# Um sistema computacional reconfigurável *embarcado* num veleiro autónomo*

José C. Alves
*FEUP - DEEC*
*jca@fe.up.pt*

Nuno A. Cruz
*FEUP - DEEC*
*nacruz@fe.up.pt*

## Resumo

*Este artigo apresenta um sistema computacional implementado em FPGA para controlo de um veleiro autónomo não tripulado. Este veleiro foi desenvolvido na FEUP para integrar a competição Microtransat que tem como objectivo promover o desenvolvimento de embarcações à vela não tripuladas capazes de oferecer grande autonomia de operação, com elevado potencial para realizar missões autónomas de longa duração em alto mar. O sistema computacional foi desenvolvido em torno de um computador embutido correndo uCLinux num processador MicroBlaze implementado numa FPGA Spartan3E, acrescido de vários sistemas dedicados para interface, processamento e controlo. A adopção de um sistema operativo baseado em Linux permitiu dispor de variados serviços (sistema de ficheiros, multi-processamento, TCP/IP) convenientes para o processo de desenvolvimento das componentes de hardware e de software. A flexibilidade da arquitectura do sistema hardware/software desenvolvido facilita a migração de tarefas entre software e hardware dedicado com o objectivo de se avaliar o impacto no consumo global de energia, para diferentes configurações dos processos de controlo envolvidos.*

## 1. Introdução

O desenvolvimento de uma embarcação à vela autónoma tendo em vista a sua operação não assistida durante longos períodos de tempo levanta vários desafios tecnológicos em áreas muito diversas, que vão desde a construção mecânica da embarcação até à implementação dos sistemas electrónicos que realizam o processo de navegação e controlo. A navegação à vela pode ser vista como um problema de optimização complexo que depende de várias variáveis fortemente interrelacionadas entre si: o rumo ou o destino desejados, o vento (direcção e velocidade), a inclinação e oscilação da embarcação ou a ondulação.

Uma embarcação à vela não pode avançar directamente contra o vento. Embora o ângulo mínimo entre a direcção de navegação (*rumo*) e a direcção do vento seja muito dependente do tipo de embarcação à vela, um valor nominal de $\pm 45$ graus é geralmente tomado como o limite prático para a generalidade das embarcações convencionais (figura 1). A direcção do vento *aparente* visto pela embarcação $\vec{v_{aw}}$ é o resultado da diferença vectorial entre a

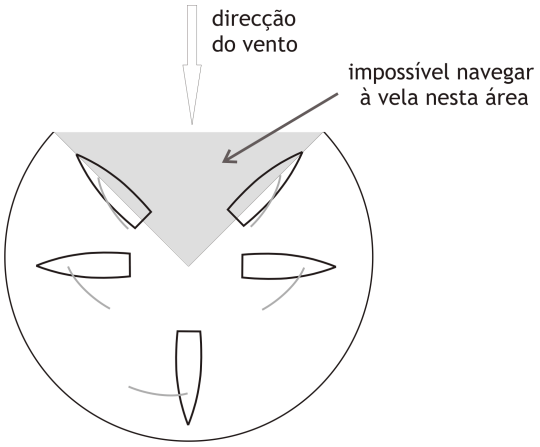


Figura 1. A navegação à vela é impossível para um ângulo com o vento inferior a 45 graus.

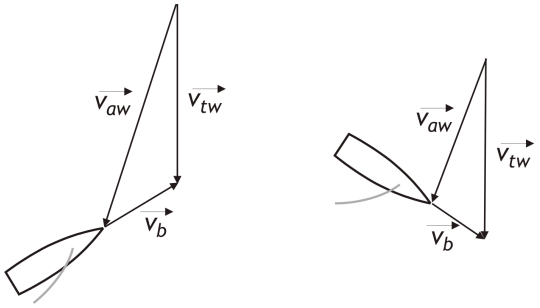


Figura 2. Relação entre o vento aparente visto pela embarcação ($\vec{v_{aw}}$), a velocidade da embarcação ($\vec{v_b}$) e o vento real ($\vec{v_{tw}}$).

velocidade do vento real $\vec{v_{tw}}$ com a velocidade da embarcação $\vec{v_b}$ (figura 2). A consequência disso é que variações da velocidade da embarcação conduzem à alteração da direcção do vento *aparente* $\vec{v_{aw}}$, o que exige ajustes constantes de rumo ou da posição das velas de forma a manter a máxima velocidade. Os efeitos da ondulação e das correntes, particularmente importantes em embarcações de pequena dimensão, contribuem com dificuldades adicionais.

Actualmente os regulamentos internacionais que regem a navegação de embarcações em águas internacionais não contemplam veículos não tripulados porque, em termos práticos, não existem ainda soluções que permitam a embarcações autónomas cumprir de forma robusta o "código da estrada" dos mares, adoptado internacionalmente. Com

*Este projecto é financiado pelo Departamento de Engenharia Electrotécnica da Faculdade de Engenharia da Universidade do Porto

o potencial exibido por veleiros autónomos para realizar missões reais, aliado à disseminação de equipamentos automáticos de identificação de baixo custo, poderá ajudar a mudar esse cenário num futuro próximo. A interpretação automática dessas regras é objecto de trabalhos recentes que recorrem a técnicas de inteligência artificial e processos de optimização multiobjectivo [1, 2], embora orientados para uma perspectiva de apoio à decisão de tripulações humanas.

