Produto 1: MI de Circuitos Digitais (TEC498)

Welton dos Santos Cerqueira[1], **João Lucas Silva Serafim Silva**[2]

[1]Engenharia de Computação –Universidade Estadual de Feira de Santana (UEFS)
Av. Transnordestina, S/N. Feira de Santana, Novo Horizonte – BA, Brasil – 44036-900.

[1]weltoncerqueira15@gmail.com, [2]joaolucasacademico@gmail.com

Abstract. *This report is a textual description regarding the development process of a structural verilog circuit project, as well as part of the documentation of the resulting product's features and specifications.*

Resumo. *Este relatório é uma descrição textual acerca do processo de desenvolvimento de um projeto de circuito em verilog estrutural, bem como parte da documentação pertencente às funcionalidades e especificações do produto resultante de tal projeto.*

1. Introdução

Uma classe de engenheiros de computação da Universidade Estadual de Feira de Santana (UEFS) foi contratada pela GoHo RSE (companhia fictícia), uma empresa multinacional do setor automobilístico que precisa de uma solução mais eficiente e segura para o acionamento de suas máquinas industriais.

Para cada equipe (divisões da classe), a GOHO RSE concebeu a missão de construir um protótipo para implementação de um painel de controle industrial em sua empresa. O protótipo deve ser capaz de atender rigorosamente aos requisitos, e deve faze-lo dentro dos prazos estabelecidos pela empresa.

Como base para a realização do projeto, foi adotada uma placa tipo FPGA (*Field Programmable Gate Array*), que faz parte de uma classe de dispositivos denominados CPLD (*Complex Programmable Logic Device*, inglês para "Dispositivo Lógico Programável Complexo"). A FPGA é um dispositivo lógico programável geralmente baseado em memória volátil que necessita de uma configuração essencialmente descrita por código HDL (*Hardware Description Language*, inglês para "linguagem de descrição de hardware").

Como requisito pré-estabelecido pela GoHo RSE, o protótipo deve conter duas interfaces de entrada (IE01 e IE02) com dois *push-buttons* (botões capazes de manter um estado arbitrário) e quatro chaves HH (interruptores "ligado/desligado")cada uma. Também deve permitir que haja execução em piloto automático, além de conter três interfaces de saída para exibir as funcionalidades.

Interface de saida 1 (ISO1): Matriz de leds Interface de saida 2 (IS02): Leds Interface de saida 3 (IS03): Display de 7 segmentos

Ao longo do relatório, será discorrido acerca do processo para a elaboração do projeto, bem como toda a lógica utilizada para a resolução do mesmo.

2. Discussões

Como é de costume no ciclo de desenvolvimento empregado pela GOHO RSE e pela UEFS, foram organizados debates entre as diversas equipes de desenvolvimento encarregas, paralelamente, do mesmo projeto, e por meio desses debates, foi possível realizar o levantamento de questões para a compreensão dos requisitos estabelecidos e a elaboração das melhores maneiras de entregar um produto que almeja cumprir tais requisitos.

Em primeira instância, foram criadas as tabela-verdades de “candidatos” a módulos, divididos apenas de forma a facilitar o entendimento do problema; A partir da lógica levantada pela elaboração das tabelas-verdade, foi possível formular a divisão definitiva correspondente a módulos físicos (não necessariamente refletidos na organização do dispositivo CPLD de prototipagem), a qual permitiu tanto a reutilização de código-fonte e hardware quanto uma abordagem metódica e eficiente quanto à simplificação dos circuitos de cada módulo.

O conjunto de tabelas-verdade iniciais do projeto pode ser visto por link disponível na seção de referências deste relatório.

Assim sendo, a solução apresentada neste relatório consiste na implementação de um módulo principal que instancia outros módulos, os quais são interligados por meio do uso de estruturas verilog denominadas ”fios” (as quais podem ou não corresponder a fios reais).

Dois dilemas encontrados durante o desenvolvimento do produto foram a execução simultânea de funcionalidades distintas por múltiplos perfis de usuário e a prioridade de execução de funcionalidades entre interfaces de entrada distintas.

Neste caso em específico, tais questões foram solucionadas por meio de módulos dedicados que recebem entrada referente ao tipo de usuário e funcionalidade executada por cada interface, e decidem se existe conflito de prioridade de qualquer tipo entre entes, utilizando retorno binário para sinalizar tanto a interface priorizada quanto a existência de conflito de funcionalidades, o qual dispensa a verificação da prioridade para ambas as interfaces.

É importante notar que o uso de mapas K (diagramas de *Veitch-Karnaugh*) para a simplificação de circuitos foi extremamente reduzida, visto que quase todos os módulos e submódulos que possuem um máximo de 5 entradas só aceitam uma combinação de entrada para valor lógico determinante (seja este ALTO ou BAIXO) por saída, dispensando assim considerações acerca de simplificações.

Módulos e submódulos que implementam circuitos em posse de 6 ou mais entradas distintas não são práticos e nem fáceis de serem simplificados pelo uso de ”mapas K”, estando portanto, neste momento, em grande parte além das capacidades individuais dos integrantes das equipes de desenvolvimento.

No entanto, tais circuitos ainda são passíveis de simplificação por meio do uso das propriedades de álgebra booleana que regem o funcionamento de todo e qualquer método alternativo de simplificação de circuitos.

Nas seções a seguir, o projeto e seus módulos serão descritos em maiores detalhes.

3. Descrição do produto: resolução e aprofundamento

3.1. Descrição da interface de usuário e funcionamento das interfaces de entrada

O produto é apresentado em uma placa de dispositivos eletrônicos de entrada e saída (interface de usuário), a qual está acoplada a um CPLD, marca *Altera*, série *MAX II*, modelo *EPM240T100C5*; O sistema criado pela integração desses dois componentes por meio do uso dos pinos da placa CPLD é denominado LEDS-CPLD.

