Acta Astronautica 147 (2018) 141–151

Contents lists available at ScienceDirect

Acta Astronautica

journal homepage: www.elsevier.com/locate/actaastro

Energy-driven scheduling algorithm for nanosatellite energy harvesting maximization

L.K. Slongo[a,*], S.V. Martínez[a], B.V.B. Eiterer[a], T.G. Pereira[a], E.A. Bezerra[a], K.V. Paiva[b]

[a] Communications and Embedded Systems Lab (LCS), Department of Electrical and Electronic Engineering (EEL), Technological Center (CTC), Federal University of Santa Catarina (UFSC), Trindade, Florianpolis, Santa Catarina, 88040-900, Brazil

[b] Thermal Fluid Flow Group, Aerospace Engineering, Department of Mobility Engineering, Technological Center of Joinville (CTJ), Federal University of Santa Catarina (UFSC), Joinville, 89218-035, Brazil

ARTICLE INFO

Keywords:
Energy-driven scheduling algorithm
CubeSat
Electrical power system
Solar energy harvesting
Maximum power point tracking

ABSTRACT

The number of tasks that a satellite may execute in orbit is strongly related to the amount of energy its Electrical Power System (EPS) is able to harvest and to store. The manner the stored energy is distributed within the satellite has also a great impact on the CubeSat's overall efficiency. Most CubeSat's EPS do not prioritize energy constraints in their formulation. Unlike that, this work proposes an innovative energy-driven scheduling algorithm based on energy harvesting maximization policy. The energy harvesting circuit is mathematically modeled and the solar panel I-V curves are presented for different temperature and irradiance levels. Considering the models and simulations, the scheduling algorithm is designed to keep solar panels working close to their maximum power point by triggering tasks in the appropriate form. Tasks execution affects battery voltage, which is coupled to the solar panels through a protection circuit. A software based Perturb and Observe strategy allows defining the tasks to be triggered. The scheduling algorithm is tested in FloripaSat, which is an 1U CubeSat. A test apparatus is proposed to emulate solar irradiance variation, considering the satellite movement around the Earth. Tests have been conducted to show that the scheduling algorithm improves the CubeSat energy harvesting capability by 4.48% in a three orbit experiment and up to 8.46% in a single orbit cycle in comparison with the CubeSat operating without the scheduling algorithm.

1. Introduction

Nanosatellites have become an affordable opportunity to reach the space. They are small satellites with total mass ranging from 1 to 10 kg with all the needed subsystems to satisfy a common satellite mission (including payloads). Through nanosatellites, universities may allow their students to work on real space application projects. Even small and medium size companies may have access to space technologies that until 20 years ago were mostly restricted to governmental space agencies. Presently, nanosatellites' launching "low price" and short development time attracts space enthusiast. This growth in interest in small satellites was empowered by the CubeSat standard definition in 1999. The Stanford University and California Polytechnic State University (Cal Poly) proposed a modular 10 cm × 10 cm x 11.35 cm (1U) cubic shaped satellite intended for Low Earth Orbit (LEO) and designed mostly with commercial off-the-shelf (COTS) components. Since then, many other universities and companies around the world have been working on nanosatellites' development, testing, launching and tracking [1,2].

Motivated by the opportunity of allowing its students to work in a full space mission, the Federal University of Santa Catarina (UFSC) has started its own 1U-CubeSat development - The FloripaSat. The project's main goal is to empower undergraduate students which have been organized in the following teams: Electrical Power System (EPS), On-Board Data Handling (OBDH), Telemetry, Tracking and Command (TT&C), Attitude Determination and Control System (ADCS), Ground Station (GS), Verification and Validation (VV), Thermal Control and Structure (TCS) and Payloads (PL). The work presented in this paper has been developed in the FloripaSat EPS context.

A satellite electrical power system has three main functions: energy harvesting, energy storage and energy distribution. The EPS is a printed circuit board (PCB) which interacts with power sources (solar panels,

* Corresponding author.
E-mail addresses: slongo@eel.ufsc.br (L.K. Slongo), sara.vega@posgrad.ufsc.br (S.V. Martínez), bruno.eiterer@grad.ufsc.br (B.V.B. Eiterer), tulio.gomes.pereira@grad.ufsc.br (T.G. Pereira), Eduardo.Bezerra@eel.ufsc.br (E.A. Bezerra), kleber.paiva@ufsc.br (K.V. Paiva).
URL: http://www.lcs.ufsc.br/ (L.K. Slongo).

https://doi.org/10.1016/j.actaastro.2018.03.052
Received 4 May 2017; Received in revised form 31 October 2017; Accepted 28 March 2018
Available online 31 March 2018
0094-5765/

thermoelectric generators, etc.), with storage units (batteries, supercapacitors, etc.) and with other satellite's subsystems (OBDH, TT&C, payload, etc.).

An ideal EPS should maximize energy extraction and manage the energy distribution to other satellite's subsystems in the most efficient manner. Mostly, these requirements conflict among each other or with other satellite subsystem's requirements. This work presents an elegant solution to maximize energy extraction only by controlling the satellite's tasks execution. Nanosatellites may be considered low power devices when compared to satellites with more than 100 kg. Therefore, the energy harvesting maximization must be addressed differently. Also, the applications for LEO nanosatellites drastically differs from the Medium Earth Orbit (MEO) and Geostationary Orbit (GEO) satellites' applications. LEO nanosatellites may not be continuously receiving solar energy, which makes the energy management highly dependent on orbit inclination. Also, nanosatellites may decay only few months after their launch, reducing mission time, which also impacts on the energy distribution strategy.

The satellite's tasks shall be somehow organized in order to accomplish the mission requirements. Since the tasks to be performed may have different priorities, execution time, resources, etc., a satellite task scheduling algorithm may be a key element to achieve a successful satellite mission. Once the scheduling algorithm may define which (and how) the satellite's tasks are going to be executed, there shall be a relation between the algorithm and the EPS, after all, tasks execution demands energy.

The satellite scheduling problem is not new. It has been formulated with a variety of perspectives, with numerous proposed solutions (Section 2). Although there are distinct manners for defining and solving problem [3–5], the goal is mostly the same: to optimize tasks execution from some perspective (maximize communication quality [6], minimize system response time [7], etc.).

As described in Section 2, most of recent satellite scheduling algorithms are not designed for nanosatellites. Besides this, none of them aims energy harvesting maximization. Energy aware task schedulers have been widely discussed for wireless sensor networks (WSN) applications [8,9]. Although some of these works focus on reducing energy consumption to extend the nodes lifetime, the problem is totally different. Most wireless sensor networks have no energy input. Also, the periodicity (both in tasks execution as in power input) imposed by the orbital motion is not verified in most WSNs. Some authors, even when considering energy harvesting embedded systems, propose algorithms based on dynamic voltage and frequency scaling technique [10]. This approach is restricted to reducing the processor power consumption only. Thus, this work tries to solve the satellite scheduling problem with a different approach for an emerging class of low power satellites.

2. Related work

The task scheduling problem in satellites is not new, referring to the late 50s and early 60s, during the Space Race, where military artificial satellites started being developed and launched. At that time, the main concern was to maximize communication time. The system factors pertinent to the scheduling problem used to be classified into three categories: satellite availability, communication requirements and quality of communication. Linear programming approach has been one of the solutions to solve a set of mathematical equations in a maximization problem. Due to the computational limitation at that time, dynamic scheduling was considered an overhead [11].

With the increasing number of launched satellites and the development of new channel access methods, the scheduling problem has become more sophisticated. For instance, scheduling algorithms have been applied to satellite systems communicating through time division multiple access (TDMA) to a channel. In this case, the proposed scheduling algorithm goal was to avoid or to reduce message conflict from ground stations when occupying a time slot. Also, the idea was to minimize the assignment procedure to shorten assignment time delay [12]. No power constraint is mentioned in this solution.

