UNIVERSIDADE ESTADUAL PAULISTA “JULIO DE MESQUITA FILHO”
FACULDADE DE ENGENHARIA
CAMPUS DE ILHA SOLTEIRA

ALISON FRANÇA QUEIROZ DA COSTA

DESENVOLVIMENTO DE COMUNICAÇÃO DE DADOS SEM FIO PARA PROBLEMAS DE DETECÇÃO DE VAZAMENTOS EM TUBULAÇÕES ENTERRADAS

Ilha Solteira

2018

ALISON FRANÇA QUEIROZ DA COSTA

DESENVOLVIMENTO DE COMUNICAÇÃO DE DADOS SEM FIO PARA PROBLEMAS DE DETECÇÃO DE VAZAMENTOS EM TUBULAÇÕES ENTERRADAS

Dissertação apresentada à Faculdade de Engenharia – UNESP – Campus de Ilha Solteira, como parte dos requisitos para obtenção do título de Mestre em Engenharia Elétrica.
Área de Conhecimento: Automação

Prof. Dr. Aílton Akira Shinoda
Orientador

Ilha Solteira

2018

FICHA CATALOGRÁFICA
Desenvolvido pelo Serviço Técnico de Biblioteca e Documentação

Costa, Alison França Queiroz da.
C837d Desenvolvimento de comunicação de dados sem fio para problemas de detecção de vazamentos em tubulações enterradas / Alison França Queiroz da Costa. -- Ilha Solteira: [s.n.], 2018
76 f. : il.

Dissertação (mestrado) - Universidade Estadual Paulista. Faculdade de Engenharia de Ilha Solteira. Área de conhecimento: Automação, 2018

Orientador: Aílton Akira Shinoda
Inclui bibliografia

1. Soc Zynq. 2. Sincronismo. 3. Correlacionador de sinais.

Raiane da Silva Santos
Supervisora Técnica da Seção
Seção Técnica de Referência, Atendimento ao usuário e Documentação
Diretoria Técnica de Biblioteca e Documentação
CRB/8 - 8900

unesp

UNIVERSIDADE ESTADUAL PAULISTA

Câmpus de Ilha Solteira

CERTIFICADO DE APROVAÇÃO

TÍTULO DA DISSERTAÇÃO: Desenvolvimento de comunicação de dados sem fio para problemas de detecção de vazamentos em tubulações enterradas

AUTOR: ALISON FRANÇA QUEIROZ DA COSTA
ORIENTADOR: AILTON AKIRA SHINODA

Aprovado como parte das exigências para obtenção do Título de Mestre em ENGENHARIA ELÉTRICA, área: Automação pela Comissão Examinadora:

Prof. Dr. AILTON AKIRA SHINODA
Departamento de Engenharia Elétrica / Faculdade de Engenharia de Ilha Solteira

Prof. Dr. FABRICIO CESAR LOBATO DE ALMEIDA
Departamento de Engenharia de Biossistemas / Faculdade de Ciências e Engenharia de Tupã

Prof. Dr. CRISTIANO QUEVEDO ANDREA
Centro de Ciências Exatas e Tecnologia / Universidade Federal de Mato Grosso do Sul

Ilha Solteira, 21 de dezembro de 2018

DEDICO

A minha mãe Greicy Mara França, que me educou e me possibilitou mais essa conquista.

AGRADECIMENTOS

A minha mãe, Greicy Mara França, a qual sempre me apoiou em minha carreira acadêmica e esteve presente em toda minha vida, cuidando e educando.

Ao professor Dr. Aílton Akira Shinoda pela sua orientação e todo conhecimento repassado para a elaboração deste trabalho.

Aos professores participantes do projeto, pela ajuda e conhecimento repassado durante todo o tempo juntos no projeto.

Aos acadêmicos da mecânica pela ajuda no desenvolvimento do projeto, nas discussões sobre o desenvolvimento do protótipo e na ajuda na realização dos testes experimentais.

A Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) pelo auxílio financeiro vinculado ao processo 2016/24974-2.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

RESUMO

Este trabalho desenvolveu um protótipo de um correlacionador de sinais aplicados a vazamento em tubulações de água aterrada. Para tal foi analisado o SoC Zynq, utilizando a placa Minized juntamente com um conversor analógico-digital, Pmod AD1, um módulo GPS, Pmod GPS, um *clock* em tempo real, Pmod RTCC e um modulo acelerômetro Pmod NAV. Nesta análise foram utilizados modelos criados no *software* Xilinx Vivado, baseados em *Intellectual Property blocks,* e o *software* Xilinx SDK. A partir dos modelos desenvolvidos foram realizados testes com o propósito de validar o sincronismo entre as placas. Inicialmente testou-se um modelo simples apenas utilizando a placa e o módulo Pmod AD1, sendo o teste realizado para verificar a resolução do relógio da placa Minized. Em seguida foi feito o experimento com a placa Minized e o Pmod RTCC visando o sincronismo entre as placas. Os testes de sincronismo foram finalizados utilizando-se o conjunto Minized, Pmod RTCC e Pmod GPS, que obteve bons resultados. Para a finalização da análise da correlação de sinais verificou-se a utilização do Pmod NAV para a aquisição de dados com testes em bancada, apresentando resultados experimentais satisfatórios.

Palavras-Chave: SoC Zynq. Sincronismo. Correlacionador de sinais.

ABSTRACT

This work developed a prototype of a signal correlator applied to leakage in buried water pipes. The Zynq SoC was analyzed using the Minized board together with an analog-to-digital converter, Pmod AD1, a module GPS, Pmod GPS, a real-time clock, Pmod RTCC and a Pmod NAV accelerometer module. For this analysis we used models created in the Xilinx Vivado software, based on Intellectual Property blocks, and the Xilinx SDK software. From the developed models tests were carried out in order to validate the synchronism between the boards. Initially, a simple model was tested using only the board and the module Pmod AD1, and the test was performed to verify the resolution of the Minized board's clock. Then the experiment was performed using the Minized board and the Pmod RTCC aiming the synchronism between the boards. In order to finalize the synchronization tests, we used the Minized set, Pmod RTCC and Pmod GPS, which presented good results. For the conclusion of the analysis of the correlation of signals, the use of the Pmod NAV for the data acquisition was verified and then benchmarked, presenting satisfactory experimental results.

Keywords: SoC Zynq. Synchronism. Ssignal correlator.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

ACP	Accelerator Coherency Port
APU	Application Processor Unit
AXI	Advanced Extensible Interface
BLE	Bluetooth Low Energy
BSP	Board Support Package
CAN	Controller Area Network
CLB	Configurable Logic Block
DDR3L	Double Date Rate Low-power
EDR	Enhanced Data Rate
eMMC	Embedded Multi-Media Controller/Card
FF	Flip-flop
FPGA	Field Programmable Gate Array
FPU	Floating Point Unit
FTDI	Join Test Action Group
GCC	GNU Compiler Connection
GPIO	General Purpose Input/Output
GPS	Global Position System
IDE	Integrated Development Enviroment
IIC	Inter-Integrated Circuit
IOBs	Input/Output Blocks
IP	Intellectual Property
JTAG	Join Test Action Group
LED	Light Emitting Diode
LUT	Look-up Table
MEMS	Microelectromechanical system
MISO	Master Input Slave Output
MMU	Memory Management Unit
MOSI	Master Output Slaver Input
MPE	Media Processing Engine
OCM	On Chip Memory
PL	Programmable Logic

PMIC	Power-management Integrated Circuit
PS	Processing System
RTCC	Real Time Clock/Calendar
Rootfs	Root File System
RTOS	Real-time Operation System
SCL	Serial Clock
SCU	Snoop Control Unit
SD	Secure Digital
SDA	Serial Data
SDK	Software Development Kit
SNIS	Sistema Nacional de Informações sobre Saneamento
SO	Sistema Operacional
SoC	System on Chip
SS	Slave Select
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
ZedBoard	Zynq Evaluation & Development Board

SUMÁRIO

1 INTRODUÇÃO

Atualmente a água potável é um recurso de extrema importância para a sociedade, visto que tal recurso está cada vez mais escasso em determinados locais,

gerando a necessidade de medidas que combatam o desperdício da mesma. No Brasil, segundo o Sistema Nacional de Informações sobre Saneamento (SNIS), cerca de 38,1% da produção de água potável é perdida ao longo da distribuição da mesma até os consumidores, é descrito na Tabela 1 as perdas de água em cada região do Brasil (SNIS, 2016).

Tabela 1 - Porcentagem regional de perda de água

Região	Porcentagem (%)
Norte	52,8
Nordeste	47,3
Sudeste	33,0
Sul	37,0
Centro-Oeste	33,2
Brasil	38,1

Fonte: (SNIS, 2016)

Segundo o SNIS, o país vem ao longo dos anos diminuindo as perdas de água ocasionadas na distribuição da mesma, porém este valor de 38,1% está muito elevado comparado a outros países. A Alemanha e o Japão reduziram suas perdas para uma média de 10%, além disso Austrália e Nova Zelândia apresentaram média abaixo dos 10% (SNIS, 2016).

Pensando na redução das perdas provenientes de vazamentos em tubulações aterradas, ao longo dos anos foram realizadas diversas pesquisas, segundo Hunaidi *et al.* (2014) os principais métodos são baseados em equipamentos, podendo ser acústicos ou não, sendo eles:

Equipamentos acústicos:

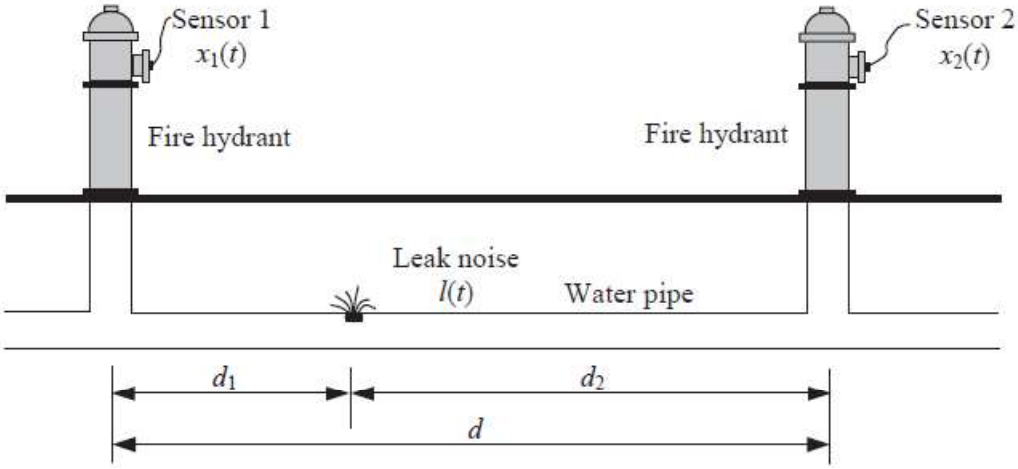
- Dispositivos de escuta: composto por uma haste de escuta e microfones capazes de escutar ruídos nas tubulações. Este método baseia-se em percorrer uma área referente a tubulação e, através de pontos de encaixe ou até mesmo pela superfície do solo, analisar o ruído provocado pelo vazamento de água. A eficácia deste método é oriunda da experiência do operador em identificar os diversos ruídos, ser capaz de diferenciar ruídos de vazamento, ruídos externos entre outros.
- Registradores de ruído: composto por um sensor de vibração (ou hidrofone) e um registrador de dados programável. São utilizados para detectar vazamento em grandes áreas, dispõe-se de vários registradores ao longo dos pontos de acessos da tubulação (hidrantes ou válvulas) e por um determinado período eles analisam os ruídos ao longo da tubulação. Este método apenas determina a existência ou não de um vazamento, não sendo capaz de determinar sua localização.
- Correlacionador de sinais: composto por um ou mais microprocessadores e dois sensores (acelerômetro, hidrofone ou geofone). Utiliza-se da técnica de correlação cruzada, no qual são dispostos os dois sensores em uma seção da tubulação e através do microprocessador, e da função de correlação, pode-se determinar o ponto exato do vazamento. Este método possui uma grande precisão em relações aos demais, porém é necessário um treinamento adequado dos operadores, possui um custo elevado e a função de correlação depende de vários fatores, o que o torna complexo na determinação exata do ponto de vazamento.

É ilustrado na Figura 1 o esquemático de um correlacionador de sinais, sendo os hidrantes os pontos de acessos das tubulações, a distância do vazamento em relação ao sensor 1, segundo Fuchs e Riehle (1991), pode ser calculada por:

$$d_1 = \frac{d - cT_0}{2} \tag{1}$$

Sendo T_0 é o atraso de sinal e c é a velocidade, em *m/s*, de propagação da onda na tubulação.

Figura 1 - Esquemático de um correlacionador de sinais



Fonte: (GAO *et al.,* 2004)

Equipamentos não-acústicos:

- Técnica gás marcador: este método consiste em injetar um gás não tóxico, solúvel na água e mais leve que o ar (hélio ou hidrogênio). Ao injetar o gás em uma seção isolada da tubulação, o gás tende a sair através dos vazamentos ao longo da tubulação e então subir até a superfície. Ao chegar na superfície, através de equipamentos adequados, é feito a detecção do gás, assim encontrando a localização do vazamento.
- Termografia: este método baseia-se na utilização de uma câmera infravermelha capaz de detectar anomalias térmicas no solo. Em locais onde há vazamentos, o solo ao seu redor sofre alterações em suas características térmicas, assim sendo possível identifica-los através da câmara.
- Radar de penetração do solo: este método é capaz de localizar vazamentos através da detecção de vazios no solo criados pelos vazamentos, quando ocorre tais vazamentos a constante dielétrica do solo aumenta. As ondas do radar são refletidas ao encontrarem anomalias na constante dielétrica do solo, indicando a posição dos vazamentos.

A presente pesquisa está focada no uso de correlacionadores de sinais para a detecção de vazamentos em tubulações aterradas de plástico, visto que este método apresenta grande precisão ao determinar o ponto de vazamento. No caso de tubulações de plástico, este método se torna ainda mais complexo, pois existe um alto

amortecimento da onda no interior da tubulação e do meio externo, diminuindo a propagação da onda. Além disto a energia de vazamento em tubulações de plástico se encontra em baixas frequências, a presença de ressonâncias limita ainda mais estas frequências, sendo necessário a utilização de filtros passa-baixa para a realização das análises (ALMEIDA *et al*., 2014).

Outro fator relevante na análise de vazamento em tubulações de plástico é a incerteza em relação ao valor da velocidade de propagação da onda, sendo este valor essencial para a determinação da localização do vazamento (BRENNAN *et al*., 2005).

Este trabalho dedicou-se em grande parte no desenvolvimento do correlacionador de sinais ao invés da comunicação sem fio, como serão explicados no decorrer do texto e outro fato importante para a mudança do foco foi a escolha da placa a ser utilizada no projeto, a mesma já possui módulo Wifi integrada, não necessitando de adaptação para seu uso.

Este trabalho está organizado em Capítulos. No Capítulo 1 é mostrado uma sucinta introdução sobre a problemática do projeto. No Capítulo 2 é apresentado as principais características do SoC Zynq, sua arquitetura e funcionalidades, além dos componentes utilizados no projeto e discutido o desenvolvimento do Linux embarcado. O Capítulo 3 descreve os experimentos realizados em bancada além da análise dos resultados obtidos em cada experimento. No Capítulo 4 são apresentadas as conclusões do trabalho bem como os trabalhos futuros.

2 MATERIAL E MÉTODO

Nesse capítulo são apresentados alguns dos assuntos abordados afim de se compreender e desenvolver um protótipo do correlacionador de sinais utilizando o SoC Zynq.

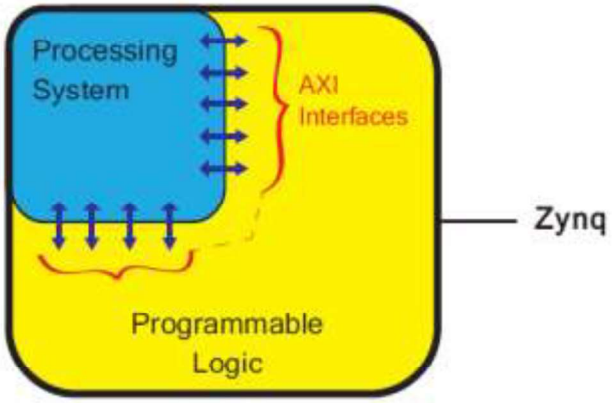
2.1 Arquitetura SoC Zynq

Zynq é um *System on Chip*, ou seja, um único circuito integrado é capaz de implementar um sistema inteiro. Sua principal característica é a combinação de um processador ARM Cortex-A9 (ARM) de um ou mais núcleos a um *Field Programmable Gate Array* (FPGA), esta combinação não é nova, porém no Zynq o processador ARM é capaz de executar sistemas operacionais completos, tal como o Linux embarcado. A união entre o sistema operacional e a arquitetura programável da FPGA evita problemas de interface entre dois dispositivos separados, reduz o tamanho físico e o custo total (CROCKETT *et al*., 2014).

O Zynq é dividido em duas partes principais: o *Processing System* (PS) e a *Programmable Logic* (PL). O PS é a área formada pelo processador ARM, onde é implementado rotinas de software e sistemas operacionais, enquanto a PL é a área formada pela FPGA, sendo capaz de implementar subsistemas de lógica, aritmética e operações que necessitem de fluxo de dados de alta velocidade (CROCKETT *et al*., 2014).

É apresentado na Figura 2 ,de forma simplificada, a arquitetura do Zynq, nela observa-se as partes que compõem o Zynq, PS e PL, e para a interconexão entre as partes há a *Advanced Extensible Interface* (AXI).

Figura 2 - Arquitetura simplificada do Zynq



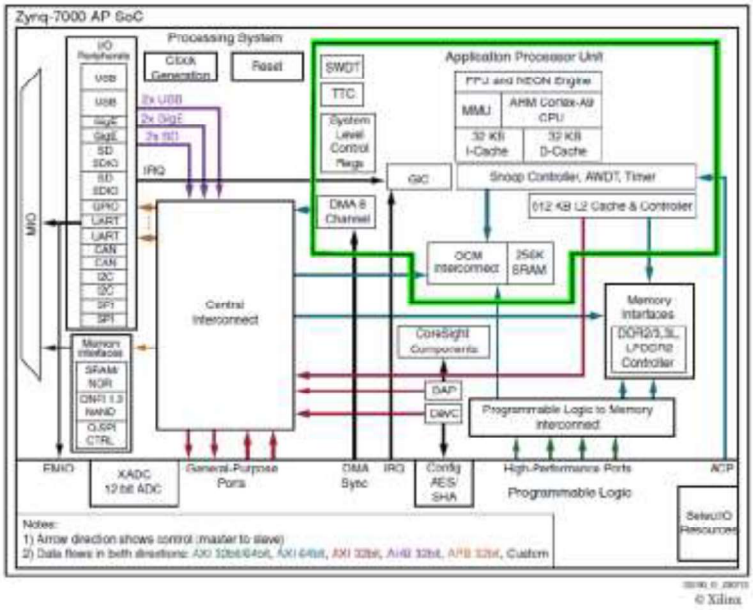
Fonte: (CROCKETT *et al*., 2014)

2.1.1 Processing System

O sistema de processamento do Zynq abrange, além do processador ARM, um conjunto de recursos de processamento no qual formam uma *Application Processor Unit* (APU), memória cache, interfaces de memória, interconexão e circuitos de geração de *clock*.

É ilustrado na Figura 3 o diagrama de blocos da arquitetura do PS do Zynq-7000, com os recursos principais presentes na Zynq, como APU, memória cache, gerador de *clock* e periféricos, alguns dos periféricos podem variar dependendo do modelo utilizado do Zynq.

