Sistema de Sonar
(Parte 1)

Versão LabEAD 2021

RESUMO

Esta experiência tem por objetivo iniciar o desenvolvimento de um circuito que realiza a varredura e a detecção de objetos próximo com a utilização de um sensor ultrassônico de distância e de um servomotor. Nesta primeira parte, os circuitos das experiências anteriores deverão ser revisados e alguns novos componentes deverão ser implementados. Todos estes componentes deverão ser testados na placa FPGA. O projeto do sistema de sonar deverá ser realizada na segunda parte e depois sintetizado na placa FPGA DE0-CV, usando a infraestrutura de bancada remota do LabEAD e MQTT Dash.

OBJETIVOS

Após a conclusão desta experiência, os seguintes tópicos devem ser conhecidos pelos alunos:

- *Aplicação da refatoração de código para revisar projetos funcionais;*
- *Desenvolvimento de circuitos de teste, para a composição de múltiplos componentes;*
- *Realização de testes de unidade e testes de integração de circuitos digitais;*
- *Projeto de circuitos em FPGA.*

1. ESPECIFICAÇÃO DO PROJETO

Sistemas digitais conhecidos como **radar, lidar** ou **sonar** podem ser caracterizados com um sistema que tem como função principal a detecção de objetos a distância. Inicialmente, o termo "*radar*" foi criado a partir do acrônimo da expressão em inglês "*radio detection and ranging*" que, em tradução livre, significa "*detecção e localização por rádio (frequência)*". Esta detecção pode ser realizada através de ondas eletromagnéticas que são emitidas pelo radar, refletidas nos objetos distantes e recebidas por sensores. A detecção destes objetos permite a localização e a medida de sua distância. Sistemas veiculares autônomos em desenvolvimento por diversas empresas e grupos de pesquisa podem usar outros tipos de sistemas de detecção. Uma das alternativas é o **lidar** (*light detection and ranging*). Neste sistema, a distância a objetos é verificada com uso de um *laser*. Dependendo do sinal eletromagnético usado, um sistema de detecção de objetos pode ser nomeado de forma diferente. Por exemplo, no caso dos submarinos, usam-se ondas acústicas para propagação na água, e seu sistema é chamado **sonar** (*sound navigation and ranging*). Dependendo da frequência acústica usada temos sistemas infrassônicos (baixas frequências) ou sistemas ultrassônicos (altas frequências). Nesta experiência, usaremos o sensor ultrassônico de distância HC-SR04, que trabalha com pulsos ultrassônicos de 40KHz.

1.1. Interface do Circuito

O projeto desta experiência visa desenvolver um circuito digital que permite rastrear objetos através da medida de distância. A interface do circuito deve seguir os sinais apresentados na figura 1. O processo de medida de distância é executado com auxílio de um sensor ultrassônico.

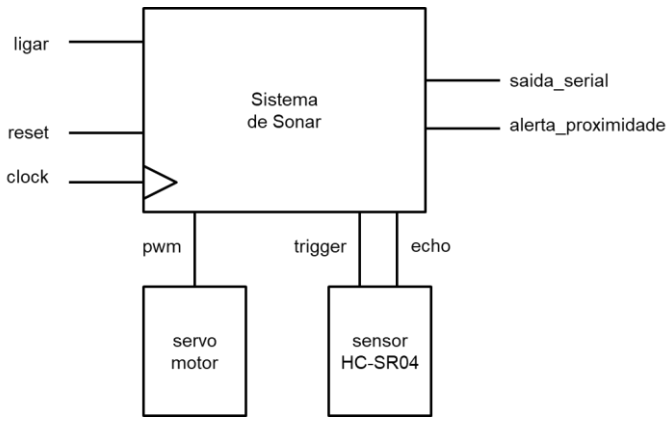


Figura 1. Interface do sistema de Sonar.

O servomotor deve posicionar o sensor ultrassônico de distância para a localização de objetos. A montagem deve permitir uma variação angular dentro dos limites especificados para o servomotor. A figura 2 ilustra uma possível montagem física.

