UNIFEI - Universidade Federal de Itajubá

IESTI - Instituto de Engenharia de Sistemas e Tecnologia da Informação

PIC Genius

Genius é um jogo eletrônico de habilidade de memória. Mundialmente conhecido como Genius ou Simon na versão internacional, o brinquedo cria uma série de tons e luzes e exige que o usuário repita a série. Se o usuário repete corretamente essa sequência, a série se torna progressivamente mais longa e mais complexa e o desafio é o jogador acertar todas as sequências até o último nível. Assim que o usuário falha, o jogo termina.

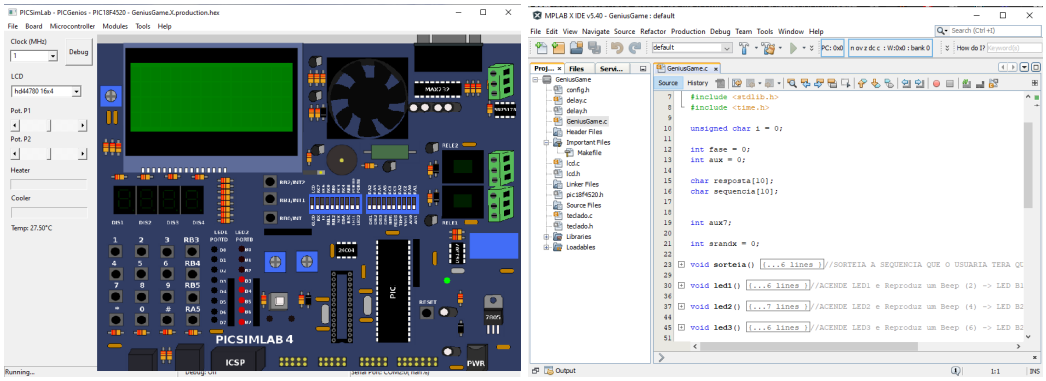
O código do projeto pode ser acessado pelo repositório no GitHub:
[JoaoTonet/PICGenius: Repositório do projeto PICGenius. Jogo similar ao Genius, desenvolvido na matéria de Programação Embarcada - UNIFEI (github.com)](https://github.com/JoaoTonet/PICGenius)

E para mais informações sobre o Projeto, Autor e Mentores:
[João Henrique Pulini Tonet | LinkedIn]

Ou no Youtube:
https://youtu.be/6FSo921Fs60

Passo 1: Ambiente e Materiais

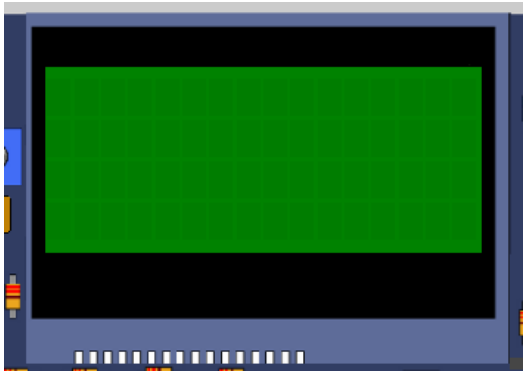
A proposta do projeto é desenvolver um jogo da memória semelhante ao brinquedo Genius, através dos conceitos apresentados na disciplina de Programação Embarcada, e pelo ambiente de simulação PICSimLab utilizando a placa PICGenios e o Microcontrolador PIC18F4520 (devido a situação pandêmica durante o semestre e a indisponibilidade dos componentes físicos).



Passo 2: Funcionamento dos Periféricos

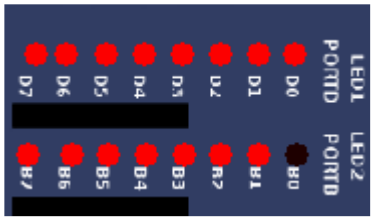
Uma vez que o projeto nessa etapa fará uso somente do ambiente simulado, utilizando uma placa pronta, deve-se possuir instalado na máquina o PICSimLab, disponível em <PICSimLab - PIC Simulator Laboratory - Browse Files at SourceForge.net> e o IDE MPLab X disponível em <MPLAB X IDE | Microchip Technology>.

Utilização do LCD:



O Display LCD 16x4, funciona como um meio de comunicação do jogo com o usuário, e será configurado para fornecer algumas informações, como o nome do jogo, telas de carregamento, instruções necessárias para um jogador iniciante e comandos para o jogador se atentar a sequência e a sua vez de jogar.

