SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL
FACULDADE DE TECNOLOGIA SENAI CIMATEC
CURSO SUPERIOR DE TECNOLOGIA EM MECATRÔNICA INDUSTRIAL

PROJETO DE UM DISPOSITIVO PARA MEDIÇÃO AUTOMATIZADA DE TEMPERATURA APLICADA A ESTAÇÃO DE TRATAMENTO SUPERFICIAL DA XEROX

Desenvolvimento dos *Firmwares*

Salvador
2008

ANDERSON ALMEIDA DE MACÊDO CARVALHO

PROJETO DE UM DISPOSITIVO PARA MEDIÇÃO AUTOMATIZADA DE TEMPERATURA APLICADA A ESTAÇÃO DE TRATAMENTO SUPERFICIAL DA XEROX

Desenvolvimento dos *Firmwares*

Trabalho de conclusão de curso submetido como requisito para aprovação no Curso Superior de Tecnologia em Mecatrônica na Faculdade de Tecnologia Senai-Cimatec.

Orientador: Msc. Cleber Vinícius Ribeiro de Almeida

Salvador
2008

Ficha catalográfica elaborada pela Biblioteca da Faculdade de Tecnologia

SENAI Cimatec

Carvalho, Anderson Almeida de Macedo.

Projeto de um dispositivo para medição automatizada de temperatura aplicada a estação de tratamento superficial da Xerox: desenvolvimento dos firmwares / Anderson Almeida de Macedo Carvalho. - Salvador, 2008.

82f.

1. Microcontroladores. 2. MSP430

CDD 621.3815

AGRADECIMENTO

Agradeço aos meus pais, minha irmã e minha namorada que estiveram ao meu lado durante esses quase quatro anos de estudo. Reconheço a qualidade de ensino da unidade SENAI Cimatec por ter oferecido a mim, e aos meus colegas, suporte e estrutura para a formação de excelentes profissionais.

A equipe de pesquisa deste projeto, os alunos Ulysses e Matheus, que juntos conseguimos promover um trabalho destaque na instituição. Os nosso esforços culminaram na aprovação do projeto para a Mostra Inova SENAI 2008 – Etapa Nacional.

A XEROX, unidade de Camaçari, que gerou esta oportunidade de trabalho onde foi possível aplicar os conhecimentos adquiridos no curso em uma situação de um problema real. Em destaque a Antonio Carlos, Marcelo Novaes, Sato, Daniel Doria e toda equipe de processo.

Na realização deste trabalho, sempre estará presente em minha memória o apoio do meu orientador Cleber e a ajuda da equipe do Núcleo de Microeletrônica, dos professores Milton e Marcelo Trautmann que orientaram este projeto nos seus outros dois segmentos. Agradeço aos professores do Cimatec Hernane Pereira, Lynn Alves, Karla Vittori e Marcelo Ueki e os da unidade do Cetind Jadson Aragão e Ildefonso. Todos estes acompanharam, cada um em sua área, o nosso trabalho. Em especial, quero lembrar dos professores Alexandre Ribeiro e Milton Bastos “Black” (mais uma vez) que apostaram e incentivaram a evolução da nossa turma.

Por fim, meus agradecimentos a todos meus colegas e amigos de turma, no qual formamos uma quase família neste curso, muito obrigado.

“When you lose small mind, you free your life“

(Serj Tankian, 2001)

RESUMO

Este trabalho visa criar uma lógica de programação para ser inserida em um projeto conjunto de três alunos do Curso Superior de Tecnologia em Mecatrônica Industrial. O objetivo em questão é de automatizar o processo de tratamento superficial dos Condutores Foto orgânicos ou OPC, do inglês *Organic Photo Conductor*, um dos produtos manufaturados da unidade XEROX de Camaçari - BA. Para este protótipo, dois *firmwares* distintos serão desenvolvidos: um embarcado em num transportador de OPC's e o outro em uma estação fixa. O primeiro tem como papel principal de criar um banco de dados com os sinais oriundos de vários sensores de temperatura e, posteriormente, enviar as informações para a estação fixa. Nesta, outro *hardware*, desenvolvido pela equipe, possuirá um microcontrolador que irá receber o banco de dados do processo e transferi-los à um computador pessoal da empresa.

Os dois programas desenvolvidos são distintos e necessitam de conhecimentos complexos. O sincronismo nas duas etapas de comunicação é um ponto muito relevante, pois proporcionará segurança e confiabilidade ao equipamento.

A última etapa do projeto contempla a implementação de um *software* que receba o banco de dados do processo e os apresente à engenharia da referida empresa, para que essas informações sejam avaliadas.

Atualmente, não existe nenhum método preventivo que acompanhe este processo da XEROX. A solução para este problema acarreta em um ganho considerável para a empresa como, por exemplo, na redução de paradas de manutenção. Um equipamento para a medição automatizada dos OPC's no processo tem como impacto ganhos significativos na produção e no processo.

Palavras Chaves: *Armazenamento de dados, Desenvolvimento de Firmwares, Supervisório com P.O.O., Comunicação wireless.*

ABSTRACT

This work aims to create a logic of programming to be included in a joint project of three students from Course of Mechatronics Industrial Technology. The object in question is to automate the process of surface treatment of Organic Photo Conductor or OPC, one of the products manufactured at unit XEROX of Camaçari - BA. For this prototype, two separate firmware will be developed: a embedded into a carrier of OPC's and other in a fixed station. The first one has the lead role to create a database with the signals from various temperature sensors and then send the information to station fixed. In this, other hardware, developed by the team, have a microcontroller that will receive the database of the process and transfer them to a personal computer of the company.

The two programs are different and require a complex knowledge. Timing the two stages of communication is very important because it will provide the reliability and safety equipment.

The last stage of the project includes the implementation of a software that receive the database of the process and provide the engineering of this company, for which that information is evaluated.

Currently, there is no preventive method to monitor this process of XEROX. The solution to this problem carries on a considerable gain for the company, such as the reduction of downtime for maintenance. An automated equipment for the measurement of OPC's in the process is to impact significant gains in production and in the process.

Keywords: *Data storage, Development of Firmwares, Supervisório with POO, wireless communication.*

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

OPC	Organic Photo-conductor (Condutor foto-orgânico)
XENOR	Unidade XEROX Nordeste
POO	Programação Orientada Objeto
PWM	Pulse Width Modulation (Modulação por largura de Pulso)
A/D	Analógico/Digital
CI	Circuito Integrado
TI	Texas Instruments
RAM	Random Access Memory (Memória de acesso aleatório)
ROM	Read Only Memory (Memória apenas de leitura)
RISC	Reduced Instruction Set Computer (Conjunto reduzido de instruções computacionais)
CISC	Complex Instruction Set Computer (Conjunto complexo de instruções computacionais)
CPU	Central Processing Unit (Unidade central de processamento)
CTL	Charge Transport Layer (Camada de transporte de carga)
CGL	Charge Generation Layer (Camada de geração de carga)
UCL	Undercoat layer (Camada de proteção interna)
JTAG	Joint Test Action Group (Grupo de teste de ação conjunta)
FET	Flash Emulation Tool (Ferramenta de emulação da FLASH)
USART	Universal Synchronous and. Asynchronous Receiver Transmitter (Transmissor/ receptor universal síncrono e assíncrona)
MCU	Microcontroller computer unit (Unidade computacional microcontrolada)
MIPS	Million instructions per second (Milhões de instruções por segundo)

SUMÁRIO

1. INTRODUÇÃO

A medição de temperatura é uma das mais importantes operações em um processo industrial. Os métodos atuais para controle e monitoração desta variável são fundamentais em quase todos os ramos do conhecimento humano devido à sua importância na maioria dos processos físicos, químicos e biológicos. Sem contar que a temperatura é uma grandeza crítica a ser considerada no que tange a proteção de pessoas e equipamentos. (BEGA, 2006; NETO, 2006)

Além da umidade do ar e da velocidade de cada estágio do processo, a temperatura é uma das principais variáveis no processo de fabricação contínua dos condutores foto orgânicos, ou OPC's (uma abreviação do inglês, Organic Photo-Conductor). A XEROX Comércio e Indústria Ltda, unidade de Camaçari – BA possui em sua lista de produtos estes fotorreceptores orgânicos. A monitoração da temperatura interna dos cilindros base nesta linha de produção é um procedimento que reflete em maior qualidade no produto final. (FIEB, 2007)

O fator crítico na fabricação do OPC deriva do alto nível de precisão ao se sobrepor em camadas delgadas de substâncias líquidas orgânicas. Estas camadas podem chegar a ter espessuras 500 vezes menores que um fio de cabelo humano de diâmetro médio, 50 mícrons. E em um processo contínuo deste produto, como é no caso da XEROX, variações mínimas na temperatura nas etapas de banho, secagem ou de aquecimento podem comprometer a funcionalidade de um lote inteiro de OPC's, chamado de *batch*. (OBINATA, TERASAKI e YUTAKA, 2006; PALHARES, 2002)

Este projeto está voltado principalmente à solução deste problema. Na planta da XENOR (unidade Nordeste da XEROX), a Seção B é a responsável por realizar todo o tratamento superficial de tubos perfilados de alumínio, base para se fabricar o OPC. Dentro desta seção, uma das etapas mais críticas é a dos banhos orgânicos, onde são colocadas as finas camadas descritas anteriormente. Sabendo da importância de se monitorar melhor a temperatura neste processo, o estudo em questão pretende criar um equipamento que viabilize a medição discretizada e automatizada da temperatura no interior dos OPC's durante a etapa de banhos no processo nesta Seção B. (FIEB, 2007; XEROX, 2007)

Por se tratar de um projeto do âmbito mecatrônico, o trabalho foi dividido em três subsistemas com características específicas: mecânica, eletrônica e programação e controle. Outra justificativa para esta divisão foi a complexidade que envolve o projeto, principalmente por estar sendo desenvolvido em um curso de graduação. As três ramificações deste trabalho são:

- A solução mecânica e de sensoriamento; (SILVA, 2008)
- Projeto e simulação do *hardware*; (SOARES, 2008)
- Desenvolvimento dos *firmwares* (a que este documento tratará).

A estrutura do projeto, como um todo, foi traduzida em um mapa conceitual (Figura 1) para aumentar o entendimento do leitor a respeito da situação problema que envolve este trabalho. O mapa conceitual é uma técnica ou um instrumento bastante abrangente, que pode ser aplicado em diferentes circunstâncias no intuito de proporcionar uma melhor análise da situação. (SOCORRO, 2005)

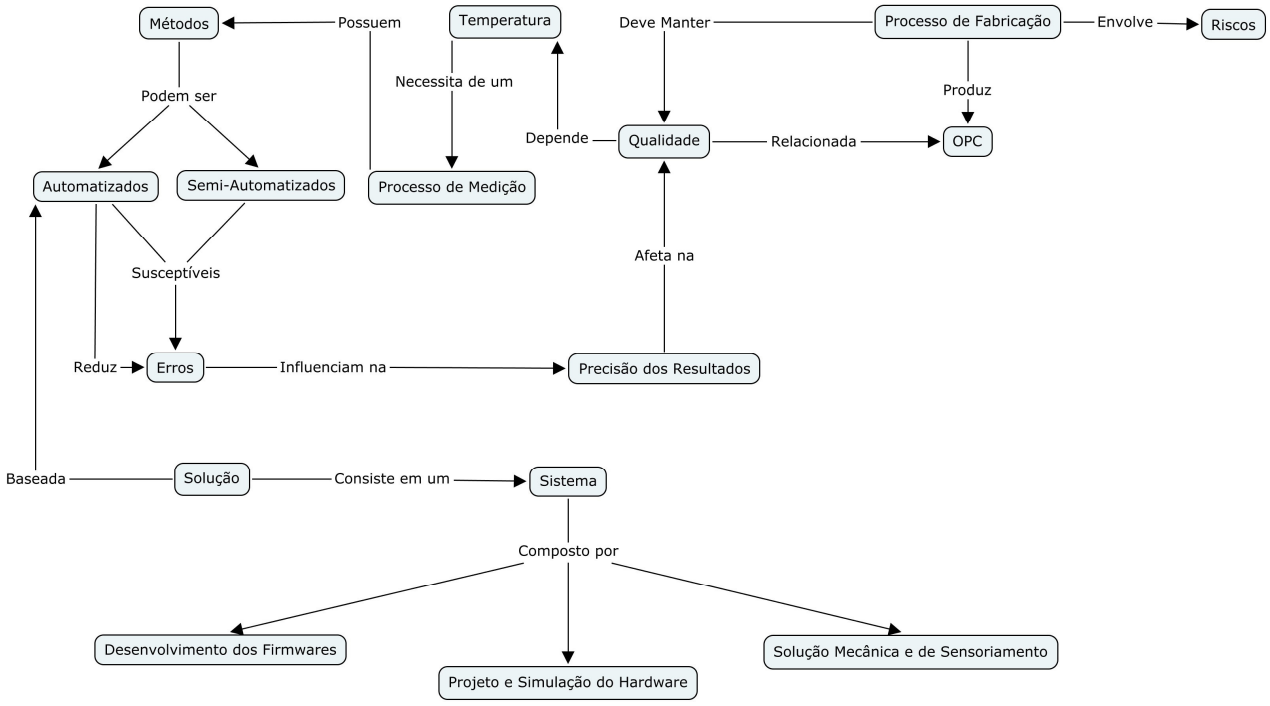

Figura 1. Mapa conceitual geral do projeto

Pode-se abstrair da Figura 1 a idéia principal do projeto. De forma geral, o trabalho objetiva uma solução automatizada no processo de medição de

temperatura na produção dos OPC's. Estes fotorreceptores são afetados por variações mínimas desta variável em uma de suas linhas de produção (Seção B). Fica explícito também a ligação entre a qualidade do produto final com a estabilidade térmica no processo de fabricação, deixando claro ou justificando este estudo aplicado. Na parte inferior do mapa apresentado, observa-se que este é um trabalho que está sendo realizado em conjunto com outros dois alunos do Curso Superior em Mecatrônica Industrial.

1.1. Descrição do Problema

Um dos componentes da visão estratégica da XEROX é a "liderança, através da qualidade". Com isso, fica claro que, para a empresa, a qualidade de seus produtos é um ponto primário. A unidade de Camaçari – BA produz, além de prestar serviços na região, o tonalizador e o fotorreceptor orgânico para copiadoras e impressoras. Neste último produto, uma etapa crítica para a sua produção possui algumas deficiências. A funcionalidade do OPC depende principalmente da espessura das camadas orgânicas que são sobrepostas sobre uma base tubular de alumínio. E para que isso seja realizado com eficiência, a superfície de cada cilindro deve permanecer em faixa de temperatura controlada sobre uma tolerância mínima. (FIEB, 2007; XEROX, 2007)

Atualmente, o único meio de monitorar ou diagnosticar algum problema na etapa de tratamento superficial da linha de produção dos OPC's é com auxílio de um método não preditivo. A solução atual foi a adaptação de um *Carrier* (veículo próprio para transportar 32 OPC's) para transportar uma placa de aquisição (*Dataloger*) com 7 termopares do tipo K. Estes sensores são fixados na parte externa dos tubos de alumínio com fitas isolantes comuns.

Por este motivo, nas principais etapas na Seção B, os banhos em substâncias orgânicas não são realizados, para não molhar os sensores. Pode-se concluir que a alternativa vigente não é muito eficiente, por não fornecer dados completos do processo. Sem contar que este procedimento apenas é realizado quando um grande número de fotorreceptores danificados são rejeitados na linha seguinte da planta, na Seção C. E mesmo não existindo um método preventivo de evitar alterações nas

temperaturas do processo, quando ele é utilizado, existe ainda a necessidade de paralisar a planta e abrir um das câmaras da Seção B. Conclui-se que o método atual pode ser considerado deficiente e impróprio para a situação. O principal fato é que, ao parar a produção por aproximadamente 2 horas, a XEROX perde tempo e, consequentemente, dinheiro.

Ao final deste procedimento, na passagem do "*Carrier* instrumentado" pelo processo são descarregados para um computador por uma interface serial RS-232. Um *software* proprietário, do mesmo fabricante da placa de armazenamento do *Dataloger*, consegue ler o banco de dados criado com as temperaturas do processo e apresenta-se ao usuário no formato de um gráfico em função do tempo.

O engenheiro de processo responsável por este setor consegue assim ter acesso às informações necessárias para avaliar e reconfigurar o sistema de controle da planta. Mas o fato do equipamento e do *software* de supervisão serem de uma empresa especializada, o algoritmo de gerenciamento da placa de aquisição de dados e o próprio programa de interface no computador são fechados. Qualquer alteração no projeto, modificação ou manutenção no equipamento deve ser realizada estritamente pela empresa que fabricou o dispositivo. Para a XEROX, isto não é muito interessante, pois além de gerar custo na manutenção, deixa o módulo parado até que o possível problema seja sanado.

O ideal era que houvesse um monitoramento periódico desta variável, contudo isto não existe. E uma justificativa observada em visitas á empresa é de o controle inserido nas câmaras do processo porta de uma eficácia suficiente para se produzir com sucesso, apesar das perdas existentes.

Provavelmente, distúrbios não previstos provocam alterações no sistema e pequenas avarias na superfície do OPC, o que o torna "não-comercializável". Esses problemas, atualmente, apenas são percebidos na linha seguinte do processo, denominada Seção C. E até que o defeito seja identificado pela primeira vez, vários OPC's já deixaram a estação de tratamento superficial e, consequentemente, muitas outras peças foram danificadas e perdidas.

A partir de todos estes inconvenientes, este projeto propõe solucionar o problema com o desenvolvimento de um dispositivo capaz de medir a temperatura na superfície interna dos OPC's de forma automatizada. Espera-se atingir este

objetivo a partir de:uma solução conjunta de: mecânica e sensoriamento, um novo projeto de *hardware* e desenvolvimento de *softwares* (embarcados para gerenciar a nova placa de aquisição e outro para monitorar os dados coletados no computador).

1.2. Justificativa

Diferente de temáticas mais usuais de trabalhos finais de graduação, este tema proposto apresenta algumas peculiaridades que promovem motivações, tornando o projeto, de certa foram, interessante.

O crescimento profissional é um fator considerável neste trabalho. A oportunidade de estar realizando um projeto para uma multinacional como a XEROX Comércio e Indústria Ltda. traz uma satisfação enorme para um aluno de graduação. Além de promover um direcionamento dos estudos a um campo real e aplicável em um ambiente industrial (o foco do curso), o potencial do profissional-projetista está sendo apresentado diretamente a diretores e supervisores da empresa.

A aceitação deste projeto por parte do autor justifica-se também pelo desafio que envolve a questão: desenvolver um equipamento em conjunto com dois colegas em um ambiente de características singulares. Não há duvidas que este trabalho enriquecerá e ampliará o leque de conhecimento deste pesquisador.

A possibilidade de visitar, conhecer com detalhes uma linha de produção e ter todo o auxílio e apoio necessário de um supervisor de uma indústria de grande porte é uma chance única. Poucos estudantes de graduação têm a oportunidade de ter um convite como este. É um excelente momento para por em prática todo o conhecimento adquirido nas aulas do curso. Com certeza, o crescimento e o amadurecimento profissional dos alunos que abraçaram o projeto serão inigualáveis.

