|  |
| --- |
| Loughborough University |
| Washing Machine Control System |
| 13ELC018 Real-time software engineering |
|  |
| **Dave Read – A913927**  **James O’Neill – B013199**  **Krishen Chauhan – B216636**  **Matt Blickem – B022220** |
| **3/28/2014** |

|  |
| --- |
| UML software design for controlling the operations of a Bytronic washing machine simulator. |

Contents

[Specification 2](#_Toc383795040)

[Structural Diagrams 3](#_Toc383795041)

[Use Case Diagrams 3](#_Toc383795042)

[Class Diagram 4](#_Toc383795043)

[Behavioural Diagrams 5](#_Toc383795044)

[Activity Diagram 5](#_Toc383795045)

[Sequence Diagrams 6](#_Toc383795046)

[Ease of maintenance 11](#_Toc383795047)

# Specification

Task 3:

* All code except drivers should be designed.
* Design should be consistent with code.
* Standalone software, no interaction with host.
* Different drum speeds implemented with PWM.
* Allow for easy modification of wash programs.

From task 1:

* Pressing the accept button will start the execution of the selected program.
  + If the door is not closed (this will be indicated by a door indicator LED embedded in the door switch becoming illuminated) when the accept button is pressed the machine will not start and a brief warning will be sounded on the buzzer.
  + Pressing the accept button during execution will advance the program status by one stage.
  + If the door open/close switch is operated whilst a program is running, the buzzer will sound briefly and the program is suspended until the door is closed again
* As the program executes the 7 segment display shall indicate the stage that the program has reached.
* Pressing the cancel button will interrupt a running program causing it to pause.
  + Pressing the accept button whilst the program is paused will cause the program to resume.
  + Pressing the cancel button whilst the program is paused will cause the machine to reset.
* The system should use timers to define the duration of each stage.

# Structural Diagrams

### Use Case Diagrams

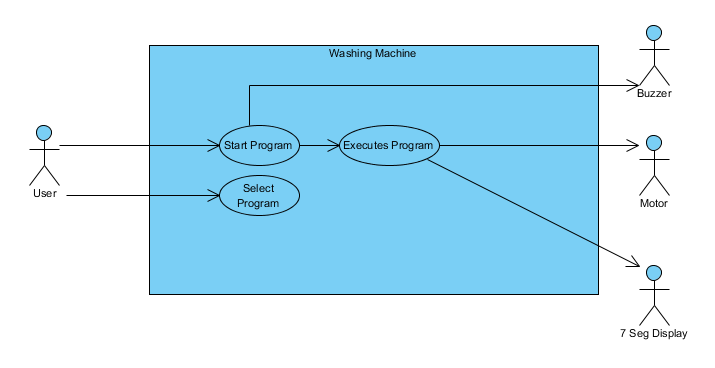


Figure 1 – Accept & Select Button Use Case

Figure 1 shows initial user interaction with the washing machine. The user can select a program and attempt to start a program. If the washing machine door is open, however, the buzzer will sound and the program will not start.

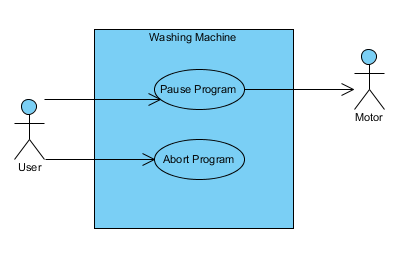


Figure 2 - Cancel Button Use Case

Figure 2 shows the user interactions with the cancel button. If there is a program running then the cancel button will pause the program, if the program is paused then subsequent presses of the cancel button will abort the program.

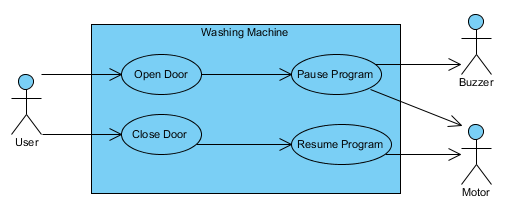


Figure 3 - Door Use Case

Figure 3 is only applicable when the program is running; it models the user interaction with door.

