CONTINUOUS VALUED DIGITS

Mitra Mirhassani

Supervisor: Dr. M. Ahmadi

Co-Supervisor: Dr. G. A. Jullien

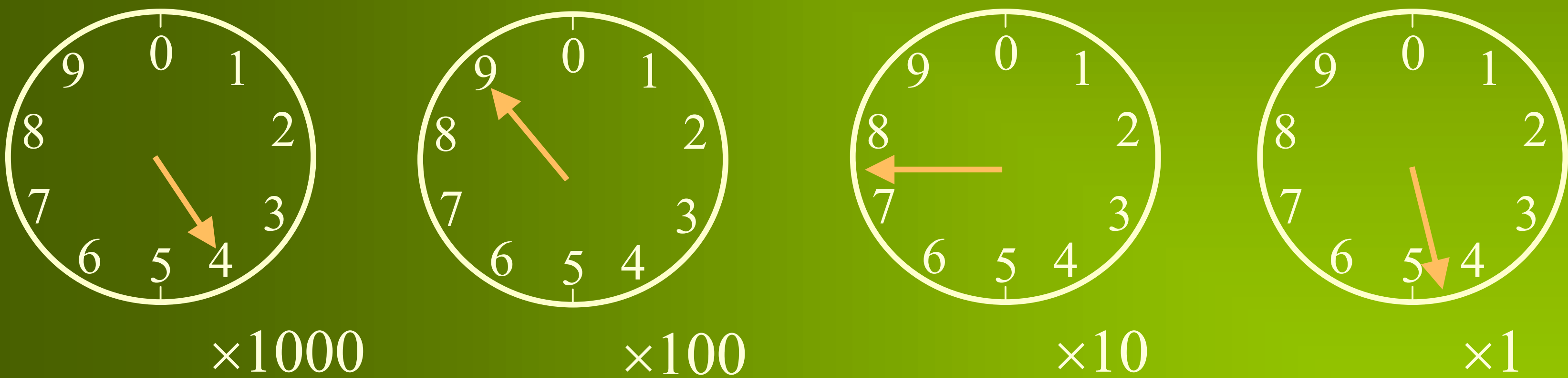
Winter 2004

OUTLINE

- Continuous Valued Digits (CVD) Idea
- CVD Representation
- CVD Generation
- CVD Implementation
- CVD Error Recovery
- Overflow Correction
- LSD Error Propagation
- Addition
- Subtraction
- Multiplication
- CVD Applications
- CVD Developments

CVD Idea

- Original idea: From “Utility Meter”, where lower digits refine higher order digits.



$X \approx (4, x_2, x_1, x_0)$ $X \approx (4, 9, x_1, x_0)$ $X \approx (4, 9, 7, x_0)$ $X \approx (4, 9, 7, 4)$

CVD Representation

- A given real number $|x| < X$ is represented by n set of CVDs.
- Radix B is defined based on the maximum dynamic range (X) as: $B^{L+1} = X$
- The format of the representation is as follows:

$$x \longrightarrow (r_L, r_{L-1}, \ldots, r_1, r_0 \mid r_{-1}, r_{-2}, \ldots, r_K) \qquad (L+1+k=n)$$

CVD Generation

- First Method

  Cascade rule

  - MSD: $r_L = B. x / X$
  - Lower order digits obtained as:

$$r_n = (r_{n+1} - \lfloor r_{n+1} \rfloor)$$

❖ $\lfloor . \rfloor$ denotes the floor function

CVD Generation

- Second Method

  Modulo operation

$$r_n = \left(\frac{x \cdot B^{L-n+1}}{X}\right) \bmod B \qquad \text{for } n \le L$$

- In both methods, $n \ge L$ digits are called ***"Excessively Evolved Digits (EED)"*** defined by:

$$r_n = r_{n-1} / B$$

❖ $(a) \bmod B = a - B \lfloor a / B \rfloor$

CVD Generation

- The original number (x) can be obtained from the MSD alone as:

