TSTE18 Digital Arithmetic
Seminar 6

Oscar Gustafsson

Division and square-root

- **Restoring division**
- **Non-restoring division**
- **SRT division**
- **Higher-radix division**

- Reciprocals
- Division by convergence
- Square-rooting

Division

- Of the four basic arithmetic operations, division is the most complex to compute
- The result of a division consists of two components, the quotient, Z, and the remainder, R, such that

$$X = ZD + R \tag{1}$$

  where X is the dividend, $D \neq 0$ is the divisor, and $|R| < D$
- By definition the sign of the remainder should be the same as that of the dividend

Normalization

- It is common to normalize the inputs and therefore the result
- A common way is to make sure that the dividend is smaller than the divisor
- This leads to $|\frac{X}{D}| \leq 1$ and can be easily implemented by a programmable shifter and a leading zero/one detector
- We can assume that this holds when discussing the algorithms

Restoring division

- Shift the dividend one position (multiply by two) and check if the divisor is larger than the dividend
- If so, set the corresponding bit of the quotient to one and subtract the divisor from the dividend
- Conceptually: first subtract the divisor from the dividend and then check if the result is positive (quotient bit is one) or negative (quotient bit is zero)
- For negative result the divisor is added again, which gives the name restoring division
- The computation in step i can be written as

$$r_i = 2r_{i-1} - z_i D \tag{2}$$

  where $r_0 = X$
- If $2r_{i-1} - D$ is positive, we set $z_i = 1$, otherwise $z_i = 0$ and $r_i = 2r_{i-1}$
- r_i is the remainder after iteration i, leading to $R = r_i 2^{-1}$
- A quotient with W_f fractional bits require W_f iterations

Restoring division example

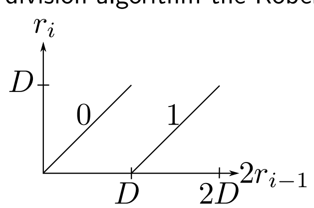
- Divide $X = 0101 = 5 = r_0$ with $D = 1001 = 9$

Iteration	$2r_{i-1} - D$	z_i	$r_i = 2r_{i-1} - z_i D$
1	01010 − 01001	1	0001
2	00010 − 01001	0	0010
3	00100 − 01001	0	0100
4	01000 − 01001	0	1000
5	10000 − 01001	1	0111
6	01110 − 01001	1	0101
7	01010 − 01001	1	0001
...			

- Result: $Z = 0.1000111$ and $R = 0001 \times 2^{-7}$
- So: $101 = 1001 \times 0.1000111 + 0.0000001$

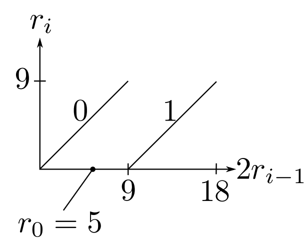
Restoring division Robertson diagram

- A Robertson diagram shows the output bit selection and remainder for the next iteration as a function of the input remainder
- For the restoring division algorithm the Robertson diagram is

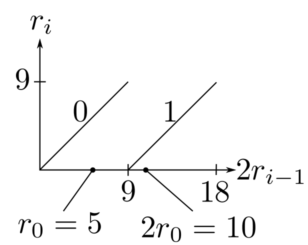


Restoring division Robertson diagram example

- Consider the previous example with $X = 0101 = 5 = r_0$ with $D = 1001 = 9$
- Initial state

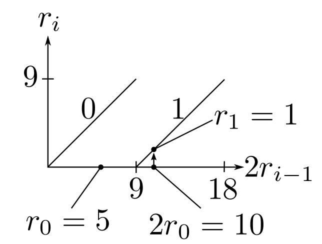


- Compute $2r_0$

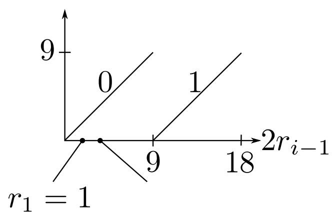


Restoring division Robertson diagram example

- Check r_1

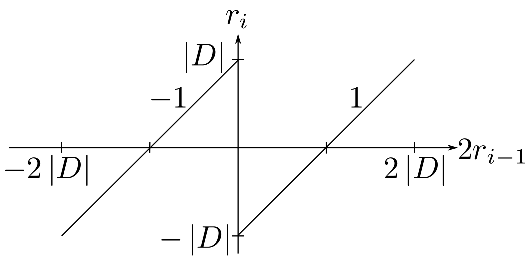


- Next iteration



Non-restoring division Robertson diagram

- For the non-restoring division algorithm the Robertson diagram is



Non-restoring division

- Instead of restoring the remainder by adding the divisor, we can assign a negative quotient digit
- This gives the non-restoring division selection rule of the quotient digits, z_i, in (2) as

$$z_i = \begin{cases} 1 & r_{i-1}D \geq 0 \text{ i.e. same sign} \\ -1 & r_{i-1}D < 0 \text{ i.e. different signs} \end{cases} \quad (3)$$

