Referências Bibliográficas

1 _, "Special Issue on Mobile Radio Propagation", **IEEE Trans. Vehic. Tech**., vol. 37, N. 1, 1988.

2 AL-NUAIMI, M.O., STEPHENS, R.B.L. "Measurements and Prediction Model Optimization for Signal Attenuation in Vegetation Media at Centimetre Wave Frequencies", **IEE Proc. Microw. Antennas Propagat.**, vol. 145, N. 3, pp. 201-206, 1998.

3 TORRICO, S.A., BERTONI, H.L., LANG, R.H., "Modeling Tree Effects on Path Loss in a Residential Environment", **IEEE Trans. Antennas Propagat**., vol. 46, N. 6, pp. 99-302, 1998.

4 TAMIR, T., "On Radio-Wave Propagation in Forest Environments", **IEEE Trans. Antennas Propagat**., vol. 15, N. 6, pp. 806-817, 1967.

5 TAMIR, T., "Radio Wave Propagation Along Mixed Paths in Forest Environments", **IEEE Trans. Antennas Propagat**., vol. 25, N. 7, pp. 471-477, 1998.

6 TEWARI, R.K., SWARUP, S., ROY, M.N., "Radio Wave Propagation Through Rain Forests of India", **IEEE Trans. Antennas Propagat**., vol. 38, N. 4, pp. 443-449,1990.

7 AL-NUAIMI, M.O., HAMMOUDEH, A.M., "Measurements and Predictions of Attenuation and Scatter of Microwave Signal by Trees", **IEE Proc. Microwave Antennas Propagat.**, vol. 141, N. 2, pp. 70-76, 1994.

8 KOH, J.H., LI, L.W., KOOI, P.S. et al., "Dominant Lateral Waves in the Canopy Layer of a Four-layered Forest", **Radio Science** , vol. 3, N. 3, pp. 681-691, 1999.

9 LA GRONE, A.H., CHAPMAN, "Propagation Characteristics of High UHF Signals in the Immediate Vicinity of Trees", **IRE Trans. Antennas Propagat**., 1961.

10 VOGEL, W.J., GOLDHIRSH, J., "Tree Attenuation at 869 MHz Derived from Remotely Piloted Aircraft Measurements", **IEEE Trans. Antennas Propagat**., vol. 34, N. 12, pp. 1460-1464, 1986.

11 DAL BELLO, J.C.R., **Caracterização da Influência da Vegetação nos Sistemas de Comunicações Móveis Celulares em Áreas Urbanas**, Tese de Doutorado, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Fevereiro de 1998.

12 SWARUP, S., TEWARI, R.K., "Depolarization of Radio Waves in Jungle Environment", **IEEE Trans. Antennas Propagat**., vol. 27, N. 14, pp. 113-116,1979.

13 REUDINK, D.O., WAZOWICS, M.F. "Some Propagation Experiments Relating Foliage Loss and Diffraction loss at X-Band and UHF Frequencies", **Trans. Communications**, vol. 21, N. 11, pp. 1198-1206, 1973.

14 SEKER, S.S., SCHNEIDER, A., "Experimental Characterisation of UHF Radiowave Propagation Through Forests", **IEE Proceedings**, vol. 140, N. 5, pp. 329-335, October 1993.

15 LOW, K., "Measurements of Seasonal Field-Strenght Variations in Forests", **IEEE Trans. Veh. Technol**., vol. 37, N. 3, pp. 121-124, 1988.

16 COX, DONALD C., "Delay-Doppler Characteristics of Multipath Propagation at 910 MHz in a Suburban Mobile Radio Environment", **IEEE Trans. Antennas Propagat**., vol. 20, N. 9, pp. 625-635, September 1972.

17 PARSONS, J.D., **The Mobile Radio Propagation Channel**, 2nd edition, New York, John Wiley & Sons, 2000.

18 Bello, P.A, Randomly Time-Variant Linear Channels, **IEEE Trans. On Comm. Systems**, vol. 11, pp. 360-393, December 1963

19 YACOUB, MICHEL DAOUD, **Foundations of Mobile Radio Engineering**, 1a edição, USA, CRC Press, 1993.

20 HOWARD, S. and PAHLAVAN, Doppler Spread Measurements of the Indoor Radio Channel, **IEE Electronic Letters**, N. 26, pp 107-109, 1990.

21 MACWILLIAMS, F.J. and SLOANE, N.J. , Pseudo Random Sequences and Arrays, **Proc. IEEE**, vol. 64, N. 12, pp 1715-1730, December 1976.

22 FANNIN, P.C., MOLINA, A., SWORDS, S.S., CULLEN, P.J..Digital Signal Processing Techniques Applied to Mobile Radio Channel Sounding, **IEE Proceedings-F**, vol. 138, N. 5, October 1991.

23 CULLEN, P.J., FANNIN, P.C., AND MOLINA, A, 'Wide-Band Measurement and Analysis Techniques for the Mobile Radio Channel", **IEEE Trans. Veh. Technol**., vol. 42, N. 4, pp. 589-603, 1993.

24 VÁSQUEZ, E.J.A., "Caracterização do Canal Móvel em Faixa Larga", **Tese de Doutorado**, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Abril de 2000.

25 NEWHALL,W.G, SALDANHA,K., RAPPAPORT,T.S, "Using RF Channel Sounding Measurements to Determine Delay Spread and Path Loss", **RF Propagation**, pp. 82-88, January 1996.

26 NEWHALL, W.G, RAPPAPORT, T.S., SWEENEY, D.G., "A Spread Spectrum Sliding Correlator System for Propagation Measurements", **RF Spread Spectrum,** pp. 40-54, April 1996.

27 DEVASIRVATHAM,D.M.J., "Time Delay Spread and Signal Level Measurements of 850 MHz Radio Waves in Building Environments", **IEEE Trans. Antennas Propagat**., vol. 34, N. 11, pp. 1300-1305, November 1986.

28 FELHAUER,T., BAIER,P.W., KÖNIG,W., AND MOHR,W., "Optimum Spread Spectrum Signals for Wideband Channel Sounding", **Electronic Letters**, vol. 29, No. 6, pp. 563-564, 18th March 1993.

29 FELHAUER,T., BAIER,P.W., KÖNIG,W., AND MOHR,W., "Optimized Wideband System for Unbiased Mobile Radio Channel with Periodic Spread Spectrum Signals", **IEICE Trans. Commun**., vol. E76-B, No. 8, pp.1016-1029, August 1993.

30 MOHR,W., STEINER,B., WEBER,P., "The Siemens Mobile Channel Sounder - An Optimised Wideband System for the Determination of Mobile Radio Channel Impulse Responses", **COST 231 TD(95)/Poznan**, pp.1-20, September 13-15, 1995

31 MOLINA, A, FANNIN, P.C., AND TIMONEY, J., "Generation of Optimum Excitation Waveforms for Mobile Radio Channel Sounding", **IEEE Trans. on Vehicular Technology**, vol. 44, No. 2, pp 275-279, May 1995.

32 -, **The Student Edition of Matlab**, Version 4, The Math Works Inc., New Jersey, Prentice Hall, 1995

33 JERUCHIM et al., **Simulation of Communication Systems**, Plenum Press, New York, 1992.

34 SHENOI, KISHAN, **Digital Signal Processing inTelecommunications**, Prentice Hall PTR, New Jersey, 1995.

35 BENVENUTO,N., "Distortion Analysis on Measuring the Impulse Response of a System Using a Crosscorrelation Method", **AT&T Bell Laboratories Technical Journal**, vol. 63, No. 10, December 1984.

36 TOBEY, GRAEME & HUELSMAN, **Operational Amplifiers,** McGraw-Hill, Kogakusha, 1971.

37 RAMOS, GLAUCIO L., **Medidas de Radio Propagação em 5,5 GHz em Ambientes Urbanos: Perda de Percurso e Variabilidade,** Dissertação de Mestrado, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Março de 2001.

38 _, DAQCard E Series User Manual, **National Instruments Corporation**, March 1999 Edition.

39 MACÊDO, L.H., **Sondagem em Freqüência do Canal *Indoor* de Faixa Larga**, Dissertação de Mestrado, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Fevereiro 2002.

40 DIAS, M.H.C., **Estimação das Respostas do Canal Real de Propagação Rádio-Móvel nos Domínios Espacial e Temporal - Análise da Supressão de Ruído por Decomposição *Wavelet* como Técnica Complementar de Processamento**, Tese de Doutorado, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Abril de 2003.

41 SOUSA, ELVINO S., JOVANOVIÉ, V.M., DAIGNEAULT, C., "Delay Spread Measurements for the Digital Cellular Channel in Toronto", **IEEE Trans. on Vehicular Technology**, vol. 43, No. 4, pp. 837-847, November 1994.

42 HARRYS, F.J., "On the Use of Windows for Analysis with the Discret Fourier Transform", **Proc. IEEE**, Vol. 66, pp. 51-73, January 1978.

43 STRUM, R.D., KIRK, D.E., **First Principles of Discrete Systems and Digital Signal Processing**, Addison-Wesley Publishing Company, Massachusetts, April 1989.

44 PARSONS, J.D., DEMERY,D.A. and TURKMANI, A.M.D., "Sounding Techniques for Wideband Mobile Radio Channels: a Review", **IEE Proceedings**, vol. 138, No. 5, pp. 437-446, October 1991.

45 FLEURY, B.H., "An Incertainty Relation for WSS Processes and Its Application to WSSUS Systems", **IEEE Trans. On Communication Systems**, vol. 44, No. 12, pp. 1632-1634, December 1996.

46 Seker, S.S., Schneider, A., "Experimental Characterisation of UHF Radiowave Propagation Through Forests", **IEE Proceedings-H**, vol. 140, No. 5, pp. 329-335, October 1993.

47 Xiongwen, Z., Kivinen, J., Vainikainen, P., "Characterization of Doppler Spectra for Mobile communications at 5.3 GHz", **IEEE Trans. on Vehicular Technology**, vol. 52, No. 1, pp. 837-847, pp. 14-23, January 2003.

48 Lee, W.C.Y., **Mobile Cellular Telecommunications Systems**, McGraw Hill, 1989.

49 Lathi, B.P., **Modern Digital and Analog Communication Systems**, Oxford University Press, 1998.

Apêndice A

50 KAJFEZ, D. and Guillon, P., **Dielectric Resonators**, Artech House, Inc.,1986

51 MOSSO,M.M., **Apostila de Microondas**

Apêndice C

52 __, **Dispositivos Lógicos Programáveis**, Datapool.

53 __, **ByteBlasterMV Parallel Port Download Cable**, Datasheet Altera, April 2000, ver 2.0

Apêndice A

Gerador de RF desenvolvido no CETUC

Um oscilador senoidal é basicamente formado por um amplificador e uma malha de realimentação positiva, como se vê na Figura 1 a seguir.

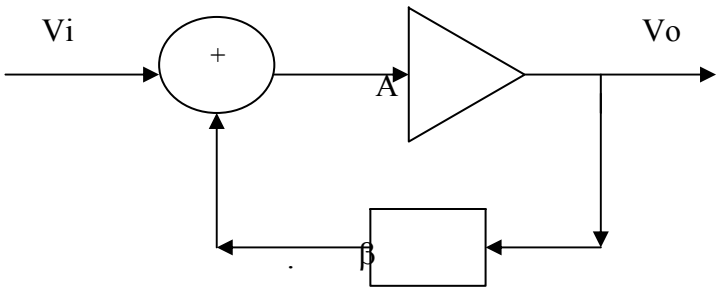


Figura 1 - Diagrama em Blocos de um Circuito Oscilador

A função de transferência do sistema, que fornece o ganho do amplificador com realimentação, é da forma:

$$H(jw) = A(jw)/(1 - \beta(jw).A(jw))$$

onde A é o ganho do amplificador, β é o ganho de malha aberta do circuito oscilador, V_i e V_o são, respectivamente, as tensões de entrada e saída do circuito.

Observa-se que, quando o ganho β.A é igual à unidade em determinada freqüência f_o, a relação V_o/V_i é infinita, significando que o circuito terá saída finita se a entrada for nula. Neste caso:

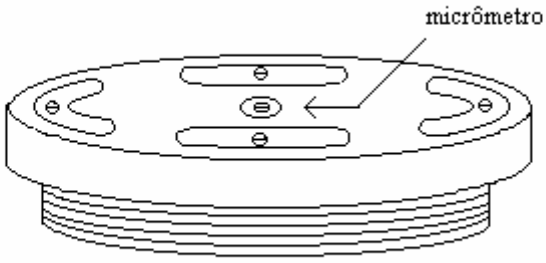
$$H_r(jw_o) = \beta(jw_o).A(jw_o) = 1$$

onde $H_r(jw_o)$ é o ganho da malha de realimentação na freqüência de ressonância f_o.

Pela equação anterior, verifica-se que a defasagem ao redor da malha é igual a 360°. Assim, a condição de oscilação senoidal é que o ganho da malha de realimentação seja unitário e a defasagem na mesma seja igual a 360°.

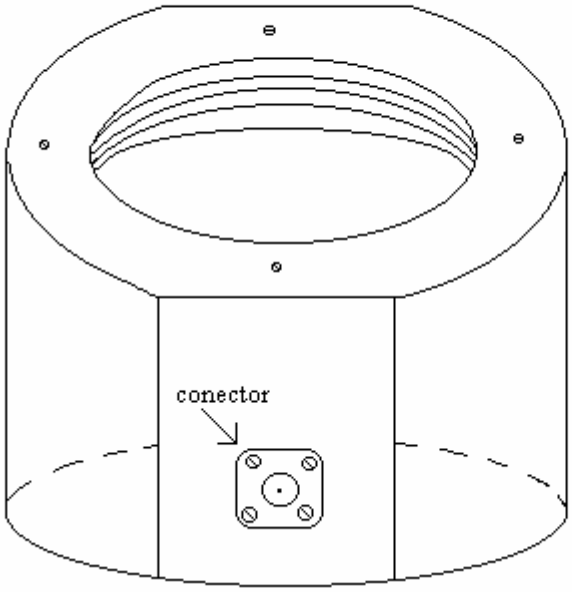
A obtenção de osciladores de microondas estabilizados é alcançada através da geração direta com estabilização por ressoadores dielétricos (DR). Estes dispositivos são de material cerâmico e apresentam modos próprios de ressonância, sendo a freqüência de ressonância dependente, principalmente, do valor da constante dielétrica relativa ε_r do material e da sua geometria. Segundo Kajfez [50], as duas maiores imperfeições dos DRs são as perdas no dielétrico e a dependência das propriedades elétricas e mecânicas com a temperatura, pois são elas que impõem limites nas duas propriedades mais importantes dos ressoadores dielétricos: o fator Q de qualidade elevado e a alta estabilidade com a temperatura. Para a aplicação nos osciladores de RF as pastilhas de DR empregadas devem, portanto, satisfazer às três condições básicas: alta constante dielétrica relativa na faixa de freqüências de interesse, com valores típicos na faixa de 15 a 50; perdas dielétricas básicas tal que o fator de qualidade seja alto, tipicamente maior que 3000 e coeficiente de temperatura de freqüência de ressonância baixo, permitindo a construção de ressoadores dielétricos bem estáveis com a temperatura. Valores típicos estão na faixa ± 5 ppm/ ^{o}C.

Dentre as diversas configurações possíveis, a aqui empregada para a confecção do gera-dor de RF envolve o DR cilíndrico colocado dentro de uma cavidade metálica cilíndrica. Empregando o ressoador D3500 a cavidade foi inicialmente testada com o ressoador dielé-trico sobre o suporte de alumina, no seu centro, tendo sido excitado por acoplamento elétrico. Por meio de roscas nas extremidades superior e inferior da cavidade, conforme mostra a Figura 2, a mesma pode ter sua altura variada de forma que diversos modos podem ser



obtidos, cada qual caracterizado por uma freqüência de ressonância.

(a)



(b)

Figura 2 – Vista da Cavidade Ressonante: a) Tampa superior e b) Corpo da Cavidade

Após os testes iniciais com a cavidade, o suporte foi trocado por outro de material com constante dielétrica próxima de 1, que forneceu melhores resultados. A freqüência desejada de 1,880 GHz foi obtida sendo o suporte colado na tampa inferior da cavidade. Sobre ele, colou-se o DR. Num passo seguinte, após nova sintonia, as tampas rosqueadas foram fixadas por parafusos de forma a manter a freqüência de ressonância invariável. Para um ajuste mais fino de freqüência, um micrômetro foi fixado na tampa superior da cavidade, como se pode observar na Figura 2(a), permitindo uma leve variação no acoplamento à cavidade, implicando numa ligeira variação da freqüência de ressonância. O fator de qualidade medido não carregado [51] foi de 9600, com uma perda de inserção de 22 dB. Feito isto, montou-se o sistema oscilador, cujo diagrama em blocos é mostrado na Figura 3. Neste sistema foram empregados: a cavidade ressonante confeccionada, um ressoador D3500 e seu suporte, dois circuladores D3C2040, um amplificador de baixo ruído ZKL-1724HLN, cuja figura de ruído é 1,5 dB, e um filtro passa-faixa MC1880-340-5AA, de 340 MHz. O curto móvel foi ajustado de forma que a defasagem total no loop fosse igual a 360°, ou seja, garantisse a ocorrência de

oscilação. Isto é conseguido quando se atinge o valor máximo dentre os picos ao se observar num analisador de espectro, conectado à saída do acoplador. A carga casada, no outro circulador, permite que a potência refletida no ressoador seja por ela absorvida, não danificando o amplificador, e o filtro passa-faixa permite que o modo de ressonância desejado, em 1880 MHz, seja selecionado.

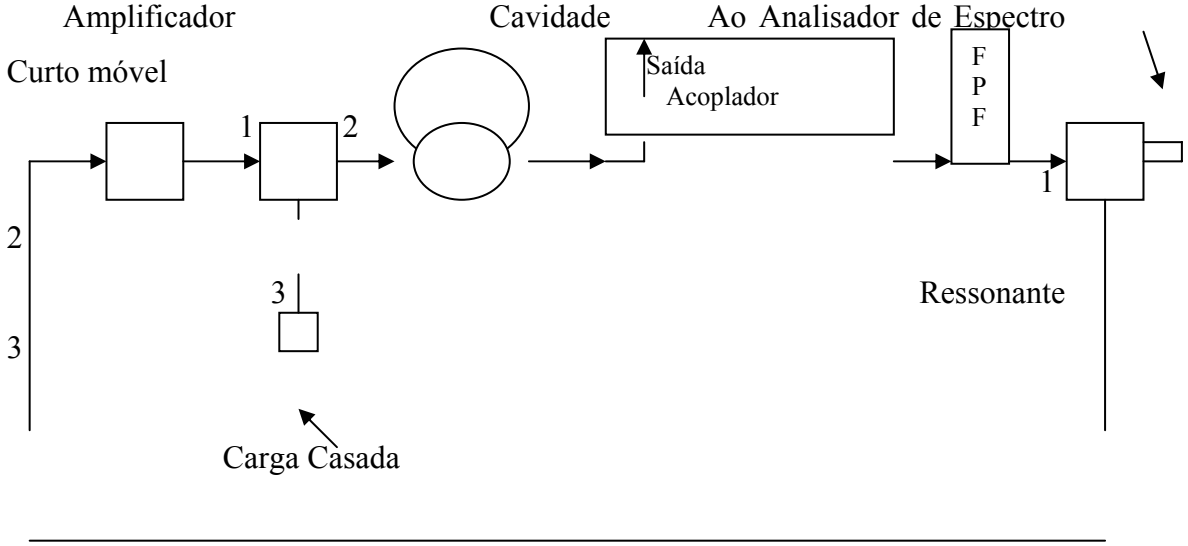


Figura 3 - Diagrama em Blocos do Gerador de RF

Montado o sistema da Figura 3 o curto móvel teve seu comprimento variado até que se obtivesse a ressonância da cavidade, onde se atingiu 6,57 dBm na saída, medido com o analisador de espectro. Trocou-se o curto móvel por conectores e uma carga em curto, num comprimento tal que garantisse a condição de ressonância acima. A seguir, com o micrô-metro, pode-se corrigir a freqüência para o valor desejado de 1,880 GHz, medindo-se com o contador.

Como o nível de sinal na entrada OL (oscilador local) do modulador I&Q empregado no sistema transmissor era 10 dBm, era necessário que a saída do oscilador fosse da ordem de 10 dBm e não 6,57 dBm. O que se fez para aumentar tal valor foi trazer o acoplador direcional para antes da cavidade ressonante, sendo inserido entre o circulador e a mesma. Com isto atingiu-se da ordem de 15 dBm na saída e um atenuador de 5 dB/ 2 W foi inserido à saída do oscilador montado de forma que o mesmo fornecesse 10 dBm à sua saída.

Pronto o oscilador, os componentes foram fixados numa base metálica que foi introdu-zida numa caixa confeccionada para servir de invólucro para o mesmo,

conforme é mostra-do na Figura 4, onde são mostrados dois terminais (+15 e 0 volts), para a alimentação do amplificador, e o terminal de saída de RF. Um suporte de espuma foi empregado entre a base metálica e a caixa de forma a amortecer qualquer movimento que se desse ao sistema, garantindo maior estabilidade na freqüência do oscilador.

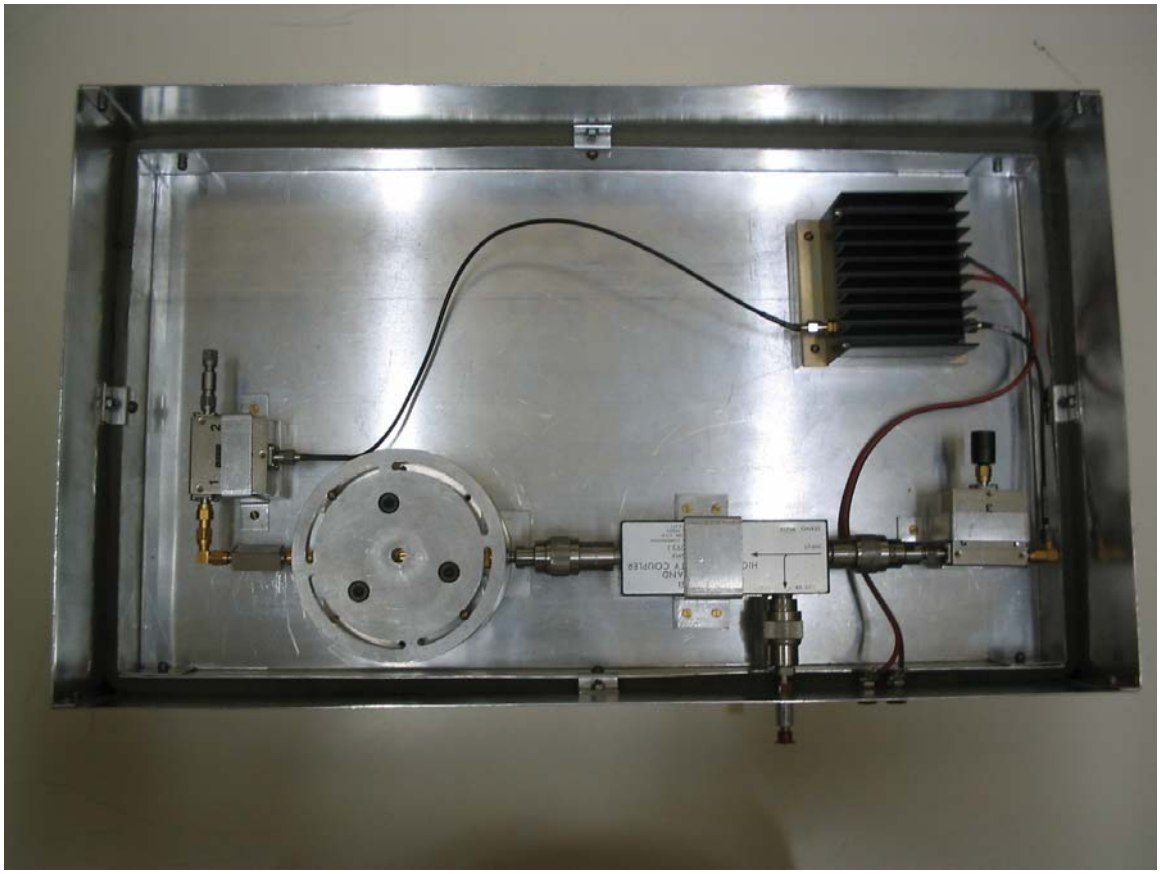


Figura 4 - Gerador de RF Desenvolvido no CETUC.

Pronto o gerador, realizou-se o teste de temperatura e o de estabilidade de freqüência, deixando o equipamento ligado por um dia. Assim se observou que o aquecimento máximo do amplificador foi 35,1^{o} C (no ambiente refrigerado) e a freqüência obtida variava de 1 880 003 550 a 1 880 003 580 Hz, portanto numa faixa igual a 30 Hz. Esta faixa, porém, ainda era elevada para o propósito desejado, já que era preciso sincronismo no sistema.

Os geradores de RF encontrados no mercado funcionam, em geral, com uma referência externa de um padrão estável, de forma que possam dar saídas mais estáveis. Testes foram realizados nos geradores do laboratório, empregando

o padrão de freqüência de rubídio e constatou-se o fato: o gerador variava 1 Hz a cada 2 minutos, garantindo grande estabilidade. Isto feito, testou-se o emprego do sinal estável do padrão de freqüência do rubídio, amplificado por um LNA, através da saída do acoplador direcional do circuito montado, embora contrariando o sentido do acoplador. Dessa forma, não seria preciso alterar o circuito e se observaria se era possível excitar a cavidade com os 10 MHz do padrão, através de um harmônico em 1880 MHz, atingindo-se a estabilidade da freqüência gerada no circuito oscilador. Embora houvesse um harmônico em 1,88 GHz, sua intensidade não era suficiente para excitar a cavidade, pois estava em – 60 dBm. Assim, o circuito teria que ser alterado de forma que tornasse possível se referenciar o oscilador pelo padrão de rubídio. Optou-se, para dar continuidade ao trabalho de tese, por usar geradores de RF que puderam ser disponibilizados, embora, mais adiante seja retomado o oscilador confeccio-nado para que se insira a referência externa ao mesmo.

