**SEQUENTIAL**

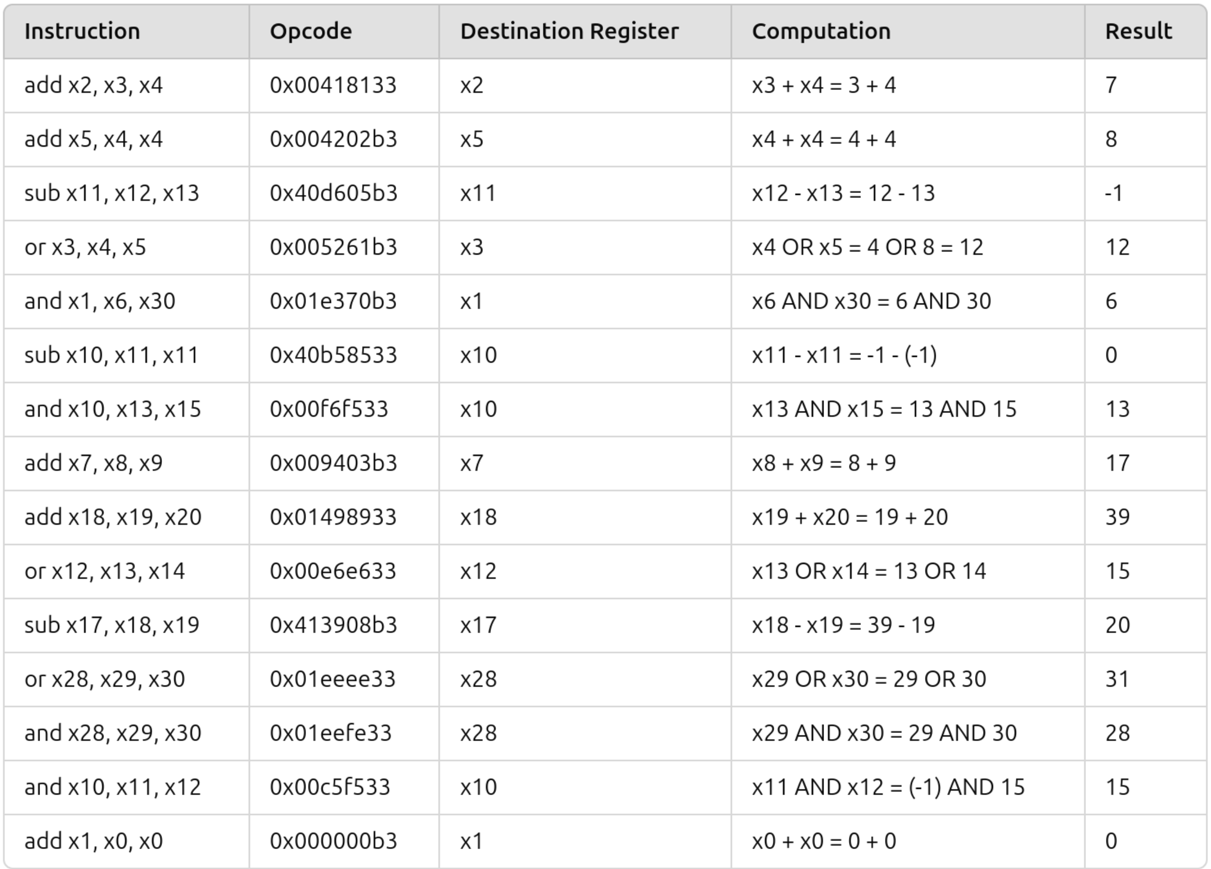
R TYPE - > instructions_test_1.hex

Dont change

```
0x00418133
0x004202b3
0x40d605b3
0x005261b3
0x01e370b3
0x40b58533
0x00f6f533
0x009403b3
0x01498933
0x00e6e633
0x413908b3
0x01eeee33
0x01eefe33
0x00c5f533
0x000000b3
0x00000133
```

```
add x2, x3, x4          x2=7
add x5, x4, x4         x5=8
sub x11, x12, x13      x11=-1
or x3, x4, x5          x3=5
and x1, x6, x30        x1=36
sub x10, x11, x11      x10=22
and x10, x13, x15      x10=27….
add x7, x8, x9
add x18, x19, x20
or x12, x13, x14
sub x17, x18, x19
or x28, x29, x30
and x28, x29, x30
and x10, x11, x12
add x1, x0, x0
```

