BitAnd Obtain bitwise AND of two 32-bit longs

#include <ToolUtils.h> **Toolbox Utilities**

long	**BitAnd(***op1*, *op2* **);**	
long	*op1* ;	32-bit value to be masked
long	*op2* ;	32-bit mask to apply
	returns	32-bit result of (*op1* **&** *op2* **)**

BitAnd returns the logical product (a bitwise AND) of two 32-bit values. The operands are not changed.

op1	and . . .
op2	are 32-bit long operands. The bit pattern of *op1* is masked by the bits of *op2*.

Returns: a long integer; the result of (*op1* **&** *op2* **).**

Notes: Bits in *op1* that are also set in *op2* are set to 1 in the result. All other bits of the result are cleared to 0. Note that *op1* and *op2* can be in either order without affecting the result.

This capability is native to the CPU and can be performed much faster using the C **&** (bitwise AND) operator.

```
long     x, op1, op2;

x = BitAnd( op1, op2);              /* is equivalent to . . . */
x = op1 & op2;                      /* . . . and this is MUCH faster */
```