Universidade de São Paulo
Escola de Engenharia de São Carlos
Departamento de Engenharia Elétrica

Estudo de Viabilidade de Sistemas Operacionais em Sistemas Embarcados de Pequeno Porte

Vitor Jordão

São Carlos - SP

VITOR JORDÃO

Estudo de Viabilidade de Sistemas Operacionais em Sistemas Embarcados de Pequeno Porte

Trabalho de Conclusão de Curso apresentado à Escola de Engenharia de São Carlos, da Universidade de São Paulo.

Orientador: Prof. Dr. Evandro L. L. Rodrigues

São Carlos
2012

Jordão, Vitor
J82e Estudo de Viabilidade de Sistemas Operacionais em Sistemas Embarcados de Pequeno Porte / Vitor Jordão; orientador Evandro Luís Linhari Rodrigues. São Carlos, 2012.

Monografia (Graduação em Engenharia de Computação) -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2012.

1. Sistemas Embarcados. 2. Sistemas Operacionais. 3. Microcontroladores de 8-bits. I. Título.

FOLHA DE APROVAÇÃO

Nome: Vitor Jordão

Título: "Estudo de Viabilidade de Sistemas Operacionais em Sistemas Embarcadas de Pequeno Porte"

Trabalho de Conclusão de Curso defendido em 22/11/2012.

Comissão Julgadora:	**Resultado:**
Prof. Associado Adilson Gonzaga SEL/EESC/USP	Aprovado
Prof. Associado Eduardo Marques SSC/ICMC/USP	Aprovado

Orientador:
Prof. Associado Evandro Luís Linhari Rodrigues - SEL/EESC/USP

Coordenador pela EESC/USP do Curso de Engenharia de Computação:

Prof. Associado Evandro Luís Linhari Rodrigues

Dedicatória

Dedico este trabalho aos meus pais pelo amor e dedicação dispensados no meu desenvolvimento pessoal, na educação oferecida e nas dificuldades que sempre me ajudaram a superar. Aos meus dois irmãos por nunca deixarem florescer em mim qualquer tipo de sentimento egoísta e mesquinho. A minha família como um todo, especialmente aos meus primos por serem sempre mais que irmãos. Aos amigos de décadas que mantenho em Rio Claro, aos que criei em São Carlos nos meus anos de graduação e aos que este ano me permitiu criar em São Paulo.

Agradecimentos

Agradeço meus pais por todo o suporte que sempre me ofereceram, aos meus irmãos, avós e familiares pelo prazer que tenho em tê-los em minha vida.

A todos que de alguma maneira participaram dessa jornada de cinco anos de universidade e que contribuíram de alguma forma, mesmo que singela, para minha formação.

À Universidade de São Paulo e todos os seus docentes e funcionários por toda a estrutura oferecida a mim durante esses anos.

Ao Grupo Ultra e a Whirlpool S.A. pelas oportunidades oferecidas a mim neste ano, de me integrar aos seus quadros de estagiários e acreditar que meu desenvolvimento profissional se encaixaria em suas necessidades.

Um agradecimento especial ao Professor Evandro Luis Linhari Rodrigues pelas conversas no início do semestre que me ajudaram a encontrar um norte para este último ano de graduação e pela orientação neste trabalho.

E, finalmente, a todos os amigos e colegas de curso que estiveram comigo todas as semanas desses últimos cinco anos, por sempre me considerarem uma boa companhia, dentro e fora da universidade.

Cuidado para não chamar de inteligentes

apenas aqueles que pensam como você.

Ugo Ojetti

Resumo

Ao analisarmos as características do mercado de desenvolvimento de Sistemas Embarcados ao redor do mundo, vemos que a utilização de Sistemas Operacionais é grande. A opção por substituir a clássica arquitetura *Superloop* é resultado da complexidade que o software embarcado vem apresentando devido ao aumento das funcionalidades exigidas de um Sistema Embarcado. Entre as principais vantagens da utilização dessa arquitetura de software está a facilidade de gerenciamento de multitarefas e a disponibilidade de bibliotecas e serviços. Contudo esse crescimento parece ser restrito a arquiteturas de grande porte, como sistemas de 32 e 64 bits. A não utilização dessa arquitetura principalmente em sistemas de 8 bits não condiz com a expansão dessa faixa de microcontroladores, que hoje representa uma fatia considerável do mercado, principalmente brasileiro. A taxa elevada de utilização também não é característica do mercado nacional, muito pressionado por prazo e custo. Esse trabalho tem o objetivo de analisar a viabilidade de utilização de Sistemas Operacionais em Sistemas Embarcados de pequeno porte (8 bits) e o porque da reticência do mercado brasileiro em adotar essa arquitetura de software. Para isso são discutidas as características das duas arquiteturas de software citadas e quais as vantagens, desvantagens e exigências de cada uma delas ao serem embarcadas em microcontroladores de pequeno porte (8 bits) e as características do estudo de Sistemas Embarcados nas universidades brasileiras, visando identificar se a formação do profissional influencia nesse processo. Os resultados mostram que é possível essa utilização, contanto que se cumpram algumas exigências de hardware, como capacidade de memória e processamento. Entre as principais famílias de microcontroladores existem opções de microcontroladores de pequeno porte com condições de atender as necessidades exigidas por um Sistema Operacional, porém enquanto o mercado brasileiro for pressionado por custo e prazo, microprocessadores de menor desempenho continuarão a ser usados, por serem mais baratos. A dificuldade de atender ao prazo também estimula o máximo de reuso possível, tanto de hardware quanto de software, o que dificulta ainda mais a troca da arquitetura do software.

Palavras-chave: Sistemas Embarcados, Sistemas Operacionais.

Abstract

Analyzing the characteristics of the Embedded Systems development market around the world, we see that the use of Operating Systems is extensive. The decision of replacing the classical Superloop architecture results from the complexity that the embedded software has shown due to the increased functionality required by an Embedded System. Among the main advantages of using this software architecture is the ease of managing multitasks and the availability of libraries and services. However, this growth seems to be restricted to large architectures, such as 32 and 64-bits systems. Not using this architecture, primarily in 8-bit systems, is not consistent with the expansion of this range of microcontrollers, which now accounts for a considerable share of the market, mainly in Brazil. The high rate of use is not characteristic of the domestic market, which is very much pressed for deadlines and cost. The purpose of this study is to examine the feasibility of using Operating Systems in small Embedded Systems (8-bits) and, because of the reticence of the Brazilian market, to adopt this software architecture. The study discusses the characteristics of the two software architectures cited and the advantages, disadvantages and requirements of each one to be embedded in small microcontrollers (8-bit), as well as the characteristics of the study of Embedded System in Brazilian universities, in order to identify whether professional education influences this process. The results show that it can be used, as long as they meet certain hardware requirements, such as memory capacity and processing consumption. Among the main microcontroller families there are options of small microcontrollers with conditions to meet the needs required by an Operating System. However, while the Brazilian market is pressed for deadlines and cost, lower performance microprocessors are still used because they are cheaper. The difficulty meeting the deadlines also encourages as much reuse as possible, both of hardware and software, which makes the exchange of software architecture even more difficult.

Keywords: Embedded Systems, Operating Systems

Sumário

Lista de Figuras

Lista de Tabelas

CAPÍTULO 1: INTRODUÇÃO

A evolução da microeletrônica é considerada por muitos como uma das principais conquistas do ser humano no século XX. Tendo como estopim a invenção do transistor, apresentado ao mundo em 23 de Dezembro de 1948, a eletrônica deu um salto descomunal nas décadas de 50 e 60.

O advento de circuitos integrados possibilitou o desenvolvimento no início da década de 60 de computadores como o AGC (*Apollo Guidance Computer*), computador de bordo que provia recursos computacionais e controles para orientação, navegação e controle do Módulo de Comando e do Módulo Lunar utilizados no Projeto Apollo e considerado o primeiro Sistema Embarcado.

Desde então o uso de Sistemas Embarcados se popularizou. Seu uso é o mais diversificado possível, passando por aplicações militares, brinquedos, telefones celulares, eletrodomésticos, automóveis, automação industrial entre outros.

Essa evolução levou a uma complexidade natural das aplicações, que cada vez mais se adaptam as necessidades do consumirdor e do mercado de desenvolvimento de Sistemas Embarcados. Os novos *tablets* e *smartphones* lançados mensalmente pelo mercado possuem capacidade de processamento superior a muitos computadores pessoais, e muitos estudos inclusive questionam sua classificação como Sistema Embarcado.

No sentido de auxiliar esse desenvolvimento e facilitar o trabalho dos desenvolvedores, os Sistemas Operacionais estão sendo cada vez mais usados em aplicações embarcadas, deixando de ser exclusividade de computadores de propósito geral.

Esse mutualismo entre o desenvolvimento de hardware e o desenvolvimento de software exige mais preparação dos profissionais do mercado de trabalho, que necessitam de versatilidade para lidar com as especificidades de duas áreas. Essa versatilidade, por sua vez, está diretamente ligada a formação desses profissionais.

1.1. Objetivos do Trabalho

O objetivo deste Trabalho de Conclusão de Curso é estudar e analisar a viabilidade de utilização de Sistemas Operacionais em Sistemas Embarcados de pequeno porte (arquitetura 8 bits) e identificar os motivos que levam o mercado de desenvolvimento de Sistemas Embarcados nacional a não incorporarem maciçamente essa arquitetura de software, principalmente nos sistemas de pequeno porte.

Também é parte do escopo deste trabalho apresentar as características da abordagem de universidades brasileiras sobre Sistemas Embarcados e a visão que seus alunos têm sobre esse nicho de atuação no mercado de trabalho, comparando-a com as características do mercado de desenvolvimento brasileiro e internacional, com o objetivo de identificar se a reticência do mercado de desenvolvimento nacional em adotar Sistemas Operacionais está relacionada à formação profissional.

1.2. Justificativas

O artigo *Operating System On The Rise*, baseado na pesquisa *CMP United Business Media 2006*, apontava que, já no ano de 2006, 70% dos projetos de Sistemas Embarcados utilizavam Sistemas Operacionais. Essa taxa elevada se mostrou constante desde então, de acordo com a pesquisa anual *Embedded Market Survey* realizada pela *UBM Electronics* desde 2008, sendo o último estudo realizado neste ano de 2012.

Essa alta porcentagem é resultado da complexidade que o software embarcado vem apresentando com o passar dos anos, em decorrência da evolução dos componentes de hardware. Essa evolução do hardware trouxe novas exigências ao software, que precisou se modernizar também.

Entre as principais vantagens da utilização dessa arquitetura de software destacam-se a facilidade de gerenciamento de multitarefas e a disponibilidade de bibliotecas e serviços que facilitam o desenvolvimento de software.

Contudo esse crescimento parece ser restrito a arquiteturas de grande porte, como sistemas de 32 e 64 bits. A não utilização dessa arquitetura principalmente em sistemas de 8 bits não condiz com a expansão dessa faixa de microcontroladores, que hoje representa uma fatia considerável do mercado, principalmente brasileiro.

Considerando as características do mercado de desenvolvimento brasileiro, apresentadas por Borges (2011b), apenas 28% dos projetos de Sistemas Embarcados utiliza Sistema Operacional, o que foge completamente ao panorama mundial.

A abordagem que as universidades brasileiras adotam sobre Sistemas Embarcados pode mostrar se a reticência do mercado brasileiro em adotar Sistemas Operacionais em projetos de Sistemas Embarcados tem origem na formação do profissional ou não, uma vez que, segundo Borges (2011b), mais de 50% dos profissionais que atuam nessa área tem menos de 5 anos de experiência.

Essa análise também visa identificar se a tecnologia presente nas universidades está se refletindo no mercado brasileiro e quais as semelhanças e lacunas entre o conhecimento adquirido na universidade e as expectativas do mercado de trabalho.

No Brasil, as universidades são os principais centros de produção científica e espera-se que o mercado de trabalho explore essa evolução, por ser o principal beneficiário dessas novas metodologias.

Um melhor casamento entre o que é desenvolvido nas universidades com o que é aplicado no mercado favorece a inserção do aluno recém-formado nesse mercado, diminui seu tempo de adaptação e representa um ganho para o empregador e os projetos desenvolvidos.

Por fim, vale ressaltar a relevância desse estudo para o complemento do curso de Engenharia de Computação com ênfase em Sistemas Embarcados, pois engloba características de hardware e software, focando sempre na integração entre ambos.

1.3. Organização da Monografia

O Capítulo 2 apresenta a fundamentação teórica necessária para o entendimento desta Monografia.

O Capítulo 3 apresenta a metodologia seguida durante o processo de elaboração do trabalho.

O Capítulo 4 apresenta os resultados da enquete realizada e analisa e discute a viabilidade de utilização de Sistemas Operacionais em Sistemas Embarcados de pequeno porte.

O Capítulo 5 apresenta as conclusões a respeito do trabalho desenvolvido e propostas de trabalhos futuros.

CAPÍTULO 2: EMBASAMENTO TEÓRICO

Neste capítulo são apresentados os conceitos necessários para o entendimento e o desenvolvimento do projeto em questão.

2.1. Sistemas Embarcados

Sistemas Embarcados são sistemas computacionais completos e independentes, encarregados de executar uma tarefa ou um conjunto restrito de tarefas específicas e correlacionadas, na qual executam, geralmente, repetidas vezes. Esses dispositivos são compostos fundamentalmente pelos mesmos componentes de um computador pessoal ou de propósito geral – processador, memória, dispositivos de armazenamento, interfaces e demais componentes – com a diferença de apresentarem maiores limitações de funcionalidades de hardware e software. (BORGES, 2011a)

Essa limitação é encontrada em um subconjunto significativo dessa família. Em termos de limitações de hardware, pode significar limitações no desempenho de processamento, no consumo de energia, na memória, na funcionalidade do hardware, e assim por diante. Em termos de software, normalmente significa aplicações em escala reduzida, nenhum sistema operacional ou um sistema operacional limitado, ou menor nível de abstração de código. (NOERGAARD, 2005)

Sistemas Embarcados são muito utilizados em nosso dia-a-dia e estão transformando o modo de vida, trabalho e diversão das pessoas. Estão presentes desde um simples brinquedo, eletrodomésticos, dispositivos de rede, eletrônica de consumo a instrumentos médicos, indústria automotiva, aviação, etc. (BORGES, 2011a)

A evolução da microeletrônica e o barateamento dos microcontroladores, além de sua versatilidade, viabilizaram o emprego de sistemas embarcados em diversos desses equipamentos. (BORGES, 2011a)

Contudo ainda não existe uma definição clara e concreta do que é um Sistema Embarcado. A principal causa dessa indefinição é o fato de o próprio campo de Sistemas Embarcados, nos últimos anos, superar muitas de suas definições clássicas. (NOERGAARD, 2005)

Um exemplo é se devemos ou não considerar os celulares de última geração como sendo Sistemas Embarcados. Apesar de ser enxergado classicamente como um sistema embarcado, possuem o poder de processamento e gerenciamento de tarefas de um computador pessoal.

As definições tradicionais devem continuar a evoluir, ou um novo campo de sistema computacional ser designado para incluir esses sistemas mais complexos. Por enquanto não há campo na indústria apoiando uma nova designação para projetos que se enquadram entre o sistema embarcado tradicional e sistemas de propósitos gerais, o que torna perfeitamente aceitável chamá-los de Sistemas Embarcados. (NOERGAARD, 2005)

Apesar dessa dificuldade em definir o que é de fato um Sistema Embarcado, a maioria divide certas propriedades importantes como apontam Buttazzo (2006) e Berger (2002):

- Recursos limitados: muitos sistemas embarcados são desenvolvidos sobre restrições de espaço, peso e energia, imposto pela aplicação.
- Sensíveis a custo: frequentemente apresentam restrições de custo devido à produção em massa e grande competição industrial, o que torna mandatório o uso altamente eficiente dos recursos computacionais.
- Limitações de tempo real: a maioria dos dispositivos embarcados interage com o ambiente e deve reagir a eventos externos e executar atividades computacionais dentro de restrições precisas de tempo.
- Comportamento dinâmico: consistem de dezenas ou centenas de tarefas concorrentes que interagem entre si para uso de recursos compartilhados.
- Diferentes processadores: sistemas embarcados são suportados por uma grande quantidade de processadores e arquiteturas de processadores.

Friedrich (2009) define onze qualidades que definem requisitos não funcionais utilizados para julgar a operação dos sistemas embarcados. São elas:

- Recursos limitados de computação: os recursos computacionais devem ser utilizados de maneira eficiente.
- Requisitos de tempo real: Sistemas Embarcados interagem com o ambiente, necessitando reagir corretamente dentro de requisitos estritos de tempo.
- Portabilidade: diferentes tipos de CPU’s, periféricos e memórias podem ser usados em sistemas embarcados.
- Alta confiabilidade: sistemas embarcados são usados remotamente e em aplicações críticas, o que torna a correção de falhas problemática, extremamente cara e até mesmo impossível de correção.

- Robustez e estabilidade do sistema: o sistema deve operar fora de condições nominais e evitar interrupções na operação.
- Tratamento de falhas: os sistemas devem identificar e tratar erros e garantir tolerância a falhas da aplicação.
- Operação segura: os sistemas devem prevenir ferimentos, perdas de vida e danos a propriedades e ao meio ambiente.
- Segurança de informações: os sistemas devem evitar que informações internas sejam usadas ou alteradas por usuários não autorizados.
- Privacidade: os sistemas devem possuir a habilidade de isolar e revelar seletivamente informações.
- Escalabilidade: o sistema deve ser capaz de gerenciar o aumento na carga de trabalho e possibilitar a expansão.
- Atualização: os sistemas devem permitir que a especificação seja aprimorada, adicionando ou substituindo componentes.

Sistemas embarcados em geral apresentam maiores requisitos de qualidade e confiabilidade, comparados a outros sistemas computacionais. (NOERGAARD, 2005)

2.2. Software Embarcado

De acordo com a pesquisa *Embedded Market Study* relizada pela *UBM Electronics* (2012), desde 2009 62% dos projetos consomem mais recursos no desenvolvimento de software do que no desenvolvimento de hardware. Essa pesquisa aponta também que, entre 40% e 50% dos projetos têm um número maior de pessoas trabalhando no desenvolvimento de software do que no desenvolvimento de hardware.

Esses números mostram a importância desse segmento no desenvolvimento de Sistemas Embarcados, sendo que o software se tornou componente chave nesse tipo de projeto.

Software Embarcado não é apenas um software para executar em computadores de pequeno porte e, por isso, necessita ser enxergado de um ponto de vista diferente da definição convencional. (LEE, 2002)

Sua função principal não é a transformação de dados, mas sim a interação com o mundo físico. Ele executa em máquinas que não são, em primeiro lugar, computadores convencionais. São automóveis, aviões, robôs, brinquedos, eletrodomésticos, e assim por diante. (LEE, 2002)

Essa classe especial de software apresenta certas características especiais que a diferencia de softwares convencionais, desenvolvidos para a computação de propósito geral: (LEE, 2002)

- Precisão Temporal: Mesmo com a evolução da computação, o software embarcado ainda tem que lidar com o tempo, já que os processos físicos, com os quais interage, envolvem tempo.
- Concorrência: Sistemas Embarcados raramente interagem com apenas um processo físico. Eles devem reagir simultaneamente a estímulos de uma variedade de sensores e, ao mesmo tempo, manter o controle sobre seus atuadores.
- Continuidade: Os programas não devem encerrar ou bloquear sua execução esperando por eventos que nunca vão ocorrer.
- Interfaces: A utilização de interfaces em software consiste em dividir um projeto grande e complicado em peças menores que oferecem interfaces que abstraem as funcionalidades dessa peça.
- Heterogeneidade: Sistemas Embarcados apresentam frequentemente uma mistura de projetos de hardware, software e estilos de manipulação de eventos, pois interagem com os eventos que ocorrem de forma irregular no tempo, como alarmes, e regulares no tempo como sinais de controle de um atuador.
- Reatividade: Sistemas reativos são aqueles que reagem de forma contínua ao seu ambiente, à velocidade do meio ambiente, diferentes de sistemas interativos, que reagem ao meio ambiente na sua própria velocidade. Sistemas reativos têm restrições de tempo real e tem a tendência de serem críticos.

Além das características citadas, pode-se acrescentar a restrição de tempo real, baixo consumo de energia e balanço entre hardware e software.

Existem diferentes arquiteturas para o desenvolvimento de software embarcado. Entre as principais arquiteturas podemos destacar: *Foreground/Background* ou arquitetura *Superloop* e os Sistemas Operacionais. (BORGES, 2011a)

2.3. Arquiteturas de Software Embarcado

Esta sessão apresenta as duas principais arquiteturas de software utilizadas em projetos de Sistemas Embarcados.

2.3.1 Arquitetura *Foreground/Background* ou Arquitetura *Superloop*

A arquitetura *Foreground/Background*, também conhecida como arquitetura *Superloop*, é a solução mais comum para aplicações embarcadas. Ela envolve um conjunto de processos orientados a interrupção chamados de *Foreground* e uma coleção de chamadas de módulos em *loop* infinito, chamados de *Background*. As tarefas de *Foreground* executam em primeiro plano e são as tarefas de maior prioridade no sistema, enquanto que as tarefas de *Background* representam as tarefas de menor prioridade e são sempre interrompidas quando existe a necessidade de execução de uma tarefa de *Foreground*. (SINGH, 2002)

Operações críticas devem ser executadas no nível de interrupção (*Foreground*) para garantir que atendam as restrições de tempo dessas tarefas. (SIMON, 1999)

A Figura 1 mostra o funcionamento dessa arquitetura. Nela, é possível verificar a interrupção de tarefas de *Background* quando existe a necessidade da execução de uma tarefa de *Foreground*.

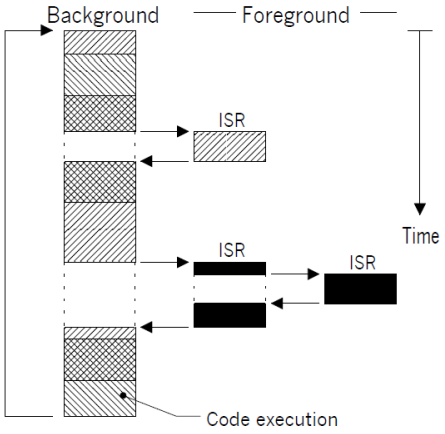

Figura 1 - Funcionamento da Arquitetura *Superloop*. Fonte: (LABROSSE 2002)

Essa arquitetura normalmente têm bons tempos de resposta, uma vez que dependem de hardware para executar o escalonamento de tarefas. (SINGH, 2002)

Apesar dessa arquitetura fornecer maior controle sobre as prioridades de tarefas, apresenta ao menos um grande problema: as interfaces para dispositivos mais complexos devem ser escritas. Esta escrita tende a ser tediosa e propensa a erros. Além disso, esse tipo de sistema é melhor implementado quando o número de tarefas de *Foreground* é conhecido a priori. Finalmente, por

conter apenas um nível de interrupção, o sistema se torna vulnerável às variações de temporização, condições de concorrência inesperadas, falhas de hardware, etc. (SINGH, 2002)

A maioria das aplicações de alto volume que utilizam microcontroladores (micro-ondas, lavadoras, refrigeradores, brinquedos, telefones, etc.) é desenvolvida utilizando sistemas *Background/Foreground*. (LABROSSE, 2002; BORGES, 2011a)

2.3.2 Sistemas Operacionais

Há muitos anos tornou-se bastante evidente a necessidade de encontrar uma maneira de isolar os programadores da complexidade do hardware. A maneira com que isso se desenvolveu gradualmente foi colocar uma camada de software por cima do hardware básico para gerenciar todas as partes do sistema e oferecer ao usuário uma interface que é mais fácil de entender e de programar. Essa camada é o Sistema Operacional (SO). (TENEMBAUM; WOODHULL, 2000)

Os Sistemas Operacionais variam muito em relação a sua composição, entretanto todos são compostos por, no mínimo, um *kernel*, ou núcleo, que é o componente que contém as funcionalidades principais do Sistema Operacional. (TENEMBAUM; WOODHULL, 2000)

São quatro as principais funcionalidades: gerenciamento de processos, gerenciamento de memória, gerenciamento do sistema de arquivos e gerenciamento de entrada e saída.

