Firmware para Computador de Bordo de Nanossatélite no Padrão CubeSat

José I. S. Camilo * **Fernando H. Couto** * **Samaherni M. Dias** *
Kurios I. P. M. Queiroz *

* *Universidade Federal do Rio Grande do Norte, UFRN, Natal, RN, Brasil (e-mail: jose.camilo.709@ufrn.edu.br)*

Abstract: In recent years, increased investments in space exploration and the miniaturization of electronic components have significantly contributed to the creation and popularization of the CubeSat standard. As a result, many companies and universities have begun to work using this new tool. In this context, the CONASAT project (Constellation of Environmental Nanosatellites) emerges as a partnership between the Regional Center of the National Institute for Space Research (INPE/CRN) and the Federal University of Rio Grande do Norte (UFRN). The project aims to present a low-cost solution for the Brazilian Environmental Data Collection System (SBCDA) using nanosatellites (CubeSats). To ensure these nanosatellites operate satisfactorily, some hardware and software solutions are required in their onboard computers (OBC) to mitigate potential failures from the space environment. This article aims to present a onboard computing system, based on FreeRTOS, for the OBCs of the satellites in the CONASAT project.

Resumo: Nos últimos anos, o aumento dos investimentos em exploração espacial e a miniaturização de componentes eletrônicos contribuíram significativamente para a criação e popularização do padrão CubeSat. Com isso, muitas empresas e universidades começaram a realizar trabalhos utilizando essa nova ferramenta. Neste contexto, surge o projeto CONASAT (Constelação de Nanossatélites Ambientais), uma parceria entre o Centro Regional do Instituto Nacional de Pesquisas Espaciais (INPE/CRN) e a Universidade Federal do Rio Grande do Norte (UFRN). O projeto tem como objetivo apresentar uma solução de baixo custo para o Sistema Brasileiro de Coleta de Dados Ambientais (SBCDA) utilizando nanossatélites (CubeSats). Para que esses nanossatélites operem de maneira satisfatória, são necessárias algumas soluções em *hardware* e *software* em seus computadores de bordo (OBC) contra possíveis falhas provenientes do ambiente espacial. Este artigo se objetiva em apresentar o *firmware* do sistema de computação de bordo, baseado em FreeRTOS, para os OBCs dos satélites do projeto CONASAT.

Keywords: CubeSats; OBC; Firmware; CONASAT; nanosatellites.
Palavras-chaves: CubeSats; OBC; Firmware; CONASAT; nanossatélites.

1. INTRODUÇÃO

Diversos são os desafios quando se trata de computação embarcada para exploração espacial, o principal deles está relacionado aos efeitos da radiação ionizante, presente em maiores quantidades nas órbitas espaciais, nos componentes de eletrônica digital embarcados (Mota, 2022). Equipamentos espaciais requerem componentes eletrônicos muito robustos (Souza et al., 2022) e subsistemas redundantes para garantir a confiabilidade do satélite, o que, por sua vez, aumentam tanto o tamanho quanto o custo do projeto (Botma e Steyn, 2011).

De acordo com Botma e Steyn (2011),essa necessidade de sistemas mais complexos dificultou para as instituições acadêmicas (como universidades) iniciarem programas de satélites, devido ao alto custo e à expertise técnica necessária, mesmo para um satélite de pequeno porte. Essa espiral crescente de complexidade e custo foi o principal impulso para o desenvolvimento de especificações técnicas padronizadas para projetos de satélites miniaturizados de órbita baixa, como o CubeSat.

O padrão CubeSat nasceu de uma colaboração entre os pesquisadores Jordi Puig-Suari, da Universidade Politécnica do Estado da Califórnia (Cal Poly), e Bob Twiggs, da Universidade de Stanford, em 1999. Esse padrão teve um impacto significativo no avanço da pesquisa e desenvolvimento de nano e pico satélites em universidades ao redor do mundo (Pestana, 2022).

Em conformidade, Targino (2021) destaca que um fator importante a ser considerado é que os CubeSats, além de utilizarem os chamados componentes de prateleira (COTS, do inglês, *commercially available off-the-shelf*), aquecem esse mercado e geram uma demanda constante junto às empresas.

Inicialmente, o projeto CubeSat visava criar uma plataforma padronizada com o objetivo de reduzir custos, diminuir o tempo de desenvolvimento, ampliar o acesso ao espaço e permitir lançamentos frequentes de pequenas cargas

úteis (Targino, 2021), porém as especificações do CubeSat acabam por se tornar um padrão internacional devido à sua grande aceitação no meio cientifico e acadêmico. Hoje, pequenas universidades e até mesmo escolas podem desenvolver seus próprios CubeSats (Pestana, 2022).

