Two methods of calculating square roots in DSP applications are described in the projects section (see separate PDF file), and these are presented here.

Project D: Newton's Method for Square Roots in QuickBasic 4.5

The file, *nwtnsqrt.bas* is an example of Newton's method. This is a generic *BASIC* program that computes the root of a 32-bit integer to within an error margin, DERROR. The root of a 32-bit integer is naturally a 16-bit integer. Emphasis is placed in what follows on speed of execution and accuracy as influenced by truncation and rounding. 32-bit integer variables are defined DEFLONG, 16-bit integers are DEFINT. Integer math in *QuickBasic* is much faster than floating-point math.

Newton's method iteratively converges on a result. Experience has shown that three to six iterations are necessary to obtain best accuracy for a 16-bit result, but here we execute as many iterations as necessary to obtain accuracy DERROR, initially defined to be one least-significant bit or $1/(2^{15}) \approx 30 \times 10^{-6}$. Note that if DERROR is small or zero, convergence may never be reached because of quantization noise. A loop counter, K, is established to count iterations. The program displays on the computer screen the argument, its root and the iteration count. Users may readily modify the program to use random numbers as arguments to time the number of roots per second it calculates.

As an experiment, try changing the line of code that produces the initial guess, OLDGUESS. An interesting try would be to set it to a low constant such as zero. When this is done, the iteration count rises for almost all arguments. This illustrates the value of a good initial guess with Newton's method.

Project E: A Fast Square-Root Algorithm Using a Small Look-Up Table

The file, *fsqrt.txt* is a machine-language example of a fast square-root algorithm. The target processor in this case is the Motorola MC68HC16Z1, a 16-bit, fixed-point DSP.