Os Sistemas Operacionais executam basicamente duas funções não relacionadas: (TENEMBAUM; WOODHULL, 2000)

O SO como Máquina Estendida: Deste ponto de vista, a função do sistema operacional é apresentar ao usuário o equivalente a uma máquina virtual que é mais fácil de programar que o hardware subjacente.

O SO como um Gerenciador de Recursos: O trabalho do sistema operacional é oferecer uma alocação ordenada dos dispositivos e outros programas, monitorando quem está utilizando qual recurso, atender requisições e controlar requisições conflitantes.

Os SO's dividem-se em dois grupos principais: Sistema Operacional de Propósito Geral ou GTOS (*General-Purpose Operating System*) e de Sistemas Operacionais de Tempo Real ou RTOS (*Real-Time Operating System*). (LI; CAROLYN, 2003)

O primeiro grupo é utilizado principalmente em computação de propósitos gerais, como PC's e Mainframes, enquanto que o segundo grupo se adapta melhor a sistemas embarcados com restrição de tempo real. (LI; CAROLYN, 2003)

RTOS's apresentam melhor confiabilidade para aplicações embarcadas, como baixos requisitos de memória, políticas de escalonamento mais adequadas a sistemas embarcados de tempo real, portabilidades para diferentes plataformas e melhor desempenho. (LI; CAROLYN, 2003)

2.3.3 Real-Time Operating System - RTOS

Um RTOS é um Sistema Operacional que gerencia recursos e escalona processos com restrição de tempo. (LI; CAROLYN, 2003)

Em algumas aplicações, o RTOS pode ser composto apenas pelo *kernel*, que é responsável pela lógica mínima do sistema, escalonamento de tarefas e gestão de recursos mínimos. Caso a aplicação exija um RTOS mais complexo, podem ser incluídos uma combinação de vários outros módulos, como sistema de arquivos, pilhas de protocolos de redes, e outros componentes necessários. (LI; CAROLYN, 2003)

Todo *kernel* possui algumas construções especiais, que ajudam os desenvolvedores a criar seus aplicativos. Os objetos *kernel* mais comuns são as tarefas, os semáforos e as filas de mensagens. (LI; CAROLYN, 2003)

Tarefas

Aplicações simples são tipicamente desenvolvidas para executar sequencialmente. Esse esquema é inapropriado para aplicações embarcadas de tempo real, que geralmente precisam lidar com diversas entradas e saídas dentro das restrições de tempo. Aplicações de tempo real devem ser desenvolvidas para serem concorrentes. (LI; CAROLYN, 2003)

Isso exige que a aplicação seja decomposta em unidades pequenas, escalonáveis e sequenciais. A concorrência permite satisfazer as exigências de desempenho e tempo de um sistema de tempo real. A maioria dos kernels de RTOS fornecem objetos chamados tarefas e serviços de gerenciamento dessas tarefas para facilitar a concorrência na aplicação. (LI; CAROLYN, 2003)

Uma tarefa é uma *thread* independente que compete com outras tarefas por tempo de execução. (LI; CAROLYN, 2003)

Multitasking é a habilidade do sistema operacional em lidar com múltiplas atividades nos prazos determinados (*deadlines*). Esta habilidade permite que tarefas pareçam executar paralelamente, sendo que, na verdade, estão tendo sua execução dividida em parcelas de tempo e entrelaçadas, de acordo com um algoritmo de escalonamento. (LI; CAROLYN, 2003)

Essa propriedade impede que uma tarefa ocupe o processador enquanto espera por algum outro recurso e, com isso, corrompa o desempenho do sistema. Ela permite ao processador suspender a execução da tarefa que espera dados externos e passar a executar outra. Quando os dados de que necessita estiverem disponíveis, a tarefa suspensa pode ser retomada no ponto onde parou. O ato de retirar um recurso de uma tarefa é denominado preempção. Sistemas que implementam esse conceito são chamados sistemas preemptivos. (MAZIERO, 2011)

O diagrama de estados da Figura 2 ilustra o comportamento de uma tarefa em um sistema *Multitasking*: (MAZIERO, 2011)

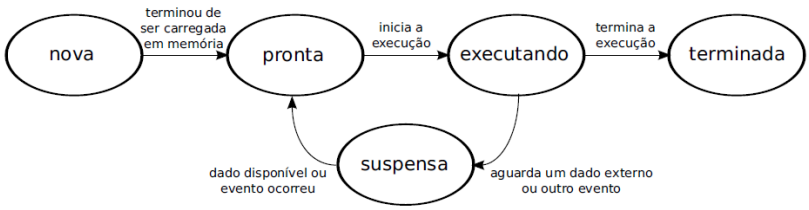

Figura 2 – Diagrama de Ciclo de Vida de uma Tarefa. Fonte: (MAZIERO, 2011)

O diagrama apresentado na Figura 2 é conhecido como diagrama de ciclo de vida das tarefas. Os estados do ciclo de vida têm o seguinte significado: (MAZIERO, 2011)

Nova : Tarefa está sendo criada, ou seja, seu código está sendo carregado em memória.

Pronta : Tarefa está pronta para executar ou continuar sua execução, apenas aguardando a disponibilidade do processador.

Executando : O processador está dedicado à tarefa.

Suspensa : Tarefa não pode executar porque depende de dados externos ainda não disponíveis, aguarda algum tipo de sincronização, como fim de outra tarefa ou a liberação de algum recurso compartilhado.

Terminada : O processamento da tarefa foi encerrado.

Semáforos

Um sistema que permita *Multitasking* deve ser capaz de sincronizar a execução das tarefas e coordenar o acesso a recursos compartilhados. Para atender esses requisitos, o *kernel* fornece um objeto chamado semáforo e serviços de gerenciamento associados a ele. (LI; CAROLYN, 2003)

Semáforo é um objeto de *kernel* que uma ou mais *threads* podem adquirir ou liberar com o propósito de sincronizar sua execução ou garantir a exclusão mútua de algum recurso. (LI; CAROLYN, 2003)

Se a tarefa adquirir o semáforo, ela pode realizar a operação pretendida ou acessar o recurso que necessita. (LI; CAROLYN, 2003)

Um valor binário ou um contador controla a utilização ou não do semáforo. Quando uma tarefa adquire o semáforo, esse valor é decrementado, quando libera o semáforo, esse valor é incrementado. Se tiver valor maior que zero, a tarefa adquire o semáforo pra ela, seu valor é decrementado e ela está apta a realizar a operação necessária. Caso contrário, se ao necessitar do semáforo, esse valor é zero, a tarefa é bloqueada pelo *kernel* e entra na fila de espera. Essa tarefa então passa a esperar a liberação do semáforo, respeitando a lógica FIFO (*First In First Out*). (LI; CAROLYN, 2003)

Um *kernel* pode suportar diferentes tipos de semáforos, como binários, contadores e semáforos de exclusão mútua (*mutex*). (LI; CAROLYN, 2003)

Filas de Mensagens

Em muitos casos, somente a sincronização das tarefas não garante que as restrições da aplicação sejam atendidas. As tarefas devem ser capazes de se comunicar via troca de mensagens. Para facilitar essa comunicação de dados entre tarefas, o *kernel* fornece um objeto chamado fila de mensagem e serviços associados a esse objeto. (LI; CAROLYN, 2003)

Uma fila de mensagem é um objeto que funciona como um *buffer*, por meio do qual tarefas enviam e recebem mensagens para comunicação e sincronização de dados. A fila de mensagens retém temporariamente mensagens de um remetente até que a tarefa destinatária esteja pronta para lê-la. (LI; CAROLYN, 2003)

A fila de mensagens tem duas listas de tarefas em espera associadas a ela. A lista de tarefas de destinatários consiste das tarefas que esperam uma mensagem e a lista de mensagens de

remetentes consiste das tarefas que esperam para enviar uma mensagem, quando a fila de mensagens está cheia. (LI; CAROLYN, 2003)

Escalonador

Um dos componentes mais importantes da gerência de tarefas é o escalonador. É ele quem fornece os algoritmos necessários para determinar qual e quando uma tarefa pronta será executada. (LI; CAROLYN, 2003; MAZIERO, 2011)

Uma entidade escalonável é um objeto de kernel que compete por uma fatia de tempo de execução em um sistema, baseado em um algoritmo de escalonamento pré-definido. Tarefas é um exemplo de entidade escalonável. (LI; CAROLYN, 2003)

O algoritmo utilizado no escalonador define o comportamento do sistema operacional, permitindo obter sistemas que tratem de forma mais eficiente e rápida a execução das tarefas. (MAZIERO, 2011)

Tarefas de tempo real exigem previsibilidade em seus tempos de resposta aos eventos externos. O escalonamento de tarefas de tempo real é um problema complexo. O algoritmo de escalonamento dessas tarefas deve garantir que sejam atendos os *deadlines* pré estabelecidos de cada tarefa. (LI; CAROLYN, 2003; MAZIERO, 2011)

O escalonador de um sistema operacional pode ser preemptivo ou não-preemptivo: (LI; CAROLYN, 2003)

Sistemas preemptivos : nestes sistemas uma tarefa pode perder o processador caso termine seu quantum de tempo, execute uma chamada de sistema ou caso ocorra uma interrupção que acorde uma tarefa de maior prioridade.

Sistemas não-preemptivos : a tarefa em execução permanece no processador tanto quanto possível, só abandonando o mesmo caso termine de executar, solicite uma operação de entrada/saída ou libere explicitamente o processador, voltando à fila de tarefas prontas.

A maioria dos *kernels* suportam dois tipos mais comuns de algoritimos de escalonamento, escalonamento preemptivo e *round-robin*, que normalmente são pré-determinados pelo fabricante do RTOS. No entanto, em alguns casos, os desenvolvedores podem criar e definir os seus próprios algoritmos de escalonamento. (LI; CAROLYN, 2003)

Um ponto importante a ser notado é que as tarefas seguem o algoritmo de escalonamento definido no *kernel*, enquanto que as ISRs são acionadas por interrupções de hardware e executadas de acordo com suas prioridades. (LI; CAROLYN, 2003)

2.4. Microprocessadores e Microcontroladores

O design de processadores de propósito geral é complexo devido a gama de recursos e funcionalidades que fornecem. Essa variedade de recursos resulta em grande consumo de energia, geração de calor, aumento do tamanho do chip e custo. (LI; CAROLYN, 2003)

Essas características inviabilizam sua utilização em sistemas embarcados, que primam pelo baixo custo, consumo, e por realizarem operações dedicadas, o que tornaria ocioso grande parte dos recursos desses microprocessadores.

Contudo, o avanço no desenvolvimento de microprocessadores permitiu o surgimento de processadores específicos para aplicações embarcadas, que fornecem diferentes focos dependendo das necessidades da aplicação.

Uma classe desses microprocessadores foca em tamanho, consumo de energia e preço. Para isso, costuma ser extremamente específico, sendo bom o suficiente somente para o tipo de aplicação para a qual foi desenvolvida. Essa dedicação extrema resulta em menos funcionalidades, redução do consumo de energia, tamanho e preço. (LI; CAROLYN, 2003)

Uma segunda classe foca em performance. Esses processadores são projetados para satisfazer aplicações com requisitos de computação intensiva, não realizável com processadores de propósito geral. (LI; CAROLYN, 2003)

Uma terceira classe de microprocessadores embarcados foca nas quatro características citadas. Um exemplo dessa classe, os DSP's (*Digital Signal Processors*), possuem unidades aritméticas especializadas e barramentos que permitem a realização de cálculos complexos com extrema velocidade. (LI; CAROLYN, 2003)

A maioria dos sistemas embarcados, no entanto, utiliza microcontroladores ao invés de microprocessadores. Microcontroladores ou MCU (*MicroControler Unit*) se diferenciam dos processadores por integrarem elementos adicionais em sua estrutura interna além dos componentes lógicos e aritméticos usuais.

2.4.1 Arquitetura Básica

Embora inicialmente todos os microcontroladores adotaram a arquitetura clássica de *von Neumann*, hoje em dia a arquitetura que se destaca é a *Harward*. A arquitetura de *von Neumann* se caracteriza por possuir uma única memória principal que armazena dados e instruções. Essa memória é acessada através de um barramento único, por onde circulam dados, endereços e sinais de controle. (PONTIFÍCIA..., 2012)

A arquitetuta *Harward* possui duas memórias independentes, uma que contém as instruções e outra que contém os dados. Ambas dispõe de seus respectivos barramentos e é possível realizar operações de acesso simultâneamente em ambas as memórias. (PONTIFÍCIA..., 2012)

A estrutura básica dos microcontroladores é muito parecida, sendo que cada microcontrolador enfatiza os recursos mais apropriados para as aplicações a que se destinam. Abaixo são apresentados os componentes básicos da arquitetura. (PONTIFÍCIA..., 2012)

Processador

O processador é o elemento mais importante do microcontrolador e determina as principais características deste, tanto em nível de hardware quanto de software.

É função do processador endereçar a memória de instruções, receber o código da instrução em curso, decodificá-lo, executar as operações designadas pela instrução e armazenar os resultados.

Memória

Nos microcontroladores, a memória de instruções e a memória de dados estão integradas no próprio chip. Parte dela é não volátil, como uma ROM (*Read-Only Memory*), e se destina a armazenar o programa de instruções que governa a aplicação. Outra parte, volátil, se destina a guardar as variáveis e os dados.

Portas de Entrada e Saída

A principal utilidade dos pinos que compõe a cápsula do microcontrolador é suportar as linhas de entrada e saída, que comunicam o computador interno com os periféricos externos. Os módulos de controle de periféricos presentes em cada microcontrolador é quem fornece o suporte para os sinais de entrada, saída e controle dessas linhas.

Clock Principal

Todos os microcontroladores possuem um circuito oscilador que gera uma onda quadrada de alta frequência, que controla a sincronização de todas as operações do sistema.

Geralmente esse circuito está incorporado ao microcontrolador e necessita de uns poucos componentes externos para selecionar e estabilizar a frequência de operação. O aumento da frequência de *clock* diminui o tempo de execução das instruções, porém eleva o consumo de energia do sistema.

2.5. Considerações sobre Projeto de Sistemas Embarcados

Não é pouco o número de microcontroladores disponíveis no mercado. Existem diversas famílias de microcontroladores divididas em diversas arquiteturas e fabricantes. Da mesma maneira, existem variações de arquiteturas de software para sistemas embarcados, cada uma com suas vantagens e desvantagens.

Essa diversidade traz uma primeira dúvida na hora de se iniciar um projeto de sistema embarcado. Qual microcontrolador e qual arquitetura de software empregar no sistema?

Na hora de escolher o microcontrolador deve-se levar em conta fatores como documentação e ferramentas de desenvolvimento disponíveis, as características do microcontrolador, como memória de programa, número de temporizadores, interrupções, etc. e, obviamente, o preço.

Por estarem presentes em aplicações com grande volume de vendas cada centavo economizado por unidade pode representar uma economia considerável na produção. Essa afirmação é ainda mais relevante ao mercado brasileiro. Segundo a pesquisa realizada por Borges (2011b), 69.7% dos desenvolvedores apontaram o custo como um dos fatores principais na seleção do microcontrolador.

Além de analisar as características do microcontrolador em si, é importante analisar as características da aplicação que está sendo desenvolvida. Caso a aplicação exija do microcontrolador cálculos críticos em tempo limitado, ou precise tratar dados com alta precisão, deve-se assegurar que o dispositivo seja adequado para isso. Se um microcontrolador não é suficiente, pode ser necessário substituí-lo por uma arquitetura diferente ou, se for o caso, procurar bibliotecas que auxiliem o tratamento de dados de alta precisão.

Novamente a pesquisa em Borges (2011b) exemplifica essa situação, ao mostrar que 43.9% dos desenvolvedores consideram o desempenho um fator crucial na escolha do microcontrolador e 40.9%, a disponibilidade de ferramentas. Esse segundo fator, segundo a CMP United Business Media (2006), é o principal fator de seleção em projetos de sistemas embarcados internacionais.

Alguns produtos que incorporam microcontroladores são alimentados por baterias. Desses produtos é esperado um baixo consumo de energia. Em aplicações como essas talvez seja mais interessante ter um microcontrolador que passe a maior parte do tempo em *stand-by* e acorde apenas em momentos de controle do que microcontroladores com grande poder de processamento e diversas ferramentas, o que aumenta o consumo.

Outro fator, de extrema importância, é determinar a quantidade de memória principal e secundária necessária à aplicação. A memória influencia em fatores como desempenho e escalabilidade do sistema, o que a liga diretamente à arquitetura de software empregada. É interessante, então, uma análise preliminar dessa arquitetura e de seus impactos antes de se determinar o quanto de memória é necessário ao sistema.

Considerando a afirmação de Bannatyne (2009) de que se há a possibilidade de solucionar um problema utilizando 8 bits, esta solução será sempre mais barata do que resolver o mesmo problema usando 32 bits, o tamanho da palavra de dados influencia no preço do produto. Microcontroladores com palavras de dados de 16 e 32 bits, que apresentam custo mais elevado que os de 4 e 8 bits, devem ser reservados para aplicações que requerem sua alta performance.

Por último, a seleção do microcontrolador implica diretamente na placa de circuitos em que será embarcado. Deve-se ter em mente que um microcontrolador mais barato e com menos recursos pode encarecer os outros componentes desta placa, o que talvez não represente uma economia no produto final.

Quanto à arquitetura de software, segundo Simon (1999), o fator mais importante que determina qual arquitetura será a mais apropriada para um determinado sistema é quanto de controle é necessário ter sobre as respostas desse sistema.

Esse fator não depende somente dos requisitos de tempo absoluto, mas também da velocidade do microcontrolador escolhido e de outros requisitos de processamento. Um sistema com poucas funcionalidades e tempos de resposta pouco rigorosos pode ser escrito com uma arquitetura mais simples. Por outro lado, um sistema que necessita responder rapidamente a diferentes eventos e com diversos requisitos de processamento, todos com diferentes *deadlines* e diferentes prioridades, irá necessitar de uma arquitetura mais complexa. (SIMON 1999)

Também segundo Simon (1999), a arquitetura escolhida deve ser a mais simples que atenda todos os requisitos do sistema, uma vez que escrever software para sistemas embarcados já é complicado o suficiente sem que seja escolhida uma arquitetura complexa e desnecessária.

Caso o sistema tenha requisitos que necessitem o uso de um RTOS, é interessante pesquisar as opções que o mercado oferece, uma vez que a maioria dos sistemas comerciais é vendida com uma coleção de ferramentas que facilitam o desenvolvimento, teste e debug do software. (SIMON 1999)

A pesquisa de Borges (2011b) aponta ser esse o principal fator na hora da escolha do Sistema Operacional. Segundo ele, 23.8% dos entrevistados apontaram a disponibilidade de ferramentas e serviços como fator determinante na hora de escolher o SO.

Se for pertinente ao sistema, pode-se considerar a utilização de uma arquitetura de software híbrida, que explore as vantagens de diversas arquiteturas para aquela aplicação. (SIMON 1999)

CAPÍTULO 3: METODOLOGIA

Este Trabalho de Conclusão de Curso pretende estudar a viabilidade de utilização de Sistemas Operacionais em Sistemas Embarcados de pequeno porte (8 bits), identificar os motivos que levam o mercado de desenvolvimento de Sistemas Embarcados nacional a não incorporarem maciçamente essa arquitetura, principalmente nos sistemas de pequeno porte e analisar qual a abordagem que as universidades brasileiras adotam sobre Sistemas Embarcados e sua relação com essa reticência do mercado em aceitar o uso de SO's.

A primeira etapa deste trabalho será justamente identificar a maneira como Sistemas Embarcados são estudados nas universidades brasileiras, qual a visão de seus alunos sobre o mercado de desenvolvimento de Sistemas Embarcados e, com o auxílio da pesquisa anual *Embedded Market Survey* realizada pela *UBM Electronics* (2012) com desenvolvedores de Sistemas Embarcados em todo o mundo e da pesquisa realizada por Borges (2011b) com desenvolvedores de Sistemas Embarcados no Brasil, identificar as semelhanças e lacunas entre o aprendizado nas universidades e as expectativas do mercado.

Para isso será realizada uma pesquisa com alunos universitários de cursos de graduação relacionados a Sistemas Embarcados. Esses cursos serão escolhidos de acordo com sua relação com a computação e a engenharia eletrônica, frentes principais relacionadas a Sistemas Embarcados, e de acordo com os dados presentes em Borges (2011b) sobre a formação dos profissionais que trabalham com desenvolvimento de Sistemas Embarcados no mercado nacional.

Essa pesquisa conta com 21 questões que pretendem avaliar três tópicos distintos: o perfil dos alunos dos cursos alcançados, qual a abordagem das universidades sobre o estudo de Sistemas Embarcados e qual a visão de seus alunos sobre o mercado de desenvolvimento de Sistemas Embarcados.

As questões relacionadas ao perfil do aluno abordam temas como formação, período letivo e pretensão de se especializar em Sistemas Embarcados. As questões relacionadas à abordagem da universidade sobre Sistemas Embarcados visam identificar quando esse assunto é normalmente introduzido, de que maneira ele é abordado e quais as arquiteturas de hardware e software exploradas. As questões relacionadas às expectativas do mercado foram elaboradas com base nas duas pesquisas citadas acima e abordam temas como arquitetura de hardware, arquitetura de software, linguagem de programação, principais desafios dos projetos e estratégias de desenvolvimento.

O questionário foi desenvolvido com o auxílio da ferramenta de Criação de Formulários, disponível no *Google Docs*. A ferramenta do *Google Docs* foi escolhida por fornecer uma maneira

rápida e fácil de desenvolvimento, pela autopublicação do formulário online, o que facilita sua divulgação, e pela facilidade que proverá para a análise de dados, por disponibilizar o conjunto de dados para análise de diferentes formas, o que possibilita diferentes visões do mesmo conjunto de dados.

O formulário foi publicado no seguinte link:

https://docs.google.com/spreadsheet/viewform?formkey=dEtsN096azN0aUhXVWNKaHZqVTNSbnc6MQ#gid=0

Esse link, antes de divulgado, foi disponibilizado a um grupo pequeno de pessoas, com conhecimento ou não de Sistemas Embarcados, que auxiliaram na correção do formulário e obtenção de clareza e objetividade tanto das questões quanto das sugestões de respostas.

Já validado, o formulário será divulgado em grupos de e-mail e em grupos de redes sociais dos cursos previamente selecionados de diversas universidades brasileiras. É essencial neste caso, então, a colaboração dos próprios alunos entrevistados na divulgação da pesquisa para que ela atinja um universo significativo e representativo de respostas.

A segunda etapa do trabalho será estudar e discutir a viabilidade de utilização de Sistemas Operacionais em Sistemas Embarcados de pequeno porte (8 bits). Essa análise será feita com base na literatura, no estudo das características dos microcontroladores e no estudo das principais arquiteturas de software para Sistemas Embarcados.

Para efeito comparativo será feita essa análise para duas diferentes arquiteturas de microcontroladores. A primeira é a arquitetura alvo de 8 bits. Serão estudadas as características da arquitetura, seus pontos fortes e fracos e como as novas tecnologias de desenvolvimento de componentes eletrônicos vêm influenciando essa arquitetura. A segunda arquitetura de estudo é a de 32 bits. Sua escolha é justamente para termos uma comparação entra arquiteturas de pequeno e grande porte. Outro fator de seleção é o fato de que, segundo Borges (2011b), são as duas arquiteturas de microcontroladores mais utilizadas em novos projetos de Sistemas Embarcados no Brasil.