Apêndice B

Esquemáticos, *Layouts* e Disposição dos Componentes nas Placas Impressas

Neste apêndice mostra-se, na Figura 1, o diagrama esquemático da parte digital do circuito transmissor, identificando no mesmo os diversos blocos que a com-põem: circuito de entrada identificado por conversor, circuito de PLL, EPLD, circuito regulador de tensão, circuito de barramento de dados, circuito de barramento de endereços, memórias EPROM, memórias SRAM, circuitos conversor digital-analógico (DAC) e circuitos de amplificação. O *layout* da sua placa impressa pode ser visto na Figura 2, pelo lado da solda (LS). Na Figura 3 se acha o *layout* da face onde estão os componentes. Também é mostrada, na Figura 4, a disposição dos componentes na placa. Uma vez que a placa empregada no bloco digital do circuito receptor é a mesma do transmissor, onde apenas um dos ramos foi empregado, mostra-se, na Figura 5, os componentes que realmente foram soldados para a operação do bloco digital na recepção. Na Figura 6 pode-se ver os *layouts* das placas impressas relativas ao mixer mais integrador, empregadas no sistema de recepção, sendo o da esquerda relativo ao ramo Q e o da direita, ao ramo I. A Figura 7 mostra a disposição dos componentes nessas placas sendo a da esquerda referente ao ramo I e a da direita, ao ramo Q.

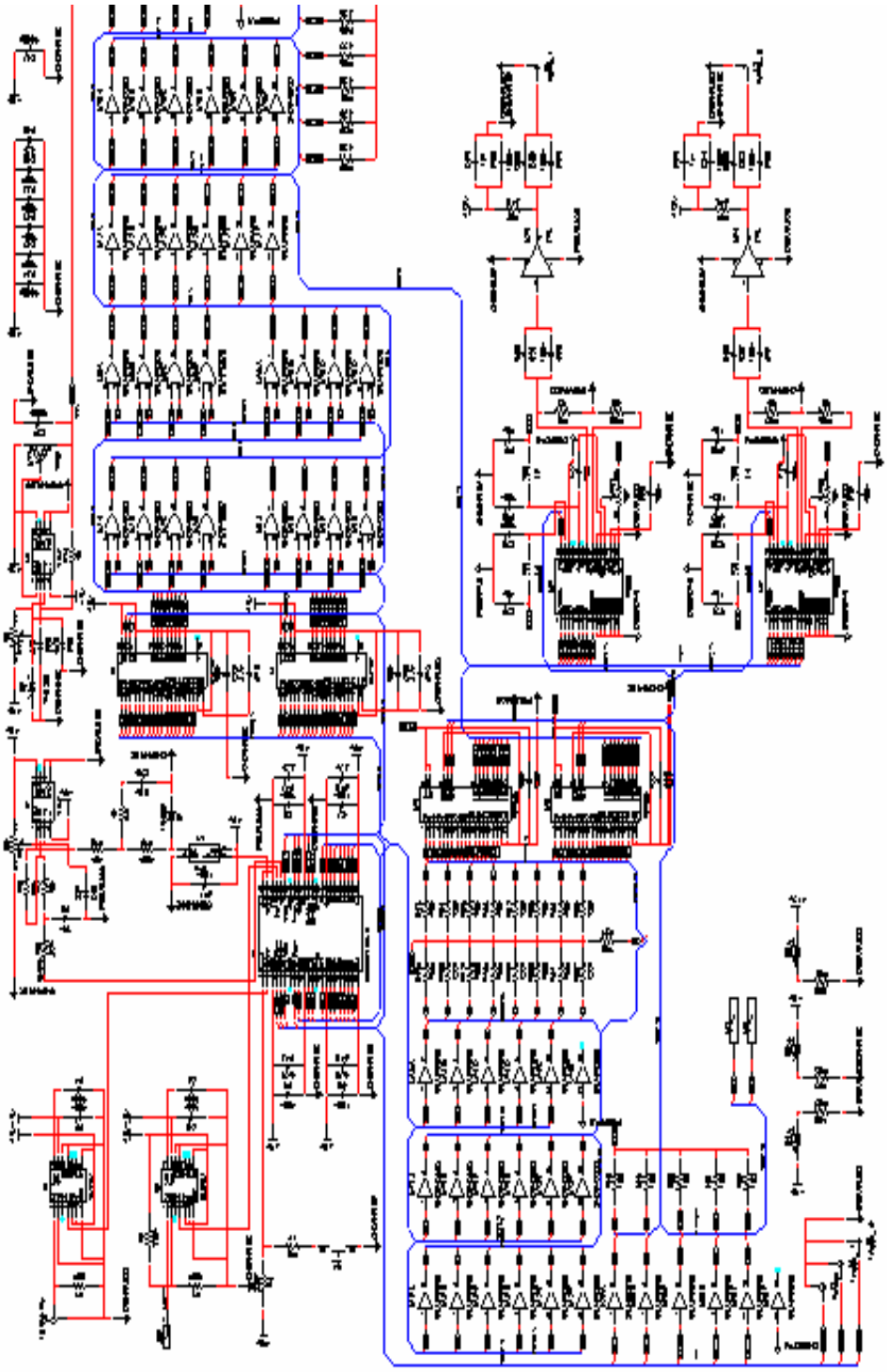


Figura 1 - Esquemático do Bloco Digital do Sistema Transmissor

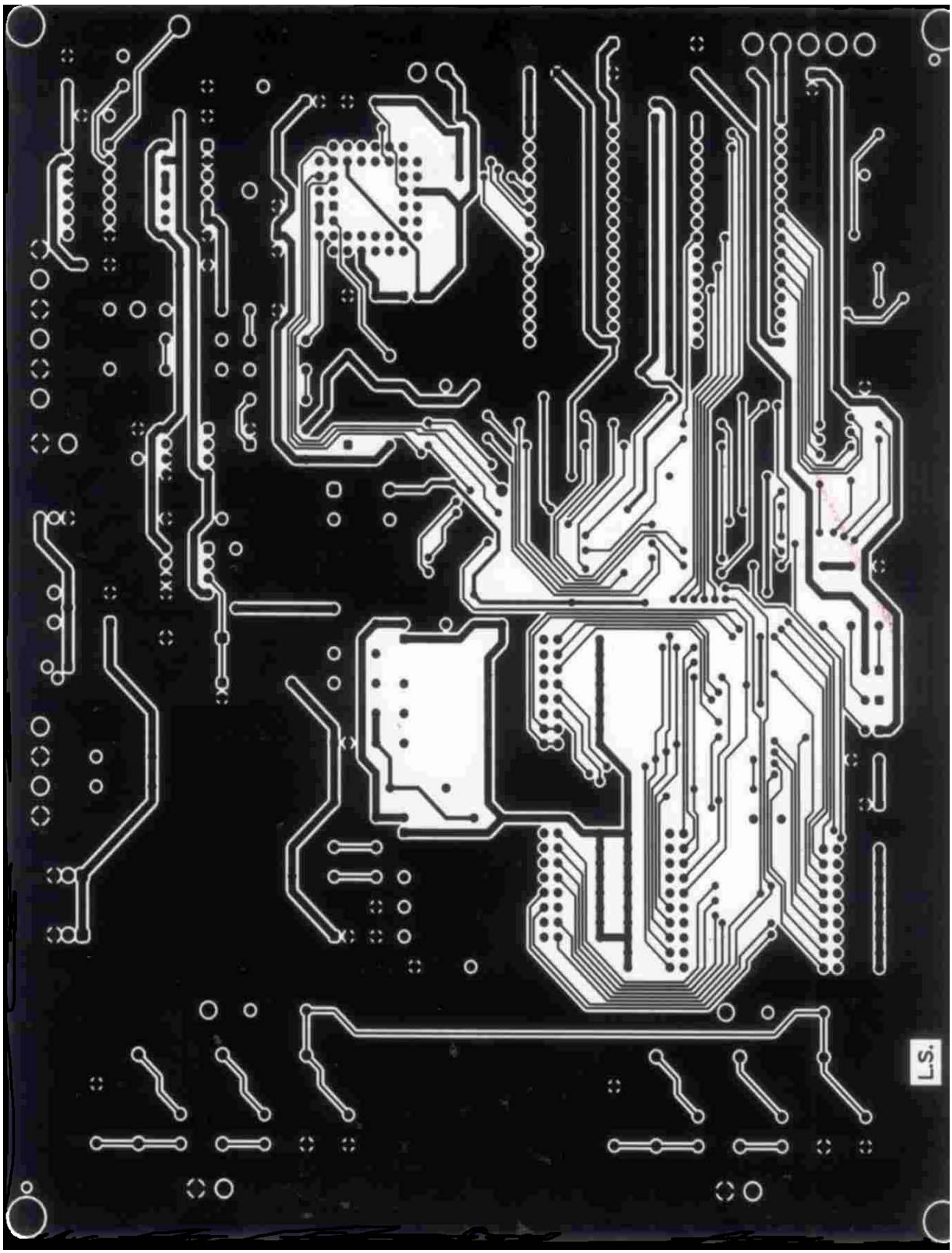


Figura 2 - Layout da Placa Impressa do Transmissor/Lado da Solda (L.S.)

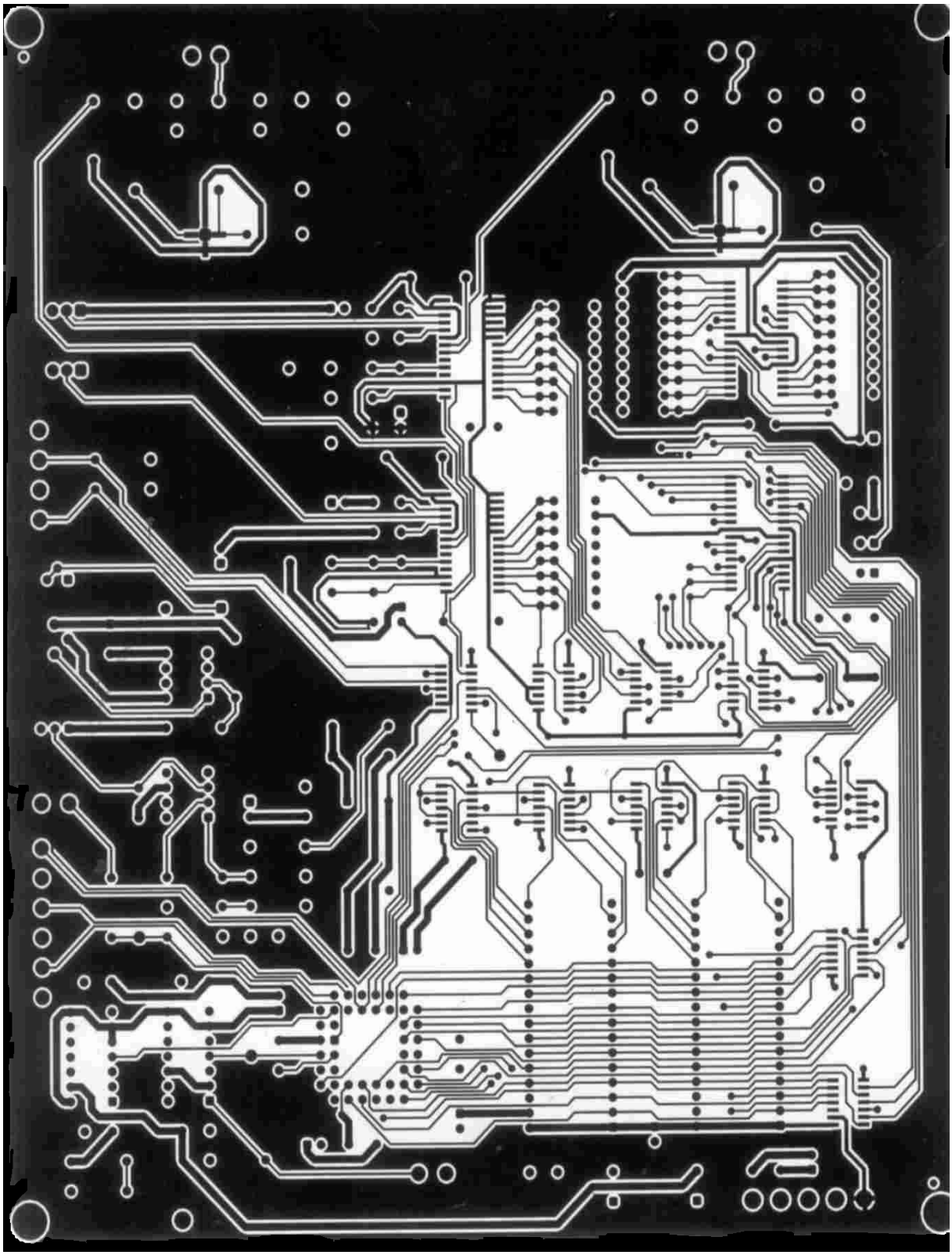


Figura 3 - Layout da Placa Impressa do Transmissor/Lado dos Componentes

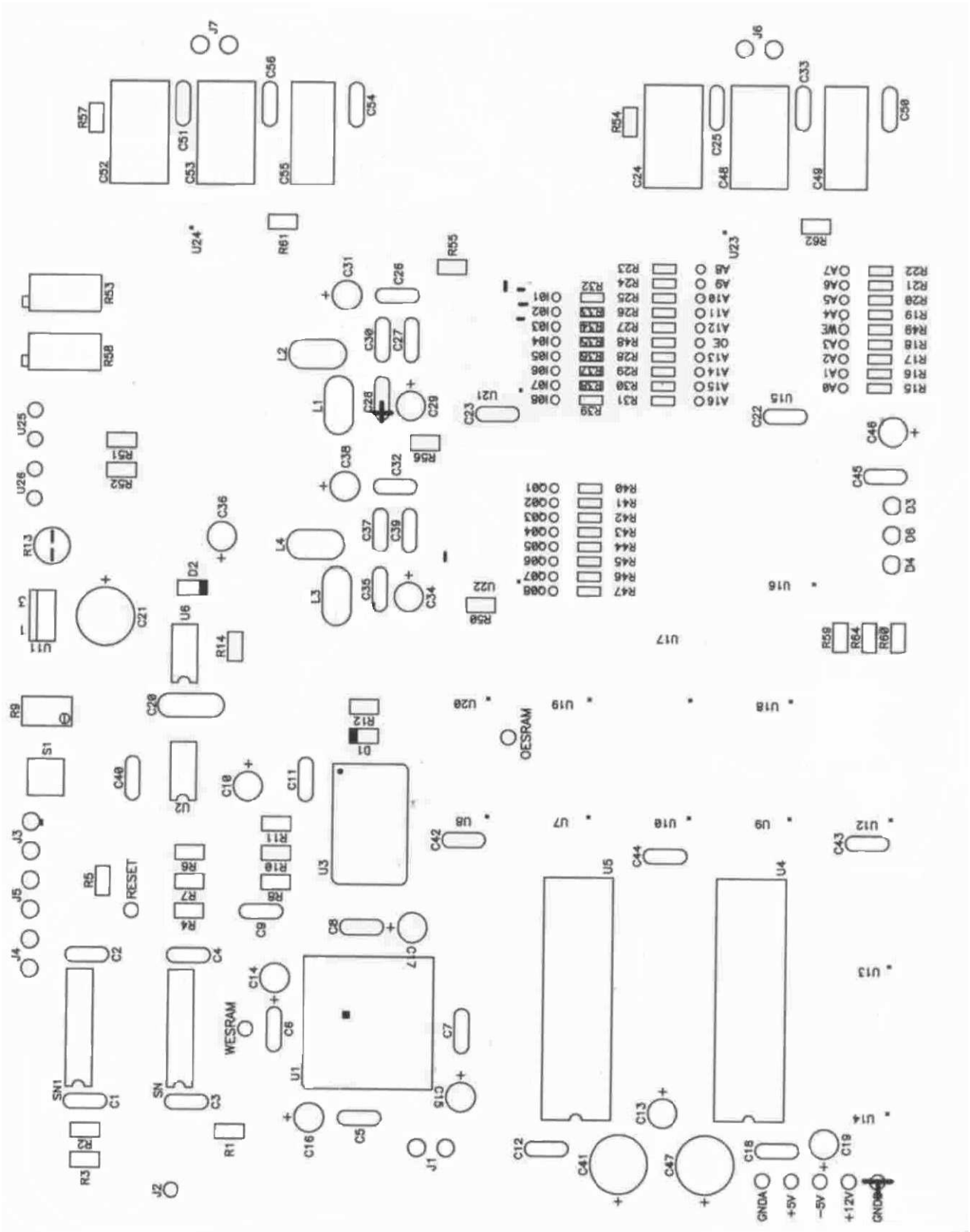


Figura 4 - Disposição dos Componentes na Placa Impressa do Transmissor

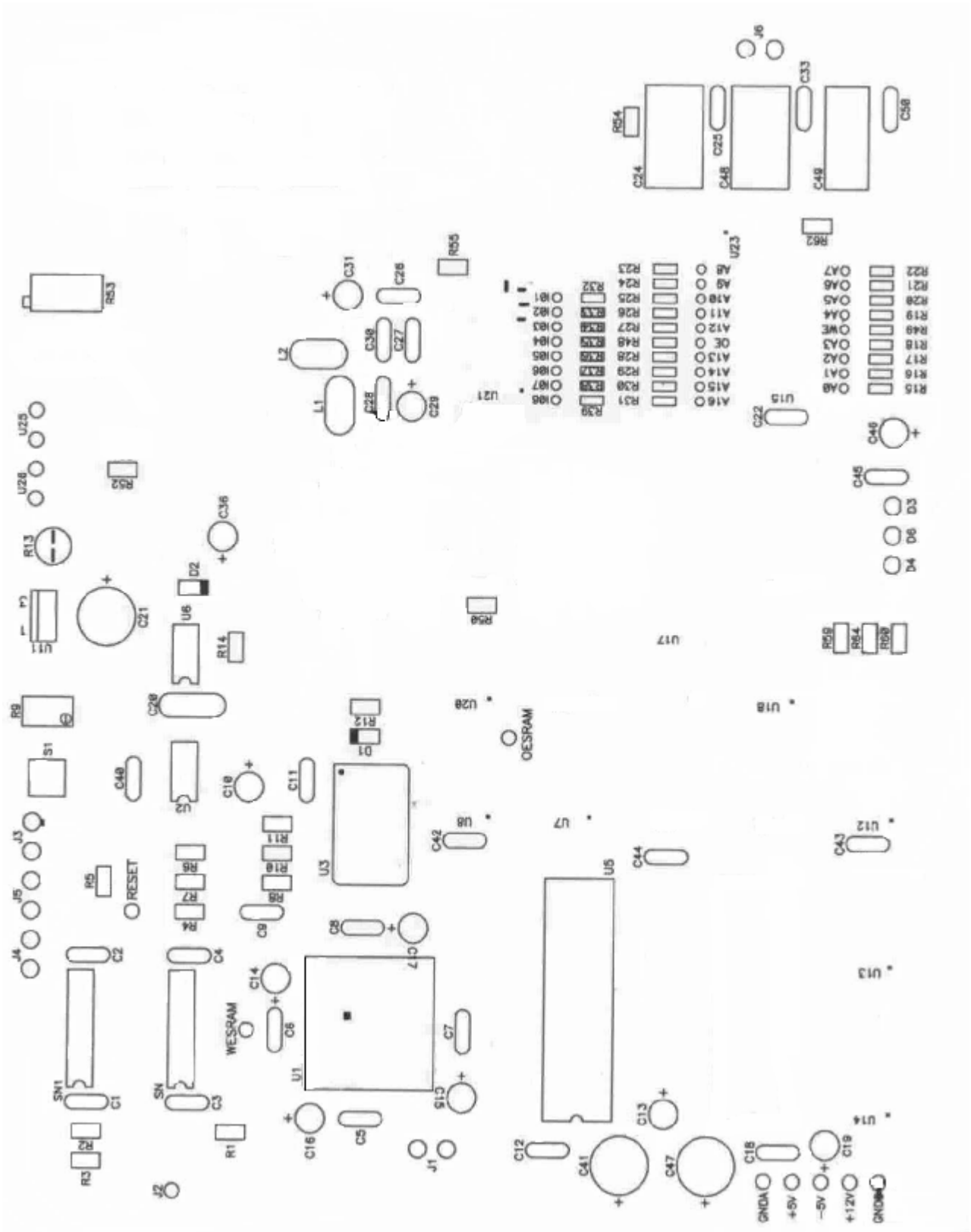
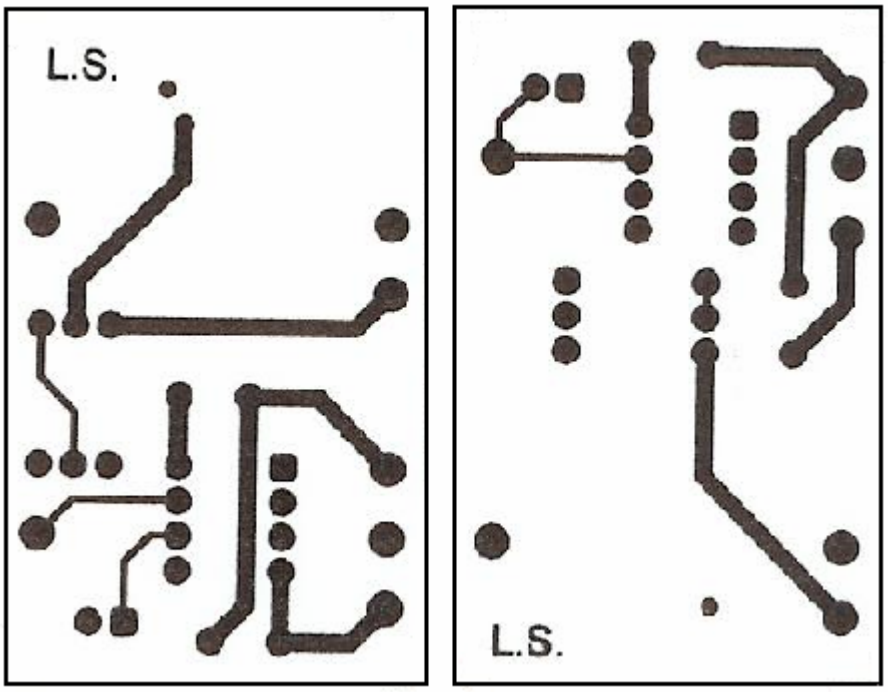
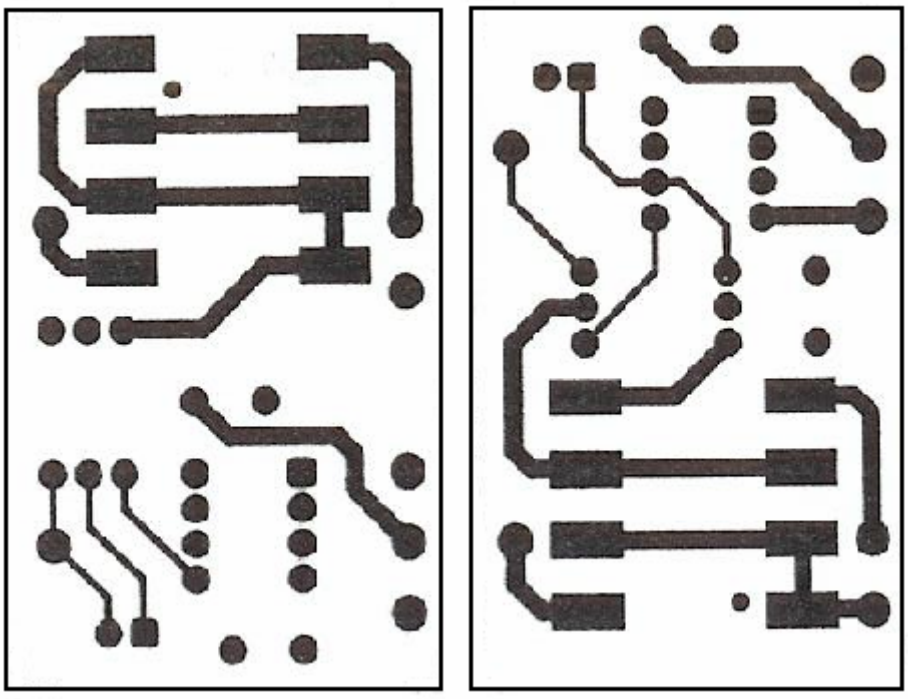


Figura 5 - Disposição dos Componentes na Placa Impressa do Receptor

***Layout* do Circuito do *Mixer* mais Integrador**



(a)



(b)

Figura 6 - Layout do Circuito do Mixer mais Integrador
(a) Lado da Solda e (b) Lado dos Componentes

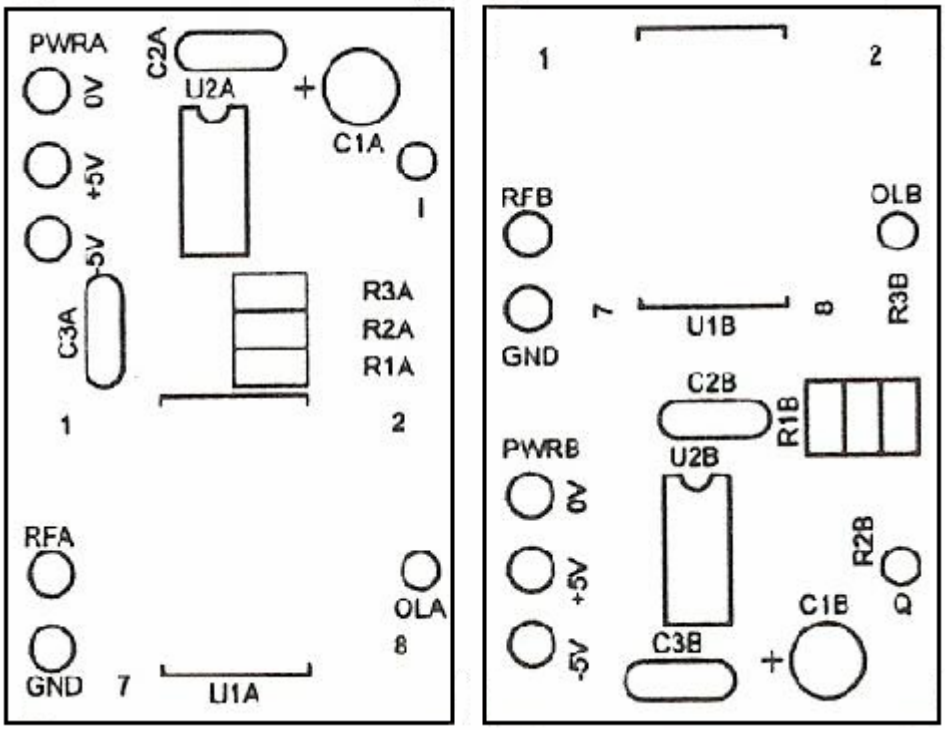


Figura 7 - Disposição dos Componentes na Placa Impressa do Mixer mais Integrador

Apêndice C

EPLD e Sistema Empregado para a sua Gravação

O EPLD, empregado em ambos circuito transmissor e receptor da sonda, é da família MAX 7000S, que incorpora a interface serial padrão IEEE 1149 ou JTAG (*Joint Test Action Group*). Essa família é de alta densidade, com programação de alta *performance* e baseada em EEPROM (*Electrically Erasable Programable Memory*).

