CURSO DE ENGENHARIA ELÉTRICA

Caroline Marquardt Mueller

DESENVOLVIMENTO DE UM TESTADOR DE CABOS IMPLY

Santa Cruz do Sul

2021

Caroline Marquardt Mueller

DESENVOLVIMENTO DE UM TESTADOR DE CABOS IMPLY

Trabalho de conclusão apresentado ao Curso de Engenharia Elétrica da Universidade de Santa Cruz do Sul para obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Adriano José Bombardieri

Santa Cruz do Sul
2021

Dedico a Fabiane Schneider de Camargo, minha primeira professora, que ensinou e mostrou um mundo de conquistas.

AGRADECIMENTOS

Agradeço aos meus familiares pelo apoio e incentivo, aos professores e colegas que estiveram comigo no ensino básico, aos professores do curso de Engenharia Elétrica da UNISC pelos ensinamentos e principalmente pela amizade.

Também agradeço imensamente à Imply Tecnologia pela disponibilização dos seus recursos, dando espaço para criar inovação e desenvolver o lado profissional de sua colaboradora e principalmente por acreditar que pequenas ideias valem o investimento quando se tem compromisso.

RESUMO

Sabendo que a eletrônica é um meio de automação muito importante e que com ela se pode desenvolver diversas soluções, se pensou em usá-la para aprimorar processos internos da Imply® no que diz respeito a qualidade e usar os recursos de maneira que não haja desperdícios. O principal objetivo foi criar um projeto com um hardware e um firmware apropriado para aplicação que resultasse em um produto interno para facilitar procedimentos de fabricação e de testes da empresa. Usado uma metodologia aplicada para o desenvolvimento do projeto sendo necessário a criação de uma solução para um problema real. O projeto resultou em um produto interno que atende as expectativas da Imply® facilitando parte do processo de fabricação de cabos.

Palavras-chave: testador-de-cabos, hardware.

ABSTRACT

Knowing that electronics is a very important means of automation and that various solutions can be developed with it, we thought about using it to improve Imply®'s internal processes with regard to quality and use resources so that there is no waste . The main objective was to create a design with appropriate hardware and firmware for the application that would result in an in-house product to facilitate the company's manufacturing and testing procedures. Used an applied methodology for the development of the project, being necessary to create a solution to a real problem. The project resulted in an internal product that meets Imply® expectations, facilitating part of the cable manufacturing process.

Keywords: cable tester, hardware.

LISTA DE ILUSTRAÇÕES

Figura 1 - Conector PTR 4 vias 90º para PCI....14
Figura 2 - Conector MOLEX 4 vias 180º para PCI....14
Figura 3 - Conector JTS 4 vias 180º para PCI....15
Figura 4 - Conector BOX HEADER 10 vias 180º para PCI....15
Figura 5 - Conector RJ45 180º para PCI....15
Figura 6 - Conector DB9 fêmea 90º para PCI....15
Figura 7 - Conector VAL-U-LOK macho 2 vias....16
Figura 8 - Conector VAL-U-LOK macho 4 vias....16
Figura 9 - Conector VAL-U-LOK fêmea 2 vias....16
Figura 10 - Conector VAL-U-LOK fêmea 4 vias....16
Figura 11 - Display LCD 2x16....17
Figura 12 - LED 5mm vermelho difuso....17
Figura 13 - LED 5mm verde difuso....17
Figura 14 - Chave táctil (push button)....18
Figura 15 – Registrado de deslocamento com entrada serial e saída paralela....19
Figura 16 – Pinout do CI 4094....20
Figura 17 - Tabela verdade do CI 4094....20
Figura 18 – Tabela de descrição dos pinos do CI 74LS244....21
Figura 19 - Pinout do CI 74LS244....22
Figura 20 - Diagrama de pinos PIC18F452....23
Figura 21 - Circuito de alimentação da PCI....24
Figura 22 - Circuito do microcontrolador....25
Figura 23 - Circuito de LEDs....26
Figura 24 - Circuito das chaves push button....26
Figura 25 - Circuito do display 2x16....27
Figura 26 - Parte do circuito dos registradores de deslocamento....27
Figura 27 - Última parte do circuito dos registradores de deslocamento....28
Figura 28 - Ligação dos conectores BOX HEADER 20 vias....28
Figura 29 - Ligação dos conectores BOX HEADER 10 vias....28
Figura 30 - Ligação dos conectores BOX HEADER 26 vias....29
Figura 31 - Ligação dos conectores do tipo MOLEX....29
Figura 32 - Ligação dos conectores do tipo PTR de 2, 3 e 4 vias....29

Figura 33 - Ligação dos conectores do tipo PTR de 5 e 6 vias 29
Figura 34 - Ligação dos conectores do tipo JST 30
Figura 35 - Ligação dos conectores do tipo DB9.....30
Figura 36 - Ligação dos conectores do tipo RJ45 30
Figura 37 - Circuitos dos CIs 74LS244.....31
Figura 38 - Circuito de resistores pull down entre as vias 1 até 13 31
Figura 39 - Circuito de resistores pull down entre as vias 14 até 24 31
Figura 40 – Placa de circuito impresso do projeto(frente). 32
Figura 41 – Placa de circuito impresso do projeto (verso) 32
Figura 42 - IPYTST08 33
Figura 43 - Vista de perspectiva do esboço da estrutura 33
Figura 44 - Vista inferior do esboço da estrutura.....34
Figura 45 - Hardware do projeto (Perspectiva).....34
Figura 46 - Hardware do projeto (Vista inferior) 34
Figura 47 - Adaptador VAL-U-LOK fêmea 2 vias 35
Figura 48 - Adaptador VAL-U-LOK macho 2 vias.....35
Figura 49 - Adaptador VAL-U-LOK fêmea 4 vias 35
Figura 50 - Adaptador VAL-U-LOK macho 4 vias.....35
Figura 51 - Sequência para informação via display.....38
Figura 52 - Código fonte (parte 1) 38
Figura 53 - Código fonte (parte 2). 39
Figura 54 - Código fonte (parte 3) 39
Figura 55 - Código fonte (parte 4). 40
Figura 56 - Código fonte (parte 5). 40
Figura 57 - Código fonte (parte 6). 41
Figura 58 - Código fonte (parte 7). 41
Figura 59 - Código fonte (parte 8). 42
Figura 60 - Código fonte (parte 9) 43
Figura 61 - Código fonte (parte 10). 43
Figura 62 - Código fonte (parte 11) 44
Figura 63 - Código fonte (parte 12) 44
Figura 64 - Código fonte (parte 13) 45
Figura 65 - Código fonte (parte 14) 45
Figura 66 - Código fonte (parte 15) 46

Figura 67 - Código fonte (parte 16) 46
Figura 68 - Código fonte (parte 17) 47
Figura 69 - Código fonte (parte 18) 48
Figura 70 - Código fonte (parte 19) 49
Figura 71 - Circuito com cristal oscilador de 16MHz 51

LISTA DE ABREVIATURAS

CI	Circuito Impresso
I/O	Input/Output (Entrada/Saída)
ICSP	In-Circuit Serial Programming
ISO	Organização Internacional de Normalização
IT	Instrução De Trabalho
LCD	Display De Cristal Líquido
LED	Diodo Emissor de Luz
LSB	Less Significant Bit
MSB	Most Sigificant Bit
P&D	Pesquisa e Desenvolvimento
PCI	Placa de Circuito Impresso
PDIP	Plastic Dual In Line Package
SMD	Surface Mounted Device

SUMÁRIO

INTRODUÇÃO 11

1. IMPLY TECNOLOGIA ELETRÔNICA LTDA 12

2. TESTADOR DE CABOS IMPLY® 14

2.1 Registrador de deslocamento 18

2.2 Octal Bus Transceiver 21

2.3 Microcontrolador PIC18F452 22

3. HARDWARE DO TESTADOR DE CABOS 24

4. FIRMWARE DO PROJETO 37

4.1 Detalhamento do firmware 38

5. CONCLUSÃO 51

REFERÊNCIAS 52

INTRODUÇÃO

Os avanços da automação já são grandes aliados no meio industrial, cada vez mais entendemos sobre como é importante usar a tecnologia para favorecer processos e qualidade nas indústrias. Junto com a automação podemos notar um ramo muito importante que é o uso da eletrônica para aprimorar procedimentos dentro das empresas.

Com a Imply® não poderia ser diferente, uma empresa que não só vende tecnologia, mas como desfruta desse meio para melhorar seus produtos e recursos internos. Pensando em aprimorar ainda mais seus processos, iniciou-se um projeto para o desenvolvimento de um testador para assegurar qualidade e resolver problemas que poderiam aparecer em determinadas produções.

Sabendo que uma grande parte do catálogo de produtos da empresa dispõe de fabricação de cabos, tanto para alimentação como para sinais de dados, se planejou a criação de um testador para certificar-se que a produção teria uma porcentagem de cabos realizados com pleno funcionamento.

A realização desse projeto visa melhorar a qualidade dos produtos e assegurar que o processo seja realizado de acordo com a ISO, já que a empresa tem certificado ISO9001. Com o testador de cabos será possível validar por meio de instruções de trabalho sua correta fabricação podendo futuramente servir como base para dados de relatórios quando necessários.

A metodologia escolhida para a abordagem do projeto é de pesquisa aplicada, pois há um problema que atualmente não se tem como assegurar se uma produção de cabos está correta ou não, e a proposta de solução sendo desenvolver um testador para suprir essa necessidade.

1. IMPLY TECNOLOGIA ELETRÔNICA LTDA

Fundada em 2003 a Imply está presente em mais de 125 países tendo uma vasta gama de soluções para Acessos & Ticketing, Autoatendimento, Bowling e Painéis. Conta com alto desenvolvimento tecnológico e 100% do desenvolvimento dos próprios Softwares e Hardwares.

Tem como missão desenvolver, produzir e comercializar produtos e serviços inovadores visado superar as expectativas dos seus clientes. A visão de ser referência em tecnóloga por muitas gerações e seus Valores são a inovação, motivação planejamento, lucratividade e yes, formando com as iniciais dos valores o nome IMPLY.

A Imply® conta com onze princípios, são eles:

1 - INOVAÇÃO: Buscar novas ideias e transformá-las em realidade para fazer a diferença. Pensar grande, ser curioso e criativo.

2 - INICIATIVA: Ter espírito empreendedor. Atitude pioneira. Ser o exemplo todos os dias em nossas ações. Agir com sabedoria, disciplina e responsabilidade.

3 - INTERATIVIDADE: Espírito de Equipe. Agir com sinergia, sermos acessíveis e dinâmicos. Compartilhar ideias e democratizar conhecimentos com o time. Buscar e disseminar as melhores práticas.

4 - INTEGRAÇÃO: Sermos abertos e inclusivos. Preservar o meio ambiente, trazendo benefícios para a comunidade e para o mundo. Investir nas relações pessoais, criando uma rede de contatos baseada em vínculos amigáveis, com ética, integridade e respeito mútuo.

5 - INVESTIDOR: Analisar cada projeto para ter lucratividade, possibilitando o crescimento. Minimizar riscos e reduzir custos, buscando aumentar nossa rentabilidade com qualidade.

6 - INTERNACIONALIZAÇÃO: Estarmos atentos ao mercado mundial e sermos reconhecidos como referência.

7 - INTUIÇÃO: Estar atento às novas oportunidades, agindo com a cabeça e com o coração. Ousadia com consciência dos riscos.

8 - IMEDIATA: Agir com rapidez, clareza e objetividade. Não desperdiçar o tempo com coisas inúteis. Não deixar para depois o que pode fazer agora. Se começou, termine.

9 - INCENTIVO: As melhores oportunidades serão das melhores pessoas. Investir no autodesenvolvimento, qualificação e no aperfeiçoamento constante para sermos os

melhores naquilo que fazemos. Desenvolver os talentos para formar melhores sucessores. Continuar a crescer e aprender.

10 - IMPRESSIONANTE: Sempre dar o melhor de si. Buscar a excelência, fazer tudo bem-feito e com qualidade. Buscar sempre os melhores resultados. Superar as expectativas. Parceria e dedicação ao sucesso de cada cliente.

11 - IMPORTANTE: Ser um excelente ambiente, com modernidade, conforto e segurança. Acreditar em si e na Imply®. Inspirar confiança. Defender nossa cultura, cultivar nossa reputação, e promover nossa marca.

Um ponto muito importante em relação a empresa é sobre a gestão de qualidade que ela emprega nos seus produtos e processos, tanto que se mantém a certificação da ISO9001 que é um sistema de gestão que tem por objetivo garantir a otimização de processos dentro da empresa para melhor desempenho.

