Construção do Jogo *Ice Cold Beer* Utilizando Sistema Embarcado IoT

Erik J. F. do Nascimento[1], **Ana C. S. Abreu**[1], **Sandro C. S. Jucá**[1], **Filipe A. Lira**[1], **José N. M. Júnior**[1]

Instituto Federal de Educação, Ciência e Tecnologia do Ceará - (IFCE)
Caixa Postal S/N – 61.919-140 – Maracanaú – CE – Brasil

{erikjhonesf,anacarolinaks19, sandro.juca, filipe.al2015}@gmail.com

junior.mourao@live.com

Abstract. *The embedded system used in this project is based on ESP32, which is a low cost microcontroller, high processing power and other features like bluetooth and WiFi. Its use in IoT applications is quite wide. The purpose of this project is to develop a prototype to emulate the arcade game known as ICE COLD BEER. ESP32 was used to control the game via WiFi. In this project it was also used internal and external interruptions, access point (AP) function, web server and RTOS (Real Time Operating System) concepts.*

Resumo. *O sistema embarcado usado neste projeto é baseado no ESP32, que é um microcontrolador de baixo custo, alto poder de processamento e outras funcionalidades como bluetooth e WiFi. Seu uso em aplicaçoõs IoT é bem vasto. O propósito deste projeto é desenvolver um protótipo para emular o jogo arcade conhecido como ICE COLD BEER. Foi usado o ESP32 para controlar o jogo via WiFi. Neste projeto foi usado também interrupções internas e externas, função de ponto de acesso (AP), servidor web e conceitos de RTOS (Sistema Operacional de Tempo Real).*

1. Introdução

O desenvolvimento de sistemas que usam algum *chip* microcontrolador está em constante expansão. Esses pequenos computadores permitem a redução drástica na quantidade de componentes da aplicação e em seu tamanho, o que impacta em menor custo de desenvolvimento. O uso de sistemas embarcados de Internet das Coisas (IoT) - do inglês *Internet of Things* - possibilita o acionamento remoto via WiFi de periféricos como aparelhos elétricos, sensores, alarmes, e também componentes eletrônicos como motores, LEDs e relés conectados a ele. Para realizar essa comunicação, pode ser utilizado algum aparelho com acesso à rede, como *smartphones* ou *notebooks*. Com essa ideia em mente, foi pensado em construir o jogo arcade mecânico *Ice Cold Beer* (ICB), em uma versão de baixo custo usando um sistema embarcado baseado no ESP32.

O jogo ICB original possui muitas funcionalidades, como tempo limite, marcação de pontuação, passagem de nível, escolha de dificuldade, acionamento do braço mecânico para jogar, retorno automático desse braço em casos de sucesso ou fracasso para o início da tela, entre outras. Isso acarreta em alguns problemas, como maior complexidade na construção do jogo, uso de muitos componentes mecânicos e alto custo de fabricação e manutenção.

Para contornar essas dificuldades, foi estudado e implementado o uso do sistema embarcado ESP32. O *chip* foi responsável por todo o gerenciamento das funções lógicas e de decisão do jogo, assim como acionar a parte mecânica através de dispositivos móveis que se comunicam com o módulo via WiFi. O uso do ESP32 foi motivado para tornar o projeto mais simples e reduzir o tamanho físico do jogo, assim como a quantidade de componentes utilizados e o custo de montagem. O objetivo deste trabalho é mostrar o quão útil é utilizar o microcontrolador ESP32 para controlar dispositivos mecânicos usando conceitos IoT, a praticidade que é montar equipamentos usando esse módulo e estudar suas funcionalidades.

