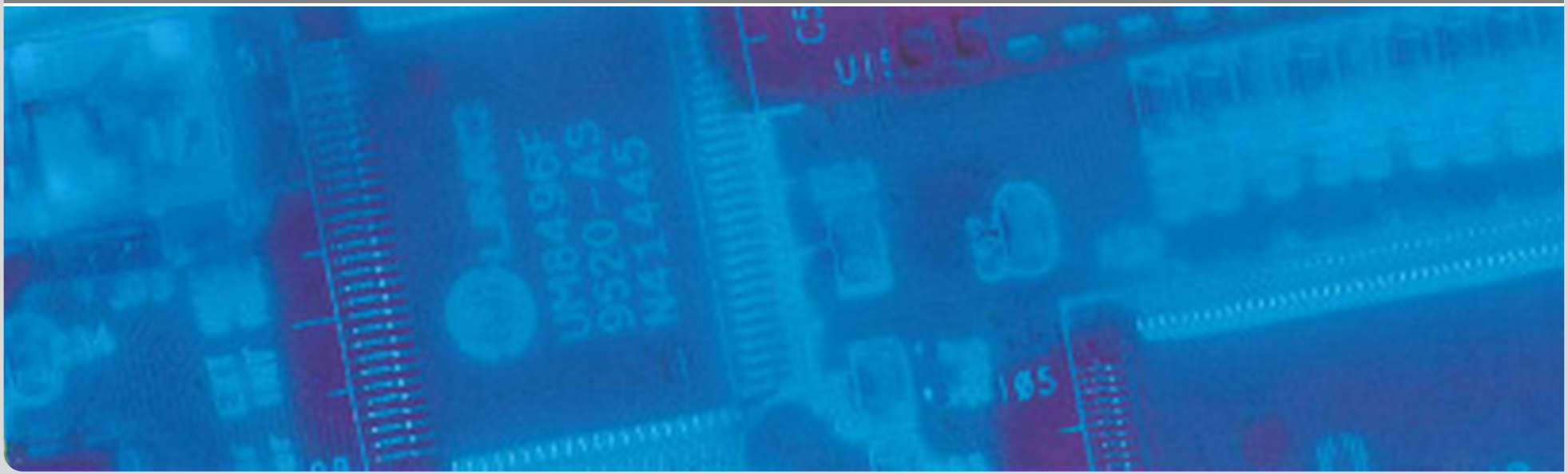
Karlsruhe Institute of Technology

Improving Usability and Reliability of an IoT-based Controller for a Coffee Machine

Simon Korz – uhelj@student.kit.edu

CES - Chair for Embedded Systems

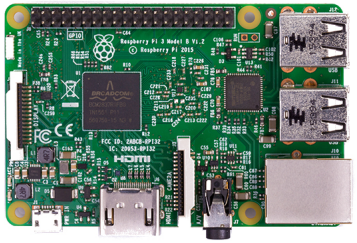


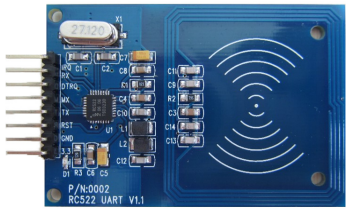
Introducing the Topic

Accounting Server

HTTP

Adapter Board





RFID tag

Goals

Solve existing issues in previous system:

- Fixing bugs
- Redesigning the UI
- Adding missing features

$\rightarrow$ **improve usability and reliability**

Structure of this Talk

Introduction

Problem Analysis

New Design & Architecture

Results

Part 1

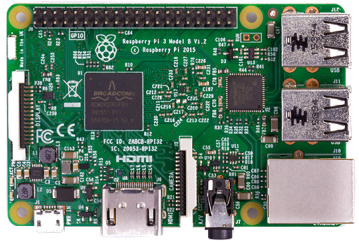
Introduction

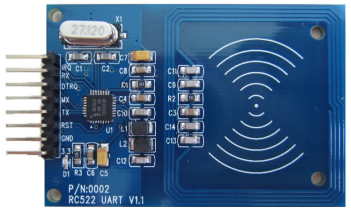
The Components

Accounting Server

HTTP

Adapter Board





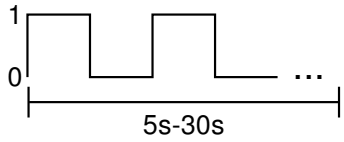
RFID tag

System Details

- Broadcom BCM2837, 4 core Cortex-A53 (ARMv8) 64-bit SoC @ 1.2GHz
- 1GB LPDDR2 SDRAM
- OS: Raspbian GNU/Linux (Debian 10 Buster)
- Language: Python
- GUI: PyQt5 Framework

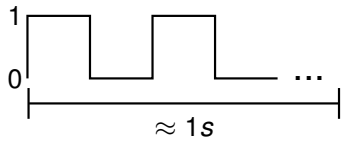
Available Sensors

- **water flow**



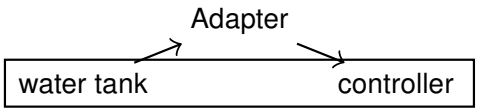
variable frequency, proportional to flow speed

- **grinder**



variable frequency

- **water level/blocking**



input AND output

adapter board connected via GPIO

Accounting Server

Treasury, keeps record of

- Transactions (withdrawals, deposits, coffee bought)
- Users
- RFIDs

Technical:

- Dedicated server, local network
- MySQL database
- Apache + PHP webserver
- Account balance results from accumulating all transaction deltas

Usability & Reliability

Usability is the "extent to which a system, product or service can be

used by specified users

to achieve specified goals

with

- effectiveness
- efficiency
- satisfaction

in a specified context of use" [ISO 9241-11:2018]

