1. (6%) (a) Construct the truth-table for the Boolean function given by the formula
\[ -((A \rightarrow \neg B) \land (C \rightarrow A)) \] .

(b) Find a Boolean circuit with AND, OR and NOT gates only that computes the Boolean function above and contains as few gates as possible.

(c) Determine whether the formula in (a) is equivalent to the formula
\[ (B \rightarrow \neg A) \rightarrow (C \land \neg A) \].

Show your working.

**Answer:**
(a) Denote the given formula by \( F \). The truth-table for \( F \) is as follows:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

(b) It follows that \( F \equiv (\neg A \land C) \lor (A \land B) \), which gives a Boolean circuit with four gates.

(c) The formula in (c) has the same truth-table as \( F \), and so these two formulas are equivalent.

2. (3%) Design a Boolean circuit for the Boolean function that checks whether a 3-bit two’s complement binary number is less than \(-1\) (hint: construct the truth-table for the input bits \( A_2, A_1, A_0 \)).

**Answer:**

<table>
<thead>
<tr>
<th>( A_2 )</th>
<th>( A_1 )</th>
<th>( A_0 )</th>
<th>less than (-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
This can be expressed by the formula $A_2 \land \neg(A_1 \land A_0)$, which gives the required Boolean circuit.

3. (5%) Formalise the following argument in Boolean logic, and decide whether it is correct or not. Explain your answer.

If Jones did not meet Smith last night then either Smith was the murderer or Jones is lying. If Smith was not the murderer then Jones did not meet Smith last night and the murder took place after midnight. If the murder took place after midnight then either Smith was the murderer or Jones is lying. Therefore, Smith was the murderer.

(Suggestion: use the propositional variable $A$ for ‘Jones did not meet Smith last night’, $B$ for ‘Smith was the murderer’, $C$ for ‘Jones is lying’, and $D$ for ‘The murder took place after midnight’.)

**Answer:** The argument is formalised as follows:

If $A \rightarrow B \lor C$, $\neg B \rightarrow A \land D$, $D \rightarrow B \lor C$ are all true, then $B$ is also true.

The argument is not correct. To show this, we should make $B = 0$ and the three premises true. Set $A = 1$, $B = 0$, $C = 1$ and $D = 1$. It is easy to check that in this case we have

$$(A \rightarrow B \lor C) = 1, \quad (\neg B \rightarrow A \land D) = 1, \quad (D \rightarrow B \lor C) = 1$$

as required.

4. (6%) Given the machine 32-bit word

1100 0001 1011 0000 0000 0000 0000 0000

find the decimal number represented by this word assuming that it is

(a) a two’s complement integer;
(b) an unsigned integer;
(c) a single precision IEEE 754 floating-point number.

**Answer:** Two’s complement: $-1 \times 2^{31} + 1 \times 2^{30} + 1 \times 2^{24} + 1 \times 2^{23} + 1 \times 2^{21} + 1 \times 2^{20}$.

Unsigned: $1 \times 2^{31} + 1 \times 2^{30} + 1 \times 2^{24} + 1 \times 2^{23} + 1 \times 2^{21} + 1 \times 2^{20}$.

IEEE 754 floating-point: $(-1)^1 \times 1.0112 \times 2^4 = -101102 = -22_{10}$.

5. (6%) Find computer representations of the following numbers:

(a) $-107$ as a two’s complement 32-bit binary number;
(b) $-107$ as an IEEE 754 32-bit floating-point number;
(c) $-14.375$ as an IEEE 754 32-bit floating-point number.
**Answer:** −107 as a two’s complement 32-bit binary number: \(107_{10} = 1101011_2\); now, take Boolean negation and add 1, which gives:

\[
1111
definition of binary, integer, real, and complex numbers.

**Answer:** In each case, there can be many solutions, not just the given one.

(a) \(f(n) = n + 1\).

(b) \(f(n) = \begin{cases} n - 1, & \text{if } n > 0, \\ 0, & \text{if } n = 0. \end{cases}\)

(c) \(f(n) = \begin{cases} 0, & \text{if } n = 0, \\ n - 1, & \text{if } n > 0 \text{ and is even}, \\ n + 1, & \text{if } n \text{ is odd}. \end{cases}\)

(d) \(f(n) = 42\).

7. (6%) For each of the following functions, determine whether or not it is one-to-one and whether or not it is onto:

(a) \(f: \mathbb{R} \to \mathbb{R}\) given by \(f(x) = (x + 1)/(x + 2)\);

(b) \(f: \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}\) given by \(f(m, n) = m + n + 1\);

(c) \(f: \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}\) given by \(f(m, n) = |m| - |n|\).

**Answer:** (a) \(f\) is not a function as \(f(-2)\) is not defined. (b) \(f\) is onto but not one-to-one because \(f(m, n) = f(n, m)\). (c) \(f\) is onto but not one-to-one because \(f(m, n) = f(-m, -n)\).

8. (8%) (a) Draw an undirected graph represented by the following adjacency matrix:

\[
\begin{bmatrix}
1 & 2 & 0 & 1 \\
2 & 0 & 3 & 0 \\
0 & 3 & 1 & 1 \\
1 & 0 & 1 & 0
\end{bmatrix}
\]
Is the graph simple? Explain your answer.

