Proposta de um Sistema para Simulação em Hardware de Nanossatélites

Alex C. R. Alves, Samaherni M. Dias e Kurios I. P. M. de Queiroz
Laboratório de Automação, Controle e Instrumentação (LACI)
Universidade Federal do Rio Grande do Norte (UFRN), Natal, Brasil
alexcarlos27@gmail.com, sama@laci.ufrn.br, kurios@laci.ufrn.br

***Resumo*—O advento da microeletrônica contribuiu para a criação de um padrão de nanossatélites, chamado CubeSat, que possibilitou uma redução no custo e no tempo de desenvolvimento de missões baseadas em satélites. Com a popularização desse padrão, várias instituições de ensino passaram a desenvolver pesquisas utilizando CubeSats. Para fins didáticos, foram criadas plataformas para simulação de nanossatélites em ambientes educacionais. Apesar dessas plataformas oferecerem uma alternativa para o ensino e o desenvolvimento da engenharia aeroespacial, elas apresentam um custo relativamente alto. Desse modo, neste artigo é apresentada uma proposta de um sistema para simulação de nanossatélites utilizando sistemas embarcados de baixo custo e uma plataforma IoT. A arquitetura do sistema proposto é descrita e são apresentados os resultados de alguns testes já realizados. Por fim, são apresentadas as conclusões e proposições para trabalhos futuros.**

***Palavras-chave*—CubeSats, IoT, Sistemas Embarcados, MQTT**

I. Introdução

Os recentes avanços tecnológicos na área da eletrônica e da computação têm colaborado com o desenvolvimento de novas tecnologias para diferentes aplicações. Nas telecomunicações, o conceito de "Internet das Coisas" (IoT, em inglês, *Internet of Things*) ganhou popularidade graças à evolução tecnológica. Na esfera espacial, desde o lançamento do primeiro satélite artificial do mundo, Sputnik I, em 1957, satélites cada vez menores têm sido colocados em órbita como resultado de pesquisas que permitiram a construção de componentes eletrônicos de tamanho reduzido.

Geralmente, a classificação dos satélites é realizada de acordo com a sua massa [1]. Apesar de não existir um consenso com relação a definição do termo "pequenos satélites", grande parte da comunidade científica o utiliza para caracterizar satélites com massa inferior a 1000 kg [1], [2]. Os pequenos satélites são ainda subdivididos em: minissatélites (massa entre 100 kg e 1000 kg); microssatélites (massa entre 10 kg e 100 kg); nanossatélites (massa entre 1kg e 10 kg); picossatélites (massa entre 0,1 kg e 1kg); e femtossatélites (massa entre 1 g e 100 g) [3].

No ano de 1999, os professores Jordi Puig-Suari da Universidade Politécnica da Califórnia (Cal Poly) e Bob Twiggs da Universidade Stanford iniciaram o projeto CubeSat, com objetivo de ampliar a acessibilidade ao espaço ao oferecer um padrão para projetos de nano e picossatélites com custo reduzido e com tempo de desenvolvimento menor [4].

Nesse mesmo ano, na área da computação e das telecomunicações, o termo "Internet das Coisas" começou a ganhar popularidade após ser usado por Kevin Ashton durante uma apresentação [5]. O conceito inicialmente proposto tratava de objetos conectados interoperáveis identificados por tecnologia de identificação por radiofrequência (RFID) [6]. Contudo, a definição de IoT foi ampliada e passou a representar uma visão em que objetos do dia a dia são "abraçados" pela Internet. De acordo com essa visão, itens físicos podem atuar como ponto de acesso fixo à serviços de Internet e podem ser controlados remotamente, de modo que não se encontram mais desconectados do mundo virtual [7]. Assim como os CubeSats, o conceito de IoT é resultado do desenvolvimento da microeletrônica e de outras tecnologias, entre elas: identificação, sensoriamento, comunicação, processamento embarcado de informações e redes de computadores [6]–[8].

Os CubeSats e a Internet das Coisas, a princípio, podem parecer desconexos. Contudo, em trabalhos recentes, o conceito de IoT tem sido expandido para aplicação com CubeSats. Em [9] e [10] é descrita uma infraestrutura de rede, cujo elemento central são CubeSats, capaz de fornecer conectividade global. Em [11] é apresentada uma arquitetura IoT para estações terrestres, conectadas em rede, com ferramentas e recursos para comunicação com CubeSats em órbita. Embora esses trabalhos sirvam para ilustrar a utilização de CubeSats em conjunto com IoT, eles apresentam aplicações complexas que fogem do escopo deste artigo.

