Implementação de um sistema de desenvolvimento para microcontroladores 8051

João Pereira de Brito Filho[1], Remy Eskinazi Sant'Anna[2]

Resumo *— Este trabalho apresenta a implementação de um sistema de desenvolvimento para hardware e software, baseado no microcontrolador 8051, desenvolvido para ser utilizado cursos técnicos de nível médio, graduação e pós-graduação de Eletrônica.. Paralelamente, é apresentada uma metodologia de implementação de projetos, utilizando-se técnicas de particionamento para hardware e softwar. Através de um sistema operacional, o sistema de desenvolvimento em microcontrolador, apresenta facilidades de acesso aos recursos de um microcontrolador 8051 standard, bem como de interfaceamento com memória e dispositivos de E/S externos, além de armazenamento e execução de programa externo, em uma operação de download de programa. Para permitir estas operações de descarga de programa em memória externa do protótipo, de forma que este permanecesse executável, foi implementado também um programa de interface em ambiente gráfico, de modo a permitir a edição de diversos programas em Assembler do microcontrolador, chamadas para compilação e verificação de erros além da operação de download do programa compilado.O protótipo construído apresentou-se perfeitamente operacional, atendendo as especificações iniciais de projeto.*

Palavras chave — Microcontroladores 8051, Metodologias de projeto, Particionamento Hardware/Software, Protótipo.

Introdução

Atualmente, podem ser observados diversas aplicações, equipamentos e sistemas de controle baseados em microcontroladores. Notadamente os dispositivos da família 8051 são bastante populares , apesar de utilizarem barramentos de 8 bits, o que na verdade torna-se vantajoso para aplicações de pequeno e médio porte, que são a maioria das aplicações encontradas [5].

Apesar deste grau de utilização e popularidade, este dispositivo ainda não é estudado com profundidade e atenção merecidas, pelos cursos da área de Engenharia Eletrônica e de nível médio (UPE – Escola Politécnica e CEFET-PE), mais ainda, existem poucos ou nenhum recursos práticos para ensino que possam dar acesso à potencialidade deste dispositivo, proporcionando um maior e melhor nível de conceitrimento a utilização pratica, projetos e desenvolvimento de *hardware* e *software* [7].

Outro aspecto que merece atenção é o fato de que praticamente não existam sistemas de treinamento semelhantes a disposição com nível de informações de *hardware* e *software* abertos ou seja, que seja possível e permitido o melhoramento de suas características e potencialidades funcionais pois na maioria das vezes estes sistemas estão protegidos de alguma forma por *copyrights*. Alem disso, com freqüência, os sistemas importados tornam os custos de expansão e até mesmo de aquisição, proibitivos ao passo que o desenvolvimento de um sistema deste tipo no âmbito da UPE teria como vantagens imediatas uma maior flexibilidade de sistema , maior índice de informações de *hardware* e *software*, além de ser facilmente duplicável através da aquisição de componentes no mercado local [8].

O presente trabalho de pesquisa tem como objetivo o desenvolvimento de um sistema de treinamento e ensino em microcontroladores da família 8051 que possa ser utilizado posteriormente nos cursos de graduação e pós graduação de eletrônica, de forma que este permita, por possuir uma arquitetura aberta, deixar a disposição todas as facilidades de expansão de hardware e de software, e melhoramento de sua operacionalidade e recursos implementados.

Outro ponto observado no projeto é como sistemas deste tipo podem ser implementados. Freqüentemente, durante a execução de um projeto baseado em microcontrolador, ocorrem diversos erros de implementação por *hardware* ou *software*. As etapas de correção de erros, freqüentemente chamadas pelos projetistas como fase de *debug*, consomem bastante tempo de projeto. Estas etapas de *debug* podem ser minimizadas caso seja adotada uma metodologia de particionamento do projeto a ser implementado. Isto quer dizer que operando-se em módulos bem especificados e pequenos e, posteriormente promovendo-se a interação entre estes módulos, sejam estes em *hardware* ou *software*, se tem maior probabilidade de acerto na montagem final do projeto. O presente trabalho de pesquisa enfoca paralelamente, a utilização desta metodologia para implementação do sistema de desenvolvimento, bem como para projetos baseados em microcontroladores.

