UNIVERSIDADE FEDERAL DE SANTA CATARINA

CENTRO TECNOLÓGICO

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Vinicius Pimenta Bernardo

Hardware-in-the-loop verification of an on-board energy-driven scheduling algorithm for CubeSats

Florianópolis

2022

Vinicius Pimenta Bernardo

Hardware-in-the-loop verification of an on-board energy-driven scheduling algorithm for CubeSats

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina para a obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Prof. Eduardo Augusto Bezerra, Dr.

Florianópolis

2022

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Bernardo, Vinicius
Hardware-in-the-loop verification of an on-board energy driven scheduling algorithm for CubeSats / Vinicius Bernardo ; orientador, Eduardo Bezerra, 2023.
55 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, 2023.

Inclui referências.

1. Engenharia Elétrica. 2. Hardware-in-the-loop. 3. Escalonamento de tarefas. 4. CubeSats. I. Bezerra, Eduardo. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

Vinicius Pimenta Bernardo

Hardware-in-the-loop verification of an on-board energy-driven scheduling algorithm for CubeSats

O presente trabalho em nível de mestrado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Laio Oriel Seman, Dr.

Universidade Federal do Vale do Itajaí

Prof. Telles Brunelli Lazzarin, Dr.

Universidade Federal de Santa Catarina

Eng. Edras Reily Pacola, Dr.

Microchip Technology Inc.

Certificamos que esta é a versão original e final do trabalho de conclusão que foi julgado adequado para obtenção do título de mestre em Engenharia Elétrica.

Documento assinado digitalmente
Telles Brunelli Lazzarin
Data: 10/02/2023 14:38:57-0300
CPF: ***.400.719-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Telles Brunelli Lazzarin, Dr.

Coordenação do Programa de Pós-Graduação

Documento assinado digitalmente
Eduardo Augusto Bezerra
Data: 09/02/2023 19:13:25-0300
CPF: ***.851.577-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Eduardo Augusto Bezerra, Dr.

Orientador

Florianópolis, 2022.

RESUMO

Aprimoramentos tecnológicos e a miniaturização de componentes permitiram a redução do tamanho médio dos satélites lançados, que passaram a conter componentes "comerciais de prateleira", que possuem custo reduzido, demandam baixa potência e estão amplamente disponíveis. Esta tendência culminou no desenvolvimento do padrão CubeSat, e no fato de que, hoje em dia, a maioria dos lançamentos de satélites estão na categoria "pequenos satélites" (até 180kg). Como consequência das dimensões reduzidas dos satélites, surgem esforços na busca de meios para otimizar a gestão e o consumo de energia. Tais esforços incluem avaliar qual arquitetura de sistema de energia fornece a melhor eficiência geral, que geralmente inclui rastreamento de ponto de energia máximo (MPPT) para coletar energia usando painéis solares, que correspondem à fonte primária de energia mais comum em pequenos satélites. Nesse contexto, este trabalho propõe o uso de uma estratégia de escalonamento de tarefas baseado no algoritmo da mochila 0-1 visando maximizar a captação de energia. Um *framework* termoelétrico integrado de nano satélites foi usado para testar funções de prioridade e diferentes estratégias de ativação de aquecedores usando parâmetros de tarefas do FloripaSat-I e outros casos gerados aleatoriamente. Os resultados mostraram que uma função de prioridade saturante apresentou a menor quantidade de perdas de prazo de execução. Embora as diferentes funções de prioridade não tenham apresentado influência significativa na temperatura da bateria, ao alocar os recursos restantes diretamente para o aquecedor após a fase de seleção das tarefas, obteve-se um aumento na correlação entre a geração de energia ideal e alcançada dos painéis solares e a temperatura da bateria se manteve mais próxima da faixa de temperatura desejada ao longo da simulação. A estratégia de controle foi posteriormente implementada em um ambiente embarcado, configurando uma simulação de *hardware-in-the-loop* (HIL). O comportamento do algoritmo foi verificado, juntamente com sua capacidade de cumprir as restrições de tempo real. Uma análise de desempenho foi realizada e verificou-se um impacto linear do número de tarefas no tempo de computação.

Palavras-chave: Hardware-in-the-loop. Escalonamento de tarefas. CubeSats.

RESUMO EXPANDIDO

Introdução

Desde o lançamento do Sputnik 1 em 1957, centenas de satélites foram lançados. Aperfeiçoamentos tecnológicos e a miniaturização de componentes permitiram a redução do tamanho médio dos satélites lançados, que passaram a conter muitos componentes "comercial off the shelf" (COTS), que são baratos, demandam baixa potência e estão amplamente disponíveis. Essa tendência culminou no conceito CubeSat. O padrão CubeSat foi criado pela Universidade Politécnica do Estado da California (Cal Poly), e pela Universidade de Stanford em 1999 e definido como 1U o cubo de 10 cm x 10 cm x 10 cm com no máximo 2kg. Desde então, a maioria dos lançamentos de satélites está na categoria "*small-sat*" (até 180kg) e envolve nanossatélites (1 a 10kg) e o padrão CubeSat. O Sistema de Energia Elétrica (EPS) é um subsistema essencial de um nanossatélite que deve colher do ambiente espacial circundante, armazenar e, finalmente, fornecer energia a outros subsistemas. O método mais comum para a colheita de energia é através do efeito fotovoltaico utilizando painéis solares. Ao operar um painel solar, para produzir a maior quantidade possível de energia, é desejável ficar o mais próximo possível do Ponto de Máxima Potência (MPP). Com a diminuição do consumo de energia dos sistemas embarcados, o uso de circuitos *Maximum Power Point Tracking* (MPPT) pode representar uma desvantagem em aplicações de baixa potência considerando seu próprio consumo adicionado ao sistema, e um circuito acoplado diretamente é mais eficiente do que usar um circuito MPPT dedicado. Estudos anteriores demonstraram matematicamente que esta última arquitetura, juntamente com a arquitetura diretamente acoplada, poderia se beneficiar do controle da execução de tarefas. Os satélites podem não receber um nível de irradiância contínuo ao longo de toda a órbita. Isso, juntamente com outros fatores relacionados à física de conversão, torna essa fonte de energia instável. Portanto, os dispositivos de armazenamento de energia são extremamente comuns. Os atuais sistemas de armazenamento de energia de última geração usam baterias de íon de lítio (Li-ion) ou de polímero de lítio (LiPo) do tipo secundário (recarregáveis). Em relação às baterias de íon-lítio, os efeitos da baixa temperatura estão muito mais relacionados ao meio ambiente, geralmente ocorrem em áreas de alta latitude e no espaço, e os efeitos envolvem desaceleração da atividade de reação química e velocidade de transferência de carga, resultando em redução de energia e potência capacidades. Os efeitos de alta temperatura não se limitam ao ambiente e ocorrem em uma ampla gama de aplicações. Geralmente, as perdas de lítio e a degradação do material ativo resultam em redução de capacidade, e o aumento da resistência interna causa perda de potência, e esses efeitos ocorrem tanto em baixa quanto em alta temperatura com diferentes causas. Com o aumento do uso de pequenos satélites, notou-se um alto índice de falhas. Entre as causas estão erros de design de *hardware* e *software*, além de falhas no processo de integração. Nesse viés, prever e testar efetivamente o sistema pode ajudar a descobrir e evitar falhas que, de outra forma, comprometeriam a missão. Um método eficaz para modelar e simular um sistema de engenharia, que consiste na combinação de simulação computacional e *hardware* em uma única plataforma, é chamado de *hardware-in-the-loop* (HIL). Devido à dinâmica das baterias e painéis fotovoltaicos quanto aos efeitos da temperatura, a realização de uma simulação térmica desses componentes é de grande importância durante as fases de desenvolvimento, pois podem reduzir custos de testes com componentes físicos e ajudar a prever perfis de temperatura em

componentes críticos em um ambiente dinâmico. No entanto, a maioria dos trabalhos que abordam simulações térmicas carecem de gerenciamento térmico ativo e não se integram aos modelos elétricos. Além disso, muitas interações imprevisíveis entre o hardware podem ser detectadas e muitos parâmetros, como ruído de sinal, queda de sinal, atraso e outros, podem ser medidos. Não apenas o hardware real pode fornecer essa resposta única, mas também pode ser testado em diferentes condições de órbita que estão sendo reproduzidas virtualmente. No contexto das limitações dos nanossatélites em termos de colheita e armazenamento de energia, as estratégias de escalonamento podem proporcionar às missões uma melhor qualidade de serviço, quer como ferramenta de planeamento – escalonamento *off-line* – quer em tempo real – algoritmo on-line, incorporado no satélite. Com base nesses trabalhos anteriores, esta pesquisa aprimora o algoritmo de escalonamento proposto, reformulando o cálculo de prioridade das tarefas do satélite e alterando sua relação com o aquecedor. Além disso, a estratégia de controle é simulada em um ambiente embarcado com muito mais semelhança com o aplicativo final, no qual a capacidade e o desempenho em tempo real da estratégia de controle podem ser acessados. O algoritmo embutido é baseado no problema da mochila 0-1, que fundamenta a formulação anterior e o presente trabalho.

Objetivos

Considerando o contexto descrito, este trabalho tem como objetivo principal desenvolver com sucesso uma configuração de *framework hardware-in-the-loop* para testes de algoritmos de escalonamento de tarefas embarcados. Os objetivos específicos desta investigação são, nomeadamente: identificar os principais aspectos do escalonamento de tarefas impulsionado pela coleta e gerenciamento de energia em nanossatélites, juntamente com o estado da arte, melhorar o algoritmo de agendamento baseado em prioridade baseado em energia introduzido por trabalhos anteriores e validar a solução proposta em um ambiente embarcado, ou seja, em uma configuração *hardware-in-the-loop*.

Metodologia

Esta seção apresenta as equações relevantes para o modelo e os algoritmos em estudo, bem como a descrição da configuração e configuração dos testes de *hardware-in-the-loop*. O modelo é usado para simular a dinâmica térmica e elétrica de um nanossatélite e combina modelos de órbita, atitude, irradiância, térmica e elétrica, incluindo circuitos equivalentes que foram usados para representar a bateria e os painéis solares. O modelo também usa uma arquitetura diretamente acoplada. Nesta arquitetura, as tensões de operação dos painéis solares, bateria, carga e aquecedor são quase as mesmas e são fortemente influenciadas pela corrente consumida pela carga e pelo aquecedor. Dado um valor inicial de estado de carga, tensão da bateria, temperatura, inclinação da órbita e altitude, os parâmetros do CubeSat são calculados para cada iteração, e o aquecedor e a potência de carga são calculados considerando um possível controlador. Neste trabalho, todas as tarefas são periódicas, portanto, seus prazos de entrega são sempre os mesmos de seus últimos prazos. Tarefas também são consideradas preemptivas, e possuem a mesma importância, não possuindo nenhum valor intrínseco que torne a execução

de uma tarefa mais valiosa que outra, deixando esta decisão inteiramente para a estratégia de controle. Em sistemas de computação distribuída, encontrar a melhor forma de atribuir prioridades é de grande importância na implementação desses sistemas. A primeira parte da estratégia utilizada compara, para cada tarefa, o tempo de computação com o tempo executado para saber que uma tarefa terminou, em seguida verifica o tempo já atingido o prazo ativo da tarefa atualmente, e se tiver, calcula o novo prazo como. Por fim, a prioridade da tarefa é calculada, sendo mantida em um valor baixo caso já tenha sido executada para o deadline ativo atual. O objetivo do algoritmo de escalonamento é buscar a quantidade ótima de energia que as tarefas dos satélites devem consumir para atingir a geração máxima de energia. Quando não está em uma situação de eclipse, esse consumo ideal de satélite é calculado usando a energia do painel solar com o algoritmo de rastreamento de ponto de potência máxima e, quando está, essa quantidade ideal é mantida em um valor fixo para evitar que o algoritmo MPPT se mova erraticamente. Uma vez calculada a potência ideal, o algoritmo precisa descobrir qual subconjunto de tarefas se ajusta melhor a essa meta de consumo de energia sem excedê-la. Uma solução para este problema é formulada como um problema da mochila 0-1. A solução para cada iteração da simulação ditará quais tarefas permanecerão ativas ou não ao longo de sua duração. O algoritmo de programação dinâmica que foi usado para resolver este problema tem uma complexidade de tempo e espaço de $O(n \cdot W)$. Com o conhecimento de quais tarefas permanecem ativas, é calculada a potência de carga consumida pelas tarefas. O consumo de energia das tarefas é então alimentado no modelo para calcular seus parâmetros, afetando diretamente o ponto de operação dos painéis solares e consequentemente sua energia gerada. Implementar a estratégia de controle apresentada em um microcontrolador é uma ótima maneira de avaliar ainda mais a viabilidade de suas capacidades de tempo real de uma forma muito mais próxima da situação do cenário real. Nesta configuração, as variáveis do modelo e do sistema são calculadas e alimentadas ao microcontrolador em formato digital ou analógico e são usadas por este último para gerar os dados de controle necessários. O computador é responsável por calcular as dinâmicas do modelo, que são enviadas ao microcontrolador via comunicação serial, e a saída final da estratégia de controle, ou seja, o status ativo das tarefas é enviado de volta ao computador, formando um *loop* fechado. Para satisfazer os requisitos de tempo real do problema, todas as comunicações e cálculos devem ocorrer dentro do intervalo de tempo da simulação de 1 segundo.