Atualmente, uma prática adotada no desenvolvimento de CubeSats e de sistemas IoT é a utilização de componentes comerciais de prateleira (COTS, em inglês, *Commercial off-the-Shelf*), como microprocessadores, microcontroladores e sistemas embarcados. O uso desses componentes torna o processo de desenvolvimento e construção de CubeSats mais barato, simples e rápido comparado ao uso de *hardware* específico para aplicações espaciais [2]. Em IoT, os sistemas embarcados são utilizados, por exemplo, na aquisição de dados, no controle de sensores conectados em rede ou no gerenciamento da informação em uma arquitetura IoT.

Diante desse contexto, a partir da parceria entre o Centro Regional do Nordeste (CRN) do Instituto Nacional de Pesquisas Espaciais (INPE) e a Universidade Federal do Rio Grande do Norte (UFRN), no desenvolvimento do projeto CONASAT (Constelação de Nanossatélites para Coleta de Dados Ambientais) [12], surgiu a ideia de aplicar IoT e sistemas

embarcados na construção de um sistema para simulação em *hardware* de nanossatélites. Tal sistema deve ser utilizado para testes de CubeSats do projeto CONASAT, bem como para o desenvolvimento de atividades de pesquisa e ensino na área de engenharia aeroespacial na UFRN.

Neste artigo, é apresentada a proposta de um sistema de simulação em *hardware* de nanossatélites, baseado em tecnologias embarcadas e IoT. A arquitetura do sistema é descrita e são apresentados alguns resultados obtidos durante dois experimentos iniciais para validação da comunicação entre uma plataforma IoT e, primeiro caso, dispositivos embarcados responsáveis por simular subsistemas de um CubeSat, segundo caso, um CubeSat real.

II. Fundamentação Teórica

A. CubeSats

Nos anos posteriores ao lançamento do Sputnik I, houve um desenvolvimento intenso de missões espaciais organizadas por entidades militares e governamentais pertencentes a nações mais avançadas tecnologicamente e ricas economicamente. O alto custo das missões, entre outros fatores, tornava essas entidades detentoras do capital financeiro e intelectual necessário para construção de satélites e veículos lançadores. Desse modo, o acesso privilegiado ao espaço fez com que essas nações obtivessem uma vantagem em relação as demais, resultando em uma posição de superioridade desfrutada por cerca de quatro décadas [13].

O cenário de polarização da exploração espacial começou a mudar no início dos anos 80 com o advento da microeletrônica, que permitiu que equipes pequenas fossem capazes de construir satélites fisicamente menores utilizando instalações modestas [13]. Apesar de pequenos satélites terem sido lançados desde o início da Era Espacial (o Sputnik I possuía uma massa de 84 kg), houve uma inovação no pensamento de como desenvolver projetos de maneira mais simples, mais barata, mais rápida e com baixo custo, usando componentes disponíveis em computadores e nas telecomunicações [14]. Esse novo modo de pensar culminou na criação do padrão CubeSat.

O CubeSat é descrito como sendo um satélite em forma de cubo com 10 cm de aresta e até 1,33 kg de massa [4], o que representa uma unidade básica chamada 1U (Fig. 1). CubeSats de tamanhos maiores são obtidos pela junção de várias unidades básicas, resultando, por exemplo, em configurações 2U, 3U, 6U ou 8U.

Figura 1. Padrão CubeSat 1U. Adaptado de [15].

Um CubeSat contém todos os subsistemas necessários para execução de uma missão espacial. Entre esses subsistemas, tem-se: subsistema de energia (EPS, *Electrical Power System*), subsistema de telemetria e telecomando (TT&C, *Telemetry, Tracking and Telecommand*), subsistema de determinação e controle de atitude (ADCS, *Attitude and Determination Control System*), computador de bordo (OBC, *On-Board Computer*) e carga útil (*Payload*) [1].

Nos primeiros anos após a criação do padrão CubeSat, poucos satélites desse tipo foram colocados em órbita. Contudo, com a popularização do padrão e com a evolução dos componentes eletrônicos, várias instituições passaram a realizar pesquisas e desenvolver missões baseadas em CubeSats. Atualmente, diferentes tipos de instituições de ensino, agências governamentais e grupos comerciais possuem um programa espacial graças aos CubeSats [15].