Vários autores têm abordado os diversos problemas associados à navegação autónoma de embarcações à vela. O problema básico do controlo de rumo (por actuação no leme) está desde há várias décadas parcialmente resolvido e é correntemente usado em embarcações de recreio e comerciais. No entanto, o desempenho de uma embarcação à vela é fortemente dependente da maneira como o leme é controlado para aproveitar os declives das ondas e as mínimas variações do vento. O recurso ao piloto automático na vela de competição é mesmo tido como um *mal necessário*, sendo assim um problema onde existe ainda espaço para novos desenvolvimentos. Esta matéria tem sido objecto de vários trabalhos onde foram aplicadas técnicas baseadas em redes neuronais [3], conjuntos difusos [4, 5] ou recorrendo a outros processos de inteligência artificial [6].

O processo de navegação completamente autónoma de um veleiro envolve outras tarefas, para além do simples controlo de rumo. Problemas como atingir um conjunto de pontos geográficos, manter-se numa área delimitada ou minimizar o tempo necessário para cumprir um percurso dado têm sido objecto de variados trabalhos [7, 8, 9, 10, 11]. Destes esforços resultaram já várias embarcações à vela autónomas como veículos demonstradores das tecnologias desenvolvidas e que têm vindo a aderir às competições organizadas no âmbito do desafio Microtransat.

O veleiro autónomo FASt (*FEUP Autonomous Sailboat*[1]—figura 3) é uma embarcação à vela não tripulada com 2.5 m de comprimento que foi concebida e desenvolvida na FEUP no âmbito de um projecto extra-curricular com alunos do Mestrado Integrado em Engenharia Electrotécnica e de Computadores. A construção do casco foi parcialmente feita num fabricante de kayaks de competição (Élio Kayaks[2]) empregando a mesma tecnologia e materiais usados para o fabrico dos kayaks. A configuração do conjunto mastro e velas é semelhante à usada em veleiros de dimensão real.

O sistema electrónico de navegação e controlo foi construído sobre uma plataforma reconfigurável contendo uma FPGA que executa o sistema operativo uCLinux num processador Microblaze. Para além do CPU e dos periféricos necessários ao sistema computacional, a FPGA implementa um conjunto de unidades de interface e processamento dedicadas que realizam variados processos de comunicação com os periféricos (sensores e actuadores). Estes módulos disponibilizam para o CPU os dados relevantes e pré-processados recolhidos de cada sensor e geram os sinais de controlo necessários ao comando dos actuadores (motores eléctricos).

[1] `www.fe.up.pt/fast`
[2] `www.elio-kayaks.com`


Figura 3. O veleiro autónomo FASt.

Este artigo apresenta o sistema computacional que foi desenvolvido para o veleiro FASt. Na secção 2 apresenta-se a organização do sistema electrónico do veleiro e do sistema computacional que foi integrado no dispositivo FPGA e na secção 3 apresentam-se algumas métricas obtidas da implementação. A organização da componente de software é descrita na secção 4 e na secção 5 conclui-se o artigo e apresentam-se os planos para desenvolvimentos futuros.

## 2. Hardware

O sistema electrónico do FASt foi construído em redor de uma plataforma comercial baseada em FPGA, que implementa o seu sistema computacional. Para além dos vários sensores e actuadores que se descrevem na secção 2.2,