- With this definition of the selection rules the remainder will sometimes be positive, sometimes negative
- Hence, division with a signed dividend and/or divisor is easy
- In that case the final remainder does not have the same sign as the dividend, we must compensate by adding or subtracting D to R and consequently subtracting or adding one LSB to Z
- The result from the non-restoring division is represented with $q_i \in \{-1, 1\}$
- On-the-fly conversion to two's complement available
- If a zero remainder is obtained, this will not remain zero in the succeeding stages, so it needs to be detected and corrected for

Non-restoring division example

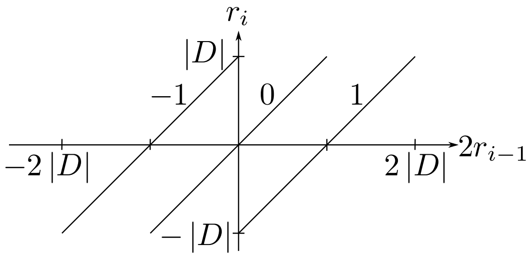
- Divide $X = 00101 = 5 = r_0$ with $D = 01001 = 9 = D_N$

Iteration	*signs*	z_i	$r_i = 2r_{i-1} - z_iD$
1	$++$	1	$2 \cdot 5 - 9 = 1$
2	$++$	1	$2 \cdot 1 - 9 = -7$
3	$-+$	-1	$2 \cdot (-7) + 9 = -5$
4	$-+$	-1	$2 \cdot (-5) + 9 = -1$
5	$-+$	-1	$2 \cdot (-1) + 9 = 7$
6	$++$	1	$2 \cdot (7) - 9 = 5$
7	$++$	1	$2 \cdot (5) - 9 = 1$
...			

- Result: $Z = 0.11\bar{1}\bar{1}\bar{1}11$ and $R = 1 \times 2^{-7}$
- So: $101 = 1001 \times 0.11\bar{1}\bar{1}\bar{1}11 + 0.0000001$
- Compare $0.1\underbrace{1\bar{1}\bar{1}\bar{1}}_{0001}11$

SRT division

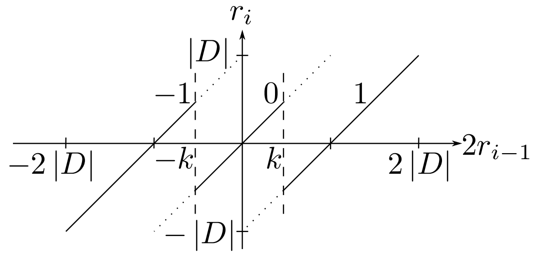
- Assume that we allow 0 as a quotient digit using non-restoring division
- The corresponding Robertson diagram is



- Clearly, there should be a well defined value for each point

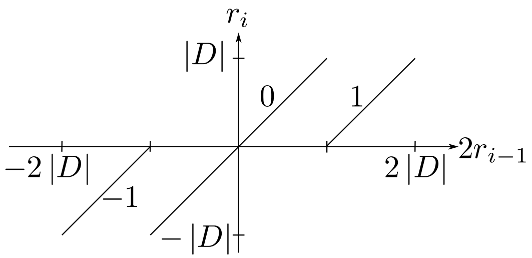
SRT division

- Select a range $-D_N \leq -k \leq 0 \leq k \leq D_N$ where the algorithm returns a zero quotient digit



SRT division

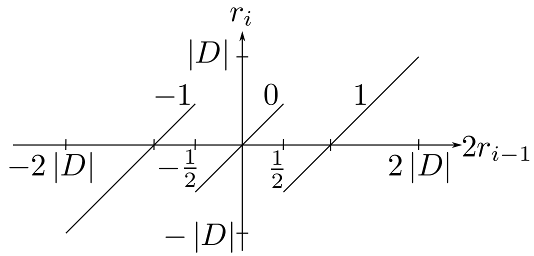
- A natural choice may be $k = |D|$



- However, this requires a comparison with $|D|$ which in principle require a carry propagation

SRT division

- An easier comparison is obtained with $k = \frac{1}{2}$



- This requires that $\frac{1}{2} \leq D \leq 1$, so the divisor must be normalized
- With $k = \frac{1}{2}$, the binary (radix-2) SRT division algorithm is obtained