Segundo dados apresentados em [16], desde o ano 2000 até o presente momento (julho de 2019), foram colocados em órbita cerca de 1058 objetos desse tipo (Fig. 2).

Entre as aplicações de CubeSats, tem-se: observação da terra e do clima; meio ambiente e agricultura; recursos minerais e hídricos; defesa; controle de fronteiras; educação e treinamento rápido de recursos humanos; desenvolvimento tecnológico; e, telecomunicações [17].

Considerando que os CubeSats têm proporcionado grandes benefícios para comunidade científica e para a população em geral, a construção de simuladores se mostra como uma alternativa de baixo custo para o ensino da tecnologia espacial aplicada em satélites desse tipo. Além disso, tais simuladores podem também ser utilizados no desenvolvimento de missões, no teste e na construção de CubeSats reais.

B. Simuladores de CubeSats

Simuladores de CubeSats têm sido desenvolvidos há alguns anos. Eles têm sido utilizados para fins educacionais, para introdução de conceitos básicos de nanossatélites e para o planejamento de missões espaciais.

Em [18], Mark Spencer apresenta um simulador de CubeSats baseado no microcontrolador PIC, resultado de anos dedicados ao ensino de tecnologias espaciais. Nesse trabalho, ele descreve os componentes do simulador e apresenta imagens

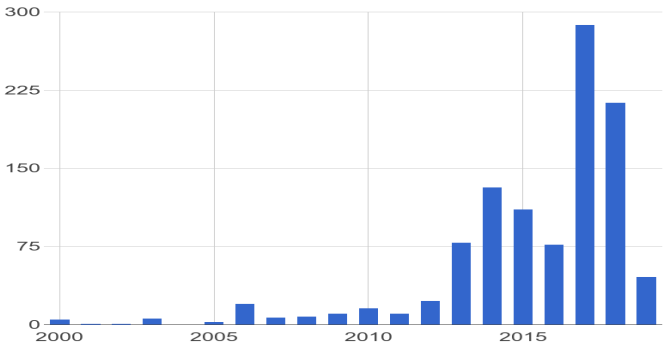

Figura 2. CubeSats lançados a cada ano. Gráfico gerado online com dados do *CubeSat Database* [16].

dos componentes eletrônicos desenvovidos. Em [19], Spencer sugere algumas atividades que podem ser desenvolvidas com seu simulador.

Outro simulador aplicado como ferramenta educacional é descrito em [20]. Ele possui os principais subsistemas de um CubeSat e permite a integração de cargas úteis e outros subsistemas desenvolvidos por usuários. Além disso, os sistemas são capazes de receber comandos e gerar telemetria por meio de uma interface fornecida com o simulador.

Uma abordagem mais recente é apresentada em [21], onde foram utilizados componentes comerciais, como o *Raspberry Pi*, para criação de um simulador de CubeSat para uso como ferramenta educacional, de treinamento ou desenvolvimento de projetos para construção de um modelo de CubeSat real.

Embora os simuladores citados sirvam como base para o ensino da engenharia aeroespacial e para o desenvolvimento de missões com CubeSats, as limitações de *hardware* detectadas em [18] e [21], os quais não são capazes de serem empregados no desenvolvimento de novas plataformas, bem como o alto custo da plataforma apresentada em [20] tornam inviável a aplicação desses simuladores ao projeto CONASAT e às pesquisas a serem realizadas na UFRN.

C. Plataformas IoT

A expansão do conceito de Internet das Coisas fez com que várias tecnologias fossem desenvolvidas para possibilitar o acesso de usuários a dispositivos conectados à Internet. Entre essas tecnologias, tem-se as plataformas IoT.

Tais plataformas consistem de vários componentes necessários para coletar, monitorar, gerenciar e analisar dados de dispositivos conectados em rede [22]. Elas oferecem um infraestrutura que permite a interação de usuários com dispositivos inteligentes [23]. Para isso, são utilizados serviços de processamento e armazenamento de dados e protocolos de comunicação.

Um protocolo bastante utilizado em plataformas IoT é o MQTT (*Message Queuing Telemetry Transport*). Ele é um protocolo de rede leve e flexível que usa um padrão de *publish/subscribe* (publicação e assinatura). Nesse protocolo são definidas duas entidades principais: um *broker* e um número de clientes. O *broker* é um servidor responsável por receber todas as mensagens dos clientes e por disponibilizar essas mensagens aos clientes relevantes. O cliente é qualquer dispositivo capaz de enviar ou receber mensagens do *broker* [24].

