23rd IEEE Symposium on Computer Arithmetic
ARITH 23
Silicon Valley, USA. July 10–13, 2016

Digit Recurrence Floating-point Division under HUB Format

Prof. Dr. Julio Villalba-Moreno
Dept. Computer Architecture
University of Malaga
SPAIN

ANDALUCIA TECH
Campus de Excelencia Internacional

Talk Outline

- Introduction
- HUB format
  - Definition
  - Floating point HUB numbers
  - Advantages and drawbacks
- Division under HUB
  - Digit recurrence algorithm
  - Data path bit-width and number of iterations
  - On-the-fly conversion
  - Unbiased round to nearest
- Summary and conclusion

HUB format

HUB format

- **HUB = <u>H</u>alf-<u>U</u>nit <u>B</u>iased format**

Digit-vector

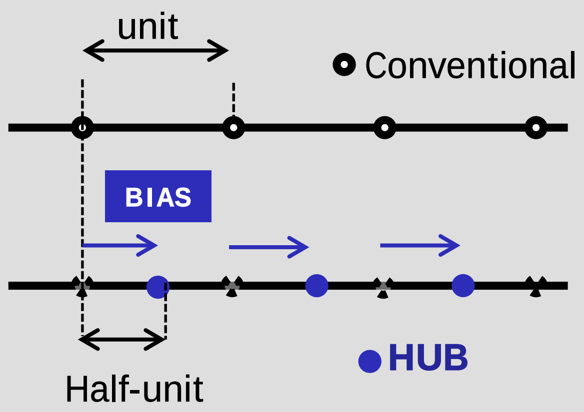
$$X = (X_{n-1}, X_{n-2}, \cdots X_1, X_0, X_{-1}, \cdots, X_{-f})$$

Conventional number in radix β

$$X = \left[\sum_{i=-f}^{n-1} X_i \cdot \beta^i\right]$$

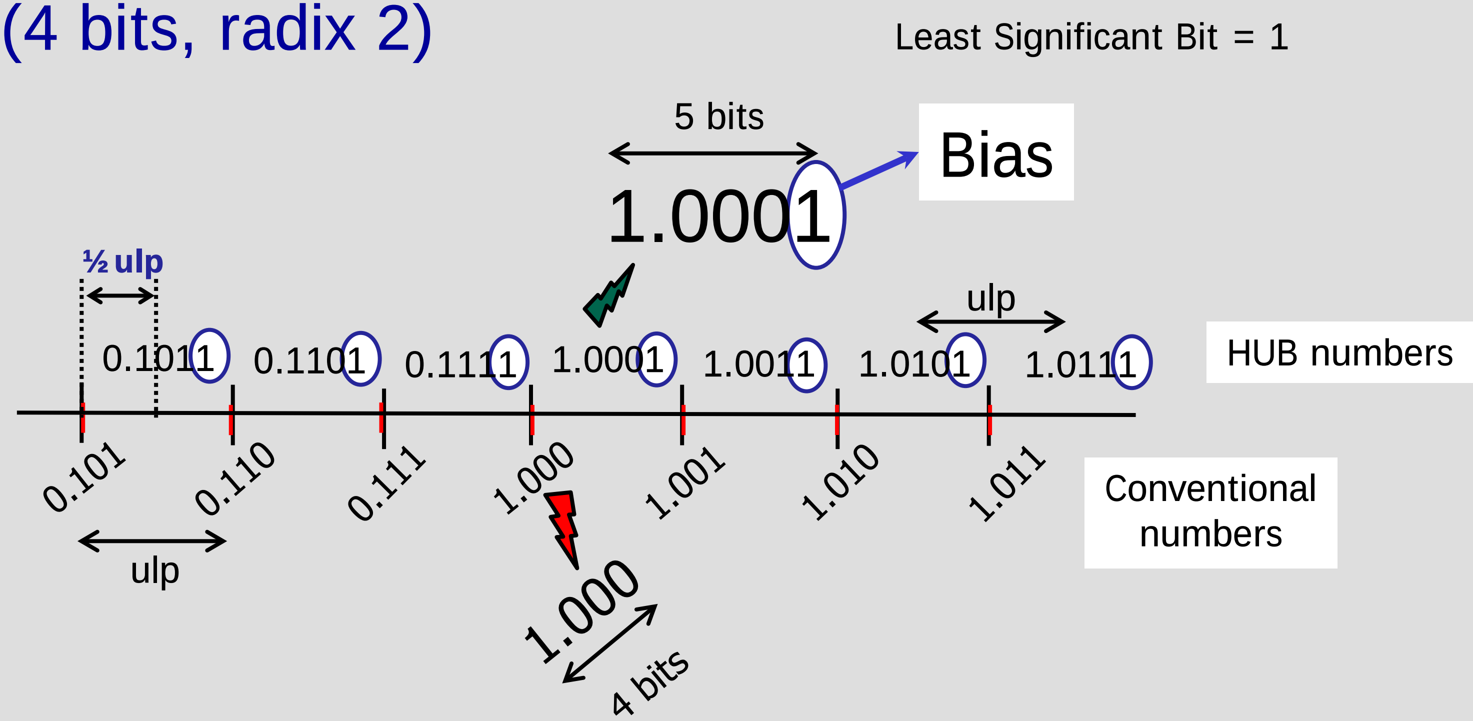
HUB number in radix β

$$X = \left[\sum_{i=-f}^{n-1} X_i \cdot \beta^i\right] + \underbrace{\frac{\beta}{2} \cdot \beta^{-f-1}}_{\text{BIAS}}$$



HUB format

- Example (4 bits, radix 2)



HUB format

- The extra LSB → ILSB Implicit Least Significant Bit

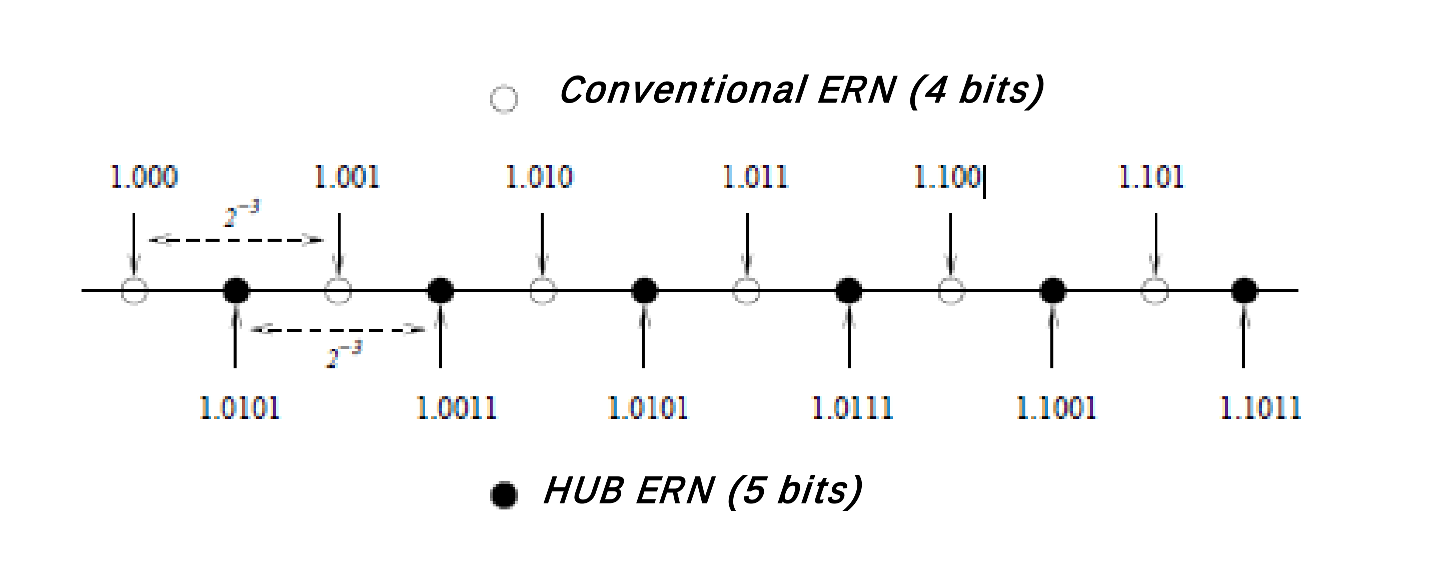
x x x . x x x x x x x x x x x x 1

ILSB

  - Value: 1
    - ➢ Implicit bit (ILSB)
      - Not stored
      - Not transmitted
      - Not needed for representation
    - ➢ Required when operating

