Exemple d'implantation de fonction mathématique sur ST200

Guillaume Revy
Encadrants : Claude-Pierre Jeannerod et Gilles Villard

Équipe INRIA Arénaire
Laboratoire de l'Informatique du Parallélisme - ENS Lyon

Journée Arénaire/Compsys/STMicroelectronics
Lyon, 12 septembre 2007






Contexte et objectifs

Bibliothèque FLIP

- Support logiciel pour l'arithmétique flottante simple précision aux processeurs entiers

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

- Fonctions mathématiques rapides et précises

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

- Racine carrée : une des fonctions les plus simples...

Contexte et objectifs

Bibliothèque FLIP

- Support logiciel pour l'arithmétique flottante simple précision aux processeurs entiers

  `https://lipforge.ens-lyon.fr/projects/flip/`
- Fonctions mathématiques rapides et précises

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

- Racine carrée : une des fonctions les plus simples...

Objectifs pour la racine carrée

- Qualité visée : arrondi correct au plus près, sans nombres dénormalisés
- Autres modes d'arrondi / Ajout des nombres dénormalisés
- Différents formats : *medium/high precision* (OpenGL ES)

Méthode générale

Soit x un nombre flottant simple précision normalisé positif (IEEE-754) :

$$x = m \cdot 2^e,$$

avec $m = 1.f_1 f_2 f_3 \ldots f_{23} \in [1, 2)$, et $e \in \mathbb{Z} \cap [-126, 127]$.

$$\Rightarrow \sqrt{x} = \sqrt{m} \times 2^{\frac{e}{2}} = \begin{cases} \sqrt{m} \times 2^{\lfloor \frac{e}{2} \rfloor} & \text{si } e \text{ est pair,} \\ \sqrt{2m} \times 2^{\lfloor \frac{e}{2} \rfloor} & \text{si } e \text{ est impair.} \end{cases}$$

Méthode générale

Soit x un nombre flottant simple précision normalisé positif (IEEE-754) :

$$x = m \cdot 2^e,$$

avec $m = 1.f_1 f_2 f_3 \ldots f_{23} \in [1, 2)$, et $e \in \mathbb{Z} \cap [-126, 127]$.

$$\Rightarrow \sqrt{x} = \sqrt{m} \times 2^{\frac{e}{2}} = \begin{cases} \sqrt{m} \times 2^{\lfloor \frac{e}{2} \rfloor} & \text{si } e \text{ est pair,} \\ \sqrt{2m} \times 2^{\lfloor \frac{e}{2} \rfloor} & \text{si } e \text{ est impair.} \end{cases}$$

Finalement : $\sqrt{x} = \ell \times 2^d$, avec $\ell = \varphi\sqrt{m}$, $\varphi \in \{1, \sqrt{2}\}$ et $d = \lfloor \frac{e}{2} \rfloor$.

Méthode générale

Soit x un nombre flottant simple précision normalisé positif (IEEE-754) :

$$x = m \cdot 2^e,$$

avec $m = 1.f_1 f_2 f_3 \ldots f_{23} \in [1, 2)$, et $e \in \mathbb{Z} \cap [-126, 127]$.

$$\Rightarrow \sqrt{x} = \sqrt{m} \times 2^{\frac{e}{2}} = \begin{cases} \sqrt{m} \times 2^{\lfloor \frac{e}{2} \rfloor} & \text{si } e \text{ est pair,} \\ \sqrt{2m} \times 2^{\lfloor \frac{e}{2} \rfloor} & \text{si } e \text{ est impair.} \end{cases}$$

Finalement : $\sqrt{x} = \ell \times 2^d$, avec $\ell = \varphi\sqrt{m}$, $\varphi \in \{1, \sqrt{2}\}$ et $d = \lfloor \frac{e}{2} \rfloor$.

Avantage : pas de renormalisation en arrondi au plus près

$\rightarrow$ $m \in [1, 2)$ donc $\circ(\varphi\sqrt{m}) \in [1, 2)$

$\rightarrow$ $\circ(\sqrt{x}) = \circ(\varphi\sqrt{m}) \times 2^d$

Principales étapes

Entrée : un nombre flottant x simple précision normalisé ou une valeur spéciale, dans un registre 32 bits.

1. Extraction de l'exposant e et de la mantisse m de x
2. Traitement des valeurs spéciales

x	NaN / $x < 0$ / $-\infty$	$+\infty$	± 0
$\sqrt{x}$	NaN	$+\infty$	± 0

3. Calcul de l'exposant du résultat : $d = \lfloor \frac{e}{2} \rfloor$
4. Calcul de $v \approx \varphi\sqrt{m}$, tel que : $0 \leq v - \varphi\sqrt{m} \leq 2^{-24}$
5. Arrondi au plus près : calcul $\circ(\varphi\sqrt{m})$ à partir de v
6. Reconstruction du résultat dans un registre 32 bits

Sortie : arrondi correct au plus près de $\sqrt{x}$, ou une exception.

Exemple détaillé

Calcul de $\circ(\sqrt{17}) : 17 = 1.0625 \times 2^4$

Entrée $x = 17$

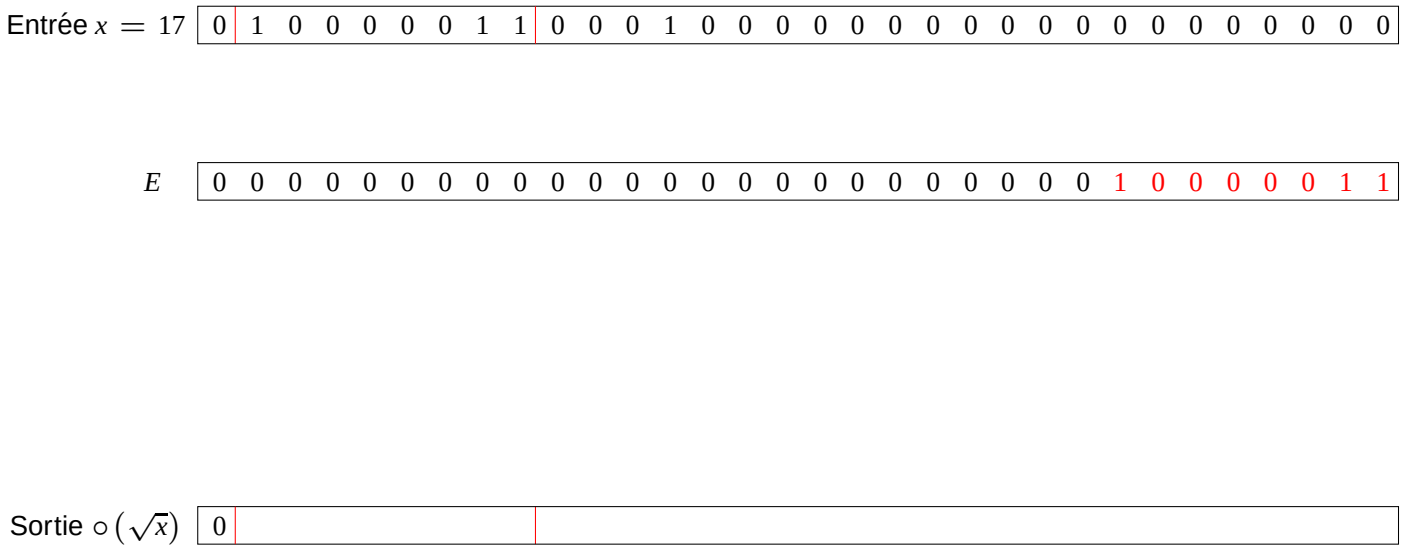
0	1	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Sortie $\circ(\sqrt{x})$

