Name: Matriculation Number:

General CS II (320201) Midterm Exam
March 29. 2005

You have one hour(sharp) for the test;
Write the solutions to the sheet.

The estimated time for solving this exam is 53 minutes, leaving you 7 minutes for revising your exam.

You can reach 23 points if you solve all problems. You will only need 20 points for a perfect score, i.e. 3 points are bonus points.

Different problems test different skills and knowledge, so do not get stuck on one problem.

	To be used for grading, do not write here								
prob.	1.1	1.2	2.1	2.2	3.1	3.2	4.1	Sum	grade
total	2	4	4	3	2	4	4	23	
reached									

Good luck to all students who take this test

1 Computational Logic

2pt
7min

Problem 1.1 (The *Nor* Connective)

All logical binary connectives can be expressed by the $\downarrow$ (*nor*) connective which is defined as $\mathbf{A} \downarrow \mathbf{B} := \neg(\mathbf{A} \vee \mathbf{B})$. Rewrite $\mathbf{P} \vee \neg\mathbf{P}$ (tertium non datur) into an expression containing only $\downarrow$ as a logical connective.

Hint: Recall that $\neg\mathbf{A} \Leftrightarrow \mathbf{A} \downarrow \mathbf{A}$.

Solution: $P \vee \neg P = \neg\neg(P \vee \neg P) = \neg(P \downarrow \neg P) = (P \downarrow (P \downarrow P)) \downarrow (P \downarrow (P \downarrow P))$

Problem 1.2 (A *Nor* Tableau Calculus) 12pt 13min

Develop a variant of the tableau calculus presented in class for propositional formulae expressed with $\downarrow$ (i.e. "not or") as the only logical connective.

Complete the following scheme of inference rules for such a tableau calculus and proof its correctness

$$\frac{\mathbf{A} \downarrow \mathbf{B}^{\mathsf{T}}}{?} \qquad \frac{\mathbf{A} \downarrow \mathbf{B}^{\mathsf{F}}}{?} \qquad \frac{\begin{array}{ll}\mathbf{A}^{\alpha} & \\ \mathbf{A}^{\beta} & \alpha \neq \beta\end{array}}{\bot}$$

Prove the formula $(P \downarrow (P \downarrow P)) \downarrow (P \downarrow (P \downarrow P))$ in your new tableau calculus.

Solution: The completed *Nor*-tableau calculus is the following.

$$\frac{\mathbf{A} \downarrow \mathbf{B}^{\mathsf{T}}}{\begin{array}{l}\mathbf{A}^{\mathsf{F}} \\ \mathbf{B}^{\mathsf{F}}\end{array}} \qquad \frac{\mathbf{A} \downarrow \mathbf{B}^{\mathsf{F}}}{\mathbf{A}^{\mathsf{T}} \mid \mathbf{B}^{\mathsf{T}}} \qquad \frac{\begin{array}{ll}\mathbf{A}^{\alpha} & \\ \mathbf{A}^{\beta} & \alpha \neq \beta\end{array}}{\bot}$$

And the proof of the formula is

$$\begin{array}{c|c} (P \downarrow (P \downarrow P)) \downarrow (P \downarrow (P \downarrow P))^{\mathsf{F}} & \\ P \downarrow (P \downarrow P)^{\mathsf{T}} & P \downarrow (P \downarrow P)^{\mathsf{T}} \\ P^{\mathsf{F}} & P^{\mathsf{F}} \\ P \downarrow P^{\mathsf{F}} & P \downarrow P^{\mathsf{F}} \\ P^{\mathsf{T}} & P^{\mathsf{T}} \\ \bot & \bot \end{array} \tag{1}$$

2 Graphs

4pt

Problem 2.1: Draw examples of

1. a directed graph with 4 nodes and 6 edges
2. a undirected graph with 7 nodes and 8 edges.

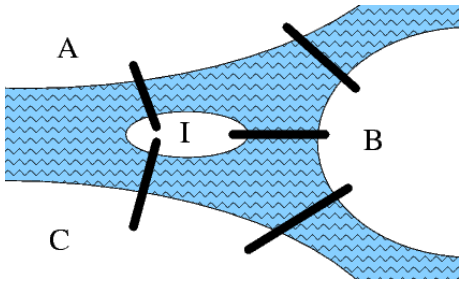
Present a mathematical representation of these graphs.

Solution: Any graph will do, for instance the following ones (we only give the mathematical formulation here, go draw them yourselves)

- $\langle \{a, b, c, d\}, \{\langle a, b\rangle, \langle b, c\rangle, \langle c, d\rangle, \langle d, a\rangle, \langle a, c\rangle, \langle b, d\rangle\}\rangle$
- $\langle \{a, b, c, d, e, f, g\}, \{\{a, b\}, \{b, c\}, \{c, d\}, \{d, e\}, \{e, f\}\{f, g\}, \{g, a\}, \{a, c\}, \{c, e\}\}\rangle$

Problem 2.2 ((Modified) Königsberg Bridge Problem) 30 10min

Consider a river fork with three banks (A,B,C) and one island (I) connected with bridges as shown in the figure.



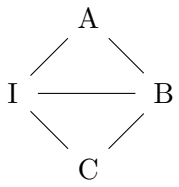
Is it possible to walk accross each of the bridges exactly once in an uninterrupted tour and return to the starting point?

