UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

CAMPUS CURITIBA

ENGENHARIA ELETRÔNICA

DENNIS GUILHERME DE MACEDO BRAGAGNOLO

RELATÓRIO DE ATIVIDADE PRÁTICA

INTERFACE TCP/IP NA PLACA ALTERA DE-2

CURITIBA

NOVEMBRO, 2020

Dennis Guilherme de Macedo Bragagnolo

RELATÓRIO DE ATIVIDADE PRÁTICA

INTERFACE TCP/IP NA PLACA ALTERA DE-2

Atividade desenvolvida como projeto final da disciplina de Lógica Reconfigurável do curso de Engenharia Eletrônica, ofertada pelo Departamento Acadêmico de Engenharia Eletrônica, do Campus Curitiba da Universidade Tecnológica Federal do Paraná.

Prof. Luiz Fernando Copetti

CURITIBA

NOVEMBRO, 2020

SUMÁRIO

1 OBJETIVO

O projeto elaborado visa ao desenvolvimento de uma aplicação no kit FPGA Altera DE-2 (Cyclone II) com a utilização da placa e de seus periféricos para o estabelecimento de comunicação com um servidor através de TCP/IP com a geração, na placa, de valores que demonstrem as ondas triangular, quadrada e dente de serra.

2 ESPECIFICAÇÕES E DESENVOLVIMENTO DO PROJETO

Para a realização do projeto e comunicação, foi utilizado o Quartus II 13.0sp1, com as ferramentas relacionadas: Programmer, Qsys, Nios II SBT for Eclipse e SignalTap II Logic Analyzer.

Como base, foram utilizados os modelos fornecidos pelo professor, exemplos fornecidos na instalação do Quartus e o projeto de gerador de sinais desenvolvido como primeira parcial da disciplina, além de terem sido utilizados os conhecimentos adquiridos durante o desenvolvimento das demais atividades da disciplina. Os elementos de hardware já existentes no projeto estão, em sua maioria, em Verilog, enquanto os novos foram desenvolvidos em VHDL e o software foi desenvolvido em C.

A comunicação TCP/IP é realizada com um servidor local desenvolvido em Python 3 e ativado para o envio de requisições ao gerador, com a conexão à rede local por meio de cabos conectados ao modem/roteador.

O gerador desenvolvido recebe a requisição da forma de onda a ser gerada por meio de registradores de escrita e o valor gerado é gravado e lido no registrador de leitura. São três as opções de forma de onda disponíveis: (i) triangular, em que, a cada ciclo, o valor é incrementado em uma unidade de 0 a 255 e, a seguir, decrementado até 0; (ii) dente de serra, cujo valor é incrementado da mesma maneira, mas é zerado quando atinge o máximo; e (iii) quadrada, em que o valor gerado alterna entre os valores mínimo e máximo. Para a seleção da forma de onda, a entrada do gerador deve receber os valores 1,2 e 3, respectivamente, de modo que a saída é zerada no caso de outro valor informado.

3 FUNCIONAMENTO

Para estabelecer a comunicação entre servidor e placa, o servidor deve estar ativo, conforme detalhamento da seção 4:

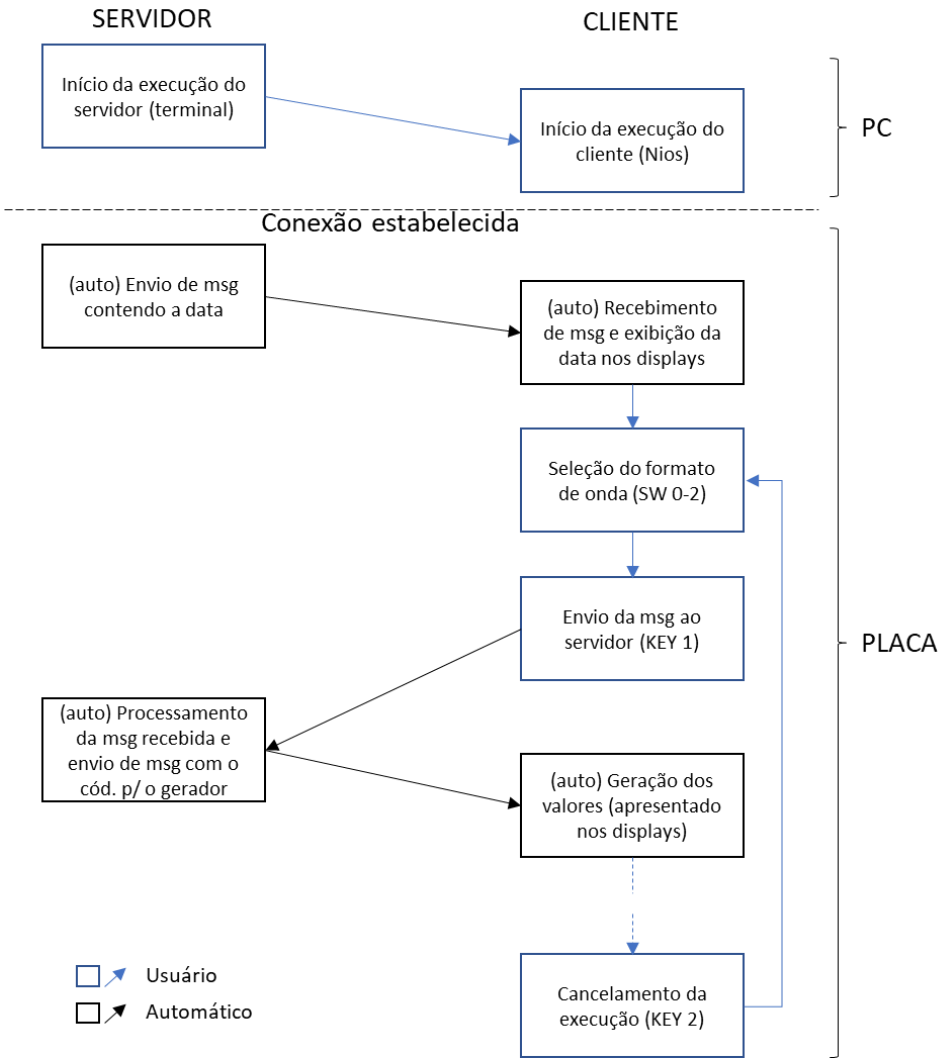

Figura 1. Fluxograma de funcionamento.

Após o estabelecimento da comunicação com o servidor, a seleção da forma de onda é realizada no próprio gerador, por meio do acionamento dos switches 0, 1 e 2 e informada ao servidor através do pressionamento do botão key 1 – ao se alterar a seleção, a informação é exibida no display LCD da placa.

Com o envio da comunicação ao servidor, este retorna uma mensagem informando o valor corresponde ao código de cada formato de onda, com a

consequente ativação da flag de execução e exibição do valor gerado nos displays de sete segmentos integrados à placa.

É possível parar a execução ao pressionar o botão key 2, com o retorno ao menu de opções inicial. Também é possível desconectar-se do servidor ativando-se os switches 0, 1 e 2, e pressionando-se o botão key 1, com o envio de mensagem ao servidor para fechamento da comunicação com o cliente.

4 EXECUÇÃO

Para rodar o projeto, é necessário ter instalado o Quartus II 13.0sp1 e Python 3 na máquina, devendo-se seguir os passos adiante enumerados:

1) Descompactar o arquivo '*projeto_final_wave_gen.zip'*, obtendo-se as pastas '*DE_NET*', '*bin*' e o arquivo '*wg_tcp_server.py*';
2) Carregar o '*sof'* na placa:
    a. Ligar a placa;
    b. Iniciar o Quartus;
    c. No menu Tools, iniciar o Programmer;
    d. Clicar em '*Add File*' e selecionar o arquivo '*DE2_NET_time_limited.sof*' na pasta '*...\WG2\bin*' (clicar em Ok, se aparecer uma janela);
    e. Marcar a caixa '*Program/Configure*' e clicar em '*Start'*;
3) Iniciar o servidor Python (no terminal, navegar até a pasta e utilizar o comando '*python wg_tcp_server.py*';
4) Carregar o '*.elf*' na placa:
    a. Abrir o '*Nios II 13.0sp1 Command Shell*' (não iniciar o Eclipse, realizar busca no menu iniciar)
    b. Na janela de "terminal" aberta, navegar até a pasta '*...\WG2\bin*'
    c. Executar o comando *'nios2-download -g wgen.elf && nios2-terminal.exe*'
    d. Digitar o endereço IP e a porta do servidor quando solicitado (não há feedback visual no Shell);
5) Pronto! Caso esteja tudo certo, uma mensagem de conexão será exibida no console do servidor e a data aparecerá nos displays da placa.

