UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

CARACTERIZAÇÃO ELÉTRICA DE DISPOSITIVOS E CIRCUITOS INTEGRADOS

HÉLDER HENRIQUE GUIMARÃES

ORIENTADOR: JOSÉ CAMARGO DA COSTA

DISSERTAÇÃO DE MESTRADO EM
ENGENHARIA ELÉTRICA

PUBLICAÇÃO: PPGENE.DM - 308/07

BRASÍLIA/DF: AGOSTO - 2007.

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

CARACTERIZAÇÃO ELÉTRICA DE DISPOSITIVOS E CIRCUITOS INTEGRADOS

HÉLDER HENRIQUE GUIMARÃES

DISSERTAÇÃO DE MESTRADO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM ENGENHARIA ELÉTRICA.

APROVADA POR:

Professor José Camargo da Costa, Doutor (ENE-UnB)
(Orientador)

Professor Demartonne Ramos França, Doutor (ENE-UnB)
(Examinador Interno)

Professor Carlos Llanos Quintero, Doutor (ENM-UnB)
(Examinador Externo)

BRASÍLIA/DF, 13 DE AGOSTO DE 2007.

FICHA CATALOGRÁFICA

GUIMARÃES, HÉLDER HENRIQUE	
Caracterização Elétrica de Dispositivos e Circuitos Integrados.	
[Distrito Federal] 2007.	
Dissertação de Mestrado - Universidade de Brasília.	
Faculdade de Tecnologia. Departamento de Engenharia Elétrica.	
1. Medidas	2. LabVIEW
3. GPIB	4. Nanoeletrônica
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

GUIMARÃES, H. H. (2007). Caracterização Elétrica de Dispositivos e Circuitos Integrados Dissertação de Mestrado, Publicação Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 178p.

CESSÃO DE DIREITOS

NOME DO AUTOR: Hélder Henreque Guimarães
TÍTULO DA DISSERTAÇÃO DE MESTRADO:
Caracterização Elétrica de Dispositivos e Circuitos Integrados
GRAU / ANO: Mestre / 2007

Hélder Henrique Guimarães

AGRADECIMENTOS

Aos meus pais por sempre terem me ajudado e me apoiado, mesmo à distância. Ao meu orientador Professor José Camargo da Costa, pelo grande empenho em construir um futuro melhor. A todos os colegas alunos e professores membros da equipe. Sem eles, este trabalho não seria possível.

RESUMO

Neste trabalho foi desenvolvido e implementado um modelo de estrutura para caracterização e teste de dispositivos eletrônicos e circuitos integrados. Este modelo é capaz de validar uma grande variedade de dispositivos e circuitos integrados, inclusive protótipos de SoC (*System on Chip*). O modelo inclui bancadas de testes, instrumentação, procedimentos e automação de processos com a criação de programas usando LabVIEW® e GPIB.

ABSTRACT

In this work, a structure for characterization and test of electronic devices and integrated circuits was developed and implemented. That structure was used to validate a large variety of devices and integrated circuits, including SoC (System on Chip) prototypes. The structure includes test benches, instrumentation, and automated measurement procedures, based upon GPIB bus with software applications developed with the LabVIEW platform.

SUMÁRIO

1 INTRODUÇÃO **1**

1.1 NECESSIDADE DE PROCEDIMENTOS DE CARACTERIZAÇÃO 1

1.2 OBJETIVO DESTE TRABALHO 2

1.3 INFRA ESTRUTURA DE CARACTERIZAÇÃO 2

1.4 CONTEÚDO DA DISSERTAÇÃO 2

1.5 DISPOSITIVOS TESTADOS 3

2 REVISÃO BIBLIOGRÁFICA **5**

2.1 GPIB E LABVIEW 5

2.1.1 O protocolo GPIB 6

2.1.2 Uma visão Geral do LabVIEW 8

2.1.3 Usando GPIB com o LabVIEW 14

2.2 MEDIDAS EM RF 19

2.2.1 Os erros de medidas 19

2.2.2 Erros repetíveis 19

2.3 PARÂMETROS DE ESPALHAMENTO 21

2.3.1 Redes de duas portas 21

2.3.2 Definição 23

2.3.3 Relação com a impedância 25

2.4 O TRANSISTOR MOS-FET 26

2.4.1 O Capacitor MOS 27

2.4.2 Estrutura do Transistor MOS 27

2.4.3 Funcionamento de um Transistor NMOS 29

2.4.4 Regiões de Operação do Transistor NMOS 33

2.4.5 Extração de Parâmetros 36

3 METODOLOGIA PARA MEDIDAS **37**
3.1 MONTANDO UMA ESTRUTURA DE TESTES 37
3.1.1 Bancadas de Testes 37
3.1.2 Operando os instrumentos 38
3.1.3 Cuidados a serem tomados 39
3.2 MODELO DE PROGRAMAÇÃO 40
3.2.1 Sistema de Medidas 40
3.2.2 Os modos de operação 41
3.2.3 Exemplo de programa 43
3.3 MODELO PARA PROGRAMAR <SPOOL> 62

4 PROCEDIMENTOS DE MEDIDAS E RESULTADOS **64**
4.1 CARACTERIZAÇÃO DE TRANSISTORES 64
4.1.1 Sistema de Medidas 64
4.1.2 Descrição do procedimento 65
4.1.3 Medidas, Resultados e Análise de Desempenho 71
4.2 MEDIDAS DINÂMICAS EM *BUFFERS* 87
4.2.1 Sistema de Medidas 87
4.2.2 Descrição do Procedimento 88
4.2.3 Medidas, Resultados e Análise de Desempenho 89
4.3 PARÂMETROS DE ESPALHAMENTO 98
4.3.1 Sistema de Medidas 98
4.3.2 Descrição do Procedimento 98
4.3.3 Medidas, Resultados e Análise de Desempenho 102

5 DISCUSSÃO **106**

6 CONCLUSÃO **109**

7 REFERÊNCIAS BIBLIOGRÁFICAS **110**

8 APÊNDICES **114**
Apêndice A - Instrumentos usados 115
Keithley 2400 Fonte e Medidor 115
Keithley 6517A Eletrômetro e medidor de alta resistência 116

Analisador de redes Agilent 9714ES 117
Gerador de Funções Agilent 33220A 118
Analisador de Espectro Rohde & Schwarz 119
Apêndice B - Comandos SCPI 120
Maiúsculos e minúsculos 120
O uso do ponto e vírgula 121
Comandos comuns 121
Comandos que perguntam 122
Apêndice C - Funções do LabVIEW 123
Função *String Subset* 123
Função *Search and Replace String* 123
Funções de conversão de número para *string* 124
Função *Format Value* 125
Conversão *String* para Número 125
Funções numéricas 126
Funções de comparação 127
Funções para trabalhar com dados indexados 127
Funções de conversão *Array* e *Spreadsheet* 128
Funções de acesso a arquivos 129
Funções de Atraso e Tempo 130
Apêndice D - LabVIEW e GPIB 131
Funções GPIB Send e SendList 131
Função GPIB *Receive* 132
Função GPIB *SetTimeOut* 132
Função *FindLstn* 133
Apêndice E - Caracterização de Transistores 134
Programa IDSxVDS. 135
Programa IDSxVGS 140
SubVI INICIAR 144
SubVI ESCALA. 147
SubVI VOLT 148
Sub-vi LER 152

Sub-vi SALVAR 154
SubVI GRÁFICO 156
Apêndice F - Parâmetros S 159
Programa S11S21 160
Programa S22S12 165
SubVI's INICIAR 169
SubVI MHz 172
SubVI LER 173
SubVI ARQUIVO 175
SubVI GRÁFICO 176

LISTA DE TABELAS

Tabela 2.1 - Exemplo de comandos SCPI 8
Tabela 3.1 - Configuração do Fonte-Medidor Keithley 2400 50
Tabela 3.2 - Comandos para gerar tensão 54
Tabela 3.3 - Comandos SCPI para o Keithley 2400 57
Tabela 4.1 - Valores extraídos das curvas $I_{DS}xV_{DS}$ 76
Tabela 4.2 - Valores extraídos da curva $I_{DS}xV_{GS}$ 76
Tabela 4.3 - Valores extraídos das curvas $I_{DS}xV_{DS}$ 78
Tabela 4.4 - Valores extraídos da curva $I_{DS}xV_{GS}$ 78
Tabela 4.5 - Valores extraídos das curvas $I_{DS}xV_{DS}$ 80
Tabela 4.6 - Valores extraídos da curva $I_{DS}xV_{GS}$ 80
Tabela 4.7 - Valores extraídos das curvas $I_{DS}xV_{DS}$ 82
Tabela 4.8 - Valores extraídos da curva $I_{DS}xV_{GS}$ 82
Tabela 4.9 - Valores extraídos das curvas $I_{DS}xV_{DS}$ 84
Tabela 4.10 - Valores extraídos da curva $I_{DS}xV_{GS}$ 84
Tabela 4.11 - Valores extraídos das curvas $I_{DS}xV_{DS}$ 86
Tabela 4.12 - Valores extraídos da curva $I_{DS}xV_{GS}$ 86
Tabela B.1 - Comandos comuns 121
Tabela E.1 - Configuração do Fonte-Medidor Keithley 2400 144
Tabela E.2 - Configuração do Eletrômetro Keithley 6517A 145
Tabela E.3 - Comandos SCPI para realizar uma medida 145
Tabela E.4 - Comandos SCPI para calibrar o eletrômetro 145
Tabela E.5 - Comandos do SubVI VOLT 148
Tabela E.6 - Comandos para gerar tensão 151
Tabela F.1 - Configurar Analisador para medir S_{11} e S_{21} 171
Tabela F.2 - Configurar Analisador para medir S_{22} e S_{12} 171
Tabela F.3 - Comandos para posicionar o marcador 172

LISTA DE FIGURAS

Figura 2.1 - Terminal GPIB, ligação série e ligação estrela 6
Figura 2.2 - Interface GPIB de um instrumento 7
Figura 2.3 - Interface PCI-GPIB 7
Figura 2.4 - Painel e seu diagrama 9
Figura 2.5 - Constantes e variáveis locais 10
Figura 2.6 - Funções numéricas 10
Figura 2.7 - Várias funções do LabVIEW 10
Figura 2.8 - Estruturas de Loops 12
Figura 2.9 - Estrutura operando com *arrays* 12
Figura 2.10 - Estrutura Case 13
Figura 2.11 - Exemplo de SubVI 14
Figura 2.12 - SubVI no programa principal 14
Figura 2.13 - Funções para trabalhar com GPIB 15
Figura 2.14 - Fluxograma do Programa LSTN 17
Figura 2.15 - Principais funções do programa 17
Figura 2.16 - Código do programa LSTN 18
Figura 2.17 - Programa LSTN em funcionamento 18
Figura 2.18 - Medidas de erros sistemáticos 20
Figura 2.19 - Erros de descasamento 20
Figura 2.20 - Rede com duas portas 21
Figura 2.21 - Ondas incidentes (a1,a2) e refletidas (b1,b2) 23
Figura 2.22 - Transistores MOS canal N e canal P 26
Figura 2.23 - Estrutura do capacitor MOS 27
Figura 2.24 - Símbolo dos transistores MOS 28
Figura 2.25 - Transistor MOS 28
Figura 2.26 - Materiais usados 28
Figura 2.27 - Transistor NMOS, fonte e dreno sem tensão de porta 29
Figura 2.28 - Transistor NMOS tipo enriquecimento 30
Figura 2.29 - Transistor NMOS tipo enriquecimento 31

Figura 2.30 - Estreitamento máximo do canal próximo ao dreno 32
Figura 2.31 - Estrutura MOS com estreitamento máximo do canal 32
Figura 2.32 - Curvas características de dreno 35
Figura 3.1 - Estação de medidas em baixa frequência 38
Figura 3.2 - Estação de medidas para RF 38
Figura 3.3 - Instrumentos no rack 39
Figura 3.4 - Keithley 2400 40
Figura 3.5 - Fluxograma para usar o LabVIEW 42
Figura 3.6 - Escolher um arquivo 43
Figura 3.7 - Fluxograma do modo carregar arquivo 43
Figura 3.8 - Ligar o dispositivo no Keithley 2400 44
Figura 3.9 - Programa DIODO 45
Figura 3.10 - SubVI's do programa DIODO 45
Figura 3.11 - Código do programa DIODO 46
Figura 3.12 - Fluxograma do Programa DIODO 46
Figura 3.13 - Cancelar 47
Figura 3.14 - Modo carregar arquivo 48
Figura 3.15 - Principais funções do programa 48
Figura 3.16 - Fluxograma do modo carregar arquivo 49
Figura 3.17 - SubVI INICIAR 50
Figura 3.18 - As funções do SubVI 50
Figura 3.19 - Fluxograma do SubVI INICIAR 51
Figura 3.20 - SubVI ESCALA 52
Figura 3.21 - Código de exemplo 53
Figura 3.22 - Fluxograma do exemplo 53
Figura 3.23 - SubVI VOLT 54
Figura 3.24 - As funções do SubVI 56
Figura 3.25 - SubVI LER 56
Figura 3.26 - Fluxograma do SubVI LER 56
Figura 3.27 - Nome do Arquivo 57
Figura 3.28 - Principais funções do SubVI 58
Figura 3.29 - SubVI SALVAR 58

Figura 3.30 - Fluxograma do SubVI SALVAR 58
Figura 3.31 - Principais funções do SubVI 60
Figura 3.32 - SubVI GRÁFICO 60
Figura 3.33 - Fluxograma do SubVI GRÁFICO 61
Figura 3.34 - Função *TestSRQ* 62
Figura 3.35 - Função *ReadStatus* 62
Figura 3.36 - SPOOL 63
Figura 3.37 - Funcionamento do Spool 63
Figura 4.1 - Keithley 2400 65
Figura 4.2 - Keithley 6517A 65
Figura 4.3 - Ligação do DUT 66
Figura 4.4 - Entradas e saídas do eletrômetro 66
Figura 4.5 - Ligando os cabos no eletrômetro 67
Figura 4.6 - Circuito equivalente 67
Figura 4.7 - Caracterização do transistor 67
Figura 4.8 - Programa IDSxVDS 68
Figura 4.9 - Fluxograma IDSxVDS 69
Figura 4.10 - Programa IDSxVGS 70
Figura 4.11 - Fluxograma IDSxVGS 70
Figura 4.12 - Identificando o transistor de RF do FAPESP 119 71
Figura 4.13 - A estação de medidas em RF Cascade 72
Figura 4.14 - Transistor de RF sendo testado 72
Figura 4.15 - Curvas do transistor de RF 73
Figura 4.16 - Transistores Isolados do FAPESP 119 74
Figura 4.17 - Curvas I_{DS}xV_{DS} e I_{DS}xV_{GS} do NMOS 1 (Simulado) 75
Figura 4.18 - Curvas I_{DS}xV_{DS} e I_{DS}xV_{GS} do NMOS 1 (Medido) 75
Figura 4.19 - Curvas I_{DS}xV_{DS} e I_{DS}xV_{GS} do NMOS 2 (Simulado) 77
Figura 4.20 - Curvas I_{DS}xV_{DS} e I_{DS}xV_{GS} do NMOS 2 (Medido) 77
Figura 4.21 - Curvas I_{DS}xV_{DS} e I_{DS}xV_{GS} do NMOS 3 (Simulado) 79
Figura 4.22 - Curvas I_{DS}xV_{DS} e I_{DS}xV_{GS} do NMOS 3 (Medido) 79
Figura 4.23 - Curvas I_{DS}xV_{DS} e I_{DS}xV_{GS} do PMOS 1 (Simulado) 81
Figura 4.24 - Curvas I_{DS}xV_{DS} e I_{DS}xV_{GS} do PMOS 1 (Medido) 81

Figura 4.25 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do PMOS 2 (Simulado) 83
Figura 4.26 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do PMOS 2 (Medido) 83
Figura 4.27 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do PMOS 3 (Simulado) 85
Figura 4.28 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do PMOS 3 (Medido) 85
Figura 4.29 - Gerador de Funções HP 3310A 87
Figura 4.30 - Osciloscópio HP 54600A 88
Figura 4.31 - Diagrama em Blocos 88
Figura 4.32 - Localização no FAPESP 112 89
Figura 4.33 - *Buffers*, *pads* de entrada e saída 89
Figura 4.34 - *Buffer* 1 na frequência de 1 kHz 90
Figura 4.35 - *Buffer* 1 na frequência de 100 kHz 90
Figura 4.36 - *Buffer* 1 na frequência de 3 MHz 91
Figura 4.37 - *Buffer* 2 na frequência de 1 kHz 91
Figura 4.38 - *Buffer* 2 na frequência de 100 kHz 91
Figura 4.39 - *Buffer* 2 na frequência de 3 MHz 92
Figura 4.40 - *Buffer* 3 na frequência de 1 kHz 92
Figura 4.41 - *Buffer* 3 na frequência de 100KH 93
Figura 4.42 - *Buffer* 3 na frequência de 3 MHz 93
Figura 4.43 - *Buffer* 4 na frequência de 1 kHz 93
Figura 4.44 - *Buffer* 4 na frequência de 100 kHz 94
Figura 4.45 - *Buffer* 4 na frequência de 3 MHz 94
Figura 4.46 - *Buffer* 5 na frequência de 1 kHz 94
Figura 4.47 - *Buffer* 5 na frequência de 100 kHz 95
Figura 4.48 - *Buffer* 5 na frequência de 3 MHz 95
Figura 4.49 - Localização no FAPESP 119 96
Figura 4.50 - *Buffer* na frequência de 1 kHz 96
Figura 4.51 - *Buffer* na frequência de 100 kHz 97
Figura 4.52 - *Buffer* na frequência de 3 MHz 97
Figura 4.53 - Analisador de redes 98
Figura 4.54 - Selecionar conectores 99
Figura 4.55 - Programa para obtenção de S_{11} e S_{21} 100
Figura 4.56 - Programa para obtenção de S_{22} e S_{12} 100

Figura 4.57 - Programa para obtenção de parâmetros S 101
Figura 4.58 - Indutor de RF 102
Figura 4.59 - Parâmetros S_{22} e S_{12} do indutor 102
Figura 4.60 - Parâmetros S_{11} e S_{21} do indutor 103
Figura 4.61 - Filtro Ressonante 104
Figura 4.62 - Parâmetros S_{12} e S_{22} do filtro 104
Figura A.1 - Keithley 2400 115
Figura A.2 - Keithley 6517A 116
Figura A.3 - Analisador de redes Agilent 9714ES 117
Figura A.4 - Gerador de Funções Agilent 33220A 118
Figura A.5 - Analisador de Espectro Rohde & Schwarz 119
Figura C.1 - Função *String Subset* 123
Figura C.2 - Exemplo de uso da função *String Subset* 123
Figura C.3 - Função *Search and Replace String* 124
Figura C.4 - Search and Replace String substitui palavras 124
Figura C.5 - Funções de conversão de número para *string* 125
Figura C.6 - Função *Format Value* 125
Figura C.7 - Conversão de *string* para numérico 125
Figura C.8 - Funções numéricas 126
Figura C.9 - Funções numéricas 126
Figura C.10 - Funções de comparação 127
Figura C.11 - Função *Array Size* 127
Figura C.12 - Função *Build Array* 128
Figura C.13 - Função *Index Array* 128
Figura C.14 - Função *Array to Spreadsheet String* 128
Figura C.15 - Função *Spreadsheet String to Array* 128
Figura C.16 - Funções de conversão *path*/*string* 129
Figura C.17 - Função *File Dialog* 129
Figura C.18 - Função *Write Characters To File* 130
Figura C.19 - Função *Read Characters From File* 130
Figura C.20 - Função *Read Lines From File* 130
Figura C.21 - Funções de Atraso 130

Figura D.1 - Função *Send* 131
Figura D.2 - Função *SendList* 132
Figura D.3 - Função *Receive* 132
Figura D.4 - Função *SetTimeOut* 132
Figura D.5 - Função *FindLstn* 133
Figura E.1 - SubVI's dos programas de caracterização 134
Figura E.2 - Programa IDSxVDS 135
Figura E.3 - Código do Programa IDSxVDS 136
Figura E.4 - Fluxograma do Programa IDSxVDS 137
Figura E.5 - Escolher um arquivo 138
Figura E.6 - Programa IDSxVDS zera o gráfico 138
Figura E.7 - Programa IDSxVDS no Carregar Arquivo 139
Figura E.8 - Fluxograma do modo Carregar Arquivo 139
Figura E.9 - Programa IDSxVGS 140
Figura E.10 - Programa IDSxVGS no modo Medir 141
Figura E.11 - Fluxograma do Programa IDSxVGS 141
Figura E.12 - Programa IDSxVGS zerar o gráfico 142
Figura E.13 - Programa IDSxVGS no modo Carregar Arquivo 143
Figura E.14 - Fluxograma do modo Carregar Arquivo 143
Figura E.15 - SubVI INICIAR 146
Figura E.16 - Fluxograma do SubVI INICIAR 146
Figura E.17 - SubVI ESCALA 147
Figura E.18 - SubVI VOLT 149
Figura E.19 - Fluxograma do SubVI VOLT. 149
Figura E.20 - Gerar sequência de tensão 150
Figura E.21 - Fluxograma do exemplo 150
Figura E.22 - SubVI LER 153
Figura E.23 - Fluxograma do SubVI LER 153
Figura E.24 - SubVI SALVAR 154
Figura E.25 - Fluxograma do SubVI SALVAR 155
Figura E.26 - Programa IDSxVDS no modo Carregar Arquivo 156
Figura E.27 - SubVI GRÁFICO 156

Figura E.28 - Estrutura CASE 157
Figura E.29 - Programa IDSxVGS no modo Carregar Arquivo. 158
Figura E.30 - SubVI GRÁFICO 158
Figura F.1 - SubVI's dos programas de parâmetros S 159
Figura F.2 - Programa S11S21 160
Figura F.3 - Código do programa S11S21 161
Figura F.4 - Processo para obtenção de parâmetros S 161
Figura F.5 - Escolher um arquivo 162
Figura F.6 - Programa S11S21 zera o gráfico 162
Figura F.7 - Programa S11S21 no modo Carregar Arquivo 163
Figura F.8 - Fluxograma do modo Carregar Arquivo 163
Figura F.9 - Programa S22S12 165
Figura F.10 - Código do programa S22S12 166
Figura F.11 - Processo para obtenção de parâmetros S 166
Figura F.12 - Programa S22S12 zera o gráfico 167
Figura F.13 - Programa S11S21 no modo Carregar Arquivo 167
Figura F.14 - Fluxograma do modo Carregar Arquivo 168
Figura F.15 - SubVI INICIAR 1 170
Figura F.16 - SubVI INICIAR 2 170
Figura F.17 - SubVI MHZ 172
Figura F.18 - SubVI LER S11S21 173
Figura F.19 - SubVI LER S22S12 174
Figura F.20 - Fluxograma do SubVI LER 174
Figura F.21 - SubVI ARQUIVO 175
Figura F.22 - Fluxograma do SubVI SALVAR 176
Figura F.23 - SubVI GRÁFICO 178

LISTA DE SÍMBOLOS, NOMENCLATURAS E ABREVIAÇÕES

ATSS: Array to Spreadsheet String

CMOS: Complementary Metal-Oxide-Semiconductor

DC: Corrente contínua

DUT: Device Under Test

FET: Field Effect Transistor

GPIB: General Purpose Interface Bus

IDS: Corrente entre dreno e fonte

IEEE: Institute of Electrical and Electronics Engineers

LPCI: Laboratório de Projeto de Circuitos Integrados

LTSD: Laboratório De Tratamento De Superfícies De Dispositivos

MOS: Metal Oxide Semiconductor

MOS-FET: Metal Oxide Semiconductor - Field Effect Transistor

NMOS: Transistor MOS canal N

PCI: Peripheral Component Interconnect

PMOS: Transistor MOS canal P

RAM: Random Access Memory

RF: Radio Frequência

ROM: Read Only Memory

SCPI: Standard Commands for Programmable Instruments

SoC: System on Chip

ULA: Unidade Lógico-aritmética

USB: Universal Serial Bus

V_{DS}: Tensão entre dreno e fonte

V_{GS}: Tensão entre porta e fonte

WCTF: Write Characters To File

FESTN: Frac/Exp String To Number

RCFF: Read Characters From File

1 INTRODUÇÃO

1.1 NECESSIDADE DE PROCEDIMENTOS DE CARACTERIZAÇÃO

No mundo atual, a evolução dos sistemas de telecomunicações e dos sistemas de segurança, assim como a automação em indústrias e a eletrônica aplicada na medicina, exige o desenvolvimento cada vez mais acelerado das tecnologias de integração de circuitos. Sistemas SoC (*System on Chip*)[1],[2],[3],[4] que integram transceptores de RF[2], processamento digital[1] e sensores em um único *chip* já estão sendo projetados.

Com isso os circuitos estão ficando cada vez mais complexos e ao mesmo tempo com dimensões menores. A velocidade de operação e a confiabilidade também estão aumentando. Por esse motivo, já se procura substitutos para o transistor MOS, pois a redução de suas dimensões já atingiu o seu limite físico. Dispositivos de efeito quântico e dispositivos eletrônicos moleculares [5] são alternativas emergentes.

Para tornar possível a fabricação desses novos circuitos em escala industrial, as estruturas de testes e caracterização também precisam evoluir. É preciso simular as condições reais do ambiente em que esses novos circuitos serão usados. A quantidade de informações geradas é grande. São necessários procedimentos para organizá-las, transmití-las e armazená-las para uma análise mais detalhada. Os métodos para avaliar os resultados também precisam ser eficientes para garantir a confiabilidade. Tudo isso precisa ser feito com os equipamentos, computadores e ferramentas de programação existentes atualmente no mercado.

1.2 OBJETIVO DESTE TRABALHO

O objetivo deste trabalho consiste em desenvolver um modelo de estrutura para caracterização e teste de circuitos integrados, que possa atender às necessidades de validação dos protótipos para a produção de dispositivos e circuitos integrados em microeletrônica e nanoeletrônica. O modelo leva em conta os cuidados que devem ser tomados no manuseio dos dispositivos e instrumentos de medidas. Os programas criados para operar instrumentos que geram sinais devem usar os recursos de proteção existentes nos mesmos, como, por exemplo, os limitadores de corrente das fontes de alimentação. Os resultados devem ser transferidos para os computadores em forma de figuras, tabelas e textos de maneira a facilitar a portabilidade, análises e conclusões. Os resultados fornecerão informações para futuros projetos.

1.3 INFRA ESTRUTURA DE CARACTERIZAÇÃO

O sistema é constituído por bancadas de teste, instrumentos e acessórios de medidas, interfaces GPIB [4],[6], computadores e a ferramenta de programação LabVIEW [4],[7],[8]. As bancadas são construídas para testar circuitos integrados não encapsulados, possuindo microscópios e posicionadores de precisão que permitem contato com *pads* internos ao *chip*. Os instrumentos de medidas incluem geradores de sinais, fontes de alimentação, osciloscópio digital, analisador lógico, analisador de redes, analisador de espectro, eletrômetro e outros[4]. Nos acessórios constam cabos e conectores, adaptadores, kits de calibração e várias ferramentas. As interfaces GPIB permitem o controle remoto dos instrumentos e a transferência de dados.

1.4 CONTEÚDO DA DISSERTAÇÃO

Primeiramente, o trabalho apresenta uma revisão bibliográfica abordando os principais conceitos que foram trabalhados durante as atividades. Uma visão geral sobre GPIB e LabVIEW permite que o leitor entenda os procedimentos

que foram automatizados com a criação de programas específicos. Um tópico sobre medidas em RF explica como testar e caracterizar dispositivos e circuitos que operam em microondas. O transistor MOS é atualmente o principal dispositivo da maioria dos circuitos integrados. Devido à versatilidade, tamanho e consumo de energia reduzidos, se constrói com eles uma variedade muito grande de células digitais e analógicas. O tópico que explica o funcionamento desses dispositivos é de fundamental importância não só para as medidas de laboratório como também para as atividades de projeto.

Em seguida, o capítulo "Metodologia para Medidas" explica como montar uma estrutura de testes e apresenta modelos criados para facilitar a criação de programas dedicados para automatizar os processos.

O capítulo "Procedimentos de Medidas e Resultados" apresenta de maneira detalhada atividades realizadas no laboratório. Os procedimentos criados são descritos com detalhes.

O capítulo "Discussão" apresenta o conjunto das atividades realizadas no LTSD relacionadas a esta dissertação de mestrado. Os três últimos capítulos são as conclusões, referências bibliográficas e os apêndices.

1.5 DISPOSITIVOS TESTADOS

O projeto Milênio [1],[2],[3],[4], mantido pelo CNPq, tem como objetivo financiar, entre outros, um sistema inteligente de irrigação [1]. Usa-se sensores de umidade, espalhados na área cultivada, para detectar se é necessário regar a plantação. Cada sensor é ligado a um SoC que controla uma válvula d'agua. Um computador central monitora o sistema, se comunicando com os SoC's espalhados através de RF.

A UnB está desenvolvendo o SoC. No LPCI (Laboratório de Projeto de Circuitos Integrados) os protótipos do SoC são projetados com o uso de ferramentas Cadence [9]. Os leiautes são envidos para fabricação. Depois os *chips* produzidos são testados e caracterizados no LTSD, onde são desenvolvidos procedimentos de caracterização.

Os protótipos do SoC para o sistema de irrigação, FAPESP 112 [2],[4] e FAPESP 119 [4], são usados como exemplo neste trabalho, para demostrar os procedimentos desenvolvidos. São *chips* que possuem cricuitos isolados e dispositivos construídos especialmente para caracterização e teste. Foram financiados pelo projeto Milênio. Também o processo de caracterização de um filtro ressonante [10] de RF é apresentado.

2 REVISÃO BIBLIOGRÁFICA

Este capítulo contém uma revisão da teoria necessária para o entendimento dos procedimentos e atividades de laboratório descritos neste trabalho. A primeira parte trata do protocolo de comunicação GPIB e da ferramenta de programação LabVIEW que no laboratório trabalham de maneira integrada. Essa estrutura é fundamental para o desenvolvimento e automação dos procedimentos de teste incluindo facilidades para análise dos resultados. Em seguida vem uma abordagem sobre medidas em RF. É feito um estudo sobre os principais métodos para superar os problemas e garantir a confiabilidade dos testes. Finalmente uma descrição do funcionamento do transistor MOS, o principal componente dos circuitos integrados. Vários transistores isolados foram construídos nos protótipos com *pads* internos para testes. Os resultados provenientes da caracterização em laboratório são de fundamental importância para o sucesso dos futuros projetos.

2.1 GPIB E LABVIEW

GPIB (*General Purpose Interface Bus*) [4],[6] é um protocolo padronizado pelo IEEE para a comunicação de instrumentos de medidas e computadores, permitindo o controle e aquisição de dados. É definido por interfaces, conectores, cabos e comandos de programação que permitem a compatibilidade independente do fabricante.

O LabVIEW [4],[7],[8] é uma ferramenta de programação criada pela National Instruments para facilitar automação em indústrias. LabVIEW é uma ferramenta de programação visual. Os comandos e funções presentes nas linguagens baseadas em texto como Pascal e C são substituídos por ícones e outras figuras que são interligados de maneira semelhante a um diagrama em blocos.

2.1.1 O protocolo GPIB

Em 1965 a Hewlett-Packard® criou um conjunto de interfaces e protocolo com o nome de HP-IB destinado à operação remota de instrumentos de medidas. Mais tarde foi padronizado pelo IEEE e ganhou o nome de GPIB. Como parte integrante do protocolo, o IEEE padronizou também, uma linguagem de programação em forma de texto para configurar e receber dados através do barramento. Surge assim os comandos SCPI (*Standard Commands for Programmable Instruments*)[3], usados para controlar remotamente os instrumentos. Atualmente o padrão está descrito no IEEE 488.2.

Assim, torna-se possível operações como configuração, calibração e realização de medidas remotamente. Pode-se também obter de informações sobre o estado dos instrumentos. Os resultados das medidas são transferidas diretamente para os computadores, evitando erros de leitura.

2.1.1.1 O barramento GPIB

O barramento GPIB possui 24 linhas, sendo 8 linhas de terra (GND), 8 para transmissão de dados, 3 linhas de sinalização para controlar a transferência de dados e 5 linhas para manutenção da interface. A figura 2.1 [6] mostra a extremidade de um cabo GPIB e duas maneiras de se ligar vários instrumentos. Um barramento suporta até 32 instrumentos. Cada instrumento deve ser configurado em um endereço de 1 a 32 para receber os comandos. Não pode haver endereços repetidos. A figura 2.2 [11], [12] mostra a parte posterior de um instrumento com interface GPIB.

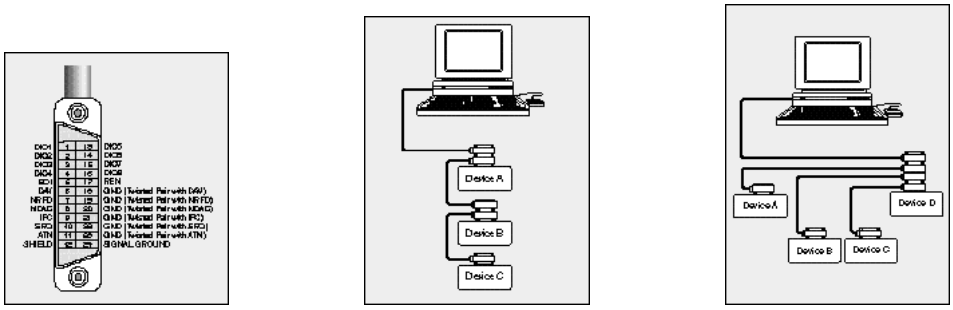


Figura 2.1 - Terminal GPIB, ligação série e ligação estrela.

Figura 2.2 - Interface GPIB de um instrumento.

2.1.1.2 Interface PCI-GPIB

Para usar um computador é necessário uma interface compatível. No nosso laboratório usa-se a placa PCI-GPIB fabricada pela Measurement Computing. Pode ser vista na figura 2.3 [13]. De maneira semelhante aos modems e placas de som, o driver fornecido pelo fabricante deve ser corretamente instalado e configurado.

Figura 2.3 - Interface PCI-GPIB.

2.1.1.3 Comandos SCPI

SCPI é uma linguagem de texto que possibilita operar os instrumentos remotamente. Embora cada instrumento tenha seu conjunto próprio de comandos, eles obedecem regras de formatação padronizadas pelo IEEE. Geralmente eles começam com o sinal dois pontos. Quando existe subcomandos eles devem ficar na mesma linha, separados por dois pontos. Os valores numéricos, *booleans* ou *strings* são separados pelo espaço. Na tabela 2.1 abaixo temos um exemplo. O apêndice B apresenta mais detalhes sobre os comandos SCPI.

*RST	Reset - Coloca na configuração padrão.
:SOUR:FUNC VOLT	Configura para fornecer tensão.
:SOUR:VOLT 6	Fornecer tensão de 6 V
:SENS:FUNC "CURR"	Configura para medir corrente.
:SENS:CURR:RANG 5.0E-6	Ajusta a escala de corrente em 5μA.
:OUTPUT ON	Ativa a saída

Tabela 2.1 - Exemplo de comandos SCPI.

2.1.2 Uma visão Geral do LabVIEW

2.1.2.1 A ferramenta de programação LabVIEW

Embora o objetivo deste trabalho não seja aprofundar na programação em LabVIEW, é necessário a explicação de alguns conceitos e funções para possibilitar o entendimento dos procedimentos de medida. Criado pela National Instruments para facilitar automação em indústrias, o LabVIEW é uma ferramenta de programação visual. Os comandos e funções presentes nas linguagens baseadas em texto, como Pascal e C, são substituídos por ícones e outras figuras que são interligados através de fios.

2.1.2.2 Ambiente de trabalho do LabVIEW

O LabVIEW apresenta duas telas. Uma delas se chama painel, onde são colocados os controles e indicadores. Essa tela será a interface do programa onde o usuário irá operar. Controles são entradas de dados representados por knobs, chaves, caixas de textos e vários outros. Indicadores são as saídas representadas por gráficos, leds e vários outros. A outra tela é o diagrama, onde será feita a programação. Os controles e indicadores colocados no painel aparecem automaticamente no diagrama representados por ícones. A cor dos ícones varia de acordo com o tipo de dados. Por exemplo, verde para *boolean*, azul para números inteiros, laranja para números fracionários e rosa para *strings*. Nessa tela, são

acrescentadas funções, estruturas e constantes numéricas interligadas por fios, determinando, assim, o funcionamento do programa. Na figura 2.4 temos um painel ao lado de seu diagrama. O painel e o diagrama são salvos no mesmo arquivo de código, chamado de VI (*Virtual Instrument*). Os arquivos de códigos poderão ser transformados em executáveis com o *LabVIEW Application Builder*, que é opcional na compra do LabVIEW. Neste trabalho, usa-se o LabVIEW versão 6.1.

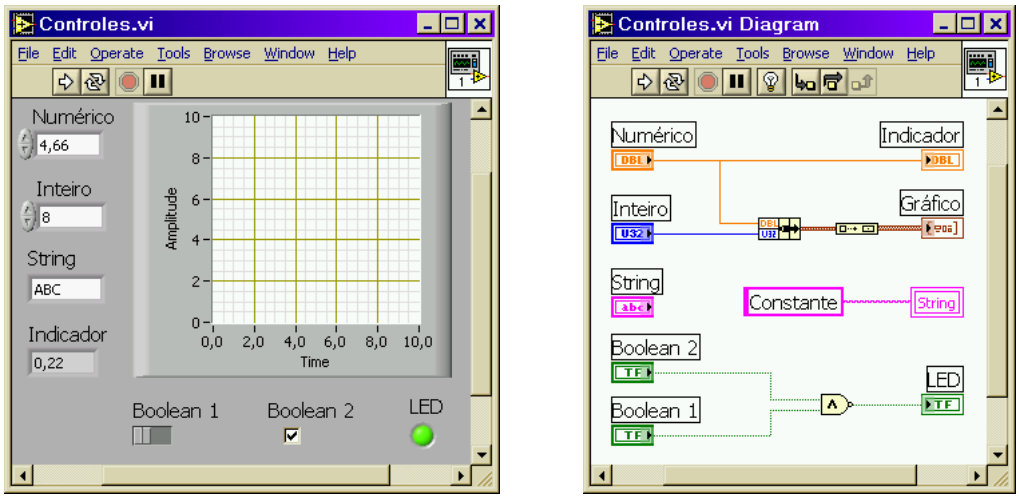


Figura 2.4 - Painel e seu diagrama

2.1.2.3 Constantes e variáveis locais

Constantes são representadas por retângulos que envolvem seus valores. A cor varia de acordo com o tipo de dado. Na figura 2.4 podemos ver o rosa para *string*, verde para *boolean*, laranja para números fracionários e azul para inteiros. Os comentários e os nomes dos controles e indicadores aparecem envolvidos por linhas pretas. Não se pode ligar fios neles. Servem apenas para orientar o programador. Variáveis locais são derivadas dos controles e indicadores. Elas permitem que o programa escreva neles quando não pode alcançá-los com fios. São representadas por dois retângulos envolvendo o nome do seu representante. No exemplo da figura 2.5 o valor das constantes "*True*" e "-610" são transferidos respectivamente ao LED (*True* = Aceso e *False* = Apagado) e ao Inteiro quando o programa é executado.

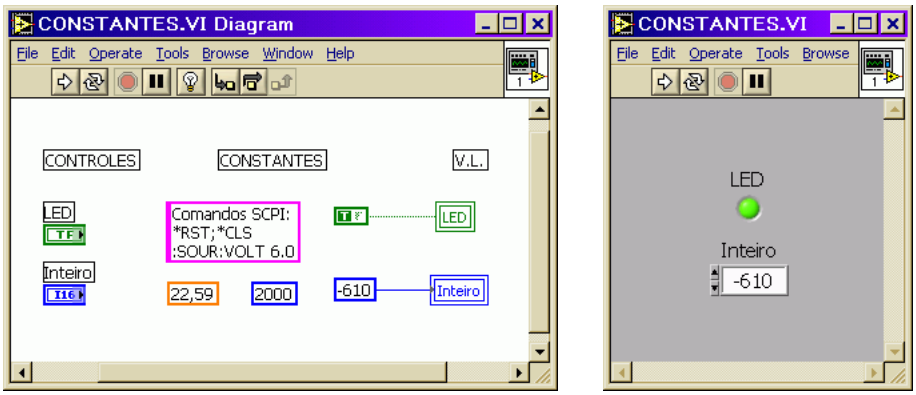


Figura 2.5 - Constantes e variáveis locais

2.1.2.4 Funções no LabView

As funções são representadas por ícones. Existe uma diversidade de funções que podem ter várias entradas e várias saídas. As funções convertem dados, fazem operações matemáticas e acessam interfaces. A figura 2.6 apresenta algumas funções numéricas. A figura 2.7 apresenta funções diversas. No apêndice C pode-se encontrar mais detalhes sobre as funções.

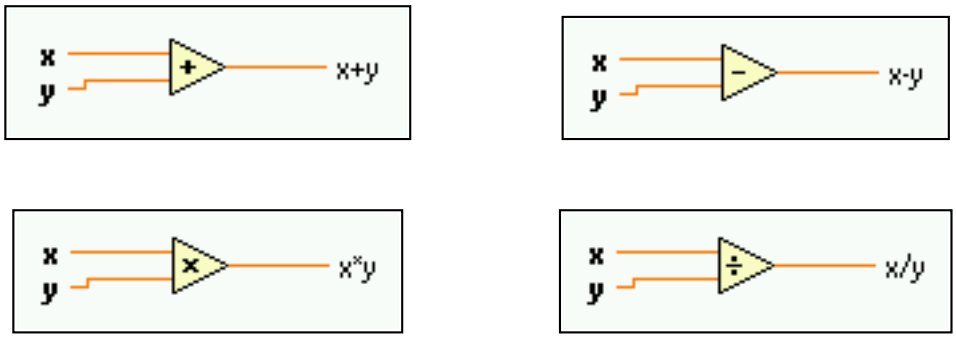


Figura 2.6 - Funções numéricas.

Figura 2.7 - Várias funções do LabVIEW

2.1.2.5 Arrays e Clusters

Os dados indexados são chamados no LabVIEW de *arrays*. Podem ser construídos por funções ou gerados nas estruturas de *loops*. Equivalem às variáveis indexadas nas linguagens de programação por texto. Por exemplo, A(1)=12; A(2)=33; B(0,1)=88; B(2,2)=40. São transportados por fios.

Os *clusters* são agrupamentos de dados em um único fio. São gerados e manipulados pelas funções de *clusters*. Os dados transportados podem ser de tipos diferentes. Um *cluster* pode transportar inclusive vários *arrays*. Os *clusters* de *arrays* são usados para enviar dados aos gráficos.

2.1.2.6 Estruturas no LabVIEW

As estruturas realizam repetições (Ex: *While Loop*) e fazem escolhas (Ex: *Case*). São representadas por quadros que envolvem parte do diagrama. Dentro delas pode conter funções, controles, indicadores e outras estruturas.

2.1.2.7 Estruturas de Loops

A estrutura *While Loop* repete as funções inseridas no seu interior até que uma condição *boolean* seja satisfeita. Já a estrutura *For Loop* repete um número de vezes especificado na entrada "N" . O índice "i" presente nas duas funções é uma saída que conta de 0 a N-1. É usado em cálculos matemáticos dentro dos programas. Funções de tempo e sincronismo são frequentemente usadas para determinar a velocidade em que os *loops* ocorrem. Na figura 2.8 podemos ver as duas estruturas contendo funções de tempo.

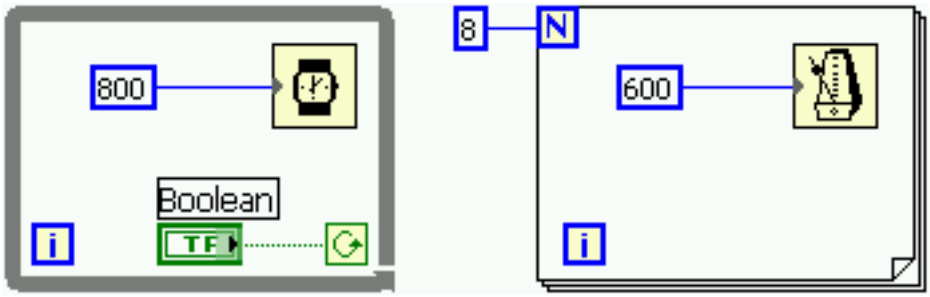


Figura 2.8 - Estruturas de Loops

Ao entrar em uma estrutura, um *array* pode ter seus dados separados. Para cada execução é selecionado o valor da posição correspondente ao índice da estrutura. Os colchetes [ ] no nó de passagem indicam a operação.

Ao sair de uma estrutura os dados podem ou não ser indexados, ou seja, colocados em *arrays*. A figura 2.9 mostra uma estrutura com um *array* entrando e outro *array* mais um número saindo. O nó do fio que vai para o indicador *número* não tem colchetes. Quando o dado não é indexado, apenas o valor resultante da última execução é transferido.

