Comware Technical Services
17922 Sky Park Circle, Suite E,
Irvine, CA 92614
949.851.9600, Fax: 949.851.8431
E-mail: *sales@comwaretech.com*
www.comwaretech.com

REVIVER-1000 Application Note AN200711250

Accuracy of Transcendental Functions on REVIVER-1000

Background

The implementation of the emulation for floating point operation produces different but 2 times more accurate result as compared to the legacy HP1000 machines (MEF or A-Series). This application note compares the results of transcendental functions between legacy (HP1000) and emulated system (REVIVER-1000 a.k.a. Kestrel).

Floating point implementation

REVIVER-1000 emulator uses INTEL i486 math coprocessor for all floating point calculations. The coprocessor implements the floating point arithmetic as defined in IEEE standard. Modern coprocessor architecture simplifies the implementation of the floating point instructions; as a result, the REVIVER-1000 system realizes better performance and reduces the error to less than half of the legacy system. The emulator maps single and double precision calculations of HP1000 into extended real precision operation implemented in INTEL i486 coprocessor.

Table 1: Real data representation at the hardware level

	HP1000		**IEEE (i486)**
	Single precision	**Double precision**	**Extended precision**
Number of Bits [1]	32 bits	64 bits	80 bits
Bits for exponent [2]	7 bits + 1 sign bit	7 bits + 1 sign bit	15 bits
Bits for fraction [3]	23 bits + 1 sign bit	55 bits + 1 sign bit	64 bits + 1 sign bit

Transcendental function implementation

HP1000 implement transcendental functions (such as sine, cosine) as part of scientific instruction set. This instruction set is implemented in firmware and uses single precision arithmetic. The firmware uses mainly Taylor series method to approximate trigonometric calculations. The error is enlarged due to error accumulations in single precision, an artifact of a series of calculation. In

[1] Total number of binary bits that represent exponent and mantissa.

[2] Exponent is the power of two by which the mantissa is multiplied. It can be considered as the scale of the floating point number.

[3] Fraction is a string of binary digits that represent the mantissa without the first digit.

contrast, the emulator leverages readily available transcendental functions of INTEL i486, which has much better accuracy as shown below.

Test setup

A small test program, written in FORTRAN-77 as shown in Figure 1 below, calculates trigonometric functions sine and cosine. This test software was run on legacy A900 and on our emulator system. The output of each test run is compiled into a spreadsheet and compared against ideal results to calculate the relative error of each point.

Figure 1: FORTRAN test source code

```
ftn7x,q,s
      program Example
      implicit none

      real*4    x, sx, cx
      integer*2 i

      do 1 i = -201, 201, 1
      x = float(i) / 64.0
      sx = sin(x)
      cx = cos(x)
    1 write(1, 99) i, x, sx, cx

   99 format(1X,I4'/64',3F15.10)

      end
```

This program lists 403 calculation results of Sin(x) and Cos(x) in single precision (real*4, 32-bits) mode. The results are listed with 10 digits after decimal point.

$$S(x_i) = Sin(x_i) \quad for\ all\ integer \quad -201 \le i \le +201$$

$$C(x_i) = Cos(x_i) \quad for\ all\ integer \quad -201 \le i \le +201$$

$$where\ \ x_i = \frac{i}{64}\ \ is\ the\ angle\ in\ rad,\ and\ 1\,rad = \frac{180^o}{\pi} \approx 57.29^o$$

The absolute error of each calculation point is defined as follow:

$$S_{AbsErr}(x_i) = \left|S(x_i) - S_{True}(x_i)\right| \quad for\ all\ integer \quad -201 \le i \le +201$$

$$C_{AbsErr}(x_i) = \left|C(x_i) - C_{True}(x_i)\right| \quad for\ all\ integer \quad -201 \le i \le +201$$

$$where\ \ S_{True}(x_i)\ \ and\ \ C_{True}(x_i)\ \ are\ the\ ideal\ values,\ derived\ from\ Excel\ spreadsheet$$

The relative error is defined as follow:

$$S_{RelErr}(x_i) = \frac{S_{AbsErr}(x_i)}{S_{True}(x_i)} \quad \textit{for all integer} \quad -201 \le i \le +201$$

$$C_{RelErr}(x_i) = \frac{C_{AbsErr}(x_i)}{C_{True}(x_i)} \quad \textit{for all integer} \quad -201 \le i \le +201$$

Test result

Figure 2: Distribution of relative error for Sin(x)

