Universidade Federal do Rio Grande do Norte
Centro de Tecnologia - CT
Departamento de Engenharia Elétrica

Central Digital de Alarme

ELE1717 - Grupo 01 - Problema 06 - Projeto

Isaac de Lyra Junior
João Matheus Bernardo Resende
Lucas Augusto Maciel da Silva
Marcelo Ferreira Mota Júnior
Rodrigo de Lima Santana

Natal, 24 de agosto de 2021

Resumo

O objetivo deste trabalho é o de implementar uma central digital de alarme capaz de acionar uma sirene sempre que um sensor de movimento for acionado. O projeto deve ser implementado em linguagem C e deverá fazer uso de um teclado 4x4, um LCD e comunicação UART. Para a atual semana, apenas o projeto deve feito, então foram criados fluxogramas de funcionamento do programa, foram definidos os componentes específicos necessários à implementação, assim como as configurações necessárias dos recursos utilizados.

Palavras-chave: central digital de alarme, comunicação UART, linguagem C, teclado 4x4, LCD.

Lista de Imagens

Sumário

1 INTRODUÇÃO

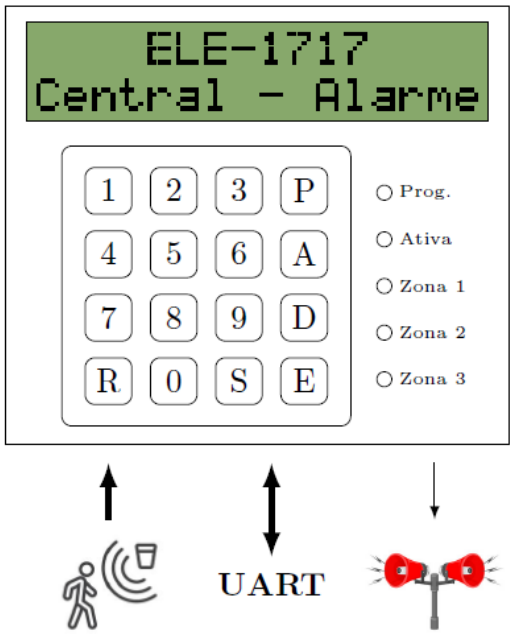
Nos últimos anos, tem sido uma tendência a utilização de tecnologia no dia-a-dia do ser humano, em diversos setores e áreas de sua vida. No que diz respeito à tecnologia no controle e automação residencial, há diversas ofertas de produtos para um mercado amplo e crescente, para quem pode pagar por mais comodidade, segurança, bem-estar e aperfeiçoamento de tarefas. A comunicação entre diferentes dispositivos é de fundamental importância quando trabalhamos com sistemas embarcados. A interface mais utilizada no cotidiano é realizada entre o microcontrolador e o computador.

Dentre as diversas tecnologias que vem sido trazidas às residências, o alarme sensorial é uma dessas tecnologias mais simples e presentes em casas e apartamentos. Essa tecnologia permite identificar qualquer invasão de área de sua residência, trazendo mais segurança ao usuário, que sabe que será alertado. No entanto, quando o alarme for acionado, tocando as sirenes, indicará a presença de pessoas não convidadas ou não autorizadas para estarem no devido recinto.

A partir dos conhecimentos de Want, Schilit e Jenson (2015), a integração de sensores nos mais diversos equipamentos utilizados no cotidiano humano visa a melhoria da qualidade de vida e segurança, sendo uma realidade bem vinculada nos dias de hoje. A técnica em incorporar sensores nos equipamentos utilizados no nosso cotidiano, aliada aos recursos de comunicações em equipamentos de segurança, configura um mecanismo ainda mais robusto, trazendo consigo a integração com a Industria 4.0.

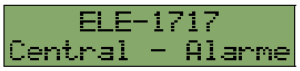
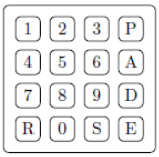
O atual trabalho tem como objetivo implementar uma central digital de alarme capaz de acionar uma sirene sempre que um sensor de movimento for acionado. O projeto deve ser implementado em linguagem C e deverá fazer uso de um teclado 4x4, um LCD e comunicação UART. A Figura 1 mostra a aparência geral da central e a Figura 2 descreve seus elementos.

Figura 1 – Aparência da interface homem-máquina da central de alarme



Fonte: Problema 06 (2021).

Figura 2 – Elementos da interface homem-máquina do programador horário

Elemento	Descrição
	Display LCD para exibição dos parâmetros e dados da central de alarme
	Teclado da central de alarme
○ Prog. ○ Ativa ○ Zona 1 ○ Zona 2 ○ Zona 3	Led para sinalização do modo de programação Led para sinalização do modo ativo Led para sinalização de movimentação na zona 1 Led para sinalização de movimentação na zona 2 Led para sinalização de movimentação na zona 3
	Canal de entrada dos sensores de presença (8 entradas)
	Canal de saída para a sirene
UART	Canal de comunicação UART

Fonte: Problema 06 (2021).

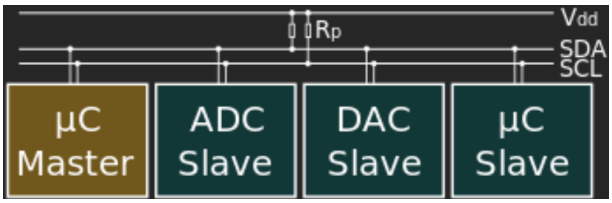
2 DESENVOLVIMENTO

O projeto da central de alarme dependerá de alguns componentes, que permitirão o correto funcionamento do sistema. A fundamentação acerca dos componentes utilizados neste projeto será descrita neste capítulo.

2.1 Protocolo TWI (I^2C)

Segundo Embed (2013), o protocolo Interface serial de 2 fios (I^2C), trata-se de um protocolo de comunicação serial que usa apenas dois pinos de um microcontrolador, isto é, relógio serial (SCL) e dados seriais (SDA). A linha SCL é responsável pelo clock do barramento, e a linha SDA pela transmissão de dados. Cada dispositivo escravo tem um endereço escravo ou um nome pelo qual eles respondem. Uma vez que um mestre envia um endereço escravo válido, aquele escravo sozinho responderá às perguntas do mestre e todos os outros escravos ignorarão qualquer conversa entre o mestre e aquele escravo em particular conforme a Figura 3.

Figura 3 – Protocolo I^2C



Fonte: I^2C Bus.

Dessa forma, os dispositivos escravos experimentam a sequência Iniciar, ele espera um endereço escravo de 7 bits juntamente com um especificador de leitura/gravação no MSB, sendo (0 - para gravação e 1 - leitura). Se o especificador estiver definido para gravar, os próximos dados escritos serão o endereço do registro para o qual os dados consecutivos devem ser gravados. O dispositivo incrementa automaticamente o ponteiro de registro após uma gravação completa de sucesso. Por outro lado, se o especificador estiver definido para ler, os dados recebidos do barramento retornarão o valor do registro para o qual o ponteiro da pilha foi apontado pela última vez e os registros consecutivos que o seguem.

No barramento I^2C são mantidos o valor digital alto em ambas as linhas de comunicação, para se iniciar a comunicação. Segundo I^2C Bus (2012), SDA é trazido para o valor digital baixo pelo mestre. Para escrever dados no barramento, SCL pulsa, e a cada pulso, o valor em SDA é lido como um bit, começando do MSB ilustrado na Figura 4.

Figura 4 – Dados de barramento

Fonte: I^2C Bus.

A comunicação sempre é iniciada e finalizada pelo mestre. O mestre é responsável por gerar o sinal de *clock* e enviar o endereço para identificar o escravo. Para iniciar a comunicação, o mestre coloca a linha SDA em nível baixo e, logo em seguida, ativa o clock (SCL). Após iniciar a comunicação, o mestre envia os 7 bits do endereço e um oitavo bit indicando a operação a ser realizada (leitura ou escrita). Após enviar o endereço e a operação, o mestre espera um retorno do escravo.

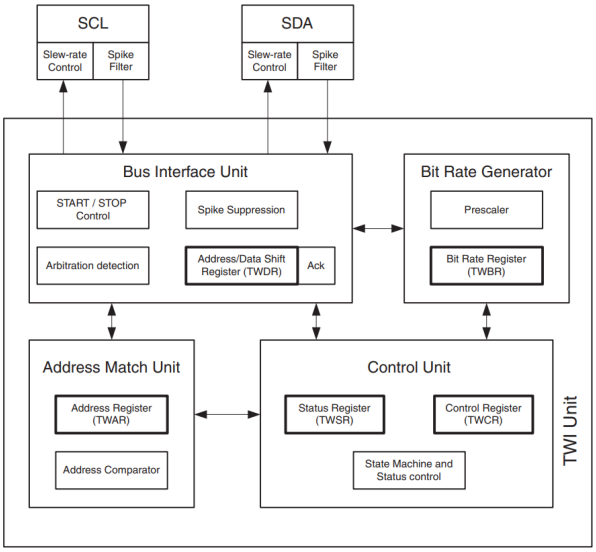
Após o mestre receber o ACK do escravo, o dado é transmitido, conforme a operação:

a. Se for escrita (bit=0), o mestre envia os 8 bits de dados para o escravo;

b. Se for leitura (bit=1), o escravo envia os 8 bits de dados para o mestre;

Segundo Kovalhuk, G. (2020), a interação entre o TWI e o ATMEGA328p, é dada de acordo com a Figura 5.

Figura 5 – TWI no ATMEGA328p



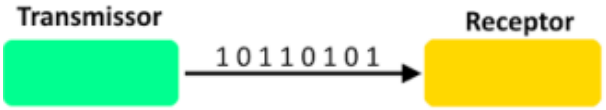
Fonte: Atmel Corporation (2015)

A Figura 5 demonstra a composição do modulo TWI, o mesmo o composto por vários submódulos, Todos os registros desenhados em uma linha que são acessíveis através do barramento de dados AVR.

2.2 Protocolo UART

O entendimento do protocolo UART é facilitado a partir do entendimento do conceito serial. Percebe-se que para haver a comunicação precisamos ter um transmissor e um receptor conforme a Figura 6, para que uma dada comunicação serial seja realizada adequadamente.

