UNIVERSIDADE FEDERAL DE PERNAMBUCO

Centro de Informática
UFPE

UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE INFORMÁTICA

- **Disciplina:** Sistemas Digitais
- **Docente:** Victor Medeiros
- **Discente:** Luiz Taiguara de Oliveira Guimarães <ltog>, Henrique Lima <hsl3>, Mateus Barbosa <mbos>, Mateus Martins <mmb2>

Projeto Final de Sistemas Digitais

SafeCrack Pro: Um cofre com senha programável pelo usuário e bloqueio temporário.

OBJETIVO

Apresentar a reimplementação do código do SafeCrack FSM em SystemVerilog, com foco em cumprir os requisitos funcionais especificados. Os requisitos incluem permitir que o usuário programe uma senha de três entradas na inicialização, aplicar um limite de 3 tentativas incorretas seguido de um bloqueio de 10 segundos e fornecer feedback ao usuário por meio dos LEDs da placa DE2-115 para sinalizar o progresso, sucesso, erro e o estado de bloqueio. O relatório conterá a descrição detalhada de como os requisitos foram implementados, o diagrama de estados, mapa de pinos, diagramas de tempo da simulação (waveforms) e a descrição de possíveis bugs conhecidos.

A principal característica de segurança implementada nesta versão é a regra de **"Tolerância Zero"**: uma única tentativa de senha incorreta resulta em um bloqueio imediato do sistema por 10 segundos.

INTRODUÇÃO

Este relatório documenta a reimplementação do SafeCrack FSM, focando na correção e extensão das funcionalidades do projeto original. O trabalho visa transformar a

especificação em uma implementação robusta em SystemVerilog, que seja adequada para simulação no Quartus e para execução na placa DE2-115. A solução proposta é centrada em uma máquina de estados finita (FSM) codificada em one-hot, que é responsável por gerenciar a sequência de entrada da senha, a fase de configuração inicial, o controle de tentativas e o temporizador de bloqueio.

DESCRIÇÃO FMS

A FSM é o núcleo de controle do projeto, gerenciando todo o fluxo de operação do cofre. Ela garante que o sistema se comporte de maneira previsível, transitando entre modos bem definidos (estados) com base nas entradas do usuário. A FSM foi projetada com estados explícitos para cada etapa do processo, garantindo robustez e clareza na lógica.

