Universidade Federal do Espírito Santo

Caio Alan littike

Gustavo Henrique Vago Brunette

Jorge Metri Miranda

Uálaci dos Anjos Júnior

8 de agosto de 2022

Sistemas Digitais

Professor: Anibal Cotrina Atencio

Engenharia da computação

Universidade Federal do Espírito Santo
DCEL

Trabalho final da disciplina

Projeto RTL Microondas

Aluno(a)s: Caio Alan littike
Gustavo Henrique Vago Brunette
Jorge Metri Miranda
Uálaci dos Anjos Júnior
Professor(a) orientador(a): Anibal Cotrina Atencio

8 de agosto de 2022

Sumário

1 Introdução

Projetos RTL é o nome dado a descrição de operação em nível de registrador de um sistema digital. Nesse tipo de projeto o comportamento do circuito é descrito por meio de um fluxo de sinais (bits), que são manipulados por meio de circuitos lógicos combinacionais e sequenciais a fim de realizar uma tarefa desejada. A abstração desse tipo de projeto é feita por meio de linguagens de descrição de hardware, como VHDL e Verilog, e diagramas de blocos. Nesse trabalho será apresentada a implementação de um projeto RTL projetado em uma placa FPGA (field programmable gate arrays), a fim operar de forma análoga ao circuito de um microondas.

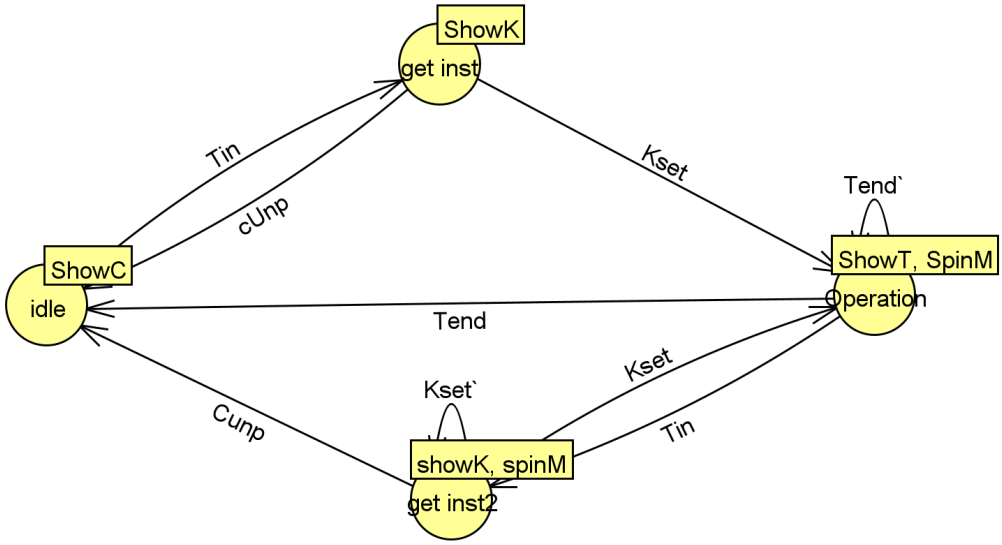
2 Metodologia

A proposta do projeto, foi de implementar um circuito com funcionamento análogo ao de um microondas, com suas principais funções. Para realizar esse objetivo, o projeto foi dividido em duas etapas. Na primeira etapa, foram separados 4 grupos. Cada grupo ficou responsável por implementar um bloco operacional do circuito do microondas. Não serão descritos nesse Trabalho a implementação dos blocos individuais, visto que esse já foram documentados por cada grupo individualmente. Após a implementação dessa primeira etapa, cada grupo recebeu os blocos operacionais implementados pelos demais grupos com o objetivo de implementar um circuito de controle e a partir desse, unificar o projeto já operando como um circuito de microondas.

