UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELETROTÉCNICA
CURSO DE TECNOLOGIA EM AUTOMAÇÃO INDUSTRIAL

PROJETO DE UM *DATALOGGER* DE BAIXO CUSTO DEDICADO PARA ESTAÇÃO METEREOLÒGICA

CORNÉLIO PROCÓPIO
2016

MARCELO VITORINO

PROJETO DE UM *DATALOGGER* DE BAIXO CUSTO DEDICADO PARA ESTAÇÃO METEREOLÒGICA

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Diplomação, do Curso Superior de Tecnologia em Automação Industrial do Departamento Acadêmico de Eletrotécnica– DAELT – da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Tecnólogo.

Orientador: Prof. Me. Ângelo Feracin Neto

CORNÉLIO PROCÓPIO
2016

MARCELO VITORINO

PROJETO DE UM *DATALOGGER* DE BAIXO CUSTO DEDICADO PARA ESTAÇÃO METEREOLÓGICA

Trabalho de conclusão de curso apresentado às 14h do dia 14 de Setembro de 2016 como requisito parcial para a obtenção do título de Tecnólogo em Automação Industrial da Universidade Tecnológica Federal do Paraná. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

_____
Prof. Me. Angelo Feracin Neto
Professor Orientador
UTFPR/ Campus Cornélio Procópio

_____
Prof. Esp. José Roberto Shimazaki
Professor Convidado
UTFPR/ Campus Cornélio Procópio

_____
Prof. Dr. Luís Marcelo Chiesse da Silva
Professor Convidado
UTFPR/ Campus Cornélio Procópio

A Folha de Aprovação assinada encontra-se na Coordenação do Curso.

A Deus por ter me mostrado o caminho

A minha família e amigos por todo apoio

E a todos que contribuíram para o bom andamento da pesquisa

AGRADECIMENTOS

A trajetória até aqui foi longa e dura, e para que fosse possível vencê-la foi necessário o auxílio de diversas pessoas, que passaram pela minha vida e ajudaram o fardo a se tornar mais leve, assim eu poderia citar diversos nomes, contudo apenas digo que agradeço imensamente pelo seu apoio, me incentivando e me dando conselhos, também não me deixando desistir jamais. Aos amigos, que sempre proporcionaram alegria, e companheirismo.

Agradeço imensamente aos meus familiares cujo apoio foi de extrema importância, sem eles seria difícil continuar o caminho e percorrê-lo firmemente, toda ajuda que me deram seja ela financeira ou emocional, me fez perceber que por mais difícil que seja o caminho deve-se segui-lo decididamente.

À minha esposa Priscila Machado Cardoso Vitorino pelo carinho, amor, compreensão, paciência e pelo amparo que sempre me deu. A ela que incessantemente me ajudou em todos os momentos tenham sido eles fáceis ou difíceis, mas que jamais me deixou desistir.

Ao meu sogro Lourival Fidelis Cardoso que partiu recentemente, mas que desde o momento em que entrei para família e enquanto viveu, acreditou em mim me dando apoio e auxílio nunca me deixando desistir.

Por fim agradeço ao Professor Ms. Ângelo Feracin Neto que se dispôs de seu tempo para me orientar, que ofereceu seu conhecimento como sustento desta pesquisa, mostrando me os rumos certos a serem seguidos, agradeço pela competência, paciência e confiança em mim depositada.

VITORINO, Marcelo. Projeto de um *datalogger* de baixo custo dedicado para estação meteorológica, 2016. 82 páginas. Trabalho de Conclusão de Curso. Universidade Tecnológica Federal do Paraná. Departamento Acadêmico de Eletrotécnica - Curso de Tecnologia em Automação Industrial, Paraná, Cornélio Procópio, 2016.

RESUMO

Esta pesquisa apresenta uma abordagem sobre a elaboração de *datalogger*. Em razão da existência de um canal USB para troca de dados, a tecnologia estudada nesse projeto foi o microcontrolador PIC 18F4550 da Microchip, utilizando-se da linguagem C para programação. O equipamento elaborado é de baixo custo com foco no auxílio em estação meteorológica. Em virtude da necessidade de criação de diversos dispositivos de armazenamento de dados que sejam acessíveis a todo perfil de usuário, estabeleceu como objetivo principal criar, dentro das exigências e necessidades mencionadas, um dispositivo de registro – *datalogger*. Considerando as carências dos pequenos produtores de grandes centrais meteorológicas em obter auxílio em suas atividades cotidianas, sobretudo na prevenção de fenômenos climáticos, a combinação de dados em curto prazo possibilita ao agricultor um melhor planejamento de seus plantios, evitando danos às culturas e uma organização técnicas das mesmas. Portanto, o dispositivo proporciona ao agricultor diversos benefícios, resultados positivos e eficazes, tornando-se uma excelente ferramenta para que a solução de problemas rotineiros ou inesperados ocorra de forma mais precisa e autônoma.

Palavras-chave: Dispositivos. Armazenamento de Dados. *Datalogger.* Agricultura. Beneficiamento Agrário. Pequeno Produtor. PIC 18f4550. USB.

VITORINO, Marcelo. Low cost automatic meteorological station datalogger project. 2016. 82 paginas. Trabalho de Conclusão de Curso. Universidade Tecnológica Federal do Paraná. Departamento Acadêmico de Eletrotécnica - Curso de Tecnologia em Automação Industrial, Paraná, Cornélio Procópio, 2016.

ABSTRACT

This research presents an approach about a datalogger development. Considering that there was an USB channel for data exchanging, the technology studied in this project was a microcontroller PIC 18F4550 from Microchip, using C programming language. The equipment developed is a low cost one for helping in meteorological stations. Due to the necessity of creating different devices for data storage accessible to all user profiles, it was aimed to create a *datalogger* appropriate to the specific needs mentioned. Because of the lack of information and support to small producers in their daily routines, especially in relation to weather conditions, data combination in short term allows them a better planning on dealing with their crops, preventing losses and improving technic organization. In conclusion, the device offers the producer diverse benefits, positive e efficient results, becoming an excellent tool to solve ordinary or unexpected problems in an accurate and autonomic way.

Keywords: Devices. Data Storage. Datalogger. Agriculture. Crop Processing. Small Producer. PIC 18f4550. USB.

LISTA DE ABREVIATURAS E SIGLAS

ADC	*Analog to Digital Conversion*
ASCII	*American Standard Code for Information Interchange*
ANSI	*American National Standards Institute*
BCD	*Binary Coded Decimal*
CCS	*Custer Computer Services inc.*
CDC	*Communications Devices Class*
CI	Circuito Intergado
CPU	*Central Processing Unit* - Unidade Central de Processamento
EEPROM	*Electrically-Erasable Programmable Read-Only Memory*
EUSART	*Stands for Enhanced Universal Synchronous Asynchronous Receiver Transmitter*
GND	*Ground*
HSPLL	*High Speed With Phase Locked Loop*
I^2C	*inter-itegrated Circit* - Circuito Inter integrado
LED	*Light Emitting Diode*
OTP	*One Time Programmable* - Um tempo programável
PIC	*Programmable Interface Controller*
PLL	*Phase Locked Loop*
PWM	*Pulse-Width Modulation* - Modulação por largura de pulso
RAM	*Random Access Memory* - Memória de Acesso Aleatório
ROM	*Read Only Memory* - Memória apenas para leitura
RTC	*Real Time Clock*
SCL	*Serial Clock*
SDA	*Serial Data*
SPI	*Serial Perepheral Interface* - Interface periférica de série
USART	*Universal Synchronous Receiver Transmiter*
USB	*Universal Serial Bus*

LISTA DE FIGURAS

LISTA DE TABELA

SUMÁRIO

1 INTRODUÇÃO

O homem sempre foi dependente da natureza e das atividades que a envolvem, buscando desta forma desenvolver meios para um aumento de produtividade, contudo apesar da busca constante por melhoramentos nas atividades agrárias e uma produção eficiente e quantitativa, os pequenos produtores principalmente, enfrentam diversos problemas.

Desde políticas mal elaboradas, que não consideram as realidades dos agricultores, o não reconhecimento da importância dos trabalhos deste tipo de agricultura, falta de recursos, até tecnologias inadequadas e escassas que não colaboram com o bom desenvolvimento das atividades agrárias.

Dentro deste quesito, vale ressaltar que a agricultura está diretamente ligada a dados meteorológicos, pois são eles que determinam as culturas a serem produzidas no momento mais propício, e ao se falar em tais dados a tecnologia é, porém ainda mais escassa.

A agricultura é a atividade mais dependente do clima, e por este motivo torna-se fundamental que se hajam tecnologias pertinentes e eficazes para auxiliar os agricultores na detecção de diversos problemas que possam atingir a produção. É de suma importância que se invista em tecnologias capazes de obter e repassar informações mais precisas a cerca da influência do tempo nas atividades agricultáveis.

Atualmente percebe-se que há uma necessidade de criação de materiais, dispositivos e mecanismos acessíveis a todo perfil de usuário, apesar do mercado já apresentar tecnologias baratas, de baixo custo e acessível, as pesquisas nessas áreas são sempre relevantes, pois é, preciso manter e fomentar ainda mais a criação destes mecanismos e deixá-los sempre atualizados para que os usuários possam ter em mãos um objeto eficaz.

Assim se apresenta como principal objetivo deste trabalho a criação de um *datalogger* de baixo custo, visando principalmente o acesso de seus maiores usuários, os pequenos agricultores.

Desta forma o instrumento a ser criado na presente proposta apresenta como características técnicas um circuito integrado de Micontroladores PIC (*Programmable Interface Controller*) 8 Bits, chip Eeprom (*Electrically-Erasable*

Programmable Read-Only Memory) , sensores e entradas entre outros aparatos que darão subsídios para o equipamento *datalogger* entrar em funcionamento.

Com bases nestas informações ressalta-se que a presente proposta se fundamenta na apresentação de requisitos básicos para dar um bom andamento a pesquisa como apontamentos de seus objetivos gerais e específicos demonstrando o que se busca com a pesquisa do referido dispositivo, justificativa, demonstrando a relevância do projeto e a descrição dos elementos que irão compor todo o equipamento a ser criado.

Desta forma pretende-se desenvolver um instrumento eficaz que vai além do baixo custo, mas que seja também capaz de suprir as principais necessidades de dados dos seus usuários, dando a eles o suporte necessário para manter suas atividades.

Dentro dos intuitos da pesquisa, engloba-se também de forma secundária, o aperfeiçoamento do pesquisador, buscando um maior conhecimento em tecnologias micro controláveis e desenvolvimento de outras formas de tecnologias que possam auxiliar e dar suporte a eventuais pesquisas nesta área.

1.1 OBJETIVOS

1.1.1 Objetivos Gerais

O principal objetivo deste trabalho consiste em desenvolver um dispositivo de registro (*datalogger*), priorizando o baixo custo do equipamento como uma alternativa acessível de coleta, armazenamento e transferências de dados para uso do pequeno agricultor para registro de dados meteorológicos.

1.1.2 Objetivos Específicos

Com a finalidade de atingir o que foi proposto, seguem os objetivos específicos da presente pesquisa:

- Estudar o microcontrolador.
- Definir as variáveis a serem medidas.
- Definir a forma de armazenamento.
- Elaborar a forma de armazenamento do dispositivo.
- Implementar um canal de comunicação.
- Desenvolver uma fonte de alimentação.

1.2 JUSTIFICATIVA

Levando em conta a importância dos dados meteorológicos para a agricultura e assim para os pequenos agricultores, dos quais não disponibilizam de equipamentos e/ou grandes centrais meteorológicas de alto custo para auxilia-los em suas atividades, se viu a necessidade e a importância do desenvolvimento de mecanismos e produtos de tecnologias eficazes e precisos que possam orientá-los no cotidiano de suas funções agricultáveis.

Dentro destes mecanismos, busca-se a elaboração de um produto que seja capaz de auxiliar o agricultor a prever fenômenos climáticos que possam alterar o ciclo de sua produção, e desta forma que o mesmo possa planejar o plantio e evitar ou se prevenir de danos às culturas.

Assim, busca-se levar ao produtor uma combinação de dados em curto prazo que o auxilie em funções tais como, organização de suas tarefas na agricultura de modo que o mesmo possa ter em mente quando vai preparar a terra para cultivo, quando iniciará o processo de plantação, evitando que sofra prejuízos.

Além da organização em caráter técnico de plantio, a precisão de dados através da utilização de um equipamento acessível, também auxilia na redução de custo, uma vez que se possa ter um bom planejamento através de um histórico dos dados adquiridos com o equipamento.

Portanto com a criação e desenvolvimento do equipamento de baixo custo, prevê a busca de dados precisos e completos de auxílio ao agricultor em problemas ou organização e planejamento do cultivo, dando a ele mais autonomia e eficácia em suas atividades agrícolas.

2 REFERENCIAL TEÓRICO

2.1 A IMPORTÂNCIA DE ESTUDOS CLIMÁTICOS NA ATIVIDADE AGRÁRIA

Ao se falar em agricultura, não é possível desvincular sua imagem do fator econômico, atualmente todo o pensar agricultável é voltado para incremento em rendimentos, risco de insucesso, redução de custo, aumento de produção e assim pensa-se diariamente no uso minucioso dos recursos financeiros, em como serão aplicados, suas finalidades entre outros.

Desta forma um dos quesitos que mais influência na economia de uma atividade agrária é o clima, que por ser apenas previsto e não controlável, gera na maioria das vezes ao prejuízo do proprietário das safras. Portanto é preciso cada vez mais se ater a estudos climáticos para prever não apenas qual fenômeno climático se aproxima, mas também prever riscos, prejuízos possíveis e até mesmo uma prevenção para proteção da lavoura.

Pode se verificar essa importância dos estudos climáticos da seguinte forma: “há muito tempo o homem se interessa pelo tempo atmosférico (...) Isto é um fato não surpreendente e esperado, pois as condições atmosféricas influenciam o homem em suas diferentes e numerosas formas de atividades.” (Ayoade, 1996, p.4)

Neste caso daremos destaque como mencionado no início para essa importância voltada à atividade da agricultura, tal atividade se define por uma atividade humana a fim de se obter aproveitamento de frutos e produtos vindos da terra, conforme explica Rocha (1999) e assim a mesma não se cumpre naturalmente, ela depende da colaboração dos diversos elementos que compõem a natureza, como radiação solar, precipitações, temperatura, umidade e etc.

Todos esses elementos estão, portanto ligados a questão do clima, e são eles que determinam os estágios da cadeia da produção agrícola, incluindo os estágios totalmente dependentes do homem, da colheita à comercialização do produto.

Segundo Torres (20--) o clima exerce influência direta na agricultura por definir variáveis diárias, sazonais e anuais, também tem de se levar em conta

neste fator os microclimas em que as lavoras se desenvolvem que são formados pelos solos e proximidades com outros terrenos tornando se diferentes em cada delimitação de terreno.

Torres (20--) evidencia ainda que para ser possível o desenvolvimento de uma lavoura, é necessário que se escolha minuciosamente as características climas que envolvem tipo de solo adequado, o calor, a umidade a ocorrência de precipitações no local e estudo detalhado para verificar se há possibilidade de dessas três naturais que possam vir a prejudicar as plantações, leva-se em conta o fator de iluminação e as incidências solares.

2.2 A INFLUÊNCIA CLIMÁTICA NA AGRICULTURA

2.2.1 Radiação Solar

A radiação solar é um dos fatos mais importantes para as lavouras, pois é ela a responsável pela realização da fotossíntese, responsável pelo desenvolvimento e crescimento da planta.

Segundo Casaroli et. al. (2007), radiação definida por conjuntos e ondas eletromagnéticas que incidem sobre a terra, ocorrendo nesta incidência o processo de espalhamento delas na superfície terrestre, do qual 51% são aproveitadas pelas plantas, que transforma essa radiação em elementos essenciais para sua sobrevivência.

2.2.2 Temperatura

Esta variável climática define e afeta diversas etapas do crescimento das plantas na agricultura, isto porque, alguns tipos de culturas necessitam de altas temperaturas para que possam se desenvolver, enquanto outras necessitam de baixas temperaturas, segundo Torres (20--,p.3):

> “O resfriamento prolongado das plantas, com temperaturas acima do ponto de congelamento, retarda o crescimento vegetal e pode até matar as plantas adaptadas somente a condições quentes. Apesar de que o resfriamento não possa matar diretamente as células vegetais, ele reduz o fluxo de água das raízes e então interfere na transpiração e na nutrição do vegetal.”

Assim, portanto pode-se ressaltar que baixas temperaturas e altas temperaturas podem ser prejudiciais às plantas desde que estas sejam sensíveis a temperatura vigente em que ela foi plantada.

2.2.3 Umidade

Umidade desempenha papel vital para as plantações, contudo evidencia-se que nem água insuficiente ou em excesso são favoráveis as culturas.

Nesta variável podemos levar em conta a umidade do ar e a umidade do solo, a primeira por que afeta parte dos processos de desenvolvimento da planta, a transpiração**,** importante para a redução da temperatura da folha, e a absorção dos nutrientes, essencial para o crescimento da planta. Já a umidade do solo.

2.2.4 Vento

O vento trata-se do movimento do ar em relação a superfície terrestre, assim como todas as variáveis climáticas já descritas, ele possui benefícios e malefícios as culturas agrícolas.

Ele age como fator importante na dispersão das plantas, sendo fonte de colonização de novas áreas, tem também uma importante função para algumas plantas, auxiliando-as em seu crescimento, conforme indicam Resende e Resende Junior (2011, p.2):

> “Os ventos afetam o crescimento das plantas sob três aspectos: transpiração, absorção de CO2 e efeito mecânico sobre as folhas e

> ramos. A transpiração aumenta com a velocidade do vento até atingir certo patamar; a partir do qual não se verifica mais alterações no valor de transpiração. A exata relação entre vento e transpiração varia e é dependente de cada espécie.”

Contudo apesar de seus benefícios algumas delas são sensíveis a este fenômeno, sendo o vento neste caso um disseminador de perdas em culturas como cafezais e bananais. Portanto é preciso atentar-se a ventos frios, contínuos e intensos nas regiões agricultáveis, pois ele pode causar danos na lavoura devida sua pressão direta nos cultivos e também favorecendo erosão do solo entre outros.

2.2.5 Seca

Assim como a umidade exerce em sua maioria um papel benéfico para as plantações a seca é extremamente o contrário, ela causa nas culturas um déficit de umidade armazenada no solo, causando assim um problema no suprimento de das necessidades hídricas da planta para seu bom desenvolvimento, sendo ela portanto um dos fatores que mais assustam e causam prejuízos as agriculturas.

Observou-se, portanto que o clima é predominantemente um agente influenciador da agricultura, podendo ser ele benéfico e auxiliando para um desenvolvimento eficaz das culturas existentes ou também um agente maléfico e destruidor, levando a devastação e perda total, causando sérios prejuízos.

Para evitar ou prever dessas três e assim auxiliar o agricultor, nestes quesitos são tantas as tecnologias e avanços científicos utilizados para tal ação, tornando o mercado destas tecnologias sempre promissor.

Devido essas ajudas tecnológicas terem um custo alto e não ser acessível as muitos produtores, vê se a necessidade de utilização de um equipamento mais barato e aberto a todos, como os dispositivos de armazenamento de dados, que são de vital importância nesta área.

2.3 *DATALOGGER*

Considerando, porém, os dispositivos que armazenam dados, temos como principal ferramenta o chamado *Datalogger* que se trata de um dispositivo que funciona como auxiliar, ou seja, ele coleta dados de outro instrumento de registro em que está conectado. Mais especificamente podemos definí-lo como:

> É um dispositivo de aquisição de dados, programável que interpreta sinais elétricos que os sensores produzem na entrada, convertendo em várias unidades de medidas, que são armazenadas em seu módulo de memória. (DIAS, 2007 p. 22)

São também chamados de aquisidor de dados e registrador gráfico os dados adquiridos e guardados por ele encontram-se dentro do espaço de tempo que seja necessário ao pesquisador, além disto, variáveis informacionais que ele guarda também dizem respeito ao interesse do pesquisador, guardando apenas aquilo que lhe for necessário. Na imagem 1 pode se verificar alguns modelos de aparelhos *datalogger*:

Figura 1 – Modelos de *Datalogger* (Pressão e temperaturas)
Fonte: www.google.com.br Organizado por: VITORINO, Marcelo.

O equipamento *datalogger*, apresenta uma importância significativa as pesquisas devidas sua utilização ser extensa, e abranger aplicações em diversas áreas, seu custo vem diminuindo com o tempo, tornando–se uma importante ferramenta acessível, com o avanço da microeletrônica sua capacidade e velocidade de armazenamento também tem aumentado.

Atualmente percebe-se que há uma necessidade de criação de materiais, dispositivos e mecanismos acessíveis a todo perfil de usuário, apesar do mercado já apresentar tecnologias baratas, de baixo custo e acessível, as pesquisas nessas áreas são sempre relevantes, pois é, preciso manter e fomentar ainda mais a criação destes mecanismos e deixá-los sempre atualizados para que os usuários possam ter em mãos um objeto eficaz.

Assim se apresenta como principal objetivo deste trabalho o desenvolvimento de um projeto de datalogger com baixo custo, visando principalmente o acesso dos pequenos agricultores. Desta forma o instrumento a ser criado na presente proposta apresenta como características técnicas um circuito integrado de Micontroladores PIC, chip Eeprom, sensores e entradas entre outros aparatos que darão subsídios para o equipamento *datalogger* entrar em funcionamento.

2.4 LINGUAGEM C DE PROGRAMAÇÃO

Em um mundo cada vez mais tecnológico, foi preciso criar uma forma de comunicação entre máquina e homem, desta forma se inventou comandos que se tratam de códigos utilizados na linguagem tecnológica.

Faz necessário compreender essa linguagem de programação, estes tipos de “vocabulários” e “gramáticas” que controlam as máquina, e então, conhecer todas as combinações necessárias de linguagem, para que os equipamentos a instrução necessária para realizar as tarefas que lhe são informadas.

