UNIVERSIDADE SÃO FRANCISCO

CURSO DE ENGENHARIA ELÉTRICA

MONITORAMENTO DO CINTO DE SEGURANÇA EM ÔNIBUS DE VIAGEM

Área de Microcontroladores

por

André Salus Vicchini
RA: 002200400085

Antonio de Assis Bento Ribeiro, Msc.
Orientador

Itatiba (SP), novembro de 2008

UNIVERSIDADE SÃO FRANCISCO

CURSO DE ENGENHARIA ELÉTRICA

MONITORAMENTO DO CINTO DE SEGURANÇA EM ÔNIBUS DE VIAGEM

Área de Microcontroladores

por

André Salus Vicchini
RA: 002200400085

Relatório apresentado à Banca Examinadora do Trabalho de Conclusão do Curso de Engenharia Elétrica para análise e aprovação.
Orientador: Antonio de Assis Bento Ribeiro, Msc

Itatiba (SP), novembro de 2008

AGRADECIMENTOS

Agradeço a Deus e meus Pais por conseguir chegar até aqui hoje.

Agradeço este trabalho ao meu orientador Msc. Antonio de Assis Bento Ribeiro pela ajuda.

Agradeço aos meus amigos que participaram direta e indiretamente para a conclusão deste trabalho.

Agradeço a minha noiva pela motivação e colaboração principalmente neste período de fundamental dedicação ao projeto.

"Da vida só se leva, a vida que se leva"

Autor: Desconhecido

SUMÁRIO

LISTA DE ABREVIATURAS

PIC	Peripheral Interface Controller
PCI	Placa de Circuito Impresso
C.I.	Circuito Impresso

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE EQUAÇÕES

RESUMO

VICCHINI, André Salus. **Monitoramento do cinto de Segurança em Ônibus de Viagem.** Itatiba, 2008. Trabalho de Conclusão de Curso, Universidade São Francisco, Itatiba, 2008.

O cinto de segurança é um equipamento de proteção para as pessoas que utilizam veículos automotores. A não utilização deste equipamento de proteção em caso de acidente acarreta em danos severos as pessoas, com isso todos os ônibus obrigatoriamente fabricados a partir de 1999 saem de fábrica com cinto de segurança. A utilização do cinto de segurança é obrigatória em viagens intermunicipais e interestaduais, mesmo sendo obrigatórios, muitos passageiros não fazem o uso e ignoram a lei. O sistema de monitoramento do cinto de segurança será uma forma de minimizar esse problema. Este recurso permitirá que o motorista tenha em tempo real o controle do uso do cinto através de um display em seu painel. O sistema tem por finalidade bloquear a partida do ônibus enquanto todos os passageiros sentados não colocarem o cinto de segurança. Após a partida o sistema apenas irá monitorar as poltronas sem atuar no seu sistema de ignição, se por ventura algum passageiro retirar o cinto de segurança o sistema irá mostrar uma mensagem no display do motorista comunicando a falta do cinto de segurança em determinada poltrona.

Palavras-chave: cinto de segurança, ônibus, sistema de monitoramento.

ABSTRACT

VICCHINI, Andre Salus. Monitoring of safety belts in bus to travel. Itatiba, 2008. Completion of work of course, San Francisco University, Itatiba, 2008.

The safety belt is an instrument of protection for people who use motor vehicles. Not using this equipment for protection in case of accident can causes severe damage in people, because of that all bus manufactured from 1999 leave the factory with safety belt. The use of safety belts is mandatory in intermunicipal and interstate trips, even though many passengers do not use it and ignore the law. The monitoring system of safety belts will be a way to minimize this problem. This feature allows the driver to have a real-time control of the use of the belt through a display on his panel. The system has purpose of blocking the bus to start while all passengers seated don`t fasten their safety belt. After starting, the system will only monitor the seats not acting in its ignition system, if perchance any a passenger remove the safety belt, the system will display a message on the driver display communicating the lack of safety belts in a certain chair.

Keywords: safety belts, bus, monitoring system.

1. INTRODUÇÃO

Este trabalho destina-se a conscientização dos passageiros na utilização do cinto de segurança. Contribuindo com a lei, que muitas vezes não é cumprida neste caso por negligência dos passageiros.

O cinto de segurança é um dispositivo simples que serve para proteger vidas e diminuir as conseqüências dos acidentes. Ele impede, em casos de colisão, que o corpo se choque contra o volante, painel e pára-brisas, ou que seja projetado para fora do veiculo. Mas não são só os motoristas que devem utilizá-lo, (o cinto de segurança) mas também deve usado por passageiros, até mesmo quando ocupam o banco traseiro dos veículos. (BECKER, on line)

O Código Nacional de Trânsito obriga o uso do cinto também em ônibus.

A Resolução 14, de 1998, diz que todos os ônibus fabricados a partir de 1999 devem "sair de fábrica" com o equipamento de segurança.

Em caso do motorista não utilizar o equipamento, poderá ser punido com a perda de pontos na carteira de habilitação e a empresa pode ser multada.

A legislação determina que as empresas de ônibus orientem os passageiros sobre a importância do cinto de segurança, podendo ser através de folhetos, vídeos ou ainda por intermédio dos funcionários da empresa.

Usar cinto de segurança em ônibus intermunicipais e interestaduais é lei segundo o Código de Trânsito Brasileiro (CTB), ou seja, quem viaja de ônibus de uma cidade para outra ou de um estado para outro necessita usá-lo.

O artigo 105 do CTB diz que só os passageiros que usam ônibus dentro do município, de um bairro para o outro, não precisam usar o equipamento porque nesses casos a lei permite que viajem "em pé". (BECKER, on line)

As empresas de ônibus intermunicipais e interestaduais cumprem o que diz a lei, mas muitos passageiros ainda ignoram a obrigatoriedade.

As empresas que permitem que os passageiros viajem sem cinto de segurança podem ser punidas com multa.

Segundo a Associação Brasileira de Ortopedia e Traumatologia (ABOT) em caso de acidente o cinto evita que o passageiro choque a cabeça na poltrona da frente, o que pode fazê-lo "perder os sentidos". (ADURA, 2004)

Sabe-se que sem o cinto de segurança a chance de se machucar em um acidente são 4 vezes maior. Oito em cada 10 pessoas que não usavam o cinto de segurança morreram em acidentes com pelo menos um dos veículos a menos de 20 km/h. (BECKER, on line)

Um impacto a 50 km/h produz um poder de destruição igual ao de uma pessoa caindo de cabeça para baixo de uma altura equivalente ao segundo andar de um edifício.

Com tudo isso ainda as pessoas continuam a não fazer o uso adequado do cinto de segurança de maneira a proteger sua vida.

1.1. OBJETIVOS

1.1.1. Objetivo Geral

Tento em vista ao alto número de pessoas que se machucam ou até morrem em acidentes de ônibus de viagem devido à falta ou utilização inadequada do cinto de segurança, motivou-me desenvolver um sistema para monitorar o uso adequado deste equipamento em passageiros que viajam de ônibus. O projeto descrito nos próximos parágrafos visa assim minimizar possíveis politraumatismos que possam ocorrer aos passageiros em caso de acidente. Sendo um instrumento a mais para se fazer cumprir a lei, pois ele além de monitorar ele não libera o sistema de partida enquanto todos os passageiros estejam em seus devidos lugares e acima de tudo com segurança.

1.1.2. Objetivos Específicos

Desenvolver um circuito eletrônico responsável por reconhecer automaticamente a presença de um passageiro em sua poltrona, comunicando ao motorista sobre o uso ou não do cinto de segurança através de um display no painel do ônibus, possibilitando:

- Uma viagem mais segura;
- Diminuindo a estatística de morte no transito.

1.2. METODOLOGIA

O Será realizado levantamento bibliográfico, sobre o assunto abordado e confecção do projeto, seguindo as seguintes etapas.

1. Identificação do problema

2. Estudo do Microcontrolador PIC 16F877A

3. Elaboração programa principal do projeto

4. Teste via software

5. Estudo sobre Multiplexação de sinais

6. Teste via software

7. Programação do microcontrolador

8. Desenho da placa de circuito impresso

9. Confecção e montagem da placa de circuito impresso

10. Teste via software

11. Teste via hardware

12. Montagem da maquete

1.3. ESTRUTURA DO TRABALHO

Será explorado no inicio do trabalho a teoria sobre o microcontrolador PIC 16F877A, que exigiu uma grande dedicação em sua pesquisa. A estrutura do trabalho seguirá conforme descrito na metodologia.

2. FUNDAMENTAÇÃO TEÓRICA

O projeto aborda o estudo do microcontrolador PIC16F877A, e por tanto, faz necessário em primeiro momento o estudo de suas características, funções e outros aspectos envolvidos.

2.1. IDENTIFICAÇÃO DO PROBLEMA

Foi observado por diversas vezes os noticiários relatarem acidentes envolvendo ônibus de viagem e na maioria dos casos com mortes fatais. Relatos de pessoas que se salvaram deste acidentes indicam o cinto de segurança como sua existência hoje, isto comprova a eficiência do cinto de segurança quando utilizado adequadamente.

É de fundamental importância a conscientização das pessoas, quanto à utilização do cinto de segurança para salvar sua vida.

Será de grande valia a possível implementação desse sistema o mais breve, em ônibus de viagem.

2.2. ESTUDO DO MICROCONTROLADOR PIC16F877A

Para elaborar o projeto em questão, foi estudado o microcontrolador PIC16F877A, pois a escolha deste microcontrolador se deve ao fato que ele disponibiliza ao programador um grande número de periféricos para serem acoplados, devido ao seu grande numero de pinos 40.

