Smart Car Washing System – ESIOT 2023/2024

Francesco Cipollone
Raffaele Francesco Marrazzo

1. Design and Architecture

Smart Car Washing System is a task-based embedded software that is composed by a cooperative scheduler that manages six tasks, modelled as synchronous finite state machines (*FSM*).

The tasks cooperate with each other via shared variables (referred to from now on as *guards* or *guard variable*), that allow each finite state machine to adjust their current state.

The `GuardsManager` entity controls the aforementioned variables, providing encapsulation and simplicity.

2. Scheduler

The cooperative scheduler’s period is defined by the Greatest Common Divisor (GCD) calculated from all the tasks’ periods. This avoids missing relevant events originated from the other tasks.

Every time that the scheduler’s period elapses, the scheduler checks for each task’s readiness, launching them if said condition is satisfied.

3. Tasks

The tasks' periods have been chosen to avoid missing events, according to the Minimum Event Separation Time (MEST).
The following table contains the tasks with their related periods.

	Period (ms)
Check-in	400
Gate	500
Button	100
Blink led	$100\ (led\ L1)$, $500\ (led\ L2)$
Washing	200
Check-out	400

3.1 Check-In

The Check-In task manages the arrival of a car to the washing system by using a passive infrared sensor (PIR) and a sonar.

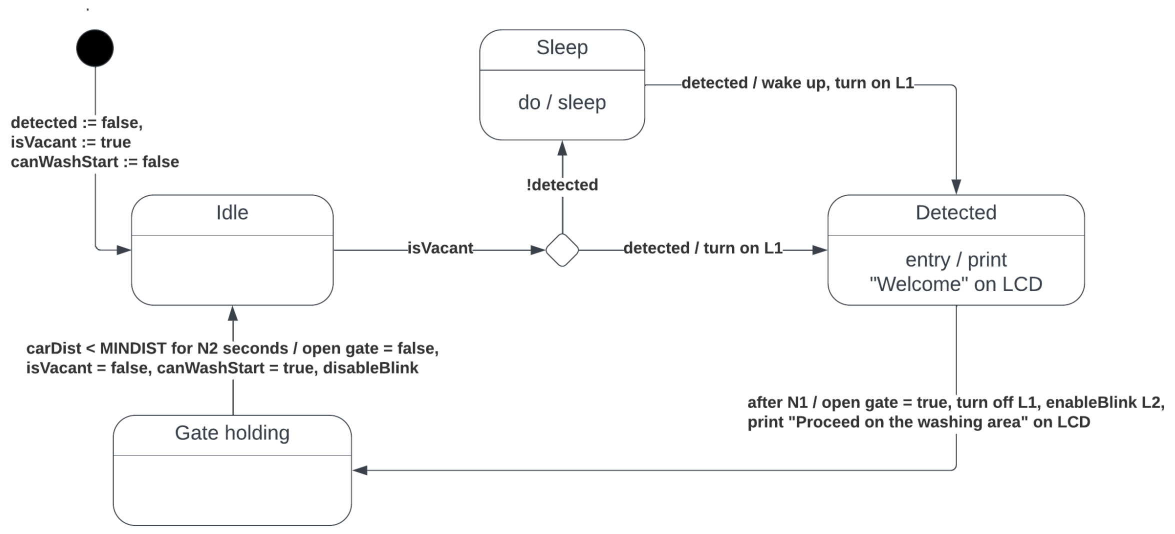
The task starts in an *idle* state: if nothing is detected by the PIR, the FSM goes into a state of *sleep*, otherwise if something is detected by the PIR for more than N1 seconds, the led L1 is turned on and a welcome message is printed on the LCD screen.

The gate, modelled with a servo motor, is then opened to let the car through and is subsequently closed after the sonar perceives the presence of the car for more than N2 seconds. During this state (*detected*) L1 is turned off and L2 starts blinking as well.

After the gate is closed, L2 stops blinking, the task goes back into the *idle* state and the system is ready to start the washing process.

