

**UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE CENTRO DE TECNOLOGIA**

**CURSO DE ENGENHARIA MECATRÔNICA SISTEMAS DIGITAIS**

**IMPLEMENTAÇÃO DE CIRCUITO BASEADO EM MICROCONTROLADOR AVR(ATMega 328P)**

Humberto Gama de Carvalho Neto - 20190001810 Oswaldo de Oliveira Vicente - 20200001097 Atyson Jaime de Sousa Martins - 20190153956

Alax Gabriel Cavalcante Lima de Oliveira - 20190001785

HUMBERTO GAMA DE CARVALH NETO

OSWALDO DE OLIVEIRA VICENTE

ATYSON JAIME DE SOUSA MARTINS

ALAX GABRIEL CAVALCANTE LIMA DE OLIVEIRA

**IMPLEMENTAÇÃO DE CIRCUITO BASEADO EM MICROCONTROLADOR AVR(ATMega 328P)**

Sexto Relatório apresentado à disciplina de Sistemas Digitais, correspondente a sexta semana da disciplina do semestre 2020.6 do curso de Engenharia Mecatrô- nica da UFRN Referente à implementação de um circuito de um cofre baseado em microcontrolador AVR(ATMega 328P).

Professor: Samaherni Morais Dias

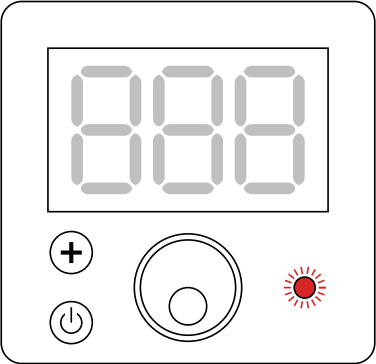
**Sumário**

1. [Introdução 1](#_TOC_250018)
   1. [Apresentação do Sistema 1](#_TOC_250017)
   2. [Etapas do Projeto 1](#_TOC_250016)
2. [Desenvolvimento 2](#_TOC_250015)
   1. [Componentes do Circuito 2](#_TOC_250014)
      1. [Microcontrolador AVR ATMega328P 2](#_TOC_250013)
   2. [Periféricos 2](#_TOC_250012)
      1. [Display de 3 Dígitos 3](#_TOC_250011)
      2. Decpdificador 7 segmentos...................................................................3
      3. [Potenciômetro 4](#_TOC_250009)
      4. [LED RGB 4](#_TOC_250008)
      5. [Push Button 5](#_TOC_250007)
   3. [Montagem dos cricuitos no Proteus 5](#_TOC_250006)
   4. [Fluxograma 6](#_TOC_250005)
   5. [Máquina de estados 8](#_TOC_250004)
   6. [Clock 10](#_TOC_250003)
   7. [Delay 10](#_TOC_250002)
   8. [Converso ADC 11](#_TOC_250001)
   9. Amostragem no display....................................................................................14
3. [Conclusão 13](#_TOC_250000)

# Introdução

## Apresentação do Sistema

O presente relatório tem o objetivo de descrever um sistema digital, assim como sua implementação, que controla o funcionamento de um cofre baseado no microcontrolador AVR (ATMega 328P). Para a abertura dele é necessário inserir uma senha que é composta por três valores variando de 0 a 999. Esses valores são inseridos através de um botão rotativo e um botão de confirmação. O sistema ainda possui um botão para iniciar/cancelar o processo de inserção de senha e um LED auxiliando o usuário. A interface do sistema é demonstrada na figura 1.

Figura 1: Interface do sistema

Fonte: Disponibilizado pelo professor

O processo de abertura do cofre segue os seguintes passos: primeiro, deve-se pressi- onar o botão *Power* para iniciar, ajustar o valor desejado no botão rotativo e então pressio- nar o botão *Add*. O processo deve ser repetido três vezes. Antes de iniciar este processo o LED deve estar na cor vermelha, na fase de ajuste do valor passado a cor deve estar azul, na fase de confirmação do valor a cor deve permanecer laranja por 0,5 segundos. Ao fim deste processo, caso a senha esteja correta a cor será verde sinalizando a abertura do cofre, caso contrário sua cor voltará para o vermelho do início, podendo repetir o processo.

Neste projeto, a senha correta será definida previamente via software e não poderá ser alterada pelo usuário.

