Problem 1: Consider the solution to Spring 2002 Homework 4, shown on the next page. (The solution was updated 19 November 2002, the PED is shown in dynamic order instead of the nearly-impossible-to-read static order.)

(*a*) Show the contents of the reorder buffer in cycle 12. For each entry show the values of the fields from the illustration below, for the PC show the instruction (`ldc1`, `mul.d`, etc.). (The fields are ST, dst, dstPR, and incumb.) If a field value cannot be determined from the solution leave it blank, that will include fields related to registers `$2` and `$3`.

Note: A solution not showing instructions 1-4 would also be correct.

```
   Solution
   "PC"               ST  dst  dstPR  incumb
 1 sdc1 0($1), f0
 2 addi $1, $1, 8     C   $1   95     98
 3 bne $2, $0 LOOP    C
 4 sub $2, $1, $3     C

 5 ldc1 f0, 0($1)     C   f0   94     96
 6 mul.d f0, f0, f2       f0   93     94
 7 sdc1 0($1), f0
 8 addi $1, $1, 8     C   $1   92     95
 9 bne $2, $0 LOOP    C
10 sub $2, $1, $3     C

11 ldc1 f0, 0($1)     C   f0   91     93
12 mul.d f0, f0, f2       f0   90     91
13 sdc1 0($1), f0
14 addi $1, $1, 8     C   $1   89     92
15 bne $2, $0 LOOP
16 sub $2, $1, $3     C

17 ldc1 f0, 0($1)         f0
18 mul.d f0, f0, f2       f0
```

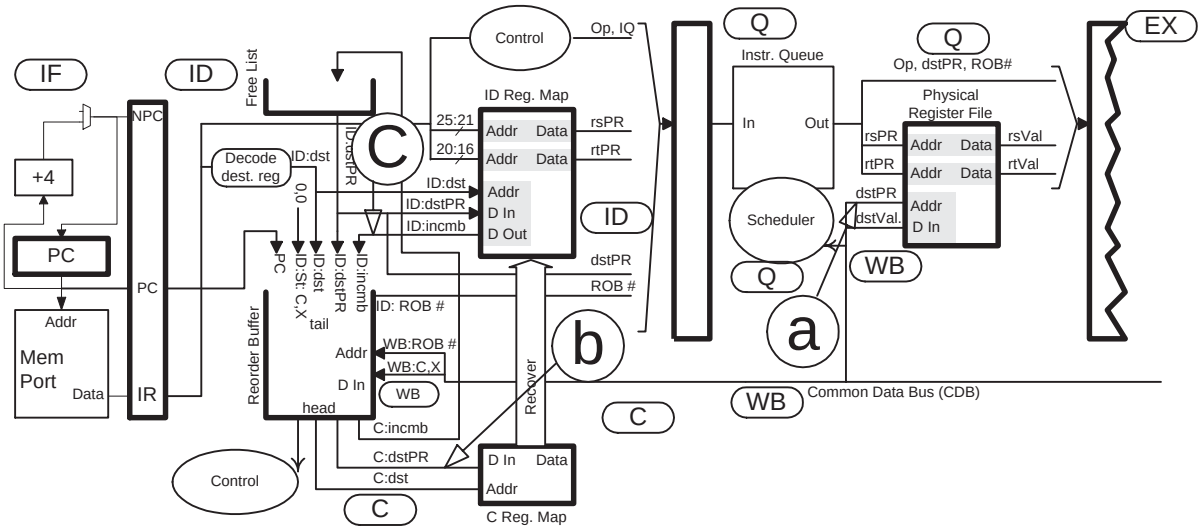
(*b*) For the solution to the part above, number each instruction. (1, 2, 3, etc.) Show the contents of the instruction queue at cycle 12 identifying each instruction by these numbers.

The instruction queue holds instructions waiting to execute, at cycle 12 only the 18th instruction above, multiply is waiting.

```
   Solution
18 mul.d f0, f0, f2       f0
```

(*c*) On the illustration there are three wires labeled with big lower-case letters, a, b, and c, and corresponding rows in a table in the middle of the next page. Based on the solution to last semester's problem, show what values are on those wires in each cycle that they are used. Omit cycles where a value cannot be determined. Note that the illustration is for a one-way (non-superscalar) processor but the program runs on a four-way system. That means each wire can hold up to four values in one cycle. *Hint: The solution for at least one of the letters already appears. Just label the row(s) in the appropriate table with the letter. At least one of the letters does not appear, so that will have to be written in.*

Row b is the same as the commit map (with the two commit map rows merged into one.)