3 Implementação

3.1 Máquina de estados

O primeiro passo na segunda etapa, foi realizar a implementação de uma máquina de estados que represente o controle do circuito do microondas, que ficou como a seguir:



FSM do circuito de controle

Estados:

- **00 (idle):** Estado que representa quando o microondas está em funcionamento fora de aquecimento. Nesse estado, o relógio é mostrado para o usuário.
- **01(get inst):** Esse estado ocorre imediatamente após o usuário pressionar alguma tecla do teclado, com exceção de ligar e cancelar. Nesse estado, um valor definido pelo usuário através do teclado é recebido e enviado para o próximo estado.
- **10 (operation):** Esse estado ocorre após o usuário pressionar o botão ligar representado pela entrada **Kset**. Nesse estado o timer é acionado e ao fim da contagem o microondas volta para o estado Idle, onde o relógio é mostrado novamente. Caso o usuário pressionar algum botão ao invés de cancelar, a máquina passa para o próximo estado, **getInst2**.
- **11(GetInst2):** Esse estado ocorre de forma análoga ao estado **get Inst**, porém o motor continua girando e o timer contando até o final da contagem. Nesse estado o usuário pode alterar a hora do relógio enquanto o timer opera.

Entradas:

- **Tin:** Representa o recebimento de qualquer input do teclado.
- **cUnp:** Representa o recebimento do input **cancelar**.
- **Kset:** Representa o recebimento do input **ligar**.
- **Tend:** Representa o recebimento do sinalizador do fim da contagem do timer.

Saídas:

- **ShowC, ShowK, ShowT:** outputs que enviam a ordem de mostrar no display as informações do relógio, teclado e timer, respectivamente.
- **SpinM:** outPut que envia ordem de acionamento do motor.

Essa é a máquina de estados que foi implementada para o circuito de controle do microondas. No entanto, a adição de algumas informações na lógica foram adicionadas que serão explicadas a seguir.

Além dos estados , entradas e saídas demonstrados pela máquina de estados acima, existem algumas outras informações que foram implementadas. O microondas possui um circuito que ativa 4 leds que ficam piscando na frequência de 10 Hz por 1 segundo que é acionado em determinada ocasião.

No circuito também foi adicionado um outro botão, que podemos chamar de **SetClock**, que ativa no Relógio o valor indicado pelo input do teclado, e o relógio começa a contar a partir desse ponto. Esse input pode ser ativado a qualquer momento que a máquina estiver nos estados getInst ou getInst2. Também há um input **ResetClock**, que reseta o tempo do relógio para 00:00 e o relógio continua sua contagem a partir do zero.

3.2 Funcionamento

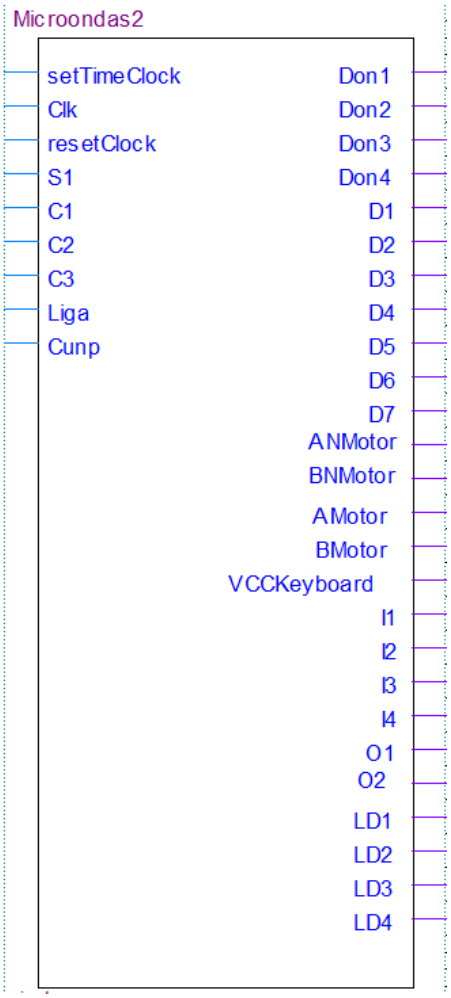
O microondas inicia no estado idle. Quando isso ocorre, o relógio inicia a partir de 00:00, e os leds sinalizadores piscam demonstrando que o microondas foi ligado. Em seguida, o circuito mostra o horário do relógio enquanto aguarda um input do teclado. Além disso, o usuário pode enviar o input de Reset para que o relógio reinicie sua contagem a partir do 0. Qualquer outra função pressionada se não as indicadas no estado ou mencionadas não fará nada. O mesmo vale para os outros estados.