0		

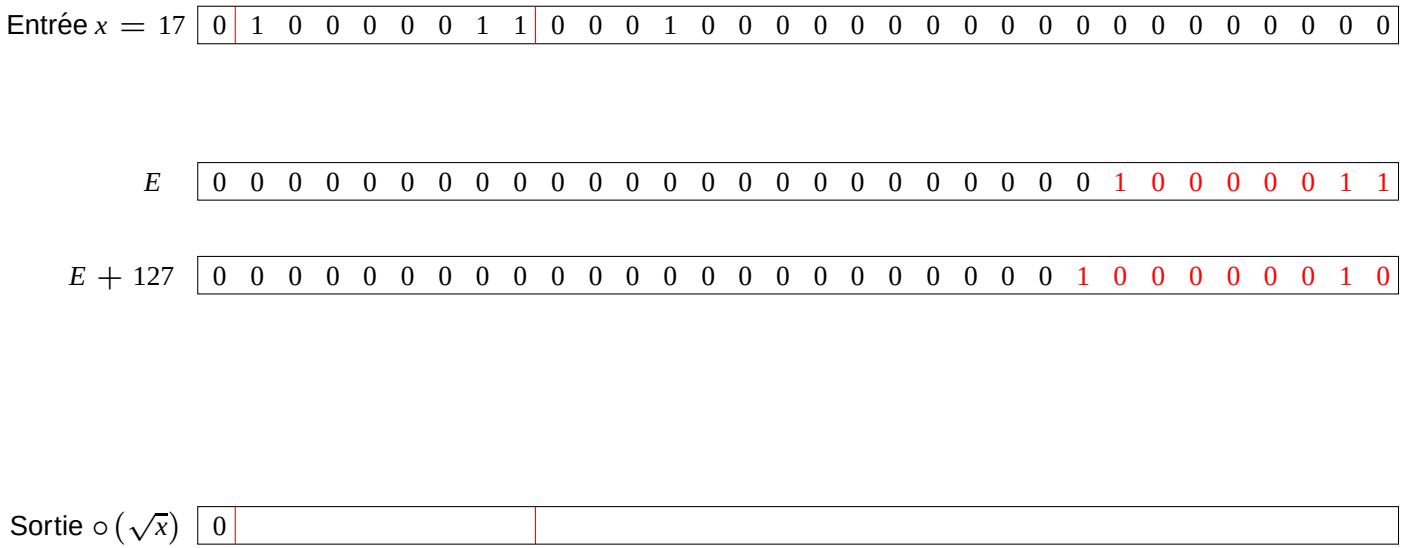
Exemple détaillé

Calcul de l'exposant du résultat



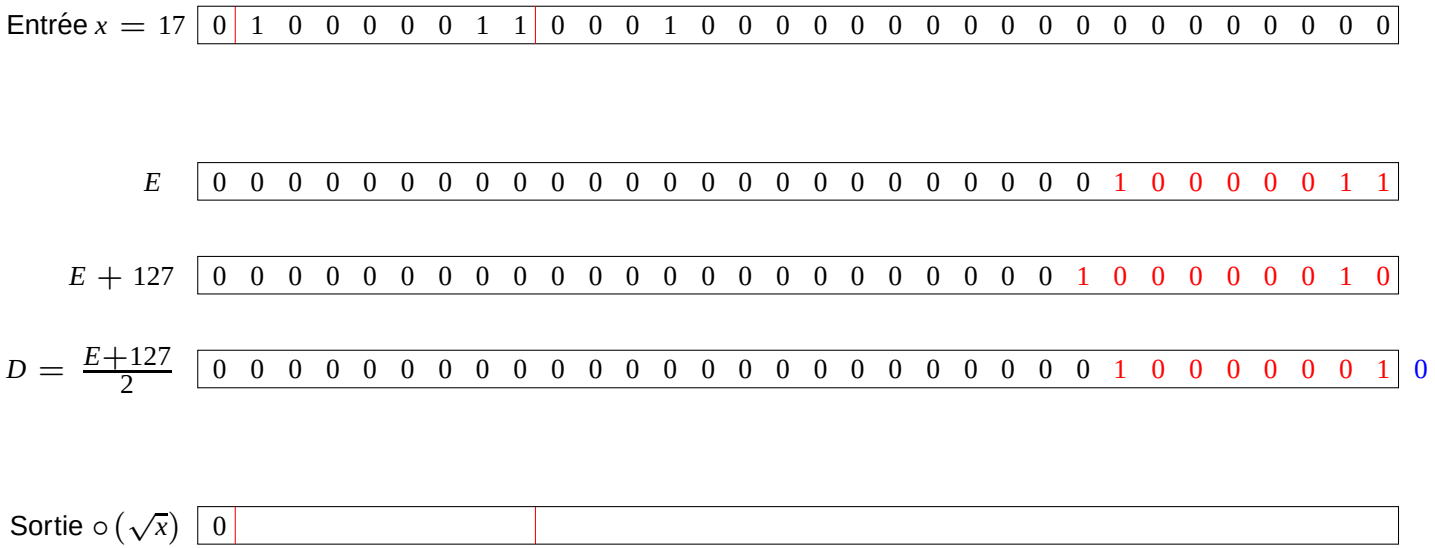
Exemple détaillé

Calcul de l'exposant du résultat



Exemple détaillé

Calcul de l'exposant du résultat



Exemple détaillé

Calcul de l'exposant du résultat

Entrée $x = 17$	0	1	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1
$E + 127$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
$D = \frac{E+127}{2}$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
Sortie $\circ(\sqrt{x})$	0	1	0	0	0	0	0	0	1																								