A seguir, é possível conferir uma imagem representativa da interface de usuário, com destaque para as interfaces de entrada e suas subdivisões.

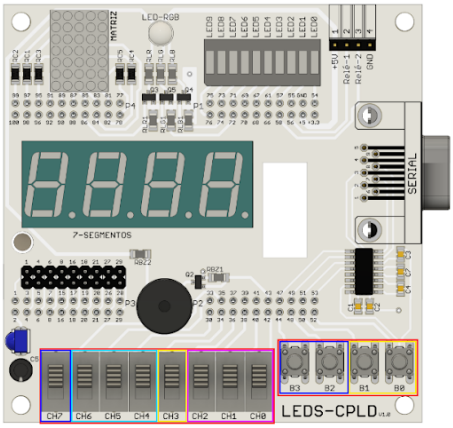

Figure 1. Em vermelho: Toda a seleção das interfaces de entrada; Em azul escuro: Dispositivos de entrada da autenticação de usuário da IE01; Em ciano: Dispositivos de entrada de funcionalidade da IE01; Em amarelo: Dispositivos de entrada da autenticação de usuário da IE02; Em magenta: Dispositivos de entrada de funcionalidade da IE02; Fonte: Manual(LEDS-CPLD); Imagem utilizada e modificada para fins educativos.

Como é possível perceber, a interface de entrada está distribuída de forma igualitária entre IE01 (Interface de Entrada Número 1) e IE02 (Interface de Entrada Número 2); Cada interface conta com controles independentes para autenticação de usuário e seleção da funcionalidade desejada, os quais serão declarados de forma alternativa (OU) na próxima tabela.

Foi definido que as chaves estarão na posição ON (ligadas) caso estejam com seus pinos seletores na parte inferior do soquete, e na posição OFF (desligadas) caso estejam com seus pinos seletores na parte superior do soquete.

Para melhor visualização da convenção, é possível voltar à figura 1, já que nessa ilustração todas as chaves estão na posição ON.

Quantos aos botões, foi definido que estarão na posição ON (ligados) caso estejam sendo pressionados, e na posição OFF (desligados) caso não estejam sendo pressionados.

Atendendo as especificações providas como requisitos para o produto, os códigos de autenticação para cada usuário são os seguintes (formato da entrada: IE01 OU IE02):

Número	Perfil	CH7 OU CH3	B3 OU B1	B2 OU B0
1	Admin	ON	OFF	ON
2	Tester	OFF	ON	ON
3	User	OFF	OFF	ON
4	Guest	ON	ON	OFF

Figure 2. Códigos de autenticação dos usuários. Fonte: Descrição do Problema 01.

No que diz respeito às funcionalidades, foi decidido utilizar código binário de 3 bits dos dígitos de 1 a 7 para sua escolha; A tabela a seguir mostra como estes códigos são representados nas duas interfaces de entrada (formato da entrada: IE01 OU IE02):

Funcionalidade	CH6 OU CH2	CH5 OU CH1	CH4 OU CH0
1	OFF	OFF	ON
2	OFF	ON	OFF
3	OFF	ON	ON
4	ON	OFF	OFF
5	ON	OFF	ON
6	ON	ON	OFF
7	ON	ON	ON

Figure 3. Códigos de entrada das funcionalidades.

É necessário notar, no entanto, que nem todos os usuários possuem acesso a todas as sete funcionalidades disponíveis; A tabela a seguir mostra uma relação dos usuários e as funcionalidades que cada um destes pode executar.

Usuário	Func. 1	Func. 2	Func. 3	Func. 4	Func. 5	Func. 6	Func. 7
Admin	SIM	SIM	SIM	SIM	SIM	SIM	SIM
Tester	SIM	SIM	SIM	SIM	NÃO	SIM	NÃO
User	SIM	NÃO	SIM	SIM	NÃO	SIM	NÃO
Guest	SIM	NÃO	NÃO	NÃO	NÃO	SIM	NÃO

Figure 4. Relação entre usuários e suas funcionalidades permitidas. Fonte: Descrição do Problema 01.

Vale lembrar que, independente de estarem executando funcionalidade permitida ou não, todos os usuários são passíveis de serem exibidos como parte da funcionalidade de número 2 caso esta esteja em execução na interface de entrada oposta; Mais detalhes na seção a seguir.

3.2. Exibição de dados nas interfaces de saída

Toda combinação referente a um usuário e uma funcionalidade (válida e permitida) gera ao menos uma resposta de saída, a menos que haja conflito de funcionalidade com a interface oposta à que "deseja" executar a funcionalidade; Neste caso, a interface de entrada que estiver sendo utilizada pelo usuário de maior prioridade (Admin ¿ Tester ¿ User ¿ Guest) terá sua funcionalidade executada. Se ambas as interfaces estiverem sendo utilizadas pelo mesmo tipo de usuário, a prioridade será da IE01.

As ISs (interfaces de saída) da placa CPLD-LEDS estão destacadas na figura a seguir:

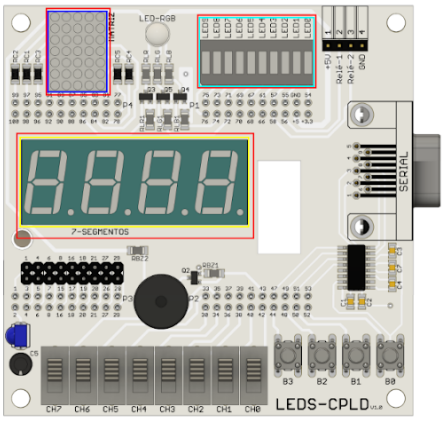

Figure 5. Em vermelho: Toda a seleção das interfaces de saída; Em azul escuro: IS01 (matriz de LEDs); Em ciano: IS02 (LEDs sequenciais); Em amarelo: IS03 (display de 7 segmentos); Fonte: Manual(LEDS-CPLD); Imagem utilizada e modificada para fins educativos.

