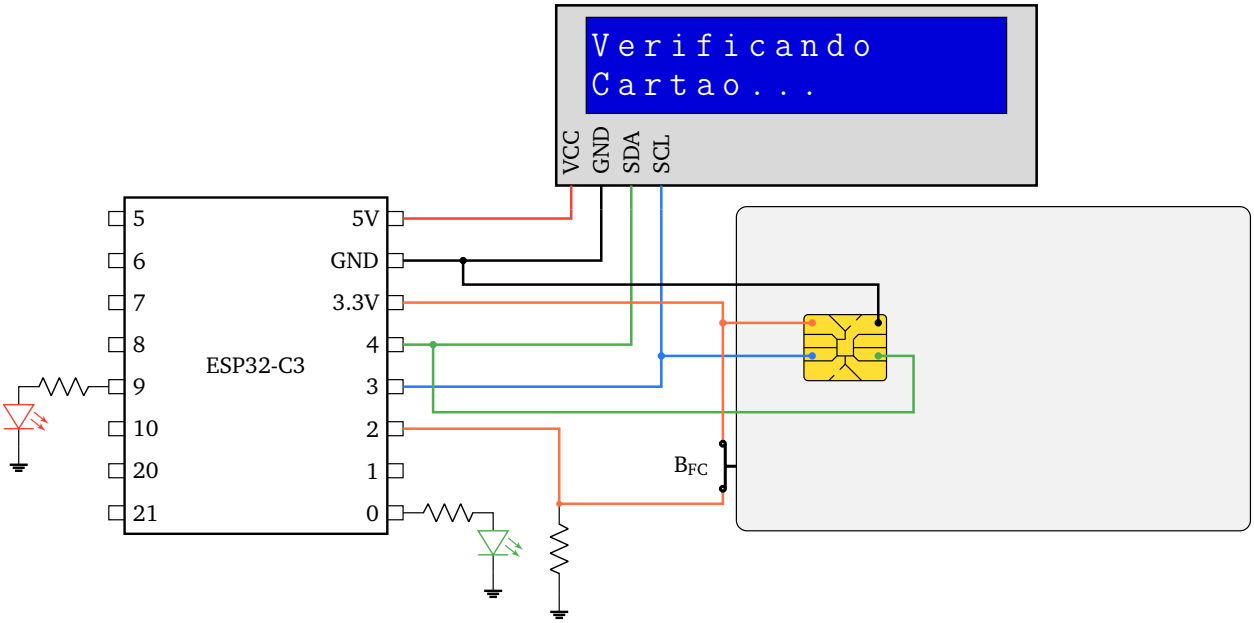
UFSC – EEL7030 Microprocessadores

Projeto 2

Controle de Acesso por *SmartCard*

Um hotel utiliza um sistema informatizado para controle de acesso aos seus quartos utilizando um *SmartCard* reprogramável. Sempre que um hóspede realiza um *check-in*, um ID aleatório de 16 caracteres é gerado e associado ao número do quarto. Esse ID é registrado em um banco de dados do hotel e gravado nas duas primeiras páginas (16 primeiros bytes) de um *SmartCard* EEPROM. Para facilitar a instalação das fechaduras, é proposto um sistema Wi-Fi que se conecte à rede do hotel e realize requisições HTTP POST para um servidor, que irá verificar o ID do cartão, o quarto associado e enviará uma resposta para decisão de abertura da porta.

O microcontrolador ESP32-C3 foi escolhido para atuar como a central de processamento da fechadura. A interação com o hóspede ocorrerá por meio do *SmartCard* EEPROM, que será inserido em um leitor. O circuito abaixo demonstra o *hardware* proposto para a fase de testes do sistema.



A primeira tarefa do ESP32-C3 é justamente a leitura desses cartões via protocolo I2C. Um botão de fim de curso (B_{FC}) é acionado sempre que um cartão é inserido no leitor. Quando um cartão for inserido, o ESP32-C3 deve extrair o ID das duas primeiras páginas da memória EEPROM (endereços base `0x00` e `0x08`, respectivamente). Uma vez lido o ID do cartão, ele deve ser enviado junto com o número do quarto via rede Wi-Fi do hotel para um servidor central utilizando uma requisição HTTP POST. O servidor irá verificar a validade do cartão em relação ao quarto. Se o servidor confirmar que o ID é válido e o hóspede tem acesso liberado ao quarto, ele enviará uma resposta positiva ao ESP32-C3. Em caso contrário, enviará uma resposta negativa. Com o resultado da validação, o ESP32-C3 deve decidir sobre a liberação do acesso. Um LED vermelho, conectado ao pino GPIO 9 atua como simulador do sistema de liberação da porta, devendo permanecer aceso por $2\,\mathrm{s}$ para abrir a fechadura.