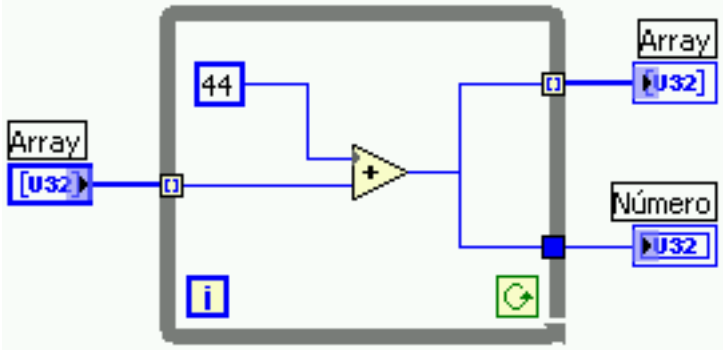


Figura 2.9 - Estrutura operando com *arrays*

2.1.2.8 Estrutura Case

A estrutura *Case* consta de duas ou mais áreas sobrepostas onde se colocam funções, controles, etc. Procedimentos diferentes são realizados de acordo com o valor de sua entrada podendo ser *boolean* ou número inteiro. A figura 2.10 mostra a mesma estrutura quando a entrada é *False* e quando é *True*. No primeiro caso é calculado o seno e no segundo o cosseno.

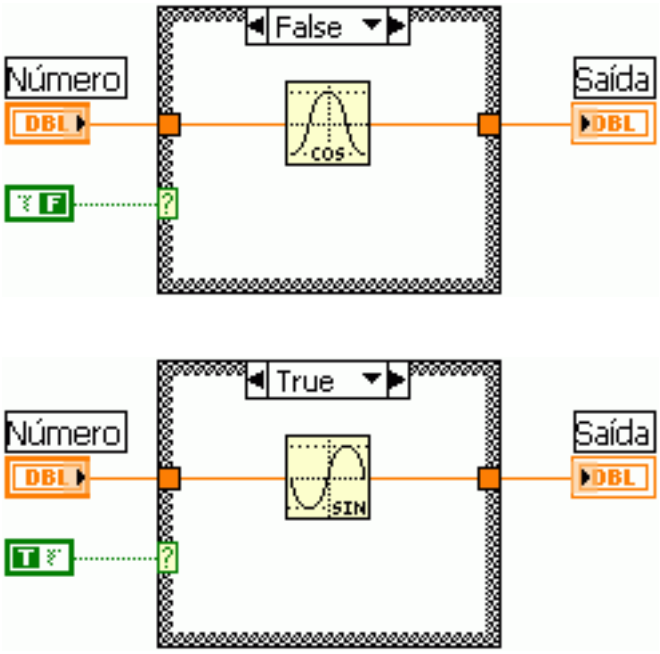


Figura 2.10 - Estrutura Case

2.1.2.9 SubVI's

Um SubVI é um arquivo de código que equivale às sub-rotinas das linguagem de programação baseadas em texto. Ao arquivo criado é atribuído um ícone, e os controles e indicadores são transformados em entradas e saídas, respectivamente. Então ele pode ser inserido no código principal de forma semelhante a uma função.

A figura 2.11 mostra o código de um SubVI. Ele configura o Medidor LCR de Precisão Agilent 4284A para medir indutância série e resistência série, representado pela abreviação LSRS. Repare o ícone com letras vermelhas acima à direita. Na figura 2.12 temos o programa principal com o SubVI inserido, podendo ser facilmente identificado pelo ícone.

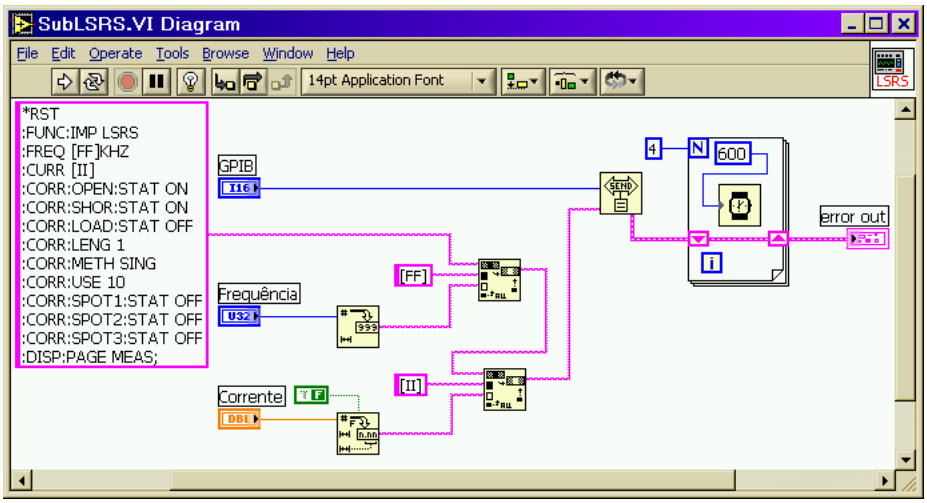


Figura 2.11 - Exemplo de SubVI.

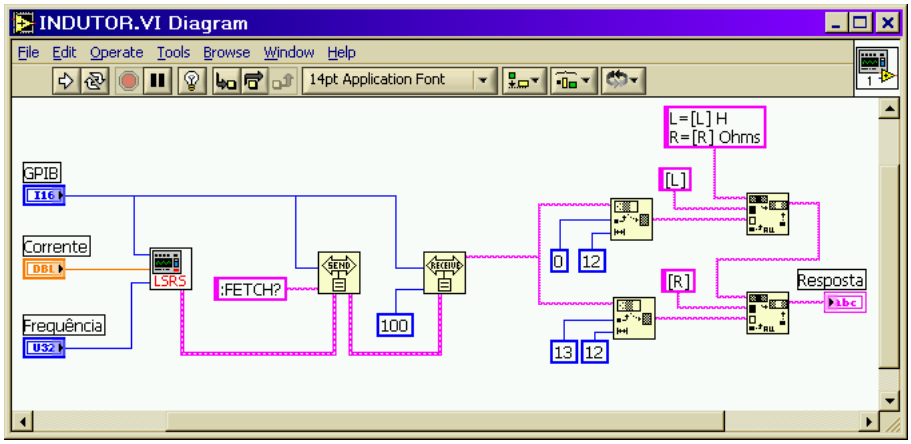


Figura 2.12 - SubVI no programa principal.

2.1.3 Usando GPIB com o LabVIEW

2.1.3.1 Funções do LabVIEW para GPIB

O LabVIEW possui um driver chamado GPIB 488.2 que traz um conjunto de funções para trabalhar com GPIB. A figura 2.13 mostra as funções disponíveis. Apenas algumas delas são usadas neste trabalho. O apêndice D apresenta mais detalhes sobre as funções GPIB que são usadas neste trabalho.

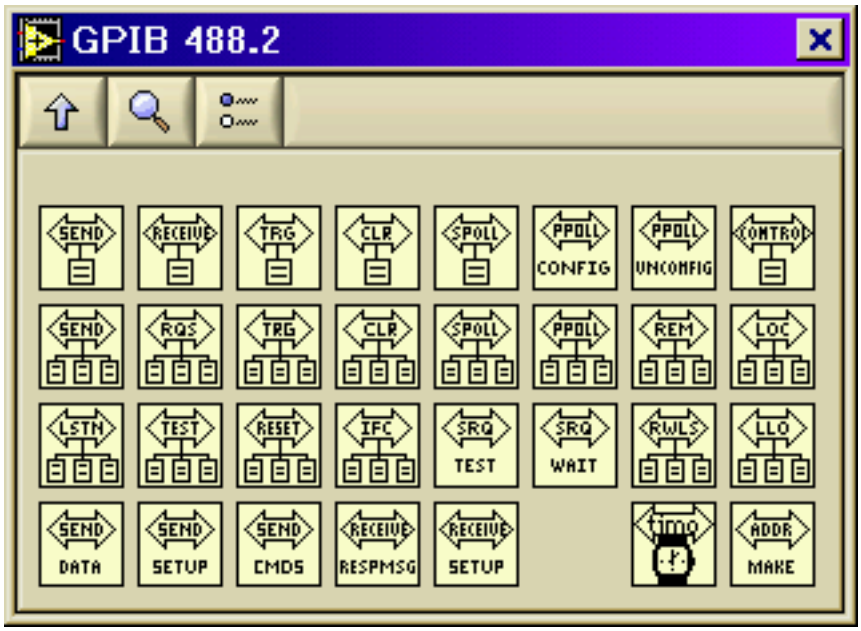


Figura 2.13 - Funções para trabalhar com GPIB.

2.1.3.2 Verificando o barramento - LSTN

Este programa mostra os endereços e identifica todos os instrumentos presentes em um barramento GPIB. Uma análise facilitará o entendimento dos programas mais complexos. Portanto é conveniente dividir o programa em cinco partes:

A primeira parte é composta basicamente por uma estrutura *For Loop* e pela função GPIB *FindLstn*. O índice da estrutura é incrementado e transformado em *array* ao atravessar a borda da estrutura. Forma uma lista com todos os endereços a serem varridos com os valores: E[0]=1, E[1]=2, E[2]=3, ... E[29]=30. Esse *array* é inserido na entrada *address list* da função *FindLstn*. Ela apresenta, na sua saída *listener address list* um *array* com os endereços que possuem instrumentos. A saída *number of listeners* contém o número de instrumentos encontrados. É ligada à entrada *N* da segunda estrutura para determinar o número de execuções.

A segunda parte é formada pela função GPIB *SendList* e pela *string* constante contendo o comando SCPI "*IDN?" . Este comando faz com que os instrumentos disponibilizem seus dados de identificação ID, contendo marca, modelo e outras informações. A função *SendList* recebe o *array* com os endereços dos instrumentos e envia o comando a todos eles.

A terceira parte, dentro da segunda estrutura *For Loop*, é composta basicamente pela função GPIB *Receive*. Recebe as informações dos instrumentos. O *array* proveniente da função *FindLstn* ao passar pela borda da estrutura é desindexado. Isso significa que é selecionado um endereço de cada vez para ser ligado à entrada *address* da função *Receive*. Assim os ID's dos instrumentos presentes são recebidos um por um.

A quarta parte, também dentro da estrutura *For Loop*, é formada pelas funções *Format Value*, *Concatenate Strings* e a constante *string* com o código de formatação "%02D:" . O número do endereço é transformado em *string* com dois caracteres seguido de dois pontos. Depois é unido ou concatenado ao ID do instrumento selecionado para formar uma palavra contendo o endereço GPIB e a identificação do instrumento.

A quinta parte organiza os dados para serem exibidos na tela. É formada pela função *Array to Spreadsheet String* (ATSS), duas constantes *strings* de formatação e o indicador LISTA. As palavras resultantes da quarta parte ao passar pela borda da estrutura são transformadas em *array*. A função *Array to Spreadsheet String* as coloca em forma de tabela, no caso com uma só coluna e entrega ao indicador. O código "\n" na entrada *delimiter* faz com que o retorno seja usado para separar os conteúdos, deixando-os um embaixo do outro. O código "%s" na entrada *format string* indica que o *array* de entrada é do tipo *string*.

O fluxograma na figura 2.14 identifica as funções com nome destacado em letras azuis. A figura 2.15 identifica as principais funções usadas no programa. A figura 2.16 mostra o código do programa. Por fim, o resultado pode ser conferido na figura 2.17, que mostra o programa em funcionamento.

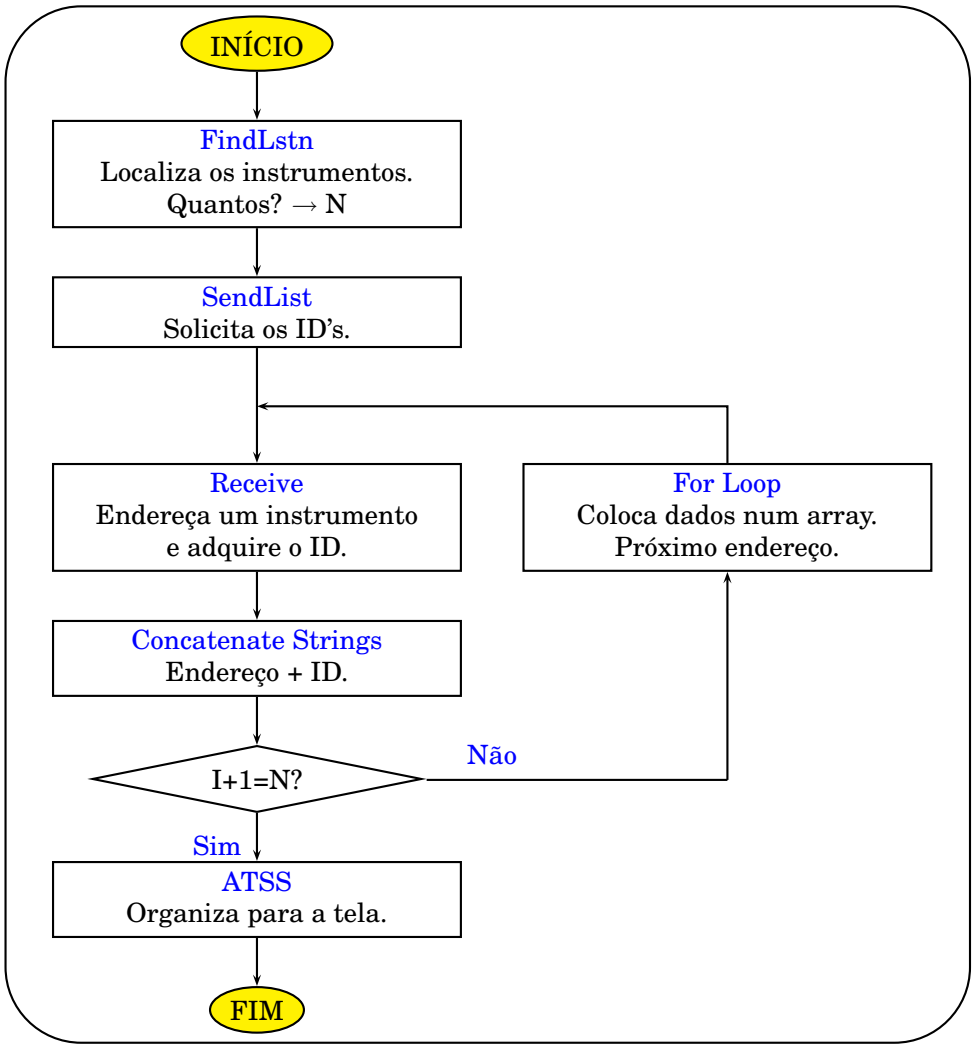


Figura 2.14 - Fluxograma do Programa LSTN

Figura 2.15 - Principais funções do programa

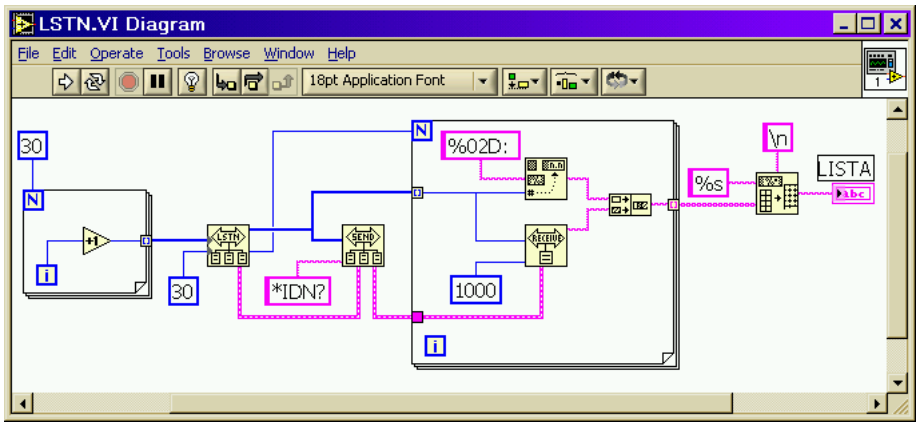


Figura 2.16 - Código do programa LSTN.

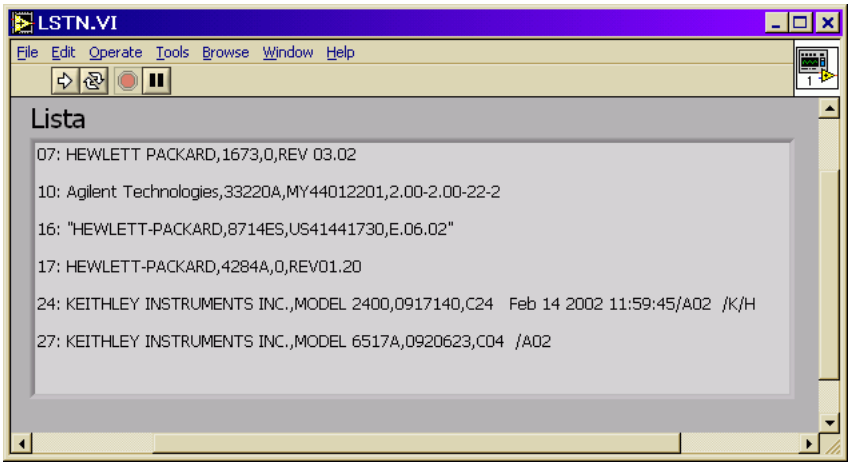


Figura 2.17 - Programa LSTN em funcionamento.

2.2 MEDIDAS EM RF

Ao realizar medidas é importante levar em consideração que os cabos, conectores, adaptadores e outros acessórios não são ideais. Problemas como atenuação, ruído, alteração dos parâmetros elétricos com a variação da temperatura devem ser levados em consideração para garantir a precisão de um processo de medida. Por isso muitos instrumentos apresentam recursos de calibração que permitem minimizar estes efeitos. Os processos de calibração são particulares aos instrumentos e dependem das condições de realização. No entanto existem conceitos que são importantes para qualquer processo de medida. Por isso o estudo a seguir é necessário. Os processos de calibração específicos dos instrumentos usados nas medidas descritas nos capítulos seguintes serão comentados durante a descrição do procedimento.

2.2.1 Os erros de medidas

Os erros gerados nas medidas podem ser classificados em três tipos: aleatórios, desvios e sistemáticos.

Os erros aleatórios como ruído ou a não repetibilidade dos conectores são erros não repetíveis e não podem ser corrigidos por calibração. Os erros de desvio, como a alteração de parâmetros elétricos com a temperatura, podem ocorrer após a calibração. Para serem eliminados, é preciso uma nova calibração.

Os erros sistemáticos, como o de rastreio e descasamento, são os mais significantes em se tratando de RF. Felizmente, são repetíveis e podem ser eliminados por calibração, embora pequenos erros residuais possam permanecer. Em geral erros sistemáticos são corrigíveis.

2.2.2 Erros repetíveis

Erros sistemáticos repetíveis ocorrem devido à resposta em frequência dos receptores, interferência entre os caminhos do sinal e descasamento nos conectores de teste.

Vamos tomar como exemplo um analisador de redes que mede os parâmetros de espalhamento. Na figura 2.18 [14] podemos ver que uma fonte de sinal excita o DUT (*Device Under Test*) através da porta A. Receptores nas portas A e B medem as potências transmitidas e refletidas.

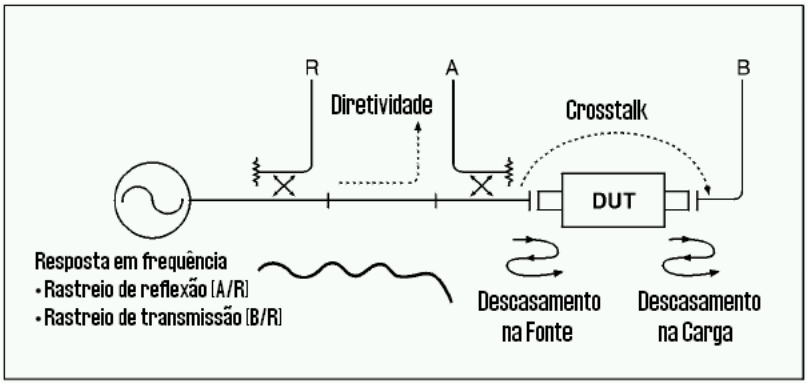


Figura 2.18 - Medidas de erros sistemáticos.

Erros de Rastreio. São causados pelas diferentes respostas em frequência dos receptores usados em uma faixa de medidas. Ao acompanhar a frequência do gerador que varia dentro da faixa estabelecida, o receptor não apresenta uma resposta constante.

Erros de Interferência. Ocorrem devido a fugas nos caminhos do sinal. Em medidas de transmissão essa interferência ocorre devido ao cruzamento entre as portas de teste. Em medidas de reflexão ocorrem devido à imperfeita diretividade dos separadores internos de sinal.

Erros de Descasamento. Ocorrem devido ao fato de que a impedância das portas do instrumento não é exatamente igual à impedância do sistema em toda a faixa de frequência. Se o DUT não estiver conectado diretamente, deve-se considerar o descasamento nos cabos. A figura 2.19 [14] mostra o seu comportamento.

Figura 2.19 - Erros de descasamento.

2.3 PARÂMETROS DE ESPALHAMENTO

2.3.1 Redes de duas portas

Embora uma rede possa ter muitas portas, os parâmetros de redes podem ser explicados, com muito mais facilidade, considerando uma rede com apenas duas portas, como mostra a figura 2.20 [15]. Para caracterizar a performance de uma rede como esta, qualquer um dos parâmetros disponíveis pode ser usado, cada um com as suas vantagens. Cada parâmetro está relacionado com um conjunto de quatro variáveis associadas com o modelo de duas portas. Duas destas variáveis representam a excitação da rede (variáveis independentes), e as outras duas restantes representam a resposta da rede à excitação (variáveis dependentes).

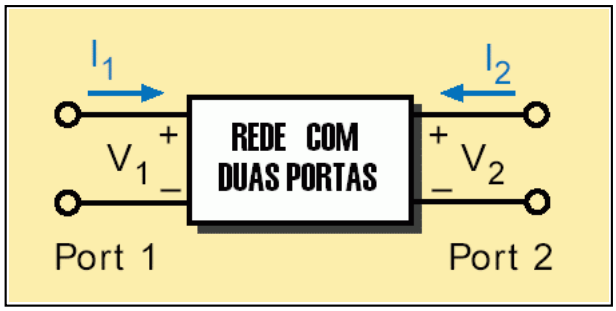


Figura 2.20 - Rede com duas portas

Supondo que a rede se comporta linearmente, se a rede da figura 2.20 for excitada pelas fontes de tensão V_1 e V_2, as correntes I_1 e I_2 da rede serão obtidas pelas seguintes equações:

$$I_1 = y_{11}V_1 + y_{12}V_2 \tag{2.1}$$

$$I_2 = y_{21}V_1 + y_{22}V_2 \tag{2.2}$$

Nesse caso com a tensão das portas selecionadas como variáveis independentes e as correntes das portas selecionadas como variáveis dependentes, os parâmetros relacionados são chamados de Parâmetros de Admitâncias de Curto Circuito, ou Parâmetros Y. Na ausência de informação adicional quatro medidas são requeridas para determinar os quatros parâmetros y_{11}, y_{12}, y_{21} e y_{22}. Cada medida é feita com uma porta da rede excitada por uma fonte de corrente enquanto a

outra porta é curto-circuitada. Por exemplo, y_{21}, a adimitância transmitida direta, é a relação entre corrente na porta 2 e a tensão na porta 1 com a porta 2 em curto, como mostrado na equação 2.3.

$$y_{21} = \left.\frac{I_2}{V_1}\right|_{V_2=0 \ \text{(SAÍDA EM CURTO)}} \tag{2.3}$$

Se outras variáveis independentes e dependentes fossem usadas a rede seria descrita, como antes, por duas equações lineares parecidas com as equações 2.1 e 2.2, exceto que as variáveis e os parâmetros descrevendo suas relações seriam diferentes. Todavia, todos os conjuntos de parâmetros contêm a mesma informação sobre a rede e é sempre possível calcular qualquer conjunto em termos de outro conjunto.

Parâmetros de espalhamento [15], comumente chamados de Parâmetros S, estão relacionados com as ondas em propagação que são dissipadas ou refletidas quando uma rede de n portas é inserida em uma linha de transmissão.

A facilidade com que os parâmetros de espalhamento podem ser medidos faz com que eles sirvam especialmente para descrever transistores e outros dispositivos ativos. Medir a maioria dos outros parâmetros exige que se coloque as entradas e saídas sucessivamente em curto ou em aberto. Isso pode ser duro de se fazer, especialmente em se tratando de RF onde indutâncias e capacitâncias parasitas dificultam a obtenção do curto e do aberto. Em altas frequências essas medidas requerem [tuning stubs], ajustados separadamente para cada frequência para refletir o curto e o aberto para o dispositivo. Isso não é apenas inconveniente e tedioso, mais [a tuning stub, shunting] a entrada ou a saída, pode fazer o transistor oscilar, tornando a medida inválida.

Parâmetros S, por outro lado, são medidos com o dispositivo inserido entre uma fonte e uma carga de 50 Ω, e a chance de ocorrer oscilações é muito pequena.

Outra vantagem dos parâmetros S consiste no fato de que as ondas propagantes, diferentemente das tensões e correntes, não variam em magnitude nos pontos ao longo de uma linha de transmissão sem perdas. Isto significa que os parâmetros de espalhamento podem ser medidos em um dispositivo localizado a certa distância dos transdutores de medidas, desde que, o DUT e os transdutores estejam conectados por linhas de transmissão de baixas perdas.

2.3.2 Definição

Os parâmetros S descrevem a inter-relação de um novo conjunto de variáveis: a_i e b_i. Estas variáveis são ondas de tensão complexas normalizadas incidentes na, ou refletidas da enésima porta da rede. Elas são definidas em termos da tensão V_i, da corrente I_i e uma referência arbitrária Z_i, onde o asterisco denota o conjugado.

$$\mathbf{a_i} = \frac{\mathbf{V_i} + \mathbf{Z_i I_i}}{2\sqrt{|\mathbf{Re}\ \mathbf{Z_i}|}} \tag{2.4}$$

$$\mathbf{b_i} = \frac{\mathbf{V_i} - \mathbf{Z_i^* I_i}}{2\sqrt{|\mathbf{Re}\ \mathbf{Z_i}|}} \tag{2.5}$$

Para a maioria das medidas e cálculos, é conveniente assumir que Z_i é real e positiva. Vamos considerar então, que todos estes parâmetros estão relacionados a uma impedância singular, positiva e real: Z_0. As funções de ondas usadas para definir os parâmetros S para uma rede de duas portas são mostradas na figura 2.21 [15].

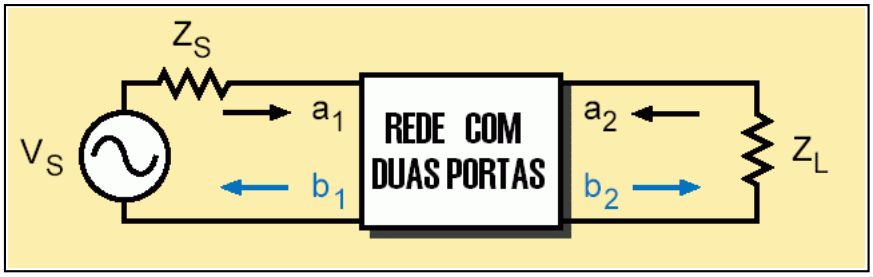


Figura 2.21 - Ondas incidentes (a_1,a_2) e refletidas (b_1,b_2)

As variáveis independentes a_1 e a_2 são tensões normalizadas incidentes.

$$\mathbf{a_1} = \frac{\mathbf{V_1 + I_1 Z_0}}{\mathbf{2\sqrt{Z_0}}} = \frac{\textbf{Onda de tensão incidente na porta 1}}{\mathbf{\sqrt{Z_0}}} = \frac{\mathbf{V_{i1}}}{\mathbf{\sqrt{Z_0}}} \tag{2.6}$$

$$\mathbf{a_2} = \frac{\mathbf{V_2 + I_2 Z_0}}{\mathbf{2\sqrt{Z_0}}} = \frac{\textbf{Onda de tensão incidente na porta 2}}{\mathbf{\sqrt{Z_0}}} = \frac{\mathbf{V_{i2}}}{\mathbf{\sqrt{Z_0}}} \tag{2.7}$$

As variáveis dependentes b_1 e b_2 são tensões normalizadas refletidas.

$$\mathbf{b_1} = \frac{\mathbf{V_1 - I_1 Z_0}}{\mathbf{2\sqrt{Z_0}}} = \frac{\textbf{Onda de tensão refletida da porta 1}}{\mathbf{\sqrt{Z_0}}} = \frac{\mathbf{V_{r1}}}{\mathbf{\sqrt{Z_0}}} \tag{2.8}$$

$$\mathbf{b_2} = \frac{\mathbf{V_2 - I_2 Z_0}}{\mathbf{2\sqrt{Z_0}}} = \frac{\textbf{Onda de tensão refletida da porta 2}}{\mathbf{\sqrt{Z_0}}} = \frac{\mathbf{V_{r2}}}{\mathbf{\sqrt{Z_0}}} \tag{2.9}$$

As equações lineares que descrevem a rede de duas portas são:

$$\mathbf{b_1} = \mathbf{S_{11} a_1 + S_{12} a_2} \tag{2.10}$$

$$\mathbf{b_2} = \mathbf{S_{21} a_1 + S_{22} a_2} \tag{2.11}$$

Os parâmetros S_{11}, S_{22}, S_{21} e S_{12} são definidos como:

S_{11}, coeficiente de reflexão na entrada com a saída casada (Z_L=Z_0 torna a_2=0).

$$\mathbf{S_{11}} = \left.\frac{\mathbf{b_1}}{\mathbf{a_1}}\right|_{\mathbf{a_2=0}} \tag{2.12}$$

S_{22}, coeficiente de reflexão na saída com a entrada casada (Z_S=Z_0 torna V_S=0).

$$\mathbf{S_{22}} = \left.\frac{\mathbf{b_2}}{\mathbf{a_2}}\right|_{\mathbf{a_1=0}} \tag{2.13}$$

S_{21}, ganho de transmissão direta com a saída casada.

$$\mathbf{S_{21}} = \left.\frac{\mathbf{b_2}}{\mathbf{a_1}}\right|_{\mathbf{a_2=0}} \tag{2.14}$$

S_{12}, ganho de transmissão reversa com a entrada casada.

$$\mathbf{S_{12}} = \left.\frac{\mathbf{b_1}}{\mathbf{a_2}}\right|_{\mathbf{a_1=0}} \tag{2.15}$$

2.3.3 Relação com a impedância

Sendo Z_1 a impedância de entrada na porta 1, é importante notar que:

$$S_{11} = \frac{b_1}{a_1} = \frac{(V_1/I_1) - Z_0}{(V_1/I_1) + Z_0} = \frac{Z_1 - Z_0}{Z_1 + Z_0} \tag{2.16}$$

$$Z_1 = Z_0 \frac{1 + S_{11}}{1 - S_{11}} \tag{2.17}$$

Nas seções 2.3.1 a 2.3.3 foi apresentada uma breve descrição dos parâmetros S, necessária para os fins desta dissertação. As equações apresentadas foram obtidas do texto "Agilent Test & Measurement Application Note 95-1 S-Parameter Techniques" [15].

2.4 O TRANSISTOR MOS-FET

No transistor de efeito de campo, conhecido como FET (*Field Effect Transistor*), o sinal de saída é controlado por uma tensão. Existem dois tipos principais de transistores de efeito de campo: o de **junção** e o de **porta isolada**. Neste trabalho vamos estudar o de porta isolada, que são constitui a base de nosso SoC.

Nestes dispositivos, o terminal metálico da porta de controle é isolado do semicondutor por uma camada isolante. Quando se aplica uma tensão na porta, o campo elétrico formado atrai os portadores minoritários no semicondutor para próximo do isolante, formando um canal condutor. Na maioria dos casos, esse isolante é um óxido do próprio semicondutor, como o SiO_2 no caso do silício [9]. Por isso o transistor é chamado de **MOS-FET** (*Metal Oxide Semiconductor - Field Effect Transistor*) [16],[17]. Uma das maiores vantagens desses dispositivos é a alta impedância de entrada, já que a tensão de controle é aplicada através de um isolante.

A figura 2.22 mostra os dois tipos de MOS-FETs: o de canal P e o de canal N. O transistor canal P é construído em um substrato tipo N. O dreno e a fonte são construídos com material tipo N difundido. Ao se aplicar uma tensão negativa na porta, as lacunas, que são minoritárias, são atraídas criando um canal condutor. O transistor canal N é construído no substrato P com fonte e dreno feitos de material tipo P. O princípio de funcionamento é o mesmo, porém com polaridade invertida.

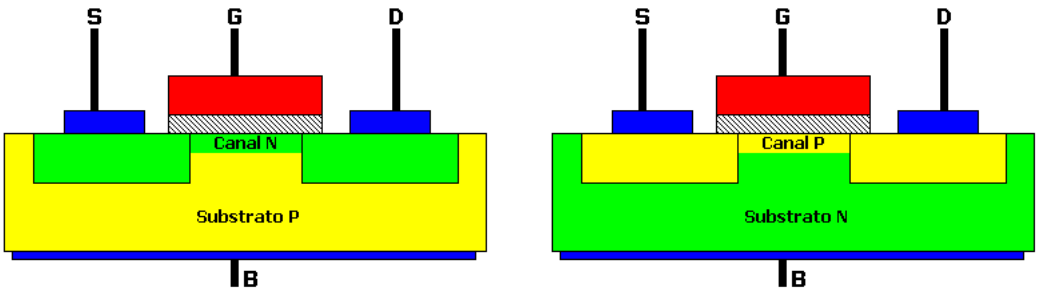


Figura 2.22 - Transistores MOS canal N e canal P.

2.4.1 O Capacitor MOS

A figura 2.23 mostra a estrutura de um capacitor de placas planas e paralelas que é base do transistor MOS [18],[19]. Nesta estrutura, a superfície condutora superior (placa) é feita de silício policristalino. O dielétrico é feito de óxido de silício ultra puro tendo espessura t_{OX}, a qual atualmente varia entre 10 nm e 100 nm, dependendo do processo. A placa inferior é feita de silício dopado adequadamente de forma a se comportar como condutor.

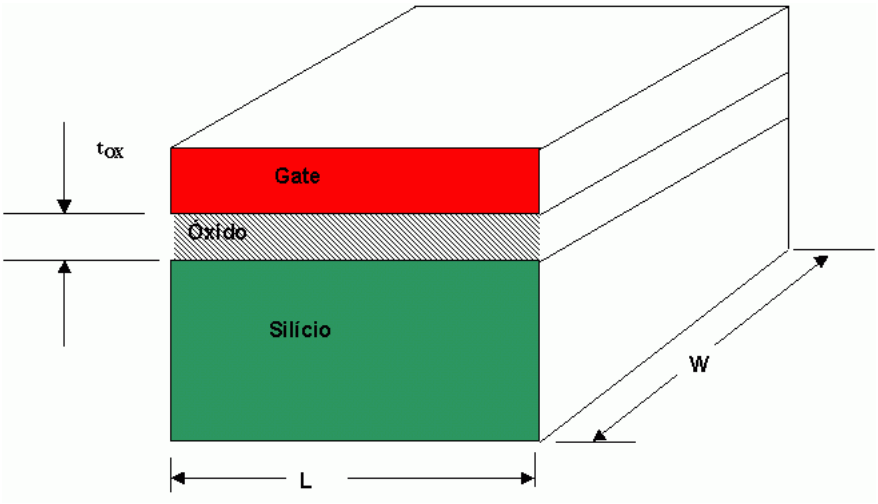


Figura 2.23 - Estrutura do capacitor MOS

A capacitância de um capacitor de placas paralelas é dada por:

$$\mathbf{C = \frac{\varepsilon_{ox}.W.L}{t_{ox}}} \quad , \tag{2.18}$$

onde ε_{OX} é a permissividade do óxido cujo valor é $3,9\varepsilon_0$;

ε_0 é a permissividade do vácuo ($\varepsilon_0 = 8,854 \times 10^{-14}$ F/cm);

W é a largura, e L o comprimento das placas; e

t_{OX} é a espessura da camada de óxido de silício.

Em geral o que se especifica é a capacitância por unidade de área:

$$\mathbf{C_{ox} = \frac{\varepsilon_{ox}}{t_{ox}} \quad [F/cm^2]} \tag{2.19}$$

2.4.2 Estrutura do Transistor MOS

A seguir mostraremos de forma simplificada a estrutura do transistor NMOS tipo enriquecimento. Para o transistor PMOS serão invertidas as correntes e tensões bem como as polaridades dos materiais. A simbologia empregada para estes componentes é mostrada na figura 2.24.

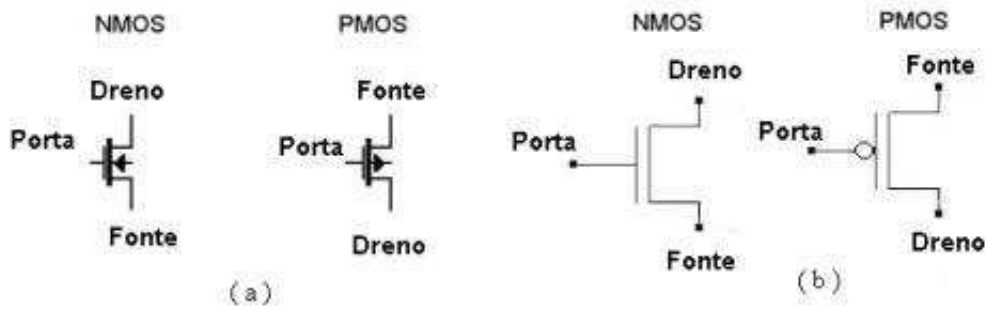


Figura 2.24 - Símbolos dos transistores MOS

A estrutura do transistor é a mesma do capacitor MOS, com a adição de regiões laterais de silício tipo *n*+ altamente dopadas onde serão colocados os terminais de dreno **D** e fonte **S** (*Source*). A porta **G** (*Gate*) é de silício policristalino, sendo o terceiro terminal. Pode existir também um quarto terminal externo ligado ao corpo **B** (*Bulk*), isto é, ao substrato tipo P (no caso de transistor NMOS). Observe as figuras 2.25 e 2.26.

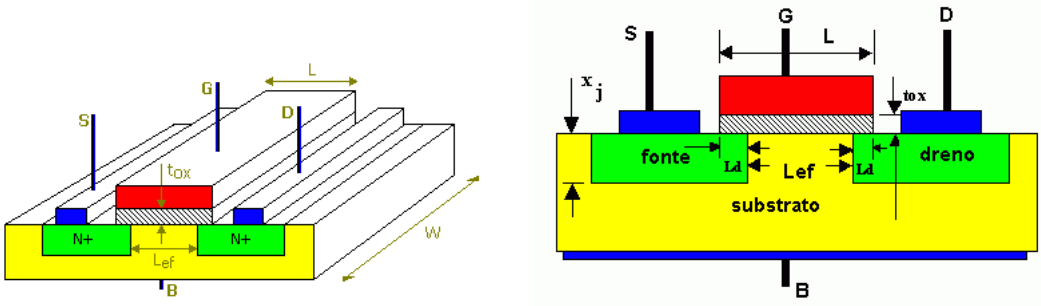


Figura 2.25 - Transistor MOS.

Figura 2.26 - Materiais Usados.

Nota-se que o comprimento efetivo do canal L_{ef} é menor do que o comprimento da porta *L*. Isso ocorre devido à difusão da região N+ para baixo da porta durante a fabricação da região de difusão. Caso a difusão lateral Ld seja desprezível, então podemos considerar $L = L_{ef}$.

Os mais importantes parâmetros geométricos de um MOS são a espessura do óxido de porta t_{ox}, o comprimento do canal, L, e a largura do canal, W. A espessura do óxido é uma característica do processo, e o projetista do circuito não tem controle sobre ele. O comprimento e a largura do canal, por outro lado, são determinados pelo projetista, sendo variáveis primárias no desenvolvimento de um projeto.

2.4.3 Funcionamento de um Transistor NMOS

A seguir, mostraremos de forma simplificada a operação do transistor NMOS. Para o PMOS basta inverter as polaridades das tensões e os sentidos das correntes.

2.4.3.1 Sem Tensão de Porta

Quando a tensão na porta é igual a zero, o dispositivo se comportará como dois diodos em série e em oposição, como mostra a figura 2.27, resultando numa resistência extremamente elevada entre dreno e fonte, da ordem de $10^{12}\Omega$.

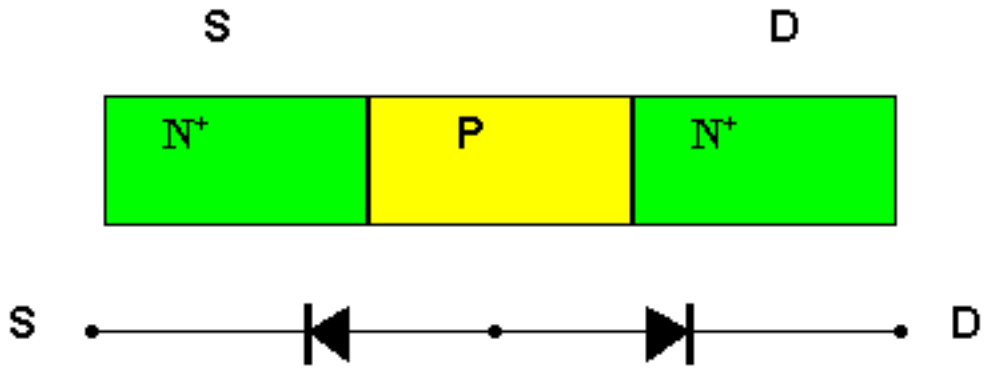


Figura 2.27 - Transistor NMOS. Fonte e dreno sem tensão de porta

2.4.3.2 Aplicando uma tensão na porta

Com a aplicação de uma tensão positiva na porta, os portadores minoritários (elétrons) do substrato são atraídos para a região abaixo do óxido de porta e

as lacunas livres do substrato são empurradas para baixo. Na região de silício abaixo da porta, quando a densidade de cargas livres negativas for maior do que a de positivas, será induzido um canal condutor ligando a região da fonte com a região do dreno. O valor da tensão de porta para o qual resulta essa condição é chamado de tensão de limiar V_T (*Threshold Voltage*)[20],[21].

Quanto maior for o valor de V_{GS}, acima de V_T, maior será a indução de cargas negativas no canal e portanto maior a condutividade do canal, isto é, a condutividade do canal será proporcional a $V_{GS}-V_T$.

Se a tensão entre dreno e fonte for $V_{DS}=0$ não existirá corrente de dreno ($I_D=0$). Observa-se na figura 2.28 a formação de uma região de depleção devido à existência de junções PN (substrato para fonte, substrato para dreno e do canal formado para o substrato).

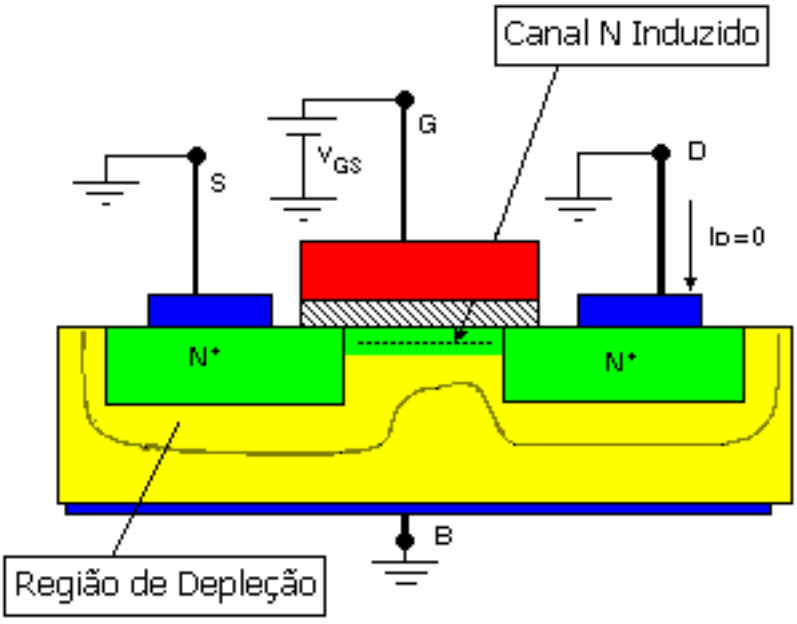


Figura 2.28 - Transistor NMOS tipo enriquecimento

$V_{GS}>V_T$ e $V_{DS}=0$

Se ao mesmo transistor for aplicado uma pequena tensão entre dreno e fonte ($0<V_{DS}<0,1$ V) aparecerá uma corrente entre dreno e fonte.

Essa corrente será proporcional à tensão V_{DS}, ou seja:

$$I_D = K \times V_{DS} \quad , \tag{2.20}$$

onde a constante de proporcionalidade dependerá do próprio transistor (dimensões) e da tensão V_{GS}. Nessa região de operação o transistor se comportará como uma resistência de valor constante:

$$R = \frac{1}{K} \tag{2.21}$$

A figura 2.29 mostra o canal formado entre dreno e fonte.