Através de um *software* é possível personalizar o EPLD empregando diagramas esquemáticos, tabelas verdadeiras de tensão, equações *booleana*s, formas de ondas de sinais de entrada e saída ou linguagem de descrição de *hardware* (HDL - ***H**ardware **D**escription **L**anguage*). A referência [52] trata detalhadamente dessas formas de personalização. Após simulação e verificação do funcionamento do mesmo, um *hardware* específico programou o EPLD para que pudesse ser utilizado no sistema transmissor da sonda. Tal programação foi realizada com o dispostivo de progra-mação serial denominado *byteblaster* com os recursos do *software* MAX+ PLUSII da Altera, através de esquemático. Detalhes sobre *byteblaster* são encontrados na referência [53].

O EPLD escolhido foi o EPM7064SLC44-6, onde 4 dos 44 pinos disponíveis são destinados à comunicação para programação do EPLD.

Dentre as principais vantagens do EPLD, podem ser citadas:

- simplificação no desenvolvimento de projetos, devido a sua programabilidade e reprogramabilidade, que permite que as funções sejam alteradas facilmente;
- simplificação no desenvolvimento de placas de circuito impresso, já que sua pinagem é determinada pelo projetista;
- consumo baixo de potência, pois são implementados em tecnologia CMOS;
- implementação de mais funções em uma mesma área de silício;
- redução no tamanho do circuito impresso, reduzindo o custo do mesmo;
- redução do tempo de obtenção final do circuito, permitindo futuras modificações.

O projeto desenvolvido foi programado serialmente no dispositivo, via conexão a um computador PC, através de um *byteblaster* e um adaptador PLCC para dip, que possibilitou a conexão do dispositivo ao *protoboard* do módulo digital 8810 da *Datapool*. Após a programação do EPLD, os recursos de entrada e saída do módulo puderam ser usados para realizar a simulação e verificação do projeto.

A Figura 1 mostra o a conexão do PC, via *Byteblaster*, para se realizar a programação do EPLD (componente mais à direita), que foi soquetado na placa de teste. Para tal, é necessário que o mesmo seja alimentado com 5 volts. Realizada a programação, o EPLD é sacado do soquete e introduzido no sistema onde irá operar, neste caso, nas placas digitais dos sistemas transmissor e receptor.

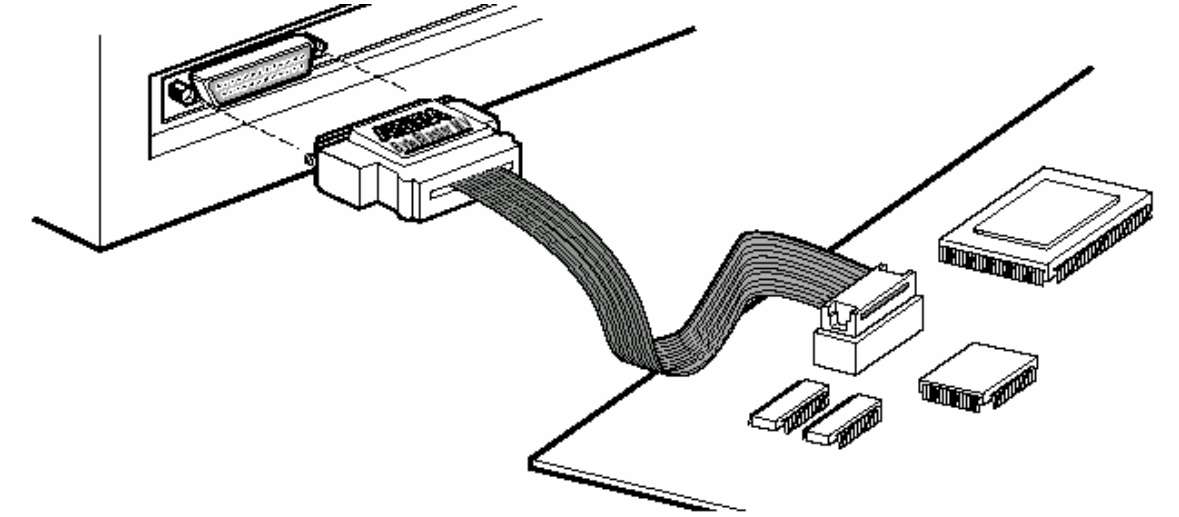


Figura 1 - Cabo de Conexão da Porta Paralela do PC à Placa de Teste

A Figura 2 mostra a identificação da pinagem do EPLD após sua programação final. O pino 39, embora escrito *RESERVED* é relativo ao segundo *trigger*, embora não se tenha escrito. Cuidado foi tomado para que tal pinagem se mantivesse a mesma, mesmo quando houve a necessidade de reprogramação do EPLD, conforme citado na seção 3.4. Com isso, pode-se continuar com a mesma placa impressa, já que a modificação foi realizada apenas no circuito interno do EPLD. Os pinos denominados TCK, TDO, TDI e TMS são reservados para a

alimentação do EPLD no momento da gravação. Os pinos VCC e GND são para a alimentação do EPLD no circuito.

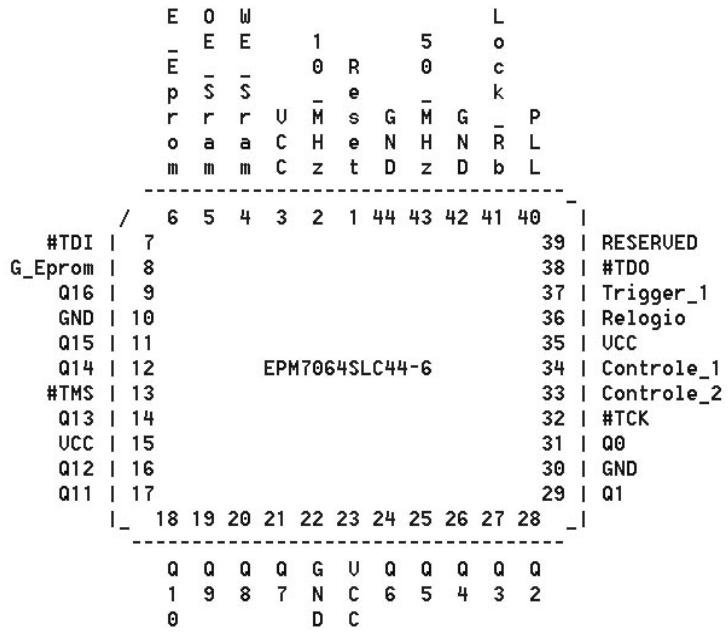


Figura 2 - Disposição Final dos Pinos de Entrada e Saída do EPLD

Apêndice D

Programação *MATLAB* Empregada na Aquisição das Amostras

1. Aquisição *On Line*

Na tela do *laptop*, enquanto as medidas são realizadas, um em cada 10 perfis aparece na tela, conforme a programação de aquisição dos dados Daqiq.m, detalhada a seguir. Tal programação é necessária para que os sinais de tensão I, Q e D sejam aquisitados e gravados e, também, para tornar possível a visualização dos perfis na tela do *laptop*, simultaneamente com os pulsos do sensor da roda. Os perfis são calculados da forma ($I^2 + Q^2$) e convertidos para dBm normalizados.

