

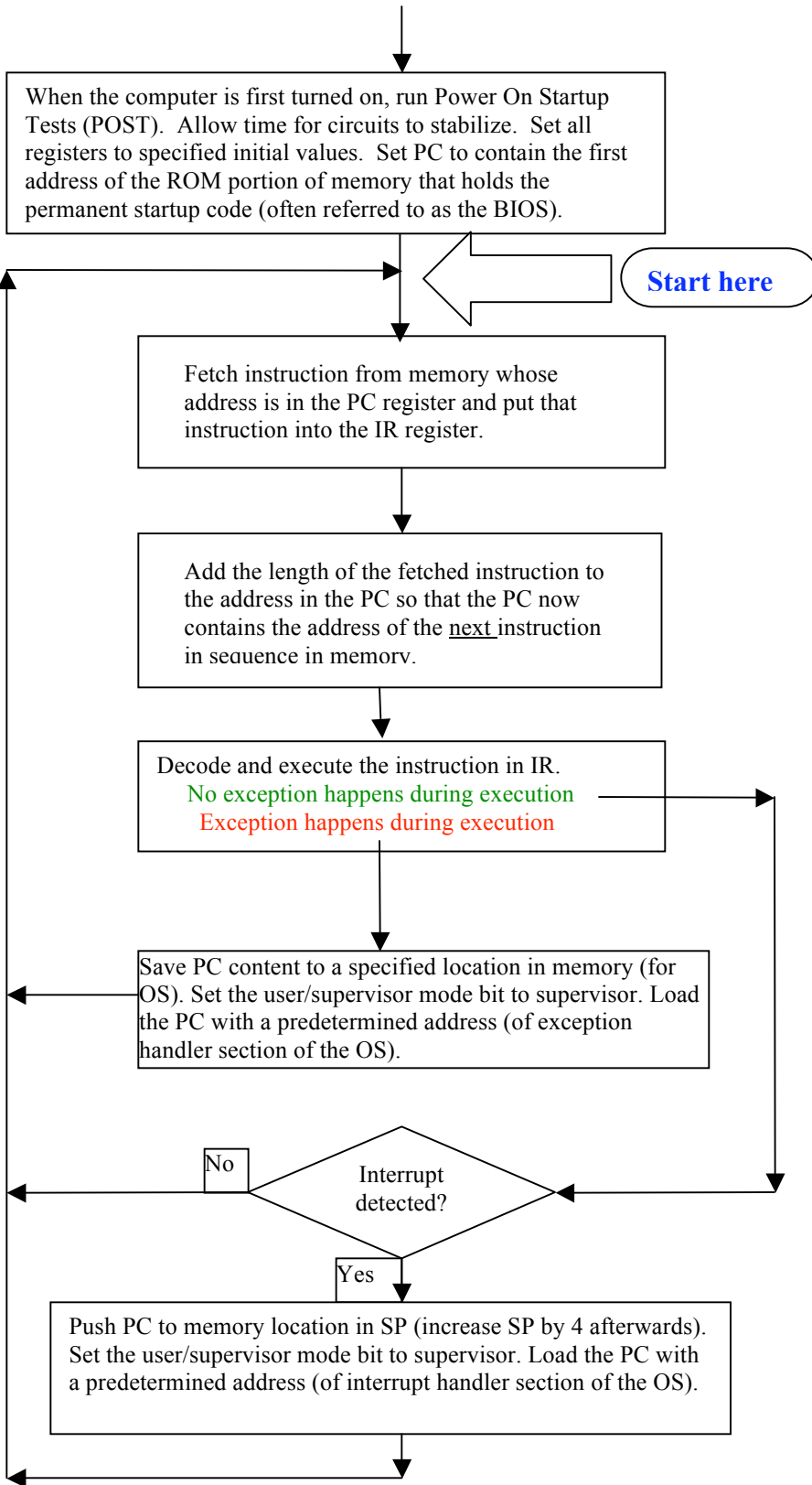
Assignment 2
IFE Cycle Trace
Due in class on Monday, September 29

This assignment is given on four pages. This first page contains the assignment directions. The second page provides the IFE cycle. The third page is a listing of assembly language instructions in main memory to be used in the assignment. The fourth page is a page you are to complete. You must turn in pages 3 and 4 in class on Monday, September 25.