## Etapas do Projeto

Os componentes eletrônicos necessários para a implementação do circuito são: microcontrolador AVR ATMega 328P, 2 pushbuttons, 1 potenciômetro, 3 displays de 7 segmentos, 1 Led RGB, 2 capacitores, 1 transistor, 1 CI 74HC14, 3 CI BCD 7446, 22 portas NOT e 28 resistores. Esses componentes são detalhados ao longo do relatório.

Outra fase importante no desenvolvimento do projeto foi a máquina de estados de alto nível que é responsável por toda a lógica do comportamento do sistema. Por fim temos a etapa de implementação por meio de instruções passadas ao microcontrolador via linguagem Assembly. Utilizamos o Proteus para fazer os circuitos e simular o código.

# Desenvolvimento

## Componentes do Circuito

### Microcontrolador AVR ATMega328P

O principal componente do sistema é o AVR ATMega 328P, esse microcontrolador de 8 bits combina alta performance com baixo consumo de energia, baseado em arqui- tetura RISC avançada, ele possui 131 instruções usando 32 registradores de propósito geral. Esse microcontrolador possui 28 pinos entre eles estão I/Os digitais e analógicos com funcionalidades de PWM (Modulção por Largura de Pulso, do inglês Pulse Width Modulation) e ADC (Conversor Analógico-Digital, também do inglês Analog-to-Digital Converter).

Para a implementação do circuito serão utilizados 18 pinos I/O. Sendo 17 digitais e 1 analógico. Sendo os digitais: 12 pinos de saída para os BCDs, 3 pinos de saída para o LED RGB, 1 pino de entrada para o botão ligar, 1 pino de entrada para o botão add. Quanto ao analógico temos 1 pino de entrada para o potenciômetro.

O clock a ser trabalhado no projeto será o de 128Khz, mais a frente explicaremos melhor.

## Periféricos

Para o circuito do cofre foi proposto que usássemos os seguintes componentes para a inserção e verificação da senha, assim como operação do sistema como um todo.

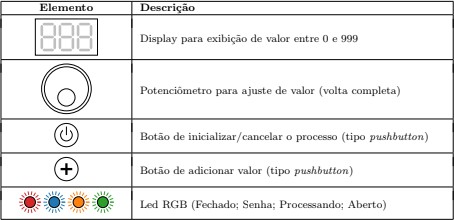


Figura 2: Componentes necessários para o funcionamento do circuito Fonte: Disponibilizado pelo professor

### 

### Displays de 7 segmentos

Inicialmente foi pensado em usar um método de multiplexação temporal baseada na persistência da visão. Esse método foi pensado porque o olho humano é incapaz de perceber alterações de um sinal luminoso, sem perceber nenhuma cintilação, com frequências acima de 48Hz, o que significa que se um LED piscar 48 vezes durante 1 segundo, ele aparenterá aos olhos humanos como ligado constantemente.

Porém, devido a dificuldade em fazer esse método em Assembly no tempo disponibilizado, tendo em vista que tinham outros paramêtros a serem definidos e implementados, foi decidido utilizar três displays separados sendo um para cada: unidade, dezena e centena.

### Decodificador 7 segmentos

Para transfromar os bits que saem do microcontrolador em uma exibição decimal nos displays é necessário um decodificador. O utilizado no projeto é o BCD 7446 para converter para 7 segmentos.

Como foram utlizados três displays para representar a unidade, dezena e centena foram necessários 12 bits de saídas do microcontrolador, sendo 4 para o BCD referente à centena, 4 para a dezena e 4 para a unidade.

### Potenciômetro

Para o ajuste dos valores da senha foi utilizado um potenciômetro linear rotativo com resistência de 10 kΩ. O potenciômetro possui três pinos VCC, GND, conectados respectivamente no microcontrolador e o terceiro pino onde é realizada a leitura da resistência conectado a uma porta ADC (PC0) do ATMega328P. Para converter o valor analogico para digital foi utilizado o conversor ADC do próprio microcontrolador.

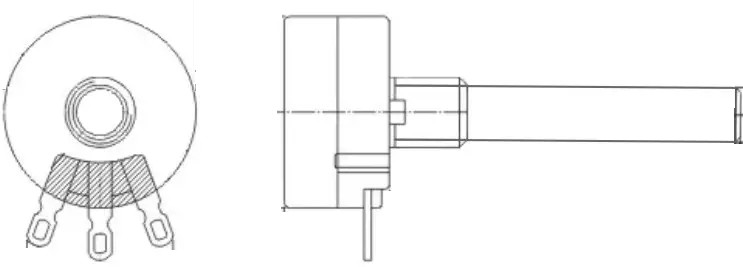


Figura 3: Potenciômetro linear rotativo

Fonte: autoria própria (Antigo grupo 1)

### LED RGB

Foi escolhido em projeto um led RGB de catodo comum que é conectado a três portas digitais do microcontrolador (PB5...PB7) para a composição de cores de acordo com o estado do cofre. As portas referentes ao RGB serão conectadas com a adição de resistores de 100 e 150 ohms.