Figura 6 – Comunicação serial

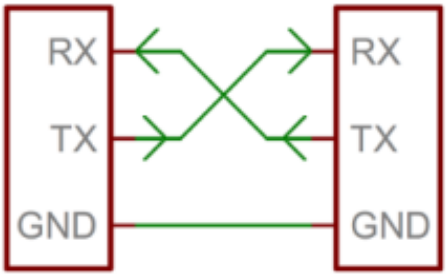


Fonte: Rech R. A. (2020)

Segundo Rech (2020), o conceito de comunicação serial trata-se uma espécie de fio ao qual passa um bit por vez de um transmissor para o receptor. Tais transmissões existem determinados modelos, são eles:

a. Comunicação Assíncrona: tem como característica uma dada velocidade de transmissão e recepção, diante de não haver uma sincronia física entre o transmissor e receptor conforme visto na Figura 7. Sendo assim, para o dispositivo compreender o que o outro está "dizendo", ambos precisam estarem "conversando" na mesma velocidade, isso é chamado de *Baud Rate* e possui unidade *bits* por segundo (bps).

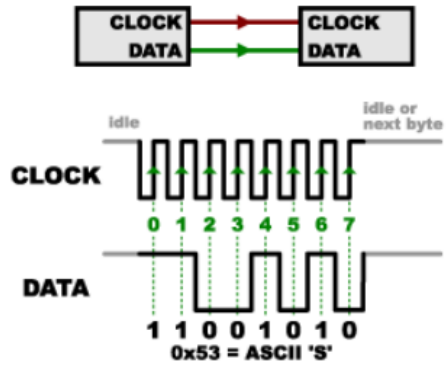
Figura 7 – Dispositivos de uma comunicação assíncrona



Fonte: Rech R. A. (2020)

b. Comunicação Síncrona: Diferentemente da comunicação assíncrona, a velocidade da comunicação é feita pelo *clock*, além, claro, do TX e RX visto na Figura 8.

Figura 8 – Comunicação síncrona

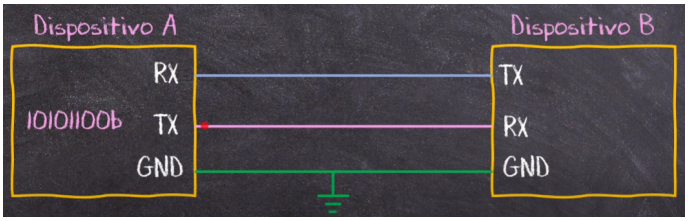


Fonte: Rech R. A. (2020)

De acordo com os conhecimento de Rech (2020), na grande maioria dos microcontroladores, existe um periférico chamado USART (*Universal Synchronous Asynchronous Receiver Transmitter*), que controla toda a comunicação serial, seja ela, síncrona ou assíncrona do microcontrolador, logo, é responsável pelo controle do I^2C, SPI e da UART. Para o referido projeto iremos utilizar o UART (*Universal Asynchronous Transmitter*).

Na comunicação serial assíncrona, por exemplo, suponhamos que desejamos transmitir a mensagem "10101100b" de um dispositivo a outro ilustrado na Figura 9. Tal informação será a priori armazenada num registrador interno de deslocamento à direita do transmissor e a cada pulso de *clock* interno do dispositivo do transmissor, o *bit* menos significativo ele vai para a linha de transmissão (TX-RX) e vai até toda mensagem ser passada.

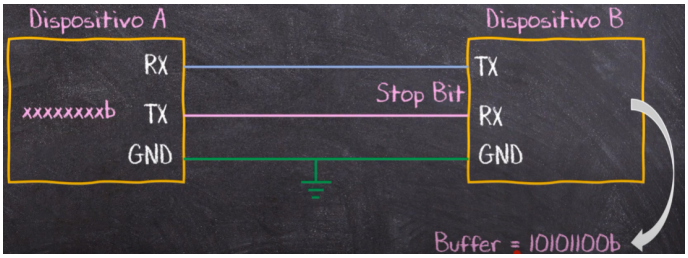
Figura 9 – Transmissão de comunicação assíncrona



Fonte: Rech R. A. (2020)

Segundo Rech (2020), quando toda a mensagem foi enviada é também enviado um outro *bit* do transmissor ao receptor, *bit* denominado "*Stop Bit*", onde este informa para o receptor que a mensagem que está armazenada no registrador de deslocamento transmissor, ele vá ao *buffer* visualizada na Figura 10. Isso faz com que uma nova mensagem possa ser recebida na linha de transmissão (TX-RX).

Figura 10 – Transmissão de comunicação assíncrona o *buffer*

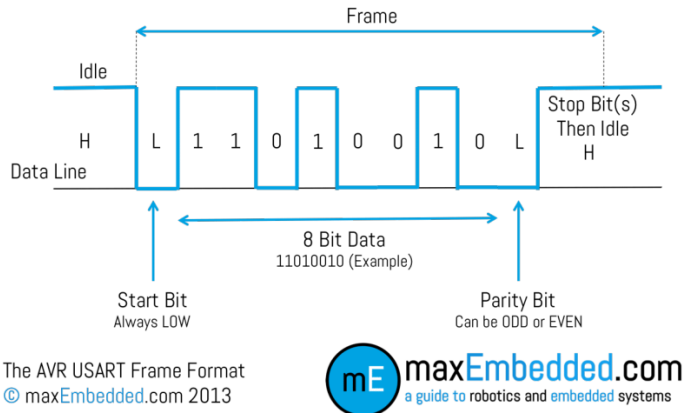


Fonte: Rech R. A. (2020)

Vale ressaltar, que o *Baud Rate* do transmissor e do receptor precisam serem iguais ou bastantes próximos, diante de não haver um fio de *clock* conforme visto na comunicação síncrona anteriormente. Caso contrário, isso significa, na prática, quando temos velocidades de transmissões diferentes, o receptor identificará que um dado *bit* não chegou, gerando assim, um erro de transmissão.

Ainda segundo Rech (2020), na comunicação UART conforme a Figura 11, os *bits* são transmitidos um na sequência do outro, respeitando o *baud rate* definido. Para o início da comunicação é dado um *Start Bit*, seguido dos oito bits de dados, tendo um caractere e depois do *Stop Bit*.

Figura 11 – Comunicação UART



Fonte: Comunicação UART

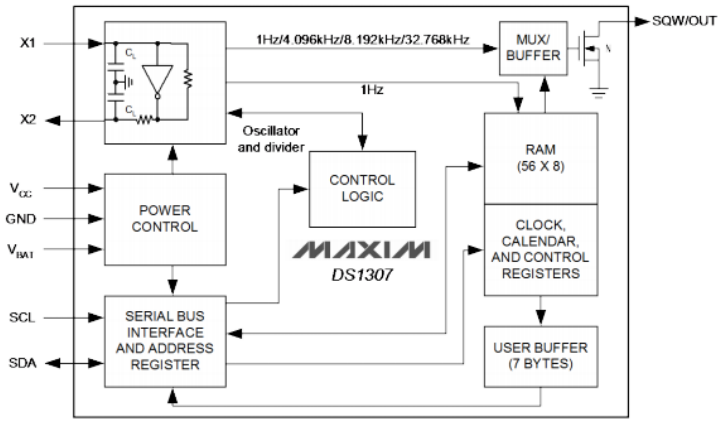
Como a comunicação é padrão, ou seja, qualquer dispositivo irá trabalhar exatamente da mesma forma.

2.3 RTC - Real Time Clock

O *real-time clock* (RTC) é um circuito integrado, responsável por manter o controle da passagem de tempo. Neste projeto, como a comunicação com o RTC será feita por

meio do protocolo I^2C, foi escolhido o modelo DS1307. Suas funcionalidades são mais que suficientes para suprir as demandas do projeto, visto que será necessário apenas consultar horário e data, durante a criação do arquivo de log. A figura 12 mostra o diagrama de blocos do RTC DS1307.

Figura 12 – Diagrama de blocos do RTC DS1307

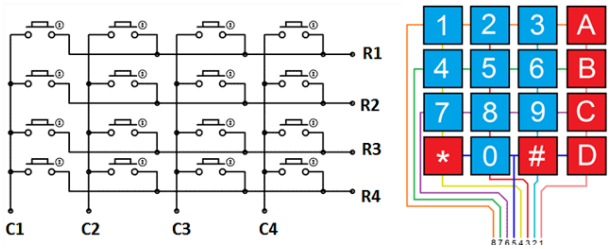


Fonte: (MAXIM, 2015).

2.4 Keypad 4x4

O teclado matriz trata-se da parte de programação para obter as informações a partir do pressionamento de uma determinada tecla, ilustrada na Figura 13. A partir dos conhecimentos de Jonh (2018), os teclados matriz são usados em sistemas painéis de controle onde a *Human Machine Interface* (HMI) é necessária para alterar os parâmetros operacionais de um sistema-máquina. O teclado matriz possui algumas teclas dispostas tanto em linhas e colunas. As teclas são identificadas como sendo "M_{11}", isto é, referente ao termo da primeira linha e da primeira coluna até M_{44}, por serem 16 termos da matriz.

Figura 13 – Keypay



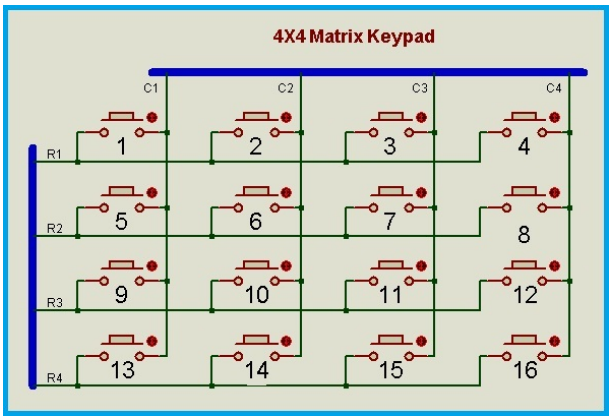
Fonte: Jayant R. (2015)

A vantagem desses teclados de matriz é que todos os *switches* da matriz podem ser lidos usando menor número de pinos de I/O de um processador ou microcontrolador. Suponhamos, por exemplo, temos um teclado 4×4 matriz, que contém teclas $4 \times 4 = 16$.