Reliability

- Result of usability
- High availability

Part 2

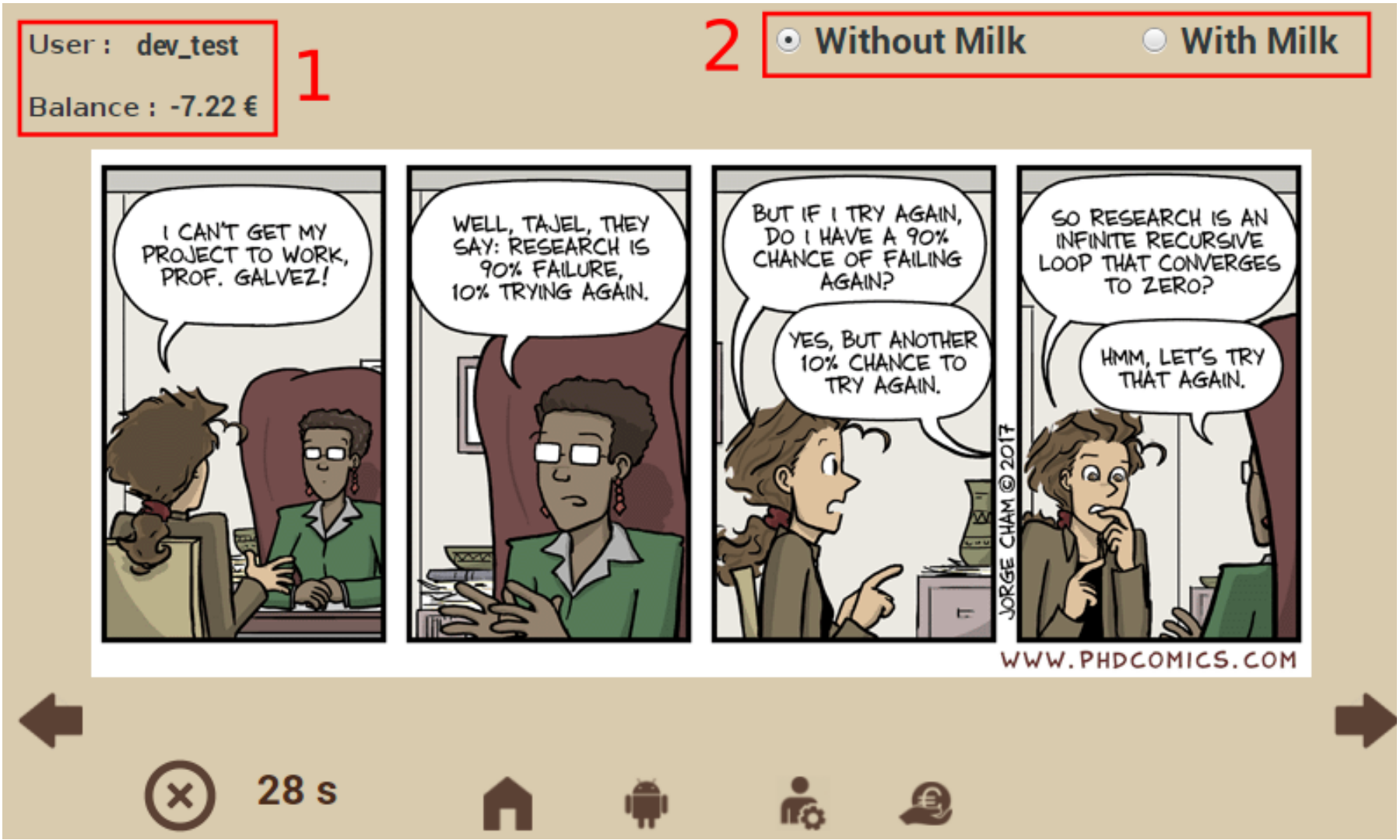
Problem Analysis

21 Issues Identified

- GUI related
  - Size of controls
  - Focus on information
  - Timeouts
- Bugs
  - Application unresponsive/freezing
  - Coffee not registered or recognized as hot water
  - Offline orders not synchronized
  - High CPU usage
- Missing features
  - Support for new KIT-Card
  - Dispensing limit for consumption
  - Various buzzer sounds
  - 3D-printed case

GUI Problems

Screenshot from previous system

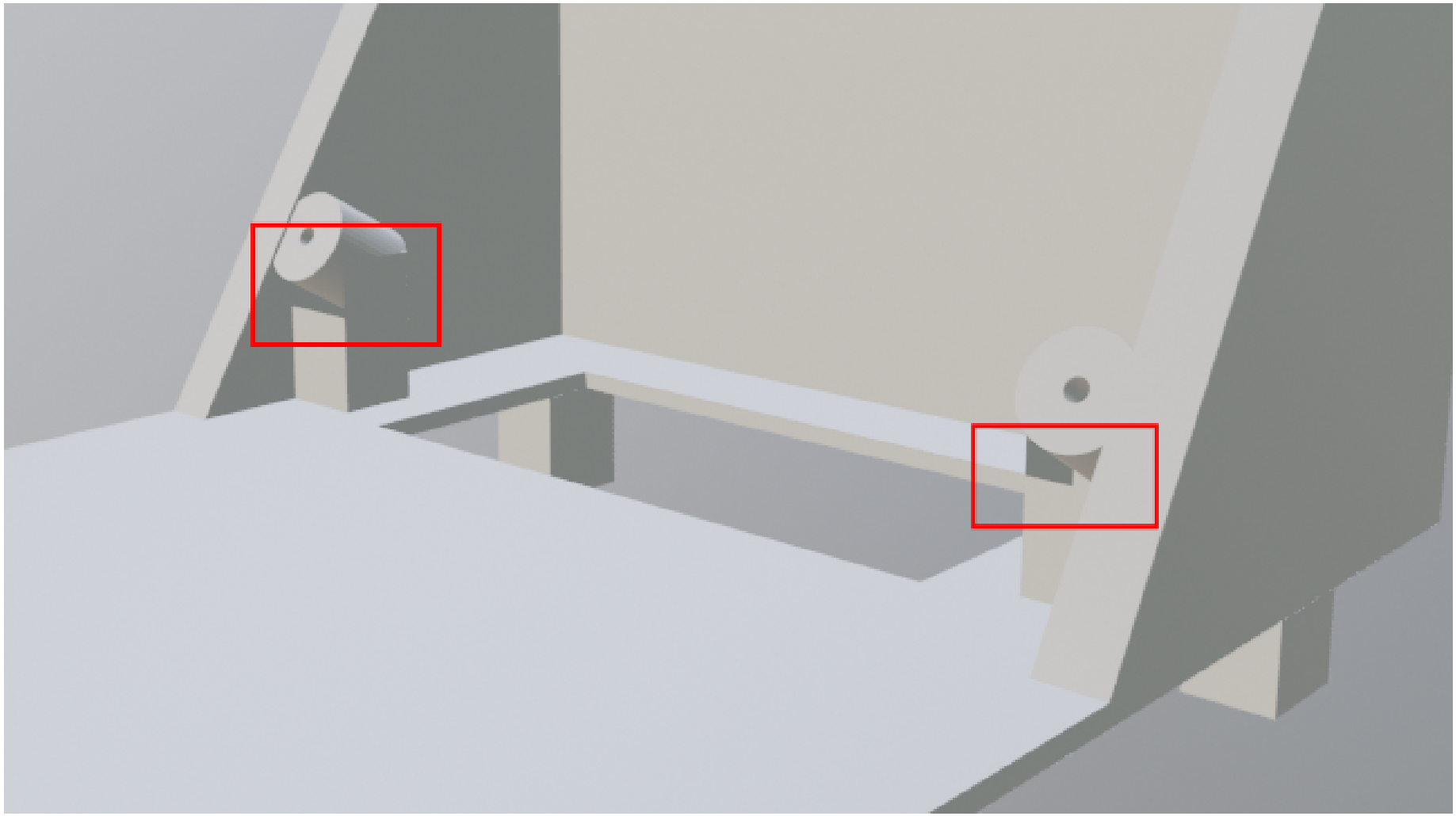


Reading GPIO + Managing State + IPC

```
1  def printLog(array):
2      ...
3      if not os.path.isfile("/home/pi/CoffeeMachine/UI/maintenance.txt"):
4          global previousGrinder
5          if GPIO.input(grinderPin) == 0 and previousGrinder == 1 and not currentOrder == 'coffee':
6              if os.path.isfile("/home/pi/CoffeeMachine/UI/order.txt") :
7                  with open("/home/pi/CoffeeMachine/UI/order.txt", "r") as j:
8                      temp = j.readline()
9                  if temp == "water\n":
10                     os.remove("/home/pi/CoffeeMachine/UI/order.txt")
11             with open("/home/pi/CoffeeMachine/UI/order.txt", "a+") as f:
12                 f.write("coffee\n")
13                 f.close()
14             with open("/home/pi/CoffeeMachine/UI/stop.txt", "a") as f:
15                 pass
16             currentOrder = 'coffee'
17             previousGrinder = 1
18         elif GPIO.input(grinderPin) == 1 and previousGrinder == 0:
19             previousGrinder = 1
20         elif not (GPIO.input(waterFlow1Pin) == previousWaterFLow1) or not (GPIO.input(waterFlow2Pin) ==
   previousWaterFLow2):
21             previousWaterFLow2 = GPIO.input(waterFlow2Pin)
22             previousWaterFLow1 = GPIO.input(waterFlow1Pin)
23             idleCount = 0
24             if not os.path.isfile("/home/pi/CoffeeMachine/UI/order.txt"):
25                 with open("/home/pi/CoffeeMachine/UI/order.txt", "a+") as f:
26                     f.write("water\n")
27                 currentOrder = 'water'
28             elif not os.path.isfile("/home/pi/CoffeeMachine/UI/unlock.txt") and os.path.isfile("/home/pi/
   CoffeeMachine/UI/order.txt") and currentOrder == 'water':
29                 with open("/home/pi/CoffeeMachine/UI/unlock.txt", "a") as f:
30                     pass
```