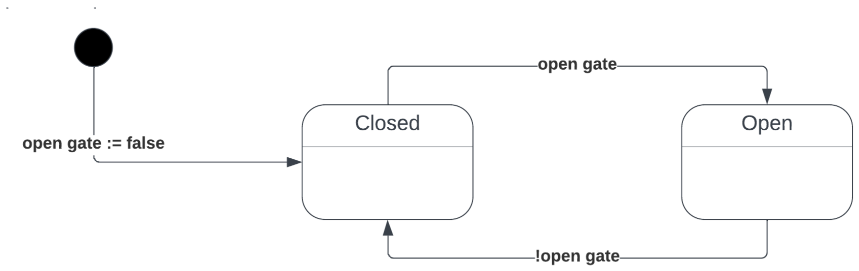
The guard variables in this FSM, `isVacant` and `canWashStart`, are respectively used to synchronise between the check-out task and the washing task.

The `isVacant` guard is used to check if the washing area is occupied by a car and `canWashStart` is used by the washing task to check if the washing process can start.


Sleep
do / sleep
detected / wake up, turn on L1
detected := false,
isVacant := true
canWashStart := false
!detected
Idle
isVacant
detected / turn on L1
Detected
entry / print "Welcome" on LCD
carDist < MINDIST for N2 seconds / open gate = false, isVacant = false, canWashStart = true, disableBlink
after N1 / open gate = true, turn off L1, enableBlink L2, print "Proceed on the washing area" on LCD
Gate holding

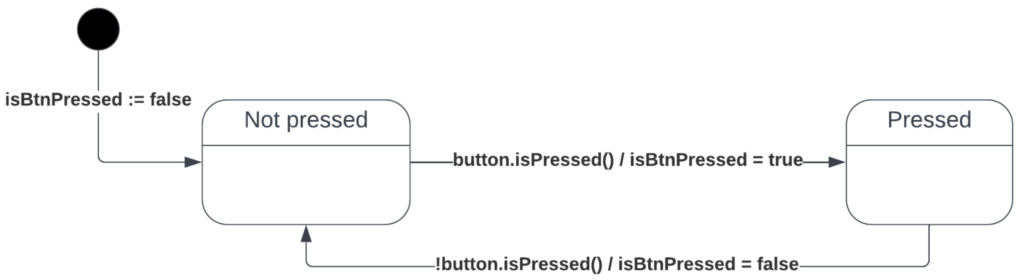
3.2 Gate

The gate task manages the opening and the closing of the gate. When the gate is closed and another task sets the *open gate* variable to `true`, the gate is opened. On the other hand, if the gate is open and a task sets the before-mentioned variable to `false`, the gate is closed.



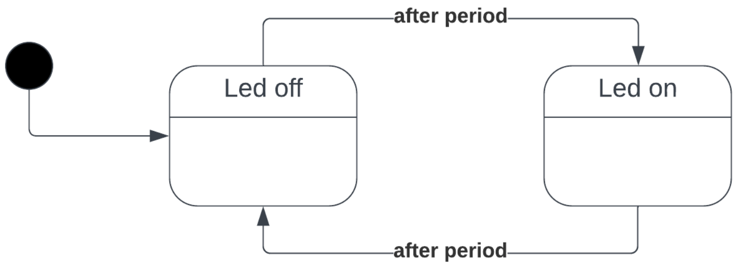
3.3 Button

The button task manages the pressing of the button. When the machine is in the state *Not pressed*, if the button is pressed, the machine goes in the *pressed* state.
Otherwise, if the machine is in the *Pressed* state, when the button is no more pressed, the machine switches in the *Not pressed* state.


isBtnPressed := false
Not pressed
button.isPressed() / isBtnPressed = true
Pressed
!button.isPressed() / isBtnPressed = false