(b) Determine whether the following graphs are isomorphic or not and explain your answer:

(c) Determine whether the following graphs are isomorphic or not and explain your answer:

Answer: (a)

The graph is not simple, as it contains multiple and loop edges.

(b) The graphs are isomorphic with the following map being an isomorphism: \( h(3) = b, \ h(1) = a, \ h(2) = c, \ h(4) = d, \ h(5) = e. \)

(c) These graphs are not isomorphic. Indeed, consider the invariant ‘containing two \( K_3 \)s as disjoint subgraphs’. Then the first graph does not have this property, while the second does.

9. (10%) Consider the following NFA:

(a) Give all the computations of the automaton on the input strings \( bb, ab, aba, \) and \( \varepsilon, \) and determine if the strings are accepted.
(b) Transform the automaton, using the subset construction, into an equivalent deterministic finite automaton and remove the unreachable states. Show your working.

(c) Describe the language of the automaton in English.

(d) Describe the language of the automaton by means of a regular expression.

(e) Describe the language of the automaton by means of a context-free grammar.

Answer: (a) 
\[(s, bb), (q, b), (s, bb), (p, bb), (q, b) \] rejected 
\[(s, ab), (s, b), (q, \varepsilon) \] accepted 
\[(s, aba), (s, ba), (q, a), (q, \varepsilon) \] accepted 
\[(s, \varepsilon) \] rejected 

(b) The construction is shown below:

After removing the unreachable states, we obtain:

(c) The language consists of those words over the alphabet \(\{a, b\}\) that contain at least one \(b\) but do not contain two consecutive \(b\)s.

(d) Regular expression: \(a^*b(a \cup ab)^*\)

(e) \(S \to AbC, \ A \to \varepsilon, \ A \to Aa, \ C \to \varepsilon, \ C \to Ca, \ C \to Cab\)

10. (5%) Design a (deterministic or nondeterministic) finite automaton \(A\) such that \(L(A)\) consists of all words over the alphabet \(\{0, 1\}\) that contain at least two 0’s and do not end with 11. Find a regular expression representing the language \(L(A)\).

Answer: (a) DFA:
(b) Regular expression: $1^*01^*(0 \cup 1)^*(0 \cup 01)$.

11. (8%) Convert the regular language $L[ab((c\cup de)ab^*)^c]$ to a finite automaton accepting it.

Answer:

![Diagram](image)

12. (8%) (a) Give a context-free grammar for the language over the alphabet \{a, b\} containing all words of even length. Show the derivation of $abba$ in your grammar.

(b) Give a context-free grammar for the language over the alphabet \{a, b\} containing all words with at most three a’s.

(c) What is the language defined by the following context-free grammar?

\[
S \rightarrow S_1, \ S \rightarrow S_2, \ S_1 \rightarrow 0S_1, \ S_1 \rightarrow 0A, \ S_2 \rightarrow S_21, \ S_2 \rightarrow A1, \ A \rightarrow \varepsilon, \ A \rightarrow 0A1
\]

Is this language regular? Give an informal explanation of your answer.

Answer: (a) The grammar $S \rightarrow aT, \ S \rightarrow bT, \ T \rightarrow \varepsilon, \ T \rightarrow aS, \ T \rightarrow bS$ with the start variable $S$ (respectively, $T$) gives all words of odd (respectively, even) length. Derivation: $T \Rightarrow aS \Rightarrow abT \Rightarrow abbS \Rightarrow abbaT \Rightarrow abba\varepsilon$.

(b) Context-free grammar:

$S \rightarrow B, \ S \rightarrow BaB, \ S \rightarrow BaBaB, \ S \rightarrow BaBaBaB,$

$B \rightarrow \varepsilon, \ B \rightarrow bB$

(c) $\{0^n1^m \mid n \neq m\}$. This language is not regular, which can be shown using the pumping lemma.

13. (8%) Construct a context free grammar and a pushdown automaton for the language of words over the alphabet \{0, 1\} that start and end with the same symbol and have the same number of 0s as 1s.

Answer: We can modify the CFG and PDA on page 14, Lecture 7. Context-free grammar:

$S \rightarrow 0L1L10, \ S \rightarrow 1L0L0L1, \ L \rightarrow \varepsilon, \ L \rightarrow 0L1, \ L \rightarrow 1L0, \ L \rightarrow LL$. The following PDA is a more or less straightforward modification of the PDA on page 14, Lecture 7 (there exist more elegant solutions):
14. (5%) Consider the following transition table of a Turing machine (with \( s \) being its initial state):

\[
\begin{array}{|c|c|c|}
\hline
s & 0 & h & 0 \\
\hline
s & 1 & q & \rightarrow \\
\hline
s & \downarrow & s & \rightarrow \\
\hline
s & \downarrow & s & \rightarrow \\
\hline
q & 0 & q & \rightarrow \\
\hline
q & 1 & q & \rightarrow \\
\hline
q & \downarrow & p & \leftarrow \\
\hline
q & \downarrow & q & \rightarrow \\
\hline
p & 0 & p & \rightarrow \\
\hline
p & 1 & h & 0 \\
\hline
p & \downarrow & h & 0 \\
\hline
p & \downarrow & p & \rightarrow \\
\hline
\end{array}
\]

(i) Give the computations of the machine on inputs 10, 111 and 110.
(ii) Describe in English what this Turing machine does.