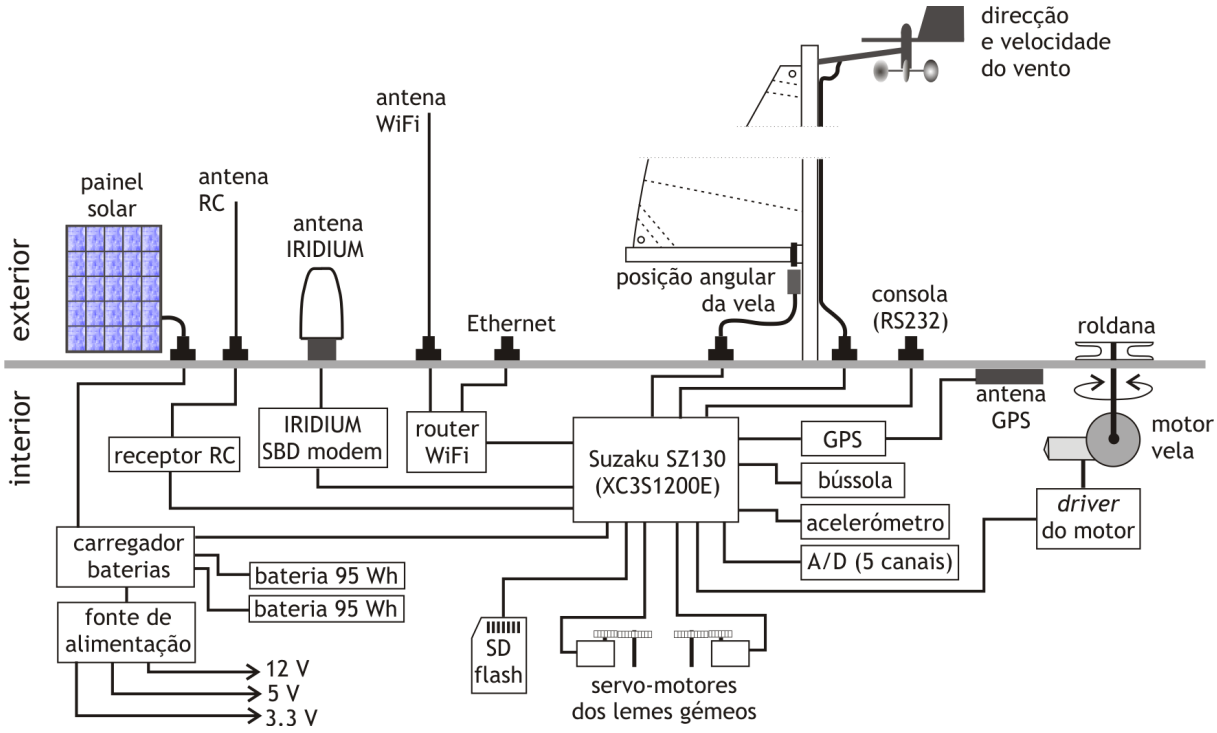


Figura 4. O sistema electrónico do FASt. Todas as ligações eléctricas com o exterior são realizadas através de conectores estanques.

o veleiro tem um *router* Ethernet com ligação sem fios, receptor de rádio comando usado para operação manual e um modem IRIDIUM para envio de mensagens curtas de dados, quando em navegação oceânica longe da costa. A energia eléctrica necessária a bordo é fornecida por um painel solar de 45 Wp (Watt de pico), duas baterias de iões de lítio com capacidade total de 190 Wh e um módulo comercial que integra o carregador das baterias e a fonte de alimentação.

A figura 4 mostra um diagrama simplificado do sistema electrónico incluído no veleiro. Uma placa-mãe agrega todos os componentes electrónicos que são usados para interface com a FPGA e disponibiliza um conjunto de conectores periféricos onde ligam todos os sensores e actuadores do FASt. É também incluído um leitor de cartão de memória *flash* (formato SD) que é usado para registo de dados e uma memória não volátil com relógio de tempo real que é usada para manter as variáveis de estado principais e para fornecer a hora real quando o GPS for colocado em modo de baixo consumo.

## 2.1. Computador reconfigurável

O sistema computacional foi implementado numa plataforma comercial construída em torno de uma FPGA Spartan3E 1200 (Suzaku SZ130 [12]). O pacote distribuído com este sistema inclui um projecto EDK com um processador Microblaze e uma distribuição de uCLinux pré-carregada na sua memória *flash*, o que se revelou ser um bom ponto de partida para a construção de um sistema embutido em FPGA tendo por base aquele sistema operativo.

Esta placa tem 32 MBytes de SDRAM, 8 MBytes de memória *flash*, ligação de rede Ethernet implementada por um circuito integrado dedicado e porta série para a consola. A imagem do sistema operativo e o arquivo de configuração da FPGA são mantidos na memória *flash* e podem ser actualizados através de comandos invocados a partir do interpretador de comandos do sistema operativo. Após a configuração da FPGA é executada uma pequena aplicação residente nas memórias BRAM, que constrói o sistema de ficheiros num disco virtual criado em RAM e inicia o sistema operativo. A placa dispõe de um total de 86 terminais ligados directamente a pinos da FPGA o que possibilita a ligação directa de vários periféricos.

## 2.2. Sensores e actuadores

Os sensores e actuadores são ligados a terminais da FPGA através de circuitos para isolamento ou adaptação de níveis eléctricos. Os sensores principais são um receptor de GPS, bússola electrónica, inclinómetro, sensor de direcção e velocidade do vento e sensor de ângulo da vela. Existem ainda disponíveis 5 canais de entrada de conversores A/D de 16 bits e várias entradas/saídas digitais, que podem ser usados para a ligação de sensores adicionais que venham a ser requeridos para a realização de missões.

Os actuadores permitem controlar os dois lemes gémeos e a posição das duas velas. O comando dos lemes é feito por 2 servo-motores independentes (um para cada leme). O comando da posição das duas velas é feito por um único motor DC que é actuado através de um modulador PWM. A placa-mãe suporta até mais 3 motores que podem vir a

ser usados para o controlo individual das duas velas ou para recolher parcialmente as velas.

## 2.3. Organização do sistema computacional

O sistema computacional construído na FPGA foi desenvolvido sobre o projecto de referência fornecido com a placa Suzaku. Foi acrescentado um bloco de interface com o barramento local do processador MicroBlaze (bloco GPIO) que interliga o processador com os vários blocos que foram criados para implementar as interfaces dedicadas com cada periférico. A figura 5 mostra a organização do sistema digital implementado na FPGA

## 2.4. Interfaces com os sensores

Para além dos protocolos de comunicação específicos de cada sensor, os blocos periféricos implementam funções adicionais, movendo para hardware toda a funcionalidade associada aos processos de recolha e pré-processamento de dados, entregando ao processador principal (Microblaze) essa informação num formato adequado à sua utilização pelas aplicações em software. Por um lado, o processador é libertado de um conjunto significativo de tarefas de baixo nível, ficando a seu cargo apenas as tarefas de navegação e controlo que são executadas a ritmos muito baixos. Presentemente, o processo que controla o leme é executado ao ritmo mais elevado de toda a aplicação e é de apenas 10 Hz; o processo mais lento—controlo da navegação—é executado a apenas 1 Hz. Por outro lado, esta abordagem também facilita a migração para hardware de processos que actualmente são realizados em software, já que os dados necessários estão disponíveis em barramentos internos do sistema implementado na FPGA.