2.2.1. Origem do PIC

O PIC16F877A é um microcontrolador que teve suas origens no ano de 1965, quando a companhia GI (General Instruments), formou a Divisão de Microeletrônica. Esta divisão foi uma das primeiras a produzir arquiteturas de EPROM e EEPROM. (SOLBET, on line)

Na década de 70, a GI criou um dos primeiros processadores de 16 bits, chamado CP16000, como este microprocessador tinha certa deficiência no processamento de entradas/saídas, a GI projetou e construiu um Controlador de Interface Periférica (em inglês, Peripheral Interface Controller, ou PIC). Este controlador foi projetado tendo em vista a rapidez, pois devia processar as E/S de uma máquina de 16 bits e tinha um conjunto de instruções muito pequeno. (SOLBET, on line)

O CP16000 não teve muito sucesso, porém o PIC evoluiu para a arquitetura PIC16C5X. Como era disponível somente nas versões em ROM, permaneceu como uma boa solução para grandes usuários, que podiam encomendar diretamente da fábrica um grande volume de circuitos já pré-programados. (SOLBET, on line)

Na década de 80, a divisão de microeletrônica da GI foi reestruturada e se transformou na GI Microeletrônica. Esta empresa foi vendida para investidores e transformada na Arizona Microchip Technology, dedicada especificamente ao desenvolvimento de produtos utilizados em sistemas dedicados. Como parte desta estratégia, grande esforço da companhia foi direcionado ao desenvolvimento de várias versões do PIC, sendo que a versão com EEPROM (eprom apagável eletricamente). O dispositivo mais flexível disponível hoje da série PIC de 16 bits e que ao mesmo tempo reúne um conjunto de características adequadas ao uso em pequenas séries de produtos e estudo de microcontroladores, é o PIC16F877A. Como o microcontrolador PIC16F877A se utiliza da arquitetura Harvard, isto possibilita que as palavras de instruções tenham um número de bits (14 bits de comprimento) diferente do tamanho da palavra de dados (8 bits para este microcontrolador). Com este tamanho de palavra de instrução é possível codificar todas as instruções, com exceção dos desvios para outras posições de programas, como instruções de uma única palavra, resultando em grande velocidade de execução, 400 ns para a versão de 10Mhz. Estes dispositivos podem endereçar direta e indiretamente seus arquivos de registros ou memória de dados. O conjunto de instruções foi projetado de tal forma que se pode realizar qualquer operação em qualquer registro utilizando qualquer modo de endereçamento. A unidade lógica aritmética do PIC16F877 pode realizar operações de adição, subtração, deslocamento e operações lógicas. (SOLBET, on line)

2.2.2. Arquitetura do Microcontrolador do PIC16F877A

Podemos dizer que o microcontrolador basicamente é um dispositivo dividido em partes, como por exemplo: Clock, CPU, Portas de Entrada/ Saída, Interrupções, Memória de Dados, Memória de Programa. (GUITIERRES, on line)

2.2.2.1. Clock: é responsável pelo sincronismo entre todas as operações do microcontrolador. Todas as funções que ocorrem dentro de um microcontrolador obedecem a uma lógica preestabelecida pelo fabricante e é processado em tempos determinados pela freqüência do clock, no caso o microcontrolador PIC 16F877A pode usar um cristal de até 20 MHz sem problemas, mais usualmente empregado é de 4 Mhz.

2.2.2.2. CPU: Unidade Central de Processamento, ela controla e coordena todos os eventos dentro do microcontrolador, organiza a execução das instruções, realiza as operações lógicas ou matemáticas sobre os dados, e envia os resultados para as portas de entrada e saída ou os diversos registros.

2.2.2.3. Portas de Entrada/ Saída: pontos através dos quais o microcontrolador se comunica com o meio externo. Portas digitais adquirem valores normalmente referenciados como 0 ou 1, que correspondentes a 0 ou 5 volts respectivamente.

2.2.2.4. Interrupção: Também são portas, só que especiais, pois estas portas não exigem que a CPU fique "monitorando" a ocorrência de algum evento, pois interrompem o programa sendo executada toda vez que isso ocorre, desta forma a CPU pode atender o evento externo assim que o mesmo ocorrer e não gasta tempo monitorando algum evento na porta, pois essa característica é utilizada em projetos que se necessita de uma rapidez de resposta a eventos assíncronos e que podem ocorrer a qualquer instante.

2.2.2.5. Memória de Dados: implementada na forma de memória RAM (memória de acesso aleatório) em que podemos gravar e ler facilmente. A leitura é não destrutiva significa que podemos ler muitas vezes o valor de uma posição de memória que esse valor não muda. A gravação de um dado na memória RAM é executada pela CPU. Os Microcontroladores PIC costumam utilizar de pequenas quantidades de memória de dados, pois se compararmos com as quantidades de memória RAM utilizadas por computadores pessoais de hoje, algo como 100 bytes para os microcontroladores, versus 2048 milhões de bytes (2Gbytes) de um típico computador pessoal.

2.2.2.6. Memória de Programas: é onde se encontram as instruções que devem ser executadas pela CPU. A maioria dos microcontroladores se utiliza de memória EEPROM, que para ser apagada necessita de ser exposta a luz ultravioleta. Os microcontroladores, tais como o PIC16F877A, substituíram a EPROM pela EEPROM, memória que pode ser apagada por meios elétricos, sem necessidade de apagamento por luz ultravioleta. Esta forma de trabalho encurta significativamente o ciclo de desenvolvimento do programa, e também não exige equipamento sofisticado para gravação e apagamento da memória de programas.

A memória de programas e a memória de dados podem ou não compartilhar de um único espaço de endereçamento. A arquitetura *Von Neuman* possui um único barramento de memória para endereçar as instruções e os dados. Dados e instruções, portanto, compartilham do mesmo espaço

de memória. Quando este tipo de microcontrolador executa uma instrução, ele primeiro pega a instrução e depois pega os dados referentes a esta instrução. Esta necessidade de dois acessos à memória torna a operação mais lenta, já os microcontroladores baseados na arquitetura *Harward* possuem um espaço de memória separado para os dados e para as instruções, pois isto permite que a leitura das instruções e dos dados ocorra em paralelo. A execução utilizando esta arquitetura pode ser muito mais rápida que a possível com a arquitetura *Von Neuman* e por isto aumenta a complexidade dos circuitos. O microcontrolador PIC16F877A utiliza da arquitetura *Harward*, o que permite uma alta velocidade de operação. (GUITIERRES, on line)

Os microcontroladores PIC16F877A são máquinas RISC, pois isto significa que são máquinas com um número de instruções reduzidas e para ser exato, são apenas 35 instruções (Tabela 1.1.) cada uma ocupando uma palavra (14 bits).

O PIC16F877A possui as seguintes características básicas:

- Conversor A/D
- Gerador PWM
- Barramento serial I²C (Inter-Integreted Circuit)
- Instruções de 14 bits.
- Dados em 8 bits.
- Endereçamento nos modos direto, indireto e relativo.
- Programa gravado em EEPROM, com até 1000000 de ciclos de apagamento e escrita, com retenção garantida por mais de 40 anos.
- Trinta e três terminais de Entrada e Saída, com controle individual por terminal.
- Capacidade de manipulação de corrente de 25 mA atuando como dreno e 20mA atuando como fonte.
- Dois temporizadores de oito bits programável, com pré-divisor também programável de oito bits.

- Um temporizador de 16 bits.

- Sistema de proteção de código na EEPROM. Impossibilita que outras pessoas leiam o seu código.

- Operação em tensões desde 2 a 6 Volts, com consumo de corrente típico em torno de 2 mA.

- Memória RAM interna de 368 bytes, juntamente com uma área de EEPROM para dados de 256 bytes.

2.2.3. Arquitetura Interna do Microcontrolador do PIC16F877A

O PIC 16F877A, possui um conversor A/D com 8 canais selecionáveis de entrada de tensão, e uma resolução de 10 bits, as tensões de referência, pode ser obtidas a partir da combinação entre VDD, VSS, e tensões externas aplicadas nos pinos RA2 e RA3. A técnica de conversão que o PIC utiliza é conhecida como aproximações sucessivas. O módulo conversor A/D possui quatro registradores associados, sendo: ADCON0, ADCON1, ADRESH e ADRESL. (ZANCO, 2006)

Três temporizadores, TIMER0, TIMER1 e TIMER2.

O TIMER0 módulo temporizador / contador tem as seguintes características: 8-bit timer / contador, leitura e escrita, 8-bit programável software, seleção de relógio interno ou externo, interrupção a partir de overflow sobre a FFH 00h e borda para selecionar relógio externo. (ZANCO, 2006)

O TIMER1 módulo é um 16-bit temporizador/ contador que consiste em 2 registradores de 8bits (TMR1H e TMR1L), que são leitura e escrita e pode operar em um de dois modos: como um temporizador ou como um contador. (ZANCO, 2006)

TIMER2 é um temporizador com 8bits e pode ser usado como a base para o tempo PWM

O circuito em forma de diagrama de blocos pode ser visto na figura 1.

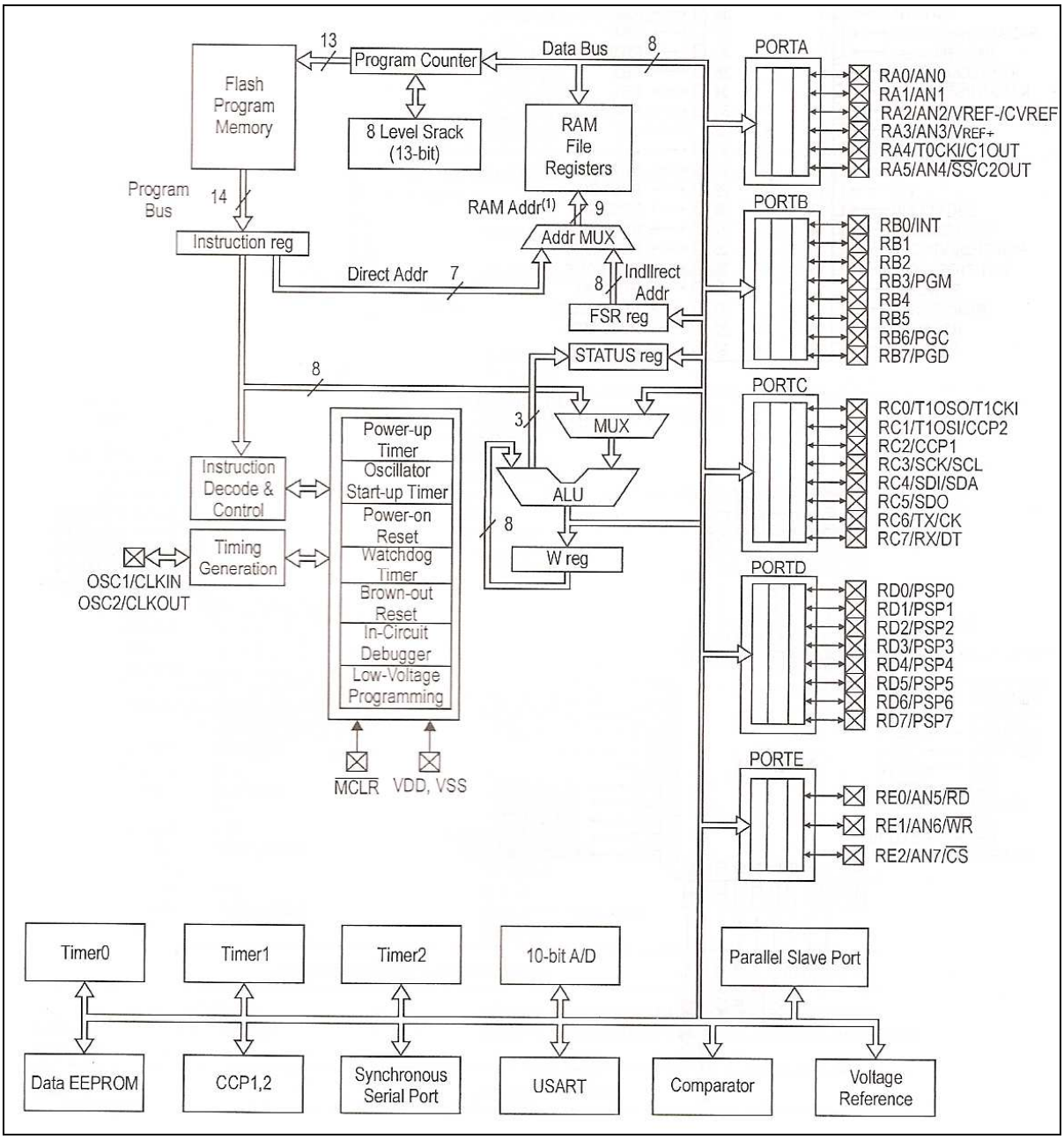


Figura 1. - Diagrama de blocos PIC 16F877A.
(Fonte: ZANCO,2006)

2.2.4. Mapeamento de Memória RAM do PIC16F877A

Observe na figura 2. que a memória de programa é organizada em quatro bancos.

