Universidade de São Paulo

Escola Politécnica, Engenharia de Computação
PCS3645 - Laboratório Digital II
Turma 3 - Professor Paulo Cugnasca
Bancada B6

Experiência 04

Trena Digital com Saída Serial – Relatório

Nome Completo	**N USP**
Henrique Freire da Silva	12555551
Mariana Dutra Diniz Costa	12550841

São Paulo, 10 de Outubro de 2023

Sumário

1 INTRODUÇÃO

No presente relato, será apresentado o projeto de um circuito digital para integração de um sensor ultrassônico, através de sua interface, a um transmissor por barramento serial.

Com o objetivo de síntese e contínuo desenvolvimento desse módulo para demais experiências, as seguintes seções compreendem uma etapa lógica do princípio de funcionamento do circuito, seguida da aplicação em VHDL, simulação com o *ModelSim*, síntese com o *Quartus Prime* e, finalmente, verificação por meio de testes práticos.

1.1 Requisitos Funcionais

A cada medição, o circuito deve registrar o valor modulado pelo sensor HC-SR04, apresentá-lo em *displays* de sete segmentos e transmiti-lo pelo cabo serial para depuração em terminal. A medida transmitida ao terminal deve corresponder ao valor numérico apresentado em placa. A interface de interação do usuário deve ser realizada através da placa programada, de forma que medidas e *resets* possam ser acionados por chaves e botões.

1.2 Requisitos Não Funcionais

A arquitetura proposta deve descrever um modelo estrutural com módulos de fluxo de dados e unidade de controle. O circuito deve ser de fácil depuração, em que há sinais duplicados para verificação por osciloscópio de sinais emitidos e recebidos e a representação da medida amostrada deve: nos *displays*, estar na base decimal (codificação BCD) e em *cm*; no terminal, ser codificada em ASCII e possuir um caractere separador a cada ciclo de transmissão. Ademais, o projeto deve ser descrito em VHDL e sintetizado pelo *Quartus Prime* para a placa DE0-CV.

2 DESCRIÇÃO DO PROJETO

O projeto discorrido nas seguintes seções descreve as conexões expostas pelo circuito que combina a interface do sensor ultrassônico ao módulo de transmissão serial, bem como a lógica adotada em etapas de processamento entre os componentes mencionados e demais fluxos.

2.1 Projeto do Circuito da Trena Digital

A entidade principal da trena desenvolvida consiste de uma arquitetura que conecta o seu fluxo de dados e unidade de controle às portas declaradas (Apêndice A.1).

Os sinais "echo"e "trigger"são redirecionados à interface para conexão com o sensor (*interface_hcsr04*), enquanto a "saida_serial"é recebida do transmissor (*tx_serial_7O1*) e conectada ao dispositivo de saída pelo barramento serial. Os demais *outputs* são apresentados em placa.

Devido à natureza *pull-up* do botão de acionamento do sistema ("mensurar"), o sinal é internamente invertido e posto em um detector de borda, tal que se detecte uma única ação e não se realize mais de um ciclo de medição e transmissão.

2.1.1 Unidade de controle

A unidade de controle desenvolvida reage a cada ciclo de medição e transmissão dos componentes integrados à trena. A entidade desse módulo segue o Apêndice A.2.

A transição de estados de execução segue o algoritmo seguinte, descrito em linguagem natural:

- Enquanto não receber o pulso PARTIDA, aguarda o acionamento do sistema;
- Inicia a medida de distância pelo sinal MEDIR;
- Aguarda o fim da medição;
- Levanta TRANSMITE e inicia um ciclo de transmissão de caractere*;
- Quando receber CHAR_ENVIADO, incrementa a contagem de transmissões performadas;
- Se DADO_ENVIADO não estiver ativo, retorna a [*];
- Gera um pulso do sinal PRONTO e aguarda uma próxima execução.

Esse mesmo comportamento pode ser verificado na arquitetura apresentada no Apêndice B.1 entre as linhas 23 e 60, em que é descrita a lógica de transições e as saídas de cada estado (máquina de Moore).

O sinal "medir"− gerado no estado "medida"− aciona o sensor através de sua interface projetada (descrita em A.4), de forma a iniciar a medição e demodulação do sinal coletado. A princípio, percebe-se que o sinal produzido não constitui um pulso unitário (largura de um ciclo de *clock*), porém essa é uma escolha de projeto que economiza um estado de controle e ainda impede a sobreposição de duas medições, visto que o fim de um primeiro processo performa a troca de estado (B.1, linha 28) e impede que o sinal seja reavaliado para um segundo processo.

Ademais, o sinal que determina o fim de uma sequência de transmissões é incrementado a cada ciclo e delimitado pelo próprio sinal de fim de contagem (*not dado_enviado*), tal que não se reinicie a contagem e se economize um bit de contagem (conforme seção 2.1.2).

2.1.2 Fluxo de dados

O fluxo de dados foi criado de forma a interligar a *interface_hcsr04* e o *tx_serial_7O1*, mencionados no topo do capítulo; um contador modular que computa a quantidade de caracteres já enviados; e um multiplexador 4x1 para selecionar o dígito a ser codificado e enviado. A entidade do módulo é descrita no Apêndice A.3.

A lógica de chaveamento e conversão dos dados a serem enviados via canal serial é como segue o trecho em B.2. Nessa passagem, percebe-se a ligação entre os componentes nessa seleção, a começar pelo elemento U3_COUNT (linha 3), que realiza a contagem, de 0 a 3, das transmissões realizadas. A saída produzida por esse contador ("s_char_select") é conectada ao seletor do multiplexador (linha 21), de forma que a parcela do sinal combinado "s_medida"a passar para o "s_digit"seja correspondente ao número da transmissão corrente.