Later, scheduling techniques have been applied to Earth Observing Satellites (EOS). Some of these works considered energy constraints in their scheduling algorithms. The Landsat 7 from National Aeronautics and Space Administration (NASA), for instance, implemented the so-called duty cycle constraint. A sensor should be limited to its operating time for a given period [13]. For the Landsat 7 a sensor should not be used for more than:

1. 34 min in any 100 min period,
2. 52 min in any 200 min period, or
3. 131 min in any 600 min period.

Since there is a correlation between the time the sensor is turned on and its power consumption, this can be considered an energy constraint. However, none of the evaluated algorithms solve the scheduling problem to reduce power consumption but to maximize the number of collected images from Earth.

An innovative work has considered fault-tolerant and real-time aspects to solve the task scheduling problem for multiple observation satellites [14]. In this innovative approach, the authors adopt the replication concept to ensure that a designated task is going to be executed. For this, they assume that a task primary copy is successfully allocated only if its corresponding task backup copy can be scheduled in another satellite. Otherwise, the primary copy shall be canceled. Even if one of the satellites fails in executing the task, the other one is able to execute it. Although the scheduling problem is rigorously well defined through a set of equations and assumptions that ensure the good performance of the algorithm, this work also does not mention power constraints or energy efficiency optimization.

Some recent works have been developed on solving the issue of ground station-satellite communication on multi-satellite missions. This problem also may be solved using scheduling algorithms. Recent ideas have emerged as applying mutation concepts of genetic algorithms to meet computation time and success rate mission requirements on satellite communication. Hybrid Dynamic Mutation has demonstrated outstanding performance in terms of speed and reliability when compared with other mutation strategies [15]. Although both algorithm have proved to be efficient they are focused on the ground station side. They do not consider the satellite tasks management nor its energy consumption impact on mission accomplishment.

A dynamic scheduling approach is proposed by Wang *et al.* for emergency tasks on distributed imaging satellites (a satellite constellation for imaging proposes). The authors defined a multi-objective mathematical programming model that contains five objects: tasks, resources, available opportunities, operational constraints and objectives [5]. Energy consumption minimization is classified as one of the scheduling objectives. The authors present a so-called merging tasks technique, which allows tasks being executed simultaneously, reducing energy consumption in comparison with other algorithms. However, authors state that the scheduling main goal is to maximize the priorities

of the scheduled tasks under operational constraints. Since the application is focused on emergency tasks, energy consumption is not the main concern.

An energy-driven scheduling algorithm has been proposed by Moser *et al.* [16]. The authors considers an embedded system which is able to harvest energy. They are based on the principle that tasks deadlines may be attended only when there is enough energy to execute them. Therefore, properties as the energy source availability, capacity of the energy storage as well as tasks power consumption shall be considered. Their scheduling algorithm considers the input energy, as well as the stored energy, in order to properly schedule the tasks, avoiding the deadlines missing for energy lack. However their formulation does not implies in energy harvesting maximization. They are able to respect more tasks deadlines when compared with the traditional Earliest Deadline First (EDF) algorithm [25,26]. However, the authors consider the input power actually fed into the energy storage. They do not analyze the task scheduling effect in the battery behavior and, consequently, in the harvested energy availability.

Finally, a very interesting and recent work has combined a nanosatellite application with an energy-efficient scheduling technique [17]. The authors emphasize the advantages of using nanosatellite swarms for synthetic aperture radar (SAR) application. Nanosatellite swarms may be more efficient and less costly than a single satellite solution. Although part of the authors' work problem definition is application dependent (SAR), it does consider energy constraints. The authors define a power function which has a minimum value to perform an assigned task.

The authors also introduced a satellite failure probability aspect in their problem formulation. Satellite's failures are modeled through a reliability function, based on Weibull distribution. Besides this, the authors have created a set of scenarios considering energy consumption and communication bandwidth, adding the failures probability component. Finally, the problem is reduced to the failures and power consumption minimization. The authors selected the following metrics to evaluate their algorithm performance: mean total energy consumption (MTEC), mean time to mission failure (MTMF), and mean time between failure (MTBF). Simulations have shown that their approach achieved better results than the general algorithms partial rescheduling (PR) and complete or full rescheduling (CR).

It is worth noting that this last discussed work took a step beyond the other mentioned above. It has not only added an energy constraint, but the proposed algorithm has been optimized to reduce power consumption. The next section shows the scheduling strategy proposed in this paper, which moves further on the energy issue, aiming the nanosatellite's energy harvesting maximization.

3. Solar panel control problem definition

The FloripaSat EPS prototype is based on the directly coupled circuit [18]. The solar panels are connected to the battery through a protection circuit. Therefore, the energy harvesting maximization depends on the battery voltage. Fig. 1 shows a simplified EPS circuit diagram. Equations (1) and (2) show the solar panel dependency on battery voltage.

$$V_{sp} = V_{bat} + V_{drop} \tag{1}$$

where V_{sp} is the solar panel voltage, V_{bat} is the battery voltage and V_{drop} is the voltage drop over the components between the solar panel and the battery.

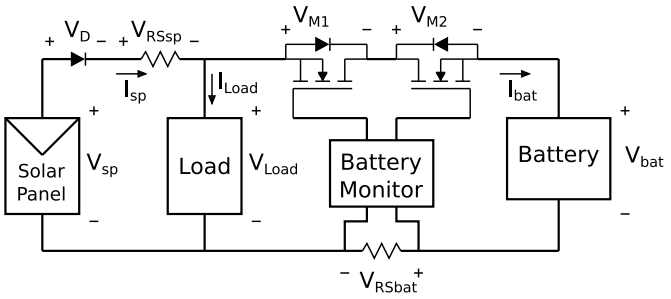


Fig. 1. EPS simplified circuit diagram.

$$V_{drop} = V_D + V_{RSsp} + V_{BD} + V_{DS} + V_{RSbat} \tag{2}$$

where V_D is the diode voltage drop, V_{RSsp} is the solar panel shunt resistor voltage drop, $V_{BD} = V_{M1}$ is the body diode voltage drop of the first battery protection mosfet, $V_{DS} = V_{M2}$ is the second battery protection mosfet drain to source voltage drop and finally V_{RSbat} is the battery shunt resistor voltage drop. The voltage drop over the diode, the shunt resistors and the mosfets are current dependent. Thus, every right term of Equation (2) may be redefined, starting from V_D:

$$I_D = I_S\left(e^{\left(\frac{V_D}{nV_T}\right)} - 1\right) \tag{3}$$

where I_D is the diode current in forward bias polarity, I_S is the saturation current, n is the ideality factor ranging between 1 and 2 and V_T is the thermal voltage defined by Equation (4).

$$V_T = \frac{kT}{q} \tag{4}$$

where k is the Boltzmann constant, T is the temperature in Kelvin and q is the electron charge. Considering $I_D \gg I_S$ Equation (3) may be simplified to Equation (5):

$$I_D \simeq I_S e^{\frac{V_D}{nV_T}} \tag{5}$$

From Equation (3) V_D may be obtained, as shown in Equation (6):

$$V_D = nV_T ln\frac{I_D}{I_S} \tag{6}$$

However, I_D is equal to the solar panel current I_{sp}. Finally the following equation is obtained:

$$V_D = nV_T ln\frac{I_{sp}}{I_S} \tag{7}$$

The voltage over the shunt resistor is directly obtained from Ohm's Law as shown in Equation (8), where R_{RSsp} is the shunt resistor through which flows the solar panel current I_{sp}.

$$V_{RSsp} = R_{RSsp} I_{sp} \tag{8}$$

For the circuit under study two n-channel mosfet operate as switches to protect the battery against over current, overvoltage and undervoltage through the battery monitoring IC (Fig. 1). Similarly to the V_D, the V_{BD} may be defined as Equation (9). It is important to note that the mosfet body diode not necessarily has the same fabrication parameters as the aforementioned diode. This implies in a different n (n_{BD}) and I_S (I_{SBD}). Also, the current flowing through this diode is not the same. From Fig. 1 it is clear that the body diode current is I_{bat}. Thus:

$$V_{BD} = n_{BD} V_T ln \frac{I_{bat}}{I_{SBD}} \tag{9}$$

Furthermore, the mosfet's drain to source current I_{DS} is given by Equation (10).