O sensor de direcção de vento foi construído com base num cata-vento mecânico que faz girar um íman sobre um sensor magnético de posição angular (AS5040). A interface digital com este sensor consiste numa interface série síncrona específica para este sensor e calcula também o valor médio sobre uma janela deslizante de 64 amostras que são lidas à cadência de 50 Hz. O processo de cálculo da média é repetido para cada nova amostra recebida, não sendo suficiente calcular apenas a média aritmética dos valores lidos devido à descontinuidade na passagem de $-180$ para $+180$ graus. O processo implementado calcula a média aritmética dos desvios em relação ao último valor calculado para a direcção do vento, adicionando aquela média a este valor. O processo de cálculo inclui um *buffer* circular com 64 palavras de 10 bits (dados lidos do sensor) que é implementado como um registo de deslocamento realizado com *LUTs*. A saída é o valor corrigido para graus como um inteiro de 9 bits com sinal, no intervalo $[-180, +180]$. A figura 6 mostra um diagrama de blocos deste circuito.

O sensor de posição angular da vela é realizado com o mesmo circuito integrado, medindo o ângulo de rotação da *retranca* (a verga horizontal que suporta a parte inferior da vela) em relação ao eixo longitudinal do veleiro. Este sensor apenas necessita de medir ângulos num intervalo não superior a 180 graus, o que é limitado fisicamente pelo deslocamento possível da retranca. Por esta razão, o cálculo da média dos valores lidos do sensor pode ser feito como uma média aritmética simples, embora na implementação actual seja usado um módulo de interface igual ao que é usado para o sensor de direcção de vento.

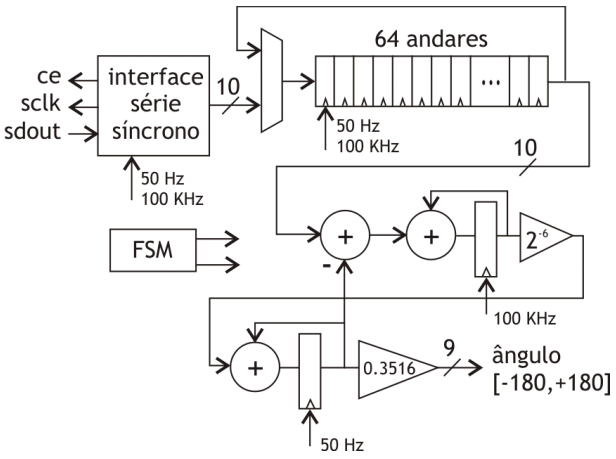


Figura 6. Diagrama de blocos do interface com o sensor de direcção de vento (CI AS5040).

A velocidade do vento é fundamental para o algoritmo de navegação à vela porque este valor é necessário para se conseguir estimar a direcção do vento real. A medida é feita com um anemómetro de copos que produz um impulso a cada rotação do rotor. A velocidade de rotação (ou velocidade do vento) é proporcional ao inverso desse tempo. O módulo de interface com este sensor amostra a velocidade à frequência de 10 Hz e filtra os valores lidos calculando uma média com uma janela deslizante de 64 amostras.

O GPS e a bússola electrónica produzem dados que são enviados periodicamente por uma porta série. A bússola é um módulo integrado da Honeywell que realiza a compensação do valor calculado para a orientação magnética com a sua inclinação lateral e longitudinal. A informação dos 3 ângulos calculados (*heading*, *pitch* e *roll*) é transmitida em formato legível como uma cadeia de caracteres ASCII que representam aqueles valores em decimal. Apesar deste formato permitir visualizar os valores transmitidos com um simples terminal ASCII, obriga à sua tradução para uma representação binária. O módulo que realiza a interface com a bússola traduz essa sequência de caracteres nos valores dos 3 ângulos, disponibilizando-os em binário à mesma cadência transmitida pela bússola (8 Hz). O GPS envia por uma porta série mensagens binárias de acordo com um formato proprietário do fabricante (protocolo UBX da ublox [13]). A informação relevante para o processo de navegação (latitude, longitude, velocidade, rumo, hora/data e estado) é extraída por 3 máquinas de estados distintas operando em paralelo, uma para cada um dos 3 tipos de mensagens que são interpretadas.

