UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FRANCO RIPOLL LEITE

PROJETO DE DIPLOMAÇÃO

SISTEMA DE AUTOMAÇÃO RESIDENCIAL

Porto Alegre
2006

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

SISTEMA DE AUTOMAÇÃO RESIDENCIAL

Projeto de Diplomação apresentado ao Departamento de Engenharia Elétrica da Universidade Federal do Rio Grande do Sul, como parte dos requisitos para Graduação em Engenharia Elétrica.

ORIENTADOR: Prof. Dr. Marcelo Lubaszewski

Porto Alegre
2006

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FRANCO RIPOLL LEITE

SISTEMA DE AUTOMAÇÃO RESIDENCIAL

Este projeto foi julgado adequado para fazer jus aos créditos da Disciplina de "Projeto de Diplomação", do Departamento de Engenharia Elétrica e aprovado em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador: _____

Prof. Dr. Marcelo Lubaszewski, UFRGS

Banca Examinadora: _____

Porto Alegre, julho de 2006

DEDICATÓRIA

Dedico este trabalho a minha família que contribuiu para a minha formação e meus amigos por me apoiarem no desenvolvimento desse projeto.

AGRADECIMENTOS

Gostaria de agradecer a todos que de alguma forma me apoiaram no desenvolvimento deste trabalho. Em especial a minha família, aos meus colegas e amigos pela compreensão nos momentos que estive envolvido com a realização desse trabalho e ao professor Marcelo Lubaszewski, pela confiança depositada em mim, ao aceitar ser meu orientador, e pelos ensinamentos transmitidos.

RESUMO

Devido ao grande número de aparelhos e dispositivos eletro-eletrônicos presentes nas residências atualmente, um sistema de automação residencial com possibilidade de programar datas e horários para ativar e desativar esses aparelhos torna-se um meio prático de comandar esses dispositivos. O sistema também pode ser operado a distância, através de uma ligação telefônica e uma senha de acesso, permitindo que aparelhos eletrodomésticos possam ser controlados mesmo quando o usuário não se encontra em casa. Fisicamente o sistema de automação residencial desenvolvido consiste num módulo central microcontrolado, com teclado e display, responsável por comandar diversos módulos espalhados na residência, sendo que cada um desses módulos aciona uma carga (eletrodoméstico) à qual está ligado.

Palavras-chaves: Automação Residencial; Sistema Microcontrolado; Controle a Distância; DTMF; Acionamento de Cargas; Eletrônica; Acionamento Programado.

ABSTRACT

Had the great number of devices and electro-electronic devices gifts in the residences currently, a system of residential automation with possibility to program dates and schedules to activate and to disactivate these devices becomes a way practical to command these devices. The system also can be operated in the distance, through a telephonic linking and an access password, allowing that household-electric devices can exactly be controlled when the user does not meet in house. Physically the developed system of residential automation consists of a microcontrolled central module, with keyboard and display, responsible for commanding diverse modules spread in the residence, being that each one of these modules sets in motion a load (household-electric) which is connected.

Keywords: Residential automation; Microcontrolled system; Control in the distance; DTMF; Load drive; Electronics; Programmed drive.

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

A/D – Analogic/Digital

CS – Chip Select

DTMF – Dual Tone Multi-Frequency

ISP - In-System Programmable

LED – Light Emissor Diode

LCD – Liquid Crystal Display

LSB – Least Significant Bit

RAM – Random Access Memory

RD – Read

RMS - Root Means Square

Rth - Resistência Térmica

RX – Receptor

SRAM – Static Random Access Memory

Tamb – Temperatura Ambiente

Tj – Temperatura da Junção

TX – Transmissor

1 INTRODUÇÃO

Hoje em dia é crescente o número de aparelhos e dispositivos eletro-eletrônicos presentes nas residências. Dessa forma um sistema de automação residencial com possibilidade de programar datas e horários para ativar e desativar esses aparelhos torna-se um meio prático de comandar esses dispositivos. A integração de sistemas de automação residencial com sensores de temperatura ambiente também é bastante útil, permitindo que cargas como ventiladores e ar-condicionados sejam acionados automaticamente.

O equipamento proposto busca atenter a essa demanda, e foi pensado desenvolver um sistema prático e funcional, tendo uma interface que o deixasse fácil de ser operado pelo usuário, no caso através de um teclado telefônico comum e um display.

Entre as funcionalidades implemantadas também está o comando a distância, através de uma ligação telefônica, uma outra opção presente no sistema, permitindo que aparelhos eletrodomésticos possam ser controlados mesmo quando o usuário não se encontra em casa, através de uma senha de acesso ao sistema.

2 SISTEMA DE AUTOMAÇÃO RESIDENCIAL

O equipamento desenvolvido consiste num módulo central microcontrolado, que comanda diversos módulos espalhados na residência, estes contendo uma etapa de potência responsável por acionar a carga à qual estão ligados.

A programação do acionamento das cargas pode ser feita e modificada a qualquer momento, através do teclado numérico matricial presente no módulo central, e essas cargas também podem ser nomeadas, para melhor identificação, de modo muito semelhante ao modo como são escritas mensagens de texto em celulares.

A temperatura ambiente também é um dos parâmetros dessa programação, permitindo que cargas como aquecedores e ventiladores possam ser acionados automaticamente conforme a temperatura.

Para maior segurança o sistema verifica os dados inseridos, impedindo a entrada de dados incoerentes, que poderiam causar mal funcionamento do aparelho e erros na programação dos acionamentos. O acesso remoto ao sistema só é permitido através de senha, e um rigoroso controle de ociosidade no acesso por telefone desconecta a ligação telefônica caso se fique muito tempo sem enviar comandos ou no caso da ligação ser perdida.

O equipamento, tanto o módulo central, como os módulos de acionamento de cargas, foram feitos com peças largamente utilizadas na indústria eletrônica, e fáceis de serem encontradas no comércio, não havendo dessa forma componentes de custo excessivamente alto.

2.1 MÓDULO CENTRAL

Do ponto de vista do módulo central cada carga a ser programada, como por exemplo um eletrodoméstico ou uma lâmpada, é vista como um canal a ser acionado ou desativado. No total são 20 canais disponíveis para o usuário. Para uma rápida visualização, além do display, o status de cada canal é sempre mostrado através de 20 LEDs, cada LED correspondendo a um canal. Abaixo se tem uma foto do módulo onde se pode observar o LCD, o teclado numérico matricial e os LEDs correspondentes aos canais.

Figura 1 - Módulo Principal

2.1.1 FLUXOGRAMA E FUNCIONAMENTO

A seguir é mostrado o fluxograma de funcionamento do sistema, e nas próximas seções será dada uma explicação detalhada de cada parte desse fluxograma.

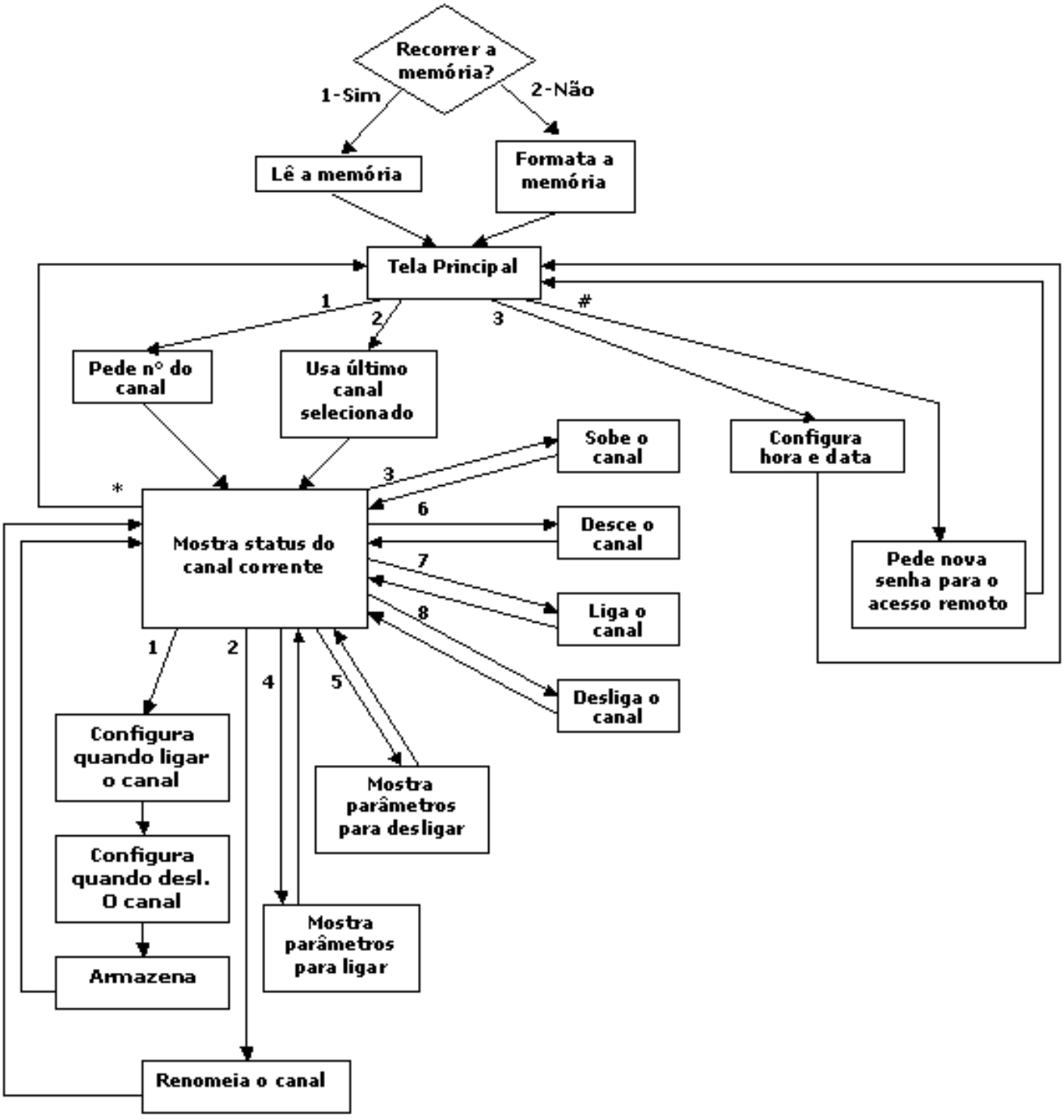


Figura 2 – Fluxograma do Módulo Principal

2.1.1.1 FORMATAÇÃO DA MEMÓRIA

Quando o sistema é iniciado é feita uma pergunta se o usuário deseja ou não recorrer a memória. Na memória são guardadas informações como parâmetros para ligar e desligar os canais e o nome desses canais. Em caso afirmativo (tecla 1) o sistema considera que os dados contidos na memória são validos, por isso, escolhendo-se essa opção deve-se ter certeza que a memória colocada no suporte contém valores corretos.

Caso não se deseje recorrer aos dados da memória (tecla 2) a mesma é preenchida com valores de forma que os nomes dos canais apareçam em branco e que nenhuma data esteja programada para acionamento dos canais.

2.1.1.2 TELA PRINCIPAL

A tela principal mostra a hora , a data (dd/mm/aa), o dia da semana e a temperatura ambiente, e aparece quando o módulo está em modo de espera. Uma rotina de interrupção pelo Timer 0 é responsável pela atualização em tempo real do relógio e data, assim como a aquisição dos dados do conversor A/D para indicação de temperatura ambiente. A tela principal é atualizada a cada segundo e também acontece a verificação da necessidade ou não de se ligar ou desligar algum canal. Logo a precisão para o acionamento de um canal é de 1 segundo. Em termos de temperatura a precisão é de 1°C.

2.1.1.3 SELEÇAO DE UM CANAL

Para ver as informações de um determinado canal existem duas opções. A primeira delas é apertando a tecla 1 e em seguida digitando o número do canal desejado. A segunda

opção é digitando a tecla 2, serão mostradas informações do último canal consultado. Para se chegar no canal desejado basta navegar pelas teclas 3 e 6, subindo e descendo os canais, até chegar no canal desejado.

2.1.1.4 TELA DE STAUS DO CANAL

Depois que um canal é selecionado, é exibida uma tela que informa o número do canal, seu nome, e na linha de baixo o status do mesmo (ligado ou desligado). Nessa tela temos as seguintes opções :

Tecla 1 – Configura parâmetros para se ligar e desligar um canal

Tecla 2 – Permite nomear um canal com até 10 caracteres

Tecla 3 – Mostra o canal acima

Tecla 6 – Mostra o canal abaixo

Tecla 4 – Mostra parâmetros que ligam o canal corrente

Tecla 5 – Mostra parâmetros que desligam o canal corrente

Tecla 7 – Liga o canal corrente

Tecla 8 – Desliga o canal corrente

Tecla * – Volta à tela principal

A ação de ligar ou desligar um canal através do teclado (teclas 7 e 8, respectivamente) força o apagamento da programação do canal, caso haja programação.

2.1.1.5 PROGRAMANDO UM CANAL

Ao programar o acionamento de um canal pela primeira vez os campos correspondentes ao horário, data e temperatura encontram-se preenchidos com tracejados “--”. Isso indica que esse parâmetro não será levado em conta quando houver a verificação da necessidade de se acionar o canal. Com isso têm-se inúmeras possibilidades na programação,

não sendo necessário digitar uma data exata. Por exemplo, caso se defina um horário, deixando os demais espaços com os caracteres tracejados "--", não importará a data, e o acionamento acontecerá todos os dias, na hora estipulada. Para se inserir os tracejados usa-se a tecla REDIAL do teclado. Para voltar para um campo anterior usa-se a tecla *. Para avançar um campo sem alterar o campo corrente usa-se a tecla #.

Para maior praticidade pode-se ainda escolher se o acionamento acontecerá todos os dias da semana, ou apenas em determinados dias da semana. Para isso é exibido na tela de configuração do canal uma seqüência, "D23456S", que corresponde aos dias, de domingo à sábado. Seleciona-se um ou mais dias da semana inserindo-se um quadrado em cima do dia da semana correspondente. Por exemplo, "D2 45 S" indica que o acionamento somente ocorrerá nas terças-feiras e sextas-feiras. Para inserir esse caracter quadrado usa-se a tecla REDIAL. Para desmarcar o dia da semana, retirando o caracter quadrado usa-se a tecla FLASH. Para voltar para um campo anterior usa-se a tecla *. Para avançar um campo sem alterar o campo corrente usa-se a tecla #.

Pode-se também escolher uma certa temperatura para o acionamento do canal, dessa forma tornando o equipamento útil para controlar a temperatura ambiente, acionando cargas como aquecedores ou ventiladores.

2.1.1.6 NOMEANDO UM CANAL

Para uma associação mais fácil e intuitiva ao usuário de qual carga (eletrodoméstico) corresponde determinado número de canal, existe a possibilidade de nomear todos os 20 canais, com até 10 caracteres cada um, melhorando assim a distinção entre eles.

Para inserir caracteres alfanuméricos no teclado telefônico foi adotado um sistema muito semelhante ao usado para se escrever mensagens de texto nos celulares. Pressiona-se várias vezes a mesma tecla até aparecer a letra desejada. Para avançar basta teclar uma tecla

diferente, ou caso se deseje inserir letras que estejam na mesma tecla, como por exemplo “A” e “B” usa-se a tecla #, que pula para o próximo campo, permitindo que dessa forma sejam escritas em seqüência letras que pertencem ao mesmo dígito do teclado telefônico, no caso o dígito “2”. Para apagar uma letra digitada usa-se a tecla *. Para inserir caracteres especiais como espaços, traços ou pontos usa-se a tecla 0. Para finalizar a operação de nomeação do canal basta apertar a tecla REDIAL. O nome será salvo na memória e retornará a tela de status do canal. Caso não se deseje salvar o que foi escrito basta apertar tantas vezes quanto for necessário a tecla *. O que foi escrito vai sendo apagado até o início, retornando a tela de status do canal, sem salvar.

2.1.1.7 ROLAGEM DE CANAIS

Através da teclas 3 e 6 é possível rolar os canais, para cima ou para baixo, respectivamente, sendo um modo prático de visualizar o status dos canais.

2.1.1.8 CANAIS ESPECIAIS

Além dos 20 canais externos disponíveis para acionamento de cargas existem canais internos ao equipamento, que controlam a luz de fundo do display e o relé que conecta ou não um telefone externo a linha telefônica usada pelo equipamento de automação residencial.

Estando na tela principal, para acessar as configurações da luz de fundo basta apertar a tecla 4. Pode configurar o acendimento ou desligamento da luz de fundo do display como se fosse um dos 20 canais externos. A diferença consiste que esse canal não pode ser renomeado e não existe a rolagem entre os canais.

Da mesma forma que a luz de fundo, apertando a tecla 5 tem-se acesso ao relé que conecta um telefone à linha telefônica. Isso é útil pois podemos programar os horários que o telefone estará na linha, desconectando nos horários que não queremos ser perturbados, como

por exemplo de madrugada. Essa ação não influencia o acesso remoto através de ligação telefônica, que continua funcionando, apenas o telefone não ira tocar caso esteja desconectado.

2.1.1.9 AJUSTE DE HORA E DATA

Estando na tela principal, digitando a tecla 3 podemos acertar a hora e data do sistema. O processo é bastante semelhante a programar um canal, com a diferença que não configuramos a temperatura, e o dia da semana aparece escrito de forma abreviada, com 3 letras. No momento de ajustar o dia da semana podemos trocar o dia apertando a tecla 3, para subir o dia da semana, e a tecla 6 para descer o dia da semana. Quando o dia da semana estiver sendo trocado ele ficará piscando na tela. Depois de ajustado, para avançar para o próximo campo, o do dia, basta apertar a tecla #. Para retroceder digita-se a tecla *.

Assim como na programação de canais, é verificado o que foi digitado pelo usuário, não sendo permitida dessa forma a entrada de dados incoerentes, que causariam erro no sistema, como por exemplo mês maior que “12” ou minutos maiores que “59”. O sistema também leva em conta anos bissextos, mantendo a data correta ao longo dos anos.

2.1.1.10 SENHA DE ACESSO REMOTO

Estando na tela principal, para definir a senha que será usada no acesso remoto deve-se apertar a tecla #. Será pedida uma nova senha. Basta digitar a senha escolhida, que deve ter no máximo 6 dígitos. Ao final aperta-se novamente # para salvar a nova senha e retornar a tela principal. Caso a senha digitada alcance o tamanho de 6 dígitos o sistema salva e retorna sem necessidade de se apertar o sustenido “#” ao final.

2.1.2 ACESSO REMOTO

Um circuito integrado decodificador DTMF, o MT8870, fica constantemente monitorando a linha telefônica, e sempre que um tom DTMF válido é detectado ele gera uma interrupção externa no microcontrolador, deixando o código equivalente de 4 bits pronto para ser lido, porém ainda com os pinos de saída em alta impedância, visto que está conectado junto com os demais periféricos no barramento de dados. O microcontrolador então carrega o endereço de memória correspondente ao decodificador DTMF e lê os dados. Os 4 bits mais significativos não são levados em conta, pois apenas os 4 bits menos significativos estão conectados ao decodificador, e com apenas esses 4 bits pode-se representar todas as possíveis teclas do telefone.

Para se ter acesso ao sistema através de uma ligação telefônica é necessário ligar para o número telefônico correspondente a linha que está conectada no equipamento. Se a linha estiver livre, começará a chamar, como acontece em uma ligação normal para telefone. Deve-se então esperar o primeiro toque de chamando, e assim que houver pausa, antes do próximo toque, deve-se digitar #. Ao receber um # o equipamento entende que é com ele que estamos querendo conversar, atendendo à ligação telefônica. A partir desse momento estamos conectados ao sistema de automação residencial. Deve-se esperar a pausa para digitar o # pois caso contrário o tom DTMF correspondente ao # será sobreposto ao sinal de chamando proveniente da central telefônica. O equipamento consegue atender a ligação telefônica pois muda a impedância da linha, colocando um resistor em paralelo com a mesma, através de um relé.

Após digitar #, digita-se a senha, e então aperta-se # novamente. Dessa forma sempre a primeira ação que o microcontrolador tem é verificar se foi digitado #. Após ele verifica se já havia sido digitado # antes. Caso tenha sido o primeiro #, ele carrega um endereço inicial

de memória para armazenar a seqüência de teclas que o usuário vai digitando. Esse endereço inicial vai sendo incrementado, de forma a deixar a seqüência armazenada na memória. Para quem está no interior da residência essa ação não influencia na funcionalidade do equipamento. O sistema não fica travado somente porque atendeu à chamada telefônica, o processo de armazenamento da seqüência digitada pelo telefone acontece como em um segundo plano, mantendo as demais funcionalidades do aparelho funcionando.

Somente após a seqüência de dados recebida por telefone estar correta, (#, senha correta, #) é que o sistema fica travado ao uso direto, de dentro da residência, e quem está no telefone recebe o comando do sistema. Isso é importante para que não hajam ações simultâneas conflitantes, uma vindo remotamente, da ligação telefônica, e outra direto do teclado do aparelho. Por outro lado, tentativas incorretas de acesso ao sistema por telefone não travarão o uso do mesmo para o usuário que está no interior da residência.

2.1.2.1 AÇÕES POSSIVEIS NO ACESSO REMOTO

Após ter acesso ao equipamento, um sinal sonoro é enviado, avisando que o sistema está esperando que seja digitado um canal. Após informar o canal temos as seguintes opções:

Tecla 1 – permite ir para outro canal, digitando o número do mesmo após a tecla 1

Tecla 3 – vai para o canal acima, e em seguida informa o status através de sinal sonoro

Tecla 6 – vai para o canal abaixo, e em seguida informa o status através de sinal sonoro

Tecla 7 – liga o canal corrente

Tecla 8 – desliga o canal corrente

Tecla * - encerra chamada telefônica

Demais teclas – é emitido um sinal sonoro indicando o status do canal

Como pode ser visto as opções no acesso remoto são bastante semelhantes às do acesso normal, pelo teclado do equipamento. A diferença consiste em que as informações são passadas ao usuário através de sinais sonoros, ao invés do display, e não há possibilidade de programar os canais. As teclas 3 e 6 permitem a rolagem dos canais, a partir de um canal primeiramente digitado. Como não temos a informação visual de qual é o canal corrente, podemos eventualmente nos perder ao rolar os canais varias vezes, por isso essa não é uma ação aconselhável.

Um meio mais seguro de se chegar em um determinado canal é digitando a tecla 1. Em seguida ouviremos um sinal sonoro duplo, indicando que podemos digitar o canal desejado. Após digitar o canal um sinal sonoro indica seu status atual, e ele passa a ser o canal corrente. Caso seja digitado um canal inválido, por exemplo maior que 20, é emitido novamente o sinal sonoro duplo, indicando que devemos digitar novamente o canal desejado.

Após a seleção do canal podemos ligar ou desligar o mesmo, através das teclas 7 e 8 respectivamente. A ação de ligar ou desligar um canal força o apagamento de sua programação, da mesma forma que acontece quando executamos essas ações direto do teclado do equipamento.

Para maior segurança existe um rigoroso controle de ociosidade na linha telefônica, de forma que a chamada é encerrada caso não haja ação por parte do usuário por um certo tempo. Caso não houvesse esse controle e a ligação telefônica fosse perdida, sem o correto encerramento do usuário, o relé manteria o resistor em paralelo com a linha, o que indicaria telefone ocupado em uma próxima tentativa de discagem para o equipamento.