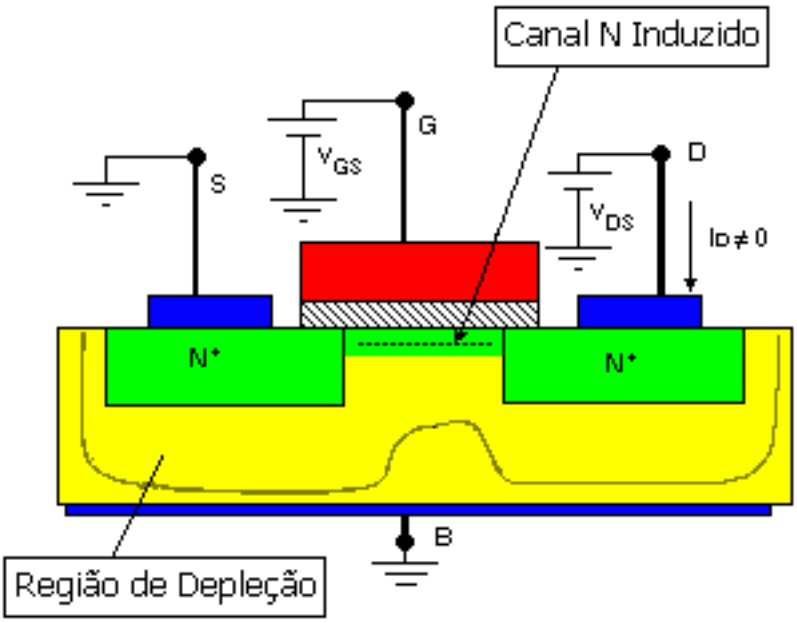


Figura 2.29 - Transistor NMOS tipo enriquecimento
$V_{GS}>V_T$ e V_{DS}=0,1 V.

Se agora V_{DS} aumentar, a corrente de dreno aumentará, mas a extremidade do canal próxima ao dreno começa a ficar mais estreita, pois a tensão entre porta e o canal na extremidade próxima ao dreno fica menor. Para $V_{DS}=V_{GS}-V_T$ o canal fechará totalmente próximo ao dreno como pode ser visto na figura 2.30. A esse valor se dá o nome de V_{DSsat}.

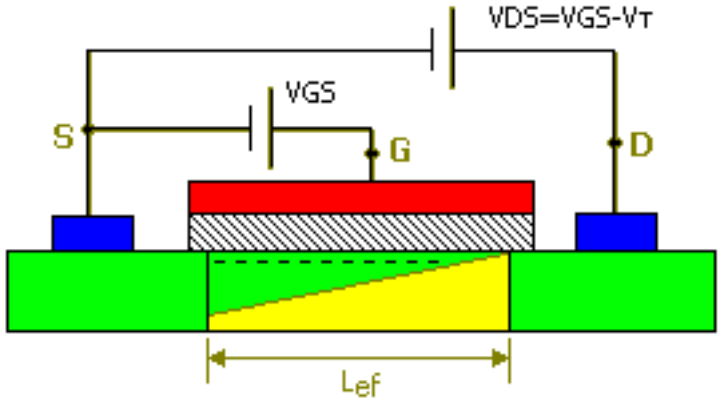


Figura 2.30 - Estreitamento máximo do canal próximo ao dreno

$V_{DS}=V_{GS}-V_T=V_{DSsat}$

Aumentando a tensão de dreno além do valor de saturação, como mostrado na figura 2.31, o estrangulamento do canal se prolongará na direção da fonte. Com a resistência do canal tornando-se muito alta, o dispositivo passa a ter comportamento de uma fonte de corrente I_D começa a ficar constante). Apesar do canal, na região próxima do dreno, deixar de existir, a corrente de dreno não se anula, ao invés disso permanece aproximadamente constante. A explicação para esse fato é que os elétrons terão energia suficiente para pularem do canal para o dreno.

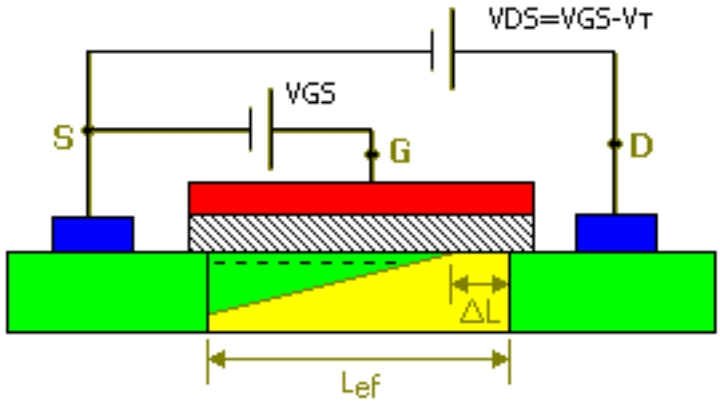


Figura 2.31 - Estrutura MOS com estreitamento máximo do canal próximo ao dreno e diminuição do canal

$V_{DS}>V_{GS}-V_T$.

2.4.4 Regiões de Operação do Transistor NMOS

Um transistor NMOS pode ter seu comportamento caracterizado pelas regiões de operação mencionadas anteriormente, em função das tensões aplicadas. As regiões de operação são classificadas de acordo com os valores relativos das tensões em:

2.4.4.1 Região de Corte

Para $V_{GS}<V_T$, $I_D=0$ qualquer que seja o V_{DS}

2.4.4.2 Região Triodo

Para $V_{GS}>V_T$ e $V_{DS}<V_{GS}-V_T$ um canal será induzido e a corrente de dreno será dada por:

$$I_D = \mu_n \times C_{OX} \times \left(\frac{W}{L_{ef}}\right) \times \left((V_{GS} - V_T) \times V_{DS} - \frac{V_{DS}^2}{2}\right) \quad , \tag{2.22}$$

onde,
μ_n é a mobilidade dos elétrons livres na superfície próxima ao óxido de porta;
C_{OX} é a capacitância por unidade de área do capacitor formada pela porta e o canal (com o óxido de porta sendo o dielétrico deste capacitor);
L_{ef} é o comprimento efetivo do canal.

O comprimento efetivo do canal é dado por:

$$L_{ef} = L - 2 \times Ld \tag{2.23}$$

Ld é o valor da difusão lateral e L o comprimento da porta. Se Ld for desprezível, podemos considerar L ao invés de L_{ef}. Se V_{DS} for pequeno (por exemplo 0,1 V) de forma que o termo ($V_{DS}{}^2/2$) seja desprezível, então a equação (2.22) pode ser aproximada por:

$$I_D = \mu_n \times C_{OX} \times \left(\frac{W}{L}\right) \times (V_{GS} - V_T) \times V_{DS} \tag{2.24}$$

Para um valor de V_{GS} constante teremos IDS = K $\times$ V_{DS}, onde K é o inverso da resistência do canal na região linear, a qual é dada por R = 1 / K:

$$\mathbf{K = \mu_n \times C_{OX} \times \left(\frac{W}{L}\right) \times (V_{GS} - V_T)} \tag{2.25}$$

2.4.4.3 Região de Saturação

Na região de saturação, $\mathbf{V_{GS} > V_T}$ e $\mathbf{V_{DS} = V_{DS}Sat > V_{GS} - V_T}$, o transistor se comportará como uma fonte de corrente ideal cujo valor de corrente será independente de V_{DS}, e a relação entre a corrente de dreno e a tensão de porta será dada por:

$$\mathbf{I_D = \frac{\mu_n \times C_{OX}}{2} \times \left(\frac{W}{L}\right) \times (V_{GS} - V_T)^2} \tag{2.26}$$

Para essa condição, a resistência incremental de saída, r_O, seria infinita. r_O é definida como:

$$\mathbf{r_O = \frac{\Delta V_{DS}}{\Delta I_{DS}}} \tag{2.27}$$

Na realidade, existe uma pequena dependência da corrente de dreno em função da tensão de dreno quando $V_{DS} > V_{DSsat}$, que é muito importante para circuitos analógicos. Quando V_{DS} aumenta, o comprimento efetivo do canal diminui de ΔL, como mostrado na figura 2.31. Este fenômeno é chamado de modulação do comprimento do canal, e como a corrente de dreno é proporcional a 1 / L_{ef}, a modulação do comprimento do canal tende a aumentar a corrente na saturação. Desta forma é preciso adicionar um fator de correção na equação 2.26 considerando, agora, a modulação do comprimento do canal. Assim I_D passa a ser:

$$\mathbf{I_D = \frac{\mu_n \times C_{OX}}{2} \times \left(\frac{W}{L}\right) \times (V_{GS} - V_T)^2 \times (1 + \lambda \times V_{DS})} \tag{2.28}$$

O λ é um parâmetro do transistor chamado **Fator de Modulação de Canal**.
Os efeitos desse parâmetro em circuitos digitais são desprezíveis e portanto poderemos considerar λ = 0. Mas em circuitos analógicos os seus efeitos devem ser considerados, pois o λ está relacionado com a resistência de saída, que por sua vez está relacionada com o ganho de tensão.

A resistência incremental de saída é dada por:

$$\mathbf{r_O} = \frac{\mathbf{\Delta V_{DS}}}{\mathbf{\Delta I_D}}\bigg|_{\mathbf{VGS=constante}} \tag{2.29}$$

, resultando em:

$$\mathbf{r_O} = \left[\lambda \times \frac{\mu_\mathbf{n} \times \mathbf{C_{OX}}}{\mathbf{2}} \times \left(\frac{\mathbf{W}}{\mathbf{L}}\right) \times (\mathbf{V_{GS}} - \mathbf{V_T})^\mathbf{2}\right]^{-1} \tag{2.30}$$

Nota-se que r_O seria infinita (ideal) se $\lambda = 0$. Negligenciando o efeito do fator $(1 + \lambda \times V_{DS})$ na equação (2.28) o valor de r_O pode ser dado aproximadamente por:

$$\mathbf{r_O} = [\lambda \times \mathbf{I_D}]^{-\mathbf{1}} \quad , \tag{2.31}$$

onde $\mathbf{I_D}$ é o valor da corrente de dreno para um determinado valor de $\mathbf{V_{GS}}$. A equação acima pode ser reescrita na forma alternativa:

$$\mathbf{r_O} = \frac{\mathbf{V_A}}{\mathbf{I_D}} \qquad \mathbf{V_A} = \frac{\mathbf{1}}{\lambda} \quad , \tag{2.32}$$

onde $\mathbf{V_A}$ é a tensão **Early**. A figura 2.32 mostra as curvas de dreno ($\mathbf{I_D x V_{DS}}$), para quatro valores de $\mathbf{V_{GS}}$ e as regiões de operação do transistor NMOS.

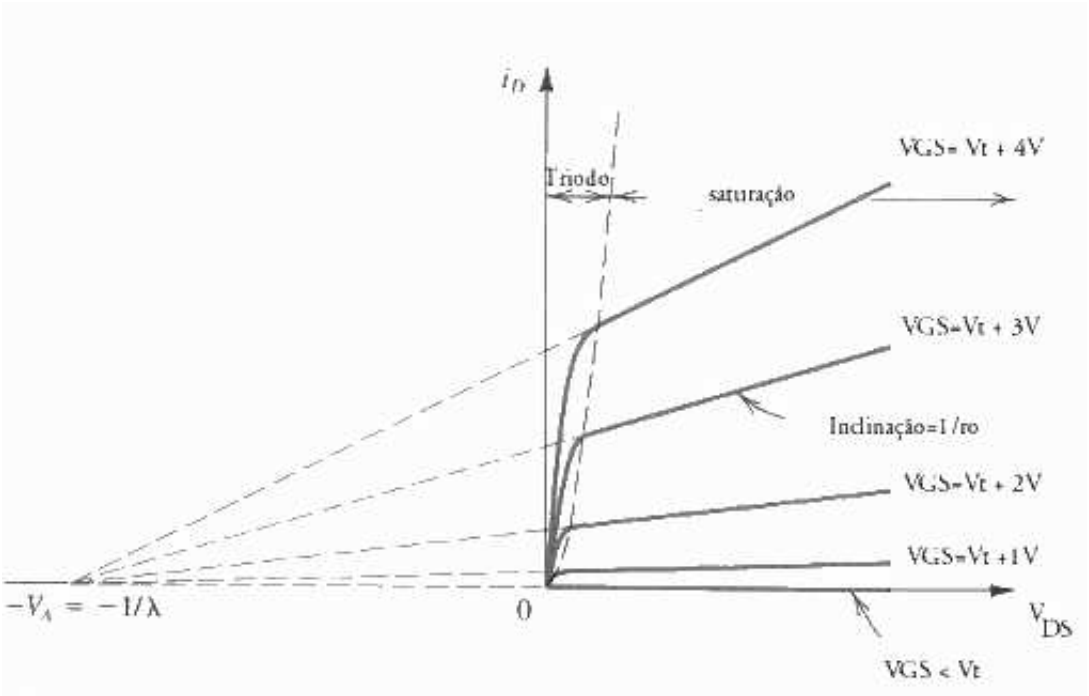


Figura 2.32 - Curvas características de dreno,
Regiões de operação e tensão Early

2.4.5 Extração de Parâmetros

Os parâmetros mais importantes, que podem ser extraídos apartir das curvas, são a Tensão de Limiar, a Transcondutância, o Fator de Ganho e a Mobilidade [20],[21].

A Tensão de Limiar V_T é a tensão aplicada na porta do transistor a partir da qual o canal está fechado e, se V_{DS} estiver polarizado adequadamente, o transistor passará a conduzir. Através da curva $I_{DS}xV_{GS}$ para V_{DS} = 0,1 V, este parâmetro é extraído da interseção da reta tangente à curva na região de maior transcondutância com o eixo das coordenadas, ou fixando-se um nível de corrente.

A Transcondutância gm é extraída diretamente do coeficiente da reta tangente à curva $I_{DS}xV_{GS}$ para V_{DS}=0,1 V:

$$\mathbf{gm} = \left.\frac{\mathbf{\Delta I_{DS}}}{\mathbf{\Delta V_{GS}}}\right|_{\mathbf{V_{DS}=0{,}1\ V}} \tag{2.33}$$

O Fator de Ganho β é o fator de amplificação da corrente que é depende da mobilidade do portador μ_n, da espessura do óxido X_{OX} e das dimensões comprimento L e largura W. Este parâmetro é extraído da transcondutância gm e da tensão V_{DS} conforme equação:

$$\beta = \mathbf{gm} \times \mathbf{V_{DS}} \tag{2.34}$$

A Mobilidade μ_{no} é a mobilidade de elétrons, independente de campo, no canal do transistor. É calculado apartir do fator de ganho pela equação:

$$\mu_{\mathbf{no}} = \frac{\beta}{\mathbf{C_{OX}}} \times \frac{\mathbf{L}}{\mathbf{W}} \tag{2.35}$$

Nas seções 2.4.1 a 2.4.5 foi apresentada uma breve descrição do funcionamento de transistores MOS, necessária para os fins desta dissertação. As equações apresentadas foram obtidas do livro texto “Microeletrônica” (Cedra/Smith) [16].

3 METODOLOGIA PARA MEDIDAS

O conteúdo deste capítulo se refere à estrutura física e ao uso de um laboratório de medidas. A primeira parte trata da montagem da estrutura de testes, incluindo as bancadas de teste, operação dos instrumentos e cuidados a serem tomados. Depois tem um modelo para agilizar o desenvolvimento de programas de medidas com LabVIEW. Inclui um programa de exemplo. Finalmente, uma estrutura de SPOOL usando o LabVIEW é apresentada. Trata-se de um recurso do protocolo GPIB que permite que os instrumentos forneçam informações sobre seus estados ao programa controlador. O programa então escolhe rotinas baseando-se nessas informações.

3.1 MONTANDO UMA ESTRUTURA DE TESTES

Em um laboratório contendo vários instrumentos, computadores e bancadas de testes é necessário organizar uma estrutura de operação. Os procedimentos devem ser anotados e os resultados dos testes devem ser armazenados nos computadores para análise posterior. Existem também cuidados na parte elétrica e no manuseio dos instrumentos e dos dispositivos a serem testados.

As estruturas e cuidados variam de acordo com o tipo de laboratório. Este trabalho trata de microeletrônica envolvendo testes em *chips* não encapsulados e medidas em RF. Os procedimentos necessários serão vistos a seguir.

3.1.1 Bancadas de Testes

Para testes em *chips* não encapsulados as bancadas devem possuir microscópio para permitir o ajuste dos posicionadores nos *pads* de testes. O dispositivo deve ficar em um suporte metálico devidamente aterrado. O suporte possui pequenos orifícios que são ligados a uma bomba de sucção para fixar o DUT. As figuras 3.1

e 3.2 mostram duas bancadas. A primeira é para testes em baixa frequência, e a segunda é para testes em RF.

É importante notar que na bancada de RF cada ponteira possui três terminais. Os dois laterais são terra e o do meio é o de sinal. Os posicionadores usam conectores APC 3,5 mm e são ligados aos instrumentos por cabos coaxiais blindados. Isso proporciona melhor propagação de sinal e maior imunidade a interferências. Os projetistas devem posicionar os *pads* dentro do padrão usado pela bancada.

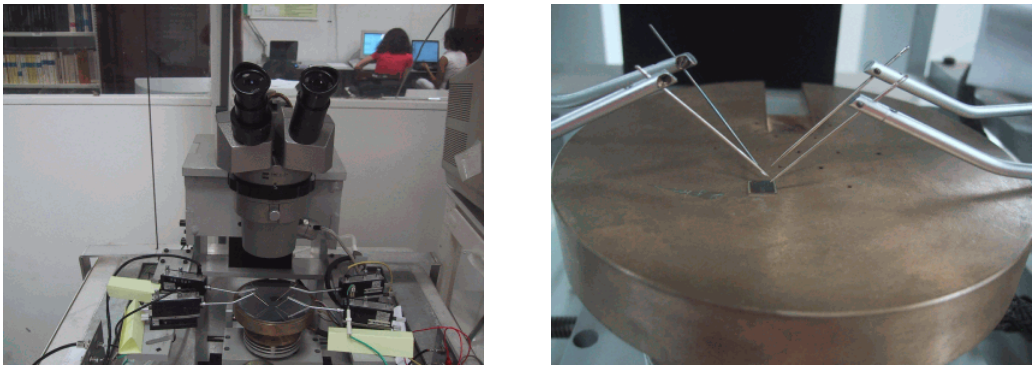


Figura 3.1 - Estação de medidas em baixa frequência

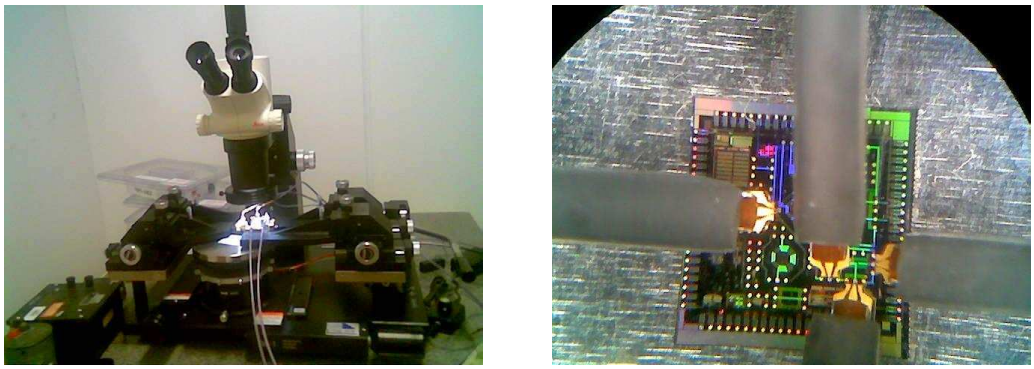


Figura 3.2 - Estação de medidas para RF

3.1.2 Operando os instrumentos

Para facilitar a operação, vários instrumentos podem ser colocados em um rack móvel de maneira que possam ser ligados facilmente às bancadas. Muitos instrumentos podem ser operados remotamente. Usamos o protocolo GPIB estudado anteriormente, pois permite que vários medidores sejam interligados e operem sincronizadamente. A figura 3.3 mostra vários instrumentos no rack.

Figura 3.3 - Instrumentos no rack

3.1.3 Cuidados a serem tomados

Para manusear circuitos integrados deve-se usar pinças adequadas e pulseiras anti-estáticas. Esses circuitos são sensíveis à cargas eletrostáticas. As bancadas devem estar devidamente aterradas.

Instrumentos como analisadores de espectro e analisadores de redes possuem entradas sensíveis com limite de potência em torno de 1 watt. É necessário ajustar devidamente as fontes de sinais e levar em conta se o DUT possui ganho. O uso de atenuadores e bloqueadores de DC pode ser necessário.

Muitos medidores ao ser ligados precisam de um tempo para se estabilizar. A não observância desse detalhe pode causar erros nos testes. Antes de medir é importante ler atentamente os manuais e observar os cuidados a serem tomados tanto para a precisão das medidas quanto para a segurança dos instrumentos e dos operadores.

3.2 MODELO DE PROGRAMAÇÃO

Será apresentado um método para facilitar a criação de aplicativos para usar instrumentos de medidas usando o protocolo GPIB e o LabVIEW.

Muitos manuais de instrumentos ainda apresentam exemplos de aplicativos feitos em linguagens obsoletas como o QBASIC que funciona no DOS. Isto torna difícil a compreensão e o processo de programação fica trabalhoso.

Este trabalho apresenta um procedimento com exemplos para se usar o LabVIEW, simplificando, ganhando tempo e aumentando a eficiência na criação de aplicativos.

3.2.1 Sistema de Medidas

Será usado como exemplo a fonte e medidor Keithley 2400 [4],[11],[12] para demonstrar os procedimentos para criar aplicativos de medidas. O instrumento pode ser visto na figura 3.4 [22]. O apêndice A apresenta mais detalhes.

Figura 3.4 - Keithley 2400

3.2.2 Os modos de operação

Esse modelo básico é seguido na maioria dos programas usados no laboratório. O programa possui dois modos de execução. O modo **Medir** e o modo **Garregar Arquivo**. No modo **Medir**, o programa controla instrumentos, realiza medidas e salva os dados em arquivo. No modo **Garregar Arquivo** o programa carrega um arquivo salvo anteriormente e mostra a curva no gráfico.

3.2.2.1 O modo Medir

O modo **Medir** é dividido em seis blocos que são executados por SubVI's:

- Iniciar: Configura o instrumento;
- Escala: define número de medidas;
- Parâmetro: define um parâmetro como tensão, corrente ou frequência;
- Ler: faz medida e captura os resultados;
- Salvar: cria um arquivo de texto e salva os dados;
- Gráfico: pega os resultados no arquivo, organiza-os e os exibe no gráfico.

Para mostrar uma série de medidas em um gráfico os programas usam uma estrutura de *loop*. Para cada ciclo é definido um parâmetro e feito uma medida. Os resultados são armazenados na memória RAM do computador. Após o último parâmetro, os resultados são salvos em um arquivo e exibidos no gráfico. A figura 3.5 mostra o processo.

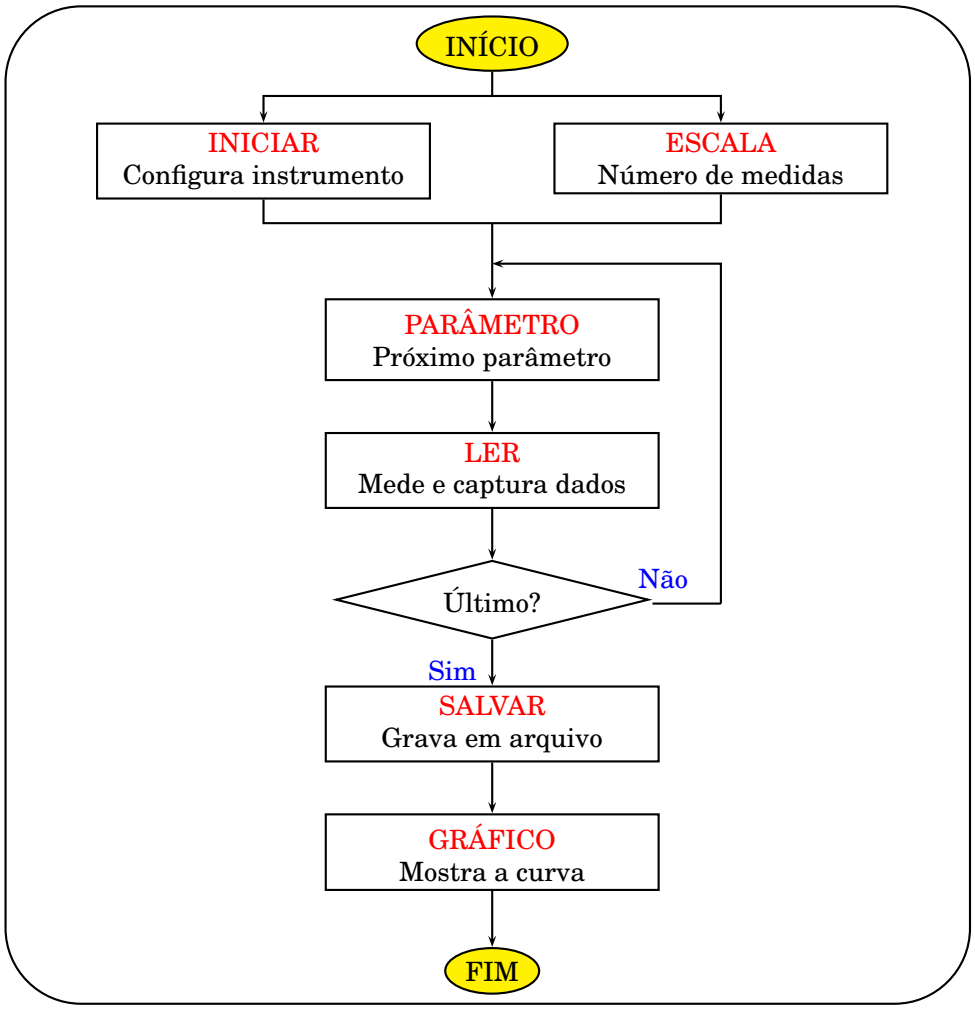


Figura 3.5 - Fluxograma para usar o LabVIEW

3.2.2.2 O modo Carregar Arquivo

Quando executado no modo **Carregar Arquivo**, o programa abre a caixa de diálogo do Windows, vista na figura 3.6. Se o usuário clicar em *Cancelar*, o gráfico do programa é zerado e o programa volta ao modo ocioso. Quando o usuário escolhe um arquivo e clica em *Abrir*, o programa usa o SubVI **GRÁFICO** para enviar os dados do arquivo escolhido ao gráfico. Os parâmetros de entrada, antes determinados pelo usuário, são recuperados e escritos nos campos. A figura 3.7 mostra o procedimento com fluxograma.

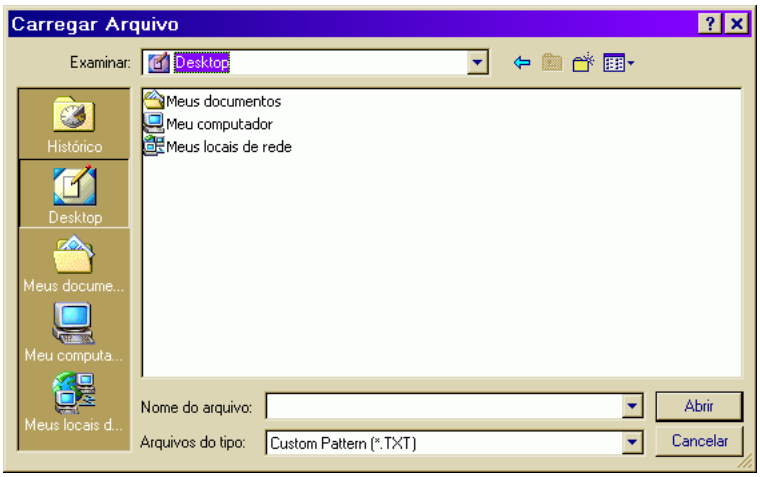


Figura 3.6 - Escolher um arquivo

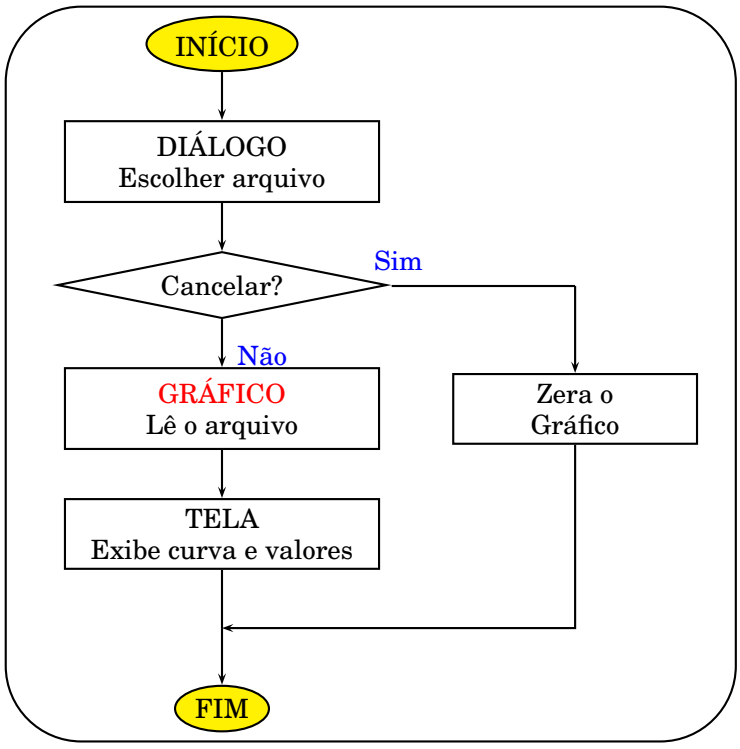


Figura 3.7 - Fluxograma do modo carregar arquivo

3.2.3 Exemplo de programa

Para exemplificar, vamos tomar como exemplo um programa que traça a curva IxV de diodos usando o Keithley 2400. No caso, o parâmetro que varia é a tensão de polarização. Para cada tensão definida, é medida a corrente resultante. O instrumento gera tensão e mede corrente através dos conectores banana fêmea *INPUT*/*OUTPUT HI* e *LO* dianteiros ou traseiros.

A figura 3.8 mostra a ligação de um DUT, neste caso o diodo, no painel frontal e o diagrama equivalente. O resultado da medição é exibido na figura 3.9.

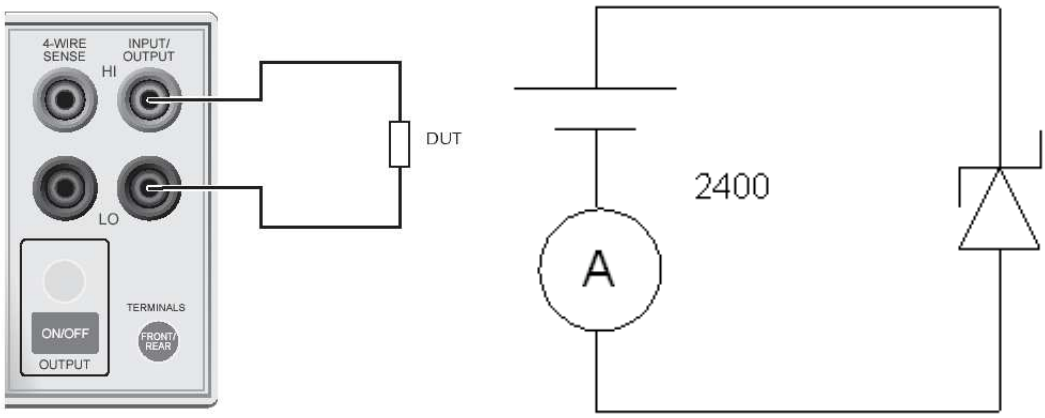


Figura 3.8 - Ligar o dispositivo no Keithley 2400.

Na figura 3.9, podemos ver o programa em funcionamento. O campo *Endereço* se refere ao endereço GPIB do instrumento, no caso com o número 24. A chave *boolean Medir-Arquivo* determina o modo de funcionamento, medir ou carregar arquivo. O teste é feito da seguinte forma: inicialmente, o instrumento aplica a tensão *V Inicial* no diodo. A tensão vai sendo incrementada no valor especificado no campo *Passo*. Para cada tensão fornecida a corrente no dispositivo é medida. O resultado é armazenado na memória do computador. Quando a tensão atinge o valor de *V Final*, o programa salva os valores das tensões e correntes no arquivo de texto com caminho e nome especificados no campo abaixo à esquerda. Em seguida mostra a curva IxV no gráfico.

Quando se faz várias medidas, os nomes dos arquivos, são salvos com um índice para diferenciá-los. No exemplo abaixo, os arquivos são salvos na área de trabalho como LED-01.TXT, LED-02.TXT, LED-03.TXT, LED-04.TXT, LED-05.TXT, LED-06.TXT, etc. O número ao lado do nome do arquivo, seis no exemplo, é o índice que indica o último arquivo que foi salvo. Pode ser alterado pelo usuário.

A tela preta com letras verdes mostra os valores de tensão e corrente que estão sendo fornecidos e medidos durante a execução. O programa retém os últimos valores após a finalização.

Quando o usuário fornece uma tensão de início maior que a final, o programa automaticamente decrementa a tensão criando uma rampla decrescente.

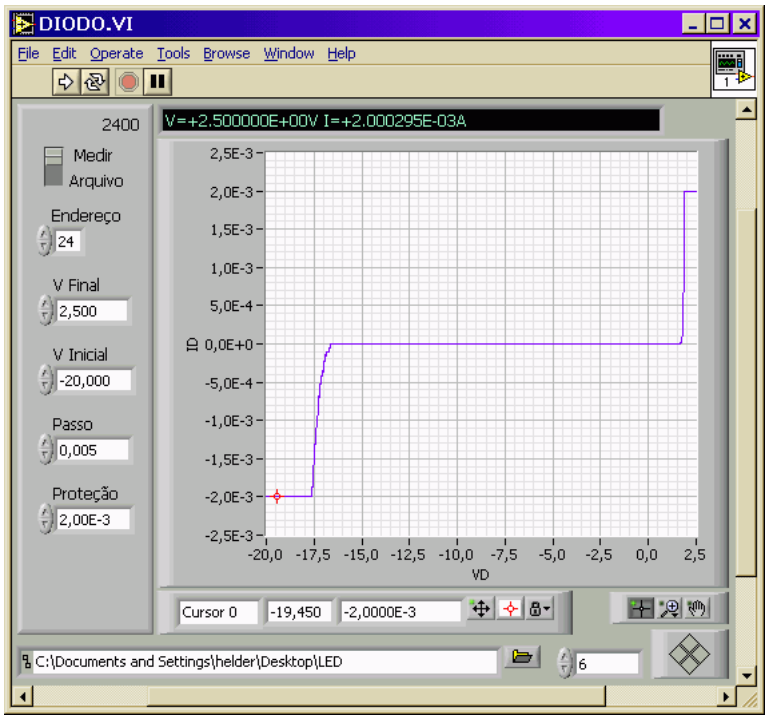


Figura 3.9 - Programa DIODO

3.2.3.1 Código no modo Medir

Na figura 3.10 podemos ver os ícones dos SubVI's identificados pelas letras vermelhas. O SubVI **VOLT** calcula o valor da tensão e envia comandos SCPI ao instrumento para que ele polarize o dispositivo.

INICIAR ESCALA VOLT LER SALVAR GRÁFICO

Figura 3.10 - SubVI's do programa DIODO

O controle *boolean Medir* (que aparece na tela do programa como chave *Medir-Arquivo*) controla a estrutura *Case* que determina o modo de funcionamento. Se o valor for *True* o código mostrado na figura 3.11 é executado. Na figura 3.12 temos o fluxograma do modo Medir. Segue a mesma lógica do modelo geral apresentado anteriormente.

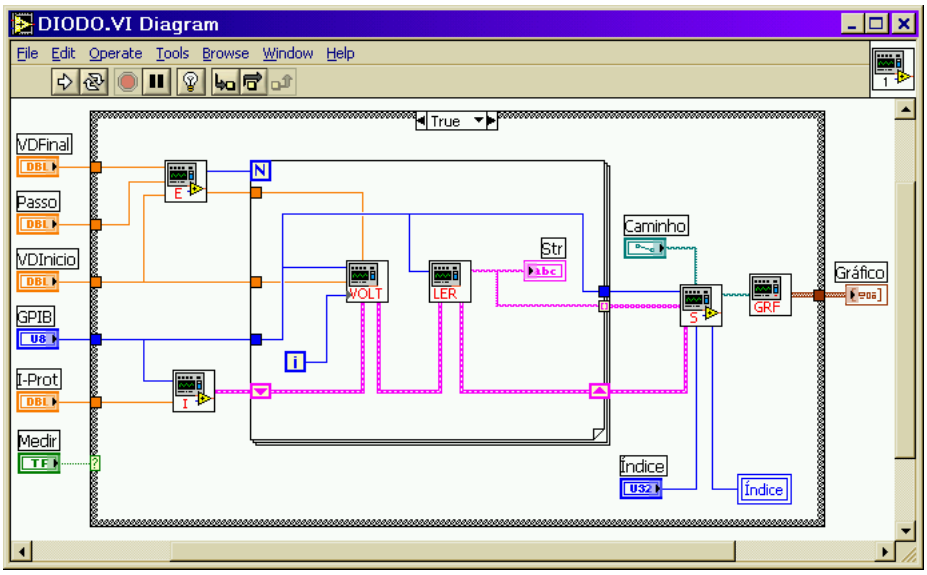


Figura 3.11 - Código do programa DIODO

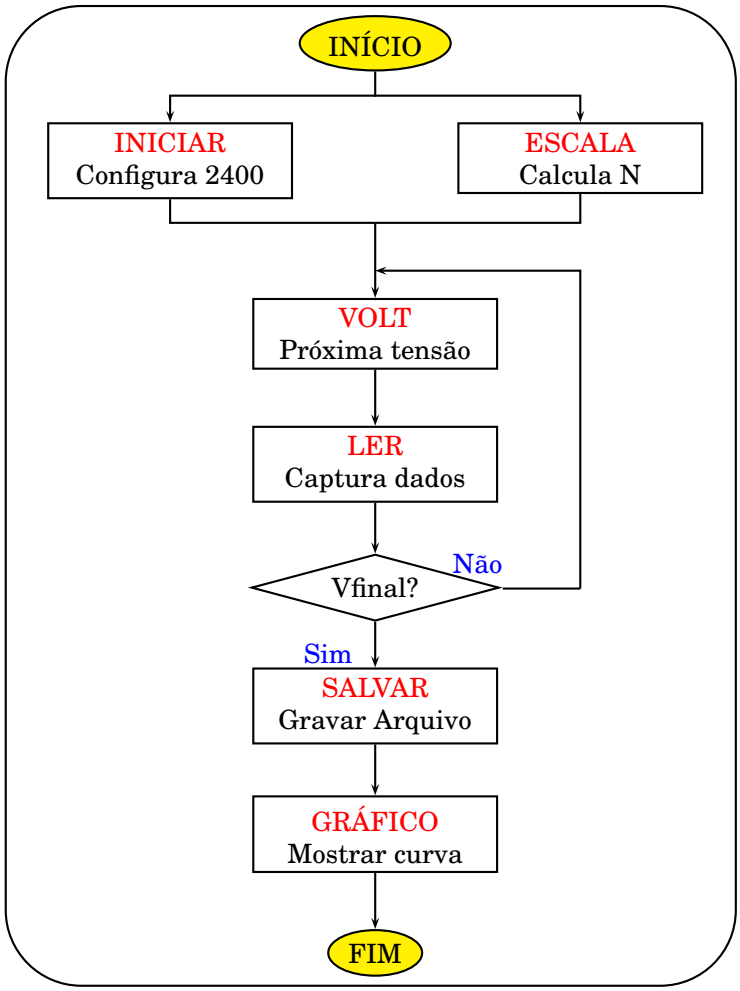


Figura 3.12 - Fluxograma do Programa DIODO

3.2.3.2 Código no modo Carregar Arquivo

Quando executado no modo Carregar Arquivo, o programa abre a caixa de diálogo do Windows para que o usuário escolha um arquivo para ser exibido.

Neste modo, a estrutura *Case* principal é posta em *False* pela chave *boolean Medir* (*Medir-Arquivo*). Podemos ver então outra estrutura *Case* ligada à saída *cancelled* da função *File Dialog*. Se o usuário cancela a ação, a saída retorna *True* colocando a estrutura na opção vista na figura 3.13. O gráfico é zerado e o programa volta ao estado inicial.

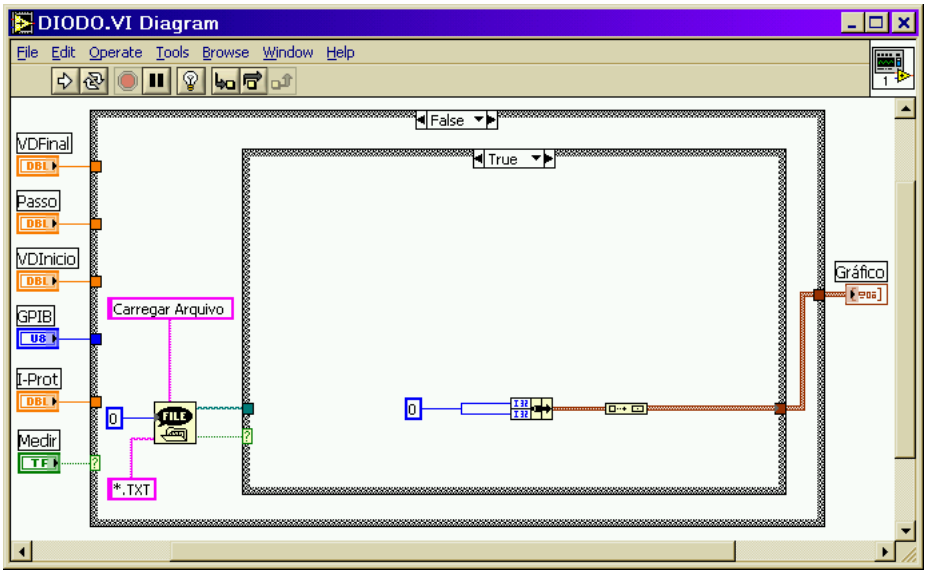


Figura 3.13 - Cancelar

Quando um arquivo é escolhido, o caminho é levado ao SubVI **GRÁFICO**, como mostra a figura 3.14. Ele envia o conteúdo do arquivo para o gráfico. O SubVI fornece também, em saídas separadas, a corrente de proteção e um *array* contendo os valores das tensões. As tensões previamente fornecidas pelo usuário são recuperadas pela função *Index Array* que fornece o valor contido no índice especificado. Os valores são escritos nos campos através das variáveis locais.

A tensão *V Final* é localizada usando as funções *Array Size* e *Decrement*. A função *Array Size* fornece o número total de elementos do *array*. Como o índice varia de "0" a "N-1", o número deve ser subtraído de um antes de chegar à função *Index Array*.

O valor de *V Inicial* se localiza no índice zero. O *Passo* é conseguido subtraindo o valor do índice zero do valor do índice um.

A figura 3.15 identifica as principais funções. Na figura 3.16, temos o fluxograma completo do modo carregar arquivo. O nome do SubVI **GRÁFICO** está destacado com letras vermelhas. Serão descritos a seguir os códigos dos SubVI's. Os outros programas usam estruturas bem semelhantes.