No próximo estado, **get Inst**, caso a tecla Cancela for pressionada, o circuito volta para o estado **idle** , Já se alguma outra tecla com exceção de ligar for pressionada, o circuito recebe o input , e o mostra no display os inputs recebidos até que o usuário pressione a tecla **ligar** , que ativa o timer e o circuito vai para o estado **Operation**. Vale ressaltar que nesse estado, caso o usuário envie a entrada de **setTime** , a entrada atual recebida do teclado que está sendo mostrada no display será ativada no relógio e esse recomeçará sua contagem a partir desse valor. Caso o usuário queira somente fazer essa operação, ele só precisa pressionar o botão cancelar para a contagem do relógio ser mostrada novamente no display e o circuito voltar para o estado **idle**, caso o usuário queira continuar, ele pode então pressionar o botão ligar para ativar o timer.

No estado **Operation**, o timer fará a contagem decrescente do valor informado no estado anterior, e concomitantemente, o motor girará. Quando a contagem chegar ao fim, ou o usuário pressionar o botão **cancela**, o circuito sinalizará com os leds sinalizadores o fim da contagem, e o circuito voltará para o estado **idle**. Durante a contagem, um led fica ativo representando que o estado de operação do aquecimento está em funcionamento.

Nesse mesmo estado, caso o usuário pressione alguma tecla do teclado com exceção da **ligar** ou **cancelar**, o circuito vai para o estado **get inst2**, onde o timer continua decrementando até que o usuário pressione o botão **cancela**. O estado **get inst2** serve para que o usuário não precise esperar o fim do funcionamento do aquecimento para alterar a hora, dessa forma, ele pode alterar o horário enquanto o aquecimento opera. Um detalhe que vale ser mencionado, é que quando o timer recebe o input de cancelamento da contagem, ele não cancela de imediato, ele pausa sua contagem, então se o usuário tentar colocar outro valor de contagem, ele ignorará e continuar contando a partir de onde a contagem anterior foi pausada. Isso serve para caso o usuário tenha pressionado não intencionalmente a função de cancelar. Para que o timer zere sua contagem, basta pressionar duas vezes o botão de cancelar. Assim, quando a entrada de acionamento for enviada, ele contará normalmente a partir do novo valor inserido.

3.3 Design Computacional



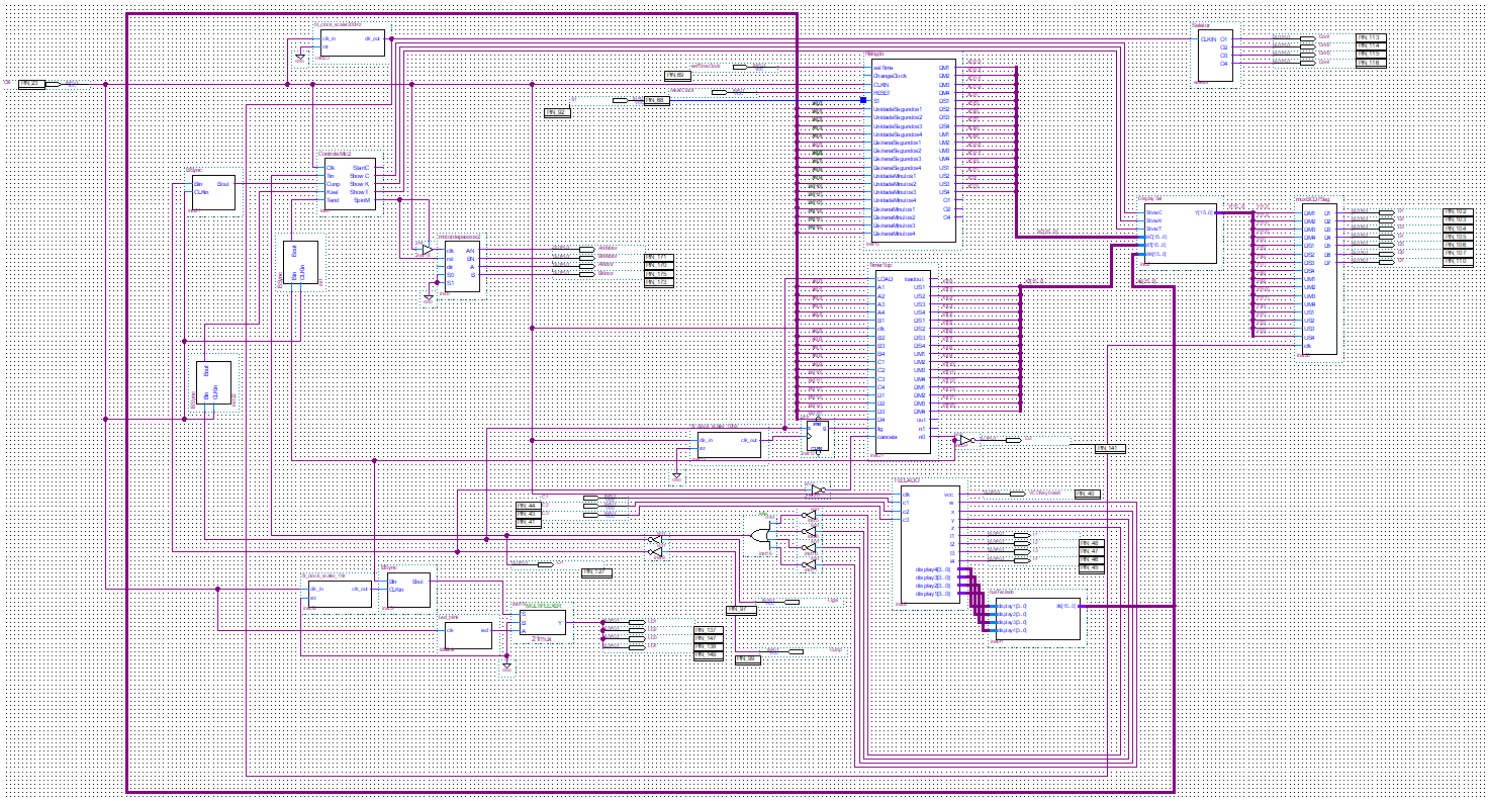
Entradas:

- **setTimeClock:** ativa os inputs referentes aos minutos e segundos do relógio e faz o mesmo recomeçar sua contagem a partir do valor definido pelosinputs recebidos.
- **clk:** entrada de clock do circuito.
- **S1:** Pausa e despausa o relógio.
- **C1:** Entrada de funcionamento do teclado.
- **C2:** Entrada de funcionamento do teclado.
- **C3:** Entrada de funcionamento do teclado.
- **Liga:** Liga o timer.
- **Cunp:** Cancela o timer ou volta para a exibição do relógio como explicado no tópico anterior.

Saídas:

- **Don1,2,3,4:** ativam os displays 1,2,3 e 4 do fpga , respectivamente.
- **D1,2,3,4,5,6,7:** Entradas para o display de sete segmentos do fpga.
- **ANMotor, BNMotor, AMotor, BMotor:** Saídas enviadas do circuito do motor de passos para a placa externa ao fpga que controle o motor de passos.
- **VCCKeyboard:** Tensão para operação do circuito do teclado.
- **I1,2,3,4:** OutPuts para operação do circuito do teclado.
- **O1,O2:** Leds sinalização de recebimento de input do teclado e operação de aquecimento.
- **LD1,2,3,4:** Leds de sinalização de quando o microondas liga e quando o timer termina a contagem.

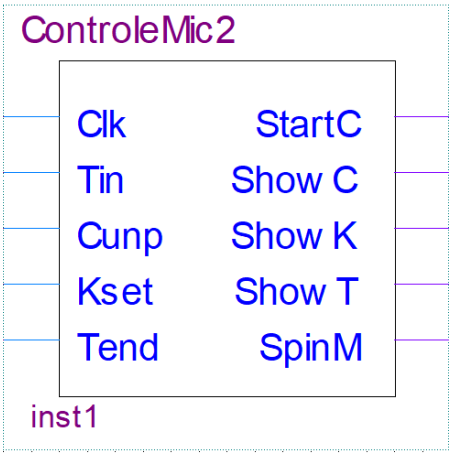
Layout do design computacional



Para melhor visualização do design computacional, clique aqui.