Unfinished Case

Top part of previous model

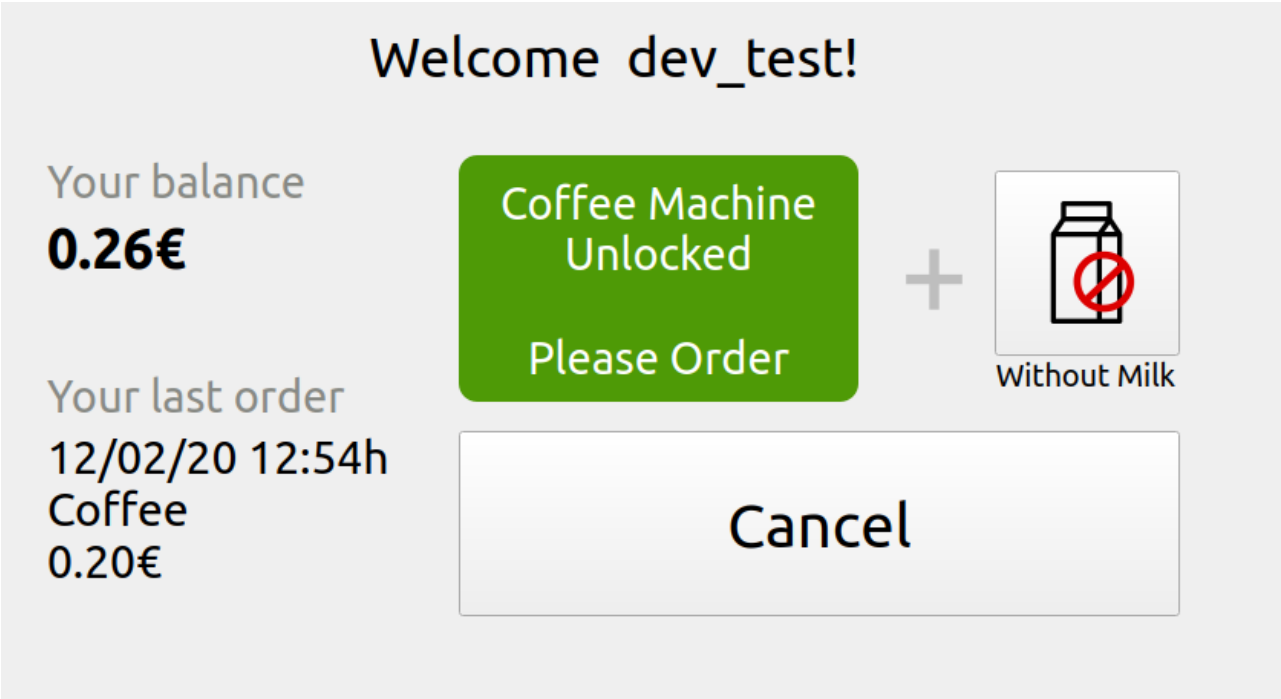


Part 3

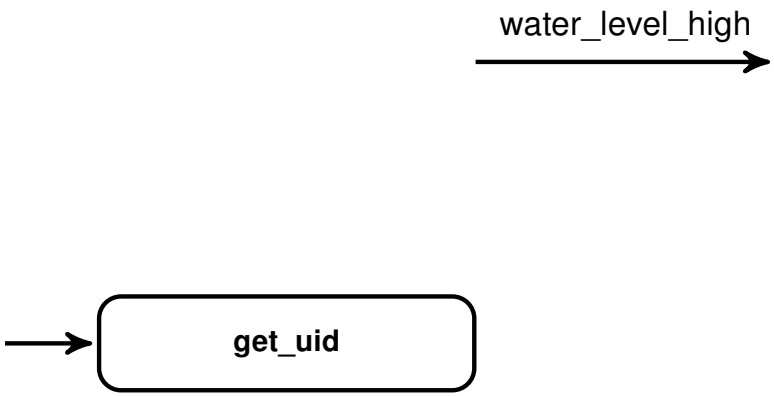
New Design & Architecture

New GUI

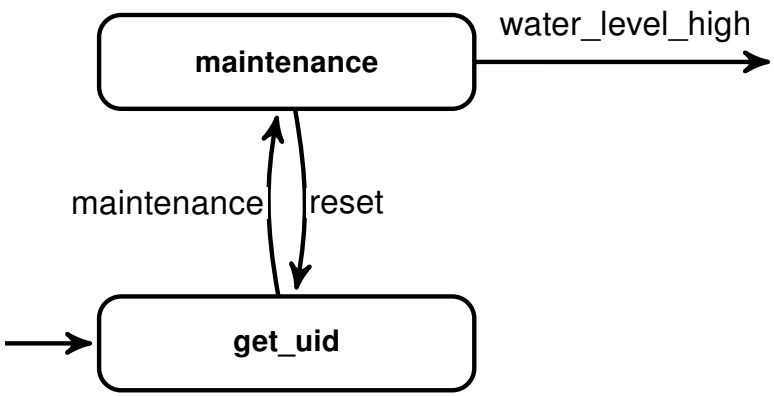
- Focus on important information
- Large buttons, suitable for touch
- Visual feedback for each step in ordering process