A saída desse componente é então codificada pelas linhas 25 e 26, em que é concatenado o literal "001"à esquerda do sinal, de forma a incrementá-lo em 48_{10} (0x30) e converter o número contido em "s_digit"a seu correspondente símbolo ASCII. Como o sinal original contém apenas 12 bits e, portanto, 3 dígitos BCD, a quarta e última entrada do multiplexador é alocada a um valor arbitrário com representação hexadecimal não numérica. Dessa forma, esse caso é filtrado na condição imposta (*s_digit <= "1001"*) e convertido à representação ASCII da cerquilha ('#'), reaproveitando a lógica aplicada para o caso particular esperado de caractere separador entre ciclos de transmissão.

Vale ressaltar que o efeito da construção do sinal "conta_char"da unidade de controle (B.1, linha 57) com base no próprio fim do contador descrito resulta em um componente menor ao evitar contagem até 4 para registrar o último envio serial, visto que em outro caso teria que alterar o contador para 3 bits e a seleção no multiplexador seria menos direta.

2.2 Estratégia de Testes

O *testbench* do circuito de interface foi configurado para adotar os casos da Tabela 1. Para cada iteração rodada, a largura de pulso é definida de acordo com a segunda coluna da tabela.

A sequência de sinais e intervalos aplicados é como segue: espera-se pela próxima borda de descida do *clock* para sincronizar o início do teste; abaixa-se o sinal "mensurar"(Apêndice A.1) por 5 ciclos (botão em *pull-up*); espera-se um tempo de 50 μs (representativo do tempo real de medição, modulação e transmissão do sinal, porém reduzido para âmbito de simulação); produz-se o *echo* com a devida largura para a interface; aguarda pela sinalização de fim da transmissão completa do dado obtido (sinal "pronto", Apêndice A.1); e adiciona um preenchimento em tempo (100 μs) antes de repetir esses passos em uma nova iteração com outro caso de teste.

Código do teste	Largura de pulso (μs)	Largura de pulso (cm)	Leitura esperada (*displays* e terminal)
1	423	7.2	7
2	1200	20.4	20
3	2047	34.8	35
4	3412	58	58

Tabela 1: Casos de teste de medição definidos no *testbench*.

Com relação à metodologia de verificação prática dos testes propostos, serão realizados os seguintes passos:

1. *Reset* global do sistema pelo acionamento da chave SW0;
2. Posicionamento do objeto com auxílio de instrumento de medição (régua ou trena);

3. Acionamento do sinal de medida do botão KEY0;
4. Comparação entre a distância apresentada nos *displays* e aquela no terminal do computador;

Os passos 1-4 serão repetidos para cada caso de teste.

2.3 Simulação e Síntese do Design

Depois de definidos os casos de testes, conforme apresentados na Tabela 1 e descritos em *testbench*, o projeto foi integrado ao *software ModelSim* para fins de simulação. Desse modo, foi possível simular os cenários propostos e acompanhar o comportamento de cada sinal de entrada, saída e depuração ao longo do funcionamento do circuito.

Para o primeiro cenário, apresentado na Figura 1, o aperto do botão "mensurar"(nível baixo) no início do caso faz com que o sensor dispare o *trigger* de 10 μs (destacado em azul) e receba, no sentido de retorno, um pulso de *echo* com duração de 423 μs, o que corresponde a uma distância de 7.2 cm. O fim da medida é indicado pela ativação do sinal de depuração "Fim Medida", destacada em rosa, momento em que o resultado é armazenado em um registrador.

A partir desse momento, a UC do circuito passa do estado "medida"para "transmissão", iniciando a transmissão serial do resultado 007 calculado pelo circuito através do contador BCD. Essa transmissão ocorre caractere a caractere, enviando o código ASCII de cada um por meio do protocolo 7O1. O circuito, então, transmite respectivamente os caracteres da centena, dezena e unidade, finalizado com o "#"indicador de fim da mensagem. Após passar 4 vezes pelo estado de transmissão, o sinal "pronto"– indicador do fim – é ativado e o circuito retorna ao seu estado inicial até que uma nova requisição de medida seja feita.

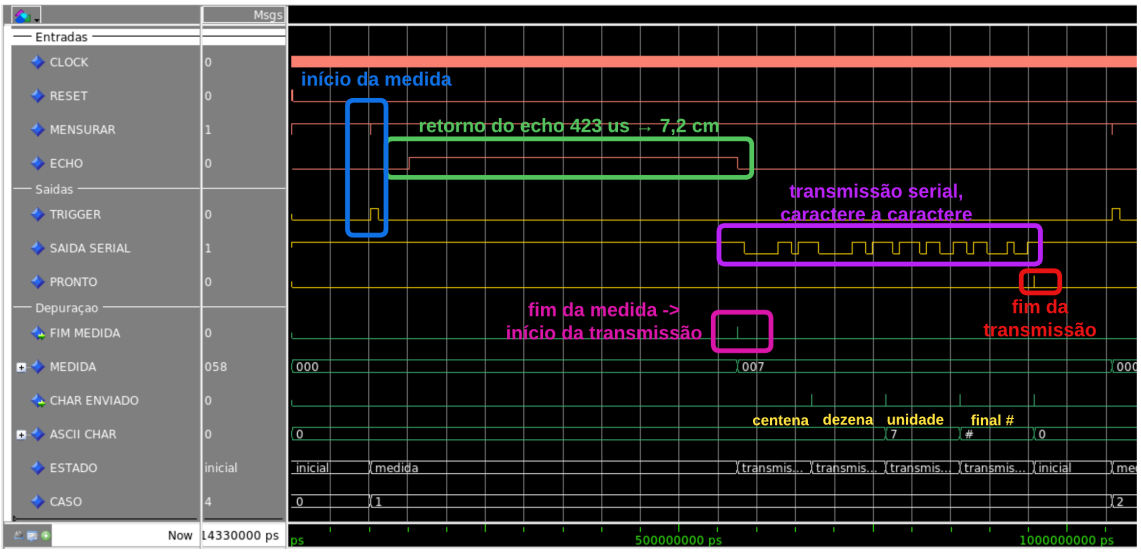

Figura 1: Formas de onda para o teste de distância 7.2cm.

