Energy Management for IoT - Report Lab 03

Samuele Yves Cerini (s256813)

February 27, 2020

Contents

1 Introduction

The goal of this third and last laboratory is to model and simulate an IoT system by carefully evaluating its capabilities from a power consumption point of view. The final goal is to leverage the model (made using *MATLAB/Simulink*) of the system and reach a satisfying level of power consumption, enough to reduce as much as possible the human intervention required to substitute the on-board battery. The system embeds four different ambient sensors, some memory and a control unit, a wireless transmission unit (using the *ZigBee* protocol), a battery and a photovoltaic panel that will act as an energy scavenger, allowing us to recharge the battery and prolonging the lifespan of the overall system.

In this report we will not discuss (as requested) the construction of the model or the configuration of the various components that are part of the system, but we will discuss directly the results obtained and the modifications done to the system itself in order to prolonge the system's battery life.

This report is hence divided into 2 main parts:

- Workload definition to reduce the power consumption
- Modifications done to the system itself to increase its battery life

2 Manipulating the system's workload

The goal of this first part of the laboratory is purely optimization-oriented: given the original model of the system, our goal is to maximize its battery life. By keeping the system as it is provided, without modifications, the only parameters we can modify are the different workloads submitted to the system, i.e. the load scheduling. The system, as mentioned before, is provided with four different sensors, a control unit and a transmission module. In order to optimize the battery life, we can modify how the different loads are activated, by choosing a good scheduling that minimizes the energy consumption.

Load	Active Time (s)	Active Current (mA)	Sleep Current (mA)
Air Quality Sensor	30	48.2	0.002
Methane Sensor	30	18	0.002
Temperature Sensor	6	3	0.002
Microphone Sensor	12	0.15	0.002
ZigBee Transmission	24	0.1	0.001
Memory + Control	6	13	0.002

Table 1: Summary of the data linked to the loads attached to the system. The active time is fixed for each load and it depends on the requirements of the component.

As a side note, we would like to inform the reader that all the following results have been taken with an activation period equal to 2 minutes. Different activation periods were initially proposed (like a period time of 10 minutes) but were finally withdrawn from the requests of the lab since such periods were sufficient to let the system run for an entire month, hence limiting our possibilities of analysis. Of course, some tests have been made with time periods equal to 10, 5 and 3 minutes, to observe the behavior of the system out of my curiosity, although the results have not been included in this report.

2.1 Sensors and Actuators characteristics

In table (1) we reported the overall data available for every load attached to the system, retrieved directly from the related datasheets. Understanding what are the more demanding loads is crucial if we want to define a correct and balanced scheduling.

From the theory of DPM (Dynamic Power Management) we know for example that a "parallel" workload scheduling is not beneficial compared to "sequential" workload scheduling that spreads the load times along an entire period. In other words, it has been demonstrated multiple times that time limited bursts of high energy requests followed by long idle times are not beneficial as we may think.

2.2 Sequential Workload

The first workload scheduling used is the one provided with the lab requirements: each one of the four sensors is activated sequentially, one after the other. The only constraints of the entire scheduling implies that we cannot process the data required before the completion of all the four sensors. At the same time, we cannot send the data (hence activating the transmitting module) prior the completion of the sensors load and the computation of the related data. Thus, given these constraints, it is clear that the scheduling will have the last two loads related to the computation and the transmission of the retrieved data.

As we mentioned before in the previous section, we expect this load to be quite good in terms of power consumption, due to its intrinsic wide scheduling (no small bursts of high energy requests are present).

The provided scheduling is composed as follows:

1. Air Quality sensor;
2. Methane sensor;
3. Temperature sensor;
4. Microphone sensor;
5. Memory and Control;
6. Transmission module;

With the provided model and components, considering a period of activation equal to 2 minutes, the simulation of such workload scheduling resulted into a battery life of 588361 seconds, that corresponds to 6 days, 19 hours, 26 minutes. As we will see in the following sections, this load guaranteed the best result in terms of power consumption.

The following image (1) shows the workload scheduling as feeded to the model and simulated by *Simulink*. In the bottom part of the graph we can see the load activation scheduling: we can appreciate the fact that the sequential approach spreads the load activation on the quasi entire period of activation (which corresponds to 2 minutes, i.e. 120 seconds). The top part of the graph shows, on the other hand, the current and the power consumption as required by the different load activations. From the graph, we can confirm that the Air Quality sensors has, for example, the highest current consumption among all the sensors ($48.2mA$), which results in a power consumption of $48.2mA \cdot 3.3V = 160mW$.

In picture (11) we reported the State of Charge and the voltage of the battery integrated in the system. As we can see, the battery is constantly discharging and the system is inevitably going to shut down itself. In some regions (that correspond to the regions where the solar panel harvests energy) the State of Charge stabilizes to a constant value or increases a little bit (still not significantly to avoid the complete discharge of the battery).

In picture (3) we can observe the trend of the current coming from the battery (`Battery Current`). This parameter if negative, indicates that the battery is being recharged. On the other hand, if the current has a positive sign, the current is absorbed from the battery to keep alive the entire system, hence discharging the battery. From this graph it is clear that more and more current is absorbed from the battery and that the graph area is greatly unbalanced towards the battery draining effect (a good system will have a graph more balanced that does not deviates towards more and more positive values, with an area beneath the graph which is equally balanced between negative zones and positive ones). From this same graph we can also observe the current provided by the photovoltaic module (`PV Current`) and the same current after the DCDC converters (`PV DCDC Current`) (that follow the module). We can observe the current attenuation due to the presence of the converters: adopting more expensive converters with higher efficiency can allow to reduce this attenuation and to obtain curves that are more and more close to themselves.