Resultados e Discussão

O *framework* termoelétrico integrado foi implementado no MATLAB, e a configuração utilizada neste trabalho é baseada no CubeSat FloripaSat-I 1U, lançado em dezembro de 2019. Sua órbita é quase circular e síncrona ao Sol (SSO), representada como perfeitamente circular e com inclinação de 90º. As diferentes políticas de prioridade introduzidas foram utilizadas nas condições mencionadas do FloripaSat-I para avaliar o impacto no número de prazos perdidos. O número total de passos de tempo foi 70024, e cada passo representa 1 segundo, totalizando 12 órbitas. Cada política de prioridade foi tratada como um cenário diferente, totalizando 6 cenários. Com relação à temperatura de operação da bateria, é possível perceber que todas as simulações tiveram desempenho extremamente semelhante, apesar da diferença nas políticas de prioridade. Quanto ao desempenho do MPPT, ao calcular o coeficiente de correlação de

Pearson entre a potência desejada e a real gerada pelos painéis solares ao longo da simulação de cada cenário, foi possível observar que o desempenho geral foi semelhante, e a diferença nas políticas de prioridade não influenciaram de forma significativa os resultados individuais. Para tentar melhorar a temperatura operacional da bateria e a quantidade de energia coletada, a estratégia de controle foi alterada, entregando o restante da saída do algoritmo MPPT ao aquecedor após alocar as tarefas após cada iteração. Foi possível observar uma melhora na correlação, indicando que o algoritmo MPPT teve melhor desempenho. Quanto à temperatura de operação da bateria, a temperatura mínima e média alcançada durante a simulação aumentou. Os experimentos foram conduzidos usando a configuração de *hardware-in-the-loop* descrita anteriormente na seção de modelagem do problema. Depois de executar o experimento com a configuração de *hardware-in-the-loop*, os resultados foram comparados com a simulação e a correção do algoritmo implementado na linguagem-alvo foi verificada. Uma análise de desempenho foi realizada variando o número de tarefas de 1 a 7, em diferentes execuções da configuração e medindo o tempo necessário para concluir uma iteração de loop completo do sistema para várias etapas de tempo. Os resultados mostram que o tempo médio que o microcontrolador leva para receber, computar e responder ao computador teve um aumento linear com base no número de tarefas. É possível concluir que a implementação satisfez os requisitos de tempo real do sistema.

Considerações Finais

Neste trabalho, um algoritmo de escalonamento baseado em energia foi testado usando uma estrutura termoelétrica integrada capaz de representar a dinâmica de componentes importantes como a bateria e os painéis solares. Foi mostrado que ao configurar a função de prioridade do algoritmo o número de deadlines perdidos pode ser reduzido para um conjunto específico de tarefas, e, ao gerenciar a alocação de carga de energia para o aquecedor, foi possível melhorar a energia captada pelo algoritmo, como mostrado pelo aumento da correlação entre a saída de energia real e ideal dos painéis solares. Essas melhorias podem levar a uma redução dos efeitos indesejados de baixa temperatura na bateria. A estrutura do modelo integrado foi então expandida para uma configuração de hardware no loop, que foi usada para implementar o algoritmo de escalonamento na linguagem de destino. Um ciclo de computação de 1 segundo foi usado, e o comportamento e as capacidades de tempo real do algoritmo foram validados, e os cálculos da estratégia de controle foram executados pelo microcontrolador em menos de um quarto de segundo.

Palavras-chave: Hardware-in-the-loop. Escalonamento de tarefas. CubeSats.

ABSTRACT

Technological improvements and the miniaturization of components enabled a reduction on the average size of launched satellites, which started to contain many “commercial off the shelf” components, that are cheap, demand low power, and are widely available. This trend culminated in the development of the CubeSat standard, and in the fact that, most satellite launches are in the “small-satellite” category (up to 180kg). As consequence of the reduced dimensions of satellites, efforts arise on pursuing means to optimize energy management and consumption. Such efforts include evaluating which electrical power system architecture provides best overall efficiency, that often includes maximum power point tracking (MPPT) for harvesting energy using solar panels, which correspond to the most common primary source of power on small satellites. In this context, this work proposes the use of a 0-1 knapsack-based task scheduling strategy aiming to maximize energy harvesting. An integrated thermal-electrical nanosatellite framework was used to test various priority functions and different heater activation strategies using tasks parameters of FloripaSat-I and other randomly generated cases. Results shown that a saturating priority function presented the least amount of deadline losses. Although the different priority functions did not present significant influence in battery temperature, by allocating the remaining resources directly to the heater after selection phase of the tasks, an increase in correlation between ideal and achieved power generation from solar panels was obtained, and battery temperature operated closer to desired temperature range throughout the simulation. The control strategy was later implemented on an embedded environment, configuring a hardware-in-the-loop (HIL) simulation. The correctness of the algorithm was verified, along with its capabilities to fulfill the real time constraints. A speed analysis was conducted and verified a linear impact of the number of tasks on the computation time.

Keywords: Hardware-in-the-loop. Task scheduling. CubeSat.

LIST OF FIGURES

LIST OF BOARDS

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

COTS	Commercial off-the-shelf
Cal Poly	California Polytechnic State university
EPS	Electrical Power System
HIL	Hardware-in-the-Loop
MPP	Maximum Power Point
I-V	Current-Voltage
P-V	Power-Voltage
I_{cc}	Short circuit current (solar panel)
V_{oc}	Open circuit voltage (solar panel)
MPPT	Maximum Power Point Tracking
P&O	Perturb and Observe
VLDO	Very Low Dropout
Li-ion	Lithium ion
LiPo	Lithium Polymer
DoD	Depth of Discharge
PC	Personal Computer
UAV	Unmanned Air Vehicle
HILS	Hardware-in-the-Loop Simulation
MCU	Microcontroller Unit
USB	Universal Serial Bus
SoC	State of Charge
ET	Elapsed Time
ERT	Estimated Response Time
NTCT	Nearness to Completion Time
OBDH	On-Board Data Handler
TTC	Telemetry, Tracking and Command

SSO	Sun-Synchronous Orbit
CCS IDE	Code Composer Studio Integrated Development Environment
TI	Texas Instruments
CPU	Central Processing Unit
USART	Universal Synchronous Asynchronous Receiver Transmitter

CONTENTS

1 INTRODUCTION

Artificial satellites distinguish themselves from natural satellites, such as the Earth's Moon, in the sense that they are objects which were intentionally placed into orbit. They can serve a variety of purposes, from communication and amateur radio to earth science and atmospheric research.

1.1 NANOSATELLITES AND THE CUBESAT STANDARD

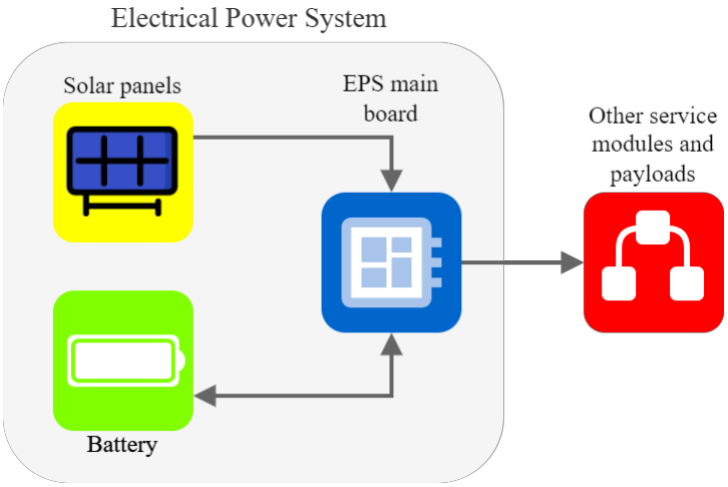
Since the launch of the Sputnik 1 in 1957, hundreds of satellites have been launched. Technological improvements and the miniaturization of components enabled a reduction on the average size of launched satellites, which started to contain many "commercial off-the-shelf" (COTS) components, that are cheap, demand low power, and are widely available (POGHOSYAN and GOLKAR, 2017). This trend culminated in the CubeSat concept. The CubeSat standard was created by the California Polytechnic State university (Cal Poly), and by the Stanford University in 1999 and defined as 1U the 10 cm x 10 cm x 10 cm cube with a maximum of 2kg (CAL POLY SLO and JOHNSTONE, 2020). Since then, most satellite launches are in the "small-sat" category (up to 180kg) and engulf nanosatellites (1 to 10kg) and the CubeSat standard (POGHOSYAN and GOLKAR, 2017) (NASA, 2021).

Typically, a nanosatellite has different subsystems that vary according to mission goals. The most common ones are listed below:

- Structure,
- Data handling,
- Telemetry,
- Power (energy management, acquisition, and storage),
- Attitude control,
- Thermal control.

The subsystem responsible for managing power distribution, often called Electrical Power System (EPS) needs to operate in robust, reliable, and stable manner. The EPS is usually comprised of a primary power source (mainly solar cells and panels) and an energy storage unit, usually a battery (NASA, 2021). Figure 1 shows a representation of a common Electrical Power System.

Figure 1 - EPS overview where Solar panels and Battery are connected to the EPS main board which serve other service modules.



Source: Author.

1.2 MOTIVATION

The power limitations derived from solar panels size restriction hinders nanosatellite's ability to function. When the collected energy is greater than the subsystems consumption, it shall be stored in the battery, and, on the other hand, if the subsystems demand more power than the source is currently producing, the battery shall suffice this difference. This management shall also take into consideration the battery charge capacity, to prevent its depletion. This conditions, among other scenarios that the EPS must be able to handle, can benefit from a system that takes the power budget in consideration when scheduling the satellite tasks execution (WANG, ZHU, *et al.*, 2015).

One effective method to model and simulate an engineering system, that consists of the combination of computer simulation and hardware in a single platform, is called hardware-in-the-loop (HIL). Typically, this method is the last stage of the testing and integration process (CORPINO and STESINA, 2014), and can be used to, among other things, to test different tasks execution control strategies.

1.3 OBJECTIVES

Considering the described context, this work aims to, as main objective, to successfully develop a hardware-in-the-loop framework configuration for testing of on-board task scheduling algorithms.

The specific objectives of this research are, namely:

- Identify the main aspects of task scheduling driven by energy harvesting and management in nanosatellites, along with the state-of-the-art.
- Improve the priority based on-board energy-driven scheduling algorithm introduced by Vega Martínez (2022).
- Validate the proposed solution on an embedded environment, i.e., in a hardware-in-the-loop setup.

1.4 PUBLISHED WORKS

The results of this work were presented at the 5th IAA Latin American CubeSat Workshop and 3rd IAA Latin American Symposium on Small Satellites, as an article:

BERNARDO, V.P.; BEZERRA, E.A.; SEMAN, L.O.; BORBA, R. Hardware-in-the-loop simulation of an on-board energy-driven scheduling algorithm for CubeSats. Proceedings of the 3rd IAA Latin American Symposium on Small Satellites.

1.5 CHAPTERS DESCRIPTION

This master's thesis is organized in chapters. Chapter 2 covers the necessary background involving nanosatellites electrical power systems and its main components, i.e., solar panels and batteries, while discussing their main areas of concern during development and operation. Chapter 3 presents the model utilized in this work along with relevant equations and diagram to explain the setups involved in the conducted experiments. Chapter 4 contains the results obtained in both the simulation and hardware-in-the-loop configurations, and final considerations are made on Chapter 5.

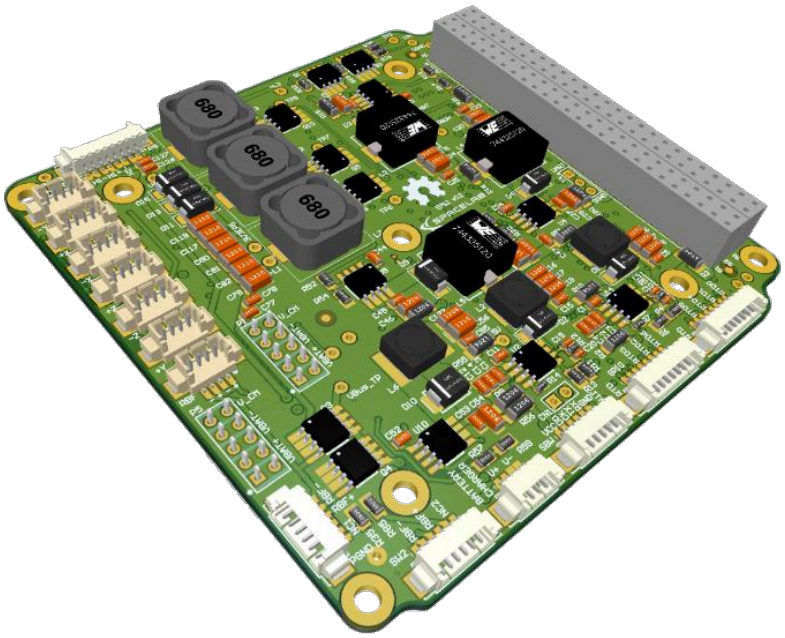
2 LITERATURE REVIEW

To better understand the impacts that a real-time scheduling algorithm can drive in a small spacecraft environment with limited capabilities, this Section will explain the main topics which concerns such an application. These topics are divided in three main sections, the Electrical Power System and its physical characteristics, tests and verification using hardware-in-the-loop, and task scheduling in nanosatellites.

2.1 ELECTRICAL POWER SYSTEM

The Electrical Power System (EPS) is an essential subsystem of a nanosatellite that shall harvest energy from the surrounding space environment, store, and ultimately deliver power to other subsystems. This harvesting process can take many forms, such as using solar heat in a thermal power cycle or a chemical process, but the most common method is through the photo-voltaic effect utilizing solar panels (GALATIS, GUO and BUURSINK, 2017). Figure 2 shows an EPS printed circuit board developed at the Federal University of Santa Catarina's Spacelab for the FloripaSat-II, a student's CubeSat project.

Figure 2 - EPS printed circuit board render from Spacelab's FloripaSat-II



Source: UFSC Spacelab[1], 2021.

