Pontifícia Universidade Católica do Rio Grande do Sul

Escola Politécnica

Sistemas Digitais | 98G02-04

Prof. Iaçanã Ianiski Weber

Trabalho II
Sistema Antifurto Automotivo

26 de abril de 2024

Deadline: 17/05/2024 - 17:30 - Moodle

1 Objetivo

Projetar um sistema digital (Sistema Antifurto) com circuito sequencial (Máquina de Estados Finitos - FSM) baseado em um conjunto de especificações. O sistema deve ser implementado em Verilog e prototipado em FPGA, em grupos de no máximo dois alunos.

2 Apresentação

Tenha o seguinte disponível durante a apresentação:

- Diagrama de Transição de Estados da FSM Antifurto.
- Código Verilog (disponível na tela do seu computador).
- O estado da FSM (em hexadecimal) e a contagem regressiva do temporizador devem ser exibidos o tempo todo no display hexadecimal.
- O sistema completo funcionando em FPGA. Seja capaz de demonstrar o seguinte:
  - Carro funcionando, ignição ligada. Desligue a chave, abra uma ou ambas as portas. Quando ambas as portas estão fechadas, o temporizador `T_ARM_DELAY` inicia.
  - Quando qualquer temporizador está em funcionamento, o estado da FSM e a contagem regressiva do temporizador devem ser exibidos.
  - Se alguma porta for aberta antes do `T_ARM_DELAY` expirar, o `T_ARM_DELAY` reinicia após as portas serem fechadas novamente. (Isso permite que o proprietário recupere itens sem usar a chave.)
  - No estado armado, o indicador de status deve piscar com um período de dois segundos; a sirene está desligada.
  - Após o carro ser armado, abrir a porta do motorista ou do passageiro iniciará o `T_DRIVER_DELAY` ou `T_PASSENGER_DELAY`. A luz indicadora de status deve permanecer constantemente ligada; a sirene está desligada. Quando o temporizador expirar, o alarme soará continuamente enquanto uma das portas estiver aberta. O `T_ALARM_ON` só começa se ambas as portas estiverem fechadas. Após o `T_ALARM_ON` expirar, o alarme retorna ao estado armado.
  - Com o alarme soando, a luz indicadora de status e a sirene devem permanecer constantemente ligadas. A sirene deve alternar os LEDs RGBs entre azul e vermelho com frequência de 2Hz. *Sirenes mais sofisticadas são incentivadas e recompensadas.*

  - Operação da bomba de combustível: quando a ignição está desligada, a energia para a bomba de combustível é cortada. A energia só é restaurada quando primeiro a ignição é ligada e então o motorista pressiona um interruptor oculto e o pedal de freio. A energia é então travada até a ignição ser desligada novamente.
  - Os parâmetros de tempo podem ser programados para qualquer valor. Sempre que um parâmetro for reprogramado, a FSM deve ser reiniciada para o estado ARMADO (após o que pode imediatamente transicionar para outro estado, dependendo das entradas dos sensores).
  - Com todos os valores de tempo definidos como zero, as portas devem comportar-se como um controle ligado/desligado para a sirene.

- Esteja pronto para explicar o fluxo de projeto do seu sistema.

- Você deve entregar o projeto compactado (.zip/.tar.gz) no Moodle. O Verilog deve incluir **código legível com comentários e indentação consistente**, uso correto de atribuições bloqueantes e não bloqueantes, uso de `default` em casos de atribuições combinacionais, uso de instruções de `parameter/define` para nome simbólico e constantes. Não deve haver longas instruções `if` aninhadas.

3 Sistema Antifurto

Uma recém-formada acabou de concluir um IPO muito bem-sucedido de sua *startup* e comemorou comprando um novo Porsche. Embora o carro tenha um sistema antifurto incorporado, ela está preocupada, pois este é uma unidade padrão de fábrica e muitas pessoas sabem como desabilitá-lo. Portanto, ela está procurando alguém para projetar e construir um sistema com alguns recursos de segurança ocultos que apenas vocês dois conhecerão! Seu trabalho é criar um protótipo funcional.

Neste trabalho, você implementará um sistema antifurto que usa várias FSMs (Máquina de Estados Finitos) interagindo para processar entradas de sensores e gerar os sinais de controle de atuadores apropriados.

4 Procedimento

Leia este documento para entender a funcionalidade desejada, determine sua abordagem e proponha um design. Embora não seja obrigatório, pode ser útil apresentar uma proposta do seu design ao professor. Isso ajudará a identificar quaisquer erros importantes no início do processo. Prototipe seu design usando o FPGA. Esteja preparado para demonstrar as funcionalidades de sua implementação listadas acima.