Tais linguagens de programação podem ser definidas em três tipos: Linguagem de máquina; Linguagem simbólica; Linguagem de alto nível. Ainda pode se utilizar a linguagem de programação C, que é de uso geral, considerada de nível

médio por combinar linguagem de alto e baixo nível e possui algumas atribuições tais como:

> Na linguagem C, todo programa é basicamente um conjunto de funções, não havendo distinção entre procedimentos e funções como em PASCAL e FORTRAN. Uma função tem um nome e argumentos associados, e é composta por declarações de variáveis e blocos de comandos, incluindo possivelmente chamadas a outras funções. Um bloco de comandos composto por mais de um comando deve ser delimitado por um par de chaves.
> Em um programa C, se um nome não foi previamente declarado, ocorre em uma expressão e é seguido por um parêntese esquerdo, ele é declarado pelo contexto como sendo o nome de uma função. A função de nome especial main (principal) contém os pontos de início e término da execução de um programa em C. Neste caso, todo programa em C deve ter pelo menos a função main. (ZUBEN, 2015 p.1)

2.5 SENSORES

Sensores tratam-se de dispositivos eletrônicos que tem a propriedade de transformar um sinal elétrico em um estimulo positivo, geralmente os sensores são um conjunto de dispositivos capazes de transformar uma energia em outra, também denominado transdutor, além de ter também uma parte que converte a energia originária em um sinal elétrico. De forma simples pode-se dizer que o sensores servem para passar informações sobre fatores externos onde um circuito eletrônico deva atuar.

Também podemos elucidar sobre o tema discorrido que:

> Termo empregado para designar dispositivos sensíveis a alguma forma de energia do ambiente que pode ser luminosa, térmica, cinética, relacionando informações sobre uma grandeza física que precisa ser mensurada (medida), como: temperatura, pressão, velocidade, corrente, aceleração, posição etc.
> Um sensor nem sempre tem as características elétricas necessárias para ser utilizado em um sistema de controle. Normalmente é sinal de saída deve ser manipulado antes de sua leitura no sistema de controle. Isso geralmente é realizado com um circuito de interface de produção de um sinal que possa ser lido pelo controlador. (WENDLING, 2010, p.4)

Existem dois grupos de sensores são eles: os sensores digitais e analógicos, onde o primeiro se comunica por pulsos e o segundo por variação de corrente ou tensão.

De forma geral para se utilizar um sensor, algumas características relevantes devem ser consideradas: Tipos de saída (digital ou binária, analógica), linearidade, alcance e velocidade de respostas.

2.6 USB

O chamado *USB* sigla para *Universal Serial Bus* - Barramento Serial Universal é denominado portas de entrada e saída de informação, estes tornam a instalação do drive fácil definindo-os como um aparelho simples e fácil de ser utilizado.

Tal instrumento pode ser conectado em dispositivos periféricos como pen-drivers, teclados, impressoras, MP4, entre outros, os dispositivos periféricos tratam-se de aparelhos instalados ao computador que servem como auxílio na comunicação homem - máquina.

Segundo Aguiar (2007, p. 1) “O padrão USB foi desenvolvido por um consórcio de empresas, entre as quais se destacam: Microsoft, Apple, Compaq, Hewlett-Packard, NEC, Intel, Agere e Philips.”.

Desta forma o mesmo autor informa que o consenso entre elas se tornou difícil e portanto dividiram-se em grupos distintos como:

> •UHCI (Universal Host Controller Interface) apoiado majoritariame nte pela Intel, que transferia parte do processamento do protocol o para o software (driver), simplificando o controlador eletrônico;
> •OHCI (Open Host Controller Interface) apoiado pela Compaq, Micros oft e National Semiconductor, que transferia a maior parte do esf orço para o controlador eletrônico, simplificando o controlador lógico (driver). (AGUIAR,2007, p. 1)

Ainda ressalta Aguiar (2007) que o dispositivo USB apresentou ao longo do tempo diversos problemas como a conexão com um aparelho costumava ser problemática, necessitavam de uma velocidade de conexão via internet razoável

que fosse capaz de satisfazer a necessidade encontrada de conexão, as portas seriais são muito lentas, porém todos estes aspectos vêm sendo trabalhado para uma melhora na utilização do aparelho USB.

São utilizados para conexão dois tipos de entrada a porta “A” representada na imagem 1ª e porta “B” representada na imagem 2B, além disso, vale ressaltar que a maioria dos aparelhos apresentam apenas 2 entradas de conexão, e para solucionar tal deficiência pode ser utilizado o Hub USB representado na imagem 2C.

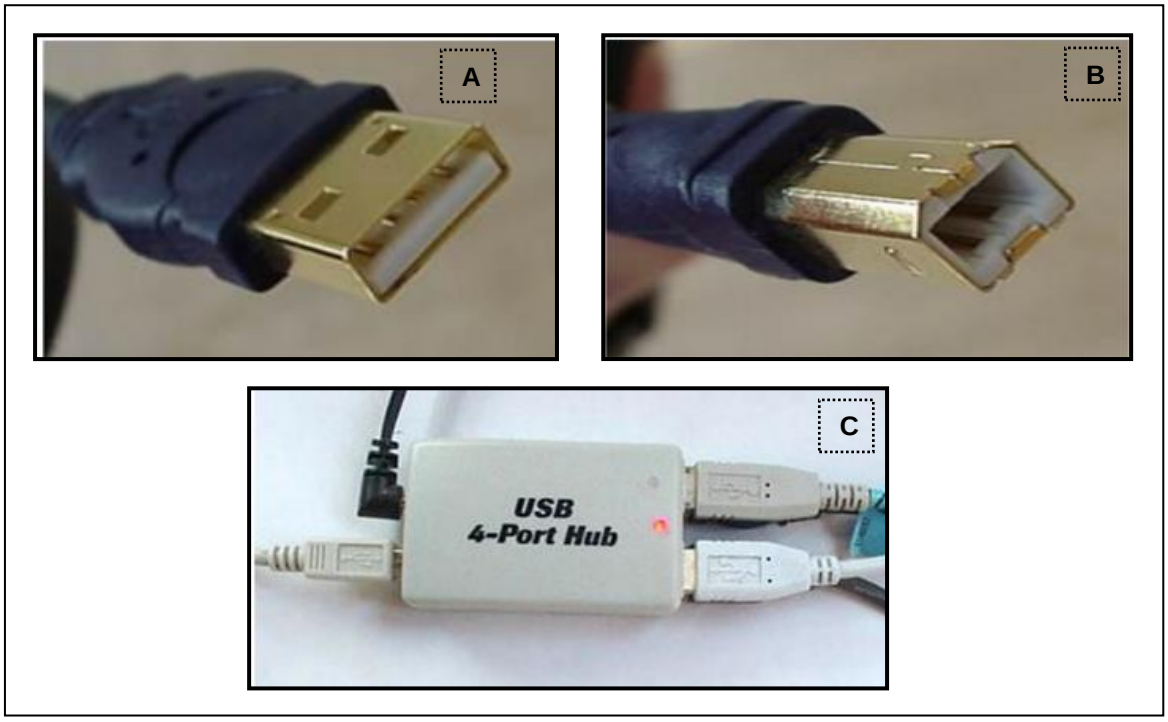


Figura 2– Portas A, B e Hub USB Fonte: Gustavo Aguiar, 2007.
Organizado por: VITORINO, Marcelo.

São diversificadas as forma de utilização do dispositivo USB, ele permite criação de pequenas redes através da conexão por cabos, permite a conexão pelo cabo Hub USB de até 127 dispositivos e é, portanto uma tecnologia que com o tempo só tende a se inovar e aperfeiçoar, buscando cada vez mais sanar as necessidades dos indivíduos, tornando as conexões e a conversa entre homem/máquina mais acessível e simples.

2.7 IMPLEMENTAÇÃO USB VIA CDC

Na montagem do circuito temos que se atentar a alguns pontos que são de grande importância para funcionamento correto da comunicação fora a ligações já vista, para esse tipo de utilização deve ligar o D+ e D- do conector USB (J1) nos pinos 24 e 23 respectivamente do microcontrolador (U1), representado na figura 7.

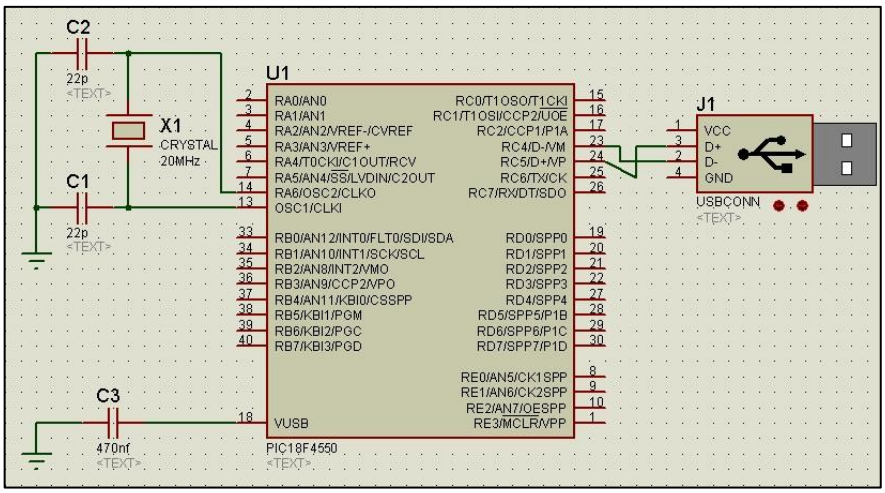


Figura 3 – ligação USB
Fonte: Autoria própria.

Segue abaixo a tabela 1 que demonstra os componentes utilizados para realização da ligação de USB evidenciada.

Tabelas 1 – componentes para ligações USB

Identificação	Componente
C1 e C2	Capacitor cerâmico 22 Pf
C3	Capacitor poliéster 470 nF
U1	PIC 18F4550
X1	Cristal de 20 MHz
J1	Conector *USB*

Fonte: Autoria própria.

O cristal oscilador que esta sendo utilizado (X1) é de 20MHz que através circuito periférico *Phase Locked Loop* (PLL), (que se trata de um circuito multiplicador de frequências de um sinal do *clock)* pode se chegar a 48MHz para o modo *Full Speed Device.*

Para isso ocorrer deve-se ativar a função dos registradores PLL, assim deve declarar nas configurações de *fuses* HSPLL. Também se faz necessário informar por qual valor o sinal do *clock* deve ser dividido no registrador pré-escalar, que nesse caso é cinco declarando PLL5, assim se obtém a frequência de 96MHz e por ultimo é declarado o comando USBDIV assim define que a frequência do sinal de *clock* que será utilizado pelo sistema USB é o proveniente do gerador de 96 MHZ dividido por dois.

Para que possa dispensar a utilização da alimentação externa no pino VUSB, tornando necessário apenas a ligação de um capacitor (C3) de 470nF a este pino e ativação do bit VREGEN, este capacitor deve ser preferencialmente do tipo multidisco cerâmico, para garantir maior estabilidade à comunicação, mas no projeto esta sendo utilizado um cerâmico normal e esta atendendo a expectativa. Na figura 8 os fuses utilizados.

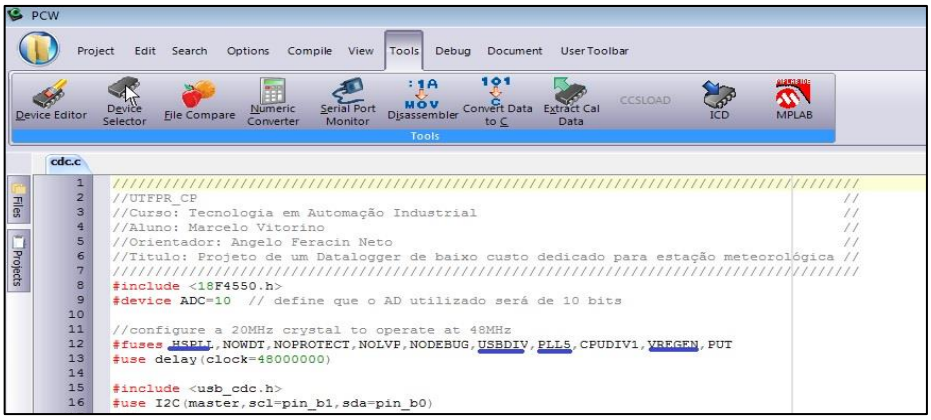


Figura 4 – fuses para o funcionamento USB
Fonte: Autoria própria.

Para programação O CCS (*Custer Computer Services Inc)* possui uma biblioteca destinada a este tipo de comunicação denominada de <USB_cdc.h>.as principais funções :

- USB_init() - Inicializa o hardware USB
- USB_cdc_init() – Inicializa o protocolo CDC.
- USB_cdc_putc() – o microcontrolador envia caracteres ASCII emulados via USB;
- USB_cdc_getc() – retém um caractere ASCII emulado pela USB;
- USB_enumerated() – verifica se o host USB enumerou o dispositivo;
- USB_task() – utilizada para verificar o estado lógico do pino

A biblioteca deve ser copiada para mesma pasta do projeto onde vai ser salvo o arquivo Hex e é encontrada na pasta *Drivers* do compilador (C:\Program Files\PICC\Drivers) a pasta pode ser visualizada na figura 5.

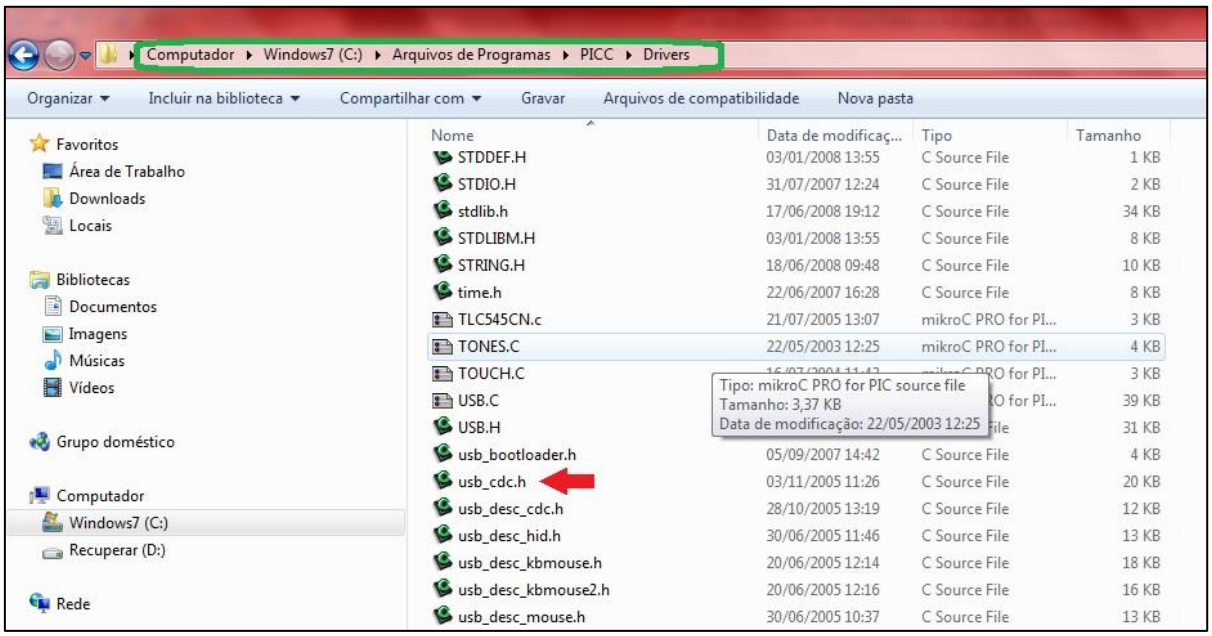


Figura 5 – Pasta dos *drivers*
Fonte: Autoria própria.

Após gravação de um programa que utilize comunicação serial CDC no microcontrolador deve ser instalado o driver CDC da seguinte forma: Primeiramente com o circuito apropriado montado deve-se ligar o microcontrolador e conectar o cabo USB no computador, ir ao menu iniciar do Windows clicando na opção dispositivos e impressora na indicação “não especificado” desta forma será visualizado uma figura com um triângulo amarelo e um ponto de exclamação, nesta figura clique com o botão direito do mouse e selecione propriedades, depois conforme o indicado na figura 6 clique na aba *hardware* e aperte o botão propriedades, na aba geral o botão “alterar configuração” e por fim na aba *driver* o botão “atualizar *driver*” com demonstrado na figura 7.

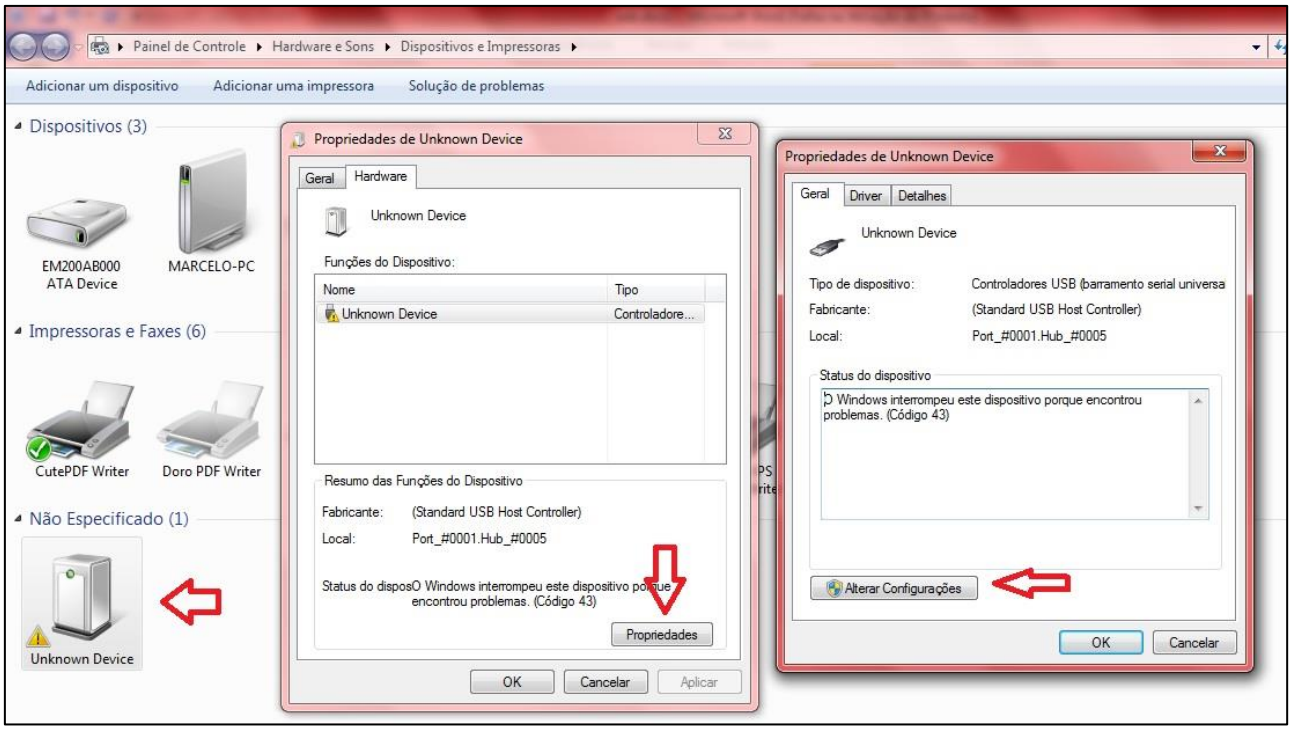


Figura 6 – Instalação de *driver* USB (a)
Fonte: Autoria própria.

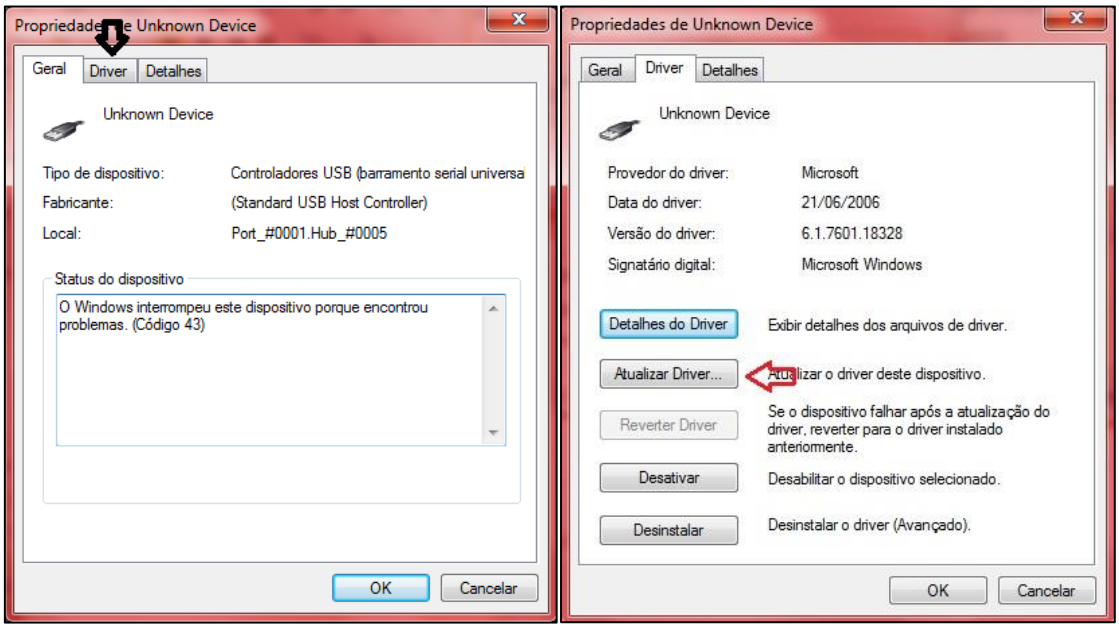


Figura 7 – Instalação de *driver* USB (b)
Fonte: Autoria própria.

Aparecerá uma tela com duas opções “pesquisar automaticamente *software* de *driver* atualizado” e “Procurar *software* de *driver* no computador” como na figura 8 escolha a segunda opção depois clique no botão “Procurar”, e desta forma deve-se apontar o local em que esta o *driver*, nesse caso ele se encontra na pasta *drivers* do compilador.