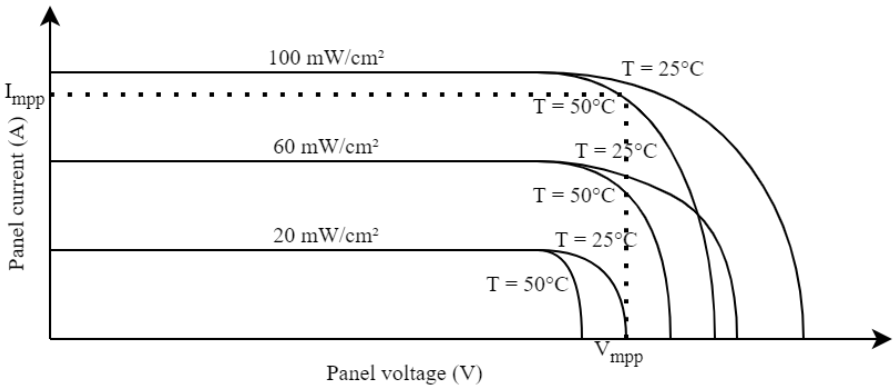
2.1.1 Solar Panel

A solar cell is a device that converts sunlight into electrical energy through a process called the photovoltaic effect. It is made up of a semiconductor material, typically silicon, which absorbs photons from the sun and releases electrons. These electrons flow through metal contacts on the cell, generating a flow of electrical current Solar cell are connected in series and parallel arrangements to form solar panels. Small and CubeSats typically use multijunction

[1] Available at the public repository of the open-source CubeSat mission GOLDS-UFSC: https://github.com/spacelab-ufsc/eps2

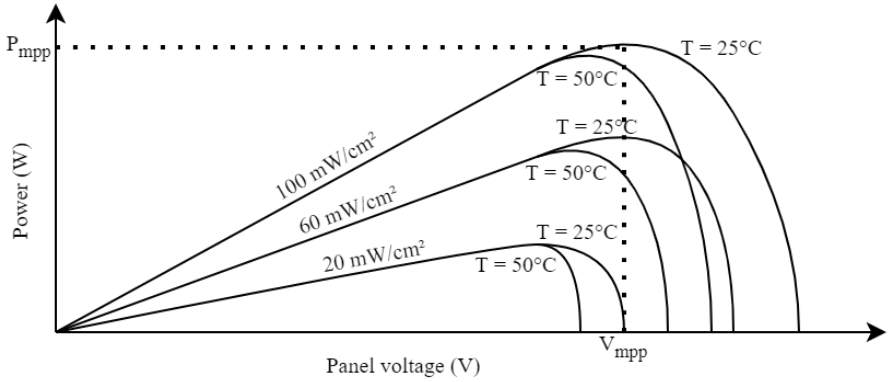
solar cells with better performance (up to 29% to 32%) than the ones commonly used in terrestrial applications (NASA, 2021). Solar panels are characterized by two curves, a current-voltage (I-V) and a power-voltage (P-V) curve. When operating a solar panel, to produce the highest possible amount of energy, it is desirable to stay as close as possible to a point known as the Maximum Power Point (MPP). This can be achieved by controlling the panel voltage to stay at the maximum power point voltage V_{mpp} (SLONGO, MARTÍNEZ, *et al.*, 2018). Figures 3 and 4 shows how the typical curves of a solar panel looks and where the MPP is. It is possible to see the effects of irradiance and temperature changes on the overall panel current output. The higher the temperature, the smaller the open-circuit voltage gets, and the higher the irradiance, the higher the short-circuit current gets.

Figure 3 - Example I-V curve.



Source: Author.

Figure 4 - Example P-V curve.



Source: Author.

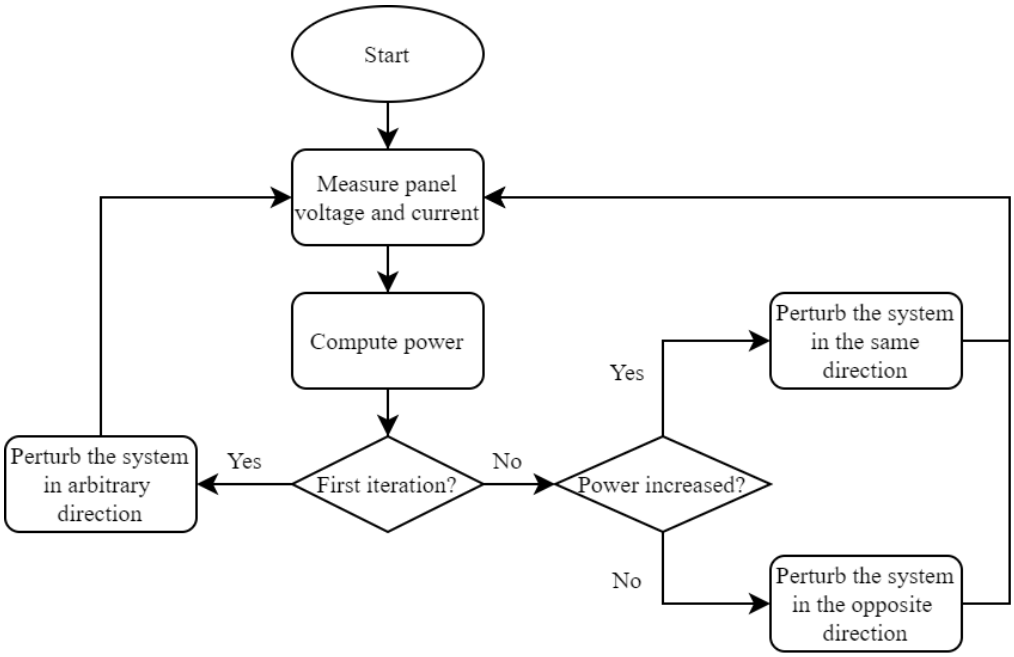
In the current and voltage curve depicted in Figure 3, it's possible to see the highest current that the panel can provide, known as short circuit current I_{cc} and the maximum voltage, known as open circuit voltage V_{oc}. The behavior of the I-V curve changes depending on the connected voltage, and external factors like temperature, irradiance level and ageing of the panels. The P-V curve is derived from the I-V curve (SLONGO, MARTÍNEZ, *et al.*, 2018).

These curves demonstrate that solar cells are very versatile since they can operate from open circuit to short circuit (FRÖHLICH, BEZERRA and SLONGO, 2015).

2.1.1.1 Maximum Power Point Tracking

Operating the solar panel in the maximum power point voltage results in transferring the maximum amount of available power to the system. The first MPPT targeted artificial satellites and later was employed on other systems that operated on the shore. This tracking can be implemented with different technics, such as Incremental Conductance, Open Circuit Voltage, and Perturb and Observe (P&O) (FRÖHLICH, BEZERRA and SLONGO, 2015). The Perturb and Observe method specifically is widely used because of the ease of implementation and low costs and requirements. The basic idea of the P&O method is to continuously adjust the operating voltage of the solar panel and observe the changes in the current and power output. The algorithm then adjusts the operating voltage in the direction that increases the power output, repeating this process until it converges on the maximum power point. In practice, the algorithm starts by setting the operating voltage to a known initial value, then perturbs the voltage in small increments. If the power output increases, the voltage is perturbed further in the same direction, and if it decreases, the voltage is perturbed in the opposite direction. When temperature and irradiance conditions are constant or change slowly, the method oscillates close to the Maximum Power Point (PIEGARI and RIZZO, 2010). Figure 5 exemplifies the flow logic of a typical Perturb and Observe algorithm for maximum power point tracking.

Figure 5 - Perturb and observe algorithm diagram representation.



Source: Author.

As electronic components reduces in size, and embedded systems power consumption decreases, the work of Fröhlich, Bezerra e Slongo (2015) aimed to discover if MPPT circuits were still advantageous in low power applications considering their own consumption added to

the system, and they found that a directly coupled circuit was able to extract more energy and is more efficient than using a dedicated MPPT circuit.

One study made by Kessler Slongo, Vega Martínez, *et al.* (2020), conducted a three-orbit experiment comparing four EPS architectures: directly coupled architecture, very low dropout (VLDO) voltage regulator architecture, maximum power point tracking (MPPT) architecture with an integrated circuit, and MPPT architecture with a discrete boost regulator.

The directly coupled circuit requires few components, and thus reduces energy losses when comparing to other architectures, but the solar panels are not actively controlled, and may not operate on their maximum power point continuously. The very low dropout (VLDO) architecture uses a high-efficiency linear regulator between the solar panels and the rest of the circuit (battery and load), and while it still may be considered a directly coupled architecture, the relation between the battery and the solar panels voltage changes compared to the former architecture. In the MPPT architecture with a discrete boost regulator, the battery voltage is used as input, and the solar panels voltage is controlled based on a Perturb and Observe algorithm performed by the EPS microcontroller. The MPPT architecture with dedicated integrated circuit is similar to the last one mentioned, as it also uses a boost regulator between the battery and solar panels, but the MPPT control is done by the dedicated circuit, which measures the panels voltage and current, reducing the processing requirements for the microcontroller while not allowing for changes or improvements on the control strategy.

The study concluded that MPPT boost regulator architecture harvested more energy, but the VLDO voltage regulator architecture presented the best overall efficiency, and mathematically demonstrated that the latter architecture, along with the directly coupled architecture, could benefit from an efficiency perspective from control of tasks execution.

Despite the Sun being always present in the Earth's orbit, satellites may not receive continuous irradiance level along the entirety of the orbit. This along with other factors related to conversion physics makes this source of energy unstable. Therefore, energy storing devices are extremely common (KESSLER SLONGO, VEGA MARTÍNEZ, *et al.*, 2020).

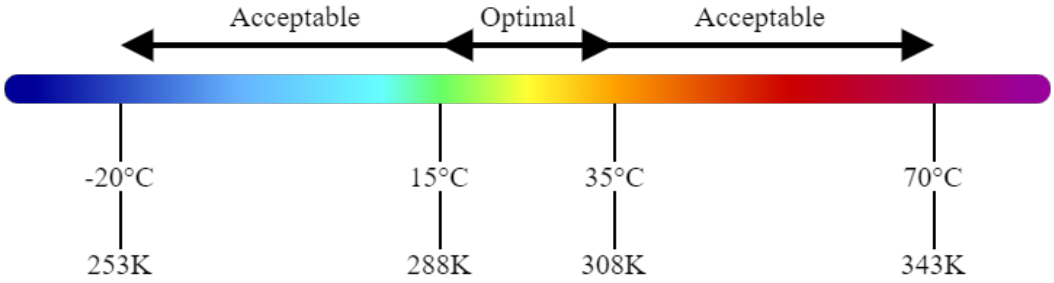
2.1.2 Battery

There are two main types of batteries, primary-type batteries, which are not rechargeable and are used as the primary source of power in short duration missions, and secondary-type batteries, which are rechargeable and among other type of batteries, comprehend the largely used lithium-based batteries that are now present in most used electronic devices, and have also been extensively used on small space applications. These secondary type batteries are usually connected to a primary energy source (e.g., the solar cells and solar arrays). Current state-of-the-art energy storage systems use lithium ion (Li-ion) or lithium polymer (LiPo) secondary-type batteries. Repeated charging cycles of the battery will result in ageing, reducing the energy storing capacity of the battery (NASA, 2021).

Battery cycle life is a term used to define the number of charge and discharge cycles that a battery can perform while maintaining a given percentage of its initial capacity. This percentage is usually 80% and depends on many factors regarding its operation conditions, such as charge and discharge profiles, temperature (both high and low) and the depth of discharge

(DoD), meaning there is not a single cause to ageing and is a result of different factors and their interactions (PALACÍN, 2018).

Figure 6 - Optimal and acceptable temperature ranges for lithium-ion batteries.



Source: Adapted from Ma, Jiang, *et al.*, 2018.

Lithium-ion batteries optimal temperature ranges from 15°C to 35°C, and the acceptable temperature ranges from -20°C to 70°C. Figure 6 depicts the aforementioned ranges of optimal and acceptable temperature for Lithium-ion batteries. Low temperature effects are much more related to the environment and usually happens in high-latitude areas and in space and the effects involve slow down chemical reaction activity and charge-transfer velocity, resulting in reduction of energy and power capabilities. High temperature effects are not limited to environment and happen in a broader range of applications. It can happen due to internal temperature rising caused by high current states during charging or discharging at fast rates. These conditions will also lead to a degradation in capacity and power. Generally, lithium losses and active material degradation will result in a capacity reduction, and the increase of internal resistance causes loss of power (MA, JIANG, *et al.*, 2018). These effects are summarized in Table 1.

Table 1 - Temperature cause and effect relations at low and high temperature.

Effect	**Low temperature cause**	**High temperature cause**
Decrease in capacity	Lithium plating	Thermal-induced lithium decomposition
Decrease in power	Increased electrolyte viscosity which rises internal resistance	Ohmic heating which rises internal resistance

Source: Adapted from Ma, Jiang, *et al.*, 2018.

2.2 TEST AND VERIFICATION OF NANOSATELLITES

With the increased use of small satellites, a high failure rate has been noticed (BOUWMEESTER and GUO, 2010). Among the causes are hardware and software design mistakes along with failures in the integration process. Effectively predicting and testing the system can help discover and avoid failures that would otherwise compromise the mission.

As previously mentioned, hardware-in-the-loop (HIL) is the final stage of the testing and integration process (CORPINO and STESINA, 2014). Some of the important stages of testing the elements of a system according to Eickhoff (2009), are:

- Algorithm in the loop is the initial stage in which the physical behavior of the system is modeled (via software or test stand) and the control algorithms are tested.
- Software in the loop is the next part of the process, in which the algorithms are coded in the target programming language and once again tested in the modeled system.
- Hardware in the loop is the final step, in which the software is loaded into the target computer to be tested in the modeled system, and later into the real system.

2.2.1 Software based simulations

Considering the dynamics of batteries and photovoltaic panels regarding the effects of temperature and knowing that both are commonly present in small satellites, a thermal simulation of these components is of great importance during development phases and can reduce costs of testing with physical components, since batteries and solar panels represents a significant portion when compared to other electronic components (VEGA MARTINEZ, FILHO, *et al.*, 2021).

Thermal simulations can help predict temperature profiles in critical components on a dynamic environment (CORPINO, CALDERA, *et al.*, 2015) (BONNICI, MOLLICONE, *et al.*, 2019), but lack active thermal management and do not integrate with the electrical models. In Aung, Soon, *et al.* (2020) a State-of-Charge and State-of-Health estimation system is presented, but the typical temperature profiles found in orbit were not considered.

One work presented an integrated thermal-electrical framework in which is possible to control a heater and see its effects on the battery temperature, while accessing the battery charge status (VEGA MARTINEZ, FILHO, *et al.*, 2021). The model takes in consideration orbit, irradiance, attitude, thermal, electrical and battery models for CubeSat mission analysis. This approach allows the user to test and analyze different control methodologies to try and maintain temperature levels within a desired range but is not capable of demonstrating how the control strategy will perform on the final controller unit, that likely has extensive resources and performance constraints compared to general purpose PCs where this type of simulation is usually ran on.