1.3. Objetivo Geral

O objetivo geral deste trabalho é desenvolver a lógica para o controle embarcado do projeto de um dispositivo que efetue a medição automatizada da temperatura na superfície interna dos OPC’s contidos em um lote de produção (na

empresa é chamado também de *batch)*. Contempla-se também criar um sistema de supervisão que permita ao engenheiro responsável por este setor, analisar e monitorar as variações desta grandeza física mensurada.

Os objetivos específicos deste estudo consideram as necessidades de:

a) Especificar as necessidades a cerca da funcionalidade do protótipo junto ao cliente, neste caso, à XEROX;
b) Analisar e estudar o *software* existente para o processo atual de medição de temperatura;
c) Compilar o projeto conceitual levantado e criar um descritivo geral para o novo projeto que será implementado;
d) Formalizar os protocolos de comunicação que serão utilizados para as duas transferências de dados existentes no projeto;
e) Construir e estruturar a lógica para os *softwares* que serão embarcados nos projetos *Carrier* e Estação;
f) Escrever os *firmwares* para os microcontroladores na linguagem de programação escolhida;
g) Simular e testar computacionalmente os programas desenvolvidos;
h) Criar um *software* supervisório para que possa substituir o que existe atualmente;
i) Propor e efetuar testes reais com todos os subsistemas do projeto interligadas, a fim de comprovar e validar o trabalho.

1.4. Impactos Esperados

Neste projeto, busca-se desenvolver um dispositivo de medição de temperatura automatizado que diminua a intrusividade ao processo. Como foi já foi dito, o método atual utilizado para medir a temperatura necessita que se pare a linha de produção. O procedimento em questão não é adequado, pois nessas paradas a empresa XEROX sofre grande prejuízo, já que esta possui uma linha de produção contínua e seqüencial.

O equipamento desenvolvido visa sanar este problema, proporcionando um ganho financeiro à XEROX, pois agora, com um monitoramento periódico será

realizado na linha em funcionamento. Deste modo, um problema pode ser corrigido antes que afete o lote de produção completo ou até o *batch* seguinte da linha.

Além destes pontos, espera-se o desenvolvimento de uma nova tecnologia que não existe em nenhuma das plantas da XEROX, o que converte em uma idéia que este seja um problema em todas as unidades da corporação na fabricação do produto.

1.5. Metodologia

A Metodologia a ser utilizada para desenvolvimento do projeto de desenvolvimento de um dispositivo que efetue a medição e acompanhamento de temperatura na estação de tratamento superficial (Seção B) do processo produtivo da XEROX será a metodologia THEOPRAX, que visa como principal objetivo solucionar um problema existente no meio industrial. (THEOPRAX, 2005)

Esta metodologia foi desenvolvida na Alemanha pelo instituto FraunHofer por Doerthe Krause e Peter Eyerer e é novidade no Brasil. Ela busca a integração da indústria com uma faculdade para que os alunos possam aplicar seus conhecimentos em um trabalho real. Desta forma, eles estarão mais preparados para o mercado com uma experiência prática em seus currículos. Para a empresa é interessante essa parceria para conseguir uma solução que não tenha os vícios do ambiente industrial. (THEOPRAX, 2005)

Os alunos que participam desta metodologia passam por um curso de aperfeiçoamento na área de projetos para que se estruturem melhor. Eles recebem uma noção em gerenciamento de projetos, empreendedorismo, apresentação pessoal e outras competências. Após este treinamento, o grupo de trabalho desenvolve uma proposta de projeto completa e a apresenta para os gerentes e responsáveis da solicitante do serviço. (THEOPRAX, 2005)

Com o THEOPRAX, o aluno irá passar por procedimentos de desenvolvimento que eles irão se deparar na sua vida profissional. Esta metodologia ajuda o aluno a estar mais preparado para enfrentar os problemas em uma empresa. (THEOPRAX, 2005)

1.6. Organização do Documento

Para uma melhor compreensão do leitor sobre os detalhes do projeto, a estrutura deste texto foi montada para que se possa entendê-lo com maior clareza. Todos os tópicos levantados foram profundamente detalhados proporcionando objetividade ao trabalho proposto

Uma divisão estratégica do manuscrito foi feita a fim de contextualizar melhor o assunto. Este **primeiro capítulo** teve com finalidade introduzir ao leitor a situação problema que justificou a realização deste estudo.

No **segundo capítulo** do documento, o ambiente de trabalho no qual será inserido o projeto foi devidamente descrito no intuito de esclarecer todas as particularidades do problema apresentado. Este capítulo é importantíssimo para que o leitor possa entender as etapas da criação de um modelo a fim de alcançar o objetivo final deste estudo. É abordado neste texto: o descritivo do processo de fabricação e de medição utilizada, as características intrínsecas de uma atmosfera explosiva e uma breve explanação dos três subsistemas que constituirão o dispositivo a ser desenvolvido.

No **terceiro capítulo**, foi documentado a revisão bibliográfica realizada e necessária para fundamentar e iniciar a construção dos alicerces do projeto. Todos os campos científicos que envolvem este trabalho foram devidamente detalhados e explicados, enfatizando principalmente as ferramentas utilizadas no processo.

O **quarto capítulo** relata, com riqueza de detalhes, todo o processo de desenvolvimento do projeto. A funcionalidade e o objetivo de cada um dos algoritmos implementados são descritos com auxilio de fluxogramas para se torne mais claro a lógica criada. No que se refere ao *software* para monitorar os dados coletados no processo, as telas e funções inerentes a este programa são apresentadas e devidamente justificadas baseado-se nas necessidades do cliente.

O **último capítulo** finaliza projeto realizado com as conclusões deste trabalho, onde sugestões de melhorias na continuação da pesquisa são sugeridas. E

por fim, segue as referências utilizadas nesta pesquisa e um anexo com o conjunto de instruções do microcontrolador estudado.

2. ESPECIFICAÇÕES DO PROJETO

O projeto em questão está sendo desenvolvido de forma aplicada a um processo de uma grande empresa, que é a XEROX Comércio e Indústria Ltda. Para melhor entendimento do leitor do contexto geral do trabalho e como o tipo de ambiente dificulta uma solução trivial é apresentada.

Neste Capítulo é descrito o escopo do projeto e quais são os subsistemas que constituem o protótipo a ser projetado. Finalmente, é enfatizado as etapas deste trabalho, e quais as funcionalidades dos programas que serão desenvolvidos.

2.1. Processo de Fabricação do OPC na XEROX

A linha de produção de fotorreceptores óticos da XEROX é composta por três Seções: A, B e C. A primeira etapa do processo e é responsável por construir a base do produto, um substrato condutor de alumínio. Perfilados deste metal chegam a fábrica e são cortados por máquinas de usinagem CNC, para que tenham as dimensões exatas de comprimento. Atualmente, o diâmetro deste material é único (20 mm). Contudo, existem projetos de mudança na produção por existir variações deste produto entre os modelos de fotocopiadoras.

Na Seção C, são realizados os procedimentos de montagem, inspeção, testes de funcionamento, e de qualificação do produto final. Esta é uma linha totalmente automatizada e integrada que conta até com inúmeros robôs para manipulação, células de montagem e um controle de qualidade que conta com inspeção visual e digital com auxilio de câmeras CCD/CMOS.

Um lote de 32 OPC's, ou *batch* como é chamado, chega a esta parte do processo através de uma esteira oriunda da Seção B em um transportador com suporte para os tubos (Figura 02).

Figura 2. Transportador na Seção C

Um primeiro robô descarrega estes tubos para uma esteira individual onde os mesmos circulam deitados. Uma espécie de código de identificação é usinada em cada um dos tubos. Uma máquina com um *laser* muito fino e preciso escreve na parte superior do cilindro um código de barras que irá garantir a identificação e a rastreabilidade do produto (Figura 03). Esta é uma operação muito delicada e pode ser comprovada pela dificuldade que uma pessoa tem ao visualizar esta marcação. Na Figura 16, o círculo vermelho indica o local da gravação. Para que se tenha uma noção de escala, esta estreita faixa cinza, onde não houve tratamento superficial, possui uma largura em torno de três milímetros.

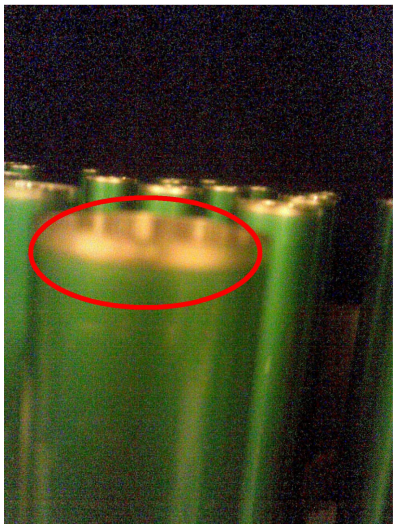

Figura 3. Indicativo do local de gravação nos OPC's

Duas células de montagem se encarregam de fixar um tipo de engrenagem de plástico em cada uma das extremidades do OPC. Estas peças determinam qual o modelo de equipamento em que o produto será empregado, já que existem variações nos tipos de encaixes entre as fotocopiadoras. Na Figura 04, um OPC foi cortado para que se visualize a fixação de uma das engrenagens no tubo.

Figura 4. Engrenagem de fixação no OPC

Na conclusão do processo de montagem final, testes de isolação elétrica, condutividade (em pontos diferentes) e de acoplamento são realizados. Uma etapa altamente criteriosa inspeciona minuciosamente a superfície dos OPC's. Duas inspeções humanas ocorrem em locais diferentes da linha e mais uma certificação é realizada através de câmeras digitais. As imagens dos fotorreceptores são capturadas e digitalizadas para ser comparados com padrões de peças com boas condições de uso armazenadas em uma unidade de memória. Esta etapa de qualidade é crítica, pois falhas mínimas na cobertura orgânica comprometem a funcionalidade do produto. Caso um produto danifica seja comercializado, o nome da empresa é prejudicado. A linha de produção é finalizada com ao se embalar os OPC's para que sejam distribuídos no mercado.

2.2. Planta de tratamento superficial

A Seção B é a planta de tratamento superficial dos OPC's na XEROX. Ela é a principal em todo processo uma vez que é nela que simples tubos metálicos recebem a funcionalidade real nesta seção. Além disso, a Seção B possui características de fabricação singulares. Esta linha é dividida em duas etapas, a primeira recebe os tubos em estado bruto da Seção A e os prepara para a segunda etapa. É realizada a limpeza e a higienização de todo alumínio e operações que irão facilitar a fixação das camadas orgânicas.

Na segunda etapa da Seção B, os cilindros de alumínio já estão preparados e terão que passam pelo processo de tratamento superficial realmente. Um robô de três eixos captura cinco ou quatro tubos e abastece uma espécie de transportador chamado de *Carrier.* Esta operação inicial é mostrada a esquerda na Figura 05.

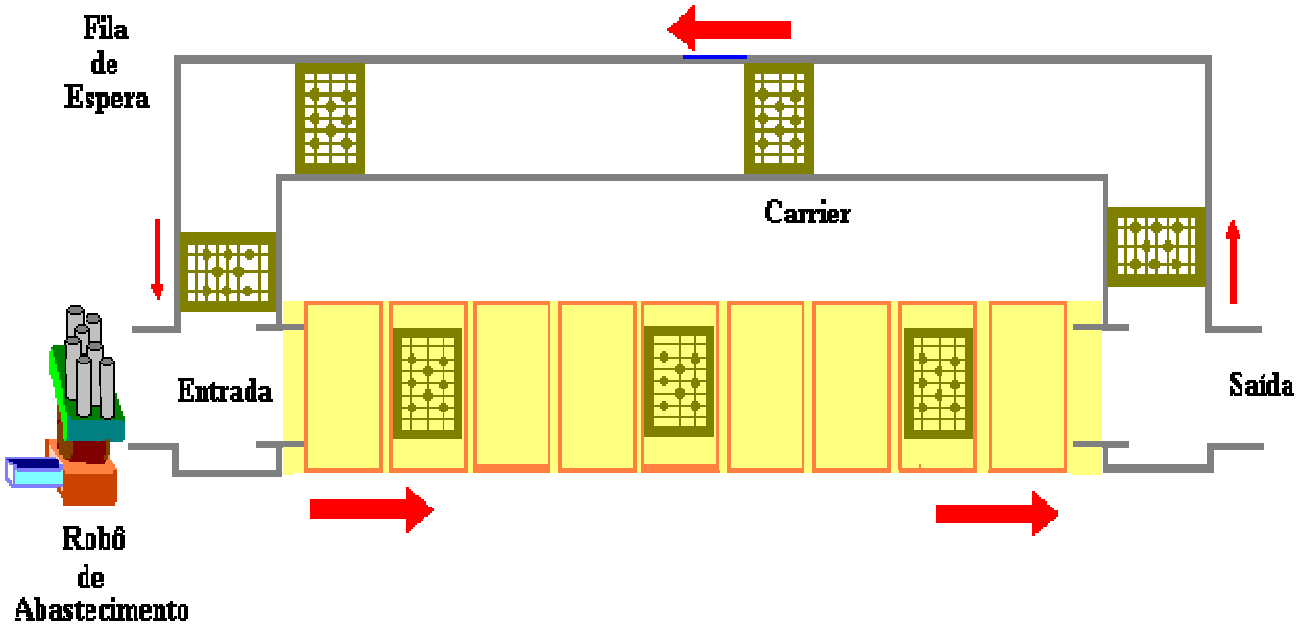

Figura 5. Processo de Fabricação XEROX (Seção B)

O *Carrier* abastecido segue por uma esteira única para um grande equipamento que é subdividido internamente em inúmeras partes. Cada uma destas etapas trabalha de forma seqüencial para que se possa realizar o processo de

sobreposição das camadas orgânicas. Na Figura 05, o trecho amarelo representa graficamente estas operações.

Este grande equipamento é visto na Figura 06, e se estende por cerca de 10 metros ao total. Também pode ser observado que existem vários compartimentos onde se alternam:

- Etapas de fornos aquecimento em temperaturas elevadas (algo em torno de 160 ℃);
- Secadores por ventilação forçada para o resfriamento;
- Processos de banhos em substâncias orgânicas.

Estas divisões seqüenciais também foram representadas, simbolicamente, na Figura 05.

Figura 6. Parte do equipamento principal da Seção B

Na totalidade, são três grandes salas divididas internamente em diferentes câmaras. Cada uma das salas é responsável por cobrir o tubo com um tipo de camada. Todos os três processos funcionam de modo semelhante, onde primeiramente um a superfície do OPC é preparada termicamente, para que depois se inicie o mergulho em uma das 3 sustâncias. Os banhos nos 32 OPC's são

realizados em pequenos compartimentos individuais, porém em um fluído re-circulante comum. Esta espécie de matriz para os banhos, com as divisões para a inserção de cada tubo é mostrada na Figura 07. Note a existência de 32 cavidades e entre elas canais para que o fluído escorra.

Figura 7. Matriz onde se realiza os banhos no OPC

Estes banhos são etapas bastante complexas, pois para um fotorreceptor funcionar ele precisa de camadas bastante delgadas de cada substância. A espessura delas varia a depender de qual seja. A Figura 08 mostra os três marteriais sobre postos, são elas: a CTL ou *Charge Transport Layer (Camada de Transporte de Carga), a CGL ou Charge Generation Layer (Camada de Geração de Carga) e* a UCL ou *Undercoat layer (Camada de Revestimento Interno).*

Cada uma destas camadas possui funções bastante especificas para transferir a imagem latente do papel que se desejar copiar em uma máquina fotocopiadora. É um processo contínuo em que o OPC se carrega eletricamente e se descarrega com a transferência do toner para a folha de papel.

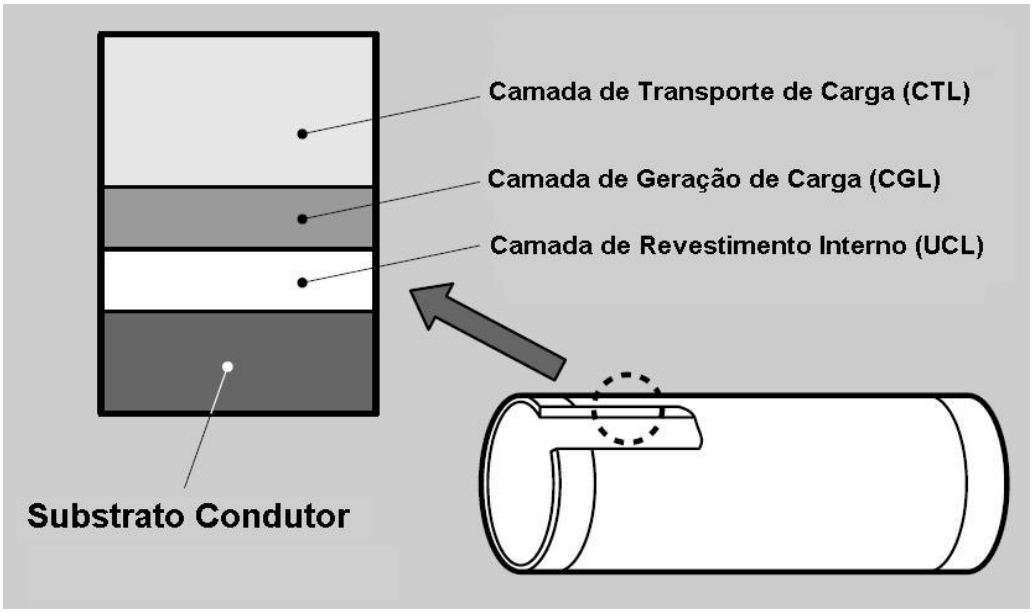

Figura 8. Camadas do OPC

O substrato de condutor alumínio proporciona a fotocondutividade em termos físicos e elétricos, mas não possui uma figura ativa no processo eletrofotográfico. A sua principal função é, realmente, oferecer suporte estrutural e mecânico, e uma boa conexão de aterramento elétrico. É sobre esta base que é colocada a primeira camada orgânica, o UCL responsável por um papel de isolação e bloqueio. Este revestimento é uma interface entre o substrato e as outras camadas fotocondutoras evitando assim vazamentos de carga além de fornecer uma adesão sólida.

A camada de geração de carga, ou CGL, é a mais fina de todas, sua espessura típica que varia de apenas 0,1 mícron até 1,0 mícron. A sensibilidade à luz do CGL é um fator essencial no desempenho do tambor OPC, e pode ser um fator limitante de velocidade de cópias na qual o produto pode funcionar eficientemente. Uma curiosidade é que a cor do foto receptor é dada pelos elementos que compõem esta camada.

A camada de transporte de carga (CTL) é a camada externa de um tambor OPC cuja espessura é de 20 a 30 mícrons. Ela é essencialmente transparente, permitindo que a luz passe diretamente através da camada de geração de carga. Da mesma forma que a CGL determina, basicamente, a sensibilidade à luz de um tambor OPC, a CTL define a sua capacidade de aceitação, velocidade e de transporte de carga. Como esta é a camada mais externa, ela entra em contato com o toner, o revelador, o papel, a lâmina de limpeza do tambor, o ozônio e outros

agentes potencialmente abrasivos, ou mesmo, contaminantes. Conseqüentemente as características de desgaste desta camada, sua durabilidade e resistência à abrasão são fatores essenciais no potencial vida útil de um tambor OPC.