Outros sensores incluem detectores de água em pontos chave no interior do casco, temperatura interior e luz ambiente. Dois conversores A/D permitem monitorizar o valor das 3 tensão de alimentação usadas no sistema (3.3V, 5V e

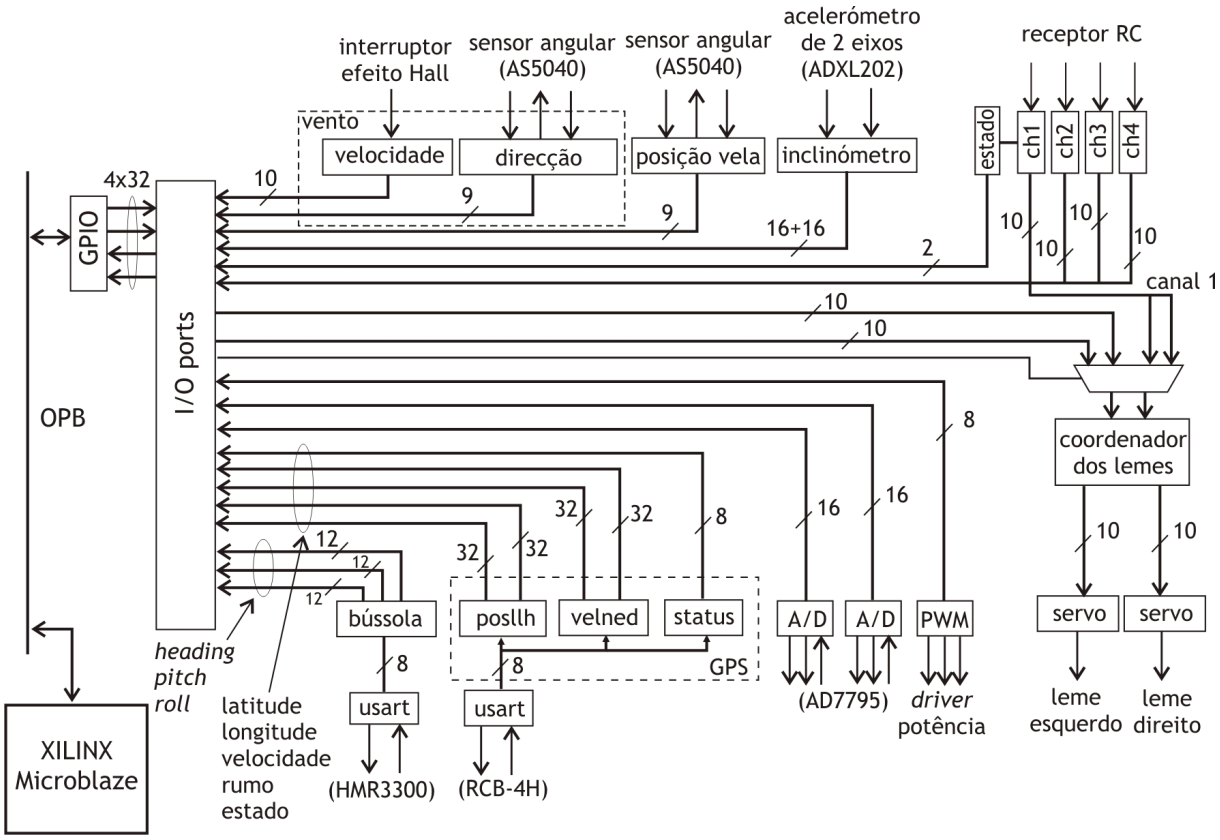


Figura 5. Representação simplificada do sistema desenvolvido para interface com os periféricos.

12V) e a tensão eléctrica presente na saída do painel solar, disponibilizando um total de 5 canais analógicos que podem ser usados para ligar sensores adicionais. Finalmente, o módulo carregador de baterias envia periodicamente informação sobre o estado de carga das baterias.

## 2.5. Interfaces com os actuadores

Os actuadores incluídos no veleiro (servo-motores) são igualmente comandados por módulos desenvolvidos à medida para esta aplicação. Os servo-motores que accionam os lemes são comandados por sinais digitais com frequência de 50 Hz e pulso positivo de duração entre 0.8 ms a 2.2 ms, proporcional à posição pretendida para o servo. A interface com o motor que define a posição das velas é realizada através de um modulador PWM implementado também por um módulo dedicado. Um filtro passa-baixo inserido na entrada do modulador PWM suaviza as transições bruscas nos valores aplicados na entrada, evitando dessa forma acelerações bruscas e consequentes picos da intensidade de corrente consumida. Presentemente o controlo de posição do motor que comanda a vela é feito por software, usando a informação de posição obtida de um potenciómetro engrenado mecânicamente com o eixo do motor. Futuramente este controlador será também implementado em hardware, usando também a informação de posição obtida do sensor de posição angular da vela.

## 2.6. Rádio-comando

O FASt inclui um receptor de rádio-comando de 4 canais proporcionais que, na presença do sinal transmitido, produz 4 sinais padrão para controlo de servo-motores que representam a posição (horizontal e vertical) das duas alavancas de controlo do transmissor. Os sinais gerados pelo receptor são traduzidos em 4 palavras de 10 bits disponibilizadas ao CPU. O sinal de saída de um dos canais (canal 1 - movimento horizontal da alavanca direita) pode ser encaminhado directamente para a entrada do módulo de controlo dos servo-motores dos lemes, permitindo o comando manual sem intervenção do software em execução no processador principal.

