Linked List for Bits

Problem Statement:

Design a linked list of 1-bit numbers with a capacity for 8 bits. It should have an insert input with 3-bit input for position and a bit for actual data. It should also have a delete input with 3-bit position input and a bit for output. Every position should know whether there is data available or not. This should be set on insert and cleared on delete.

Input

Clock Pulse	**CP**
Insert	**IA**
Delete	**DA**
Position	**P0…P2**
Insert Data	**ID**

Output

Delete Data	**DD**
Nothing to Delete	**DF** (delete failure)
Data Already present	**IF** (insert failure)

Introduction:

A linked list is a linear data structure connected with the help of links (in this case position). Each link contains a connection to another link.

In this project, we have designed a circuit that meets the requirements stated in the problem statement which are as follows. First, the insertion of data (given by the user) is done at a specific position (also given by the user) of the 8-bit linked list. Further, when a position is filled (denoted by a separate bit) no data insertion operation can be performed (**Insert Failure**) at the accessed position. Lastly, the **Data Deleted** operation can be performed at any position if there is some data already present otherwise data deletion cannot be done (**Delete Failure**) at the position accessed by the user.

Methodology:

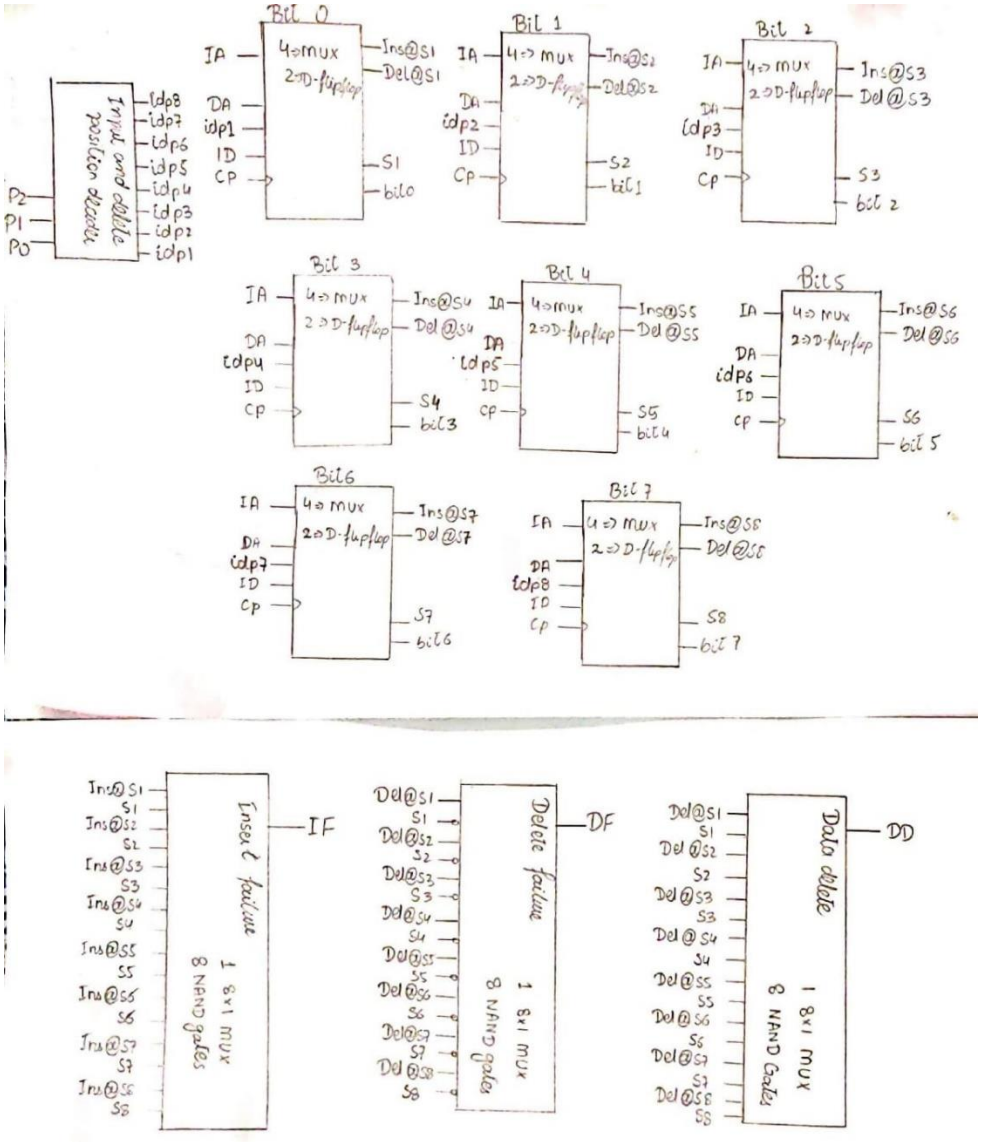
In order to meet the design requirements, we need the combination of several logic circuits that can perform the required operations. The detail of the required logic components and their use is as follows.