4.1 Descrição do Sistema Antifurto

Como sua cliente está totalmente focada em sua startup, ela quer um sistema antifurto altamente automatizado. O sistema é armado automaticamente depois que ela desliga a ignição, sai do carro (ou seja, a porta do motorista foi aberta e fechada) e passa o T_ARM_DELAY. Se houver um passageiro e ambas as portas, do motorista e do passageiro, estiverem abertas, o sistema se arma depois que todas as portas forem fechadas e o T_ARM_DELAY passar; esse atraso é reiniciado se uma porta for aberta e fechada novamente antes do alarme ser armado.

Uma vez que o sistema tenha sido armado, ao abrir a porta do motorista o sistema inicia uma contagem regressiva. Se a ignição não for ligada dentro do intervalo da contagem regressiva (T_DRIVER_DELAY), a sirene será acionada. A sirene permanece ligada enquanto a porta estiver aberta e por um intervalo adicional (T_ALARM_ON) após a porta fechar, momento em que o sistema é redefinido para o estado armado, mas silencioso. Se a ignição for ligada dentro do intervalo da contagem regressiva, o sistema é desarmado.

Sempre um exemplo de cortesia, a sua cliente abre a porta do passageiro primeiro se estiver transportando um convidado. Quando a porta do passageiro é aberta primeiro, um atraso separado, presumivelmente mais longo, (T_PASSENGER_DELAY) é usado para o intervalo da contagem regressiva, dando a ela mais tempo para contornar o carro até a porta do motorista e inserir a chave na ignição para desarmar o sistema.

Há um LED indicador de status no painel. Ele pisca com um período de dois segundos quando o sistema está armado. Ele é constantemente iluminado quando o sistema está na contagem regressiva à espera que a ignição seja ligada ou quando a sirene está ativada. O LED fica desligado quando o sistema está desarmado.

Até agora, todas essas são funcionalidades comuns de alarme. Mas você está preocupado que um ladrão experiente possa desativar a sirene e simplesmente levar o carro. Portanto, você adicionou um impedimento secreto adicional - controle de energia para a bomba de combustível. Quando a ignição está desligada, a energia para a bomba de combustível é cortada. A energia só é restaurada quando a ignição é ligada e, em seguida, o motorista pressiona simultaneamente um interruptor oculto e o pedal do freio. A energia então permanece ligada até que a ignição seja desligada novamente.

O diagrama abaixo lista todos os sensores (entradas) e atuadores (saídas) conectados ao sistema:

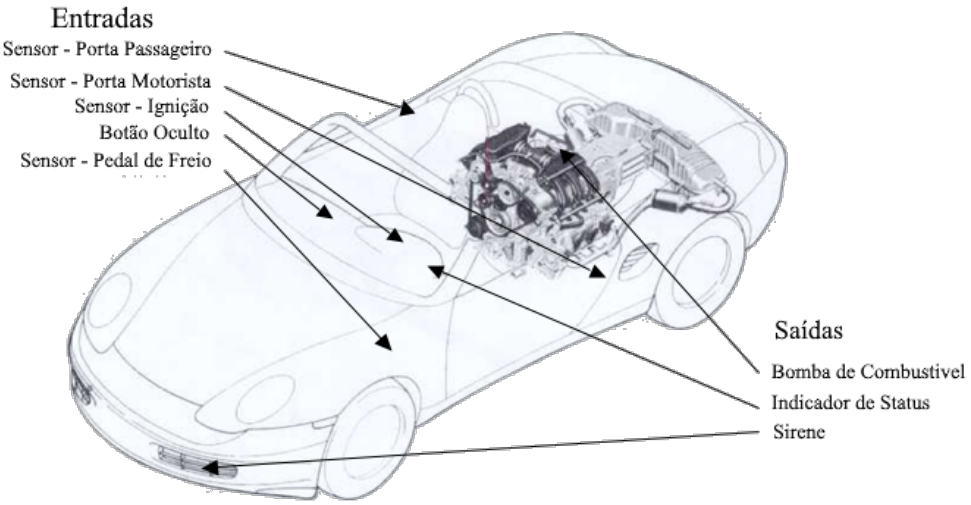

Figura 1: Diagrama do Carro indicando sensores (entradas) e atuadores (saídas).

Os tempos do sistema são baseados em quatro parâmetros (**em segundos**):

- O atraso entre sair do carro e o armamento do alarme (`T_ARM_DELAY`);
- A duração da contagem regressiva antes do alarme soar após abrir a porta do motorista (`T_DRIVER_DELAY`) ou a porta do passageiro (`T_PASSENGER_DELAY`);
- A duração do som da sirene (`T_ALARM_ON`).