Figura 3 - Diagrama de blocos da arquitetura do OS do Zynq-7000



Fonte: (CROCKETT *et al.*, 2014, modificada)

Os principais periféricos de entrada ou saída (E/S) são:

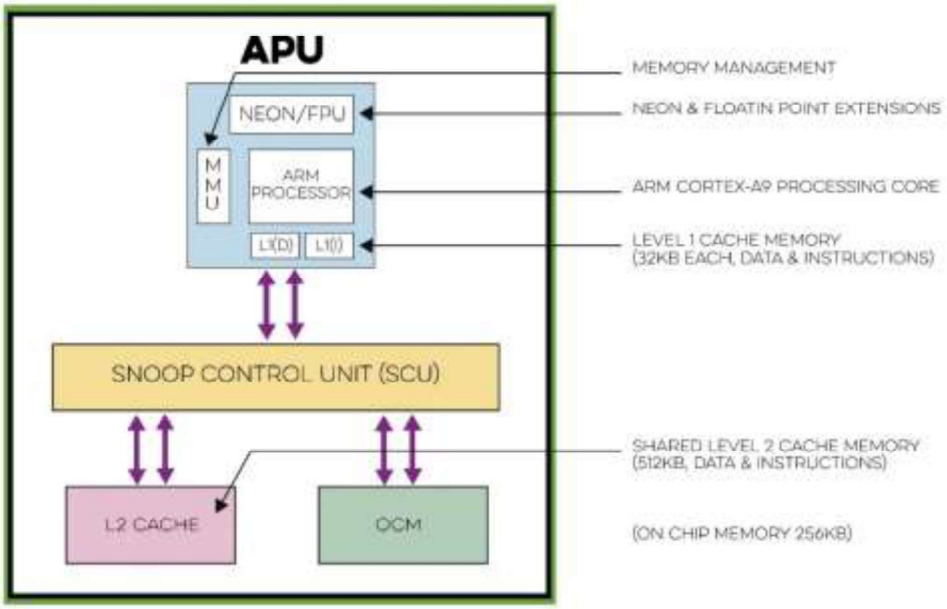
- *Serial Peripheral Interface* (SPI);
- *Inter-Integrated Circuit* (IIC);
- *Controller Area Network* (CAN);
- *Universal Asynchronous Receiver/Transmitter* (UART);
- *General Purpose Input/Output* (GPIO);
- *Secure Digital* (SD);
- *Universal Serial Bus* (USB);
- Ethernet.

2.1.1.1Unidade de Processamento

A APU normalmente possui dois núcleos de processamento ARM, sendo que ambos estão associados a unidades computacionais: um NEON™ *Media Processing Engine* (MPE) e *Floating Point Unit* (FPU); uma *Memory Management Unit* (MMU); e uma memória cache de nível 1. A APU também possui uma memória cache de nível 2 e uma *On Chip Memory* (OCM) e para interligá-los com os núcleos dos ARMs há a

Snoop Control Unit (SCU). É ilustrado na Figura 4 o diagrama de blocos de uma APU (CROCKETT *et al*., 2014).

Figura 4 - Diagrama de blocos da APU



Fonte: (CROCKETT *et al.*, 2014, modificada)

O processador ARM Cortex-A9 possui memórias caches separadas para dados e para instruções, permitindo o armazenamento local de dados e instruções com tempo de acesso rápidos e com ótimo desempenho do processador. A SCU tem a função de interligar os processadores ARMs com a memória de nível 2 e o OCM além de gerenciar as transações entre o PS e o PL através da *Accelerator Coherency Port* (ACP).

2.1.2 Programmable Logic

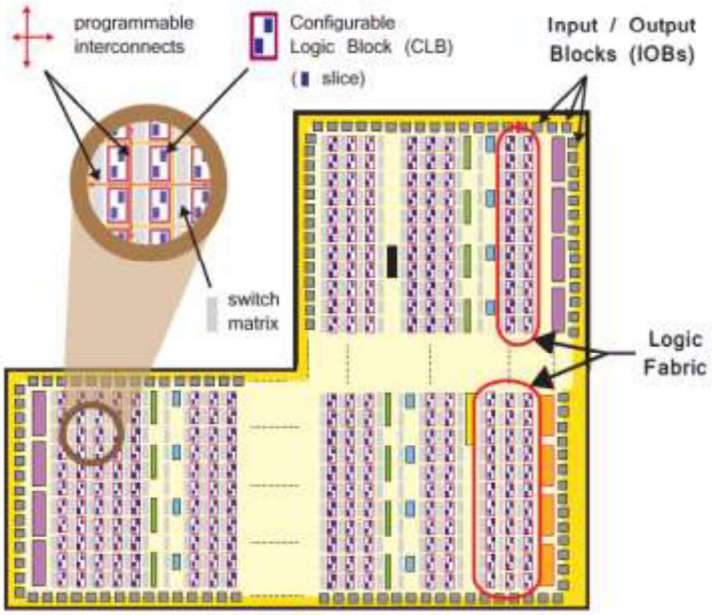
A parte da lógica programável no Zynq é baseado nas FPGA Artix®-7 e a Kintex®-7. O PL é composto por 7 partes essenciais para seu funcionamento:

- *Configurable Logic Block* (CLB);
- *Slices*;
- *Look-up table* (LUT);

- *Flip-flop* (FF);
- Matriz de comutação;
- *Input/Output Blocks* (IOBs).

A Figura 5 ilustra a representação do tecido lógico do PL e seus respectivos componentes.

Figura 5 - Tecido lógico do PL e seus componentes



Fonte: (CROCKETT et al., 2014)

Os LUTs são blocos nos quais são realizadas funções lógicas, sendo possível apenas guardar valor lógico, 0 ou 1. Além de ser capaz da implementação de funções lógicas, ele pode implementar uma pequena memória somente de leitura, uma pequena memória de acesso rápido ou um registro de mudanças.

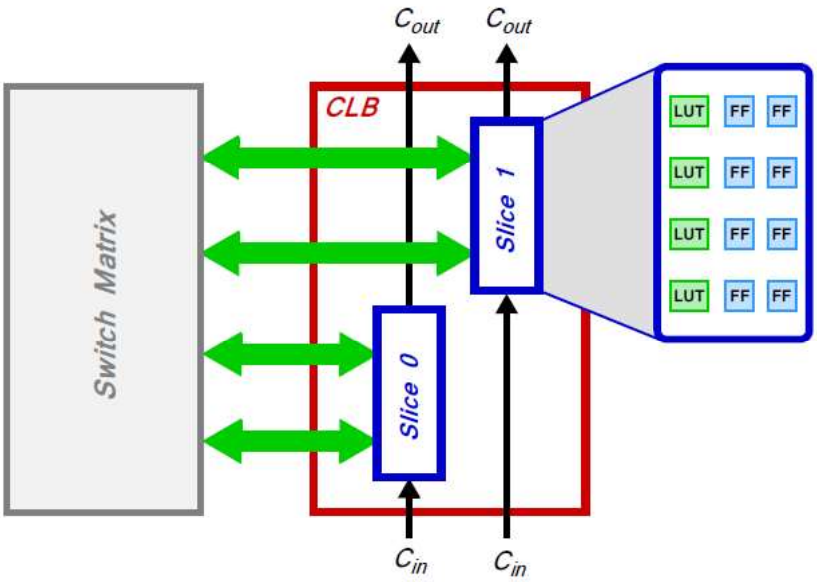
O *Flip-flop* é um elemento de armazenamento básico, sempre estando emparelhado com o LUT para fins de armazenamento de dados e auxiliar na lógica *pipeline*.

Slice é o termo referente a subunidade do CLB, podendo implementar circuitos lógicos combinacionais e sequenciais. Os *slices* no Zynq são compostos por 4 LUTs e 8 FFs.

A Matriz de Comutação é responsável por facilitar as conexões entre os CLBs ou entre o CLB e outro elemento do PL.

O CLB é o agrupamento de elementos lógicos e é conectado a outros recursos através de interconexões programáveis. Dentro de um CLB existe uma matriz de comutação e dois *slices*. A Figura 6 apresenta o diagrama de blocos do CLB e seus elementos.

Figura 6 - Diagrama de blocos do CLB



Fonte: (CROCKETT *et al.*, 2014)

Os Blocos de entrada/saída são recursos para realizar a interface entre os dispositivos externos e os recursos do PL.

2.2 – Minized

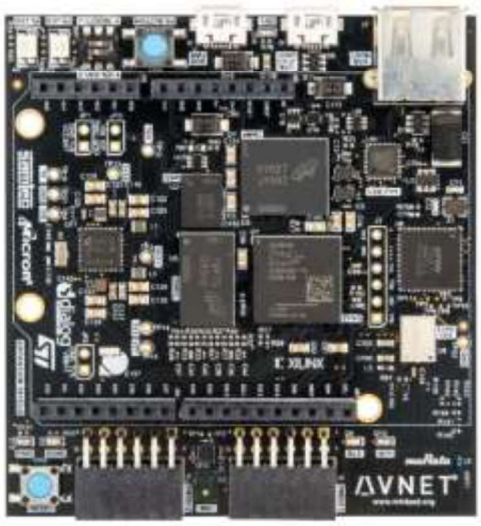
A placa Minized (AVNET, 2017) possui esse nome devido a placa de desenvolvimento chamada ZedBoard (Zynq *Evaluation & Development Board*), a Minized é uma versão de placa de desenvolvimento da empresa Avnet (AVNET, 2017) com tamanho reduzido, 71,12 mm x 76,2 mm, menor custo e utiliza o Zynq como sistema principal. Este modelo de placa foi selecionado para o projeto, pelo seu baixo custo, além da incorporação da placa Murata LBEE5KL1DX (MURATA, 2017), a qual realiza transmissões sem fio, tanto via *bluetooth* quanto via *Wi-fi*.

Outra característica importante é a presença de 2 portas *Peripheral Module* (Pmod) (DIGILENT, 2011), tais portas permitem a utilização de periféricos externos,

tais como sensores, acessando periféricos que funcionem pelas interfaces UART , SPI e I2C.

A Figura 7 apresenta a vista superior da Minized.

Figura 7 – Vista superior da placa de desenvolvimento Minized



Fonte: (AVNET, 2017)

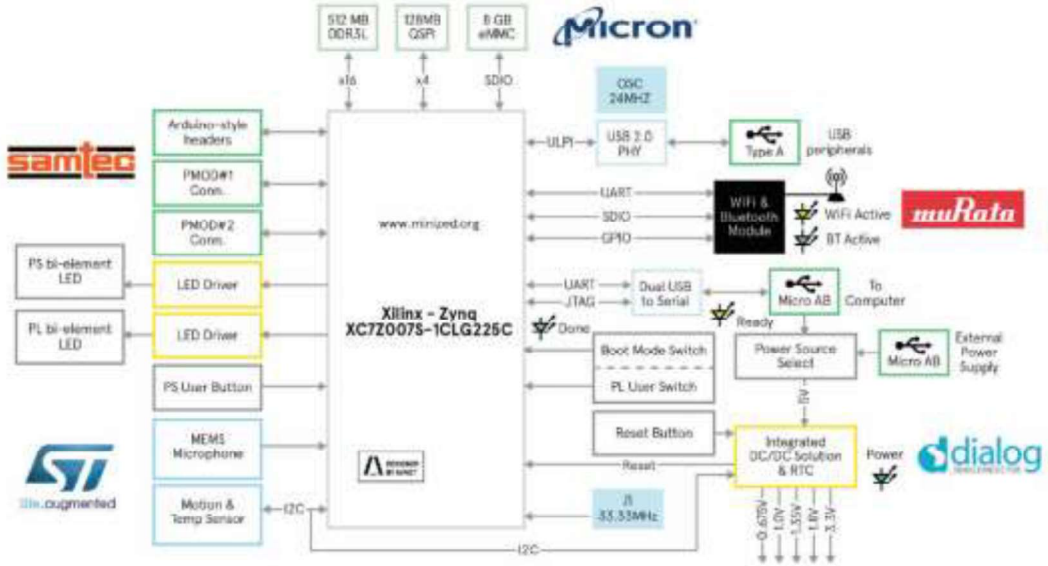
A seguir são listados os principais recursos da Minized:

- Dispositivo Zynq SoC de um núcleo: XC7Z007S-1CLG225C;
- Armazenamento:
  - Micron 512 MB DDR3L (*Double Date Rate Low-power*);
  - Micron 128 MB Quad SPI NOR *flash*;
  - Micron 8 GB eMMC (*Embedded Multi-Media Controller/Card*);
- Conector Micro USB para JTAG (*Join Test Action Group*) e UART via dispositivo FTDI (*Future Technology Devices Internacional*);
- Conector Micro USB para alimentação externa para o modo de alta potência;
- Conector hospedeiro USB tipo A para uma conexão de dispositivos USB escravo;
- 2 soquetes Pmod;
- Módulo Murata sem fio tipo: LBEE5KL1DX:
  - Baseado no Cypress BCM4343W;
  - *Wi-fi* 2,4 GHz está em conformidade com 802.11b/g/n;
  - *Bluetooth*:

        - *Bluetooth* versão 4.1 mais EDR (*Enhanced Data Rate*);
        - Classe de potência 1 (10dBm max) e BLE (*Bluetooth Low Energy*);
    - Seguindo a certificação do *design* de referência, a antena é integrada na placa;
- Dialogo PMIC (*power-management IC*) com interface I2C: DA9062;
- 2 LEDs (*Light Emitting Diode*) de usuário de dois elementos;
- LED de cor única para energia, configuração, *Wi-fi* e *Bluetooth*;
- Botão de reinicialização e botão de acionamento do usuário;
- Interruptor de usuário para o PL;
- Acelerômetro ST Micro e sensor de temperatura: LIS2DS12;
- Sensor de microfone ST Micro MEMS (*Microelectromechanical system*): MP34DT05.

É exemplificado na Figura 8, em forma de diagrama de blocos, os componentes e conexões.

Figura 8 - Diagrama de blocos da placa Minized



Fonte: (AVNET, 2017)

2.3 Pmod

A interface Pmod da Digilent é utilizada para conectar periféricos de baixas frequências e com baixo número de pinos de E/S. A interface Pmod está disponível em dois formatos, um contendo 6 pinos e outro com 12 pinos. A versão com 6 pinos possui 4 pinos digitais de E/S, 1 pino de alimentação e um pino para o terra, enquanto que a versão com 12 pinos possui 8 pinos digitais de E/S, 2 pinos de alimentação e 2 pinos para o terra (DIGILENT, 2011).

A interface Pmod proporciona 4 tipos de conexão entre os módulos periféricos e a placa que utiliza a interface Pmod, sendo eles:

- Interface I2C;
- GPIO;
- SPI;
- UART;

2.3.1 Interface I2C

A interface I2C consiste na utilização do barramento I2C existente na placa, esse barramento requer apenas duas vias de transmissão, uma para o envio da *serial data* (SDA) e outra para o envio do *serial clock* (SCL). Cada dispositivo conectado ao barramento receberá um endereço de memória único e será definido qual dispositivo será mestre e qual será escravo (NXP SEMICONDUCTORS, 2014).

O barramento I2C funciona de forma *serial*, utiliza 8 bits ordenados, possui transferência de dados unidirecional ou bidirecional dependendo da velocidade de transmissão necessária para a aplicação.

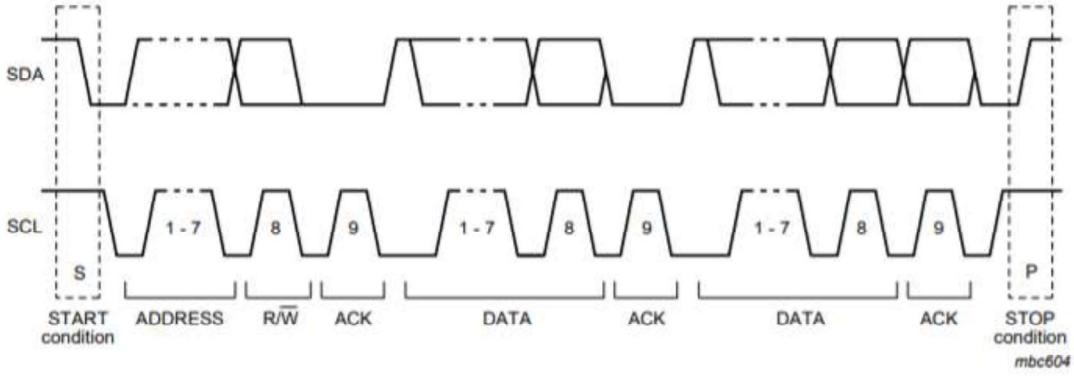
O barramento opera seguindo as definições do protocolo I2C, que é um conjunto de regras que definem a transmissão dos sinais de *clock* e dados *serial* dentro do barramento I2C. A seguir é apresentado de forma simplificada o funcionamento do protocolo I2C.

- Define os dispositivos mestres ou escravos;
- Verificação do nível logico do SDA e do SCL;

- Validação dos dados a serem enviados;
- Verificação das condições de início e término da transmissão;
- Envio dos dados, sendo transmitidos por linha da dados apenas de 8 bits por vez, sendo o oitavo bit responsável por definir se o dispositivo é mestre ou escravo;
- Adiciona um bit de *Acknowledge* ou *Not Acknowledge* ao final de cada pacote de 8 bits transmitidos, sendo esse bit responsável por verificar o recebimento ou não dos dados.
- Continuar a transmissão até ser verificado a condição de término.

É ilustrado na Figura 9 o processo de transmissão de dados baseado no protocolo I2C.

Figura 9 - Transmissão de dados no barramento I2C



Fonte: (NXP SEMICONDUCTORS, 2014)

Para a utilização da interface Pmod através do barramento I2C é necessário adicionar aos pinos de SDA e SCL resistores *pull up* para elevar o nível lógico baixo para o nível lógico alto, algumas placas de desenvolvimento já possuem esses resistores internamente, porém na placa Minized, segundo o manual, é necessário adicionar resistores *pull up* de 4,7 kΩ (AVNET, 2017).

A Tabela 2 apresenta a pinagem do Pmod e seus respectivos sinais.

Tabela 2 - Pmod interface I2C

Número do pino	Sinal	Número do pino	Sinal
1	SCL	2	SCL
3	DAS	4	DAS
5	GND	6	GND
7	VCC	8	VCC

Fonte: (DIGILENT, 2011, traduzido)

2.3.1 GPIO

A comunicação utilizando a GPIO é bastante simples, visto que, a interface Pmod funciona como pinos E/S, sendo assim, o modo como os sinais são utilizados dependerá apenas do usuário que irá programar a placa.

A Tabela 3 apresenta a pinagem do Pmod e a direção de operação do pino.

Tabela 3 - Pmod interface GPIO

Número do pino	Sinal	Direção
1	IO1	E/S
2	IO2	E/S
3	IO3	E/S
4	IO4	E/S
5	GND	
6	VCC	

Fonte: (DIGILENT, 2011, tradução minha)

2.3.2 SPI

A comunicação SPI baseia-se no protocolo SPI, ele é utilizado para a comunicação entre um microcontrolador e um ou mais periféricos de forma *serial*

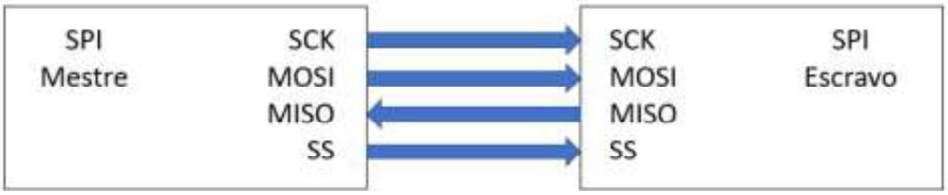
síncrona. Assim como no protocolo I2C, o SPI também utiliza a metodologia mestre e escravo, porém só pode haver um mestre por conjunto (LI-LI *et al.*, 2014).

