Rescaling random integers

Here is a C++ code fragment shows how one can use a sequential stream of random integers to produce a stream of (x, y) pairs. Each integer generates one (x, y) pair with both x and y in the range -1 to +1. The code is fairly self explanatory.

```
typedef unsigned long ULONG;

...

// Evaluate (ax+c) mod m
ULONG modlin(ULONG a, ULONG x, ULONG c, ULONG m)
{
  return (a * x + c) % m;
}

// Put integer n in range x1,x2 with the maximum integer value
double rescale(ULONG N, ULONG n, double x1, double x2)
{
  double f = static_cast<double>(n) / static_cast<double>(N);
  return x1 + f * (x2 - x1);
}

int main(int argc, char* argv[])
{

  // For the sequential random number generator
  const ULONG a = 1664525;
  const ULONG c = 1013904223;
  const ULONG m = 4294967296;

  const ULONG sidelen = 65536;    // sqrt of m

  const ULONG numtrials = 1000000;

  ULONG i_prev = 12345;  // Seed value
  for (ULONG n = 0; n < numtrials; ++n) {

    ULONG i_next = modlin(a, i_prev, c, m);
    i+prev = i_next;

    // Scale the random number to a random 2-d position
    ULONG ix = i_random % sidelen;
    ULONG iy = i_random / sidelen;

    // Scale current random integer to value from 0-1
    double x = rescale(sidelen, ix, -1, 1);
    double y = rescale(sidelen, iy, -1, 1);

    // Now we have an (x,y) pair generated from a single random integer
  }

}
```