Para além de possibilitar um comando completamente manual do veleiro, o rádio comando é também usado para definir o modo de funcionamento do sistema de navegação e controlo. Na ausência do sinal de rádio, o sistema entra em modo autónomo e inicia a execução da missão que lhe foi previamente programada: percorrer em ordem um conjunto dado de pontos geográficos. Com o sinal de rádio-comando presente durante mais de 5 segundos o modo autónomo é abandonado e é activado o modo rádio-controlo. Neste modo, a alavanca direita é usada para o controlo manual e a alavanca esquerda é usada para seleccionar 3 modos de operação: completamente manual, rumo constante ou ângulo ao vento constante. A figura 7 ilustra os 3 modos de operação possíveis com o rádio comando e a função que

os 2 *joysticks* assumem em cada um dos modos.

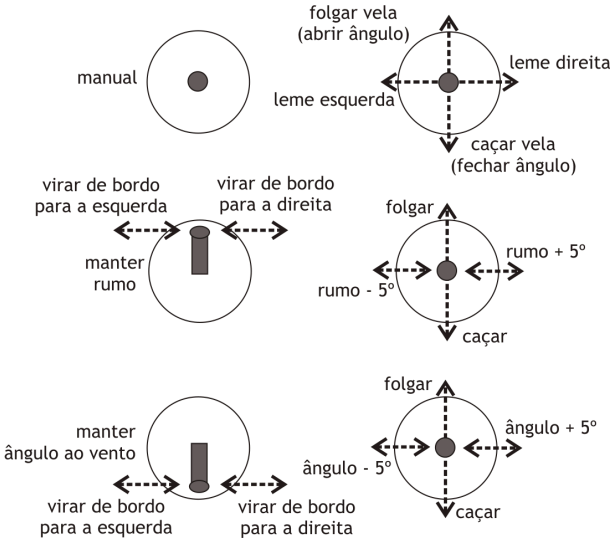


Figura 7. Função dos 2 *joysticks* do transmissor de rádio-comando nos 3 modos de funcionamento: manual, manter o rumo e manter o ângulo com o vento.

## 3. Implementação e resultados

O sistema descrito na secção 2 foi implementado na FPGA Spartan3E 1200 da placa Suzaku, juntamente com o sistema base incluído no projecto de referência. O sistema actual ocupa aproximadamente metade dos recursos da FPGA (tabela 1), existindo por isso espaço livre significativo para implementar em hardware algumas das funcionalidades actualmente realizadas em software. Apesar de todos os terminais disponíveis da FPGA estarem fisicamente ligados a conectores na placa-mãe, alguns desses conectores foram previstos para suportar futuras expansões e podem naturalmente ser usados para ligar qualquer outro dispositivo com interface digital.

Tabela 1. Resumo da ocupação da FPGA Spartan 3E 1200.

| | | |
|---|---|---|
| LUTs de 4 entradas | 7.314 | (42%) |
| Flip-flops | 3.997 | (23%) |
| Block RAMs | 14 | (50%) |
| Multiplicadores dedicados | 8 | (28%) |

### 3.1. Consumo de energia

O consumo de energia eléctrica a bordo é um dos factores mais críticos no desempenho global de um veleiro autónomo. Apesar de a propulsão ser feita apenas com recurso à navegação à vela, é necessária energia eléctrica para operar os sistemas electrónicos e os actuadores que substituem a força manual usada em embarcações reais. A forma mais adequada de produzir energia eléctrica a bordo, sem prejudicar o desempenho da navegação à vela, é transformando energia solar com painéis foto-voltaicos.

A opção pelo recurso a um sistema computacional baseado em FPGA resultou da extrema flexibilidade que permite, durante a fase de desenvolvimento, para a criação de sistemas de interface dedicados com um conjunto muito heterogéneo de dispositivos. Apesar de existirem no mercado vários sistemas computacionais orientados para aplicações embutidas e de mais baixo consumo do que a solução FPGA que foi adoptada, incluem naturalmente um conjunto de periféricos padrão para uso geral, o que obrigaria certamente a construir um sistema digital dedicado para concentrar as ligações aos periféricos, e a desenvolver em software os protocolos de comunicação necessários.

O sistema electrónico completo, incluindo a placa SZ130, o GPS, a bússola, o receptor de rádio-controlo e todos os circuitos de adaptação eléctrica incluídos na placa-mãe, consome uma potência global que não ultrapassa 1.7 W, com o Microblaze a operar a 50 MHz e o circuito de interface com a rede Ethernet activo. Desactivando a rede Ethernet (não é necessária em navegação de longo curso) o consumo de potência decresce aproximadamente 200 mW e reduzindo a frequência de relógio do Microblaze para metade consegue-se baixar mais 100 mW. Todos os circuitos de interface actuais são síncronos com o sinal global de relógio de 50 MHz, embora quase todos operem, na realidade, a ritmos muito mais baixos. Com a migração de funções computacionalmente críticas para hardware dedicado (por exemplo, o cálculo de funções trignométricas ou a verificação de intersecções de segmentos de recta) será possível reduzir ainda mais a frequência de relógio do Microblaze, embora mantendo a funcionalidade oferecida pelo sistema operativo.

O painel solar produz uma potência máxima de 45 W, que só é atingida com o sol incidente na perpendicular sobre o painel. Em condições reais, a energia total que será possível recuperar ao longo de um dia é muito dependente do estado do tempo e também da navegação que o veleiro fizer. Na realidade, qualquer sombra no painel solar provocada pelas velas ou mastro faz baixar muito o seu rendimento e, pelo menos com a configuração actual, é impossível evitar completamente as sombras em todas as situações. Até agora não foram ainda obtidos dados que permitam quantificar a energia recuperada em condições reais de navegação durante longos períodos.

