**Module: SV for Verification**

**Section:** Threads & Interprocess Communication **Task:** Mailboxes

**Task 3 -** [**Bounded-MailBox**](https://www.edaplayground.com/x/9P26)

Bounded Mailbox

* **Code:**

// Author: Noman Rafiq

// Dated: Sep 09, 2024

program automatic test;

mailbox #(int) mbx;

int value;

initial begin

mbx = new(1);

$display("mbx.num()=%0d", mbx.num());

$display("mbx.try\_get= %0d", mbx.try\_get(value));

mbx.put(2);

$display("mbx.try\_put= %0d", mbx.try\_put(value));

$display("mbx.num()=%0d", mbx.num());

mbx.peek(value);

$display("value=%0d", value);

end

endprogram

* **Expected Output:**

The output reflects the state of the mailbox at each step. Initially, the mailbox is created with a size of 1 and is empty, so **mbx.num()** returns 0. When trying to retrieve a value from the empty mailbox using **mbx.try\_get()**, it fails, returning 0. After placing the value 2 into the mailbox with **mbx.put(2)**, the mailbox is now full.

The subsequent attempts to put another value using **mbx.try\_put()** fails because the mailbox has reached its capacity, so it returns 0. After checking the number of elements in the mailbox with **mbx.num()**, it shows 1 because the value 2 is still inside. Finally, the **mbx.peek()** function confirms that the value stored in the mailbox is indeed **2**, which it displays without removing it.

mbx.num()=0

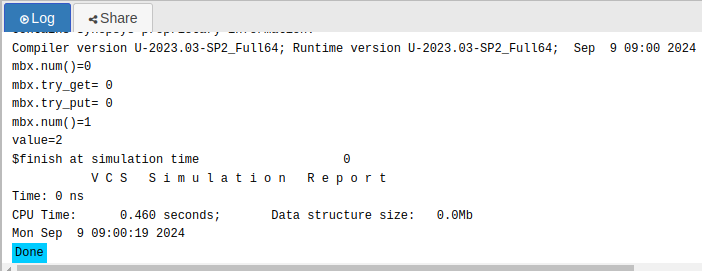
mbx.try\_get= 0

mbx.try\_put= 0

mbx.num()=1

value=2

**Actual Output:**

****