Disaster Warning Device

Team Members:

1. 241CS140, P Sai Jishnu, saijishnup.241cs140@nitk.edu.in, 9059645654
2. 241CS161, U Venkatesh, utkoorvenkatesh.241cs161@nitk.edu.in, 9182974835
3. 241CS163, Vikash Patel, vikashpatel.241cs163@nitk.edu.in, 6267521706

GitHub Link:

https://github.com/thezishnu/S1-T19-25-26

Abstract

Natural disasters such as floods, cyclones, earthquakes, and tsunamis cause significant damage to life and property. Early detection and warning systems play a crucial role in minimizing these impacts. This project presents a digital logic–based **Disaster Warning Device** that automatically identifies potential disasters using simple sensor-derived 2-bit inputs for four environmental parameters: *Rainfall, Wind Speed, Seismic Activity*, and *Sea Level*. Each parameter is represented by a 2-bit code indicating intensity levels — Low, Medium, High, or Very High. The system processes these inputs through combinational logic circuits using AND, OR, and comparator operations to determine the occurrence of specific disasters. Logical expressions detect conditions for Flood, Cyclone, Earthquake, and Tsunami, and the results are encoded through a priority encoder to ensure that only the highest-priority disaster is indicated. A **mode selection feature** allows the device to operate in two modes: *Unique Mode*, where only the highest-priority disaster LED glows, and *Multi Mode*, where all detected disasters are displayed simultaneously. The final output is visualized using LEDs corresponding to each disaster type. This simple, low-cost, and hardware-efficient design demonstrates practical applications of combinational and sequential digital logic concepts in real-world safety and alert systems, making it an effective educational project for digital design learning.

1 Introduction

Natural disasters are among the most destructive and unpredictable events faced by humanity, capable of causing immense damage to life, property, and the environment. Events such as floods, cyclones, earthquakes, and tsunamis have repeatedly disrupted civilizations throughout history, leading to loss of life, displacement, and economic instability. Despite tremendous scientific progress, many disasters still occur without sufficient warning, leaving vulnerable communities unprepared to respond effectively. The destruction caused extends beyond physical damage—it includes emotional and psychological trauma, long-term social disruption, and financial setbacks that can take years to recover from. With the growing effects of climate change, deforestation, and unplanned urbanization, the frequency and intensity of such calamities have increased significantly. Floods, cyclones, and earthquakes often strike with little notice, damaging critical infrastructure, breaking communication networks, and isolating affected populations. Hence, the ability to detect and provide early warning for potential disasters has become essential to safeguard lives and reduce economic losses worldwide.

The consequences of natural disasters are far more severe in rural and underdeveloped regions where access to advanced technology, communication, and monitoring systems is minimal. Many such areas still rely on manual observation methods, such as checking river levels or monitoring weather changes visually, which are neither accurate nor timely. Due to poor connectivity, limited resources, and inadequate disaster management systems, rural communities remain highly vulnerable to natural calamities. The lack of cost-effective and sustainable warning systems results in thousands of preventable deaths and large-scale property damage every year. In this context, there is a pressing humanitarian and technological need for simple, reliable, and low-cost devices capable of identifying early signs of disaster conditions. Such systems should be able to operate independently of internet or satellite infrastructure and provide direct, real-time alerts through intuitive visual or audible indicators. This forms the basis for the development of a logic-based hardware system capable of interpreting environmental data through combinational and sequential digital circuits, offering a practical approach to early disaster detection and community safety.

The proposed project, titled **"Disaster Warning Device Using Combinational and Sequential Logic Circuits"**, aims to design an affordable, efficient, and educational prototype for early disaster indication. The system conceptually receives simple two-bit digital inputs representing intensity levels of four environmental parameters—rainfall, wind, seismic activity, and sea level—and processes them through predefined Boolean logic expressions to identify potential disaster conditions. Logical outputs corresponding to flood, cyclone, earthquake, and tsunami are then fed into a priority encoder, which ensures that when multiple disasters are detected, only the most critical one is displayed first. A **Mode Selection** feature enhances the design's flexibility, allowing it to operate in two modes: *Unique Mode*, where only the highest-priority LED glows, and *Multi Mode*, where all active disasters are displayed simultaneously. Built entirely from basic digital components such as gates, decoders, and multiplexers, the system is low-cost, durable, and easy to implement for academic and demonstration purposes. Its modular architecture allows for future enhancements, such as adding memory storage, alarms, or wireless interfaces for remote monitoring.

In summary, this project not only provides a practical model for real-time disaster indication but also serves as an educational demonstration of digital logic principles applied to real-world challenges. It highlights how simple, hardware-based technology can contribute to safety awareness and early response, particularly in rural or low-resource settings. By combining affordability, simplicity, and social relevance, the Disaster Warning Device exemplifies the power of engineering innovation directed toward community resilience and public safety.

