IEEE International Symposium on Industrial Embedded Systems
Lisbon, Portugal, 4-6 July 2007

Faster floating-point square root for integer processors

Guillaume Revy[a], jointly with Claude-Pierre Jeannerod[a],
Hervé Knochel[b] and Christophe Monat[b].

Arénaire INRIA project-team[a] / STMicroelectronics Compilation Expertise Centre[b]

Context and motivation

Context

- Fast and accurate software for floating-point mathematical functions

$$+, -, \times, /, \sqrt{\ }, 1/\sqrt{\ }, ...$$

- ST231 = integer processor for embedded media systems $\rightarrow$ no FPU

- Floating-Point Library for Integer Processors
  $\longrightarrow$ single precision / various rounding modes (IEEE-754 standard)

  `https://lipforge.ens-lyon.fr/projects/flip/`

Why start with square root ?

- ST231 : used in audio/video domains (HD-IPTV, cell phones, ...)
  $\longrightarrow$ highly demanding on floating-point square root computations

- The square root : one of the simplest functions

Goal

Exploit at best the ST231 architecture features : ILP / 32-bit registers

$\longrightarrow$ to implement fast algorithms using polynomial approximation evaluation,

$\longrightarrow$ to achieve correct rounding-to-nearest of square root.

Some properties of square root computation

Input : normal single precision floating-point number (IEEE-754 standard)

$$x = (-1)^s \cdot 1.f \cdot 2^e,$$

with $s \in \{0,1\}$, $e \in \mathbb{N} \cap [-126, 127]$, and $f = 0.f_1 f_2 \dots f_{23} \in [0,1)$.

s	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	f_{22}	f_{23}

$E = e + 127$

Output : correct rounding-to-nearest of $\sqrt{x}$: $\circ(\sqrt{x})$ or exception.

Main step : computation of $\sqrt{1.f}$

$\rightarrow$ Restoring/Non-restoring methods

$\rightarrow$ Newton-Raphson/Goldschmidt iterations

Some properties of square root computation

Input : normal single precision floating-point number (IEEE-754 standard)

$$x = (-1)^s \cdot 1.f \cdot 2^e,$$

with $s \in \{0,1\}$, $e \in \mathbb{N} \cap [-126, 127]$, and $f = 0.f_1 f_2 \ldots f_{23} \in [0,1)$.

s	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	f_{22}	f_{23}

$E = e + 127$

Output : correct rounding-to-nearest of $\sqrt{x}$: $\circ(\sqrt{x})$ or exception.

Main step : computation of $\sqrt{1.f}$

$\longrightarrow$ Restoring/Non-restoring methods

$\longrightarrow$ Newton-Raphson/Goldschmidt iterations

Results and perspectives

Evaluation of polynomial approximation

- Approximation of $\sqrt{1+X}$, for $X \in [0,1)$, using one or several polynomials
- Evaluation using fast, parallel schemes (Estrin-like)

Results for rounding-to-nearest and normal numbers

- 2 polynomials of degree 6
- 25 cycles $\rightarrow$ speedup $\approx$ 50 % (previous version : 48 cycles)

Actual works

- Other rounding modes / Subnormal floating-point numbers

Perspectives

- Extension to other functions : $1/\sqrt{x}$, $x^{-1/4}$, $1/x$, x/y, ...

Results and perspectives

Evaluation of polynomial approximation

- Approximation of $\sqrt{1+X}$, for $X \in [0,1)$, using one or several polynomials
- Evaluation using fast, parallel schemes (Estrin-like)

Results for rounding-to-nearest and normal numbers

- 2 polynomials of degree 6
- 25 cycles $\rightarrow$ speedup $\approx$ 50 % (previous version : 48 cycles)

Actual works

- Other rounding modes / Subnormal floating-point numbers

Perspectives

- Extension to other functions : $1/\sqrt{x}$, $x^{-1/4}$, $1/x$, x/y, ...