Quanto à divisão entre quais grupos de LEDs serão acionados, é definido que perfis do tipo “Admin” e “Tester” acenderão linhas de LEDs da matriz (IS01), enquanto que perfis do tipo “User” e “Guest” acenderão LEDs únicos sequenciais (IS02).

O número do LED sequencial aceso corresponde ao número da funcionalidade menos um. Já quanto à matriz, é possível notar que esta carece de indicações da forma como suas linhas são indexadas, portanto é preciso salientar que **a contagem das linhas é feita de cima para baixo, começando pelo número 1**.

Assim sendo, a funcionalidade de número 7 executada por um perfil de “Admin” deve resultar no acionamento de todos os LEDs da linha inferior da matriz de LEDs (IS01), enquanto que a funcionalidade 6 executada por um perfil de “Guest” deve acender o led sequencial (parte da IS02) de número 5 (LED5).

A interface de saída de número 3 (IS03, display de 7 segmentos) é acionada quando ao menos uma das interfaces de entrada está executando a funcionalidade de número 2, e exibe número referente ao tipo de perfil de usuário (“Admin”: 1, “Tester”: 2, “User”: 3, “Guest”: 4) que está registrado na interface de entrada oposta àquela reconhecida como executando a funcionalidade referida, mesmo que o usuário exibido não esteja executando funcionalidade alguma no momento.

A exibição referente à funcionalidade de número 2 também respeita as prioridades de hierarquia de usuário e da IE01, e nesta ordem, mesmo que a prioridade de interface efetivamente não interfira no número exibido; Portanto, caso um perfil de “Admin” e um perfil de “User” estejam tentando utilizar a funcionalidade de número 2, podemos esperar não somente que, dentre a IS01 e a IS02, apenas a matriz de LEDs (IS01) seja acionada, mas também que o número 2, referente ao perfil tipo “User”, seja exibido na display de 7 segmentos da IS03.

3.3. Descrição do funcionamento dos módulos lógicos

Módulo de Permissão *(allpermission)*

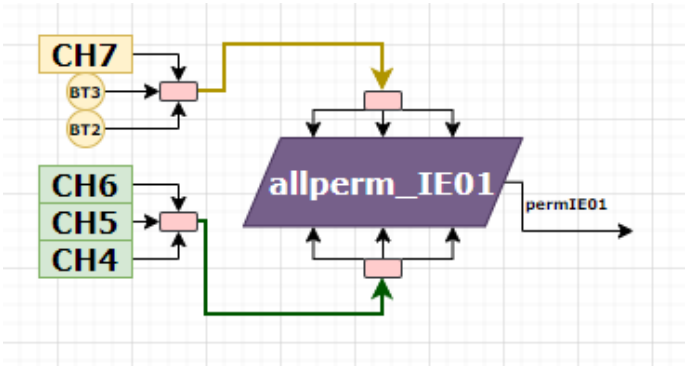

Figure 6. Esquema referente ao módulo de permissão da IE01.

Os módulos de permissão são responsáveis por avaliar se a combinação de usuário e funcionalidade de cada entrada é válida, ou seja, verificar se o usuário selecionado atualmente possui permissão para executar a função selecionada.

O retorno esperado para uma permissão concedida é 1 (nível lógico ALTO), enquanto que a falha na obtenção da permissão resultará em resultado 0 (nível lógico BAIXO). A saída é conectada ao fio "permIE0[1-2]" (1 para a IE01, 2 para a IE02), e a expressão simplificada que é utilizada para construir o módulo é (A!BC + !AC!DF + !ABC!DE + !ACD!F + AB!C!D!EF + AB!CDE!F), onde [A,B,C] são os bits relacionados à autenticação do usuário ("A" mais significativo), [D, E, F] são os bits relacionados à escolha da funcionalidade ("D" mais significativo) e "!" representa a negação do bit.

Módulo de seleção de prioridade da interface de entrada (entryinterfaceselector)

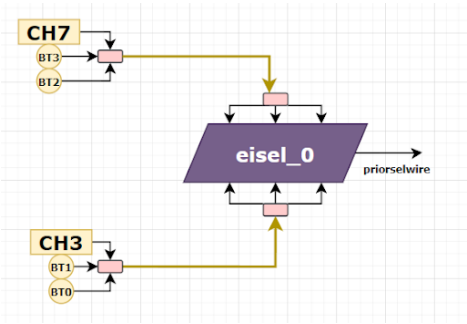

Figure 7. Esquema referente ao módulo de seleção de prioridade.

O módulo de seleção de prioridade avalia as entradas referentes à autenticação de ambas as interfaces de entrada (IE01 e IE02).

A lógica por trás do resultado da expressão do módulo é de retornar, pelo fio "prioselwire", 1 (nível lógico ALTO) para o caso da entrada referente à autenticação da IE01 estar inválida (nenhum tipo de usuário corresponde ao código) ou para o caso do tipo

de usuário autenticado na IE02 possuir nível de prioridade (hierarquia) maior que aquele autenticado na IE01. Nos demais casos, o módulo retornará 0 (nível lógico BAIXO).

