



MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2015/05.28.17.26 -RPQ

MODELAGEM MATEMÁTICA EM MICROELETRÔNICA RECONFIGURÁVEL: ESTUDO DE CASO SOBRE MODULADORES BPSK

Francisco de Assis Tavares Ferreira da Silva
Magno Prudêncio de Almeida Filho
Nicolas de Araújo Moreira
Clauson Sales do Nascimento Rios
Paulo Daving Lima de Oliveira
Paulo Jarbas Camurça
Antonio Macilio Pereira de Lucena

Relatório Técnico-Científico ge-
rado dentro do projeto de pes-
quisa: Prototipação de Circuitos
e Equipamentos para Comunica-
ções Espaciais via Síntese de For-
malismos Matemáticos em Micro-
eletrônica Reconfigurável (Edital
MCT/CNPq/AEB n° 33/2010)

URL do documento original:

<<http://urlib.net/8JMKD3MGP3W34P/3JJ6M6E>>

INPE
São José dos Campos
2015

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE

Gabinete do Diretor (GB)

Serviço de Informação e Documentação (SID)

Caixa Postal 515 - CEP 12.245-970

São José dos Campos - SP - Brasil

Tel.:(012) 3208-6923/6921

Fax: (012) 3208-6919

E-mail: pubtc@sid.inpe.br

COMISSÃO DO CONSELHO DE EDITORAÇÃO E PRESERVAÇÃO DA PRODUÇÃO INTELECTUAL DO INPE (DE/DIR-544):

Presidente:

Marciana Leite Ribeiro - Serviço de Informação e Documentação (SID)

Membros:

Dr. Gerald Jean Francis Banon - Coordenação Observação da Terra (OBT)

Dr. Amauri Silva Montes - Coordenação Engenharia e Tecnologia Espaciais (ETE)

Dr. André de Castro Milone - Coordenação Ciências Espaciais e Atmosféricas
(CEA)

Dr. Joaquim José Barroso de Castro - Centro de Tecnologias Espaciais (CTE)

Dr. Manoel Alonso Gan - Centro de Previsão de Tempo e Estudos Climáticos
(CPT)

Dr^a Maria do Carmo de Andrade Nono - Conselho de Pós-Graduação

Dr. Plínio Carlos Alvalá - Centro de Ciência do Sistema Terrestre (CST)

BIBLIOTECA DIGITAL:

Dr. Gerald Jean Francis Banon - Coordenação de Observação da Terra (OBT)

Clayton Martins Pereira - Serviço de Informação e Documentação (SID)

REVISÃO E NORMALIZAÇÃO DOCUMENTÁRIA:

Simone Angélica Del Duca Barbedo - Serviço de Informação e Documentação
(SID)

Yolanda Ribeiro da Silva Souza - Serviço de Informação e Documentação (SID)

EDITORAÇÃO ELETRÔNICA:

Marcelo de Castro Pazos - Serviço de Informação e Documentação (SID)

André Luis Dias Fernandes - Serviço de Informação e Documentação (SID)



MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2015/05.28.17.26 -RPQ

MODELAGEM MATEMÁTICA EM MICROELETRÔNICA RECONFIGURÁVEL: ESTUDO DE CASO SOBRE MODULADORES BPSK

Francisco de Assis Tavares Ferreira da Silva
Magno Prudêncio de Almeida Filho
Nicolas de Araújo Moreira
Clauson Sales do Nascimento Rios
Paulo Daving Lima de Oliveira
Paulo Jarbas Camurça
Antonio Macilio Pereira de Lucena

Relatório Técnico-Científico ge-
rado dentro do projeto de pes-
quisa: Prototipação de Circuitos
e Equipamentos para Comunica-
ções Espaciais via Síntese de For-
malismos Matemáticos em Micro-
eletrônica Reconfigurável (Edital
MCT/CNPq/AEB nº 33/2010)

URL do documento original:

<<http://urlib.net/8JMKD3MGP3W34P/3JJ6M6E>>

INPE
São José dos Campos
2015



Esta obra foi licenciada sob uma Licença Creative Commons Atribuição-NãoComercial 3.0 Não Adaptada.

This work is licensed under a Creative Commons Attribution-NonCommercial 3.0 Unported License.

AGRADECIMENTOS

Agradecemos aos fomentadores do INPE, AEB e CNPq, pela iniciativa da promoção e gestão do Edital 33/2010, o qual tem contribuído no exercício da formação de recursos humanos, principalmente em áreas estratégicas, e em novas conquistas técnico-científicas para o país.

Agradecemos também aos colegas do INPE e das instituições parceiras que de alguma forma colaboraram para o êxito do projeto.

RESUMO

Este relatório apresenta uma metodologia para modelagem matemática de subsistemas eletrônicos a serem simulados, em ambiente computacional, e emulados, em circuitos integrados baseados em microeletrônica reconfigurável tipo *FPGA* (Field Programmable Gate Array) genérico. O processo de desenvolvimento é iniciado a partir das expressões matemáticas, as quais definem o modelo de simulação computacional. Em seguida, é utilizado um compilador o qual converte, automaticamente, o modelo matemático, simulado, em códigos tipo *HDL* (Hardware Description Language), os quais podem ser aproveitados por *FPGAs* genéricos. Para exemplificar o processo de desenvolvimento, foram considerados vários aplicativos necessários às etapas de representação e simulação de modelos matemáticos, considerando um estudo de caso sobre moduladores *BPSK* (Binary Phase Shift Keying), e as respectivas definições para emulação em *hardware*.

Palavras-chave: *FPGA*, Microeletrônica Reconfigurável, Subsistemas de Comunicação, Aplicações Espaciais, Moduladores *BPSK*.

MATHEMATICAL MODELING IN RECONFIGURABLE MICROELECTRONICS: CASE STUDY REGARDING BPSK MODULATORS

ABSTRACT

This report presents a mathematical modeling methodology applied to electronic subsystems development to be simulated in a computer environment and emulated in integrated circuits based on reconfigurable microelectronics as generic FPGA (Field Programmable Gate Array). The development process starts from the mathematical expressions, which define the computer simulation model. In a second stage this simulated model is feed to a compiler which automatically converts the mathematical model simulated to HDL codes (Hardware Description Language), which can be applied to generic FPGAs. To illustrate the development process, the necessary software is presented as well the steps for the representation, simulation and hardware emulation of the mathematical models. For that, a case study on BPSK modulators (Binary Phase Shift Keying) is considered to be applied on reconfigurable modulation experiments.

Keywords: FPGA, Reconfigurable Microelectronic, Communication Subsystems, Spatial Applications, BPSK Modulators

LISTA DE FIGURAS

| | <u>Pág.</u> |
|--|-------------|
| Figura 3.1: Interface gráfica do software <i>Mathworks Simulink</i> | 10 |
| Figura 3.2: Interface gráfica do software <i>Xilinx ISE Project Navigator</i> | 13 |
| Figura 3.3: Interface gráfica do software <i>Xilinx PlanAhead</i> | 14 |
| Figura 4.1: Gerador BPSK..... | 17 |
| Figura 4.2: Caixa de diálogo do gerador BPSK..... | 17 |
| Figura 4.3: Diagrama de blocos do gerador BPSK..... | 18 |
| Figura 4.4: Implementação do bloco geração de dados aleatórios. | 19 |
| Figura 4.5: Sobreamostragem e filtragem do sinal..... | 20 |
| Figura 4.6: Implementação do atraso de propagação..... | 20 |
| Figura 4.7: Implementação do bloco geração da onda portadora. | 21 |
| Figura 4.8: Implementação do bloco de ruído. | 22 |
| Figura 4.9: Esquema de geração do sinal BPSK. | 22 |
| Figura 4.10: Espectro do sinal BPSK centrado em 70 MHz. | 23 |
| Figura 4.11: Gerador BPSK $\pm\pi/3$ | 25 |
| Figura 4.12: Caixa de diálogo do gerador BPSK $\pm\pi/3$ | 25 |
| Figura 4.13: Diagrama de blocos do gerador BPSK $\pm\pi/3$ | 26 |
| Figura 4.14: Implementação do bloco geração de dados de PCD. | 27 |
| Figura 4.15: Implementação do bloco geração da onda portadora. | 28 |
| Figura 4.16: Esquema de geração do sinal BPSK $\pm\pi/3$ | 29 |
| Figura 4.17: Espectro do sinal BPSK $\pm\pi/3$ em torno de 401,65 MHz. | 29 |
| Figura C.1: <i>Xilinx Blockset</i> | 49 |
| Figura C.2: Projeto de controle de LCD com <i>Black Box</i> | 50 |

| | |
|--|----|
| Figura C.3: Seleção de arquivo. | 51 |
| Figura C.4: Adicionando arquivos com dependência a uma <i>Black Box</i> | 52 |

LISTA DE SIGLAS E ABREVIATURAS

| | |
|--------|---|
| ASIC | <i>Application Specific Integrated Circuit</i> |
| AWGN | <i>Additive white Gaussian noise</i> |
| BPSK | <i>Binary Phase-Shift Keying</i> |
| CI | Circuito Integrado |
| CLR | Células Lógicas Reconfiguráveis |
| COTS | <i>Commercial Off-The-Shelf</i> |
| CPLD | <i>Complex Programmable Logic Device</i> |
| CRN | Centro Regional do Nordeste |
| EEPROM | <i>Electrically Erasable Read-Only Memory</i> |
| EPROM | <i>Erasable Read-Only Memory</i> |
| FPGA | <i>Field Programmable Gate Array</i> |
| GPS | <i>Global Positioning System</i> |
| HDL | <i>Hardware Description Language</i> |
| INPE | Instituto Nacional de Pesquisas Espaciais |
| LUT | <i>Look Up Table</i> |
| MLC | Módulos Lógicos Configuráveis |
| PCD | Plataforma de Coleta de Dados |
| PLA | <i>Programmable Logic Array</i> |
| PLD | <i>Programmable Logic Device</i> |
| PN | <i>Pseudorandom Noise</i> |
| RTL | <i>Register Transfer Level</i> |
| SBCDA | Sistema Brasileiro de Coleta de Dados Ambientais |
| SINDA | Sistema Integrado de Dados Ambientais |
| SMMH | Síntese de Modelagem Matemática em Hardware |
| SRAM | <i>Static Random Access Memory</i> |
| TMR | <i>Triple Modular Redundancy</i> |
| VHDL | <i>Very High Speed Integrated Circuit Hardware Description Language</i> |
| VLSI | <i>Very-Large-Scale Integration</i> |

SUMÁRIO

| | <u>Pág.</u> |
|-----|---|
| 1 | INTRODUÇÃO 1 |
| 2 | DESENVOLVIMENTO DE CIRCUITOS E EQUIPAMENTOS VIA MICROELETRÔNICA RECONFIGURÁVEL 3 |
| 3 | APLICATIVOS DE DESENVOLVIMENTO 9 |
| 4 | DESENVOLVIMENTO DE GERADORES BPSK PARA FPGA GENÉRICO 15 |
| 4.1 | Modelo simplificado de um gerador BPSK 16 |
| 4.2 | Modelo simplificado de um gerador BPSK com índice de modulação $\pi/3$ 23 |
| 5 | DEFINIÇÃO DE BLACK BOX PARA INTEGRAÇÃO DE COMPONENTES DE TERCEIROS A EXTENSÃO DA BIBLIOTECA <i>HDL CODER</i> 31 |
| 6 | DIFICULDADES ENCONTRADAS 33 |
| 7 | ANÁLISE DOS RESULTADOS E PERSPECTIVAS 35 |
| | REFERÊNCIAS BIBLIOGRÁFICAS 39 |
| | APÊNDICE 45 |

1 INTRODUÇÃO

A unidade do INPE em Eusébio - CE (EUS/CRN) é parte do Centro Regional do Nordeste e pretende contribuir de forma significativa para o cumprimento da missão do CRN, principalmente no que concernem as ações já em desenvolvimento nas áreas de processamento de sinais e telecomunicações espaciais (1-7), por meio do estudo e desenvolvimento de sistemas, subsistemas, algoritmos ou firmwares, dentre outros dispositivos aplicados em telecomunicações dos segmentos solo e/ou de bordo. Além disso, pretende também contribuir na formação de recursos humanos em processamento digital de sinais, considerando aplicações espaciais e áreas correlatas, bem como gerar conhecimento e promover a produção científica e tecnológica nessas áreas.

Dentro desse contexto surgiu o projeto “Prototipação de Circuitos e Equipamentos para Comunicações Espaciais via Síntese de Formalismos Matemáticos em Microeletrônica Reconfigurável”.

O corrente projeto tem por objetivo a formação de recursos humanos e o desenvolvimento de métodos a serem aplicados na modelagem matemática e/ou construção de algoritmos visando prototipação de circuitos, subsistemas e equipamentos definidos por software, considerando aplicações de processamento digital de sinais e comunicação digital, baseados em tecnologia de microeletrônica reconfigurável.

Dentre as várias tecnologias de microeletrônica reconfigurável, este projeto também tem contribuído em provas de conceitos das técnicas de geração ou síntese semiautomática, e automática, de dispositivos eletrônicos tipo FPGA (do inglês *Field Programmable Gate Array*). Para uma visão das contribuições do projeto, apresentam-se nesse relatório os estudos, provas de conceitos e resultados, considerando-se as facilidades de desenvolvimento da tecnologia de microeletrônica reconfigurável, em direção ao aperfeiçoamento das

metodologias de Síntese de Modelagem Matemática em Hardware-SMMH, inspirada em (8-11), principalmente quando considerando aplicações espaciais.

Para situar a importância das atividades e resultados obtidos, na seção 2 é apresentada uma visão geral e cronológica do desenvolvimento da microeletrônica reconfigurável em direção ao desenvolvimento da área de comunicação digital, visando aplicações espaciais. Na seção 3, é apresentado um resumo dos métodos e aplicativos utilizados no estudo de caso proposto.

O desenvolvimento da modelagem, simulações, emulações e apresentação dos espectros gerados são abordados na seção 4. Na seção 5 é apresentado um método para integração de componentes de terceiros, como extensão da biblioteca de módulos do *Mathworks Simulink®*, para o aproveitamento de códigos HDL na geração de *bitstreams*. Na seção 6 é apresentado um resumo das dificuldades e soluções encontradas. Na seção 7 é apresentado um resumo dos resultados, considerando as vantagens e desvantagens da metodologia empregada, e perspectivas de desenvolvimento visando novas aplicações.

