

UNIVERSIDADE DO OESTE DE SANTA CATARINA – UNOESC

ANDRÉ UMBERTO FACCIÓNI

**DESENVOLVIMENTO DE UM KIT DIDÁTICO DESTINADO AOS ESTUDOS DE
CLP E AUTOMAÇÃO**

Joaçaba

2017

ANDRÉ UMBERTO FACCIÓNI

**DESENVOLVIMENTO DE UM KIT DIDÁTICO DESTINADO AOS ESTUDOS DE
CLP E AUTOMAÇÃO**

Trabalho de conclusão de curso apresentado à
Universidade do Oeste de Santa Catarina como
requisito à obtenção do grau especialização em
*Engenharia de Sistemas Computacionais
Embarcados.*

Professor orientador: Dr. Geovani Rodrigo Scolaro

Joaçaba

2017

“Uma pessoa inteligente resolve um problema, um sábio o previne”.

(Albert Einstein)

RESUMO

Este trabalho apresenta o desenvolvimento de um sistema eletrônico para fins didáticos, onde aborda em método passo-a-passo os procedimentos de programação e estudos de sistemas automatizados utilizando Microcontrolador. A estrutura final, apresenta um hardware capaz de implementar as principais funções de um Controlador Logico Programado (CLP), além de utilizar um software capaz de realizar a programação usando a linguagem ladder. O sistema desenvolvido para fins didáticos, também é capaz de ser implementado em pequenas automações, pois foi criado para ser inserido em um gabinete apropriado e instalado em um painel elétrico. O trabalho em um todo será aplicado para o ensino de sistemas automatizado na unidade SENAI-Xanxerê, onde o autor trabalha como professor em cursos técnicos.

Palavras – chave: Microcontrolador. Controlador lógico programável. CLP. Automação.

ABSTRACT

This Project presents the developing of an electronic system, which will be used for didactic purposes. It will approach, step by step, all the procedures of programming and studies of automated systems using microcontrollers. The final Structure presents a hardware, which is able to implement the main functions of a Programmable Logical Controller (PLC), besides using a software able to realize the programming using the Ladder language. The system developed for didactic purposes, also can be implemented in small automations, because it was created to be inserted in an appropriated cabinet and installed in an electric panel. Therefore, the work, will be applied when teaching automated systems at the unit of SENAI-Xanxerê, where the author works as a teacher in technical courses.

Key – Words: Microncontroller. Programmable Logic Controller. PLC. Automation.

LISTA DE TABELAS

Tabela 1 - Custos placa V1.1	51
Tabela 2 - Componentes circuito Clock	59
Tabela 3 - Componentes circuito Reset.....	59
Tabela 4 – Componentes circuitos de alimentação	60
Tabela 5 – Componente sensor de temperatura	61
Tabela 6 – Componentes entrada analógica	62
Tabela 7 - Custos unitário do CLP	68

LISTA DE ILUSTRAÇÕES

Figura 1 - Organização interna do Microcontrolador PIC.....	15
Figura 2 - Pinagem PIC16f876A	17
Figura 3 - Exemplo de programação no LDmicro.....	18
Figura 4 - Descrição do Menu Arquivo	20
Figura 5 - Exemplo de arquivo Exportado como Texto	21
Figura 6 - Descrição do Menu Editar	21
Figura 7 - Exemplo de linha inseria antes do cursor.....	21
Figura 8 - Exemplo de linha inseria depois do cursor.....	22
Figura 9 - Menu de Configurações	22
Figura 10 - Configuração do Microcontrolador	23
Figura 11 - Menu de escolha do Microcontrolador.....	24
Figura 12 - Arquivo salvo em ANSI C.....	24
Figura 13 - Menu Instruções	25
Figura 14 - Seleção de tipos de contatos NA.....	26
Figura 15 - Seleção de tipos de contatos NF	26
Figura 16 - Seleção de bobina de saída Normal	27
Figura 17 - Seleção de bobina de saída Negado	27
Figura 18 - Seleção de bobina de saída SET	28
Figura 19 - Seleção de bobina de saída RESET.....	29
Figura 20 - Utilização do temporizador TON.....	29
Figura 21 - Utilização do temporizador TOF	30
Figura 22 - Utilização do temporizador RTO.....	31
Figura 23 - Utilização da função RESET	31
Figura 24 - Utilização da função Borda de Subida.....	32
Figura 25 - Utilização da função Borda de Descida.....	32
Figura 26 - Utilização Circuito Fechado e Circuito Aberto	33
Figura 27 - Utilização Relé Principal.....	33
Figura 28 - Utilização instrução MOV	34
Figura 29 - Utilização instrução ADD	35
Figura 30 - Utilização instrução de Comparação	36
Figura 31 - Utilização instrução de Contador (CTU).....	36

Figura 32 - Utilização instrução de Contador Circular	37
Figura 33 - Utilização instrução de Shift Register	38
Figura 34 - Utilização instrução de Look-up Table.....	38
Figura 35 - Utilização instrução de Linearização Por Segmento	39
Figura 36 - Utilização Conversor A/D	40
Figura 37 - Utilização instrução PWM	41
Figura 38 - Utilização instrução Salva EEPROM	42
Figura 39 - Utilização instrução Recebe UART Serial.....	42
Figura 40 - Utilização instrução Envia UART Serial.....	43
Figura 41 - Utilização instrução Envia String	44
Figura 42 - Menu Simular	47
Figura 43 - Menu Simular	47
Figura 44 - Menu Compilar.....	48
Figura 45 - Menu Ajuda	48
Figura 46 - Software de Programação Tiny.....	49
Figura 47 - Montagem do protótipo em protoboard	50
Figura 48 - Placa protótipo Versão 1.0.....	51
Figura 49 - Placa protótipo Versão 1.1.....	52
Figura 50 - Placa com Componentes	53
Figura 51 - Dimensões Gabinete PATOLA	54
Figura 52 - Gabinete e placa eletrônica	54
Figura 53 - Conexões microcontrolado PIC16f876A	55
Figura 54 - Circuito de clock e circuito de reset.....	55
Figura 55 - Circuitos de comunicação	56
Figura 56 - Circuito da fonte de alimentação	56
Figura 57 - Entrada de sinais analógicos.....	56
Figura 58 - Saídas digitais a relé.....	57
Figura 59 - Entradas digitais	57
Figura 60 - Layout PCI.....	58
Figura 61 - Circuito Reset e Clock	59
Figura 62 - Circuito de alimentação	61
Figura 63 - Sensor de temperatura.....	61
Figura 64 - Configuração de Entrada Analógica.....	62

SUMÁRIO

1.	INTRODUÇÃO	12
1.1	TEMA DE PESQUISA	12
1.2	FORMULAÇÃO DO PROBLEMA	12
1.3	HIPÓTESE DE PESQUISA	13
1.4	OBJETIVOS	13
1.4.1	Objetivo Geral	13
1.4.2	Objetivos Específicos	13
1.5	JUSTIFICATIVA	14
2	FUNDAMENTAÇÃO TEÓRICA	14
2.2	MICROCONTROLADORES	14
2.2.1	Memória	15
2.2.2	Temporizadores e Contadores	15
2.2.3	ULA (Unidade Lógica e Aritmética)	16
2.3	MICROCONTROLADOR PIC16F876A	16
2.3.1	Processador RISC (Reduce Instruction Set Computer)	16
2.3.2	Características da memória	16
2.3.3	Características de periféricos	17
2.3.4	Pinagem do PIC 16F876A	17
2.4	SOFTWARE LDmicro	17
3	METODOLOGIA	19
3.1.	SOFTWARE DE PROGRAMAÇÃO	19
3.1.1.	Menus de parâmetros do LDmicro	19
3.2	MENU ARQUIVO	20
3.2.1.	Menu Editar	21
3.2.1.1	Menu Configurações	22
3.2.1.2	Menu Instruções	25
3.3	SOFTWARE DE GRAVAÇÃO	48
3.4	DESENVOLVIMENTO DO PROTÓTIPO	49
3.4.1	Gabinete ABS	53
3.4.2	Diagrama Eletrônico	55
3.4.3	Montagem PCI	58

3.4.4	Circuito de Clock e Reset.....	58
3.4.5	Circuito de Alimentação.....	60
3.4.6	Sensor de temperatura LM35.....	61
3.4.7	Entrada analógica e trimpot.....	62
3.4.8	Lista de Exercícios	63
3.4.9	Apresentação PowerPoint.....	64
3.4.10	Valores	67
4	CONCLUSÕES	69

1. INTRODUÇÃO

O trabalho de conclusão proposto foi desenvolvido como material didático aplicado a cursos técnicos e cursos de qualificação de base tecnológica, onde o autor atua como docente na unidade SENAI/Xanxerê, possibilitando o preparo de estudantes para o mercado de trabalho. Com o conhecimento e habilidades adquiridos pelos estudantes, torna possível o desenvolvimento de novos produtos tecnológicos, como por exemplo, ao setor agroindustrial, que é o grande foco econômico da região oeste de Santa Catarina. Permitindo assim a permanência de jovens estudantes em seus locais de origem e não migrando para grandes centros a procura de ensino e emprego, pois na região tem-se uma grande deficiência de profissionais com conhecimento de sistemas tecnológico.

O presente trabalho agrega documentação técnica e didática, utilizando a plataforma com microcontrolador PIC16F876A, que promoverá o aprendizado do aluno no desenvolvimento e aplicação de um sistema automatizado específico. Esta plataforma integra uma Placa de Circuito Impresso (PCI), que será confeccionada em uma empresa especializada.

No decorrer do desenvolvimento deste trabalho, identificou-se que além de ser um equipamento para fins didático, pôde ser elaborado de uma forma que possa atender pequenas automações para o setor agroindustrial ou outras aplicações que necessitem um controle automatizado, o layout da placa PCI foi desenvolvido para ser inserida em um gabinete plástico em (ABS) que atende a norma (DIN), dessa forma protegendo o circuito eletrônico e tornando possível a inserção em painel elétrico.

Para financiar os custos com o desenvolvimento deste trabalho, o autor utilizará partes do recurso oriundos do Programa de Bolsas Universitárias de Santa Catarina UNIEDU Chamada Pública 19/SED/2015.

1.1 TEMA DE PESQUISA

Sistemas embarcados Controlador Lógico Programável (CLP) e educação

1.2 FORMULAÇÃO DO PROBLEMA

Como desenvolver um kit didático de baixo custo, para ensino de sistemas automatizados utilizando CLP em unidade ensino profissionalizante.

1.3 HIPÓTESE DE PESQUISA

Com este trabalho pretende-se desenvolver uma plataforma didática incluindo hardware, software e material descritivo para promover o ensino e compreensão de sistemas automatizados utilizando CLP.

1.4 OBJETIVOS

1.4.1 Objetivo Geral

Desenvolver uma plataforma didática para estudos e aprendizado em cursos técnicos e de qualificação em eletrônica e automação.

1.4.2 Objetivos Específicos

- Desenvolver um protótipo de hardware para auxiliar nos estudos de sistemas eletroeletrônicos;
- Produzir documentação técnica para estudos em sistema eletrônicos e automação;
- Testar em laboratório a funcionalidade do kit didático;
- Avaliar a funcionalidade de todos o sistema, hardware, software e bibliografia construída.

1.5 JUSTIFICATIVA

É possível observar que na região oeste de Santa Catarina no setor agroindustrial está defasado tecnologicamente. Algumas esferas na agroindústria poderiam ser mais produtivas, e deixam de ser devido ao baixo conhecimento das tecnologias existentes. Um exemplo clássico da região é o controle de temperatura de aviários ou chiqueirões, onde há a necessidade do produtor controlar constantemente a abertura e fechamento das cortinas, este controle é feito muitas vezes manualmente. Esta deficiência de controle prejudica o desenvolvimento dos animais, e exige uma dedicação constante do produtor.

Com a implementação um sistema automatizado para controle, todo o sistema produtivo torna-se mais eficiente, o recurso humano envolvido no processo poderá dedicar seu tempo em outras atividades. Portanto, o trabalho aqui proposto poderá solucionar problemas como o mencionado no exemplo acima, facilitando a vida das pessoas no campo e tornando os produtos mais competitivos no mercado, assim observa-se surgimento de um nicho de mercado para a região, gerando emprego e renda na economia local.

2 FUNDAMENTAÇÃO TEÓRICA

Neste tópico serão apresentados os estudos e literatura envolvendo o trabalho. Entre os assuntos estão sistemas embarcados, microcontroladores.

2.2 MICROCONTROLADORES

Microcontrolador é um dispositivo semicondutor em forma de CI, que integra todas as partes básicas de um microcomputador: microprocessador, memórias e portas de entrada e saídas. Ele é conhecido como um microcomputador implementado em um único CI. Geralmente é limitado em termos de quantidade de

memória. É muito utilizado em processos de automação, robótica, controladores lógicos programáveis, alarmes, máquinas de lavar roupa etc. Apresenta um custo bastante baixo e várias empresas dedicam-se a fabricação de microcontroladores, tais como: Motorola, Microchip, Mitsubishi, NEC, Philips, SGS, Intel, Hitachi, Toshiba, ATmel etc. (GIMENEZ, 2002).

Conforme pode ser observado na Figura1 a arquitetura em alto nível do Microcontrolador PIC. Essa figura apresenta módulos de memória, CPU, temporizadores, interfaces de entrada e saída.

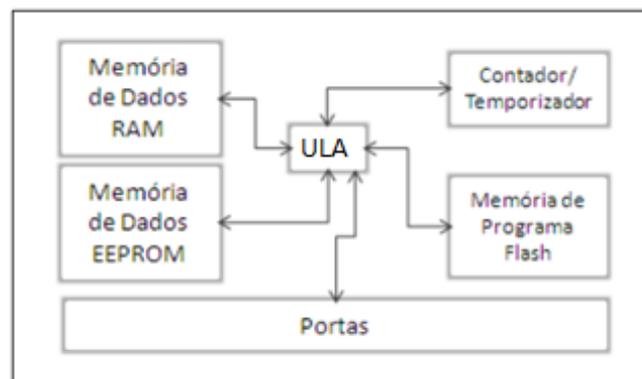


Figura 1 - Organização interna do Microcontrolador PIC
Fonte: (GIMENEZ, 2002)

2.2.1 Memória

A memória é um dos componentes internos principais do microcontrolador, pode-se dividi-la em duas categorias: memória de programa e memória de dados.

A memória de programa é onde são carregados os programas em execução e nela os programas podem gerenciar comunicação de entrada e saída, executar instruções, etc.

A memória de dados é utilizada para o armazenamento de resultados e dados que a CPU deve executar.

2.2.2 Temporizadores e Contadores

Os temporizadores são programados por software e podem operar independente dos demais sistemas do chip. São utilizados normalmente para retardos, geração de pulsos, entre outras rotinas onde há uma necessidade de um controle temporal ou de uma contagem.

2.2.3 ULA (Unidade Lógica e Aritmética)

A ULA está presente em todos os microprocessadores. Ela contém circuitos destinados a realizar funções de cálculo e manipulação de dados durante a execução de um programa. (Santos, 2009)

2.3 MICROCONTROLADOR PIC16F876A

O PIC16F876A fabricado com a tecnologia CMOS da fabricante Microchip, seguem algumas especificações deste dispositivo:

2.3.1 Processador RISC (Reduce Instruction Set Computer)

- Possui 35 instruções de 14 bits;
- Frequência máxima de funcionamento - 20Mhz (frequência do cristal);
- Cada ciclo de relógio corresponde à frequência do cristal / 4 = 5us, efetuando a cada segundo 5 MIPS (milhões de instruções por segundo);
- Tempo de execução das instruções: 1 ciclo de relógio.

2.3.2 Características da memória

- Memória de programa (FLASH) de 8K (word) de 14 bits;
- Cada instrução é codificada numa word de 14 bits;
- Memória de dados RAM de 368 bytes;
- Memória de dados EEPROM de 256 bytes.

2.3.3 Características de periféricos

- 22 linhas de entrada/saída, agrupadas em 3 ports (PORTA, PORTB e PORTC);
- Conversor analógico à digital de 10 bits, com um máximo de 5 canais de entrada analógica;
- USART (Universal Synchronous Asynchronous Receiver Transmitter);
- 13 tipos de interrupções, por exemplo externa RB0/INT, TMR0.

2.3.4 Pinagem do PIC 16F876A

A Figura 2 ilustra a pinagem do PIC 16F876A e suas ligações específicas para cada porta de conexão.

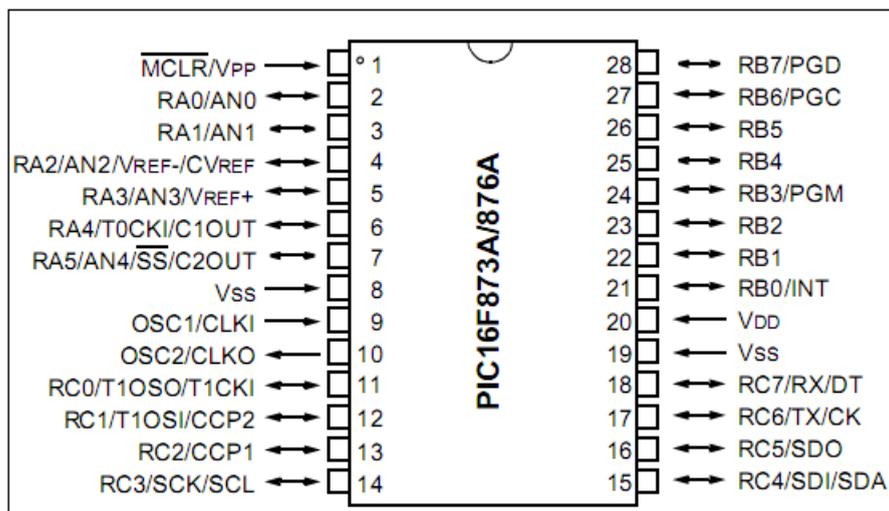


Figura 2 - Pinagem PIC16f876A

Fonte: (MICROCHIP, 2013)

2.4 SOFTWARE LDmicro

LDmicro é um editor de lógica ladder, simulador e compilador para Microcontroladores de 8 bits. Ele pode gerar código nativo para Atmel AVR e Microchip família PIC16F.