A tabela 2 apresenta uma partição do consumo de energia pelos diversos componentes do sistema, assumindo um cenário pessimista para a actividade dos motores que comandam os lemes e a vela. Os lemes podem ser actuados a uma taxa máxima de 10 Hz, embora tenham sido conseguidos bons resultados com apenas 2 Hz. O motor que comanda as velas só tem que ser actuado para ajustar as velas, o que acontece apenas quando é modificado o rumo em relação ao vento. Durante a navegação em áreas alargadas a frequência de ajuste das velas poderá sempre muito baixa, podendo ir de vários minutos até mesmo algumas horas.

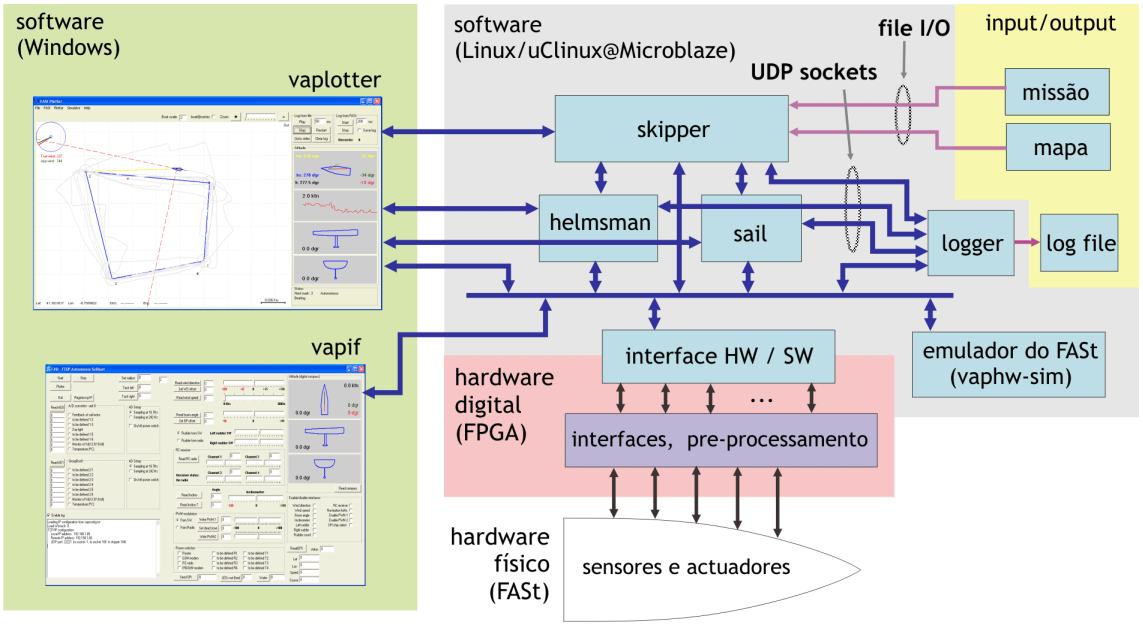


Figura 8. Organização do software do FASt.

Tabela 2. Consumos médios de energia eléctrica estimados para os vários componentes do FASt (a linha "Outros" agrega os consumos das luzes de navegação, comunicações e outros dispositivos incluídos na placa-mãe).

| Componente | tempo útil | potência média (mW) | % potencia total |
|---|---|---|---|
| SZ130 e sensores | 100% | 1700 | 51% |
| GPS | 100% | 170 | 5% |
| Servos (lemes) | 50 % | 1000 | 30% |
| Motor vela | 0.25% | 160 | 5% |
| Outros | | 300 | 9% |
| Total | | 3330 | |

## 4. Software

A componente de software foi desenvolvida em C sobre as bibliotecas padrão de Linux também disponíveis no uCLinux. Os processos de navegação e controlo são divididos em vários processos concorrentes comunicando entre si através de *sockets* UDP (figura 8). A adopção de um sistema operativo baseado em Linux e que suporta a maioria das suas funções padrão permite um ambiente de desenvolvimento de software muito flexível. Qualquer um dos programas que implementam as funções de navegação e controlo pode ser compilado e executado em qualquer máquina Linux que esteja acessível via TCP/IP ao computador do FASt. Esta é uma vantagem evidente resultante da opção por este sistema operativo face a outros que existem para sistemas embutidos, apesar de o uCLinux não ser orientado para aplicações de tempo real.

Para suportar o desenvolvimento e validação do software foi construído um emulador do FASt que, em ambiente laboratorial, substitui a camada que implementa a interface com o hardware real. O emulador (`vaphw-sim`) inclui um modelo dinâmico não linear do veleiro que simula o seu comportamento em função do vento e dos comandos recebidos para posicionar o leme e as velas. Um simulador de vento modela uma direcção e velocidade variáveis e o emulador responde para o resto do sistema os dados obtidos dos sensores que actualmente são usados no processo de navegação: posição geográfica, velocidade, orientação (*heading*), direcção e velocidade do vento aparente. Isto revelou-se essencial para permitir o desenvolvimento de software de forma independente do hardware e assim limitar o número de ensaios de campo necessários, o que exige uma logística, deslocação de recursos e tempo significativos.