SRT division

- The selection rule for the binary SRT algorithm, assuming $\frac{1}{2} \le D < 1$ is

$$z_i = \begin{cases} 1, & 2r_{i-1} \ge 1/2 \quad \Leftrightarrow \quad 2r_{i-1} = 0.1\ldots \\ 0, & -1/2 \le 2r_{i-1} < 1/2 \quad \Leftrightarrow \quad 2r_{i-1} = \begin{cases} 0.0\ldots \\ 1.1\ldots \end{cases} \\ -1, & 2r_{i-1} < -1/2 \quad \Leftrightarrow \quad 2r_{i-1} = 1.0\ldots \end{cases} \tag{4}$$

- This has two main advantages
  - When $z_i = 0$ there is no need to add or subtract
  - Comparing with $1/2$ or $-1/2$ only requires two bits of $2r_{i-1}$
- Example: Divide $X = 0.0101 = 5/16 = r_0$ with $D = 0.1001 = 9/16$

Higher-radix division

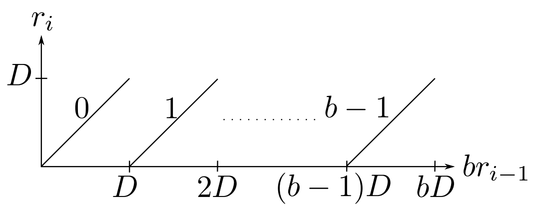
- The number of additions/subtractions is reduced in the SRT scheme, but not the number of cycles
- To reduce the number of cycles higher-radix division algorithms can be used
- Using radix-$b = 2^m$ reduces the number of iterations to $\left\lceil \frac{W_f}{m} \right\rceil$
- The basic recurrence is now

$$r_i = br_{i-1} - z_i D \tag{5}$$

where $z_i \in \{0, 1, 2, \ldots, b-1\}$ for restoring division

Higher-radix division

- The Robertson diagram for radix-b restoring division is



- Hence, we need to compare against several different values to determine the correct quotient digit
- Non-restoring division can be derived in a similar way

Higher-radix division

- For higher-radix SRT division we can, similar to redundant number systems, use $z_i \in \{-a, -a+1, \ldots, -1, 0, 1, \ldots, a\}$ where $\lceil (b-1)/2 \rceil \le a \le (b-1)$
- Let us consider the recurrence

$$r_i = br_{i-1} - z_i D \tag{6}$$

- The purpose is to select a quotient digit, z_i, such that the remainder, r_i, is within the given bounds, say $-|D| \le r_{\min} \le r_i \le r_{\max} \le |D|$
- Now, define an interval $L_d \le br_{i-1} \le U_d$ for a digit $d \in \{-a, -a+1, \ldots, -1, 0, 1, \ldots, a\}$ such that we should select d as the quotient digit and still meet the bound on the remainder above
- Relating these equations we get

$$L_d = dD + r_{\min} \tag{7}$$
$$U_d = dD + r_{\max} \tag{8}$$

Higher-radix division

- For the tail digits we obtain

$$L_{-a} = -aD + r_{\min} \quad (9)$$
$$U_a = aD + r_{\max} \quad (10)$$

- This gives

$$r_{\min} = -\mu|D| \quad (11)$$
$$r_{\max} = \mu|D| \quad (12)$$

where $\mu = \frac{a}{b-1}$, i.e., the redundancy factor $\left(\frac{1}{2} \leq \mu \leq 1\right)$

Higher-radix division

- This leads to the following bounds on the regions to guarantee convergence

$$L_d = (d-\mu)|D| \quad (13)$$
$$U_d = (d+\mu)|D| \quad (14)$$

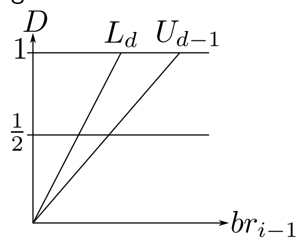
and $U_{d-1} \geq L_d$ (overlapping regions)
- The latter constraint holds as

$$U_{d-1} - L_d = (d-1+\mu)|D| - (d-\mu)|D| = (2\mu-1)|D| \geq 0 \quad (15)$$

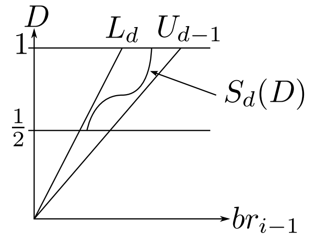
- Hence, the larger redundancy the larger overlap between digit selection regions

Higher-radix division

- Consider such a region



- It is required to select a function $S_d(D)$ such that if $S_d(D) \leq br_{i-1} < S_{d+1}(D)$ we select $z_i = d$



