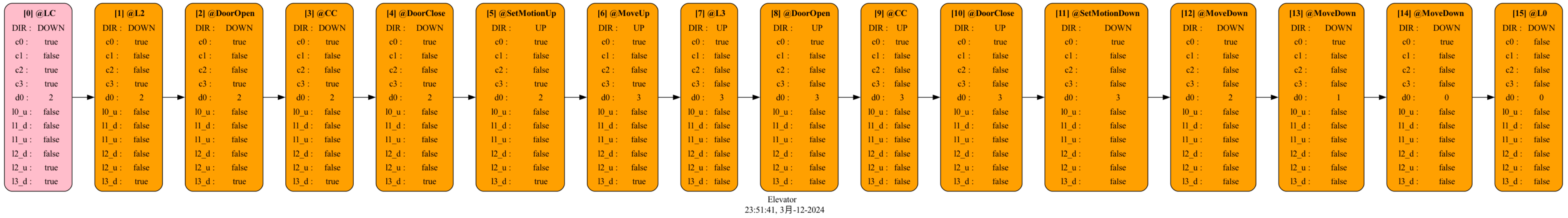
1) if DIR==DOWN and any one of cc below is on, the elevator keeps current DIR.

2) if DIR==UP and any one of cc above is on, the elevator keeps current DIR.

3) if DIR==DOWN and none of cc below is on, and one of cc above is on, the elevator changes its DIR to UP.

4) if DIR==UP and none of cc above is on, and one of the cc below is on, the elevator changes its DIR to DOWN.



	[0] @LC	[1] @L2	[2] @DoorOpen	[3] @CC	[4] @DoorClose	[5] @SetMotionUp	[6] @MoveUp	[7] @L3	[8] @DoorOpen	[9] @CC	[10] @DoorClose	[11] @SetMotionDown	[12] @MoveDown	[13] @MoveDown	[14] @MoveDown	[15] @L0
DIR :	DOWN	DOWN	DOWN	DOWN	DOWN	UP	UP	UP	UP	UP	UP	DOWN	DOWN	DOWN	DOWN	DOWN
c0 :	true	true	true	true	true	true	true	true	true	true	true	true	true	true	true	false
c1 :	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false
c2 :	true	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false
c3 :	true	true	true	true	true	true	true	false	false	false	false	false	false	false	false	false
d0 :	2	2	2	2	2	2	3	3	3	3	3	3	2	1	0	0
l0_u :	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false
l1_d :	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false
l1_u :	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false
l2_d :	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false
l2_u :	true	false	false	false	false	false	false	false	false	false	false	false	false	false	false	false
l3_d :	true	true	true	true	true	true	true	false	false	false	false	false	false	false	false	false

```
Path found: <15>
LC->L2->DoorOpen->CC->DoorClose->SetMotionUp->MoveUp->L3->DoorOpen->CC->DoorClose->SetMotionDown->MoveDown->MoveDown->MoveDown->
L0
```

Issue Found During Testing:

Between Step 4 and Step 5, the transition should be from `DoorClose -> SetMotionDown`, not `SetMotionUp`.

When the elevator is at the 2nd floor (`L2`) and the door closes, it evaluates the condition for `SetMotionUp`. The condition is:

```
edge { DoorClose -> SetMotionUp
    where (d0==0 && (c1 || c2 || c3)) ||
          (d0==1 && (c2 || c3)) ||
          (d0==2 && (c3));
}
```

This condition is met.

At floor 2 (`d0==2`), this condition checks if there is a call from the 3rd floor (`c3`). If there is a call from the 3rd floor, the elevator sets direction to up (`SetMotionUp`), even if there's a call from the 0th floor (`c0 == true`). This indicates a priority rule in the elevator logic: even when there are calls from lower floors, the elevator will prioritize responding to upper floor calls.

That's why `SetMotionUp` was triggered.

Solution:

Add an extra condition to ensure `SetMotionUp` is not triggered when there are calls from lower floors.

To fix the logic, it's necessary to check for any requests from lower floors before deciding to move upward. If lower-floor requests exist, the elevator should prioritize moving downward.

So, in the `edge { DoorClose -> SetMotionUp }`, a check for requests from lower floors has been added.

Before:

```
edge { DoorClose -> SetMotionUp
    where (d0==0 && (c1 || c2 || c3)) ||
          (d0==1 && (c2 || c3)) ||
          (d0==2 && (c3));
    }
       edge { DoorClose -> SetMotionDown
    where ((d0==3) && (c0 || c1 || c2)) ||
        ((d0==2) && (c0 || c1)) ||
        ((d0==1) && c0);
   }
```

After:

```
edge { DoorClose -> SetMotionUp
where ((DIR == #UP && !((d0 > 0 && c0) || (d0 > 1 && c1) || (d0 > 2 && c2))) ||
(DIR == #DOWN && !((d0 > 0 && c0) || (d0 > 1 && c1) || (d0 > 2 && c2)))) &&
((d0 < 3 && c3) || (d0 < 2 && c2) || (d0 < 1 && c1));
}



edge { DoorClos -> SetMotionDown
where ((DIR == #DOWN && !((d0 < 3 && c3) || (d0 < 2 && c2) || (d0 < 1 && c1))) ||
(DIR == #UP && !((d0 < 3 && c3) || (d0 < 2 && c2) || (d0 < 1 && c1)))) &&
((d0 > 0 && c0) || (d0 > 1 && c1) || (d0 > 2 && c2));
}
```

Rule 1 & 2 - Maintain Current Direction

These rules state that the elevator should continue in its current direction if there are calls in that direction.