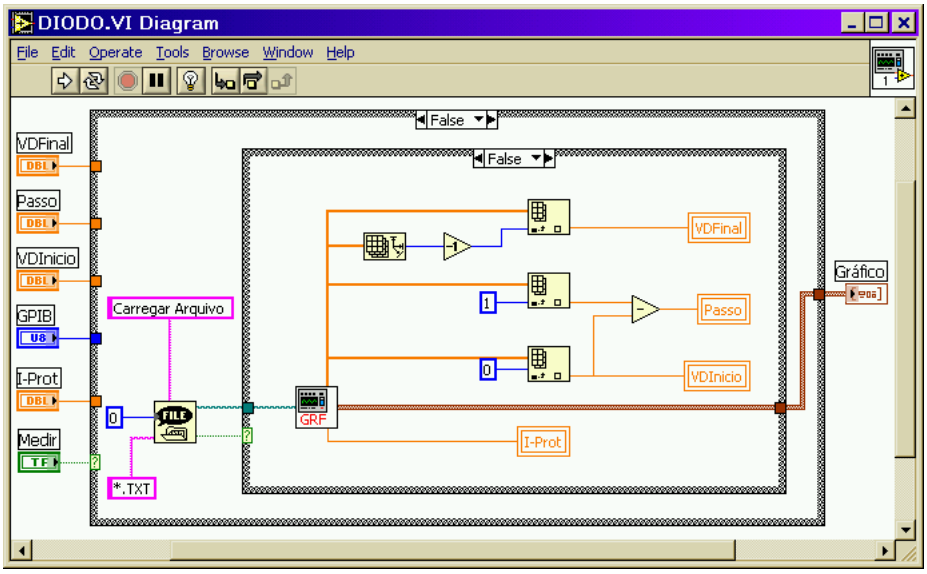


Figura 3.14 - Modo carregar arquivo

File Dialog GRÁFICO Array Size Index Array

Figura 3.15 - Principais funções do programa

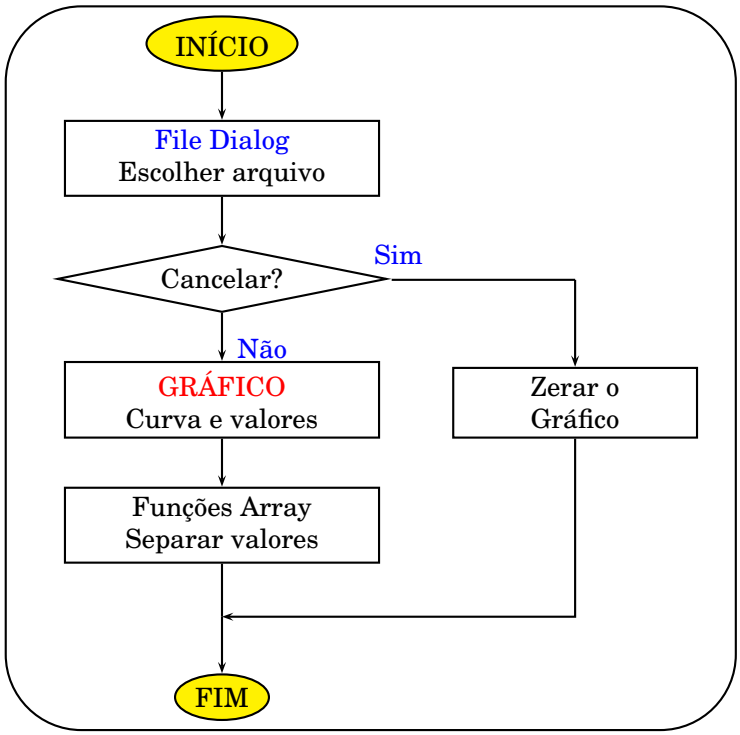


Figura 3.16 - Fluxograma do modo carregar arquivo

3.2.3.3 SubVI INICIAR

O SubVI **INICIAR** configura o instrumento fonte e medidor Keithley 2400 para gerar tensão e medir corrente enviando comandos SCPI através da interface GPIB. Na figura 3.17 temos o código feito em LabVIEW. O valor da corrente de proteção fornecido pelo usuário é transformado em *string* pela função *Format Value* e inserido nos comandos pela função *Search and Replace String*, substituindo a palavra [II]. Os comandos de configuração são enviados pela função GPIB *Send*. A entrada *GPIB* possui o endereço do instrumento.

Em seguida, a estrutura com a função de tempo faz com que o programa espere alguns segundos para que o instrumento possa processar os comandos. Finalmente, o comando ":OUTPUT ON" é enviado para que o Keithley 2400 ative a sua saída. A tabela 3.1 apresenta as descrições dos comandos. A figura 3.18 identifica as funções usadas. A figura 3.19 mostra o fluxograma.

*RST	Reset - Coloca na configuração padrão.
:SOUR:FUNC VOLT	Configura para fornecer tensão.
:SOUR:VOLT:MODE FIX	Fonte de tensão no modo fixo.
:SOUR:VOLT:RANG 210	Escala de 210 V
:SOUR:DEL 0.02	Intervalo entre alimentar e medir
:SOUR:VOLT 0	Zerar - Fonte inicia em 0 V
:SENS:FUNC "CURR"	Configura para medir corrente.
:SENS:CURR:RANG 1.05E-6	Ajusta a escala de corrente
:SENS:CURR:RANG:AUTO ON	Escala de corrente no modo automático.
:SENS:CURR:PROT [II]	Define a corrente de proteção.
:OUTPUT ON	Ativa a saída

Tabela 3.1 - Configuração do instrumento Keithley 2400

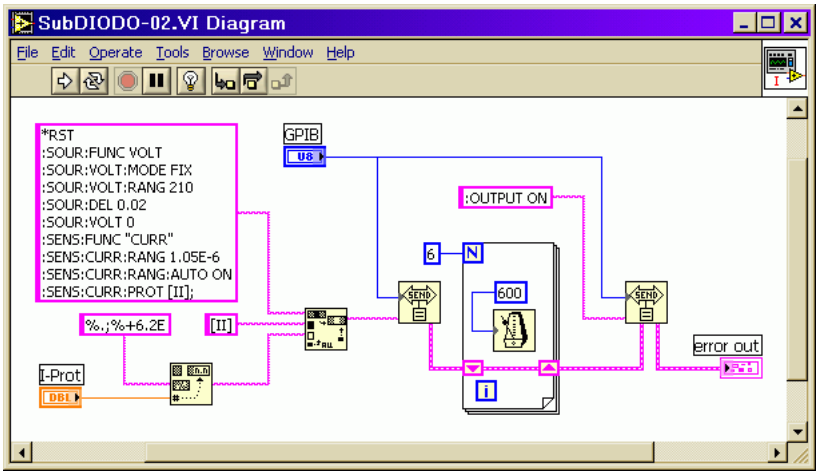


Figura 3.17 - SubVI **INICIAR**

Figura 3.18 - As funções do SubVI

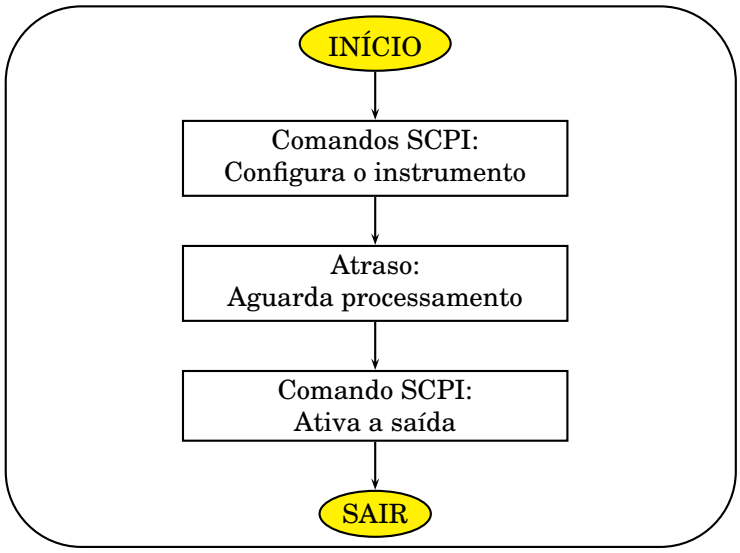


Figura 3.19 - Fluxograma do SubVI **INICIAR**

3.2.3.4 SubVI ESCALA

O SubVI **ESCALA** calcula o número de ciclos "N" para a estrutura *For Loop*. Também determina o sinal do passo, positivo para tensão crescente e negativo para tensão decrescente. O número de execuções pode ser calculado seguindo a lógica:

$$N = (VFinal - VInicial) \div Passo$$

Entretanto, algumas operações foram acrescentadas para resolver problemas de arredondamento e conversão. A estrutura trabalha apenas com números inteiros e positivos. O valor absoluto da diferença entre as tensões de início e final e o valor absoluto do passo são multiplicados por mil para garantir uma precisão de três casas decimais. Após dividir os valores, é somado "0,6" para garantir que o arredondamento será sempre para cima. O resultado é arredondado para inteiro e entregue à saída *N*.

O sinal do passo é obtido com a função *Sign*. Quando *VFinal* é menor que *VInício* a função fornece "-1"; caso contrário, resulta em "+1". O valor é multiplicado pelo valor absoluto da entrada *Passo* e entregue à saída *Passo+*. O código é apresentado na figura 3.20.

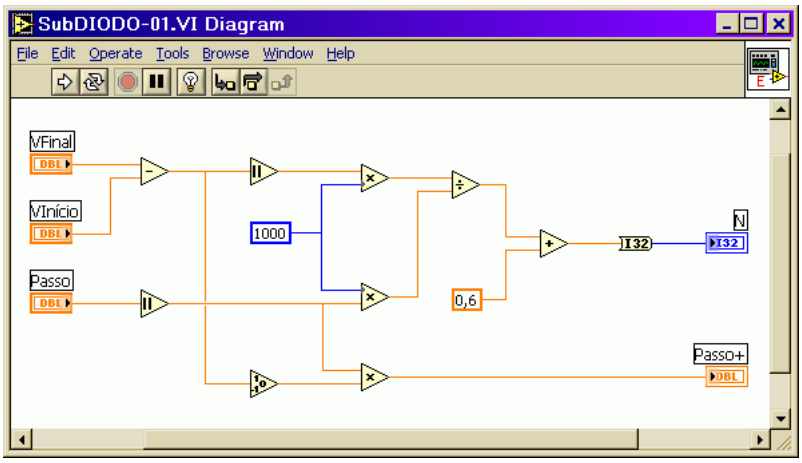


Figura 3.20 - SubVI **ESCALA**

Se o usuário forneçesse os valores: *VInício = 1 V*; *VFinal = 4 V*; e *Passo = 0,6 V*, as tensões geradas seriam:

1,0 V : 1,6 V : 2,2 V : 2,8 V : 3,4 V : 4,0 V

Logo, o "N" deve ser igual a seis. O cálculo ficaria assim:

$|4,0 - 1,0| \times 1000 = 3000$ $\quad |0,6| \times 1000 = 600$ $\quad N = 3000 / 600 + 0,6 = 5,6$

Arredondando para inteiro, implica em **N = 6**

3.2.3.5 SubVI VOLT

O SubVI **VOLT** calcula o valor da tensão e envia comandos ao instrumento para alimentar o dispositivo. O cálculo é feito da seguinte forma:

$$\text{Tensão} = \text{Viníco} + (\text{Passo} \times \text{Índice})$$

Para explicar melhor o processo, consideremos o código da figura 3.21:

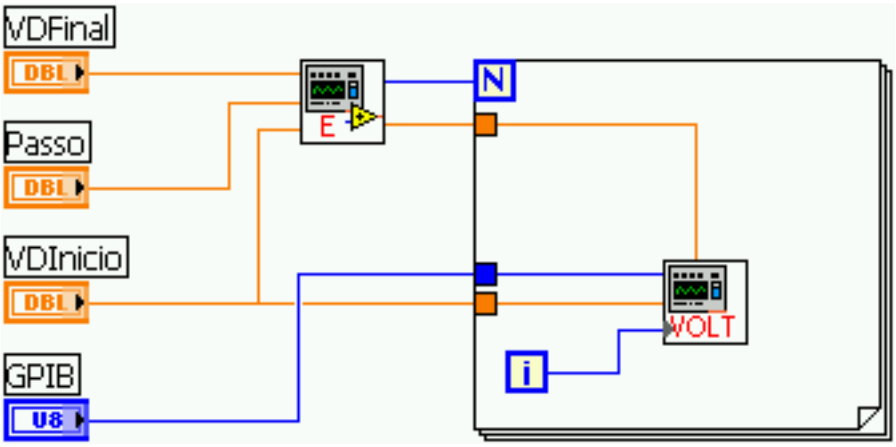


Figura 3.21 - Código de exemplo

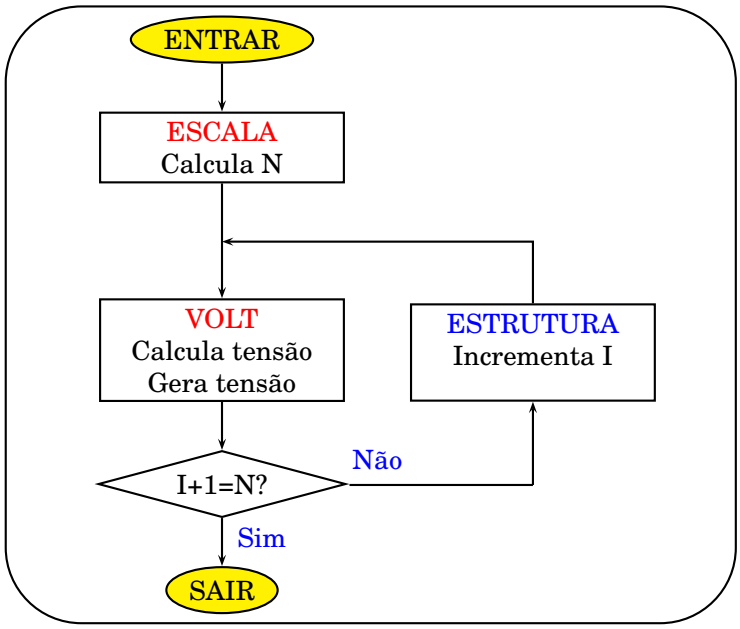


Figura 3.22 - Fluxograma do exemplo

Se o usuário fornecer *VInício = 1 V*; *VFinal = 4 V*; e *Passo = 0,6 V*, o SubVI **ESCALA** calcula N=6. O índice "I" da estrutura varia de "0" a "N-1". O SubVI **VOLT** fará os seguintes cálculos:

$1{,}0 + 0{,}6 \times 0 = 1{,}0$ $\quad$ $1{,}0 + 0{,}6 \times 1 = 1{,}6$ $\quad$ $1{,}0 + 0{,}6 \times 2 = 2{,}2$

$1{,}0 + 0{,}6 \times 3 = 2{,}8$ $\quad$ $1{,}0 + 0{,}6 \times 4 = 3{,}4$ $\quad$ $1{,}0 + 0{,}6 \times 5 = 4{,}0$

O código do SubVI é mostrado na figura 3.23. O valor numérico é convertido em *string* pela função *Format Value*. O código de formatação "%.;%+08.3" significa oito caracteres com três casas decimais. Os lugares vazios são preenchidos por zero. O número sempre apresenta sinal e o ponto é usado como separador de casas decimais. Exemplo: -065.300 e +008.020.

A função *Search and Replace String* é usada para montar o comando que será enviado ao instrumento. A entrada *GPIB* fornece o endereço do instrumento à função GPIB *Send*.

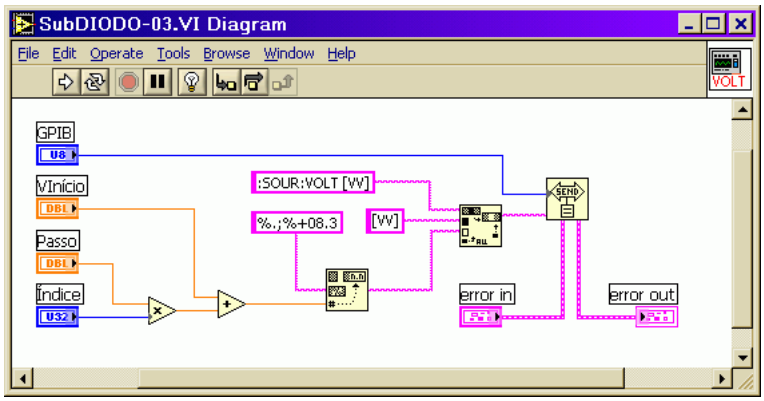


Figura 3.23 - SubVI **VOLT**

A entrada *error in* e a saída *error out* são usadas para controlar a sequência de execução do programa. Servem também para depuração. Para mais detalhes, os manuais e a ajuda do LabVIEW [7], [8] podem ser consultados. Os comandos SCPI da tabela 3.2 serão enviados:

Índice **I**	COMANDOS SCPI
0	:SOUR:VOLT +001.000
1	:SOUR:VOLT +001.600
2	:SOUR:VOLT +002.200
3	:SOUR:VOLT +002.800
4	:SOUR:VOLT +003.400
5	:SOUR:VOLT +004.000

Tabela 3.2 - Comandos para geração de tensão

3.2.3.6 SubVI LER

O SubVI **LER** envia um comando para o instrumento realizar uma medida, pega o resultado e o formata para que seja gravado no arquivo.

A entrada *GPIB* informa o endereço do instrumento. Primeiramente envia-se o comando ":READ?". O Keithley 2400 fornece uma palavra *string* com 5 números com 13 caracteres separados por vírgulas. O leitor deve se referir ao exemplo abaixo. O primeiro valor é a tensão gerada, o segundo mostra a corrente consumida, o terceiro é a resistência da carga, o quarto informa o tempo e o quinto o estado do instrumento.

+6.000000E-01,+1.000236E-04,+5.998584E+03,+7.282600E+01,+4.813200E+04

A resposta do instrumento é capturada pela função GPIB *Receive*. Os valores da tensão e da corrente são separados pelas funções *String Subset*. A entrada *offset* da primeira função não está ligada, de maneira que o valor zero é assumido. Essa função *String Subset* começa ler a partir do primeiro caracter, o que corresponde ao valor da tensão. A segunda função *String Subset* tem o *offset* ligado ao valor 14, que corresponde ao primeiro caracter do valor da corrente. Ambas estão programadas para ler 13 caracteres através da entrada *length*.

As funções *Search and Replace String* são usadas para fornecer o resultado à saída *VdId*. Elas substituem os caracteres "[VV]" e "[II]" na palavra "V=[VV]V I=[II]A". A resposta é apresentada no seguinte formato:

V=+6.000000E-01V I=+1.000236E-04A

A figura 3.24 identifica as funções. Na figura 3.25, temos o código fonte em LabVIEW, e na figura 3.26, o fluxograma equivalente.

Figura 3.24 - As funções do SubVI

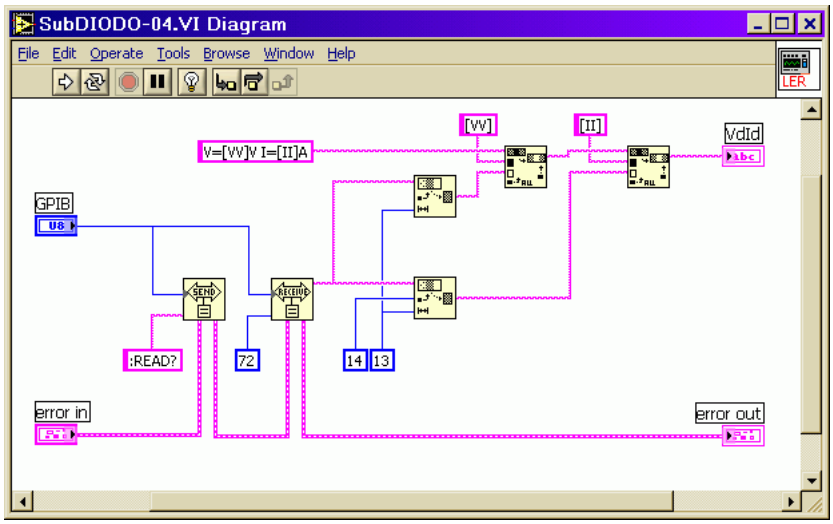


Figura 3.25 - SubVI **LER**

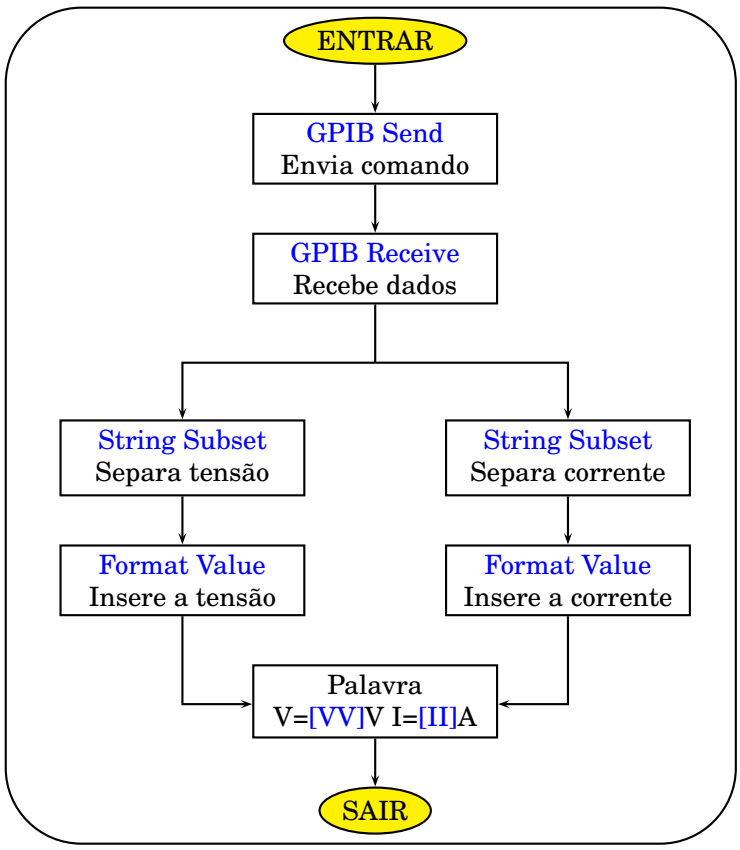


Figura 3.26 - Fluxograma do SubVI **LER**

3.2.3.7 SubVI SALVAR

O SubVI **SALVAR** cria os arquivos de texto e salva o resultado das medidas. O nome do arquivo e o diretório fornecido pelo usuário é recebido na entrada *Caminho*. O valor é convertido de *path* para *string* e entregue à função *Format Value*. Com o código de formatação "-%02D.TXT", a função acrescenta um hífen, um índice com dois algarismos e a extenção ".TXT". Supondo que o usuário forneça o valor " C:\MEDIDAS\LED", como na figura 3.27, e faça seis medidas, os resultados são salvos no diretório C:\MEDIDAS como: LED-01.TXT, LED-02.TXT, LED-03.TXT, LED-04.TXT, LED-05.TXT e LED-06.TXT.

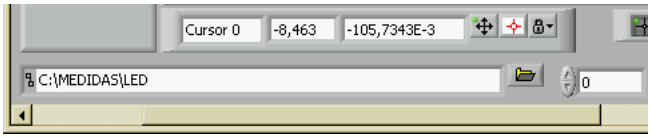


Figura 3.27 - Nome do Arquivo

Os resultados presentes na entrada *Medidas* são organizados pela função *Array to Spreadsheet String* (ATSS) . O código "%s" indica que o *array* de entrada é do tipo *string*. O código "\n" indica que o caracter de retorno é usado para separar os valores do *array*.

Na entrada *GPIB* é fornecido o endereço do instrumento. A tabela 3.3 mostra os comandos SCPI enviados ao instrumento. O valor da corrente de proteção (IPROT), apresentado pela função GPIB *Receive*, é anexado ao arquivo pela função *Concatenate Strings*. A função *Write Characters To File* (WCTF) cria o arquivo e salva os dados. A saída *Arquivo* informa o caminho e nome completo do arquivo.

:OUTPUT OFF	Desliga a saída
:ABORT	Volta ao estado ocioso.
:SENS:CURR:PROT?	Informa a corrente de proteção.

Tabela 3.3 - Comandos SCPI para o Keithley 2400

A figura 3.28 identifica as principais funções usadas. Nas figuras 3.29 e 3.30 são exibidos o código e o fluxograma, respectivamente.

Figura 3.28 - Principais funções do SubVI

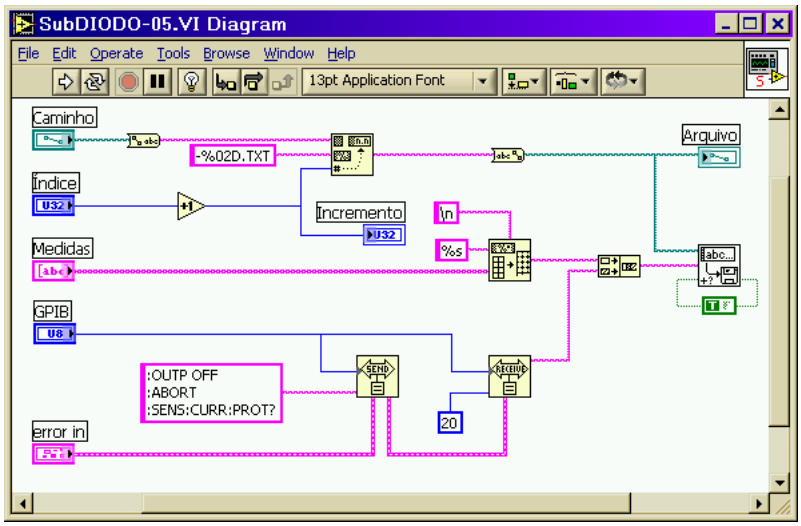


Figura 3.29 - SubVI **SALVAR**

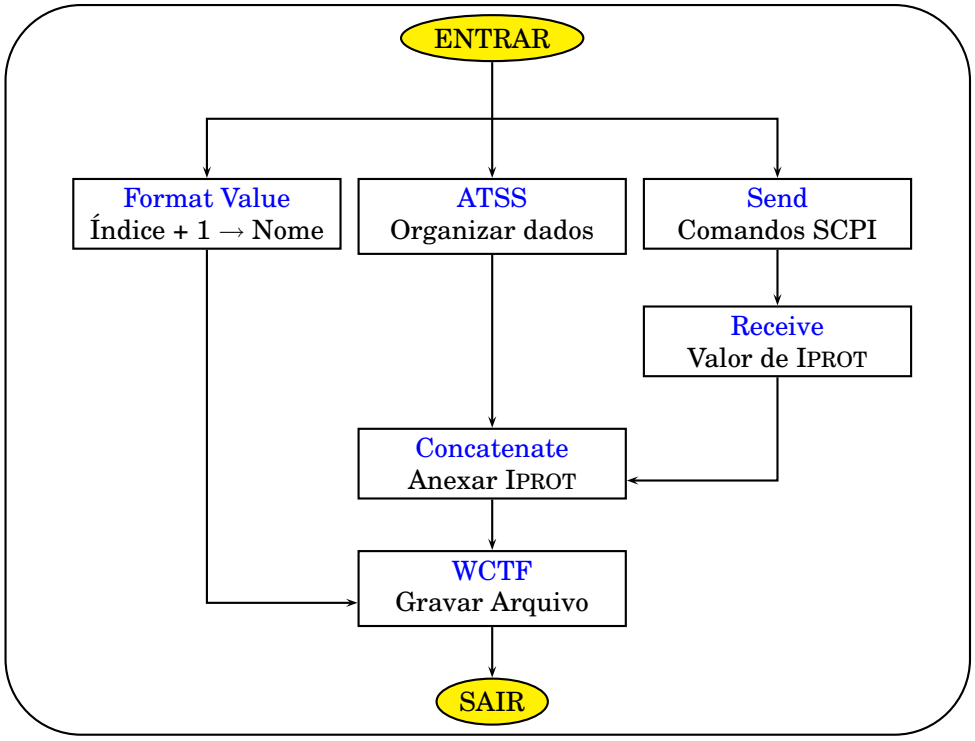


Figura 3.30 - Fluxograma do SubVI **SALVAR**

3.2.3.8 SubVI GRÁFICO

O SubVI GRÁFICO lê os dados de um arquivo e os formata para serem exibidos no gráfico. Também fornece em saídas separadas a corrente de proteção e um *array* contendo os valores das tensões para serem exibidos no modo carregar arquivo.

O arquivo de texto salvo possui o formato mostrado a seguir. O último valor é o da corrente de proteção.

V=+2.475000E+00V I=+2.000294E-03A
V=+2.480000E+00V I=+2.000293E-03A
V=+2.485000E+00V I=+2.000292E-03A
V=+2.490000E+00V I=+2.000294E-03A
V=+2.495000E+00V I=+2.000292E-03A
V=+2.500000E+00V I=+2.000295E-03A
2.000000E-03

A entrada *Arquivo* fornece o nome e o caminho do arquivo. A função *Read Characters From File* (RCFF) dentro da estrutura *While Loop* lê os valores até chegar no final do arquivo (*eof*). Então interrompe a estrutura.

Todas as linhas, exceto a última, possuem 35 caracteres, incluindo o retorno que não é visível. Dentro da estrutura *While Loop*, o índice é multiplicado por 35 e ligado na entrada *offset* da função *Read Characters From File* (RCFF). Por isso a cada execução da estrutura (*Loop*), a leitura começa na próxima linha. São lidos 50 caracteres por execução, o que inclui parte da linha posterior. Quando chega na penúltima linha, a função detecta o fim do arquivo (eof) e para a estrutura. Apesar disso, a corrente de proteção na última linha é lida. Os valores então são entregues às saídas.

Para cada leitura os valores da tensão e corrente são convertidos de *string* para numérico. As funções *Frac/Exp String To Number* (FESTN) possuem suas entradas *offset* numeradas de forma a começar a ler apartir do primeiro caracter

dos valores. Param quando encontram um caracter não numérico, como um espaço. Em seguida, os números são juntados em um *cluster* pela função *Bundle*, e finalmente organizados em um *array*. Esse *array* de *clusters* é o formato padrão do LabVIEW para exibição gráficas. As tensões também são fornecidas à uma saída separada em forma de *array*.

No caso da corrente de proteção, o valor não é indexado. Desta maneira apenas o último valor é fornecido ao sair da estrutura. Neste exemplo o valor é 2.000000E-03.

As figuras 3.31, 3.32 e 3.33 mostram as principais funções, o código em LabVIEW e o fluxograma.

RCFF FESTN Bundle

Figura 3.31 - Principais funções do SubVI

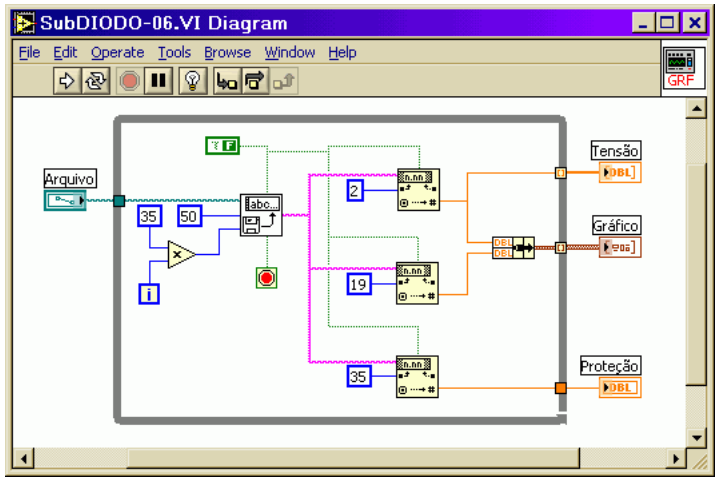


Figura 3.32 - SubVI **GRÁFICO**

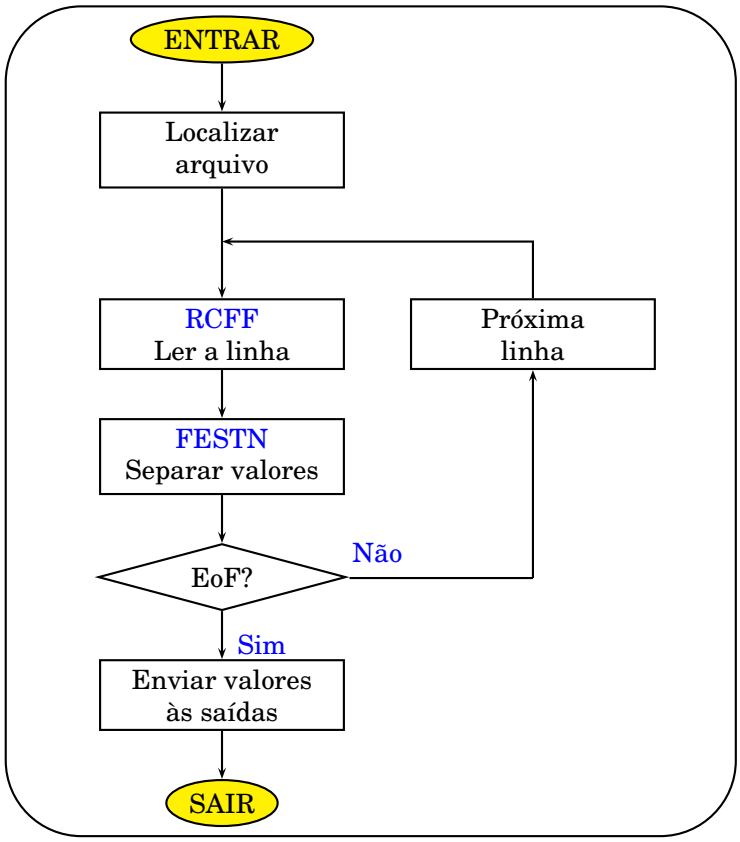


Figura 3.33 - Fluxograma do SubVI **GRÁFICO**

3.3 MODELO PARA PROGRAMAR <SPOOL>

O modelo **SPOOL** usa como núcleo as funções GPIB *TestSRQ* e *ReadStatus*. Tem como objetivo verificar a ocorrência de um evento no instrumento, como, por exemplo, se há resultados disponíveis, se ocorreu algum erro ou quando seu *buffer* está cheio. O instrumento, no caso, é previamente programado com comandos SCPI para gerar SRQ (*Service Request*) no barramento quando ocorrer o evento desejado.

A linha SRQ é uma das 5 linhas de manutenção do barramento GPIB. O instrumento coloca esse bit em nível alto quando deseja se comunicar. A função *TestSRQ* verifica a presença do sinal SRQ no barramento e indica com sua saída *boolean*. A função *ReadStatus* é endereçada a um instrumento, lê o registro de estado do instrumento, retorna o seu valor convertido de binário para decimal e zera a saída SRQ. Este procedimento também é chamado de **Serial Pool**. As figuras 3.34 e 3.35 mostram as duas funções.

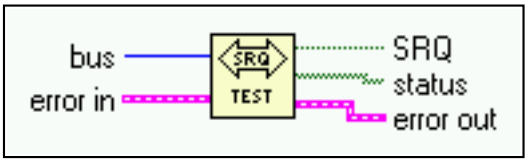


Figura 3.34 - Função TestSRQ

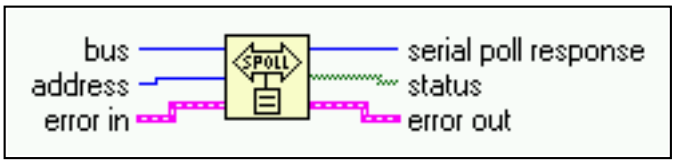


Figura 3.35 - Função ReadStatus.

A figura 3.36 mostra essas funções operando em conjunto com uma estrutura *Case* dentro de um *loop*. A figura 3.37 mostra o fluxograma correspondente. Este trecho de código tem a função de fazer um programa esperar até que uma determinada condição ocorra em um instrumento. O instrumento irá sinalizar com SRQ no barramento. O barramento GPIB é lido a cada segundo. Se o sinal estiver ausente, envia-se *False* ao controle do *loop*, mantendo o programa estacionado.

Quando o sinal SRQ estiver presente no barramento, a saída é *True* e a estrutura *CASE* executa a função *ReadStatus* que endereça o instrumento desejado e lê o seu registro de estado. O resultado é comparado com o valor desejado, no caso 65. Se a condição for verdadeira, o programa sai do *loop* e vai para o próximo passo. Caso contrário, o programa volta ao estado anterior. Para maiores detalhes sobre o registro de estados, programação, SRQ e SPOOL, o manual do instrumento usado deve ser consultado.

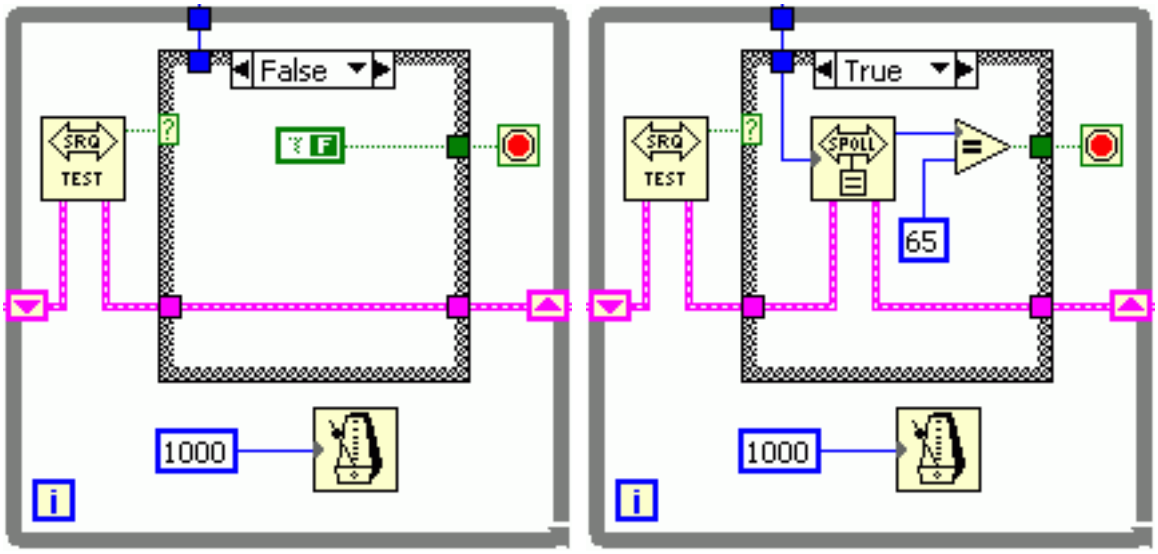


Figura 3.36 - SPOOL

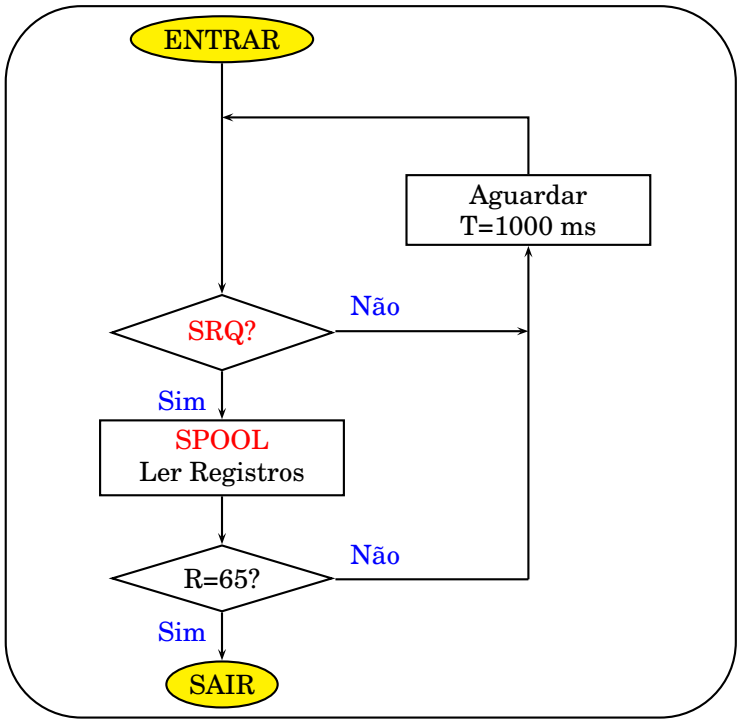


Figura 3.37 - Funcionamento do Spool

4 PROCEDIMENTOS DE MEDIDAS E RESULTADOS

Este capítulo apresenta de maneira detalhada uma seleção de procedimentos de medidas implantados no laboratório. A escolha desses procedimentos define um conjunto representativo das atividades desenvolvidas no âmbito da dissertação de mestrado. Para cada circuito ou tipo de componente testado é desenvolvido um procedimento. Os procedimentos são divididos em três partes. A primeira parte se chama Sistema de Medidas. Trata-se dos instrumentos e bancadas usados para os testes. A segunda, Descrição do Procedimento, descreve detalhes de execução. Mostra diagramas de ligações e os parâmetros elétricos envolvidos. Nos procedimentos em que foram desenvolvidos programas, o funcionamento e a utilização dos mesmos são explicados. Finalmente, em Medidas, Resultados e Análise de Desempenho, são apresentados os resultados e conclusões sobre os mesmos.

4.1 CARACTERIZAÇÃO DE TRANSISTORES

Consiste em traçar em um computador as curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ [23], [24], [25] de transistores MOS usando os instrumentos fonte e medidor Keithley 2400 e o eletrômetro Keithley 6517A [26], [27]. Os instrumentos são controlados através das interfaces GPIB por programas feitos em LabVIEW. São programados para gerar tensão e medir corrente. Os resultados são salvos em arquivos de texto para análise posterior.

4.1.1 Sistema de Medidas

Os instrumentos usados podem ser vistos nas figuras 4.1 [22] e 4.2. O Keithley 2400 é fonte de tensão e corrente. Mede resistência, corrente e tensão. O Keithley 6517A é um eletrômetro. Mede também carga capacitiva e outros parâmetros. O apêndice A apresenta uma descrição mais detalhada desses instrumentos. Para essa aplicação, ambos são configurados como geradores de tensão e medidores de corrente. Os testes foram feitos em protótipos não encapsulados, usando as estações de medidas apropriadas.

Figura 4.1 - Keithley 2400

Figura 4.2 - Keithley 6517A

4.1.2 Descrição do procedimento

Para obter as curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ dos transistores MOS, os instrumentos são configurados para gerar tensão e medir corrente. O Keithley 2400 é ligado entre o dreno e a fonte do transistor. O eletrômetro polariza a porta do dispositivo. Possui uma fonte de tensão que é ligada em série com a sua entrada configurada como amperímetro.

4.1.2.1 Ligação dos instrumentos

O Keithley 2400 gera tensão e mede corrente através dos conectores banana fêmea *INPUT*/*OUTPUT HI* e *LO* dianteiros ou traseiros. A figura 4.3 mostra a ligação de um DUT no painel frontal.

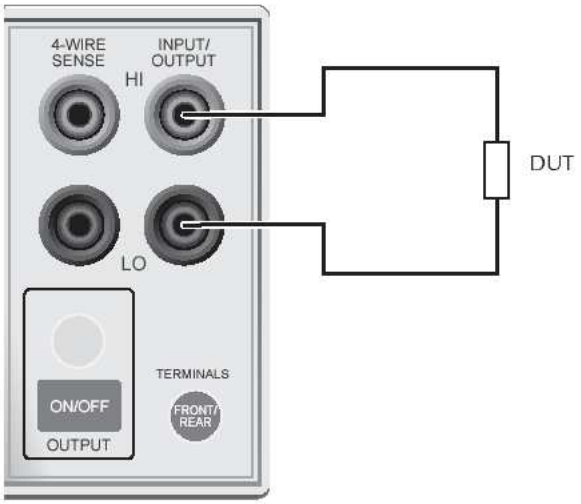


Figura 4.3 - Ligação do DUT

Já o Eletrômetro Keithley 6517A possui entradas e saídas separadas e exige uma ligação mais complexa. Se for programado para gerar tensão e medir corrente, a tensão é gerada nas saídas *V SOURCE HI* e *LO* que usam conectores tipo banana. A corrente é medida através da entrada *INPUT* que usa o cabo 7078-TRX-3 fornecido com o instrumento, possuindo um conector triax macho em uma extremidade e 3 conectores tipo jacaré na outra. O jacaré com capa vermelha (VM) é a entrada positiva do amperímetro. O jacaré com capa preta (PT) é a entrada negativa. O jacaré de capa verde (VD) é o terra, estando em contato com o neutro da rede elétrica, com a carcaça e com o conector banana fêmea verde atrás do instrumento. A figura 4.4 mostra os conectores traseiros do instrumento.

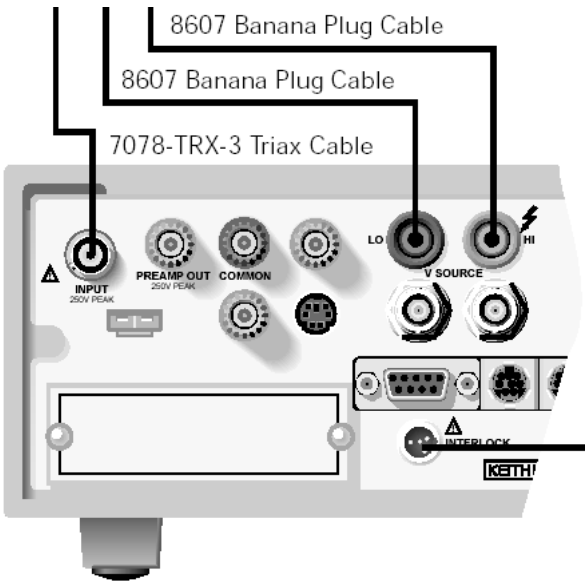


Figura 4.4 - Entradas e saídas do eletrômetro

Para extrair as curvas, deve-se ligar o amperímetro em série com a fonte de tensão e com a entrada positiva aterrada. A figura 4.5 mostra como ligar os cabos. A figura 4.6, por sua vez, ilustra o diagrama do circuito equivalente.