Particularmente, destaca-se a parceria entre a unidade do Centro Regional do Instituto Nacional de Pesquisas Espaciais (INPE/CRN), situada em Natal, e a Universidade Federal do Rio Grande do Norte (UFRN), que culminou no desenvolvimento do projeto CONASAT. O objetivo principal deste projeto é oferecer uma solução econômica para o Sistema Brasileiro de Coleta de Dados Ambientais (SBCDA) através de uma constelação de nanossatélites (1 kg a 10 kg), empregando tecnologias emergentes nos ramos da eletrônica e das telecomunicações (Queiroz et al., 2018).

Os nanossatélites do projeto CONASAT, seguindo o padrão CubeSat, compreendem os subsistemas de Energia Elétrica (EPS, do inglês, *Electrical Power System*), Determinação e Controle de Atitude (ADCS, do inglês, *Attitude Determination and Control System*), Telemetria e Telecomando (TT&C, do inglês, *Telemetry, Tracking & Command*), além do controle de bordo, junto com a carga útil. O Subsistema de Controle de Bordo, em especial, com o computador de bordo (OBC, do inglês, *On-Board Computer*) como peça central, desempenha uma função essencial no processamento e na coordenação dos dados entre os diferentes subsistemas (Alves et al., 2019).

No seu estudo, Barbosa et al. (2022) menciona que os computadores de bordo de CubeSats requerem capacidade de resposta em tempo real em vários níveis de prioridade para cumprir essas funções. Para alcançar isso, eles podem incorporar sistemas operacionais de tempo real (RTOS, do inglês *Real-time Operating System*).

O RTOS atua como *software* do sistema, gerenciando o uso e o acesso à memória, bem como a outros periféricos dentro do microcontrolador (MCU) (Lumbwe, 2013). O FreeRTOS, uma classe dos RTOS projetada para sistemas reduzidos, tornou-se padrão para RTOS em microcontroladores (Rajulu et al., 2014).

Tendo em vista a importância do OBC para o nanossatélite, há uma necessidade de se projetar um *firmware* com um alto nível de confiabilidade operacional contra as intempéries do ambiente espacial. Com isso, esse trabalho apresenta uma proposta de sistema computacional (*firmware*) baseado no FreeRTOS, desde do projeto de sua arquitetura até a implementação do mesmo em um microcontrolador.

O trabalho está organizado da seguinte maneira: a Seção 2 aborda os conceitos fundamentais para o trabalho; na Seção 3 é discutido o OBC utilizado neste estudo; enquanto a Seção 4 detalha a estrutura do sistema proposto; Já na Seção 5 são apresentados os resultados obtidos e, por fim, na Seção 6 são discutidas as consideração finais.

2. CUBESATS

O padrão CubeSat (Figura 1) define as especificações de tamanho e massa que um nanossatélite deve possuir. A partir de uma unidade básica chamada de 1U, que possui dimensões de 10x10x10 cm e massa de até 1,33 Kg, é possível desenvolver nanossatélites de variados tamanho e massa apenas agrupando-se mais de uma unidade (2U, 3U ou mais) (Pestana, 2022).

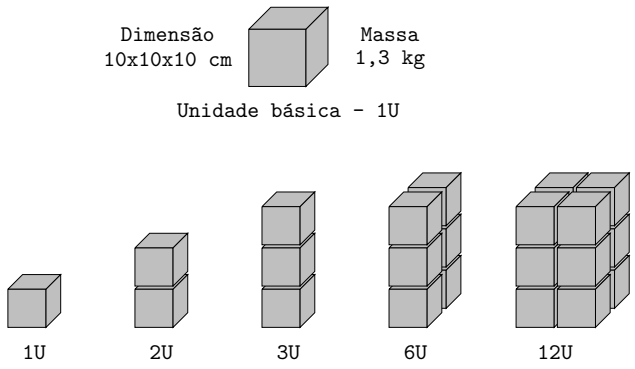

Figura 1. Dimensões de CubeSats

Mota (2022) destaca que além de massa e volume, o padrão CubeSat também definiu uma interface de comunicação entre os subsistemas existentes no satélite, a qual é encontrada facilmente em componentes eletrônicos COTS e em componentes *Rad-Hard*, como *Inter-Integrated Circuit* (I_2C), *Controller Area Network* (CAN) e *Universal Asynchronous Receiver/Transmitter* (UART).

Um Cubesat é formado por módulos, isto é, é dividido em subsistemas que executam diferentes tarefas. Cada subsistema é responsável por um conjunto de tarefas, porém, todos estes são interdependentes formando o sistema complexo do satélite (Pestana, 2022). Estes podem ser vistos, geralmente, da seguinte forma:

- EPS que é responsável pelo fornecimento de energia elétrica provinda dos painés solares;
- TT&C gerencia o envio da telemetria e o recebimento de comandos via radiofrequência (UHF/VHF);
- ADCS faz a determinação e controle de atitude do satélite;
- *Payload* (carga útil) é o experimento da missão;
- OBC permite a comunicação entre os diferentes subsistemas e faz o controle dos mesmos.