Conhecendo melhor uma parte do processo para podermos seguir com o projeto do testador, vamos expor um exemplo de como era alguns aspectos da produção.

O setor responsável pela produção do cabeamento dos produtos Imply® é a Eletrônica, então eles recebem um pedido de fabricação desse cabeamento e ele é realizado com base nas instruções de trabalho fornecidas pelo setor de pesquisa e desenvolvimento (P&D), quando finalizado essa produção é entregue para próximo setor, o setor da Montagem.

Mas havia nesse processo uma falha, pois quando um produto, finalizado pelo setor de Montagem, apresentasse um problema de cabeamento só era realmente visto na Montagem pois não havia nenhum procedimento de teste do cabeamento no setor de fabricação.

Se o pedido fosse, por exemplo, cem unidades de um determinado tipo de cabo, feito de maneira equivocada, só seria descoberto o erro na montagem do produto, desperdiçando tempo e materiais que muitas vezes não podem ser reaproveitados.

Havendo uma necessidade de melhoria, foi pensado no desenvolvimento de um testador de cabos para o setor de Eletrônica, podendo garantir o perfeito funcionamento dos mesmos. Sabe-se que esse projeto pode não atender 100% dos tipos de cabos fabricados, mas é capaz de atender entre 95% e 97% das variedades de cabos fabricados pela Imply®.

2. TESTADOR DE CABOS IMPLY®

Para iniciar o projeto temos que ter noção do que deverá ser realizado. O esperado pela empresa é uma PCI com um firmware capaz de suprir os testes para cabeamento. A ideia inicial é apenas um teste de continuidade, sem a necessidade de envolver questões mais complexas como ruídos ou impedância de comprimento de cabos. Para tal temos que conhecer quais os conectores de cabos são usados na Imply®, já que no mercado existe muitas variações de conectores, para saber a quantidade de vias que devemos testar.

Com auxílio do software de gestão que a empresa usa sabemos os conectores para PCI que ela dispõe. São eles: conectores do tipo PTR de duas, três, quatro, cinco e seis vias, conectores do tipo MOLEX KK de duas, três, quatro e seis vias, conectores do tipo JST de duas, três, quatro e seis vias, conectores do tipo box header de dez, vinte e vinte e seis vias, conectores do tipo RJ45 com oito vias e conectores DB9 com nove vias, exemplos ilustrados das figuras 1 a 6.

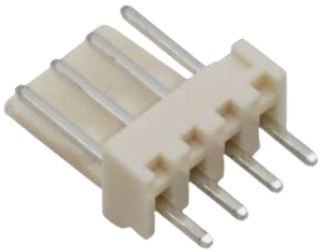
Figura 1 - Conector PTR 4 vias 90º para PCI

Fonte: Imply®

Figura 2 - Conector MOLEX 4 vias 180º para PCI

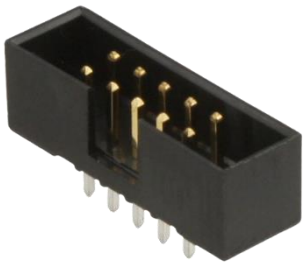
Fonte: Imply®

Figura 3 - Conector JTS 4 vias 180º para PCI



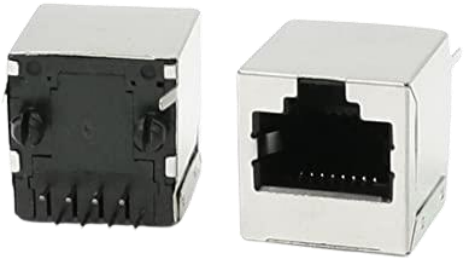
Fonte: Imply®

Figura 4 - Conector BOX HEADER 10 vias 180º para PCI



Fonte: Imply®

Figura 5 - Conector RJ45 180º para PCI



Fonte: Imply®

Figura 6 - Conector DB9 fêmea 90º para PCI

Fonte: Imply®

Existe um conector do tipo VAL-U-LOK, macho e fêmea com duas e quatro vias que são muito utilizados no cabeamento, ilustrados nas figuras 7, 8, 9 e 10

respectivamente, mas como não dispõem de conectores específicos para PCI esses deverão ser adaptados.

Figura 7 - Conector VAL-U-LOK macho 2 vias

Fonte: Imply®

Figura 8 - Conector VAL-U-LOK macho 4 vias

Fonte: Imply®

Figura 9 - Conector VAL-U-LOK fêmea 2 vias

Fonte: Imply®

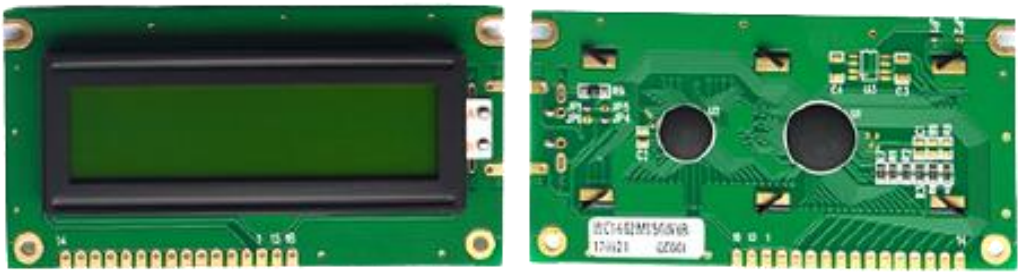
Figura 10 - Conector VAL-U-LOK fêmea 4 vias

Fonte: Imply®

A PCI do projeto deverá ser capaz de dispor de todos os conectores para atender os especificados anteriormente, além de possuir uma interface para o usuário. Como interface será usado um display 2x16 muito comum no mercado e usado pela empresa, conforme figura 11.

Será incluído da mesma forma dois LED para indicar cabos com defeitos e cabos corretos, sendo eles de cor vermelha para os errados e cor verde para os cabos corretos, ilustrados nas figuras 12 e 13 respectivamente. Será utilizado também duas chaves “push button” para usuário ter como configurar o teste, mostrado na figura 14.

Figura 11 - Display LCD 2x16



Fonte: Imply®

Figura 12 - LED 5mm vermelho difuso

Fonte Imply®

Figura 13 - LED 5mm verde difuso

Fonte Imply®

Figura 14 - Chave táctil (push button)

Fonte: Imply®

Começando do fato que testaremos apenas cabos com conexões do tipo pino-a-pino podemos dispor todos os conectores na placa em paralelo, conhecendo que o maior cabo em número de vias seria um cabo flat de vinte e seis vias esse será o número máximo de saídas que teremos que administrar, além de suas entradas.

Um ponto importante para se basear é no pedido de usar os componentes que a empresa dispõe em seu catálogo de materiais, criando a lógica do testador vamos realizar o teste da seguinte maneira: acionar uma saída e verificar qual entrada foi ligada, para isso podemos usar os circuitos integrados de registradores de deslocamento.

A leitura das entradas ligadas pode ser feita pelo controle de um circuito integrado chamado octal bus transceiver. Fazendo com que não precise de muitos pinos do microcontrolador pois associando esses CIs em paralelo reduz o número de pinos a serem usados, além de utilizar um microcontrolador para realizar a lógica do teste.

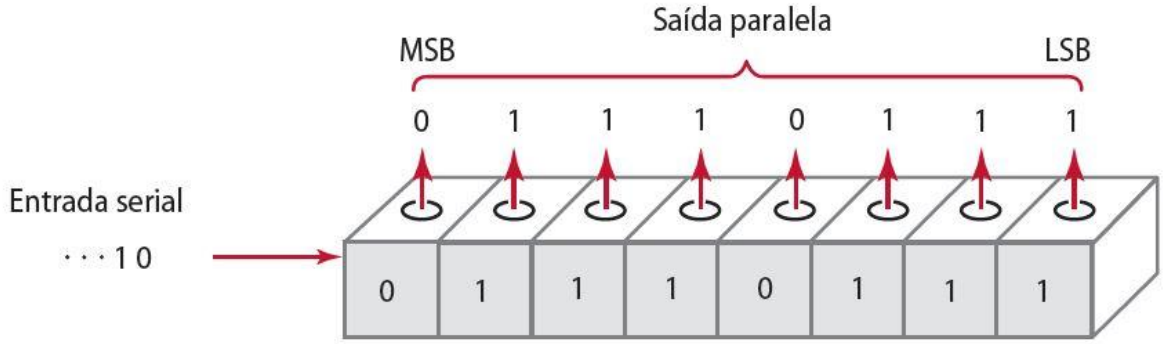
O entendimento de usar esses circuitos integrados é para facilitar a lógica, já que um microcontrolador não tem muitos pinos de I/O (entradas e saídas) e o maior número de vias que devemos atender seriam acionando 26 saídas e lendo 26 entradas respectivamente.

2.1 Registrador de deslocamento

> Um registrador é um conjunto de células de memória arranjadas como um único dispositivo. Por exemplo, um registrador de 8 bits pode ser utilizado para armazenar informações que serão posteriormente utilizadas, ou o registrador pode ser projetado para manipular os dados, como no caso do registrador de deslocamento. Este último dispositivo pode modificar o conteúdo dos dados, deslocando-os para a direita ou para a esquerda. (TOKHEIM, 2013, p: 269)

O registrador escolhido para o projeto é o CI 4094, um shift register de 8 bits com entrada serial e saída paralela conforme ilustração da figura 15, onde MSB significa “Most Sigificant Bit”, bit mais significativo e LSB “Less Significant Bit” é o bit menos significativo:

Figura 15 – Registrado de deslocamento com entrada serial e saída paralela



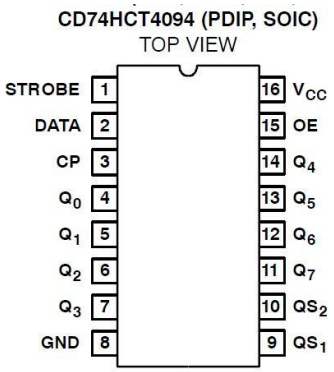
Fonte: TOKHEIM, 2013

Como temos uma quantia de pinos que devemos monitorar de vinte e seis unidade iremos usar três registradores, totalizando vinte e quatro pinos, mais dois pinos diretos do microcontrolador para realizar os acionamentos das saídas e como vamos trabalhar apenas com cabos pino-a-pino podemos acionar uma saída a cada vez verificando quais entradas foram acionadas, nos levando para um resultado de quais saídas acionaram as entradas individualmente.

A folha de dados do componente pode ser acessada via fabricante, o CI que a Imply® dispõe é da Texas Instruments® de nome CD74HC4094 com invólucro do tipo PDIP com a pinagem conforme figura 16 abaixo, sua alimentação pode ser entre 2 V e 6 V, tendo pino de STROBE, que funciona para atualizar todas as saídas do CI juntas, o pino DATA, onde chega a informação serial fornecida pelo microcontrolador, o pino CLOCK (CP) para realizar o ciclo dos dados enviados pela serial, pino OE que habilita ou não as saídas.

Nesse projeto não será necessário, pinos Q0 à Q7 sendo as saídas paralelas dos dados onde vamos adotar que o Q0 será MSB e Q7 o LSB, e os pinos Q’S e QS saídas seriais para quando usar mais de um registrador.

Figura 16 – Pinout do CI 4094



Fonte: Texas Instruments®

Figura 17 - Tabela verdade do CI 4094

TRUTH TABLE

INPUTS				PARALLEL OUTPUTS		SERIAL OUTPUTS	
CP	OE	STR	D	Q_0	Q_n	QS_1 (NOTE 1)	QS_2
↑	L	X	X	Z	Z	QÕ6	NC
↓	L	X	X	Z	Z	NC	Q_7
↑	H	L	X	NC	NC	QÕ6	NC
↑	H	H	L	L	Q_n –1	QÕ6	NC
↑	H	H	H	H	Q_n –1	QÕ6	NC
↓	H	H	H	NC	NC	NC	Q_7

H = High Voltage Level, L = Low Voltage Level, X = DonÕt Care, NC = No charge, Z = High Impedance Off–state,
↑ = Transition from Low to High Level, ↓ = Transition from High to Low.
NOTE:
1. At the positive clock edge the information in the seventh register stage is transferred to the 8th register stage and QS1 output.