Para recompilar o código ou fazer alterações no software, o procedimento é um pouco diferente:

1) Descompactar o arquivo "*projeto_final_wave_gen.zip"*, obtendo-se as pastas "*DE_NET*", "*bin*" e o arquivo "*wg_tcp_server.py*";
2) Abrir o projeto "*DE_NET.qpf*" no Quartus;
3) No menu "*Tools*", abrir o "*Qsys*":
    a. Selecionar o arquivo "*system_0.qsys*" (na pasta "*DE2_NET*");
    b. Na aba "*Generation*", clicar em "*Generate*";
    c. Após o final da geração, clicar em "close" e fechar o Qsys.
4) No "*Project navigator*", na aba "*Files*", clicar com o botão direito na pasta e clicar em "*Add/Remove files in Project*":
    a. Navegar até a pasta "*DE2_NET\system_0*", selecionar o arquivo "*system_0.qip*" e clicar em *Abrir*, *Add*, *Apply* e *Ok*.
5) Compilar o projeto todo (Ctrl + L);
6) No menu Tools, iniciar o Programmer (clicar em Ok, se aparecer uma janela) e clicar em Start;
7) Com o Programmer e a janela OpenCore Plus Status ainda abertos, iniciar (no menu Tools) o Nios II STB for Eclipse;
8) O workspace deve ser a pasta ...\DE2_NET\software;
9) A aplicação simple_socket e simple_socket_bsp devem aparecer no workspace (caso não apareçam, importar no menu File);
10) Dar build nos projetos (Build All = Ctrl +B);
11) Criar um novo projeto em File -> New -> New Nios Application and BSP from Template:
    a. Selecionar o ".*sopcinfo*" correspondente (atenção para a pasta correta!);
    b. Definir um nome qualquer para o projeto (não precisa ser relacionado com as entidades VHDL);
    c. Utilizar a mesma pasta do workspace para o projeto;
    d. Selecionar o "Simple Socket Server" (1ª opção - não o RGMII);
    e. Clicar em "Finish".

12) Criado o projeto, copiar os arquivos da pasta “base” (em software) para a pasta do projeto criado substituindo os arquivos existentes (apenas da aplicação, pois o BSP deve permanecer inalterado);
13) Dar build no novo projeto e seu bsp;
14) Iniciar o servidor Python (no terminal, navegar até a pasta e utilizar o comando “python wg_tcp_server.py”;
15) No Eclipse, no menu Run -> Run Configurations:
    a. Clicar 2x em Nios II hardware para gerar “New Configuration”;
    b. Na guia Project, selecionar o projeto e seu .*elf*;
    c. Na guia Target Connection selecionar a placa e marcar todas as caixas de seleção exceto “*Disable ‘Nios II Console’ view*”;
16) Clicar em *Apply* e *Run*;
17) O projeto deverá ser executado, e no console será necessário digitar o endereço IP e a porta do servidor quando solicitado.
    a. Caso não ocorra a conexão nas primeiras tentativas, repetir os passos 6, 10 e 15.

5 CONCLUSÕES

O desenvolvimento do projeto permitiu compreender melhor o funcionamento dos componentes de hardware dos sistemas digitais modernos e sua lógica de programação, o que só foi possível com o desenvolvimento das primeiras atividades da disciplina, responsáveis pelo aprofundamento no conhecimento da linguagem VHDL e das ferramentas ora utilizadas.

Em se tratando de um sistema descontinuado pelo fabricante, houve grande dificuldade para utilização de exemplos e partes de código disponíveis, ante a inexistência de suporte para o dispositivo utilizado. Além disso, limitações do computador e do modem/roteador utilizados no projeto, houve complicações no estabelecimento da conexão entre placa e servidor, cuja superação foi possível com o suporte fornecido pelo professor.

De maneira geral, o resultado obtido com o projeto foi bastante satisfatório, com o comportamento do componente desenvolvido conforme o esperado e tendo sido possível desenvolver e ampliar os conhecimentos necessários ao projeto, simulação e programação de sistemas digitais.