Higher-radix division

- The larger μ the larger overlap between the regions and easier to find a good $S_d(D)$, but at the same time more $S_d(D)$ are required
- Hence, there is a non-trivial complexity trade-off
- To further speed it up, it is possible to overlap the complete computation of the partial remainder in step i and the selection of the quotient digit in step $i+1$ since not all bits of the remainder must be known if S_d is selected in a clever way
- It is also possible to compute the remainder in a redundant number system, causing comparisons to be slightly harder

Higher-radix SRT division example

- Consider a radix-4 SRT division using the digit-set $a \in \{-2,-1,0,1,2\}$ and assume $\frac{1}{2} \le D < 1$
- Redundancy index $\mu = \frac{a}{b-1} = \frac{2}{3}$
- Initial selection ranges

d	L_d	U_d
-2	$-\frac{8}{3}D$	$-\frac{4}{3}D$
-1	$-\frac{5}{3}D$	$-\frac{1}{3}D$
0	$-\frac{2}{3}D$	$\frac{2}{3}D$
1	$\frac{1}{3}D$	$\frac{5}{3}D$
2	$\frac{4}{3}D$	$\frac{8}{3}D$

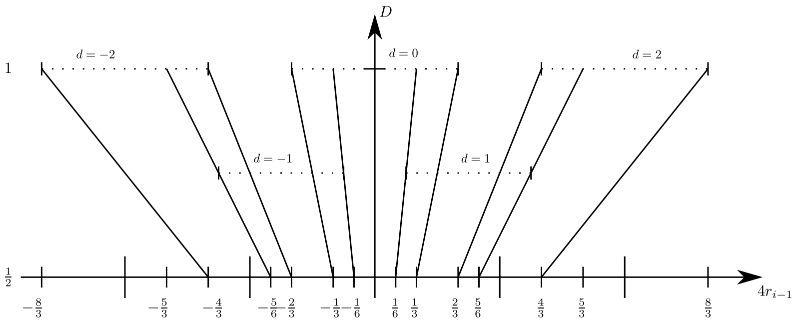
Higher-radix SRT division example

- With $\frac{1}{2} \le D < 1$ the possible ranges are
- Initial selection ranges

	$D=\frac{1}{2}$		$D=1$	
d	L_d	U_d	L_d	U_d
-2	$-\frac{4}{3}$	$-\frac{2}{3}$	$-\frac{8}{3}$	$-\frac{4}{3}$
-1	$-\frac{5}{6}$	$-\frac{1}{6}$	$-\frac{5}{3}$	$-\frac{1}{3}$
0	$-\frac{1}{3}$	$\frac{1}{3}$	$-\frac{2}{3}$	$\frac{2}{3}$
1	$\frac{1}{6}$	$\frac{5}{6}$	$\frac{1}{3}$	$\frac{5}{3}$
2	$\frac{2}{3}$	$\frac{4}{3}$	$\frac{5}{3}$	$\frac{8}{3}$

Higher-radix SRT division example

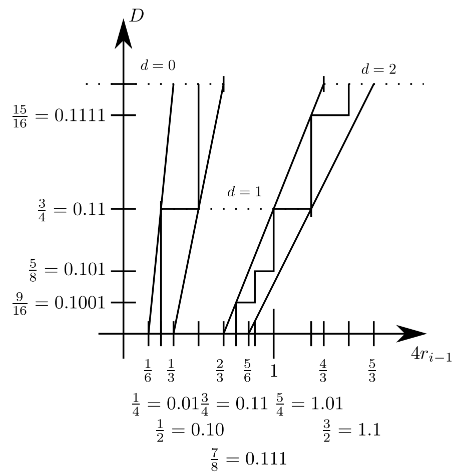
- Resulting ranges



- Find suitable selection function (consider only positive values due to symmetry)

Higher-radix SRT division example

- Possible ranges



Higher-radix SRT division example

- Assuming $4r_{i-1} \geq 0$

$$d_i = \begin{cases} 0, & 4r_{i-1} = 0.00X \wedge D = 0.10X \\ 0, & 4r_{i-1} = 0.01X \wedge D = 0.11X \\ 2, & 4r_{i-1} = 0.110X \wedge D = 0.1000X \\ 2, & 4r_{i-1} = 0.111X \wedge D = 0.1001X \\ 2, & 4r_{i-1} = 1.00X \wedge D = 0.101X \\ 2, & 4r_{i-1} = 1.01X \wedge D = 0.11X \wedge D \neq 0.1111X \\ 2, & 4r_{i-1} = 1.1X \wedge D = 0.1111X \\ 1, & \text{otherwise} \end{cases}$$