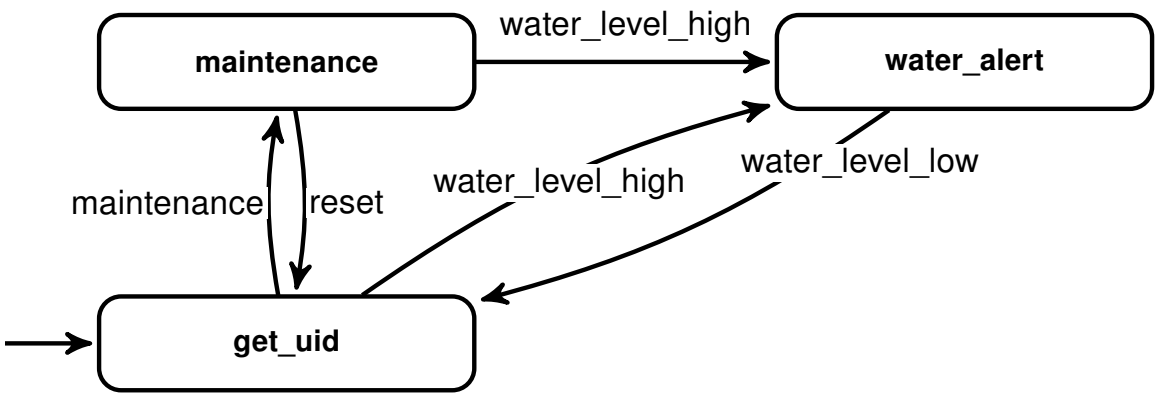
The State Machine



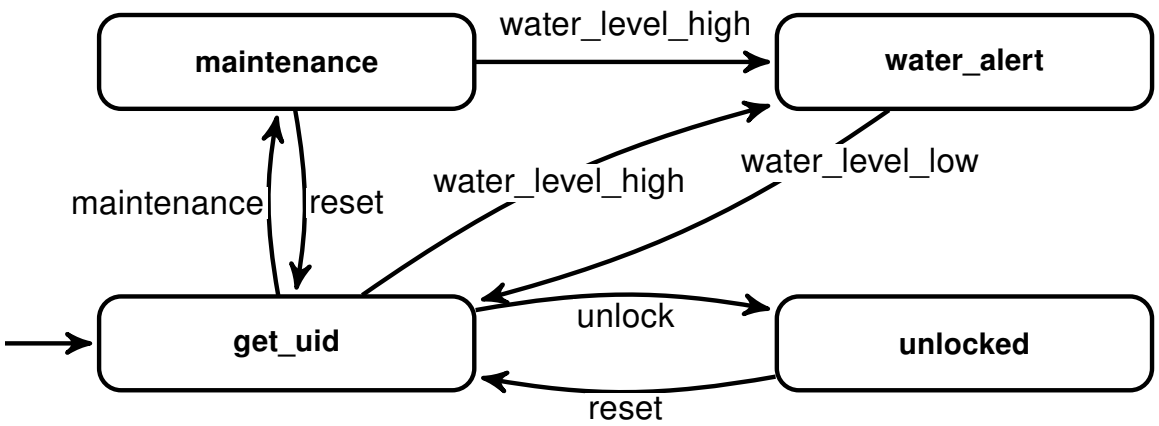
The State Machine



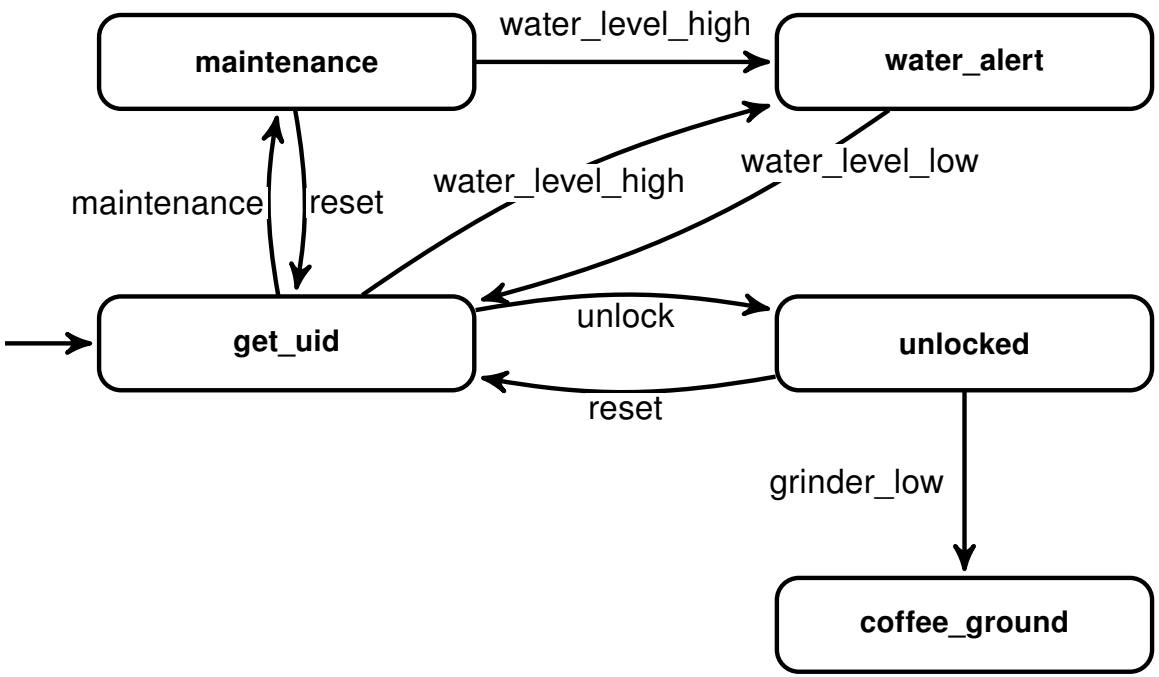
The State Machine



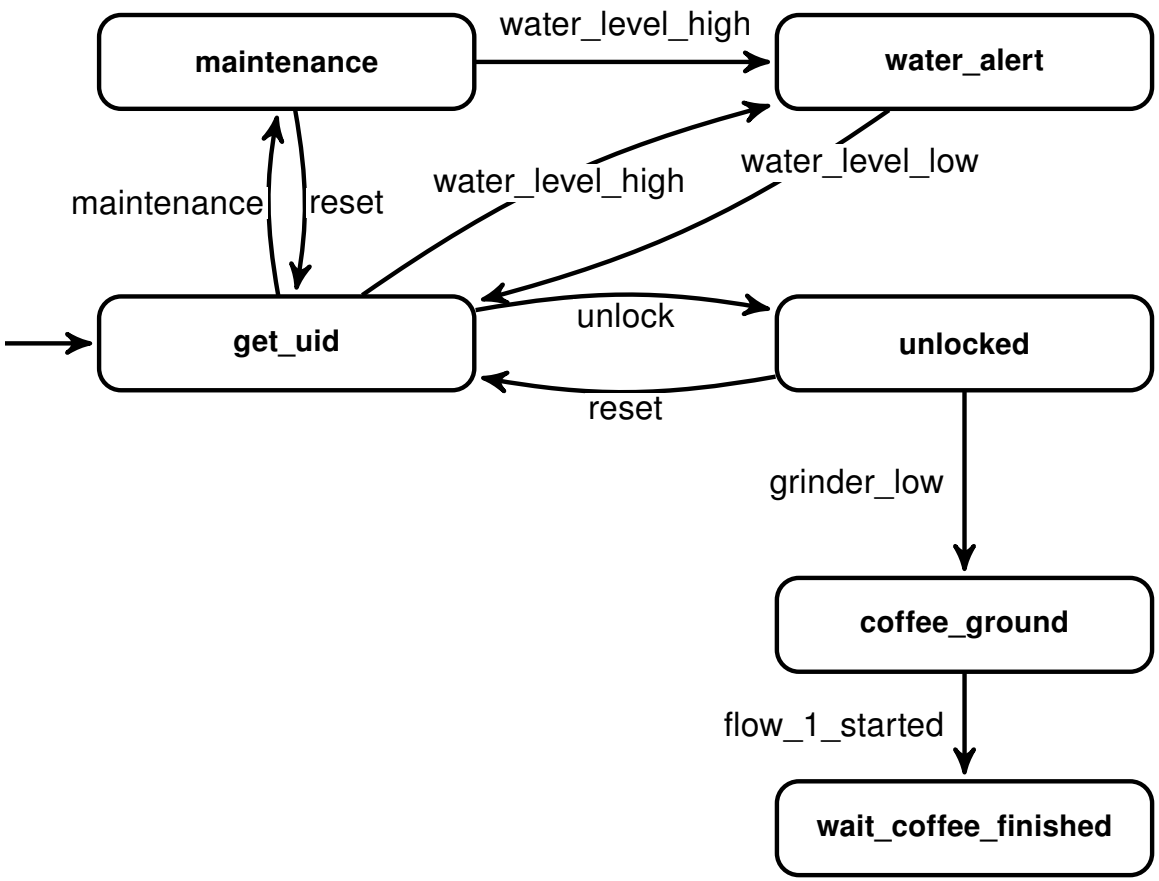
The State Machine



