TSTE18 Digital Arithmetic
Seminar 1

Oscar Gustafsson

Involved persons

- Oscar Gustafsson, oscarg@isy.liu.se, 013-28 40 59
  - Course responsible
  - Seminars
  - Assignment
  - Examiner
- Syed Asad Alam, asad@isy.liu.se
  - Labs
  - Hand-ins
  - Assignment

Practical details

- Course homepage:
  `http://www.isy.liu.se/en/edu/kurs/TSTE18/`
- E-mail list (make sure you are signed up)
- Thirteen (plus two) seminars
  - Joint lectures (and exercises)
  - Presents the course contents
  - Slides available on the homepage 24h in advance, if not, there will be printed ones at the lecture
  - Second to last seminar will (hopefully) include a guest lecturer
- Laboratory work
  - Six labs focusing on computer-based problem solving
  - Not supervised, computer labs booked for you
- Assignment
  - One overview seminar
  - One seminar for presentations (mandatory)

Practical details – examination

- LAB1 – 4HP
  - Hand-ins – a number of INDIVIDUAL hand-ins following each seminar
  - Laboratory – INDIVIDUAL computer-based problems
    - Solved using e.g. Matlab, VHDL, and/or Verilog (you pick, some recommendations given)
    - Other languages at request
- UPG1 – 2HP
  - Small “project” to be done individually or in pairs
  - More details and presentation of suggested topics on final seminar this study period
  - Written report and short oral presentation (mandatory attendance)
  - Presentations not scheduled yet

Course material

- Hand-outs (slides)
- Problems (hand-ins and labs)
- Scientific paper(s)
- Suggested books:
  - I. Koren, Computer Arithmetic Algorithms, A. K. Peters, Natick, MA, 2002. **Cheapest**
  - M. Ercegovac and T. Lang, Digital Arithmetic, Morgan Kaufmann Publishers - An Imprint of Elsevier Science, 2004. **Good coverage, out-of-print**
  - B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs, 2nd edition, Oxford University Press, New York, 2010. **Includes one of my methods**
  - P. Kornerup and D. W. Matula, Finite Precision Number Systems and Arithmetic, Cambridge, 2010. **Massive coverage, mathematical**

Course motivation

- `d <= a*b + c;` is a valid VHDL statement
- But how do we implement `d <= mod(a/b,c);`? (Which is also a valid statement, btw)
- Or how do we generate a sinusoid without storing all the sine values in a huge table?
- Focus will be on arithmetic algorithms rather than circuits, but when relevant this will also be discussed
- The aim is that you should get a thorough understanding of the arithmetic algorithms

Course contents

- Number representations (today)
- Addition/subtraction (Sems. 2+3)
- Multiplication (Sems. 4+5)
- Division and square root (Sems. 6+7)
- Floating-point arithmetic (Sems. 8+9)
- Elementary functions (Sems. 10+11)
- Alternative number systems (Sem. 12)
- Industrial example, guest lecture and summary (Sem. 13)
- *Assignment planning (Sem. 14)*
- *Assignment presentations (Sem. 15)*

Why bother?

- In 1991, during the Gulf war, an American Patriot Missile failed to take out an incoming Iraqi Scud Missile
- The Patriot measured the time in tenths of second, which was then multiplied with 1/10 to get the time in seconds
- The value of 1/10 was stored in a 24-bit register leading to an approximation error of about 0.000000095
- As the Patriot battery had been operative for about 100 hours, the total error amounted to about 0.34 seconds
- The Scud travels at 1676 m/s...

Why bother?

- In 1996, an Ariane 5 rocket exploded 40 seconds after take-off after $7 billion development
- The computer stored the horizontal speed relative to the platform as a 64-bit floating-point value
- This value was then converted to a 16-bit integer
- When the speed got above 32768 (whatever unit)...

Why bother?

- In 1982, the Vancouver stock exchange introduced a new index with an initial value of 1000.000
- After each transaction, the value was automatically updated
- 22 months later it had dropped to 520
- In fact, the correct value should have been 1098.892 given that rounding had been used instead of truncation...