Exemple détaillé

Calcul de la fraction du résultat

Entrée $x = 17$

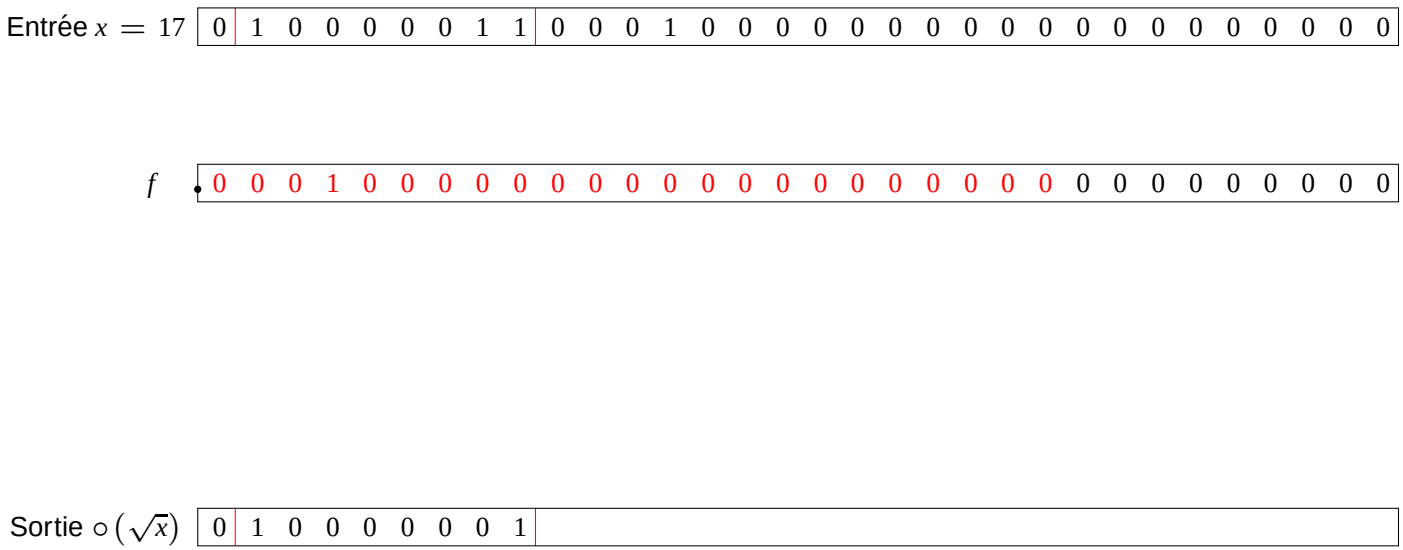
0	1	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Sortie $\circ\left(\sqrt{x}\right)$