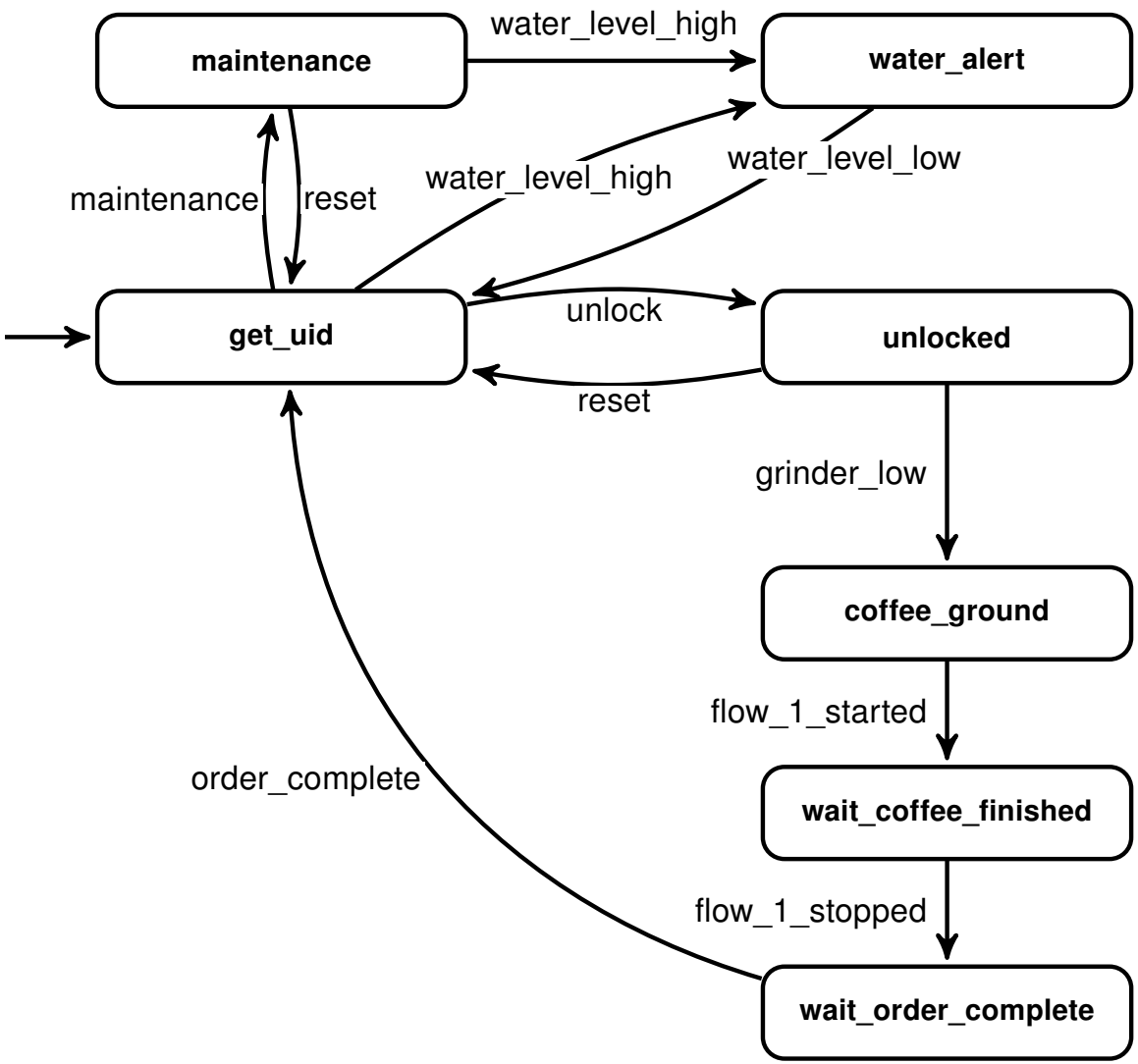
The State Machine



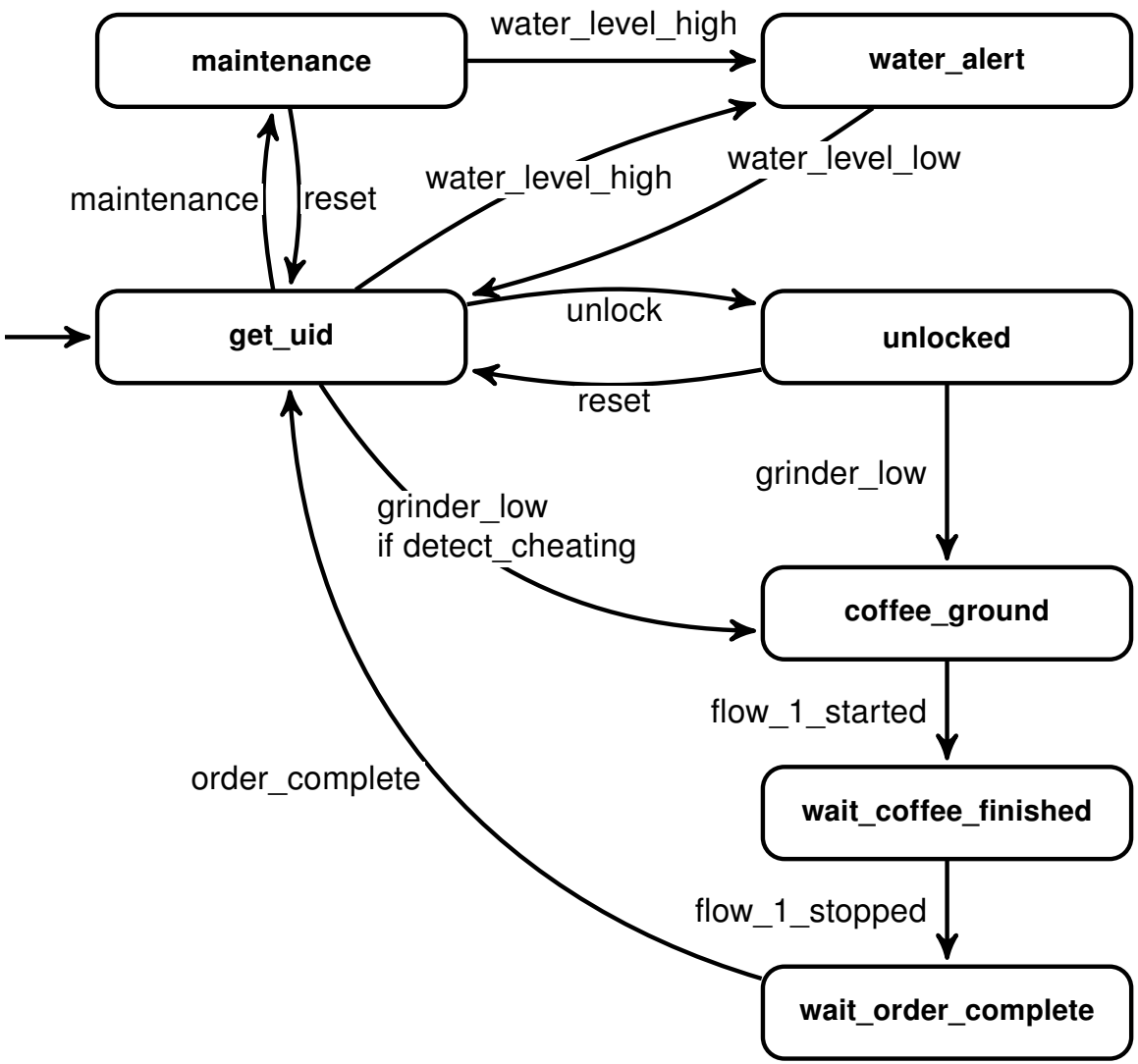
The State Machine



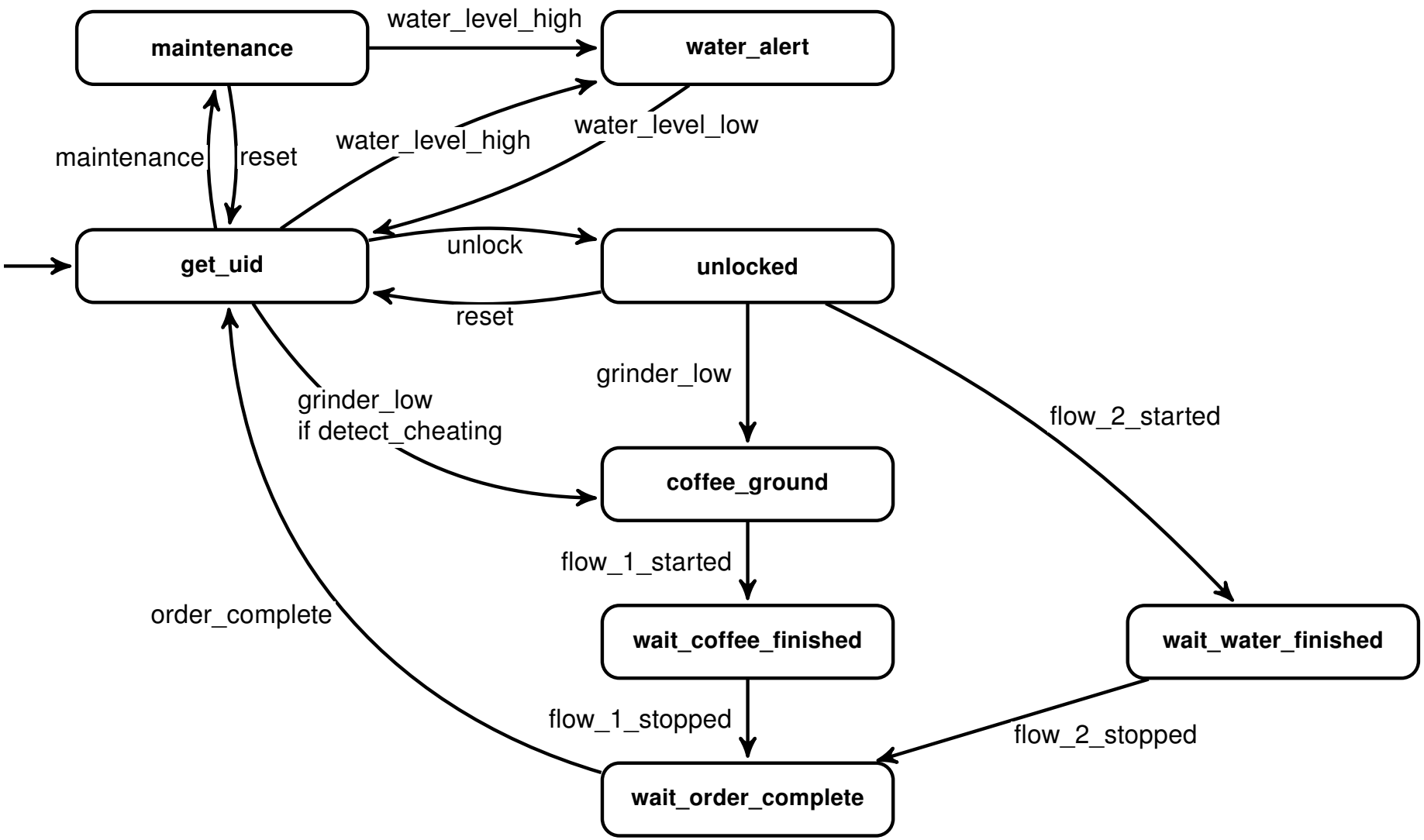
The State Machine



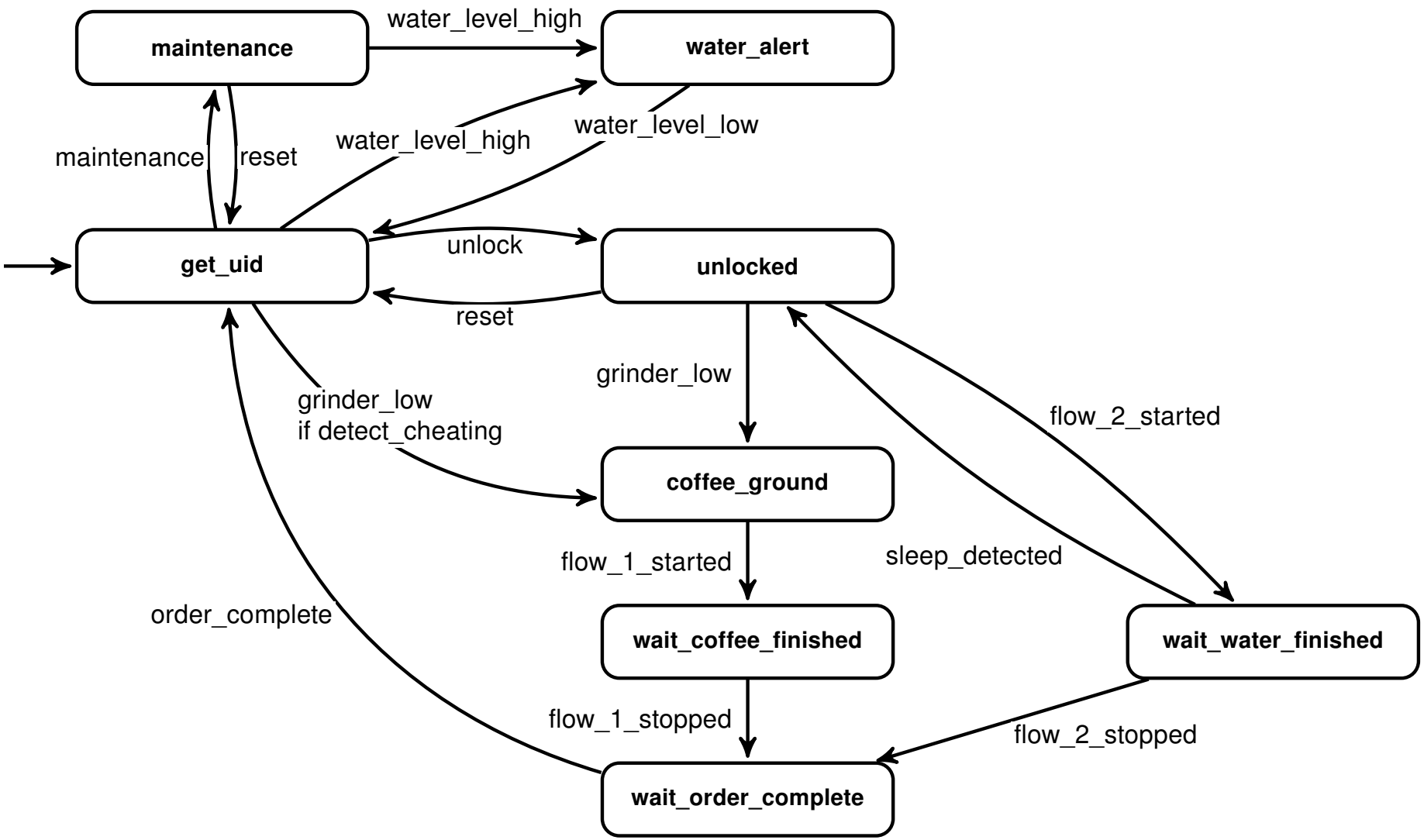
