CAPÍTULO 1 A INICIAÇÃO CIENTÍFICA NO INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS: UMA RETROSPECTIVA DE NOVE ANOS DE EXISTÊNCIA DO PROGRAMA PIBIC/CNPQ NO INPE

PROJETO DE IMPLEMENTAÇÃO EM FPGA DE EM MODULADOR PM COM APLICAÇÃO NO SISTEMA BRASILEIRO DE COLETA DE DADOS
(PIBIC/CNPq/INPE)

Francisco Assis de Sousa Júnior (UFRN, Bolsista PIBIC/CNPq)
E-mail: sjapodi@yahoo.com.br

Manoel Jozeane Mafra de carvalho (INPE/CRN, Orientador)
E-mail: manoel@crn2.inpe.br

Prof: Ivan Saraiva (Dimap/UFRN, Co-orientador)
E-mail: Ivan@dimap.ufrn.br

Julho de 2006

SUMÁRIO

FOLHA DE APROVAÇÃO

"Do not worry about your problems with mathematics; I assure you mine are far greater."

ALBERT EINSTEIN

A meus família,
que tanto acredita em mim.

AGRADECIMENTOS

Agradeço a todas pessoas que me ajudaram a vencer mais esta etapa da vida.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq, pelo auxilio financeiro de dois anos de bolsa de iniciação científica

À colega de trabalho tais como José Marcelo, Kurios Iuri, Yegor Melo, Rosciano e Eduardo, o Dudu que me receberam com muita simpatia.

Ao Instituto Nacional de Pesquisas Espaciais - INPE pela oportunidade de estudos e utilização de suas instalações.

Ao nosso orientador e coordenador Manoel Mafra por ter depositado em mim tanta confiança que por esta mim faço todo dia ser merecedor.

Aos meus amigos e simpatizantes da Residência Universitária da UFRN – Bravos pela resistência.
DeAssis Oliveira (O galego),
Francisco Thalles (Cizinha),
Higor Morais (Fortão),

A meus pais por sempre acreditarem na importância do estudo.

E aos demais de minha convivência, como Prof Luis Carlos, Sabino Neto, meu irmão Kaiser Jackson . A todos por refletirem junto comigo sobre os caminhos a ser trilhados nessa nossa longa jornada.

RESUMO

Este trabalho, iniciado em fevereiro de 2006, tem como objetivo o desenvolvimento de códigos que descrevam circuitos lógicos visando implementá-los em dispositivo re-configurável do tipo FPGA um demodulador do tipo PM para ser usado na Estação Multimissão de Natal (EMM-Natal) a ser incorporada a cadeia de Telemetria do Sistema Brasileiro de Coleta de Dados possibilitando esta estação a receber as mensagens transmitidas pelas PCDs através dos satélites SCDs. A aplicação inicial desse projeto foi dar auxílio a um outro projeto de Demodulação de Sinais usando o Costas Loop a apartir de montagem de módulo de tratamento PLL "Phase-Locked Loop" (ou Malha de Captura de Fase) com o objetivo de fazer o rastreamento para obtenção de dados de fase e frequência de sinais. O auxílio dado ao projeto é referente à Programação em VHDL "VHSIC Hardware Description Language" (Circuito Integrado de Altíssima Velocidade em Linguagem de Descrição de Hardware), linguagem usada para facilitar o design de circuitos digitais em FPGAs "Field Programable Gate Array" (Matriz de Portas Lógicas Programáveis no "terreno").Um FPGA é um dispositivo semicondutor que é largamente utilizado para o processamento de informações digitais.O projeto foi iniciado com o estudo deVHDL, logo após foi realizado o estudo da placa Cyclone II EP2C35 FPGA, fabricada pela Altera, juntamente com o programa de simulação e implementação Quartus II, fornecido pelo fabricante, adquirido pelo INPE. No momento, estamos concluindo dois programas que simulam um somador ripple carry e somador lookahead carry, em VHDL. O objetivo é adaptar-se a linguagem utilizada, a implementação na placa da Cyclone II e interação com a ferramenta Quartus II, à medida que se eleva a arquitetura de computadores no desenvolvimento dos programas, o permitira-nos ter plena capacidade no manuseio destas ferramentas para o desenvolvimento do demodulador.

