Sistema De Controle Em Tempo Real Reconfigurável De Experimento Em Astrobiologia

Rafael Corsi Ferrão, Tiago Sanches Da Silva, Sergio Ribeiro Augusto, Vanderlei Cunha Parro[a]
Instituto Mauá de Tecnologia - Núcleo de Sistemas Eletrônicos Embarcados
São Caetano do Sul - SP - Brazil

***Resumo*—Este trabalho apresenta uma arquitetura de um sistema em tempo real e reconfigurável para controle de experimentos em aplicações aeroespacial. A arquitetura de software proposta tem como base dois tipos de serviços: o PBS (*Primary Boot Service*), o qual é responsável pela inicialização do sistema bem como sua reconfiguração e atualização remota, e o APS (*Application Service*), responsável por controlar as diversas partes dos experimentos e também pela comunicação com a plataforma. Ao final do artigo é apresentando uma topologia baseada no processador Leon e no protocolo espacial SpaceWire desenvolvida para a implementação da arquitetura proposta.**

***Palavras-chave*—RTOS, Leon, Sistema Reconfigurável, SpaceWire,**

I. Introdução

COM o aumento da complexidade de experimentos embarcados em aplicações aerospaciais surge a demanda do desenvolvimento de uma arquitetura de software capaz de suprir as necessidades de um experimento científico de tempo real, e que possa ser remotamente atualizado em partes ou total. Essa necessidade é baseada no fato de que correções ou melhorias no software pós lançamento pode corrigir falhas que o experimento possa sofrer durante seu tempo de operação podendo assim aumentando a vida útil do experimento e/ou a qualidade dos resultados.

A presente plataforma foi inspirada do ponto de vista de protocolos e sub sistemas no satélite CoRoT[1] [1], com enfoque no experimento de Astrobiologia proposto para ser embarcado na primeira sonda de espaço profundo brasileira (Aster).

Esta proposta implica em definir claramente as interfaces de comunicação e o sistema operacional da carga útil. Sistemas para aplicações aerospaciais são geralmente implementado com funcionalidade bem definida visando aumentar a robustez da aplicação. Entende-se por aumento de robustez a capacidade do software de não causar situações que possam interromper, parcialmente ou totalmente, o funcionamento do experimento quando em modo de operação.

O sistema proposto é baseado em quatro serviços: o PBS (*Primary Boot Service*), responsável pelo sistema de boot após reinicio físico, permite uma manipulação mínima do sistema de processamento e inclui o kernel das funções elementares do software bem como a verificação da integridade do código no APS e suas atualizações. E o APS (*Application Service*) serviço relativo a aplicação,ou seja, o gerenciamento do experimento propriamente dito. O mesmo possibilita telemetria. Este serviço pode ser atualizado e reconfigurado, essa tarefa é executada pelo PBS. O Monitor de comunicação externa é responsável pelo gerenciamento da comunicação e o Sistema de controle de experimentos que gerência em baixo nível os experimentos.

Foi escolhido como meio de comunicação entre experimento e plataforma o protocolo SpaceWire (SpW) [2], amplamente utilizado em missões espaciais possui grande aderência internacional pelo fato de possuir alta taxa de transmissão atrelado a um baixo consumo energético.

O processador Leon 3 [3] foi escolhido como plataforma de desenvolvimento do software proposto. É um processador amplamente utilizado em missões espaciais [4], vem sendo estudado pelo INPE como sucessor dos processadores atualmente em uso.

II. Arquitetura do Software

A proposta visa a criação de uma arquitetura capaz de ser utilizada em diferentes tipos de experimento. O software foi estruturado com base em dois serviços básicos (Fig 3), onde cada serviço possui três estados de operação, sendo apenas um estado de cada serviço é executado simultaneamente. O sistema utiliza como base de implementação um sistema operacional de tempo real (RTOS) como o RTEMS [5], utilizando suas funcionalidades de multi-tarefas, comunicação entre-tarefas e escalonamento entre tarefas.

Definiu-se quatro

A. Monitor de comunicação externa

O monitor de comunicação externa é uma tarefa responsável pelo controle, codificação/decodificação dos telecomandos (TC) que são enviados e recebidos pelo experimento. Como forma de comunicação, foi escolhido o protocolo SpaceWire, porém outras alternativas tais como o protocolo MIL-1553 [6] podem ser emprega.

As alterações de estados ocorrem por telecomando enviados da plataforma ao experimento, os TC são interpretados pelo módulo de comunicação que implementa um protocolo baseado em comando/resposta (ack) . Os comandos interpretados pelo módulo de comunicação são colocados em uma fila no experimento . Após a realização da operação, sendo bem sucedida ou não, o experimento envia uma resposta a respeito da operação, o que chamamos neste trabalho de

[a]Autor para correspondência: Vanderlei Cunha Parro, E-mail: vparro@mac.com .