Utilização do LEDs:

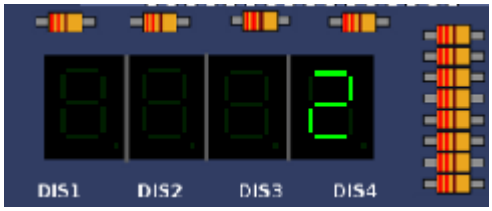


Os Leds, deveriam ser “coloridos”, e utilizamos somente 4 deles, uma para cada cor, contudo, o ambiente de simulação possui apenas LEDs vermelhos enfileirados, e por isso serão utilizados de acordo com o número de suas posições para representar as cores. Os Leds piscam na sequência que será a sequência que deverá ser repetida a cada fase que o usuário avança.

Utilização do Teclado:

O teclado será por onde o usuário de fato jogará, ele deve apertar as teclas que serão atribuídas a cada LED, na sequência fornecida.

Utilização do Display de 7 Segmentos:



O Display de 7 Segmentos funcionará como um como um contador de fases, para que o jogador possa se basear. O jogo será configurado para possuir 10 níveis.

Utilização do Buzzer:

Haverá um Buzzer que emitirá um som ao clicar em um botão e ao acender de um LED, assim como no brinquedo original, para que o usuário possa receber uma confirmação e o jogo se torne mais imersivo.

Passo 3: O Código e o Funcionamento (as mais importantes)

A main:

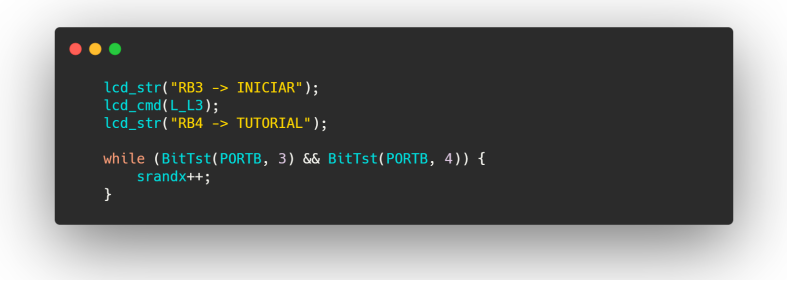
Na main do programa, foi configurado somente o loop que faz o jogo acontecer por 10 fases e as chamadas das funções. Quando o programa inicia, são feitas as configurações e são chamadas as funções em sequência. Caso o usuário consiga concluir as 10 fases, é executada a função ganhou( ), caso contrário, o programa para na função leitura( ), que realiza a verificação da entrada do jogador.

```
void main() {
    //Inicializações
    TRISA = 0x00;
    ADCON1 = 0x06;
    TRISB = 0xF8;
    TRISD = 0x00;
    TRISC = 0x01;
    PORTC = 0b00000010;
    TRISE = 0x00;
    lcd_init();
    start_game();
    TRISB = 0x00;
    while (fase < 10) {
        atencao();
        d7seg();
        show_seq();
        BitClr(PORTA, 5);
        lcd_cmd(L_CLR);
        lcd_str("    SUA VEZ");
        leitura();
        prox_fase();
    }
    ganhou();
}
```

(função main)

A função start_game:

Essa função reproduz no LCD uma animação de carregamento e posteriormente uma mensagem para o usuário que deve escolher através do teclado, se deseja assistir a um breve tutorial ou iniciar o jogo diretamente. Posteriormente a escolha, exibe novamente no LCD o tutorial ou o início do jogo.



```
    lcd_str("RB3 -> INICIAR");
    lcd_cmd(L_L3);
    lcd_str("RB4 -> TUTORIAL");

    while (BitTst(PORTB, 3) && BitTst(PORTB, 4)) {
        srandx++;
    }
```

(trecho da função start_game)

Há também nessa parte do código, um loop que espera o pressionar do botão RB3 para iniciar o jogo. Nessa parte foi feito uma modificação e acrescentado uma variável que vai incrementando conforme o usuário demora para clicar no botão. Esse trecho foi implementado para solucionar um problema que me deparei ao tentar randomizar a sequência de cores do jogo uma vez que não era possível randomizar com os comandos da linguagem C srand(time(null)). A variável será utilizada posteriormente como uma seed para gerar a sequência randômica.