0	1	0	0	0	0	0	0	1																							

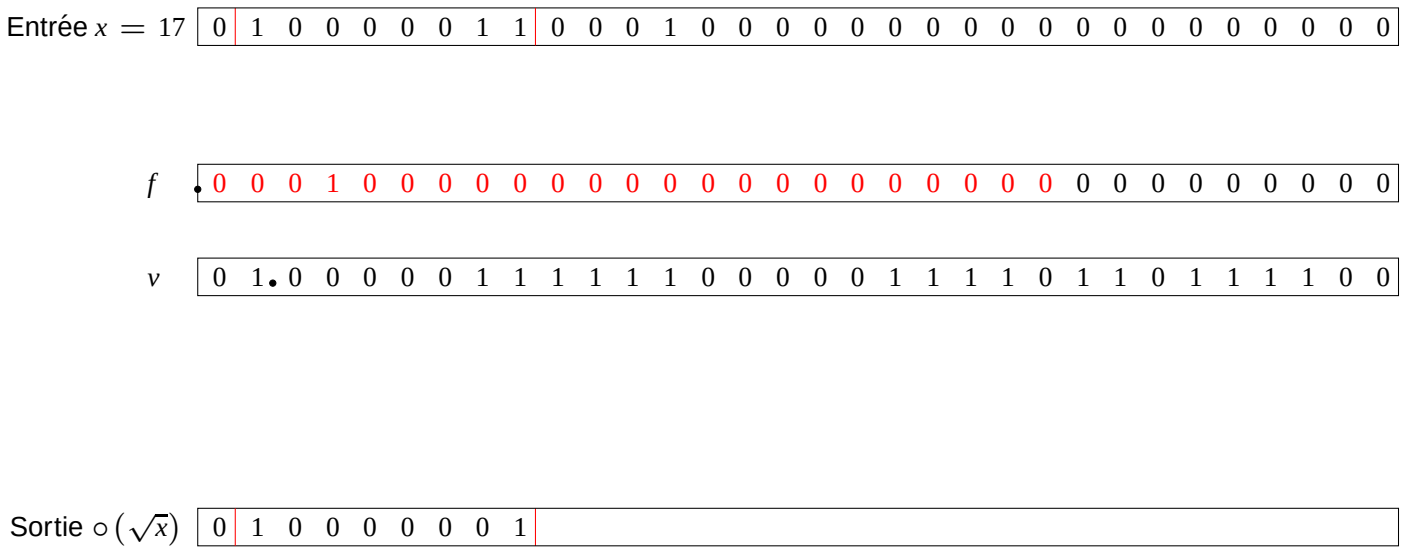
Exemple détaillé

Calcul de la fraction du résultat



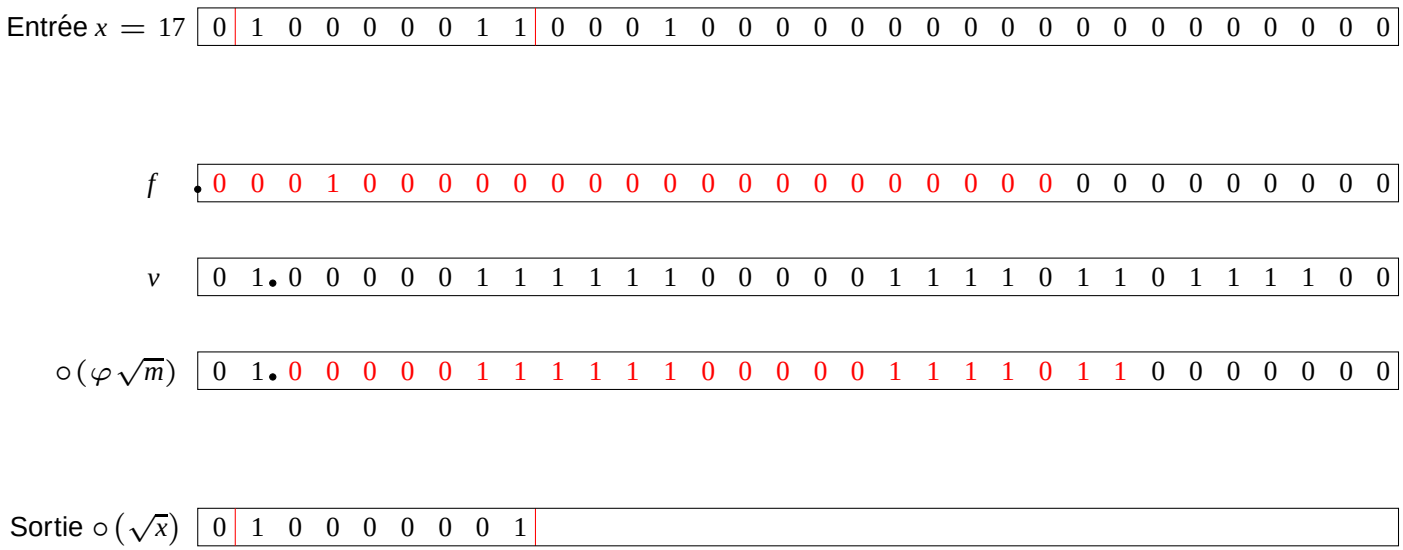
Exemple détaillé

Calcul de la fraction du résultat



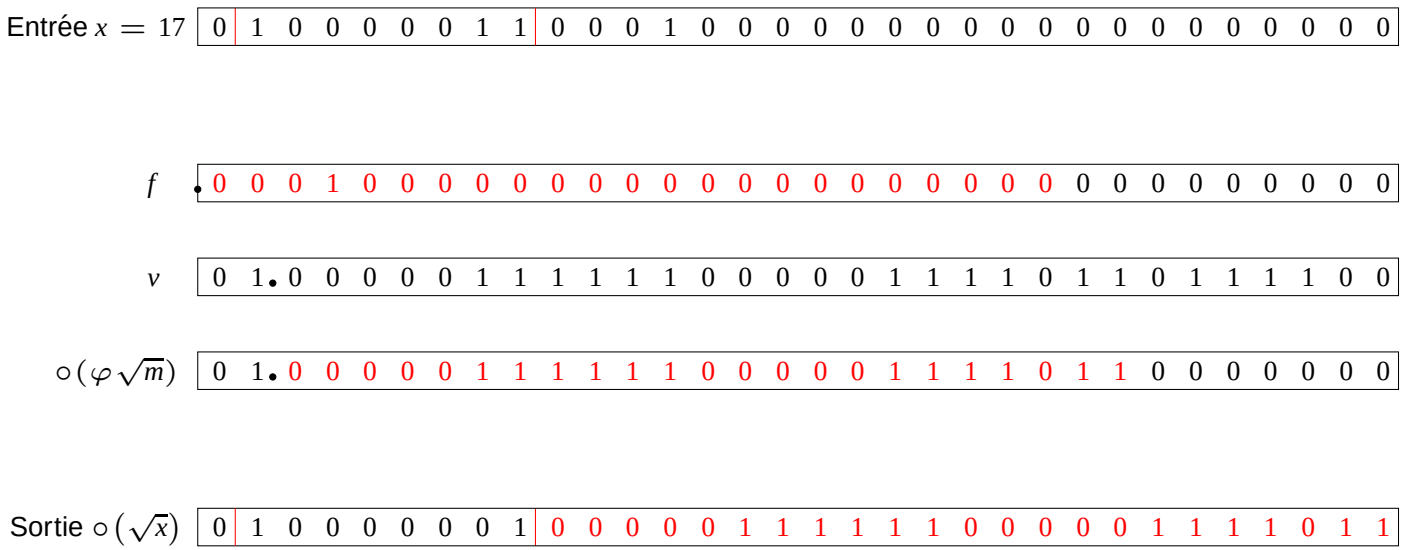
Exemple détaillé

Calcul de la fraction du résultat



Exemple détaillé

Calcul de la fraction du résultat



Exemple détaillé

Calcul de $\circ(\sqrt{17}) : 17 = 1.0625 \times 2^4$

Entrée $x = 17$

0	1	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Sortie $\circ(\sqrt{x})$

0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	0	1	1

$\approx 4.123105... \approx \circ(\sqrt{17})$

Quelle méthode utiliser pour calculer $\varphi\sqrt{m}$?

Méthodes directes : restaurante / non-restaurante

- 1 bit du résultat calculé à chaque itération : 24 itérations
- méthodes lentes

Quelle méthode utiliser pour calculer $\varphi\sqrt{m}$?

Méthodes directes : restaurante / non-restaurante

- 1 bit du résultat calculé à chaque itération : 24 itérations
- méthodes lentes

Méthodes itératives Newton-Raphson / Goldschmidt

- première approximation de $\sqrt{m}$ ou $\frac{1}{\sqrt{m}}$
- en gros, la précision double à chaque itération
- méthode précédente : 1 itération de Goldschmidt

Quelle méthode utiliser pour calculer $\varphi\sqrt{m}$?

Méthodes directes : restaurante / non-restaurante

- 1 bit du résultat calculé à chaque itération : 24 itérations
- méthodes lentes

Méthodes itératives Newton-Raphson / Goldschmidt

- première approximation de $\sqrt{m}$ ou $\frac{1}{\sqrt{m}}$
- en gros, la précision double à chaque itération
- méthode précédente : 1 itération de Goldschmidt

Autres méthodes méthodes SRT, approximations par série entière, ...

Quelle méthode utiliser pour calculer $\varphi\sqrt{m}$?

Méthodes directes : restaurante / non-restaurante

- 1 bit du résultat calculé à chaque itération : 24 itérations
- méthodes lentes

Méthodes itératives Newton-Raphson / Goldschmidt

- première approximation de $\sqrt{m}$ ou $\frac{1}{\sqrt{m}}$
- en gros, la précision double à chaque itération
- méthode précédente : 1 itération de Goldschmidt

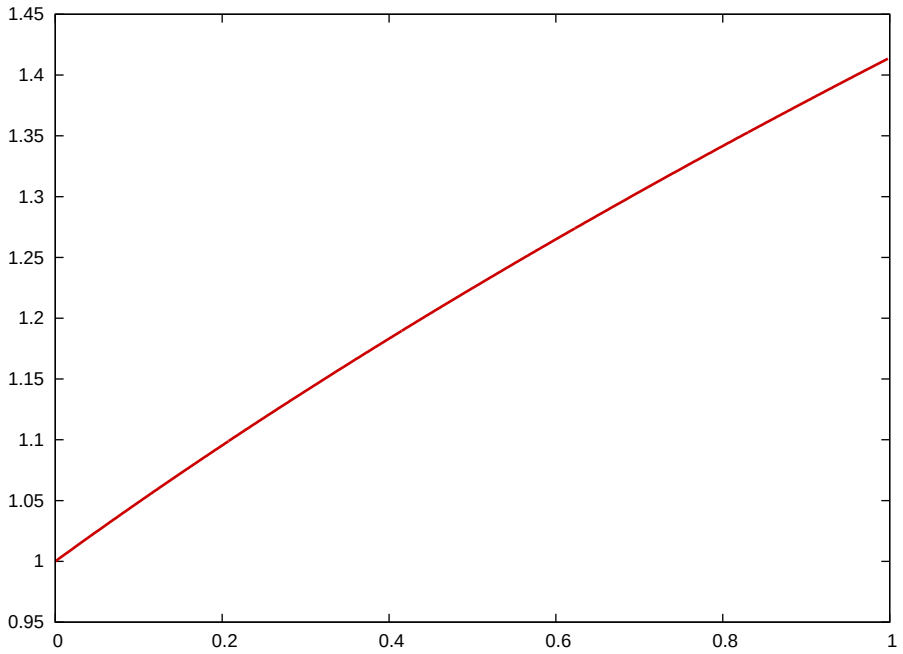
Autres méthodes méthodes SRT, approximations par série entière, ...