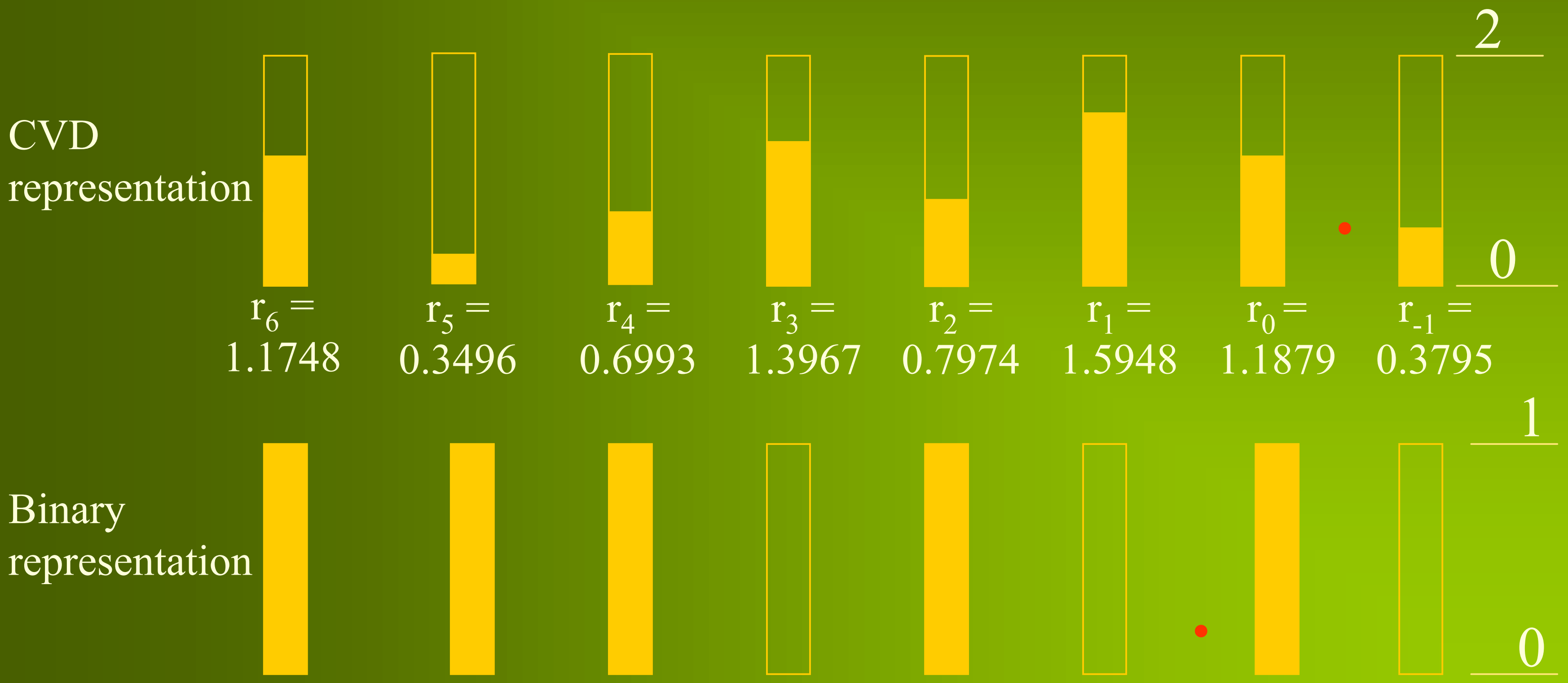
$$\tilde{x} = r_L . X / B$$

- CVD representation is not an approximation, $\tilde{x} = x$
- x can also be obtained from the less significant digits as follows:

$$x = \frac{X.r_k}{B^{L-k+1}} + \frac{X}{B} \sum_{i=0}^{L-n-1} r_{L-i} . B^{-i}$$

Number Representation

- x =58.742, B=2



CVD Implementation

- Each digit is represented by a continuous value
- If the electronic variable (current, charge, voltage) ranges from 0 to Q units, the CVD of x is matched as $q_n = r_n.Q / B$.

x = 58.742

X = 100

Range = [0-50] μA

B = 10

n	r_n	$\lfloor r_n \rfloor$	q_n (μA)
6	1.1748	1	29.371
5	0.3496	0	8.742
4	0.6993	0	17.484
3	1.3967	1	34.918
2	0.7974	0	19.936
1	1.5948	1	39.872
0	1.1879	1	29.699
-1	0.3795	0	9.488

CVD Error Recovery

- The increased resolution of higher digits can not be implemented exactly in VLSI.
- A CVD does not have noise margin, unlike binary digit representation.
- Protection against noise and other impairments is warranted by the redundancy among the digits.
- The error can be corrected using ***"Reverse Evolution"*** method.