2 DESENVOLVIMENTO DE CIRCUITOS E EQUIPAMENTOS VIA MICROELETRÔNICA RECONFIGURÁVEL

A engenharia espacial é uma das áreas de maior importância para o domínio tecnológico e desenvolvimento da maioria dos equipamentos responsáveis pelas atividades espaciais. Por sua vez, atualmente a engenharia eletrônica e a engenharia da computação formam um dos principais vetores de desenvolvimento na área de telecomunicações espaciais. Por outro lado, até o início da década de 1990 a eletrônica analógica e a eletrônica digital, desenvolvidas pela adoção de componentes discretos e circuitos integrados dedicados ou combinados em tecnologia ASIC, predominavam na maioria dos projetos da engenharia espacial.

No entanto, ainda na década de 1970 e durante os anos 80's, foram observados os primeiros grandes avanços na área de microeletrônica (12-15). Estes avanços também contribuíram sobremaneira na geração de componentes definidos via programação de *firmwares* (códigos para configuração de *hardware*), os quais, por sua vez, deram origem às tecnologias mais flexíveis de projetos de Dispositivos Lógicos Programáveis (do inglês PLDs), conforme pode ser observado em (15,16), em conjunto com o desenvolvido em (17).

Por sua vez, a tecnologia de áreas de portas lógicas programáveis (do inglês-PLAs) e as matrizes de portas de campo programáveis, ainda que rudimentarmente, deram origem ao desenvolvimento de projetos de circuitos eletrônicos definidos por software (18,19), ou ainda projetos baseados em *firmwares* para geração de circuitos passíveis de definição em microeletrônica pós-encapsulada, principalmente, quando fazendo uso da tecnologia de FPGA. Para ilustração do potencial da tecnologia de FPGA e da importância das respectivas metodologias de desenvolvimento, na área de comunicação espacial, é apresentada a seguir uma introdução sobre estes dispositivos e

respectivos procedimentos de desenvolvimento, deste tipo de tecnologia, quando aplicados à área de telecomunicações.

Ainda na década de 1990, os FPGAs foram substancialmente melhorados em termos de sofisticação de arquiteturas em VLSI contemplando novos ambientes de desenvolvimento, maior densidade funcional em silício e aumento no volume de produção. Nesta década, os circuitos integrados (CIs) de FPGAs obtiveram destaque quando aplicados em projetos de telecomunicações, como, por exemplo, pode ser observado em (19).

No final dos anos 2000 e início da década de 2010, foi desenvolvido e introduzido no mercado eletrônico o conceito de plataforma de processamento baseada em microcontroladores/microprocessadores, como por exemplo, Picoblaze™, Microblaze™ e NIOS® II (20-23), ou ainda tipo ARM® (24) implementados em FPGAs. Isto permitiu aos projetistas de sistemas digitais, principalmente desenvolvedores de sistemas embarcados, o desenvolvimento de aplicações considerando a combinação de processamento sequencial, via emulação de microprocessadores, e circuitos paralelos, em um mesmo CI de FPGA (20,24).

Por outro lado, a utilização da combinação de processamento sequencial e paralelo é útil para atender às demandas crescentes nas implementações de funções mais complexas, considerando ações de respostas em tempo real. Houve, portanto, a união das já conhecidas plataformas de microprocessadores, contemplando as vantagens de sistemas eletrônicos de áreas programáveis definidos por *software*, numa única plataforma FPGA, em lugar do uso de ciclos tradicionais de projetos associados à tecnologia de produção apenas via ASICs (20,24).

Em (25), pode-se observar que o dinamismo da tecnologia de FPGA tem provocado também grandes mudanças no desenvolvimento de projetos, quando envolvendo tecnologia de microeletrônica reconfigurável, a qual, por outro lado, impõe novas metodologias que possam acompanhar a migração

e/ou combinação entre paradigmas ASIC/FPGA. Principalmente quando considerados os seguintes aspectos:

- Os custos de circuitos integrados com alta densidade aumentam sensivelmente.
- A complexidade ASIC encarece e alonga o tempo de desenvolvimento.
- A demanda de recursos humanos de P&D está aumentando e a oferta de pessoal qualificado está diminuindo
- O custo de manutenção, ou atualização, pela adoção apenas da tecnologia ASIC onera sensivelmente o ciclo de vida dos dispositivos eletrônicos.
- As restrições financeiras dos países em desenvolvimento demanda tecnologia de baixo custo.

Essas tendências fazem da tecnologia de FPGA uma alternativa atrativa, para um maior número de aplicações, usando-se apenas chips de FPGAs ou em combinação com ASICs, do que têm sido historicamente observados quando considerando apenas a tecnologia ASIC.

Visão geral da tecnologia de FPGAs:

A maior parte da arquitetura de um chip ou CI de FPGA consiste de componentes lógicos programáveis conhecidos como blocos lógicos (também denominados MLC ou CLR), organizados através de uma hierarquia de interconexões configuráveis ou reconfiguráveis, ao nível da microeletrônica, as quais permitem que os módulos possam ser flexivelmente conectados e/ou reconectados entre si, através de um arquivo de definição de hardware conhecido como *bitstream*. Módulos lógicos podem ser configurados para executar funções de combinações complexas ou portas lógicas simples, como, por exemplo: *AND*, *NAND* e *XOR*. Estes blocos lógicos incluem algumas células lógicas chamadas *slices*. A célula típica consiste de LUTs, Somadores Completos (SC), e *Flip-Flops* (FF).

Para definir o comportamento dos FPGAs, o projetista pode definir a configuração do circuito, via microeletrônica configurável ou reconfigurável, através de algoritmos, ou através de esquemas ou modelos de simulação, os quais, em geral, são convertidos para um script usando *Hardware Description Language* – HDL (26), e em seguida sintetizado ou compilado para *firmware* através do arquivo *bitstream* (códigos de configuração do FPGA definidos por cada fabricante do chip alvo em específico). Ou seja, o projeto é convencionalmente compilado/sintetizado pela conversão de algoritmos ou esquemas em um *firmware*, de definição de microeletrônica, em formato *bitstream*.

Esta tecnologia permitiu que milhões de transistores pudessem ser reconfigurados através de HDL em um único CI dentro de um curto espaço de tempo. HDL pode ser utilizada em várias formas: (i) A partir de circuitos lógicos via códigos sintetizados (ii) Compilação de circuitos gerados através de códigos comportamentais (*Behavioral Modelling*), e (iii) Representação de circuitos sintetizados a partir de códigos estruturais (*Structural Modelling*). Desse modo, os circuitos e/ou algoritmos devem ser projetados e compilados para códigos HDL, visando-se a implementação ou configuração do *hardware* em FPGA, via *bitstream* específico de cada fabricante.

As linguagens de descrição de *hardware* (HDL) mais comuns são Verilog® e VHDL® (26,27). Ambas definem projetos no interior de CIs FPGAs, CPLDs ou ASICs. Em geral, a liberdade de edição de um projeto nessas linguagens pode gerar menor volume de código, e/ou menor consumo de recursos do chip quando comparada a uma geração de *firmware* a partir de esquemáticos.

No entanto, quando adotada a metodologia de esquemas, em geral, verifica-se as seguintes características: (i) É uma abordagem intuitiva, ou seja, não exige grande experiência em FPGA; (ii) Por possuir um maior nível de abstração, pode ser utilizada para definição de requisitos específicos de circuitos eletrônicos, até mesmo por usuários com pouca experiência em *hardware*; (iii)

Permite a definição funcional e a integração rápida de subsistemas, de maneira que os projetos podem ser concluídos em curto prazo.

Quando da necessidade de prototipação, principalmente voltada à demanda de novas provas de conceitos, este tipo de tecnologia, por ser flexível, no sentido de reconfiguração, e ao mesmo tempo robusta, no sentido de permitir geração de *hardware* a partir da modelagem matemática, tem mostrado também altos índices de produtividade. Pois permite gerar vários ciclos de depuração, via simulação e co-simulação e/ou emulação, em poucas horas, quando antes demandavam semanas ou meses para instrumentar falhas em um sistema já implementado em placas de *wire-wrap*, ou de circuitos impressos na forma final.

Além disso, pode-se constatar a sensível redução de custos de desenvolvimento, pois além da prova de conceito, ou prototipação, ser realizada diretamente no *hardware* próximo a forma final, *bread board*, ou *hardware* definitivo, em muitos casos, evitam-se os retrabalhos, no *hardware* de circuito impresso, em caso de detecção de efeitos idiossincráticos, verificados apenas quando em tempo de integração com outros subsistemas.

No entanto, embora várias instituições já estejam adotando a tecnologia de microeletrônica reconfigurável em muitos projetos de engenharia, inclusive na área espacial, por ser uma tecnologia recente, em relação à área espacial, vários experimentos e provas de conceitos precisam ser realizados para consolidar este tipo de abordagem, principalmente quando aplicada a área de telecomunicações (28). Quando aplicada a área de telecomunicações espaciais (29-34), isto é ratificado pela dificuldade de aquisição de componentes “endurecidos” (do inglês *radiation hardening*) ou tolerantes a radiação, o que impõe um contorno via COTS pela adoção de outras estratégias, como por exemplo, via redundâncias e/ou invólucros protetores, dentre outras estratégias, objetivando-se o domínio e o emprego desta tecnologia para dispositivos de comunicação, no âmbito da engenharia espacial.

Metodologia empregada:

O presente projeto deu início aos vários procedimentos para realização de provas de conceitos dessas novas metodologias, considerando a definição de componentes e circuitos, definidos por *software*, que possam auxiliar o desenvolvimento de dispositivos e/ou subsistemas a partir de modelagem matemática adequada a simulação computacional em direção à implementação em *hardware*. Nesse sentido, o presente trabalho também pretende realizar provas de conceito visando o domínio das técnicas de prototipação de instrumentos e equipamentos de comunicação digital, para fins espaciais, considerando o aprimoramento da metodologia de Síntese de Modelagem Matemática em Hardware-SMMH (8-11), baseada em microeletrônica reconfigurável.

Dentre os vários desafios, logrou-se êxito na implantação de ambiente computacional para o desenvolvimento e testes de componentes e subsistemas de processamento digital de sinais, em direção a uma biblioteca de módulos funcionais de simulação (blocos em *Simulink*®), para geração de *firmwares* de componentes e/ou subsistemas, a serem aplicados em processamento de sinais. Deve-se destacar que os módulos em desenvolvimento são provenientes diretamente de modelos matemáticos encontrados na literatura de telecomunicações espaciais (29-34), incluindo as referências mais específicas, como as apresentadas nos seguintes trabalhos (35-37).

Estes modelos são inerentes aos elementos necessários aos projetos envolvendo modulação e demodulação, como subsistemas de telecomunicações, ou equipamentos, como geradores de sinais, ou ainda instrumentos, como por exemplo, medidores de taxas de erros, a serem usados quando em tempo de projetos de moduladores, demoduladores, modems, transceptores e/ou transponders, dentre outros dispositivos correlatos, usados na área espacial.

3 APLICATIVOS DE DESENVOLVIMENTO

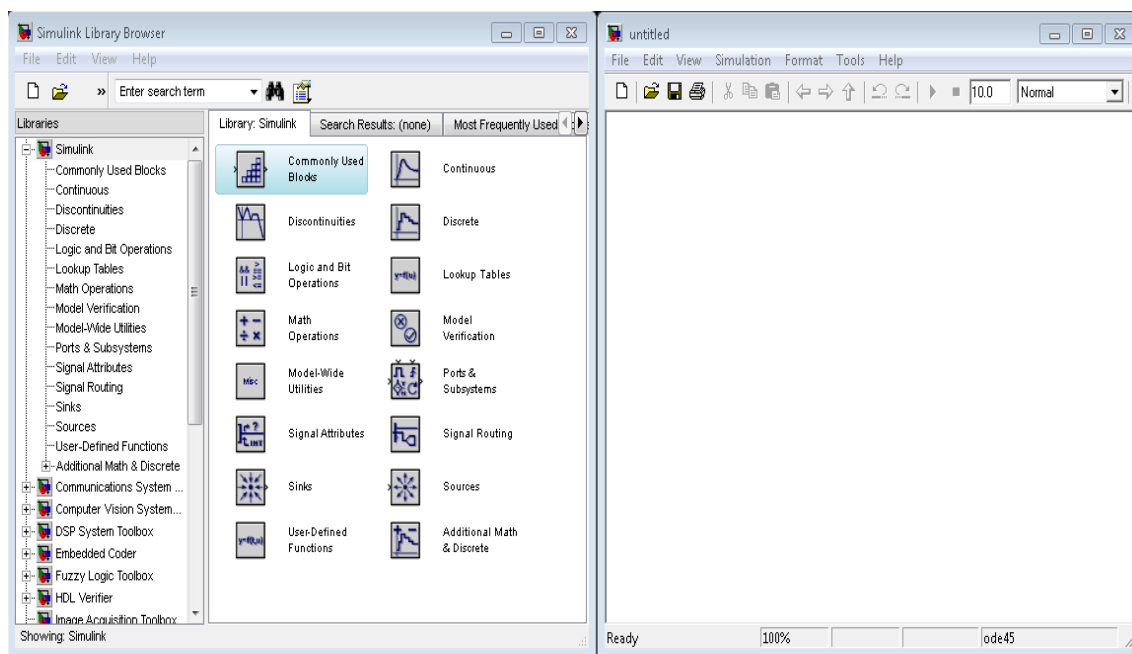
Ao longo do desenvolvimento do projeto, foram analisadas diversas ferramentas para modelagem, desenvolvimento, síntese e testes de circuitos digitais usando microeletrônica definida por *software*. No entanto, um primeiro estudo foi efetuado em ambientes e plataformas já disponíveis, via outros projetos, inerentes a infraestrutura da Unidade do INPE/CRN-Eusébio. Portanto, segue uma descrição geral dos aplicativos utilizados, suas funções, e uma descrição breve dos seus principais recursos. Neste primeiro estudo, os aplicativos utilizados foram: *Mathworks® Matlab®*, *Mathworks Simulink®* e *Toolboxes*, *Xilinx® System Generator™*, *Xilinx ISE® Design Suite*, *Xilinx Plan Ahead®*, *Xilinx iMPACT®*.