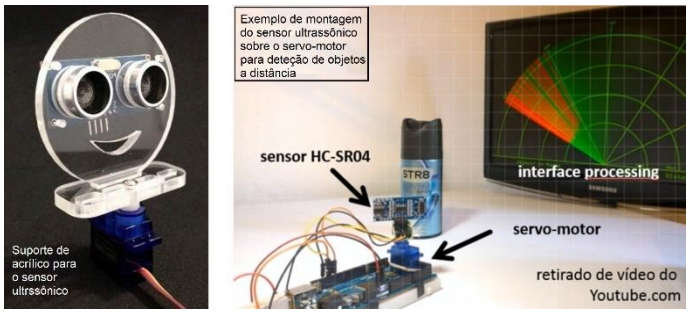


Figura 2. Uma possível montagem do servomotor com o sensor ultrassônico.

A figura acima também apresenta os principais elementos do projeto e sua interação durante seu funcionamento. O servomotor é acoplado ao sensor HC-SR04 e permite sua rotação em relação ao seu eixo. A cada posição angular estabelecida, a distância ao objeto mais próximo deve ser medida. Em seguida, um bloco de informação composto por posição angular e distância deve ser enviado pela interface serial, que posteriormente será representado graficamente na tela do computador.

1.2. Descrição Sucinta do Funcionamento

O circuito deve somente iniciar sua operação com o acionamento do sinal LIGAR. Há qualquer momento, o desacionamento do sinal LIGAR deve interromper o funcionamento do sistema. No modo de localização, o sistema deve continuamente realizar o rastreamento e a medição de distância a uma taxa de 1 medida/segundo. O modo atual de funcionamento do circuito deve ser apresentado no *led* de saída MODO (ligado ou desligado). A saída do circuito SAIDA_SERIAL é um sinal RS-232C que deve ser conectado a uma porta serial e a informação enviada é composta por dois valores: o ângulo e a distância ao objeto nesta posição. Esta saída deve ser transmitida por um sinal RS232-C em formato "`ângulo,distância.`", usando caracteres ASCII. Cada informação (ângulo e distância) deve ser composta por 3 dígitos BCD em código ASCII, totalizando 8 dados ASCII enviados. Por exemplo, uma saída indicando um objeto na posição a 153º a 17 cm de distância deve ser composta pela sequência de caracteres ASCII "`153,017.`". Quando o circuito detecta um objeto a menos de 20 cm, a saída ALERTA_PROXIMIDADE deve ser ativada.

1.3. Considerações para o Desenvolvimento do Projeto

Seguem abaixo algumas considerações sobre o desenvolvimento do projeto do circuito do sistema de sonar.

1.3.1. Refatoração de Código

Inicialmente, deve-se executar a **refatoração do código fonte** dos circuitos projetados nas experiências passadas. Embora a refatoração seja uma atividade mais ampla em projetos de Engenharia (de Software, principalmente), que envolve a alteração do código fonte (do projeto ou programa) visando por exemplo a melhora da legibilidade e entendimento ou de sua eficiência, usaremos aqui a técnica para revisar o desenvolvimento dos componentes de forma a adequá-los para as próximas experiências.

Convém mencionar que qualquer refatoração, revisão ou modificação de código deve ser seguida da realização do **teste** deste código, uma vez que ele foi alterado de alguma forma. Recomenda-se o uso do ModelSim para a execução dos testes para verificação de funcionamento após as refatorações.

A) Revisão do circuito de controle do servomotor

A revisão desse componente envolve uma modificação além da revisão de código. O circuito de controle do servomotor (`controle_servo.vhd`) deve ser modificado para gerar o sinal PWM de saída com as seguintes larguras de pulso.

posicao	largura do pulso (ms)	ciclos de *clock*
000	1	50.000
001	1,143	57.143
010	1,286	
011	1,429	
100	1,571	
101	1,714	
110	1,857	
111	2	10.000

A entidade refatorada `controle_servo_3` deve seguir a seguinte interface.

```
entity controle_servo_3 is
    port(
        clock:      in  std_logic;
        reset:      in  std_logic;
        posicao:    in  std_logic_vector(2 downto 0);
        pwm:        out std_logic;
        db_reset:   out std_logic;
        db_pwm:     out std_logic;
        db_posicao: out std_logic_vector(2 downto 0)
    );
end entity;
```

Os sinais de depuração `db_reset` e `db_posicao` foram pré-definidos para serem usados na depuração do circuito em caso de mal funcionamento na placa FPGA. O sinal `db_reset` pode ser usado para verificar a conexão com o MQTT Dash, e os sinais `db_posicao` e `db_pwm` servem para depurar o circuito. Por exemplo, o sinal `db_pwm` pode ser ligado em osciloscópio.