A expressão referente ao módulo é (!A!C + !B!C + ABC + !ACD!EF + AB!C!EF + !A!BC!DEF + AB!C!DEF), onde [A,B,C] são os bits relacionados à autenticação do usuário na IE01 (“A” mais significativo), [D, E, F] são os bits relacionados à autenticação do usuário na IE02 (“D” mais significativo) e “!” representa a negação do bit.

Módulo de verificação de conflito de funcionalidade *(featureconflictchecker)*

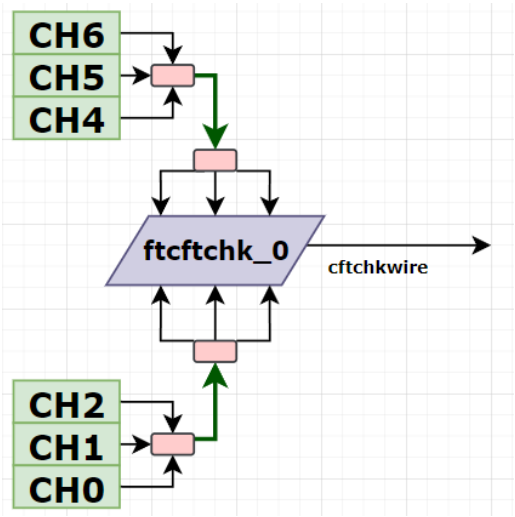

Figure 8. Esquema referente ao módulo de verificação de conflitos de funcionalidade.

O módulo de verificação de conflito possui uma função simples: Verificar a possibilidade de execução simultânea de 2 funcionalidades de acordo com os comandos inseridos por ambas as interfaces de entrada. Para que isto aconteça, é necessário que tais funcionalidades não sejam as mesmas.

O retorno esperado para este módulo é 1 (nível lógico ALTO) para o caso de NÃO existir conflito, e 0 (nível lógico BAIXO), caso exista conflito. A expressão que define o circuito empregado para tal propósito é (A$D + B$E + C$F)$, onde [A,B,C] são os bits relacionados à escolha da funcionalidade na IE01 (“A” mais significativo), [D, E, F] são os bits relacionados à escolha da funcionalidade na IE02 (“D” mais significativo) e $ representa operação de lógica booleana do tipo XOR.

Módulo de controle da informação de funcionalidade (unidirbustrans3b)

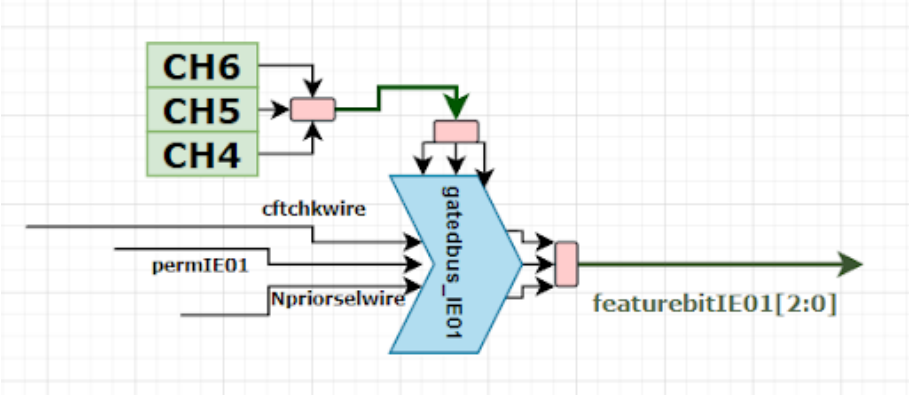

Figure 9. Esquema referente ao módulo de controle da informação de funcionalidade da IE01.

Os módulos de controle da informação da funcionalidade controlam a passagem da informação acerca da funcionalidade para as áreas que exigem autenticação e as devidas verificações de conflito ou prioridade (como o seletor da entrada do display de sete segmentos e as interfaces de saída).

Cada saída destes módulos (três fios do vetor de fios featurebitIE0[1-2]), pode resultar, individualmente, em 1 (nível lógico ALTO), caso a entrada referente a esta saída (A para o saída do fio de índice 0, B para o saída do fio de índice 1 e C para o saída do fio de índice 2), bem como a entrada de permissão (permIE0[1-2]), além da entrada de checagem de conflito (cftchwire) OU a entrada referente à prioridade (Npriorselwire, negação de priorselwire, para IE01, e priorselwire para IE02) estejam em nível lógico ALTO.

Em outras palavras, é permitido passar à frente a informação referente à funcionalidade selecionada por cada uma das interfaces de entrada se e somente se o usuário autenticado possuir permissão para tal e não houver empecilhos na execução da funcionalidade (interface de entrada possui prioridade ou não existe conflito de funcionalidade).

Expressões para cada saída:

B0 = A0*(flowvalve)*(conflictstatus+prioritystatus)

B1 = A1*(flowvalve)*(conflictstatus+prioritystatus)

B2 = A2*(flowvalve)*(conflictstatus+prioritystatus)

Onde [A0, A1, A2] são as entradas do módulo, [B0, B1, B2] são as saídas do módulo, “flowvalve” é o valor referente ao módulo de permissão (pelo fio permIE0[1-2]), conflictstatus é o valor referente ao módulo de verificação de conflito (cftchwire) e “prioritystatus” é o valor referente ao módulo de seleção de prioridade (pelos fios Npriorselwire e priorselwire).

