PESQUISADOR LÍDER:
PROF. DR. MARCO ANTONIO ASSIS DE MELO
PESQUISADORES DOCENTES:
PROF. DR. PAULO BATISTA LOPES
PROF. MSC. JOSÉ GOMES GONÇALVES FILHO
PROFA. DRA. YARA MARIA BOTTI MENDES DE OLIVEIRA
PROF. MSC. ANTONIO MARCOS FERRAZ DE CAMPOS
PROFA. DRA. IVANILDA MATILE
PESQUISADOR DISCENTE
VICTOR FIUZA MELLO

SIMULAÇÃO E IMPLEMENTAÇÃO DE UM SISTEMA DE COMUNICAÇÃO OFDM

São Paulo

2011

AGRADECIMENTOS

Ao Engenheiro Nuncio Perrella, da Texas Instruments, que forneceu à Escola de Engenharia da UPM o kit de desenvolvimento que possibilitou a implementação realizada nesta pesquisa.
Ao técnico do Laboratório de Eletrônica da Escola de Engenharia, Francisco Jailson L. Oliveira, pela colaboração na implementação e testes realizadas.

RESUMO

O objetivo desta pesquisa é a criação de uma plataforma de teste e verificação de algoritmos de modulação, demodulação e equalização de sistemas OFDM. Foi também implementado somente o modulador do sistema OFDM. A plataforma foi construída através do uso de processadores digitais de sinais, DSP, cujo software pode ser rapidamente modificado e recarregado nos processadores. Assim, qualquer nova concepção teórica poderá ser prontamente testada e avaliada. Esta plataforma em tempo real possibilitará o estudo e o desenvolvimento tecnológico do sistema OFDM já utilizado na TV Digital, PLC e outros equipamentos de transmissão de dados. Novas pesquisas, artigos e trabalhos técnicos com alunos também serão gerados.

Palavras-chave: HDTV. OFDM. PLC. Processamento de Sinais.

SUMÁRIO

1 INTRODUÇÃO

A tecnologia de modulação utilizando portadoras múltiplas tem se tornado crescentemente e populares devido às suas características de robustez a limitação de freqüência dos canais, maior imunidade ao ruído, maximização da utilização da capacidade do canal e facilidade de implementação (MENEZES; PANAZIO; ROMANO, 2007). Dentre as técnicas de modulação com portadoras múltiplas destaca-se a técnica OFDM (Orthogonal Frequency Division Multiplex). Esta técnica é utilizada em vários sistemas comerciais como, por exemplo, as redes locais sem fio (IEEE 802.11), TV digital (DVB e SBDVD-T), Internet em banda larga (ADSL) e comunicação em linhas de transmissão de eletricidade (PLC), de acordo com Liu e Li (2005).

Não existia na universidade um ambiente de simulação e teste em tempo real de sinais OFDM que possa ser usado para entender o sistema, detectar problemas e testar soluções e novas propostas. Tal ambiente seria constituído por dois transceivers OFDM cujas características pudessem ser programadas por alteração de firmware. Mais especificamente, a implementação de um receptor e um transmissor OFDM em processadores digitais de sinal atingiria o objetivo de servir de “testbed” de trabalhos em modulação com portadoras múltiplas.

A intenção deste projeto é começar a construir uma infra-estrutura que possibilite a pesquisa e desenvolvimento de estudos de tecnologia de modulação com portadoras múltiplas e, em especial OFDM. Os recursos advindos serão empregados em futuros trabalhos de graduação (TGI), pesquisas acadêmicas tais como teses, consultorias externas, disciplinas de cursos regulares ou de extensão, etc, cobrindo tanto aplicações quanto puramente a modulação. Para isso, é realizada a implementação de transceptores OFDM em plataforma de processador digital de sinais.

1.1 OBJETIVOS

O objetivo desta pesquisa é a criação de uma plataforma de implementação, teste e verificação de algoritmos de modulação, demodulação e equalização de sistemas OFDM. Esta plataforma foi construída através do uso de processadores digitais de sinais, cujo software pode ser rapidamente modificado e recarregado. Assim, qualquer nova concepção teórica poderá ser prontamente testada e avaliada. Também será objeto de estudo o

demodulador de sistemas OFDM. Ressalta-se que foi estudado, implementado e simulado somente o modulador.

Um processador digital de sinais (DSP) é um microprocessador cuja arquitetura e o seu conjunto de instruções são otimizados para realizar cálculos numéricos em sinais amostrados com enorme eficiência e rapidez. Atualmente, estes dispositivos são utilizados em várias aplicações práticas (telefones celulares, tocadores de MP3, modems, sistemas de controle, etc.). Como são dispositivos programáveis, possui flexibilidade suficiente para servirem de base para a plataforma aqui proposta.

Inicialmente, o firmware carregado nesta plataforma constituiu-se de um transmissor. Futuras pesquisas serão levadas a cabo através de modificações de blocos deste firmware. Por isso, este programa deve ser estruturado e particionado de forma muito inteligente.

1.2 METODOLOGIA

O diagrama de blocos do sistema OFDM é mostrado na figura 1, na qual os blocos a serem implementados em firmware, no DSP, estão assinalados em azul e os blocos de hardware estão em vermelho, baseados em [3].

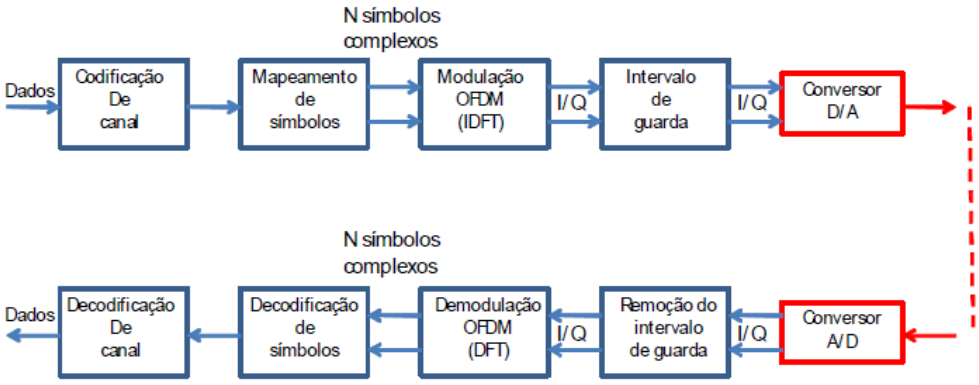


Figura 1: Diagrama de blocos de um sistema OFDM

Salienta-se que a implementação realizada refere-se somente aos blocos em azul do modulador (parte superior da figura, a menos da codificação de canal e do de intervalo de guarda).

Após esta verificação, foi realizada uma transcrição do programa em ambiente MATLAB para a linguagem C no qual foi testada no hardware do Módulo de Avaliação DSP

TMS320C6713 da Texas Instruments. Esta empresa comercializa este Módulo de Avaliação DSP que pode ser empregado no desenvolvimento de aplicativos de tempo real.

Figura 2: Módulo de Avaliação de hardware do DSP TMS320C6713 da Texas Instruments.

A próxima fase da pesquisa consistiu no projeto de um sistema eletrônico para servir de base de desenvolvimento contendo condicionamento de sinais e conexões para os equipamentos geradores de dados e os equipamentos de medidas elétricas. O dispositivo concebido foi montado em conjunto com o Módulo de Avaliação de hardware do DSP TMS320C6713, (EVM) e sua memória de programa foi programada através de uma interface JTAG para que novos algoritmos possam ser prontamente testados.

Considerando o exposto acima, esta pesquisa foi realizada em 5 etapas:

a) Simulação dos algoritmos em MATLAB
b) Transcrição dos algoritmos para linguagem C
c) Implementação do Projeto no módulo DSP TMS320C6713
d) Projeto de um sistema eletrônico para servir de base de desenvolvimento.
e) Integração e documentação

São apresentados, em arquivos anexos, os programas fontes em Matlab e em linguagem C.

2 REFERENCIAL TEÓRICO

Nesta seção, os princípios básicos da modulação OFDM são explicados, assim como as técnicas utilizadas para a sua transmissão e recepção.

2.1 ORTHOGONAL FREQUENCY DIVISION MULTIPLEX (OFDM)

Esquemas de modulação utilizando portadoras múltiplas tem se tornado crescentemente populares devido às suas características de robustez a limitação de freqüência dos canais, maximização da utilização da capacidade do canal e facilidade de implementação (MENEZES, 2007). Dentre as técnicas de modulação com portadoras múltiplas destaca-se o *Orthogonal Frequency Division Multiplex* (OFDM), utilizado em vários sistemas comerciais como, por exemplo, as redes locais sem fio (IEEE 802.11), TV digital (DVB e SBDVD-T), Internet em banda larga (ADSL) e comunicação em linhas de transmissão de eletricidade, PLC. (LIU, 2005)

O conceito de OFDM baseia-se na idéia de dividir a mensagem digital a ser transmitida em pequenos grupos de bits e empregar portadoras diferentes para modular cada um destes grupos. O conjunto de portadoras deverá ser ortogonal, ou seja, os pontos de máximo de uma determinada portadora deve coincidir com a passagem por zero das outras. O símbolo OFDM é constituído pela soma dos grupos de bits modulados nas freqüências das portadoras ortogonais.

A vantagem da técnica OFDM pode ser entendida se considerarmos o sinal transmitido como sendo a união de vários sub-canais de banda estreita que se propagam em um canal não ideal com largura de banda mais larga. Como as não-idealidades (interferências, limitação de banda, entre outras) do canal se tornam amenizadas em cada faixa de um dos sub-canais, o resultado geral é melhor do que se implemente um sinal banda larga for enviado.

2.1.1 Ortogonalidade:

O conceito de ortogonalidade é derivado diretamente de Álgebra Linear. Considere-se o espaço linear que contém todas as portadoras senoidais expressas como apresentado na equação 1:

$$s_k(t) = \cos(2\pi f_k t), k = 0,1,2,...,N-1 \quad (1)$$

Na equação 1, f_k é a *k*-ésima portadora, expressa em Hertz, e, também, a frequência central do sinal transmitido no *k*-ésimo sub-canal do OFDM. Admitindo-se que cada portadora possa sofrer um desvio de fase igual a ϕ_k, dois sinais s_k serão ditos ortogonais se a condição expressa na equação 2 for satisfeita:

$$\int_0^T \cos(2\pi f_k t + \phi_k)\cos(2\pi f_i t + \phi_i)dt = 0 \qquad (2)$$

Uma das formas de satisfazer esta condição é impor as seguintes restrições ao nosso sinal:

a) T é feita igual a taxa de símbolos em cada sub-canal

b) A separação entre as freqüências fk é feita igual a um múltiplo do inverso de T, isto é $f_k - f_i = n/T$, para n = 1, 2,...., N-1

Se estas condições forem satisfeitas o somatório dos sinais em cada sub-canal formará o sinal OFDM.

Observe-se, ainda, que nada foi dito sobre a modulação em cada um dos sub-canais. Portanto, estes sinais de banda estreita poderão ser modulados da forma que mais convier aos projetistas (QAM, PSK, BPSK, FM, etc), desde que as condições mencionadas acima sejam satisfeitas.

O gráfico 1 ilustra este conceito de sinal OFDM ser constituído por múltiplos sinais em portadoras ortogonais:

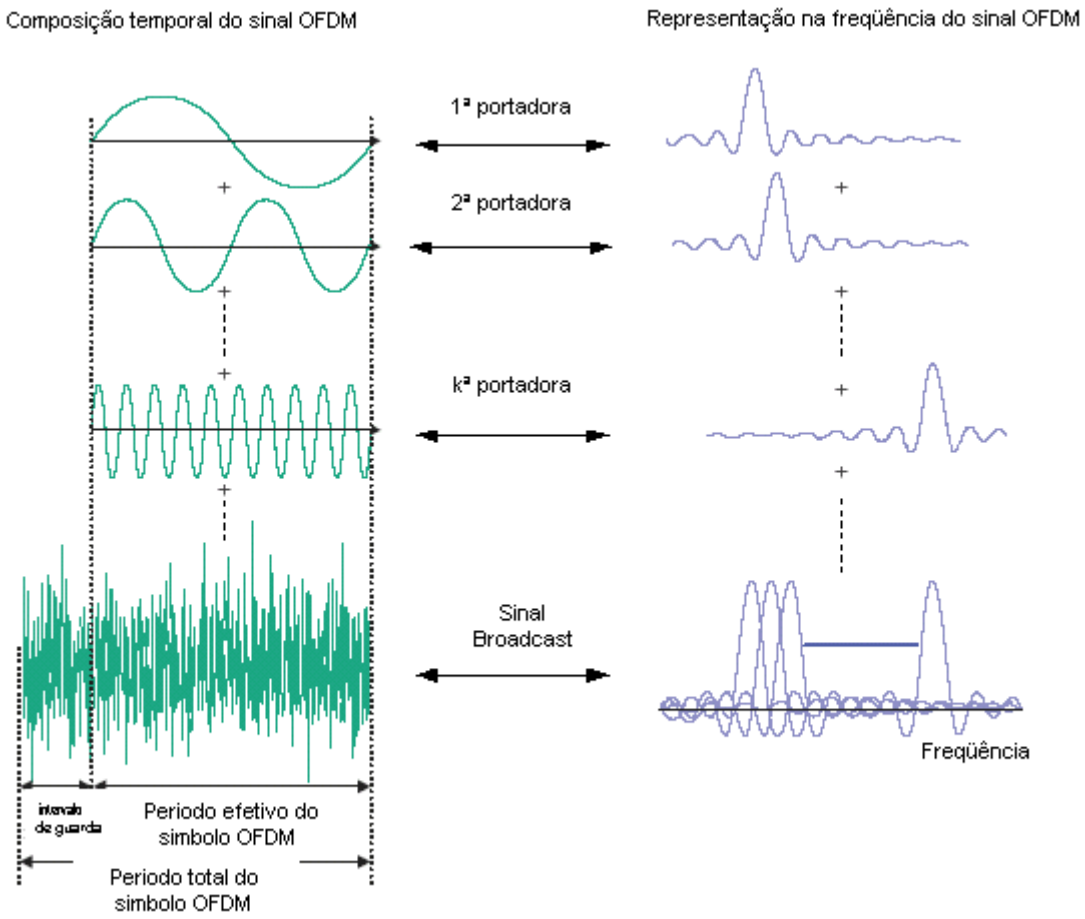


Gráfico 1: Estrutura do sinal OFDM
Fonte: Dias (2006)

2.1.2 Modulação e demodulação OFDM:

Os princípios fundamentais de um modulador OFDM podem ser discutidos com base no diagrama 1. Os dados a ser transmitidos são fornecidos ao modulador de forma serial em grande velocidade. O primeiro bloco, um conversor serial-paralelo, divide estes bits em grupos que são entregues aos moduladores em banda base. Estes moduladores codificam o sinal de acordo com a modulação a ser empregada em cada sub-canal (QAM, PSK, BPSK, etc). Cada um dos sinais em banda-base é, então, modulado pela respectiva portadora, cuja frequência está de acordo com as condições expressas na seção 2.2. A soma de todos estes sinais em banda-estreita é o sinal OFDM. O diagrama de blocos (diagrama1) apresenta um modulador OFDM.