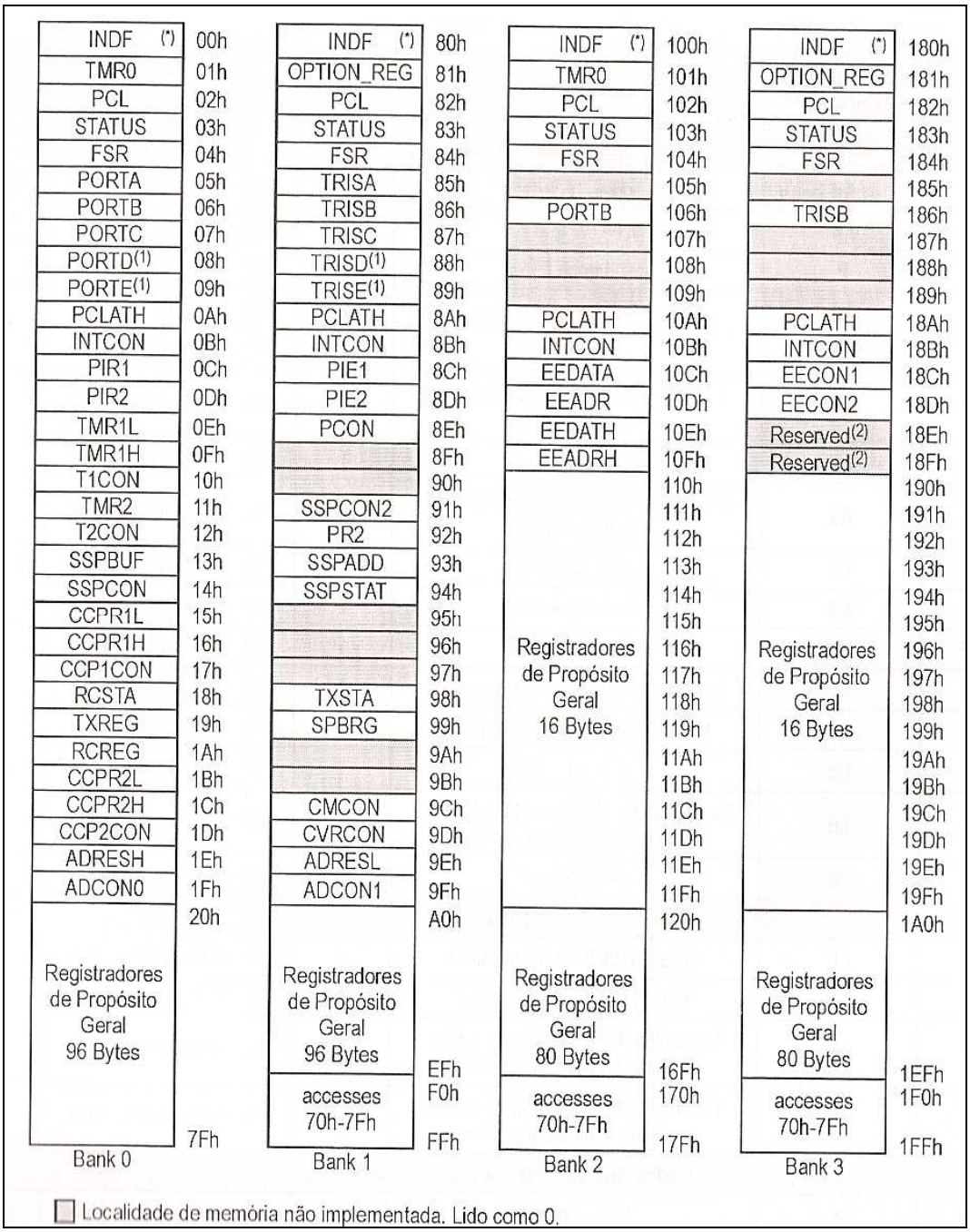


Figura 2. - Mapeamento da memória interna PIC 16F877A.
(Fonte: ZANCO, 2006)

2.2.5. Set de Instruções do PIC16F877A

Tabela 1. - As 35 instruções do PIC 16F877A

Mnemônicos	Operando ou Argumento	Descrição	Ciclos	14 bits de código MSB LSB	Status Afetado
Operações com registradores					
ADDWF	f,d	d ← (W+f)	1	00 0111 dfff ffff	C,DC,Z
ANDWF	f,d	d ← (W and f)	1	00 0101 dfff ffff	Z

CLRF	F	Limpa f	1	00 0001 1fff ffff	Z
COMF	f,d	d ← complemento de f	1	00 1001 dfff ffff	Z
DECF	f,d	d (f – 1)	1	00 0011 dfff ffff	Z
DECFSZ	f,d	d ← (f – 1) e salta próxima linha se resultado for zero	1(2)	00 1011 dfff ffff	
INCF	f,d	d (f + 1)	1	00 1010 dfff ffff	Z
INCFSZ	f,d	d ← (f+1) e salta próxima linha se resultado for zero	1(2)	00 1111 dfff ffff	
IORWF	f,d	d ← (W ou f)	1	00 0100 dfff ffff	Z
MOVF	f,d	d ← cópia de f	1	00 1000 dfff ffff	Z
MOVWF	F	f ← cópia de W	1	00 0000 1fff ffff	
RLF	F,d	Rotaciona f um bit para a esquerda	1	00 1101 dfff ffff	C
RRF	f,d	Rotaciona f um bit para a direita	1	00 1100 dfff ffff	C
SUBWF	f,d	d ←(f – W)	1	00 0010 dfff ffff	C,DC,Z
SWAPF	f,d	Inverte nibble alto com nibble baixo e guarda resultado em d	1	00 1110 dfff ffff	
XORWF	f,d	d ← (W xor f)	1	00 0110 dfff ffff	Z
Operações com bits					
BCF	f,b	Clear (0) bit b do registrador f	1	01 00bb bfff ffff	
BSF	f,b	Set (1) bit b do registrador f	1	01 01bb bfff ffff	
BTFSC	f,b	Testa bit b do registrador f e salta a próxima linha se ele for zero	1	01 10bb bfff ffff	
BTFSS	f,b	Testa bit b do registrador f e salta a próxima linha se ele for um	1	01 11bb bfff ffff	
Operações com literais (valores numéricos)					
ADDLW	K	W ← (W + K)	1	11 111x kkkk kkkk	C,DC,Z
ANDLW	K	W ← (W and K)	1	11 1001 kkkk kkkk	Z
IORLW	K	W ← (W ou K)	1	11 1000 kkkk kkkk	Z
MOVLW	K	W ← K	1	11 00xx kkkk kkkk	C,DC,Z
SUBLW	K	W ← (K – W)	1	11 110x kkkk kkkk	
XORLW	K	W ← (W xor K)	1	11 1010 kkkk kkkk	Z
Operações de controle					
CLRW	-	Limpa Work	1	00 0001 0000 0011	Z
NOP	-	Não faz nada, apenas gasta tempo	1	00 0000 0xx0 0000	
CALL	K	Chamada à sub-rotina	2	10 0kkk kkkk kkkk	
CLRWDT	-	Limpa WDT	1	00 0000 0110 0100	/TO,/PD
GOTO	K	Desvio para endereço	2	10 1kkk kkkk kkkk	
RETFIE	-	Retorno de interrupção	2	00 0000 0000 1001	

RETLW	K	Retorno de sub-rotina com K em W	2	11 01kk kkkk kkkk	
RETURN	-	Retorno de sub-rotina	2	00 0000 0000 1000	
SLEEP	-	Coloca PIC em modo Sleep para economia de energia	2	00 0000 0110 0011	/TO,/PD

2.3. ESTUDO SOBRE MULTIPLEXAÇÃO DE SINAIS

Para elaborar o projeto em questão, foi realizado uma pesquisa sobre Multiplexação de sinais devido à existência de um número elevado de entradas de sinais, gerando assim certo interesse em diminuir essas entradas para otimizar o projeto.

2.3.1. Definição de Multiplexação

O Mux ou multiplexador é um circuito combinacional dedicado com a finalidade de selecionar através de variáveis de seleção uma de suas entradas conectando-a eletronicamente a uma única saída, esta operação é denominada multiplex ou Multiplexação. Tanto as entradas como a saída são denominadas também de canais de entrada e saída, como exemplo pode citar, quando se escolhe um canal de televisão através do controle remoto se efetua na verdade uma seleção entre as várias emissoras existentes, as emissoras correspondem às entradas e a tela de TV à saída e o controle remoto faz a função do Mux.

Sendo um circuito que seleciona um dos diversos sinais na entrada e o transfere para a saída, esta seleção é feita através de entradas denominadas de seletoras. A Figura 3 abaixo mostra o esquema geral de um multiplexador e podemos observar que apenas uma entrada será selecionada para a saída, através da habilitação das entradas seletoras.