O protocolo opera em modo *full duplex*, ou seja, transmissão simultânea nas duas direções. São necessárias pelo menos 4 vias de transmissão, sendo uma para o envio de dados do escravo para o mestre, *Master Input Slave Output* (MISO), uma para o envido de dados do mestre para o escravo, *Master Output Slaver Input* (MOSI), uma para o *serial clock* (SCK) e outra para selecionar para qual dispositivo escravo deve ser transmitido os dados, *Slave Select* (SS) (LI-LI *et al.*, 2014).

Diferente do protocolo I2C o protocolo SPI não é limitado a transferência de dados via 8 bits, além disso não é necessário adicionar resistores *pull up* para seu correto funcionamento. Em contrapartida esse protocolo não possui o bit de *Acknowledge*, fazendo deste protocolo inseguro, pois não há certeza do recebimento dos dados entre os dispositivos.

É apresentado na Figura 10 o diagrama de blocos das conexões entre dispositivos SPI.

Figura 10 - Comunicação SPI



Fonte: autoria própria.

A Tabela exibe a pinagem do Pmod, seus respectivos sinais e direções.

Tabela 4 - Pmod interface SPI

Número do pino	Sinal	Direção
1	SS	Saída
2	MOSI	Saída
3	MISO	Entrada
4	SCK	Saída
5	GND	
6	VCC	

Fonte: (DIGILENT, 2011, tradução minha)

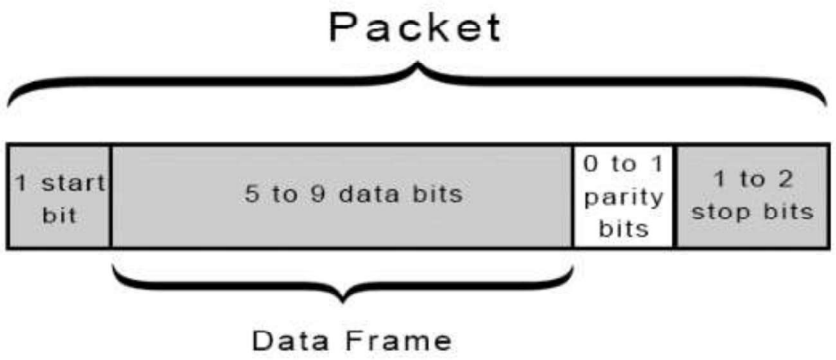
2.3.3 UART

O UART, diferente do SPI e do I2C que utilizam protocolos para realizar a comunicação entre dispositivos, é um circuito físico no qual o microcontrolador ou circuito integrado possui em seu dispositivo tendo a função de enviar e receber dados seriais.

A transmissão utilizando o UART é feita entre dois dispositivos, convertendo dados paralelos em dados seriais, além de não necessitar de um *clock*, pois ela é uma transmissão assíncrona. Para realizar a transmissão são necessárias apenas duas vias de transmissão, uma para a transmissão de dados T_X e uma para o recebimento de dados R_X (CIRCUIT BASICS, 2016).

Esta transmissão pode conter de 5 a 9 bits de dados, no caso de 9 bits de dados não será possível utilizar um bit para verificação de recebimento de dados. É ilustrado na Figura 11 o funcionamento da transmissão de dados utilizando dispositivos UART.

Figura 11 - Transferência de dados UART



Fonte: (CIRCUIT BASICS, 2016)

Na Figura 11 observa-se que para realizar a transmissão entre dispositivos UART é necessário um bit para determinar o início da transmissão e um ou dois bits para determinar o término da transmissão. Esse tipo de transmissão possui a vantagem de não necessitar de um *clock* e utilizar apenas duas vias de comunicação, porém só pode ser realizada entre dois dispositivos.

A Tabela 5 apresenta a pinagem do Pmod, seus respectivos sinais e direções.

Tabela 5 - Pmod interface UART

Número do pino	Sinal	Direção
1	CTS	Saída
2	RTS	Entrada
3	T_X	Entrada
4	R_X	Saída
5	GND	
6	VCC	

Fonte: (DIGILENT, 2011, tradução minha)

Sendo:

- CTS um sinal de permissão de transferência de dados;
- RTS um sinal de requerimento de transferência de dados.

2.4 Módulos Pmod

Neste projeto utilizou-se de 4 módulos distintos de Pmods, sendo eles:

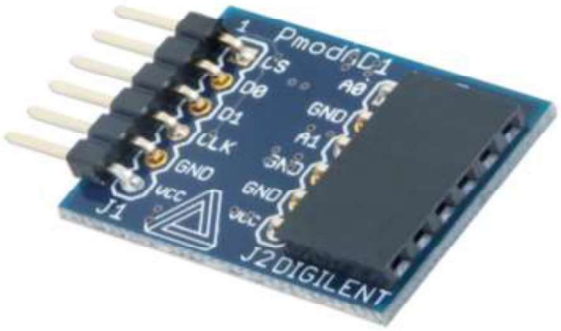
- Pmod AD1;
- Pmod RTCC (*Real Time Clock/Calendar)*;
- Pmod GPS (*Global Position System*);
- Pmod NAV.

2.4.1 Pmod AD1

O módulo Pmod AD1 é um conversor analógico para digital de 12 bits com 2 canais, o conversor utilizado nesse módulo é o modelo AD7476A da *Analog Devices* (ANALOG DEVICES, 2017). Possui uma taxa de amostragem de até um milhão de amostras por segundo, além de possuir um tamanho reduzido (DIGILENT, 2016a).

Na Figura 12 é apresentado o Pmod AD1, observa-se que este módulo utiliza apenas 6 pinos Pmod para seu funcionamento.

Figura 12 - Módulo Pmod AD1

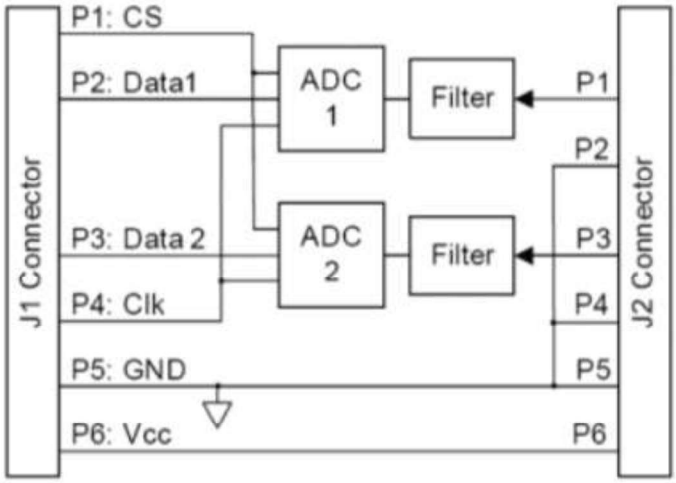


Fonte: (DIGILENT, 2016a)

Este módulo emprega a interface GPIO para a comunicação de dados, porém as portas GPIOs são adaptadas para funcionarem de forma semelhante a interface SPI. Como este módulo funciona simultaneamente com dois canais de transmissão de dados, ao invés de utilizar uma porta MOSI e um MISO, ambas as portas são do tipo MISO, sendo assim não ocorre envio de dados do mestre para o escravo, somente ocorrerá transmissão de dados do escravo para o mestre (DIGILENT, 2016a).

É ilustrado na Figura 13 o circuito interno do Pmod AD1, com os nomes e pinos nas Tabelas 6 e 7.

Figura 13 - Circuito interno do módulo Pmod AD1



Fonte: (DIGILENT, 2016a)

Tabela 6 - Pmod AD1 conector J1

Conector J1		
Pino	Sinal	Descrição
1	CS	*Chip Select*
2	D0	Entrada de dados 1
3	D1	Entrada de dados 2
4	SCK	*Serial Clock*
5	GND	Terra
6	VCC	Fonte de tensão

Fonte: (DIGILENT, 2016a, tradução minha)

Tabela 7 - Pmod AD1 conector J2

Conector J2		
Pino	Sinal	Descrição
1	A0	Entrada de dados 1
2	GND	Terra
3	A1	Entrada de dados 2
4	GND	Terra
5	GND	Terra
6	VCC	Fonte de tensão

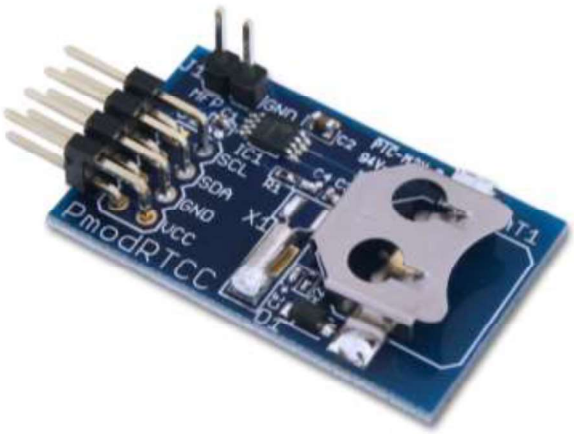
Fonte: (DIGILENT, 2016a, tradução minha)

2.4.2 Pmod RTCC

O módulo Pmod RTCC é um relógio em tempo real externo, utiliza o *chip* MCP79410 da Microchip® (MICROCHIP, 2018), o módulo usa a interface I2C para transmissão de dados, possui 2 alarmes disponível para o usuário, além de uma EEPROM interna de 128 *bytes* (DIGILENT, 2016c).

A Figura 14 ilustra o módulo Pmod RTCC, observa-se a presença de um soquete para bateria de litium tipo moeda (lado direito da figura), esta bateria é necessária para que o Pmod RTCC permaneça operando mesmo com o dispositivo principal desligado.

Figura 14 - Módulo Pmod RTCC



Fonte: (DIGILENT, 2016c)

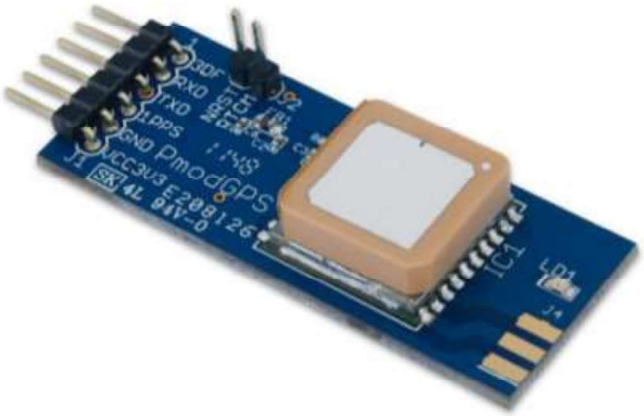
Na Figura 14 além dos pinos padrões para a utilização da interface I2C, o módulo possui 2 pinos auxiliares, sendo um pino de sinal e outro para o terra do circuito, o pino de sinal (J1) é capaz de gerar uma onda quadrada, o que possibilita verificar quando o alarme, previamente ajustado, estará em nível lógico alto, facilitando a análise do sistema. Para a correta utilização deste pino é necessário adicionar um resistor *pull up* entre o pino J1 e uma fonte de tensão (DIGILENT, 2016c).

2.4.3 Pmod GPS

O módulo Pmod GPS permite a localização por satélite com precisão para qualquer tipo de sistema embarcado. Ele possui precisão de 3 m quando operando com localização 2D, possui baixo consumo de energia e taxa de atualização de até 10 Hz (DIGILENT, 2016b). O modelo de GPS utilizado é o FGPMMOPA6H da GlobalTop (GLOBALTOP TECHNOLOGY INC., 2012).

A Figura 15 mostra o módulo Pmod GPS.

Figura 15 - Módulo Pmod GPS



Fonte: (DIGILENT, 2016b)

A comunicação utilizada pelo Pmod GPS é a do tipo UART com mudanças apenas na ordem dos pinos Pmod, a Tabela 8 apresenta os pinos e suas respectivas descrições.

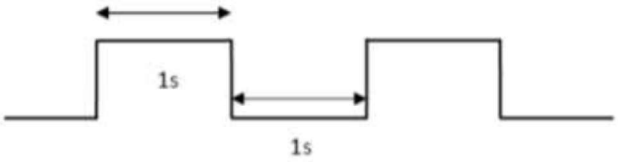
Tabela 8 - Pinos Pmod GPS

Pino	Sinal	Descrição
1	3DF	Indicador fixo 3D
2	RX	Receptor
3	TX	Transmissor
4	1PPS	Pulso por segundo
5	GND	Terra
6	VCC	Fonte de tensão

Fonte: (DIGILENT, 2016b, tradução minha)

Pela Figura 15 e pela Tabela 8 observa-se a presença dos sinais 3DF e 1PPS. O sinal 3DF irá indicar o *status* do GPS, ou seja, ele irá informar se o GPS tem satélites suficientes para o correto funcionamento do sistema. Para indicar que o GPS possui sinais de satélites suficientes o pino 3DF estará em nível lógico baixo, caso contrário será gerado uma onda quadrada na qual trocará seu nível lógico a cada segundo, conforme ilustrado na Figura 16.

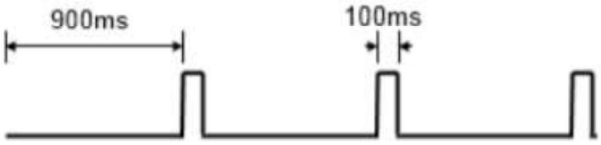
Figura 16 - Sinal 3DF quando não possui sinal de GPS



Fonte: (DIGILENT, 2016b)

O pino 1PPS irá gerar um sinal de onda, sincronizado com o sinal de GPS, do tipo pulso por segundo, ou seja, a cada segundo o nível lógico do sinal será alto e permanecerá neste estado por 100 ms, após esse período irá para nível lógico baixo permanecendo por 900 ms neste estado, conforme a Figura 17.

Figura 17 - Sinal do pino 1PPS



Fonte: (DIGILENT, 2016b)

2.4.4 Pmod NAV

O Pmod NAV (DIGILENT, 2017) consiste de um conjunto de quatro sensores, tal conjunto é dividido em duas partes, a primeira possui um acelerômetro, giroscópio e magnetômetro de 3 eixos, sendo encapsulados no *chip* de modelo LSM9DS1 (ST LIFE.AUGMENTED, 2015) e a segunda parte possui um barômetro digital do modelo LPS25HB (ST LIFE.AUGMENTED, 2016). Na Figura 18 é ilustrado o módulo Pmod NAV.

Figura 18 - Módulo Pmod NAV

Fonte: (DIGILENT, 2017)

O módulo utiliza os 12 pinos Pmod para seu funcionamento e comunica-se pela interface SPI, sendo esta interface expandida para a utilização com os 12 pinos. A Tabela 9 apresenta a pinagem e suas respectivas descrições.

Tabela 9 - Pinos Pmod NAV

Pino	Sinal	Descrição
1	CS_A/G	*Chip select* para o acelerômetro/giroscópio
2	SDI	MOSI
3	SDO	MISO
4	SPC	*Serial Clock*
5	GND	Terra
6	VCC	Fonte de tensão
7	INT	Pino de interrupção para todos os componentes
8	DRDY_M	Dados pronto para o magnetômetro
9	CS_M	*Chip select* para o magnetômetro
10	CS_ALT	*Chip select* para o altímetro
11	GND	Terra
12	VCC	Fonte de Tensão

Fonte: (DIGILENT, 2017, tradução minha)

Neste projeto o Pmod NAV será utilizado apenas como acelerômetro, para medir o as vibrações das tubulações enterradas. A Tabela 10 apresenta a faixa de medição de aceleração linear e a correspondente sensibilidade de aceleração linear (ST LIFE.AUGMENTED, 2015).

Tabela 10 - Parâmetros do acelerômetro

Faixa de medição	Sensibilidade
±2 g	0,061 mg
±4 g	0,122 mg
±6 g	0,244 mg
±8 g	0,732 mg

Fonte: Adaptado (ST LIFE.AUGMENTED, 2015)

Ao utilizar o Pmod NAV no modo acelerômetro, pode-se escolher a frequência de amostragem a ser utilizada, o LSM9DS1 possui 6 frequências de amostragens, sendo: 14,9 Hz, 59,5 Hz, 119 Hz, 238 Hz, 476 Hz e 952 Hz (ST LIFE.AUGMENTED, 2015).

2.5 Xilinx Vivado Design Suite

No desenvolvimento deste projeto utilizou-se o software Xilinx Vivado Design Suite, versão 2018.1, com licença *WebPACK*®, sendo está a licença gratuita do *software*. Nesta versão está incluso o *software* principal para criação de projetos no SoC Zynq, sendo eles, Vivado *Integrated Development Enviroment* (IDE), responsável pela modelagem do hardware; *Software Development Kit* (SDK) e GNU *Compiler Connection* (GCC), estes responsáveis pela programação do *software*; e o Vivado *Simulator* responsável para verificação dos resultados (CROCKETT *et al*., 2014).

O Vivado IDE é um ambiente de desenvolvimento responsável pela modelagem do PL SoC, como processador, memória, periféricos, interfaces externas, entre outros. Além disso o Vivado IDE possui a facilidade de integrar os *Intellectual Property* (IP) *blocks* no desenvolvimento do projeto (CROCKETT *et al*., 2014).

Os IPs são blocos criados a partir da linguagem VHDL, os quais já estão previamente prontos para serem utilizados, devendo apenas realizar algumas

mudanças para cada tipo de projeto. Existem 2 tipos de IP *blocks*, o primeiro é o *soft* IP *block*, este IP permite que o usuário realize mudanças em suas configurações. E também há o *hard* IP *block*, o qual não permite modificações em suas configurações.

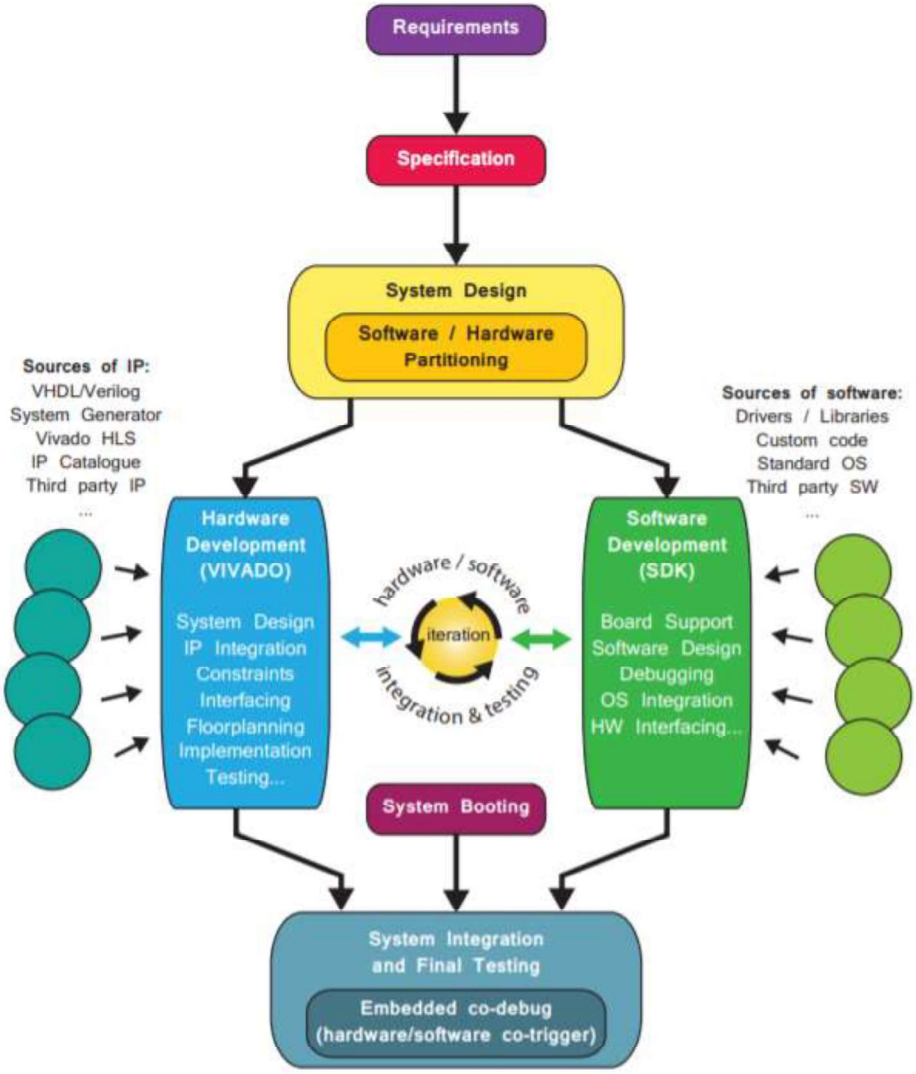
Dentro dos *soft* IP *blocks* tem-se os IPs desenvolvidos pela própria Xilinx e por outras empresas, denominadas como *third-party* IP. Neste projeto serão utilizados os *third-party* IPs da Digilent, pois a própria Digilent desenvolve IPs de seus respectivos módulos Pmods, afim de aumentar a abrangência de seus dispositivos.

