Problem 1. Building the Beta (10 points)

For this problem assume that each register has been initialized to the value 0x0000??00 where "??" is the register number as a two-digit hex number. So R0 is initialized to 0x00000000, R1 to 0x00000100, ..., and R30 to 0x00001E00. R31 of course always reads 0.

For the instruction below, please indicate the values that will be found in the unpipelined Beta datapath just before the end of the clock cycle in which the instruction is executed. If the value doesn’t matter since it’s not used during the execution of the instruction or can’t be determined, write “—”. Otherwise enter the value in decimal, hex (use a “0x” prefix), or binary (use a “0b” prefix). You may wish to refer to the Unpipelined Beta diagram included in the reference material.

```
. = 0x100
LD(R3,-0x200,R7)
// hex for instruction
0x60E3FE00
```

RA1 input to register file: 3
RA2 input to register file: —
RD1 output from register file (JT): 0x300
RD2 output from register file (MWD): —
A input to ALU: 0x300
B input to ALU: 0xFFFFFE00
Output from ALU (MA): 0x100
Output from Data Memory (MRD): 0x60E3FE00
WD input to Register File: 0x60E3FE00
Input to the PC register (output of RESET mux): 0x104
Problem 2. Caches (11 points)

(A) (1 point) The timing for a particular cache is as follows: checking the cache takes 1 cycle. If there’s a hit the data is returned to the CPU at the end of the first cycle. If there’s a miss, it takes 10 additional cycles to retrieve the word from main memory, store it in the cache, and return it to the CPU. If we want an average memory access time of 1.4 cycles, what is the minimum possible value for the cache’s hit ratio?

\[
1.4 = 1 + (1 - \alpha) \times 10 \\
0.4 = 1 - \alpha
\]

Minimum possible value of hit ratio: \(0.96\)

(B) (3 points) If the cache block size, i.e., words/cache line, is doubled but the total number of data words in the cache is unchanged, how will the following cache parameters change? Please circle the best answer.

- \# of offset bits: UNCHANGED \(\ldots +1 \ldots -1 \ldots 2x \ldots 0.5x \ldots\) CAN’T TELL
- \# of tag bits: UNCHANGED \(\ldots +1 \ldots -1 \ldots 2x \ldots 0.5x \ldots\) CAN’T TELL
- \# of cache lines: UNCHANGED \(\ldots +1 \ldots -1 \ldots 2x \ldots 0.5x \ldots\) CAN’T TELL

Consider a direct-mapped cache with 64 total data words with 1 word/cache line, which uses a LRU replacement strategy and a write-back write strategy. This cache architecture is used for parts (C) through (F).

(C) (1 point) If cache line number 5 is valid and its tag field has the value 0x1234, what is the address in main memory of the data word currently residing in cache line 5?

Main memory address of data word in cache line 5: 0x0123414

The program shown on the right repeatedly executes an inner loop that sums the 16 elements of an array that is stored starting in location 0x310.

The program is executed for many iterations, then a measurement of the cache statistics is made during one iteration through all the code, i.e., starting with the execution of the instruction labeled outer_loop: until just before the next time that instruction is executed.

inner_loop:
CMOVE(16,R0) // initialize loop index J
CMOVE(0,R1)
loop: // add up elements in array
SUBC(R0,1,R0) // decrement index
MULC(R0,4,R2) // convert to byte offset
LD(R2,0x310,R3) // load value from A[J]
ADDC(R3,0,R1) // add to sum
BNE(R0,loop) // loop until all words are summed
BR(outer_loop) // perform test again!
(D) (2 points) In total, how many instruction fetches occur during one complete iteration of the outer loop? How many data reads?

\[
\text{number of instruction fetches: } 83
\]

\[
\text{number of data reads: } 16
\]

(E) (2 points) How many instruction fetch misses occur during one complete iteration of the outer loop? How many data read misses? Hint: remember that the array starts at address 0x310.

Number of instruction fetch misses: 4

Number of data read misses: 4

(F) (2 points) What is the hit ratio measured after one complete iteration of the outer loop?

\[
\text{Hit ratio: } \frac{99-8}{99} = \frac{91}{99}
\]
Problem 3. Pipelined Beta (9 points)

The program shown on the right is executed on a 5-stage pipelined Beta with full bypassing and annulment of instructions following taken branches.

The program has been running for a while and execution is halted at the end of cycle 108.

The pipeline diagram shown below shows the history of execution at the time the program was halted.

Please indicate on which cycle(s), 100 through 108, each of the following actions occurred. If the action did not occur in any cycle, write “NONE”. You may wish to refer to the signal names in the 5-stage Pipelined Beta Diagram included in the reference material.

Register value used from Register File: 100, 106, 108

Register value bypassed from ALU stage to RF stage: 101, 102, [103]*

Register value bypassed from MEM stage to RF stage: NONE, [104]*

Register value bypassed from WB stage to RF stage: 105

IRS reclaimed by IF was 1: 106

IRS reclaimed by IF was 2: NONE

STALL was 1: 103, 104

BNE ← PCSEL was 1: 106

LD ← WDSEL was 2: 105

END OF QUIZ 3!