CoffeeTime - Firmware Overview

The **CoffeeTime** firmware is modular, ensuring **scalability, readability, and maintainability**. It integrates multiple **embedded system components** to automate coffee preparation, handling sensors, actuators, user interaction, and scheduling.

📂 Project Structure

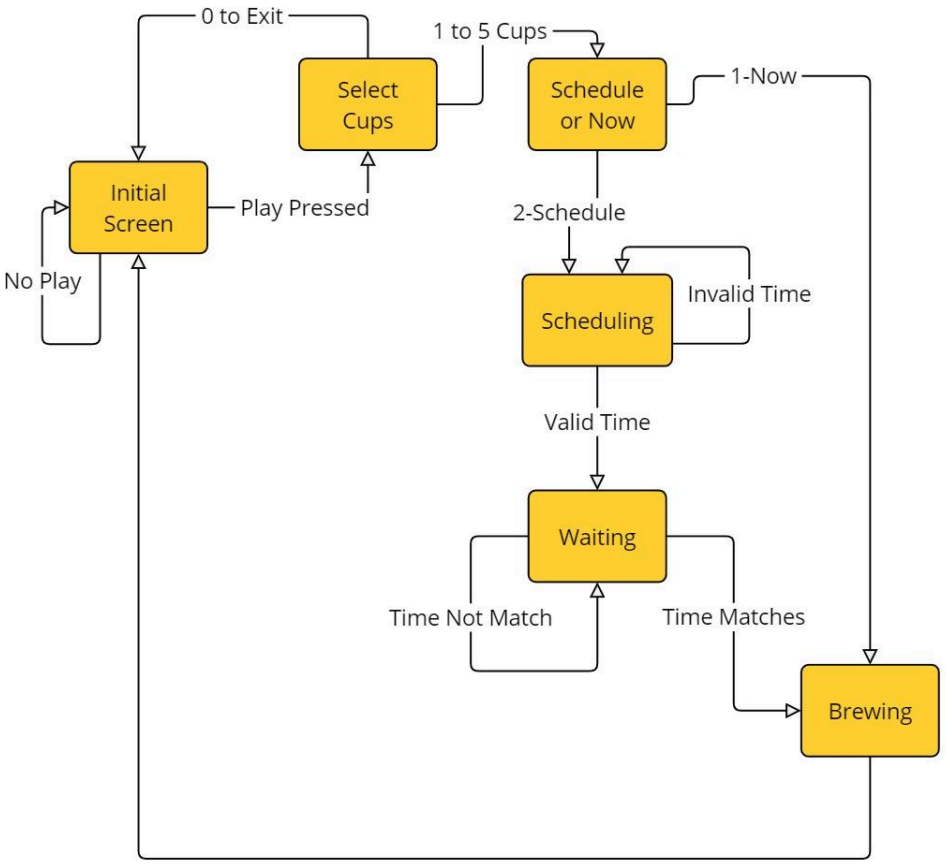
```
📂 CoffeeTime-SmartCoffeeMachine
├── main.c                  → Main function and control loop
├── sensors.h / sensors.c   → ADC (potentiometers), DHT22, and RTC integration
├── actuators.h / actuators.c  → LED bar, buzzer, servo motors, and stepper motor
├── user_interface.h / user_interface.c → Menus, screens, and user interaction logic
├── state.h / state.c       → Machine state management and transitions
├── ir_control.h / ir_control.c → IR remote control command processing
└── lcd_i2c.h / lcd_i2c.c   →LCD display control and animations
```

Functional Overview

Function	Description	Module
`read_dht22()`	Reads temperature and humidity from DHT22	**sensors.c**
`rtc_get_time()`	Retrieves the current time from the RTC	**sensors.c**
`read_intensity()`	Reads coffee intensity from potentiometer	**sensors.c**
`grind_coffee()`	Controls stepper motor for coffee grinding	**actuators.c**
`servo_dispense_water()`	Controls servo to dispense water	**actuators.c**
`schedule_coffee()`	Uses RTC to start coffee preparation at set time	**state.c**
`ir_handle_input()`	Processes IR remote input	**ir_control.c**
`lcd_display_status()`	Updates brewing status on LCD	**lcd_i2c.c**

State Machine Diagram

The **CoffeeTime** firmware operates as a **finite state machine**, transitioning through distinct states based on user input and system conditions:



Idle State (`STATE_INITIAL_SCREEN`)

- The system is waiting for user input.
- The LCD displays water and coffee bean levels, as well as temperature and humidity.
- The **IR remote control** is used to start the coffee-making process.
- **Transitions to:** `STATE_SELECT_CUPS` when PLAY is pressed.

Select Cups (`STATE_SELECT_CUPS`)

- The user chooses the number of cups (1-5).
- **Transitions to:** `STATE_SCHEDULE_OR_NOW`.

Schedule or Immediate Brewing (`STATE_SCHEDULE_OR_NOW`)

- The user selects:
  Option 1 → Brew coffee **now** → `STATE_BREWING`.
  Option 2 → Schedule coffee brewing → `STATE_SCHEDULING`.

Scheduling Coffee (`STATE_SCHEDULING`)

- The system prompts the user to **set a date and time** using the RTC.
- Once a valid time is set, the machine enters `STATE_WAITING`.

Waiting for Scheduled Time (`STATE_WAITING`)

- The system continuously **monitors the RTC** for the scheduled time.
- When the clock matches the scheduled time → Transition to `STATE_BREWING`.

Brewing Process (`STATE_BREWING`)

- **Grinding Phase:** The **stepper motor** grinds coffee beans.
- **Heating Phase:** The **water heater simulation** warms water to the desired temperature.
- **Extraction Phase:** The **servo motors** dispense water and ground coffee into the cup and the **LED bar** updates based on coffee strength.

- **Transitions to:** `STATE_INITIAL_SCREEN` once brewing is complete.

Final State

Once brewing is finished, the system returns to **Idle State**, displaying:

- “Coffee is Ready!” on the LCD.
- Buzzer melody signals completion.
- LED indicators turn off.

This state machine ensures a **structured and modular coffee preparation process** using **sensors, actuators, and real-time scheduling**.

Memory Organization

Variable	Purpose	Data Type
`coffee_strength`	Stores the selected coffee strength level	`int`
`brew_temperature`	Stores the brewing temperature in °C	`float`
`scheduled_time`	Holds the user-defined brewing time	`struct`
`grind_active`	Indicates if the grinder is running	`bool`
`water_ml`	Tracks available water volume	`float`
`coffee_beans_g`	Tracks remaining coffee beans	`float`

Example Code Snippet

```
void prepare_coffee(int cups) {
    int pressure = read_intensity();
    float desired_temp = read_desired_temperature();
    check_simulated_resources(cups);

    update_led_bar(pressure);
    simulate_water_heating(desired_temp);

    grind_coffee();
    servo_dispense_water();

    lcd_display_status("Brewing...");
    play_success_tone();
}
```

This document provides a structured overview of the **CoffeeTime** firmware, detailing:

- **Project architecture & file structure**
- **Core functionalities of each module**
- **State machine logic**
- **Memory organization and variable usage**

With this modular approach, CoffeeTime efficiently manages **user interaction, coffee brewing, and automated scheduling**.