Dessa forma é possível atingir as 4 cores necessárias para o projeto, com uma ressalva do laranja, não é possível obte-la com essa configuração, com isso, decidimos usar a cor amarela para representar o laranja, uma vez que é possível atingi-la combinando o R (Red/Vermelho) com o G (Green/Verde). Decidimos fazer dessa forma para facilitar o entendimento e a montagem do circuito não sendo necessário usar postas PWM.

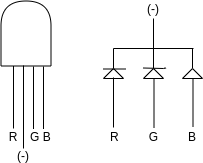


Figura 4: LED rgb de catodo comum

Fonte: autoriaprópria (Antigogrupo1)

### Push Button

O circuito dos pushbuttons usa uma técnica de debouncing por hardware, onde são utilizados dois resistores, um capacitor e uma porta inversora do CI 74HC14 (Schmitt Trigger).

Escolhemos essa técnica para evitar o efeito boucing que seria um problema de oscilação do sinal no momento em que a tecla é pressionada podendo acontecer do processador acreditar que a chave foi pressionada mais de uma vez em intervalos curtos.

De forma breve, nesse circuito o capacitor faz um papel de amortecedor do sinal, criando um tempo de atraso para o sinal necessário para carregar o capacitor. Com isso as alterações rápidas do sinal são filtradas evitando o problema do bouncing.

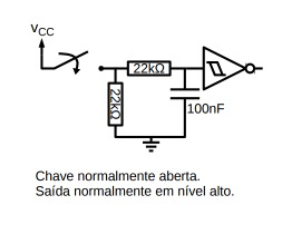


Figura 5: montagem de um pushButton normalmente aberto

Fonte: PDF: Aula 10B – Circuitos de Debounce por Nikolas Libert.

## 2.3 Montagem dos circutios no Proteus

Abaixo é mostrado detalhadamente os circuitos de todos os perfiféricos falados acima, relacionando com os resistores necessários, e suas respectivas entradas e saídas para com o ATMega.