Fonte: Texas Instruments®

Como observado na figura 17, a folha de dados do registrador nos fornece a tabela verdade que tem como objetivo entender a lógica necessária para operar o CI, sendo que a letra H é sinal de nível lógico alto, letra L é o sinal de nível lógico baixo, letra X não tem nível de importância, letra Z indica alta impedância, a flecha para cima significa a transição para a borda de subida de um sinal e a flecha para baixo significa a transição para a borda de descida de um sinal.

Usaremos o Q'S para dar sequência no registrador seguinte pois o CLOCK dos registradores deve ser o mesmo para os três, fazendo com que a mudança entre um registrador e outro ocorra na borda de descida do CLOCK.

Ainda com a tabela verdade sabemos que o sinal que está no pino DATA só é transmitido para as saídas quando gerado o pulso de CLOCK por causa da borda de operação e quando o pino STROBE estiver habilitado em nível lógico a alto.

2.2 Octal Bus Transceiver

O circuito integrado 74LS244 é um componente que de acordo com sua folha de dados, disponível pela Texas Instruments®, pode ser aplicado para drivers de motores, servidores, tipos de chaveamento para rede entre outros, mas o que usaremos é a sua capacidade de expandir o número de I/O, podendo lembrar um funcionamento de um buffer[1].

Figura 18 – Tabela de descrição dos pinos do CI 74LS244

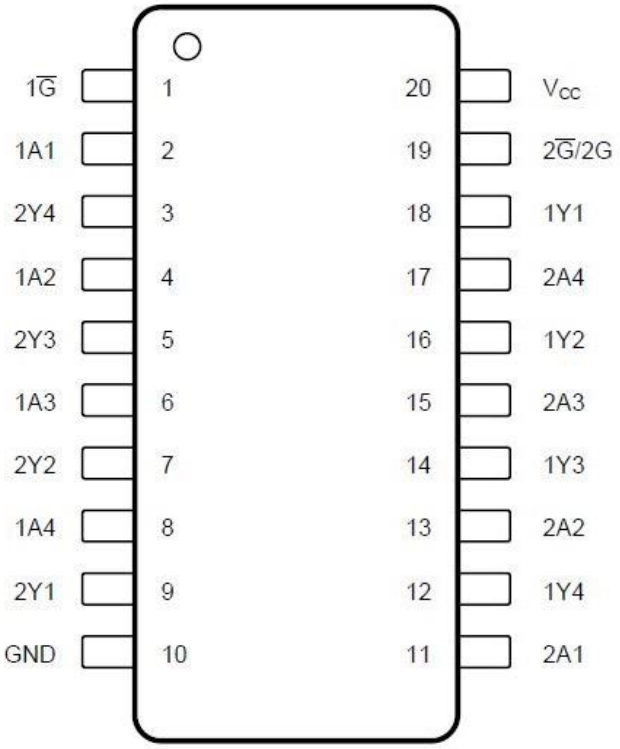
Pin Functions

PIN		I/O	DESCRIPTION
NO.	NAME		
1	$1\overline{G}$	I	Channel 1 output enable
2	1A1	I	Channel 1, A side 1
3	2Y4	O	Channel 2, Y side 4
4	1A2	I	Channel 1, A side 2
5	2Y3	O	Channel 2, Y side 3
6	1A3	I	Channel 1, A side 3
7	2Y2	O	Channel 2, Y side 2
8	1A4	I	Channel 1, A side 4
9	2Y1	O	Channel 2, Y side 1
10	GND	—	Ground
11	2A1	I	Channel 2, A side 1
12	1Y4	O	Channel 1, Y side 4
13	2A2	I	Channel 2, A side 2
14	1Y3	O	Channel 1, Y side 3
15	2A3	I	Channel 2, A side 3
16	1Y2	O	Channel 1, Y side 2
17	2A4	I	Channel 2, A side 4
18	1Y1	O	Channel 1, Y side 1
19	$2\overline{G}/2G$ (1)	I	Channel 2 output enable
20	V_{CC}	—	Power supply

Fonte: Texas Instruments®

[1] De acordo com Infopedia: buffer - parte da memória utilizada para guardar dados temporariamente, enquanto estes se transferem entre dispositivos que operam a velocidades ou em formatos diferentes. Porto: Porto Editora. Disponível em https://www.infopedia.pt/dicionarios/lingua-portuguesa/buffer

Figura 19 - Pinout do CI 74LS244



Fonte: Texas Instruments®

Observando as figuras 18 e 19 acima, serão usados todos os canais disponíveis no CI tendo os pinos de enable dos dois canais em comum sendo controlados pelo microcontrolador do projeto. Cada CI dispõe de oito canais, para o projeto serão usados ao todo três unidades onde as saídas dos canais serão paralelas e chegarão como entradas o microcontrolador, exemplo todas as três saídas 1Y1 dos Cis deverão estar ligadas ao mesmo pino do microcontrolador, para ele poder realizar a lógica necessária.

2.3 Microcontrolador PIC18F452

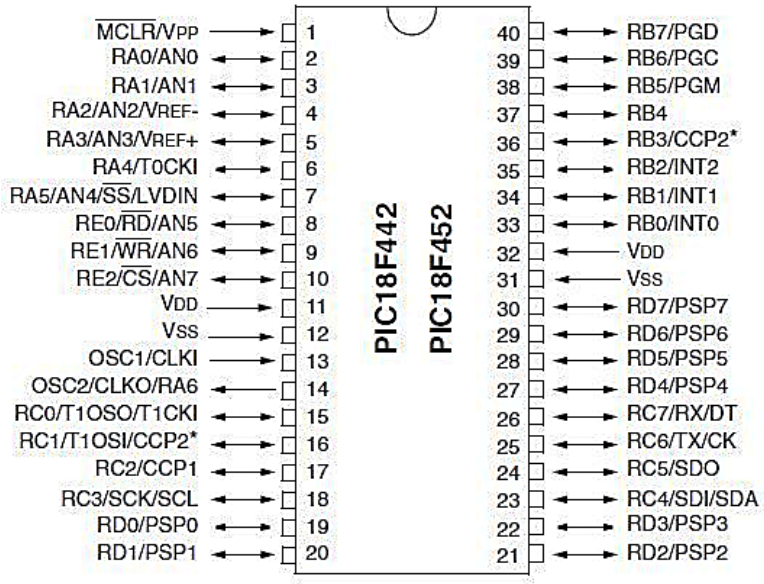
Um microcontrolador é um dispositivo semicondutor em forma de circuito integrado que consiste em várias partes básicas de outros elementos, como microprocessador, memórias não voláteis (aquelas capazes de armazenar dados que mesmo desligando de uma fonte de energia seus dados são conservados) e voláteis (aquelas que tem o comportamento em contraste das não-voláteis), portas de comunicação serial, comunicação paralela, conversores analógicos/digitais,

conversores digitais/analógicos, entradas e saídas digitais, entre outros elementos. (GIMENEZ, 2002)

O PIC18F452 é um componente usado em algumas aplicações pela Imply® tendo em estoque quantidade significativa, por este motivo escolhido para ser o microcontrolador do projeto, responsável pela execução de toda a lógica. Sua folha de dados pode ser achada facilmente no site da fabricante, a Microchip®.

A tensão de uso desse microcontrolador pode variar entre 2 V até 5 V, tensão essa aplicada a qualquer pino dele, sendo usado para canais analógicos e entradas ou saídas digitais. A pinagem pode ser conferida de acordo com a figura 15, usaremos o invólucro DIP disponível na empesa.

Figura 20 - Diagrama de pinos PIC18F452



Fonte: Microchip®

Os pinos podem ser chamados de acordo com as portas em que se encontram. Sendo da porta A até E nesse caso. Cada pino de I/O suporta até a tensão de entrada dita anteriormente, como usaremos 5 V, cada pinos de I/O irá operar com 5 V, tanto para entradas como saídas.

Sabendo que serão utilizados LEDs, pode-se estabelecer um valor de resistor para eles com base na tensão dos pinos do microcontrolador. De acordo com a primeira Lei de Ohm, a resistência é a diferença de potencial dividido pela intensidade de corrente. Usando um LED que opera com máximo de 2 V, existe uma diferença de potencial de 3 V para o resistor e como um LED opera com corrente de até 20mA a resistência mínima seria em torno de 150Ω.

3. HARDWARE DO TESTADOR DE CABOS

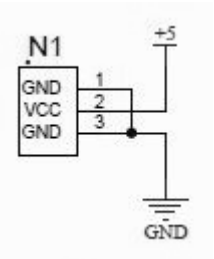
Quando falamos em hardware podemos usar como a definição de objetos tangíveis, como circuitos integrados, placas de circuito impresso, cabos, fontes de alimentação, memórias e impressoras, algo que não seja abstrato como algoritmos ou instruções. (TANENBAUM, 2013)

Para Imply® o hardware de um projeto é toda a parte física em relação à eletrônica dele, desde a sua placa de circuito impresso até os componentes usados e quando necessário quaisquer tipos de adaptadores referentes ao uso do hardware. Além de englobar, em alguns projetos, a parte estrutural de engenharia, no projeto do testador teremos além do hardware e firmware a parte da estrutura, onde disporemos de uma base para a PCI facilitando o uso do testador.

A empresa disponibiliza um programa para realizar o esquema elétrico do projeto e roteamento da placa, usaremos ele para criar o desenho da placa de circuito impresso, para posteriormente pedir a fabricação da placa com empresas do ramo e que tem a Imply® como cliente.

Na figura 21 abaixo mostra a alimentação da placa do testador, será usado uma fonte comercial de 5V/2A com plug P4 para alimentar os periféricos, assim na placa será usado um conector P4.

Figura 21 - Circuito de alimentação da PCI



Fonte: Do autor

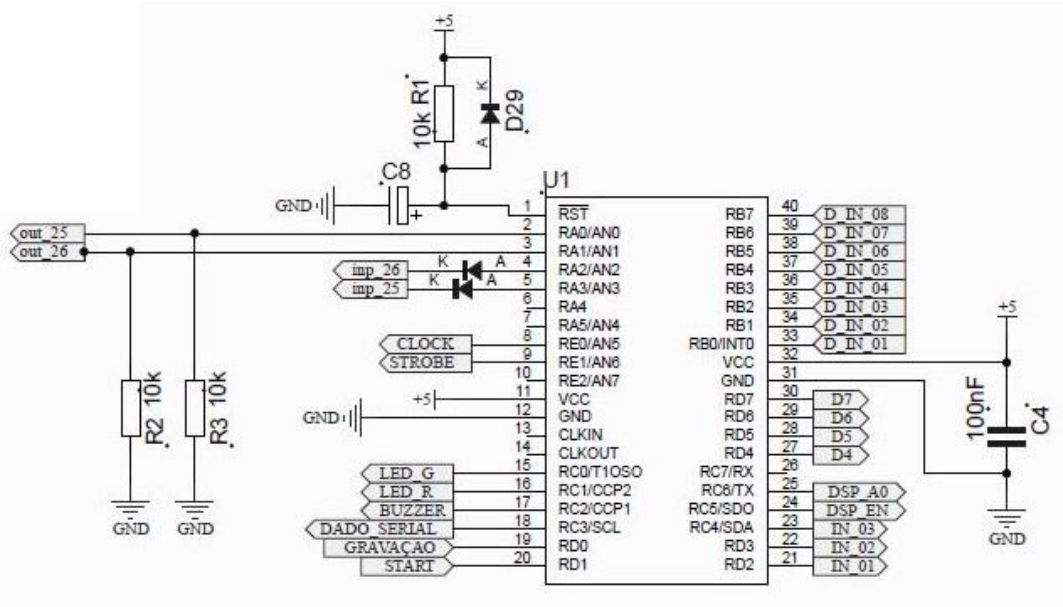
Na parte do circuito do microcontrolador podemos observar a figura 22, onde se tem um capacitor de desacoplamento no pino 1, servindo para suavizar a ligação do CI. Os pinos 2 e 3 serão as duas entradas que faltariam no 74LS244 tendo resistores de pull down assegurando nível lógico baixo quando não há acionamentos, evitando um nível desconhecido para o microcontrolador.

Pinos 4 e 5 são as duas saídas que faltariam no registrador de deslocamento, usando diodo 1N4148 para evitar que algum sinal retornasse, queimando o pino do microcontrolador.

Pino 8 e 9 serão o CLOCK e o STROBE, respectivamente, dos registradores sendo comum aos três, os pinos 15, 16 e 17 são os acionamentos dos LEDs e BUZZER, esse que não será mais implementado pois como terá sinalização luminosa se preferiu apenas manter o circuito caso queiram futuramente implementar um sinal sonoro com um novo firmware.