As Figuras de 2 a 4 a seguir seguem a mesma lógica da primeira, representando sempre o disparo de *trigger* seguido do retorno *echo*, o registro do resultado da medida em centímetros inteiros e, finalmente, a transmissão serial de cada um dos quatro caracteres. É relevante notar, em cada uma das imagens de simulação, o funcionamento correto do mecanismo de arredondamento da medida obtida para o inteiro mais próximo encontrado e as transições de estado ocorrendo adequadamente, o que demonstra a boa integração entre os módulos do projeto.

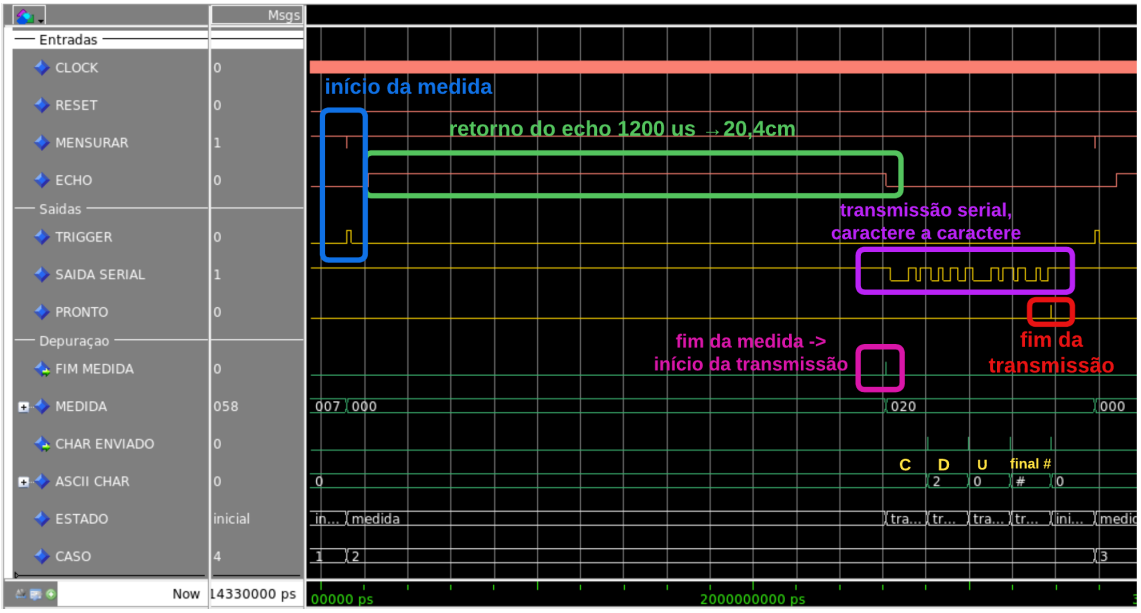

Figura 2: Formas de onda para o teste de distância 20.4cm.

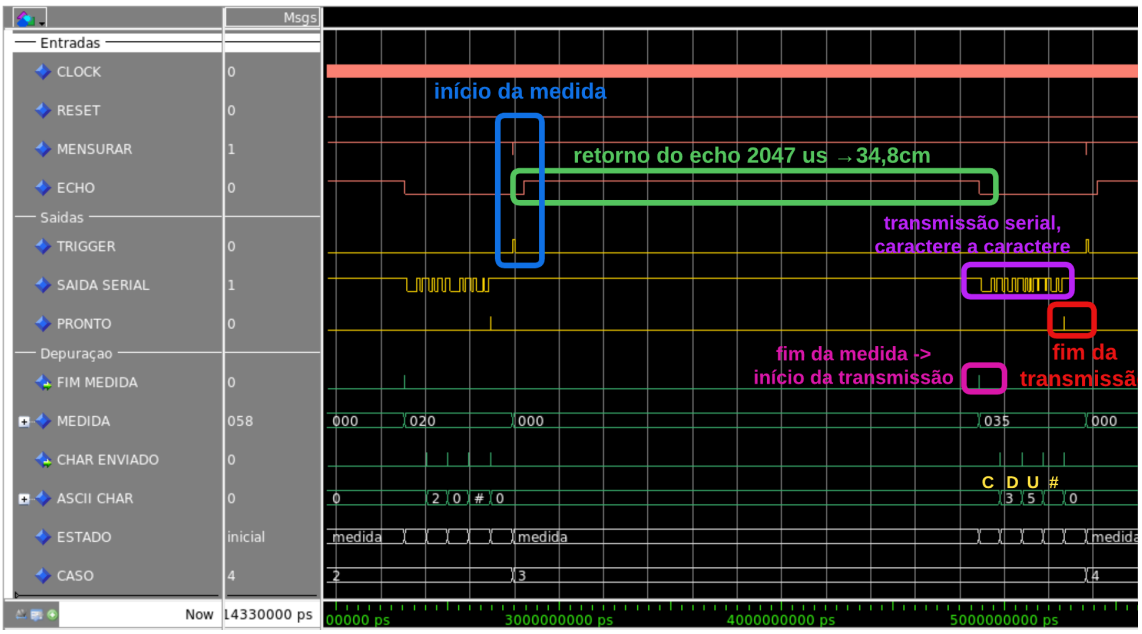

Figura 3: Formas de onda para o teste de distância 34.8cm.