A função sorteia( ):

A função sorteia é chamada no final da função start game, e utilizando a seed gerada anteriormente, randomiza um sequência com 10 números de números 2/4/6 e 8, que posteriormente é armazenada num vetor sequencia[].

```
void sorteia() {//SORTEIA A SEQUENCIA QUE O USUARIA TERA QUE REPRODUZIR
    srand(srandx * 7919); //A Seed depende do tempo q demora para startar o
game
    for (i = 0; i < 10; i++) {
        sequencia[i] = ((rand() % 4 + 1)*2 + '0');
    }
}//SORTEIA A SEQUENCIA QUE O USUARIA TERA QUE REPRODUZIR
```

(função sorteia)

As funções ledx( ):

As funções ledx vão de led1( ) a led4( ). São chamadas pela funções show_seq( ) e são responsáveis por acender o LED correspondente em PORTD e emitir o som no Buzzer. A única diferença entre as funções é o endereço do PORTD que será alterado.

```
void led1() {//ACENDE LED1 e Reproduz um Beep (2) -> LED B1 RED
    PORTB = 0x02;
    PORTC = 0b00000000;
    atraso_ms(600);
    PORTC = 0b00000010;
}//ACENDE LED1 e Reproduz um Beep (2) -> LED B1 RED
```

(função led1)

A função leitura( ):

Na função leitura, o jogador entra com o número correspondente ao LED da sequência que deve ser reproduzida e já é feita uma verificação se a entrada corresponde ao valor que deveria ser com a função verifica( ).

```
void leitura() {//LEITURA DAS TECLAS, EXCLUSIVO PARA (2)(4)(6)(8)
    unsigned char tmp;
    aux = 0;
    while (aux <= fase) {
        TRISD = 0x0F;
        tmp = tc_tecla(0) + 0x30;
        if (tmp == '2' || tmp == '4' || tmp == '6' || tmp == '8') {
            TRISB = 0x00;
            TRISD = 0x00;
            resposta[aux] = tmp;
            verifica(aux);
            aux++;
        }
    }
}
```

(função leitura)

A função verifica( ):

Sempre que ocorre a entrada do usuário no teclado, essa função chama a função ledx( ) que irá acender o led correspondente e depois verifica se está de acordo com a sequência inicial que foi definida previamente pela função sorteia( ).

```
void verifica(int aux) {//VERIFICA SEQUENCIA ENTRADA PELO DO USUARIO
    if (resposta[aux] == '2') {
        led1();
    }
  .
  .
  .
    PORTB = 0x00;

    if (resposta[aux] != sequencia[aux]) {
        game_over();
    }
}//VERIFICA SEQUENCIA ENTRADA PELO DO USUARIO
```

(função verifica)

As demais funções:

As funções que não foram exemplificadas acima, possuem um funcionamento mais cosmético mas que deixam o jogo mais imersivo e bonito. As funções ganhou( ) e game_over( ) por exemplo, emitem uma sequência característica de sons no Buzzer, e mensagens para o usuário, já a função atencao( ), mostra no Display LCD para o usuário se preparar para decorar a sequência que está chegando. O repositório do projeto pode ser encontrado no GitHub

<JoaoTonet/PICGenius: Repositório do projeto PICGenius. Jogo similar ao Genius, desenvolvido na matéria de Programação Embarcada - UNIFEI (github.com)>

Passo 4: Funcionamento do Jogo

Em poucas palavras, o jogo é bastante intuitivo, ele inicia com uma tela de carregamento no Display LCD e em sequência mostra duas opções, iniciar ou tutorial. O usuário escolhe pelo teclado pelos botões RB3 e RB4. A opção tutorial é uma explicação de como o jogo funciona, e quais teclas representam cada LED. Quando o jogo de fato inicia, uma mensagem de atenção é colocada no LCD e a sequência é exibida pelos LEDs, juntamente de um som do Buzzer. Ao acabar a sequência para ser repetida, aparecerá no LCD que é a vez do jogador, que deve repetir a sequência que acabou de ver pelas teclas de acordo com os LEDs que acenderam. Nessa parte, cada clique do usuário é confirmado novamente pelo som do buzzer e com o acender do LED correspondente. Caso o usuário clique em um botão inválido nada acontecerá, mas caso aperte um botão errado da sequência o jogo é finalizado e é exibido GameOver, oferecendo a oportunidade de reiniciar. Caso acerte a sequência passando pelos 10 níveis, o jogador vence.

Para uma explicação mais detalhada acesse <https://youtu.be/6FSo921Fs60>