PROJECT OF IMPLEMENTATION IN FPGA OF IN MODULATING P.M. WITH APPLICATION IN THE BRAZILIAN SYSTEM OF COLLECTION OF DATA

ABSTRACT

This work has as objective the development of codes that describe logical circuits aiming at to implement them in re-configurável device of type FPGA a demodulator of type p.m. to be used in the EMM-Christmas to be incorporated the chain of Telemetry of the Brazilian System of Collection of Data making possible this station to receive the messages transmitted for the PCDs through the SCDs satellites. The initial application of this project was to give to aid to one another project of Demodulation of Signals using the Coasts Loop to apartir of assembly of module of treatment PLL with the objective to make the tracking for attainment of phase data and frequency of signals. The aid given to the project is referring to the Programming in VHDL, used language to facilitate design of digital circuits in FPGAs. A FPGA wide is used for the processing of digital information.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE SIGLAS E ABREVIATURAS

FPGA	Field Programable Gate Array (Matriz de Portas Lógicas Programáveis no Terreno.)
PM	Phase Modulated (Fase modulada.)
EMM	Estação MultiMissão (Multimission Station)
SCD	Satélite de Coleta de Dados. (Satellite of Collection of Data.)
PLL	Phase-Locked Loop (Malha de Captura de Fase.)
VHDL	Very high Speed integrated circuit Hardware Description Language. (Circuito Integrado de Altíssima Velocidade em Linguagem de Descrição de Hardware.)
PCD	Plataforma de Coleta de Dados. (Platform of Collection of Data.)
CBERS2	China-Brazil Earth Resources Satellite. (Satélite Sino-Brasileiro de Recursos Terrestre.)
VHSIC	Very High Speed Integrated Circuit (Circuito Integrado de Altíssima Velocidade)

CAPÍTULO 2

INTRODUÇÃO

O sistema brasileiro de coleta de dados é constituído pela constelação de satélites SCD1, SCD2 e CBERS2 (segmento espacial), pelas diversas redes de plataformas de coleta de dados espalhadas pelo território nacional, pelas estações de recepção de Cuiabá, Alcântara e pelo centro de missão de coleta dados.

Neste sistema, os satélites funcionam como retransmissores de mensagens. Assim, a comunicação entre uma estação ambiental (PCDs) e as estações de recepção é estabelecida através dos satélites.

Os dados das PCDs retransmitidos pelos satélites e recebidos nas estações de Cuiabá ou Alcântara são enviados para o Centro de Missão de Coleta de Dados em Cachoeira Paulista para processamento, armazenamento e disseminação para os usuários. O envio desses dados é feito através da internet, em no máximo 30 minutos após a recepção. Dessa forma temos uma visão geral de como as informações recebidas e coletadas transitam no sistema brasileiro de coleta de dados.

2.1 Objetivos e métodos para sua concepção.

O objetivo de nosso trabalho é o desenvolvimento de códigos que descrevam circuitos lógicos visando implementá-los em dispositivo re-configurável do tipo FPGA. E a partir deste, desenvolver através de 2 equipes um modulador e um demodulador – este último o qual estou envolvido - do tipo PM para ser usado na Estação Multimissão de Natal (EMM-Natal) a ser incorporada a cadeia de Telemetria do Sistema Brasileiro de Coleta de Dados possibilitando esta estação a receber as mensagens transmitidas pelas PCDs através dos SCDs.

A aplicação inicial desse projeto foi dar auxílio a um outro projeto de Demodulação de Sinais usando o Costas Loop a apartir de montagem de módulo de tratamento PLLcom o objetivo de fazer o rastreamento para obtenção de dados de fase e frequência de sinais.

