SMiLE – Software Functional Specifications

1 Main Operational Modes

1.1 Injection and Recording

A linear actuator stepper motor pumps water from the reservoir into the viewing chamber. The outlet solenoid valve is set to open state. Droplet formation will be recorded by two Raspberry Pi cameras. Droplets are illuminated by electroluminescent panels. Temperature information is recorded.

1. Test number looked up
2. Outlet solenoid valve set to open, inlet solenoid valve is set to closed
3. Electroluminescent panels are activated
4. Raspberry Pi cameras start recording
5. Stepper motor is extended (operating at specific steps per second, for a specified amount of time)
6. Temperature information is recorded (how are we going to handle this? Single value average over time of injection?).

1.2 Spin Up and Recording

Once the reservoir is depleted and the stepper motor has ceased operation, the centrifuge motor will start operating. Centrifugal force is expected to cause the droplets to move to the sloped outer wall of the viewing chamber, and from there be funnelled to the return line and on to the reservoir. The solenoid valves will be closed in this initial spin up, and the cameras will still be recording.

1. Outlet solenoid valve set to closed. Inlet solenoid valve set to closed.
2. Electroluminescent panels activated
3. Raspberry Pi cameras recording
4. Temperature information recorded
5. Centrifuge motor is activated (ramp up to full rpm)

1.3 Drainage and Return

At this point, the cameras are turned off, and the stepper motor is run in reverse. This creates a back pressure on the collected water and draws the water to the reservoir by suction. The inlet solenoid valve is set to open state. A bubble sensor monitors the fluid line leaving the viewing chamber to ensure that no air bubbles are drawn into the reservoir.

1. Outlet solenoid valve set to closed. Inlet solenoid valve set to open
2. Electroluminescent panels off
3. Raspberry Pi cameras off
4. Centrifuge motor running at full rpm
5. Stepper motor retracted (specified steps per second, specified amount of time)
6. Bubble sensor operating

1.3.1 Bubble Detected

If during drainage and return a bubble is detected, the stepper motor needs to reverse its operation to push the air bubble back into the viewing chamber. Centrifuge motor is still spinning.

1. N=N+1
2. Outlet solenoid valve set to closed. Inlet solenoid valve set to open.
3. Electroluminescent panels off
4. Raspberry Pi cameras off
5. Centrifuge motor running at full rpm
6. Stepper motor extending (specified sps, specified amount of time)
7. Bubble sensor working.
8. Wait X amount of time (for extra spinning)
    a. Go back to Drainage and Return

1.4 Charging

To power the experiment, super-capacitors are required. In between the high power consumption stages, all of the motors and lights are turned off to allow the super-capacitors to recharge.

1. When microcontroller detects certain power level, it sends signal to R-Pi.
2. All internal off (except for R-Pi)
3. Supercapacitors charged

1.5 Post-Operation Data Manipulation

After recording, the data is converted as described in Section Data Processing.

1. All internal off except for R-Pi

2 Startup

2.1 System flush

This will occur after a new start up or recovery. Inlet solenoid valve is opened, and the stepper motor retracts to starting position. Full system flush occurs (working through all operational modes, though not recording).

1. Recovery / Start up occurs
2. Charge to full capacitance
3. Outlet solenoid valve set to closed. Inlet solenoid valve is set to open.
4. Run through all operational modes with the following changes (possibly if bubble detected runs into failure state continually run it into
    a. No recording
    b. No electroluminescent panels
    c. No data manipulation

3 Error Handling

3.1 Expected Errors

3.2 Recovery mechanisms

4 Data Processing

4.1 Metadata

- Test number
- Current mode
- Temperature
- Bubble sensor failure?
- Error

- Recovery
- External input / output

4.2 Video Recording

- Saving video

4.3 Video post processing

- Centre of mass of droplets?

5 Microcontroller

5.1 Components/Capabilities

- Supercapacitor charging board
- Brains to interpret communication (USB and R-Pi)
- Switch controlling power between caps / USB and R-Pi

5.2 Monitoring

- Constantly monitoring state of supercapacitors (emergency shutdown?)

5.3 USB to Microcontroller communication

Section 5 of the ICD?

5.4 Microcontroller to R-Pi communication

- R-Pi pings microcontroller

6 Notes

- R-Pi must let micro-controller know what mode its currently in. Threshold levels of acceptable power will change according to mode.
- As part of the operational mode, Pi must ping microcontroller for current charge state – if charge is below threshold required for specific phase, charging must begin
- This leads to 'half' modes, such as charge required for a spinup phase with no recording.
- Microcontroller is never turned off. R-Pi is likely only put into stasis/sleep mode