O *Matlab* (MATrix LABoratory) é uma ferramenta utilizada para cálculos computacionais, desenvolvimento de algoritmos, modelagem, projeto e simulação de sistemas, incluindo sistemas de controle, processamento de sinais e sistemas de telecomunicações, dentre outros segmentos passíveis de modelagem matemática/computacional. Este ambiente é composto por um interpretador de comandos, uma linguagem de scripts, simulador e ambiente de modelagem. Embora o ambiente principal seja fechado, a maioria dos algoritmos tem código fonte e, em geral, são categorizados por *plug-ins* denominados *Toolboxes* (38).

O *Simulink*, ambiente integrado ao *Matlab*, Figura 3.1, já disponível nesta Unidade do INPE-CRN, permite o desenvolvimento e simulação de diagramas de blocos, contendo diversos módulos os quais facilitam a modelagem matemática, desde a geração de sinais, processamento destes sinais e visualização (38). Além disso, este ambiente possui uma *toolbox* denominada *HDL Coder™*, que é capaz de gerar códigos em HDL a partir de diagramas de blocos, contendo funções lógicas e aritméticas básicas, compatíveis com os ambientes de síntese ou compilação de projetos em FPGA, os quais, a partir

de uma linguagem intermediária, possam gerar *bitstreams*, compatíveis com os CIs já disponíveis ou a serem especificados.

Figura 3.1: Interface gráfica do software *Mathworks Simulink*.



Fonte: Produção dos autores.

O *ISE Design Suite* é um conjunto de aplicativos oferecidos pela fabricante *Xilinx* para projetos nas várias famílias de seus FPGAs.

Para os primeiros experimentos de síntese de códigos foi utilizada a ferramenta *System Generator*, a qual faz parte do *ISE Design Suite*. Trata-se de um *software* que trabalha em conjunto com o *Simulink* e o *Matlab* e permite a geração automática do arquivo de configuração *bitstream* ou de códigos em HDL a partir de um modelo gerado no *Simulink* e/ou componentes ou subsistemas de bibliotecas de terceiros e/ou do próprio projetista.

O *Xilinx System Generator* é uma das ferramentas de desenvolvimento de sistemas de Processamento Digitais de Sinais (PDS), usando FPGAs, adequada a metodologia SMMH. A sua conexão com o *software Matlab/Simulink* tornam o projeto e a prototipação, de sistemas de PDS

complexos, uma tarefa relativamente simples. Uma experiência aprofundada em arquiteturas de FPGAs da *Xilinx* ou metodologias de projeto RTL não é requerida quando se utiliza o *System Generator*. Os modelos matemáticos, representados por blocos de simulação, são sintetizados numa forma amigável, usando-se conjuntos de blocos específicos da *Xilinx* no ambiente de modelagem *Simulink*. As etapas de síntese e roteamento são automaticamente realizadas para a geração de *bitstreams* de programação do FPGA (39-41).

O *Xilinx System Generator* provê um conjunto de blocos compatíveis com o *Simulink* para diversas operações de *hardware* que podem ser implementadas em diversos tipos do FPGAs da *Xilinx*. Esses blocos podem ser utilizados para simular a funcionalidade de sistemas de *hardware* através do ambiente *Simulink*. O *System Generator* também provê alguns blocos para transformar dados fornecidos pelo ambiente de simulação (no caso, o *Simulink*) para o *hardware* (blocos do *System Generator*).

A natureza da maioria das aplicações de PDS requer formato em ponto flutuante para a representação dos dados. Em ambientes de alto nível de abstração, como, por exemplo, no *Simulink*, isto é uma tarefa fácil de implementar. No entanto, em tempo de síntese de códigos para configuração de *hardware*, esta é uma tarefa muito mais desafiadora, devido à complexidade de se implementar aritmética em ponto flutuante. No entanto, quando da necessidade de um processamento mais eficiente, o *Xilinx System Generator* utiliza o formato em ponto fixo para processar os valores numéricos do sistema (39-41).

Dentre as principais vantagens dessas ferramentas, deve-se destacar:

- a) Permite o aproveitamento das vantagens da metodologia via esquemas;

- b) Permite a realização da síntese *bitstream*, seja via módulos funcionais ou via HDL (seja Verilog ou VHDL), por exemplo, encapsulado em *Black Box*;
- c) Apresenta uma diversidade de bibliotecas de *IP Cores*, conforme apresentado em (42), prontas para uso, realizando uma chamada automática do *Core Generator*TM para utilização dos mesmos;
- d) Permite a integração com os módulos nativos do *Simulink*, durante a simulação;
- e) Permite a geração de vetores de testes em HDL *testbench*;
- f) Possibilita a co-simulação, diretamente do ambiente *Matlab/Simulink*.

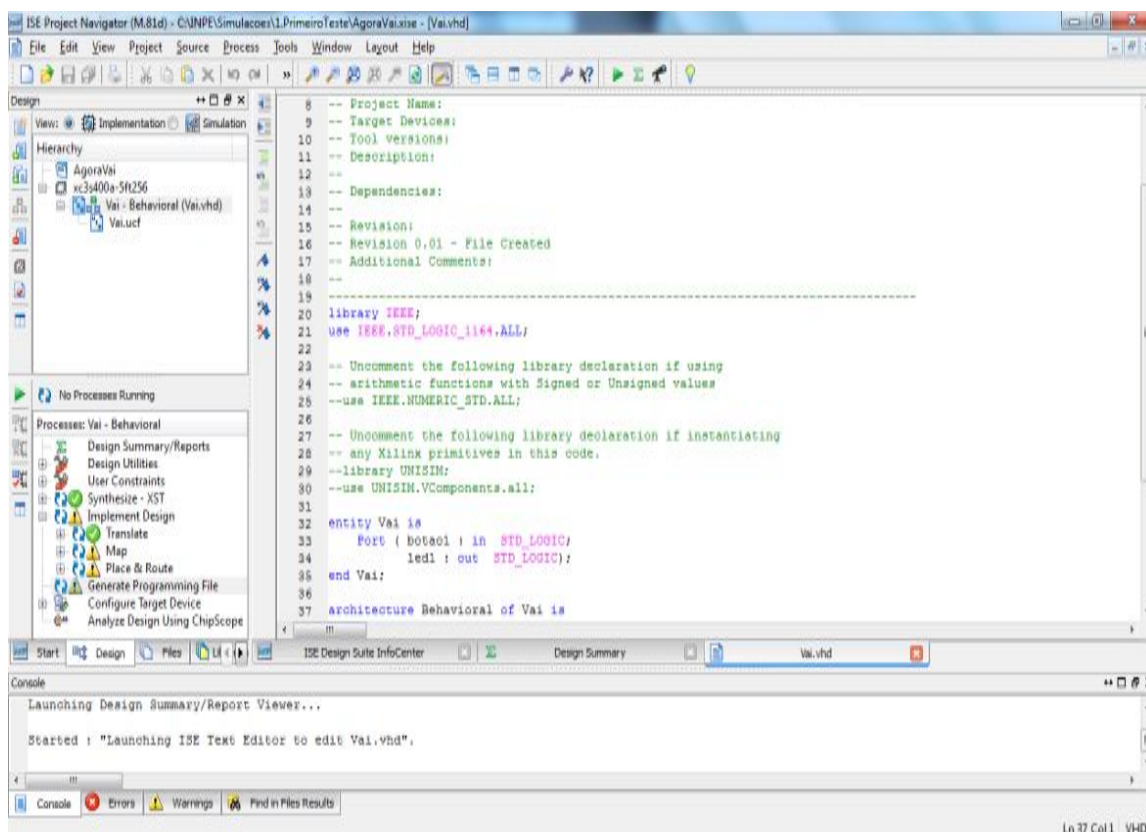
O último recurso citado, a co-simulação, merece destaque: pois esta permite que o modelo seja testado no próprio CI de FPGA, em um ambiente integrado a simulação do modelo matemático já desenvolvido ou em desenvolvimento.

Dentre as limitações do *System Generator*, deve-se destacar as seguintes restrições:

- a) Diferente da geração de códigos provenientes do *HDL Coder* (da *Mathworks*), a geração de códigos HDL, provenientes dos módulos funcionais do *System Generator*, é dedicada aos CIs da família *Xilinx*. Isto dificulta o reaproveitamento de códigos quando da necessidade de mudança de fornecedores de CIs FPGA;
- b) Até o momento, nem todas as plataformas dotadas de interfaces *ethernet* permitem a co-simulação em alta velocidade, pela adoção desse dispositivo;
- c) Até a versão 14.3 do *System Generator*, nem todos seus blocos são compatíveis com todas as famílias de FPGA da própria *Xilinx*.

A Ferramenta *ISE Project Navigator*, Figura 3.2, é a ferramenta de edição, compilação, simulação e conversão de HDL, especificamente VHDL e Verilog. Apresenta funções de debug e destaque de sintaxe. A versão do *ISE* utilizada no desenvolvimento do projeto foi a 14.3.

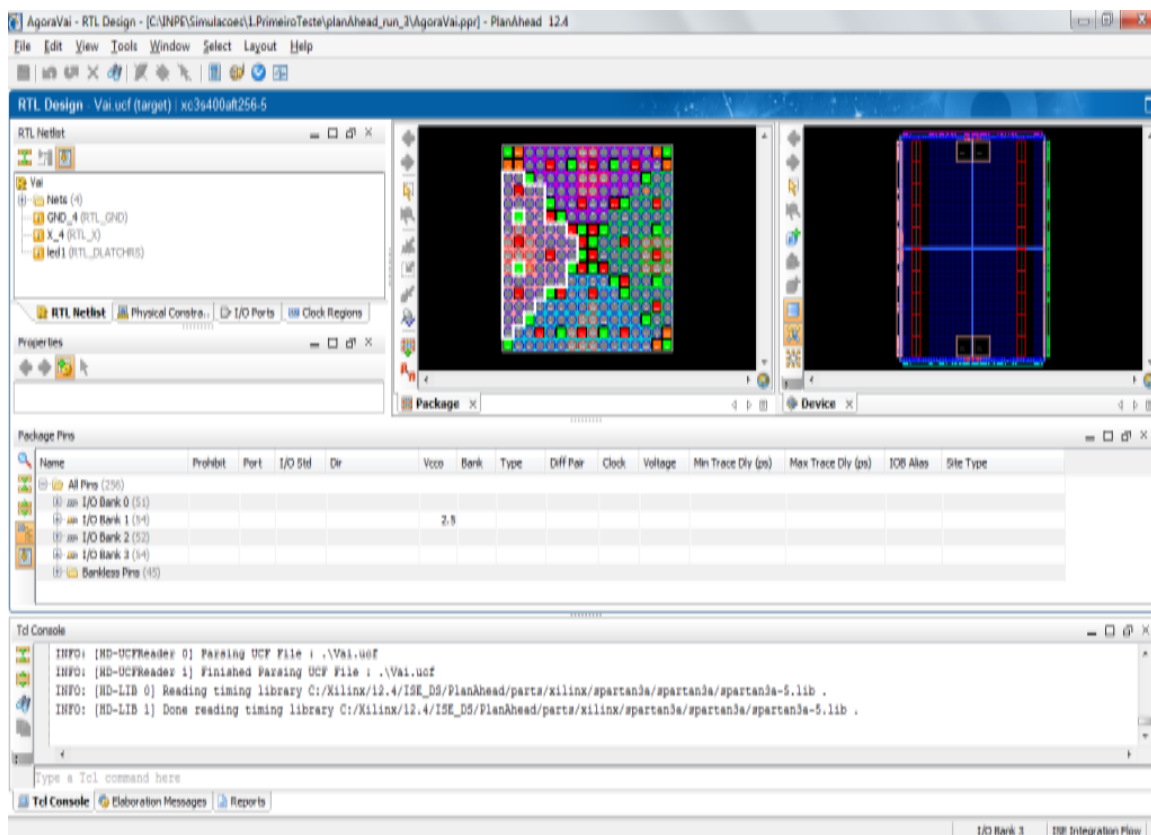
Figura 3.2: Interface gráfica do software *Xilinx ISE Project Navigator*.



Fonte: Produção dos autores.

A partir do *ISE*, pode-se executar a ferramenta *Xilinx PlanAhead* (Figura 3.3), utilizada para associar as entradas e saídas definidas na descrição do hardware aos pinos do FPGA utilizado. Em sua tela inicial temos o *footprint* do FPGA e abaixo uma janela com uma tabela onde constam os pinos e seus respectivos sinais e tipos.

Figura 3.3: Interface gráfica do software *Xilinx PlanAhead*.



Fonte: Produção dos autores.

A seguir serão apresentados alguns exemplos de utilização dessas ferramentas como um facilitador da modelagem matemática em ambiente computacional, simulação, geração automática de códigos HDL a partir da modelagem matemática, conversão de *bitstreams* e emulação em hardware, de geradores de sinais BPSK e BPSK+/-pi/3 (BPSK com índice de modulação pi/3). Embora, no desenvolvimento da seção 4 sejam gerados códigos HDL para FPGA não específico, optou-se por uma implementação final em FPGA da *Xilinx* e pela adoção do *System Generator*, no sentido de facilitar futuras comparações, dos recursos utilizados e respectivo consumo de energia da FPGA apresentado em (1) e a implementação, via *HDL Coder*, da subseção 4.1 a seguir.

4 DESENVOLVIMENTO DE GERADORES BPSK PARA FPGA GENÉRICO

Para ilustrar a presente prova de conceito da metodologia SMMH, aplicada a prototipação de subsistemas de telecomunicações, foi realizado o desenvolvimento de geradores BPSK (1) e BPSK $\pm\pi/3$, considerando a utilização da extensão da biblioteca do *Simulink/Matlab*, desenvolvida no presente projeto, a ser processada pela adoção do *HDL Coder*.

Para facilitar possíveis comparações entre a abordagem via *System Generator* (1) e a abordagem “genérica”, via HDL proveniente do *HDL Coder*, foi desenvolvido um Gerador BPSK, numa versão similar ao projeto apresentado em (1), o qual foi desenvolvido com os recursos do *System Generator* da *Xilinx*, dedicado aos FPGAs da *Xilinx*. No entanto, diferente da abordagem em (1), aqui foi utilizado o *HDL Coder* a partir do *Simulink/Matlab*, para geração automática do código HDL, a ser compilado para um FPGA genérico, a ser definido, independente de um fornecedor específico.