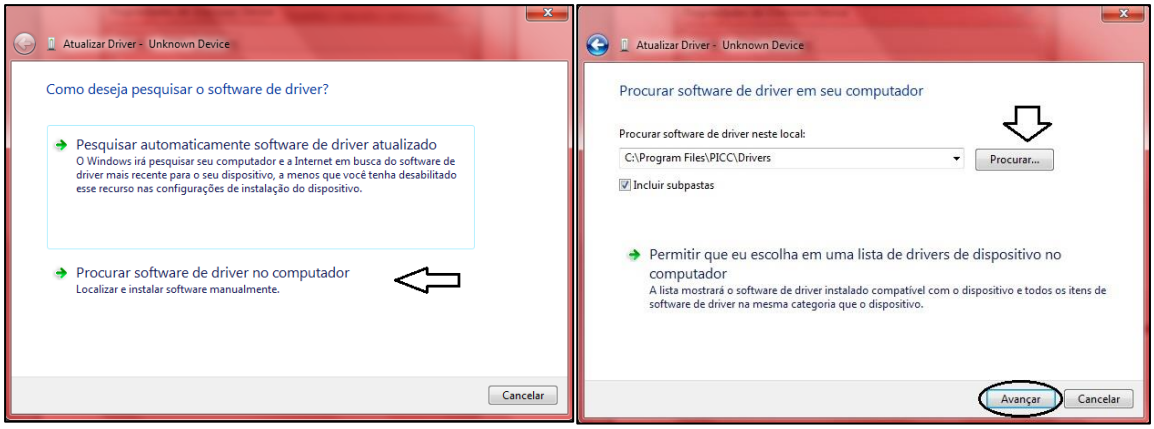


Figura 8 – Instalação de *driver* USB (c)
Fonte: Autoria própria.

No Windows 7, foi utilizado o *driver* cdc_NTXPVista.inf e o resultado foi satisfatório. O *driver* CDC instalado no PC e o programa aplicativo gravado no PIC, com a biblioteca CDC (#include <USB_cdc.h>), são os responsáveis por esta emulação da porta RS-232 virtual através da USB. Abaixo a figura 9 mostra a pasta e o arquivo

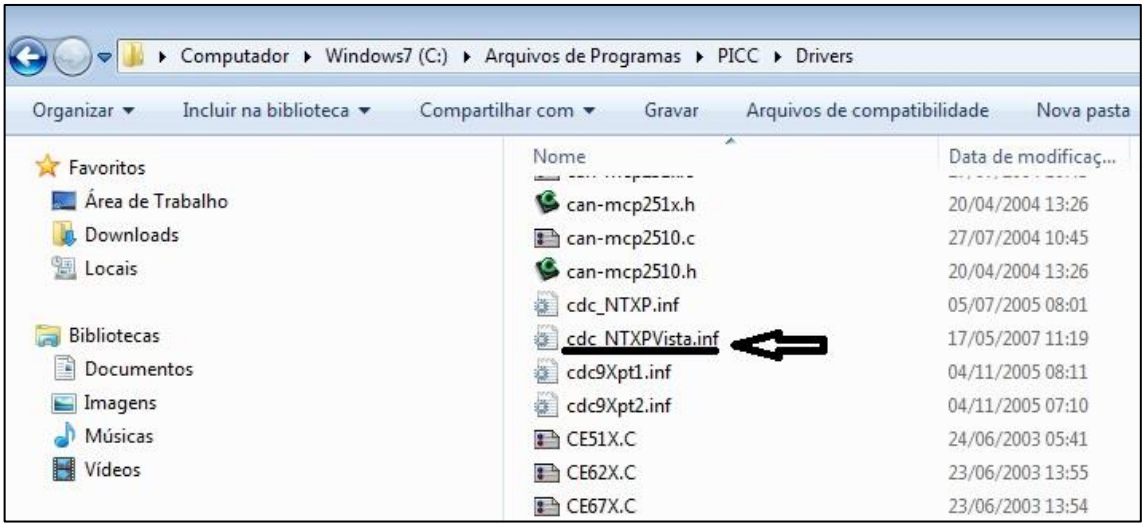


Figura 9 – Instalação de *driver* USB (d)
Fonte: Autoria própria.

Depois de instalado o *driver* no computador o ícone aparecerá com o titulo de “USB to UART” e logo abaixo entre parentes a porta que esta simulando, no exemplo da figura abaixo pode ser observado a porta COM 6.

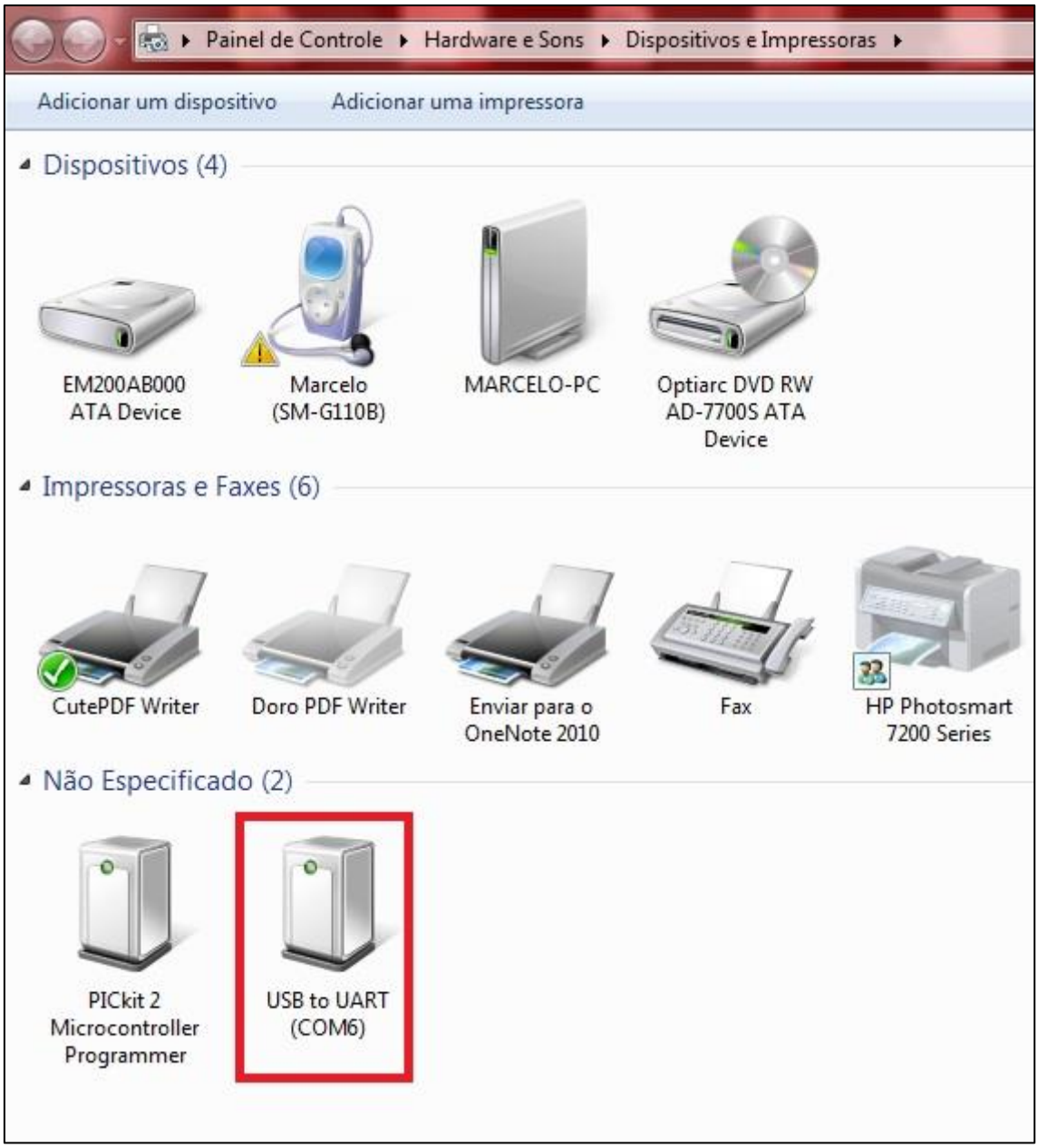


Figura 10 – Porta virtual COM 6
Fonte: Autoria própria.

2.8 *PROTOBORD*

Protoboard, chamado como matriz de contatos é utilizada para diversos fins no meio eletrônico como montagens provisórias, realização de teste de projetos, por exemplo, e inúmeras outras aplicações, equipamento de ágil, pratico e de baixo custo, com diversas marcas e modelos disponíveis no mercado.

O *protoboard* trata-se instrumento de placa que permite a construção

de circuitos experimentais, de base plástica o aparelho contem inúmeros orifícios para à inserção de terminais de componentes eletrônicos. Em sua parte interna existem ligações que permitem interconectar os orifícios e assim sendo possível a montagem de circuitos eletrônicos sem a utilização de solda, a imagem 2 A, 2B e 2C pode ser visto modelos de *protoboard* encontrados no mercado.

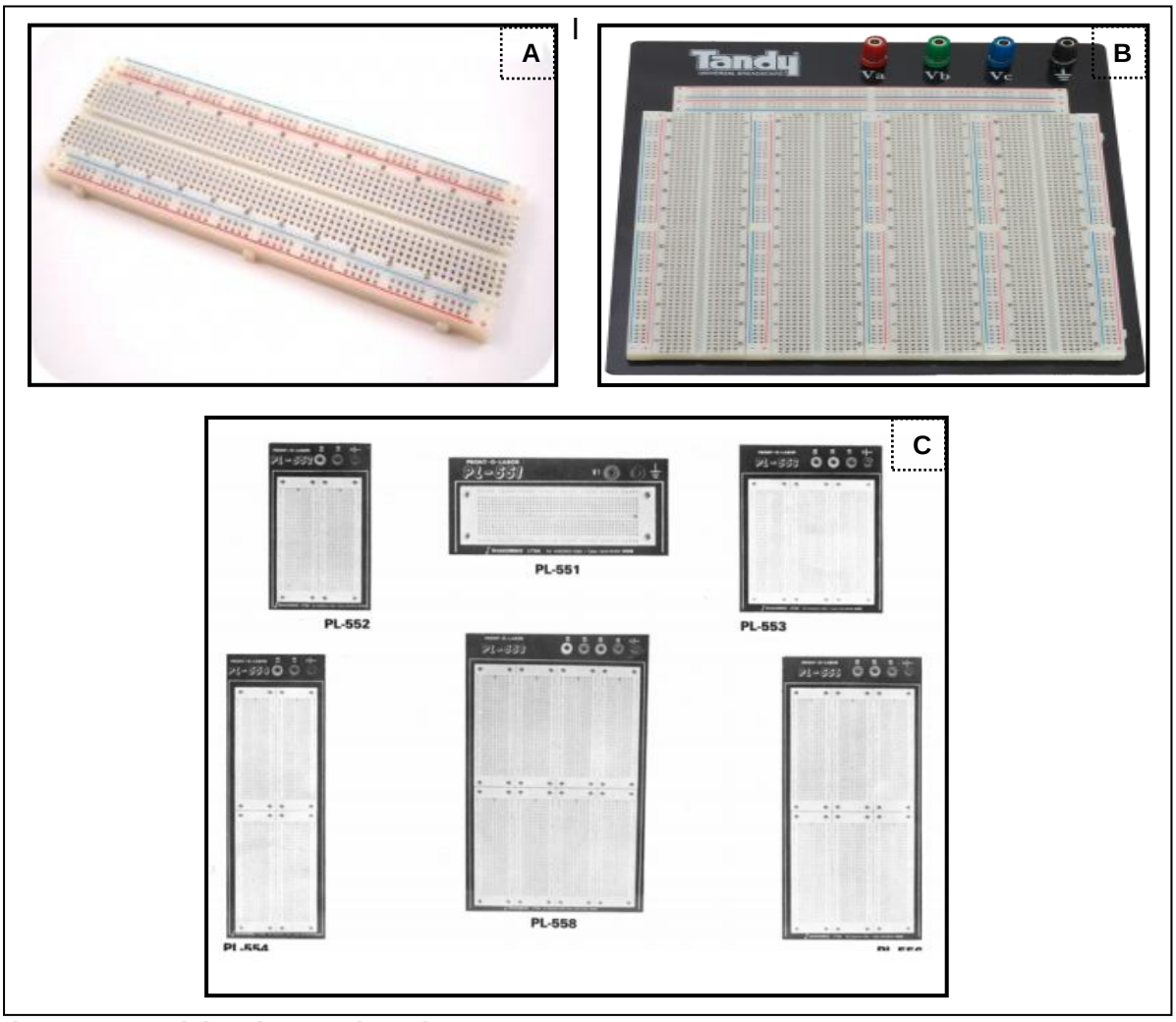


Figura 11– Modelos de *Protoboard*
Fonte: www.google.com.br Organizado por: VITORINO, Marcelo.

A utilização dos *protoboards* ou matriz de contatos apresentam algumas vantagens segundo o LASA - Laboratório avançado de sistemas de Automação (2015, p. 3) informa:

> – Não há necessidade de solda para fixação dos componentes;
> - Possibilidade de trabalhar com circuitos integrados;
> - Manutenção de ligações curtas mesmo em montagens complexas, o que permite o trabalho com circuito de médias e até de altas frequências;
> - Possibilidade de realização de montagens bem complexas;

> - Facilidade de alteração, a qualquer momento, do projeto pela simples retirada do componente, por desencaixe, e colocação de outro.

Para uma melhor compreensão se deve entender o funcionamento de um *protoboard*, desta forma iniciar-se- a explicação pelas filas horizontais e verticais dos terminais, a primeira servem como alimentação sendo elas positivas e negativas, e a segunda está disposta de forma a permitir o encaixe de diversos componentes. E desta forma o LASA (2015, p. 5) informa que:

> As interligações entre pontos ou entre componentes são feitas pelo simples encaixe de pedaços de fios rígidos (ou fios usados em linhas telefônicas internas e componentes de cabos são aos mais indicados para trabalhos nestas matrizes).

Vale ressaltar que as matrizes de contatos são instrumentos fáceis de utilizar, requer um pouco de estudo e habilidade, mas com prática é possível administrá-la da forma correta, basta que os conhecimentos necessários em eletrônicas, circuitos entre outros sejam adquiridos previamente.

2.9 MICROCONTROLADOR

Os microcontroladores também conhecidos como MUC são pequenos dispositivos compostos basicamente por CPU (*Central Processing Unit* - Unidade Central de Processamento), memória e periféricos; suas dimensões reduzidas são possíveis pela tecnologia de circuitos integrados (CI) que é capaz de juntar milhões de componentes em uma única pastilha de silício (MIYADAIRA, 2011).

CPU mais conhecido como processador nada mais é que um circuito eletrônico que executa as instruções de processamento aritmético lógico e movimentação de dados definidas por um programa. Popularmente falando CPU é o “cérebro” de um computador ele que toma todas as decisões.

As memórias são responsáveis pelo armazenamento do programa e essa deve ter extensão suficiente para comportar o código desejado. Segue alguns tipos de memórias utilizadas pelo MUC:

ROM (*Read Only Memory* - Memória apenas para leitura) não permite que o conteúdo seja alterado pelo usuário, aceita somente leitura.

EPROM (E*rasable Programmable Read Only Memory* - Memória programável apagável somente de leitura*)* pode ser programada e apagada varias vezes, porem só pode ser apagada por exposição à luz ultravioleta.

OTP (*one time programmable* - Um tempo programável*)* tolera somente uma gravação.

EEPROM (Electrically-Erasable Programmable Read-Only Memory – memória eletricamente apagável de leitura programável) pode ter seu conteúdo apagado eletricamente e o mesmo pode ser alterado durante a execução do programa.

Flash é um tipo de memória EEPROM essa a mais flexível, pois pode ser apagada eletricamente e reprogramada de 100.000 a 1.000.000 de vezes dependendo da tecnologia empregada (MIYADAIRA, 2011).

A memória de dados do sistema é a RAM (*Random Access Memory* - Memória de Acesso Aleatório) que é utilizada para guardar todas as variáveis e registradores utilizados pelo programa, geralmente é dividida em mais de um banco de dados, para possibilitar o acesso aos endereços com auxílio de chaves que controlam o banco que esta sendo utilizado no momento, e ao contrário da memória de programa ela é volátil, ou seja, é apagada com corte de alimentação (SOUZA, 2002).

Em um computador os periféricos entendem-se como dispositivos de entrada e saída de dados, para melhor compreensão pode ser exemplificado como dispositivos de saída, monitores e impressoras e como dispositivos de entrada, mouse e teclado. No MUC não é diferente além de contar com um processador e bancos de memória ele ainda possui alguns periféricos integrado e isso que o faz um componente especial. Nele podem ser encontrados alguns periféricos importantes para projetos eletrônicos como portas de entrada e saída, PWM (*Pulse-Width Modulation* - Modulação por largura de pulso), conversores A/D (converte sinal analógico para digital) além de portas de comunicação tais como a USART (*Universal Synchronous Receiver Transmiter* - universal síncrono receptor transmissor), SPI (*Serial Perepheral Interface* - Interface periférica de série), I^2C (*Inter-itegrated Circit* - Circuito Inter integrado) e atualmente em alguns muc's mais robusto é encontrado a porta USB (U*niversal Serial Bus* - Barramento serial universal) (MIYADAIRA, 2011).

Segundo Myadaiara (2011, p. 22) “a velocidade de processamento do microcontrolador esta diretamente relacionada com a frequência de *clock*, quanto maior a frequência de trabalho maior será a capacidade de processamento”.

No MUC a divisão de *clock* se concretiza em quatro formas: As fases Q1, Q2, Q3 e Q4, o programa **Conter (PC)** é incrementado automaticamente na fase Q1 e a instrução seguinte é buscada da memória do programa e armazenada no registrador de instrução do ciclo Q4, ela é decodificada e executa no próximo ciclo no intervalo de Q1 até Q4, como pode ser visualizada no diagrama de pulso na figura 12 (SOUZA, 2002).

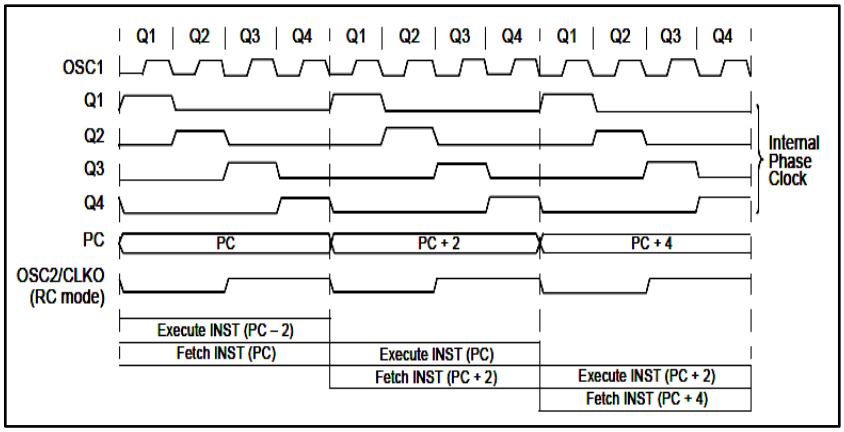


Figura 12 – Diagrama de pulso
Fonte: microchip (2007 p.61).

Por isso o *clock* interno (CK_{int}) é equivalente com o *clock* externo (CK_{ext}) dividido por quatro. Assim será obtida a seguinte fórmula:

$$CK_{int} = \frac{CK_{ext}}{4} \quad (1)$$

Com esse valor pode-se obter o resultado do período dessa frequência que é equivalente a um ciclo de máquina ou tempo de máquina (T_{maq}), utilizando essa fórmula:

$$T_{maq} = \frac{1}{CK_{int}} \quad (2)$$

2.10 COMUNICAÇÃO IC2

A comunicação ic2 será utilizada para comunicação entre o RTC (*real time clock*) e a memória eeprom com o microcontrolador. No I2C a transmissão da informação entre os dispositivos é feita através de 2 fios (Serial Data SDA e Serial Clock SCL).

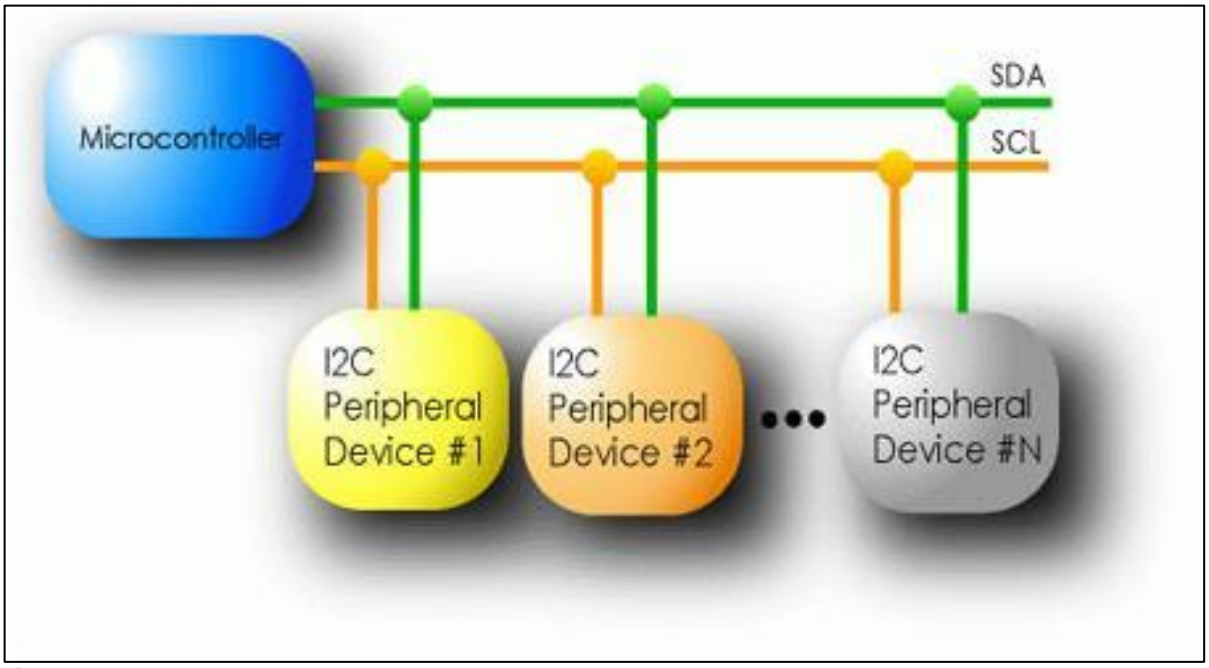


Figura 13 – Barramento I2C
Fonte: MELO (2012).

Os dispositivos ligados em Inter IC possuem um endereço fixo (cada componente recebe um endereço específico), e podemos configurá-los para receber ou transmitir dados.

