ICS Homework 6

March 24, 2021

1 Pipeline

Suppose we add a new instruction *rjmp rB* to Y86 instruction sets. It will jmp to the address stored in *rB*.
1. Fill in the function of each stage for *rjmp rB* instructionin Y86 sequential implementation. (NOTE: use *valB* to update PC)

2. As shown in the new PIPE logic figure, we add a forwarding logic from *E_valB* to *f_pc* to support *rjmp* instruction, since the target address require read from register file. Please describe all possible hazards due to new instruction *rjmp*. You need provide detail explanation and list detection conditions like Figure 4.64 and control action like Figure 4.66. (Do not consider the hazard combinations here.)

3. Please list all hazard combinations (arise simultaneously) including *rjmp* instruction. You need draw pipeline states figures like Figure 4.67 and list pipeline control action like tables after Figure 4.67 for each combination.

4.The original PIPE implementation of Y86 should be modified to support the *rjmp* instruction. Please describe the modification and provide HCL of *f_pc* and *D_bubble* logic. (NOTE: Only need to write the code about *rjmp* instruction)

2 Machine Independent Optimization

Suppose we have some codes as below.

```
1
2  typedef struct {
3      int vals[3];
4  } block_t;
5
6  typedef struct {
7      int length;
8      block_t *blocks;
9  } blocklist;
```

```
10
11 int get_length(blocklist *bl) {
12     return bl->length;
13 }
14
15 block_t* get_blocks(blocklist *bl) {
16     return bl->blocks;
17 }
18
19 void SUM(blocklist *bl, long *dest) {
20     for (int i = 0; i < get_length(bl); i++) {
21         int size = 1;
22         for (int j = 0; j < 3; j++)
23             size = size * get_blocks(bl)[i].vals[j];
24             *dest = *dest + size;
25         }
26 }
```

Try to optimize the function SUM with a combination of optimizations you have learned in the ICS class. Comment briefly on your optimizations.