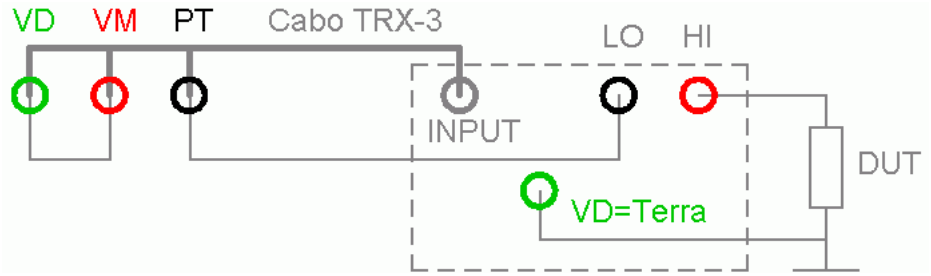


Figura 4.5 - Ligando os cabos no eletrômetro

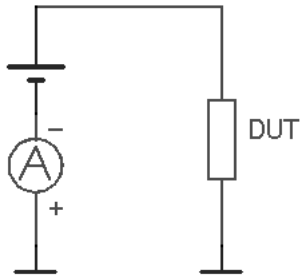


Figura 4.6 - Circuito equivalente

Para caracterizar transistores MOS, o diagrama completo usando os dois instrumentos é mostrado na figura 4.7. O Keithley 2400 gera V_{DS} e mede I_{DS}. O Eletrômetro gera a tensão na porta e mede a corrente que teoricamente deveria ser nula. Na prática sempre é detectado uma pequena fuga em torno de nano ampères ou menos.

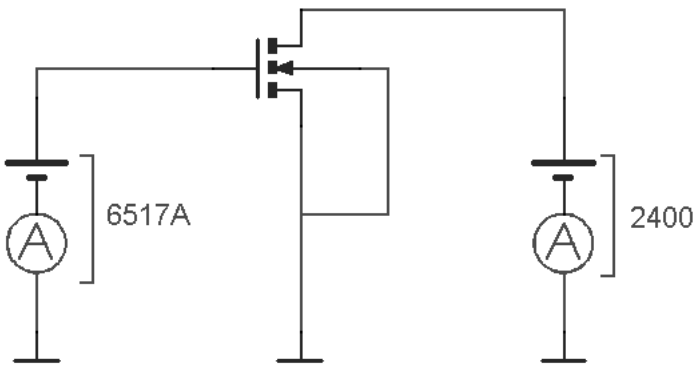


Figura 4.7 - Caracterização do transistor

4.1.2.2 Extraindo a curva $I_{DS}xV_{DS}$.

O procedimento é executado automaticamente por um programa feito em LabVIEW que controla os instrumentos. São usados seis valores de V_{GS} determinados pelo usuário. Com o Eletrômetro aplicando a tensão $V_{GS\ 1}$ na porta, o 2400 varia a tensão de 0 até o valor de $V_{DS\ MAX}$ determinado pelo usuário. O incremento de tensão é definido no campo *Passo*. Para cada valor de V_{DS}, os valores de corrente de dreno e porta do transistor são medidos e armazenados na memória do computador. Então aplica-se $V_{GS\ 2}$. O processo é repetido até chegar em $V_{GS\ 6}$. No fim do processo, os resultados das medidas são plotados no gráfico e gravados em arquivo.

Na figura 4.8 podemos ver o programa em funcionamento. Na figura 4.9 temos o fluxograma correspondente. Existem dois campos chamados *Endereço*. O primeiro se refere ao endereço GPIB do instrumento fonte e medidor Keithley 2400 com o número 24. O segundo é do eletrômetro Keithley 6517A que está no endereço 27. Repare os seis campos de tensão $V_{GS\ 1}$ a $V_{GS\ 6}$. No campo *Proteção*, o usuário fornece o limite de corrente que o Keithley 2400 deve fornecer para não danificar o dispositivo.

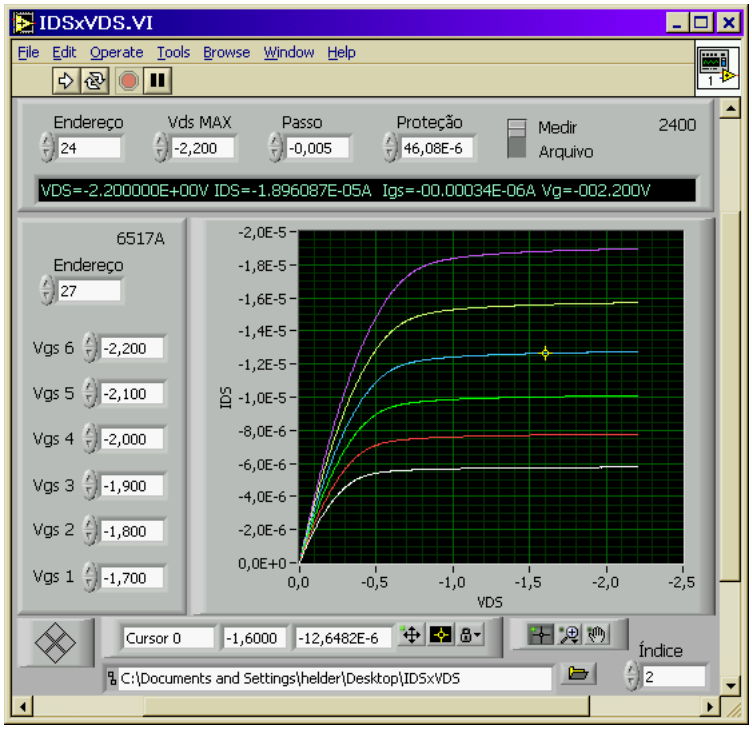


Figura 4.8 - Programa IDSxVDS.

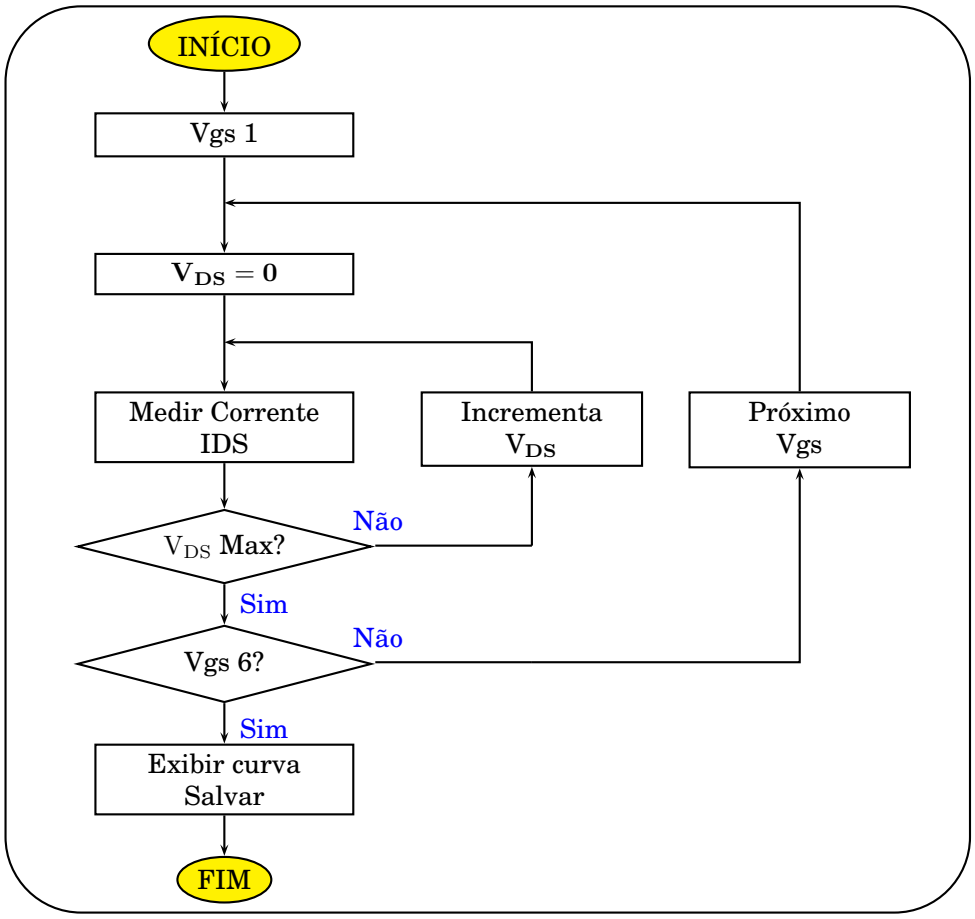


Figura 4.9 - Fluxograma IDSxVDS

O apêndice E explica de maneira detalada o código fonte do programa e seu funcionamento.

4.1.2.3 Extraindo a curva $I_{DS}xV_{GS}$

Mantendo as mesmas ligações do procedimento anterior, usa-se outro programa. Agora o eletrômetro Keithley 6517A varia a tensão na porta V_{GS} com o incremento fornecido no campo *Passo*. A tensão V_{DS} é fixa. Para cada valor de V_{GS} é medido a corrente de dreno I_{DS}. Na figura 4.10 podemos ver a tela do programa em funcionamento, e na figura 4.11, seu fluxograma. A curva é mostrada no gráfico e salva em arquivo no HD do computador. O apêndice E explica de maneira detalada o código fonte e o funcionamentodo do programa.

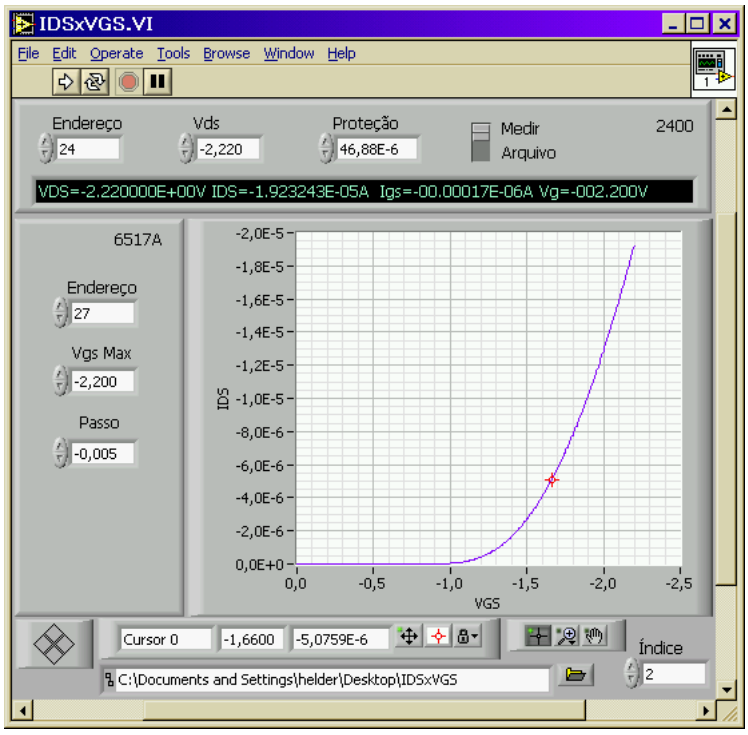


Figura 4.10 - Programa IDSxVGS.

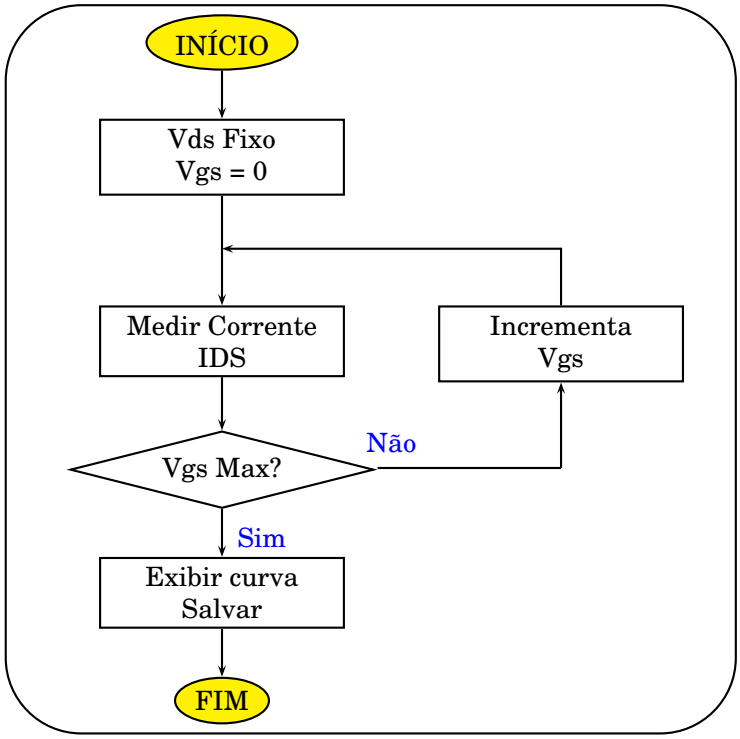


Figura 4.11 - Fluxograma IDSxVGS

4.1.3 Medidas, Resultados e Análise de Desempenho

4.1.3.1 Transistor de RF

No FAPESP 119, foi construído um transistor isolado igual ao utilizado no transceptor de RF [28], [29] para testes e caracterização. Os *pads* foram posicionados com o espaçamento padrão para as pontas de teste de RF da estação de medidas. As medidas foram feitas em um *chip* não encapsulado usando a estação de medidas para RF Cascade. A figura 4.12 mostra os *pads* do transistor de RF. O dreno é indicado pela letra "D", as duas portas por "G1" e "G2" e a fonte, que é ligada ao terra, possui vários *pads*, indicados com a letra "S". O Substrato é especificado pela letra "B".

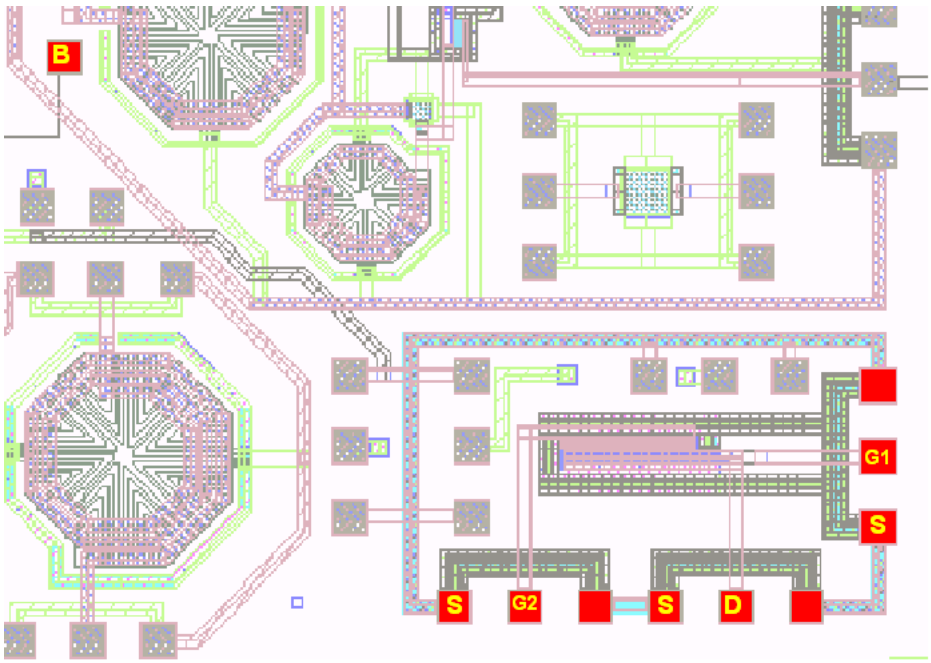


Figura 4.12 - Identificando o transistor de RF do FAPESP 119.

A figura 4.13 mostra a estação de medidas Cascade. A figura 4.14 mostra o *chip* sendo testado na estação de medidas Cascade.

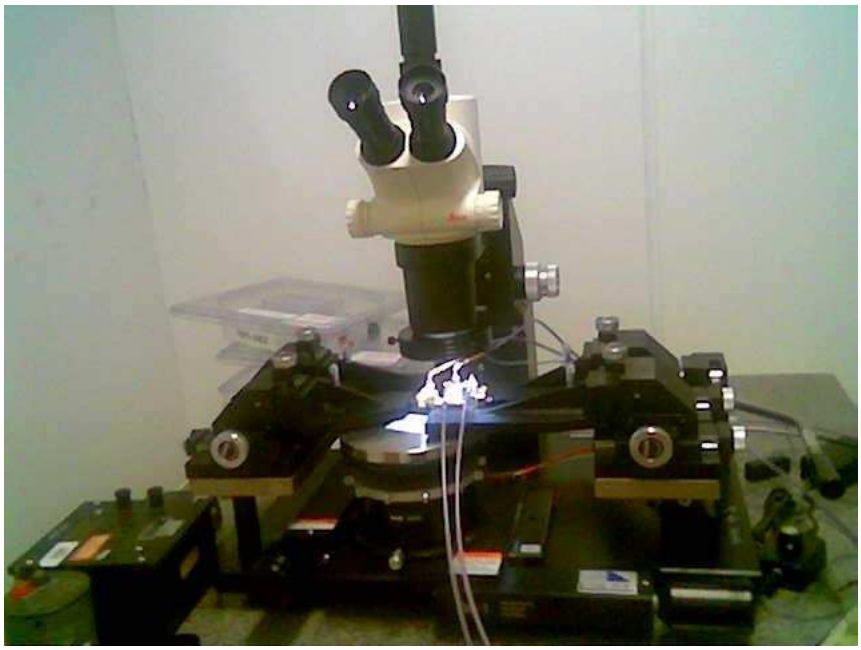


Figura 4.13 - A estação de medidas em RF Cascade

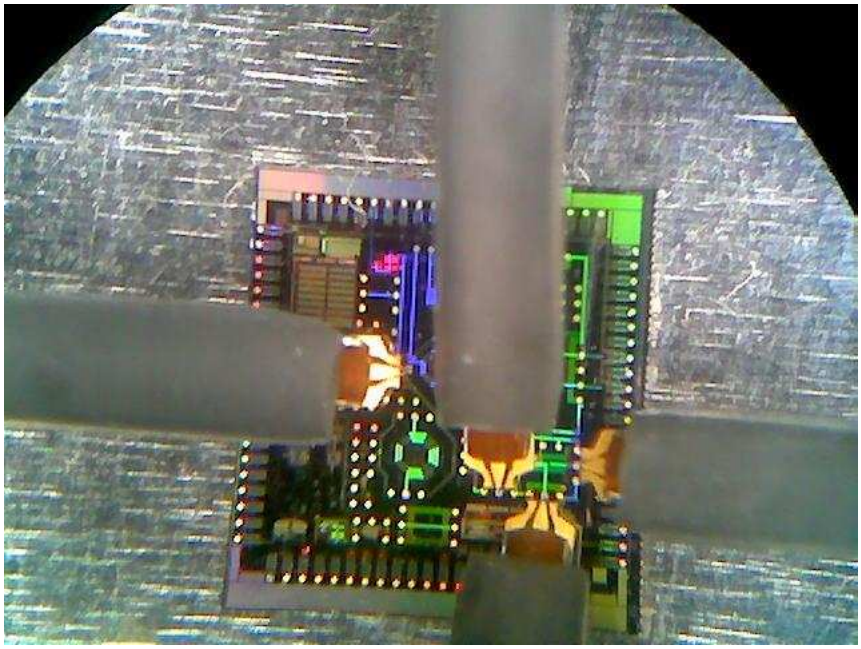


Figura 4.14 - Transistor de RF sendo testado

Os testes foram feitos com tensões de no máximo 0,6 V e com incrementos de até 0,005 V .As medidas apresentaram curvas estáveis. O transistor funciona adequadamente operando com baixas tensões. Isso é um bom resultado porque o transceptor em que o dispositivo é usado deve operar com tensões de alimentação e excitação baixas. As curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ são exibidas na figura 4.15.

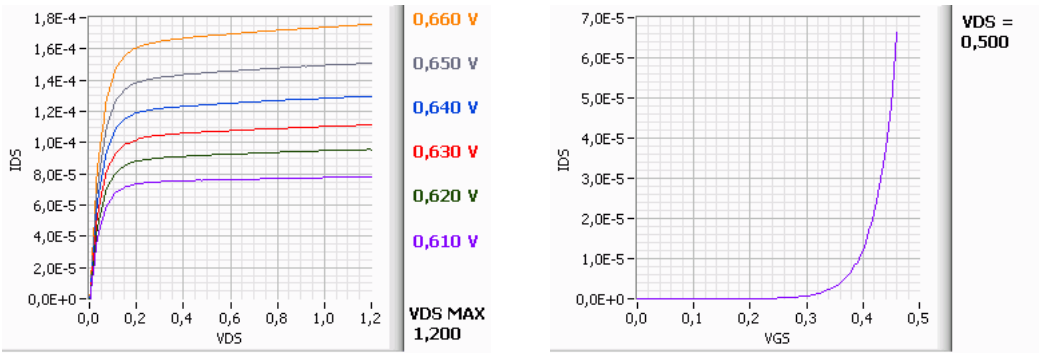


Figura 4.15 - Curvas do transistor de RF

O transistor não foi extraído de uma biblioteca existente mais foi projetado e desenhado usando as ferramentas do Cadence. O dispositivo funciona bem com pequenas tensões e parece ser adequado para o transceptor de RF. Entretanto, é necessário que sejam realizadas medidas dinâmicas com o dispositivo usando as frequências de trabalho. O procedimento para esses testes já foi idealizado. A realização dos testes, todavia, depende da aquisição de acessórios para o analisador de redes, que no momento está sendo providenciada.

4.1.3.2 Transistores isolados PMOS e NMOS

Foram também construídos no FAPESP 119 mais 6 transistores isolados. Três são do tipo PMOS e os outros do tipo NMOS. Cada conjunto PMOS/NMOS possui a fonte em comum. Estão dispostos com *pads* internos para testes, conforme mostrado na figura 4.16. Nos *pads* destacados em vermelho a letra **B** significa ligado ao substrato, **D** faz contato com os drenos, **S** faz contato com as fontes e **G** faz contato com as portas.

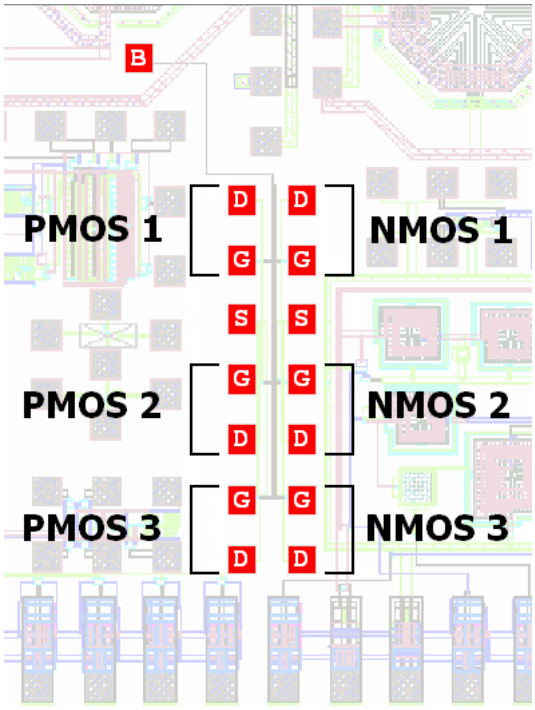


Figura 4.16 - Transistores isolados do FAPESP 119

Os testes foram feitos na estação de medidas com ponteiras em *chips* não encapsulados. Os transistores PMOS 1 e NMOS 1 possuem W=10 μm e L=0,5 μm. Os dispositivos PMOS 2 e NMOS 2 possuem W=10 μm e L=10 μm. Finalmente, os transistores PMOS 3 e NMOS 3 possuem W=0,5 μm e L=10 μm. A espessura do óxido de silício é de 7,6 nm. Os transistores foram simulados no Cadence e os resultados em forma de tabelas foram exportados para um programa feito em LabVIEW. Assim pode-se comparar os resultados simulados com os resultados obtidos na prática.

A figura 4.17 mostra as curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do transistor NMOS 1, resultante da simulação no Cadence. A figura 4.18 mostra os resultados obtidos no laboratório.

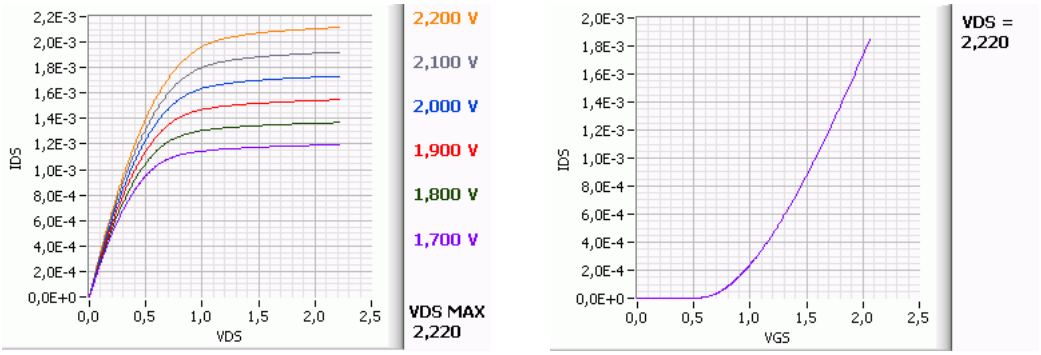


Figura 4.17 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do NMOS 1 (Simulado)

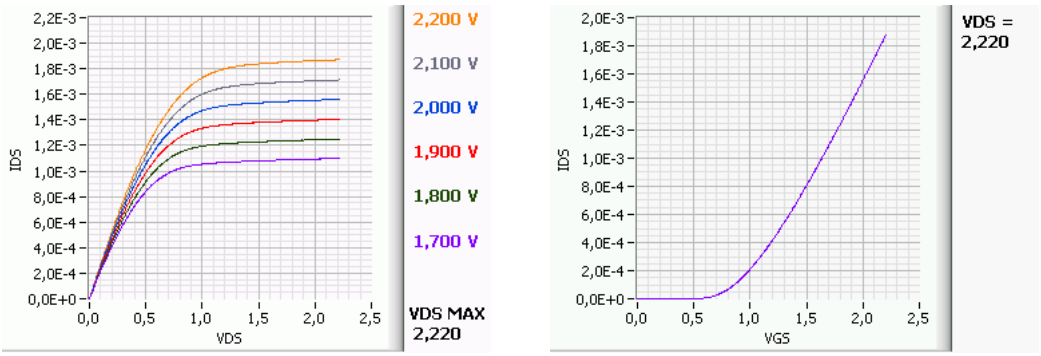


Figura 4.18 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do NMOS 1 (Medido)

Os resultados apresentam valores bem próximos. Fatores externos como a variação da temperatura ambiente podem influenciar nos resultados. Pode-se concluir que os resultados da simulação são válidos na prática e podem ser usados no projeto de novos circuitos.

Para o transistor NMOS 1, os valores podem ser comparados nas tabelas 4.1 e 4.2.

I_{DS}xV_{DS} para V_{DS}=2.0		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
+1,700 V	$+1.191000\times10^{-03}$ A	$+1.093898\times10^{-03}$ A
+1,800 V	$+1.364000\times10^{-03}$ A	$+1.244530\times10^{-03}$ A
+1,900 V	$+1.541000\times10^{-03}$ A	$+1.397831\times10^{-03}$ A
+2,000 V	$+1.724000\times10^{-03}$ A	$+1.553208\times10^{-03}$ A
+2,100 V	$+1.913000\times10^{-03}$ A	$+1.706157\times10^{-03}$ A
+2,200 V	$+2.105000\times10^{-03}$ A	$+1.864217\times10^{-03}$ A

Tabela 4.1 - Valores extraídos das curvas I_{DS}xV_{DS}

I_{DS}xV_{GS}		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
+0,500 V	$+9.399000\times10^{-07}$ A	$+6.143026\times10^{-07}$ A
+1,000 V	$+2.420000\times10^{-04}$ A	$+2.107197\times10^{-04}$ A
+1,500 V	$+8.907000\times10^{-04}$ A	$+8.076247\times10^{-04}$ A
+2,000 V	$+1.760000\times10^{-03}$ A	$+1.558038\times10^{-03}$ A

Tabela 4.2 - Valores extraídos da curva I_{DS}xV_{GS}

A figura 4.19 mostra as curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do transistor NMOS 2, resultante da simulação no Cadence. A figura 4.20 mostra os resultados obtidos no laboratório.

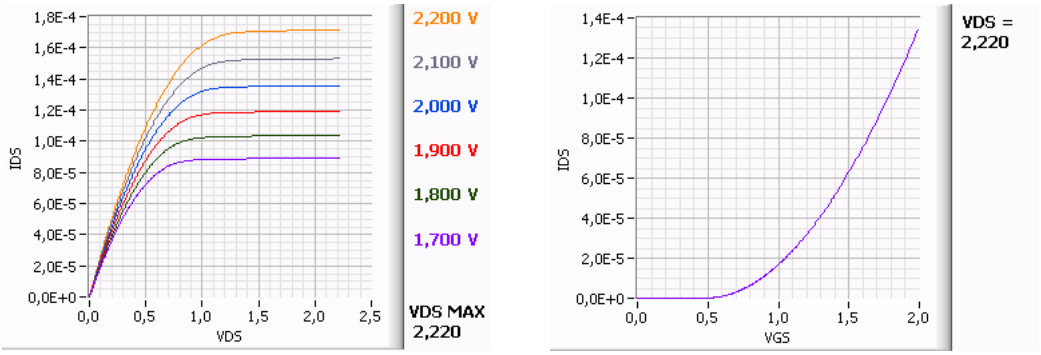


Figura 4.19 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do NMOS 2 (Simulado)

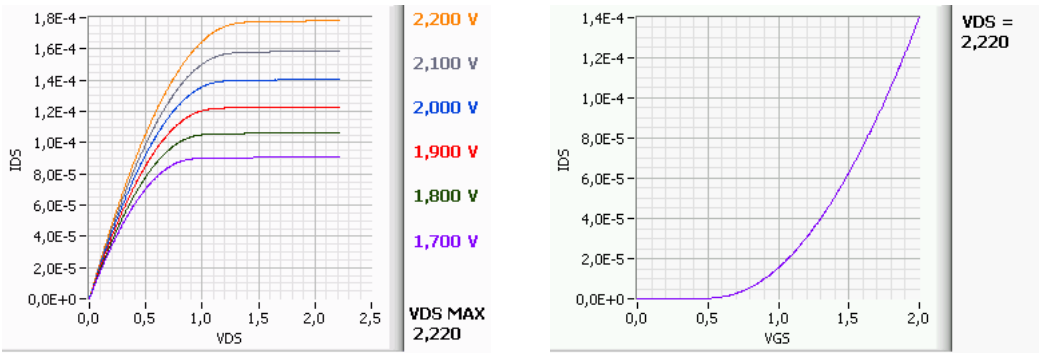


Figura 4.20 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do NMOS 2 (Medido)

Os resultados apresentam valores ainda mais próximos que os do transistor anterior. O aumento do L de 0,5μm para 10μm pode ter gerado um dispositivo mais estável. Novamente conclui-se que os resultados da simulação são válidos na prática e podem ser usados no projeto de novos circuitos.

Para o transistor NMOS 2, os valores podem ser comparados nas tabelas 4.3 e 4.4.

I_{DS} para V_{DS}=2.0		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
+1,700 V	$+8.896000\times10^{-05}$ A	$+9.061683\times10^{-05}$ A
+1,800 V	$+1.035000\times10^{-04}$ A	$+1.061208\times10^{-04}$ A
+1,900 V	$+1.191000\times10^{-04}$ A	$+1.226951\times10^{-04}$ A
+2,000 V	$+1.356000\times10^{-04}$ A	$+1.403280\times10^{-04}$ A
+2,100 V	$+1.530000\times10^{-04}$ A	$+1.583958\times10^{-04}$ A
+2,200 V	$+1.713000\times10^{-04}$ A	$+1.779001\times10^{-04}$ A

Tabela 4.3 - Valores extraídos das curvas I_{DS}xV_{DS}.

I_{DS}xV_{GS}		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
+0,500 V	$+1.572000\times10^{-07}$ A	$+2.025017\times10^{-07}$ A
+1,000 V	$+1.703000\times10^{-05}$ A	$+1.540533\times10^{-05}$ A
+1,500 V	$+6.309000\times10^{-05}$ A	$+6.285871\times10^{-05}$ A
+2,000 V	$+1.371000\times10^{-04}$ A	$+1.401969\times10^{-04}$ A

Tabela 4.4 - Valores extraídos da curva I_{DS}xV_{GS}.

A figura 4.21 mostra as curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do transistor NMOS 3, resultante da simulação no Cadence. A figura 4.22 mostra os resultados obtidos no laboratório.

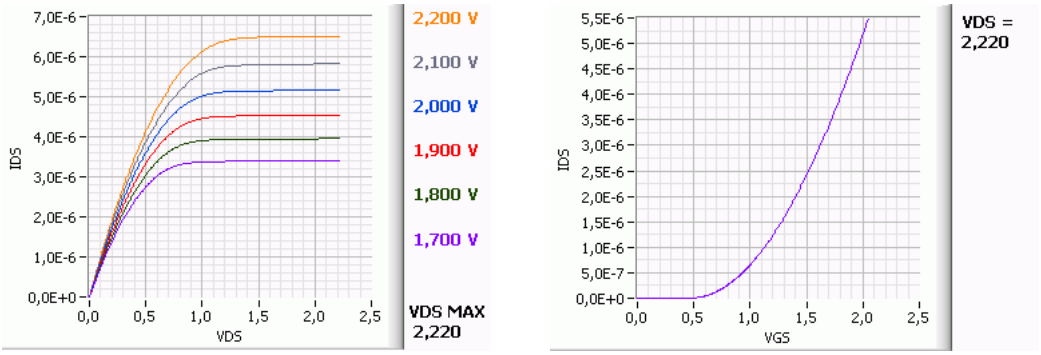


Figura 4.21 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do NMOS 3 (Simulado)

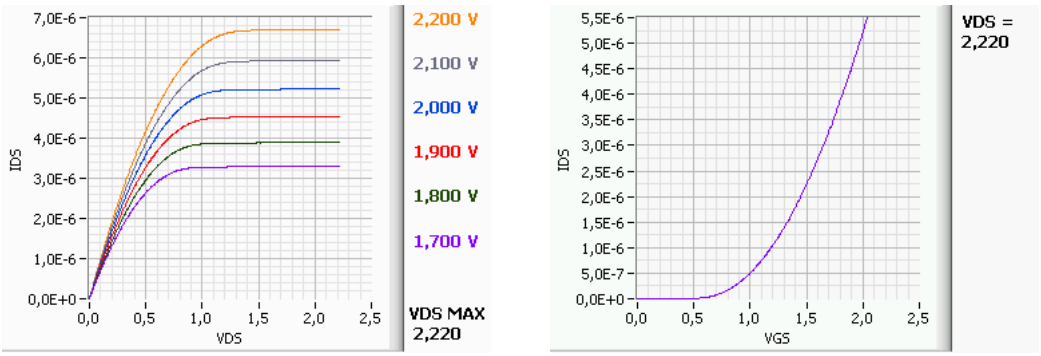


Figura 4.22 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do NMOS 3 (Medido)

Os resultados estão praticamente idênticos. Como nesse caso o L foi mantido e o W caiu de 10μm para 0,5μm, pode-se concluir que o dispositivo se torna mais estável com a diminuição de W em relação a L.

Podemos também concluir pelos gráficos que o transistor 1 apresentou maior transcondutância, seguido pelo transistor 2 e o transistor 3 obteve a menor. Como visto anteriormente esse parâmetro é proporcional à relação W/L.

Para o transistor NMOS 3, os valores podem ser comparados nas tabelas 4.5 e 4.6.

I_{DS} para V_{DS}=2.0		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
+1,700 V	$+3.398000\times10^{-06}$ A	$+3.295248\times10^{-06}$ A
+1,800 V	$+3.949000\times10^{-06}$ A	$+3.889775\times10^{-06}$ A
+1,900 V	$+4.536000\times10^{-06}$ A	$+4.533322\times10^{-06}$ A
+2,000 V	$+5.160000\times10^{-06}$ A	$+5.222399\times10^{-06}$ A
+2,100 V	$+5.817000\times10^{-06}$ A	$+5.934009\times10^{-06}$ A
+2,200 V	$+6.508000\times10^{-06}$ A	$+6.708457\times10^{-06}$ A

Tabela 4.5 - Valores extraídos das curvas I_{DS}xV_{DS}.

I_{DS}xV_{GS}		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
+0,500 V	$+7.471000\times10^{-09}$ A	$+5.116628\times10^{-09}$ A
+1,000 V	$+6.656000\times10^{-07}$ A	$+5.036175\times10^{-07}$ A
+1,500 V	$+2.438000\times10^{-06}$ A	$+2.245446\times10^{-06}$ A
+2,000 V	$+5.245000\times10^{-06}$ A	$+5.223842\times10^{-06}$ A

Tabela 4.6 - Valores extraídos da curva I_{DS}xV_{GS}.

Serão apresentados agora os resultados dos testes dos transistores PMOS. As tensões usadas foram as mesmas, porém com polaridade invertida. A figura 4.23 mostra as curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do transistor PMOS 1, resultante da simulação no Cadence. A figura 4.24 mostra os resultados obtidos no laboratório.

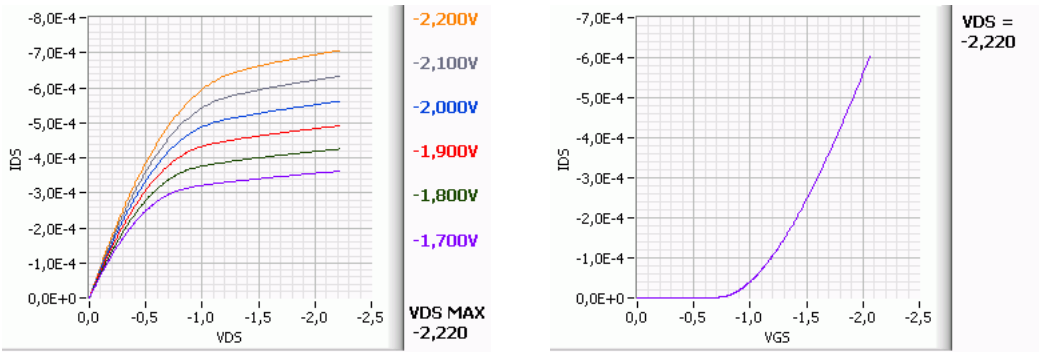


Figura 4.23 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do PMOS 1 (Simulado)

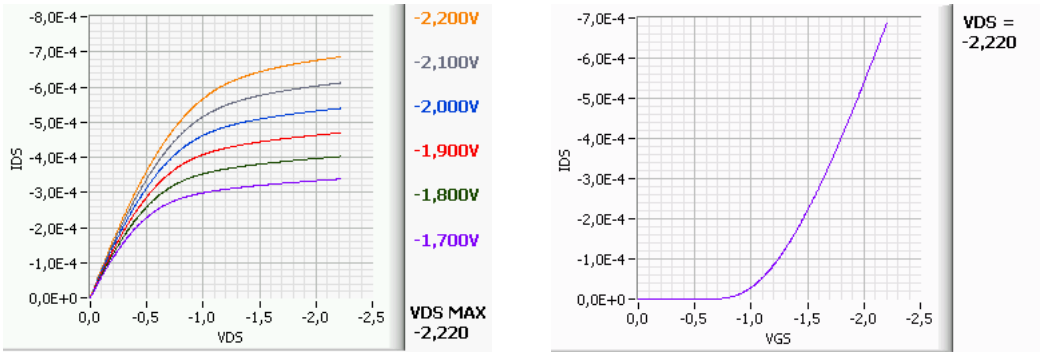


Figura 4.24 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do PMOS 1 (Medido)

O transistor PMOS apresenta resultados ainda melhores que o NMOS de mesmas dimensões. Os testes foram feitos com tensões negativas. Os eixos foram invertidos para deixar as curvas na mesma posição e facilitar o entendimento.

Para o transistor PMOS 1, os valores podem ser comparados nas tabelas 4.7 e 4.8.

I_{DS} para V_{DS}=2.0		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
-1,700 V	-3.557000$\times 10^{-04}$ A	-3.331276$\times 10^{-04}$ A
-1,800 V	-4.184000$\times 10^{-04}$ A	-3.963028$\times 10^{-04}$ A
-1,900 V	-4.836000$\times 10^{-04}$ A	-4.626578$\times 10^{-04}$ A
-2,000 V	-5.513000$\times 10^{-04}$ A	-5.315721$\times 10^{-04}$ A
-2,100 V	-6.222000$\times 10^{-04}$ A	-6.027163$\times 10^{-04}$ A
-2,200 V	-6.943000$\times 10^{-04}$ A	-6.755054$\times 10^{-04}$ A

Tabela 4.7 - Valores das curvas I_{DS}xV_{DS}.

I_{DS}xV_{GS}		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
-0,500 V	-7.392000$\times 10^{-09}$ A	-6.480892$\times 10^{-09}$ A
-1,000 V	-4.144000$\times 10^{-05}$ A	-2.910947$\times 10^{-05}$ A
-1,500 V	-2.485000$\times 10^{-04}$ A	-2.219711$\times 10^{-04}$ A
-2,000 V	-5.710000$\times 10^{-04}$ A	-5.388353$\times 10^{-04}$ A

Tabela 4.8 - Valores extraídos da curva I_{DS}xV_{GS}.

A figura 4.25 mostra as curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do transistor PMOS 2, resultante da simulação no Cadence. A figura 4.26 mostra os resultados obtidos no laboratório.

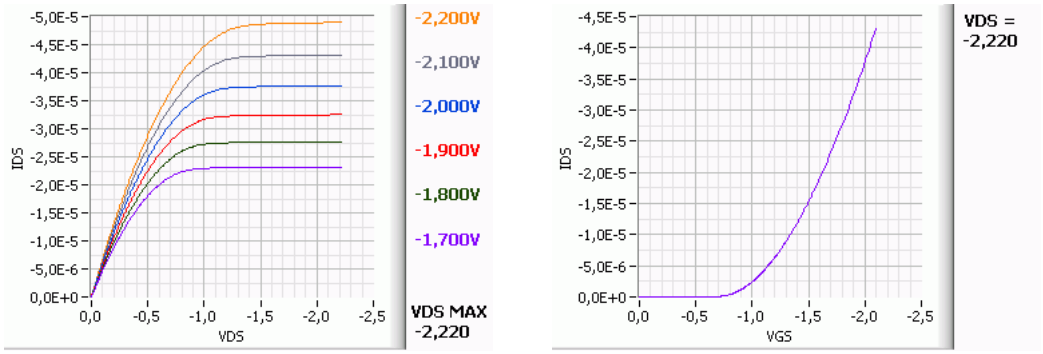


Figura 4.25 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do PMOS 2 (Simulado)

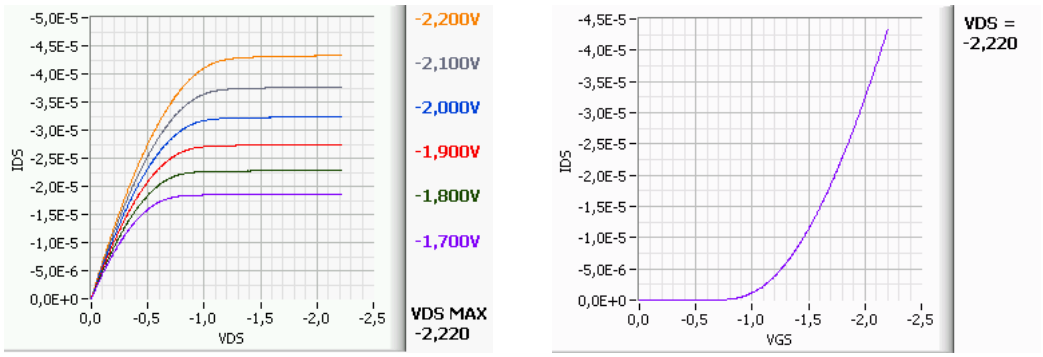


Figura 4.26 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do PMOS 2 (Medido)

Os resultados estão bem próximos. O transistor anterior está mais preciso.

Para o transistor PMOS 2, os valores podem ser comparados nas tabelas 4.9 e 4.10.

I_{DS} para V_{DS}=2.0		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
-1,700 V	-2.317000$\times 10^{-05}$ A	-1.862056$\times 10^{-05}$ A
-1,800 V	-2.766000$\times 10^{-05}$ A	-2.283333$\times 10^{-05}$ A
-1,900 V	-3.249000$\times 10^{-05}$ A	-2.741718$\times 10^{-05}$ A
-2,000 V	-3.765000$\times 10^{-05}$ A	-3.235625$\times 10^{-05}$ A
-2,100 V	-4.313000$\times 10^{-05}$ A	-3.763747$\times 10^{-05}$ A
-2,200 V	-4.891000$\times 10^{-05}$ A	-4.322182$\times 10^{-05}$ A

Tabela 4.9 - Valores extraídos das curvas I_{DS}xV_{DS}.

I_{DS}xV_{GS}		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
-0,500 V	-4.770000$\times 10^{-10}$ A	-2.167883$\times 10^{-10}$ A
-1,000 V	-2.440000$\times 10^{-06}$ A	-1.183058$\times 10^{-06}$ A
-1,500 V	-1.550000$\times 10^{-05}$ A	-1.144335$\times 10^{-05}$ A
-2,000 V	-3.785000$\times 10^{-05}$ A	-3.233038$\times 10^{-05}$ A

Tabela 4.10 - Valores extraídos da curva I_{DS}xV_{GS}.