$$I_{DS} = k'_n \frac{W}{L} \left[(v_{GS} - V_t) V_{DS} - \frac{1}{2} V_{DS}^2 \right] \tag{10}$$

where $k'_n = \mu_n C_{ox}$ is the transconductance parameter determined by the fabrication technology, W/L is the channel aspect ratio, v_{GS} is the gate to source voltage, V_{DS} is the drain to source voltage and V_t is the threshold voltage. Assuming that V_{DS} is considerably small, the term V_{DS}^2 may be neglected resulting in Equation (11):

$$I_{DS} \simeq k'_n \frac{W}{L} (v_{GS} - V_t) V_{DS} \tag{11}$$

This equation shows a linear relation between I_{DS} and V_{DS}. Rewriting Equation (11) the R_{DSON} may be defined, which is a common parameter found in mosfet datasheets:

$$R_{DSON} = \frac{V_{DS}}{I_{DS}} = \frac{1}{k'_n \frac{W}{L} (v_{GS} - V_t)} \tag{12}$$

Therefore:

$$V_{DS} = R_{DSON} I_{DS} = \frac{I_{DS}}{k'_n \frac{W}{L} (v_{GS} - V_t)} = \frac{I_{bat}}{k'_n \frac{W}{L} (v_{GS} - V_t)} \tag{13}$$

The voltage over the battery shunt resistor is also directly obtained from Ohm's Law as shown in Equation (14), where R_{RSbat} is the shunt resistor through which flows the battery current I_{bat}.

$$V_{RSbat} = R_{RSbat} I_{bat} \tag{14}$$

Finally, Equation (2) may be rewritten from Equations (7)–(9) and (13) and (14) as follows:

$$V_{drop}(I_{sp}, I_{bat}) = n V_T ln \frac{I_{sp}}{I_S} + R_{RSsp} I_{sp} + n_{BD} V_T ln \frac{I_{bat}}{I_{SBD}} + \frac{I_{bat}}{k'_n \frac{W}{L} (v_{GS} - V_t)} + R_{RSbat} I_{bat} \tag{15}$$

All this formulation has shown that V_{drop} may be obtained from the solar panel current I_{sp} and from the battery current I_{bat}. Although the EPS provides individual solar panel and battery current measurements, there is no need for computing the V_{drop} in the microcontroller. This is the elegance of the problem formulation. By using the Perturb and Observe (P&O) maximum power point algorithm (Section 4), all that the EPS microcontroller has to do is to trigger the tasks and measure the solar panel delivered power (solar panel voltage and current measurements). A Maximum Power Point Tracking (MPPT) algorithm has been proposed and implemented in C language, which runs inside the EPS MSP430F249 microcontroller.

Recalling Equation (1), it is clear that the solar panel voltage (V_{sp}) depends on solar panel current I_{sp}, battery current I_{bat} and battery voltage V_{bat}. Solar panel current depends on the solar irradiance. The battery voltage and current, however, may be directly affected by the nanosatellite power consumption. This means that, according to the tasks being executed, the nanosatellite power consumption may vary, consequently, increasing or decreasing the battery voltage and current. This directly affects the nanosatellite power input.

4. Proposed scheduling algorithm strategy

Since the solar panel control problem has already been mathematically defined, the scheduling algorithm may be designed. This section clarifies how the scheduling algorithm has been implemented, taking into account the energy harvesting maximization and the tasks execution issues.

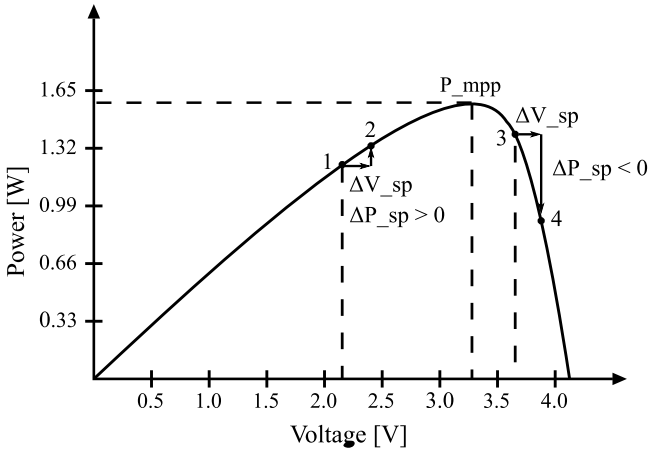


Fig. 2. P&O algorithm.

4.1. Energy harvesting maximization

As demonstrated in Section 3, the nanosatellite's tasks execution affects the operating point of the solar panels through the battery charging and discharging effect. Therefore, the proposed algorithm controls the nanosatellite load's current in order to keep the solar panels working as close as possible to their MPPT. To achieve this, the Perturb and Observe (P&O) method [19] is implemented in the EPS microcontroller.

The MSP430F249 internal analog to digital converter (ADC) is used to measure the solar panels voltage. ADC channels are also used to measure the solar panels current through shunt resistors and amplification circuits. These measurements allow the instantaneous input power calculation in the microcontroller (Equation (16)). The microcontroller calculates the input power in every iteration. For each power measurement the microcontroller compares the actual value ($P_{sp}(n)$) with the previous one ($P_{sp}(n-1)$). According to the measurements, more or less nanosatellite tasks are triggered, causing the battery voltage to vary and the solar panels to approximate to their optimum operating points.

$$P_{sp} = V_{sp} I_{sp} \tag{16}$$

Fig. 2 explains the algorithm. Unlike a hardware MPPT, the proposed algorithm does not change the solar panels voltage directly. For points 1 and 2 in Fig. 2, the energy-driven scheduling algorithm would reduce the number of tasks being executed, in order to increase battery voltage and, consequently, to increase the solar panel voltage (*Vsp*). This sets a new operating point for the solar panels, increasing the nanosatellite power input. The opposite behavior is verified in points 3 and 4, where the solar panel voltage increases and the input power moves away from the *Pmpp*. The algorithm shall identify this and shall increase the number of tasks to be executed in the next iteration. The conclusion is: solar energy harvesting maximization is ensured by the dynamic control of the task's execution. From this conclusion it is clear that the scheduling algorithm is energy-driven. Algorithm 1 shows a pseudo code for the scheduling algorithm, where perturb the system, in this case, means to execute more or less tasks.

Algorithm 1 MPPT Algorithm

```
initialize variables
measure voltage V_sp(n − 1)
measure current I_sp(n − 1)
compute power P_sp(n − 1) = V_sp(n − 1) · I_sp(n − 1)
perturb the system in arbitrary direction
loop
    measure voltage V_sp(n)
    measure current I_sp(n)
    compute power P_sp(n) = V_sp(n) · I_sp(n)
    if P_sp(n) > P_sp(n − 1) then
        perturb the system in the same direction as previous iteration
    else
        perturb the system in the opposite direction of previous iteration
    end if
    V_sp(n − 1) = V_sp(n)
    P_sp(n − 1) = P_sp(n)
end loop
```

For any set of tasks to be scheduled, independently of their power consumption and duration, it is possible to maximize the solar panels delivered power. Anytime that the battery voltage shall be changed, tasks will be anticipated or preempted. This means that the algorithm would give preference to maximizing the input energy rather than meeting the tasks' deadlines. The problem is that, not ensuring the tasks to be properly executed may be extremely harmful for the mission. For critical application satellites this approach may conflict with mission requirements. However, for scientific missions (most current nanosatellites' application), it may be more relevant to harvest more energy (increasing the overall satellite's number of tasks performed in orbit) than meeting non-critical tasks deadlines. Even though, the energy-driven approach shall be able to deal with tasks with different power consumption, duration and priority ensuring their proper execution. Therefore, next section explains how the energy-driven scheduling algorithm may address this issue.