Este programa é um software livre que utiliza o princípio GNU ou GPL e foi desenvolvido por *Jonathan Westhues*. (Westhues, 2007)

Em termos gerais, a GPL baseia-se em 4 liberdades:

- 1- A liberdade de executar o programa, para qualquer propósito (liberdade nº 0)
- 2- A liberdade de estudar como o programa funciona e adaptá-lo às suas necessidades (liberdade nº 1). O acesso ao código-fonte é um pré-requisito para esta liberdade.
- 3- A liberdade de redistribuir cópias de modo que você possa ajudar ao seu próximo (liberdade nº 2).
- 4- A liberdade de aperfeiçoar o programa e liberar os seus aperfeiçoamentos, de modo que toda a comunidade beneficie deles (liberdade nº 3). O acesso ao código-fonte é um pré-requisito para esta liberdade.

LDmicro é um compilador que ao contrário dos compiladores típicos, ele não aceita um texto arquivo como sua entrada, em vez disso, o programa é editado graficamente. O editor representa o programa como uma estrutura de árvore que pode ser observado na Figura 3, onde cada nó representa uma instrução (como relés contatos ou uma bobina) ou um subcircuito em série ou paralelo, exatamente como é utilizada a programação tradicional Ladder usada para programar a grande maioria dos CLPs encontrados no mercado.

A programação também pode ser simulada no próprio programa LDmicro e posteriormente ser compilada em formato .hex para ser gravada no Microcontrolador.

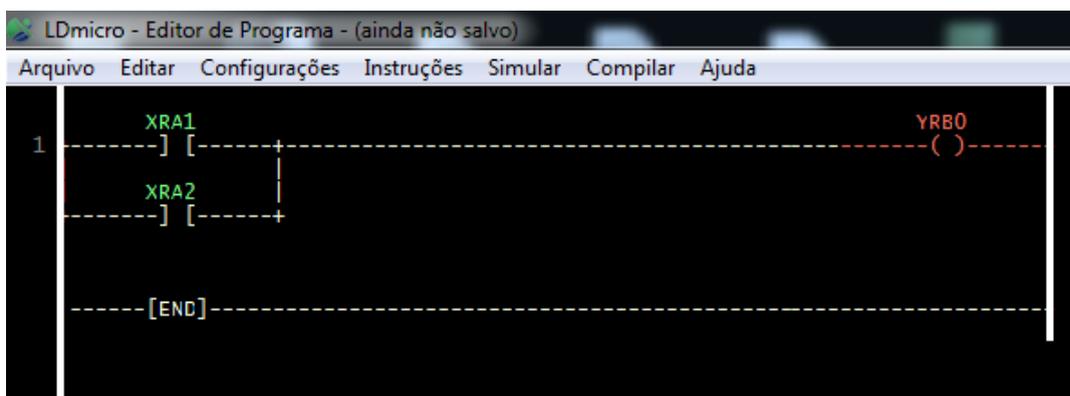


Figura 3 - Exemplo de programação no LDmicro

Fonte: O Autor, 2017

3 METODOLOGIA

Para o desenvolvimento deste trabalho serão necessárias as seguintes etapas:

- a) **Software de programação:** Identificar um software gratuito de programação ladder para utilizar com o protótipo.
- b) **Desenvolvimento de um protótipo:** Desenvolver um protótipo de kit didático para sistemas automatizados.
- c) **Testes:** Elaborar inúmeros testes com o protótipo desenvolvido.
- d) **Documentação:** Elaborar documentação técnica para o ensino de programação de CLP e sistemas automatizados.

3.1. SOFTWARE DE PROGRAMAÇÃO

Depois de muitas pesquisas foi encontrado um software que atenderia as características essenciais deste projeto. A principal característica exigida para a escolha do programa, é que ele deveria ter como linguagem padrão de programação a Ladder e que seja uma versão gratuita.

O software LDmicro atendeu as características e possui o seu código fonte disponível para qualquer usuário ou desenvolvedor possa fazer modificações e aprimoramento, desde que atenda os critérios de desenvolvimento GNU.

A seguir será descrito todas as funções implementadas no programa LDmicro, além de apresentar e descrever o funcionamento de cada item do menu do programa.

3.1.1. Menus de parâmetros do LDmicro

O menu de parametrização do software LDmicro está dividido em grupos de funcionalidade, como Arquivo, Editar, Configurações, Instruções, Simular, Compilar e Ajuda. A seguir será descrito cada um destes itens.

3.2 MENU ARQUIVO

Descrição do menu Arquivo.

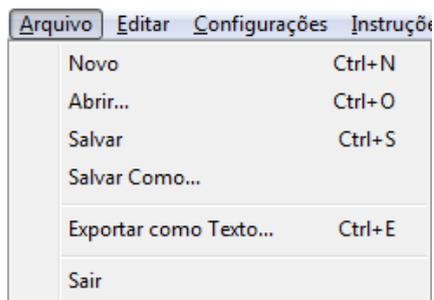


Figura 4 - Descrição do Menu Arquivo

Fonte: O Autor, 2017

Novo: Inicia um novo arquivo para realizar uma nova programação.

Abrir: Abre um arquivo já existente.

Salvar: Salva o arquivo que está em execução no momento.

Salvar Como: Salva uma cópia do arquivo que está em execução no momento.

Exportar como Texto: Salva o arquivo de programação em formato txt, como pode ser observado na figura 5.

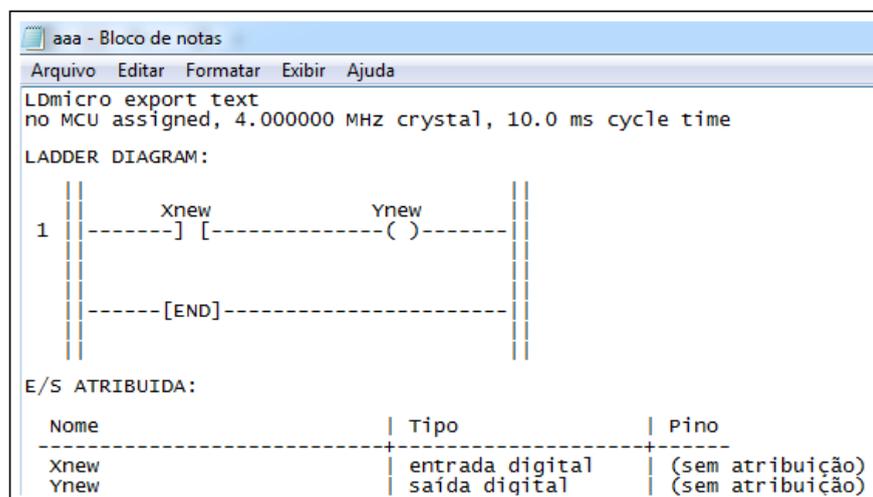


Figura 5 - Exemplo de arquivo Exportado como Texto
 Fonte: O Autor, 2017

Sair: Sai do programa LDmicro

3.2.1. Menu Editar

Descrição do menu Editar.

Editar		Configurações	Instruções	Simular	Compilar	Aju
Desfazer						Ctrl+Z
Refazer						Ctrl+Y
Inserir Linha (Rung) Antes						Shift+6
Inserir Linha (Rung) Depois						Shift+V
Mover Linha Seleccionada (Rung) Acima						Shift+Up
Mover Linha Seleccionada(Rung) Abaixo						Shift+Down
Apagar Elemento Seleccionado						Del
Apagar Linha (Rung)						Shift+Del

Figura 6 - Descrição do Menu Editar
 Fonte: O Autor, 2017

Desfazer: Desfaz a última ação.

Refazer: Refaz a última função desfeita.

Inserir Linha (Rung) Antes: Insere uma linha de programação antes da posição que o cursor se encontra.

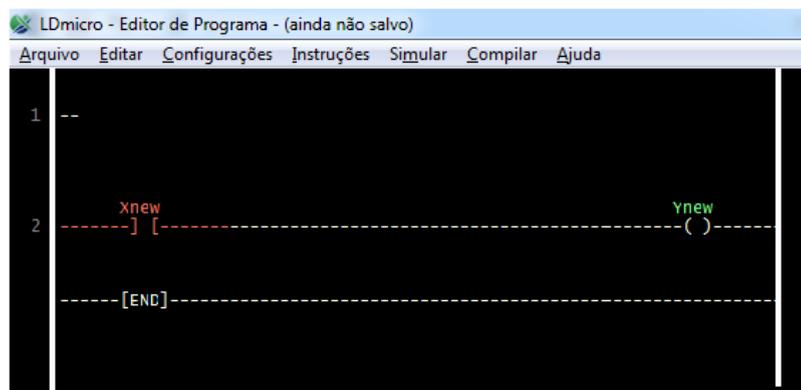


Figura 7 - Exemplo de linha inseria antes do cursor
 Fonte: O Autor, 2017

Inserir Linha (Rung) Depois: Insere linha de programação depois onde o cursor se encontra.

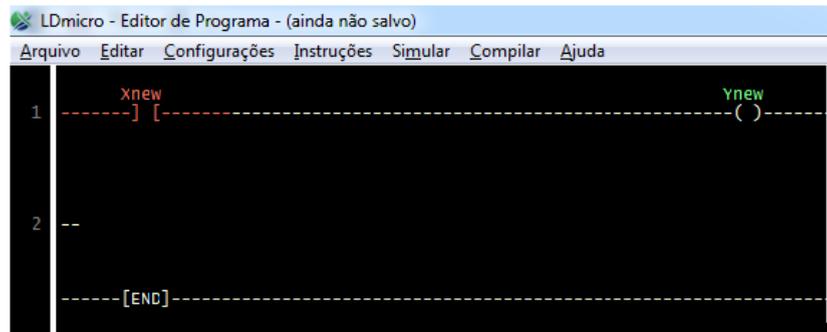


Figura 8 - Exemplo de linha inseria depois do cursor

Fonte: O Autor, 2017

Mover Linha Selecionada (Rung) Acima: Desloca a linha atual para uma posição superior.

Mover Linha Selecionada (Rung) Abaixo: Desloca a linha atual para uma posição abaixo.

Apaga Elemento Selecionado: Deleta item selecionado

Apaga Linha (Rung): Remove linha vazia. Obs. se o programa tiver uma linha vazia não irá compilar.

3.2.1.1 Menu Configurações

Descrição menu Configurações.

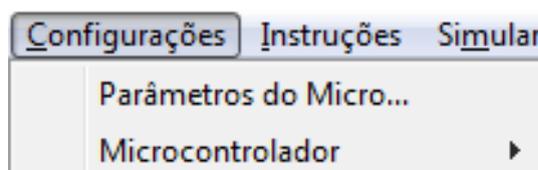


Figura 9 - Menu de Configurações

Fonte: O Autor, 2017

Parâmetros do Micro...: Determina os parâmetros de utilização do Microcontrolador.

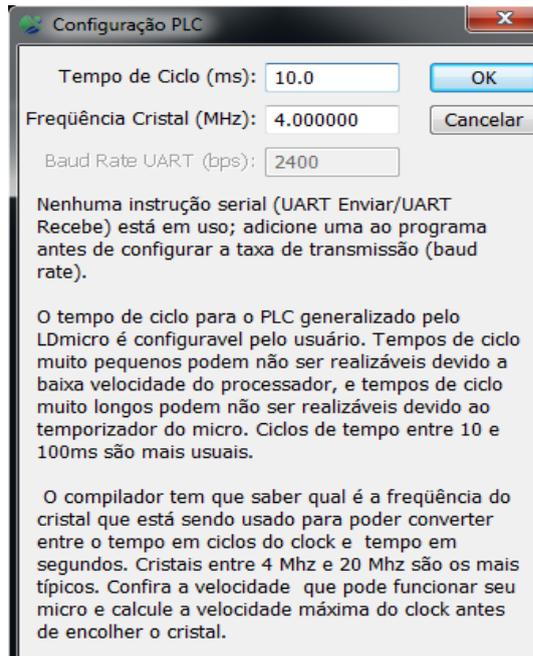


Figura 10 - Configuração do Microcontrolador

Fonte: O Autor, 2017

Tempo de Ciclo (ms): Termina o tempo de ciclo que máquina a ser utilizado. “Recomendado usar 10.0”.

Frequência Cristal (MHz): Frequência usado no cristal da placa do CLP.

Baud Rate USART (bps): Define a taxa de comunicação da porta serial.

Microcontrolador:

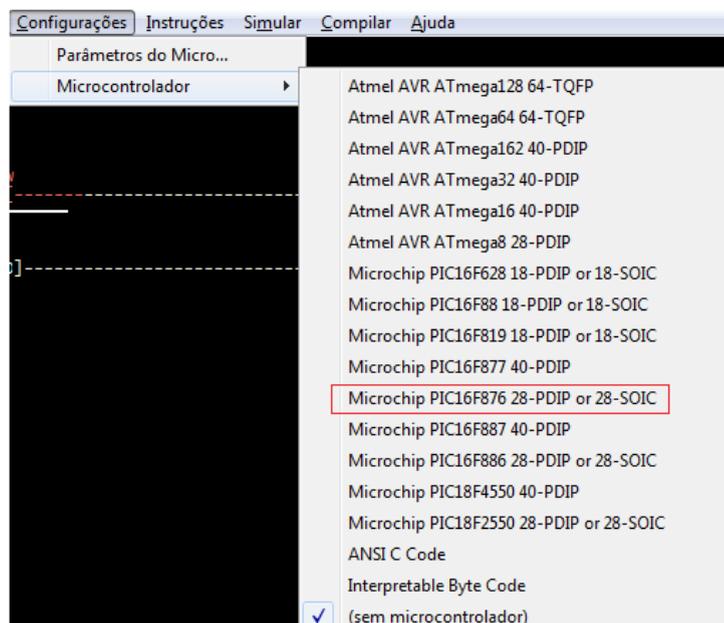


Figura 11 - Menu de escolha do Microcontrolador

Fonte: O Autor, 2017

Microcontrolador Atmel: Escolha do Microcontrolador da fabricante Atmel conforme suas devidas características.

Microcontrolador Microchip16F: Microcontroladores da fabricante Microchip.

Destaque para o PIC16F876 que é o Microcontrolador utilizado neste projeto.

Microcontrolador Microchip18F: A implementação para Microcontrolador PIC18F não é original do programa LDmicro, foi adicionado pelo autor do presente trabalho para futuras implementações nesta família de microcontroladores, portanto, PIC18F ainda não é funcional.

ANSI C Code: Permite salvar o arquivo no formato ANSI C sem definição de qualquer Microcontrolador como pode ser visto na Figura 12.

```

1  LDmicro0.1
2  MICRO=ANSI C Code
3  CYCLE=10000
4  CRYSTAL=4000000
5  BAUD=2400
6
7  IO LIST
8      Xnew at 0
9      Ynew at 0
10 END
11
12 PROGRAM
13 RUNG
14     PLACEHOLDER
15 END
16 RUNG
17     CONTACTS Xnew 0
18     COIL Ynew 0 0 0
19 END

```

Figura 12 - Arquivo salvo em ANSI C

Fonte: O Autor, 2017

Interpretable Byte Code: Sem identificação da funcionalidade

(Sem Microcontrolador): Utilizado para fazer simulação sem definição de Microcontrolador.

3.2.1.2 Menu Instruções

O menu instruções é extenso e, portanto, será abordado por grupo de componentes.

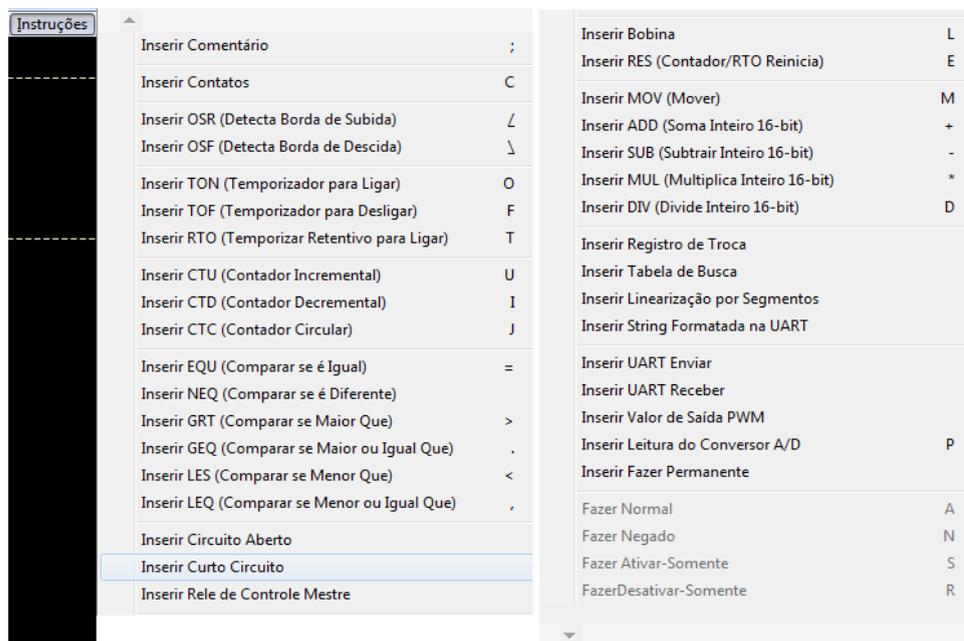


Figura 13 - Menu Instruções

Fonte: O Autor, 2017

a) CONTATO, NORMALMENTE ABERTO

Xname Rname Yname
 ----] [---- ----] [---- ----] [----

Tecla de atalho: C

Pode ser aplicado à: Pinos de entrada, Pinos de saída e Relés internos

Se o sinal da instrução (contato) é falso, a saída de sinal também será falsa.

