Advanced Parallel Architecture

Lesson 8

Annalisa Massini - 2016/2017

Redundant number systems

Redundant number systems

- Conventional radix-r systems use [0, r-1] digit set

  radix-10 → 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

- If the digit set (in radix-r system) contains more than **r** digits, the system is **redundant**
  - radix-2 → 0, 1, 2 or -1, 0, 1
  - radix-10 → 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
  - radix-10 → -6, -5,- 4, -3, -2, -1, 0, 1, 2, 3, 4, 5
- **Redundancy** may result from adopting the digit set wider than radix and the number interpretation is conventional
- **Redundancy** – representation of numbers is *not unique*

Redundant number systems

- Redundant numbers with [0,m] digit set can be represented by two numbers of [0,n] digit sets, where m=2n
- Conversion requires ordinary addition of two such numbers with [0,n] digit set representation

$$
\begin{array}{cccccccl}
 & & 11 & 9 & 17 & 10 & 12 & 18 \quad \text{radix-10, digit set [0,18]} \\
\hline
 & & 9 & 9 & 9 & 9 & 9 & 9 \\
+ & & 2 & 0 & 8 & 1 & 3 & 9 \\
\hline
 & 1 & 2 & 0 & 8 & 1 & 3 & 8 \quad \text{radix-10, digit set [0,9]}
\end{array}
$$

- Decomposed representation is not unique, but the sum amounts to correct result

Redundant number systems

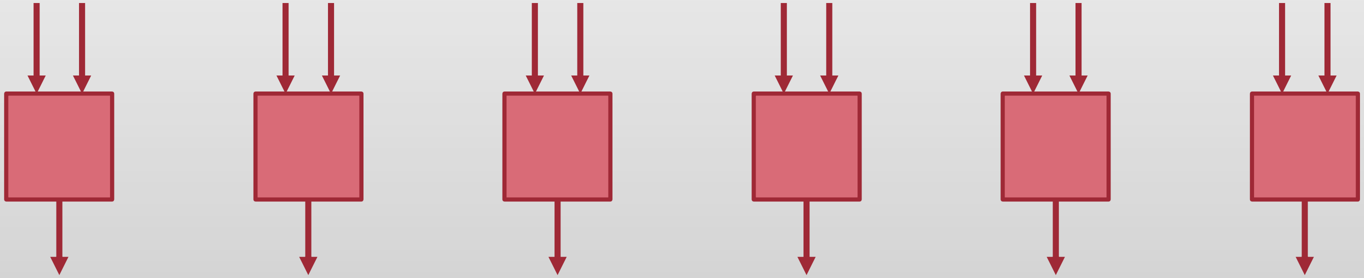
- Redundant binary numbers may be coded with bit-fields, e.g:
  - 0: (0,0),
  - 1: (0,1) or (1,0),
  - 2: (1,1)
- Decomposed representation is not unique, but the sum amounts to correct result

$$
\begin{array}{cccccccc l}
 & & 1 & 1 & 2 & 0 & 2 & 0 & \quad\text{radix-2, digit set [0,2]} \\
\cline{1-8}
 & & 1 & 1 & 1 & 0 & 1 & 0 & \\
+ & & 0 & 0 & 1 & 0 & 1 & 0 & \\
\cline{1-8}
 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & \quad\text{radix-2, digit set [0,1]}
\end{array}
$$

Redundant number systems

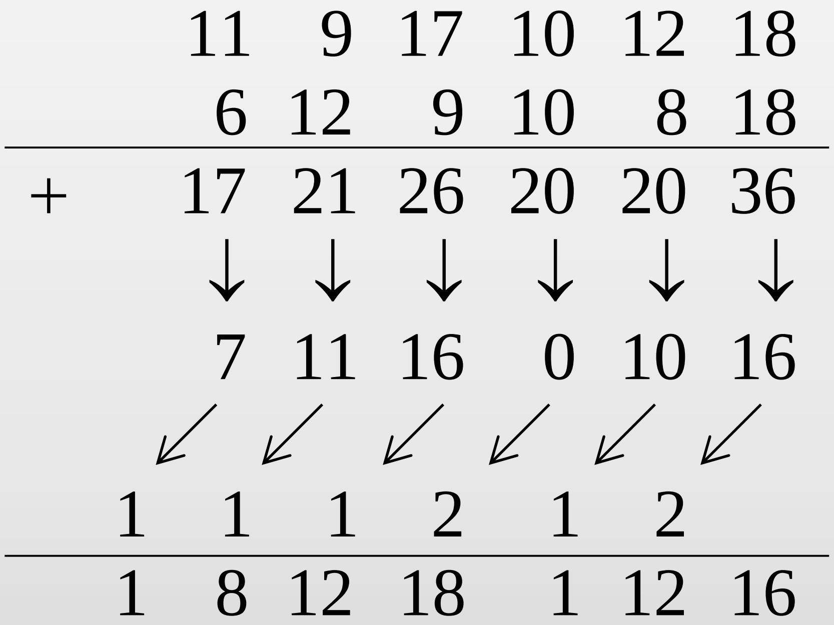
- **Carry-free** addition → no carry propagation
- All digit additions can be done simultaneously
- Carry-free addition is possible with widening of the digit set