O processo para a sobreposição destes revestimentos é composto por etapas bastante complexas, pois para um fotorreceptor funcionar ele precisa de camadas bastante delgadas de cada substância. A questão da precisão nas espessuras, como dito, é um detalhe crítico e só é possível através de um controle totalmente avançado no processo.

Em todas as câmaras da Seção B, é extremamente necessário que haja um controle: na variável temperatura ambiente e na superfície de cada tudo para garantir a homogeneidade do processo, nas velocidades de mergulho e de retirada do *Carrier* em cada um dos banhos. Lembrar que a aderência precisa ser perfeita sem nenhuma descontinuidade.

2.2.1. Medição de temperatura atual

Caso alguns dos tubos de alumínio variem muito sua temperatura, em pontos diferentes, na superfície um processo de dilatação ou contração pode desestruturar o desenho perfeitos das camadas. Este fato provoca inúmeras rejeições da etapa de qualidade da XEROX.

Não existe um método preventivo de monitoração da temperatura diretamente nos tubos de alumínio. Contudo, devido ao excelente controle das salas e dos fluídos de banho o efeito negativo relatado é minimizado consideravelmente.

O que a fábrica dispõem de mais próximo de um dispositivo para a monitoração no intuito de evitar este problemas é o MOLE (Figura 09). Este foi desenvolvido pela própria fábrica e permite a instalação de até 6 sensores de temperatura. O equipamento conta ainda com um *Dataloger* montado por um terceiro, a OMEGA. (OMEGA, 2003)

Figura 9. Sistema de medição atual, o MOLE

Este sistema só é utilizado quando o número de OPC's rejeitados aumentam muito na Seção C. Os sensores de temperatura, são fixados nos tubos de alumínio e os sinais são lidos periodicamente pela placa de aquisição da OMEGA (Figura 10). Após cerca de duas horas e meia (tempo total do processo na Seção B), o MOLE é retirado da linha e os dados armazenados da temperatura no processo são descarregados no comutador pessoal do engenheiro responsável pela área. Avaliando os resultados, são propostas medidas a serem e realizadas (caso aprovadas). Estas decisões podem ser: mudanças das constantes de controle e/ou a velocidade do *Carrier* em cada câmara, por exemplo.

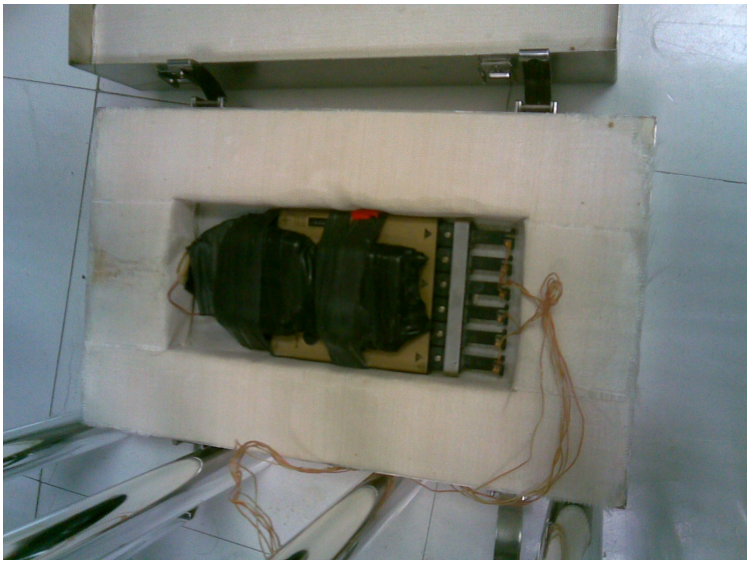

Figura 10. *Dataloger* com a placa de aquisição

2.2.2. Problemas Atuais

O MOLE não é nem de longe de ser um dispositivo profissional e eficiente para monitorar a temperatura do modo apresentado. Ele foi construído sobre um *Carrier* comum e os fixadores (os *chuck*) dos tubos de alumínio, do interior do transportador, foram substituídos pelo *Dataloger* (Figura 10) com uma placa de aquisição. Este pode ser considerado o primeiro problema deste dispositivo, pois mudando a massa de entrada no sistema, as trocas térmicas são alteradas.

Outra característica muito clara é o nível de improvisação do equipamento. Os sensores são fixados aos cilindros com auxílio de fitas adesivas. Isto impede que os banhos (uma das principais etapas) sejam realizados, para que os sensores não sejam molhados e danificados. Nestes estágios, o MOLE atravessa por cima da câmara como se não existisse. Desta forma, outra incoerência pode ser constatada: a não retração veraz do processo.

O mesmo artifício, relatado no parágrafo anterior, é utilizado para prender as baterias de alimentação do *Dataloger*. E para completar, a fiação dos seis termopares, que a placa de aquisição suporta, é enrolada livremente ao redor de um do *chucks.* Alguns destes detalhes que comprometem a eficiência deste método podem ser vistos na Figura 11.

Lembrar que o ideal é que a temperatura seja medida no interior do OPC, e MOLE realiza isto na parte externa. Baseando-se em dados estatísticos em cima do locais onde mais identificaram defeitos nos fotorreceptores rejeitados, se modifica o ponto de fixação dos termopares.

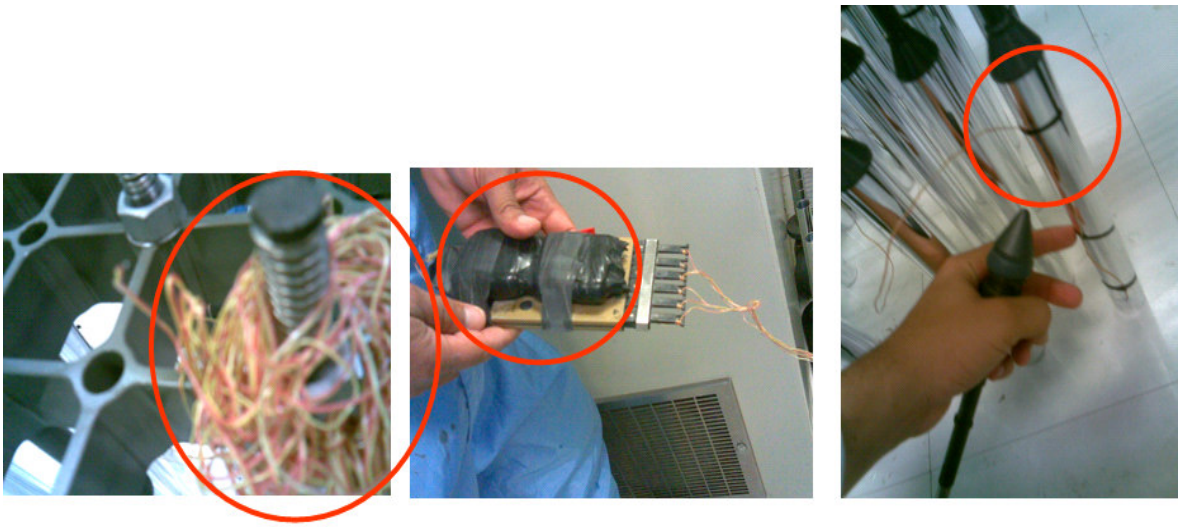

Figura 11. Pontos negativos no processo atual

Fica claro que este não é um método preventivo e sim um de diagnostico. Ele é utilizado apenas quando a empresa já rejeitou uma quantidade grande de produtos, ou seja, quando há prejuízo.

Outra grande inconveniência neste procedimento é o alto grau de intrusividade no sistema. Quando seu uso é necessário, é preciso parar a planta (esta Seção). Além disso, vale lembrar que o MOLE já está carregado, e pela linha estar paralisada, não pode ser inserido no processo no inicio (robô de abastecimento). Sua inserção é feita em uma das etapas intermediárias, abrindo-se manualmente uma das câmaras vista na Figura 05. O equipamento é retirado em uma das últimas câmaras, por um compartimento similar ao primeiro, de forma manual. Isso causa uma interferência grave do procedimento, que ao se abrir um dos fornos, por exemplo, a temperatura interna é totalmente alterada, não sendo mais a mesma que no processo de fabricação.

2.3. Subsistemas do Projeto

O **tópico 3.2.2**, descreveu os problemas existentes no método de diagnóstico na fábrica. E por sinal, automatizar este procedimento de forma que se permita monitorar o inconveniente de para a linha é o alvo do projeto geral. Todo o trabalho de desenvolvimento desse dispositivo de medição de temperatura automatizada consiste em um equipamento mecatrônico. E naturalmente, sua implementação foi dividida em três grandes partes: A solução mecânica e de sensoriamento, o Projeto e Simulação do *Hardware* e O desenvolvimento dos *Firmwares*.

2.3.1. Solução mecânica e de sensoriamento

Nesta etapa, serão criados todos os aspectos de medição da temperatura. Isto, no que inclui dimensionamento de sensores e modo de atuação. Por o ambiente é uma área classificada, essa etapa do projeto possui diversas restrições principalmente ao se orçar um sensor e o tipo de acionamento que levara este elemento a superfície interno do OPC. (SILVA, 2008)

Como em todo projeto de caráter mecatrônico, a etapa mecânica é importantíssima para o sucesso global. Pois nela, se gera uma máquina pouco flexível acarretando em muitas das principais definições que serão atribuídas na solução das outras partes. (SILVA, 2008)

Um dos principais desafios na solução mecânica é o projeto do atuador. Em uma zona onde o risco de explosão é contínuo, não é aconselhável o uso de motores elétricos de qualquer origem. Provavelmente a resposta para este problema é o aproveitamento do curso, já existente, de fixação dos tubos de alumínio *chuck*. Esta operação é feita nas etapas de carga e descarga do *Carrier.* (SILVA, 2008)

Este trabalho vai envolver um estudo maciço sobre mecânicas de precisão sem esquecer de se criar um projeto que desfavoreça altos níveis de manutenção futuras. (SILVA, 2008)

2.3.2. Projeto e Simulação do *hardware*

O objetivo desse trabalho é criar um *link* entre os dois outros projetos. Esta etapa é responsável por criar uma ligação dos sinais dos elementos sensores até o *software* de supervisão. (SOARES, 2008)

Dois circuitos serão construídos, o que será embarcado no *Carrier* e o da estação de transferência de dados. O primeiro condicionará os sinais de temperatura (33 no total) e os enviará de forma multiplexadas ao microcontrolador. Um módulo de armazenamento de dados, com o intuito de se expandir a memória do dispositivo, é também prevista neste trabalho. Uma interface para a transferência *wireless* dos dados será projetada. (SOARES, 2008)

O segundo circuito é responsável por receber a comunicação dos dados por rádio freqüência e direciona-las para o microcontrolador num sinal adequado. A outra interface, entre o MCU e um computador pessoal é também prevista.

Essa etapa do projeto necessita bastante das definições geradas pela Solução mecânica e de sensoriamento e fundamenta a estrutura para o último trabalho, o Desenvolvimento dos *Firmwares*. (SOARES, 2008)

2.3.3. Desenvolvimento dos *firmwares*

Esta etapa é a que este documento se refere. Nela, será desenvolvida a lógica de gerenciamento de dados em cima dos 2 *hardwares* confeccionados, além do software para a análise dos dados coletados.

Na primeira parte, o *firmware* irá armazenar periodicamente, em uma unidade de memória, os sinais dos 32 OPC's durante o processo de tratamento superficial do mesmo. E ao final da operação, estes dados serão enviados de forma ordenada via um circuito de comunicação *wireless.* Na segunda parte, esta informação será captada e passada para uma outra interface, mas esta será do tipo RS-232 (provavelmente). A saída final do projeto, ou melhor, o acesso humano ao projeto será feito por um software em um computador pessoal comum. O banco de dados com todas as leituras entrará por uma porta serial simples e serão apresentadas graficamente ao usuário. Assim, possíveis erros no processo podem ser analisados de forma mais clara e intuitiva.

3. FUNDAMENTAÇÃO TEÓRICA

3.1. Introdução

Para todo e qualquer tipo de projeto com cunho acadêmico tem previsto entre suas etapas de desenvolvimento um razoável período de pesquisa. Estudar os campos que circundam o trabalho desejado possibilita ao pesquisador um alicerce seguro para que ele ponha em prática suas idéias.

Identificar quais ciências envolve o projeto proposto é considerado por muitos cientistas o primeiro passo na concepção de um trabalho bem feito. É previsto no escopo deste projeto a criação de três códigos de programas. Dois serão embarcados em cada um dos *hardwares* projetados. E um outro que na verdade, terá a funcionalidade de mostrar de modo intuitivo um gráfico de tendências da temperatura a um engenheiro responsável do setor, por exemplo.

Para se construir a lógica destes três algoritmos, campos e ferramentas novas precisaram ser estudados. Um bom exemplo é que foi definido que seria utilizado um microcontrolador como os da família MSP430 que possuem menor consumo elétrico e uma tecnologia diferente das utilizadas por alunos de graduação em práticas de laboratório. Tudo para agregar ao protótipo desenvolvido um valor maior de profissionalismo. (PEREIRA, 2005)

Conhecimentos sobre Programação Orientada Objeto (POO) são essenciais para desenvolver um *software* que permita o usuário interagir com o mesmo. Na verdade, a POO é uma das ferramentas mais utilizadas para este tipo de aplicação. E como este conteúdo não foi devidamente explorado no curso, é clara a importância que haja uma fundamentação sobre este conteúdo.

Entender formas e protocolos de comunicação também devem ser estudados, pois estes serão utilizados para realizar transferências de dados entre controladores e entre um destes para um computador. Uma breve revisão sobre os meios físicos de comunicação e possíveis protocolos que possam ser implementados é totalmente válida.

Um trabalho de pesquisa sobre estes campos apresentados é, sem dúvidas, extremamente necessário. Um estudo maciço nestas áreas completamente distintas fundamenta e justifica cada decisão ou dimensionamento que envolve o trabalho. E com uma base de idéias, para o projeto, fortalecida teoricamente, fica muito mais fácil validar e comprovar o modelo criado (mais detalhes serão vistos no **Capítulo 5**) para solucionar a situação-problema.

3.2. Unidades microprocessadas

3.2.1. Breve Histórico

Os primeiros projetos eletrônicos da história tinham um inconveniente grave, que é a baixa flexibilidade em se realizar uma atualização para um novo produto. Em uma linha de produção, uma mudança era complicada já que os circuitos eram bastante específicos para um propósito. Nesta situação, era preciso mudar praticamente todo o processo de manufatura para se produzir um novo produto. (ANDRIC e MATIC, 2000; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002; SCHWAMBACH, 2004)

Na década de 70, uma grande empresa japonesa (BUSICOM) fabricava diversos tipos de calculadoras. Cada um de seus produtos tinha um *hardware* completamente diferente do outro. Ao receber uma proposta para diversificar sua produção, esta empresa teve que rever seus projetos. A proposta foi entregue à INTEL e Marcian Hoff, que possuía uma boa experiência em trabalho com computadores (o PDP8) e foi o responsável pela sua concretização. (ANDRIC e MATIC, 2000; JÁCOBO, 2001; PEREIRA, 2002; POLONSKII, 1996; SCHWAMBACH, 2004)

A solução para este problema deveria ser substancialmente diferente da que existia atualmente. Finalmente, em 1971, a equipe de Hoff desenvolveu um dispositivo revolucionário na época, o microprocessador i4004 (Figura 12). Este pequeno *chip* era um Circuito Integrado (CI) que podia ter sua funcionalidade modificada, pois seguia apenas as instruções estruturadas de um programa que

podia ser gravado no mesmo. (ANDRIC e MATIC, 2000; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002; POLONSKII, 1996; SCHWAMBACH, 2004)

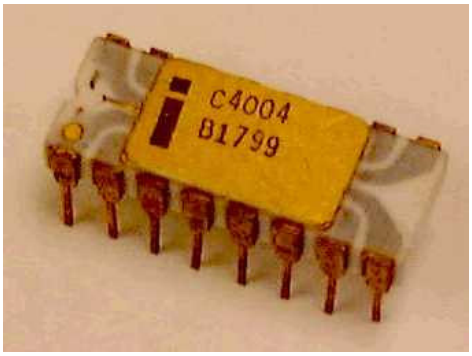

Figura 12. Primeiro microcontrolador, i40004

Após esta revolução, as calculadoras dos japoneses poderiam ser produzidas de maneira idêntica, no que diz respeito ao *hardware*. A diferença então de cada modelo seria dada pelo *software* inserido no CI, ou *firmware*. Os produtos deste fabricante teriam maior flexibilidade com custo de manufatura reduzido, pois apenas uma placa de circuito impresso precisava ser produzida agora. (ANDRIC e MATIC, 2000; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002; POLONSKII, 1996; SCHWAMBACH, 2004)

A partir deste marco, houve uma evolução considerável nas pesquisas sobre esta nova tecnologia. O aumento da velocidade de processamento de instruções e a complexidade das funções de cada *chip* crescem a uma ordem exponencial. De acordo com Gordon Moore (um dos fundadores da poderosa Intel) a capacidade de processamento e a número de transistores encapsulados em um *chip* deverá dobrar a cada 18 meses. Esta afirmação ficou conhecida com a “Lei de Moore”. A enorme versatilidade e o potencial dos microprocessadores permitiram que estes pudessem ser usados em cada vez mais tipos diferentes de sistemas. O surgimento destes componentes foi decisivo para o surgimento e popularização dos computadores pessoais. A Figura 13 mostra um Athlon 64 da AMD, um dos melhores processadores existentes, junto com o Pentium IV Core 2 Extreme XC6800 (Figura 14). (LIRA, 2006; POLONSKII, 1996)

Fonte: Cortesia da AMD

Figura 13. Athlon 64, um poderoso processador

Porém, o uso de microprocessadores tinha um inconveniente destes não serem muito flexíveis para aplicações específicas. E a tendência tecnológica da época era se desenvolver projetos embarcados que exigiam tais características. Isto promoveu a evolução dos *chips* que procuravam cada vez mais terem em um único encapsulamento com mais funções e utilidades. (ANDRIC e MATIC, 2000; JÁCOBO, 2001; PEREIRA, 2002; SCHWAMBACH, 2004)

Fonte: Cortesia da Intel

Figura 14. O X6800, o melhor em performance

Então, a *Texas Instruments* (TI) lançou o chamado microcontrolador, que além da função de controle de um microprocessador, agrega no mesmo circuito integrado memórias RAM e ROM, temporizadores, contadores e outros periféricos. Sua invenção, como dito, foi justificada pelo desenvolvimento ascendente de produtos cada vez mais dedicados. (ANDRIC e MATIC, 2000; JÁCOBO, 2001; PEREIRA, 2002; SCHWAMBACH, 2004)

O primeiro MCU comercial foi lançado pela TI em 1974, o TMS 1000 (Figura 15) de 4 bits, que portava memórias RAM e ROM e suporte a I/O em um único encapsulamento. Mais tarde, a Intel lançou os famosos microcontroladores da "família 8051", que teve como primeiro chip o 8048, em 1977. A partir destes marcos históricos, os microcontroladores cresceram em poder de processamento e na quantidade de periféricos inseridos. (ANDRIC e MATIC, 2000; JÁCOBO, 2001; PEREIRA, 2002; SCHWAMBACH, 2004)