Interpretado como uma chave com conexões em contato normalmente aberto, que quando acionado fecha o circuito.



Figura 14 - Seleção de tipos de contatos NA
 Fonte: O Autor, 2017

b) CONTATO, NORMALMENTE FECHADO

Xname Rname Yname
 ----]/[---- ----]/[---- ----]/[----

Tecla de atalho: C

Pode ser aplicado à: Pinos de entrada, Pinos de saída, Relés internos. Se o sinal de entrada for falso, a saída é verdadeira. Se o sinal de entrada for verdadeiro, a saída é falsa. Entenda como uma chave com conexões em contato normalmente fechado, que quando é acionada irá abrir o circuito, interrompendo a passagem de sinal.

Esta instrução é similar a anterior, com a diferença da opção Negado ativada.



Figura 15 - Seleção de tipos de contatos NF
 Fonte: O Autor, 2017

c) BOBINA (COIL), NORMAL

Rname Yname
 ----()---- ----()----

Tecla de atalho: L

Aplica-se à: Relé interno e pinos de saída, se o sinal vindo das instruções for verdadeiro, então a saída em questão irá ser ativada. Se o sinal for falso (circuito aberto), a saída será desativada. Esta instrução sempre estará mais à direita possível no diagrama ladder. Podem ser inseridas em paralelo, mas não em série.

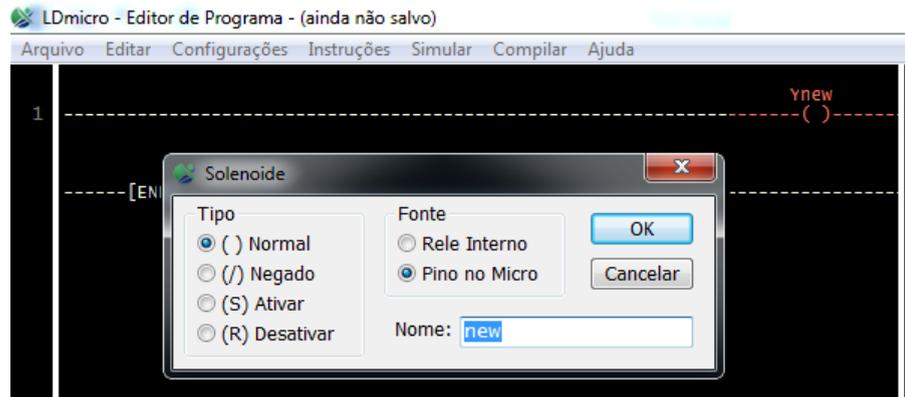


Figura 16 - Seleção de bobina de saída Normal

Fonte: O Autor

d) BOBINA (COIL), NEGATED

Rname Yname

----(/)---- ----(/)----

Tecla de atalho: L

Aplica-se à: Relé interno e pinos de saída, se o sinal que chega à instrução é verdadeiro, então a saída em questão é desativada. Caso o sinal de entrada seja falso (circuito aberto), a saída será ativada. Similar à instrução anterior com o valor invertido.

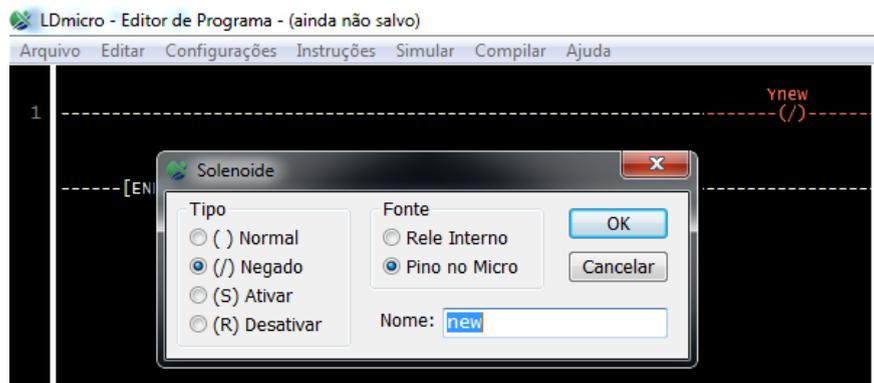


Figura 17 - Seleção de bobina de saída Negado

Fonte: O Autor

e) BOBINA (COIL), SET-ONLY

Rname	Yname
----(S)----	----(S)----

Tecla de atalho: L

Aplica-se à: Relé interno e pinos de saída, com esta opção, estabelece-se que a saída em questão irá ter seu estado modificado para ATIVADO quando a entrada de sinal for verdadeira, e não desativará quando o sinal de entrada for falso.

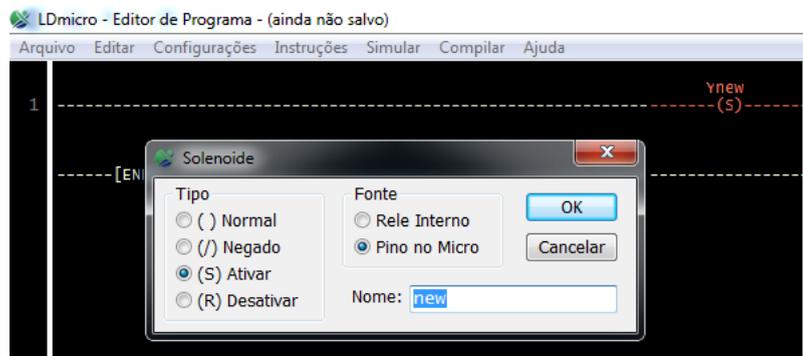


Figura 18 - Seleção de bobina de saída SET

Fonte: O Autor

f) BOBINA (COIL), RESET-ONLY

Rname	Yname
----(R)----	----(R)----

Se o sinal que chega a instrução for verdadeiro, então o relé interno ou pino de saída será desativado. Caso contrário, o pino de saída ou relé interno não terá seu estado modificado. Esta instrução somente poderá mudar o estado do pino de LIGADO para DESLIGADO, e geralmente isso é utilizado em combinação com uma ação BOBINA SET-ONLY.

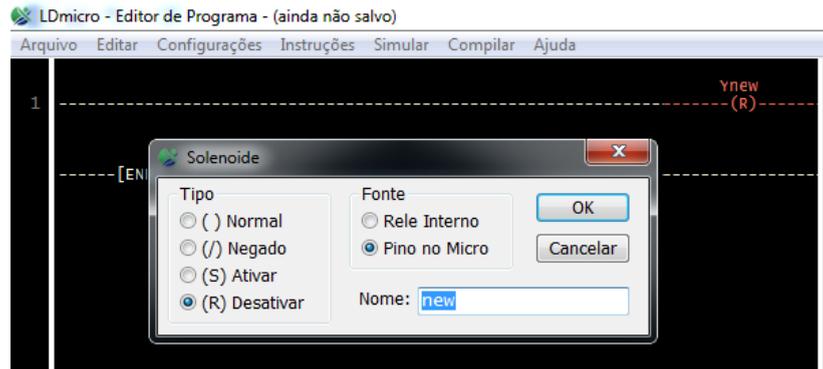


Figura 19 - Seleção de bobina de saída RESET

Fonte: O Autor

g) TEMPORIZADOR PARA LIGAR (TON)

Tdon
-[TON 100.0 ms]-

Quando o sinal que vai para a entrada instrução comutar de DESLIGADO para LIGADO, a saída permanecerá falsa por N segundos antes de ser ativado.

Quando o sinal de entrada mudar de LIGADO para DESLIGADO, a saída irá para DESLIGADO imediatamente. O temporizador é reiniciado sempre que a entrada estiver em nível DESLIGADO. O tempo (N) é configurável.

No exemplo da figura a seguir a função *TDelay* irá iniciar a contagem do tempo assim que receber sinal de *XEntrada* e contará 100 mili-segundos e após ligara a saída *YSaida*.

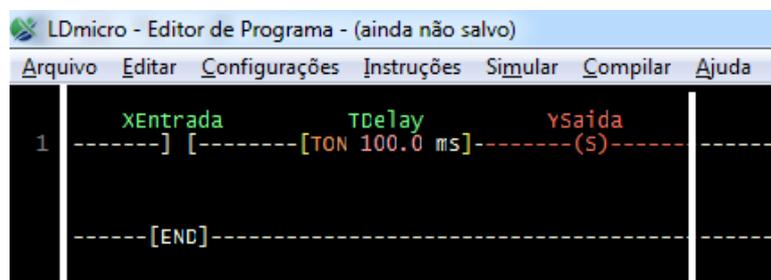


Figura 20 - Utilização do temporizador TON

Fonte: O Autor

h) TEMPORIZADOR PARA DESLIGAR (TOF)

Tdoff
-[TOF 1.000s]-

Quando o sinal vindo da instrução muda de LIGADO para DESLIGADO, o sinal de saída ainda permanecerá ativado por N segundos antes de desativar. Quando o sinal vindo da instrução mudar de DESLIGADO para LIGADO, a saída irá mudar para LIGADO imediatamente. O temporizador é reiniciado sempre que a entrada mudar para nível DESLIGADO. A entrada precisará estar desativada pelo tempo determinado para que a saída mude para estado FALSO. O tempo (N) é configurável.

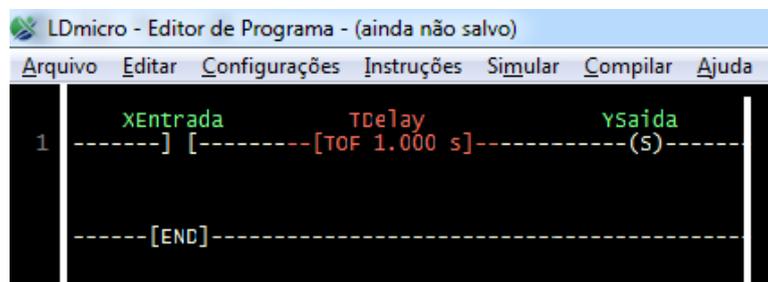


Figura 21 - Utilização do temporizador TOF

Fonte: O Autor

i) TEMPORIZADOR RETENTIVO PARA LIGAR (RTO)

Trto
-[RTO 1.000 s]-

Esta instrução está associada ao tempo de duração do pulso da entrada. Se a entrada permanecer ligada por mais de N segundos, a saída será ativada. Em outro caso, a saída permanecerá falsa. A entrada precisa permanecer acionada por, no mínimo N segundos para ativar a saída, que uma vez ativada, assim o permanecerá até que a variável 'Trto' em questão seja reiniciada, ou através de uma instrução manual de reset.

É possível manipular o estado da variável contadora de tempo através de instruções de manipulação de memória, como a instrução MOV ou RESET.

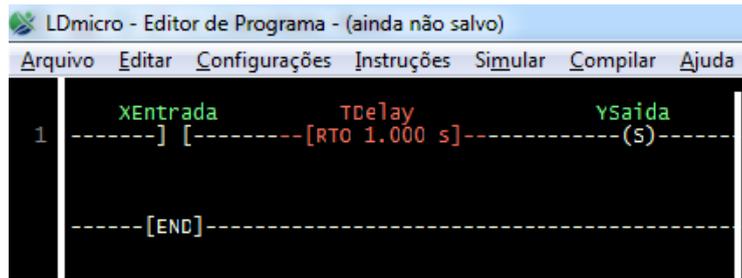


Figura 22 - Utilização do temporizador RTO

Fonte: O Autor

j) RESET (RES)



Esta instrução reinicia um timer ou contador. Retardos do tipo TON e TOF são automaticamente reiniciados quando as suas entradas mudarem de estado, mas o temporizador retentivo não. O temporizador RTO, contadores CTU e CTD não são reiniciados automaticamente, e devem ser reiniciados manualmente usando a instrução RES. Quando a entrada é verdadeira, o contador ou timer associado é reiniciado. Esta instrução necessita ser a instrução mais à direita na programação ladder.

Na imagem a seguir é possível observar uma aplicação de exemplo de utilização da função RES. A Entrada *XEntrada_Reset* permite um sinal em *TDelay* (RES), reiniciando o temporizador *RTO*.

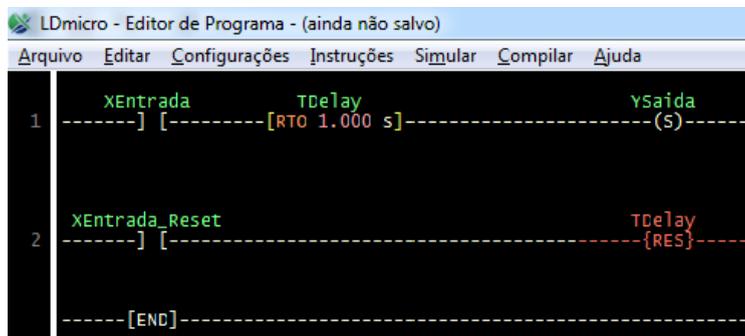


Figura 23 - Utilização da função RESET

Fonte: O Autor

k) BORDA DE SUBIDA (OSR)

--[OSR_/]--

Esta instrução produz, por padrão, saída DESLIGADA. Se o sinal de entrada mudar durante o processo de DESLIGADO para LIGADO a saída será ativada. Isso gera um pulso de um ciclo na saída, e pode ser usado para disparar eventos baseados na borda de subida de um sinal.

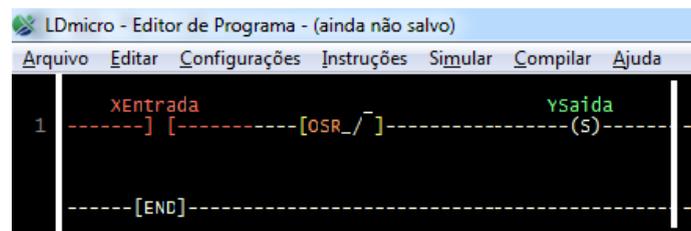


Figura 24 - Utilização da função Borda de Subida
Fonte: O Autor

l) BORDA DE DESCIDA (OSF)

--[OSF _]--

Esta instrução produz, por padrão, saída DESLIGADA. Se o sinal de entrada mudar durante o processo de LIGADO para DESLIGADO a saída será ativada. Isso gera um pulso de um ciclo na saída, e pode ser usado para disparar eventos baseados na borda de descida de um sinal.

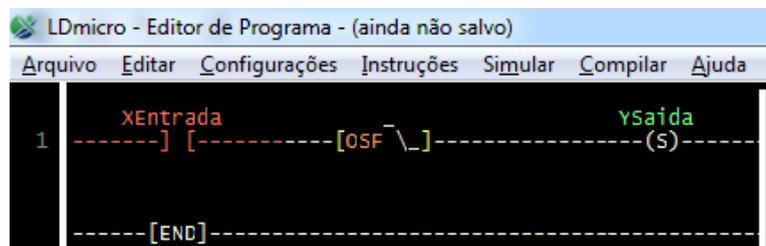


Figura 25 - Utilização da função Borda de Descida
Fonte: O Autor

m) CIRCUITO FECHADO, CIRCUITO ABERTO (SHORT CIRCUIT, OPEN CIRCUIT)

-----+-----+----- -----+ +-----

A condição de saída de um CIRCUITO FECHADO é sempre igual a sua condição de entrada, e o estado do sinal da saída de um CIRCUITO ABERTO é sempre desligado. Isso pode ser útil para depurar o programa em situações de análise e teste.

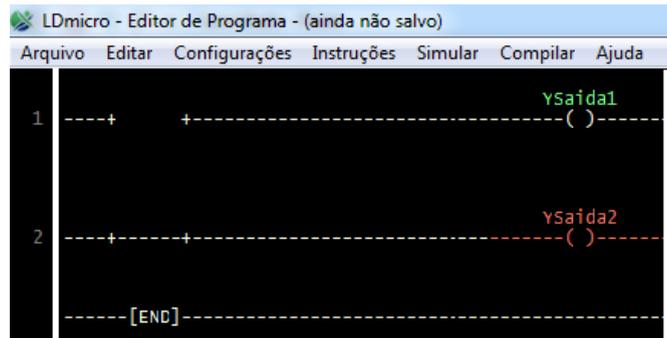


Figura 26 - Utilização Circuito Fechado e Circuito Aberto
Fonte: O Autor

n) RELÉ PRINCIPAL (MASTER CONTROL RELAY)

-{MASTER RLY}-

Por padrão, todos os degraus têm condição de entrada LIGADA. Este relé principal realiza controle de entrada, e pode desativar estas entradas. Dentro do ciclo do programa, podem ser adicionadas instruções para ativação e desativação parciais do MASTER RLY, sendo isso bastante útil para depuração.

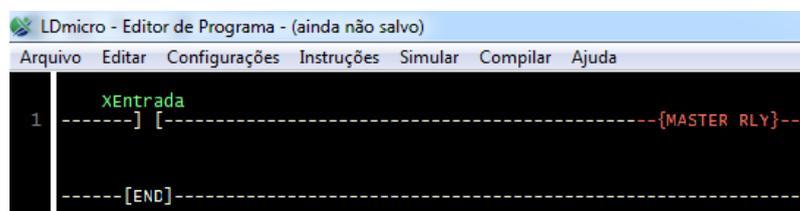


Figura 27 - Utilização Relé Principal
Fonte: O Autor

o) MOVE

{destvar := } {Tret := }

-{ 123 MOV}- {- srcvar MOV}-

Quando o sinal de entrada desta instrução for LIGADO, a variável de destino será carregada com o valor da variável de origem ou da constante. Se o sinal de entrada estiver DESLIGADO, nada acontece com a variável. Permite utilizar a instrução MOVE com qualquer variável, incluindo contadores e temporizadores. Esta instrução precisa estar mais à direita na tela de programação.

Um exemplo de utilização pode-se observar na figura 28 a seguir, a instrução MOV irá mover o valor 10 para o contador CTU.

Quando a variável *XEntrada* for acionada permitirá que o valor 10 seja movido para o *CCont* iniciando a contagem de 10 até 15. Quando o contador atingir o valor máximo irá ligar a variável *YSaida*.