Para a realização de testes do componente refatorado usando o **ModelSim**, recomenda-se o desenvolvimento de *testbench* que gere o sinal de saída para as 8 posições definidas. Isso pode ser realizado em um único teste incluindo todos os casos de testes, um para cada posição do servomotor.

B) Revisão do circuito de interface com o sensor ultrassônico de distância

A funcionalidade do circuito interno da interface com o sensor ultrassônico de distância HC-SR04 (`interface_hcsr04.vhd`) não precisa ser modificada para esta experiência. Recomenda-se apenas realizar uma revisão do código.

A entidade refatorada de `interface_hcsr04` deve seguir a seguinte interface.

```
entity interface_hcsr04 is
    port (
        clock:      in  std_logic;
        reset:      in  std_logic;
        medir:      in  std_logic;
        echo:       in  std_logic;
        trigger:    out std_logic;
        medida:     out std_logic_vector(11 downto 0);
        pronto:     out std_logic;
        db_reset:   out std_logic;
        db_medir:   out std_logic;
        db_estado:  out std_logic_vector(3 downto 0)
    );
end entity;
```

Os sinais de depuração definidos servem para os procedimentos de verificação de mal funcionamento na placa FPGA. Por exemplo, os sinais `db_reset` e `db_medir` podem ser usados para verificar a conexão com o MQTT Dash, ao passo que o sinal `db_estado` pode ser usado para checar o funcionamento interno do circuito sequencial.

Este componente refatorado pode ser testado e verificado com o ModelSim usando como base o *testbench* fornecido na experiência "*Interface com Sensor Ultrassônico de Distância*".

C) Revisão dos circuitos de comunicação serial

Os circuitos de transmissão serial assíncrona (`tx_serial_8N2.vhd`) e de recepção serial assíncrona (`rx_serial_8N2.vhd`) com o modo de transmissão 8N2, com dados de 8 *bits*, a 9600 *bauds* devem ser refatorados antes de serem usados como componentes internos. Os elementos para interface com dispositivos externos, como por exemplo, detector de borda (usado para tratar pulsos largos de entrada) e codificadores de saída para *displays* de 7 segmentos, devem ser retirados destes componentes.

A definição das entidades revisadas é apresentada a seguir.

```
entity tx_serial_8N2 is
    port (
        clock:           in  std_logic;
        reset:           in  std_logic;
        partida:         in  std_logic;
        dados_ascii:     in  std_logic_vector(7 downto 0);
        saida_serial:    out std_logic;
        pronto_tx:       out std_logic;
        db_partida:      out std_logic;
        db_saida_serial: out std_logic;
        db_estado:       out std_logic_vector(3 downto 0)
      );
end entity;

entity rx_serial_8N2 is
    port (
        clock:           in  std_logic;
        reset:           in  std_logic;
        dado_serial:     in  std_logic;
        recebe_dado:     in  std_logic;
        dado_recebido:   out std_logic_vector(7 downto 0);
        tem_dado:        out std_logic;
        pronto_rx:       out std_logic;
        db_recebe_dado:  out std_logic;
        db_dado_serial:  out std_logic;
        db_estado:       out std_logic_vector(3 downto 0)
    );
end entity;
```

Os sinais de depuração `db_partida` e `db_recebe_dado` podem ser usados para verificar a conexão com o MQTT Dash. Já os sinais `db_saida_serial` e `db_dado_serial` podem ser conectados em osciloscópio para verificar os sinais seriais. E os respectivos sinais `db_estado` de ambos os componentes podem verificar o funcionamento de cada circuito sequencial.

Os respectivos *testbenches* aplicados nas experiências de comunicação serial devem ser adaptados para as simulações com o ModelSim.

1.3.2. Desenvolvimento de Novos Componentes

Na próxima etapa de projeto, novos componentes precisam ser desenvolvidos para compor o sistema de sonar. Estes componentes serão responsáveis pela comunicação com a saída serial do sistema.

D) UART (*Universal Asynchronous Receiver Transmitter*)

A comunicação serial do sistema de sonar com a porta serial do computador deve ser realizada por um único componente denominado `uart_8N2`. Este componente deve ser criado como uma composição dos circuitos de transmissão e de recepção serial assíncronos, conforme ilustrado na figura 4.