$$\begin{array}{rrrrrrrl} & 1 & 2 & 3 & 4 & 5 & 6 & \text{radix-10, digit set } [0,9] \\ + & 4 & 5 & 6 & 7 & 8 & 9 & \text{radix-10, digit set } [0,9] \\ \hline & 5 & 7 & 9 & 11 & 13 & 15 & \text{radix-10, digit set } [0,18] \end{array}$$



Redundant number systems

- Reduction of digit set by **carry propagation by only one position**



		11	9	17	10	12	18	radix-10, digit set [0,18]
		6	12	9	10	8	18	radix-10, digit set [0,18]
+		17	21	26	20	20	36	radix-10, digit set [0,36]
		↓	↓	↓	↓	↓	↓	
		7	11	16	0	10	16	Intermediate sums [0,16]
	↙	↙	↙	↙	↙	↙		
	1	1	1	2	1	2		Transfer digit set [0,2]
	1	8	12	18	1	12	16	sum [0,18]

Signed-digit numbers

- All digits have weights r^p (p-position, r-radix)
- Digits can have signed values
- Any set digit $[-\alpha, \beta]$ including 0, can be used
- If $\alpha+\beta+1 > r$ the numbering system is redundant

[-1,1] radix-2 → $1\ -1\ 0\ -1\ 0 = 6_{(10)}$

[-1,3] radix-4 → $1\ -1\ 2\ 0\ 3 = 227_{(10)}$

1111 (2's compl.) → $-1\ 1\ 1\ 1 = -1$

Signed-digit numbers

- A **radix-r redundant signed-digit** number system is based on digit set S = {- β, -(β - 1), … , -1, 0, 1, … , α},

  where $1 \leq \alpha, \beta \leq r - 1$

- The digit set S contains more than r values → multiple representations for any number in signed digit format → redundant
- A symmetric signed digit has α = β
- **Carry-free addition** is an attractive property of redundant signed-digit numbers

Signed digit representation

- In mathematical notation for numbers, **signed-digit representation** is a **positional system** with signed digits
- The representation may not be unique
- Signed-digit representation can be used to accomplish fast addition of integers because it can eliminate chains of dependent carries

Modified signed digit representation

A. K. Cherri, M. A. Karim, “Modified-signed digit arithmetic using an efficient symbolic substitution”, Appl. Opt. (1988)

Modified signed digit representation

- The set of digit is $\{-1,0,1\} = \{\bar{1},0,1\}$

- The representation is not unique:

$$\bar{1}01\bar{1} = -8+2-1 = -7$$
$$\bar{1}001 = -8+1 = -7$$
$$\bar{1}1\bar{1}\bar{1} = -8+4-2-1 = -7$$

- The number of possible representation depends on the length of the sequence of digits
- To perform the addition, truth table are used

Modified signed digit representation

- Truth tables

		First addend		
		-1	0	1
Second addend	-1	0 -1	1 -1	0 0
	0	1 -1	0 0	-1 1
	1	0 0	-1 1	0 1

		First addend		
		-1	0	1
Second addend	-1	0 -1	-1 0	0 0
	0	-1 0	0 0	1 0
	1	0 0	1 0	0 1

- Three steps are needed to obtain the sum
  - Left table is applied in step 1 and 3
  - Right table is applied in step 2
- Output: sum → lower row - complemented sum → upper row

Modified signed digit representation

- Example

$$\begin{array}{ccccccc} 1 & \bar{1} & 0 & 1 & \bar{1} & & 9 \\ \bar{1} & 1 & \bar{1} & 1 & 0 & & -10 \end{array}$$

		First addend		
		-1	0	1
Second addend	1	0 -1	1 -1	0 0
	0	1 -1	0 0	-1 1
	1	0 0	-1 1	0 1

		First addend		
		-1	0	1
Second addend	-1	0 -1	-1 0	0 0
	0	-1 0	0 0	1 0
	1	0 0	1 0	0 1