Dessa forma, precisamos de 16 pinos do microcontrolador para ler o status desses 16 botões de interruptores ou apertar individualmente. Porém, utilizando o tipo matriz de configuração apenas 8 Pinos são suficientes. Sendo assim, a contagem de pinos é reduzida pela metade.

No intuito em facilitar o entendimento por parte do leitor, vamos organizar esses 16 botões como elementos de uma Matriz 4×4, ou seja, em quatro linhas e quatro colunas. Estes estão eletricamente conectados conforme ilustrado na Figura 14.

Figura 14 – Circuito interno ao *Keypad*



Fonte: Jayant R. (2015)

Como cada interruptor de botão tem dois terminais, o Terminal 1 de cada interruptor em uma linha é conectado a outros *switches*. Da mesma forma, os interruptores em cada coluna têm um de seus terminais conectados aos interruptores restantes na mesma coluna, distribuindo assim 4 pinos representando quatro linhas e 4 pinos representando quatro colunas.

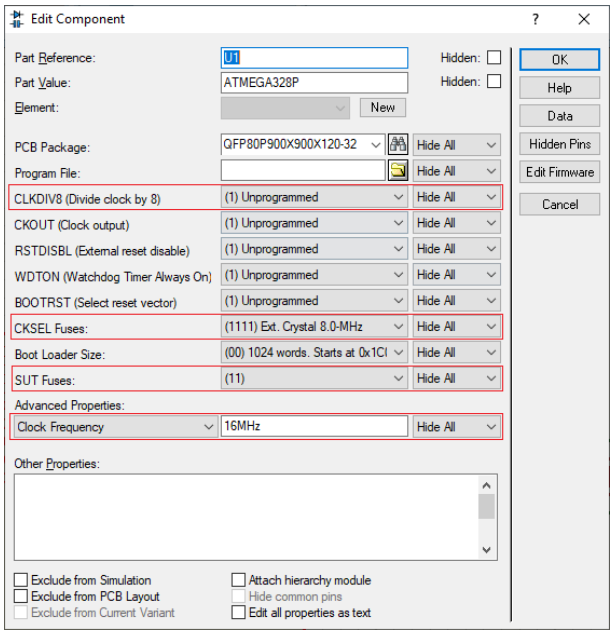
3 PROJETO

3.1 Clock do sistema

O *clock* utilizado para este projeto é de 16 MHz, sua necessidade está na melhoria da precisão na comunicação via UART. Para configurar o ATmega328P para funcionar em 16 MHz, deve-se configurar os *fusebits*. Estes *fusebits*, de acordo com Vargas (2020), são responsáveis por controlar, por meio de 3 registradores, vários recursos importantes do ATmega328P. Entre eles: origem e comportamento do clock; habilitação do *reset* externo; *WatchDog Timer*; e programação do próprio microcontrolador via SPI. Como somente será preciso configurar o clock, as únicas mudanças nos fusebits serão feitas no registrador CKSEL (ou Fuse Low Byte).

O CKSEL deve ser configurado no modo *"Low Power Crystal Oscillator"* para frequências entre "8.0 - 16.0" (é a mesma configuração do Arduino) e "Start-up Time" (tempo de inicialização) configurado, preferencialmente, para "Crystal Oscillator, slowly rising power" (Como o projeto será simulado via software, esta última opção se torna irrelevante). Além disso, o bit CKDIV8, que divide o clock por 8, deve ser desabilitado setando nível lógico alto nele. Essas configurações podem ser feitas por meio do Proteus, clicando com o botão direito sobre o ATmega328P e selecionando "Edit Properties". As configurações devem estar como mostradas na Figura 15. Note que, no Proteus, é necessário definir o valor do clock em "Advanced Properties", ou as configurações não funcionarão.

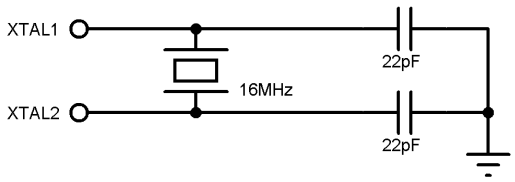
Figura 15 – Configurações do clock pelo Proteus



Fonte: Elaborado pelos autores.

Por fim, para efetivamente adicionar um *clock* externo de 16MHz, deve-se montar um circuito com um cristal oscilador de 16MHz como o apresentado no circuito da Figura 16. O valor dos capacitores está definido em tabela no *datasheet* do ATmega328P para o modo *"Low Power Crystal Oscillator"* para frequências entre "8.0 - 16.0".

Figura 16 – Conexões do cristal oscilador



Fonte: Elaborado pelos autores.

3.2 Configurações do Protocolo TWI/I^{2}C

3.2.1 TWBR - TWI Bit Register

É o registrador utilizado para fazer o cálculo da frequência de comunicação. Como esperamos que a frequência seja de 100 KHz, o valor atribuído ao mesmo deve ser de 18 (em Hexadecimal é igual a h12 e em Binário é igual a b0001 0010).

3.2.2 TWCR - TWI Control Register

Esse registrador fica responsável pelo controle da comunicação, o mesmo tem 8 bits sendo possível ativar ou desativar algumas flags ou bits de controle. Nesse registrador temos os seguintes bits:

a. TWIE: Habilita a interrupção;

b. Não utilizado

c. TWWC: Flag de aviso de colisão no barramento. É setado quando TWDR recebe um dado com TWINT em nível baixo;

d. TWSTO: Envia uma condição de STOP;

e. TWSTA: Envia uma condição de START;

f. TWEA: Gera o bit de ACK. 1 – ACK, 0 – NACK;

g. TWINT: Flag da interrupção do periférico;

No caso do projeto precisa-se fazer a habilitação obrigatória do TWEN e os outros bits ficaram a caráter dos casos que ocorrerem na implementação, como por exemplo, se o usuário quiser iniciar uma comunicação I^2C(TWI) com um dispositivo, necessariamente ele tera que enviar um bit de start, então os bits a serem setados são os seguintes: TWINT, TWEN e TWSTA.

3.2.3 TWSR - TWI Status Register

Como o próprio nome já sugere, possui bits de status para a verificação das condições de comunicação além de 2 bits responsáveis pelo prescaler. Na FIGURA 17 podemos observar a representação desse registrador.

Figura 17 – TWSR

Bit	7	6	5	4	3	2	1	0	
(0xB9)	TWS7	TWS6	TWS5	TWS4	TWS3	–	TWPS1	TWPS0	TWSR
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	1	1	1	1	1	0	0	0	

Fonte: (ATMEL CORPORATION, 2015).

3.2.4 TWPS - TWI Prescaler Bits

O TWPS faz parte do cálculo da frequência, como adotamos a frequência de 100KHz, o TWPS deve assumir o valor de 1. Caso fosse necessário alterar o valor do *prescaler* deve-se seguir a FIGURA 18.

Figura 18 – TWPS

TWPS1	TWPS0	Prescaler Value
0	0	1
0	1	4
1	0	16
1	1	64

Fonte: (ATMEL CORPORATION, 2015).

3.2.5 TWDR - TWI Data Register

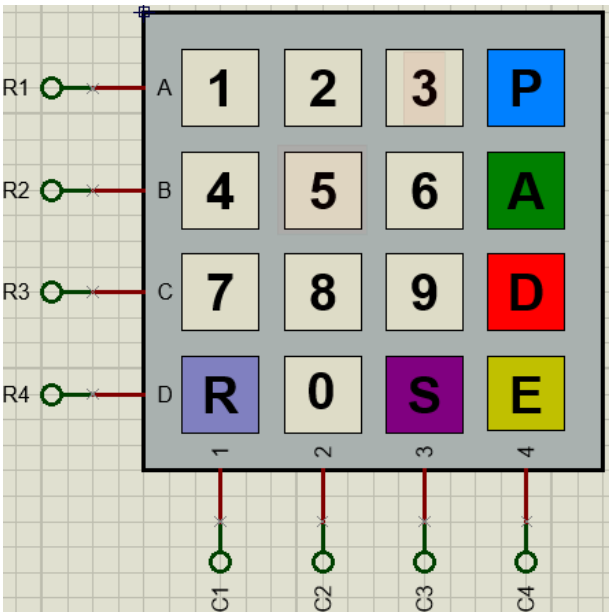
Responsável pelo envio e recepção dos dados.

3.3 *Keypad*

Em relação ao projeto em questão, o grupo sugere a utilização do componente "*Keypad - Smallcalc*", como também, o designer visto na Figura 19 deve ser respeitado

para evitar futuros problemas. É bom ressaltar que este componente está disponível no software “Proteus”.

Figura 19 – *Keypad* do Proteus



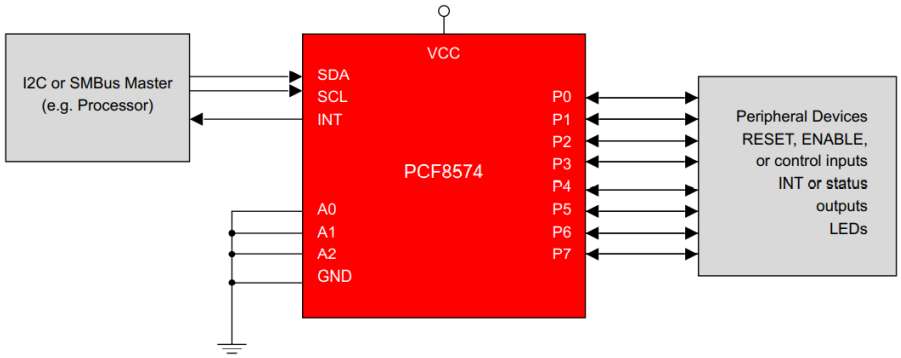
Fonte: Elaborado pelos autores.

Além disso, verifica-se que de R1 à R4 tratam-se das linhas da matriz. A nomenclatura baseou-se em linha que em inglês é “row”. E de C1 à C4 tratam-se das colunas da respectiva matriz.