The State Machine



The State Machine

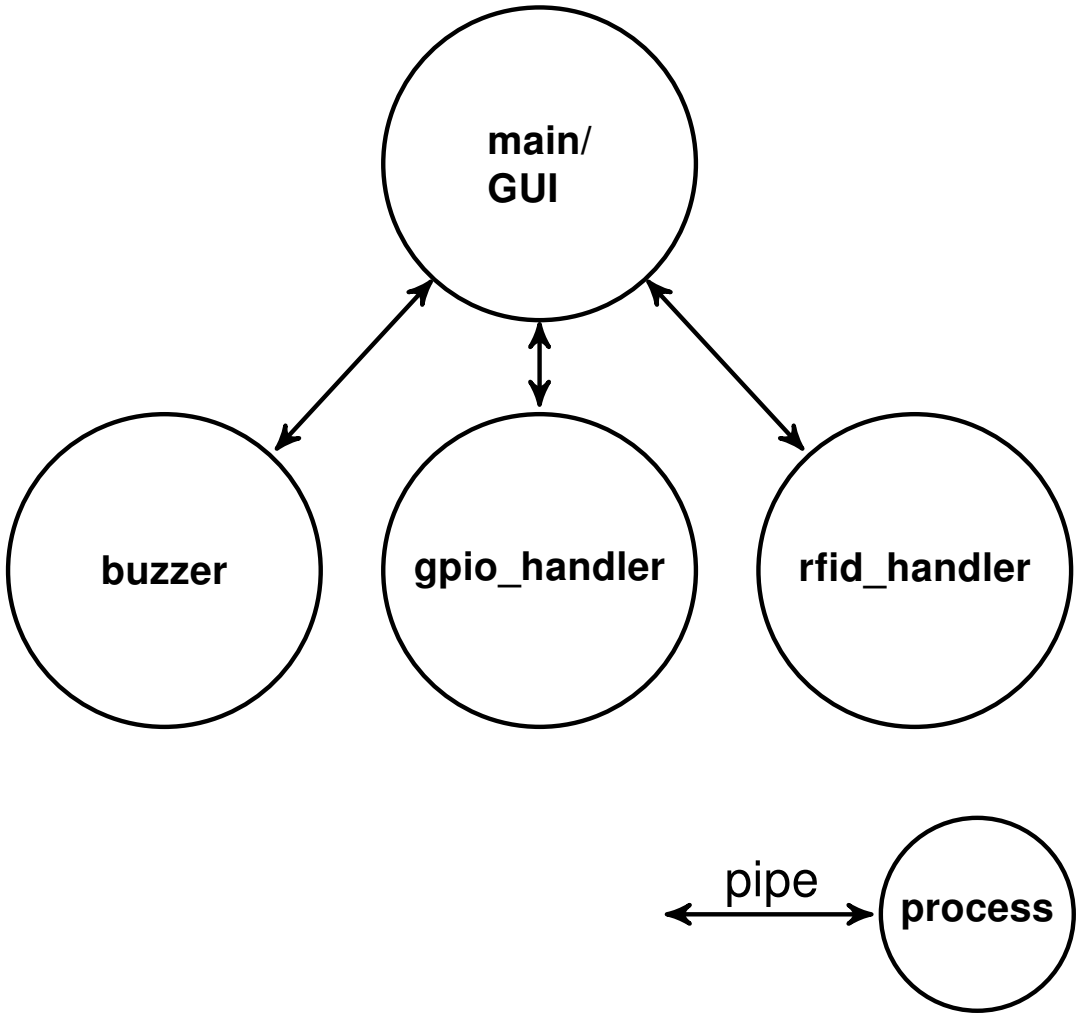


The State Machine



Leveraging Multiple CPU Cores

- 4 cores available
- Bidirectional pipes
- Predefined messages e.g, CMD_PAUSE, CMD_RESUME, CMD_LOCK, E_GOT_ID