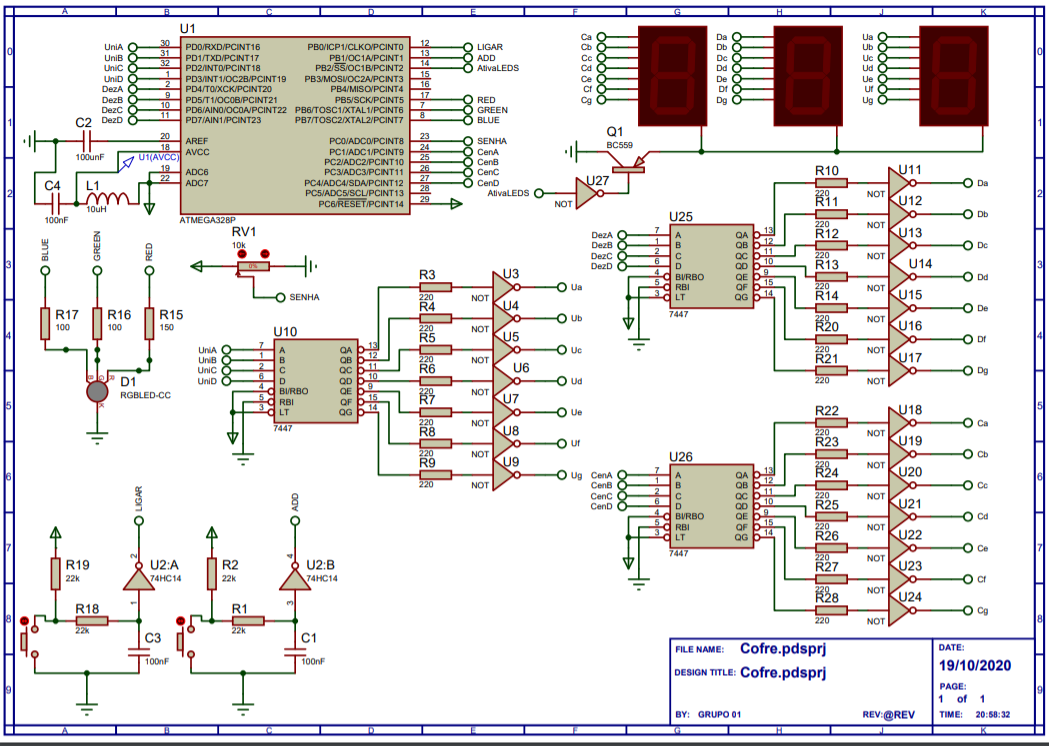


Figura 6: Montagem dos circuitos

Fonte: Autoria Própria

## Fluxograma

O fluxograma utilizado é o mesmo do grupo anterior, decidimos não modifica-lo pois aparenta está correto, a máquina de estados segue o seu funcionamento.

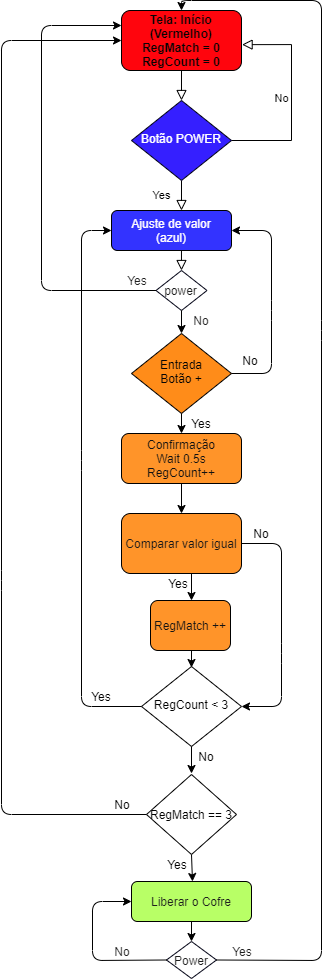


Figura 7: Fluxograma do funcionamento do projeto

Fonte: Autoria própria (Antigo grupo 1)

## Máquina de estados

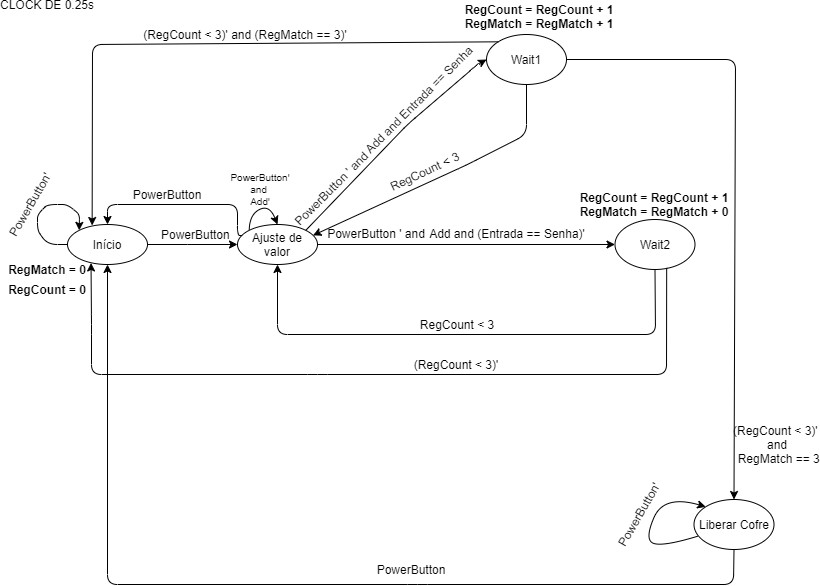
Anteriormente foram feitas três máquinas de estados para demonstrar o processo de contrução da máquina final, porém se tornou um pouco confuso, por isso, decidimos implemnetar apenas a última máquina, pois demonstra estar mais correta.

Figura 8: Máquina de estados

Fonte: Autoria própria (Antgo grupo 1)

Na máquina de estados os estados de Wait vão guardar o valor da quantidade erros ou acertos de acordo com a senha inserida. Para isso, foi adicionado um outro estado de Wait, (Wait2), no caso em que não haja o acerto da senha. Em ambos os casos, a distribuição de estados é a ideal, ou seja, depois do "ADD" da senha, o próximo estado será o de wait, e, por conseguinte, já voltará imediatamente para o estado de Ajuste de Valor, como deveria ser. Neste caso, a conexão entre o estado de Wait e Liberar Cofre será feita apenas para o Wait1, visto que essa possibilidade (liberar o cofre) nunca irá acontecer caso passe pelo estado de Wait2, que significa que pelo menos 1 das 3 entradas foram erradas.