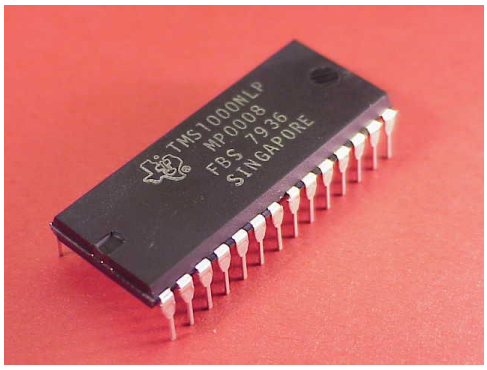

Figura 15. Primeiro microcontrolador, TMS 1000

3.2.2. Microcontroladores

Um microcontrolador é um pequeno dispositivo eletrônico, dotado de uma "inteligência" programável, utilizado no controle de processos lógicos. Este elemento possui um microprocessador integrado, além de uma enorme variedade de periféricos, como: memórias RAM e ROM, conversores A/D, interrupções externas, módulos de captura de sinais, PWM, etc. O detalhe mais importante é, talvez, que o microcontrolador tem todos estes utilitários eletrônicos encapsulados em um só *chip* eletrônico, o que facilita muito a utilização deste componente, pois estará ocupando pouco espaço em uma PCI (Placa de Circuito Impresso). (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; JÁCOBO, 2001; PEREIRA, 2002; SCHWAMBACH, 2004)

Os microcontroladores foram criados para realizar trabalhos dedicados. Apesar de serem facilmente programáveis e bastante confiáveis, em operação, eles não são tão flexíveis. Os MCUs, como são conhecidos, estão empregados

normalmente em equipamentos chamados de Sistemas Embutidos (*Embedded System*). Atualmente, o uso dos microcontroladores é bastante diversificado, podendo estar presentes em: relógios, eletrodomésticos em geral, carros, brinquedos, na instrumentação industrial, nos periféricos de computadores, celulares e etc. No mundo moderno, praticamente todos os equipamentos eletrônicos possuem uma pequena unidade realizando atividades de controle específicas ao sistema. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; JÁCOBO, 2001; PEREIRA, 2002)

No mercado, existem muitos fabricantes de microcontroladores, alguns exercendo atividades cada vez mais específicas em cada área. A capacidade de cada um, mesmo de uma mesma família de MCU's, é bem variada, dependendo apenas de como será empregado em cada projeto. O uso destes dispositivos se tornou bastante popular na última década em aplicações cada vez mais diversificadas como de simples projetos de estudantes e hobbystas a equipamentos de mercado de alto desempenho. Alguns exemplos de famílias muito utilizadas são: o PIC (Microchip), 8051(Atmel) e o MSP430 da *Texas Instruments*. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; JÁCOBO, 2001; PEREIRA, 2002)

3.2.3. Elementos Internos de um MCU

A organização interna de um microcontrolador varia muito de um fabricante para outro. Contudo, alguns elementos e estruturas são comuns entre os diversos modelos existentes no mercado. De forma geral, os MCU's possuem uma estrutura similar a mostrada na Figura 16. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; BRAGA, 2005; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002 e 2005; TEXAS, 2000 e 2007)

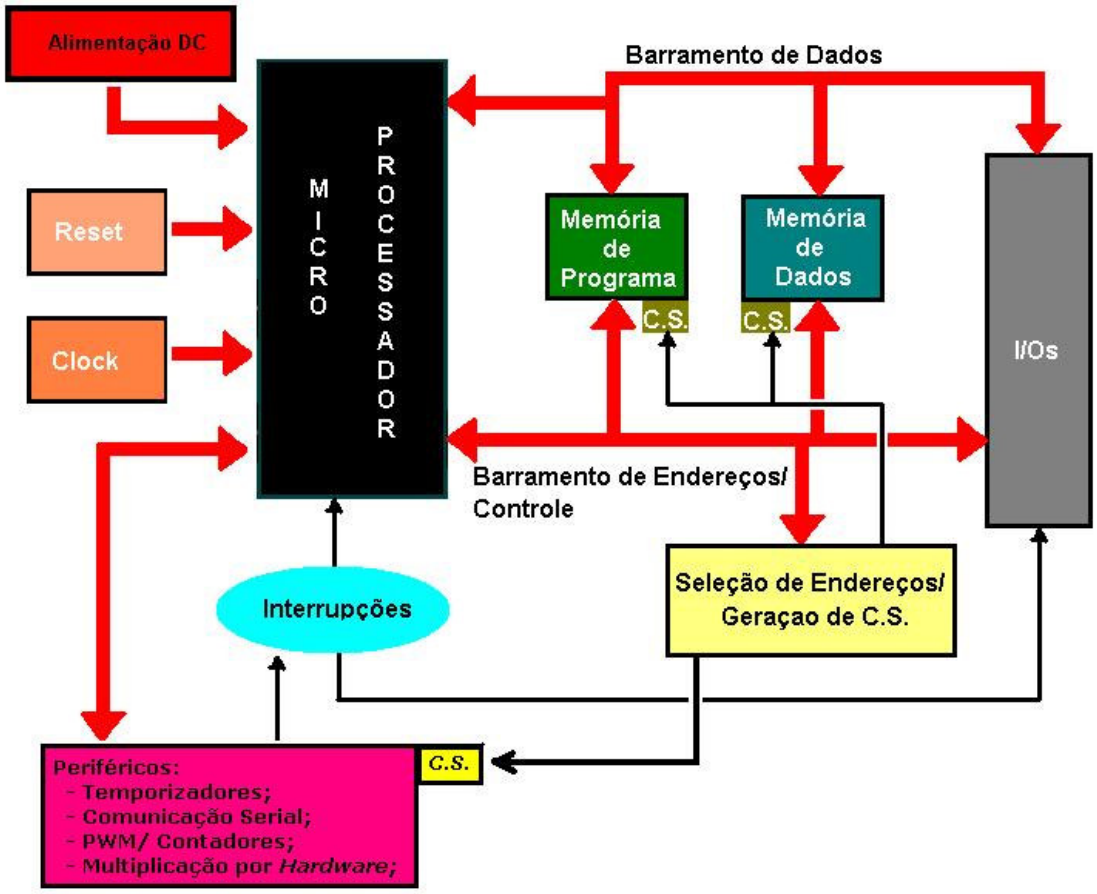

Figura 16. Organização interna genérica de um MCU

Pode ser visto na Figura 6 que um microcontrolador é formado por inúmeros componentes. Estes elementos atuam em conjunto e são todos gerenciados ou solicitados por uma unidade de controle central ou microprocessador. Desta forma, fica mais clara a versatilidade que possui um MCU ao unir em apenas um só sistema vários dispositivos. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; BRAGA, 2005; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002 e 2005; TEXAS, 2000 e 2007)

Todos estes itens que integram o microcontrolador são interligados através de barramentos ou vias de dados. Cada ligação pode ter uma função diferente e possui uma determinada classificação, podendo ser de dados, endereço ou controle. Dependendo da arquitetura da CPU, estas linhas que conduzem as informações podem ser projetadas em estruturas diferentes, sendo mais enxutas em alguns modelos ou até mesmo duplicadas. O próximo tópico deste trabalho aborda melhor estas características. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000;

BRAGA, 2005; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002 e 2005; TEXAS, 2000 e 2007)

Os circuitos de *reset* e de *clock* são essenciais para o funcionamento do MCU. O primeiro é responsável por inicializar o sistema quando for acionado e o segundo gera uma freqüência de tensão constante importantíssima para a CPU. Os pulsos de *clock* possuem a função de sincronizar as ações dos elementos do microcontrolador e principalmente sua unidade de controle. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; BRAGA, 2005; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002 e 2005; TEXAS, 2000 e 2007)

Toda a seqüência de processamento do microcontrolador vem de um *software* embarcado em uma eletrônica, também chamado de *firmware,* que é gravado na memória do *chip*. Esta é a chamada memória de programa, que deve ser do tipo não-volátil, para que as instruções não se percam ao se reiniciar o dispositivo. Contudo, praticamente todas as lógicas de programação exigem uma memória que auxilie na leitura e no processamento e registro de comandos. Os exemplos mais usuais são as criações de constantes e variáveis para o programa. Este segundo tipo de memória é denominada memória de dados e normalmente é do tipo volátil. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; BRAGA, 2005; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002 e 2005; TEXAS, 2000 e 2007)

A interface com o meio externo ao microcontrolador é “percebida” por suas entradas e saídas. Os terminais do *chip* enviam ou recebem sinais com o objetivo de ligar ou não um equipamento ou realizar algum tipo de comunicação. Os parâmetros de entrada podem ser reconhecidos no processamento como interrupções. Estas são paradas na execução do algoritmo já aguardadas, mas que podem ocorrer em um determinado intervalo de tempo não muito exato. As interrupções também podem ser geradas através de elementos internos do MCU, como: temporizadores, contadores, blocos de comunicação e de captura de sinais, etc. Ao ocorrer uma interrupção, independente da sua origem, um desvio no programa é feito para que se trate o fenômeno para depois voltar a instrução que seria executada antes que o processamento do código fosse interrompido. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; BRAGA, 2005; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002 e 2005; TEXAS, 2000 e 2007)

3.2.4. Arquiteturas de CPUs

A arquitetura computacional existente hoje se baseia no modo que são dispostas as memórias de programa e de dados. O acesso a estes periféricos pode ser realizado de modo individual, cada um com uma via própria, ou de forma única, utilizando-se apenas um barramento. A maneira que esta organização e feita define o modelo organizacional de uma CPU que pode ser classificada como arquiteturas de Harvard ou de Von Neumann. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; BRAGA, 2005; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002 e 2005; TEXAS, 2000 e 2007)

A arquitetura Harvard esteve presente nos projetos dos primeiro computadores com o Marck I e o ENIAC. Sua estrutura básica é definida pela separação das memórias do sistema: de programa e de dados. Ambas são acessadas independentemente por barramentos próprios. Esta máquina, apesar de ter um aspecto construtivo mais complexo possui alto desempenho e permite que a CPU busque a próxima instrução ao estar executando a atual (o chamado *pipeline*). Na Figura 17, podem ser vistos a separação física e de acesso (flechas vermelhas) aos dois tipos de memórias. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; BRAGA, 2005; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002 e 2005; TEXAS, 2000 e 2007)

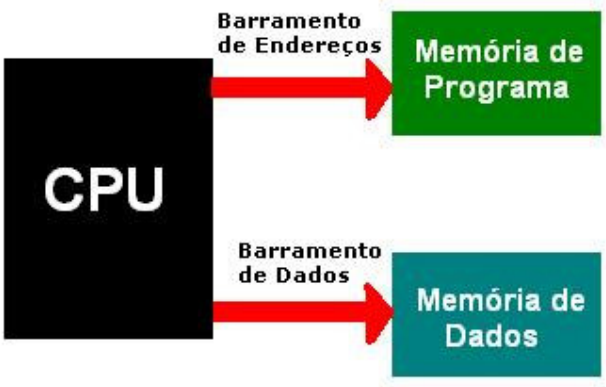

Figura 17. Arquitetura de Harvard

A arquitetura de Von Neumann (Figura 18) é mais atual e se caracteriza por permitir que uma máquina possa armazenar suas informações na mesma memória que foi gravada as instruções de programa. Apesar de sua construção ser mais simplificada, o acesso as memórias é apenas possível por um único barramento, o que deixa o sistema ligeiramente mais lento ao se comparar com a arquitetura anterior. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; BRAGA, 2005; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002 e 2005; TEXAS, 2000 e 2007)

Esta nova proposta organizacional foi criada por um grupo de engenheiros liderados pelo matemático húngaro John Von Neumann. Muitos dos projetos de computadores atuais ainda são fortemente influenciados pelas premissas deste pesquisador. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; BRAGA, 2005; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002 e 2005; TEXAS, 2000 e 2007)

Figura 18. Arquitetura de Von Neumann

Um tipo de classificação específica dada aos microprocessadores se refere a quantidade de instruções que o mesmo dispõem. Alguns microcontroladores, por exemplo, possuem um *set* de instruções reduzido (designa-se RISC, da abreviação em inglês). Neste caso os projetos dos *chips* são mais baratos, contudo a uma dificuldade maior ao desenvolver um *firmware*. Outros processadores possuem características contrárias e portam de mais instruções, visando facilitar e objetivar os códigos de programação. Em modelos de mesma classe, essa diferença pode variar de 35 a mais de 100 comandos diferentes. Esta categoria é chamada de Conjunto Complexo de Instruções Computacionais ou CISC do inglês *Complex Instruction Set Computer*. (ANDRADE e OLIVEIRA, 2006; ANDRIC e MATIC, 2000; BRAGA, 2005; JÁCOBO, 2001; NERYS, 2006; PEREIRA, 2002 e 2005; TEXAS, 2000 e 2007)

3.2.5. Microcontroladores da Família MSP 430

A família de processadores MSP430 foi desenvolvida pela *Texas Instruments* para aplicações de baixo consumo de energia. Estes microcontroladores possuem design simples e poderoso, processando instruções em altíssima velocidade. Projetado sobre a clássica arquitetura de Von Neumann, o MCU permite que a CPU (Unidade Central de Processamento) possua um modo de endereçamento a suas principais unidades de memória (dados e programa). (PEREIRA, 2005; TEXAS, 2000 e 2007)

É muito comum relacionar a arquitetura de Harvard com a RISC e a de Von Neuman com a CISC. Justificado pela facilidade de se estruturar os *chips* com estas duas características. Contudo, na família MSP430, o projeto de seus microcontroladores inclui em sua estrutura interna uma arquitetura baseada na de Von Neumann aliada a um conjunto de instruções reduzido (RISC). Uma grande diferença para os outros microcontroladores é o fato da família MSP430 ser do tipo RISC. (PEREIRA, 2005; TEXAS, 2000 e 2007)

Este MCU possui 27 instruções em *Assembly* e mais 24 simuladas. Esse acríssimo é possível pela existência de registradores geradores de constantes numéricas. Para isso existe a flexibilidade de que os registradores R2 e R3 assumam os valores 4 ou 8 (para o R2) e 0, 1, 2 ou 0xFFFF (para o R3). Alguns especialistas de arquiteturas computacionais não concordam que a família MSP430 seja RISC por existir variações entre os tempos de processamento de determinadas instruções, uma característica inerente a CPU's do tipo CISC. (PEREIRA, 2005; TEXAS, 2000 e 2007)

Um dos maiores aspectos que destacam os microcontroladores MSP430 é sem dúvidas seu baixo consumo. Esta família possui 5 modos de operações que permite modificar com rapidez e eficiência a velocidade de processamento e, consequentemente, o gasto de energia. No *Active Mode* (Modo Ativo), onde todas as funções do MCU estão em funcionamento pleno, se consome em média apenas 250 μA por MIP (milhões de instruções sendo executadas por segundo) e no *Power Down Mode* (Modo de baixa energia), apenas a alimentação da memória RAM é conservada, em torno de 100 nA. (PEREIRA, 2005; TEXAS, 2000 e 2007)

Modos intermediários de operação também são possíveis. No *Stand by Mode* (Modo de espera), por exemplo, os periféricos continuam funcionando, porém com uma redução nos pulso de *clock*. Neste estágio, o consumo é de 800 nA. (PEREIRA, 2005; TEXAS, 2000 e 2007)

A alternância entre os modos de operação é muito comum em aplicações com estes microcontroladores. A facilidade, rapidez e eficiência que isto ocorre na linha MSP430 é uma outra excelente característica. Ao alterar alguns *bits* em registradores de configuração consegue-se migrar, por exemplo, do *Power Down Mode* ao *Active Mode*, isto em um tempo próximo de 1 μs. (PEREIRA, 2005; TEXAS, 2000 e 2007)

Estes microcontroladores podem operar em baixas tensões que podem variar de 1,8 V a 3,6 V e as tensões mínimas para programação da FLASH variam entre os modelos, podendo ser de 2,2 V ou 2,7 V. Isso permite que os *hardwares* desenvolvidos sejam alimentados com simples baterias por um bastante tempo. (PEREIRA, 2005; TEXAS, 2000 e 2007)

A linha MSP430 é subdividida em cinco famílias de acordo com parâmetros como o tipo de memória de programação existente, são elas: MSP430x1xx, x2xx, x3xx, x4xx e x5xx. Os encapsulamentos variam de simples *chips* QFN de 24 pinos e dimensões 4 x 4 mm aos poderosos LQFP, com 100 pinos de 16,20 x 16,20 mm. Não existem versões de encapsulamento DIP (do inglês, *dual in line*) para as famílias 1xx, 2xx e 4xx. Neste caso, são utilizados componentes com tecnologia SMT (*Surface Mounting Technology* – Tecnologia de montagem em superfície). (PEREIRA, 2005; TEXAS, 2000 e 2007)

3.2.6. Conjunto de Instruções (*Assembly*)

Como foi apresentado, a família MSP430 tem um CPU RISC com 27 instruções físicas, e por características internas do processador, é possível simular mais 24 o*p-codes*. Para estabelecer um melhor entendimento sobre a funcionalidade de cada um destes códigos, será realizada uma divisão entre os mesmos em: (PEREIRA, 2005; TEXAS, 2000 e 2007)

- Instruções de movimentação e manipulação de dados: Utilizadas em operações que se pretende transportar, copiar ou apagar um dado da memória ou de um registrador.
- Instruções aritméticas e lógicas: Realizam operações matemáticas ou lógicas entre registradores.
- Instruções de teste e desvio: Testam um registrador ou um *bit* e promovem ou não desvios calculados do programa. O destino destes saltos podem ser sub-rotinas (quando existe o retorno) ou outros trechos do programa.
- Instruções de controle da CPU: Estabelecem o controle dos estados da CPU, principalmente dos *flags* do processador.