Esses tempos foram pré-determinados como: 6 segundos para digitar a senha, 180 segundos para selecionar um canal e 240 segundos para tomar uma ação.

2.1.2.2 SINALIZAÇÕES SONORAS NO ACESSO REMOTO

O fato de não termos acesso visual ao display do equipamento durante uma chamada telefônica implica que as informações devem ser enviadas através de sinais sonoros, adequados ao canal telefônico.

Com base nisso usou-se o microcontrolador para gerar 2 sons de freqüências diferentes, um de 380Hz e outro de 1100Hz, e foram adotadas as seguintes convenções:

Som de 1100Hz durante 0,3 seg. seguido de pausa de 0,15 seg. seguido de mais um som de 1100Hz durante 0,3 seg – significa que o deve ser digitado o número do canal desejado.

Som 1100Hz durante 0,3 seg – significa que o canal está ligado

Som 380Hz durante 0,3 seg – significa que o canal está desligado

Som 380Hz durante 0,6 seg – significa que a ligação telefônica vai ser encerrada

Essas freqüências são injetadas na linha telefônica através de um transistor com resistor de pull-up no emissor e capacitor de desacoplamento.

2.1.3 HARDWARE DO MÓDULO PRINCIPAL

Principais componentes do módulo central:

- Microcontrolador - AT89S52
- Decodificador DTMF – MT8870
- Memória SRAM 32KB – ICM62256M-10
- Sensor de temperatura – LM35
- Conversor A/D – TLC0820
- Display de cristal líquido – LCD
- Teclado de telefone

2.1.4 DIAGRAMA DO CIRCUITO

Para fazer o desenho dos circuitos foi utilizado o aplicativo CIS Capture, do programa ORCAD. O diagrama esquemático completo do módulo principal encontra-se no apêndice 1 e a seguir serão detalhados os principais pontos do mesmo. O código assemby que roda no microcontrolador encontra-se no apêndice 3.

2.1.4.1 DECODIFICADOR DTMF

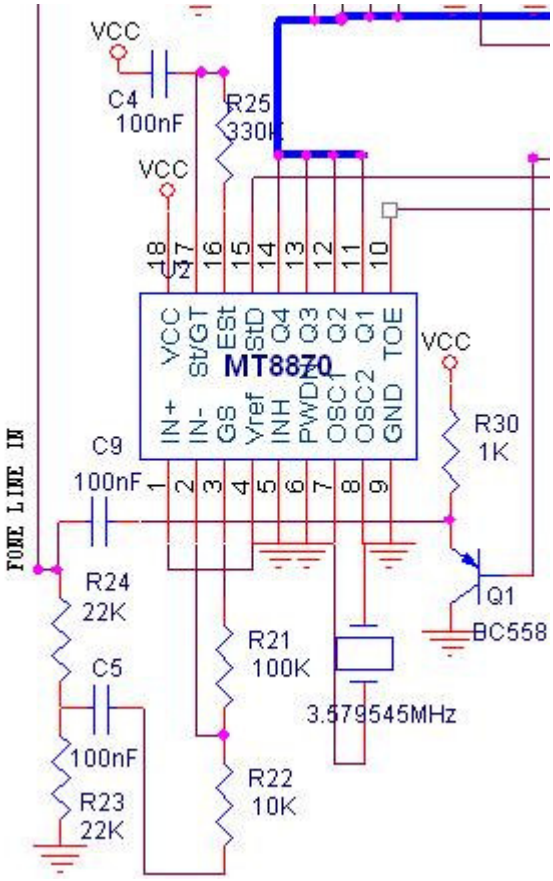


Figura 3 – Detalhe do Decodificador

O circuito é bastante parecido com o sugerido pelo data-sheet do fabricante do MT8870. No ponto “FONE LINE IN” temos conectado diretamente um dos fios da linha

telefônica, o outro fio é ligado ao terra. Os resistores R24 e R23 formam um divisor de tensão, e o capacitor C5 serve para desacoplamento. Internamente o CI possuiu um amplificador operacional e uma tensão de referência, Vref, cujo valor equivale a metade de VCC. Com isso os resistores R21 e R22 causam um ganho de 10 vezes no sinal de entrada. O oscilador de 3,579545MHz deve ser bem preciso, com erro menor à 0,1%, para o correto funcionamento do MT8870. O capacitor C4 e o resistor R25 permitem ajustar o tempo mínimo de duração para considerar um sinal DTMF como válido, e foram escolhidos conforme o data-sheet do fabricante, para ficar em concordância com os padrões do sistema de telefonia.

O resistor R30, o transistor Q1 e o capacitor C9, de desacoplamento, servem para injetar as freqüências das sinalizações sonoras provenientes do microcontrolador na linha telefônica.

2.1.4.2 SENSOR DE TEMPERATURA E CONVERSOR A/D

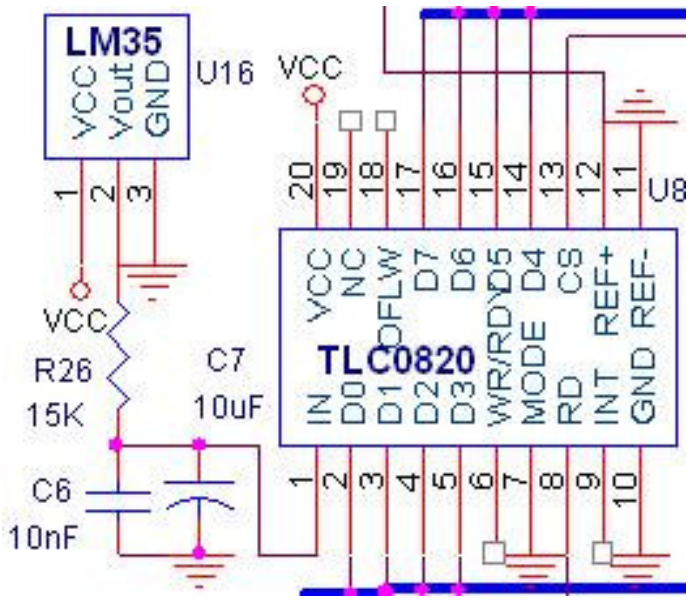


Figura 4 – Detalhe do Sensor e Conversor

O sensor escolhido foi o LM35, com encapsulamento TO-3. Esse sensor tem uma saída em tensão que é linear com a variação de temperatura, e onde 10ºC correspondem a 10mV. Deve ser montado do lado de fora da caixa do equipamento, pois o calor dissipado pelo transformador, regulador e os demais componentes do circuito afetariam na medição da temperatura ambiente.

O filtro passa-baixas de entrada do A/D serve para evitar o "Alising" na conversão do sinal, e para não permitir que o microcontrolador perceba mudanças muito bruscas de temperatura. Levando em conta que a temperatura é atualizada no display somente a cada segundo, e que variações de temperatura ambiente costumam ser lentas, foram escolhidos componentes que dessem uma freqüência de corte do filtro aproximadamente igual a 1Hz (período 1 segundo):

freq.corte = $1/(2\pi RC)$ em Hz

freq.corte = 1/(2 x 3,1415 x 15K x 10u) = 1,06 Hz

Periodo = 1/1,06 ≈ 1 segundo

O conversor A/D usado possuiu vários modos de funcionamento e o modo escolhido permite fácil comunicação com o microcontrolador, permitindo que os dados sejam lidos como se fosse uma memória, usando os pino RD, CS e o barramento de dados. È um conversor de 8 bits, porém devido ao erro de máximo de ± 1 LSB somente temos 7 bits significativos. Isso não é um problema, já que a resolução no display é de 1ºC, e podem ser mostradas temperaturas de 0ºC até 51ºC. Internamente o programa faz a média da temperatura coletada com as 4 anteriores.

2.1.4.3 CIRCUITO DE RELÉS

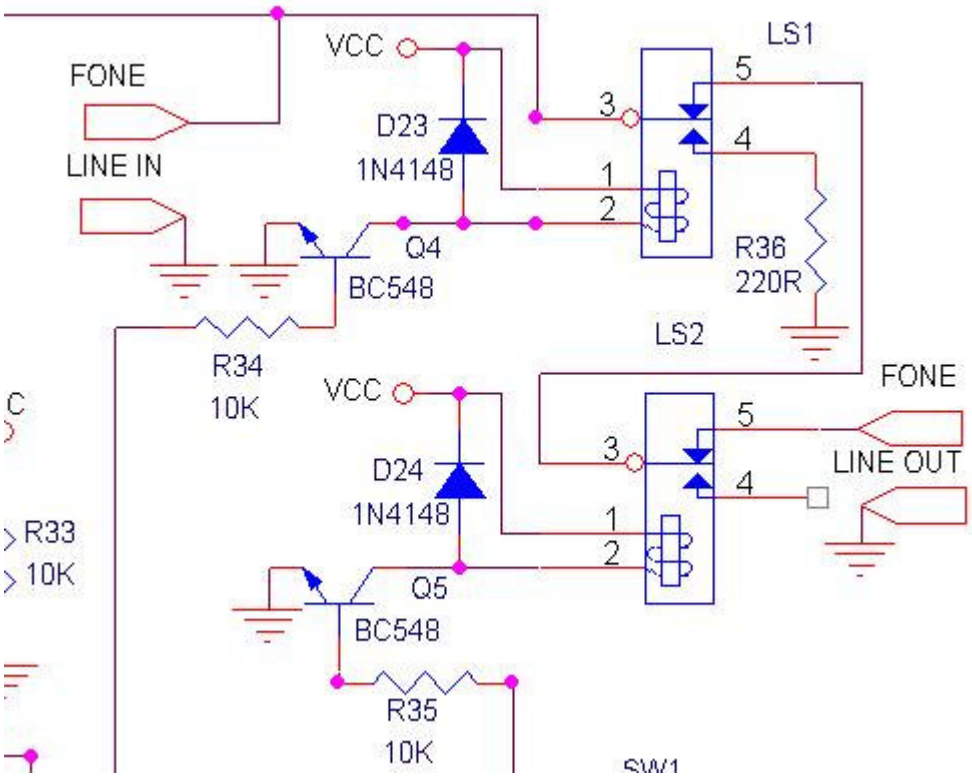


Figura 5 – Detalhe do Circuito de Relés

O primeiro relé é responsável por colocar uma resistência em paralelo com a linha telefônica (R36), permitindo dessa forma o atendimento da chamada. Ao mesmo tempo desconecta a linha telefônica de um telefone que eventualmente estaria ligado no conector de saída de linha telefônica. Com isso não há a possibilidade de alguém levantar o gancho do telefone e interferir na comunicação do sistema com o telefone remoto que originou a chamada.

O segundo relé simplesmente permite conectar ou não a linha telefônica ao telefone, conforme já comentado anteriormente, possibilitando programar horários que não queremos receber chamadas no telefone. De forma independente, o acesso remoto ao sistema está sempre ativo.

2.1.4.4 SAÍDA SERIAL

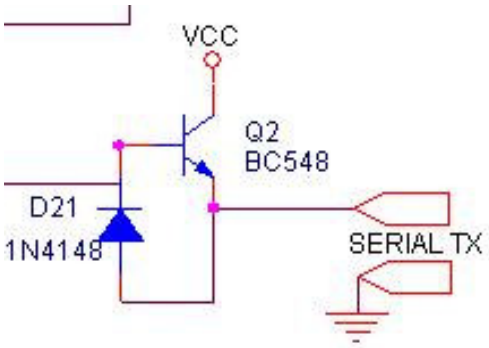


Figura 6 – Detalhe da Saída Serial

Os pinos de saída do microcontrolador usado são do tipo coletor aberto, com resistor de "pull-up", e podem fornecer, em nível baixo, no máximo 10mA. Cada módulo de acionamento consome aproximadamente 3mA, e num caso limite de 20 módulos, teríamos um consumo de 60mA. Logo é necessário ter um buffer para o nível alto. Foi escolhido o transistor de uso geral BC548, que suporta até 100mA.

2.1.5 CUSTOS DO MÓDULO PRINCIPAL

Na confecção do equipamento não foi utilizado nenhum componente de custo excessivamente alto, abaixo está uma relação de preços de cada peça:

Componente	Custo (unidade) R$	Qntde	Custo R$
AT89S52	20,00	01	20,00
M62256	8,00	01	8,00
MT8870	5,00	01	5,00

LM35	7,20	01	7,20
TLC0820	*	01	*
74573	2,00	04	8,00
74138	1,40	01	1,40
7400	0,95	01	0,95
7402	0,95	01	0,95
LCD	28,00	01	28,00
BC548	0,30	03	0,90
BC558	0,30	02	0,60
LEDs 3mm	0,20	20	4,00
Resistores	0,03	34	1,02
Potenciômetros	2,50	02	5,00
Capacitores cerâmicos	0,10	21	2,10
Capacitores eletrolíticos	1,50	04	6,00
Cristais	2,50	02	5,00
Relê	5,30	02	10,60
L7805	1,40	01	1,40
Dissipador	2,50	01	2,50
Teclado	15,00	01	15,00
Placa circuito impresso	15,50	01	15,50
Ponte Retificadora	1,00	01	1,00
Diodos 1N4148	0,10	03	0,30
Transformador 400mA	8,00	01	8,00
Fio plug de tomada	3,00	01	3,00
Chave 110/220	0,90	01	0,90

Conector telefônico	2.00	02	4,00
fios	0,80	5	4,00
Soquete torneado 40p	2,00	01	2,00
Soquete torneado 28p	1,45	01	1,45
Soquete torneado 20p	1,00	05	5,00
Soquete torneado 18p	0,95	01	0,95
Soquete torneado 16p	0,85	03	2,55
Soquete torneado 14p	0,75	02	1,50
Barra de pinos	2,00	01	2,00
Caixa plástica	8,30	01	8,30
TOTAL			**194,07**

Tabela 1 – Custos do Módulo Principal

O conversor A/D usado, o TLC0820 foi adquirido da Texas Instruments como amostra grátis, por isso não foi contabilizado no preço total, mas seu preço é de 2 dólares cada, comprando mil unidades.

2.2 SISTEMA DE TRANSMISSÃO

O sistema de transmissão é serial com par de fios, e velocidade de 300 Bauds. A comunicação é unilateral, de forma que o módulo principal só envia, e os módulos de acionamento só recebem informação. A conexão do par de fios aos módulos é feita através de

conector com parafuso, bastando descascar as pontas do fio, inserir no conector e apertar o parafuso. Como será visto adiante não importa a polaridade com a qual os fios são inseridos, facilitando assim a instalação pelo usuário.

Um bite apenas carrega a informação de qual canal estamos querendo acionar e se é para ligar ou desligar o mesmo. Os primeiros 5 bits do bite carregam a informação do número do canal, e o último bit, o menos significativo carrega a informação de ligar (1) ou desligar (0) o canal. O sexto e o sétimo bit não são usados. Por exemplo, o bite 10010001 significa ligar o canal 18.

A precisão de acionamento de um canal é de 1 segundo. Logo dentro de 1 segundo podem trafegar 20 bites. Como na transmissão serial são inseridos os bits de inicio e fim de bite, um bite fica com o tamanho aproximado de 10 bits. Multiplicando por 20 canais temos 200 bits por segundo, ou 200 bauds de banda necessária. Como os bits de início e final geralmente são um pouco mais largos que o normal e há pequenos atrasos de processamento, a banda deve ser um pouco maior que 200 bauds. Usando o critério de estabelecer a menor velocidade de transmissão necessária, para evitar erros, e respeitando a banda mínima necessária, foi escolhida a velocidade de transmissão de 300 bauds.

2.3 MÓDULOS DE ACIONAMENTO

Os módulos de acionamento das cargas serão responsáveis por ligar e desligar os dispositivos a eles conectados, que podem ser lâmpadas, ventiladores, eletroeletrônicos ou qualquer outro eletrodoméstico. Dessa forma do ponto de vista do módulo central, cada módulo de acionamento será um canal a ser comandado.

Abaixo se tem a foto de um módulo de 1 canal e também de um módulo de 3 canais, fornecendo 3 saídas de tomada, cada uma correspondendo a um canal. Ambos foram projetados para serem ligados em uma tomada de parede.

Figura 7 – Módulo de Acionamento de 1 Canal

Figura 8 – Módulo de Acionamento de 3 Canais

2.3.1 FLUXOGRAMA E FUNCIONAMENTO

A seguir é mostrado o fluxograma de funcionamento do módulo de acionamento de 3 canais. O funcionamento de um módulo de apenas um canal é bastante semelhante.

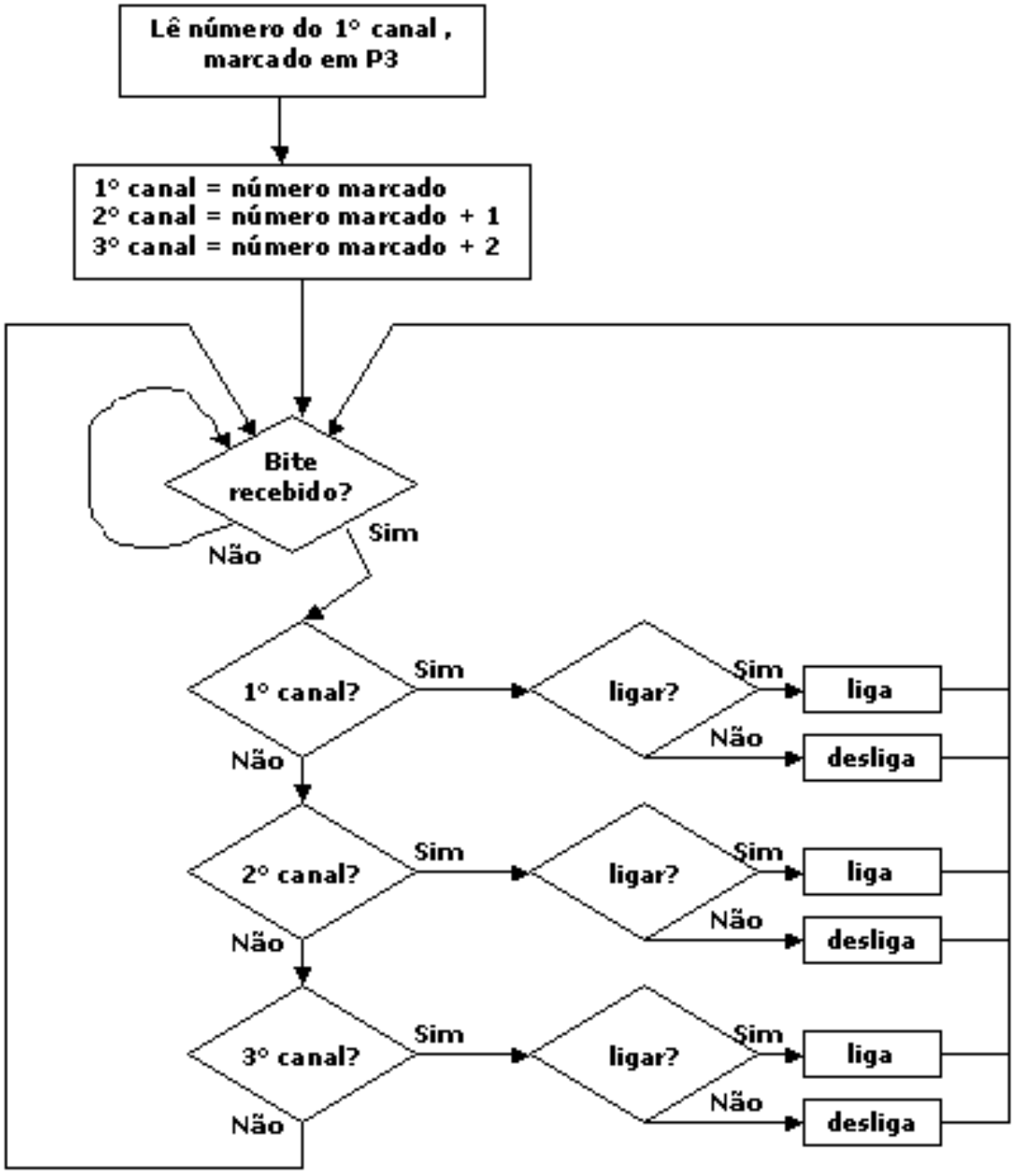


Figura 9 – Fluxograma do Módulo de Acionamento

Quando o módulo de acionamento é conectado à tomada ele é iniciado. A primeira ação é coletar informação de qual será o número do canal que acionará a primeira tomada de força do módulo. O canal seguinte acionará a segunda tomada e o seguinte a terceira. Logo a ordem de associação de canais num módulo é sempre seqüencial. A seleção do canal inicial é

feita através de “jumpers” conectados aos pinos P3.3, P.4, P3.5, P.6 e P3.7, formando uma combinação de 5 bits, o suficiente para selecionar qualquer um dos 20 canais. Os “jumpers” colocam o bit em zero e a ausência do jumper coloca o bit em 1 devido ao resistor interno de pull-up do microcontrolador. P3.3 é o bit mais significativo e P3.7 o menos significativo.

Após a coleta do número do canal o módulo fica aguardando ser recebido um bite. Quando um bite é recebido o sistema verifica se o canal corresponde a algum dos canais definidos para o módulo em questão. Se não corresponder a nenhum dos 3 canais do módulo o sistema volta ao modo de espera. Se corresponder a um canal, o módulo verifica se o bit menos significativo tem valor 1 ou 0, ligando ou desligando, respectivamente, o canal associado. Após volta ao estado de espera por um novo bite.

2.3.2 HARDWARE

Principais componentes do módulo de acionamento:

- Microcontrolador - AT89S51
- Acoplador óptico 4N35
- Driver de triac MOC3023
- Triac BT139

Para uma montagem mais compacta foi usado um divisor capacitivo e retificador para gerar a tensão de alimentação do circuito

2.3.3 DIAGRAMA DO CIRCUITO

O diagrama esquemático completo do módulo de acionamento encontra-se no apêndice 2. O código assembly encontra-se no apêndice 4. A seguir serão detalhados os principais pontos do mesmo:

2.3.3.1 ALIMENTAÇÃO

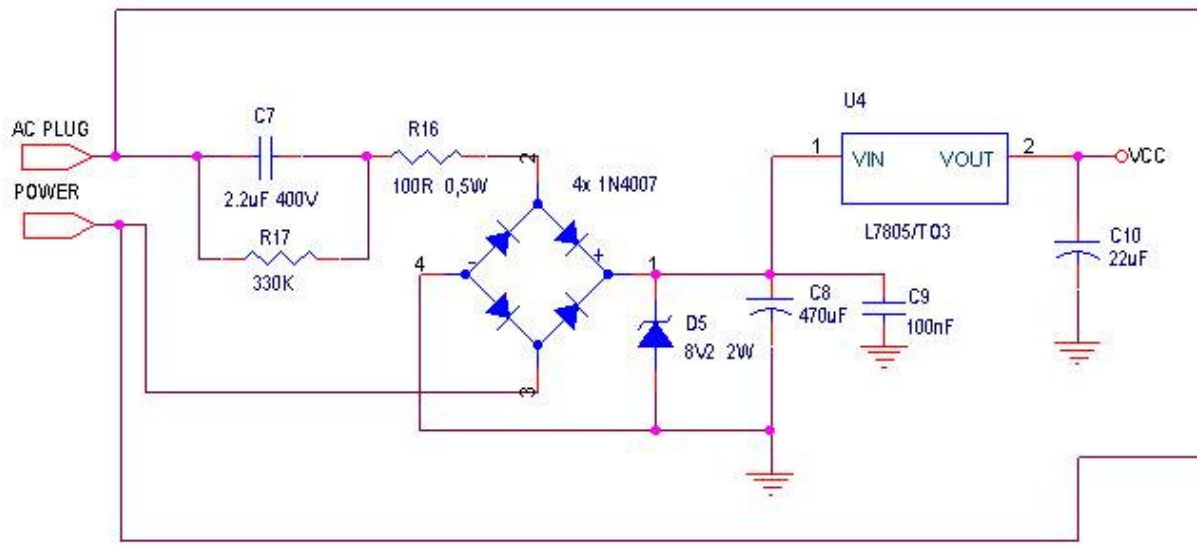


Figura 10 – Detalhe da Alimentação

Para uma montagem mais compacta, dispensando o uso de um transformador foi usado um divisor capacitivo. Com isso a tensão fica dividida entre a ponte retificadora e o capacitor. Para limitar a tensão na saída da ponte foi usado em diodo zenner de 8,2 volts. O capacitor C8 serve como filtro e deve manter o ripple pequeno o suficiente para que a tensão não caia abaixo de 6,5 volts, tensão mínima exigida na entrada do regulador de 5V.