Diferente da abordagem utilizada em (1), a presente implementação, usando apenas módulos do *Simulink* compatíveis com o *HDL Coder*, para modelagem e simulação, demandou o desenvolvimento de novos módulos no *Simulink*, observando-se a compatibilidade com o *HDL Coder*. Do ponto de vista da metodologia de desenvolvimento, esta prova de conceito permitiu verificar o nível de automação de geração de códigos HDL, diretamente da modelagem matemática definida via *Simulink/Matlab*. Do ponto de vista de aproveitamento operacional, pretende-se usar estes geradores na realização de testes de protótipos de demoduladores BPSK.

No desenvolvimento a seguir, será apresentado o modelo matemático, a ser definido no *Simulink*, o qual, uma vez submetido ao *HDL Coder*, gera automaticamente o código HDL. Numa segunda etapa, o código HDL gerado é submetido a uma ferramenta de *síntese* (compilação de HDL para *bitstream*) compatível com o FPGA a ser utilizado. Deve-se observar que, diferente da metodologia usando *System Generator* da *Xilinx*, como utilizada em (1), aqui, a

etapa de geração de códigos HDL, a partir do modelo matemático simulado, é executada automaticamente pelo conversor *HDL Coder*. Ou seja, uma vez definido o modelo no *Simulink*, compatível com o *HDL Coder*, não há necessidade de intervenção humana nas tarefas de conversão da modelagem matemática simulada para geração dos códigos HDL.

Além disso, diferente de (1) a metodologia em desenvolvimento aproveita todas as vantagens do maior nível de abstração de hardware, via modelagem matemática, sem a dependência de um único fornecedor de FPGAs e/ou de um FPGA específico. No entanto, deve-se observar a necessidade de um conversor de HDL para *bitstream* dedicado ao FPGA da versão do *hardware* final.

4.1 Modelo simplificado de um gerador BPSK

Conforme descrito em (1), a expressão matemática para o sinal BPSK é dada por:

$$s(t) = A\{\sum_{n=0}^{\infty} x[n]g[(t - nT) + \tau]\} \cos[2\pi(f_c + f_d)t + \phi] + n(t) \quad (1)$$

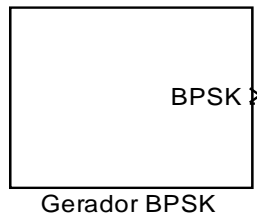
Em que os termos $s(t)$, A , T , τ , f_c , f_d e ϕ são respectivamente o sinal modulado no tempo, a amplitude do sinal, o período de bit, o atraso de símbolo, a frequência da portadora; desvios de frequência e fase da portadora. O termo $x[n]$ é o sinal de símbolo BPSK tendo valores de +1 ou -1, $g(t)$ representa um pulso retangular em banda-base e o termo $n(t)$ é o ruído Gaussiano de média zero e densidade de potência espectral $N_0/2$.

E na forma discreta, onde $t = k/f_s$ em que f_s é a frequência de amostragem, o sinal é expresso por:

$$s(k) = A[\sum_{n=0}^{\infty} x[n]g\left(\frac{k}{f_s} - nT + \tau\right)] \left[\cos\left(2\pi\frac{f_d}{f_s}k + \phi\right) \cos\left(2\pi\frac{f_c}{f_s}k\right) - \right. \\ \left. \text{sen}\left(2\pi\frac{f_d}{f_s}k + \phi\right) \text{sen}\left(2\pi\frac{f_c}{f_s}k\right) \right] + n[k] \quad (2)$$

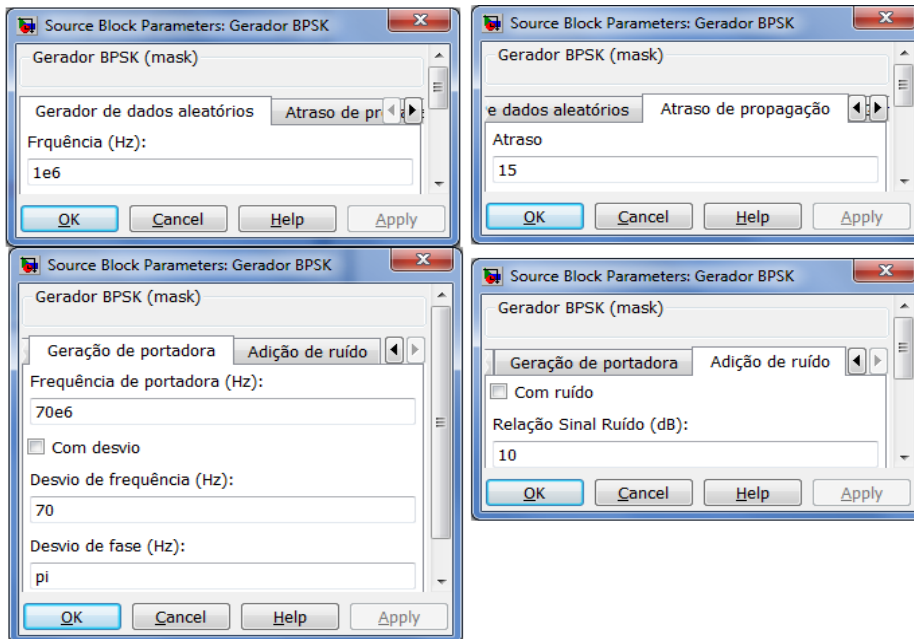
Um modelo de gerador BPSK baseado na Equação (2) foi implementado no ambiente de simulação *Mathworks Simulink*, utilizando os blocos compatíveis com a *toolbox HDL Coder* do *Matlab*. A Figura 4.1 mostra o bloco "Gerador BPSK" implementado, em que é possível configurar através de uma caixa de diálogo mostrada na Figura 4.2 os seguintes parâmetros: Taxa de geração de dados aleatórios, atraso de propagação, frequência e desvios de frequência e fase de portadora e adição de ruído.

Figura 4.1: Gerador BPSK.



Fonte: Produção dos autores.

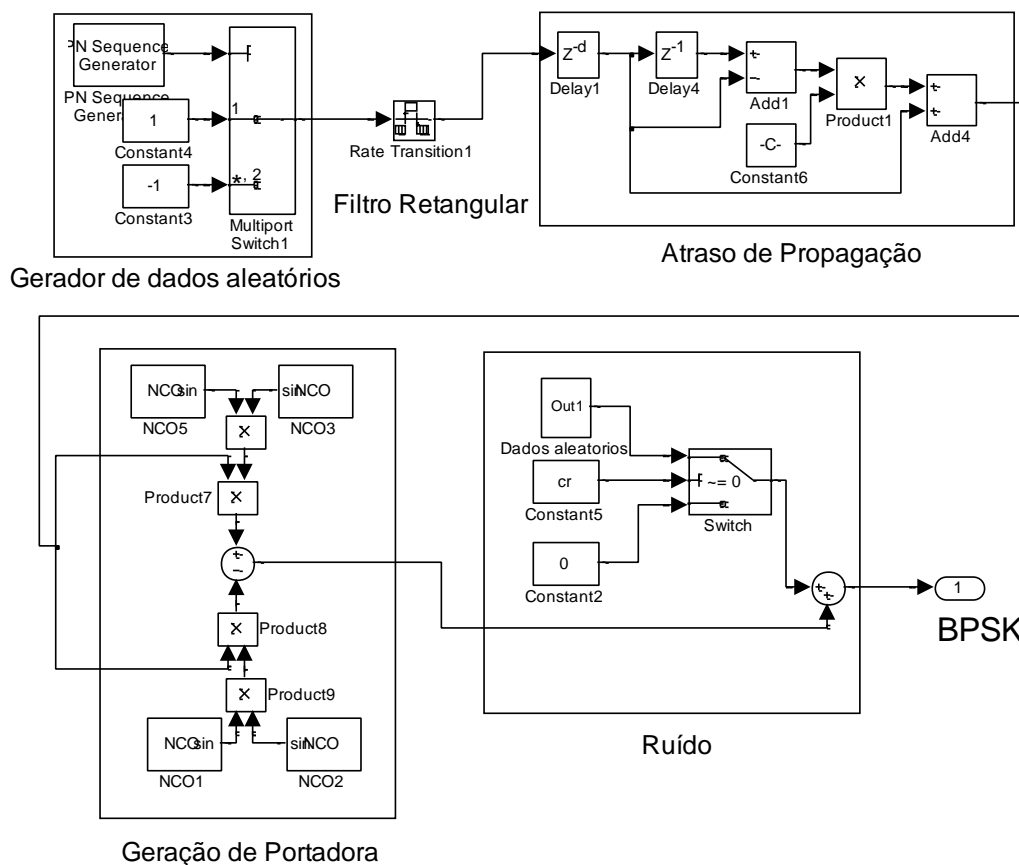
Figura 4.2: Caixa de diálogo do gerador BPSK.



Fonte: Produção dos autores.

Os sub-blocos que compõem o gerador BPSK estão dispostos no diagrama de alto nível de abstração no ambiente do *Simulink* mostrado na Figura 4.3.

Figura 4.3: Diagrama de blocos do gerador BPSK.



Fonte: Produção dos autores.

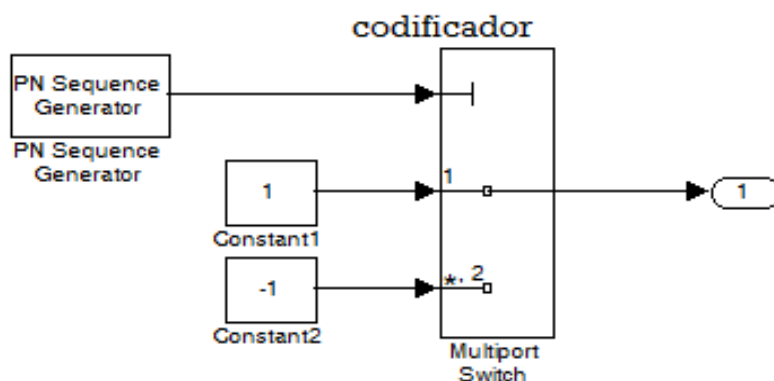
A descrição dos blocos é detalhada a seguir:

- Gerador de dados aleatórios: Gera os dados de entrada aleatórios com valores de +1 ou -1 com taxa definida pelo usuário.
- Filtro retangular: Realiza uma sobreamostragem do sinal.
- Atraso de propagação: Incorpora um atraso de propagação pré-determinado no sinal de entrada através de interpolação linear.

- Geração de portadora: Gera uma onda senoidal com desvios de frequência e de fase, em que tanto a frequência da portadora, como os valores dos desvios de fase e frequência são definidos pelo usuário.
- Ruído: Gera amostras de ruído branco e Gaussiano (AWGN).

A geração de dados aleatórios, que representa o sinal de informação, foi implementada a partir dos blocos *PN sequence generator* e codificador bipolar, como mostrado na Figura 4.4. O bloco *PN sequence generator* gera os bits aleatórios ('0' e '1') e o codificador bipolar realiza a conversão desses bits para um sinal de amplitude +1 ou -1.

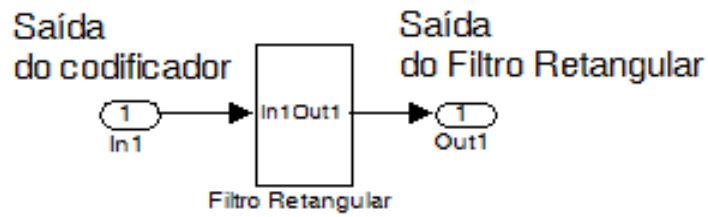
Figura 4.4: Implementação do bloco geração de dados aleatórios.



Fonte: Produção dos autores.

Após passar pelo codificador, o sinal é sobre-amostrado através de um bloco de *Rate Transition* atuando como um filtro retangular, conforme visto pelo bloco da Figura 4.5.

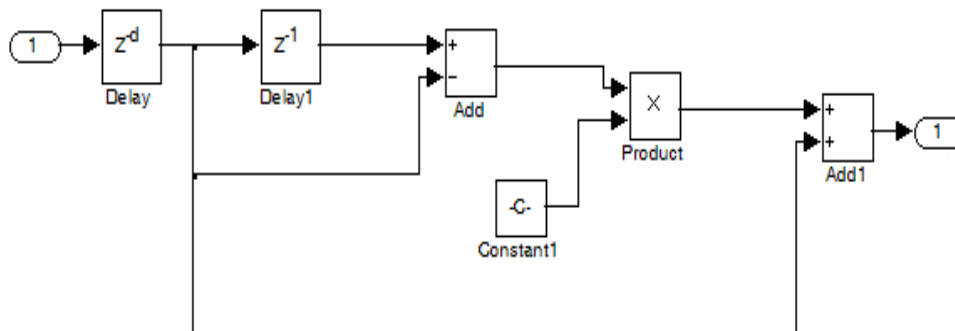
Figura 4.5: Sobreamostragem e filtragem do sinal.



Fonte: Produção dos autores.

A etapa seguinte é a inserção de um atraso de propagação característico ao canal espacial, que foi implementado através uma interpolação linear, de acordo como mostrado na Figura 4.6.

Figura 4.6: Implementação do atraso de propagação.



Fonte: Produção dos autores.

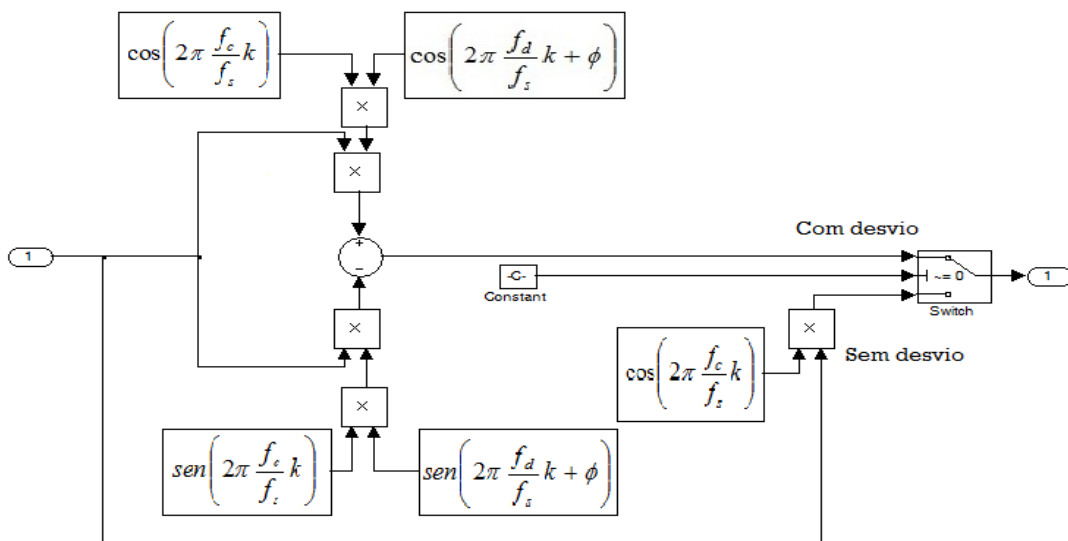
Para finalizar a geração do sinal BPSK, é realizada a multiplicação do sinal atrasado por uma onda portadora com ou sem desvios de fase e frequência, seguido da adição de ruído AWGN.