Módulo de verificação da funcionalidade 2 (displayvalidator)

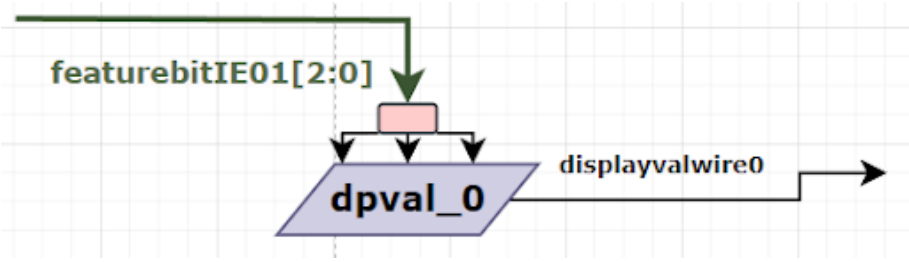

Figure 10. Esquema referente ao módulo de verificação da funcionalidade 2 da IE01.

Tais módulos possuem a simples função de verificar se suas respectivas interfaces de entrada possuem a funcionalidade 2 selecionada.

Desta forma, esperamos nível lógico ALTO na saída (displayselwire[0-1]) caso a funcionalidade 2 esteja selecionada para aquela interface de entrada, usando para tal o circuito expresso por (!A + B + !C), onde [A, B, C] são os bits relacionados à escolha da funcionalidade ("A" mais significativo) e "!" representa a negação do bit.

Módulo de seleção da entrada do display de 7 segmentos (sevendisplayselector)

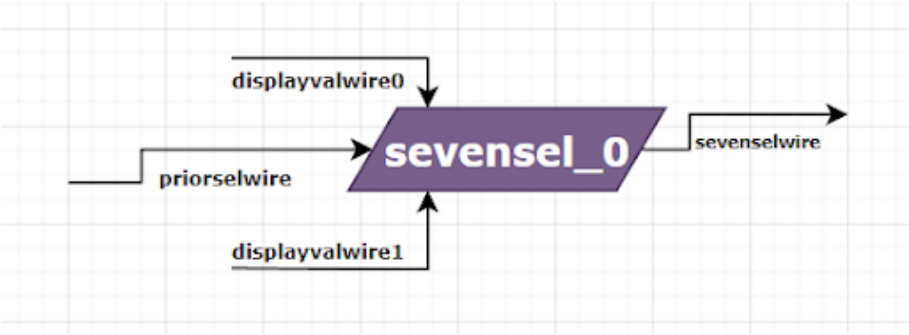

Figure 11. Esquema referente ao módulo de seleção da entrada do display de 7 segmentos.

O módulo de seleção da entrada do display de 7 segmentos recebe informações referentes ao status de prioridade (priorselwire) e execução da funcionalidade de número 2 (displayvalwire[0-1]) e decide qual interface de entrada poderá ter seu usuário autenticado (se aplicável) exibido, como número, na tela do display de 7 segmentos da interface de saída número 3 (IS03).

Com este intuito, o módulo retorna, por meio da saída conectada ao fio "sevenselwire", 1 (nível lógico ALTO) caso a interface candidata a exibição de usuário seja a IE02, ou 0 (nível lógico BAIXO) caso contrário (permitindo a exibição do usuário autenticado na IE01, mas também incluindo o caso de não existir funcionalidade 2 sendo executada).

A expressão utilizada para avaliar tal processo é (A*(!priorsel + !B)), onde A é o status de execução da funcionalidade 2 na IE01, B é o status de execução da funcionalidade 2 na IE02, "priorsel" é o status de prioridade e "!" é a negação da variável booleana.

Módulo tipo multiplex da entrada do display de 7 segmentos (mux6to3) e de seu

verificador final (mux2to1)

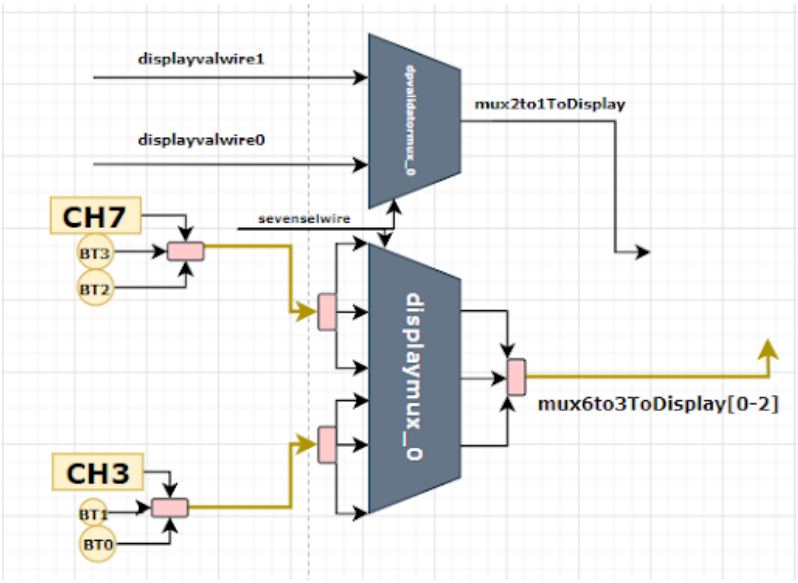

Figure 12. Esquema referente aos módulos multiplex da entrada do display de 7 segmentos e de seu verificador final.

Módulos do tipo *multiplex* são bem conhecidos em desenhos de circuitos digitais, dispensando portanto explicação sobre seu funcionamento (no entanto, o código-fonte deste produto possui documentação sobre os módulos *multiplex*).

Os módulos *multiplex* utilizados para selecionar a fonte da informação exibida no display de 7 segmentos e aquela usada para validar tal exibição são do tipo 6:3 e 2:1, respectivamente, ambos apresentando um fator de 2, e portanto fazendo uso de apenas um bit para a seleção da entrada.