Sequential_Design_TestCase_1



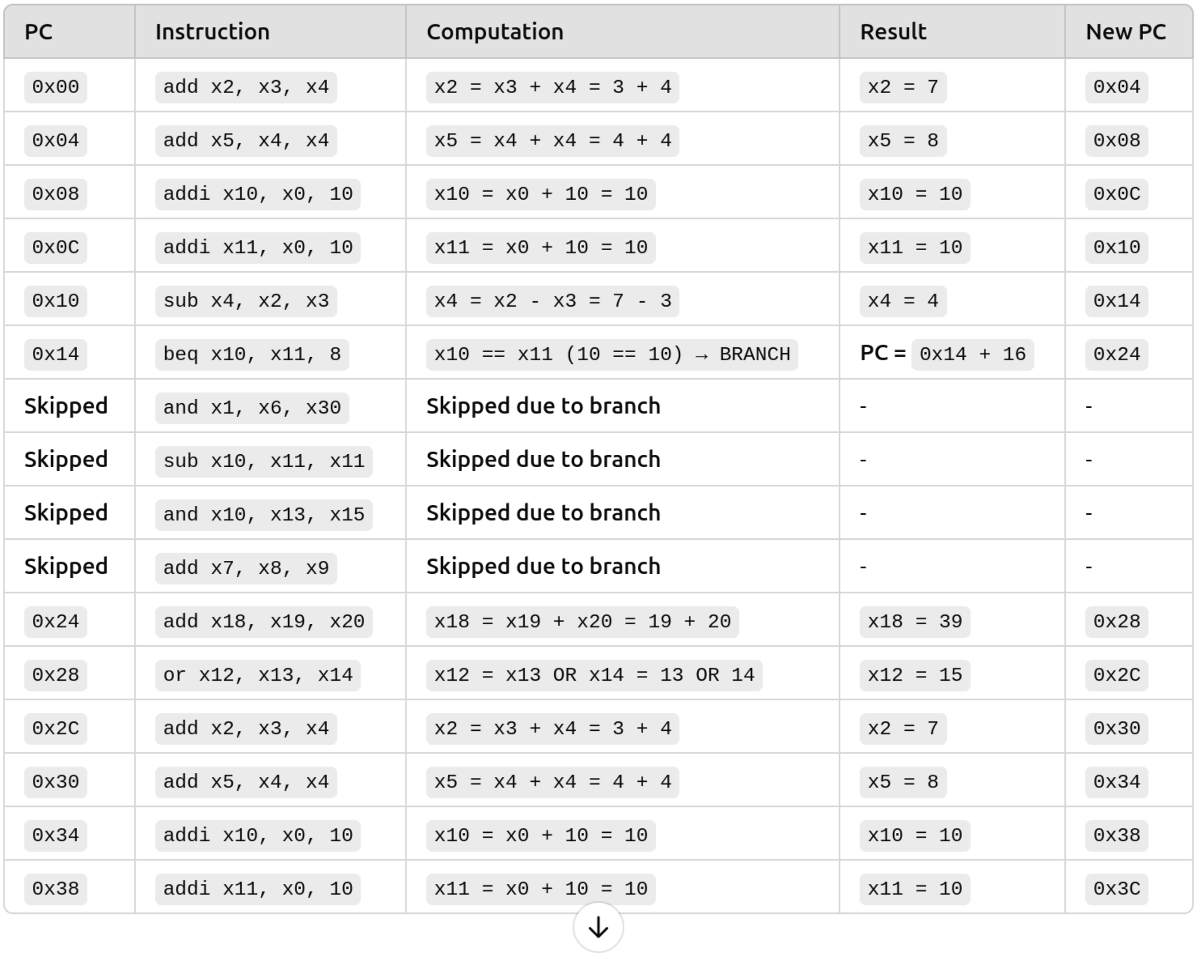
| Instruction | Opcode | Destination Register | Computation | Result |
|---|---|---|---|---|
| add x2, x3, x4 | 0x00418133 | x2 | x3 + x4 = 3 + 4 | 7 |
| add x5, x4, x4 | 0x004202b3 | x5 | x4 + x4 = 4 + 4 | 8 |
| sub x11, x12, x13 | 0x40d605b3 | x11 | x12 - x13 = 12 - 13 | -1 |
| or x3, x4, x5 | 0x005261b3 | x3 | x4 OR x5 = 4 OR 8 = 12 | 12 |
| and x1, x6, x30 | 0x01e370b3 | x1 | x6 AND x30 = 6 AND 30 | 6 |
| sub x10, x11, x11 | 0x40b58533 | x10 | x11 - x11 = -1 - (-1) | 0 |
| and x10, x13, x15 | 0x00f6f533 | x10 | x13 AND x15 = 13 AND 15 | 13 |
| add x7, x8, x9 | 0x009403b3 | x7 | x8 + x9 = 8 + 9 | 17 |
| add x18, x19, x20 | 0x01498933 | x18 | x19 + x20 = 19 + 20 | 39 |
| or x12, x13, x14 | 0x00e6e633 | x12 | x13 OR x14 = 13 OR 14 | 15 |
| sub x17, x18, x19 | 0x413908b3 | x17 | x18 - x19 = 39 - 19 | 20 |
| or x28, x29, x30 | 0x01eeee33 | x28 | x29 OR x30 = 29 OR 30 | 31 |
| and x28, x29, x30 | 0x01eefe33 | x28 | x29 AND x30 = 29 AND 30 | 28 |
| and x10, x11, x12 | 0x00c5f533 | x10 | x11 AND x12 = (-1) AND 15 | 15 |
| add x1, x0, x0 | 0x000000b3 | x1 | x0 + x0 = 0 + 0 | 0 |