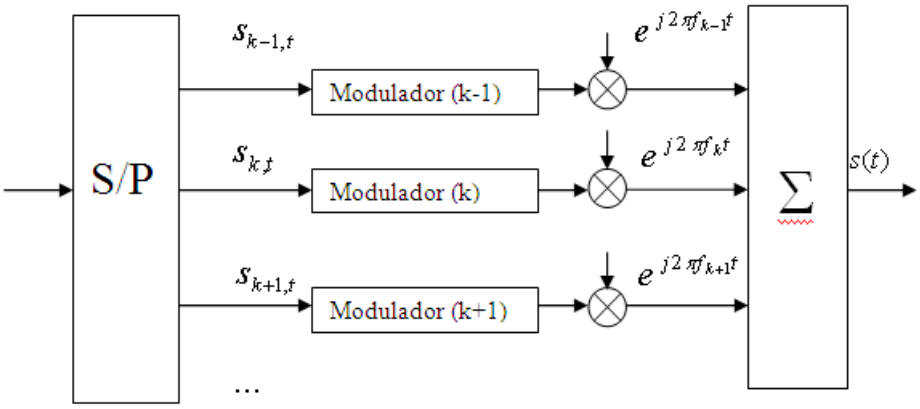


Diagrama 1: Diagrama de blocos de um modulador OFDM simplificado
Fonte: Baseado em Proakis e Salehi (2008)

Em um sistema com N subportadoras, a taxa de símbolos é reduzida por um fator igual a N em comparação com um esquema de modulação com portadora única que ocupasse toda a banda de frequência disponível e transmitisse a mesma quantidade de informação que o sistema OFDM.

A título de exemplo, considere-se que cada sub-canal utiliza a modulação QAM. Assim, este sinal pode ser expresso de acordo com a equação 3:

$$u_k(t) = \mathrm{Re}\left[\sqrt{\frac{2}{T}} A_k e^{j2\pi f_k t}\right] = \mathrm{Re}\left[\sqrt{\frac{2}{T}} X_k e^{j2\pi f_k t}\right] \quad (3)$$

Na equação 3, X_k é o ponto da constelação QAM transmitido pela *k*-ésima portadora.

Se o número de portadoras, N, for suficientemente grande, cada sub-canal disporá de uma banda de freqüências suficientemente estreita para que sua resposta possa ser considerado um valor complexo, $C(f_k)$, dependente da freqüência, expresso como indicado na equação 4:

$$C(f_k) = C_k = |C_k| e^{j\phi_k} \quad (4)$$

Portanto, o sinal recebido no *k*-ésimo sub-canal é dado pela equação 5:

$$r_k(t) = \mathrm{Re}\left[\sqrt{\frac{2}{T}} C_k X_k e^{j2\pi f_k t}\right] + n_k(t) \quad (5)$$

O último termo representa o ruído aditivo presente no sub-canal que é considerado como gaussiano com média nula neste documento.

Uma consideração importante é o fato de que tanto $|C_k|$ quanto X_k podem ser estimados através do envio da portadora não modulada e, portanto, são conhecidos pelo receptor.

A demodulação do sinal recebido na *k*-ésima portadora pode ser feita pela correlação cruzada entre o sinal recebido, $r_k(t)$, com as duas funções determinadas a partir do conhecimento do canal, indicadas pelas equações 6 e 7 respectivamente.

$$\Psi_{k1(t)} = \sqrt{\frac{2}{T}} \cos\left(\pi f_k t + \phi_k\right) \tag{6}$$

$$\Psi_{k2(t)} = \sqrt{\frac{2}{T}} sen\left(\pi f_k t + \phi_k\right) \tag{7}$$

E, amostrando-se a saída dos correlacionadores no instante t = T, o sinal recuperado é dado pela equação 8.

$$Y_k = |C_k| X_k + n_k \tag{8}$$

O efeito do canal é removido dividindo-se este sinal por $|C_k|$ (obtido através do envio da portadora não modulada).

Finalmente, um circuito de decisão decide qual ponto da constelação QAM foi enviado, computando a distância euclidiana entre o sinal recebido e os pontos ideais.

2.1.3 OFDM e a Transformada Rápida de Fourier

A técnica de demodulação explicada na seção anterior requer o emprego de 2 correlacionadores ou filtros casados por sub-canal. Utilizando a Transformada Rápida de Fourier, o esforço computacional de modulação e demodulação pode ser reduzido consideravelmente. Para isso, as definições da Transformada Discreta de Fourier (TDF) e da Transformada Discreta de Fourier Inversa (TDFI) devem ser consideradas, como indicado nas equações 9 e 10, respectivamente.

$$TDF: X(k) = \sum_{n=0}^{N-1} x(n) e^{-j(2\pi/N)nk}, k = 0,1,\ldots,N-1 \tag{9}$$

$$TDFI: x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j(2\pi/N)nk}, n = 0,1,\ldots,N-1 \quad (10)$$

O exame destas duas equações revela que tanto a TDF e a TDFI são semelhantes à modulação em freqüência. Consequentemente, os algoritmos altamente eficientes de cálculo destas transformadas, coletivamente chamados de Transformada Rápida de Fourier (TRF), podem ser utilizados para a implementação de um sistema de transmissão de sinais OFDM, conforme os diagramas 2 e 3.

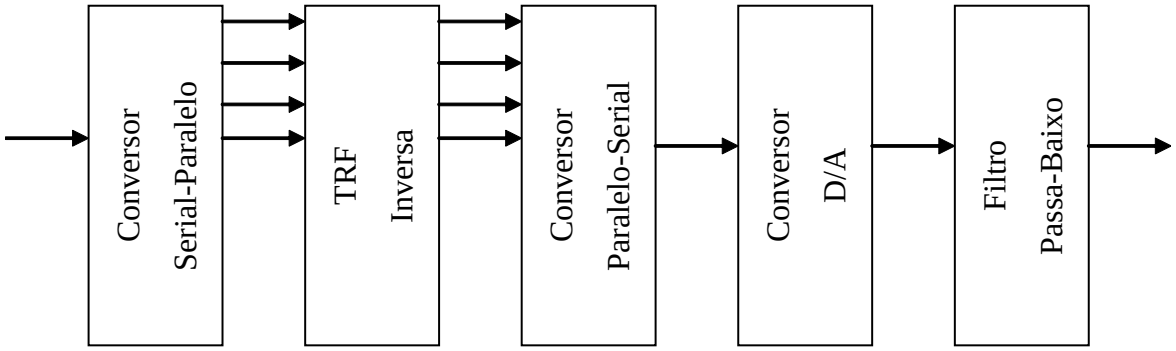


Diagrama 2: Transmissor OFDM utilizando a TRF inversa
Fonte: Adaptado de Proakis e Salehi (2008, p. 750)

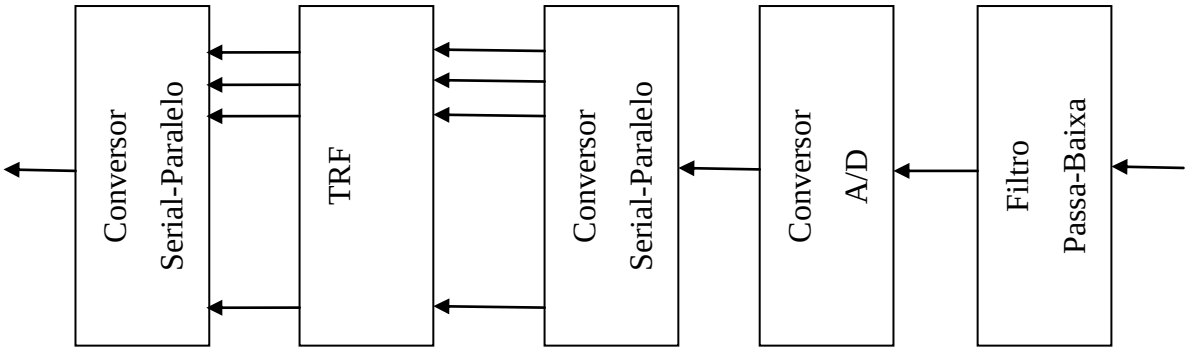


Diagrama 3: Receptor OFDM utilizando a TRF
Fonte: Adaptado de Proakis e Salehi (2008, p. 750)

2.2 O MICROPROCESSADOR DSP (DIGITAL SIGNAL PROCESSOR).

Os microprocessadores DSP são projetados para otimizar a execução de produtos e imediata soma ou acumulação. A multiplicação e subseqüente adição são feitas em *hardware* ao invés do microcódigo dos microprocessadores de uso geral. Isso reduz o tempo

de execução para o mínimo de um ciclo (geralmente 1 clock). Os microprocessadores de uso geral gastam diversos ciclos

A grande maioria dos algoritmos de Processamento Digital de Sinais é executada através de operações matemáticas do tipo soma de produtos (SOP – Sum Of Products). Exemplos claros desta observação são os filtros digitais e a transformada discreta de Fourier. Consequentemente, para a implementação eficiente destes algoritmos, são necessários sistemas (hardware ou software) que calculem esta acumulação de produtos muito rapidamente.

Os processadores digitais de sinais (DSPs) são microprocessadores cuja arquitetura e conjunto de instruções são projetados para otimizar a execução de produtos e imediata soma ou acumulação. Rotineiramente, estes DSPs agregam um unidade em hardware capaz de realizar a operação de multiplicação e acumulação em 1 ou menos ciclo de máquina (neste caso, o DSP deve trabalhar em um esquema de pipeline e realizar esta operação várias vezes em sequência). Adicionalmente, os DSPs devem ser capazes de sustentar transferências de dados a uma taxa suficientemente elevada para que as operações de multiplicação-acumulação não sejam paralisadas ou se tornem lentas.

Neste projeto, o DSP utilizado foi o TMS320C6713 da Texas Instruments, um dispositivo em ponto flutuante (TEXAS INSTRUMENTS. 2006). Nas seções seguintes, o funcionamento deste processador será examinado.

Os elementos do processador ligados a registradores de controle (Control registers), lógica de controle (Control logic), teste (Test), emulação (Emulation), interrupção (Interrupts) e decodificação de instruções (Instruction decode) são comuns a todos microprocessadores, não importando se DSP ou de uso geral. Pela complexidade de detalhes e pela pouca importância para a compreensão da arquitetura do DSP em questão, eles não serão descritos neste relatório que focará os aspectos que realmente diferenciam um DSP de um processador de uso geral e que suportam o desempenho numérico daqueles.

2.2.1 A arquitetura da família TMS320C6000.

Este projeto foi desenvolvido na placa de desenvolvimento *Spectrum Digital* TMS320C6713 DSK (DSP Starter Kit) existente no laboratório da Escola de Engenharia da Universidade Presbiteriana Mackenzie. A placa, doravante chamada DSK, possui um microprocessador DSP TMS320C6713, uns dos diversos integrantes da família TMS320C6000.

O diagrama em blocos de um DSP genérico é mostrado no diagrama 4. Na verdade, este diagrama não difere do diagrama equivalente para um microprocessador de uso geral.

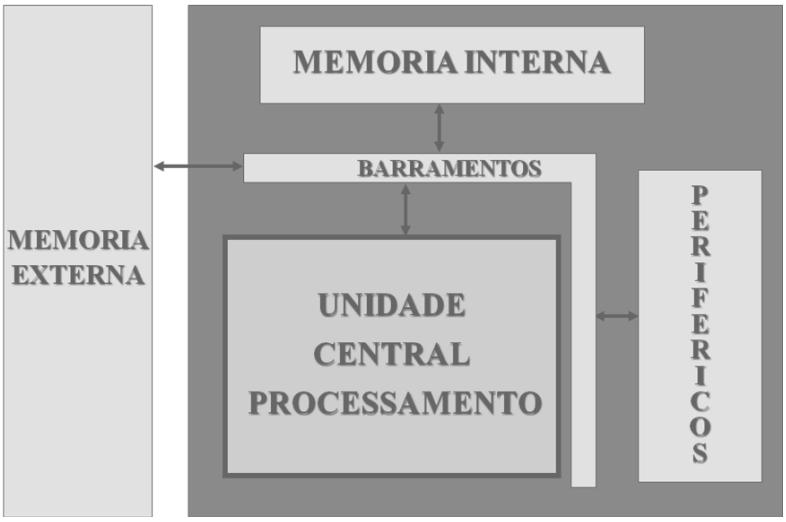


Diagrama 4 – Diagrama em blocos
Fonte: Adaptado TEXAS INSTRUMENTS (2006)

Onde está, então, a diferença? Nos detalhes de desempenho e recursos. Na descrição que segue as principais diferenças entre um DSP e um microprocessador de uso geral serão abordadas.

O diagrama 5 a seguir é o diagrama em blocos da família TMS320C67xx. A unidade central de processamento (CPU) é constituída por 8 unidades funcionais capazes de operarem simultaneamente. As instruções de programa deste processador são formadas pela concatenação de palavras de comando para cada uma das unidades residentes na CPU. Este tipo de arquitetura é chamada de VLIW, pela sigla em Inglês *Very Long Instruction Word* (palavra de instrução muito extensa). Adicionalmente, cada uma das unidades que compõem a CPU são capazes de realizarem operações de multiplicação e acumulação em palavras codificadas em ponto flutuante. Observe-se que no barramento de 256 bits podem trafegar até 8 instruções de 32 bits (256=8*32). Portanto podem ser acessadas até 8 instruções simultaneamente. (TEXAS INSTRUMENTS (2006) .

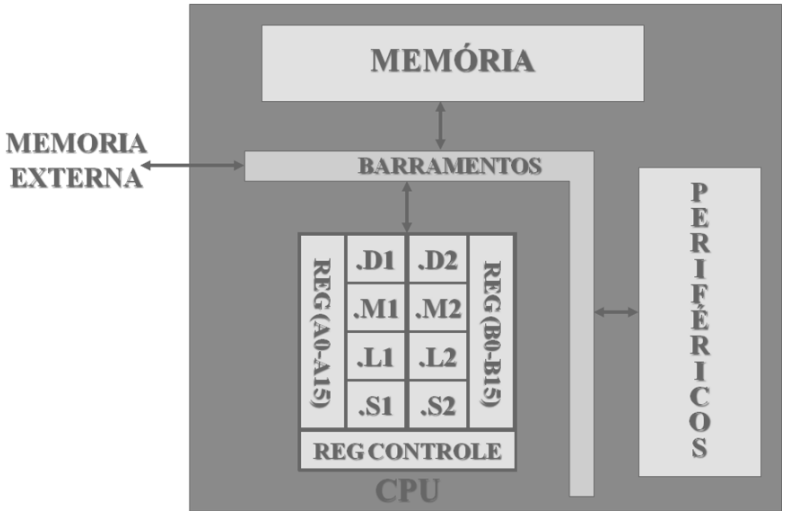


Diagrama 5 - Diagrama em blocos da família TMS320C67xx destacando CPU
Fonte: Adaptado de Texas Instruments (2006).