## Class Diagram

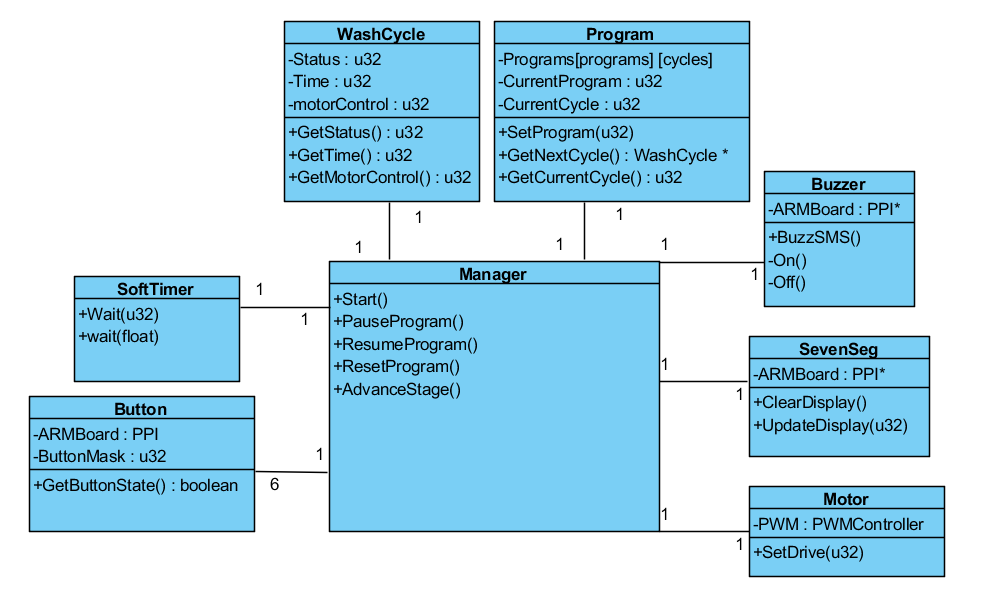


Figure 4 - Class Diagram

The 6 buttons in Figure 4 are sel1, sel2, sel3, Accept, Cancel and the Door. The Manager class controls the overall running of the software. Program class consists of an array of WashCycles which is a class for each wash cycle within a Program.

# Behavioural Diagrams

Figure 5 - Activity Diagram

## Activity Diagram

Figure 5 starts at power on of the device and as the system is an embedded real-time system there is no end node. The default (idle) activity is Check for Accept Button. All inputs (buttons) are polled and the timer which is a software timer pauses execution after checking for any input to ensure no button bounce occurs.

## Sequence Diagrams

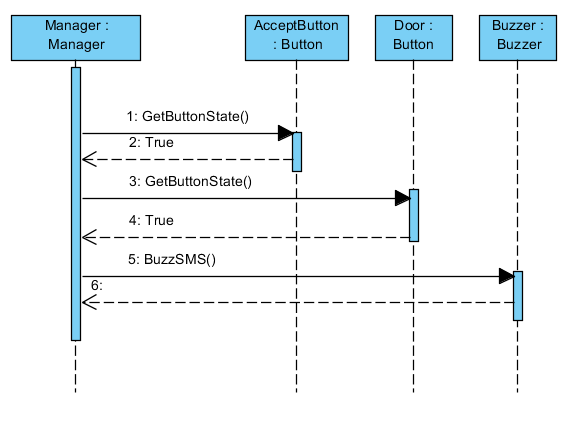


Figure 6 - Start Program Door Open

Figure 6 shows the function calls for the user pressing the accept button while the door is open, this must be acknowledged by the manager as the input is polled.

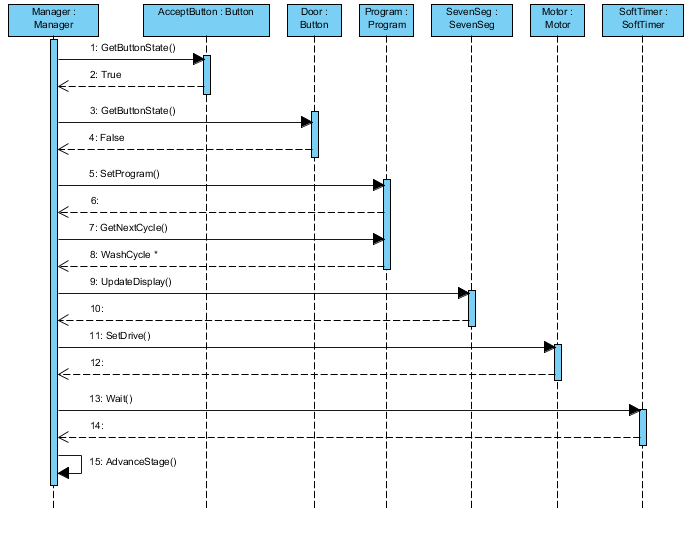


Figure 7 - Start Program Door Closed

Figure 7 shows the function calls for the user pressing the accept button while the door is closed, this must be acknowledged by the manager as the inputs are polled. Once the user has entered a valid program then the wash will start. Each wash loops through checking the inputs and waiting for a fraction of a second. Once the wash cycle time has completed then AdvanceStage() is called and the next wash cycle information is loaded.

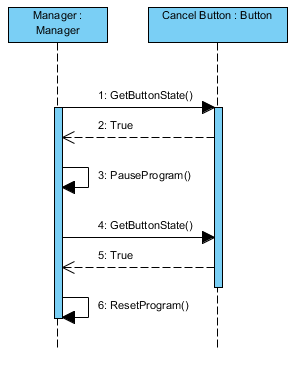


Figure 8 - Reset Program

Figure 8 shows the function calls for pausing and resetting the program by pressing the cancel button. See figure 9 for PauseProgram(). ResetProgram() calls the function AdvanceStage() until the current stage is equal to complete (the final stage of any wash).

