

NOTE AC: MARSAGLIA'S MWC256 RNG

Static Function MWC256()

'Marsaglia's Multiply with carry 256 RNG, adapted to vb from C

Dim init As Integer

Dim xj, xa, xc, xcarry, ya, reg, upper, block As Double

Dim qv(1 To 256) As Double

Dim I As Integer

Dim t, TU, tl, tld, tldu, xb, xd, U2, u3, u4 As Double

Dim xhu, x1 As Double

If init = 0 Then

 xj = 123456789#

 xa = 69069#

 xc = 12345#

 xcarry = 362436#

 ya = 809430660#

 reg = 4294967296#

 upper = 281474976710656#

 block = 65536#

For I = 1 To 256

 xhu = xa * xj + xc

 x1 = xhu / reg

 xj = Fix(xhu - reg * Fix(x1))

 qv(I) = x1

Next I

init = 10

I = 0

End If

If init > 0 Then

 I = I + 1

 If I > 256 Then

 I = 1

 End If

 t = ya * qv(I) + xcarry

'Marsaglia's basic MWC long-long integer equation

 TU = Fix(t / reg)

'correct upper 32 bit number

 If t > upper Then

 tld = t - reg * TU

'defective lower 32 bit number

 tldu = Fix(tld / block)

'extract correct upper 16 bit portion of tld

 xb = ya - block * (Fix(ya / block))

'partial multiplicand of t

 xd = qv(I) - block * (Fix(qv(I) / block))

'partial multiplicand of t

 U2 = xd * xb + xcarry

'total partial with correct lower 2 bytes

 u3 = Fix(U2 / block)

'incomplete upper 2 bytes

 u4 = U2 - block * u3

'extract correct lower 2 bytes from u2

```

        t1 = tldu * block + u4          'correct lower 32 bits
Else
        t1 = t - reg * TU              'form when all 52 bits of mantissa are correct
        'Stop
End If
End If
xcarry = TU
qv(I) = t1
MWC256 = t1 / reg
End Function

```

The original version in C++ is much shorter and direct. The problem with VBA is that it does not have unsigned long integers, nor does it allow access to registers for shift operations. C++ also has other shorthand cuts that take some hard figuring on how to implement them in VBA. This makes implementation of a lot of good PRNGs written in C++ cumbersome when translated to VBA.