2 Functional Diagram

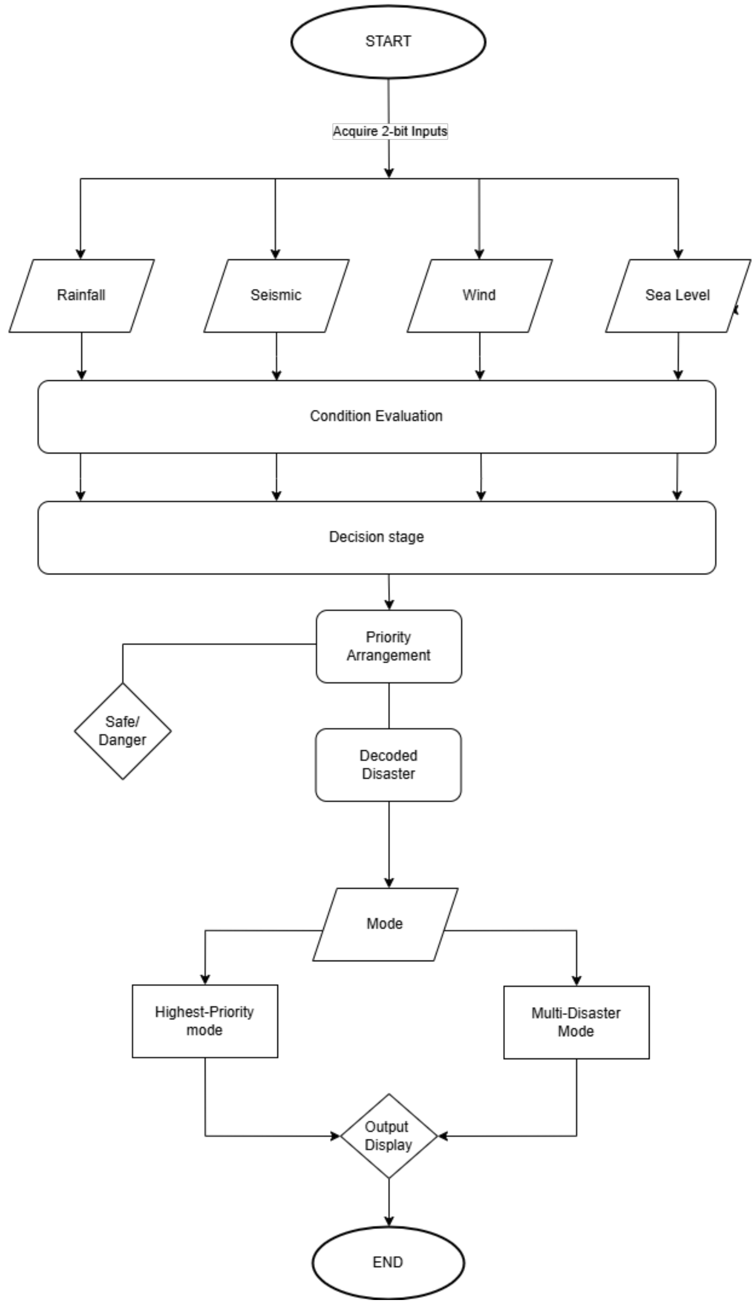


Figure 1: Functional Flowchart of the Disaster Warning Device

The flowchart in Fig. 1 represents the functional operation of the proposed **Disaster Warning Device**. It outlines the step-by-step data flow from input acquisition to final output display. The process begins at the **Start** block, where the system initializes to receive environmental data from four primary parameters — *Rainfall*, *Wind Speed*, *Seismic Activity*, and *Sea Level*. Each parameter is represented by a 2-bit digital input that defines the intensity level as:

$$00 = \text{Low}, \quad 01 = \text{Medium}, \quad 10 = \text{High}, \quad 11 = \text{Very High}.$$

These eight digital input bits are grouped and forwarded to the **Condition Evaluation Stage**, which contains four separate logic decision blocks. Each block corresponds to one disaster — **Flood**, **Cyclone**, **Earthquake**, or **Tsunami** — and checks whether the incoming input combinations satisfy the defined threshold conditions for each event. If the respective condition is true, that disaster's detection output is set to logic 1; otherwise, it remains 0.

The four outputs from this stage then proceed to the **Detection and Priority Encoding Block**. Here, the system assigns a binary code to represent the most severe disaster that has been detected. The disasters are arranged in descending order of priority as:

$$\text{Tsunami (11)} > \text{Earthquake (10)} > \text{Cyclone (01)} > \text{Flood (00)}.$$

This ensures that when multiple disaster conditions occur simultaneously, the most critical one is displayed first.

Next, the 2-bit output code generated by the encoder is fed into a **Decoder and Display Section**. The decoder activates one of four output lines, each corresponding to a specific LED that represents a disaster condition — Flood, Cyclone, Earthquake, or Tsunami. Only the line matching the encoded output becomes logic HIGH, ensuring that the correct LED glows for the detected event.

The system then enters the **Mode Selection Block**, which allows the user to choose how the disaster indications are displayed. When the *Mode Input* is set to 0 (**Unique Disaster Mode**), only the highest-priority LED is illuminated. When the mode is set to 1 (**Multi-Disaster Mode**), all LEDs for the active disasters light up simultaneously. This adds flexibility for both alert indication and testing purposes.

Finally, the process concludes at the **Output Display Stage**, where the LED indicators present a clear visual warning based on the detected disaster conditions. In unique mode, a single LED glows to indicate the most severe disaster, while in multi mode, multiple LEDs may glow to display all detected events simultaneously. The process then terminates at the **End** block, completing one functional cycle of the Disaster Warning Device.

Thus, the flowchart effectively summarizes the complete operational sequence — from input reception, condition analysis, and priority encoding to mode-based display — providing a clear visual understanding of how the device identifies and displays disaster conditions.

3 Design

This section presents the design of the proposed system, including the internal modules, truth tables, and logical expressions used for disaster detection. The complete **Functional Block Diagram** of the **Disaster Warning Device** is shown in Fig. 2. The diagram illustrates how environmental inputs are processed through logical comparators, a priority encoder, a decoder, and a mode-controlled display section.