HUB format

- Exactly representable numbers (ERN)



  - The set of ERN is different for both representations
  - Set(HUB) ∩ Set(Conventional) = Ø (Disjoint sets)
  - Disantance between two consequtive numbers is the same
  - The amount of ERN is the same
    - Both representations have the same precision

HUB format

- Floating point HUB number in radix-2 *(β=2)*

$$(S_x, M_x, E_x)$$

$$x = (-1)^{S_x} M_x 2^{E_x}$$

E_x → Exponen (conventional)

M_x → Significand (HUB magnitude)

S_x → Sign (conventional)

– Normalized HUB significand: $1 < M_x < 2$

Digit-vector

$$M_x = (M_{x_0}, M_{x_{-1}}, M_{x_{-2}}, \cdots, M_{x_{-f}})$$

$$M_x = \left[\sum_{i=0}^{f} M_{x_i} \cdot 2^{-i}\right] + \underbrace{2^{-f-1}}_{\text{BIAS}}$$

$$M_x = 1.M_{x_{-1}} M_{x_{-2}} \cdots M_{x_{-f}} 1$$

HUB format

- Normalized Floating point HUB number in radix-2

$$M_x = \left[\sum_{i=0}^{f} M_{x_i} \cdot 2^{-i}\right] + 2^{-f-1}$$

ILSB

$$M_x = 1.M_{x_{-1}} M_{x_{-2}} \cdots M_{x_{-f}}$$

Representative form

Digit-vector

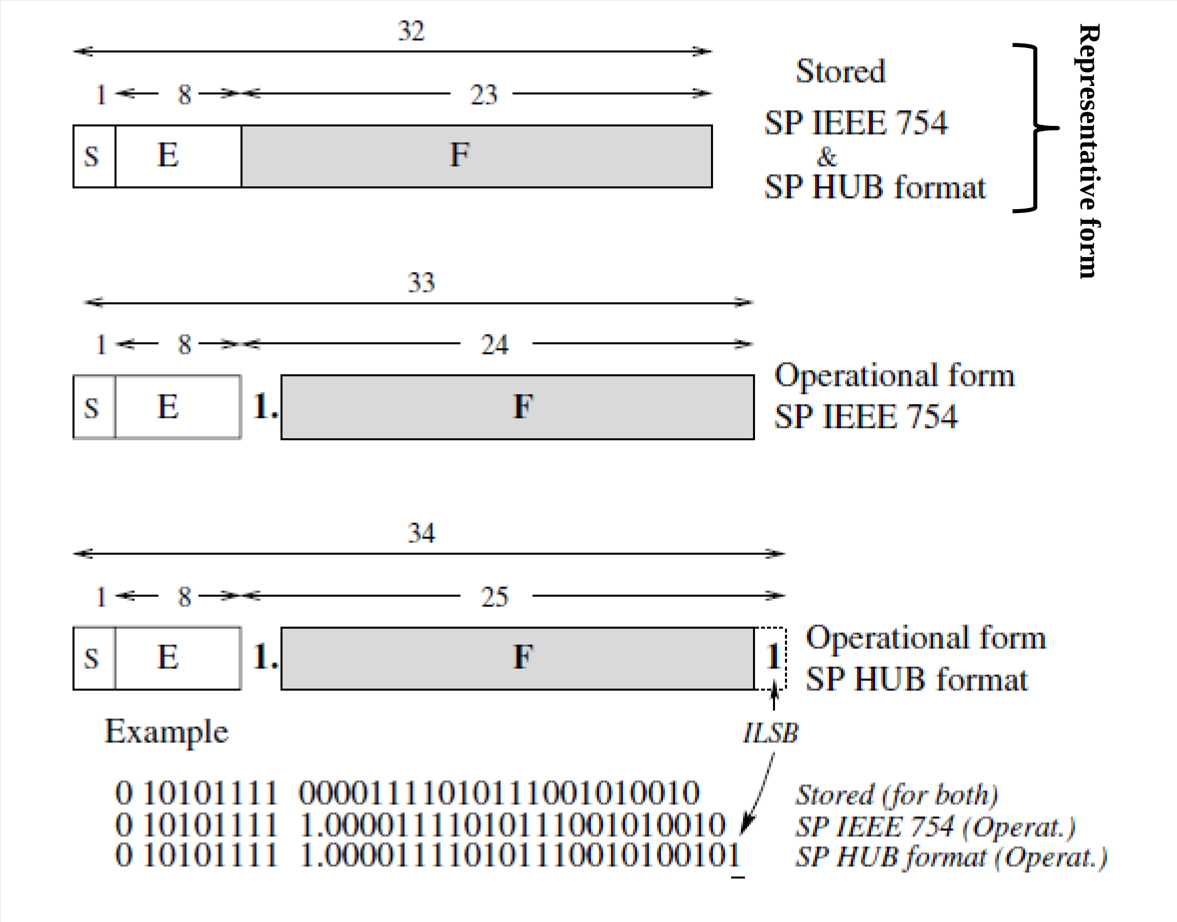
$$M_x = 1.M_{x_{-1}} M_{x_{-2}} \cdots M_{x_{-f}} 1$$

Operational form

To operate

HUB format

Single precision (SP) IEEE-754 and its HUB counterpart



HUB format

- Advantages
  - Two's complement → bit-wise
  - Round to nearest → by truncation
  - No double rounding error
  - Simplicity
- Drawbacks
  - Not valid for integers
  - Other rounding modes require carry propagation
  - Not IEEE compliance

HUB format

- Two's complement of a HUB number
  - Invert the bits of the representative form (bit-wise)
    - » The ILSB=1 → No carry propagation

HUB number (operational form)

01.010 0111

Always 1

Bit-wise Inversion

10.101 1000

Always 0

+1

10.101 1001

Always 1, No carry propagation

HUB number (representative form)