Um olhar mais atento ao diagrama da figura 2 acima nos revela a existência de 2 grupos independentes de registradores (memórias de uso geral – *Register File* A e B) de 32 bits de tamanho de palavra, totalizando 16 posições independentes (A0-A15 e B0-B15) em cada grupo. Oito unidades funcionais independentes (4 para cada Data Path) também são mostradas:

a) 2 multiplicadores (.M1 e .M2);
b) 2 ALU (.L1 e .L2) – soma e subtração apenas;
c) 2 ALU (.S1 e .S2) – soma, subtração, desvios e manipulação de bits;
d) 2 ALU (.D1 e .D2) – soma, subtração e transferência de dados.

Existem 8 unidades funcionais independentes e o DSP pode buscar 8 instruções simultaneamente. Mas somente quando cada instrução se referir a uma unidade diferente e que os dados necessários estejam disponíveis o DSP trabalhará de acordo com a sua máxima capacidade de processamento. Não é possível executar uma instrução onde pelo menos um dado de entrada seja um resultado, ou saída, de outra instrução executada simultaneamente (TEXAS INSTRUMENTS, 2006). Para atenuar esta característica, estas unidades funcionais estão separadas em grupos de 4 e cada grupo está ligado a barramentos específicos chamados de *datapath.* O uso de *datapaths* adequados permite otimizar o desempenho do processador em termos de velocidade de processamento.

Isso faz refletir sobre um aspecto fundamental: a ordem de execução das instruções pode, e deve, ser cuidadosamente analisada e modificada quando possível para garantir o máximo de eficiência na execução do programa. Felizmente, a própria fabricante deste circuito integrado oferece compiladores inteligentes que, a partir de um programa em

linguagem de alto nível (em geral, C ou C++), realizam a alocação dos recursos de hardware para otimizar a execução.

Outra característica importante comum a praticamente todos os DSPs pode ser inferida a partir do diagrama 6 abaixo. Nela, o espaço reservado a memória de dados é separado do espaço reservado a memória de programa. Este tipo de arquitetura, denominada Harvard, torna possível o acesso simultâneo ao programa a ser executado e ao conjunto de dados a serem processados. Desta maneira, a execução de uma série de instruções não é intercalada com os ciclos de busca das próprias instruções. Adicionalmente, a separação dos espaços de memória permite a implementação de uma arquitetura em pipeline, na qual várias instruções podem ser executadas ao mesmo tempo.

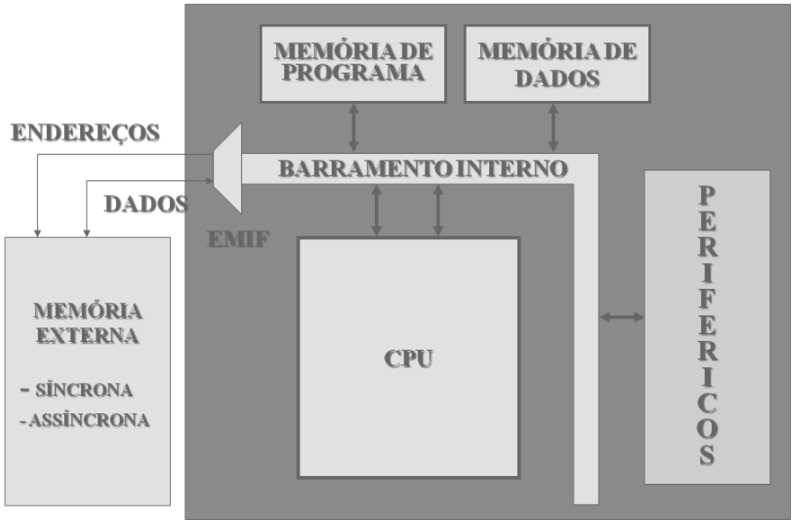


Diagrama 6 - Acesso a memória
Fonte: Texas Instruments (2006)

O *pipeline* pode ser mais bem entendido se cada instrução for desmembrada em três fases: busca (também chamada de *fetch* por causa da terminologia em Inglês), decodificação e execução.

Na primeira fase, a instrução é acessada na memória de programa e transferida para uma unidade decodificadora. Como os DSPs empregam a arquitetura Harvard, a busca da instrução pode ser feita de forma concomitante com as outras fases sem prejuízo da velocidade de operação. Adicionalmente, a busca da instrução também é dividida em 4 subfases (que serão referenciadas por siglas derivadas de expressões em língua inglesa) :

a) PG: Geração do endereço ma memória de programa;
b) PS: Ativação do endereço no barramento;
c) PW: Espera pela resposta do bloco de memória;

d) PR: Recepção da palavra de instrução pela CPU.

A decodificação é a etapa na qual a instrução é interpretada para que seja executada corretamente. Esta fase é composta por 2 operações distintas:

a) DP: a instrução é desmembrada em instruções para cada unidade funcional;

b) DC: os operandos de origem e de destino, assim como os recursos de hardware são alocados para cada unidade funcional.

Na etapa de execução, os operandos são transferidos para a unidade funcional apropriada, modificados de acordo com a instrução e transferidos para as posições de memória ou registros assinalados. Para implementação desta fase, a máquina de estado executa até 10 subfases denominadas E1 a E10. Diferentes instruções requerem um número diferente de subfases.

Como a disponibilidade dos recursos de hardware permite que estas subfases sejam ativadas simultaneamente, várias instruções podem ser trabalhadas concomitantemente. Esta situação é chamada de *pipeline* de execução e é ilustrada no quadro 1 abaixo. É exatamente este fluxo de instruções que permite que as operações de multiplicação e acumulação sejam executadas a taxas maiores que a freqüência de operação do circuito integrado.

	Subfase															
Instrução	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
n	PG	PS	PW	PR	DP	DC	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	
n+1		PG	PS	PW	PR	DP	DC	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10
n+2			PG	PS	PW	PR	DP	DC	E1	E2	E3	E4	E5	E6	E7	E8	E9
n+3				PG	PS	PW	PR	DP	DC	E1	E2	E3	E4	E5	E6	E7	E8
n+4					PG	PS	PW	PR	DP	DC	E1	E2	E3	E4	E5	E6	E7
n+5						PG	PS	PW	PR	DP	DC	E1	E2	E3	E4	E5	E6
n+6							PG	PS	PW	PR	DP	DC	E1	E2	E3	E4	E5
n+7								PG	PS	PW	PR	DP	DC	E1	E2	E3	E4
n+8									PG	PS	PW	PR	DP	DC	E1	E2	E3
n+9										PG	PS	PW	PR	DP	DC	E1	E2
n+10											PG	PS	PW	PR	DP	DC	E1

Quadro 1 – Pipeline de instrução
Fonte: Texas Instruments (2006)

Graças às características descritas acima, o TMS320C6713B, operando a partir de um relógio de 300 MHz, executa até 1800 milhões de operações em ponto flutuante por segundo e até 600 milhões de multiplicações-acumulações por segundo.

2.2.2 Organização da memória:

Um alto desempenho do núcleo de processamento só é possível quando a organização de memória do DSP possibilite a transferência de dados a uma taxa que sustente

a operação contínua e rápida da CPU. Por esta razão, a organização de memória tanto TMS320C6713 quanto da placa DSK deve ser examinada em detalhe. Os mapas de endereçamento tanto do processador quanto da placa utilizada neste projeto estão ilustradas na diagrama 5 abaixo. O espaço de endereçamento é acessado via um barramento de 32 bits. (TEXAS INSTRUMENTS, 2006).

Diagrama 7 – Mapa de memória do TMS320C6713 e da placa DSK
Fonte: Spectrum Digital (2004).

A memória interna do TMS320C6713 é organizada em espaços separados para programa e dados. Quando memória externa é utilizada, estes espaços são unificados e acessados via a interface de memória externa (EMIF). Os DSPs da família TMS320C67xx possuem dois barramentos para acesso a memória de dados interna e um barramento para acesso a memória de programa interna.

Para acelerar ainda mais o processamento, este DSP ainda utiliza memória cachê tanto na memória de programa (L1 com 4 kB) quanto na memória de dados (L1 com 4 kB e L2 com 256 kB).

Embora exista uma quantidade considerável de memória interna, para dados e programa, dificilmente ela atenderá toda e qualquer aplicação.

A possível expansão de memória deverá ser externa e terá, necessariamente, que se conectar ao barramento interno para que os dados e/ou instruções cheguem ao seu destino.

Para que o desempenho do DSP não seja comprometido pelo acesso externo, sempre mais lento, uma parte considerável do esforço de projeto do chip e da área de silício interna é dedicada a esse acesso (TEXAS INSTRUMENTS, 2006).

O bloco identificado por *External Memory Interface* (EMIF) é responsável pelo acesso à memória externa. (TEXAS INSTRUMENTS, 2006). Algumas características relevantes do EMIF são:

a) acesso a memória síncrona (SDRAM) ou assíncrona;

b) acesso direto sem a necessidade de chips de lógica externos (*glueless access*);

c) possibilidade de utilizar memória RAM dinâmica do tipo empregado nos PC´s (PC100/PC133);

d) acesso em 16 ou 32 bits sempre. E 64 bytes apenas para alguns integrantes da família;

e) capacidade máxima teórica de acesso: 4 GB.

2.2.3 Periféricos.

Os periféricos de entrada/saída, ou I/O (input/output), devem proporcionar a maior taxa de transferência de dados possível.

Mas isso nem sempre ocorre, pois algumas portas de I/O respeitam padrões conhecidos e consolidados. Por exemplo, a interface descrita adiante, PCI, é um dos mais divulgados e utilizados padrões. É o conhecido barramento (slot) de microcomputadores PC IBM. Os dados trafegam em 32 bits e a uma velocidade de 33 ou 66 MHz. Mesmo que o DSP possa ir muito além disso, a velocidade, e a norma, é respeitada.

No diagrama 8 a seguir são apresentados, resumidamente, os periféricos disponíveis na família TMS320C60000.

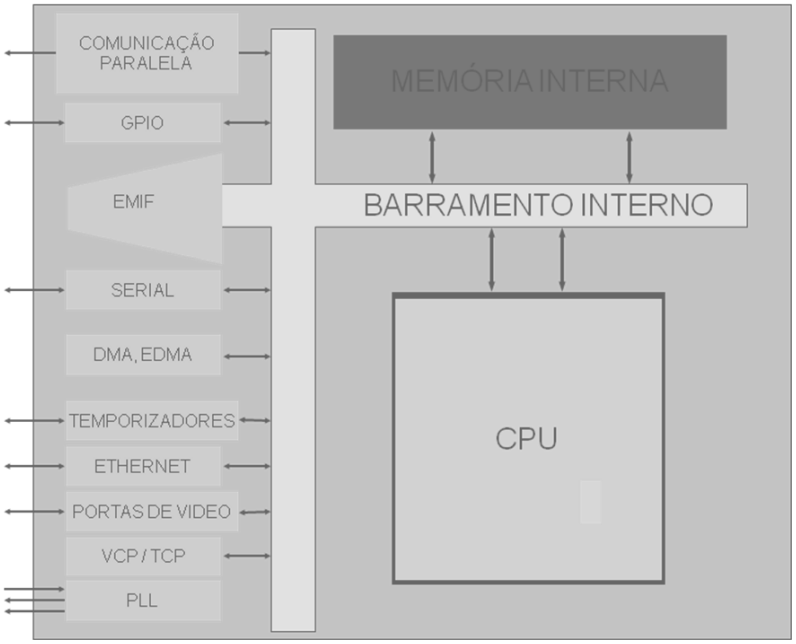


Diagrama 8 - Periféricos da família
Fonte: Texas Instruments (2006).

Devido ao número de dispositivos periféricos e aos diferentes modos de operação de cada um deles, torna-se inviável descrevê-los em detalhes. Segue-se, então, um resumo de suas principais características:

a) Comunicação paralela padrão: suporta barramentos padrão como HPI, XBUS e PCI 33 ou PCI 66;

b) GPIO (*General Purpose Input Output*) barramento genérico, nesta família em 8 e 16 bits, especificado apenas pelo fabricante para entrada e saída de dados. Permite que o projetista, geralmente com auxílio de componentes externos, interligue os mais variados tipos de circuitos e interfaces externas. Embora pareça limitado por ser de 8 ou 16 bits é extremamente rápido;

c) Comunicação serial: implementada por três diferentes blocos: o McBSP (*Multi-Channel Buffered Serial Port*), que realiza comunicação serial geral de alto desempenho, o McASP (*Multi-Channel Audio Serial Port*), empregada para comunicação serial com dispositivos de áudio (até 8 canais estéreo), e UTOPIA, utilizada para conexão em sistemas de comunicação ATM;

d) DMA (*Direct Memory Access*): usada para transferência rápida de blocos de memória por hardware interno sem envolver execução de instruções e independente da CPU.

e) Temporizadores e contadores: o DSP incorpora dois ou três temporizadores ou contadores de 32 bits com capacidade para gerar interrupções e empregando pinos de entrada (*clock*) e saída independentes;

f) Ethernet: Porta padrão Ethernet de 10/100 Mbps com possibilidade de execução de protocolo TCP/IP;

g) Portas de vídeo: podem ser configuradas para entrada (captura) ou saída (vídeo);

h) VCP/TCP: TCP (Turbo Coprocessador) e VTB (Viterbi Coprocessador) são blocos aceleradores que suportam aplicações de comunicação de dados (TCP) e canais de voz (VTC) para telefonia móvel;

i) PLL (*Phase Locked Loop):* para geração/multiplicação de *clock* com ampla gama de recursos.

3 METODOLOGIA E DESENVOLVIMENTO DA PESQUISA

Nesta seção, são descritas tanto a metodologia geral de desenvolvimento com Processadores Digitais de Sinais quanto a sua aplicação no desenvolvimento desta pesquisa, de forma a possibilitar a execução do objetivo específico, o sistema OFDM.

