VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

EMBEDDED SYSTEMS

Homework Result

State Machine Model for Washing Machine

Advisor: Pham Hoang Anh
Students: Tran Long Vi - 1814804

HO CHI MINH CITY, APRIL 2021

1 STATE-MACHINE DIAGRAM

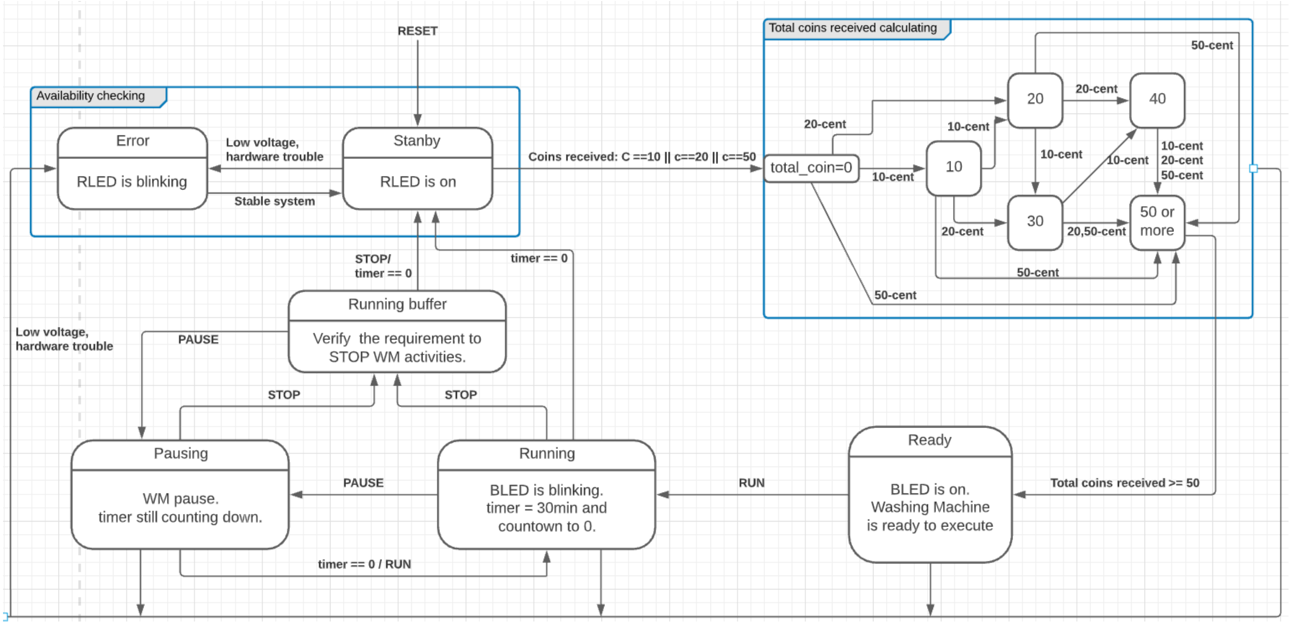

Figure 1: *State-Machine Diagram for Washing Machine*

2 DETAILED ANALYSIS OF THIS SYSTEM

- Availability checking block: this block is used to check the availability of the Washing Machine, by:
  - If this system is available to serve, the current statement is *Stanby*. System can't execute any processes (Except *Error* state) if it never cross *Stanby* state before.
  - If this system has problems that affects to activities of system (like low voltage,corrupted components,...), the current statement is *Error*. By all other state, the state will change to *Error* whenever the system is down or running incorrectly in comparison to the algorithm.
- Total coins received calculating block: This block is used to calculate and send feedback that system had received enough coin ($total\_coin \geq 50$).This block is created base on the rules:
  - This block only accepts 10-cent, 20 cent and 50-cent coins.
  - If customer use other denominations, system will still receive coins but $total\_coin$ doesn't change.
  - System has no service provided to return excess cash if $total\_coin \geq 50$.
- *Ready* state: this state is executed if and only if $total\_coin \geq 50$. In this state, system must wait for customer requirement by press *RUN* button.

- *Running* state: this state is activated when customer press button *RUN* in *Ready* state.
  - A value *timer* is initiated to countdown from min 30 to 0, this is the maximum time the system processes a washing process required by customer.
  - If $timer = 0$, system will go to *Stanby* state to wait next requirement.
  - if *PAUSE* button is pressed, system will go to *Pausing* state.
- *Pausing* state: this state is used to pause the system execution when *PAUSE* button is pressed.
  In this state, *timer* is still counting down. If $timer = 0$ or RUN button is pressed, system will go back to *Running* state.
- If for some reason, customer wanna stop the system execution, he/she has just pressed *STOP* button double time.
  - At the first press (System is in *Running* state or *Pausing* state), system will change to *Running Buffer* state.
  - *Running Buffer* state is the same function as *Running* state, but it provides the customer with the function of stopping the system from executing without waiting for $time = 0$.