IF
ID
EX
Free List
Control
Op, IQ
Q
Instr. Queue
Q
Op, dstPR, ROB#
Physical
Register File
NPC
+4
PC
Addr
Mem
Port
Data
IR
PC
Decode
dest. reg
ID:dst
ID:dstPR
C
25:21
20:16
ID Reg. Map
Addr
Data
rsPR
rtPR
ID:dst
ID:dstPR
ID:incmb
D In
D Out
ID
dstPR
ROB #
In
Out
Scheduler
Q
WB
rsPR
rtPR
dstPR
dstVal.
rsVal
rtVal
a
b
Reorder Buffer
PC
ID:St:C,X
ID:dst
ID:dstPR
ID:incmb
ID: ROB #
tail
Addr
D In
WB:ROB #
WB:C,X
WB
head
C:incmb
C:dstPR
C:dst
Recover
C
WB
Common Data Bus (CDB)
Control
C
D In
Data
Addr
C Reg. Map

```
LOOP:  # Instructions shown in dynamic order. (Instructions repeated.)
 # Cycle          0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16
 ldc1 f0, 0($1)   IF ID Q  L1 L2 WC
 mul.d f0, f0, f2 IF ID Q        M1 M2 M3 M4 M5 M6 WC
 sdc1 0($1), f0   IF ID Q  L1                    L2 WC
 addi $1, $1, 8   IF ID Q  EX WB                    C
 bne $2, $0 LOOP     IF ID Q  B  WB                 C
 sub $2, $1, $3      IF ID Q  EX WB                 C
 # Cycle          0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17
 ldc1 f0, 0($1)            IF ID Q  L1 L2 WB              C
 mul.d f0, f0, f2          IF ID Q        M1 M2 M3 M4 M5 M6 WC
 sdc1 0($1), f0            IF ID Q  L1                    L2 WC
 addi $1, $1, 8            IF ID Q  EX WB                    C
 bne $2, $0 LOOP              IF ID Q  B  WB                 C
 sub $2, $1, $3               IF ID Q  EX WB                 C
 # Cycle          0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17
 ldc1 f0, 0($1)                     IF ID Q  L1 L2 WB             C
 mul.d f0, f0, f2                   IF ID Q        M1 M2 M3 M4 M5 M6 WC
 sdc1 0($1), f0                     IF ID Q  L1                       L2 WC
 addi $1, $1, 8                     IF ID Q  EX WB                       C
 bne $2, $0 LOOP                       IF ID Q     B  WB                C
 sub $2, $1, $3                        IF ID Q  EX WB                   C
 # Cycle          0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17
 ldc1 f0, 0($1)                           IF ID Q  L1 L2 WB
 mul.d f0, f0, f2                         IF ID Q        M1 M2 M3 M4 M5
 ...
 # Cycle          0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17
 ID Map
 f0 99               97,96    94,93    91,90
 $1 98               95       92       89
 # In cycle one first 97 is assigned to f0, then 96 (replacing 97).  The
 # same sort of replacement occurs in cycles 4 and 7.
 # Cycle         0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17
 # FALL 2002 SOLUTION HERE
 a                         95 97    92 94    89 91       93       90
 a (continued)                                  96
 b                            97               96 95 94 93 92 91 90 89
 c                 99,97,98 96,95,94 93,92,91 ...
 # Cycle          0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17
 Commit Map
 f0 99                            97                96    94 93    91 90
 $1 98                                                 95       92       89
 # Cycle          0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17
 Physical Register File
 99               1.0            ]
 98               0x1000                           ]
 97                 [            10               ]
 96                 [                             11    ]
 95                 [         0x1008                        ]
 94                           [           20               ]
 93                           [                            2.2   ]
 92                           [        0x1010                        ]
 # Cycle          0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17
```