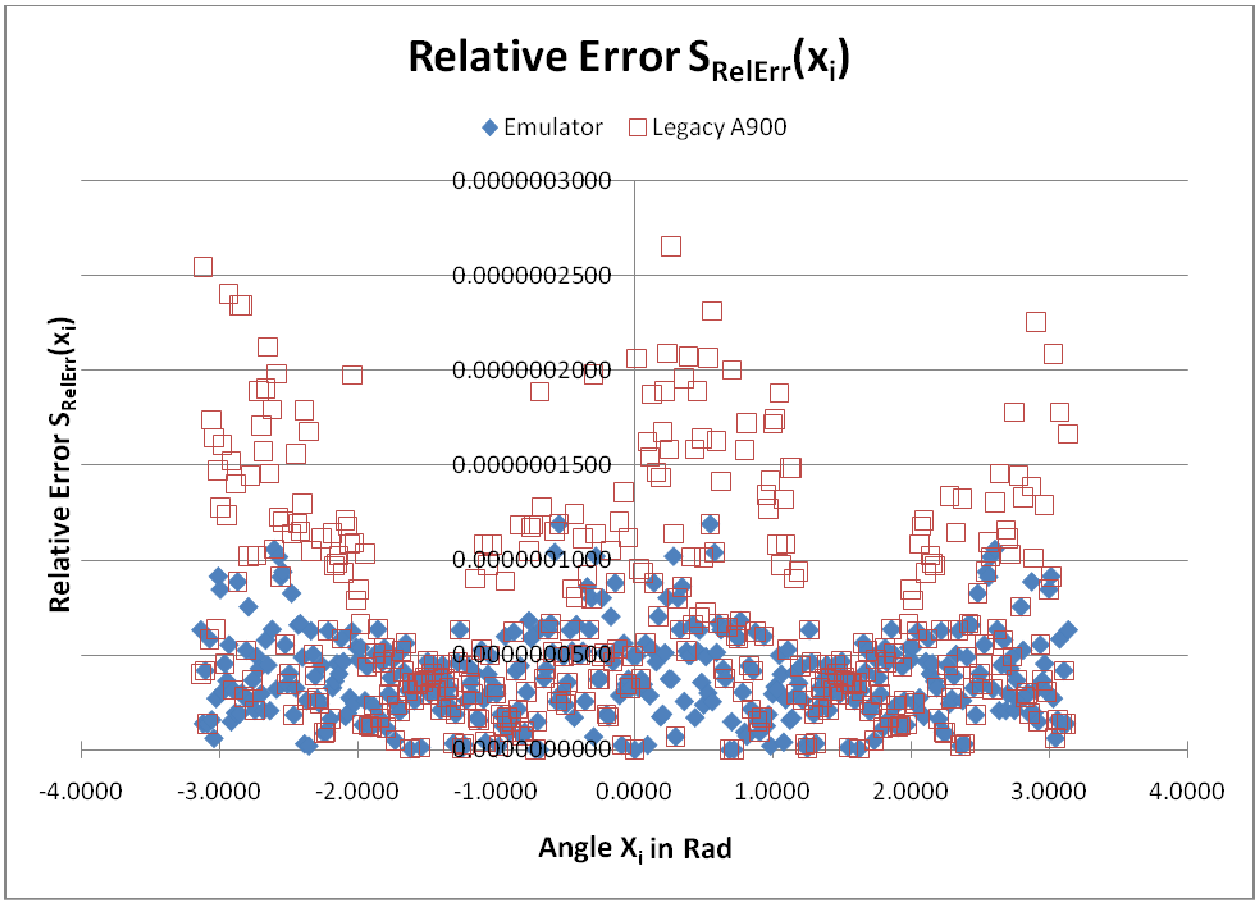
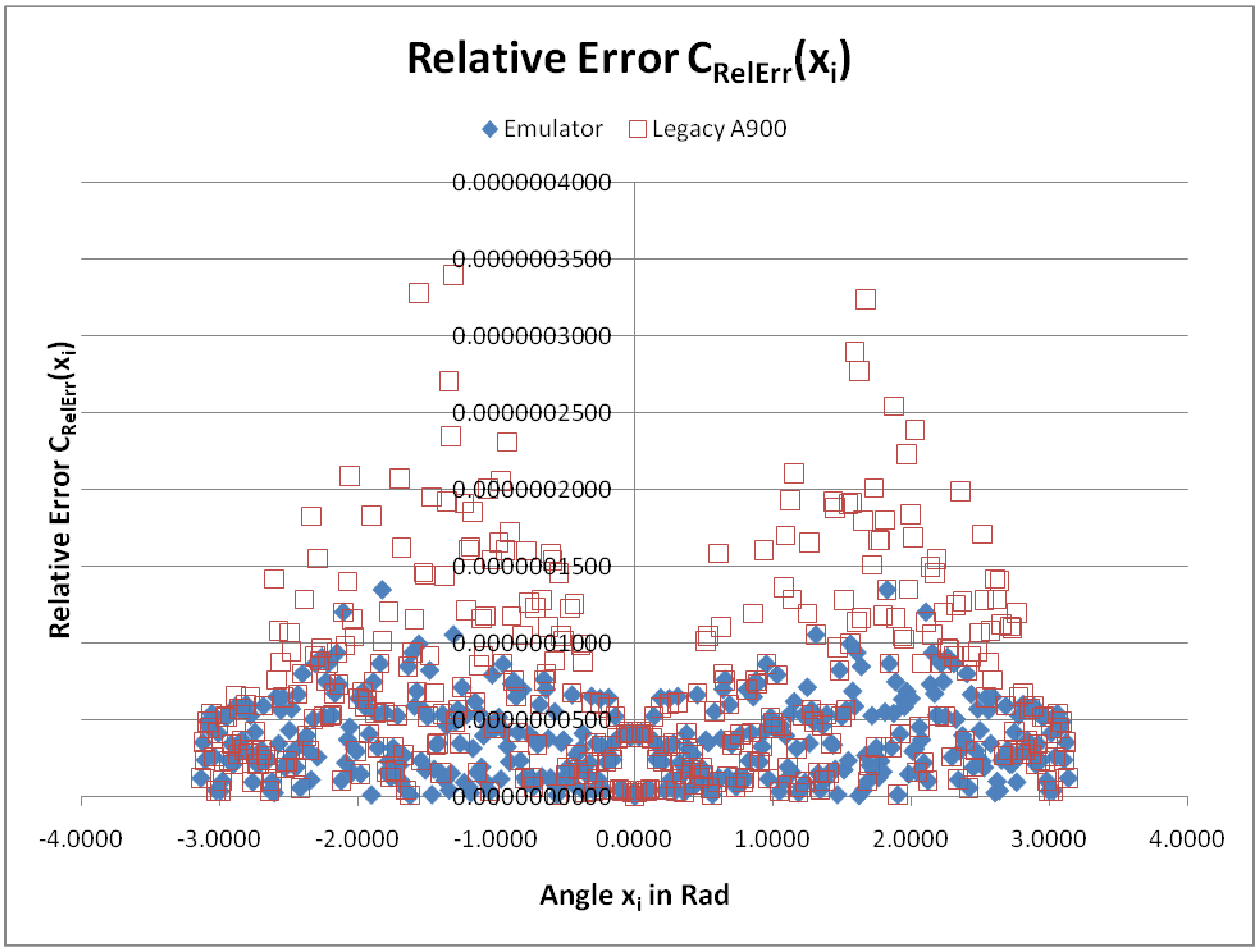