**Estado "Início->** caso o PushButton de inicializar/cancelar o processo, o qual cha- mamos de PowerButton, esteja Desativado, irá permanecer no estado. Caso contrário, se for ativado, a máquina será direcionada para o estado de "Ajuste de valor". Além disso, nesse estado, serão zerados os registradores responsáveis por guardar a quantidade de va- lores colocados, bem como a quantidade de comparações de equidade entre as entradas e senhas.

**Estado "Ajuste de valor"** -> Caso seja pressionado o PowerButton, haverá o direci- onamento para o estado de Início. Caso o Powerbutton não seja pressionado e o botão de adicionar valor, o qual chamamos de ADD, também não seja pressionado, irá permanecer no mesmo estado. Na hipótese do PowerButton estar desativado e ADD ativado, depen- derá da comparação, como mostrada no fluxograma, entre a Entrada e a Senha. Caso igual, a direção será o Estado Wait1, caso diferente, Wait2.

**Estado "Wait2->** O estado Wait2 corresponde a hipótese em que a entrada do usuário não foi igual à senha registrada. Portanto, irá haver o incremento na contagem de vezes que o usuário adicionou um valor e, posteriormente, se essa contagem não tiver chegado em 3, retornará ao estado de "Ajuste de Valor". Caso tenha chegado em 3, retornará ao início.

**Estado "Wait1->** O estado Wait1 corresponde a hipótese em que a entrada do usuário foi igual à senha registrada. Portanto, haverá um incremento na quantidade de vezes em que a entrada foi igual à senha e, também, irá haver o incremento na contagem de vezes que o usuário adicionou um valor. Posteriormente, se a contagem de quantidade de inserções de valor não tiver chegado em 3, retornará ao estado de "Ajuste de Valor". Caso essa tenha chegado em 3 e o registrador que guarda a quantidade de acertos de senha não estiver em 3, retornará ao início. Por fim, se a quantidade de inserções de valor tenha chegado em 3 e também constar 3 equidades entre as entradas e senhas, o destino será o estado de "Liberar Cofre".

**Estado "Liberar Cofre->** Finalmente, após a tranca ser liberada, caso não haja o pressionamento do PowerButton, a tranca continuará no estado de "Liberar Cofre", ou seja, liberada. Caso o usuário deseje trancar, novamente, basta apertar o PowerButton e haverá o direcionamento para o estado de Início.

## Clock

Como dito anteriormente o clock utilizado foi o de 128KHz. Ele foi escolhido para ser utilizado por ser uma das opções de clock disnponíveis no ATMega 328P, sendo necessário apenas configurar os bits do vetor de seleção interno do microcontrolador: CKSEL[3..0] para “0011”. Seu período é menor que o tempo de espera necessário de 0,5s.