3.4 Blink led

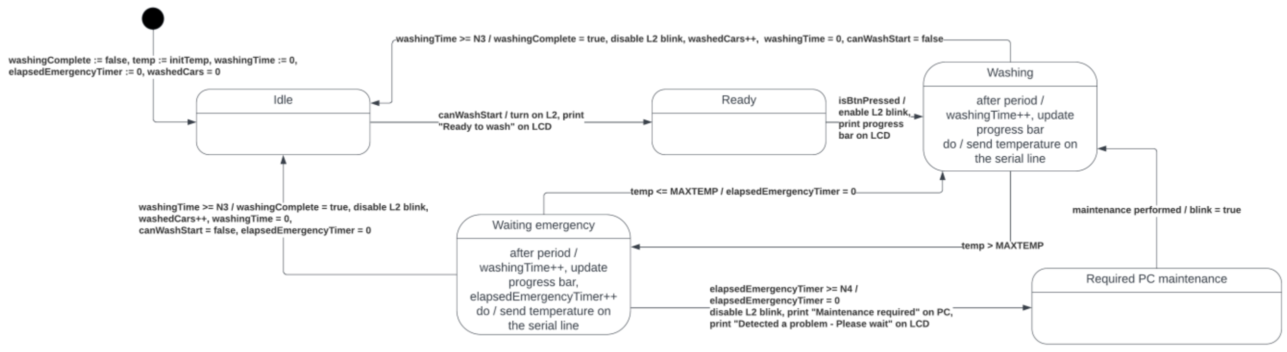
This task manages the blinking of a led. When the task is started, the LED is off. After *period* seconds, the LED is switched on, and after *period* seconds, the LED is switched off, and so on.



3.5 Washing

The *Washing* state manages the washing phase. At the beginning, the machine is in the *Idle* state. When the Check-in task is terminated, the variable `canWashStart` is set to `true`, and so the machine goes into the *Ready* state. In this state, the machine waits for the user to press the button to start the washing phase. During the washing, a progress bar is shown on the user monitor, and the temperature of the system is sampled to avoid overheating. If the temperature is greater than MAXTEMP, then the machine goes in the *Waiting Emergency* state. In this state the washing process continues but, if the temperature is greater than MAXTEMP for more than N4 seconds, the FSM goes in the *Required PC Maintenance* state. In this state the user must wait for the maintainer to solve

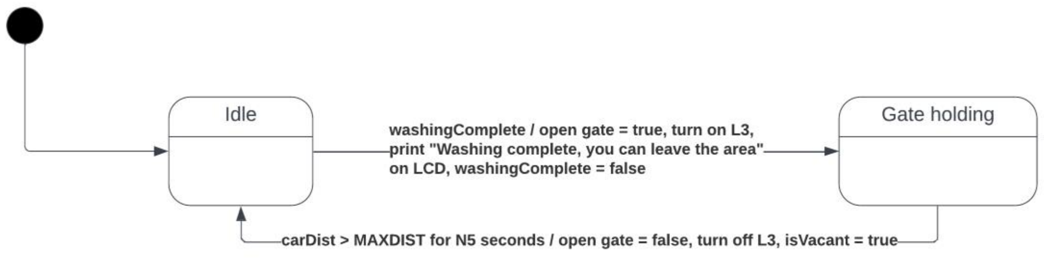
the issue. When the problem is solved, the system continues to wash the car, by returning in the *washing* state. When the washing process is finished, the FSM goes back to the *Idle* state.
The `washingComplete` guard signals the check-out task that the washing process has ended.



washingComplete := false, temp := initTemp, washingTime := 0, elapsedEmergencyTimer := 0, washedCars = 0
washingTime >= N3 / washingComplete = true, disable L2 blink, washedCars++, washingTime = 0, canWashStart = false
Idle
canWashStart / turn on L2, print "Ready to wash" on LCD
Ready
isBtnPressed / enable L2 blink, print progress bar on LCD
Washing
after period / washingTime++, update progress bar
do / send temperature on the serial line
temp <= MAXTEMP / elapsedEmergencyTimer = 0
washingTime >= N3 / washingComplete = true, disable L2 blink, washedCars++, washingTime = 0, canWashStart = false, elapsedEmergencyTimer = 0
Waiting emergency
after period / washingTime++, update progress bar, elapsedEmergencyTimer++
do / send temperature on the serial line
temp > MAXTEMP
maintenance performed / blink = true
elapsedEmergencyTimer >= N4 / elapsedEmergencyTimer = 0
disable L2 blink, print "Maintenance required" on PC, print "Detected a problem - Please wait" on LCD
Required PC maintenance