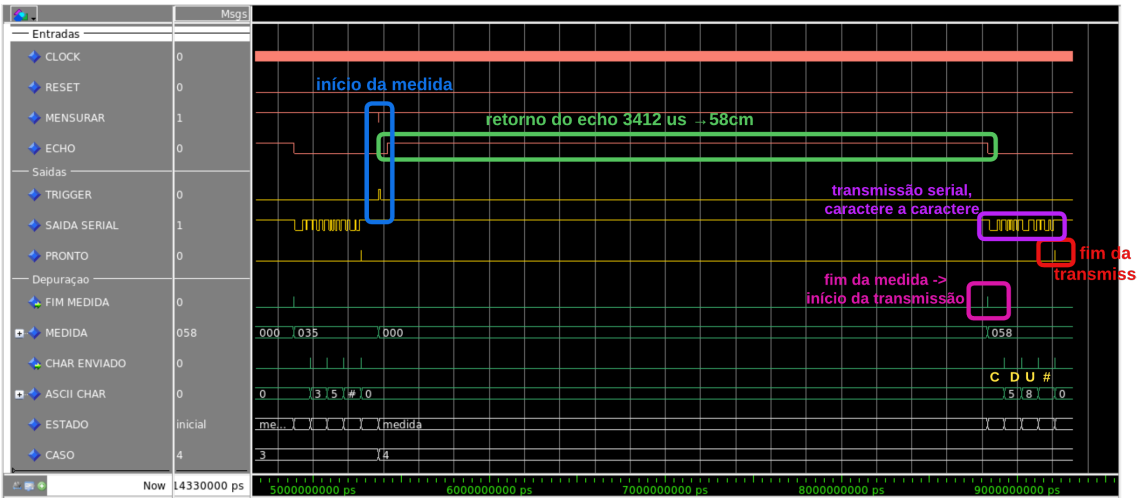

Figura 4: Formas de onda para o teste de distância 58cm.

Após realizar as simulações e confirmar, de fato, o bom funcionamento do circuito projetado para a integração do sensor ultrassônico com a porta serial, os componentes do projeto foram importados para um novo arquivo na ferramenta *Intel Quartus Prime*, onde foi possível analisar cada componente e sintetizar o circuito final.

Partindo dessa sintetização, foram criadas visualizações por meio da ferramenta *RTL viewer*, através da qual foi possível verificar os diagramas de bloco do circuito e confirmar sua compatibilidade com as especificações definidas no capítulo 2. Nesse sentido, foram ilustrados o circuito completo da trena digital (Figura 5) e os componentes formadores do Fluxo de Dados (Figura 6).

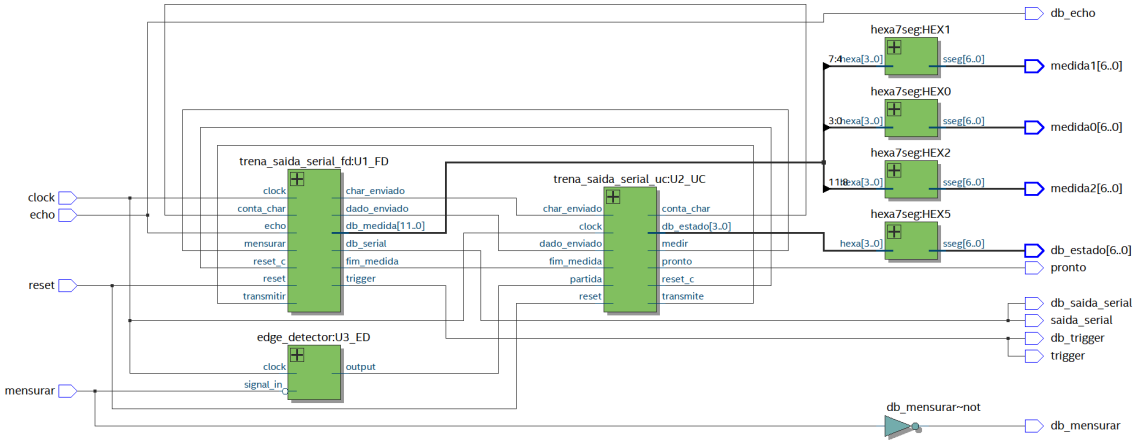

Figura 5: Diagrama de blocos do circuito completo da trena digital.

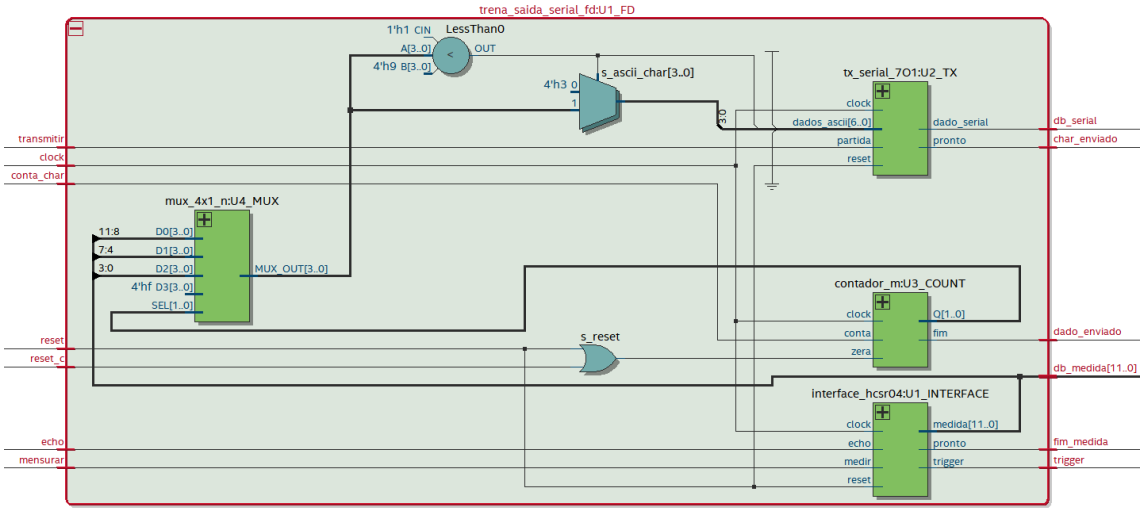

Figura 6: Diagrama de blocos do fluxo de dados da trena digital.

Com isso, a ferramenta interna *State Machine Viewer* foi utilizada para conferir a estrutura da máquina de estados que rege a Unidade de Controle do projeto, assim como as transições entre estados e os sinais de condição que influenciam no seu funcionamento. Esses aspectos estão representados na Figura 7.