No **Anexo 1**, a Tabela 01 mostra as 51 instruções em *Assembly* e o padrão de cada mnemônico com sua respectiva descrição e as possíveis alterações nos sinalizadores da CPU (tabelas retiradas de PEREIRA (2005), pg. 403-404). (PEREIRA, 2005; TEXAS, 2000 e 2007)

3.2.7. Periféricos e módulos internos (MSP430)

A família MSP430 possui em sua arquitetura um enorme número de periféricos que auxiliam bastante o desenvolvimento de programas com estes dispositivos. Contudo, nem todos os microcontroladores desta linha portam de todos módulos disponíveis pelo fabricante. Por isso, cada projeto deve ter seu MCU devidamente dimensionado partindo do princípio de quais periféricos serão necessários. Um dos módulos internos que mais se destaca são os conversores A/D multiplexados de 10 e 12 canais. (PEREIRA, 2005; TEXAS, 2000 e 2007)

Os ADC10 podem ser acessados em 5 canais (modelos 11x2) ou 8 canais (modelos 12x2) externos com mais 4 internos. Pelo *firmware,* o programador seleciona qual entrada será utilizada pelo conversor. Este A/D pode usar como referencia fontes externas ou internas (a alimentação do MCU) e a taxa de amostragem é controlada, podendo variar a depender de configurações feitas no programa. (PEREIRA, 2005; TEXAS, 2000 e 2007)

O conversor analógico digital de 12 *bits* da família MSP430 é muito parecido com sua versão de menor resolução. Porém, alguns modelos possuem até 12 canais de leitura externos e mais 4 internos. E para facilitar a programação e leitura destes valores, os 16 canais possuem registradores próprios e geram, individualmente, uma interrupção específica. Duas outras interrupções são geradas para sinalizar erros na conversão. (PEREIRA, 2005; TEXAS, 2000 e 2007)

Este periférico foi destacado por ser o mais bem explorado nos programas embarcados neste projeto. Os sinais oriundos dos sensores de temperatura serão condicionados por um circuito específico até terminais do microcontrolador. Essas tensões são muito pequenas com poucas variações. E para promover maior confiabilidade e exatidão nas leituras e preciso conhecer melhor o funcionamento dos conversores A/D do microcontrolador a ser utilizado. (PEREIRA, 2005; TEXAS, 2000 e 2007)

Outros periféricos disponíveis na família MSP430 são:

- Temporizadores e contadores: Gera bases de tempo necessárias para o funcionamento de quase todos os outros módulos e para a realização de tarefas macro que depende do tempo. Os contadores são utilizados para criar intervalos periódicos de tempo. Utiliza *clock*s internos ou externos.
- Comunicação USART: Possibilita uma comunicação serial síncrona/ assíncrona e universal entre dispositivos como: computadores, microcontroladores e outras interfaces de comunicação.
- Modos de captura, comparação e geração de sinais PWM: A partir de bases de tempos criadas pelos temporizadores, realiza tais operações.
- Conversor Digital Analógico: Emite um sinal analógico proporcional a um valor base de 12 *bits*.
- Multiplicação por *Hardware:* Realiza operações matemáticas de multiplicação de 8 ou 16 bits, sinalizadas ou não, por *hardware*. Este bloco independente da CPU.
- Controlador de LCD: Nas famílias 4xx, é possível controlar *displays* de cristal liquido através de multiplexadores digitais aliados a temporizadores (*Timer* 1).

- Controlador de DMA: Acessa diretamente uma memória de forma automática sem a necessidade de se implementar um protocolo específico.
- Controlador de Memória FLASH: Apaga ou escreve diretamente na memória FLASH do *chip*. Utiliza um circuito próprio de temporizarão para efetuar estas tarefas.

(PEREIRA, 2005; TEXAS, 2000 e 2007)

3.2.8. Programando em C na família MSP 430

Uma comum discussão ao se tratar de programação com microcontroladores é de qual a melhor linguagem para se realizar um projeto. A linguagem de baixo nível exige muito tempo e conhecimento do profissional. Ao se fazer esta escolha, precisa-se conhecer o funcionamento com detalhes e o papel de que cada instrução no *hardware*. De forma sucinta, a complexidade de se programar numa linguagem tão próxima da de máquina exige um enorme tempo de desenvolvimento. (GACLI, 1998; PEREIRA, 2005; TEXAS, 2000 e 2007)

Para a maioria dos fabricantes de microcontroladores, existem compiladores que convertem para o *Assembly* linguagens de alto nível e mais confortáveis para se programar. A C é uma delas, criada em 1972, no centro de pesquisas *Bell Laboratories*, por Dennis Ritchie (Figura 19) e é definida por uma linguagem de propósito geral, sendo adequada a programação estruturada. (GACLI, 1998; PEREIRA, 2005; TEXAS, 2000 e 2007)

Fonte: http://cm.bell-labs.com/who/dmr/

Figura 19. Dennis Ritchie

Existem dois grandes problemas em se programar em alto nível em microcontroladores. O primeiro deles é o tamanho do código que pode ser bastante alongado ao se comparar a um programa em *Assembly*. Ocupando, assim, muito mais espaço na memória do *chip*. O outro inconveniente clássico é em projetos em que o tempo é uma variável importante, e por isso, crítica. Nestes casos uma programação em C, por exemplo, pode mascarar pra o programador o tempo de execução de uma determinada função ou rotina. Resultando assim, num comprometimento da funcionalidade de um dispositivo desenvolvido em para operar em ambientes com alto nível de dinamismo. (GACLI, 1998; PEREIRA, 2005; TEXAS, 2000 e 2007)

Contudo, para a maioria das aplicações não existe nenhum problema em utilizar a linguagem C. E pela sua popularidade, os próprios fabricantes de microcontroladores permitem e viabilizam em seus *softwares* de desenvolvimento de projetos com a mesma. Sem contar que escrever um programa em C, com a mesma eficiência de um *Assembly,* e totalmente exeqüível. E isso pode ser feito em muito menos tempo, o que favorece o aprendizado de uma nova família dede MCU's. (GACLI, 1998; PEREIRA, 2005; TEXAS, 2000 e 2007)

3.2.9. Técnicas de Programação com MCU

Um microprocessador executa suas instruções de maneira seqüencial. Este processo pode ocorrer de modo linear ou circular. Na Figura 20 é possível ver o primeiro caso, onde as instruções do programa são processadas, uma por uma, até que se atinja o final do código. E, consequentemente, a atividade da CPU é encerrada. (ALMEIDA, 2007)

Figura 20. Forma de execução linear

Na segunda forma de execução (Figura 21), a leitura do programa é feita num ciclo, estabelecendo continuidade a sua aplicação. Nesta espécie de laço de repetição, o processador trabalha por um ciclo infinito e apenas se encerra caso uma determinada instrução desta seqüência realize um salto para uma linha de programa que promova tal condição de finalização. Este método é bastante eficiente, pois permite que várias funções, periféricos ou módulos sejam acessados ou monitorados periodicamente. (ALMEIDA, 2007)

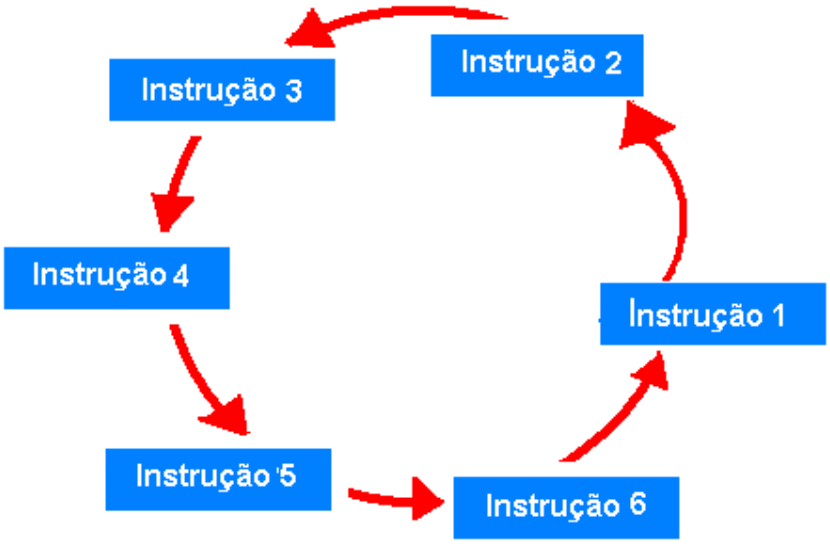

Figura 21. Forma de execução circular

Na maioria dos projetos, há necessidade de se estabelecer uma interface com o mundo real, que em relação ao mundo da eletrônica é regido numa escala de tempo muito superior. Nestas aplicações, é comum criar referências de tempo ou temporizações para se torne possível esta ligação. Um exemplo simples que pode retratar melhor este problema é um simples piscar de um LED. Para que uma pessoa consiga perceber o efeito “pisca-pisca”, é preciso que este componente seja energizado e desligado em período por um tempo superior ao de relaxamento do olho humano. (ALMEIDA, 2007)

Em uma atividade aparentemente simples, como a apresentada no parágrafo anterior, um problema comum é gerado pela maioria dos programadores de microcontroladores. A geração de atrasos ou funções de *delays* (atrasos), como são mais conhecidas, define-se na implementação de *loops* na seqüência de execução de programa, onde uma instrução só será executada caso uma determinada condição for satisfeita. Uma operação de teste é feita continuamente em um *bit* ou *word* que apenas será validado após um certo período de tempo. (ALMEIDA, 2007)

Na Figura 22, fica claro que a Instrução 3 só será executada caso a condição de teste seja satisfeita. A variável X pode ser um registrador que tem seu valor incrementado ao se processar a Instrução 2 ou então é um *flag* indicador de uma interrupção temporizada, por exemplo. (ALMEIDA, 2007)

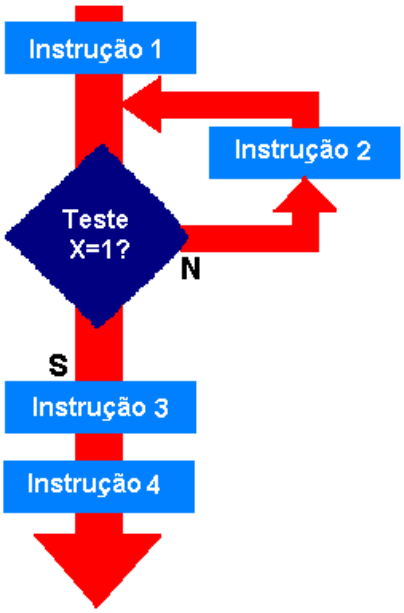

Figura 22. Gerando bases de tempo por *delay*

Esta técnica é bastante utilizada por principiantes ou amadores, que utilizam de rotinas baseadas na Figura 22 para gerar temporizações em seus programas. Essa forma de programação promove um problema muito grave: a paralisação momentânea da CPU. Em um *delay* (atrazado) gerado por um *loop*, a execução do programa permanece estacionada até que o tal ciclo seja finalizado. (ALMEIDA, 2007)

Quando se pretende atingir uma temporização com a técnica apresentada, é preciso travar o processamento. Isto pode provocar perda de informações, como no caso em que uma operação paralela esteja ocorrendo e necessite de um tratamento digital, por exemplo. É válido lembrar que na maioria das aplicações de caráter profissional, os projetos possuem um dinamismo cada vez maior, reforçando ainda mais as deficiências deste método. (ALMEIDA, 2007)

Um modo que permita ao programador utilizar com maior eficiência mais de um periférico em um projeto microcontrolado de forma praticamente simultânea, é com o auxilio de interrupções por *hardware*. (ALMEIDA, 2007)

Os *TIMERS* geram interrupções comuns, encontradas em praticamente todos os modelos de microcontroladores. E já retornando para o problema inicial de criar um temporizador, pode-se desenvolver uma simples solução baseado neste periférico interno. Sabe-se que um *flag* gerador de interrupção é “setado” (levado ao nível lógico alto) sempre que um TIMER encerra sua contagem. E dependendo de

como for configurado este bloco, o sinalizador tem seu nível lógico modificado em um intervalo de tempo definido. (ALMEIDA, 2007)

A cada interrupção do TIMER configurado, o processamento salta do *loop* principal para uma linha específica do programa. Neste novo “local”, pode ser criada uma base de tempo. Sua representação é feita, basicamente, por um *bit* que apenas será setado no momento em que uma interrupção de temporizarão ocorrer. Por exemplo, se o TIMER estourar a cada milisegundo, um *bit* sinalizará que esta base de tempo (BT_1ms) aconteceu. No programa, agora, todas as funções ou rotinas que dependem desta base de tempo podem ser executadas. Após este tratamento, é importante limpar (levar ao nível lógico baixo) o *flag* BT_1ms. (ALMEIDA, 2007)

Uma base de tempo nem sempre é suficiente para todas as necessidades de sincronização em um projeto. Além disto, pode ser inviável utilizar todos os módulos de temporização, pois muitos dos periféricos necessitam também dos mesmos. Esta nova filosofia apresentada permite ainda uma solução prática para este problema: criar bases de tempo (*bits* sinalizadores) múltiplos da criada na interrupção do TIMER. Para exemplificar, pode-se criar uma sub-rotina com variáveis que contam quantas vezes a base de tempo foi setada. Quando estes contadores atingem um valor especifico, geram-se novas bases de tempo: BT_2ms, BT_10ms, BT_100ms e etc. (ALMEIDA, 2007)

A nova estrutura de um programa que utiliza esta técnica de programação é mostrada na Figura 23. É importante evidenciar que a CPU não fica presa no tratamento da interrupção, onde apenas a base de tempo é setada, proporcionando um retorno rápido ao *loop* principal. O retângulo azul “Instrução 2” simboliza um bloco de instruções que gera uma base de tempo secundária (BT_20ms). (ALMEIDA, 2007)

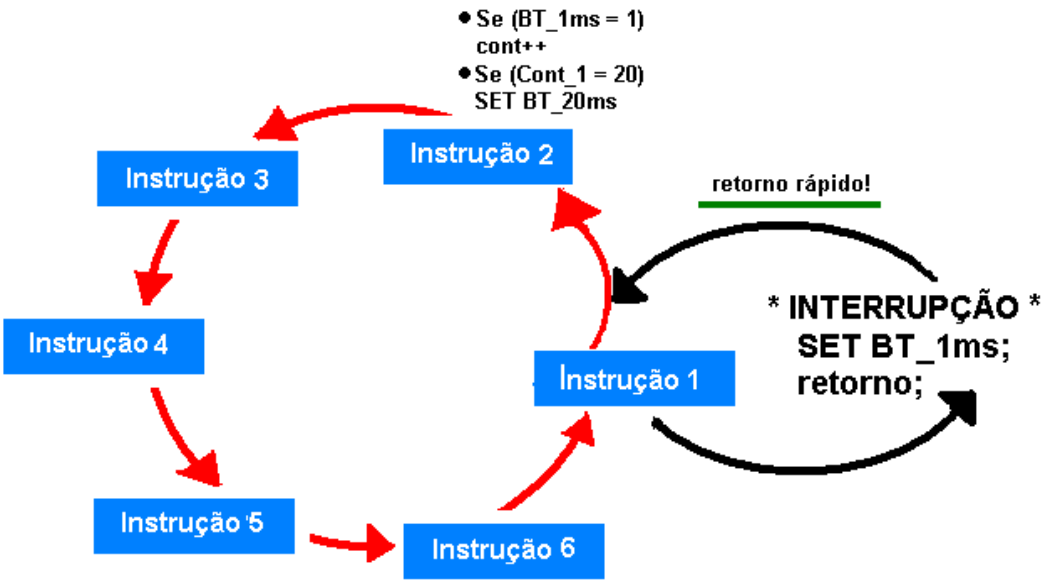

Figura 23. Utilizando uma base de tempo

Inserir novos estados ou funções com dependência do tempo a um programa em que se utiliza esta técnica se torna mais fácil. A Figura 24 é similar à 23, contudo, foram acrescentados novos blocos com instruções. Neste exemplo, três LEDs piscam em freqüências diferentes, o primeiro a cada 1 ms, o segundo em 20 ms e o último em 500 ms. (ALMEIDA, 2007)

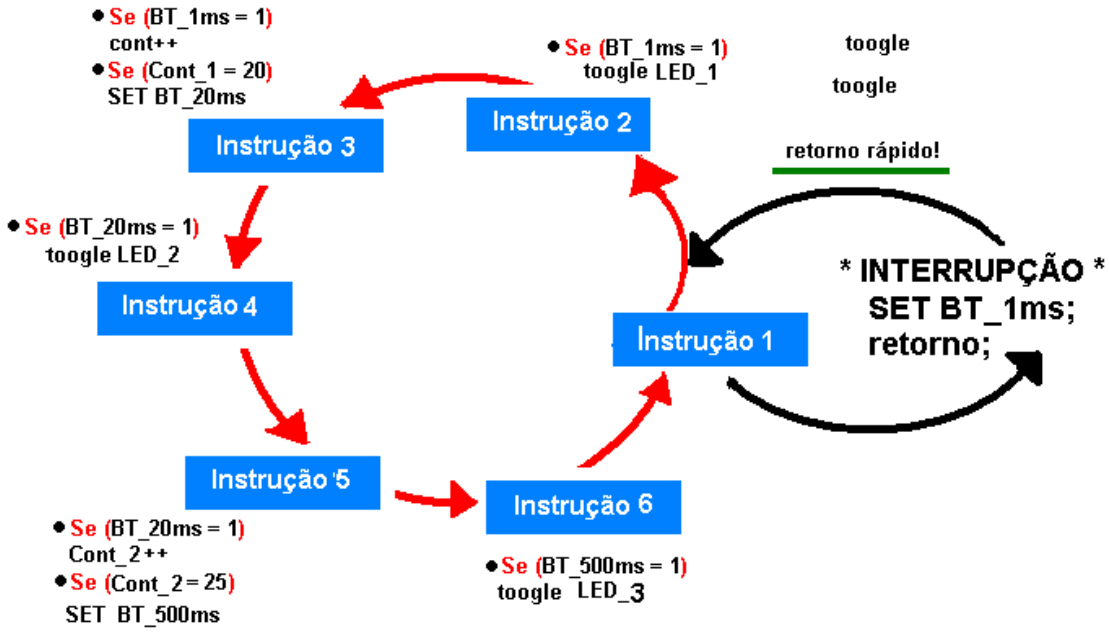

Figura 24. Acrescentando estados ao programa

É fato que cada programador possui uma técnica própria para desenvolver seus programas. Dentre uma enorme diversidade de métodos, que são bastante é preciso ter uma precaução maior quando se trata de *softwares* embarcados. O tempo, na maioria dos projetos, é uma variável crítica que pode afetar a funcionalidade do equipamento. Utilizar técnicas com *loops* extensos que bloqueiam temporariamente a CPU é uma alternativa que deve ser desconsideradas nas condições apresentadas. (ALMEIDA, 2007)

3.3. Comunicação Serial de Dados

Em um circuito eletrônico, dados são enviados entre os componentes do mesmo a todo instante. Sinais de tensão, por exemplo, são emitidos através de trilhas por dezenas de centímetros até atingir seu destino. Contudo, em algumas situações, é preciso enviar dados a dispositivos que estão em equipamentos diferentes dos de origem. Nestes casos, pode-se aumentar consideravelmente a distância e a confiabilidade desta transmissão. (ALMEIDA, 2007; CANZIAN, 2002)

A comunicação de dados define como: equipamentos distintos, um destes externo ao outro, como dispositivos que portam de alimentações próprias e independentes. (ALMEIDA, 2007; CANZIAN, 2002)

Problemas muito comuns em comunicações a grandes distâncias são causadas pelo meio externo que interferem no sinal enviado, e distorções na carga, provocado pelo tamanho dos condutores por onde trafegam estes dados. Dependendo do ambiente em que um projeto de comunicação se instale, as distorções e os ruídos podem ser tão severos ao ponto de destruir a informação passada. (ALMEIDA, 2007; CANZIAN, 2002)

Dois conceitos importantes para o entendimento de uma comunicação de dados são: a potência do sinal e a taxa de transmissão. O primeiro deles está relacionado ao consumo necessário para se estabelecer o envio da informação. O segundo conceito se refere à velocidade da comunicação que esta ligada diretamente ao primeiro termo. Nestes sistemas, é importante em um projeto estabelecer a comunicação com a menor potência possível junto com o menor nível de ruído. (ALMEIDA, 2007; CANZIAN, 2002)

3.3.1. Comunicação de dados

Em um projeto, onde a comunicação entre *hardwares* ocorre por de longas distâncias, normalmente, as mensagens digitais são maiores que simples *bits*. Criar um meio físico que permita uma transferência de toda a informação simultânea não é fácil. A comunicação serial consiste em "quebrar" a mensagem para que cada parte desta seja enviada individualmente por um canal. No destino, todos estes *bits* são reunidos para que a unidade de controle local possa entender a informação e realizar os devidos processamentos. (ALMEIDA, 2007; CANZIAN, 2002)

Uma comunicação em paralelo apresenta taxas altíssimas de transmissão, contudo, o custo para é muito maior. E este tipo de envio, não é muito aconselhável para distâncias muito grandes, para que não haja perda de informação. Este problema não é comum na comunicação serial entre pontos distantes. (ALMEIDA, 2007; CANZIAN, 2002)

3.3.2. Meio Físico RS-232

A interface RS-232 foi criada na década de 60 do século XX por um comitê conhecido como *Electronic Industries Association* - EIA. Esta padronização especifica tensões, temporizações, conexões mecânicas, as funções de cada sinal e um protocolo para uma troca de informação entre dois equipamentos. Anteriormente, a comunicação digital era realizada entre um computador central ou *Mainframe e* os terminais remotos de dois ou mais computadores. (ALMEIDA, 2007; CANZIAN, 2002)

Os usuários desta interface adotam diversas soluções mais simplificadas que tornam possível a simplificação da padronização proposta. As duas maiores dificuldades na utilização do RS-232 (sendo que RS vem do inglês *Recommended Standard* ou Padrão Recomendado) são:

- A ausência ou conexão errada de sinais de controle o que resultam em estouro do buffer "*overflow*" ou travamento da comunicação por excesso de fluxo.

