Programming 2 (SS 2023)
Saarland University
Faculty MI
Compiler Design Lab

Sample Solution 1
Arithmetic

Prof. Dr. Sebastian Hack
Pascal Lauer, M. Sc.
Marcel Ullrich, B. Sc.

The calendar indicates which script chapters you should study in conjuction with each lecture. The exercises are designed to enhance your understanding of the lecture material and prepare you for the mini-tests and the final exam. Additional exercises can be found at the end of each chapter in the script.
The difficulty of an exercise on the sheet is determined by the number of annotated 'X' and 'O' marks in the tic-tac-toe field, with four levels (1-4) increasing by one mark per level.

Exercise 1.1:

Represent the following numbers in the respectively specified number system:

(a) $\langle 43\rangle_{10}$ into the base 7 system

(b) $\langle 212\rangle_{3}$ into the base 9 system

(c) $\langle 313\rangle_{4}$ into the base 6 system

(d) $\langle 3142\rangle_{5}$ into the base 3 system

(e) $\langle 621\rangle_{11}$ into the base 5 system

Solution

(a) $\langle 43\rangle_{10} = 6\times 7 + 1\times 1 = \langle 61\rangle_7$

(b) $\langle 212\rangle_3 = 2\times 9 + 1\times 3 + 2\times 1 = \langle 23\rangle_{10} = 2\times 9 + 5\times 1 = \langle 25\rangle_9$

(c) $\langle 313\rangle_4 = 3\times 16 + 1\times 4 + 3\times 1 = \langle 55\rangle_{10} = 1\times 36 + 3\times 6 + 1\times 1 = \langle 131\rangle_6$

(d) $\langle 3142\rangle_5 = 3\times125+1\times25+4\times5+2\times1 = \langle 422\rangle_{10} = 1\times243+2\times81+0\times27+1\times9+2\times3+2\times1 = \langle 120122\rangle_3$

(e) $\langle 621\rangle_{11} = 6\times 121 + 2\times 11 + 1\times 1 = \langle 749\rangle_{10} = 1\times 625 + 0\times 125 + 4\times 25 + 4\times 5 + 4\times 1 = \langle 10444\rangle_5$

Exercise 1.2:

We consider the following positional number systems:

Name	Base	Prefix	Digits
Binary System	2	0b	0,1
Octal System	8	0	0,...,7
Hexadecimal System	16	0x	0,...,9,A,B,C,D,E,F

The digits increase in value from left to right, with 0 assigned the lowest value and each subsequent digit assigned a value one higher than the previous. E.g. `F` corresponds to 15.
To specify the number system, a number is prefixed by the content of the column "Prefix". For example, `013` is an octal number, `0x13` is a hexadecimal number and `13` is a decimal number.

(a) Represent the following decimal numbers in each of the three number systems. Use pen and paper. Do not use a calculator or computer.

5   16   49   81   257   317   1721   4096

(b) Can you explain your conversion steps in a generic way that can be used as a guide for other students?

(c) Assume that the function $dig : [0,15] \rightarrow \{0,\dots,9,A,\dots,F\}$ computes the hexadecimal digit for any decimal number $n \in [0,15]$. Complete the following (recursive) definition of a function $\rangle \cdot \langle_b$ to compute any natural number in base $b \leq 16$. E.g. $\rangle 257 \langle_8 = 401$.

$$\mathbb{N} \rightarrow \{0,\dots,9,A,\dots,F\}^+$$

$$\rangle n \langle_b = \begin{cases} & \text{if} \\ & \text{else} \end{cases}$$

Solution

(a) Table:

10	**2**	**8**	**16**
5	0b101	05	0x5
16	0b10000	020	0x10
49	0b110001	061	0x31
81	0b1010001	0121	0x51
257	0b100000001	0401	0x101
317	0b100111101	0475	0x13D
1721	0b11010111001	03271	0x6B9
4096	0b1000000000000	010000	0x1000

(b) Translation of a number n into base b using the example $\langle 401 \rangle_8 = 257$

i. Note all $b^k < n$

$$8^2 \quad 8^1 \quad 8^0$$

ii. Starting on the left, determine the maximal a such that $a \cdot b^k < n$. Compute n' where $n' = n - (a \cdot b^k)$. Here we have $n' = 1$.

