Elevator Control System Documentation

This document provides a detailed analysis of the `.circ` file, which represents an **elevator control system** designed in Logisim. The system consists of several subcircuits, each performing specific functions. Below is the breakdown of each subcircuit, its functionality, logic diagrams, truth tables, and considerations.

Table of Contents

Main Circuit

Functionality

The **main circuit** integrates all subcircuits to control the elevator system. It includes components like LEDs, buttons, clocks, gates, flip-flops, and encoders. The circuit manages floor selection, elevator movement, and state transitions.

Components

- **LEDs**: Indicate elevator states (e.g., current floor, direction).
- **Buttons**: Inputs for floor selection or control commands.
- **Clocks**: Provide timing signals for synchronization.
- **Gates**: Perform logical operations (AND, OR, NOT).
- **Flip-Flops**: Store state information (e.g., current floor).

Logic Diagram

Main Circuit Logic Diagram

Truth Tables

AND Gate

A	B	Output
0	0	0
0	1	0

A	B	Output
1	0	0
1	1	1

OR Gate

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

NOT Gate

A	Output
0	1
1	0

D Flip-Flop

Clock	D	Q (Next State)
↑	0	0
↑	1	1
-	X	Q (Hold)

Data Sheet

Component	Description	Inputs	Outputs
LED	Floor indicator	State signal	Visual output
Button	Floor selection input	User input	Signal
Clock	Timing signal	-	Clock signal
AND Gate	Logical AND operation	2 inputs	1 output
OR Gate	Logical OR operation	2 inputs	1 output
NOT Gate	Logical NOT operation	1 input	1 output
Flip-Flop	State storage	Clock, Reset	State output

Considerations

- Ensure proper synchronization between subcircuits.
- Verify that timing signals (clocks) are correctly configured.
- Test all input combinations to ensure correct behavior.

Encoder

Functionality

The **encoder** converts multiple inputs (e.g., floor selection buttons) into a binary output. It encodes floor selection inputs into a binary representation for processing by other components.

Components

- **Inputs**: Floor selection buttons (`0`, `1`, `2`, `3`).
- **Outputs**: Binary representation (`B1`, `B0`).

Logic Diagram

Encoder Logic Diagram

Truth Table

Input (Floor)	B1	B0
0	0	0
1	0	1
2	1	0
3	1	1

Data Sheet

Component	Description	Inputs	Outputs
Button	Floor selection input	User input	Signal
Encoder	Binary encoding	4 inputs	2 outputs

Considerations

- Ensure the encoder correctly maps inputs to outputs.
- Verify that the binary output is consistent with the input.

Comparateur (Comparator)

Functionality

The **comparator** compares two binary inputs and produces outputs indicating equality, greater-than, or less-than relationships. It compares the current floor with the target floor to determine the elevator's direction.

Components

- **Inputs**: Binary values (`B1`, `B0`, `O1`, `O2`).
- **Outputs**: Comparison results (`=`, `<`, `>`).

Logic Diagram

Comparator Logic Diagram

Truth Table

B1	B0	O1	O2	=	<	>
0	0	0	0	1	0	0
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	0
0	1	0	0	0	0	1
0	1	0	1	1	0	0
0	1	1	0	0	1	0
0	1	1	1	0	1	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	1	0	0
1	0	1	1	0	1	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	1	0	0

Data Sheet

Component	Description	Inputs	Outputs
Comparator	Binary comparison	4 inputs	3 outputs

Considerations

- Ensure the comparator handles all possible input combinations.

- Verify that the outputs correctly reflect the comparison results.

Counter

Functionality

The **counter** counts based on input signals and outputs the current count. It tracks the elevator's current floor or the number of floors traveled.

Components

- **Inputs**: Clock signal (`CLK`), reset signal.
- **Outputs**: Current count (`O1`, `O2`).

Logic Diagram

Counter Logic Diagram

Truth Table

Clock	Reset	O1	O2
↑	0	Increment	
↑	1	0	0

Data Sheet

Component	Description	Inputs	Outputs
Counter	Counts input signals	Clock, Reset	2 outputs

Considerations

- Ensure the counter resets correctly when needed.
- Verify that the count increments or decrements as expected.

Triger (Trigger)

Functionality

The **trigger** acts as a state machine, transitioning between states based on inputs. It controls the elevator's movement (e.g., start, stop, change direction).

Components

- **Inputs**: Control signals (`0`, `1`, `2`, `3`).
- **Outputs**: State outputs (`0`, `1`, `2`, `3`).

Logic Diagram

Trigger Logic Diagram

Truth Table

Input	Current State	Next State
0	0	1
1	1	2
2	2	3
3	3	0

Data Sheet

Component	Description	Inputs	Outputs
Trigger	State machine	4 inputs	4 outputs

Considerations

- Ensure the state transitions are well-defined.
- Verify that the outputs correctly reflect the current state.

LiftingMachine (Lifting Machine)

Functionality

The **lifting machine** controls the elevator's lifting mechanism. It processes inputs (e.g., floor requests) and generates control signals for the motor.

Components

- **Inputs**: Floor selection buttons (`i0`, `i1`, `i2`).
- **Outputs**: Control signals (`O'1`, `O'2`).

Logic Diagram

Lifting Machine Logic Diagram

Truth Table

i0	i1	i2	O'1	O'2
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	0
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1

Data Sheet

Component	Description	Inputs	Outputs
LiftingMachine	Controls lifting mechanism	3 inputs	2 outputs

Considerations

- Ensure the lifting mechanism responds correctly to inputs.
- Verify that the control signals are generated as expected.

Lifting Machine 2.0

Functionality

This is an updated version of the **lifting machine** subcircuit with additional features or optimizations. It provides more precise control over the elevator's movement.

Components

- **Inputs**: Floor selection buttons (`i0`, `i1`, `i2`).
- **Outputs**: Control signals (`S2`).

Logic Diagram

Lifting Machine 2.0 Logic Diagram

Truth Table

i0	i1	i2	S2
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Data Sheet

Component	Description	Inputs	Outputs
LiftingMachine 2.0	Enhanced lifting control	3 inputs	1 output

Considerations

- Ensure compatibility with other subcircuits.
- Verify that the updated logic improves performance or functionality.

Conclusion

This documentation provides a detailed analysis of each subcircuit in the `.circ` file, including truth tables for all components. You can now insert the logic diagrams for each subcircuit and add any additional considerations or notes. Let me know if you need further assistance!