4.2. Definition of tasks execution

The first thing to keep in mind is that the nanosatellite may be considered a multicore problem. Every single nanosatellite subsystem/module has its own microcontroller, co-working in a distributed control architecture. Fig. 3 shows how the nanosatellite tasks may be distributed in two subsequent orbit cycles. The tasks are executed by the nanosatellite subsystems/modules EPS, OBDH and TT&C. Therefore, for a given orbit O_n the tasks may be classified in three different sets: tasks already executed T_{ae}, tasks under execution T_{ue}, and tasks to be executed T_{tbe}. Thus, the total number of tasks T_T executed in an orbit O_n may be described as:

$$T_T = T_{ae} \cup T_{ue} \cup T_{tbe} \tag{17}$$

The energy-driven scheduling algorithm does not consider the already executed tasks in order to determine which tasks shall be executed next. Therefore, only a subset of T_T is taken into consideration, which is the T_c, consisting of the tasks under execution and the tasks to be executed within the orbit O_n. This defines the set of tasks considered by the energy-driven scheduling algorithm for each iteration.

$$T_c = T_{ue} \cup T_{tbe} \tag{18}$$

From Algorithm 1 in Section 4.1, there are two options every time the energy-driven scheduling algorithm is triggered: tasks under execution shall be preempted (t_{pp}); or tasks to be executed shall be anticipated (t_{atp}). The question is: may all the satellite tasks be preempted without prejudicing the mission's goal? For most applications the answer is no. But, as already discussed, for nanosatellites under scientific missions, the number of preemptive tasks tends to be much greater than for critical application satellites. Even though, the energy-driven scheduling algorithm shall ensure that the non-preemptive tasks are going

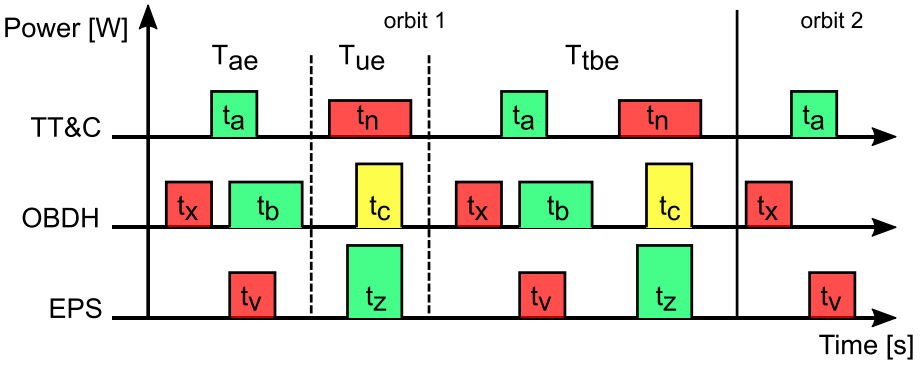


Fig. 3. Tasks distribution in orbits.

to be executed properly. Besides this, which task from the set T_{tbe} shall be triggered first when the algorithm identifies the need for tasks anticipation?

The solution proposed for these questions is to split the tasks' set T_c in three tasks' lists: high priority list L_{hp}; low priority list L_{lp}; and non-preemptive list L_{np}. These three lists shall be defined by the energy-driven scheduling algorithm designer, considering the nanosatellite mission requirements. Some rules must be defined in order to ensure that the energy-driven scheduling algorithm will execute all the tasks properly:

1. Only one task from L_{hp} may be executed at time;
2. Tasks to be anticipated are defined by the Earliest Deadline First (EDF) criteria;
3. Tasks under execution from L_{np} must not be preempted.

The traditional EDF algorithm has been already defined as inefficient for attending the tasks deadlines in such energy harvesting embedded systems [16]. However, in this case, it is combined with the energy maximization algorithm, which considers the energy input aspect every iteration.

Equations (21)–(23) show a hypothetical example for the lists configuration for the tasks of Fig. 3. Besides the tasks distribution, Fig. 3 uses colors to represent tasks nature: red - high priority; green - low priority; yellow - non-preemptive. Considering those lists and set of tasks, and supposing that the energy-driven scheduling algorithm has determined that a task shall be anticipated (t_{atp}), that task shall be t_a. Based on the proposed strategy, follows:

$$T_{ue} = \{t_n, t_c, t_z\} \tag{19}$$

$$T_{tbe} = \{t_a, t_n, t_x, t_b, t_c, t_v, t_z\} \tag{20}$$

$$L_{hp} = \{t_n, t_x, t_v\} \tag{21}$$

$$L_{lp} = \{t_a, t_b, t_z\} \tag{22}$$

$$L_{np} = \{t_c\} \tag{23}$$

$$t_{atp} = t_a \tag{24}$$

Although t_x has the first deadline, it is in L_{hp} as well as t_n, and two high priority tasks must not be executed simultaneously. Therefore $t_{atp} = t_a$. Now, considering the same scenario of Fig. 3, but supposing that a solar irradiance variation has caused the algorithm to preempt some task. That task shall be t_z. This occurs because t_c is in L_{np} and must not be preempted. The tasks organization in these lists, as well as the rules to execute that tasks, allow both the energy harvesting maximization, as well as the tasks to be properly executed. Even for tasks with different power consumption and duration, as shown in Fig. 3, the tasks' deadlines may be accomplished.

However, an important remark shall be made here. The number of non-preemptive tasks to be scheduled is inversely proportional to the energy harvesting maximization efficiency. Also, the more energy the tasks takes the lower is the efficiency of the scheduling algorithm. This introduces the concept of task granularity, which may be considered equivalent to the tasks consumed energy. Keeping the grains small (low energy tasks) allows the algorithm to respond faster to the solar irradiance and temperature variations, which results in harvesting more energy.

$$t_E = \int_a^b t_P dt \tag{25}$$

where t_E is the task energy; t_P is the task power consumption; and $b - a$ is the time to execute the task.

4.3. Algorithm iteration interval

In addition to the tasks execution issue, another important aspect that must be taken into account when designing the energy-driven scheduling algorithm is the iteration interval. Ideally, the shorter the scheduling iteration interval the better is the result on maximizing the energy harvested. This occurs due to the satellite fast dynamics (position on orbit and tasks execution). Since the scheduling algorithm controls the satellite's tasks in order to keep the solar panels voltage on the optimum value (maximum power point voltage Vmpp) and because the optimum condition continually changes, the algorithm shall be triggered as fast as possible. However two trade-offs must be considered to define the optimum algorithm iteration interval: power consumption and computational capacity.

Starting from the algorithm power consumption issue, as shown in Section 4.1, the operations to maximize the solar energy input are the following: measuring the solar panels voltage and current (ADC readings), computing the instantaneous delivered power, comparing the instantaneous input power with the previous value and enabling/disabling the satellite's tasks (IOs configuration or communication through I^2C). All of these operations are executed by the ultra low power MSP430 microcontroller, which consumes few micro watts to execute them. Therefore, the power consumption to perform one algorithm computation may be considered insignificant when compared with the nanosatellite overall power consumption, the EPS power consumption, or even with the energy harvesting gain by triggering the scheduling algorithm.

However, the problem arises when the scheduling algorithm iteration interval is so small that the microcontroller ultra low power consumption level is higher than the energy harvested. This case rarely occurs before the microcontroller computational resources limit is achieved. However, it is worthy to define limits to the iteration interval. The following three conditions shall be attended in order to achieve the best performance for the proposed energy-driven scheduling algorithm:

1. $t_{aii} > t_{aci}$;
2. $t_{aii} > t_{pc} \rightarrow E_{ai} < E_{hg}$;
3. t_{aii} shall be made as smaller as possible, respecting the two above conditions.

where t_{aii} is the energy-driven scheduling algorithm iteration interval, t_{aci} is the available interval for the computational resource to compute the scheduling algorithm outputs, and t_{pc} is the interval limit for which the algorithm iteration energy consumption E_{ai} is smaller than the energy harvesting gain E_{hg}.