Answer:

(i) \(- (s, \rightarrow 10), (q, \rightarrow 10), (q, \rightarrow 10\omega), (p, \rightarrow 10), (p, \rightarrow 10\omega), (h, \rightarrow 100)\)

\(- (s, \rightarrow 111), (q, \rightarrow 111), (q, \rightarrow 111\omega), (p, \rightarrow 111), (h, \rightarrow 110)\)

\(- (s, \rightarrow 110), (q, \rightarrow 110), (q, \rightarrow 110\omega), (p, \rightarrow 110), (p, \rightarrow 110\omega), (h, \rightarrow 1100)\)

(ii) The machine computes the \(\mathbb{N} \rightarrow \mathbb{N}\) function

\[ f(n) = \begin{cases} 
  n - 1 & \text{if } n \text{ is odd}, \\
  2n & \text{if } n \text{ is even}.
\end{cases} \]

Or: If the input word starts with 0, then the machine immediately halts. If the input word starts with \(\omega\), then the machine never terminates. In any other case, the machine scans the tape to the right until it finds the first \(\omega\). If the symbol in the previous cell is 0, then writes a 0 to the blank cell following it and halts. If the symbol in the previous cell is 1, then changes it to 0 and halts.

15. (10%) Consider the following \(\mathbb{N} \rightarrow \mathbb{N}\) function \(f\):

\[ f(n) = \begin{cases} 
  2n + 1 & \text{if } n \text{ is even}, \\
  2n - 2 & \text{if } n \text{ is odd}.
\end{cases} \]

(Do not forget that all numbers are represented in binary.)

(i) Explain what it means to say that a Turing machine computes this function \(f\).

(ii) Give an implementation level description in English of a Turing machine that computes this \(f\).

(iii) Give the complete transition table of this Turing machine.

(iv) Give the computations of your Turing machine on inputs 0, 11 and 100.

Answer:

(i) Given any natural number \(n\) in binary as input, it should be written to the left end of the tape, the rest of the tape is blank, the head scans the leftmost symbol of the binary code of \(n\). The machine halts on every natural number \(n\), and when halts, the binary code of \(f(n)\) is written to the left end of the tape, the rest of the tape is blank. In particular,

\(- \text{ if } n = 0 \text{ then it is changed to 1;}
\)

\(- \text{ if } n > 0 \text{ is even (that is, its last digit is 0) then a digit } 1 \text{ is written at the end of the input;}
\)

\(- \text{ if } n \text{ is odd (that is, its last digit is 1) then (since } 2n - 2 = 2 \cdot (n - 1)\) \text{ this last digit is changed to 0 and a digit } 0 \text{ is written at the end of the input.}
(ii) If a number is even, then the last digit of its binary code is 0. And if a number is odd, then the last digit of its binary code is 1. Computing $2n + 1$ in binary (for even $n$’s) means writing 1 at the end of the input string. Computing $2n - 2$ in binary (for odd $n$’s) means replacing the last digit (a 1) with 0. So an implementation-level description of a Turing machine computing $f$ is as follows:

1. If the first digit of the input is 0 (meaning that the input number is 0) then it halts and writes 1 (as 0 is even and $0/2 = 0$).
2. If the first digit of the input is 1, then the head goes to the right end of the input string (by searching for the first blank). When it reaches the first blank, moves the head one cell back to the left.
3. It checks the content of this cell (the last bit of the input): If it is 0, then moves the head one cell to the right, writes a 1 and halts. If it is 1, then replaces it with 0 and then halts.

(iii) There are many possible solutions, here is one:

\[
\begin{array}{ccc}
 s & 0 & h & 1 \\
 s & 1 & t & \rightarrow \\
 t & 0 & t & \rightarrow \\
 t & 1 & t & \rightarrow \\
 t & \rightarrow & p & \leftarrow \\
 p & 0 & q & \rightarrow \\
 q & \rightarrow & h & 1 \\
 p & 1 & r & 0 \\
 r & 0 & r & \rightarrow \\
 r & \rightarrow & h & 0 \\
\end{array}
\]

The other transitions are arbitrary (of course, $(\ldots, \leftarrow)$ should always go to $(\ldots, \rightarrow)$).

(iv) Here the solutions of course depend on the machine you gave in (iii):

- $(s, \rightarrow 0), (h, \rightarrow 1)$
- $(s, \rightarrow 11), (t, \rightarrow 11), (t, \rightarrow 11\_\rightarrow), (p, \rightarrow 11), (r, \rightarrow 10), (r, \rightarrow 11\_\leftarrow), (h, \rightarrow 100)$
- $(s, \rightarrow 100), (t, \rightarrow 100), (t, \rightarrow 100\_\rightarrow), (p, \rightarrow 100), (q, 100\_\leftarrow), (h, \rightarrow 1001)$