Também serão analisadas duas arquiteturas distintas de software para Sistemas Embarcados. A primeira, os SO’s, serão estudados e analisados para que se possa levantar suas características, vantagens e desvantagens em projetos de Sistemas Embarcados. A segunda arquitetura de software a ser estudada é a arquitetura *Superloop*, que é principal arquitetura de software utilizada em projetos de Sistemas Embarcados, e as mesmas considerações que no caso anterior serão feitas.

Para o caso dos SO's serão consideradas apenas as características do tipo de SO mais comumente utilizados em Sistemas Embarcados, os Sistemas Operacionais de Tempo-Real. Nenhum dos estudos que serão realizados fará qualquer consideração ou análise acerca de Sistemas Operacionais de Propósitos Gerais.

Para o caso da arquitetura Superloop, ou arquitetura *Foreground/Background*, foi considerada a abordagem mais comum de programação, onde no nível de *Foreground* são incluídas as tarefas de maior prioridade da aplicação e o nível de *Background* é divido em *slots* de tempo iguais, sendo que as tarefas de menor prioridade do sistema, ou seja, as que não foram alocadas no nível de *Foreground* são distribuídas entre esses *slots*. Nesse caso, as tarefas de *Foreground* são ativadas por interrupções do sistema e as tarefas de *Background* são ativadas periodicamente de acordo com a execução do *slot* de tempo em que foi alocada.

Feito os estudos necessários sobre as arquiteturas de hardware e software, seguirá a análise conjunta das características, vantagens e desvantagens de cada arquitetura de software em relação a cada arquitetura de hardware. Tendo por base os dados presentes em Borges (2011b) e *UBM Electronics* (2012) que mostram a preferência em se utilizar SO's em arquiteturas de 32 bits e de se utilizar o *Superloop* em arquiteturas de 8 bits, será identificado quais são os pontos positivos e negativos de cada uma das duas associações.

Essa análise será dividida em quatro tópicos distintos relacionados à exigência e ao desempenho do Sistema Embarcado: consumo de memória, exigência de processamento, gerenciamento de multitarefas e atendimento do *deadline* das tarefas. Ponderando esses quatro pontos, será, então, discutido se é viável ou não a utilização de SO's em Sistemas Embarcados que utilizem microcontroladores de 8 bits ou o que poderia tornar esse uso viável.

A terceira etapa será identificar o que motiva a baixa utilização de SO's pelo mercado de desenvolvimento de Sistemas Embarcados brasileiro, uma vez que, segundo Borges (2011b) essa arquitetura de software se restringe a apenas 28% dos projetos nacionais, contra 70% dos projetos desenvolvidos ao redor do mundo, segundo *UBM Electronics* (2012).

Para elaborar essa análise, serão ponderados fatores como a característica da formação do profissional que atua no desenvolvimento de sistemas embarcados no Brasil, com base nos resultados que serão obtidos na primeira etapa deste projeto. Essa informação é relevante uma vez que, segundo os dados mostrados em Borges (2011a), os profissionais que trabalham com desenvolvimento de Sistemas Embarcados no Brasil são jovens e em sua maioria tem menos de cinco anos de experiência.

Também será considerada para essa terceira etapa as características das arquiteturas de hardware e software mais comuns em projetos de Sistemas Embarcados, de acordo com o estudo que será realizado na segunda etapa deste projeto.

O ponto principal que será a base para essa terceira etapa é o estudo da pesquisa *Embedded Market Survey*, presente em UBM Electronics (2012), realizado anualmente desde 2008 e do artigo *Embedded System Design: An Overview Brazilian Development*, presente em Borges (2011b), que fazem uma análise do mercado de desenvolvimento de Sistemas Embarcados global e nacional, respectivamente.

CAPÍTULO 4: RESULTADOS E DISCUSSÕES

Esse capítulo apresenta e discute os principais resultados do estudo desenvolvido durante esse trabalho, divididos entre análise do estudo de Sistemas Embarcados nas Universidades brasileiras e da visão dos alunos sobre o mercado de desenvolvimento, estudo de viabilidade de utilização de SO's em Sistemas Embarcados de pequeno porte e análise do mercado nacional de desenvolvimento de Sistemas Embarcados.

4.1.Estudo de Sistemas Embarcados nas Universidades Brasileiras

Esta sessão apresenta e discute os resultados da pesquisa sobre o Estudo de Sistemas Embarcados em Universidades Brasileiras.

A pesquisa foi realizada entre os dias 10 de setembro de 2012 e 22 de outubro de 2012. A pesquisa, disponibilizada de maneira eletrônica, atingiu um total de 182 respostas, de 15 universidades diferentes.

Alguns dados foram análise de duas maneiras distintas. A primeira considera a totalidade das respostas e a segunda considera apenas as respostas dos alunos da USP (Universidade de São Paulo). O objetivo desta distinção é fazer uma comparação entre o estudo de Sistemas Embarcados na USP e em todas as universidades no geral.

4.1.1 Perfil dos Alunos e Características da Abordagem do Estudo de Sistemas Embarcados nas Universidades Brasileiras

Dentre os entrevistados, 92% eram alunos de cursos de graduação, 5% alunos de mestrado e 3% alunos de doutorado. As perguntas eram direcionadas aos cursos de graduação, inclusive aos alunos pertencentes à pós-graduação. Para efeito de análise, os 8% de alunos pós-graduandos são classificados como alunos formados.

A pesquisa se destinava a cursos ligados à área de Sistemas Embarcados. A participação maciça foi de alunos dos cursos de Engenharia de Computação - 42,31%, Ciência da Computação - 28,02% - e Engenharia Elétrica/Eletrônica - 26,92%. Esses números são representativos para o escopo da pesquisa uma vez que, segundo Borges (2011b), 84% dos desenvolvedores envolvidos com projetos de sistemas embarcados no mercado brasileiro são formados em um desses três cursos.

O mercado brasileiro de desenvolvimento de Sistemas Embarcados é dominado por engenheiros, sendo que apenas 5% são bacharéis em computação. Esse fato deve-se provavelmente a maior familiaridade dos engenheiros eletricistas e engenheiros de computação com o hardware intrinsecamente ligado ao software embarcado e em muitas vezes desenvolvido em conjunto. (BORGES 2011a)

A Tabela 1 abaixo apresenta a divisão dos participantes por curso considerando todos os alunos e apenas os alunos da USP.

Tabela 1 - Divisão das respostas por curso de graduação

Curso	Participantes	Participantes - USP
Engenharia de Computação	77	53
Engenharia Elétrica/Eletrônica	49	27
Ciências da Computação	51	16
Engenharia Mecatrônica/Mecânica	3	1
Informática	2	2
Total	**182**	**99**

A pesquisa atingiu participantes de todos os períodos acadêmicos, com destaque, obviamente, para os períodos correspondentes ao segundo semestre letivo, devido aos meses em que a pesquisa foi vinculada - Setembro e Outubro. A pesquisa também contou com grande participação de alunos já formados - 22,5% - e de alunos do último ano letivo de seus respectivos cursos, o que permite ter uma visão completa da abordagem em relação a Sistemas Embarcados que as universidades costumam adotar para esses cursos.

O gráfico da Figura 3 mostra a distribuição percentual dos alunos participantes da pesquisa por período acadêmico.

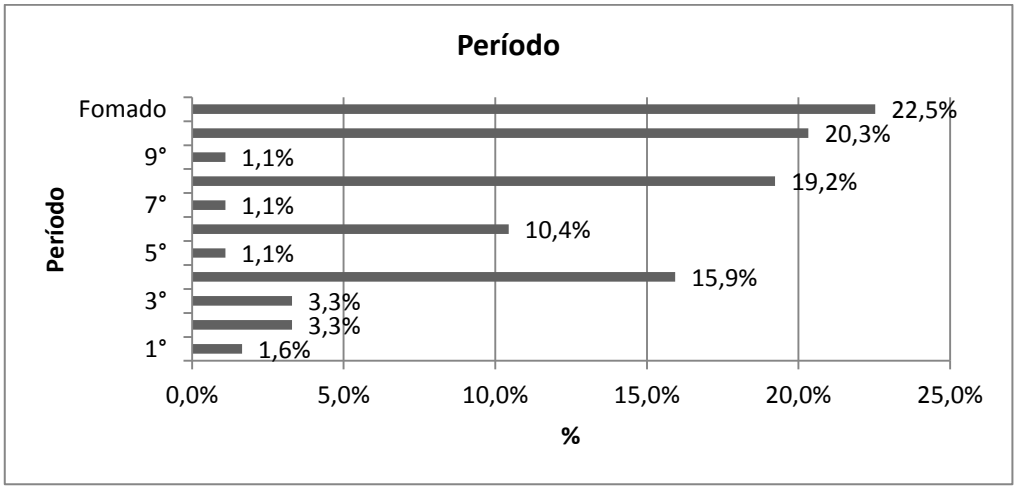

Figura 3 - Distribuição Percentual das Respostas x Período Acadêmico

Entre os entrevistados, 79,7% disseram ter tido contato com Sistemas Embarcados na Universidade. São considerados, nesse percentual, alunos que cursaram alguma disciplina relacionada com Sistemas Embarcados, seja ela obrigatória ou optativa, ou alunos que procuraram estudar o assunto por meio de atividades extracurriculares, como programas de Iniciação Científica ou grupos de estudo.

A análise da distribuição desses alunos por período acadêmico permite identificar em que momento as disciplinas relacionadas a Sistemas Embarcados costumam ser abordadas nas universidades brasileiras.

O gráfico da Figura 4 mostra a variação da taxa de alunos que tiveram contato com Sistemas Embarcados por ano letivo. Considerando que 100% dos alunos do 1° ano de graduação que já tiveram contato com Sistemas Embarcados na universidade disseram que esse contato foi devido a alguma atividade extracurricular, como Iniciação Científica ou participação em grupo de estudos e esse número entre os alunos do 2° ano é de 20%, o gráfico permite concluir que o assunto é normalmente abordado a partir do 3° ano de graduação. Nele verificamos que, do 3° ano em diante, mais de 70% dos alunos já tiveram algum contato com Sistema Embarcado.

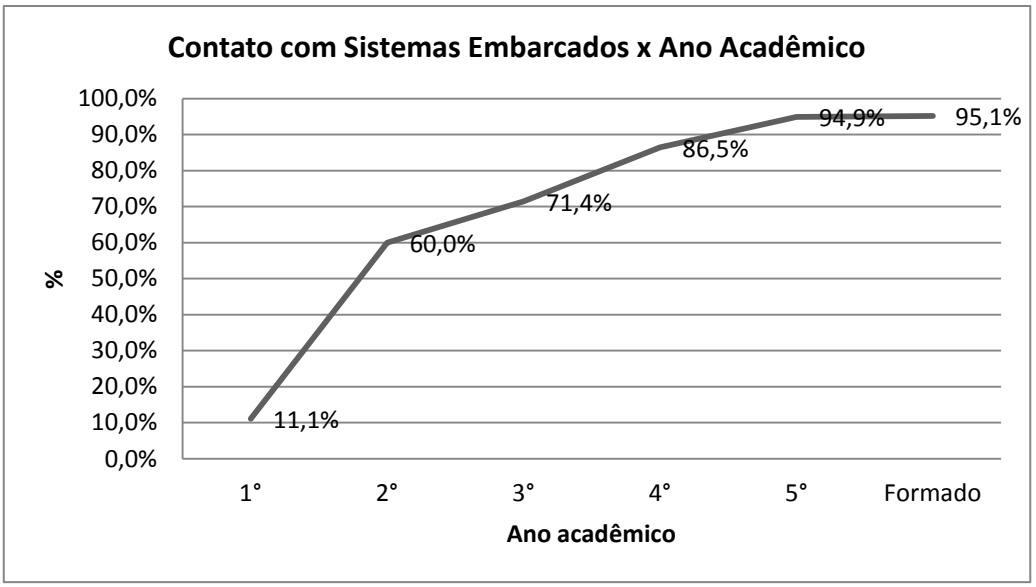

Figura 4 - Porcentagem de alunos que tiveram contato com Sistemas Embarcados x Ano acadêmico

A curva da Figura 5, considerando apenas alunos da USP, é semelhante a da Figura 4. Novamente verificamos que o assunto na USP, assim como a média geral, é abordado no a partir do 3° ano letivo. Apesar de mais da metade dos alunos do 2° ano afirmarem já terem tido contato

com Sistemas Embarcados na USP, 30% desse total é em decorrência de atividades extracurriculares.

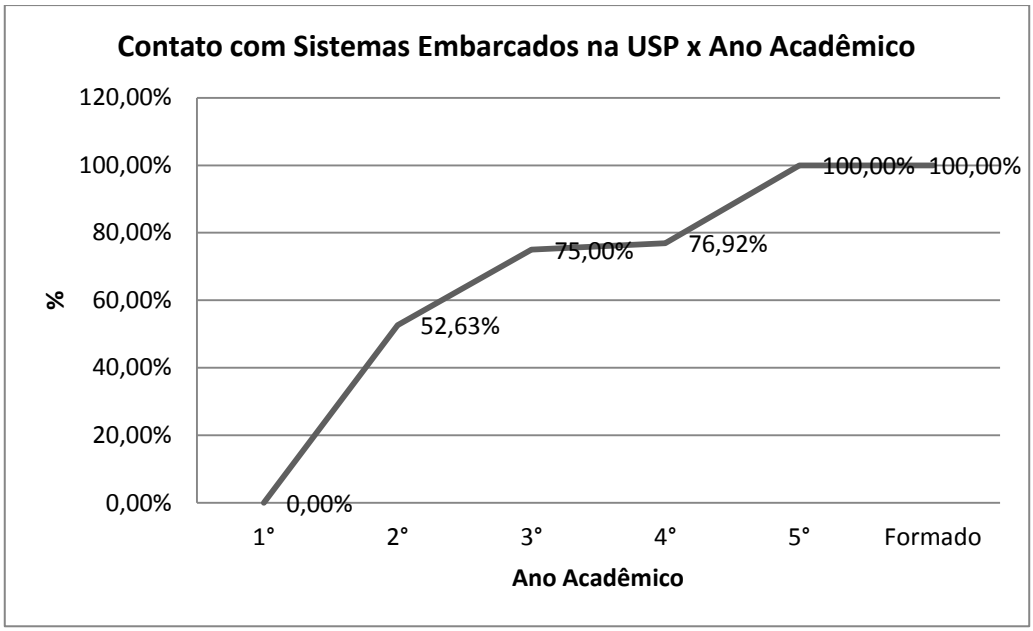

Figura 5 - Porcentagem de alunos da USP que tiveram contato com Sistemas Embarcados x Ano Acadêmico

Ao analisarmos a distribuição das disciplinas obrigatórias relacionadas a Sistemas Embarcados por período, a curva segue uma distribuição praticamente idêntica da Figura 4. Da mesma forma que esse contato é normalmente abordado no 3° ano, o número de alunos que cumprem matérias obrigatórias relacionadas a Sistemas Embarcados também cresce a partir do 3° ano.

Essa distribuição é consequência do fato das universidades destinarem os primeiros anos de graduação às disciplinas ditas de formação básica, principalmente considerando os cursos de engenharia. Sistema Embarcado é uma área de atuação dentro da área de computação/eletrônica, o que necessita um entendimento um pouco mais elaborado dos alunos antes de serem introduzidos ao assunto. Noções de arquitetura de computadores, lógica digital e programação de computadores são essenciais para que o entendimento do que é um Sistema Embarcado não seja raso por parte dos alunos.

A Figura 6 mostra essa distribuição entre todas as respostas e a Figura 7, somente entre os alunos da USP.

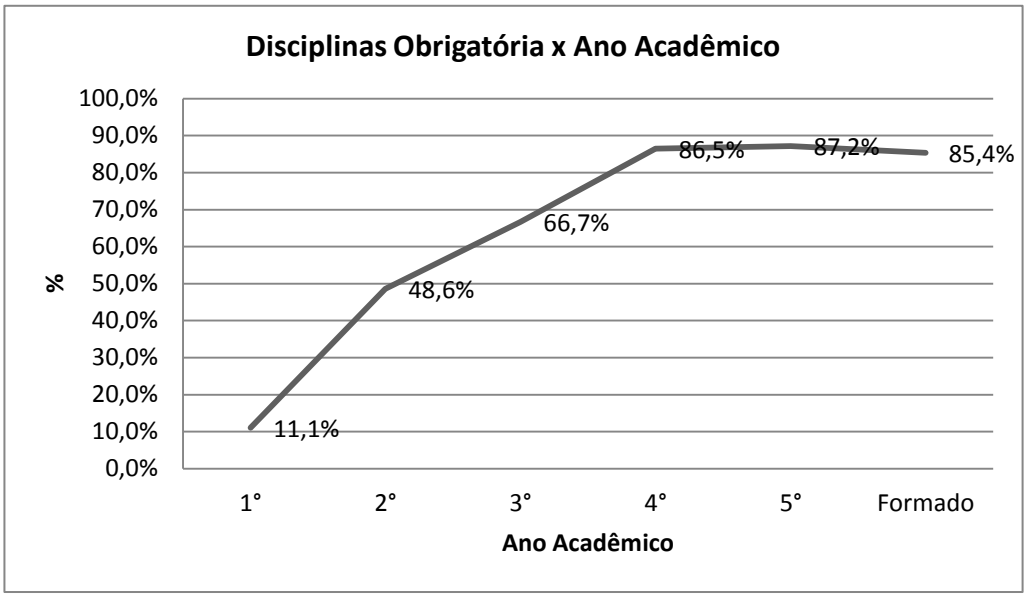

Figura 6 - Porcentagem de alunos que cursaram disciplinas relacionadas a Sistemas Embarcados x Ano Acadêmico

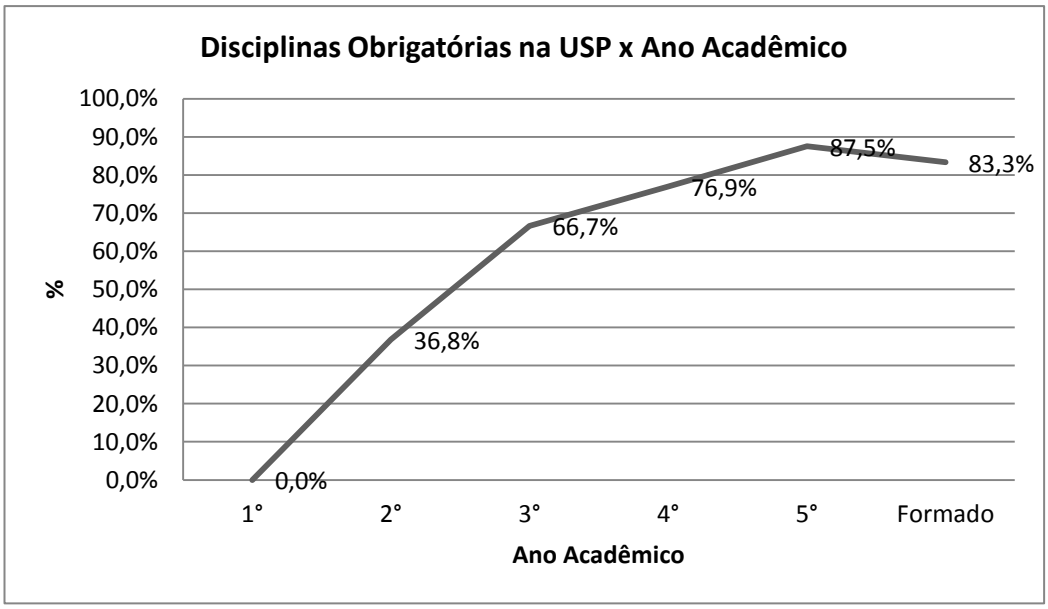

Figura 7 - Porcentagem de alunos da USP que cursaram disciplinas relacionadas a Sistemas Embarcados x Ano Acadêmico

A média de disciplinas obrigatórias relacionadas a Sistemas Embarcados cursadas, considerando alunos formados e no último período é de 1,7 disciplinas. Entre os formados essa média é de 2,0 disciplinas. Uma vez que os cursos abordados tem um caráter generalista a principio, com a política de formar uma grade curricular que apresente todos os ramos de atuação

pertencentes ao escopo do curso, essa média próxima de duas disciplinas parece ser considerada pelas universidades o suficiente para dar aos alunos um entendimento geral desse assunto.

Aos alunos que têm interesse em se especializar em Sistemas Embarcados ainda durante o curso de graduação, existe a possibilidade de cursar matérias optativas relacionadas ao tema. Essas matérias costumam aparecer na grade curricular após as matérias obrigatórias, como forma de complementá-las.

Dentre os alunos que já tiveram contato com Sistemas Embarcados, 37% cursam ou cursaram, além das matérias obrigatórias, alguma matéria optativa relacionada a Sistemas Embarcados. Desse percentual, 89,5% estão no penúltimo ou último ano de gradação.

A Figura 8 trás a porcentagem de alunos que cursaram ou cursam disciplinas optativas relacionadas a Sistemas Embarcados depois de terem contato com o tema anteriormente.

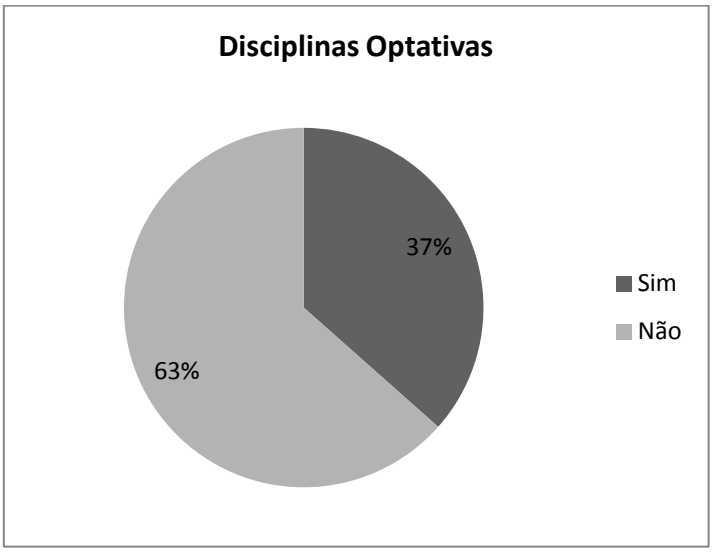

Figura 8 - Porcentagem de alunos que cursam matérias optativas relacionadas a Sistemas Embarcados após o primeiro contato

A média de disciplinas optativas cursadas, considerando alunos formados e alunos no último ano, é de 2,1 disciplinas. Considerando apenas os alunos formados, essa média é de 2,2 disciplinas.

35% dos entrevistados tiveram contato com Sistemas Embarcados em alguma atividade extracurricular, como uma Iniciação Científica ou participação em grupos de estudo. Esse número, considerando alunos do último ano de seus respectivos cursos e alunos já formados é de 41%.

A pesquisa também a ponta que 31,3% dos alunos pretendem se especializar ou trabalhar com Sistemas Embarcados, como mostrado na Figura 9. Ao considerarmos apenas os alunos da USP, esse número sobe a 34,3%.

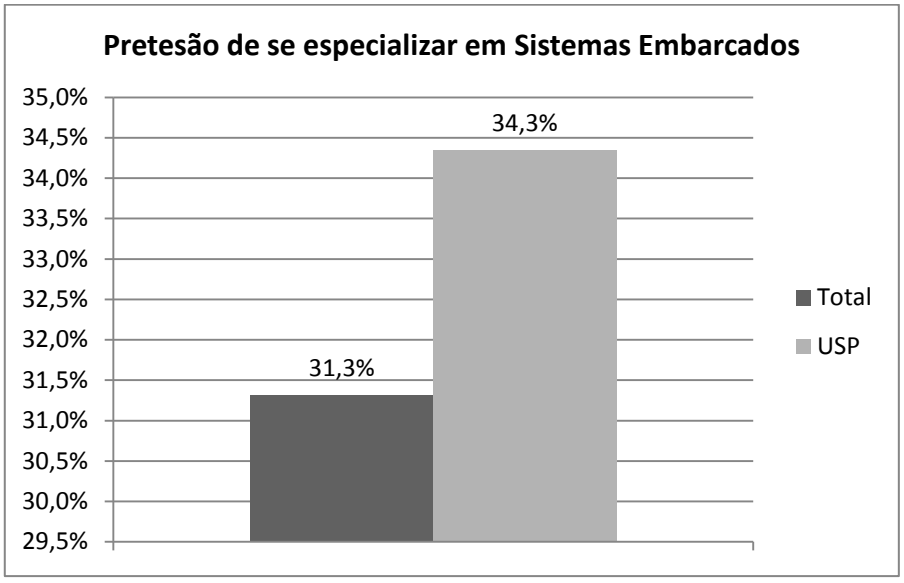

Figura 9 - Porcentagem de alunos que pretendem se especializar ou trabalhar com Sistemas Embarcados.

