Vending Machine - Hot Dog

ESDM Project

Figure 1: Hot Dog Vending Machine

General Description

1. Create and test a Simulink model containing a state machine implementing the logic module of a vending machine.
2. Write a report on the project, containing:
    a. An overall description of the design (how it works, states, transitions etc,).

    b. Some tests of the functionality (2-3 tests, depending on complexity, covering normal usage and some error scenario)

    For each test, indicate:

    - The test scenario: what are the inputs, what are the desired outputs
    - The test results: include screenshots from the tests, to prove the tests work

Requirements

1. The vending machine has 3 products categories available: Hot Dog, Double Dog, Veggie Dog.
2. Each product can have mustard and/or ketchup (fixed amount).
3. List of inputs and outputs of the model:

   Inputs:

   - ProductSelection: integer (0 to 3)
     - 0, no product is selected
     - 1: Hot Dog = Bun + Sausage
     - 2: Double Dog = Bun + Two Sausages
     - 3: Veggie Dog = Bun + a Veggie Sausage
   - MustardSelection: boolean
     - TRUE = customer wants mustard
     - FALSE = no mustard
   - KetchupSelection: boolean
     - TRUE = customer wants ketchup
     - FALSE = no ketchup
   - MoneyInput: integer
     - when 0, no money is inserted
     - when non-zero, it is the current value of the money inserted by the customer
   - Cancel: boolean
     - when True, cancels an ongoing operation. All money input until this moment shall be returned to the customer.
   - ResetStock
     - when True, the stock for all buns, sausages and veggie sausages to 10 (e.g. the machine was refilled).

   Outputs:

   - DispenseBun: boolean
     - the transition from False to True activates the dispensing of one bread bun
   - DispenseSausage: boolean
     - the transition from False to True activates the dispensing of one sausage
   - DispenseVeggieSausage: boolean
     - the transition from False to True activates the dispensing of one veggie sausage

- MoneyReturn: integer, controls the money returned to the customer
  - when 0, nothing happens
  - when non-zero, the specified amount of money will be returned to the customer
- Status: integer, a status message indicating the current state
  - 0 = Idle, awaiting operation
  - 1 = Operation in progress
  - 2 = Success
  - 3 = Incorrect product code
  - 4 = Product out of stock

4. The vending machine operates in 4 basic steps:

   - first you enter the product code of the product
   - then you enter the money
   - then you indicate if you want mustard or not
   - then you indicate if you want ketchup or not
   - then the product is dispensed: bun + sausages (correct type and number)
   - then the rest of the money is returned

5. The vending machine starts with 10 buns, 10 sausage, 10 veggie sausages, unlimited mustard and ketchup

6. The price of every type of product is fixed and known (you pick some value, e.g. 6).

7. The vending machine keeps track of the number of products it has available at any time moment.

8. The machine shall detect if the user requests an invalid product code, and signal this at the Status output

9. The machine shall detect if the user requests a product which is currently out of stock, and signal at the Status output.

10. The machine shall calculate the rest of the money and provide back the change (Note: assume the machine has an infinite supply of coins/notes).

11. After dispensing a product, the machine will wait 5 seconds before accepting any new operation (to wait until the dispensing mechanism finishes).

12. The number of products available can be reset back to the value of 10 when the input `ResetStock` is activated.

13. The machine shall always provide a status code output.

14. The `MustardSelection` input button shall be debounced both ways, with a time duration of 0.25 seconds.

15. Use parameters from Matlab for all values you consider necessary (e.g. duration of delays, prices etc.). Our customer may want to adjust the parameters at any time.
16. Test your state machine (use one/multiple separate test models if necessary)