Alguns blocos do circuito foram implementados em verilog, System Verilog e VHDL. Todos os arquivos dos blocos do circuito estão disponíveis para acesso neste Repositório

Bloco de controle do microondas



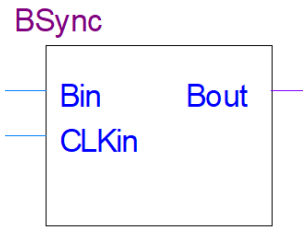
Esse bloco é o bloco responsável por controlar todos os demais blocos. A partir dele são geradas as saídas de controle. Essas saídas são:

- **startC:** Saída que não foi utilizada por não haver necessidade. Inicialmente ela foi implementada com o intuito de iniciar a contagem do relógio. Porém o circuito do relógio foi modificado para começar a contar automaticamente, tirando a utilidade dessa saída.
- **ShowC,K,T:** Essas saídas são enviadas ao bloco operacional DisplaySel. Elas são responsáveis por enviar o comando que indica qual informação deve ser exibida no display, podendo ser a do Relógio, Teclado, ou Timer respectivamente.
- **SpinM:** saída que indica quando o motor deve ser ligado.

Entradas:

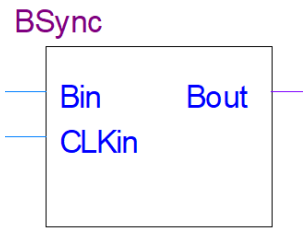
- **Tin:** Entrada que indica que foi recebido um input do teclado. Quando isso ocorre, o circuito de controle ativa a saída **ShowK** para que as informações associadas ao teclado sejam exibidas no display.
- **Cunp:** Entrada que indica que o botão cancela foi pressionado. Quando isso ocorre, dependendo do estado, o circuito de controle irá sinalizar para o bloco operacional que o timer deve ser resetado e o motor parado, ou que a informação exibida no display deve voltar a ser a do relógio.
- **Tend:** Entrada que indica quando o Timer finaliza sua contagem ou é resetado. Ela serve para que o circuito de controle saiba quando é o momento de enviar o sinal de parada do motor e voltar para a contagem do relógio.

Bloco de sincronização de aperto de botão



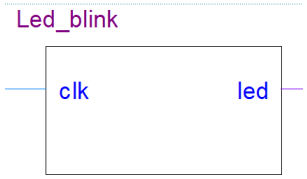
Esse bloco é responsável por ajustar um input recebido. Ele faz com que o input recebido ocorre somente por um ciclo do clock. A entrada **Bin** é o input e **Bout** o output que ocorre somente por um ciclo do clock de entrada **CLKin** independente de quanto tempo o input ficar ativo.

Blocos de mudança da frequência do clock



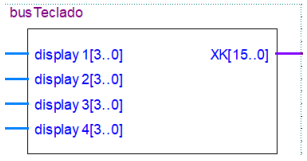
Todos os blocos com os nomes bl_clock_scaler são responsáveis por ajustar o clock a uma determinada frequência, que é indicada pelo número no fim do nome. Nesse caso, o bloco ajusta a frequência para 1 Hertz. A entrada clk_in é o clock geral do circuito, e rst é um input de enable barrado. Ou seja, quando rst for 1, o clock para de operar. Do contrário, o clock opera normalmente.

bloco de pisque do led



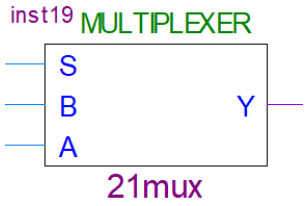
Como o nome sugere, o bloco simplesmente faz uma entrada ficar oscilando numa determinada frequência, o output desse bloco é normalmente conectado a um led. O intuito desse bloco é fazer um led sinalizador.

Bloco distribuidor do teclado



Esse bloco tem como intuito receber as entradas do display do teclado, sendo 4 entradas de 4 bits cada uma, que são recebidas nos inputs "display 1,2,3 [3..0]" , respectivamente. Essas entradas por meio de código são unificadas num único Bus que sai através da saída XK[15..0].