O pino 18 é onde sairá os dados seriais para o primeiro registrador e os pinos 19 e 20 são as leituras das chaves.

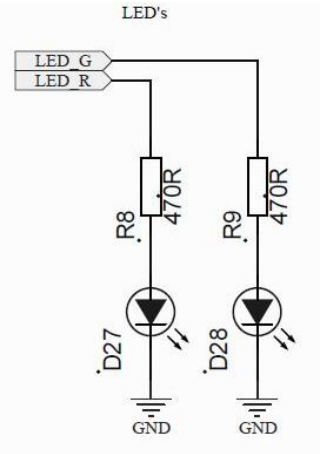
Figura 22 - Circuito do microcontrolador



Fonte: Do autor

Ainda em relação a figura 22 temos os pinos da porta B do microcontrolador que estão as ligações do três 74LS244 para facilitar o firmware posteriormente, pinos 27 ao 30 que serão os dados para o display, os pinos 24 e 25 que são os sinais de configuração do display e por fim os pinos 21, 22 e 23 que são para selecionar os 74LS244 para leitura, habilitando ou não suas saídas.

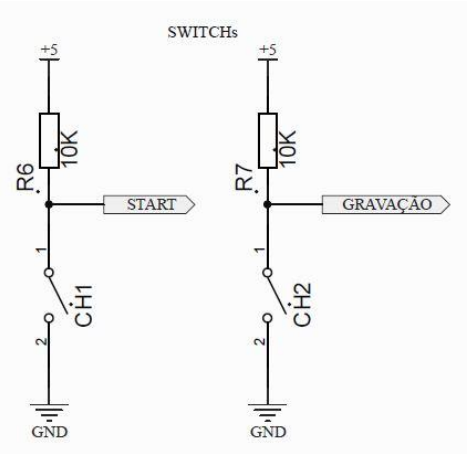
Figura 23 - Circuito de LEDs



Fonte: Do autor

A figura 23 acima mostra o circuito de acionamento dos LEDs através dos pinos do microcontrolador vistos anteriormente, já a figura 24 abaixo se vê os circuitos das chaves para leitura do microcontrolador, tendo elas as funções da configuração para o usuário do testador podem ajustar o que for necessário.

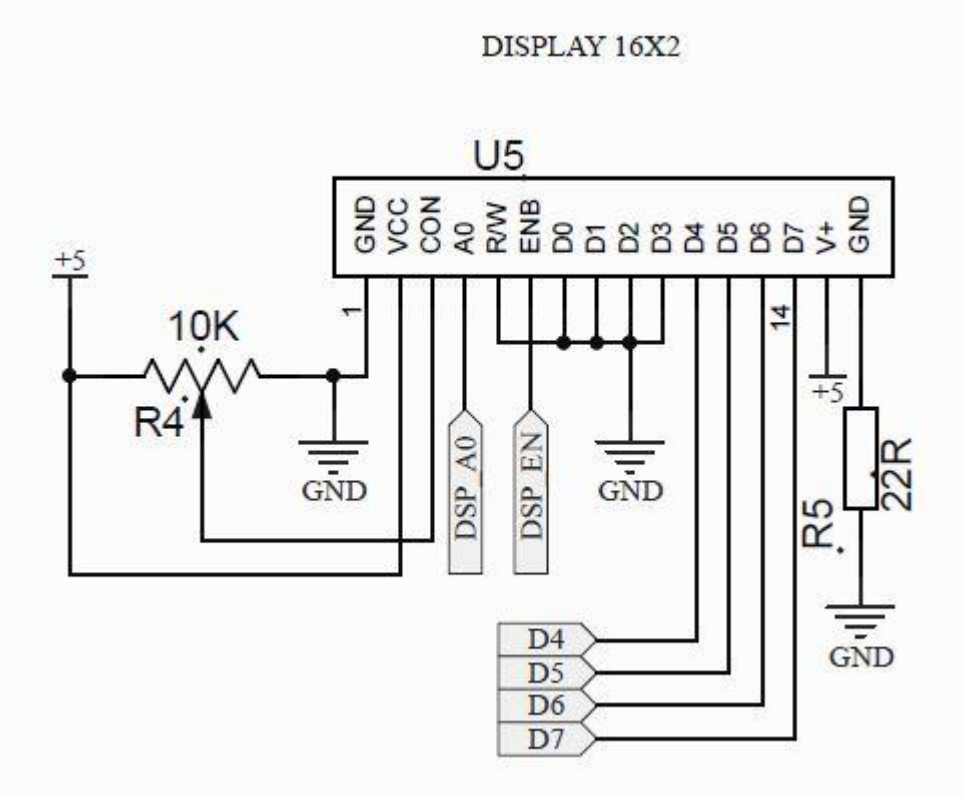
Figura 24 - Circuito das chaves push button



Fonte: Do autor

Na figura 25 temos o circuito do display, tendo os pinos de dados e configurações diretamente do microcontrolador e os pinos de alimentação diretos da PCI com um trimpot (tipo um resistor variável) para regular a intensidade da tela do display.

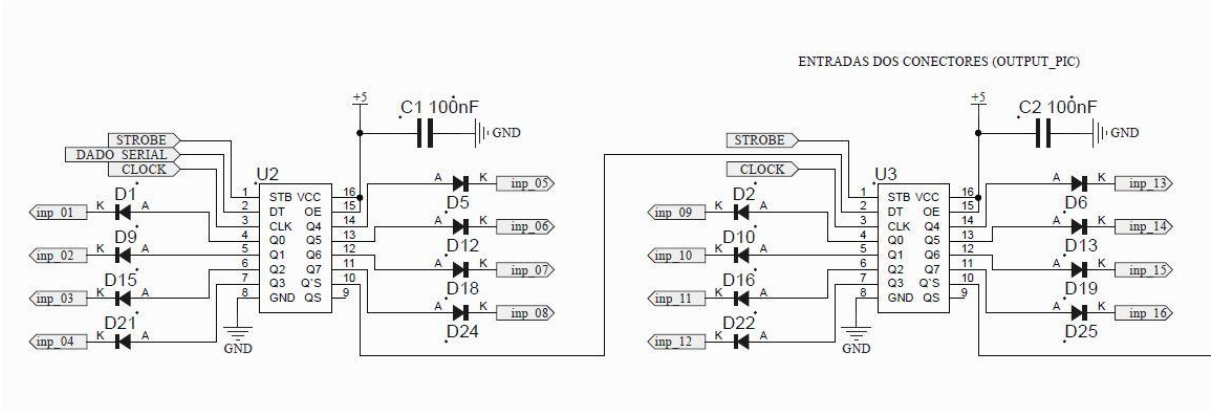
Figura 25 - Circuito do display 2x16



Fonte: Do autor

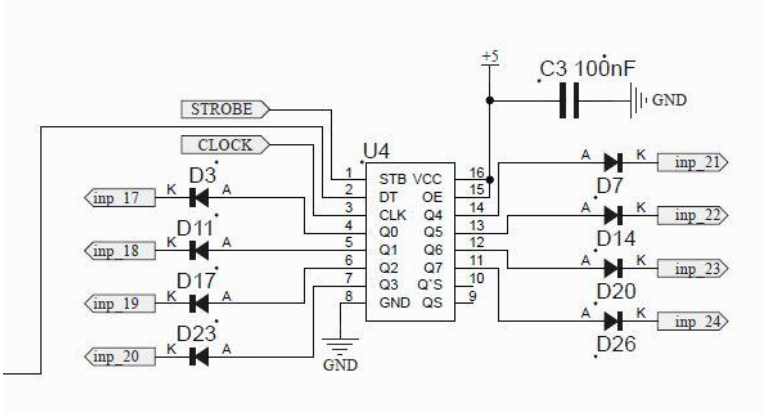
O circuito dos registradores de deslocamento, estão divididos nas duas figuras abaixo a 26 e 27, nelas podemos ver que cada saída dos Cis tem um diodo com a finalidade de não ter nenhum sinal de retorno, evitando que o pino seja danificado, podemos observar também que os pinos de STROBE e CLOCK são os mesmos que saem do microcontrolador.

Figura 26 - Parte do circuito dos registradores de deslocamento



Fonte: Do autor

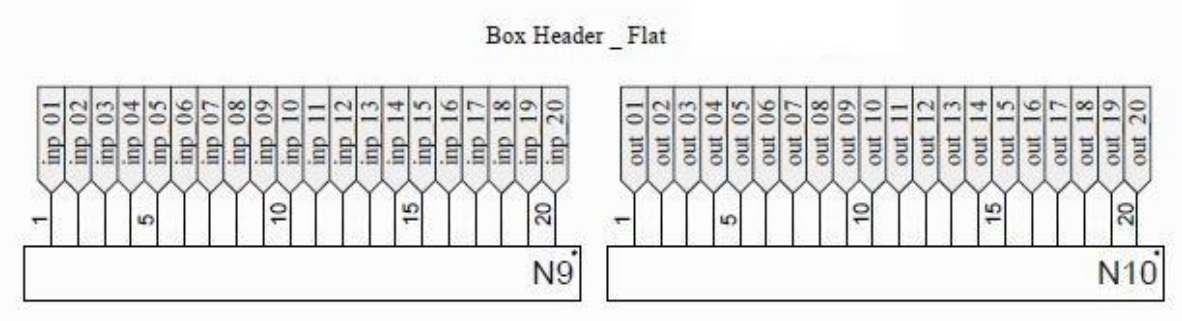
Figura 27 - Última parte do circuito dos registradores de deslocamento



Fonte: Do autor

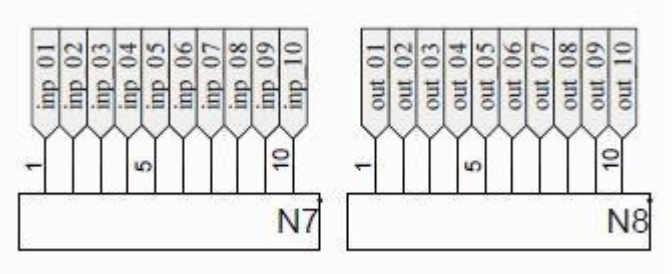
De acordo com as figuras anteriores se nota que o primeiro CI 4094 é o LSB e o último CI será o MSB, cada uma dessas saídas dos registradores estão associadas aos conectores, cada saída do CI 4094 é uma via dos conectores de acordo com as quantidades deles.