2. Referencial Teórico

O microcontrolador ESP32 foi lançado em 2016/2017 pela Espressif. Ele é basicamente um computador com as dimensões de um pequeno *chip* [Morais 2017]. Ele possui comunicação *Bluetooth* e acesso à rede via WiFi. Com isso, é possível desenvolver uma maior variedade de projetos IoT. O ESP32 pode atuar como uma estação WiFi, como um *Access Point* (AP) ou ambos. Ao definir o ESP32 como um AP, é possível conectar-se a ele usando qualquer dispositivo com acesso WiFi sem a necessidade de se conectar ao roteador. Essa indepência da rede mundial de Internet, somado com as múltiplas características do ESP32, permite uma flexibilidade susbstancial no controle e monitoramento de dispositivos usando esse microcontrolador. Como ele pode funcionar como estação WiFi e AP simultaneamente, então ele é capaz de enviar constantemente dados de monitoramento da aplicação que ele está gerenciando diretamente para alguma página ou servidor na web. Isso garante maior controle da tarefa realizada e pode ser mostrado em dados estatísticos o desempenho do projeto [Santos et al. 2019].

O ESP32 não se limita somente a aplicações de automatização redizencial baseado em IoT. [Sharp and Vagapov 2017] mostra um exemplo de aplicação do ESP32 utilizando suas funções de acesso a Internet via WiFi e AP. [Rai and Rehman 2019] propôs a implementação de um sistema de vigilância inteligente usando o microcontrolador ESP32. Ele utilizou dos conceitos de acesso a Internet do *chip* para enviar os dados diretamente para um site.

Neste projeto foi utilizado o jogo *Ice Cold Beer* para mostrar a grande versatilidade desse sistema embarcado como aplicação real e funcional. O jogo *Ice Cold Beer* é um arcade mecânico criado pela Taito em 1983. O objetivo do jogo é levar uma bolinha com uma barra erguida por motores bidirecionais laterais e colocar a bolinha nos buracos (10 no total) indicados na tela do jogo antes do estouro de um tempo pré-definido. Se a bolinha cair em outro buraco, o jogador começa no início da tela novamente (o tempo não é alterado). A tela possui vários outros buracos dispostos em quantidades e locais diferentes ao longo dela para definir a dificuldade, [TAITO 1983]. Outros conceitos necessários usados neste projeto são as interrupções do ESP32. Uma interrupção é um evento gerado internamente ou externamente que, ao ser acionada, interrompe a tarefa atual do processador para processar o evento que a causou [Fernando 2018]. As interrupções são úteis para executar automaticamente programas de microcontroladores e podem ajudar a resolver problemas de sincronia de tempo.

3. Materiais e Métodos

Para o desenvolvimento do trabalho foram usados os seguintes materiais: 1 módulo ESP32 DevKit de 30 pinos; 2 Motores de passo bipolares de 6 fios, 12v cada; 1 *Driver* de potência ULN2803; *Protoboard*; 1 resistor de 2.2 KΩ; 1 resistor de 390 Ω; 1 fonte de alimentação de 9v; Fios para a conexão dos componentes. A parte física em que foram colocados os componentes foi construída utilizando uma gaveta de madeira como suporte. A parte da tela onde estão os buracos foi feita usando papelão. Foram furados alguns buracos no papelão, onde o furo mais próximo do topo é o buraco objetivo do jogo. Todo o sistema de movimento dos ”carros”em que ficam presos os motores foi aproveitado de impressoras velhas, definindo o projeto como utilização da metareciclagem. Ao final da montagem da parte mecânica e física, o projeto ficou como mostrado na Figura 1.