01.010 011

Bit-wise Inversion

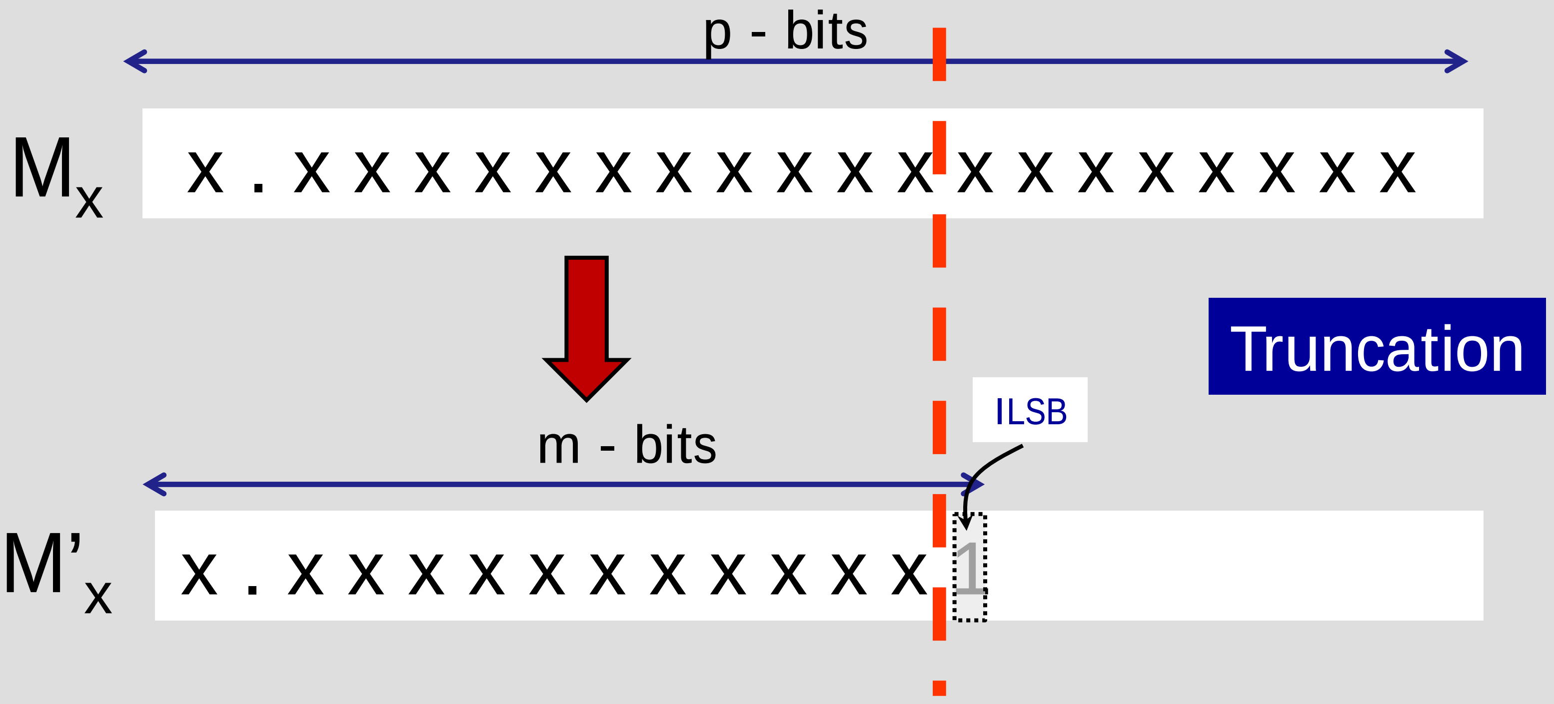
10.101 100

10.101 1001

ILSB

HUB format

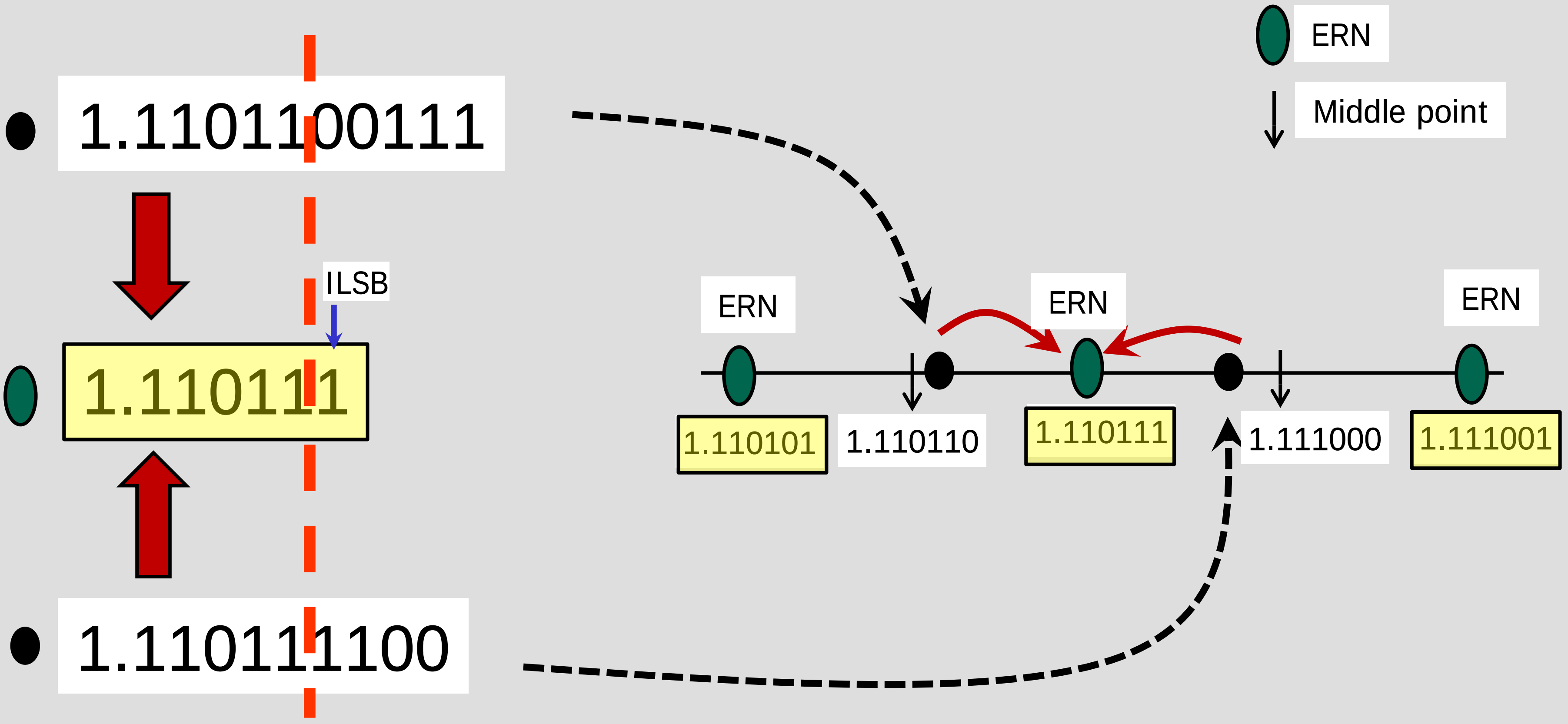
- Round to nearest of a HUB number: by truncation