Figura 28 - Ligação dos conectores BOX HEADER 20 vias



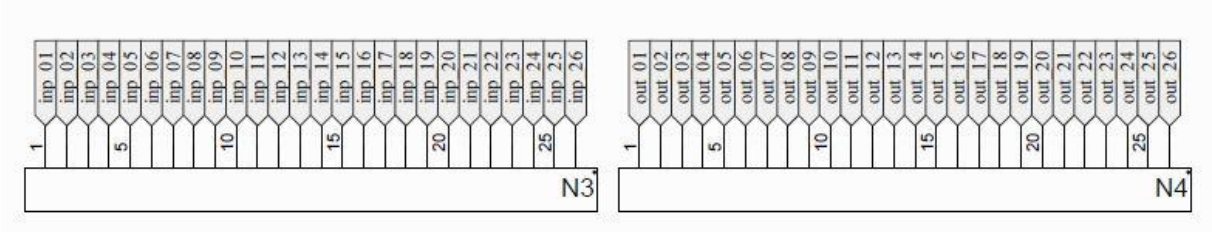
Fonte: Do autor

Figura 29 - Ligação dos conectores BOX HEADER 10 vias



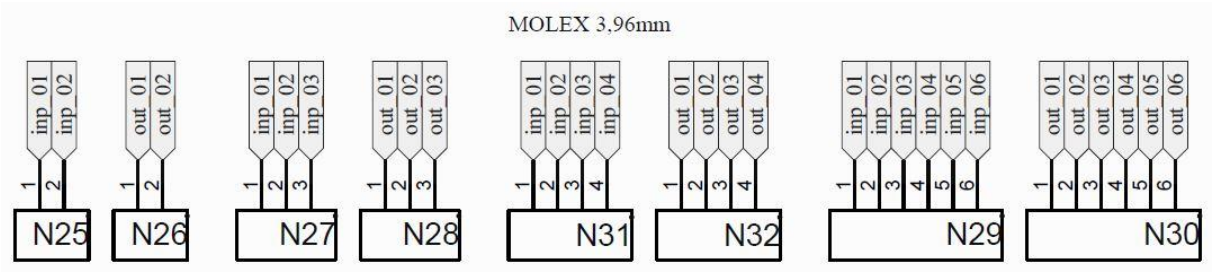
Fonte: Do autor

Figura 30 - Ligação dos conectores BOX HEADER 26 vias



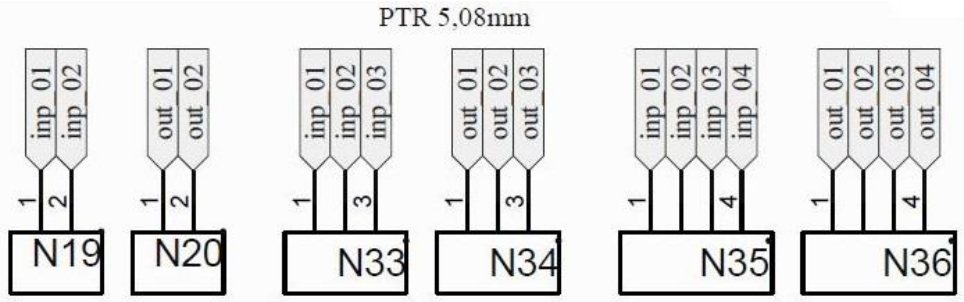
Fonte: Do autor

Figura 31 - Ligação dos conectores do tipo MOLEX



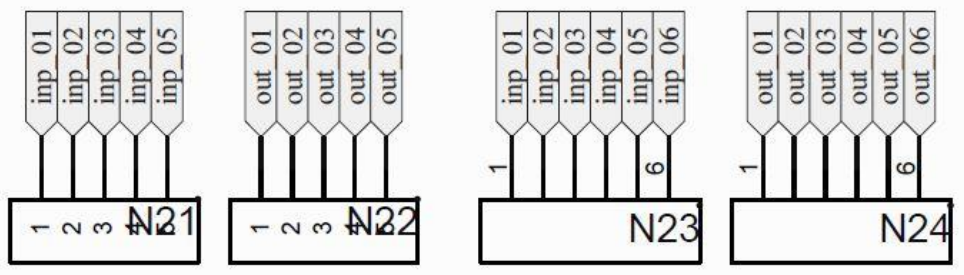
Fonte: Do autor

Figura 32 - Ligação dos conectores do tipo PTR de 2, 3 e 4 vias



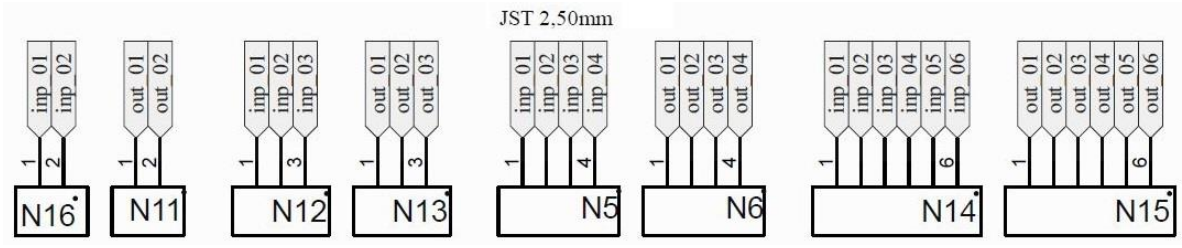
Fonte: Do autor

Figura 33 - Ligação dos conectores do tipo PTR de 5 e 6 vias



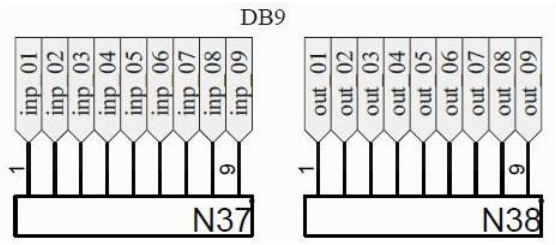
Fonte: Do autor

Figura 34 - Ligação dos conectores do tipo JST



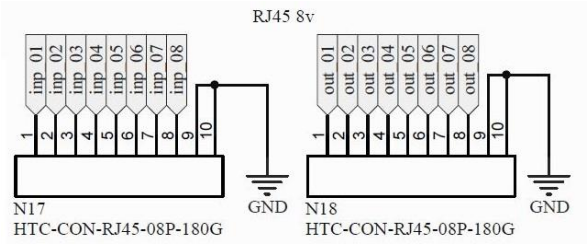
Fonte: Do autor

Figura 35 - Ligação dos conectores do tipo DB9



Fonte: Do autor

Figura 36 - Ligação dos conectores do tipo RJ45

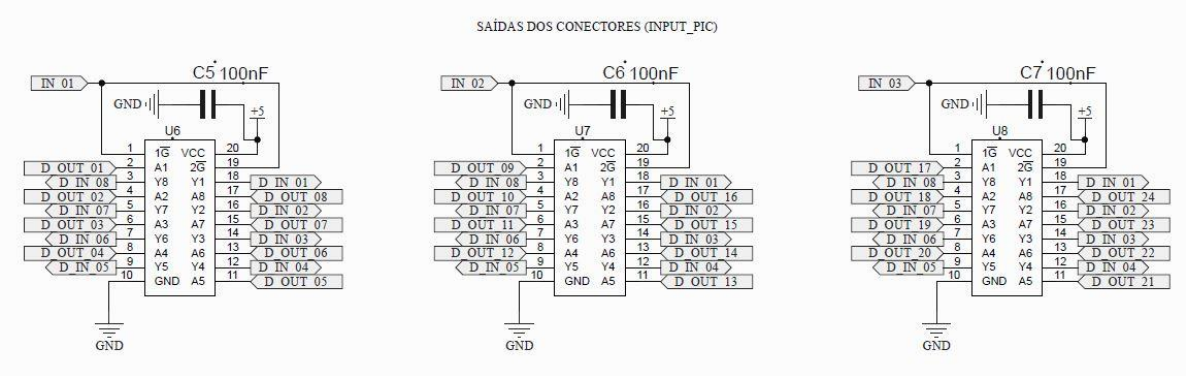


Fonte: Do autor

Das figuras 28 até a 36 acima, estão dispostas todas as ligações dos conectores que foi utilizado no projeto, podendo ser observado que os pinos estão em paralelo, tanto as saídas como as entradas.

De acordo com a figura 37 abaixo temos o circuito dos Cis 74LS244 tendo a entrada do sinal dos componentes os conectores e a saída do CI indo para o microcontrolador além de ter os pinos de seleção para criar a lógica no firmware posteriormente.

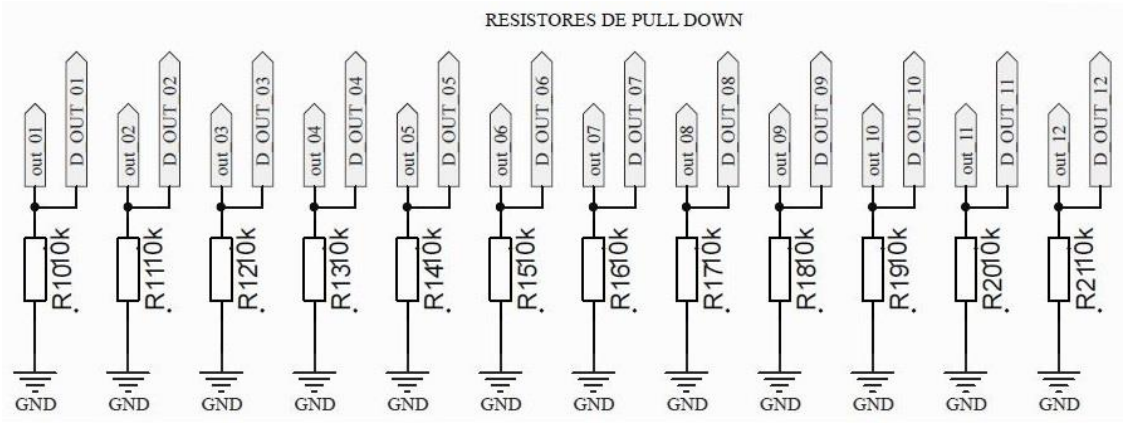
Figura 37 - Circuitos dos CIs 74LS244



Fonte: Do autor

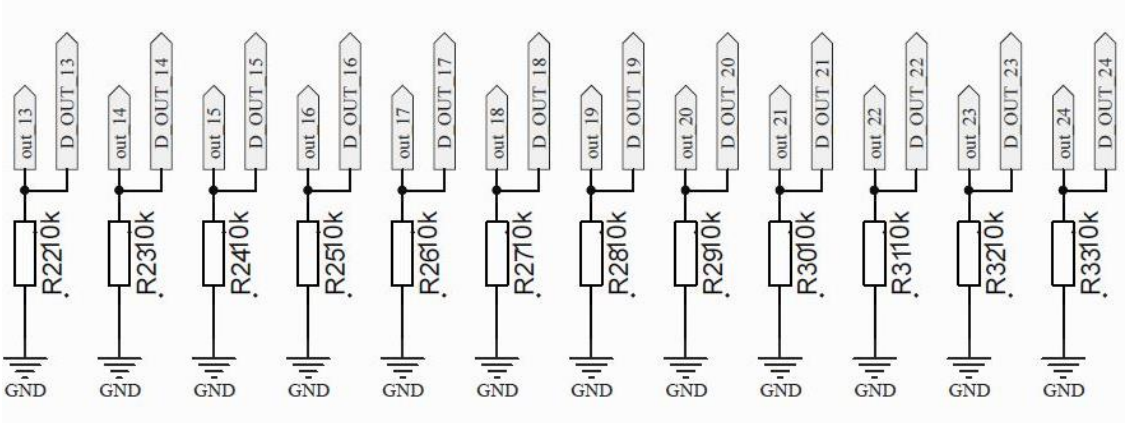
Para garantir nível lógico adequado para a leitura dos 74LS244 se usou de resistores de pull down conforme figura 38 e figura 39, garantindo nível lógico baixo quando não há acionamentos de pinos.

Figura 38 - Circuito de resistores pull down entre as vias 1 até 13



Fonte: Do autor

Figura 39 - Circuito de resistores pull down entre as vias 14 até 24

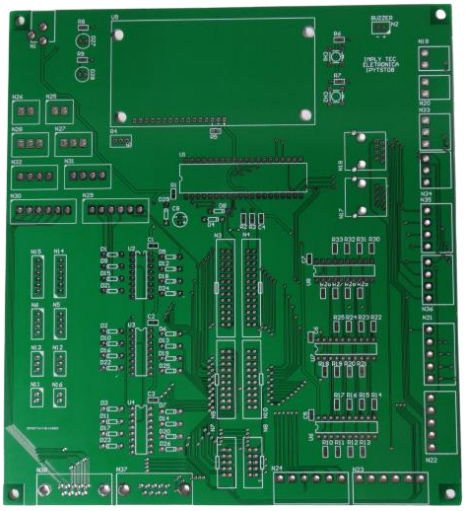


Fonte: Do autor

Após o esquemático ser criado foi roteado a placa de circuito impresso seguindo os padrões da Imply® conforme se vê na figura 40 e figura 41 abaixo. O circuito impresso sem os componentes na figura 42 se observa a placa já com todos os componentes soldados.

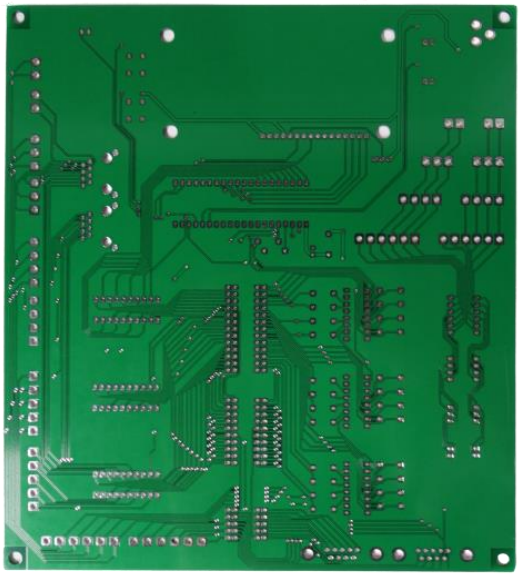
Nesse momento perante a gestão da Imply® essa placa com os componentes recebe um código de gestão e uma descrição, a placa da figura 42 passa a ser definida como IPYTST08, sendo o “IPY” abreviação de IMPLY, “TST” para placa de testes e o número “08” pois segue uma sequência desse tipo de placas na empresa.

Figura 40 – Placa de circuito impresso do projeto(frente).