Ao fazermos a mesma análise antes e depois do primeiro contato dos alunos com Sistemas Embarcados, vemos que o número de alunos que responderam afirmativamente à pergunta aumenta de 16,2% para 35,2% no total geral dos alunos e de 14,3% para 39,7% entre os alunos da USP, como mostrado nas Figuras 10 e 11.

Esses números são condizentes com os apresentados anteriormente sobre a procura pelo contato na universidade além das disciplinas obrigatórias, mostrando que de 35% a 40% dos alunos dos cursos analisados seguirão no mercado de Sistemas Embarcados. Podemos, então, verificar que esta é uma área que chama a atenção dos alunos dentro da universidade. Esse aumento da pretensão devido ao contato na universidade é reflexo da abrangência do tema. Antes do contato com o tema muitas vezes o aluno desconhece essa abrangência sendo que, apesar de saber o nicho da engenharia ou da computação que têm pretensão de se especializar ou trabalhar, não tenha o conhecimento de que pertença ao escopo de Sistemas Embarcados.

Essa alta porcentagem é mais um motivo para que o tipo de formação oferecida pelas universidades esteja alinhado com as expectativas do mercado de desenvolvimento de Sistemas Embarcados.

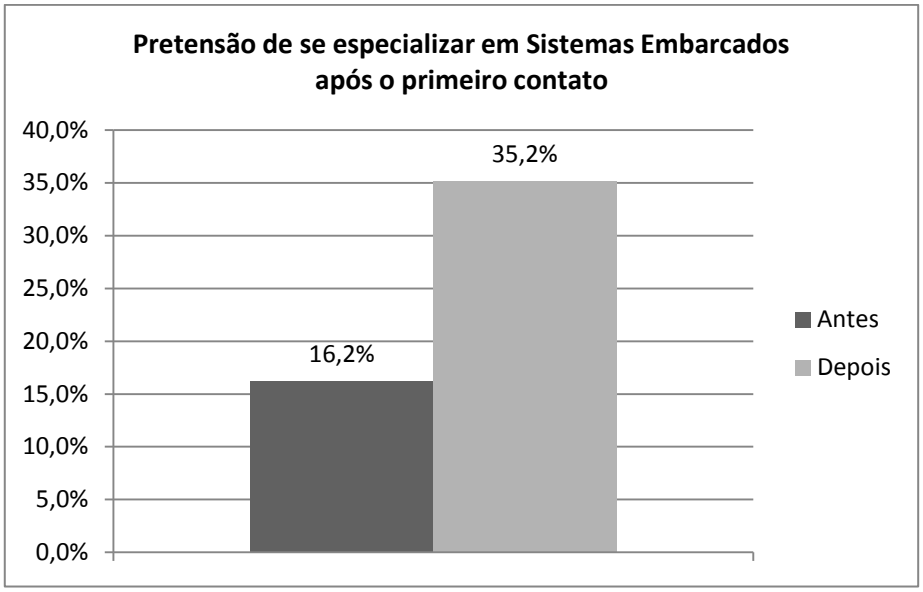

Figura 10 - Porcentagem de alunos que pretendem se especializar ou trabalhar com Sistemas Embarcados antes e depois do primeiro contato com Sistemas Embarcados.

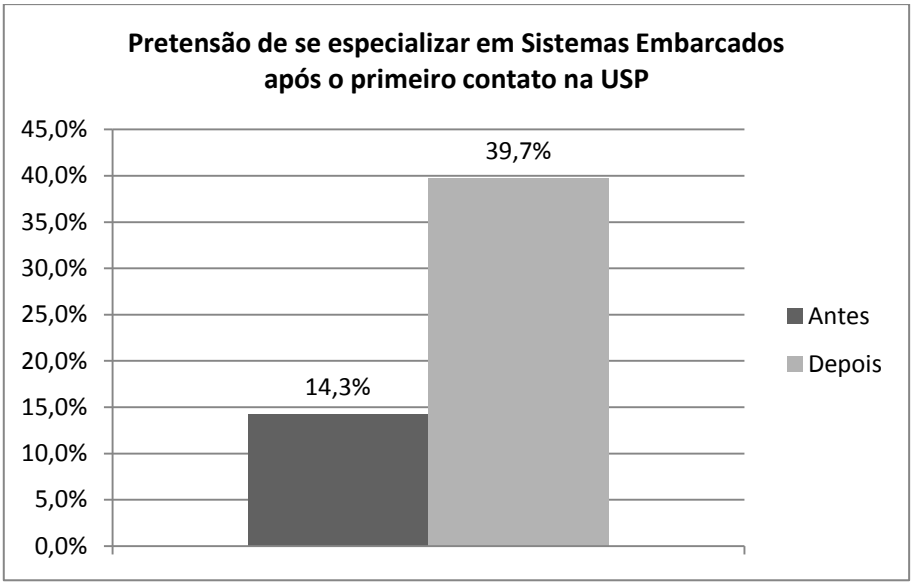

Figura 11 - Porcentagem de alunos da USP que pretendem se especializar ou trabalhar com Sistemas Embarcados antes e depois do primeiro contato com Sistemas Embarcados.

Outro motivo diz respeito à experiência dos desenvolvedores no mercado de desenvolvimento de embarcados no Brasil. Segundo Borges (2011b), 70,7% dos desenvolvedores têm menos de 30 anos de idade e 91,0% deles tem menos de 10 anos de experiência no desenvolvimento de Sistemas Embarcados, sendo que, 50,7% tem menos de cinco anos de experiência.

Esses dados mostram que o mercado de desenvolvimento brasileiro é composto por pessoas jovens, que iniciam o trabalho logo após se formarem, levando consigo toda a bagagem recém-adquirida na universidade.

A título de comparação, segundo *CMP United Business Media* (2006), o mercado de desenvolvimento mundial é formado majoritariamente por desenvolvedores entre 35 e 50 anos de idade – 65% – e por desenvolvedores com mais de 10 anos de experiência – 63%.

Analisando a pretensão de se especializar ou trabalhar com Sistemas Embarcados de acordo com o curso de graduação, vemos uma preferência maior dos alunos do curso de Engenharia de Computação em seguir nesse mercado. A característica de Sistemas Embarcados, que cada vez mais primam pela integração entre hardware e software o torna atrativo para engenheiros de computação.

A Tabela 2 mostra a distribuição dessa pretensão por curso de graduação.

Tabela 2 - Número de alunos que pretendem se especializar/trabalhar com Sistemas Embarcados de acordo com o curso de graduação

Curso	Total de alunos	Pretendem se especializar	%
Engenharia de Computação	77	37	48,05
Engenharia Elétrica/Eletrônica	49	12	24,49
Ciências da Computação	51	8	15,69

Essa distribuição é praticamente a mesma ao olharmos apenas para os alunos da USP. Novamente vemos uma pretensão maior dos alunos de Engenharia de Computação em se incluir nesse nicho de atuação. A Tabela 3 trás essa distribuição dentro da USP.

Tabela 3 - Número de alunos da USP que pretendem se especializar/trabalhar com Sistemas Embarcados de acordo com o curso de graduação

Curso	Total de alunos	Pretendem se especializar	%
Engenharia de Computação	53	24	45,28
Engenharia Elétrica/Eletrônica	27	8	29,63
Ciências da Computação	16	2	12,50

Com relação à arquitetura de microcontroladores estudada, ao analisarmos o quadro geral de alunos, vemos que 55,9% tiveram contato com a arquitetura de 8-bits. Esse contato, ao consideramos apenas os alunos formados e de último ano é praticamente o mesmo – 56,5%. A Tabela 4 mostra a distribuição do conhecimento entre diferentes arquiteturas.

Tabela 4 - Porcentagem de alunos com conhecimento em diferentes arquiteturas

Arquitetura	Total de alunos (%)	Formados e Formandos (%)
4 bits	12,5	12,5
8 bits	55,9	56,5
16 bits	45,4	48,2
32 bits	41,4	43,4
64 bits	14,5	13,2
FPGA, SOC, etc.	54,6	65,1

Isso mostra a função didática dos microprocessadores de 8 bits, frequentemente usados para introduzir o aprendizado de desenvolvimento de hardware. Sua arquitetura simplificada é ideal para o entendimento de todos os módulos que compõe um microprocessador ou um microcontrolador, e permite um raciocínio mais claro com relação a conjunto de instruções, endereçamento de memória, banco de registradores, etc.

Na USP o aprendizado de microcontroladores a partir da arquitetura de 8 bits é ainda maior. 70,3% dos alunos da USP tiveram contato com essa arquitetura em alguma das disciplinas que cursaram. Esse número que cresce entre os alunos formados e de último ano – 72,5%. A Tabela 5 mostra quais arquiteturas são mais abordadas na USP.

Tabela 5 - Porcentagem de alunos da USP com conhecimento em diferentes arquiteturas

Arquitetura	Total de alunos (%)	Formados e Formandos (%)
4 bits	13,5	13,7
8 bits	70,3	72,5
16 bits	54,0	56,9
32 bits	35,1	37,2
64 bits	8,1	9,8
FPGA, SOC, etc.	56,8	62,7

Outra vantagem de se utilizar um microprocessador de 8 bits como forma de aprendizado é poder explorar a linguagem *Assembly*. 73,2% dos alunos tiveram contato com esse tipo de linguagem para programação de microcontroladores.

Assim como as arquiteturas de 8 bits são usadas para iniciar os estudos de arquiteturas de microcontroladores, a linguagem *Assembly* é a primeira linguagem de programação de microcontroladores normalmente abordada. Sua função, assim como a arquitetura de 8 bits é colocar o aluno mais próximo das peculiaridades do hardware. A porcentagem de alunos formados e no último ano que tiveram contato com *Assembly* é de 81,4% no geral e 83,0% na USP. As Tabelas 6 e 7 mostram o conhecimento dos alunos dividido entre diferentes linguagens de programação de Sistemas Embarcados.

Tabela 6 - Linguagens de Programação para Sistemas Embarcados abordas nas Universidades

Linguagem	Total de alunos (%)	Formados e Formandos (%)
Assembly	73,2	81,4
C	55,6	68,6
C++	13,4	16,3
Java	12,0	12,8
VHDL	6,3	5,8

Tabela 7 - Linguagens de Programação para Sistemas Embarcados abordas na USP

Linguagem	Total de alunos (%)	Formados e Formandos (%)
Assembly	72,4	83,0
C	61,8	68,0
C++	11,8	11,3
Java	6,6	7,5
VHDL	9,2	11,3

Arquiteturas como FPGA (*Field-Programmable Gate Array*) também merece destaque, por serem de conhecimento de 54,6% dos alunos. Esse número é de 65,0% ao olharmos apenas para os alunos formados ou no último ano. Na USP observa-se também um crescimento do percentual de alunos que tiveram contato com FPGA ao final do curso. Ao analisarmos seus alunos, vemos que 62,7% dos formados e formandos tem conhecimento dessa arquitetura, contra 56,8% do geral.

Uma vez que a programação de microcontroladores com arquiteturas mais complexas, como 16, 32, 64 bits e FPGA praticamente impossibilita o desenvolvimento do software em *Assembly*, 55,6% afirmam terem tido contato com a linguagem C para Sistemas Embarcados. Na USP esse número á ainda maior, atingindo 61,8% dos alunos. Entre os alunos formados e em via de se formar, essa porcentagem é de 68,6% no geral e 68,0% na USP.

Essa formação sólida dos alunos em linguagem C para Sistemas Embarcados, considerando que cerca de 70% dos alunos saem da faculdade com esse conhecimento, é condizente com a tendência do mercado, tanto brasileiro quanto internacional, de usar linguagens de alto nível, em detrimento da linguagem *Assembly*. Segundo Borges (2011b), 83,3% dos novos projetos desenvolvidos no Brasil usam C como linguagem principal, sendo que *Assemby* está presente em apenas 1,5% dos projetos.

De acordo com UBM Electronics (2012) a porcentagem de projetos que se baseiam em C está acima de 60% desde 2008, atingindo 65% dos projetos esse ano. *Assembly* é usada globalmente em apenas 5% dos projetos, desde 2008.

A primeira lacuna que encontramos diz respeito ao estudo de Sistemas Operacionais para Sistemas Embarcados. Apenas 26,4% dos alunos que saíram ou estão saindo da universidade

adquiriram esse conhecimento nela, sendo que esse número é de 27,8% na USP, como mostrado na Figura 12.

Figura 12 - Porcentagem de alunos que saem da universidade com conhecimento de Sistemas Operacionais para Sistemas Embarcados.

Ao olharmos apenas para os alunos formados e formandos com pretensão de se especializar ou trabalhar com Sistemas Embarcados, apenas 34,8% deles tiveram contato com Sistemas Operacionais para Sistemas Embarcados na universidade.

Ao analisarmos o mercado de desenvolvimento de Sistemas Embarcados, de acordo com a pesquisa *Embedded Market Survey* da *UBM Electronic* (2012), a porcentagem de projetos que utilizam algum tipo de SO é praticamente constante de 2008 a 2012, girando em torno de 70% dos projetos. Contudo, o mercado brasileiro não segue essa tendência. No Brasil, segundo Borges (2011b), apenas 28% dos projetos de Sistemas Embarcados utilizam SO's.

O estudo desse tema nas universidades parece, então, condizente com a estratégia que o mercado de desenvolvimento de Sistemas Embarcados nacional adota, porém, assim como esse mercado, não se encaixa no modelo seguido pela maioria dos desenvolvedores pelo mundo.

Em todos os casos, a maioria dos estudantes que tiveram contato com SO's direcionados a Sistemas Embarcados são do curso de Engenharia de Computação. Cerca de 30%, tanto no geral quanto na USP, saem da universidade com esse conhecimento, contra um percentual de

aproximadamente 20% entre os alunos dos cursos de Engenharia Elétrica/Eletrônica e Ciências da Computação.

Isso mostra uma preocupação mais acentuada de integrar hardware e software no curso de Engenharia de Computação do que em outros. Contudo, a USP não oferece matérias obrigatórias que cubram esse assunto, ficando a cargo do aluno procurar o conhecimento em matérias optativas ou em atividades extracurriculares.

4.1.2. Análise dos Alunos sobre o Mercado de Desenvolvimento de Sistemas Embarcados

Considerando a alta taxa de alunos que pretendem se especializar ou trabalhar com Sistemas Embarcados, é interessante identificar qual a visão desses alunos sobre o mercado ao qual pretendem se incluir. Apesar de a universidade ser a principal fornecedora de novas tecnologias no mercado brasileiro, seus alunos costumam não ser muito familiarizados com suas peculiaridades. Isso atrasa a adaptação do aluno ao mercado de trabalho e dificulta a troca de novas experiências entre recém-formados e empregadores.

A primeira análise diz respeito às arquiteturas usadas em projetos de Sistemas Embarcados. A pesquisa *Embedded Market Survey* realizada pela *UBM Electronics* (2012) aponta que 63% dos novos projetos desenvolvidos no mercado mundial usam arquitetura de 32 bits. Esse número é superior a 60% desde 2009, mostrando que essa arquitetura é a dominante no mercado de desenvolvimento.

Os dados da pesquisa mostram que 36,2% dos alunos acreditam ser essa arquitetura a mais utilizada pelo mercado. Essa análise é assertiva se considerarmos o mercado mundial em geral, ao considerar a dominância da arquitetura de 32 bits, e ainda mais assertiva ao olharmos para o mercado de desenvolvimento brasileiro. Nele, a arquitetura de 32 bits foi usada em 34% dos novos produtos em 2011, segundo Borges (2011b).

A Figura 13 mostra qual a opinião dos alunos com relação às arquiteturas utilizadas no mercado e essa mesma distribuição entre os alunos que pretendem se especializar em Sistemas Embarcados. Apesar de apresentar uma porcentagem menor, os alunos que pretendem se especializar ou trabalhar com Sistemas Embarcados também considera a arquitetura de 32 bits como sendo a dominante no mercado.

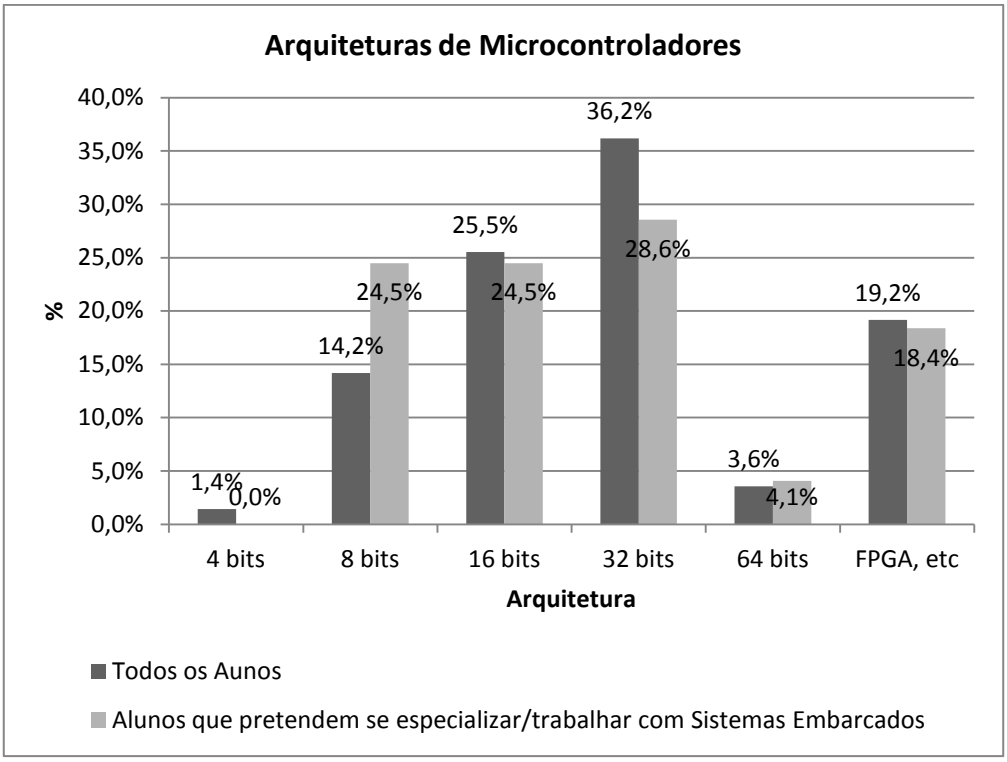

Figura 13 - Distribuição percentual da opinião dos alunos sobre as principais arquiteturas de microcontroladores utilizadas pelo mercado de desenvolvimento.

A principal diferença entre as duas análises está na arquitetura de 8 bits. A análise geral é de que essa arquitetura seja usada em pouco mais de 14% dos novos projetos desenvolvidos, enquanto que, entre quem pretende seguir adiante com Sistemas Embarcados, esse percentual é de quase 25%.

A visão global dos estudantes é de que arquiteturas mais complexas dominem o mercado, uma vez que esse parece cada vez mais desenvolvido e adepto a novas tecnologias.

O que vemos, porém, é que o mercado brasileiro, ao contrário do observado no mercado mundial, ainda utiliza maciçamente microcontroladores de 8 bits em seus projetos. No Brasil, de acordo com a pesquisa em Borges (2011b), 31% dos novos projetos são de 8 bits, contra apenas 13% em 2012 nos projetos ao redor do mundo, como mostrado em *UBM Electronics* (2012).

A explicação para essa diferença, segundo Borges (2011a), é simples. No mercado brasileiro, e de países emergentes em geral, a pressão por custo é maior, o que faz dos microcontroladores de 8 bits uma opção mais viável ao projeto do que uma arquitetura mais complexa e, consequentemente, mais cara.

Podemos somar a esse fator as características do estudo de microcontroladores nas universidades brasileiras, apresentado na Tabela 4. A alta porcentagem de alunos que se formam

com conhecimento da arquitetura de 8 bits é superior às outras arquiteturas. A inclusão desses alunos no mercado de desenvolvimento contribui para a manutenção desses microcontroladores no mercado nacional.

A Figura 14 faz a mesma consideração em relação aos alunos da USP. É interessante ressaltar que o gráfico referente aos alunos da USP que pretendem se especializar ou trabalhar com sistemas embarcados é extremamente condizente com o apresentado por Borges (2011b) sobre as arquiteturas utilizadas no desenvolvimento de Sistemas Embarcados no mercado brasileiro. Isso mostra que a leitura dos alunos da USP sobre o mercado é excelente nesse aspecto.

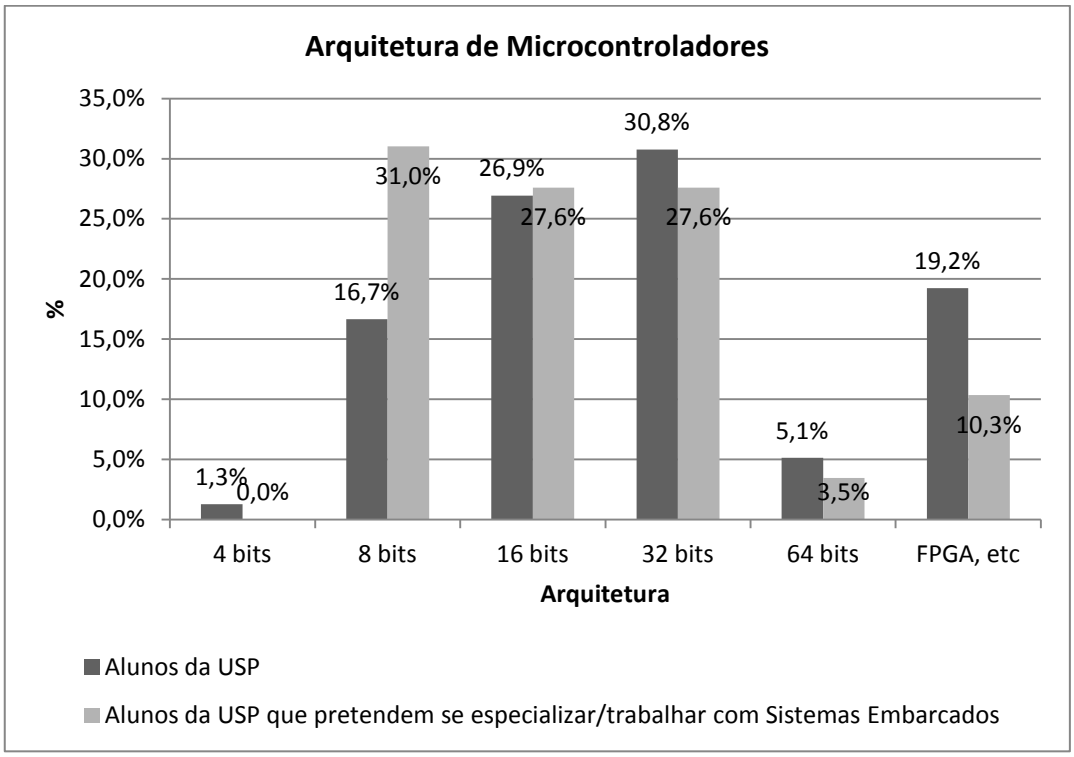

Figura 14 - Distribuição percentual da opinião dos alunos da USP sobre as principais arquiteturas de microcontroladores utilizadas pelo mercado de desenvolvimento.