$$\begin{matrix} 8^2 & 8^1 & 8^0 \\ 4 & & \end{matrix}$$

iii. Repeat the previous step from left to right until all digits are computed:

$$\begin{matrix} 8^2 & 8^1 & 8^0 \\ 4 & 0 & 1 \end{matrix}$$

(c) Idea: The inverse of the tree example from the lecture notes. In every recursion step until $n < b$ holds: $n \bmod b$ is the index of the child in the currently considered subtree. The factor $\frac{n}{b}$ ensures that the algorithm descends one layer in every step.

$$\rangle n \langle_b = \begin{cases} \rangle \lfloor \frac{n}{b} \rfloor \langle_b \cdot dig(n \bmod b) & \text{if} \\ dig(n) & \text{else} \end{cases}$$

Exercise 1.3:

Fill in the table:

binary $\langle\cdot\rangle_2$	**octal** $\langle\cdot\rangle_8$	**decimal** $\langle\cdot\rangle_{10}$	**hexadecimal** $\langle\cdot\rangle_{16}$
			$\langle 21\rangle_{16}$
$\langle 1101\rangle_2$			
	$\langle 103\rangle_8$		
$\langle 1011\rangle_2$			
			$\langle 2D\rangle_{16}$
	$\langle 417\rangle_8$		
			$\langle 17\rangle_{16}$
$\langle 111101\rangle_2$			
	$\langle 315\rangle_8$		

Solution

binary $\langle\cdot\rangle_2$	**octal** $\langle\cdot\rangle_8$	**decimal** $\langle\cdot\rangle_{10}$	**hexadecimal** $\langle\cdot\rangle_{16}$
$\langle 100001\rangle_2$	$\langle 41\rangle_8$	$\langle 33\rangle_{10}$	$\langle 21\rangle_{16}$
$\langle 1101\rangle_2$	$\langle 15\rangle_8$	$\langle 13\rangle_{10}$	$\langle D\rangle_{16}$
$\langle 1000011\rangle_2$	$\langle 103\rangle_8$	$\langle 67\rangle_{10}$	$\langle 43\rangle_{16}$
$\langle 1011\rangle_2$	$\langle 13\rangle_8$	$\langle 11\rangle_{10}$	$\langle B\rangle_{16}$
$\langle 101101\rangle_2$	$\langle 55\rangle_8$	$\langle 45\rangle_{10}$	$\langle 2D\rangle_{16}$
$\langle 100001111\rangle_2$	$\langle 417\rangle_8$	$\langle 271\rangle_{10}$	$\langle 10F\rangle_{16}$
$\langle 10111\rangle_2$	$\langle 27\rangle_8$	$\langle 23\rangle_{10}$	$\langle 17\rangle_{16}$
$\langle 111101\rangle_2$	$\langle 75\rangle_8$	$\langle 61\rangle_{10}$	$\langle 3D\rangle_{16}$
$\langle 11001101\rangle_2$	$\langle 315\rangle_8$	$\langle 205\rangle_{10}$	$\langle CD\rangle_{16}$

Exercise 1.4:

In this exercise we use $\langle\cdot\rangle$ instead of $\langle\cdot\rangle_2$.

(a) State the decimal numbers represented as:

i. $\langle 11\rangle$
ii. $\langle 011\rangle$
iii. $\langle 1001\rangle$
iv. $\langle 1111\rangle$
v. $\langle 101111\rangle$

(b) Compute the sum and represent the result in the binary system:

i. $\langle 1011\rangle + \langle 1101\rangle$
ii. $\langle 11\rangle + \langle 1101\rangle$
iii. $\langle 10111\rangle + \langle 1101\rangle$
iv. $\langle 1011 + 1101\rangle$

Solution

(a)
  i. $\langle 11 \rangle = 3$
  ii. $\langle 011 \rangle = 3$
  iii. $\langle 1001 \rangle = 9$
  iv. $\langle 1111 \rangle = 15$
  v. $\langle 101111 \rangle = 47$

(b)
  i. $\langle 1011 \rangle + \langle 1101 \rangle = 11 + 13 = 24 = \langle 11000 \rangle$
  ii. $\langle 11 \rangle + \langle 1101 \rangle = 3 + 13 = 16 = \langle 10000 \rangle$
  iii. $\langle 10111 \rangle + \langle 1101 \rangle = 23 + 13 = 36 = \langle 100100 \rangle$
  iv. $\langle 1011 + 1101 \rangle = \langle 11000 \rangle = 24 = \langle 11000 \rangle$

Exercise 1.5:

(a) Show that $\hat{}$ and | are associative using a truth table.