Considering an EPS based on a single core microcontroller, and that there is no operating system to allow multithreading, then, the scheduling algorithm low period iteration may be considered a computational capacity restriction problem. However, it is important to mention that, our case study (FloripaSat), as well as most of recent nanosatellites, is based on a distributed computational architecture. This means that each satellite module/subsystem (EPS, TT&C, OBDH, ADCS) has its own microcontroller. This drastically reduces the computational resource problem, since the EPS has a dedicated microcontroller, with only two main functions: battery management and scheduling algorithm execution. The satellite tasks control shall be performed by each module/subsystem. In most cases, the EPS function is only to inform, through the I^2C protocol (which connects all modules/subsystems), that more or less tasks shall be executed. In short, our suggestion is to firstly design the EPS battery management embedded software (the most critical function), allocating the remaining EPS microcontroller computational capacity to be used to the scheduling algorithm.

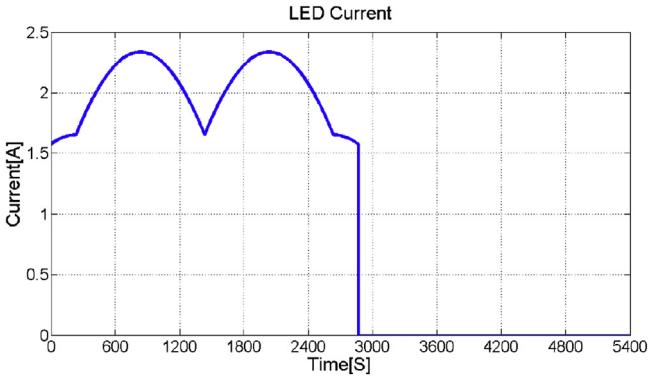


Fig. 4. LED current.

The optimal energy-driven scheduling algorithm iteration interval may be experimentally obtained, considering the mission requirements (computational resource, microcontroller power consumption, etc). For our case study (FloripaSat), t_{aii} was defined as 455 ms (see Section 5.2). Tests have been performed with smaller iteration interval (400 ms), leading to even better results (more energy harvested). This means that $t_{aii} = 400\ ms > t_{pc}$. However, the iteration interval could not be made smaller than 455 ms because this was the smallest interval to avoid conflicts with the critical functions of battery management, which are also performed by the EPS microcontroller.

5. Experimental setup

5.1. Test stand

In order to evaluate the above mentioned scheduling algorithm an EPS test stand is proposed. This test stand consists of a controlled light emitter system, a four channel power analyzer (N6705B) and a computer. Many different tests may be performed as follows:

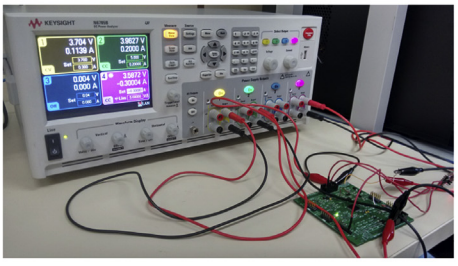
1 **Solar panel emulation:** One of the four power analyzer channels is set with positive voltage and positive current, emulating the solar panel behavior. This first test allows the evaluation of EPS features regarding the energy input, like solar panel voltage and current measurements;
2 **Battery emulation:** One of the four power analyzer channels is set with positive voltage and positive current (discharging mode) or negative current (charging mode), emulating the nanosatellite battery. This test allows the evaluation of EPS battery monitoring features like: input/output battery current measurement, battery voltage measurement, overvoltage and undervoltage protection;
3 **Nanosatellite power consumption emulation:** This test may be performed setting one power analyzer channel with positive voltage and negative current, working as a controlled electronic load. The parameters (voltage and current) may be set by equations, by stored data vectors or even in real time, by a LabVIEW virtual instrument (VI). This feature allows to test the EPS with a variable electrical load as suggested in Section 4;
4 **Orbit-position input power emulation:** A light emitter system (high power LEDs) is controlled by one of the power analyzer channels. The light emitter system illuminates one or more solar panels, which deliver energy to the EPS. Since the power analyzer has three other available channels, one or more of the above mentioned tests may be performed simultaneously. For instance, the whole scenario may be emulated selecting channel one for controlling the light emitter system, channel two for the battery emulation and channel three for the power consumption emulation. In this configuration, a long term test may be performed, where many orbit cycles may be emulated, experimentally evaluating not only the EPS itself, but mainly, the scheduling algorithm;
5 **Computer communication:** The EPS has a debug USART serial interface for sending important data to the computer. This interface is used only for on-shore tests. This allows, for instance, to identify all the decision taken by the EPS, including the triggered tasks.

5.2. Test configuration

The results presented in Section 6 have been obtained from a three orbital cycle test performed at the laboratory. A LED lamp had its light intensity controlled by one power analyzer channel working on current setting mode. For this channel, a current curve is generated, emulating the behavior of the power delivered to the solar panels in orbit (Fig. 4). The input power behavior is well known for nanosatellites in LEO [20]. Two solar panels have been connected in parallel and placed inside a box to avoid external light influences. The FloripaSat is provided with six solar panels, however, not all of them are going to deliver power to the system simultaneously. Due to the satellite rotation, only two or three panels are hit by the sunlight simultaneously (according to orbital positioning). Fig. 5a shows a picture of the FloripaSat's prototype and Fig. 5b shows the EPS under test.

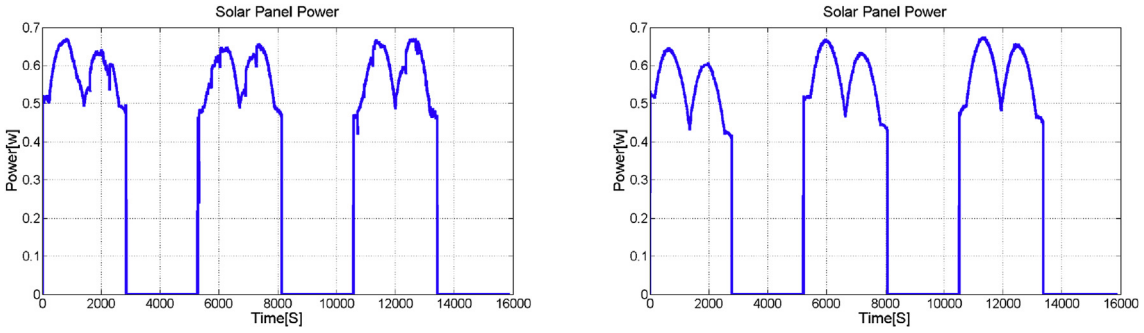
The second power analyzer channel emulates the satellite load (working as an electronic load). It represents the nanosatellite tasks in

(a) FloripaSat prototype



(b) FloripaSat EPS under test

Fig. 5. FloripaSat prototype and EPS under test.



(a) Test with the scheduling algorithm (b) Test without the scheduling algorithm

Fig. 6. Solar panel power.

execution. For each algorithm iteration the load current is increased or decreased in 10 mA, according to the calculation performed by the microcontroller (every 455 ms). The power analyzer control has been implemented in LabVIEW. The computer receives the EPS command (through serial interface) to increase or decrease the load current and the LabVIEW actuate on the power analyzer. In normal operation, the EPS shall send the commands to the OBDH, which will trigger the nanosatellite tasks. The decision of using an electronic load instead of the real FloripaSat tasks was made to simplify the experiment. Not all the FloripaSat software control is ready to use yet. Besides this, the electronic load provides more flexibility without compromising the test consistency.