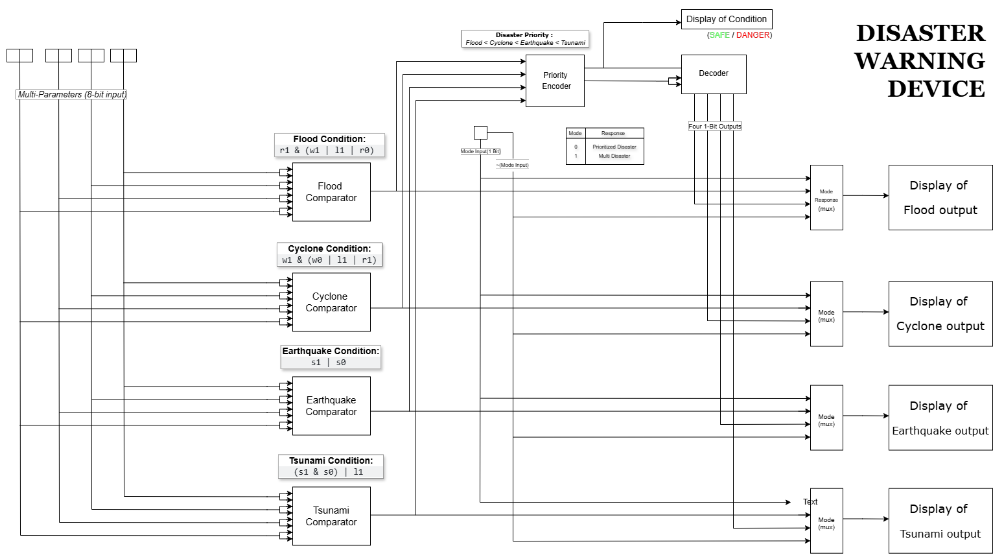


Figure 2: Functional Block Diagram of the Disaster Warning Device

3.1 Functional Table

Table 1: Functional Table of the Disaster Warning Device

r_1r_0	w_1w_0	s_1s_0	l_1l_0	F	C	E	T	Status	Mode	Final Output
10	00	00	00	1	0	0	0	Danger	0	F
11	01	00	10	1	1	0	0	Danger	1	F, C
00	00	01	00	0	0	1	0	Danger	0	E
00	00	11	10	0	0	1	1	Danger	1	E, T
10	10	00	01	1	1	0	0	Danger	1	F, C
01	11	10	01	1	1	1	0	Danger	1	F, C, E
11	11	11	11	1	1	1	1	Danger	1	F, C, E, T
00	00	00	00	0	0	0	0	Safe	0	None

Each 2-bit pair ($r_1r_0, w_1w_0, s_1s_0, l_1l_0$) represents the intensity levels of *Rainfall*, *Wind Speed*, *Seismic Activity*, and *Sea Level* respectively. The outputs (F, C, E, T) indicate the detected disasters (1 = Active, 0 = Inactive). The **Status** column shows whether the system is in a **Safe** or **Danger** state. The **Mode** column indicates the display configuration:

- **Mode 0 – Unique Mode:** Only the highest-priority disaster LED glows.
- **Mode 1 – Multi Mode:** All LEDs of active disasters glow simultaneously.

The **Final Output** column displays which LEDs are illuminated according to the selected mode.

3.2 Formulas Used

1. **Disaster Detection Equations** Each disaster condition is represented by a Boolean equation derived from the logical relationship among the 2-bit inputs of Rainfall (r_1r_0), Wind (w_1w_0), Seismic Activity (s_1s_0), and Sea Level (l_1l_0). (where "·" denotes logical AND and "+" denotes logical OR.)

$$\text{Flood (F)} = r_1 \cdot (w_1 + l_1 + r_0)$$
$$\text{Cyclone (C)} = w_1 \cdot (w_0 + l_1 + r_1)$$
$$\text{Earthquake (E)} = s_1 + s_0$$
$$\text{Tsunami (T)} = (s_1 \cdot s_0) + l_1$$

2. **Priority Encoding Formula** The detected disaster outputs (F, C, E, T) are encoded into a 2-bit binary code (y_1y_0) based on disaster severity:

$$(y_1y_0) = \begin{cases} 11, & \text{if } T = 1 \\ 10, & \text{if } E = 1 \text{ and } T = 0 \\ 01, & \text{if } C = 1 \text{ and } E = T = 0 \\ 00, & \text{if } F = 1 \text{ and } C = E = T = 0 \end{cases}$$

3. **Mode Selection Logic** The output behavior is determined by the Mode input (M):

$$\text{If } M = 0 \Rightarrow \text{Unique Mode (Only highest-priority disaster LED ON)}$$

$$\text{If } M = 1 \Rightarrow \text{Multi Mode (All detected disaster LEDs ON)}$$

4. **Final Output Function** The overall system output is a function of the detected disasters

and the mode:

$$O = \begin{cases} f_{\text{unique}}(F, C, E, T), & \text{if } M = 0 \\ f_{\text{multi}}(F, C, E, T), & \text{if } M = 1 \end{cases}$$

where:

$$f_{\text{unique}} = \text{Highest-priority disaster only}, \quad f_{\text{multi}} = \text{All active disasters simultaneously.}$$

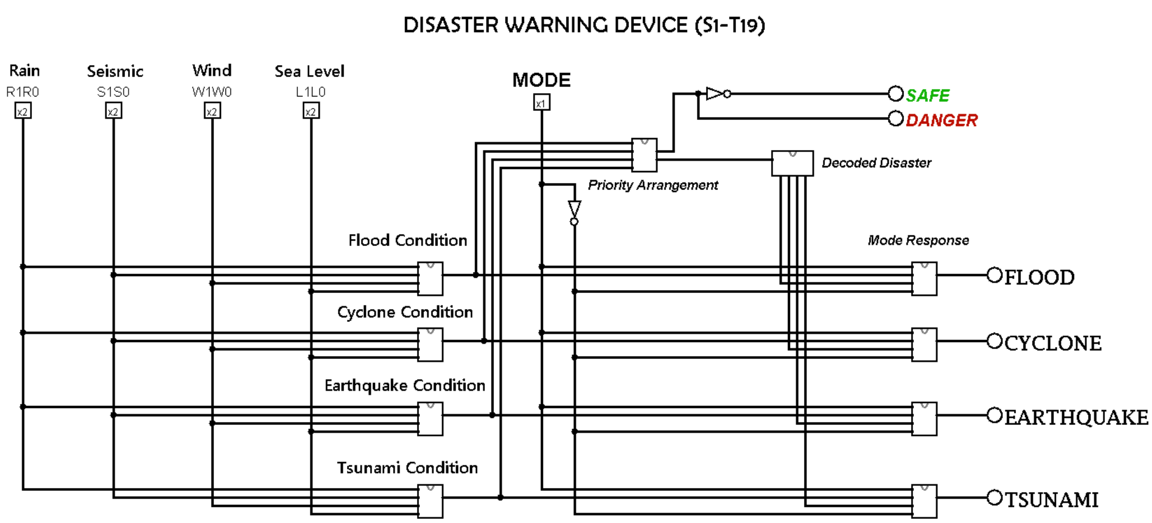
5. **Output Mapping Formula** The LED indicators are activated according to:

$$[Y_3, Y_2, Y_1, Y_0] = \begin{cases} [T, E, C, F], & \text{if } M = 1 \\ \text{Decoder}(y_1 y_0), & \text{if } M = 0 \end{cases}$$

where Y_3, Y_2, Y_1, and Y_0 correspond to Tsunami, Earthquake, Cyclone, and Flood LEDs respectively.

4 Logisim Modules

4.1 Main Module – DEVICE



MODE	TYPE
0	UNIQUE DISASTER
1	MULTI DISASTER

INPUT	IMPACT
00	Low/None
01	Medium
10	High
11	Very High

Figure 3: Main Logisim Module – Disaster Warning Device

The module accepts four 2-bit environmental inputs — Rainfall (`R1R0`), Wind Speed (`W1W0`), Seismic Activity (`S1S0`), and Sea Level (`L1L0`) — representing intensity levels: `00 = Low`, `01 = Medium`, `10 = High`, and `11 = Very High`. Each input pair is fed to its respective **Condition Comparator Module**, which evaluates the specific logical expressions for different disasters:

The binary outputs from these comparators (F, C, E, and T) are passed into the **Priority Arrangement Module**. This block ensures that when multiple disasters are active simultaneously, only the highest-priority disaster is encoded for display, based on the sequence:

$$\text{Tsunami (11)} > \text{Earthquake (10)} > \text{Cyclone (01)} > \text{Flood (00)}$$

The resulting 2-bit encoded signal is then processed by a **Decoder Module**, which activates the corresponding LED to indicate the detected disaster. A **Mode Selector** input allows two distinct operating configurations:

- **Mode = 0 (Unique Disaster Mode):** Only the LED of the highest-priority detected disaster glows.
- **Mode = 1 (Multi-Disaster Mode):** All LEDs of active disasters glow simultaneously.

The outputs are visually represented using LEDs labeled as `FLOOD`, `CYCLONE`, `EARTHQUAKE`, and `TSUNAMI`. An additional set of LEDs labeled `SAFE` (Green) and `DANGER` (Red) display the overall system condition. The **Index Table** at the bottom of the circuit defines the input coding and operational modes for quick reference:

$$00 = \text{Low/None}, \quad 01 = \text{Medium}, \quad 10 = \text{High}, \quad 11 = \text{Very High}.$$

4.1.1 Flood Condition Module

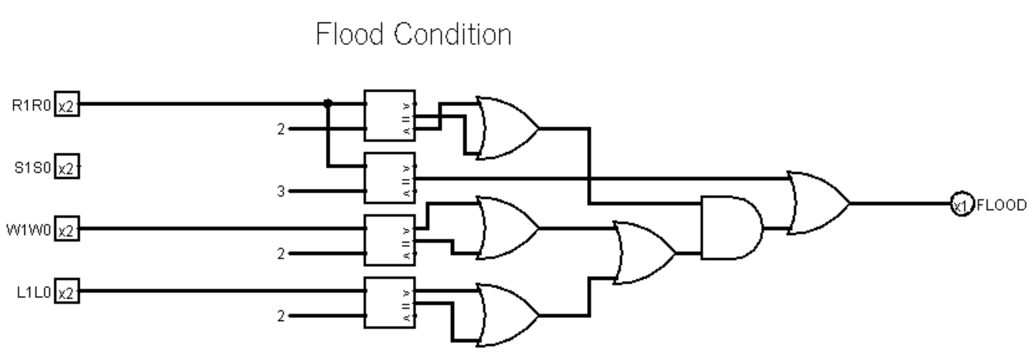


Figure 4: Flood Condition Submodule in Logisim

The Flood Condition module detects flooding when there is high rainfall combined with either strong wind, elevated sea level, or continuous rain. It implements the Boolean expression:

$$F = r_1 \cdot (w_1 + l_1 + r_0)$$

Inputs: r_1r_0, w_1w_0, l_1l_0 **Output:** F (Flood indicator)

In Logisim, the circuit uses:

- One 2-input AND gate for r_1.
- Three OR gates to combine $(w_1 + l_1 + r_0)$.
- One final AND gate to compute F.

When the output $F = 1$, the flood condition is considered active.