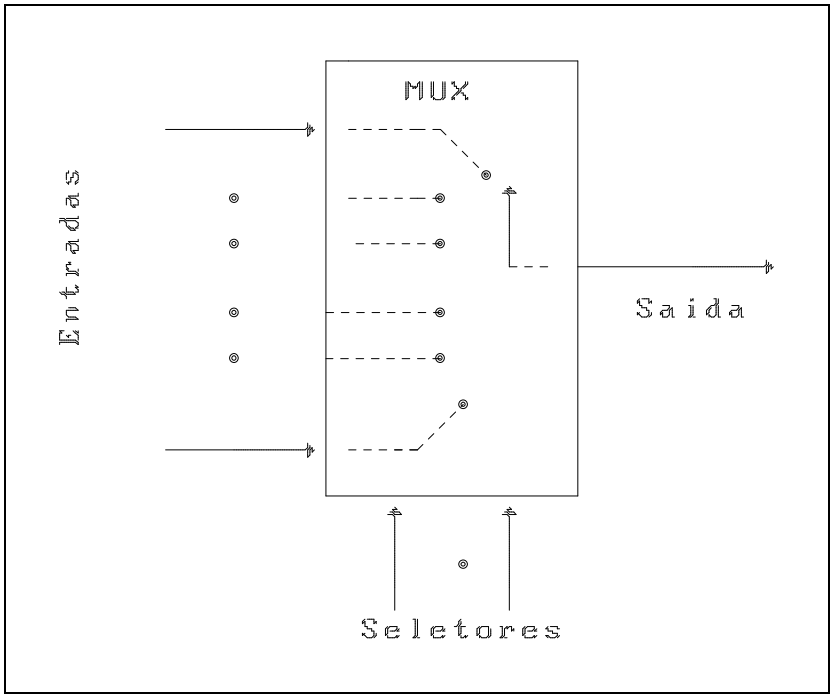


Figura 3. – Esquema geral de um multiplexador

De forma geral podemos escrever a relação entre o número de entradas e o número de seletores, supondo que o multiplexador possua N entradas, será necessário M entradas seletoras de acordo com a relação.

$I_0 - I_{N-1} =$ N entradas;

$S_0 - S_{M-1} =$ M entradas seletoras;

$N = 2^M$

Existem diversos tipos de multiplexadores classificados pelo número de entradas.

Por exemplo, um Mux de dois canais ou entradas precisa de apenas uma variável de seleção, pois:

$$\boxed{N=2^M=2^1=2}$$

Equação 1. – Calcular quantas entradas

Como a seleção das entradas não depende do nível lógico das mesmas, a tabela-verdade que representa o funcionamento deste multiplexador deve ter na mesma coluna da saída, ao invés de níveis lógicos, o nome das variáveis de entrada:

Tabela 2. – Tabela verdade Mux 2 canais

S0	**O**
0	**I0**
1	**I1**

Conforme citado, através do valor de $\mathbf{S}_0$, podemos disponibilizar na saída ou a entrada $\mathbf{I}_0$ ou a entrada $\mathbf{I}_1$.

Expressão booleana da saída:

$$O = \overline{S}_0 . I_0 + S_0 . I_0$$

O circuito que implementa o multiplexador de duas entradas é mostrado na Figura 4.

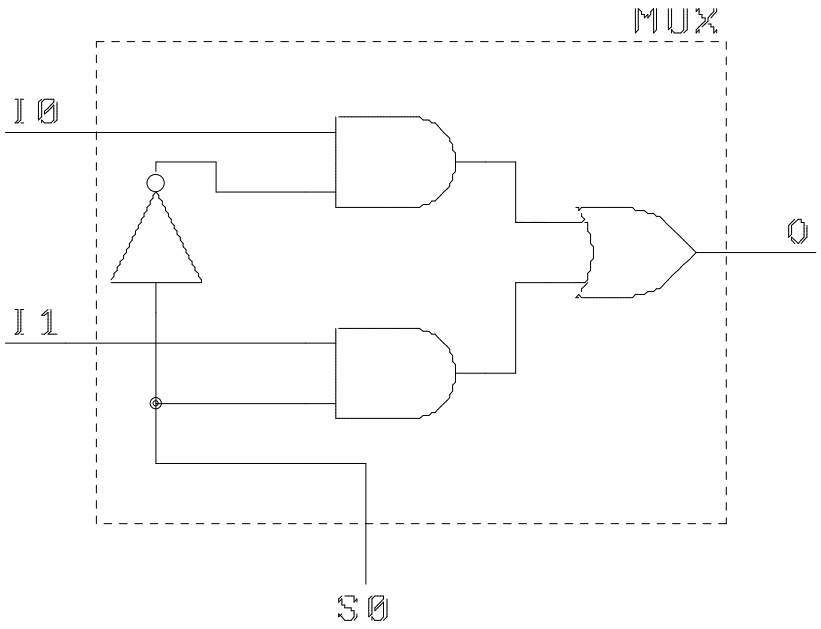


Figura 4. – Circuito multiplexador de duas entradas.

Cabe observar que os índices das entradas representam no sistema decimal os códigos das variáveis de seleção correspondentes no sistema binário e, portanto, é mais importante sempre destacar qual variável é a mais significativa (MSB) e qual é a menos significativa (LSB).

Um Multiplexador de quatro canais ou entradas precisa de duas variáveis de seleção, pois:

$$N = 2^M = 2^2 = 4$$

Portanto, a tabela verdade do Multiplexador é mostrada abaixo.

Tabela 3. – Tabela verdade Mux 4 canais

$\mathbf{S_1}$	$\mathbf{S_0}$	**O**
0	0	$\mathbf{I_0}$
0	1	$\mathbf{I_1}$
1	0	$\mathbf{I_2}$
1	1	$\mathbf{I_3}$

Expressão booleana da saída:

$$O = \overline{S}_1 . \overline{S}_0 . I_0 + \overline{S}_1 . S_0 . I_1 + S_1 . \overline{S}_0 . I_2 + S_1 . S_0 . I_3$$

O circuito que implementa o multiplexador de duas entradas é mostrado na Figura 5.

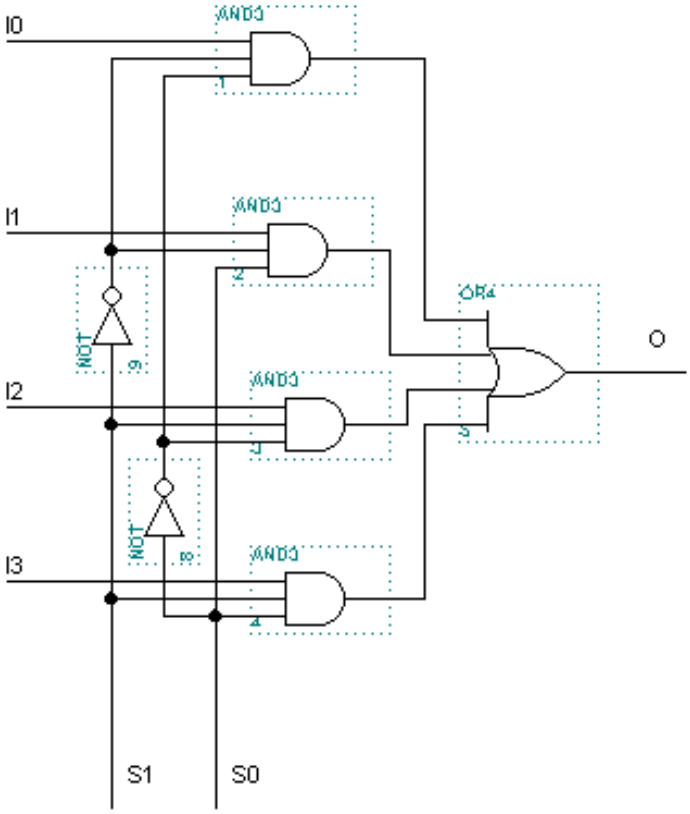


Figura 5. – Circuito do multiplexador de quatro entradas.

A tabela-verdade do multiplexador de oito entradas é mostrada abaixo. Neste caso são necessárias três entradas seletoras para disponibilizar o dado na saída.

Tabela 4. – Tabela verdade Mux 8 canais

S_2	S_1	S_0	O
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7

A representação, a tabela verdade e o processo para o projeto de multiplexadores de dezesseis canais é similar ao de oito canais, incrementando-se o número de entradas e o número de variáveis de seleção (A, B, C, e D).

2.3.2. Associação de Multiplexadores

Os multiplexadores podem ser encontrados prontos em circuitos integrados comerciais, mas o número de entradas é limitado em cada CI. Quando se necessita de um Multiplexador com uma quantidade de canais de entrada maior do que os encontrados comercialmente ou quando é necessário multiplexar vários canais simultaneamente, basta fazer a associação conveniente de vários multiplexadores de forma a ampliar o número de canais de entrada para uma única saída ou ampliar o número de saídas para se obter mais de um canal de entrada ativo simultaneamente.

2.3.2.1. Associação paralela de multiplexadores: Esta associação é importante quando se necessita selecionar informações digitais de vários bits simultaneamente. Para isto, basta utilizar um Multiplexador com um número de canais de entrada igual ao número de informações a serem multiplexadas sendo o número de Mux’s igual ao número de bits destas informações.

2.3.2.2. Associação série de multiplexadores: Esta associação é uma ampliação da capacidade dos canais de entrada, consiste em uma variação da associação paralela pois, para ampliar a capacidade de canais de entrada, basta multiplexar os Multiplexadores de entrada através de um Multiplexador de saída.

3. PROJETO

Com o aumento do número de acidentes de trânsito envolvendo ônibus de viagem, pensou-se em um projeto que ajude a cumprir a lei quanto à obrigatoriedade do uso de cinto de segurança. Sabe-se que a partir de 1999 todos os ônibus de viagem fabricados já saem com o cinto de segurança, mas em contra partida, na maioria das vezes o passageiro faz negligência quanto ao uso, acha que este equipamento de proteção incômoda ou simplesmente não vêem importância do uso do cinto de segurança em ônibus.

Este projeto tem por finalidade monitorar as poltronas verificando se há passageiros sentados sem cinto de segurança e bloquear a partida do ônibus em primeiro momento, pois somente após todos os passageiros se acomodarem e colocarem o cinto de segurança o sistema é liberado para a partida do motor sendo assim por diante apenas o monitoramento das poltronas que retirarem o cinto de segurança fornecendo esta informação para o motorista através do LCD no painel.

O hardware desenvolvido para executar essa função será mostrado na figura 6. Ao centro o maior C.I. é o PIC 16F877A, os dois mais abaixo são os C.I.'s multiplexadores 74LS150 e logo mais abaixo as chaves DIP's que simulam as poltronas.

Figura 6. – Hardware do projeto

Para um entendimento melhor sobre como funciona o sistema, abaixo será mostrado a figura 7 do fluxograma.