CVD Error Recovery

- The error is corrected from the lower digit toward the higher digit.

r_n (original digit)

→ r'_n (errored digit) $= r_n + \varepsilon_n$

→ r''_n (corrected digit) $= \begin{cases} \lfloor r'_n \rfloor + r'_{n-1} / B & n > K \\ r'_k & n = K \end{cases}$

CVD Error Recovery

- x = 9.9845, B= 10

n	2	1	0	-1	-2
r_n	9.9845	9.845	8.45	4.5	5

Applying 5% error to each digit

r'_n	10.483725	10.33725	8.8725	4.725	5.25
	10 + 10.84525/10	10 + 8.4525/10	8 + 4.525/10	4 + 5.25/10	
r''_n	11.084525	10.84525	8.4525	4.525	5.25

Improved Algorithm

- The error correction is improved using the rounded computation of the floor function.

r_n (original digit)

→ r'_n (errored digit) $= x_n + e_n$

→ $\lfloor r'_n \rfloor_R = \lfloor r'_n - r''_n / B \rfloor$

→ r''_n (corrected digit) $= \begin{cases} \lfloor r'_n \rfloor_R + r''_{n-1} / B & n > K \\ r'_k & n = K \end{cases}$

Improved Algorithm

- $x = 9.9845$, $B = 10$

n	2	1	0	-1	-2
r_n	9.9845	9.845	8.45	4.5	5

Applying 5% error to each digit

r'_n	10.48372	10.3372	8.8725	4.725	5.25
$\lfloor r'_n \rfloor_R$	$\lfloor 9.39 \rfloor_R = 9$	$\lfloor 9.492 \rfloor_R = 9$	$\lfloor 8.42 \rfloor_R = 8$	$\lfloor 4.2 \rfloor_R = 4$	-
r''_n	9.9845	9.8452	8.4525	4.525	5.25

Overflow Correction

- By redefining the rounded floor function the overflow problem is eliminated.

$$\lfloor r'_n \rfloor_R = \lfloor r'_n - r''_n / B \rfloor_R \bmod^{+} B \qquad r' \geq 0$$

$$\lfloor r'_n \rfloor_R = \lfloor r'_n - r''_n / B \rfloor_R \bmod^{-} B \qquad r' < 0$$

$$r''_n = \begin{cases} \lfloor r'_n \rfloor_R + r''_{n-1} / B & n > K \\ r'_k & n = K \end{cases}$$

- $(a) \bmod^{+} B = (a \bmod B + B) \bmod B \qquad 0 \leq (a) \bmod^{+} B < B$

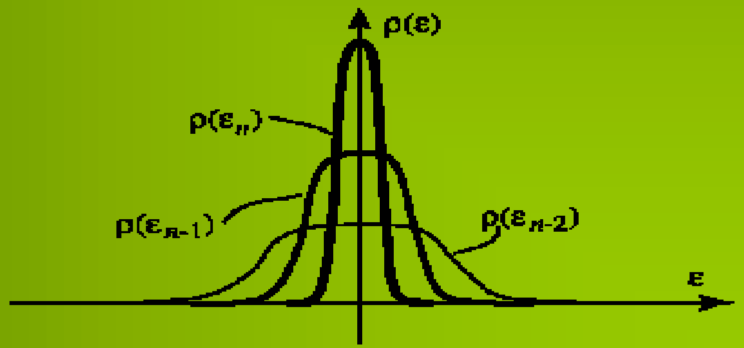
- $(a) \bmod^{-} B = (a \bmod B - B) \bmod B \qquad -B < (a) \bmod^{-} B \leq B$

LSD Error Propagation

- LSD error propagates upward by the reverse evolution process.
- The correction error is reduced by factor B in each step of the reverse evolution for each digit.
- The error propagated to the MSD from the LSD is $\varepsilon'_L = \varepsilon_k / B^{L-k}$

$$\rho^*(\varepsilon_{n+1}) = B.\ \rho(\varepsilon_n / B)$$

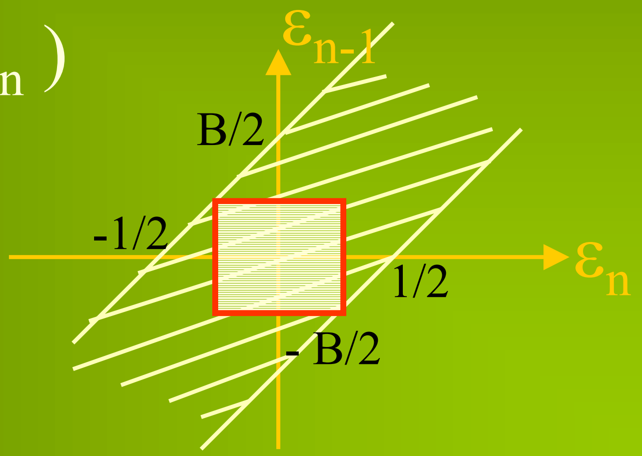
* ρ is the error probability density



Error probability density propagation

Error Tolerance

- All digit errors have a equal error probability density function.
- The reverse evolution process holds providing that $| \varepsilon_n - \varepsilon_{n-1} / B| < 0.5$
- The error threshold ($\nu \geq \varepsilon_n$) is $\frac{B}{2(B+1)}$.