Os dispositivos interessados em receber mensagens do *broker* devem assinar (subscribe) tópicos específicos. Desse modo, eles são informados pelo *broker* sempre que ocorre uma publicação no tópico assinado. O cliente responsável por publicar as mensagens atua como gerador de dados importantes [25]. Na Fig. 3, observa-se a arquitetura do protocolo MQTT.

Alguns sistemas embarcados com módulo Wi-FI possuem suporte a esse protocolo, o que torna possível a interação desses sistemas com plataformas IoT que também o utilizam.

III. Sistema para Simulação de CubeSats

O sistema para simulação de CubeSats é composto por uma plataforma IoT para controle e monitoramento [26] e por dispositivos embarcados programados para simular diferentes subsistemas de um nanossatélite. A plataforma permite o armazenamento e o processamento de dados na nuvem e possibilita a visualização dos dados na Internet. Os detalhes do sistema são apresentados a seguir.

A. Arquitetura do Sistema

A arquitetura do sistema pode ser observada na Fig. 4. Cada subsistema real de um CubeSat pode ser simulado em *hardware* por um dispositivo embarcado (microcontrolador, microprocessador) com ou sem interface de comunicação sem fio, Wi-Fi. Vários dispositivos são conectados e se comunicam por meio de um barramento I^2C, de modo a representar a estrutura principal de um nanossatélite. Os dispositivos com módulo Wi-FI são capazes de receber e enviar dados para uma plataforma na nuvem utilizando o protocolo de rede MQTT. Tais dados são disponibilizados *online* para que possam ser observados em computadores, remotamente, via Internet. Além disso, o sistema possibilita a conexão de subsistemas reais ao barramento I^2C, o que permite a interação entre componentes simulados e componentes reais.

B. Plataforma IoT e Dispositivos Embarcados

A plataforma IoT possibilita a criação de panéis (*dashboards*) para visualização dos dados enviados pelos subsistemas simulados. Além disso, também é possível enviar comandos (telecomandos) da plataforma para algum dos subsistemas que possuem módulo Wi-Fi.

Com o auxílio das ferramentas (*widgets*) disponíveis na plataforma, podem ser criados *dashboards* para cada um dos dispositivos utilizados na simulação do nanossatélite. Desse modo, pode ser realizado o monitoramento e o controle de componentes específicos do sistema, o que possibilita a indução de erros aos dispositivos, de modo a avaliar o comportamento do satélite.

Vários tipos de microcontroladores ou microprocessadores podem ser programados para simular os subsistemas reais. Contudo, o dispositivo encarregado de simular o TT&C deve possuir, necessariamente, módulo Wi-Fi embarcado, uma vez que o envio de telecomando ao satélite simulado ocorre por

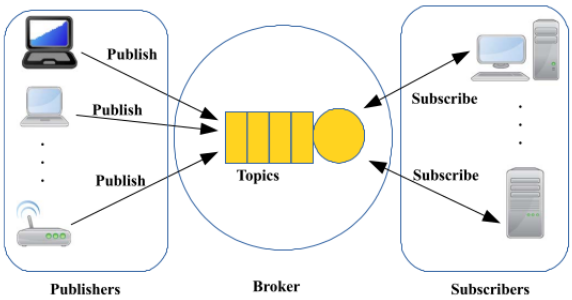

Figura 3. Arquitetura do protocolo MQTT [25].