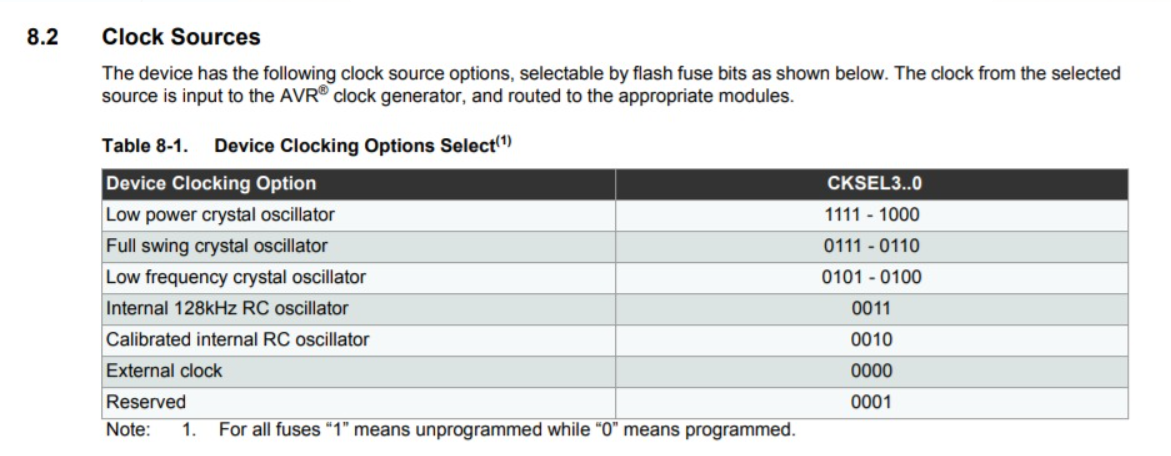


Figura 9: Fontes de clock

Fonte: Datasheet do AVR ATMega 328P

### Delay

Uma das imposições feitas pelo projeto se referiu ao tempo de espera de meio segundo entre a adição de uma senha e a mudança de cor do led para que outro número seja inserido. Para tal tarefa, pensamos em implementar da seguinte forma.

Foi definido via código uma sub-rotina com uma série de comandos "inúteis", cada qual gastando ciclos de clock. Assim como no modo de utilização dos timers, se faz necessário calcular quantos ciclos de clock geram uma parada de 0,5 segundos, e assim calcula-se quantas vezes o laço deve ser ativado.

No livro “AVR e Arduino Técnicas de Projeto” é mostrado um exemplo de código em assembly, para o microcontrolador estudado, de chamada de sub-rotina de atraso para manter um led aceso por determinado tempo.

DEC R3 //Decrementa o registrador R3, começa com o valor 0x00.

BRNE Atraso //Enquanto R3>0 fica decrementando R3, desvio para atraso.

DEC R2 //Decrementa R2, começa com o valor 0x00.

BRNE Atraso //Equanto R2 > 0 volta a decrementar R3.

RET //Retorno da sub-rotina.

Dessa forma, a instrução DEC consome um ciclo, a instrução BRNE consome dois ciclos e na última vez, quando não há mais desvios, consome um ciclo. Como os valores de R3 e R2 começam em zero, o primeiro decremento leva ao valor 255 (considerando que ambos são números de 8 bits) e o decremento de R3 é repetido dentro do laço de R2, espera-se que haja 256 decrementos de R3 vezes 256 decrementos de R2, considerando 3 ciclos para DEC e BRNE temos aproximadamente (3 x 256 x 256 ) + ( 3 x 256) o que resulta em 197.376 ciclos. A parcela 3 x 256 é o tempo gasto no decremento de R2.

Nesse exmplo foi utilizado um clock de 16Mhz ( periodo de 62,5 ns), com isso temos que da chamada da sub-rotina ate o seu retorno foi gasto cerca de 12,23 ms.

Como o clock utilizado no nosso projeto é de 128Khz (periodo de 7,81us), a quantidade de ciclos necessária para obter-se 0,5 segundos é de aproximadamente:

X = 0,5 / 7,81us 🡪 X = 64.020 ciclos

O que resulta em um laço de decrementação de R3 em R2 com ambos os valores iniciando em 146, dando assim um total de (3x146x146) + (3x146) = 64.386 ciclos.

### Conversor ADC

Um dos pontos principais para serem estudados foi como seria feita a leitura do potênciômetro em Assembly.

Para essa primeira questão foi visto que para habilitar a conversão ADC do microcontrolador é necessário inicializar dois registradoes: ADCSRA e ADMUX. Ambos são de 8 bits.

Cada bit desses registradores tem uma determinada função, no caso do ADCSRA o bit 7 (ADEN) e 6 (ADSC) dão enable e iniciam a conversão respectivamente, ambos devem estar em 1 para que isso ocorra. O bit 4 (ADIF) é setado em 1 quando a conversão ADC tiver sido compleatda e os registros de dados atualizados. Os bits de 0 a 2 (ADPS 2:0) determinam o fator de divisão entre a frequência do sistema e o clock de entrada para o conversor ADC. Por padrão a frequência do conversor ADC precisa ser entre 50Khz e 200Khz, no projeto estamos utilizando um clock de 128Khz, ou seja, já esta dentro da margem, porém, foi utilizado o fator de conversão 2 [001], resultando em 64Khz.