Os CubeSats são usados para diversas finalidades, que vão da detecção de sinais eletromagnéticos que antecedem os terremotos a sistemas de sensoriamento de condições atmosféricas, passando pelos testes de sistemas biológicos, como a produção de proteínas bacterianas no espaço, até a observação de fenômenos no solo, entre outras aplicações (Targino, 2021)

2.1 Computador de bordo

O computador de bordo (Figura 2), é um sistema de computação que processa as várias informações transmitidas ao satélite, da estação base ou dos outros subsistemas embarcados. O principal objetivo do OBC é fornecer uma plataforma CDHS (*Command and Data Handling Subsystem*) que faz interface com outros subsistemas de satélite e controla suas operações. Em outras palavras, ele é o principal componente do satélite, podendo ser caracterizado como o cérebro do sistema (Mota, 2022).

O OBC é composto por um elemento central, o processador, juntamente com memória de programa (*Flash*),

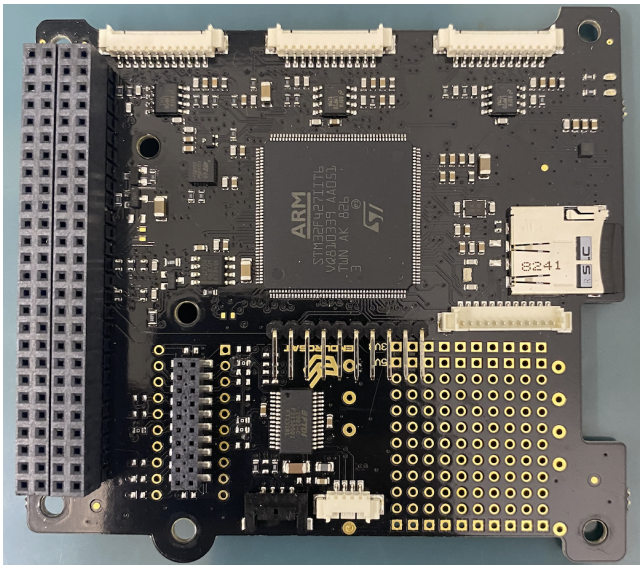

Figura 2. Computador de Bordo do CONASAT-1

memória de execução (SRAM), fonte de alimentação e interfaces de comunicação (Mota, 2022). A escolha do processador afeta diretamente o projeto do OBC e do nanossatélite. CubeSats podem usar várias unidades de processamento, como FPGAs, microcontroladores e plataformas de código aberto como Arduino e Raspberry Pi (Barbosa et al., 2022).

Lumbwe (2013) cita que as principais funções do OBC, são:

- Gravação e armazenamento de telemetria e dados da carga útil do satélite para transmissão à estação base para análise;
- Codificação e decodificação de pacotes de dados entre o satélite e a estação base;
- Processamento de comandos da estação base;
- Monitoramento contínuo dos subsistemas do satélite, implementação de funções de vigilância e reinicialização de subsistemas críticos, se necessário.

2.2 Arquitetura do OBC

O sucesso do desenvolvimento de um OBC está diretamente ligado à elaboração de uma arquitetura bem definida para suas funções. Essa arquitetura representa as partes do sistema (*hardware* ou *software*) e suas interações, com detalhes de implementação abstraídos, mantendo apenas informações de comunicação. Ela tende a ser flexível e de uso geral, visando aumentar a capacidade de reutilização do OBC. No entanto, a confiabilidade das arquiteturas pode variar dependendo dos componentes utilizados e das técnicas de tolerância a falhas adotadas (Mota, 2022).

Em *hardware*, técnicas como redundância fria, redundância quente e redundância de módulo triplo são utilizadas para substituir componentes defeituosos. Por exemplo, dois processadores em um mesmo OBC executam operações sincronizadas, detectando erros do tipo SEU (do inglês, *Single Event Upset*) quando os valores de saída diferem. Para corrigi-los, o sistema pode reiniciar. Já em *software*, uma técnica comum é o desenvolvimento de uma arquitetura modular, onde os módulos possuem baixa dependência entre si. Isso aumenta a tolerância à falhas, uma vez que os erros em um único módulo não se propagam por todo o sistema (Alves et al., 2019).

A arquitetura de *hardware* do OBC a ser empregada neste projeto está detalhada em Barbosa et al. (2022). Como resultado, este trabalho se propõe a desenvolver soluções em *software* que garantam um desempenho satisfatório, promovendo assim uma maior robustez ao sistema.