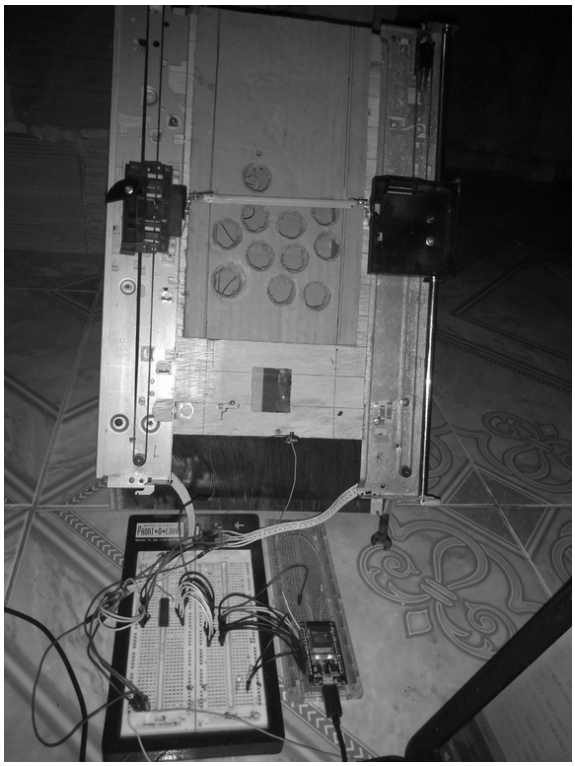

Figura 1. Visão da estrutura montada do ICB

Os dois motores de passo foram primeiramente fixados na estrutura. Depois foram identificados os fios de cada um dos motores, as bobinas e os seus fios comuns. Em seguida, foram conectados todos os fios comuns dos dois motores ao pólo positivo da fonte de alimentação. Os outros fios restantes foram nomeados em relação a sua disposição em cada bobina de cada motor, e em seguida eles foram conectados ao *driver* de potência, respeitando a orientação que será usada para cada passo de cada motor (O pino 9 do *driver* foi conectado ao terra da fonte de 9v, os pinos 11 à 18 foram conectados aos motores de passo, os pinos de 1 à 8 foram conectados ao ESP32). O pino 10 do *driver* foi conectado ao +9v para descarregar a energia gerada pela força contra eletromotriz, evitando danos aos componentes. Em seguida, foram conectados os pinos do ESP32 aos pinos do *driver* de potência seguindo a ordem de acionamento dos passos do motor. O evento que causa a vitória é definido por uma interrupção externa. Um fio que simula um botão foi

colocado no buraco alvo, e quando a bolinha cai nesse buraco, faz os fios se unirem momentaneamente, o que permite a passagem de tensão até o pino 23 do ESP32. O módulo captura esse aumento de nível lógico, indica que houve uma interrupção e chama a função adequada. Após acionado a função de interrupção, ela trata o evento de vitória exibindo uma mensagem na tela e a pontuação alcançada. Foi colocado um resistor de 2.2 kΩ ao ”botão”para reduzir a tensão recebida para não danificar o microcontrolador. A Figura 2 ilustra a identificação, nomeação e montagem dos motores, além dos passos listados acima para a montagem do circuito.

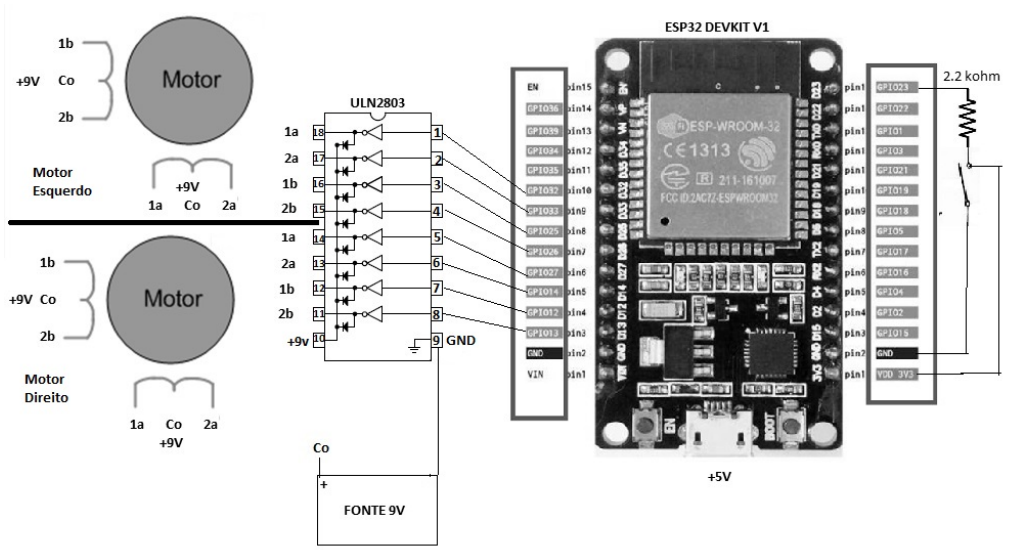

Figura 2. Esquema de montagem da parte elétrica do projeto

O LED azul do próprio ESP32 foi usado como indicador de que a interrupção foi gerada (quando a bolinha cai no buraco indicado, o LED azul apaga e uma mensagem de vitória é exibida na tela).

