**Module: SV for Verification**

**Section:** Typedef/Enum **Task:** Mailboxes

**Task -** [**Typedef/Enum**](https://www.edaplayground.com/x/ZgfY)

Typedef-Enum

* **Code:**

// Author: Noman Rafiq

// Dated: Sep 10, 2024

typedef bit[15:0] data\_bus\_t; // custom 16-bit data-type of 'data\_bus\_t' type

typedef enum bit[2:0] {BEQ, BNE, NO\_CONDITION, BLT, BGE=3'b101, BLTU, BGEU} br\_cond\_e; // custom enumerated variable for conditional branch types

module tb;

data\_bus\_t data; // 16-bit variable data of 'data\_bus\_t' type

br\_cond\_e branch; // branch variable of enum 'br\_cond\_e' type

initial begin

branch = 0;

$display("\nRunning...\n");

for (int i = 0; i < 8; i++) begin

case(branch)

3'b000 : $display("@i = %0d :: Branch = %s", i, branch.name());

3'b001 : $display("@i = %0d :: Branch = %s", i, branch.name());

3'b010 : $display("@i = %0d :: Branch = %s", i, branch.name());

3'b011 : $display("@i = %0d :: Branch = %s", i, branch.name());

3'b101 : $display("@i = %0d :: Branch = %s", i, branch.name());

3'b110 : $display("@i = %0d :: Branch = %s", i, branch.name());

3'b111 : $display("@i = %0d :: Branch = %s", i, branch.name());

default: $display("@i = %0d :: Branch doesn't exist!", i);

endcase

branch=branch + 1;

end

end

endmodule

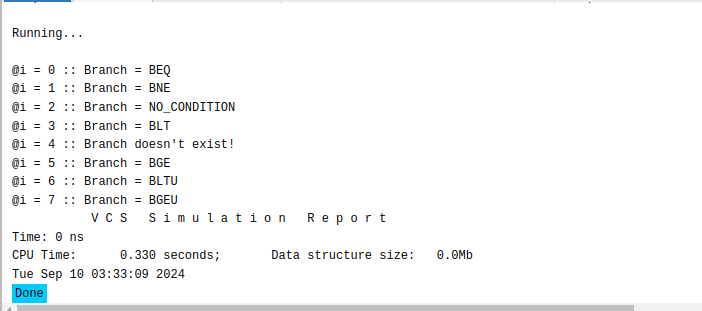
* **Difference b/w bit[15:0] and data\_bus\_t:**

The difference between **bit [15:0] data** and **data\_bus\_t data** is largely syntactical.

1. **bit [15:0] data:**
   * This explicitly declares a 16-bit bit vector.
2. **data\_bus\_t data:**
   * This uses a **typedef** alias for **bit [15:0]**, which is essentially the same type.
   * **data\_bus\_t** is a more abstract way to represent the data. Using the typedef improves code readability and maintainability, especially if the bit-width or type of the data bus needs to change later.

In practice, both declarations refer to a 16-bit **bit** vector, but **data\_bus\_t** provides more flexibility by allowing you to change the underlying type in one place if necessary.

**Output:**

****