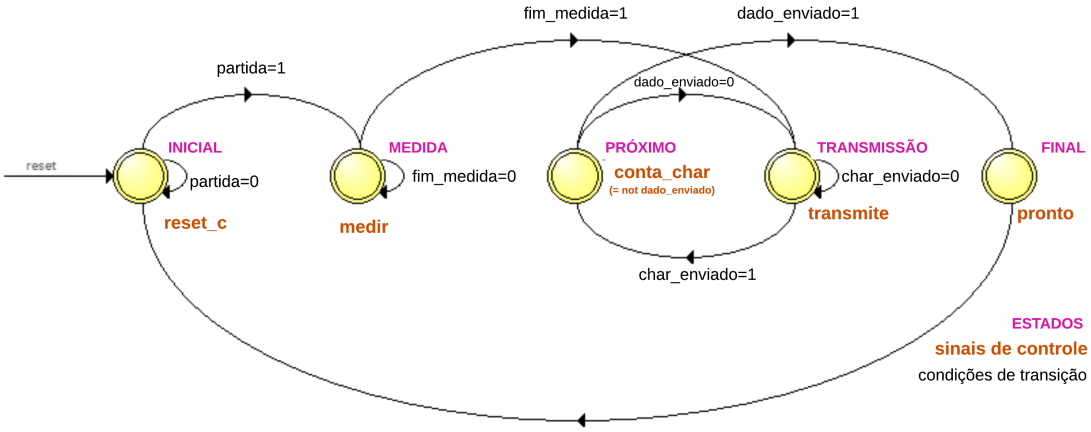

Figura 7: Máquina de Estados da Unidade de Controle do circuito completo da trena digital.

Finalmente, foi configurada a pinagem conforme a Tabela 2 descrita na Seção 3 e o projeto foi exportado para um arquivo de extensão *qar* que poderá ser diretamente utilizado em laboratório para elaboração na placa FPGA.

3 PLANEJAMENTO DA AULA PRÁTICA

Em laboratório, será realizada a síntese do projeto arquivado (extensão *qar*) na placa FPGA DE0-CV segundo a pinagem da Tabela 2.

A montagem experimental será, então, realizada em etapas intermediárias para testar o correto funcionamento de todos os componentes do projeto e suas relações. Esse processo seguirá de acordo com o aqui descrito Plano de Execução Experimental.

3.1 Plano de Execução Experimental

Inicialmente, será testada a integração com o sensor ultrassônico. Para isso, todos os componentes externos ao circuito projetado (Analog Discovery, módulo do sensor HC-SR04 e CI conversor de nível de tensão 74HC4050) serão referenciados em um mesmo potencial da placa programada (GND). Em seguida, os pinos de alimentação da FPGA serão ligados a esses componentes de acordo com o nível de tensão esperado: 3,3 V de referencial nas placas conversoras 74HC4050; e 5 V para o funcionamento do sensor ultrassônico. Nesse momento, o osciloscópio do Analog Discovery poderá ser usado para verificar a adequação da tensão dos sinais e garantir que não haverá sobrecarga.

Depois, será preciso testar a porta serial-USB, conectando o GND, TX e RX nos pinos correspondentes da placa MAX3232 e deixando em curto os terminais TD e RD. Do outro lado, a porta USB será conectada à entrada devida no computador. Com isso, a ferramenta *TeraTerm* será configurada de acordo com a codificação 7O1 utilizada e serão realizados testes de eco ao digitar um caractere ASCII no terminal, verificando a correta volta dele pela porta.

Após averiguação desses dois componentes e correção de quaisquer falhas, os sinais de interface do circuito poderão, então, ser conectados. A saída *trigger* na ponte da FPGA pode ser ligada diretamente ao pino de mesmo nome no sensor ultrassônico, o *echo* do sensor deve ser ligado à entrada de um par entrada-saída do CI 7HC4050 e coletado em seu dual para a entrada do sinal *echo* efetivo do circuito pela porta B12.

Em seguida, os sinais de depuração - *db_trigger* e *db_echo* - serão associados aos canais +1 e +2 do Analog Discovery, respectivamente, para possibilitar a depuração via osciloscópio da ferramenta *Scope*. A saída serial do circuito (porta B16 da FPGA), por sua vez, deve ser direcionada à entrada RD do CI MAX3232, enquanto a saída RX do conversor será ligada diretamente à porta serial-USB para o computador, onde esse sinal poderá ser depurado via *TeraTerm*, comparando-o com a medida aparente dos *displays*. É importante recordar, desde o início, de referenciar todos os componentes experimentais a um mesmo potencial terra (GND).

A partir desse momento, a placa será ligada e programada com o projeto em questão para a realização dos testes propostos (Tabela 1).

Sinal	Ligação na placa FPGA	Pino na FPGA	Analog Discovery
clock	CLOCK_50	PIN_M9	-
reset	chave SW0	PIN_U13	-
mensurar	botão KEY0	PIN_U7	-
medida[0]	display HEX0	PIN_U21 PIN_V21 PIN_W22 PIN_W21 PIN_Y22 PIN_Y21 PIN_AA22	-
medida[1]	display HEX1	PIN_AA20 PIN_AB20 PIN_AA19 PIN_AA18 PIN_AB18 PIN_AA17 PIN_U22	-
medida[2]	display HEX2	PIN_Y19 PIN_AB17 PIN_AA10 PIN_Y14 PIN_V14 PIN_AB22 PIN_AB21	-
saida_serial	GPIO_0_D1	PIN_B16	-
trigger	GPIO_1_D1	PIN_A12	-
echo	GPIO_1_D3	PIN_B12	-
pronto	led LEDR[0]	PIN_AA2	-
db_mensurar	led LEDR[1]	PIN_AA1	-
db_saida_serial	GPIO_0_D35	PIN_T15	-
db_trigger	GPIO_1_D33	PIN_G12	CH1+ (Scope)
db_echo	GPIO_1_D35	PIN_K16	CH2+ (Scope)
db_estado	display HEX5	PIN_N9 PIN_M8 PIN_T14 PIN_P14 PIN_C1 PIN_C2 PIN_W19	-

Tabela 2: Pinagem para a montagem experimental.