Notre approche méthode à base d'évaluation polynomiale

- approximation de $\sqrt{m}$ par un polynôme
- évaluation du polynôme avec un schéma rapide

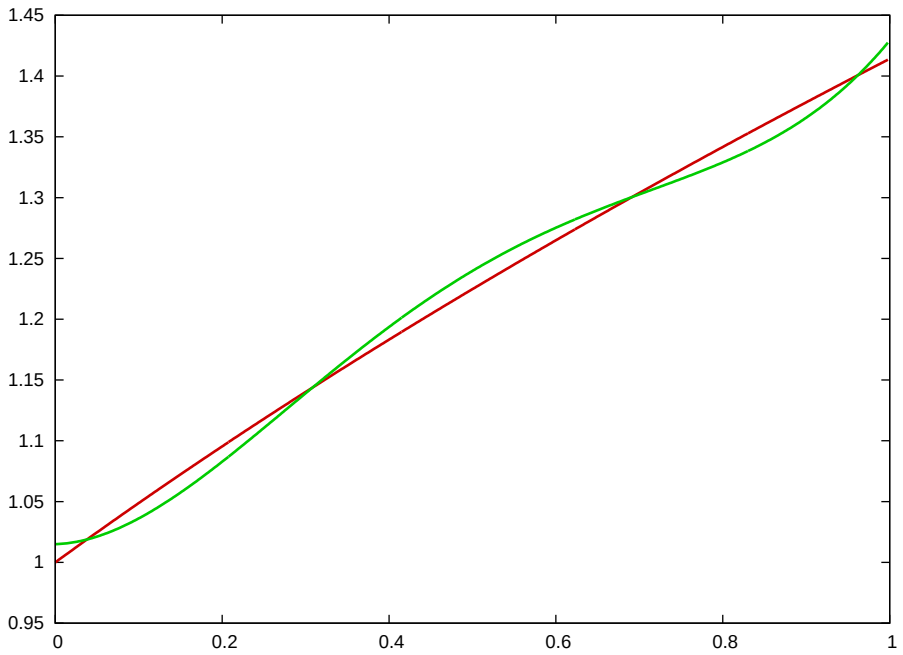
Approximation de $\sqrt{m}$ pour $m \in [1, 2)$

Pour $m \in [1, 2)$, $\sqrt{m} = \sqrt{1 + t}$ avec $t \in [0, 1)$.



Approximation de $\sqrt{m}$ pour $m \in [1, 2)$

Pour $m \in [1, 2)$, $\sqrt{m} = \sqrt{1+t}$ avec $t \in [0, 1)$.

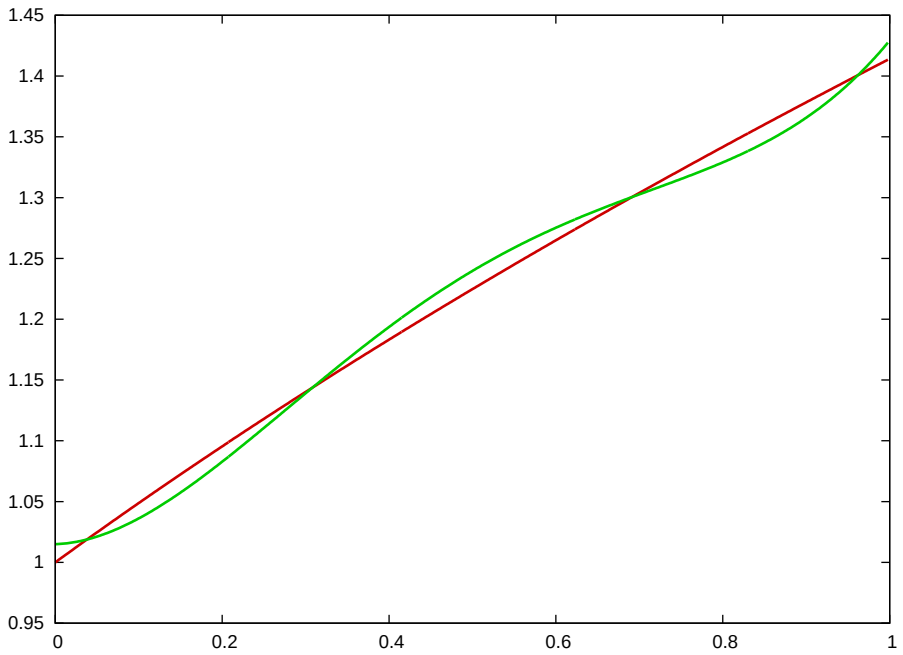


- approximation de $\sqrt{1+t}$ par $a(t)$, pour $t \in [0, 1)$
- $a(t)$: polynôme de degré 8
- obtenu avec `minimax` (MAPLE)

Cf. travaux de N. Brisebarre, S. Chevillard, C. Lauter, JM. Muller et S. Torres

Approximation de $\sqrt{m}$ pour $m \in [1, 2)$

Pour $m \in [1, 2)$, $\sqrt{m} = \sqrt{1+t}$ avec $t \in [0, 1)$.

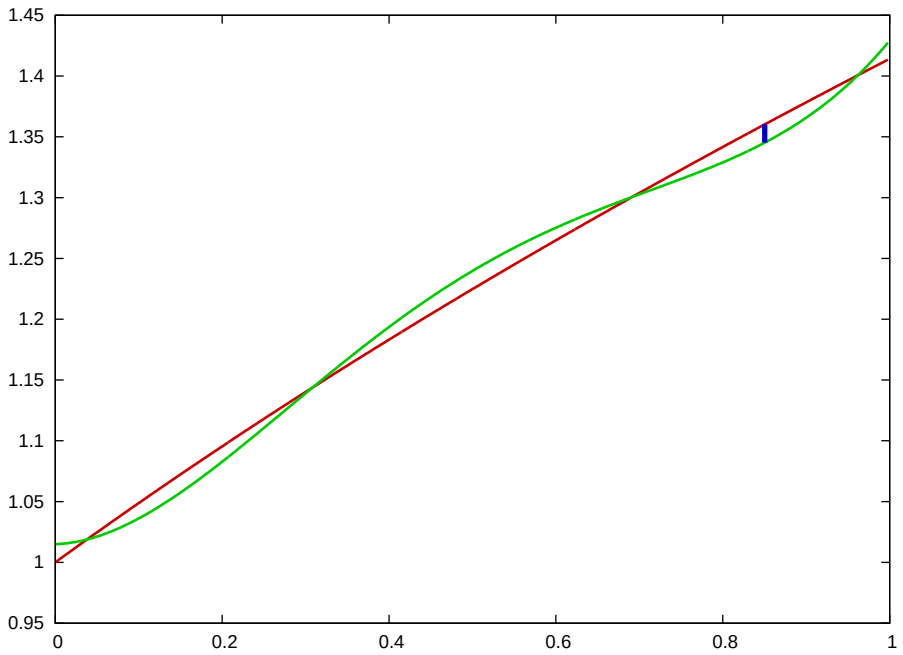


- approximation de $\sqrt{1+t}$ par $a(t)$, pour $t \in [0, 1)$
- $a(t)$: polynôme de degré 8
- obtenu avec `minimax` (MAPLE)
- coefficients structurés :
  - $\rightarrow$ $a_0 = 1$
  - $\rightarrow$ $a_1 = 1/2$
  - $\rightarrow$ $a_8 = 1/2^{10}$