Weighted positional number representations

- An integer X can be expressed in a weighted positional number representation using an ordered sequence of K digits $(x_{K-1}, x_{K-2}, \ldots, x_1, x_0)$ where

$$X = \sum_{i=0}^{K-1} x_i w_i \quad (1)$$

- Typically, we will denote numbers with upper case letters, i.e., X and the digits of X with lower case letters, normally indexed, i.e., x_i
- This number representation is *weighted* as each digit has a corresponding weight, w_i, and *positional* since the position of a digit in the sequence, i, determines the weight

Weighted positional number representations

- Let us denote the maximum and minimum values of X as $X_{\max}$ and $X_{\min}$
- If each number X between $X_{\max}$ and $X_{\min}$ has one and only one representation $(x_{K-1}, x_{K-2}, \ldots, x_1, x_0)$ the number representation is non-redundant
- The most common number representations are fixed-radix number representations, i.e., the ratio of the weights are constant and equal to the *radix*, denoted r

$$w_i = r^i, \; X = \sum_{i=0}^{K-1} x_i r^i \quad (2)$$

- In such a representation we will include the radix as $(x_{K-1}, x_{K-2}, \ldots, x_1, x_0)_r$

Weighted positional number representations

- If each digit can take on all integer values between $r-1$ and 0 it is possible to represent all numbers between the maximum and the minimum value in a unique way and, hence, the number representation is non-redundant
- If a digit can take on a value of r (or greater), the number representation is redundant since

$$x_i r^i = (x_i - r) r^i + r^{i+1} \tag{3}$$

so

$$(\dots, x_{i+1}, x_i, \dots)_r = (\dots, x_{i+1} + 1, x_i - r, \dots)_r \tag{4}$$

Weighted positional number representations

- It is also of interest to represent non-integer numbers
- This can easily be obtained by introducing weights smaller than one
- In the fixed-radix system this corresponds to negative radix exponents, i.e., $i < 0$
- The number is then often represented using an integer part and a fractional part with a radix-point in between $(x_{M-1}, x_{M-2}, \dots, x_1, x_0 . x_{-1}, x_{-2}, \dots, x_{-L+1}, x_{-L})_r$

$$X = \underbrace{\sum_{i=0}^{M-1} x_i r^i}_{\textit{integer part}} + \underbrace{\sum_{i=-L}^{-1} x_i r^i}_{\textit{fractional part}} = \sum_{i=-L}^{M-1} x_i r^i \tag{5}$$

where now $K = M + L$

Weighted positional number representations

- The position of the radix-point is in general not stored, instead it is implied to be in a fixed position between the M MSB digits and L LSB digits, hence a *fixed-point* representation
- The unit in the last position is denoted *ulp* and corresponds to the weight of the least significant digit, i.e., r^{-L} (sometimes also denoted Q, the quantization step)
- From an operation point of view the operands can be scaled with the same scaling factor, s
- Addition and subtraction gives a correct answer: $sX \pm sY = s(X \pm Y)$
- Multiplication and division must be corrected: $sX \times sY = s^2 X \times Y$, $\frac{sX}{sY} = \frac{X}{Y}$

Weighted positional number representations

- Common choices of radix
  - 10 – decimal
  - 2 – binary
  - 8 – octal
  - 16 – hexadecimal
- Not so common choices (anymore)
  - 12 – duodecimal (England, dozen, hours, ...)
  - 20 – vigesimal (Mesoamerica, evident in French, 80 = quatre-vingts = 4×20, Danish, 50 = halvtreds, 60 = tres, ...)
  - 60 – sexagesimal (Mesopotamia, seconds, minutes, ...)
  - 5 – used in the Gumatj dialect of Australian Aboriginal language Yolnu
  - 15 – used in the Huli language spoken in Papua New Guinea
  - 27 – used in the Telefol language spoken in Papua New Guinea

Examples

- $(1100000101)_2 = 2^9+2^8+2^2+2^0 = 512+256+4+1 = 773$
- $(12021)_3 = 1\cdot 3^4+2\cdot 3^3+2\cdot 3^1+1\cdot 3^0 = 81+54+6+1 = 142$
- $(4573)_8 =$
- $33 = (100001)_2 = (120)_3 = (201)_4 = (113)_5 = (53)_6 = (45)_7 = (41)_8 = (36)_9 = (33)_{10} = \cdots = (23)_{15} = (21)_{16} = \ldots$
- $55 = (\quad)_2 = (\quad)_4 = (\quad)_5 = (\quad)_{11} = (\quad)_{16}$