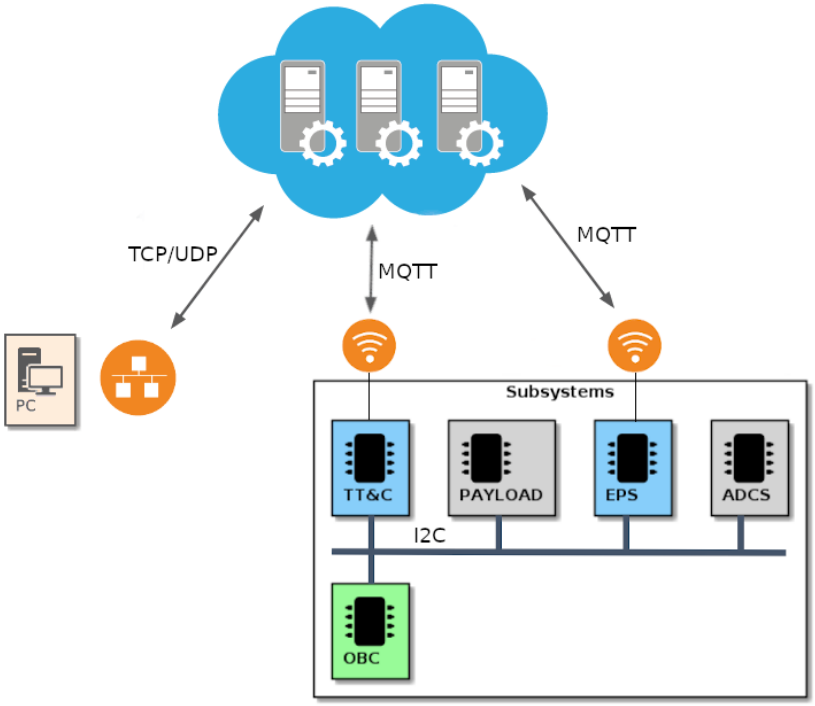

Figura 4. Arquitetura do Sistema. Os blocos em azul (TT&C e EPS) representam dispositivos com suporte a Wi-Fi, os blocos em cinza (PAYLOAD e ADCS) representam dispositivos sem módulo Wi-Fi embarcado e o bloco em verde representa um subsistema real. Os componentes com Wi-Fi enviam e recebem dados para um servidor na nuvem usando MQTT. Os dados podem, então, ser acessados por um dispositivo (PC) conectado a Internet. (Adaptado de [27]).

meio dessa interface de comunicação. Além disso, os dados de telemetria também são obtidos por meio da interface Wi-Fi.

Em um CubeSat real, o OBC é responsável pelo controle e armazenamento dos dados provenientes dos demais subsistemas. No sistema simulado, essa função deve ser desempenhada por qualquer dispositivo embarcado capaz de atuar como "mestre" na comunicação pelo barramento I^2C. Desse modo, os outros dispositivos são configurados como "escravos".

O uso de sistemas embarcados com suporte a sistemas operacionais, como Linux e *FreeRTOS* [28], facilita o desenvolvimento dos sistemas simulados e os torna mais robustos. O *FreeRTOS*, por exemplo, possui recursos que permitem a criação de tarefas (*tasks*) que são executadas de forma independente de outras aplicações do sistema. Desse modo, cada *task* funciona como um pequeno programa que executa, geralmente, em um *loop* infinito [28], [29].

IV. Procedimentos e Resultados

A primeira etapa realizada no desenvolvimento do sistema proposto foi a validação da comunicação entre a plataforma IoT e os dispositivos embarcados. Para isso, dois experimentos foram desenvolvidos com base no SoC (*System on Chip*) ESP32 [30], que possui Wi-Fi e Bluetooth integrados, duas interfaces I^2C e suporte para o sistema operacional em tempo real *FreeRTOS* e para o protocolo de rede MQTT. Esse SoC foi utilizado para simulação do subsistema de telemetria e telecomando.

Inicialmente, foi criado um *dashboard* na plataforma IoT, onde foram adicionados dois *widgets*: um terminal para envio de mensagens (telecomando) para o ESP32 e uma tabela para exibição dos dados recebidos do dispositivo (telemetria). Além disso, foram feitas configurações na plataforma para a obtenção da chave de acesso utilizada na comunicação com o ESP32 via MQTT.

Em seguida, utilizando um *framework* de desenvolvimento para o ESP32 [31], foi desenvolvido o *software*, na linguagem de programação C, responsável pela simulação de um TT&C com comunicação *Full-Duplex* (envio e recebimento de dados de forma simultânea). Para isso, as duas interfaces I^2C do ESP32 foram configuradas para operar em modo "escravo" (com endereços distintos) e foram criadas, por meio dos recursos do *FreeRTOS*, duas tarefas (*tasks*) principais. A primeira tarefa foi projetada para funcionar como receptora dos telecomandos enviados, por MQTT, da plataforma para o ESP32 e para transmiti-los ao OBC (dispositivo mestre), por meio do barramento I^2C, quando solicitado. Para essa tarefa, foram implementados comandos I^2C de acordo com a Tabela I. A segunda tarefa foi desenvolvida para atuar como transmissora de telemetria do ESP32 para a plataforma na nuvem, bem como para receber do OBC por I^2C os dados a serem enviados via MQTT. Além disso, o ESP32 foi programado para conectar, durante a inicialização, à uma rede Wi-Fi pré-configurada e em seguida assinar o tópico MQTT onde os telecomandos da plataforma IoT foram publicados.