Obs: A instrução MOV não aceita no campo *Destino* uma variável com mais de 8 caracteres.

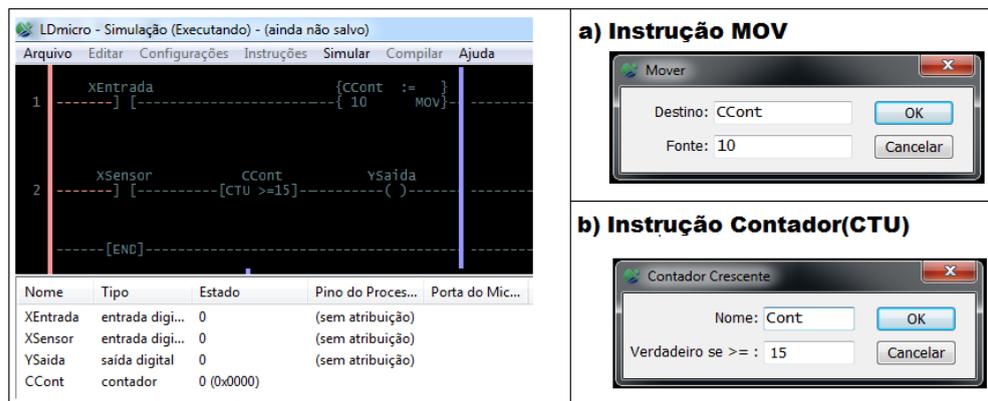


Figura 28 - Utilização instrução MOV

Fonte: O Autor

p) OPERAÇÃO ARITMÉTICA (ARITHMETIC OPERATION)

{ADD kay :=} {SUB Ccnt :=}
 -{ 'a' + 10 }- -{ Ccnt - 10 }-

{MUL dest :=} {DIV dv := }
 -{ var * -990 }- -{ dv / -10000}-

Quando o sinal de entrada desta instrução é verdadeiro, a variável de destino é carregada com o resultado da expressão. Isso pode ser realizado também em variáveis de contagem e temporizadores. Estas instruções aritméticas utilizam um processamento aritmético de números 16 bits com sinal. Cada ciclo de processamento desencadeia uma operação aritmética, caso o sinal de entrada esteja verdadeiro. Se usar esta instrução para incrementar ou decrementar uma variável, e isso estiver associado a um sinal de entrada (tipo um botão), use a instrução de borda de subida ou descida para que o incremento ou decremento ocorra somente em uma operação por pulso de entrada. A divisão é truncada, e não arredondada. $8/3$ resulta em 2.

Através da Figura 29 é possível observar um exemplo de utilização da função ADD, onde a soma dos valores “2+2” é atribuída ao contador, possibilitando a contagem a partir do resultado “4”. Desta mesma forma pode ser utilizado as outras operações aritméticas, SUB, MUL e DIV.

Nome	Tipo	Estado	Pino do Proces...	Porta do Mic...
XEntrada	entrada digi...	1	(sem atribuição)	
XSensor	entrada digi...	0	(sem atribuição)	
YSaida	saída digital	0	(sem atribuição)	
CCont	contador	4 (0x0004)		

a) Instrução ADD

Somar

Destino:

esta acionado := :

+ :

b) Instrução Contador(CTU)

Contador Crescente

Nome:

Verdadeiro se >= :

Figura 29 - Utilização instrução ADD

Fonte: O Autor

q) COMPARAÇÃO (COMPARE)

[var ==] [var >] [1 >=]

-[var2]- -[1]- -[Ton]-

[var /=] [-4 <] [1 <=]

-[var2]- -[vartwo]- -[Cup]-

Se o sinal de entrada para esta instrução é DESLIGADO, então a saída também é DESLIGADA. Se o sinal de entrada for LIGADO, então a saída será LIGADA se a instrução de comparação for verdadeira. Esta instrução pode ser usada para comparação (igual, maior que, maior e igual, diferente, menor que, menor e igual) entre variáveis, ou para comparar uma variável com uma constante numérica de 16 bits.

```

1  Ysaida {ADD var :=}
   -----] / [----- {var + 1}
2  [var > 100] Ysaida
   ----- [----- ( )
   ----- [END]

```

Figura 30 - Utilização instrução de Comparação

Fonte: O Autor

r) (CONTADOR) COUNTER

```

Cname Cname
--[CTU >=5]-- --[CTD >=5]--

```

Um contador incremental (CTU, count up) ou decrementa (CTD, count down) é associado à detecção de borda de uma determinada condição de entrada. O sinal de saída desta instrução é verdadeiro caso o valor do contador tenha atingido o valor limite.

Pode-se utilizar instrução de incremento (CTU) e decremento (CTD) com variáveis de mesmo nome, e a instrução RES para reiniciar o valor do mesmo.

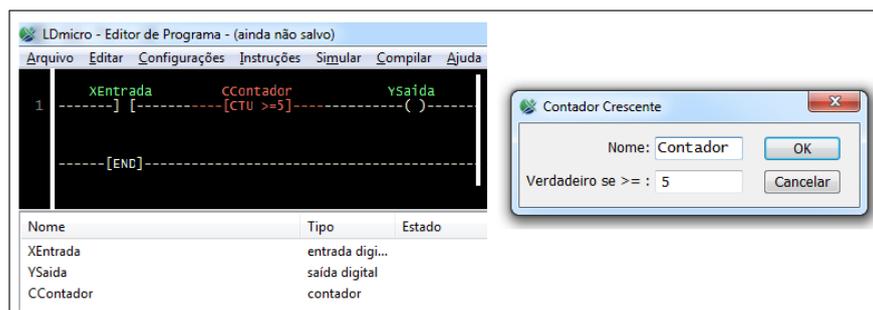


Figura 31 - Utilização instrução de Contador (CTU)

Fonte: O Autor

s) CONTADOR CIRCULAR (CIRCULAR COUNTER)

Cname
-->{CTC 0:7}--

Um contador circular trabalha como um contador normal, exceto pelo fato de, após atingir o limite, ele reinicia voltando ao valor inicial. Por exemplo, o contador acima, se incrementado, contará 0, 1, 2, 3, 4, 5, 6, 7, 0, 1, 2, 3, 4, 5, 6, 7, 0, 1, 2, ...

Isso pode ser muito útil se combinado com estruturas condicionais na variável 'Cname', podendo ser criado o efeito de sequenciador. CTC conta na borda de subida do sinal de entrada, e esta instrução deve ser a mais à direita da programação ladder.

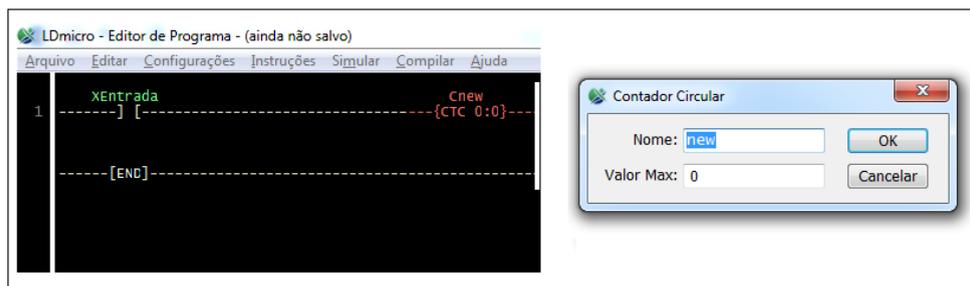


Figura 32 - Utilização instrução de Contador Circular

Fonte: O Autor

t) REGISTRO DE TROCA (SHIFT REGISTER)

{SHIFT REG }
- { reg0..3 } -

Um SHIFT REGISTER (Registrador de Troca) é associado a um grupo de variáveis. Por exemplo, este registrador pode ser associado com as variáveis 'reg0', 'reg1', 'reg2' e 'reg3'.

A cada pulso (borda de subida) na entrada desta instrução, ocorre o deslocamento (reg3 \square reg2, reg2 \square reg1, reg1 \square reg0, e reg0 permanece inalterado).

Observe que esta instrução consome uma área significativa de memória, e deve ser usada com cuidado.

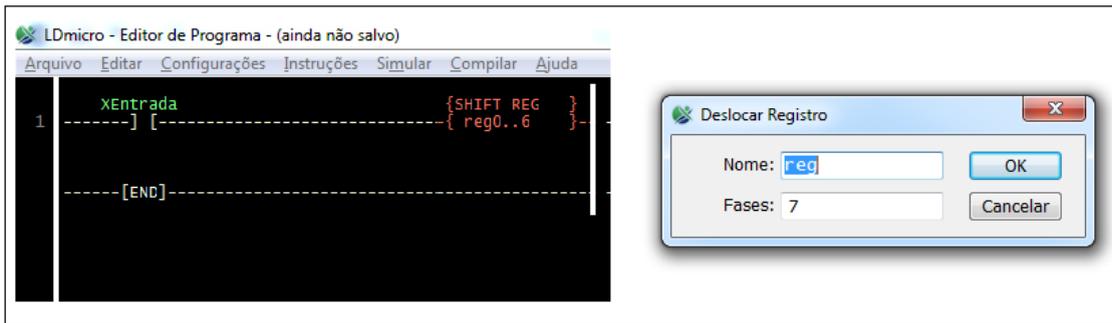


Figura 33 - Utilização instrução de Shift Register

Fonte: O Autor

u) TABELA DE BUSCA (LOOK-UP TABLE)

```
{dest := }
-{ LUT[i] }-
```

A tabela de busca (Look-up Table) é um conjunto de dados ordenados. Quando a condição de entrada for LIGADA, a variável de destino 'dest' será carregada com o valor da variável de origem na ocorrência 'i'. O índice (i) inicia em zero.

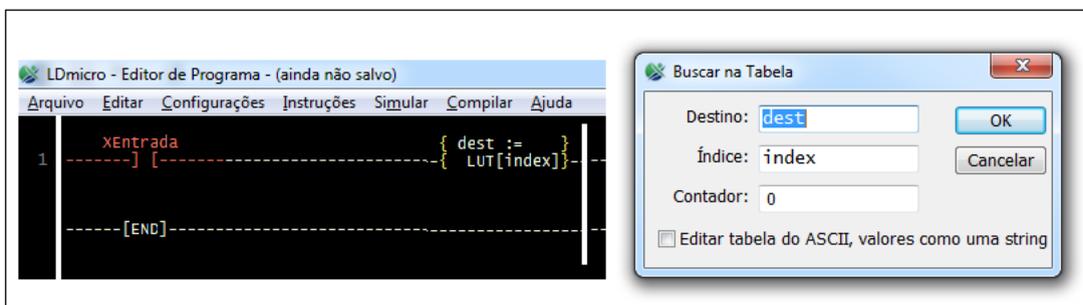


Figura 34 - Utilização instrução de Look-up Table

Fonte: O Autor

v) LINEARIZAÇÃO POR SEGMENTO (PIECEWISE LINEAR TABLE)

```
{yvar := }
-{ PWL[xvar] }-
```

Esta é a melhor forma de aproximar uma complicada função de transformação não linear. Pode ser útil, por exemplo, se tiver que associar os valores lidos a uma curva de calibração baseado no valor de entrada de um sensor.

Pode-se entrar com dados na tabela associando com um valor linear desejado. O valor será atribuído a variável xvar, e isso define o valor da variável yvar.

Se atribuir a variável xvar um valor entre dois pontos lançados na tabela, o valor de yvar será computado em função da proximidade destes dois valores.

Os pontos precisam ser especificados em ordem crescente em xvar. Em alguns casos, por questões matemáticas e de limitações do processamento aritmético de 16 bits, o LDmicro poderá gerar mensagens de erro relativas a conversão. Neste caso, adicione mais dados à tabela.

Ex: Isso irá produzir um erro, se lançado estes dois pontos:

$$\begin{aligned}(x_0, y_0) &= (0, 0) \\ (x_1, y_1) &= (300, 300)\end{aligned}$$

Que pode ser corrigido desta forma:

$$\begin{aligned}(x_0, y_0) &= (0, 0) \\ (x_1, y_1) &= (150, 150) \\ (x_2, y_2) &= (300, 300)\end{aligned}$$

Em casos extremos pode ser necessário usar mais de cinco ou seis pontos.

Adicionando mais pontos o código fica maior e mais lento para ser executado.

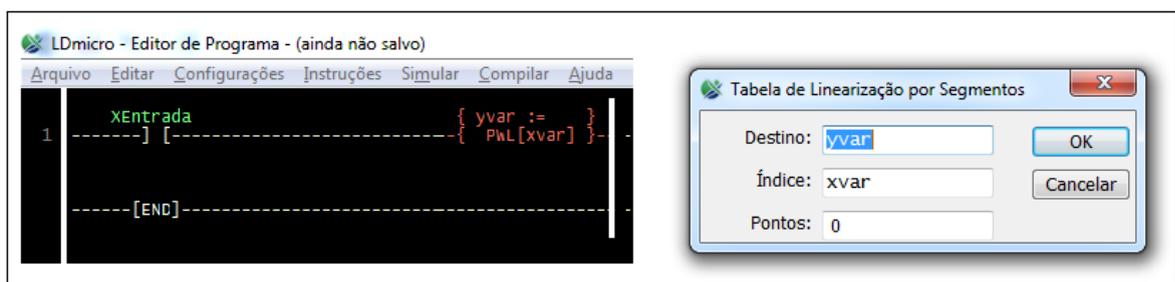


Figura 35 - Utilização instrução de Linearização Por Segmento

Fonte: O Autor

w) LEITURA DE CONVERSOR A/D (A/D CONVERTER READ)

Aname
-->{READ ADC}--

LDmicro pode gerar código para usar o conversor A/D embutido em certos microcontroladores. Se a condição de entrada para esta instrução for LIGADA, então será obtida uma simples amostra do conversor A/D e isso é armazenado na variável chamada 'Aname'. Testa a variável e pode ser manipulada com operações genéricas, como comparação e atribuição.

Associa-se o pino à variável 'Axxx' da mesma forma que é associado um pino de entrada/saída digital, clicando duplamente na parte inferior da tela.

Para todos os microcontroladores suportados atualmente, Entradas de 0 Volts corresponde ao valor 0, e uma entrada na tensão máxima do sistema (Vdd) corresponde ao valor 1023 (AD de 10 bits). O software não irá permitir que associe pinos que não sejam entradas analógicas às mesmas.

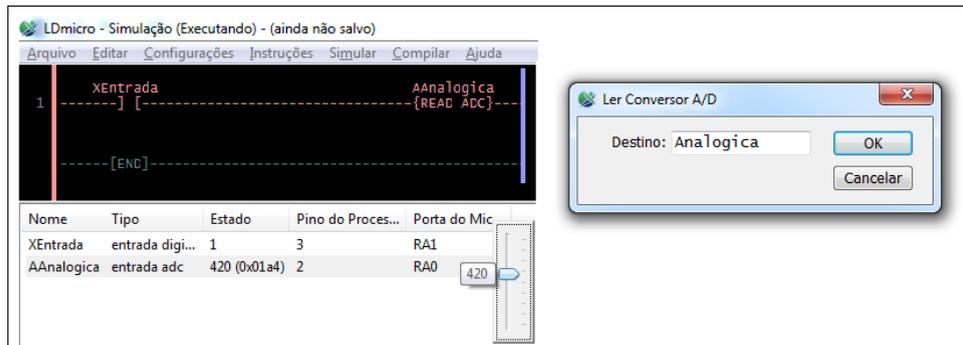


Figura 36 - Utilização Conversor A/D

Fonte: O Autor

x) VALOR DE SAÍDA PWM (SET PWM DUTY CYCLE)

duty_cycle
 -{PWM 32.8 kHz}-

LDmicro pode gerar código para usar o periférico PWM embutido em certos microcontroladores. Se a condição de entrada desta instrução for verdadeira, então o duty cycle do periférico PWMW é definido com o valor da variável duty_cycle. O duty cycle precisa ser um número entre 0 e 100, onde 0 corresponde a “sempre desligado”, e 100 corresponde a “sempre ligado”. (O procedimento que os periféricos PWM utilizam, fará a conversão proporcional do número 0 a 100 em valores binários correspondentes para os períodos de clock necessários).

Deve-se especificar a frequência, em Hz. No entanto, a frequência especificada pode não ser a exata a ser utilizada, dependendo dos divisores internos do microcontrolador e da frequência de clock utilizada por este. Caso for definido um valor fora da faixa permitida, o LDmicro irá alertá-lo.

Obs. Na atual versão do hardware utilizado neste trabalho, não está contemplado um periférico PWM, ficando como sugestão de melhorias para as próximas versões.

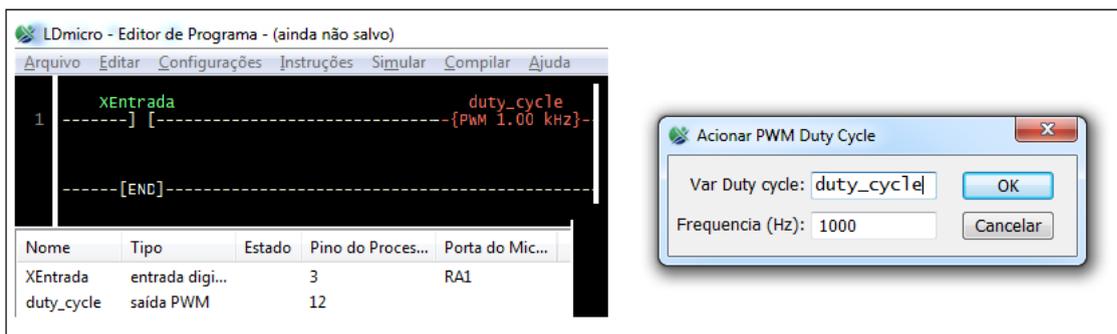


Figura 37 - Utilização instrução PWM

Fonte: O Autor

y) FAZER PERMANENTE (MAKE PERSISTENT)

```

saved_var
--{PERSIST}--
  
```

Quando a instrução de entrada é LIGADA, isso fará com que determinada variável seja salva na EEPROM. Isso significa que o valor permanecerá quando o sistema for desativado (desconectado da energia elétrica).