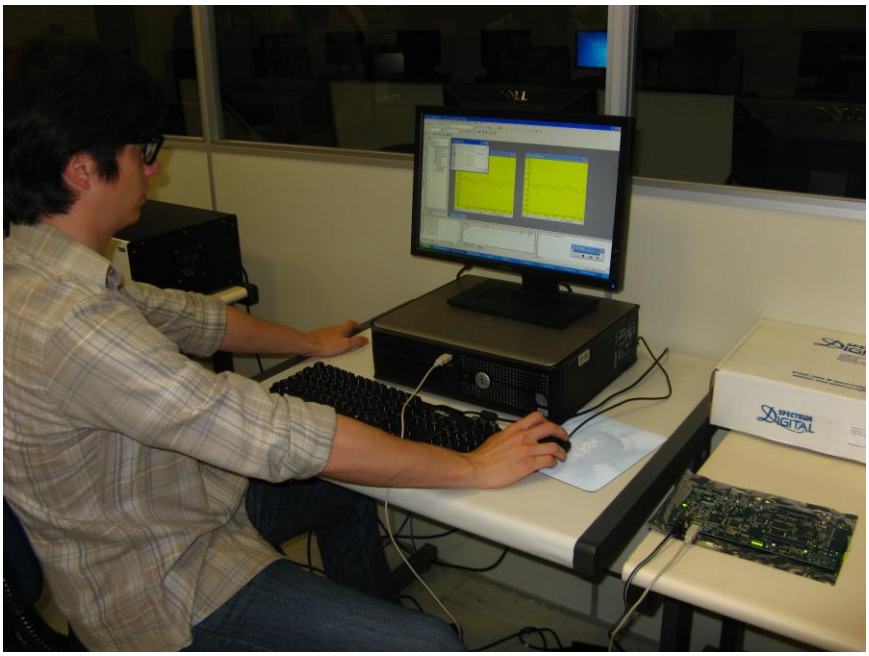
3.1 O AMBIENTE DE DESENVOLVIMENTO:

Para a compreensão da metodologia utilizada nesta pesquisa, é necessária uma certa familiarização tanto com o sistema de desenvolvimento fornecido pela Texas Instruments quanto com a sistemática de projeto com Processadores Digitais de Sinais.

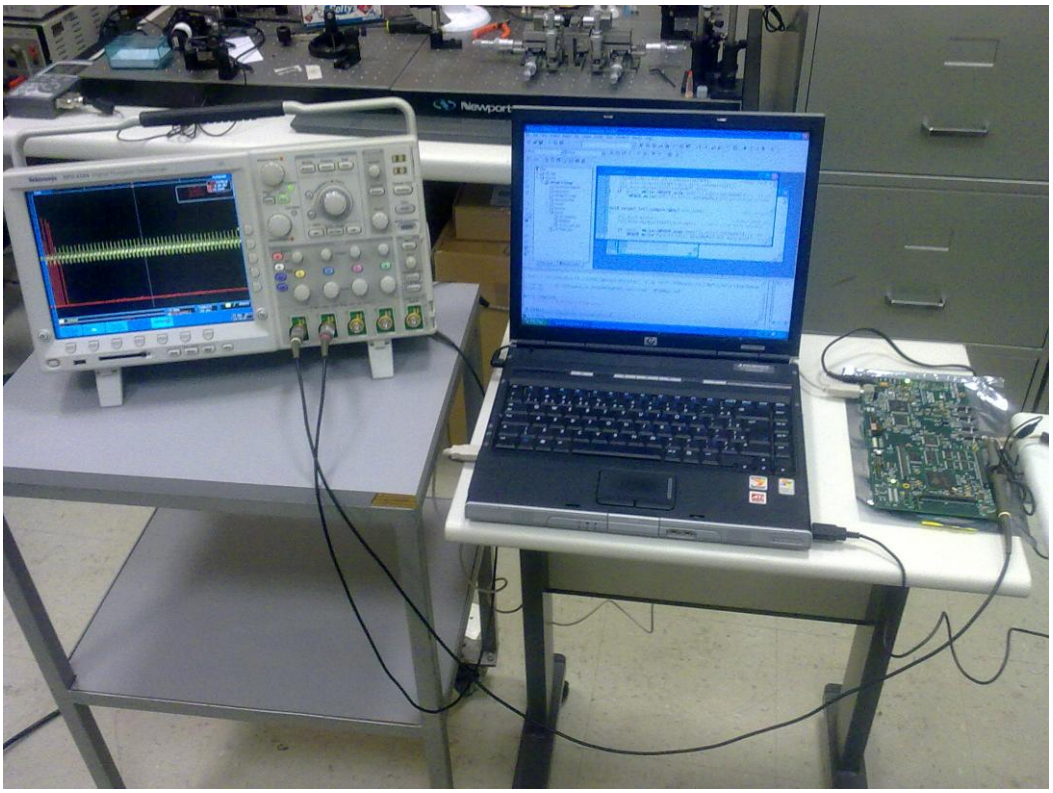
Infelizmente, o kit de desenvolvimento *Spectrum Digital* TMS320C6713 DSK (DSP Starter Kit) solicitado para esta pesquisa não chegou a ser adquirido pela instituição por problemas internos. Todavia, o escritório local da Texas Instruments doou 3 conjuntos de placas para que este projeto pudesse ser realizado.

As fotografias 1 a 4 mostram o ambiente de desenvolvimento utilizado no projeto.

Fotografia 1. Placa KIT de desenvolvimento DSK TMS320C6713.



Fotografia 2. Sistema de desenvolvimento usando o KIT de desenvolvimento DSK TMS320C6713, no Laboratório de Processamento de Sinais e Sistemas de Controle da EE Mackenzie.



Fotografia 3. Sistema de desenvolvimento usando o KIT de desenvolvimento DSK TMS320C6713, acoplado no osciloscópio digital, no Laboratório de Processamento de Sinais e Sistemas de Controle da EE Mackenzie.

O diagrama 9 mostra o diagrama de blocos do KIT de desenvolvimento DSK TMS320C6713.

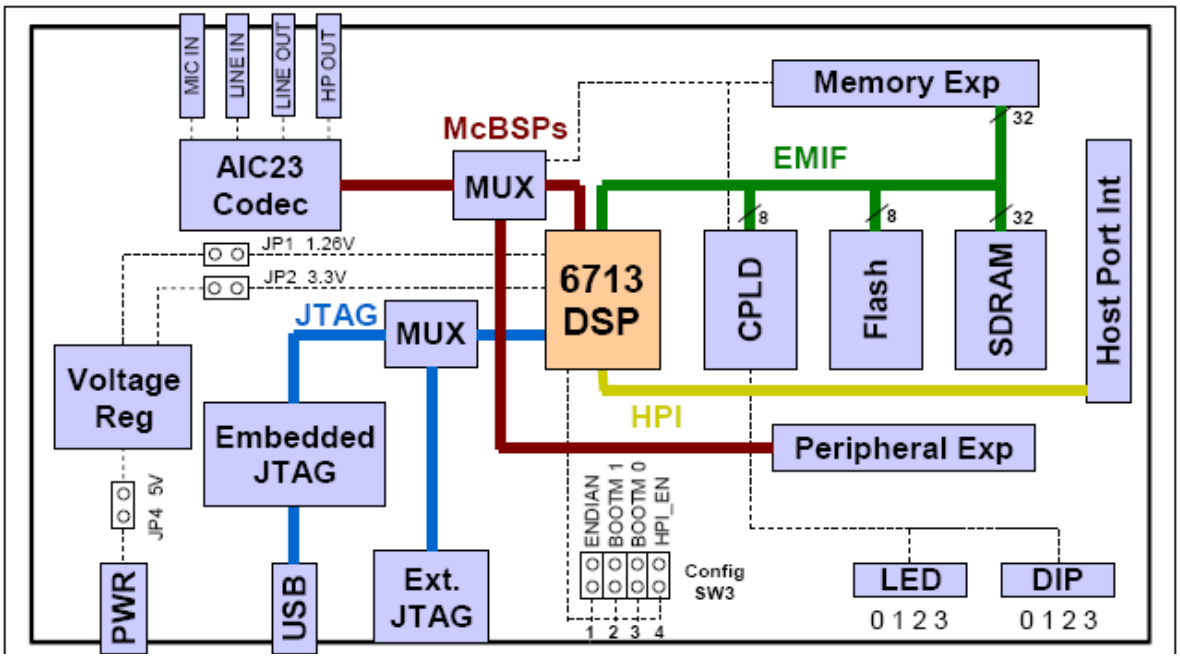
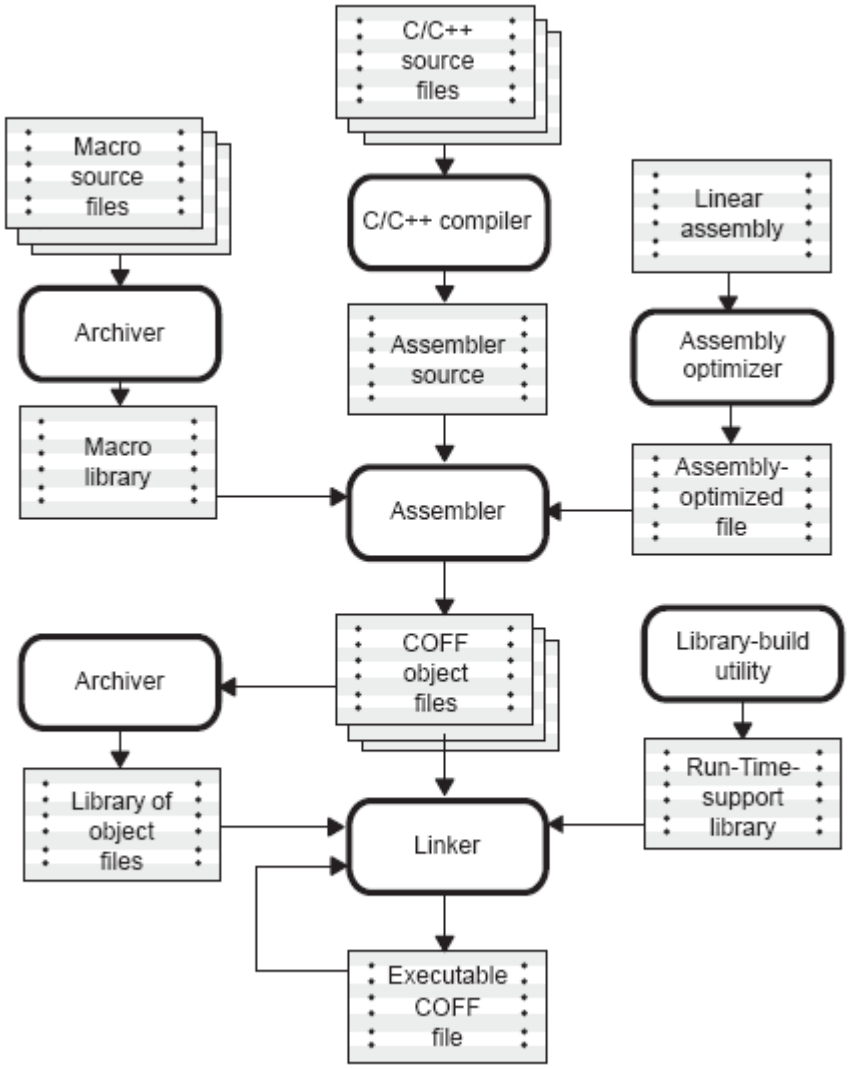


Diagrama 9. Diagrama de blocos do KIT de desenvolvimento DSK TMS320C6713
Retirada da documentação do fabricante do KIT:.
Fonte: Spectrum Digital (2003)

O processo de geração de código para o processador é mostrado no fluxograma 1.



Fluxograma 1. Fluxograma do processo de geração do arquivo executável para o KIT de desenvolvimento DSK TMS320C6713.
Fonte: Texas Instruments (2008)

A partir deste fluxograma, é possível entender o processo de criação de software comumente utilizado no desenvolvimento de software para aplicações embarcadas. O primeiro passo consiste na geração de um código fonte inicial em linguagem C. A seguir, um compilador desta linguagem é usado para gerar um arquivo em linguagem Assembly. Como um projeto pode incluir vários arquivos separados e bibliotecas diferentes, o terceiro passo é a montagem (também chamado de linkagem) de um código único, gerado a partir dos diversos arquivos envovlidos no projeto. Em geral, existem bibliotecas de rotinas que implementam tanto funções básicas do processamento quanto o gerenciamento dos diversos periféricos (portas seriais, conversores A/D e D/A, temporizadores, etc). O passo seguinte

consiste na geração de um arquivo executável que será carregado na memória do KIT para ser executado pelo processador.

Como toda versão inicial de software contém falhas, é necessário proceder com o teste e depuração deste programa gerado a partir do processo descrito no parágrafo anterior. Para isso, utiliza-se um emulador em tempo real que permite acompanhar a execução do programa, observando-se os valores de variáveis e registros internos do processador.

Incluído no kit de desenvolvimento, a Texas oferece um ambiente integrado de desenvolvimento (IDE – Integrated Development Environment) que é capaz de auxiliar tanto na escrita do software quanto na depuração da aplicação em tempo real, realizando todas estas tarefas a partir de acionamentos de botões numa tela de ambiente Windows. Este ambiente é chamado de Code Composer Studio (CCS).

Uma das características que o CCS apresenta é a facilidade de integração com o popular software de simulação matemática MATLAB. A partir de um modelo criado no MATLAB, é possível gerar um código em linguagem C que implemente a aplicação no processador digital de sinais, prosseguindo-se da simulação até o código final de forma conveniente em um único ambiente de desenvolvimento.

As imagens 1 a 13 mostram o processo de compilação e carga do programa executável na placa do KIT de desenvolvimento DSK TMS320C6713, como descrito no fluxograma 1.

Ao executar o *Code Composer Studio,* é apresentada a tela indicada na imagem 1.

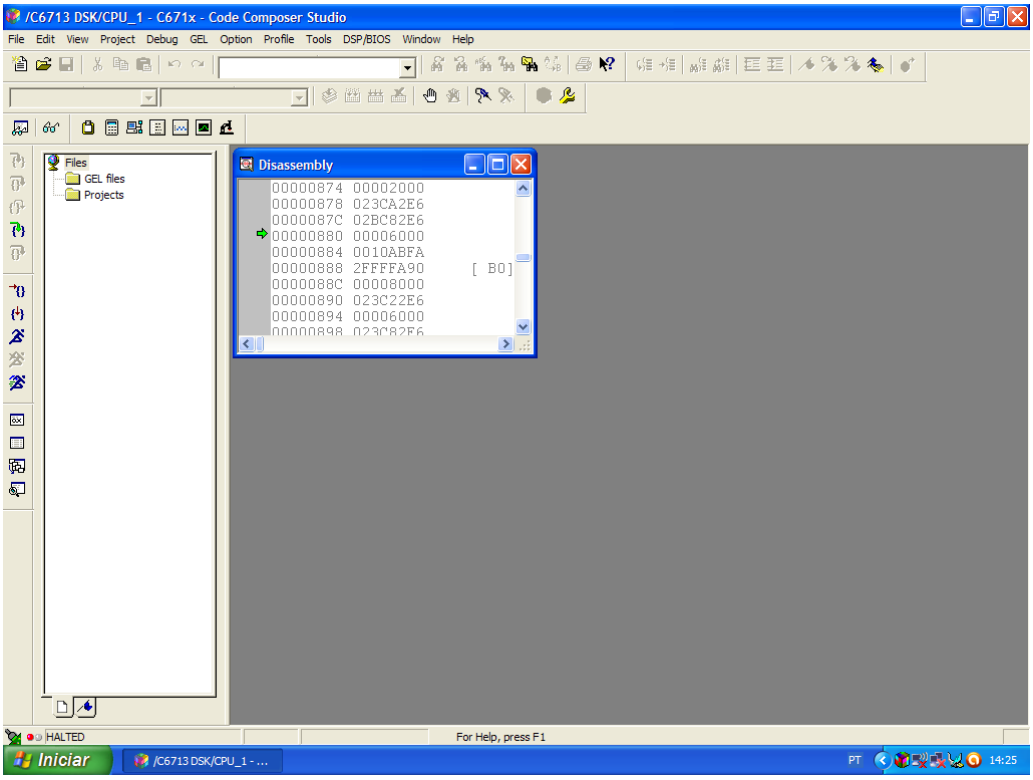


Imagem 1. Tela inicial do software Code Composer Studio

Clicando na aba Project, é selecionada a opção Open para a abertura do projeto. (Imagem 2).