O resistor R17, em paralelo com o capacitor, é importante, pois tem a função de descarregar o capacitor quando o módulo é retirado da tomada, evitando assim que o usuário

leve um choque ao encostar-se nos plugues de alimentação do módulo. O resistor R16 em série com o capacitor serve para limitar os picos de corrente que surgem na ponte retificadora quando o módulo é conectado à tomada.

O módulo é bivolt, e a corrente que podemos drenar do ciruito de alimentação depente da tensão da rede e do valor escolhido para o capacitor C7. Supondo que a rede seja de 127V (caso em que poderá fornecer menos corrente), e lembrando que I = V/Z e $|Z| = 1/2\pi fC$, temos Icurto = 120x2x3,1415x60x2,2uF = 100mA. A corrente drenada máxima testada que ainda mantém a tensão acima 6,5V, para o funcionamento do regulador foi de 35mA, mais do que suficiente para a alimentação do circuito.

2.3.3.2 ENTRADA SERIAL

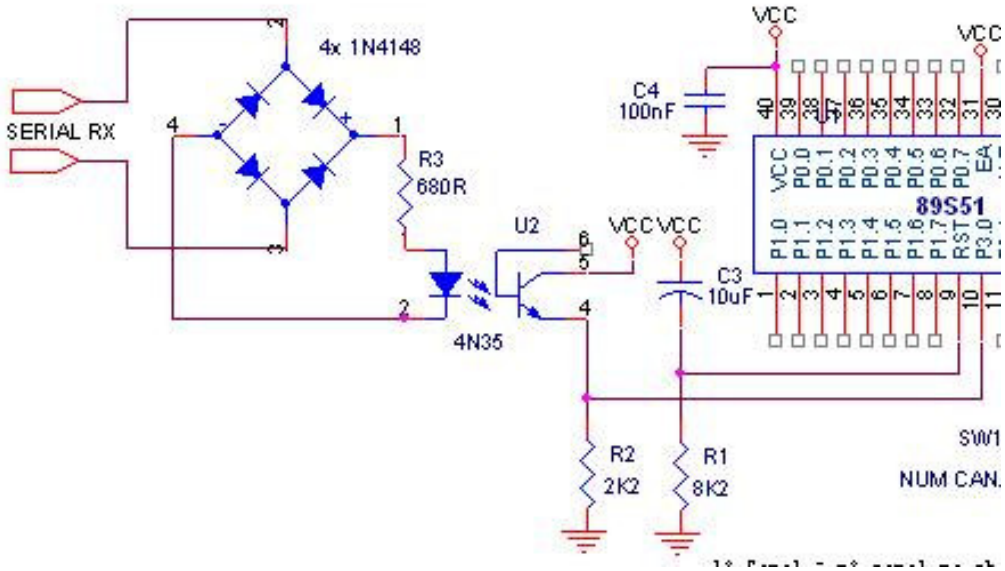


Figura 11 – Detalhe da Entrada Serial

Para maior facilidade na conexão dos fios de entrada foi usada uma ponte retificadora, feita com diodos de sinal, não importando assim a polaridade de entrada. Foi utilizado um

acoplador óptico para que ruídos da rede elétrica não fossem acoplados a linha telefônica. O valor de R3 foi escolhido de forma que a corrente no emissor do acoplador ficasse em 3mA. Levando em conta a queda de tensão no transmissor, de 0,6V, a queda de tensão na ponte, de 1,2V e a queda de tensão no Led emissor, de 1,2V, temos que R = (5 – 0,6 – 1,2 – 1,2) / 0,003 = 666,67, logo foi utilizado um resistor de valor comercial de 680Ω.

O resistor R2 deve ter um valor bem menor que o resistor interno de pull-up (≈15KΩ) do microcontrolador, e foi calculado para que passasse uma corrente de aproximadamente 2mA no fototransistor do 4N35.

O capacitor C3, de 10uF, e o resistor R1, de 8,2KΩ, que também aparecem no detalhe formam o circuito de auto-reset do microprocessador quando o mesmo é alimentado.

2.3.3.3 ETAPA DE POTÊNCIA

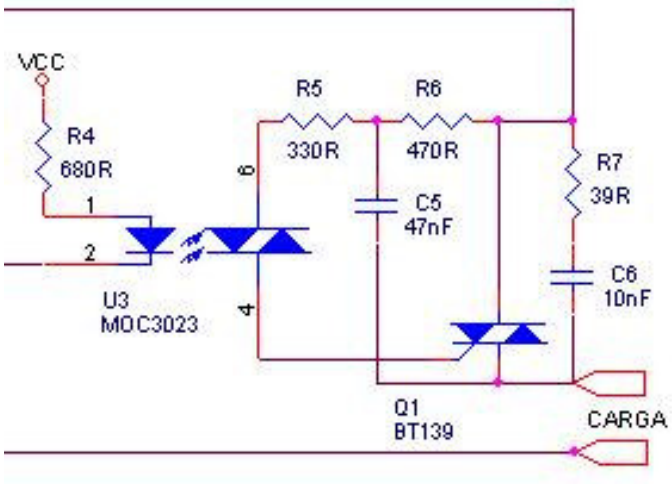


Figura 12 – Detalhe da Etapa de Potência

Para o acionamento da carga foi escolhido o triac BT139. Para calcular a potência máxima que ele pode dissipar, sem uso de dissipador, temos os seguintes dados: Resistência

térmica junção-ambiente, Rth = 60 K/W. Temperatura máxima na junção, Tj = 125ºC. Temperatura ambiente, Tamb = 45ºC.

P = (Tj - Tamb) / Rth

P= (125 - 45) / 60 = 1,333 Watts

Com base nessa potência calculada e na queda de tensão entre os terminais, de 1,2V, calculamos a corrente máxima que pode circular pelo triac:

I = P / V

I = 1,333 / 1,2 = 1,111A

Com isso temos que a potência máxima de saída é de 1,111x127 = 141 Watts na rede de 127V e 1,111x220 = 244 Watts na rede de 220V. Essas potências são mais do que suficientes para acionar a maioria dos eletrodomésticos presentes nas residências.

No módulo de um canal foi usado na etapa de potência um relé ao invés de triac, com limitação de tensão de 250V e corrente de 10A, permitindo dessa forma acionar cargas de até 1270W em 127V e 2200 W em 220V.

O acionamento do triac foi feito através do acoplador MOC3023, sendo que o valor dos componentes foi escolhido com base no data-sheet do fabricante. O circuito inclui ainda um snubber, para reduzir a possibilidade de disparos falsos.

O resistor R4 foi calculado de forma que passassem pelo menos 5mA no emissor, o mínimo recomendado no data-sheet. R4 = (5 – 1,2) / 0,005 = 760, valor comercial = 680Ω.

2.3.4 CUSTOS DO MÓDULO DE ACIONAMENTO

Abaixo encontra-se uma tabela com os preços das peças usadas na montagem do módulo de acionamento de 3 canais:

Componente	Custo (unidade) R$	Qntde	Custo R$
AT89S51	8,00	01	8,00
4N35	1,35	01	1,35
MOC3023	2,00	03	6,00
BT139	3,60	03	10,80
L7805/TO-3	0,90	01	0,90
1N4148	0,10	04	0,40
1N4007	0,10	02	0,20
Zenner 8V2 1W	0,30	02	0,60
Cristal	2,50	01	2,50
Resistores	0,03	15	0,45
Capacitores cerâmicos	0,10	03	0,30
Demais capacitores		10	6,25
Conector	2,00	01	2,00
Tomada de força	3,00	03	9,00
Caixa plastica	4,00	01	4,00
Placa de cicuito	8,00	01	8,00
Suporte torneado 40p	2,00	01	2,00
Suporte torneado 6p	0,30	04	1,20
TOTAL			**63,95**

Tabela 2 – Custos do Módulo de Acionamento de 3 Canais

2.4 MONTAGEM DO EQUIPAMENTO

Antes da montagem final o circuito foi montado e testado em proto-board, como pode ser visto abaixo:

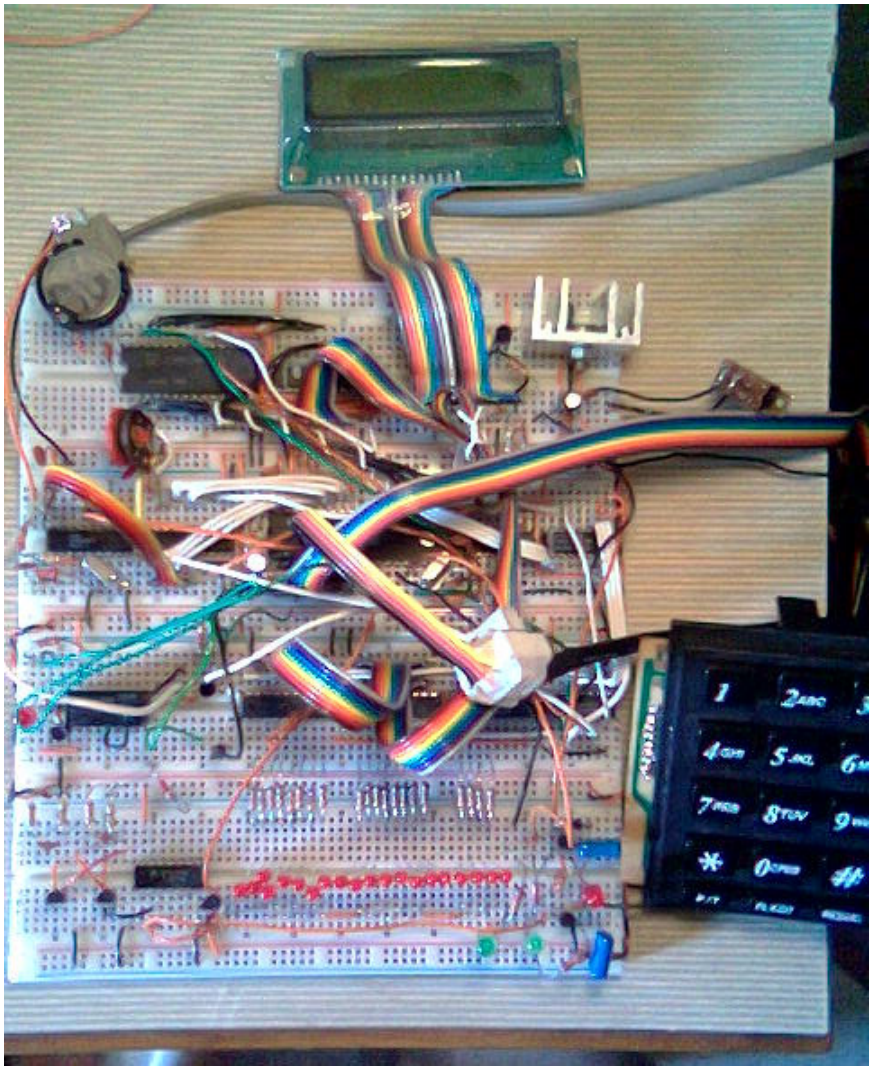


Figura 13 – Proto-board do Módulo Principal

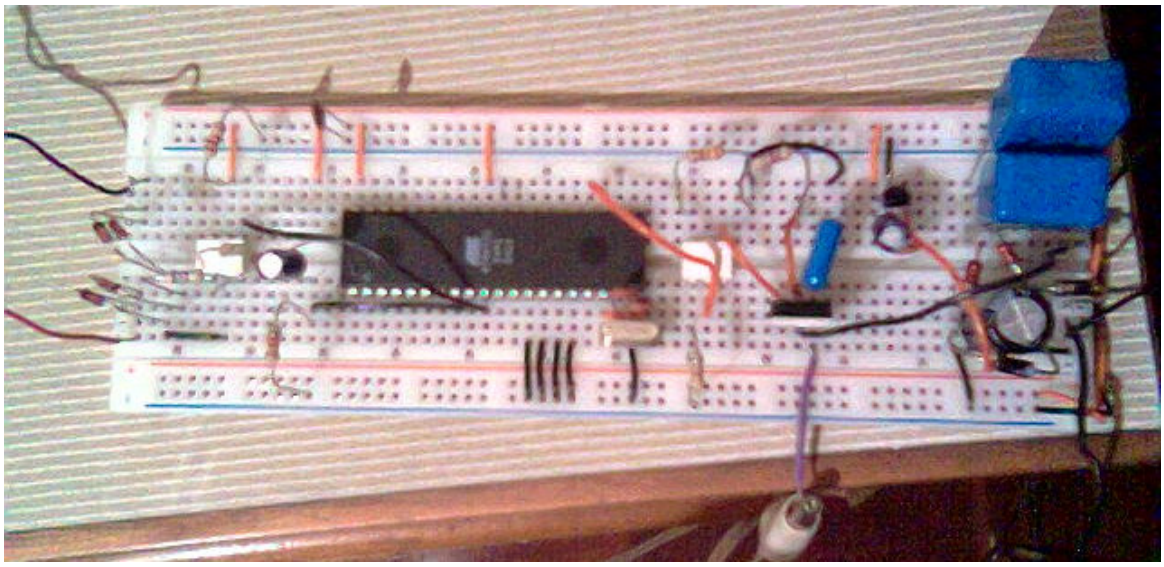


Figura 14 – Proto-board do Módulo de Acionamento

Para a montagem final do módulo principal foi usada uma placa de circuito impresso pré-furada de 15x12cm, com trilhas dispostas de forma semelhante ao proto-board. A placa teve que ser serrada para caber dentro da caixa plástica escolhida (modelo PB119), ficando com o tamanho de 15x10,5cm. O fato de usar uma placa pré-furada permitiu uma montagem compacta, visto o grande número de barramentos do circuito. Primeiramente foram dispostos os suportes torneados de forma conveniente, e depois soltados os fios correspondentes aos barramentos. Depois foram soldados os componentes menores e por último o transformador.

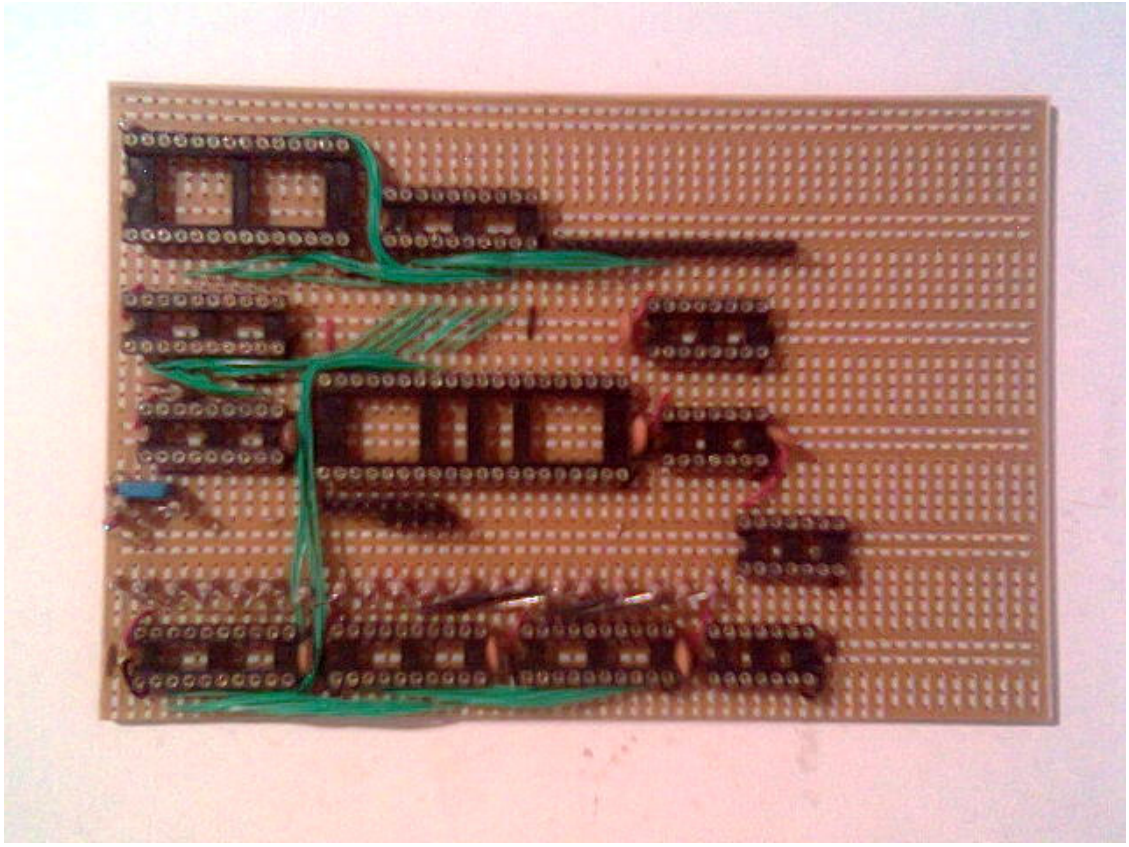


Figura 15 – Construção da Placa do Módulo Principal

A caixa plástica do módulo principal foi perfurada com broca de 3mm para os Leds e as demais aberturas foram feitas com uma pequena serra circular de corte.

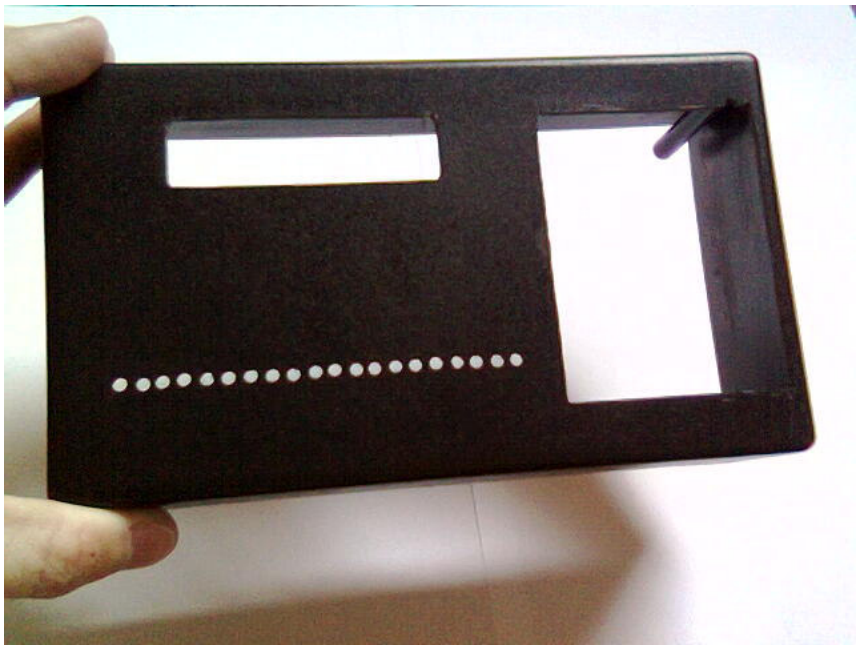


Figura 16 – Construção da Caixa do Módulo Principal

Figura 17 – Construção do Módulo de Acionamento de 3 Canais

Figura 18 – Construção do Módulo de Acionamento de 1 Canal

2.5 MODO DE GRAVAÇÃO

O sistema usa um microcontrolador da família 8051, o 89S52 para o módulo principal e o 89S51 para os módulos de acionamento, e em ambos o modo de gravação é ISP (In-System Programmable), ou seja, pode ser facilmente programado, apenas conectando um cabo ao equipamento, não necessitando ser retirado do mesmo para sua programação. Isso é uma grande vantagem, pois permite que o software seja carregado rapidamente no microprocessador, agilizando o teste de funcionamento do equipamento enquanto o código vai sendo escrito.

No módulo principal teve-se que usar o 89S52, que tem 8KB de memória de programa interna, já que o código assembly ficou com um tamanho de 5,5KB. Já nos módulos de acionamento, onde o código assembly ficou com apenas 143 bites foi usado o 89S51, que tem 4KB de memória, mais do que suficiente para carregar o programa.

Nos apêndices 3 e 4 temos na integra os códigos do módulo principal e do módulo de acionamento de 3 canais. O código de ambos foi totalmente escrito em assembly.

3 IMPLEMENTAÇÕES FUTURAS

O equipamento funcionou bem dentro do que foi proposto, porém sempre há a possibilidade de melhorias e expansão. O sistema ainda tem sobra de memória RAM e espaço disponível para mapeamento em memória de mais componentes. Possíveis implementações futuras poderiam ser:

- Utilização do equipamento como bina, aproveitando o decodificador DTMF e registro de chamadas recebidas, não atendidas e discadas, aproveitando ao máximo a memória RAM disponível.
- Transmissão de dados usando a rede elétrica da casa, aproveitando a alta capilaridade proporcionada pelos fios de energia já existentes.
- Integração com sensores de passagem e de presença, permitindo acionamento de iluminação dos cômodos da residência de forma automática e conforme a necessidade.
- Usar um transmissor de rádio freqüência como controle remoto, sendo outra alternativa além do comando por chamada telefônica.

4 CONCLUSÃO

A execução de um Projeto de Conclusão permite aplicar o conhecimento adquirido ao longo do curso, e os diversos conteúdos vistos nas cadeiras cursadas. A montagem de um produto final proporciona ao aluno vivenciar na prática as dificuldades e imprevistos que podem ocorrer durante a confecção de um equipamento, desde a escolha dos componentes, levando em conta custo e facilidade de obtenção, até a adaptação de materiais de acordo com nossas necessidades, mostrando que um projeto de engenharia envolve não somente a parte técnica, mas também a parte financeira e visão de mercado.

A elaboração do trabalho de conclusão lança diversos desafios, porém, a cada etapa do projeto realizada com êxito tomamos ânimo para seguir adiante e crescemos com as experiências adquiridas.

O sucesso na finalização desse trabalho mostra não somente a capacidade do aluno, mas também a preparação proporcionada pelo curso de Engenharia Elétrica da UFRGS e reflete o esforço de excelentes professores que se preocupam com o aprendizado do universitário.