BRANCH -> test_2.hex
Dont change

1. 0x00418133
2. 0x004202b3
3. 0x00a00513
4. 0x00a00593
5. 0x40310233
6. 0x00b50463
7. 0x01e370b3
8. 0x40b58533
9. 0x00f6f533
10. 0x009403b3
11. 0x01498933
12. 0x00e6e633
13. 0x00418133
14. 0x004202b3
15. 0x00a00513
16. 0x00a00593

1. add x2, x3, x4
2. add x5,x4,x4
3. addi x10, x0, 10
4. addi x11, x0, 10
5. sub x4, x2, x3
6. beq x10, x11, 16(12 pc)
7. and x1, x6, x30
8. sub x10, x11, x11
9. and x10, x13, x15
10. add x7, x8, x9
11. add x18, x19, x20
12. or x12, x13, x14
13. add x2, x3, x4
14. add x5,x4,x4
15. addi x10, x0, 10
16. addi x11, x0, 10

Sequential_Design_TestCase_2

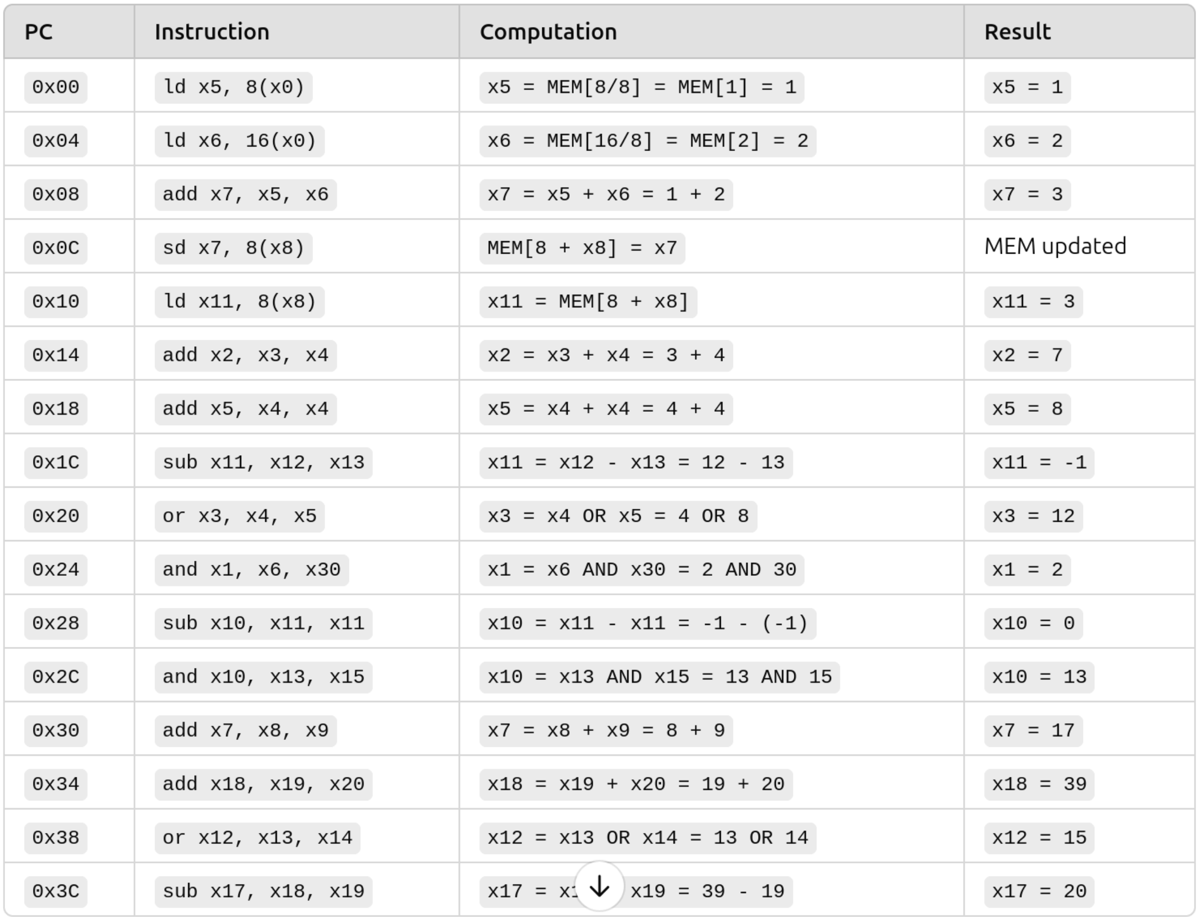


| PC | Instruction | Computation | Result | New PC |
|---|---|---|---|---|
| `0x00` | `add x2, x3, x4` | `x2 = x3 + x4 = 3 + 4` | `x2 = 7` | `0x04` |
| `0x04` | `add x5, x4, x4` | `x5 = x4 + x4 = 4 + 4` | `x5 = 8` | `0x08` |
| `0x08` | `addi x10, x0, 10` | `x10 = x0 + 10 = 10` | `x10 = 10` | `0x0C` |
| `0x0C` | `addi x11, x0, 10` | `x11 = x0 + 10 = 10` | `x11 = 10` | `0x10` |
| `0x10` | `sub x4, x2, x3` | `x4 = x2 - x3 = 7 - 3` | `x4 = 4` | `0x14` |
| `0x14` | `beq x10, x11, 8` | `x10 == x11 (10 == 10) → BRANCH` | **PC =** `0x14 + 16` | `0x24` |
| **Skipped** | `and x1, x6, x30` | **Skipped due to branch** | - | - |