Formally: $M'[0:m-2] = M[0:m-2]$

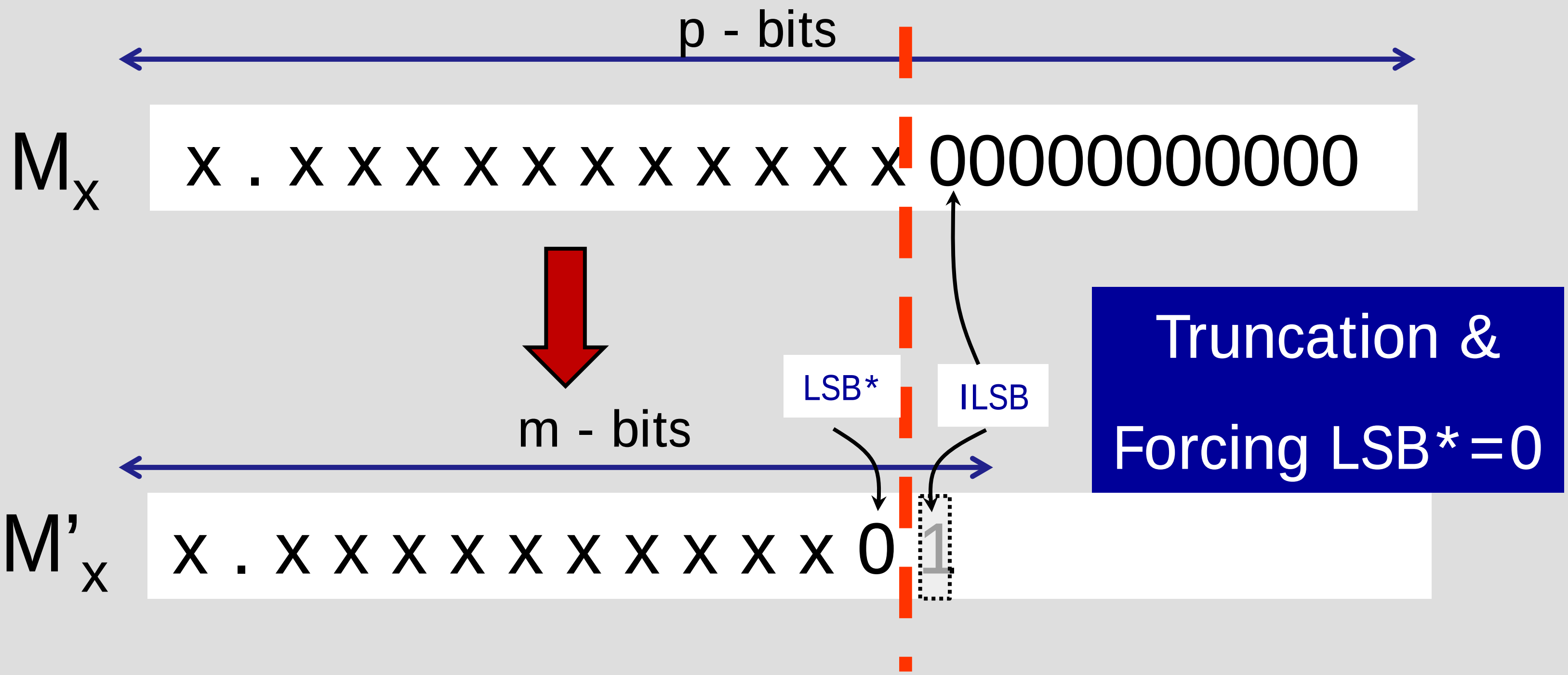
HUB format

- Proposed rounding: round to nearest by truncation



HUB format

- The tie case *(0 for all bits starting at ILSB)*



Formally: M’[0:m-2]=M[0:m-3,0]

LSB* = LSB of the representative form

HUB format

- The tie case, example

Tie case

M_x 1. 0 1 0 1 1 0 0 0 0 0

LSB* ILSB

M_x 1. 0 1 0 1 <u>0</u> 1

Effective rounding down

Tie case

M_x 1. 0 1 0 1 0 0 0 0 0 0

LSB* ILSB

M_x 1. 0 1 0 1 <u>0</u> 1

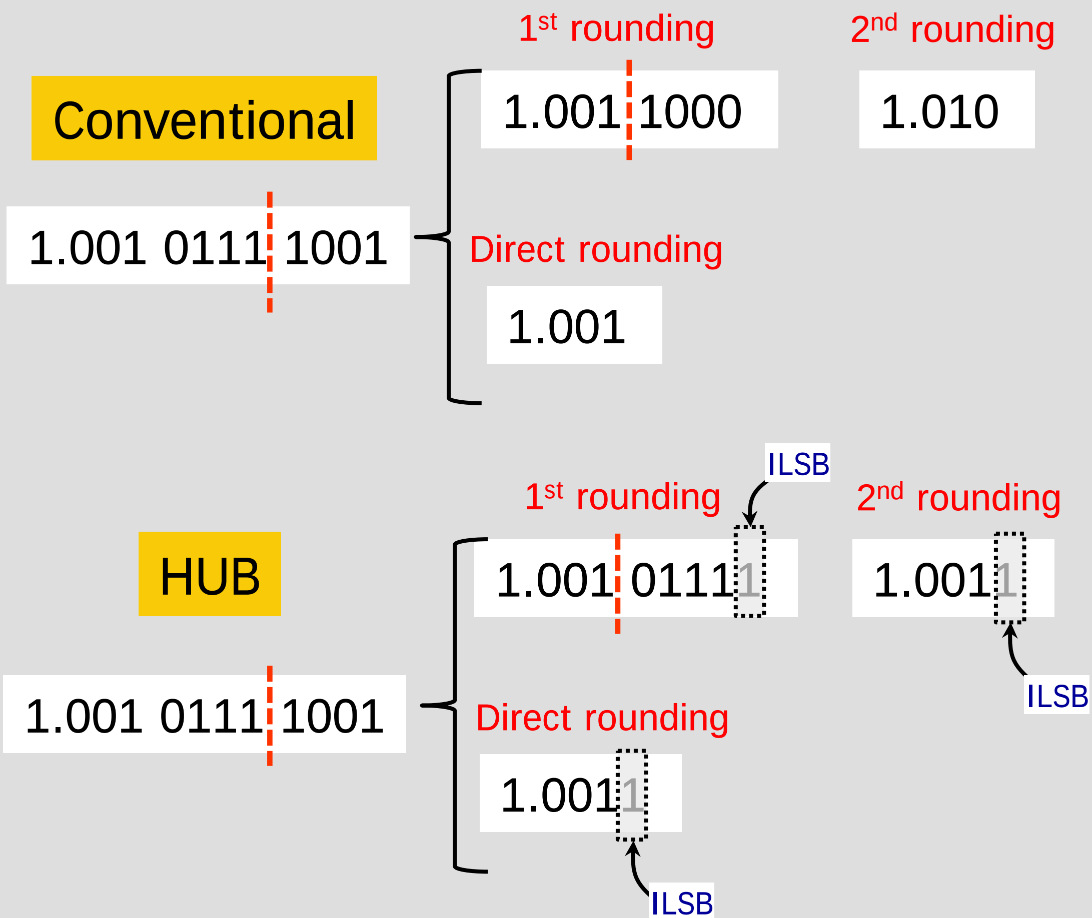
Effective rounding up

Truncation &

Forcing LSB*=0

HUB format

- No double rounding error for HUB numbers
  - Truncation avoids the double rounding error



HUB format

- Efficiency of HUB format
  - Fixed-point
    - [6]: Optimization of FIR filters
    - [7]: QR decomposition
  - Floating-point
    - [9]: High dynamic range image and video systems
    - [8]: Quantitative analysis of HUB for floating point adders, multipliers and conveters

Floating point [5,8]

- Efficiency of HUB format
  - Floating-point
    - Adder:
      - » Speed-up: 14%
      - » Area reduction: 38%
      - » Power reduction: 25% (single), 15% (double)
    - Multiplier
      - » Speedup: 17%
      - » Area reduction: 22%
      - » Power reduction: 2% (single), -2.6% (double)

[8] *Measuring the improvement when using HUB formats to implement floating point systems under round to nearest*, IEEE Transactions on VLSI 2015, DOI: 10.1109/TVLSI.2015.2502318

Floating point [5,8]

- Efficiency of HUB format
  - Floating-point
    - Division:
      - Not studied yet

Objective: extend the HUB format to

Floating-point Division

Floating-point division under HUB

Floating-point division for HUB

– Division of two FP HUB numbers

HUB number

x → (S_x, M_x, E_x)

d → (S_d, M_d, E_d)

HUB number

$$q = \frac{x}{q}$$

HUB number

q → (S_q, M_q, E_q)

M → magnitud & normalized

Floating-point division for HUB

– Digit recurrence algorithm [10]

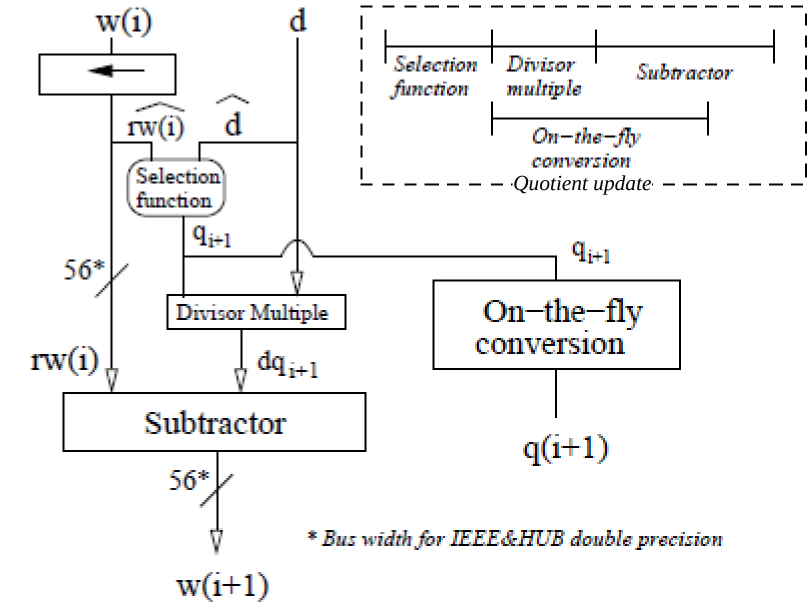
$$q(i) = q(0) + \sum_{j=1}^{i} q_j r^{-j}$$