Sendo assim, os temas abordados no presente artigo são os seguintes:

- Implementação de um sistema de desenvolvimento em microcontroladores, aberto, de baixo custo, que

[1]João Pereira de Brito Filho, UFPE - DES, R. Acadêmico Hélio Ramos, s/n, 50740-530 Recife-PE,Brasil jbrito@ufpe.br
[2]Remy Eskinazi Sant'Anna, UPE - Escola Politécnica, Rua Benfica 455, CEP 50.750-410, Recife - PE – Brasil res@cin.ufpe.br
CEFET-PE R Prof. Luiz Freire 500 C. Universitária, Recife-PE Brasil

possa ser utilizado no âmbito dos cursos Eletrônica de nível médio, graduação, pós-graduação e atividades de pesquisa;

- Estudo de uma metodologia de implementação prática, através de particionamento de *hardware* e *software* de modo a facilitar o desenvolvimento de circuitos microcontrolados;

- Aprimoramento das técnicas de implementação de projetos baseados em controle por microcontroladores.

Metodologia de implementação de projeto baseado em microcontrolador

Apesar de se enfatizar aplicações da metodologia para microntroladores 8051, ela poderá ser aplicada, em grande parte, para qualquer sistema baseado em determinado dispositivo controlador, ou seja, aprenta-se uma metodologia de desenvolvimento de *hardware* e *software* que pode ser aplicada como técnica de projeto de sistemas microcontrolados. Neste ponto, pode-se estabelecer o que será denominado de etapas de desenvolvimento de um sistema baseado em microcontrolador. Estas etapas unificadas podem gerar na verdade um processo cíclico, ou seja, quando na etapa final de testes não forem alcançados os objetivos traçados na etapa inicial, o processo poderá ser revisto ou retornado em algum ponto.

Pode-se estabelecer pelo menos os seguintes pontos como etapas de desenvolvimento de sistemas [1]:

- Requerimentos funcionais do sistema;
- Especificações de projeto do sistema;
- Particionamento;
- Modularização de *hardware* e *software;*
- Integração de *hardware* e *software;*
- Elaboração da documentação de projeto;
- Verificação funcional e avaliação do sistema.

Estas etapas serão adaptadas às características do sistema a ser desenvolvido. Tem-se a seguir, uma descrição generalizada de cada etapa de desenvolvimento.

Requerimentos funcionais do sistema

Nesta etapa é confeccionada a documentação que descreve exatamente as capacidades, características e modo funcional do sistema quando terminado. Esta documentação deve conter, por exemplo, uma relação de todas as tarefas que o sistema deve realizar para o usuário final. Nesta etapa, deve existir um intercâmbio entre projetistas e usuários finais do sistema de forma a gerar os requerimentos funcionais de maneira a mais definitiva possível (que será o alicerce para todas as outras etapas de desenvolvimento) de forma consistente e viável.

Especificações de projeto de sistema

Uma vez que estejam definidos os requerimentos funcionais do sistema, devem ser feitas as especificações do projeto do mesmo. Isto quer dizer que, nesta etapa, deve-se decidir como o sistema deverá interagir com o usuário bem como especificar as entradas e saídas dos dispositivos envolvidos. Talvez seja fundamental, a confecção de uma relação das funções que o sistema deve realizar de forma a implementar as tarefas definidas pelo usuário.

Desta forma pode-se estabelecer que enquanto os requerimentos funcionais do sistema devem mostrar o *que* o dispositivo final deve fazer, as especificações de projeto do sistema devem detalhar *como* o sistema deve operar de forma a realizar as tarefas especificadas na etapa anterior.

Particionamento

Uma vez estabelecidas as especificações de projeto, poderá ser feito o particionamento entre *hardware* e *software,* ou seja, quais as funções do sistema que serão realizadas por *hardware* e *software*. Para cada projeto sempre ocorrerá situações em que uma função do sistema deve ser implementada apenas em *hardware* (ou *software*). Caso exista a possibilidade de a função ser implementada de outra maneira, devem ser levados em consideração outros aspectos, tais como, custo ou tempo requerido de implementação da nova solução. Existe nesta etapa, dependendo da complexidade do projeto, a possibilidade de ser preciso realizar um estudo de *hardware*/*software* codesign.