3.4 Configurações do LCD (PCF8574 + LM016L)

O PCF8574 é um extensor I/O de 8-bits que utiliza o protocolo I^2C, dessa forma, é possível realizar toda a comunicação entre o display e o microcontrolador através deste componente. A frequência máxima suportada por este componente é de $100kHz$ e a configuração básica para utilização está exposta na Figura 20, onde o SCL será a entrada responsável por sincronizar o *clock* da comunicação entre o microcontrolador e o componente, a entrada SDA será para envio/recebimento de dados, a saída INT funciona como um sinal de *interrupt* e as entradas $A0 \sim A2$ funcionam como customizadores de endereçamento do dispositivo.

Figura 20 – Configuração básica para PCF8574



Fonte: Datasheet do PCF8574 (TEXAS INSTRUMENTS, 2015).

Como iremos utilizar dois PCF8574 é necessário customizar o endereçamento de cada um para que não haja conflito de informações, como já foi falado anteriormente, isso se dá pelas entradas A0, A1 e A2 do componente, para o caso do PCF8574 que será utilizado no LCD, todas essas entradas devem ser conectadas ao terra, a Figura 21 mostra as diferentes possibilidades de endereço.

Figura 21 – Endereços do PCF8574

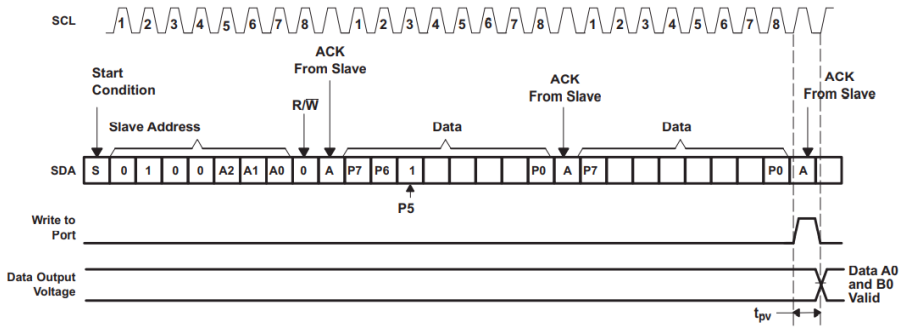
INPUTS			I^2C BUS SLAVE 8-BIT READ ADDRESS	I^2C BUS SLAVE 8-BIT WRITE ADDRESS
A2	A1	A0		
L	L	L	65 (decimal), 41 (hexadecimal)	64 (decimal), 40 (hexadecimal)
L	L	H	67 (decimal), 43 (hexadecimal)	66 (decimal), 42 (hexadecimal)
L	H	L	69 (decimal), 45 (hexadecimal)	68 (decimal), 44 (hexadecimal)
L	H	H	71 (decimal), 47 (hexadecimal)	70 (decimal), 46 (hexadecimal)
H	L	L	73 (decimal), 49 (hexadecimal)	72 (decimal), 48 (hexadecimal)
H	L	H	75 (decimal), 4B (hexadecimal)	74 (decimal), 4A (hexadecimal)
H	H	L	77 (decimal), 4D (hexadecimal)	76 (decimal), 4C (hexadecimal)
H	H	H	79 (decimal), 4F (hexadecimal)	78 (decimal), 4E (hexadecimal)

Fonte: Datasheet do PCF8574 (TEXAS INSTRUMENTS, 2015).

Para realizar a comunicação com o componente, após o *START* que inicia a comunicação I2C, deve-se enviar o endereço do dispositivo, sendo o bit menos significativo responsável por identificar se iremos enviar ou receber dados, como no caso do display só iremos enviar dados, este bit deve ser sempre ”0”, dessa forma, para endereçar o componente que irá se comunicar com o display, deve ser enviado sempre ”01000000” nesse

estágio da comunicação. A comunicação de escrita funciona da maneira que está exposta na Figura 22

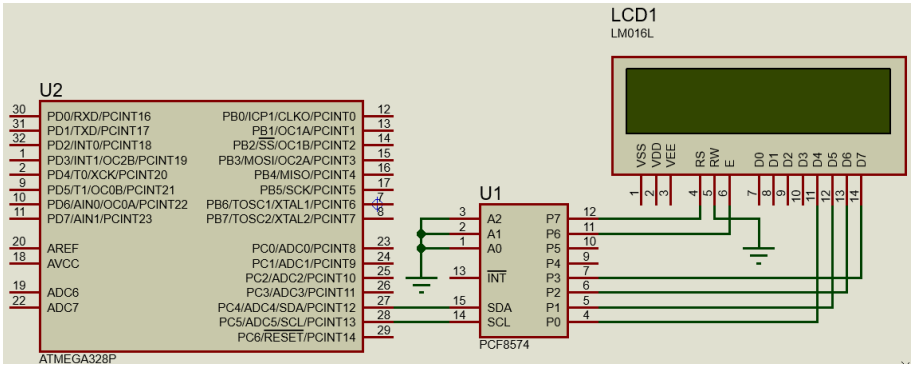
Figura 22 – Modo de escrita do PCF8574



Fonte: Datasheet do PCF8574 (TEXAS INSTRUMENTS, 2015).

Com a comunicação já iniciada com o dispositivo, deve-se enviar os dados, nesse estágio é necessário primeiramente realizar a configuração do display, onde deve ser setado para o modo de 4 bits, a partir dessa configuração inicial, pode ser enviado os dados de qual caractere deve ser apresentado para 4 pinos da saída, como sobram 4 pinos, pode ser utilizado para envio do RS e do EN do LCD. A configuração e pinagens da implementação deve ser algo parecido com oque mostra a Figura 23.

Figura 23 – Montagem para comunicação entre microcontrolador e display LCD



Fonte: Elaborado pelos autores.

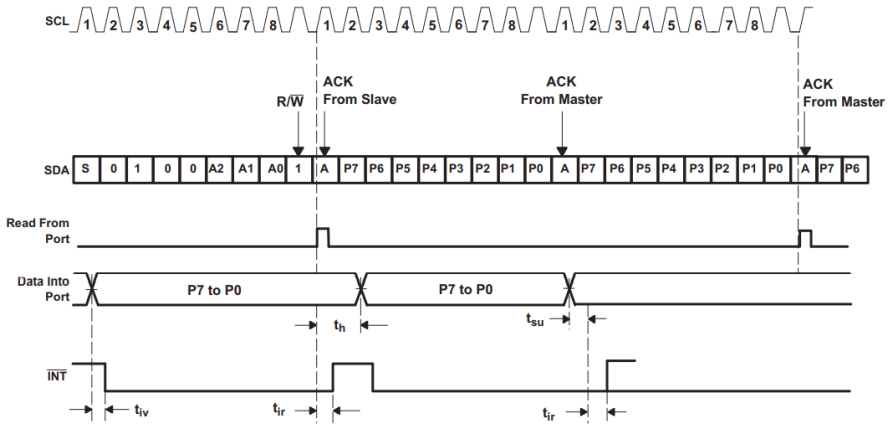
3.5 Sensores via PCF8574

De maneira semelhante a comunicação com o LCD 16x2, os sensores se comunicarão com o microcontrolador através do componente PCF8574, isso possibilita a utilização de apenas dois pinos do microcontrolador para ambas as comunicações. Entretanto, como os sensores serão apenas sinais de entrada do nosso microcontrolador, o bit menos significativo responsável por indicar se queremos ler ou escrever um dado dessa vez deve ser definido

como ”1” sempre que quisermos fazer uma leitura dos 8 sensores disponíveis no projeto. Ao contrário do display LCD, onde não queríamos trabalhar com interrupções pois só iríamos enviar dados, dessa vez o pino responsável por gerar um sinal de interrupção deve ser utilizado, isso por que toda vez que um sensor identificar algo, o alarme deve tocar imediatamente.

Ainda em relação aos pinos do PCF8574, dessa vez o pino A0 deve ser conectado em nível lógico alto, para que o endereçamento do componente responsável pela comunicação entre os sensores e o microcontrolador seja diferente do responsável pela comunicação com o display LCD. Desta forma, após iniciar a condição de *START* deve ser passado endereço ”01000011” para estabelecer a comunicação com este componente. A comunicação de leitura deste componente é demonstrada na Figura 24.

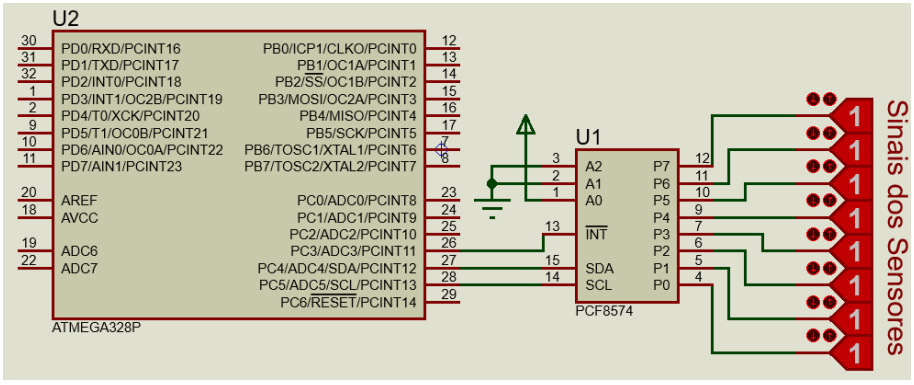
Figura 24 – Modo de leitura do PCF8574



Fonte: Datasheet do PCF8574 (TEXAS INSTRUMENTS, 2015).

Diante das configurações descritas, a montagem final da implementação deve ser algo parecido com o que mostra a Figura 25.

Figura 25 – Montagem para comunicação entre sensores e microcontrolador



Fonte: Elaborado pelos autores.

3.6 Configurações do RTC - DS1307

O DS1307 foi escolhido como componente para solucionar a problemática da informação de data e hora real, dados que serão utilizados para gerar o log da central do alarme. Este componente utiliza o protocolo de comunicação I^2C, foi pensado em utilizar este componente para que fosse possível utilizar as mesmas funções criadas para a comunicação deste protocolo. O mapa de endereços deste componente pode ser visto na Figura 26.