2.2.2 Hardware-in-the-loop simulation

This hybrid architecture composed of hardware and software is especially useful when testing systems that operate in conditions that are hard to replicate in a laboratory. One of the benefits of this verification method is that including the real hardware in the simulation simplifies modeling activities, since modeling the controller itself is not simple, as most commercial off-the-shelf components, which are largely used in electronic products and have become extremely popular in small satellites, do not have readily available models. Also, many unpredictable interactions between hardware can be detected, and many parameters such as signal noise, signal dropout, lag and others can be measured. Not only the real hardware will

provide this unique response, but it can also be tested in different orbit conditions that are being virtually reproduced (CORPINO and STESINA, 2014).

One work by Gans, Dixon, *et al.* (2009) enforced the necessity of tests involving the real hardware, which was stated to reflect aspects that a simulation environment cannot, such real sensor noises and actuator lag, whilst testing unmanned air vehicles (UAVs). A novel hardware in the loop simulation (HILS) was proposed using virtual reality to produce realistic scenes and test vision-based control of UAV's, which could also be mounted in a wind tunnel for further testing of control surfaces. An article by De Carufel, Martin E Piedbœuf (2000) encountered scenarios that were not possible to simulate using software since the accuracy of some models were yet to be validated. By using a hardware-in-the-loop simulation it was possible to investigate the control of a space robot and compare a position-based control against a newer acceleration control approach, which was later preferred. These works all reinforce the advantages of conducting hardware-in-the-loop simulations in space applications.

In their work (CORPINO and STESINA, 2014) were able to verify the functional requirements of the e-st@ar-1 CubeSat using a hardware-in-the-loop-simulation and obtained satisfactory results regarding its behavior. They stated the value of this type of simulation for small satellite verification, and how it can help both with time and costs reductions of the development phase while increasing mission reliability. Similar works were conducted in the field of small satellites, such as the one by (WU and VLADIMIROVA, 2008) that used a hardware-in-the-loop simulator to demonstrate a satellite sensor network concept with a simulator capable of emulating orbit dynamics in Low Earth Orbit.

The work by (FINNSET, K. RAO and ANTONSEN, 2006) carried out closed loop hardware-in-the-loop simulations using a Microchip PIC18F452 and MATLAB in a standard PC to simulate the attitude control system of the ESMO satellite and test the dynamics of the system. In this work, the MCU and the computer used serial communication complying to the standard RS232 protocol through an USB cable. The MCU was used to implement the attitude control algorithm whilst the computer was responsible for simulating the attitude dynamics of the satellite, receive the actuator signals from the MCU and send measurements signals back, forming a closed loop. As part of the results, it is stated that the controller achieved satisfactory performance in a real time environment.

2.3 TASK SCHEDULING IN NANOSATELLITES

As discussed in Chapter 1, due to their small dimensions, nanosatellites have limitations in terms of energy harvesting and storing. In this sense, scheduling strategies can provide missions with a better quality of service, either as a planning tool – offline scheduling – or in real time – online algorithm, embedded in the satellite. This Chapter will present related works in the field of task scheduling in nanosatellites, and important definitions regarding the knapsack combinatorial optimization problem, used in this research.

One approach is proposed by Pang *et al.* (2014) for dealing with power and bandwidth limitations on nanosatellites. The study is conducted upon a robust formulation based on convex optimization and application of cutting-plane algorithm. The formulation is to be applied in the field of nanosatellite swarm for synthetic aperture radar.

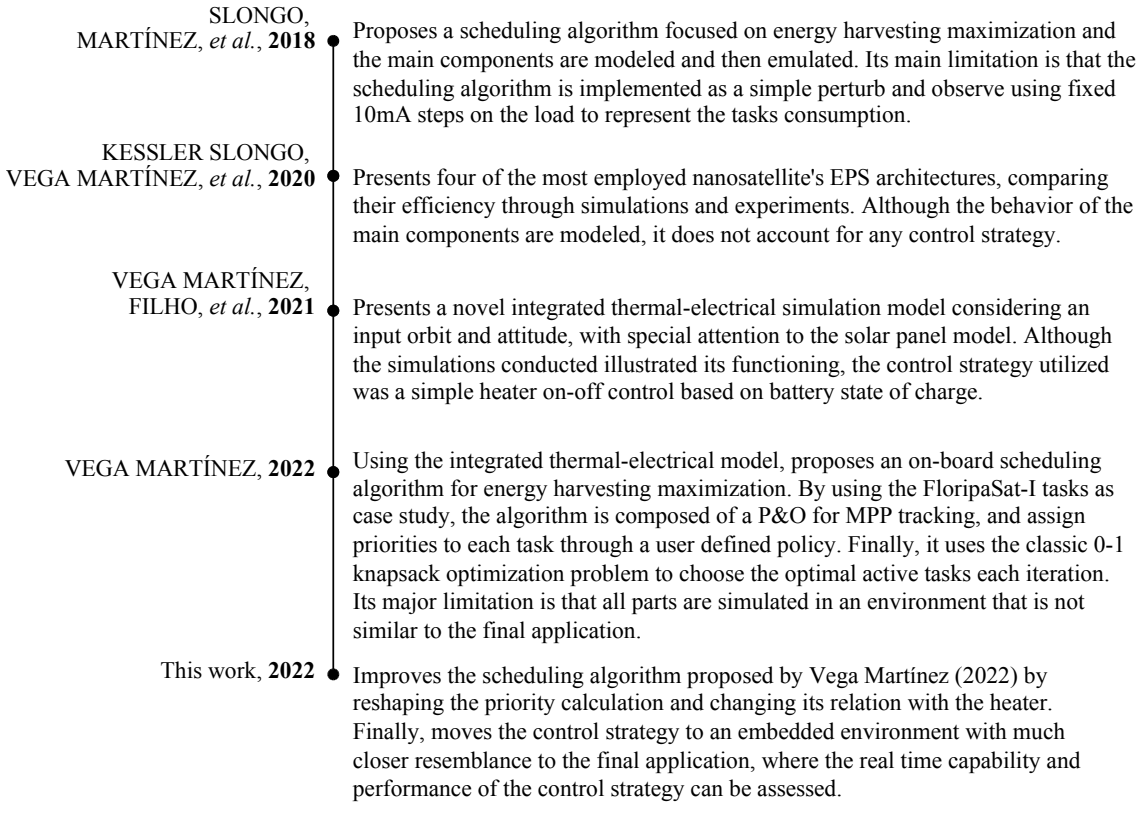
Online task scheduling algorithms based on "Earliest Deadline", "Earliest Arrival Time" and "Minimum Slack" ordering policies are proposed by Dobiáš, Casseau and Sinnen (2020). Their approach considers energy consumption constraints, taking advantage of multicore processors in CubeSats through software and proposes to increase the system's fault tolerance. The presented algorithms aim to minimize the "rejection rate" of tasks, i.e., the ratio between arrived and lost deadlines. This research does not address, in a quantified manner, power availability from primary and secondary sources of energy – i.e., solar panels and batteries.

Zhang, Behbahani and Eltawil (2021) address the inter-satellite communication problem between CubeSats. They propose to maximize transmission data rate based on online scheduling of operation frequencies and constrained by transmit power.

An offline mixed integer formulation is described by Rigo, Seman, *et al.*, (2021) for optimizing quality of service in nanosatellites missions through task scheduling. The model is energy constrained, provides exact optimal solutions, and implements Fuzzy constraints for considering battery usage. Only depth of discharge is considered for preserving battery lifetime, temperature analysis or constraints are not included.

Regarding online embedded optimization constrained energy availability, Martínez and Slongo, within multiple works, research and analyze different MPPT architectures and task scheduling technics. The timeline (Figure 7) below shows the previous works that were conducted which this work is based upon:

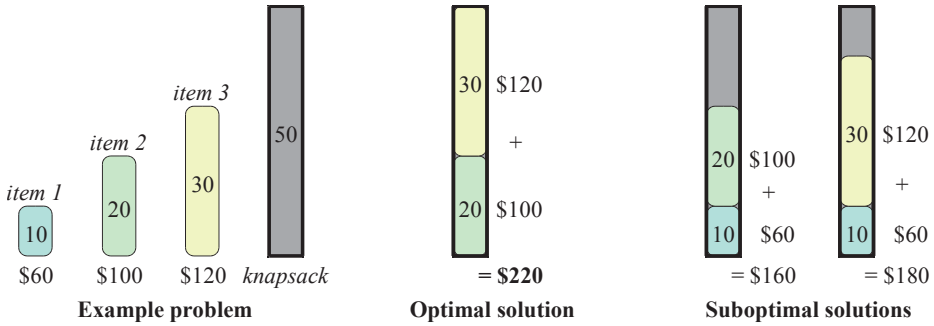
Figure 7 - Timeline of previous works.



Source: Author.

The 0-1 knapsack problem, which underlies Martínez (2022) formulation and the present work, is a combinatorial optimization problem to determine the best solution to fill a backpack that supports at maximum a certain weight. The backpack is filled with items, each having a specific value and weight. The goal is to maximize the total value of the items inside the backpack without exceeding the maximum weight (FRÉVILLE and HANAFI, 2005). In the example displayed in Figure 8, the backpack (or knapsack) supports a weight of 50 and must be filled with the most valuable possible (optimal) combination of three items, each of it with its specific weight and value. The 0-1 version of the problem does not allow items to be fractioned.

Figure 8 – 0-1 knapsack problem example



Source: Adapted from Cormen, *et al.*, (2009).

In this literature review chapter, the key concepts and relevant previous works to the research topic were presented and discussed. The information presented serves as the foundation for understanding the problem and subsequent modeling in the next chapter, while also providing context for the proposed solution.

3 PROBLEM MODELING

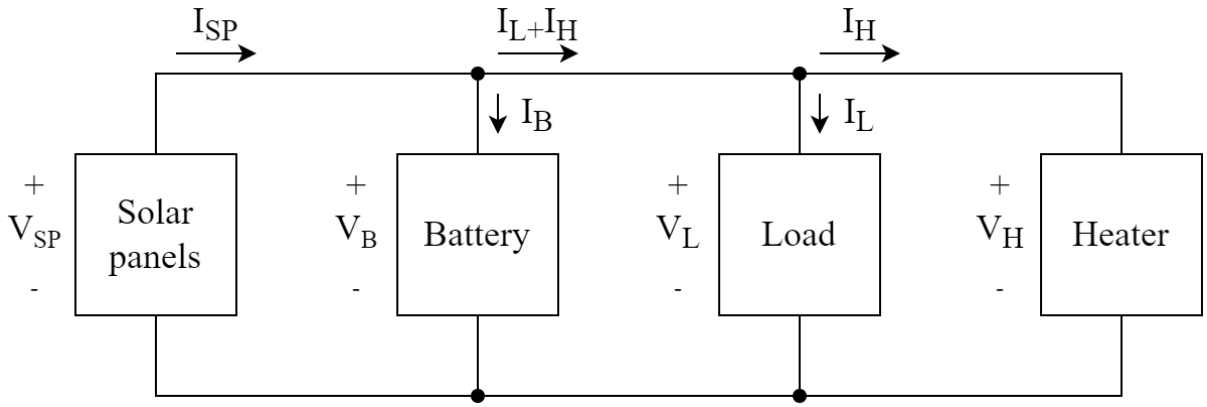
This Section presents the relevant equations to the model and the algorithms under study, as well as description of the hardware-in-the-loop tests configuration and setup.

The model used to simulate the thermal and electric dynamics of a nanosatellite is based on the integrated model of Vega Martinez, Filho, *et al.* (2021), which combines together orbit, attitude, irradiance, thermal, and electrical models, including equivalent circuits that were used to represent the battery and solar panels, and uses a directly coupled architecture, represented in Figure 9, as it allows different active thermal management strategies using a heater, and as stated in their work, does not present the overhead consumption of a MPPT architecture, and can benefit greatly from scheduling the tasks in order to alter the solar panels operating point. In this architecture the solar panels, battery, load, and heater operation voltages (V_{SP}, V_B, V_L and V_H respectively) are almost the same Equation (1), and the current drawn by the load and heater heavily influence this common voltage operating point, as per Equation (2).

$$V_{SP} \approx V_B = V_L = V_H \tag{1}$$

$$I_B = I_{SP} - I_L - I_H \tag{2}$$

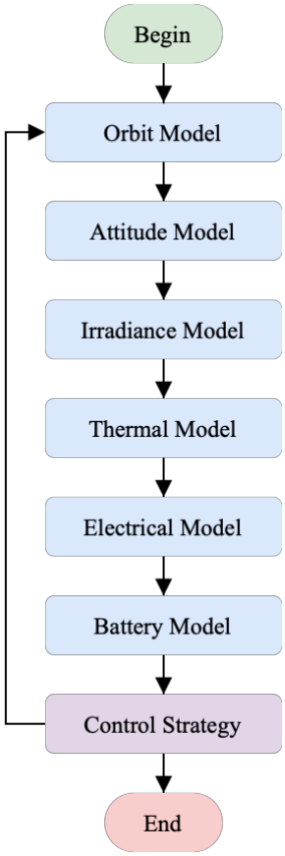
Figure 9 - Directly coupled architecture simplified circuit representation.



Source: Author.

Given an initial SoC value, battery voltage, temperature, orbit inclination and altitude, the CubeSat parameters are calculated for each iteration, and the heater and load power are calculated considering a possible controller. These steps performed by the simulation framework are represented by the flowchart in Figure 10.

Figure 10 - Integrated simulation model fluxogram.



Source: Author.

3.1 TASKS SCHEDULING

The proposed solution in this work revolves around performing a previous study and characterization of the satellite's tasks. This is necessary in order to use the proposed algorithm in orbit. The term task refers to a single or a group of actions that needs to be performed by the satellite during its mission, and can range from imaging, communications, maneuvering, among other activities. Tasks must be characterized according to their maximum computing time, submission and deadline times, preemptive characteristics, resource utilization (i.e. power consumption), and priority. Tasks can also be periodic, i.e., its submission time is exactly after the last deadline, repeating itself after a predetermined time interval, or sporadic, where the submission time is not known beforehand. Tasks can also demand a specific resource other than power from the satellite during its execution, such as transceivers and antennas, communication ports, sensors, etc. Different tasks can demand the same resource, and possible collisions must be handled by the scheduler.