Cf. travaux de N. Brisebarre, S. Chevillard, C. Lauter, JM. Muller et S. Torres

Approximation de $\sqrt{m}$ pour $m \in [1, 2)$

Pour $m \in [1, 2)$, $\sqrt{m} = \sqrt{1+t}$ avec $t \in [0, 1)$.



- approximation de $\sqrt{1+t}$ par $a(t)$, pour $t \in [0, 1)$
- $a(t)$: polynôme de degré 8
- obtenu avec `minimax` (MAPLE)
- coefficients structurés :
  - $\rightarrow$ $a_0 = 1$
  - $\rightarrow$ $a_1 = 1/2$
  - $\rightarrow$ $a_8 = 1/2^{10}$
- erreur d'approximation $\delta(t) = |a(t) - \sqrt{1+t}| \lesssim 2^{-26}$

Cf. travaux de N. Brisebarre, S. Chevillard, C. Lauter, JM. Muller et S. Torres

Quel schéma utiliser pour évaluer $\varphi a(t)$ sur ST240 ?

Quelques caractéristiques

- VLIW 4 voies
- latences : 3 cycles/multiplication et 1 cycle/addition

Quel schéma utiliser pour évaluer $\varphi a(t)$ sur ST240 ?

Quelques caractéristiques

- VLIW 4 voies
- latences : 3 cycles/multiplication et 1 cycle/addition

Schéma de Horner $\varphi a(t) = \varphi\Big[\Big(\cdots\big((a_8 t + a_7)t + a_6\big)\ldots\Big)t + a_0\Big]$

- schéma séquentiel
- utilisation d'une seule des 4 voies
- latence = 29 cycles

Quel schéma utiliser pour évaluer $\varphi a(t)$ sur ST240 ?

Quelques caractéristiques

- VLIW 4 voies
- latences : 3 cycles/multiplication et 1 cycle/addition

Schéma de Horner $\varphi a(t) = \varphi\Big[\Big(\cdots\big((a_8 t + a_7)t + a_6\big)\ldots\Big)t + a_0\Big]$

- schéma séquentiel
- utilisation d'une seule des 4 voies
- latence = 29 cycles

Schéma rapide proposé

- coefficients positifs et valeurs intermédiaires positives
- incorporation de la multiplication par φ et ajout de 2^{-25} (utile pour l'arrondi)

$$\varphi a(t) = \Big(\big(\varphi(a_0+a_1 t)+2^{-25}\big)-\big((a_2-a_3 t)+(a_4-a_5 t)t^2\big)\varphi t^2\Big)-\Big(\big((a_6-a_7 t)+(a_8 t^2)\big)t^4\Big)\varphi t^2$$

Ordonnancement du schéma d'évaluation sur ST240

$$\varphi a(t) = \Big(\big(\varphi(a_0+a_1t)+2^{-25}\big) - \big((a_2-a_3t)+(a_4-a_5t)t^2\big)\varphi t^2 \Big) - \Big(\big((a_6-a_7t)+(a_8t^2)\big)t^4 \Big)\varphi t^2$$

	— Voie 1 —	— Voie 2 —	— Voie 3 —	— Voie 4 —
cycle 1	$t^2 = t \times t$	$a_5 \times t$		
cycle 2	$a_7 \times t$	$a_1 \times t$		
cycle 3	$a_3 \times t$			
cycle 4	$t^4 = t^2 \times t^2$	$a_{45} = a_4 - a_5t$		
cycle 5	$a_{45} \times t^2$	$a_8 \times t^2$	$a_{67} = a_6 - a_7t$	
cycle 6	$a_{01} = a_0 + a_1t$	$\varphi \times t^2$	$a_{68} = a_{67} + a_8t^2$	
cycle 7	$a_{23} = a_2 - a_3t$	$a_{68} \times t^4$		
cycle 8	$a_{01} \times \varphi$	$a_{25} = a_{23} + a_{45}t^2$		
cycle 9	$a_{25} \times \varphi t^2$			
cycle 10	$(a_{68}t^4) \times \varphi t^2$			
cycle 11	$a'_{01} = a_{01}\varphi + 2^{-25}$			
cycle 12	$a_{05} = a'_{01} - a_{25}\varphi t^2$			
cycle 13	$\varphi a(t) = a_{05} - (a_{68}t^4)\varphi t^2$			

- Ordonnancement obtenu = ordonnancement attendu
- Évaluation en 13 cycles (≈ 2.2 fois plus rapide que Horner)

Ordonnancement du schéma d'évaluation sur ST240

$$\varphi a(t) = \Big(\big(\varphi(a_0 + a_1 t) + 2^{-25} \big) - \big((a_2 - a_3 t) + (a_4 - a_5 t) t^2 \big) \varphi t^2 \Big) - \Big(\big((a_6 - a_7 t) + (a_8 t^2) \big) t^4 \Big) \varphi t^2$$

	— Voie 1 —	— Voie 2 —	— Voie 3 —	— Voie 4 —
cycle 1	$t^2 = t \times t$	$a_5 \times t$	-	
cycle 2	$a_7 \times t$	$a_1 \times t$	-	
cycle 3	$a_3 \times t$	-		
cycle 4	$t^4 = t^2 \times t^2$	$a_{45} = a_4 - a_5 t$	-	
cycle 5	$a_{45} \times t^2$	$a_8 \times t^2$	$a_{67} = a_6 - a_7 t$	
cycle 6	$a_{01} = a_0 + a_1 t$	$\varphi \times t^2$	$a_{68} = a_{67} + a_8 t^2$	-
cycle 7	$a_{23} = a_2 - a_3 t$	$a_{68} \times t^4$	-	
cycle 8	$a_{01} \times \varphi$	$a_{25} = a_{23} + a_{45} t^2$	-	-
cycle 9	$a_{25} \times \varphi t^2$	-	-	
cycle 10	$(a_{68} t^4) \times \varphi t^2$	-	-	
cycle 11	$a'_{01} = a_{01} \varphi + 2^{-25}$	-	-	
cycle 12	$a_{05} = a'_{01} - a_{25} \varphi t^2$	-	-	
cycle 13	$\varphi a(t) = a_{05} - (a_{68} t^4) \varphi t^2$	-	-	