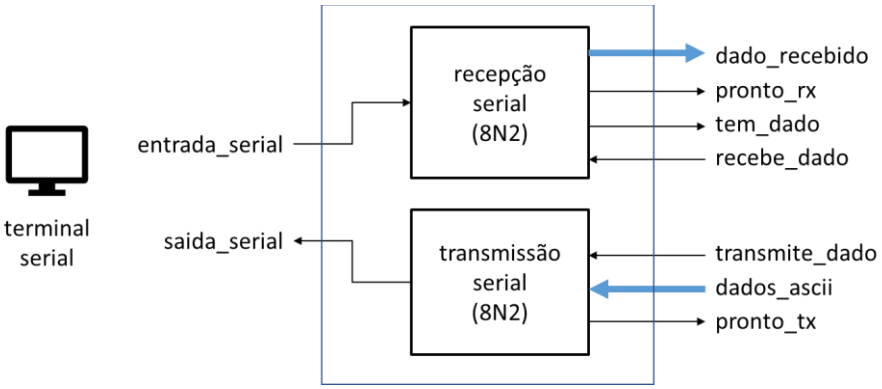


Figura 4 – Estrutura interna e interface de sinais do componente `uart_8N2`.

A interface de sinais do componente `uart_8N2` deve seguir a seguinte especificação.

```
entity uart_8N2 is
  port (
    clock             : in  std_logic;
    reset             : in  std_logic;
    transmite_dado    : in  std_logic;
    dados_ascii       : in  std_logic_vector(7 downto 0);
    dado_serial       : in  std_logic;
    recebe_dado       : in  std_logic;
    saida_serial      : out std_logic;
    pronto_tx         : out std_logic;
    dado_recebido_rx  : out std_logic_vector(7 downto 0);
    tem_dado          : out std_logic;
    pronto_rx         : out std_logic;
    db_transmite_dado : out std_logic;
    db_saida_serial   : out std_logic;
    db_estado_tx      : out std_logic_vector(3 downto 0);
    db_recebe_dado    : out std_logic;
    db_dado_serial    : out std_logic;
    db_estado_rx      : out std_logic_vector(3 downto 0)
    );
end entity;
```

E) Transmissão de dados seriais do sonar

Os dados seriais de ângulo e distância para o objeto medidos pelo sistema de sonar devem ser transmitidos para uma saída serial e podem ser apresentados por uma interface gráfica em um computador, usando o componente `uart_8N2` (definido acima), através do componente `tx_dados_sonar`. A entidade deve seguir a especificação apresentada a seguir.

```
entity tx_dados_sonar is
  port (
    clock:           in  std_logic;
    reset:           in  std_logic;
    transmitir:      in  std_logic;
    angulo2:         in  std_logic_vector(3 downto 0); -- digitos BCD
    angulo1:         in  std_logic_vector(3 downto 0); -- de angulo
    angulo0:         in  std_logic_vector(3 downto 0);
    distancia2:      in  std_logic_vector(3 downto 0); -- e de distancia
    distancia1:      in  std_logic_vector(3 downto 0);
    distancia0:      in  std_logic_vector(3 downto 0);
    saida_serial:    out std_logic;
    pronto:          out std_logic;
    db_transmitir:   out std_logic;
    db_saida_serial: out std_logic;
    db_estado:       out std_logic_vector(3 downto 0)
  );
end entity;
```

O sinal TRANSMITIR inicia a transmissão dos 8 caracteres ASCII no formato "$a_2a_1a_0,d_2d_1d_0$.", onde os dígitos BCD de ângulo e de distância devem ser primeiramente codificados para código ASCII.

DICA1: a **codificação de dígitos BCD para código ASCII** pode ser facilmente realizada justapondo os *bits* "011" antes do código BCD. Por exemplo, o dígito "5" possui código BCD igual a "0101" (5_{16}). Se concatenarmos "011" (3_{16}) com "0101" (5_{16}) temos o código binário para o ASCII de 7 *bits* "0110101" (35_{16}).

DICA2: o operador VHDL para concatenação de vetores de *bits* é **&**. Por exemplo, os vetores de *bits* "011" e "0101" são concatenados pelo comando VHDL:

```
ascii <= "011" & "0101";
```

de forma a obter o conteúdo "0110101" em `ascii`.

Uma alternativa de projeto para a seleção dos caracteres ASCII para transmissão pela saída serial é usar um multiplexador 8x1 com sua entrada de seleção controlada por um contador. A figura 5 ilustra esta alternativa.