| **Skipped** | `sub x10, x11, x11` | **Skipped due to branch** | - | - |
| **Skipped** | `and x10, x13, x15` | **Skipped due to branch** | - | - |
| **Skipped** | `add x7, x8, x9` | **Skipped due to branch** | - | - |
| `0x24` | `add x18, x19, x20` | `x18 = x19 + x20 = 19 + 20` | `x18 = 39` | `0x28` |
| `0x28` | `or x12, x13, x14` | `x12 = x13 OR x14 = 13 OR 14` | `x12 = 15` | `0x2C` |
| `0x2C` | `add x2, x3, x4` | `x2 = x3 + x4 = 3 + 4` | `x2 = 7` | `0x30` |
| `0x30` | `add x5, x4, x4` | `x5 = x4 + x4 = 4 + 4` | `x5 = 8` | `0x34` |
| `0x34` | `addi x10, x0, 10` | `x10 = x0 + 10 = 10` | `x10 = 10` | `0x38` |
| `0x38` | `addi x11, x0, 10` | `x11 = x0 + 10 = 10` | `x11 = 10` | `0x3C` |

LOAD / STORE /ARITHMETIC -> instructions_test_3.hex
Dont change
0x00803283
0x01003303
0x006283b3
0x00743423
0x00843583
0x00418133
0x004202b3
0x40d605b3
0x005261b3
0x01e370b3
0x40b58533
0x00f6f533
0x009403b3
0x01498933
0x00e6e633
0x413908b3

1. ld x5, 8(x0)->0
2. ld x6, 16(x0)->1
3. add x7, x5, x6
4. sd x7, 8(x8)
5. Ld x11,8(x8)
6. add x2, x3, x4
7. add x5, x4, x4
8. sub x11, x12, x13
9. or x3, x4, x5
10. and x1, x6, x30
11. sub x10, x11, x11
12. and x10, x13, x15
13. add x7, x8, x9
14. add x18, x19, x20
15. or x12, x13, x14