$q_i \in [-a, a],$
$a \geq \lceil r/2 \rceil$

$$\rho = \frac{a}{r-1}, \qquad \frac{1}{2} < \rho \leq 1$$

Residual $$w(i) = r^i (x - dq(i))$$

Bound $$|w(i)| \leq \rho \cdot d$$



Recurrence $$\boxed{w(i+1) = rw(i) - dq_{i+1}}$$

Selection function

$$q_{i+1} = \begin{cases} 1 & \text{if} \quad 0 \leq \widehat{rw(i)} \leq 3/2 \\ 0 & \text{if} \quad \widehat{rw(i)} = 3/2 \\ -1 & \text{if} \quad -5/2 \leq \widehat{rw(i)} \leq -1 \end{cases}$$

Floating-point division for HUB

– Digit recurrence algorithm for HUB

M_x = 1.xxx…xxx1 $\quad 1 < M_x < 2$

M_d = 1.ddd…ddd1 $\quad 1 < M_d < 2$

$\Rightarrow$ $M_q \in (½, 2)$ $\Rightarrow$ Normalization required

– Initialization step

  – w(0) = x/2 or w(0)=x-d if ρ =1 (maximum redundancy)
  – w(0) = x/4 if ρ <1

– Termination step

  – Correction of the initialization (1 or 2 positions left shift)
  – Correction if final negative residue
  – Normalization if quotient ϵ (½<1)
  – **Rounding-to-nearest: by truncation / adding 1 ulp (conventional)**
  – Check zero condition if exact quotient is needed

Floating-point division for HUB

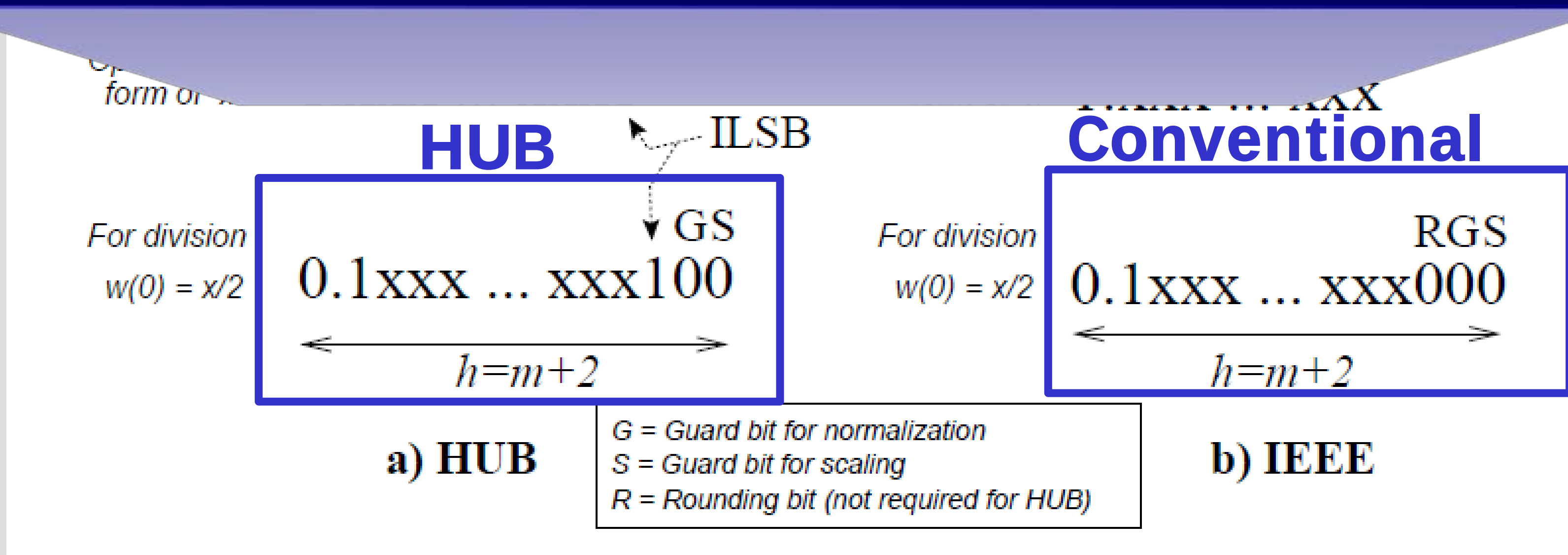
- Data path bit-width (h)
  - m (operational form of HUB number) plus
    - » 1 guard bits due to normalization (bit ***G***)
    - » 1 bit if ρ =1 or 2 bits if ρ <1 (bit ***S***, scaling of the intitilization step)

$$h = m + 1 + (2 - \lfloor \rho \rfloor) = m + 3 - \lfloor \rho \rfloor$$

Floating-point division for HUB

– Data path bit-width (h) $h = m + 1 + (2 - \lfloor \rho \rfloor) = m + 3 - \lfloor \rho \rfloor$

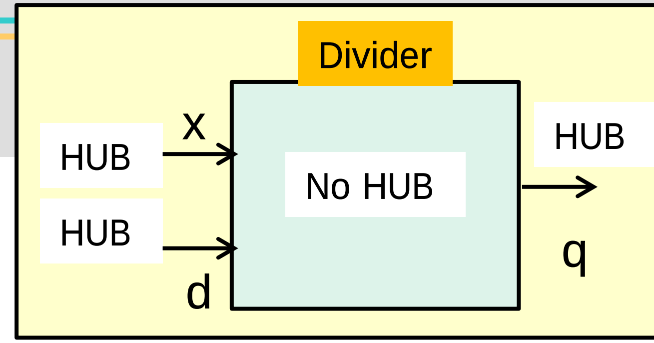
Conclusion: same data path bit-width for HUB and for its conventional counterpart



Example for ρ=1

Floating-point division for HUB

– Operation inside the divisor



d= 1. xxx ... xxx1 HUB
x= 1.xxx ... xxx1 HUB

RECURRENCE			QUOTIENT	
w(0)= 0.1xxx ... xxx100	*(no HUB)*	→	q_1	*(no HUB)*
w(1)= x.xxxx ... xxxxxx	*(no HUB)*	→	$q_1 q_2$	*(no HUB)*
w(2)= x.xxxx ... xxxxxx	*(no HUB)*	→	$q_1 q_2 q_3$	*(no HUB)*
. . . . . .			. . .	
w(N)= x.xxxx ... xxxxxx	*(no HUB)*	→	$q_1 q_2 q_3 \cdots q_N$	*(no HUB)*
w(N)= x.xxxx ... xxx1	*(HUB op.)*		M_q= 1.xxx ... xxx1	*(HUB, op.)*
x.xxxx ... xxx	*(HUB rep.)*		1.xxx ... xxx	*(HUB, rep.)*

Quotient (significand) after correction, normalization and rounding by truncation