No primeiro experimento, um Arduino foi utilizado como dispositivo mestre e foi programado para enviar os comandos apresentados na Tabela I. Desse modo, inicialmente, o co-

Tabela I
COMANDOS I^2C DA TAREFA DE RECEPÇÃO

Comando	Descrição
0x21 (HEX)	Retorna 1 se houver telecomando a ser enviado para o OBC e 0 caso contrário
0x22 (HEX)	Retorna o Telecomando armazenado no ESP32, se houver
0x24 (HEX)	Apaga o Telecomando armazenado no ESP32, se houver

mando 0x21 (hexadecimal) foi enviado ao ESP32 e o valor de resposta foi verificado. O *software* do Arduino foi projetado para repetir o comando, após alguns segundos, em caso de resposta igual a 0 (ausência de telecomando) e para enviar o comando 0x22 em caso de resposta igual a 1 (telecomando disponível). Nesse caso, o ESP32 retornou para o Arduino o telecomando armazenado, o qual foi utilizado para transmitir à plataforma telemetrias de acordo com a Tabela II. Por fim, o telecomando armazenado no ESP32 foi apagado após receber o comando 0x24 do Arduino.

Na Fig. 5, pode ser observada a sequência de informações enviadas pelo barramento I^2C no momento em que havia telecomando armazenado no ESP32.

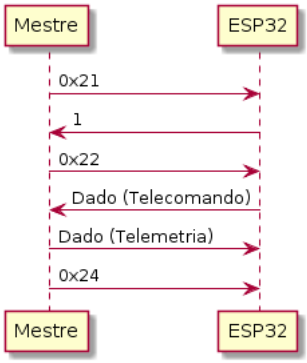

Figura 5. Sequência de comunicação entre o dispositivo mestre e o ESP32.

Os telecomandos foram enviados para o ESP32 utilizando o terminal criado na plataforma IoT. O resultado desse experimento pode ser verificado na Fig. 6. Observa-se que as telemetrias recebidas como resposta ao envio de telecomandos correspondem àquelas apresentadas na Tabela II.

No segundo experimento, o ESP32 foi ligado ao barramento I^2C da estrutura apresentada na Fig. 7, que corresponde a parte de um modelo de desenvolvimento de CubeSats instalado no laboratório do projeto CONASAT no INPE/CRN. Nessa estrutura, estão empilhados os seguintes subsistemas utilizados em CubeSats reais: um OBC com Linux embarcado e configurado como dispositivo mestre da comunicação I^2C; um EPS responsável por fornecer energia à estrutura; e, um TT&C que possibilita envio de telemetria e recebimento de telecomando por radiofrequência (UHF/VHF)

Nesse experimento, o *software* utilizado no OBC foi similar àquele descrito no experimento com o Arduino. A sequência de comandos enviados ao ESP32 foi a mesma do teste anterior, porém, nesse caso, o *software* foi desenvolvido para que o OBC pudesse obter dados do EPS de acordo com o telecomando recebido. Portanto, durante o experimento, após receber telecomando, o OBC retornou ao ESP32 informações como tensão (em milivolts) e temperatura (em °C) da bateria do EPS. Essas informações foram então transmitidas do ESP32 para a plataforma IoT como telemetria. Na Tabela II são apresentadas as telemetrias de acordo com o telecomando recebido pelo OBC. Observa-se que o x foi utilizado para representar o valor da tensão ou da temperatura. O resultado desse experimento pode ser verificado na Fig. 8.

Tabela II
TELEMETRIA ENVIADA PELO MESTRE

Telecomando	Telemetria	
	Arduino	***OBC***
101	cmd101	Tensao (mV): x
102	cmd102	Temperatura x
103	invalido	Telecomando Invalido
qualquer valor	-	Telecomando Invalido

V. CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho foi apresentada a proposta de um sistema para simulação em *hardware* de nanossatélites utilizando sistemas embarcados e tecnologias IoT. A primeira etapa realizada, como parte da validação da proposta, foi o teste de comunicação entre uma plataforma IoT e um SoC encarregado de simular o funcionamento de um subsistema de telemetria e telecomando real. Os testes realizados demonstraram ser viável o envio de telecomandos da plataforma IoT para os subsistemas simulado e real por meio do protocolo de rede MQTT. Além disso, também foi verificada a possibilidade de visualizar, na plataforma, dados de telemetria recebidos dos subsistemas por meio do mesmo protocolo.