- **Clock Pulse**
  One 4x1 Multiplexer for clock control.
- **Position Input**
  One 3x8 decoder is used for the determination of the position entered by the user at which insertion and deletion should take place.
- **Insert, Delete and Reset inputs**
  Two separate STDP pushbuttons get the input from a user to perform Insertion and Deletion of data while an STDP pushbutton is used to reset the whole circuit.
- **Data Input and Enablers**
  Three binary switches, one for the **Input Data** bit and the other two are used as enablers for decoders, multiplexers, and as the input tags for the multiplexers.
- **Insertion and Deletion**
  Eight 4x1 Multiplexers are used to determine which operation (insertion or deletion) the user wants to perform.
- **For Position availability**
  The combination of eight 8x1 Multiplexer and D-flipflop is used to change the status of the **Data Availability** bit after successful Data Insertion or Data Deletion.

- **For Data insertion**
  The combination of another eight 8x1 Multiplexers and D-flipflop is used to store **Insert Data** input at the linked list bit of specified position after verifying the validity of the operation (i.e., Data already present or not).
- **For Insert Failure**
  Eight NAND gates and one 8x1 Multiplexer are used to indicate insert failure if data is already present at the accessed position.
- **For Data Failure**
  Eight NAND gates with one inverse input each and one 8x1 Multiplexer are used to indicate delete failure if no data is present at the accessed position.
- **For Data Delete**
  Eight NAND gates and one 8x1 Multiplexer is used to indicate data deletion if data is successfully deleted at the accessed position.

High-level Circuit Diagram:

The high-level circuit diagram or block diagram of the designed circuit is as follows:



Where P0, P1, and P2 are position inputs given by the user. IA, DA, ID, and CP are Insert, Delete, Insert Data, and Clock Pulse respectively. Ins@Sx denotes the position at which the user wants to perform the insert operation (After verification of data availability). Del@Sx denotes the position at which the user wants to perform the delete operation (After verification of data availability). IF, DF and DD represent Insert Failure, Delete Failure, and Data Delete respectively. The idpx denotes the position determined through input provided by the user. Where x = 1,2,3,4,….8.

Relation of Inputs and Outputs:

The following two tables show the relationship between the inputs and the corresponding outputs for both Insert and Delete operations. And it also shows the next state of data availability bit after the operation.

Inputs			Outputs		
S(t) (Data availability)	**IA** (Insert)	**ID** (Insert Data)	**Linked Bit** (Output data)	**IF** (Insert Failure)	**S(t+1)** (Data availability)
0	0	0	Unchanged	0	0
0	0	1	Unchanged	0	0
0	1	0	0	0	1
0	1	1	1	0	1
1	0	0	Unchanged	0	1
1	0	1	Unchanged	0	1
1	1	0	Unchanged	1	1
1	1	1	Unchanged	1	1

Table 1 – For Insert Operation

Inputs		Outputs		
S(t) (Data availability)	**DA** (Delete)	**DD** (Data Delete)	**DF** (Delete Failure)	**S(t+1)** (Data availability)
0	0	0	0	0
0	1	0	1	0
1	0	0	0	1
1	1	1	0	0

Table 2 – For Delete Operation

Internal Circuitry:

The following figures are the internal circuitry of this project:

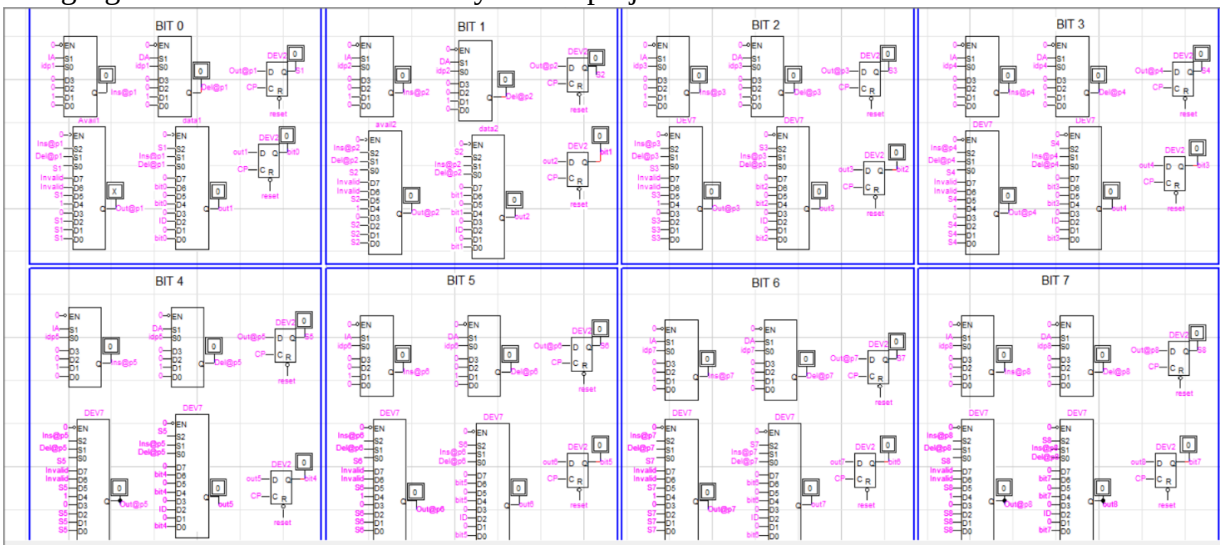


Figure 1: Internal circuitry of Data availability and Linked list of bits

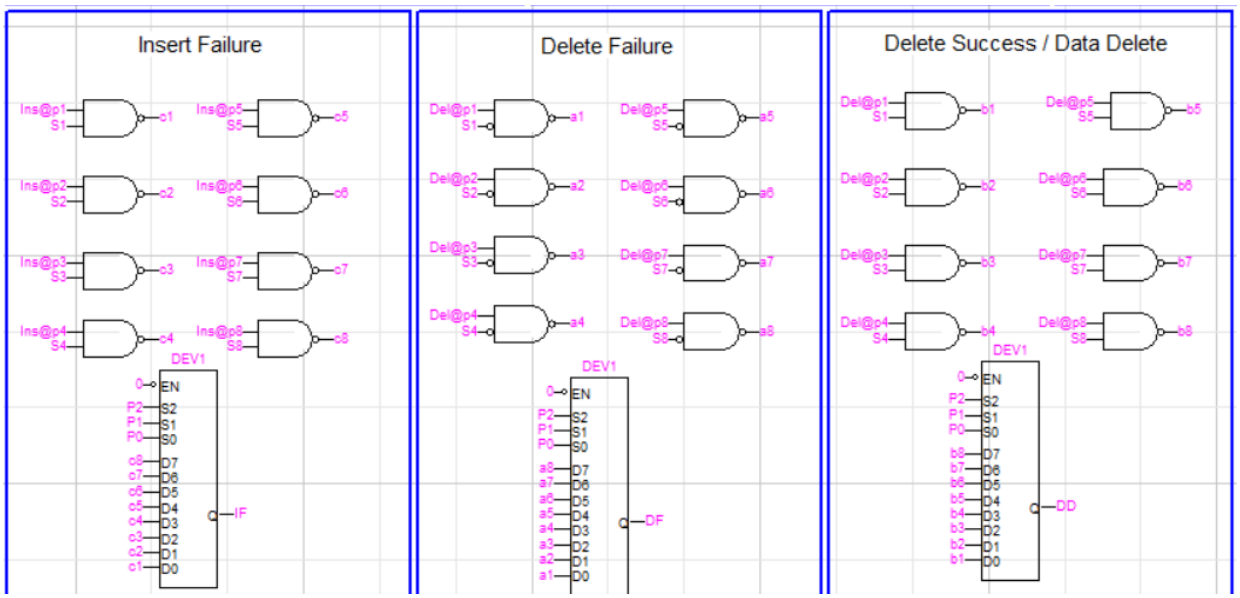


Figure 2: Internal circuitry of Insert Failure, Data Failure, and Data Delete

Results:

Following are the different scenarios of Inputs and their corresponding outputs:

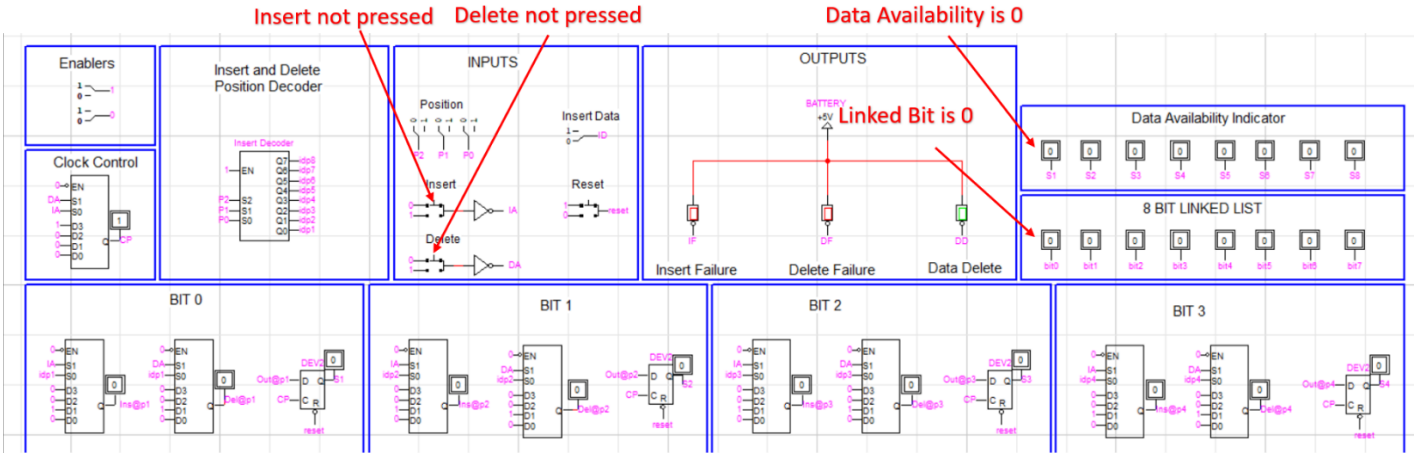


Figure 3: Initial State of circuit

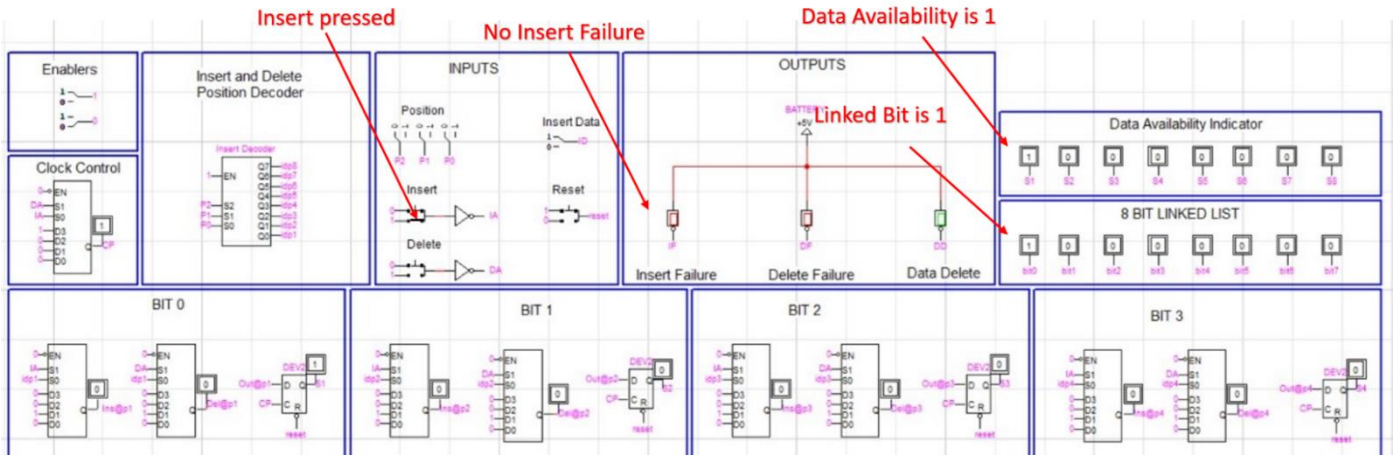


Figure 4: Insertion of Data

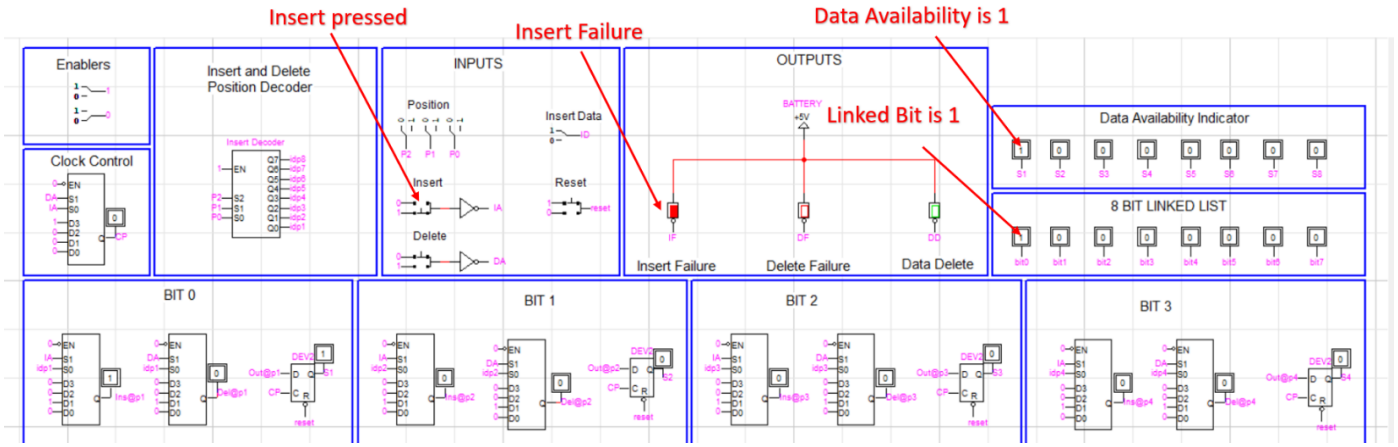


Figure 5: Insert Failure

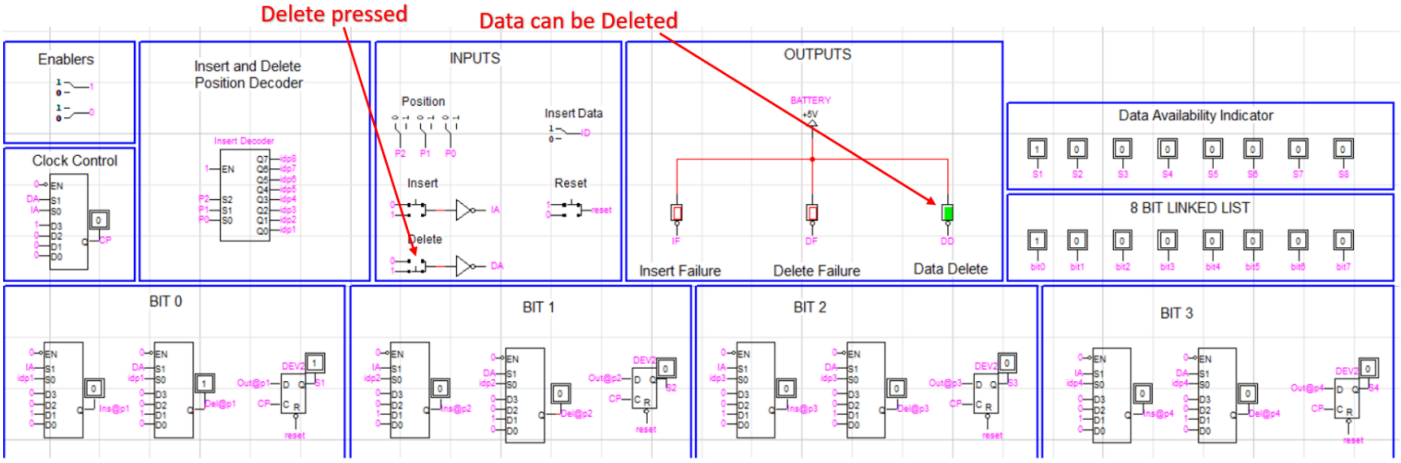