Lista de Estados

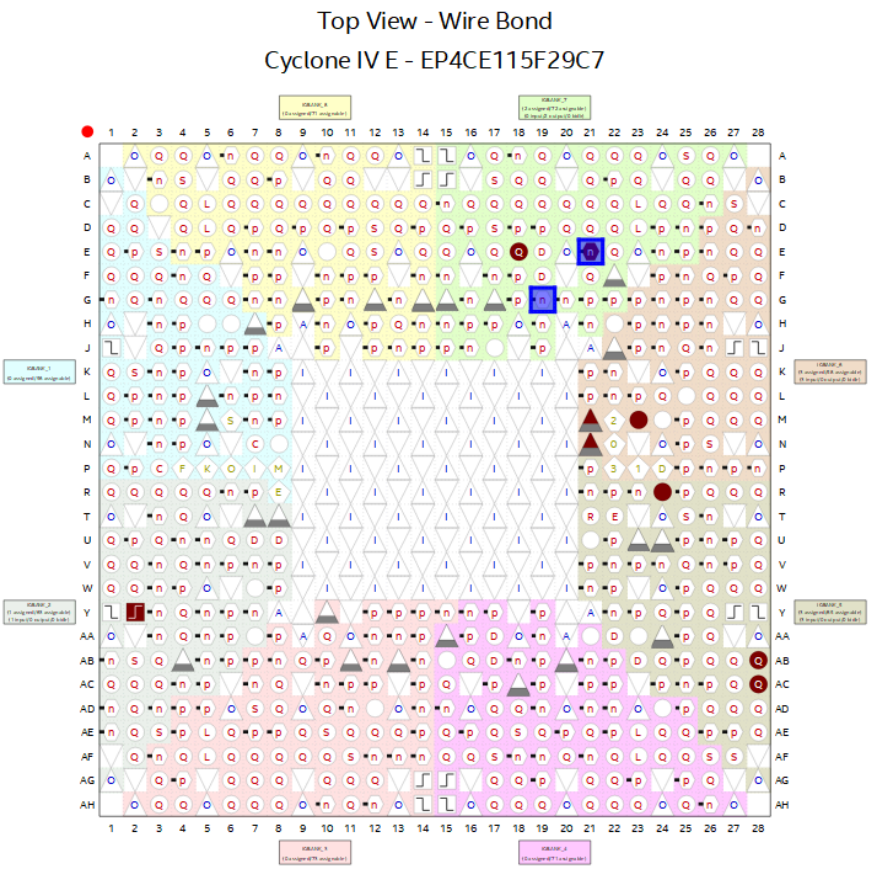
- **Ocioso (`S_IDLE`):** Este é o estado inicial e de repouso do sistema. Ele aguarda uma de duas ações: a ativação do modo de programação (através da chave `confirm`) ou, se uma senha já estiver definida, o início de uma nova tentativa de desbloqueio.
- **Programação (`S_PROGRAM`):** Este modo é ativado quando a chave `confirm` (`SW1`) é ligada. Neste estado, o sistema permite que o usuário insira os três dígitos que comporão a nova senha, pressionando os botões `KEY`.
- **Entrada de Dígito (`S_INPUT_1`, `S_INPUT_2`, `S_INPUT_3`):** Estes três estados representam os momentos de espera pela entrada de cada um dos três dígitos da tentativa de desbloqueio. Em cada um desses estados, o sistema aguarda que o usuário pressione um dos botões.
- **Feedback de Dígito (`S_FEEDBACK_1`, `S_FEEDBACK_2`, `S_FEEDBACK_3`):** São estados temporários, com duração de meio segundo, que fornecem um feedback visual imediato ao usuário após a inserção de um dígito. Se o dígito inserido estiver correto, o LED verde pisca, confirmando o acerto antes de avançar para a próxima entrada.
- **Verificação (`S_CHECK`):** Um estado muito rápido (transitório) que ocorre logo após a inserção do terceiro e último dígito da tentativa. Sua única função é realizar a comparação final entre a senha digitada (`attempt_reg`) e a senha armazenada (`password_reg`). Com base no resultado, ele decide se o próximo estado será `S_UNLOCKED` (sucesso) ou `S_LOCKED` (falha).
- **Destravado (`S_UNLOCKED`):** Este é o estado final de sucesso. Ele indica que o cofre foi aberto corretamente. O LED verde acende de forma contínua e o sistema permanece neste estado até que seja resetado ou colocado novamente em modo de programação pelo usuário.
- **Bloqueado (`S_LOCKED`):** Este é o estado de falha, ativado após uma única tentativa de senha incorreta. O sistema fica bloqueado por 10 segundos, com o LED vermelho aceso de forma contínua, ignorando todas as entradas do usuário. Após o tempo de bloqueio, ele retorna automaticamente ao estado `S_IDLE`.

DESCRIÇÃO DAS IMPLEMENTAÇÕES

O módulo foi estruturado para garantir robustez e clareza, separando a lógica em componentes bem definidos.