Programa Daqiq

```
%Daqiq.m - Aquisição dos dados e programação da tela do Laptop

function Daqiq(varargin)
%   Programa de Aquisicao de Dados
%   Para que o programa funcione corretamente, é necessário que a placa de
%   aquisição da NI esteja instalada no computador. Caso contrário, o programa
%   será inicializado com a placa de rede.
%
%   Este programa aquisitará os dados de potência obrigatoriamente, sendo
%   necessário a seleção de 2 canais. Há a opção de aquisição de outro
%   canal com outra aplicação como, por exemplo, trigger ou sensor de posição
%
%   Ao ser iniciada a aquisição, o autoset é setado automaticamente, mas
%   é possivel desativá-lo e reativá-lo quando desejar.
%
%   A escala de visualização pode ser escolhida pelo usuário através da caixa
%   unidades/div e pelo nível de referência. Quando este está desativado, o nível
%   de referência é zero. Este nível indica o centro do eixo y.
%   A aquisicao é feita a cada 51.1ms, porém o gráfico é mostrado de 10
%   em 10 ciclos. A taxa de amostragem, que deve ser um parâmetro de entrada,
%   define quantas amostras por segundo serão aquisitadas, logo o número de
%   amostras por trigger será a taxa*0.0511.
%
%   Os dados são salvos no fim de cada iteração num arquivo .txt. Ao
%   abrí-lo no matlab os dados são, automaticamente, convertidos para
%   matrizes, facilitando a análise e manipulação.
%
%   O nome do arquivo deve ser entrado como parâmetro. O “default” é
%   teste.txt, porém a cada vez que o programa abre, este arquivo é
%   reiniciado. Ao se pressionar a tecla 'Enter' no teclado, apaga-se o
%   arquivo e ele pode ser regravado a partir do zero.


% Parâmetros de saída não são suportados;
if nargout > 0
    error('Nao exsitem parametros de saida.');
end

% De acordo com o numero de parametros de entrada, chama a funcao correta;
```

```
switch nargin
    case 0
        % Cria a figura e inicializa a strutura de dados;
        data = Janela;
        fig = data.handle.figure;
    case 1
        error('Nao sao necessarios parametros de entrada.');
    case 2
        % Funcoes que sao chamdas quando a funcao Daqiq eh chamada com 2
        % parametros: o primeiro eh uma palavra chave e o segundo a figura
        % do programa que tem armazenada nela a estrutura de dados.
        acao = varargin{1};
        fig = varargin{2};
        data = get(fig, 'UserData');
        switch acao
            case 'arqsaida'
                data = Arqsaida(data);
            case 'taxa'
                data = Taxa(data);
            case 'autoset'
                data = Autoset(data);
            case 'manual'
                data = Manual(data);
            case 'canal'
                data = Mudacanal(data);
            case 'graf'
                data = Grafico(data);
            case 'inicia'
                data = Inicia(data);
            case 'para'
                data = Para(data);
            case 'reinicializa'
                data = Reinicializa(data);
            case 'fecha'
                Fecha(data);
            case 'daqhelp'
                DaqHelp;
            case 'ajuda';
                Ajudasa;
        end
    case 3
        % Funcoes que sao chamdas quando a funcao Daqiq eh chamada com 3
        % parametros: o primeiro eh uma palavra chave, o segundo eh um
        % numero  e o terceiro a figura do programa que tem armazenada
        % nela a estrutura de dados.
        if ~(isa(varargin{1},'analoginput'))
            % Inicializa variaveis;
            acao = varargin{1};
            numero = str2num(varargin{2});
            fig = varargin{3};
            data = get(fig, 'UserData');
            % Chama a funcao apropriada do callback;
            switch acao
                case 'escala'
                    data = Escala(data,numero);
                case 'unidade'
                    data = Unidade(data,numero);
                case 'referencia'
                    data = Referencia(data,numero);
            end
        % Neste caso, o primeiro parametro eh o objeto da placa, o segundo
        % um evento e o terceiro um parametro. Por definicao, qualquer
        % funcao do tipo callback recebe pelo menos 2 parametro: obj e
        % event.
        elseif (isa(varargin{1},'analoginput'))
            obj = varargin{1};
            event = varargin{2};
            acao = varargin{3};
            switch acao
                case 'timeraction'
                    data = Repeticao(obj,event);
            end
            return;
        end
    case 4
        % Neste caso, os 2 primeiros parametros nao sao importantes, mas o
        % 3o e o 4o sao como os 2 primeiros do 'case 2'.
```

```
        acao = varargin{3};
        fig = varargin{4};
        data = get(fig, 'UserData');
        switch acao
            case 'para'
                data = Para(data);
        end
    otherwise
        error('Ha parametros de entrada em excesso.');
end

% Atualiza os dados da tela;
if ~isempty(fig)&ishandle(fig),
    set(fig,'UserData',data);
end

% Armazena as estruturas no objeto;
if ~isempty(data.ai) & isvalid(data.ai)
    set(data.ai, 'UserData', data);
end

%*********************************************************************
% Cria arquivo para serem gravados os dados;
function data = Arqsaida(data)

%le o nome do arquivo se ele existir. Este arquivo deve ser texto;
if exist(data.arquivo)  & (~isempty(findstr('.txt',data.arquivo)))
    lasterr = [];
    try
        dados = load(data.arquivo);
    catch
        aviso = warnlg('O arquivo especificado está corrompido ou não foi gerado
por
este programa','Erro de Leitura');
    end
    % Se nao houver erro, le o arquivo texto e converte para '.mat' gravando
    % as variaveis desejadas Q, I, T, D; Como podem ter de 2 a 4 variaveis,
    % o programa checa se as 2 ultimas existem antes de grava-las;
    if isempty(lasterr)
        arquivo = data.arquivo(1:length(data.arquivo)-4);
        tam = size(dados,2);
        Q = dados(:,1);
        I = dados(:,2);
        save(arquivo,'Q','I');
        if tam>2
            T = dados(:,3);
            save(arquivo,'T','-append');
            if tam==4
                D = dados(:,4);
                save(arquivo,'D','-append');
            end
        end
    end
    % Apaga o arquivo texto;
    delete(data.arquivo);
end
% Le o nome do novo arquivo;
arquivo = get(data.handle.uicontrol(2),'String');
% Verifica se a pessoa digitou '.mat' ou nao;
% Caso tenha, substitui por '.txt' e senao,o adiciona;
if isempty(findstr('.mat',arquivo))
    arquivo = [arquivo,'.txt'];
else
    arquivo(length(arquivo)-3:length(arquivo))='.txt';
end
% O arquivo onde os dados sao gravados durante a duracao do programa eh do tipo
texto;
% Verifica se o arquivo texto existe e apaga-o. Se nao existir, eh
% necessario alterar o nome do arquivo gravado na estrutura;
if exist(arquivo) & strcmp(arquivo,data.arquivo)
    delete(arquivo);
else
    data.arquivo = arquivo;
end
%o arquivo sera gravado a cada iteracao;
% *********************************************************************
% Esta funcao apenas define qual grafico sera exibido na tela de baixo,
```

```
% caso ambos os canais(trigger e posicao) tenham sido ativados;
function data = Grafico(data)
data.graf = get(data.handle.uicontrol(24),'Value');
% ********************************************************************
% Muda canais a serem plotados;
function data = Mudacanal(data)

% Le os canais de i e q selecionados;
canal = cell2mat(get(data.handle.uicontrol(6), {'Value'}));
% 2 canais devem ser selecionados;
if length(canal) ~= 2
    aviso = findobj(findall(0), 'Tag', '2canais');
    if isempty(aviso)
        aviso=warndlg('2 canais devem ser selecionados.','Erro de Configuracao');
        set(aviso, 'Tag','2canais');
    else
        figure(aviso(1));
    end
    % Corrige a lista colocando apenas 2 canais caso + tenham sido
    % selecionados;
    if length(canal) > 2
        set(data.handle.uicontrol(6),{'Value'},{canal(1:2)});
    end
    data.canais=[];
    % Sai da funcao;
    return;
end
% canal de trigger nao precisa ser obrigatoriamente selecionado;
% ele so sera ativado se a unidade da escala de baixo estiver selecionada;
if (data.unidades(2)~=1)
%   Checa se existem canais repetidos;
    canal(3) = get(data.handle.uicontrol(12),'Value');
    if canal(3)==canal(1) | canal(3) == canal(2)
        aviso = findobj(findall(0), 'Tag', 'canaisiguais1');
        if isempty(aviso)
            aviso=warndlg('O canal de trigger nao pode ser o mesmo dos canais I e
Q.','Erro de Configuracao');
            set(aviso, 'Tag','canaisiguais1');
        else
            figure(aviso(1));
        end
        data.canais=[];
        return;
    end
    % Le o canal de posicao se o de trigger estiver ok;
    c = get(data.handle.uicontrol(22),'Value');
    if c==canal(1) | c==canal(2) | c==canal(3)
        aviso = findobj(findall(0), 'Tag', 'canaisiguais1');
        if isempty(aviso)
            aviso=warndlg('O canal de posicao nao pode ser o mesmo dos canais I e
Q
nem ao de trigger.','Erro de Configuracao');
            set(aviso, 'Tag','canaisiguais1');
        else
            figure(aviso(1));
        end
        data.canais=[];
        return;
    elseif c~=1
        canal(4)=c;
    end
end
%Para a aquisicao, no caso de haver ocorrido algum erro;
if isvalid(data.ai) & strcmp(lower(get(data.ai, 'Running')), 'on')
    stop(data.ai);
end
% Apaga todos os canais existentes;
delete(data.ai.channel);
%Cria os canais necessarios e seta alguns parametros para maior precisao;
for i=1:length(canal)
    addchannel(data.ai,canal(i)-1);
end
sa.channel.SensorRange = [-.5 .5];
sa.channel.UnitsRange = [-.5 .5];
% Armazena os canais iq e de posicao se houver;
data.canais.qi = canal(1:2);
if length(canal) > 2
```

```
    data.canais.trig = canal(3);
    if length(canal)==4
        data.canais.pos = canal(4);
    else
        data.canais.pos = [];
    end
else
    data.canais.trig = [];
    data.canais.pos = [];
end
% ********************************************************************
% Esta funcao permite ajustar manualmente a escala de visualizacao do
% primeiro grafico;
function data = Manual(data)
% Se o grafico ja esta em modo manual, continua no manual;
if data.autoset==0
    set(data.handle.uicontrol(20),'Value',1);
else
    % Senao, desliga o autoset e ativa todos os parametros de visualizacao;
    set(data.handle.uicontrol(19),'Value',0);
    set(data.handle.uicontrol(7:10),'Enable','On');
    set(data.handle.uicontrol(9),'Value',1);
    data.check(1)=1;
    data.autoset=0;
    data = Escala(data,1);
end
% ********************************************************************
% Esta funcao ativa o autoset para escala de visualizacao do primeiro
% grafico;
function data = Autoset(data)
% O autoset so pode ser ativado quando a aquisicao esta ativada;
if strcmp(get(data.ai, 'Running'), 'Off')
    set(data.handle.uicontrol(19), 'Value', 0);
    aviso = findobj(findall(0), 'Tag', 'autoset');
    if isempty(aviso)
        aviso = warndlg('Aquisição deve estar ativada.','Aviso de Aquisição');
        set(aviso, 'Tag', 'autoset');
    else
        figure(aviso(1));
    end
    return;
else
    % caso o autoset ja esteja selecionado;
    if data.autoset==1
        set(data.handle.uicontrol(19),'Value',1);
        return;
    end
    % Comeca o autoset propriamente dito;
    %inicializa variaveis;
    taxa=floor(data.taxa*.047);
    canais = data.canais;
    unidades=data.unidades;
    cont=data.cont;

    % Desliga o manual;
    set(data.handle.uicontrol(20),'Value',0);
    set(data.handle.uicontrol(7), 'Enable', 'off');
    set(data.handle.uicontrol(9:10), 'Enable', 'off');

    % Aquisita uma amostra para fazer o autoset;
    pause(1);
    d = peekdata(data.ai, get(data.ai, 'SamplesPerTrigger'));
    dados(:,1)= d(:,1).^2 + d(:,2).^2;
    if ~isempty(canais.trig)
        if data.graf==1
            dados(:,2)=d(:,3);
        else
            dados(:,2)=d(:,4);
        end
    end
    for i=1:size(dados,2)
        switch unidades(i)
            case 2
                dados(:,i) = 1000*dados(:,i);
            case 4
                dados(:,i) = 10*log10(1000*dados(:,i));
                dados(:,i) = dados(:,i) - max(dados(:,i));
```

```
        end
    end

    % Atualiza o grafico;
    for i=1:size(dados,2)
        set(linha(i), 'Parent', data.handle.axes(i),...
            'XData',    1:length(dados),...
            'YData',    dados(:,i));
    end

    % Verifica os limites inferior e superior;
    yinf = min(dados(:,1));
    ysup = max(dados(:,1));

    % Se NaNs sao retornados ou se ambos forem 0;
    if (isnan(yinf) & isnan(ysup)) | (yinf ==0 & ysup ==0)
        yinf = -0.01;
        ysup = 0.01;
    elseif isnan(yinf)
        yinf = ysup - (abs(ysup)/2);
    elseif isnan(ysup)
        ysup = yinf + (abs(yinf)/2);
    end

    % Quando os limites sao iguais;
    if yinf == ysup
        yinf = yinf - (abs(yinf)/2);
        ysup = ysup + (abs(ysup)/2);
    end

    % Ajusta a escala em y;
    set(data.handle.axes(1), 'YLim', [yinf ysup],...
        'YTick', linspace(yinf, ysup, 11),...
        'YTickLabel', {num2str(yinf,4), '','','', '','',...
        '','','','',num2str(ysup,4)});

    % Salva a amostra e atualiza o nivel de referencia e a escala;
    save(data.arquivo,'d','-ascii','-append');
    % Seta a escala/div e o centro;
    set(data.handle.uicontrol(7), 'String', num2str((ysup-yinf)/10,3));
    set(data.handle.uicontrol(10),'String',num2str((ysup+yinf)/2));
    data.autoset = 1;
    data.cont=cont+1;
    drawnow;
end

% ********************************************************************
% Para a varredura e fecha a janela;
function Fecha(data)

% Para a varredura se ela estiver ocorrendo;
if isvalid(data.ai) & strcmp(lower(get(data.ai, 'Running')), 'on')
    stop(data.ai);
end
% Verifica se ha um arquivo nao salvo;
if exist(data.arquivo)  & (~isempty(findstr('.txt',data.arquivo)))
    dados = load(data.arquivo);
    arquivo = data.arquivo(1:length(data.arquivo)-4);
    tam = size(dados,2);
    Q = dados(:,1);
    I = dados(:,2);
    save(arquivo,'Q','I');
    if tam>2
        T = dados(:,3);
        save(arquivo,'T','-append');
        if tam==4
        D = dados(:,4);
        save(arquivo,'D','-append');
        end
    end
    delete(data.arquivo);
end
% Apaga o objeto;
delete(data.ai);

% Fecha a janela;
delete(data.handle.figure);
```

```
% ********************************************************************
% Reinicializa o programa com todos os parametros originais;
function data = Reinicializa(data)
% Verifica se a aquisicao esta ativada e para;
if (isvalid(data.ai) & data.estado==1)
    stop(data.ai);
    delete(data.ai.channel);
    % Permite ao usuario mudar os canais,taxa de amostragem e arquivo de
    % saida;
    for i=0:1
        set(data.handle.uicontrol(6+6*i), 'Enable', 'on');
        set(data.handle.uicontrol(2+2*i), 'Enable', 'on');
    end
    % Retorna o botao para Inicio;
    set(data.handle.uicontrol(17), 'Enable', 'on');
    set(data.handle.menu(5), 'Enable' ,'on');
    set(data.handle.menu(6), 'Enable' ,'off');
end
% Salva os dados se estiverem sendo aquisitados;
if exist(data.arquivo)  & (~isempty(findstr('.txt',data.arquivo)))
    dados = load(data.arquivo);
    arquivo = data.arquivo(1:length(data.arquivo)-4);
    tam = size(dados,2);
    Q = dados(:,1);
    I = dados(:,2);
    save(arquivo,'Q','I');
    if tam>2
        T = dados(:,3);
        save(arquivo,'T','-append');
        if tam==4
        D = dados(:,4);
        save(arquivo,'D','-append');
        end
    end
    delete(data.arquivo);
end

% Configuracoes iniciais;
data.estado = 0;
if ~isempty(data.handle.line)
    delete(data.handle.line);
    data.handle.line = [];
end

data.autoset=0;
set(data.handle.uicontrol(19),'Value',0);
set(data.handle.uicontrol(20),'Value',1);

% se o canal 2 tiver sido desabilitado, habilita-o;
if get(data.handle.axes(2),'Color') == data.cordefundo
    set(data.handle.uicontrol(11:15),'Enable','on');
    set(data.handle.axes(2),'Color',[1 1 1]);
end

set(data.handle.uicontrol(21:24),'Enable','on');
data.canais.qi=[3 4];
set(data.handle.uicontrol(6),{'Value'},{data.canais.qi});
data.canais.trig=5;
set(data.handle.uicontrol(12),{'Value'},{data.canais.trig});
data.canais.pos=1;
set(data.handle.uicontrol(22),{'Value'},{data.canais.pos});
data.graf = 1;
data.escalas(1:2)=[.1 .1];
data.unidades(1:2)=[4 1];
data.referencias=[0 0];
data.check=[0 0];
for i=0:1
    set(data.handle.uicontrol(7+i*6),'String',num2str(data.escalas(i+1)));
    set(data.handle.uicontrol(8+i*6),'Value',data.unidades(i+1));
    set(data.handle.uicontrol(9+i*6),'Value',data.check(i+1));
    set(data.handle.uicontrol(10+i*6),'String',num2str(data.referencias(i+1)));
    set(data.handle.uicontrol(10+i*6),'Enable','Off');
end
data.taxa=1000;
set(data.handle.uicontrol(4),'String',num2str(1000));
data.arquivo='teste.mat';
set(data.handle.uicontrol(2),'String','teste.mat');
```

```
data.cont=0;

% ********************************************************************
% Ajuda para o Data Acquisition Toolbox;
function DaqHelp

doc('daq');

% ********************************************************************
% Ajuda para o Analisador;
function Ajudasa

doc('Daqiq');

% ********************************************************************
% Comeca a varredura;
function data = Inicia(data)

% Se a primeira unidade nao estiver selecionada;
if data.unidades(1)==1
    set(data.handle.uicontrol(17), 'Value', 0);
    aviso = findobj(findall(0), 'Tag', 'canais');
    if isempty(aviso)
        aviso=warndlg('O primeiro canal deve ser obrigatoriamente
selecionado','Erro
de Aquisicao');
        set(aviso, 'Tag','canais');
    else
        figure(aviso(1));
    end
    return;
end

%Cria canais;
data=Mudacanal(data);
if isempty(data.canais);
    warnlg('Nao ha canais selecionados.','Erro de Configuracao');
    return;
end

%Inicializa variaveis;
taxa=floor(data.taxa*.047);
unidades=data.unidades;
canais = data.canais;
cont=data.cont;

% Ajusta o eixo x;
for i=1:2
    set(data.handle.axes(i),...
        'XLim'              ,[1 taxa]    ,...
        'XTick'             ,linspace(1,taxa,9));
end

%Inicializa as escalas/divisao;
data=Escala(data,1);
data=Escala(data,2);

% A partir de agora os canais nao podem ser modificados;
for i=0:1
    set(data.handle.uicontrol(6+6*i), 'Enable', 'off');
    set(data.handle.uicontrol(2+2*i), 'Enable', 'off');
end

% Verifica se o grafico de baixo sera utilizado;
if unidades(2)==1 | isempty(canais.trig)
    set(data.handle.uicontrol(11:16),'Enable','off');
    set(data.handle.uicontrol(21:24),'Enable','off');
    set(data.handle.axes(2),'Color',data.cordefundo);
elseif isempty(canais.pos)
   set(data.handle.uicontrol(24),...
   'Enable'        ,'off',...
   'Value'         ,1);
   set(data.handle.uicontrol(21:22),'Enable','off');
   data.graf=1;
else
    set(data.handle.uicontrol(12),'Enable','off');
    set(data.handle.uicontrol(22),'Enable','off');
```

```
end
% Ajusta o programa para pegar os dados a cada 47ms;
% O tempo esta ajustado para 46 pois o matlab tem um delay de 1ms, neste
% caso;
% O programa so plotara de 10 em 10 ciclos, ou seja, de 470ms em 470ms;
set(data.ai,...
    'TriggerRepeat', 1,...
    'TriggerType', 'Manual',...
    'SamplesPerTrigger', taxa,...
    'TimerPeriod', 0.046,...
    'TimerFcn', @Repeticao,...
    'StopFcn', {'Daqiq', 'para', gcbf});
% Abre a comunicacao com a placa;
try
    start(data.ai)
catch
    if findstr('daqregister', lower(lasterr))
        aviso = findobj(findall(0), 'Tag', 'placa');
        if isempty(aviso)
            warndlg('A placa nao esta bem instalada. Cheque as conexoes.', 'Erro
de
Instalacao');
            set(aviso, 'Tag', 'placa');
        else
            figure(aviso(1));
        end
        return;
    elseif findstr('samples', lower(lastwarn))
        aviso = findobj(findall(0), 'Tag', 'taxa');
        if isempty(aviso)
            warndlg('A taxa de amostragem nao eh valida.', 'Erro de
Configuracao');
            set(aviso, 'Tag', 'taxa');
        else
            figure(aviso(1));
        end
        return;
    else
        error(lasterr);
        error(lastwarn);
    end
end
%pega as primeiras amostras;
if isempty(findstr('.txt',data.arquivo))
    data = Arqsaida(data);
end
trigger(data.ai);
% tic
d = getdata(data.ai, get(data.ai, 'SamplesPerTrigger'));
% faz i2 +q2;
dados(:,1)= d(:,1).^2 + d(:,2).^2;
% Verifica se o graf 2 sera usado e qual canal;
if ~isempty(canais.trig)
    if data.graf==1
        dados(:,2)=d(:,3);
    else
        dados(:,2)=d(:,4);
    end
end
% Ajusta a escala dos 2 graficos;
for i=1:size(dados,2)
    switch unidades(i)
        case 2
            dados(:,i) = 1000*dados(:,i);
        case 4
            dados(:,i) = 10*log10(1000*dados(:,i));
            dados(:,i) = dados(:,i) - max(dados(:,i));
    end
end

if ~isempty(data.handle.line)
    delete(data.handle.line);
    data.handle.line = [];
end
% % seta automaticamente o nivel dc do canal 2;
% div = str2num(get(data.handle.uicontrol(13), 'String'));
% yref = mean(dados(:,2));
```

```
% ysup = 5*div+yref;
% yinf = (-5*div)+yref;
% set(data.handle.axes(2),...
%     'YLim', [yinf ysup],...
%     'YTick', linspace(yinf, ysup, 11),...
%     'YTickLabel', {num2str(yinf,4), '','','','','',...
%     '','','','',num2str(ysup,4)});
% set(data.handle.uicontrol(16), 'String',num2str(yref));
% data.referencias(2)=yref;

% Seta o autoset automaticamente e desliga o manual;
data.autoset = 1;
set(data.handle.uicontrol(19),'Value',1);
set(data.handle.uicontrol(20),'Value',0);
set(data.handle.uicontrol(7), 'Enable', 'off');
set(data.handle.uicontrol(9:10), 'Enable', 'off');

yinf = min(dados(:,1));
ysup = max(dados(:,1));

% Se Nans sao retornados ou se ambos forem 0;
if (isnan(yinf) & isnan(ysup)) | (yinf ==0 & ysup ==0)
    yinf = -0.01;
    ysup = 0.01;
elseif isnan(yinf)
    yinf = ysup - (abs(ysup)/2);
elseif isnan(ysup)
    ysup = yinf + (abs(yinf)/2);
end

% Quando os limites sao iguais;
if yinf == ysup
    yinf = yinf - (abs(yinf)/2);
    ysup = ysup + (abs(ysup)/2);
end

set(data.handle.axes(1), 'YLim', [yinf ysup],...
    'YTick', linspace(yinf, ysup, 11),...
    'YTickLabel', {num2str(yinf,4), '','','', '','',...
    '','','','',num2str(ysup,4)});

set(data.handle.uicontrol(7), 'String', num2str((ysup-yinf)/10,3));
set(data.handle.uicontrol(10),'String',num2str((ysup+yinf)/2));
% Plota as primeiras amostras;
linha=[];

for i = 1:size(dados,2)
    linha(i) = line('Parent', data.handle.axes(i),...
        'Xdata', 1:length(dados),...
        'Ydata', dados(:,i),...
        'HandleVisibility', 'off');
end
drawnow;
%muda os parametros da estrutura;
data.cont=cont+1;
data.canais=canais;
% Muda o estado;
data.estado = 1;
%Altera a parte grafica;
set(data.handle.uicontrol(17), 'Value', 1);
set(data.handle.uicontrol(17), 'Enable', 'off');
set(data.handle.menu(5), 'Enable' ,'off');
set(data.handle.menu(6), 'Enable' ,'on');
%salva os dados no arquivo desejado;
save(data.arquivo,'d','-ascii','-append');
% Armazena as linhas dos graficos;
data.handle.line = linha;

% ********************************************************************
% Para a varredura;
function data = Para(data)

if (isvalid(data.ai) & data.estado==1)
    stop(data.ai);
    % Permite ao usuario mudar os canais,taxa de amostragem e arquivo de
    % saida;
    for i=0:1
```

```
        set(data.handle.uicontrol(2+2*i), 'Enable', 'on');
    end
    for i=0:length(data.canais.trig)
        set(data.handle.uicontrol(6+6*i), 'Enable', 'on');
    end
    if ~isempty(data.canais.pos)
        set(data.handle.uicontrol(22), 'Enable', 'on');
    end
    % Retira o autoset e retorna os botoes para o estado inicial;
    data.estado = 0;
    set(data.handle.uicontrol(19), 'Value', 0);
    set(data.handle.uicontrol(20), 'Value', 1);
    data = Manual(data);
    set(data.handle.uicontrol(17), 'Value', 0);
    set(data.handle.uicontrol(17), 'Enable', 'on');
    set(data.handle.menu(5), 'Enable' ,'on');
    set(data.handle.menu(6), 'Enable' ,'off');