- Função incorreta de comunicação para o cabo em uso que resulta em inversão das linhas de Transmissão (TX) e Recepção (RX), bem como a troca de uma ou mais linhas de controle ou "*handshaking*".

(ALMEIDA, 2007; CANZIAN, 2002)

4. MODELO DE SOLUÇÃO PROPOSTA

4.1. Especificidades do projeto

Na Figura 25, pode-se visualizar uma idéia esquematizada do planejado neste trabalho. E em uma visão mais voltada para a lógica de controle do sistema, é possível tornar mais claro a compreensão do que será realizado em matéria de programação.

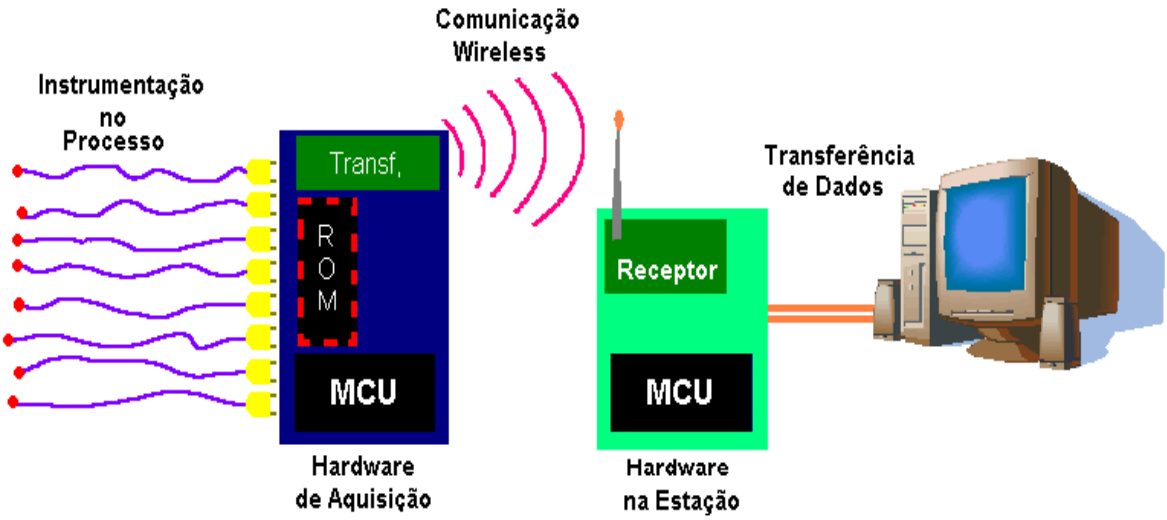

Figura 25. Esquema geral do projeto (*Hardware* e *Software*)

Sensores no processo irão fornecer de forma contínua sinais que corresponderão a temperatura do meio que se deseja medir. O sinal de cada ponto deverá ser tratado digitalmente com a finalidade de eliminar ruídos inerentes as condições do sistema.

Os dados serão lidos pela unidade central de controle em um período pré-determinado e estas informações serão armazenadas em uma memória. Um estudo sobre o dimensionamento dos dispositivos internos está sendo feito para definir se há, ou não, a possibilidade de se utilizar a memória interna do próprio microcontrolador. Em caso de um grande volume de informações, que será muito provável, poderá ser inserida no *hardware* uma EEPROM externa.

O tempo médio que o *Carrier* leva para percorrer toda a Secção B é de 2h e 30min. As tomadas de temperatura e a armazenagem em um banco de dados devem abranger todo este período.

A transferência destas informações será feita através de uma comunicação *wireless*. É necessário que haja um cuidado especial sobre o protocolo, sincronia e outras configurações feitas entre os microcontroladores dos dois *hardwares*. Esta etapa é importantíssima, visto que não pode existir a possibilidade de que os elementos transferidos se percam ou sejam alterados neste procedimento.

Após o *Carrier* ser liberado da estação, o próprio *firmware* se encarregará de iniciar o procedimento para que possa se transferir os dados para um meio que seja acessível para que aquelas informações possam ser analisadas por um engenheiro. Por questão de facilidade, este será direcionado para um simples computador pessoal (existem pelo menos dois na Seção B).

Este protótipo deverá ser totalmente autônomo, e a medição ou coleta dos dados da variável temperatura será feita de modo automatizado. O que irá impedir que haja alguma intervenção humana no processo e evitando, assim, inúmeros distúrbios na variável a ser mensurada, como acontece atualmente.

Como o maior interesse do cliente é ter um registro geral da temperatura de superfície dos cilindros durante a sua passagem pelo interior da segunda etapa do processo de produção, não basta apenas medir e capturar os dados. O projeto de uma estação se justifica em viabilizar o acesso dos engenheiros da XEROX a estas informações. Por inúmeras questões, é descartada a uma comunicação *on-line*, seja por fios (extensão, isolação física, dinâmica do processo e etc.) ou até mesmo *wireless* (zona classificada, ambiente que reproduz uma “gaiola de Faraday”, etc).

Por esta razão, em um trilho já existente fora da linha de produção será implantado um local que identificará o dispositivo desenvolvido, que estará embarcado em um *Carrier* ou transportador do processo. Este transportador conterá toda a instrumentação necessária para a mensuração e armazenamento da variável. Com este reconhecimento feito, a comunicação destas duas entidades pode ser estabelecida.

É importante ressaltar que um dos principais objetivos do trabalho é realizar todos os procedimentos ditos sem provocar nenhum, ou o mínimo efeito sobre a produção, e na qualidade dos OPC’s produzidos. Não é pretendido que nenhum dos

fotorreceptores seja perdido ao passar pelo protótipo que será desenvolvido neste projeto.

4.2. Propostas de Validação

4.2.1. Simulação por *Software*

O microcontrolador utilizado neste projeto é um da família MSP430. A IAR desenvolveu um *software* que entre suas ferramentas permite se realizar simulações e depurações de erros no algoritmo desenvolvido antes mesmo de gravá-lo no *chip*. O *Embeldded Workbench* é um ambiente de integrado de desenvolvimento que facilita muito o uso dos MCU desta família, nele pode-se escrever o código, depurar, compilar, montar, gravar e testa-lo em *hardware*.

Este será o primeiro método de validação do trabalho, contudo é claro que passando por este estágio, não significa que não há erros no *firmware*. Pois muitos dos "famosos *bugs*" são difíceis de serem identificados, uma vez que nestas simulações, algumas variáveis não são consideradas.

4.2.2. Depuração por *Hardware*

Os microcontroladores da família MSP430 possuem um excelente conjunto de interface de programação e depuração de *software*. Com simples circuitos que podem ser inseridos do próprio projeto, o programador pode testar no seu próprio *hardware* o funcionamento do algoritmo.

A interface JTAG (do inglês, *Joint Test Action Group*) é um padrão industrial para testes, programação e depuração de *softwares*. A família MSP430 adota este meio e disponibiliza pinos específicos para o uso desta ferramenta.

Por isso, um seguinte meio de testar tanto o a funcionalidade do hardware quanto a objetividade dos *firmwares* será com o auxílio desta interface. O conector padronizado pela *Texas Instruments* para a comunicação da ferramenta de depuração é o chamado *Flash Emulation Tool* ou FET (Figura 26).

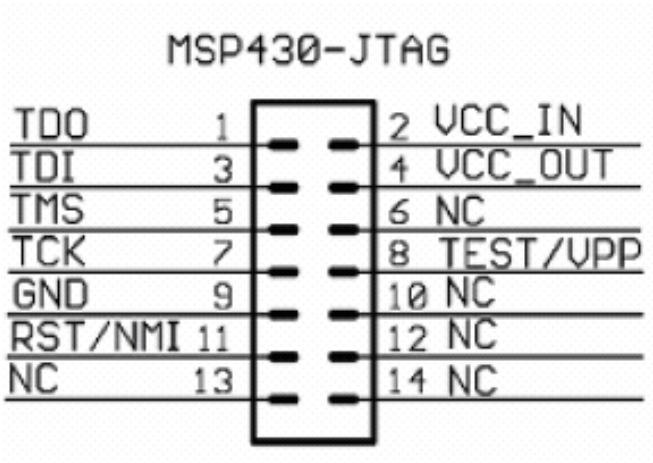

Figura 26. Padrão FET

4.2.3. Experimentos em campo

O gerente de produção da XEROX disse que seria favorável a criação de um ambiente agressivo muito similar ao da linha da produção da própria fábrica. Com o protótipo desenvolvido, é bastante interessante testá-lo nesse local a fim de se validar totalmente o trabalho.

Esta simulação de um ambiente simulado pode ser também construída pela equipe de trabalho. Contudo, isto acarretará tempo e disponibilidade que outras atividades deste projeto também irão exigir.

Vale lembrar que em um projeto mecatrônico, seu funcionamento só é garantido quando todas as partes trabalham integradas e com eficiência. Então, esta proposta é a que, definitivamente, irá validar o trabalho.

5. DESENVOLVIMENTO DO PROJETO

5.1. Introdução

A metodologia TheoPrax prega também, que os 3 alunos desenvolvedores apresentem formalmente uma proposta de solução para a empresa contratante. E só a partir de uma aprovação, esta empresa libere o orçamento para que o projeto seja montado e testado.

A proposta da equipe foi aceita pela Gerência de Processo da XEROX. Contudo, o setor financeiro da unidade só pode liberar o orçamento após o mês de Julho de 2008. Nesta situação não foi possível iniciar a implementação física do projeto neste período de pesquisa.

Para a ramificação que este documento aborda as etapas de testes de campo do projeto não puderam ser realizadas. Então, este capítulo estará abordando: o projeto de telas do supervisório, os fluxogramas dos códigos desenvolvidos e o trabalho paralelo a este tema que foi levado ao evento INOVA SENAI 2008.

5.2. Projeto de Telas

A idéia de se criar um S*oftware* supervisório neste projeto surgiu depois de uma visita a XEROX, quando foi apresentado um programa da OMEGA que armazenava e plotava os dados lidos pelo “MOLE”.

A função do programa desenvolvido neste projeto foi estruturada a partir das necessidades que o engenheiro de processos da Seção B possui e das premissas que foram levantadas pela equipe deste projeto. Estes “adicionais” foram pensados com o objetivo de facilitar a visualização do banco de dados colhidos no processo. Deve-se considerar que no *software* atual, apenas comporta dados de 6 sensores enquanto neste projeto se pretende utilizar até 33.

Outras funções importantes neste programa são: se comunicar com os controladores nas duas placas eletrônicas projetadas. Receber as informações dos sensores no *Carrier* e enviar informações de controle são inovações neste projeto.

Para a implementação deste programa foi utilizado o *software* Borland C++ Builder 6.0. Este programa possui ferramentas para a criação de programas que facilitam muito este processo de desenvolvimento. Na Figura 27, são mostradas algumas das telas criadas.

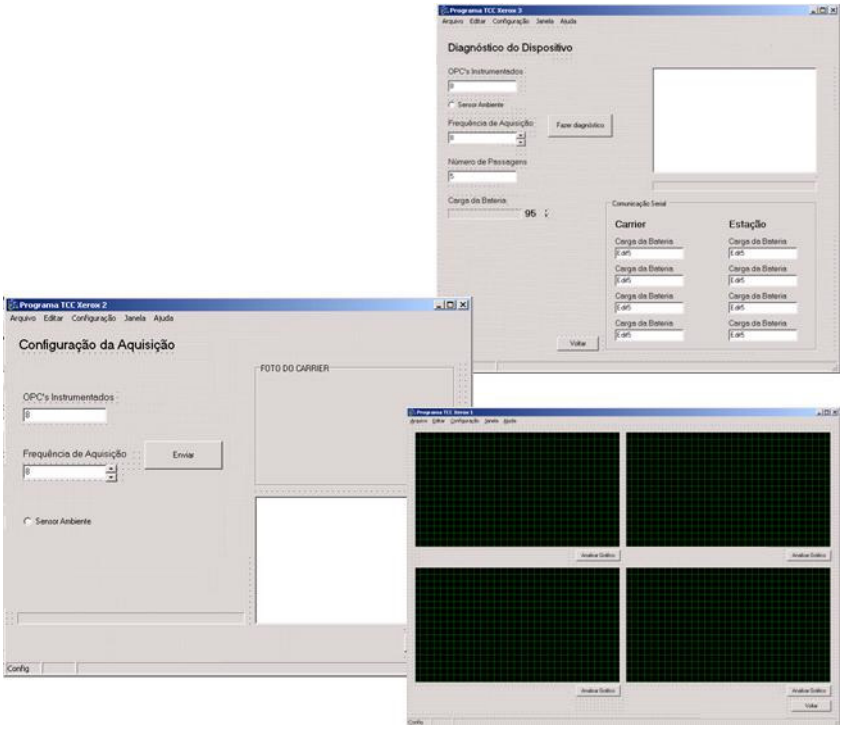

Figura 27. Telas projetadas no *Builder*

O programa precisa ser simples e intuitivo. Uma tela de configuração de envio de dados para os dispositivos vai configurar os as taxas de transmissão e freqüências dos microcontroladores. Outra tela será destinada para se levantar as atuais configurações dos MCU's.

Na última e mais importante tela, o usuário escolhe quais sensores serão plotados nas janelas destinadas à gráficos. E neste mesmo local pode-se analisar o valor de temperatura em cada ponto do processo. A opção de salvar uma tabela de dados em um arquivo *.txt também é possível, caso necessário.

5.3. Fluxogramas

Para que os códigos fiquem mais fáceis de serem elaborados, o uso de fluxogramas é uma ferramenta eficiente. Em uma programação em *assembly*, que possui inúmeras peculiaridades, é difícil pensar em um programa completo sem nenhum auxilio. Nos dois *firmwares* do projeto, foi construído os seus fluxogramas para servir de suporte no desenvolvimento do trabalho.

5.3.1. Fluxogramas do *Carrier*

Este programa deve fazer a leitura multiplexada dos canais AD's do próprio microcontrolador e enviar estes dados pelo canal USART para um receptor. É importante estar atento que esta transmissão só seja feita após a passagem pelos fornos da linha de produção. Um sensor mecânico de presença pulsa na passagem e na saída deste equipamento. Com esta informação, deve-se monitorar no programa se dispositivo deste projeto está dentro ou fora do forno. Na Figura 28, pode-se analisar o fluxograma principal do programa.

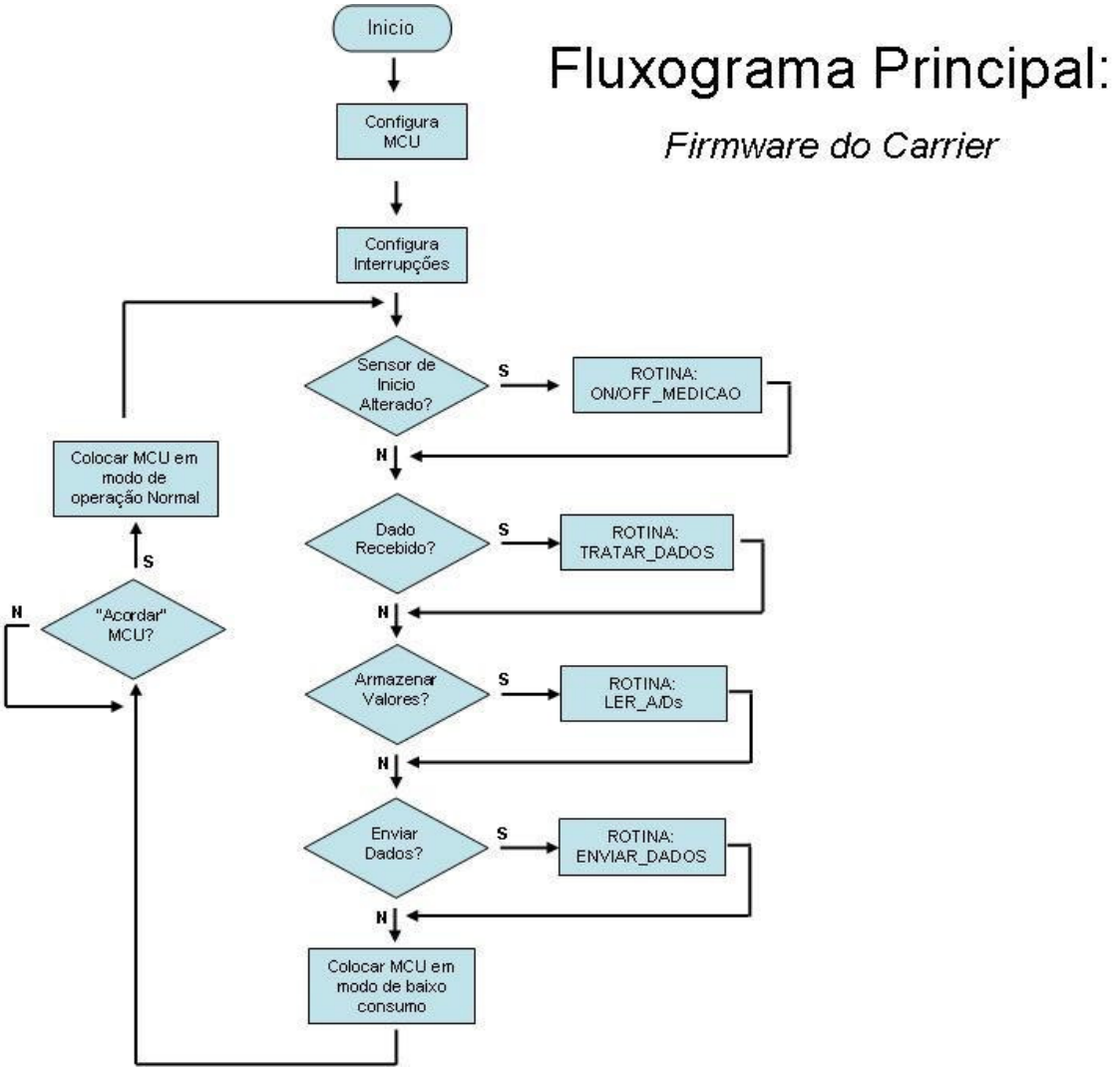

Figura 28. Fluxograma Principal do programa do *Carrier*

No inicio do programa se realiza as configurações iniciais do MCU e em seguida, tem-se a preocupação de se monitorar a entrada que está localizada o pino com o sensor que sinaliza o inicio ou final do forno. As outras atividades que este programa trata são se é o momento de se enviar ou de receber dados (e armazená-los).