O SDK é um software baseado na plataforma *Eclipse*, no qual possui suporte para todos os Xilinx IP *blocks*, bibliotecas desenvolvidas para o ARM e extensões NEON. A programação realizada no SDK pode ser feita utilizando linguagem C ou C++, facilitando o desenvolvimento dos projetos, pois são linguagens amplamente difundidas.

Para a verificação do *hardware* e do *software* desenvolvidos pelo Vivado IDE e o SDK, respectivamente, utiliza-se o Vivado *Simulator*, este é um ambiente de teste capaz de testar os componentes (*hardware*) com o sistema (*software*).

É apresentado na Figura 19 o diagrama de blocos, simplificado, para a realização da modelagem do SoC Zynq.

Figura 19 - Diagrama de blocos da modelagem do SoC Zynq



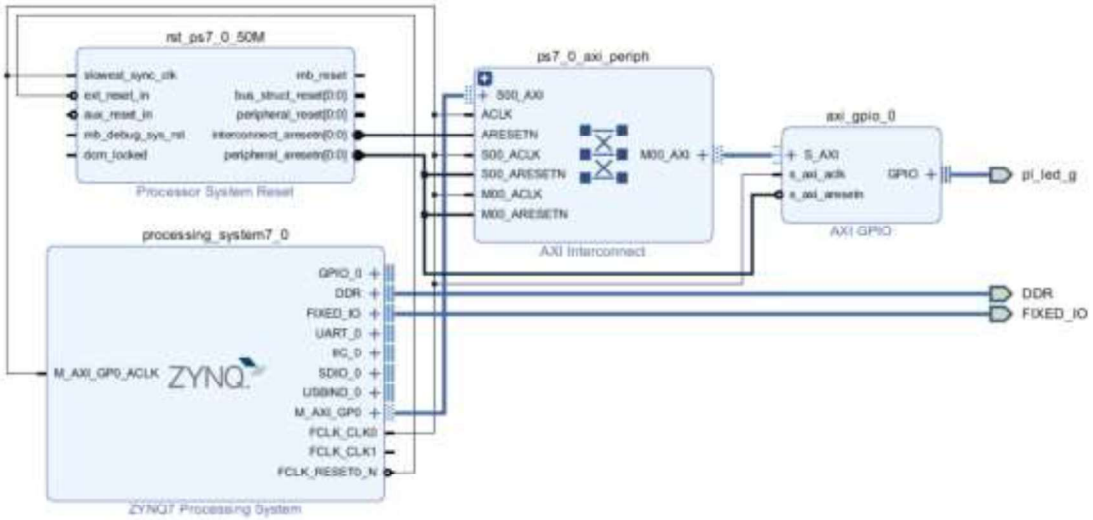
Fonte: (CROCKETT *et al.*, 2014)

Pela Figura 19 podemos verificar duas frentes de desenvolvimento, a parte do *hardware* e a parte do *software*, ambas podem ser desenvolvidas separadamente, mas realizando sua integração ao final do processo.

Para o desenvolvimento do *hardware* inicia-se o Vivado IDE e em seguida é necessário a modelagem do *block design*, este sendo um conjunto de IP *blocks* responsável pela configuração do *hardware*, na Figura 20 tem-se um exemplo de *block design*, nela há 4 IP *blocks*, o ZYNQ 7 *processing system*, responsável pela configuração do PS afim de fazer sua interconexão com o PL, também é possível configurar relógio, interrupções, periféricos entre outros através deste IP; há o *Processor System Reset*, responsável por interligar os relógios do PS com os relógios dos demais IP *blocks* conectados; o AXI *Interconnect* é o bloco utilizado para realizar as interconexões entre as partes do PS e do PL; e por fim há o AXI GPIO, este é um

IP *block* de utilidade geral, podendo acessar a pinagem do SoC Zynq e utilização pelo usuário conforme as suas necessidades.

Figura 20 - Exemplo de block design

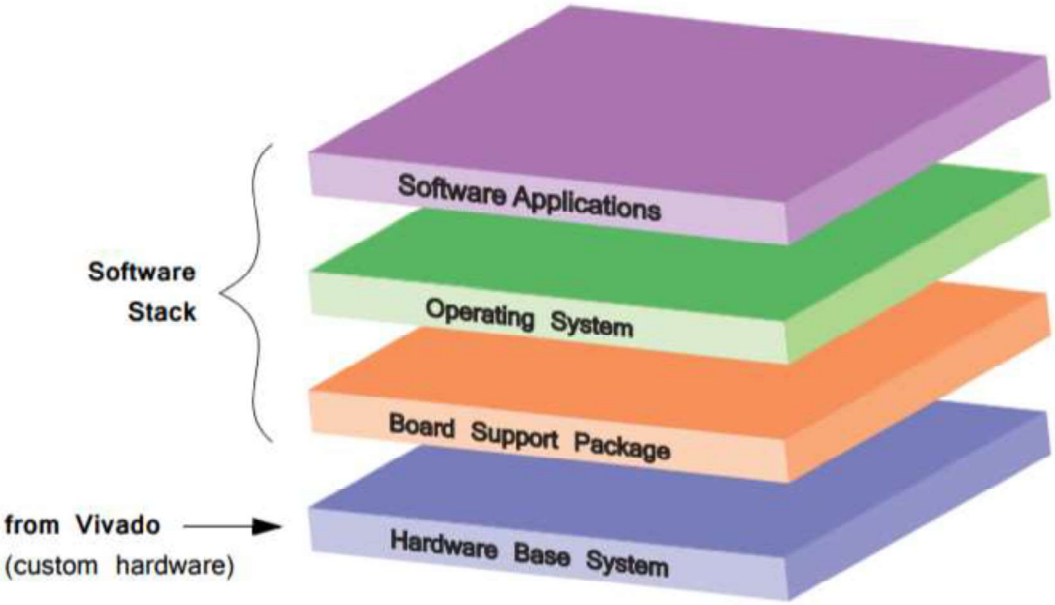


Fonte: autoria própria

Ao finalizar a modelagem do *block design* é necessário gerar um arquivo *bitstream*, este arquivo irá conter todos os dados referente a modelagem do *hardware* e poderá ser utilizado na programação do *software.* Com o arquivo *bitstream* gerado, exporta-se o projeto juntamente com o *bitstream* para o SDK, iniciando-se a programação no *software* do SoC Zynq.

Na parte do *software* do SoC Zynq há um conjunto de camadas responsáveis por sua configuração, na Figura 21 é ilustrado as camadas que o SoC Zynq apresenta afim de realizar a programação do *software* e interligá-lo ao *hardware*.

Figura 21 - Sistema de camadas do SoC Zynq



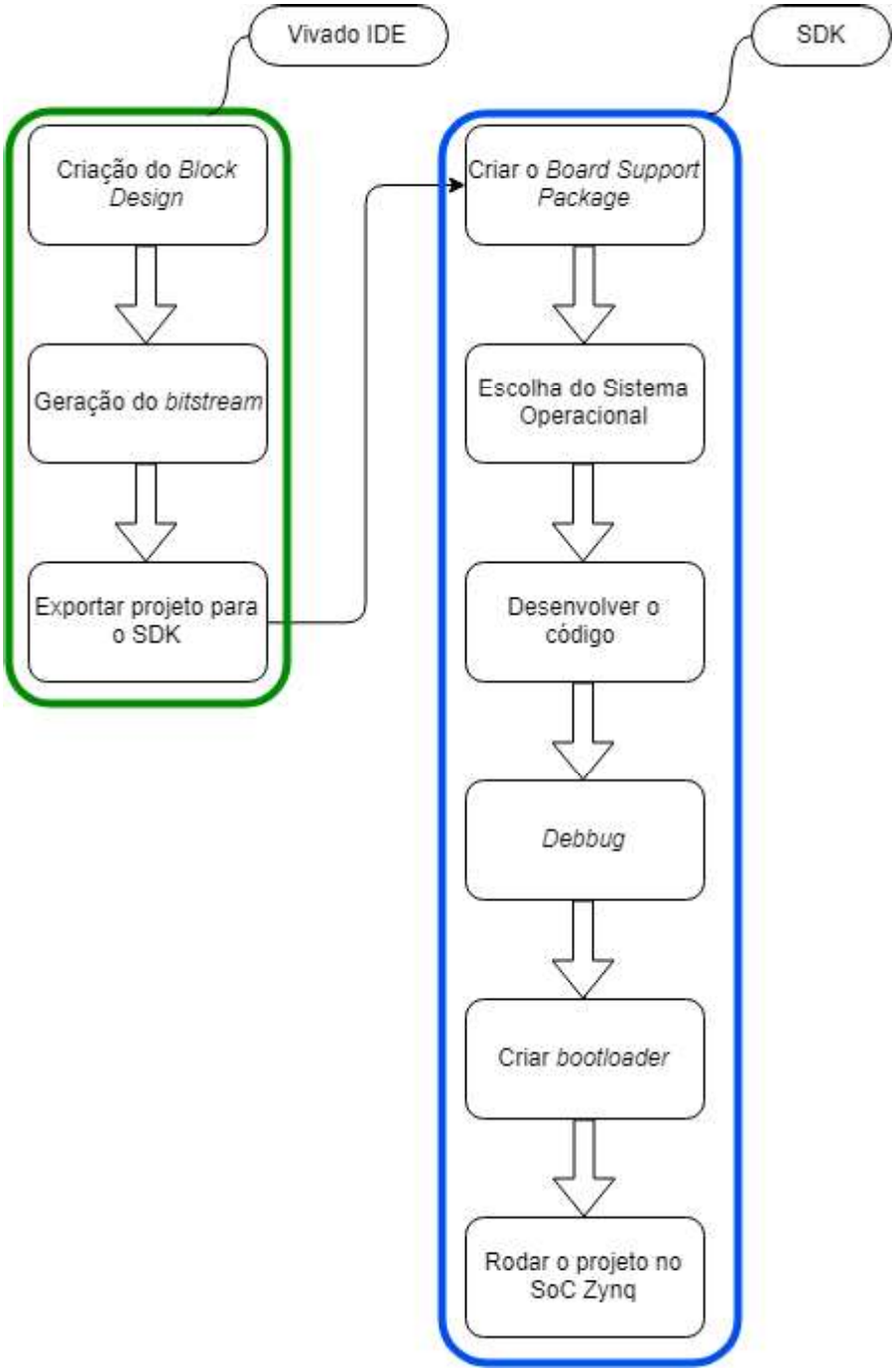
Fonte: (CROCKETT *et al.*, 2014)

Pela Figura 21 observa-se 3 camadas referente ao *software*, primeiro a *Board Support Package* (BSP) que configura *drivers* de baixo nível e funções que são necessárias para a comunicação entre os níveis acima e o *hardware*; a camada seguinte é o Sistema Operacional (SO), no caso do SoC Zynq há o Linux embarcado e o *Real-time Operation System* (RTOS), além do *baremetal* quando não há o SO; e na última camada tem-se o *Software Applications*, onde são criados os códigos em C/C++ responsáveis pela programação do *software*.

Para a criação do *software* utiliza-se o SDK, nele é necessário criar o BSP, afim de disponibilizar as ferramentas necessárias para a compilação do código. Em seguida gera-se um *Application Project*, seleciona-se o SO que será utilizado e ao mesmo tempo cria-se um ambiente para a programação em C/C++. Ao finalizar o código pode-se utilizar a ferramenta *debbug* para descobrir possíveis erros no programa. Com o código finalizado e devidamente corrigido gera-se um arquivo *bootloader*, tal arquivo é utilizado para inicializar o programa através do SO escolhido e por fim pode-se testar o programa na placa.

É apresentado na Figura 22, através de diagrama de blocos, os passos para a geração de um projeto em SoC Zynq.

Figura 22 - Diagrama de blocos referente ao desenvolvimento do projeto em SoC Zynq



Fonte: autoria própria

2.6 Linux Embarcado

Sistemas embarcados são sistemas eletrônicos microprocessados nos quais são projetados para realizar uma função específica. Ele é capaz de gerenciar recursos

complexos do sistema para assim concluir o gerenciamento do processo, da memória, dos dispositivos e interrupções de tarefas, também fornece bibliotecas, *drivers*, ferramentas e aplicações ao usuário (GUO, WANG e WANG, 2010).

Linux embarcado é um dos diversos tipos de sistemas embarcados que existem atualmente. Existem inúmeras vantagens para a utilização do Linux embarcado, de acordo com (BENAYAS, 2013), as razões mais importantes são:

- Apoio comunitário e possibilidade da participação do mesmo;
- Cobertura de dispositivos;
- Facilidade no teste de novos recursos;
- Controle total;
- Baixo Custo;
- Reutilização da plataforma;
- Qualidade;
- Possibilidade de RTOS (sistema operacional em tempo real).

O Linux é um sistema OpenSource, portanto muitas pessoas utilizam o sistema, gerando uma grande comunidade de usuários e desenvolvedores e tornando o sistema cada vez melhor. Por ser um sistema OpenSource, o Linux não possui custo, o desenvolvedor pode ter total acesso ao código do sistema e fazer as mudanças que desejar (BENAYAS, 2013).

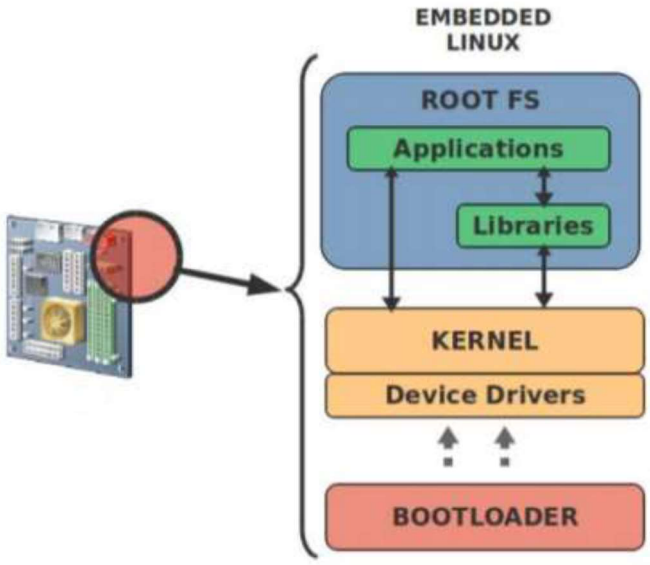
2.6.1 Elementos do Linux Embarcado

Segundo Sharma et al. (2011), a arquitetura básica do Linux embarcado é formada por:

- Bootloader;
- Kernel Linux;
- Rootfs;

É apresentado na Figura 23 os componentes básicos do sistema Linux embarcado.

Figura 23 - Componentes básicos do sistema Linux embarcado



Fonte: (SHARMA *et al.*, 2011, modificada)

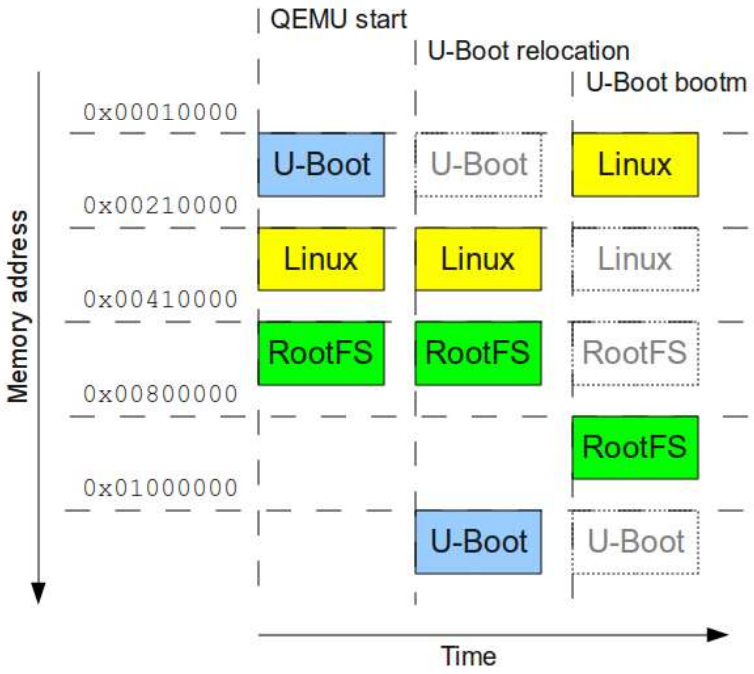
2.6.2 Bootloader

O Bootloader normalmente é o primeiro software a rodar após inicializar a placa, suas principais funções são (SHARMA *et al*., 2011):

- Inicializar o hardware antes de executar o Kernel;
- Passar parâmetros para o Kernel;
- Alocação de memória do Kernel;
- Rotinas de diagnóstico de hardware.

É apresentado na Figura 24 o processo de inicialização de um sistema de Linux embarcado. Na Figura 24 o Bootloader utilizado foi o U-Boot, um dos vários tipos de Bootloaders existentes.

Figura 24 - Processo de boot de um sistema Linux embarcado



Fonte: (MACIEL, 2014)

2.6.3 Kernel Linux

O Kernel é o componente principal e o núcleo do sistema, sendo este o código base e central que mantém o sistema funcionando. O Kernel gerencia alocação de memória, periféricos internos e externos à SoC, ciclos de processador, sendo assim um intermediário entre o hardware e as aplicações desenvolvidas pelo usuário (TAURION, 2005; SHARMA *et al.,* 2011). Na fase de inicialização do Linux embarcado também é responsável por Trivedi, Patel e Chauhan (2016):

- Gerenciamento de processos;
- Gerenciamento da memória;
- Controle de dispositivos;
- Iniciar o escalonador de tarefas;
- Gerenciamento de redes;
- Montar o sistema de arquivos principais (Rootfs) e chamar o processo de inicialização.

2.6.4 Rottfs

O Rootfs (*Root file system*) é uma maneira de representar diretórios de forma hierárquica, sendo sua estrutura em formato de árvore, onde os arquivos seriam as folhas e os diretórios os nós internos. Sendo responsável por prover as bibliotecas e programas baseado em sua hierarquia predefinida (SHARMA *et al.,* 2011).