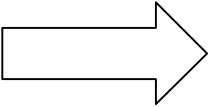
On page 4 you are to do a trace of the execution of the instructions found on page 3.

1. Page 4 has current values for the registers for starting the trace at the top. There are numbered blank lines below the registers. The numbers represent the current pass through the IFE cycle.
2. You must fill in each line with the values that would be in the registers **after** the pass through the IFE cycle corresponding to the line number, **just before** the IFE cycle would continue back to the fetch phase, **as marked on the IFE diagram** on page 2.
3. If any address location in memory changes value, record that change at the proper address on the memory sheet, page 3.
4. Instructions and data are assumed to take up 4 bytes each, so memory is only listed at 4-byte boundary values.
5. PC is the program counter register. When the PC is updated, it should be updated by 4, as all instructions are 4 bytes long.
6. IR is the instruction register.
7. r0 and r1 are general purpose registers.
8. PS is the program status register.
9. SP is the stack pointer register.
10. cp is a special register used by the operating system to hold the address of the PCB of the currently running process.
11. rdy is a special register used by the operating system to hold the address of the PCB at the front of the ready to run queue.
12. The PS (status) register contains only two bits. The first is the user/supervisor bit; 0 indicates user mode and 1 indicates supervisor mode. The second bit is a compare bit; it is set to 1 by hardware if the effect of executing a compare instruction is true, and 0 otherwise.
13. The SP (stack pointer) register points to the top (first non-used element) of the currently running user process stack.
14. If a clock interrupt occurs the hardware will always place address 300 into the PC as part of its interrupt circuitry.
15. If the PC is saved by the IFE cycle, it is pushed onto the stack of the currently running process.
16. **A clock interrupt is detected on the 6th pass through the IFE cycle.**

Fill in lines until it is impossible to continue (e.g., because the address needed is not in memory). Explain why you stop where you do. **ONLY FILL IN VALUES ON THE TRACE SHEET THAT CHANGE.**



Fill in trace line I at this point after Ith pass through IFE cycle.



Main Memory

Name _____

```
0000 --- start of OS in memory
. . .
. --- begin of clock interrupt/process switch in OS
0300 store r0, 8(cp) -- store contents of r0 into memory location 8+cp
0304 store r1, 12(cp) -- store contents of r1 into memory location 12+cp
0308 pop 4(cp)      -- pop the contents of memory location referred to by SP into
                   -- memory location 4+cp; decrement SP by 4
0312 load cp, rdy  -- copy value in register rdy into register cp
0316 load r0, 8(cp) -- load r0 with contents of memory location 8+cp
0320 load r1, 12(cp) -- load r1 with contents of memory location 12+cp
0324 setTimer #100 -- set the timer to 100 microseconds (timer not on trace sheet)
0328 setUSbit #0   -- set the US bit in the PS register to 0
0332 load pc, 4(cp) -- load the pc register with contents of memory location 4+cp
0336 load r1, #30  -- load r1 with the constant 30
. . .
1080 680
1084 860
1088 860
1092 1000
1096 1800
. . .
1200 600
1204 2980
1208 2509
1212 300
1216 95
. . .
1890 - end of OS in memory
. . .
2000 --- start of process 35 in memory
. . .
2400 --- load r0, #4      -- load r0 with the constant 4
2404 --- add r0, r1, r1  -- add r0 to r1 and store the result in r1
2408 --- sub r0, #1, r0  -- subtract 1 from the contents of r0 and store the result in r0
2412 --- cmpge r0, #0    -- set PS compare bit to 1 (true) if r0 >= 0; 0 otherwise
2416 --- jt 2404         -- jump true (if PS compare bit is 1) to memory location 2404
2420 --- load r0, #5     -- load r0 with constant 5
.
. --- more of process 35
.
2600 --- 500
2604 --- 501
. . .
2800 --- end of process 35 in memory
. . .
2890 --- start of process 65 in memory
. . .
2980 --- load r0, #15    -- load r0 with constant 15
. . .
3800 --- end of process 65 in memory
. . .
9999 --- end of memory
```