Fonte: Do autor

Figura 41 – Placa de circuito impresso do projeto (verso)



Fonte: Do autor

Figura 42 - IPYTST08

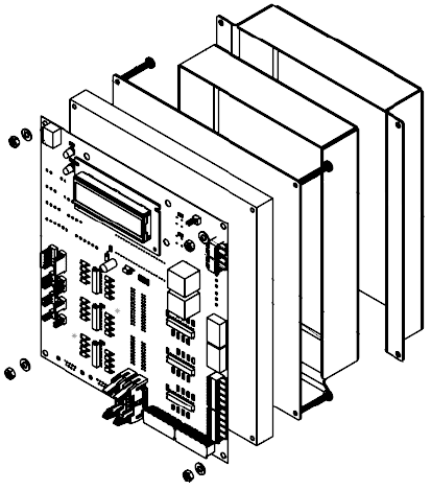
Fonte: Do autor

Importante destacar que foi usado resistores do tipo SMD e soquetes para o microcontrolador, CI 74LS244 e o CI 4094 para ter uma melhor disposição dos componentes na placa e facilitar a troca do CIs caso seja necessário.

A gravação do microcontrolador não é feita pela placa, por isso não foi necessário o circuito de gravação via ICSP, na Imply® se usa gravador externo as placas de circuitos.

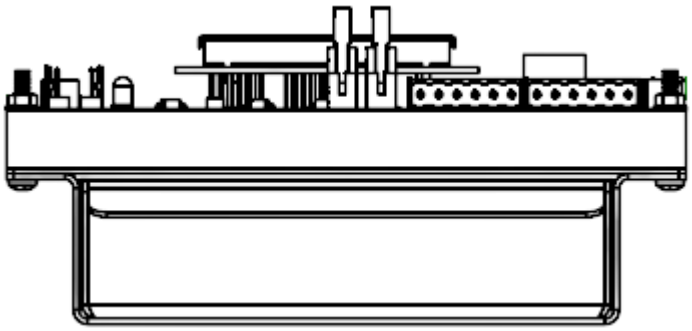
Para a estrutura ou suporte da IPYTST08 foi criado algo com uma pequena gaveta para se ter um espaço para guardar os adaptadores e fonte se necessário, de acordo com as seguintes figuras podemos ver o esboço:

Figura 43 - Vista de perspectiva do esboço da estrutura



Fonte: Imply®

Figura 44 - Vista inferior do esboço da estrutura

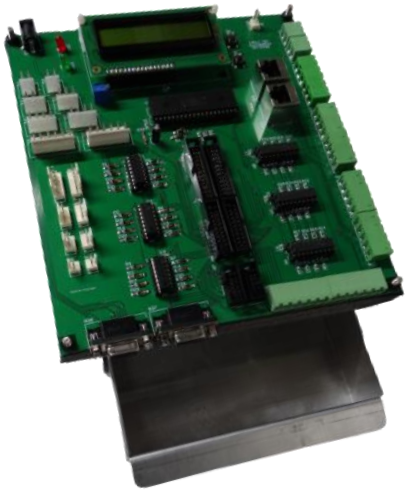


Fonte: Imply®

A parte da estrutura do projeto da IPYTST08 foi criado pelo setor de P&D Engenharia da Imply® e disponibilizado os esboços para a publicação desse trabalho, pois a empresa conta com engenheiros mecânicos para seus projetos onde há necessidade de estruturas.

Tendo os esboços e a IPYTST08 em mãos, foi finalizado a estrutura do projeto conforme figuras 45 e 46 abaixo.

Figura 45 - Hardware do projeto (Perspectiva).



Fonte: Do autor

Figura 46 - Hardware do projeto (Vista inferior)

Fonte: Do autor

Como falado anteriormente, existe alguns tipos de conectores na Imply® que não estão disponíveis para PCI, mas como são muito usados nos projetos da empresa então foi criado alguns adaptadores para esses casos, conforme as figuras abaixo vemos os adaptadores que poderão ser usados na IPYTST08 através dos conectores PTR 2 vias e 4 vias.

Figura 47 - Adaptador VAL-U-LOK fêmea 2 vias

Fonte: Do autor

Figura 48 - Adaptador VAL-U-LOK macho 2 vias

Fonte: Do autor

Figura 49 - Adaptador VAL-U-LOK fêmea 4 vias

Fonte: Do autor

Figura 50 - Adaptador VAL-U-LOK macho 4 vias

Fonte: Do autor

Na figura 47 mostra o adaptador para usar algum cabo que tenha o conector do tipo VAL-U-LOK macho com 2 vias e na figura 48 o adaptador para cabos com conectores VAL-U-LOK fêmea de 2 vias.

Enquanto na figura 49 tem o adaptador para cabos com VAL-U-LOK macho de 4 vias e na figura 50 o adaptador para usar em cabos com conectores do tipo VAL-U-LOK fêmea com 4 vias.

Realizado os adaptadores juntamente com a parte de estrutura, mais a PCI IPYTST08, se dá como finalizado a parte do hardware do projeto.

4. FIRMWARE DO PROJETO

> Firmware é um programa que está exclusivamente armazenado em uma memória não volátil de um equipamento. No mínimo, o firmware tem a finalidade principal de programar a forma de operação do hardware, como números de portas de entradas e saídas, forma de comunicação serial, forma de operação com timers e contadores entre outras coisas. (Gimenez, 2002, p. 3)

O firmware do projeto deve conter a lógica necessária para o funcionamento dele, para isso devemos saber o que será usado e como, para então poder definir parâmetros, variáveis e até como fazer a programação. Na Imply® se usa um compilador para escrever códigos para microcontroladores, a linguagem usada será a C, muito conhecida e de fácil entendimento para essa aplicação.

Como já foi constatado anteriormente já sabemos o que será saída e entradas dos conectores, mas não havíamos determinado como identificar o teste, para isso vamos descrever como deverá ser realizado um teste de um cabo qualquer.

1º Alimentar a placa com a fonte;

2º Plugar o cabo que se deseja testar nos respectivos conectores ou adaptadores;

3º Verificar se a descrição do display está conforme a informação contida na instrução de trabalho desse cabo.

4º Caso a informação do display estiver correta, validar o cabo com o botão de GRAVAR, se não estiver correta refazer cabo e voltar ao passo 2.

5º Testar os demais cabos da instrução verificando qual LED liga, sendo vermelho para cabo com defeito e verde para cabo correto.

Podemos notar que se trata de um teste simples, o usuário devera interagir com os botões e o display apenas na validação da primeira unidade e após apenas ir colocado e tirando os cabos para testar.

Como usamos um display de duas linhas por dezesseis colunas vamos dispor na linha superior os pinos de saída e na linha inferior qual pino de entrada foi acionado. Para facilitar a visualização usaremos de 1 a 9 e as letras de A até Q formando uma sequência de 1 até 26 conforme figura 48 abaixo:

Figura 51 - Sequência para informação via display

PINO	1	2	3	4	5	6	7	8	9	10	11	12	13
DISPLAY	1	2	3	4	5	6	7	8	9	A	B	C	D

PINO	14	15	16	17	18	19	20	21	22	23	24	25	26
DISPLAY	E	F	G	H	I	J	K	L	M	N	O	P	Q

Fonte: Do autor

Entre cada número ou letra terá um espaço para melhor visualização, assim podemos mostrar até oito pinos por vez na tela do display, tendo umas das chaves para navegar entre as telas quando um cabo tiver mais que oito vias.

4.1 Detalhamento do firmware

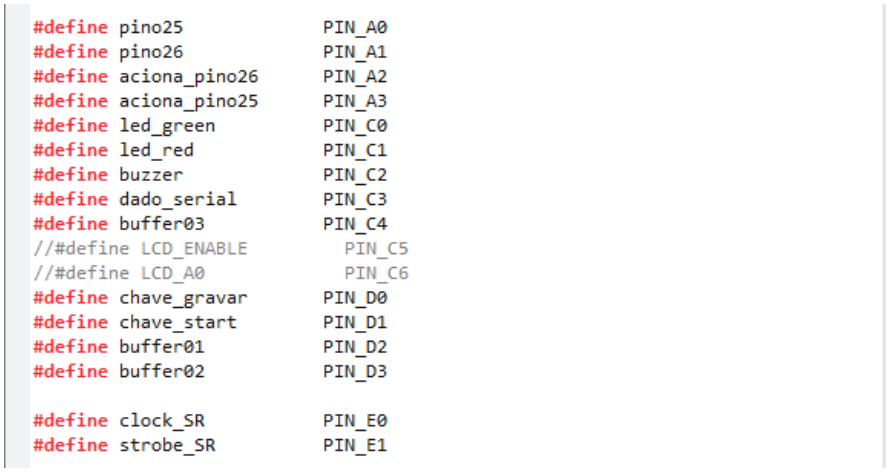
Figura 52 - Código fonte (parte 1)

```
/*
   FIRWMARE: IPYTST08, PLACA DE TESTADOR DE CABOS
   AUTOR: CAROLINE MARQUARDT MUELLER
   DATA: 06/09/2021

*/

#include <18F452.h>

#device ADC=10

#include <string.h>

//
#use      fast_io(A)
#use      fast_io(B)
#use      fast_io(C)
#use      fast_io(D)
#use      fast_io(E)

#fuses    HS, WDT, PUT, PROTECT, NODEBUG, BORV45, NOLVP

#use delay( CLOCK= 16000000, RESTART_WDT )
#use FIXED_IO( A_outputs=PIN_A3,PIN_A2 )
#use FIXED_IO( B_outputs=PIN_B0,PIN_B1,PIN_B2,PIN_B3,PIN_B4,PIN_B5,PIN_B6, PIN_B7 )
#use FIXED_IO( C_outputs=PIN_C6,PIN_C5,PIN_C4,PIN_C3,PIN_C2,PIN_C1,PIN_C0 )
#use FIXED_IO( D_outputs=PIN_D7,PIN_D6,PIN_D5,PIN_D4,PIN_D3,PIN_D2 )
#use FIXED_IO( E_outputs=PIN_E1,PIN_E0 )
```

Fonte: Do autor

Conforme figura 52 foi incluído a biblioteca do microcontrolador e definido modos de operações de acordo com o usado nos projetos da Imply® além de definir os pinos de saídas e de entradas do microcontrolador e a frequência de operação de 16MHz.

Figura 53 - Código fonte (parte 2).

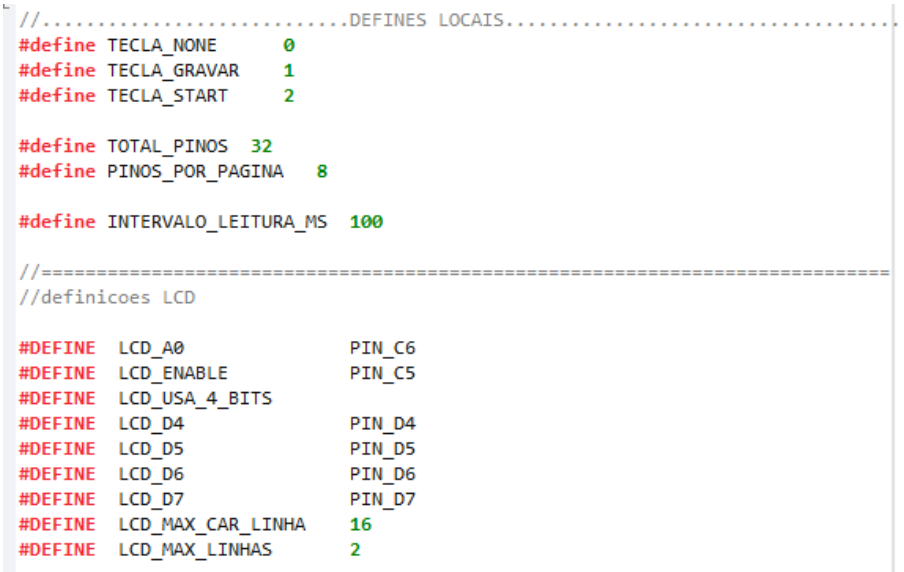


```
#define pino25              PIN_A0
#define pino26              PIN_A1
#define aciona_pino26       PIN_A2
#define aciona_pino25       PIN_A3
#define led_green           PIN_C0
#define led_red             PIN_C1
#define buzzer              PIN_C2
#define dado_serial         PIN_C3
#define buffer03            PIN_C4
//#define LCD_ENABLE          PIN_C5
//#define LCD_A0              PIN_C6
#define chave_gravar        PIN_D0
#define chave_start         PIN_D1
#define buffer01            PIN_D2
#define buffer02            PIN_D3

#define clock_SR            PIN_E0
#define strobe_SR           PIN_E1
```

Fonte: Do autor

Já pela figura 53, mostra os nomes definidos dos pinos utilizados, para maior facilidade no decorrer do código fonte, dar nomes que referenciam as aplicações simplifica na hora de criar as lógicas necessárias.