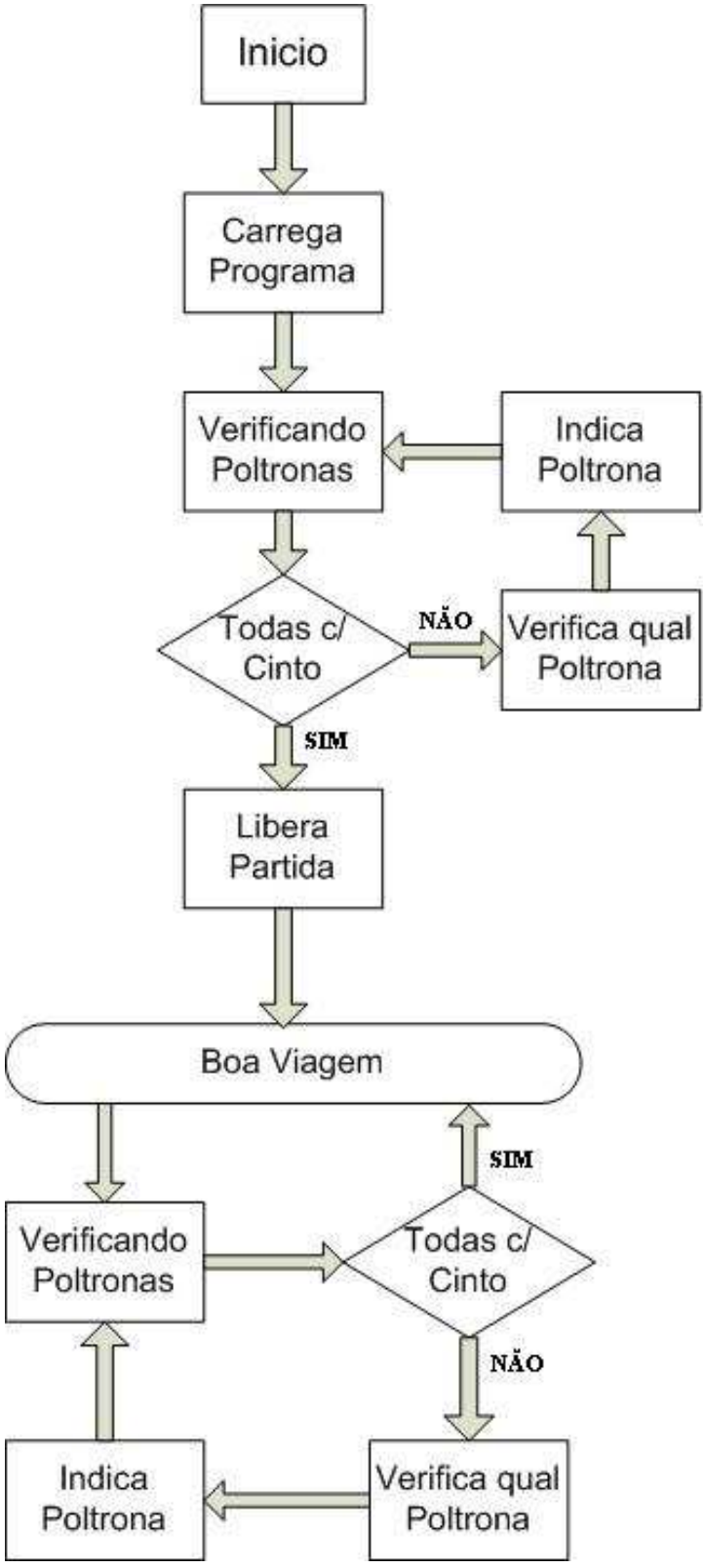


Figura 7. – Fluxograma de funcionamento do programa

O fluxograma acima mostra que a partir do momento em que o motorista girar a chave do ônibus no primeiro estágio onde acende todas as luzes do painel, o sistema de segurança começará a atuar.

No primeiro momento o microprocessador carrega o programa. (mesmo que o motorista tente dar a partida, o motor do ônibus não liga devido o sistema bloquear a ignição).

Em seguida o sistema começa a varrer (verificar) as poltronas do ônibus;

Esta tudo ok?

Se sim, está tudo ok;

O sistema de ignição é liberado para a partida do motor.

Se não, está tudo ok!

O sistema identifica qual a poltrona;

Indica a poltrona para o motorista através do LCD;

Volta a verificar as poltronas novamente;

Isso é feito enquanto todos os passageiros, não colocarem o cinto de segurança para a partida do ônibus.

Assim que é liberada a partida para uma boa viagem;

O programa se desloca para a rotina de monitoramento e não influencia mais no funcionamento do ônibus (bloqueando a ignição).

Sendo o monitoramento igual ao de liberação da ignição;

Verificando as poltronas o tempo todo;

E indicando, se necessário ao motorista através do LCD.

3.1. ENTENDENDO O PROGRAMA

Para um melhor entendimento do programa, será separado e explicado por partes.

3.1.1. Configurando o LCD

```
'////////////////// Configura o LCD para usar no pic16F877A ////////////////////////////
ADCON1 = 7 'deixa as entradas em modo digital                                        '//
define LCD_dreg portb                                                               '//
define lcd_dbit 4                                                                   '//
                                                                                    '//
define lcd_rsreg porta                                                              '//
define lcd_rsbit 4                                                                  '//
                                                                                    '//
define lcd_ereg porta                                                               '//
define lcd_ebit 5                                                                   '//
                                                                                    '//
define lcd_bits 4                                                                   '//
define lcd_lines 2                                                                  '//
'////////////////////////////////////////////////////////////////////////////////////////
```

Figura 8. – Configuração do LCD

Para compreendermos a configuração do LCD, precisamos primeiro entender o que é o **define**, como mostra na figura 8 acima.

Define: É a definição de alguns elementos como, por exemplo, a freqüência de clock do oscilador e as posições dos pinos do LCD, pois já são predefinidas. O define permite que um programador altere estas definições assim que desejado.

Quando a utilização do LCD é de fundamental importância essa definição de maneira que o programador, deixe o mais claro possível a sua configuração, pois pode acarretar em mau funcionamento do equipamento.

Agora vamos entender o que significa cada *define* utilizado no programa para fazer funcionar o LCD:

- *define* lcd_dreg portb: fixa a porta de dados do LCD;
- *define* lcd_dbit 4: fixa bit de dados inicial (0 a 4);
- *define* lcd_rsreg porta: fixa a porta de seleção do registrador do LCD;
- *define* lcd_rsbit 4: fixa o bit de seleção do registrador do LCD;

- *define* lcd_ereg porta: fixa a porta de habilitação do LCD;
- *define* lcd_ebit 5: fixa o bit de habilitação do LCD;
- *define* lcd_bits 4: fixa tamanho do barramento do LCD (4 ou 8bits);
- *define* lcd_lines 2: fixa o numero de linhas do LCD;

Estes *define* acima configuram o LCD de maneira a funcionar com: o PortB de dados, com 4 bits de dados, PortA de seleção do registrador, com 4 bits de seleção do registrador, com PortA de habilitação, com tamanho do barramento de 4 bits e com duas linhas do LCD.

3.1.2. Definição das Variáveis Utilizadas

```
'///////////////// Definição das variaveis utilizadas ////////////////////////////////////
z var byte
x var byte
x=0
z=0

'////////////// Definição dos pinos de entrada de sinais ///////////////////////////////
multi_1 var portd.0 '//resposta do multiplexer 1
multi_2 var portd.1 '//resposta do multiplexer 2
'////////////////////////////////////////////////////////////////////////////////////////


'////////////// Definição dos pinos de saida de sinais ////////////////////////////////
a   var portd.2 '//
b   var portd.3 '// Bits que realizarão a contagem binaria
c   var portd.4 '// de 1 a 15 dividido em 2 partes
d   var portd.5 '//

habilita_U1 var portd.6 '//habilita o multiplexer U1 quando colocado em zero
habilita_U2 var portd.7 '//habilita o multiolexer U2 quando colocado em zero

led_ok      var porte.0 '//liga quando estiver tudo ok
led_avaria  var porte.1 '//liga quando algum passageiro estiver sem cinto

desabilita_partida  var portc.7 '//liga o rele caso no momento inicial falte cinto
'////////////////////////////////////////////////////////////////////////////////////////
```

Figura 9. – Definição das variáveis

A definição das variáveis se desenvolve conforme o programa vai crescendo, a idéia inicial parte com apenas algumas variáveis, neste caso estamos utilizando 13 variáveis, que conforme visto na figura 8 acima revela uma breve explicação relativa a cada variável.

A variável z: é responsável em liberar o sistema de ignição.

A variável x: é responsável por indicar a poltrona que encontra-se sem cinto de segurança.

A variável multi_1: é responsável por indicar que o multipexador 1 esta atuando.

A variável multi_2: é responsável por indicar que o multipexador 2 esta atuando.

As variáveis a,b,c e d: são responsáveis pela contagem binária.

A variável habilita_U1: é responsável por habilitar o funcionamento do multiplexador 1.

A variável habilita_U2: é responsável por habilitar o funcionamento do multiplexador 2.

A variável led_ok: é responsável por indicar sistema em funcionamento ok.

A variável led_avaria: é responsável por indicar que o sistema tem algum cinto não fechado.

A variável desabilita_partida: é responsável por liberar o sistema quando tudo estiver ok.

3.1.3. Garantindo Funcionamento Adequado

```
'////////////// coloca todas as saidas em nivel logico baixo ////////////////////////////
low desabilita_partida
low led_avaria
low led_ok
low a
low b
low c
low d
'////////////////////////////////////////////////////////////////////////////////////////
'// coloca os pinos que habilitam os multiplex em nivel logico alto para impedir falta/
'// leitura pois são habilitados em nivel logico baixo ////////////////////////////////
high habilita_U1
high habilita_U2
'////////////////////////////////////////////////////////////////////////////////////////
```

Figura 10. – Garantindo bom funcionamento

Para garantir um funcionamento adequado do programa, foi necessário colocar todas as variáveis de saídas em nível lógico baixo e as variáveis de habilitação do multiplexador em nível lógico alto, que melhor será abordado porque dessa adequação inicial.