[1]*COnvection ROtation et Transits planétaires*

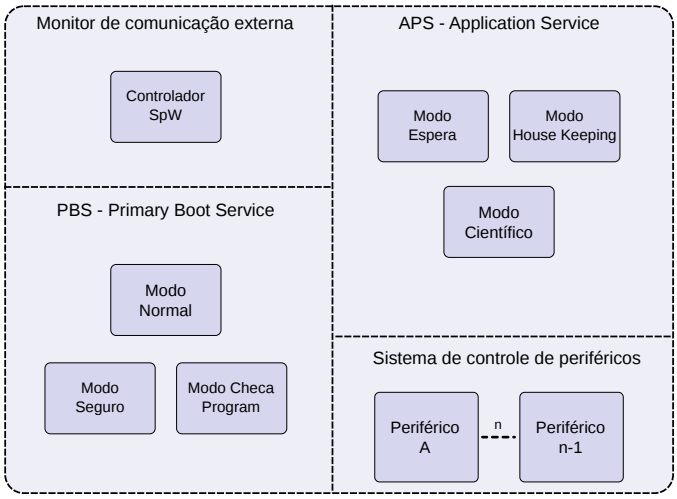

Figura 1. Diagrama de Harel do sistema proposto

	Conteúdo
Cabeçalho	Identificador
	Tipo
	Tamanho
Comando	Id
	Serviço
Parâmetro	Parâmetro 1
	Parâmetro 2
	Parâmetro 3

Tabela I
ESTRUTURA DO DATAGRAMA PROPOSTO

OReply (Operational Reply). O diagrama do modelo do OReply é mais simples que o de envio de comando, pois não existe a necessidade de reenvios e respostas, o OReply é encarado apenas como informativo, ou seja, não é um tipo de comunicação critica.

O datagrama desenvolvido para comunicação é subdividido em três partes, cabeçalho, comando e parâmetros, totalizando 8 bytes, como demonstrado na Tab. I. O Cabeçalho, composto por três partes: possui o *Identificador* do comando que é um valor único que identifica o datagrama; *Tipo* o qual define se o datagrama é do tipo ack ou OReply; *Tamanho* indicando a quantidade de dados que a mensagem possui. A parte do datagrama que diz respeito ao *Comando* identifica o comando a ser executado pelo sistema. Nos parâmetros: *Serviço* indica para qual tarefa o comando está sendo direcionado e os *Parâmetros* são utilizado para complementar alguns comandos, como por exemplo, o comando inicializar, que que envia como parâmetro a versão do firmware a ser inicializado.

B. PBS

O PBS é o controlador geral do experimento, nele estão o núcleo do sistema operacional e todas as tarefas críticas, não sendo reconfigurável. Os possíveis estados de operação são :

- **Modo Normal** - Executa uma versão do APS na inicialização do sistema e recebe comandos externos da comunicação, sendo que estes comandos servem para mudar o PBS ou o APS de estado;
- **Modo Seguro** - O PBS entra neste modo para atualizar o código do APS ou parte dele. Este modo não pode ser acionado no meio de uma operação de risco do APS, ou seja, quando APS estiver interagindo diretamente com os sensores e atuadores;
- **Modo Checa Programa** - Modo onde é verificado se o código gravado na EEPROM (APS) está íntegro e sem alterações.

Na inicialização do sistema, o PBS é colocado em Modo Checa Programa, onde verifica se a última versão do software armazenada na memória está integro. Verificada essa informação passa-se para o Modo Normal, onde inicializa-se o APS , possibilitando que comandos sejam transmitidos para o APS e que atualizações possam ser enviadas ao PBS.

C. APS

O APS é o serviço relativo ao gerenciamento do experimento propriamente dito. O mesmo possibilita telemetria de aquisição de dados dos sensores e varredura do instrumento para verificação de sua integridade. Este serviço pode ser atualizado e reconfigurado pela plataforma, possui os seguintes modos de operação:

- **Modo de Espera** - Este modo monitora a comunicação entre tarefas de maneira a verificar a existência de comandos enviados da plataforma para o APS;
- **Modo Housekeeping** - Este modo é responsável pela varredura dos sensores e atuadores do experimento, verificando se estão comunicáveis e íntegros;
- **Modo Científico** - Responsável pela operação do experimento. Também realiza o empacotamento das informações geradas pela telemetria para que seja enviada para a plataforma ou armazenada em memória.