2.7 Projeto do sistema Linux embarcado

Neste trabalho foi necessário a utilização do sistema Linux embarcado, pois o modulo Wi-fi Murata somente funciona no sistema Linux. O Linux embarcado utilizado no Minized é o Petalinux, esta versão é utilizada em todas as placas que utilizam o SoC Zynq.

Para o desenvolvimento de um sistema Linux embarcado funcional e com o módulo Murata em funcionamento, foi utilizado como base o *Reference Guide* do Petalinux (XILINX, 2018) e o curso da Avnet *Integrating Sensors on Minized with PetaLinux* (AVNET, 2017).

Inicialmente é necessário instalar o PetaLinux em um sistema operacional Linux, neste projeto foi utilizado o sistema Linux Ubuntu versão 16.04 lts (*long term support*) e o PetaLinux versão 2017.1. Em seguida utiliza-se um BSP, este pode ser baixado diretamente no site da placa utilizada, para gerar os componetes básicos do sistema PetaLinux, portanto, nele encontramos o Kernel, o Rootfs e o *bootloader*.

Gerado os componentes básicos é necessário atualizar o sistema PetaLinux ao *hardware* a ser utilizado no projeto, para tal utiliza-se o Vivado IDE para desenvolver um *block design* e por fim um arquivo *bitstream*, a partir do arquivo *bitstream* é possível adaptar o Petalinux, gerado anteriormente, para o *hardware* desejado. Caso o usuário deseje desenvolver programas a serem utilizados dentro do PetaLinux deve-se utilizar o Xilinx SDK, selecionando o sistema Linux como sistema operacional e em seguida adicionar o programa aos arquivos do PetaLinux.

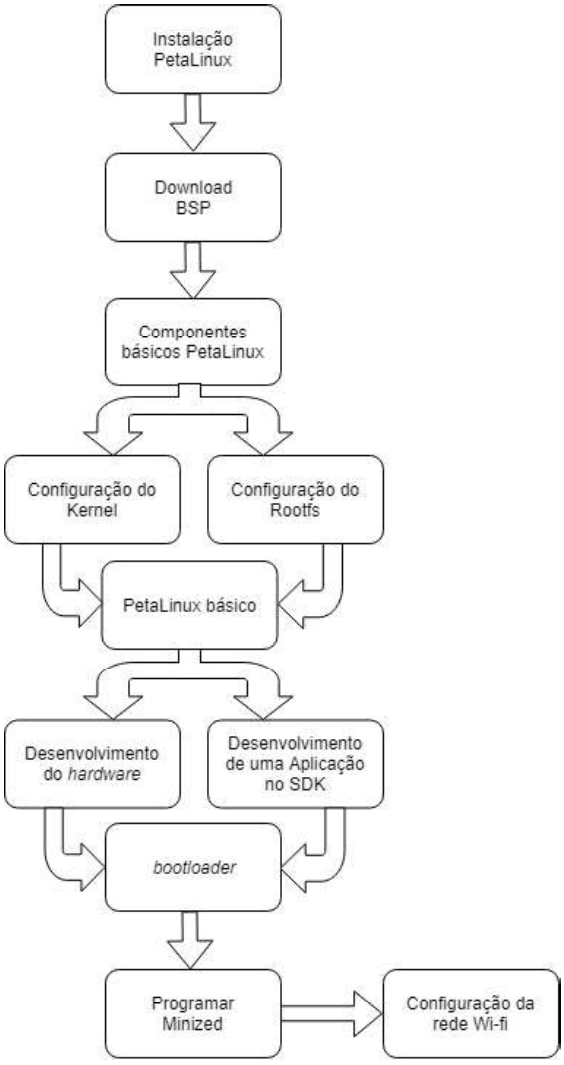
Ao final do projeto do PetaLinux será gerado um arquivo .BIN, *bootloader*, este arquivo será responsável por inicializar e gerar o PetaLinux dentro da placa Minized. Utilizando o Xilinx SDK programamos o arquivo .BIN na Minized. Finalizado estes

procedimentos temos um sistema Linux embarcado funcional em operação na Minized.

Para utilizar o módulo Murata basta acessar, através do PetaLinux da Minized, o arquivo wpa_supplicant.conf, neste arquivo pode-se adicionar o nome e a senha da rede a ser utilizada para a comunicação sem fio.

É apresentado na Figura 25, em diagrama de blocos, os passos necessários para o desenvolvimento do sistema Linux embarcado.

Figura 25 - Diagrama de blocos referente ao desenvolvimento do sistema Linux embarcado

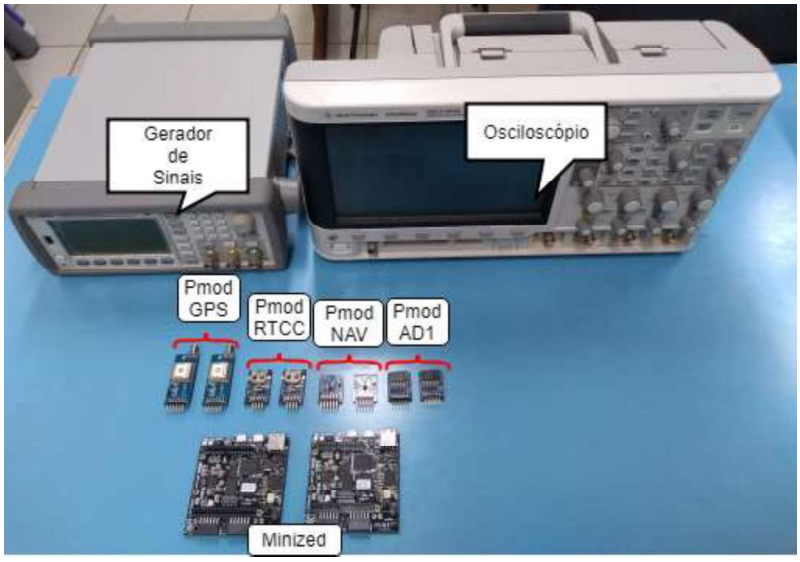


Fonte: autoria própria.

3 RESULTADOS

Neste capítulo são descritos os experimentos realizados em bancada e em sequência as análises dos resultados de cada teste realizado. Para os testes montou-se um *setup* experimental que consiste em: dois Minizeds, 2 Pmods AD1, 2 Pmods RTCC, 2 Pmods GPS, 2 Pmods NAV, um osciloscópio, um gerador de sinais e um computador. É mostrado na Figura 26 os equipamentos, exceto o computador, usado para a realização dos testes.

Figura 26 - Setup experimental



Fonte: autoria própria

3.1 Minized+ADC

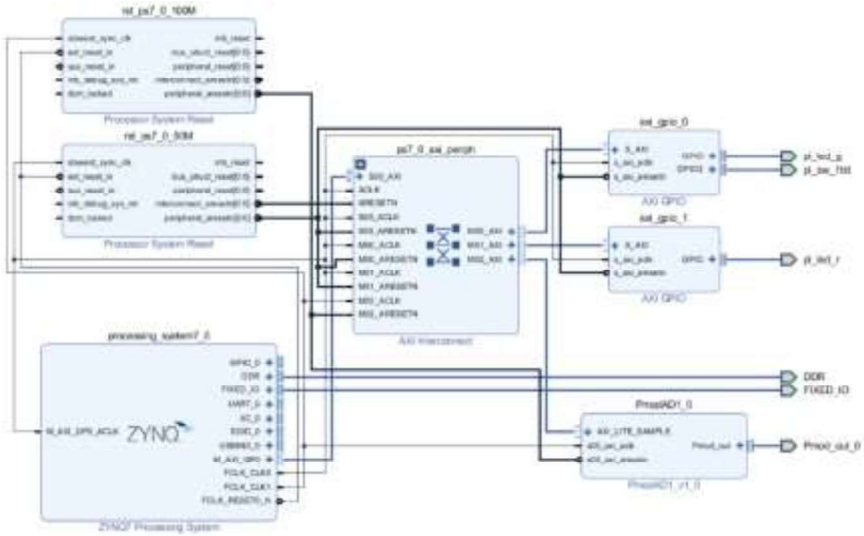
Esse experimento foi realizado afim de descobrir se a resolução do relógio da Minized, através de medida indireta, atende ou não o requisito do projeto, sendo o requisito do projeto tempos de atrasos entre as placas em escalas iguais ou inferiores a microssegundos.

Para o experimento foi utilizado os seguintes componentes:

- 1 Minized;
- 1 Pmod AD1;
- 1 gerador de sinal;
- 1 computador.

Inicialmente foi realizado a modelagem do *hardware* através do *software* Vivado IDE, para tal gerou-se o *block design* como apresentado na Figura 27.

Figura 27 - Block design Minized+ADC

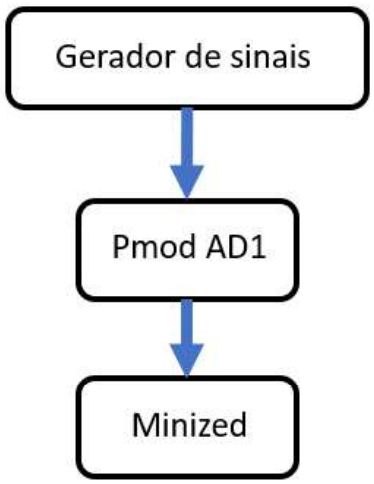


Fonte: autoria própria

Após o PL do SoC Zynq estar configurado, programou-se um código em linguagem C, no *software* Xilinx SDK, capaz de realizar a aquisição de dados a uma frequência de amostragem de 5 KHz.

O teste foi realizado acoplando o Pmod AD1 ao Minized e em seguida o Pmod foi conectado a saída do gerador de sinais. A Figura 28 mostra o *setup* experimental através do diagrama de blocos.

Figura 28 - Diagrama de blocos Minized+ADC



Fonte: autoria própria

Para o teste foi utilizado uma onda senoidal de amplitude de 2 Vpp, com frequência de 1 KHz e *offset* de 1 V, tal *offset* foi necessário para que a onda possuísse apenas valores positivos, visto que o Pmod AD1 aceita apenas valores positivos, sendo este valor subtraído quando realizado os cálculos para o experimento.

Para a determinação da resolução do relógio da Minized, é necessário calcular a diferença de tempo entre amostras e para realizar esta diferença utilizou-se a seguinte equação:

$$\Delta t[n] = \left| \frac{\text{acos}(V[n]) - \text{acos}(V[n-1])}{2 * \pi * f} \right| \quad (2)$$

Sendo:

⊿t[n] é a n-ésima variação de tempo (s)

V[n] é a n-ésima tensão (V);

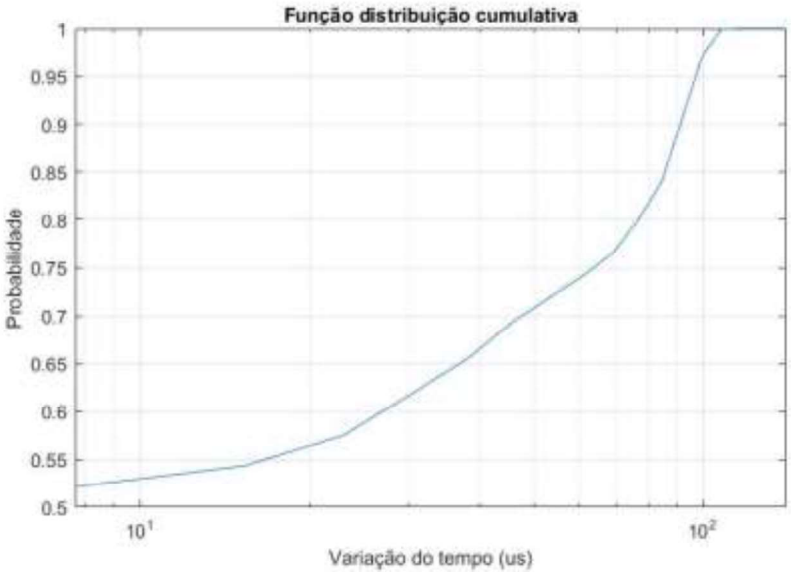
f é a frequência da senóide.

Ao coletar os dados do gerador de sinais, o valor de 1 foi subtraído de cada amostra, afim de corrigir o *offset* aplicado anteriormente. Posteriormente foi aplicado, nos dados corrigidos, a equação (3).

$$y[n] = \Delta t[n] - E\{\Delta t\} \quad (3)$$

Sendo *E{Δt}* a média dos valores da sequência obtida em (2). A partir dessa nova sequência, gerou-se a função distribuição cumulativa, como mostrado na Figura 29.

Figura 29 - Função distribuição cumulativa Pmod AD1



Fonte: autoria própria

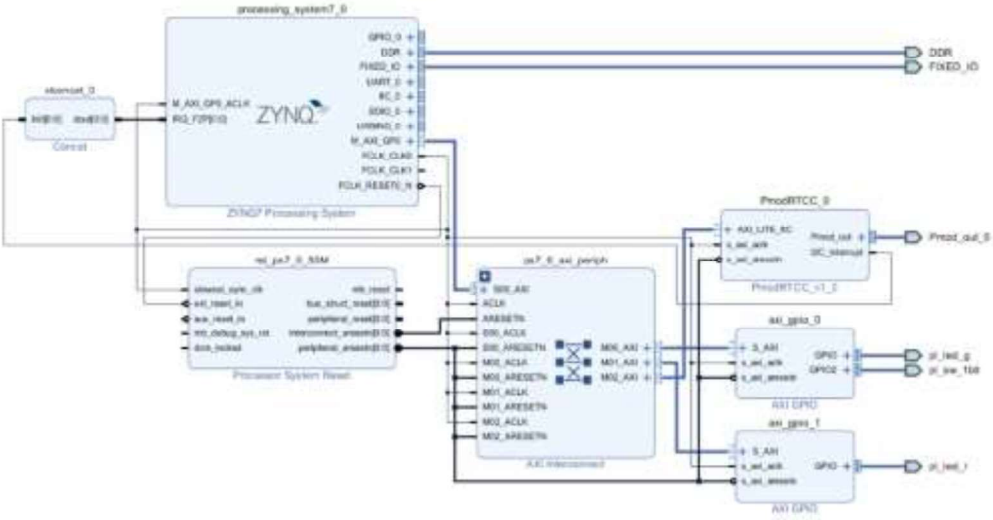
Analisando o resultado do teste da resolução do relógio, pode-se notar que a variação do tempo entre as amostras encontra-se em algumas dezenas até centena de µs. Como a frequência de amostragem utilizada é de 5 KHz, o intervalo entre duas amostras sucessivas é 200 µs e a imprecisão máxima é metade desse valor, ou seja, 100 µs. Observe que a função distribuição cumulativa indica um valor menor ou igual 100 µs em 100% das amostras.

3.2 Minized+RTCC

Este experimento foi realizado afim de verificar a precisão do módulo Pmod RTCC na utilização do correlacionamento de sinais.

A Figura 30 apresenta o *block design* gerado no *software* Xilinx Vivado.

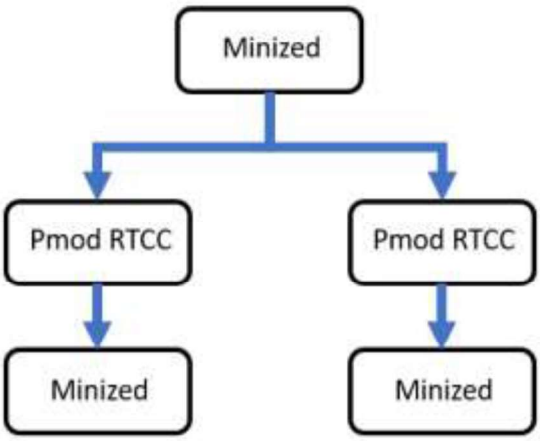
Figura 30 - Block design Minized+RTCC



Fonte: autoria própria

Neste experimento utilizou-se dois Minizeds, dois Pmod RTCC e um computador. O teste consistiu em configurar previamente os dois Pmods RTCC simultaneamente (a partir de uma placa Minized) e analisar a estabilidade dos relógios entre os dois Pmods. Caso a diferença entre os relógios fosse pequena (algumas centenas de µs), seria possível empregar apenas os módulos RTCC na operação de sincronismo entre as placas e acionar o Pmod AD1 na aquisição dos dados. É mostrado na Figura 31 o diagrama de blocos do *setup* experimental.

Figura 31 - Diagrama de blocos Minized+RTCC

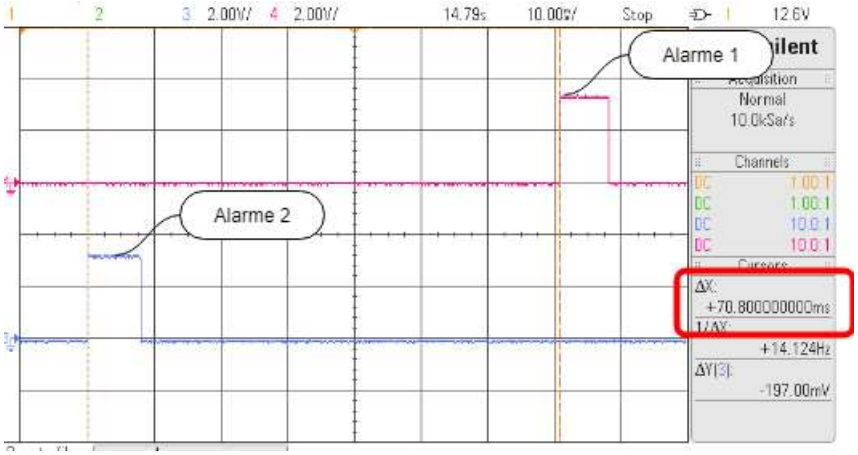


Fonte: autoria própria

No teste analisou-se os sinais de alarmes gerados pelo pino J1 em ambos os módulos Pmod RTCC, a partir deles foi verificado a diferença de tempo entre os

alarmes. É apresentado na Figura 32 os sinais de alarme dos dois Pmod RTCC obtidos no osciloscópio.

Figura 32 - Sinais de alarme Minized+Pmod RTCC



Fonte: autoria própria

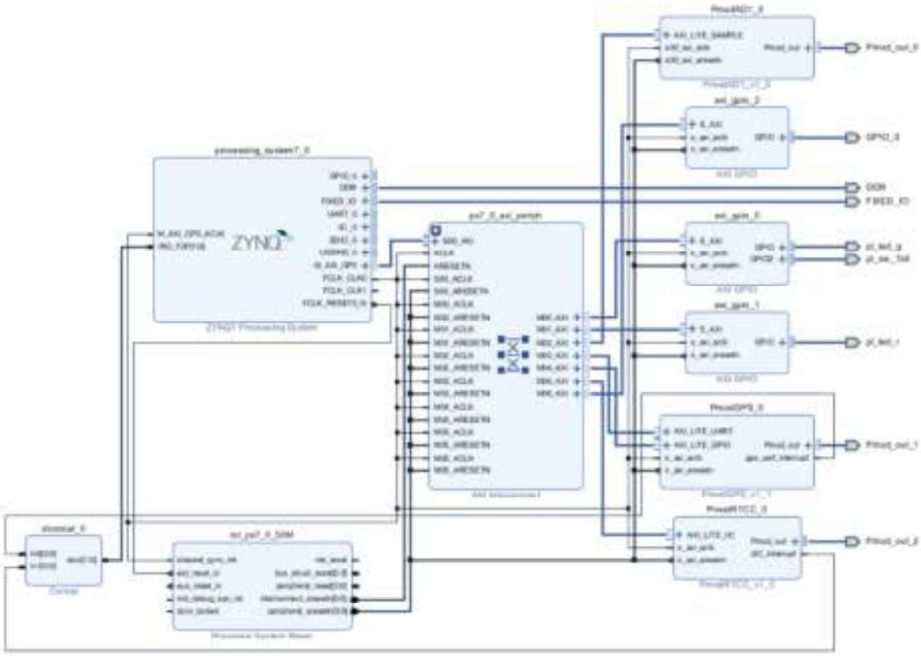
Pela Figura 32 observa-se que a diferença entre os dois sinais de alarmes foi de 70,8 ms. Baseado na equação (1) e considerando a velocidade de propagação do ruído de vazamento nas tubulações plásticas de 544 m/s, segundo Brennan (2018), para essa diferença de tempo tem-se um erro de ±38,51 m. Portanto, o emprego somente de módulos Pmod RTCC na operação de sincronismo não atende o requisito do projeto.