Dentro da operação I2C, temos os procedimentos *start* e *stop*. Uma transição de nível alto para baixo na linha SDA, enquanto a linha SCL está no nível alto, é o indicativo da situação de *START*. Já uma transição do nível baixo para o nível alto da linha SDA enquanto a linha SCL se mantém no nível alto, define uma condição *STOP*. Sempre o mestre é o responsável pela geração dessas condições.

Após uma condição de *START* o barramento é considerado ocupado, e apenas volta a ficar livre algum tempo depois da condição de *STOP*, todo esse processo é controlada pela biblioteca i2c que deve ser salva na pasta raiz onde esta sendo salvo o programa.

Na programação do pic devemos declarar os pinos utilizados como scl e sda com a expressão:

#use I2C(master,scl=pin_b1,sda=pin_b0).

Nesse caso os pinos utilizados são os indicados no *datasheet* do Pic 18F4550 b1 para scl e b0 para sda, depois deve ser declarada a biblioteca i2c com o comando:

#include <i2c.c>

É importante lembrar-se de abrir o arquivo da biblioteca salvo da pasta raiz, modificar os pinos de scl e sda para os mesmo declarados no comando acima e salvar o arquivo.

3 MATERIAIS E MÉTODOS

O projeto tem como principal objetivo a criação de um *datalogger,* com a finalidade de armazenamento de dados, visando primeiramente como já proposto o baixo custo, isto é, criar um dispositivo acessível a pequenos produtores. O principal componente do projeto será o microcontrolador, sendo escolhido nesse caso o pic 18f4550 por já obter uma interface USB.

O Microcontrolador será programado em linguagem C com a utilização do *software* compilador PCWH da CSS, pela sua grande diversidade de funções e bibliotecas da linguagem C (padrão ANSI), tais como: entrada/saída serial, manipulações de *strings* e caracteres, funções matemáticas C, etc. (PEREIRA, 2002).

Para gravar o programa de linguagem c no microcontrolador será utilizado o Gravador e depurador MicroICD ZIF é um excelente gravador via USB 2.0. Suporta a família de microcontroladores PIC10, PIC12, PIC16, PIC18 e dsPIC30, além de gravar as EEPROM serial 24LC, 25LC e 93LC Microchip. Permite a depuração do programa via MPLAB. Esse deve ser utilizado em conjunto com *software* PICKit 2 Programmer da Microsoft, abaixo na figura 14 encontrasse a imagem do gravador e *software*.

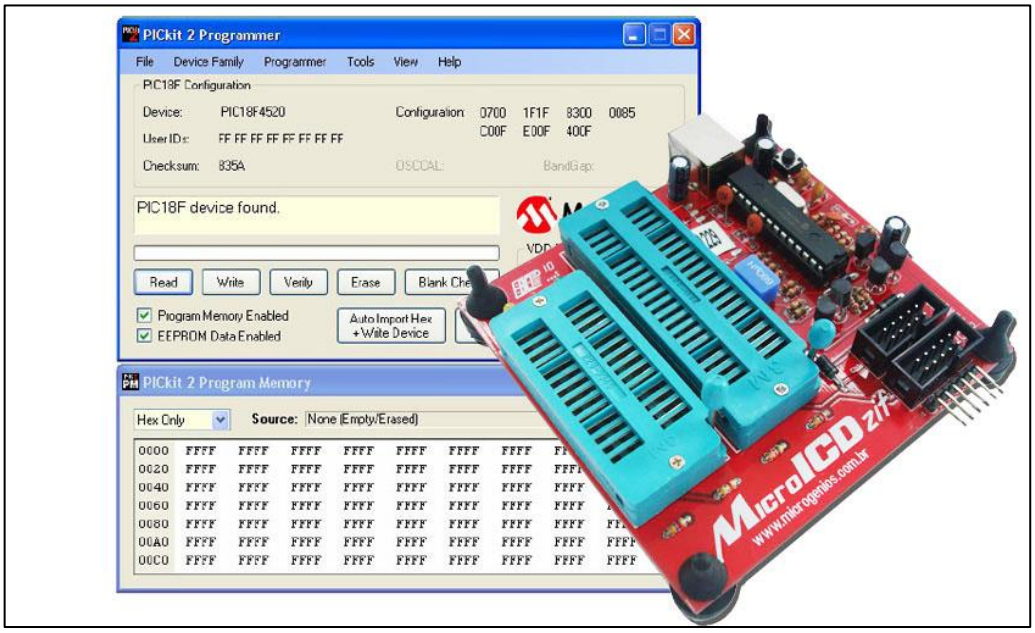


Figura 14 – Gravador MicroICD
Fonte: MICROGENIO SOLUÇÕES ELETRONICA.

Os circuitos foram primeiramente testados através de um *protoboards*, para um melhor aproveitamento na hora de montar o protótipo assegurando que está correto o circuito e cada componente, evitando assim o “monta e desmonta” de placas, como podemos verificar na imagem da figura 15.

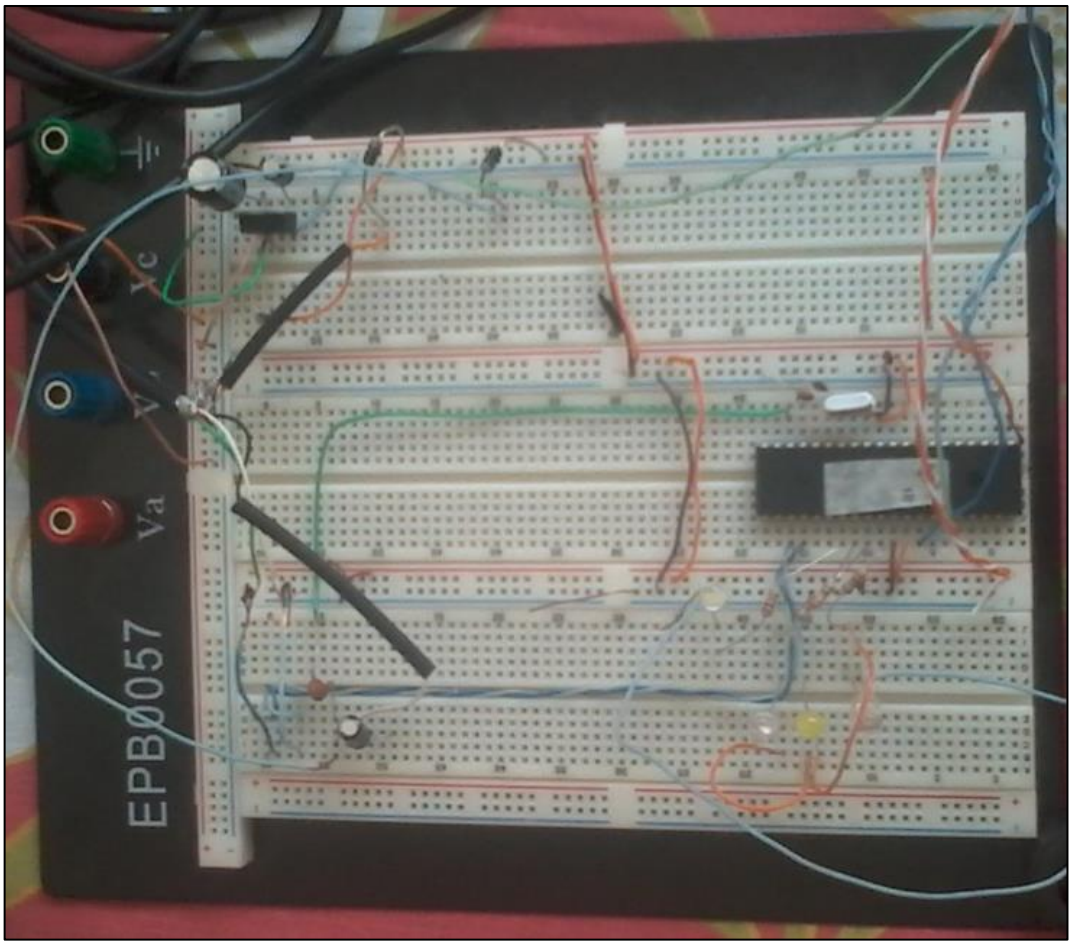


Figura 15 – *Protoboards*
Fonte: Autoria própria.

Os componentes para montagem do projeto segue na tabela 1 com seus devidos valores expressos, para que futuramente possa ser analisada a viabilidade do projeto.

Tabela 1 – lista materias do sistema.

Quantidade	Material	Preço un.	Preço
01	Microcontrolador Pic 18f4550	30,00	30,00
02	Resistor 10K	0,10	0,20
01	Resistor 4K7	0,10	0,10
02	Resistor 91R	0,10	0,20
01	Resistor 150R	0,10	0,10
01	Resistor 75R	0,10	0,10
04	Capacitor cerâmico 22 pF	0,15	0,60
01	Capacitor cerâmico 470 pF	0,15	0,15
01	Capacitor eletrolítico 0.33µF 10V	0,20	0,20
01	Capacitor eletrolítico 0.1µF 10V	0,10	0,10
01	RF 7805	1,50	1,50
01	Conector usb b femea	2,00	2,00
01	Led branco 5mm alto brilho	1,00	1,00
01	Led amarelo 5mm alto brilho	1,00	1,00
01	Led verde / vermelho 5mm alto brilho	1,50	1,50
04	Jack p2 mono	2,50	10,00
04	Plug p2 mono	1,20	4,80
01	Placa fenolite	18,00	18,00
01	Sensor anemômetro	119,50	119,50
01	Sensor indicador de posição de vento	98,50	98,50
01	Sensor pluviômetro	149,50	149,50
01	Soquete zif 40 pinos	9,50	9,50
01	Memoria eeprom 24c512 smd	5,00	5,00
01	Lm 35	7,50	7,50
		Total	R$ 461,05

Fonte: Autoria própria.

3.1 LM 35

O sensor LM35 é um sensor de precisão, que apresenta uma saída de tensão linear relativa à temperatura em que ele se encontrar no momento em que for alimentado por uma tensão de 4-20Vdc e GND, tendo em sua saída um sinal de 10mV para cada Grau *Celsius* de temperatura, sendo assim, apresenta uma boa vantagem com relação aos demais sensores de temperatura calibrados em

“KELVIN”, não necessitando nenhuma subtração de variáveis para que se obtenha uma escala de temperatura em Grau *Celsius*. Abaixo a configuração dos pinos

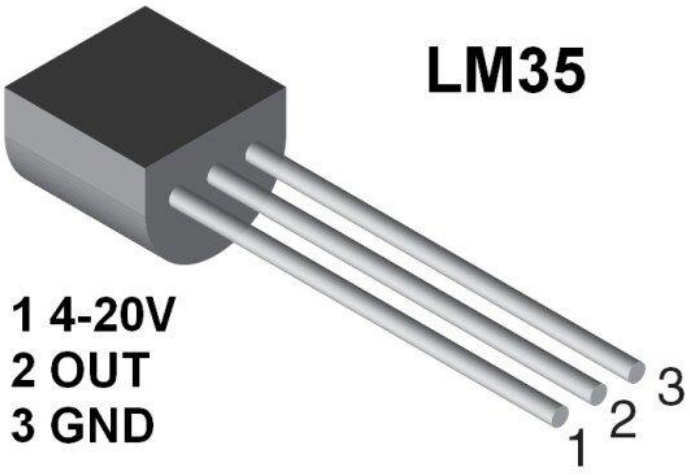


Figura 16 – Configuração dos pinos LM35.
Fonte: www.google.com.br Organizado por: VITORINO, Marcelo.

O LM35 não necessita de qualquer calibração externa ou “*trimming*” para fornecer com exatidão, valores temperatura com variações de 0,25ºC ou até mesmo 0,75ºC dentro da faixa de temperatura de –55ºC à 150ºC. O sensor foi configurado para atuar da faixa positiva entre 2 á 150ºC, a configuração pode ser observada através da figura 17.

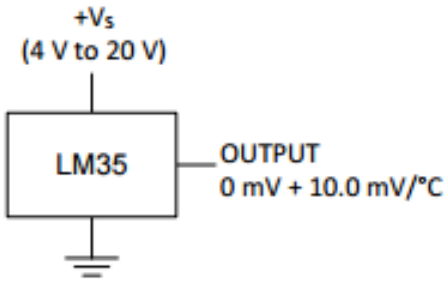


Figura 17 – Configuração para temperatura 0 até 150ºC
Fonte: TEXAS INTRUMETS (2016).

3.2 PIC18F4550

O microcontrolador Pic18F450 é um dispositivo de 8 bits com capacidade de memória de 32Kbites (flash) para programa e 2.048 bytes (RAM)

para dados. Sua tensão de alimentação esta na faixa de 4 a 5,5 Volts (v) e pode operar na frequência de até 48MHz, possui 40 pinos onde 35 podem ser utilizados como portas de entradas ou saídas (I/O - input / output - entrada/saída) e vários periféricos como memória EEPROM de 256 bytes, um módulo de SPI e I^2C, treze conversores A/D de 10 bits, dois comparadores analógico, uma comunicação EUSART, um módulo USB 2.0 capaz de operar no modo *Low-speed* (Baixa velocidade) 1,5Mbps ou *Full-speed* (velocidade total) 12Mbps (MIYADAIRA, 2011).

3.2.1 Oscilador

A CPU do microcontrolador suporta três diferentes fontes osciladoras, as mesmas são selecionadas através dos bit's SCS1:SCS0(OSCCON<1:0>), os quais permitem as seguintes combinação:

- OSCCON<1:0>= 1X : oscilador interno.
- OSCCON<1:0>= 01 : oscilador TIMER 1.
- OSCCON<1:0>= 00 : oscilador primario.

3.2.2 Função do oscilador para USB

Como ja foi visto o módulo USB interno do PIC 18f4550 pode operar no modo *Low-speed* (6MHz) ou *Full-speed* (48MHz) porem a necessidade de satisfazer as frequências impostas pelo módulo.

O modo *Full-speed* é selecionado através do bit FSEN (UCFG<2> = 1) e necessita de um clock de 48MHz que pode ser obtido através de um oscilador primário de 48MHz ou pelo bloco 96MHz PLL dividido por 2, as duas opções podem ser selecionada pela diretiva #PROGMA CONFIG:

- # PROGMA CONFIG USBDIV = 1 : Oscilador primário de 48MHz.
- # PROGMA CONFIG USBDIV = 2 : Bloco PLL 96MHz dividido por 2.

O modo *Low-speed* é selecionado através do bit FSEN (UCFG<2> = 0) e necessita de um *clock* de 6MHz que pode ser obtido pelo oscilador primário de 24MHz dividido por 4 ou pelo bloco 96MHz PLL (MIYADAIRA, 2011).

Abaixo na figura 18 a configurações dos pinos do microontrolador alimentação, entradas, saídas entre outros.

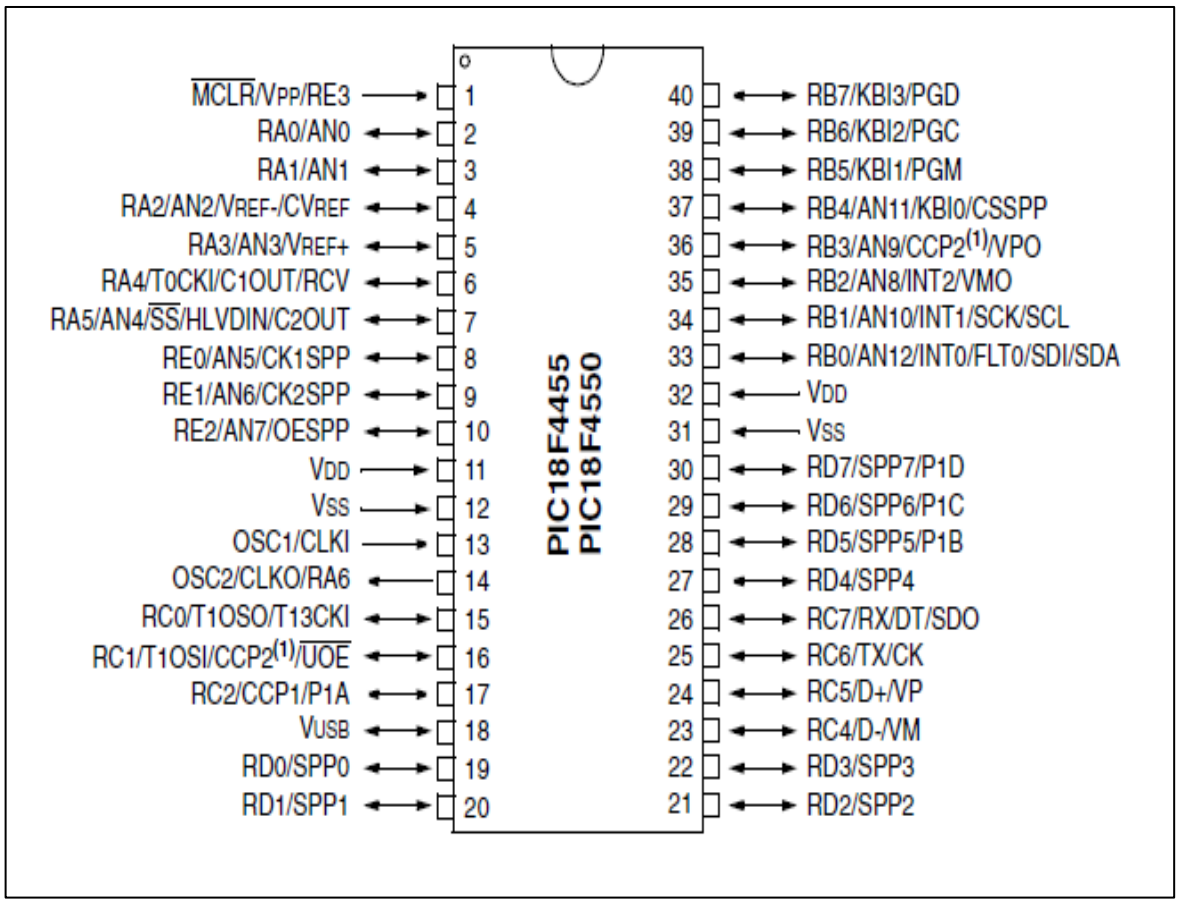


Figura 18 – Descrição dos pinos do Pic 18F4550
Fonte: microchip (2007).

3.3 RELÓGIO DE TEMPO REAL RTC

O *Real Time Clock* I2C DS1307 é um relógio/calendário serial de baixo custo que nesse projeto vai ser controlado por um cristal externo de 32.768 Hz entre os pinos X1 e X2 como demonstra na figura. A comunicação com o DS1307 é através de interface serial I2C (SCL e SDA). O DS1307 possui 7 registros internos destinados para a contagem dos segundos, minutos, horas, dia da semana, dia do

mês, mês e ano. Oito *bytes* de RAM do RTC são usados para função relógio/calendário e são configurados na forma *Binary Coded Decimal* – BCD.

É possível a retenção dos dados na falta de energia utilizando uma bateria de lítio de 3V - 500mA/h conectada ao pino 3, segue a configuração dos pinos na figura 19 e o circuito na figura 20.

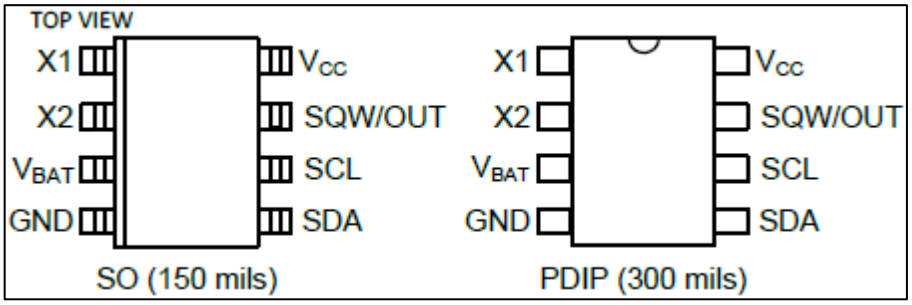


Figura 19 – Pinagem do RTC
Fonte: MAXIM INTEGRATED (2015).

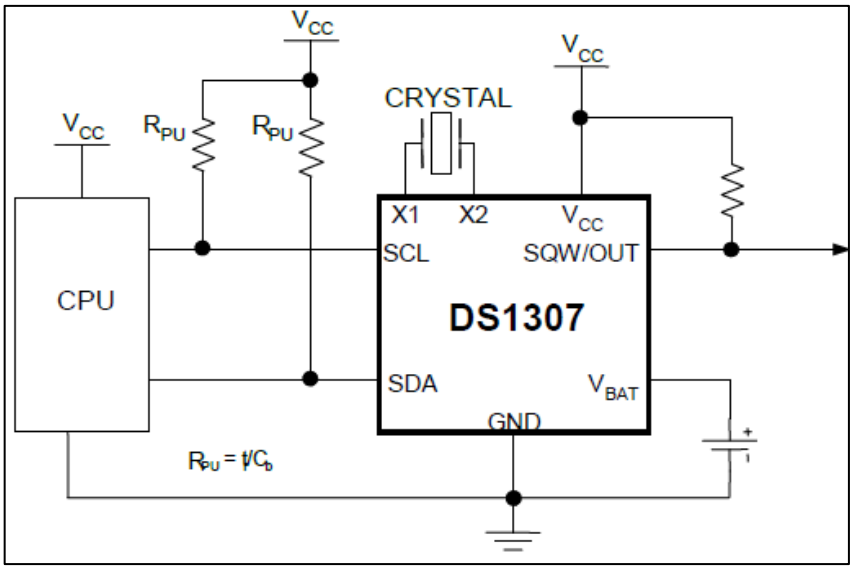


Figura 20 – Configuração do circuito - RTC
Fonte: MAXIM INTEGRATED (2015).

Na programação foi utilizada a biblioteca #include <DS1307.c> encontrado na pasta drivers do compilador, lembrando que ela tem que ser colada na mesma pasta do projeto (arquivo HEX). Na biblioteca tem que mudar os pinos que estão definidos SDA e SCL para ficar igual o definido para a comunicação I²C se necessário, os resistores Rpu da figura do circuito são os utilizados para comunicação.

Os principais comandos são:

- ds1307_init(); / Inicia o módulo.
- ds1307_set_date_time(dia, mês, ano,d/semana,hora,min,sec); / Configura o rtc com o dia, mês, ano, dia da semana, hora, minuto, segundo.
- ds1307_get_date(dia, mês, ano, sem); / Faz leitura do dia, mês, ano e dia da semana atual do relógio e grava nas variáveis indicada no código.
- ds1307_get_time(hora,min,sec); / Faz leitura da hora, minuto, segundo atual do relógio e grava nas variáveis indicada no código.

