CONSOLE DE JOGOS PORTÁTIL UTILIZANDO MATRIZES DE LED

André Luiz Lima Souza, Lucas Morais Freire, Maria Clara Moura de Freitas
UFRN - Universidade Federal do Rio Grande do Norte
Natal – RN, Brasil
andreluizlimaa@gmail.com, lucasmoraisf@gmail.com, maria.clarafreitas@hotmail.com

Resumo **-** Utilizando matrizes de LED, um Arduino Uno como controle e um Arduino AtMega 2560 para o processamento do jogo e dos gráficos, foi montado um console portátil capaz de exibir o clássico Tetris em matrizes de LED de 3mm.

1. INTRODUÇÃO

A partir da necessidade de desenvolver um projeto que explorasse a comunicação serial entre dois Arduinos, surgiu a ideia de criar um console de jogos portátil. Para colocar em prática essa proposta, o clássico Tetris foi escolhido como o jogo a ser implementado no dispositivo.

A comunicação serial entre os dois Arduinos desempenha um papel fundamental no projeto, permitindo a troca de informações em tempo real durante o jogo. Um dos Arduinos é responsável por gerenciar a lógica do jogo e os gráficos, sendo este o Atmega2560, onde ele processa as entradas do jogador advindas do outro microcontrolador e atualiza matrizes de LED que funcionam como *display*. O outro Arduino, por sua vez sendo o Uno, é encarregado de receber as entradas dos botões, servindo como um controle com fio.

O desenvolvimento do console portátil envolveu diversas etapas, desde a programação da lógica do Tetris em um dos Arduinos até a implementação da comunicação serial e o gerenciamento dos quadros de animação. Além disso, foi necessário a escolha dos componentes eletrônicos, a montagem do circuito e a criação de uma estrutura física para abrigar os componentes.

O projeto do console portátil com Tetris com comunicação serial entre dois Arduinos não apenas proporcionou um desafio interessante em termos de programação e eletrônica, mas também resultou em um dispositivo funcional e divertido, demonstrando o potencial da plataforma Arduino para a criação de projetos criativos e inovadores.

2. TEORIA

O Arduino Uno e o ATmega2560, ambos microcontroladores da família AVR da Atmel (agora Microchip), são pilares no mundo da prototipagem e projetos eletrônicos. Apesar de compartilharem a mesma arquitetura, suas características e capacidades diferem, atendendo a diferentes necessidades.

2.1. Microcontroladores:

- ATmega328P (Arduino Uno): Cérebro do Arduino Uno, este microcontrolador de 8 bits oferece 32 KB de memória flash para armazenar o código do programa, 2 KB de SRAM para dados temporários e 1 KB de EEPROM para armazenamento persistente. Opera a 16 MHz, possui 23 pinos de entrada/saída digitais (dos quais 6 podem ser usados como PWM) e 6 entradas analógicas.
- ATmega2560: Com capacidades expandidas, o ATmega2560 possui 256 KB de memória flash, 8 KB de SRAM e 4 KB de EEPROM. Opera também a 16 MHz e oferece 54 pinos de entrada/saída digitais (15 PWM) e 16 entradas analógicas.

2.1.1. Pinagem:

A pinagem de ambos os microcontroladores inclui:

- Pinos Digitais: Usados para entrada/saída de sinais digitais (HIGH/LOW).
- Pinos Analógicos: Medem tensões analógicas e as convertem em valores digitais.
- Pinos de Alimentação: Fornecem energia (5V, 3.3V, GND) aos componentes.
- Pinos de Comunicação Serial: UART (TX/RX) para comunicação serial.
- Pinos de Reset: Reiniciam o microcontrolador.
- Pinos de Interrupção: Disparam rotinas específicas quando ocorrem eventos externos.
- Pinos PWM: Geram sinais PWM para controle de motores e LEDs.

2.1.2. Comunicação Serial:

Ambos são capazes de utilizar a interface UART para comunicação serial com computadores ou outros dispositivos. Os registradores de controle relevantes serão manipulados para tal.

2.1.3. Alimentação:

Podem ser alimentados por USB, fonte externa (7-12V) ou bateria (9V). A tensão é regulada internamente para 5V.

2.1.4. Clock:

Ambos operam com um clock de 16MHz, que determina a velocidade de execução das instruções.

2.1.5. Timers:

Possuem timers internos para gerar atrasos precisos, gerar sinais PWM e realizar outras tarefas que exigem temporização

2.1.6. Interrupções:

Permitem que eventos externos ou internos interrompam a execução normal do programa e executem rotinas específicas. Isso é útil para responder a sensores, botões e outros dispositivos.

2.1.7. Comunicação I2C/SPI:

Ambos suportam os protocolos I2C e SPI para comunicação com dispositivos como sensores, displays e módulos de memória.