Quando questionados sobre os principais desafios em um projeto de Sistema Embarcado, a maioria dos estudantes apontou Qualidade - 28,5% - e Inovação - 24,5% - como principais desafios. No mercado brasileiro, esses fatores são, respectivamente, o 4°, em 15,9% dos projetos, e 3°, em 17,4% dos projetos, mais desafiadores, sendo suprimidos por Prazo/*Time-to-market*, majoritário em 31,9% dos casos e por Custo, em 23,2% dos casos. A Figura 15 mostra os dados referentes a todas as respostas e as respostas de alunos que querem se especializar ou trabalhar com Sistemas Embarcados.

Os dois gráficos são parecidos e mostra uma análise igual dos alunos sobre esse assunto, independente da pretensão ou não de focar a carreira em Sistemas Embarcados. Essa análise é fruto de uma visão ainda de consumidor por parte dos alunos, que enxergam fatores como qualidade e novas funcionalidades na hora de escolher um produto.

Exemplo disso é que, para os alunos, a importância do Prazo/*Time-to-market* é pouca, sendo apontada como principal desafio por apenas 10,6% dos participantes. O custo, por sua vez, teve um resultado mais elevado e mais condizente com o mercado.

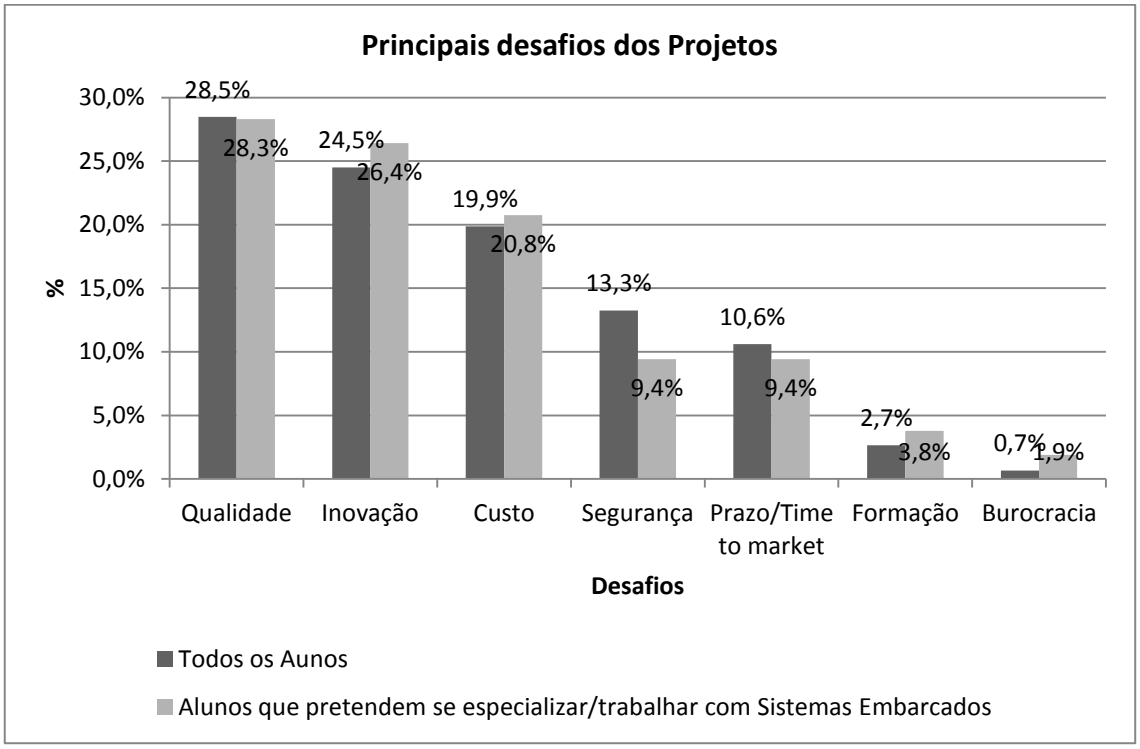

Figura 15 - Distribuição percentual da opinião dos alunos sobre os principais desafios do mercado de desenvolvimento.

Entre os alunos da USP a preferência por Qualidade é ainda mais acentuada, chegando a 35,6% das respostas. A Figura mostra os números referentes aos alunos da USP e, por ela, podemos tirar as mesmas conclusões citadas anteriormente.

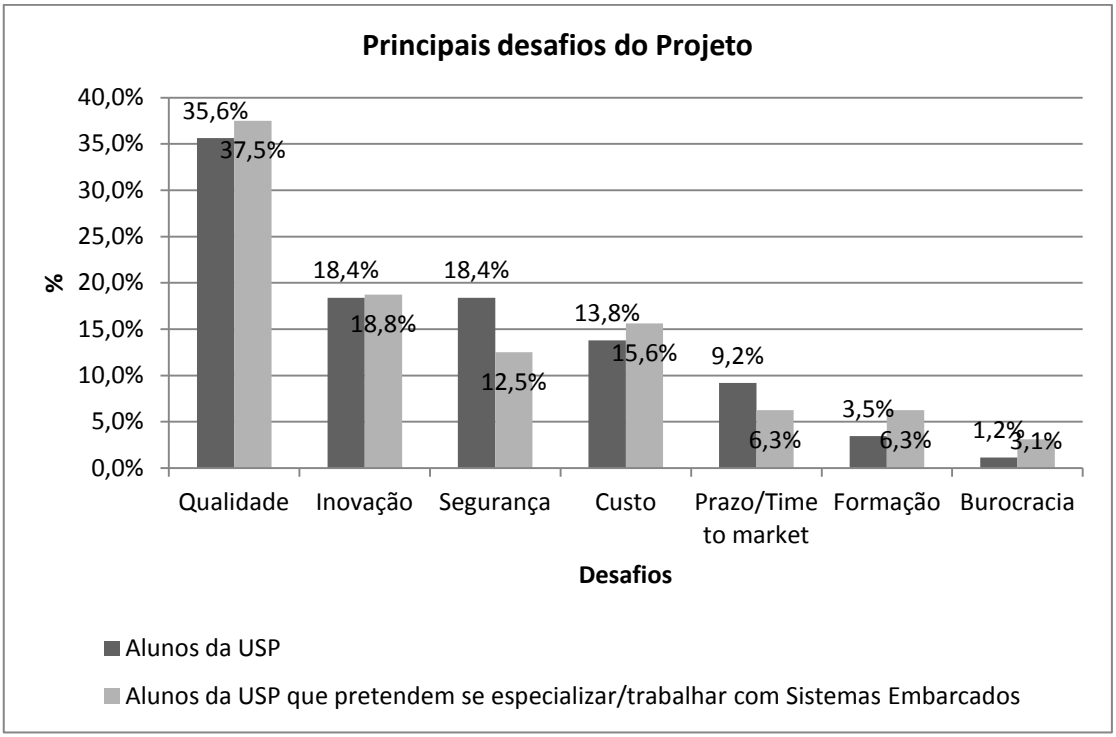

Figura 16 - Distribuição percentual da opinião dos alunos da USP sobre os principais desafios do mercado de desenvolvimento.

Quando questionados sobre os principais fatores que permeiam a escolha do microcontrolador a ser usado em um projeto, os desenvolvedores brasileiros apontaram o custo em 69,7% dos casos, Periféricos, em 45,5% dos casos, Desempenho, em 43,9% e Ferramentas Disponíveis, em 40,9%. (BORGES 2011a)

Existe, nessa preferência do mercado nacional, novamente uma diferenciação com relação ao mercado internacional. De acordo com a *UBM Electronics* (2012), o mercado global prima por Ferramentas de Software Disponíveis em 71% dos casos desde o início da pesquisa em 2008, sendo o Desempenho vindo em segundo lugar em cerca de 50% dos projetos e o Custo em terceiro, em cerca de 40% dos casos.

Novamente a análise dos alunos se mostrou condizente com o mercado nacional, sendo que o custo foi apontado pelos alunos como fator de exclusão em 64,5% das respostas, seguido por Desempenho em 59,4% delas. A Figura 17 faz o comparativo entre os fatores de seleção apontados pelos alunos e os mesmos apontados na pesquisa feita no mercado de desenvolvimento nacional por Borges (2011b).

A principal diferença vista no gráfico comparativo está presente na avaliação do Consumo de Energia como fator determinante na escolha do microcontrolador. O mercado brasileiro da

atenção e este fator em apenas 12% dos casos, contra uma previsão de 50% dos alunos. Segundo a *UBM Electronics* (2012), este ano o consumo foi relevante em 22% dos projetos ao redor do mundo, contra 25% nos anos anteriores.

Em ambos os mercados, brasileiro e internacional, a baixa preocupação com Consumo de Energia, em relação a outros fatores, é consequência da evolução que os microcontroladores tiveram ao longo dos anos e a preocupação que os fornecedores de microcontroladores sempre tiveram de oferecer produtos cada vez mais econômicos. Essa preocupação é ainda menor no mercado brasileiro, que utiliza arquiteturas mais simples que o mercado internacional em geral.

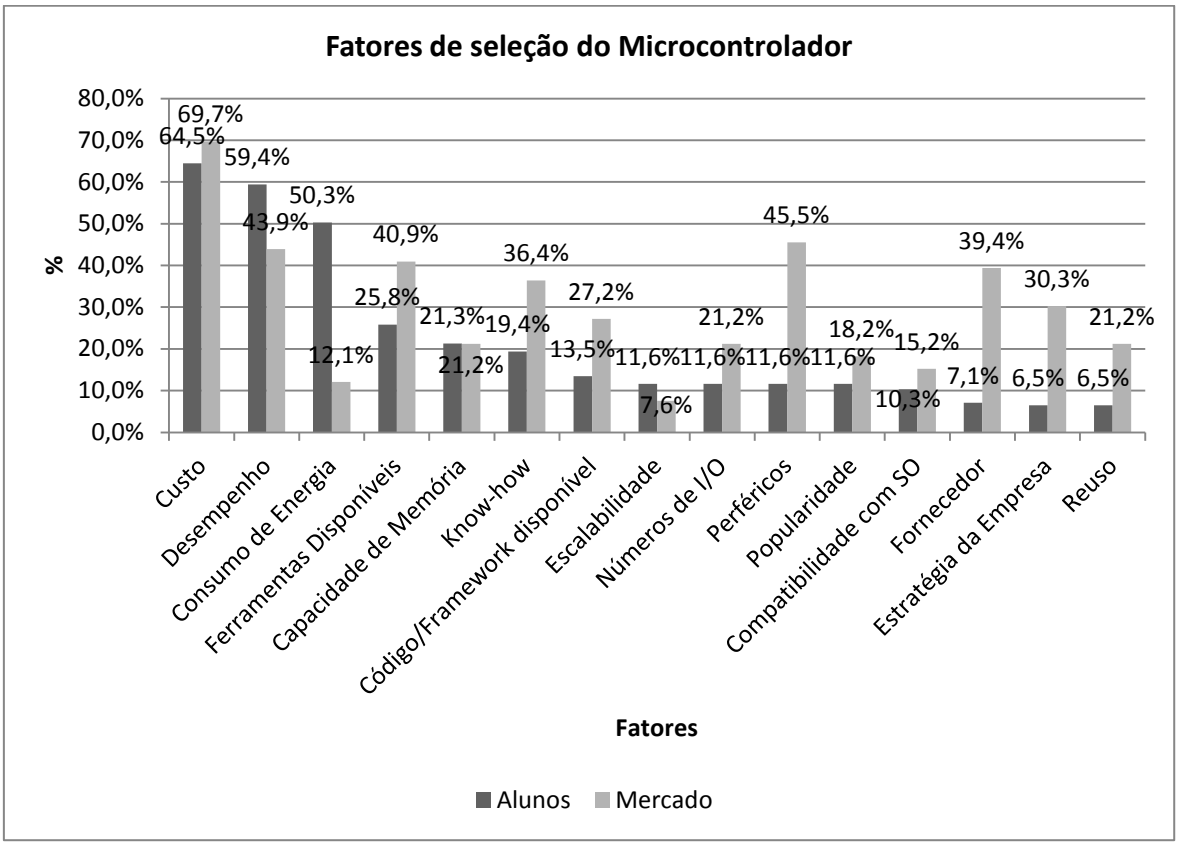

Figura 17 - Opinião dos alunos versus opinião dos desenvolvedores do mercado brasileiro sobre os principais fatores de seleção do microcontrolador.

Os alunos acreditam que, no mercado profissional de desenvolvimento de Sistemas Embarcados, 97,9% dos projetos reusam software. Esse valor é superestimado, devido provavelmente as vantagens constantemente apresentadas na universidade com relação ao reuso de software. Essa superestimação é ainda maior entre os que pretendem se especializar. Nenhum aluno deste grupo apontou que não houvesse reuso de software nos projetos de Sistemas Embarcados.

A Figura 18 mostra o percentual de reuso de software estimado pelos alunos.

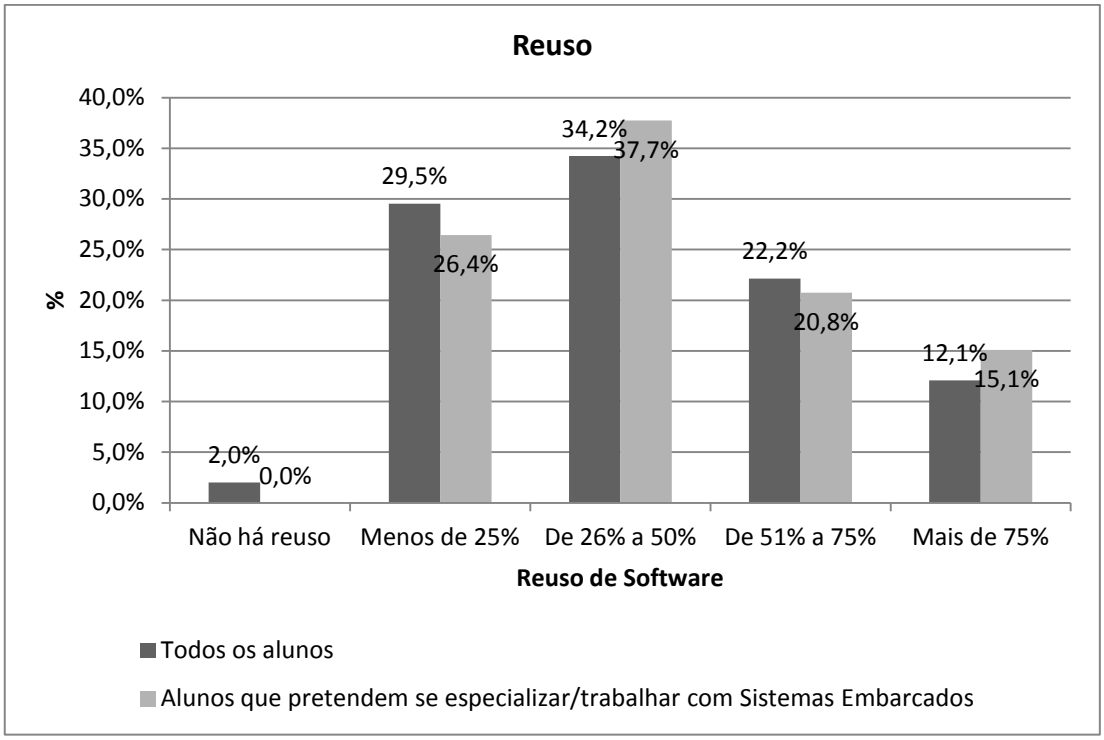

Figura 18 - Distribuição percentual da opinião dos alunos sobre o reuso de software nos projetos desenvolvidos no mercado.

Apesar de supervalorizada, essa perspectiva é válida. No mercado brasileiro, 86% dos projetos reusaram software no ano passado, segundo Borges (2011b), e a taxa mundial, que era de 89% em 2008, vem caindo gradativamente desde então até os 85% apontados esse ano, segundo *UBM Electronics* (2012).

Apesar de alta a porcentagem de projetos que reusam software, a pesquisa realizada por Borges (2011b) mostra que apenas 30% dos projetos reusam mais de 50% de código. O reuso de software acelera o processo de desenvolvimento e diminui problemas com teste e qualidade do software, uma vez que o código reutilizado já foi previamente testado, otimizado e aprovado.

Segundo a *UBM Electronics* (2012), desde 2009, 62% dos projetos de Sistemas Embarcados ao redor do mundo gastam mais recursos com o desenvolvimento do software do que com o desenvolvimento do hardware. Esse número reforça ainda mais a vantagem de reutilizar software. Além de acelerar o desenvolvimento, como já dito, pode ser fator determinante no custo final total do projeto.

A análise dos alunos da USP, como mostrado na Figura 19, é condizente com a análise geral dos alunos entrevistados.

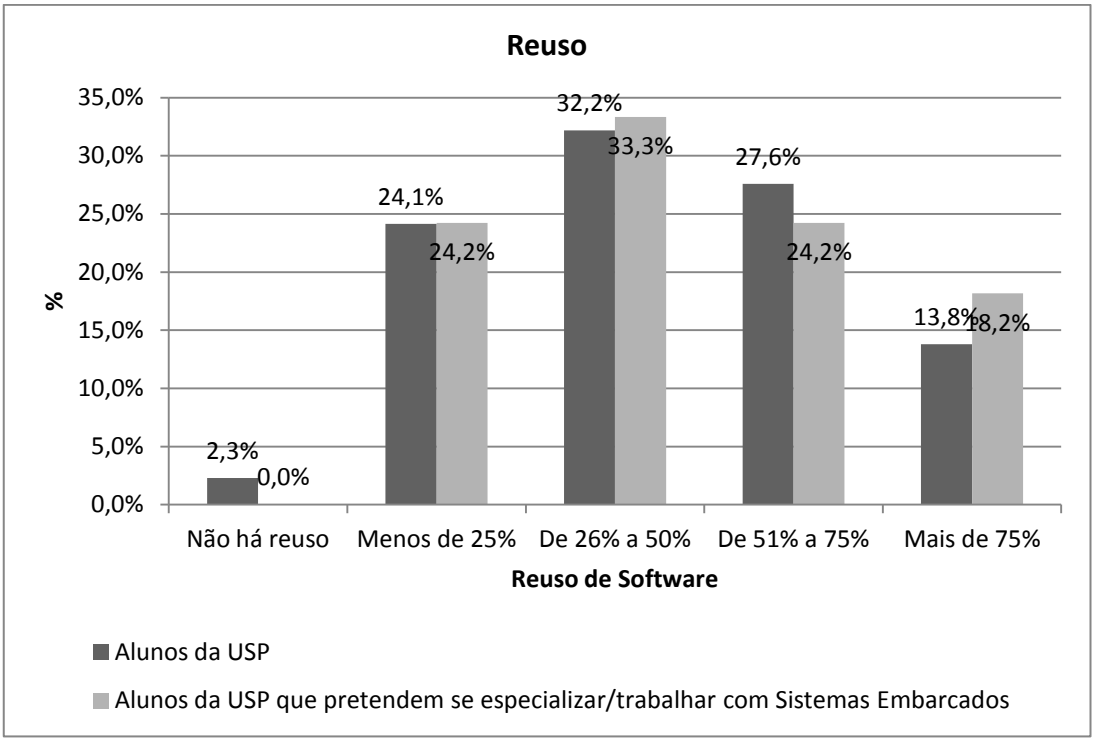

Figura 19 - Distribuição percentual da opinião dos alunos da USP sobre o reuso de software nos projetos desenvolvidos no mercado.

32% dos estudantes acreditam que o uso de SO's em projetos de Sistemas Embarcados está na faixa entre 21% e 40%. Esse número é de 41,5% entre os alunos que pretendem se especializar ou trabalhar com Sistemas Embarcados, como mostra a Figura 20.

Essa análise é correta ao considerarmos apenas mercado de desenvolvimento brasileiro. Segundo Borges (2011b), no país, 28% dos projetos de Sistemas Embarcados fazem uso e algum SO. Contudo, ela não se encaixa nas expectativas do mercado internacional. Globalmente, de acordo com *UBM Electronics* (2012) e *CMP United Business Media* (2006), o uso de SO's em projetos de Sistemas Embarcados gira em torno de 70% desde 2006. Segundo a pesquisa, apenas 11,6% dos alunos mostraram expectativa de uso entre 60% e 80%.

Também é interessante notar que cerca de 30% dos entrevistados acreditam que esse uso se restringe a menos de 20% dos projetos. Esse pouco crédito dado a SO's em Sistemas Embarcados é reflexo do pouco contato que os alunos têm com essa arquitetura de software para Sistemas Embarcados.

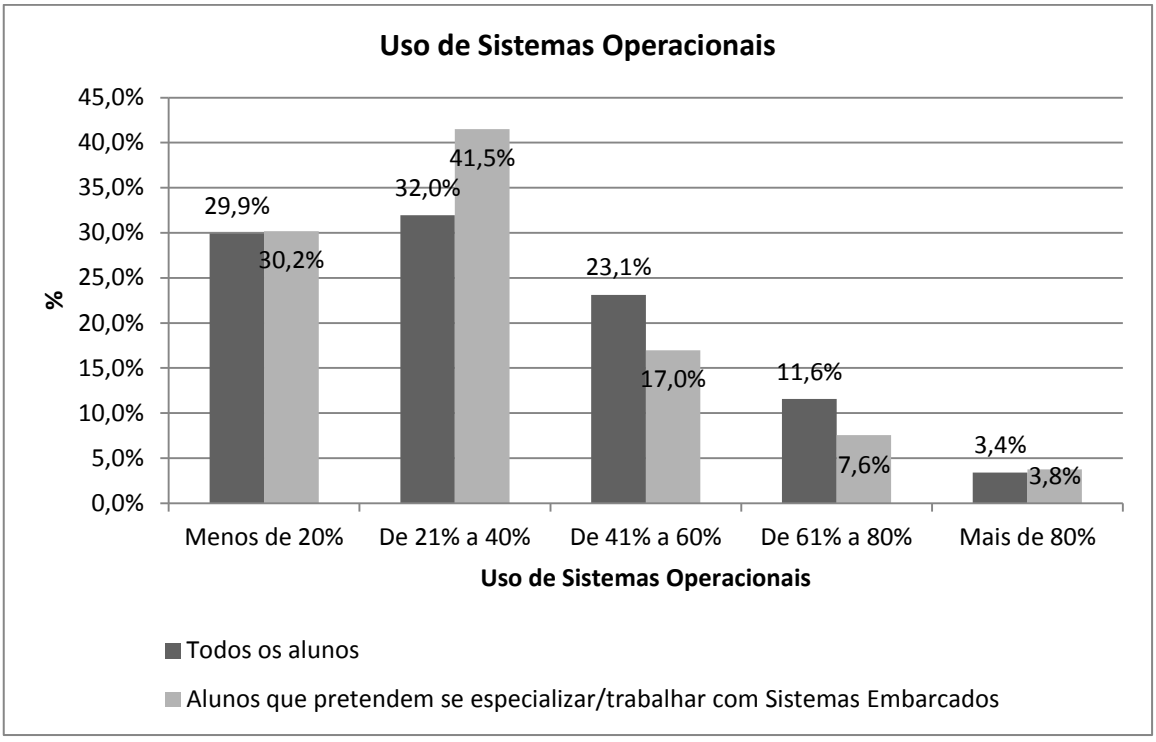

Figura 20 - Distribuição percentual da opinião dos alunos sobre o uso de Sistemas Operacionais pelo mercado de desenvolvimento.

A mesma perspectiva é enxergada ao analisarmos apenas os alunos da USP, como mostrado na Figura 21.