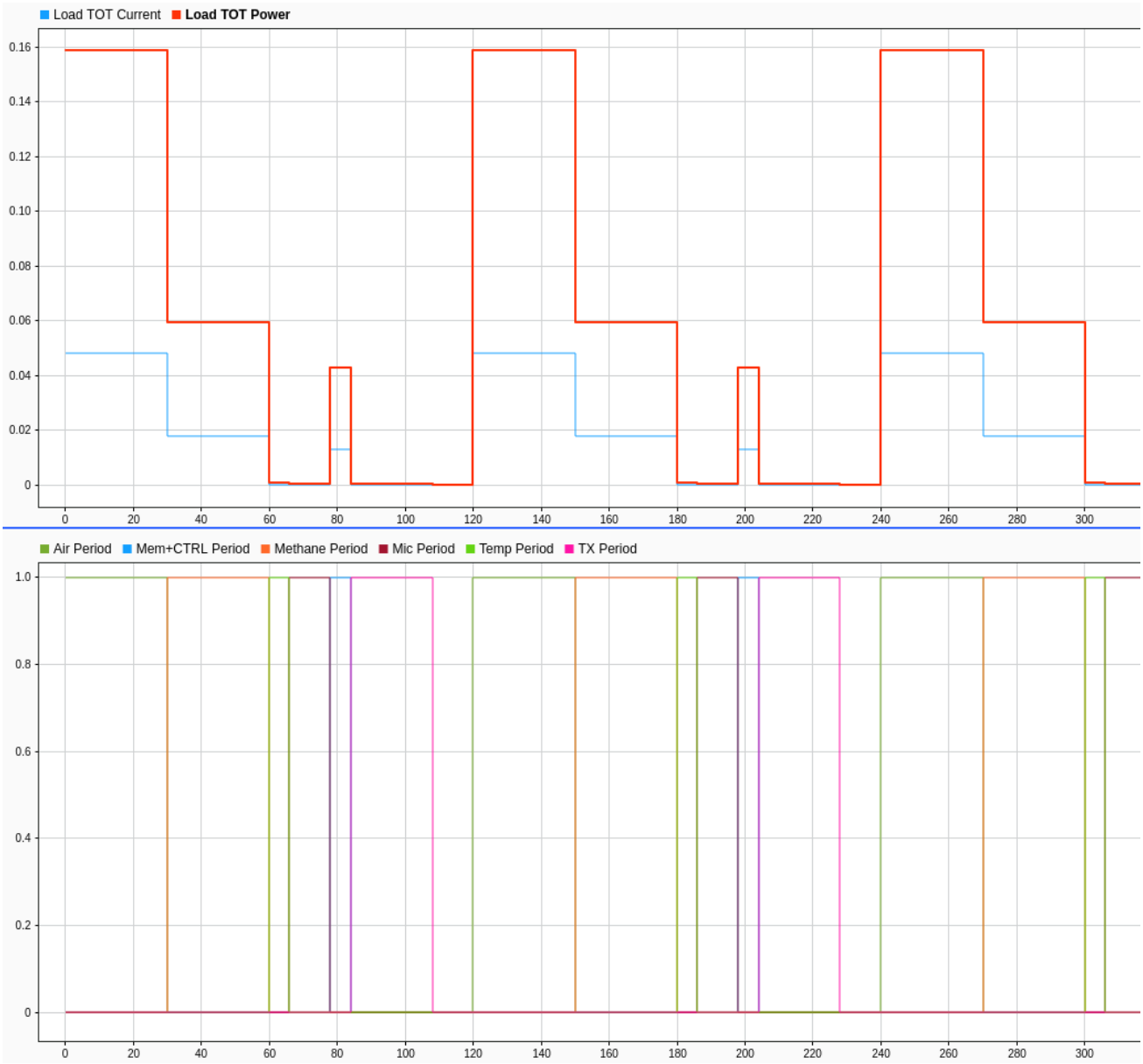

Figure 1: [Sequential Activation] Graph representing, in the bottom part, the sequential activation of all loads embedded in the system. In the top part, we can see the corresponding current and power consumption.