In order to prove your answer first translate the question into a graph problem where the banks and the island are modeled as nodes and the bridges as undirected edges.

Hint: Consider the degree of each node (i.e.the number for edges connected to it). Relate the degrees of the nodes to the constraint of an uninterrupted tour.

Solution: If there is a round trip path which passes all edges of the graph once then each node must have an even degree, because whenever the path enters a node it must leave it again.

Since in the given bridge problem the nodes I and B have odd degree there is no such round trip.



---

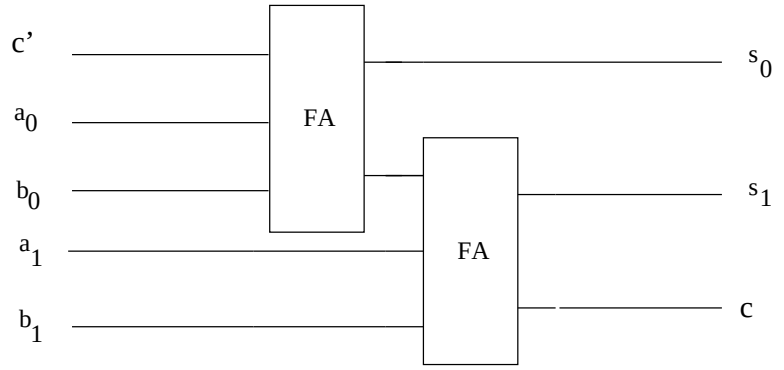
3 Combinatory Circuits

Problem 3.1 (Carry Chain Adder and Subtractor for TCN) 2pt 6min

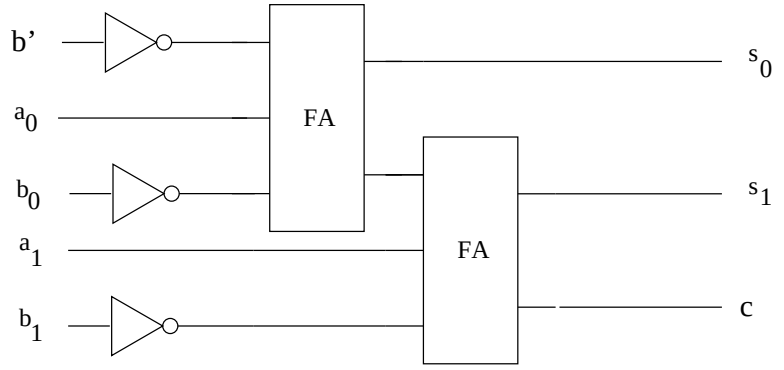
- Draw a 2-bit carry chain adder only using (1-bit) full adders as primitives.
- Draw a 2-bit subtractor for two's complement numbers using (1-bit) full-adders and Boolean gates of your choice.

Hint: Remember: An n-bit subtractor $f^n_{\mathrm{SUB}}(a, b, b')$ can be implemented as n-bit full-adder $\mathrm{FA}^n(a, \overline{b}, \overline{b'})$

Solution: 2 bit carry chain adder:



2 bit TCN subtractor:



Problem 3.2 (The Structure Theorem for TCN) 7 min

Write down the structure theorem for two's complement numbers (TCN) and make use of it to convert

- the integer -53 into a 8-bit TCN.
- the 8-bit TCN 10110101 into an integer.

Furthermore convert

- the integer -53 into a 10-bit TCN.
- the 10-bit TCN 1110110101 into an integer.

The 10-bit version of the conversion task shouldn't be any effort after solving the 8-bit version. You just have to remember the appropriate lemma to transfer an n-bit TCN to an $n+1$-bit TCN. How is the lemma called and what does it state?

Solution:

The 8-bit version:

- $B(()53) = 00110101$, $\overline{B(53)} = 11001010$, $B(\langle\!\langle\overline{B(53)}\rangle\!\rangle + 1) = 11001011$
- $\overline{B(\langle\!\langle 10110101\rangle\!\rangle - 1)} = 01001011$, $-\langle\!\langle\overline{B(\langle\!\langle b\rangle\!\rangle - 1)}\rangle\!\rangle = -75$

The 10-bit version: We make use of the *sign bit duplication lemma*

- We just have prepend two ïïn front to get from the 8-bit to the 10-bit version: $B(\langle\!\langle\overline{B(53)}\rangle\!\rangle + 1) = 1111001011$
- With the argument reverted we see that 1110110101 is the 10-bit variant of 10110101, hence it represents the same integer as above namely -75.

---

4 Assembler Programming

4pt
10min

Problem 4.1 (A Shift Program)

Write an assembler program that shifts the values of only the first n cells to its upper neighbor, where n is the content of the accumulator; i.e. if $P(k) = a_k$ for $k = 1 \dots n$ is the state before the execution of the program then it must be $P(k+1) = a_k$ for $k = 1 \dots n$ afterwards and the program must terminate.

Solution:

P	instruction	comment
0	`MOVE` $ACC\ IN1$	$IN1 := ACC$
1	`LOADIN` 1 0	$ACC := P(IN1)$
2	`STOREIN` 1 1	$P(IN1+1) := ACC$
3	`MOVE` $IN1\ ACC$	$ACC := IN1$
4	`SUBI` 1	$ACC := ACC - 1$
5	`JUMP`$_{=}$ > 1	if $ACC > 0$ goto step 0
6	`STOP` 0	Stop