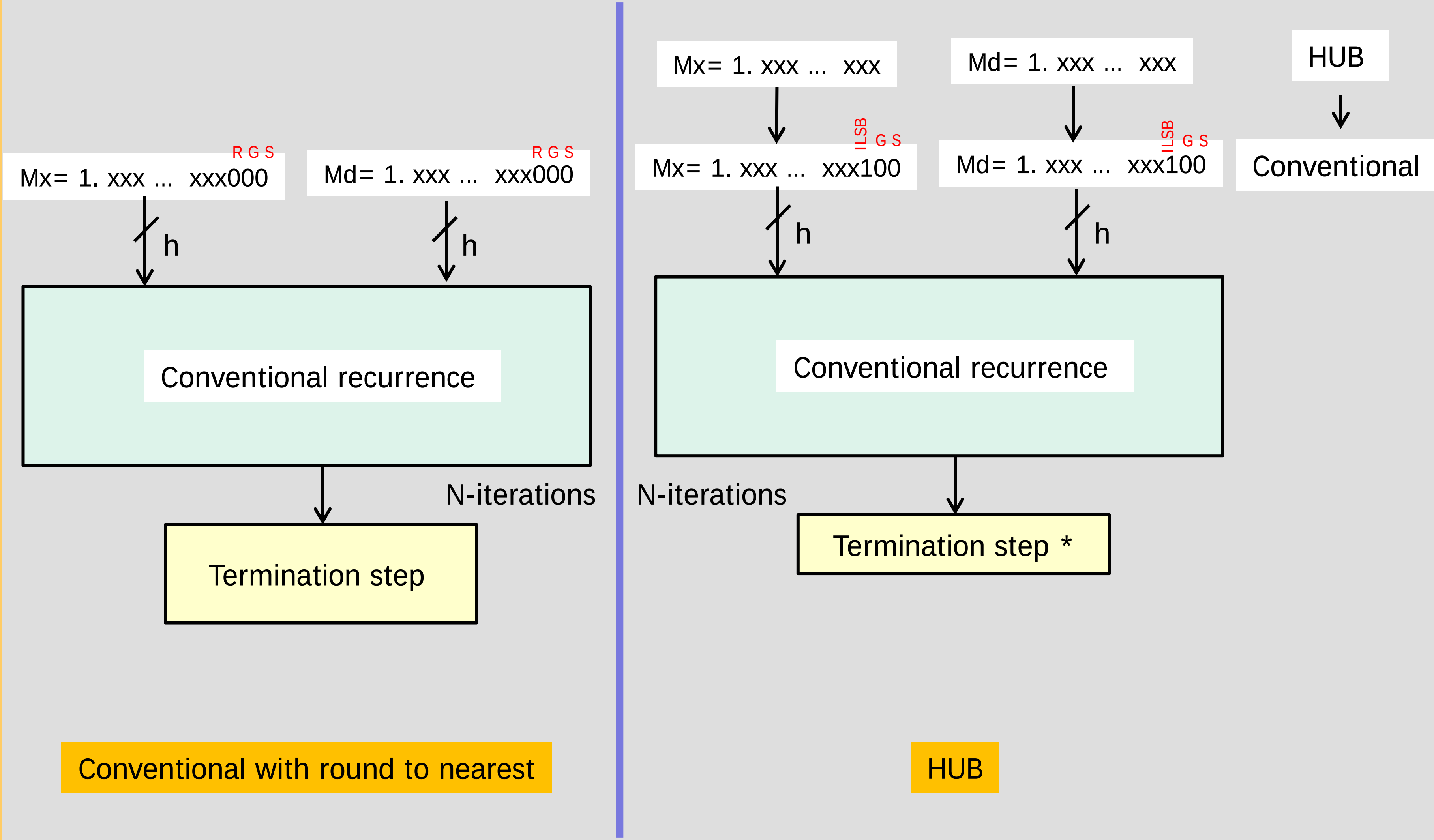
Floating-point division for HUB

- Number of iterations
  - Depends on the number of bits to be computed by the iterations (**h**) and the radix (**r**)

$$N = \left\lceil \frac{h}{\log_2 r} \right\rceil$$

Conclusion: same number of iterations for HUB and for its conventional counterpart

Floating-point division for HUB



On-the-fly conversion and rounding

- On the fly conversion
  - Overlapping the final conversion from redundant to conventional with the iterations
  - Rounding, correction and normalization is integrated in the on-the-fly conversion of the last digit

On-the-fly conversion and rounding

– On the fly conversion [10,12]

Q(i) → *i* MSD of the converted quotient (digit vector)

$$Q(i) = \sum_{j=1}^{i} q_i r^{-i}$$

To avoid carry propagation:

$$QD(i) = Q(i) - r^{-i}$$

On-the-fly algorithm: Concatenation

$$Q(i+1) = \begin{cases} (Q(i) \parallel q_{i+1}) & \text{if } q_{i+1} \geq 0 \\ (QD(i) \parallel (r - |q_{i+1}|)) & \text{if } q_{i+1} < 0 \end{cases}$$

$$QD(i+1) = \begin{cases} (Q(i) \parallel q_{i+1} - 1) & \text{if } q_{i+1} > 0 \\ (QD(i) \parallel (r - 1 - |q_{i+1}|)) & \text{if } q_{i+1} \leq 0 \end{cases}$$

On-the-fly conversion and rounding

– On the fly conversion [10,12]

  – Round to nearest → addition of 1 ulp after regular iterations

    » Last digit can be out of range of the digit set [-a,a] if $a \geq r-2$

    » $q_N = a \rightarrow q_N+1 \notin [-a,a]$

  – A new form *QR(i)* is defined to combine the correction, normalization and rounding: $QR(i) = Q(i) + r^{-i}$

$$QR(i+1) = \begin{cases} (QR(i) \parallel 0) & \text{if } q_{i+1} = r-1 \\ (Q(i) \parallel q_{i+1}+1) & \text{if } -1 \leq q_{i+1} \leq r-2 \\ (QD(i) \parallel (r+1-|q_{i+1}|)) & \text{if } q_{i+1} < -1 \end{cases}$$

On-the-fly conversion and rounding

– On the fly conversion [10,12]

Rounded signficand before normalization and truncation

$$MMq = \begin{cases} (QR(N-1) \parallel u) & \text{if } q_N^* \geq r \\ (Q(N-1) \parallel u) & \text{if } 0 \leq q_N^* \leq r-1 \\ (QD(N-1) \parallel u) & \text{if } q_N^* < r \end{cases}$$

$$q_N^* \in \{-a, a+1\}$$

$$u = q_N^* \, mod \; r$$

Final nomalized signficand:

$$M_q[0:m-2] = \begin{cases} MMq[0:m-2] & \text{if } MMq[0] = 1 \\ MMq[1:m-1] & \text{if } MMq[0] = 0 \end{cases}$$

On-the-fly conversion and rounding

– On the fly conversion for HUB

$$Q(i+1) = \begin{cases} (Q(i) \parallel q_{i+1}) & \text{if } q_{i+1} \geq 0 \\ (QD(i) \parallel (r - |q_{i+1}|)) & \text{if } q_{i+1} < 0 \end{cases}$$

$$QD(i+1) = \begin{cases} (Q(i) \parallel q_{i+1} - 1) & \text{if } q_{i+1} > 0 \\ (QD(i) \parallel (r - 1 - |q_{i+1}|)) & \text{if } q_{i+1} \leq 0 \end{cases}$$

(Same as [10,12])

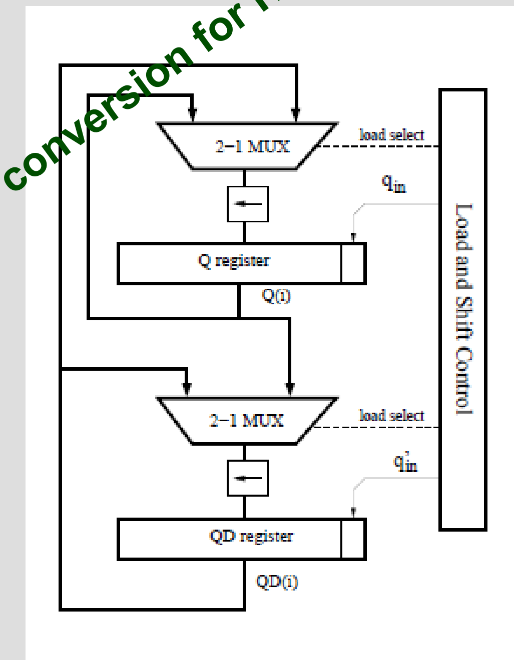
- – Round to nearest is carried out by truncation → No carry propagation → no addition of 1 ulp after regular iterations
  - » Last digit is always inside the range of the digit set [-a,a]
- – The form *QR(i)* is not required any more

$$QR(i) = Q(i) + r^{-i}$$

$$QR(i+1) = \begin{cases} (QR(i) \parallel 0) & \text{if } q_{i+1} = r-1 \\ (Q(i) \parallel q_{i+1}+1) & \text{if } -1 \leq q_{i+1} \leq r-2 \\ (QD(i) \parallel (r+1-|q_{i+1}|)) & \text{if } q_{i+1} < -1 \end{cases}$$

