Assignment 5 - Semil Jain (u1417989)

1. **Out-of-order processing (100 points)**
Loop for at least 2 iterations:

line1: L.D LR1 0(LR2)
DADD LR1, LR1, LR3
DADD LR1, LR1, LR1
ST.D LR1, 0(LR2)
DADD LR2, LR2, 8
BNE LR2, LR4, line1

Solution:

- We have 32 Logical registers(LR1,LR2,…LR32) and 38 physical registers (6 available registers - PR33, PR34, PR35, PR36, PR37, PR38)
- Width Given = 3. Hence, at any given cycle there can be at most 3 instructions brought in the Issue Queue /issued from queue /committed.
- Branch prediction is perfect, Hence we know the 12 instructions to execute at the start of our cycle.
- Dependent of DADD has 0 stall cycle in order to leave the issue queue(next cycle). Dependent of LD/ST has 1 stall cycle in order to leave the issue queue(next to next cycle).
- A ROB, an issue queue, and an LSQ with 20 entries each. Hence No stalling due to hardware limitations.
- Assume that before execution, All Logical registers are mapped to their physical counterparts. (LR1->PR1, LR2->PR2,…and so on)
- As mentioned above we have the six remaining physical registers available.
- 5 pipeline stages before placed in the issue queue, 5 additional stages (6 for LD/ST) after leaving issue queue
- Assume that after the 5 stages before the IQ are completed the Instruction arrives at the Issue Queue at the end of cycle i (First 3 instructions would arrive at i becoz of Width)
- Fine the Solution Below: (Same template used as per the Modal Solution)

- InQ - Cycle at which the instruction arrived into the Issue Queue
- Issued - Cycle at which the instruction is issued (leaves Issue Queue)
- Complete - Cycle at which the instruction completes
- Commit - Cycle at which the instruction gets committed

								On commit, entry added to	
Inst #	**Original code**	**Renamed Code**	**Entry added to Spec. Reg. Map**	**InQ**	**Issued**	**Complete**	**Commit**	**Committed Reg. Map**	**Reg. Free List**
1	LD LR1, 0(LR2)	LD PR33, 0(PR2)	LR1->PR33	i	i+1	i+7	i+7	LR1->PR33	PR1
2	DADD LR1, LR1, LR3	DADD PR34, PR33, PR3	LR1->PR34	i	i+3	i+8	i+8	LR1->PR34	PR33
3	DADD LR1, LR1, LR1	DADD PR35, PR34, PR34	LR1->PR35	i	i+4	i+9	i+9	LR1->PR35	PR34
4	ST.D LR1, 0(LR2)	ST.D PR35, 0(PR2)	-	i+1*1	i+5	i+11	i+11	-	-
5	DADD LR2, LR2, 8	DADD PR36, PR2, 8	LR2->PR36	i+1	i+2	i+7	i+11 *3	LR2->PR36	PR2
6	BNE LR2, LR4, line1	BNE PR36, PR4, line1	-	i+1	i+3	i+8	i+11 *3	-	-
7	LD LR1, 0(LR2)	LD PR37, 0(PR36)	LR1->PR37	i+2*1	i+3	i+9	i+12 *2	LR1->PR37	PR35
8	DADD LR1, LR1, LR3	DADD PR38, PR37, PR3	LR1->PR38	i+2	i+5	i+10	i+12 *3	LR1->PR38	PR37
9	DADD LR1, LR1, LR1	DADD PR1, PR38, PR38	LR1->PR1	i+8*4	i+9	i+14	i+14	LR1->PR1	PR38
10	ST.D LR1, 0(LR2)	ST.D PR1, 0(PR36)	-	i+8	i+10	i+16	i+16	-	-
11	DADD LR2, LR2, 8	DADD PR33, PR36, 8	LR2->PR33	i+9*4	i+10	i+15	i+16 *3	LR2->PR33	PR36
12	BNE LR2, LR4, line1	BNE PR33, PR4, line1	-	i+9	i+11	i+16	i+16	-	-

*1: Fetch width full. Fetched in next cycle.
*2: Commit width full. Committed in next cycle
*3: Commit delayed in order to commit in order.
*4: No free register in Free Register List. Must wait until a physical register frees up.

Dependence Relation

In Total after Loop unrolling twice we have 12 instructions. The second column indicates the Code after renaming to Physical registers based on the availability from the Reg Free list. After an instruction is renamed but not committed we maintain the mapping in Speculative Reg map. InQ, Issued and Commit columns can only have at most 3 instructions in the same cycle. After commit we add the entry to Committed Reg Map and then we can free the not required Registers. **The 12th instruction will get committed in the 'I+16' cycle.**