Assignment 9: Power Measurements and Tuning: Report on findings

In addition to the required measurements, we also collected a few interesting data points. If the resistors on the buttons are replaced with 270Ω, the power consumption increases significantly, from 10% extra power draw on the highest frequency setting (72mA to 79.5mA) to 150% extra on the lowest (7.4mA to 18.8mA). The board generally doesn't consume much power even in light sleep(1.43mA), but powering the display adds another 1.17mA on top for a total of 2.6mA.

Deep sleep is definitely the most efficient way, however without WiFi capabilities it is not feasible to use it with our corona architecture. To increase our power efficiency we therefore need to look in different places. Most power is consumed when counting enter/exit events and publishing the count to MQTT, but this is a necessary task. We can however reduce our dynamic frequency to 80Mhz to trade performance for some efficiency. We cannot go lower though, since WiFi requires at least this setting. Places to optimize our code would therefore be the display and using deep sleep at fixed times.

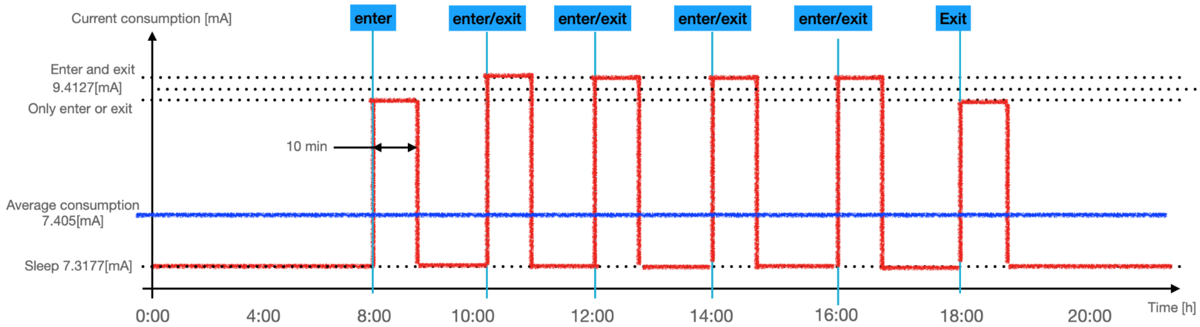
We could only refresh the display when there is an actual change to a displayed value, not periodically. We can also turn off the display altogether after e.g. 2 minutes without a change, and instead add a button or sensor to temporarily turn it on when somebody wants to read it.

Lastly, the seminar room appears to be closed at night and on the weekends, according to the data. Using this fact, we can enable deep sleep during this time and save a lot of energy. This would however remove our ability to see if somebody does in fact enter the room on a weekend (we sadly can't catch a burglar)

Required Result: Math Question

How long is the lifetime of a 600 mAh battery? Assume that every 2 hours between 8:00 and 18:00 there will be a batch of enter/exit events. Assume a clock frequency of 160Mhz and that the automatic light sleep mode is switched on.

Timeline:



Two main phases for power consumption can be distinguished:

SLEEP PHASE:

Configs:
time in [s]: 82800
current consumption sleep mode [mA]: 2.6
current consumption 160 Hz mode [mA]: 55
display refresh current [mA]: 60
display refresh time [ms]: 50
display refresh time interval [s]: 1
WIFI receive current [mA]: 94
WIFI receive time [ms]: 100
WIFI receive time interval [s]: 5
WIFI transmit current [mA]: 200
WIFI transmit time [s]: 0.1
WIFI transmit time interval [s]: 900
Calculations:
Display_refresh time share of total sleep phase:
((82800[s] / 1[s]) * 50 [ms] / 82800[s]) = 0.05
Receive time share of total sleep phase:
((82800[s] / 5[s]) * 100 [ms] / 82800[s]) = 0.02
Transmit time share of total sleep phase:
((82800[s] / 900[s]) * 0.1[s]) / 82800[s]) = 0.0001
Light_sleep_mode time share of total sleep phase: 1 - 0.05 - 0.02 - 0.0001 = 0.9299
Current consumption of sleep phase:
0.05 * 60[mA] + 0.02 * 94[mA] + 0.0001 * 200[mA] + 0.9299 * 2.6[mA] ≈ **7.3[mA]**

BUSY PHASE:

Configs:
time in [ms]: 3600000
current consumption sleep mode [mA]: 2.6
current consumption 160 Hz mode [mA]: 55
display refresh current [mA]: 60
display refresh time [ms]: 50
display refresh time interval [s]: 1
WIFI receive current [mA]: 94
WIFI receive time [ms]: 100
WIFI receive time interval [s]: 5
WIFI transmit current [mA]: 200
WIFI transmit time [ms]: 100
WIFI transmit time interval [s]: 900
Calculations:
Average counts in busy phase: 30 + 60 + 60 + 60 + 60 + 30 = 300
Display_refresh time share of total busy phase:
((3600[s] / 1[s]) * 50 [ms] / 3600[s]) = 0.05
Receive time share of total busy phase:
((3600[s] / 5[s]) * 100 [ms] / 3600[s]) = 0.02
Transmit time share of total busy phase:
((3600[s] / 900[s]) * 100[ms]) / 3600[s]) = 0.0001
Transmit time share of total busy phase:
((3600[s] / 900[s]) + 300[counts]) * 100 [ms] / 3600[s]) = 0.0084
Light_sleep_mode time share of total busy phase: 1 - 0.05 - 0.02 - 0.0084 = 0.9216
Current consumption of busy phase:
0.05 * 60[mA] + 0.02 * 94[mA] + 0.0084 * 200[mA] + 0.9216 * 2.6[mA] + 0.0083 * 55[mA] ≈ **9.4[mA]**

Total average consumption: 23/24 * 7.3[mA] + 1/24 * 9.4[mA] ≈ **7.4[mA]**

<u>Estimated lifetime: 600mAh / 7.4[mA] ≈ 81[h]</u>

This would mean that we need to charge it every 3.5 days. This is highly impractical, since the device is monitoring a room on its own for weeks at a time. However, if we can optimize the code to run for 5 days on one charge, this would enable us to charge it on the weekends. We could get there by optimizing our power usage and using deep sleep.