O FPGA é um dispositivo semicondutor que é largamente utilizado para o processamento de informações digitais. O FPGA será programado usandoa linguagem VHDL, usada para facilitar o design de circuitos digitais.

O desenvolvimento deste trabalho, tendo como coordenador Manuel Mafra e como orientador Ivan Saraiva do Dimap-UFRN que optou pela metodologia de desenvolvimento de projeto descrevendo circuitos usando VHDL seguindo o livro Descrição e Síntese de Circuitos Digitais do autor Roberto d´Amore no que se refere a VHDL e sistemas digitais princípios e aplicações do autores Tocci e Widmer com relação a teoria que envolve circuitos lógicos e seqüências.

2.2 Esboço Geral.

Este trabalho foi dividido em mais cinco capítulos, descritos a seguir:

Capítulo 2 - Descrevendo Circuitos em VHDL: Neste capítulo são apresentados algumas características de da linguagem VHDL e alguns programas feito com a mesma.

Capítulo 3 - Ferramenta Quartus II e Kit: Neste capítulo é apresentado a ferramenta escolhida para descrever os programas e algumas de suas propriedades.

Capítulo 4 - Programas Desenvolvidos: Neste capítulo são apresentados alguns dos programas descritos com o uso do Quartus II em VHDL

Apêndice – Códigos referente aos programas comentados no capítulo 2.

CAPÍTULO 3

Descrevendo Circuitos em VHDL

3.1 Um breve histórico sobre VHDL.

A linguagem VHDL deve o seu desenvolvimento à necessidade de uma ferramenta d projeto e documentação padrão para o projeto VHSIC do Departamento de Defesa do Estados Unidos (DARPA). Em 1981 o DARPA patrocinou um encontro de especialistas para discutir os métodos para descrição de circuitos. Mais a diante, a padronização da linguagem pelo IEEE – Institute of Eletrical and Eletronics Engeneer, teve como base a versão 7.2. No ano de 1987, após algumas revisões, surgiu o padrão IEEE 1076-1987.

A IEEE 1076-1993 surge em 1993 sem muitas diferenças significativas. E com o objetivo de adicionar facilidades à linguagem, foram proposto dois padrões , o IEEE 1164 e IEEE 1076.3.

3.2 Aspectos Gerais da Linguagem.

A linguagem VHDL suporta projetos com múltiplos níveis de hierarquia.

Com exceção de regiões específicas no código, todos os comandos são executados concorrentemente. Isto significa que a ordem na apresentação dos comandos é irrelevante para o comportamento da descrição. A ocorrência de um evento em um sinal leva à execução de todos os comandos sensíveis àquele sinal, da mesma forma que, em um circuito, a mudança de um valor em um determinado nó afeta todas as entradas ligadas a esse ponto do circuito.

CAPÍTULO 4

FERRAMENTA QUARTUS II E KIT

A placa de desenvolvimento Cyclone II EP2C35 é da empresa Altera.