(b) We say that an operation $\circ$ distributes over an operation $\circ'$ if:

$$a \circ (b \circ' c) = (a \circ b) \circ' (a \circ c)$$

An example over the natural numbers is that multiplication distributes over addition. I.e. $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$. Determine which operations $\circ$ distrubutes over $\circ'$ using the operations $\circ, \circ' \in \{\&, |, \hat{}\,\}$ from (a) for $a, b, c \in \mathbb{B}$.

Solution

(a)

a	b	c	$(b \mathbin{\hat{}} c)$	$a \mathbin{\hat{}} (b \mathbin{\hat{}} c)$	$(a \mathbin{\hat{}} b)$	$(a \mathbin{\hat{}} b) \mathbin{\hat{}} c$
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	1	1	1
0	1	1	0	0	1	0
1	0	0	0	1	1	1
1	0	1	1	0	1	0
1	1	0	1	0	0	0
1	1	1	0	1	0	1

a	b	c	$(b \mid c)$	$a \mid (b \mid c)$	$(a \mid b)$	$(a \mid b) \mid c$
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	0	1	1	1
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1

(b) The following distributive laws hold:

- $a \mid (b \,\&\, c) = (a \mid b) \,\&\, (a \mid c)$
- $a \,\&\, (b \mid c) = (a \,\&\, b) \mid (a \,\&\, c)$
- $a \,\&\, (b \mathbin{\hat{}} c) = (a \,\&\, b) \mathbin{\hat{}} (a \,\&\, c)$

Exercise 1.6:

We consider & (and), | (or), and ˆ (xor) to be the elementary logical operations. However, it would be possible to define other basic operations that can be used to express these three operations. One such operation is the nand (not-and), which is defined as the negation of the and operation $\overline{a \,\&\, b}$. That is, nand is true (or has a value of 1) unless both inputs are true (have a value of 1). Define and, or, xor using nand.

Bonus: Can you create or name other elementary operations?

Solution

a	b	$\overline{a \,\&\, b}$
0	0	1
0	1	1
1	0	1
1	1	0

We can express negation using nand:

a	$\overline{a \,\&\, a}$	$\overline{a}$
0	1	1
1	0	0

We can express conjunction (and) using nand. We can use plain old negation, since we already know how to encode it just using nand.

a	b	$\overline{\overline{a \,\&\, b}}$	$a \,\&\, b$
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	1

We can now express disjunction (or) using nand. Again, we also use negation and conjunction since we know how to express them.

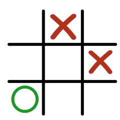
a	b	$\overline{a}$	$\overline{b}$	$\overline{\overline{a} \,\&\, \overline{b}}$	$a \mid b$
0	0	1	1	0	0
0	1	1	0	1	1
1	0	0	1	1	1
1	1	0	0	1	1

We have already shown that we can express and and or with nand. So it is enough to show that xor can be expressed with and and or to prove that we can only express it with nand.

a	b	$\overline{a}$	$\overline{b}$	$\overline{a} \,\&\, b$	$a \,\&\, \overline{b}$	$(\overline{a} \,\&\, b) \mid (a \,\&\, \overline{b})$	$a \,\hat{}\, b$
0	0	1	1	0	0	0	0
0	1	1	0	1	0	1	1
1	0	0	1	0	1	1	1
1	1	0	0	0	0	0	0

Another elementary operation like nand is nor. It is 1 if $a = b = 0$ and 0 otherwise.

Exercise 1.7:



A boolean function $f : \mathbb{B}^2 \to \mathbb{B}$ is called universal if it is possible to represent any other boolean function $f' : \mathbb{B}^2 \to \mathbb{B}$ without the need for additional operations. Refer to the definition of nand from exercise 1.6. Show that nand is universal.

Hint: First, show that any boolean function $f' : \mathbb{B}^2 \to \mathbb{B}$ can be represented using and, not and or. Then use the other exercise to relate this to nand. It may help to prove that a the negation can be expressed using nand.