- Ordonnancement obtenu = ordonnancement attendu
- Évaluation en 13 cycles (≈ 2.2 fois plus rapide que Horner)

Bornes d'erreur sur l'évaluation de $\varphi a(t) + 2^{-25}$

Exemple pour $\varphi = 1$

```
Results for x in [0, 0.999998]:

T in [0, 4294967295b-32 {1, 2^(0)}]
T2 in [0, 4294967295b-32 {1, 2^(0)}]
phiT2 in [0, 4294967295b-31 {2, 2^(1)}]
T4 in [0, 4294967295b-32 {1, 2^(0)}]

a01 in [0, 4294967295b-31 {2, 2^(1)}]
a01_p in [0, 4294967295b-30 {4, 2^(2)}]
a05 in [0, 4294967295b-30 {4, 2^(2)}]
a23 in [0, 4294967295b-31 {2, 2^(1)}]
a25 in [0, 4294967295b-31 {2, 2^(1)}]
a45 in [0, 4294967295b-31 {2, 2^(1)}]
a67 in [0, 4294967295b-31 {2, 2^(1)}]
a678 in [0, 4294967295b-31 {2, 2^(1)}]
a68 in [0, 4294967295b-30 {4, 2^(2)}]
e_round in [-206156857347b-67 {-1.39697e-09, -2^(-29.415)}, 1040500664211090463b-88 {3.36204e-09, 2^(-28.148)}]

a in [0, 4294967295b-31 {2, 2^(1)}]
```

$$\Rightarrow \; -2^{-27.5} < v - (\varphi a(t) + 2^{-25}) < 2^{-27.5}$$

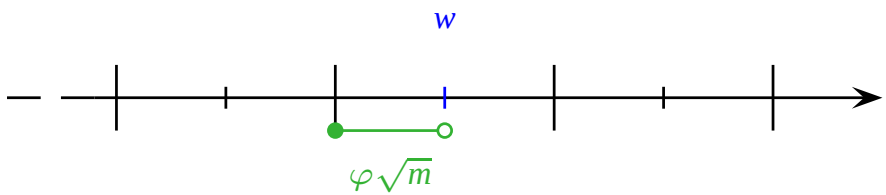
Arrondi correct au plus près

On connaît v telle que $0 \leq v - \varphi\sqrt{m} \leq 2^{-24}$.

- $v = 01.v_1v_2v_3 \ldots v_{29}v_{30}$
- $w = v$ tronqué sur 24 bits $\Rightarrow w = 01.w_1w_2w_3 \ldots w_{24}000000$

$$-2^{-24} \leq w - \varphi\sqrt{m} \leq 2^{-24}$$

- Si $w \otimes w \geq \varphi^2 m$ alors $w \geq \varphi\sqrt{m}$
  $\rightarrow$ $\varphi\sqrt{m}$ ne peut être pas le milieu de deux nombres flottants



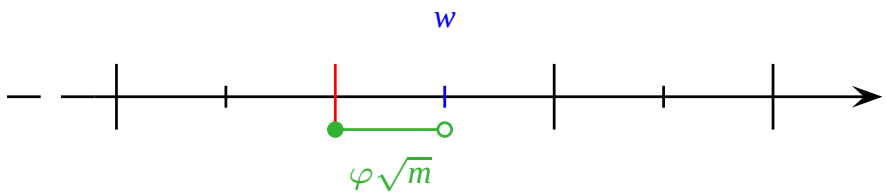
Arrondi correct au plus près

On connaît v telle que $0 \leq v - \varphi\sqrt{m} \leq 2^{-24}$.

- $v = 01.v_1v_2v_3 \ldots v_{29}v_{30}$
- $w = v$ tronqué sur 24 bits $\Rightarrow w = 01.w_1w_2w_3 \ldots w_{24}000000$

$$-2^{-24} \leq w - \varphi\sqrt{m} \leq 2^{-24}$$

- Si $w \otimes w \geq \varphi^2 m$ alors $w \geq \varphi\sqrt{m}$
  - $\rightarrow$ $\varphi\sqrt{m}$ ne peut être pas le milieu de deux nombres flottants



  - $\rightarrow$ $\circ(\varphi\sqrt{m}) = w$ tronqué sur 23 bits

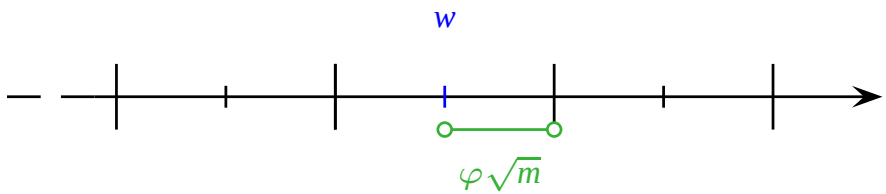
Arrondi correct au plus près

On connaît v telle que $0 \leq v - \varphi\sqrt{m} \leq 2^{-24}$.

- $v = 01.v_1v_2v_3 \ldots v_{29}v_{30}$
- $w = v$ tronqué sur 24 bits $\Rightarrow w = 01.w_1w_2w_3 \ldots w_{24}000000$

$$-2^{-24} \leq w - \varphi\sqrt{m} \leq 2^{-24}$$

- Si $w \otimes w < \varphi^2 m$ alors $w < \varphi\sqrt{m}$



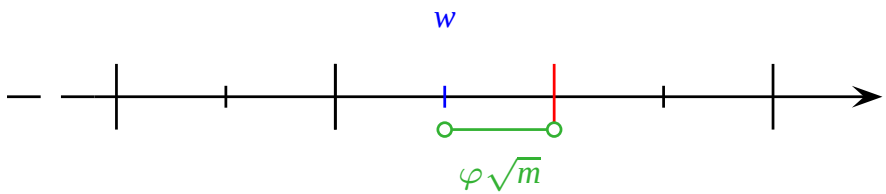
Arrondi correct au plus près

On connaît v telle que $0 \leq v - \varphi\sqrt{m} \leq 2^{-24}$.

- $v = 01.v_1v_2v_3 \ldots v_{29}v_{30}$
- $w = v$ tronqué sur 24 bits $\Rightarrow w = 01.w_1w_2w_3 \ldots w_{24}000000$

$$-2^{-24} \leq w - \varphi\sqrt{m} \leq 2^{-24}$$

- Si $w \otimes w < \varphi^2 m$ alors $w < \varphi\sqrt{m}$



$\rightarrow$ $\circ(\varphi\sqrt{m}) = w + 2^{-24}$ tronqué sur 23 bits

Résumé de l'approche

Comment implanter une fonction ?

- Calculer les coefficients d'un bon polynôme d'approximation
  - → quelle fonction sur quel intervalle ? quel degré ? quelle structure de coefficients ?
- Proposer un code d'évaluation de ce polynôme en virgule fixe
  - → quel degré de parallélisme ? quelles latences ?
  - → quel schéma d'évaluation ?
  - → analyse numérique du schéma d'évaluation avec Gappa

Résumé de l'approche

Comment implanter une fonction ?