3.4 MEMÓRIA EEPROM

O chip 24LC512 é uma memória Eeprom compatível com a interface i2c este dispositivo também tem uma capacidade de gravação de até 512K bit (64000X8) sessenta e quatro mil endereços funcionais, permite até oito dispositivos em um mesmo barramento.

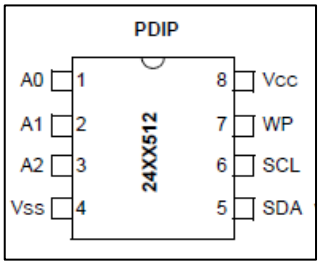


Figura 21– Pinagem da Eeprom
Fonte: MICROCHIP (2010).

No projeto a memória Eeprom externa será utilizada para guarda os dados adquiridos através dos sensores e é ligado o mesmo barramento que o rtc, a figura 22 mostra a configuração do barramento e endereço zero do dispositivo.

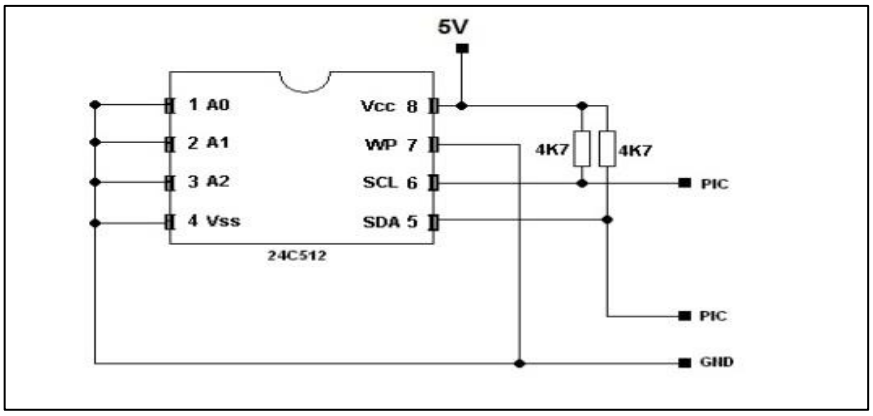


Figura 22 – Configuração do circuito da memória Eeprom
Fonte MICROCHIP (2010).

Os principais comandos são:

- escreve_eeprom(x1, x2, x3) / esse comando escreve um valor até oito bit em um endereço, x1 é endereço do dispositivo de zero a sete, x2 o endereço de memória e x3 o valor a ser gravado.
- le_eeprom(x1, x2) / lê o valor escrito, x1 é endereço do dispositivo e x2 o endereço de memória.

Esses comandos são da biblioteca i2c e a mesma deve ser declarada no começo do programa e o arquivo deve ficar a pasta raiz onde esta sendo salvo o programa do compilador.

3.5 MÓDULO TINY RTC / EEPROM

Para implementar o RTC no projeto foi adquirido um módulo Tiny RTC I2C, que é dotado com uma bateria recarregável que mantém o módulo energizado quando sistema não estiver ligado, para manter os dados, o módulo ainda conta com uma memória eeprom 24c32, seu chip principal e o ds1307, como citado logo acima e o circuito já vem preparado com todos componentes para funcionamento dos chips e para receber a comunicação I2c, o módulo:

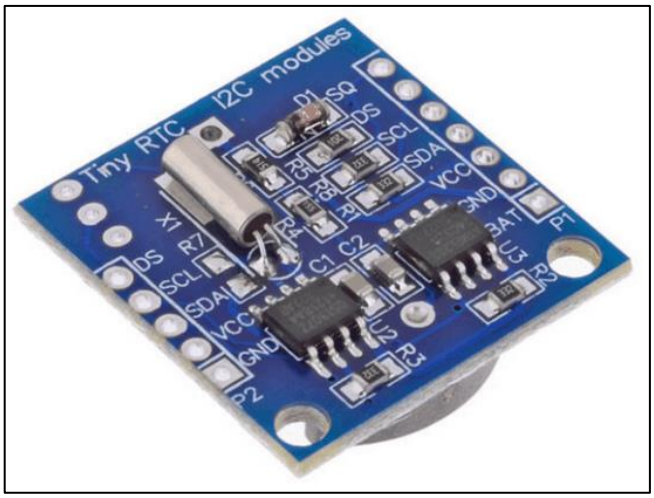


Figura 23 – Módulo Tiny
Fonte: SACCO (2015).

A configuração do circuito do módulo pode ser observada na figura 24.

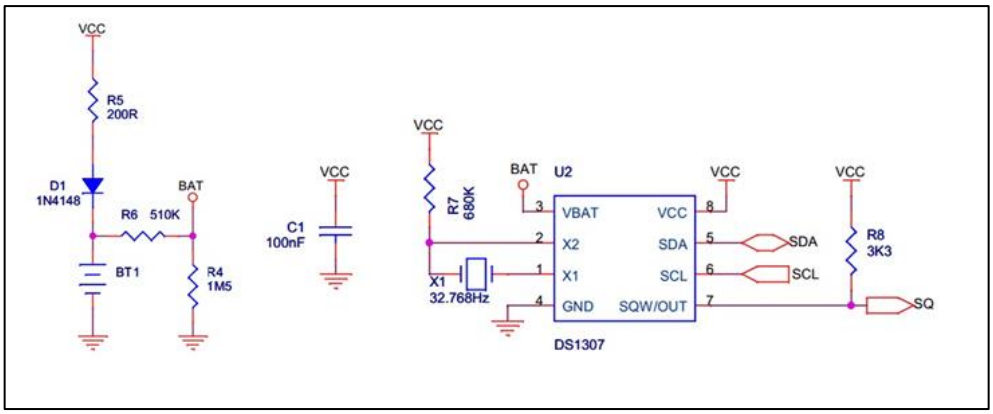


Figura 24 – Circuito RTC do módulo
Fonte: SACCO (2015).

Na eeprom os pinos a0, a1 e a2 estão aterrados o que significa que o endereço dela na rede é zero, conforme a figura 25.

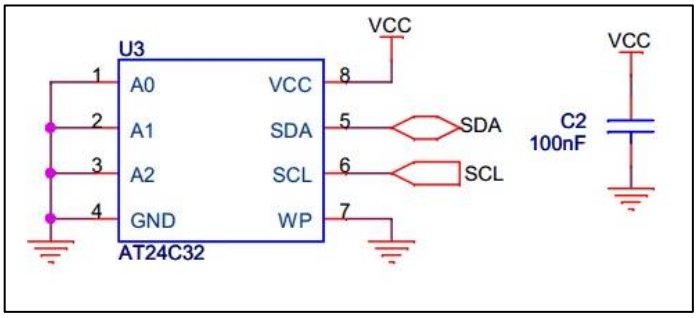


Figura 25 – Circuito eeprom do módulo
Fonte: SACCO (2015).

Na próxima figura esta o esquema dos dois bornes de ligação do módulo, r2 e r3 são os resistores de *pull up* necessário para a comunicação i2c.

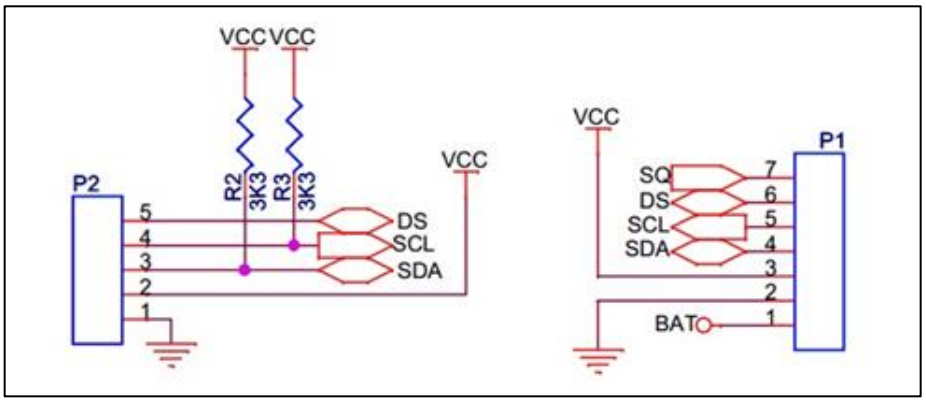


Figura 26 – Bordas de Ligações
Fonte: SACCO (2015).

4 DESENVOLVIMENTO DO PROJETO

4.1 COMUNICAÇÃO USB

A comunicação USB será utilizada nesse projeto para transmissão de dados entre o *datalogger* e usuário onde o mesmo poderá visualizar as medidas através de um microcomputador mais especificamente um *notebook*. Isso ocorrerá através do canal USB do PIC18F2550. Uma das formas mais simples é através do protocolo *Communications Devices Class* (CDC), que emula uma porta COM RS-232 virtual, através do canal USB 2.0. Dessa forma, é possível se comunicar com caracteres ASCII via USB através de qualquer *software* monitor serial RS-232 como o *HyperTermina*l, o SIOW do CCS *Compiler*. O passo a passo para implementação desse método foi descrita no referencial teórico.

Nesse projeto esta sendo utilizado o *Serial Port Monitor do Pic C Compliler* para visualizar os dados adquiridos pelo microcontrolador, porém podem ser utilizados quaisquer programas de monitoramento de porta serial, o importante é que se configure corretamente o mesmo.

O *Serial Port Monitor* é encontrado na aba *Tools* do PIC C, quando aberto o programa inicializa com uma tela de configuração para se comunicar com a Porta serial, primeiro passo é escolher a porta que vai ser monitorada, que no caso do exemplo acima foi a COM6, na próxima figura pode ser observar os outros parâmetros a ser configurado com velocidade de transmissão entre outros.

1. *Baud Rate = 9600bps*
2. *Stop Bits = 1(0)*
3. *Paridade = None(0)*
4. *Data Bits = 8*

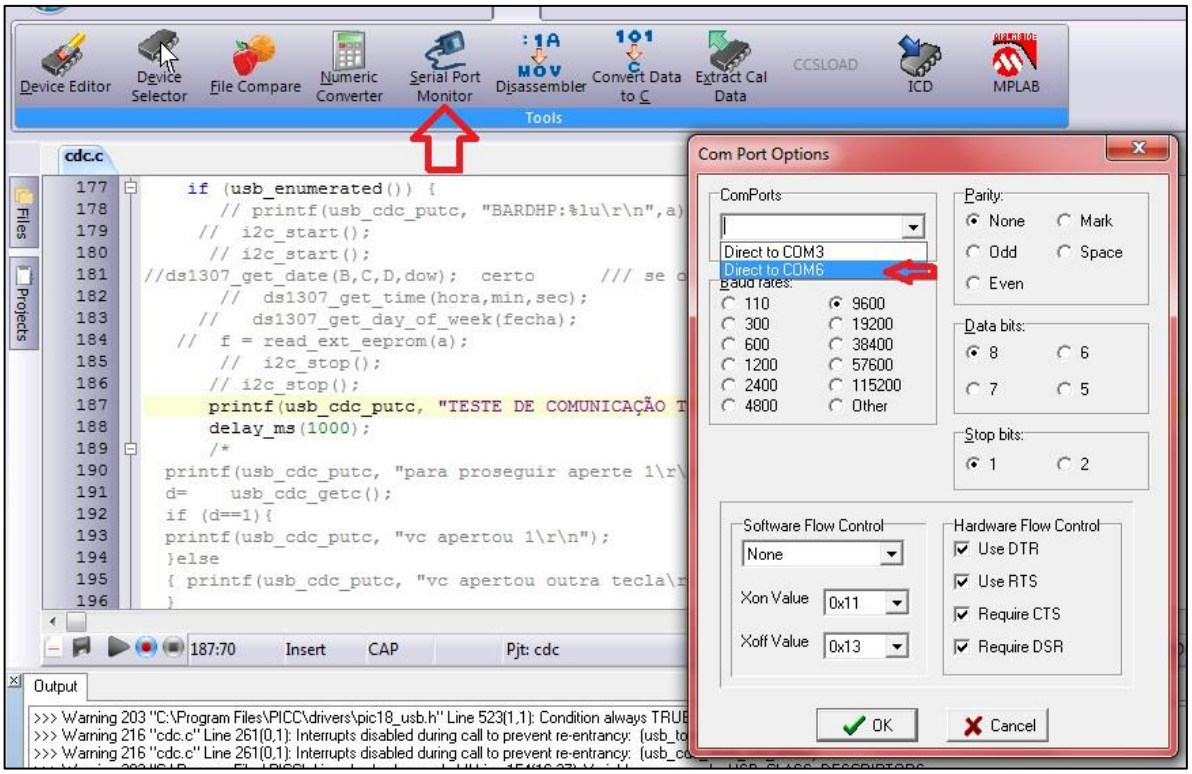


Figura 27 – Configuração do terminal serial
Fonte: Autoria própria.

Para testar a comunicação foi utilizado o comando “printf”:

Printf(USB_cdc_putc, "TESTE DE COMUNICACAO TCC MARCELO VITORINO - AUTOMACAO UTFPR \r\n").

Abaixo pode ser visualizado o resultado do teste:

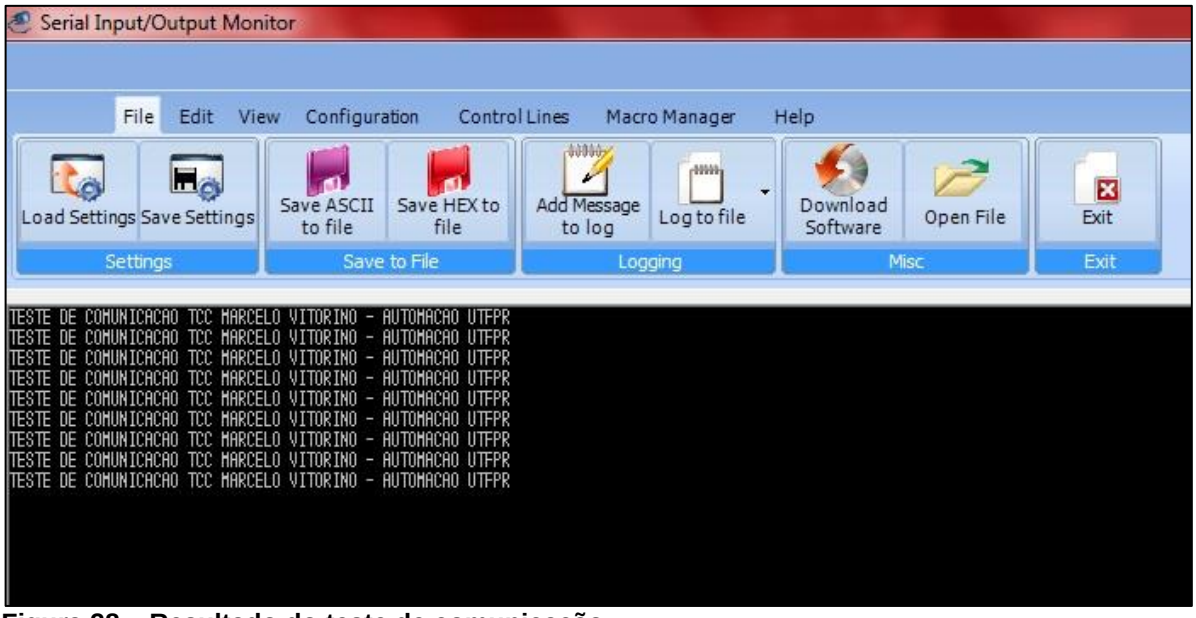
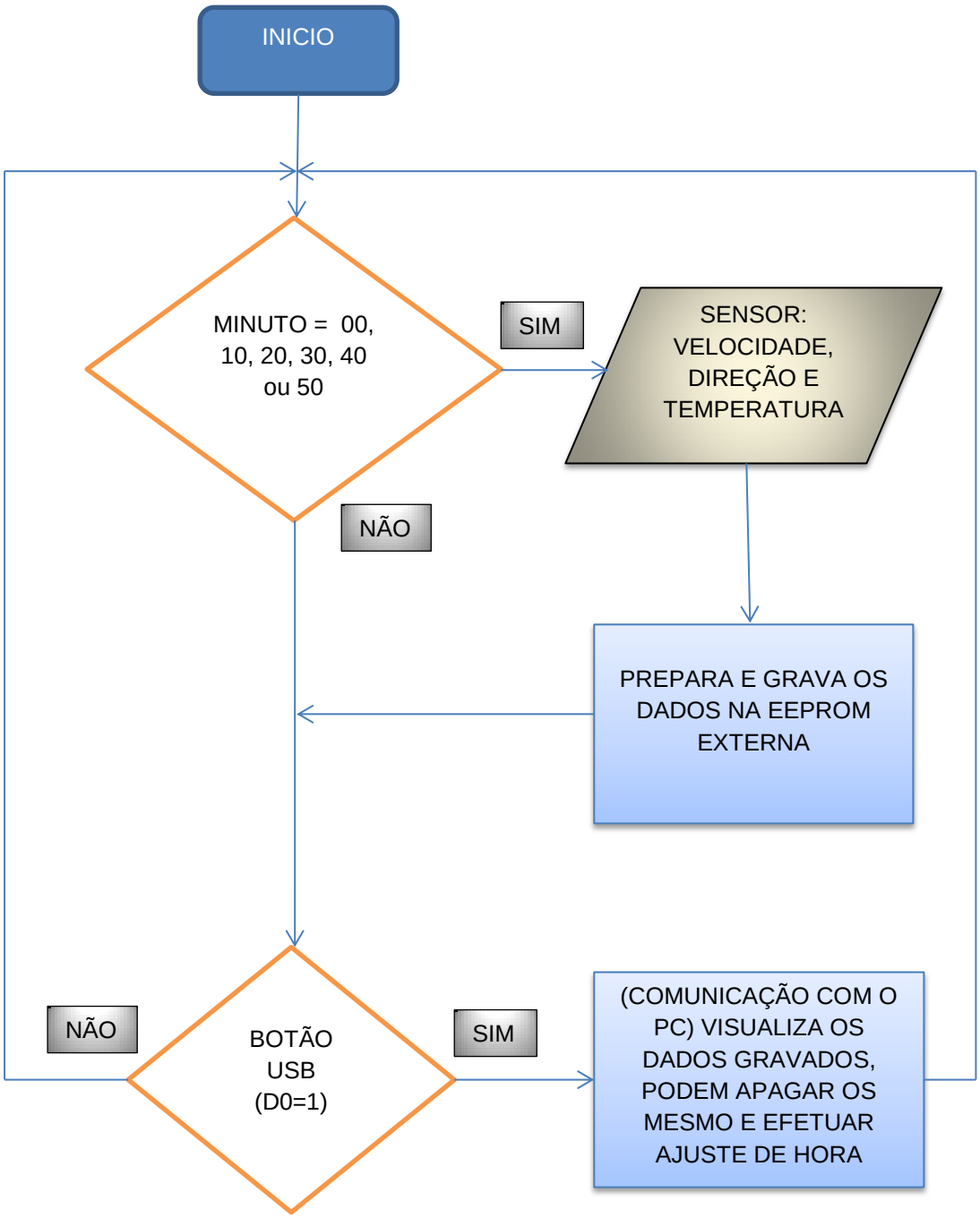


Figura 28 – Resultado do teste de comunicação
Fonte: Autoria própria.

4.2 SÍNTESE DO PROGRAMA

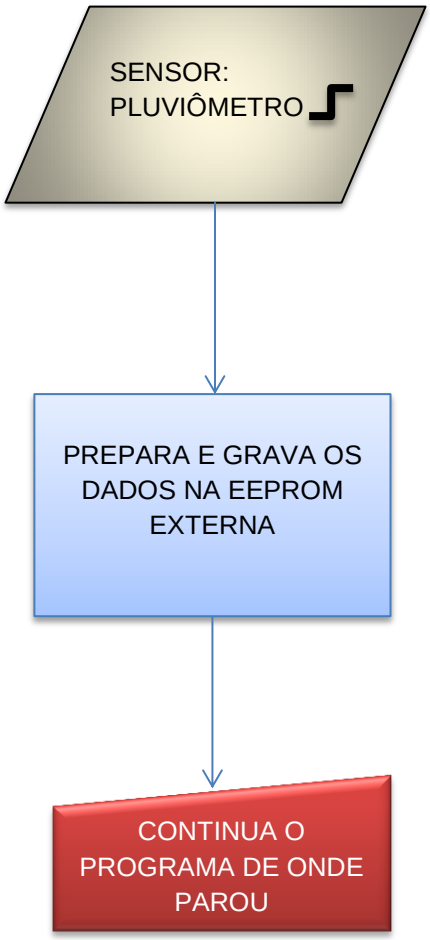
O programa inicia carregando e habilitando todos os recursos a serem utilizados, e acendendo o led amarelo para demostrar que o programa início, já na rotina principal foi definido que a cada dez minutos é efetuada uma varredura dos sensores de temperatura, velocidade e direção do vento. O programa faz a leitura de cada sensor e armazena na eeprom junto com a hora e o dia realizado.

Após sair do laço de varredura ou na sequência após comparação do minuto de varredura, se não verdadeiro, o programa faz a leitura do pino D0, se D0 for igual a zero retorna para comparação de minuto para varredura, se igual a um o programa entra na rotina de comunicação USB, nessa o programa se conecta com o computador através de comandos guiados por menu, pode ser visualizado e apagado o dados dos sensores de temperatura, pluviômetro, anemômetro, e direção também pode se realizar o ajuste de hora do sistema. O programa pode ser melhor compreendido através do fluxograma 1.



Fluxograma 1 – Rotina resumida
Fonte: Autoria própria.

Para se gravar os dados de quantidade de chuva, foi utilizada uma interrupção a int 2, que interrompe o programa onde estiver para realizar a tarefa depois volta de onde parou, essa interrupção é realizada toda vez que há uma borda de subida, no caso do projeto um pulso do pluviômetro que está ligado no pino INT2, desta maneira não se perde nenhum pulso, abaixo o fluxograma2.



Fluxograma 2 – Interrupção
Fonte: Autoria própria.

4.3 FONTE DE ALIMENTAÇÃO

Para alimentação do projeto será utilizado uma bateria nove Volts e para atingir a tesão correta um regulador de tensão o lm 7805, que recebe uma entrada 7 a 25 volts no seu pino de entrada e dispõem cinco volts em sua saída, para estabilidade do sistema capacitores são colocados em paralelo com a entrada e saída, como indica o *datasheet* o circuito pode ser visualizado através da figura 29.