4 ATIVIDADES EXPERIMENTAIS

Em laboratório, o circuito final do projeto foi sintetizado e programado na placa FPGA DE0-CV. O primeiro teste foi realizado com a montagem experimental parcial, inicialmente com a saída GPIO do sinal de *trigger* conectada ao *trigger* do sensor e, este, ligado ao Canal 1 do osciloscópio do *Analog Discovery*. No Canal 2 do osciloscópio, foi capturado o sinal de *echo* saído do sensor HC-SR04, finalizado a primeira montagem definida no Plano de Execução Experimental, ilustrada na figura 8.

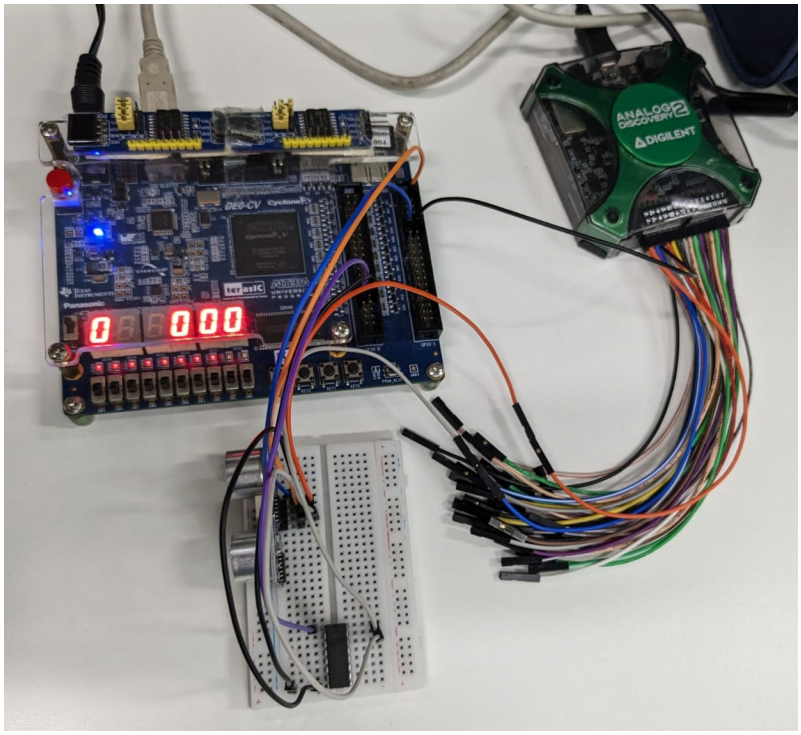

Figura 8: Montagem experimental parcial para teste do sensor HC-SR04.

Com isso, foi verificado o envio correto do Echo pelo sensor ao receber o sinal de Trigger, adequando a largura do pulso à distância medida pelo ultrassom. A partir das ferramentas disponíveis no *Scope* do Analog Discovery, foi possível verificar a tensão adequada próxima a 3V dos sinais, além da correta definição das larguras de Trigger (10 μs) e Echo (conforme distância), como representado na figura 9.

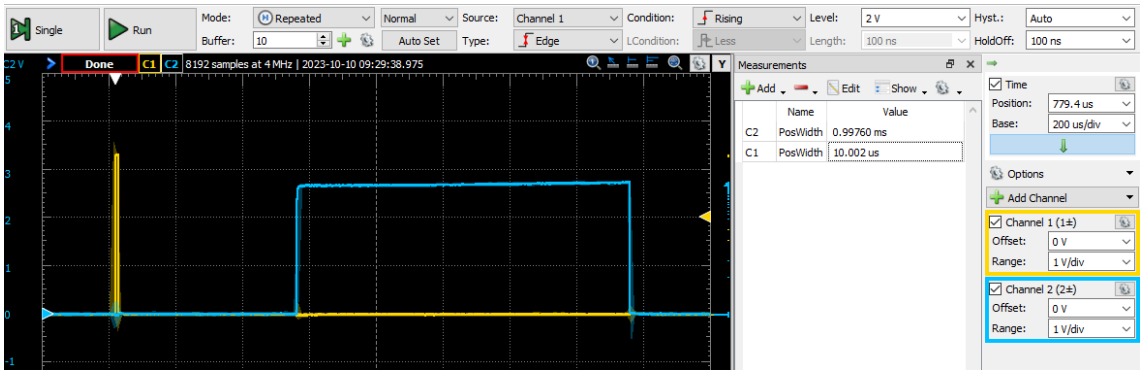

Figura 9: Sinais do osciloscópio para a relação entre Trigger e Echo.

Em seguida, incrementou-se a montagem intermediária para testar a comunicação serial, como mostra a figura 8. Nesse sentido, as referências de Terra foram unificadas, os terminais RD e TD foram colocados em curto e a porta serial foi conectada à entrada USB computador. Com isso, o terminal TeraTerm foi utilizado e com ele verificado a transmissão de caracteres via serial no protocolo 7O1.