Figura 54 - Código fonte (parte 3)



```
//.........................DEFINES LOCAIS.....................................
#define TECLA_NONE      0
#define TECLA_GRAVAR    1
#define TECLA_START     2

#define TOTAL_PINOS  32
#define PINOS_POR_PAGINA   8

#define INTERVALO_LEITURA_MS  100

//===============================================================================
//definicoes LCD

#DEFINE  LCD_A0              PIN_C6
#DEFINE  LCD_ENABLE          PIN_C5
#DEFINE  LCD_USA_4_BITS
#DEFINE  LCD_D4              PIN_D4
#DEFINE  LCD_D5              PIN_D5
#DEFINE  LCD_D6              PIN_D6
#DEFINE  LCD_D7              PIN_D7
#DEFINE  LCD_MAX_CAR_LINHA   16
#DEFINE  LCD_MAX_LINHAS      2
```

Fonte: Do autor

Aqui na figura 54 se observa a definição dos pinos para uso do display, será usado uma biblioteca específica da Imply®, então por motivos de confidencialidade

será apenas mostrado seu uso e não os detalhes de foram feitas as suas funções e definidos variáveis com valores fixos.

Figura 55 - Código fonte (parte 4).

```
//.....................VARIÁVEIS GLOBAIS.................................//
unsigned int8
   Tecla = TECLA_NONE;

unsigned int16
   Timer= 0;

//declarada na Flash
const unsigned int8 tabelaCaracteres[]={"123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"};

unsigned int1
   LerEntradas= False,
   Timeout= False;
```

Fonte: Do autor

Na figura acima trás as declarações das variáveis globais, ou seja, são variáveis que podem ser acessadas de qualquer parte do programa. (PEREIRA, 2009)

Figura 56 - Código fonte (parte 5).

```
void init_pinos (void){
   output_high(buffer01);
   output_high(buffer02);
   output_high(buffer03);

   output_low(strobe_SR);
   output_low(clock_SR);

   output_low(aciona_pino25);
   output_low(aciona_pino26);

   output_low(buzzer);

}
```

Fonte: Do autor

Figura 56 acima mostra uma função para inicialização dos pinos no começo do nosso firmware, uma função basicamente é um subprograma, muito usada para agrupar comandos melhorando a lógica.

Os pinos de seleção dos 74LS244 são descritos como “buffer01”, “buffer02” e “buffer03” eles devem iniciar em nível alto pois conforme descrito anteriormente o 74LS244 é selecionado através do nível lógico baixo, ou seja, em “low”.

Já os pinos do CI 4094 devem iniciar em nível baixo pois a seleção desses é realizada pelo nível lógico alto, tanto o “clock_SR” quando o “strobe_SR”. O pino que seria utilizado o buzzer ficara em nível lógico baixo para permanece desligado, não sendo implementado nesse projeto.

Figura 57 - Código fonte (parte 6).

```
void clockSR (void){
   output_high(clock_SR);
   delay_us(10);
   output_low(clock_SR);
}

void strobeSR (void){
   output_high(clock_SR);
   delay_us(10);
   output_low(clock_SR);
}
```

Fonte: Do autor

Conforme figura 57 existe duas funções sobre o CI 4094, os registradores de deslocamento. Como os pinos foram iniciados em nível baixo ao chamar essas funções fazendo com que o nível lógico suba faz a borda de subida necessária para o funcionamento do CI e após devem voltar ao estado “low” para um próximo ciclo.

Figura 58 - Código fonte (parte 7).

```
void EnviaSerial_SR ( unsigned int32 valor){

   unsigned int8 bitNum;

   for ( bitNum = 24; bitNum; bitNum -- ){
      if(bit_test(valor, bitNum-1)){
         output_high(dado_serial);
      }else{
         output_low(dado_serial);
      }
      clockSR();
   }

   strobeSR( );

}
```

Fonte: Do autor

A função da figura 58 serve para enviar os dados para o CI 4094, como usaremos três unidades sempre enviaremos os vinte e quatro bits correspondentes a cada pino, por isso usado uma variável de “unsigned int32”. O valor do dado, sendo 1 (nível lógico alto) ou 0 (nível lógico baixo), é controlador pela saída “dado_serial”. Após

o envio de cada bit é executado um pulso do Clock do CI com a função já descrita e depois enviado todos os vinte e quatro bits é habilitado com a função do Strobe.

Figura 59 - Código fonte (parte 8).

```
unsigned int32 leEntradas( void ){
   unsigned int32 aux;

   output_low(buffer01);
   aux = input_b();
   output_high(buffer01);

   output_low(buffer02);
   aux |= (unsigned int32)input_b() << 8;
   output_high(buffer02);

   output_low( buffer03 );
   aux |= (unsigned int32)input_b() << 16;
   output_high(buffer03);

   if( input(pino25) )
      aux |= (unsigned int32)1 << 24;

   if( input(pino26) )
      aux |= (unsigned int32)1 << 25;

   return aux;

}
```

Fonte: Do autor

Esta função da figura 59 acima retorna uma variável “int32” que possui a informação de qual entrada foi acionada, no primeiro momento é lido o primeiro 74LS244 adicionando esse valor na variável sendo nos bits menos significativos, já a segunda leitura dos pinos de entradas acionadas é adicionada a variável nos bits seguintes não sobre escrevendo o que já foi lido.

Num terceiro momento é acrescentado ao valor da variável a soma desta terceira leitura deslocado em dezesseis bits, sendo as últimas leituras realizadas pelos pinos diretos do microcontrolador e deslocados para ocupar os bits vinte e quatro e vinte e cinco da variável.

Figura 60 - Código fonte (parte 9)

```
void atualizarVetor (unsigned int32 vetorEntradas[]){

   unsigned int8 bitNum;
   unsigned int32 valor;

   for( bitNum = 0; bitNum < 24; bitNum++ ){
      valor = 0;
      bit_set( valor, bitNum );
      EnviaSerial_SR( valor );

      vetorEntradas[bitNum] = leEntradas();
   }

   EnviaSerial_SR( 0 );

   output_high(aciona_Pino25);
   vetorEntradas[24] = leEntradas();
   output_low(aciona_Pino25);

   output_high(aciona_Pino26);
   vetorEntradas[25] = leEntradas();
   output_low(aciona_Pino26);
}
```

Fonte: Do autor

A função da figura 60 serve para preencher um vetor “int32”, ou seja, cada posição do vetor pode armazenar uma variável com valor de até 32 bits, ela aciona uma saída e faz uma leitura para identificar qual entrada foi ligada, usando as funções do CI 4094 e 74LS2440 vistas anteriormente.

Figura 61 - Código fonte (parte 10).

```
unsigned int1 comparaVetores( unsigned int32 vetorEntradas[], unsigned int32 vetorAux[] ){
   unsigned int8
      index;

   for( index= 0; index < TOTAL_PINOS; index++ ){
      if( vetorEntradas[index] != vetorAux[index] )
         return False;
   }

   return true;
}
```

Fonte: Do autor

Na figura 61 se observa uma função que compara dois vetores, ela retorna apenas verdadeiro (valor 1) caso os vetores sejam iguais, ou seja, o valor da mesma posição dos dois vetores deve ser igual, ou falso (valor 0) caso algum valor não corresponder.

Figura 62 - Código fonte (parte 11)

```
unsigned int1 vetorTemValor( unsigned int32 vetor[] ){
   unsigned int8
      index;

   for( index= 0; index < TOTAL_PINOS; index++ ){
      if( vetor[index] )
         return True;
   }

   return False;

}
```

Fonte: Do autor

Na função da figura 62 acima mostra uma função que percorre todo o vetor verificando se em alguma posição tem algum valor diferente de zero, ou seja, verifica se o vetor não está zerado, ela retorna apenas verdadeiro (valor 1) caso o vetor tenha algum valor ou falso (valor 0) caso o vetor esteja vazio.

Figura 63 - Código fonte (parte 12)

```
unsigned int8 montaStrDisplay( unsigned int32 vetorEntradas[], unsigned int8 indexInicio,
                               char resultSaidas[], char resultEntradas[] ){
   unsigned int8
      index= indexInicio,
      lastIndex= 0,
      numBit,
      sizeResSaidas= 0,
      sizeResEntradas= 0,
      count= 0;


   while( (index < TOTAL_PINOS) && (count < PINOS_POR_PAGINA)  ){

      for( numBit= 0; numBit < TOTAL_PINOS; numBit++ ){

         if( bit_test(vetorEntradas[index], numBit) ){
            sizeResSaidas += sprintf( resultSaidas + sizeResSaidas, "%c ", tabelaCaracteres[index] );
            sizeResEntradas += sprintf( resultEntradas + sizeResEntradas, "%c ", tabelaCaracteres[numBit] );
            lastIndex= index;
            count++;
            break;
         }

      }

      index++;

   }
   if( index == TOTAL_PINOS )
      lastIndex= TOTAL_PINOS - 1;


   return lastIndex;

}
```

Fonte: Do autor

Na figura 63 acima se vê uma função para montar uma string[2] de parâmetros para posteriormente enviar ao display a informação.

Nela é testada cada entrada acionada formando a string para a linha de saídas acionadas e a linha de suas respectivas entradas, tendo como valores a sequência da tabela de caracteres declarada no início do programa.

Figura 64 - Código fonte (parte 13)

```
unsigned int8 showPage( unsigned int32 vetorEntradas[], unsigned int8 index ){
   static char strSaidas[LCD_MAX_CAR_LINHA + 1], strEntradas[LCD_MAX_CAR_LINHA + 1];
   unsigned int8 result;

   memset( strEntradas, 0, sizeof(strEntradas) );
   result= montaStrDisplay( vetorEntradas, index, strSaidas, strEntradas );

   if( strlen(strEntradas) ){
      strcpy( LCDBuffer, strSaidas );
      LCD_EscreveBuffer( True, 0x80 );
      strcpy( LCDBuffer, strEntradas );
      LCD_EscreveBuffer( False, 0xC0 );
   }

   return result;
}
```

Fonte: Do autor

Na figura 64 acima mostra a função que envia para o display a string de saída e entrada que foram montadas na função anterior.

Figura 65 - Código fonte (parte 14)

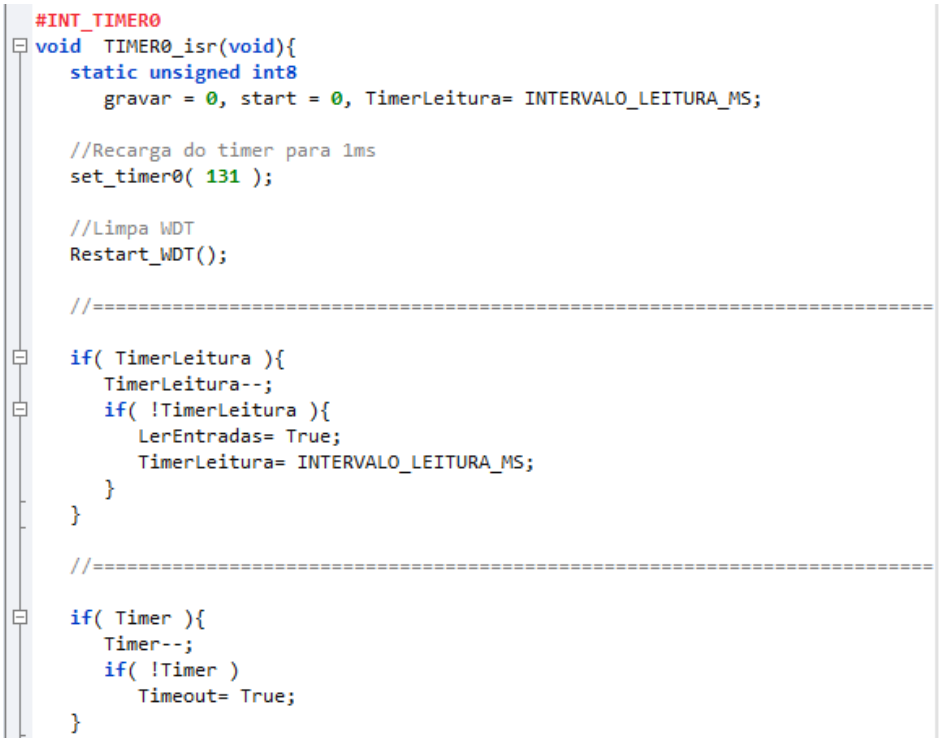
```
void TemporizaMs( unsigned int16 Valor ){

   Timer= Valor;
   Timeout= False;

}
```

Fonte: Do autor

A função da figura 65 funciona como um temporizador setando uma variável para falso (valor 0) caso o valor de entrada da função corresponda com um timer que veremos mais abaixo, ela substitui a função que o compilador oferece de delay, evitando que o microcontrolador fique “parado” em algum delay desnecessário, podendo continuar com a função de interrupção do timer que veremos abaixo.