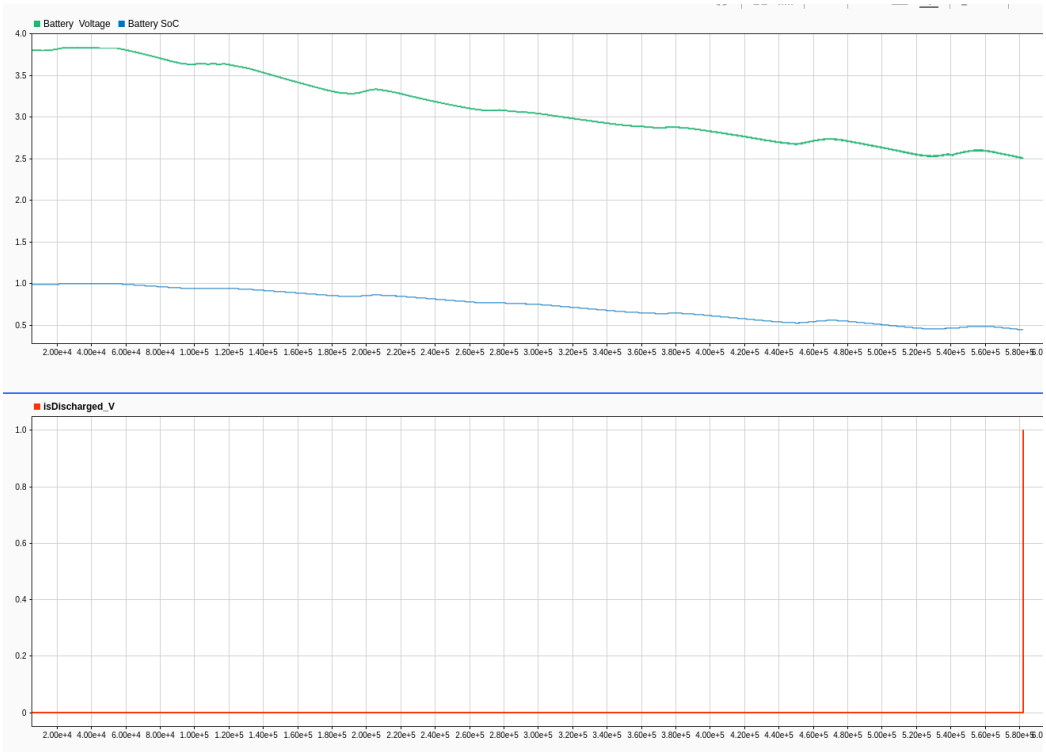

Figure 2: [Sequential Activation] Graphical representation of the State of Charge and the voltage of the battery. As we can see the battery is constantly discharged until the voltage reaches a threshold at which the battery is considered completely discharged (also indicated by the `isDischarged_V` curve).

2.3 Parallel Workload

The second workload scheduling tested is the one that allows the parallel execution of loads, hence realizing a scheduling characterized by short bursts of current consumption and long idle times for each period (which remains also in this case set to 2 minutes). As mentioned before, we expected this workload scheduling to be the one less efficient in terms of energy consumption, leading to the poorest result in terms of battery life of the system.

This hypothesis revealed to be true since we obtained a simulation that lasted up to 586229 seconds, equivalent to 6 days, 18 hours, 50 minutes and 29 seconds (and so approximately 36 minutes less than the Serial Scheduling tested in the previous section).

From the first image (4) we can appreciate the load active time disposition, which follows as expected a parallel situation. On the top part of the graph we can, for example, observe that most of the power consumption happens in a short burst that lasts less than 40 seconds (even though the data transmission lasts up to 60 seconds, however its current consumption is minimal with respect to the one that happens before the 40 second threshold).

From the second image (5) we appreciate once again the State of Charge and the voltage of the battery. The bottom part indicates, once again, that the system is shut down due to the reaching of the lower threshold for the battery voltage: below that threshold the battery is considered to be completely discharged.

Finally, the third figure (4) shows once again the current flowing to and from the battery, demonstrating the imbalance of the system once again.

2.4 Custom Solutions

I implemented two different workload schedulings as an attempt to reach higher lifetime of the system. As a general approach, I modified the provided Serial Scheduling, considered as a good starting point. The main idea is to better distribute the current consumption of the Serial Scheduling avoiding sharp peaks in energy consumption and having an overall "smoother" shape, following the shape of a gaussian distribution. Both schedulings simulated allowed to obtain very similar results:

- Custom Scheduling #1: 588360 seconds, equivalent to 6 days, 19 hours and 26 minutes (equivalent to the Serial Activation scheduling).
- Custom Scheduling #2: 588291 seconds, equivalent to 6 days, 19 hours, 24 minutes and 51 seconds.

Figure 3: [Sequential Activation] Graph representing the battery current and the current provided by the photovoltaic panel (in the bottom section). In the top section we can appreciate the power produced by the solar panel.