O valor padrão para cada parâmetro está listado na tabela abaixo, mas cada um pode ser configurado para outros valores usando os sinais `Time_Param_Sel`, `Time_Value`, e `Reprogram`. Os interruptores `Time_Param_Sel` especificam o número do parâmetro a ser alterado. Os interruptores `Time_Value` são um valor de 4 bits representando o valor a ser programado — um valor em segundos entre 0 e 15. Ao pressionar o botão `Reprogram`, o sistema é instruído a definir o parâmetro selecionado atualmente para o `Time_Value`. Note que seu sistema deve funcionar corretamente mesmo que um ou mais dos parâmetros seja configurado para 0.

Interval Name	Symbol	Parameter Number	Default Time (sec)	Time Value
Timer para Armar	T_ARM_DELAY	00	6	0110
Countdown - Porta do Motorista	T_DRIVER_DELAY	01	8	1000
Countdown - Porta do Passageiro	T_PASSENGER_DELAY	10	15	1111
Timer da Sirene	T_ALARM_ON	11	10	1010

4.2 Implementação e Descrição dos Blocos

O diagrama a seguir ilustra uma possível organização do seu projeto em módulos:

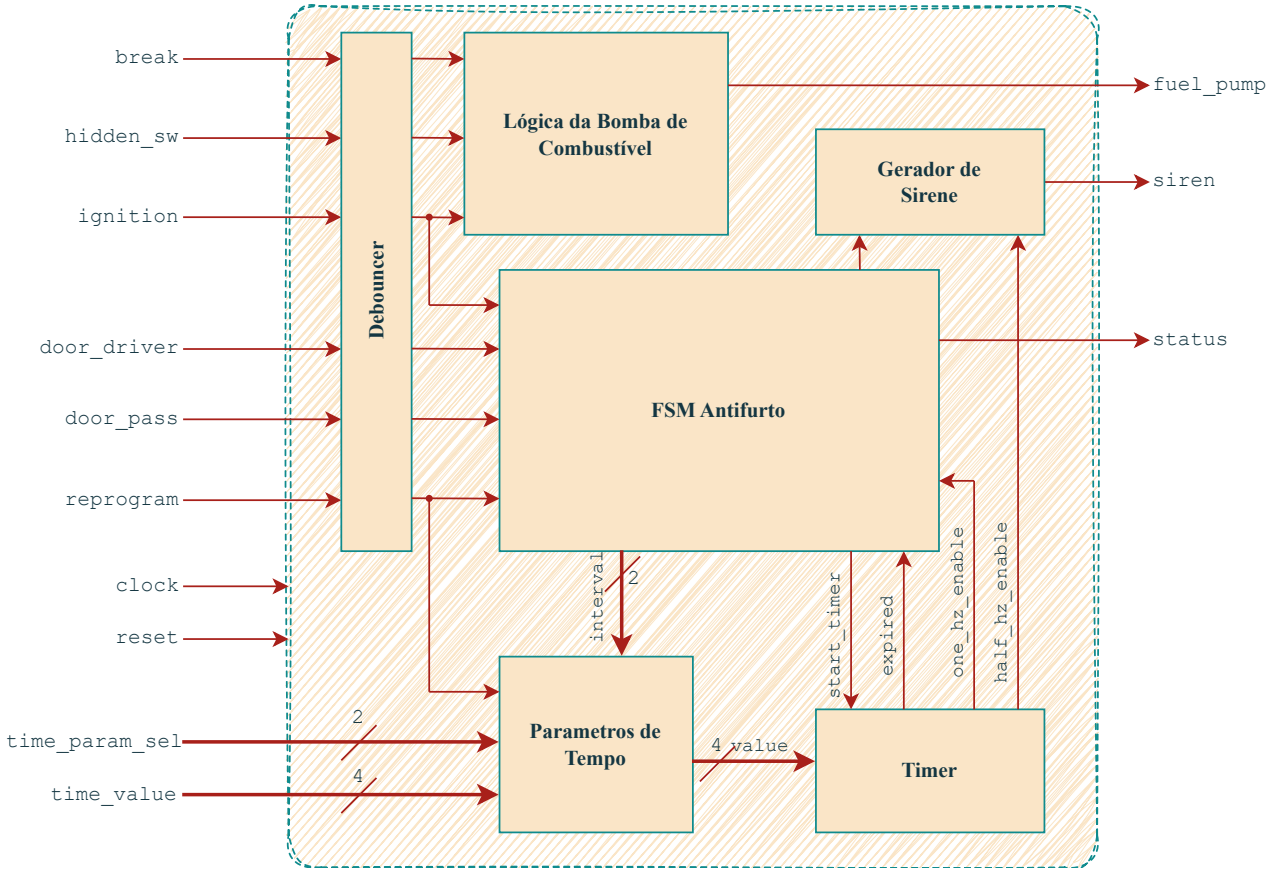

Figura 2: Diagrama de Blocos do Sistema Antifurto

Você deve implementar este trabalho criando cada módulo individualmente e, em seguida, instanciando e conectando os módulos no módulo principal *toplevel* `t2_main.v`. Depois, compile sua implementação usando o Vivado, faça o download para o FPGA e demonstre seu funcionamento.

Use os seguintes dispositivos FPGA para os diversos sensores e atuadores:

- Botão oculto - BTNU(M18)
- Sensor do Pedal de Freio - BTND(P18)
- Sensor Porta do Motorista - BTNL(P17)
- Sensor Porta do Motorista - BTNR(M17)
- Reset do Sistema - SW[15](V10)

- Sensor de Ignição - SW[14](U11)
- Time_param_sel - SW[5:4](T18-R17)
- Time_value - SW[3:0](R15-M13-L16-J15)
- Reprogram - BTNC(N17)
- LED de Status - LED[0](H17)
- Energia Bomba de Combustivel - LED[1](K15)
- Sirene - RGB(N15-M16-R12)

Aqui está uma descrição mais detalhada de cada módulo:

4.2.1 Debouncer

Sua máquina de estados é controlada por várias entradas assíncronas que podem ser alteradas pelo usuário a qualquer momento, potencialmente criando um problema de metastabilidade (*falaremos mais sobre esse tópico ainda nesta disciplina*) nos registros de estado se uma das entradas mudar próximo demais de uma borda de subida do relógio. Em geral, as entradas assíncronas precisam ser sincronizadas com o relógio interno antes de poderem ser utilizadas pela lógica interna. Um segundo problema surge do "quique"mecânico inerente aos interruptores: à medida que um contato metálico abre e fecha, ele pode quicar algumas vezes, criando uma sequência de transições ligado/desligado em rápida sucessão. Portanto, você precisa usar circuitos de *debouncing* para filtrar essas transições indesejadas. Para este trabalho o debounder exige que uma transição de entrada seja estável por 0,01s antes de produzir uma transição em sua saída. Este módulo, por acaso, produz uma saída síncrona, então um sincronizador separado não é necessário. Você deve usar uma instância do módulo de debouncing para filtrar quaisquer entradas que você use em seu design. Está disponível para vocês utilizarem nas entradas, uma implementação de um *debouncer* no material de apoio do trabalho.

4.2.2 Parâmetros de Tempo

O módulo de parâmetros de tempo armazena os quatro diferentes valores de parâmetros de tempo. O módulo funciona como uma memória de 4 locais que é inicializada com valores padrão ao ligar, mas pode ser reprogramada pelo usuário a qualquer momento. Usando o sinal de `interval` de 2 bits, o FSM Anti-furto seleciona um dos quatro parâmetros para ser usado pelo módulo de Timer.

Ao ligar, os parâmetros devem ser definidos para os valores padrão especificados na tabela acima. No entanto, o usuário pode modificar qualquer um dos valores manipulando o `time_param_sel` (2 bits), `time_value` (4 bits) e `reprogram`. Sempre que um parâmetro é reprogramado, o FSM deve ser reiniciado para seu estado ARMADO (após o qual pode transitar imediatamente para outro estado dependendo das entradas de sensores).

4.2.3 Temporizador

O temporizador conta regressivamente o número de segundos especificado pelo módulo do Parâmetro de Tempo. Ele inicializa seu contador interno com o `value` especificado quando `start_timer` é acionado e decrementa o contador quando `one_hz_enable` é acionado. Quando o contador interno chega a zero, o sinal `expired` é acionado e a contagem regressiva é interrompida até que `start_timer` seja acionado novamente. Dentro do temporizador está o divisor que converte o relógio mestre de 100MHz em um sinal `one_hz_enable` que é acionado por apenas 1 ciclo a cada 100.000.000 ciclos (ou seja, uma vez por segundo). O `one_hz_enable` é usado pelo módulo FSM Antifurto e para fazer o LED piscar com um

período de dois segundos. O divisor precisa ser reiniciado quando `start_timer` é acionado para que o primeiro `one_hz_enable` após o início da contagem do temporizador ocorra um segundo inteiro após o temporizador ter sido iniciado.