4.1.2 Cyclone Condition Module

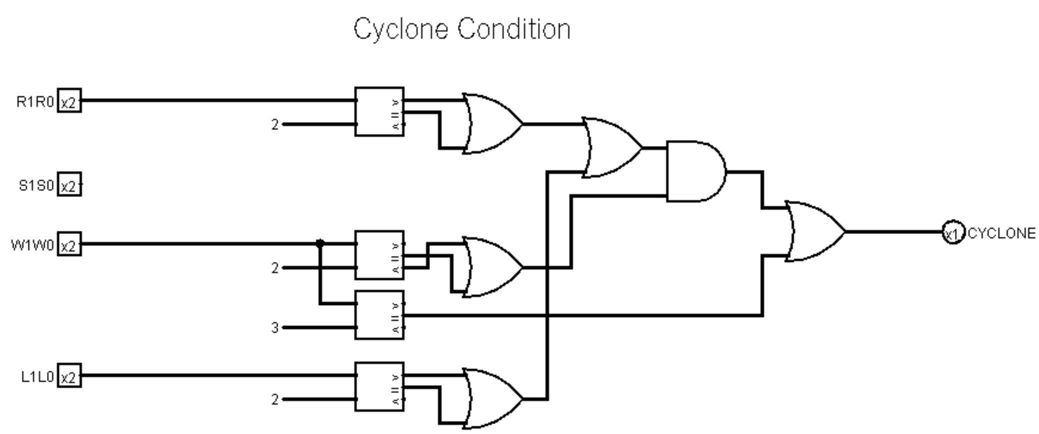


Figure 5: Cyclone Condition Submodule in Logisim

The Cyclone Condition module detects high wind activity accompanied by either sea-level rise or heavy rainfall. It is defined by the Boolean expression:

$$C = w_1 \cdot (w_0 + l_1 + r_1)$$

Inputs: w_1w_0, l_1l_0, r_1r_0 **Output:** C (Cyclone indicator)

The circuit is implemented using:

- Two 2-input OR gates for intermediate combinations.
- One 3-input OR gate to aggregate $(w_0 + l_1 + r_1)$.

- One final AND gate with w_1 to generate C.

When $C = 1$, it indicates that conditions suitable for a cyclone are met.

4.1.3 Earthquake Condition Module

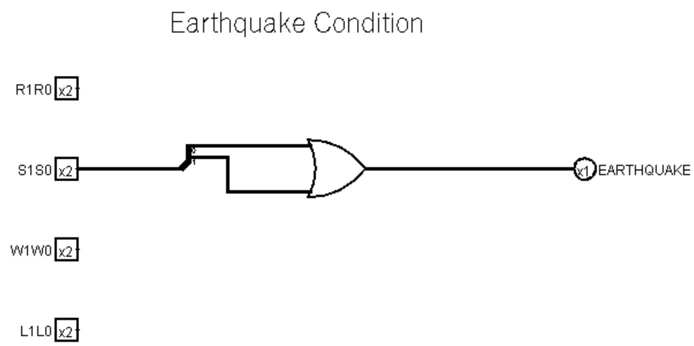


Figure 6: Earthquake Condition Submodule in Logisim

The Earthquake Condition module is the simplest among all. It checks for any seismic activity where either s_1 or s_0 is high. Its Boolean expression is:

$$E = s_1 + s_0$$

Inputs: $s_1 s_0$ **Output:** E (Earthquake indicator)

In Logisim, this is implemented using a single 2-input OR gate. When either input is high, the output $E = 1$, indicating earthquake detection.

4.1.4 Tsunami Condition Module

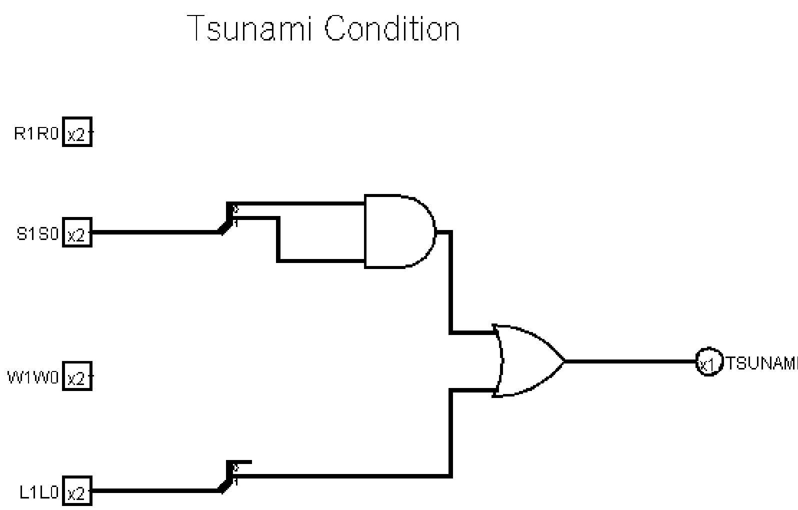


Figure 7: Tsunami Condition Submodule in Logisim

The Tsunami Condition module detects high seismic disturbance coupled with a high sea-level rise. It uses both AND and OR logic as defined by:

$$T = (s_1 \cdot s_0) + l_1$$

Inputs: $s_1 s_0$, $l_1 l_0$ **Output:** T (Tsunami indicator)

Implementation in Logisim:

- One 2-input AND gate for $(s_1 \cdot s_0)$.
- One 2-input OR gate to combine $(s_1 \cdot s_0)$ with l_1.

The output $T = 1$ indicates that both seismic and sea-level parameters are critically high, triggering a tsunami warning.