5 REFERÊNCIAS

CARRO, LUIGI. **Projeto e Prototipação de Sistemas Digitais.** Porto Alegre, Ed. Universidade/UFRGS, 2001, ISBN 85-7025-589-6. 93p

BLAUTH, YEDDO BRAGA. **Apostila de Aplicações Industriais da Eletrônica.** Porto Alegre, 2000. 12p

NICOLOSI, DENYS EMÍLIO CAMPION. **Microcontrolador 8051 Detalhado.** 5ª Edição, São Paulo, Érica, 2000, ISBN: 85-7194-721-X

ROSA, BRÁS FELIPE PATTA; TESSMER, VICENTE R. **Apostila: Curso Básico de OrCAD.** Porto Alegre, Semana Acadêmica UFRGS, 2005.

SEDRA, ADEL S.; SMITH, KENNETH, C. **Microeletrônica.** 4ª Edição, Pearson Education do Brasil, 2000, ISBN: 85-346-1044-4

Datasheet do microcontrolador AT89S52

Datasheet do conversor TLC0820

Datasheet do componente MT8870

Datasheet do componente MOC3023

Datasheet do componente 4N35

Datasheet do triac BT139

Datasheet da memória M62256

Datasheet do controlador de LCD HD44780U

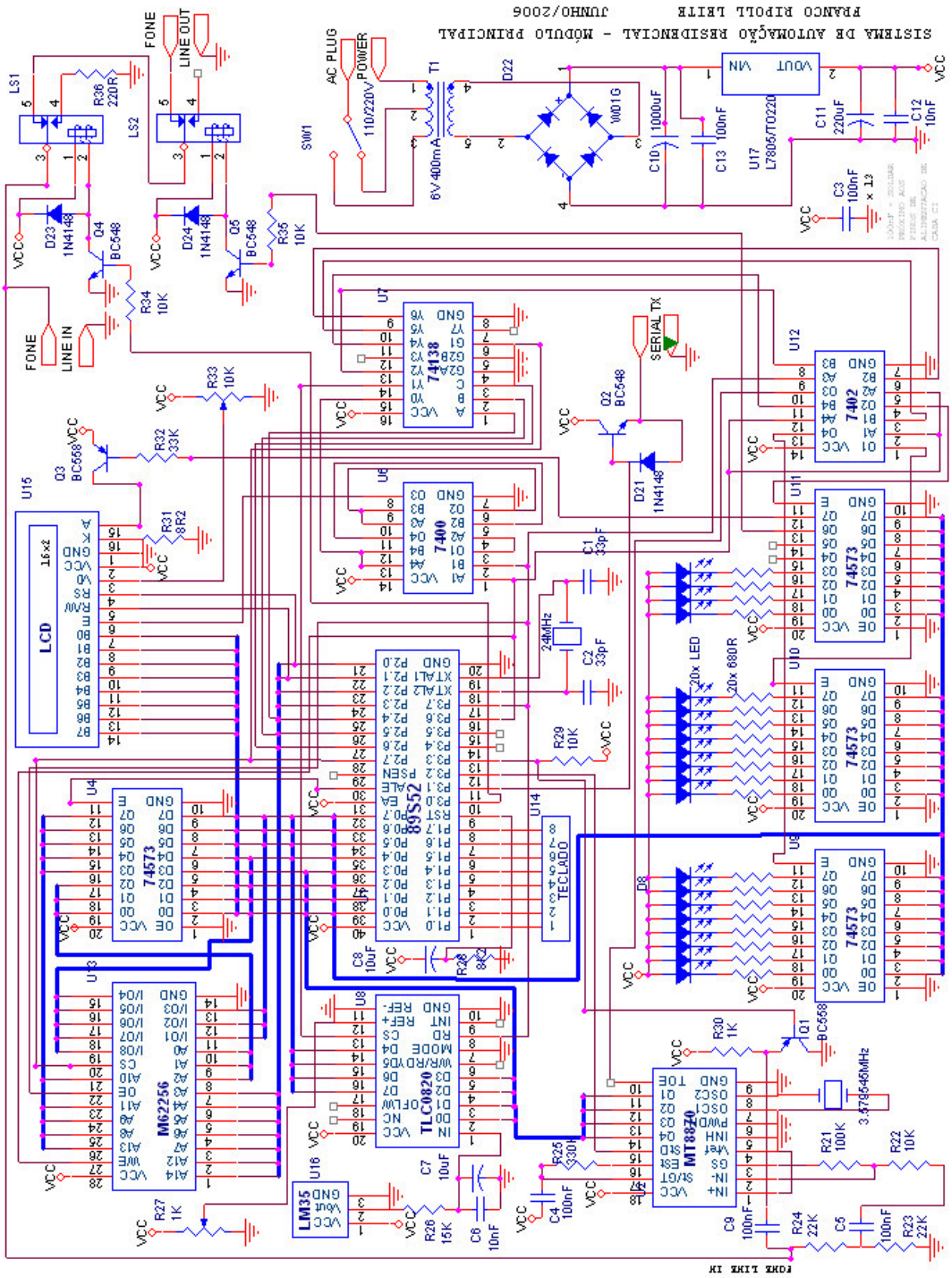
Datasheet do componente 74LS00

Datasheet do componente 74LS02

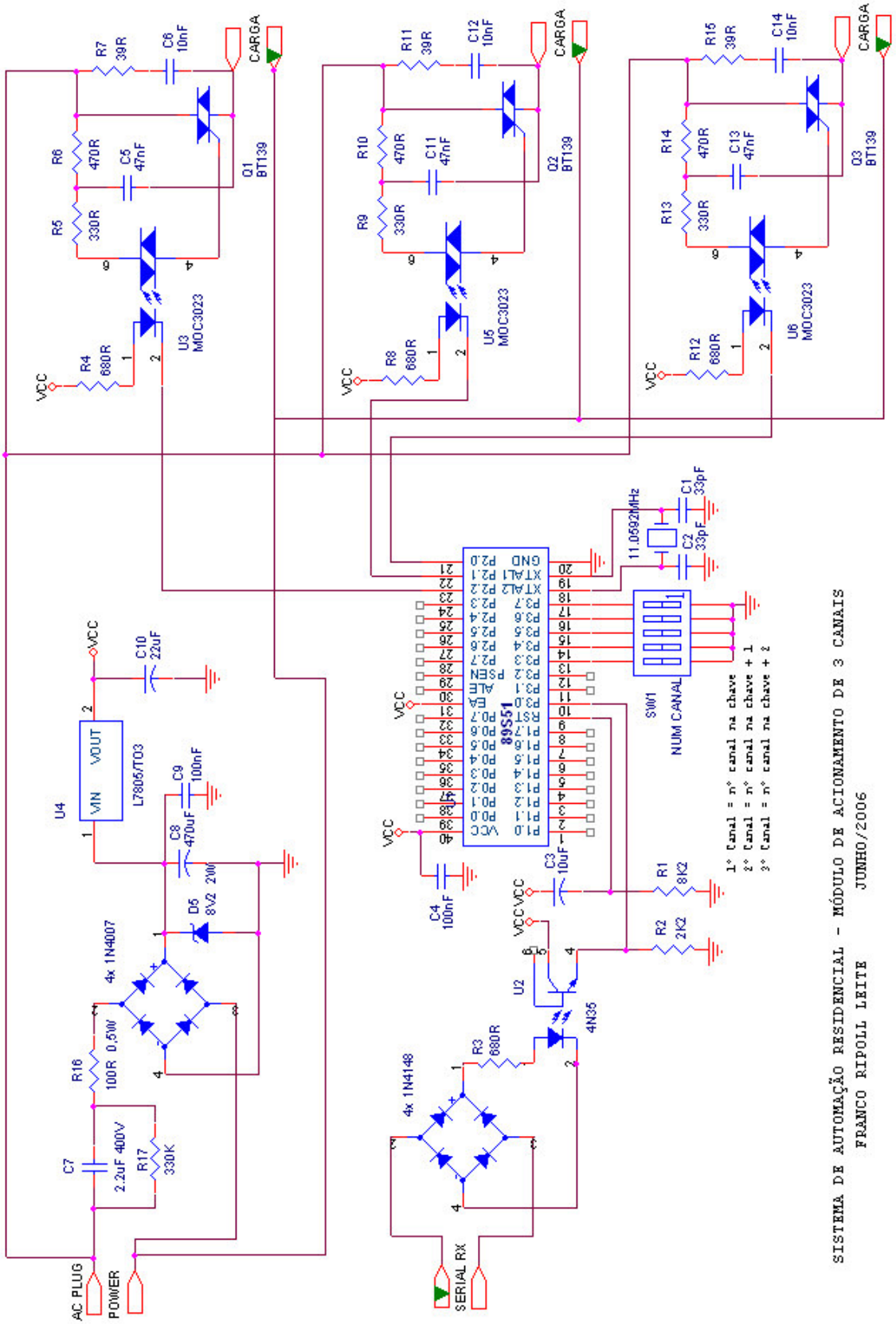
Datasheet do componente 74HC138

Datasheet do componente 74HC573

Apêndice 1


SISTEMA DE AUTOMAÇÃO RESIDENCIAL - MÓDULO PRINCIPAL
FRANCO RIPOLL LEITE JUNHO/2006
AC PLUG
POWER
110/220V
6V 400mA
T1
D22
W01G
C10 1000uF
C13 100nF
U17 L7805/TO220
C11 220uF
C12 10nF
C3 100nF x 13
VCC
LS1
LS2
FONE
LINE OUT
LINE IN
R36 220R
D23 1N4148
D24 1N4148
Q4 BC548
Q5 BC548
R34 10K
R35 10K
R33 10K
R32 33K
Q3 BC558
R31 8R2
U15
LCD 16x2
U7
74138
U6
7400
U12
7402
U11
74573
U10
74573
U9
74573
20x LED
20x 680R
D8
SERIAL TX
Q2 BC548
D21 1N4148
C1 33pF
C2 33pF
24MHz
R29 10K
U14
TECLADO
U4
74573
U3
M62256
U8
TLC0820
C7 10uF
C8 10uF
R28 8K2
U16
LM35
R27 1K
R26 15K
C6 10nF
C4 100nF
R25 330R
Q1 BC558
R30 1K
U18
MT8870
3.579545MHz
R21 100K
R22 10K
C9 100nF
R24 22K
C5 100nF
R23 22K

Apêndice 2


SISTEMA DE AUTOMAÇÃO RESIDENCIAL - MÓDULO DE ACIONAMENTO DE 3 CANAIS
FRANCO RIPOLL LEITE JUNHO/2006
1° canal = n° canal na chave
2° canal = n° canal na chave + 1
3° canal = n° canal na chave + 2
CARGA
Q1 BT139
Q2 BT139
Q3 BT139
U3 MOC3023
U5 MOC3023
U6 MOC3023
U4 L7805/TO3
U2 4N35
89S51
SW01 NUM CANAL
11.0592MHz
4x 1N4007
4x 1N4148
AC PLUG
POWER
SERIAL RX
VCC