Radix conversion

- Radix conversion is performed on integer and fractional parts separately

Radix conversion – integers

- Radix-r to radix-10: evaluate expression using decimal numbers
- Radix-10 to radix-r: divide by r and use the remainder as least significant digit, divide the quotient by r and take the remainder as second least significant digit etc. until the quotient is zero
- Example: convert 321 to radix-6

Value	Quotient	Remainder	Digit
321	53	3	x_0
53	8	5	x_1
8	1	2	x_2
1	0	1	x_3

- Hence, $321 = (1253)_6$

Radix conversion – integers

- Radix-r to radix-r^k: group digits into clusters of k digits and replace each cluster with a radix-r^k digit
- Example: convert $(10001101)_2$ to radix-4

$$(10001101)_2 = (\underbrace{10}_{2}\underbrace{00}_{0}\underbrace{11}_{3}\underbrace{01}_{1})_2 = (2013)_4 \qquad (6)$$

- Radix-r^k to radix-r: Replace each digit with the corresponding k digit-representation using radix-r digits
- Example: convert $(745)_9$ to radix-3

$$(745)_9 = (\underbrace{7}_{21}\underbrace{4}_{11}\underbrace{5}_{12})_9 = (211112)_3 \qquad (7)$$

Radix conversion – fractions

- All integer values can be exactly converted from one integer radix to another
- However, this does not hold for fractional values
- Example:
  $\frac{1}{3} = (0.1)_3 = (0.3)_9 \approx (0.333\ldots)_{10} \approx (0.010101\ldots)_2$

Radix conversion – fractions

- Radix-r to radix-10: evaluate expression using decimal numbers
- Radix-10 to radix-r: multiply by r and use the integer part as most significant fractional digit, multiply the resulting fractional part by r and take the integer part as second most significant fractional digit etc. until the fractional part is zero (which may never happen)
- Example: convert 0.32 to radix-6

Value	Fractional part	Integer part	Digit
0.32	0.92	1	x_{-1}
0.92	0.52	5	x_{-2}
0.52	0.12	3	x_{-3}
0.12	0.72	0	x_{-4}
0.72	0.32	4	x_{-5}
0.32	0.92	1	x_{-6}
...			

- Hence, $0.32 = (0.15304\ldots)_6$

Negative numbers

- Two ways of representing negative numbers
- Using the sign ($+$ and $-$) and the magnitude
- Using complement representation
  - Radix complement
  - Diminished-radix complement

Sign-magnitude representation

- Use one digit to represent the sign
- Typically map $+$ to 0 and $-$ to $r-1$ (1 for binary)
- The most significant digit is not fully utilized for $r > 2$
- Two representations of 0
- Addition and subtraction of signed-magnitude numbers is complicated as the actual operation is determined by the signs and the magnitudes of the operand
- Example:
  - Add a positive X and a negative number $-Y$ as $X + (-Y)$
  - If $X > Y$ the result is $X - Y$ with positive sign while if $Y > X$ the result is $Y - X$ with negative sign

Complement representation

- In a complement representation a negative number $-X$ is represented by $C - X$ for some constant C
- This representation satisfies $-(-X) = X$ as $C - (C - X) = X$
- Consider the previous example $X + (-Y)$
  - As $-Y$ is represented by $C - Y$ we get $X + (C - Y) = C - (Y - X)$
  - If $Y > X$ the correct answer, $-(Y - X)$, is already represented
  - If $X > Y$ the result is $C + X - Y$ and the additional C must be discarded
- How should C be selected such that it can be easily discarded and also such that $C - Y$ can be easily computed from Y?

Complement representation

- Define the complement of a digit as

$$\bar{x}_i = (r-1) - x_i \tag{8}$$

and for a number (ordered sequence) as

$$\bar{X} = (\bar{x}_{M-1}, \bar{x}_{M-2}, \ldots, \bar{x}_{-L+1}, \bar{x}_{-L})_r \tag{9}$$

- Two options for selecting C:
  - Compute $C - X$ by complementing (diminished-radix complement): $C - X = \bar{X} = \underbrace{r^M - ulp}_{C} - X$
  - Select $C = r^M$ which corresponds to the weight of the digit more significant than the most significant digit (radix complement): $C - X = r^k - X = \bar{X} + ulp$