Also, as discussed in Section 6 the temperature affects the solar panels efficiency. For this reason, the solar panels temperature is also measured. Two T-type thermocouples (one for each solar panel) have been attached on the solar panels surface. The temperature has been sampled every second and the data was sent to the computer through the Omega HH127 Thermocouple Datalogger and Thermometer.

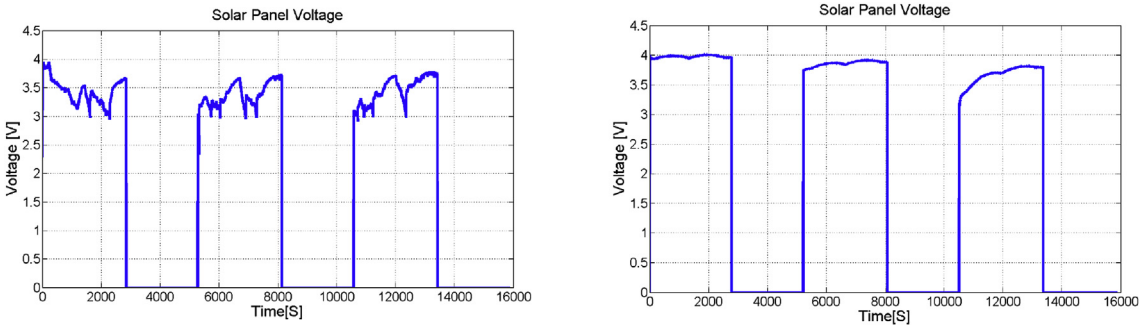
6. Results

The results presented in this section have been obtained from tests of three orbital cycles. Each cycle has nearly 90 min, therefore, each test has approximately 270 min. First, the EPS running the proposed energy-driven scheduling algorithm has been evaluated. The algorithm works only during the orbit period, where the sunlight hits the solar panels. During the eclipse the current provided to the load is kept constant (50 mA). After the test with the scheduling algorithm is finished, the average current provided to the load is calculated. Than, a test without scheduling algorithm is performed. In the test without the algorithm, the load current is kept constant (with the average current from the test with the scheduling algorithm) during the three orbit cycles.

Fig. 6a shows the power delivered by the solar panels during the test with the scheduling algorithm. Comparing with the result obtained from the test without the scheduling algorithm (Fig. 6b), one notes that the nanosatellite has harvested more energy with the energy-driven scheduling algorithm. Considering the complete test for three orbits, the scheduling algorithm allowed the EPS to harvest 4.48% more energy than the EPS running without the scheduling algorithm. The greater difference occurs in the first orbit cycle, where the proposed algorithm has allowed an energy harvesting gain of 8.46%.

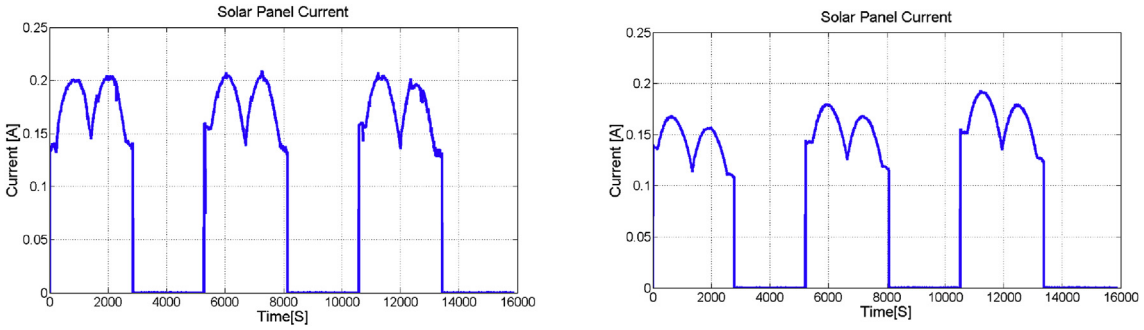
The solar panels voltage curve allows one to better understand why the difference was greater in the first orbit cycle. Fig. 7 presents the solar panels voltage for both tests. Note that the solar panel voltage drops in the beginning of Fig. 7a. The algorithm identifies that the battery voltage is too high, causing the solar panels voltage to operate far away from the MPP. Therefore, more tasks start to be triggered (electronic load current increased in steps of 10 mA). The battery voltage drop (due to the increased current provided to the load) causes the solar panels voltage also to decrease. Therefore, the solar panels operate closer to the MPP in this cycle. Fig. 7b shows the opposite behavior for the experiment without the energy-driven scheduling algorithm. Since the load current is kept constant in this test (no scheduling algorithm) the battery voltage increases as the solar irradiance increases. This causes the solar panels to operate far away from the MPP, harvesting less energy.

Note that for the two next cycles, the difference in the power delivered by the solar panels decreases (Fig. 6). This happens due to the battery voltage drop in the test without the energy-driven scheduling algorithm. Even with no control of the load current, the solar panel voltage drops for the two last cycles, due to the natural battery discharge providing constant current to the load. Longer tests may show



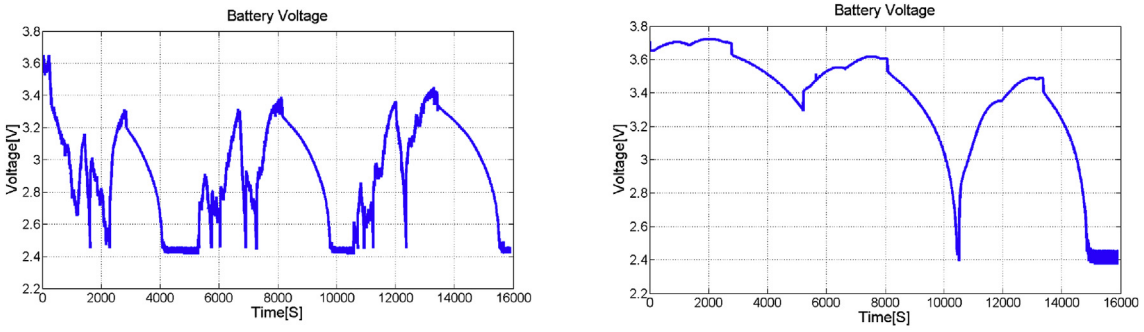
(a) Test with the scheduling algorithm (b) Test without the scheduling algorithm

Fig. 7. Solar panel voltage.



(a) Test with the scheduling algorithm (b) Test without the scheduling algorithm

Fig. 8. Solar panel current.



(a) Test with the scheduling algorithm (b) Test without the scheduling algorithm

Fig. 9. Battery voltage.

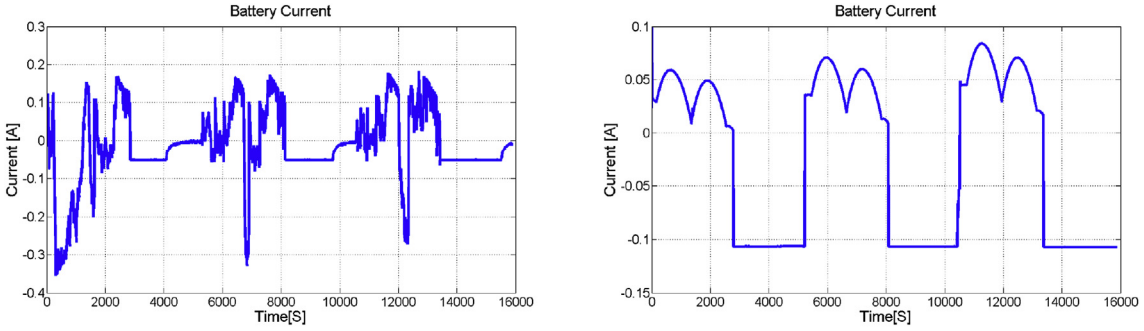
that, after the battery reaches its under-voltage limits the system without the energy-driven scheduling algorithm would be even more inefficient.