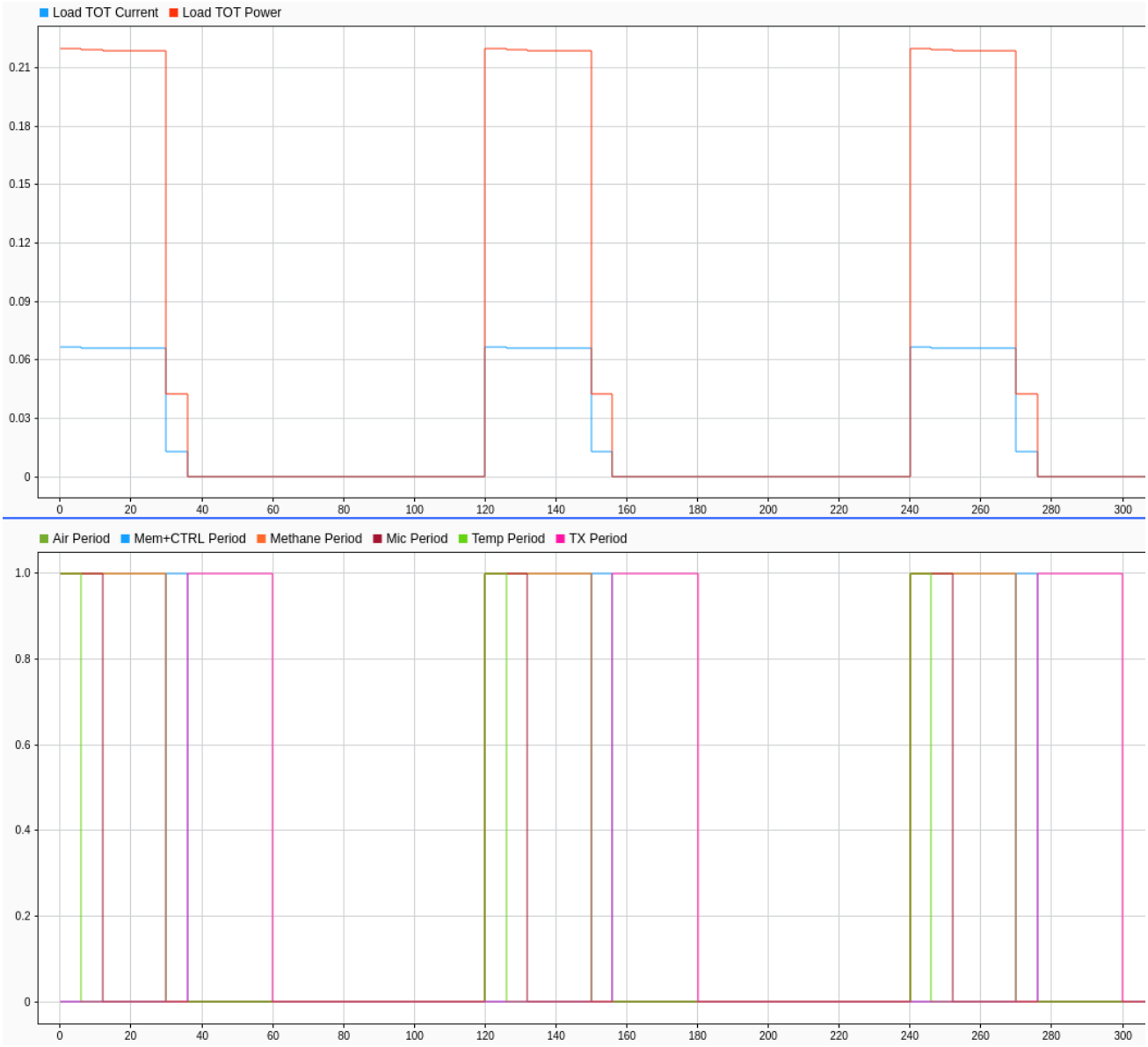

Figure 4: [Parallel Activation] Graph representing, in the bottom part, the sequential activation of all loads embedded in the system. In the top part, we can see the corresponding current and power consumption.

Figure 5: [Parallel Activation] Graphical representation of the State of Charge and the voltage of the battery. As we can see the battery is constantly discharged until the voltage reaches a threshold at which the battery is considered completely discharged (also indicated by the `isDischarged_V` curve).

In the following image, we report the custom scheduling organization #2 and the current consumption due to the load progression. If compared to the workload related to the Serial Activation, we can appreciate how high peaks of current consumption have been "smoothed" thanks to the different position of the loads activations. This allowed to obtain less noticeable current consumption swings.

I decided to not report other graphs like the ones previously showed since the differences are not so marked and the two graphs will be quite comparable.