Para escrever o código fonte, compilar e gravá-lo no ESP32, foi usado o *software* Arduino IDE. Foi necessário configurar o ambiente de desenvolvimento de modo a compilar e gravar o ESP32. Isso foi feito acessando a barra de ferramentas e escolhendo a placa adequada do *chip* usado (neste caso foi a DevKit V1), assim como outras configurações de sincronia de gravação. Primeiro foi definido o módulo ESP32 como um AP e servidor web. O nome da rede escolhida para o AP foi *ICE_COLD_BEER* e não foi definido senha para ela. Quando o ESP32 é ligado, ele inicia seu AP e o nome da rede se torna visível para os dispositivos próximos. Ao conectar um dispositivo à rede criada, ele estará pronto para carregar a página com o controle do jogo. O dispositivo conectado mostrará que a rede está sem Internet. No entando, isso não afetará o carregamento da página pelo navegador e nem o controle do jogo. Isso acontece porque o ESP32 não está conectado à rede mundial de Internet, mas sim foi definido o módulo como um servidor web e a página HTML (*Hypertext Markup Language*) é acessada diretamente nele. Após definido os parâmetros da rede do AP e conexão ao servidor web, foi criada uma página HTML que foi resposável por controlar os motores de passo do projeto. Essa página foi escrita junto ao código principal e definida dentro do loop principal. Nela, foram definidos os quatro botões responsáveis por acionar os motores em ambas as direções. Foram utilizados conceitos de HTML e *Cascading Style Sheets* (CSS) para a parte visual da página.

Os botões são inicialmente deixados em vermelho, indicando que ambos os motores estão desligados. Quando um ou mais botões são acionados, eles mudam de cor para verde. Quando um botão é pressionado, uma nova *Uniform Resource Locator* (URL) é gerada e carregada, então o ESP32 verifica essa URL e extrai dela a informação de qual botão foi pressionado. Então isso é verificado em uma rotina condicional dentro do *loop* principal, que por sua vez chama a função de acionamento de motor adequada.

Para a lógica de controle dos motores de passo, após o apertar os botões da página, foram definidas alguns métodos e rotinas. Primeiro foi definido que os motores serão acionados usando passo *full-step* de uma fase. Isso tornou o código mais simples. Depois, foram definidos os pinos GPIOs 12, 13, 14 e 27 para o controle do motor de passo esquerdo. Para o motor direito foram usados os pinos 25, 26, 32, 33. Esses pinos foram programados como pinos de saída. Isso foi necessário para aplicar nivel lógico alto nos motores e assim acionar os passos. Foram escritas quatro funções de passo para cada motor. Essas funções configuram o nível lógico de cada um dos pinos de modo a realizar o passo indicado. Cada passo usa quatro pinos. O pino correspondente ao número do passo é configurado em nível alto, enquanto os outros são configurados em nível baixo. Após serem configurados os passos, foram definidas as funções que movimentam os motores para cima e para baixo. Elas chamam as funções de passo em uma determinada ordem, de forma a acionar o motor de modo apropriado para gerar o movimento. A ordem dos passos é definida tanto pelo motor usado quanto pela ordem da conexão dos seus pinos ao ESP32. Para fazer o motor descer, basta inverter a ordem dos passos. É importante notar que caso o tempo de *delay* (tempo de espera para acionamento do próximo passo) seja muito curto, o motor apenas vibrará em vez de girar, pois não tem tempo suficiente para a troca de orientação das bobinas internas.

