CS 201 Section 1 Homework 8 Suggested Solutions

1. The problem description is misleading, because `if_icmpeq4` does not actually compute (MDR-TOS); this is computed later on, using OPC as a temporary in the meantime. But the point of the question is that it is not possible to subtract TOS from MDR; the left (A) operand to every ALU operation is hardwired to H. The only way to compute (MDR-TOS) is to first copy MDR to H.

2. The obvious code sequence is:

```
ILOAD j
ILOAD k
IADD
ISTORE i
```

The total number of microinstructions is therefore $6 + 6 + 4 + 7 = 23$. At 200 MHz, each microinstruction takes 5 nsec, so 23 of them take 115 nsec.

4. The following would work:

```
pop21 SP = SP - 1
pop22 MAR = SP = SP - 1; rd
pop23
pop4 TOS = MDR; goto Main1
```

Or, even more simply:

```
pop21 SP = SP - 1; goto pop1
```

5. Since it's too tedious to insert a picture here, I'll show the machine in tabular form. Each row corresponds to a state; each non-final state has two out-transitions, labeled A and B, with target specified by the corresponding column entry in the table. The initial state is (0,0). The final states are A-wins and B-wins. The choice of state names is suggested by normal tennis terminology, but the actual names you use don't matter.

State	$\xrightarrow{A}$	$\xrightarrow{B}$
(0,0)	(1,0)	(0,1)
(1,0)	(2,0)	(1,1)
(0,1)	(1,1)	(0,2)
(2,0)	(3,0)	(2,1)
(1,1)	(2,1)	(1,2)
(0,2)	(1,2)	(0,3)
(3,0)	A-wins	(3,1)
(2,1)	(3,1)	deuce
(1,2)	deuce	(1,3)
(0,3)	(1,3)	B wins
(3,1)	A-wins	ad-A
deuce	ad-A	ad-B
(1,3)	ad-B	B-wins
ad-A	A-wins	deuce
ad-B	deuce	A-wins

6. The problem is that the `fetch` in the second microinstruction, which fetches the byte *following* the next opcode into MBR, will have completed by the end of the third microinstruction's cycle. So the `goto (MBR)` in the third microinstruction will try to jump to the opcode (or parameter) specified by the newly fetched byte, skipping over the true next opcode. The simple solution is just to defer the `fetch` until the third microinstruction:

```
pop1        MAR = SP = SP - 1; rd
Main1.pop   PC = PC + 1
pop3        TOS = MDR; fetch; goto (MBR)
```