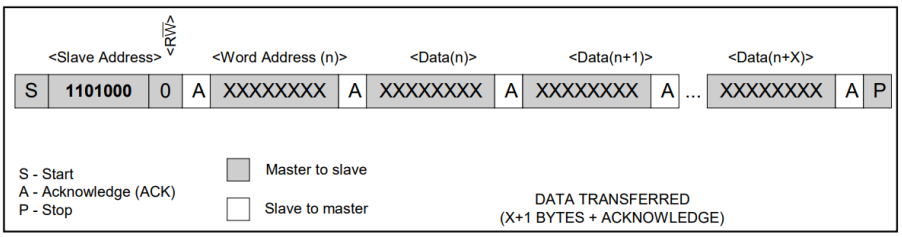
Figura 26 – Endereços do DS1307

ADDRESS	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	FUNCTION	RANGE
00h	CH	10 Seconds			Seconds				Seconds	00–59
01h	0	10 Minutes			Minutes				Minutes	00–59
02h	0	12	10 Hour	10 Hour	Hours				Hours	1–12 +AM/PM 00–23
		24	PM/ AM							
03h	0	0	0	0	0	DAY			Day	01–07
04h	0	0	10 Date		Date				Date	01–31
05h	0	0	0	10 Month	Month				Month	01–12
06h	10 Year				Year				Year	00–99
07h	OUT	0	0	SQWE	0	0	RS1	RS0	Control	—
08h–3Fh									RAM 56 x 8	00h–FFh

Fonte: Datasheet do DS1307 (MAXIM, 2015).

O componente DS1307 pode vir resetado ou pode ser necessário o ajuste da hora configurada para ele, desta forma é necessário acessar os registradores internos do componente, para isso ser possível, basta inicializar a condição de start, depois passar o endereço do componente, que nesse caso é definido como ”1101000” e o bit menos significativo sendo "0", indicando para o componente que haverá uma escrita nos registradores internos. Em seguida, deve ser passado o endereço de qual dos registradores disponíveis no componente eu quero escrever um dado, e no conjunto de 8 bits posterior, eu passo o dado que quero escrever. Desta maneira, pode ser alterado os valores de cada um dos registradores disponíveis no componente DS1307. A Figura 27 mostra como funciona o modo de escrita do DS1307.

Figura 27 – Modo de escrita do DS1307

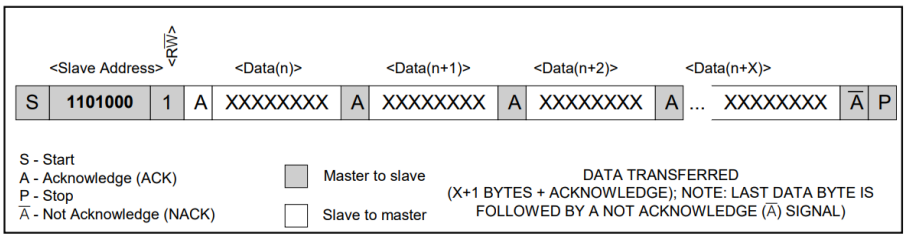


Fonte: Datasheet do DS1307 (MAXIM, 2015).

Com o horário e data já configurados com os dados corretos, basta inicializar a

comunicação passando o bit menos significativo como ”1”, indicando para o componente que o microcontrolador irá apenas ler os dados de cada registrador do componente, desta forma, o DS1307 irá mandar de volta os dados armazenados em cada um dos registradores internos. A Figura 28 mostra o modo de leitura do DS1307.

Figura 28 – Modo de leitura do DS1307



Fonte: Datasheet do DS1307 (MAXIM, 2015).

3.7 Configurações da UART para Datalogger

Para fazer uso do protocolo UART, deve-se configurá-lo de acordo com o clock do microprocessador e com as especificações do componente com o qual irá se comunicar. Considerando as especificações do *"Virtual Terminal"* do Proteus, este possui as seguintes definições:

a. Baud Rate: 9600 bps;

b. Data bits: 8;

c. Parity: Não possui;

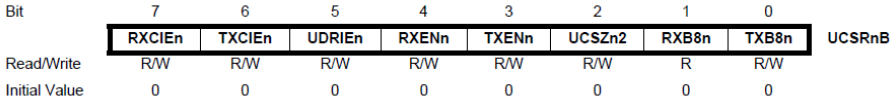
d. Stop Bits: 1;

Com base nestas informações, se pode configurar o protocolo UART através de seus registradores UBRR0, UCSR0B e UCSR0C. UBRR0 é um registrador de 16 bits que deve receber um valor relacionado ao *baud rate* da comunicação, e que também depende do *clock* do sistema. O datasheet do ATmega328P possui este valor tabelado, que é 103 para *baud rate* de 9600bps e clock de 16MHz. Mas também possui uma equação para encontrar esse valor.

$$UBRR0 = \frac{f_{OSC}}{16BAUD} - 1 = \frac{16M}{16 \times 9600} - 1 \cong 103 \tag{3.1}$$

Para o registrador UCSR0B, visível na Figura 29, deve-se setar somente o bit TXEN0 (bit 3), uma vez que a central somente fará transmissão. Os demais bits são zerados.

Figura 29 – UCSR0B

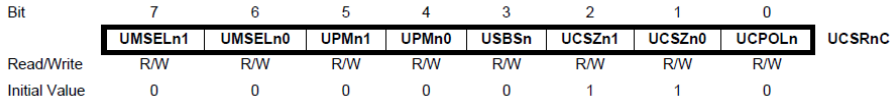


Bit	7	6	5	4	3	2	1	0	
	RXCIEn	TXCIEn	UDRIEn	RXENn	TXENn	UCSZn2	RXB8n	TXB8n	UCSRnB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fonte: Elaborado pelos autores.

Para o registrador UCSR0C, visível na Figura 30, deve-se setar somente os bits UCSZ01 e UCSZ00, definindo 8 bits de dados. Os demais bits são zerados, definindo o modo assíncrono, paridade desabilitada, 1 bit para stop bit.

Figura 30 – UCSR0C



Bit	7	6	5	4	3	2	1	0	
	UMSELn1	UMSELn0	UPMn1	UPMn0	USBSn	UCSZn1	UCSZn0	UCPOLn	UCSRnC
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	1	1	0	

Fonte: Elaborado pelos autores.

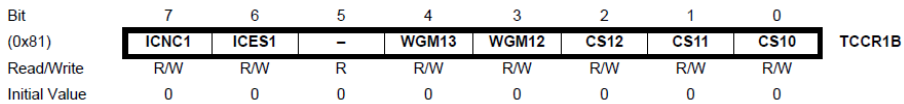
É recomendado se certificar de que o *"Virtual Terminal"* possua as mesmas especificações apresentadas acima.

3.8 Configurações do Timer/Counter1

Como é necessário fazer contagens a cada segundo, as configurações do Timer1 serão para interromper o programa principal a cada segundo passado. Para isso, deve-se configurar os registradores OCR1A, TCCR1B e TIMSK1.

No registrador TCCR1B, visível na Figura 31, serão definidos o *pre-scale* e o modo de operação. Setando o WGM12 (bit 3), define o Timer1 no modo CTC com máximo em OCR1A, setando o CS12 (bit 2), define o *pre-scale* para 256.

Figura 31 – TCCR1B



Bit	7	6	5	4	3	2	1	0	
(0x81)	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fonte: Elaborado pelos autores.

O registrador OCR1A deve receber o valor que corresponde a 1 segundo. Para definí-lo utiliza-se a seguinte equação. sendo f_{clk} o clock do sistema, e t o tempo desejado.

$$OCIE1A = \frac{f_{clk} \times t}{prescale} = \frac{16M \times 1}{256} = 62500 \tag{3.2}$$

Por fim, como o OCR1A será definido como máximo, e atingir o máximo significa que se passou 1 segundo, então deve-se habilitar a interrupção pelo *match* com o comparador A. Para isso, o bit OCIE1A (bit 1) deve ser setado.

Figura 32 – TIMSK1

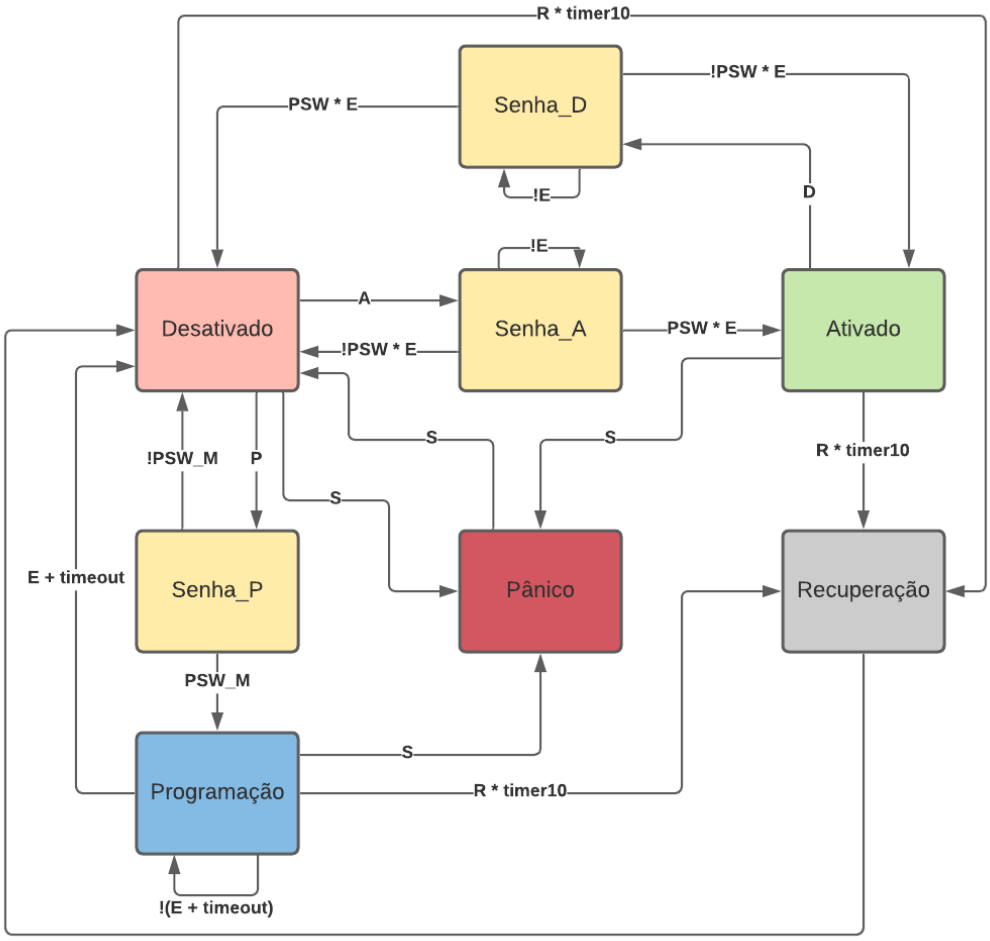
Bit	7	6	5	4	3	2	1	0	
(0x6F)	–	–	ICIE1	–	–	OCIE1B	OCIE1A	TOIE1	TIMSK1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fonte: Elaborado pelos autores.