Modified signed digit representation

- Example

$$\begin{array}{ccccccc} & 1 & \bar{1} & 0 & 1 & \bar{1} & 9 \\ & \bar{1} & 1 & \bar{1} & 1 & 0 & -10 \\ \hline & 0 & 0 & 1 & 0 & 1 & \\ 0 & 0 & \bar{1} & 1 & \bar{1} & \mathit{0} & \end{array}$$

		First addend		
		-1	0	1
Second addend	-1	0 -1	1 -1	0 0
	0	1 -1	0 0	-1 1
	1	0 0	-1 1	0 1

		First addend		
		-1	0	1
Second addend	-1	0 -1	-1 0	0 0
	0	-1 0	0 0	1 0
	1	0 0	1 0	0 1

Modified signed digit representation

- Example

$$
\begin{array}{cccccccc}
 & 1 & \bar{1} & 0 & 1 & \bar{1} & & 9 \\
 & \bar{1} & 1 & \bar{1} & 1 & 0 & & -10 \\
\hline
 & 0 & 0 & 1 & 0 & 1 & & \\
0 & 0 & \bar{1} & 1 & \bar{1} & \mathit{0} & & \\
\hline
 & 0 & \bar{1} & 0 & \bar{1} & 1 & & \\
0 & 0 & 1 & 0 & 0 & \mathit{0} & &
\end{array}
$$

		First addend		
		-1	0	1
Second addend	-1	0 -1	1 -1	0 0
	0	1 -1	0 0	-1 1
	1	0 0	-1 1	0 1

		First addend		
		-1	0	1
Second addend	-1	0 -1	-1 0	0 0
	0	-1 0	0 0	1 0
	1	0 0	1 0	0 1

Modified signed digit representation

- Example

$$
\begin{array}{cccccccc}
 & 1 & \bar{1} & 0 & 1 & \bar{1} & & 9 \\
 & \bar{1} & 1 & \bar{1} & 1 & 0 & & -10 \\
\hline
 & 0 & 0 & 1 & 0 & 1 & & \\
0 & 0 & \bar{1} & 1 & \bar{1} & \mathit{0} & & \\
\hline
 & 0 & \bar{1} & 0 & \bar{1} & 1 & & \\
0 & 0 & 1 & 0 & 0 & \mathit{0} & & \\
\hline
 & 0 & 0 & 0 & 1 & \bar{1} & & \\
0 & 0 & 0 & \bar{1} & 1 & & &
\end{array}
$$

		First addend		
		-1	0	1
Second addend	-1	0 -1	1 -1	0 0
	0	1 -1	0 0	-1 1
	1	0 0	-1 1	0 1

		First addend		
		-1	0	1
Second addend	-1	0 -1	-1 0	0 0
	0	-1 0	0 0	1 0
	1	0 0	1 0	0 1

RB - Redundant binary number representation

G. A. De Biase, A. Massini “Redundant binary number representation for an inherently parallel arithmetic on optical computers”,

Appl. Opt., 32 (1993)

RB - Redundant Binary Representation

- An integer D obtained by

$$D = \sum_{i=0}^{n-1} a_i 2^{i - \lceil i/2 \rceil}$$

- This weight sequence characterizes the RB number representation and is:

$$\begin{array}{ccccccccc} \cdots & 8 & 8 & 4 & 4 & 2 & 2 & 1 & 1 \\ & r & n & r & n & r & n & r & n \end{array}$$

- ***All position weights are doubled***: the left digit is called *r* (*redundant*) and the right digit *n* (*normal*)

RB - Redundant Binary Representation

- RB representation of a number can be obtained from its binary representation by the following recoding rules:

  0 → 00 1 → 01

- The RB number obtained in this way is in **canonical form**

- This coding operation is performable in parallel in constant time (one elemental logic step)

RB - Redundant Binary Representation

- Each RB number has a canonical form and several redundant representations

- Examples of ***unsigned*** RB numbers (canonical and redundant)

0	000	000000				
1	001	000001	000010			
2	010	000100	001000	000011		
3	011	000101	001001	001010		
4	100	010000	100000	001100	000111	
5	101	010001	010010	100001	100010	
6	110	010100	011000	101000	010011	
7	111	010101	010110	101001	101010	

Table for addition

- Truth table

	00	01	10	11
00	00 00	10 00	00 01	10 01
01	00 01	10 01	00 10	10 10
10	00 01	10 01	00 10	10 10
11	00 10	10 10	00 11	10 11

Table for addition

- **Two steps**: parallel application of the table 2 on all *rn* pairs
- Output: **sum** on the lower row and **zero** on the upper row