Ao final de cada varredura deste "*loop*" é feita a configuração do microcontrolador para o modo de baixo consumo para que aumentar a autonomia do sistema. Fato muito importante para que se evite uma freqüência maior de troca da bateria do protótipo.

5.3.1.1. Rotina de Medição ON/OFF

Esta rotina é chamada toda vez que o sensor de presença instalado no *carrier* tem seu sinal alterado. Um pulso indica que o transportador entrou ou saiu do processo na Seção B. Na Figura 29, é mostrado o fluxograma que descreve o funcionamento deste trecho do programa.

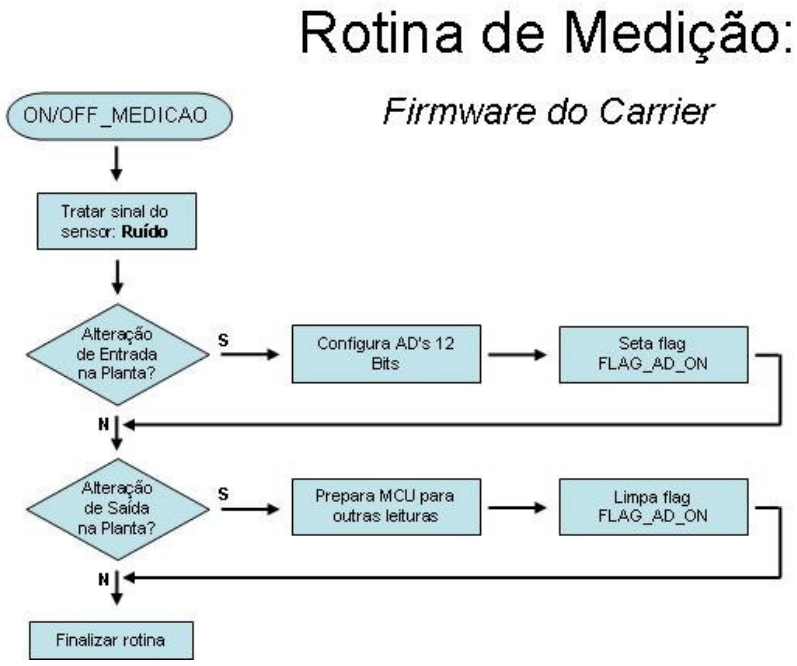

Figura 29. Rotina de habilitar medição no *Carrier*

Identificar se o sinal vindo do sensor mecânico de presença é um ruído ou não é importante para não iniciar uma transmissão dentro do forno. Um tratamento de "*bouncing*" (repique) é suficiente pra garantir que o sinal não seja um ruído. Um filtro por *software* é realizado lendo a alteração do pino e confirmando se este pulso se mantém constante depois de um determinado tempo.

Considerando que o dispositivo de medição seja ligado fora do forno, a primeira alteração deste sensor deve ser na entrada do processo. Neste caso, essa rotina prepara o MCU para iniciar a leitura dos sensores pelo periférico de conversão A/D.

Em um segundo pulso no sensor, considera-se que o dispositivo esteja saindo do forno. A rotina, então, prepara microcontrolador para enviar os dados. De acordo

com funcionários deste Setor da linha, o tempo médio de espera de um *Carrier* no trilho externo é de 10 minutos. Este é tempo útil para a transmissão.

5.3.1.2. Rotina de Recebimento de Dados

É previsto nas premissas do projeto que o usuário do *software* Supervisório possa configurar o *hardware* embarcado no *Carrier*. Estas configurações podem ser de freqüência de leitura dos sinais de temperatura, quantos e quais OPC's terão sensores monitorados, freqüência de transmissão da USART e uma "ordem" de envio de informação para o PC.

Esta comunicação é feita com o *hardware* da Estação esta estruturada no fluxograma da Figura 30.

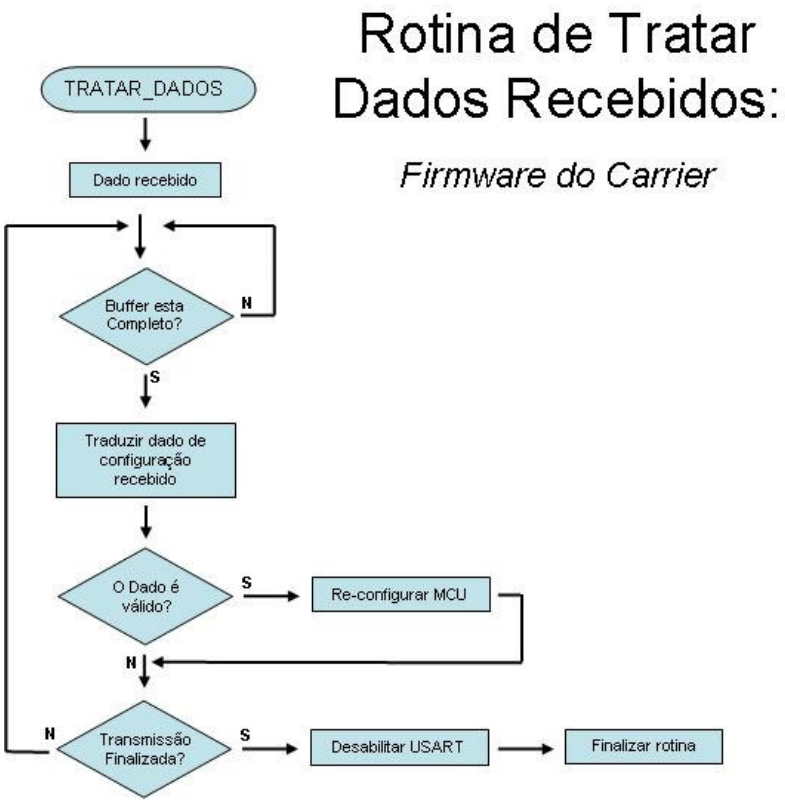

Figura 30. Rotina de tratar dados recebidos do *Carrier*

Quando esta rotina é habilitada esperasse que cada dado seja recebido por completo, já que a comunicação é do tipo serial. Este dado é traduzido em uma tabela interna no MCU para que este seja testado. Caso a informação seja válida, a

configuração é realizada para que o ciclo se reinicie até que todos os pacotes de dados sejam recebidos.

5.3.1.3. Rotina de Leitura do A/D

Após a entrada do *Carrier* no forno inicia-se a leitura dos sinais analógicos de tensão do A/D do microcontrolador. O programa deve estar preparado para ler de 1 a 32 valores de temperatura. Este número é configurado pelo *software* Supervisório, e torna-se possível pelo meio de uma multiplexação feita por *hardware*. Esta rotina é mostrada no fluxograma da Figura 31.

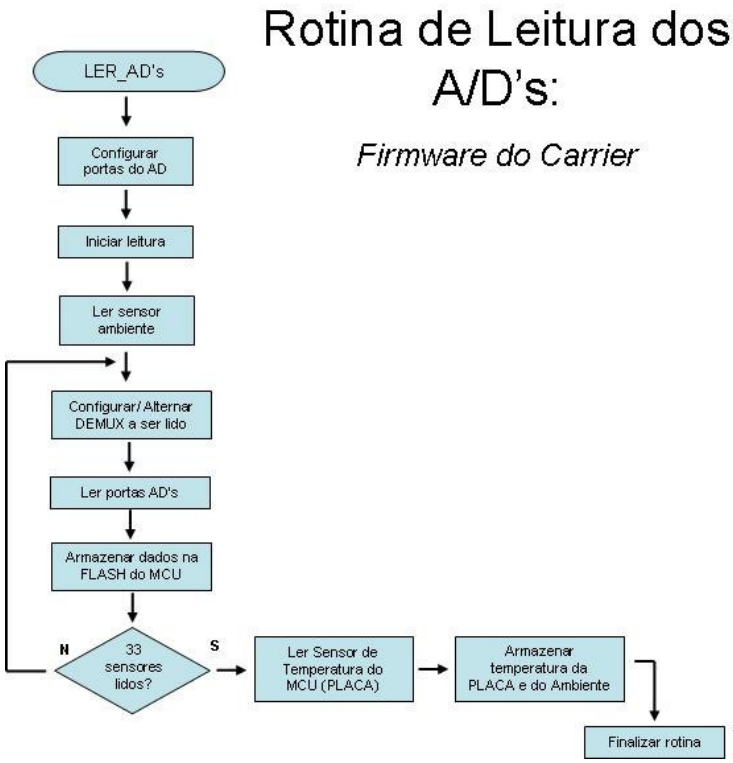

Figura 31. Rotina de leitura do AD do *Carrier*

A partir desta informação de quais sensores serão lidos, é feita uma varredura nestes que foram selecionados pelo operador. Altera-se o endereço enviado para os CI's multiplexadores (em número de 04) e se faz a leitura seqüencial dos quatro canais A/D do microcontrolador. Esta operação é repetida por todo o percurso interno do Seção B em uma freqüência, também pré-determinada pelo Supervisório. Processo.

Após sair desta rotina, o MCU é configurado para entrar em modo de baixo consumo, já que não envio de dados e leitura de A/D ao mesmo tempo. Somente com o estouro do temporizador na freqüência configurada, o controlador volta ao seu modo normal de operação.

5.3.1.4. Rotina de Envio de Dados

O envio de dados do *Carrier* para a Estação é habilitado somente fora do processo em duas situações: quando o sensor de saída for ativado ou no caso de o botão de envio ser pressionado (manualmente) no *hardware*. Esta lógica é mostrada na Figura 32.

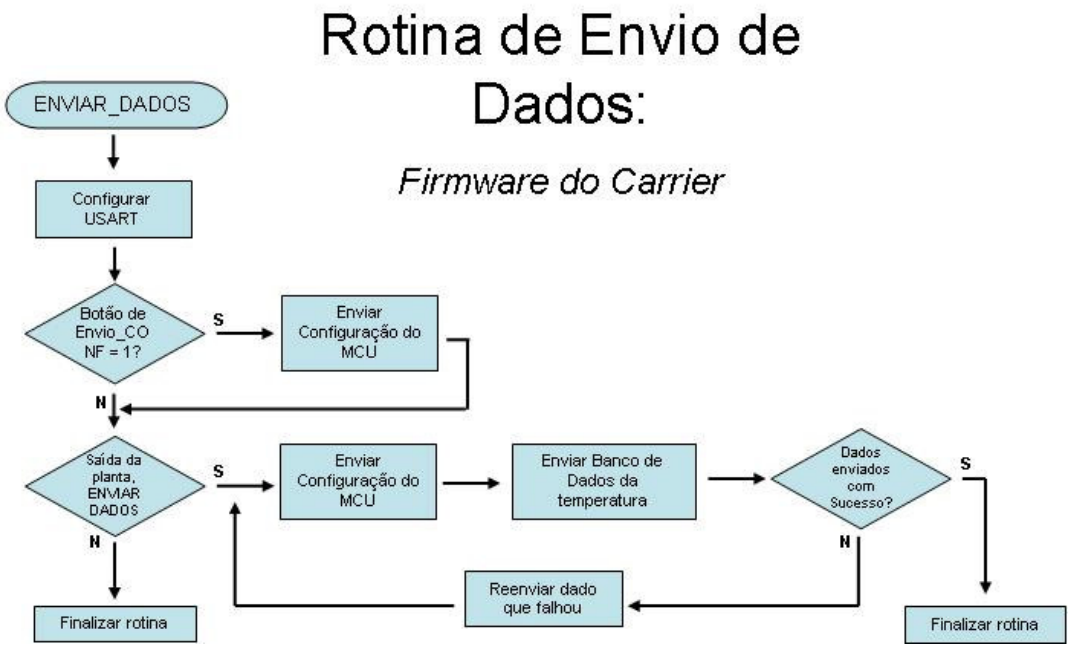

Figura 32. Rotina de envio de dados do *Carrier*

Depois da confirmação de que foi solicitado o envio de dados, o programa envia primeiro a atual configuração do *firmware* do *Carrier*. Se for uma situação de operação do dispositivo, é enviado o banco de dados com as temperaturas lidas no processo.

Na falha de envio de algum do dado por uma interferência, por exemplo, o mesmo é re-enviado para a estação. O próprio módulo de comunicação serial

assíncrona USART permite que se use um protocolo que aumenta a qualidade de da comunicação.

5.3.2. Fluxogramas da Estação

Este segundo *firmware* implementado faz parte de uma interface entre um computador pessoal e o *hardware* do *Carrier.* Ele deve servir apenas como um canal para a passagem de informação entre o protótipo de medição de temperatura e o PC. Esta função e vista da Figura 33.

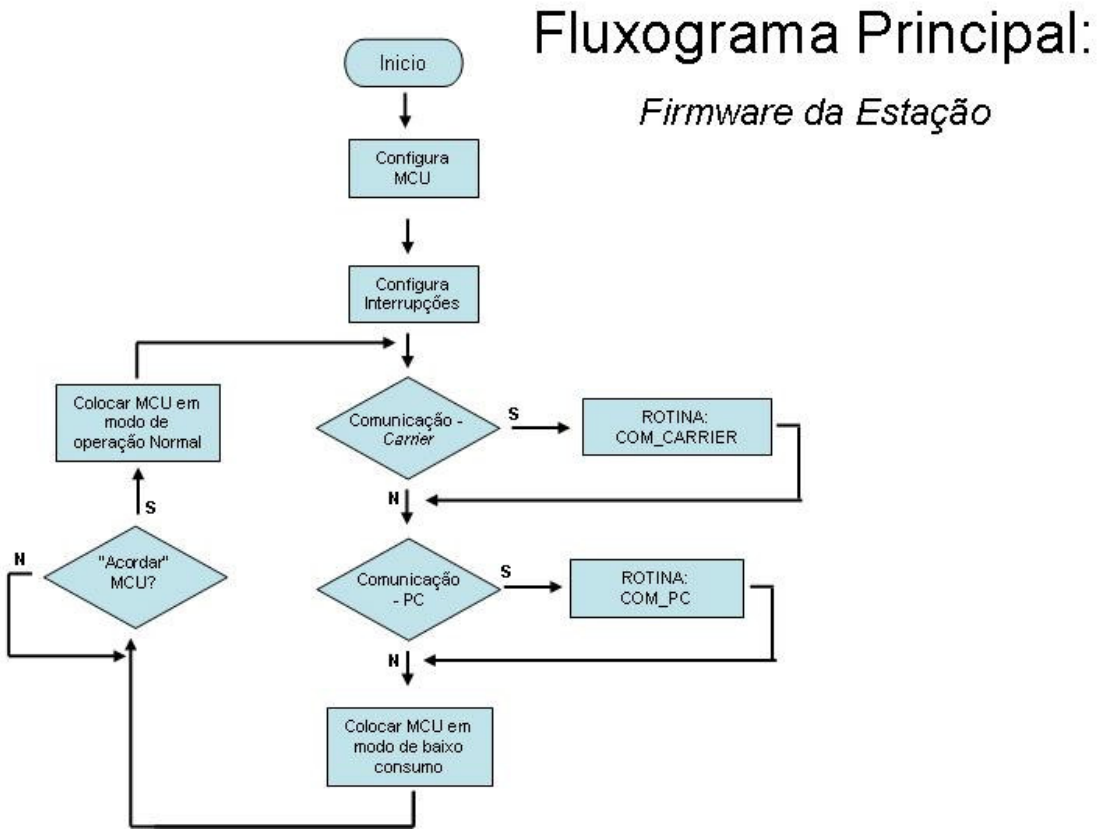

Figura 33. Fluxograma Principal do programa da Estação

A comunicação entre estes dois pontos é feita pelos dois módulos de USART do MSP430F149. Por isto no programa principal deste *firmware*, é apenas feita a análise de qual a via de comunicação que será estabelecida naquele momento.

Por questão de economia de energia, o sistema é mantido em baixo consumo já que não é iniciada uma comunicação a todo instante. O MCU volta para a operação normal em uma freqüência maior (500 ms).

5.3.2.1. Rotina de Comunicação com o *Carrier*

A comunicação Estação / *Carrier* é feita nos dois sentidos (*full-duplex*). A primeira tarefa desta rotina é identificar qual é este sentido como visto na Figura 34.

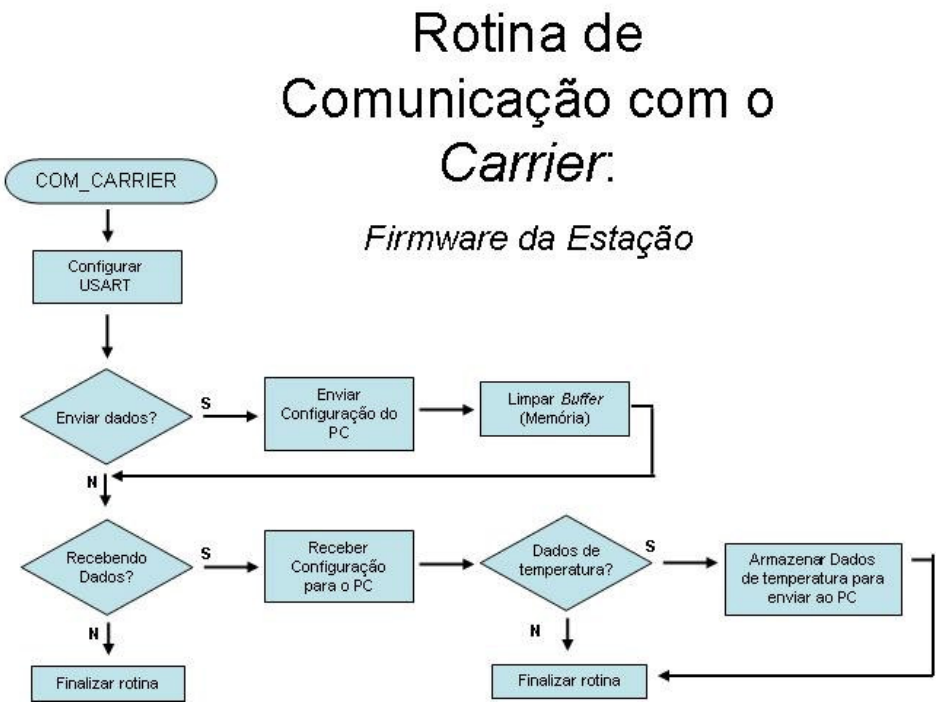

Figura 34. Rotina de Comunicação com o *Carrier* da Estação

Se for estabelecida uma comunicação de envio, significa que a Estação deve repassar uma configuração recebida do PC. A informação a ser repassada está armazenada em na memória *Flash* do MCU em um endereço conhecido. Estes dados são enviados e apagados da memória.