3.3 Minized+ADC+RTCC+GPS

Como o módulo Pmod RTCC não possui precisão suficiente exigido pelo projeto, adicionou-se um módulo Pmod GPS para configurar o RTCC toda vez que fosse realizada uma medição. Além de funcionar como relógio inicial para os módulos RTCC, o módulo GPS atua como uma sincronização fina para a inicialização do Pmod AD1 através do PPS (Pulso Por Segundo). Assim, para iniciar a aquisição de sinais será necessário esperar dois sinais de disparo: o primeiro é o alarme gerado pelo Pmod RTCC e em seguida pelo próximo PPS gerado pelo GPS. Havendo os dois sinais, inicia-se a aquisição de dados pelo Pmod AD1.

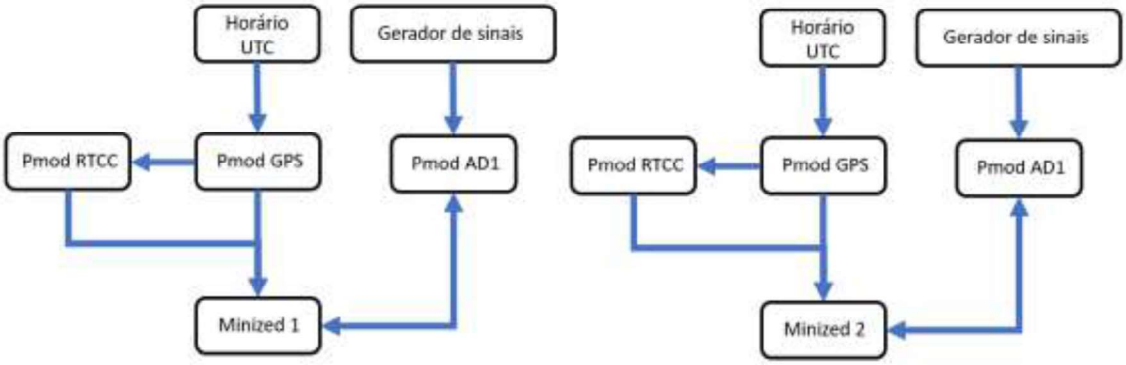
A Figura 33 apresenta o *block design* desenvolvido no *software* Xilinx Vivado IDE e a Figura 34 ilustra o diagrama de blocos da integração de uma das placas Minized com os módulos Pmod.

Figura 33 - Block design Minized+ADC+RTCC+GPS



Fonte: autoria própria

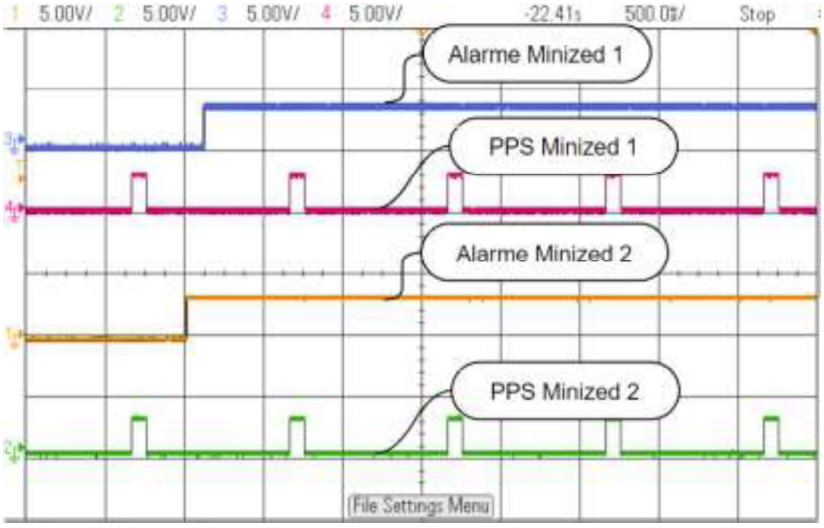
Figura 34 - Diagrama de blocos Minized+ADC+RTCC+GPS



Fonte: autoria própria

Neste experimento configurou-se o Pmod RTCC de cada um dos Minizeds através do módulo GPS, após um tempo pré-determinado o RTCC aciona um alarme o qual irá esperar o próximo PPS para inicializar o Pmod AD1 para executar as medições. É mostrado na Figura 35 os sinais do PPS e alarme das duas placas Minizeds no osciloscópio.

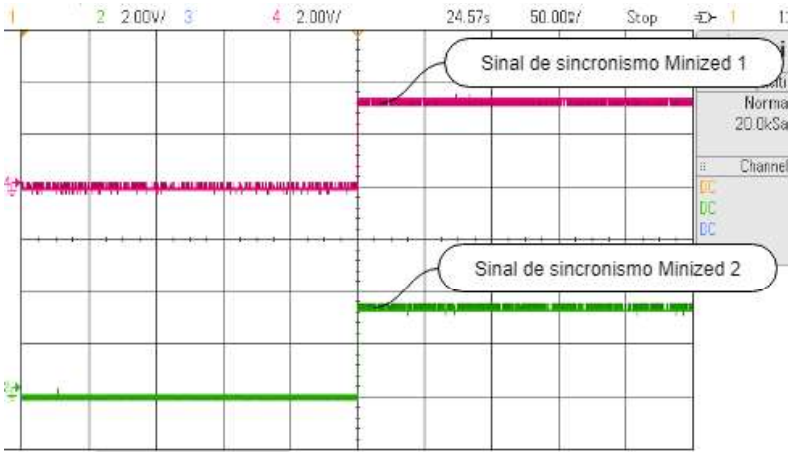
Figura 35 - Sinais de alarme e PPS



Fonte: autoria própria

Para visualizar e medir o sincronismo através do osciloscópio, adicionou-se um pino na programação realizada no SDK, este pino é mantido em nível lógico baixo, quando os sinais de alarme e PPS são identificados o pino altera para nível lógico alto, sendo possível visualizar o momento exato no qual o Pmod AD1 estará entrando em funcionamento. A Figura 36 apresenta o sinal de sincronismo das duas placas Minizeds, analisando a Figura 36 observa-se que as placas estão em sincronismo, vale ressaltar que o atraso depende da resolução temporal empregada no osciloscópio.

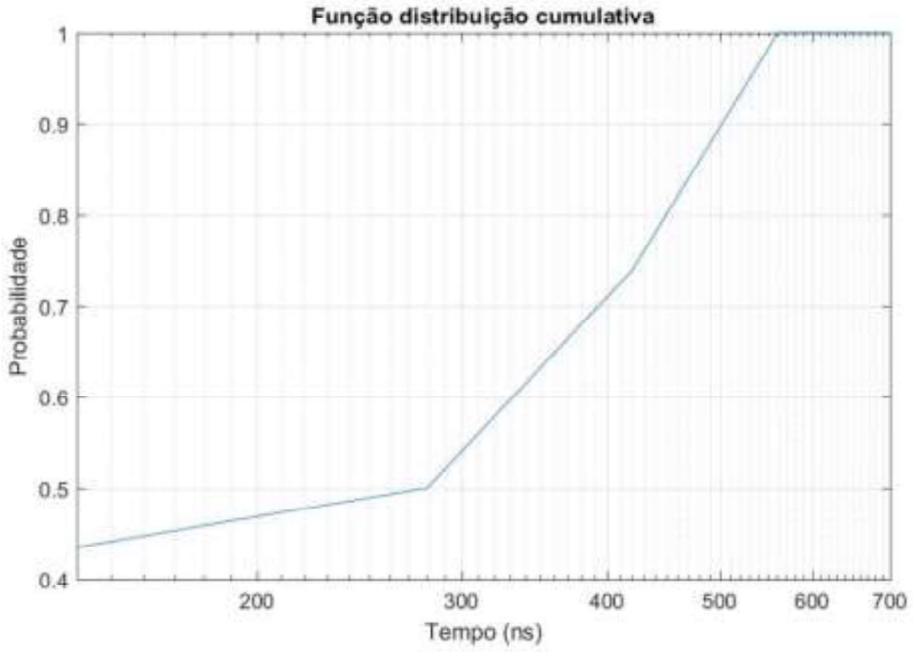
Figura 36 - Sinais de sincronismo



Fonte: autoria própria

Para a análise do sincronismo foram obtidas 50 amostras em períodos diferentes (dia e hora), afim de verificar eventuais diferenças. Com essas amostras gerou-se uma Função Distribuição Cumulativa, mostrada na Figura 37.

Figura 37 - Função distribuição cumulativa Minized+ADC+RTCC+GPS



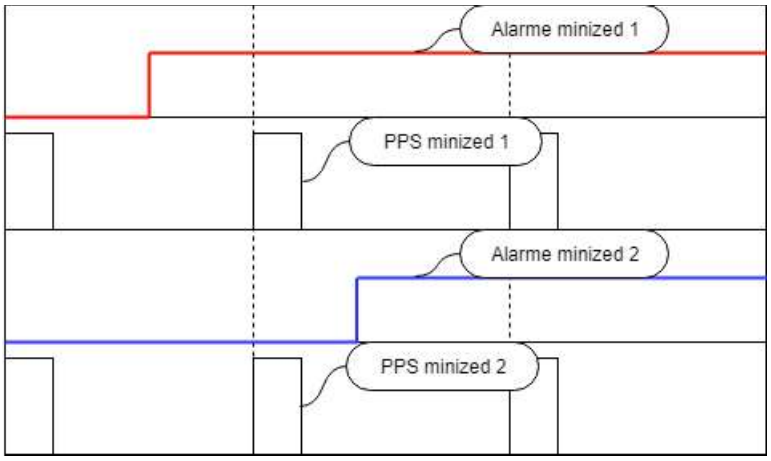
Fonte: autoria própria

Analisando a Figura 37 observa-se que a diferença de tempo entre os sinais de sincronismo está na faixa de centena de ns e com 90% das medidas obtidas com atraso menor ou igual a 500 ns.

Considerando a velocidade de propagação do ruído de vazamento nas tubulações plásticas, segundo Brennan (2018), para essa diferença de tempo tem-se um erro menor ou igual de ± 0,25 mm em 90% das tentativas de sincronismo. Esse resultado valida o protótipo com relação ao sincronismo do correlacionador de sinais.

Analisando o comportamento do sinal do PPS e os dois alarmes, verifica-se que há a possibilidade de os alarmes estarem posicionados de tal forma que o pulso do PPS esteja entre os dois sinais de alarme. É ilustrado na Figura 38 o comportamento descrito anteriormente.

Figura 38 - Possível erro de sincronismo

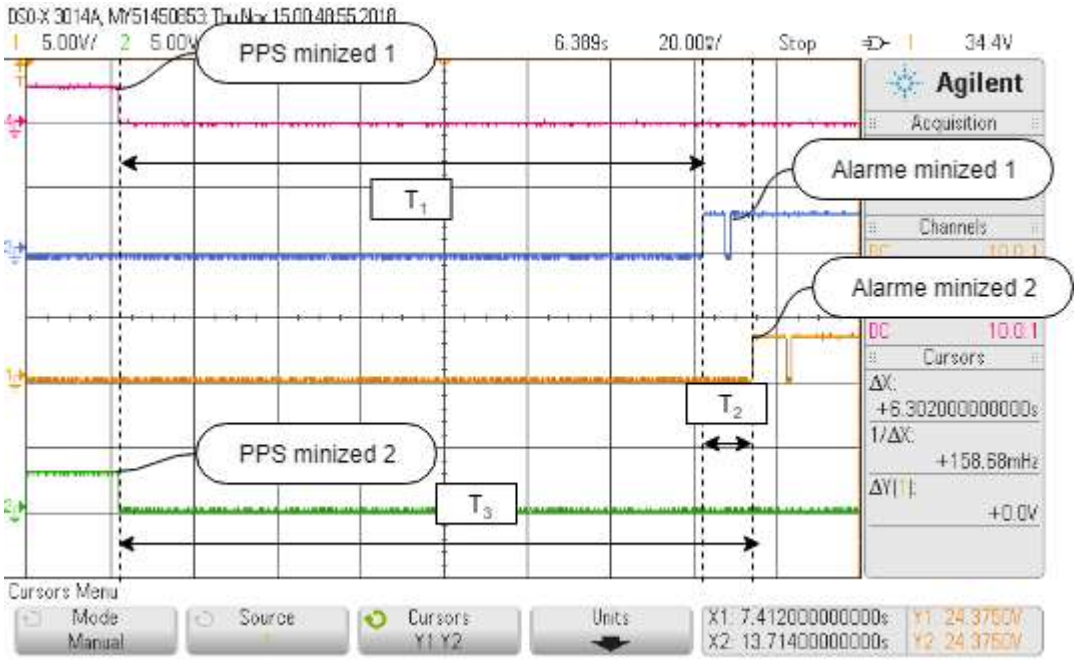


Fonte: autoria própria

Para determinar se este erro de sincronismo pode ocorrer e qual a probabilidade dele ocorrer, foram realizadas 60 medições em períodos diferentes (dia e hora), com o intuito de verificar a diferença de tempo entre o sinal de PPS e o sinal de alarme mais próximo (T_1) e a diferença de tempo entre os dois sinais de alarme (T_2), caso o somatório dos tempos ($T_3 = T_1 + T_2$) for menor que 900 ms, não há o erro de sincronismo.

É apresentado na Figura 39 como a medição foi realizada.

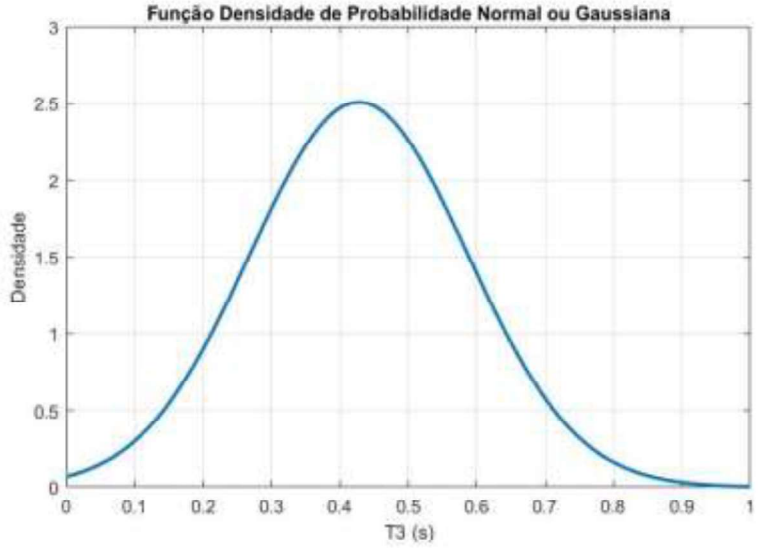
Figura 39 - Medição realizada



Fonte: autoria própria

A partir dos dados coletados gerou-se a Função Densidade de Probabilidade Normal ou Gaussiana, como mostrada na Figura 40.

Figura 40 - Função Densidade de Probabilidade



Fonte: autoria própria

Para analisar a probabilidade da ocorrência de T_3 maior do que 900 ms utilizou-se a seguinte equação.

$$Prob\{T_3 > 0{,}9\} = \int_{0{,}9}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-(t-\mu)^2/2\sigma^2} dt \quad (4)$$

Sendo:

μ é a média com valor de 0,4276;

σ é o desvio padrão com valor de 0,1591.

De (4) tem-se que a probabilidade de ocorrência deste erro é de 0,009043%.

3.4 Minized+NAV

Este teste foi realizado com o intuito de verificar se o Pmod NAV possui a sensibilidade exigida pelo projeto. O uso do Pmod NAV dispensaria a utilização do acelerômetro PCB 333b (PCB PIEZOTRONICS, 2002) e do Pmod AD1 na aquisição

dos dados, pois o Pmod NAV possui um circuito conversor analógico-digital interno além do acelerômetro.

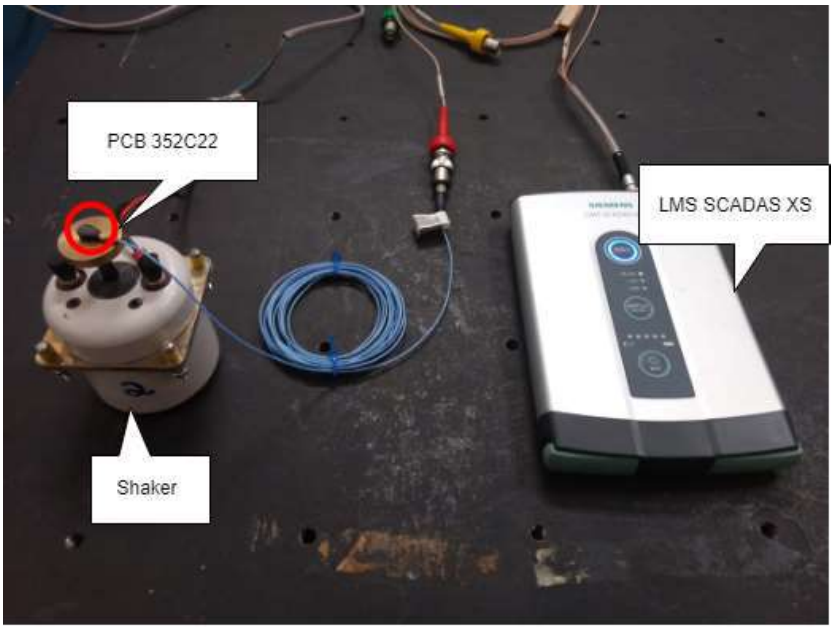
O *setup* experimental usou um Minized, um Pmod NAV, um *shaker* modelo LDS V 101 (LDS, 2002), sendo este responsável pela simulação de m vazamento, um amplificador modelo XLS 1000 (CROWN, 2015), o equipamento LMS SCADAS XS (SIEMENS, 2018) e um acelerômetro modelo PCB 352C22 (PCB PIEZOTRONICS, 2018).

O teste consiste em gerar um ruído branco através de um software (micro), a saída é transmitida para o amplificador XLS 1000 que regula o ganho do ruído antes de enviar ao *shaker,* que por sua vez reproduz um ruído branco (vibração) com frequências variando entre 0 a 1 KHz.

Empregou-se o LMS SCADAS XS e o acelerômetro PCB 352C22 como equipamento de referência na aquisição de dados, com taxa de amostragem de 8192 Hz, da vibração do *shaker*. Foram obtidas 8192 amostras.

A Figura 41 apresenta o *setup* experimental utilizado para a medição utilizando o LMS SCADAS XS.