3.9 Fluxograma geral da central de alarme

A priori foi feito uma fluxograma que explica melhor os modos de operação do sistema de alarme. Para isso foram separados em blocos e cada bloco refere-se a um modo de operação, são eles: “Desativado, Ativado, Pânico, Recuperação, Programação, Senha_A, Senha_D e Senha_P” conforme pode ser visto na Figura 33.

Figura 33 – Fluxograma geral da central



Fonte: Autores.

A central de alarme deve iniciar seu funcionamento no modo "Desativado". Para realizar a ativação da central, estando em "Desativado", deve-se pressionar o botão *A*, em seguida o sistema irá solicitar uma das senhas cadastradas, caso a senha esteja certa e o botão *E* tenha sido pressionado, a central entrará no modo "Ativado". O processo de desativação é similar ao anterior, porém, inicia-se no estado "Ativado" e ao fim do processo a central estará no modo "Desativado".

Para acessar o modo "Pânico" o usuário deve pressionar o botão *S*, estando desativado, ativado ou em programação, neste estado a sirene deve se manter ativa até que o botão *S* seja ativado novamente, o sistema então sai do modo "Pânico"e retorna para o estado "Desativado".

O processo de recuperação da central de alarme, assim como caso anterior, pode ser solicitado enquanto o sistema está desativado, ativado ou em programação. Para iniciar o procedimento é necessário que o usuário mantenha o botão *R* pressionado por dez segundos.

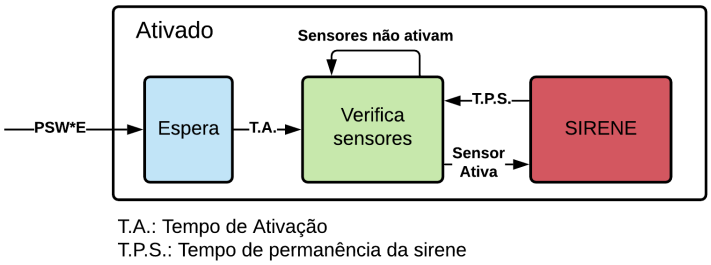
Nos momentos em que a central espera receber uma senha, seja ela a mestre ou não, caso o usuário insira uma senha que não foi cadastrada, a central deve retornar para o processo que se encontrava antes da solicitação de senha.

Para iniciar o modo de programação da central, estando em "Desativado", deve-se pressionar o botão *P*, a central solicitará a senha mestre, caso o usuário entre com a senha correta, se dará inicio ao processo de programação.

3.9.1 Ativado

Na Figura 34, observa-se os detalhes de funcionamento do modo ativado. O estado “Espera” irá atrasar a ativação fazendo com que o programa permaneça nele até o temporizador de ativação chegar ao tempo determinado, passando para o estado seguinte. No estado “Verifica sensores”, o programa continuamente irá verificar os sensores. Se algum sensor for ativado, o programa irá para o estado “SIRENE”, que acionará as sirenes e só sairá desse estado quando passar o tempo de permanência da sirene ou o usuário pressionar o botão D e entrar com uma senha correta (Esta ação não está representada na Figura 34, mas sim na Figura 33).

Figura 34 – Fluxo interno do modo Ativado



Fonte: Autores.

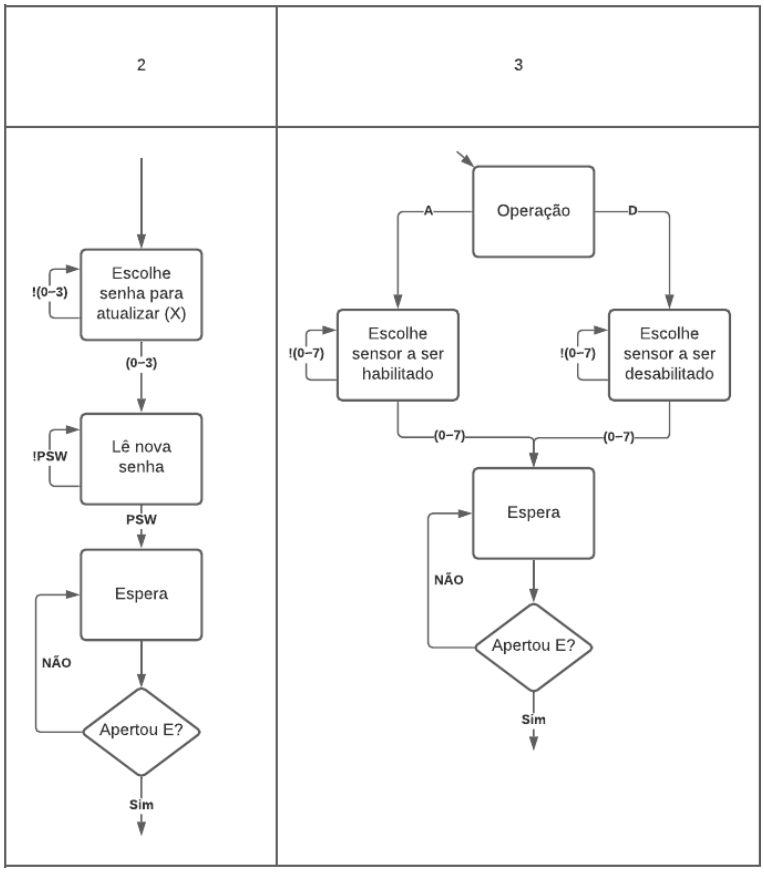
3.10 Fluxograma de programação

Durante a programação da central, dependendo da funcionalidade escolhida, um fluxo diferente deve ser seguido. A Figura 35 indica os fluxos que devem ser seguidos nas funcionalidades 2 e 3 da central.

Ao escolher a função 2, o usuário deseja iniciar o processo de alteração de senha, para isso é necessário indicar qual a senha que deve ser alterada, podendo escolher um valor entre 0 e 3, na sequência deve-se esperar até que a nova senha seja inserida, o processo finaliza apenas após a ativação do botão *E*.

Para a função 3, a central necessita que seja indicado qual sensor deve ser habilitado ou desabilitado, nas operações "A" e "D", respectivamente. Para finalizar a configuração é

Figura 35 – Fluxo das funcionalidades 2 e 3



Fonte: Autores.

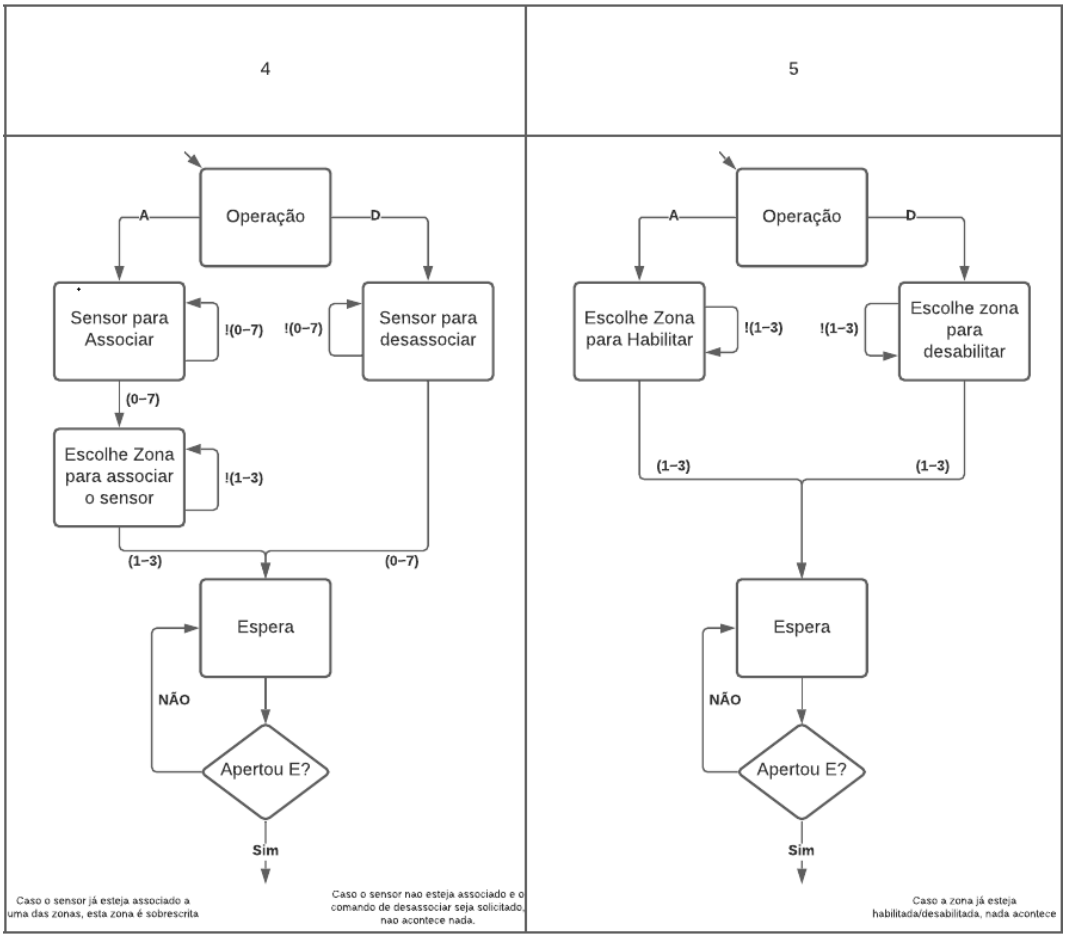
necessário que o botão E seja ativado.