In this work, a set of n tasks is represented by $J = \{1 \dots n\}$. Each individual task has three characteristics, a maximum computing time c_j, a period dl_j, and a power consumption r_j, all of which are immutable. Along with this, all the tasks considered can be preempted, which means once turned on, they shall be interrupted and resumed as needed, and if the total amount

of time it stays active equals the entire length of its computing time before reaching the deadline, the task can be considered as completed.

Along with this, tasks have a value assigned to them that determines which tasks shall be executed in each moment. This value is known as priority, and can be fixed, or change over time. The priority of a task j is represented by u_j. To further complement the model, each task also has a submission time s_j and a deadline d_j such that $d_j = dl_j + s_j$. There is also a_j which stores the current time a task stayed active since its last submission time passed, and a remaining time $rt_j = c_j - a_j$. The last two parameter used to describe the tasks are x_j, which represents whether a task is currently running, i.e., is on or off, and ex_j, which represents whether a task already finished at least once for the currently active deadline. Board 1 summarizes the relevant variables that characterizes the tasks as well as model variables relevant to the control strategy:

Board 1 - Tasks characterization variables

Notation	Description
n	Number of tasks. $n \in \mathbb{N}$.
P_L	Load power. Power consumed by the active tasks at a given time instant. $P_L \in \mathbb{R}^+$.
P_{SP}	Power generated by the solar panels in Watts. $P_{SP} \in \mathbb{R}^+$.
W	Satellite's optimal power consumption in Watts. Output of MPPT algorithm. $W \in \mathbb{R}^+$.
$W_{eclipse}$	Satellite's constant consumption goal during eclipse. $W_{eclipse} \in \mathbb{R}^+$
T	Number of simulation steps. $T \in \mathbb{N}$.
J	The set of tasks. $J = \{j \mid j \in \mathbb{N}, j \leq n\}$.
r_j	Power consumption of task j in Watts. $r_j \in \mathbb{R}^+$.
c_j	Computing time of task j in seconds. $c_j \in \mathbb{N}, c_j \leq T$.
dl_j	Period of task j in seconds. $dl_j \in \mathbb{N}, dl_j \leq T$.
u_j	Priority of task j. $u_j \in \mathbb{N}$.
s_j	Submission time of task j in seconds. $s_j \in \mathbb{N}, s_j \leq T$.
d_j	Deadline of task j in seconds. $d_j \in \mathbb{N}, d_j \leq T$.
a_j	Executed time of task j in seconds. $a_j \in \mathbb{N}, a_j \leq T$.
rt_j	Remaining time of task j in seconds. $rt_j \in \mathbb{N}, rt_j \leq T$.
x_j	On-off status of task j. $x_j \in \{0,1\}$.
ex_j	Already executed flag of task j. $ex_j \in \{0,1\}$.

Source: Author.

In this work, all tasks are periodic, so their submission times are always the same as their last deadlines. Tasks are also considered preemptive, and have the same amount of importance, having no intrinsic value that makes the execution of one task more valuable than another, leaving this decision entirely for the control strategy to decide based on whatever

metrics it uses to decide upon. In distributed computing systems, finding the best way to assign priorities poses great importance when implementing these systems.

The first part of the strategy used (represented by Algorithm 1) compares the computation time c_j with the executed time a_j to know that a task has ended, and if it has, sets the already executed ex_j value to 1. Then it checks if the time t already reached the task currently active deadline d_j, and if it has, calculates the new deadline as $d_j = d_j + dl_j$. Finally, the task priority is calculated, and is kept at a low value if it has already been executed for the current active deadline.

Algorithm 1: Dynamic priority calculation

```
1  for j = 0 ... n do
2      if c_j − a_j ≤ 0 then
3          x_j = 0
4          a_j = 0
5          u_j = 0
6          ex_j = 1
7
8      if t = d_j then
9          d_j = d_j + dl_j
10         ex_j = 0
11
12     if x_j = 1 then
13         a_j = a_j + 1
14
15     if ex_j = 1 then
16         u_j = 1
17     else
18         u_j = priority function (j)
```

Vanderster, Dimopoulos, *et al.*, (2009) presents a few common priority calculations policies used in distributed systems, some of which were used in this work. The first method of calculating priority is the Elapsed Time priority function (ET), which is calculated as follows (Equation 3) for each task $j \in J$:

$$u_j = 100\frac{t-\mathrm{s_j}}{dl_j} = 100\frac{t+\mathrm{dl_j}-\mathrm{d_j}}{dl_j} \quad (3)$$

This policy gives a normalized linear priority value from 0 to 100 between the last deadline (which is the same as the submission time) and the current active one. This amount of time between deadlines is the period of the task. The calculation can be improved by finding a critical point in time in which the task must become active and stay active until it reaches its deadline in order to be able to satisfy its completion. This can be achieved by assigning a sufficiently big priority value M to a task once it reaches this critical point in time so that no

other single task or combination of tasks takes priority over it. The priority calculation (which will be called Elapsed Time with M, or ET+M) then becomes (Equation 4):

$$u_j = \begin{cases} 100\frac{t+dl_j-d_j}{dl_j}, d_j - t > c_j - a_j \\ M \qquad\qquad\quad , d_j - t \leq c_j - a_j \end{cases} \tag{4}$$

This calculation can be further iterated upon by taking into consideration the amount of computation time left for the task to finish in relation to its currently active deadline. This is done by adding the remaining time to the current time. The new calculation (Equation 5), which will be called Estimated Response Time (ERT) priority function, then becomes:

$$u_j = 100\frac{t-s_j+rt_j}{dl_j} = 100\frac{t+dl_j-d_j+c_j-a_j}{dl_j} \tag{5}$$

Another term that raises the priority of tasks that are close to completing can be added. This term is called Nearness to Completion Time and (NTCT) and is the ratio of the period dl_j of the task over the remaining time rt_j. Thus, the priority function of ERT+NTCT is (Equation 6):

$$u_j = 100\frac{t-s_j+rt_j}{dl_j} + \frac{dl_j}{rt_j} \tag{6.a}$$

$$u_j = 100\frac{t+dl_j-d_j+c_j-a_j}{dl_j} + \frac{dl_j}{c_j+a_j} \tag{6.b}$$

In order to reshape the priority function, it is possible to use another function such as the exponential function. Considering the previously presented Elapsed Time function now as G(j), the Exponential priority function (Equation 7) is:

$$G(j) = \frac{t+dl_j-d_j}{dl_j} \tag{7.a}$$

$$u_j = \begin{cases} 100^{G(j)}, d_j - t > c_j - a_j \\ M \qquad\quad , d_j - t \leq c_j - a_j \end{cases} \tag{7.b}$$

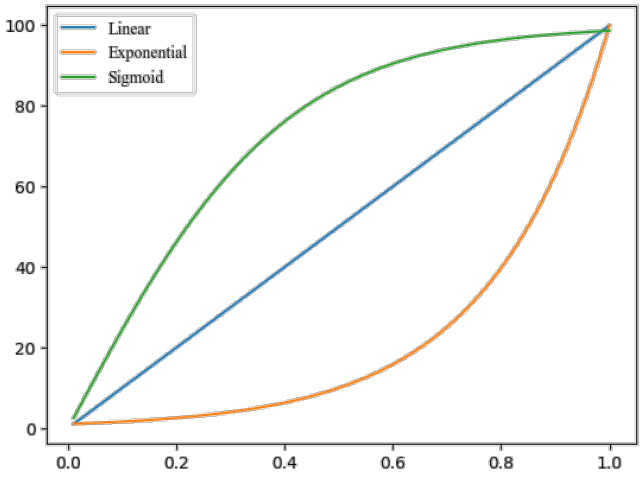
Following this logic, another way to shape the priority function is with a sigmoidal function (Equation 8):

$$G(j) = \frac{t+dl_j-d_j}{dl_j} \tag{8.a}$$

$$u_j = \begin{cases} 100\left(\frac{2}{1+e^{-5\cdot G(j)}} - 1\right), d_j - t > c_j - a_j \\ M \qquad\qquad\qquad\quad , d_j - t \leq c_j - a_j \end{cases} \tag{8.b}$$

Figure 11 shows the Linear Elapsed Time, Exponential Elapsed Time and Sigmoidal Elapsed Time functions shapes in perspective to each other.

Figure 11 - Normalized priority functions.



Source: Author.

The goal of the scheduling algorithm is to find the optimal amount of power that the satellites tasks should consume in for the solar panels to operate as close as possible to their maximum power point. When not in an eclipse situation, this optimal satellite consumption (W) is calculated using the solar panel power P_{SP} with the maximum power point tracking algorithm presented in Section 2.1.1.1, and when it is, W is maintained at a fixed value $W_{eclipse}$ to prevent the MPPT algorithm from moving erratically. Once W is calculated, the algorithm needs to find which subset of tasks best fit this power consumption goal without exceeding it. One solution to this problem is formulated as a 0-1 knapsack problem, mentioned in Section 2.3, which is (Equations 9):

$$\max \sum_{j=1}^{n} u_j x_j \quad (9.a)$$

$$s.t. \ \sum_{j=1}^{n} r_j x_j \leq W \quad (9.b)$$

$$x_j \in \{0, 1\} \quad (9.c)$$

The solution to each iteration of the simulation will dictate which tasks will stay active or not throughout its duration. The dynamic programming algorithm solution to this problem, which both time and space complexity[2] is $O(n \cdot W)$, is as follows (Algorithm 2):

[2] In computer science, big O notation is used to describe the limiting behavior of a function when the argument tends towards a particular value or infinity. It is useful to classify algorithms according to how their run time or space requirements grow as the input size grows (CORMEN, LEISERSON, *et al.*, 2009).

Algorithm 2: Dynamic programming solution

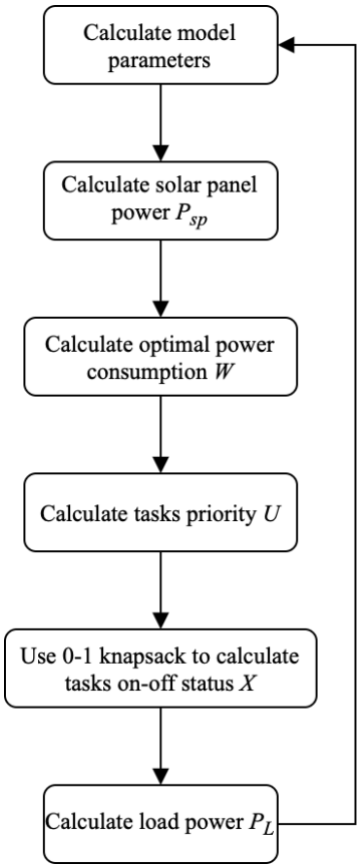
```
1  if W > ∑_{j=1}^{n} r_j then
2  |  W = ∑_{j=1}^{n} r_j
3
4  for j = 0 ... n do
5  |  for i = 0 ... W do
6  |  |  m[j, i] = 0
7
8  for j = 0 ... n do
9  |  for i = 0 ... W do
10 |  |  if j = 0 or i = 0 then
11 |  |  |  m[j, i] = 0
12 |  |  else if i ≥ r_j then
13 |  |  |  m[j, i] = max(u_j + m[j−1, i−r_j], m[j−1, i])
14 |  |  else
15 |  |  |  m[j, i] = m[j−1, i]
16
17 best = m[n, W]
18 k = W
19
20 for j = n ... 1 do
21 |  if m[j−1, k] ≠ best then
22 |  |  x_j = 1
23 |  |  k = k−r_j
24 |  |  best = m[j, k]
```

With the knowledge of which tasks remain active, the load power drawn by the tasks is calculated with (Equation 10):

$$P_L(t) = \sum_{j=1}^{n} r_j x_j \tag{10}$$

This value P_L is then fed into the model to calculate its parameters, directly affecting the solar panels operating point and consequently its generated power. The three steps of the control strategy are: calculate the optimal power consumption W from P_{SP}, then calculate tasks priority U, and using the 0-1 knapsack algorithm to calculate tasks on-off status X. These steps are represented in a diagram in Figure 12.

Figure 12 - On-board scheduling algorithm control strategy.

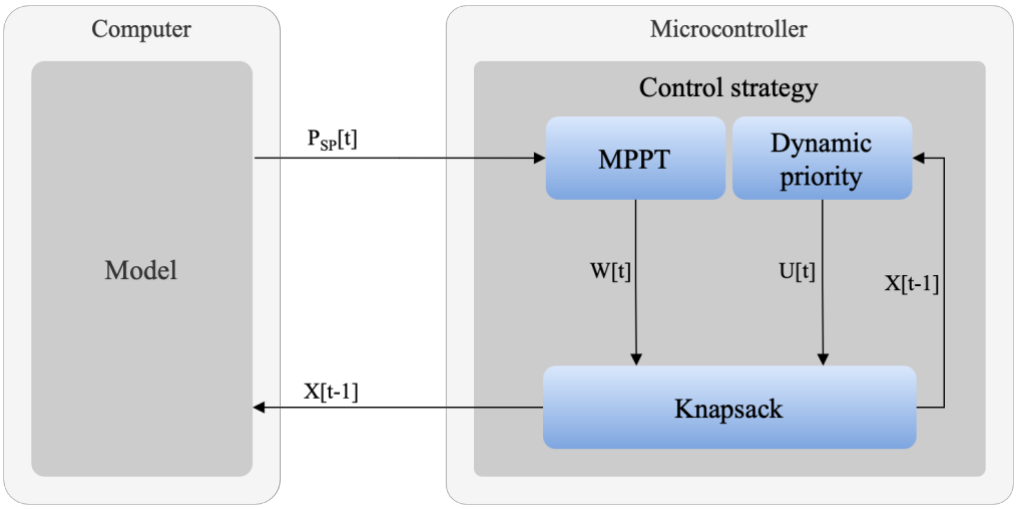


Source: Adapted from VEGA MARTINEZ, FILHO, *et al.*, 2021.