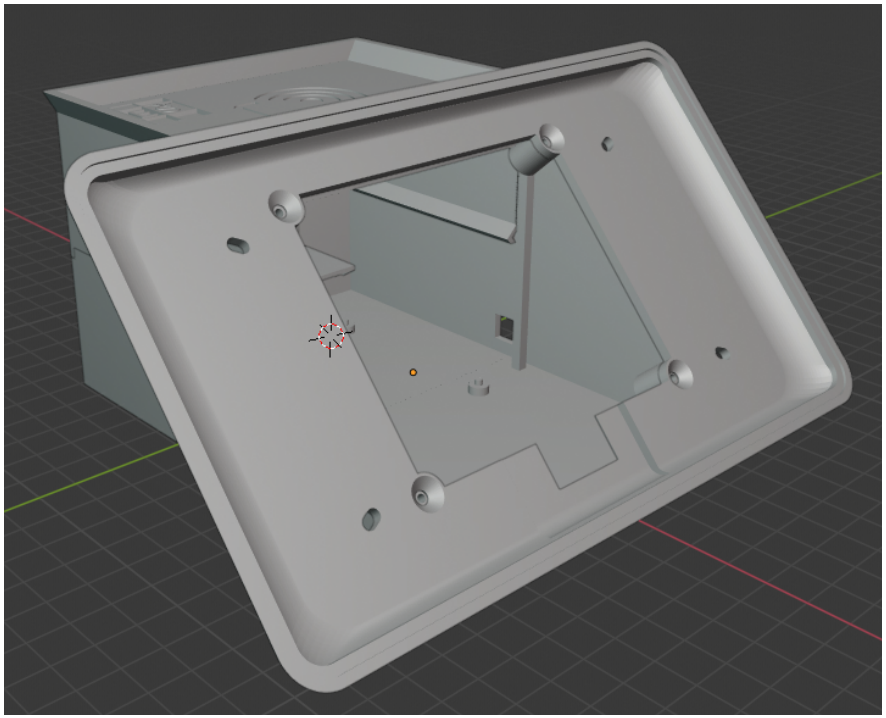


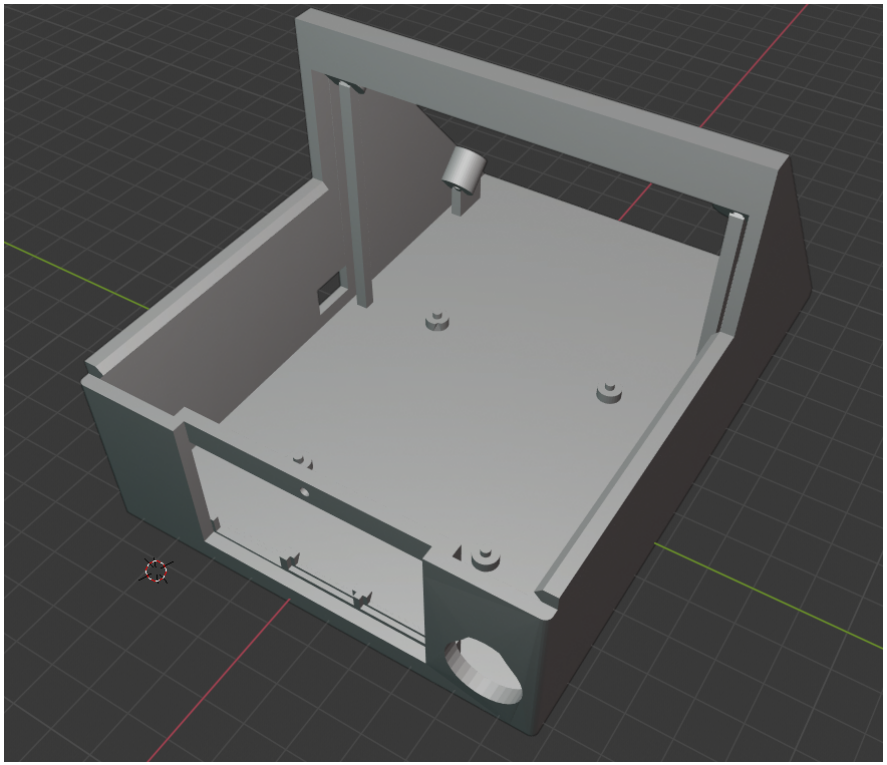
Changes in GPIO Handling

- **Callbacks** instead of **polling**
- Using pigpio library instead of RPi.GPIO
  - Supports callbacks through hardware interrupts
  - Noise filters

3D-Printed Case

- Designed with Blender
- Stereolithography printer, resin fluid hardened by laser
- Challenges: Non-manifold geometry i.e. "not watertight", walls too thin/unstable, characteristics of the material





Part 4

Results

Printing Costs

- Resin tank: €65.45
- 1L grey resin: €160.65

$$\frac{\text{resin tank } €65.45 + \text{resin } €160.65}{1000ml} = €0.2261\,/\,ml$$

- Top part: 90.79ml => €20,53
- Bottom part: 150.08ml => €33.93
- Screen frame: 53.60+ml => €12.12
- Total €66.58