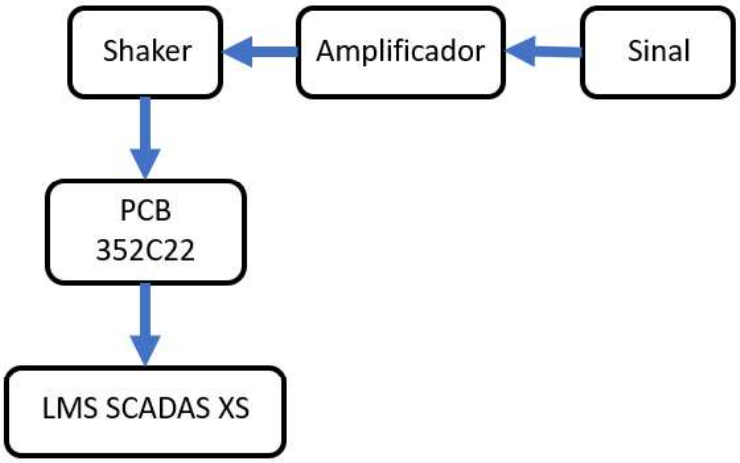
Figura 41 - Setup experimental LMS SCADAS XS + PCB 352C22



Fonte: autoria própria

Na Figura 42 é mostrado o diagrama de blocos referente a aquisição de dados utilizando o LMS SCADAS XS e o acelerômetro PCB 352C22.

Figura 42 - Diagrama de blocos referente aquisição de dados pelo LMS SCADAS XS

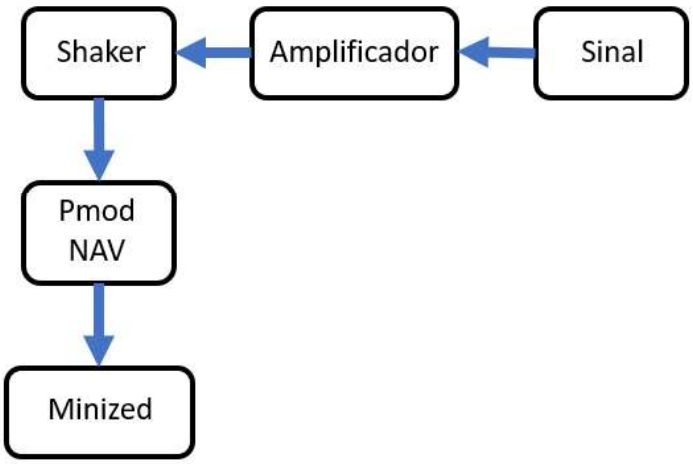


Fonte: autoria própria

Em seguida foi realizado o mesmo teste, porém utilizando a placa Minized e o módulo Pmod NAV. O Pmod NAV, diferente do PCB 352C22, possui uma frequência de aquisição máxima de 952 Hz, portanto esta foi a frequência de amostragem utilizada no teste e foram medidas 1024 amostras.

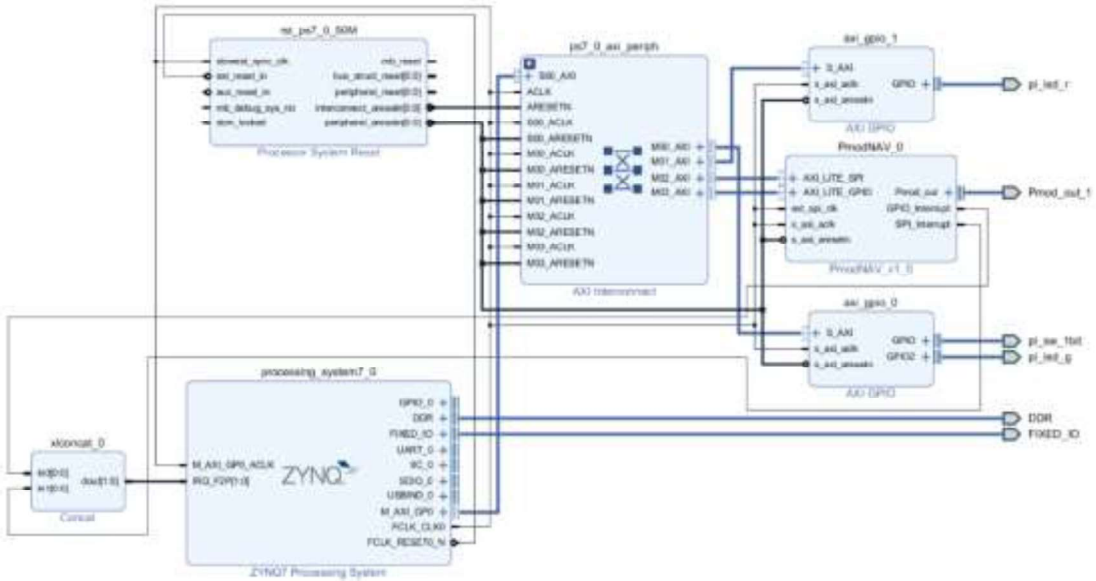
É ilustrado na Figura 43 o diagrama de blocos referente a aquisição de dados utilizando a placa Minized e a Figura 44 apresenta o *block design* desenvolvido no Xilinx Vivado IDE.

Figura 43 - Diagrama de blocos referente a aquisição de dados utilizando a placa Minized



Fonte: autoria própria

Figura 44 - Block design Minized+NAV



Fonte: autoria própria

Como o Pmod NAV não possui uma frequência de amostragem equivalente ao do PCB 352C22, foi necessário realizar a interpolação linear das amostras obtidas. Com o intuito de utilizarmos a mesma quantidade de amostras entre o Pmod NAV e o PCB 352C22, realizou-se uma interpolação linear de 8 nas amostras obtidas pelo Pmod NAV.

Para uma melhor comparação entre as amostras coletadas pelo LMS SCADAS XS e a placa Minized, utilizou-se a Equação (5) para que a sequência da amostra tenha sua média centrada em zero.

$$y[n] = x[n] - \bar{x} \tag{5}$$

Sendo *y[n]* é o novo valor da amostra original *x[n]* subtraído do valor da média das amostras da sequência ($\bar{x}$).

Em seguida foi realizada a normalização das amostras, afim de excluir a influência da sensibilidade e ganho entre os dispositivos, utilizou-se a seguinte equação:

$$z[n] = \frac{y[n]}{\sqrt{\frac{1}{N}\sum_{k=1}^{N} y^2[k]}} \tag{6}$$

A seguir calculou-se a Densidade Espectral de Potência (PSD) das duas sequências de amostras (referência e Pmod Nav) normalizadas, na faixa de frequência de 0 a 2 KHz, sendo (7), segundo (KIHONG, HAMMOND, 2008).

$$\bar{S}_{xx} = \frac{(\Delta t)^2}{T}\left|\sum_{n=1}^{N} z[n]e^{i\omega n\Delta t}\right|^2 \quad (7)$$

Sendo:

Δt a variação de tempo entre as amostras;

T o período do sinal analisado;

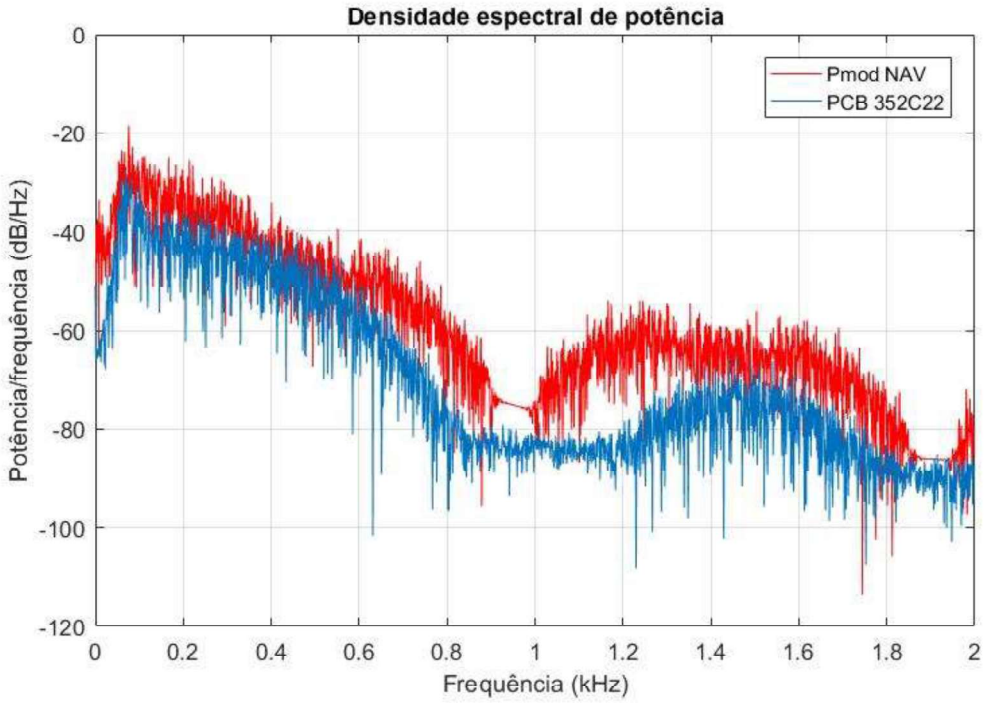
N o número total de amostras;

z[n] o valor da *n-ésima* amostra;

ω a frequência angular do sinal analisado.

Na Figura 45 é mostrado o PSD de cada acelerômetro.

Figura 45 - Densidade Espectral de Potência



Fonte: autoria própria

Analisando a Figura 45 observa-se que a faixa de passagem dos dois PSD situa-se em torno de 600 Hz e a faixa de transição entre 600 a 800 Hz. A faixa de parada/rejeição dos dois PSD seguem a mesma tendência até 1 KHz, mas a partir dessa frequência os dois PSD divergem ligeiramente no comportamento. Uma das

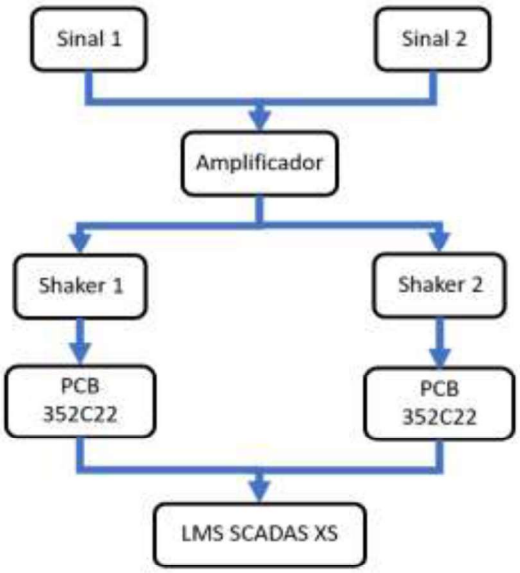
possíveis causas da divergência deve-se a interpolação linear de 8 usada no Pmod NAV, mas independentemente disso o PSD está coerente pelo fato do ruído branco gerado no *shaker* ter uma largura de banda de 1 KHz. A princípio o PmodNAV pode ser utilizado no correlacionador desde que o ruído possua uma largura de banda em torno de 1 KHz.

3.5 Minized+RTCC+GPS+NAV

Este teste foi realizado para verificar o sincronismo das placas juntamente com a aquisição dos dados utilizando o Pmod NAV. O teste consiste na utilização de dois *shakers*, os quais simulam o vazamento em tubulações enterradas através de dados coletados no campo de prova da Sabesp. Inicialmente foi realizado o teste utilizando o LMS SCADAS XS e dois acelerômetros PCB 352C22, para fins de comparação com o sistema utilizando a Minized.

Na Figura 46 é mostrado o diagrama de blocos referente ao experimento utilizando o LMS SCADAS XS.

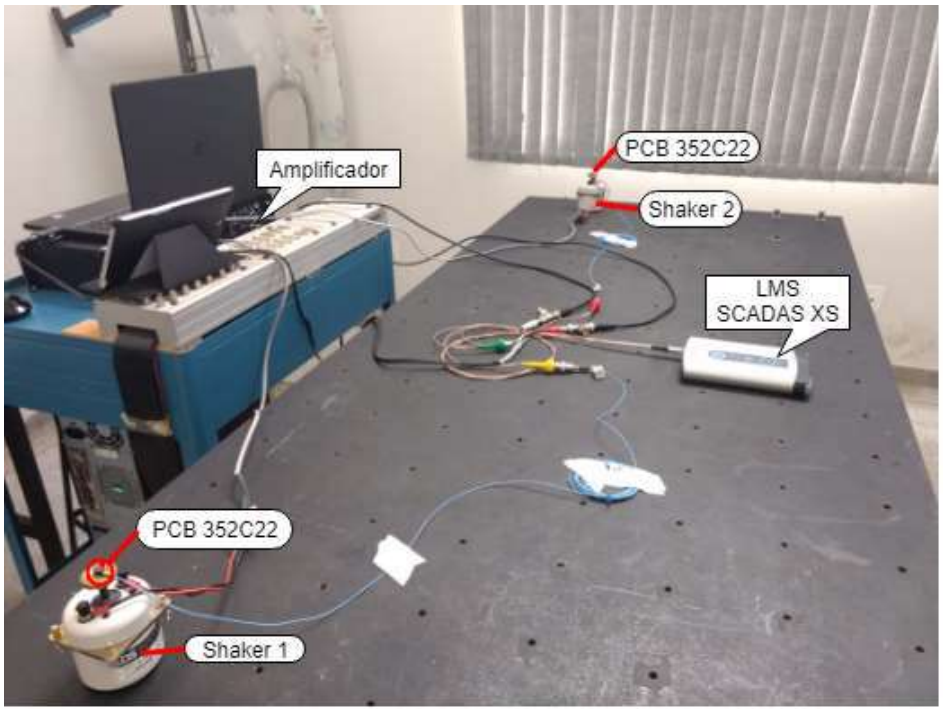
Figura 46 - Diagrama de blocos referente a simulação do vazamento com o LMS SCADAS XS



Fonte: autoria própria

A Figura 47 apresenta o *setup* experimental utilizado no teste empregando o LMS SCADAS XS.

Figura 47 - Setup experimental LMS SCADAS XS teste de sincronismo



Fonte: autoria própria

O teste consistiu na geração de dois sinais de áudio, baseado nos dados coletados na Sabesp, cada um dos sinais passa por um amplificador para que possa ser controlado a amplitude dos sinais e em seguida cada sinal será enviado para um dos *shakers*, sendo possível simular um vazamento em uma tubulação enterrada.

Preparado os equipamentos responsáveis pela simulação do vazamento, utilizou-se o LMS SCADAS XS e 2 acelerômetros PCB 352C22 para realizar a medição em cada um dos *shakers*, as aquisições foram realizadas na frequência de amostragem de 8192 Hz.

Foi realizado a medição num período de 1 segundo, coletando um total de 8192 amostras , após a coleta dos dados, aplicou-se as Equações (5) e (6) afim de realizar a normalização dos dados, em seguida realizou-se a correlação cruzada entre os dados medidos no *shaker* 1 e os medidos no *shaker* 2, com o intuito de encontrar o ponto de maior similaridade entre os dados, este dado é de extrema importância, visto

que através deste dado é possível calcular o atraso de tempo entre os dois sinais e aplicando na Equação (1) é possível determinar a posição do vazamento.

Para a realização da correlação cruzada utilizou-se a equação a seguir (KIDO, 2014).

$$R_{XY}[m] = \sum_{n=1}^{N} x[n+m]y[n] \tag{8}$$

Sendo:

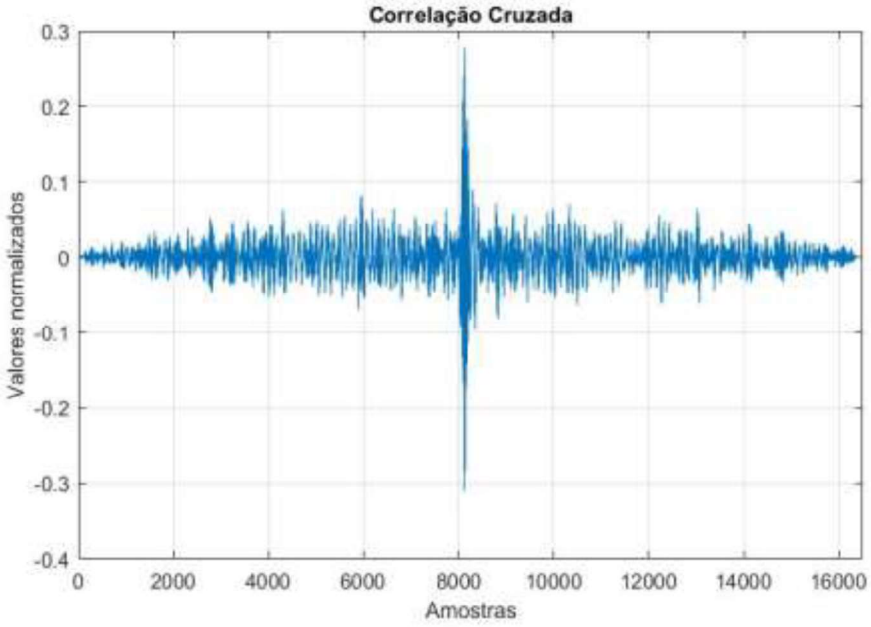
N o número total de amostras;

x[n] o valor da *n-ésima* amostra do *shaker* 1

y[n] o valor da *n-ésima* amostra do *shaker* 2

A Figura 48 mostra o gráfico da correlação cruzada.

Figura 48 - Correlação cruzada experimento LMS SCADAS XS



Fonte: autoria própria

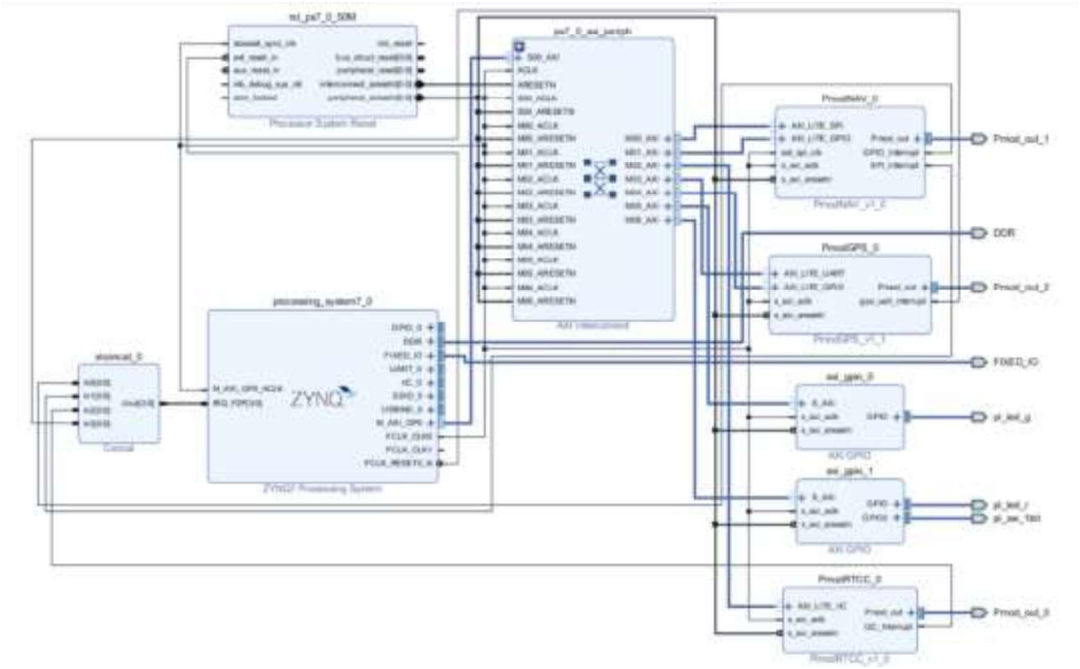
Através da correlação cruzada encontrou-se que o ponto de maior amplitude se localiza na amostra 8129, equivalente ao tempo de 0,9923 s.