3. OBC DO CONASAT-1

O projeto CONASAT objetiva oferecer uma opção tecnologicamente mais avançada e a custos reduzidos para garantir a continuidade do SBCDA, através de uma constelação de nanosatélites, que possibilitem melhorar a qualidade do serviço, no que diz respeito à capacidade, abrangência geográfica e tempos de revisita (Carvalho et al., 2013).

Os nanossatélites são projetados para coletar dados ambientais de plataformas remotas em todo o território brasileiro. Esses dados incluem volume de chuva, temperatura, umidade, poluição do ar, correntes oceânicas e perigos ambientais. Posteriormente, os dados coletados são retransmitidos ao centro da missão (Alves et al., 2019).

O primeiro satélite da constelação será o CONASAT-1. Ele está sendo desenvolvido para carregar o EDC (*Environmental Data Collector*), responsável por receber dados das Plataformas de Coleta de Dados (PCDs) em todo o território nacional. Esses dados serão transmitidos por telemetria para a estação base (Barbosa et al., 2022).

Segundo Barbosa et al. (2022), boa parte dos subsistemas do CONASAT-1, foram adquiridos da empresa búlgara EnduroSat e tinham como requisitos:

(1) Deve seguir o padrão CubeSat 1U;
(2) Não deve possuir sistema de propulsão;
(3) Deve conter uma única carga útil, sendo o EDC do INPE;
(4) Deve utilizar antenas omnidirecionais;
(5) Deve ser capaz de monitorar seus próprios parâmetros e os da carga útil;
(6) Deve operar em órbita baixa de até 700 km.

Visando criar um sistema próprio que substitua o OBC da empresa búlgara, o projeto CONASAT já desenvolveu um hardware compatível apresentado no trabalho de Barbosa et al. (2022). O mesmo seguiu várias especificações, conforme descrito abaixo:

(1) Compatibilidade total com o *hardware* e os *softwares* desenvolvidos para projeto CONASAT;
(2) Deve ser construído apenas com componentes COTS;
(3) Deve ser robusto o suficiente para resistir aos efeitos da radiação no ambiente de órbita baixa;
(4) Deve estar preparado para futuras atualizações de *hardware* e *software*;
(5) Deve possuir capacidade de processamento para suportar a inclusão de um módulo de controle de atitude (ADCS) e de filtros para os sensores.

O modelo de MCU escolhido para a unidade de processamento central do projeto foi o STM32F427, um microcontrolador de 32 bits baseado no ARM Cortex-M4 da família STM32. Além disto, o OBC possui seis conectores

destinados aos fotodiodos e magnetorques, os quais são instalados nos painéis solares do CubeSat. Ademais, a Placa de Circuito Impresso (PCB) incorpora um barramento PC/104, um conector designado para permitir a leitura do cartão SD por outros módulos do satélite, um *slot* para cartão Micro SD e um conector destinado ao módulo UHF.

Adicionalmente, o CONASAT é equipado com uma memória *flash* com capacidade de 1 gigabyte, reguladores de tensão de 3,3 volts e 5 volts, um par de acelerômetros e um par de magnetômetros, ambos com redundância fria, e um conjunto de MOSFETs, os quais são empregados para garantir a seletividade dos componentes das placas. O projeto foi concebido de tal maneira que apenas um dos componentes redundantes possa estar ativo em determinado momento, com o objetivo de evitar o aumento do consumo de energia devido à duplicação dos componentes.

3.1 Arquitetura de processamento

A arquitetura empregada no projeto CONASAT adota um barramento de dados compartilhado por todos os seus módulos, caracterizando-se como uma abordagem típica da arquitetura descentralizada, conforme ilustrado na Figura 3. Segundo Alves et al. (2019), essa estratégia faz com que os pontos de falha sejam isolados e, consequentemente, a falha em um subsistema não compromete a operação do sistema como um todo

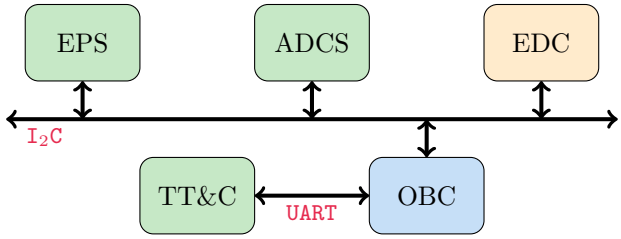

Figura 3. Arquitetura descentralizada do CONASAT

Cada um dos subsistemas do CubeSat é equipado com suas próprias unidades de processamento e interfaces de comunicação. A comunicação entre os subsistemas e o OBC ocorre por meio do protocolo I^2C ou UART. Nesse modelo, o OBC é responsável pelo gerenciamento e processamento do fluxo de dados contidos no barramento I^2C.