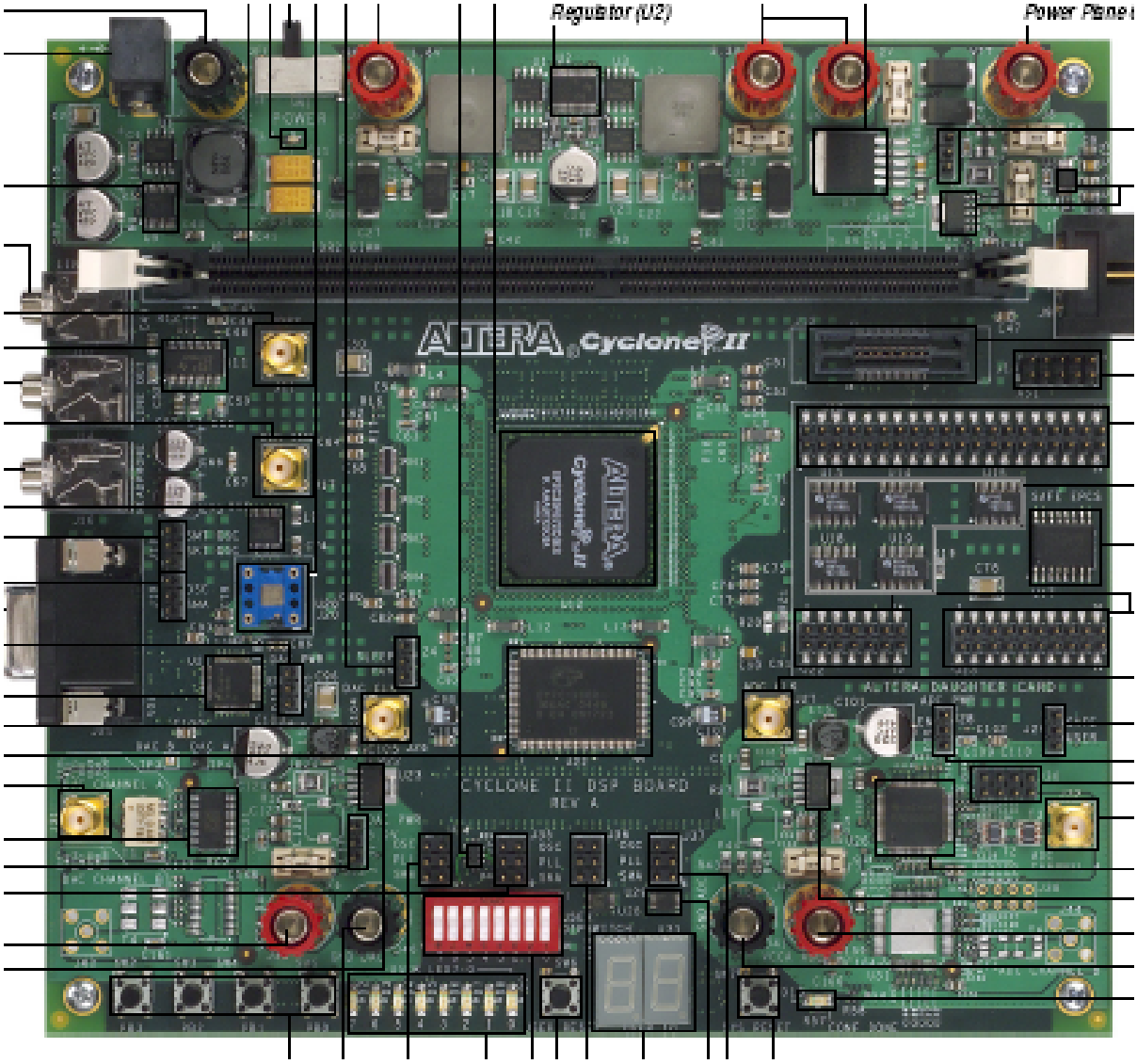

Figura 3.1 – Placa da Altera.

É nesta que futuramente o programa final descrito em VHDL será salvo.

Após o programa ser salvo neste dispositivo ele se comporta tal como as características do programa.

CAPÍTULO 5

PROGRAMAS DESENVOLVIDOS

5.1 Projeto Somador Ripple Carry

Onda de carry - Característica desse somador é o atraso na soma seguinte provocado pelo atraso no recebimento do carry para a computação dessa soma. Gerando, assim, uma "onda" de atraso que se acumula no último carry.

5.2 Projeto Somador Carry LookAhead

O Somador Carry Lookahead propõe computar todos os carries simultaneamente, evitando assim, o atraso provocado pelo acumulo da computação de cada carry individualmente, como é o caso do Somador Ripple Carry. Como as equações crescem muito rapidamente, opta-se pela construção de agrupamentos de 4 somadores de bit unitários.