end
% ****************************************************************
% Plota os dados continuamente;
function data = Repeticao(obj,event)

% Pega a estrutura existente;
data = obj.UserData;
% Inicializa Variaveis;
cont=data.cont;
canais = data.canais;
taxa=floor(data.taxa*.047);
unidades=data.unidades;
if data.estado == 1
    % Atualiza Linhas;
    linha = data.handle.line;
    % pega amostras;
    % toc
    d = peekdata(obj, obj.SamplesPerTrigger);
    % tic
    if cont==10
        dados(:,1)= d(:,1).^2 + d(:,2).^2;
        if ~isempty(canais.trig)
            if data.graf==1
                dados(:,2)=d(:,3);
            else
                dados(:,2)=d(:,4);
            end
        end
        for i=1:size(dados,2)
            switch unidades(i)
                case 2
                    dados(:,i) = 1000*dados(:,i);
                case 4
                    dados(:,i) = 10*log10(1000*dados(:,i));
                    dados(:,i) = dados(:,i) - max(dados(:,i));
            end
        end

        % Atualiza o grafico;
        for i=1:size(dados,2)
            set(linha(i), 'Parent', data.handle.axes(i),...
                'XData',    1:length(dados),...
                'YData',    dados(:,i));
        end

%         % seta automaticamente o nivel dc do canal 2;
%         div = str2num(get(data.handle.uicontrol(13), 'String'));
%         yref = mean(dados(:,2));
%         ysup = 5*div+yref;
%         yinf = (-5*div)+yref;
%         set(data.handle.axes(2),...
%             'YLim', [yinf ysup],...
%             'YTick', linspace(yinf, ysup, 11),...
%             'YTickLabel', {num2str(yinf,4), '','','','','',...
%             '','','','',num2str(ysup,4)});
%         set(data.handle.uicontrol(16), 'String',num2str(yref));
%         data.referencias(2)=yref;
%
        if data.autoset
            yinf = min(dados(:,1));
```

```
            ysup = max(dados(:,1));

            % Se Nans sao retornados ou se ambos forem 0;
            if (isnan(yinf) & isnan(ysup)) | (yinf ==0 & ysup ==0)
                yinf = -0.01;
                ysup = 0.01;
            elseif isnan(yinf)
                yinf = ysup - (abs(ysup)/2);
            elseif isnan(ysup)
                ysup = yinf + (abs(yinf)/2);
            end

            % Quando os limites sao iguais;
            if yinf == ysup
                yinf = yinf - (abs(yinf)/2);
                ysup = ysup + (abs(ysup)/2);
            end

            set(data.handle.axes(1), 'YLim', [yinf ysup],...
                'YTick', linspace(yinf, ysup, 11),...
                'YTickLabel', {num2str(yinf,4), '','','', '','',...
                '','','','',num2str(ysup,4)});

            set(data.handle.uicontrol(7), 'String', num2str((ysup-yinf)/10,3));
            set(data.handle.uicontrol(10),'String',num2str((ysup+yinf)/2));
        end
        data.cont=1;
    else
        data.cont=cont+1;
    end
    % Armazena as estruturas no objeto e na tela;
    set(obj,'UserData',data);
    save(data.arquivo,'d','-ascii','-append');
end
drawnow;

% ********************************************************************
function data = Unidade(data,numero)
% Se a aquisicao estiver ativada;
if data.estado == 1
    set(data.handle.uicontrol(8+(numero-1)*6), 'value',data.unidades(numero));
    aviso = findobj(findall(0), 'Tag', 'unidade');
    if isempty(aviso)
        aviso = warndlg('A aquisicao nao pode estar ativada.','Erro de
Configuracao!');
        set(aviso, 'Tag', 'unidade');
    else
        figure(aviso(1));
    end
    % caso contrario;
else
    escdiv = get(data.handle.uicontrol(8+(numero-1)*6), 'value');
    data.unidades(numero)=escdiv;
end
% ********************************************************************
% Permite ao usuario selecionar o nivel de referencia;
function data = Referencia(data,numero)

val=get(data.handle.uicontrol(9+(numero-1)*6),'Value');
data.check(numero)=val;
% Se o checkbox estiver selecionado;
if val
    set(data.handle.uicontrol(10+(numero-1)*6),...
        'Enable', 'on'              ,...
        'BackGroundColor', [1 1 1]  ,...
        'string', num2str(data.referencias(numero)));
    data = Escala(data,numero);
    % Se o checkbox nao estiver selecionado;
else
    set(data.handle.uicontrol(10+(numero-1)*6),...
        'Enable', 'off'             ,...
        'String',  ' '              ,...
        'BackGroundColor', data.cordefundo);
    data = Escala(data,numero);
end
% ********************************************************************
% Seta a escala por divisao determinada pelo usuario;
```

```
function data = Escala(data,numero)

div = str2num(get(data.handle.uicontrol(7+(numero-1)*6), 'String'));
% A escala nao pode ser negativa;
if div <= 0
    aviso = findobj(findall(0), 'Tag', 'escala');
    if isempty(aviso)
        aviso = warndlg('A escala nao pode ser negativa.','Valor invalido!!!');
        set(aviso, 'Tag', 'escala');
    else
        figure(aviso(1));
    end
    set(data.handle.uicontrol(7+(numero-
1)*6),'string',num2str(data.escalas(numero)));
    return;
end
% Verifica o nivel de referencia;
if data.check(numero)
    yref = str2num(get(data.handle.uicontrol(10+(numero-1)*6), 'String'));
    data.referencias(numero)=yref;
else
    yref = 0;
end
ysup = 5*div+yref;
yinf = (-5*div)+yref;
set(data.handle.axes(numero),...
    'YLim', [yinf ysup],...
    'YTick', linspace(yinf, ysup, 11),...
    'YTickLabel', {num2str(yinf,4), '','','','','',...
    '','','','',num2str(ysup,4)});
data.escalas(numero)=div;
% ********************************************************************
% Muda a taxa de amostrage;
function data = Taxa(data)

taxa = str2num(get(data.handle.uicontrol(4), 'String'));
try
    set(data.ai, 'SampleRate', taxa);
catch
    % Caso a taxa seja invalida;
    if findstr('daqdevice', lower(lasterr))
        aviso = findobj(findall(0), 'Tag', 'taxa');
        set(data.handle.uicontrol(4),'String',num2str(data.taxa));
        if isempty(aviso)
            warndlg('A taxa de amostragem nao eh valida.', 'Erro de
Configuracao');
            set(aviso, 'Tag', 'taxa');
        else
            figure(aviso(1));
        end
        return;
    else
        error(lasterr);
    end
end
data.taxa=taxa;

% ********************************************************************
% Inicializa a janela do programa;
function data = Janela(data)
clc;
daqreset;
clear all;
% Inicializa variavel;
cordefundo = get(0,'DefaultUIControlBackgroundColor');

% Posiciona a janela no meio da tela;
unidadepadrao='normalized';
set(0,'Units',unidadepadrao);
tamanho=get(0,'ScreenSize');

x0=.1;
largura=.8;

y0=.1;
altura=.8;
```

```
posjan=[x0 y0 largura altura];

% informacao geral para todos objetos graficos;
geninfo.HandleVisibility='off';
geninfo.Interruptible='off';
geninfo.BusyAction='queue';

% Cria a janela em si;
fig=figure(geninfo,...
    'Color'             ,cordefundo                 ,...
    'DeleteFcn'         ,'Daqiq(''fecha'', gcbf)'    ,...
    'DoubleBuffer'      ,'on'                       ,...
    'IntegerHandle'     ,'off'                      ,...
    'MenuBar'           ,'none'                     ,...
    'Name'              ,'Perfil de Potencia'       ,...
    'Tag'               ,'Perfil de Potencia'       ,...
    'NumberTitle'       ,'off'                      ,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,posjan                     ,...
    'Resize'            ,'off'                      ,...
    'UserData'          ,[]                         ,...
    'Colormap'          ,[]                         ,...
    'Pointer'           ,'arrow'                    ,...
    'Visible'           ,'off');

geninfo.Parent = fig;

% informacao para todos uicontrol (User Interface Control);
uiinfo=geninfo;
uiinfo.BackGroundColor=cordefundo;
uiinfo.ForeGroundColor=[0 0 0];

% Cria os graficos;
posgraf1 = [x0/2 .45+y0/2 .65 .4];
posgraf2 = [x0/2 y0/2 .65 .4];

graf1 = axes(geninfo,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,posgraf1                   ,...
    'Box'               ,'On'                       ,...
    'TickLength'        ,[0 0]                      ,...
    'DrawMode'          ,'fast'                     ,...
    'Color'             ,[1 1 1]                    ,...
    'ColorOrder'        ,[0 0 1]                    ,...
    'XLim'              ,[0 512]                    ,...
    'XTick'             ,linspace(0,512,9)          ,...
    'XTickLabel'        ,{'','','','','','','','',''},...
    'YLim'              ,[-.5 .5]                   ,...
    'YTick'             ,linspace(-.5,.5,11)        ,...
    'YTickLabel'        ,{'-.5','','','','',''      ,...
    '','','','','.5'}           ,...
    'XGrid'             ,'on'                       ,...
    'YGrid'             ,'on'                       ,...
    'GridLineStyle'     ,':');

graf2 = axes(geninfo,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,posgraf2                   ,...
    'Box'               ,'On'                       ,...
    'TickLength'        ,[0 0]                      ,...
    'DrawMode'          ,'fast'                     ,...
    'Color'             ,[1 1 1]                    ,...
    'ColorOrder'        ,[1 0 0]                    ,...
    'XLim'              ,[0 512]                    ,...
    'XTick'             ,linspace(0,512,9)          ,...
    'XTickLabel'        ,{'','','','','','','','',''},...
    'YLim'              ,[-.5 .5]                   ,...
    'YTick'             ,linspace(-.5,.5,11)        ,...
    'YTickLabel'        ,{'-.5','','','','',''      ,...
    '','','','','.5'}           ,...
    'XGrid'             ,'on'                       ,...
    'YGrid'             ,'on'                       ,...
    'GridLineStyle'     ,':');

% Cria molduras;
posmoldura = {[posgraf1(1)+posgraf1(3)+0.05 posgraf1(2) .2 .4],...
    [posgraf2(1)+posgraf2(3)+0.05 posgraf2(2) .2 .4]};
```

```
for i = 1:2
    molduras(i) = uicontrol(uiinfo,...
        'Style'          , 'frame'          ,...
        'Units'          , unidadepadrao    ,...
        'Position'       , posmoldura{i});
end

% Inicializa a placa de aquisicao;
lasterr=[];
lastwarn=[];
try
    daqregister('nidaq');
catch (lasterr);
end
if isempty(findstr('daqregister',lower(lasterr)))
    sa = analoginput('nidaq','1');
    set(sa, 'InputType', 'SingleEnded');
    %cria canais para preencher os ListBox[1,2];
    out = daqhwinfo(sa);
    nomes=makenames('Canal ',out.SingleEndedIDs);
else
    warndlg('A placa de aquisicao da NI nao esta instalada. O programa sera
inicializado com a Placa de Som','Aviso de Inicializacao');
    daqregister('winsound');
    sa = analoginput('winsound');
    out = daqhwinfo(sa);
    nomes=makenames('Canal ',out.SingleEndedIDs);
    pause(1);
end
% Cria vetor de escala;
esc1=['Escala    ';'mW/Div    ';'Watts/Div ';'dBm/div   '];
esc2=['Escala   ';'mV/div   ';'Volts/div'];
escala1=cellstr(esc1);
escala2=cellstr(esc2);
% Descricao de data.handle.uicontrol;
% 1: text: Arquivo de saida;
% 2: edit: nome do arquivo;
% 3: text: taxa de amostragem;
% 4: edit: taxa de amostragem;
% 5: text: Canal de Aquisicao;
% 6: listbox: canal de aquisicao;
% 7: edit: escala por divisao;
% 8: popup: unidade do canal de aquisicao;
% 9: checkbox: nivel de referencia;
% 10: edit: nivel de referencia;
% 11: text: Canal de trigger;
% 12: listbox: canal de trigger;
% 13: edit: escala por divisao;
% 14: popup: unidade do canal de aquisicao;
% 15: Checkbox: nivel de referencia;
% 16: edit: nivel de referencia;
% 17: Botao Inicia;
% 18: Botao Para;
% 19: radiobutton: autoset;
% 20: radiobutton: config. manual;
% 21: text: Canal de posicao;
% 22: popup: canal de posicao;
% 23: text: grafico exibido;
% 24: popup: grafico exibido

% Cria parametros;

posarq1=[.15 .93 .13 .03];
par(1) = uicontrol(uiinfo,...
    'Style'              ,'text'                    ,...
    'Units'              ,unidadepadrao             ,...
    'Position'           ,posarq1                   ,...
    'ForegroundColor'    ,[0 0 0]                   ,...
    'String'             ,'Arquivo de Saida:');
posarq2=[.27 .93 .2 .04];
par(2) = uicontrol(uiinfo,...
    'Style'              ,'edit'                    ,...
    'Units'              ,unidadepadrao             ,...
    'Position'           ,posarq2                   ,...
    'String'             ,'teste.mat'               ,...
    'BackGroundColor'    ,[1 1 1]                   ,...
```

```
    'Callback'          ,'Daqiq(''arqsaida'', gcbf);');

posamos1=[.55 .93 .13 .03];
par(3) = uicontrol(uiinfo,...
    'Style'             ,'text'                     ,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,posamos1                   ,...
    'ForegroundColor'   ,[0 0 0]                    ,...
    'String'            ,'Taxa de Amostragem (Hz):');
posamos2=[.7 .93 .2 .04];
par(4) = uicontrol(uiinfo,...
    'Style'             ,'edit'                     ,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,posamos2                   ,...
    'String'            ,'1000'                     ,...
    'BackGroundColor'   ,[1 1 1]                    ,...
    'Callback'          ,'Daqiq(''taxa'', gcbf);');

pm=cell2mat(posmoldura);
poscan1=[pm(1)+.01 pm(2)+pm(4)-.04 .18 .03];
par(5) = uicontrol(uiinfo,...
    'Style'             ,'text'                     ,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,poscan1                    ,...
    'ForegroundColor'   ,[0 0 0]                    ,...
    'String'            ,'Canais I e Q:');

poslist1=[poscan1(1) poscan1(2)-pm(4)/2 .18 pm(4)/2];
poslist1_aux=[poscan1(1) poscan1(2)-pm(4)/2.5 .18 pm(4)/2.5];
par(6) = uicontrol(uiinfo,...
    'Style'             ,'listbox'                  ,...
    'Max'               ,3                          ,...
    'Min'               ,1                          ,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,poslist1_aux               ,...
    'BackGroundColor'   ,[1 1 1]                    ,...
    'String'            ,nomes                      ,...
    {'Value'}           ,{[3 4]}                    ,...
    'ListboxTop'        ,1);

posautoset=[poscan1(1) poscan1(2)-pm(4)/2 .18 .034];
par(19) = uicontrol(uiinfo,...
    'Style'             ,'radiobutton'              ,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,posautoset                 ,...
    'Value'             ,0                          ,...
    'String'            ,'Autoset'                  ,...
    'Callback'          ,'Daqiq(''autoset'',gcbf);');

posmanual=[poscan1(1) poscan1(2)-pm(4)/2-.03 .18 .034];
par(20) = uicontrol(uiinfo,...
    'Style'             ,'radiobutton'              ,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,posmanual                  ,...
    'Value'             ,1                          ,...
    'String'            ,'Config. Manual'           ,...
    'Callback'          ,'Daqiq(''manual'',gcbf);');

posescala1=[poslist1(1) poslist1(2)-.07 .08 .034];
par(7)= uicontrol(uiinfo,...
    'Style'             ,'edit'                     ,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,posescala1                 ,...
    'String'            ,.1                         ,...
    'BackGroundColor'   ,[1 1 1]                    ,...
    'Callback'          ,'Daqiq(''escala'',''1'', gcbf);');

pospop1=[poslist1(1)+.08 poslist1(2)-.066 .1 .03];
par(8)= uicontrol(uiinfo,...
    'Style'             ,'popup'                    ,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,pospop1                    ,...
    'String'            ,escala1                    ,...
    'Value'             ,4                          ,...
    'BackGroundColor'   ,[1 1 1]                    ,...
    'Callback'          ,'Daqiq(''unidade'',''1'', gcbf);');
```

```
poscheck1=[posescala1(1) posescala1(2)-.034 .18 .03];
par(9) = uicontrol(uiinfo,...
    'Style'                ,'checkbox'            ,...
    'Value'                ,0                     ,...
    'Units'                ,unidadepadrao         ,...
    'Position'             ,poscheck1             ,...
    'String'               ,'Nivel de Referencia' ,...
    'HorizontalAlignment'  ,'left'                ,...
    'Callback'             ,'Daqiq(''referencia'',''1'',gcbf);');
posref1=poscheck1 +[0 -.04 0 +.01];
par(10) = uicontrol(uiinfo,...
    'Style'               ,'edit'                 ,...
    'Enable'              ,'off'                  ,...
    'Units'               ,unidadepadrao          ,...
    'Position'            ,posref1                ,...
    'String'              ,' '                    ,...
    'HorizontalAlignment' ,'left'                 ,...
    'Callback'            ,'Daqiq(''escala'',''1'', gcbf);');

poscan2=[pm(5)+.01 pm(6)+pm(8)-.04 .18 .03];
par(11) = uicontrol(uiinfo,...
    'Style'              ,'text'                   ,...
    'Units'              ,unidadepadrao            ,...
    'Position'           ,poscan2                  ,...
    'ForegroundColor'    ,[0 0 0]                  ,...
    'String'             ,'Canal de Trigger:');

poslist2=[poscan2(1) poscan2(2)-.03 .18 .03];
par(12) = uicontrol(uiinfo,...
    'Style'             ,'popup  '                 ,...
    'Units'             ,unidadepadrao             ,...
    'Position'          ,poslist2                  ,...
    'BackGroundColor'   ,[1 1 1]                   ,...
    'Value'             ,5                         ,...
    'String'            ,nomes);

posdist1=[poscan2(1) poslist2(2)-.04 .18 .03];
par(21) = uicontrol(uiinfo,...
    'Style'              ,'text'                   ,...
    'Units'              ,unidadepadrao            ,...
    'Position'           ,posdist1                 ,...
    'ForegroundColor'    ,[0 0 0]                  ,...
    'String'             ,'Canal de Posicao:');

posdist2=[poscan2(1) posdist1(2)-.03 .18 .03];
par(22) = uicontrol(uiinfo,...
    'Style'             ,'popup  '                 ,...
    'Units'             ,unidadepadrao             ,...
    'Position'          ,posdist2                  ,...
    'BackGroundColor'   ,[1 1 1]                   ,...
    'String'            ,nomes);

posgr1=[poscan2(1) posdist2(2)-.05 .18 .03];
par(23) = uicontrol(uiinfo,...
    'Style'              ,'text'                   ,...
    'Units'              ,unidadepadrao            ,...
    'Position'           ,posgr1                  ,...
    'ForegroundColor'    ,[0 0 0]                   ,...
    'String'             ,'Grafico exibido:');

posgr2=[poscan2(1) posgr1(2)-.03 .18 .03];
par(24) = uicontrol(uiinfo,...
    'Style'             ,'popup  '                 ,...
    'Units'             ,unidadepadrao             ,...
    'Position'          ,posgr2                    ,...
    'BackGroundColor'   ,[1 1 1]                   ,...
    'String'            ,{'Trigger';'Posicao'}     ,...
    'Callback'          ,'Daqiq(''graf'',gcbf);');

posescala2=[poslist2(1) poslist2(2)-pm(8)/2-.004 .08 .033];
par(13)= uicontrol(uiinfo,...
    'Style'              ,'edit'                   ,...
    'Units'              ,unidadepadrao            ,...
    'Position'           ,posescala2               ,...
    'String'             ,.1                       ,...
    'BackGroundColor'    ,[1 1 1]                  ,...
    'Callback'           ,'Daqiq(''escala'',''2'', gcbf);');
```

```
pospop2=[posescala2(1)+.08 poslist2(2)-pm(8)/2-.002 .1 .031];
par(14)= uicontrol(uiinfo,...
    'Style'             ,'popup'                    ,...
    'Units'             ,unidadepadrao              ,...
    'Position'          ,pospop2                    ,...
    'String'            ,escala2                    ,...
    'BackGroundColor'   ,[1 1 1]                    ,...
    'Callback'          ,'Daqiq(''unidade'',''2'', gcbf);');

poscheck2=[posescala2(1) posescala2(2)-.05 .18 .03];
par(15) = uicontrol(uiinfo,...
    'Style'                 ,'checkbox'             ,...
    'Value'                 ,0                      ,...
    'Units'                 ,unidadepadrao          ,...
    'Position'              ,poscheck2              ,...
    'String'                ,'Nivel de Referencia'  ,...
    'HorizontalAlignment'   ,'left'                 ,...
    'Callback'              ,'Daqiq(''referencia'',''2'',gcbf);');
posref2=poscheck2 +[0 -.04 0 +.01];
par(16) = uicontrol(uiinfo,...
    'Style'               ,'edit'                   ,...
    'Enable'              ,'off'                    ,...
    'Units'               ,unidadepadrao            ,...
    'Position'            ,posref2                  ,...
    'String'              ,' '                      ,...
    'HorizontalAlignment' ,'left'                   ,...
    'Callback'            ,'Daqiq(''escala'',''2'', gcbf);');

% Cria o quadrado do botao Parada;
quadrado = zeros(18,18,3);
quadrado(:, :, 1) = 1;

% Cria o triangulo do botao Inicia;
triangulo = zeros(18,18,3);
for c=1:3
    for i=1:9
        triangulo(i,1:2*i,2)=1;
        triangulo(19-i,1:2*i,2)=1;
        triangulo(i,2*i:18,c)=cordefundo(c);
        triangulo(19-i,2*i:18,c)=cordefundo(c);
    end
end

% Cria o Botao inicia;
par(17) = uicontrol(uiinfo,...
    'Style'               ,'pushbutton'         ,...
    'Value'               ,0                     ,...
    'Units'               ,unidadepadrao              ,...
    'Position'            ,[.01 .93 .04 .05]             ,...
    'CData'               ,triangulo       ,...
    'TooltipString'       ,'Inicia Aquisicao'    ,...
    'Callback'            ,'Daqiq(''inicia'',gcbf);');
par(18) = uicontrol(uiinfo,...
    'Style'               ,'pushbutton'         ,...
    'Value'               ,0                     ,...
    'Units'               ,unidadepadrao              ,...
    'Position'            ,[.065 .93 .04 .05]             ,...
    'CData'               ,quadrado       ,...
    'TooltipString'       ,'Para Aquisicao'    ,...
    'Callback'            ,'Daqiq(''para'',gcbf);');

% Descricao dos menus;
% 1: arquivo >>;
% 2: Reinicializa;
% 3: Fechar;
% 4: Acoes >>;
% 5: Inicia;
% 6: Para;
% 7: Ajuda >>;
% 8: Ajuda do Analisador;
% 9: Ajuda do Matlab;



% Cria o menu arquivo;
menu(1) = uimenu('Label', 'Arquivo',...
```

```
    'Parent'            ,fig);
menu(2) = uimenu(menu(1),...
    'Label'             ,'Reinicializar'      ,...
    'Callback'          ,'Daqiq(''reinicializa'', gcbf);');
menu(3) = uimenu(menu(1),...
    'Label'             ,'Fechar Analisador'      ,...
    'Callback'          ,'Daqiq(''fecha'', gcbf);');

% Cria o menu acoes;
menu(4) = uimenu('Label', 'Acoes',...
    'Parent'            ,fig);
menu(5) = uimenu(menu(4),...
    'Label'             ,'Inicia'                 ,...
    'Callback'          ,'Daqiq(''inicia'',gcbf);');
menu(6) = uimenu(menu(4),...
    'Label'             ,'Para'                   ,...
    'Enable'            ,'off'                    ,...
    'Callback'          ,'Daqiq(''para'',gcbf);');

% Cria o menu ajuda;
menu(7) = uimenu('Label', 'Ajuda',...
    'Parent'            ,fig);
menu(8) = uimenu(menu(7),...
    'Label'             ,'Ajuda do DaqIQ'     ,...
    'Callback'          ,'Daqiq(''ajuda'', gcbf);');
menu(9) = uimenu(menu(7),...
    'Label'             ,'Ajuda do MatLab para Aquisicao de dados (em Ingles)'
,...
    'Callback'          ,'Daqiq(''daqhelp'', gcbf);');

% Cria a estrutura de dados:

% Tela do programa;
data.handle.figure = fig;
data.handle.uicontrol = par;
data.handle.menu = menu;
% variavel global;
data.cordefundo = cordefundo;
% Graficos;
data.handle.axes = [graf1 graf2];
% variaveis relativas a aquisicao de dados;
data.ai = sa;
% Configuracoes iniciais;
data.estado = 0;
data.handle.line = [];
data.autoset=0;
data.canais.qi=[3 4];
data.canais.pos=1;
data.canais.trig=5;
data.escalas(1:2)=[.1 .1];
data.unidades(1:2)=[4 1];
data.referencias=[0 0];
data.check=[0 0];
data.taxa=1000;
data.arquivo='teste.mat';
if exist('teste.txt')
    delete teste.txt;
end
data.cont=0;
data.graf = 1;
% Armazena a estrutura de dados e abre a tela do programa;
set(fig,'Visible','on','UserData',data);
% Armazena a estrutura de dados no objeto;
set(data.ai, 'UserData', data);
```