4.1.5 Priority Encoder Module

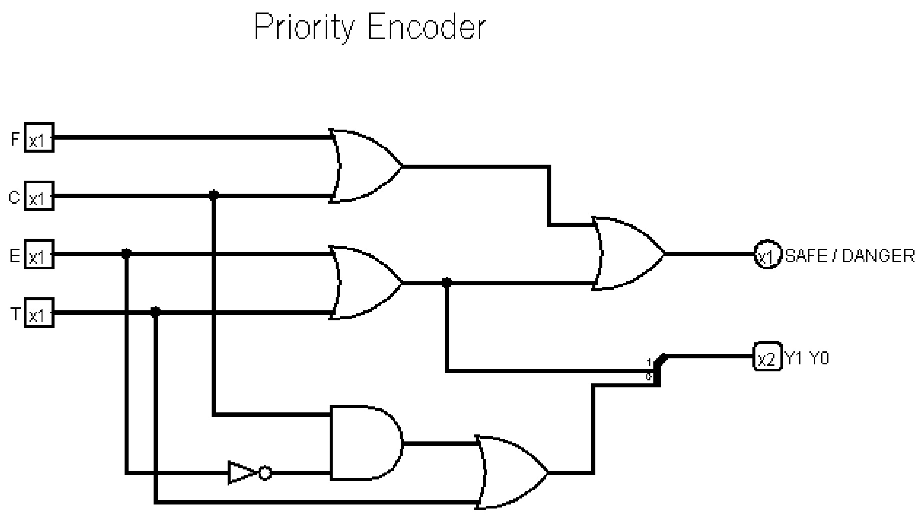


Figure 8: Priority Encoder Submodule in Logisim

The **Priority Encoder Module** determines which disaster condition has the highest precedence among the four detected signals — Flood (F), Cyclone (C), Earthquake (E), and Tsunami (T). It converts the active disaster signals into a 2-bit binary code (y_1y_0) and simultaneously provides a `Safe/Danger` output for overall system status.

The priority order is defined as:

$$\text{Tsunami (11)} > \text{Earthquake (10)} > \text{Cyclone (01)} > \text{Flood (00)}$$

The Boolean expressions for the encoded outputs are:

$$y_1 = E + T$$
$$y_0 = C + T$$
$$\text{Safe/Danger} = F + C + E + T$$

If no disaster input is active ($F = C = E = T = 0$), the `Safe/Danger` output remains at logic LOW, indicating a safe condition.

Implementation:

- Inputs: F, C, E, T (1-bit each)
- Outputs: y_1, y_0 (2-bit encoded output), Safe/Danger (1-bit)
- Components used: OR gates, AND gate, and one NOT gate for priority masking.

4.1.6 Decoder Module

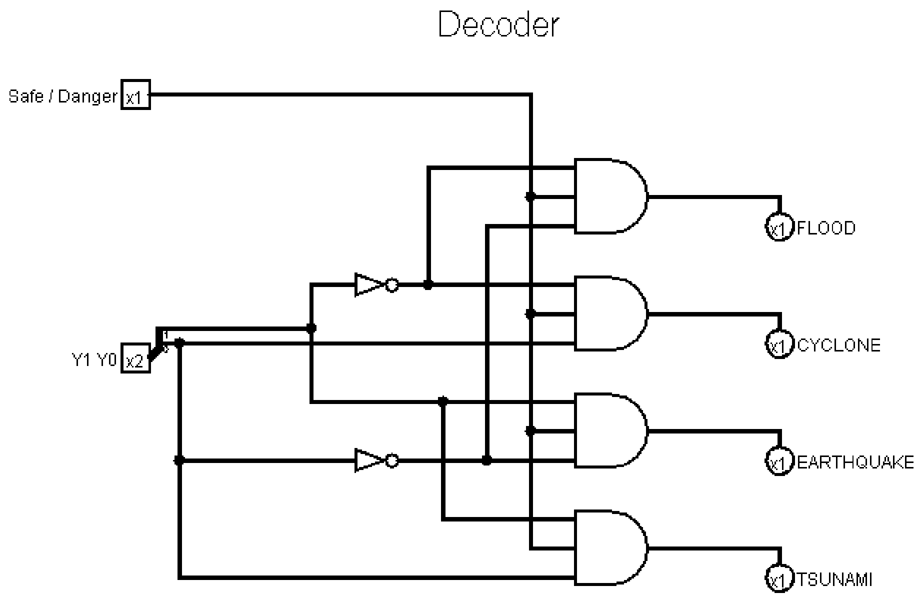


Figure 9: Decoder Submodule in Logisim

The **Decoder Module** takes the 2-bit encoded output (y_1y_0) from the priority encoder and converts it back into four one-hot outputs to activate the respective disaster LEDs. It operates only when the system is in a danger state.

The logical output mapping is:

y_1	y_0	FLOOD	CYCLONE	EARTHQUAKE	TSUNAMI
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Each output line is enabled through an AND gate, combining the `Safe/Danger` signal with decoded inputs (y_1, y_0).

Implementation:

- Inputs: y_1, y_0 (from encoder), Safe/Danger (control)
- Outputs: Four disaster indicators (F, C, E, T)
- Components used: AND gates, NOT gates for inversion, and direct LED outputs.

4.1.7 Mode Response Module

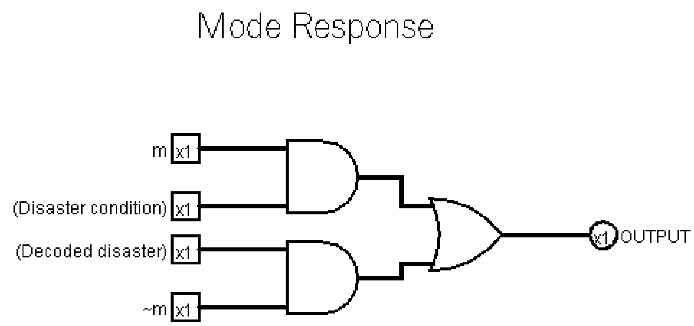


Figure 10: Mode Response Submodule in Logisim

The **Mode Response Module** determines how the disaster outputs are displayed based on the operating mode. It takes three inputs: the `Mode` control signal (m), the `Decoded Disaster Output`, and the `Raw Disaster Condition` signal. Its output is the final signal sent to each LED.
The Boolean expression governing the mode response is:

$$\text{Output} = (\overline{m} \cdot \text{Decoded Output}) + (m \cdot \text{Disaster Condition})$$

- For $m = 0$ (Unique Mode): only the highest-priority decoded output activates.
- For $m = 1$ (Multi Mode): all active disaster condition outputs are displayed simultaneously.