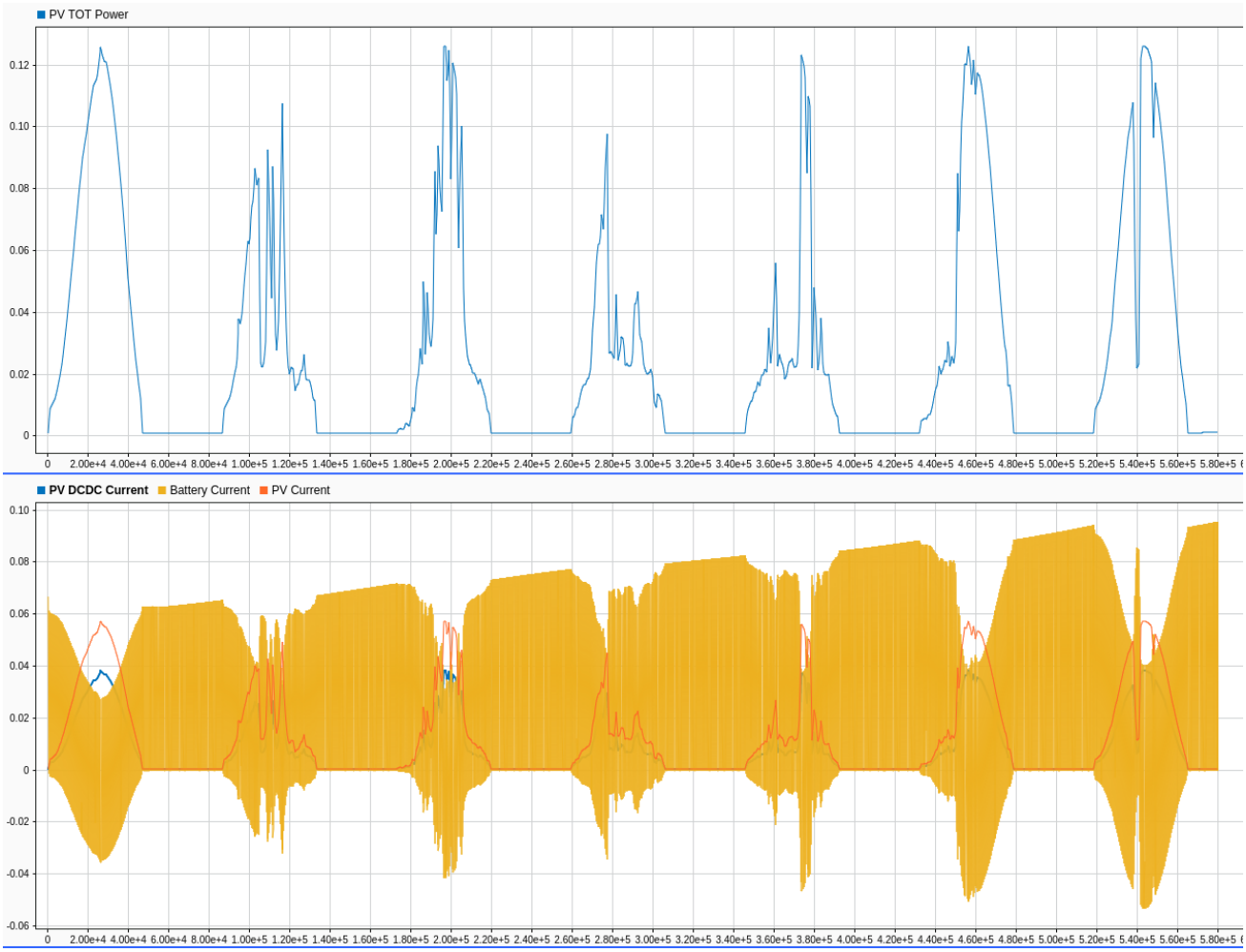

Figure 6: [Parallel Activation] Graph representing the battery current and the current provided by the photovoltaic panel (in the bottom section). In the top section we can appreciate the power produced by the solar panel.

In the following piece of code I report the construction of the two custom workloads:

```
% Custom #1: 588360 sec
air_delay = 0;                        % first to execute
methane_delay = air_time;             % second to execute
temp_delay = air_time+mic_time;       % third to execute
mic_delay = air_time;                 % fourth to execute
mc_delay = air_time + methane_time; % fifth to execute
transmit_delay = mc_delay + mc_time;% sixth to execute

% Custom #2: 588291 sec
methane_delay = 0;                                  % first to execute
air_delay = methane_time;                           % second to execute
temp_delay = methane_time-mic_time-temp_time;    % third to execute
mic_delay = methane_time-mic_time;                  % fourth to execute
mc_delay = air_time + air_time;                     % fifth to execute
transmit_delay = mc_delay + mc_time;                % sixth to execute
```