On-the-fly conversion and rounding

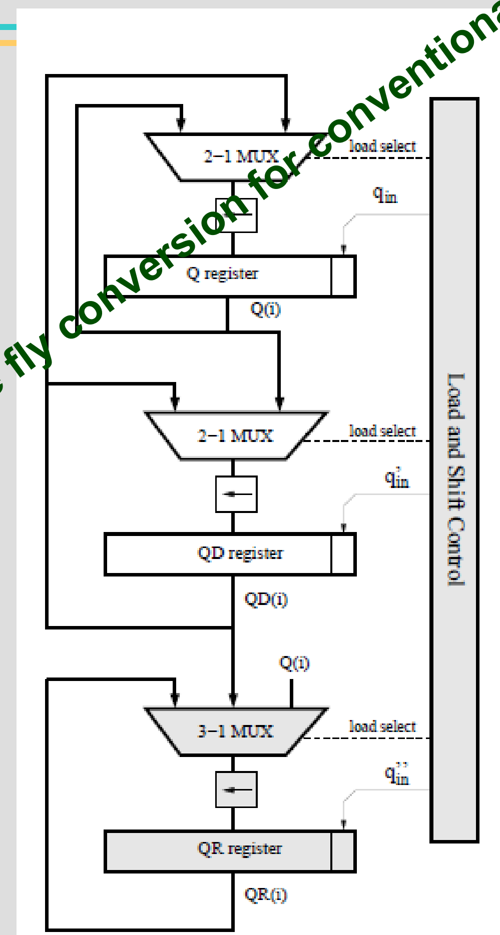
On the fly conversion for HUB



$$Q(i+1) = \begin{cases} (Q(i) \parallel q_{i+1}) & \text{if } q_{i+1} \geq 0 \\ (QD(i) \parallel (r - |q_{i+1}|)) & \text{if } q_{i+1} < 0 \end{cases}$$

$$QD(i+1) = \begin{cases} (Q(i) \parallel q_{i+1} - 1) & \text{if } q_{i+1} > 0 \\ (QD(i) \parallel (r - 1 - |q_{i+1}|)) & \text{if } q_{i+1} \leq 0 \end{cases}$$

On the fly conversion for conventional



$$Q(i+1) = \begin{cases} (Q(i) \parallel q_{i+1}) & \text{if } q_{i+1} \geq 0 \\ (QD(i) \parallel (r - |q_{i+1}|)) & \text{if } q_{i+1} < 0 \end{cases}$$

$$QD(i+1) = \begin{cases} (Q(i) \parallel q_{i+1} - 1) & \text{if } q_{i+1} > 0 \\ (QD(i) \parallel (r - 1 - |q_{i+1}|)) & \text{if } q_{i+1} \leq 0 \end{cases}$$

$$QR(i+1) = \begin{cases} (QR(i) \parallel 0) & \text{if } q_{i+1} = r - 1 \\ (Q(i) \parallel q_{i+1} + 1) & \text{if } -1 \leq q_{i+1} \leq r - 2 \\ (QD(i) \parallel (r + 1 - |q_{i+1}|)) & \text{if } q_{i+1} < -1 \end{cases}$$

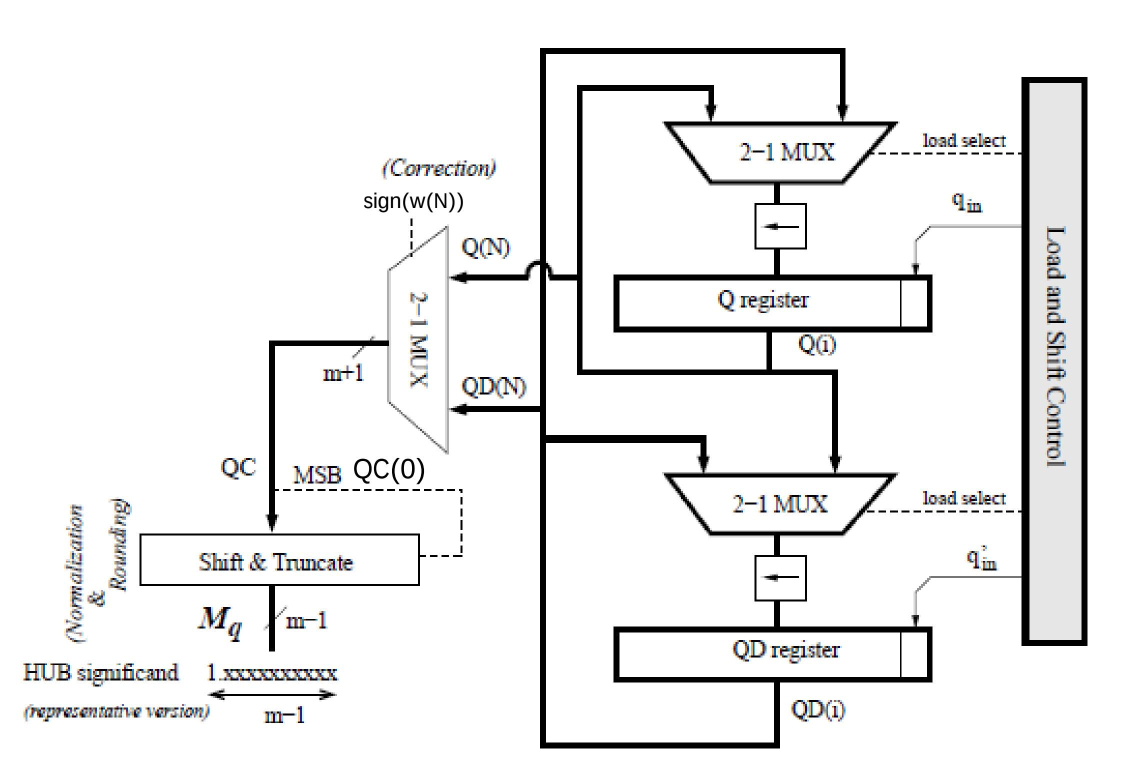
On-the-fly conversion and rounding

– On the fly conversion for HUB

We define QC as a function

of the sign of w(N):

$$QC = \begin{cases} Q(N) & \text{if } sign = 0 \\ QD(N) & \text{if } sign = 1 \end{cases}$$



Final normalizaed and round to nearest

HUB quotient (representative form):

$$M_q[0:m-2] = \begin{cases} QC[0:m-2] & \text{if } QC[0]=1 \\ QC[1:m-1] & \text{if } QC[0]=0 \end{cases}$$

On-the-fly conversion and rounding

– Optimizing the on-the-fly conversion

  – QC can be obtained from *Q(N-1)* and *QD(N-1)*

Last cycle:

$$QC = \begin{cases} Q(N) & \text{if } sign = 0 \\ QD(N) & \text{if } sign = 1 \end{cases}$$

$$Q(N) = \begin{cases} (Q(N-1) \parallel q_N) & \text{if } q_N \geq 0 \\ (QD(N-1) \parallel (r - |q_{i+1}|)) & \text{if } q_N < 0 \end{cases}$$

$$QD(N) = \begin{cases} (Q(N-1) \parallel q_N - 1) & \text{if } q_N > 0 \\ (QD(N-1) \parallel (r - 1 - |q_N|)) & \text{if } q_N \leq 0 \end{cases}$$

$$QC = \begin{cases} (Q(N-1) \parallel q_N - sign) & \text{if } q_N \geq 0 \\ (QD(N-1) \parallel r - |q_N| - sign) & \text{if } q_N < 0 \end{cases}$$