3.2 HARDWARE-IN-THE-LOOP

Trying and implementing the control strategy presented in the last section in a microcontroller is a great way to further assess the feasibility of its real time capabilities in a much closer to the real scenario situation. In this settings, model and system variables are calculated and fed to the microcontroller on either digital or analog format and are used by the latter to generate the necessary control data. The computer is responsible to calculate the dynamics of the model, which are sent to the microcontroller using serial communication, and the final output of the control strategy, i.e., the tasks active status x_j is sent back to the computer, forming a closed loop (Figure 13).

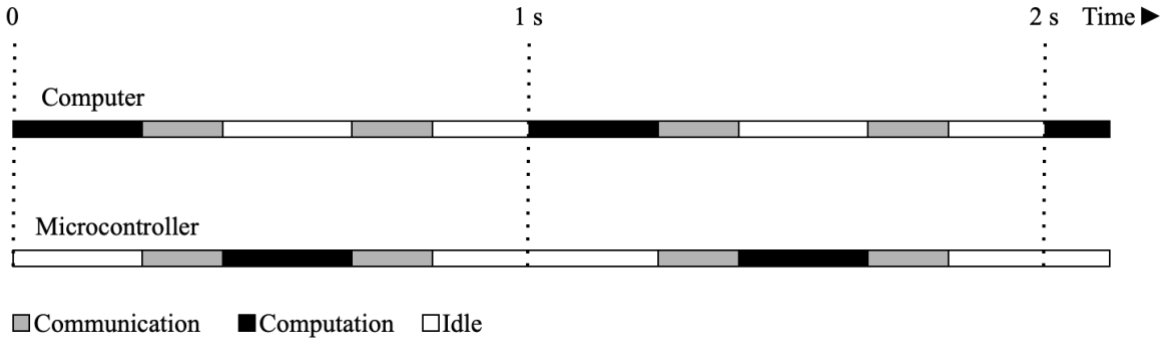
Figure 13 - Closed loop system.



Source: Author.

There are four main stages that occur in a predetermined sequence: the model parameters are calculated by the computer, the solar panel power is sent to the microcontroller, which then chooses which tasks will stay active, and send this information back to the computer. In order to satisfy the real time requirements of the problem, all communications and calculations must happen within time step frame of the simulation (which is of 1 second for all the simulations performed in this work). Figure 14 shows the timing diagram for the system.

Figure 14 - HIL timing diagram.



Source: Author.

4 EXPERIMENTATION AND RESULTS

The integrated thermal-electrical framework was implemented using MATLAB, and the configuration used in this work is based on the 1U CubeSat FloripaSat-I, which has a service module with three boards: the On-Board Data Handler (OBDH), the Electrical Power System (EPS), and the Telemetry, Tracking and Command (TTC), along with two payloads (MARCELINO, VEGA-MARTINEZ, *et al.*, 2020), and launched in December 2019. Its orbit is nearly circular and Sun-synchronous (SSO). In the model it is represented as perfectly circular and with an inclination of 90º, as represented in Board 2.

Board 2 - Orbit parameters.

Altitude	620 km
Type	SSO
Inclination	90º
Period	96 minutes

Source: Author.

The results are divided in two parts, the first one consists entirely of computer simulations, and the second one in hardware-in-the-loop simulations. In the first part the integrated framework was experimented upon with the goal of improving the control strategy used in the work of Vega Martínez (2022). The main points of improvement being focused on were the number of missed deadlines, energy harvesting capabilities, and battery temperature operating conditions, and will be discussed in Section 4.1. In the second part, the control strategy was implemented on the target language (C language) on an embedded environment with three main goals, to assert the correctness of implementation, verify that it satisfies the real time requirements of the system, and finally, to conduct a performance analysis of the execution times with different cases. This second part will be further explained and discussed in Section 4.2.

4.1 COMPUTER SIMULATIONS

The different priority policies introduced in Section 3.1 were used in the aforementioned FloripaSat-I conditions in order to evaluate the impact on the number of missed deadlines. The total number of time steps was 70024, and each step represents 1 second, totaling 12 orbits. This information is displayed in Board 3.

Board 3 - Simulation parameters.

Number of steps	70024
Step duration	1 [s]
Number of orbits	12

Source: Author.

Each priority policy was treated as a different scenario s, with a total of $S = 6$ scenarios, as per Table 2.

Table 2 - List of scenarios.

Scenario (s)	Priority policy
1	ET Linear + M (eq. 4)
2	ET Exponential + M (eq. 7)
3	ET Sigmoid + M (eq. 8)
4	ET Linear (eq. 3)
5	ERT (eq. 5)
6	ERT + NTCT (eq. 6)

Source: Author.

After running the simulation for the different priority policies, the results were as following in Table 3.

Table 3 - Deadline misses with different priority policies.

Priority policy	Percentage of missed deadlines [%]
ET Linear + M (eq. 4)	0.23
ET Exponential + M (eq. 7)	0.22
ET Sigmoid + M (eq. 8)	0.17
ET Linear (eq. 3)	0.23
ERT (eq. 5)	0.20
ERT + NTCT (eq. 6)	0.28

Source: Author.

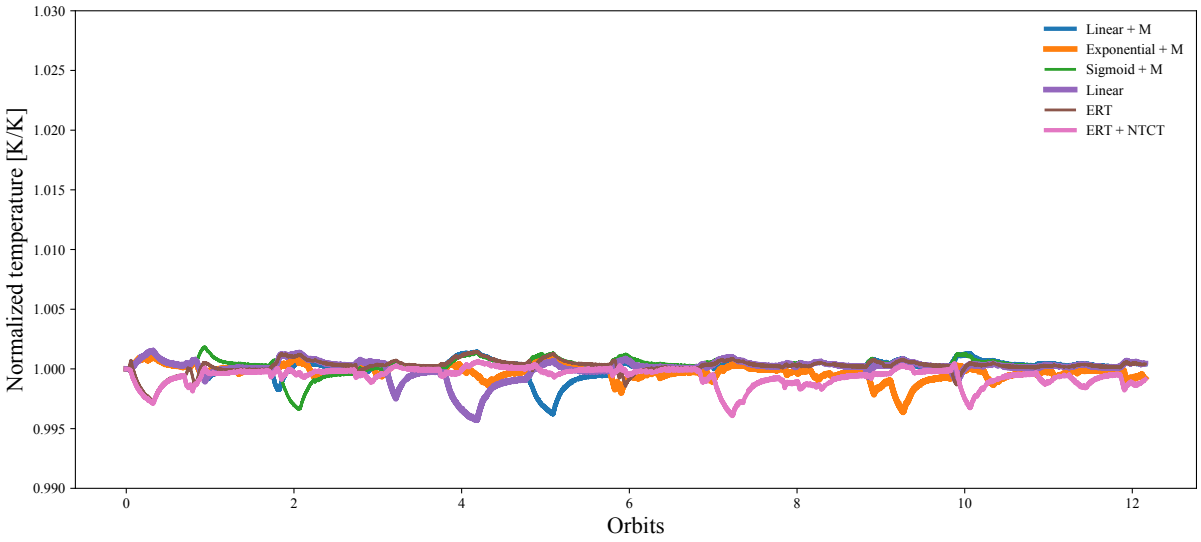
From the results, it can be observed that the number of missed deadlines was reduced when compared to the original control strategy presented by Vega Martínez (2022). The policies with linear shape (Linear + M, Linear, and ERT) had similar results, with the ERT performing better among those. The addition of the Nearness To Completion Time (NTCT) to the ERT policy worsened the result, while the Sigmoid + M presented the best result with the least amount of deadline misses.

Regarding the operating temperature of the battery, normalizing the values of each scenario s as per (Equation 11):

$$\widetilde{\text{temperature}}_s(t) = \frac{\text{temperature}_s(t)}{\left(\frac{\sum_{S=1}^{S} \text{temperature}_S(t)}{S}\right)}, \ \forall s \in S \tag{11}$$

It is possible to see that all simulations performed extremely similar, despite the difference in priority policies, as shown in Figure 15.

Figure 15 – Normalized battery operating temperature.



Source: Author.

As for the MPPT performance, by calculating the Euclidian distance between the desired and actual power generated from the solar panels throughout the simulation of each scenario, it was possible to observe that the overall performance was similar, and the difference in priority policies did not influence in a significant way the individual results. In general, for two points p and q given by Cartesian coordinates in n-dimensional Euclidian spaces, the distance is as shown below in Equation (12). When evaluating two time series, which is this case, the number of dimensions is represented by the number of time points.

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} \qquad (12)$$

Another way to evaluate the scenarios is by using Pearson's correlation coefficient, which showed similar results, i.e., that the performance did not vary significantly. Pearson's correlation coefficient value, often represented by r, although less adequate for time series evaluation, represents the strength in relation between two linear variables X and Y, both of size n, and is calculated as the covariance of X and Y divided by the square root of their multiplied variances (Equation 13). The r value can go from -1 (when there is a strong negative relation) to 0 (when there is no relation) to 1 (when there is a strong positive relation). The calculated correlation coefficient values for each scenario are displayed in Table 4.

$$r = \frac{cov(X,Y)}{\sqrt{var(X) \cdot var(Y)}} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \cdot \sum_{i=1}^{n}(y_i - \bar{y})^2}} \qquad (13)$$

Table 4 - Ideal and actual solar panels power comparison.

Priority policy	Mean ideal power [W]	Mean actual generated power [W]	Euclidean distance	Correlation coefficient value
ET Linear + M (eq. 4)	2.0069	1.7906	83.74	0.8833
ET Exponential + M (eq. 7)	2.0069	1.8855	54.71	0.9156
ET Sigmoid + M (eq. 8)	2.0070	1.7736	86.73	0.9017
ET Linear (eq. 3)	2.0069	1.8989	45.67	0.9490
ERT (eq. 5)	2.0070	1.8374	70.05	0.8942
ERT + NTCT (eq. 6)	2.0068	1.8044	86.71	0.8279

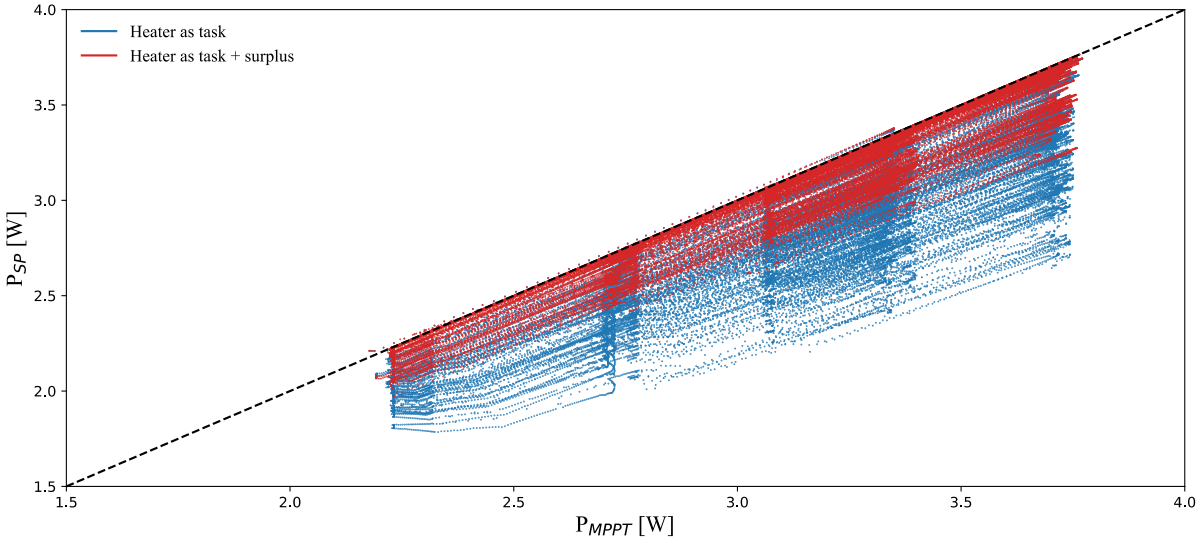
Source: Author.

4.1.1 Changes to control strategy

To improve the operating temperature of the battery and the amount of harvested energy the control strategy was changed by delivering the remainder of the MPPT algorithm output (i.e., the difference between the desired power W and the load power P_L) to the heater after allocating the tasks after each iteration.

The mean ideal power for the solar panels stayed at 2.0109 W compared to 2.0069 W, and the mean actual generated power increased to 1.9579 W from 1.7906 W. The Euclidian distance using this modified control strategy was $d = 27.64$ compared to $d = 83.74$, and the correlation coefficient was $r = 0.9713$ compared to $r = 0.8833$. Figure 16 was obtained by plotting the ideal (Pmppt) against the actual (PspTotal) generated power throughout the 12 orbits. It is possible to observe the improvement in correlation, as the points that represent the stay closer to the identity line, indicating that the MPPT algorithm performed better.

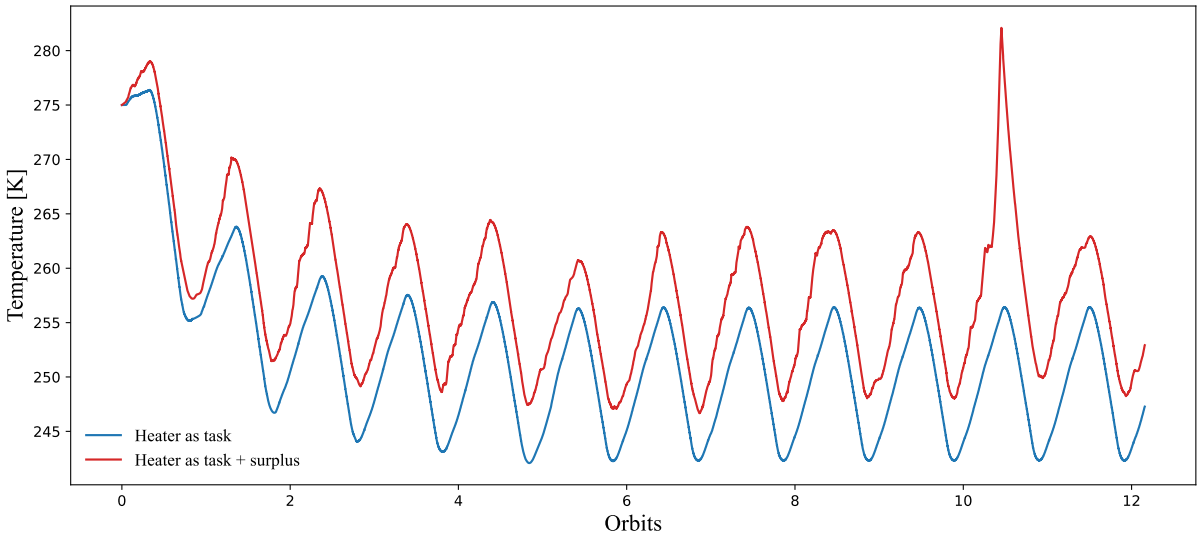
Figure 16 - Ideal (Pmppt) against actual (PspTotal) generated power from solar panels with different control strategies and Linear + M priority policy.