Modularização de *hardware e software*

Após a definição das funções que serão implementadas por *hardware* e por *software* no sistema, o projeto pode tomar dois caminhos distintos, geralmente dependentes da sua complexidade:

- Desenvolvimento de *hardware* e *software* por apenas um projetista;
- Desenvolvimento de *hardware* e *software* paralelo por projetistas ou equipe de projetos distintos

A primeira situação ocorre com freqüência em projetos de baixa complexidade que envolvem ciclos de implementação curtos e custo reduzido. Sistemas mais complexos como projetos de instrumentação ou controle de sistemas distribuídos, exigem um particionamento (até mesmo integração) de projeto em *hardware* e *software*. Em ambos os casos existe então a modularização do projeto de *hardware* e *software* ou seja, o projeto de *hardware* é dividido em módulos, cada qual

implementando uma ou mais funções de *hardware*. O *software* também é particionado em uma hierarquia de módulos, cada um dos quais implementa uma ou mais tarefas. Estes módulos podem posteriormente ser divididos em *procedures* que seriam rotinas para realizar funções específicas. O módulo final de *software* consistiria de grupos de *procedures* que estariam ligadas umas as outras.

Integração de *hardware* e *software*

A integração entre *hardware* e *software* pode ser vista como a etapa de compatibilidade entre os dois projetos especialmente se estes forem implementados de maneira paralela. Isto é possível de ser implementado normalmente pela instalação de uma memória não volátil com o sistema de *software* desenvolvido, adicionado ao *hardware* e fazendo posteriormente um teste de desempenho do sistema. Neste caso, deverá também ser implementada uma interface entre o *hardware* e o *software* ou seja, algum programa que permita a operação do *software* sobre o *hardware* projetado. Como exemplo desta interface, pode-se ter um módulo anterior ao *software* a ser integrado que identifique quais as funções ou recursos de *hardware* que estão implementados e que deverá interagir com o *software*. Outra possibilidade seria a utilização de um emulador do microcontrolador utilizado para verificação de funcionalidade do *software*.

Elaboração da documentação de projeto

Uma etapa não menos importante seria a geração de relatórios e anotações das tarefas realizadas e resultados obtidos durante o ciclo de desenvolvimento do projeto. Posteriormente, a unificação destes relatórios de forma lógica e ordenada formaria a documentação do sistema. A importância deste ponto pode ser percebida em um caso particular de desenvolvimento de um projeto de grande complexidade. Se as anotações estão incompletas ou escritas de forma não clara, surgirá a dificuldade de manutenção de *software* o que muitas vezes podem alcançar grandes percentuais de custos de desenvolvimento de *software*. Em resumo, técnicas bem direcionadas de projeto associadas a documentação de desenvolvimento e a objetivos bem definidos de projeto, conduzem a facilidades de implementação e menor custo de manutenção do sistema em desenvolvimento.

Verificação e avaliação do sistema

Finalmente, a ultima etapa seria a avaliação completa do sistema para a verificação dos requerimentos funcionais idealizados na primeira etapa. Pode-se perceber que todo processo de execução do projeto é interativo, ou seja, a detecção de erros em qualquer ponto significa o retorno para o ponto anterior do ciclo. A Fig. 1 mostra um fluxograma do processo de execução de projeto:

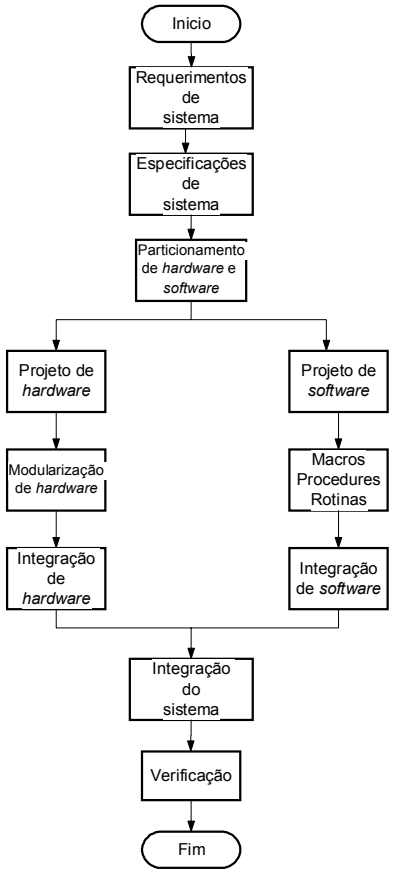

FIGURA 1 – METODOLOGIA DE IMPLEMENTAÇÃO DE PROJETO

As etapas para desenvolvimento do *hardware* [3] podem ser colocadas em um fluxograma como o ilustrado na Fig. 2, como considerações a serem observadas no desenvolvimento do *hardware* que atenda as especificações do sistema alvo.