Os blocos para geração da portadora estão descritos na Figura 4.7, onde a opção de inserir ou não desvios de fase e frequência se dá por meio do bloco *switch* que tem como critério de seleção a escolha definida pelo usuário.

A adição de ruído ao sinal é implementada pelos blocos mostrados na Figura 4.8, onde se tem a geração dos dados aleatórios por meio do bloco *PN sequence generator*, em seguida é realizada a definição do nível de ruído através do bloco de ganho e por fim, tem-se um bloco de *switch* em que é possível inserir ou não ruído através da escolha do usuário.

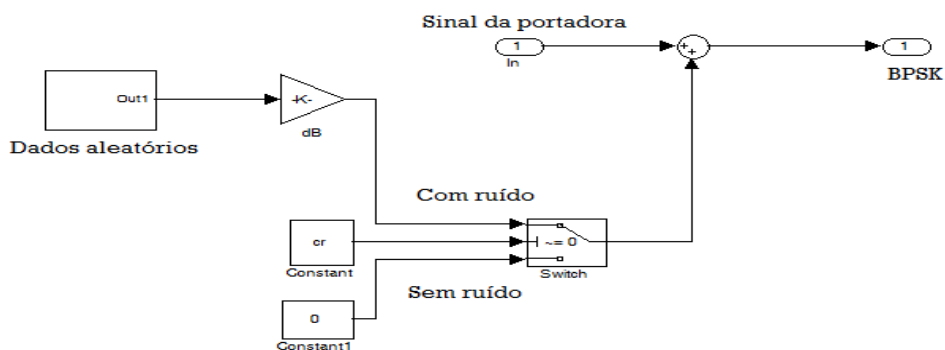
A seguir foi investigada também a integração entre o modelo *HDL Coder* desenvolvido e uma interface de conversor DAC da 4DSP (ref. FMC-150 (43), (usando o conceito denominado pela *Xilinx* de *Black Box*, conforme seção 5 e Apêndice C) incluso nas plataformas ML605 (44) baseada no CI de FPGA Virtex® 6 e KC705 (45) baseada no CI Kintex® 7 da *Xilinx/Avnet*).

Figura 4.7: Implementação do bloco geração da onda portadora.



Fonte: Adaptado de (1).

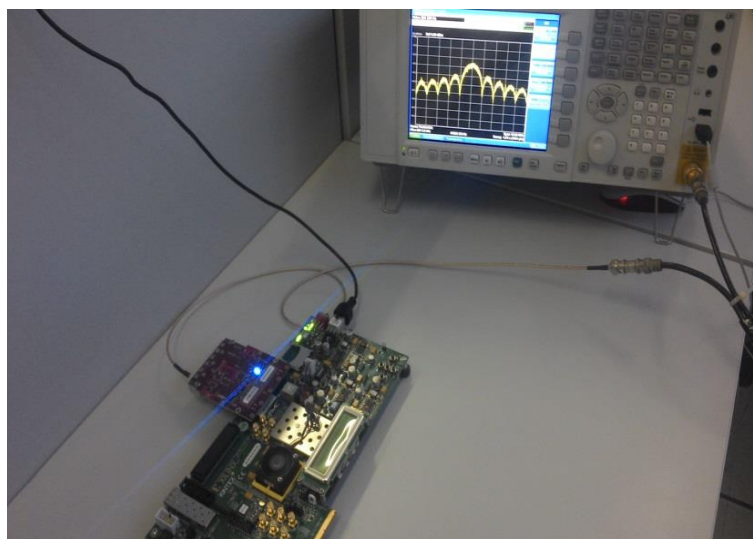
Figura 4.8: Implementação do bloco de ruído.



Fonte: Produção dos autores.

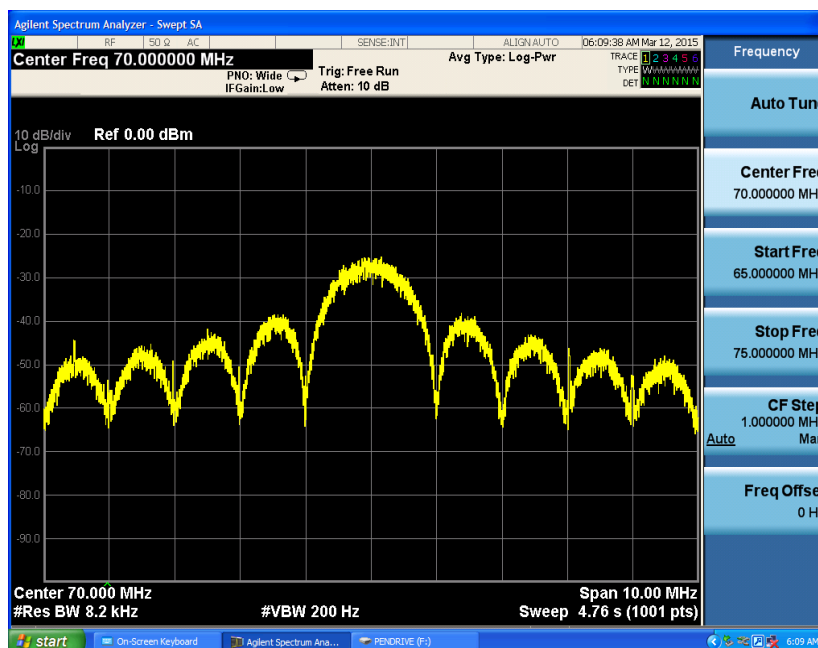
As Figuras 4.9 e 4.10 mostram o esquema de geração e o espectro do sinal BPSK gerado na placa de desenvolvimento em conjunto com o módulo FMC-150. Foi utilizada uma taxa de transmissão de 1 Mbps e uma portadora centrada em 70 MHz.

Figura 4.9: Esquema de geração do sinal BPSK.



Fonte: Produção dos autores.

Figura 4.10: Espectro do sinal BPSK centrado em 70 MHz.



Fonte: Produção dos autores.

4.2 Modelo simplificado de um gerador BPSK com índice de modulação $\pi/3$

No sentido de exemplificar a flexibilidade da metodologia de desenvolvimento dessa etapa do projeto, foi concebido um gerador BPSK com índice de modulação de $\pi/3$ (BPSK $\pm\pi/3$), incluindo a modelagem do canal. Este protótipo de gerador BPSK foi desenvolvido para possíveis emulações de sinais similares aos recebidos por satélites de coleta de dados, como uma primeira aproximação aos sinais transmitidos por Plataformas de Coleta de Dados-PCDs, a ser utilizado como referência para futuros testes de demoduladores BPSK.

A seguir será definido o modelo matemático, similar ao já apresentado em (1), no entanto, este modelo foi adaptado para incorporar o índice de modulação $\pi/3$.

Dessa maneira, o sinal binário de informação modifica a fase da onda portadora em dois ângulos: $+\pi/3$ e $-\pi/3$. A expressão matemática do sinal para esse tipo de modulação é apresentado a seguir:

$$s(t) = A \cos[2\pi(f_c + f_d)t + k_p \{\sum_{n=0}^{\infty} x[n]g[(t - nT) + \tau]\} + \phi] + n(t) \quad (3)$$

Em que os termos A , T , τ , f_c , f_d , ϕ e k_p são respectivamente a amplitude do sinal, o período de bit, o atraso de símbolo, a frequência da portadora; desvios de frequência e fase da portadora e sensibilidade a fase do modulador. O termo $x[n]$ indica o símbolo BPSK tendo valores de +1 ou -1, $g(t)$ representa um pulso retangular em banda-base e o termo $n(t)$ é o ruído Gaussiano de média zero e densidade de potência espectral $N_0/2$.

E na forma discreta, onde $t = k/f_s$ em que f_s é a frequência de amostragem, o sinal é expresso por:

$$s[k] = A \cos \left[2\pi(f_c + f_d) \frac{k}{f_s} + k_p \left\{ \sum_{n=0}^{\infty} x[n]g \left[\left(\frac{k}{f_s} - nT \right) + \tau \right] \right\} + \phi \right] + n[k] \quad (4)$$

Um modelo de gerador com modulação BPSK com índice de modulação de $\pi/3$ (BPSK $\pm\pi/3$) baseado na Equação (2) foi implementado no ambiente de simulação *Mathworks Simulink*, a partir de blocos compatíveis com a *toolbox HDL Coder do Matlab*.

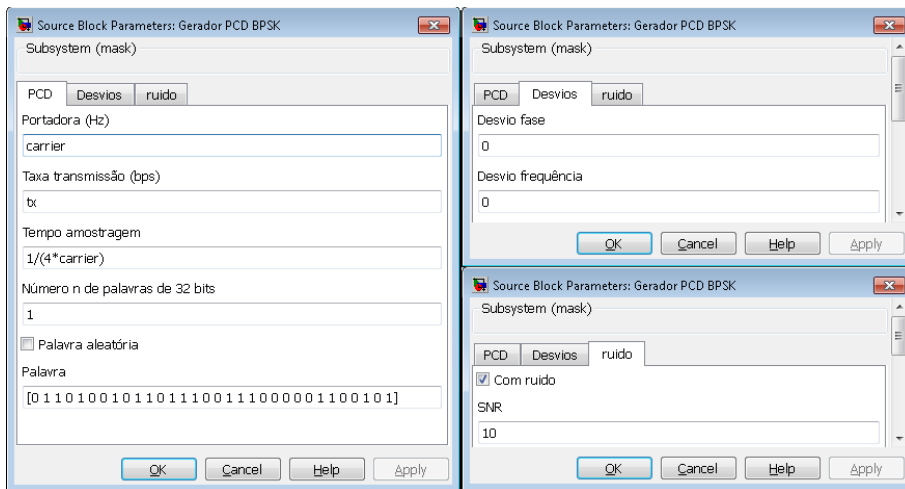
A Figura 4.11 mostra o bloco "Gerador BPSK $\pm\pi/3$ " implementado, em que é possível configurar através de uma caixa de diálogo, mostrada na Figura 4.12, os seguintes parâmetros: taxa de geração de dados, frequência de portadora, adição de ruído e dados de serviço (ex.: dados de sensores dentre outros) como informação a ser transmitida.

Figura 4.11: Gerador BPSK+/-pi/3



Fonte: Produção dos autores.

Figura 4.12: Caixa de diálogo do gerador BPSK+/-pi/3.



Fonte: Produção dos autores.

O sinal padrão a ser transmitido por essa PCD hipotética, possui duração que varia entre 360 ms e 920 ms, dependendo do tamanho da informação, a ser enviada, e obedece ao seguinte modelo:

| | | | | | | |
|----------------|--------------------|---|---|---|---|---|
| Portadora Pura | Mensagem | | | | | |
| | Portadora Modulada | | | | | |
| | a | b | c | d | e | f |

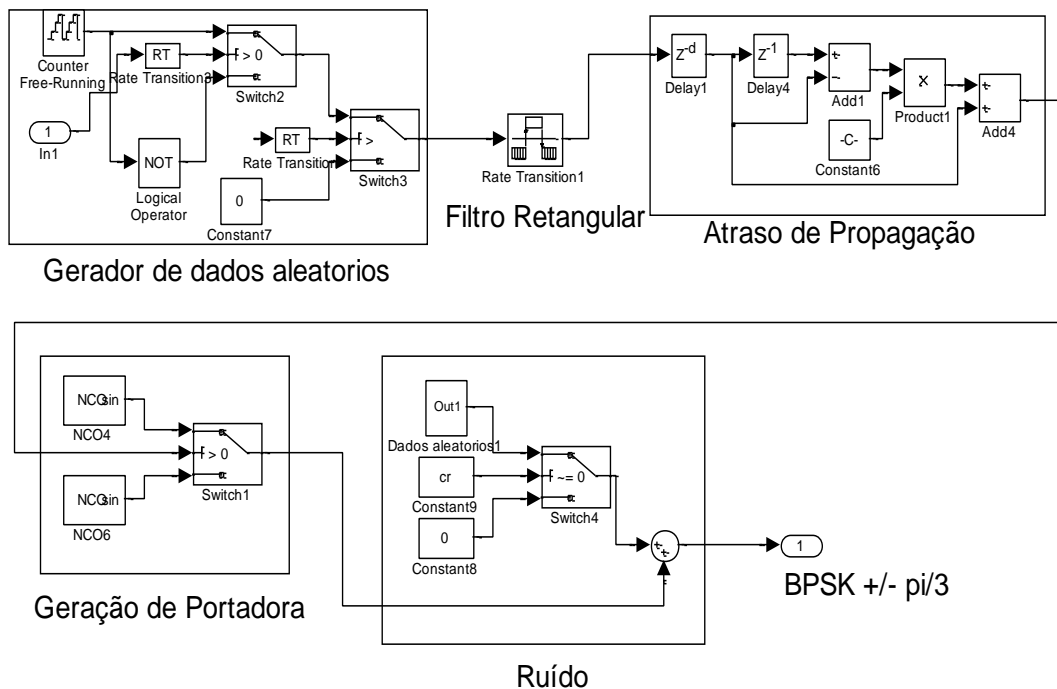
Taxa de Dados: 400 bps.

Codificação: Bifase-L.

| | |
|------------------------------------|-------------------------------|
| Portadora Pura: | Duração 160 ms. |
| a) Sincronização de bits: | 15 bits '1'. |
| b) Sincronismo de Palavra: | 8 bits '00010111'. |
| c) Inicialização: | 1 bit '1'. |
| d) Identificação de PCD: | 20 bits. |
| e) Número n de blocos da mensagem: | 4 bits ($1 \leq n \leq 8$). |
| f) Campo de dados: | $n \cdot 32$ bits. |
| Duração total do sinal: | 360 ms ($n = 1$) |
| | 920 ms ($n = 8$) |

Os sub-blocos que compõem o gerador BPSK+/-pi/3 são mostrados na Figura 4.13.