A figura 4.27 mostra as curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do transistor PMOS 3, resultante da simulação no Cadence. A figura 4.28 mostra os resultados obtidos no laboratório.

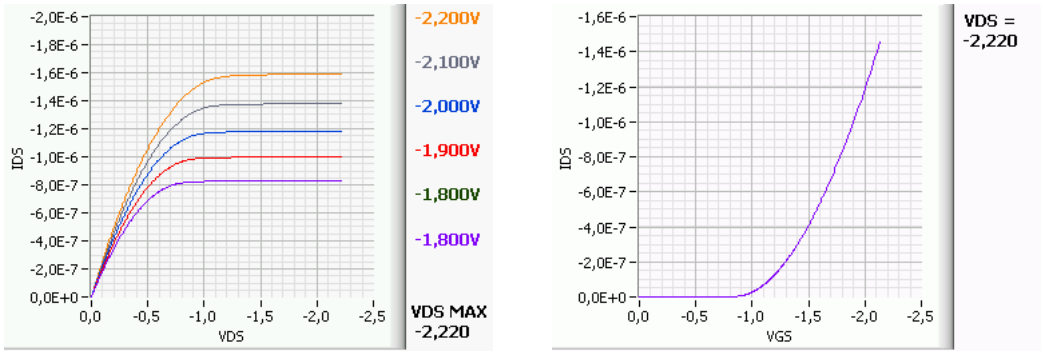


Figura 4.27 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do PMOS 3 (Simulado)

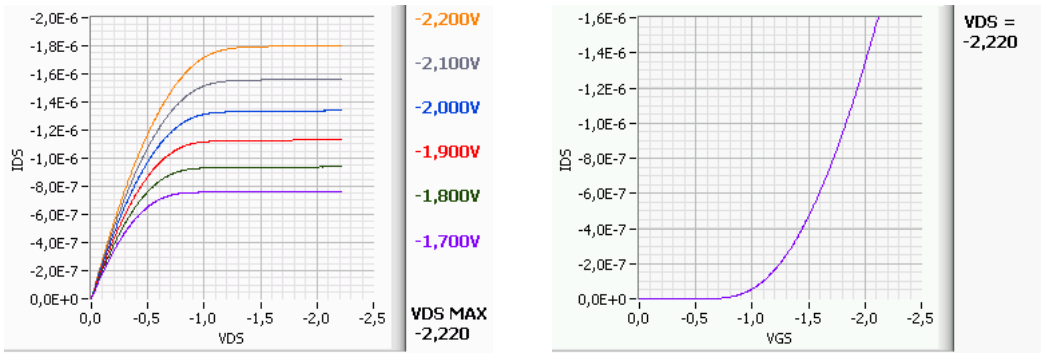


Figura 4.28 - Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ do PMOS 3 (Medido)

Os resultados estão bem parecidos. No caso do PMOS parece não haver alteração da precisão com as dimensões do transistor. Outras medidas podem ser feitas para se chegar a conclusões mais confiáveis.

Para o transistor PMOS 3, os valores podem ser comparados nas tabelas 4.11 e 4.12.

I_{DS} para V_{DS}=2.0		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
-1,700 V	-8.301000$\times 10^{-07}$ A	-7.631947$\times 10^{-07}$ A
-1,800 V	-8.301000$\times 10^{-07}$ A	-9.379158$\times 10^{-07}$ A
-1,900 V	-1.000000$\times 10^{-06}$ A	-1.129874$\times 10^{-06}$ A
-2,000 V	-1.184000$\times 10^{-06}$ A	-1.337723$\times 10^{-06}$ A
-2,100 V	-1.380000$\times 10^{-06}$ A	-1.561663$\times 10^{-06}$ A
-2,200 V	-1.590000$\times 10^{-06}$ A	-1.800061$\times 10^{-06}$ A

Tabela 4.11 - Valores extraídos das curvas I_{DS}xV_{DS}.

I_{DS}xV_{GS}		
V_{GS}	I_{DS} SIMULADO	I_{DS} MEDIDO
-0,500 V	-2.392000$\times 10^{-12}$ A	-1.092096$\times 10^{-10}$ A
-1,000 V	-2.798000$\times 10^{-08}$ A	-5.589822$\times 10^{-08}$ A
-1,500 V	-4.122000$\times 10^{-07}$ A	-4.706414$\times 10^{-07}$ A
-2,000 V	-1.190000$\times 10^{-06}$ A	-1.337764$\times 10^{-06}$ A

Tabela 4.12 - Valores extraídos da curva I_{DS}xV_{GS}.

4.2 MEDIDAS DINÂMICAS EM *BUFFERS*

O objetivo deste trabalho é verificar o funcionamento e a resposta em frequência dos *buffers* inversores e não inversores projetados para o SoC do projeto Milênio [4]. Os protótipos FAPESP 112 e 119 possuem *buffers* isolado com *pads* para teste. Um gerador de funções é usado para fornecer o sinal de entrada. O sinal aplicado na entrada é comparado com o sinal de saída através de um osciloscópio digital de dois canais. Assim pode-se observar com detalhes o comportamento do circuito em várias frequências.

4.2.1 Sistema de Medidas

O gerador de funções usado é o HP 3310A, mostrado na figura 4.29. O Keithley 2400 é usado para alimentar o circuito. O osciloscópio digital de dois canais HP 54600A pode ser visto na figura 4.30. O osciloscópio é ligado a um computador usando uma interface GPIB.

Dos trabalhos apresentados, apenas neste não foi usado o LabVIEW. Os instrumentos foram configurados manualmente. As curvas foram capturadas pelo programa HP BenchLink®[4] e salvas no formato de figura. O osciloscópio usado, por ser um instrumento mais antigo, não possui o protocolo GPIB padronizado pelo IEEE. Usa o protocolo original HP-IB criado pela Hewlett-Packard®. O programa usado apenas captura dados do instrumento e salva no formato de tabelas ou figuras.

Figura 4.29 - Gerador de Funções HP 3310A

Figura 4.30 - Osciloscópio HP 54600A

Os testes foram feitos em *chips* não encapsulados usando a bancada de teste com posicionadores de precisão e agulhas. Cabos coaxiais e conectores BNC são usados para melhor acoplamento e maior resistência à ruídos externos.

4.2.2 Descrição do Procedimento

O Keithley 2400 é usado como fonte de alimentação para o circuito e ajustado em cinco volts. Através dele pode-se verificar o consumo de corrente. O gerador de funções é ajustado para gerar uma senóide. O teste é feito nas frequências de 1 kHz, 100 kHz e 3 MHz. Um adaptador T é conectado à saída de maneira que se possa ligar o sinal ao *buffer* e também ao canal 1 do osciloscópio. A saída do *buffer* é ligada ao canal 2 do osciloscópio. Na figura 4.31 pode-se ver o diagrama de ligação.

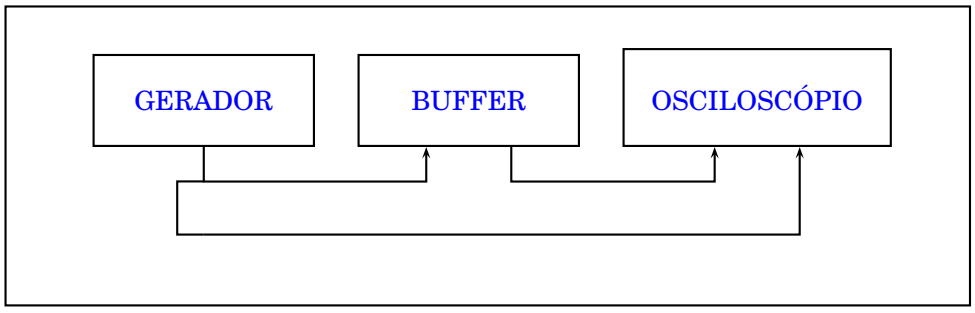


Figura 4.31 - Diagrama em Blocos

Após ajustar adequadamente o gerador de funções e o osciloscópio para que os dois sinais fiquem visíveis na tela, usa-se o programa HP BenchLink para salvar as formas de ondas.

4.2.3 Medidas, Resultados e Análise de Desempenho

4.2.3.1 Os *buffers* do FAPESP 112

São cinco *buffers* que compartilham os *pads* de alimentação. A figura 4.32 mostra a localização no *chip* e também os pinos de alimentação. A figura 4.33 identifica os pinos de entrada *En* e saída *Sn* de cada circuito. Os *pads E1* e *S1* correspondem ao *buffer* 1. Os *pads E2* e *S2* são respectivamente entrada e saída do *buffer* 2. E assim até o *buffer* 5. Os *buffers* 2 e 4 são inversores. Os *buffers* 1, 3 e 5 não são inversores.

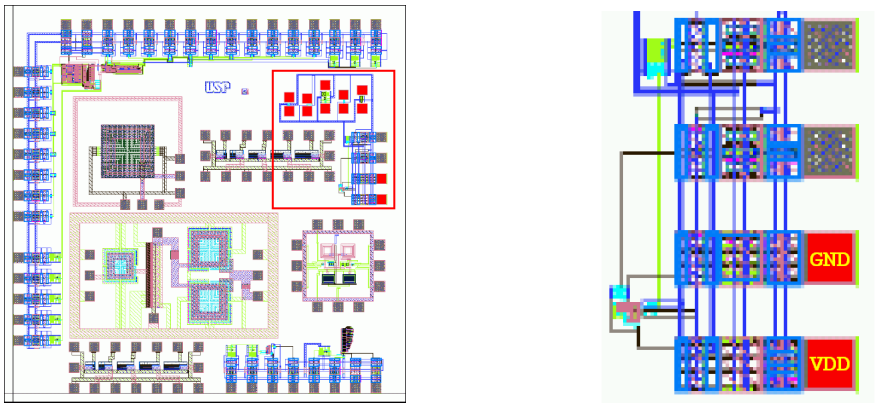


Figura 4.32 - Localização no FAPESP 112

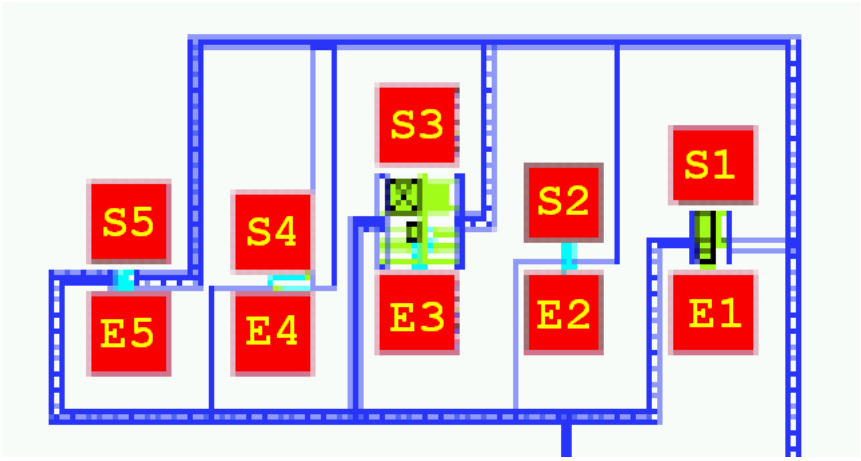


Figura 4.33 - *Buffers*, *pads* de entrada e saída

4.2.3.2 Medidas e resultados

Os *buffers* devem transformar um sinal senoidal em onda quadrada. As frequências usadas são 1 kHz, 100 kHz e 3 MHz. As figuras 4.34 e 4.35 apresentam as respostas do *buffer* 1, que é do tipo não inversor, para as frequencias de 1 kHz e 100 kHz, respectivamente. Foi aplicado uma senóide com 3,2 Vpp. Em 1 kHz, o *buffer* funciona. Porém existe fuga da entrada para a saída. O sinal de saída apresenta deformações e interfere no sinal de entrada.

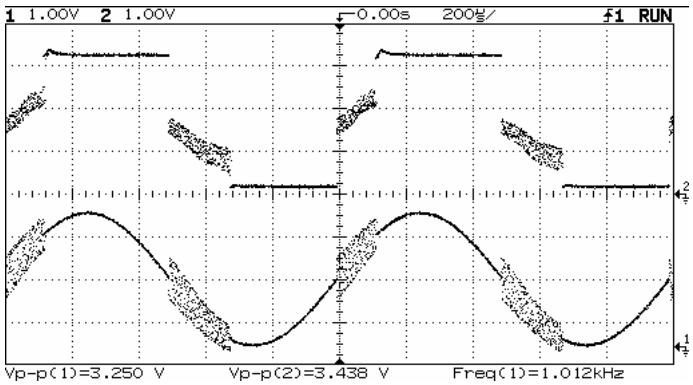


Figura 4.34 - *Buffer* 1 na frequência de 1 kHz

Em 100 kHz, pode-se observar melhor o efeito. O circuito oscila pois a fuga provoca uma realimentação.

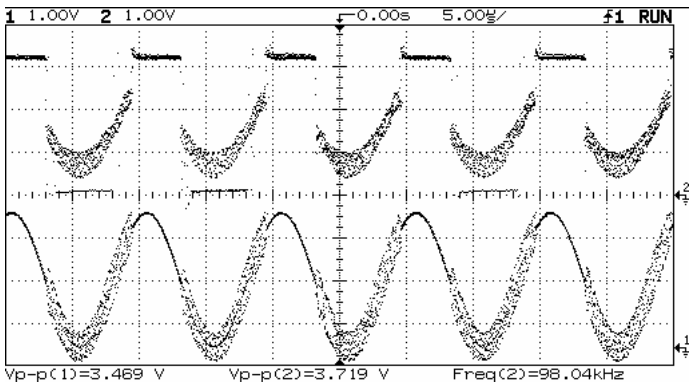


Figura 4.35 - *Buffer* 1 na frequência de 100 kHz

A figura 4.36 apresenta a resposta do *buffer* 1 em 3 GMz. Para essa frequência interferência entre a entrada e a saída é desprezível. Porém o sinal de entrada aparece na saída com um defasamento devido à um acoplamento capacitivo parasita.

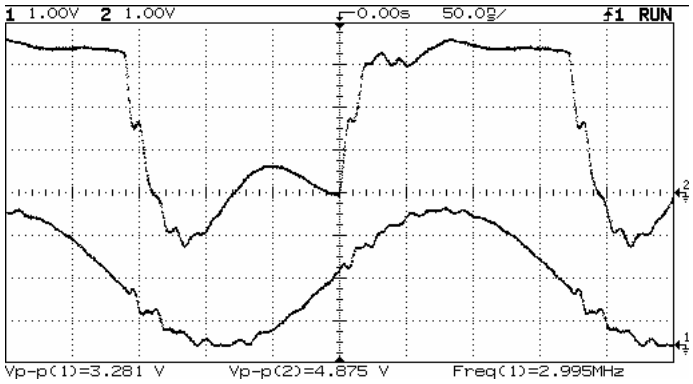


Figura 4.36 - *Buffer* 1 na frequência de 3 MHz

O mesmo teste foi aplicado ao *buffer* 2. Trata-se de um *buffer* inversor. O efeito de interferência é mínimo pois os sinais de entrada e saída estão defasados em 90 graus. Uma pequena deformação aparece na parte superior do sinal de saída. Deveria ser constante, mais acompanha a curva da tensão de entrada no semiciclo negativo. A figura 4.37 mostra o resultado para a frequência de 1 kHz.

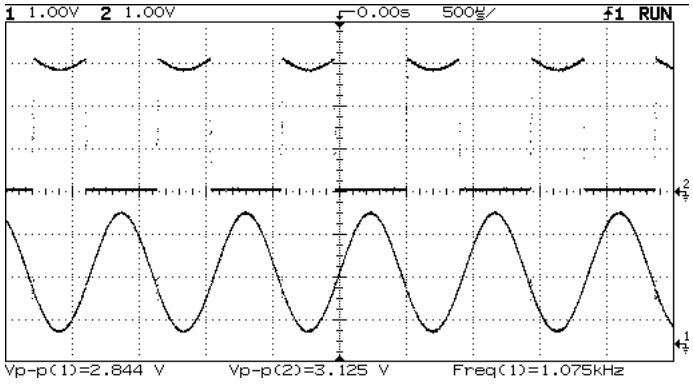


Figura 4.37 - *Buffer* 2 na frequência de 1 kHz

O desempenho do circuito melhora em 100 kHz. O sinal de entrada não interfere no sinal de saída. O resultado pode ser visto na figura 4.38.

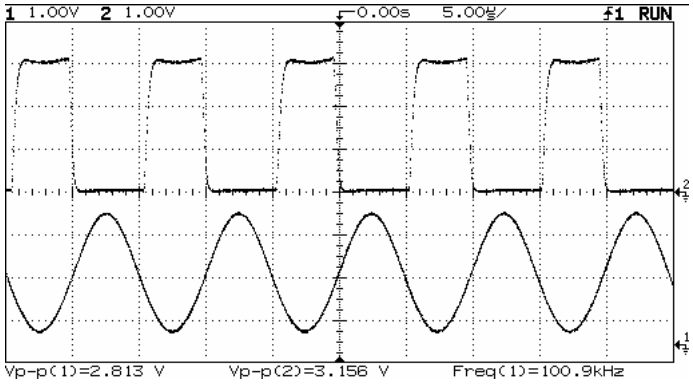


Figura 4.38 - *Buffer* 2 na frequência de 100 kHz

Em frequências mais elevadas o sinal de saída é atenuado. Devido ao acoplamento capacitivo parasita com a entrada, a tensão de saída é sobreposta ao sinal senoidal do gerador. Estando este invertido a interferência é destrutiva. O resultado pode ser visto na figura 4.39.

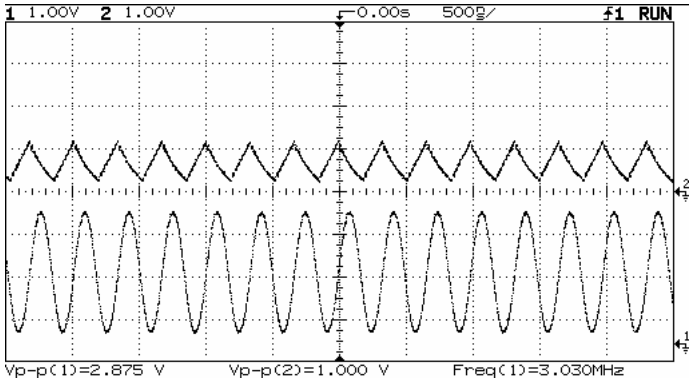


Figura 4.39 - *Buffer* 2 na frequência de 3 MHz

O teste foi aplicado também ao *buffer* 3, que é do tipo não inversor. As interferências entre entrada e saída são bem menores do que no *buffer* 1. O circuito apresenta um melhor desempenho. A figura 4.40 mostra o desempenho em 1 kHz.

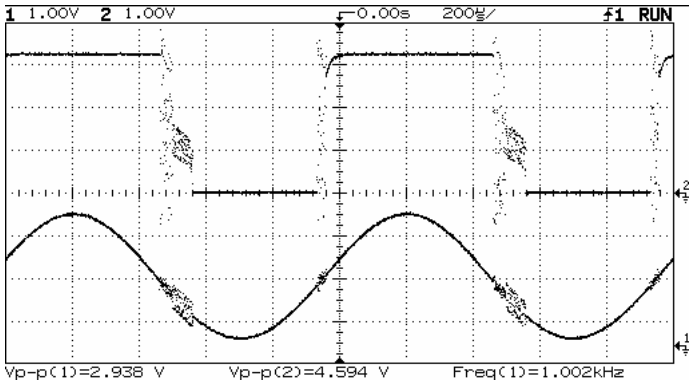


Figura 4.40 - *Buffer* 3 na frequência de 1 kHz

A figura 4.41 mostra o desempenho em 100 kHz. Na figura 4.42, pode-se ver novamente o efeito observado no *buffer* 1. Em 3 MHz, o sinal de entrada aparece na saída com um defasamento devido à um acoplamento capacitivo parasita.

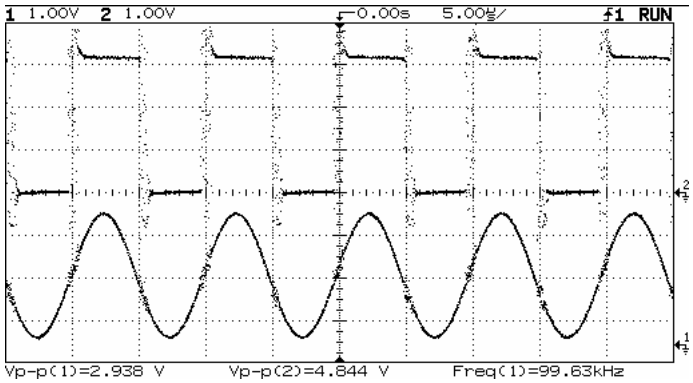


Figura 4.41 - *Buffer* 3 na frequência de 100 kHz

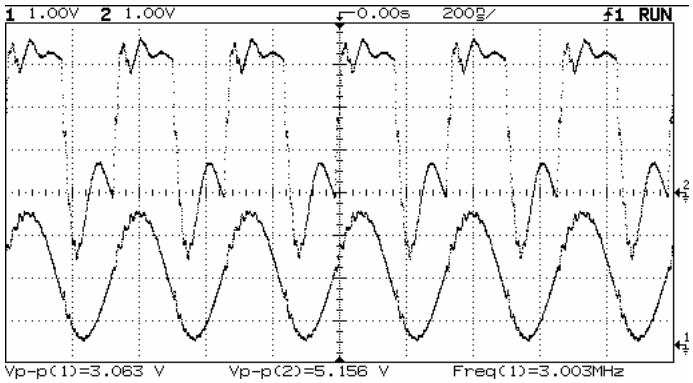


Figura 4.42 - *Buffer* 3 na frequência de 3 MHz

A figura 4.43 mostra o *buffer* 4, que é do tipo inversor, operando em 1 kHz. O sinal é perfeito. O circuito apresenta excelente desempenho. A figura 4.44 mostra o *buffer* operando em 100 kHz. Existe uma pequena deformação no sinal de saída. Mais isso não é significativo pois não compromete o funcionamento do sistema.

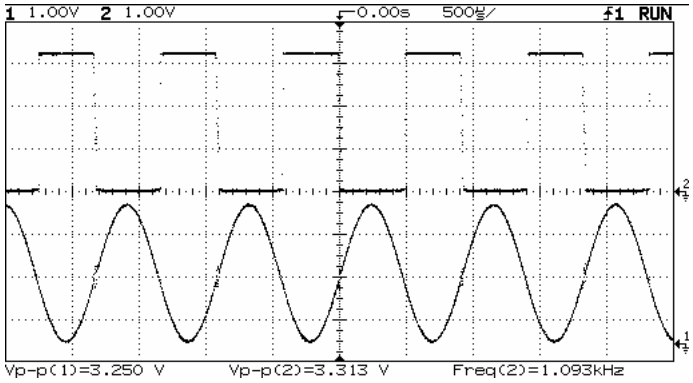


Figura 4.43 - *Buffer* 4 na frequência de 1 kHz

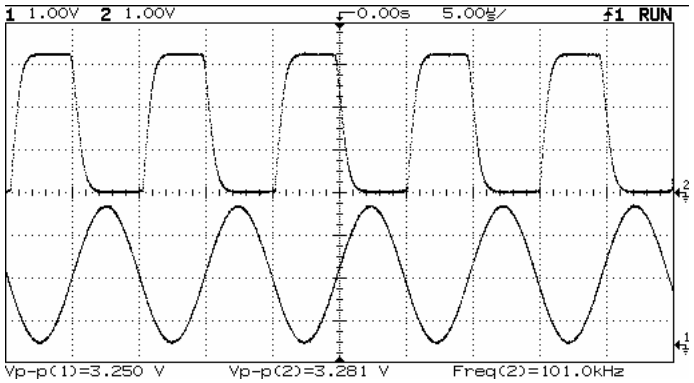


Figura 4.44 - *Buffer* 4 na frequência de 100 kHz

A figura 4.45 mostra o *buffer* operando em 3 MHz. Verifica-se o mesmo problema que apareceu no *buffer* 2. Devido ao acoplamento capacitivo parasita com a entrada, a tensão de saída é sobreposta ao sinal da entrada. Estando este invertido a interferência é destrutiva. No *buffer* 4, a atenuação é ainda maior.

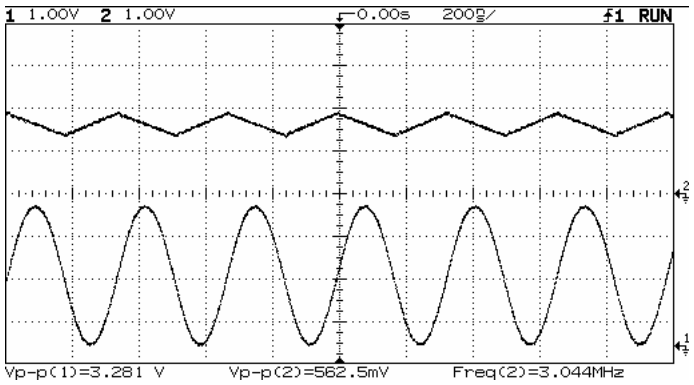


Figura 4.45 - *Buffer* 4 na frequência de 3 MHz

O teste agora é aplicado ao *buffer* 5, que é do tipo não inversor. A figura 4.46 apresenta a resposta à frequência de 1 kHz. O sinal é perfeito.

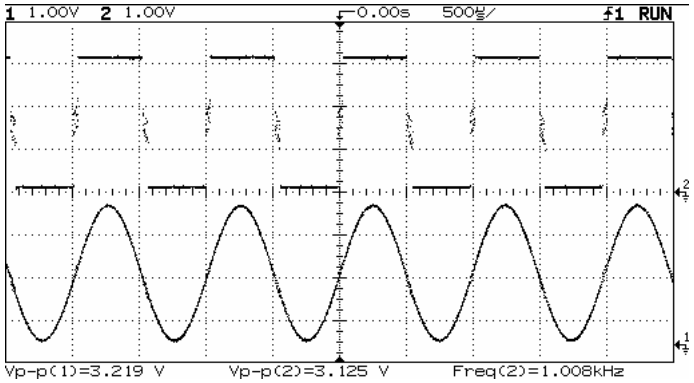


Figura 4.46 - *Buffer* 5 na frequência de 1 kHz

A figura 4.47 mostra o teste com a frequência de 100 kHz. O circuito apresenta excelente desempenho.

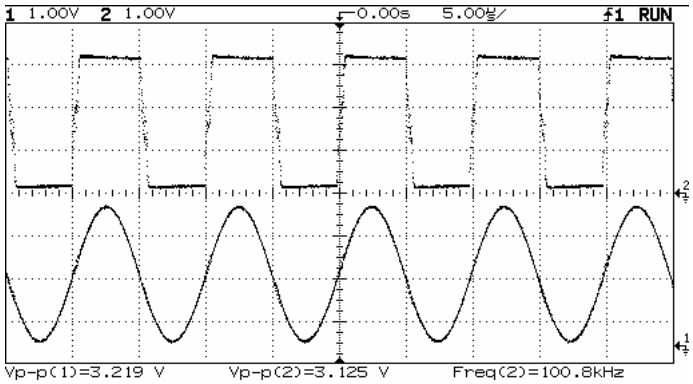


Figura 4.47 - *Buffer* 5 na frequência de 100 kHz

A figura 4.48 mostra a resposta ao sinal de 3 MHz. O sinal apresenta distorções devido à capacitâncias parasitas. Nesse caso não há acoplamento entre a entrada e a saída. As capacitâncias estão situadas entre a saída e o terra.

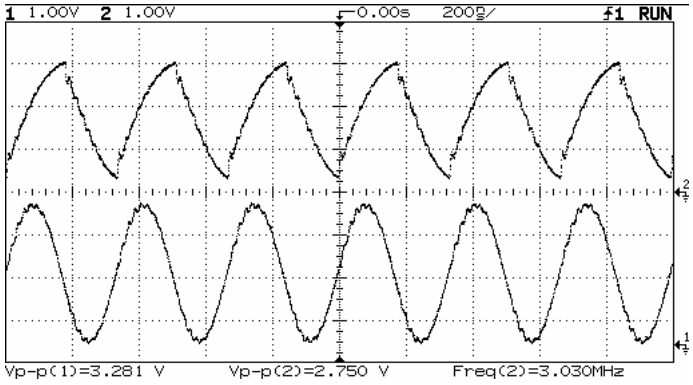


Figura 4.48 - *Buffer* 5 na frequência de 3 MHz

4.2.3.3 O *buffer* do FAPESP 119

Esse *chip* possui um *buffer* inversor com *pads* isolados para teste. A figura 4.49 mostra a localização no *chip* e também os pinos de alimentação. Estão destacados os pinos de alimentação *VDD* e *GND* e também o pino de entrada de sinal *IN* e o pino de saída *OUT*.

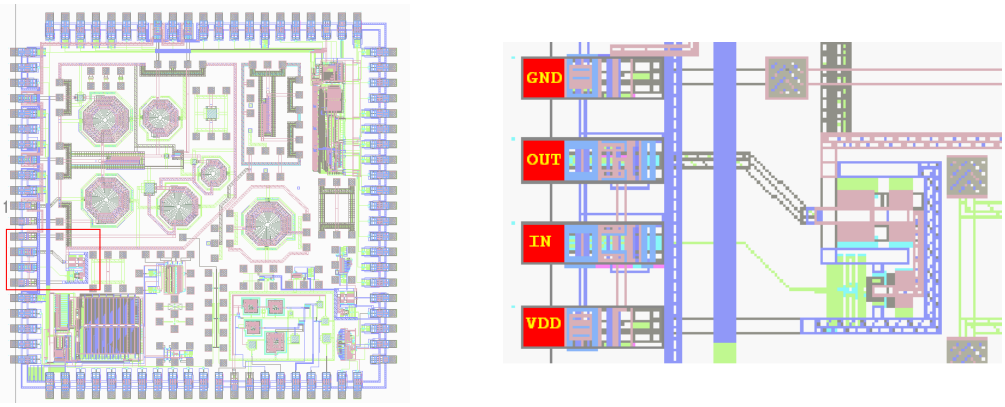


Figura 4.49 - Localização no FAPESP 119

4.2.3.4 Medidas e resultados

O *buffer* deve transformar um sinal senoidal em onda quadrada. As figuras 4.50, 4.51 e 4.52 apresentam os resultados do *buffer*, que é do tipo inversor. Foi aplicado uma senóide com 3,2 Vpp. As frequências usadas são 1 kHz, 100 kHz e 3 MHz. Em 1 kHz circuito funciona. Porém, o sinal de entrada interfere na saída.

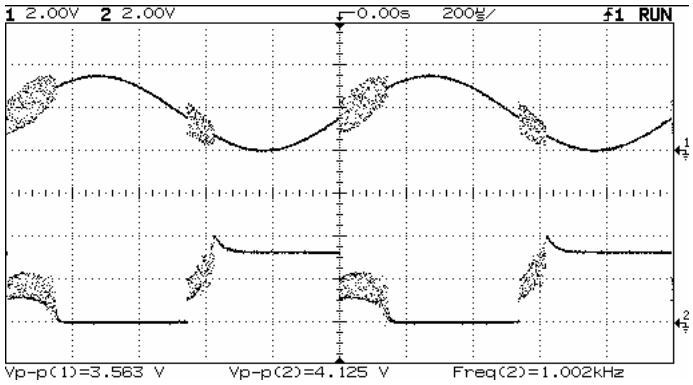


Figura 4.50 - *Buffer* na frequência de 1 kHz

O mesmo acontece para a frequência de 100 kHz. O sinal de entrada continua interferindo na saída. O problema deve ser baixa isolação.

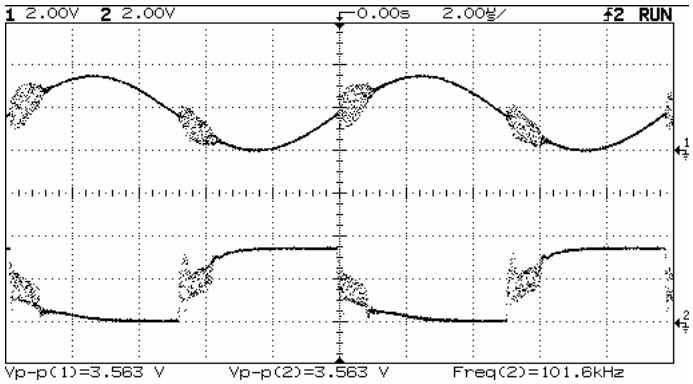


Figura 4.51 - *Buffer* na frequência de 100 kHz

Em 3 MHz, os sinais de entrada e saída se misturam. A senóide chega à saída com atraso de maneira que é somado à onda quadrada. O resultado é um sinal deformado e com amplitude maior que a entrada.

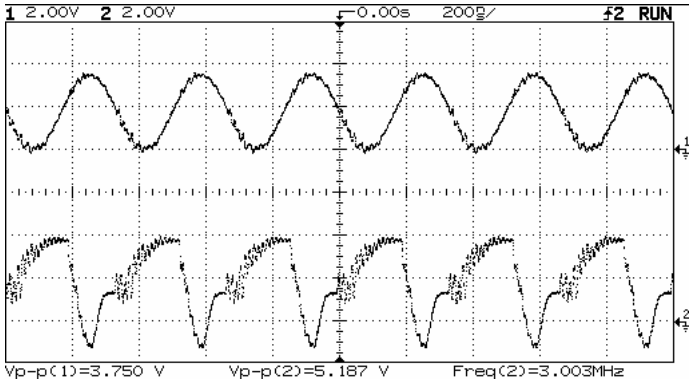


Figura 4.52 - *Buffer* na frequência de 3 MHz

4.3 PARÂMETROS DE ESPALHAMENTO

Conforme estudado anteriormente, é conveniente caracterizar dispositivos e circuitos de RF através dos parâmetros S. Os procedimentos convencionais tornam-se imprecisos quando se trabalha em alta frequência. Nesse caso, fica mais fácil trabalhar com os parâmetros S usando instrumentos como o analisador de redes. Obtendo os parâmetros S pode-se calcular outras características de um DUT, como potência, impedância, ganho, e atenuação.

4.3.1 Sistema de Medidas

O principal instrumento usado é o analisador de redes Agilent 8714ES [14], [30]. Este instrumento pode ser visto na figura 4.53. No apêndice A encontra-se uma descrição mais detalhada de suas características. É importante o uso de cabos coaxiais e conectores tipo N ou APC para perfeito casamento de impedância e minimização das perdas e interferências. A estação de medidas de RF Cascade é usada no caso de testes em *chips* não encapsulados.

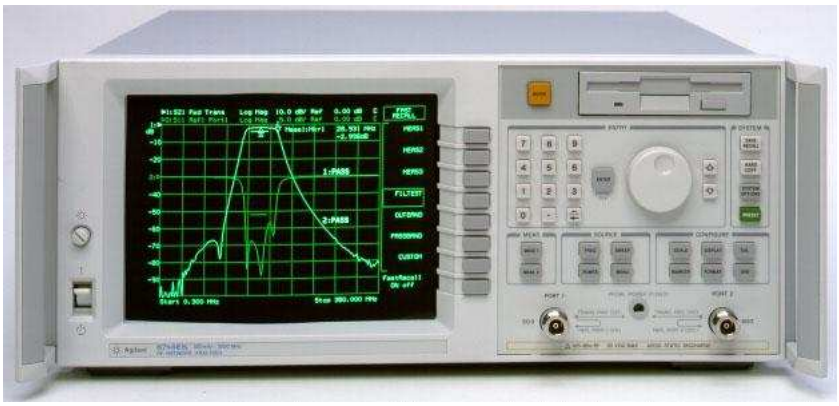


Figura 4.53 - Analisador de redes

4.3.2 Descrição do Procedimento

Em baixas frequências as perdas e fugas nos conectores e cabos são irrelevantes na maioria dos casos. Por isso são desconsideradas. Porém, quando se mede em RF essas distorções conduzem a resultados errôneos e inviabilizam as medidas. Portanto o procedimento foi dividido em duas etapas. Primeiramente, fazemos a calibração do instrumento, e depois usamos programas para capturar os dados medidos via GPIB.

4.3.2.1 Calibração do Instrumento

Devido aos problemas apresentados no item 2.2, é muito importante uma calibração adequada do instrumento na faixa de frequência em que será feito os testes. Quanto menor a faixa de frequência, melhor a precisão das medidas. O manual do instrumento [14] apresenta vários processos passo a passo de calibração para DUTs com conectores tipo N. Basicamente, o processo consiste em usar cargas conhecidas que são conectadas nas extremidades dos cabos ou diretamente nas portas do instrumento. As cargas, fornecidas pelo fabricante, simulam circuito aberto, curto e impedância casada de 50Ω. O instrumento, quando na função calibração, indica o que deve ser conectado nos seus terminais e quando a correção dos erros repetíveis é concluída.

Para os testes usando a estação de medidas de RF, deve ser usado o padrão de testes Cascade para as pontas de provas. É uma placa que possui trilhas que fornecem cargas, curto e transferência direta entre ponteiras. Como essa estação usa conectores do tipo APC 3,5 mm, esta opção deve ser escolhida durante a calibração. Como as portas do instrumento são conectores tipo N fêmeas, usam-se dois adaptadores. A figura 4.54 mostra a tela do instrumento com a opção selecionada.

Port #	System Z0	Cal Kit	Port (m/f)
Port 1	50	3.5 mm	m/f
Port 2	50	3.5 mm	m/f

Figura 4.54 - Seleção de conectores

4.3.2.2 Capturando dados via GPIB

Foram desenvolvidos dois programas em LabVIEW para configurar e adquirir os parâmetros. O primeiro programa mostra as curvas S_{11} e S_{21}, e o segundo, as curvas S_{22} e S_{12}. Os programas seguem o modelo padrão proposto.

Os resultados são salvos em arquivos de texto ASCII. Podem ser reabertos pelos programas que os criaram ou exportados para programas como o MATLAB. As figuras 4.55 e 4.56 mostram as telas dos programas.

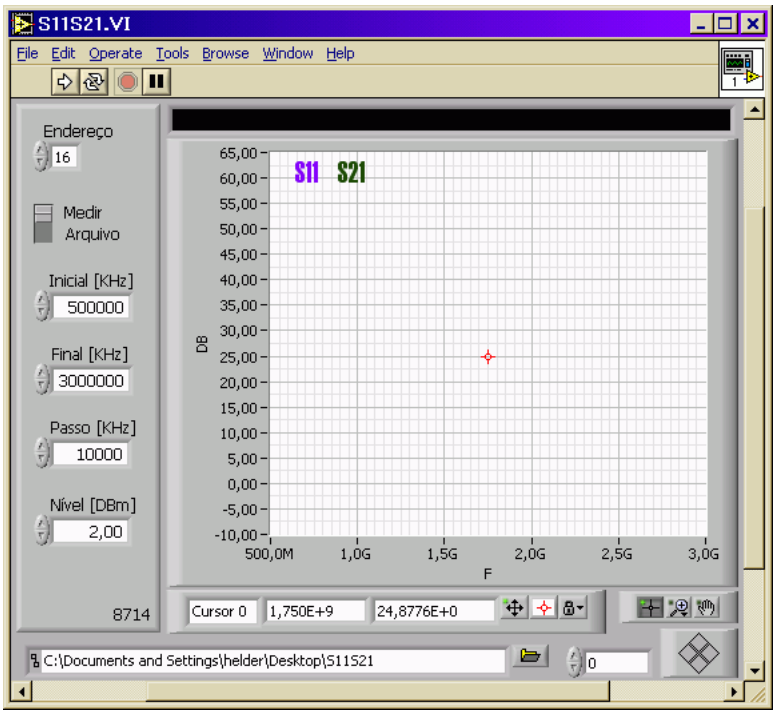


Figura 4.55 - Programa para obtenção de S_{11} e S_{21}

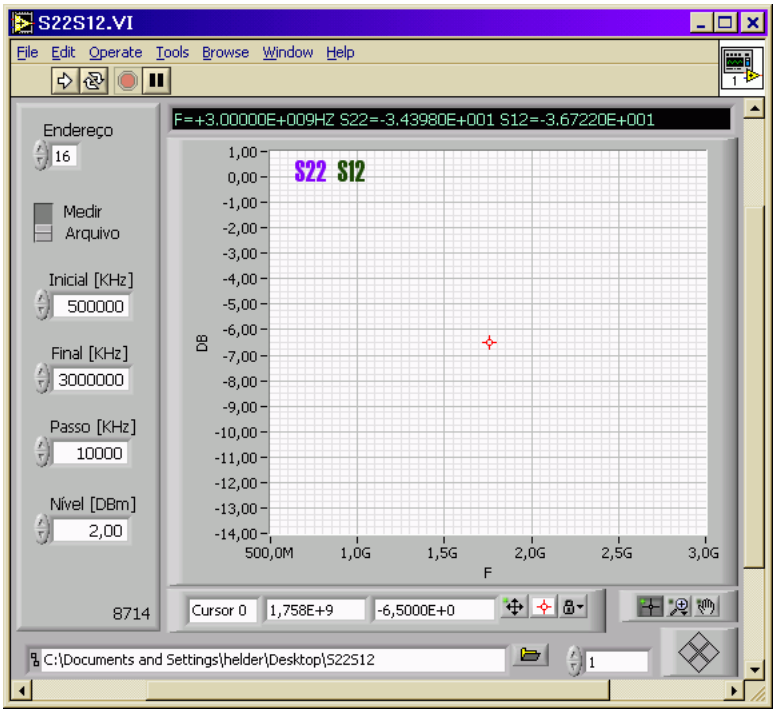


Figura 4.56 - Programa para obtenção de S_{22} e S_{12}

Em ambos os programas, o usuário deve fornecer as frequências inicial e final e o passo para que o instrumento faça a varredura. O nível de potência de saída em dBm e o nome do arquivo para guardar os resultados também são escolhidos pelo usuário. O analisador de redes é por padrão acessado no endereço GPIB 16. O endereço pode ser facilmente alterado caso necessário.

O funcionamento dos programas é simples. Primeiramente, o instrumento é configurado para traçar as curvas desejadas. A frequência inicial é sintonizada e uma leitura dos parâmetros medidos é feita. Os resultados são armazenados na memória RAM do computador. Em seguida, a frequência é incrementada no valor do campo *Passo* fornecido pelo usuário. O processo se repete até atingir a frequência final. Os resultados são salvos em arquivos e depois exibidos no gráfico. A figura 4.57 mostra o fluxograma do programa. O código completo dos programas está descrito no apêndice F.

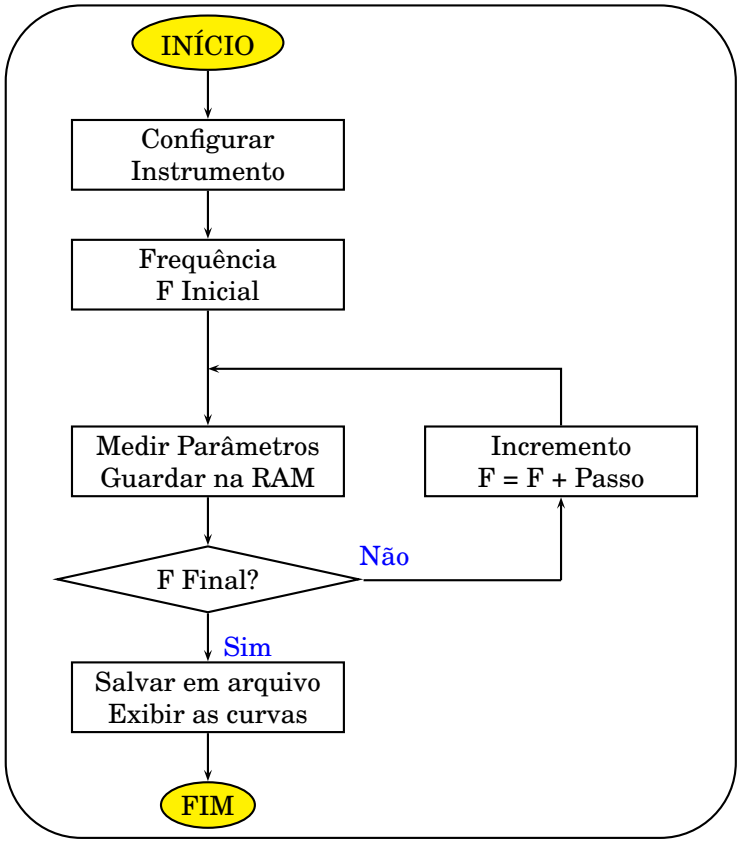


Figura 4.57 - Processo para obtenção de parâmetros S

4.3.3 Medidas, Resultados e Análise de Desempenho

4.3.3.1 Teste no Indutor

No *chip* FAPESP 119, um indutor de RF isolado foi construído para ser caracterizado. Seu valor é calculado através dos parâmetros de espalhamento [15]. As medidas são realizadas na estação de medidas Cascade. A figura 4.58 mostra o indutor com seus *pads* internos. A faixa de frequência usada é de 800 MHz a 1 GHz. As figuras 4.59 e 4.60 mostram as curvas obtidas. O cursor está posicionado na frequência de 915 MHz.