4. PROJETO DO FIRMWARE

A arquitetura do *firmware* (*software* embarcado) do OBC é dividida em vários níveis ou camadas (Figura 4), sendo que as camadas inferiores estão mais relacionadas com a organização do próprio *hardware*, como memórias e periféricos. Subindo um pouco no nível das camadas temos a camada de abstração de *hardware* (HAL, do inglês *Hardware Abstract Layer*), essa camada fornece um conjunto simples e genérico de funcionalidades fornecida pelo fabricante a *STMicroelectronics* para todos os periféricos do MCU. Isso mantém a compatibilidade e aumenta a portabilidade do *firmware* desenvolvido.

Usando as funcionalidades das camadas anteriores, podemos criar um conjunto de *drivers* para operar os dispositivos contidos no sistema. Esses *drivers* atuam como interfaces entre o *software* embarcado e o *hardware*, permitindo que o RTOS e os serviços interajam corretamente com os módulos e dispositivos.

No que diz respeito ao *middleware*, neste nível é onde ocorre todo o gerenciamento das atividades (*tasks*) do OBC pelo FreeRTOS. Além disso, é onde ocorre o processamento de dados de entrada e saída do sistema, desde os telecomandos vindos da estação base até os dados dos módulos e sensores. Por fim, a camada de aplicação é onde estão dispostas as aplicações do sistema, as quais ocorrem de maneira preemptiva, com sua execução sendo moderadas pelo RTOS. Cada tarefa deve ter dependência mínima das demais, a fim de evitar a propagação de erros em casos de falhas.

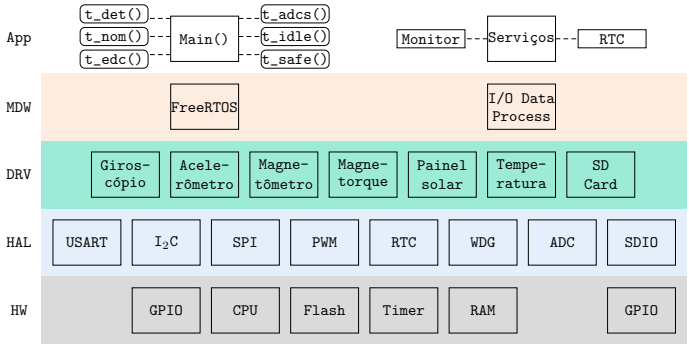

Figura 4. Arquitetura do *firmware* do OBC

4.1 Operação do sistema

A operação do sistema é totalmente gerida pelo RTOS. Com isso, cada tipo de operação do OBC é definido dentro de uma tarefa (*task*). No projeto CONASAT, seus satélites operam com base em seis modos de operação, sendo estes: *Control*, *Detumbling*, *Idle*, *Nominal*, *Payload* e *Safe*.

Cada um destes opera da seguinte maneira:

- *Control*: Neste modo, o controle de altitude é realizado, ativando-se apenas o ADCS;
- *Detumbling*: Este modo é responsável pelo ajuste do momento angular do CubeSat logo após o *deploy*, ou seja, a primeira etapa após ser colocado em órbita;
- *Idle*: Todos os subsistemas devem operar normalmente neste modo, e o OBC deve gerenciar todos os dados recebidos destes. Este estado serve como um estado de transição, e a saída deste modo é feita por meio de telecomandos vindos da estação de controle.
- *Nominal*: Todos os subsistemas devem estar operando, além de ativar o subsistema da carga útil.
- *Payload*: A carga útil (EDC) é o único subsistema que deve estar operando neste modo.
- *Safe*: Este modo é ativado sempre que houver uma situação crítica, mantendo ativos apenas os subsistemas fundamentais para o funcionamento do satélite, como por exemplo o EPS. O modo também é usado no momento de inicialização do sistema.

A Figura 5 representa a operação do sistema por meio de uma máquina de estados. Nela, são mostrados os possíveis estados e suas conexões. Além disso, é possível notar as variáveis t_c e c_m que são responsáveis pelas transições entre os modos de operação. A váriavel t_c representa um telecomando vindo da estação de controle, enquanto a

variável c_m indica um comando do próprio sistema. Por fim, na Figura 5 é representado por meio de uma seta apenas com indicação de destino para o estado em que o sistema deve inicialmente operar, correspondendo ao modo *safe*.