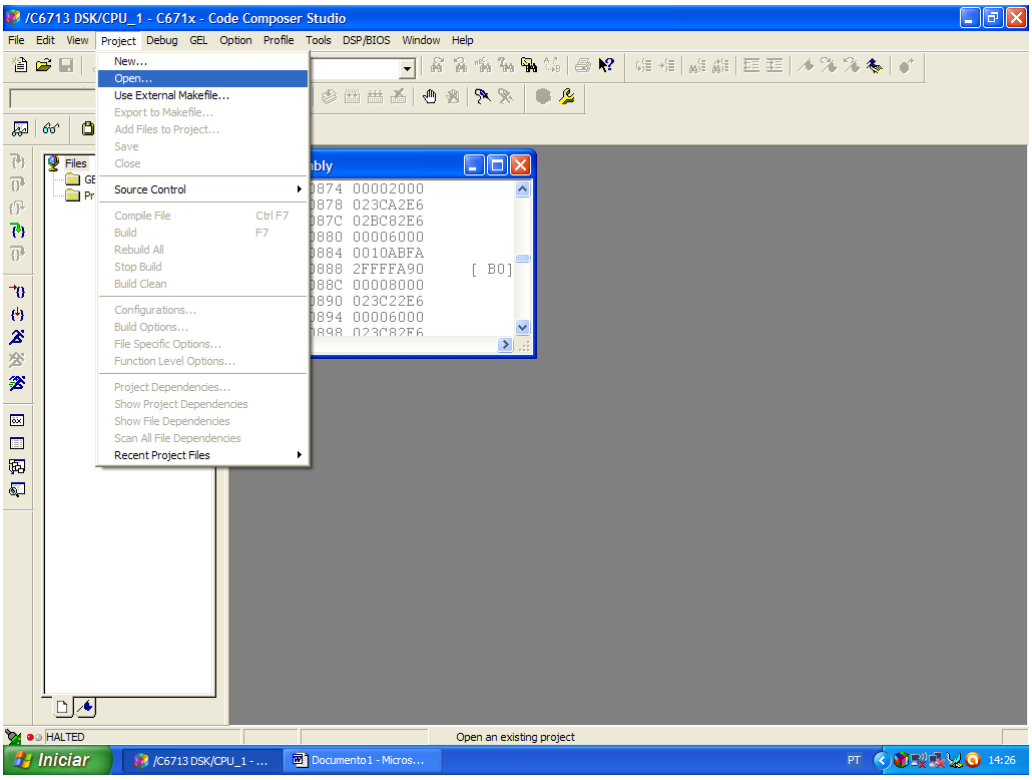


Imagem 2. Tela do software Code Composer Studio mostrando o processo de abertura de um arquivo de projeto.

O próximo passo foi carregar o projeto chamado de AM.pjt no Code Composer Studio, como indicado na imagem 3.

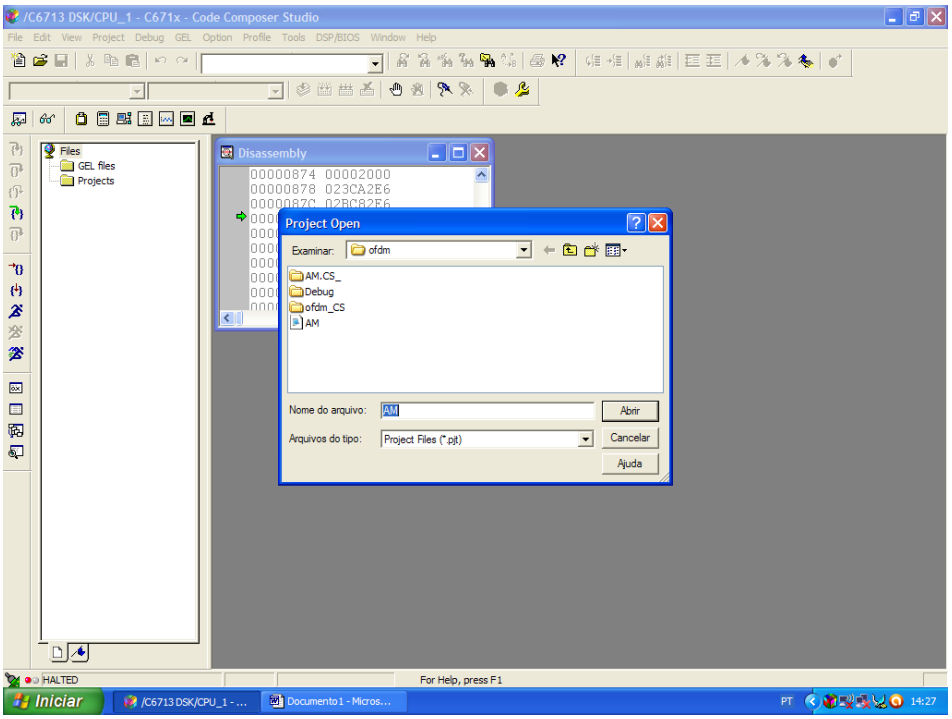


Imagem 3. Tela do software Code Composer Studio mostrando o processo de abertura do arquivo de projeto com nome AM.

A seguir é mostrada a tela (Imagem 4) que apresenta o projeto já carregado no Code Composer Studio.

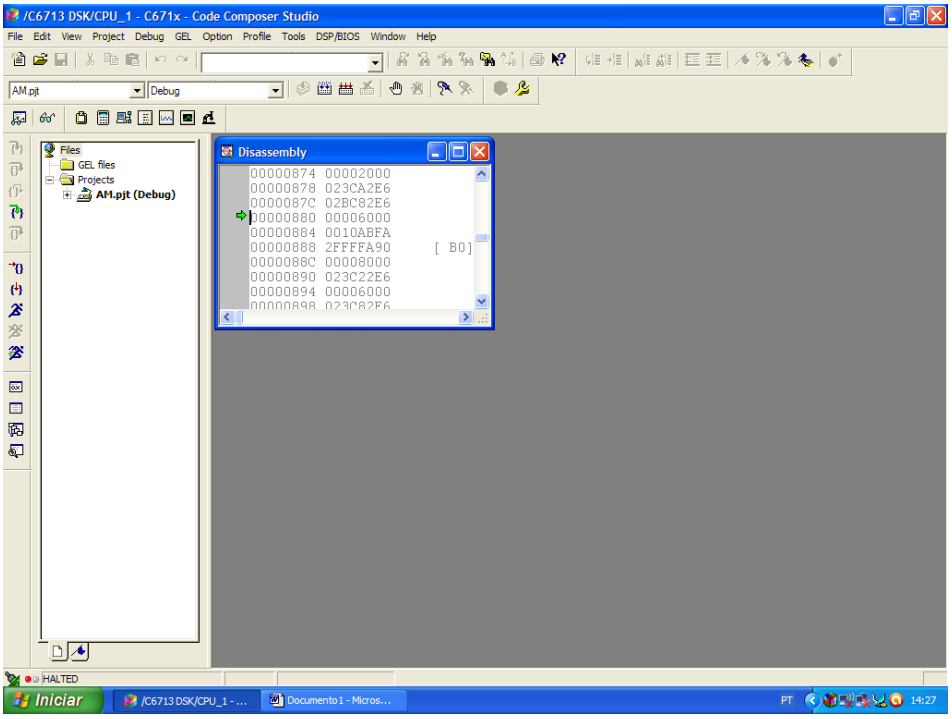


Imagem 4. Tela do software Code Composer Studio mostrando o projeto com nome AM já carregado corretamente.

Após a carga do projeto no ambiente de desenvolvimento de software, na área de trabalho pode-se visualizar todas as bibliotecas com os arquivos de configuração dos dispositivos do processador. Esses arquivos fazem parte da biblioteca include, comop mostrado na imagem 5. Nestes arquivos estão as informações de programação dos periféricos como timers, sistemas de interrupções conversor analógico/digital e conversor digital/analógico, entre outros. (

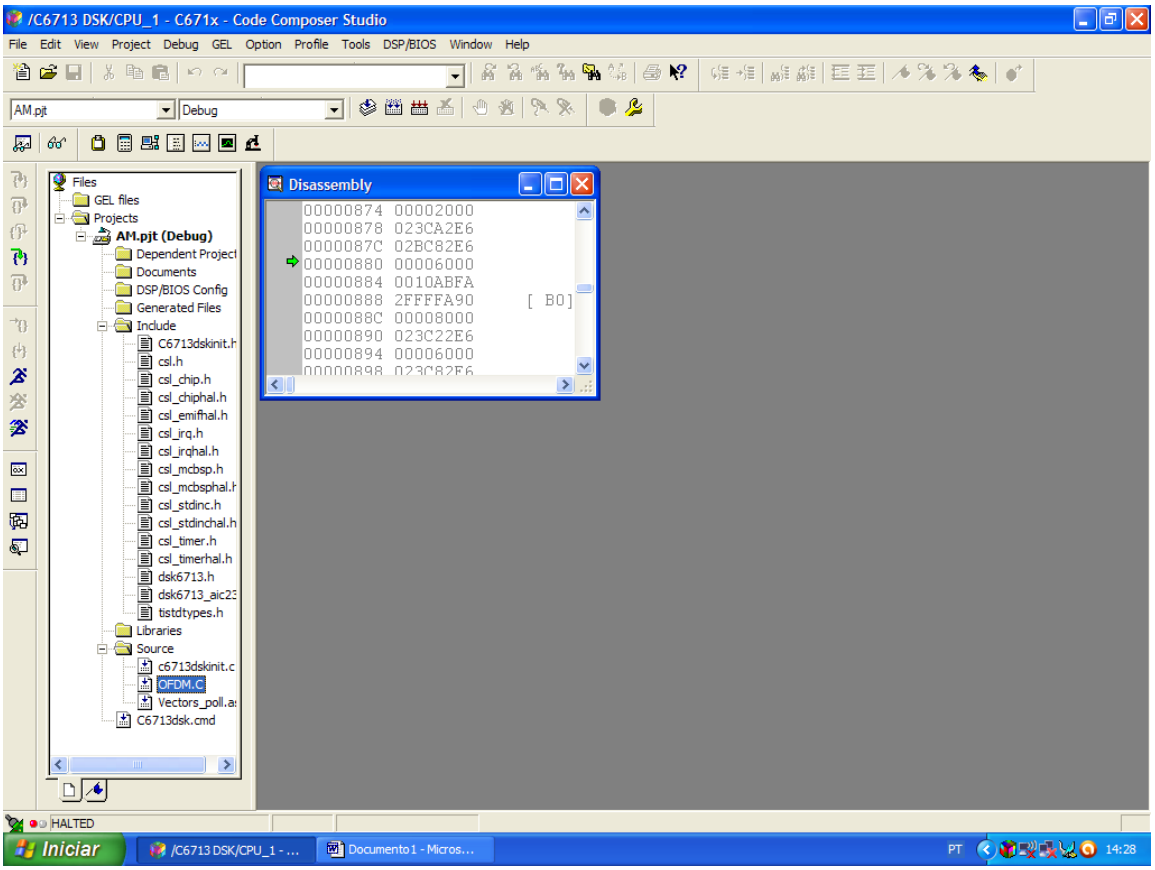


Imagem 5. Tela do software Code Composer Studio mostrando o projeto com todos os arquivos Include

O software desenvolvido em linguagem C foi adaptado de Chassaing (2005) e é mostrado abaixo na imagem 6, em que essencialmente é definida uma seqüência de dados a ser transmitidas pelo modulador OFDM (variável denominada baseband), bem como as portadoras OFDM (denominadas Carrier 1 e Carrier 2, de 1 e 2 kHz, repectivamente).

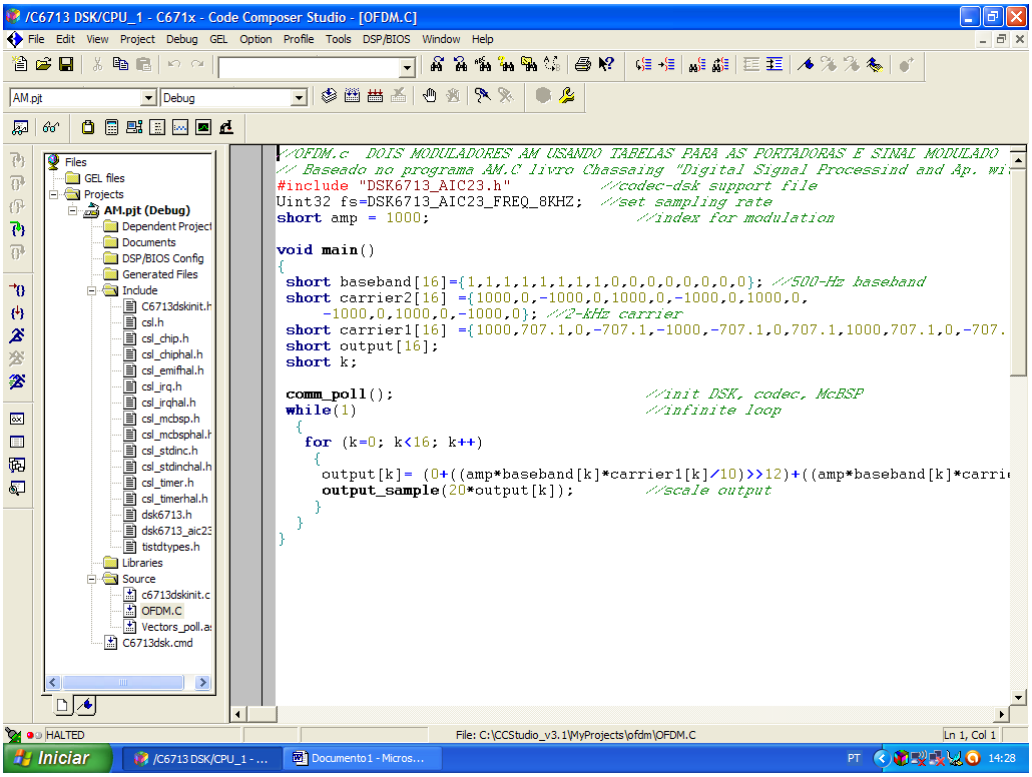


Imagem 6. Tela do software Code Composer Studio mostrando o arquivo OFDM carregado corretamente.

A seguir é feita a compilação de todo o projeto, usando o comando **rebuild all,** na aba **Project,** indicado na imagem 7.