	00	01	10	11
00	00 00	10 00	00 01	10 01
01	00 01	10 01	00 10	10 10
10	00 01	10 01	00 10	10 10
11	00 10	10 10	00 11	10 11

RB - Redundant Binary Representation

- Example

0 0	0 1	0 1	1 1	8
0 0	1 1	0 1	1 0	11

	00	01	10	11
00	00 00	10 00	00 01	10 01
01	00 01	10 01	00 10	10 10
10	00 01	10 01	00 10	10 10
11	00 10	10 10	00 11	10 11

RB - Redundant Binary Representation

- Example

0	0	0	1	0	1	1	1	8
0	0	1	1	0	1	1	0	11
0	0	1	0	1	0	1	0	
0	1	0	0	1	1	0	*0*	

	00	01	10	11
00	00 00	10 00	00 01	10 01
01	00 01	10 01	00 10	10 10
10	00 01	10 01	00 10	10 10
11	00 10	10 10	00 11	10 11

RB - Redundant Binary Representation

- Example

0	0	0	1	0	1	1	1	8
0	0	1	1	0	1	1	0	11
0	0	1	0	1	0	1	0	*7*
0	1	0	0	1	1	0	*0*	*12*
0	0	0	0	0	0	0	0	0
1	0	1	1	1	0	1	*0*	19

	00	01	10	11
00	00 00	10 00	00 01	10 01
01	00 01	10 01	00 10	10 10
10	00 01	10 01	00 10	10 10
11	00 10	10 10	00 11	10 11

RB - Redundant Binary Representation

- In analogy with the 2's complement binary system, a **signed RB number** is obtained by

$$D = -\sum_{i=n-2}^{n-1} a_i 2^{i-\lceil i/2 \rceil} + \sum_{i=0}^{n-3} a_i 2^{i-\lceil i/2 \rceil} \qquad n \text{ even}$$

- The same procedure of the addition of two unsigned RB numbers obtains the algebraic sum of two signed RB numbers

RB - Redundant Binary Representation

- The additive inverse of an RB number is obtained by following a procedure similar to that used in the 2's complement number system, taking into account that the negation of all RB representations of the number 0 is $(-2)_{10}$ whereas in the 2's complement binary system it is $(-1)_{10}$
- ***Procedure***
  - Step 1 - all digits of the RB number are complemented
  - Step 2 - algebraic sum between the RB canonical form of $(2)_{10}$ and the RB number
  - The output is the **additive inverse** of the considered RB number

RB - Redundant Binary Representation

- The **decoding** of RB numbers, with the correct truncation, can be performed with the following procedure that derives directly from the RB number definition
- ***Procedure***
  - The input is RB*n* and RB*r*
  - Binary addition RB + RBr.
  - Only the first n/2 bits are considered
  - The output is the corresponding binary or 2's complement binary number

RB - Redundant Binary Representation

- **Zero and Its Detection**
- In the case of unsigned RB numbers the $(0)_{10}$ has only the RB canonical form and is easily detectable
- In the case of signed RB numbers, $(0)_{10}$ has many RB representations
- Example for six-digit signed RB numbers:

(000000)	(101011)	(101100)
(100111)	(010111)	(011100)

- This difficulty can be overcome by using the number $(-1)_{10}$ instead of $(0)_{10}$

RB - Redundant Binary Representation

- **Zero and Its Detection**
- In fact, any redundant representation of the number $(-1)_{10}$ obtains the canonical representation of the $(-1)_{10}$ if the following rules acting on *rn* pairs are applied

$$01 \rightarrow 01 \qquad 10 \rightarrow 01$$

- Then, if the result of an algebraic sum between an RB number and an RB representation of $(-1)_{10}$ is an RB representation of the number $(-1)_{10}$ again, this RB number is a representation of $(0)_{10}$

RB - Redundant Binary Representation

- **Zero and Its Detection**
- Then the procedure to detect the number $(0)_{10}$ is:

Procedure

  - Input an RB number
  - Step 1 - algebraic sum between the RB canonical form of $(- 1)_{10}$ and the RB number
  - Step 2 - application of rules to the result
  - Output is the RB canonical form of $(-1)_{10}$ or of another RB number

RB - Redundant Binary Representation

- **Comparison of two RB numbers**

Procedure

- Input - the first RB number and the additive inverse of a second RB number
- Step 1 - algebraic sum between the two RB numbers
- Step 2 - Procedure for the zero detection applied to the result
- The output is the RB canonical form of $(-1)_{10}$ or of another RB number