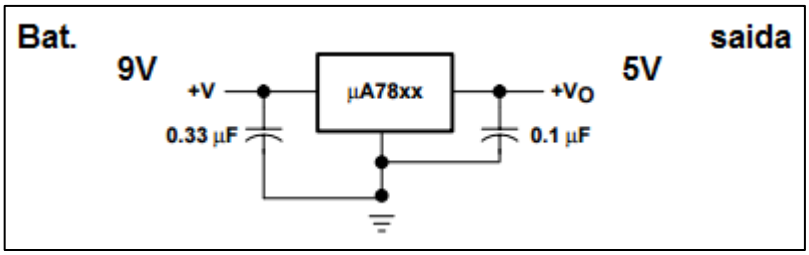


Figura 29 – Resultado do teste de comunicação
Fonte: TEXAS INTRUMETS (2003)

Também foi colocado um capacitor cerâmico de 22 pF entre o positivo e negativo de cada par de pinos da alimentação do pic 18f4550, para filtrar ruídos.

4.4 LED INDICADOR

Alguns led's (*Light Emitting Diode*) foram adicionados no projeto para indicar o estado da programação como, por exemplo, led vermelho acesso indica que está sendo realizada varredura dos sensores de temperatura, velocidade e indicador de vento. No decorrer da descrição do projeto serão citados outros.

Para ligação de cada led, deve ser calculado o resistor de dissipação já que a alimentação nominal dos mesmos é menor do que a de saída do microcontrolador, a fórmula é simples:

$$R = (Vf - Vr) \div I \qquad (3)$$

Fonte que é a saída do Pic de 5 volts menos a tensão de led que é passada pelo fabricante, tudo isso dividido pela corrente que nesse caso é de 20 mA, esse valor também é fornecido pelo fabricante, ficando assim:

Led branco 3,6 V e 20 mA

$$\mathrm{R} = (5 - 3{,}6) \div 0{,}02$$

$$\mathrm{R} = 70$$

Resistor comercial 75 Ω

Led amarelo 2,1 V e 20 mA

$$R = (5 - 2{,}1) \div 0{,}02$$

$$R = 145$$

Resistor comercial 150 Ω

Led vermelho 3,2 V e 20 mA

$$\mathrm{R} = (5 - 3{,}2) \div 0{,}02$$

$$\mathrm{R} = 90$$

Resistor comercial 91 Ω

Led verde 3,2 V e 20 mA

$$\mathrm{R} = (5 - 3{,}2) \div 0{,}02$$

$$\mathrm{R} = 90$$

Resistor comercial 91 Ω

Abaixo na figura 30 mostra como foi efetuada a ligação dos led, vermelho na saída D4, verde em D5, amarelo D6 e branco em D7.

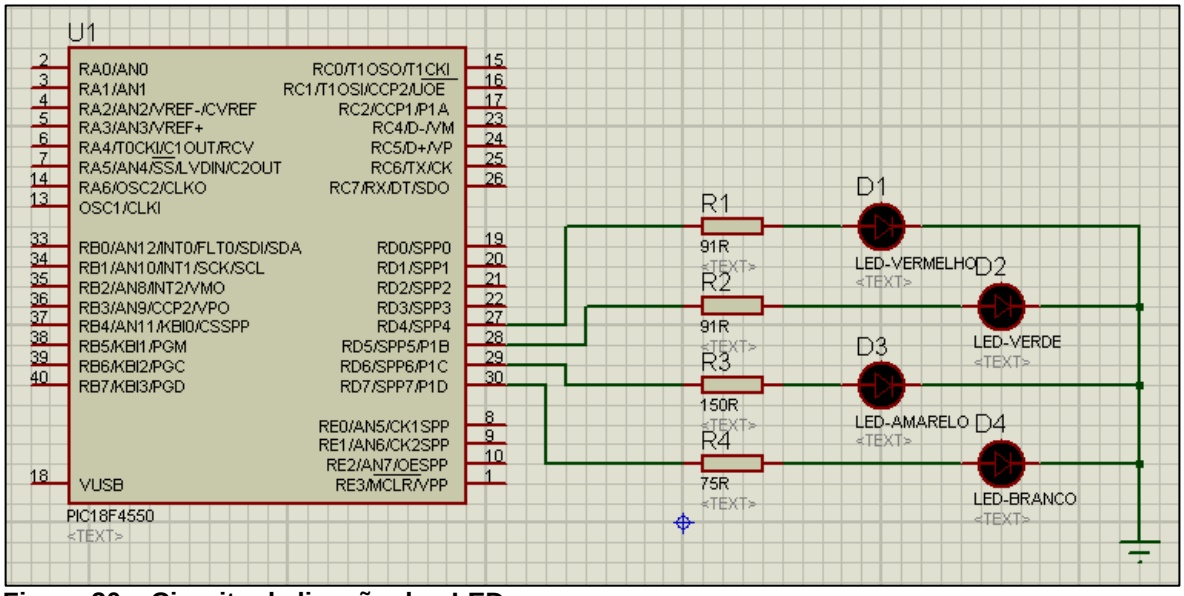


Figura 30 – Circuito de ligação dos LED
Fonte: Autoria própria.

4.5 SENSORES TESTADOS

4.5.1 Indicador de direção de vento

O Indicador de posição do vento serve para saber se o vento vem do sul, norte entre outros. Esse sensor foi adquirido da empresa WRFcomercial através do *site* de compras mercado livre.

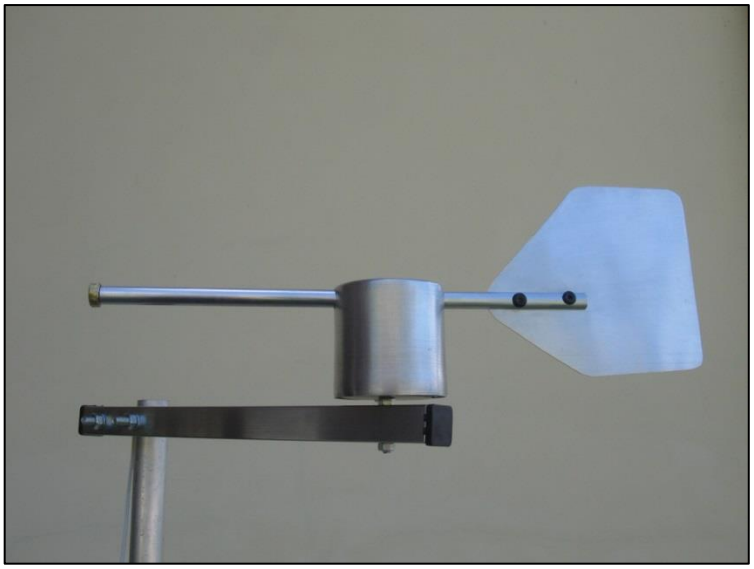


Figura 31 – Indicador de vento
Fonte: Autoria própria.

Características:

- Sensor resistivo lacrado , protegido contra intempéries.
- Giro de 360° com grande sensibilidade.
- Eixo com rolamento lacrado.
- Montado todo em alumínio.
- Eixo indicador com 350 mm.
- Suporte com 250 mm.
- Peso de 350 gr.
- Acompanha cabo com 15 metros.

O princípio de funcionamento desse sensor é uma resistência que varia conforme gira o eixo como um potenciômetro, o circuito para aquisição de dados necessita de uma fonte 5 V, que no caso do projeto é a mesma que alimenta o microcontrolador por motivo de referência, também é necessário um resistor de 4k7 Ω em serie com o sensor como mostra a figura32.

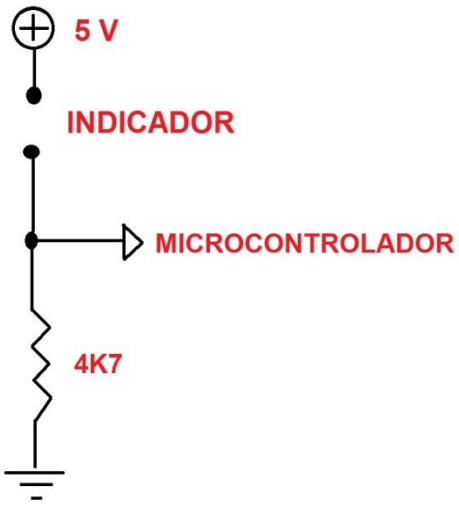


Figura 32 – Circuito para funcionamento do sensor.
Fonte: WRFComercial.

Esse circuito é indicado pelo fabricante para que possa ser utilizada a tabela de referência de coordenadas também fornecida por eles que segue abaixo intitulada tabela 3:

Tabela 3 – direção x tensão.

Tensão em Volts	Direção do vento em pontos cardiais
1,59	NE
0,94	E
0,65	SE
0,50	S
0,42	SO
0,35	O
0,30	NO
0,25	N

Fonte: WRFCcomercial.

Para coletar os dados desse sensor serão ativadas as entradas analógicas do PIC18F4550. Primeiramente no cabeçalho da programação, antes

dos fusíveis deve ser adicionado o comando “#device ADC=10” que define que o AD utilizado será de 10 bits.

Próximo passo é a criação de uma função antes da “void main()”, lê a porta escolhida e retorna o valor quando requisitada, abaixo na figura 33 pode ser visto os comandos e suas funções.

```
43  long int AD(int CANAL) // declara função (subrotina) usada para ler entrada analógica
44  {
45  int32 VALOR; //Declara uma variável de 16 bits para canal de 10 bits
46  set_adc_channel(CANAL); //Congiguração do canal do conversor AD
47  delay_us(100); //Tempo para carregar capacitor canal escolhido
48  VALOR = read_adc(); //Faz a leitura e armazena na variável AUXILIAR
49  return(VALOR); //Retorna valor analógico lido
50  }
51
52  void main() {
```

Figura 33 – Função lê canal
Fonte: Autoria própria

Já dentro da função “void main” deve ser criada uma rotina para chamar a função de aquisição dos dados e também a manipulação dos mesmos. O primeiro passo é habilitar as entradas analógicas que no caso do exemplo abaixo, são as portas NA0 até NA4 (a pinagem encontra-se no datasheet do Pic 18F4550). Pode-se observar que a variável “valorlido0” chama a função AD e entre parentes o número da porta escolhido para leitura; abaixo a variável “valtensao0” multiplica o valor da porta por 5000, que se trata do valor de referência máxima (5V ou 5000 mV), depois divide por 1023 que se trata do valor máximo obtido no canal AD de 10 bits. Essa função é tirada da multiplicação cruzada:

$$\begin{array}{ll} 5000\ mV & \text{-------} \quad 1023 \\ X & \text{-------} \quad valorlido0 \end{array} \qquad (4)$$

O “X” representa o valor de tensão obtido (valtensao0), na figura 34 pode ser visualizado, que a mesma função esta sendo chamadas duas vezes, a diferença esta no canal utilizado mais o processo é o mesmo, para “valorlido0” lê o canal NA0 e “valorlido1” canal NA1, ainda pode ser utilizado da mesma forma no projeto os canais NA2, NA3 e NA4.

```
////////////////////////////////////CONVERSOR  AD/////////////////////////////////////////
//set_tris_a (0b00000000);
setup_adc_ports(AN0_TO_AN4|VSS_VDD); //Habilita entradas analógicas AN0 AN1 AN2 AN3 AN4
setup_adc(ADC_CLOCK_INTERNAL);

delay_us(30);



valorlido0 = AD(0); // Chama AD e lê o retorno do AD canal 0.
valtensao0 = (5000 * (int32)valorlido0) / 1023;



valorlido1 = AD(1); // Chama AD e lê o retorno do AD canal 1 = VREF.
valtensao1 = (5000 * (int32)valorlido1) / 1023;
```

Figura 34 – Inicia função AD
Fonte: Autoria própria.

Na programação foi criada uma função a “vdirecao()”, o objetivo dela é pegar o valor de entrada do canal NA0 já convertido para mili Volts e baseado na tabela de direção x volts fornecida pelo fabricante do sensor incrementar um número para cada direção. A função recebe o número através do comando que a inicia “direcaovento = vdirecao(valtensao0);” esse valor vai sendo testado através das funções “if” até ser verdadeira, como por exemplo, o primeiro “if”, que pode ser visualizado na figura abaixo, nela diz que se o valor for menor que trezentos recebe o valor um que no final do ciclo é incrementada na variável “ddirecaovento” se for maior passa para próxima até se torna verdadeiro. Programação na figura 35.

```
int vdirecao(long int valtensac)
{
int direcao;
 if(valtensao < 300){
direcao=1;
}
  else {
   if(valtensao < 350){
   direcao=2;
    }
     else {
      if(valtensao < 420){
      direcao=3;
      }
       else {
         if(valtensao < 500){
         direcao=4;
         }
         else {
          if(valtensao < 650){
          direcac=5;
          }
           else {
             if(valtensao < 940){
             direcao=6;
             }
             else {
               if(valtensao < 1590){
                direcao=7;
                }
                 else {
                  direcao=8;
                 }}}}}}}
                 return(direcac);
}
```

Figura 35 – Função lê direção
Fonte: Autoria própria.

4.5.2 Anemômetro

O anemômetro serve para medir a Velocidade do vento. Esse sensor foi adquirido da empresa WRFcomercial através do site de compras mercado livre.

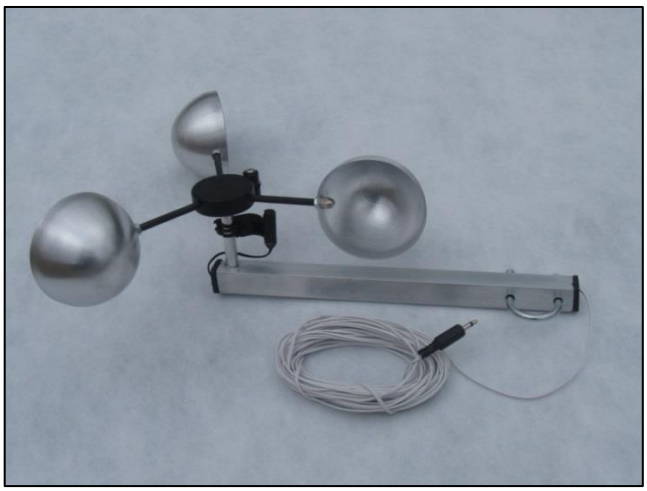


Figura 36 – Anemômetro
Fonte: Autoria própria.

Características:

- “Canecos” em alumínio c/ 78 mm diâmetro.
- Suporte em alumínio c/ 260 mm comprimento.
- Eixo com rolamento lacrado (livre de manutenção), diâmetro total de 145 mm.
- Abraçadeiras p/ fixação.
- Sensor magnético lacrado.
- Cabo com 10 m e plug P2 p/ interligação.
- Resistente a intempéries.
- Suporta altas velocidades (+100 km /h).
- Peso total (c/ cabo e velocímetro) 740 gr.
- Acompanha cabo com 15 metros.

O princípio de funcionamento do sensor é um *chave reed switch* na base, com um imã na parte rotativa que emite um pulso a cada volta completa, como podemos ver na figura 36.

Para obter informação do sensor teremos que encontrar o RPM (rotação por minuto) do mesmo, para isso pode ser utilizada uma entrada digital do microcontrolador. No caso desse projeto o valor de RPM vai ser encontrado com o auxilio de um timer do 18F4550 o timer 1. Segue abaixo a programação.

```
65  ////////////////////////////////////////rpm////////////////////////////////////////////////////////
66
67  setup_timer_1(T1_EXTERNAL|T1_DIV_BY_1); //Prescale de 1 (sem prescale);
68
69  setup_timer_1(t1_external);//Modo Contador Assíncrono;
70  set_timer1(00000);// zera contador
71  delay_ms(3000);
72  rpm=get_timer1(); //recebe o resultado apos 3 seg contando
73  rpm=(rpm*20); //multiplica o resultado por 20 para interar 60 seg ou 1 min (rotação por minuto)
74
```

Figura 37 – Captura RPM
Fonte: Autoria própria.

Dentro da função “main” o *timer* 1 tem que ser configurado como modo contador assíncrono e com *prescale* 1 que no caso é o mesmo que sem *prescale*, desse modo o timer 1 vai ser incrementado a cada borda de subida no pino T1CKI (pino 15), para melhor funcionamento deve ser colocado um resistor de *pull down* que liga o pino ao terra garantido zero volts quando o sensor estiver aberto, eliminando problemas com ruído e flutuação na entrada. A aplicação do resistor na figura 38.

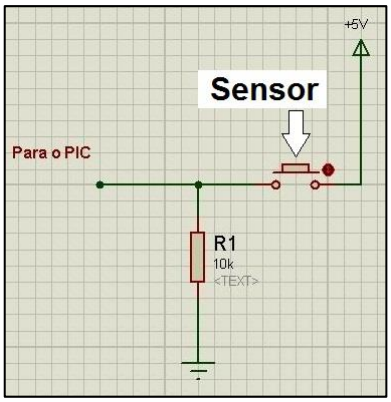


Figura 38 – Resistor de *pull down*
Fonte: Autoria própria.

Com isso a cada volta completa do sensor emitirá um pulso ao pino para contagem, visando desta forma conseguir o valor de rotação por minuto (RPM) o programa zera o timer gera uma atraso de 3 segundos, depois pega a quantidade de pulso desse intervalo multiplica por vinte para totalizar um minuto, como pode ser visto na figura.

Para descobrir o valor da velocidade do vento o fornecedor do produto envia as duas equações.

Calculo da velocidade do vento em M/s:

$$V\frac{M}{s} = \left((4 \times \pi \times \emptyset \times RPM) \div 60\right) \div 1000 \quad (5)$$

Calculo da velocidade do vento em Km/h:

$$V\frac{Km}{h} = \left(\left((4 \times \pi \times \emptyset \times RPM) \div 60\right) \div 1000\right) \times 3{,}6 \quad (6)$$

Onde $\pi = 3{,}14159265$ e $\emptyset = 147$ (raio em mm).

4.5.3 Pluviômetro

O pluviômetro serve para medir a precipitação da água da chuva, ou irrigação artificial em coberta ou ao ar livre. Esse sensor foi adquirido pela empresa WRF Comercial através do site de compras mercado livre.

Figura 39 – Pluviômetro
Fonte: Autoria própria.

Características:

- Montado todo em alumínio, grande durabilidade.

- Sensor com auto esvaziamento.
- Diâmetro coletor 150mm.
- Leitura a cada 0,25 mm.
- Altura de 160 mm.
- Peso de 280 gr.
- Acompanha cabo com 15 metros.

O esse sensor, opera como uma chave que se fecha e se abre a cada um quarto de milímetro de chuva, assim o sistema pode coletar os dados através de uma porta digital que contara os pulsos para saber a quantidade de chuva no dia.

Para aquisição dos dados do sensor pode ser utilizado uma interrupção externa, que no caso do projeto será int_ext2. Essa interrupção é acionada através da porta RB2 pino 35 (pode ser observado no *datasheet* do pic 18F4550) através de um pulso, quando isso ocorrer o microcontrolador realiza as rotinas descritas na função que é chamada através da #int_ext2 como pode ser observado na figura 40.

```
#int_ext2
void ext2 (void){
}
```

Figura 40 – interrupção 2
Fonte: Autoria própria.

Para perfeito funcionamento do sistema, dentro da função “main” deve ser configurada e habilitada a interrupção. A primeira função visualizada na figura 41 configura o acionamento da int2 através borda de subida, quando o nível sai de zero para um, por isso é utilizado um resistor *pull down*, como visto no anemômetro, a segunda habilita interrupção externa dois e a terceira habilita todas as interrupção configuradas.

```
ext_int_edge(2,L_TO_H);
enable_interrupts(INT_EXT2);
enable_interrupts(GLOBAL);
```

Figura 41– configura interrupção
Fonte: Autoria própria.

4.5.4 Temperatura Lm 35

Para implementação desse sensor foi conectado a saída do mesmo com o pino de entrada analógica AN1do Microcontrolador, essa foi programada para fazer a leitura em *mili volts* assim todo vez que efetuado a leitura o valor era dividido por dez desta forma chegando ao valor da temperatura em graus *celsius*. Mais detalhes desse sensor podem ser visualizados no capitulo de matérias e métodos descrito anteriormente.

4.6 VARREDURA DOS SENSORES DE TEMPERATURA, VELOCIDADE E DIREÇÃO DO VENTO.

Foi criado um laço de varredura que primeiramente busca a hora atual e depois compara com os minutos escolhidos para varredura dos sensores de velocidade e direção do vento e de temperatura, se for igual realiza a rotina do laço, e não ele verifica o sinal no pino D0, caso seja nível lógico alto, ele o programa e entra na rotina modo USB.

Quando o programa passa para a varredura dos sensores inicialmente prepara se dia, hora, minuto para ser gravado; minuto é dividido por dez para ser representado por zero a cinco, onde zero representa zero minuto, um é dez e assim sucessivamente, depois esse número multiplica-se por dez mil, em seguida multiplica a hora por cem em seguida, logo após soma o valor obtido com o minuto e a hora com o dia chegando a um número só, por exemplo:

11:20 do dia 15 fica 21115

Para gravar esse número na Eeprom precisa dividí-lo em duas partes já que cada espaço de memória obtém apenas oito *bits* e assim é possível gravar um valor de zero a quinhentos e cinquenta e cinco.

O método utilizado nesse projeto foi dividir o número por quinhentos e cinquenta e cinco e pegar o resulto e o resto, utilizando o exemplo a cima fica dado1=82 e dado2=205.

Programação segue abaixo através da figura 42.

```
while (true){
enable_interrupts(INT_EXT2);
enable_interrupts(GLOBAL);
i2c_start();
ds1307_get_date(dia,mes,anio,dow);        /// data atual
ds1307_get_time(hora,min,sec);      //hora atual
i2c_stop();
delay_ms(40);
if (min==10||min==20||min==30||min==40||min==50||min==00){ //mimutos de coleta dos dados
   output_high(PIN_D4); // led indica que esta dentro do laço de gravação
   ////////////////////Prepera data e hora/////////////////////////////////////
   teste=min;
   gmin=min/10;
   gmin=gmin*10000;
   testehora=hora;
   ghora=testehora*100;
   gdata=gmin+ghora+dia;
   gdata1=gdata/255;
   gdata2=gdata%255;
```

Figura 42 – Laço de leitura de sensor (a)
Fonte: Autoria própria.