3.1.4. Iniciado LCD

```
'////////////////////////// iniciando sistema LCD ////////////////////////////////////
lcdout $fe,1,"iniciado........"
pause 500
lcdout $fe,1,"     A"
pause 200
lcdout $fe,14,"n"
pause 200
lcdout $fe,14,"d"
pause 200
lcdout $fe,14,"r"
pause 200
lcdout $fe,14,"e"
pause 200
```

Figura 11. – Iniciando o LCD

Para iniciar o LCD é necessário enviar alguns comandos. Esses comandos enviados devem seguir a seguinte ordem, primeiro envia-se um $FE e depois o comando. Abaixo alguns comandos úteis estão listados na tabela 5:

Tabela 5. – Tabela de comando LCD

COMANDO	OPERAÇÃO
$ FE, 1	Limpa display
$ FE, 2	Retorno (inicio da primeira linha)
$ FE, $ 0C	Cursor desligado
$ FE, $ 0E	Cursor sublinhado ligado
$ FE, $ 0F	Cursor piscante ligado
$ FE, $ 10	Move cursor uma posição à esquerda
$ FE, $ 14	Move cursor uma posição à direita
$ FE, $ C0	Move cursor ao inicio da segunda linha

A figura 11 mostra a instrução lcdout seguida de um comando para o LCD, que por sua vez emite a mensagem para visualização do motorista.

Cada instrução tem um intervalo de tempo que é definido pelo tempo de pause, como mostra a figura 11, este tempo de pause são 500ms e 200ms.

3.1.5. Inicio do Programa

```
'////////////////////////////////////////////////////////////////////////////////////////////////

inicio:
high habilita_U2
low  habilita_U1
low a   '/////////////////
low b   '//
low c   '//     1
low d   '/////////////////
pause 10
if habilita_U1=0 and a=0 and b=0 and c=0 and d=0 and multi_1=0 then x=1

high a  '///////////////
low b   '//
low c   '//     2
low d   '//////////////
pause 10
if habilita_U1=0 and a=1 and b=0 and c=0 and d=0 and multi_1=0 then x=2

low  a  '///////////////
high b  '//
low  c  '//     3
low  d  '//////////////
pause 10
if habilita_U1=0 and a=0 and b=1 and c=0 and d=0 and multi_1=0 then x=3
```

Figura 12. – Inicio do programa

Como pode ser visto na figura 12, a instrução high, tem a finalidade de colocar, neste caso, a variável habilita_U2 em nível lógico alto (significa que este pino tem 5volts), desabilitando o multiplexador 2, tendo em vista que seu pino de enable é barrado como mostra seu data sheet no ANEXO I.

A instrução low, neste caso, coloca a variável habilita_U1 em nível lógico baixo, habilitando o multiplexador 1.

Assim por diante o programa começa a fazer uma contagem binária com as variáveis a, b,c e d, com essa combinação a contagem começa de 0 e termina em 15, varrendo assim todas as entradas do multiplexador 1, ou seja, as poltronas de 1 a 16.

Em seguida inverte-se as instruções, colocando em funcionamento o multiplexador 2 e desabilitando o multiplexador 1, concluindo-se assim as poltronas de 17 a 32.

3.1.6. Escrever Mensagem

```
'////////////////////////////////////////////////////////////////////////////////////////////////

if x=0 then
        lcdout $FE,1,"   Boa Viagem"
        pause 50
        lcdout $fe,$c0,"   TCC-Andre"
        high led_ok
        x=33
        z=1
endif

if x>=1 and x<=32 then
        lcdout $FE,1,"   Atencao"
        pause 50
        lcdout $fe,$c0,"Banco ",#x," s/cinto"
goto sem_cinto
endif

goto inicio
```

Figura 13. – Escrevendo mensagem no LCD

Para escrever a mensagem no LCD, o programa depende apenas da variável x, devido ela se responsável por indicar a poltrona, conforme mostra a figura 13. Quando a variável x é igual a 0, significa que todas as poltronas estão com cinto de segurança e quando a variável x tem valor entre 1 e 32, ela indica a poltrona correspondente a falta do cinto de segurança.

3.1.7. Liberando Partida

```
'////////////////////////////////////////////////////////////////////////////////////////////////

sem_cinto:
if z=0 then low desabilita_partida

low led_ok

high led_avaria
pause 200
low led_avaria
pause 200

if z=1 then
high desabilita_partida
endif
```

Figura 14. – Rotina de liberação da partida

Esta rotina sendo de liberação do sistema entende-se que quando z for igual a 0, a partida estará desabilitada e quando z for igual a 1, a partida será liberada.

A variável z somente é incrementada uma vez no sistema, isso só ocorre quando todos os cintos de segurança estiverem colocados, a partir disso não existira interrupção da partida, sendo apenas monitoradas as poltronas.

3.1.8. Verificado Poltronas

```
if x=1 or x=17 then
    if x=1 then
        low  habilita_U1
        high habilita_U2

    endif

    if x=17 then
        low  habilita_U2
        high habilita_U1
    endif

    low a   '////////////////
    low b   '//
    low c   '//     1
    low d   '////////////////
        if x=1  and multi_1=1 then
            x=0
            goto inicio
        endif
        if x=17 and multi_2=1 then
            x=0
            goto inicio
        endif
endif
```

```
if x=2 or x=18 then
    if x=2 then
        low  habilita_U1
        high habilita_U2
    endif

    if x=18 then
        low  habilita_U2
        high habilita_U1
    endif

    high a   '//////////////
    low b    '//
    low c    '//     2
    low d    '//////////////
        if x=2  and multi_1=1 then
            x=0
            goto inicio
        endif
        if x=18 and multi_2=1 then
            x=0
            goto inicio
        endif
endif
```

Figura 15. – Verificando qual poltrona

A rotina de verificação das poltronas também necessita de informação da variável x, neste caso ela é responsável por determinar qual poltrona esta sem cinto, pois a comparação é igual em ambos os multiplexadores, apenas muda se a forma de leitura do dado multiplexado devido a configuração ser pré definida via programa.

3.2. SIMULAÇÃO E TESTES VIA SOFTWARE

Após a elaboração do programa, foi realizada a montagem do projeto via software Proteus 7 Professional, para a fase de testes e simulações.

Abaixo a figura 16 mostra o esquema do projeto montado no software e a figura 17 mostra a simulação sendo executada pelo software.

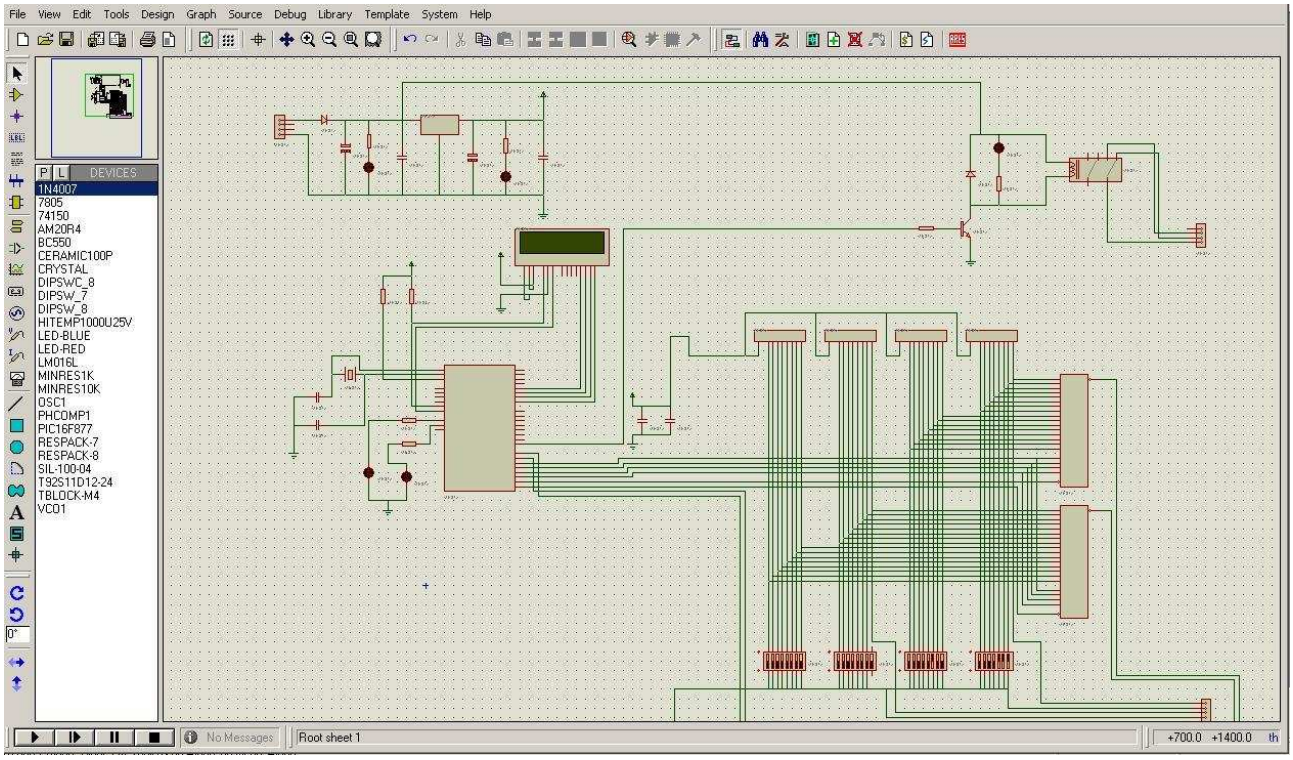


Figura 16. – Esquema completo do projeto.