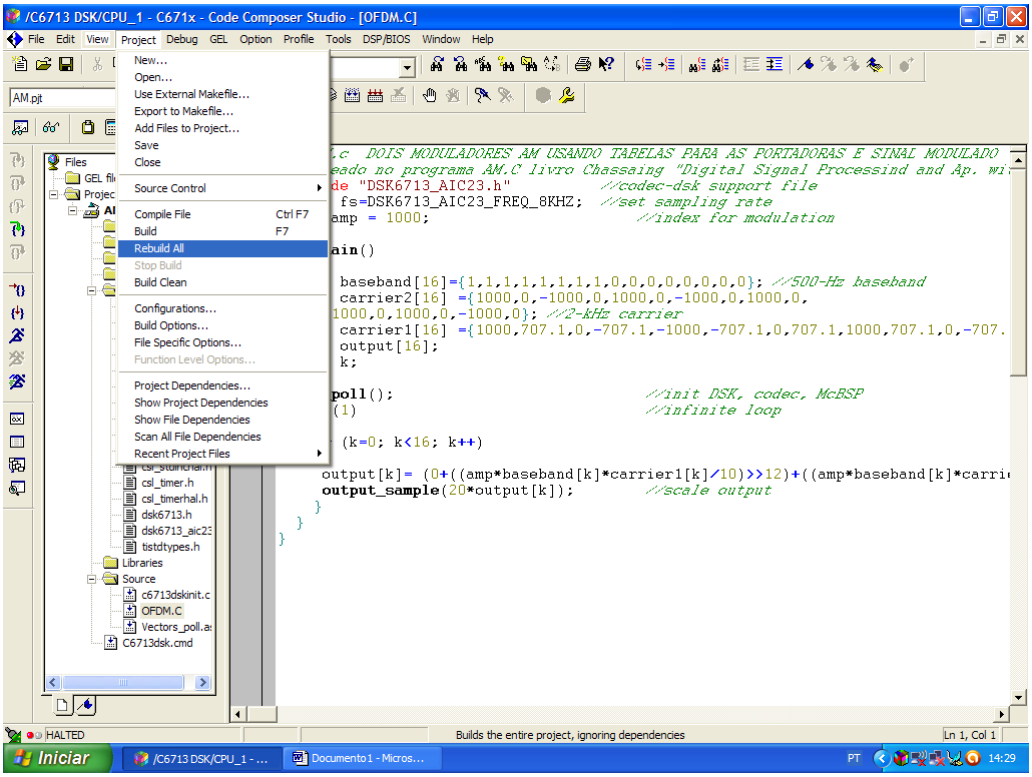


Imagem 7. Tela do software Code Composer Studio mostrando o processo de compilação do projeto.

A imagem 8 mostra o resultado do processo de compilação, que indica que não ocorreram *errors* nem *warnings*. Nesta etapa foi realizado um estudo detalhado das configurações da ferramenta referentes as áreas de memória onde as variáveis são armazenadas. Somente então após esta configuração foi realizada a compilação com sucesso. Na imagem destacam-se as linhas apresentadas na cor azul que indicam a configuração utilizada (**-g –q –fr**). Para maiores detalhes, sugere-se consultar Texas Instruments (2008). O arquivo gerado é o AM.out.

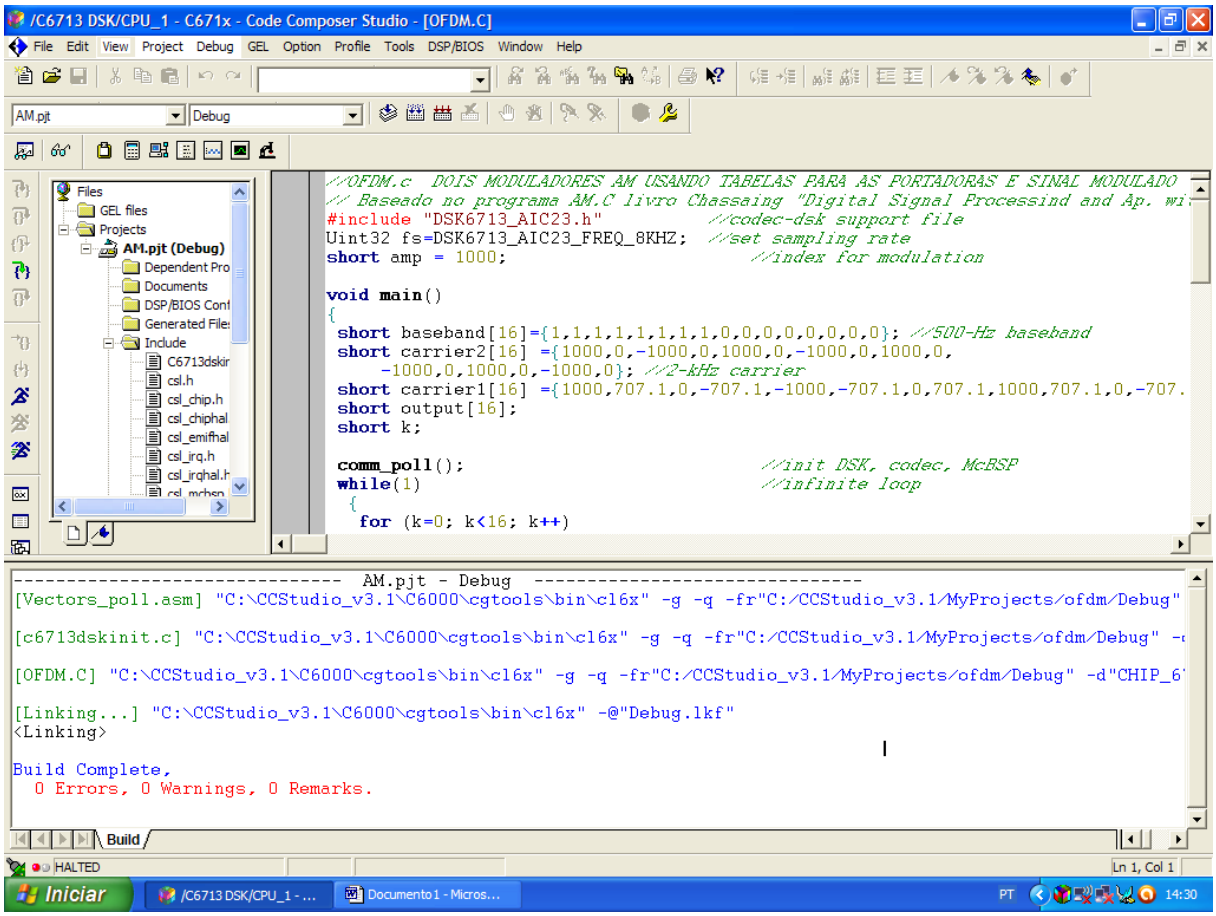


Imagem 8. Tela do software Code Composer Studio mostrando o final da compilação corretamente executada.

Finalizado o processo de compilação, o próximo passo é realizar o carregamento do código executável gerado pelo compilador na placa do KIT de desenvolvimento DSK TMS320C6713. O processo de carregamento é feito acessando-se a aba **File/Load Program.** Isto é mostrado na imagem 9.

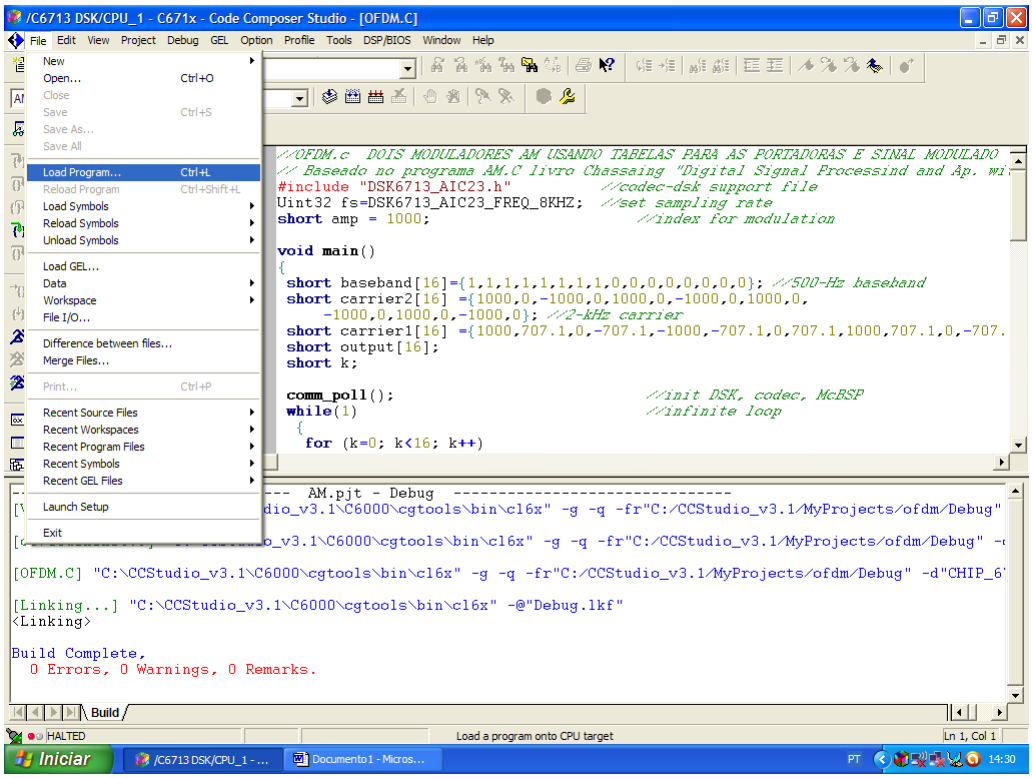


Imagem 9. Tela do software Code Composer Studio mostrando o processo de carga do arquivo executável no KIT de desenvolvimento DSK TMS320C6713.

Inicialmente é pedido o nome do arquivo a ser carregado: AM.out, como mostrado na imagens 10 e 11.

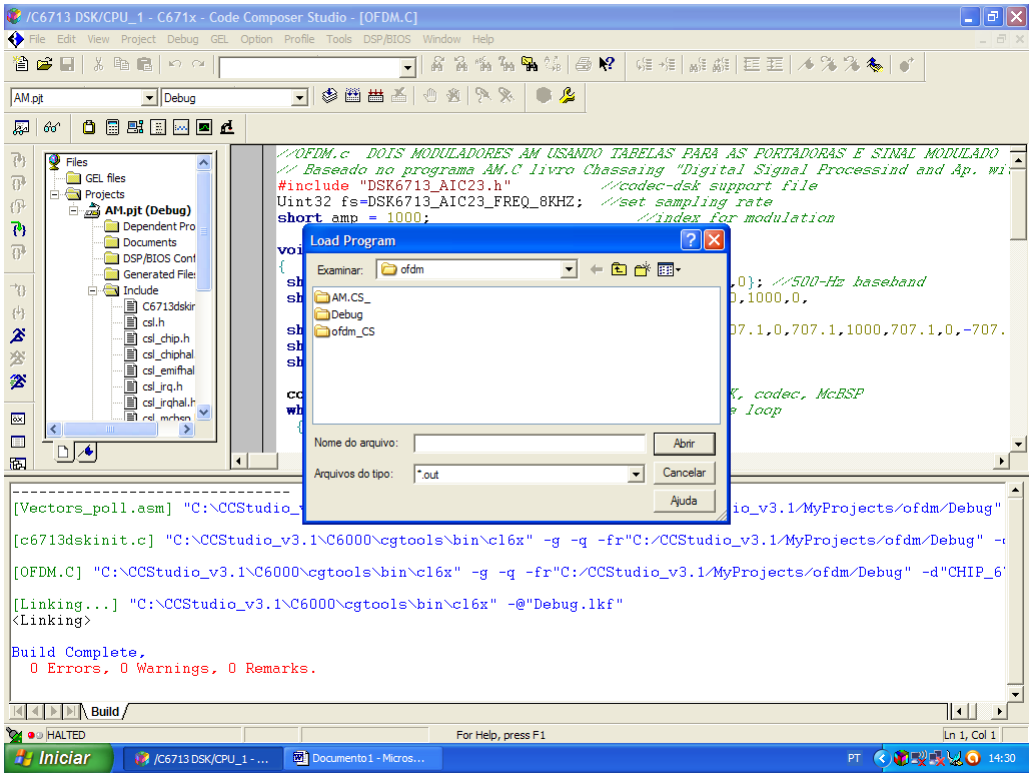


Imagem 10. Tela do software Code Composer Studio mostrando o processo de carga do arquivo executável que está na pastaDebug.

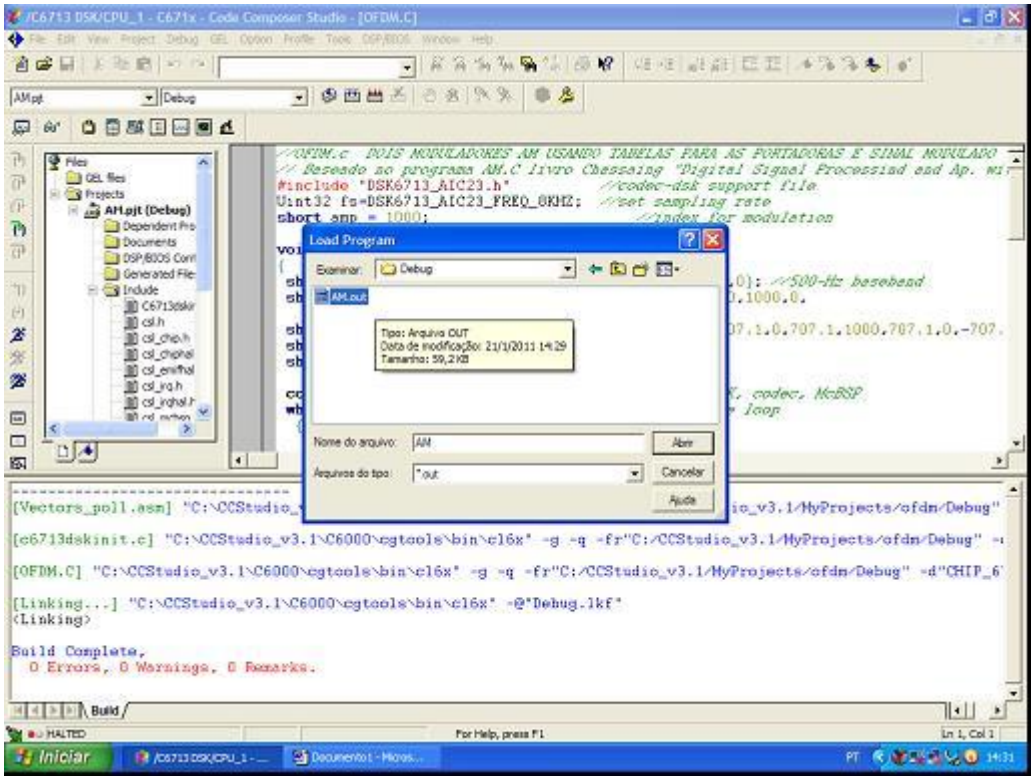


Imagem 11. Tela do software Code Composer Studio mostrando a seleção do arquivo executável AM.out que contem a programação para geração do sinal OFDM no KIT de desenvolvimento DSK TMS320C6713.