Total cost estimate incl. prototypes > €300

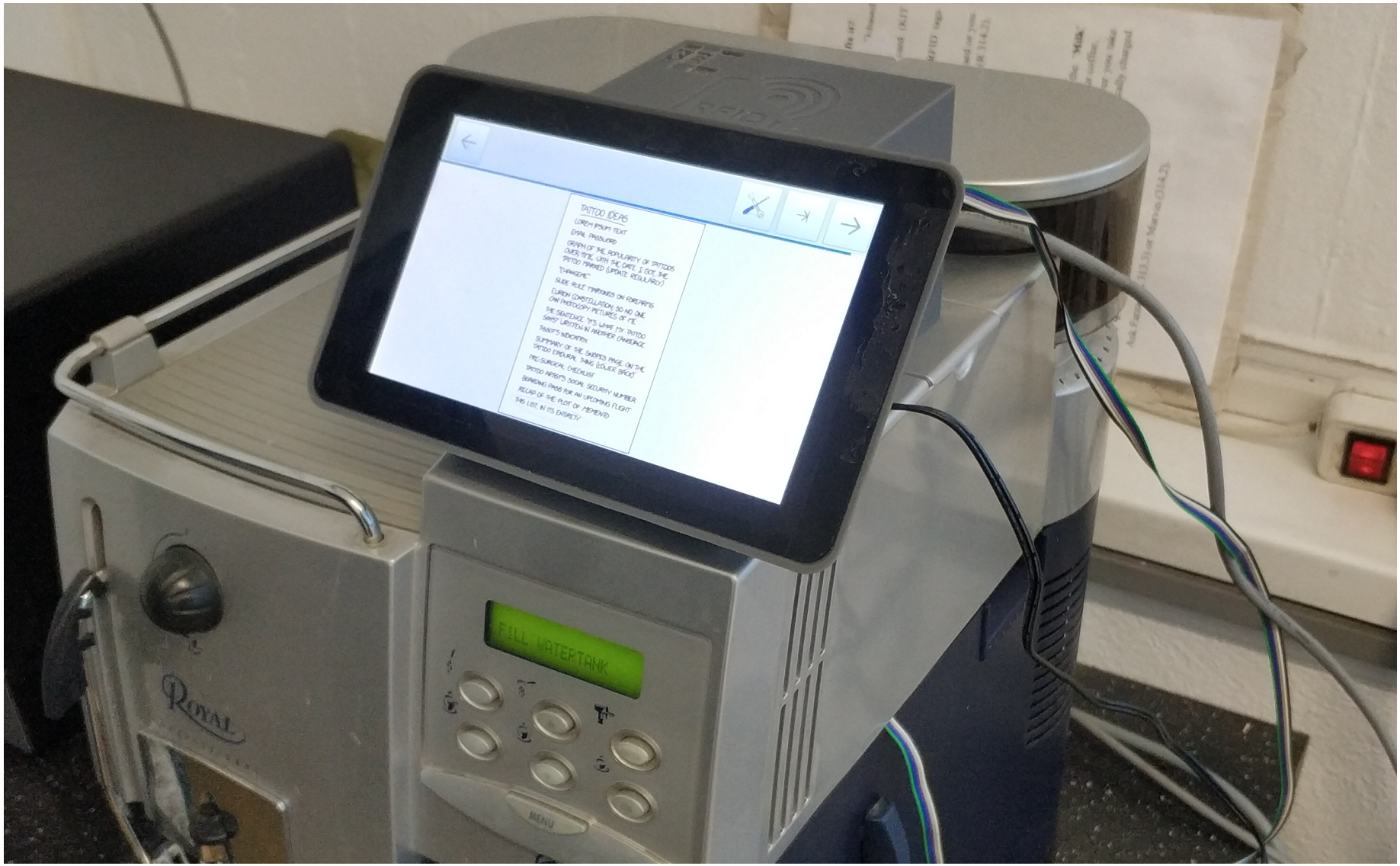
Reduced CPU Usage

Measurements made with `ps` command one hour after boot

	%CPU	python module
Old	98.1	inputGPIO.py (GPIO & buzzer)
	12.4	inputGPIO.py (RFID)
	0.1	main.py (GUI)
	0.0	inputGPIO.py (locking)
New	5.9	pigpiod (GPIO)
	16.3	main.py (RFID)
	0.7	main.py (GUI)
	0.4	main.py (GPIO & locking)
	0.0	main.py (buzzer)

% CPU is the “cpu utilization of the process in "##.#" format. Currently, it is the CPU time used divided by the time the process has been running (cputime/realtime ratio), expressed as a percentage." [ps(1) manpage]

The (Almost) Final System



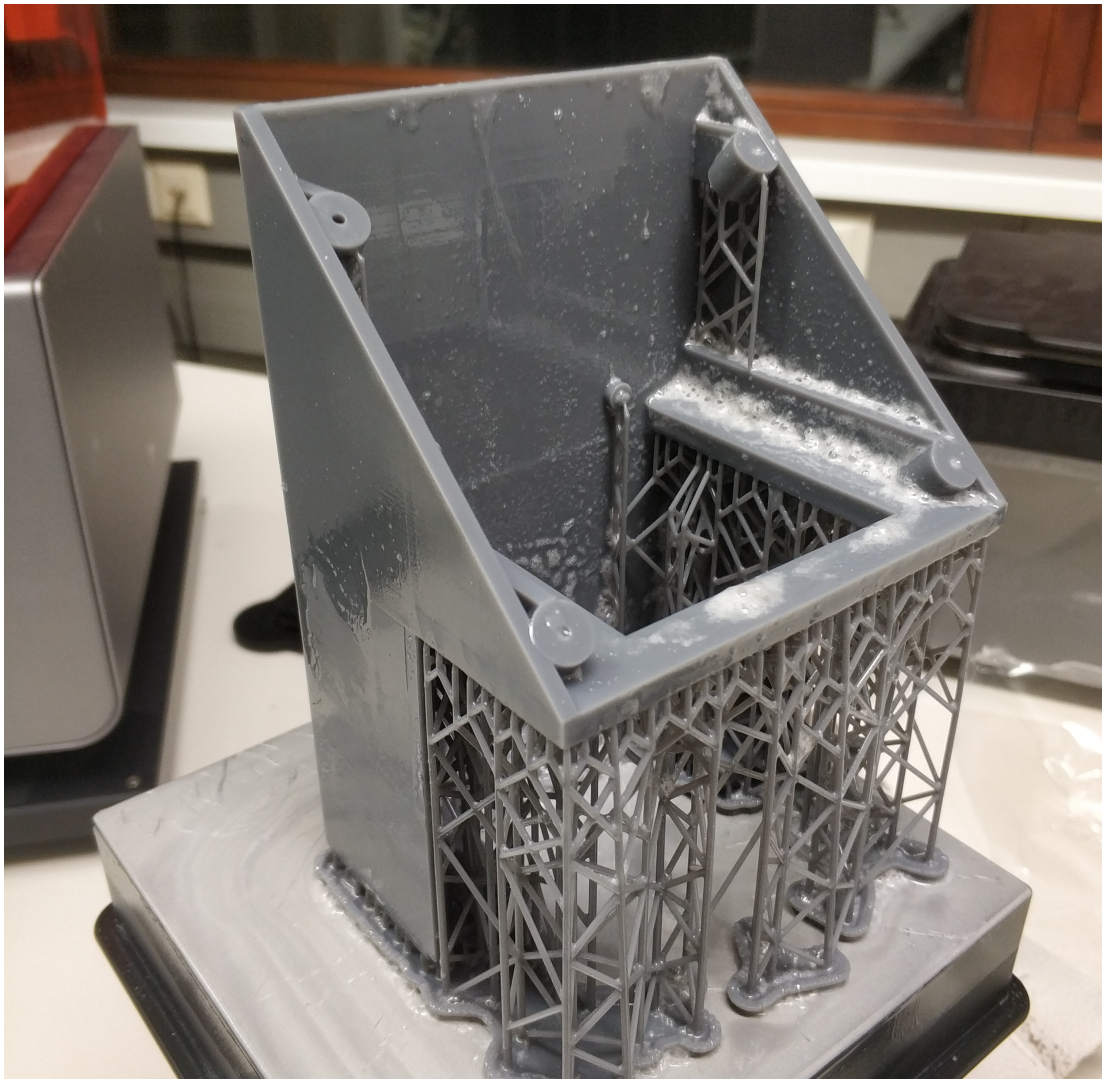
Conclusion

21/21 issues resolved
However: Occasionally coffee brewed without grinder signal
But: workaround in place

- **18/21 issues directly address usability**
- Removed bugs, increased reliability
- System observed to run for 1 month without restart
- Offline mode
- Dispensing limit
- Better handling of warm up

→ **goal of improving usability and reliability accomplished**

3D-Printed Case

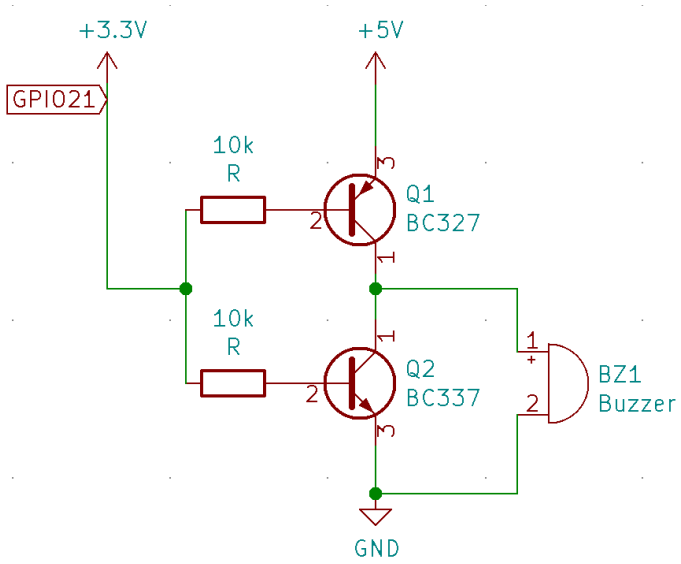


New Buzzer

- Moving piezo christal
- Changing input voltage from low to high -> "clicking" sound
- Frequency Modulation

- 440Hz is A4

Buzzer



Amplifier Circuit