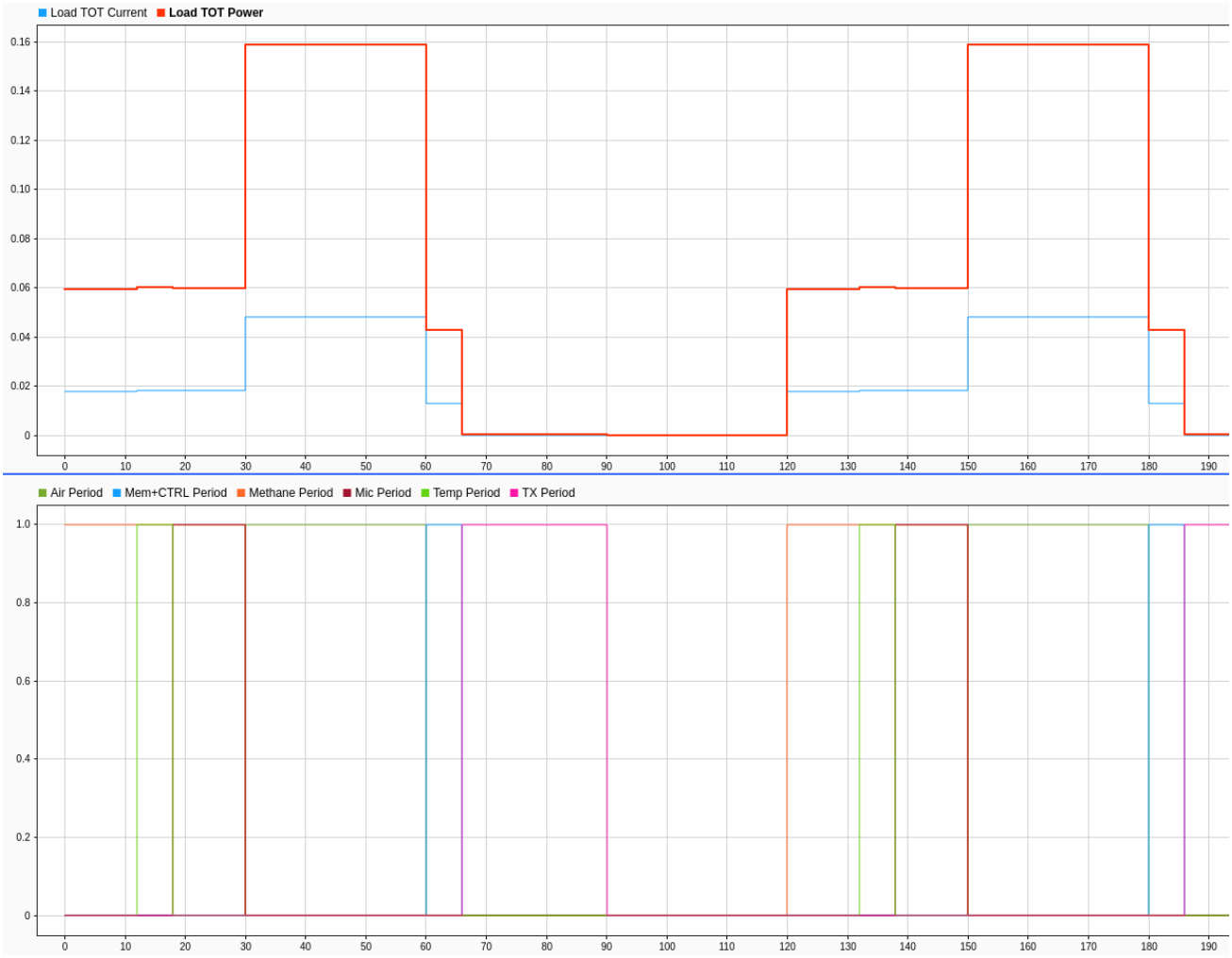

Figure 7: [Custom Activation #2] Graph representing, in the bottom part, the sequential activation of all loads embedded in the system. In the top part, we can see the corresponding current and power consumption. Please, notice how the total load current consumption has been "smoothed", allowing to have current swings less pronounced than the case obtained with the Sequential Activation (even more if we compare it to the Parallel Activation).

2.4.1 Comments

Although the custom solutions seems to not provide evident benefices in terms of overall battery life of the system, we must take in mind the overall result "volatility" of the system and its dependency to the model construction and components definition. These custom workloads have in fact have been simulated twice: the first time the coefficients of the converters were very similar to the actual ones but enough different to draw a quite different scenario in our system: with these latest coefficients that I've tried, all the workloads gained about 2 hours of lifetime. At the same time, these last coefficients resulted in a larger lifetime for the Serial Activation, overriding the previous best results I obtained with my custom workloads. That's why I was not able to apparently obtain a custom workload better than the one provided.

Finally, I consider my workloads to be "less stressful" from a battery point of view with respect to the Serial Activation one: the reduction of the overall current swing probably allows to obtain less wear on the battery total life, finally allowing it to be recharged more times before its complete depletion.

3 System modifications and additions

This second part of the laboratory requests to modify the system in order to reach an even higher battery lifetime, following these guidelines that refer to a good IoT device design:

- an IoT device should be autonomous as much as possible;
- an IoT device should not require frequent battery changes;

Of course, each modification done to the system comes to a cost (in both the economic term and in the total area/weight domain). Hence, some tradeoff evaluation must be taken into account. Given the requests of the lab, we as designers have the following possibilities:

- Activate the heaviest sensor less frequently (i.e with a longer period);
- Increase the photovoltaic power production (one PV panel costs $5.50);
- Increase the battery capacity (one battery costs $4.99);
- Other ideas;

Also, we have to take in mind the cost that each addition brings: the budget dedicated to additions is equal to $11.00.

The following considerations and hypothesis have been taken into consideration and lead to the results I will illustrate in the following sections. The slower pace activation of the heaviest sensor in terms of power consumption (in this case, the Air Quality sensor) heavily depends on the application domain: some applications may not allow such solution to be taken into consideration since such a system may simply fall out of scope and become useless. In our case I decided to test two solutions: one with an activation period for the Air Quality sensor of 2 minutes (like before) and one with an activation period of 6 minutes. Considering the possibility to add a photovoltaic panel we must take in mind the overall external environment conditions (are we installing our IoT device in a position with very good irradiance or not? Will the addition of our panel be worth?). Also, some constraints like the area required and the transportability of the device may arise, depending once again on the application constraints. Finally, considering the addition of a battery pack, once again we must face the issues related to the final weight of the IoT device. Also, if the power consumption is high, placing a secondary battery in parallel to the original one may only double (in a best case scenario) the overall lifetime of the device, which however may still be not sufficient.

3.1 Modifications done to the system

My choices were mainly driven by the considerations I made by observing the results obtained with the previous tests. I also assumed an activation period of 2 minutes as fixed and mandatory (since the system was proven behaving really well with a period of 10 minutes, being of no use for educational purposes). Finally, by observing the previous graphs like (3 and 6), I was led to the conclusion that the addition of a secondary battery in parallel (increasing the overall capacity) would not have been enough to keep the system alive, especially considering the rate of discharge (current drained from the battery) proven in those graphs. Hence, I decided to add only a secondary photovoltaic module in parallel to the system, along with a secondary DCDC converter (so as to avoid a parameter drifting and a complete re-design of the entire PVmodule+DCDCconverter subsystem, which already proven to be on-point in terms of overall efficiency). Thus, I assumed that the provided photovoltaic panel includes a DCDC converter, as we can see in some solutions available on the market. This addition required also the addition of a second "port" in the *DCBus* component, as we can see in the following figures (9 and 8).

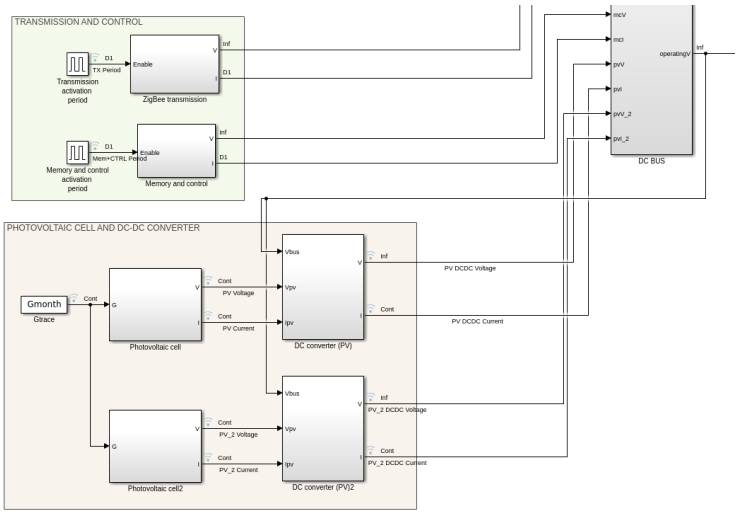

Figure 8: In this figure we can appreciate the modifications made to the overall module, especially the Photovoltaic Cell and the DCDC converters. Both PV panels are affected by the same irradiance (Gmonth).