O próximo passo do laço e ler os canais analógicos que por enquanto são apenas dois AN0 e AN1, o programa lê a entrada e transforma para milivots, primeiramente AN0 que esta ligada ao sensor de direção de vento que em seguida chama a função “vdirecao” que atribui um valor a cada direção, como vista acima quando relatado o sensor.

Logo após, o AN1 onde esta o sensor de temperatura que em seguida divide o valor por dez para obter a temperatura. Na sequência o programa zera o contador do timer 1 espera três segundos e a variável “rpm0” recebe a quantidade de pulso, no pino ligado ao anemômetro, esse valor será utilizado logo mais para descobrir o valor de rotação do sensor programação na figura 43.

```
//////////////leitura dos canais analogico/////////////////////////////////////
   valorlido0 = AD(0); // Chama AD e lê o retorno do AD canal 0(direção vento).
   valtensao0 = (5000 * (int32)valorlido0) / 1023;
   direcaovento = vdirecao(valtensao0);  // chama a função vdirecao
   valorlido1 = AD(1); // Chama AD e lê o retorno do AD canal 1 (temperatura).
   valtensao1 = (5000 * (int32)valorlido0) / 1023;
   temp= valtensao1/10; //recebe valor da temperatura 1 grau a cada 10 miliVolt.
///////////////RPM/////////////////////////////////////////////////////////////
   set_timer1(00000);// zera cotador
   delay_ms(3000);// espera 3seg.
   rpm0=get_timer1();//recebe qauntidade de pulso no perioldo.
```

Figura 43 – Laço de leitura de sensor (b)
Fonte: Autoria própria.

Para gravar os dados o programa primeiro lê a quantidade de vezes que se passou pelo laço de leituras dos sensores, essa informação fica gravada no

endereço zero da eeprom, em sequência para gravar os dados busca se o próximo endereço, sem dados através da expressão “n=(ng*5)+1;” ela multiplica a quantidade de laços vezes cinco, que é a quantidade de dados gravado por vez, e soma um. Por exemplo, se o programa já passou pelo laço uma vez ele grava no endereço um, dois, três, quatro e cinco, e nas próximas vezes ele faz a conta:

$$n = (1 \times 5) + 1$$
$$n = (5) + 1$$
$$n = 6$$

O próximo endereço utilizado será o seis e em seguida o programa grava o primeiro dado de data/hora incrementa um ao endereço, grava segundo dado de data/hora e continua com direção de vento, quantidade de pulso do anemômetro e temperatura, e por sequência incrementa um ao endereço zero como pode ser visto na figura 44.

```
 ng= le_eeprom (0,0);// le a quantidade de dados gravados.
 delay_ms(40);
 n=(ng*5)+1;// prepara posicção da memoria
 escreve_eeprom(0,n,gdata1);//grava dados de data/ hora.
  delay_ms(40);
 n++;
 escreve_eeprom(0,n,gdata2);//grava dados de data/ hora.
  delay_ms(40);
 n++;
 escreve_eeprom(0,n,direcaovento);//grava dados de direção.
  delay_ms(40);
 n++;
 escreve_eeprom(0,n,rpm0);//grava dados de RPM.
  delay_ms(40);
     n++;
escreve_eeprom(0,n,temp); //grava dados de temperatura.
  delay_ms(40);
 ng=ng+1;
 escreve_eeprom(0,0,ng); // grava quantidade de dados
  delay_ms(40);
```

Figura 44– Laço de leitura de sensor (c)
Fonte: Autoria própria.

Para finalizar o laço o programa lê a hora atual compara o minuto atual com o de quando entrou no laço, que ficou gravado na variável teste se igual, que é muito provável, ele fica retido em um laço *while* que o testa até ficar diferente. Este processo é feito para evitar o programa entrar mais de uma vez no laço no mesmo minuto, quando sair do laço é apagado o *led* vermelho no pino D0 que acesso no começo do laço como mostra a programação na figura 45.

```
    i2c_start();
   ds1307_get_time(hora,min,sec);
    i2c_stop();
    while (min==teste){
     i2c_start();
   ds1307_get_time(hora,min,sec);
    i2c_stop();
    }
      output_low(PIN_D4);

}
```

Figura 45 – Laço de leitura de sensor (d)
Fonte: Autoria própria.

4.7 ARMAZENAMENTO DE QUANTIDADE DE CHUVA

Para para captar os dados de quantidade de chuva como visto acima, quando solicitado o sensor aciona uma interrupção para que não se perca nenhum dado, a função é chamada quando a um nível lógico alto no pino RB2/INT2 onde está ligado o sensor, a interrupção utilizada foi INT2.

Quando ocorre um pulso do sensor, que acontece a cada 0,25 mm de água, inicia a fução “void ext2 (void){”, esta começa como no laço de varredura lendo a hora e data atual e o sistema para gravar os dados é o mesmo, a diferença é que nesse caso como a leitura pode ocorrer a qualquer minuto é utilizado um espaço de memória só para o os minutos.

Outro detalhe é que a quantidade de vez em que a função é acionada fica no espaço de memória três mil e os dados gravados a partir de três mil e um, a quantidade de chuva é baseado a partir das vezes em que a função foi chamada. Logo abaixo, segue a programação na figura 46.

```
#int_ext2
void ext2 (void){
i2c_start();
ds1307_get_date(dia,mes,anio,dow);
ds1307_get_time(hora,min,sec);
i2c_stop();
delay_ms(50);

   gmin=min/10;
   gmin=gmin*10000;
   testehora=hora;
   ghora=testehora*100;
   gdata=gmin+ghora+dia;
   gdata1=gdata/255;
   gdata2=gdata%255;
    ng2= le_eeprom (0,3000);
   delay_ms(40);
   n=(ng*3)+3001;
   escreve_eeprom(0,n,gdata1);
    delay_ms(40);
   n++;
  escreve_eeprom(0,n,gdata2);
    delay_ms(40);
    n++;
  escreve_eeprom(0,n,min);
    delay_ms(40);
    ng2=ng2+1;
    escreve_eeprom(0,3000,ng2);
    delay_ms(40);
```

Figura 46 – Interrupção 2 quantidade de chuva
Fonte: Autoria própria.

4.8 COMUNICAÇÃO USB COM O COMPUTADOR

Para conectar o microcomputador ao datalogger primeiramente deve-se apertar o interruptor de acionamento USB (uma chave que comuta entre positivo e negativo com a entrada D0 em comum) que muda o estado da porta “d0” de zero para um, assim quando o laço de varredura passar pela função “if (input(PIN_d0)){” ele reconhece como verdade e realiza a rotina dessa função.

Quando isso acontece o led do indicador de estado (pino D5) fica verde podendo assim retornar a chave de acionamento USB para posição de origem, assim o USB é inicializado pelas suas funções “USB_cdc_init()” e “USB_init();” com isso o USB estará pronto para receber sinal. Quando conectado o pc, a programação entra no laço “while(!USB_cdc_connected()), pisca o led branco que esta no pino D7 ;{” o “USB_enumerated()” verifica se o host USB enumerou o dispositivo e segue a rotina, a programação pode ser analisada na figura 47.

```
if (input(PIN_d0)){
a=10;
usb_cdc_init();
usb_init();
output_high(PIN_D5);
while(!usb_cdc_connected()) ;
{



do {
output_high(PIN_D7);
delay_ms(2000);
output_low(PIN_D7);
    usb_task();
    if (usb_enumerated()) {
```

Figura 47 – Programação: início de entrada de USB
Fonte: Autoria própria.

O projeto quando aberto o *Serial Port Monitor* é escrito na tela um cabeçalho, toda escrita na tela do computador se realiza através da função printf(USB_cdc_putc,"\r\n"); e esta sendo utilizado um *delay* entre as escritas para evitar que o programa que recebe os dados pule linha, o imprima em uma posição errada. A programação segue na figura 48.

```
delay_ms(200);
printf(usb_cdc_putc, "////////////////////////////////////////////////////////////////////////////////\r\n");
delay_ms(200);
printf(usb_cdc_putc, "//              UNIVERSIDADE TECNOLOGICA FEDERAL DO PARANA - CP             //\r\n");
delay_ms(200);
printf(usb_cdc_putc, "//              TRABALHO DE CONCLUSAO DE CURSO                              //\r\n");
delay_ms(200);
printf(usb_cdc_putc, "//              ALUNO: MARCELO VITORINO                                     //\r\n");
delay_ms(200);
printf(usb_cdc_putc, "//              ORIENTADOR: ANGELO FERACIN NETO                            //\r\n");
delay_ms(200);
printf(usb_cdc_putc, "////////////////////////////////////////////////////////////////////////////////\r\n");
```

Figura 48 – Programação do cabeçalho
Fonte: Autoria própria.

Após o cabeçalho o programa cai em um laço “while (a<40){” que repete a rotina enquanto “a” for menor que quarenta e em seguida mostra a mensagem “PARA ACESSAR MENU DIGITE 1 PARA SAIR DIGITE QUALQUER OUTRO NÚMERO”, a função “c= USB_cdc_getc();” fica aguardando um número ser digitado através do computador, para depois ser analisado pela função “if (c==1){”, se verdade, imprimi o menu e espera ser selecionada a opção, se não, “a”

recebe o valor de cinquenta e sai do laço apagando o led verde e mostrando uma mensagem de até mais e a hora atual do sistema.

Para seleção das opções do menu, primeiro a variável “c1” espera o número escolhido referente a opção desejada através do comando “c1= USB_cdc_getc();” após o usuário digitar o número referente a aba desejada a função.

“switch(c1){” roda a rotina escolhida, a figura 49 mostra a estrutura da função.

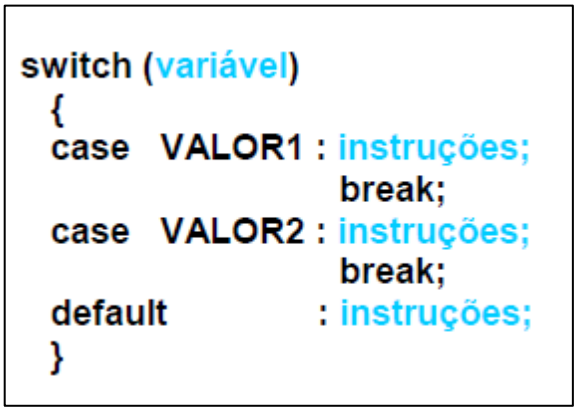


Figura 49 – Sínteses laço case
Fonte: Autoria própria.

No projeto o “case 1:” representa ajuste de hora, “case 2:” velocidade e direção do vento, “case 3:” quantidade de chuva, “Case 4:” temperatura e “case 5” sair. Abaixo estará descrita a rotina de cada um desses casos e a figura 50 demostra como fica o menu no computador.

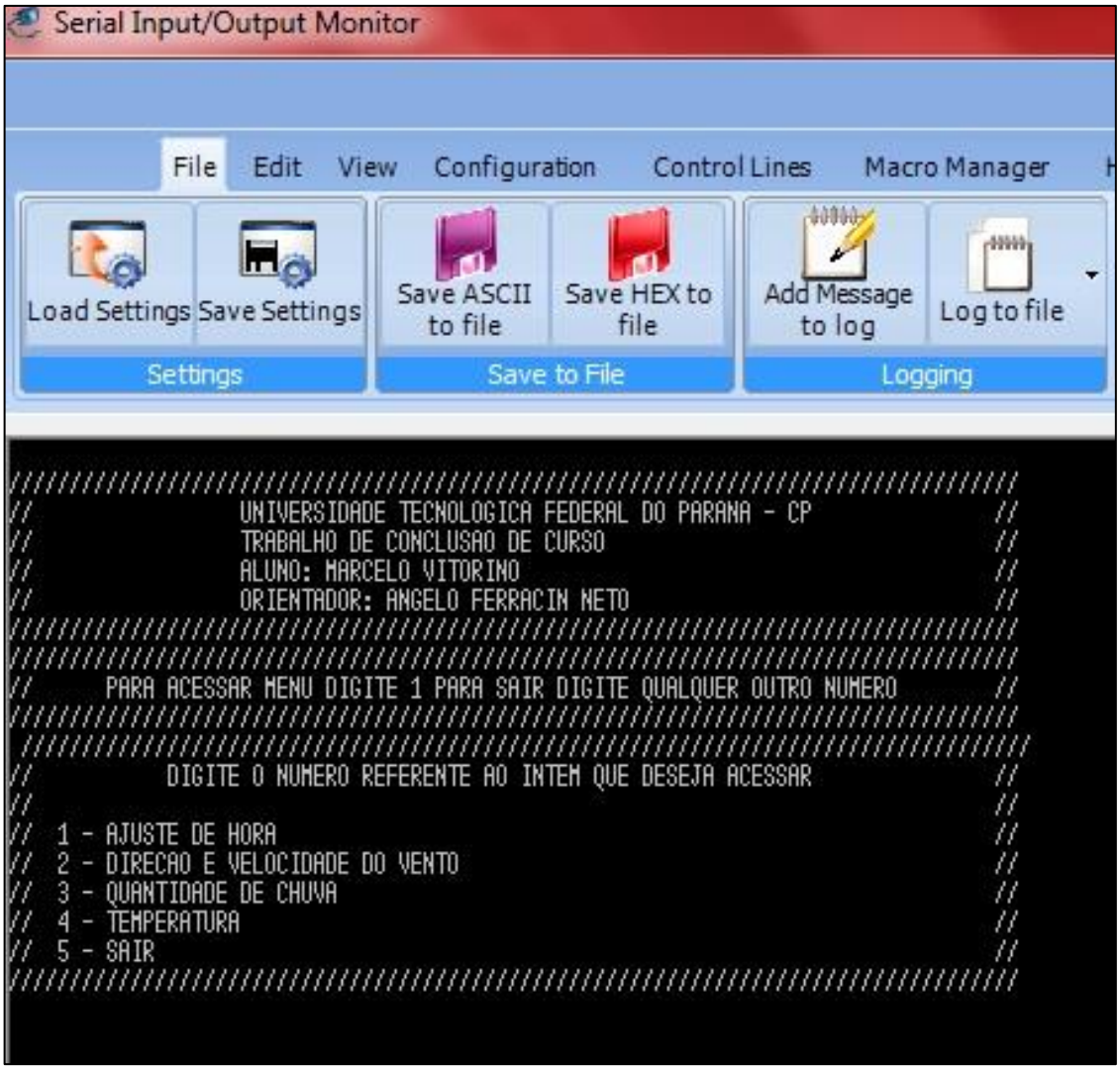


Figura 50 – Tela menu
Fonte: Autoria própria.

4.8.1 Ajuste de hora

Quando acessado pelo menu, o ajuste de hora mostra na tela a hora e data atual e uma mensagem “PARA AJUSTAR RELOGIO DIGITE 1 PARA SAIR OUTRO NÚMERO” o número digitado e lido através da função “r= USB_cdc_getc();” e a ação selecionada pela função “if (r==1){”, se for escolhido um e rodada a função que mostra mensagens pedindo para digitar dia, mês, ano, semana, hora, minuto e segundo e cada número e recebido por sua respectiva variável como pode ser vista na figura 51.

```
printf(usb_cdc_putc, "    PARA AJUSTAR RELOGIO DIGITE 1 PARA SAIR OUTRO NUMERO\r\n");
 r=    usb_cdc_getc();
if (r==1){
delay_ms(200);
printf(usb_cdc_putc, "    DIGITE O DIA\r\n");
 dia = usb_cdc_getc();
delay_ms(200);
printf(usb_cdc_putc, "    DIGITE O MES\r\n");
mes = usb_cdc_getc();
delay_ms(200);
printf(usb_cdc_putc, "    DIGITE O ANO\r\n");
 anio = usb_cdc_getc();
delay_ms(200);
printf(usb_cdc_putc, "    DIGITE A SEMANA 1-DOM 2-SEG 3-TER 4-QUA 5-QUI 6-SEX 7-SAB\r\n");
dow = usb_cdc_getc();
  delay_ms(200);
printf(usb_cdc_putc, "    DIGITE A HORA\r\n");
hora = usb_cdc_getc();
delay_ms(200);
printf(usb_cdc_putc, "    DIGITE O MINUTO\r\n");
 min = usb_cdc_getc();
delay_ms(200);
printf(usb_cdc_putc, "    DIGITE O SEGUNDO\r\n");
sec = usb_cdc_getc();
delay_ms(200);
```

Figura 51 – Programação de ajuste de hora
Fonte: Autoria própria.

Após ter digitado os dados o relógio é atualizado pelo comando “ds1307_set_date_time(dia,mes,anio,dow,hora,min,sec);” e antes de voltar para menu mostra a hora.

4.8.2 Visualização dos dados de direção e velocidade do vento

Quando escolhido a opção dois do menu, o programa passa para o “case 2”, nesse local é impresso os dados de direção e velocidade do vento obtido na varredura dos sensores.

O programa começa lendo a quantidade de laço gravado para saber quantas vezes terá que rodar para imprimir todos dados, em seguida vai para o *loop* que executará essa tarefa. Primeiramente é pego os pacotes de dados de data/hora fazendo a leitura da memória de endereço e incrementando o endereço como pode ser visualizado na figura 52.

```
 delay_ms(200);
   printf(usb_cdc_putc, "////////////////////////////////////////////////
  delay_ms(200);
    printf(usb_cdc_putc, "//  2 - DIRECAO E VELOCIDADE DO VENTO
  delay_ms(200);
    printf(usb_cdc_putc, "////////////////////////////////////////////////
 i2c_start();
      ng1 = le_eeprom (0,0); // lê  quantidade de varredura de sensores
      delay_ms(40);
      i2c_stop();
      n1=1;
      b=0;
  delay_ms(200);
while (b<ng1){   // laço para imprimir dados

 i2c_start();
ldata1=le_eeprom (0,n1);// le dado data/hora 1.
 delay_ms(40);
n1++;
ldata2=le_eeprom (0,n1);// le dado data/hora 2.
i2c_stop();
 delay_ms(40);
n1++;
```

Figura 52 – Programação de impressão de dados(a)
Fonte: Autoria própria.

Em seguida os dados são preparados para leitura, e como foi visto acima, a forma que os dados gravados no programa multiplica o primeiro por duzentos e cinquenta e cinco e soma com o segundo para um único número, utilizando os números do exemplo:

$ldata1 = 82 \times 255 = 20910$

$ldata = 20910 + 205 = 21115$

Para conseguir minutos divide o número por dez mil e depois multiplica por dez, com o exemplo fica:

$m = 21115 \div 10000 = 2{,}1115 \cong 2$

$m = 2 \times 10 = 20$

20 min.

Obs. Utiliza só o número inteiro.

A hora é o resto da divisão por dez mil, dividido por cem e o dia o resto da divisão, exemplo:

$h = 1115 \div 100$

$h = 11{,}15 \cong 11$

11 hora e dia 11

Ficando 11:20 do dia 15, em seguida é impresso a data na tela do computador e realizado a leitura do dado de direção e incrementando o endereço para leitura da velocidade.

Para calcular o rpm é pego o dado de velocidade, que é a quantidade de pulso do sensor em três segundo e multiplicado por vinte na sequência a variável “Vv” recebe a equação de velocidade, fornecida pelo fabricante do sensor, e impresso a direção em números, programação na figura 53.

```
//////////////////prepara dados para leitura////////////////////////////////
 ldata1=ldata1*255;
 ldata=ldata1+ldata2;
 m=ldata/10000;
 h=ldata%10000;
 d=h%100;
 h=h/100;
 m=m*10;
 printf(usb_cdc_putc, "%02lu:%02lu/%02lu\r\n",h, m, d);  // imprimi hora que dado foi coletado.
  delay_ms(500);

  i2c_start();
 dir=le_eeprom (0,n1);  //le dado de direção.
  delay_ms(40);
 n1++;
 vel=le_eeprom (0,n1);  //le dado de velocidade.
 delay_ms(40);
 i2c_stop();
 rpm=(vel*20);// como aquantidade de pulso foi baseada em tres 3seg mutiplica por 20 = 60 seg.
v=(((4*pi*raio*rpm)/60)/1000)*3,6; // equação de v elocidade do vento em Km/h.
  delay_ms(40);
 n1++;
 n1++;
 printf(usb_cdc_putc, "direcao:%01u\r\n", dir);//imprimi direção em numero
```

Figura 53 – Programação de impressão de dados(b)
Fonte: Autoria própria.

Para imprimir a direção por coordenada o programa compara o dado pela função “if” também conhecida com se, em baixo imprimi a variável “Vv” que foi calculado a velocidade, programação em seguida figura 54.

```
    //////////////imprimi direção em cordenada cartesiana////////////
    if (dir==1){
     printf(usb_cdc_putc, " N\r\n");
    }
     if (dir==2){
     printf(usb_cdc_putc, " NO\r\n");
    }
     if (dir==3){
     printf(usb_cdc_putc, " O\r\n");
    }
     if (dir==4){
     printf(usb_cdc_putc, " SO\r\n");
    }
     if (dir==5){
     printf(usb_cdc_putc, " S\r\n");
    }
     if (dir==6){
     printf(usb_cdc_putc, " SE\r\n");
    }
     if (dir==7){
     printf(usb_cdc_putc, " E\r\n");
    }
     if (dir==8){
     printf(usb_cdc_putc, " NE\r\n");
    }
    delay_ms(500);
    printf(usb_cdc_putc, "velocidade:%f Km/h\r\n",Vv); //imprimi velocidade.
    delay_ms(500);
    b++;
    }
```

Figura 54 – Programação de impressão de dados(c)
Fonte: Autoria própria.

O laço é repetido até ser mostrados todos os dados de direção e velocidade do vento, e a hora e o dia adquirido, depois é impresso uma mensagem na tela onde se pergunta se quer apagar os dados demostrados, se sim o programa escreve zero no espaço da memória reservado para quantidade de vez que o programa fez leitura dos sensores, assim o programa começa a gravar a partir do endereço de memória um. Abaixo na figura 55 pode ser visto essa programação e na figura 56 o resultado de como é mostrado no PC.

```
    printf(usb_cdc_putc, "  PARA APAGAR DADOS DIGITE 1 OU OUTRO NUMERO PARA IGNORAR  //\r\n");
    P = usb_cdc_getc();
    if (p==1){

    i2c_start();
escreve_eeprom(0,0,0);
```

Figura 55 – Programação para apagar dados
Fonte: Autoria própria.