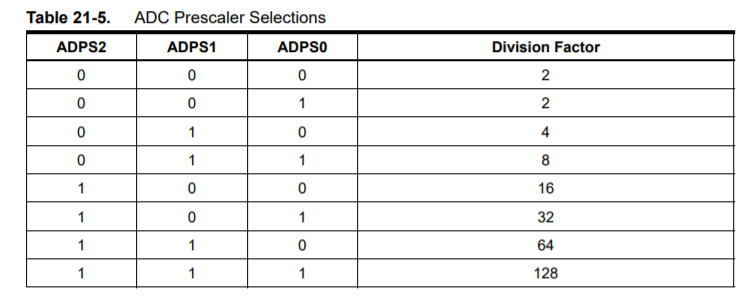


Figura 10: Tabela de redução de frequência

Fonte: Datasheet do AVR ATMega 328P

Quanto ao ADMUX os bits 7 e 6 (REFS 1:0) escolhem a tensão de referência para o conversor ADC que pode ser escolhida entre essas opções. A opção escolhida foi o de [01], Avcc.

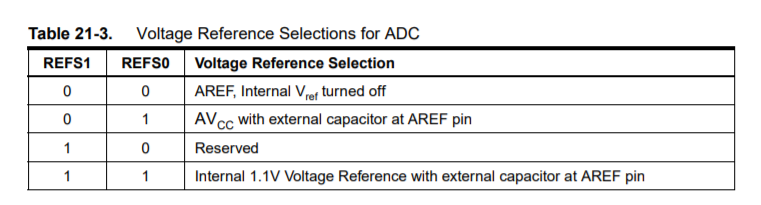


Figura 11: Tabela de seleção da tensão de referência

Fonte: Datasheet do AVR ATMega 328P

A conversão ADC segue a seguinte fórmula.

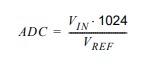


Figura 12: Equação de conversão ADC

Fonte: Datasheet do AVR ATMega 328P

Onde o ADC é o valor máximo que o potenciômetro pode atingir que no caso é de 999 e o VIN é a tensão máxima do potenciômetro que é de aproximadamente 4,88V. Dessa forma podemos encontrar que a tensão de referência é de aproximadamente 5V. Foi necessário utilizar um filtro passa-baixa no pino do AVCC.

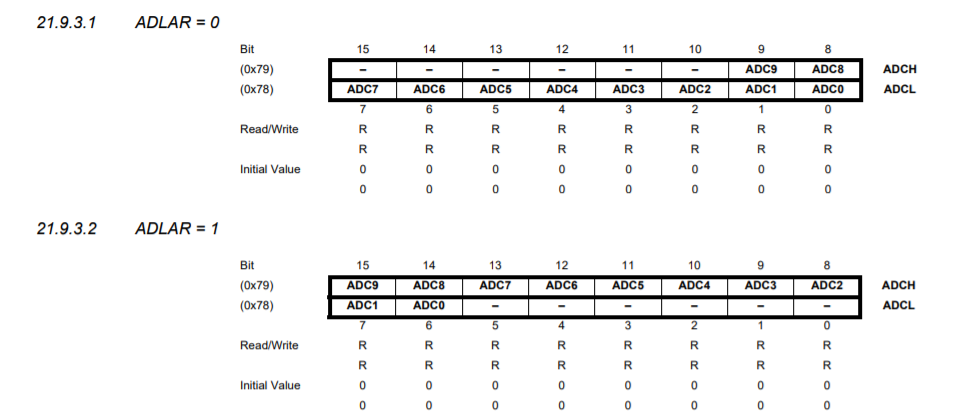
O bit 5 (ADLAR) é o bit que vai definir como será salvo os resultado da conversão.

Figura 13: Tabela do ADLAR

### Fonte: Datasheet do AVR ATMega 328P

O resultado da conversão é salvo em dois registradores de 8 bits: ADCH e ADCL que é feito de acorco com o bit ADLAR. No projeto foi tulizado o ADLAR = 0 resultando nos 2 MSB salvos em ADCH e outros em ADCL.

Por fim, os bits 3..0 são responsáveis por selecionar a entrada do microcontrolador a ser concectada ao conversor, no projeto nos utilizamos a primeira opção: ADC0 [0000].