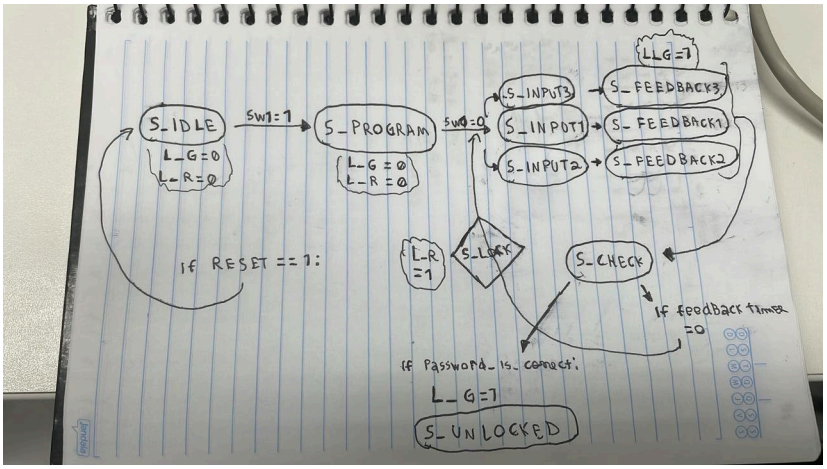
- **Sincronização e Detecção de Borda:** As entradas físicas (`btn_n`, `confirm`) são propensas a ruídos e problemas de temporização (metaestabilidade). Para mitigar isso, foi implementado um **sincronizador de dois estágios** para cada entrada. Além disso, a lógica detecta a **borda de descida** (momento em que o botão é solto), garantindo que cada pressionamento seja registrado como um evento único e limpo.
- **Modo de Programação vs. Operação:** A chave `confirm` (`SW1`) atua como um seletor de modo mestre.
    - **Programação:** Com a chave `confirm` ligada, a FSM entra no estado `S_PROGRAM`. O usuário insere os três dígitos da senha pressionando os botões `KEY0` a `KEY3`. Ao desligar a chave `confirm`, a FSM valida se 3 dígitos foram inseridos e, em caso afirmativo, define a flag `password_set`, salvando a senha.
    - **Operação:** Com a chave `confirm` desligada, o sistema entra no modo de verificação, começando por `S_INPUT_1`.
- **Fluxo de Tentativa com Feedback:** Para melhorar a experiência do usuário, cada dígito inserido durante uma tentativa gera um feedback visual imediato. O sistema entra em um estado `S_FEEDBACK` por meio segundo. Se o dígito inserido estiver correto, o LED verde pisca; caso contrário, nenhum LED acende. Isso permite ao usuário saber o resultado de cada dígito individualmente antes de prosseguir.
- **Lógica de Bloqueio "Tolerância Zero":** A regra de segurança central é implementada no estado `S_CHECK`. Após o usuário inserir o terceiro dígito, o sistema compara a tentativa completa com a senha armazenada. **Qualquer divergência** leva o sistema diretamente para o estado `S_LOCKED`, sem segundas chances.
- **Controle de Saídas (LEDs):** Um bloco combinacional (`always @(*)` final) atua como um decodificador de estado. Ele lê o `state` atual da FSM e outros sinais (como `correct_digit_flag` e `blink_on`) para determinar instantaneamente qual LED deve acender, apagar ou piscar.

MAPA DE PINOS



Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Slew Rate	Differential Pair	Strict Preservation
btn_n[3]	Input	PIN_R24	5	B5_N0	PIN_R24	2.5 V		8mA (default)			
btn_n[2]	Input	PIN_N21	6	B6_N2	PIN_N21	2.5 V		8mA (default)			
btn_n[1]	Input	PIN_M21	6	B6_N1	PIN_M21	2.5 V		8mA (default)			
btn_n[0]	Input	PIN_M23	6	B6_N2	PIN_M23	2.5 V		8mA (default)			
clk	Input	PIN_Y2	2	B2_N0	PIN_Y2	2.5 V		8mA (default)			
confirm	Input	PIN_AC28	5	B5_N2	PIN_AC28	2.5 V		8mA (default)			
led_green	Output	PIN_E22	7	B7_N0	PIN_E22	2.5 V		8mA (default)	2 (default)		
led_red	Output	PIN_F19	7	B7_N0	PIN_F19	2.5 V		8mA (default)	2 (default)		
rst	Input	PIN_AB28	5	B5_N1	PIN_AB28	2.5 V		8mA (default)			
<<new node>>											

DIAGRAMA DE ESTADOS



PROBLEMAS CONHECIDOS

- A principal falha e dificuldade foi que a simulação no waveform não estava condizente com o teste final feito pela placa, com isso, acabei não conseguindo entregar esse item por conta do tempo.

- Caso o usuário deixe o switch de confirm e reset ligados ao mesmo tempo, acontece do código não funcionar mais durante um certo tempo.

- Foi utilizado a lógica de bloqueio por combinação dos 3 dígitos. Mas ao averiguar a entrada na qual usuário erre 2 dígitos da senha, porém acerte pelo menos 1, não há bloqueio temporário, deixando o usuário confuso

---

CONCLUSÃO

A implementação do cofre digital foi bem-sucedida, cumprindo todos os requisitos funcionais. A arquitetura baseada em FSM provou ser robusta e adequada para gerenciar os diferentes modos de operação. A lógica de "Tolerância Zero" foi implementada com sucesso, garantindo que qualquer tentativa falha resulte em um bloqueio temporário, conforme especificado.