Os testes realizados integram um ponto de partida no desenvolvimento do sistema. Desse modo, outros experimentos e implementações devem ser realizados com intuito de

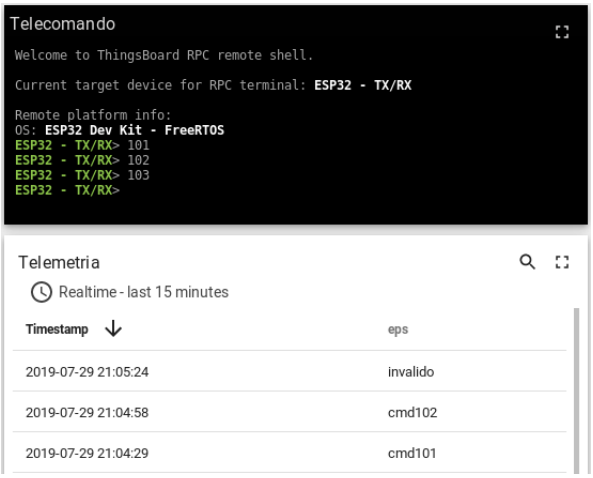

Figura 6. Teste de telemetria e telecomando com Arduino.

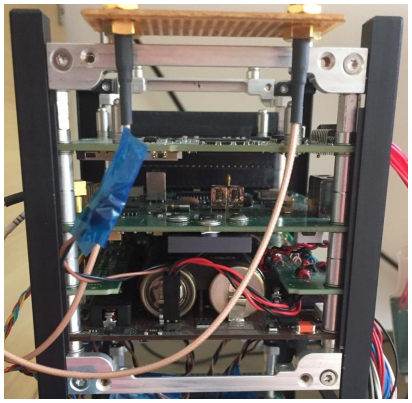

Figura 7. Modelo de desenvolvimento.

fornecer um sistema de simulação em *hardware* robusto e aplicável ao projeto CONASAT e às pesquisas desenvolvidas por alunos e professores da UFRN. Alguns subsistemas reais são complexos e a simulação em *hardware* requer estudos detalhados e tempo de desenvolvimento.

AGRADECIMENTOS

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

REFERÊNCIAS

[1] O. Koudelka, "Micro/nano/picosatellite-activities: Challenges towards space education and utilisation," in *Small Satellites: Regulatory Challenges and Chances*, ser. Studies in Space Law, I. Marboe, Ed. Brill, 2016.

[2] N. Palkovitz, "Small satellites: Innovative activities, tranditional laws, and the industry perspective," in *Small Satellites: Regulatory Challenges and Chances*, ser. Studies in Space Law, I. Marboe, Ed. Brill, 2016.

[3] H. J. Kramer and A. P. Cracknell, "An overview of small satellites in remote sensing," *International journal of remote Sensing*, vol. 29, no. 15, pp. 4285–4337, 2008.

[4] A. Mehrparvar, D. Pignatelli, J. Carnahan, R. Munakata, W. Lan, A. Toorian, A. Hutputanasin, and S. Lee, "CubeSat Design Specification rev.13," The CubeSat Program, Cal Poly, SLO, Tech. Rep., 2015.

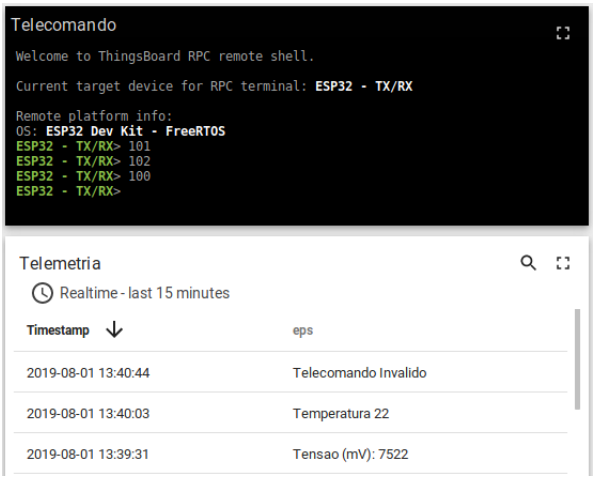

Figura 8. Teste de telemetria e telecomando com OBC.