Sempre que uma tentativa de acesso for realizada, independente de seu resultado, um *timestamp* de 64 bits (8 bytes) deve ser salvo na página 4 do *SmartCard* (endereço base `0x18`) com a data e

Desenvolvido por
Eduardo Augusto Bezerra e Raul Guedert
com base no Espressif IoT Development Framework <idf.espressif.com>

hora atuais da tentativa de acesso.

Para garantir a verificação dos estados do sistema, um *display* LCD pode ser conectado à fechadura, também via I2C. O *display* LCD deve exibir os estados desde o momento em que o ESP32-C3 é iniciado e tenta se conectar à rede Wi-Fi (“Conectando ao Wi-Fi”), passando pela espera por um cartão (“Aguardando Cartão”), a etapa de verificação (“Verificando Cartão”), até o resultado final, seja ele “Acesso Liberado” ou “Acesso Negado”.

Adicionalmente, um LED Verde conectado à porta GPIO 0 realiza a exibição do *status* através de sinais intermitentes. O LED pisca lentamente enquanto o ESP32-C3 não estiver conectado; permanece aceso em caso de conexão; pisca uma única vez quando o processo de leitura do cartão for iniciado; pisca duas vezes em caso de liberação do acesso e seis vezes quando o acesso for negado.

Para alcançar o objetivo do projeto, você deve escrever um código em C para o ESP32-C3 que atenda aos seguintes critérios:

- O código deve ser implementado de forma legível, com nomes de variáveis claras, **utilizando os conceitos abordados na disciplina**. Utilize o componente disponibilizado no Moodle para o controle do *display* LCD via protocolo I2C.
- O seu código deve estar **totalmente compreendido em um único arquivo .c** (com exceção do componente disponibilizado). Sempre que possível, **modularize seu código em funções auxiliares** que executam ações específicas.
- O sistema deve ser implementado **utilizando o sistema operacional FreeRTOS** para gerenciamento das tarefas. Utilize as ferramentas vistas nas aulas para realizar a comunicação e sinalização entre o sistema e as tarefas.
- O processo de leitura deve ser realizado sempre que um cartão for inserido utilizando uma interrupção relacionada ao botão de fim de curso (B_{FC}). Quando ocorrer a interrupção, utilize um *debouce* de $100\,\mathrm{ms}$, verificando o estado do botão após esse período. A leitura só deve ser realizada caso o botão ainda esteja pressionado.
- Uma vez que o ID do *SmartCard* for lido, ele deve ser enviado por HTTP POST para o servidor central. O corpo da requisição (*request body*) deve conter o ID e o número do quarto ao qual o ESP32-C3 está relacionado. Informações sobre o servidor, a requisição esperada e as respostas possíveis podem ser encontradas na documentação disponível no Moodle.
- Sempre que uma leitura for realizada, o *timestamp* atual (utilizando *timestamp* POSIX 64 bits) deve ser armazenado na página 4 (endereço base `0x18`) do cartão.
- As informações do *SmartCard* e de seu funcionamento podem ser encontradas no Roteiro 10 e no *Datasheet* do componente, disponíveis no Moodle.
- O *display* deve exibir o status atual do sistema.
  - “Conectando ao Wi-Fi” enquanto o processo de conexão estiver sendo realizado;
  - “Aguardando Cartao” enquanto o sistema estiver aguardando um cartão ser inserido;
  - “Verificando Cartao” enquanto o sistema estiver verificando a validade do cartão, isto é, lendo o valor do ID e realizando a requisição para o servidor;
  - “Acesso Liberado” quando o servidor retornar uma resposta positiva quanto à validação.
  - “Acesso Negado” quando o servidor retornar uma resposta negativa quanto à validação.

- O LED conectado ao pino GPIO 0 (LED Verde) é utilizado para representar o *status* do sistema com sinais intermitentes:
  - Piscar de forma intermitente lentamente ($1\,\mathrm{s}$ de intervalo) quando o sistema estiver desconectado do servidor.
  - Permanecer aceso quando o sistema estiver conectado ao servidor e aguardando um cartão ser inserido.
  - Piscar rapidamente (intervalo de $200\,\mathrm{ms}$) uma única vez quando um cartão for inserido e sua leitura tiver iniciado.
  - Piscar rapidamente (intervalo de $200\,\mathrm{ms}$) duas vezes quando o acesso for liberado.
  - Piscar rapidamente (intervalo de $200\,\mathrm{ms}$) seis vezes quando o acesso for negado.
- O LED conectado ao pino GPIO 9 (LED Vermelho) é utilizado para simular a o sistema eletrônico da fechadura devendo permanecer aceso por $2\,\mathrm{s}$ para abrir a fechadura.
- O sistema proposto deve realizar o controle de acesso do **Quarto 101**.