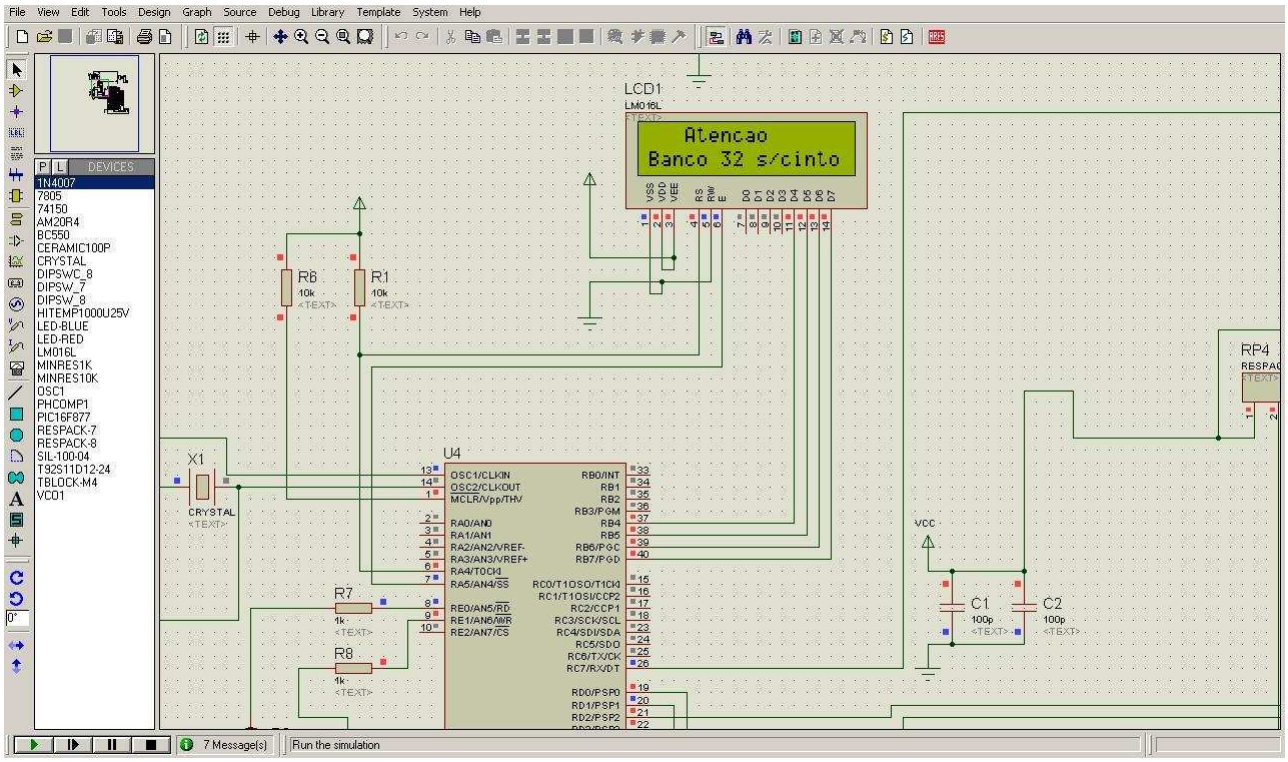


Figura 17. – Projeto sendo simulado.

3.3. GRAVANDO PROGRAMA NO PIC

Para gravar o programa no PIC, utiliza-se a porta paralela do computador, uma placa gravadora (conforme figura 18) e um software de gravação chamado de IC-Prog.

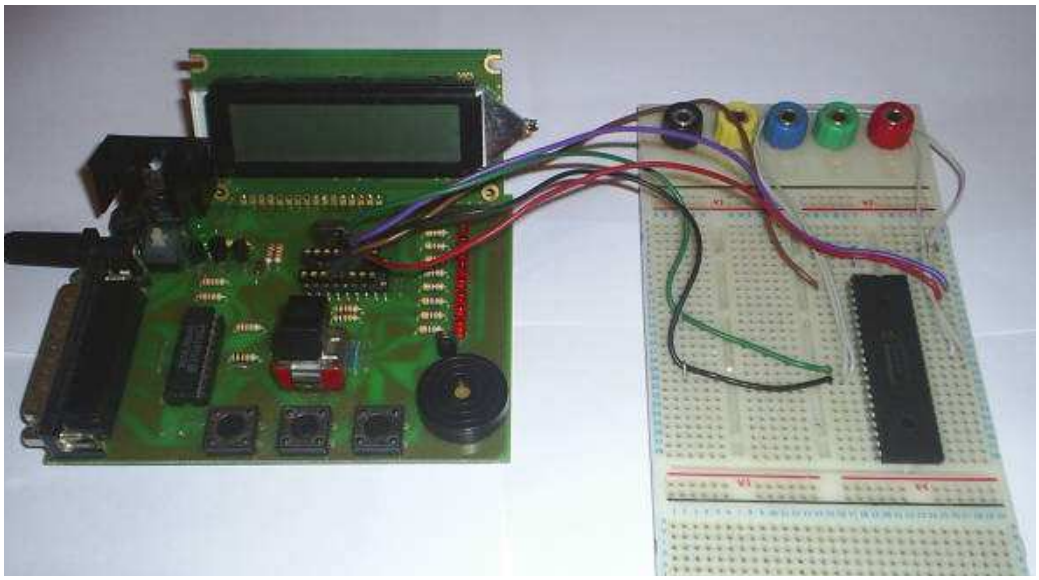


Fig. 18 – Placa gravadora do Pic

A utilização do software IC-Prog é muito simples, pois sua interface é bem amigável, como podemos ver na figura 19 abaixo. O procedimento básico é carregar o programa já compilado com extensão ".hex," escolher o modelo do pic que será gravado e efetuar a gravação através do comando Programar Tudo ou pelo botão de atalho .

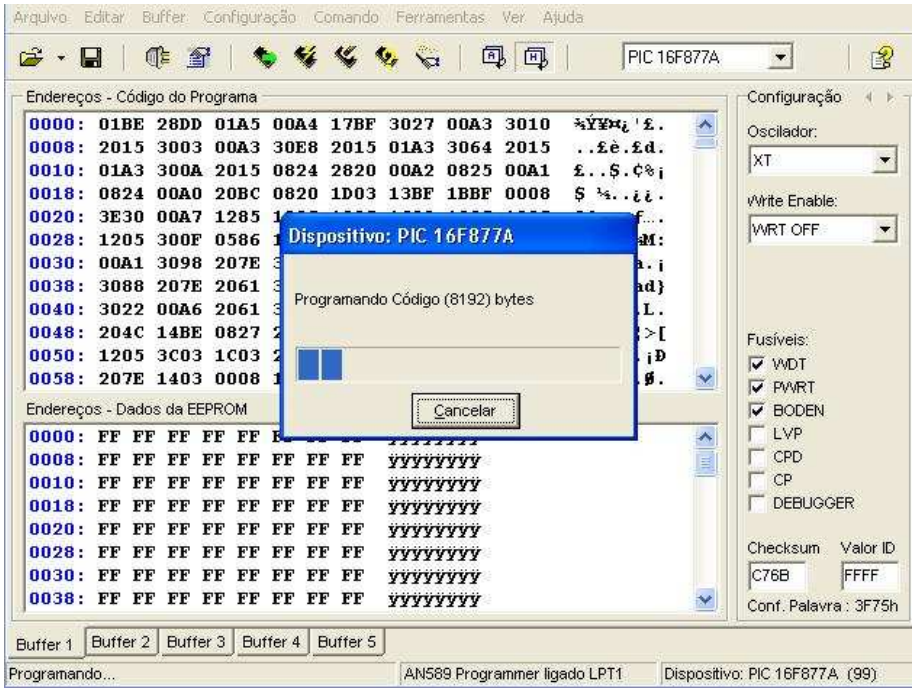


Fig. 19 – Interface do Programa IC-Prog

3.4. CONFECÇÃO DA PCI

Para elaborar e testar o layout da placa de circuito impresso, foi utilizado também o software Proteus, pois este software disponibiliza ferramenta que consegue dispor os componentes da maneira mais adequada para uma placa ser confeccionada sem erros.

O software dispõe de visões da placa pronta em 3D, facilitando assim sua visualização para conclusão do projeto.

Depois de finalizado o projeto da placa no software, sua confecção será por meio de transferência térmica.

A seqüência da transferência térmica é a seguinte.

- Imprimir com impressora a laser o layout da placa em uma folha de fotolito.
- Limpar a placa virgem de cobre, com álcool isopropilico e lã de aço.
- Lavar com água e guardar a placa para não sujar novamente.
- Com o layout em mãos, centraliza-se com a placa já limpa em uma superfície plana.
- Aquecer o ferro de passar roupa.
- Primeiro passar o ferro nos quatro cantos para fixar o papel e assim por diante com movimentos circulares e fazendo pressão, continuar a passar por toda a placa até aquecê-la o suficiente para transferir o toner para a placa.
- Assim que terminar essa etapa, colocar a placa no congelador e aguarda alguns minutos (depende do tamanho da placa).
- Retirar a placa do congelador e com cuidado desprender o papel de fotolito, verificando a transferência.
- Assim com o layout já transferido, o procedimento de corrosão será através do mergulho da placa na solução de percloreto de ferro.

- Depois de terminado a corrosão limpar a placa para executar a furação referente aos componentes.

- Esta furação será executada com uma furadeira elétrica pequena com mandril para broca de 0.8mm.

3.5. MAQUETE

Para uma visualização próxima do real, foi elaborada uma maquete para simular uma poltrona do ônibus, conforme mostra a figura 20.

Nesta maquete o sensor que monitora a poltrona, é um sensor óptico que foi instalado na lateral do apoio de braço, e o sistema é liberado pelo cinto quando colocado, devido a ter outro sensor em sua trava, esse por sua vez é um sensor reed que funciona através do campo magnético de um imã.

Figura 20. – Maquete da Poltrona.

4. CONSIDERAÇÕES FINAIS

Nota-se que com a tecnologia desenvolvendo cada vez mais rápido e melhor, em breve será investido mais na segurança dos passageiros de ônibus de viagem, pois hoje já existe a lei, mas não é sempre que os passageiros colaboram em cumpri-la, mesmo sendo para sua própria integridade física.

A utilização de microcontroladores para essa finalidade, como recurso de segurança, aplicado de maneira adequada e consciente, pode se tornar cada vez mais comum, devido ao crescimento de sua popularidade ser cada vez maior.

Este projeto pode ser considerado como o “ponta pé” inicial nisso, pois existindo um responsável monitorando o uso do cinto de segurança, seu uso será mais respeitado. E a partir desta idéia muitas outras podem surgir aprimorando a existente ou inovando.

REFERÊNCIAS BIBLIOGRÁFICAS

ADURA, Flávio Emir; Montal, José Heverardo da Costa; Sabbag, Alberto Francisco ***Uso do cinto de segurança durante a gravidez.*** Revista da Associação Médica Brasileira, 2004, vol.50, n. 1, ISSN 0104-4230.

BECKER, Marcelo. Diário Catarinense. ***Cinto poupa vidas no trânsito há uma década.*** Disponível em: <http://www.perkons.com.br/imprensa.php?id=1644&pg=0> Acesso em 19 set 2008.

GUITIERRES, Prof. Adilson. ***Microcontrolador PIC 16F84 – Arquitetura Harvard.*** Disponível em:< http://www.edutecbauru.com.br/cursopic/aula09.htm> Acesso em 2 ago 2008.

Mecanique, ***Microcontrollers.*** Disponível em: < http://www.mecanique.co.uk/products/parts/index.html#PIC16F877A> Acesso em 02 ago 2008.

Microchip. ***PIC16F877A.*** Disponível em: <http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010242> Acesso em 19 set 2008.