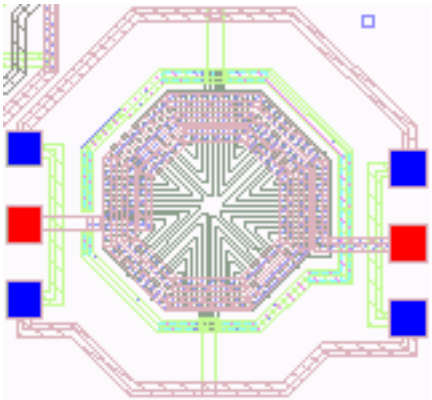


Figura 4.58 - Indutor de RF

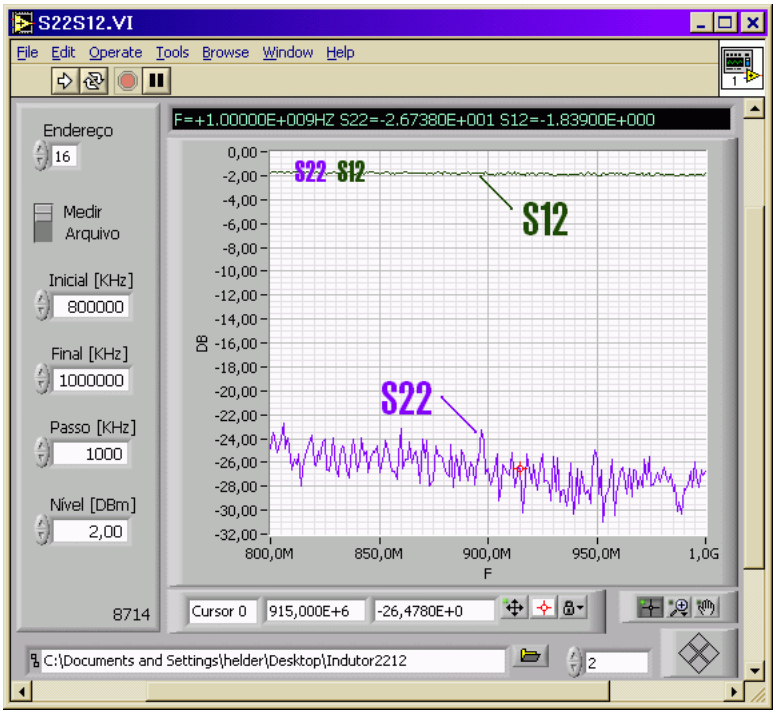


Figura 4.59 - Parâmetros S_{22} e S_{12} do indutor

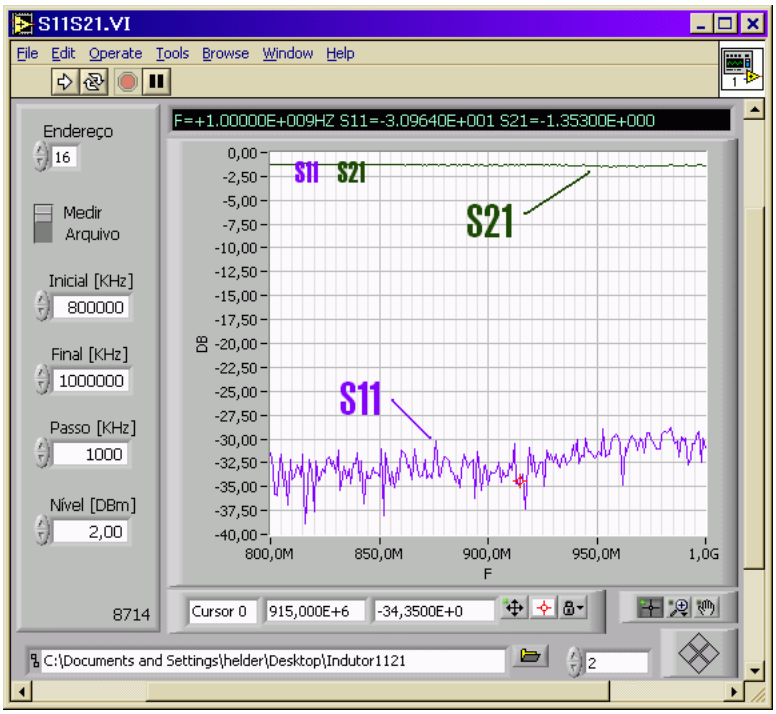


Figura 4.60 - Parâmetros S_{11} e S_{21} do indutor

Para calcular o valor da indutância, usa-se a equação (2.17), substituíndo Z_1 por Z_L:

$$Z_L = Z_0 \times \frac{1 + S_{11}}{1 - S_{11}} \quad , \tag{4.1}$$

onde Z_L é a impedância de entrada vista pela porta 1. No nosso caso é a impedância do indutor, i.e., Z_L=ωL. Z_0 é a impedância característica do sistema, que é de 50Ω. Como se pode ver, conhecendo o valor de S_{11} para uma determinada frequência, pode-se calcular o valor da indutância. Na figura 4.60, para a frequência de 915 MHz, o S11=-34,35db.

$$S11 = 10^{-34,35/10} \quad \Rightarrow \quad S11 = 3,6728 \times 10^{-4}$$

Pode-se usar esse valor para calcular a impedância:

$$Z_L = 50 \times \frac{1 + 3,6728 \times 10^{-4}}{1 - 3,6728 \times 10^{-4}} \quad \Rightarrow \quad Z_L = 50,0367$$

Para a frequência de 915 MHZ:

$$L = \frac{Z_L}{\omega} \quad \Rightarrow \quad L = 8,70nH$$

4.3.3.2 Teste em um Filtro Ressonante

O filtro foi construído em placas CER-10, de fabricação TACONIC. Esta placa apresenta constante dielétrica ε_r=10, espessura do dielétrico h=1,57 mm. A figura 4.61 [10] mostra o dispositivo construído. Os conectores são do tipo APC.

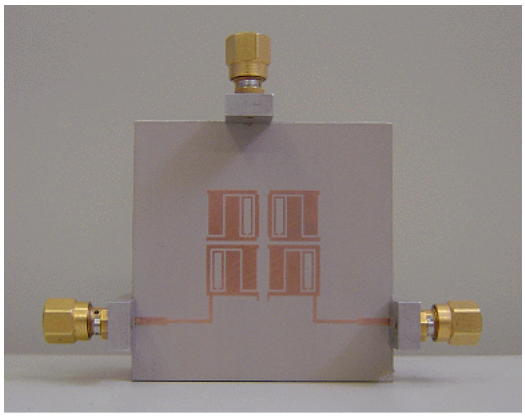


Figura 4.61 - Filtro Ressonante

O filtro testado possui ressonância na banda de 900 MHz. A banda de 900 MHz é utilizada em comunicações móveis. A figura 4.62 mostra o programa com os resultados obtidos.

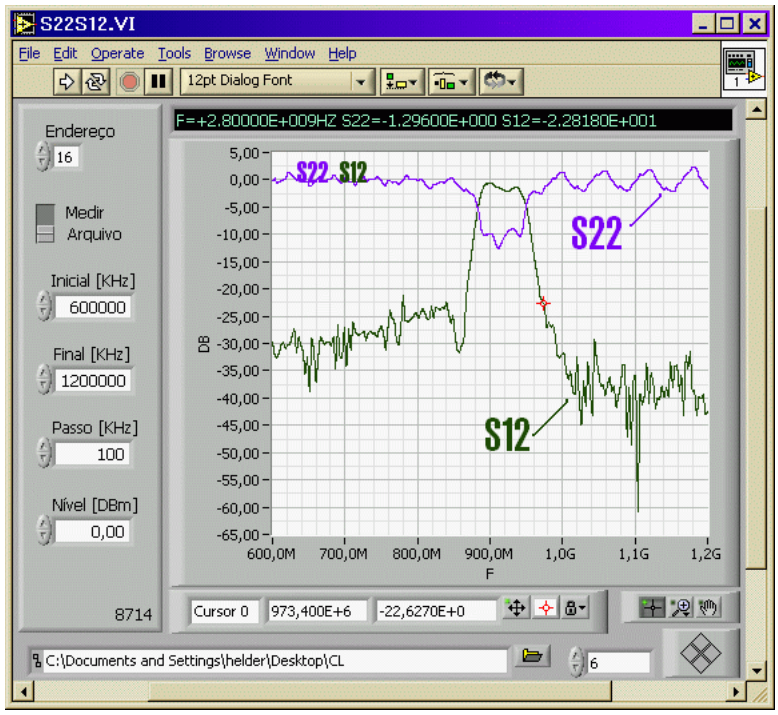


Figura 4.62 - Parâmetros S_{12} e S_{22} do filtro

O filtro funciona, apresentando baixa atenuação na faixa de frequência desejada. Pode-se, então, verificar a funcionalidade do modelo de caracterização.

A dissertação “Um novo filtro com banda passante dupla utilizando ressoadores miniaturizados” [10], apresenta a teoria, o projeto e compara os resultados medidos com valores obtidos por simuladores.

5 DISCUSSÃO

Este capítulo apresenta uma relação das atividades realizadas no laboratório. Inclui instalação e configuração de programas, equipamentos e acessórios.

17/06/2004	Instalação do LabVIEW
19/07/2004	Teste de transistores MOS do *chip* N-MOS CCS
20/07/2004	Teste de transistores MOS do *chip* N-MOS CCS
26/07/2004	Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ no *chip* N-MOS CCS
16/08/2004	Medidas na circular do N-MOS CCS
17/08/2004	Medida dinâmica do NMOS CCS
26/08/2004	Curvas $I_{DS}xV_{DS}$ e $I_{DS}xV_{GS}$ no FAPESP 119
24/09/2004	Medida no Indutor do *chip* FAPESP 119
30/09/2004	Teste no transistor de RF do FAPESP 119
14/10/2004	Medidas dinâmicas nos *Buffers* do FAPESP 112
01/11/2004	Teste das portas lógicas e do somador no FAPESP 112
11/11/2004	Teste dinâmico da ULA com o Analisador Lógico [4]
08/12/2004	Teste do LNA do FAPESP 119
08/01/2005	Testes nos capacitores do FAPESP 119
12/01/2005	Teste digital no FAPESP 119
13/01/2005	Teste dos MUX 8 e MUX 4 no FAPESP 112 [4]
14/01/2005	Teste das portas lógicas do FAPESP 112 [4]
14/01/2005	Medidas na alimentação digital VDD e GND do FAPESP 119
15/01/2005	Teste nos pinos alimentação do FAPESP 119
17/01/2005	Teste na alimentação digital com tensão baixa
18/01/2005	Teste com o Eletrômetro
18/01/2005	Avaliação do Keithley 2400 como fonte de alimentação

19/01/2005	Teste dinâmico da ULA do FAPESP 119
20/01/2005	Teste da ULA de 1 BIT do FAPESP 112 [4]
26/01/2005	Teste do divisor de frequências 3334
02/02/2005	Novo teste dos MUX 8 e MUX 4 no FAPESP 112 [4]
11/04/2005	Medidas no capacitor para RF com o LCR
03/06/2005	Curvas de um diodo zener de 6,8 V
05/07/2005	Análise das junções de leds
11/07/2005	Ruptura da porta dos transistores MOS
15/07/2005	Curvas IV de junções e substrato dos transistores MOS
16/07/2005	Análise das junções dos diodos zeners 1N4739 (9,1 V)
22/07/2005	Análise de ruptura nas junções de transistores MOS
08/08/2005	Caracterização de dispositivos orgânicos [31]
05/09/2005	Teste em transistor do FUNCAMP_SOC
24/09/2005	Medida na ROM do FUMCAMP_SOC
26/09/2005	Teste do decodificador do FUMCAMP_SOC
30/09/2005	Teste no conversor serial/paralelo
03/10/2005	Medida no conversor serial/paralelo
10/03/2005	Novas medidas no conversor serial/paralelo
18/10/2005	Caracterização do transistor isolado do Funcamp_SOC
24/10/2005	Medida na RAM do Funcamp_SOC
24/10/2005	Teste da ROM do Funcamp_SOC
09/03/2006	Calibração do Analisador de Redes
09/03/2006	Teste com o atenuador
15/05/2006	Teste no PA do FAPESP 119 e Funcamp_SOC [4], [28]
22/05/2006	Teste no transistor isolado do Fumcamp_SOC
24/05/2006	Medida estática do transistor de RF [28]
08/06/2006	Medida estática do transistor de RF na Cascade [28]
20/03/2007	Instalação e teste do Analisador de Espectro R&S FSL [32],[33]
02/06/2007	Instalação e teste das interfaces USB-GPIB

Nesse quadro está o conjunto das contribuições desenvolvidas para a caracterização de dispositivos e circuitos integrados no laboratório LTSD ao longo deste trabalho de mestrado. Todos os procedimentos listados no quadro acima encontram-se operacionais, disponíveis e documentados.

Além dos trabalhos de medidas, foi ministrado um curso sobre Testes e Medidas usando LabVIEW e GPIB. O curso inclui um treinamento de programação com essa ferramenta com o desenvolvimento de aplicativos práticos. Em seguida, são fornecidas informações teóricas sobre o protocolo GPIB, e comunicação com instrumentos de medidas. São realizadas também, experiências práticas com a construção de programas que se comunicam com instrumentos e realizam medidas.

6 CONCLUSÃO

Implantamos um conjunto de metodologias e procedimentos para a caracterização elétrica de dispositivos eletrônicos e circuitos integrados. Os procedimentos foram todos documentados e podem servir de referência para projetos futuros.

Construímos uma estrutura de caracterização elétrica de dispositivos eletrônicos e circuitos integrados que está em operação. Vários instrumentos estão interligados via barramento GPIB e estão sendo controlados por aplicativos criados no LTSD. Isso torna a estrutura bastante versátil, possibilitando testar e caracterizar uma grande variedade de circuitos. Os testes podem ser automatizados com o controle simultâneo de vários instrumentos por programas.

A maioria dos dispositivos e circuitos integrados implementados no laboratório no período de desenvolvimento da dissertação foram caracterizados. Alguns procedimentos não puderam ser completados por dificuldades burocráticas na importação de acessórios para os instrumentos de medidas. Também foi implantado e executado treinamentos em caracterização elétrica com o auxílio do ambiente LabVIEW.

Para o futuro, prevemos a implantação de novos procedimentos de caracterização para dispositivos de RF e dispositivos orgânicos. Os procedimentos de automação deverão ser estendidos para medidas ópticas em dispositivos.

7 REFERÊNCIAS BIBLIOGRÁFICAS

[1] Projeto de Estruturas de um Processador Risc para Aplicação em um SoC para Controle de Irrigação - J. D. Costa, G. S. Beserra, G. M. Araújo, J. C. Marra, A. F. Rocha, J. C. Costa - Universidade de Brasília -Brasil - IBERCHIP

[2] Sintetizador de Frequências para Transceptor de RF CMOS em Sistema em *Chip* - Rafael R. P. Soares, Pablo R. O. Vogel, José C. da Costa - Universidade de Brasília. Faculdade de Tecnologia - Departamento de Engenharia Elétrica - IBERCHIP

[3] IMPLEMENTAÇÃO DE UM PROCESSADOR RISC 16-BITS CMOS NUM SISTEMA EM CHIP - Costa, Janaína Domingues - Dissertação de Mestrado - Universidade de Brasília - Faculdade de Tecnologia. Departamento de Engenharia Elétrica - Distrito Federal 2004.

[4] CARACTERIZAÇÃO E TESTE DE MÓDULOS PARA UM SISTEMA EM CHIP CMOS - Rafael R. P. Soares, Wellington A. do Amaral, Hélder H. Guimarães, José C. da Costa - Universidade de Brasília - Departamento de Engenharia Elétrica - LTSD - IBERCHIP 2005

[5] Arquiteturas de Redes Neurais Nanoeletrônicas para Processadores em Escala Giga Ou Tera - Guimarães, Janaina Gonçalves - Tese de Doutorado - Universidade de Brasília - Faculdade de Tecnologia - Departamento de Engenharia Elétrica - Distrito Federal 2005.

[6] GPIB Tutorial - http://www.hit.bme.hu/people/papay/edu/GPIB/tutor.htm

[7] LabVIEW - Getting Started with LabVIEW - July 2000 Edition
Part Number 321527D-01 - http://www.ni.com/pdf/manuals/321527d.pdf

[8] LabVIEW - User Manual - July 2000 Edition - Part Number 320999C-01
http://www.ni.com/pdf/manuals/320999c.pdf

[9] Cadence Design Systems: http://www.cadence.com/

[10] Um novo filtro com banda passante dupla utilizando ressoadores miniaturizados. - Célio Ramos Alves - Dissertação de Mestrado - Faculdade de Tecnologia - Departamento de Engenharia Elétrica - Distrito Federal 2007.

[11] Keithley Model 2400 Series SourceMeter® Quick Results Guide
Third Printing, May 2002 - Document Number: 2400S-903-01 Rev. C
http://www.keithley.com/data?asset=888

[12] Keithley Model 2400 Series SourceMeter® User's Manual
Seventh Printing, May 2002 - Document Number: 2400S-900-01 Rev. G
http://www.keithley.com/data?asset=887

[13] Measurement Computing® - http://www.measurementcomputing.com/

[14] User's Guide Agilent Technologies 8712ES and 8714ES
RF Network Analyzers - Print Date: June 2000 - Part No. 08714-90012

[15] Agilent Test & Measurement Application Note 95-1
S-Parameter Techniques
http://contact.tm.agilent.com/Agilent/tmo/an-95-1/index.html

[16] Livro Microeletrônica - Quarta Edição - Adel S. Sedra / Kenneth C. Smith - Ed. Pearson Makron Books.

[17] Eletronica 24h - http://www.eletronica24h.com.br/

[18] DETERMINAÇÃO DE PARÂMETROS CINÉTICOS E CARACTERIZAÇÃO ELÉTRICA DE FILMES DE SiO_2 PRODUZIDOS POR OXIDAÇÃO TÉRMICA PIROGÊNICA DO SI - José Camargo da Costa - Dissertação de Mestrado - Universidade Estadual de Campinas - Faculdade de Engenharia de Campinas - Departamento de Engenharia Elétrica - Laboratório de Eletrônica e Dispositivos - Dezembro 1982

[19] MOS PHYSICS AND TECHNOLOGY - Nicollian, E.H. and I.Brews, J.R. A Wiley-Interscience Publication - 1982

[20] A Simple and Unambiguous Definition of Threshold Voltage and its Implications in Deep-Submicron MOS Device Modeling - X. Zhou, K. Y. Lim, and D. Lim - IEEE TRANSACTIONS ON ELECTRON DEVICES, VOL. 46, NO. 4, APRIL 1999

[21] THE EFFECT OF CHANNEL IMPLANTS ON MOS TRANSISTOR CHARACTERIZATION - Richard V. Booth, student member, IEEE, and Thomas J. Krutsick, Student member, IEEE - IEEE TRANSACTIONS ON ELECTRON DEVICES, VOL. ED-34, NO. 12, DECEMBER 1987

[22] Nortelco - http://www.nortelco.no/default.asp?FILE=items/798/271/

[23] THE DESIGN, CHARACTERIZATION, AND MODELING OF RF LDMOSFETS ON SILICON-ON-INSULATOR MATERIAL - E. McShane and Krishna Shenai, Fellow, IEEE - IEEE TRANSACTIONS ON ELECTRON DEVICES, VOL. 49, NO. 4, APRIL 2002

[24] A DATA BASE DRIVEN AUTOMATED SYSTEM FOR MOS DEVICE CHARACTERIZATION, PARAMETER OPTIMIZATION AND MODELING - Orlin Melstrand, member, IEEE, ED'O NEILL, member IEEE, Gerald E. Sobelman, member, IEEE, and Dimitri Dokos, member, IEEE - IEEE TRANSACTIONS ON COMPUTER-AIDED DISIGN, VOL. CAD 3, NO. 1, JANUARY 1984

[25] Verification of Models for Fabrication of Arsenic Source-Drains in VLSI MOSFET's - John Albers, P. Roitman, And Charles L. Wilson, member, IEEE - Manuscript received March 3, 1983; revised June 30, 1983

[26] Keithley Model 6517A Electrometer / High Resistance Meter Getting Started Manual - Second Printing, June 2003 - Document Number: 6517A-903-01 Rev. B - http://www.keithley.com/data?asset=1621

[27] Keithley Model 6517A Electrometer User's Manual - Fourth Printing, May 2003 - Document Number: 6517A-900-01 Rev. D http://www.keithley.com/data?asset=1079

[28] AMPLIFICADOR DE POTÊNCIA COM CONTROLE DIGITAL EM TECNOLOGIA CMOS PARA APLICAÇÃO EM TELEMETRIA - Assunção, Leandro Santana- Dissertação de Mestrado - Universidade de Brasília - Faculdade de Tecnologia - Departamento de Engenharia Elétrica - Distrito Federal - 2004.

[29] A 0-10dBm, 915-927.5 MHz, 0.35μm CMOS CLASS E POWER AMPLIFIER WITH DIGITAL POWER CONTROL AND DIRECT MODULATION - Leandro Santana Assunção, José Camargo da Costa - Universidade de Brasília - Faculdade de Tecnologia - Departamento de Engenharia Elétrica - LPCI - Caixa Postal 4386 - Brasília - DF - 70919-970

[30] Programmer's Guide Agilent Technologies 8712ET/ES and 8714ET/ES RF Network Analyzers - Print Date: June 2000 - Part No. 08714-90015

[31] Observation of Negative Differential Resistance and Hysteretic Effect on Buriti Oil:Polystyrene Organic Devices - J. A. Durães and M. J. Araújo Sales - Instituto de Química, Universidade de Brasília, Caixa-Postal 4478, CEP70904-970, Brasília, DF Brasil; R. F. Souto, A. Romariz, J. C. da Costa, and A. M. Ceschina - Departamento de Engenharia Elétrica - Universidade de Brasília, Caixa-Postal 4386, CEP70919-970, Brasília, DF Brasil; S. G. C. Moreira - Departamento de Física - Universidade Federal do Pará, CEP66075-900, Belém/PA Brasil - APPLIED PHYSICS LETTERS 89, 133502 - Received 11 April 2006; accepted 1 August 2006; published online 25 September 2006.

[32] Quick Start Guide Spectrum Analyzer R&S® FSL

[33] Operating Manual Spectrum Analyzer R&S® FSL

APÊNDICES

Apêndice A - Instrumentos usados

Temos aqui um resumo das principais características dos instrumentos usados. Podem ser operados manualmente ou remotamente.

Keithley 2400 Fonte e Medidor

Pode ser usado como fonte de tensão ou de corrente e também mede resistência, corrente e tensão. Pode ser programado para executar sequências de testes. Possui memória (*buffer*) para armazenar resultados de medidas. Pode ser operado manualmente ou remotamente usando comandos SCPI através da serial RS232 ou GPIB. Ilustrado na figura A.1.

Figura A.1 - Keithley 2400

Características:

- Fornece tensão de 5 μV até 210 V e mede tensão de 1 μV a 211 V
- Fornece corrente de 50 pA até 1,05 A e mede corrente de 10 pA até 1,055 A
- Mede resistência de 100 $\mu\Omega$ (< 100 $\mu\Omega$ em ohms manual) até 211 MΩ
- Potência máxima da fonte de 22 W

Keithley 6517A Eletrômetro e medidor de alta resistência

Possui as saídas separadas na parte de trás para gerar tensão medir corrente e tensão. Ilustrado na figura A.2. Pode também medir carga, resistividade, temperatura ambiente e umidade relativa do ar. Pode ser controlado remotamente usando RS232 e GPIB. Pode ser programado para executar sequências de testes e possui memória *buffer* para armazenar resultados de medidas.

Figura A.2 - Keithley 6517A

Características:

- Mede tensão DC de 1 μV a 210 V;
- Mede corrente de 10 aA até 21 mA;
- Mede carga capacitiva de 10 fC até 2,1 μC;
- Mede resistência de 10 Ω a 210 TΩ;
- Mede Resistividade de volume e superfície;
- Mede temperatura externa de -25° C a 150° C usando o acessório termelétrico modelo 6517-TP;
- Mede umidade relativa de 0% a 100% usando o acessório 6517-RH;
- Fonte de tensão bipolar de 1 W que fornece até $\pm$ 1000 V.

Analisador de redes Agilent 9714ES

É um analisador de redes de RF fácil de usar otimizado para fornecer medidas de parâmetros de espalhamento. O instrumento contem uma fonte de RF, receptor multi-modo e uma tela de vídeo em um bloco compacto.

A fonte contém 1Hz de resolução, tempo de varredura de 40 ms e potência máxima de saída de +13dBm. Os receptores possuem processamento digital de sinais controle microprocessado para agilizar a operação. A figura A.3 mostra o instrumento.

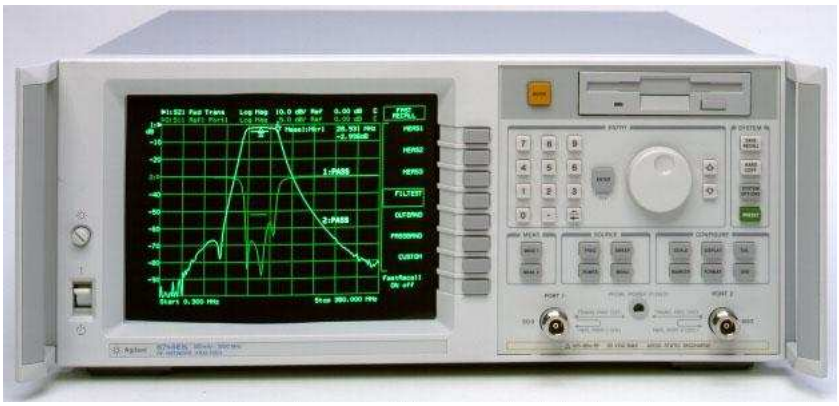


Figura A.3 - Analisador de redes Agilent 9714ES

Características:

- Detecção em faixa estreita e faixa larga;
- Escala dinâmica de 100 db;
- Varredura em tempo real (40 ms/varredura);
- Conjunto de testes de Parâmetros S integrado;
- Fonte sintetizada com 1 Hz de resolução;
- Interface LAN padrão;
- Interface GPIB padrão IEEE 488.

Gerador de Funções Agilent 33220A

Oferece 11 padrões de formas de ondas mais pulso e formas aleatórias de ondas. É o gerador mais estável em frequência e de menor distorção de sua classe. Pode ser visto na figura A.4.

Figura A.4 - Gerador de Funções Agilent 33220A

Características:

- Formas de onda senóide e quadrada até 20 MHz;
- Sinais em forma de rampla, triangular, ruído e DC;
- Geração de pulso com borda variável;
- Modulação AM, FM, PM, FSK, e PWM;
- Varredura no modo linear e lugarítimica;
- Medo gráfico para verificação visual da configuração do sinal;
- Interfaces USB, GPIB e LAN

Analisador de Espectro Rohde & Schwarz

É um analisador compacto e de baixo peso com uma ampla faixa de aplicações em desenvolvimento, manutenção e produção. Veja a figura A.5 Ele é o único instrumento nesta classe que provê gerador de rastreio até 6 GHz e uma largura de faixa de demodulação de sinais de 20 MHz.

Figura A.5 - Analisador de Espectro Rohde & Schwarz

Características:

- Faixa de Frequência de 9 kHz a 6 GHz;
- Largura de faixa de demodulação de 20 MHz;
- Incerteza total de medição <0.5 dB;
- Bateria interna;
- Medição de potência no domínio do tempo.;
- Números de pontos amostragem ajustável: máx. 32001;
- Unidades: dBm, dBμV, dBmV, dBμA, dBpW, V, W, A;
- Contador de Frequência com resolução de 1 Hz e tempo de medição de 50 ms;
- Filtros de canal, forma de estado < 2, 300 Hz a 5 MHz, opcional 100 Hz a 5 MHz, incluindo, por exemplo, 8.5 kHz, 12.5 kHz, 150 kHz, 192 kHz, 1 MHz, 1.5 MHz;
- interface LAN, VXI11 compatível para controle remoto;
- interface USB e GPIB;
- Medição de potência com sensores de potência R&S NRP.

Apêndice B - Comandos SCPI

É uma linguagem de texto que possibilita operar os instrumentos remotamente. Embora cada instrumento tenha seu conjunto próprio de comandos, eles obedecem regras de formatação padronizadas pelo IEEE. Geralmente eles começam com o sinal dois pontos. Quando existe subcomandos eles devem ficar na mesma linha, separados por dois pontos. Os valores numéricos, *booleans* ou *strings* são separados pelo espaço. Abaixo temos um exemplo.

```
:SOURce:FUNCtion CURRent
:SOURce:CURRent:MODE FIXed
:SOURce:CURRent:RANGe 1.05
:SOURce:CURRent:LEVel 2E-3
```

Maiúsculos e minúsculos

Os comandos SCPI não são caso sensível. Muitos comandos possuem uma forma abreviada. Nos manuais dos instrumentos é comum que se coloque a forma abreviada em maiúscula e o resto em minúscula. As linhas de comandos abaixo são idênticas para o instrumento. Configuram uma fonte de tensão para fornecer 20 V na saída. Na primeira linha temos os comandos como mostrado nos manuais. Na segunda linha os mesmos comandos na forma abreviada. As outras linhas de comando também são válidas.

```
:SOURce:VOLTage:LEVel 20
:SOUR:VOLT:LEV 20

:SOURCE:VOLTAGE:LEVEL 20
:source:voltage:level 20
:sour:volt:lev 20
```

O uso do ponto e vírgula

O ponto e vírgula possui duas funções. Pode ser usado para substituir a mudança de linha. Assim, uma linha pode conter vários comandos. Os comandos:

*RST
*CLS
:FUNC:IMP LSRS
:FREQ 60 kHz
:CURR 200E-6

Podem ser escritos em uma linha assim:

*RST;*CLS;:FUNC:IMP LSRS;:FREQ 60 kHz;:CURR 200E-6

Muitos comandos possuem subcomandos. O ponto e vírgula pode ser usado para anexar vários subcomandos em uma linha, evitando a repetição. Nesse caso o dois pontos são retirados. Veja o exemplo abaixo.

:SOUR:VOLT:MODE FIX
:SOUR:VOLT:RANG 210
:SOUR:VOLT:LEVEL 2

Pode ser escrito assim:

:SOUR:VOLT:MODE FIX;RANG 210;LEVEL 2

Comandos comuns

Alguns comandos são aceitos por qualquer instrumento que obedece o padrão IEEE, independente do fabricante. Esses comandos são chamados de comandos comuns e começam com asterisco. Na tabela B.1 abaixo temos alguns exemplos.

*RST	Reset - Coloca o instrumento na configuração padrão.
*CLS	Clear - Limpa os registradores de estado.
*TRG	Trigger - Dispara um evento programado.
*WAI	Wait - Espera que outros comandos sejam completados.

Tabela B.1 - Comandos comuns

Comandos que perguntam

Os comandos ou linhas de comandos que terminam com ponto de interrogação requisitam dados dos instrumentos. Após o envio deve-se fazer uma operação de leitura, que será explicada mais adiante. Os comandos de configuração possuem uma versão de pergunta, que permitem descobrir a configuração presente no instrumento. É obtida acrescentando um ponto de interrogação no lugar do valor de configuração. Se quisermos saber a tensão que a fonte de tensão do exemplo anterior está fornecendo, enviaríamos o seguinte comando:

:SOURCE:VOLTAGE:LEVEL?

Outros comandos existem apenas na forma de pergunta. Abaixo temos alguns exemplos:

*IDN?
:READ?

O primeiro faz com que o instrumento forneça seus dados de identificação como marca e modelo. Ao receber o segundo, o instrumento fornece os resultados de medidas armazenados em seu *buffer*.

Apêndice C - Funções do LabVIEW

Função *String Subset*

Extrai parte de uma palavra maior. É usada para separar os dados enviados pelos instrumentos de medidas. Na figura C.1 podemos ver suas entradas e saídas. Na entrada *string* colocamos os caracteres. Em *offset* informamos o número do caracter em que começará a seleção. Na entrada *length* informamos a largura ou quantidades de caracteres que queremos separar. A saída *substring* fornece o resultado.

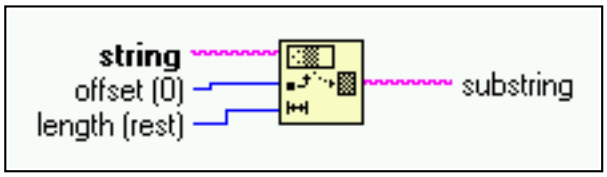


Figura C.1 - Função *String Subset*

A figura C.2 mostra um exemplo de uso dessa função. A palavra DIODO é extraída dos caracteres ZX41wwDIODOppT.

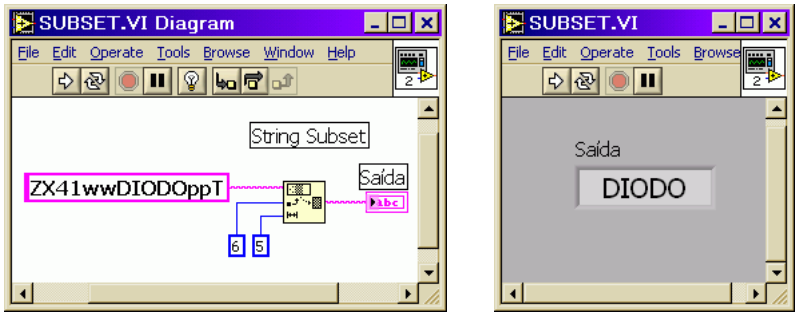


Figura C.2 - Exemplo de uso da função *Strig Subset*

Função *Search and Replace String*

Localiza e substitui palavras em um texto. A figura C.3 mostra suas entradas à esquerda e suas saídas à direita. Nas entradas podem ser ligados dados vindos de controles, de outras funções ou constantes. A entrada *offset* e a saída *offset past replacement* não são usadas nos nossos programas. Na entrada *string* insere-se o texto principal. Na entrada *search string* coloca-se a palavra que será localizada no texto principal. Na entrada *replace string* insere-se a palavra substituta. Os valores entre parênteses são os padrões (default) que são usados quando a entrada é deixada em aberto. A saída *result string* fornece o resultado.

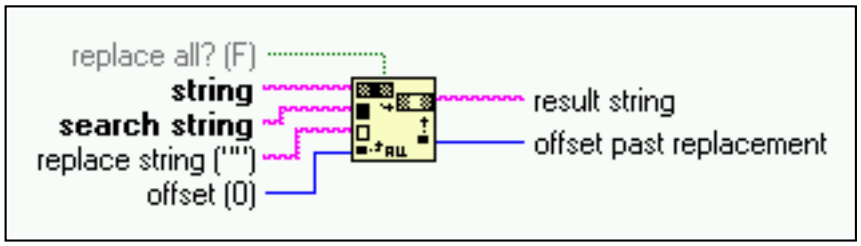


Figura C.3 - Função *Search and Replace String*

A figura C.4 mostra um pequeno programa com essa função em funcionamento. A palavra UNICAMP é substituída pela palavra UNB!. Nos nossos programas essa função será usada para alterar comandos SCPI.

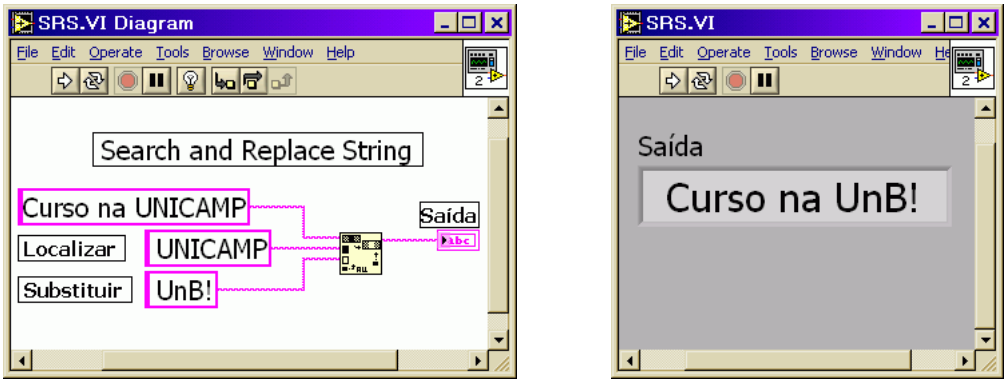
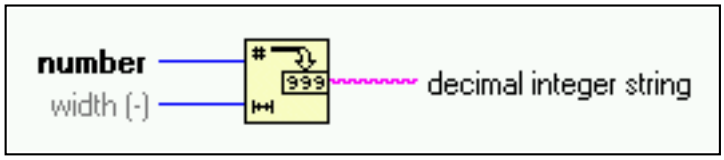


Figura C.4 - *Search and Replace String* substitui palavras

Funções de conversão de número para *string*

No labVIEW, as funções de cálculos matemáticos e os gráficos só trabalham com números. Por outro lado os comandos SCPI e os dados enviados pelos instrumentos são do tipo ASCII. Várias funções são usadas para converter de numérico para *string* e vice versa. A figura C.5 mostra duas funções que serão muito usadas nos programas. Convertem dados numéricos para *strings* inteiros e fracionários. São as funções *Number to Decimal String*, *Number to Fractional String* e *Number to Exponential String*.



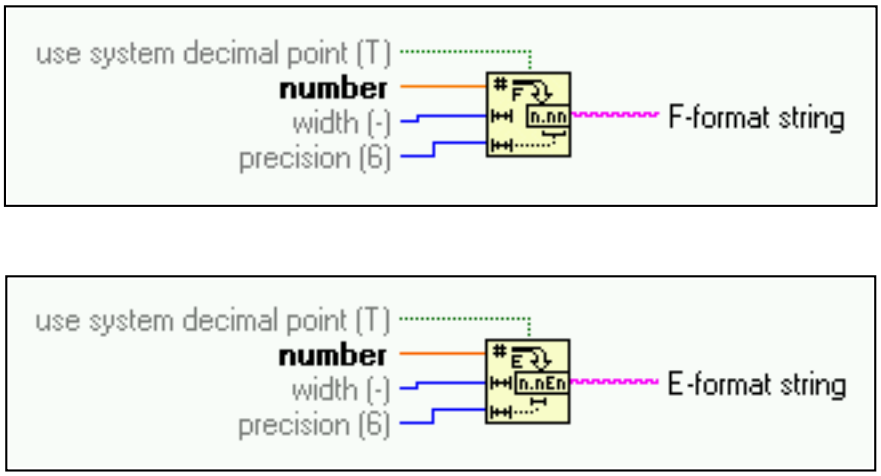


Figura C.5 - Funções de conversão de número para *string*

Função *Format Value*

Converte o número que estiver na entrada *value* em *string* de acordo com o código da entrada *format string* e anexa à palavra da entrada *string*. Poderá ser melhor entendida na descrição dos programas. Suas entradas e a saída podem ser conferida na figura C.6.

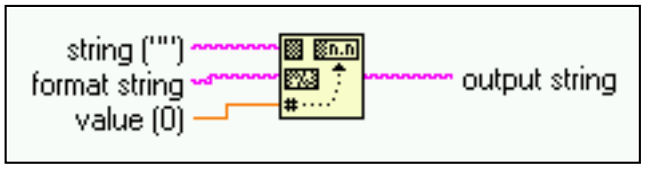


Figura C.6 - Função *Format Value*.

Conversão *String* para Número

Também é necessária a conversão de *string* para número. Uma das funções que fazem isso é a função *Frac/Exp String To Number*. Pode ser vista na figura C.7.

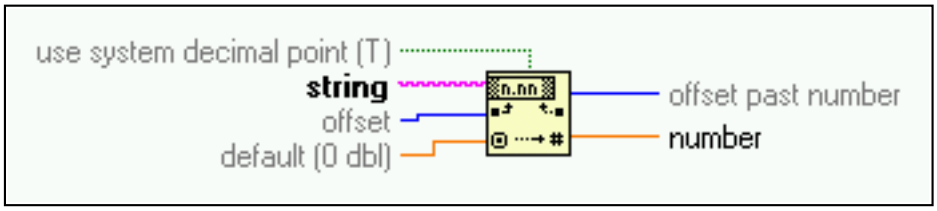


Figura C.7 - Conversão de *string* para numérico.

Funções numéricas

Trabalham com números fazendo cálculos matemáticos. São fáceis de serem compreendidas pois têm o símbolo da operação no ícone. Nas figuras C.8 e C.9 temos alguns exemplos.

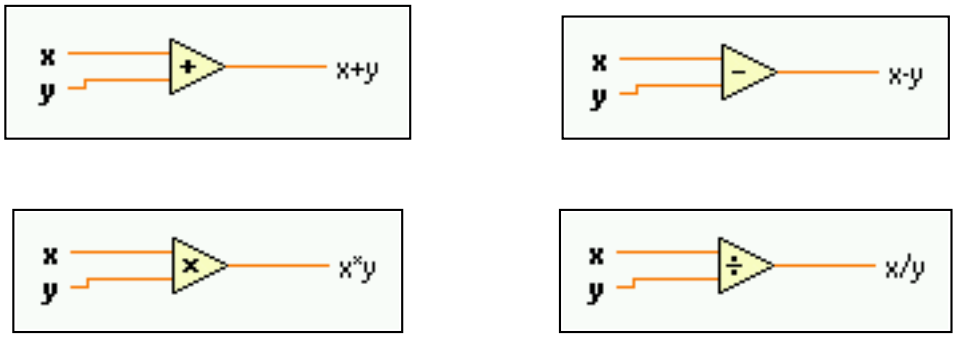


Figura C.8 - Funções numéricas.

Algumas funções possuem apenas uma entrada. A função *Sign* retorna -1 se o número for negativo, 0 se for zero e +1 se o número for positivo. Na figura C.9 abaixo temos também as funções de valor absoluto, incremento e decremento.

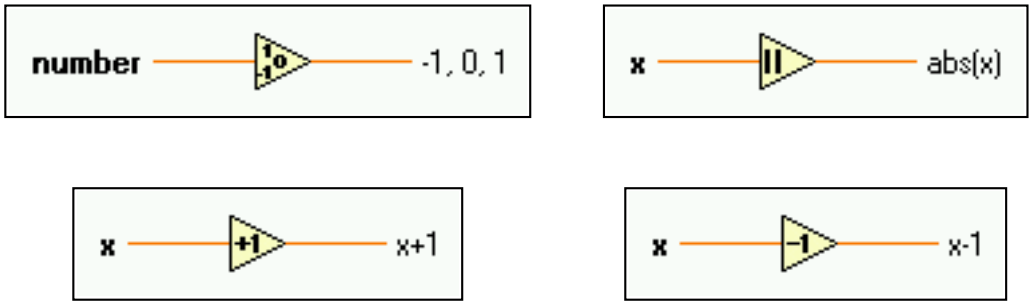


Figura C.9 - Funções numéricas.

Funções de comparação

A saída *boolean* retorna *True* se a comparação for verdadeira e *False* caso contrário. Por exemplo, a saída da função *Equal?* é *True* quando X=Y e *False* quando $X \neq Y$. Na figura C.10 temos alguns exemplos.

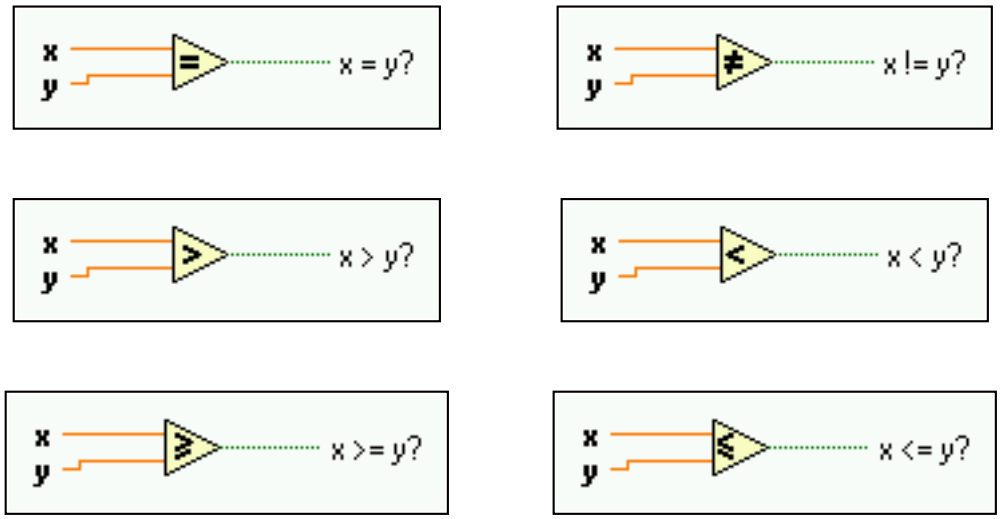


Figura C.10 - Funções de comparação.

Funções para trabalhar com dados indexados

Os dados indexados são chamados no LabVIEW de *arrays*. Podem ser construídos por funções adequadas ou gerados nas estruturas de *loops*. Equivalem às variáveis indexadas nas linguagens de programação por texto. Por exemplo A(1)=12; A(2)=33; B(0,1)=88; B(2,2)=40. São transportados por fios. Existem várias funções para se trabalhar com *arrays*. A função *Array Size* fornece a quantidade de dados de um *array*. Pode ser vista na figura C.11.