Quanto à finalidade dos módulos *multiplex* dentro do produto, o módulo (6:3) faz a opção da entrada dos 3 bits correspondentes ao usuário a ser exibido no display de 7 segmentos dentre as duas interfaces de entrada, enquanto que o módulo (2:1) seleciona qual das verificações de execução da funcionalidade 2 (displayvalwire0 ou displayvalwire1) será repassada ao módulo decodificador do display de 7 segmentos.

É importante notar que o *multiplex* (2:1) possui lógica das interfaces de entrada "cruzada" (nível lógico BAIXO em seu seletor está associado à entrada que corresponde ao fio displaywire1, enquanto que o nível lógico ALTO em seu seletor está associado à entrada que corresponde ao fio displaywire0); Isto é necessário tendo em vista que a verificação da execução da funcionalidade dois deve acontecer para a interface de entrada oposta àquela selecionada pelo módulo de seleção do display de 7 segmentos (sevendisplayselector).

O seletor de ambos os módulos *multiplex* acima mencionados está associado à entrada que corresponde ao fio sevenselwire, cuja lógica é controlada pelo módulo de seleção do display de 7 segmentos.

Módulo decodificador do display de 7 segmentos (sevensegmentdisplay)

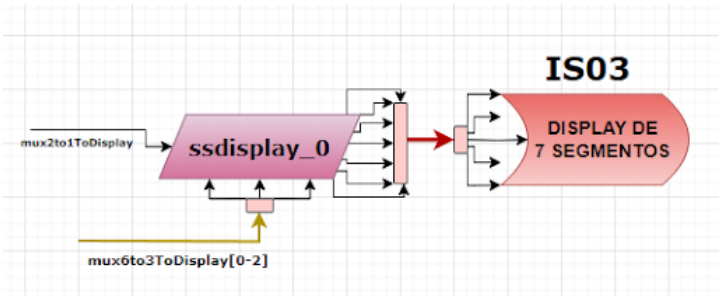

Figure 13. Esquema referente ao módulo decodificador do display de 7 segmentos e à interface de saída número 3 (IS03).

O módulo decodificador do display de 7 segmentos é responsável por receber a informação, em formato binário, referente ao usuário autenticado em uma das interfaces de entrada e acionar os segmentos do display relacionados ao número correspondente ao usuário autenticado (“Admin”: 1, “Tester”: 2, “User”: 3, “Guest”: 4).

Para chegar a tal resultado, é necessário obter expressões referentes a cada segmento do display a ser aceso, fazer o produto pela verificação da funcionalidade 2, e finalmente negar as expressões (visto que os segmentos estão acesos em nível lógico BAIXO, e não ALTO).

Expressões:

(segA, segD) = (!AC * dpval)

(segB) = ((!AC + !BC + AB!C) * dpval)

(segC) = ((!BC + AB!C) * dpval)

(segE) = ((!ABC) * dpval)

(segF) = ((AB!C) * dpval)

(segG) = ((!AC + AB!C) * dpval)

Onde [A, B, C] são os bits relacionados ao código do usuário cujo número será exibido (“A” mais significativo), “dpval” é a entrada da validação da funcionalidade 2 na interface oposta àquela cujo usuário terá seu número exibido, e “ ” / “!” são negações.

Módulo decodificador das interfaces de saída tipo LED (ledsequencedecoder)

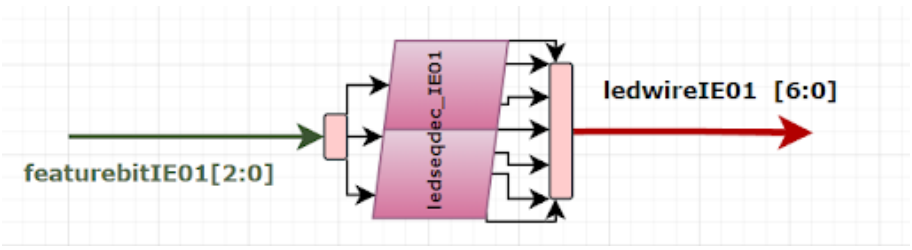

Figure 14. Esquema referente ao módulo decodificador das interfaces de saída tipo LED da IE01.

Os módulos decodificadores das interfaces de saída tipo LED são decodificadores simples, pois apenas uma única combinação das entradas pode resultar em nível lógico ALTO em cada uma das saídas (decodificador BIN(3b) -¿ [0-6]).

Expressões:

led0and = (!A + !B + C)

led1and = (!A + B + !C)

led2and = (!A + B + C)

led3and = (A + !B + !C)

led4and = (A + !B + C)

led5and = (A + B + !C)

led6and = (A + B + C)

Onde [A, B, C] são os bits relacionados à funcionalidade escolhida, já "passada" pela verificação de permissão e não-conflito ou prioridade (fio featurebitIE0[1-2][0-2]), led[0-6]and são as saídas para cada LED sequencial (IS02) ou linha da matriz de LEDs (IS01), e "!" é uma negação.

Módulo de seleção da interface de saída LED (ledinterfaceselector)

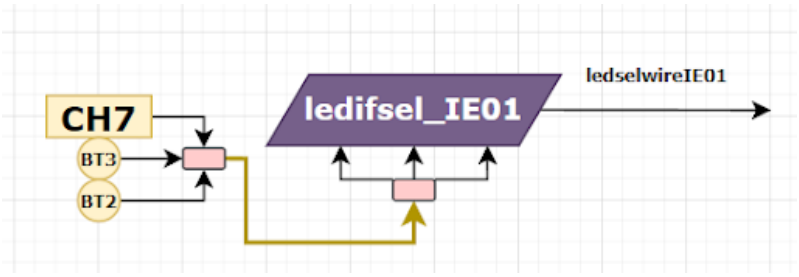

Figure 15. Esquema referente ao módulo de seleção da interface de saída LED da IE01.