SOARES, Marcio José. ***Monte um Teclado Matricial***. Revista Eletrônica Total. Ano 17, n. 115 p.20-23, maio – junho/06

SOLBET, ***Tutoriais sobre programação de microcontroladores.*** Versão 1.0 – Campinas, 2007. Disponível em: http://www.solbet.com.br Acesso em 20 set 2008.

SOUZA, Vitor Amadeu. ***Programação em BASIC para o Microcontrolador PIC18F1220: conceitos e aplicações.*** 1. Ed. – São Paulo: Editora Érica, 2006.

ZANCO, Wagner da Silva. ***Microcontroladores PIC: técnicas de software e hardware para projetos de circutos eletrônicos com base no PIC 16F877A .*** 1. ed. – São Paulo: Editora Érica, 2006.

WEIGEL, Richard. ***Uso do IC – Prog para gravação de PIC.*** Revista Eletrônica Total. Ano 18, n.119, p.50-56, nov – dez/06.

WEIGEL, Richard. ***Gravadores de PIC.*** Revista Eletrônica Total. Ano 17, n.116, p.06-13, junho/06.

GLOSSÁRIO

Microcontrolador	É um microprocessador e vários periféricos num único componente eletrônico.
Software	Informação de que o microcontrolador necessita, para poder funcionar. O software pode ser escrito em diversas linguagens tais como: Basic, C, Pascal ou assembler.
Simulador	Software para “rodar” num PC que simula o funcionamento interno do microcontrolador. É um instrumento ideal para verificar as rotinas de software e todas as porções de código que não implicam ligação com o mundo exterior.
Assembler	Pacote de software que traduz código fonte em código que o microcontrolador pode compreender. Uma parte deste software destina-se também à detecção dos erros cometidos ao escrever o programa.
Hardware	Microcontrolador, memória, alimentação, circuitos de sinal e todos os componentes ligados ao microcontrolador.
Byte, Kilobyte, Megabyte	Termos relacionados com quantidades de informação. A unidade básica é o byte que corresponde a 8 bits. Um kilobyte são 1024 bytes e um megabyte tem 1024 kilobytes.

.

ANEXO I – DATA SHEET C.I. 74LS150

SDLS054

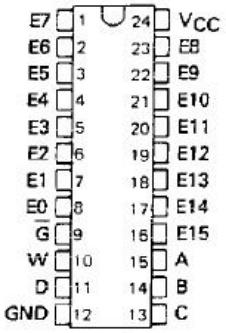
SN54150, SN54151A, SN54LS151, SN54S151,
SN74150, SN74151A, SN74LS151, SN74S151
DATA SELECTORS/MULTIPLEXERS

DECEMBER 1972 – REVISED MARCH 1988

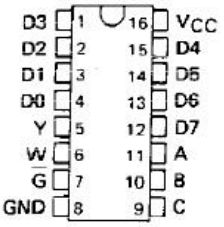
- '150 Selects One-of-Sixteen Data Sources
- Others Select One-of-Eight Data Sources
- All Perform Parallel-to-Serial Conversion
- All Permit Multiplexing from N Lines to One Line
- Also For Use as Boolean Function Generator
- Input-Clamping Diodes Simplify System Design
- Fully Compatible with Most TTL Circuits

TYPE	TYPICAL AVERAGE PROPAGATION DELAY TIME DATA INPUT TO W OUTPUT	TYPICAL POWER DISSIPATION
'150	13 ns	200 mW
'151A	8 ns	145 mW
'LS151	13 ns	30 mW
'S151	4.5 ns	225 mW

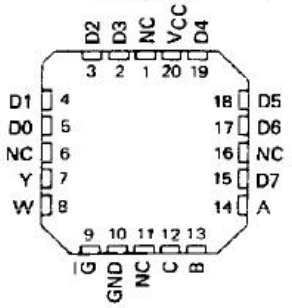
SN54150 . . . J OR W PACKAGE
SN74150 . . . N PACKAGE
(TOP VIEW)



SN54151A, SN54LS151, SN54S151 . . . J OR W PACKAGE
SN74151A . . . N PACKAGE
SN74LS151, SN74S151 . . . D OR N PACKAGE
(TOP VIEW)



SN54LS151, SN54S151 . . . FK PACKAGE
(TOP VIEW)



NC - No internal connection

description

These monolithic data selectors/multiplexers contain full on-chip binary decoding to select the desired data source. The '150 selects one-of-sixteen data sources; the '151A, 'LS151, and 'S151 select one-of-eight data sources. The '150, '151A, 'LS151, and 'S151 have a strobe input which must be at a low logic level to enable these devices. A high level at the strobe forces the W output high, and the Y output (as applicable) low.

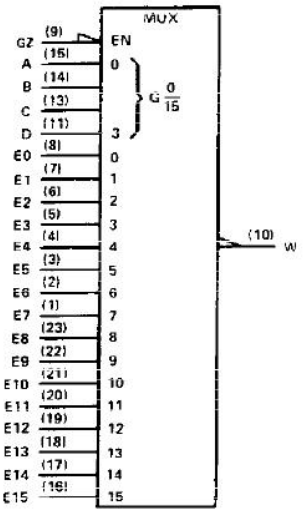
The '150 has only an inverted W output; the '151A, 'LS151, and 'S151 feature complementary W and Y outputs.

The '151A and '152A incorporate address buffers that have symmetrical propagation delay times through the complementary paths. This reduces the possibility of transients occurring at the output(s) due to changes made at the select inputs, even when the '151A outputs are enabled (i.e., strobe low).

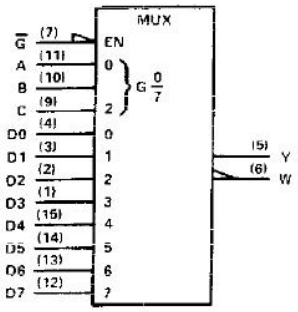
TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

logic symbols†

'150



'151A, 'LS151, 'S151



†These symbols are in accordance with ANSI/IEEE Std. 91-1984 and IEC Publication 617-12.
Pin numbers shown are D, J, N, and W packages.

'150

FUNCTION TABLE

INPUTS					OUTPUT
SELECT				STROBE	
D	C	B	A	$\overline{G}$	W
X	X	X	X	H	H
L	L	L	L	L	$\overline{E0}$
L	L	L	H	L	$\overline{E1}$
L	L	H	L	L	$\overline{E2}$
L	L	H	H	L	$\overline{E3}$
L	H	L	L	L	$\overline{E4}$
L	H	L	H	L	$\overline{E5}$
L	H	H	L	L	$\overline{E6}$
L	H	H	H	L	$\overline{E7}$
H	L	L	L	L	$\overline{E8}$
H	L	L	H	L	$\overline{E9}$
H	L	H	L	L	$\overline{E10}$
H	L	H	H	L	$\overline{E11}$
H	H	L	L	L	$\overline{E12}$
H	H	L	H	L	$\overline{E13}$
H	H	H	L	L	$\overline{E14}$
H	H	H	H	L	$\overline{E15}$

'151A, 'LS151, 'S151

FUNCTION TABLE

INPUTS				OUTPUTS	
SELECT			STROBE		
C	B	A	$\overline{G}$	Y	W
X	X	X	H	L	H
L	L	L	L	D0	$\overline{D0}$
L	L	H	L	D1	$\overline{D1}$
L	H	L	L	D2	$\overline{D2}$
L	H	H	L	D3	$\overline{D3}$
H	L	L	L	D4	$\overline{D4}$
H	L	H	L	D5	$\overline{D5}$
H	H	L	L	D6	$\overline{D6}$
H	H	H	L	D7	$\overline{D7}$

H = high level, L = low level, X = irrelevant
$\overline{E0}$, $\overline{E1}$ . . . $\overline{E15}$ = the complement of the level of the respective E input
D0, D1 . . . D7 = the level of the D respective input

recommended operating conditions

	SN54'			SN74'			UNIT
	MIN	NOM	MAX	MIN	NOM	MAX	
Supply voltage, V_{CC}	4.5	5	5.5	4.75	5	5.25	V
High-level output current, I_{OH}			−800			−800	μA
Low-level output current, I_{OL}			16			16	mA
Operating free-air temperature, T_A	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS†		'150			'151A			UNIT
				MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IH}	High-level input voltage			2			2			V
V_{IL}	Low-level input voltage					0.8			0.8	V
V_{IK}	Input clamp voltage	V_{CC} = MIN, I_I = −8 mA				−1.5			−1.5	V
V_{OH}	High-level output voltage	V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = 0.8 V, I_{OH} = −800 μA		2.4	3.4		2.4	3.4		V
V_{OL}	Low-level output voltage	V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = 0.8 V, I_{OL} = 16 mA			0.2	0.4		0.2	0.4	V
I_I	Input current at maximum input voltage	V_{CC} = MAX, V_I = 5.5 V				1			1	mA
I_{IH}	High-level input current	V_{CC} = MAX, V_I = 2.4 V				40			40	μA
I_{IL}	Low-level input current	V_{CC} = MAX, V_I = 0.4 V				−1.6			−1.6	mA
I_{OS}	Short-circuit output current§	V_{CC} = MAX	SN54'	−20		−55	−20		−55	mA
			SN74'	−18		−55	−18		−55	
I_{CC}	Supply current	V_{CC} = MAX, See Note 3			40	68		29	48	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.
‡ All typical values at V_{CC} = 5 V, T_A = 25°C.
§ Not more than one output of the '151A should be shorted at a time.
NOTE 3: I_{CC} is measured with the strobe and data select inputs at 4.5 V, all other inputs and outputs open.

switching characteristics, V_{CC} = 5 V, T_A = 25°C

PARAMETER¶	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	'150			'151A			UNIT
				MIN	TYP	MAX	MIN	TYP	MAX	
t_{PLH}	A, B, or C (4 levels)	Y	C_L = 15 pF, R_L = 400 Ω, See Note 4					25	38	ns
t_{PHL}								25	38	
t_{PLH}	A, B, C, or D (3 levels)	W			23	35		17	26	ns
t_{PHL}					22	33		19	30	
t_{PLH}	Strobe $\overline{G}$	Y						21	33	ns
t_{PHL}								22	33	
t_{PLH}	Strobe $\overline{G}$	W			15.5	24		14	21	ns
t_{PHL}					21	30		15	23	
t_{PLH}	D0 thru D7	Y						13	20	ns
t_{PHL}								18	27	
t_{PLH}	E0 thru E15, or D0 thru D7	W			8.5	14		8	14	ns
t_{PHL}					13	20		8	14	

¶ t_{PLH} = propagation delay time, low-to-high-level output
t_{PHL} = propagation delay time, high-to-low-level output
NOTE 4: Load circuits and voltage waveforms are shown in Section 1.