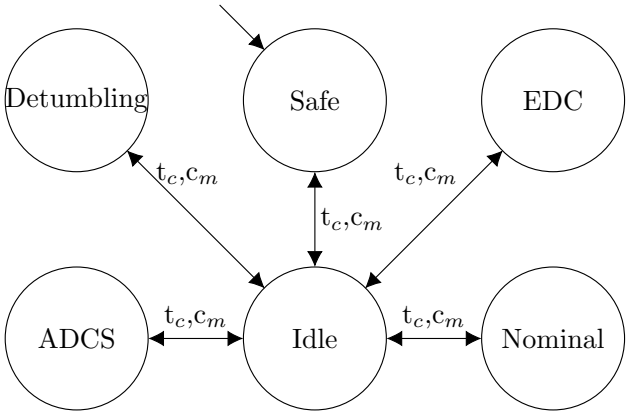

Figura 5. Máquina de estados do sistema

5. RESULTADOS PARCIAIS

Com o propósito de verificar o *firmware* em desenvolvimento antes de aplicá-lo no OBC real, foram realizados testes utilizando o *firmware* do OBC carregado na plataforma de desenvolvimento STM32 Nucleo-64, equipada com um MCU STM32F446. Para estimular o *firmware* de forma semelhante ao caso real, foram emulados dois subsistemas do CONASAT-1, o subsistema de Telemetria e Telecomando (TT&C) e o subsistema de Energia Elétrica (EPS), cada um em um microcontrolador ATMEGA328P. Para supervisionar o teste, um PC era conectado ao emulador e ao OBC para auxiliar na depuração de toda a comunicação trocada entres os subsistemas conforme as Figura 6 e 7.

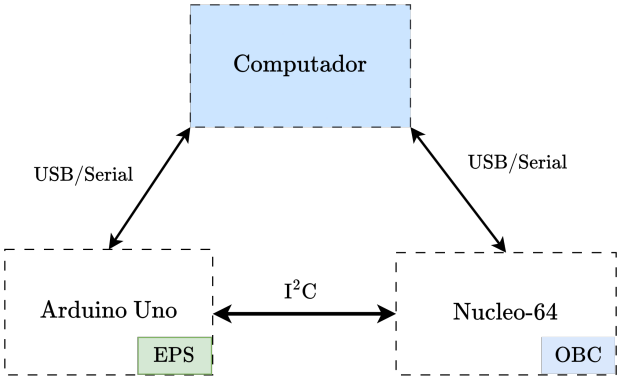

Figura 6. Diagrama de comunicação entre os sistemas

A comunicação entre os subsistema foi realizada através de uma conexão UART e outra I_2C, sendo a primeira entre o TT&C e o OBC, já a segunda entre o OBC e a EPS. Toda a comunicação entre os subsistemas seguiram o protocolo ESTTC *(EnduroSat's Telemetry and Telecommand)* desenvolvido pela EnduroSat. Este protocolo formata os comandos de maneira que os 3 primeiros *bytes* são destinados ao cabeçalho do comando e são representados por

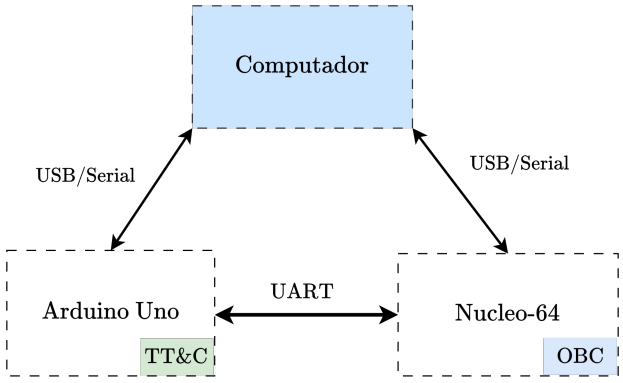

Figura 7. Diagrama de comunicação entre os sistemas

'ES+' em ASCII. O *byte* seguinte carrega o sinal do tipo do comando enviado/recebido e recebe o valor de `W` para escrita de dados, `R` para leituras e `C` para demais comandos. Além disso, há um *byte* para o endereço do dispositivo que será realizada a operação, há dois *bytes* referentes ao comando enviado e, logo em seguida, os *bytes* dos dados recebidos/enviados. Vale lembrar que os comandos com o sinal `C` não devem ter nenhum retorno. A Figura 8 expressa o formato especificado do protocolo e apresenta um exemplo de comando de escrita de dados em um dispositivo endereçado em 0x11.