A funcionalidade 4 é responsável por associar ou desassociar um sensor em uma das zonas existentes no projeto. Caso o usuário deseje associar um sensor, deve-se indicar qual sensor será utilizado na operação, na sequência o usuário deve entrar com o valor da zona que deseja aplicar o sensor. Se porventura, o sensor escolhido já esteja associado a uma zona, esta operação move o sensor da antiga zona para a nova escolhida, neste projeto um sensor não pode pertencer a mais de uma zona. Já para o caso de desassociação, é necessário indicar apenas o sensor que será removido de uma das zonas. Caso o sensor escolhido não esteja associado a uma zona, nada acontece. Assim como nos casos anteriores, a operação necessita que o botão E seja ativado para confirmar a configuração.

Ao escolher a função 5, o usuário deve escolher qual a zona será ativada ou desativada, confirmando com o botão E na sequência.

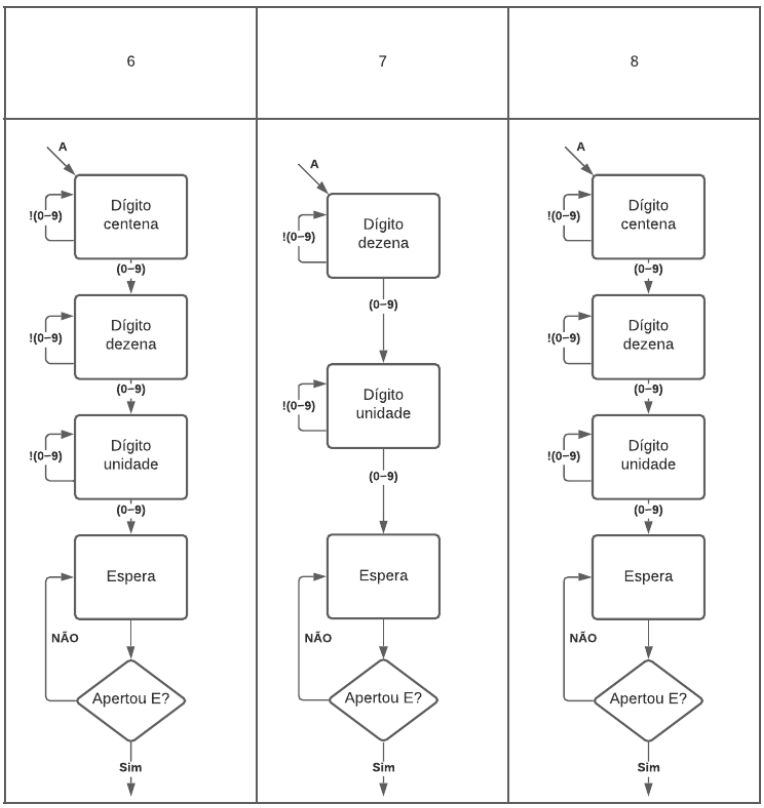
As funcionalidades 6, 7 e 8 são responsáveis por configurar temporizadores, seus fluxos serão semelhantes, mudando apenas a quantidade de dígitos lidos no caso do *timeout*. A figura 37 mostra o fluxograma das funcionalidades citadas.

Figura 36 – Fluxo das funcionalidades 4 e 5



Fonte: Autores.

Figura 37 – Fluxo das funcionalidades 6, 7 e 8

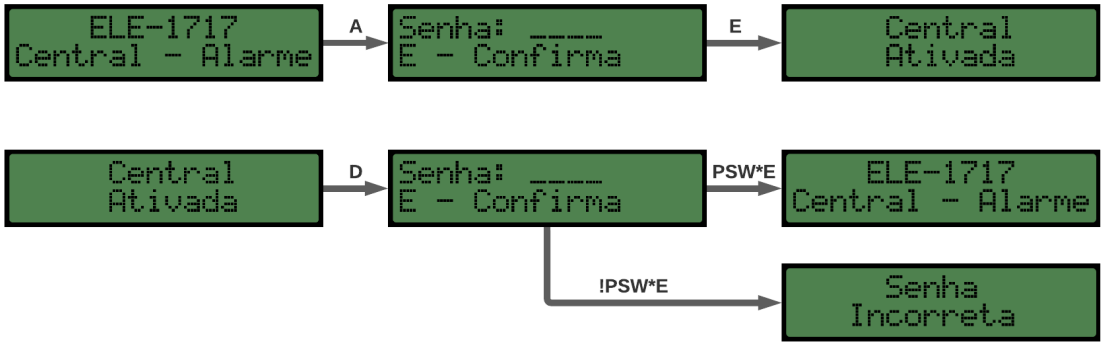


Fonte: Autores.

3.11 Fluxograma de telas

Nesta seção serão expostas as telas que a central de alarme deve apresentar durante seu funcionamento.

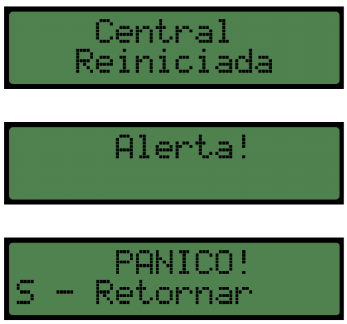
Figura 38 – Telas na ativação/desativação



Fonte: Autores.

A figura 38 mostra as telas que devem aparecer durante os processos de ativação e desativação da central de alarme. Caso em algum momento o usuário entre com uma senha errada, o visor apresenta a mensagem "Senha Incorreta" antes de retornar para o estado em que se encontrava.

Figura 39 – Telas de Recuperação, sirene ativa e pânico

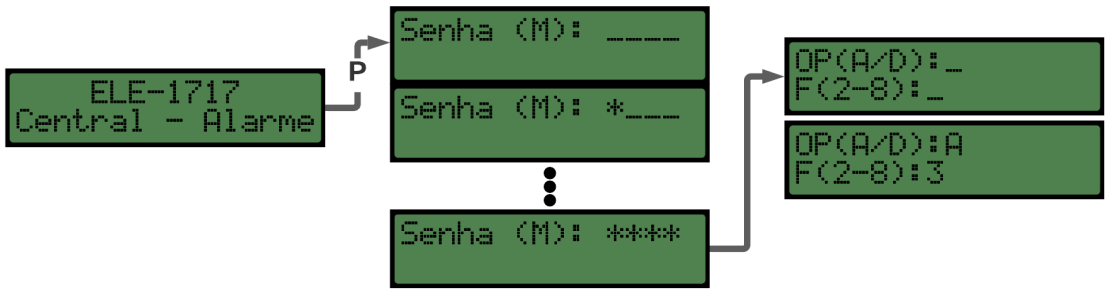


Fonte: Autores.

Quando o sistema for recuperado, antes de retornar ao repouso, a central apresenta a mensagem "Central Reiniciada". Durante o modo ativado, caso seja detectado movimento por algum dos sensores ativados, a central deve ativar a sirene e apresentar "Alerta!" no visor. No modo de pânico, a central apresenta o texto "PANICO!", seguido do botão que deve ser pressionado para deixar o modo.

Para acessar a programação, a central deve apresentar as telas da figura 40, nota-se que neste caso em específico, a senha para o acesso é a mestre. Sempre que for necessário receber uma senha do usuário, o display deve ir substituindo as linhas por asteriscos, na

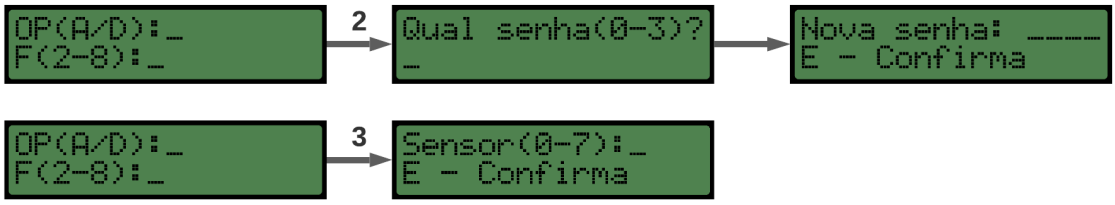
Figura 40 – Telas de transição de desativado para o modo programação



Fonte: Autores.

medida que os valores são inseridos, já na leitura dos demais dados, deve-se substituir os traços pelos números ou letra.

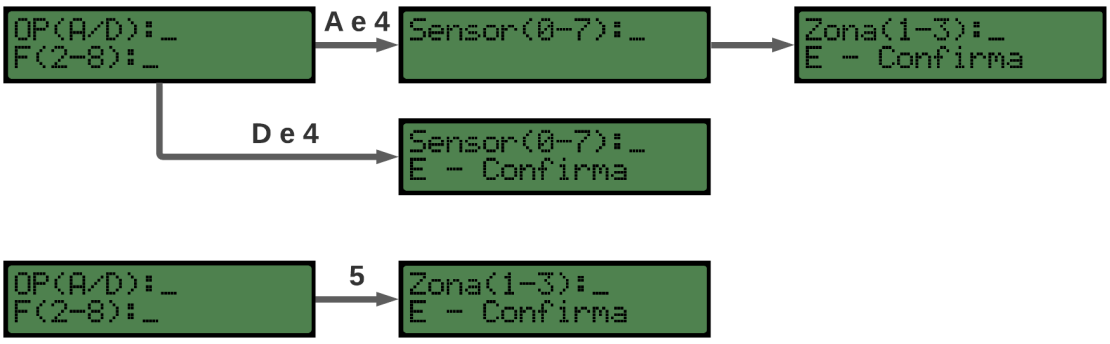
Figura 41 – Telas das funções 2 e 3



Fonte: Autores.