2.2 Matrizes de LED

Matrizes de LED são dispositivos que contém 64 LEDs arranjados em uma malha 8x8. Para possibilitar a amostragem de imagens quaisquer, é necessário ligar cada linha da matriz separadamente, e isto deve ser feito rápido o suficiente para que o olho humano não perceba, fazendo parecer uma imagem constante e sólida. Cada matriz possui 16 pinos, 8 deles sendo ligados aos anodos de uma linha de LEDs cada. Já os outros 8 são ligados aos catodos de uma coluna de LEDs cada.

3. MATERIAIS NECESSÁRIOS

Para o desenvolvimento do protótipo de console para o jogo Tetris. Foi-se necessário: Arduino UNO(Atmega 328P), Arduino AtMega2560, Jumpers - Macho/Fêmea, 16 resistores de 220Ω, sete push-buttons e seis matrizes de led 1088AS.

Figura 1 - Microcontrolador Arduíno UNO

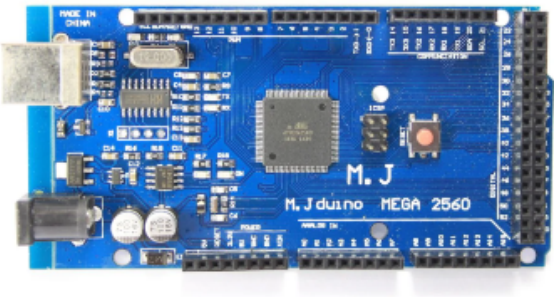

Figura 2 - Microcontrolador ArduinoAtmega260

Figura 3 – Jumpers - Macho/Fêmea

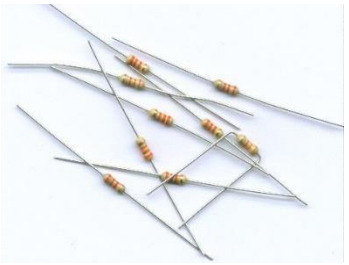

Figura 4 – Resistores de 220 ohms

Figura 5 – Botão Tactil 6x6x5cm

Figura 6 – Matriz de LEDs 1088AS

4. METODOLOGIA

A metodologia utilizada para desenvolver o projeto, após ter sido idealizado e verificado que seria possível realizá-lo, foi fazer uma pesquisa e orçamento dos materiais necessários que não estavam disponíveis para uso no laboratório da universidade, sendo esses as matrizes de LED e jumpers que precisariam ser soldados. Em seguida, implementamos o código necessário, realizamos algumas simulações e fizemos a montagem de todo o projeto presencialmente. Durante a fase de idealização, foi necessário realizar várias pesquisas para compreender a lógica de gerenciamento de quadros de animação e como operar as matrizes de LED. Decidimos fazer um controle em um microcontrolador separado, conectado ao gerenciador da lógica do jogo e gráficos via comunicação UART. Depois de muita pesquisa e estudos para verificar a viabilidade, escolhemos desenvolver o jogo Tetris para o dispositivo, mas foi feita uma biblioteca que possibilita a implementação de qualquer tipo jogo desde que possua lógica que não seja custosa de tal forma a impossibilitar uma taxa de quadros fluída e uma experiência decente para o usuário.

Tivemos dificuldade nas simulações pois muitas ferramentas online não possuíam os componentes necessários ou não suportavam comunicação entre vários microcontroladores. Algumas das partes do projeto foram simuladas nos sites Wokwi (https://wokwi.com), que não tem suporte a comunicação serial entre microcontroladores, e no Tinkercad (https://www.tinkercad.com), que possui suporte a comunicação, porém não possui as matrizes de LED. Para a implementação do código do jogo em si, utilizamos o site p5.js (https://p5js.org/), que, apesar de ser em JavaScript, é o suficiente para construir a lógica do jogo para que consigamos traduzi-la para o C++ no Arduino. Muitos dos testes foram feitos presencialmente sem o uso de simuladores devido às limitações encontradas.

Após tudo isso, montamos o projeto final, primeiramente no laboratório para garantir que estava funcionando, e depois fizemos uma montagem utilizando papelão para criar uma base. Alguns dos componentes foram soldados para evitar problemas de mal-contato e para encaixar melhor no design idealizado.