2. Aquisição *Off Line*

Neste caso os arquivos de dados gravados *on line* foram processados no laboratório através do programa AQUISIÇÃO, mostrado a seguir, de forma que os parâmetros desejados pudessem ser determinados. Mais adiante o programa SIMULAÇÃO é mostrado, onde tem-se todo o

processamento da seqüência pseudo-aleatória, permitindo que suas amostras filtradas, pré-distorcidas e equalizadas possam ser gravadas nas EPROM's e faz-se uma simulação do sinal à saída do transmissor.

Programa AQUISIÇÃO

```
%Aquisição1
%--------------------------------------------------------------------------------
clear; clc; %Limpa variáveis e a tela de comando
disp(' ');disp(' ');disp(' ');disp(' ');disp(' ');
disp(' ------------------------------*------------*--------------------------------
');
disp('         Programa para tratamento das medidas coletadas');
disp(' ');
disp('   O objetivo deste programa é tratar as amostras coletadas, repondo
amostras');
disp('  perdidas e selecionando trechos dos perfis onde se dispõe de pulsos de
tensão
disp(' da roda válidos, referentes a movimento do carro. Os dados são compostos de
medidas');
disp('  de tensão I( em fase), Q( em quadratura), T( Trigger ) e D( Pulsos do
sensor'
disp('  de  posição na roda ). ');
disp(' ');
disp(' ------------------------------*------------*--------------------------------
');
ent=input('Pressione enter para iniciar o programa.');


% ------------------ Carregando os dados medidos para o programa -----------------

% Os dados coletados estão armazenados da seguinte forma:
% o arquivo armazenado é uma matriz 4 por n, onde a primeira coluna contém as
tensões de Trigger
% a segunda coluna contém as medidas Q, a terceira as medidas de I e a quarta as
medidas da roda D.
% O nome do arquivo será o nome do local onde as medidas foram tomadas.


addpath c:\teste;  %diretorio onde estao os arquivos
clc;
disp(' ');
nome_do_arquivo = input(' Digite o nome das medidas que serão tratadas: ','s');
load(nome_do_arquivo); %lê arquivo com os dados do diretório c:\teste
disp(' ');
disp('  Carregando o arquivo... ');
disp(' ');
ent=input('Pressione enter para continuar.');
EOF=size(T,2); % É o tamanho do arquivo - número de amostras total;
RED=input('Fator de Redução no "replay" da fita do gravador','s');
NoAP=RED.*1000;

%----------Decidindo entre 0 e 1 para os níveis de tensão do Trigger--------------
clc;
disp(' ');
disp('  Tratando as amostras do Trigger... ');

%      Foi definido um Limiar de 2.5, de tal forma que tensões acima do Limiar
%   representam 1 e abaixo do Limiar representam 0. Como é sabido o número de 1´s
%   ou 0´s que tem que aparecer (2 x RED), esse algoritmo checa se os 1´s ou 0´s
formam
%   uma sequência completa.  Após essa checagem ele corrige erros, incluindo no
Limiar
%   uma margem de 0.5.

Limiar=2.5; %Limiar de decisão para a Tensão do Trigger
Margem=0.5; %Margem usada para decidir se uma amostra está errada ou não

if T(1)>=Limiar
   Status=1;
   t(1)=1;
else               %Define Status em função da primeira amostra
   Status=0;
```

```
   t(1)=0;
end
cont=1;
Trocou=0;
for n=2:EOF
   if T(n)>=Limiar
      if Status==0
         Trocou=1;   %determina se Trocou de 1 para 0
         Status=1;
      else
         Trocou=0;
      end
   else
      if Status==1
         Trocou=1;   %determina se Trocou de 0 para 1
         Status=0;
      else
         Trocou=0;
      end
   end

   if Trocou==1
      if cont==2*RED      %Trocou no momento certo
         t(n)=Status;
         cont=1;
      else             %Trocou fora de hora...
         if T(n)<=Limiar      %de 1 para 0...
            if T(n)<=Limiar-Margem    %mas Trocou certo
               t(n)=Status;
               cont=1;
            else
               Status=not(Status);           %Trocou errado
               t(n)=Status;
               cont=cont+1;
            end
         else                      %de 0 para 1...
               if T(n)>=Limiar + Margem  %mas Trocou certo
                  t(n)=Status;
                  cont=1;
               else                              %e Trocou errado
                  Status=not(Status);
                  t(n)=Status;
                  cont=cont+1;
               end
         end
      end
   else       %não Trocou...
      if cont>=2*RED              %mas deveria ter trocado
         cont=1;
         Status=not(Status);
         t(n)=Status;
      else         %OK
         t(n)=Status;
         cont=cont+1;
      end
   end
end


%--------------------------- Completando amostras perdidas -----------------------

%  Existem três situações possíveis para grupos de amostras: grupo completo, uma
% amostra perdida e mais de uma amostra perdida, sob a condição de que o número
% máximo de amostras perdidas por bit de trigger seja menor do que a metade do
número % total de amostras. Assim, por exemplo, para RED = 4, o grupo completo
possui oito % % amostras e o máximo que pode faltar é três amostras.
%  Para corrigir uma falta, é colocado um valor médio entre os valores do meio da
% sequência; para mais de uma amostra perdida, é inserido um valor médio entre o
1º e
% o 2º, entre o 2º e o 3º... até que se tenha inserido o número de amostras que %
% %
% faltou.

T=t;  %leva o vetor gerado no algoritmo anterior de volta para T
%n é o ponteiro que cuida de T enquanto j é o ponteiro que cuidará dos novos
campos
disp(' ');
```

```
disp('    A seguir serão exibidas as primeiras amostras de Trigger para que se
possa ');
disp(' informar ao programa onde está o começo das amostras que devem ser
completadas.');
disp('  ');
clear Aux;
for n=1:50;%Tomando as 50 primeiras amostras dos pulsos de trigger
   Aux(n)=T(n);
end
disp(sprintf('%g ',Aux));
clear Aux;
disp(' ');
entnum=input('  A partir de qual amostra o trigger deve ser completado? Digite um
número válido.');
n=entnum;
%n é o ponteiro que cuida de T, enquanto j é o ponteiro que cuidará dos novos
campos
clc;
disp(' ');
disp('  Completando as amostras de Trigger... ');
disp(' ');

for l=1:n-1   %Valores antes do início da correção
   t(l)=T(l);
   q(l)=Q(l);
   i(l)=I(l);
   d(l)=D(l);
end

j=n;
while n<EOF;
   Amostras=0;
   l=n;
   Status=T(l);
   while T(l)==Status       % conta as amostras de uma sequência
      % disp(sprintf('l = %g',l));
      Amostras=Amostras+1;
      l=l+1;
      if l==EOF+1 break;
      end
   end
   NAP=2*RED-Amostras;%NAP - Número de Amostras Perdidas
   k=0;
   if NAP==0                  %Sequência completa
      for cont=0:2*RED-1
         t(j+cont)=T(n+cont);
         q(j+cont)=Q(n+cont);
         i(j+cont)=I(n+cont);
         d(j+cont)=D(n+cont);
      end
      n=n+2*RED;
   elseif NAP==1              %Uma amostra perdida - Insere média no meio
         for cont=0:2*RED-1
            t(j+cont)=Status;
            if cont==RED
               q(j+cont)=(Q(n+(cont-1))+Q(n+cont))/2;
               i(j+cont)=(I(n+(cont-1))+I(n+cont))/2;
               d(j+cont)=0;
               n=n-1;
            else
               q(j+cont)=Q(n+cont);
               i(j+cont)=I(n+cont);
               d(j+cont)=D(n+cont);
            end
         end
         n=n+2*RED;
      elseif NAP<RED/2+1  %Mais de uma amostra perdida - Médias entre os primeiros
termos
      l=0;
      while l<=NAP*2-1
         t(j+l)=Status;
         t(j+l+1)=Status;
         q(j+l)=Q(n);
         i(j+l)=I(n);
         d(j+l)=D(n);
         q(j+l+1)=(Q(n)+Q(n+1))/2;
         i(j+l+1)=(I(n)+Q(n+1))/2;
```

```
            d(j+l+1)=0;
            l=l+2;
            n=n+1;
         end
         for l=NAP*2:2*RED-1
            t(j+l)=Status;
            q(j+l)=Q(n);
            i(j+l)=I(n);
            d(j+l)=D(n);
            n=n+1;
         end
      else
         if n+RED>=EOF
            disp(sprintf('  A última amostra não está completa e por isso não foi
corrigida'));
            for n1=0:NAP-1
               t(j+n1)=Status;
               q(j+n1)=Q(n);
               i(j+n1)=I(n);
               d(j+n1)=D(n);
               n=n+1;
            end
         else
            disp(sprintf('  Amostra irrecuperável.  Ela perdeu mais bits do que
RED/2'));
            disp(sprintf('  O MATLAB será fechado!'));
            ent=input('  Pressione enter para encerrar o programa.');
            exit;
         end
      end
      j=j+2*RED;
   end
   disp(' ');
   ent= input('  Amostras tratadas. Pressione enter para continuar ');
   EOF=size(t,2);% Atualiza EOF devido ao aumento do nº de amostras ao se completar
   as faltantes
   T=t; %Devolve os valores corrigidos e completos para suas variáveis de origem
   Q=q;
   I=i;
   D=d;

   %----------------------Identificando o começo do primeiro perfil-----------------
   clc;
   disp(' Será plotado um gráfico com as primeiras 10000 amostras, para que se
   determine em');
   disp(' qual delas se inicía o primeiro perfil válido, relacionado ao movimento do
   carro.');
   disp(' ');
   disp('  Pressione enter para ver o gráfico.');
   P=I.^2 + Q.^2;
   for n=1:EOF
      Aux(n)=P(n);
   end
   plot(Aux);zoom on; grid on;
   disp(' ');
   inicio=input('Qual o número da amostra que representa o início do primeiro perfil
   válido?');
   disp(' ');
   if inicio~=1
      for n=1:EOF-inicio+1
         T(n)=T(inicio+n-1);
         I(n)=I(inicio+n-1);
         Q(n)=Q(inicio+n-1);
         D(n)=D(inicio+n-1);
      end
      EOF=EOF-inicio;
   end

   m=0;  % Retirando o excesso do final
   while m<=EOF
      m=m+NoAP;
   end

   clear EOF

   m=m-NoAP;
   EOF=m;
```

```
clear t d i q;
for n=1:EOF
    t(n)=T(n);
    i(n)=I(n);
    d(n)=D(n);
    q(n)=Q(n);
end

clear T D I Q P;
/
T=t;
D=d;
I=i;
Q=q;

clear t d i q;

disp('  O perfil plotado foi deslocado à esquerda para o inicio indicado.');
disp('  Também foram retiradas as amostras do final que não completavam um
perfil');
P=I.^2 + Q.^2;
plot(P);zoom on; grid on;
inicio=input(' Pressione enter  para continuar.');

%----- Acertando os bits do vetor com os bits relativos à distância percorrida --
%  As sequências de 1´s serão representadas apenas por seus 1´s iniciais e depois
esses 1's
% serão substituídos por suas respectivas ordens de aparição. Por exemplo, o
quinto 1 será
%5. Dessa forma todos eles estarão referenciados ao primeiro 1, conforme
necessário para se
% calcular a distância percorrida pelo móvel para cada perfil.
% Um exemplo:
% 0 0 1 1 1 0 0 0 1 0 0 1 0 0 0 0 1 0 0 1 1 1 0 0 0 0 1 1 1 1 0
% 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 0 0 0
% 0 0 1 0 0 0 0 0 2 0 0 3 0 0 0 0 4 0 0 5 0 0 0 0 0 0 6 0 0 0 0
% 0 0 1 1 1 1 1 1 2 2 2 3 3 3 3 3 4 4 4 5 5 5 5 5 5 5 6 6 6 6 6

clc;
disp(' ');
disp('  Tratando as amostras coletadas do sensor da roda do carro de medidas... ');

%---------- Retirando amostras referentes a paradas do carro de medidas ----------
disp(' ');
disp('A seguir será plotado o gráfico com as tensões do sensor da roda, para que
se');
disp('determine quais intervalos devem ser excluídos das medidas a serem usadas
pois representam paradas do carro.')
disp(' ');
ent=input(' Pressione enter para visualizar o gráfico.');
disp(' ');
stem(D);zoom on; grid on; % Imprime o gráfico das tensões do sensor da roda para
sua visualização
ent='a'; % Pergunta
while and(ent~='s',ent~='n')
   ent=input('Existem amostras que devem ser excluídas? <s/n> ','s');
end
Tamanho=EOF;
while ent=='s'
   inicio=input('Digite o número inicial do corte: ');
   fim=input('Digite o número final do corte: ');
   n=0;    % Identificando quais perfis saem. Tem que ser um número inteiro de
perfis.
   while n*NoAP <=inicio
      n=n+1;
   end
   n=n-1;
   m=0;
   while m*NoAP <=fim
      m=m+1;
   end
   disp('  Serão excluídos os perfis que contêm amostras no intervalo a ser
cortado.');
   inicio=n*NoAP+1;
   fim=m*NoAP;  % Pega valores quebrados e os substitui por valores
                      %tais que um perfil não fique pela metade
```

```
   for n=0:EOF-fim-1   % Deslocando os valores para a esquerda
      T(inicio+n)=T(fim+n+1);
      Q(inicio+n)=Q(fim+n+1);
      I(inicio+n)=I(fim+n+1);
      D(inicio+n)=D(fim+n+1);
   end

   for n=EOF-fim+inicio:EOF   % Deslocando os valores para a esquerda
      D(n)=nan;;
   end

   Tamanho=EOF-fim+inicio-1;
   close;
   stem(D);zoom on; grid on;
   ent='a'; % Pergunta
   while and(ent~='s',ent~='n')
      ent=input('Deseja continuar os cortes? <s/n> ','s');
   end
end
EOF=Tamanho; %Atualiza EOF

for n=1:EOF  % Substituindo os vetores exixtentes pelos novos vetores com tamanhos
alterados
   Aux(n)=T(n);
   Aux1(n)=Q(n);
   Aux2(n)=I(n);
   Aux3(n)=D(n);
end
clear T;
T=Aux;
clear Q;
Q=Aux1;
clear I;
I=Aux2;
clear D;
D=Aux3;
% ------------------------------------------------------------------------------
clc;
disp(' ');
disp('Tratando as amostras coletadas do sensor da roda do carro de medidas... ');
disp(' ');
disp('A seguir será plotado o gráfico com as tensões a serem usadas no tratamento,
');
disp('para que se determine qual o limiar entre 0 e 1.');
ent=input('Pressione enter para visualizar o gráfico.');
stem(D,'b');zoom on; grid on;
hold on;
Limiar=input('A partir do gráfico para as tensões da roda, qual limiar usar?');
for n=1:EOF
   if D(n)<Limiar  %Zera tudo o que estiver abaixo do limiar
      D(n)=0;
   end
end

cont=0;
for n=1:EOF  % Esse loop identifica e corrige erros ao redor do limiar
   if D(n)==0
      cont=cont+1;
   elseif D(n)>=Limiar
      if cont<2  % Número de amostras que não caracteriza o espaço entre furos da
roda
         for m=1:cont
            D(n-m)=Limiar;
         end
         cont=0;
      else
         cont=0;
      end
   end
end

D(EOF+1)=0;
for n=1:EOF  % Esse loop zera os valores que não são os máximos das sequências
   max=0;   % com valores acima do limiar
   if D(n)>=Limiar
      cont=0;
```

```
      while D(n+cont)>=Limiar
         if D(n+cont)>max
            max=D(n+cont);
         end
         cont=cont+1;
      end
      for j=0:cont
         if D(n+j)~=max
            D(n+j)=0;
         end
      end
      n=n+cont-1;
   end
end


stem(D,'c');zoom on; grid on;
hold on;

for n=1:EOF % Designando 1 aos máximos e 0 a qualquer outro
   if D(n)> Limiar
      D(n)=1;
   else
      D(n)=0;
   end
end

%A contagem deve ser iniciada quando se tem D=1, para que se possa calcular
%corretamente as distâncias.

%Tomando-se o primeiro "1" de cada série de "1's" e igualando os outros a zero
n=1;
while n<=EOF
   if D(n)==1
      cont=1;
      while D(n+cont)==1
         cont=cont+1;
      end
      for m=1:cont-1
         D(n+m)=0;
      end
   end
   n=n+1;
end

%Substituindo-se os "1's" por sua posição na seqüência
m=1;
while D(m)==0  %  Detecta o primeiro 1 - As distâncias serão referenciadas a ele
   m=m+1;
end
cont=2;
for n=m+1:EOF
   if D(n)==1
      D(n)=cont;
       cont=cont+1;
   end
end

%Igualando-se ao valor do contador todas as amostras até o próximo furo
cont=0;
for n=1:EOF
   if D(n)~=0
      cont=D(n);
   else
      D(n)=cont;
   end
end

disp(' ');
disp('No gráfico o acerto que foi feito nos dados da tensão da roda.');
ent=input('Pressione qualquer tecla para continuar.','s');

for n=1:EOF
   if D(n)==1

n1=n;
      break;%Se tem-se amostras iniciais nulas, estas serão eliminadas
```

```
   end;
end

%Tem-se que eliminar as amostras iniciais, arquivando-se apenas a partir da 1a
válida
k=EOF-n1+1;
for n=1:k
   T1(n)=T(n+n1-1);
   Q1(n)=Q(n+n1-1);
   I1(n)=I(n+n1-1);
   D1(n)=D(n+n1-1);
end;
clear T,Q,I,D

clc;
disp(' ');
disp(' Os dados serão, agora que foram devidamente tratados e completados,
guardados');
disp(' em um arquivo composto pelo número da rota, por ex., Rota1');
disp(' ');

arq=input('No da rota medida:','s');%Rota1,Rota2,etc...
save arq T1 Q1 I1 D1

ent=input('O arquivo foi gravado em Rota... . Pressione enter para encerrar o
programa.');
clc;


% Aquisição2
%-------------------------------------------------------------------------------
clear; clc; %Limpa variáveis e a tela de comando
disp(' ');disp(' ');disp(' ');disp(' ');disp(' ');
disp('  ------------------------------*------------*--------------------------------
');
disp('         Programa para cálculos com as medidas coletadas e tratadas');
disp(' ');
disp('  O objetivo deste programa é fazer os cálculos com as amostras
coletadas,');
disp('         encontrando os parâmetros do canal rádio móvel.');
disp(' ');
disp('  ------------------------------*------------*--------------------------------
');
ent=input('Pressione enter para iniciar o programa.');


% -----------------Carregamento dos dados gravados para o programa--------------
% Os dados  estão armazenados da seguinte forma:
% uma matriz 4 por n, onde a primeira coluna contém as tensões de Trigger
% a segunda coluna contém as medidas do ramo Q, a terceira as medidas do ramo I e
a
%quarta, as tensões do sensor de posição.
% O nome do arquivo será o nome do local onde as medidas foram tomadas.

addpath c:\Leni\MedidasLeni;  %diretorio onde estao os arquivos
clc;close all
disp(' ');

arq=input('No da rota medida:','s');%Rota1,Rota2,etc...
load arq T1 Q1 I1 D1

disp(' ');
disp('  Carregando o arquivo... ');
disp(' ');

EOF=size(T1,2); % Tamanho do arquivo - número de amostras total;
T=T1';%volts
Q=Q1';%V
I=I1';%V
D=D1';%Número que identifica a quantidade de arcos de roda corridos
clear T1,Q1,I1,D1

%-------------------------Obtenção dos perfis de retardo--------------------------
%------------- Alinhamento dos perfis pelo 1o perfil com visada direta------------

%Os perfis devem ser alinhados pelo seu raio direto. Como pode occorrer do raio
%direto nem sempre ser o de maior potência, devido a alguma obstrução, será usada
```

```
%como parâmetro a distância entre a posição inicial de amostragem e a posição
atual.
%Essa distância será determinada em função do número de pulsos contados a partir
%do sensor de distância do móvel. Tendo a distância percorrida, sabe-se o tempo de
%percurso e pode-se alinhar os perfis sem correr o risco de alinhá-los
erradamente.

%Definindo o 1o perfil, pois este dá Distância = 0 e não será deslocado. Ele
apresenta
%um retardo, que é uma constante que aparece em todos os perfis, pois é
correspondente
%à propagação na distância em visada direta TX - RX1

RED = input('Qual o valor do fator de redução da velocidade da fita RED ?');
NoAP = RED.*1000;%NoAP - Número de Amostras por Perfil
P=I.^2 + Q.^2;%A potência é a soma dos quadrados de I e Q,watts
Fi=atan(Q./I);%Informação de fase do sinal de tensão,rad
PdBm=10.*log10(P.*1000);% Potência em dBm
TPN1 = 511./9.99; % Duração real do perfil, em MICROSSEGUNDO
DeltaT =TPN1./(NoAP-1);%Intervalo entre as amostras em microsseg/Resolução de
retardos
retardo=0:DeltaT:TPN1;%Retardos em microsseg.

Dist_AB = input('Qual é a distância inicial entre Tx e Rx?')
disp(sprintf('  A distância inicial entre Tx e Rx é: %g m  ',Dist_AB));

Dist_AC = input('Qual é a distância final entre Tx e Rx?')
disp(sprintf('  A distância final entre Tx e Rx é: %g m  ',Dist_AC));

Dist_BC = input('Qual é a distância percorrida pelo Rx na rota?')
disp(sprintf('  A distância percorrida pelo Rx na rota é: %g m  ',Dist_BC));

disp(sprintf(' '));
disp('Calculando as distâncias entre Tx e Rx para cada perfil... ');
ent=input('Pressione enter para continuar.');
Cos_Angulo = cos((Dist_AC.^2 - Dist_AB.^2 - Dist_BC.^2)./(-2.*Dist_AB.*Dist_BC));

Distancia(1)=Dist_AB;%Distância inicial entre TX-RX

%Será feito um alinhamento inicial nas amostras aquisitadas, eliminando-se as
extras ao
%final de cada perfil, oriundas do deslocamento do receptor, e que são apenas
ruído.
m=0;
n=1;
while n <EOF %Esse while criará a matriz Perfis[m,NoAP] e cada linha é um perfil
alinhado
   m=m+1;
   for cont=1:NoAP
      Perfis(m,cont)=PdBm(n);%Perfis em dBm
      Fi_Matriz(m,cont)=Fi(n);%rad
      Q_Matriz(m,cont)=Q(n);%Volts
      I_Matriz(m,cont)=I(n);%Volts
      if cont==1
         Dperfil(m)=D(n);%No de arcos de roda associado ao início de cada perfil
alinhado
      end;
      n=n+1;%Incremento de cada amostra de retardo
      if n>EOF
         break; % Evita que n estoure o vetor
      end
   end
   k=n-1;
   Arco_da_Roda = 0.0148178;% Distância, em metros, entre 2 furos da roda
   dif(m)=(D(k)-Dperfil).*Arco_da_Roda;%Distância percorrida por cada perfil de
NoAP amostras
   Dist(m)=(Dperfil(m)-1).*Arco_da_Roda;%Distância percorrida entre início de cada
perfil e
   %o primeiro perfil(com retardos incluído)

   if m>=2
     %Distância real entre TX e RX na posição do perfil m
     Dist_aux(m)=(Dist_AB.^2 + Dist(m).^2 -2.*Dist_AB.*Dist(m)*Cos_Angulo).^1./2;

   %Distância real a mais que o RX percorreu a partir da posição anterior até a
atual
     Distancia(m)=Dist_aux(m) - Dist_aux(m-1);
```

```
   end;
   Ret(m)=Distancia(m)./300;%Retardo de cada perfil em relação ao anterior, em
microsseg
   NAm(m)=round(Ret(m)./DeltaT)+1;%No inteiro de amostras associado ao retardo
considerado
   n=k+NAm(m);%Índice do início de cada perfil alinhado
end
clc;
clear Dist_aux

disp(sprintf('Foram encontrados %g perfis',m));
ent='a'; % Pergunta
while and(ent~='s',ent~='n')
   ent=input('Deseja visualizar alguns perfis medidos e alinhados? <s/n> ','s');
end
while ent=='s'
   clc;%Apaga a janela de comando para visualizar novo perfil
   entnum=input('Digite o número do perfil cujo gráfico será visualizado: ');
   if entnum<=m;
      figure(1)
      plot(retardo,Perfis(entnum,:));
      ylabel('Potência (dBm)'),xlabel('T(microsseg)'),title(sprintf('Perfil de
retardos nº %g',entnum));
      zoom on,grid on
   else
      disp('Número inválido');
      disp(sprintf('Foram encontrados apenas %g perfis',m));
   end
   ent='a';%Enquanto não teclar sim ou não, não continua
   while and(ent~='s',ent~='n')
      ent=input('Deseja continuar a visualização? <s/n> ','s');
   end
end

No_Perfis = m; % Número total de Perfis

Max_PdBm = max(Perfis,[],2); % Cria um vetor Max_PdBm[No_Perfis] onde cada
% elemento é o máximo de cada linha, no caso, máximo de cada perfil

Tperfis=input('Duração dos perfis na rota atual, em ms:');%Marcados com o GPS os
tempos
%inicial e final de medida na rota
Deltaeta=Tperfis./(No_Perfis-1);%Intervalo entre os perfis, em ms/Resolução no
tempo(eta)
NPPS=No_Perfis./Tperfis;
tempo=0:Deltaeta:Tperfis;

%----------------------- Normalização dos perfis pelo máximo ---------------------
for m=1:No_Perfis
    Perfis(m,:)=Perfis(m,:)-Max_PdBm(m); %Normaliza os perfis pelo máximo.
Resposta em dBm!
end

%Imprimindo apenas o último perfil para verificação
figure(2)
plot(retardo,Perfis(entnum,:)),ylabel('Potência Relativa Normalizada(dBm)'),...
   xlabel('T(microsseg)'),title(sprintf('Perfil Ruidoso e Alinhado nº
%g',entnum)),...
   zoom on,grid on

ent=input('Continua? <s/n> ','s');
if input=='n'
   break
end;

%Perfis de potência x retardo x tempo(eta)
figure(3)
[retardo,tempo]=meshgrid(retardo,tempo);
mesh(retardo,tempo,Perfis),view(20,10),xlabel('microsseg'),ylabel('ms'),zlabel('dB
m'),...
   title('Perfis de Retardo Ruidosos e Alinhados')

disp('Pausa para checar alinhamento dos perfis pelo retardo e inspecionar o')
disp('percentual de multipercursos válidos')

ent=input('Continua? <s/n> ','s');
if input=='n'
```

```
   break
end;

Perfis_Alinhados(1,:)=Perfis(1,:);%dBm, normalizados
Fi_Matriz_Alinhados(1,:)=Fi_Matriz(1,:);%rad
Q_Matriz_Alinhados(1,:)=Q_Matriz(1,:); %V
I_Matriz_Alinhados(1,:)=I_Matriz(1,:); %V

clear retardo,Perfis,Fi_Matriz,Q_Matriz,I_Matriz


%----------------------------- Suavização do ruído -----------------------------

Perfis_Sem_Ruido=Perfis_Alinhados;%dBm, normalizados
Fase_Sem_Ruido=Fi_Matriz_Alinhados;%rad(As fases são mantidas e necessárias para