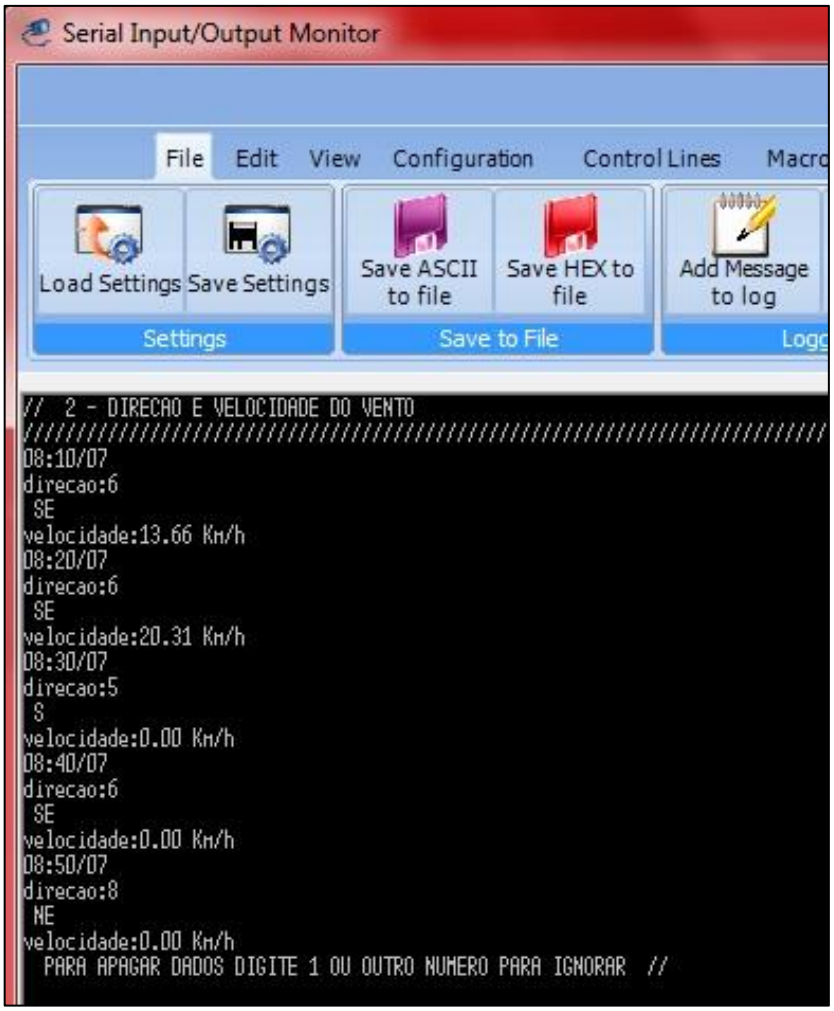


Figura 56 – Resultado de impressão
Fonte: Autoria própria.

4.8.3 Visualização dos dados do pluviômetro

Se escolhido o três no menu, será impresso os dados de quantidade de chuva no computador, o princípio de programação é parecido com o do tópico acima, ele lê a quantidade de dados através da leitura do endereço de memória três mil, as variáveis “n1” e “m1” serão utilizados para acerto de sequência para leitura dos endereços de memória, depois o programa entre no laço para impressão de todos dados e efetuada a leitura dos dados de data e hora como mostra a figura 57.

```
      case 3:
   printf(usb_cdc_putc, "////////////////////////////////////
   delay_ms(200);
   printf(usb_cdc_putc, "//  3 - QUANTIDADE DE CHUVA
   delay_ms(200);
   printf(usb_cdc_putc, "////////////////////////////////////
    i2c_start();
    ng1 = le_eeprom (0,3000);
    delay_ms(40);
    i2c_stop();
    n1=3004;
    b=0;
    delay_ms(200);
    m1=3001;

    while (b<ng1){    //roda até imprimir todos os dados
////////////////lê data e hora/////////////////
  i2c_start();
 ldata1=le_eeprom (0,m1);
  delay_ms(40);
 m1++;
 ldata2=le_eeprom (0,m1);
  delay_ms(40);
 m1++;
  m=le_eeprom (0,m1);
  delay_ms(40);
 i2c_stop();
```

Figura 57 – Programação, impressão anemômetro (1)
Fonte: Autoria própria.

Como no programa de velocidade e direção é preparados os dados para imprimir hora da coleta dos dados, a diferença é que “m” já recebe os minutos exatos como visto na programação de leitura do sensor, “b” recebe zero para começar contagem de milímetro de chuva por dia através da varável “qntd” que multiplica “b” por 0,25 milímetros em seguida entra em um laço que imprimi os dados até acabar ou mudar de dia, se mudar de dia o programa sai e zera “b” novamente para contar a partir do novo dia figura 58.

```
//////////configura data e hora/////////////////////
    ldata1=ldata1*255;
    ldata=ldata1+ldata2;
    m=ldata/10000;
    h=ldata%10000;
    d=h%100;
    h=h/100;
   // m=m*10;
    dia1=d;  // dia1 recebe o dia
    b1=0;   // zera para conta quantidade
    //if (dia1==d){
    while ((dia1==d)&&(b<ng1)){    // roda até imprimir todos os dados ou até mudar de dia
      b=b+1; // incrementa até atingir quantidade de dados total (memoria 3000)
     b1=b1+1;

    delay_ms(500);
    printf(usb_cdc_putc, "%02u:%02u/%02u \r\n",h,m,d);
    delay_ms(500);

     qntd=b1*0.25;    //calcula a quantidade em milimitros.

    printf(usb_cdc_putc, "chuva dia %02u: %f mm\r\n",dia,qntd);
    i2c_start();
```

Figura 58 – Programação, impressão anemômetro (2)
Fonte: Autoria própria.

Na figura 59 encontra-se segunda parte do laço que faz a leitura da data para ver se o dia é diferente de quando entrou, se sim sai do laço com m1 recebendo o mesmo endereço para ser lido novamente, se não, imprimi mais esse dado rodando novamente o programa.

```
/////faz a leitura para ver se mudou de dia se sim sai do laço e zera b1//////
///// se não continua imprimindo/////////////////////////////////////////////////
   ldata1=le_eeprom (0,n1);
    delay_ms(40);
   n1++;
   ldata2=le_eeprom (0,n1);
    delay_ms(40);
   n1++;
     m=le_eeprom (0,n1);
     delay_ms(40);
   i2c_stop();
   n1++;

   ldata1=ldata1*255;
   ldata=ldata1+ldata2;
   m=ldata/10000;
   h=ldata%10000;
   d=h%100;
   h=h/100;
   }
   m1=n1-3; //acerta o endereço memoria se sair do laço por dia diferete.
```

Figura 59 – Programação, impressão anemômetro (3)
Fonte: Autoria própria.

Como já visto no final é impresso na tela se quer apagar os dados, se sim, só seguir a instrução do programa apertando o número um, se não, aperta-se outro número como mostra a programação.

```
   printf(usb_cdc_putc, "  PARA APAGAR DADOS DIGITE 1 OU OUTRO NUMERO PARA IGNORAR
   P = usb_cdc_getc();
   if (p==1){

   i2c_start();
  escreve_eeprom(0,3000,0);
   delay_ms(40);
   i2c_stop();
   }
```

Figura 60 – Programação apagar dados
Fonte: Autoria própria.

Na figura 61 o resultado da impressão no computador feito através de uma simulação de chuva.

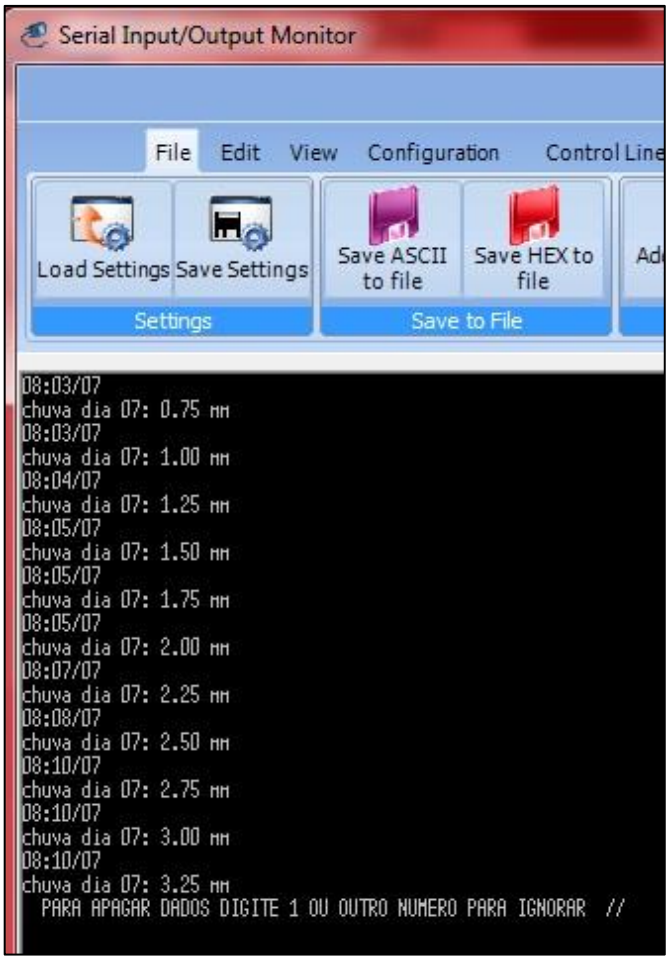


Figura 61 – Tela de dados de quantidade de chuva
Fonte: Autoria própria.

4.8.4 Visualização dos dados de Temperatura

No case 4 será impresso os dados de temperatura, o programa é parecido com de velocidade e direção do vento, ele lê os dados e imprimi data hora, a diferença é que incrementando três vezes o endereço de memória para chegar ao endereço de temperatura, já que nos dois abaixo de data e hora é gravadas direção e velocidade do vento, a temperatura e impresso do jeito que foi gravado na memória sem nenhuma manipulação.

```
    while (b<ng1){
i2c_start();
ldata1=le_eeprom (0,n1);
 delay_ms(40);
n1++;
ldata2=le_eeprom (0,n1);
i2c_stop();
 delay_ms(40);
n1++;
n1++;
n1++;
ldata1=ldata1*255;
ldata=ldata1+ldata2;
m=ldata/10000;
h=ldata%10000;
d=h%100;
h=h/100;
m=m*10;
printf(usb_cdc_putc, "%02lu:%02lu/%02lu\r\n",h, m, d);
 delay_ms(200);
 i2c_start();
temperatura =le_eeprom (0,n1);
i2c_stop();
 delay_ms(40);
n1++;
 delay_ms(200);
printf(usb_cdc_putc, "temperatura:%02u °c \r\n",temperatura);
 delay_ms(200);
b++;
}
```

Figura 62 – Programação visualização de dados temperatura
Fonte: Autoria própria.

Nesse caso também tem se a opção “apagar os dados”, lembrando que os dados são gravados no mesmo laço de varredura então se apagado aqui ou no de direção e velocidade irá perder os três dados.

4.9 PROTÓTIPO

Para os testes finais, coleta dos dados e amostra dos resultados descritos acima, os circuitos foram montados em uma placa de fenolite, (estilo universal furada) os sensores ficaram instalados no fundo de um terreno e foram ligados com a placa através de plugues P2, o resultado na imagem da figura 63.

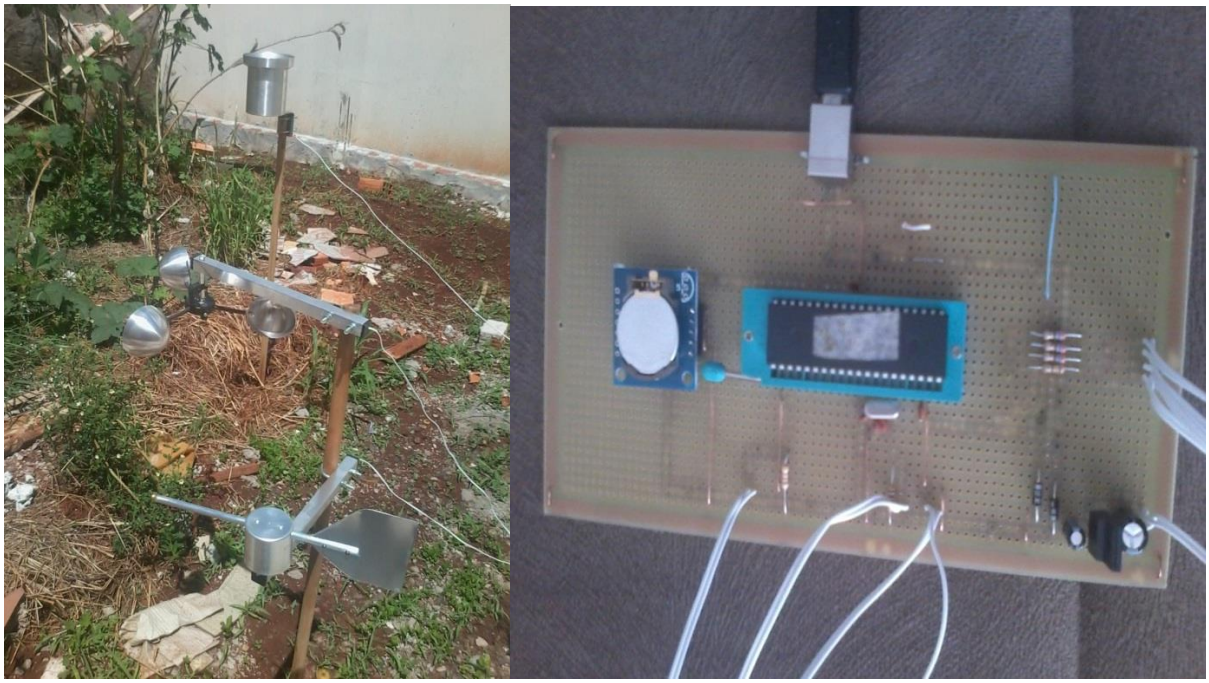


Figura 63 – Protótipo.
Fonte: Autoria própria

Para testar a eficiência de leitura e armazenamento, foram realizados alguns testes, para o indicador de vento o sensor foi fixado em quatro posições norte, sul, leste e oeste na leitura dos dados gravado indicaram as quatro referências: Para velocidade, foi simulada uma determinada quantidade de pulso no período de leitura e depois calculado na formula indicada pelo fabricante do sensor o resultado foi o mesmo, para quantidade de chuva foi colocado água no suporte do pluviômetro foi possível saber a hora do pulso de 0,25 milímetros pelo barulho do mecanismo o resultado foi satisfatório, foi incrementado cada pulso na hora exata e para temperatura foi medido a saída do sensor com um voltímetro sendo 10 *milivolts* por grau *célsius*.

Não foi realizado teste de eficiência dos sensores, contudo ressalta-se que o foco do trabalho nesse momento é o desenvolvimento do projeto de um *datalogger*.

5 CONSIDERAÇÕES FINAIS

5.1 ANALISE DOS RESULTADOS

Os resultados dessa pesquisa foi satisfatório primeiramente pelo conhecimento obtido com estudo do microcontrolador PIC18f4550 que foi o principal componente estudado para criação desse projeto e também pelos objetivos alcançado tais como a definições das vareáveis a serem mensuradas e a implementação a traveis dos sensores, que foi um excelente teste de entradas para o *datalogger,* foram utilizadas entradas analógicas onde cada referência de tensão representa um resultado e também entradas digitais, onde o resultado se faz através de pulsos, e esses dois tipos abrangem as maiorias de sensores existentes. Para entradas analógicas o programa está configurado para mais três entradas AN2, AN3 e AN4. Podendo assim ser instalados sensores que não foram testados mais que possam ser acrescentado ao projeto como sensor de radiação e umidade.

A forma de armazenamento através do chip EEPROM atendeu as expectativas por ser fácil o armazenamento e por ter a possibilidade de acrescentar mais sete EEPROM no barramento, aumentando muito a autonomia de armazenamento de dados. No que se trata de comunicação USB o sistema do projeto se portou muito bem, sem travamento e se conectando ao computador assim que solicitado. A fonte de alimentação não gerou nenhum tipo de ruído e ou interferência, garantindo assim um bom funcionamento do sistema e também da à possibilidade do protótipo ser testado em campo já que é alimentado por uma bateria de nove *volts*

5.2 CONCLUSÃO

O protótipo do projeto teve um custo aproximado sem os sensores de oitenta e seis reais, já com os sensores um pouco mais de quatrocentos e

sessenta reais como pôde ser visto na lista de materiais na seção de matérias e métodos, mostrando que o projeto pode ser viável financeiramente.

Portanto com os resultados obtidos e com melhorias que possam ser aplicadas em trabalhos futuros o dispositivo pode ser uma excelente ferramenta para o pequeno agricultor auxiliando em atividades e tomadas de decisão, com um histórico meteorológico pode ser mais bem conhecido o clima especifico de uma determinada região, como por exemplo, temperatura, radiação solar, velocidade do vento entre outros, e com isso pode se escolher melhor o que cultivar e ou as ações a se tomar para melhorar o cultivo como a irrigação ou utilização de estufas por exemplo.

5.3 SUGESTÕES PARA TRABALHOS FUTUROS

Por não ter tempo hábil para desenvolver um produto pronto para ser utilizado no campo segue algumas ideias de melhorias e de continuidade do sistema para trabalhos futuros.

A interface com o usuário foi de grande importância para o projeto demostrando os dados corretamente de uma forma bem didática e testando a comunicação via porta serial, contudo a disponibilização para usuários deve ser desenvolvida uma nova interface onde facilite a transferência de dados para computador já que no utilizado no projeto, quando há uma grande quantidade de dados impresso na tela os primeiros desaparecem. A também a opção de criação de uma IHM (Interação Homem-Máquina) onde possa ser visualizado e manipulados os dados obtido, assim não havendo a necessidade de utilizar.

Para o produto final se faz necessário o desenvolvimento de uma carcaça e placas de circuitos com proteção contra condições do ambiente tais como poeira e umidade.

Como melhoria do projeto descrito nesse trabalho é entendido que os intervalos temporais para leitura dos sensores podem ser alterados ou até mesmo criado um sistema para controle do usuário para que ele coloque o tempo para cada coleta, adição de novos sensores como já citado acima e outras ideias que possam ser inspiradas com a leitura do mesmo.

REFERÊNCIAS

AYOADE, J.O. **Introdução á climatologia para os trópicos**. 4ª ed. Rio de Janeiro: Bertrand Brasil, 1996.

CASAROLI, Derblai. et al. Radiação Solar e aspectos fisiológicos na cultura de soja - uma revisão. **Revista da FZVA**, Uruguaiana, v.14, n.2, p. 102-120. 2007. Disponível em: <http://revistaseletronicas.pucrs.br/ojs/index.php/fzva/article/download/2502/1961. >. Acesso em: 18 out. 2015

DIAS, Carlos Alberto Antunes. **Procedimentos de medição e aquisição de dados de uma torre Micrometeorológica em Sinop-mt.** 2007. 75 p. Dissertação (Pós-graduação em Física em Meio Ambiente) - Universidade Federal de Mato Grosso, Cuiabá, 2007. Disponível em: < http://www.pgfa.ufmt.br/index.php/br/utilidades/dissertacoes/99-carlos-alberto-antunes-dias/file. >. Acesso em: 24 de Set. 2015.

MAXIM INTEGRATED. **DS1307 64 x 8, Serial, I2 C Real-Time Clock.** Data Sheet. 2015.

MELO, T. Henrique. Comunicação IC2. **Microcontrolandos,** Dez. 2012. Disponível em:<http://microcontrolandos.blogspot.com.br/2012/12/comunicacao-i2c.html> Acesso em: 10 nov. 2015.

MICROCHIP TECHNOLOGY. **PIC 18F4550**: Data Sheet. 2007.

MICROCHIP TECHNOLOGY. **512K I2C Serial EEPROM:** Data Sheet. 2010.

MICROGENIO SOLUÇÕES ELETRONICA. **Manual MicroICD.** Disponível em: <http://www.microgenios.web2253.uni5.net/news/manuais_kits/manual_microicd_zif.pdf> acesso em: 22 fev. 2016.

MIYADAIRA, Alberto Noboru. **Microcontroladores PIC18:** aprenda e programe em linguagem C. 2. ed. São Paulo: Érica, 2011.

RESENDE ,Sebastião Antônio Azevedo. RESENDE JÚNIOR , Joaquim Carlos de. Interferência dos ventos no cultivo de plantas: Efeitos prejudiciais e práticas preventivas. **ENCICLOPÉDIA BIOSFERA**, Goiânia, V.7, N.12, 2011. Disponível em: <http://www.conhecer.org.br/enciclop/2011a/agrarias/interferencia%20dos%20ventos.pdf>.Acesso em 20 de set. 2015.

ROCHA, Olavo Acyr de Lima. **Atividade Agrária conceito clássico conceito moderno de Antônio Carrozza.** 1999, São Paulo. Disponível em: <http://www.revistas.usp.br/rfdusp/article/view/67431/70041>.Acesso em: 15 de out. 2015.

SACCO, Francesco**.** Módulo Tiny RTC I2C. **Embarcados**, jul. 2015. Disponível em: < http://www.embarcados.com.br/modulo-tiny-rtc-i2c-parte-1/> Acesso em: 20 set. 2015.

SOUZA, David José de. **Desbravando o PIC.** 4. ed. São Paulo: Érica, 2001.

TEXAS INTRUMETS. **µa7800 Series positive-voltage regulators: Data Sheet**. 2003.

TEXAS INTRUMETS. **LM35 Precision Centigrade Temperature Sensors**: Data Sheet. 2016.

TORRES, Eloiza Cristiane. **Adaptação do texto Clima e agricultura**. 20--. Disponível em:< http://pesca.iff.edu.br/curso-de-especializacao-em-pesca-aquicultura-e-ambiente/meteorologia-aplicada-a pesca/artigos/Adaptacao%20do%20texto%20CLIMA%20E%20AGRICULTURA.pdf/at_download/file.>. Acesso em 20 de set. 2015

WENDLING, Marcelo. **Sensores.** 2010, 19 p**.** disponível em: < http://www2.feg.unesp.br/Home/PaginasPessoais/ProfMarceloWendling/4---sensores-v2.0.pdf>. Acesso em: 12 de out. 2015.

WRFCOMERCIAL. [mensagem pessoal]. Mensagem recebida por <marcelovitorino_pnh@hotmail.com> em 12 maio 2015.

ZUBEN, Von. **Introdução à Programação em Linguagem C**. 2015. Disponível em: <ftp://ftp.dca.fee.unicamp.br/pub/docs/vonzuben/ea869_1s04/linguagemC1.pdf>.Acesso em: 23 de set. 2015.