Para o Arduino gerenciador dos gráficos, a arquitetura do sistema e da biblioteca feita funciona da seguinte maneira: primeiro escolhe-se quais PORTs do Arduino serão utilizados para iterar sobre as linhas das matrizes de LED (3 PORTs), e quais PORTs serão utilizados para impor 0V ou alta impedância nas colunas das matrizes. Ao impor 0V, o LED daquela coluna liga, e ao impor alta impedância, o LED daquela coluna desliga. Após isso, escolhe-se a taxa de quadros do jogo. A taxa de quadros da tela é 60 quadros por segundo, mas a do jogo pode ser qualquer divisor deste número. O programador irá inicializar a biblioteca e esta irá cuidar do gerenciamento dos quadros e entradas. no início de cada quadro, o programador pode solicitar quais entradas foram realizadas pelo usuário durante o quadro anterior, e a partir destas ele irá construir a lógica do jogo com o objetivo de montar um quadro (matriz de booleanos 24x16), que será passado para uma função no fim do loop de execução. esta matriz será convertida pelo sistema em bytes que serão aplicados nos PORTs configurados, efetivamente mostrando o quadro que foi montado na tela. Cada quadro é mostrado a partir de uma rotina de interrupção, que é executada a cada $\frac{1}{60\times24}$ segundos. Isso porque o *display* tem taxa de atualização de 60 quadros por segundo e possui 24 linhas; quando a última linha é desenhada, realiza-se a troca de quadro se um quadro estiver pronto. Também na troca de quadro é solicitado ao controle que envie as entradas que ele coletou desde a última solicitação. Para cada quadro, é possível solicitar do sistema as entradas do usuário, gerenciada por uma interrupção de recebimento de bytes via UART. A quantidade de entradas e a contagem global de quadros também fica disponível para o programador. Com estas informações, um programador é capaz de desenvolver qualquer jogo viável para a capacidade de processamento do Arduino.

5. RESULTADOS

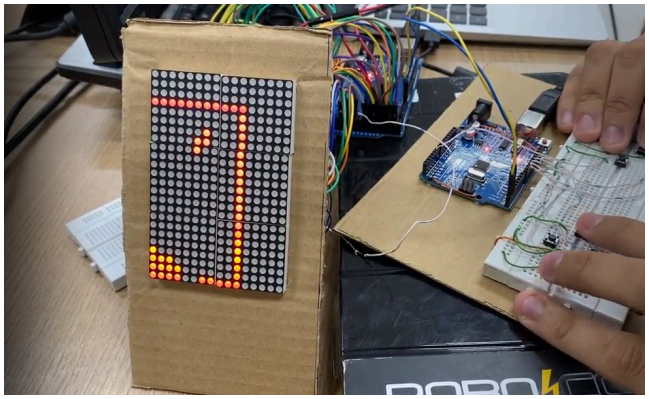

Figura 6 – Dispositivo em funcionamento.

Como podemos observar na Figura 6, o dispositivo está em pleno funcionamento. Ele não apresentou travamentos, e executou a lógica do jogo pretendido de maneira consistente.

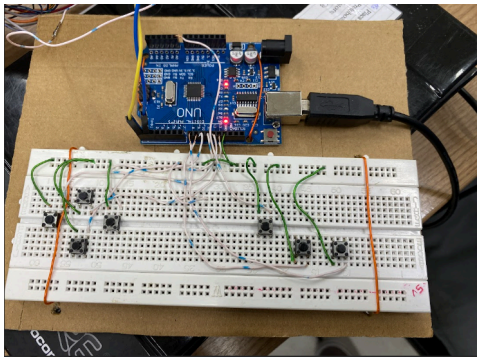

Figura 7 – Layout do controle.

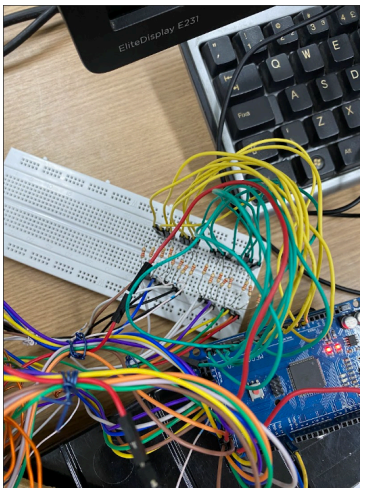

Figura 8 – Ligações no AtMega2560.

6. CONCLUSÃO

Em suma, o projeto foi de um aprendizado imprescindível, exigindo habilidades práticas como solda e montagem, como também um bom entendimento do funcionamento dos microcontroladores em questão e dos protocolos de comunicação. A possibilidade de desenvolver jogos diversos para esta plataforma é uma característica muito interessante, e a natureza lúdica permite um engajamento maior dos interessados.

7. REFERÊNCIAS BIBLIOGRÁFICAS

[1] ARDUINO WEBSITE. Disponível em < https://www.arduino.cc/> . Acesso em 5 de junho de 2024.

[2] Cplus plus. (19 de Junho de 2016). Fonte: cplusplus: http://www.cplusplus.com/reference/cstdlib/atoi/?kw=atoi

[3] LIMA, C. B. & VILLAÇA, M. V. M. AVR e Arduino: Técnicas de Projeto. Florianópolis: Ed. dos autores, 2012.