16. sub x17, x18, x19

# Sequential_Design_TestCase_3



| PC | Instruction | Computation | Result |
|---|---|---|---|
| `0x00` | `ld x5, 8(x0)` | `x5 = MEM[8/8] = MEM[1] = 1` | `x5 = 1` |
| `0x04` | `ld x6, 16(x0)` | `x6 = MEM[16/8] = MEM[2] = 2` | `x6 = 2` |
| `0x08` | `add x7, x5, x6` | `x7 = x5 + x6 = 1 + 2` | `x7 = 3` |
| `0x0C` | `sd x7, 8(x8)` | `MEM[8 + x8] = x7` | MEM updated |
| `0x10` | `ld x11, 8(x8)` | `x11 = MEM[8 + x8]` | `x11 = 3` |
| `0x14` | `add x2, x3, x4` | `x2 = x3 + x4 = 3 + 4` | `x2 = 7` |
| `0x18` | `add x5, x4, x4` | `x5 = x4 + x4 = 4 + 4` | `x5 = 8` |
| `0x1C` | `sub x11, x12, x13` | `x11 = x12 - x13 = 12 - 13` | `x11 = -1` |
| `0x20` | `or x3, x4, x5` | `x3 = x4 OR x5 = 4 OR 8` | `x3 = 12` |
| `0x24` | `and x1, x6, x30` | `x1 = x6 AND x30 = 2 AND 30` | `x1 = 2` |
| `0x28` | `sub x10, x11, x11` | `x10 = x11 - x11 = -1 - (-1)` | `x10 = 0` |
| `0x2C` | `and x10, x13, x15` | `x10 = x13 AND x15 = 13 AND 15` | `x10 = 13` |
| `0x30` | `add x7, x8, x9` | `x7 = x8 + x9 = 8 + 9` | `x7 = 17` |
| `0x34` | `add x18, x19, x20` | `x18 = x19 + x20 = 19 + 20` | `x18 = 39` |
| `0x38` | `or x12, x13, x14` | `x12 = x13 OR x14 = 13 OR 14` | `x12 = 15` |
| `0x3C` | `sub x17, x18, x19` | `x17 = x1` [illegible] `x19 = 39 - 19` | `x17 = 20` |