Doppler)
Q_Sem_Ruido=Q_Matriz_Alinhados;%V(Necessário para suavização do ruído por wden)
I_Sem_Ruido=I_Matriz_Alinhados;%V(idem)

clear Perfis_Alinhados,Fi_Matriz_Alinhados,Q_Matriz_Alinhados,I_Matriz_Alinhados

entruido=input('Tratamento do ruído segundo Sousa?:<s,n>','s');

if entruido=='s'
%Método de Sousa - Emprego do Limiar de Alarme Falso(CFAR)
%Suaviza o Perfil de Potência NORMALIZADO

   %Segundo Sousa, para medidas outdoor: em geral 5% dos multipercursos são
válidos
   pmv=input('De acordo com a Figura 3, qual é o percentual de multipercursos
válidos?')

   if pmv=0.05;%5% dos multipercursos válidos
      pmvdB=10.*log10(-2.*log(0.5));% Desvio padrão do ruído térmico abaixo da
mediana
   else%Quando o percentual de multipercursos válidos é diferente de 5%
      pmvdB=10.*log10(-2.*log(0.5.*(1-pmv)));
   end;

   %Criando 1 vetor com as medianas dos perfis ruidosos
   Mediana_Perfil = median(Perfis_Sem_Ruido');%mediana em dBm
   eta=-log(5e-6./2)./3;%5e-6=Probabilidade de alarme falso empregada
   etadB=10.*log10(eta);
   CFAR=input('Deseja alterar a Probabilidade de Alarme Falso?:','s')
   if CFAR=='s'
      etadB=input('Qual o novo etadB desejado?:')
   end;

   for m=1:No_Perfis
      Var_Ruido(m)=Mediana_Perfil(m)-pmvdB;%Variância do ruído
      Limiar_Ruido(m)=Var_Ruido(m)+etadB; %Limiar, em dB, abaixo do qual
      %só se tem ruído. Tudo abaixo do limiar é posto ao nível do limite inferior.
   end

   %Limite Inferior do perfil suavizado
   %O limite inferior depende da relação adotada para Psinal/Pruído. Aqui é
   %adotado nível da mediana como o limite inferior, equivalendo a S/N>15 dB
   Limite_Inferior=Mediana_Perfil;
%Passando o que está abaixo do limiar para o limite inferior
   for m=1:No_Perfis
      for n=1:NoAP
         if Perfis_Sem_Ruido(m,n)<=Limiar_Ruido;%dBm normalizado
            Perfis_Sem_Ruido(m,n)=Limite_Inferior(m);
         end
      end
   end

 %Eliminando Ruído Impulsivo e tomando apenas amostras que satisfazem ao critério:
 %um multipercurso válido(> limiar) deve se achar entre 2 amostras maiores que o
 %limiar e ser > ambas.

 for m=1:No_Perfis
    amp(m,1)=Perfis_Sem_Ruido(m,1);%dBm normalizados
    fase(m,1)=Fase_Sem_Ruido(m,1);
    amp(m,NoAP)=Perfis_Sem_Ruido(m,NoAP);
    fase(m,NoAp)=Fase_Sem_Ruido(m,NoAP);
    dnorm(m,1)=abs(Perfis_Sem_Ruido(m,1)-Limite_Inferior(m));%desnormalizando
```

```
    dnorm(m,NoAP)=0;
   for n=2:NoAP-1
      if(  ( Perfis_Sem_Ruido(m,n)> Perfis_Sem_Ruido(m,n+1))&(
Perfis_Sem_Ruido(m,n)>=...
             Perfis_Sem_Ruido(m,n-1))&( ( Perfis_Sem_Ruido(m,n-
1)>Limiar_Ruido(m))|...
             (PSR(m,n+1)>Limiar_Ruido(m)) )  )
          amp(m,n)= Perfis_Sem_Ruido(m,n);
          fase(m,n)=Fase_Sem_Ruido(m,n);
          dnorm(m,n)=abs( Perfis_Sem_Ruido(m,n)-Limite_Inferior(m)
);%desnormalizando
      else
         amp(m,n)=Limite_Inferior(m);
         fase(m,n)=NaN;
         dnorm(m,n)=0;
      end
   end
end;

%Plotando os perfis que se deseja visualizar

   clear entnum,ent

   ent='a'; % Pergunta
   while and(ent~='s',ent~='n')
       ent=input('Deseja visualizar alguns perfis sem ruído(Sousa) <s/n> ?','s');
   end
   while ent=='s'
        clc;%Apaga a janela de comando para visualizar novo perfil
        entnum=input('Digite o número do perfil cujo gráfico será visualizado: ');
        if entnum<=m;
          figure(4)
          plot(retardo,amp(entnum,:),'b',retardo,Limiar_Ruido(entnum),'g',...
          retardo,Limite_Inferior(entnum),'r',ylabel('Potência Relativa
(dBm)'),xlabel...
          ('Retardo(microseg)'),title(sprintf('Perfil de Retardos Suavizado(Sousa)
nº%g',...
          entnum)),legend('g-limiar,r-lim inf'),zoom on,grid on
        else
          disp('Número inválido');
        end
          ent='a';%Enquanto não teclar sim ou não, não continua
        while and(ent~='s',ent~='n')
            ent=input('Deseja continuar a visualização? <s/n> ','s');
        end
   end

 %Buscando tensão complexa para o perfil suavizado(p/ cálculo de Doppler)
 for m=1:No_Perfis
   for n=1:NoAP
      if dnorm(m,n)~=0
         Perfis_Sem_RuidoW(m,n)=(10.^(dnorm(m,n)./10))./1000;%Perfis sem ruído, em
W, ref=0
      else
         Perfis_Sem_RuidoW(m,n)=0;
      end;
   end;
 end;
 R_Sem_Ruido=(Perfis_Sem_RuidoW).^(1./2);

 cfi=cos(fase);
 sfi=sin(fase);
 V_Sem_Ruido=R_Sem_Ruido.*cfi+j.*R_Sem_Ruido.*sfi;
 auxW=Perfis_Sem_RuidoW;%Perfis suavizados pelo método de Sousa, em watts

   clear retardo,tempo,Perfis_Sem_RuidoW,cfi,sfi,R_Sem_Ruido

%Perfis suavizados pelo método de Sousa, em watts
   %Perfil de retardos tridimensional P(tempo(eta),retardos)
   retardo=0:DeltaT:TPN1;%Retardos em microssegundos
   tempo=0:Deltaeta:Tperfis;

   figure(5)
   [retardo,tempo]=meshgrid(retardo,tempo);
   mesh(retardo,tempo,dnorm),view(30,10),xlabel('microsseg'),ylabel('ms'),...
      zlabel('dBm'),title('Perfis de Retardo Suavizados(Sousa)')
```

```
   ent=input('Continua? <s/n> ','s');
if input=='n'
   break
end;

   figure(6)
   %Vetor com o perfil médio entre os perfis sem ruído
   Perfil_Medio_Sousa=mean(Perfis_Sem_RuidoW);
   plot(retardo,Perfil_Medio_Sousa),xlabel('microsseg'),ylabel('dBm'),title...
      ('Perfil Médio Sousa'),zoom on,grid on

%Se for o segundo método(wden)
else
   %Método de "denoising" via função wden do MATLAB
   %Limpa partes real e imaginária do sinal de tensão: I e Q

   disp('Usando a função WDEN');
   %Cálculo de L:
   wname=input('Função wavelet empregada:');%wname=função wavelet usada:sym8
   L=wmaxlev(NoAP,'sym8');
   select = 'sqtwolog'; %Limiar universal usado pela WDEN
   sh = 's';  %s é limiar "soft" e h é limiar "hard"
   %sln representa estimação simples do nível de ruído

   for m=1:No_Perfis;
     VI_Sem_Ruido(m,:)=wden((I_Sem_Ruido(m,:)),select,sh,'sln',L,wname);%I sem
ruído
     VQ_Sem_Ruido(m,:)=wden((Q_Sem_Ruido(m,:)),select,sh,'sln',L,wname);%Q sem
ruído
     V_Sem_Ruido(m,:)=VI_Sem_Ruido(m,:).+j.*VQ_Sem_Ruido(m,:);%Tensão complexa
     Perfis_Sem_RuidoW(m,:)=(abs(V_Sem_Ruido(m,:)).^2;%Potência em watts
     Fase_Sem_Ruido(m,:)=atan(VQ_Sem_Ruido(m,:)./VI_Sem_Ruido(m,:));%Fase em rad
     Perfis_Sem_Ruido(m,:)=10.*log10(Perfis_Sem_RuidoW(m,:).*1000);%dBm
   end;

   clear entnum,ent

   ent='a';
   while and(ent~='s',ent~='n')
       ent=input('Deseja visualizar alguns perfis sem ruído(wden)? <s/n> ','s');
   end
   while ent=='s'
        clc;%Apaga a janela de comando para visualizar novo perfil
        entnum=input('Digite o número do perfil cujo gráfico será visualizado: ');
        if entnum<=m;
           figure(7)
           plot(retardo,Perfis_Sem_Ruido(entnum,:)),ylabel('Potência
Relativa(dBm)'),...
              xlabel('Retardo(microseg)'),title(sprintf('Perfil de Retardos
Suavizado(wden) nº %g',...
              entnum)),zoom on,grid on
        else
           disp('Número inválido');
        end
           ent='a';%Enquanto não teclar sim ou não, não continua
           while and(ent~='s',ent~='n')
                ent=input('Deseja continuar a visualização? <s/n> ','s');
           end
        end

        clear retardo tempo

clear VI VQ VI_Sem_Ruido VQ_Sem_Ruido Perfis_Sem_Ruido

%-------------------Iniciando os passos para o cálculo de Doppler----------------
   %--------1.Janelamento dos Perfis de Retardo ao Longo dos M Perfis(eta)--------
-

%Tomando 1 de cada 3 perfis, pois já foi verificado a estacionariedade
Vcomp=V_Sem_Ruido;

clear V_Sem_Ruido

disp('Tomando 1 a cada 3 perfis: NPmed:')
NPred=fix(No_Perfis./3),%Novo numero de perfis
for i=1:NPred;
    m=3.*(i-1)+1;
```

```
    VSRuido(i,:)=Vcomp(m,:);
end;

clear Vcomp

%Passando a janela de Kaiser de 9 pontos
w=kaiser(NPred,9);%Vetor coluna

for k=1:Nred;
   auxv=VSRuido(:,k);
   auxv1(:,k)=auxv.*w;%Resultado de cada coluna janelada
end
V_Janelados=auxv1;%Vetor coluna

clear auxv auxv1 w V_Sem_Ruido

% ------------2.Cálculo da Transformada Discreta de Fourier em ETA--------------

%A FFT será calculada a partir da tensão e a potência será calculada em seguida.
% Para essa Transformada, deve-se usar a tensão e sua fase
F_ETA = fft(V_Janelados,NPred); %Cria 1 matriz com a FFT em eta p/ cd retardo

for j=1:NPred;
   FFT_ETA(j,:)=F_ETA(j,:);
end;

%Eixos
DeltaT=20e-3;%Microssegundos
T=DeltaT.*(Nred-1);
retardo=0:DeltaT:T;
a=input('Rota unica? <s,n> ','s')

NtotalP=input('Numero total de perfis na rota completa:')
NPPS=NtotalP./Tperfis,%Número de perfis/seg, já excluídas as paradas do móvel
mi=NPPS./2,
DeltaDop=NPPS./(NPred-1),%Resolucao de Doppler
doppler=-mi:DeltaDop:mi;%Hertz

save Dop1 FFT_ETA doppler

Perfis_FFT_ETA=(abs(FFT_ETA)).^2;%Amplitude do perfil de Doppler( >=0 )


%Em dBm
%Calculo da Potencia de Doppler, em dBm
for n=1:Nred;
    PdopdBm(:,n)=10.*log10(Perfis_FFT_ETA(:,n).*1000);%dBm
end;

clear Perfis_FFT_ETA

%Normalizando Doppler
for n=1:Nred;
     Max_Dop(n)= max(PdopdBm(:,n)); %Toma o maximo de cada coluna( Perfil de
Doppler )
 end;

for n=1:Nred;
    Pdopn(:,n)=PdopdBm(:,n)-Max_Dop(n); %Perfis em dBm
end

clear PdopdBm Max_Dop

disp('Tamanho de Pdopn - Perfis de Doppler normalizados')
size(Pdopn)

pause

% O numero de Perfis Doppler sera igual ao numero de amostras de retardo:2048
%Devo tomar apenas a 1a metade do espectro, de 1 a 1024 pontos e por num
%eixo (-10,+10) Hz
 ent='a'; % Pergunta
   while and(ent~='s',ent~='n')
       ent=input('Deseja visualizar alguns perfis de Doppler <s/n> ?','s');
   end
   while ent=='s'
        clc;%Apaga a janela de comando para visualizar novo perfil
```

```
        entnum=input('Digite o número do perfil cujo gráfico será visualizado: ');
        disp('Lembrar de salvar 2 perfis de Doppler:LOS e VEGETACAO')
        if entnum<=Nred;
          ret=retardo(entnum);
          figure(1)

plot(doppler,Pdopn(:,entnum),doppler,Pdopn(:,entnum),'r.');ylabel('Potência
Relativa Normalizada(dBm)'),xlabel('Doppler(Hz)'),...
          title(sprintf('Perfil de Doppler nº%g',entnum)),gtext(sprintf('Retardo
em microsseg = %g',ret)),zoom on,grid on
        else
          disp('Número inválido');
      end;
          ent='a';%Enquanto não teclar sim ou não, não continua
        while and(ent~='s',ent~='n')
            ent=input('Deseja continuar a visualização?(Lembrar de salvar
2)','s');
        end;
    end;

%---------------Cálculo do retardo médio e do espalhamento de retardo-------------
for m=1: No_Perfis,
i=1;
   for n=1:pontos
      if Perfis_Sem_RuidoW(m,n)~=0
         contador(m,i)=n;
         PSemRuidoW(m,n)= (10.^(amp(m,n)./10))./1000;
         i=i+1;
      else
         PSemRuidoW(m,n)=1./1000;
      end;
   end;

% Retardo Médio
for m=1:No_Perfis
   Denominador(m)=0;
end

for m=1:No_Perfis
   Numerador = 0;
   if entruido=='sousa';
      PSemRuidoW(m,n)=auxW(m,n);
   end;
   for n=1:NoAP
      if Perfis_Sem_RuidoW(m,n)~=0;%Quando o perfil é nulo, nada soma aos termos
         Numerador = Numerador + (n-
contador(m,1)).*resolucao.*PSemRuidoW(m,n);%Perfil Limpo em Watts
         Denominador(m) = Denominador(m) + PSemRuidoW(m,n);
      end;
   end;
   Retardo_Medio(m) = Numerador./Denominador(m);
end;

%Espalhamento de Retardo
for m=1:No_Perfis
   Numerador = 0; %O denominador é o mesmo calculado anteriormente
   for n=1:NoAP
      if Perfis_Sem_RuidoW(m,n)~=0;%Quando o perfil é nulo, nada soma aos termos
        Numerador=Numerador+(((n-contador(m,1)).*resolucao-
Retardo_Medio(m)).^2).*PSemRuidoW(m,n);
      end;
   end
      SigmaT(m) = sqrt(Numerador./Denominador(m));
end

clear Numerador,Denominador

Retardo_Medio(1:No_Perfis), SigmaT(1:No_Perfis)

%Calculando as médias e desvios do retardo médio e do espalhamento de retardo por
Rota
disp('Média e desvio padrão do retardo médio')
Media_Retardo_Medio=mean(Retardo_Medio),Desvio_Retardo_Medio=std(Retardo_Medio)

disp('Média e desvio padrão do retardoespalhamento de retardo')
Media_SigmaT=mean(SigmaT),Desvio_SigmaT=std(SigmaT)
```

```
ent=input('Continua? <s/n> ','s');
if input=='n'
   break
end;
clc;

arq=input('No da Rota e Espalhamento de
Retardo','s');%Rota1EspRet,Rota2EspRet,etc...
save arq retardo Sigma_T

%----------Janelamento dos Perfis de Retardo ao Longo dos M Perfis(eta)----------

%Passando a janela de Kaiser de 9 pontos
w=kaiser(No_Perfis,9);

for k=1:NoAP
   auxv=V_Sem_Ruido(:,k);
   auxv1(:,k)=auxv.*w;%Resultado de cada coluna janelada
end
V_Janelados=auxv1;

clear auxv,auxv1,retardo,w

% --------------Cálculo da Transformada Discreta de Fourier em ETA--------------

%A FFT será calculada a partir da tensão e a potência será calculada em seguida.
while and(ent~='s',ent~='n')
    ent=input('Deseja calcular a Transformada de Fourier em ETA (s/n)?','s');
end
if ent=='s'
% Para essa Transformada, deve-se usar tensão e sua fase
Perfis_FFT_ETA = fft(V_Janelados,No_Perfis); %Cria 1 matriz com a FFT em eta p/ cd
retardo
end;
abPerfis_FFT_ETA=abs(Perfis_FFT_ETA);%Amplitude do perfil

clear Perfis_FFT_ETA,retardo,V_Janelados

arq=input('No da Rota e Potência de Doppler','s');%Rota1Dop,Rota2Dop,etc...
save arq doppler abPerfis_FFT_ETA

%------------------------Cálculo do espalhamento Doppler--------------------------

%Perfil Doppler tridimensional P(deslocamento Doppler,retardo)
retardo=0:DeltaT:TPN1;
mi=NPPS./2;DeltaDop=NPPS./(N_Perfis-1);
doppler=-mi:DeltaDop:mi;

figure(10)
[retardo,doppler]=meshgrid(retardo,doppler);
mesh(retardo,doppler,abPerfis_FFT_ETA),view(60,60),ylabel('Hz'),xlabel('microsseg'
),zlabel('dBm'),title('Perfis de Doppler')

%-------------Janelamento dos perfis de retardos ao longo dos retardos------------

%Definindo NBc como o número de amostras alongado para cálculo da Bcoerência
%Alonga-se os perfis antes de janelar

NBc=input('Número de amostras de perfil alongado(potência de 2):')
for m=1:No_Perfis
   for n=1:NoAP;
     if entruido=='s';
       Perfis_Sem_RuidoW(m,n)=auxW(m,n);
     end;
       Matriz_Aux(m,n) = Perfis_Sem_RuidoW(m,n);
   end
   %Crescendo a matriz para diminuir a resolução na freqüência no cálculo da Banda
de Coer.
   for n=NoAP+1:NBc;
      if entruido=='s'; %Se método de "Sousa"
      Matriz_Aux(m,n)=auxW(m,NoAP);%Igualei ao último valor do perfil
      else;%if "wden", = ao último valor.Não preciso da fase, pois é p/calcular
Bcoerência
      Matriz_Aux(m,n)=Perfis_Sem_RuidoW(m,NoAP);
      end
   end;
```

```
end;

% Passando a janela de Kaiser  de 9 pontos no Perfil de retardos
%A janela de Blackman Harris de 3 termos não foi usada porque pede um no ímpar de
pontos
%e o que se tem é uma potência de 2!

w=kaiser(NBc,9);%Retorna um vetor coluna de NBc pontos
% Tenho que multiplicar cada coluna i por w(i)
for k=1:No_Perfis
   aux=Matriz_Aux(k,:);
   aux1(k,:)=aux.*w';%Multiplicando cada perfil por cada elemento da janela
end;
Perfis_Janelados=aux1;

clear aux,aux1,tempo

% -----------Cálculo da Transformada Discreta de Fourier no Retardo-----------

% Passando a Transformada de Fourier nos perfis complexos sem ruído, alongados e
janelados
% Transformada em T, nas NoAP amostras
Perfis_FFT_T_Aux = fft(Perfis_Janelados,[],2);%DFFT em NoAPde cada linha da
matriz(perfis)
abPerfis_FFT_T_Aux=abs(Perfis_FFT_T_Aux);

for k=1:m
   mc(k)=max(abPerfis_FFT_Aux(k,:));
   RTn(k,:)=abPerfis_FFT_T_Aux(k,:)./mc(k);%RT normalizado
end;

% A duração do perfil aumentou para DeltaT.*(NBc-1) e DeltaF vai diminuir
DeltaF = 1/(DeltaT*(NBc-1));%MHz
Fmax=1./DeltaT;

%Função Autocorrelação na freqüencia, tridimensional: RT(tempo,freqüência)
tempo=0:Deltaeta:Tperfis;
Fmeio=Fmax./2;
fr=0:DeltaF:Fmeio;
NM=NBc/2;
n=1:NM;m=1:No_Perfis;
R(m,n)=RTn(m,n);

[fr,tempo]=meshgrid(fr,tempo);
figure(11)
mesh(fr,tempo,R),view(20,10),xlabel('MHz'),ylabel('s'),zlabel('Perfis de
Correlação Normalizados')

clear w,Perfis_Janelados,Perfis_Correlacao,Perfis_FFT_T_Aux,fr

ent=input('Continua? <s/n> ','s');
if input=='n'
   break
end;

% -------------------- Cálculo da banda de coerência dos perfis-------------------

%O máximo de cada perfil é obtido no ponto inicial. Busca-se o ponto a partir
deste
%onde o perfil de RTn já caiu abaixo de 90%(50%).O ponto procurado está entre este
%e o ponto anterior a este, onde RTn está acima de 90%.
for m=1:No_Perfis
   %Vetor com as posições de 50%
   for n=1:NBc
      if RTn(m,n)<0.5
         f50(m)=n; % Guarda a posição n do ponto de 50%/Está depois do ponto
exato.
         f49(m)=n-1; % Guarda a posição n-1 do ponto de 50%/Está antes do ponto
exato.
         P50(m)=RTn(m,n) ;
         P49(m)=RTn(m,n-1);
         break;
      end
   end
   %O que fazer? Ligar os 2 pontos por 1 reta e achar o pto entre eles q dá
RTn=0.5
   %Vetor com as posições de 90%
```

```
   for n=1:NBc
      if RTn(m,n)<0.9
         f90(m)=n; % Guarda a posição n do ponto de 90%/Está depois do ponto
exato.
         f89(m)=n-1; % Guarda a posição n do ponto de 90%/Está antes do ponto
exato.
         P90(m)=RTn(m,n) ;
         P89(m)=RTn(m,n-1);
         break;
      end
   end
end

for m=1:No_Perfis
   Banda_Coerencia_50(m) =(  ( 0.5-P49(m) )./( P50(m)-P49(m) )+( f49(m)-1 )
).*DeltaF;
   Banda_Coerencia_90(m) =(  ( 0.9-P89(m) )./( P90(m)-P89(m) )+( f89(m)-1 )
).*DeltaF;
end
disp('Interpolação Linear - BC90 e BC50')
Banda_Coerencia_90(1:16),Banda_Coerencia_50(1:16)
pause

clear entnum,ent

ent='a'; % Pergunta
while and(ent~='s',ent~='n')
   ent=input('Deseja visualizar alguns perfis de Correlação? <s/n> ','s');
end;
while ent=='s'
   clc;%Apaga a janela de comando para visualizar novo perfil
entnum=input('Digite o número do perfil cujo gráfico será visualizado: ');
   if entnum<=m;
      figure(12)
      n=1:NBc./2;
      fr=(n-1).*DeltaF;
      R(entnum,n)=RTn(entnum,n);
      plot(fr,RTn(entnum,n)),title(sprintf('Perfil de Correlação Normalizado nº
%g',entnum))
      zoom on, grid on
   else
      disp('Número inválido');
   end;
   ent='a';%Enquanto não teclar sim ou não, não continua
   while and(ent~='s',ent~='n')
      ent=input('Deseja continuar a visualização? <s/n> ','s');
   end;
end;

%Média e desvio padrão da banda de coerência
Media_BC_50=mean(Banda_Coerencia_50);
Desvio_BC_50=std(Banda_Coerencia_90);

Media_BC_90=mean(Banda_Coerencia_50);
Desvio_BC_90=std(Banda_Coerencia_90);

arq=input('No da Rota e variável','s');%Rota1BC,Rota2BC,etc...
save arq BC90 BC50 Media_BC90 Desvio_BC90 Media_BC50 Desvio_BC50

%------------------------------ Limiar de Fleury----------------------------------

% Achando os valores máximos e mínimos de SigmaT(espalhamento de retardo)
Disp('Testando o Limiar de Fleury')
SigmaT_Max= max(SigmaT);%Espalhamento de retardo máximo
SigmaT_Min= min(SigmaT);%Espalhamento de retardo mínimo
deltas=(SigmaT_Max-SigmaT_Min)./(No_Perfis-1);%Intervalo de espalhamento
inters=SigmaT_Min:deltas:SigmaT_Max;

LF50=(acos(0.5))./(2.*pi);
LF90=(acos(0.9))./(2.*pi);

for m=1:No_Perfis-1
   BC_Fleury_50(m) = LF50./ SigmaT(m);
    BC_Fleury_90(m) = LF90./ SigmaT(m);
end

%Plotagem da Bc x Espalhamento de retardo juntamente com o limiar de Fleury
```

```
NP=input('Qual perfil deseja visualizar?');
figure(13)
subplot(211),plot(inters,log(Banda_Coerencia_50),'bo',inters,log(BC_Fleury_50),'r*
'),...
   title('blue-BC(50%)/red-BC(50%)Fleury'),xlabel('espalhamento de retardo-
microsseg'),...
   ylabel('banda de coerência-MHz'),zoom on,grid on
subplot(212),plot(inters,log(Banda_Coerencia_90),'bo',inters,log(BC_Fleury_90),'r*
'),...
   title('blue-BC(90%)/red-BC(90%)Fleury'),xlabel('espalhamento de retardo-
microsseg'),...
   ylabel('banda de coerência-MHz'),zoom on,grid on


%------------------------- Lista de variáveis e seus significados ----------------

% Amostras - Variável usada para contar as amostras de cada sequência de Trigger
% Arco_da_Roda - Constante que contem o valor do arco da roda do carro de medição
% Banda_Coerencia_50 - Variável para guardar as BC dos perfis de 50%
% Banda_Coerencia_90 - Variável para guardar as BC dos perfis de 90%
% cont - contador simples.
% D - Informação de distância percorrida
% DeltaT - Intervalo entre amostras em microssegundos
% DeltaF - Intervalo entre frequencias
% Denominador - Vetor usado para calcular somatórios
% Desvio_BC_50 - Desvio padrão das Bandas de Coerência de 50%
% Desvio_BC_90 - Desvio padrão das Bandas de Coerência de 90%
% Desvio_SigmaT - Desvio padrão do espalhamento de retardo
% Desvio_Retardo_Medio - Desvio padrão do retardo médio
% Dist - Vetor que armazena o valor da roda de cada perfil
% Distancia_Inicial - Variável com a distância entre Tx e Rx
% EOF  -  End of File.  Representará o número de dados no arquivo
% eta - valor para cálculo do Limiar de Ruídoruído
% ent - variável para entrada de textos via teclado
% entnum - variável para entrada de números via teclado
% Fi - Informação de fase do sinal
% Fi_Matriz(m,n) - Matriz de Fi
% Fi_Matriz_Alinhados(m,n) - Matriz respetiva a Fi após o alinhamento dos perfis
% I(n) - Vetor com os valores de I medidos em campo.
% j - contador simples.
% k - contador simples.
% l - contador simples.
% Limiar - Valor usado para limiar de decisão entre 0 e 1 para as tensões de
trigger
% Limiar_Ruido - Limiar tal que abaixo só há ruído
% Margem - Tolerância usada para checar se é 0 ou 1 quando há dúvida
% Max_PdBm - Vetor que armazena o valor máximo de cada perfil
% Max_PdBm_Pos - Vetor que armazena a posição dos valores máximos
% Max_Perfis_FFT -Vetor com os máximos de cada perfil de Perfis_FFT
% Mediana_Perfil - Vetor com as medianas dos perfis
% Media_BC_50 - Media da Banda de Coerência de 50%
% Media_BC_90 - Media da Banda de Coerência de 90%
% Media_SigmaT - Média do espalhamento de retardo
% Media_Retardo_Medio - Média do retardo médio
% n - contador simples.
% NAP - Número de amostras perdidas.  Amostras que faltam em uma sequência de
Trigger.
% n1 - Começo de onde as amostras serão tratadas
% NoAP - Número de amostras por perfil
% No_Perfis - Número de perfis total
% Numerador - Variável usada para calcular somatórios
% P - Vetor com potência calculada em função de I e Q.
% P_Matriz(m,n) - Matriz de P
% P_Matriz_Alinhados(m,n) - Matriz respetivas a P após o alinhamento dos perfis
% Pafalso - Limiar de retardo
% PdBm - Potência P em dBm
% Perfil_Medio = Vetor com o perfil médio entre os perfis sem ruído
% Perfis(m,n) - Matriz de PdBm
% Perfis_Alinhados(m,n) - Matriz respetivas a Perfis após serem alinhados pelo
máximo do 1º perfil
% Perfis_FFT_T - Matriz(m,n) com as transformadas discretas de Fourier em T de
Perfis_Janelados
% Perfis_FFT_ETA - Matriz(m,n) com as transformadas discretas de Fourier em ETA de
Perfis_Janelados
% Perfis_FFT_50 - Vetor com os valores de pot. 50% da máxima
% Perfis_FFT_90 - Vetor com os valores de pot. 90% da máxima
% Perfis_Janelados(m,n) - Matriz respetivas a Perfis após o janelamento escolhido
```

```
% Perfis_Sem_Ruido(m,n) - Matriz com os perfis após tratamento dos ruídos
% Perfis_Sem_Ruido_Kaiser(m,n) - Matriz janelada por Kaiser
% Perfis_Sem_Ruido_Harris(m,n) - Matriz janelada por Harris
% Q(n) - Vetor com os valores de Q medidos em campo.
% R - Raiz de P
% R_Matriz(m,n) - Matriz de R
% R_Matriz_Alinhados(m,n) - Matriz respetivas a R após o alinhamento dos perfis
% RED - Fator de redução da fita com os dados
% Retardo_Medio - Vetor com os Retardos médios de cada perfil
% SigmaT - Vetor com os espalhamentos de retardo de cada perfil
% Status - Variável que guarda o valor atual de Trigger, 0 ou 1.
% t(n), q(n), i(n) - Vetores temporários usados para trabalhar os valores de T(n),
Q(n) e I(n);
% T(n) - Vetor com os dados colhidos em campo.  Contém os valores de tensão do
trigger.
% Trocou - Variável que é setada em 1 quando o trigger troca de estado
% w - variável para a função janela de Harris
% x - contador simples.%
```