A seguir, as imagens 12 e 13 mostram o programa em execução na placa do DSP. Deve-se observar a conexão em verde (no canto inferior esquerdo da imagem 12) e quando o processamento é suspenso, esta conexão aparece em vermelho, (no canto inferior esquerdo da imagem 13)

Vale ressaltar que durante a execução do programa AM.out, o sinal OFDM é gerado na saída do conversor digita/analógico. Este sinal é analisado com indicado na seção 3.2.

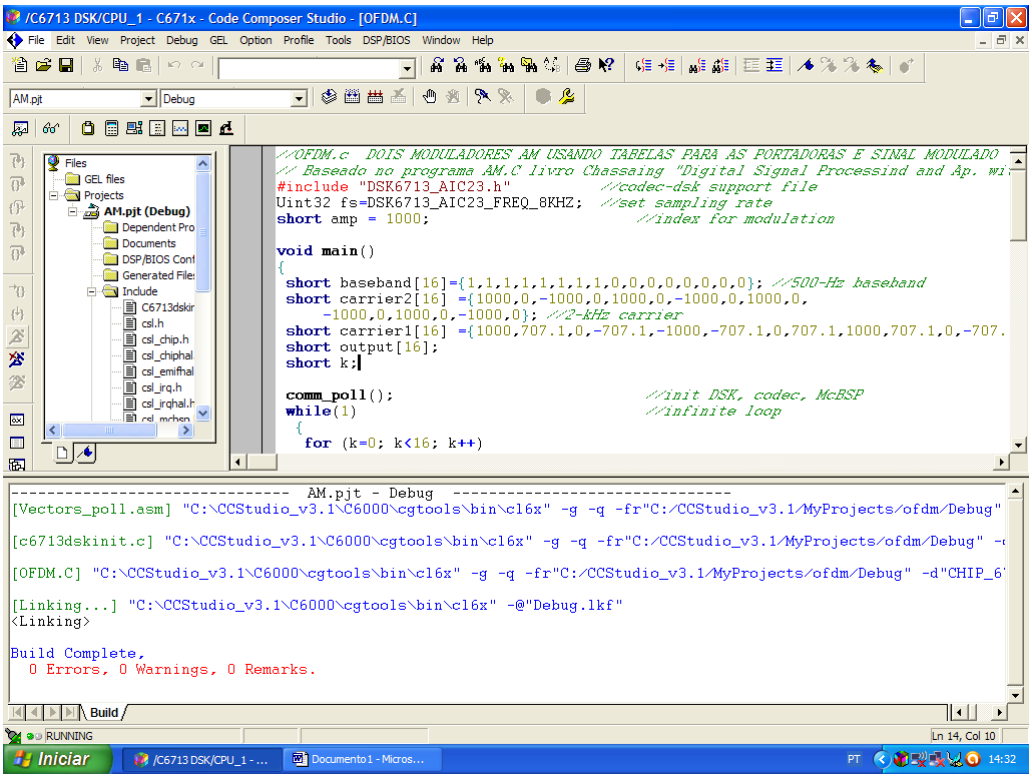


Imagem 12. Tela do software Code Composer Studio mostrando que o arquivo executável no KIT de desenvolvimento DSK TMS320C6713 está em execução.

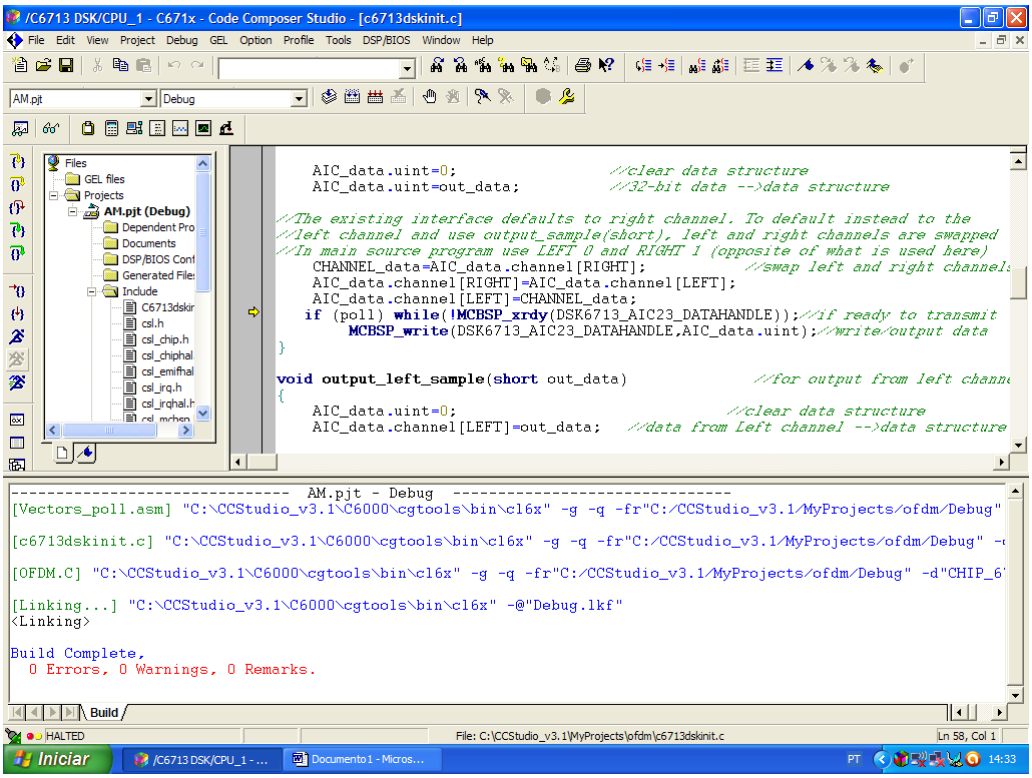


Imagem 13. Tela do software Code Composer Studio mostrando que o arquivo executável no KIT de desenvolvimento DSK TMS320C6713 está Halt.

3.2 A IMPLEMENTAÇÃO DO SISTEMA OFDM:

Nesta implementação, foi utilizado o modulador digital ASK (*amplitude shift keying*). O modulador digital ASK gera símbolos formados pela presença ou ausência de portadora, de acordo com a amplitude do sinal modulador. De modo geral, quando for bit 1, a portadora é transmitida e bit zero, a portadora não é transmitida. O modulador OFDM implementado neste trabalho é baseado no modulador OFDM ASK o qual foi desenvolvido, simulado em Matlab e implementado no KIT de desenvolvimento DSK TMS320C6713 para constatar o funcionamento do mesmo.

Foi utilizado um programa que gera sinal AM (modulação em amplitude) contido no livro do Chassaing (2005) "Digital Signal Processing and Ap. with C6713 and C6416 DSK", o qual foi modificado em um outro programa que gera símbolos OFDM ASK. Foram utilizados dois moduladores ASK's com duas portadoras de 1000 Hz e 2000 Hz onde foram moduladas com dados (sinal modulante) que são representados por um sinal digital periódico armazenado na previamente na memória do KIT, que gera espectro discreto, como mostrado na simulação em Matlab, e comprovada pela implementação no KIT de desenvolvimento DSK TMS320C6713, onde foi observada na tela do osciloscópio no modo FFT em tempo real.

A justificativa para a utilização das portadoras em 1000 Hz e 2000 Hz é devido aos conversores (CODEC AIC 23) contidos no KIT de desenvolvimento utilizado nesta pesquisa funcionarem em baixa freqüência.

A seguir é mostrado o programa OFDM.C que gera duas portadoras com modulação ASK, Baseado no programa AM.C livro Chassaing (2005) "Digital Signal Processind and Ap. with C6713 and C6416 DSK".

```
//OFDM.c  DOIS MODULADORES AM USANDO TABELAS PARA AS PORTADORAS E SINAL MODULADO
// Baseado no programa AM.C livro Chassaing (2005) "Digital Signal Processing and Ap. with C6713 and C6416 DSK
#include "DSK6713_AIC23.h"              //codec-dsk support file
Uint32 fs=DSK6713_AIC23_FREQ_8KHZ; //set sampling rate
short amp = 1000;                       //index for modulation

void main()
{
 short baseband[16]={1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0}; //500-Hz baseband
 short carrier2[16] ={1000,0,-1000,0,1000,0,-1000,0,1000,0,
       -1000,0,1000,0,-1000,0}; //2-kHz carrier
 short   carrier1[16]   ={1000,707.1,0,-707.1,-1000,-707.1,0,707.1,1000,707.1,0,-707.1,-1000,-707.1,0,707.1}; //1-kHz carrier
 short output[16];
 short k;

 comm_poll();                    //init DSK, codec, McBSP
 while(1)                        //infinite loop
  {
   for (k=0; k<16; k++)
    {
     output[k]=
(0+((amp*baseband[k]*carrier1[k]/10)>>12)+((amp*baseband[k]*carrier2[k]/10)>>12));
     output_sample(20*output[k]);       //scale output
    }
  }
}
```

A seguir é mostrado o programa em MATLAB para gerar as amostras das portadoras a serem inseridas no programa OFDM.c

```
%Programa gerador de amostras das portadoras
n=0:1/8000:15*1/8000;
a1=cos(2*pi*1000*n)
a2=cos(2*pi*2000*n)

a1 =

  Columns 1 through 6

   1.000000000000000   0.707106781186548   0.000000000000000  -0.707106781186547  -1.000000000000000  -0.707106781186548

  Columns 7 through 12

  -0.000000000000000   0.707106781186547   1.000000000000000   0.707106781186548   0.000000000000002  -0.707106781186547

  Columns 13 through 16

  -1.000000000000000  -0.707106781186548  -0.000000000000002   0.707106781186547


a2 =

  Columns 1 through 6

   1.000000000000000   0.000000000000000  -1.000000000000000  -0.000000000000000   1.000000000000000   0.000000000000000

  Columns 7 through 12
```

-1.000000000000000 -0.000000000000000 1.000000000000000 0.000000000000002 -1.000000000000000 -0.000000000000002

Columns 13 through 16

1.000000000000000 0.000000000000003 -1.000000000000000 -0.000000000000003

A seguir é mostrado o programa em MATLAB que simula o gerador implementado no programa OFDM.c

```
%Programa que gera sinal OFDM
%neste programa são usadas duas portadoras com freq. f1=1000Hz f2=2000Hz
%os dados de entrada apresentam taxa de bits de 2000 bps.

dados=1000*[1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0]*10;%n de dados = 16  =   8000/((2000/2)/2)
%8000 é a fs
%2000/2 é devido nosso modulador possui duas portadoras
%/2 é devido que um ciclo transporta duas amostras
portadora2=[1000 0 -1000 0 1000 0 -1000 0 1000 0 -1000 0 1000 0 -1000 0];
portadora1=1000.*[1.0000  0.7071  0.0000  -0.7071  -1.0000  -0.7071  -0.0000  0.7071
1.0000  0.7071  0.0000  -0.7071 -1.0000  -0.7071  -0.0000  0.7071];
modulado1=portadora1.*dados/10;
modulado2=portadora2.*dados/10;
modulado=modulado1+modulado2;
plot([0:1/8000:15*1/8000],modulado);
title('sinal modulado');
xlabel('tempo s');
figure;
f=(abs(fft([20.*modulado])));
stem([0:.5:7.5],f);
title('espectro do sinal modulado');
xlabel('frequencia KHz');
```

Os gráficos 2 e 3 a seguir mostram os sinais simulados no MATLAB, os quais devem ser os mesmos obtidos na saída do KIT de desenvolvimento DSK TMS320C6713.

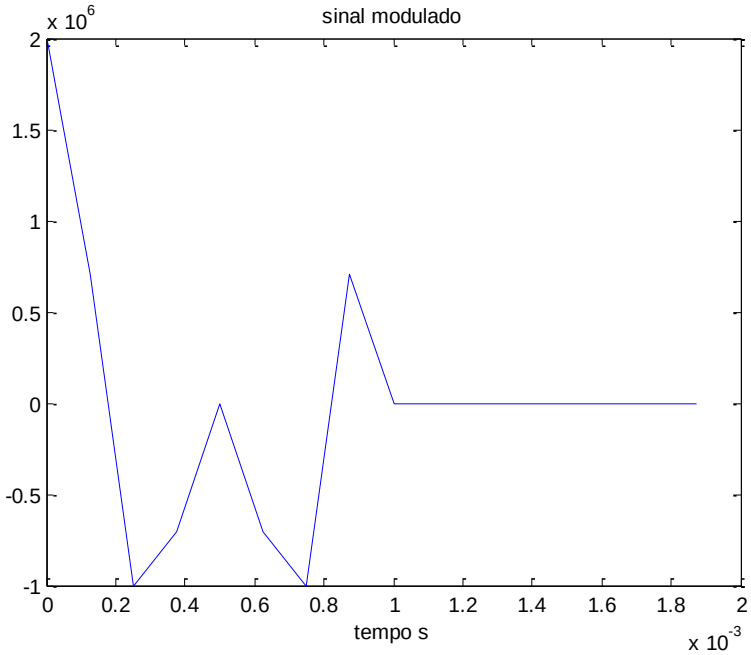


Gráfico 2. Trecho do sinal OFDM no domínio do tempo simulado no MATLAB.

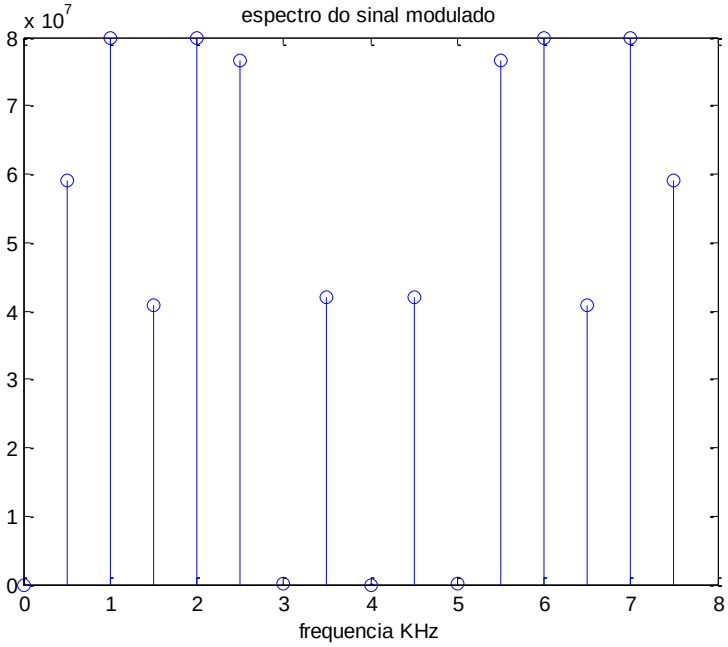


Gráfico 3. Sinal OFDM no domínio da freqüência simulado no MATLAB. (taxa de Nyquist de 4 kHz).