REFERÊNCIAS BIBLIOGRÁFICAS

Tocci, Ronald J., Widmer, Neal S., Sistemas Digitais Princípios e aplicações, LTC, 2000.

D`Amore, Roberto., Descrição e Síntese de Circuitos Digitais, LTC, 2005.

APÊNDICE A

CÓDIGOS REFERENTE AOS PROGRAMAS COMENTADOS NO CAPÍTULO 4.

Somador Ripple Carry

```
--Ministério da ciência e tecnologia
--Instituto Nacional de Pesquisas Espaciais - INPE
--Programa institucional de bolsa de iniciação científica - INPE/PIBIC/CNPq
--Projeto: Implementação em FPGA de um modulador PM com aplicação
--        no Sistema Brasileiro de Coleta de Dados.

--Orientador Pelo INPE: Manoel Jozeane Mafra de Carvalho. E-mail: manoel@crn2.inpe.br
--Orientador Pelo Dimap - UFRN: Ivan Saraiva. E-mail: ivan@dimap.ufrn.br
--Bolsista: Francisco Assis de Sousa Júnior. E-mail: sjapodi@yahoo.com.br,
sjapodi@crn.inpe.br

--                          Projeto Somador Ripple Carry

entity somador_propagacao_carry is

 port(                              hablt : in bit;   --Declaração das variáveis
                      a, b : in bit_vector (3 downto 0);   -- utilizadas no programa.
              Cinout, Cinout1, Cinout2, Cinout3 : inout bit;
                     seg_H : out bit_vector (7 downto 0);
                     seg_L : out bit_vector (7 downto 0);
                      q : buffer bit_vector (3 downto 0)
           );
end somador_propagacao_carry;

architecture bit_type of somador_propagacao_carry is
                              --abcdefgp  ('p', ponto do display 7 seguimentos)
constant ds0 : bit_vector (7 downto 0) := "00000011"; --codigo numero 0
constant ds1 : bit_vector (7 downto 0) := "10011111"; --codigo numero 1
constant ds2 : bit_vector (7 downto 0) := "00100101"; --codigo numero 2
constant ds3 : bit_vector (7 downto 0) := "00001101"; --codigo numero 3
constant ds4 : bit_vector (7 downto 0) := "10011001"; --codigo numero 4
constant ds5 : bit_vector (7 downto 0) := "01001001"; --codigo numero 5
constant ds6 : bit_vector (7 downto 0) := "01000001"; --codigo numero 6
constant ds7 : bit_vector (7 downto 0) := "00011111"; --codigo numero 7
constant ds8 : bit_vector (7 downto 0) := "00000001"; --codigo numero 8
constant ds9 : bit_vector (7 downto 0) := "00001001"; --codigo numero 9
constant dsn : bit_vector (7 downto 0) := "11111111"; --apagado

signal S : bit_vector (3 downto 0);

begin
```

```
Somador_bit_0: block          --bloco 0
 begin
  S(0) <= (a(0) xor b(0));
  Cinout <= (a(0) and b(0));
 end block Somador_bit_0;

Somador_bit_1: block          --bloco 1
 begin
  S(1) <= (a(1) xor b(1)) xor Cinout;
  Cinout1 <= (Cinout and (a(1) xor b(1))) or (a(1) and b(1));
 end block Somador_bit_1;

Somador_bit_2: block          --bloco 2
 begin
  S(2) <= (a(2) xor b(2)) xor Cinout1;
  Cinout2 <= (Cinout1 and (a(2) xor b(2))) or (a(2) and b(2));
 end block Somador_bit_2;

Somador_bit_3: block          --bloco 3
 begin
  S(3) <= (a(3) xor b(3)) xor Cinout2;
  Cinout3 <= (Cinout2 and (a(3) xor b(3))) or (a(3) and b(3));
 end block Somador_bit_3;

Flip_Flops: block --Flip-flop usado para set o resultado da soma nos despositivos de saídas
do FPGA.
 begin
q(0) <= S(0) when hablt = '0' else
     q(0);

q(1) <= S(1) when hablt = '0' else
     q(1);

q(2) <= S(2) when hablt = '0' else
     q(2);

q(3) <= S(3) when hablt = '0' else
     q(3);
 end block Flip_Flops;

Display_M: block

 begin --Referente ao display mais significativo
seg_H <= dsn when q="0000" else
        dsn when q="0001" else
        dsn when q="0010" else
        dsn when q="0011" else
```

```
        dsn when q="0100" else
        dsn when q="0101" else
        dsn when q="0110" else
        dsn when q="0111" else
        dsn when q="1000" else
        dsn when q="1001" else
        ds1;
 end block Display_M;

Display_L: block

 begin --Referente ao display menos significativo
seg_L <= dsn when q="0000" else
        ds1 when q="0001" or q="1011" else
        ds2 when q="1100" or q="0010" else
        ds3 when q="1101" or q="0011" else
        ds4 when q="1110" or q="0100" else
        ds5 when q="1111" or q="0101" else
        ds6 when q="0110" else
        ds7 when q="0111" else
        ds8 when q="1000" else
        ds9;
 end block Display_L;

end bit_type;
```