Critérios de Avaliação

O projeto deve ser feito de forma **individual**. A nota do projeto será calculada de acordo com a seguinte fórmula:

$$\mathrm{NP} = \frac{A}{10}\left(\frac{8C}{10} + \frac{2D}{10}\right)$$

Onde:

- A é a nota da arguição, podendo variar entre 0 e 10.
  - A arguição será realizada na última aula reservada ao projeto (ver cronograma), **exclusivamente no ambiente de sala de aula**, durante o horário da aula. A nota será atribuída de forma que 0 representa nenhum domínio sobre o código desenvolvido e 10 representa domínio completo sobre o código desenvolvido.
  - O aluno será arguido com perguntas sobre a implementação do projeto e suas funcionalidades, devendo respondê-las em, no máximo, 15 minutos.
  - Durante a arguição, será permitido acesso **apenas** ao código desenvolvido. Dessa forma, é vedada a pesquisa em qualquer ambiente (Google, IAs, etc). Se observado o uso de qualquer ambiente de pesquisa, a nota da arguição será considerada 0.
- C é a nota do código, podendo variar entre 0 e 10.
  - O código será avaliado quanto a sua legibilidade, adequação à estrutura padrão de códigos em C e o desenvolvimento das funcionalidades do projeto. **Serão considerados desenvolvimentos parciais das funcionalidades**.
  - As funcionalidades avaliadas serão:
    - Conexão ao Wi-Fi em Modo STA (1,00 ponto);
    - Interrupção do Botão de Fim de Curso e *Debounce* (1,00 ponto);

    - Leitura do ID do SmartCard *EEPROM* (1,00 ponto);
    - Configuração do SNTP e escrita do *timestamp* no *SmartCard* (1,50 ponto);
    - Requisição HTTP POST com as informações necessárias (1,50 ponto);
    - Tratamento da resposta e liberação da porta (LED Vermelho) (1,00 ponto);
    - Lógica para exibição do *status* com o LED verde (1,50 ponto);
    - Configuração do display LCD e exibição do *status* (1,50 pontos);
- D é a nota da documentação, podendo variar entre 0 e 10.

Importante! Códigos que sejam copiados entre os alunos terão sua nota zerada (todos os alunos envolvidos).

Documentação

O código deve ser documentado utilizando o Doxygen. Adicionalmente, você deverá gerar um arquivo .PDF contendo a documentação do seu projeto. O arquivo é gerado automaticamente pelo doxygen se este foi corretamente implementado. Acesse o tutorial disponível no Moodle para detalhes de como realizar esse processo.

Entrega e Arguição

A entrega do Projeto deverá ser realizada até a data limite da tarefa disponível no Moodle da turma. Deverão ser enviados o arquivo main.c (arquivo principal) contendo todo o código e o arquivo de documentação doc.pdf (gerado a partir do doxygen). A arguição será realizada exclusivamente em ambiente de sala de aula, na última semana reservada ao projeto, conforme consta no Plano de Ensino.

Dicas para Implementação

- Esboce um diagrama do seu sistema com as tarefas principais a serem realizadas e quais seus eventos de bloqueio e desbloqueio. Isso ajuda a organizar a sua implementação, uma vez que você terá um panorama geral do sistema.
- Utilize grupos de eventos, semáforos ou filas para sinalizar informações sobre ações que devem ser tomadas pelas tarefas com base em informações do sistema.
- Por exemplo, para implementação da leitura do cartão com *debounce*, você pode criar uma tarefa que ficará bloqueada até que uma interrupção ocorra no pino conectado ao botão de fim de curso. Sempre que a interrupção ocorrer, a ISR desabilita a interrupção no pino e desbloqueia uma tarefa responsável pela leitura do cartão. Ao desbloquear a tarefa, realize um *delay* de $100\,\mathrm{ms}$, ativando a interrupção novamente após esse período. Em seguida, verifique o estado do pino para evitar acionamentos espúrios. Caso o botão ainda esteja pressionado, realize a leitura do cartão.
- A resposta do Servidor HTTP gera diferentes *status* HTTP, sendo o *status* 200 para o acesso liberado. Ao invés de avaliar o corpo da resposta, você pode avaliar apenas o *status* da resposta com a função `esp_http_client_get_status_code()`, tomando uma ação baseado no *status* retornado.