For **Upward Direction**:

If `DIR == #UP` and there are no lower-floor calls (`!((d0 > 0 && c0) || ...)`) but there are upper-floor calls, the elevator continues or switches to move upward. This satisfies Rule 2.

For **Downward Direction**:

If `DIR == #DOWN` and there are no upper-floor calls, but lower-floor calls exist, the elevator continues or switches to move downward. This satisfies Rule 1.

Rule 3 & 4 - Change Direction

These rules handle direction change when no calls exist in the current direction, but there are calls in the opposite direction.

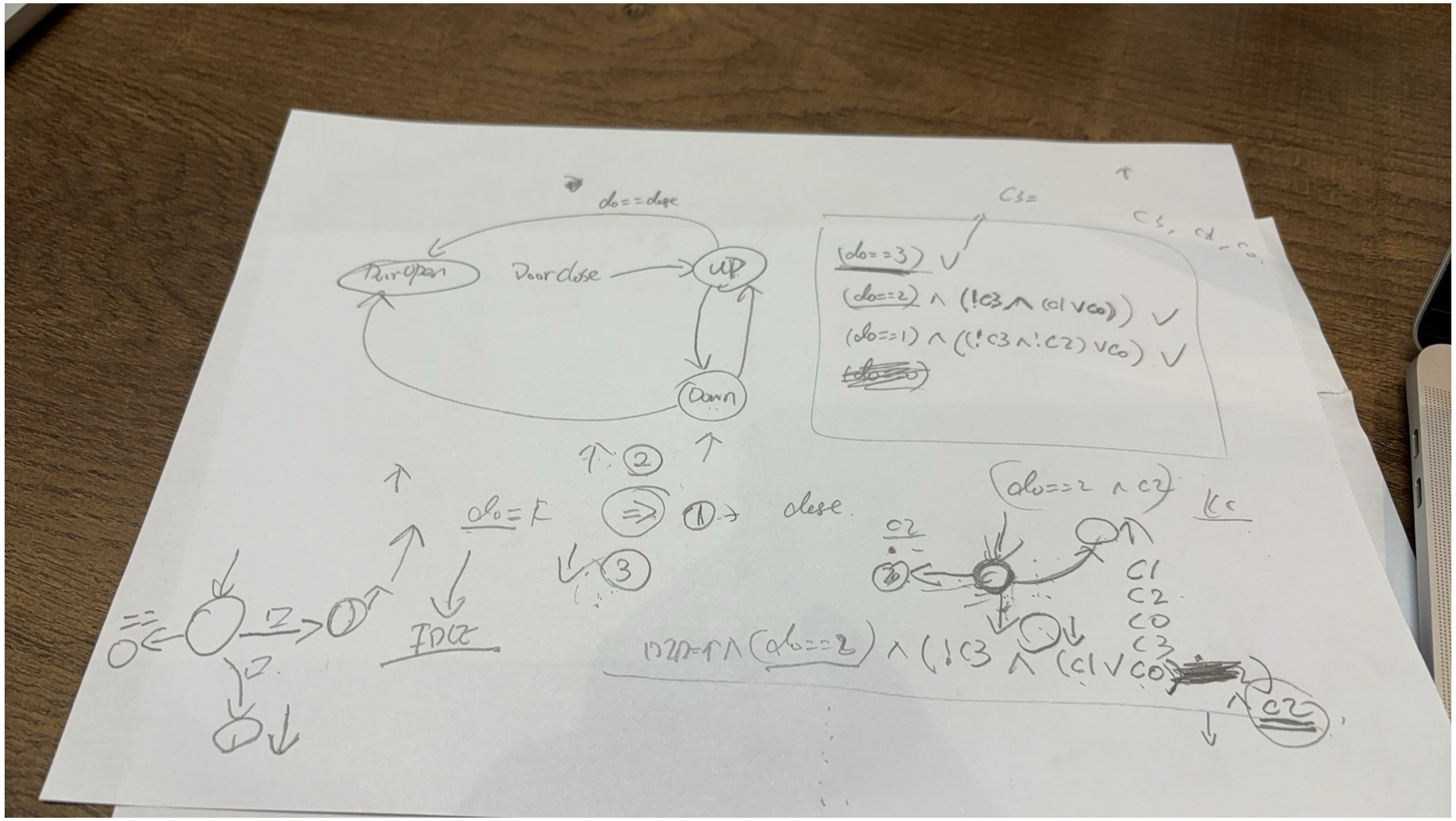
From DOWN to UP:

The first `SetMotionUp` rule checks this condition when `DIR == #DOWN` and no calls below, but calls above exist. This meets Rule 3.

From UP to DOWN:

Similarly, the `SetMotionDown` rule handles switching from UP to DOWN when the condition is met. This satisfies Rule 4.

⚠️ **Note:** **New logic flow diagram needs to be updated. A new node should be added.**



do==close
Door open
Door close
UP
Down
C3=
C3, c1, c0.
(do==3) ✓
(do==2) ∧ (!C3 ∧ (c1 ∨ c0)) ✓
(do==1) ∧ ((!C3 ∧ !C2) ∨ C0) ✓
~~(do==0)~~
do=F
close
(do==2 ∧ c2)
C1
C2
C0
C3
IDLE
(do==2) ∧ (!C3 ∧ (c1 ∨ c0)
C2