Os módulos de seleção da interface de saída LED são estruturas que possuem como objetivo avaliar o tipo de usuário autenticado nas suas respectivas interfaces de entrada, de modo a produzir resultado que poderá ser utilizado como seletor para um módulo multiplex

Como requisitado para o produto, os perfis de usuário "Admin" e "Tester" deverão ter suas funcionalidades exibidas na matriz de LEDs (IS01), enquanto que os perfis de usuário "User" e "Guest" deverão ter suas funcionalidades selecionadas exibidas na barra de LEDs sequenciais (IS02).

Com este objetivo, esperamos o retorno, por meio da saída que é conectada ao fio "ledselwireIE0[1-2]", de nível lógico BAIXO caso o perfil de usuário corresponda a "Admin" ou "Tester", ou alto, caso contrário; A expressão que permite construir um circuito que desempenha tal função é ((C*(A$B))), onde [A, B, C] são os bits referentes à entrada da autenticação do usuário ("A" mais significativo), $ corresponde à operação

booleana XOR, e “ ” é a negação da expressão (visto que sem a negação estamos retornado nível lógico ALTO para “Admin” ou “Tester”).

Módulo tipo (de)*multiplex* das interfaces de saída tipo LED (mux7to14)

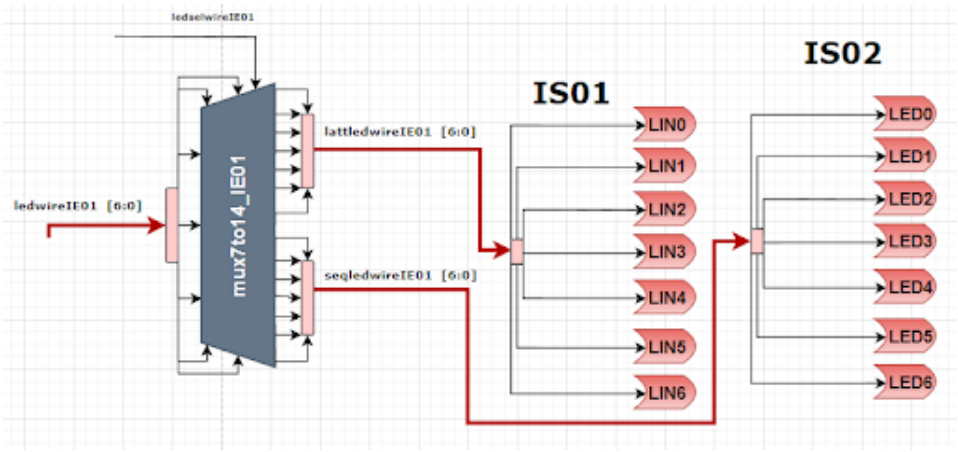

Figure 16. Esquema referente ao módulo multiplex (7:14) das interfaces de saída tipo LED e às interfaces de saída de número 1 e 2 (IS01 e IS02).

Como já mencionado, módulos tipo *multiplex* dispensam explicações sobre seu funcionamento lógico (porém vale ressaltar que a documentação do código possui informações detalhadas sobre o funcionamento dos mesmos).

Quanto ao papel desempenhado pelo módulo apresentando acima, este é o de selecionar, por meio de leitura do seletor representado pelo fio “ledselwireIE0[1-2]”, o conjunto de saídas referente à interface de saída que pode ser acessado pelo usuário atualmente autenticado na interface de entrada correspondente ao (de)*multiplex* (7:14) em questão.

O seletor com entrada 0 (nível lógico BAIXO) resultará na passagem das informações de entrada (carregadas pelos 7 fios dos vetores de fios “ledwireIE0[1-2]”) para a saída “superior” (vetor de fios “lattledwireIE0[1-2]”), correspondente à interface de saída de número 1 (IS01), enquanto que entrada no seletor 0 (nível lógico ALTO) resultará na passagem das mesmas informações de entrada para a saída “inferior” (vetor de fios “seqledwireIE0[1-2]”), correspondente à interface de saída de número 2 (IS02).

4. Descrição do código

Para não tornar o relatório muito extenso, somente o módulo principal será melhor detalhado, pois ele já é suficiente para entender quais são as funções que se esperam dos outros módulos.

É válido ressaltar que todos os nomes atribuídos as variáveis e módulos seguem um padrão lógico.

4.1. Módulo principal

O módulo principal chama-se ”mainmodule”. È nele que serão instanciados todos os demais módulos citados, os ligando como se fossem um só código.

2.3.1 Inputs e outputs

No módulo mainmodule é declarado logo no inicio os inputs, que são (CH0, CH1, CH2, CH3, CH4, CH5, CH6, CH7, BT0, BT1, BT2, BT3) cada uma dessas variaveis representam as chaves HH e push-boton da placa FPGA.

Os outputs desse módulo são declarados como sendo (LED0, LED1, LED2, LED3, LED4, LED5, LED6). Cada uma das variaveriaveis representam um led da placa de circuito digital

2.3.2 Fios (wire)

Os fios ou wire existentes no código, são uma espécie de condutor que liga uma porta lógica com a outra. Elas são realmente importantes para preservar a troca de informações entre todos os equipamentos digitais a serem utilizados na placa. Será explicitado no decorrer dessa sessão a estrutura e lógica de cada um dos dos wires contidos no código.

Os fios (wire) que conectam um módulo com o outro estão declarados como sendo (NBT0, NBT1, NBT2, NBT3) Cada fio representa a negação dos botões, ou seja, caso um botão tenha o nível lógico alto, o fio será de nivel lógico baixo. Em situação contrária, a mesma lógica é aplicada.