Actual bound for adjacent digits.

Addition

- Given two numbers, x and y, the CVD digits of the sum , z= x+y is given as $z_n = (r_n + p_n) \mod B$.
- Addition operation is digit wise with no carry
- The only interaction between the digits is for error recovery.
- The summation of the EEDs maintains the validity of the addition in case of overflow.

Addition

- Adding x=58.34, y=72.89 in radix 10.

n	2 (EED)	1	0	-1	-2
x → r_n	0.5834	5.834	8.34	3.4	4
r'_n	0.8834	6.134	8.64	3.7	4.3
y → p_n	0.7289	7.289	2.89	8.9	9
p'_n	1.0289	7.589	3.19	9.2	9.3
z_n	1.9123	3.723	1.83	2.9	3.6
z'_n	2.2123	4.023	2.13	3.2	3.9
$\lfloor z'_n \rfloor_R$	1	3	1	2	3.9
z''_n	1.31239	3.1239	1.239	2.39	3.9

$$\begin{array}{r} 58.34 \\ +\ 72.89 \\ \hline 131.23 \end{array}$$

Subtraction

- Subtraction is performed by the addition of the negative numbers as $z_n = (x_n + (-y_n)) \bmod B$.
- Addition computation which generates the correct sign is performed as:

$$z_n = \begin{cases} (r_n + p_n) \bmod^{+} B & (r_L + p_L) \geq 0 \\ (r_n + p_n) \bmod^{-} B & (r_n + p_n) < 0 \end{cases}$$

Subtraction

- Adding x=58.34, y=-72.89 in radix 10.

n	2 (EED)	1	0	-1	-2
r_n	0.5834	5.834	8.34	3.4	4
r'_n	0.8834	6.134	8.64	3.7	4.3
p_n	-0.7289	-7.289	-2.89	-8.9	-9
p_n	-0.4289	-6.989	-2.59	-8.6	-8.7
z_n	0.1545	-0.855	-3.95	-4.9	-4.4
z'_n	-0.1455	-0.555	-3.65	-4.6	-4.1
$\lfloor z'_n \rfloor_R$	0	-1	-4	-5	-4.1
z''_n	-0.14541	-1.4541	-4.541	-5.41	-4.1

58.34
\- 72.89

\- 14.55

Multiplication

- Multiplication is not digit wise.
- Multiplication is performed by summation of partial products.
- The multiplicand is presented to the multiplier in positional number system with same radix.
- The CVD of a product λ.x (λ is integer) is:

$$(\lambda.r)_n = (\lambda.r_n) \bmod B$$

- Partial sums of product x.y are:

$$z_n = \left[\sum_{\forall K} p_k . r_{n-k} \right] \bmod B$$

Multiplication

- Multiple x=31.89 and y=19.5, in radix 10.

n	2	1	0	-1	-2
r_n	0.3189	3.189	1.89	8.9	9
r'_n	0.3489	3.219	1.92	8.93	9.03
$(\lambda_1 r'_n)'$	3.2254	1.9238	8.9479	9.0481	0
$(\lambda_0 r'_n)'$	3.1464	8.989	7.2946	0.3707	1.2725
$(\lambda_{-1} r'_n)'$	0.2045	1.7745	6.125	9.63	4.68
z'_n	6.6063	2.7173	2.3974	9.0788	5.9825
$\lfloor z'_n \rfloor$	6	2	1	8	5.9825
z''_n	6.2186	2.1859	1.8595	8.5953	5.9825

$$r'_n = r_n + \varepsilon^+$$

$$(\lambda_i r'_n)' = (1+\varepsilon^*)(\lambda_i r'_n)$$

$\lambda \rightarrow (1,9,5)$, $\varepsilon^+ = 0.03$, $\varepsilon^* = 0.2\%$, $31.89 \times 19.5 = 621.855$

CVD Applications

- Target application: Digital filters and Neural networks.
- CVD may provides silicon efficient signal processing methods (power, area, speed, complexity).
- The CVD operations are performed in analog.
- The accuracy can be made identical to digital number system.
- Digital filters benefit from the proposed array multiplier

CVD Developments

- Digit correction
- Interfacing with digital circuits
- Singed-number adder implementation
- Optimizing the subtraction algorithm
- Efficient multiplication algorithm
- Multiplier implementation