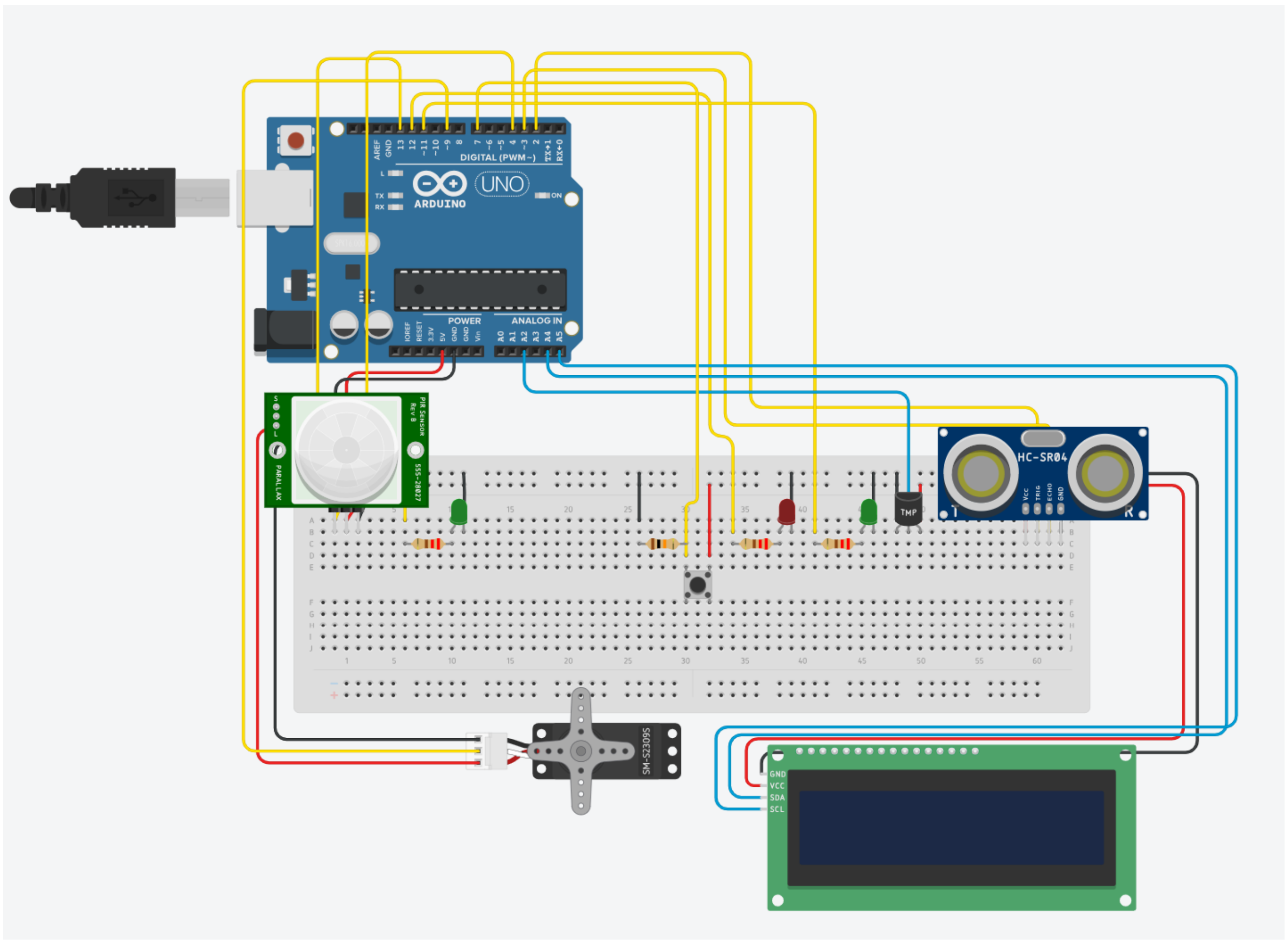
3.6 Check-Out

The *Check-Out* task starts in the *Idle* state. When the washing phase is terminated, the gate is opened and, when the sonar measures a distance greater than MAXDIST for N5 seconds, the FSM closes the gate and comes back in the *Idle* state.



Idle
washingComplete / open gate = true, turn on L3, print "Washing complete, you can leave the area" on LCD, washingComplete = false
Gate holding
carDist > MAXDIST for N5 seconds / open gate = false, turn off L3, isVacant = true

4. Circuit Design



DIGITAL (PWM~)
UNO
ARDUINO
POWER
ANALOG IN
HC-SR04
TMP
PIR Sensor
PARALLAX
SM-S2309S
GND
VCC
SDA
SCL