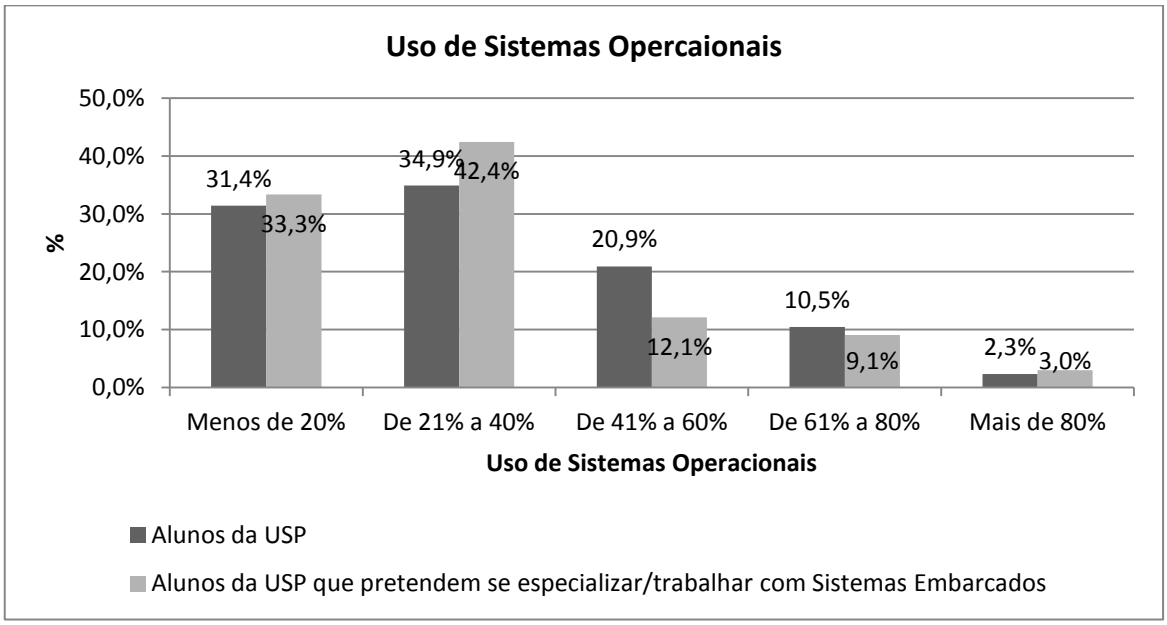

Figura 21 - Distribuição percentual da opinião dos alunos da USP sobre o uso de Sistemas Operacionais pelo mercado de desenvolvimento.

Ao analisarmos os principais motivos para se usar um SO em um Sistema Embarcado, os desenvolvedores brasileiros citam a Necessidade de Gerenciamento de Multitarefas, em 47,7% dos casos, Exigência de Tempo-real, em 15,8% dos casos e a Velocidade no Desenvolvimento, em 15,8% dos casos. A Modularidade (10,5%), o Reuso (5,3%) e a Robustez (5,3%) também são apontados como fatores para o uso. (BORGES 2011a)

Em posse desses dados, podemos verificar que a maioria dos alunos está correta ao apontar a Necessidade de Gerenciamento de Multitarefas como fator principal, presente em 57,4% das respostas. Essa mesma leitura é identificada ao mudarmos a forma de análise, considerando apenas alunos com pretensão de se especializar ou de trabalhar com Sistemas Embarcados, como mostrado na Figura 22.

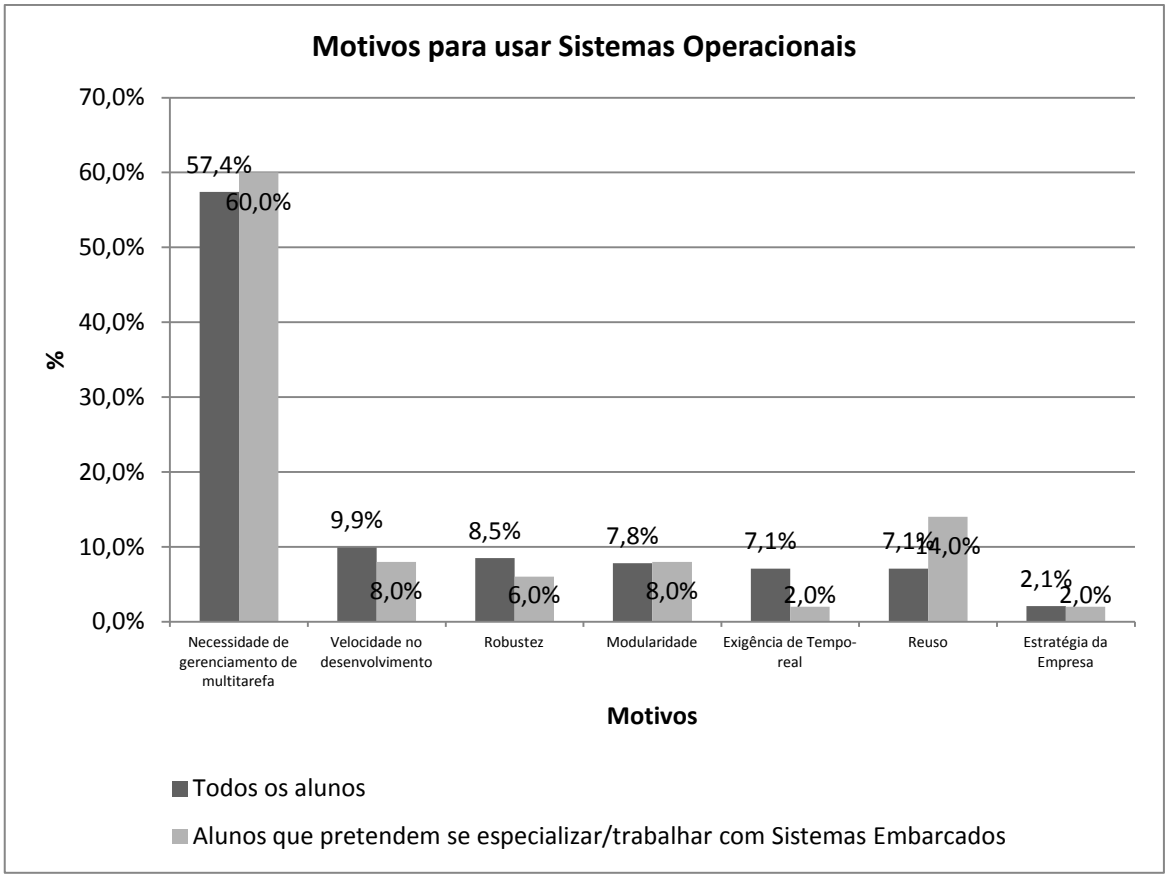

Figura 22 - Distribuição percentual da opinião dos alunos sobre os principais motivos para o uso de Sistemas Operacionais pelo mercado de desenvolvimento.

O pouco contato que os alunos tiveram com SO's para Sistemas Embarcados faz com que a Exigência de Tempo-real seja pouco cotada como fator de utilização, sendo lembrada por apenas 2% dos alunos que pretendem se especializar ou trabalhar com Sistemas Embarcados. Em qualquer sistema seja ele embarcado ou de propósito geral, quando existe a necessidade de gerenciamento de

multitarefas, o uso de SO é sempre uma primeira opção. Isso mostra que os alunos trazem essa leitura do contato com Sistema Operacional de Propósito Geral, não estando familiarizados com as peculiaridades dessa arquitetura para Sistemas Embarcados.

Entre os alunos da USP, a leitura é a mesma, com a diferença que alunos da USP que pretendem se especializar ou trabalhar com Sistemas Embarcados dão mais valor ao Reuso como fator de seleção. A Figura 23 trás os dados dos alunos da USP.

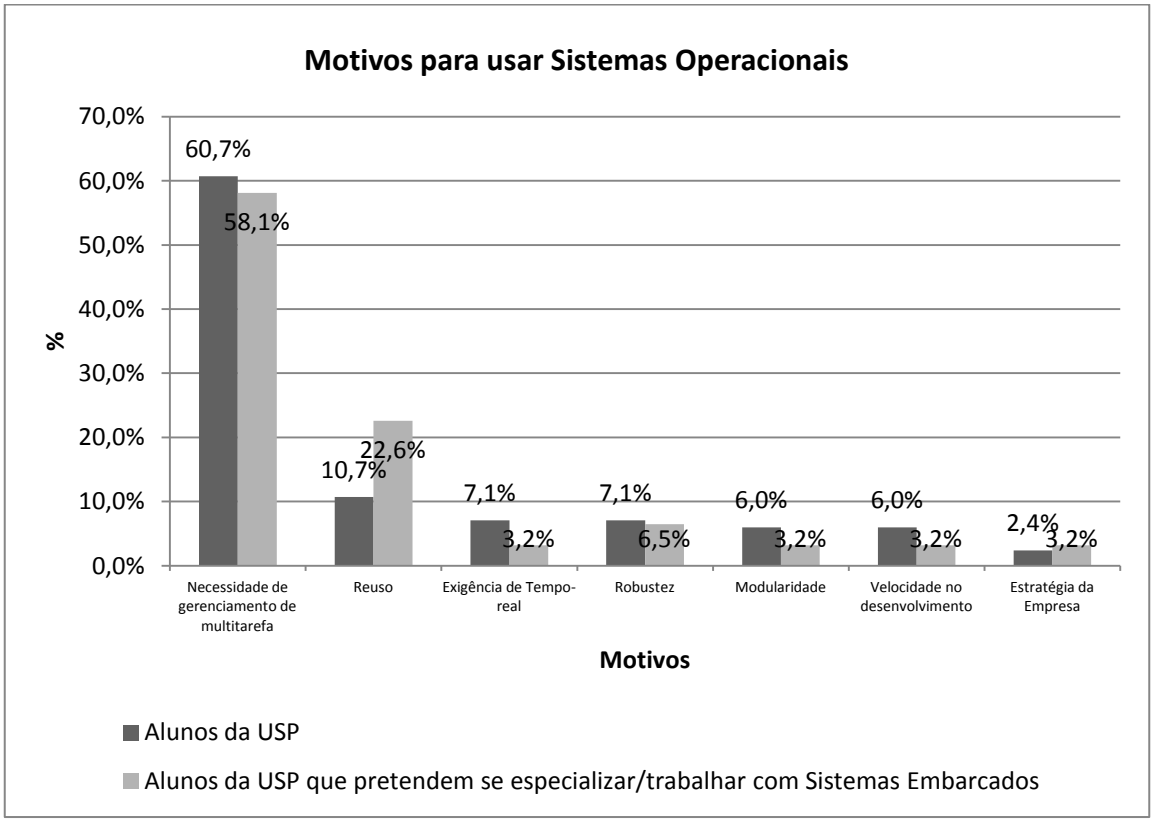

Figura 23 - Distribuição percentual da opinião dos alunos da USP sobre os principais motivos para o uso de Sistemas Operacionais pelo mercado de desenvolvimento.

Considerando os motivos para a não utilização de um SO, 32,4% dos alunos apontaram a Exigência de Grandes Recursos de Processamento como motivo principal e 19,0%, o Consumo Excessivo de Memória, como mostra a Figura 24.

Essa expectativa não condiz com o mercado brasileiro de desenvolvimento, onde, segundo Borges (2011b), 45,8% dos projetos não utilizam SO por Falta de Exigência. A análise dos alunos que pretendem se especializar ou trabalhar com Sistemas Embarcados foi mais próximo da realidade, ao colocar a Não Exigência do Projeto como segundo fator para não utilização.

O primeiro fator apontado pelos alunos é apenas o 5° mais relevante no mercado nacional, com apenas 4,2% de citações entre os participantes da pesquisa realizada por Borges (2011b).

Assim, como no mercado nacional, o que mais justifica a não utilização de Sistemas Embarcados em projetos de SO's internacionalmente, segundo a *UBM Electronics* (2012) e a *CMP Business Media* (2006), desde 2006, é a Não Exigência do Projeto, em cerca de 80% dos projetos, número ainda mais considerável que no mercado nacional. O consumo de memória e processamento foi relevante esse ano de 2012, juntos, para apenas 8% dos desenvolvedores.

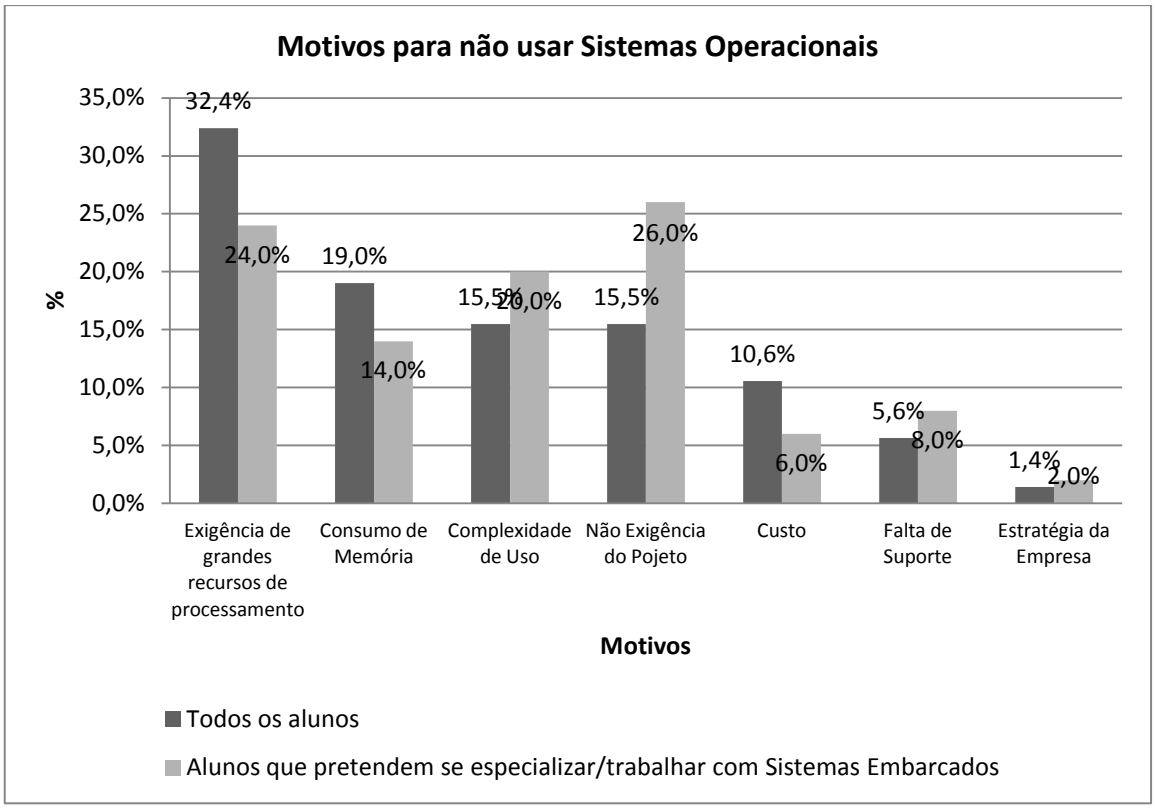

Figura 24 - Distribuição percentual da opinião dos sobre os principais motivos para não se utilizar Sistemas Operacionais pelo mercado de desenvolvimento.

Novamente os dados mostram pouco conhecimento dos alunos com relação às peculiaridades do mercado de trabalho, o que é extremamente compreensível.

Os alunos da USP que pretendem se especializar ou trabalhar com Sistemas Embarcados, por sua vez, acreditam que a Não Exigência do Projeto seja o principal fator para a não utilização um SO. Apesar desses alunos ainda darem bastante crédito à Exigência de Grandes Recursos de Processamento, essa é a análise mais próxima da realidade. A Figura 25 trás a opinião dos alunos da USP sobre esse assunto.

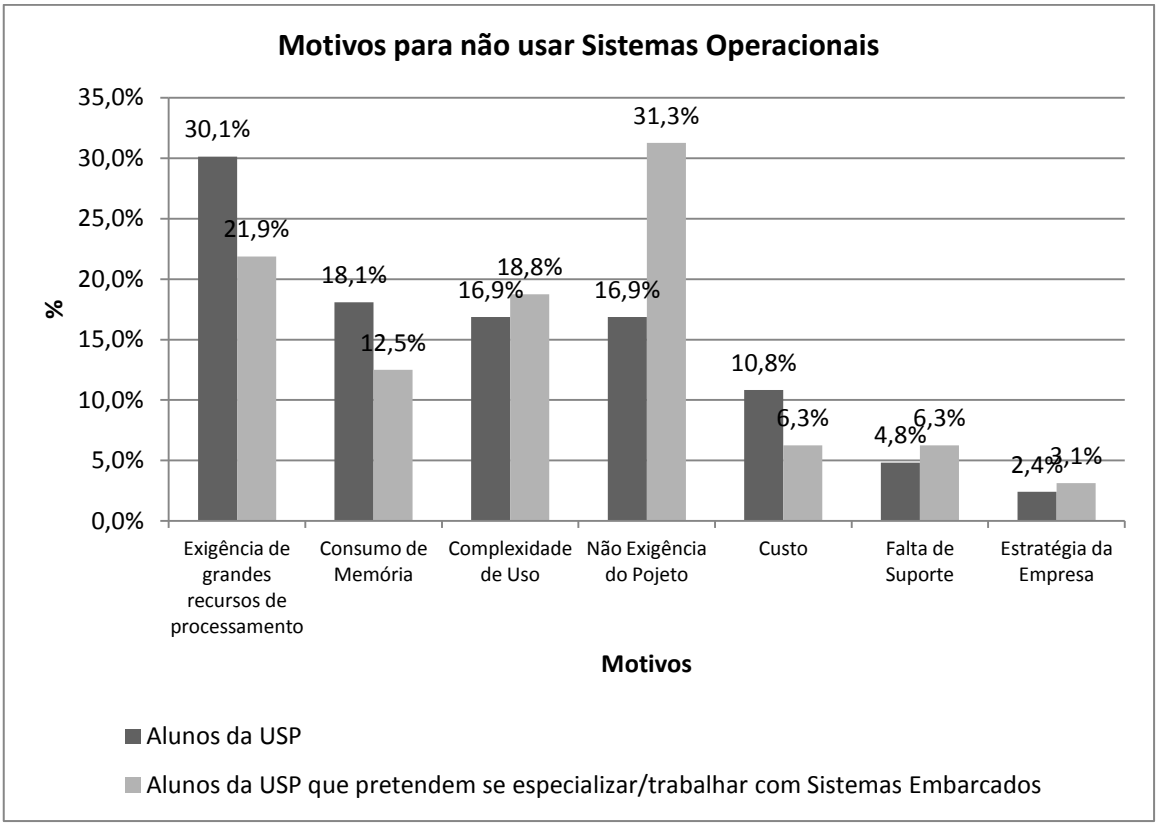

Figura 25 - Distribuição percentual da opinião dos da USP sobre os principais motivos para não se utilizar Sistemas Operacionais pelo mercado de desenvolvimento.

4.2. Estudo de Viabilidade

De acordo com a pesquisa *Embedded Market Survey*, realizada anualmente pela *UBM Electronics* (2012) desde 2008, 62% dos projetos de Sistemas Embarcados consomem mais recursos no desenvolvimento do software do que do hardware.

Esse número reforça a pressão por um desenvolvimento de software mais assertivo, preciso e que atenda as necessidades da aplicação da melhor forma possível, porém que seja otimizado e envolva o menor tempo de desenvolvimento e o menor custo possível.

Essa estratégia passa diretamente pela escolha da arquitetura do software embarcado. Como já dito, duas arquiteturas se destacam: a clássica *Superloop* e os SO's. A segunda arquitetura vem sendo usada em cerca de 70% dos projetos internacionais desde 2008, segundo a UBM Electronics (2012).

No Brasil, esse uso é consideravelmente menor. Apenas 28% dos projetos de Sistemas Embarcados brasileiros usam SO's, segundo a pesquisa realizada por Borges (2011b). No entanto,

70% dos entrevistados pela mesma pesquisa disseram considerar o uso de SO's em projetos futuros.

A diferença entre os dois mercados é compreensível, uma vez que a utilização de microcontroladores de grande porte, como 32 bits, é maior fora do país. Quando o mercado brasileiro começar a explorar mais os projetos de porte maior, é natural que essa utilização aumente, já que o maior fator de exclusão para os SO's em projetos de Sistemas Embarcados no Brasil é a falta de exigência do projeto. Com projetos mais complexos, a tendência é procurar arquiteturas de software mais adequadas.

4.2.1. Arquitetura do Software

A arquitetura Superloop é amplamente utilizada e tem a vantagem de necessitar de pouca memória e exigir poucos recursos de processamento. O sistema é dividido em dois níveis, *Background* e *Foreground*. No primeiro geralmente são alocadas as tarefas de maior prioridade no sistema. No segundo, são alocadas as tarefas de menor prioridade no sistema. O *Background* é geralmente dividido em *slots* de tempo, e as tarefas alocadas nesse nível são divididas entre esses *slots* de tempo de forma a garantir o atendimento do *deadline* de cada tarefa. Isso requer maior conhecimento do hardware e análise prévia das tarefas para determinação de seus *deadlines* antes de dividi-las entre os *slots*. Muitos fabricantes disponibilizam informações sobre os tempos de respostas dos módulos na especificação do hardware, o que facilita este trabalho.

O código fica restrito apenas à implementação das tarefas necessárias ao funcionamento de aplicação e ao gerenciamento das tarefas, que é simples. As tarefas de *Background* executam sequencialmente em *loop* infinito, sendo um *slot* de tempo a cada *loop*, sendo interrompidas apenas no caso de uma interrupção do sistema disparar uma tarefa de *Foreground*. Essa tarefa, então, é atendida e a execução retorna ao ponto em que havia parado no *Background*.

Porém, no *Superloop*, existem apenas esses dois níveis de prioridade, *Foreground* e *Background*. Caso aumente o número de tarefas de *Foreground*, ou seja, as tarefas com *deadline* curto, serão necessário mais tempo de processamento do sistema para essas tarefas, o que dificultaria o atendimento do *deadline* das tarefas de *Background*. Caso aumente o número de tarefas de *Background*, os *slots* de tempo podem não ser suficientes para atender todas as tarefas em seus respectivos tempos – estouro dos *slots* – o que pode exigir uma redefinição desses *slots* e maior dificuldade em atender os *deadlines* de suas tarefas.

Além disso, essa estrutura desperdiça processamento aguardando o tempo para a execução do próximo *slot*. Mesmo que nenhuma das tarefas alocadas a um determinado *slot* esteja pronta para ser executada, esse *slot* consumirá todo o seu tempo antes de liberar o processador para o próximo *slot*. (BORGES 2011a)

As desvantagens da arquitetura *Superloop* podem inviabilizar o seu uso se prejudicarem a principal vantagem dela, que é sua simplicidade.

Os SO's por sua vez, facilitam o gerenciamento de tempo, o tratamento de multitarefas e a priorização de tarefas. Usando essa arquitetura é possível configurar a prioridade que achar necessária à tarefa e deixar que o próprio sistema gerencie o atendimento de seus *deadlines*. Isso não exclui o mesmo conhecimento prévio das tarefas e de seus *deadlines* descritos para a arquitetura *Superloop*, uma vez que uma pré-determinação mal feita da prioridade das tarefas pode levar a degradação do sistema, mas facilita na hora de testar e alterar essas prioridades.

Outra vantagem do SO é possuir ferramentas de sincronização e escalonamento de tarefas, que facilitam sua coordenação e execução. Um bom algoritmo de escalonamento garante que o processador não ficará ocioso, uma vez que consegue distribuir as tarefas da melhor forma a atender seus *deadlines* e otimizar o uso do processador. As ferramentas de sincronização, como os semáforos, facilitam o acesso aos recursos de hardware e a troca de mensagens evita o que é comum no *Superloop*, que uma tarefa tente executar sem estar pronta para isso.

As principais vantagens dos SO's, então, são os múltiplos níveis de prioridade, que permitem uma melhor organização do sistema em comparação ao *Superloop* e o algoritmo de escalonamento permite uma distribuição mais otimizada da execução das tarefas.

Suas desvantagens, no entanto, são justamente as vantagens da arquitetura *Superloop*. O código não é restrito apenas às tarefas do sistema, sendo que é adicionada a ele toda a estrutura do SO, bloco de controle e pilhas das tarefas. Essa estrutura adicional, obviamente, reflete em um consumo maior de memória que no caso da arquitetura *Superloop*.