4.2.4 FSM Antifurto

Esta máquina de estados finitos controla a sequência de operações do sistema. O sistema possui quatro modos principais de operação:

- **Armado**: O indicador de status deve piscar com um período de dois segundos; a sirene está desligada. Se a chave de ignição for ligada, passe para o modo Desarmado, caso contrário, quando uma porta for aberta, inicie a contagem regressiva apropriada e vá para o modo Acionado. Este é o estado que a FSM deve ter quando o sistema é ligado.
- **Acionado**: A luz indicadora de status deve ficar constantemente acesa; a sirene está desligada. Se a chave de ignição for ligada, vá para o modo Desarmado. Se a contagem regressiva expirar antes de a ignição ser ligada, vá para o modo de Ativar Alarme.
- **Ativar Alarme**: A luz indicadora de status e a sirene devem ficar constantemente ligadas. O alarme deve continuar até que `T_ALARM_ON` segundos após todas as portas estarem fechadas (neste ponto, vá para o modo Armado) ou a chave de ignição seja ligada (neste ponto, vá para o modo Desarmado).
- **Desarmado**: A luz indicadora de status e sirene devem estar desligadas. Aguarde até que a chave de ignição seja desligada, seguida pela abertura e fechamento da porta do motorista, depois de `T_ARM_DELAY` segundos, vá para o modo Armado.

Observe que mais de um estado FSM pode ser necessário para implementar a funcionalidade requerida de cada modo, ou seja, seu diagrama de transição de estado terá muito mais do que 4 estados.

4.2.5 Lógica da Bomba de Combustível

Esta é implementada por uma simples FSM que controla a energia para a bomba de combustível. A energia é desativada quando a chave de ignição é desligada e somente reativada quando a sequência apropriada de valores de sensores é recebida.

4.2.6 Gerador de Sirene

Este módulo gera o sinal que imita uma sirene utilizando os LEDs RGB da FPGA. Seu funcionamento se dá pela leitura de dois sinais: o `enable_siren` gerado pela FSM Antifurto e o `two_hz_enable` que é gerado pelo módulo temporizador. A sirene deve emitir luz sempre que o sinal `enable_siren` estiver alto, e toda vez que o sinal `two_hz_enable` ficar alto, a cor emitida deve trocar, alternado entre azul e vermelho.

Modificações do comportamento visual da sirene (de forma que fique mais complexa) será bem avaliada.

5 Para começar

Neste ponto, você já deve ser suficientemente proficiente para criar seus próprios arquivos de projeto. Para começar, faça o download do material de apoio, que contem o xdc padrão. Descomente as portas I/O necessárias no arquivo xdc. Crie um novo projeto e é útil começar fazendo um projeto simples, como controlar um LED com um interruptor, para verificar o processo. Pode ser útil ter interruptores e LEDs adicionais disponíveis para ajudar na depuração. Você precisará do display de sete segmentos para mostrar os valores do cronômetro. Nomeie seu arquivo principal como `t2_main.v`.

Um dos módulos mais difíceis para este laboratório é o módulo do cronômetro. O módulo do cronômetro deve ser capaz de lidar com qualquer valor de 0 a 15 segundos. Durante a contagem regressiva (ou se você optar pela contagem progressiva), ele deve ser capaz de ser reiniciado. Para um valor de zero (uma entrada válida), expire é acionado no próximo ciclo de relógio após `start_timer`. Recomendo usar um pulso para iniciar o cronômetro e um pulso de um ciclo para a saída `expire`. Outras abordagens funcionarão, mas com alguma dificuldade.

6 Recomendações

Você pode encurtar o tempo de desenvolvimento construindo um banco de testes para cada módulo do seu projeto e então um banco de testes geral para verificar o comportamento esperado do sistema em várias sequências de entradas que você espera testar num FPGA real. Uma vez que o sistema passe nestes testes, você pode prosseguir com a implementação no FPGA.

Para a apresentação, mostre uma simulação do seu temporizador sendo reiniciado no meio do processo e com um valor de tempo zero. Seu temporizador deve ser capaz de reiniciar enquanto conta e atuar conforme especificado com um valor de tempo zero. Acertar o temporizador é uma parte integral do alarme do carro.

Para depuração no FPGA, você tem várias opções. Use o display de sete segmentos de oito dígitos que implementamos em aula. Você precisará do display para mostrar que seu temporizador está funcionando corretamente. Use um dígito para mostrar o estado atual da FSM, outro dígito para indicar o parâmetro de tempo selecionado atualmente, mais um dígito para mostrar o valor atual do contador interno do módulo do temporizador, etc. Você vai querer usar esse módulo útil para mostrar informações de depuração para todos os seus projetos!