Design Priority Encoder(4*2) In Proteus And Coded In Arduino Andwrite Down The Properties, And Function

Design Priority Encoder(42) In Proteus And Coded In Arduino: Properties And Functions

The evolution of digital systems has led to the development of various encoding mechanisms that efficiently manage multiple input signals. Among these, the Priority Encoder stands out as a vital component in digital electronics, enabling systems to identify the highest priority input among several active signals. This article provides a comprehensive overview of designing a 4-to-2 Priority Encoder using Proteus simulation software and implementing its logic in Arduino. We will delve into its properties, working principles, practical implementation, and advantages.

Understanding Priority Encoders

What Is a Priority Encoder?

A Priority Encoder is a digital circuit that converts multiple input signals into a binary code based on the highest priority input. The key feature of a Priority Encoder is that it not only encodes active inputs but also considers their priority order, ensuring that the highest-priority input is always represented in the output.

Applications of Priority Encoders

- Interrupt handling in microprocessors
- Data compression algorithms
- Multiplexer systems
- Priority-based decision-making in control systems
- Digital systems requiring prioritized input processing

Designing a 42 Priority Encoder

Basic Concept

A 42 Priority Encoder takes four input signals (I0, I1, I2, I3) and encodes them into a 2-bit binary output (Y1, Y0). The encoder also provides a 'valid' output indicating whether any input is active.

Truth Table

I3	I2	I1	I0	Y1	Y0	Valid
1	x	x	x	1	1	1
0	1	x	x	1	0	1
0	0	1	x	0	1	1
0	0	0	1	0	0	1
0	0	0	0	0	0	0

Note: 'x' denotes 'don't care' condition.

Logic Implementation

The priority is assigned such that I3 has the highest priority, followed by I2, I1, and I0. The output is determined based on the highest active input.

- When I3 = 1, regardless of others, output Y1Y0 = 11.
- When I3 = 0 and I2 = 1, output Y1Y0 = 10.
- When I3 = 0, I2 = 0, and I1 = 1, output Y1Y0 = 01.
- When only I0 = 1, output Y1Y0 = 00.
- When no inputs are active, output is invalid, and 'Valid' is 0.

Implementing the Priority Encoder in Proteus

Components Required

- Proteus Design Suite Software
- Logic gates (AND, OR, NOT)
- LED indicators for output display
- Push buttons or switches for inputs

Step-by-Step Procedure

1. Open Proteus and create a new project.
2. Place switches to represent inputs I0, I1, I2, I3, and connect them to logic gates following the truth table logic.
3. Use AND, OR, and NOT gates to implement the encoding logic based on the priority conditions.
4. Connect the output of logic circuits to LEDs to visualize the binary output Y1 and Y0.
5. Integrate a 'Valid' indicator LED to show whether any input is active.
6. Simulate the circuit, toggle switches, and observe the outputs.

Programming the Priority Encoder in Arduino

Why Use Arduino?

Arduino provides a flexible platform to implement and test the logic of a Priority Encoder through programming. It allows for dynamic input handling and easy modifications, making it ideal for prototyping.

Required Components

- Arduino Uno or compatible microcontroller
- Push buttons or switches for inputs
- LEDs for output display

- Resistors (220Ω) for LEDs
- Connecting wires

Sample Arduino Code

```
const int inputPins[4] = {2, 3, 4, 5}; // I0, I1, I2, I3
const int outputPins[2] = {8, 9}; // Y0, Y1
const int validPin = 10; // Valid output indicator

void setup() {
// Initialize input pins
for (int i = 0; i < 4; i++) {
pinMode(inputPins[i], INPUT_PULLUP);
}
// Initialize output pins
for (int i = 0; i < 2; i++) {
pinMode(outputPins[i], OUTPUT);
}
pinMode(validPin, OUTPUT);
Serial.begin(9600);
}

void loop() {
// Read inputs (active low configuration)
int inputs[4];
for (int i = 0; i < 4; i++) {
inputs[i] = digitalRead(inputPins[i]) == LOW ? 1 : 0;
}

int Y1 = 0, Y0 = 0;
int valid = 0;

// Priority check
if (inputs[3] == 1) {
// I3 high
Y1 = 1;
Y0 = 1;
valid = 1;
} else if (inputs[2] == 1) {
// I2 high
Y1 = 1;
Y0 = 0;
valid = 1;
} else if (inputs[1] == 1) {
```

```
// I1 high
Y1 = 0;
Y0 = 1;
valid = 1;
} else if (inputs[0] == 1) {
// I0 high
Y1 = 0;
Y0 = 0;
valid = 1;
} else {
// No inputs active
valid = 0;
}

// Set outputs
digitalWrite(outputPins[0], Y0);
digitalWrite(outputPins[1], Y1);
digitalWrite(validPin, valid);

// Optional: print status
Serial.print("Y1: "); Serial.print(Y1);
Serial.print(" Y0: "); Serial.print(Y0);
Serial.print(" Valid: "); Serial.println(valid);

delay(500);
}
```

Properties of 42 Priority Encoder

- **Encoding Capacity:** Converts 4 input signals into a 2-bit binary code.
- **Priority Handling:** Always encodes the highest priority active input.
- **Output Signals:** Produces binary outputs (Y1, Y0) representing the highest priority input.
- **Valid Signal:** Indicates whether any input is active or not.
- **Deterministic Output:** Ensures the output reflects the highest priority input regardless of other active inputs.
- **Scalability:** Can be expanded or combined with other encoders for larger systems.