Multiplexador 2x1:

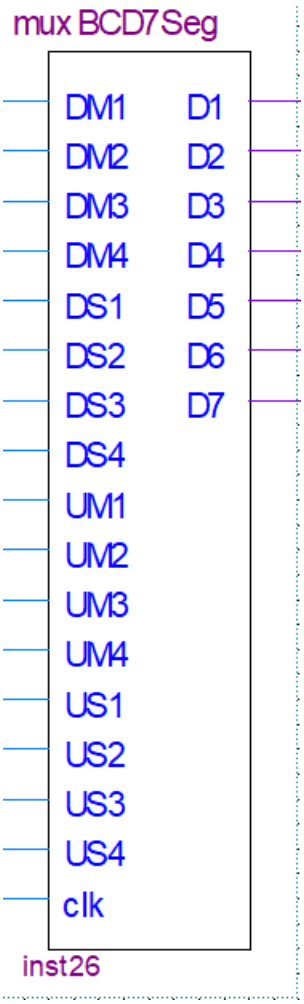


Multiplexador básico que seleciona uma das duas entradas por meio da entrada **S** e a transfere para a saída.

Bloco operacional de seleção da informação mostrada no display

Esse circuito é responsável por selecionar adequadamente qual informação será exibida no display. As entradas ShowC,K,T são recebidas do bloco de controle, e a partir delas ele envia para o display as informações referentes ao relógio, timer ou teclado, conforme a lógica da máquina de estados.

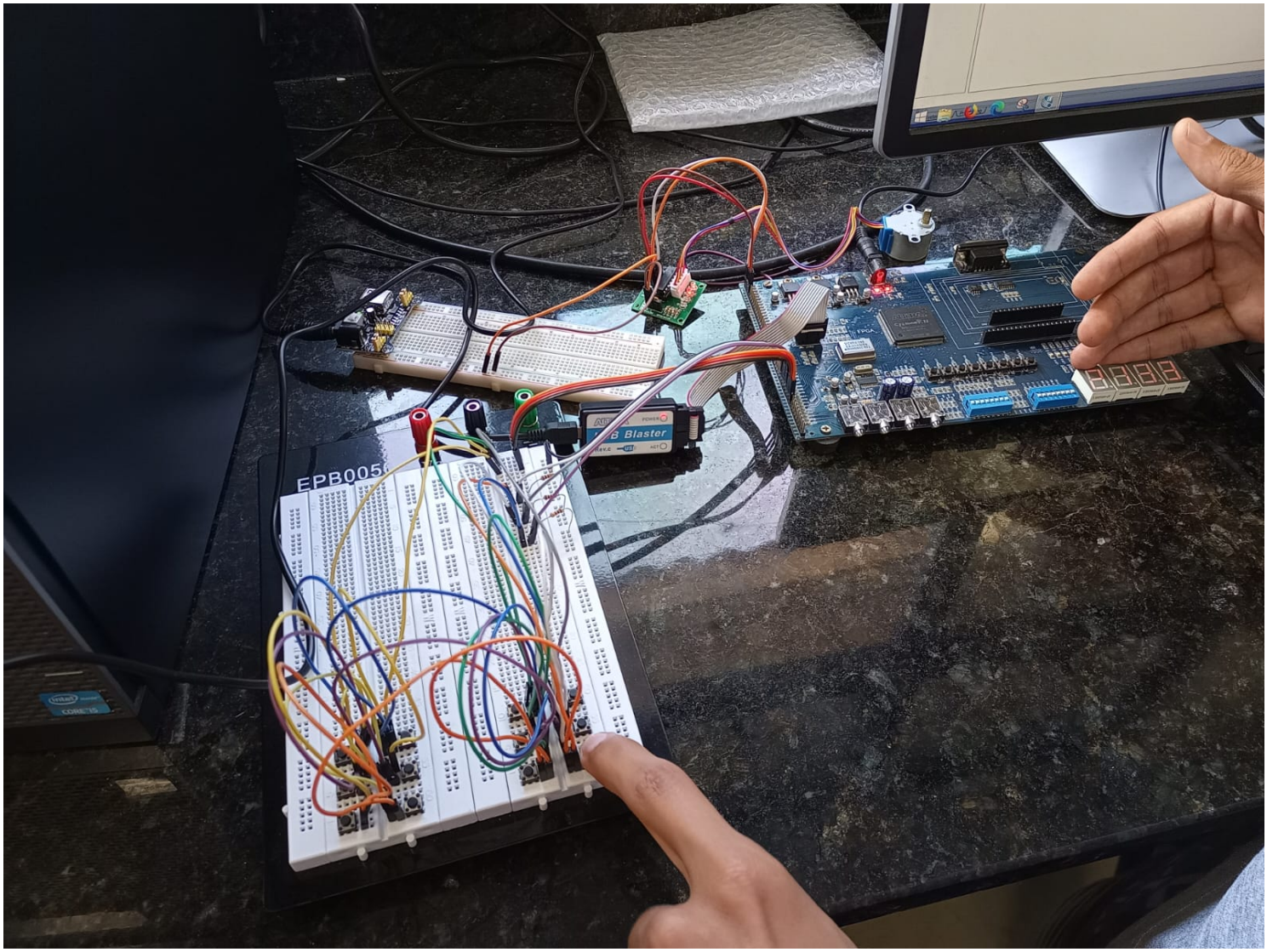
Bloco de exibição das informações no display