[2] Por Eduardo Casavella: O termo string serve para identificar uma sequência de caracteres. Acesso pelo link: http://linguagemc.com.br/string-em-c-vetor-de-caracteres/

Figura 66 - Código fonte (parte 15)

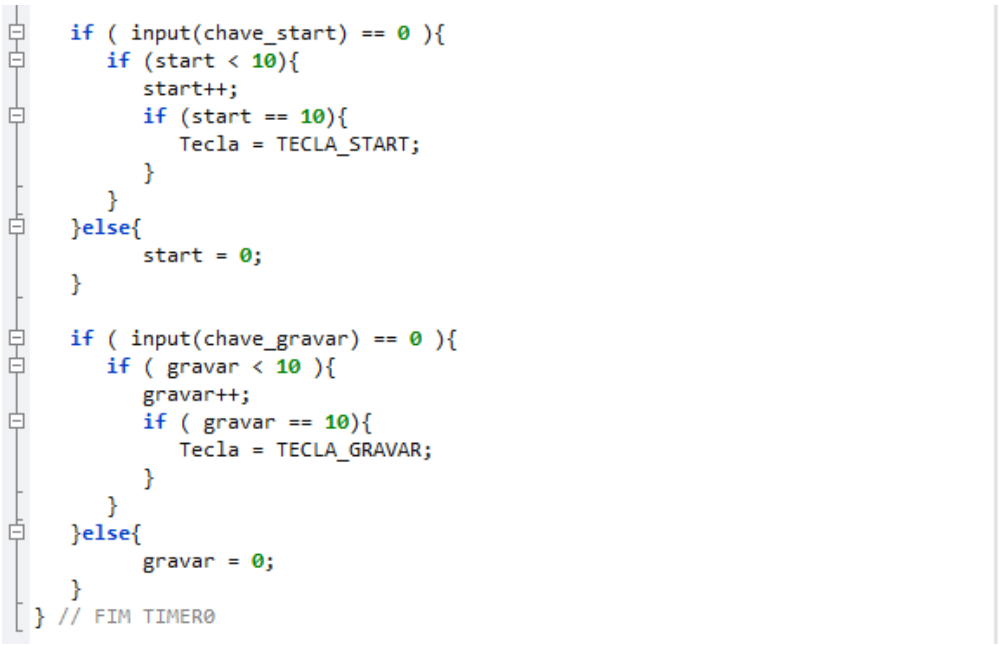


```
#INT_TIMER0
void  TIMER0_isr(void){
   static unsigned int8
      gravar = 0, start = 0, TimerLeitura= INTERVALO_LEITURA_MS;

   //Recarga do timer para 1ms
   set_timer0( 131 );

   //Limpa WDT
   Restart_WDT();

   //=============================================================================

   if( TimerLeitura ){
      TimerLeitura--;
      if( !TimerLeitura ){
         LerEntradas= True;
         TimerLeitura= INTERVALO_LEITURA_MS;
      }
   }

   //=============================================================================

   if( Timer ){
      Timer--;
      if( !Timer )
         Timeout= True;
   }
```

Fonte: Do autor

Conforme figura 66 temos o início da função de interrupção do TIMER 0 do microcontrolador, essa função ocorre a cada 1ms. Aqui temos breves funções que setam variáveis e flags com base na contagem do tempo do próprio timer.

Figura 67 - Código fonte (parte 16)



```
   if ( input(chave_start) == 0 ){
      if (start < 10){
         start++;
         if (start == 10){
            Tecla = TECLA_START;
         }
      }
   }else{
         start = 0;
   }

   if ( input(chave_gravar) == 0 ){
      if ( gravar < 10 ){
         gravar++;
         if ( gravar == 10){
            Tecla = TECLA_GRAVAR;
         }
      }
   }else{
         gravar = 0;
   }
} // FIM TIMER0
```

Fonte: Do autor

Já na figura 67 se observa o restante do TIMER 0, onde se pecebe o controle das chaves, setando as variáveis de acordo com cada chave pressionada e finalizando a função de interrupção do timer.

Figura 68 - Código fonte (parte 17)

```
void main()
{
   //.............................VARIÁVEIS LOCAIS.....................................//
   unsigned int8 index= 0, lastIndex= 0;
   unsigned int32 vetorEntradas[TOTAL_PINOS], vetor_ref[TOTAL_PINOS], vetor_aux[TOTAL_PINOS];


   setup_timer_0(RTCC_INTERNAL|RTCC_DIV_32|RTCC_8_BIT);

   init_pinos();

   delay_ms(250);
   LCD_Init(False); //False: cursor apagado

   strcpy( LCDBuffer, "CONECTE O CABO" );
   LCD_EscreveBuffer( True, 0x80 );
   strcpy( LCDBuffer, "DESEJADO" );
   LCD_EscreveBuffer( False, 0xC0 );

   enable_interrupts(INT_TIMER0);
   enable_interrupts(GLOBAL);

   memset( vetorEntradas, 0, sizeof(vetorEntradas) );
   memset( vetor_aux, 0, sizeof(vetor_aux) );
   memset( vetor_ref, 0, sizeof(vetor_ref) );
```

Fonte: Do autor

O código de baseia na função principal, chama de “main” como mostrado no início dela na figura 68 acima. São declaradas algumas variáveis locais, usadas apenas na “main” e configurados alguns requisitos, além de iniciar o display e os vetores que serão utilizados para a lógica.

Figura 69 - Código fonte (parte 18)

```
while(TRUE) { //TODO: User Code

   //Limpa WDT
   Restart_WDT();

   switch (Tecla){

      case TECLA_START:

         index= lastIndex + 1;
         if( index >= TOTAL_PINOS )
            index= 0;

         lastIndex= showPage( vetorEntradas, index );

         Tecla = TECLA_NONE;
         break;

      case TECLA_GRAVAR:

         memcpy( vetor_ref, vetorEntradas, sizeof(vetor_ref) );

         Tecla = TECLA_NONE;
         break;

   }// fim do switch
```

Fonte: Do autor

Dentro da função principal existe um laço infinito que é por meio dele que se garante o funcionamento da programação enquanto equipamento estiver ligado. Na figura 69 se observa a parte inicial desse laço que mostra o que o programa faz caso alguma chave tenha sido pressionada.

No programa a tecla “START” é usada para a navegação entre as páginas do display, facilitando a visualização dos pinos quando um cabo conectado no testador possuir mais de oito vias e a tecla “GRAVAR” serve para copiar o vetor recém lido para um vetor de referência que será usado para os próximos cabos da mesma instrução que deverão ser testados.

Figura 70 - Código fonte (parte 19)

```
      if( LerEntradas ){

         LerEntradas= False;

         atualizarVetor( vetorEntradas );

         if( comparaVetores(vetorEntradas, vetor_aux) ){
            if( !vetorTemValor(vetorEntradas) ){
               if( input(led_green) || input(led_red) ){
                  strcpy( LCDBuffer, "CONECTE O CABO" );
                  LCD_EscreveBuffer( True, 0x80 );
                  strcpy( LCDBuffer, "DESEJADO" );
                  LCD_EscreveBuffer( False, 0xC0 );
                  index= 0;
               }
               output_low( led_green );
               output_low( led_red );
            }else{
                  if( comparaVetores(vetorEntradas, vetor_ref) ){
                     output_low( led_red );
                     output_high( led_green );
                  }else{
                     output_low( led_green );
                     output_high( led_red );
                     }
                  }
            }else{
               lastIndex= showPage( vetorEntradas, index );
               memcpy( vetor_aux, vetorEntradas, sizeof(vetor_aux) );
            }
      }
   }//fim do loop principal
}//fim do main
```

Fonte: Do autor

Na figura 70 mostra o restante do código fonte, onde a lógica começa preenchendo o vetor de entradas para as condições se o vetor preenchido estiver vazio ou correto, tendo que para validar o cabo lido com o de referência uma quantidade de três vezes.

Exemplo descrito seria ao plugar um cabo qualquer o LED vermelho ligaria, pois ainda não foi verificado com a IT se está correto ou não, deveria ser verificado no display quais saídas acionaram entradas.

Podendo navegar pelas páginas ou não, validar o cabo pressionando a chave para gravar o cabo ligando o LED verde e apagando LED vermelho. Retirar o cabo do testador e colocar o próximo cabo onde automaticamente ligaria o LED verde para caso o cabo esteja correto ou vermelho para cabo incorreto.

Como não temos uma contagem de tempo não temos como garantir que sempre ao plugar um cabo para testar a leitura das entradas estará no pino 1 por esse

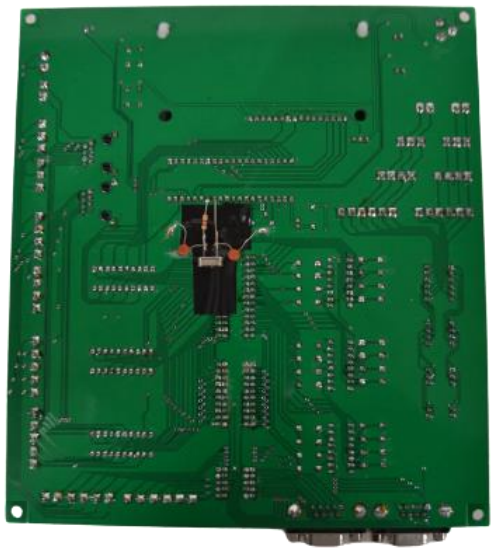
motivo é verificado que tem algum cabo plugado e só na terceira leitura do vetor que validamos com o referencial previamente gravado.

A flag “LerEntradas” troca de estado a cada 100ms pois foi definido pelo TIMER0, garantindo uma estabilidade na leitura e teste ao plugar o cabo nos conectores da IPYTST08.

5. CONCLUSÃO

Com o trabalho concluído até o momento se constatou que a placa IPYTST08 não iria ter um funcionamento, pois não foi pensado em um circuito oscilador para o microcontrolador. Por esse motivo realizou-se a colocação do cristal de 16MHz posteriormente conforme mostrado na figura 71 abaixo.

Figura 71 - Circuito com cristal oscilador de 16MHz



Fonte: Do autor

Alterações concluídas, pode-se validar o firmware de maneira satisfatória. O projeto foi submetido a uma fabricação de cem unidades e não apresentou falhas de funcionamento.

Os colaboradores da Imply® que usaram o testador de cabos qualificaram o mesmo como ótimo, pois facilitou o entendimento das instruções de trabalho e garantiu uma fabricação correta dos cabos testados até então.

Usaram o testador para uma produção de oito cabos de rede, trinta cabos de dados e dois cabos de alimentação, onde se observou eu não houve erros na fabricação, mesmo processo que há alguns meses antes tinham em torno de 70% a 80% de cabos feitos de maneira errada.

Destaca-se também que o projeto atendeu as necessidades da empresa, que tinha uma expectativa de melhora no seu processo produto de cabeamento, com o testador pode-se esperar que erros no cabeamento não devem mais ocorrer no setor de Montagem, como acontecia antes do projeto ser finalizado.

REFERÊNCIAS

FURNIEL, Igor. ISO 9001 – Sistema de Gestão da Qualidade. Disponível em: https://certificacaoiso.com.br/iso-9001/. Acesso em: 08/11/2021

GIMENEZ, Salvador P. Microcontroladores 8051: Teoria do hardware e do software. 1. ed. Rio de Janeiro: Pearson, 2002. 272 p.

IMPLY TECNOLOGIA LTDA, Disponível em: https://www.imply.com/pt/. Acesso em: 08/11/2021

TANENBAUM, Andrew S.; AUSTIN, Todd. Organização estruturada de computadores. 6. ed. Rio de Janeiro: Pearson, 2013. 460 p.

TOKHEIM, Roger L. Fundamentos de Eletrônica Digital: Sistemas Sequenciais. 1. ed. Porto Alegre: AMGH, 2013. 274 p.