Source: Author.

As for the battery operating temperature, the minimum temperature reached during the simulation increased from 242.0 K to 246.6 K, with the average temperature increasing by 6.3 K, from 251.5 K to 257.8 K (Figure 17). This increase in temperature brings it closer to desirable range, discussed in Section 2.1.2.

Figure 17 - Battery temperature during 12 orbits with different control strategies and Linear + M priority policy.



Source: Author.

4.1.2 Random cases

Ten random tasks sets were generated (Table 7) using the original FloripaSat-I tasks set to further test the proposed control strategies. This was done calculating the mean and

standard deviation values of the original tasks set characteristics, while maintaining the task that represents the heater out of this calculation, since it was left unchanged across the randomly generated tasks. Calculated values from original tasks set are presented in Table 5.

Table 5 - Mean value and standard deviation of FloripaSat-I tasks set.

Characteristic	**Value**	**Standard deviation**
Mean computing time [s]	8.00	21.53
Mean period [s]	639.11	1474.97
Total power consumption [mW]	7924.00	511.76

Source: Author.

Because the standard deviation is greater than the mean value for all three characteristics, it was possible for negative values to appear, so all negative values that appeared during the process of generating the random tasks sets were discarded. To ensure the randomly generated tasks sets are still a good fit for the framework when it is configured with the FloripaSat-I dimensions, the mean energy expenditure for all random sets combined was kept sufficiently close to that of the FloripaSat-I original task set (Table 6). The mean energy expenditure is calculated using the following equation (Equation 14) where n is the total number of tasks, r_j, c_j and dl_j are respectively the power consumption, computing time, and period of task j.

$$\text{mean energy expenditure} = \sum_{j=1}^{n} r_j \cdot \frac{c_j}{dl_j} \tag{14}$$

Table 6 - Random tasks sets characteristics.

Tasks set name	Mean computing time [s]	Mean period [s]	Total power consumption [mW]
Random set 1	16.16±6.51	1021.00±938.31	6821.00
Random set 2	24.16±16.58	1077.83±985.49	6874.00
Random set 3	39.16±15.86	1104.83±1035.92	7905.00
Random set 4	27.50±25.27	911.33±581.74	9357.00
Random set 5	18.83±13.72	1174.16±1028.34	7644.00
Random set 6	24.83±16.65	999.50±1010.45	6150.00
Random set 7	21.66±10.32	705.16±354.27	7991.00
Random set 8	30.66±10.32	2079.33±1382.57	7473.00
Random set 9	23.66±9.32	2072.66±1899.20	7425.00
Random set 10	34.83±16.44	1739.16±1388.16	7347.00

Source: Author.

Table 7 - Mean energy expenditure of tasks sets.

Tasks set name	**Mean energy expenditure [mW]**
Random set 1	599.03
Random set 2	319.51
Random set 3	639.31
Random set 4	416.35
Random set 5	691.17
Random set 6	549.36
Random set 7	402.59
Random set 8	362.96
Random set 9	282.31
Random set 10	453.87
Random sets mean	**471.65**
FloripaSat-I	462.54

Source: Author.

The random cases were simulated using the same priority policies used in the real case to verify if the improvement in quality-of-service (i.e., reduced deadline misses) held similar results regarding which policies performed better. The results, shown in Table 8, differed for each random set. This indicates that the best performing priority policy may vary for different missions based on its tasks characteristics. For this reason, the framework can be an important tool for testing control strategies prior to its on-board and online implementation and use.

Table 8 - Percentage of missed deadlines for random cases using different priority policies, with best performing policies for each case marked in green.

Tasks set name	Linear + M	Exponential + M	Sigmoid + M	Linear	ERT	ERT + NTCT
Random set 1	36.59%	37.18%	37.28%	38.31%	**36.36%**	37.41%
Random set 2	27.76%	27.76%	28.87%	**26.69%**	28.73%	30.97%
Random set 3	35.52%	37.19%	33.85%	36.38%	**32.03%**	36.02%
Random set 4	32.00%	27.99%	29.85%	34.81%	27.44%	**25.23%**
Random set 5	38.72%	40.75%	41.33%	40.02%	41.02%	**37.97%**
Random set 6	31.79%	**30.25%**	31.19%	31.23%	33.57%	31.48%
Random set 7	25.90%	23.79%	23.55%	25.22%	**23.48%**	24.82%
Random set 8	25.02%	23.58%	23.14%	25.02%	22.50%	**20.45%**
Random set 9	27.61%	28.12%	**27.19%**	27.61%	28.52%	27.75%
Random set 10	30.38%	30.53%	30.20%	**28.82%**	30.53%	29.77%

Source: Author.

As for the battery temperature and correlation between ideal and actual generated power from solar, it is possible to see that the normalized temperature values (Table 9) as well as mean ideal power (Table 10), mean actual generated power (Table 11), Euclidian distance values (Table 12) and correlation values (Table 13) did not vary in a significant way for any of the randomly generated cases, while the plots for normalized temperature values kept a very similar shape as Figure 15, maintaining consistency with the results obtained in with the real case using the FloripaSat-I tasks.

Table 9 - Normalized mean battery temperature.

Tasks set name	Linear + M	Exponential + M	Sigmoid + M	Linear	ERT	ERT + NTCT
Random set 1	1.000099	0.999989	1.000050	0.999866	1.000107	0.999886
Random set 2	1.000155	1.000031	1.000010	1.000178	1.000004	0.999619
Random set 3	0.999960	0.997829	1.000130	0.999943	1.000267	0.999915
Random set 4	0.999846	1.000167	0.999970	0.999524	1.000230	1.000261
Random set 5	1.000261	0.999904	0.999904	1.000007	0.999946	1.000128
Random set 6	1.000028	1.000098	1.000050	1.000103	0.999938	0.999781
Random set 7	0.999904	1.000026	1.000218	0.999995	1.000256	0.999598
Random set 8	0.999840	0.999956	1.000017	0.999840	1.000073	1.000271
Random set 9	1.000023	0.999940	1.000079	1.000023	0.999961	0.999970
Random set 10	0.999972	0.999927	1.000043	1.000068	0.999958	1.000030

Source: Author.

Table 10 - Mean ideal power from solar panels throughout the 12 orbits.

Tasks set name	Linear + M	Exponential + M	Sigmoid + M	Linear	ERT	ERT + NTCT
Random set 1	2.0066	2.0066	2.0066	2.0065	2.0066	2.0065
Random set 2	2.0066	2.0066	2.0066	2.0066	2.0066	2.0065
Random set 3	2.0066	2.0065	2.0066	2.0066	2.0066	2.0066
Random set 4	2.0066	2.0066	2.0066	2.0065	2.0066	2.0066
Random set 5	2.0066	2.0065	2.0066	2.0066	2.0066	2.0066
Random set 6	2.0066	2.0066	2.0066	2.0066	2.0066	2.0066
Random set 7	2.0066	2.0066	2.0066	2.0066	2.0066	2.0065
Random set 8	2.0066	2.0066	2.0066	2.0066	2.0066	2.0067
Random set 9	2.0066	2.0066	2.0066	2.0066	2.0066	2.0066
Random set 10	2.0066	2.0066	2.0066	2.0066	2.0066	2.0066

Source: Author.

Table 11 - Mean actual generated power from solar panels throughout the 12 orbits.

Tasks set name	Linear + M	Exponential + M	Sigmoid + M	Linear	ERT	ERT + NTCT
Random set 1	1.9761	1.9857	1.9455	1.9456	1.9395	1.9830
Random set 2	1.9663	1.9616	1.9492	1.9679	1.9610	1.9450
Random set 3	1.9337	1.9450	1.9645	1.9459	1.9567	1.9385
Random set 4	1.9279	1.9804	1.9356	1.9022	1.9801	1.9569
Random set 5	1.9572	1.9126	1.9641	1.9309	1.9521	1.9451
Random set 6	1.9846	1.9840	1.9642	1.9681	1.9413	1.9660
Random set 7	1.9736	1.9880	1.9841	1.9733	1.9894	1.9803
Random set 8	1.9628	1.9856	1.9630	1.9628	1.9800	1.9915
Random set 9	1.9704	1.9203	1.9567	1.9704	1.9842	1.9163
Random set 10	1.9585	1.9643	1.9775	1.9537	1.9670	1.9809

Source: Author.

Table 12 - Euclidian distance between ideal and actual generated power from solar panels throughout the 12 orbits.

Tasks set name	Linear + M	Exponential + M	Sigmoid + M	Linear	ERT	ERT + NTCT
Random set 1	15.96	11.40	29.85	30.91	32.18	12.18
Random set 2	20.78	21.59	29.46	23.24	22.54	29.40
Random set 3	34.78	31.16	22.82	31.84	27.92	34.93
Random set 4	40.89	15.96	39.75	48.46	16.63	32.10
Random set 5	26.14	39.81	23.19	35.73	27.27	30.15
Random set 6	11.12	11.16	21.80	18.64	29.64	19.60
Random set 7	17.87	9.97	12.57	18.09	9.65	14.24
Random set 8	26.00	12.25	25.87	26.00	15.06	9.39
Random set 9	18.70	37.94	23.38	18.70	11.92	42.58
Random set 10	26.17	28.44	25.34	30.76	20.63	13.50

Source: Author.

Table 13 – Pearson's correlation coefficient between ideal and actual generated power from solar panels throughout the 12 orbits.

Tasks set name	Linear + M	Exponential + M	Sigmoid + M	Linear	ERT	ERT + NTCT
Random set 1	0.99018	0.99475	0.96892	0.96499	0.96507	0.99456
Random set 2	0.98366	0.98427	0.96758	0.97595	0.98195	0.97102
Random set 3	0.95998	0.96489	0.97910	0.96129	0.96774	0.95520
Random set 4	0.93754	0.98845	0.93610	0.92733	0.98716	0.95242
Random set 5	0.97345	0.96075	0.97817	0.95860	0.97336	0.96857
Random set 6	0.99550	0.99561	0.98206	0.98812	0.97347	0.98689
Random set 7	0.98710	0.99608	0.99342	0.98673	0.99613	0.99146
Random set 8	0.97046	0.99344	0.97072	0.97046	0.99040	0.99597
Random set 9	0.98675	0.95995	0.98237	0.98675	0.99441	0.94186
Random set 10	0.97235	0.96754	0.99083	0.95968	0.98358	0.99295

Source: Author.

4.1.2.1 Changes to control strategy

The same changes to control strategy presented in Section 4.1.1 were implemented for the random cases simulation. These changes resulted in same benefits seen in the real case simulations, with minimum temperatures reached increasing throughout all cases, along with significant increases in mean temperature (Table 14).

Table 14 - Minimum reached temperature and temperature difference of randomly generated cases simulation.

Tasks set name	Minimum temperature (heater as task)	Minimum temperature (heater as task + excedent)	Difference in mean temperature
Random set 1	241.3 K	250.9 K	13.2 K
Random set 2	241.5 K	255.3 K	16.0 K
Random set 3	241.3 K	252.1 K	13.7 K
Random set 4	241.5 K	252.8 K	12.5 K
Random set 5	241.5 K	256.5 K	16.5 K
Random set 6	241.5 K	254.4 K	17.6 K
Random set 7	241.4 K	250.9 K	12.3 K
Random set 8	241.4 K	250.8 K	13.5 K
Random set 9	241.5 K	246.5 K	9.0 K
Random set 10	241.4 K	252.5 K	14.3 K

Source: Author.

As for the Euclidian distance and correlation coefficient, the improvement was present in almost all randomly generated cases, and on the ones that was not, the values were very close. This lack of improvement could not be traced back to a single characteristic, nor the mean energy expenditure of the random cases, therefore, a relation between them could not be established.

Table 15 - Mean ideal and actual generated power of randomly generated cases simulation

Tasks set name	Mean ideal power (heater as task)	Mean actual generated power (heater as task)	Mean ideal power (heater as task + excedent)	Mean actual generated power (heater as task + excedent)
Random set 1	2.0066	1.9761	2.0141	1.9568
Random set 2	2.0066	1.9663	2.0155	1.9819
Random set 3	2.0066	1.9337	2.0144	1.9715
Random set 4	2.0066	1.9279	2.0139	1.9662
Random set 5	2.0066	1.9572	2.0161	1.9810
Random set 6	2.0066	1.9846	2.0161	1.9633
Random set 7	2.0066	1.9736	2.0127	1.9742
Random set 8	2.0066	1.9628	2.0141	1.9670
Random set 9	2.0066	1.9704	2.0118	1.9445
Random set 10	2.0066	1.9585	2.0145	1.9671

Source: Author.

Table 16 – Euclidian distance and correlation coefficient between ideal and actual harvested energy of randomly generated cases simulation.

Tasks set name	Euclidian distance (heater as task)	Euclidian distance (heater as task + excedent)	Correlation coefficient (heater as task)	Correlation coefficient (heater as task + excedent)
Random set 1	15.96	24.03	0.99018	0.98617
Random set 2	20.78	16.68	0.98366	0.99010
Random set 3	34.78	20.03	0.95998	0.98719
Random set 4	40.89	20.77	0.93754	0.98838
Random set 5	26.14	16.84	0.97345	0.99048
Random set 6	11.12	23.64	0.99550	0.98374
Random set 7	17.87	18.71	0.98710	0.98853
Random set 8	26.00	21.02	0.97046	0.98738
Random set 9	18.70	28.13	0.98675	0.98109
Random set 10	26.17	21.91	0.97235	0.98494

Source: Author.