Esse bloco recebe as informações do bloco Display cel referentes às informações do relógio, timer ou teclado, e as envia adequadamente para serem exibidas no display, já decodificadas e com a multiplexação necessária para os quatro displays serem exibidos com a informação adequada em cada um em sincronia com o circuito seletor que realiza a ilusão de simultaneidade de exibição dos 4 displays.

4 Resultados

O circuito ficou como mostrado a seguir:



Um videoclipe com o funcionamento do circuito está disponível neste link.

5 Conclusão

5.1 Dificuldades

Com relação às dificuldades encontradas durante o decorrer do trabalho, as principais foram:

- **Documentação:** A documentação dos grupos , inclusive o nosso, não cumpriram adequadamente seu objetivo, que foi proporcionar um entendimento macro a respeito do circuito implementado por cada grupo. Percebeu-se que a exigência de uma documentação mais formalizada e detalhada poderia possibilitar uma redução grande no esforço para compreensão dos circuitos implementados pelos demais grupos. Devido a isso, foi necessário "debugar"cada circuito implementado separadamente para compreender razoavelmente o funcionamento de cada um a fim de possibilitar o funcionamento dos blocos em conjunto.
- **Software quartus:** algumas funcionalidades do software são pouco intuitivas ou complexas de serem compreendidas, por consequência disso, foi necessário em vários momentos, fazer uma pesquisa de ferramentas em específico do software para utilizá-las durante a implementação do projeto.
- **lógica computacional:** Outra dificuldade foi compreender o funcionamento dos CI'S que foram necessários para a implementação. Além da compreensão da lógica das linguagens de descrição de hardware, que possuem uma lógica de implementações diferentes das linguagens de programação nas quais nós estamos habituados. Devido a esse contraste, foi necessário um estudo e pesquisa a respeito desses CI's e linguagens para conseguir utilizar tais ferramentas.
- **Tempo:** O projeto demandou um tempo considerável para implementação. Principalmente devido aos fatores citados anteriormente. Devido a falta de mais tempo disponível, algumas melhorias que poderiam ser implementadas infelizmente não puderam ser implementadas. Uma função que ficou em falta no nosso projeto, foi o sensor de abertura e fechamento da porta do microondas, que poderia ter sido facilmente implementado, porém devido a falta de tempo e revisão, acabou sendo deixada de lado.

5.2 Aspectos Positivos

Durante o processo de implementação do trabalho, pudemos desenvolver muitos conhecimentos associados a criação de projetos RTL, criação de circuitos práticos em FPGAs, e linguagens de descrição de hardware. Tivemos contato com muitas tecnologias cruciais do campo de prototipagem de sistemas embarcados, tais como o software quartus, linguagens como vhdl e verilog e conhecimento de circuitos integrados clássicos.

O resultado obtido no trabalho foi satisfatório. A proposta inicial foi cumprida, que foi unir os blocos de todos os grupos atrás de um circuito de controle sequencial gerado a partir de uma máquina de estados finitos. Conseguimos adaptar os circuitos dos demais grupos , assim como a primeira versão do nosso, para funcionar em conjunto com os demais circuitos.

O circuito de controle de comportou bem, todas as funções básicas necessárias para a simulação do funcionamento de um circuito de um microondas foram implementadas.

Pudemos concluir, que os diagramas de blocos são ótimas ferramentas para entrada na área de implementação de circuitos em FPGAs. No entanto, percebeu-se que a implementação através de linguagens de descrição de hardware é menos dispendiosa apesar de menos intuitiva. O trabalho despertou em alguns componentes do grupo o interesse em se aprofundar na área de sistemas digitais.