Figura 4.13: Diagrama de blocos do gerador BPSK+/-pi/3.



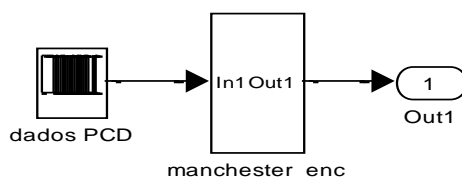
Fonte: Produção dos autores.

A descrição dos blocos é detalhada a seguir:

- Geração de dados de PCD: Geração da mensagem em banda base, baseada no esquema da Figura 4.14.
- Filtro retangular: Realiza uma sobreamostragem no sinal.
- Atraso de propagação: Incorpora um atraso de propagação pré-determinado no sinal de entrada através de interpolação linear.
- Geração de portadora: Gera duas ondas senoidais com fases iguais a $+\pi/3$ e $-\pi/3$, que multiplexadas formam o sinal BPSK com índice de modulação de $\pi/3$.
- Ruído: Gera amostras de ruído AWGN.

A geração de dados a serem utilizados para emulação de um transmissor de PCD, representando o sinal de informação, foi implementada a partir dos blocos *Repeating Sequence Stair* e codificador Manchester, como mostrado na Figura 4.14. O bloco *Repeating Sequence Stair* gera o padrão de mensagem da PCD (bits '0' e '1') e o codificador *Manchester* realiza uma codificação bifase nos dados de entrada. Durante o período de 160 ms, é gerado apenas um sinal de portadora pura, sem qualquer modulação, ou seja, neste período o codificador *Manchester* permanece desabilitado.

Figura 4.14: Implementação do bloco geração de dados de PCD.

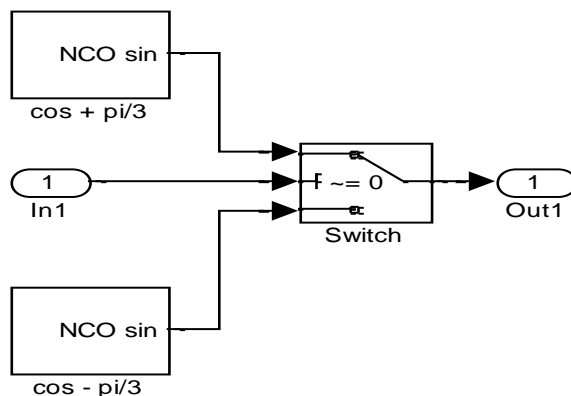


Fonte: Produção dos autores.

Os blocos “Filtro retangular”, “Atraso de propagação” e “Ruído” já foram descritos no desenvolvimento do gerador BPSK da seção 4.1.

Enquanto que o bloco para geração da portadora é apresentado na Figura 4.15, a seguir.

Figura 4.15: Implementação do bloco geração da onda portadora.



Fonte: Produção dos autores.

Este bloco gera duas ondas senoidais com fases iguais a $+\pi/3$ e $-\pi/3$, que multiplexadas a partir do sinal de informação, formam o sinal bpsk com índice de modulação de $\pi/3$.

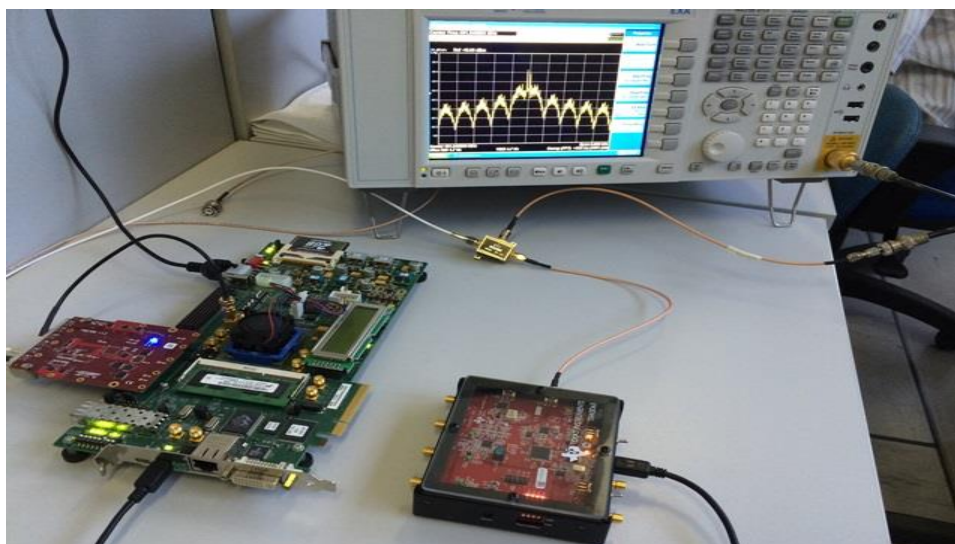
Para a implementação prática desse gerador baseado em FPGA, a portadora foi ajustada em 70 MHz. Outro sinal senoidal de frequência 331,65 MHz foi gerado a partir da placa de oscilador local TSW3065EVM da Texas Instruments. A mixagem desses dois sinais gera um terceiro sinal modulado próximo a 401,65 MHz.

As Figuras 4.16 e 4.17 mostram o esquema de geração do sinal BPSK e o respectivo espectro com índice de modulação $\pi/3$.

Para realização desses experimentos, foram utilizados os seguintes equipamentos: placa de desenvolvimento KC705 em conjunto com o módulo FMC-150, oscilador local TSW3065EVM e mixer ETI M2X250M. Para os modelos apresentados, a ferramenta *HDL Coder* foi configurada para geração

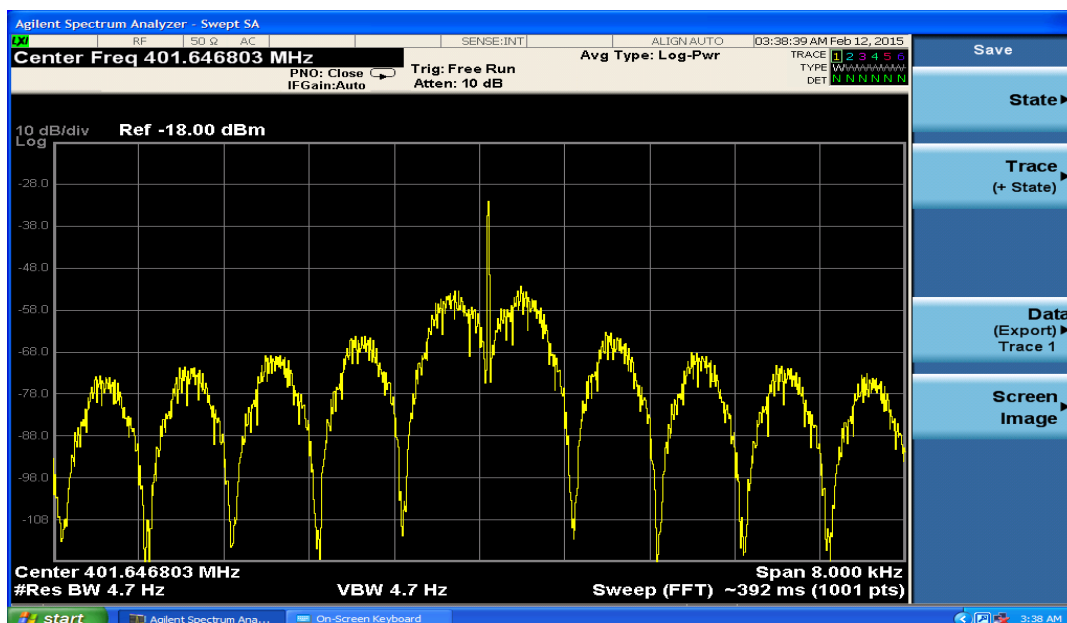
de códigos em *VHDL* (a mesma também permite geração de códigos em *Verilog*), e o formato numérico máximo considerado foi *Fixdt(1,16,12)*.

Figura 4.16: Esquema de geração do sinal BPSK+/-pi/3.



Fonte: Produção dos autores.

Figura 4.17: Espectro do sinal BPSK+/-pi/3 em torno de 401,65 MHz.



Fonte: Produção dos autores.

5 DEFINIÇÃO DE BLACK BOX PARA INTEGRAÇÃO DE COMPONENTES DE TERCEIROS A EXTENSÃO DA BIBLIOTECA *HDL CODER*

Dentre os itens não disponíveis nativamente no ambiente de desenvolvimento, foi verificada a necessidade da definição de novos *drivers* considerando o controle direto de conversores analógicos digitais (*ADC*) e digitais analógicos (*DAC*), mezanino FMC150, bem como um gerenciador de *display* das plataformas baseadas na Virtex6 e Kintex7 (Plataforma ML605 e KC705, da *Xilinx/Avnet*), já disponíveis nesta Unidade do INPE, via outros projetos.

Diferente dos blocos *M-Codes*, que suportam apenas simples códigos *Matlab*, as estruturas tipo *Black Box* permitem códigos HDL, os quais podem conter uma lógica de programação mais elaborada.

As estruturas tipo *Black Box* podem ser integradas ao ambiente *Matlab/Simulink/System Generator*, quando, por exemplo, em tempo de geração de *bitstreams*. Isto permite o aproveitamento de códigos específicos como, por exemplo, *drivers* (gerenciamento) de *DAC*, *ADC* e *display*, usados nas Plataformas da *Xilinx*. Portanto, em tempo de síntese de códigos *bitstreams*, as estruturas tipo *Black Box* são compiladas similarmente aos blocos nativos da biblioteca já disponível no *System Generator*.

No apêndice C é apresentado um exemplo da metodologia de implementação de estruturas tipo *Black Box*, contendo um roteiro do desenvolvimento de um módulo para gerenciamento de *display* da plataforma ML605.

6 DIFICULDADES ENCONTRADAS

O projeto sofreu dificuldades com relação à disponibilidade de recursos humanos qualificados, bem como algumas dificuldades técnicas, como detalhadas a seguir:

- Claramente há uma carência de engenheiros no Brasil, principalmente de engenharia eletrônica e mais especificamente de telecomunicações, de modo que os salários pagos pelo mercado para este tipo de profissional são bem superiores aos valores das bolsas DTI.
- Portanto, devido à escassez de pessoal qualificado em engenharia, e, também ao número de projetos aprovados em áreas correlatas, houve grande dificuldade em atrair bolsistas capacitados a serem contratados para o projeto. Esta dificuldade foi contornada com treinamento interno por parte da própria equipe do projeto.
- Do ponto de vista técnico, devido ao fato dos mais modernos componentes utilizados, tais como FPGAs, DACs e ADCs de última geração, serem produzidos com tecnologia muito recente, faz-se necessário o desenvolvimento de *firmwares*. Seja via módulos *M-Code* ou *Black Box*, e adaptações de *Reference Designs*, dos respectivos fabricantes.
- Por outro lado, este desenvolvimento adicional, tornou possível tirar o máximo proveito destes dispositivos e ao mesmo tempo preparar a infraestrutura de todas as provas de conceitos a serem efetuadas. Além disso, o êxito na solução dessas dificuldades tem definido um ganho maior de *expertise* da equipe.

7 ANÁLISE DOS RESULTADOS E PERSPECTIVAS

Os resultados, das tarefas realizadas nesta primeira etapa, conforme Apêndice A, foram satisfatórios. Isto é, todos os resultados experimentais obtidos estavam em conformidade, iguais ou próximos ao esperado, ou seja, os valores teóricos estão próximos aos resultados obtidos por meio de simulação/emulação. Eventuais diferenças de temporização existentes entre a simulação e os resultados de implementações devem-se ao fato das ferramentas de simulação não levarem em consideração as especificidades de cada arquitetura de hardware, principalmente no que diz respeito à interconexão dos componentes do FPGA e dos atrasos de sinais por ela causados.

O ambiente *Matlab/Simulink* em conjunto com o *Xilinx System Generator* se mostraram ferramentas bastante versáteis para simulação e desenvolvimento de sistemas digitais em FPGA. As ferramentas utilizadas simplificaram o processo de simulação e implementação e provaram não haver necessidade de conhecimentos aprofundados, do *hardware* final, quando do desenvolvimento de sistemas digitais. Tal fato representa uma redução de tempo de desenvolvimento e de depuração.

Durante a realização de algumas simulações, observou-se a inexistência de blocos funcionais comuns em circuitos para comunicação, como, por exemplo, circuitos divisores de frequência, o qual foi implementado a partir de estruturas mais simples do *System Generator* e compatíveis com o *HDL Coder* do *Simulink*. Tal deficiência motivou o desenvolvimento de vários outros blocos funcionais amplamente utilizados em diversos circuitos digitais, principalmente aplicados em comunicação digital. Tais blocos, que já se encontram em desenvolvimento ou em fase de testes, devem servir como uma biblioteca própria para o INPE em projetos correntes, e, também devem facilitar novas provas de conceito, através da construção de blocos mais voltados à área de comunicações espaciais.

Portanto, pode-se afirmar que a primeira etapa do projeto atingiu os objetivos previstos no Plano de Trabalho, considerando o estudo de ferramentas e experimentos de prototipação de circuitos e subsistemas, implementados em microeletrônica reconfigurável, a serem aplicados principalmente em comunicação digital.

Além disso, a realização de modelagens matemática e computacional, simulações computacionais e experimentos usando os circuitos implementados, em forma de subsistemas digitais, bem como a respectiva geração de bibliografia de referência, tem contribuído sobremaneira para formação de recursos humanos na fronteira do conhecimento de prototipação de circuitos, via modelagem matemática/computacional.

