>2</td><td><b>Status() = 2</b> ;* invalid branch or conversion, display standard error message window</td></tr>
<tr><td>3</td><td><b>Status() = 3</b> ;* failed conversion, do not display standard error message window</td></tr>
</table>
</td>
</tr>
</table>