Não é necessário declarar o local onde a informação será gravada, isso ocorrerá de forma automática, e a variável será automaticamente carregada quando o sistema for reiniciado.

Atenção para não abusar deste recurso, pois gravando muito frequentemente na EEPROM, este recurso pode ser danificado, pois muitos sistemas garantem um limite de 100000 gravações somente.

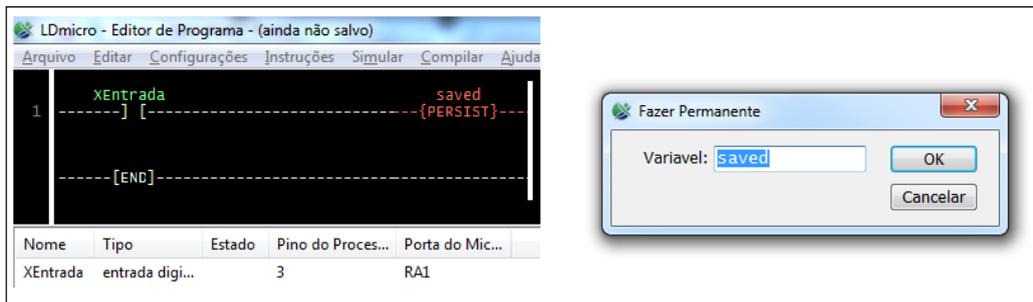


Figura 38 - Utilização instrução Salva EEPROM

Fonte: O Autor

z) UART RECEBE SERIAL (UART RECEIVE)

```
var
--{UART RECV}--
```

LDmicro pode gerar código para usar a UART embutida em certos microcontroladores. Configurar a taxa de transferência (baud rate) usando a opção "Parâmetros do Micro...". Certas taxas podem não ser aceitas em certas faixas de frequência de clock. O LDmicro irá alertá-lo, nestes casos.

Se a condição de entrada desta instrução for DESLIGADA, então nada irá ocorrer. Caso contrário a instrução tentará receber um simples caractere da UART. Se nenhum caractere for lido, então a condição de saída será FALSO. Se um caractere for lido, então o mesmo será armazenado na variável 'var' e a condição de saída da instrução será LIGADO (por um único ciclo de execução).

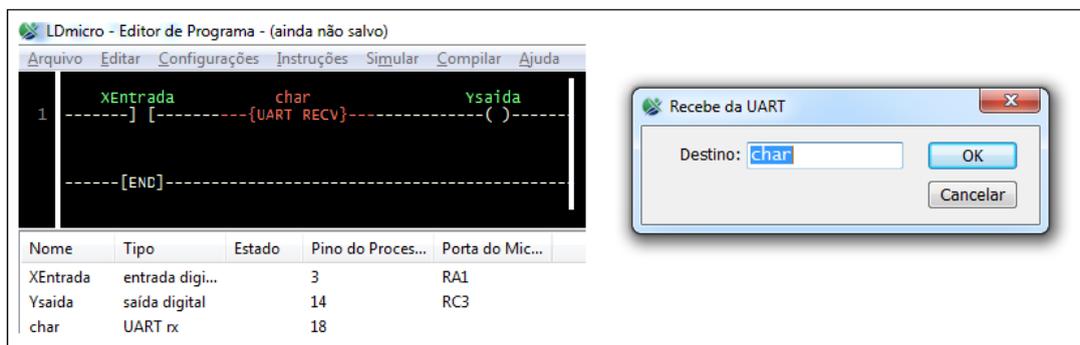


Figura 39 - Utilização instrução Recebe UART Serial

Fonte: O Autor

aa)UART ENVIA SERIAL (UART SEND)

```

var
--{UART SEND}--

```

Se a entrada desta instrução estiver DESLIGADA, então nada acontecerá. Se a condição estiver LIGADA, a instrução irá enviar um simples caractere na UART. O valor ASCII do caractere a ser enviado deve ter sido previamente armazenado na variável 'var'. A opção de saída do degrau será LIGADA enquanto a transmissão estiver ocorrendo, e DESLIGADO quando o processo terminar.

O caractere leva algum tempo para ser transmitido. Deve ser verificado a condição de saída para se certificar que o processo de transmissão do primeiro caractere terminou antes de enviar um segundo.

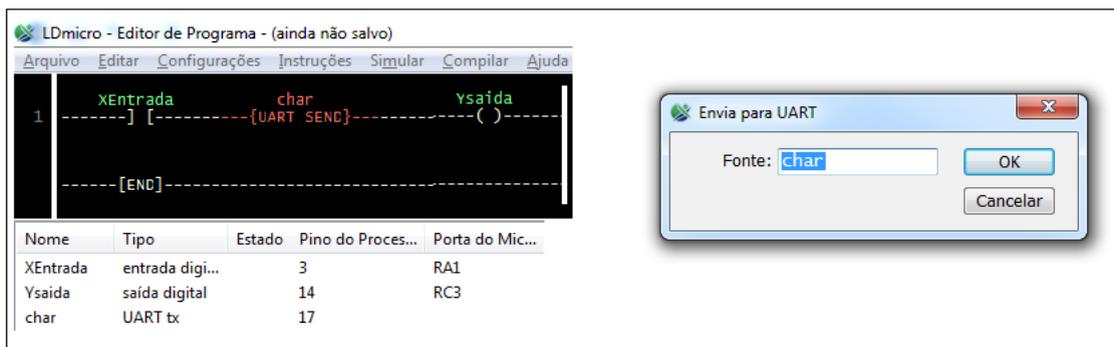


Figura 40 - Utilização instrução Envia UART Serial

Fonte: O Autor

bb)STRING FORMATADA NA UART (FORMATTED STRING OVER UART)

```

var
-{"Pressure: \3\r\n"}-

```

Quando esta instrução é utilizada (com sinal de entrada LIGADO), ela começa a enviar uma sequência de caracteres (STRING) através da porta serial. Na string, onde houver a sequência \3, será substituída pela variável em questão. O \3 significa que o valor irá tomar exatos 3 caracteres; Por exemplo, se a variável 'var' estiver comportando o valor 35, então a string exata que será enviada seria: 'Pressure: 35\r\n' (observe o espaço extra antes do número). Se, por outro lado, a variável 'var' poderá

assumir número de mais dígitos, como por exemplo o número 1432, então deverá ser mudada esta definição. Será necessário usar o '\4'.

Se a variável em questão poderá assumir valores negativos, use '\-3d' ou '\-4d'. O dígito aparecerá somente quando os valores forem negativos. Para valores positivos, o sinal será substituído por um espaço.

Se múltiplas strings formatadas forem acionadas simultaneamente (ou se uma for acionada antes de outra terminar), ou se estas instruções estiverem sendo usadas simultaneamente com instruções UART TX, o resultado poderá ser indefinido. Isso é permitido para que possa ser enviada uma simples string de dados pelo programa.

Use o caractere especial '\\' para exibir uma contrabarra. Abaixo a lista de caracteres especiais que podem ser usados:

- \r -- carriage return (retorno de carro. Volta para primeira coluna)
- \n -- newline (Nova linha.)
- \f -- formfeed (Alimenta formulário. Geralmente usado como clear)
- \b -- backspace (Volta um caractere)
- \xAB -- character with ASCII value 0xAB (hex) (exibir carac. especial)

A saída desta instrução é verdadeira enquanto ela estiver transmitindo dados. Esta instrução consome muita memória de programa, e deve ser usada com cuidado. Na implementação atual do LDMicro, esta função não é muito otimizada consumindo muito recurso de processamento.

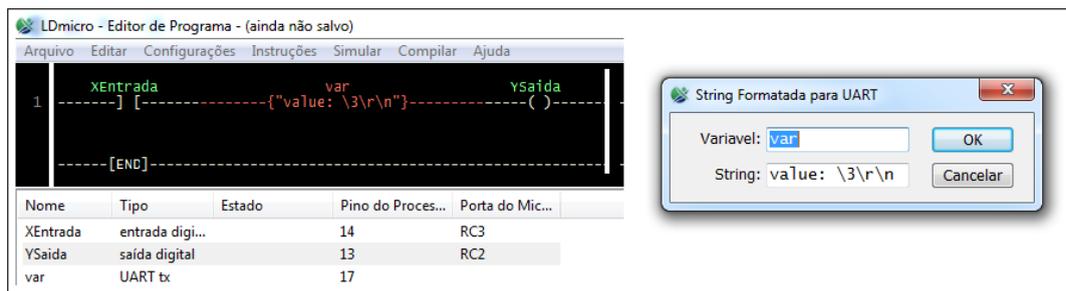


Figura 41 - Utilização instrução Envia String

Fonte: O Autor

cc) Observações ao usar recursos matemáticos

Lembre que o LDmicro realiza somente instruções matemáticas com inteiros de 16 bits. Isso significa que o resultado final de qualquer cálculo que for realizado deverá ser um valor inteiro entre -32768 e 327667. Isso também quer dizer que valores intermediários de cálculos também devem permanecer nesta faixa. Por exemplo, supondo o cálculo a seguir $y = (1/x)*1200$, onde x é um valor entre 1 e 20. Então y poderá valer 1200 a 60, armazenados em uma variável inteira de 16 bits. Para isso ser calculado, teoricamente há duas maneiras de se escrever o código:

```

||      {DIV temp :=}      ||
||-----{ 1 / x    }-----||
||                          ||
||      {MUL y := }      ||
||-----{ temp * 1200}-----||
||                          ||

```

Ou pode realizar a divisão em um passo único:

```

||      {DIV y :=}      ||
||-----{ 1200 / x }-----||

```

Matematicamente, as duas são equivalentes. Mas se testá-las, é possível observar que a primeira sempre dará um resultado incorreto $y = 0$. Isso é porque a variável 'temp' somente armazenará valores inteiros, e o resultado do primeiro degrau $(1 / x)$ geralmente será um número menor que 1 (para $x = 3$, tempo será 0,3333...) e isso não é possível de se armazenar em uma variável inteira.

Caso ter problemas nos resultados de equações, deve-se verificar os valores intermediários, observando se nenhum valor gerado irá resultar em dados não armazenáveis em variáveis de 16 bits com sinal.

Em uma multiplicação de variável por uma fração, deve-se usar respectivamente as instruções de multiplicação e divisão. Por exemplo, para multiplicar y por $1.8 * x$, usa-se: $y = (9/5)*x$ (lembrando que $9/5 = 1.8$). E, no código ladder, a multiplicação ocorrer antes da divisão.

```

||      {MUL temp :=}      ||

```

```

||-----{ x * 9 }-----||
||                               ||
||   {DIV y :=}   ||
||-----{ temp / 5 }-----||

```

Isso funcionará para qualquer $x < (32767 / 9)$, ou $x < 3640$. Para valores maiores de x , a variável 'temp' irá sair de seu limite.

dd) Estilos de Codificação

É permitido múltiplas bobinas em paralelo em um simples degrau. Isso significa que é possível fazer algo como a seguir:

```

||   Xa       Ya   || |
||-----] [-----(-)-----||
||                               ||
||   Xb       Yb   ||
||-----] [-----+-----(-)-----||
||           |       ||
||           |       Yc  ||
||           +-----(-)-----||
||                               ||

```

No lugar disso:

```

||   Xa       Ya   ||
1 ||-----] [-----(-)-----||
||                               ||
||                               ||
||                               ||
||                               ||
||   Xb       Yb   ||
2 ||-----] [-----(-)-----||
||                               ||
||                               ||

```

```

||                ||
||                ||
||   Xb          Yc   ||
3 ||-----] [----- ( )-----||
||                ||

```

Isso torna possível escrever programas em um único degrau gigante. Na prática isso pode não ser uma boa ideia, porque os degraus ficarão mais complexos e mais difíceis de serem editados, lidos e interpretados. Mesmo assim, isso pode ser uma boa ideia para grupos de mesma relevância lógica, tratando-as como um simples degrau.

1. Menu Simular

O menu Simular Figura 42, permite que o software LDmicro faça uma simulação do funcionamento da programação antes que seja compilado.

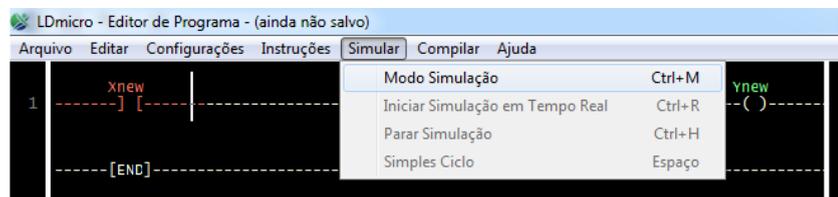


Figura 42 - Menu Simular

Fonte: O Autor

Para iniciar deve clicar em “*Modo de Simulação*” e posteriormente clicar em “*Iniciar Simulação em Tempo Real*”, conforme indicado na Figura 43.

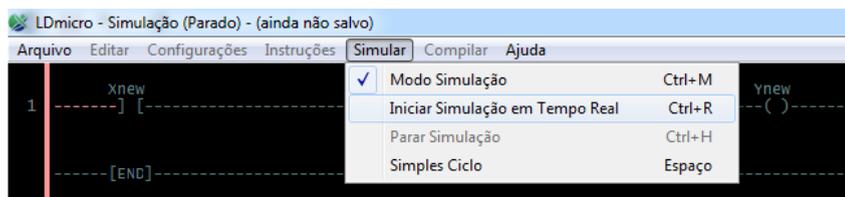


Figura 43 - Menu Simular

Fonte: O Autor

2. Menu Compilar

O menu **Compilar** irá gerar o código Hex conforme os parâmetros, configurações e programação realizada anteriormente.

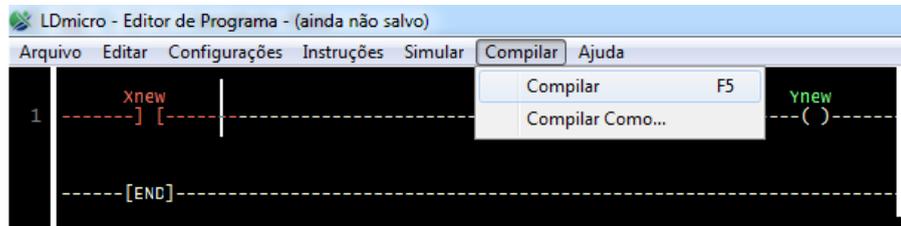


Figura 44 - Menu Compilar

Fonte: O Autor

Assim que for clicado em *“Compilar”* ou utilizar o atalho F5, o software irá pedir para indicar um nome e local a ser salvo o arquivo .hex que posteriormente poderá ser gravado no Microcontrolador.

O botão *“Compilar Como”* fará uma cópia do programa que está sendo desenvolvido.

3. Menu Ajuda

O menu Ajuda traz o *“Manual”* com descrição em inglês do funcionamento das instruções do Software LDmicro que pode ser acionado através do atalho no teclado F1.

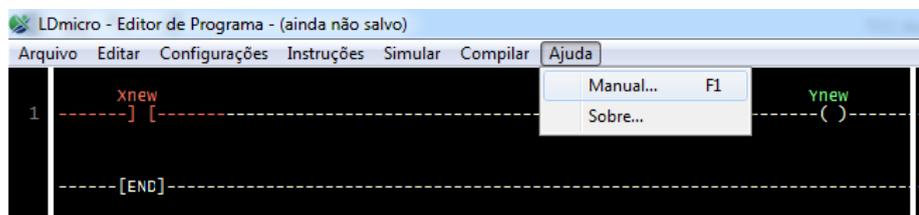


Figura 45 - Menu Ajuda

Fonte: O Autor

3.3 SOFTWARE DE GRAVAÇÃO

Para realizar a gravação do firmware desenvolvido no Microcontrolador, foi testado diversos softwares gratuitos encontrados na internet, pode-se citar o ICProg, TinybldWin, WinPic800. O Único que funcionou e atendeu as necessidades da aplica foi o TinybldWin, que é um programa fácil de utilizar e é possível utilizar para gravação do bootloader no Microcontrolador. A Figura 46 demonstra a interface do programa Tiny.

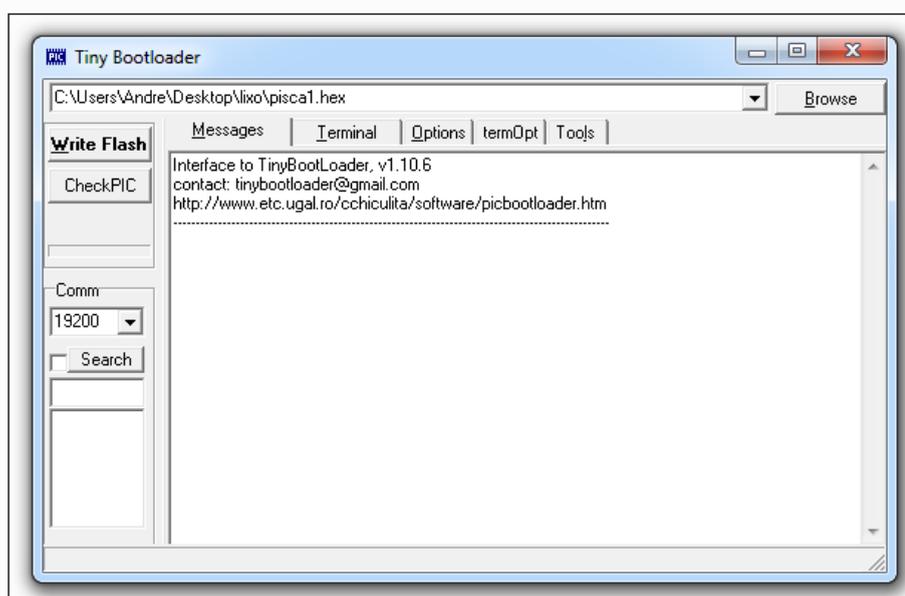


Figura 46 - Software de Programação Tiny

Fonte: O Autor

Para realizar a gravação do firmware no Microcontrolador deve proceder da seguinte maneira:

- a) Iniciar o programa Tiny
- b) Configurar a taxa de comunicação para 19200bps
- c) Clicar no botão "CheckPIC" e logo no **reset** no Microcontrolador ele irá reconhecer o PIC conectado via USB.
- d) Escolher o programa a ser gravado clicando no botão "*Browse*".
- e) Clicar no botão "*Write Flash*" e logo no **reset** no Microcontrolador
- f) Após este procedimento o Microcontrolador estará gravado e basta fazer a utilização desejada.