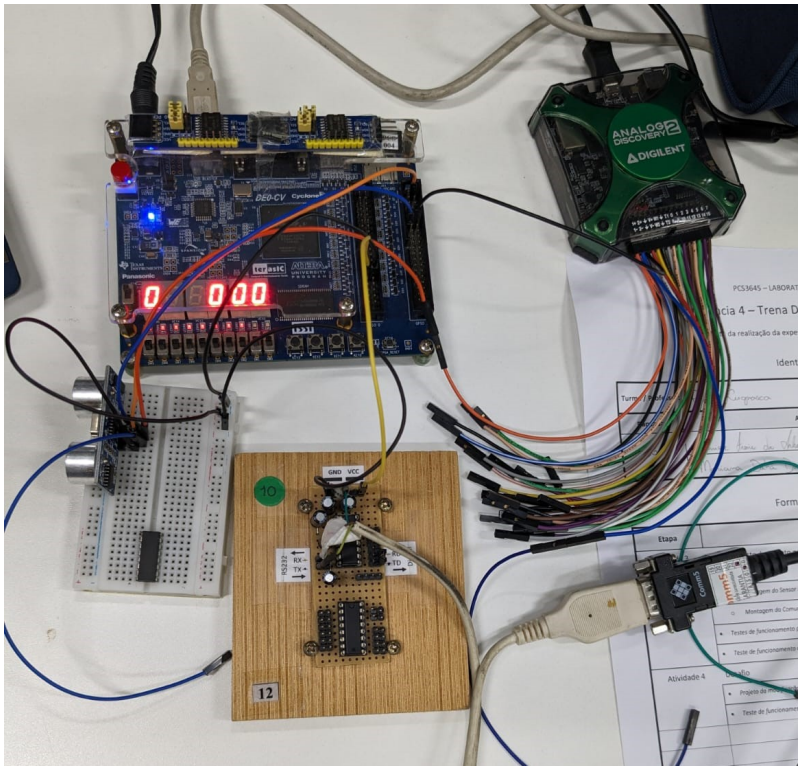

Figura 10: Montagem experimental parcial para teste da porta serial.

Com a devida montagem do circuito em etapas e verificação do funcionamento intermediário de cada componente integrado, foi possível finalizar a montagem integrando todos os módulos, conforme a figura 11, e realizar os testes propostos na Tabela 1. Nesse contexto, foi extraído o programa *processing*, a partir do qual a saída serial no monitor do computador foi comparada à medida

apresentada pelos *displays* de 7 segmentos.

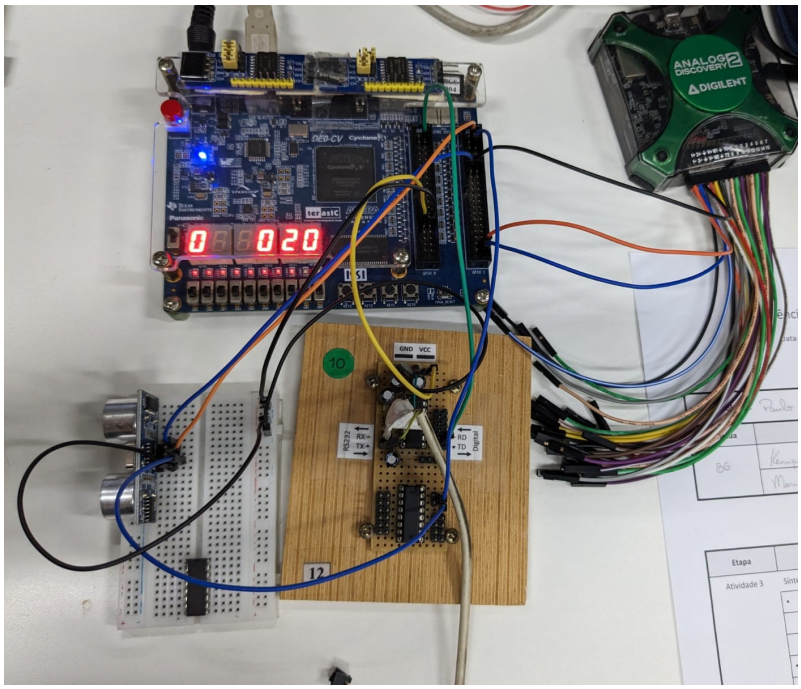

Figura 11: Montagem experimental final para a trena digital com saída serial.

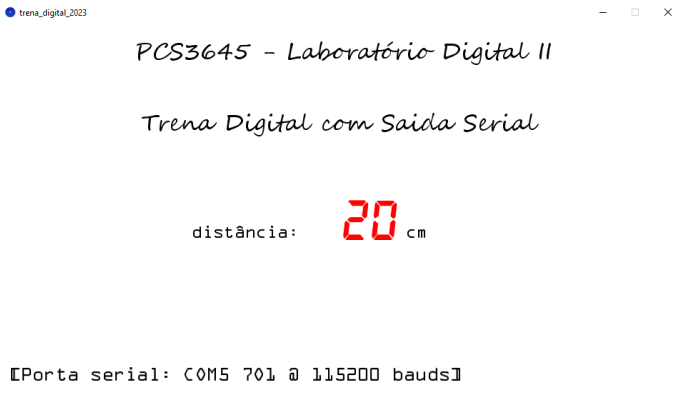

Figura 12: Captura de tela do programa Processing.

5 DESAFIO

A proposta de desafio foi de sintetizar um novo modo de operação ao circuito descrito em 2.1, em que os ciclos de medição e transmissão são realizados de forma cíclica e automática, a cada 2 segundos. Nesse modo, o acionamento do botão de medida pelo usuário não afeta o tempo entre medidas.

Uma chave foi associada à escolha de modo, conforme entidades atualizadas em A.1 e A.2. O sinal é condicionado na transição de estados da unidade de controle, de forma a esperar por uma partida manual (modo 0) ou o fim de uma contagem externa ao módulo (modo 1), vide B.1 linhas 25-27.

No fluxo de dados, há um novo contador modular com os parâmetros M e N iguais a $10\hat{8}$ e 27, respectivamente, tal que o período até que a contagem termine se iguale a 2 segundos, como especificado. Esse contador está ativo (contando) enquanto o circuito se encontra em estado de espera ("inicial"no arquitetura em B.1).

6 CONSIDERAÇÕES FINAIS

Ao longo da montagem prática, a utilização de técnicas de depuração, tal como o uso de osciloscópio pela ferramenta *Scope* na verificação da largura de pulso e nível de tensão, auxiliou na validação do circuito desenvolvido e em um aprimoramento do procedimento a ser aplicado em testes previamente planejados.

Foi possível constatar a importância dos testes unitários para verificar o funcionamento independente de cada componente antes de se realizar a integração dos módulos em um único contexto de projeto mais complexo. Nesse sentido, é relevante perceber o processo de construção incremental do projeto, que converge progressivamente à montagem final do sonar.