Apêndice 3

```
; ========================================================================
; Sistema de Automação Residencial - Módulo Principal
; Franco Ripoll Leite
;========================================================================
;Mapeamento de 8 periféricos com 74HC138:
lcd_wr_inst     equ     08000h ;1000
lcd_wr_dado     equ     08100h
lcd_rd_inst     equ     08200h
lcd_rd_dado     equ     08300h

AD              equ     09000h ;1001

DTMF            equ     0A000h
;vazio          equ     0B000h

REGISTRADOR1EX  equ     0C000h
REGISTRADOR2EX  equ     0D000h
REGISTRADOR3EX  equ     0E000h

;vazio          equ     0F000h

;Mapeamento da memória SRAM de 32KB:
PROGCANAIS      equ     00000h
;TAMANHOMEIOCANAL       equ     8
TAMANHOCANAL    equ     16

NOMES           equ     384     ;16x24
NOMESH          equ     HIGH(NOMES)
NOMESL          equ     LOW(NOMES)
TAMANHONOME     equ     10

FIMNOMES        equ     584     ;fim nomes 10x20
FIMNOMESH       equ     HIGH(FIMNOMES)
FIMNOMESL       equ     LOW(FIMNOMES)

;memória interna:
LIN1            equ     P1.6;P1.7       ;pinos correspondentes ao teclado
LIN2            equ     P1.7;P1.6
LIN3            equ     P1.5;P1.5
LIN4            equ     P1.4;P1.4
COL1            equ     P1.1;P1.0
COL2            equ     P1.2;P1.1
COL3            equ     P1.3;P1.2
LIN5            equ     P1.0;P1.3

PINORELE1       equ     P3.0

DADOPRONTO              equ     P3.2    ;indica que foi recebido um DTMF
                                        ;(deve ser o mesmo pino da INT0)
SOM             equ     P3.3

ESPACOSENHAS                    equ     30h
APONTADORSENHA                  equ     ESPACOSENHAS
INICIOSENHA             equ     ESPACOSENHAS+1h
APONTADORSENHAPADRAO    equ     ESPACOSENHAS+8h
INICIOSENHAPADRAO               equ     ESPACOSENHAS+9h
```

```
RSOM1                   equ     40h
RSOM2                   equ     41h
RSOM3                   equ     42h
RSOM4                   equ     43h
RSOM5                   equ     44h
FLAGREG1        equ     45h;41h
SEGUNDOS        equ     FLAGREG1
MINUTOS                 equ     FLAGREG1+1h
HORAS                   equ     FLAGREG1+2h
DIA             equ     FLAGREG1+3h
MES             equ     FLAGREG1+4h
ANO             equ     FLAGREG1+5h
DIASEMANA       equ     FLAGREG1+6h
REGTEMPERATURA  equ     FLAGREG1+7h
;registradores auxiliares (temporários):
AUXSEGUNDOS             equ     FLAGREG1+8h
AUXMINUTOS equ     FLAGREG1+9h
AUXHORAS        equ     FLAGREG1+0Ah
AUXDIA                  equ     FLAGREG1+0Bh
AUXMES                  equ     FLAGREG1+0Ch
AUXANO                  equ     FLAGREG1+0Dh
AUXDIASEMANA            equ     FLAGREG1+0Eh
AUXREGTEMPequ     FLAGREG1+0Fh

VERSEGUNDOS             equ     FLAGREG1+10h
VERMINUTOS equ     FLAGREG1+11h
VERHORAS        equ     FLAGREG1+12h
VERDIA                  equ     FLAGREG1+13h
VERMES                  equ     FLAGREG1+14h
VERANO                  equ     FLAGREG1+15h
VERDIASEMANA            equ     20h;FLAGREG1+16h
VERREGTEMP              equ     FLAGREG1+17h
;Registradores de programação do CH:
CHLSEGUNDOS             equ     FLAGREG1+18h
CHLMINUTOS equ     FLAGREG1+19h
CHLHORAS        equ     FLAGREG1+1Ah
CHLDIA                  equ     FLAGREG1+1Bh
CHLMES                  equ     FLAGREG1+1Ch
CHLANO                  equ     FLAGREG1+1Dh
CHLDIASEMANA            equ     FLAGREG1+1Eh
CHLREGTEMP              equ     FLAGREG1+1Fh
CHDSEGUNDOS             equ     FLAGREG1+20h
CHDMINUTOS equ     FLAGREG1+21h
CHDHORAS        equ     FLAGREG1+22h
CHDDIA                  equ     FLAGREG1+23h
CHDMES                  equ     FLAGREG1+24h
CHDANO                  equ     FLAGREG1+25h
CHDDIASEMANA            equ     FLAGREG1+26h
CHDREGTEMP                      equ     FLAGREG1+27h

NUMTEMP1        equ     FLAGREG1+28h
NUMCANAL        equ     FLAGREG1+29h
AUXNUMCANAL             equ     FLAGREG1+2Ah
VERNUMCANAL             equ     FLAGREG1+2Bh
;TEMPTIMER      equ     FLAGREG1+33h
DPTRH           equ     FLAGREG1+2Ch
DPTRL           equ     FLAGREG1+2Dh
NUMTEMP2        equ     FLAGREG1+2Eh
SEGUNDOSCHAMADA         equ     FLAGREG1+2Fh
;MINUTOSCHAMADA  equ     FLAGREG1+30h
CARACTERANTERIOR        equ     FLAGREG1+31h
```

```
LOCALMENS   equ     FLAGREG1+32h
INICIOMENS  equ     FLAGREG1+33h


;;;VERDIASEMANA      equ     20h;FLAGREG1+16h
FLAGREG2    equ     2Eh
BYTEBIT1    equ     FLAGREG2-1h ;byte que armazena os dias da semana
REGISTRADOR1        equ     FLAGREG2-2h
REGISTRADOR2        equ     FLAGREG2-3h
REGISTRADOR3        equ     FLAGREG2-4h
MEDIA1              equ     FLAGREG2-5h
MEDIA2              equ     FLAGREG2-6h
MEDIA3              equ     FLAGREG2-7h
MEDIA4              equ     FLAGREG2-8h
MEDIA5              equ     FLAGREG2-9h
;REGISTRADOR1P      equ     FLAGREG2-5h
;REGISTRADOR2P      equ     FLAGREG2-6h
;REGISTRADOR3P      equ     FLAGREG2-7h
;REGISTRADOR1B      equ     FLAGREG2-8h
;REGISTRADOR2B      equ     FLAGREG2-9h
;REGISTRADOR3B      equ     FLAGREG2-0Ah

FLAGBIT1    equ     78h     ;2Fh
BITDATA             equ     FLAGBIT1
MOSTRATEMPERATURA   equ     FLAGBIT1+1
BITPISCADIASEMANA equ     FLAGBIT1+2
BITCONTROLE         equ     FLAGBIT1+3
BITLIGADESLIGA      equ     FLAGBIT1+4
SALVAMENS           equ     FLAGBIT1+5
SALVABITDATA        equ     FLAGBIT1+6
;BITTEMPORESTANTE   equ     FLAGBIT1+7

org     0000h
jmp     INICIO
org     0003h
jmp     INTZERO     ;INT0
;org    0013h
;jmp    INTUM ;INT1
org     000bh
jmp     ROTINA      ;TIMER0
;org    001bh
;jmp    ROTINA      ;TIMER1
;org    0023h
;jmp    SERIAL
org     002bh
jmp     TIMER2      ;TIMER2
INICIO:
;###LCD#################################################################
        ;Deve-se esperar pelo menos 30ms antes de iniciar o LCD:
        mov     R0,#200
        mov     R1,#150
        ESPERA30:   ;30ms para 24MHZ
        djnz    R0,ESPERA30
        mov     R0,#200
        djnz    R1,ESPERA30
        ;Inicia o LCD:
        mov     a,#00111000b  ;comunicação de 8bits, 2 linhas, matriz 5x7
        call    WR_INSTRUCAO
        mov     a,#00000110b  ;desloca cursor p/ dir., mensagem não desloca
        call    WR_INSTRUCAO
        call    CURSOROFF
```

```
        call    LIMPA
;#########################################################################
;buscar os dados da memoria externa?
mov     dptr,#MSRECORRER
call    ESCREVE
call    PULALINHA
mov     dptr,#MS1SIM2NAO
call    ESCREVE
clr     LIN1
AQUIRECORRER:
;call    TECLADO
jb      COL1,TESTA2NAO
jmp     NAOFORMATA
TESTA2NAO:
jb      COL2,AQUIRECORRER
call    LIMPA
mov     dptr,#MSINICIANDO
call    ESCREVE

mov     NUMCANAL,#24
LIMPACANAIS:
call    FORMATACANAL
djnz    NUMCANAL,LIMPACANAIS


;Limpa nomes:
mov     dph,#NOMESH
mov     dpl,#NOMESL

;dec     dpl
;dec     dpl

LIMPA02:
mov     a,dph
cjne    a,#FIMNOMESH,LIMPA01
mov     a,dpl
cjne    a,#FIMNOMESL,LIMPA01
jmp     LIMPA03
LIMPA01:
mov     a,#' '
movx    @dptr,a
inc     dptr
jmp     LIMPA02
LIMPA03:

NAOFORMATA:

mov     SCON,#01000000b     ;serial em modo 1
mov     PCON,#00000000b     ;SMOD = 0
        mov     TMOD,#00100001b     ;timer0 no modo 1 e timer1 no modo 2
;       mov     TH0,#HIGH(65535-50000)
;       mov     TL0,#LOW(65535-50000)
        mov     0CEh,#HIGH(25);TH2
        mov     0CDh,#LOW(25);TL2
        mov     0CCh,#HIGH(25);RCAP2H
        mov     0CBh,#LOW(25);RCAP2L
mov     TH1,#48
mov     TL1,#48
        setb    IT0             ;define INT0 por borda de descida
;       setb    IT1             ;define INT1 por borda de descida
        mov     IE,#10100010b ;habilita interrupção por Timer0 e Timer2
```

```
        setb    PT0             ;TIMER0 - prioridade alta
        mov     R2,#0
        mov     SEGUNDOS,#0
        mov     MINUTOS,#0
        mov     HORAS,#14
        mov     DIA,#1
        mov     MES,#1
        mov     ANO,#6
        mov     DIASEMANA,#2

mov     MEDIA2,#20
mov     MEDIA3,#20
mov     MEDIA4,#20
mov     MEDIA5,#20

mov     NUMCANAL,#1

mov     REGISTRADOR1,#255 ;limpa canais
mov     REGISTRADOR2,#255
mov     REGISTRADOR3,#00111111b
call    ATUALIZAREGISTRADOR1EX
call    ATUALIZAREGISTRADOR2EX
call    ATUALIZAREGISTRADOR3EX

mov     APONTADORSENHA,#(INICIOSENHA-1)
mov     INICIOSENHAPADRAO,#'#'

        setb    BITDATA
;       clr     BITTEMPORESTANTE
        setb    MOSTRATEMPERATURA
        clr     BITPISCADIASEMANA

        setb    TR0             ;liga o TIMER0
        setb    TR1             ;liga o TIMER1
setb    0C8h.2;TR2              ;liga o TIMER2
        clr     IE0             ;limpa flag de INT0
;       clr     IE1             ;limpa flag de INT1
        setb    EX0             ;habilita INT0
;       setb    EX1             ;habilita INT1
clr     PINORELE1

;MENU PRINCIPAL
MENUPRINCIPAL01:
call    CURSOROFF
setb    BITDATA
setb    MOSTRATEMPERATURA
call    TECLADO

cjne    a,#'1',MENUPRINCIPAL02
clr     BITDATA
setb    MOSTRATEMPERATURA

PERGUNTACANAL:
call    LIMPA
mov     dptr,#MSDIGITEOCH
call    ESCREVE
call    CURSORON
call    TECLADO
        cjne    a,#'*',PERGUNTACANAL01
        jmp     MENUPRINCIPAL01
PERGUNTACANAL01:
```

```
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIPERGUNTACANAL02
        jmp     MENUPRINCIPAL01
AQUIPERGUNTACANAL02:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        ;verifica se é maior que 0
        jz      PERGUNTACANAL
        ;verifica se é menor que 21
        mov     NUMTEMP1,a
        mov     b,#21
        div     ab
        jnz     PERGUNTACANAL
        mov     NUMCANAL,NUMTEMP1
jmp     MENUCANAIS

MENUPRINCIPAL02:
cjne    a,#'2',MENUPRINCIPAL03
jmp     MENUCANAIS

MENUPRINCIPAL03:
cjne    a,#'3',MENUPRINCIPAL04
clr     BITDATA                 ;AJUSTA HORÁRIO E CALENDÁRIO
clr     MOSTRATEMPERATURA
mov     AUXSEGUNDOS,SEGUNDOS
mov     AUXMINUTOS,MINUTOS
mov     AUXHORAS,HORAS
mov     AUXDIA,DIA
mov     AUXMES,MES
mov     AUXANO,ANO
mov     AUXDIASEMANA,DIASEMANA
call    AJUSTARRELOGIO
mov     SEGUNDOS,AUXSEGUNDOS
mov     MINUTOS,AUXMINUTOS
mov     HORAS,AUXHORAS
mov     DIA,AUXDIA
mov     MES,AUXMES
mov     ANO,AUXANO
mov     DIASEMANA,AUXDIASEMANA
jmp     MENUPRINCIPAL01

MENUPRINCIPAL04:
cjne    a,#'4',MENUPRINCIPAL05
;canal luz
mov     AUXNUMCANAL,NUMCANAL
mov     NUMCANAL,#24
jmp     MENULUZ

MENUPRINCIPAL05:
cjne    a,#'5',MENUPRINCIPAL06
;canal toca telefone
mov     AUXNUMCANAL,NUMCANAL
mov     NUMCANAL,#23
jmp     MENULUZ
```

```
MENUPRINCIPAL06:
cjne    a,#'6',MENUPRINCIPAL07
;canal aceita chamada
jmp     MENUPRINCIPAL01

MENUPRINCIPAL07:
cjne    a,#'7',MENUPRINCIPAL08
;funçao especial 1
CALL    SOM1
jmp     MENUPRINCIPAL01

MENUPRINCIPAL08:
cjne    a,#'8',MENUPRINCIPAL09
;funçao especial 2
CALL    SOM2
jmp     MENUPRINCIPAL01

MENUPRINCIPAL09:
cjne    a,#'9',MENUPRINCIPALASTERISCO
;funçao especial 3
CALL    SOM2
CALL    PAUSA
CALL    SOM2
jmp     MENUPRINCIPAL01

MENUPRINCIPALASTERISCO:
cjne    a,#'*',MENUPRINCIPAL00
;desativa funções especiais
jmp     MENUPRINCIPAL01

MENUPRINCIPAL00:
cjne    a,#'0',MENUPRINCIPALSUSTENIDO
;bateria e memoria
jmp     MENUPRINCIPAL01

MENUPRINCIPALSUSTENIDO:
cjne    a,#'#',MENUPRINCIPALF
clr     BITDATA
call    TROCASENHA
setb    BITDATA
jmp     MENUPRINCIPAL01

MENUPRINCIPALF:
cjne    a,#'F',MENUPRINCIPALR
;sem funçao
jmp     MENUPRINCIPAL01

MENUPRINCIPALR:
;cjne    a,#'R',MENUPRINCIPAL01
;sem funçao
jmp     MENUPRINCIPAL01


;MENU NÍVEL 2
MENUCANAIS:
;rola
call    CURSOROFF
clr     BITDATA

MENUCANAIS01:
;Mostra CH selecionado:
```

```
call    POSICIONADPTRNOME
call    LIMPA
call    ESCREVECH
call    ESCREVENOME
call    PULALINHA
mov     dptr,#MSSTATUS
call    ESCREVE
call    VERIFICASTATUS
;ver se esta ligado ou desligado
jc      STATUSDESL
mov     dptr,#MSLIGADO
jmp     STATUS2
STATUSDESL:
mov     dptr,#MSDESLIGADO
STATUS2:
call    ESCREVE
;L,D,P

MENUCANAISTECLADO:
        call    TECLADO
        cjne    a,#'1',MENUCANAIS02
call    PROGRAMACH
jmp     MENUCANAIS
MENUCANAIS02:
        cjne    a,#'2',MENUCANAIS03
call    NOMEIACH
jmp     MENUCANAIS
MENUCANAIS03:
        cjne    a,#'3',MENUCANAIS04
        inc     NUMCANAL
        mov     a,NUMCANAL
        cjne    a,#21,MENUCANAIS
        mov     NUMCANAL,#1
jmp     MENUCANAIS
MENUCANAIS04:
        cjne    a,#'4',MENUCANAIS05
call    VERLIGA
jmp     MENUCANAISTECLADO
MENUCANAIS05:
        cjne    a,#'5',MENUCANAIS06
call    VERDESLIGA
jmp     MENUCANAISTECLADO
MENUCANAIS06:
        cjne    a,#'6',MENUCANAIS07
        dec     NUMCANAL
        mov     a,NUMCANAL
        cjne    a,#0,MENUCANAIS
        mov     NUMCANAL,#20
        jmp     MENUCANAIS
MENUCANAIS07:
        cjne    a,#'7',MENUCANAIS08
;liga
call    APAGAPROGLIGA
jmp     MENUCANAIS
MENUCANAIS08:
        cjne    a,#'8',MENUCANAIS09
;desliga
call    APAGAPROGDESLIGA
jmp     MENUCANAIS
MENUCANAIS09:
        cjne    a,#'9',MENUCANAISASTERISCO
```

```
;prog
jmp     MENUCANAIS
MENUCANAISASTERISCO:
        cjne    a,#'*',MENUCANAIS
jmp     MENUPRINCIPAL01

;MENU NÍVEL 2 - LUZ
MENULUZ:
;mov    AUXNUMCANAL,NUMCANAL
;mov    NUMCANAL,#24
MENULUZ01:
call    CURSOROFF
clr     BITDATA
call    LIMPA
mov     dptr,#MSLUZDEFUNDO
mov     a,NUMCANAL
cjne    a,#23,FONEOUT01
mov     dptr,#MSLINHATELEFONICA
FONEOUT01:
call    ESCREVE
call    PULALINHA
mov     dptr,#MSSTATUS
call    ESCREVE
call    VERIFICASTATUS
;ver se esta ligado ou desligado
jc      STATUSLUZDESL
mov     dptr,#MSLIGADA
jmp     STATUSLUZ2
STATUSLUZDESL:
mov     dptr,#MSDESLIGADO
STATUSLUZ2:
call    ESCREVE
;L,D,P

MENULUZTECLADO:
        call    TECLADO
        cjne    a,#'1',MENULUZ02
call    PROGRAMACH
jmp     MENULUZ01
MENULUZ02:
        cjne    a,#'2',MENULUZ03
jmp     MENULUZ01
MENULUZ03:
        cjne    a,#'3',MENULUZ04
jmp     MENULUZ01
MENULUZ04:
        cjne    a,#'4',MENULUZ05
call    VERLIGA
jmp     MENULUZTECLADO
MENULUZ05:
        cjne    a,#'5',MENULUZ06
call    VERDESLIGA
jmp     MENULUZTECLADO
MENULUZ06:
        cjne    a,#'6',MENULUZ07
        jmp     MENULUZ01
MENULUZ07:
        cjne    a,#'7',MENULUZ08
;liga
call    APAGAPROGLIGA
jmp     MENULUZ01
```

```
MENULUZ08:
        cjne    a,#'8',MENULUZ09
;desliga
call    APAGAPROGDESLIGA
jmp     MENULUZ01
MENULUZ09:
        cjne    a,#'9',MENULUZASTERISCO
;prog
jmp     MENULUZ01
MENULUZASTERISCO:
        cjne    a,#'*',MENULUZ01
mov     NUMCANAL,AUXNUMCANAL
jmp     MENUPRINCIPAL01


jmp     $

clr     BITDATA
setb    MOSTRATEMPERATURA

call    PROGRAMACH
call    NOMEIACH

;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
INTZERO:
        mov     PSW.5,c
        push    acc
        mov     a,R0
        push    acc
        push    dph
        push    dpl
        mov     dptr,#DTMF
        movx    a,@dptr
        anl     a,#00001111b            ;pega a metade menos significativa
        cjne    a,#00001100b,TESTANUMEROS01     ;verifica se foi digitado #
mov     a,APONTADORSENHA
cjne    a,#(INICIOSENHA-1),TESTAPOSICAO01
mov     SEGUNDOSCHAMADA,#6
setb    PINORELE1
inc     APONTADORSENHA
jmp     SAIINTZERO
TESTAPOSICAO01:
mov     R0,APONTADORSENHA
mov     @R0,#'#'
;verifica senhas
call    VERIFICASENHA
jc      PULATELECOMANDO
        call    TELECOMANDO         ;se corretas
PULATELECOMANDO:
clr     PINORELE1               ;se erradas
mov     APONTADORSENHA,#(INICIOSENHA-1)
jmp     SAIINTZERO

TESTANUMEROS01:
JNB     PINORELE1,SAIINTZERO
mov     SEGUNDOSCHAMADA,#4
        orl     a,#00110000b   ;adiciona a metade mais signif. (ver tab. ASC)
        mov     R0,APONTADORSENHA
        mov     @R0,a
        inc     APONTADORSENHA
mov     a,APONTADORSENHA;VERIFICAR SE SENHA MAIOR QUE 6
```

```
cjne    a,#(INICIOSENHA+6),SAIINTZERO
jmp     TESTAPOSICAO01

SAIINTZERO:
        mov     c,PSW.5
        pop     dpl
        pop     dph
        pop     acc
        mov     R0,a
        pop     acc
        clr     IE0             ;barreira - limpa flag de INT0
        reti                    ;para não entrar de novo na rotina
;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;========================================================================
TELECOMANDO:
mov     SEGUNDOSCHAMADA,#30

        mov     a,NUMCANAL
        push    acc
;       mov     SALVABITDATA,BITDATA
;       clr     BITDATA
;       setb    BITTEMPORESTANTE
;       call    LIMPA
;       mov     dptr,#MSCHAMADAEMCURSO
;       call    ESCREVE

PERGUNTACANALFONE:
call    SOM2
call    PAUSA
call    SOM2
mov     SEGUNDOSCHAMADA,#180
call    TELEFONE
        cjne    a,#'*',PERGUNTACANALFONE01
        jmp     SAITELECOMANDO
PERGUNTACANALFONE01:
;       call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
mov     SEGUNDOSCHAMADA,#180
        call    TELEFONE
        cjne    a,#'*',AQUIPERGUNTACANALFONE02
        jmp     SAITELECOMANDO
AQUIPERGUNTACANALFONE02:
;       call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        ;verifica se é maior que 0
        jz      PERGUNTACANALFONE
        ;verifica se é menor que 21
        mov     NUMTEMP1,a
        mov     b,#21
        div     ab
        jnz     PERGUNTACANALFONE
        mov     NUMCANAL,NUMTEMP1
;verifica status
;manda sinal sonoro de status
MENUCANAISTELEFONE01:
call    VERIFICASTATUS
jnc     SOMSTATUS01
```

```
call    SOM1
jmp     SOMSTATUS02
SOMSTATUS01:
call    SOM2
SOMSTATUS02:
;fica esperando comando
MENUCANAISTELEFONE:
mov     SEGUNDOSCHAMADA,#240
        call    TELEFONE
        cjne    a,#'1',MENUCANAISFONE02
jmp     PERGUNTACANALFONE
MENUCANAISFONE02:
        cjne    a,#'3',MENUCANAISFONE04
        inc     NUMCANAL
        mov     a,NUMCANAL
        cjne    a,#21,MENUCANAISTELEFONE01
        mov     NUMCANAL,#1
jmp     MENUCANAISTELEFONE01
MENUCANAISFONE04:
        cjne    a,#'6',MENUCANAISFONE07
        dec     NUMCANAL
        mov     a,NUMCANAL
        cjne    a,#0,MENUCANAISTELEFONE01
        mov     NUMCANAL,#20
        jmp     MENUCANAISTELEFONE01
MENUCANAISFONE07:
        cjne    a,#'7',MENUCANAISFONE08
;liga
call    APAGAPROGLIGA
jmp     MENUCANAISTELEFONE01
MENUCANAISFONE08:
        cjne    a,#'8',MENUCANAISFONE09
;desliga
call    APAGAPROGDESLIGA
jmp     MENUCANAISTELEFONE01
MENUCANAISFONE09:
        cjne    a,#'*',MENUCANAISTELEFONE01



SAITELECOMANDO:
clr     PINORELE1
call    SOM1
call    SOM1
        pop     acc
        mov     NUMCANAL,a
;       clr     BITTEMPORESTANTE
;       mov     BITDATA,SALVABITDATA
ret
;=====================================================================
;=====================================================================
TELEFONE:
        mov     a,SEGUNDOSCHAMADA
        cjne    a,#0,AQUI37
        jmp     FORCASAIDA
AQUI37:
        jnb     DADOPRONTO,TELEFONE
AQUI38:
        mov     a,SEGUNDOSCHAMADA
        cjne    a,#0,AQUI39
        jmp     FORCASAIDA
```

```
AQUI39:
        jb      DADOPRONTO,AQUI38       ;espera novo número ser digitado (tb verifica
ociosidade na linha)
        mov     dptr,#DTMF
        movx    a,@dptr
        anl     a,#00001111b  ;pega a metade menos significativa
        orl     a,#00110000b  ;adiciona a metade mais signif. (ver tab. ASC)
        cjne    a,#00111011b,AQUI16 ;verifica se foi apertado * no telefone
FORCASAIDA:
        mov     a,#'*'          ;o * precisa ser corrigido
AQUI16:
        cjne    a,#00111010b,AQUI19 ;verifica se foi apertado 0 no telefone
        mov     a,#'0'          ;o 0 precisa ser corrigido de 1010 para 0000
AQUI19:
        cjne    a,#00111100b,AQUI20 ;verifica se foi apertado # no telefone
        mov     a,#'#'          ;o # precisa ser corrigido
AQUI20:
        ret
;=====================================================================
;=====================================================================
;TEMPORESTANTE:          ;escreve tempo restante
;;      push    acc             ;salva acc
;       call    PULALINHA
;       mov     dptr,#MSTEMPORESTANTE
;       call    ESCREVE
;       mov     a,#0c8h          ;vai para a C8
;       call    WR_INSTRUCAO
;       mov     a,SEGUNDOSCHAMADA
;       call    ESCREVE3DIGITOS
;;      pop     acc
;       ret
;=====================================================================
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;INTUM:

;       clr     IE1             ;barreira - limpa flag de INT0
;       reti                    ;para não entrar de novo na rotina
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
ROTINA:                 ;atualiza relógio
        mov     TL0,#LOW(65535-50000)
        mov     TH0,#HIGH(65535-50000)
        push    acc
mov     a,b
        push    acc
        push    dph
        push    dpl

        inc     R2
jnb     BITPISCADIASEMANA,NAOPISCADIASEMANA
mov     a,#082h                 ;vai para dia da semana
call    WR_INSTRUCAO
call    MOSTRADS
call    MOSTRADATA
mov     a,R2
mov     b,#20
div     ab
jnz     NAOPISCADIASEMANA
        mov     a,#082h                 ;vai para dia da semana
        call    WR_INSTRUCAO
        mov     a,#' '
```

```
        call    CARACTER
        call    CARACTER
        call    CARACTER
NAOPISCADIASEMANA:

        cjne    R2,#40,FIM
        mov     R2,#0

jnb     BITDATA,NAOMOSTRADATA
mov     AUXSEGUNDOS,SEGUNDOS
mov     AUXMINUTOS,MINUTOS
mov     AUXHORAS,HORAS
mov     AUXDIA,DIA
mov     AUXMES,MES
mov     AUXANO,ANO
mov     AUXDIASEMANA,DIASEMANA
mov     AUXREGTEMP,REGTEMPERATURA
call    TELAPADRAO
NAOMOSTRADATA:

;verifica se é necessário ligar/desl. algum canal:
mov     VERNUMCANAL,#1
VEREFICANAROTINA:
mov     a,VERNUMCANAL
dec     a
mov     b,#TAMANHOCANAL
mul     ab
mov     dph,b
mov     dpl,a
call    MOVEMEMORIAVER

clr     BITCONTROLE
call    VERIFICACANAL
jnb     BITCONTROLE,NAOLIGA
call    LIGACANAL
NAOLIGA:

mov     a,VERNUMCANAL
dec     a
mov     b,#TAMANHOCANAL
mul     ab
mov     dph,b
mov     dpl,a
inc     dptr
inc     dptr
inc     dptr
inc     dptr
inc     dptr
inc     dptr
inc     dptr
inc     dptr
call    MOVEMEMORIAVER

clr     BITCONTROLE
call    VERIFICACANAL
jnb     BITCONTROLE,NAODESLIGA
call    DESLIGACANAL
NAODESLIGA:

inc     VERNUMCANAL
mov     a,VERNUMCANAL
```

```
cjne    a,#25,VEREFICANAROTINA

call    ADIQUIRE
call    ATUALIZAHORARIO
FIM:
        pop     dpl
        pop     dph
        pop     acc
mov     b,a
        pop     acc
        reti
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
TIMER2:
clr     0C8h.7;TF2
CPL     P3.5
        reti
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;============================================================================
ADIQUIRE:
;pega do AD
;       push    dph
;       push    dpl
        mov     dptr,#AD
        movx    a,@dptr
mov     b,#5
div     ab
mov     MEDIA1,MEDIA2
mov     MEDIA2,MEDIA3
mov     MEDIA3,MEDIA4
mov     MEDIA4,MEDIA5
mov     MEDIA5,a
add     a,MEDIA4
add     a,MEDIA3
add     a,MEDIA2
add     a,MEDIA1
mov     b,#5
div     ab
mov     REGTEMPERATURA,a
;       pop     dpl
;       pop     dph
ret
;============================================================================
;============================================================================
TROCASENHA:
        mov     a,R0
        push    acc

        call    LIMPA
        mov     dptr,#MSNOVASENHA
        call    ESCREVE
        call    PULALINHA
        mov     dptr,#MSSETAS
        call    ESCREVE
        mov     a,#0c5h         ;vai para C5
        call    WR_INSTRUCAO
        call    CURSORON
        mov     R0,#INICIOSENHAPADRAO   ;R0 na posição inicial de SENHA
ARMAZENASENHA:
        call    TECLADO
;       cjne    a,#'*',TROCASENHA01         ;* significa cancelar
```

```
;       pop     acc
;       mov     R0,a
;       ret
;TROCASENHA01:                          ;não houve cancelamento
        call    CARACTER
        mov     @R0,a                   ;salva
        inc     R0
        cjne    R0,#(INICIOSENHAPADRAO+6),TROCASENHA02      ;verifica se a senha já
ultrapassou 8 digitos
        mov     @R0,#'#'                ;coloca caracter de término
        mov     a,#'#'                  ;cjne deve verificar a, e nao @R0
TROCASENHA02:                           ;nao se pode verificar @R0 ao invés do acc
        cjne    a,#'#',ARMAZENASENHA    ;pq @R0 incrementado pode conter qualquer coisa

        pop     acc
        mov     R0,a
        ret
;=========================================================================
;=========================================================================
VERIFICASENHA:
        mov     R0,#INICIOSENHAPADRAO
        mov     R1,#INICIOSENHA
        clr     c               ;0=correto, 1=erro

CONFERE:
        cjne    @R0,#'#',VERIFICASENHA01;INICIOSENHAPADRAO chegou no fim?
        cjne    @R1,#'#',INDICAERRO     ;INICIOSENHA chegou no fim?
        ret
VERIFICASENHA01:
        mov     a,@R0
        subb    a,@R1
        mov     @R1,' ' ;INICIOSENHA
        jnz     INDICAERRO
        inc     R0
        inc     R1
        jmp     CONFERE
INDICAERRO:  ;TEMPs com valores ou tamanhos diferentes
        setb    c
        ret
;=========================================================================
;=========================================================================
VERIFICACANAL:
mov     a,VERSEGUNDOS
anl     a,VERMINUTOS
anl     a,VERHORAS
anl     a,VERDIA
anl     a,VERMES
anl     a,VERANO
cjne    a,#255,AQUIEVER00
jmp     FIMVERIFICA
AQUIEVER00:

mov     a,VERSEGUNDOS
cjne    a,#255,AQUIEVER01
jmp     AQUIEVER02
AQUIEVER01:
xrl     a,SEGUNDOS
jnz     FIMVERIFICA
;call    LIGACANAL01
AQUIEVER02:
```

```
mov     a,VERMINUTOS
cjne    a,#255,AQUIEVER03
jmp     AQUIEVER04
AQUIEVER03:
xrl     a,MINUTOS
jnz     FIMVERIFICA
;call   LIGACANAL01
AQUIEVER04:

mov     a,VERHORAS
cjne    a,#255,AQUIEVER05
jmp     AQUIEVER06
AQUIEVER05:
xrl     a,HORAS
jnz     FIMVERIFICA
AQUIEVER06:

mov     a,VERDIA
cjne    a,#255,AQUIEVER07
jmp     AQUIEVER08
AQUIEVER07:
xrl     a,DIA
jnz     FIMVERIFICA
AQUIEVER08:

mov     a,VERMES
cjne    a,#255,AQUIEVER09
jmp     AQUIEVER10
AQUIEVER09:
xrl     a,MES
jnz     FIMVERIFICA
AQUIEVER10:

mov     a,VERANO
cjne    a,#255,AQUIEVER11
jmp     AQUIEVER12
AQUIEVER11:
xrl     a,ANO
jnz     FIMVERIFICA
AQUIEVER12:

;verifica dias da semana marcados:
mov     a,VERDIASEMANA
anl     a,#01111111b
jz      NAOVERIFICADIASEMANA
cjne    a,#255,VERIFICADIASEMANA
jmp     NAOVERIFICADIASEMANA
VERIFICADIASEMANA:

mov     a,DIASEMANA
cjne    a,#1,VERDS01
jnb     VERDIASEMANA.6,FIMVERIFICA
jmp     NAOVERIFICADIASEMANA
VERDS01:
cjne    a,#2,VERDS02
jnb     VERDIASEMANA.5,FIMVERIFICA
jmp     NAOVERIFICADIASEMANA
VERDS02:
cjne    a,#3,VERDS03
jnb     VERDIASEMANA.4,FIMVERIFICA
jmp     NAOVERIFICADIASEMANA
```

```
VERDS03:
cjne    a,#4,VERDS04
jnb     VERDIASEMANA.3,FIMVERIFICA
jmp     NAOVERIFICADIASEMANA
VERDS04:
cjne    a,#5,VERDS05
jnb     VERDIASEMANA.2,FIMVERIFICA
jmp     NAOVERIFICADIASEMANA
VERDS05:
cjne    a,#6,VERDS06
jnb     VERDIASEMANA.1,FIMVERIFICA
jmp     NAOVERIFICADIASEMANA
VERDS06:
;cjne   a,#7,VERDS07
jnb     VERDIASEMANA.0,FIMVERIFICA
;jmp    NAOVERIFICADIASEMANA
;VERDS07:
NAOVERIFICADIASEMANA:

setb    BITCONTROLE
ret
FIMVERIFICA:
;verifica temperatura
mov     a,VERREGTEMP
cjne    a,#255,VERIFICATEMP
jmp     FIMVERIFICATEMP
VERIFICATEMP:
xrl     a,REGTEMPERATURA
jnz     FIMVERIFICATEMP
setb    BITCONTROLE
FIMVERIFICATEMP:
ret
;=====================================================================
MOVEMEMORIAVER:
        movx    a,@dptr
        mov     VERSEGUNDOS,a
        inc     dptr
        movx    a,@dptr
        mov     VERMINUTOS,a
        inc     dptr
        movx    a,@dptr
        mov     VERHORAS,a
        inc     dptr
        movx    a,@dptr
        mov     VERDIA,a
        inc     dptr
        movx    a,@dptr
        mov     VERMES,a
        inc     dptr
        movx    a,@dptr
        mov     VERANO,a
        inc     dptr
        movx    a,@dptr
        mov     VERDIASEMANA,a
        inc     dptr
        movx    a,@dptr
        mov     VERREGTEMP,a
        ret
;=====================================================================
LIGACANAL:
        mov     a,VERNUMCANAL
```

```
;Verifica 1º registrador:
        cjne    a,#1,AQUILIGA11
        clr     REGISTRADOR1.0
AQUILIGA11:
        cjne    a,#2,AQUILIGA12
        clr     REGISTRADOR1.1
AQUILIGA12:
        cjne    a,#3,AQUILIGA13
        clr     REGISTRADOR1.2
AQUILIGA13:
        cjne    a,#4,AQUILIGA14
        clr     REGISTRADOR1.3
AQUILIGA14:
        cjne    a,#5,AQUILIGA15
        clr     REGISTRADOR1.4
AQUILIGA15:
        cjne    a,#6,AQUILIGA16
        clr     REGISTRADOR1.5
AQUILIGA16:
        cjne    a,#7,AQUILIGA17
        clr     REGISTRADOR1.6
AQUILIGA17:
        cjne    a,#8,AQUILIGA18
        clr     REGISTRADOR1.7
AQUILIGA18:
;Verifica 2º registrador:
        cjne    a,#9,AQUILIGA21
        clr     REGISTRADOR2.0
AQUILIGA21:
        cjne    a,#10,AQUILIGA22
        clr     REGISTRADOR2.1
AQUILIGA22:
        cjne    a,#11,AQUILIGA23
        clr     REGISTRADOR2.2
AQUILIGA23:
        cjne    a,#12,AQUILIGA24
        clr     REGISTRADOR2.3
AQUILIGA24:
        cjne    a,#13,AQUILIGA25
        clr     REGISTRADOR2.4
AQUILIGA25:
        cjne    a,#14,AQUILIGA26
        clr     REGISTRADOR2.5
AQUILIGA26:
        cjne    a,#15,AQUILIGA27
        clr     REGISTRADOR2.6
AQUILIGA27:
        cjne    a,#16,AQUILIGA28
        clr     REGISTRADOR2.7
AQUILIGA28:
;Verifica 3º registrador:
        cjne    a,#17,AQUILIGA31
        clr     REGISTRADOR3.0
AQUILIGA31:
        cjne    a,#18,AQUILIGA32
        clr     REGISTRADOR3.1
AQUILIGA32:
        cjne    a,#19,AQUILIGA33
        clr     REGISTRADOR3.2
AQUILIGA33:
        cjne    a,#20,AQUILIGA34
```

```
        clr     REGISTRADOR3.3
AQUILIGA34:
        cjne    a,#21,AQUILIGA35
        clr     REGISTRADOR3.4
        call    ATUALIZAREGISTRADOR3EX
        ret
AQUILIGA35:
        cjne    a,#22,AQUILIGA36
        clr     REGISTRADOR3.5
        call    ATUALIZAREGISTRADOR3EX
        ret
AQUILIGA36:
        cjne    a,#23,AQUILIGA37
        clr     REGISTRADOR3.6
        call    ATUALIZAREGISTRADOR3EX
        ret
AQUILIGA37:
        cjne    a,#24,AQUILIGA38
        clr     REGISTRADOR3.7
        call    ATUALIZAREGISTRADOR3EX
        ret
AQUILIGA38:
        setb    BITLIGADESLIGA
        call    ENVIASERIAL
        call    ATUALIZAREGISTRADOR1EX
        call    ATUALIZAREGISTRADOR2EX
        call    ATUALIZAREGISTRADOR3EX
        ret
;=====================================================================
;=====================================================================
DESLIGACANAL:
        mov     a,VERNUMCANAL
;Verifica 1º registrador:
        cjne    a,#1,AQUIDESLIGA11
        setb    REGISTRADOR1.0
AQUIDESLIGA11:
        cjne    a,#2,AQUIDESLIGA12
        setb    REGISTRADOR1.1
AQUIDESLIGA12:
        cjne    a,#3,AQUIDESLIGA13
        setb    REGISTRADOR1.2
AQUIDESLIGA13:
        cjne    a,#4,AQUIDESLIGA14
        setb    REGISTRADOR1.3
AQUIDESLIGA14:
        cjne    a,#5,AQUIDESLIGA15
        setb    REGISTRADOR1.4
AQUIDESLIGA15:
        cjne    a,#6,AQUIDESLIGA16
        setb    REGISTRADOR1.5
AQUIDESLIGA16:
        cjne    a,#7,AQUIDESLIGA17
        setb    REGISTRADOR1.6
AQUIDESLIGA17:
        cjne    a,#8,AQUIDESLIGA18
        setb    REGISTRADOR1.7
AQUIDESLIGA18:
;Verifica 2º registrador:
        cjne    a,#9,AQUIDESLIGA21
        setb    REGISTRADOR2.0
AQUIDESLIGA21:
```

```
        cjne    a,#10,AQUIDESLIGA22
        setb    REGISTRADOR2.1
AQUIDESLIGA22:
        cjne    a,#11,AQUIDESLIGA23
        setb    REGISTRADOR2.2
AQUIDESLIGA23:
        cjne    a,#12,AQUIDESLIGA24
        setb    REGISTRADOR2.3
AQUIDESLIGA24:
        cjne    a,#13,AQUIDESLIGA25
        setb    REGISTRADOR2.4
AQUIDESLIGA25:
        cjne    a,#14,AQUIDESLIGA26
        setb    REGISTRADOR2.5
AQUIDESLIGA26:
        cjne    a,#15,AQUIDESLIGA27
        setb    REGISTRADOR2.6
AQUIDESLIGA27:
        cjne    a,#16,AQUIDESLIGA28
        setb    REGISTRADOR2.7
AQUIDESLIGA28:
;Verifica 3º registrador:
        cjne    a,#17,AQUIDESLIGA31
        setb    REGISTRADOR3.0
AQUIDESLIGA31:
        cjne    a,#18,AQUIDESLIGA32
        setb    REGISTRADOR3.1
AQUIDESLIGA32:
        cjne    a,#19,AQUIDESLIGA33
        setb    REGISTRADOR3.2
AQUIDESLIGA33:
        cjne    a,#20,AQUIDESLIGA34
        setb    REGISTRADOR3.3
AQUIDESLIGA34:
        cjne    a,#21,AQUIDESLIGA35
        setb    REGISTRADOR3.4
        call    ATUALIZAREGISTRADOR3EX
        ret
AQUIDESLIGA35:
        cjne    a,#22,AQUIDESLIGA36
        setb    REGISTRADOR3.5
        call    ATUALIZAREGISTRADOR3EX
        ret
AQUIDESLIGA36:
        cjne    a,#23,AQUIDESLIGA37
        setb    REGISTRADOR3.6
        call    ATUALIZAREGISTRADOR3EX
        ret
AQUIDESLIGA37:
        cjne    a,#24,AQUIDESLIGA38
        setb    REGISTRADOR3.7
        call    ATUALIZAREGISTRADOR3EX
        ret
AQUIDESLIGA38:
        clr     BITLIGADESLIGA
        call    ENVIASERIAL
        call    ATUALIZAREGISTRADOR1EX
        call    ATUALIZAREGISTRADOR2EX
        call    ATUALIZAREGISTRADOR3EX
        ret
;=====================================================================
```

```
;=========================================================================
ATUALIZAREGISTRADOR1EX:
        mov     dptr,#REGISTRADOR1EX
        mov     a,REGISTRADOR1
        movx    @dptr,a
        ret
;=========================================================================
;=========================================================================
ATUALIZAREGISTRADOR2EX:
        mov     dptr,#REGISTRADOR2EX
        mov     a,REGISTRADOR2
        movx    @dptr,a
        ret
;=========================================================================
;=========================================================================
ATUALIZAREGISTRADOR3EX:
        mov     dptr,#REGISTRADOR3EX
        mov     a,REGISTRADOR3
        movx    @dptr,a
        ret
;=========================================================================
;=========================================================================
ENVIASERIAL:
mov     a,VERNUMCANAL
rl      a
rl      a
rl      a
jnb     BITLIGADESLIGA,ENVIASERIAL01
orl     a,#00000111b
jmp     ENVIASERIAL02
ENVIASERIAL01:
anl     a,#11111000b
ENVIASERIAL02:

mov     SBUF,a
jnb     TI,$
clr     TI

        ret
;=========================================================================
;=========================================================================
VERIFICASTATUS:
mov     a,NUMCANAL
dec     a
        mov     b,#8
        div     ab
        jnz     VERIFICASTATUS2

mov     a,REGISTRADOR1
mov     b,NUMCANAL
jmp     FIMVERIFICASTATUS

VERIFICASTATUS2:
cjne    a,#1,VERIFICASTATUS3
mov     a,REGISTRADOR2
inc     b
jmp     FIMVERIFICASTATUS

VERIFICASTATUS3:
mov     a,REGISTRADOR3
inc     b
```

```
FIMVERIFICASTATUS:
RODAREG:
rrc     a
djnz    b,RODAREG
ret
;=========================================================================
;=========================================================================
APAGAPROGLIGA:
mov     VERNUMCANAL,NUMCANAL
call    FORMATACANAL
call    LIGACANAL
ret
;=========================================================================
;=========================================================================
APAGAPROGDESLIGA:
mov     VERNUMCANAL,NUMCANAL
call    FORMATACANAL
call    DESLIGACANAL
ret
;=========================================================================
;=========================================================================
FORMATACANAL:
call    POSICIONADPTRCH
mov     NUMTEMP1,#6
FORMATACANAL01:
mov     a,#255
movx    @dptr,a
inc     dptr
djnz    NUMTEMP1,FORMATACANAL01
mov     a,#0
movx    @dptr,a
inc     dptr
mov     a,#255
movx    @dptr,a
inc     dptr    ;posiciona nos parâmetros de desligar
mov     NUMTEMP1,#6
FORMATACANAL02:
mov     a,#255
movx    @dptr,a
inc     dptr
djnz    NUMTEMP1,FORMATACANAL02
mov     a,#0
movx    @dptr,a
inc     dptr
mov     a,#255
movx    @dptr,a
ret
;=========================================================================
;=========================================================================
PROGRAMACH:

call    POSICIONADPTRCH
call    MOVEMEMORIACHS

mov     AUXSEGUNDOS,CHLSEGUNDOS
mov     AUXMINUTOS,CHLMINUTOS
mov     AUXHORAS,CHLHORAS
mov     AUXDIA,CHLDIA
mov     AUXMES,CHLMES
mov     AUXANO,CHLANO
```

```
mov     AUXDIASEMANA,CHLDIASEMANA
mov     AUXREGTEMP,CHLREGTEMP

call    LIMPA
mov     dptr,#MSPROGRAMEQUANDO
call    ESCREVE
call    PULALINHA
mov     dptr,#MSLIGAR
call    ESCREVE
        ;verifica se é menor que 21
        mov     a,NUMCANAL
        mov     b,#21
        div     ab
        jnz     CANALINTERNO
mov     dptr,#MSOCANAL
call    ESCREVE
mov     a,NUMCANAL
call    ESCREVE2DIGITOS
CANALINTERNO:
call    TECLADO

call    PROGRELOGIO

mov     CHLSEGUNDOS,AUXSEGUNDOS
mov     CHLMINUTOS,AUXMINUTOS
mov     CHLHORAS,AUXHORAS
mov     CHLDIA,AUXDIA
mov     CHLMES,AUXMES
mov     CHLANO,AUXANO
mov     CHLDIASEMANA,AUXDIASEMANA
mov     CHLREGTEMP,AUXREGTEMP

;programa quando desligar:
mov     AUXSEGUNDOS,CHDSEGUNDOS
mov     AUXMINUTOS,CHDMINUTOS
mov     AUXHORAS,CHDHORAS
mov     AUXDIA,CHDDIA
mov     AUXMES,CHDMES
mov     AUXANO,CHDANO
mov     AUXDIASEMANA,CHDDIASEMANA
mov     AUXREGTEMP,CHDREGTEMP

call    LIMPA
mov     dptr,#MSPROGRAMEQUANDO
call    ESCREVE
call    PULALINHA
mov     dptr,#MSDESL
call    ESCREVE
        ;verifica se é menor que 21
        mov     a,NUMCANAL
        mov     b,#21
        div     ab
        jnz     CANALINTERNO2
mov     dptr,#MSOCANAL
call    ESCREVE
mov     a,NUMCANAL
call    ESCREVE2DIGITOS
CANALINTERNO2:
call    TECLADO

call    PROGRELOGIO
```

```
mov     CHDSEGUNDOS,AUXSEGUNDOS
mov     CHDMINUTOS,AUXMINUTOS
mov     CHDHORAS,AUXHORAS
mov     CHDDIA,AUXDIA
mov     CHDMES,AUXMES
mov     CHDANO,AUXANO
mov     CHDDIASEMANA,AUXDIASEMANA
mov     CHDREGTEMP,AUXREGTEMP

call    POSICIONADPTRCH
call    MOVECHSMEMORIA

        ret
;=====================================================================
;=====================================================================
VERLIGA:

call    POSICIONADPTRCH
call    MOVEMEMORIACHS

mov     AUXSEGUNDOS,CHLSEGUNDOS
mov     AUXMINUTOS,CHLMINUTOS
mov     AUXHORAS,CHLHORAS
mov     AUXDIA,CHLDIA
mov     AUXMES,CHLMES
mov     AUXANO,CHLANO
mov     AUXDIASEMANA,CHLDIASEMANA
mov     AUXREGTEMP,CHLREGTEMP

call    TELAPADRAOPROG
        ret
;=====================================================================
;=====================================================================
VERDESLIGA:

call    POSICIONADPTRCH
call    MOVEMEMORIACHS

mov     AUXSEGUNDOS,CHDSEGUNDOS
mov     AUXMINUTOS,CHDMINUTOS
mov     AUXHORAS,CHDHORAS
mov     AUXDIA,CHDDIA
mov     AUXMES,CHDMES
mov     AUXANO,CHDANO
mov     AUXDIASEMANA,CHDDIASEMANA
mov     AUXREGTEMP,CHDREGTEMP

call    TELAPADRAOPROG
        ret
;=====================================================================
;=====================================================================
POSICIONADPTRCH:
mov     a,NUMCANAL
dec     a
mov     b,#TAMANHOCANAL
mul     ab
mov     dph,b
mov     dpl,a
ret
;=====================================================================
```

```
;=====================================================================
NOMEIACH:
call    POSICIONADPTRNOME

mov     DPTRH,dph      ;salva dptr
mov     DPTRL,dpl

call    LIMPA
call    ESCREVECH

call    ESCREVENOME

call    PULALINHA
mov     dptr,#MSREN
call    ESCREVE

call    CURSORON
mov     LOCALMENS,#INICIOMENS
;limpa
mov     R0,#INICIOMENS
mov     NUMTEMP2,#10
NOMEIACH07:
        mov     @R0,#' '
        inc     R0
djnz    NUMTEMP2,NOMEIACH07

call    TECLADOABC
jnb     SALVAMENS,NAOSALVAMENS
;Salva na MEMORIA EXTERNA
mov     dph,DPTRH
mov     dpl,DPTRL
mov     R0,#INICIOMENS
mov     NUMTEMP2,#10
NOMEIACH06:
        mov     a,@R0
        movx    @dptr,a
        inc     dptr
        inc     R0
djnz    NUMTEMP2,NOMEIACH06
NAOSALVAMENS:
ret
;=====================================================================
;=====================================================================
POSICIONADPTRNOME:
mov     a,NUMCANAL
dec     a
mov     b,#TAMANHONOME
mul     ab
mov     dph,b
mov     dpl,a

;soma 256 ao dptr:
mov     NUMTEMP2,#0
NOMEIACH01:
inc     dptr
djnz    NUMTEMP2,NOMEIACH01
;soma 128 ao dptr:
mov     NUMTEMP2,#128
NOMEIACH02:
inc     dptr
djnz    NUMTEMP2,NOMEIACH02
```

```
ret
;=========================================================================
;=========================================================================
ESCREVENOME:
mov     NUMTEMP2,#9 ;escreve conteudo da memória externa na tela
NOMEIACH03:
        movx    a,@dptr
        call    CARACTER
        inc     dptr
djnz    NUMTEMP2,NOMEIACH03
        movx    a,@dptr
        call    CARACTER
        inc     dptr
ret
;=========================================================================
;=========================================================================
ESCREVECH:
mov     a,#'C'
call    CARACTER
mov     a,#'H'
call    CARACTER
mov     a,NUMCANAL
call    ESCREVE2DIGITOS
mov     a,#':'
call    CARACTER
mov     a,#' '
call    CARACTER
ret
;=========================================================================
;=========================================================================
TECLADOABC:
call    TECLADO

RETORNATECLADOABC:
cjne    a,#'1',TESTATECLAASTERISCO
mov     R0,LOCALMENS
mov     @R0,a
inc     LOCALMENS
        call    CARACTER
        jmp     TECLADOABC
TESTATECLAASTERISCO:
cjne    a,#'*',TESTATECLASUSTENIDO
mov     a,LOCALMENS
cjne    a,#INICIOMENS,TESTATECLASUSTENIDO2
clr     SALVAMENS   ;não salva
ret
TESTATECLASUSTENIDO2:
call    VOLTACURSOR
        mov     a,#' '
        call    CARACTER
call    VOLTACURSOR

dec     LOCALMENS
mov     R0,LOCALMENS
mov     @R0,#' '
        jmp     TECLADOABC
TESTATECLASUSTENIDO:
cjne    a,#'#',TESTATECLAREDIAL
jmp     TECLADOABC
TESTATECLAREDIAL:
cjne    a,#'R',TESTATECLA2
```

```
setb    SALVAMENS   ;salva
RET
TESTATECLA2:
cjne    a,#'2',TESTATECLA3
TESTATECLA2A:
cjne    a,#'2',TESTATECLA3B
        mov     a,#'A'
call    AUXTECLADOABC1
cjne    a,#'2',TESTATECLA3B
        mov     a,#'B'
call    AUXTECLADOABC1
cjne    a,#'2',TESTATECLA3B
        mov     a,#'C'
call    AUXTECLADOABC1
cjne    a,#'2',TESTATECLA3B
        mov     a,#'2'
call    AUXTECLADOABC1
jmp     TESTATECLA2A
TESTATECLA3B:
call    AUXTECLADOABC2
TESTATECLA3:
cjne    a,#'3',TESTATECLA4
TESTATECLA3A:
cjne    a,#'3',TESTATECLA4B
        mov     a,#'D'
call    AUXTECLADOABC1
cjne    a,#'3',TESTATECLA4B
        mov     a,#'E'
call    AUXTECLADOABC1
cjne    a,#'3',TESTATECLA4B
        mov     a,#'F'
call    AUXTECLADOABC1
cjne    a,#'3',TESTATECLA4B
        mov     a,#'3'
call    AUXTECLADOABC1
jmp     TESTATECLA3A
TESTATECLA4B:
call    AUXTECLADOABC2
TESTATECLA4:
cjne    a,#'4',TESTATECLA5
TESTATECLA4A:
cjne    a,#'4',TESTATECLA5B
        mov     a,#'G'
call    AUXTECLADOABC1
cjne    a,#'4',TESTATECLA5B
        mov     a,#'H'
call    AUXTECLADOABC1
cjne    a,#'4',TESTATECLA5B
        mov     a,#'I'
call    AUXTECLADOABC1
cjne    a,#'4',TESTATECLA5B
        mov     a,#'4'
call    AUXTECLADOABC1
jmp     TESTATECLA4A
TESTATECLA5B:
call    AUXTECLADOABC2
TESTATECLA5:
cjne    a,#'5',TESTATECLA6
TESTATECLA5A:
cjne    a,#'5',TESTATECLA6B
        mov     a,#'J'
```

```
call    AUXTECLADOABC1
cjne    a,#'5',TESTATECLA6B
        mov     a,#'K'
call    AUXTECLADOABC1
cjne    a,#'5',TESTATECLA6B
        mov     a,#'L'
call    AUXTECLADOABC1
cjne    a,#'5',TESTATECLA6B
        mov     a,#'5'
call    AUXTECLADOABC1
jmp     TESTATECLA5A
TESTATECLA6B:
call    AUXTECLADOABC2
TESTATECLA6:
cjne    a,#'6',TESTATECLA7
TESTATECLA6A:
cjne    a,#'6',TESTATECLA7B
        mov     a,#'M'
call    AUXTECLADOABC1
cjne    a,#'6',TESTATECLA7B
        mov     a,#'N'
call    AUXTECLADOABC1
cjne    a,#'6',TESTATECLA7B
        mov     a,#'O'
call    AUXTECLADOABC1
cjne    a,#'6',TESTATECLA7B
        mov     a,#'6'
call    AUXTECLADOABC1
jmp     TESTATECLA6A
TESTATECLA7B:
call    AUXTECLADOABC2
TESTATECLA7:
cjne    a,#'7',TESTATECLA8
TESTATECLA7A:
cjne    a,#'7',TESTATECLA8B
        mov     a,#'P'
call    AUXTECLADOABC1
cjne    a,#'7',TESTATECLA8B
        mov     a,#'Q'
call    AUXTECLADOABC1
cjne    a,#'7',TESTATECLA8B
        mov     a,#'R'
call    AUXTECLADOABC1
cjne    a,#'7',TESTATECLA8B
        mov     a,#'S'
call    AUXTECLADOABC1
cjne    a,#'7',TESTATECLA8B
        mov     a,#'7'
call    AUXTECLADOABC1
jmp     TESTATECLA7A
TESTATECLA8B:
call    AUXTECLADOABC2
TESTATECLA8:
cjne    a,#'8',TESTATECLA9
TESTATECLA8A:
cjne    a,#'8',TESTATECLA9B
        mov     a,#'T'
call    AUXTECLADOABC1
cjne    a,#'8',TESTATECLA9B
        mov     a,#'U'
call    AUXTECLADOABC1
```

```
cjne    a,#'8',TESTATECLA9B
        mov     a,#'V'
call    AUXTECLADOABC1
cjne    a,#'8',TESTATECLA9B
        mov     a,#'8'
call    AUXTECLADOABC1
jmp     TESTATECLA8A
TESTATECLA9B:
call    AUXTECLADOABC2
TESTATECLA9:
cjne    a,#'9',TESTATECLA0
TESTATECLA9A:
cjne    a,#'9',TESTATECLA0B
        mov     a,#'W'
call    AUXTECLADOABC1
cjne    a,#'9',TESTATECLA0B
        mov     a,#'X'
call    AUXTECLADOABC1
cjne    a,#'9',TESTATECLA0B
        mov     a,#'Y'
call    AUXTECLADOABC1
cjne    a,#'9',TESTATECLA0B
        mov     a,#'Z'
call    AUXTECLADOABC1
cjne    a,#'9',TESTATECLA0B
        mov     a,#'9'
call    AUXTECLADOABC1
jmp     TESTATECLA9A
TESTATECLA0B:
call    AUXTECLADOABC2
TESTATECLA0:
cjne    a,#'0',TESTATECLAF
TESTATECLA0A:
cjne    a,#'0',TESTATECLAFB
        mov     a,#' '
call    AUXTECLADOABC1
cjne    a,#'0',TESTATECLAFB
        mov     a,#'.'
call    AUXTECLADOABC1
cjne    a,#'0',TESTATECLAFB
        mov     a,#'-'
call    AUXTECLADOABC1
cjne    a,#'0',TESTATECLAFB
        mov     a,#':'
mov     CARACTERANTERIOR,a
call    AUXTECLADOABC1
cjne    a,#'0',TESTATECLAFB
        mov     a,#'#'
call    AUXTECLADOABC1
cjne    a,#'0',TESTATECLAFB
        mov     a,#'*'
call    AUXTECLADOABC1
cjne    a,#'0',TESTATECLAFB
        mov     a,#'0'
call    AUXTECLADOABC1
jmp     TESTATECLA0A
TESTATECLAFB:
call    AUXTECLADOABC2
TESTATECLAF:
cjne    a,#'F',RETORNATECLADOABC2
clr     SALVAMENS    ;não salva
```

```
ret
RETORNATECLADOABC2:
jmp     RETORNATECLADOABC
;=========================================================================
;=========================================================================
AUXTECLADOABC1:
        mov     CARACTERANTERIOR,a
        call    CARACTER
        call    VOLTACURSOR
        call    TECLADO
        ret
;=========================================================================
;=========================================================================
AUXTECLADOABC2:
        mov     R0,LOCALMENS
        mov     @R0,CARACTERANTERIOR
        inc     LOCALMENS
        call    AVANCACURSOR
        ret
;=========================================================================
;=========================================================================
MOVEMEMORIACHS:
        movx    a,@dptr
        mov     CHLSEGUNDOS,a
        inc     dptr
        movx    a,@dptr
        mov     CHLMINUTOS,a
        inc     dptr
        movx    a,@dptr
        mov     CHLHORAS,a
        inc     dptr
        movx    a,@dptr
        mov     CHLDIA,a
        inc     dptr
        movx    a,@dptr
        mov     CHLMES,a
        inc     dptr
        movx    a,@dptr
        mov     CHLANO,a
        inc     dptr
        movx    a,@dptr
        mov     CHLDIASEMANA,a
        inc     dptr
        movx    a,@dptr
        mov     CHLREGTEMP,a

        inc     dptr
        movx    a,@dptr
        mov     CHDSEGUNDOS,a
        inc     dptr
        movx    a,@dptr
        mov     CHDMINUTOS,a
        inc     dptr
        movx    a,@dptr
        mov     CHDHORAS,a
        inc     dptr
        movx    a,@dptr
        mov     CHDDIA,a
        inc     dptr
        movx    a,@dptr
        mov     CHDMES,a
```

```
        inc     dptr
        movx    a,@dptr
        mov     CHDANO,a
        inc     dptr
        movx    a,@dptr
        mov     CHDDIASEMANA,a
        inc     dptr
        movx    a,@dptr
        mov     CHDREGTEMP,a
        ret
;=====================================================================
;=====================================================================
MOVECHSMEMORIA:
        mov     a,CHLSEGUNDOS
        movx    @dptr,a
        inc     dptr
        movx    a,@dptr
        mov     a,CHLMINUTOS
        movx    @dptr,a
        inc     dptr
        mov     a,CHLHORAS
        movx    @dptr,a
        inc     dptr
        mov     a,CHLDIA
        movx    @dptr,a
        inc     dptr
        mov     a,CHLMES
        movx    @dptr,a
        inc     dptr
        mov     a,CHLANO
        movx    @dptr,a
        inc     dptr
        mov     a,CHLDIASEMANA
        movx    @dptr,a
        inc     dptr
        mov     a,CHLREGTEMP

        movx    @dptr,a
        inc     dptr
        mov     a,CHDSEGUNDOS
        movx    @dptr,a
        inc     dptr
        mov     a,CHDMINUTOS
        movx    @dptr,a
        inc     dptr
        mov     a,CHDHORAS
        movx    @dptr,a
        inc     dptr
        mov     a,CHDDIA
        movx    @dptr,a
        inc     dptr
        mov     a,CHDMES
        movx    @dptr,a
        inc     dptr
        mov     a,CHDANO
        movx    @dptr,a
        inc     dptr
        mov     a,CHDDIASEMANA
        movx    @dptr,a
        inc     dptr
        mov     a,CHDREGTEMP
```

```
        movx    @dptr,a
        ret
;=======================================================================
;=======================================================================
MOVEMEMORIAAUX:;não usada
movx    a,@dptr
mov     AUXSEGUNDOS,a
inc     dptr
movx    a,@dptr
mov     AUXMINUTOS,a
inc     dptr
movx    a,@dptr
mov     AUXHORAS,a
inc     dptr
movx    a,@dptr
mov     AUXDIA,a
inc     dptr
movx    a,@dptr
mov     AUXMES,a
inc     dptr
movx    a,@dptr
mov     AUXANO,a
inc     dptr
movx    a,@dptr
mov     AUXDIASEMANA,a
inc     dptr
movx    a,@dptr
mov     AUXREGTEMP,a
        ret
;=======================================================================
;=======================================================================
MOVEAUXMEMORIA:;não usada
mov     a,AUXSEGUNDOS
movx    @dptr,a
inc     dptr
mov     a,AUXMINUTOS
movx    @dptr,a
inc     dptr
mov     a,AUXHORAS
movx    @dptr,a
inc     dptr
mov     a,AUXDIA
movx    @dptr,a
inc     dptr
mov     a,AUXMES
movx    @dptr,a
inc     dptr
mov     a,AUXANO
movx    @dptr,a
inc     dptr
mov     a,AUXDIASEMANA
movx    @dptr,a
inc     dptr
mov     a,AUXREGTEMP
movx    @dptr,a
        ret
;=======================================================================
;=======================================================================
ATUALIZAHORARIO:

        mov     a,SEGUNDOSCHAMADA
```

```
        cjne    a,#0,SEGUNDOSCHAMADA01
        clr     PINORELE1
mov     APONTADORSENHA,#(INICIOSENHA-1)
        jmp     SEGUNDOSCHAMADA02
SEGUNDOSCHAMADA01:
        dec     SEGUNDOSCHAMADA
;       jnb     BITTEMPORESTANTE,SEGUNDOSCHAMADA02
;       call    TEMPORESTANTE
SEGUNDOSCHAMADA02:


CPL     P3.4
        inc     SEGUNDOS
        mov     a,SEGUNDOS
        cjne    a,#60,FIM2
        mov     SEGUNDOS,#0
        inc     MINUTOS
        mov     a,MINUTOS
        cjne    a,#60,FIM2
        mov     MINUTOS,#0
        inc     HORAS
        mov     a,HORAS
        cjne    a,#24,FIM2
        mov     HORAS,#0
        call    ATUALIZACALENDARIO
FIM2:
        ret
;=====================================================================
;=====================================================================
ATUALIZACALENDARIO:
        inc     DIASEMANA
        mov     a,DIASEMANA
        cjne    a,#8,PULADIA
        mov     DIASEMANA,#1
PULADIA:

        inc     DIA
mov     a,MES
cjne    a,#1,TESTAMES02
jmp     MES31DIAS
TESTAMES02:
cjne    a,#2,TESTAMES03
jmp     MES28DIAS
TESTAMES03:
cjne    a,#3,TESTAMES04
jmp     MES31DIAS
TESTAMES04:
cjne    a,#4,TESTAMES05
jmp     MES30DIAS
TESTAMES05:
cjne    a,#5,TESTAMES06
jmp     MES31DIAS
TESTAMES06:
cjne    a,#6,TESTAMES07
jmp     MES30DIAS
TESTAMES07:
cjne    a,#7,TESTAMES08
jmp     MES31DIAS
TESTAMES08:
cjne    a,#8,TESTAMES09
jmp     MES31DIAS
```

```
TESTAMES09:
cjne    a,#9,TESTAMES10
jmp     MES30DIAS
TESTAMES10:
cjne    a,#10,TESTAMES11
jmp     MES31DIAS
TESTAMES11:
cjne    a,#11,MES31DIAS
jmp     MES30DIAS

MES28DIAS:
mov     a,ANO
mov     b,#4
div     ab
mov     a,b
jz      ANOBISEXTO
mov     a,DIA
cjne    a,#28,FIM3
jmp     DIAPRIMEIRO
ANOBISEXTO:
        mov     a,DIA
        cjne    a,#29,FIM3
jmp     DIAPRIMEIRO

MES30DIAS:
        mov     a,DIA
        cjne    a,#31,FIM3
jmp     DIAPRIMEIRO

MES31DIAS:
        mov     a,DIA
        cjne    a,#32,FIM3
;jmp    DIAPRIMEIRO
DIAPRIMEIRO:
        mov     DIA,#1

        inc     MES
        mov     a,MES
        cjne    a,#13,FIM3
        mov     MES,#1

        inc     ANO
        mov     a,ANO
        cjne    a,#100,FIM3
        mov     ANO,#0
FIM3:
        ret
;=====================================================================
;=====================================================================
TELAPADRAO:
        call    LIMPA
        mov     a,#' '
        call    CARACTER
        call    CARACTER
call    MOSTRADS
        call    MOSTRADATA
        call    PULALINHA
        mov     a,#' '
        call    CARACTER
        call    MOSTRARELOGIO
jnb     MOSTRATEMPERATURA,NAOMOSTRATEMP
```

```
        mov     a,#' '
        call    CARACTER
        call    CARACTER
mov     a,AUXREGTEMP
;call   VERIFICA255
call    ESCREVE2DIGITOS
        mov     a,#11011111b  ;°
        call    CARACTER
        mov     a,#'C'
        call    CARACTER
NAOMOSTRATEMP:
ret
;=====================================================================
;=====================================================================
TELAPADRAOPROG:
push    dph
push    dpl
        call    LIMPA
        call    MOSTRADATA
        mov     a,#' '
        call    CARACTER
        mov     dptr,#MSD23456S
        call    ESCREVE
        mov     a,#089h                 ;vai para dias da semana
        call    WR_INSTRUCAO
mov     a,AUXDIASEMANA
rlc     a
mov     NUMTEMP1,#0
xch     a,NUMTEMP1
GIRADSDIR:
xch     a,NUMTEMP1
rlc     a
jc      MARCAD123456S
call    CURSORPULADIR
;call   CURSOROFF
jmp     VERIFICAGIRADSDIR
MARCAD123456S:
call    MARCADSX
;jmp    VERIFICAGIRADSDIR
VERIFICAGIRADSDIR:

inc     NUMTEMP1
xch     a,NUMTEMP1
cjne    a,#9,GIRADSDIR

        call    PULALINHA
        mov     a,#' '
        call    CARACTER
        call    MOSTRARELOGIO
        mov     a,#' '
        call    CARACTER
        call    CARACTER
mov     a,AUXREGTEMP
call    VERIFICA255
;call   ESCREVE2DIGITOS
        mov     a,#11011111b  ;°
        call    CARACTER
        mov     a,#'C'
        call    CARACTER

pop     dpl
```

```
pop     dph

ret
;=========================================================================
;=========================================================================
;PULADS:
;       call    CURSORPULADIR
;       ret
;=========================================================================
;=========================================================================
MARCADSX:
        push    acc
        mov     a,#11011011b
        call    CARACTER
        pop     acc
        ret
;=========================================================================
;=========================================================================
MOSTRARELOGIO:
;       push    acc             ;salva acc
        mov     a,AUXHORAS          ;escreve as horas
        call    VERIFICA255
;       call    ESCREVE2DIGITOS
        mov     a,#':'
        call    CARACTER
        mov     a,AUXMINUTOS                ;escreve os minutos
        call    VERIFICA255
;       call    ESCREVE2DIGITOS
        mov     a,#':'
        call    CARACTER
        mov     a,AUXSEGUNDOS               ;escreve os segundos
        call    VERIFICA255
;       call    ESCREVE2DIGITOS
;       pop     acc
        ret
;=========================================================================
;=========================================================================
MOSTRADS:
;       push    acc             ;salva acc
        mov     a,AUXDIASEMANA
        cjne    a,#1,TESTASEG
        mov     dptr,#MSDOMINGO
TESTASEG:
        cjne    a,#2,TESTATER
        mov     dptr,#MSSEGUNDA
TESTATER:
        cjne    a,#3,TESTAQUA
        mov     dptr,#MSTERCA
TESTAQUA:
        cjne    a,#4,TESTAQUI
        mov     dptr,#MSQUARTA
TESTAQUI:
        cjne    a,#5,TESTASEX
        mov     dptr,#MSQUINTA
TESTASEX:
        cjne    a,#6,TESTASAB
        mov     dptr,#MSSEXTA
TESTASAB:
        cjne    a,#7,TESTE255DS
        mov     dptr,#MSSABADO
TESTE255DS:
```

```
        cjne    a,#255,FIMTESTE
        mov     dptr,#MS3TRACOS
FIMTESTE:
        call    ESCREVE
ret
;=========================================================================
;=========================================================================
MOSTRADATA:
        mov     a,AUXDIA                ;escreve o dia
        call    VERIFICA255
;       call    ESCREVE2DIGITOS
        mov     a,#'/'
        call    CARACTER