**PIPELINING NO HAZARD(ENUF INSTRUCTIONS IN BETWEEN)**

```
0x003100b3
0x40310233
0x003172b3
0x00316333
0x009403b3
0x409403b3
0x00a40513
0x009465b3
0x00946633
0x010786b3
0x41078733
0x00a00013
0x0020e8b3
0x00000e93
0x00000f13
0x00000513
```

1. add x1, x2, x3
2. sub x4, x2, x3
3. and x5, x2, x3
4. or x6, x2, x3
5. add x7, x8, x9
6. Sub x7,x8,x9
7. addi x10, x8, 10
8. or x11, x8, x9
9. or x12, x8, x9
10. add x13, x15, x16
11. sub x14, x15, x16
12. addi x0, x0, 10//NOTE
13. or x17, x1, x2
14. addi x29, x0, 0
15. addi x30, x0, 0
16. addi x10, x0, 0

## Pipeline Execution Table

| Cycle | IF | ID | EX | MEM | WB |
|---|---|---|---|---|---|
| **CC0** | `add x1, x2, x3` | - | - | - | - |
| **CC1** | `sub x4, x2, x3` | `add x1, x2, x3` | - | - | - |
| **CC2** | `and x5, x2, x3` | `sub x4, x2, x3` | `add x1, x2, x3` | - | - |
| **CC3** | `or x6, x2, x3` | `and x5, x2, x3` | `sub x4, x2, x3` | `add x1, x2, x3` | - |
| **CC4** | `add x7, x8, x9` | `or x6, x2, x3` | `and x5, x2, x3` | `sub x4, x2, x3` | `add x1, x2, x3` |
| **CC5** | `sub x7, x8, x9` | `add x7, x8, x9` | `or x6, x2, x3` | `and x5, x2, x3` | `sub x4, x2, x3` |
| **CC6** | `addi x10, x8, 10` | `sub x7, x8, x9` | `add x7, x8, x9` | `or x6, x2, x3` | `and x5, x2, x3` |
| **CC7** | `or x11, x8, x9` | `addi x10, x8, 10` | `sub x7, x8, x9` | `add x7, x8, x9` | `or x6, x2, x3` |
| **CC8** | `or x12, x8, x9` | `or x11, x8, x9` | `addi x10, x8, 10` | `sub x7, x8, x9` | `add x7, x8, x9` |
| **CC9** | `add x13, x15, x16` | `or x12, x8, x9` | `or x11, x8, x9` | `addi x10, x8, 10` | `sub x7, x8, x9` |
| **CC10** | `sub x14, x15, x16` | `add x13, x15, x16` | `or x12, x8, x9` | `or x11, x8, x9` | `addi x10, x8, 10` |
| **CC11** | `addi x0, x0, 10` | `sub x14, x15, x16` | `add x13, x15, x16` | `or x12, x8, x9` | `or x11, x8, x9` |
| **CC12** | `or x17, x1, x2` | `addi x0, x0, 10` | `sub x14, x15, x16` | `add x13, x15, x16` | `or x12, x8, x9` |
| **CC13** | `addi x29, x0, 0` | `or x17, x1, x2` | `addi x0, x0, 10` | `sub x14, x15, x16` | `add x13, x15, x16` |
| **CC14** | `addi x30, x0, 0` | `addi x29, x0, 0` | `or x17, x1, x2` | `addi x0, x0, 10` | `sub x14, x15, x16` |
| **CC15** | `addi x31, x0, 0` | `addi x30, x0, 0` | `addi x29, x0, 0` | `or x17, x1, x2` | `addi x0, x0, 10` |
| **CC16** | `addi x10, x0, 0` | `addi x31, x0, 0` | `addi x30, x0, 0` | `addi x29, x0, 0` | `or x17, x1, x2` |
| **CC17** | - | `addi x10, x0, 0` | `addi x31, x0, 0` | `addi x30, x0, 0` | `addi x29, x0, 0` |
| **CC18** | - | - | `addi x10, x0, 0` | `addi x31, x0, 0` | `addi x30, x0, 0` |
| **CC19** | - | - | - | `addi x10, x0, 0` | `addi x31, x0, 0` |
| **CC20** | - | - | - | - | `addi x10, x0, 0` |