A figura 41 mostra a sequência de telas para as funções 2 e 3, para o caso 2 é feita a leitura da senha que será modificada e depois o nova chave, para o caso 3 apenas o valor do sensor é passado.

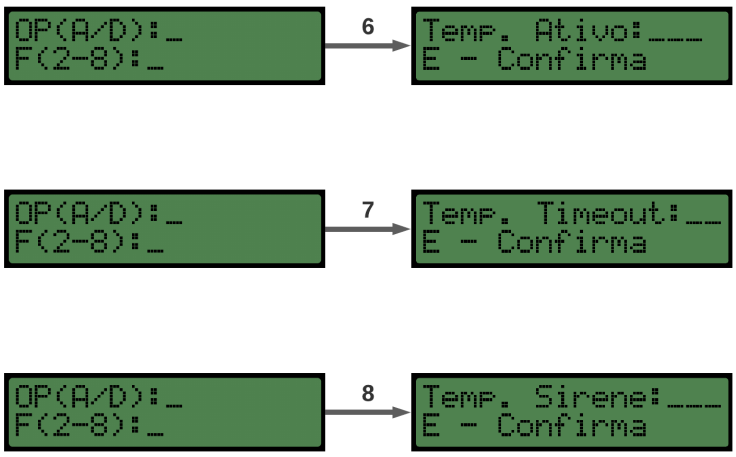
Figura 42 – Telas das funções 4 e 5



Fonte: Autores.

Na funcionalidade 4 existe a possibilidade de pedir o sensor e a zona de interesse ou apenas o sensor, dependendo da operação escolhida. Já na função 5, apenas a zona deve ser inserida, antes da confirmação. A figura 42 mostra as telas que devem aparecer nestas funções.

Figura 43 – Telas das funções 6, 7 e 8



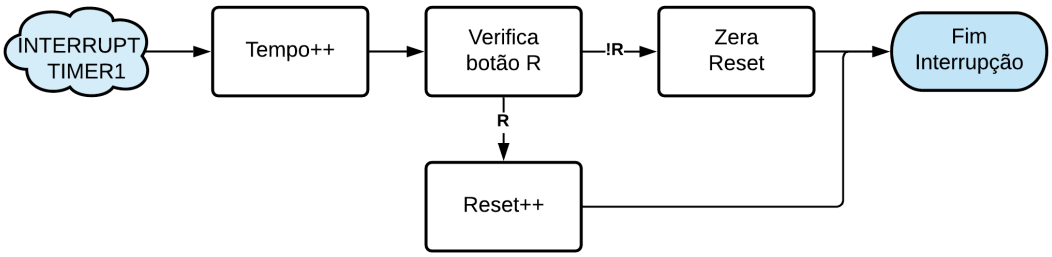
Fonte: Autores.

Por fim, nas funções 6, 7 e 8 é necessário entrar com o valor do temporizador de ativação, *timeout* e tempo de permanência da sirene, respectivamente. As telas estão apresentadas na figura 43.

3.12 Fluxograma de interrupção

Para implementar os temporizadores da central, faz-se necessário utilizar do Timer/-Counter1 para fazer contagens de 1 segundo. Para que o Timer1 não influencie na execução do código principal enquanto realiza a contagem dos temporizadores, é possível configurar o Timer1 para gerar uma interrupção a cada 1 segundo. Dentro da interrupção, é possível executar uma rotina que será responsável por realizar a contagem dos 3 temporizadores e do botão de restauração. O fluxograma dessa rotina está Figura 44.

Figura 44 – Rotina da interrupção



Fonte: Autores.

Sabendo que os três temporizadores (temporizador de ativação, *timeout* e temporizador de permanência da sirene) sempre funcionarão em momentos distintos, pode-se usar

a mesma variável de contagem para os três. Essa contagem estará ocorrendo no estado "Tempo++". Vale lembrar que é necessário zerar a variável antes da contagem.

Toda vez que o programa chega no estado "Verifica botão R", é verificado se o botão R está sendo pressionado. Se estiver, o contador do *reset* incrementa ("Reset++"), se não estiver sendo pressionado, o contador do *reset* zera. Quando este contador chega a 10, o encaminhamento para o modo de "Recuperação" deve ser feito no estado atual do programa principal. Alguma lógica é necessária para não permitir que a recuperação ocorra enquanto a sirene estiver ligada no modo "Ativado".

Referências

ATMEL CORPORATION. *ATMEL 8-BIT MICROCONTROLLER WITH 4/8/16/32KBYTES IN-SYSTEM PROGRAMMABLE FLASH DATASHEET*. [S.l.], 2015. Rev. 8271A.

JAYANT R. *4x4 Matrix Keypad Interfacing with 8051 Microcontroller*. [S.l.], 2015. Disponível em: <https://circuitdigest.com/microcontroller-projects/keypad-interfacing-with-8051-microcontroller>.

KOVALHUK, G. *Microcontroladores 2 - EL08D Turma M12*. [S.l.], 2020. Disponível em: <http://paginapessoal.utfpr.edu.br/kovalhuk/disciplinas-1/el08d-microcontroladores-2/atmega328/gk_2020_01_Aula07-twi.pdf/at_download/file>.

MAXIM. *DS1307 64 x 8, Serial, I2C Real-Time Clock*. [S.l.], 2015. Disponível em: <https://datasheets.maximintegrated.com/en/ds/DS1307.pdf>.

PROBLEMA 06. *sistemas digitais - Problema 06 - Projeto*. [S.l.], 2021.

RECH R. A. *Aula 10: Comunicação Serial com o PC com o perfiérico USART*. [S.l.], 2020. Disponível em: <https://www.youtube.com/watch?v=bFs6_cfkxVI&ab_channel=ProfessorVargasp>.

TEXAS INSTRUMENTS. *PCF8574 Remote 8-Bit I/O Expander for I2C Bus*. [S.l.], 2015. Disponível em: <https://circuitdigest.com/microcontroller-projects/keypad-interfacing-with-8051-microcontroller>.

VAHID, F. *Sistemas Digitais: Projeto, Otimização e HDLs*. [S.l.]: Artmed Bookman, 2008.

VARGAS, L. A. *ATMEGA328p os fusebit necessários para configurá-lo*. 2020. Acessado em: 23 ago. 2021. Disponível em: <https://www.youtube.com/watch?v=bFs6_cfkxVI&ab_channel=ProfessorVargasp>.

WANT, R.; SCHILIT, B. N.; JENSON, S. Enabling the internet of things. *Computer*, IEEE, v. 48, n. 1, p. 28–35, 2015.

ANEXO A – Relato semanal

Líder: Lucas Augusto Maciel da Silva

A.1 Equipe

Tabela 1 – Identificação da equipe

Função no grupo	**Nome completo do aluno**
Redator	Rodrigo de Lima Santana
Debatedor	João Matheus Bernardo Resende
Videomaker	Marcelo Ferreira Mota Júnior
Auxiliar	Isaac de Lyra Junior

Fonte: Produzido pelos autores.

A.2 Defina o problema

O problema consiste em projetar uma central digital de alarme utilizando um circuito baseado em um uC AVR, mais precisamente o ATMega328p. Esta central deve apresentar um display e um teclado matricial 4x4, voltados para o manuseio da central pelos usuários, também possui 5 leds de *status*, 8 sensores de presença, como entradas, um canal de saída para a sirene e um canal de comunicação UART.

A central contém 5 modos de operação, sendo eles o ativado, desativado, programação, recuperação e pânico, destes, a recuperação e o pânico podem ser acessados a partir dos 3 outros modos, dependendo apenas que a condição para a entrada no estado seja satisfeita. Durante o seu funcionamento, a central solicita que o usuário entre com uma das senhas pré-cadastradas, onde o limite de senhas que podem ser armazenadas é de 4, sendo uma delas a senha mestre. Já no modo de programação, a central apresenta 7 funcionalidades distintas.

A.3 Registro de *brainstorming*

No primeiro encontro, a equipe fez a divisão dos cargos de cada componente, também foi decidido que o dia seguinte seria destinado para um estudo inicial do problema proposto.

No segundo encontro demos início a montagem dos fluxogramas do projeto, onde foi feito o fluxograma geral e os fluxos de duas funções da programação. No fim do encontro,

as demais funcionalidades foram divididas entre os componentes, na tentativa de acelerar o processo.

O quarto encontro foi destinado para a finalização dos fluxos restantes e algumas definições de projeto necessárias, como os componentes a serem utilizados, protocolos de comunicação viáveis para o projeto, dentre outros. Ao fim do dia, foi destinado que metade do grupo iria estudar o protocolo de comunicação I^2C e a outra metade o protocolo UART, ambos utilizados no projeto.

No quinto encontro foram feitas as telas que a central de alarme deve possuir, bem como as situações em que cada uma deve ser apresentada, foram definidas as configurações necessárias do microcontrolador para a aplicação dos protocolos de comunicação escolhidos e se deu início ao processo de criação do relatório.

Os dois últimos encontros foram voltados para a finalização das tarefas restantes, que foram a finalização do relatório e elaboração do vídeo semanal.

A.4 Pontos-chave

Os pontos-chave do projeto foram: a escolha do protocolo I^2C para a comunicação com o RTC, LCD e sensores, com o objetivo de reduzir a quantidade de portas necessárias, visto que a demanda de portas era superior ao que se tinha disponível; a escolha das telas que a central possui, já que se trata de uma parte importante no manuseio do sistema.

A.5 Questões de pesquisa

- Protocolo UART;
- Protocolo TWI (I^2C);
- Teclado matricial;
- RTC;
- LCD;

A.6 Planejamento da pesquisa

Para os protocolos, a pesquisa foi feita de forma paralela, onde metade do grupo estudou um dos protocolos e a outra metade o outro. Quanto aos demais tópicos de pesquisa, foram realizados em conjunto durante as reuniões da semana, onde foram consultados os *datasheets* dos componentes escolhidos e alguns exemplos de aplicação na internet.