3.4 DESENVOLVIMENTO DO PROTÓTIPO

Para iniciar o desenvolvimento do protótipo primeiramente foi definido os requisitos do sistema e a partir disso decidido o microcontrolador. Utilizando o programa Proteus foi realizado o diagrama eletrônico e com o auxílio do professor orientador ajustes foram realizados. Posteriormente montado o diagrama em protoboard para realizar alguns testes reais conforme pode ser observado na Figura 47.

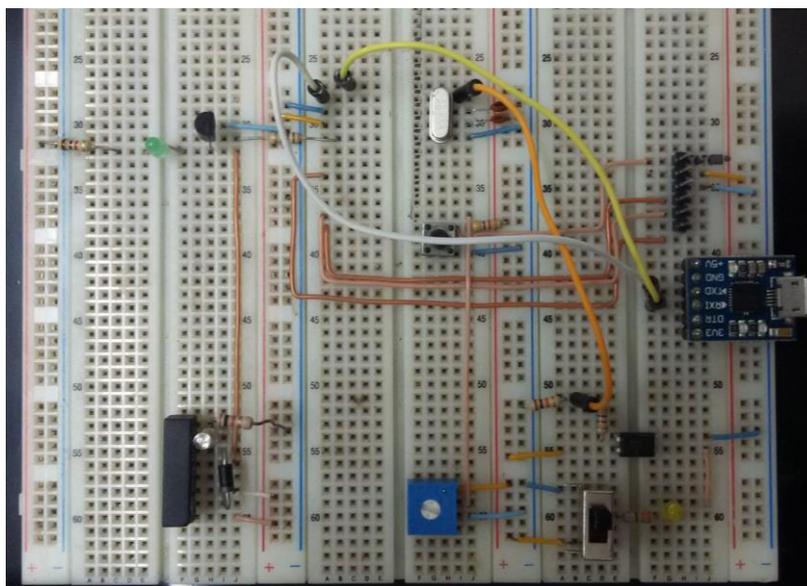


Figura 47 - Montagem do protótipo em protoboard

Fonte: O Autor

Após o desenvolvimento do diagrama eletrônico e layout da placa, teve-se a necessidade em fabricar uma placa protótipo para checar as conexões e validar ajustes mecânicos de posição dos componentes. Os fabricantes de PCI convencionais exigem a fabricação de no mínimo de 10 placas, tornando inviável financeiramente a prototipagem da placa.

Foi realizado diversas pesquisas na internet para encontrar uma forma de prototipagem mais barata. Uma empresa localizada através do site de vendas Mercado Livre fez a prototipagem de apenas uma peça no valor de R\$80,00. Infelizmente não foi possível fazer testes de validação de funcionalidade elétrica, pois como a placa eletrônica possui dupla face e o processo de fabricação desta empresa não contempla a furação metalizada, mas foi possível encontra um problema de layout da placa, o componente regulador de tensão LM7805 possui sua altura

instalado em aproximadamente 20mm conforme pode ser observado na Figura 48A, o grande problema foi identificado logo na instalação do driver USB Figura 48B onde não é possível instalar os dois componentes na mesma posição. Posteriormente foi corrigido o problema no layout da placa eletrônica.

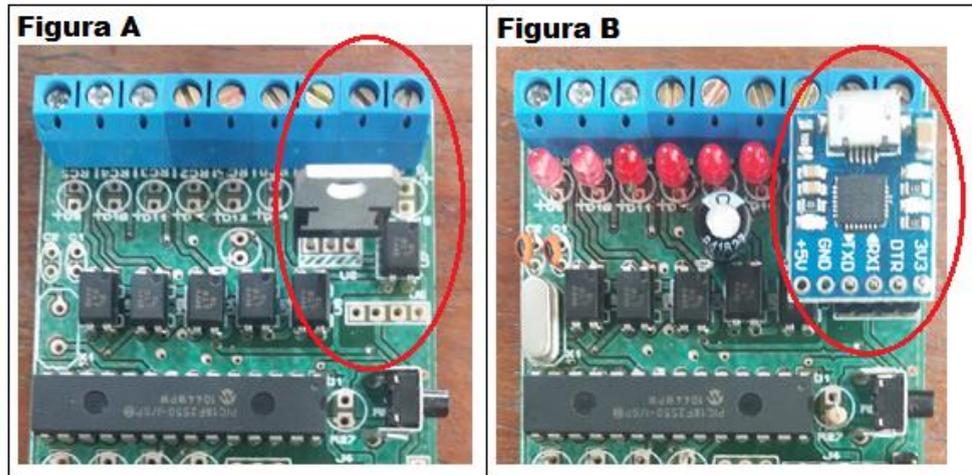


Figura 48 - Placa protótipo Versão 1.0

Fonte: O Autor

Uma dificuldade encontrada foi no processo de roteamento automático do software de desenvolvimento, observou-se que não estava realizando o roteamento completo e algumas trilhas estavam fazendo um percurso muito longo até chegar no destino, ajustes manuais teve que ser feito.

Uma nova versão da placa foi desenvolvida “Versão V1.1”, e todo o roteamento foi realizado manualmente. Este processo exigiu uma dedicação de tempo maior, mas a qualidade do roteamento ficou mais coerente.

A nova versão foi enviada para fabricação em uma empresa especializada de Nova Trento-SC. Foi confeccionado um total de 10 placas com os custos apresentados na tabela a seguir:

Tabela 1 - Custos placa V1.1

Quant.	Descrição	Preço unitário	Valor
10	FIBRA 1.6mm 1-OZ FACE DUPLA METALIZADO	R\$ 21,00	R\$ 210,00
1	Fotoplotagem	R\$ 85,00	R\$ 85,00
1	Frete e Impostos	R\$ 67,68	R\$ 67,68
	Total		R\$ 362,68

Fonte: O Autor

O resultado da placa confeccionada é possível ver na figura 49, em destaque a qualidade muito superior comparada com a placa da versão V1.0.

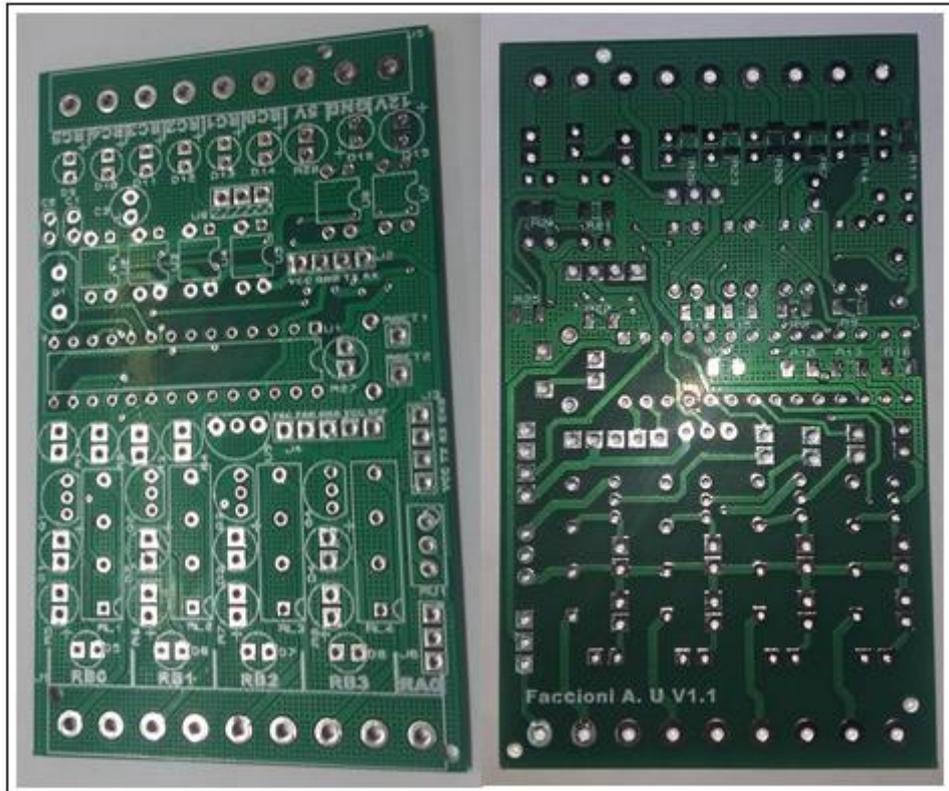


Figura 49 - Placa protótipo Versão 1.1

Fonte: O Autor

Na imagem a seguir é possível observar os componentes eletrônicos já inseridos e devidamente soldados na placa, inclusive os componentes SMD na face de baixo da PCI.

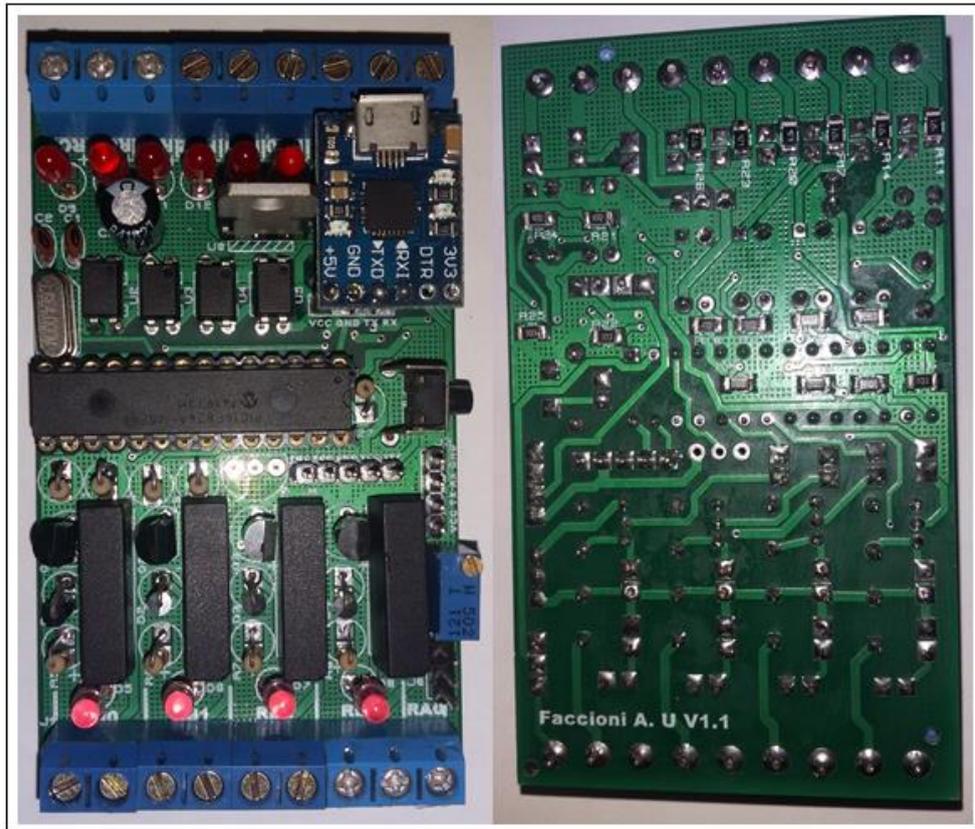


Figura 50 - Placa com Componentes
 Fonte: O Autor

3.4.1 Gabinete ABS

Para atender a aplicação, e gerar uma proteção extra para a placa eletrônica, foi decidido projetar a placa a ser inserida em um gabinete. Após diversas pesquisas optou-se por utilizar o gabinete da linha PATOLA, uma empresa especializada em fabricação de gabinetes modulares para o setor eletrônico, que atua no mercado desde 1975.

O gabinete escolhido (fundo DIN 075 - 18 terminais), possibilita a integração total da placa deste projeto, além de possíveis implementações futuras como a integração de um display alfanumérico.

Na figura a seguir é possível observar o desenho técnico do gabinete com suas devidas medidas.

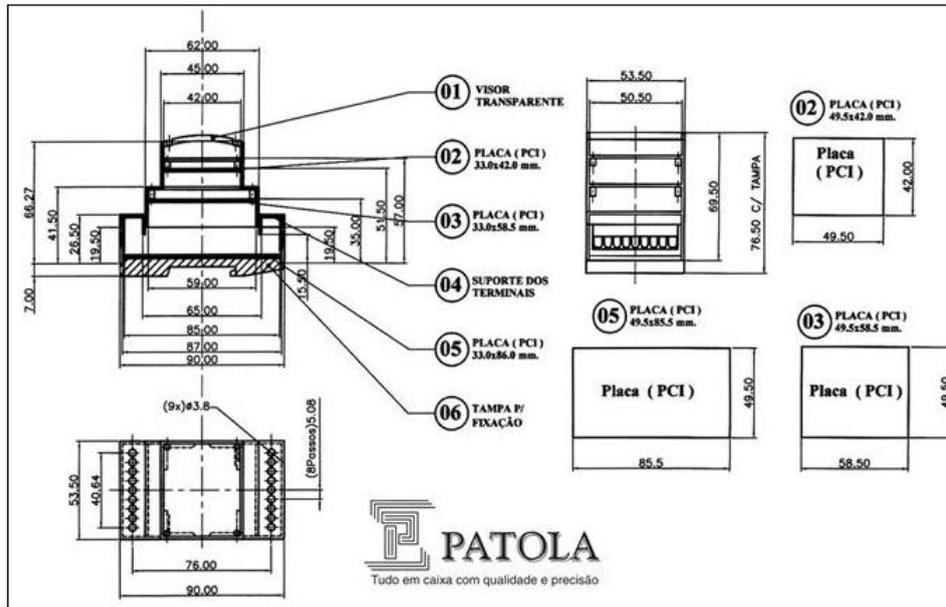


Figura 51 - Dimensões Gabinete PATOLA

Fonte: O Autor

Logo a seguir na figura 52 a placa eletrônica inserida perfeitamente dentro do gabinete e logo a cima em destaque a tampa em acrílico que possibilita futuramente implementar um display para facilitar a visualização por exemplo de sinais analógicos ou outras informações que possa ser implementada pelos usuários.

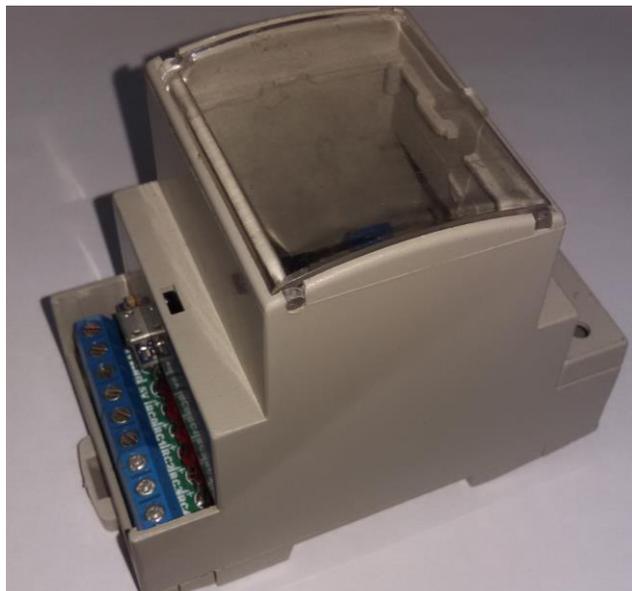


Figura 52 - Gabinete e placa eletrônica

Fonte: O Autor

3.4.2 Diagrama Eletrônico

O diagrama foi desenvolvido com auxílio do programa Proteus, realizado diversos testes no simulador validando assim o diagrama eletrônico. Nas imagens a seguir é possível observar o diagrama devidamente separado e identificado com suas funcionalidades:

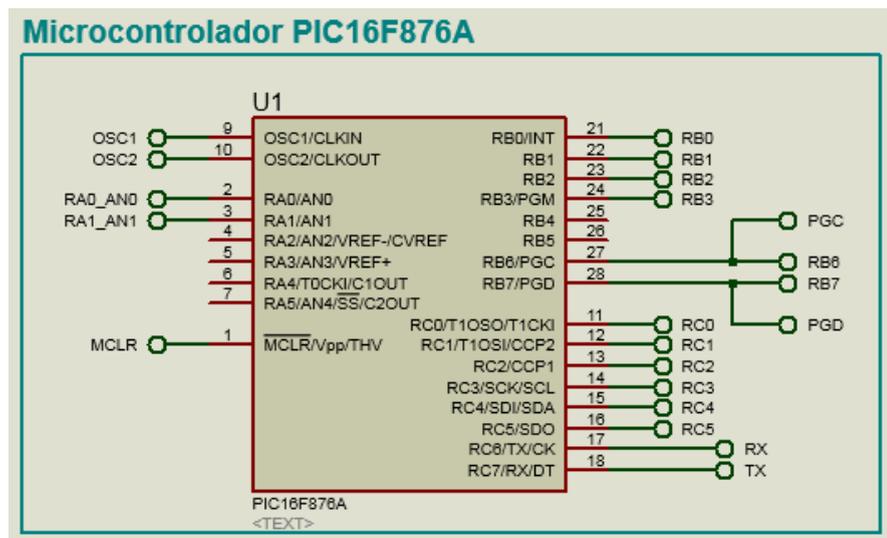


Figura 53 - Conexões microcontrolado PIC16f876A

Fonte: O Autor

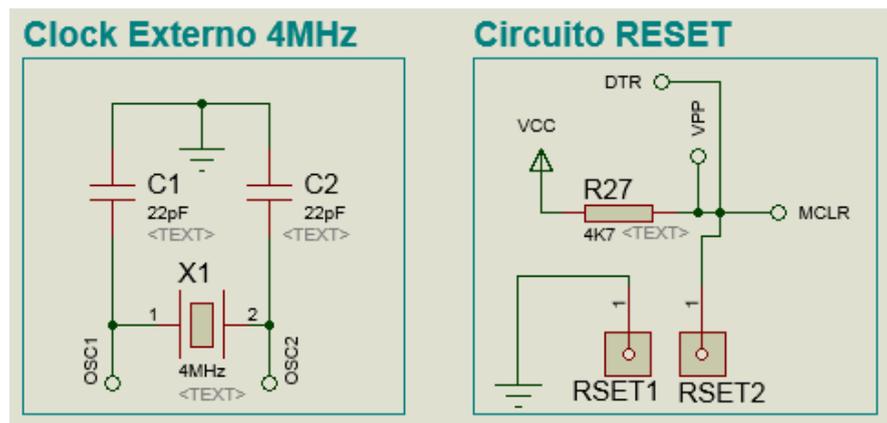


Figura 54 - Circuito de clock e circuito de reset

Fonte: O Autor

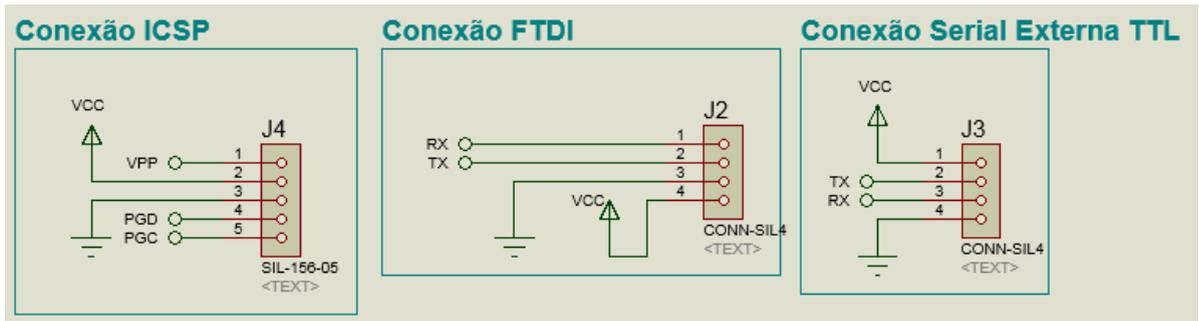


Figura 55 - Circuitos de comunicação

Fonte: O Autor

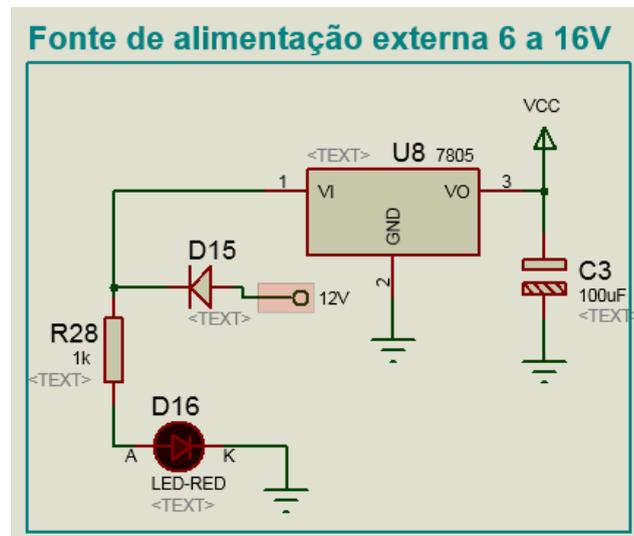


Figura 56 - Circuito da fonte de alimentação

Fonte: O Autor

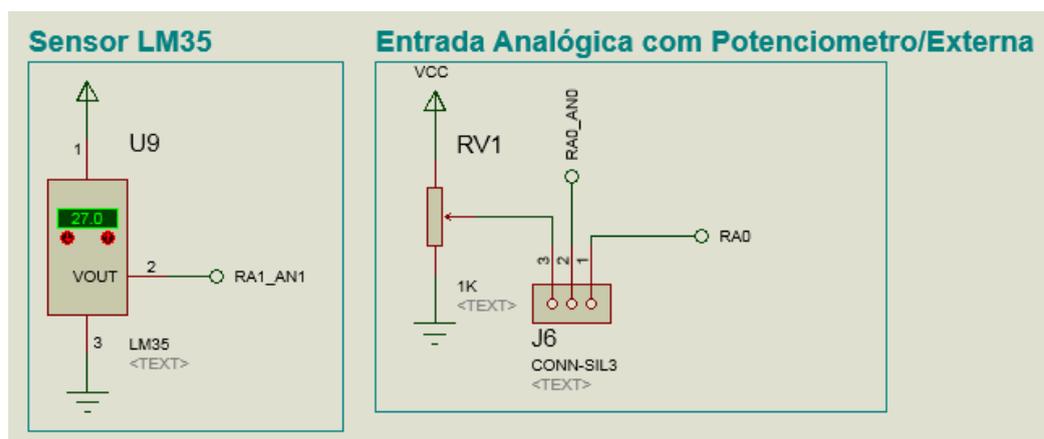


Figura 57 - Entrada de sinais analógicos

Fonte: O Autor

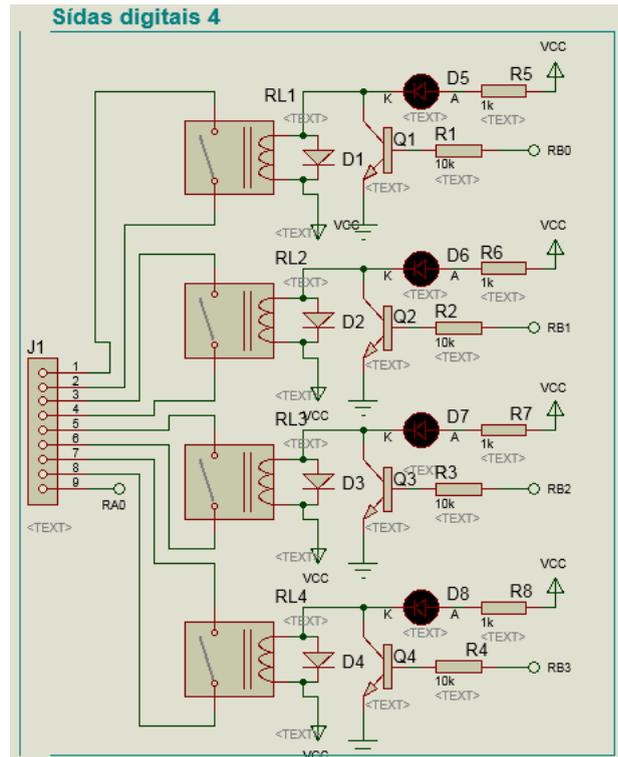


Figura 58 - Saídas digitais a relé

Fonte: O Autor

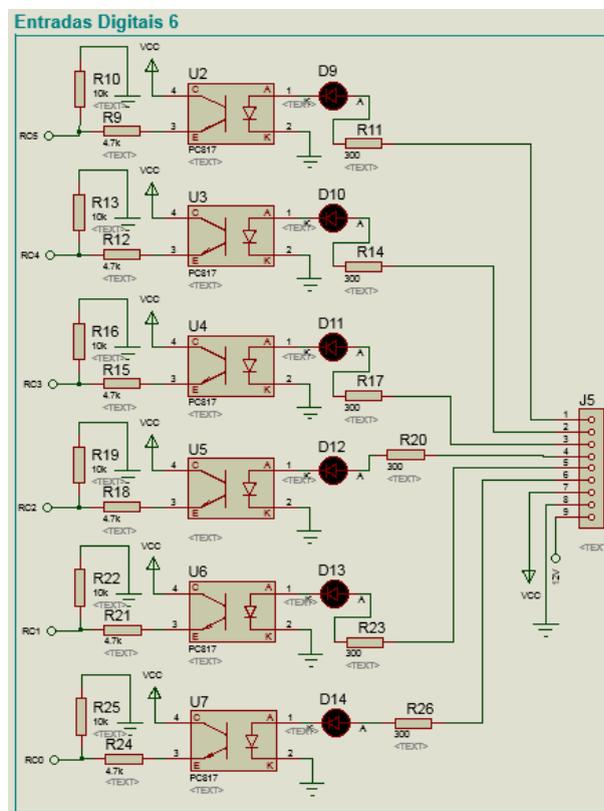


Figura 59 - Entradas digitais

Fonte: O Autor

3.4.3 Montagem PCI

Para facilitar a montagem da placa eletrônica, ela está dividida em circuitos, como por exemplo circuito de entrada, circuito de saída, clock etc. A figura 26 exibe o layout completo da PCI desenvolvido com o software Corel Draw para se obter uma melhor resolução da imagem.

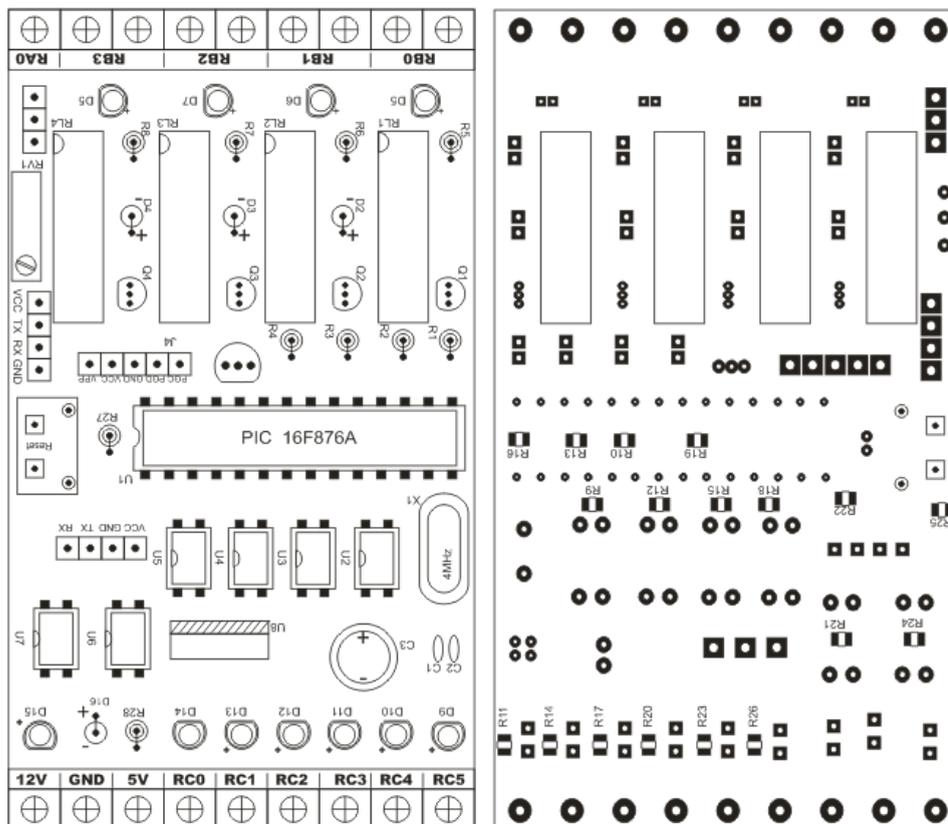


Figura 60 - Layout PCI

Fonte: O Autor

3.4.4 Circuito de Clock e Reset

O circuito de Clock é composto por apenas 3 componentes são eles;

Tabela 2 - Componentes circuito Clock

ID	Descrição	Quantidade	Preço Un.	Imagem
X1	Cristal 4 MHz Meia Caneca	1	R\$ 0,84	
C1, C2	Capacitor Cerâmico 22pf	2	R\$ 0,11	

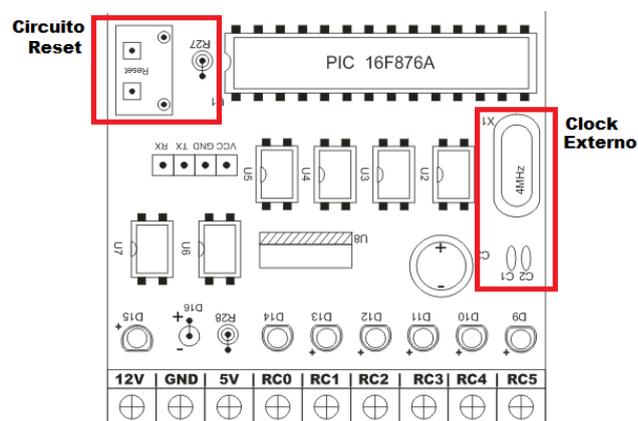
Fonte: O Autor

Logo a baixo é exposto o circuito de Reset com sua respectiva lista de componentes.

Tabela 3 - Componentes circuito Reset

ID	Descrição	Quantidade	Preço Un.	Imagem
Reset	Chave Táctil 6x6x7mm 90° 4T	1	R\$ 0,31	
R27	Resistor 4k7 1/8W Through Hole	1	R\$ 0,03	

Fonte: O Autor

**Figura 61 - Circuito Reset e Clock**

Fonte: O Autor

3.4.5 Circuito de Alimentação

O circuito de alimentação é constituído de uma tensão de entrada de até 12V logo a pos passando por um regulador de tensão 7805-U8, pois o circuito eletrônico opera com tensão de 5V. O diodo D16 serve como segurança para que não seja invertido a alimentação entre positivo e negativo.

Tabela 4 – Componentes circuitos de alimentação

ID	Descrição	Quantidade	Preço Un.	Imagem
U8	Regulador 7805 (TO-220) PACKAGE	1	R\$ 1,28	
C3	Capacitor Eletrolítico 100uf 25V	1	R\$ 0,16	
D15	Diodo 1n4007 Through Hole	1	R\$ 0,09	
R28	Resistor 1K 1/8W Through Hole	1	R\$ 0,03	
D16	LED 3mm Vermelho Through Hole	1	R\$ 0,10	

Fonte: O autor, 2017

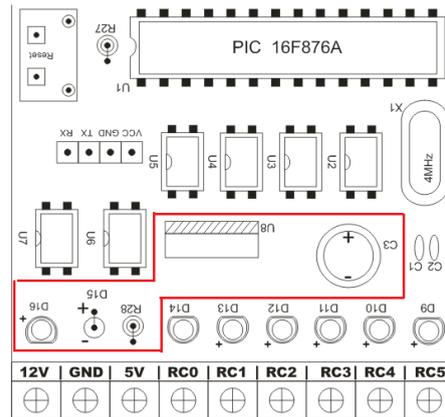


Figura 62 - Circuito de alimentação
Fonte: O Autor

3.4.6 Sensor de temperatura LM35

O sensor de temperatura LM35 está conectado diretamente na porta RA0 do Microcontrolador PIC, esta porta é uma entrada analógica permitindo realizar testes de programação com sinais analógicos sem a necessidade de implementar um sensor externo, serve apenas para fins didáticos.

Tabela 5 – Componente sensor de temperatura

ID	Descrição	Quantidade	Preço Un.	Imagem
U9	Sensor de temperatura LM35	1	R\$ 0,31	

Fonte: O autor, 2017

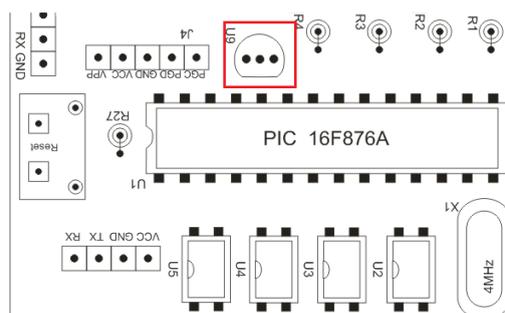


Figura 63 - Sensor de temperatura
Fonte: O Autor

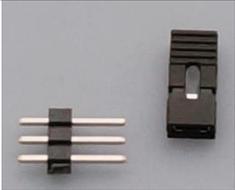
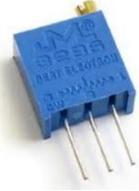
3.4.7 Entrada analógica e trimpot

A figura 64 apresenta a forma de utilização da entrada analógica, através da porta RA0 do Microcontrolador ela pode ser utilizada de 2 modos conforma a seguir.

Modo a: Através do jumper J6 selecionado conforme a figura 64-a, seleciona a entrada analógica através do trimpot multivoltas, este modo permite simular um sinal analógico direto a porta RA0 do Microcontrolador entre 0 a 10V.

Modo b: Se o jumper J6 estiver selecionado de acordo com a figura 64-b, permite usar a mesma porta RA0 do Microcontrolador como uma entrada externa de sinal analógico entre 0 a 10V.