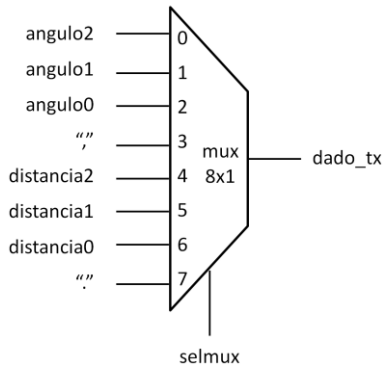


Figura 5 – Circuito de seleção de caracteres ASCII dos dados do sistema de sonar.

Outros componentes podem ser criados no Projeto do Grupo. Estes componentes foram especificados para auxiliar o desenvolvimento do projeto do sistema de sonar.

2. PARTE EXPERIMENTAL

2.1. Atividade 1 – Refatoração de Código e Projeto de Novos Componentes

a) Desenvolva a refatoração de código dos componentes desenvolvidos nas experiências anteriores. Documente no Planejamento os resultados desta refatoração, descrevendo as modificações que foram necessárias.

b) Documente a simulação dos componentes refatorados usando o **ModelSim**. Anexe as formas de onda obtidas, junto com uma breve descrição dos sinais das cartas de tempo. DICA: anote as figuras com observações dos resultados obtidos.

c) Projete os novos componentes descritos na seção 1.4.2. Documente o funcionamento destes componentes.

d) Defina casos de teste destes novos componentes.

e) Execute a simulação dos casos de teste definidos para cada componente usando o ModelSim. Anote as figuras das formas de onda obtidas para mostrar o correto funcionamento.

f) Submeter os arquivos VHDL dos componentes refatorados, junto com os respectivos *testbenches* (arquivos ZIP).

2.2. Atividade 2 – Preparação dos Circuitos de Teste dos Módulos

A atividade experimental visa testar os códigos refatorados e os novos componentes projetados nesta experiência. O objetivo desta atividade é validar o funcionamento de todos os componentes através do desenvolvimento de um conjunto de circuitos de teste na bancada remota do Laboratório Digital.

DICA: nomeie os projetos dos circuitos de teste: `teste_movimentacao_servomotor` e `teste_tx_dados_sonar`.

g) Projetar um circuito de teste que execute a movimentação contínua do servomotor percorrendo as 8 posições definidas. A movimentação deve seguir um padrão de movimento do "vai e volta", com intervalo de 1 segundo em cada posição.

   DICA: o componente `contador_g_updown_m` será fornecido.

h) Projetar um circuito de teste que envia a sequência de 8 caracteres ASCII com dados de ângulo e distância do sistema de sonar.

   DICA: o componente `mux_8x1_n` será fornecido.

i) Execute o teste de funcionamento destes circuitos de teste usando o ModelSim.

   DICA: nomeie os projetos dos circuitos de teste: `teste_movimentacao_servomotor` e `teste_tx_dados_sonar`.

j) Submeta os arquivos QAR dos projetos dos circuitos de teste com o Planejamento.

   DICA: nomeie os arquivos como `exp5_txbyy_teste_movimentacao_servomotor.qar` e `exp5_txbyy_teste_tx_dados_sonar.qar`.

2.3. Atividade 3 – Montagem Experimental no Laboratório

O objetivo desta atividade é validar o funcionamento de todos os componentes através da síntese do circuito na bancada remota do Laboratório Digital.

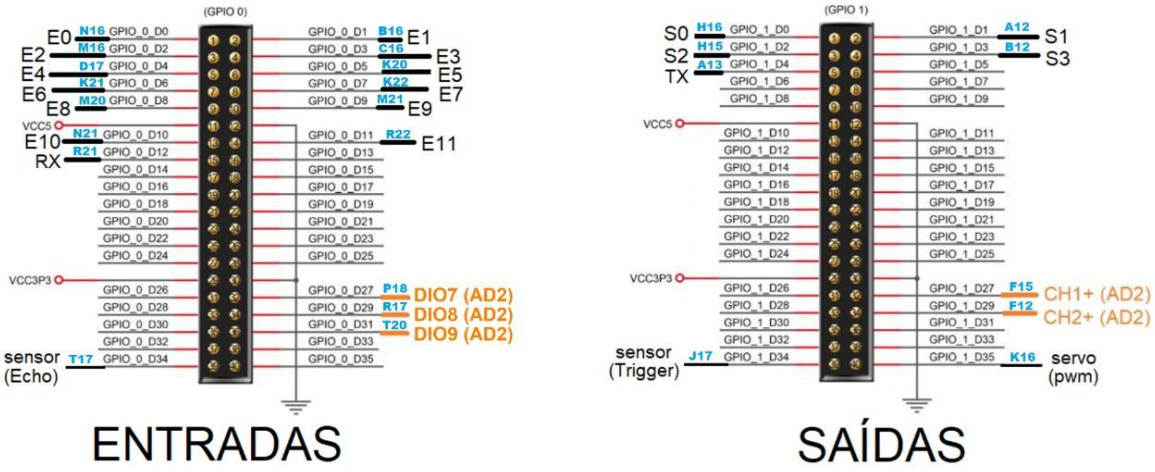
Cada grupo deve:

1) Elaborar um Plano de ações para validar cada módulo desenvolvido;
2) Elaborar um Plano de Testes dos circuitos na placa FPGA da bancada remota, usando qualquer um dos recursos disponíveis (p.ex. Analog Discovery, MQTT Dash);
3) Escolher o Roteiro de Testes dos componentes, com a definição da sequência de componentes validados a cada teste executado com sucesso;
4) Executar o roteiro definido durante a experiência;
5) Comprovar o correto funcionamento de todos os componentes ao professor.

O Planejamento deve incluir uma descrição detalhada de todas as etapas de teste dos componentes, assim como dos resultados esperados em cada um deles.

k) Os **circuitos de teste** projetados devem ser implementados na bancada remota no Laboratório Digital usando a placa FPGA DE0-CV. Defina também as **designações de sinais** aos pinos da FPGA e aos pinos dos projetos no MQTT Dash (se for usado).

Dica: use as designações definidas a partir da experiência 3 e a informação de mapeamento de sinais da placa FPGA e ao ESP8266 da plataforma LabEAD são mostradas abaixo.



l) Relate quaisquer ocorrências experimentais no Relatório.

m) Submeta os arquivos QAR dos projetos finais da experiência com o Relatório.

3. BIBLIOGRAFIA

- ALMEIDA, F.V. de; SATO, L.M.; MIDORIKAWA, E.T. **Tutorial para criação de circuitos digitais em VHDL no Quartus Prime 16.1**. Apostila de Laboratório Digital. Departamento de Engenharia de Computação e Sistemas Digitais, Escola Politécnica da USP. Edição de 2017.
- ALTERA. **DE0-CV User Manual**. 2015.
- ALTERA. **Quartus Prime Introduction Using VHDL Designs.** 2016.
- ALTERA. **Quartus Prime Introduction to Simulation of VHDL Designs.** 2016.
- D'AMORE, R. **VHDL - descrição e síntese de circuitos digitais**. 2ª edição, LTC, 2012.
- MEALY, B.; TAPPERO, F. **Free Range VHDL** - *The no-frills guide to writing powerful code for your digital implementations*. Free Range Factory, Janeiro de 2018 (v.1.21).
- MIDORIKAWA, E.T. **Metodologia de Projeto com Dispositivos Programáveis**. Apostila de Laboratório Digital. PCS-EPUSP, 2016.
- PROCESSING. Site do programa *processing*. `http://processing.org`. Acesso em 23/09/2021.
- TOCCI, R. J.; WIDMER, N.S.; MOSS, G.L. **Sistemas Digitais: Princípios e Aplicações**. Prentice-Hall, 11ª ed., 2011.
- VAHID, F. **Sistemas Digitais: Projeto, Otimização e HDLs**. Bookman, 2008.
- WAKERLY, John F. **Digital Design Principles & Practices**. 4th edition, Prentice Hall, 2006.

4. MATERIAL DISPONÍVEL

- Circuito Integrado: 74HC4050 (*buffer* e conversão de tensão de 5V para 3,3V)
- 1 sensor ultrassônico HC-SR04
- 1 servomotor
- 1 *protoboard* ou outra plataforma de montagem
- 1 *kit* Lab do LabEAD

5. EQUIPAMENTOS NECESSÁRIOS

- 1 computador com interface serial, software de comunicação serial e softwares Intel Quartus Prime e ModelSim
- 1 dispositivo Analog Discovery da Digilent.
- 1 placa de desenvolvimento FPGA DE0-CV com o dispositivo Cyclone V 5CEBA4F23C7N.

Histórico de Revisões

E.T.M./2015 – versão inicial
E.T.M./2019 – revisão e atualização
E.T.M./2020 – revisão e reorganização da experiência para acesso remoto.
E.T.M./2021 – revisão