Para definir a contagem de tempo de jogo, foi usado o *timer* interno do ESP32. Foi feita uma função que conta de 0 até 100 de forma regressiva. Essa função roda de forma paralela e idependente da rotina principal, assim as ações do jogo não interferem na contagem do tempo. Para ser usado o *timer* interno, foi necessário configurar o *timer*. Isso é feito com a função *timerBegin()*. Quando esse tempo chega ao fim, uma interrupção interna é gerada e uma função é chamada, indicando que o jogador foi derrotado e exibindo uma mensagem na tela. A função do *loop* principal do ESP32 é responsável por criar a página HTML e rodar as rotinas de acionamento dos motores. Dentro da função também é feito iterativamente a verificação de novos acessos ao AP definido. Após conectado à rede do jogo, o *loop* dá início ao carregamento da página e o usuário pode acessá-la através do endereço do *gateway* mostrada ao acessar a rede. Por fim, a metodologia para o desenvolvimento deste projeto se baseou nos conceitos de ABP (Aprendizado Baseado em Projeto). Isso permitiu que o trabalho fosse evoluído de uma forma bem mais rápida e eficiente.

4. Resultados e Discussões

O ponto de acesso criado no ESP32 não possui falhas aparentes e o sinal é estável o suficiente para manter o jogador conectado durante todo o jogo. Ao levar a bolinha até o objetivo definido, a interrupção é gerada como previsto e a rotina de vitória é executada sem problemas. Também tem-se a contagem do tempo que usa o *timer* interno do *clock* do ESP32, que não interfere nas outras tarefas. A tabela a seguir mostra os resultados de alguns testes com o ICB realizados por 3 pessoas durante uma apresentação do projeto

numa aula de microcontrolodades. Ela aponta o grau de satisfação de cada usuário indo de excelente, bom, regular, ruim e péssimo.

Tabela 1. Tabela demonstrativa das execuções do ICB

Usuário	Aparência	Tempo decorrido	Sinal do AP	Dificuldade	Controle	Geral
1	Regular	Bom	Excelente	Bom	Bom	Bom
2	Regular	Regular	Excelente	Bom	Regular	Regular
3	Bom	Ruim	Excelente	Regular	Ruim	Regular

Percebe-se pela tabela que o sinal do AP definido no ESP32 é o ponto forte da aplicação. O controle foi a parte que deixou a desejar. Isso se deve ao fato dos botões não serem dinâmicos e necessitarem de constante pressionamento para movimentar os braços. Isso acarreta em pouca flexibilidade no movimento e controle da bolinha. O tempo usado para alcançar o objetivo e a rotina gerada em casos de derrota e vitória respondem em tempo relativamente baixo quando são acionadas pelas devidas interrupções. Para acessar o código fonte e ter uma visão do projeto em funcionamento e breve explicação dele como um todo, acesse [do Nascimento 2019].

5. Considerações Finais

O sistema embarcado ESP32, com todo seu poder de processamento, memória, e funções adicionais, realizou todas as funções programadas de modo eficaz, eficiente e previsível. O uso desse *chip* reduziu consideravelmente o gasto com componentes adicionais, tempo de programação e de teste. A grande vantagem foi o suporte a WiFi de alta velocidade nativo do ESP32, que permitiu uma interação quase que instantânea entre o controle do jogo acessado pelo *smartphone* e os motores. As funções de interrupções também permitiram trabalhar com a lógica de derrota e vitória mais flexivelmente e facilmente. para trabalhos futuros pretende-se colocar uma trilha sonoro do jogo emulado.

Referências

do Nascimento, E. J. F. (2019). *Repositório com código, vídeo e imagens.* https://github.com/ErikJhones/ICE-COLD-BEER/tree/master.

Fernando, K. (2018). *Conhecendo o ESP32*. www.fernandok.com/2018/06/os-profissionais-sabem-disso-interrupt.html.

Morais, J. (2017). *Os profissionais sabem disso: Interrupt ISR*. https://portal.vidadesilicio.com.br/conhecendo-o-esp32.

Rai, P. and Rehman, M. (2019). Esp32 based smart surveillance system. In *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pages 1–3. IEEE.

Santos, J. W., Junior, L., et al. (2019). Sistema de automatização residencial de baixo custo controlado pelo microcontrolador esp32 e monitorado via smartphone. B.S. thesis, Universidade Tecnológica Federal do Paraná.

Sharp, A. and Vagapov, Y. (2017). Comparative analysis and practical implementation of the esp32 microcontroller module for the internet of things.

TAITO (1983). Ice cold beer. In TAITO, editor, *ICE COLD BEER: Operation, Maintenance and Service Manual*, pages 2–3. TAITO.