A seguir é apresentada uma primeira análise, das vantagens x desvantagens, da adaptação da metodologia SMMH, conforme o Apêndice B, empregada no corrente projeto:

Principais vantagens da metodologia em desenvolvimento:

- Permite definição de protótipos ao nível sistêmico ou funcional;
- Permite sintetizar protótipos diretamente da modelagem matemática simulada, considerando provas de conceitos de dispositivos sofisticados, e consequentemente complexos, simulados em ambiente *Matlab/Simulink*, e co-simulado e/ou emulado em *hardware* de eletrônica reconfigurável, com maior produtividade quando comparado ao desenvolvimento baseado em programação HDL;
- A partir de uma biblioteca desenvolvida diretamente da modelagem matemática, exaustivamente testada, pode reduzir os erros observados em codificação de algoritmos com menor nível de abstração em relação ao hardware;

- Permite atualizações ao nível de *firmware* (*bitstream*), numa forma incremental e simplificada, via circuitos/subsistemas definidos por *software* a posteriori;
- Permite depuração do *firmware/hardware* de propósito geral, em forma co-simulada ou emulada, equivalente ao modelo matemático;
- Sensível aumento de facilidade de depuração ou aperfeiçoamento de projetos (ao nível de *firmware*), mesmo após a implementação do *hardware* final;
- Permite boa abstração de *hardware* (e conseqüentemente de um fornecedor específico de FPGA), o que é extremamente importante, principalmente, quando da prova de conceito de novos subsistemas de comunicação;
- Facilidade para estimativa de recursos e demanda de energia;

Principais desvantagens da metodologia em desenvolvimento:

- Possível aumento do número de linhas de código e redução da facilidade de inteligibilidade, destes códigos, em baixo nível de abstração, por exemplo, em HDL. Isto pode ser minimizado quando do desenvolvimento da biblioteca consolidada (exaustivamente testada), de dispositivos dedicados, usando um compilador com tecnologia nacional, em direção a uma futura plataforma de hardware nacional (ou seja, através de uma otimização para um CI bem conhecido).
- Adequada apenas após exaustiva bateria de testes para eliminação de possíveis erros na implementação de componentes de menor nível de abstração. Solução decorrente do contorno da dificuldade anterior.
- Aumento da dificuldade de otimização em um nível mais baixo de abstração em relação ao *hardware* ou CI final. Dificuldade similar à utilização de qualquer projeto codificado em alto nível de abstração. Embora seja totalmente justificável quando da prova de conceito de novos modelos matemáticos (ex.:

novos modelos de modulação e/ou demodulação), isto demanda uma análise de custo benefício para uma implementação final.

- Embora o ambiente *Matlab/Simulink* esteja consolidado como uma das principais plataformas de simulação de dispositivos a serem desenvolvidos em FPGA, via modelagem matemática, deve-se ressaltar que até o momento, considerando-se as versões do *Malab/Simulink 2012a*, ainda foram encontradas idiossincrasias, como por exemplo: a perda de amostras pelo bloco “*Downsample*”, causando um adiantamento indesejado no sinal, bem como o mau funcionamento do bloco “*Matrix Concatenate*”, o qual, durante as simulações, gerou alguns erros de concatenação.

No que concerne às perspectivas, espera-se que, nas próximas etapas do projeto, a extensão da biblioteca em desenvolvimento facilite novas provas de conceitos concernentes aos dispositivos a serem aplicados em novos subsistemas de telecomunicações e/ou processamento digital de sinais, bem como em equipamentos para instrumentação a partir das diferentes plataformas de geração de *bitstreams*, a fim de obter-se uma estimativa comparativa de recursos e consumo entre os CIs de diferentes famílias de FPGAs, considerando as facilidades de desenvolvimento para geração de códigos e/ou respectivas implementações em *hardware*.

No sentido de aproveitar o trabalho desenvolvido, pode-se destacar uma possível aplicação através da integração do subsistema de modulador BPSK+/- $\pi/3$, já desenvolvido, a um receptor GPS, a ser implementado também em FPGA, como um subsistema similar aos já encontrados na literatura (46, 47). A integração, de um receptor GPS a um transmissor de PCD, aproveitando o modulador já desenvolvido, deverá facilitar a implementação de todo o processamento digital de uma nova PCD, incluindo a função de localização geográfica (48), em um único CI de FPGA e/ou ASIC, considerando a expansão da infraestrutura do Sistema Brasileiro de Coleta de Dados Ambientais-SBCDA/Sistema Integrado de Dados Ambientais-SINDA (49).

REFERÊNCIAS BIBLIOGRÁFICAS

- (1) LUCENA, A. M. P.; OLIVEIRA, P. D. L.; RIOS, C. S. N.; ALMEIDA FILHO, M. P.; SILVA, F. A. T. F. Flexible FPGA-based BPSK Signal Generator for Space Applications. **International Journal of Circuits, Systems and Signal Processing**, v. 8, p. 160-165, 2014.
- (2) LUCENA, A. M. P.; OLIVEIRA, P. D. L.; RIOS, C. S. N.; ALMEIDA FILHO, M. P.; PIMENTEL, D. P.; COUTINHO, K. M. H.; SILVA, F. A. T. F. **Gerador de sinais bpsk em fpga para aplicações espaciais**. São José dos Campos: INPE. 2014. (INPE/12.12.15.26-RPQ).
- (3) CARVALHO, M. J. M.; TRAVEIROS, F. E. V. Simulador de sinais de plataformas de coleta de dados (PCD's) para testes de transponders do sistema brasileiro de coleta de dados (SBCD). In: Simpósio Brasileiro DE SENSORIAMENTO REMOTO - SBSR, 15., Curitiba, PR, Brasil, **Anais...** São José dos Campos: INPE, 2011. p. 8978.
- (4) CARVALHO, M. J. M.; LINS, R. A. S.; NOWOSAD, A. G. **Demodulador de BPSK com recuperação de portadoras definido em software para os satélites do sistema brasileiro de colecta de dados**. Relatório final de projeto de iniciação científica (PIBIC/CNPq/INPE), 2007.
- (5) CARVALHO, M. J. M.; LIMA, J. S. S.; JOTHA, L. S. CONASAT-Constelação de Nano Satélites para Coleta de Dados Ambientais. In: SIMPÓSIO BRASILEIRO DE SENSORIAMENTO REMOTO - SBSR, 16., 2013, Foz do Iguaçu, PR, Brasil. **Anais...** São José dos Campos: INPE.
- (6) CARVALHO, M. J. M.; TRAVEIROS, F. E. V. **Simulador de sinais de plataformas de coleta de dados**. São José dos Campos: INPE. 2012. 18 p. (sid.inpe.br/mtc-m19/2012/07.17.13.20-MAN). (INPE/07.17.13.20-MAN). Disponível em: <<http://urlib.net/8JMKD3MGP7W/3CA8R3P>>. Acesso em: 08 jun. 2015.
- (7) CARVALHO, M. J. M.; SOUZA, R. C. S. **Transmissor realizado em software**. Relatório final de projeto de iniciação científica (PIBIC/CNPq/INPE), 2010.
- (8) DESCHAMPS, J. P.; BIOUL, G. J. A.; SUTTER, G. D. **Synthesis of arithmetic circuits**. Hoboken, New Jersey: John Wiley and Sons, 2006. 578 p.
- (9) MILDER, P. A.; FRANCHETTI, F.; HOE, J; C.; PÜSCHEL, M. **Discrete Fourier transform compiler**: from mathematical representation to efficient hardware. Carnegie Mellon University: Center for Silicon System Implementation, , 2007. Technical Report CSSI 07-01.
- (10) BEDNARA, M.; TEICH, J. Automatic synthesis of FPGA processor arrays from loop algorithms. **The Journal of Supercomputing**, v. 26, n. 2, p.149-183, 2004. doi>10.1023/A:1024447517501.

- (11) SEELAENDER, G. **Emulação e co-simulação do sistema de controle de atitude da PMM e do sistema eletro-hidráulico de uma aeronave usando FPGAs**. 2009. 195 p. (INPE-15764-TDI/1507). Dissertação (Mestrado em Mecânica Espacial e Controle) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2009. Disponível em: <<http://urlib.net/8JMKD3MGP8W/354UNE8>>. Acesso em: 08 jun. 2015.
- (12) HOME, C. **Computer memory history in Computer hard drive history**. Disponível em: <<http://www.computerhope.com/history/memory.htm>>. Acesso em: abr 2014.
- (13) ABBAS, S.; BARILE, C.; LANE, R.; LIU, P. **electrically erasable floating gate fet memory cell**. US Patent 3,836,992 Assignee: International Business Machines Corporation, 1974.
- (14) GREER, D. L. **Electrically programmable logic circuits** US Patent 3,818,452. Assignee: General Electric, 1974.
- (15) DIETMEYER, D. L.; DOSHI, M. H. Automated PLA synthesis of the combinational logic of a DDL description. **Journal of Design Automation & Fault-Tolerant Computing**, v. 3, n. 3-4, p. 241-257, 1979.
- (16) GAJSKI, D.; KHUN, R. Introduction: New VLSI Tools, **IEEE Computer**, v. 16, n. 12, p. 11-14, Dec. 1983
- (17) KAHNG, D.; SZE, S.M. A Float gate and its application to memory devices. **Bell System Technical Journal**, v. 46, n. 6, p. 1288-1295, 1967.
- (18) CARTER, W.; DUONG, K.; FREEMAN, R. H.; HSIEH, H.; JA, J. Y.; MAHONEY, J. E.; NGO, L. T.; SZE, S. L. A user programmable reconfigurable gate array. In: IEEE CUSTOM INTEGRATED CIRCUITS CONFERENCE, May 1986, Rochester, NY. **Proceedings...** IEEE, 1986. p. 233-235.
- (19) TSUTSUI, A.; MIYAZAKI, T.; YAMADA, K.; OHTA, N. Special purpose FPGA for high-speed digital telecommunication systems. In: COMPUTER DESIGN: VLSI IN COMPUTERS AND PROCESSORS, 1995, Austin. **Proceedings...** Austin: IEEE, April, 1995. p. 68-75. Proc. ICCD'95.
- (20) MCCONNELL, T. **ESC-Xilinx extensible processing platform combines best of serial and parallel processing**. EETimes, 2010. Disponível em: <http://www.eetimes.com/document.asp?doc_id=1313958>. Acesso em: jan 2013.
- (21) XILINX. **PicoBlaze 8-bit embedded microcontroller user guide for extended Spartan®-3 and Virtex®- 5 FPGAs introducing PicoBlaze for Spartan-6, Virtex-6, and 7 Series FPGAs**. 2011. Disponível em: <http://www.xilinx.com/support/documentation/ip_documentation/ug129.pdf>. Acesso em: dez 2014
- (22) XILINX. **MicroBlaze processor reference guide embedded development kit EDK 14.7**. Disponível em:

<http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_7/mb_ref_guide.pdf>. Acesso em: dez 2014.

(23) ALTERA. **Nios ii classic processor reference guide**. 2015. Disponível em: <https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/nios2/n2cpu_nii5v1.pdf>. Acesso em: abr 2015.

(24) NASS, R. **Xilinx puts ARM core into its FPGAs**. Embedded 2010. Disponível em: <<http://www.embedded.com/electronics-products/electronic-product-reviews/embedded-tools/4115523/Xilinx-puts-ARM-core-into-its-FPGAs>>. Acesso em: jan 2013.

(25) MAXFIELD, C. **The design warrior's guide to FPGAs: devices, tools and flows**. Elsevier, 2004.

(26) D'AMORE, R. **VHDL: descrição e síntese de circuitos digitais**. 1. ed. LTC Editora, 2005. 276 p. ISBN (85-21-61452-7).

(27) CILETTI, M. D. **Modeling, synthesis, and rapid prototyping with the Verilog HDL**. 1. ed. Prentice-Hall, 1999.

(28) KOLUMBAN, G.; KREBESZ, T. I.; LAU, F. CM. Theory and application of software defined electronics: Design concepts for the next generation of telecommunications and measurement systems. **Circuits and Systems Magazine, IEEE**, v. 12, n. 2, p. 8-34, 2012.

(29) IPPOLITO JR, L. J. **Satellite communications systems engineering, atmospheric effects, satellite link design and system performance**. Wiley Series on Wireless Communications and Mobile Computing, 2008.

(30) MARAL, G.; BUSQUET, M., **Satellite communications systems: systems, techniques and technologies**. 5. ed. West Sussex, UK: John Wiley & Sons Ltd., 2009.

(31) Radio Frequency and Modulation Systems-Part 1: earth stations and spacecraft. recommendation for space data system standards, CCSDS 401.0-B-21. Blue Book. Issue 21. Washington, D.C.: CCSDS, July 2011.

(32) PROAKIS, J. G.; SALEHI, M. **Communication systems engineering**. 2. ed. Prentice-Hall Inc, 2002.

(33) MENGALI, U.; ANDREA, A. N. D. **Synchronization techniques for digital receivers**. New York, NY, USA: Plenum Press, 1997.

(34) MEYR, H.; MOENECLAHEY, M.; FECHTEL, S. A. **Digital communication receivers: synchronization, channel estimation, and signal processing**. New York, NY, USA: JohnWiley & Sons, 1997.

(35) SILVA, A. S. **Demodulador OQPSK completamente digital para aplicações espaciais**. Dissertação (Mestrado em Engenharia Elétrica) -

Universidade Federal do Ceará, Departamento de Engenharia de Teleinformática, UFC, 2011.

(36) SILVA, A. S.; MOTA, J. C. M.; LUCENA, A. M. P.. Amostragem em banda passante e conversão de frequência em um demodulador OQPSK completamente digital. In: SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES - SBrT'11, 29., Outubro 2011, Curitiba, PR, Brasil. **Anais...** Curitiba, 2011.

(37) FIGUEREDO, C. G. **Demodulador BPSK completamente digital para aplicações espaciais**. B.Sc. Monografia, Universidade Federal do Ceará-UFC, Fortaleza, Brasil, 2013.

(38) MATLAB, Matlab. 8: Getting Started Guide. Disponível em: <http://www.mathworks.com/access/helpdesk/help/pdf_doc/matlab/getstart.pdf>. Acesso em: dez 2014.

(39) KNEZEVIC, M. **An introduction to Xilinx System Generator**. Disponível em: <<http://docslide.us/documents/system-generator.html>>. Acesso em: abr 2013.

(40) XILINX. Xilinx System Generator v2.1 for simulink. (Tutorial). Disponível em: <https://safe.nrao.edu/wiki/pub/CICADA/WebHome/xilinx_ref_guide.pdf>. Acesso em: abr 2013.

(41) XILINX. System Generator for DSP user guide. Manual, 2008. Disponível em: <http://www.Xilinx.com/support/sw_manuals/sysgen_user.pdf>. Acesso em: abr 2013.

(42) XILINX. System Generator for DSP Reference Guide. v.13.4. Manual, 2011. Disponível em: <http://www.xilinx.com/support/documentation/sw_manuals_j/xilinx13_4/sysgen_gs.pdf>. Acesso em: dez 2014.

(43) 4DSP. FMC150 User Manual. r1.9. 4DSP LLC, USA, 2010. Disponível em: <http://www.4dsp.com/pdf/FMC150_user_manual.pdf>. Acesso em: dez 2014.