**LOAD/USE INSTRUCTIONS WITHOUT HAZARDS**

```
00843183
00883203
00000513
00000593
00000613
00000693
00000713
00443023
00000513
00000593
00000613
00000693
00000713
```

1. ld x3, 8(x8)
2. ld x4, 8(x16)
3. addi x10, x0, 0
4. addi x11, x0, 0
5. addi x12, x0, 0
6. addi x13, x0, 0
7. addi x14, x0, 0
8. sd x4, 0(x8)
9. addi x10, x0, 0
10. addi x11, x0, 0
11. addi x12, x0, 0
12. addi x13, x0, 0
13. addi x14, x0, 0

**FORWARDING UNIT WITH DOUBLE AND SINGLE FORWARDING WITHOUT LD/USE
(NTNG BUT NO HAZARD DETECTION)**

SINGLE DATA HAZARD :

```
0x00308133
0x00510633
0x002306b3
0x00210733
```

```
add x2, x1,x3  //x2=4
Add x12,x2,x5  //x12=9
add x13,x6,x2 //x13=10
add x14,x2,x2//x14=8
```

DOUBLE DATA HAZARD:

```
0x002080b3
0x003080b3
0x004080b3
```

```
add x1,x1,x2
add x1,x1,x3
add x1,x1,x4
```

**HAZARD UNIT(ALMOST  PEAK LIFE)**

```
0x01043103
0x00228233
0x40610433
0x002204b3
```

```
Ld x2,16(x8)
Add x4,x5,x2
Sub x8,x2,x6
Add x9,x4,x2
```

1. **For control hazard**
   **Same as seq_branch check**

   1.0x00418133
2. 0x004202b3
3. 0x00a00513
4. 0x00a00593
5. 0x40310233
6. 0x00b50463
7. 0x01e370b3
8. 0x40b58533
9. 0x00f6f533
10. 0x00940 3b3
11. 0x01498933
12. 0x00e6e633
13. 0x00418133
14. 0x004202b3
15. 0x00a00513
16. 0x00a00593

1.add x2, x3, x4
2. add x5,x4,x4
3. addi x10, x0, 10
4. addi x11, x0, 10
5. sub x4, x2, x3
6. beq x10, x11, 8 (9th pc)
7. and x1, x6, x30
8. sub x10, x11, x11
9. and x10, x13, x15
10. add x7, x8, x9
11. add x18, x19, x20
12. or x12, x13, x14
13. add x2, x3, x4
14. add x5,x4,x4
15. addi x10, x0, 10
16. addi x11, x0, 10

**Including load store beq**

1. 0x00803103
2. 0x00803203
3. 0x00410463
4. 0x00418133
5. 0x004202b3
6. 0x40d605b3
7. 0x005261b3
8. 0x01e370b3
9. 0x40b58533
10. 0x00f6f533
11. 0x009403b3

1.

   Ld x2,8(x0)
2. Ld x4,8(x0)
3. Beq x2,x4,8 //jump to 6th instruction
4. add x2, x3, x4 x2=7
5. add x5, x4, x4 x5=8
6. sub x11, x12, x13 x11=-1
7. or x3, x4, x5 x3=5
8. and x1, x6, x30 x1=36
9. sub x10, x11, x11 x10=22
10. and x10, x13, x15 x10=27….
11. add x7, x8, x9

THIS HEX FILE IS ONE INDEX, SO PC = 24/4=6, U GO CHECK 7TH INSTRUCTION