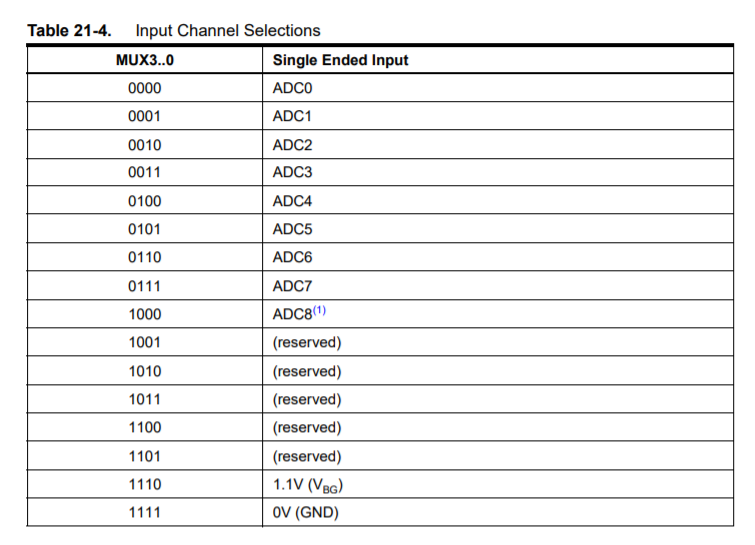


Figura 14: Tabela do ADLAR

### Fonte: Datasheet do AVR ATMega 328P

### Amostragem no display

A segmentação dos valores para os diplay é baseada numa divisão de grupos seguida de sucessivas subtrações. Com a divisão do valor lido em ADCH e ADCL, há 4 combinações possíveis:

* ADCH = “00000000” + ADCL = “xxxxxxxx”, o que representa, na prática, 000 + um valor até 255;
* ADCH = “00000001” + ADCL = “xxxxxxxx”, o que representa, na prática, 256 + um valor até 255;
* ADCH = “00000010” + ADCL = “xxxxxxxx”, o que representa, na prática, 512 + um valor até 255;
* ADCH = “00000011” + ADCL = “xxxxxxxx”, o que representa, na prática, 768 + um valor até 255;

De acordo com a combinação dos dois bits menos significativos do ADCH, os valores práticos de centena, dezena e unidade serão somados aos valores segmentados do ADCL.

Por exemplo: se o valor é 527, ADCH contém “00000010” e ADCL contém “00001111”. O valor prático de ADCH representa 512, enquanto o valor prático de ADCL representa 15. ADCL, então, será sujeito a repetidas subtrações de modo que o registrador da centena contenha 0, o registrador da dezena contenha 1 e o registrador da unidade contenha 5. Em seguida, o registrador da centena é somado com 5, o registrador da dezena é somado com 1 e o registrador da unidade é somado com 2, resultando assim nos dígitos 5, 2 e 7 nos registradores.

# Conclusão

A simulação do código funcionou corretamente, passando por todos os estados e realizando a verificação das senhas, esperando o intervalo de 0,5 a cada inserção de senha e liberando o cofre quando houvesse o acerto das três senhas ou trancando o cofre caso houvesse alguma senha errada, o único problema foi na parte de amostragem do valor do potenciômetro do display, apenas a unidade está funcionando corretamente, a dezena e centena ficam alternando valores repetidamente. Esse problema se deu porque o valor do potenciômetro está sendo atualizado a todo instante nos displays, para resolver isso seria necessário criar um estado ou algum label de verificação de conversão, onde apenas atualizaria o valor caso houvesse alguma variação do valor do potenciômetro, porém, devido ao tempo não conseguimos consertar.

# Referências

1. VAHID, Frank. , *Sistemas digitais: projeto, otimização e HDLs.*, Artmed, Porto Alegre, 2008.
2. *ATMega 328P Datasheet*. Disponível em: <https://ww1.microchip.com/downloads/en

/DeviceDoc/Atmel-7810-Automotive-Microcontrollers- ATmega328P\_Datasheet.pdf> Acesso em: 10 de out. de 2020.

1. *Display Datasheet*. Disponível em: <https://pdf1.alldatasheet.com/datasheet- pdf/view/703367/BETLUX/BL-T56A-31.html> Acesso em: 10 de out. de 2020.
2. *Decoder Datasheet*. Disponível em: [<https://www.ti.com/lit/ds/sdls111/sdls111.pdf?](http://www.ti.com/lit/ds/sdls111/sdls111.pdf) ts=1602356882572&ref\_url=https%253A%252F%252Fwww.google.com%252F> Acesso em: 10 de out. de 2012
3. Leituas de chaves mecânicas e o processo de debouncing. **Embarcados**. Disponível em: < <https://www.embarcados.com.br/leitura-de-chaves-debounce/> > Acesso em: 17/10/2020
4. E-book Avr e Arduino: Técnicas de projeto. Disponícel em: <[https://pt.scribd.com/read/421356943/Avr-E-Arduino-Tecnicas-De-Projeto#](https://pt.scribd.com/read/421356943/Avr-E-Arduino-Tecnicas-De-Projeto)> Acesso em: 15/10/2020