Tabela 6 – Componentes entrada analógica

ID	Descrição	Quantidade	Preço Un.	Imagem
J6	Barra de Pinos 1x40 vias 11,2mm 180 graus	1	R\$ 0,08	
RV1	Trimpot 3296W 5K	1	R\$ 1,36	

Fonte: O autor, 2017

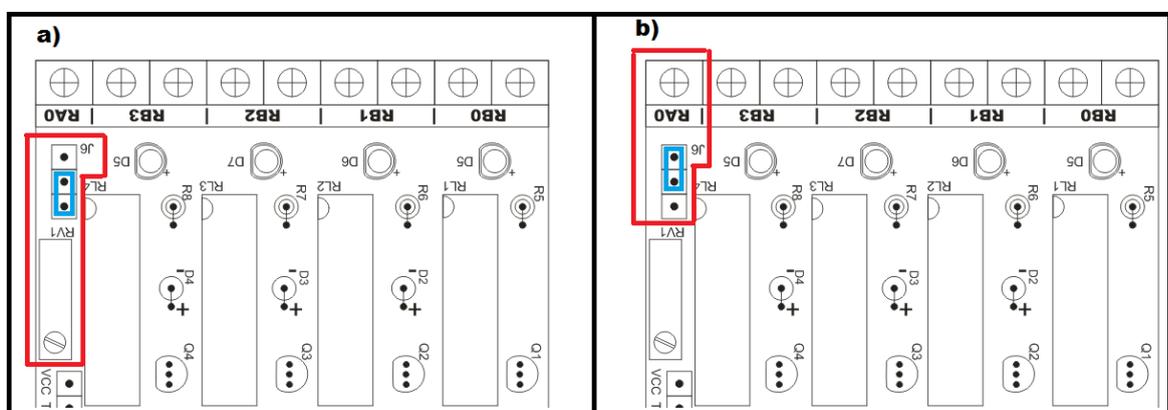


Figura 64 - Configuração de Entrada Analógica

Fonte: O Autor

3.4.8 Lista de Exercícios

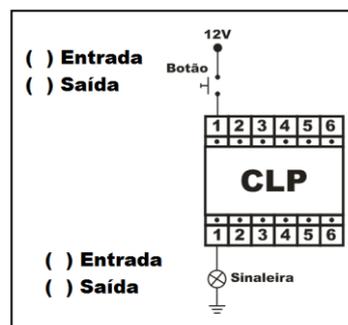
Para atender a proposta do presente trabalho, a baixo é apresentado algumas sugestões de exercícios e atividades a serem aplicados aos alunos. Outras atividades serão incluídas conforme o andamento da aplicação do trabalho.

Exercícios

1- O Controlador Lógico Programável (CLP) é um equipamento muito utilizado em indústrias. Qual a aplicação e função deste equipamento?

R: _____

2- Represente na figura a seguir quais são entradas e quais são saídas digitais do CLP.



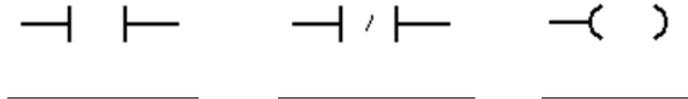
3- Cite 3 exemplos de elementos de saída digital em CLP.

1- _____ 2- _____ 3- _____

4- Cite 3 exemplos de elementos de Entrada digital em CLP.

1- _____ 2- _____ 3- _____

5- Com no Máximo 2 palavras, descreva cada um dos elementos a seguir.



- 6- Utilize a figura a baixo para desenvolver a programação em LADDER para que quando uma botoeira conectada a entrada 1 do CLP for acionada, ligar a saída 3 do CLP onde possui uma sinaleira conectada. Representar no desenho as ligações elétricas da botoeira e da sinaleira no CLP.



3.4.9 Apresentação PowerPoint

Para auxiliar no processo de aprendizagem, foi produzido uma apresentação para expor aos alunos como foi desenvolvido este trabalho, destacando as instituições de ensino envolvidas, SENAI-SC e UNOESC, além do apoio do Governo do Estado de SC através do programa de bolsas de estudos UNIEDU.

Nesta apresentação também é exposto a defasagem tecnológica do setor agroindustrial da região oeste de SC, demonstrando a importância deste trabalho para auxiliar no ensino e aprendizagem, tornando possível o desenvolvimento de sistemas automatizados para facilitar a vida do produtor rural.

Através de imagens é demonstrado o funcionamento do CLP e seus periféricos incluindo a eletrônica em blocos funcionais como entradas e saídas de sinais. Para finalizar, uma breve introdução à linguagens de programação e em destaque a linguagem LADDER utilizada para programar o CLP.

CLP Didático FIESC SENAI






Professor: André Umberto Faccioni

Objetivo FIESC SENAI

- Desenvolver uma plataforma didática para estudos e aprendizado em cursos técnicos e de qualificação em eletrônica e automação.




Justificativa FIESC SENAI

- Setor agroindustrial esta defasado tecnologicamente.



Justificativa FIESC SENAI

- Setor produtivo deve ser mais eficiente




O que é Automação? FIESC SENAI

- Automação é a aplicação de técnicas computadorizadas ou mecânicas para diminuir o uso de mão-de-obra em qualquer processo.
- A automação diminui os custos e aumenta a velocidade da produção.




O que é CLP? FIESC SENAI

- Controlador Lógico Programável (CLP) pode ser definido como um pequeno computador dedicado para Automação.
- Conceito:** É um equipamento eletrônico digital com hardware e software capaz de monitorar e controlar sistemas elétricos.





Como funciona o CLP? FIESC SENAI

- Os sinais elétricos oriundos através das entradas passam pelo processamento originando então as saídas do CLP.



```

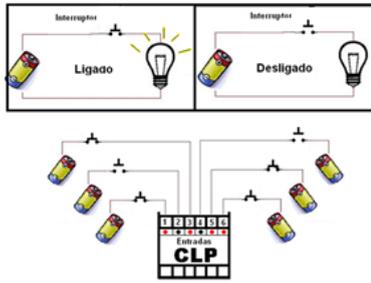
graph TD
    A[Entradas] --> B[Processamento]
    B --> C[Saídas]
    subgraph CLP
        A
        B
        C
    end
            
```

O que são Entradas? FIESC SENAI

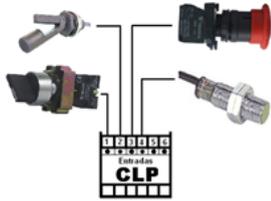
- São sinais elétricos ao qual provem de dispositivos como sensores, fim de curso, interruptor, botoeiras, entre outros.
- São dispositivos que dependendo de seus estado representa **Ligado** ou **Desligado**.



Entrada Digital *FIESC SENAI*
A FORÇA DA INICIAÇÃO CATARINENSE



Entrada Digital *FIESC SENAI*
A FORÇA DA INICIAÇÃO CATARINENSE



O que são Saídas? *FIESC SENAI*
A FORÇA DA INICIAÇÃO CATARINENSE

- As saídas de um CLP são destinadas a ligar e desligar dispositivos que estejam conectados a ele.

Exemplos de elementos de Saídas Digitais:

- Lâmpadas
- Sirenes
- Relês
- Contactores
- Eletroválvulas
- Motores



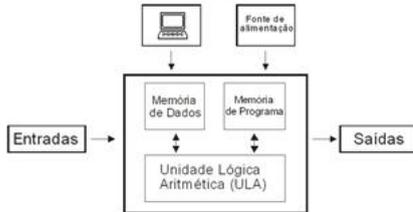
Saídas Digitais *FIESC SENAI*
A FORÇA DA INICIAÇÃO CATARINENSE

- Inúmeros são os equipamentos que podemos controlar utilizando um CLP.

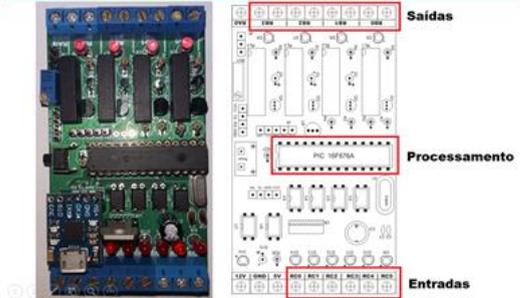


O que é o Processamento? *FIESC SENAI*
A FORÇA DA INICIAÇÃO CATARINENSE

- O CLP tem sua estrutura baseada no hardware de um computador.

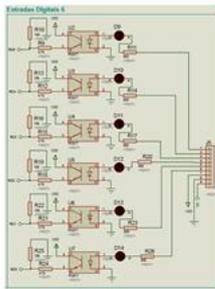


CLP Didático *FIESC SENAI*
A FORÇA DA INICIAÇÃO CATARINENSE



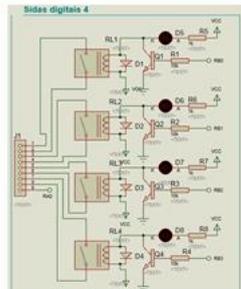
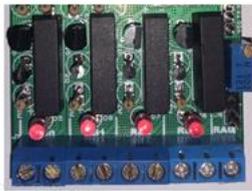
Eletrônica Entradas Digitais *FIESC SENAI*
A FORÇA DA INICIAÇÃO CATARINENSE

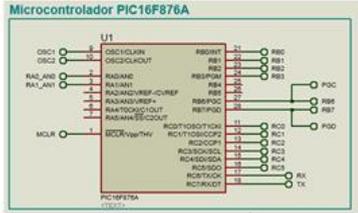
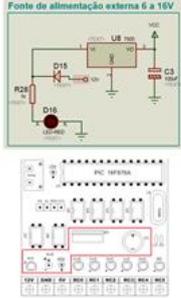
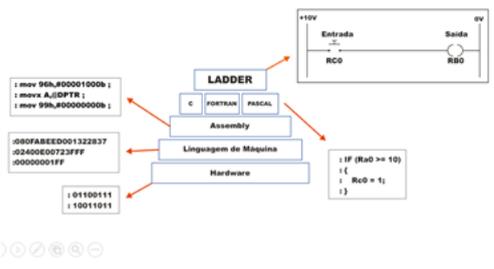
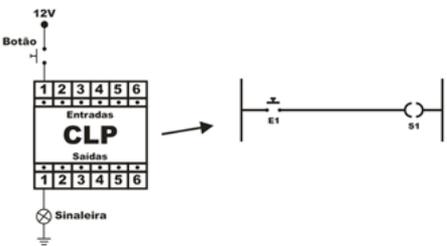
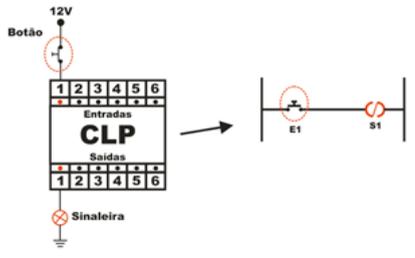
- 6 Entradas digitais
- Opto-acopladores de proteção.
- Indicação através de LED's



Eletrônica Saídas Digitais *FIESC SENAI*
A FORÇA DA INICIAÇÃO CATARINENSE

- Saída a Relé de 1A.
- Indicação de saída ativa com LED's.



<p>Microcontrolador Microchip PIC16F876A</p> <p style="text-align: right;">FIESC SENAI A FORÇA DA INDÚSTRIA CATARINENSE</p> <ul style="list-style-type: none"> Um dos mais utilizados em eletrônica 	<p>Fonte de Alimentação</p> <p style="text-align: right;">FIESC SENAI A FORÇA DA INDÚSTRIA CATARINENSE</p>  <ul style="list-style-type: none"> Entrada de tensão de 6 a 16V. Regulador de tensão 7805. Diodo de proteção contra inversão de polaridade. Capacitor de filtragem LED de indicação ligado.
<p>Linguagens de Programação</p> <p style="text-align: right;">FIESC SENAI A FORÇA DA INDÚSTRIA CATARINENSE</p> 	<p>Ladder</p> <p style="text-align: right;">FIESC SENAI A FORÇA DA INDÚSTRIA CATARINENSE</p> <ul style="list-style-type: none"> As instruções em Ladder são muito semelhantes aos usados em esquemas elétricos. 
<p>Programação</p> <p style="text-align: right;">FIESC SENAI A FORÇA DA INDÚSTRIA CATARINENSE</p> <ul style="list-style-type: none"> Distinção de Software e Hardware 	<p>Programação</p> <p style="text-align: right;">FIESC SENAI A FORÇA DA INDÚSTRIA CATARINENSE</p> 

3.4.10 Valores

A tabela de custos, expõem os valores unitários e valor total para confecção de uma placa CLP completa.

Estes valores podem se tornar menores, devido a compra dos itens em maior quantidade proporcionando assim um custo final ainda menor.

Tabela 7 - Custos unitário do CLP

Placa CLP Didático			
	Quant.	Valor U.	Total
Placa PCI Fabricada na DIGICART	1	R\$ 36,27	R\$ 36,27
Microcontrolador PIC16F876A-I/SP Through Hole	1	R\$ 8,00	R\$ 8,00
Micro usb para uart ttl 6pin módulo cp2102 conversor serial (obs. Mesma pinagem da foto)	1	R\$ 4,50	R\$ 4,50
Reed Relê 5Vcc 1A	4	R\$ 2,00	R\$ 8,00
Transistor BC337 Through Hole	4	R\$ 0,14	R\$ 0,56
Chave SS-12D07 180 Graus 3 Terminais	6	R\$ 0,30	R\$ 1,80
LED 3mm Vermelho Through Hole	11	R\$ 0,10	R\$ 1,10
Diodo 1n4007 Through Hole	9	R\$ 0,09	R\$ 0,81
Opto acoplador PC 817	6	R\$ 0,42	R\$ 2,52
Resistor through hole 1/4W			
1k	6	R\$ 0,03	R\$ 0,18
10k	5	R\$ 0,03	R\$ 0,15
4,7k	2	R\$ 0,03	R\$ 0,06
Resistor SMD 1206			R\$ -
330 ohms	7	R\$ 0,04	R\$ 0,28
4,7k	7	R\$ 0,04	R\$ 0,28
10k	7	R\$ 0,04	R\$ 0,28
Regulador 7805 (TO-220) PACKAGE	1	R\$ 1,28	R\$ 1,28
Chave Táctil 6x6x7mm 90º 4T	1	R\$ 0,31	R\$ 0,31
Soquete Torneado 28 Pinos SLIN	1	R\$ 3,81	R\$ 3,81
Capacitor			R\$ -
Capacitor Eletrolítico 100uf 25V	1	R\$ 0,16	R\$ 0,16
Capacitor Cerâmico 22pf	2	R\$ 0,11	R\$ 0,22
Conectores			R\$ -
Borne KF-301 3T	6	R\$ 0,96	R\$ 5,76
Barra de Pinos 1x40 vias 11,2mm 180 graus	1	R\$ 1,00	R\$ 1,00
Cristal 4 MHz Meia Caneca	1	R\$ 0,84	R\$ 0,84
Trimpot 3296W 5K	1	R\$ 1,36	R\$ 1,36
Caixa Patola DIN FUNDO 075	1	R\$ 25,57	R\$ 25,57
Sensor de temperatura LM35	1	R\$ 3,50	R\$ 3,50
TOTAL			R\$ 108,60

Fonte: O Autor

4 CONCLUSÕES

O objetivo principal deste trabalho, consistiu em desenvolver uma plataforma didática para ensino e estudos voltados para automação e eletrônica. Para criar esta plataforma foi necessário desenvolver um protótipo capaz de contemplar os requisitos mínimos de um CLP de uso industrial. No decorrer do desenvolvimento deste protótipo algumas dificuldades foram encontradas e solucionadas com o auxílio do Professor orientador.

Destaca-se a importância em fazer um bom planejamento da placa de circuito impresso, pois o primeiro layout da placa foi enviado para confecção e posteriormente detectado um erro mecânico na placa, que ficou entre o componente regulador de tensão 7805 e a placa de driver USB onde ambos ocupavam o mesmo espaço e assim não possibilitando a montagem. Este erro não agregou prejuízos muito altos pois foi confeccionado apenas uma peça. Posteriormente corrigido o erro e posto em fabricação 10 peças e não foi identificado outros erros.

Os resultados até o momento foram satisfatórios devido a funcionalidade da placa eletrônica, inclusive por ter sido projetada com um gabinete no padrão DIN, porque possibilitara a inserção em um painel elétrico para atender a aplicação no ensino nos cursos técnicos em Eletrotécnica e Eletromecânica do SENAI-Xanxerê. Esta inclusão em um painel elétrico vem de encontro para atender a estratégia de ensino do SENAI-SC em uma atividade denominada de Situação de Aprendizagem, que possibilita ao aluno desenvolver, planejar e aplicar os conhecimentos e habilidades adquiridas no decorrer do curso em uma situação problema real.

Melhorias serão realizadas no futuro breve, como, a inclusão de um display LCD, pois o gabinete escolhido possibilita esta inserção. Uma PCI com 6 chaves HH para facilitar a simulação de sinais de entrada. Novas atividades e exercícios práticos de programação Ladder e outros avanços poderão ser feitos.

REFERÊNCIAS

Assis, P. D. (Dezembro de 2004). **MICROCONTROLADOR**.

Luz, C. E. (2011). **Programando Microcontroladores PIC** (Vol. 1). (F. L. Dias, Ed.) São Paulo: Ensino Profissional.

Microchip, T. (2013). **PIC16F87XA Data Sheet**. Acesso em 2016

Miranda, L. C., Sampaio , F. F., & Borges , J. A. (2010). **RoboFácil**: Especificação e Implementação de um Kit de Robótica para a Realidade Educacional Brasileira. *Revista Brasileira de Informática na Educação*, pp. 47-58.

Miyadaira, A. N. (2013). **Microcontroladores PIC18** (4ª ed., Vol. 1). (R. A. Silva, Ed.) São Paulo: Editora Érica.

Santos, L. d. (Junho de 2009). **SISTEMA DE COMUNICAÇÃO USB**. Fonte: <http://tcc.ecomp.poli.br/20091/TCC%20-%20Leonardo%20Santos.pdf>

Souza, V. A. (2007). **Projetando com os Microcontroladores da Família PIC18**. (F. Dias, Ed.) São Paulo: Ensino Prfissional.

Westhues, J. (2007). **BUILDING LDMICRO**. Seattle, .

Xavier da Silva, S. R., & Barreto, P. L. (3 de Outubro de 2011). **ANÁLISE COMPARATIVA DE KITS DE ROBÓTICA EDUCATIVA**. *Congresso Brasileiro de Educação em Engenharia*, pp. 2-10.

Zanco, W. d. (2008). **Microcontroladores PIC** (2º ed.). (A. M. Cipelli, Ed.) São Paulo: Editora Érica.

Zanco, W. d. (2010). **Microcontroladores PIC18 com Linguagem C** (2ª ed., Vol. 1). (R. A. Silva, Ed.) São Paulo: Editora Érica.