[5] K. Ashton, "That 'internet of things' thing," *RFID Journal*, Jun. 2009. [Online]. Available: https://www.rfidjournal.com/articles/pdf?4986

[6] S. Li, L. Da Xu, and S. Zhao, "The internet of things: a survey," *Information Systems Frontiers*, vol. 17, no. 2, pp. 243–259, 2015.

[7] F. Mattern and C. Floerkemeier, "From the internet of computers to the internet of things," in *From active data management to event-based systems and more*. Springer, 2010, pp. 242–259.

[8] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[9] I. F. Akyildiz and A. Kak, "The internet of space things/cubesats: A ubiquitous cyber-physical system for the connected world," *Computer Networks*, vol. 150, pp. 134–149, 2019.

[10] A. Kak, E. Guven, U. E. Ergin, and I. F. Akyildiz, "Performance evaluation of sdn-based internet of space things," in *2018 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2018, pp. 1–6.

[11] B. Zufelt and J. Aarested, "Cloudsat: Centralized-adaptable ground station architecture utilizing an iot design methodology," in *2018 World Engineering Education Forum - Global Engineering Deans Council (WEEF-GEDC)*, Nov 2018, pp. 1–6.

[12] K. I. P. M. Queiroz, S. M. Dias, J. M. L. Duarte, and M. J. M. de Carvalho, "Uma solução para o sistema brasileiro de coleta de dados ambientais baseada em nanossatélites," *HOLOS*, vol. 7, pp. 132–142, 2018.

[13] M. N. Sweeting, "Modern small satellites-changing the economics of space," *Proceedings of the IEEE*, vol. 106, no. 3, pp. 343–361, Mar. 2018.

[14] S. Madry, P. Martinez, and R. Laufer, *Innovative Design, Manufacturing and Testing of Small Satellites*. Springer, 2018.

[15] NASA CubeSat Launch Initiative, *CubeSat 101: Basic Concepts and Processes for First-Time CubeSat Developers*. NASA, 2017.

[16] M. Swartwout. (2019, Jul.) CubeSat Database. [Online]. Available: https://sites.google.com/a/slu.edu/swartwout/home/cubesat-database

[17] CENTRO DE GESTÃO E ESTUDOS ESTRATÉGICOS - CGEE, "CubeSats," CGEE, Brasília, Tech. Rep., 2018.

[18] M. Spencer, "ETP CubeSat Simulator (Part 1, the techinical part)," *The AMSAT Journal*, pp. 4–8, Sept./Oct. 2009.

[19] ——, "ETP CubeSat Simulator (Part 2, theClassroom Activity Part)," *The AMSAT Journal*, pp. 4–8, Nov./Dec. 2009.

[20] J. J. Sellers, C. A. Bishop, J. R. Gossner, J. J. White, and J. B. Clark, "Eyassat: A revolution in teaching and learning space systems engineering," 2005.

[21] A. Johnston and P. Kilroy. (2019, Jul.) The AMSAT CubeSat Simulator: A New Tool for Education and Outreach. [Online]. Available: https://countingfromzero.net/amsat/CubeSatSimPaper.pdf

[22] A. A. Ismail, H. S. Hamza, and A. M. Kotb, "Performance evaluation of open source iot platforms," in *2018 IEEE Global Conference on Internet of Things (GCIoT)*, Dec 2018, pp. 1–5.

[23] J. Mineraud, O. Mazhelis, X. Su, and S. Tarkoma, "A gap analysis of internet-of-things platforms," *Computer Communications*, vol. 89-90, pp. 5–16, 2016.

[24] M. Yuan. (2017) Getting to know MQTT. [Online]. Available: https://developer.ibm.com/articles/iot-mqtt-why-good-for-iot/

[25] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[26] ThingsBoard. (2019, Jul.) ThingsboardBoard open-source iot platform. [Online]. Available: https://thingsboard.io/

[27] P. Fremantle. (2015) A reference architecture for the internet of things. [Online]. Available: https://wso2.com/whitepapers/a-reference-architecture-for-the-internet-of-things/

[28] R. Barry, *Mastering the FreeRTOS Real Time Kernel: A Hands-On Tutorial Guide*. Real Time Engineers Ltd, 2016.

[29] FreeRTOS. (2019) Tasks and co-routines. [Online]. Available: https://www.freertos.org/taskandcr.html

[30] Espressif Inc. (2019, Jul.) ESP32 Series Datasheet. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

[31] Espressif Systems. (2019, Jul.) ESP-IDF Programming Guide. [Online]. Available: https://docs.espressif.com/projects/esp-idf/en/latest/