3.2 Results and conclusions

I tested the system using the `Gmonths` data table (containing an irradiance trace up to 3 months). The system simulation stopped after 4229041 seconds, equal to 48 days, 22 hours and 44 minutes.

These results proved the correctness of the original assumption, proving the efficacy of the presence of an additional PV panel in order to reduce the overall energy directly drained from the battery, hence heavily increasing the lifetime of the IoT device. Please refer to figures (11 and 10) for additional considerations.

Finally, the second modification to the system, the one that includes the secondary PV panel and executes the Air Quality load every 6 minutes (activation period equal to 6 minutes) was able to simulate through the entire 3 months-long simulation. Sadly, I could not be able to include any figure or proof of this solution in this report since *Matlab/Simulink* kept crashing once the simulation terminated, not allowing me to open the inspector.

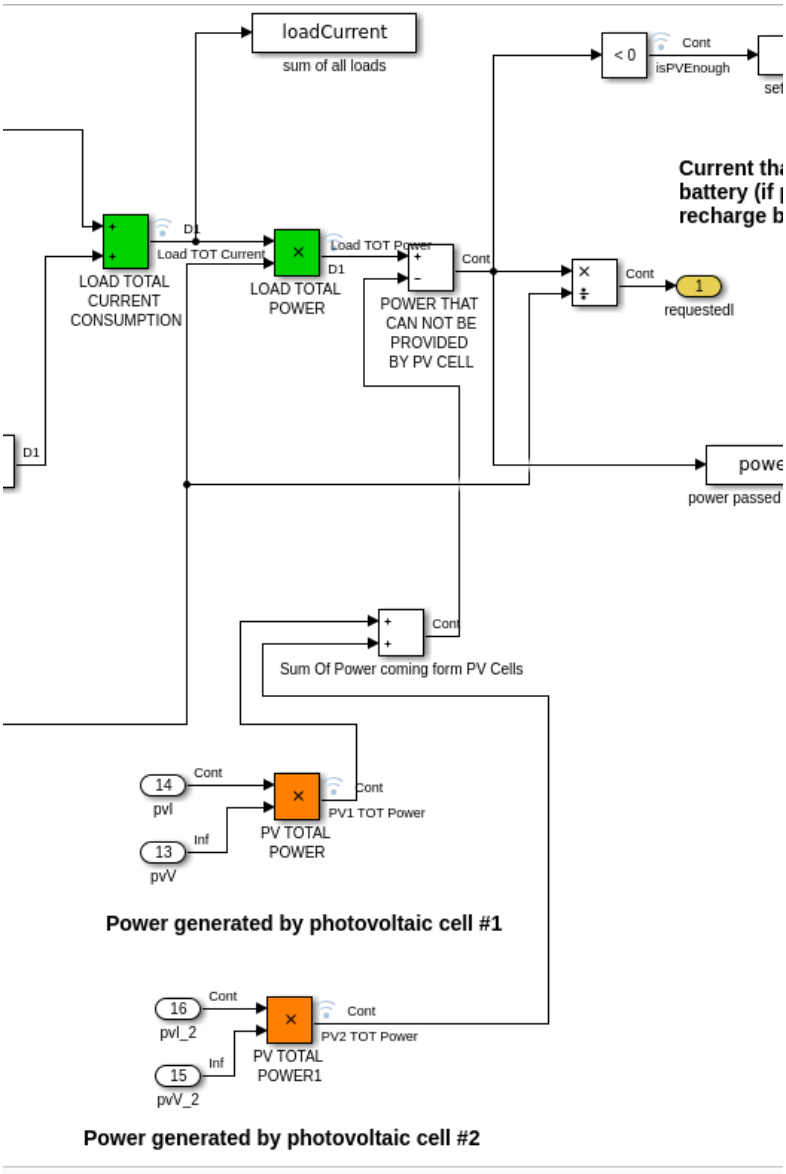

Figure 9: This figure shows the modifications made to the *DCBus* component of the overall model: a secondary "port" as been added to connect the *DCBus* to the current and voltage values coming from the second couple of PV panel and DCDC converters. The power components provided by the two subsystems are then summed together.