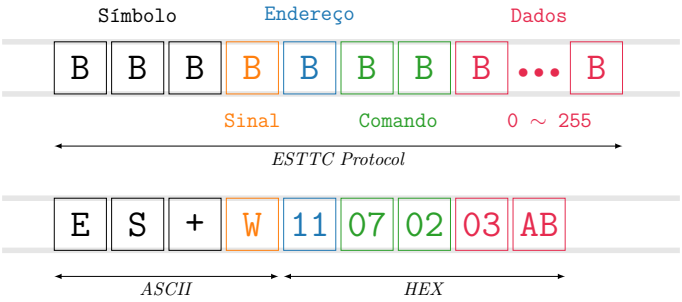

Figura 8. Protocolo ESTTC

Inicialmente, foram realizados testes no sistema contendo apenas o OBC e o EPS emulado. O *firmware* do OBC utiliza o FreeRTOS como sistema operacional e foram desenvolvidos um conjunto de códigos na linguagem de programação C para implementarem uma série de tarefas que lidavam com a comunicação (envio de comandos de leitura e escrita) e o processamento dos dados. Essas tarefas configuravam os periféricos do OBC, monitoravam o fluxo de dados no barramento I^2C e enviavam os resultados do recebimento de dados via serial para o PC. Já para o EPS foi desenvolvido um código em C++ que identificava se o comando enviado pelo OBC era de leitura ou escrita e indicava o comando enviado ao PC. Caso fosse de leitura, o EPS emulado enviava um valor padrão como resposta à solicitação do OBC e, por fim, enviava essas informações ao PC via serial. A Figura 6 demonstra o esquema de comunicação utilizado.

Nos testes, caso a comunicação fosse bem-sucedida, os dados enviados e recebidos pelos subsistemas poderiam ser observados nas portas seriais/USB de um PC. Em caso de falha, uma mensagem de erro era apresentada. As Figuras 9 e 10 mostram o resultado do envio de um comando de

leitura seguido por um comando de escrita. Esse comandos podem ser consultados em ENDUROSAT (2019).

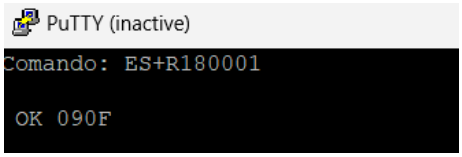

Figura 9. Dados recebidos pelo OBC

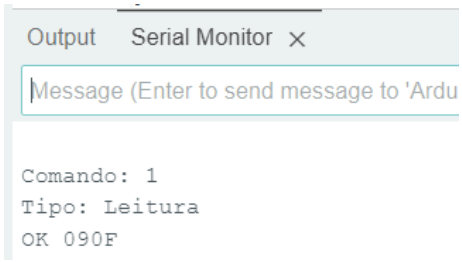

Figura 10. Dados enviados e recebidos pelo EPS

Um segundo teste foi realizado com o intuito de validar a comunicação entre o subsistema TT&C e o OBC no que diz respeito às mudanças de modos de operação. Para isso, o subsistema TT&C foi emulado em um microcontrolador e, este, enviava comandos via UART para o OBC, conforme descrito na Figura 7.

A Figura 11 apresenta o resultado do teste da comunicação entre o TT&C e o OBC. Na Figura 11 podemos ver os modos atuais do OBC sendo alterados conforme o recebimento de alguns telecomandos. Além disto, é possível notar que o OBC inicializa no modo *Safe*, conforme esperado, e realiza a troca para o modo *Nominal* passando pelo modo *Idle*.

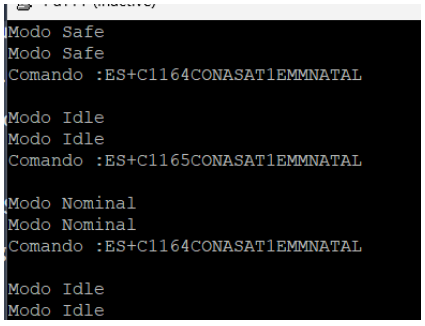

Figura 11. Dados recebidos do OBC via UART

Por fim, foi realizado um teste envolvendo a aquisição de dados do EPS emulado, utilizando comandos enviados a partir do subsistema TT&C emulado, todos gerenciados pelo OBC. Nesse teste, o modo de operação do OBC foi alterado para o modo *Nominal* por meio de comandos recebidos do TT&C. Após o OBC estar no modo desejado, foi solicitado o dado ao EPS por meio de um comando de leitura. O EPS retornou o dado ao OBC, que então o enviou ao TT&C no formato "OK VVVV", onde VVVV são os valores do dado em hexadecimal.

Como exibido nas Figuras 12 e 13, o modo de operação do OBC foi alterado até atingir o estado *Nominal* por meio de comandos. O dado solicitado ao EPS, via comando ES+R18001 (*Batt. BUS voltage*), foi retornado e formatado conforme o padrão esperado de retorno do protocolo ESTTC.