These experimentations using the computer simulation environment lead to the conclusion that although the presented control strategies lead to good results in the FloripaSat-I real task set, the results from the randomly generated cases showed that the outcome can differ for different tasks sets with different characteristics whilst using the same control strategy, and

missions can benefit from having a framework in which different strategies can be tested prior to its implementation in the embedded final environment.

4.2 HARDWARE-IN-THE-LOOP SIMULATIONS

The second part of the experiments were conducted using the hardware-in-the-loop setup described in Section 3.2. The goal was to assert whether the algorithm could succeed in a real time environment and evaluate its performance while doing so.

The microcontroller used was the TMS320F28379D from Texas Instruments, which is based on a 200 MHz dual 32-bit C28x CPUs. The development board used in which the microcontroller is present was the LAUNCHXL-F28379D which has 1 MB Flash memory. The code was written in C language using TI's "Code Composer Studio" integrated development environment (CCS IDE), which comes with tools for configuring, compiling and debugging for TI's microcontrollers.

Board 4 - Hardware-in-the-loop development information.

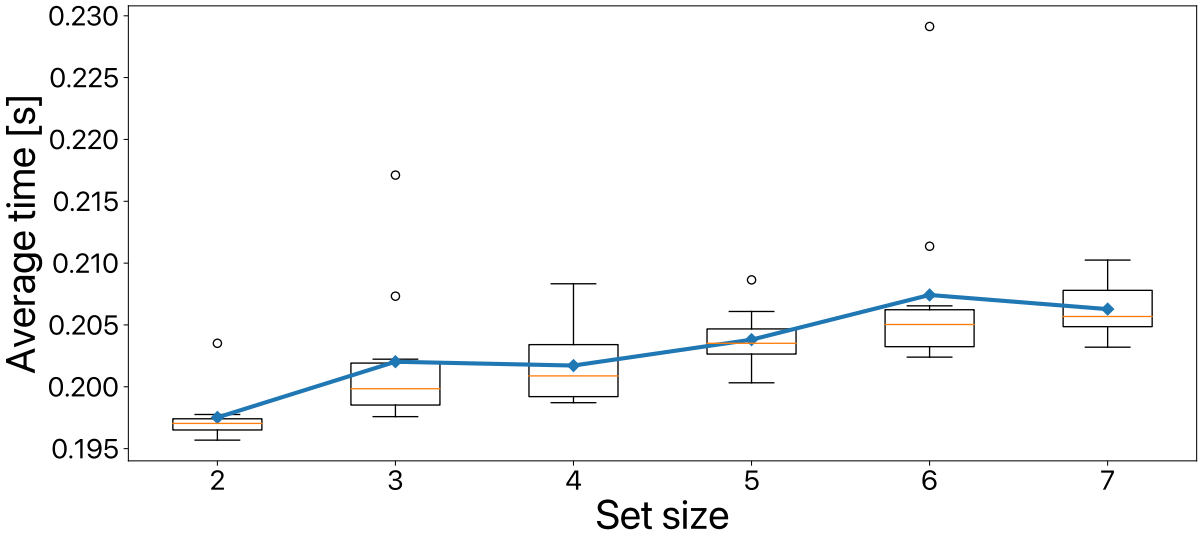
Microcontroller	TMS320F28379D
Manufacturer	Texas Instruments
Environment	Code Composer Studio
Language	C

Source: Author.

The computer and the MCU both have a Universal Synchronous Asynchronous Receiver Transmitter (USART) and use the standard RS232 protocol to send and receive data. The USART was configured with the highest possible transmission rate that did not cause any errors during transmissions, which was 9600 bits per second. Each byte is accompanied by a start bit and a stop bit, and no parity bit, leading to approximately 1.04 milliseconds to transmit each byte if sent continuously, but in the real system more time may be spent because of numerous factors, such as the host computer operating system and drift in the amount of time to send the bytes (FINNSET, K. RAO and ANTONSEN, 2006). The total amount of bytes necessary to perform all the communication for a full iteration of the simulation is equal to the number of tasks plus one, so for seven tasks, communication amounts for roughly 8.33 milliseconds.

The integrated model was kept on the computer and the control strategy was rewritten in C for the micro-controller. After running the experiment with the hardware-in-the-loop setup, the results were compared with the simulation and the correctness of the algorithm implemented in the target language was verified. A performance analysis was conducted by varying the number of tasks in different runs of the setup and measuring the time it took to complete a full loop iteration of the system for multiple time steps. The results are shown in Figure 18, which shows the average time that the microcontroller takes to receive, compute, and answer the computer, and from them it is possible to conclude that the implementation could satisfy the real time requirements of the system.

Figure 18 - Performance analysis of the hardware-in-the-loop simulation with different set sizes.



Source: Author.

From Figure 18 it is possible to observe a linear impact of the number of tasks on the computation time taken by the control strategy running on the micro controller. This time always stayed below 250 milliseconds, with the average time for all cases being 203 milliseconds, which means it took less than a quarter of the model time step.

CONCLUSION

Electrical power system for nanosatellites have been extensively studied by the Space Technology Research Laboratory at the Federal University of Santa Catarina, with the FloripaSat-I being the object of many studies developed there. Among the results of these research are the conclusion that the directly coupled architecture outweighs the benefits of a discrete MPPT integrated circuit and the proposal of an integrated thermal-electrical simulation framework using this architecture. Correctly managing energy and battery temperature is crucial to any satellite mission success, since the environment poses extreme temperature conditions which hinder the performance of the battery if not dealt with. The load significantly influences the operating point of the solar panels in the directly coupled architecture and is determined by which tasks are currently running. The purpose of this work was to develop a hardware-in-the-loop (HIL) framework for testing on-board task scheduling algorithms in nanosatellites. The main objective was to validate a priority-based, energy-drive scheduling algorithm introduced by Vega Martinez (2022).

Using the knapsack-based scheduling algorithm, this work evaluated different scenarios: one with the real task set of the FloripaSat-I mission, and others with randomly generated sets using the mean and standard deviation calculated from the real task set. Different priority functions and two heater utilization strategies were utilized. For the real case scenario, the results showed that by shaping the priority calculation using a sigmoid function reduced the number of missed deadlines by 26% from the base calculation (ET+M). On the randomly generated cases showed that the best priority function cannot be defined universally, for it depends on different properties of the tasks, and it was not possible to define relationships with simple and objective statistical parameters. As for the heater allocation strategies, on the real case scenario, the heater plus excedent strategy improved the amount of energy harvested from solar panels, as shown by the reduction of the Euclidian distance (from $d = 84.74$ to $d = 27.64$). This improvement can also be observed on most of the randomly generated cases, with the ones that did not improve showing very similar values (of Euclidian distance and correlation coefficient) with both strategies. The mean and minimum temperature also increased in all cases, with a 6.3 K increase in mean temperature on the real case and an average increase of 13.86 K throughout the randomly generated cases.

The integrated simulation model framework was then expanded into a hardware-in-the-loop setup, which kept the model on the computer and moved the control strategy to a microcontroller that closely represents the final embedded environment in which the algorithm will perform, which then was used to implement the scheduling algorithm in the target language. A computation cycle of 1 second was used, and the correctness and real time capabilities of the algorithm was validated, and the control strategy calculations were performed by the microcontroller in under a quarter of a second.

In this context, this work presents an on-board embedded algorithm together with a hardware-in-the-loop framework which enables previous evaluation of different priority functions when aiming to maintain quality of service whilst maximizing energy harvesting potential, depending on the mission tasks characteristics, and helps in the verification process of nanosatellites.

One way in which future works could improve the hardware-in-the-loop simulations would be to implement the algorithm together with final implementation of a mission, to verify if the algorithm can still suffice its real time constraints when sharing CPU time with other actions that shall be performed by the rest of the software on the MCU.

5 REFERENCES

AUNG, H. et al. Battery Management System With State-of-Charge and Opportunistic State-of-Health for a Miniaturized Satellite. **IEEE Transactions on Aerospace and Electronic Systems**, v. 56, p. 2978-2989, 2020.

BONNICI, M. et al. Analytical and numerical models for thermal related design of a new pico-satellite. **Applied Thermal Engineering**, v. 159, p. 113908, 2019. ISSN 1359-4311.

BOUWMEESTER, J.; GUO, J. Survey of worldwide pico- and nanosatellite missions, distributions and subsystem technology. **Acta Astronautica**, v. 67, p. 854-862, 2010. ISSN 0094-5765.

CAL POLY SLO; JOHNSTONE, A. **CubeSat Design Specification Rev. 14**, 2020.

CORMEN, T. et al. **Introduction to Algorithms**. [S.l.]: MIT Press, 2009.

CORPINO, S. et al. Thermal design and analysis of a nanosatellite in low earth orbit. **Acta Astronautica**, v. 115, p. 247-261, 2015. ISSN 0094-5765.

CORPINO, S.; STESINA, F. Verification of a CubeSat via hardware-in-the-loop simulation. **IEEE Transactions on Aerospace and Electronic Systems**, v. 50, p. 2807-2818, 2014.

DE CARUFEL, J.; MARTIN, E.; PIEDBœUF, J.-C. Control strategies for hardware-in-the-loop simulation of flexible space robots. **IEE Proceedings - Control Theory and Applications**, v. 147, n. 6, p. 569-579(10), November 2000. ISSN 1350-2379.

DOBIÁš, P.; CASSEAU, E.; SINNEN, O. Fault-Tolerant Online Scheduling Algorithms for CubeSats. **In Proceedings of the 11th Workshop on Parallel Programming and Run-Time Management Techniques for Many-core Architectures / 9th Workshop on Design Tools and Architectures for Multicore Embedded Computing Platforms (PARMA-DITAM'2020)**, New York, NY, USA, 2020. 1-6.

EICKHOFF, J. **Simulating Spacecraft Systems**. New York: Springer, 2009.

FINNSET, R.; K. RAO, S.; ANTONSEN, J. Real time hardware-in-loop simulation of ESMO satellite attitude control system. **Modeling, Identification and Control**, v. 27, p. 125–140, 2006.

FRÖHLICH, A. A.; BEZERRA, E. A.; SLONGO, L. K. Experimental analysis of solar energy harvesting circuits efficiency for low power applications. **Computers & Electrical Engineering**, v. 45, p. 143-154, 2015. ISSN 0045-7906.

FRÉVILLE, A.; HANAFI, S. The Multidimensional 0-1 Knapsack Problem—Bounds and Computational Aspects. **Annals of Operations Research**, October 2005. 195-227.

GALATIS, G.; GUO, J.; BUURSINK, J. Development of a solar array drive mechanism for micro-satellite platforms. **Acta Astronautica**, v. 139, p. 407-418, 2017. ISSN 0094-5765.

GANS, N. R. et al. A hardware in the loop simulation platform for vision-based control of unmanned air vehicles. **Mechatronics**, v. 19, p. 1043-1056, 2009. ISSN 0957-4158. Special Issue on Hardware-in-the-loop simulation.

KESSLER SLONGO, L. et al. Nanosatellite electrical power system architectures: Models, simulations, and tests. **International Journal of Circuit Theory and Applications**, v. 48, p. 2153-2189, 2020.

MA, S. et al. Temperature effect and thermal impact in lithium-ion batteries: A review. **Progress in Natural Science: Materials International**, v. 28, p. 653-666, 2018. ISSN 1002-0071.

MARCELINO, G. M. et al. A Critical Embedded System Challenge: The FloripaSat-1 Mission. **IEEE Latin America Transactions**, v. 18, p. 249-256, 2020.

NASA. **State-of-the-Art:** Small Spacecraft Technology. Mountain View: NASA, 2021.

PALACÍN, M. R. Understanding ageing in Li-ion batteries: a chemical issue. **Chem. Soc. Rev.**, v. 47, n. 13, p. 4924-4933, 2018.

PANG, C. K. et al. Design and robust scheduling of nano-satellite swarm for synthetic aperture radar applications. **2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)**, 2014. 705-710.

PIEGARI, L.; RIZZO, R. Adaptive perturb and observe algorithm for photovoltaic maximum power point tracking. **IET Renewable Power Generation**, v. 4, n. 4, p. 317-328(11), July 2010. ISSN 1752-1416.

POGHOSYAN, A.; GOLKAR, A. CubeSat evolution: Analyzing CubeSat capabilities for conducting science missions. **Progress in Aerospace Sciences**, v. 88, p. 59-83, 2017. ISSN 0376-0421.

RIGO, C. A. et al. A nanosatellite task scheduling framework to improve mission value using fuzzy constraints. **Expert Systems with Applications, Volume 175**, 2021.

SLONGO, L. K. et al. Energy-driven scheduling algorithm for nanosatellite energy harvesting maximization. **Acta Astronautica**, v. 147, p. 141-151, 2018. ISSN 0094-5765.

VANDERSTER, D. C. et al. Resource allocation on computational grids using a utility model and the knapsack problem. **Future Generation Computer Systems**, v. 25, p. 35-50, 2009. ISSN ISSN: 0167-739X.

VEGA MARTINEZ, S. et al. An Integrated Thermal-Electrical Model for Simulations of Battery Behavior in CubeSats. **Applied Sciences**, v. 11, 2021. ISSN 2076-3417.

VEGA MARTÍNEZ, S. Nanosatellite task scheduling algorithm for energy harvesting and QoS enhancement tested through a self-proposed thermal-electrical model. **Thesis**, Florianópolis, 2022. 112.

W. ZHANG, A.; BEHBAHANI, A. S.; ELTAWIL, A. M. Joint frequency scheduling and power allocation for CubeSat communication. **38th International Communications Satellite Systems Conference (ICSSC 2021)**, 2021. 184-188.

WANG, J. et al. Towards dynamic real-time scheduling for multiple earth observation satellites. **Journal of Computer and System Sciences**, v. 81, p. 110-124, 2015. ISSN 0022-0000.

WU, X.; VLADIMIROVA, T. **Hardware-in-Loop Simulation of a Satellite Sensor Network for Distributed Space Applications**. 2008 NASA/ESA Conference on Adaptive Hardware and Systems. [S.l.]: [s.n.]. 2008. p. 424-431.