Para facilitar a depuração do hardware desenvolvido e do software de interface foi criada uma aplicação visual interactiva que permite aceder directamente a todos os periféricos do FASt (`vapif`). Este programa comunica apenas com a camada de baixo nível (`vapd`) e permite ler qualquer sensor ou escrever em qualquer actuador. A monitorização em tempo real do veleiro enquanto a navegar (navegação real ou simulada), é feita através de outra aplicação (`vaplotter`) que regista graficamente a posição da embarcação, incluindo todos os parâmetros que são relevantes para os algoritmos de navegação e controlo. A mesma aplicação permite programar a missão a realizar pelo veleiro como uma sequência de pontos geográficos que devem ser atingidos, e também para reproduzir os dados gravados durante uma missão de navegação (real ou simulada).

## 5. Conclusões

Neste artigo apresentou-se um sistema computacional reconfigurável integrado num veleiro autónomo não tripulado. Este sistema foi desenvolvido sobre uma plataforma

reconfigurável comercial executando o sistema operativo uCLinux, tendo sido desenvolvido um conjunto de módulos dedicados para interface com os periféricos e pré-processamento dos dados recolhidos de sensores.

Até agora foram apenas implementados processos de controlo simples mas que já permitiram demonstrar o potencial de navegação autónoma do veleiro FASt. A plataforma de computação desenvolvida disponibiliza um conjunto de recursos convenientes para a experimentação de diferentes técnicas de controlo dinâmico e de estratégias de decisão que serão agora desenvolvidas e que são vitais para se conseguir uma eficaz navegação à vela.

Para além do melhoramento dos processos de navegação e controlo, futuros desenvolvimentos serão focados na exploração das potencialidades da FPGA como tecnologia de suporte a esta implementação visando minimizar o consumo de energia global do sistema. Trabalhos em curso incluem a gestão dinâmica do sinal de relógio usado pelo processador Microblaze e a migração para hardware dedicado de alguns dos processos actualmente implementados em software. Outras ideias a avaliar consistem em explorar o processo de reconfiguração (total) da FPGA que é suportado por ferramentas incluídas no sistema operativo para seleccionar entre um conjunto de configurações hardware/software adequadas ao tipo de navegação em curso.

## Referências


[1] T. Statheros, G. Howells, and K. McDonald-Maier, "Autonomous ship collision avoidance navigation concepts, technologies and techniques," *The Journal of Navigation*, no. 61, pp. 129–142, 2008.

[2] M. R. Benjamin, J. J. Leonard, J. A. Curcio, and P. M. Newman, "A method for protocol-based collision avoidance between autonomous marine surface craft," *Journal of Field Robotics*, vol. 5, no. 23, pp. 333–346, 2006.

[3] A. Tiano, A. Zirilli, C. Yang, and C. Xiao, "A neural autopilot for sailing yachts," in *Proceedings of the 9th Mediterranean Conference on Control and Automation*, 2001, pp. 27–29.

[4] E. C. Yeh and J.-C. Bin, "Fuzzy control for self-steering of a sailboat," in *Proceedings of the Singapore International Conference on Intelligent Control and Instrumentation*, vol. 2, 1992, pp. 1339–1344.

[5] R. Stelzer, T. Prol, and J. Robert I, "Fuzzy logic control system for autonomous sailboats," in *Proceedings of the IEEE International Fuzzy Systems Conference*, 2007, pp. 1–6.

[6] M. L. van Aartrijk, C. P. Tagliola, and P. W. Adriaans, "AI on the ocean: the robosail project," in *Proceedings of the European Conference on Artificial Intelligence*, 2002, pp. 653–657.

[7] W. H. Warden, "A control system model for autonomous sailboat navigation," in *IEEE Proceedings of the Southeast Conference*, vol. 2, 1991, pp. 944–947.

[8] M. Neal, "A hardware proof of concept of a sailing robot for ocean observation," *IEEE Journal of Oceanic Engineering*, vol. 31, no. 2, pp. 462–469, 2006.

[9] R. Stelzer and T. Proll, "Autonomous sailboat navigation for short course racing," *Robotics and Autonomous Systems*, vol. 56, no. 7, pp. 604–614, 2008.

[10] G. Elkaim and R. Kelbley, "Station keeping and segmented trajectory control of a wind-propelled autonomous catamaran," in *45th IEEE Conference on Decision and Control*, vol. 13, no. 15, 2006, pp. 2424–2429.

[11] C. O. Boyce and G. Elkaim, "Control system performance of an unmanned wind-propelled catamaran," in *IFAC Conference on Control Applications in Marine Systems*, 2007.

[12] Atmark-Techno, "SUZAKU-SZ130-U00, Hardware Manual (English version) V1.0.2, [online]. Atmark-Techno [cited 22 november 2008]. Available from World Wide Web: <http://www.atmark-techno.com>," 2006.

[13] u-blox, "u-blox 5 NMEA, UBX Protocol Specification, Public Release [online]. u-blox AG, Switzerland [cited 22 november 2008]. Available from World Wide Web: <http://www.u-blox.com/customersupport/gps.g5/u-blox5_Protocol_Specifications(GPS.G5-X-07036).pdf>," 2008.