APÊNDICES

A Declaração de entidades da interface

A.1 Entidade principal

```
1   entity trena_saida_serial is
2       port (
3           -- inputs
4           clock         : in  std_logic;
5           reset         : in  std_logic;
6           modo          : in  std_logic;
7           mensurar      : in  std_logic;
8           echo          : in  std_logic;
9           -- outputs
10          trigger       : out std_logic;
11          saida_serial  : out std_logic;
12          medida0       : out std_logic_vector(6 downto 0);
13          medida1       : out std_logic_vector(6 downto 0);
14          medida2       : out std_logic_vector(6 downto 0);
15          pronto        : out std_logic;
16          -- debug
17          db_mensurar      : out std_logic;
18          db_saida_serial  : out std_logic;
19          db_trigger       : out std_logic;
20          db_echo          : out std_logic;
21          db_estado        : out std_logic_vector(6 downto 0)
22      );
23  end trena_saida_serial;
```

A.2 Unidade de controle

```
1   entity trena_saida_serial_uc is
2       port (
3           -- inputs
4           clock         : in std_logic;
5           reset         : in std_logic;
6           modo          : in std_logic;
7           partida       : in std_logic;
8           fim_espera    : in std_logic;
9           fim_medida    : in std_logic;
10          char_enviado  : in std_logic;
11          medir         : in std_logic;
12          dado_enviado  : in std_logic;
13          -- outputs
14          reset_c     : out std_logic;
15          transmite   : out std_logic;
16          conta_char  : out std_logic;
17          pronto      : out std_logic;
```

```
18              -- debug
19              db_estado   : out std_logic_vector(3 downto 0)
20          );
21      end entity;
```

A.3 Fluxo de dados

```
1       entity trena_saida_serial_fd is
2           port (
3               -- inputs
4               clock         : in  std_logic;
5               reset         : in  std_logic;
6               echo          : in  std_logic;
7               mensurar      : in  std_logic;
8               transmitir    : in  std_logic;
9               conta_char    : in  std_logic;
10              -- outputs
11              trigger       : out std_logic;
12              fim_espera    : out std_logic;
13              fim_medida    : out std_logic;
14              char_enviado  : out std_logic;
15              dado_enviado  : out std_logic;
16              -- debug
17              db_serial     : out std_logic;
18              db_medida     : out std_logic_vector(11 downto 0)
19          );
20      end entity;
```

A.4 Interface HC-SR04

```
1       entity interface_hcsr04 is
2           port (
3               clock      : in  std_logic;
4               reset      : in  std_logic;
5               medir      : in  std_logic;
6               echo       : in  std_logic;
7               trigger    : out std_logic;
8               medida     : out std_logic_vector(11 downto 0);
9               pronto     : out std_logic
10          );
11      end entity interface_hcsr04;
```

B Implementação de arquiteturas

B.1 Unidade de controle

```
1  architecture trena_saida_serial_uc_arch of trena_saida_serial_uc is
2
3      type tipo_estado is (inicial, medida, transmissao, proximo, final
           );
4      signal Eatual: tipo_estado;  -- estado atual
5      signal Eprox:  tipo_estado;  -- proximo estado
6
7  begin
8
9    -- memoria de estado
10   process (reset, clock)
11   begin
12       if reset = '1' then
13           Eatual <= inicial;
14       elsif clock'event and clock = '1' then
15           Eatual <= Eprox;
16       end if;
17   end process;
18
19   -- logica de proximo estado
20   process (partida, fim_medida, char_enviado, dado_enviado, Eatual)
21   begin
22
23     case Eatual is
24
25       when inicial     =>  if (partida='1' and modo='0') or
26                               (fim_espera='1' and modo='1')
27                              then Eprox <= medida;
28                            else                Eprox <= inicial;
29                            end if;
30       when medida      =>  if fim_medida='1' then
31                                               Eprox <= transmissao;
32                            else               Eprox <= medida;
33                            end if;
34       when transmissao =>  if char_enviado='1' then
35                                               Eprox <= proximo;
36                            else               Eprox <= transmissao;
37                            end if;
38       when proximo     =>  if dado_enviado='1' then Eprox <= final;
39                            else                    Eprox <=
                                  transmissao;
40                            end if;
41       when final       =>  Eprox <= inicial;
42       when others      =>  Eprox <= inicial;
43
44     end case;
45
46   end process;
```

```
47
48      -- logica de saida (Moore)
49      with Eatual select
50          reset_c    <= '1' when inicial, '0' when others;
51
52      with Eatual select
53          medir      <= '1' when medida, '0' when others;
54
55      with Eatual select
56          transmite  <= '1' when transmissao, '0' when others;
57
58      with Eatual select
59          conta_char <= not dado_enviado when proximo, '0' when others;
60
61      with Eatual select
62          pronto     <= '1' when final, '0' when others;
63
64      with Eatual select
65          db_estado <= "0000" when inicial,
66                       "0001" when medida,
67                       "0010" when transmissao,
68                       "0100" when proximo,
69                       "1111" when final,     -- Final
70                       "1110" when others;    -- Erro
71
72  end architecture;
```

B.2 Escolha e codificação de caracteres no fluxo de dados

```
1       ...
2
3       U3_COUNT: contador_m
4           generic map (M => 4, N => 2)
5           port map (
6               clock => clock,
7               zera  => s_reset,
8               conta => conta_char,
9               Q     => s_char_select,
10              fim   => dado_enviado,
11              meio  => open
12          );
13
14      U4_MUX: mux_4x1_n
15          generic map (bits => 4)
16          port map (
17              d3      => "1111",
18              d2      => s_medida(3 downto 0),
19              d1      => s_medida(7 downto 4),
20              d0      => s_medida(11 downto 8),
21              sel     => s_char_select,
22              mux_out => s_digit
```

```
23          );
24
25      s_ascii_char <= "011" & s_digit when s_digit <= "1001"
26                                          else "0100011";
```