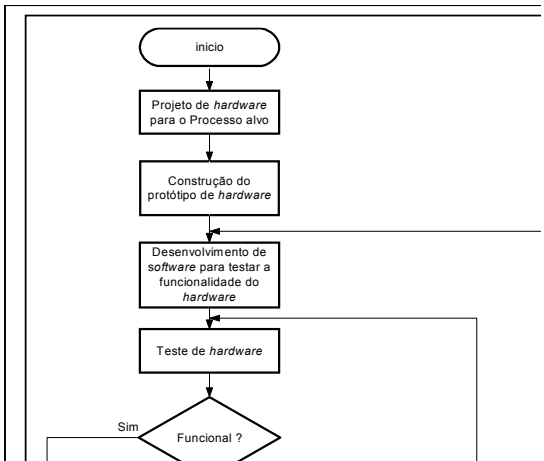

FIGURA 2 – METODOLOGIA DE IMPLEMENTAÇÃO DE *HARDWARE*

Para o desenvolvimento do sistema opracional foi utilizada uma metodologia Top-Down de particionamento de software. Neste tipo de implementação, as funções do sistema são especificadas e o projeto é dividido em subfunções onde, inclusive, podem ocorrer outros níveis mais abaixo destas subfunções. Como cada subfunção está especificada, isto limita e define as suas propriedades. Desta maneira, as propriedades das funções em níveis mais baixos tem uma maior probabilidade de integração nas etapas onde o *hardware* e *software* vão ser integradas. Percebe-se que o projeto procede das considerações dos maiores para os menores níveis ou do mais generalizado para os mais específicos. As funções de níveis mais subalternos deverão ser especificadas posteriormente às de maior nível, quando ficar claro de que maneira será a interação com estes módulos de níveis mais elevados, sendo que a implementação das funções de menor nível não poderá ocorrer , até que o processo de particionamento esteja completo.

Pode-se então fazer algumas considerações de como se apresenta em linhas gerais, um programa estruturado na metodologia "*top-down*". Na realidade existe normalmente um arranjo hierárquico de todas as rotinas do *software* desenvolvido. No nível de topo, existe uma rotina principal, freqüentemente chamada de rotina ou programa principal. A função desta rotina principal é gerenciar o fluxo de todo o programa sendo que as tarefas que devem ser executadas são implementadas como uma série de subrotinas ou *procedures*. O programa principal inicializa o sistema e então ele realiza entradas, *poolings* e *loops* de programa que examinam as entradas e controles do sistema, muitas vezes por subrotinas específicas para tal. Estas condições são testadas e se necessário, o programa principal chama as rotinas adequadas para realizar a tarefa necessária. Em sistemas mais complexos, o programa principal pode chamar uma das diversas subrotinas e muitas das procedures que estão operando, podem ter suas próprias subrotinas em níveis mais baixos. Em qualquer caso, o fluxo do programa ocorre a partir do programa principal para as subrotinas ou subdivisões destas e sempre retorna para o programa principal. A Fig. 3 mostra uma metodologia de implementação "*top-down*" [2].