A partir de 4 kHz há aliasing.

Os gráficos 4 e 5 mostram os sinais obtidos na saída do KIT de desenvolvimento DSK TMS320C6713.

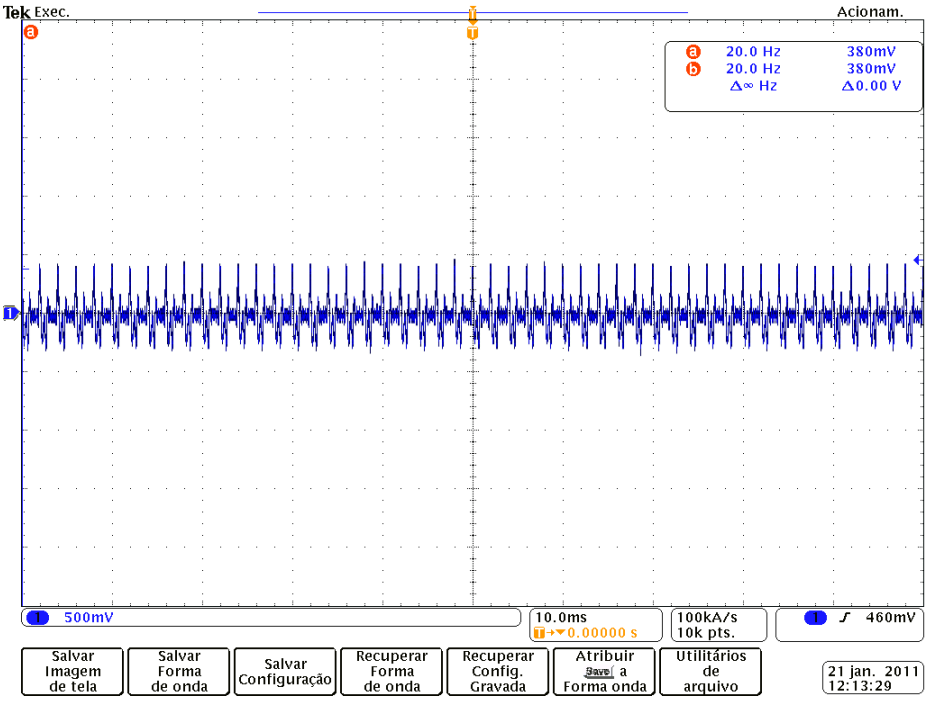


Gráfico 4. Sinal real OFDM no domínio do tempo capturado no osciloscópio digital no Laboratório de Processamento de Sinais e Sistemas de Controle da EE Mackenzie.

Pode-se observar que o sinal real mostrado no gráfico 4 corresponde à repetição periódica mostrada no sinal simulado em MATLAB mostrado pelo gráfico 2.

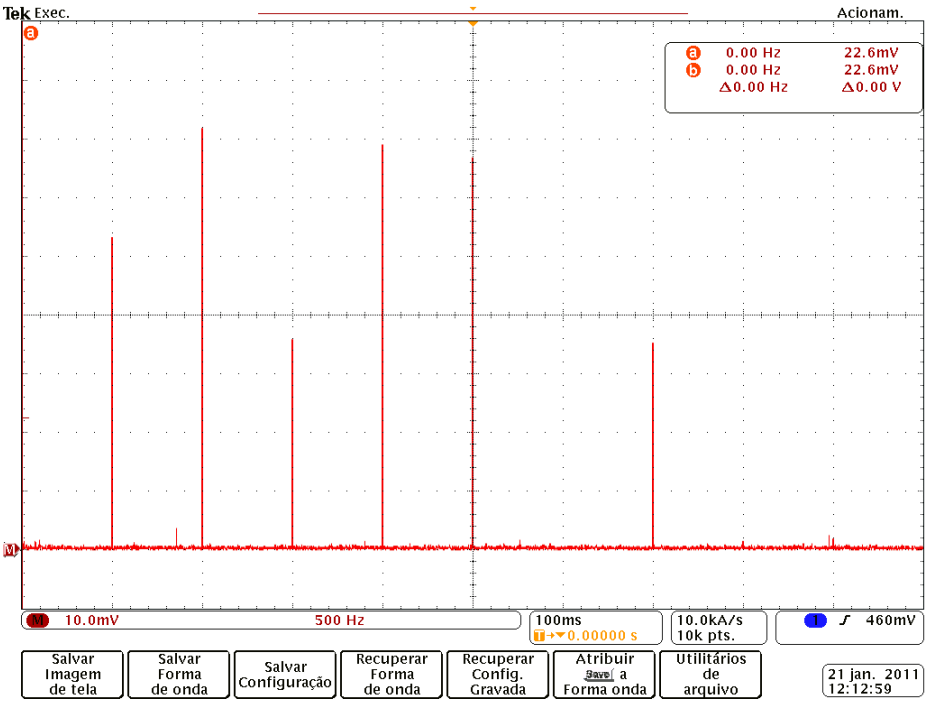
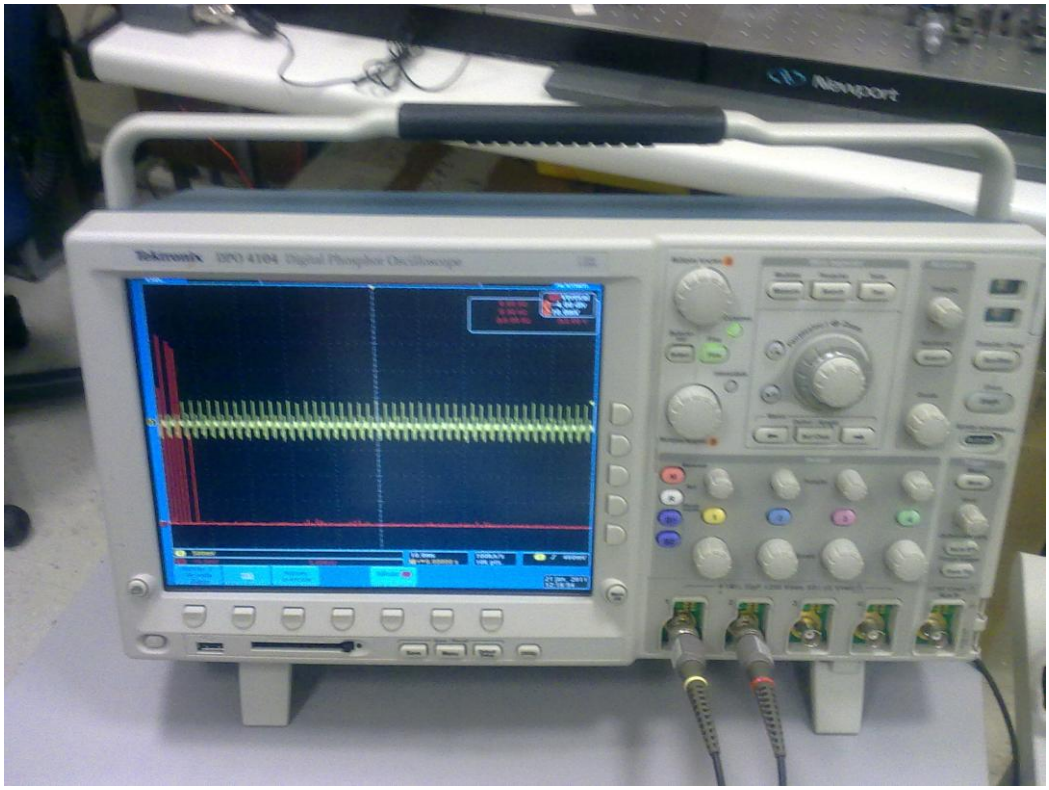


Gráfico 5. Sinal real OFDM no domínio da freqüência simulado no MATLAB.(taxa de Nyquist de 4 kHz) capturado do osciloscópio no modo FFT, no Laboratório de Processamento de Sinais e Sistemas de Controle da EE Mackenzie.

Pode-se observar que o sinal real mostrado no gráfico 5 corresponde à repetição periódica mostrada no sinal simulado em MATLAB mostrado pelo gráfico 3

As simulações práticas e teóricas mostram a mesma informação.

Abaixo segue a fotografia 4, da tela do osciloscópio, com o sinal OFDM no domínio do tempo mostrado na cor amarela e o seu respectivo espectro (em vermelho). Ressalta-se que o sinal está sendo processado em tempo real pela placa de DSP.



Fotografia 4. Sinal OFDM e seu respectivo espectro, em tempo real usando o KIT de desenvolvimento DSK TMS320C6713, no Laboratório de Processamento de Sinais e Sistemas de Controle da EE Mackenzie.

4 CONCLUSÃO

Os sistemas de comunicação via rádio tornaram-se imensamente difundidos na atualidade por permitirem o acesso da população a serviços de voz, dados e multimídia. Exemplos ao nosso redor são facilmente percebidos: telefonia celular, redes wifi, etc. Neste contexto, é importante observar que a técnica de transmissão por portadoras ortogonais (OFDM) converteu-se na opção tecnológica preferencial para a implementação destes sistemas. Além dos sistemas via rádio, o OFDM é utilizado em transmissão por fibras ópticas, comunicação pelas redes de energia (PLC), etc.

Dependendo da aplicação específica, o OFDM pode ser modificado para aumentar a qualidade de serviço oferecido aos usuários. Portanto, há várias nuances na implementação do transmissor e receptor OFDM. Tais alterações devem ser feitas com cautela e testadas de acordo com o ambiente no qual elas vão ser utilizadas. Daí infere-se a importância do estudo e da realização de testes em tais sistemas.

Este projeto de pesquisa permitiu o estabelecimento de um banco de teste completamente programável para possibilitar o estudo, a análise e modificações em Sistemas de Comunicações que utilizam a técnica OFDM. A modularidade das funções implementadas em software permite que novas alterações possam ser rapidamente testadas, fomentando novas idéias para pesquisa.

Este banco de testes foi implementado em um processador digital de sinais de uso geral, mas com um sistema de desenvolvimento suficientemente flexível para permitir sua interligação com o ambiente de simulação MATLAB. Desta forma, um algoritmo pode ser desenvolvido e simulado em MATLAB e portado para um processador físico, unindo o conhecimento teórico ao prático

Um objetivo secundário também alcançado foi a incorporação de técnicas de projeto de sistemas com Processadores Digitais de Sinais ao universo de conhecimento disponível na Escola de Engenharia. Tais técnicas são utilizadas, também, no desenvolvimento de equipamentos de comunicação, de instrumentação biomédica, de entretenimento, etc. Tanto os alunos quanto os professores da equipe puderam se envolver nos diversos aspectos do projeto de sistemas processados numericamente e se tornaram aptos a prestar serviços de consultoria nestes segmentos da indústria.

Entretanto, algumas dificuldades apareceram durante a execução do projeto. A primeira consistiu no prazo extremamente longo para chegada do material solicitado. O projeto teve de ser retardado por causa desta demora. Em segundo lugar, a demora no

recebimento não permitiu que os alunos bolsistas permanecessem até o final do projeto, pois suas bolsas foram encerradas no prazo estabelecido. E, finalmente, as placas de desenvolvimento não foram compradas ainda. O projeto só pode ser finalizado graças a uma doação da empresa Texas Instruments.

Acredita-se que o resultado do projeto servirá de base para novas pesquisas e treinamentos baseados nas técnicas de processamento Digital da Informação, contribuindo para um aumento no número de trabalhos de graduação, dissertações e teses nesta área de conhecimento. O desenvolvimento em MATLAB e em linguagem C dos códigos para implementação do Modulador OFDM formam uma poderosa combinação de ferramentas que poderão ser utilizadas em futuras pesquisas por alunos e pesquisadores. Todas as rotinas estão disponibilizadas nos anexos.

A DEMODULAÇÃO SERÁ ESTUDA PARA VERIFICAÇÃOP DA POSSIBILIDADE DE SUA IMPLEMENTAÇÃO EM TRABALHO POSTERIOR

REFERÊNCIAS

BRASIL, M. A. A. *Análise do OFDM em sistemas CDMA*. 2001. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Presbiteriana Mackenzie, São Paulo, 2001.

CHASSAING, R. *Digital Signal Processing and Application with C6713 and C6416 DSK*. New Jersey: John Wiley and Sons, 2005.

CIUPKA, A. et al. *PLC*: uma proposta de nova tecnologia em transmissão de dados. 2002. Trabalho de Conclusão de Curso (Graduação em Tecnologia Elétrica) - Universidade Presbiteriana Mackenzie, São Paulo, 2002.

DIAS, JOÃO TERÊNCIO. Desempenho de algoritmos para estimação de Parâmetros de sincronização em sistemas OFDM. Dissertação (Mestrado em Engenharia Elétrica. Rio de Janeiro, 2006. Disponível em: http://www.pgee.ime.eb.br/pdf/joao_dias.pdf. Acesso em: 20 fev. 2011.

HATAE, D. L.; EISENCRAFT, M.; AKAMINE, C.; DANTAS C. E. S.; SILVA, M. T. M.; Miranda, M. D. OFDM systems for Brazilian digital television channels. In: 10th International OFDM Workshop, 2005, Hamburgo. Proceedings of the 10th INTERNATIONAL OFDM WORKSHOP. Hamburg: Technische Universität Hamburg-Harburg, 2005. v. 1. p. 61-63.

LIU, H.; LI, G. *OFDM-Based Broadband Wireless Networks Design and Optimization*, Hoboken, New Jersey: John Wiley & Sons, 2005.

MENESES, A.S.; PANAZIO, C. M.; ROMANO, J. M. T., Equalização Cega de Canais Espaço-Temporais Variantes no Tempo usando Predição Linear Adaptativa em Sistemas OFDM. XXV Simpósio Brasileiro de telecomunicações - SbrT, 2007 Setembro de 2007, Recife, PE, Brasil.

PRASAD, R. *OFDM for Wireless Communications Systems*. Boston: Artech House, 2004.

PROAKIS, J. G. *Digital Communications*. 5th ed. Singapore: McGrawHill, 2008

SILVA, E. G. *Sobre a equalização adaptativa e a transmissão digital com múltiplas portadoras*. 2002. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Presbiteriana Mackenzie, São Paulo, 2002.

SPECTRUM DIGITAL. TMS320C6713 DSK: Technical Reference. Stafford: [s. n.], 2003. Disponível em: http://c6000.spectrumdigital.com/dsk6713/V2/docs/dsk6713_TechRef.pdf. Acesso em: 20 fev. 2011.

TEXAS INSTRUMENTS. TMS320C6000 Optimizing Compiler v 6.1: User's guide. [S. l.: s. n.], 2008. Disponível em: http://www.ti.com.cn/cn/lit/ug/spru187o/spru187o.pdf. Acesso em: 20 fev. 2011.

TEXAS INSTRUMENTS. TMS320C6713B. Floating-point digital signal processor. [S. l.: s. n.], 2005. Disponível em:http://www.ti.com/lit/ds/sprs294b/sprs294b.pdf. Acesso em: 20 fev. 2011.