D. Sistema de controle de periféricos

O sistema de controle de periféricos é um conjunto de tarefas, responsáveis pelo controle dos diversos periféricos do experimento. Esses periféricos podem ser atuadores, sensores ou memórias. Cada periférico é controlado por uma ou mais tarefas, flexibilizando a arquitetura. Os modos de operação de cada periférico são controlados individualmente pelo modo de operação do APS.

E. Alocação de memória

A alocação de memória proposta para o sistema, possui arquitetura ilustrada na Fig. 2, onde são utilizados uma memória somente de leitura (MRAM), e outra volátil (SRAM). Os serviços críticos do sistema, formados pelo PBS e pelo Monitor de comunicação externa são armazenados. Já as partes reconfiguráveis e as varáveis de programa são armazenadas na memória SRAM.

III. BANCADA DE IMPLEMENTAÇÃO

A bancada desenvolvida para a implementação do sistema proposto é constituída de um processador Leon 3 com frequência de operação de 60 MHz embarcado em uma FPGA Virtex 5 com dois cores SpaceWire funcionando a 200

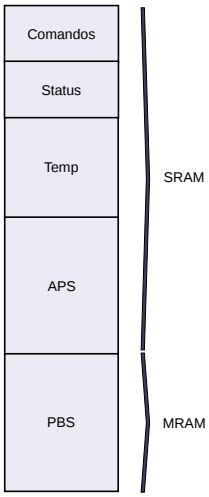

Figura 2. Alocação de memória proposta

Mbps cada, pretende-se com o sistema embarcado simular o experimento. O sistema embarcado possui comunicação com o computador via os dois canais SpW, simulando a interface de comunicação experimento/plataforma, para a ponte SpW-Computador, foi usando um hardware da Star-Dundee desenvolvido para esse fim. A fim de validar a bancada e possibilitar um estudo do desempenho tanto do protocolo SpW quanto do Leon, uma aplicação que simula o funcionamento de um CCD conectado a uma unidade de processamento.

A. *SpaceWire*

Um grande esforço tem sido realizado por várias pessoas ligadas à Agência Espacial Européia, à indústria espacial europeia e a área acadêmica no sentido de realizar um aperfeiçoamento e uma padronização das transferências de dados em sistemas embarcados específicos para a área espacial. Um desses esforços tem sido a criação do protocolo Spacewire "ESA ECSS-E-50-12A" de 24 de Janeiro de 2003 e sua atualização em 31 de Julho de 2008 ("ESA ECSS-E-50-12C").

Um dos objetivos da utilização deste protocolo é aumentar a compatibilidade entre subsistemas e componentes aeroespaciais a fim de possibilitar uma maior reutilização dos mesmos em missões diferentes, como, por exemplo, o reuso de unidades de processamento, unidades de armazenamento de memória, unidades de telemetria, sensores óticos, previamente testados e validados. Esse reuso possibilita além de uma redução de custos como também um menor prazo de desenvolvimento.

O protocolo SpaceWire leva em consideração outros dois padrões, a saber, o IEEE 1355-1995 (IEEE Standard for Heterogeneous InterConnect) e ANSI/TIA/EIA-644 (LVDS). O protocolo SpaceWire provê um link full-duplex, ponto a ponto e serial com taxa máxima de 10Mbps e taxa máxima de 400 Mbps ponto a ponto a uma distancia de no máximo 10 metros.

Utilizou-se o Codec SpaceWire Light disponível em ,

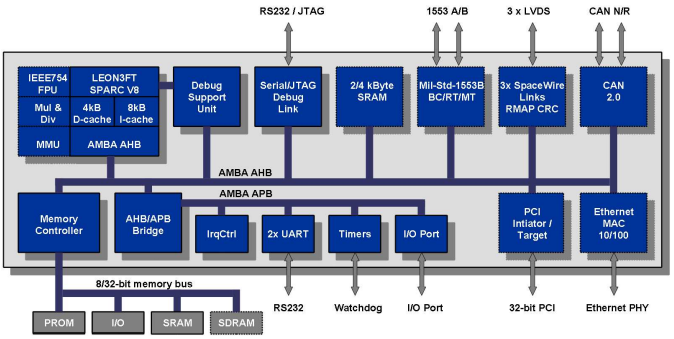

Figura 3. Biblioteca GRLIB

B. *Leon 3*

O LEON3 é um processador de 32-bits baseado na arquitetura SPARC V8, desenvolvido para aplicações embarcadas, combinando alta performance com baixa complexidade e baixo consumo energético. Distribuído através do VHDL GRLIB IP [7] sobe licença GPL (*General Public License*) contém além do processador, módulos como: controlador de memória, interface PCI, USART, Ethernet MAC. Um exemplo de aplicação das bibliotecas presentes no GRLIB pode ser visto na Fig 3.