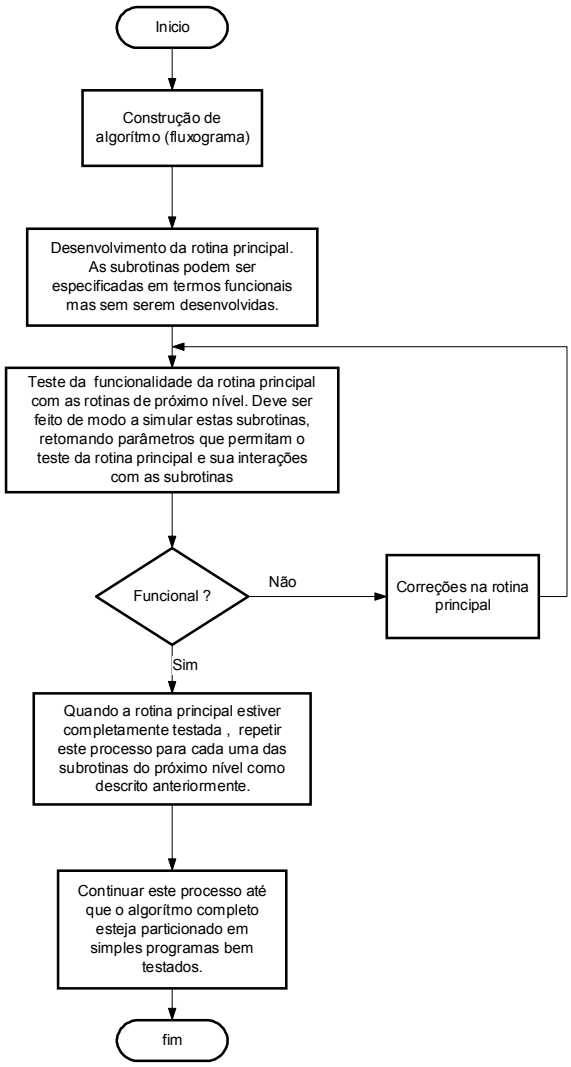

FIGURA 3 – METODOLOGIA DE IMPLEMENTAÇÃO DE *SOFTWARE TOP -DOWN*

Implementação do sistema de desenvolvimento

A partir dos requerimentos para um sistema de desenvolvimento para microcontrolador 8051, foram feitas

as especificações e o diagrama em blocos, mostrado na Fig.4:

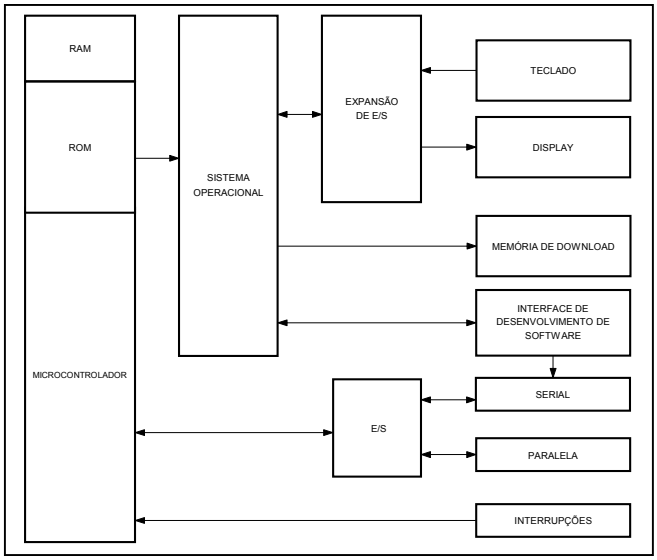

FIGURA 4 – DIAGRAMA EM BLOCOS DO HARDWARE DO SISTEMA

Implementação do sistema operacional

O sistema operacional foi implementado de acordo com os requerimentos do sistema, para atender a 8 funções básicas como pode ser visto na Fig. 6.

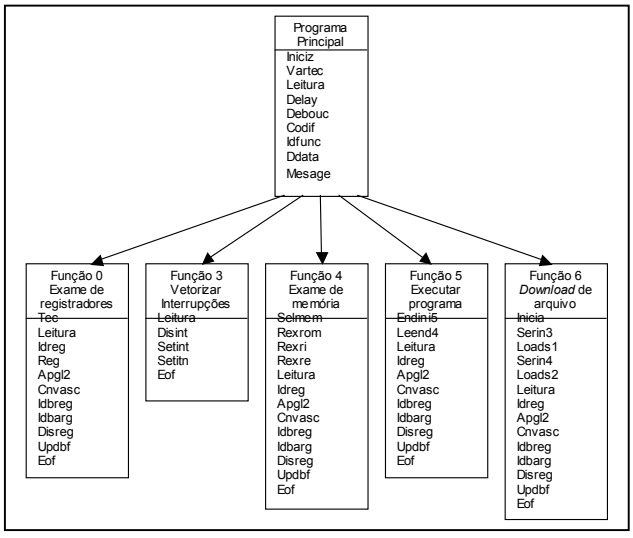

FIGURA 5 – PARTICIONAMENTO DO SISTEMA OPERACIONAL

As funções 1 e 2 estão reservadas a expansão pelo usuário. A função 7 está reservada para finalizar as outras funções e remeter o S.O. ao estado de reset. A figura mostra o particionamento do S.O com as principais subrotinas utilizadas de acordo com a metodologia Top-Down.