Também estão presentes em SO's tarefas que controlam toda essa estrutura do adicional do Sistema. Para essas tarefas, normalmente por recomendação do próprio fabricante do software, são atribuídas as maiores prioridades do sistema, para que nenhuma tarefa da aplicação interfira no funcionamento geral do SO. Essas tarefas adicionais exigem mais do processador, principalmente por serem prioritárias. O processador precisa continuar garantindo os *deadlines* das tarefas da aplicação mesmo com o tempo de processamento gasto com as tarefas estruturais do SO.

Outra desvantagem dessa arquitetura é que geralmente envolvem uma complexidade maior de uso. A utilização de um SO exige um estudo ou um conhecimento prévio do software, além de o tratamento e alocação das tarefas não serem tão intuitivos como no caso da arquitetura *Superloop*.

As duas arquiteturas tem a mesma característica de permitir o reuso de software. Para ambas existe a possibilidade de definir *frameworks* que facilitam e aceleram a programação. Outra vantagem do reuso é que esses *frameworks* já foram previamente testados e validados, o que garante maior robustez e qualidade ao sistema.

4.2.2 Arquitetura do Hardware

Esta análise será feita considerando duas arquiteturas de microcontroladores, de 8 e 32 bits. Essa escolha se deu por serem as duas arquiteturas mais utilizadas no desenvolvimento de sistemas no Brasil, em, respectivamente, 31% e 34% dos novos projetos, segundo Borges (2011b). Fora do país, segundo a *UBM Electronics* (2012) e a *CMP United Business Media* (2006), o uso da arquitetura de 32 bits e novos projetos vem crescendo desde 2006, passando de 54% a 63% esse ano, enquanto a arquitetura de 8 bits mantém seu mercado de atuação em torno dos 13%. Apesar da parcela significante dos microcontroladores de 16 bits, seu uso vem caindo, segundo a *UBM Electronics* (2012) e a *CMP United Business Media* (2006), passando de 20% dos novos projetos em 2006 pra 16% dos novos projetos em 2012. Essa queda é resultado da competitividade que a arquitetura de 32 bits vem alcançando no mercado.

Os microcontroladores de 8 bits dominam o mercado de desenvolvimento de Sistemas Embarcados de pequeno porte. A principal vantagem é que esses microcontroladores são extremamente baratos, principalmente quando comprados em grandes volumes. Esse barateamento, juntamente com uma gama maior de funcionalidades, praticamente descontinuou o uso de microcontroladores de 4 bits, que perderam competitividade e hoje são usados em apenas 2% dos projetos de Sistemas Embarcados no país, segundo Borges (2011a).

Esses microcontroladores são originalmente destinados a aplicações que exigem poucas funcionalidades e apresentam maiores restrições de consumo de energia, tamanho e custo. Para que essas exigências sejam atendidas, costumam possuir menos recursos de memória e processamento.

Contudo os microcontroladores de 8 bits se beneficiaram com o desenvolvimento da microeletrônica, tendo evoluído em robustez, desempenho e funcionalidades. Ao analisarmos os principais fabricantes de microcontroladores de 8 bits apontados pelos entrevistados da pesquisa *UMB Electronics* (2012), verificamos diferentes capacidades de memória e processamento na

mesma família, existindo microcontroladores com as mesmas especificações de desempenho que um microcontrolador de grande porte.

Essa evolução, logicamente, reflete em custo. Os microcontroladores de 8 bits mais modernos e com essas novas especificações de desempenho são mais caros que microcontroladores mais antigos e com capacidades reduzidas.

Os microcontroladores de 32 bits são os maiores competidores do mercado de microcontroladores, segundo Borges (2011a). É a principal arquitetura usada nos mercados de desenvolvimento nacional e internacional, segundo Borges (2011b) e *UBM Electronics* (2012), e sua utilização continua crescendo.

Esses microcontroladores são destinados a aplicações com mais exigência de funcionalidades, oferecem maiores capacidades de memória e processamento. Apesar da capacidade elevada, seu uso em Sistemas Operacionais continua exigindo baixo consumo de energia, uma vez que muitas das aplicações a que são destinadas trabalham com recursos limitados de energia, como pilhas e baterias.

Assim como no caso dos microcontroladores de 8 bits, a evolução da microeletrônica também favoreceu os microcontroladores de 32 bits. A principal vantagem que essa evolução trouxe a essa arquitetura foi a diminuição do seu preço no mercado, o que é um dos grandes fator que tornam esses microcontroladores dominantes no mercado de desenvolvimento.

Apesar da diminuição de preço, que aproxima os microcontroladores de 32 bits dos microcontroladores de 8 bits, continuam sendo menos viáveis principalmente em aplicações muito pressionadas pelo custo. Mesmo possuindo mais funcionalidades, o que poderia justificar a diferença de preço na hora de especificar um projeto, essa diferença, mesmo ela sendo pouca, pode significar um aumento do custo final do projeto que não compense essas funcionalidades.

4.2.3. Consumo de Memória

Por não ter sua memória ocupada pela parte estrutural do SO, bloco de controle das tarefas e pilhas, sendo a memória apenas ocupada pela parte do software que implementa as funcionalidades exigidas pela aplicação, a arquitetura Superloop se torna imbatível quando falamos em consumo de memória. Como já analisado, o SO exigirá do microcontrolador melhores recursos de memória, por mais otimizado que esteja o software.

A restrição de memória, aliada ao uma ocupação elevada, impede que o sistema possa ser expandido em caso de necessidade de novas funcionalidades ou limita as alterações que possam ser

exigidas durante ou após a conclusão do projeto. Além do mais, ao se tentar embarcar um SO em um microcontrolador com pouca capacidade de memória, pode ser necessário alterações na estrutura das tarefas originais da aplicação de modo a possibilitar que o consumo total do software não exceda a memória disponível. Essa alteração pode degradar o funcionamento original do sistema caso a alteração dificulte o atendimento correto das tarefas.

Ao analisarmos o que o Borges (2011a) diz sobre o consumo de memória, verificamos que, segundo seus experimentos, a utilização de um SO em um Sistema Embarcado só é viável para microcontroladores com mais que 4 Kbytes de memória RAM.

Nesse sentido a arquitetura de 32 bits parece mais adequada, uma vez que geralmente apresentam melhores especificações de memória, enquanto os microcontroladores de 8 bits costumam possuir memórias mais reduzidas.

Levando em consideração a resposta dos entrevistados da pesquisa *UBM Electronics* (2012) sobre quais famílias de microcontroladores de 8 bits consideram utilizar no próximo projeto, verificamos que para todas essas famílias existem várias opções de capacidade de memória. Por exemplo, 44% dos desenvolvedores, segundo a *UBM Electronics* (2012) consideram a utilização de um microcontrolador PIC de 8 bits de Microchip para o próximo projeto. Nessa família existem microcontroladores com menos de 1 Kbytes de memória RAM como existem micros com 4 Kbytes de memória RAM, compatíveis em pinagem e na maioria das funcionalidades, sendo a principal diferença um acréscimo no preço do produto. Ao fazermos a cotação, microcontroladores dessa família com no mínimo 4 Kbytes de memória RAM tem um custo mínimo próximo de U$2.50, contra um custo abaixo de U$1 em microcontroladores com menos especificações de memória.

Essa mesma análise pode ser estendida as outras três famílias mais citadas pelos desenvolvedores entrevistados pela *UBM Electronics* (2012). Microcontroladores de 8 bits das famílias AVR da *Atmel*, HC08/HC011 da *Freescale* e ST08 da *STMicroelectronics* possuem opções com mais de 4 Kbytes de memória RAM e mais de 64 Kbytes de RAM, todas com acréscimo no preço.

O problema de alto consumo de memória, então, pode ser contornado pelos próprios microcontroladores de 8 bits, uma vez que as principais famílias desses microcontroladores possuem opções de memória que atendam as restrições de memória impostas pelo SO.

Ao analisarmos os motivos para desenvolvedores brasileiros não utilizarem SO’s em seus projetos, presente em Borges (2011b), o consumo de memória é o fator de maior relevância para 21% deles, enquanto fatores relacionados exclusivamente ao mercado, como a falta de exigência do projeto e a estratégia da empresa, chega a 60% das respostas. Isso mostra que, apesar de relevante

na hora de decidir ou não pela utilização do SO, o consumo de memória é preterido por questões internas da empresa.

Essas questões muitas vezes são dirigidas por prazo e custo. Por estarmos incluídos em um mercado que raramente atende aos prazos pré-estabelecidos e que têm uma pressão por custo muito alta, segundo Borges (2011a), na hora de determinar as características do novo projeto, é preferível um microcontrolador que seja mais barato e que permita expansão sem custo adicional de hardware do que investir em uma solução que permita funcionalidades mais robustas.

Colaborando ainda com a diminuição do custo, quanto mais hardware e software forem reusados, menos se gasta com desenvolvimento e teste, e mais se ganha em tempo. 30% dos novos produtos do mercado brasileiro, segundo Borges (2011a), são de arquitetura de 8 bits, muitos por já fazerem uso de estruturas de hardware e frameworks de software já desenvolvidos para outros projetos.

O consumidor brasileiro, diferente de outros países, principalmente desenvolvidos, ainda pesa muito o custo na hora de comprar o produto, em detrimento de fatores como novas funcionalidades e novos recursos. Isso se reflete no mercado, onde cada centavo economizado no projeto é crucial para o sucesso ou não de um produto em campo.

4.2.4. Consumo de Processamento

A principal vantagem da arquitetura *Superloop* com relação ao consumo ou exigência de processamento é justamente o pouco consumo e a pouca exigência. O processador de uma aplicação que funcione com a arquitetura *Superloop* será responsável apenas pelo processamento das tarefas referentes à aplicação, sendo que a estrutura do software em si não agrega nenhuma exigência ao hardware. O SO, no entanto, apresenta tarefas ligadas a sua estrutura, e essas tarefas são prioritárias às da aplicação, o que gera maior necessidade de processamento.

Por sua vez, a divisão em *slots* de tempo da arquitetura Superloop pode levar o processador à ociosidade. Essa ociosidade ocorre quando as tarefas programadas para executar naquele *slot* não ocupam todo o tempo disponível para ele. Apesar de o processador não estar sendo usado nessas situações, ele só será liberado para uso quando o tempo do *slot* finalizar, gerando um tempo perdido em que seria possível o processamento de outra tarefa. O escalonador no SO faz essa função de garantir o mínimo de ociosidade do processador, contribuindo para a eficiência do sistema.

A determinação correta das prioridades das tarefas é essencial para que o processador não seja inteiramente consumido pela execução de poucas tarefas. Em ambas as arquiteturas isso exige um estudo prévio das características e exigências de cada tarefa, porém os SO's possuem mais opções de priorização e maneiras mais fáceis de alterá-las. A alteração da prioridade envolve apenas a redefinição de valores numéricos, enquanto que na arquitetura *Superloop* pode envolver a redefinição do nível de *Foreground* ou de um ou vários slots de tempo do nível de *Background*.

Sem dúvidas o SO exige maior poder de processamento que a arquitetura *Superloop*. Por esse motivo, assim como no caso do consumo de memória, os microcontroladores de 8 bits ficam em desvantagem. O uso de um SO em um Sistema Embarcado de 8 bits, que costuma apresentar menor capacidade de processamento que microprocessadores de 32 bits, por exemplo, pode levar o processador ao limite de seu funcionamento.

Quando analisamos as consequências de um sistema no limite dos recursos de processamento, podemos tirar as mesmas conclusões do caso dos recursos de memória. Um sistema carregado muitas vezes dificulta ou inviabiliza sua expansão ou atualização.

Sendo mais próximo ao mercado, segundo Ganssle (2006), um sistema que ocupa 90% do seu limite de processamento duplica o tempo de desenvolvimento sobre um que ocupa 70% de seu limite e triplica quando essa porcentagem chega a 95%.

A maior exigência de recursos de processamento do SO é um fator de exclusão para sua utilização em microcontroladores de 8 bits. Ao analisarmos as principais famílias de microcontroladores apontados pelos entrevistados pela *UBM Electronics* (2012), da mesma forma que na sessão anterior, vemos que contornar essa exigência substituindo o microcontrolador por um da mesma família é mais complicado que no caso do consumo de memória, porém ainda assim é possível. Novamente, como no caso anterior, a diferença está no preço do componente.

Juntando-se a isso o fato de apenas 4,2% dos desenvolvedores brasileiros, segundo Borges (2011b), considerarem a grande exigência de recursos de processamento como fator de exclusão do SO no projeto de um Sistema Embarcado, novamente temos a pressão por custo como principal fator de exclusão.

4.2.5. Gerenciamento de Multitarefas

A necessidade de gerenciamento de multitarefas é o principal motivo apontado por 47,4% dos desenvolvedores brasileiros para justificar a utilização de um SO em um projeto de Sistema

Embarcado, segundo Borges (2011b). O SO facilita o gerenciamento de tempo, priorização e sincronização das tarefas do sistema.

O gerenciamento das tarefas está ligado diretamente as suas prioridades dentro do sistema. Para as duas arquiteturas de software são necessárias análises prévias das tarefas com a função de determinar suas prioridades. Muitos SO's possuem essa determinação dinâmica, sem a necessidade de interferência e estudo do programador, porém a maioria dos Sistemas Operacionais de Tempo-Real usados em Sistemas Embarcados não é provida dessa funcionalidade.

A diferença é que a arquitetura Superloop apresenta apenas dois níveis de prioridades, o nível de *Foreground* e o nível de *Background*, enquanto que os SO's possuem diversos desses níveis, além de possuir ferramentas de sincronização e escalonamento, que facilitam a coordenação e execução das tarefas. Um bom algoritmo de escalonamento garante uma boa distribuição da execução das tarefas.

A possibilidade de definir diferentes prioridades possibilita a inclusão de novas funcionalidades de alta prioridade ao sistema e deixar a cargo do escalonador gerenciar sua execução. No caso da arquitetura *Superloop*, o aumento do número de tarefas de alta prioridade compromete o atendimento das tarefas de baixa prioridade, e o aumento do número de tarefas de baixa prioridade pode causar estouro dos *slots* de tempo. Ambos os casos levam à degradação do sistema.

Toda essa análise de gerenciamento de multitarefas é praticamente independente da arquitetura de hardware utilizada no projeto. Seja em projetos com arquiteturas de 8 bits, seja em projetos de arquiteturas de 32 bits, se o hardware tiver as especificações necessárias, o gerenciamento de multitarefas via SO será mais eficiente que o gerenciamento de multitarefas via arquitetura *Superloop*.

4.2.6. Atendimento do *Deadline*

Uma das principais funções do software, principalmente quando relacionado a um aplicação com restrições de tempo, é garantir que todas as tarefas sejam atendidas sem que essa restrição seja quebrada. O não atendimento desses tempos, ou *deadlines*, pode causar problemas sérios na funcionalidade do produto ou degradar seu funcionamento. Mesmo que o não atendimento do *deadline* de uma tarefa não acarrete nenhum problema mais sério ao sistema, ao determinarmos um *deadline* para cada tarefa, é interessante que ele seja cumprido.

São esses tempos de resposta das tarefas que determinam a prioridade que elas terão no sistema. Tarefas com tempos de resposta curtos são prioritárias em relação a tarefas com tempos de respostas mais longos.

Nesse sentido, a arquitetura *Superloop* dificulta o gerenciamento de tempo das tarefas, uma vez que possui apenas dois níveis de prioridade. No nível de *Foreground*, são incluídas as tarefas com *deadlines* menores e no nível de *Background*, as tarefas com *deadlines* maiores. Caso aumente o número de tarefas com *deadlines* curtos, o sistema exigirá mais do processador e comprometerá o atendimento das tarefas de *Background*. Da mesma maneira, se aumentar o número de tarefas de *Background*, pode acarretar o estouro dos *slots* de tempo, prejudicando o funcionamento correto do sistema.

A vantagem do SO é novamente a desvantagem da arquitetura *Superloop*. O SO funciona como um catalisador no processo de desenvolvimento, facilitando o escalonamento de tarefas e agilizando a execução das de maior prioridade. O SO possui diversos níveis de prioridade, o que facilita a inclusão de novas tarefas ou alteração de suas prioridades. A determinação da execução das tarefas é de responsabilidade do escalonador, e não mais do programador, como no caso do *Superloop*. Esse mesmo escalonador garante um uso mais eficiente do processador, evitando a ociosidade que pode ser gerada pelos *slots*.

Novamente a vantagem da arquitetura *Superloop* é possuir apenas tarefas ligadas à aplicação em si. Um SO, por outro lado, possui tarefas ligadas ao funcionamento, gerenciamento e execução das atividades indispensáveis ao seu funcionamento. Essas tarefas, como dito anteriormente, são prioritárias por recomendação dos próprios desenvolvedores do SO. A simples existência dessas tarefas já reduz o tempo disponível para a execução das tarefas referentes à aplicação, sendo essas tarefas prioritárias, então, esse tempo é ainda mais disputado.

O principal problema da utilização de um SO no atendimento do *deadline* das tarefas, então, é poder ter o processador alocado a maior parte do tempo para o atendimento das tarefas relacionadas à sua estrutura. Caso isso aconteça, será difícil ao sistema garantir o atendimento do *deadline* das tarefas pertencentes à aplicação, ainda mais quando envolve aplicações com *deadlines* baixos. A solução pra isso é ter um processador que suporte tanto as tarefas do SO quanto as tarefas da aplicação.

Isso mostra que o atendimento ou não do *deadline* das tarefas está ligado mais as especificações de desempenho, validação constante, construção e alocação correta das tarefas do sistema do que com a arquitetura do microcontrolador. Considerando os apontamentos feitos sobre a evolução do poder de processamento dos microcontroladores de 8 bits, eles tem total condições

de garantir as restrições de tempos de uma aplicação contanto que possuam requisitos de desempenhos necessários.

4.3. Análise do Mercado Brasileiro e Internacional de Desenvolvimento de Sistemas Embarcados

O objetivo desta sessão é identificar quanto que os fatores relacionados ao mercado influenciam na utilização de SO's em projetos de Sistemas Embarcados no Brasil, principalmente nos que utilizam microcontroladores de pequeno porte (arquitetura 8 bits). A análise será baseada na pesquisa *Embedded Market Survey*, presente em *UBM Electronics* (2012), realizado anualmente desde 2008 e do artigo *Embedded System Design: An Overview Brazilian Development*, desenvolvido pro Borges (2011b), que fazem uma análise dos mercados de desenvolvimento de Sistemas Embarcados global e nacional, respectivamente. Ao final dessa sessão, é apresentada considerações sobre os principais Sistemas Operacionais Embarcados utilizados por desenvolvedores no Brasil e ao redor do mundo.

De acordo com os tópicos estudados na sessão anterior, foi possível descobrir que os problemas técnicos que costuma inviabilizar o uso de SO's em projetos de Sistemas Embarcados podem ser contornados. Para complementar, segundo Ganssle (2006), produtos utilizando microprocessadores de 8 bits podem usar os benefícios de um *kernel* dependendo da complexidade do produto. Ainda segundo ele, desenvolvendo produtos usando micros de 8 bits, é possível descobrir que o uso de um *kernel* simplifica a programação sem adicionar muito custo.

Ao olharmos para o mercado de desenvolvimento de Sistemas Embarcados, apenas 28% dos projetos nacionais, segundo Borges (2011b), faz uso de algum SO. Desses 28%, 63% são em projetos com microcontroladores de 32 bits e apenas 5% são em projetos com microcontroladores de 8 bits.

Essa diferenciação não é explicada pelo volume de projetos desenvolvidos com cada uma das arquiteturas, uma vez que ambas dominam o mercado de desenvolvimento nacional. Dos novos projetos, 31% fazem uso de microcontroladores de 8 bits e 34% de microcontroladores de 32 bits. (BORGES 2011a)

Ao compararmos com o mercado global de desenvolvimento de Sistemas Embarcados, onde cerca de 70% dos projetos fazem uso de algum SO, segundo a *UBM Electronics* (2012), vemos que ele é menos reticente em utilizar esse tipo de arquitetura. Segundo a mesma pesquisa o uso de microcontroladores de 32 bits no mercado mundial é maior que no nacional, chegando a

63% dos novos projetos, contra 13% dos microcontroladores de 8 bits. Isso mostra que a utilização de SO's nos dos mercados está associada à utilização de microcontroladores de grande porte.

Como verificado anteriormente, o uso de um SO demanda maior poder de processamento e memória do microcontrolador, uma vez que existem tarefas intrínsecas ao sistema a serem adicionadas as tarefas da aplicação. Isso justifica sua utilização maior em microcontroladores de 32 bits.

Não podemos, também, justificar a não utilização de SO's, principalmente no mercado brasileiro, pela complexidade de utilização dessa arquitetura de software. Apenas 8% dos desenvolvedores nacionais consideram a complexidade um problema e esse número, globalmente, é de 4% dos desenvolvedores.

Também não podemos justificar a falta de utilização por termos um mercado ligado à manutenção e alteração de projetos já em campo. 62% dos projetos brasileiros são ligados a novos produtos, porcentagem inclusive maior que a média mundial, de 44% em 2012. Contudo, ao iniciar um novo projeto, os desenvolvedores tem todo um *know-how* de problemas com custo e prazo, que os direcionam ao reuso de software e hardware.

No Brasil, segundo Borges (2011b), o principal desafio dos projetos de Sistemas Embarcados é o prazo. Apesar de, segundo Borges (2011b) e *UBM Electronics* (2012), os projetos nacionais e globais terem ciclos de vida parecidos, sendo que a maioria deles tem duração de 6 a 12 meses, o atraso no cronograma é maior no Brasil. Apenas 30% dos projetos nacionais são entregues no prazo estabelecido, contra 42% do mercado global em 2012. A principal diferença é que, no Brasil, 36% dos projetos sofrem prorrogação de mais de seis meses, contra 12% dos projetos internacionais.

A pressão por prazo estimula o reuso por parte do desenvolvedor, ao invés de buscar novas tecnologias para a aplicação, tanto de software quanto de hardware. O reuso como já dito diminui o tempo de codificação do software e de teste de hardware e software, acelerando o processo de desenvolvimento e trazendo maior robustez ao projeto final. Software e hardware reusado tem também a vantagem de já ser de conhecimento dos desenvolvedores, que não perdem o tempo que perderiam estudando novas funcionalidades.

Além disso, para praticamente 70% dos desenvolvedores brasileiros entrevistados por Borges (2011b) apontaram o custo como o principal fator na hora da escolha do microcontrolador. Isso mostra que o mercado brasileiro é fortemente pressionado pelo custo em seus projetos, o que justifica a grande utilização de microcontroladores de 8 bits, geralmente mais baratos que microcontroladores de 32 bits. Novamente temos o reuso como agente facilitador nesse caso. O

reuso de hardware e software diminui o custo nos projetos, pois evitam que novos módulos de hardware e software sejam desenvolvidos.

O principal motivo apontado, tanto no Brasil quanto fora dele para a não utilização de SO's em Sistemas Embarcados, é a falta de exigência do projeto, sendo fator majoritário para 45% dos desenvolvedores nacionais e 80% dos desenvolvedores ao redor do mundo.

A alta taxa de aceitação internacional, em adição aos dados citados anteriormente nos mostra que o mercado de desenvolvimento global só não usa SO em projetos em que seu uso seria realmente desnecessário.

No Brasil, por outro lado, a principal justificativa da não utilização é a própria não utilização. Um mercado que prima pelo custo como o mercado brasileiro e sofre constantemente com o prazo tem a tendência em sempre buscar a solução mais rápida e menos custosa. A solução mais rápida está diretamente ligada ao reuso e, a mais barata, a menos funcionalidades de hardware.