Complement representation

- For the radix-r case:
  - Radix complement = r's complement
    - Negate by complementing each digit and add *ulp*
  - Diminished-radix complement = $r-1$'s complement
    - Negate by complementing each digit
  - Value of a representation $X = (x_{M-1}, x_{M-2}, \ldots, x_{-L+1}, x_{-L})_r$:

$$\begin{cases} -C + \sum x_i r^i & x_{M-1} \geq \frac{r}{2} \\ \sum x_i r^i & \text{otherwise} \end{cases}$$

- For the binary case:
  - Radix complement = two's complement
  - Diminished-radix complement = one's complement

Binary representations

- To illustrate the different number representations we will consider the following properties and operations
  - Representation and numerical range
  - Increasing the number of bits on the least significant side (left shift) and on the most significant side (right shift/sign-extension)
  - Negation

Sign-magnitude

- Use one bit to represent the sign and the rest of the bits for the magnitude

$$X = (1 - 2x_0) \sum_{i=-L}^{M-2} x_i 2^i = (-1)^{x_0} \sum_{i=-L}^{M-2} x_i 2^i \qquad (10)$$

- Numerical range $-2^{M-1} + ulp \leq X \leq 2^{M-1} - ulp$
- Examples
  - $0 = 00.000$ or 10.000
  - $\frac{13}{8} = 01.101$
  - $-\frac{13}{8} = 11.101$
- Increasing word length on MSB side (sign extend)/right shift
  - Add zeros after sign bit
  - $\frac{13}{8} = 001.101$
  - $-\frac{13}{8} = 101.101$
  - $\frac{13}{16} = 0.1101$
  - $-\frac{13}{16} = 1.1101$

Sign-magnitude

- Increasing word length on LSB side/left shift
  - Add zeros after the LSB
  - $\frac{13}{8} = 01.1010$
  - $-\frac{13}{8} = 11.1010$
  - $\frac{13}{4} = 011.010$
  - $-\frac{13}{4} = 111.010$
- Negation
  - Invert sign-bit

One's complement

- Negate by inverting all bits

$$X = -x_0(2^{M-1} - ulp) + \sum_{i=-L}^{M-2} x_i 2^i \qquad (11)$$

- Numerical range $-2^{M-1} + ulp \leq X \leq 2^{M-1} - ulp$
- Examples
  - $0 = 00.000$ or 11.111
  - $\frac{13}{8} = 01.101$
  - $-\frac{13}{8} = 10.010$
- Increasing word length on MSB side (sign extend)/right shift
  - Add sign-bit before the MSB
  - $\frac{13}{8} = 001.101$
  - $-\frac{13}{8} = 110.010$
  - $\frac{13}{16} = 0.1101$
  - $-\frac{13}{16} = 1.0010$

One's complement

- Increasing word length on LSB side/left shift
  - Add sign-bit after the LSB
  - $\frac{13}{8} = 01.1010$
  - $-\frac{13}{8} = 10.0101$
  - $\frac{13}{4} = 011.010$
  - $-\frac{13}{4} = 100.101$
- Negation
  - Invert all bits

Two's complement

$$X = -x_0 2^{M-1} + \sum_{i=-L}^{M-2} x_i 2^i \qquad (12)$$

- Numerical range $-2^{M-1} \leq X \leq 2^{M-1} - ulp$
- Examples
  - $0 = 0.000$
  - $\frac{13}{8} = 01.101$
  - $-\frac{13}{8} = 10.011$
- Increasing word length on MSB side (sign extend)/right shift
  - Add sign-bit before the MSB
  - $\frac{13}{8} = 001.101$
  - $-\frac{13}{8} = 110.011$
  - $\frac{13}{16} = 0.1101$
  - $-\frac{13}{16} = 1.0011$

Two's complement

- Increasing word length on LSB side/left shift
  - Add zeros after the LSB
  - $\frac{13}{8} = 01.1010$
  - $-\frac{13}{8} = 10.0110$
  - $\frac{13}{4} = 011.010$
  - $-\frac{13}{4} = 100.110$
- Negation
  - Invert all bits and add one at the LSB position

Negative radix

- To avoid the complement representations one suggestion is to use a negative-radix system, i.e., $r = -\beta$ where β is a positive integer

$$X = \sum_{i=L}^{M-1} x_i (-\beta)^i \qquad (13)$$

- This number representation has some interesting properties, but suffers from an unbalanced range with typically a factor β difference in the number of positive and negative numbers

Examples

- Express -54 in radix and diminished-radix complement radix-5 number systems, respectively
- Express -35 in a nega-binary, i.e., radix -2, number system