Programa SIMULAÇÃO

```
%-------------------------------------------------------------------------------
%SIMULAÇÃO DA PN ATÉ A SAÍDA DO PA
*
%Geração das amostras armazenadas nas EPROMs e simulação do sinal à saída do
amplificador de potência(PA)
%-------------------------------------------------------------------------------
%O programa calcula a seqüência PN, filtra, pré-distorce, equaliza e quantiza com
nbits. Fornece as amostras a armazenar nas EPROMs dos ramos I e Q. Tais amostras
são entregues ao DAC. O sinal obtido segue ao FPB, modulador I&Q, FPF,
amplificador linear ZJL-3G e, finalmente, ao PA, para ser entregue à antena e
irradiado.
%-------------------------------------------------------------------------------
%N é o número de bits da PN, Ns é o no de amostras/bit e MN é o número total de
amostras da seqüência
%n é o no de coeficientes do filtro, fre é a freqüência da PN TX
%ts é o intervalo de amostragem e fc é a freqüência da portadora
%To é a largura do bit da PN e Tpn é a duração da PN

clear all,close all,

nbits=input('No de bits no DAC:');

Ns=5;n=5;N=511;MN=Ns.*N;NF=(MN+1)./2;fre=9.9.*10.^6;
MN2=2.*MN;To=1./fre;Tpn=N.*To;ts=Tpn./MN;fc=1.88*10.^9;
gtorad=pi./180;delta=50e06./Tpn;
%-------------------------------------------------------------------------------
%DADOS DO FILTRO FIR DIGITAL REMEZ

f=[0 0.3 0.305 1];m=[1 1 0 0];

%Cálculo do Filtro FIR digital de n+1 coeficientes
b=remez(n,f,m);
[h,w]=freqz(b,1,NF);%1 após b faz o denominador de h igual a 1, caso do filtro FIR

HFI(1:NF)=h(1:NF);
HFI(NF+1:MN)=fliplr(conj(HFI(2:NF)));%Filtro FIR Espelhado

figure(1)
plot(abs(HFI(1:MN))),title('|H(w) do Filtro Remez|'),zoom on,grid on
%-------------------------------------------------------------------------------
%GERAÇÃO DA PN DE 511 BITS

% Polinômio Gerador: P(x)=x9+x4+1
%Valores iniciais escolhidos para o shift register:nk1 a nk8

nk=1;nk1=0;nk2=0;nk3=0;nk4=0;nk5=0;nk6=0;nk7=0;nk8=0;
for it=1:N;
   nk9=xor(nk,nk5);
   s=nk9;
   sn(it)=s;
   nk=nk1;nk1=nk2;nk2=nk3;nk3=nk4;nk4=nk5;nk5=nk6;nk6=nk7;
   nk7=nk8;nk8=nk9;
end;
```

```
%Tirando o nível DC da PN e igualando nível "1" a 0,1 V e nível "0" a -0,1 V
%Tais níveis garantem que se possa empregar a pré-distorção na PN filtrada, sem se
chegar à saturação máxima do PA

for i=1:N;
   ss=sn(i);si(i)=ss;
   if ss==0,
      si(i)=0.1;
   end;
   if ss==1,
      si(i)=-0.1;
   end;
end;

%Amostrando a seqüência PN
ja=0;
for i=1:N;
   sinal=si(i);
   ja1=ja+1;jm=ja+Ns;
   for ja=ja1:jm;
      pn(ja)=sinal;%Amostras da PN
   end;
end;

%Transformada da PN amostrada
F=ts.*fft(pn);

figure(2)
stem(pn(1:MN)),title('PN(t)'),zoom on,grid on
figure(3)
plot(abs(F(1:MN))),title('PN(w)'),zoom on,grid on

%Densidade espectral de potência relativa da PN
for i=1:MN;
   cFi=conj(F(i));
   Ppn(i)=F(i).*cFi./MN;
end;
j1=find(Ppn==max(Ppn));im1=j1;Ppnmax=Ppn(im1);
for i=1:MN;
   Prpn(i)=10.*log10(Ppn(i)./Ppnmax);
   if Prpn(i)<=-50,
      Prpn(i)=-50;
   end;
end;
%Potência média e nível da PN
Pot=0;
for i=1:MN;
   P(i)=Pot+Ppn(i);
   Pot=P(i);
end;
PotPN=delta.*Pot./50;
PotPNdBm=10.*log10(PotPN.*1000)
%-------------------------------------------------------------------------------
%SINAL À SAÍDA DO FILTRO DIGITAL(=função de transferência do FIR digital x
transformada da PN)

for i=1:MN;
   Filt=HFI(i);
   Prod(i)=Filt.*F(i);%PN após o filtro, na freqüência
end;

%pn(t) após o filtro
saidt=1./ts.*ifft(Prod);% saidt é o valor da inversa da PN filtrada

for i=1:MN;
   sai(i)=saidt(i);saii(i)=imag(sai(i));
   sair(i)=real(sai(i));%A pn(t) filtrada dá uma parte imaginária desprezível.

   %As fases da pn filtrada são as da pn, atrasada de 2 amostras, por inspeção
gráfica
   alfa(1)=0;alfa(2)=0;alfa(3:MN)=angle(pn(1:MN-2));
   asai(i)=abs(sai(i));
end;

%Máximo valor da pn(t) filtrada
j2=find(asai==max(asai));im2=j2;asaimax=asai(im2);asaimax
j2=find(asai==min(asai));im2=j2;asaimin=asai(im2);asaimin
```

```
%Densidade espectral de potência relativa da PN filtrada
for i=1:MN;
   cP=conj(Prod(i));
   Ppnf(i)=Prod(i).*cP./MN;
end;
j2=find(Ppnf==max(Ppnf));im2=j2;Ppnfmax=Ppnf(im2);
for i=1:MN;
   Prpnf(i)=10.*log10(Ppnf(i)./Ppnfmax);
   if Prpnf(i)<=-50,
      Prpnf(i)=-50;
   end;
end;
%Potência média da PN filtrada
Pot=0;
for i=1:MN;
   P(i)=Pot+Ppnf(i);
   Pot=P(i);
end;
PotPNf=delta.*Pot./50;
PotPNfdBm=10.*log10(PotPNf.*1000)
%MEDIDAS DE AMPLITUDE E FASE DO PA OBTIDAS COM O ANALISADOR DE REDE

%Ajuste das curvas(NORMALIZADAS)por polinômios de ordem n
%Varia-se n e manda-se plotar a curva ajustada sobre a original, buscando-se o
melhor ajuste.
%n deve ser, no máximo, 1 ordem menor que o número de pontos
%GdB é o ganho medido na freqüência de 1,880 GHz quando Pin é potência de entrada
no PA

Pin=[-5 -4.8 -4.6 -4.4 -4.2 -4.0 -3.8 -3.6 -3.4 -3.2 -3.0 -2.8 -2.6 -2.4 -2.2 -2.0
-1.8 -1.6 -1.4 -1.2 -1.0 -0.8 -0.6 -0.4 -0.2 0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6
1.8 2.0 2.2 2.4 2.6 2.8 3.0 3.2 3.4 3.6 3.8 4.0 4.2 4.4 4.6 4.8 5.0];%Valores em
dBm

x=10.^((Pin-30)./10);Vi=sqrt(50.*x);%Tensão rms de entrada,em volts, relativa à
potência Pin

k=find(Vi==max(Vi));k1=k;Vemax=Vi(k1);

GdB=[49.477 49.321 49.268 49.183 49.112 49.014 48.953 48.903 48.767 48.680 48.610
48.434 48.214 48.142 48.131 47.988 47.868 47.749 47.553 47.431 47.288 47.110
46.844 46.653 46.560 46.387 46.230 46.073 45.899 45.714 45.534 45.354 45.1 44.877
44.703 44.509 44.313 44.130 43.948 43.733 43.538 43.292 43.118 42.931 42.713
42.541 42.311 42.128 41.909 41.753 41.509];

Vo=Vi.*10.^(GdB./20);%Tensão de Saída do PA

ka=find(Vo==max(Vo));ka1=ka;Vsmax=Vo(ka1);
Vout=[Vo./Vsmax]; Vin=[Vi./Vemax];%Vin e Vout normalizados

%Variação de fase
%Fase é a fase de sinal medida à saída do PA para cada Pin considerado
Fase=[-151.35 -150.21 -149.91 -149.30 -149.17 -147.99 -147.18 -146.15 -146.02 -
145.58 -144.91 -145.12 -145.88 -145.08
-141.63 -141.44 -140.50 -139.69 -139.92 -138.26 -137.26
-136.89 -138.23 -137.62 -133.99 -133.38 -132.04 -131.21
-131.02 -130.44 -128.84 -128.23 -127.00 -126.14 -125.29
-124.27 -122.80 -123.95 -122.41 -120.85 -119.58 -118.58
-117.30 -114.80 -113.64 -112.92 -111.87 -111.00 -111.51
-112.48 -110.53];

Fas2=[Fase./360];%Fase normalizada

%Incluindo o ponto Vo=Vi=0 na amplitude e fase,troca-se Vin por Ve e Vout por Vs
Ve=[0 Vin];Vs=[0 Vout];Fas=[-1 Fas2];%Normalizados, incluindo origem

%Interpolando as CURVAS NORMALIZADAS para amplitude e fase do PA
%Polinômio de ajuste de H
pd=polyfit(Ve,Vs,7);%Melhor ajuste alcançado: Polinômio de 7a ordem
piHd=linspace(0.0,1,70);
zD=polyval(pd,piHd);%Polinômio que ajusta a amplitude

%Polinômio de ajuste de G(Tangente à curva de saturação obtida para o PA, partindo
de zero)
Vce=[Ve(1) 0.2041];Vcs=[Vs(1) 0.5404];
pG=polyfit(Vce,Vcs,1);
```

```
piG=linspace(0.0,0.375,70);
G=polyval(pG,0.2041)./0.2041,%Ganho normaliizado
zG=polyval(pG,piG);

%Polinômio de ajuste de fase: interpolação por 2 trechos(reta e curva)
Fas1=[Fas(1) Fas(2)];Ve1=[Ve(1) Ve(2)];
%1o trecho
pf11=polyfit(Ve1,Fas1,1);
pif1=linspace(0,Ve(2),35);
zf1=polyval(pf11,pif1);
%2o trecho
pf00=polyfit(Vin,Fas2,2);%2 é a ordem do polinômio de ajuste
pif=linspace(Ve(2),1,35);%Ve(2) é intervalo aberto
zf=polyval(pf00,pif);
%-------------------------------------------------------------------------------
%PRÉ-DISTORÇÃO DA PN FILTRADA sai(i)
%Pré-distorção da amplitude
for i=1:MN;
   ampli(i)=asai(i)./Vemax;%Módulo da entrada do PA normalizada
   GVin(i)=G.*ampli(i);%normalizado
   pd8(i)=pd(8)-GVin(i);%Aproximado por polinômio de grau 7:grau 0,- pd(1), a grau
7, - pd(8)
   poli=[pd(1) pd(2) pd(3) pd(4) pd(5) pd(6) pd(7) pd8(i)];
   r=roots(poli);ri=[r];vj=ampli(i);
    for j=1:7,
      rj=ri(j);
      imgr=imag(rj);realr=real(rj);
      if imgr==0.0,
         if realr>=ampli(i) & realr<=0.2041,%Vi distorcido=Vi
            vd=vj;
         end;
         if realr>0.2041 & realr<0.7347,%Vi distorcido=raiz do polinômio
             vd=rj;
         end;
         if realr>=0.7347 & realr<1,%Vi distorcido=máximo
            vd=0.9965;
         end;
      end;
   end;
    vdis(i)=vd;%Módulo da amplitude de Ve distorcido, normalizado, no tempo
 end;

 %Módulo máximo e mínimo da tensão pré-distorcida, normalizada
 j1=find(vdis==max(vdis));im1=j1;vdistmax=vdis(im1);vdistmax
 j1=find(vdis==min(vdis));im1=j1;vdistmin=vdis(im1);vdistmin

clear j
%Pré-distorção da fase:
%Fase da saída = fase da entrada - deltafi(vid)
%Fase de entrada: fase da pn filtrada

for i=1:MN;
   vid=vdis(i);%vdist normalizado
   fapa=alfa(i);%Fase de entrada desnormalizada
   if vid <= Ve(2),
      fap=polyval(pf11,vid);%Valor da variação de fase relativa à Vid
   end;
   if vid > Ve(2),
      fap=polyval(pf00,vid);%Valor da variação de fase relativa à Vid
   end;
  fvid=fapa-fap.*2.*pi;%Fase da saída pré-distorcida, desnormalizada
  ffa(i)=fvid;%Fase da saída pré-distorcida, desnormalizada
  cf=cos(ffa(i));sf=sin(ffa(i));

  %Tensão de entrada pré-distorcida, no tempo, em volts:
   ppfa=ppr+j*ppi

   moda=cf+j.*sf;%cos+jsen
   ppm(i)=Vemax.*vdis(i);%|vid(t)| desnormalizado
   ppfa(i)=moda.*ppm(i); %vid(t)desnormalizado
   ppr(i)=real(ppfa(i));ppi(i)=imag(ppfa(i));%Real e Imag de
   vid, desnormalizados
end;

%Sinal prédistorcido na freqüência
PA=ts.*fft(ppfa);
```

```
%Densidade espectral de potência relativa da PN filtrada e pré-distorcida
for i=1:MN;
   cP=conj(PA(i));
   Ppnfpd(i)=PA(i).*cP./MN;
end;
j2=find(Ppnfpd==max(Ppnfpd));im2=j2;Ppnfpdmax=Ppnf(im2);
for i=1:MN;
   Prpnfpd(i)=10.*log10(Ppnfpd(i)./Ppnfpdmax);
   if Prpnfpd(i)<=-90,
      Prpnfpd(i)=-90;
   end;
end;
%Potência média da PN filtrada e pré-distorcida
Pot=0;
for i=1:MN;
   P(i)=Pot+Ppnfpd(i);
   Pot=P(i);
end;
PotPNfpd=delta.*Pot./50;
PotPNfpddBm=10.*log10(PotPNfpd.*1000)
%FUNÇÃO DE TRANSFERÊNCIA H(w) do FPF em dB, MEDIDA com o ANALISADOR DE REDE

%Ajuste de Amplitude por 4 trechos:[1.805 a 1.85),[1.85 a 1.88],(1.88 a
1.91),[1.91 a 1.955]
%Freqüências medidas em Hz
f1=[1.805e9 1.81e9 1.82e9 1.83e9 1.84e9 1.85e9 ];
f3=[1.85e9 1.86e9 1.87e9 1.88e9];f4=[1.88e9 1.89e9 1.90e9 1.91e9];
f2=[1.91e9 1.92e9 1.93e9 1.94e9 1.955e9];

%Amplitudes medidas em dB
M1=[-23 -22.1 -19.2 -14.2 -6.4 -2.6];
M3=[-2.6 -2.3 -2.0 -1.7 ];
M4=[-1.7 -2.0 -1.9 -2.135];
M2=[-2.5 -4.9 -14.8 -24.9 -40];

%Polinômios de Ajuste para a Amplitude
p1e=polyfit(f1,M1,4);p2e=polyfit(f2,M2,3);p3e=polyfit(f3,M3,3);p4e=polyfit(f4,M4,2
);
fi1=linspace(1.805e9,1.85e9,20);fi2=linspace(1.91e9,1.955e9,20);
fi3=linspace(1.85e9,1.88e9,20);fi4=linspace(1.88e9,1.91e9,20);
fpb1=polyval(p1e,fi1);fpb2=polyval(p2e,fi2);fpb3=polyval(p3e,fi3);fpb4=polyval(p4e
,fi4);

%Polinômios de Ajuste para Fase
fa=[1.805e9 1.81e9 1.82e9 1.83e9 1.84e9 1.85e9 1.86e9 1.87e9 1.88e9 1.89e9 1.90e9
1.91e9 1.92e9 1.93e9 1.94e9 1.955e9];
Mf=[471 450.5 437 407.5 343.3 244 168 110 55.3 1.5 -54.7 -122.5 -206.8 -276.8 -
305.5 -311];
pfa=polyfit(fa,Mf,5);%Fases em graus
ff=linspace(1.805e9,1.955e9,50);fap=polyval(pfa,ff);

%Definindo o FPF(Banda de 60 MHz)
MNQ=(MN.*150./50)-2;%7663
%hpb é a amplitude do FPF, em dB
for i=1:MNQ;
   f=1.805e9 + (i-1).*0.15e9./(MNQ-1);
   if f>=1.805e9 & f< 1.85e9,
     hpb=polyval(p1e,f);
     fpf(i)=hpb;
   end;
   if f >=1.85e9 & f<1.88e9,
      hpb=polyval(p3e,f);
      fpf(i)=hpb;
   end;
   if f>=1.88e9 & f<=1.91e9,
      hpb=polyval(p4e,f);
      fpf(i)=hpb;
   end;
   if f>1.91e9 & f<=1.955e9,
      hpb=polyval(p2e,f);
      fpf(i)=hpb;
   end;
   FPF(i)=10.^(fpf(i)./20);%FPF é a amplitude da função de transferência do FPF,
adimensional
   fpfas=polyval(pfa,f);%fpfas é a fase da função de transferência do FPF, em
graus
   fpfase(i)=gtorad.*fpfas;%Fase em radianos
```

```
end;

%Transladando o espectro do FPF para a origem(Equivalente passa-baixa)
%FPF foi suposto igual em ambos os lados. Isto porque o espectro medido para as
freqüências negativas do FPF foi testado, não causando diferença visível no
resultado. Depois é só espelhar.
NFQ=(MNQ+1)./2;%3832
%FPF de 0 a 50 MHz
FPFori(1:MN)=FPF(NFQ:6386);%Módulo
FAori(1:MN)=fpfase(NFQ:6386);%Fase
clear j
%Equivalente passa-baixa complexo do FPF, de 0 a 50 MHz
for i=1:MN;
   ca(i)=cos(FAori(i));sa(i)=sin(FAori(i));
   mcs=ca(i)+j.*sa(i);
   Fcomp(i)=mcs.*(FPFori(i));%EPB complexo do FPF, 0 a 50 MHz
end;
%Para a equalização só será preciso a faixa de 0 a 25 MHz, onde existe a PN
digital

%FUNÇÃO DE TRANSFERÊNCIA DO DAC Hdac NA FAIXA DA PN DIGITAL:0 a 25 MHz
clear j
Hdac(1)=1;
for i=2:NF;
   ij=(i-1)./(MN-1);fs=50;%fs em MHz
   fr=50.*ij;%f em MHz
   argu1(i)=fr./fs;
   Hdac(i)=abs(sinc(argu1(i)));%Função contínua
end;
%-------------------------------------------------------------------------------
%FUNÇÃO DE TRANSFERÊNCIA DO FPB

%Em 25 MHz sua amplitude já é bem baixa: -56.4 dB
%Aproximação polinomial da Amplitude, em dB
%Freqüências medidas

fr=[0 0.3e06 0.4e06 0.45e06 0.5e06 0.55e06 0.65e06 0.75e06 0.8e06 0.9e06 1e06 2e06
3e06 4e06 5e06 6e06 8e06 9e06 10e06 11e06 12e06];
fr1=[12e06  13e06  14e06  16e06  18e06  20e06  23e06  25e06];%fr  aberto  em  12  e  fr1
fechado em 12

%Amplitudes medidas
adB=[0 -0.07 -0.07 -0.07 -0.07 -0.04 -0.06 -0.07 -0.06 -0.07 -0.07 -0.09 -0.12 -
0.14 ...
      -0.21 -0.24 -0.29 -0.34 -0.4 -0.5 -1.13];
adB1=[-1.13 -4.32 -10.27 -22.2 -31.89 -40.03 -50.4 -56.4];

%1o trecho:melhor ajuste com polinômio de 3a ordem
pA=polyfit(fr,adB,3);%
pHd=linspace(0.0,12e06,30);
fpb1=polyval(pA,pHd);%pol. direto

%2o trecho:melhor ajuste com polinômio de 6a ordem
pA1=polyfit(fr1,adB1,6);
pHd1=linspace(12e06,25e06,50);
fpb12=polyval(pA1,pHd1);%pol. direto

%Aproximação polinomial de Fase em graus
fr=[0  0.3e06  0.4e06  0.45e06  0.5e06  0.55e06  0.65e06  0.7e06  0.75e06  0.8e06  0.9e06
1e06 2e06 3e06 4e06 5e06 6e06 8e06 9e06 10e06 11e06 12e06 13e06 14e06 15e06 16e06
18e06 20e06      23e06 25e06];
fgrau=[0 -7 -9 -10.5 -12 -13.3 -15.7 -17 -18 -19.3 -21.8 -24 -48 -72 -98 -122 -148
-204 -235 -268 -305 -352 -399 -450 -477 -496 -520 -535 -552 -560 ];
frad=fgrau.*gtorad;
pf=polyfit(fr,frad,9);%Melhor ajuste: 9a ordem
piF=linspace(0,25e06,50);
fajus=polyval(pf,piF);%Ajuste em rad

%Tomando 1278 amostras em 25 MHz de FPB
%HFPB é a amplitude do FPB e fapb é a fase do FPB, em rad
for i=1:NF;
     f=(i-1).*25e6./(NF-1);
   if f>=0 & f<12e06,
     hfpb=polyval(pA,f);
     HFPB(i)=hfpb;
  end;
  if f>=12e06 & f<=25e06;
```

```
     hfpb=polyval(pA1,f);
     HFPB(i)=hfpb;
  end
  if f>=0 & f<=25e6,
     fafpb=polyval(pf,f);
     fapb(i)=fafpb.*gtorad;
  end;
end;
HFPBad(1:NF)=10.^(HFPB(1:NF)./20);%|H(w)| do FPB, em adimensional

%FPB complexo de 0 a 25 MHz (Fpb)
clear j
for i=1:NF;
   cb(i)=cos(fapb(i));sb(i)=sin(fapb(i));
   mc=cb(i)+j.*sb(i);
   Fpb(i)=mc.*(HFPBad(i));
end;
Fpb(1:NF)=10.^(HFPB(1:NF)./20);%|H(w)| do FPB, em adimensional
Fpb(NF+1:MN)=fliplr(conj(Fpb(2:NF)));%|H(w)| do FPB espelhado

%EQUALIZAÇÃO

%Realizada na faixa de 0-25 MHz da PN digital
%Produto de Hdac com Fpb e EPF(Fcomp), com NF amostras(0 a 25 MHz)
for i=1:NF;
   HD=Hdac(i);Haux=HD;HDNovo(i)=Haux;Fc=Fcomp(i);
   HDxHF(i)=HD.*Fpb(i);%Produto Hdac x Hfpb, complexo
   Hconj(i)=Fc.*HD.*Fpb(i);%Produto DAC x EPF x FPB, complexo. Hconj(25 MHz)= -
60.6 dB
end;
%Espelhando DAC, HDACXFPB e HDACXFPB
DACxFPBxEPB Hconj(NF+1:MN)=fliplr(conj(Hconj(2:NF)));%DACxFPBxEPB
HDxHF(NF+1:MN)=fliplr(conj(HDxHF(2:NF)));%DACxFPB
HDNovo(NF+1:MN)=fliplr(conj(HDNovo(2:NF)));%DAC

clear j
%Equalizando a pn filtrada e pré-distorcida (PA, na freqüência)
for i=1:MN;
   Heq=1./Hconj(i);
   PPY(i)=Heq.*PA(i);%PA equalizada, já espelhada
end;

%Densidade espectral de potência relativa da PN filtrada, pré-distorcida e
equalizada
for i=1:MN;
   cP=conj(PPY(i));
   Peq(i)=PPY(i).*cP./MN;
end;
j2=find(Peq==max(Peq));im2=j2;Peqmax=Peq(im2);
for i=1:MN;
   Preq(i)=10.*log10(Peq(i)./Peqmax);
   if Preq(i)<=-90,
      Preq(i)=-90;
   end;
end;
%Potência média da PN filtrada, pré-distorcida e equalizada
Pot=0;
for i=1:MN;
   P(i)=Pot+Peq(i);
   Pot=P(i);
end;
Poteq=delta.*Pot./50;
PoteqdBm=10.*log10(Poteq.*1000)
%-------------------------------------------------------------------------------
%O QUE VAI PARA AS EPROMs

ppy=1./ts.*ifft(PPY);%2555 amostras no tempo, com ts=20 ns(5am/bit)
pyr(1:MN)=(real(ppy(1:MN)))./20;pyi(1:MN)=(imag(ppy(1:MN)))./20;

m=find(pyr==max(pyr));mn=m;pyrmax=pyr(mn)
m=find(pyr==min(pyr));mn=m;pyrmin=pyr(mn)
m=find(pyi==max(pyi));mn=m;pyimax=pyi(mn)
m=find(pyi==min(pyi));mn=m;pyimin=pyi(mn)

%Quantizando as amostras da PN filtrada,pré-distorcida e equalizada
NumNiveis = 2^nbits;
stepr=(pyrmax - pyrmin)./(NumNiveis-1),%Intervalo de quantização ramo I
```

```
stepi=(pyimax - pyimin)./(NumNiveis-1),%Intervalo de quantização ramo Q

%Truncando em magnitude e somando nível DC
for i=1:MN;
   ppI(i) = round((pyr(i)-pyrmin)./stepr);
   ppQ(i) = round((pyi(i)-pyimin)./stepi);
end;

disp('ppI e ppQ são os valores a serem armazenados na EPROM')

amostras=input('Quer os valores mínimo e máximo das amostras a serem armazenadas
nas EPROMs?<s,n>','s')

if amostras=='s';
k=find(ppI==max(ppI));ij=k;ppImax=ppI(ij);ppImax
k=find(ppI==min(ppI));ij=k;ppImin=ppI(ij);ppImin
k=find(ppQ==max(ppQ));ij=k;ppQmax=ppQ(ij);ppQmax
k=find(ppQ==min(ppQ));ij=k;ppQmin=ppQ(ij);ppQmin
end;

%ARMAZENAMENTO NAS EPROMs

armaz=input('Armazenar os dados nas EPROMs <s,n>?','s',);
if armaz=='s'

%Preenche o vetor da EPROM com 0's
TamMem = 128*1024;%Tamanho da memória EPROM
EPTI = zeros(1,TamMem);
EPTQ = zeros(1,TamMem);

%Coloca as 2555 amostras da sequência pn, de 1 a 2555
PosIni = 1;
PosFin = 2555;
EPTI(PosIni:PosFin) = ppI;
EPTQ(PosIni:PosFin) = ppQ;

%Abre os arquivos intitulados EPTI.BIN e EPTQ.BIN para escrever dado binário
fidI = fopen('EPI2004.bin','wb');
fidQ = fopen('EPQ2004.bin','wb');

%Loop para escrever cada componente do vetor no arquivo aberto com o tipo UNSIGNED
CHAR    que significa 1 byte binário puro
for ind=1:TamMem,
   fwrite(fidI,EPTI(ind),'uchar');
   fwrite(fidQ,EPTQ(ind),'uchar');
end;

%Fecha os arquivos previamente abertos
fclose(fidI);
fclose(fidQ);

%Elimina os vetores da eprom
clear EPTI,EPTQ;

%Término do que é armazenado nas EPROMs
end;
%-------------------------------------------------------------------------------
%SAÍDA DO QUANTIZADOR

ppIa(1:MN)=stepr.*ppI(1:MN)+pyrmin;%Tirando o nível DC
ppQa(1:MN)=stepi.*ppQ(1:MN)+pyimin;%Tirando o nível DC

maxq=input('Quer os valores mínimo e máximo à saída do quantizador?<s,n>','s');

if maxq=='s';
m=find(ppIa==max(ppIa));mn=m;ppIamax=ppIa(mn); ppIamax
m=find(ppIa==min(ppIa));mn=m;ppIamin=ppIa(mn); ppIamin
m=find(ppQa==max(ppQa));mn=m;ppQamax=ppQa(mn); ppQamax
m=find(ppQa==min(ppQa));mn=m;ppQamin=ppQa(mn); ppQamin
end;

%Saída do Quantizador, na freqüência
PPI=ts.*fft(ppIa);PPQ=ts.*fft(ppQa);
%-------------------------------------------------------------------------------
%SAÍDA DO DAC

%Antes da inserção do amplificador ERA-3
```

```
for i=1:MN;
   HDNo=HDNovo(i);
   PII(i)=HDNo.*PPI(i);
   PQQ(i)=HDNo.*PPQ(i);
end;

%Densidade espectral de potência relativa à saída do DAC, ramo I
for i=1:MN;
   cP=conj(PII(i));
   Peq(i)=PII(i).*cP./MN;
end;
j2=find(Peq==max(Peq));im2=j2;Peqmax=Peq(im2);
for i=1:MN;
   Preq(i)=10.*log10(Peq(i)./Peqmax);
   if Preq(i)<=-90,
      Preq(i)=-90;
   end;
end;
%Potência média à saída do DAC, ramo I, antes do ERA-5
Pot=0;
for i=1:MN;
   P(i)=Pot+Peq(i);
   Pot=P(i);
end;
Poteq=delta.*Pot./50;
PotPosDACIdBm=10.*log10(Poteq.*1000)

%Densidade espectral de potência relativa à saída do DAC, ramo Q
for i=1:MN;
   cP=conj(PQQ(i));
   Peq(i)=PQQ(i).*cP./MN;
end;
j2=find(Peq==max(Peq));im2=j2;Peqmax=Peq(im2);
for i=1:MN;
   Preq(i)=10.*log10(Peq(i)./Peqmax);
   if Preq(i)<=-90,
      Preq(i)=-90;
   end;
end;
%Potência média à saída do DAC, ramo Q, antes do ERA-5
Pot=0;
for i=1:MN;
   P(i)=Pot+Peq(i);
   Pot=P(i);
end;
Poteq=delta.*Pot./50;
PotPosDACQdBm=10.*log10(Poteq.*1000);
%-------------------------------------------------------------------------------
SAÍDA DO AMPLIFICADOR ERA-5

%Potência média à saída do amplificador ERA-5 de G=18.46 dB
(= 8.375 x), quando usado sem choke
PotPosERAIdBm=PotPosDACIdBm + 18.46
PotPosERAQdBm=PotPosDACQdBm + 18.46

%Sinal de tensão após o amplificador ERA-5
ppii=8.375./ts.*ifft(PII);
ppqq=8.375./ts.*ifft(PQQ);

teste=input('Quer testar se os sinais de tensão após o amplificador ERA-5 são
reais?<s,n>','s');

if teste=='s';
i=1:MN;
figure(4)
plot(i,real(ppii),'b',i,imag(ppii),'r'),title('Após ERA-5,ramo I,blue-real'),zoom
on,grid on
figure(5)
plot(i,real(ppqq),'b',i,imag(ppqq),'r'),title('Após ERA-5,ramo Q,blue-real'),zoom
on,grid on
end;

%Saída, após ERA-5, na freqüência
PPII=8.375.*PII;
PPQQ=8.375.*PQQ;
%-------------------------------------------------------------------------------
SAÍDA DO FPB
```

```
%Saída do FPB, entrada do modulador I&Q
A(1:MN)=Fpb(1:MN).*PPII(1:MN);%Entrada do I&Q, ramo I
B(1:MN)=Fpb(1:MN).*PPQQ(1:MN);%Entrada do I&Q, ramo Q

Teste1=input('Quer testar se os sinais de tensão após o FPB são reais?<s,n>','s');

if Teste1=='s';
i=1:MN;
a=1./ts.*ifft(A);
b=1./ts.*ifft(B);
i=1:MN;
figure(6)
plot(i,real(a),'b',i,imag(a),'r'),title('Blue-real I(t),Red-imag I(t)'),zoom
on,grid on
figure(7)
plot(i,real(b),'b',i,imag(b),'r'),title('Blue-real Q(t),Red-imag Q(t)'),zoom
on,grid on
end;

%Densidade espectral de potência relativa da PN filtrada, pré-distorcida e
equalizada, após FPB
for i=1:MN;
   cFi=conj(A(i));
   Pa(i)=A(i).*cFi./MN;
end;
j1=find(Pa==max(Pa));im1=j1;Pamax=Pa(im1);
for i=1:MN;
   Pra(i)=10.*log10(Pa(i)./Pamax);
   if Pra(i)<=-90,
      Pra(i)=-90;
   end;
end;
%Potência média à entrada do I&Q, ramo I
Pot=0;
for i=1:MN;
   P(i)=Pot+Pa(i);
   Pot=P(i);
end;
PotI=delta.*Pot./50;
PotIdBm=10.*log10(PotI.*1000)

%Densidade espectral de potência relativa
for i=1:MN;
   cFi=conj(B(i));
   Pb(i)=B(i).*cFi./MN;
end;
j1=find(Pb==max(Pb));im1=j1;Pbmax=Pb(im1);
for i=1:MN;
   Prb(i)=10.*log10(Pb(i)./Pbmax);
   if Prb(i)<=-90,
      Prb(i)=-90;
   end;
end;
%Potência média à entrada do I&Q, ramo Q
Pot=0;
for i=1:MN;
   P(i)=Pot+Pb(i);
   Pot=P(i);
end;
PotQ=delta.*Pot./50;
PotQdBm=10.*log10(PotQ.*1000)
%-------------------------------------------------------------------------
SAÍDA DO MODULADOR I&Q

clear j
%Passando no I&Q, espelhado
SAMOD(1:MN)=A(1:MN)+j.*B(1:MN); %Saída do modulador I&Q, na freqüência

%Retirando 10 dB de Perda de Conversão do I&Q(Equivale a dividir por 3.162 as
tensões e suas transformadas)
SMOD(1:MN)=SAMOD(1:MN)./3.162;%Saída real do modulador

%Densidade espectral de potência relativa após I&Q
for i=1:MN;
   cFi=conj(SMOD(i));
   P(i)=SMOD(i).*cFi./MN;
```

```
end;
j1=find(P==max(P));im1=j1;Pmax=P(im1);
for i=1:MN;
   Pr(i)=10.*log10(P(i)./Pmax);
   if Pr(i)<=-90,
      Pr(i)=-90;
   end;
end;
%Potência média à saída do I&Q
Pot=0;
for i=1:MN;
   P(i)=Pot+P(i);
   Pot=P(i);
end;
PotposIeQ=delta.*Pot./50;
PotposIeQdBm=10.*log10(PotposIeQ.*1000)
%-------------------------------------------------------------------------------
SAÍDA DO FPF

%Passando no Equivalente passa-baixa complexo do FPF, espelhado
Fcomp(NF+1:MN)=fliplr(conj(Fcomp(2:NF)));%Espelhando o FPF
Fout(1:MN)=Fcomp(1:MN).*SMOD(1:MN);%Saída do FPF

%Densidade espectral de potência relativa após FPF
for i=1:MN;
   cFi=conj(Fout(i));
   P(i)=Fout(i).*cFi./MN;
end;
j1=find(P==max(P));im1=j1;Pmax=P(im1);
for i=1:MN;
   Pr(i)=10.*log10(P(i)./Pmax);
   if Pr(i)<=-90,
      Pr(i)=-90;
   end;
end;
%Potência média após FPF
Pot=0;
for i=1:MN;
   P(i)=Pot+P(i);
   Pot=P(i);
end;
PotposFPF=delta.*Pot./50;
PotposFPFdBm=10.*log10(PotposFPF.*1000)
%-------------------------------------------------------------------------------
%SINAL À ENTRADA DO PA

viPA=1./ts.*ifft(Fout);
disp('Amplificador ZJL-3G = 18.5 dB');

%O ganho do amplificador ZJL=18.5 dB(8,4139 x) é multiplicado ao sinal fout
%Usando atenuador de 6 dB, 18.5-6=12.5. Total: + 12.5 dB = fout x 4.217
%Usando atenuador de 5 dB, 18.5-5=13.5. Total: + 13.5 dB = fout x 4.315
%Usando atenuador de 3 dB, 18.5-3=15.5. Total: + 15.5 dB = fout x 5.956
%Usando atenuador de 0.9484 dB,18.5-0.9484=17.5516. Total: + 17.5516 dB =fout x
7,5436
%Sem atenuador:0 dB.Total:18.5 dB =fout x 8,4139

%Acrescentando o ganho citado acima e normalizando a entrada do PA
disp('Escolha do atenuador:0,3,5 ou 6 dB')

atdB=input('Qual o valor do atenuador em dB?')

fatormult=10.^((18.5-atdB)./20);
ampl(1:MN)=fatormult.*abs(viPA(1:MN))./Vemax;%Módulo de vi(t)
%normalizado à entrada do PA

ka=find(ampl==max(ampl));ka1=ka;viPAmax=ampl(ka1)
%-------------------------------------------------------------------------------
%Ganho de Compressão gc
if viPAmax<=0.2041,
   HVi=polyval(pG,viPAmax);
end;
if viPAmax>0.2041,
   HVi=polyval(pd,viPAmax);%HVi normalizada
end;
gc=20.*log10((G.*viPAmax./HVi))
%-------------------------------------------------------------------------------
```

```
%SAÍDA DO PA

clear j
for i=1:MN;
     alf(i)=angle(viPA(i));%Fase de vi(t)não normalizada, à entrada do PA
     amplii=ampl(i);%amplii está normalizado
   %Fase de saída
   if amplii<=Ve(2),
      fasPA(i)=polyval(pf11,amplii);
   end;
   if amplii>Ve(2),
      fasPA(i)=polyval(pf00,amplii);
   end;
   fsaida(i)=2.*pi.*fasPA(i)+alf(i);%Fase de saída desnormalizada

   %Amplitude de saída
   if amplii >= 0 & amplii<= 0.2041;
      ampPA(i)=(polyval(pG,amplii)).*Vsmax;%desnormalizada
   end;
    if amplii > 0.2041 & amplii< 0.7347;
      ampPA(i)=(polyval(pd,amplii)).*Vsmax;%desnormalizada
   end;
   if amplii >= 0.7347 &amplii<1,
      ampPA(i)=0.9965.*Vsmax;%desnormalizada
   end;
   cmaisjs=cos(fsaida(i))+j.*sin(fsaida(i));
   saida(i)=cmaisjs.*ampPA(i);%Saída desnormalizada
end;

%Saída da PNfiltrada após PA, na freqüência
SAIDA=ts.*fft(saida);

%Densidade Espectral de Potência Relativa da PN à saída do PA
for i=1:MN;
   cFo=conj(SAIDA(i));
   Pout(i)=SAIDA(i).*cFo./MN;
end;
j1=find(Pout==max(Pout));im1=j1;Poutmax=Pout(im1);
for i=1:MN;
   Prout(i)=10.*log10(Pout(i)./Poutmax);
   if Prout(i)<=-90,
      Prout(i)=-90;
   end;
end;

%Potência média à saída do PA
delta=50e06./Tpn;
Pot=0;
for i=1:MN;
   P(i)=Pot+Pout(i);
   Pot=P(i);
end;
PotsaidaPA=delta.*Pot./50;
PotsaidaPAdBm=10.*log10(PotsaidaPA.*1000)
%--------------------------------------------------------------------------------
%PLOTAGEM DOS SINAIS

Prp(NF:MN)=Prpn(1:NF);
Prp(1:NF-1)=Prpn(NF+1:MN);

Prf(NF+1:MN)=Prpnf(1:NF);
Prf(1:NF-1)=Prpnf(NF+1:MN);

PrPA(NF+1:MN)=Prout(1:NF);
PrPA(1:NF-1)=Prout(NF+1:MN);

k=1:MN;
ij=(k-1)./(MN-1);
fr=50.*ij-25;
figure(8)
plot(fr,Prp,'g',fr,Prf,'y',fr,PrPA,'m'),title('Densidade  Espectral  Relativa  de
Potência'),legend('verde-PN','amarelo-
PNFiltrada','magenta-Saída do PA'),xlabel('Freqüência(MHz)'),
ylabel('dBm'),zoom on,grid on
```