Implementação da Interface de Desenvolvimento

Este programa foi desenvolvido em Delphi, de forma a disponibilizar ao usuário a edição, chamada para compilação e download de arquivos binários para o sistema de desenvolvimento em ambiênte gráfico. A Fig. 7 mostra a tela do programa com 2 arquivos sendo editados.

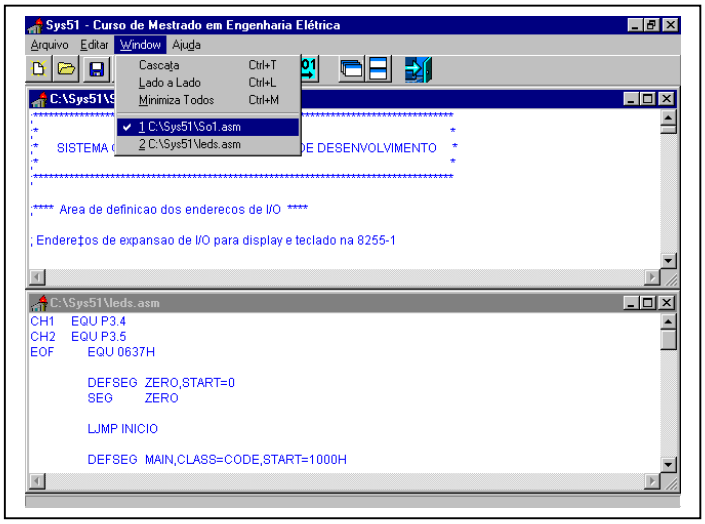

FIGURA 6 – TELA DA INTERFACE DE DESENVOLVIMENTO

Atualmente está sendo implementado uma interface de simulação do microcontrolador 8051, que ainda não se encontra disponível, dentro da filosofia deste sistema ser aberto a melhoramentos e modificações dentro do ambiente acadêmico. Desta forma, os programas poderão ser simulados antes de serem executados na plataforma de desenvolvimento.

Conclusões

A implementação deste sistema de desenvolvimento para microcontroladores 8051, sendo um sistema aberto, trouxe vantagens imediatas no processo de ensino-aprendizagem nas disciplinas correlacionadas a projetos com microcontroladores. Mesmo não apresentando uma arquitetura tão moderna, este microcontrolador (ou algum dos seus variantes) ainda é encontrado fartamente em aplicações de controle.

Com as informações de projeto disponibilizadas, o sistema tende a se desenvolver com diversos melhoramentos que podem ser implementados. A metodologia de projeto ora apresentada, sendo bastante simples e generalista, pode ser empregada em qualquer projeto que envolva o uso de microcontrolador, mesmo aquelas de arquitetura distinta (RISC por exemplo) [4].

Referências

[1] - Stwart, James W. – The 8051 microcontroller, regents/Pretice Hall Books (1995) New York

[2] - Peatman, John B. – Design With Microcontrollers, McGraw Hill International Editions – Computer Engineering series – (1992)

[3] - Atmel Corporation – AT89C51 Microcontroller Data sheets (1994)

[4] – Atmel AVR 901200 series data sheet, Disponível para download em http://www.atmel.com

[5] - The 8051 web page tutorial at http://www.8052.com.

[6] - [11] Schultz, Thomas W.; " C and the 8051 – Hardware, Modular Programming and Multitasking" Second Edition –Vol 1 - 1998.

[7] - Keil web site: http://www.keil.com

[8] – Sant´Anna, Remy E. – " Uma Metodologia de Desenvolvimento para Sistemas de Controle Baseada no Microcontrolador 8051" Tese de Mestrado – Universidade Federal de Pernambuco- DES – Depertamento de Eletrônica e Sistemas, 1998.