Em seguida realizou-se o mesmo experimento, porém utilizando duas placas Minized, 2 Pmods RTCC, 2 Pmods GPS e 2 Pmods NAV. Devido a limitação da

frequência de aquisição do Pmod NAV, a frequência de aquisição foi de 952 Hz, portanto para um período de 1 s obteve-se 952 amostras.

A Figura 49 apresenta o *block design* desenvolvido no Xilinx Vivado IDE.

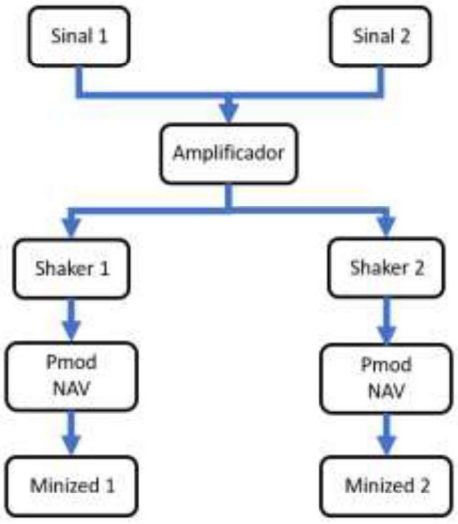
Figura 49 - Block design Minized+RTCC+GPS+NAV



Fonte: autoria própria

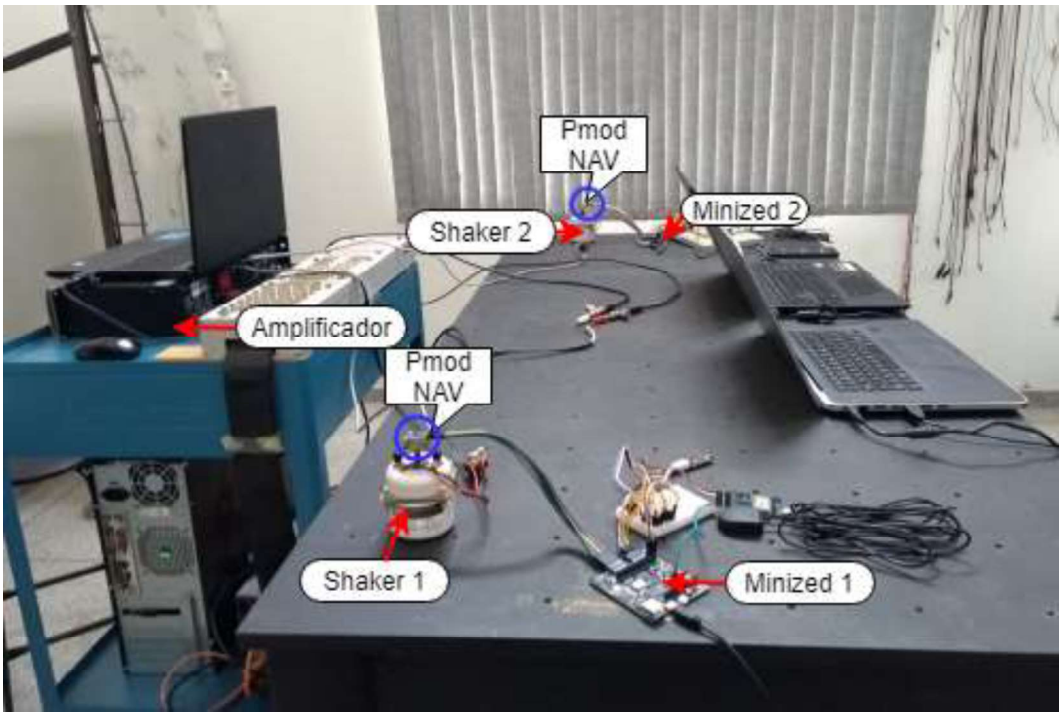
É apresentado na Figura 50 o diagrama de blocos referente ao experimento utilizando a placa Minized e na Figura 51 é apresentado o *setup* experimental utilizado no teste empregando a placa Minized.

Figura 50 - Diagrama de blocos referente a simulação do vazamento com a Minized



Fonte: autoria própria

Figura 51 - Setup experimental placa Minized teste de sincronismo



Fonte: autoria própria

Aplicou-se os mesmos sinais utilizados no teste usando o LMS SCADAS XS, diferentemente do teste do LMS SCADAS XS foi necessário utilizar dois conjuntos Minized+RTCC+GPS+NAV, portanto foi necessário que ambos os conjuntos estivessem sincronizados para a realização das aquisições.

Ao aplicar a correlação cruzada entre os sinais medidos, percebeu-se a presença de ruídos, o que interferia na análise do mesmo. Para diminuir o ruído presente nos sinais medidos utilizou-se o filtro de média móvel como descrito na Equação (9) (SMITH, 1999).

$$y[n] = \frac{1}{W}\sum_{m=0}^{W-1} x[n+m] \tag{9}$$

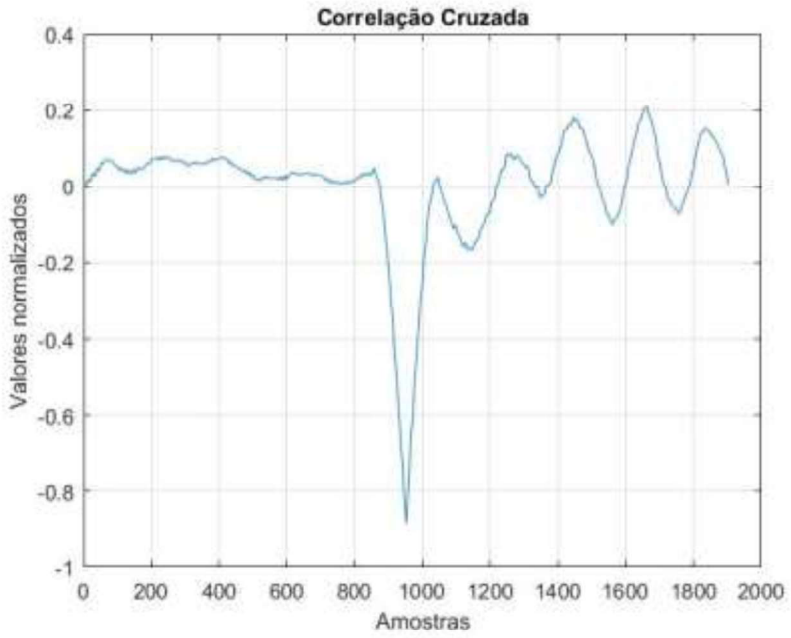
Sendo:

W é o tamanho, em amostras, da média móvel;

x[n] o valor da *n-ésima* amostra.

Após a diminuição do ruído nos sinais amostrados, empregando um tamanho de janela (*W*) de 100, aplicou-se as Equações (5) e (6) para a normalização dos dados e em seguida empregou-se a Equação (8) para obter a correlação cruzada entre os sinais. É apresentado na Figura 52 o gráfico de correlação cruzada entre os sinais medidos utilizando a placa Minized.

Figura 52 - Correlação cruzada experimento Minized



Fonte: autoria própria

Analisando a Figura 52 verificou-se que o valor máximo, em módulo, encontra-se na amostra 952, equivalente ao tempo de 1 s.

Através dos experimentos realizados verifica-se uma diferença de 7,7 ms no tempo entre o LMS SCADAS XS e a placa Minized, esta diferença pode ter ocorrido devido à resistência física dos cabos utilizados no Pmod NAV, como seus cabos possuíam uma certa resistência física e os sinais gerados pelos *shakers* eram de baixas amplitudes, pode ter ocorrido uma pequena diferença na aquisição devido a essa influência, porém como mostrado anteriormente o resultado encontrado foi muito próximo ao resultado do LMS SDACAS XS, assim sendo possível a sua utilização na detecção de vazamentos em tubulações enterradas.

4 CONSIDERAÇÕES FINAIS

Analisando o primeiro experimento pode-se concluir que o relógio interno da placa Minized possui uma resolução suficiente para atender aos requisitos do projeto, sendo assim, o tempo de processamento da placa não influenciará no resultado final do projeto. No segundo experimento pretendia-se realizar o sincronismo entre as placas utilizando o Pmod RTCC, este previamente configurado, porém com os resultados do teste verificou-se que ao longo do tempo os relógios dos Pmods RTCC apresentaram uma diferença muito elevada para os parâmetros do projeto, assim sendo, essa solução foi descartada.

Afim de solucionar o problema do sincronismo, foi utilizado simultaneamente o Pmod RTCC e o Pmod GPS, assim pode-se configurar o RTCC todas as vezes que fosse feito a aquisição dos dados, resolvendo assim o problema da discrepância de tempo entre os RTCCs ao longo do tempo, portanto esse método de sincronismo é eficaz e atende aos requisitos do projeto. Finalizado o sincronismo entre as placas, analisou-se a utilização do acelerômetro, de baixo custo, do Pmod NAV, realizando testes entre o protótipo da placa Minized e o LMS SCADAS XS, verificou-se que o Pmod NAV quando submetido a uma simulação de vazamento apresentou um erro de 0,776% no resultado em relação ao equipamento de referência.

Pode-se concluir que a partir dos resultados apresentados neste trabalho, a utilização do SoC Zynq em aplicações de correlacionamento de sinais é viável, visto que atende os principais requisitos da técnica e além disso, o SoC Zynq mesmo utilizando uma FPGA, cuja programação é menos difundida que as demais linguagens de programação, possui ferramentas que facilitam sua programação, tanto na parte do *hardware*, na qual utilizou-se IP *blocks* quanto na do *software,* onde a linguagem utilizada foi a linguagem C.

Os trabalhos futuros estarão voltados na união dos códigos realizados nos experimentos com o Linux embarcado, possibilitando a aquisição e transferências dos dados de forma que o protótipo possa ser utilizado de forma rápida e sem complicações ao usuário.

Também será realizado uma modificação no código de sincronismo, afim de evitar o erro relacionado a diferença de tempo entre os alarmes dos Pmods RTCC,

para tal será realizado a aquisição do relógio do GPS sempre após ocorrer o PPS, assim evitando que os RTCC possuam diferenças de tempo.

Por fim serão realizados testes no campo de prova da Sabesp para verificar se o protótipo desenvolvido atende os requisitos necessários fora de um ambiente controlado como no laboratório.

REFERÊNCIAS

ALMEIDA, F. C. L. et al. An investigation into the effects of resonances on the time delay estimate for leak detection in buried plastic water distribution pipes. In: INTERNATIONAL CONFERENCE ON STRUCTURAL DYNAMICS - EURODYN, 9th 2014. **Proceedings** […] Porto: [s. n.], 2014.

ANALOG DEVICES. 2.35 V to 5.25 V, 1 MSPS,12-/10-/8-Bit ADCs in 6-Lead SC70 AD7476A/AD7477A/AD7478A. **Analog devices**. [S. l.], 2018. Disponivel em: <http://www.analog.com/media/cn/technical-documentation/evaluation-documentation/AD7476A_7477A_7478A.pdf>. Acesso em: 5 out. 2018.

ARM. Cortex-A9. **armDeveloper**. [S. l.], 2018. Disponivel em: <https://developer.arm.com/products/processors/cortex-a/cortex-a9>. Acesso em: 12 out. 2018.

AVNET. Integrating Sensors on Minized with PetaLinux (2017.1 and 2017.4). **Avnet.** [S. l.], 2017. Disponivel em: <http://zedboard.org/course/integrating-sensors-Minized-petalinux-20171-and-20174>. Acesso em: 28 Out 2018.

AVNET. **Minized Hardware User Guide**. [S. l.], 2017.

AVNET. Minized. **Zedboard**. [S. l.], 2018. Disponivel em: <http://zedboard.org/product/Minized>. Acesso em: 1 out. 2018.

AVNET. ZedBoard. **Zedboard**. [S. l.], 2018. Disponivel em: <http://zedboard.org/product/zedboard>. Acesso em: 2 outubro 2018.

BRASIL. Ministério do Desenvolvimento Regional. Sistema Nacional de Informações sobre Saneamento - SNIS. **Diagnóstico dos serviços de água e esgotos**. Brasília, DF, 2016.

BENAYAS, Á. B. **Development of embedded linux applications using zedboard**. Madrid: [s. n.], 2013.

BRENNAN, M. J. et al. On the effects of soil properties on leak noise propagation in plastic water distribuition pipes. **Journal of Sound and Vibration**, London, v. 427, p. 120-133, 2018.

BRENNAN, M. J. et al. **Some Recent research results on the use of acoustic methods to detect water leaks in buried plastic water pipes**. [S. l.]: ISVR Technical Memorandum,. 2005.

CIRCUIT BASICS. **Basics of uart communication.** [S. l.], 2016. Disponivel em: <http://www.circuitbasics.com/basics-uart-communication/>. Acesso em: 4 out. 2018.

CROCKETT, L. H. et al. **The zynq book**. [S.l.]: Strathclyde Academic Media, 2014.

CROWN. **XLS DriveCore Series**. [S. l.], 2015. Disponivel em: <https://3e7777c294b9bcaa5486-

bc95634e606bab3d0a267a5a7901c44d.ssl.cf2.rackcdn.com/product_documents/documents/2563_1431033281/XLS_Install_DataSheet_Web_050615_original.pdf>. Acesso em: 28 out. 2018.

DIGILENT. **Getting Started with Digilent Pmod IPs**. [S. l., 201-?]. Disponivel em: <https://reference.digilentinc.com/learn/programmable-logic/tutorials/pmod-ips/start>. Acesso em: 19 jul. 2018.

DIGILENT. **Digilent Pmod Interface Specification**. [S. l.], 2011. Disponível em: <https://www.digilentinc.com/Pmods/Digilent-Pmod_%20Interface_Specification.pdf>. Acesso em: 3 out. 2018.

DIGILENT. **PmodAD1 Reference Manua**l. [S. l.], 2016. Disponível em: <https://reference.digilentinc.com/_media/pmod:pmod:pmodAD1_rm.pdf>. Acesso em: 5 out. 2018.

DIGILENT. **PmodGPS™ Reference Manual**. [S. l.], 2016. Disponivel em: <https://reference.digilentinc.com/_media/reference/pmod/pmodgps/pmodgps_rm.pdf>. Acesso em: 5 out. 2018.

DIGILENT. **PmodRTCC™ Reference Manual.** [S. l.], 2016.

DIGILENT. **Pmod NAV Reference Manual**. [S. l.], 2017. Disponivel em: <https://reference.digilentinc.com/_media/reference/pmod/pmodnav/pmod_nav_rm.pdf>. Acesso em: 5 out. 2018.

FUCHS, H. V.; RIEHLE, R. Ten years of experience with leak detection by acoustic signal analysis. **Applied Acoustics**, Kidlington, v. 33, p. 1-19, 1991.

GAO, Y. et al. A model of the correlation function of leak noise in buried plastic pipes. **Journal of Sound and Vibration,** London, v. 277, p. 133-148, 2004.

GLOBALTOP TECHNOLOGY INC. FGPMMOPA6H GPS Standalone Module Data Sheet. **G.Top**. [S. l.], 2012. Disponivel em: <https://cdn-shop.adafruit.com/datasheets/GlobalTop-FGPMMOPA6H-Datasheet-V0A.pdf>. Acesso em: 5 out. 2018.

GUO, H.; WANG, Z.; WANG, X. Transplant of Linux and Embedded System of Boot Loader and LED Driver. In: INTERNATIONAL CONFERENCE ON MACHINE VISION AND HUMAN-MACHINE INTERFACE, 2010, Kaifeng. **Proceedings** […] [S. l.: s. n.], 2010.

HUNAIDI, O. et al. Acoustic methods for locating leaks in municipal water pipe networks. In: INTERNATIONAL CONFERENCE ON WATER DEMAND MANAGEMENT. Dead Sea: **Proceedings** […] [S. l.: s. n.], 2004. p. 1-14.

KIDO, K. **Digital fourier analysis:** advanced techniques. [S.l.]: Springer-Verlag New York INC, 2014.

LDS. LDS V 101, V 102 and V 103 Shakers. **Midebien.** [S. l.], 2002. Disponivel em: <https://www.midebien.com/PDF/Bruel_Kjaer/Pruebas%20vibracion/hoja-t%C3%A9cnica-shaker-V101-V102-V201-V203-LDS.pdf>. Acesso em: 28 out. 2018.

LI-LI, L. et al. Design of Microcontroller Standard SPI interface. **Applied Mechanics and Materials**, Pfaffikon, v. 618, p. 563-568, 2014.

KIHONG, S.; HAMMOND, J. K. **Fundamentals of signal processing**. New York: John Wiley & Sons, 2008.

MACIEL, F. **O que é Linux Embarcado**. [S. l.]: Software Livre, 2014. Disponivel em: <http://softwarelivre.blog.br/2014/05/24/o-que-e-linux-embarcado/>. Acesso em: 20 ago. 2017.

MICROCHIP. **Battery-Backed I2C Real-Time Clock/Calendar with SRAM, EEPROM and Protected EEPROM**. [S.l.], 2018. Disponivel em: <http://ww1.microchip.com/downloads/en/devicedoc/20002266h.pdf>. Acesso em: 5 out. 2018.

MURATA. **W-LAN+Bluetooth Combo Module Data Sheet.** [S.l.], 2017. Disponivel em: <https://wireless.murata.com/RFM/data/lbee5kl1dx.pdf>. Acesso em: 13 out. 2018.

NXP SEMICONDUCTORS. **UM10204 I2C-bus specification and user manual**. 6. ed. [S.l.: s.n.], 2014.

PCB PIEZOTRONICS. **Model 333B Modal array, ceramic shear ICP®accel, 100 mV/g, 2 to 1k Hz, 3-pin conn Installation and Operating**. [S. l.: s. n.], 2002.

PCB PIEZOTRONICS. Model: 352C22_NC. **PCB**. Disponível em: <http://www.pcb.com/Products.aspx?m=352C22_NC>. Acesso em: 28 out. 2018.

SHARMA, S. D. et al. GNU/Linux shell access through a web-browser for an embedded Linux e-learning system. In: INTERNATIONAL CONFERENCE ON ELECTRONICS COMPUTER TECHNOLOGY, 3rd, 2011, Kanyakumari. **Proceedings** […] Kanyakumari: [s. n.], 2011.

SIEMENS. **LMS SCADAS XS provides maximum testing flexibility**. [S. l. : s. n.], 2018. Disponivel em: <https://www.plm.automation.siemens.com/en_us/Images/Siemens-PLM-LMS-SCADAS-XS-br_tcm1023-221179.pdf>. Acesso em: 28 out. 2018.

SMITH, S. W. **The scientist and engineer's guide to digital signal processing**. 2. ed. San Diego: California Technical, 1999.

ST LIFE.AUGMENTED. LSM9DS1. **ST life.augmented**. [S. l.], 2015. Disponível em: <https://www.st.com/resource/en/datasheet/lsm9ds1.pdf>. Acesso em: 5 out. 2018.

ST LIFE.AUGMENTED. LPS25HB. **ST life.augmented**. [S. l.], 2016. Disponível em: <https://www.st.com/resource/en/datasheet/lps25hb.pdf>. Acesso em: 5 out. 2018.

TAURION, C. **Software embarcado:** a nova onda da informática chips e software em todos os objetos. Rio de Janeiro: Brasport, 2005.

TRIVEDI, N.; PATEL, H.; CHAUHAN, D. Fundamental Structure of Linux Kernel based Device Driver and Implementation on Linux Host Machine. **International Journal of Applied Information Systems (IJAIS)**, New York, v. 10, n. 4, 2016.

XILINX. **PetaLinux Tools Documentation**: reference guide. [S.l.: s.n.], 2018.