O processador já é utilizado em diversas missões [], possui uma versão chamada de Leon3-FT que contém arquitetura enriquecida para detectar e corrigir SEU (*Single Event Upset*) , correção de erros no cache e arquitetura com tripa redundância.

O processador LEON3 implementa uma arquitetura Harvard com instruções e *cache* de dados separados. Possui divisão e multiplicação em hardware e uma unidade de processamento de ponto flutuante pode ser adicionada. Contém também uma unidade de gerenciamento de memória que possibilita o mapeamento de 32-bits de endereço virtual e 32-bits de memória. Pode funcionar com múltiplos processadores, os quais podem compartilham uma mesma memória.

C. *Aplicação*

A aplicação proposta, utiliza a arquitetura da Fig. 4 baseada no satélite PLATO [8] onde a comunicação do subsistema de imageamento (N-FEE) de utilização científica é feito a partir de dois canais SpW funcionando a 100Mbps casa. O N-FEE é conectado uma unidade de processamento (DPU) responsável pela extração de dados científicos das imagens. Uma nova imagem é transmitida a cada 6s, sendo que a transmissão deve ocupar no máximo 50% do tempo total.

A proposta visa emular um sistema real de transmissão de dados entre o N-FEE e a DPU, para tanto desenvolveu-se uma aplicação que utiliza o conjunto computador/ USB-Brick, que possibilita a serialização e envio de imagens ao processador Leon, podendo assim validar algorítimos de extração de dados voltados para aplicações aerospaciais. Uma futura implementação visa utilizar tal bancada para a implementação da arquitetura do sistema em tempo real e reconfigurável para controle de experimentos.

Foi utilizado como sistema operacional o RTEMS (*Real-Time Executive for Multiprocessor Systems*) com duas tarefas distintas, uma voltada para recepção de dados (TRX) e outra

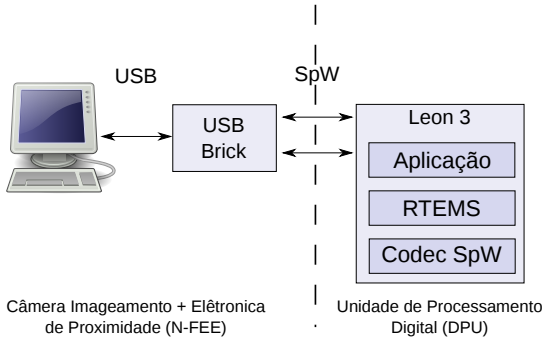

Figura 4. Bancada desenvolvida para implementação de aplicações aerospaciais

para transmissão (TTX) da imagem, a comunicação entre tarefas é feito por um *mail box* do TRX ao TTX contendo o endereço da imagem recebida (em memória local), e o tamanho armazenado. Com a proposta conseguiu-se uma taxa de 50Mbps de transmissão, contando latência do link de comunicação e acesso a memória.

IV. Conclusão

Nesse trabalho, definiu-se uma topologia robusta capaz de servir como base para o desenvolvimento de softwares para experimentos científicos que necessitem de uma abordagem de tempo real e que possam ser atualizados após lançamento.

Foi desenvolvido uma bancada para validação e desenvolvimento de aplicações com processador Leon e o protocolo SpaceWire, pretende-se com essa bancada implementar a proposta de sistema reconfigurável.

Referências

[1] Damien Cailliau and Remy Bellenger. The corot instruments software: towards intrinsically reconfigurable real-time embedded processing software in space-borne instruments. In *High-Assurance Systems Engineering, 1999. Proceedings. 4th IEEE International Symposium on*, pages 75–80. IEEE, 1999.

[2] ESA Requirements and Standards Division. SpaceWire Links, nodes, routers and networks - ECSS-E-ST-50-12C. *Space engineering*, 2008.

[3] Christopher P. Enhanced LEON3 Core for Superscalar Processing. 1, 2007.

[4] Hamed Abbasitabar, Hamid R Zarandi, and Ronak Salamat. Susceptibility analysis of leon3 embedded processor against multiple event transients and upsets. In *Computational Science and Engineering (CSE), 2012 IEEE 15th International Conference on*, pages 548–553. IEEE, 2012.

[5] Bingxue Zhang, Xudan Xu, and Bo Li. Research on the design of software fault tolerance based on rtems. In *Computer, Mechatronics, Control and Electronic Engineering (CMCE), 2010 International Conference on*, volume 1, pages 402–405. IEEE, 2010.

[6] Alta Data Technologies LLC. MIL STD 1553 Tutorial and Reference. 2, 2007.

[7] Gaisler. GRLIB IP Core Users Manual. 1, 2010.

[8] C. Catala and PLATO consortium. PLATO PLAnetary Transits and Oscillations of stars - A study of exoplanetary systems . *proposal*, 1, 2008.