        mov     a,AUXMES                ;escreve o mês
        call    VERIFICA255
;       call    ESCREVE2DIGITOS
        mov     a,#'/'
        call    CARACTER
        mov     a,AUXANO                ;escreve o ano
        call    VERIFICA255
;       call    ESCREVE2DIGITOS
;       pop     acc
        ret
;=========================================================================
;=========================================================================
TECLADO:
        ;certifica que não há tecla pressionada antes da varredura:
        clr     LIN1
        clr     LIN2
        clr     LIN3
        clr     LIN4
        setb    LIN5
        jnb     COL1,$          ;espera a tecla ser solta
        jnb     COL2,$
        jnb     COL3,$
        jnb     LIN5,$          ;a linha 5 é uma como uma coluna na horizontal
        call    ESPERA1                 ;espera 10ms (debounce)
        setb    LIN4
REPETE:
        setb    LIN2
        setb    LIN3
        ;pronto para varrer a linha 1
        clr     LIN1
        mov     a,#'1'          ;1
        jnb     COL1,VOLTA
        inc     a               ;2 ("inc" ocupa menos espaço do que "mov")
        jnb     COL2,VOLTA
        inc     a               ;3
        jnb     COL3,VOLTA
        setb    LIN1
        ;pronto para varrer a linha 2
        clr     LIN2
        inc     a               ;4
        jnb     COL1,VOLTA
        inc     a               ;5
        jnb     COL2,VOLTA
        inc     a               ;6
        jnb     COL3,VOLTA
        setb    LIN2
        ;pronto para varrer a linha 3
        clr     LIN3
```

```
        inc     a               ;7
        jnb     COL1,VOLTA
        inc     a               ;8
        jnb     COL2,VOLTA
        inc     a               ;9
        jnb     COL3,VOLTA
        setb    LIN3
        ;pronto para varrer a linha 4
        clr     LIN4
        mov     a,#'*'    ;os caracteres da última linha devem ser corrigidos
        jnb     COL1,VOLTA
        mov     a,#'0'
        jnb     COL2,VOLTA
        mov     a,#'#'
        jnb     COL3,VOLTA
        setb    LIN4
        ;pronto para varrer a linha 5
        clr     LIN5
        mov     a,#'F'    ;FLASH
        jnb     LIN1,VOLTA
        jnb     LIN2,VOLTA
        mov     a,#'R'    ;REDIAL
        jnb     LIN4,VOLTA
        setb    LIN5
        jmp     REPETE          ;começa a varrer o teclado de novo
VOLTA:
        call    ESPERA1         ;espera 10ms (debounce)
        ret
;=====================================================================
;=====================================================================
AJUSTARRELOGIO:
        call    TELAPADRAO
CORRIGEHORAS:
        mov     a,#0c1h         ;vai para HORAS
        call    WR_INSTRUCAO
        call    CURSORON
;       call    DIGITOS         ;pega 2 nibles e junta em um byte decimal
        call    TECLADO
        cjne    a,#'*',AQUIAR1
        call    CURSOROFF
        ret
AQUIAR1:
        cjne    a,#'#',AQUIAR2
        jmp     CORRIGEMINUTOS
AQUIAR2:
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIAR3
        call    CURSOROFF
        ret
AQUIAR3:
        cjne    a,#'#',AQUIAR4
        jmp     CORRIGEMINUTOS
AQUIAR4:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX          ;transforma em hex
```

```
        ;verifica se é menor ou igual a 23horas
        mov     NUMTEMP1,a
        mov     b,#24
        div     ab
        jnz     CORRIGEHORAS
        mov     AUXHORAS,NUMTEMP1
CORRIGEMINUTOS:
        mov     a,#0c4h                 ;vai para MINUTOS
        call    WR_INSTRUCAO
;       call    DIGITOS
        call    TECLADO
        cjne    a,#'*',AQUIAR5
        jmp     CORRIGEHORAS
AQUIAR5:
        cjne    a,#'#',AQUIAR6
        jmp     CORRIGESEGUNDOS
AQUIAR6:
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIAR7
        jmp     CORRIGEHORAS
AQUIAR7:
        cjne    a,#'#',AQUIAR8
        jmp     CORRIGESEGUNDOS
AQUIAR8:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        ;verifica se é menor ou igual a 59minutos
        mov     NUMTEMP1,a
        mov     b,#59
        div     ab                      ;acc deve ser 0
        jnz     CORRIGEMINUTOS          ;recomeça se houver erro
        mov     AUXMINUTOS,NUMTEMP1
CORRIGESEGUNDOS:
        mov     a,#0c7h                 ;vai para SEGUNDOS
        call    WR_INSTRUCAO
;       call    DIGITOS
        call    TECLADO
        cjne    a,#'*',AQUIAR9
        jmp     CORRIGEMINUTOS
AQUIAR9:
        cjne    a,#'#',AQUIAR10
        jmp     CORRIGEDIASEMANA
AQUIAR10:
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIAR11
        jmp     CORRIGEMINUTOS
AQUIAR11:
        cjne    a,#'#',AQUIAR12
        jmp     CORRIGEDIASEMANA
AQUIAR12:
        call    CARACTER
```

```
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        ;verifica se é menor ou igual a 59segundos
        mov     NUMTEMP1,a
        mov     b,#59
        div     ab
        jnz     CORRIGESEGUNDOS
        mov     AUXSEGUNDOS,NUMTEMP1

CORRIGEDIASEMANA:
        setb    BITPISCADIASEMANA
;       mov     a,#082h                 ;vai para dia da semana
;       call    WR_INSTRUCAO
        call    CURSOROFF
;       call    MOSTRADATA
        call    TECLADO
        cjne    a,#'*',AQUIDM1
        clr     BITPISCADIASEMANA
        call    CURSORON
        mov     a,#082h                 ;vai para dia da semana
        call    WR_INSTRUCAO
call    MOSTRADS
        call    MOSTRADATA
        jmp     CORRIGESEGUNDOS
AQUIDM1:
        cjne    a,#'3',AQUIDM2
        inc     AUXDIASEMANA
        mov     a,AUXDIASEMANA
        cjne    a,#8,CORRIGEDIASEMANA
        mov     AUXDIASEMANA,#1
        jmp     CORRIGEDIASEMANA
AQUIDM2:
        cjne    a,#'6',AQUIDM3
        dec     AUXDIASEMANA
        mov     a,AUXDIASEMANA
        cjne    a,#0,CORRIGEDIASEMANA
        mov     AUXDIASEMANA,#7
        jmp     CORRIGEDIASEMANA
AQUIDM3:
        cjne    a,#'#',CORRIGEDIASEMANA
        clr     BITPISCADIASEMANA
        call    CURSORON
        mov     a,#082h                 ;vai para dia da semana
        call    WR_INSTRUCAO
call    MOSTRADS
        call    MOSTRADATA
;       jmp     CORRIGEDIA
CORRIGEDIA:
        mov     a,#086h                 ;vai para dia
        call    WR_INSTRUCAO
;       call    DIGITOS
        call    TECLADO
        cjne    a,#'*',AQUIAR13
        jmp     CORRIGEDIASEMANA
AQUIAR13:
        cjne    a,#'#',AQUIAR14
        jmp     CORRIGEMES
AQUIAR14:
        call    CARACTER
        anl     a,#00001111b
```

```
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIAR15
        jmp     CORRIGEDIASEMANA
AQUIAR15:
        cjne    a,#'#',AQUIAR16
        jmp     CORRIGEMES
AQUIAR16:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        ;verifica se é maior que 0
        jz      CORRIGEDIA
        ;verifica se é menor ou igual a 31
        mov     NUMTEMP1,a
        mov     b,#32
        div     ab                  ;acc deve ser 0
        jnz     CORRIGEDIA          ;recomeça se houver erro
        mov     AUXDIA,NUMTEMP1
CORRIGEMES:
        mov     a,#089h             ;vai para mês
        call    WR_INSTRUCAO
;       call    DIGITOS
        call    TECLADO
        cjne    a,#'*',AQUIAR17
        jmp     CORRIGEDIA
AQUIAR17:
        cjne    a,#'#',AQUIAR18
        jmp     CORRIGEANO
AQUIAR18:
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIAR19
        jmp     CORRIGEDIA
AQUIAR19:
        cjne    a,#'#',AQUIAR20
        jmp     CORRIGEANO
AQUIAR20:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        ;verifica se é maior que 0
        jz      CORRIGEMES
        ;verifica se é menor ou igual a 12
        mov     NUMTEMP1,a
        mov     b,#13
        div     ab                  ;acc deve ser 0
        jnz     CORRIGEMES          ;recomeça se houver erro
        mov     AUXMES,NUMTEMP1
CORRIGEANO:
        mov     a,#08ch             ;vai para ano
        call    WR_INSTRUCAO
;       call    DIGITOS
        call    TECLADO
        cjne    a,#'*',AQUIAR21
```

```
        jmp     CORRIGEMES
AQUIAR21:
        cjne    a,#'#',AQUIAR22
jmp     CALENDARIOATUALIZADO
AQUIAR22:
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIAR23
        jmp     CORRIGEMES
AQUIAR23:
        cjne    a,#'#',AQUIAR24
jmp     CALENDARIOATUALIZADO
AQUIAR24:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        mov     AUXANO,a

CALENDARIOATUALIZADO:
;calendário atualizado
        call    CURSOROFF
        ret
;=====================================================================
;%%%%%%%%%%%%%%%%
;=====================================================================
PROGRELOGIO:
push    dph
push    dpl
        call    TELAPADRAOPROG
PROGHORAS:
        mov     a,#0c1h                 ;vai para HORAS
        call    WR_INSTRUCAO
        call    CURSORON
;       call    DIGITOS                 ;pega 2 nibles e junta em um byte decimal
        call    TECLADO
        cjne    a,#'*',AQUIPROG1
jmp     CALENDARIOFIM
AQUIPROG1:
        cjne    a,#'#',AQUIPROG2
        jmp     PROGMINUTOS
AQUIPROG2:
        cjne    a,#'R',AQUIPROG2R
        mov     AUXHORAS,#255
        call    ESCREVE2TRACOS
        jmp     PROGMINUTOS
AQUIPROG2R:
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIPROG3
jmp     CALENDARIOFIM
AQUIPROG3:
        cjne    a,#'#',AQUIPROG4
        jmp     PROGMINUTOS
AQUIPROG4:
```

```
        cjne    a,#'R',AQUIPROG4R
        mov     AUXHORAS,#255
        mov     a,#0c1h                 ;vai para HORAS
        call    WR_INSTRUCAO
        call    ESCREVE2TRACOS
        jmp     PROGMINUTOS
AQUIPROG4R:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX                  ;transforma em hex
        ;verifica se é menor ou igual a 23horas
        mov     NUMTEMP1,a
        mov     b,#24
        div     ab
        jnz     PROGHORAS
        mov     AUXHORAS,NUMTEMP1
PROGMINUTOS:
        mov     a,#0c4h                 ;vai para MINUTOS
        call    WR_INSTRUCAO
;       call    DIGITOS
        call    TECLADO
        cjne    a,#'*',AQUIPROG5
        jmp     PROGHORAS
AQUIPROG5:
        cjne    a,#'#',AQUIPROG6
        jmp     PROGSEGUNDOS
AQUIPROG6:
        cjne    a,#'R',AQUIPROG6R
        mov     AUXMINUTOS,#255
        call    ESCREVE2TRACOS
        jmp     PROGSEGUNDOS
AQUIPROG6R:
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIPROG7
        jmp     PROGHORAS
AQUIPROG7:
        cjne    a,#'#',AQUIPROG8
        jmp     PROGSEGUNDOS
AQUIPROG8:
        cjne    a,#'R',AQUIPROG8R
        mov     AUXMINUTOS,#255
        mov     a,#0c4h                 ;vai para MINUTOS
        call    WR_INSTRUCAO
        call    ESCREVE2TRACOS
        jmp     PROGSEGUNDOS
AQUIPROG8R:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        ;verifica se é menor ou igual a 59minutos
        mov     NUMTEMP1,a
        mov     b,#59
        div     ab                      ;acc deve ser 0
        jnz     PROGMINUTOS             ;recomeça se houver erro
        mov     AUXMINUTOS,NUMTEMP1
```

```
PROGSEGUNDOS:
        mov     a,#0c7h                 ;vai para SEGUNDOS
        call    WR_INSTRUCAO
;       call    DIGITOS
        call    TECLADO
        cjne    a,#'*',AQUIPROG9
        jmp     PROGMINUTOS
AQUIPROG9:
        cjne    a,#'#',AQUIPROG10
        jmp     PROGDIA
AQUIPROG10:
        cjne    a,#'R',AQUIPROG10R
        mov     AUXSEGUNDOS,#255
        call    ESCREVE2TRACOS
        jmp     PROGDIA
AQUIPROG10R:
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIPROG11
        jmp     PROGMINUTOS
AQUIPROG11:
        cjne    a,#'#',AQUIPROG12
        jmp     PROGDIA
AQUIPROG12:
        cjne    a,#'R',AQUIPROG12R
        mov     AUXSEGUNDOS,#255
        mov     a,#0c7h                 ;vai para SEGUNDOS
        call    WR_INSTRUCAO
        call    ESCREVE2TRACOS
        jmp     PROGDIA
AQUIPROG12R:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        ;verifica se é menor ou igual a 59segundos
        mov     NUMTEMP1,a
        mov     b,#59
        div     ab
        jnz     PROGSEGUNDOS
        mov     AUXSEGUNDOS,NUMTEMP1

PROGDIA:
        mov     a,#080h                 ;vai para dia
        call    WR_INSTRUCAO
;       call    DIGITOS
        call    TECLADO
        cjne    a,#'*',AQUIPROG13
        jmp     PROGSEGUNDOS
AQUIPROG13:
        cjne    a,#'#',AQUIPROG14
        jmp     PROGMES
AQUIPROG14:
        cjne    a,#'R',AQUIPROG14R
        mov     AUXDIA,#255
        call    ESCREVE2TRACOS
        jmp     PROGMES
AQUIPROG14R:
```

```
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIPROG15
        jmp     PROGSEGUNDOS
AQUIPROG15:
        cjne    a,#'#',AQUIPROG16
        jmp     PROGMES
AQUIPROG16:
        cjne    a,#'R',AQUIPROG16R
        mov     AUXDIA,#255
        mov     a,#080h                 ;vai para dia
        call    WR_INSTRUCAO
        call    ESCREVE2TRACOS
        jmp     PROGMES
AQUIPROG16R:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        ;verifica se é maior que 0
        jz      PROGDIA
        ;verifica se é menor ou igual a 31
        mov     NUMTEMP1,a
        mov     b,#32
        div     ab                      ;acc deve ser 0
        jnz     PROGDIA                 ;recomeça se houver erro
        mov     AUXDIA,NUMTEMP1
PROGMES:
        mov     a,#083h                 ;vai para mês
        call    WR_INSTRUCAO
;       call    DIGITOS
        call    TECLADO
        cjne    a,#'*',AQUIPROG17
        jmp     PROGDIA
AQUIPROG17:
        cjne    a,#'#',AQUIPROG18
        jmp     PROGANO
AQUIPROG18:
        cjne    a,#'R',AQUIPROG18R
        mov     AUXMES,#255
        call    ESCREVE2TRACOS
        jmp     PROGANO
AQUIPROG18R:
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIPROG19
        jmp     PROGDIA
AQUIPROG19:
        cjne    a,#'#',AQUIPROG20
        jmp     PROGANO
AQUIPROG20:
        cjne    a,#'R',AQUIPROG20R
        mov     AUXMES,#255
        mov     a,#083h                 ;vai para mês
        call    WR_INSTRUCAO
```

```
        call    ESCREVE2TRACOS
        jmp     PROGANO
AQUIPROG20R:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        ;verifica se é maior que 0
        jz      PROGMES
        ;verifica se é menor ou igual a 12
        mov     NUMTEMP1,a
        mov     b,#13
        div     ab                      ;acc deve ser 0
        jnz     PROGMES                 ;recomeça se houver erro
        mov     AUXMES,NUMTEMP1
PROGANO:
        mov     a,#086h                 ;vai para ano
        call    WR_INSTRUCAO
;       call    DIGITOS
        call    TECLADO
        cjne    a,#'*',AQUIPROG21
        jmp     PROGMES
AQUIPROG21:
        cjne    a,#'#',AQUIPROG22
jmp     PROGDIASSEMANA
AQUIPROG22:
        cjne    a,#'R',AQUIPROG22R
        mov     AUXANO,#255
        call    ESCREVE2TRACOS
        jmp     PROGDIASSEMANA
AQUIPROG22R:
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUIPROG23
        jmp     PROGMES
AQUIPROG23:
        cjne    a,#'#',AQUIPROG24
jmp     PROGDIASSEMANA
AQUIPROG24:
        cjne    a,#'R',AQUIPROG24R
        mov     AUXANO,#255
        mov     a,#083h                 ;vai para ano
        call    WR_INSTRUCAO
        call    ESCREVE2TRACOS
        jmp     PROGDIASSEMANA
AQUIPROG24R:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        mov     AUXANO,a

PROGDIASSEMANA:
        mov     BYTEBIT1,AUXDIASEMANA
DSDOMINGO:
        mov     a,#089h                 ;vai para D
        call    WR_INSTRUCAO
        call    TECLADO
```

```
        cjne    a,#'*',PROGDS01
        jmp     PROGANO
PROGDS01:
        cjne    a,#'#',PROGDS02
        jmp     DSSEGUNDA
PROGDS02:
        cjne    a,#'R',PROGDS03
        setb    BYTEBIT1.6
        call    MARCADSX
        jmp     DSSEGUNDA
PROGDS03:
        cjne    a,#'F',DSDOMINGO
        clr     BYTEBIT1.6
        mov     a,#'D'
        call    CARACTER

DSSEGUNDA:
        mov     a,#08ah             ;vai para 2
        call    WR_INSTRUCAO
        call    TECLADO
        cjne    a,#'*',PROGDS04
        jmp     DSDOMINGO
PROGDS04:
        cjne    a,#'#',PROGDS05
        jmp     DSTERCA
PROGDS05:
        cjne    a,#'R',PROGDS06
        setb    BYTEBIT1.5
        call    MARCADSX
        jmp     DSTERCA
PROGDS06:
        cjne    a,#'F',DSSEGUNDA
        clr     BYTEBIT1.5
        mov     a,#'2'
        call    CARACTER

DSTERCA:
        mov     a,#08bh             ;vai para 3
        call    WR_INSTRUCAO
        call    TECLADO
        cjne    a,#'*',PROGDS07
        jmp     DSSEGUNDA
PROGDS07:
        cjne    a,#'#',PROGDS08
        jmp     DSQUARTA
PROGDS08:
        cjne    a,#'R',PROGDS09
        setb    BYTEBIT1.4
        call    MARCADSX
        jmp     DSQUARTA
PROGDS09:
        cjne    a,#'F',DSTERCA
        clr     BYTEBIT1.4
        mov     a,#'3'
        call    CARACTER

DSQUARTA:
        mov     a,#08ch             ;vai para 4
        call    WR_INSTRUCAO
        call    TECLADO
        cjne    a,#'*',PROGDS10
```

```
        jmp     DSTERCA
PROGDS10:
        cjne    a,#'#',PROGDS11
        jmp     DSQUINTA
PROGDS11:
        cjne    a,#'R',PROGDS12
        setb    BYTEBIT1.3
        call    MARCADSX
        jmp     DSQUINTA
PROGDS12:
        cjne    a,#'F',DSQUARTA
        clr     BYTEBIT1.3
mov     a,#'4'
call    CARACTER

DSQUINTA:
        mov     a,#08dh                 ;vai para 5
        call    WR_INSTRUCAO
        call    TECLADO
        cjne    a,#'*',PROGDS13
        jmp     DSQUARTA
PROGDS13:
        cjne    a,#'#',PROGDS14
        jmp     DSSEXTA
PROGDS14:
        cjne    a,#'R',PROGDS15
        setb    BYTEBIT1.2
        call    MARCADSX
        jmp     DSSEXTA
PROGDS15:
        cjne    a,#'F',DSQUINTA
        clr     BYTEBIT1.2
        mov     a,#'5'
        call    CARACTER

DSSEXTA:
        mov     a,#08eh                 ;vai para 6
        call    WR_INSTRUCAO
        call    TECLADO
        cjne    a,#'*',PROGDS16
        jmp     DSQUINTA
PROGDS16:
        cjne    a,#'#',PROGDS17
        jmp     DSSABADO
PROGDS17:
        cjne    a,#'R',PROGDS18
        setb    BYTEBIT1.1
        call    MARCADSX
        jmp     DSSABADO
PROGDS18:
        cjne    a,#'F',DSSEXTA
        clr     BYTEBIT1.1
        mov     a,#'6'
        call    CARACTER

DSSABADO:
        mov     a,#08fh         ;vai para S
        call    WR_INSTRUCAO
        call    TECLADO
        cjne    a,#'*',PROGDS19
        jmp     DSSEXTA
```

```
PROGDS19:
        cjne    a,#'#',PROGDS20
        jmp     DIASDASEMANAPROGRAMADOS
PROGDS20:
        cjne    a,#'R',PROGDS21
        setb    BYTEBIT1.0
        call    MARCADSX
        jmp     DIASDASEMANAPROGRAMADOS
PROGDS21:
        cjne    a,#'F',DSSABADO
        clr     BYTEBIT1.0
mov     a,#'S'
call    CARACTER
DIASDASEMANAPROGRAMADOS:

mov     AUXDIASEMANA,BYTEBIT1

CONFIGTEMP:
        mov     a,#0cbh                 ;vai para temperatura
        call    WR_INSTRUCAO
;       call    DIGITOS
        call    TECLADO
        cjne    a,#'*',AQUITEMP1
        jmp     DSSABADO
AQUITEMP1:
        cjne    a,#'#',AQUITEMP2
jmp     CALENDARIOFIM
AQUITEMP2:
        cjne    a,#'R',AQUIPROGTEMP2R
        mov     AUXREGTEMP,#255
        call    ESCREVE2TRACOS
jmp     CALENDARIOFIM
AQUIPROGTEMP2R:
        call    CARACTER
        anl     a,#00001111b
        swap    a
        mov     NUMTEMP1,a
        call    TECLADO
        cjne    a,#'*',AQUITEMP3
        jmp     DSSABADO
AQUITEMP3:
        cjne    a,#'#',AQUITEMP4
jmp     CALENDARIOFIM
AQUITEMP4:
        cjne    a,#'R',AQUIPROGTEMP4R
        mov     AUXREGTEMP,#255
        mov     a,#0cbh                 ;vai para temperatura
        call    WR_INSTRUCAO
        call    ESCREVE2TRACOS
jmp     CALENDARIOFIM
AQUIPROGTEMP4R:
        call    CARACTER
        anl     a,#00001111b
        orl     a,NUMTEMP1
        call    BCDHEX
        mov     AUXREGTEMP,a

;calendário atualizado
CALENDARIOFIM:
        call    CURSOROFF
        pop     dpl
```

```
        pop     dph
        ret
;===============================================================================
;===============================================================================
;EXEC:
;       clr     EN
;       setb    EN
;ESPERA2: ;pelo menos 20ms
;       jnb     TF0,$
;       jb      TF0,$
ESPERA1: ;pelo menos 10ms
        jnb     TF0,$
        jb      TF0,$
        jnb     TF0,$
        ret
;===============================================================================
;===============================================================================
BCDHEX:
        mov     R1,a
        anl     a,#11110000b
        swap    a
        mov     b,a
        mov     a,#10
        mul     ab
        mov     b,a
        mov     a,R1
        anl     a,#00001111b
        add     a,b
        ret
;===============================================================================
;===============================================================================
VERIFICA255:
        cjne    a,#255,VALORNAO255
call    ESCREVE2TRACOS
        ret
VALORNAO255:
        call    ESCREVE2DIGITOS
        ret
;===============================================================================
;===============================================================================
ESCREVE2TRACOS:
        mov     a,#'-'
        call    CARACTER
        call    CARACTER
        ret
;===============================================================================
;===============================================================================
ESCREVE3DIGITOS:   ;escreve '255', nível 3
        mov     b,#100
        div     ab
        orl     a,#00110000b   ;adiciona a metade mais signif. (ver mapa LCD)
        call    CARACTER
        mov     a,b
ESCREVE2DIGITOS:
        mov     b,#10
        div     ab
        orl     a,#00110000b   ;adiciona a metade mais signif. (ver mapa LCD)
        call    CARACTER
        mov     a,b
        orl     a,#00110000b
        call    CARACTER
```

```
        ret
;=======================================================================
;=======================================================================
WR_INSTRUCAO:       ;move cursor, nível 2
        push    dph
        push    dpl
        mov     dptr,#lcd_wr_inst
        movx    @dptr,a
        call    ESPERABF
        pop     dpl
        pop     dph
        ret
;=======================================================================
;=======================================================================
ESPERABF:   ;nível 1
        push    acc
LOOP:
        mov     dptr,#lcd_rd_inst
        movx    a,@dptr
        rlc     a
        jc      LOOP
        pop     acc
        ret
;=======================================================================
;=======================================================================
CURSORPULADIR:      ;cursor desloca para a direita
        push    acc
        mov     a,#00010100b
        call    WR_INSTRUCAO
        pop     acc
        ret
;=======================================================================
;=======================================================================
;CURSORPULAESQ:   ;cursor desloca para a espera
;       push    acc
;       mov     a,#00010000b
;       call    WR_INSTRUCAO
;       pop     acc
;       ret
;=======================================================================
;=======================================================================
CURSORON:   ;liga display, desl. cursor e pisca cursor
        push    acc
        mov     a,#00001101b
        call    WR_INSTRUCAO
        pop     acc
        ret
;=======================================================================
;=======================================================================
CURSOROFF:  ;liga display, desl. cursor
        push    acc
        mov     a,#00001100b
        call    WR_INSTRUCAO
        pop     acc
        ret
;=======================================================================
;=======================================================================
VOLTACURSOR:
        push    acc
        mov     a,#00010000b
        call    WR_INSTRUCAO
```

```
        pop     acc
        ret
;=====================================================================
;=====================================================================
AVANCACURSOR:
        push    acc
        mov     a,#00010100b
        call    WR_INSTRUCAO
        pop     acc
        ret
;=====================================================================
;=====================================================================
LIMPA:                          ;limpa o LCD
        push    acc
        mov     a,#00000001b
        call    WR_INSTRUCAO
        pop     acc
        ret
;=====================================================================
;=====================================================================
CARACTER:   ;escreve 'a', nível 2
        push    dph
        push    dpl
        mov     dptr,#lcd_wr_dado
        movx    @dptr,a
        call    ESPERABF
        pop     dpl
        pop     dph
        ret
;=====================================================================
;=====================================================================
ESCREVE:    ;escreve string, nível 3
        push    acc
ESCREVE1:
        mov     a,#0
        movc    a,@a+dptr       ;manda caracter pro acc
        jz      SAI             ;verifica se a mensagem acabou
        call    CARACTER
        inc     dptr
        jmp     ESCREVE1
        SAI:
        pop     acc
        ret
;=====================================================================
;=====================================================================
PULALINHA:  ;nível 3
        push    acc
        mov     a,#0c0h         ;vai para a 2 linha
        call    WR_INSTRUCAO
        pop     acc
        ret
;=====================================================================
;=====================================================================
PAUSA:      ;(0,15seg)
        mov     RSOM1,#251
        mov     RSOM2,#201
        mov     RSOM3,#4
DENOVO0:
        djnz    RSOM1,DENOVO0
        mov     RSOM1,#251
        djnz    RSOM2,DENOVO0
```

```
        mov     RSOM2,#201
        djnz    RSOM3,DENOVO0
        ret
;=========================================================================
;=========================================================================
SOM1:  ;(380Hz , 0,3seg)
        mov     RSOM1,#47
        mov     RSOM2,#28;14
        mov     RSOM3,#228
        mov     RSOM4,#2;nada
DENOVO1:
        djnz    RSOM1,DENOVO1
        mov     RSOM1,#47
        djnz    RSOM2,DENOVO1
        mov     RSOM2,#28;14
        cpl     SOM
        djnz    RSOM3,DENOVO1
        djnz    RSOM4,DENOVO1;nada
        setb    SOM
        ret
;=========================================================================
;=========================================================================
SOM2:  ;(1100Hz , 0,3seg)    66 x 10
        mov     RSOM1,#225
        mov     RSOM2,#2;nada
        mov     RSOM3,#220
        mov     RSOM4,#5;3
DENOVO2:
        djnz    RSOM1,DENOVO2
        mov     RSOM1,#225
        djnz    RSOM2,DENOVO2;nada
        mov     RSOM2,#2;nada
        cpl     SOM
        djnz    RSOM3,DENOVO2
        djnz    RSOM4,DENOVO2
        setb    SOM
        ret
;=========================================================================
;Mensagens usadas no programa:
MSRECORRER:
db      "Recorrer a mem.?",0
MS1SIM2NAO:
db      " 1-Sim    2-Nao",0
;MSRECORRENDO:
;db      "Recorrendo...",0
MSINICIANDO:
db      "  Iniciando...",0
MSDOMINGO:
db      "DOM ",0
MSSEGUNDA:
db      "SEG ",0
MSTERCA:
db      "TER ",0
MSQUARTA:
db      "QUA ",0
MSQUINTA:
db      "QUI ",0
MSSEXTA:
db      "SEX ",0
MSSABADO:
db      "SAB ",0
```

```
MS3TRACOS:
db        "--- ",0
MSD23456S:
db        "D23456S",0
;MSTECLADO:
;db       "TECLADO: ",0
MSDIGITEOCH:
db        "Digite o CH: ",0
MSSTATUS:
db        "Status: ",0
MSLIGADO:
db        "Ligado ",0
MSDESLIGADO:
db        " Desl. ",0
MSLIGAR:
db        "ligar ",0
MSDESL:
db        "desl. ",0
MSOCANAL:
db        "o Canal ",0
;MSPARAMETROSPARA:
;db       "Parametros para",0
MSPROGRAMEQUANDO:
db        "Programe quando",0
MSREN:
db        "Ren.: ",0
MSLUZDEFUNDO:
db        "  Luz de Fundo",0
MSLIGADA:
db        "Ligada ",0
MSLINHATELEFONICA:
db        "Linha Telefonica",0
;db       "1234567890123456",0
;MSTEMPORESTANTE:
;db       " Restam XXX seg.",0
MSCHAMADAEMCURSO:
db        "Chamada em curso",0
MSNOVASENHA:
db        "   Nova senha:",0
MSSETAS:
db        "---->      <----",0
;db       "1234567890123456",0
end
```