Solution

In total, there are 16 different possibilities for boolean functions with two arguments: Since $f(a,b)$ takes two arguments, each of which can be either 0 or 1, there are exactly 4 possible different inputs. For each of these inputs there are 2 different possible results: 0 and 1. This means in total we can find $2^4 = 16$ functions. All of functions can be represented only using and, or, and negation, as follows:

$f(0,0)$	0	1	0	1	0	1	0	1
$f(0,1)$	0	0	1	1	0	0	1	1
$f(1,0)$	0	0	0	0	1	1	1	1
$f(1,1)$	0	0	0	0	0	0	0	0
$f(a,b)$	$a \,\&\, \overline{a}$	$\overline{a} \,\&\, \overline{b}$	$\overline{a} \,\&\, b$	$\overline{a}$	$a \,\&\, \overline{b}$	$\overline{b}$	$(a \,\&\, \overline{b}) \mid (b \,\&\, \overline{a})$	$\overline{a} \mid \overline{b}$

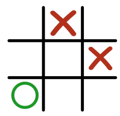
$f(0,0)$	0	1	0	1	0	1	0	1
$f(0,1)$	0	0	1	1	0	0	1	1
$f(1,0)$	0	0	0	0	1	1	1	1
$f(1,1)$	1	1	1	1	1	1	1	1
$f(a,b)$	$a \,\&\, b$	$(a \,\&\, b) \mid (\overline{a} \,\&\, \overline{b})$	b	$b \mid \overline{a}$	a	$a \mid \overline{b}$	$a \mid b$	$a \mid \overline{a}$

Now that we have shown that every boolean function with two arguments can be represented with and, not, and or and that we have shown in exercise 6 that and and or can be expressed with nand, the only thing left to show is that not can also be expressed using only nand.

a	$\overline{a}$	$\overline{a \,\&\, a}$
0	1	1
1	0	0

In conclusion, we have now shown that and, or and not can be expressed using only nand, because we have also shown that every boolean function with two arguments can be represented by and, or and not we can conclude that nand is universal.

Exercise 1.8:



(a) Let x be a sequence of digits of length n and let y be a sequence of digits of length m. Show that $\langle x \cdot y\rangle_\beta = \langle x\rangle_\beta \cdot \beta^m + \langle y\rangle_\beta$ holds for any $\beta \in \mathbb{N}^*$.

Hint: Recall that $\cdot$ acts as concatenation on sequences of digit. E.g. $ab \cdot cd = abcd$.

(b) Show that for all $n > 0$, $\langle(\beta-1)^n\rangle_\beta + 1 = \beta^n$.

(c) Let x be a sequence of digits of length n. Show that $\langle x\rangle_\beta < \beta^n$.

Solution

(a)

$$\begin{aligned}
\langle x \cdot y\rangle_\beta &= \sum_{i=0}^{n-1} x_i \beta^{m+i} + \sum_{i=0}^{m-1} y_i \beta^i \\
&= \beta^m \cdot \sum_{i=0}^{n-1} x_i \beta^i + \sum_{i=0}^{m-1} y_i \beta^i \\
&= \beta^m \cdot \langle x\rangle + \langle y\rangle \qquad \square
\end{aligned}$$

(b)

$$
\begin{aligned}
\langle(\beta-1)^n\rangle_\beta + 1 &= \sum_{i=0}^{n-1}(\beta-1)\beta^i + 1\\
&= (\beta-1)\beta^{n-1} + (\beta-1)\beta^{n-2} + \cdots + (\beta-1)\beta + (\beta-1) + 1\\
&= \beta^n - \underbrace{\beta^{n-1} + \beta^{n-1}}_{0} + \cdots \underbrace{-\beta + \beta}_{0} \underbrace{-1 + 1}_{0}\\
&= \beta^n \qquad\square
\end{aligned}
$$

(c) Using induction:

$n = 1$:

$$x \leq \beta - 1 < \beta^1.$$

$n \to n+1$: Induction hypothesis: $\langle x_{n-1} \cdots x_0\rangle_\beta < \beta^n$

$$
\begin{aligned}
\langle x_n \cdots x_0\rangle_\beta &= x_n \cdot \beta^n + \langle x_{n-1} \cdots x_0\rangle_\beta\\
&< x_n \cdot \beta^n + \beta^n && \mid \text{induction hypothesis}\\
&\leq (\beta-1)\cdot\beta^n + \beta^n && \mid x_n \leq \beta - 1\\
&= \beta^{n+1} - \beta^n + \beta^n\\
&= \beta^{n+1} \qquad\square
\end{aligned}
$$