(44) XILINX. ML605 Hardware User Guide. v.1.8. USA: XILINX, 2012. p. 96. Disponível em: <http://www.Xilinx.com/support/documentation/boards_and_kits/ug534.pdf>. Acesso em: ago 2013.

(45) XILINX. KC705 Hardware User Guide. v.1.6. USA: XILINX, 2014. p. 110. Disponível em: <http://www.Xilinx.com/support/documentation/boards_and_kits/kc705/ug810_KC705_Eval_Bd.pdf>. Acesso em: dez 2014.

(46) KAVYA, G.; THULASIBAI, V. Implementation of GPS Signal Acquisition and Tracking in FPGA. **IJRET: International Journal of Research in Engineering and Technology**. v. 03 May-2014.

(47) WANG, E.; ZHANG, S.; HU, Q.; YI, J.; SUN, X. Implementation of an embedded GPS receiver based on FPGA and microblaze. In: INTERNATIONAL CONFERENCE ON WIRELESS COMMUNICATIONS, NETWORKING AND MOBILE COMPUTING(WiCOM'08) 4., Dalian, China. **Proceedings...** Dalian: IEEE, 2008. p. 1-4.

(48) DUBUT, J. P.; CIRÍACO, R. D.; GOMES, M. P.; MANTOVANI, J. E.; SANTOS, E. F. Terminal transmissor localizável para uso em embarcações pesqueiras. In: SIMPÓSIO BRASILEIRO DE SENSORIAMENTO REMOTO - SBSR, 17., 2015, João Pessoa, PB, Brasil, 25 a 29 de abril de 2015. **Anais...** São José dos Campos: INPE.

(49) INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **SINDA - Sistema Integrado de Dados Ambientais**. Disponível em: <<http://www.sinda.crn2.inpe.br/PCD/>>. Acesso em: mai 2015.

APÊNDICE

Apêndice A. Resumo das principais tarefas de cada atividade desenvolvida.

- a. Realização de estudos teóricos sobre Dispositivos Lógicos Programáveis, arquiteturas de FPGAs e Linguagens de Descrição de *Hardware*;
- b. Instalação de diversas ferramentas de simulação, compilação e síntese de circuitos digitais;
- c. Estudos, familiarização, comparação e execução de testes envolvendo diversas ferramentas de desenvolvimento baseado em simuladores, compiladores e instaladores de sintetizadores;
- d. Familiarização com os kits de desenvolvimento baseados nas plataformas *Xilinx Spartan® 3*, *Xilinx Spartan 6* e *Xilinx Virtex 6*: realização de testes iniciais para verificação de funcionamento (validação);
- e. Desenvolvimento, simulação e co-simulação de circuitos digitais envolvendo geradores senoidais e de sequências, filtros gaussianos e retangulares, comparadores, contadores, detectores de borda, divisores de frequência, integradores, codificadores, sistema de eliminação de *jitter*, interface de *buffer* através da plataforma *Xilinx System Generator*;
- f. Início do desenvolvimento de *toolbox* dedicada para o ambiente *Xilinx Systems Generator* contendo módulos de síntese dos circuitos a serem integrados;
- g. Comparação de desempenho de sistemas envolvendo uso de *Look-Up Tables* (LUTs) e algoritmo CORDIC;
- h. Pesquisa sobre conversores Analógico/Digital compatíveis com os kits de desenvolvimento e comparação dos recursos existentes nos principais modelos existentes no mercado;
- i. Estudo e familiarização do módulo ADC/DAC FMC150 FPGA *Mezzanine Card*;

- j. Estudo sobre a adequação de módulos conversores A/D e D/A para plataforma *Xilinx Virtex 6*;
- k. Revisão de documentação e elaboração de relatório.

Apêndice B. Resumo do estado atual da metodologia SMMH:

A Metodologia SMMH consiste em:

1. Definir o modelo matemático do circuito ou subsistema (ex.: modelo adequado a modelagem matemática no ambiente *Matlab/Simulink*) a ser implementado (sintetizado em *hardware*);
2. Decompor o dispositivo funcionalmente, de acordo com os módulos da biblioteca nativa e/ou biblioteca de terceiros do ambiente de simulação;
3. Caso seja verificada alguma função ainda não prevista na biblioteca nativa do ambiente de simulação, ou na biblioteca construída, pode-se desenvolver um novo módulo de biblioteca que implemente a função necessária (ex.: através de blocos funcionais mais simples, via *script*, ou através de *M-Code* ou ainda através de *Black Box*, conforme Apêndice C);
4. Simular o dispositivo no ambiente de simulação, pela integração/interligação (composição funcional) do dispositivo compatível com o gerador de código HDL, visando código genérico, ou de acordo com a família de FPGA a ser especificada;
5. Definir resolução de bits a ser usada no dispositivo;
6. Definir as taxas de *clocks* reais, considerando as especificações e limitações do hardware final ou hardware de desenvolvimento para prova de conceitos, a serem usados de acordo com a taxa de bits e especificações do dispositivo (a ser sintetizado), compatível com os itens 7 e 8;
7. Definir CI de FPGA para o modelo de prova de conceito e/ou versão final;

8. Definir pinos da FPGA a serem usados como entrada de dados, saída de dados, *clock(s)*, controle e de instrumentação, quando necessária;
9. Gerar código *bitstream* usando o compilador adequado a FPGA já definida;
10. Efetuar o *upload* do código para a plataforma de suporte a FPGA;
11. Efetuar co-simulação *single step* ou *free running*. Este passo é necessário apenas no caso da necessidade de instrumentação de co-simulação (quando disponível pelo fornecedor de FPGA);
12. Efetuar emulação em circuito *stand alone* "protótipo" ou modelo de engenharia;
13. Verificar possível otimização do circuito, se possível incluindo *downsizing*, considerando código nativo, no sentido de verificar otimização de recursos e, conseqüentemente, menor consumo de energia do dispositivo alvo (CI para modelo de implementação final);
14. Para o caso de dispositivos a serem embarcados, aplicar redundância via técnicas de Tripla Redundância Modular (do inglês TMR) e verificar possibilidade de dispositivo endurecido para o modelo de voo;
15. Compilar para CI ou CIs de FPGA/ASIC do modelo de produto final ou modelo de voo (para o caso de sistema a ser embarcado).

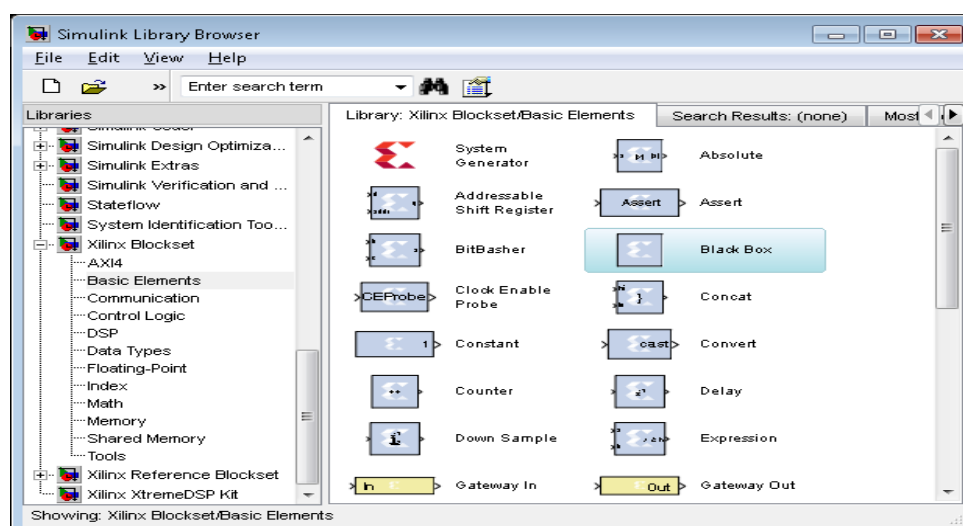
Apêndice C. Desenvolvimento de *Black Box* para o driver de display do Kit ML605

A *Black Box* é um bloco do *software System Generator* da *Xilinx* que permite incorporar modelos de linguagem de programação HDL a um projeto no ambiente de modelagem *Simulink* do *Matlab*.

Apesar das vantagens já descritas na utilização dos blocos funcionais do *System Generator*, algumas aplicações, como *drivers* de controle ou protocolos podem ser desenvolvidas a partir de linguagens de programação. Dessa maneira, a implementação de um protocolo de controle para a escrita em um *display* LCD foi realizada a partir da linguagem VHDL e incorporada ao *System Generator* através de uma *Black Box*.

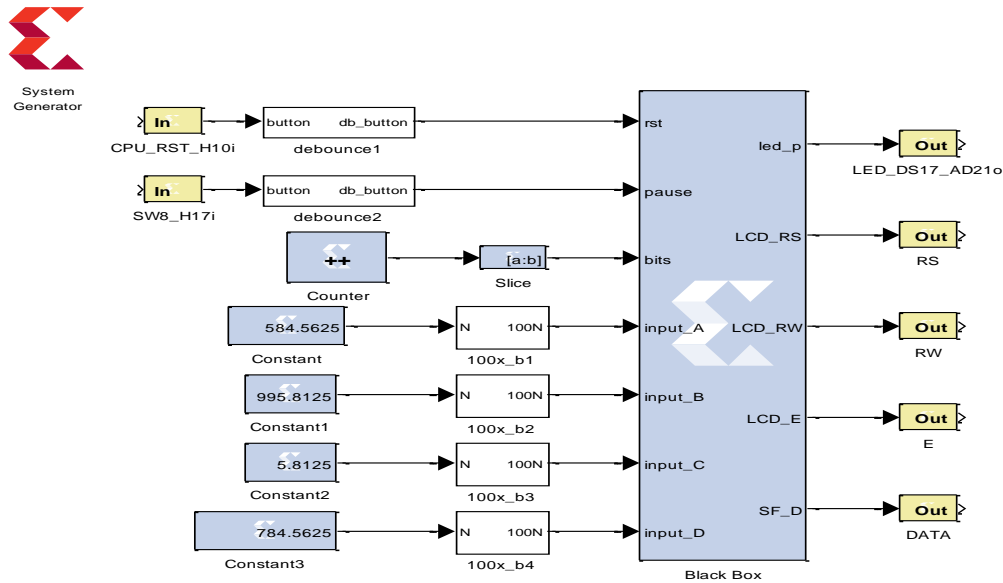
Para utilizar uma *Black Box* deve-se acessar a biblioteca do *System Generator* listada no *Simulink Library Browser* do *Matlab*, na guia *Xilinx Blockset*, conforme apresentada na Figura C.1. A partir dos blocos dessa biblioteca foi implementado o projeto de controle de LCD mostrado na Figura C.2.

Figura C.1: *Xilinx Blockset*.



Fonte: Produção dos autores.

Figura C.2: Projeto de controle de LCD com *Black Box*.



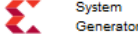
Fonte: Produção dos autores.

A *Black Box* pode ser encontrada nas guias

- *Basic Elements*
- *Control Logic*
- *Floating-Point*
- *Index*

1. Criando um projeto com *Black Box*:

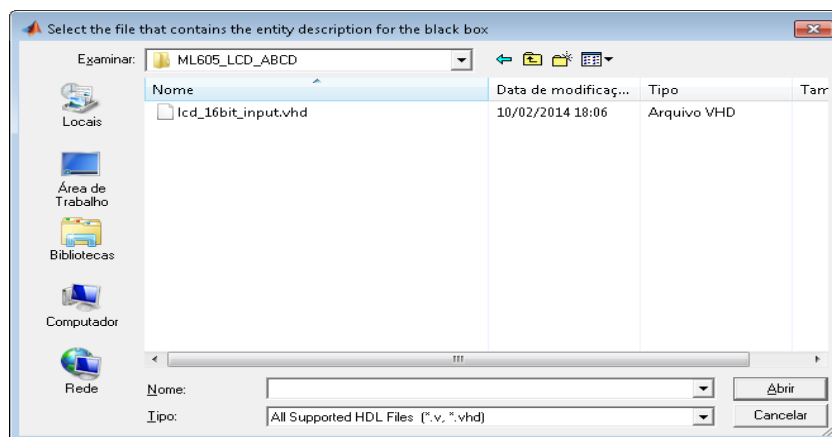
- Criar novo projeto no *Simulink*: File ➔ New ➔ Model
- Salvar o projeto (o nome do projeto não deve coincidir com nenhum outro nome que é reservado para o *System Generator*).

- Copie os arquivos que definem os componentes HDL da *Black Box* para o diretório que contém o modelo.
- Inserir Token *System Generator* 
- Inserir *Black Box*
- Inserir demais blocos

Ao inserir a *Black Box* uma janela aparecerá para a seleção dos arquivos HDL conforme mostrado na Figura C.3.

O arquivo “lcd_16bit_input.vhd” contém o código em linguagem VHDL que implementa o protocolo de comunicação entre o FPGA e o *display* LCD. Esse código divide o *display* em quatro regiões (duas linhas e duas colunas), e possui quatro variáveis de entrada para a escrita de seu conteúdo.

Figura C.3: Seleção de arquivo.



Fonte: Produção dos autores.

Após adicionados os arquivos em HDL, um arquivo de configuração *M-function* associado à *Black Box* é criado automaticamente. No entanto, se houver alguma dependência, de arquivos HDL, para execução da *Black Box*, faz-se

necessário adicionar, no arquivo de configuração *M-function*, a ordem de dependência de chamada dos arquivos como visto na Figura C.4. No exemplo demonstrativo da Figura C.4, “unit2.vhd” é instanciado por “unit1.vhd”.

Figura C.4: Adicionando arquivos com dependência a uma *Black Box*.

```
52  
53  
54     % Add additional source files as needed.  
55     % |-----  
56     % | Add files in the order in which they should be compiled.  
57     % | If two files "a.vhd" and "b.vhd" contain the entities  
58     % | entity_a and entity_b, and entity_a contains a  
59     % | component of type entity_b, the correct sequence of  
60     % | addFile() calls would be:  
61     % |     this_block.addFile('b.vhd');  
62     % |     this_block.addFile('a.vhd');  
63     % |-----  
64  
65     %     this_block.addFile('');  
66     %     this_block.addFile('');  
67  
68 -   this_block.addFile('unit2.vhd');  
69 -   this_block.addFile('unit1.vhd');
```

Fonte: Produção dos autores.