No caso de uma comunicação em que se receba dados, o MCU da Estação deve capturar a informação e armazenar temporariamente na memória *Flash*. Os dados recebidos serão, em primeiro lugar, as configurações atuais do *Carrier* e, caso solicitado, um banco de dados com as temperaturas dos OPC's armazenadas no processo. Esta rotina é fechada com o termino da comunicação.

5.3.2.2. Rotina de Comunicação com o PC

A rotina de comunicação com o PC tem um objetivo muito similar a rotina anterior (Comunicação com o *Carrier*). Na Figura 35, pode-se observar que a estrutura do fluxograma deste código é similar ao da Figura 34. Contudo, esta semelhança é esperada, uma vez que, a função básica deste *hardware* é de ser uma interface entre os dispositivos.

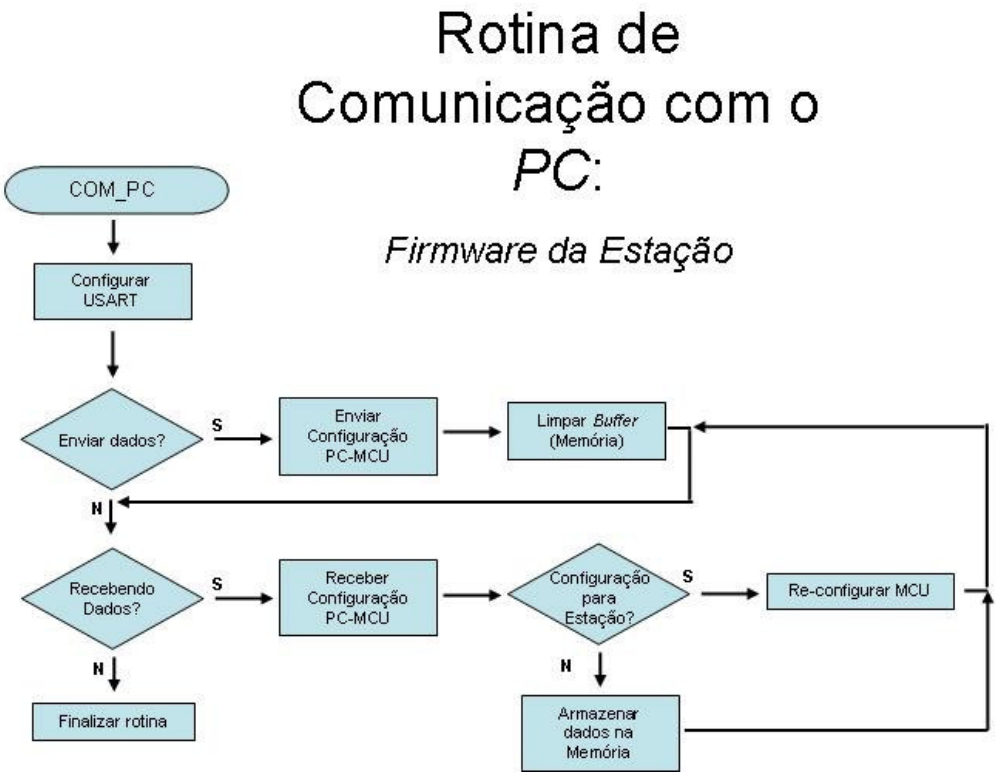

Figura 35. Rotina de Comunicação com o PC da Estação

Ao enviar dados para o PC, é informado se o dado é da Estação ou do *Carrier.* Uma situação mais crítica é no recebimento de dados, pois estes podem ser destinados ao *Carrier* ou a própria Estação. Por este motivo, estas informações são identificadas e, ou são executadas (Re-configurar *firmware* da Estação) ou armazenadas na memória do MCU para o envio das mesmas.

5.4. Projeto Inova SENAI 2008

Na semana de 10 à 15 de Junho foi realizada a Mostra Inova SENAI 2008 junto a Olimpíada do Conhecimento. São eventos bi-anuais que, este ano, foram sediados na cidade de Blumenau, Santa Catarina. O Inova SENAI reuniu os 30 melhores projetos das unidades regionais dos SENAI's do pais. Os trabalhos selecionados tinham que atender a requisitos de inovação e aplicabilidade.

Felizmente, este trabalho foi aceito e foi levado para ser apresentado em um dos maiores eventos educacionais, organizado pelo SENAI, do país.

Já foi dito que a implementação física do projeto não foi seria possível até o inicio do próximo semestre. Como seria interessante que se levasse um equipamento que ajudasse na explicação do projeto no *stand* da Mostra Inova SENAI 2008, foi desenvolvido um pequeno projeto que terminou ilustrando, com limitações, a idéia do trabalho completo.

Por questão de tempo e facilidade de aquisição, dois *hardwares* com tecnologia PTH foram montados com controladores da família PIC na gerência dos mesmos. As placas confeccionadas do circuito podem ser vistas na Figura 36.

Figura 36. *Hardwares* projetados para o Inova SENAI 2008

Mesmo estando desenvolvendo o projeto em uma plataforma diferente da que foi pensada no projeto real. Foi bastante interessante testar certos conceitos não

estudados no curso, como por exemplo, a comunicação *wireless* via rádio freqüência.

CONCLUSÃO

É importante destacar neste projeto que o trabalho foi desenvolvido em um grupo de três alunos. A condução deste projeto foi feita de forma única, mas constituído de 3 trabalhos inter-relacionados. Apesar de que todo estudo foi conduzido de modo simultâneo, em um projeto mecatrônico, as etapas de programação estão ligadas e dependem da conclusão de determinadas tarefas (eletrônicas e mecânicas).

Este é um agente inerente a um trabalho desta área que tornou tardia a evolução da criação dos códigos.

Na metodologia TheoPrax algumas etapas iniciais no projeto como: a montagem de um orçamento, a criação do conceito de solução, a elaboração da proposta, entre outros, desvirtuam o foco do trabalho na área de programação. Isso porque em um trabalho deste tipo, o interesse maior para a empresa contratante e saber detalhes da parte mecânica, por exemplo. Os *softwares* implementados seriam um diferencial para o projeto. (THEOPRAX, 2005)

Como não foram realizados os testes dos códigos em campo em *hardware*, não se pode dizer que o trabalho foi concluído. Com a liberação do orçamento pela empresa será feita neste próximo semestre, a implementação do projeto poderá ser concluída. Apesar deste trabalho de conclusão de curso estar vinculado a esta metodologia, não se pode deixar de considerar estes imprevistos ligados a terceiros.

A continuidade do trabalho pelos alunos é mais que uma questão de compromisso e responsabilidade. É o nome destes futuros profissionais que estão sendo avaliados por uma multinacional que é um potencial local de trabalho. O sucesso de um projeto deste tipo tem um impacto considerável grande na empresa e trás bons olhos a esta equipe de desenvolvedores.

O importante é deixar claro que este relatório final não simboliza a conclusão desse trabalho de pesquisa e desenvolvimento.

Como a XEROX é uma multinacional, ela possui inúmeras linhas de produção de fotoreceptores orgânicos espalhadas no mundo. A primeira proposta de trabalho que se pode aplicar os conceitos desenvolvidos neste estudo é a adaptação deste protótipo as outras linhas.

No histórico de qualidade da XEROX, eles observaram que a temperatura dos tubos de alumínio, antes mesmo da entrada na Seção B, exerce uma boa influência no processo. Sem contar que já foi exposto o interesse da empresa em criar um projeto similar para ser aplicado a Seção A-2. Esta possui características de transporte de tubos metálicos um pouco particulares, mas não é uma zona classificada (risco de explosão).

Era explicito o interesse e a expectativa que este projeto piloto tenha seja concluído com sucesso para que estas importantes parcerias indústria-faculdade se consolidem.

O inicio do projeto dos programas está quase concluído por depender ainda de algumas definições de *hardware*. Contudo, o trabalho levado à Mostra Inova SENAI 2008 permitiu a avaliação do conceito desenvolvido até aqui. Mesmo sendo em plataformas de trabalho diferentes, o estudo de estratégias de transmissão e o tratamento dos sinais de transmissão foi um ponto que só começou a ser analisado depois destes testes.

Outra preocupação que apareceu como aprendizado que será melhora pensado na implementação são as interferências na comunicação *wireless*. A equipe não imaginava que os testes em um ambiente e com equipamentos diferentes iram contribuir tanto para melhorias no projeto.

Um projeto desta magnitude proporcionou um ganho em novas áreas de conhecimento. E a idéia de realizar um trabalho de conclusão de curso em conjunto com três colegas foi muito proveitosa, pois isto justificou o emprego de várias das propostas de gerenciamento de projetos disseminadas no Modelo *TheoPrax*.

A avaliação do trabalho foi focada apenas nas etapas que foram concluídas até então.

REFERÊNCIAS

ALMEIDA, C. V. R.,. **Equipamento para estimação do Torque em Motores de Indução Trifásicos pelo Método do Escorregamento auxiliado pela Análise Espectral do Sinal de Corrente do Estator - Desenvolvimento e Implementação**, 2007. Dissertação (Mestrado Engenharia Elétrica) - Escola Politécnica, Universidade Federal da Bahia, Salvador . 2007.

BEGA, Egidio Alberto. **Instrumentação industrial. 2. ed.**. Rio de Janeiro: Interciência, 2006.

BRAGA, Newton C. **Eletrônica Básica para Mecatrônica**. São Paulo: Editora Saber, 2005.

CANZIAN, Edmur, 2002, **Comunicação Serial - RS232**, Minicurso, CNZ Engenharia e Informática Ltda., Cotia, São Paulo.

FIEB, **Guia da Industrias do Estado da Bahia**, 2007. Disponível em: www.fieb.org.br/guia/dados_industria.asp?industria=6. Acessado em Novembro de 2007.

GACLI, Centro de Computação UNICAMP, 1998, **Introdução à Linguagem C**, Versão 2.0, Divisão de Serviços à Comunidade UNICAMP, São Paulo.

JÁCOBO, J. A. E.. **Desenvolvimento de um Robô Autônomo Móvel versátil utilizando Arquitetura Subsumption**. 2001. Tese (Doutorado) - UFSC, Santa Catarina, 2001.

LIRA, Patrícia F. A., 2006, **Análise para inclusão do fluxo de desenvolvimento de chips no ipProcess**, Trabalho de Graduação, UFPE, Pernambuco.

MATIC, Nebjosa e ANDRIC, Dragan, 2000, **The PIC Microcontroler Book 1 – Paperback**. Traduzido para Português por Alberto Jerônimo. Disponível em: http://www.mec.ua.pt/activities/graduationprojects/graduationprojectpages/2003-2004/H1/PICs/picbook/pt/00.htm. Acesso em Dezembro, 2007

NERYS, Prof. Jose W. L., 2006, **Notas de aula de Microprocessador 8085/8088**, UFG, Goiânia.

NETO, Dr. Eng. Antonio J. Steidle e ZOLNIER, Dr. Eng. Sérgio, 2006, **Avaliação de circuito eletrônico para medição de temperatura em instalações agrícolas por meio da porta paralela de um computador**, Artigo Cientifico. Minas Gerais.

OLIVEIRA, André S. e ANDRADE, Fernando S., 2006, **Sistemas Embarcados: *Hardware* e *Firmware* na prática**, Ed. Érica, São Paulo.

OMEGA, Engineering INC, 2003. Disponível em: www.omega.com. Acesso em Dezembro, 20007

PALHARES, Rogério Boucault, 2002, **Estudo do Funcionamento de uma Máquina Fotocopiadora**, Unicamp, SP.

PEREIRA, Fabio, 2005, **Microcontroladores MSP430: Teoria e Prática**, Erica. São Paulo.

POLONSKII, Milkhail M., 1996, **Introdução à Robótica e Mecatrônica**. EDUCS: Caxias do Sul.

SCHWAMBACH, Rodrigo Evaldo, 2004, **Microcontroladores 8051 e PIC**, Apostila Didática do SENAI CIMATEC. Bahia.

SILVA, Matheus de Oliveira e, 2008, **Projeto de um dispositivo de medição de temperatura automatizado aplicado a estação de tratamento superficial da XEROX: Solução Mecânica e de Sensoriamento**, Trabalho de Conclusão de Curso, Faculdade de Tecnologia SENAI CIMATEC, Bahia.

SOARES, Ulysses Moraes, 2008, **Projeto de um dispositivo de medição de temperatura automatizado aplicado a estação de tratamento superficial da XEROX: Projeto e Simulação do *Hardware***, Trabalho de Conclusão de Curso, Faculdade de Tecnologia SENAI CIMATEC, Bahia.

SOCORRO, Marlene Santos, 2005, **A construção de mapas conceituais como estratégia de verificação da aprendizagem**. Artigo científico apresentado ao XVI Simpósio Nascional de Ensino de Física, 2005, Bahia.

TERASAKI, Seishi, YUTAKA, Ikeda and OBINATA, Tochiki, 2006, **Organic Photoconductor for Printers**, Fuji Electric Journal. Japan.

TEXAS, Instruments, 2000, **MSP430x13x, MSP430x14x, MSP430x14x1 Mixed Signal Microcontroller**, Revisado em Junho de 2004.

TEXAS Instruments, **MSP430**, 2007. Disponível em: http://www.ti.com/sc/brasil/ws_msp430.htm. Acesso em Dezembro, 2007

THEOPRAX, Metodology, 2005. Disponível em: http://www.theoprax.com/. Acesso em Outubro, 2007

XEROX, Corporation, 2007,. Disponível em: www.xerox.com. Acesso em Dezembro, 2007

ANEXO 1: Set de Instruções do MSP430

Conjunto de instruções em *Assembly* da família MSP430. A descrição e operação (caso exista) de cada *op-code* é mostrada junto com possíveis alterações nos *flags* da CPU.

Um asterisco indica que dependendo do resultado do processamento, possa ocorrer uma mudança no respectivo valor de: V, N, Z e/ou C. Um hífen simboliza que não haverá alteração no *flag*. Por fim, a indicação na tabela com os dígitos 0 e 1 mostra que independente da resposta da CPU, o valor do sinalizador será o apresentado.

Mnemônico		Descrição	Operação	*Flags*			
				V	N	Z	C
ADC(.B)	dest	Adiciona C ao destino	dest = dest + C	*	*	*	*
ADD(.B)	fonte,dest	Adiciona a fonte ao destino	dest = fonte + dest	*	*	*	*
ADDC(.B)	fonte,dest	Adiciona a fonte mais o C ao destino	dest = fonte + C + dest	*	*	*	*
AND(.B)	fonte,dest	Operação lógica AND da fonte com o destino	dest = fonte AND dest	0	*	*	*
BIC(.B)	fonte,dest	Apaga os *bits* da fonte no destino	dest = not(fonte) AND dest	-	-	-	-
BIS(.B)	fonte,dest	Seta os *bits* da fonte no destino	dest = fonte OR dest	-	-	-	-
BIT(.B)	fonte,dest	Testa os *bits* da fonte no destino	dest AND fonte	0	-	-	-
BR	dest	Desvia para o destino	PC = dest	-	-	-	-
CALL	dest	Chamada de sub-rotina no destino	Pilha = PC + 2, PC = dest	-	-	-	-
CLR(.B)	dest	Apaga o conteúdo do destino	dest = 0	-	-	-	-
CLRC		Apaga o *flag* C	C = 0	-	-	-	0
CLRN		Apaga o *flag* N	N = 0	-	0	-	-
CLRZ		Apaga o *flag* Z	Z = 0	-	-	0	-
CMP(.B)	fonte,dest	Compara a fonte e o destino	dest - src	*	*	*	*
DADC(.B)	dest	Adição decimal do C com o destino	dest = dest + C (decimal)	*	*	*	*
DADD(.B)	fonte,dest	Adição decimal da fonte com o C ao destino	dest = fonte + C + dest (decimal)	*	*	*	*
DEC(.B)	dest	Decrementa o destino	dest = dest - 1	*	*	*	*
DECD(.B)	dest	Decremento duplo do destino	dest = dest - 2	*	*	*	*
DINT		Desabilita as interrupções	GIE = 0	-	-	-	-
EINT		Habilita as interrupções	GIE = 1	-	-	-	-
INC(.B)	dest	Incrementa o destino	dest = dest + 1	*	*	*	*
INCD(.B)	dest	Incremento duplo do destino	dest = dest + 2	*	*	*	*
INV(.B)	dest	Complementa os *bits* do destino	dest = NOT (dest)	*	*	*	*
JC/JHS	label	Pula se C = 1 / Pula se maior ou igual		-	-	-	-
JEQ/JZ	label	Pula se igual / Pula se Z = 1		-	-	-	-
JGE	label	Pula se maior ou igual		-	-	-	-
JL	label	Pula se menor		-	-	-	-
JMP	label	Pula	PC = PC + 2 * *offset*	-	-	-	-
JN	label	Pula se negativo (se N = 1)		-	-	-	-
JNC/JLO	label	Pula se C = 0 / Pula se menor		-	-	-	-
JNE/JNZ	label	Pula se diferente / Pula se Z = 0		-	-	-	-
MOV(.B)	fonte,dest	Copia o conteúdo de fonte para destino	dest = fonte	-	-	-	-
NOP		Nenhuma operação		-	-	-	-
POP(.B)	dest	Lê um dado da pilha e guarda no destino	dest = @SP, SP = SP + 2	-	-	-	-
PUSH(.B)	fonte	Armazena o conteúdo da fonte na pilha	SP = SP – 2, @SP = fonte	-	-	-	-

Mnemônico		Descrição	Operação	Flags			
				V	N	Z	C
RET		Retorna de uma sub-rotina	PC = @SP, SP = SP + 2	-	-	-	-
RETI		Retorna de uma interrupção		*	*	*	*
RLA(.B)	dest	Rotação aritmética do destino à esquerda		*	*	*	*
RLC(.B)	dest	Rotação do destino à esquerda através do *Carry*		*	*	*	*
RRA(.B)	dest	Rotação aritmética do destino à direita		0	*	*	*
RRC(.B)	dest	Rotação do destino à direita através do *Carry*		*	*	*	*
SBC(.B)	dest	Subtrai o C invertido do destino	dest = dest + 0xFFFF + C	*	*	*	*
SETC		Seta o *flag* C	C = 1	-	-	-	1
SETN		Seta o *flag* N	N = 1	-	1	-	-
SETZ		Seta o *flag* Z	Z = 1	-	-	1	-
SUB(.B)	fonte,dest	Subtrai a fonte do destino	dest = dest – fonte	*	*	*	*
SUBC(.B)	fonte,dest	Subtrai a fonte e o C invertido do destino	dest = dest – fonte – NOT(C)	*	*	*	*
SWPB	dest	Troca os *bytes* do destino		-	-	-	-
SXT	dest	Extensão do sinal do destino		0	*	*	*
TST(.B)	dest	Testa o destino	dest = 0 ?	0	*	*	1
XOR(.B)	fonte,dest	Operação lógica XOR da fonte com o destino	dest = fonte XOR dest	*	*	*	*

Tabela 1. Lista dos Mnemônicos da Família MSP430