- Calculer les coefficients d'un bon polynôme d'approximation
  - → quelle fonction sur quel intervalle ? quel degré ? quelle structure de coefficients ?
- Proposer un code d'évaluation de ce polynôme en virgule fixe
  - → quel degré de parallélisme ? quelles latences ?
  - → quel schéma d'évaluation ?
  - → analyse numérique du schéma d'évaluation avec Gappa

BESOIN D'AUTOMATISATION

Performances sur ST240

- Pour toute entrée x (valeurs spéciales comprises) :

	RN	RU	RD/RZ	RF
Sans dénormaux	22	22	22	20
Avec dénormaux	25	25	25	22

TAB.: Timings (cycles) sur ST240, suivant différents modes d'arrondi

- Accélération $\approx 55\%$ par rapport à la version précédente
- Même latence pour les quatre modes d'arrondi
- Surcoût dû à la prise en compte des nombres dénormalisés = 3 cycles (2 cycles en théorie)

Performances sur ST240

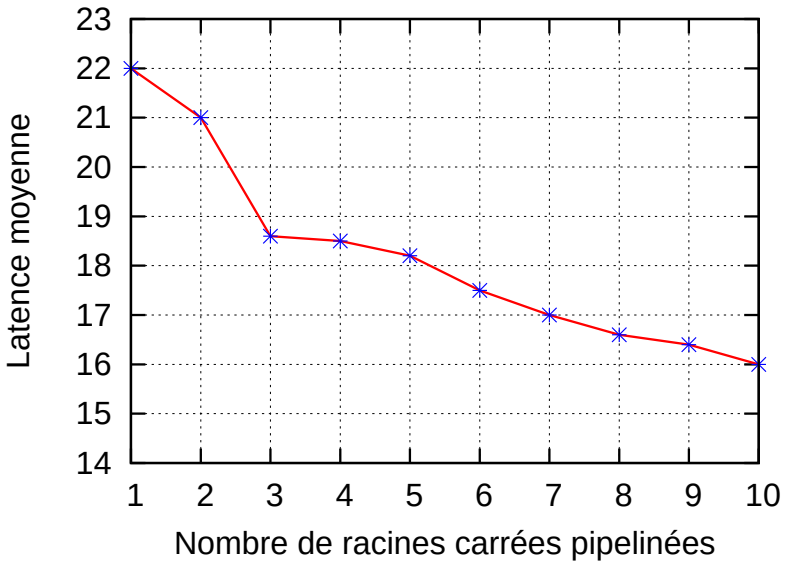
- Pour toute entrée x (valeurs spéciales comprises) :

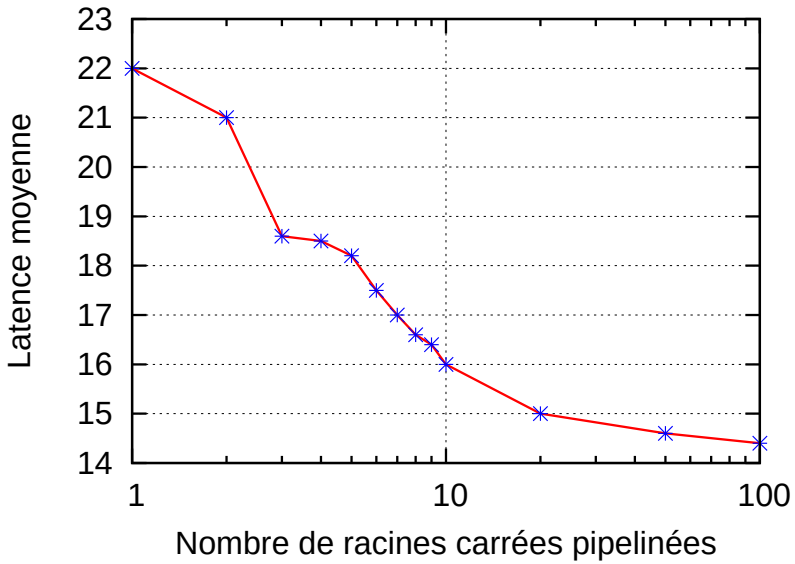
	RN	RU	RD/RZ	RF
Sans dénormaux	22	22	22	20
Avec dénormaux	25	25	25	22

TAB.: Timings (cycles) sur ST240, suivant différents modes d'arrondi

- Accélération $\approx 55\%$ par rapport à la version précédente
- Même latence pour les quatre modes d'arrondi
- Surcoût dû à la prise en compte des nombres dénormalisés = 3 cycles (2 cycles en théorie)
- Méthode facilement adaptable :
  - à l'arrondi fidèle (RF), mais polynôme *trop précis*
  - à d'autres formats (*medium/high precision*)

Exécutions pipelinées sur ST240





$\Rightarrow$ latence moyenne ≈ 14 cycles

Extension à d'autres fonctions...

- Racine carrée inverse : $x^{-\frac{1}{2}}$

	RN	RU	RD/RZ
Sans dénormaux	30	31	31
Avec dénormaux	32	32	32

TAB.: Timings (cycles) sur ST240, suivant différents modes d'arrondi

$\rightarrow$ accélération de 50%

- Racine quatrième inverse : $x^{-\frac{1}{4}}$

$\rightarrow$ composition de $\sqrt{x}$ et $x^{-\frac{1}{2}} = 52$ cycles

$\rightarrow$ notre méthode : 38 cycles en arrondi au plus près

Extension à d'autres fonctions...

- Racine carrée inverse : $x^{-\frac{1}{2}}$

	RN	RU	RD/RZ
Sans dénormaux	30	31	31
Avec dénormaux	32	32	32

TAB.: Timings (cycles) sur ST240, suivant différents modes d'arrondi

$\rightarrow$ accélération de 50%

- Racine quatrième inverse : $x^{-\frac{1}{4}}$
  - $\rightarrow$ composition de $\sqrt{x}$ et $x^{-\frac{1}{2}} = 52$ cycles
  - $\rightarrow$ notre méthode : 38 cycles en arrondi au plus près

- Inverse de norme euclidienne : $1/\sqrt{x^2+y^2}$
  - $\rightarrow$ problématique : approximations et schémas d'évaluation bivariés

Conclusions

- Racine carrée simple précision correctement arrondie
- Implantation à base d'évaluation polynomiale
- Efficace sur ST240 : accélération de plus de $\approx 55\%$

- Efficacité de cette méthode pour l'implantation d'autres fonctions algébriques : racine carrée inverse, racine quatrième inverse...
- Extension aux autres fonctions de FLIP (inverse, division, ...)

- Besoin d'automatiser la conception :
  - $\longrightarrow$ des schémas d'évaluation et des procédures d'arrondi,
  - $\longrightarrow$ et plus généralement de fonctions.