Another important issue to keep in mind is the matching between the solar panels and the battery [21]. For any satellite and specially for the EPS with the directly coupled circuit architecture, the correct definition of batteries and solar panels may drastically affect the overall satellite efficiency. For our case study, the solar panels may be designed to have a higher V_{mpp}. This would avoid the solar panels voltage to move far from their optimal operation point during the eclipse. Since the solar panels V_{mpp} is close to the battery voltage knee, it is harder to keep the battery voltage around this point. Therefore, the EPS hardware design may also affect the energy-driven scheduling algorithm.

Fig. 8 shows the great difference in the solar panel current with and without the energy-driven scheduling algorithm. For all three orbit cycles one may note that the solar panel current has almost the same behavior for the test with the scheduling algorithm (Fig. 8a). This demonstrates the efficient control to operate the system on the MPP. For the case without the energy-driven scheduling algorithm (Fig. 8b), one may note that the harvested current increases every cycle, since the solar panel approximates the MPP, due to the natural discharge of battery. For longer tests, this efficiency shall also decrease when the battery voltage drops below the optimal voltage that causes the solar panels to operate close to the maximum power point.

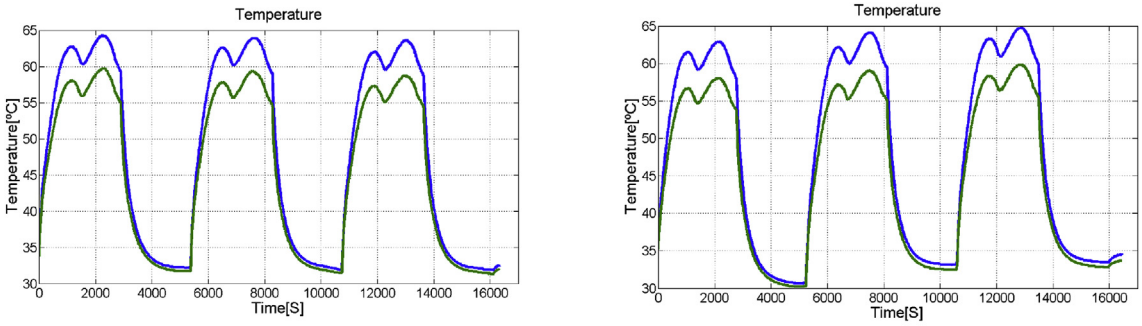
Fig. 9 shows the battery voltage for both tests. Fig. 9a shows the algorithm impact in battery voltage, by changing the load current. On the other hand, Fig. 9b shows the natural charging and discharging process caused by the solar panel input current and the constant load current (104.8 mA). Another important aspect in these plots is the under-voltage limit around 2.4 V. Note that, due to the algorithm load current control, this limit is reached since the first orbit cycle, where the battery voltage is reduced when the solar irradiance reduces. For the test without the energy-driven scheduling algorithm this limit occurs only in the third orbit cycle.

Fig. 10 also shows the battery current for both tests. In these plots the difference is more clear. For the case with the energy-driven



(a) Test with the scheduling algorithm (b) Test without the scheduling algorithm

Fig. 10. Battery current.



(a) Test with the scheduling algorithm (b) Test without the scheduling algorithm

Fig. 11. Solar panel temperature.

scheduling algorithm (Fig. 10a), the battery current varies significantly, in order to compensate temperature and irradiance changes, keeping the solar panel voltage as close as possible to the MPP. Fig. 10b shows a more regular behavior, where the current varies only due to the solar panel current input (since the load current is constant for this case).

Finally, Fig. 11 shows the solar panels temperature during the tests. The experiments were performed inside a closed chamber, in order to avoid external light to affect the tests. LEDs lights were focused on solar panels increasing their temperatures due to irradiation effect. During the shadow period, solar panels temperature decreases, but it increases gradually as the LEDs are turned on again. A cooler to remove the warm air from the chamber was used, however it was not effective enough due to convection heat transfer limitation. A new test-bed with two more powerful coolers is been designed to decrease the heating effect on the solar panels. Nevertheless, nanosatellites face similar conditions in space, where properly temperature control on solar panels is not possible. Indeed, temperature ranges will be different in space, but heating effect after leaving the shadow region will always be present on the solar panels.

Temperature variation is the main cause to change the solar panel V_{MPP}. Comparing Fig. 11 with Figs. 8 and 6 one may note that the temperature variation affects the solar panel current and power output. The higher the temperature, the lower the delivered power. This occurs due to the great variation in the V_{MPP} for different temperatures. This justifies the power and current peaks difference in the same orbit cycle in Figs. 6b and 8b. Note that for the test with the energy-driven scheduling algorithm this effect is reduced due to the dynamic change in load's current, reducing the battery voltage when the temperature increases (Figs. 6a and 9a).

7. Conclusions and future work

The proposed algorithm has demonstrated its energy harvesting maximization capability. Considering the complete test with three orbits, the EPS running with the energy-driven scheduling algorithm harvested 4.48% more energy than the EPS running without the algorithm. An energy gain of 8.46% has been achieved for the first orbit cycle, where the initial battery voltage causes the solar panels to operate far from the V_{mpp}. The experiments also have shown that the natural battery discharge (operation without scheduling algorithm) momentarily approximates the solar panel voltage to the V_{mpp}. However, due to the cyclical nature of the battery charge/discharge process in orbit, tests longer than three orbit will show that the EPS harvesting efficiency without the energy-driven scheduling algorithm will decrease again. Besides this, the temperature effects on solar panel power output has been reduced by the load's current control provided by the energy-driven scheduling algorithm. This is also a benefit of the proposed algorithm.

This work has experimentally proved that tasks control may optimize energy harvesting capability for directly coupled EPS. However, in this paper, the tasks have been emulated as steps of 10 mA in an electronic load. The next step is to model the CubeSat tasks considering their power consumption and duration, using the lists' tasks organization proposed in Section 4.2. The Perturb and Observe algorithm shall then be tuned to operate with this set of tasks. Also, tasks priorities optimization shall be implemented in the new version of the energy-driven scheduling algorithm.

Finally, an energy-driven scheduling algorithm method shall consider the battery state of charge in order to choose the tasks to be performed. The battery monitoring chip provides this feature as well. However, the chip does not consider the aging effect caused by the charging/discharging cycles. Thus, the idea is also to include in the scheduling policy a battery aging prediction model [22–24]. This will allow the scheduling algorithm to adapt its priorities based on the battery condition.

Acknowledgement

The authors acknowledge the Coordination for the Improvement of Higher Education Personnel (CAPES), which partially supported this research through the PhD grant number 3447735; the National Council for Scientific and Technological Development (CNPq), which partially supported this work under the project number 402184/2013-0, through the PhD grant number 141696/2017-6, and the postdoctoral grant number 150012/2018-7; and the Brazilian Space Agency (AEB), for having partially supported this work under Uniespaço Program, Opportunity Announce AO/2013.