Somedor Carry LookAhead

```
--Ministério da ciência e tecnologia
--Instituto Nacional de Pesquisas Espaciais - INPE
--Programa institucional de bolsa de iniciação científica - INPE/PIBIC/CNPq
--Projeto: Implementação em FPGA de um modulador PM com aplicação
--         no Sistema Brasileiro de Coleta de Dados.

--Orientador Pelo INPE: Manoel Jozeane Mafra de Carvalho. E-mail: manoel@crn2.inpe.br
--Orientador Pelo Dimap - UFRN: Ivan Saraiva. E-mail: ivan@dimap.ufrn.br
--Bolsista: Francisco Assis de Sousa Júnior. E-mail: sjapodi@yahoo.com.br,
sjapodi@crn.inpe.br

--                          O Somador Carry Lookahead

entity Somador_carry_lookahead is
 port (                    a, b : in bit_vector (3 downto 0); --Declaração de variáveis
       Cinout, Cinout0, Cinout1, Cinout2, Cinout3 : inout bit; --utilizadas no programa.
                           S: out bit_vector (3 downto 0);
                      p, g : inout bit_vector (3 downto 0)
     );
end Somador_carry_lookahead;
```

```
architecture teste of Somador_carry_lookahead is

begin

p(0)<= (a(0) and b(0));
p(1)<= (a(1) and b(1));
p(2)<= (a(2) and b(2));
p(3)<= (a(3) and b(3));

g(0)<=  (a(0) or b(0));
g(1)<=  (a(1) or b(1));
g(2)<=  (a(2) or b(2));
g(3)<=  (a(3) or b(3));

Somador_bit_0: block          --bloco 0
 begin
  S(0) <= (a(0) xor b(0));
  Cinout0 <= g(0) or (p(0) and Cinout);
 end block Somador_bit_0;

Somador_bit_1: block          --bloco 1
 begin
  S(1) <= (a(1) xor b(1)) xor Cinout0;
  Cinout1 <= g(1) or (p(1) and (g(0) or (p(0) and Cinout)));
 end block Somador_bit_1;

Somador_bit_2: block          --bloco 2
 begin
 S(2) <= (a(2) xor b(2)) xor Cinout1;
  Cinout2 <= g(2) or (p(2) and (g(1) or (p(1) and (g(0) or (p(0) and Cinout)))));
 end block Somador_bit_2;

Somador_bit_3: block          --bloco 3
 begin
  S(3) <= (a(3) xor b(3)) xor Cinout2;
  Cinout3 <=   g(3) or (p(3) and (g(2) or (p(2) and (g(1) or (p(1) and (g(0) or (p(0) and
Cinout)))))));
 end block Somador_bit_3;

end teste;
```