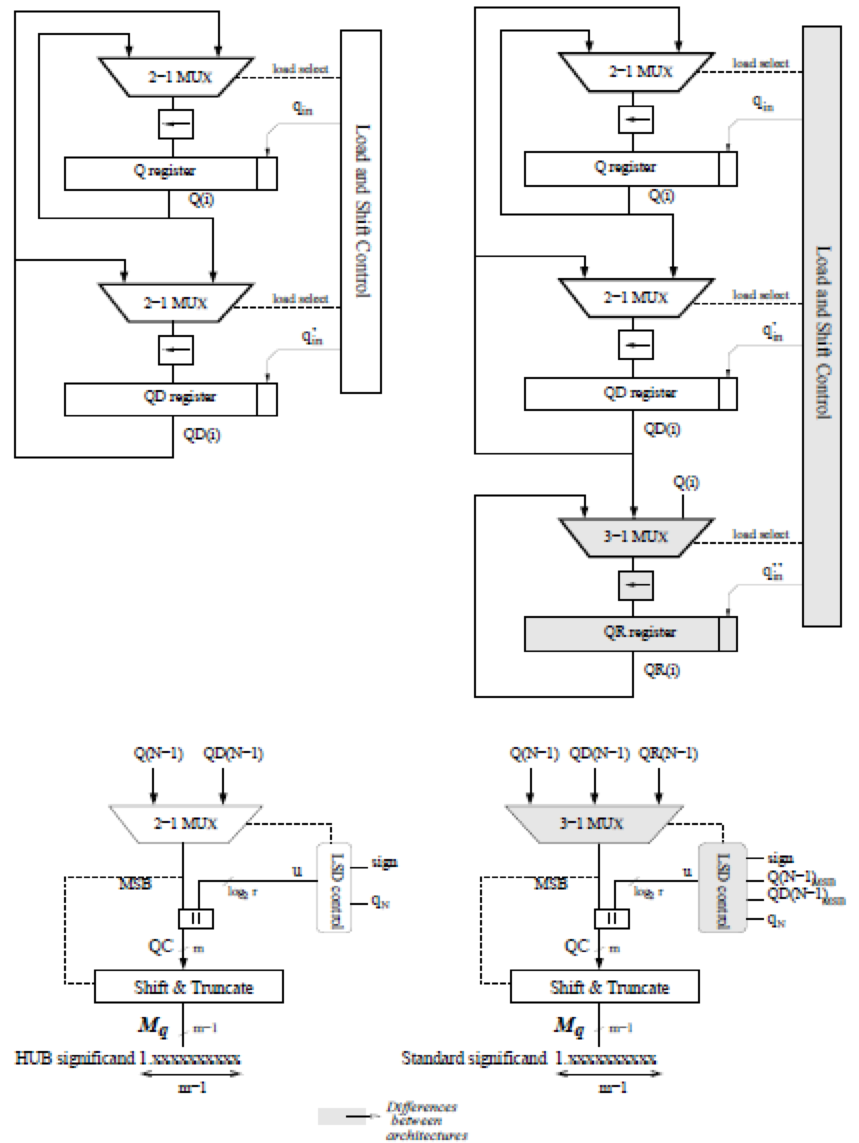
$$u = \begin{cases} q_N - sign & \text{if } q_N \geq 0 \\ r - |q_N| - sign & \text{if } q_N < 0 \end{cases}$$

$$QC = \begin{cases} (Q(N-1) \parallel u & \text{if } q_N \geq 0 \\ (QD(N-1) \parallel u & \text{if } q_N < 0 \end{cases}$$

On-the-fly conversion and rounding

HUB

Conventional

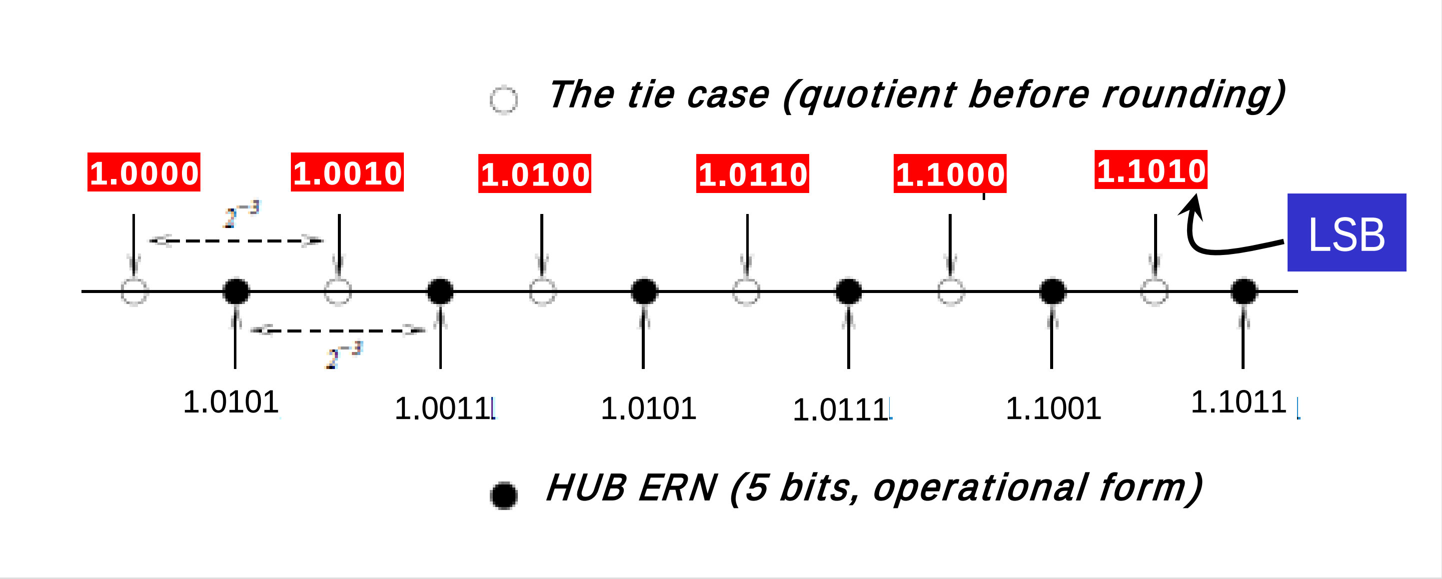


a) Architecture for HUB format

b) Architecture for standard format

On-the-fly conversion and rounding

- Unbiased round to nearest
  - In conventional IEEE standard the tie case never happens [10]
  - In HUB, it can happen
- The tie case in HUB:
  - Remainder is zero & LSB of the quotient is zero



On-the-fly conversion and rounding

– Unbiased round to nearest

Normalized Quotient before rounding

1.xxx...xxxKL

```
z   KL --> K'L'
0   xy     x 1 (not tie)
1   x1     x 1 (not tie)
1   00     0 1 (tie, rounding up)
1   10     0 1 (tie, rounding down)
```

Normalized Quotient before rounding (tie case) 1.xxx...xxx**K0** *(operational form)*

Normalized Quotient after rounding (tie case) 1.xxx...xxx**01** *(operational form)*

1.xxx...xxx**0** *(representative form)*

Summary and conclusion

- HUB format
    - Simplify some important operations
    - Represented with the same number of bits as its conventional counterpart
    - In general, it reduces the complexity of the hardware
- Digit recurrence division under HUB
    - The same data path bit-width as conventional (counterpart)
    - The same number of iterations
    - The same selection function
    - The same hardware than conventional format for the iterations
    - On-the-fly conversion for $a \geq r-2$
        - » Less harware requirements
        - » The selection of the last digit is simplified

THANK YOU!

QUESTIONS?