Implementation:

- Inputs: m, Decoded Output, Disaster Condition
- Output: Final LED output (1-bit per disaster)
- Components used: AND gates, OR gate, and NOT gate.

In essence, the Mode Response module merges the outputs from both the encoder-decoder and direct condition paths, allowing flexible visualization of single or multiple disasters.

5 Verilog

This section contains the Verilog code for the system, presented in four modeling styles.

5.1 Gate-Level Modeling

```verilog

module disaster_gate (
    input  r1, r0, s1, s0, w1, w0, l1, l0, mode,
    output flood_led, cyclone_led, earthquake_led, tsunami_led
);

    // --- Intermediate disaster detection signals ---
    wire flood, cyclone, earthquake, tsunami;

    // Earthquake detection: if any seismic sensor active
    or  or_e1 (earthquake, s1, s0);

    // Tsunami detection: requires both seismic signals + land level
    and and_ts1 (ts_and, s1, s0);
    or  or_ts1 (tsunami, ts_and, l1);

    // Flood detection: depends on rain and water-level combination
    or  or_f2 (flood_or_branch, w1, l1, r0);
    and and_f1 (flood, r1, flood_or_branch);

    // Cyclone detection: depends on wind and rain interactions
    or  or_c1 (cyclone_or_branch, w0, l1, r1);
    and and_c1 (cyclone, w1, cyclone_or_branch);

    // --- Generate disaster code bits (2-bit encoding) ---
    wire n_earthquake;
    not not_e (n_earthquake, earthquake);

    wire cyclone_and_not_e;
    and and_c_nE (cyclone_and_not_e, cyclone, n_earthquake);

    wire code1, code0;
    or  or_code1 (code1, tsunami, earthquake);
    or  or_code0 (code0, tsunami, cyclone_and_not_e);
```

```

    // --- Decoder section (4 outputs for 4 disasters) ---
    wire ncode1, ncode0;
    not not_c1 (ncode1, code1);
    not not_c0 (ncode0, code0);

    wire Df, Dc, De, Dt;
    and and_Df (Df, ncode1, ncode0);   // Flood
    and and_Dc (Dc, ncode1, code0);    // Cyclone
    and and_De (De, code1, ncode0);    // Earthquake
    and and_Dt (Dt, code1, code0);     // Tsunami

    // --- Mode control: select between decoder and direct signals ---
    wire nm;
    not notm (nm, mode);

    wire uf, uc, ue, ut; // decoder mode outputs
    and (uf, nm, Df);
    and (uc, nm, Dc);
    and (ue, nm, De);
    and (ut, nm, Dt);

    wire mf, mc, me, mt; // direct mode outputs

endmodule
```

Listing 1: Gate-Level Disaster Detection System

5.2 Dataflow Modeling

```

module disaster_dataflow(
    input  r1, r0, s1, s0, w1, w0, l1, l0, mode,
    output flood_led, cyclone_led, earthquake_led, tsunami_led
);
    // Disaster condition logic using Boolean equations
    wire flood      = r1 & (w1 | l1 | r0);
    wire cyclone    = w1 & (w0 | l1 | r1);
    wire earthquake = s1 | s0;
    wire tsunami    = s1 & l1;
```

```

    // Priority encoding of disasters
    wire [1:0] code = flood      ? 2'b00 :
                      cyclone    ? 2'b01 :
                      earthquake ? 2'b10 :
                      tsunami    ? 2'b11 :
                      2'b00;

    // Decoder logic
    wire Df = (code == 2'b00);
    wire Dc = (code == 2'b01);
    wire De = (code == 2'b10);
    wire Dt = (code == 2'b11);

    // Mode selection (0 = priority output, 1 = direct disaster LEDs)
    assign flood_led      = (~mode & Df) | (mode & flood);
    assign cyclone_led    = (~mode & Dc) | (mode & cyclone);
    assign earthquake_led = (~mode & De) | (mode & earthquake);
    assign tsunami_led    = (~mode & Dt) | (mode & tsunami);
endmodule
```

Listing 2: Dataflow Modeling of Disaster Detection System

5.3 Behavioral Modeling

```

module disaster_behavioral(
    input  r1, r0, s1, s0, w1, w0, l1, l0, mode,
    output reg flood_led, cyclone_led, earthquake_led, tsunami_led
);
    reg flood, cyclone, earthquake, tsunami;
    reg [1:0] code;
    reg Df, Dc, De, Dt;

    // Behavioral block describing all logic operations
    always @(*) begin
        // --- Disaster condition logic ---
        flood      = r1 & (w1 | l1 | r0);
        cyclone    = w1 & (w0 | l1 | r1);
        earthquake = s1 | s0;
```

```
        tsunami    = (s1 & s0) | l1;

        // --- Priority encoder for disaster severity ---
        if      (tsunami)    code = 2'b11;
        else if (earthquake) code = 2'b10;
        else if (cyclone)    code = 2'b01;
        else if (flood)      code = 2'b00;
        else                 code = 2'b00;

        // --- Decoder logic for 4 disaster indicators ---
        case (code)
            2'b00: {Df, Dc, De, Dt} = 4'b1000; // Flood
            2'b01: {Df, Dc, De, Dt} = 4'b0100; // Cyclone
            2'b10: {Df, Dc, De, Dt} = 4'b0010; // Earthquake
            2'b11: {Df, Dc, De, Dt} = 4'b0001; // Tsunami
            default: {Df, Dc, De, Dt} = 4'b0000;
        endcase

        // --- Mode selection ---
        flood_led      = (~mode & Df) | (mode & flood);
        cyclone_led    = (~mode & Dc) | (mode & cyclone);
        earthquake_led = (~mode & De) | (mode & earthquake);
        tsunami_led    = (~mode & Dt) | (mode & tsunami);
    end
endmodule
```