Figure 6: Deletion

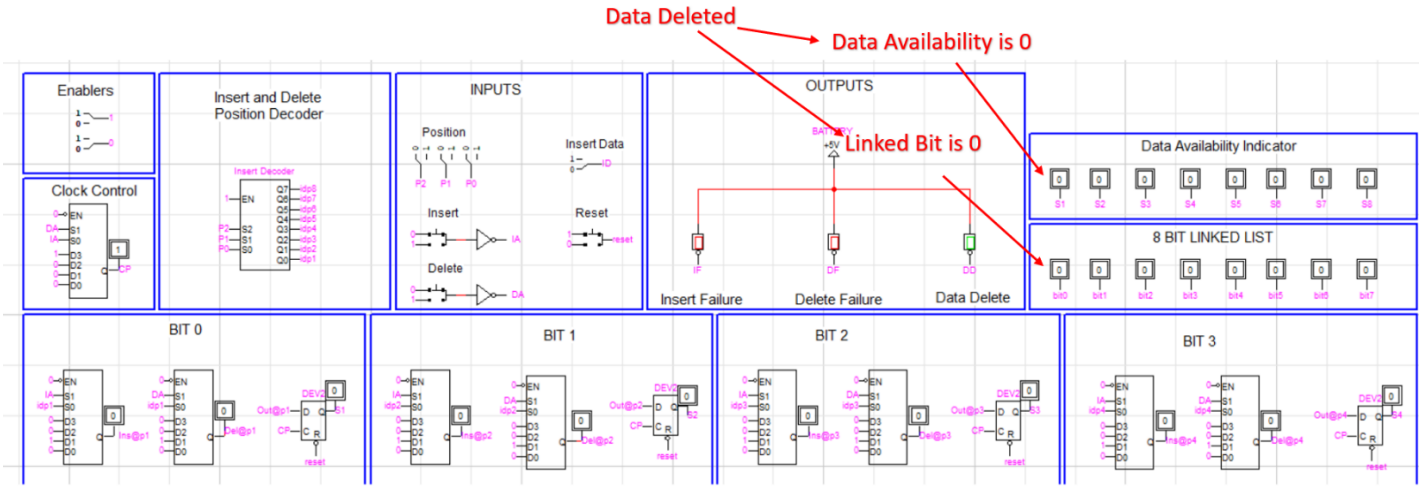


Figure 7: Data Deleted

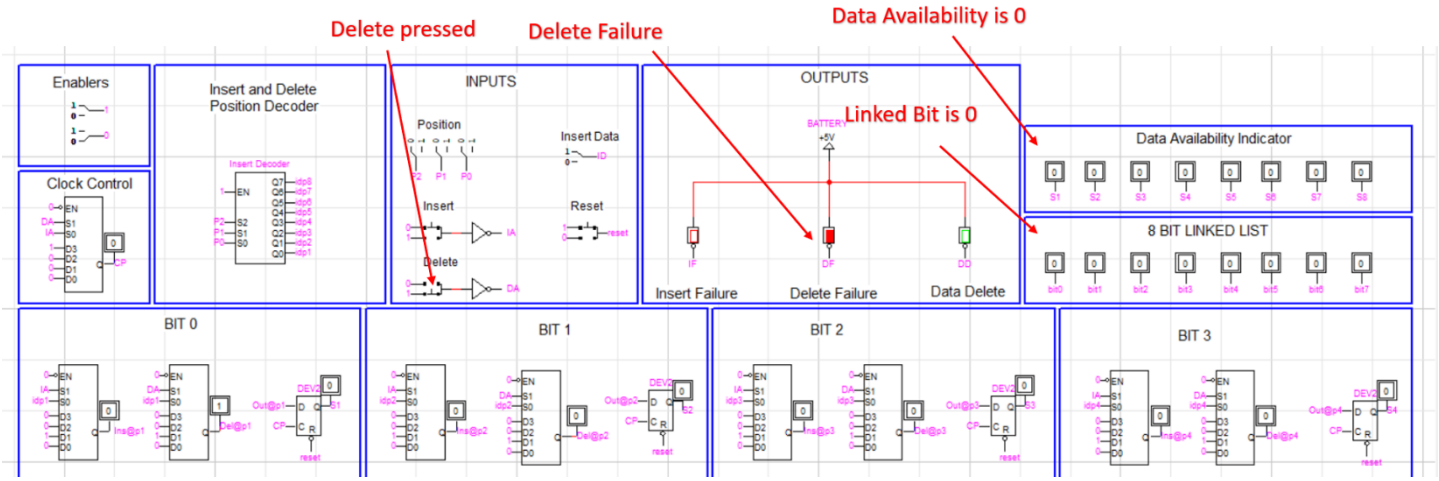


Figure 8: Delete Failure

References:

1. nesoacademy. (2020, June 22). *Introduction to linked list*. YouTube. Retrieved May 16, 2022, from https://www.youtube.com/watch?v=R9PTBwOzceo&ab_channel=NesoAcademy
2. *Data Structure - Linked List*. (n.d.).Tutorialspoint. Retrieved May 19, 2022, from https://www.tutorialspoint.com/data_structures_algorithms/linked_lists_algorithm.html