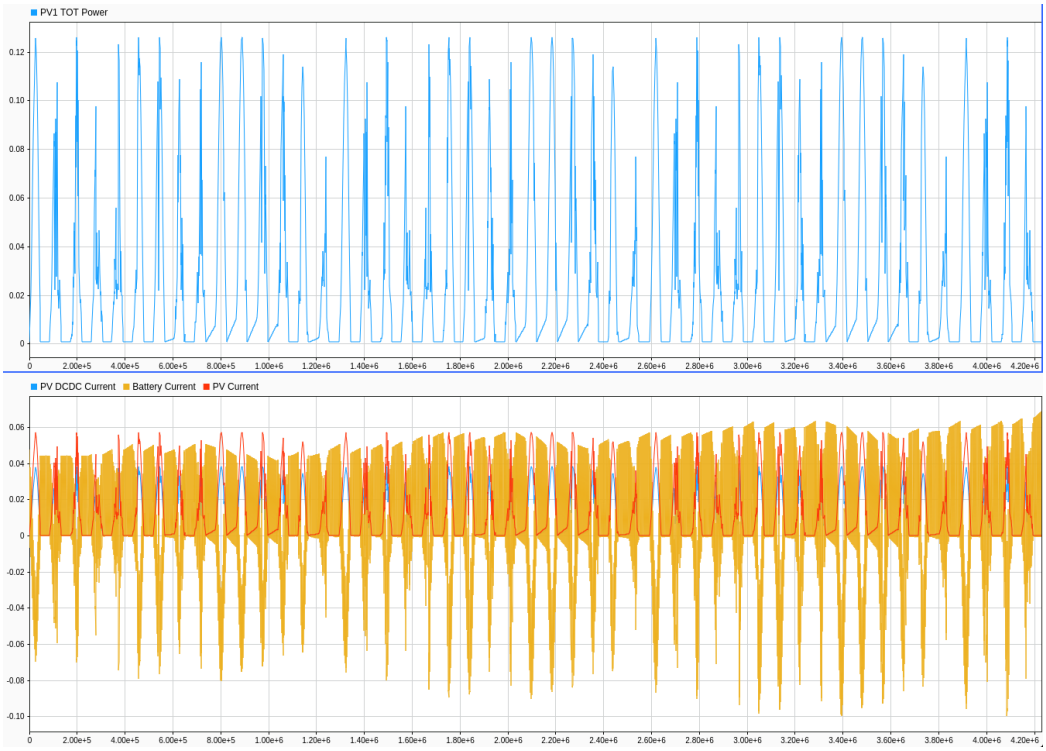

Figure 10: This figure shows, for the entire simulation length, the values associated to the current drained from the battery (`Battery Current`), the power produced by one PV panel (the same for both PV panels since they share the same irradiance values) and the current provided by a single PV panel (`PV Current` and `PV DCDC Current`) (since we have now 2 panels, the final current is double the value represented in the graph).

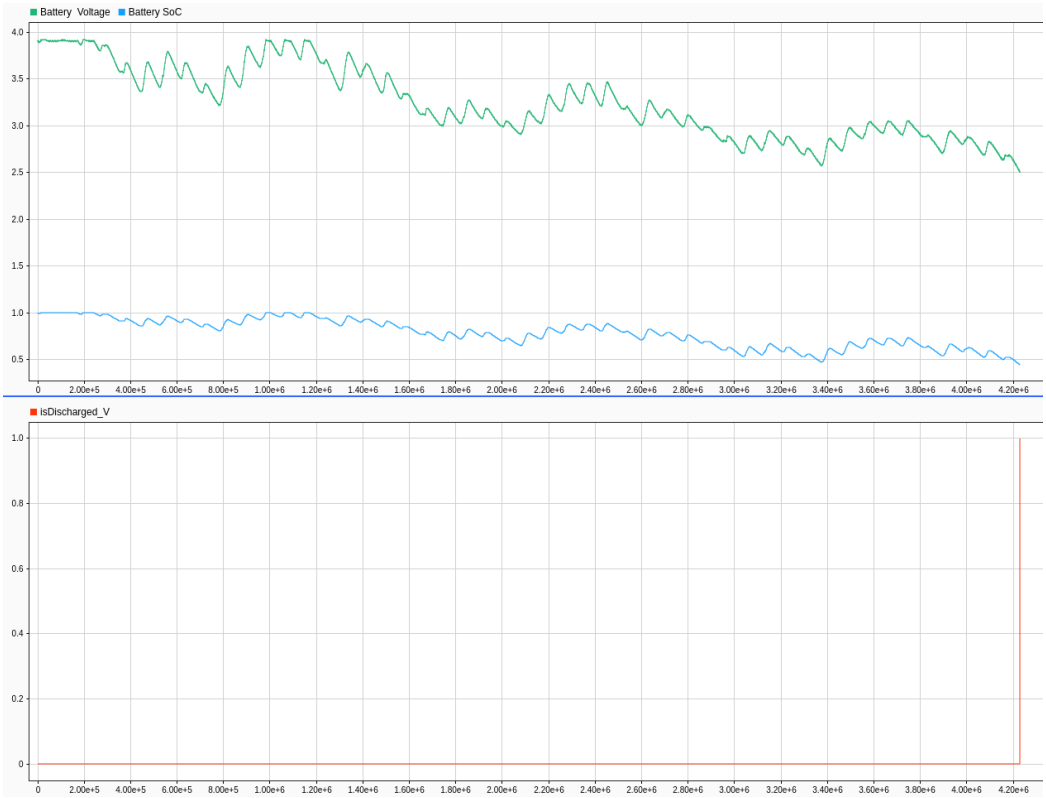

Figure 11: This figure shows, for the entire simulation length, the battery State of Charge and Voltage. As we can see, the addition of a secondary PV panel allows to recharge more significantly the battery and significantly increases the lifetime of the system. However, once again, the system is considered to be "dead" not because of the low value of the State of Charge, but due to the low value of the overall voltage provided by the battery, as clearly visible in the bottom part of the graph.