Apêndice 4

```
; ========================================================================
; Sistema de Automação Residencial - Módulo de Acionamento
; Franco Ripoll Leite
;========================================================================
FLAGREG1      equ     30h
BYTERECEBIDO          equ     FLAGREG1
CANAL1                equ     FLAGREG1+1
CANAL2                equ     FLAGREG1+2
CANAL3                equ     FLAGREG1+3

CANAL         equ     20h;FLAGREG1+1

SAIDA1                equ     P2.2;FLAGBIT1
SAIDA2                equ     P2.1;FLAGBIT1
SAIDA3                equ     P2.0;FLAGBIT1

;org     0000h
;jmp     INICIO

;INICIO:
mov     CANAL,#0
mov     c,P3.3
mov     CANAL.4,c
mov     c,P3.4
mov     CANAL.3,c
mov     c,P3.5
mov     CANAL.2,c
mov     c,P3.6
mov     CANAL.1,c
mov     c,P3.7
mov     CANAL.0,c

mov     a,CANAL
rl      a
rl      a
rl      a
mov     CANAL1,a
inc     CANAL
mov     a,CANAL
rl      a
rl      a
rl      a
mov     CANAL2,a
inc     CANAL
mov     a,CANAL
rl      a
rl      a
rl      a
mov     CANAL3,a


        mov     SCON,#01010000b     ;serial em modo 1, REN = 1
        mov     PCON,#00000000b     ;SMOD = 0
        mov     TMOD,#00100001b     ;timer0 no modo 1 e timer1 no modo 2
        mov     TH1,#160
        mov     TL1,#160
```

```
        mov     IE,#0
        setb    TR1             ;liga o TIMER1

AGUARDANDO:
        clr     RI
        jnb     RI,$
        mov     a,SBUF
        mov     BYTERECEBIDO,a
anl     a,#11111000b
xrl     a,CANAL1
jnz     VERIFICACH2
mov     a,BYTERECEBIDO
rrc     a
jc      LIGA1
setb    SAIDA1          ;desliga canal
jmp     AGUARDANDO
LIGA1:
clr     SAIDA1          ;liga canal
jmp     AGUARDANDO

VERIFICACH2:
        mov     a,BYTERECEBIDO
anl     a,#11111000b
xrl     a,CANAL2
jnz     VERIFICACH3
mov     a,BYTERECEBIDO
rrc     a
jc      LIGA2
setb    SAIDA2          ;desliga canal
jmp     AGUARDANDO
LIGA2:
clr     SAIDA2          ;liga canal
jmp     AGUARDANDO

VERIFICACH3:
        mov     a,BYTERECEBIDO
anl     a,#11111000b
xrl     a,CANAL3
jnz     AGUARDANDO
mov     a,BYTERECEBIDO
rrc     a
jc      LIGA3
setb    SAIDA3          ;desliga canal
jmp     AGUARDANDO
LIGA3:
clr     SAIDA3          ;liga canal
jmp     AGUARDANDO


jmp     $
end
```