Figure 3: Distribution of relative error for Cos(x)



The relative error of the legacy A900 is more than twice of the result generated by the emulator, as shown in the distribution charts above and the table below. The shape of the error distribution for both calculations is very similar.

Table 2: Error

	Legacy A900	**Emulator**
Maximum Absolute Error $Sin_{MaxAbsErr} = \max\{S_{AbsErr}(x_i)\| -201 \le i \le +201\}$	0.0000001756	0.0000000620
Maximum Absolute Error $Cos_{MaxAbsErr} = \max\{C_{AbsErr}(x_i)\| -201 \le i \le +201\}$	0.0000001410	0.0000000620
Maximum Relative Error $Sin_{Max\,RelErr} = \max\{S_{RelErr}(x_i)\| -201 \le i \le +201\}$	0.0000002658	0.0000001191

	Legacy A900	**Emulator**
Maximum Relative Error $Cos_{Max\,RelErr} = \max\{C_{RelErr}(x_i) \mid -201 \le i \le +201\}$	0.0000003402	0.0000001346
RMS Relative Error $Sin_{Rms\,RelErr} = \sqrt{\frac{1}{403} \bullet \sum_{i=-201}^{+201} S_{RelErr}^2(x_i)}$	0.0000000935	0.0000000456
RMS Relative Error $Cos_{Rms\,RelErr} = \sqrt{\frac{1}{403} \bullet \sum_{i=-201}^{+201} C_{RelErr}^2(x_i)}$	0.0000000968	0.0000000462

The error generated by the emulator is within the accuracy limits for single precision calculation (with 23 bits mantissa). The expected error in a single precision floating point representation of a random real number is at most one half of the least significant bit, which yields a relative error between approximately 2^{-23} and 2^{-24}, (i.e., between 0.0000001192 and 0.0000000596).

Conclusion

The result of transcendental calculations from the emulator is more accurate, thanks to modern IEEE floating point hardware.