1_5.

Since the state transition of the machine becomes complicated when it's in `SERVICE_BUSY, two assertions are mainly focus on catch the potential bugs here.

First monitor(p2):

```
assign p2 = initialized && (serviceTypeOut == `SERVICE_OFF) && (inputValue != changeAndItem);
```

where changeAndItem is the sum of the change and the value of the item.

This asserted that when the output change calculation is finished, the machine should
return correct change so that the value between user's input and the item and change
are the same, if the relation is incorrect, that means whether the machine "eats money" or makes money out of nothing.

Second monitor(p3):

```
assign p3 = initialized && ((countNTD_50 + coinOutNTD_50 > 3'd7)
                          ||(countNTD_10 + coinOutNTD_10 > 3'd7)
                          ||(countNTD_5 + coinOutNTD_5 > 3'd7)
                          ||(countNTD_1 + coinOutNTD_1 > 3'd7));
```

This asserted that the numbers of a coin in the machine and change should be less than 7. Since change is given by the coins stored in the machine, the sum of the two should be less than the maximum capacity of the machine, which is 7. If the assertion
fails, that means there are coins that comes from nothing.

The input pattern in verify directory contains 177 random requests during 2000 cycles, including requests that does and doesn't give enough money. It turns out that the two monitors is triggered in these requests, which means no bug is found by the verification.