Os fios (permIE01, permIE02) são importates, pois são eles que dão permissão de passagem para as funcionalidades aplicadas pelas duas interfaces de entrada existentes (IE0 e IE1). Caso o nível lógico for alto, o programa deve permitir a passagem da funcionalidade, caso o nível lógico seja alto, a passagem da funcionalidade deverá ser negada.

Fios declados como (priorselwire e Npriorselwire) são fios de seleção de prioridade de interfaces e suas negações. Caso o nivel lógico seja alto, a prioridade deve ser da IE02, caso seja baixo, a prioridade é da IE01.

O ”cftchkwire” é o fio que verifica se há conflito com a passagem de prioridade do circuito. Caso o nível lógico for alto, há existência de conflito, caso for baixo, não há conflito.

”sevenselwire” é o fio do seletor do multiplex (6:3) ligado ao display de 7 segmentos.

”displayvalwire0” e ”displayvalwire1” são os fios dos módulos da funcionalidade 2 que validam cada interface de entrada.

O fio que verifica se as saídas estão ligadas é declarado como ”noneOnWire”.

Os fios ”mux2to1ToDisplay”, ”displayvalwire0” e ”displayvalwire1”são fios cruzados da validação da funcionalidade 2. Se o MUX 2:1 receber ALTO no ”sel”, passa o fio ”displayvalwire0”, caso contrario, passa o sio ”displayvalwire1”.

”mux6to3ToDisplay0”, ”mux6to3ToDisplay1” e ”mux6to3ToDisplay2” são os fios selecionados do usuario a ser exibido no display, de acordo com o MUX 6:3.

Os fios ”ledselwireIE01” e ”ledselwireIE02” são fios de seleção da interface de saída para cada interface de entrada.

Os vetores de fios de 3 bits referentes as funcionalidades após passarem pelo BUS transceiver de autorizacao são declarados como ”featurebitIE01[2:0]” e ”featurebitIE02[2:0]”.

Os vetores de fios de entrada no MUX 7:14 de 7 bits para cada interface de entrada são declarados como ”ledwireIE01 [6:0]” e ”ledwireIE02 [6:0]”.

Vetores de fios de saída do MUX 7:14 de 7 bits da IS01 (matrix de LEDs) para cada interface de entrada são declarados como ”lattledwireIE01[6:0]” e ”lattledwireIE02[6:0]”.

Vetores de fios de saída do MUX 7:14 de 7 bits da IS02 (LEDs sequenciais) para cada interface de entrada são declarados como ”seqledwireIE01[6:0]” e ”seqledwireIE02[6:0]”.

No restante do código são instanciados os outros módulos citados na sessão anterior; A documentação de cada módulo, feita por meio de comentários nos arquivos-fonte do projeto, possui informações detalhadas acerca do código, incluindo as aqui apresentadas.

5. espercificações do projeto

5.1. Especificações do circuito

Total de portas lógicas utilizadas: 144

Comprimento máximo (incluindo inversores): 10 portas lógicas

Largura (fan-in de porta lógica) máxima: 7 entradas.

5.2. Especificações da última compilação do protótipo

Marca do CPLD: Altera

Série: Max II

Modelo: EPM240T100C5

Versão do Intel Quartus Prime: 20.1.0 Build 711 06/05/2020 SJ Lite Edition

LEs (Elementos Lógicos) totais: 63/240

LABs utilizados: 8/24

Pinos utilizados: 38/80

6. Conclusão

Após diversas horas semanais de plena dedicação aos estudos para a entrega do projeto, existem poucas dúvidas de que o protótipo descrito neste relatório foi feito da melhor forma possível dentro da interpretação adotada pela equipe de desenvolvimento.

Todos os critérios estabelecidos, com exceção da possível execução encontrada na ausência de ”piloto automático” (requisito que é, tendo em vista as limitações da interface de entrada e a proibição do uso de elementos sequenciais, aparentemente impossível de estabelecer para mais de uma única funcionalidade, e mesmo assim ao custo de não permitir estado ”desligado”) foram muito bem cumpridos.

É válido ressaltar que foram levados em conta todos os problemas e questões passíveis de comprometer o bom funcionamento do protótipo, incluindo até certo ponto erros humanos (a exemplo da escolha dos botões para autenticação, visto que não existe

risco ao executar uma autenticação ”acidental”, o que não pode ser dito das funcionalidades). Deste modo, em situações reais, as falhas são reduzidas, tornando o produto mais eficiente e seguro para o uso.

7. Referências

João Lucas Silva Serafim Silva, Welton dos Santos Cerqueira; Tabela-Verdade MI CD; Disponível em: https://docs.google.com/spreadsheets/d/1goVYxl-u-5STlJ099Og-evxvLh7D5HOIs_nBsheqaE4/edit?usp=sharing

Ivan Valeije ldoeta; Francisco Gabriel Capuano; Elementos de Eletrônica Digital, 4ª Edição. São Paulo, 2008. Editora Érica Ltda.

Universidade Estadual de Feira de Santana, Departamento de Tecnologia, Área de Eletrônica e Sistemas, Laboratório de Eletrônica Digital e Sistemas; Manual(LEDS-CPLD); Disponível em: https://drive.google.com/file/d/168zWlJU0rbnq3q8QJXnrwRY8iO6Ds2xQ/view

Universidade Estadual De Feira De Santana, Departamento De Tecnologia, Curso De Engenharia De Computação; Descrição do Problema 1; Disponível em: https://docs.google.com/document/d/1sOL7cPMFXA3jiQVD6n7QKrNkjwKXw2dTWnW09HbGpX0/edit?usp=sharing