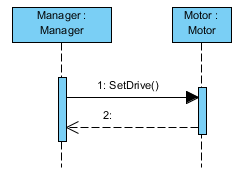


Figure 9 - PauseProgram()

Figure 9 shows the function PauseProgram(). Time is already paused so only drive needs to be paused. SetDrive() is called with an argument of Off.

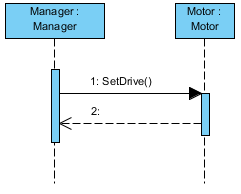


Figure 10 – ResumeProgram()

Figure 10 shows the function ResumeProgram(). SetDrive() is called with an argument of what the desired speed for the motor is for the current cycle.

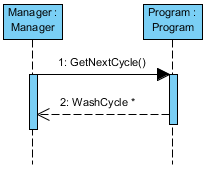


Figure 11 - AdvanceStage()

Figure 11 shows the function calls for advancing a stage in a program. The function GetNextCycle() returns a pointer to the next cycle in the program array for the wash.

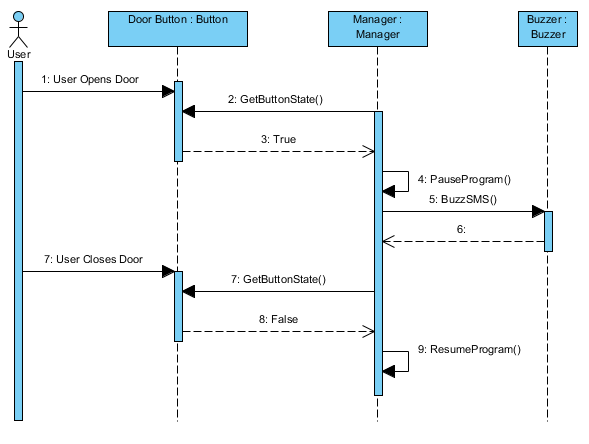


Figure 12 - Door Open While Running

Figure 12 shows the function calls after the user opens and then closes the door while a program is running. See figure 9 for PauseProgram().See figure 10 for ResumeProgram(). As the inputs are polled the action of opening the door does not initiate a function call but instead the manager must acknowledge the status of the door.

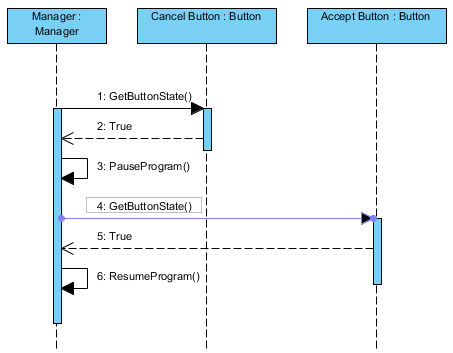


Figure 13 - Pause & Resume Program

Figure 13 shows the function calls after the user presses the cancel button and then the accept button while a program is running. See figure 9 for PauseProgram().See figure 10 for ResumeProgram(). As the inputs are polled the action of opening the door does not initiate a function call but instead the manager must acknowledge the status of the buttons.

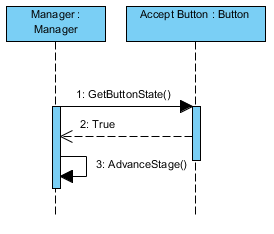


Figure 14 - Accept Button While Running

Figure 14 shows the function calls after the accept button is pressed while a program is running. Figure 11 shows the function AdvanceStage().

## Ease of maintenance

With our project is an included program called FileConverter.exe. This allows for easy modification of source code files to update the program array and modify the programs/washcycles included with the software. This enables someone who would be unable to modify the source code correctly to be able to modify a simpler text file with the program information in a user friendly way.

It converts whatever is in the file Programs.txt in the specified format and modifies the source code files program.cpp and program.h.

The format to enter in Programs.txt is:

1:ColourWash

Fill, 5, Off

Heat, 2, Off

Wash, 3, Low

Empty, 4, Off

Fill, 4, Off

Rinse, 4, Low

Empty, 3, Off

Spin, 6, High

Dry, 5, Off

Complete, 0, Off

After running FileConverter.exe, this would modify the program array generated in the program constructor in program.cpp to:

Programs[0][0] = WashCycle(Fill, 5, Off);

Programs[0][1] = WashCycle(Heat, 2, Off);

Programs[0][2] = WashCycle(Wash, 3, Low);

Programs[0][3] = WashCycle(Empty, 4, Off);

Programs[0][4] = WashCycle(Fill, 4, Off);

Programs[0][5] = WashCycle(Rinse, 4, Low);

Programs[0][6] = WashCycle(Empty, 3, Off);

Programs[0][7] = WashCycle(Spin, 6, High);

Programs[0][8] = WashCycle(Dry, 5, Off);

Programs[0][9] = WashCycle(Complete, 0, Off)

This would also modify the definitions MaxPrograms and MaxCycles in program.h to the correct values.