References

[1] A. Ali, M. Mughal, H. Ali, L. Reyneri, Innovative power management, attitude determination and control tile for cubesat standard nanosatellites, Acta Astronaut. 96 (2014) 116–127, http://dx.doi.org/10.1016/j.actaastro.2013.11.013 http://www.sciencedirect.com/science/article/pii/S0094576513004165.
[2] The CubeSat Program, Cubesat Design Specification Rev. 13, Tech. Rep., Cal Poly SLO (vol. 02 2014).
[3] Q. Dishan, H. Chuan, L. Jin, M. Manhao, A dynamic scheduling method of earth-observing satellites by employing rolling horizon strategy, Sci. World J. 2013 (2013) 11.
[4] L. Del Consuelo Hernandez Ruiz Gaytan, Z. Pan, J. Liu, S. Shimamoto, Dynamic Scheduling for High Throughput Satellites Employing Priority Code Scheme, Access vol. 3, IEEE, 2015, pp. 2044–2054.
[5] J. Wang, X. Zhu, D. Qiu, L. Yang, Dynamic scheduling for emergency tasks on distributed imaging satellites with task merging, Parallel and Distributed Systems, IEEE Transactions 25 (9) (2014) 2275–2285.
[6] D. Christopoulos, S. Chatzinotas, B. Ottersten, Multicast multigroup precoding and user scheduling for frame-based satellite communications, Wireless Communications, IEEE Transactions 14 (9) (2015) 4695–4707.
[7] H. Kim, Y.K. Chang, Mission scheduling optimization of sar satellite constellation for minimizing system response time, Aero. Sci. Technol. 40 (2015) 17–32.
[8] K. Han, Y. Liu, J. Luo, Duty-cycle-aware minimum-energy multicasting in wireless sensor networks, Networking, IEEE/ACM Transactions 21 (3) (2013) 910–923.
[9] A.S. Hoeller, A.A. Fröhlich, Evaluation of energy-efficient heuristics for acobased routing in mobile wireless sensor networks, Advanced Studies in Computer Science and Engineering, Int. J. 4 (2015) 8.
[10] Y. Tan, X. Yin, A dynamic scheduling algorithm for energy harvesting embedded

systems, EURASIP J. Wirel. Commun. Netw. 2016 (1) (2016) 114, http://dx.doi.org/10.1186/s13638-016-0602-8 https://doi.org/10.1186/s13638-016-0602-8.
[11] C. Ahara, J. Rossbach, The scheduling problem in satellite communications systems, Communication Technology, IEEE Transactions on 15 (3) (1967) 364–371.
[12] S. Durrani, K. Jo, Efficient scheduling algorithm for demand-assigned tdma satellite systems, Aerospace and Electronic Systems, IEEE Transactions on 25 (2) (1989) 259–267.
[13] A. Globus, J. Crawford, J. Lohn, A. Pryor, A comparison of techniques for scheduling earth observing satellites, AAAI, 2004, pp. 836–843.
[14] X. Zhu, J. Wang, X. Qin, J. Wang, Z. Liu, E. Demeulemeester, Fault-tolerant scheduling for real-time tasks on multiple earth-observation satellites, Parallel and Distributed Systems, IEEE Transactions 26 (11) (2015) 3012–3026.
[15] Z. Zheng, J. Guo, E. Gill, Swarm satellite mission scheduling & planning using hybrid dynamic mutation genetic algorithm, Acta Astronaut. 137 (2017) 243–253, http://dx.doi.org/10.1016/j.actaastro.2017.04.027 http://www.sciencedirect.com/science/article/pii/S0094576516310396.
[16] C. Moser, D. Brunelli, L. Thiele, L. Benini, Real-time scheduling with regenerative energy, 18th Euromicro Conference on Real-time Systems (ECRTS'06), 2006, p. 10, , http://dx.doi.org/10.1109/ECRTS.2006.23 pp.–270.
[17] C.K. Pang, A. Kumar, C.H. Goh, C.V. Le, Nano-satellite swarm for sar applications: design and robust scheduling, Aerospace and Electronic Systems, IEEE Transactions 51 (2) (2015) 853–865.
[18] L.K. Slongo, S.V. Martnez, B.V.B. Eiterer, E.A. Bezerra, Towards cubesat electrical power system efficient designs, I Latin American IAA CubeSat Workshop, 2014.
[19] S. Saravanan, N.R. Babu, Maximum power point tracking algorithms for photovoltaic system a review, Renew. Sustain. Energy Rev. 57 (2016) 192–204, http://dx.doi.org/10.1016/j.rser.2015.12.105 http://www.sciencedirect.com/science/article/pii/S1364032115014884.
[20] D.Y. Lee, J.W. Cutler, J. Mancewicz, A.J. Ridley, Maximizing photovoltaic power generation of a space-dart configured satellite, Acta Astronaut. 111 (2015) 283–299, http://dx.doi.org/10.1016/j.actaastro.2015.01.022 http://www.sciencedirect.com/science/article/pii/S0094576515000375.
[21] J. Lee, E. Kim, K.G. Shin, Design and management of satellite power systems, 2013 IEEE 34th Real-time Systems Symposium, 2013, pp. 97–106, , http://dx.doi.org/10.1109/RTSS.2013.18.
[22] V. Agarwal, K. Uthaichana, R. DeCarlo, L. Tsoukalas, Development and validation of a battery model useful for discharging and charging power control and lifetime estimation, Energy Conversion, IEEE Transactions 25 (3) (2010) 821–835.
[23] M. Gholizadeh, F. Salmasi, Estimation of state of charge, unknown nonlinearities, and state of health of a lithium-ion battery based on a comprehensive unobservable model, Industrial Electronics, IEEE Transactions 61 (3) (2014) 1335–1344.
[24] J. Schmalstieg, S. Kabitz, M. Ecker, D. Sauer, From accelerated aging tests to a lifetime prediction model: analyzing lithium-ion batteries, Electric Vehicle Symposium and Exhibition (EVS27), 2013 World, 2013, pp. 1–12.
[25] C.L. Liu, James W. Layland, Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment, J. ACM 0004-5411, 20 (1) (Jan. 1973) 46–61, http://dx.doi.org/10.1145/321738.321743 acmid: 321743, ACM, New York, NY, USA http://doi.acm.org/10.1145/321738.321743.
[26] Feng Shan, Junzhou Luo, Xiaojun Shen, Optimal energy efficient packet scheduling with arbitrary individual deadline guarantee, Comput. Network. 75 (2014) 351–366 1389-1286 https://doi.org/10.1016/j.comnet.2014.10.022 http://www.sciencedirect.com/science/article/pii/S1389128614003843.

Leonardo Kessler Slongo is a post-doc researcher at Federal University of Santa Catarina (UFSC). He currently conducts his research at Communications and Embedded Systems Lab (LCS). He received his Bachelor, Master and PhD degrees in Electrical Engineering from UFSC in 2011, 2013 and 2017 respectively. His research areas include scheduling algorithms, embedded systems for space applications, solar energy harvesting systems, nanosatellite electrical power systems and low power electronics.

Sara Vega Martínez is a Ph.D. student in Electrical Engineering at Federal University of Santa Catarina (UFSC). She received her Bachelor and Master degrees in Electrical Engineering from University of Las Palmas de Gran Canarias (ULPGC), Spain, in 2016. Her research areas include embedded systems hardware and software design, solar energy harvesting systems and nanosatellite electrical power systems.

Bruno Vale Barbosa Eiterer is a Bachelor student in Electrical Engineering at Federal University of Santa Catarina (UFSC). His research areas include embedded systems software design, solar energy harvesting systems and nanosatellite electrical power systems.

Túlio Gomez Pereira is a Bachelor student in Electrical Engineering at Federal University of Santa Catarina (UFSC). His research areas include embedded systems hardware design and test, solar energy harvesting systems and nanosatellite electrical power systems.

Eduardo Augusto Bezerra is a Researcher and Lecturer of Computer Engineering at Federal University of Santa Catarina (UFSC), where he is with the Department of Electrical and Electronic Engineering since 2010. He received his Ph.D. in Computer Engineering from the University of Sussex (Space Science Centre), England, UK, in 2002. His research interests are in the areas of embedded systems, space applications, computer architecture, reconfigurable systems (FPGAs), software & hardware testing, fault tolerance and microprocessor applications.

Kleber Vieira de Paiva is currently a professor at the Federal University of Santa Catarina (UFSC), Brazil. He is the header of the research group Thermal Fluid Flow Group - UFSC. He received his Ph.D. in Mechanical Engineering and Thermal Sciences from UFSC in 2011 and his master's degree in the same area and institution, in 2007. He has experience in the development of compact heat exchangers for thermal control of electronic components for space applications: mini heat pipes and thermosyphons.