Listing 3: Behavioral Modeling of Disaster Detection System

5.4 Testbench Modeling

```
`timescale 1ns/1ps
module tb_disaster_all;


    // Input and output declarations

    reg  r1, r0, s1, s0, w1, w0, l1, l0, mode;
    wire flood_led_g, cyclone_led_g, earthquake_led_g, tsunami_led_g;
    wire flood_led_d, cyclone_led_d, earthquake_led_d, tsunami_led_d;
    wire flood_led_b, cyclone_led_b, earthquake_led_b, tsunami_led_b;
```

```
    // Instantiate all three models

    disaster_gate       U_GATE (.r1(r1), .r0(r0), .s1(s1), .s0(s0), .w1(w1),
        .w0(w0),
                               .l1(l1), .l0(l0), .mode(mode),
                               .flood_led(flood_led_g), .cyclone_led(
                                   cyclone_led_g),
                               .earthquake_led(earthquake_led_g), .
                                   tsunami_led(tsunami_led_g));

    disaster_dataflow   U_DATA (.r1(r1), .r0(r0), .s1(s1), .s0(s0), .w1(w1),
        .w0(w0),
                               .l1(l1), .l0(l0), .mode(mode),
                               .flood_led(flood_led_d), .cyclone_led(
                                   cyclone_led_d),
                               .earthquake_led(earthquake_led_d), .
                                   tsunami_led(tsunami_led_d));

    disaster_behavioral U_BEH  (.r1(r1), .r0(r0), .s1(s1), .s0(s0), .w1(w1),
        .w0(w0),
                               .l1(l1), .l0(l0), .mode(mode),
                               .flood_led(flood_led_b), .cyclone_led(
                                   cyclone_led_b),
                               .earthquake_led(earthquake_led_b), .
                                   tsunami_led(tsunami_led_b));


    // Dump waveform data for simulation

    initial begin
        $dumpfile("disaster.vcd");
        $dumpvars(0, tb_disaster_all);
    end

    // Test variables

    integer i, m, sno, curr_active, max_active;
```

```
    reg [7:0] max_active_vector;
    reg [8*256:1] outstr;

    // Main stimulus and display logic

    initial begin
        sno = 0;
        max_active = 0;
        max_active_vector = 8'hFF;

        $display("Sno | Mode | R1R0 | S1S0 | W1W0 | L1L0 | Output (Flood, Cyclone, Earthquake, Tsunami)");
        $display("----+------+-------+-------+-------+-------+-----------------");

        // Test all input combinations for both modes
        for (m = 0; m <= 1; m = m + 1) begin
            mode = m;

            for (i = 0; i < 256; i = i + 1) begin
                {r1, r0, s1, s0, w1, w0, l1, l0} = i[7:0];
                #1;
                sno = sno + 1;
                outstr = "";

                // --- Mode 0: Single disaster output ---
                if (mode == 0) begin
                    if      (flood_led_g)      outstr = "flood";
                    else if (cyclone_led_g)    outstr = "cyclone";
                    else if (earthquake_led_g) outstr = "earthquake";
                    else if (tsunami_led_g)    outstr = "tsunami";
                    else                       outstr = "none";
                end
                // --- Mode 1: Multiple simultaneous outputs ---
                else begin
                    if (flood_led_g)      $sformat(outstr, "%s%s", outstr, (outstr=="" ? "flood" : ", flood"));
                    if (cyclone_led_g)    $sformat(outstr, "%s%s", outstr, (outstr=="" ? "cyclone" : ", cyclone"));
```

```
                    if (earthquake_led_g) $sformat(outstr, "%s%s", outstr, (outstr=="" ? "earthquake" : ", earthquake"));
                    if (tsunami_led_g)    $sformat(outstr, "%s%s", outstr, (outstr=="" ? "tsunami" : ", tsunami"));
                    if (outstr == "") outstr = "none";
                end

                // Count active disaster LEDs
                curr_active = flood_led_g + cyclone_led_g + earthquake_led_g + tsunami_led_g;
                if (curr_active > max_active) begin
                    max_active = curr_active;
                    max_active_vector = i[7:0];
                end
                $display("%4d |  %b    |  %b%b   |  %b%b   |  %b%b   |  %b%b   | %-45s",
                         sno, mode, r1, r0, s1, s0, w1, w0, l1, l0, outstr);
            end
        end

        #5;
        $finish;
    end
endmodule
```

Listing 4: Testbench for All Verilog Models

References

1. M. Morris Mano. *Digital Design*, Prentice Hall, 1995.
   https://digitalsystemdesign.in/wp-content/uploads/2018/05/digital_design-__morris_mano-fifth_edition.pdf

2. Mitheu, F. K. *A Model for Impact-Based Flood Early Warning and Anticipatory Actions in Uganda*. University of Reading, 2023.
   https://centaur.reading.ac.uk/112918/1/MITHEU_Thesis.pdf

3. U.S. Fire Administration (FEMA). *Fire Service and Disaster Response Integration: Emergency Preparedness Framework*, 2024.
   https://apps.usfa.fema.gov/pdf/efop/efo34486.pdf

4. United States Patent. *US4153881A – Early Warning and Control System for Flood, Earthquake, and Other Natural Disasters*, 2024.
   https://patents.google.com/patent/US4153881A/en

5. Electronics Tutorials. *Magnitude Comparator in Digital Logic.*
   https://www.electronics-tutorials.ws/combination/comb_8.html

6. GeeksforGeeks. *Encoders and Decoders in Digital Logic.*
   https://www.geeksforgeeks.org/digital-logic/encoders-and-decoders-in-digital-logic/