Seguindo com a análise, segundo *UBM Electronics* (2012), cerca de 50% dos entrevistados consideram que as mudanças mais dramáticas nos últimos cinco anos estão relacionadas à tecnologia do microcontrolador. Essa evolução da tecnologia de desenvolvimento de hardware vem favorecendo os microcontroladores de 8 bits, cada vez mais robustos, com maior capacidade de memória e maior poder de processamento. Como já verificado, existem microcontroladores de pequeno porte com poder de memória e processamento suficientes para atender as necessidades do SO. Somado as facilidades já descritas que essa arquitetura trás ao desenvolvimento, era de se esperar que o volume de aplicações de 8 bits que utilizam SO's crescesse.

Essa evolução tecnológica obviamente não beneficiou apenas a arquitetura de 8 bits. Os microcontroladores de 32 bits estão cada vez mais baratos e ainda mais competitivos no mercado. Essa queda faz com que seu preço, em alguns casos, se aproxime do preço de microcontroladores de pequeno porte. Ao ponderarmos a diferença de custo com a diferença de funcionalidades entre os dois microcontroladores, é de se esperar que as funcionalidades superem a diferença de preço. Contudo, isso depende da pressão que o custo exerce sobre o produto.

O momento em que o mercado de desenvolvimento nacional resolver trocar seu paradigma e adotar o uso de SO's será o momento em que a pressão por prazo diminua a ponto de o projeto possuir um ciclo de vida maior que justifique e viabilize a mudança da estratégia do software e a adaptação a esta mudança.

Quando essa mudança acontecer, o mercado tem que estar preparado para investir mais, não só no desenvolvimento do software, como no hardware. Nesse momento talvez o investimento não justifique a utilização do um microcontrolador de 8 bits. Com mais funcionalidades e preço decrescente, os microcontroladores de 32 bits podem, então, ser escolhidos para esta tarefa.

O que vemos no mercado brasileiro hoje, contudo, é o processo inverso. Empresas investem na busca por redução de custo em seus projetos. Essa busca muitas vezes passa pela diminuição das funcionalidades do hardware e alterações no software que justifiquem a troca do microcontrolador por um com menos capacidade, porém mais barato.

Um dado interessante trazido em Borges (2011a) é que 70% dos entrevistados tem pretensão de usar SO no próximo projeto e 100% dos que utilizam dizem que esse uso se estenderá ao próximo projeto. Isso mostra que essa arquitetura tem aceitação entre os desenvolvedores nacionais e que seu uso é ponderado a cada novo projeto.

Quanto às universidades, o estudo de SO's para Sistemas Embarcados crescerá quando o mercado começar a exigir profissionais com esse conhecimento. Contudo, se a universidade se antecipar e criar em seus alunos o conceito de que a utilização de SO's é vantajosa, eles mesmos comecem a mudar os paradigmas do mercado antes que esse mude o paradigma das universidades.

4.3.1 Principais Sistemas Operacionais Embarcados usados no mercado brasileiro e internacional

Os desenvolvedores de Sistemas Embarcados brasileiros, quando questionados sobre o tipo de SO que utilizam em seus projetos mostram uma preferência pelos sistemas *open-source*, segundo Borges (2011a). 65,1% dos desenvolvedores nacionais preferem esse tipo de SO, contra 17,5% que preferem distribuições comerciais e 17,5% que preferem desenvolver o SO internamente. Novamente vemos que a pressão por custo influencia os desenvolvedores brasileiros, inclusive na escolha do SO a ser usado.

Ao analisarmos os dados presentes na pesquisa realizada pela *UBM Electronics* (2012), vemos que a preferência dos desenvolvedores ao redor do mundo é por sistemas comerciais, em 40% dos casos esse ano, contra 31% da preferência por sistemas *open-source* e 20% de preferência por sistemas desenvolvidos internamente.

Contudo a preferência por sistemas comerciais vem diminuindo de 2008 pra cá. Há quatro anos o uso de sistemas comerciais atingia 49% dos projetos mundiais, e a preferência por sistemas *open-source* era de apenas 19%. Acrescentando o fato de que 37% dos desenvolvedores ao redor

do mundo, segundo a *UBM Electronics*, pretendem usar sistemas *opens-source* no próximo projeto, contra 31% que pretendem usas sistemas comerciais, vemos a tendência de que a utilização de sistemas *open-source* supere a utilização de sistemas comerciais.

A principal vantagem de se utilizar um sistema comercial é a redução no tempo de desenvolvimento. Soma-se a essas vantagens a disponibilidade de serviços, o suporte técnico e o fato de muitas empresas estarem certificando seus SO's segundo regulamentações e padrões de diversos setores.

A desvantagem, logicamente, é o fato de terem custo incluído, que irá impactar o valor final do produto ou do projeto. Alguns sistemas operacionais cobram *royalties* por unidade vendida que utiliza o software. Nestes casos há um impacto direto no custo do produto. Outros SO's cobram licenças para o uso do software, impactando o custo final do projeto.

Os sistemas *open-source* não impactam o custo final do projeto ou do produto, como no caso de sistemas comerciais, porém não oferecem o suporte e a documentação oferecida por estes, o que pode dificultar o desenvolvimento do software embarcado.

Quando questionados sobre quais SO's consideram utilizar, a pesquisa realizada por Borges (2011a) mostra que 43% dos desenvolvedores nacionais consideram a utilização de *Linux* Embarcado e 34% deles a utilização de um *FreeRTOS*.

A preferência dos desenvolvedores ao redor do mundo, segundo a *UBM Electronics* (2012) é de utilização de *Android*, apontado por 34% dos entrevistados. Isso é resultado da popularização nos últimos anos de *Smartphones* e *Tablets* que fazem uso dessa distribuição de SO. Em segundo lugar na preferência está o *FreeRTOS*, presente em 23% das respostas.

Esses dados são condizentes com o anteriormente apresentados que sugerem o crescimento de utilização de SO's *open-source* em Sistemas Embarcados. Mesmo que por motivos diferentes, tanto o mercado nacional quanto o internacional de desenvolvimento de Sistemas Embarcados serão cada vez mais dominados por sistemas *open-source*, com destaque para o *Linux* Embarcado.

O mercado nacional ainda não incorporou massiçamente o uso de Sistemas Operacionais em seus projetos de Sistemas Embarcados, mas, quando o fizer, a tendência é de utilizar opções *open-source* principalmente por que o mercado de desenvolvimento internacional está seguindo essa direção e tende a ter mais aplicações desenvolvidas com sistemas *open-source* do que com sistemas comerciais em pouco tempo.

CAPÍTULO 5: CONCLUSÃO

5.1. Conclusões

O estudo de Sistemas Embarcados nas universidades começa geralmente no 3° ano acadêmico, sendo que o contato com esse tipo de sistema antes desse período costuma ser devido a atividades extracurriculares, como programas de Iniciação Científica ou grupos de estudos.

Na USP, 100% dos alunos dos cursos que tem alguma ligação com Sistema Embarcado saem da universidade com esse conhecimento. Tanto na USP quanto nas demais universidades o assunto é abordado em matérias obrigatórias e posteriormente em matérias optativas. A média de matérias obrigatórias cursadas relacionadas ao tema é de duas matérias ao final do curso.

Entre 35% e 40% dos alunos procuram continuar os estudos em Sistemas Embarcados ainda na universidade por meio de disciplinas obrigatórias ou atividades extracurriculares. Essa é a mesma variação percentual de alunos que pretendem se especializar ou trabalhar com Sistemas Embarcados. Esses números mostram o interesse que tal área desperta nos alunos, ainda mais ao considerarmos que o número de alunos que pretendem seguir nesse nicho do mercado aumenta após o primeiro contato na universidade.

Os alunos do curso de Engenharia de Computação são os mais interessados em se especializar ou trabalhar com Sistemas Embarcados, tanto na USP quanto nas demais universidades. Quase 50% dos alunos desse curso têm essa pretensão. Isso é resultado da crescente integração entre hardware e software que os projetos de Sistemas Embarcados vêm buscando, o que o torna um mercado agradável para alunos com uma formação que abrange as duas áreas.

O estudo de Sistemas Embarcados nas universidades brasileiros está bastante relacionado aos microcontroladores de 8 bits. Quase 60% dos alunos saem da universidade com conhecimento dessa arquitetura e esse número na USP ultrapassa 70%. Isso mostra a função didática desses microcontroladores, geralmente usados para introduzir o estudo de microcontroladores.

Com o mesmo caráter introdutório que a arquitetura de 8 bits, a linguagem de programação de microcontroladores *Assembly* é a mais estudada nas universidades. Mais de 80% dos alunos saem da universidade com conhecimento dessa linguagem. Vale destacar que a linguagem C para Sistemas Embarcados é de conhecimento de cerca de 70% dos alunos quando formados, o que é condizente com o mercado de desenvolvimento, onde a utilização dessa linguagem é dominante.

Ao analisarmos o estudo de Sistemas Operacionais para Sistemas Embarcados, vemos que as universidades brasileiras ainda não incorporaram essa ementa às suas grades curriculares. Apenas 27% dos alunos saem das universidades com o conhecimento desse tipo de arquitetura de software para Sistemas Embarcados. Quando olhamos apenas os alunos que pretendem se especializar ou trabalhar com Sistemas Embarcados, esse número é um pouco maior, atingindo 35% deles.

Isso mostra que as universidades, além de não oferecerem esse conteúdo de forma obrigatória, não disponibilizam muitas maneiras de o aluno obter esse conhecimento e ainda não criou em seus alunos o conceito da importância que essa arquitetura de software vem tendo para o desenvolvimento de Sistemas Embarcados, mesmo naqueles alunos que querem seguir nesse mercado.

Isso é consequência do próprio mercado de desenvolvimento brasileiro ainda não ter assimilado essa tendência mundial. Enquanto os SO's são usados em cerca de 70% dos projetos ao redor do mundo desde 2006, no Brasil, esse percentual não chega aos 30%.

Sabendo que o mercado brasileiro de desenvolvimento emprega uma maioria de profissionais jovens e com menos de cinco anos de experiência, quanto melhor a preparação do aluno dentro de universidade antes de ingressar nesse mercado, mais rápida será sua adaptação e maior será o ganho para o empregador e seus projetos.

Quando analisamos a perspectiva dos alunos sobre o mercado de desenvolvimento de Sistemas Embarcados, vemos que os alunos são corretos ao especularem sobre questões técnicas, como quais arquiteturas de microcontroladores dominam o mercado. Também percebemos que muitas de suas perspectivas são direcionadas por questões aprendidas na universidade. As constantes vantagens apresentadas na universidade sobre o reuso de software fazem os alunos sobre-estimarem esse reuso no mercado profissional, assim como o pouco contato com SO's para Sistemas Embarcados faz os alunos subestimarem esse uso nos novos projetos desenvolvidos.

Muitas respostas dos alunos também são influenciadas por ainda terem uma visão de consumidor. Certos fatores como qualidade e inovação em um produto são mais representativos a um consumidor do que questões internas da empresa, como custo do projeto, prazos estimados e estratégia da empresa. Essas mesmas questões internas da empresa permeiam a maioria dos projetos desenvolvidos no Brasil e ao redor do mundo.

Apesar de relevante, não é a formação do profissional que influencia na pouca utilização de SO's no mercado brasileiro de desenvolvimento de Sistemas Embarcados. O mercado brasileiro é muito dependente de questões relacionadas ao custo e é muito pressionado pelo prazo, que na

maioria das vezes precisa ser estendido. Para reforçar esse fato, vale ressaltar que 70% dos desenvolvedores no mercado brasileiro consideram o uso de SO's em seus projetos, e esse número é de 100% entre os que já utilizaram. Isso mostra que os desenvolvedores conhecem as vantagens dessa arquitetura, não podendo, em algumas situações, implementá-la pelas exigências dos projetos em que são alocados.

Essas pressões fazem com que o desenvolvedor busque sempre a solução mais rápida e menos custosa. Essa solução passa diretamente pelo reuso do hardware e do software. Utilizar um hardware que permita a utilização de um SO demandaria custo extra com o hardware, que precisaria ser mais robusto, e custo extra com o software, que precisaria ser atualizado para a nova arquitetura. Considerando que em mais de 60% dos projetos ao redor do mundo o gasto no desenvolvimento do software é maior que no hardware, a migração para uma nova arquitetura de software precisaria ser extremamente necessária ou lucrativa.

A grande utilização de microcontroladores de 8 bits nos projetos brasileiros mostra que temos um mercado de desenvolvimento que ainda não prima pelas funcionalidades que poderiam ser incluídas ao se utilizar um microcontrolador de grande porte. Soma-se a isso o fato de a maioria dos projetos não utilizarem SO's simplesmente por falta de exigência, o mercado não faz mais uso dessa arquitetura porque as questões que envolvem a mudança de paradigma não são economicamente viáveis.

Quando tivermos um mercado de consumo que priorize produtos com maiores funcionalidades ao invés de produtos mais baratos, o mercado de desenvolvimento se verá obrigado a mudar esse paradigma e utilizar SO's em seus projetos, pelas facilidades de gerenciamento e desenvolvimento que essa arquitetura apresenta em comparação com a arquitetura comumente utilizada, a *Superloop*. Essa última exige menos de processador e memória do que um SO, o que a torna mai atraente a projetos com pressão de custo, porém dificulta a inclusão de novas funcionalidades, principalmente se essas forem críticas ao sistema.

Porém, no momento em que essa mudança acontecer, ela continue não sendo aplicada a microcontroladores de pequeno porte. A evolução da tecnologia de hardware favoreceu não só os microcontroladores de 8 bits, mais robustos e mais capazes, mas também barateou microcontroladores de grande porte, como de 32 bits, o que tornaram essa arquitetura a maior competidora do mercado.

Quando a necessidade de mais funcionalidades justificar o aumento da utilização de Sistemas Operacionais nos projetos de Sistemas Embarcados brasileiros, a comparação custo versus funcionalidades talvez faça o mercado seguir na direção dos microcontroladores de 32 bits

ao invés de continuar investindo nos microcontroladores de 8 bits. Isso fará que o desenvolvimento de projetos com arquiteturas de pequeno porte fiquem restritas as aplicações que realmente não necessitam de uma arquitetura de software diferente as usada hoje em dia.

Para que seja viável a utilização de SO's em microcontroladores de pequeno porte (8 bits) é necessário que esse microcontrolador atenda as necessidades de memória e processamento exigidos da arquitetura. Por possuir tarefas relacionadas à estrutura do SO, essa arquitetura exige mais do processador para que esse possa atender tanto as novas tarefas quanto as tarefas da aplicação, sem exceder o *deadline* dessas tarefas. Da mesma maneira, necessita de maior poder de memória, uma vez que precisa alocar espaço para as pilhas das tarefas, objetos relacionados ao SO e ao próprio sistema.

Todas as principais famílias de microcontroladores existentes no mercado tem processadores de 8 bits com capacidade de atender essas necessidades. Em contrapartida esses microcontroladores são mais caros, e essa diferença de preço, em grandes volumes, pode significar um aumento de custo muito elevado no final de um projeto.

Para finalizar, o estudo de diferentes arquiteturas de software e hardware permitiu abranger as duas frentes de estudo do curso de Engenharia de Computação, focando sempre na integração entre hardware e software e nas vantagens em unir as duas frentes de pesquisa. Sistema Embarcado é uma frente de estudo dentro da engenharia elétrica e da computação que tem a característica de permitir o conhecimento de diferentes soluções de software vinculadas ao hardware.

5.2. Trabalhos Futuros

Analisar a grade curricular dos cursos abordados e propor melhorias que permitam um percentual maior de alunos com conhecimento de Sistemas Operacionais para Sistemas Embarcados.

Desenvolver um estudo de caso experimental que confirmem as conclusões tiradas de acordo com o estudo teórico realizado.

Implementar esse estudo de caso em diferentes arquiteturas de hardware e para diferentes Sistemas Operacionais.

REFERÊNCIAS

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR10520: Informações e documentação: Apresentação de citações em documentos**, Rio de Janeiro, 2002.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR6023: Informações e documentação: Referências – Elaboração**, Rio de Janeiro, 2002.

BANNATYNE, R. **Don't write-off 8-bit microcontrollers**. Disponível em: <http://www.electronicsweekly.com/Articles/24/08/2009/46802/dont-write-off-8-bit-microcontrollers.htm>. Acesso em: set. 2012.

BERGER, A. **Embedded System Design**, [s. l.]: CMP Books, 2002.

BORGES, R. W. **Aplicabilidade de Sistemas Operacionais de Tempo Real (RTOS) para sistemas embarcados de pequeno porte e baixo custo**, Dissertação (Mestrado) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2011a.

BORGES, R. W. **Embedded System Design: An Overview of Brazilian Development**, In: IEEE International Workshop of Ubiquitous Media Embedded System (UMES2011), Buzan, Coréia do Sul, 2011b.

BUTAZZO, G. **Research trends in real-time computing for embedded system,** [s. l.]: ACM SIGBED, 2006.

CMP UNITED BUSINESS MEDIA. **2006 Embedded System Design: State of Embedded Market Survey**. [s. l.]: CMP United Business Media, 2006.

FRIEDRICH, L. F. **A Survey on Operating System Support for Embedded Systems Properties** In: VI Workshop de Sistemas Operacionais (WSO2009), Bento Gonçalves, 2009.

GANSSLE, J. **RTOS Dissatisfaction,** 2006. Disponível em: <http://www.embedded.com/electronics-blogs/break-points/4025675/RTOS-dissatisfaction>. Acesso em: set. 2012.

LABROSSE, J. J. **MicroC/OS-II – The Real Time Kernel**, 2nd ed. San Francisco, CA: CMP Books, 2002.

LEE, E. A. **Embedded Software**. Berkeley, CA: UCB ERL, 2002.

LI, Q.; CAROLYN, Y. **Real Time Concepts for Embedded Systems**, [s. l.]: CMP Books, 2003.

MAZIERO, C. A. **Sistemas Operacionais II: Gerência de Tarefas**, Curitiba, PR: Universidade Tecnológica Federal do Paraná, 2011.

NOERGAARD, T. **Embedded System Architecture: A comprehensive Guide for Engineers and Programmers**, Burlington, MA: Elsevier, 2005.

PRADO, S. **Mbed: Integrando o FreeRTOS em um Cortex-M3**, 2012. Disponível em: <http://sergioprado.org/mbed-integrando-o-freertos-em-um-cortex-m3/>. Acesso em: set. 2012a.

PRADO, S. **O primeiro treinamento de RTOS do Brasil**, 2012. Disponível em: <http://sergioprado.org/o-primeiro-treinamento-de-rtos-do-brasil/>. Acesso em: set. 2012b.

PONTIFÍCIA UNIVERSIDAD CATÓLICA DE CHILE. **PIC: completo guia**. Disponível em: <http://www2.ing.puc.cl/~iee3912/files/pic.pdf>. Acesso em: set. 2012.

SHANDLE, J. **More for Less: Stable Future for 8-bit Microcontrollers**, 2004. Disponível em: <http://www.eetimes.com/electronics-news/4196930/More-for-Less-Stable-Future-for-8-bit-Microcontrollers>. Acesso em: set. 2012.

SIMON, D. E. **An Embedded Software Primer**, 1st ed. India: Pearson Education, 1999.

SINGH, K. **Design and Evaluation of an Embedded Real-Time Micro-kernel**, Blacksburg, VA: Faculty of the Virginia Polytechnic Institute and State University, 2002.

SPERLING, E. **Hot Market for 8-bit microcontrollers**, 2008. Disponível em: <http://eecatalog.com/8bit/2008/04/18/hot-market-for-8-bit-microcontrollers/>. Acesso em: set. 2012.

TENEMBAUM, A. S.; WOODHULL, A. S. Sistemas Operacionais: Projeto e Implementação, 2ª ed.Porto Alegre: Bookman, 2000.

TURLEY, J **Operating System on the rise**, 2006. Disponível em: <http://www.eetimes.com/discussion/other/4025674/Operating-systems-on-the-rise>. Acesso em: set. 2012.

UBM ELECTRONICS, 2012 **Embedded Market Survey**, [s. l.], UBM Electronics, 2012.

APÊNDICE

Apêndice A – Enquete de Desenvolvimento de Embarcados

1. Universidade *

2. Formação *

a) Graduação
b) Mestrado
c) Doutorado
d) Pós-doutorado

3. Curso *

a) Engenharia de Computação
b) Engenharia Elétrica/Eletrônica
c) Engenharia Mecatrônica
d) Ciências da Computação
e) Informática
f) Outro:_____

4. Período *

a) 1°
b) 2°
c) 3°
d) 4°
e) 5°
f) 6°
g) 7°
h) 8°
i) 9°
j) 10°
k) Formado

5. Teve contato com Sistemas Embarcados na Universidade? *

a) Sim
b) Não

6. Quantas disciplinas OBRIGATÓRIAS relacionadas a Sistemas Embarcados cursou?

a) 1
b) 2

c) 3
d) 4
e) Outro: _____

7. Quantas disciplinas OPTATIVAS relacionadas a Sistemas Embarcados cursou?

a) 1
b) 2
c) 3
d) 4
e) 5
f) 6
g) Outro: _____

8. Teve contato com Sistemas Embarcados em atividades extracurriculares ou IC?

a) Sim
b) Não

9. Quais arquiteturas teve contato na Universidade?

a) 4-bits
b) 8-bits
c) 16-bits
d) 32-bits
e) 64-bits
f) FPGA's, SOC, etc.

10. Quais microprocessadores ou microcontroladores teve contato na Universidade?

11. Quais linguagens de programação para Sistemas Embarcados utilizou na Universidade?

a) Assembly
b) C
c) C++
d) Java
e) Outro: _____

12. Teve contato com Sistemas Operacionais para Sistemas Embarcados na Universidade? *

a) Sim
b) Não

13. Pretende se especializar ou trabalhar com Sistemas Embarcados? *

a) Sim
b) Não

14. Liste algumas utilizações de Sistemas Embarcados que conheça

15. Qual arquitetura acredita ser a mais utilizada pelo mercado de Sistemas Embarcados?

a) 4-bits
b) 8-bits
c) 16-bits
d) 32-bits
e) 64-bits
f) FPGA's, SOC, etc.

16. O que acredita ser o principal desafio no desenvolvimento de Sistemas Embarcados?

a) Custo
b) Qualidade
c) Segurança
d) Prazo/*Time-to-market*
e) Inovação
f) Outro: _____

17. Quais acredita serem os três principais fatores para a seleção do microprocessador em um projeto de Sistemas Embarcados?

a) Ferramentas Disponíveis
b) Desempenho
c) Custo
d) Compatibilidade com Sistemas Operacional
e) Código/Framework Disponível
f) Escalabilidade
g) Popularidade
h) Fornecedor
i) Consumo de Energia
j) Capacidade de Memória
k) Periféricos
l) Reuso
m) Número de I/Os
n) Estratégia de Empresa
o) Know-how
p) Outro: _____

18. Qual acredita ser a porcentagem de código reutilizado nos projetos de Sistemas Embarcados?

a) Não há reuso
b) Menos de 25%
c) De 26% a 50%
d) De 50% a 75%
e) Mais de 75%

19. Qual acredita ser a porcentagem de projetos de Sistemas Embarcados que utilizam Sistemas Operacionais?

a) Menos de 20%
b) De 21% a 40%
c) De 41% a 60%
d) De 61% a 80%
e) Mais de 80%

20. Qual acredita ser o principal motivo para a utilização de um Sistema Operacional em um projeto de Sistema Embarcado?

a) Necessidade de gerenciamento de multitarefas
b) Velocidade no desenvolvimento
c) Modularidade
d) Estratégia de Empresa
e) Reuso
f) Robustez
g) Exigência de tempo real
h) Outro: _____

21. Qual acredita ser o principal motivo para a NÃO utilização de um Sistema Operacional em um projeto de Sistema Embarcado?

a) Exigência de grandes recursos de processamento
b) Consumo de memória
c) Custo
d) Estratégia da Empresa
e) Complexidade de uso
f) Falta de suporte
g) Não exigência do projeto
h) Outro: _____

22. Comentários e Sugestões