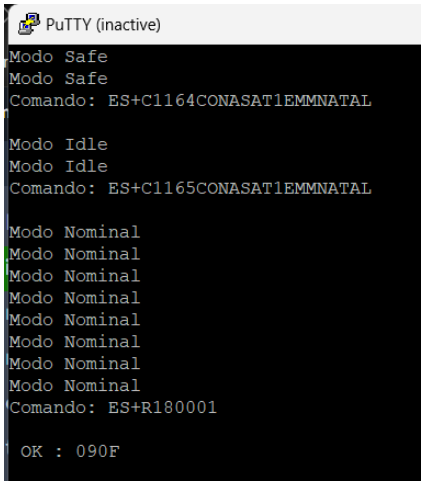

Figura 12. Dados recebidos do OBC via UART

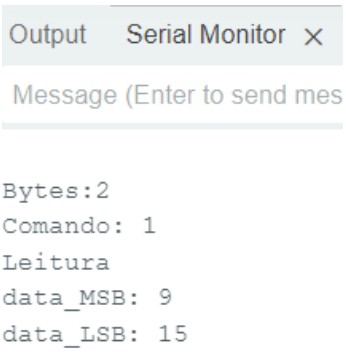

Figura 13. Dados enviados pelo EPS

6. CONCLUSÃO

Este trabalho atuou em mais uma etapa do desenvolvimento do sistema de computação de bordo para nanossatélites no padrão CubeSat do projeto CONASAT. Nesse texto, o foco foi o *software* embarcado (*firmware*) no OBC, o qual ainda está em desenvolvimento. Porém, foram apresentados resultados iniciais do *firmware* em funcionamento com dois dos principais subsistemas (TT&C e EPS) do CONASAT-1 emulados em hardware. Com os testes foi possível validar o gerenciamento de modos do OBC, a comunicação com o subsistemas de telemetria e telecomando e a comunicação com o subsistema de gerenciamento de energia elétrica.

Os próximos passos do desenvolvimento do *firmware* serão a integração do OBC com os demais subsistemas e a carga útil, bem como testes do *firmware* embarcado na PCB projetada para o OBC e conectada aos demais subsistemas reais.

AGRADECIMENTOS

Agradecemos ao apoio do INPE/CRN e da equipe do projeto CONASAT.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

REFERÊNCIAS

Alves, A., Dias, S.M., e Queiroz, K.I. (2019). Desenvolvimento de um sistema para controle de bordo de nanossatélite. In *Anais do 14º Simpósio Brasileiro de Automação Inteligente*. Galoá, Ouro Preto.

Barbosa, C., Dias, S., e Queiroz, K. (2022). Computador de bordo para nanossatélites em Órbita baixa no padrão cubesat. *XXIV Congresso Brasileiro de Automática*. doi:10.20906/CBA2022/3741.

Botma, P.J. e Steyn, W.H. (2011). *Design and Development of an ADCS OBC for a CubeSat*. Master's thesis, Stellenbosch University, Stellenbosch, Cabo Ocidental, África do Sul.

Carvalho, M.J.M., Lima, J.S.S., Jotha, L.S., e Aquino, P.S. (2013). Conasat - constelação de nano satélites para coleta de dados ambientais. *XVI Simpósio Brasileiro de Sensoriamento Remoto*.

ENDUROSAT (2019). *User Manual - Electrical Power System (EPS I & EPS I Plus) - I2C Protocol*.

Lumbwe, L.T. (2013). *Development of an onboard computer (OBC) for a CubeSat*. Master's thesis, Cape Peninsula University of Technology, Cidade do Cabo, África do Sul.

Mota, D.F.M. (2022). *Robust OBC: um sistema de computação de bordo tolerante a falhas*. Ph.D. thesis, Universidade Federal do Ceará, Fortaleza-CE, Brasil.

Pestana, E.P. (2022). *Metodologia Para Detecção e Correção de Erros Causados Por Radiação em Computador de Bordo de Um Cubesat*. Master's thesis, Universidade Federal do Maranhão, São Luís-MA, Brasil.

Queiroz, K.P., Dias, S.M., Duarte, J.M., e Carvalho, M.M. (2018). Uma solução para o sistema brasileiro de coleta de dados ambientais baseada em nanossatélites. *HOLOS*, 7, 132–142. doi:10.15628/holos.2018.6307.

Rajulu, B., Dasiga, S., e Iyer, N.R. (2014). Open source rtos implementation for on-board computer (obc) in studsat-2. In *2014 IEEE Aerospace Conference*, 1–13. doi:10.1109/AERO.2014.6836377.

Souza, K.V.C.K., Bouslimani, Y., e Ghribi, M. (2022). Flight software development for a cubesat application. *IEEE Journal on Miniaturization for Air and Space Systems*, 3(4), 184–196. doi:10.1109/JMASS.2022.3206713.

Targino, H.C. (2021). Desenvolvimento e implementação de um de plano AIT para nanosatelites. *Mens Agitat*, 16, 98–101.