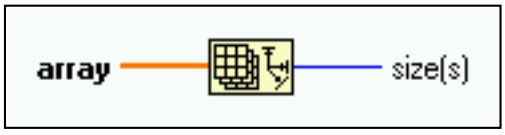


Figura C.11 - Função *Array Size*.

A função *Build Array* mostrada na figura C.12 gera um *array*. Pode ser redimensionada para formar *arrays* de “n” dimensões.

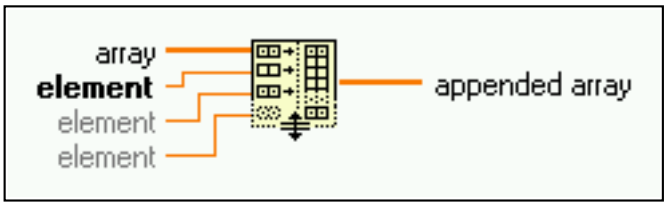


Figura C.12 - Função *Build Array*.

A função *Index Array* extrai um elemento de um *array*. Pode ser vista na figura C.13. Também pode ser redimensionada de acordo com o *array* de entrada.

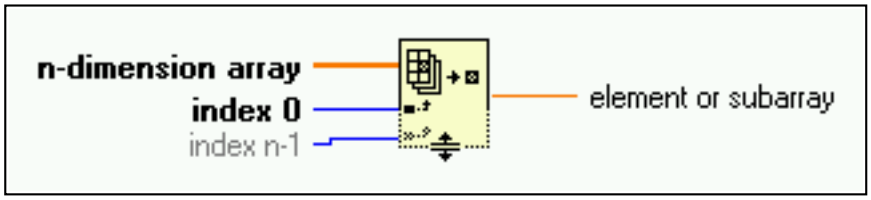


Figura C.13 - Função *Index Array*.

Funções de conversão *Array* e *Spreadsheet*

A função *Array to Spreadsheet String* converte um *array* em tabela de *string*. A função *Spreadsheet String to Array* faz o processo inverso. São usadas em nossos programas para organizar dados para serem salvos em arquivo. Veja as figuras C.14 e C.15

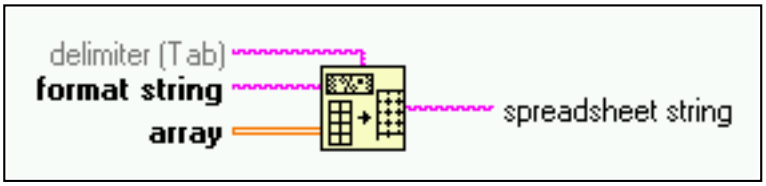


Figura C.14 - Função *Array to Spreadsheet String*.

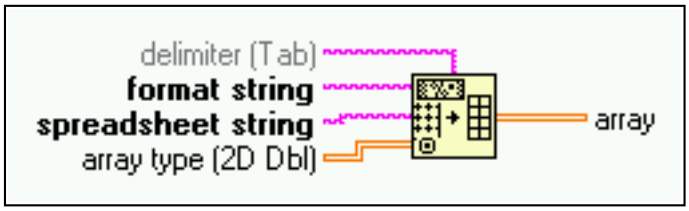


Figura C.15 - Função *Spreadsheet String to Array*.

Funções de acesso a arquivos

Trabalham com dados no disco rígido. Fazem leituras, editam e salvam dados. Também abrem caixas de diálogo do Windows, no nosso caso. O LabVIEW também possui versões para Linux. Além de *strings*, números e *booleans*, há um tipo de dado chamado *PATH*. É representado pela cor verde escuro. Não confundir do *boolean* que é verde claro e o fio é pontilhado. Transporta caminho de arquivos e diretórios do disco rígido. Na figura C.16 temos duas funções de conversão *Path To String* e *String To Path*. São usadas em conjunto com outras funções de *strings* para editar nomes de arquivos.

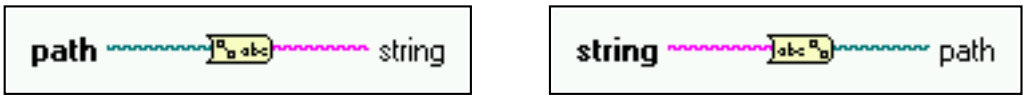


Figura C.16 - Funções de conversão *path* / *string*.

Na figura C.17 temos a função *File Dialog*, que serve para localizar ou salvar arquivos. Sua saída principal *Path* fornece o caminho do arquivo escolhido. Temos também as saídas *booleans* que indicam se o arquivo existe ou se o usuário clicou em *Cancelar*. Nas suas entradas podem ser fornecidos o prompt(texto que aparece na parte superior da janela), o caminho inicial, o modo de operação(Ex: abrir, salvar), o nome padrão do arquivo e a extenção.

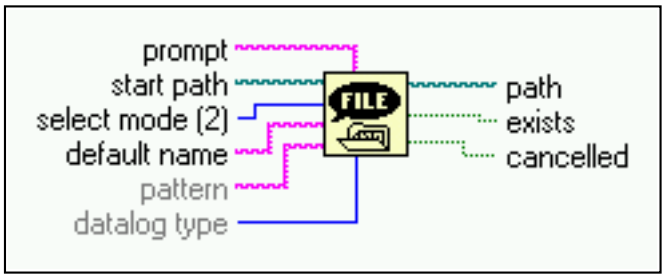


Figura C.17 - Função *File Dialog*.

Outras funções que são muito usadas são *Write Characters To File*, *Read Characters From File* e *Read Lines From File*. Podem ser vistas nas figuras C.18, C.19 e C.20. A primeira escreve os caracteres fornecidos na entrada *character strings* no arquivo fornecido na entrada *file path*. A segunda e a terceira extraem o que está escrito em um arquivo.

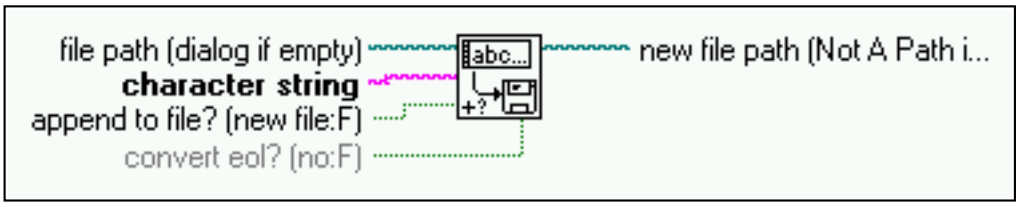


Figura C.18 - Função *Write Characters To File*.

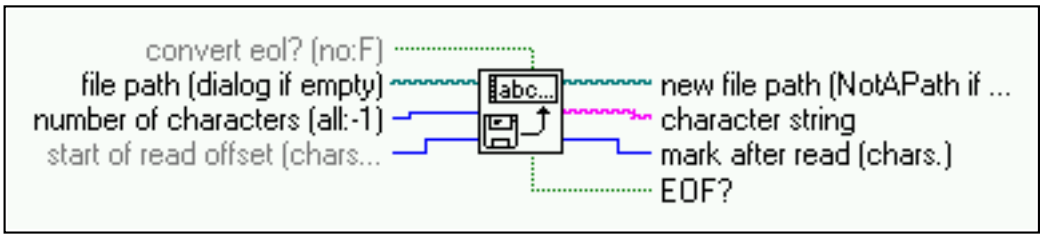


Figura C.19 - Função *Read Characters From File*.

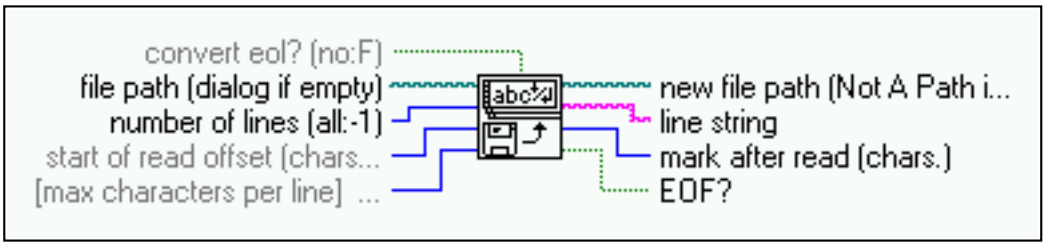


Figura C.20 - Função *Read Lines From File*.

Funções de Atraso e Tempo

São usadas para parar o programa durante um tempo predeterminado. São colocadas normalmente dentro das estruturas de *loop*. Na figura C.21 temos a função *Wait Until Next MS Multiple* e a função *Wait (ms)*.

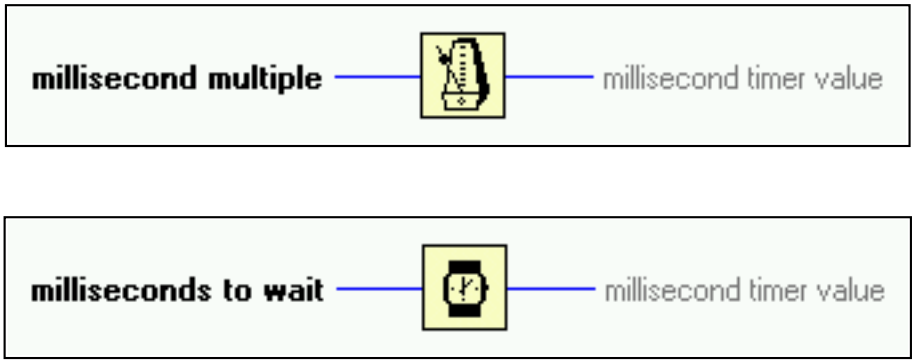


Figura C.21 - Funções de Atraso

Apêndice D - LabVIEW e GPIB

Usando GPIB e LabVIEW podemos criar procedimentos de caracterização e teste de maneira automatizada. A maior vantagem é que dois ou mais instrumentos podem trabalhar de maneira sincronizada possibilitando testes mais complexos. Podemos também salvar os resultados em arquivos para análise posterior. O LabVIEW possui várias funções para se trabalhar com GPIB. Além de trocar informações com os instrumentos é possível monitorar o barramento e obter informações sobre seu estado. A seguir serão explicadas as funções que serão usadas nos programas.

Funções GPIB Send e SendList

Nas figuras D.1 e D.2 temos as entradas e saídas das duas funções. A função *Send* envia os comandos SCPI presentes na sua entrada *data string* ao instrumento cujo endereço é inserido na sua entrada address. A entrada *bus* é usada quando temos mais de uma interface GPIB no computador. Seleciona qual interface será usada. Não é usada nos nosso programas. A saída *byte cont* fornece o número de caracteres enviados. A entrada *error in* e a saída *error out* presentes na maioria das funções GPIB permitem ligar várias funções GPIB em série determinando a ordem de execução. Servem também para depuração. Se colocarmos um indicador na saída *error out* da última função de uma sequência, caso ocorra algum erro, teremos informações sobre o seu tipo e o local de ocorrência. A função *Send List* funciona de forma semelhante porém envia os mesmos comandos para vários instrumentos simultaneamente. Se diferencia da função *Send* pela entrada *address list* que recebe um *array* com os endereços de vários instrumentos.

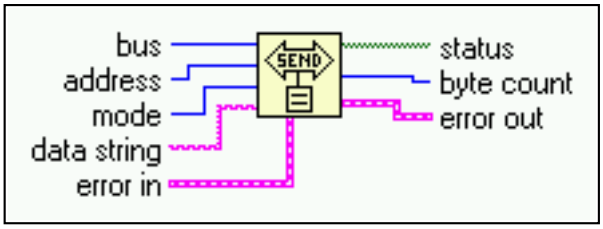


Figura D.1 - Função *Send*

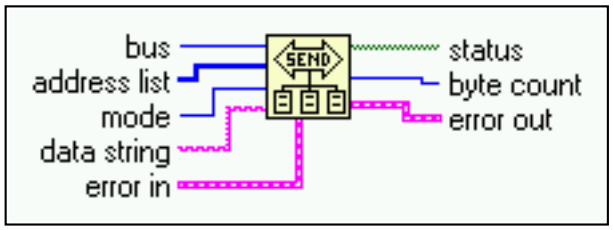


Figura D.2 - Função *SendList*

Função GPIB *Receive*

É mostrada na figura D.3. Recebe dados do instrumento cujo endereço é inserido na entrada *address*. Os dados podem ser resultados de medidas ou informações sobre o instrumento, dependendo dos comandos previamente enviados pelas funções *Send* e *SendList*. São disponibilizados na saída *data string*. A saída *byte count* fornece o número de caracteres recebidos. Na entrada *count* fornecemos o número máximo de caracteres que a função deverá receber.

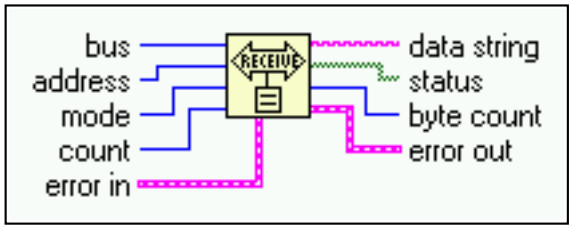


Figura D.3 - Função *Receive*.

Função GPIB *SetTimeOut*

Determina o tempo disponível para que as outras funções GPIB presentes possam transferir dados. É usada quando trabalhamos com grande quantidade de dados. Veja a figura D.4

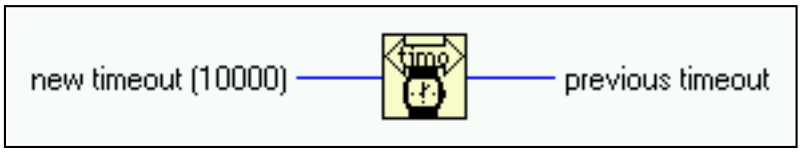


Figura D.4 - Função *SetTimeOut*.

Função *FindLstn*

Mostra quantos instrumentos estão ligados e seus endereços. Veista na figura D.5. Na sua entrada *address list* inserimos os endereços que queremos checar. Repare que é um *array*. Na entrada *limit* colocamos o máximo de endereços que podem ser mostrados. Na saída *listener address list* temos um *array* com os endereços dos instrumentos presentes no barramento. A saída *number of listeners* fornece a quantidade de instrumentos encontrados.

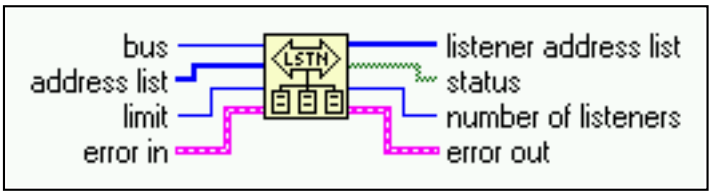


Figura D.5 - Função *FindLstn*.

Apêndice E - Caracterização de Transistores

Usa-se dois programas, um que traça a curva $I_{DS}xV_{DS}$ e outro que traça a curva $I_{DS}xV_{GS}$. Ambos usam os instrumentos Fonte-Medidor Keithley 2400 e o Eletrômetro Keithley 6517A.

Os programas possuem duas possibilidades de funcionamento. O modo Medir e o modo Carregar Arquivo. No modo de medida, os programas controlam os instrumentos, colhem os resultados das medidas, gravam em arquivo e traçam os gráficos. No modo Carregar Arquivo os programas carregam arquivos previamente gravados e traçam o gráfico para análise.

Os códigos dos dois programas feitos em LabVIEW compartilham 6 SubVI's. O SubVI **INICIAR** configura os dois instrumentos para gerar tensão e medir corrente. O SubVI **ESCALA** calcula o número de execuções "N" para estruturas *For Loop*. Também indica se a tensão é crescente ou decrescente determinando o sinal do passo. O SubVI **VOLT** calcula a tensão a ser gerada. Em seguida envia comandos SCPI aos instrumentos para gerar tensão. O SubVI **LER** recebe dos instrumentos os valores das correntes I_{DS} e I_{GS} e organiza para exibição no gráfico e salvar em arquivo. O SubVI **SALVAR** cria um arquivo de texto e salva os resultados. Também desliga as fontes de tensão dos instrumentos. O SubVI **GRÁFICO** é usado apenas no modo leitura. Ele captura dados nos arquivos salvos e exibe no gráfico. A figura E.1 mostra os SubVI's. Elas são identificadas por letras em vermelho nos ícones.

Os programas seguem o mesmo modelo padrão, porém controlam dois instrumentos simultaneamente. O SubVI **VOLT** é usado duas vezes em cada programa, ora endereça ao Keithley 2400, ora ao Eletrômetro Keithley 6517A.

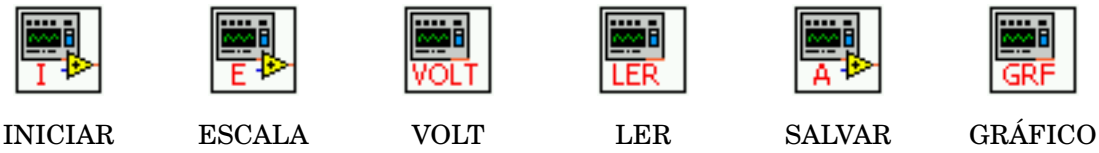


Figura E.1 - SubVI's dos programas de caracterização.

O programa IDSxVDS usa duas estruturas *For Loop* para controlar os instrumentos em sequência.

Programa IDSxVDS.

Na figura E.2 podemos ver o programa em funcionamento. Existem dois campos chamados *Endereço*. O primeiro se refere ao endereço GPIB do fonte e medidor Keithley 2400 com o número 24. O segundo é do Eletrômetro Keithley 6517A que está no endereço 27. A chave *boolean Medir-Arquivo* determina o modo de funcionamento Medir ou Carregar Arquivo controlando a estrutura *Case* principal.

Quando o programa é executado no modo medir, o teste é feito da seguinte forma. Com o Eletrômetro aplicando a tensão $V_{GS\ 1}$ na porta, o Keithley 2400 varia a tensão de 0 até o valor de $V_{DS\ MAX}$ determinado pelo usuário. O incremento de tensão é definido no campo *Passo*. Para cada valor de V_{DS}, os valores de corrente de dreno e porta do transistor são lidos e armazenados na memória. Então aplica-se $V_{GS\ 2}$. O processo é repetido até chegar em $V_{GS\ 6}$. No fim do processo, o resultado das medidas é exibido no gráfico e gravado em arquivo.

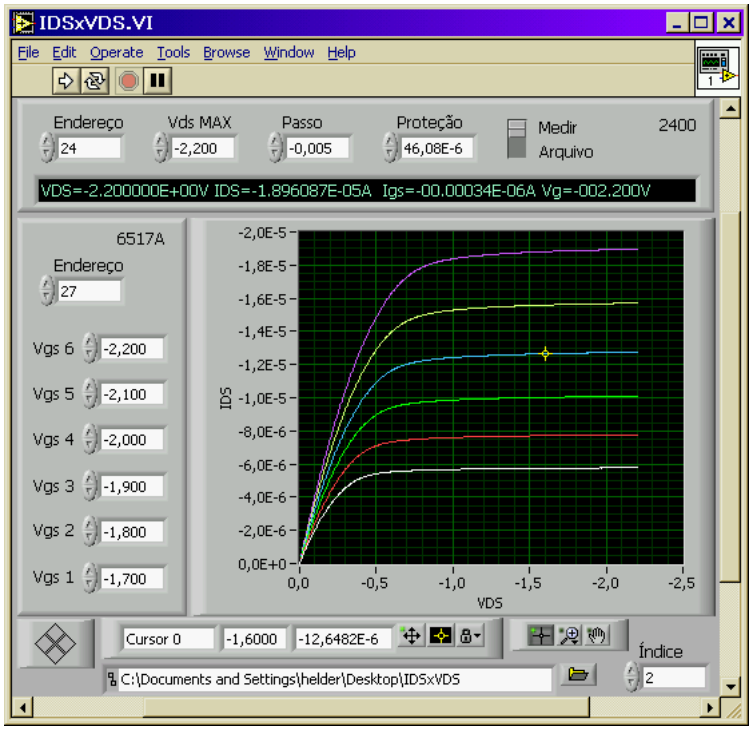


Figura E.2 - Programa IDSxVGS.

O procedimento para traçar as curvas é conseguido com duas estruturas de *loop*. O *Loop* VD que controla V_{DS} e o *Loop* VG que controla V_{GS}. O *Loop* VD fica dentro do *Loop* VG. Inicialmente, o SubVI **INICIAR** configura os dois instrumentos. O SubVI **ESCALA**, com os valores de $V_{DS}(Max)$ e do $Passo$ fornecidos pelo usuário, calcula o número de vezes que o *Loop* VD será executado. Em seguida *Loop* VG seleciona os valor de $V_{GS}01$ através da estrutura *Case* e envia ao SubVI **VOLT**. Observe o índice fixado em 1. O SubVI **VOLT** é endereçada ao Eletrômetro Keithley 6517A para polarizar a porta do transistor. Dentro do *Loop* VD outra instância do SubVI **VOLT** endereçada ao Fonte-Medidor Keithley 2400 vai incrementando o valor da tensão de dreno começando de zero até atingir $V_{DS}(Max)$. Isso é conseguido multiplicando o valor do passo pelo índice da estrutura. Para cada valor de tensão fornecido o SubVI **LER** mede e armazena os valores das correntes na memória. O processo é repetido para $V_{GS}02$, $V_{GS}03$, $V_{GS}04$, $V_{GS}05$ e $V_{GS}06$. Os resultados são então exibidos no gráfico e salvos num arquivo. Na figura E.3 temos o código do programa no modo Medir e na figura E.4 o seu fluxograma.

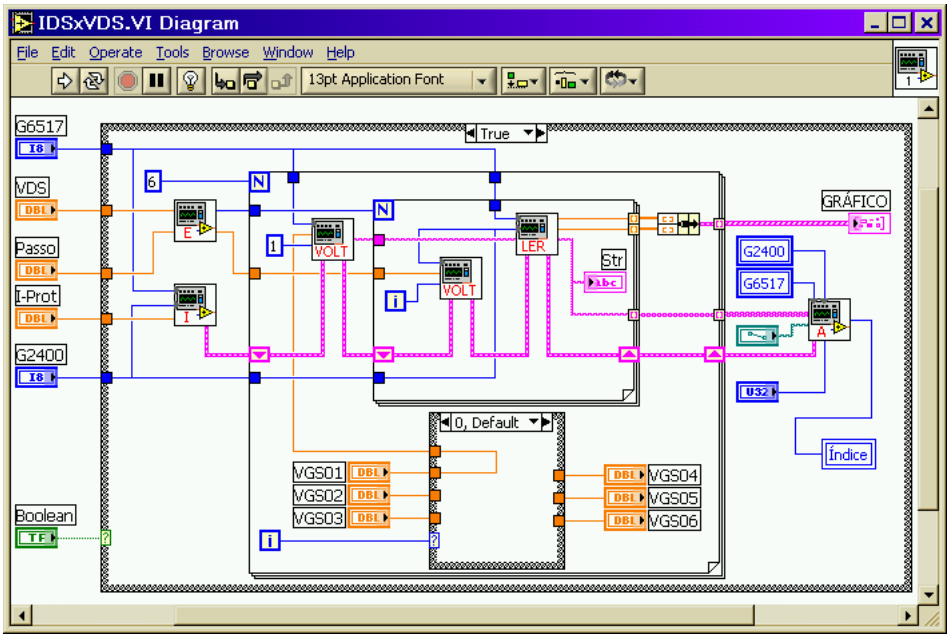


Figura E.3 - Código do Programa IDSxVDS.

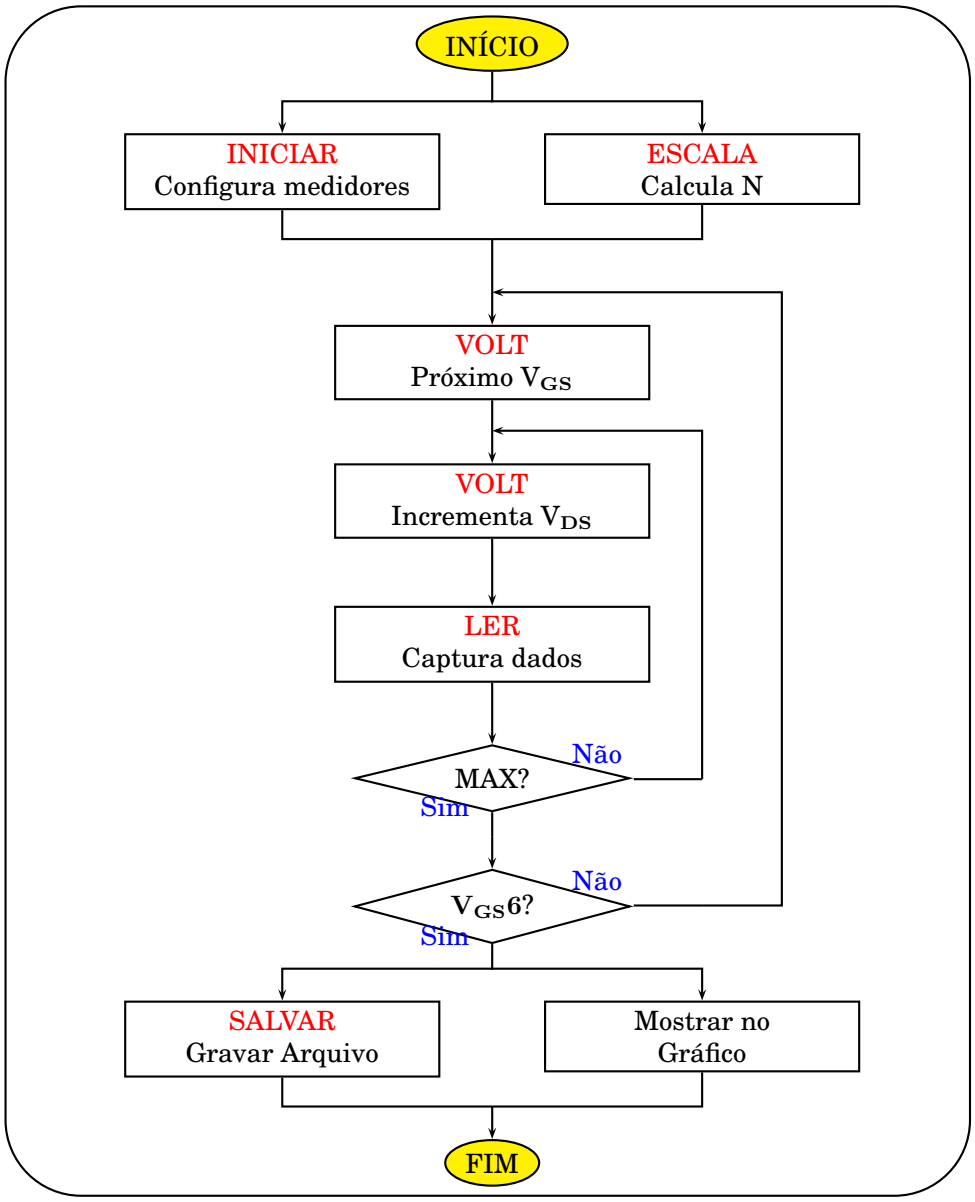


Figura E.4 - Fluxograma do Programa IDSxVDS

Quando executado no modo Carregar Arquivo, a função *File Dialog* abre a caixa de diálogo do Windows vista na figura E.5. O usuário pode escolher um arquivo previamente gravado para ser exibido no gráfico. Pode também clicar em *Cancelar*. Quando um arquivo é escolhido o caminho é levado ao SubVI **GRÁFICO**, como mostra a figura E.7. Ele fornece o valor da corrente de proteção I-Prot, um *cluster* para o gráfico e mais dois *arrays* com os valores de V_{DS} e V_{GS}. Os seis valores de V_{GS} são recuperados através de uma estrutura *Case* dentro de uma estrutura *For Loop*. As seis variáveis locais são postas uma em cada área da estrutura case. Esta é controlada pelo índice da estrutura *For Loop*. Cada variável local recebe o valor correspondente do *array*, de *VGS01* a *VGS06*.

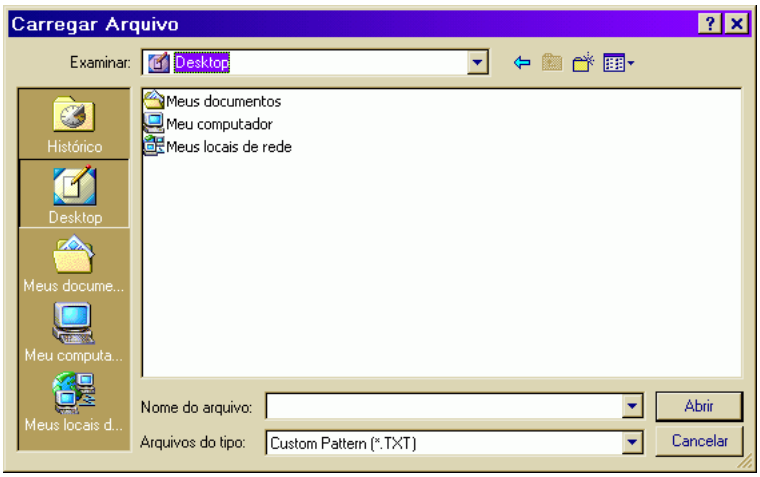


Figura E.5 - Escolher um arquivo

O valor de $V_{DS}(Max)$ está na última posição do *array*. O tamanho do *array* "N" é obtido pela função *Array Size* e decrementado pois o índice varia de "0" a "N-1". Com a função *Index Array* recupera-se V_{DS} máximo. Como o V_{DS} começa sempre do zero, na segunda posição do *array* temos sempre o valor do passo.

Quando o usuário clica em *Cancelar* a saída *Cancelled* coloca a estrutura em *True*. O gráfico é zerado, como mostra a figura E.6. Na figura E.8 temos o fluxograma para o modo Carregar Arquivos.

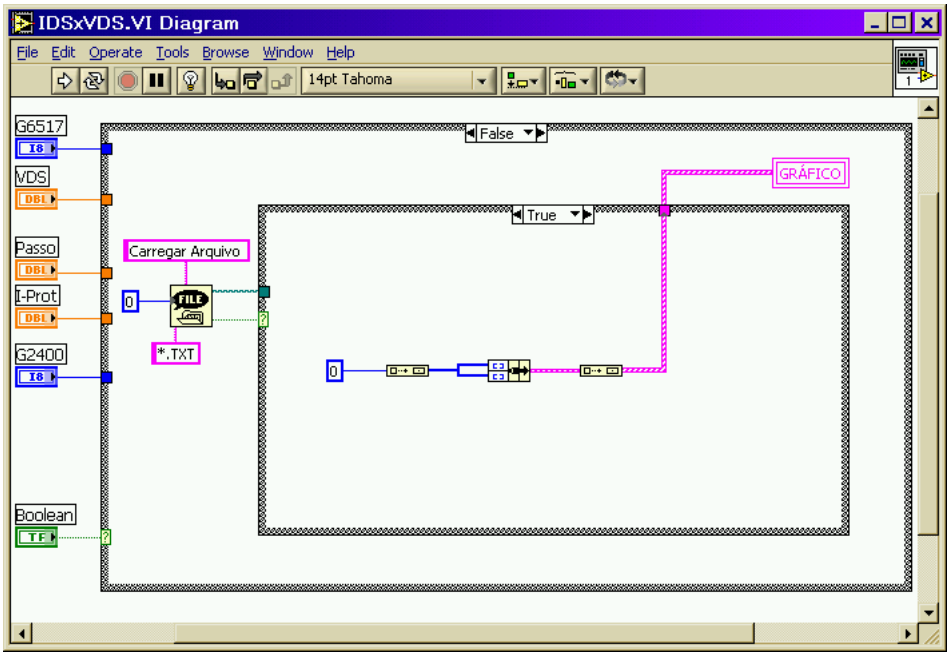


Figura E.6 - Programa IDSxVDS zera o gráfico.

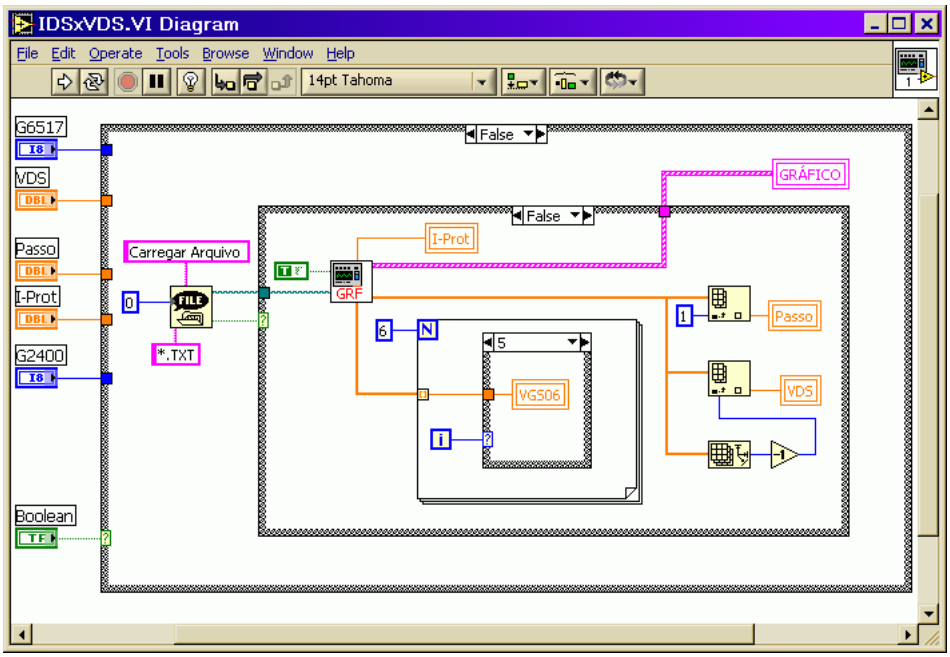


Figura E.7 - Programa IDSxVDS no Carregar Arquivo.

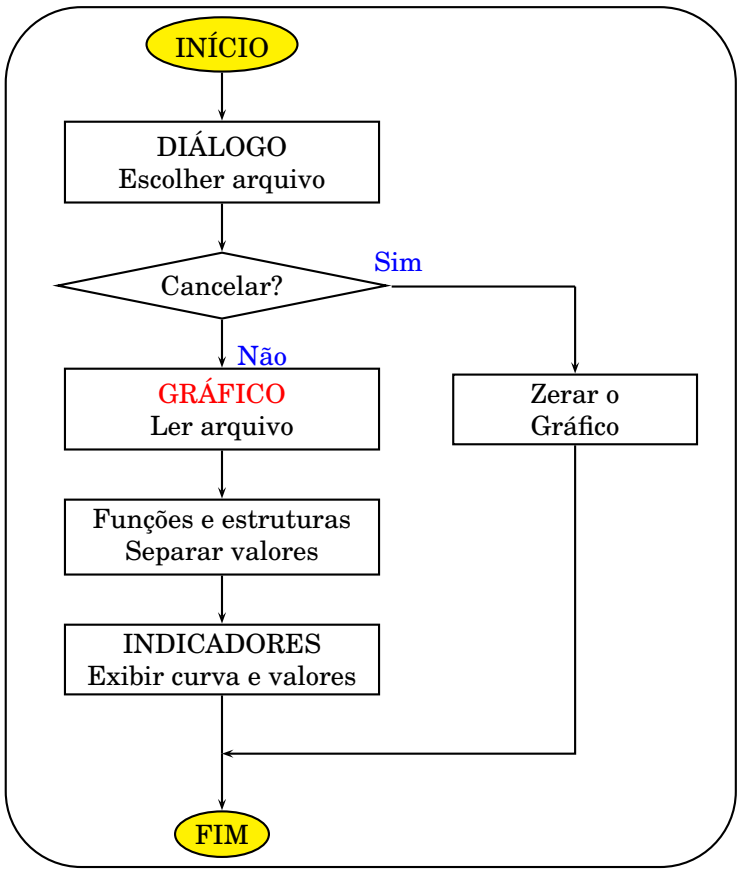


Figura E.8 - Fluxograma do modo Carregar Arquivo

Programa IDSxVGS

Nesse caso, a tensão V_{DS} é fixa. Varia-se a tensão de porta e mede a corrente IDS para cada valor de VGS. Na figura E.9 podemos ver a tela do programa em funcionamento.

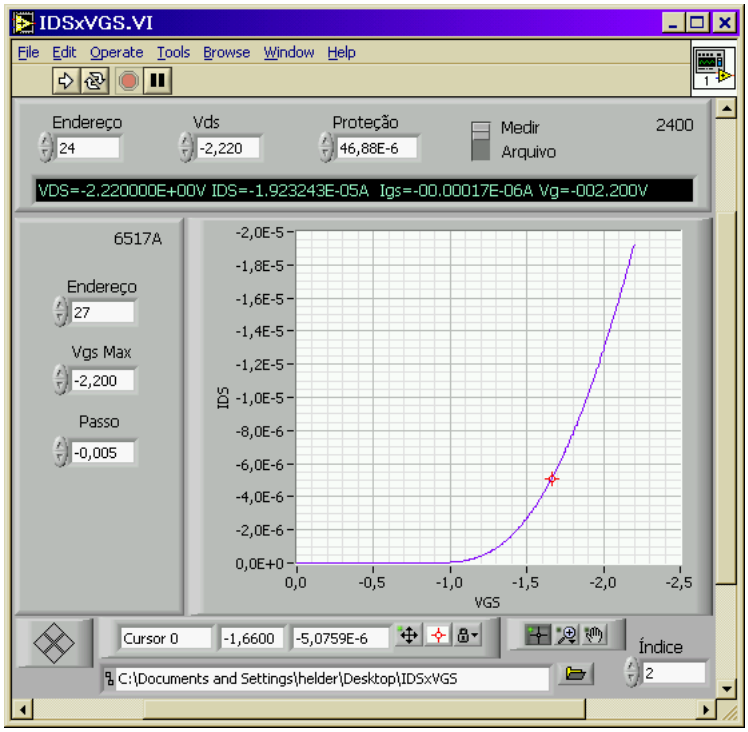


Figura E.9 - Programa IDSxVGS.

Na figura E.10 temos o código fonte no modo Medir e na figura E.11 o fluxograma. Primeiramente o SubVI **INICIAR** configura os dois instrumentos. O SubVI **ESCALA** calcula o número de execuções "N" para a estrutura de *loop*. Uma instância do SubVI **VOLT** comanda o Fonte-Medidor Keithley 2400 para gerar V_{DS} fixo. Observe a entrada índice do SubVI fixada em 1. Dentro do *loop*, outra instância do SubVI **VOLT** varia a tensão na porta do transistor de 0 a V_{GS} máximo, controlando o Eletrômetro Keithley 6517. Os valores são obtidos multiplicando o valor do passo pelo índice da estrutura. Para cada valor de V_{GS} o SubVI **LER** captura os resultados medidos. Finalmente, o resultado é mostrado no gráfico e gravado em arquivo.

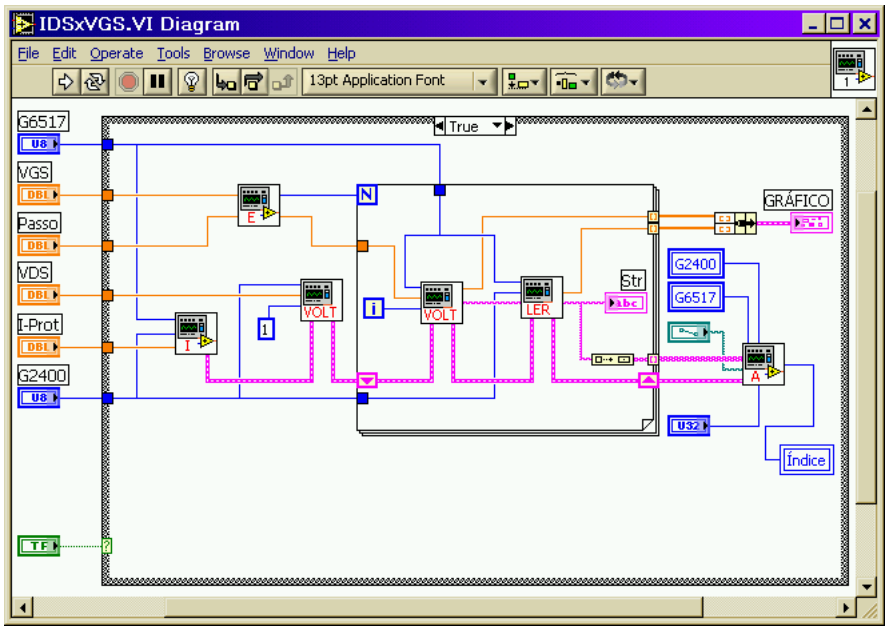


Figura E.10 - Programa IDSxVGS no modo Medir.

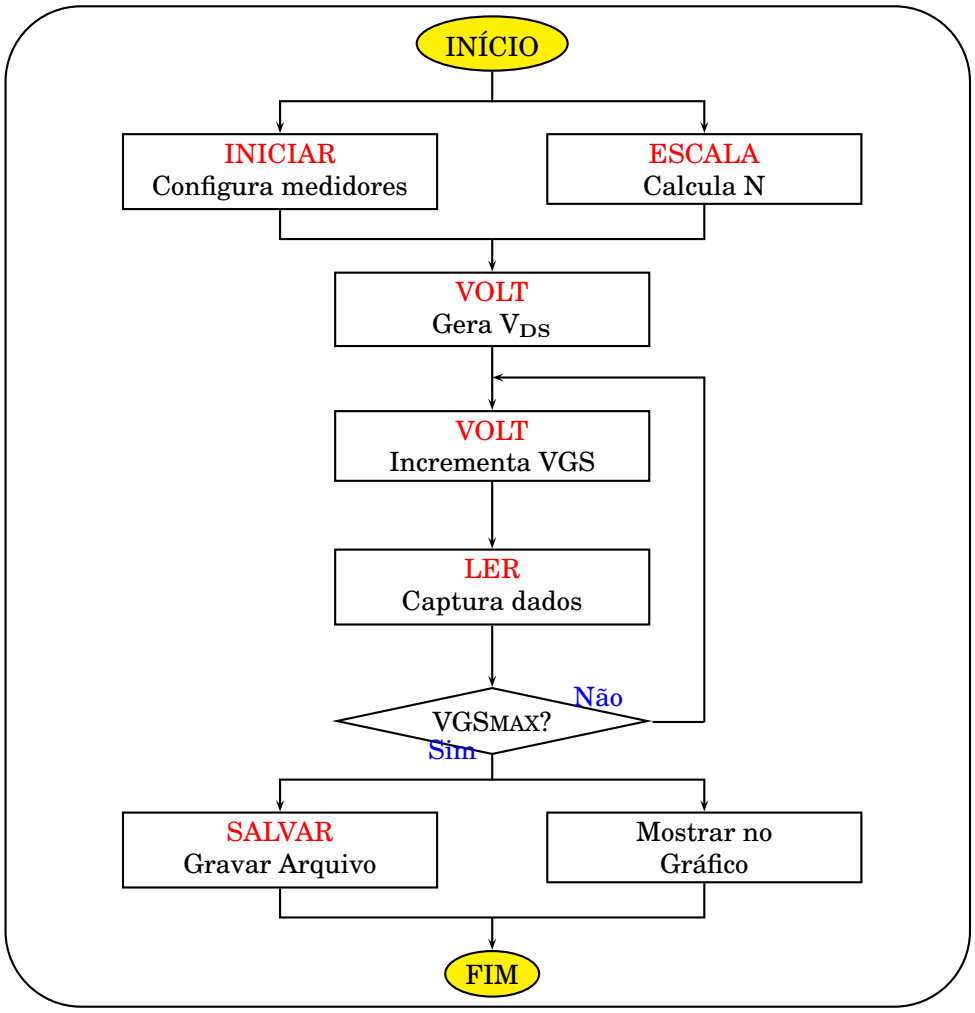


Figura E.11 - Fluxograma do Programa IDSxVGS

Quando executado no modo Carregar Arquivo, a função *File Dialog* abre a caixa de diálogo do Windows igual a do programa interior, figura E.5. O usuário pode escolher um arquivo previamente gravado para ser exibido no gráfico. Pode também clicar em *Cancelar*. Quando um arquivo é escolhido o caminho é levado ao SubVI **GRÁFICO**, como mostra a figura E.13. Ela fornece o valor da corrente de proteção I-Prot, *arrays* com IDS e V_{GS} que vão para o gráfico e mais um *array* com os valores de V_{DS}.

O valor de V_{GS} Max está na última posição do *array*. O tamanho do *array* "N" é obtido pela função *Array Size* e decrementado pois o índice varia de "0" a "N-1". Com a função *Index Array* recupera-se V_{GS} Max.

Subtraindo de V_{GS} Max o valor da penúltima posição, consegue-se o valor do passo. V_{DS}, fixo é igual em todas as posições. É obtido na posição zero. Os valores são escritos através de variáveis locais.

Quando o usuário clica em *Cancelar* a saída *Cancelled* coloca a estrutura em *True*. O gráfico é zerado, como mostra a figura E.12. Na figura E.14 temos o fluxograma para o modo Carregar Arquivos.

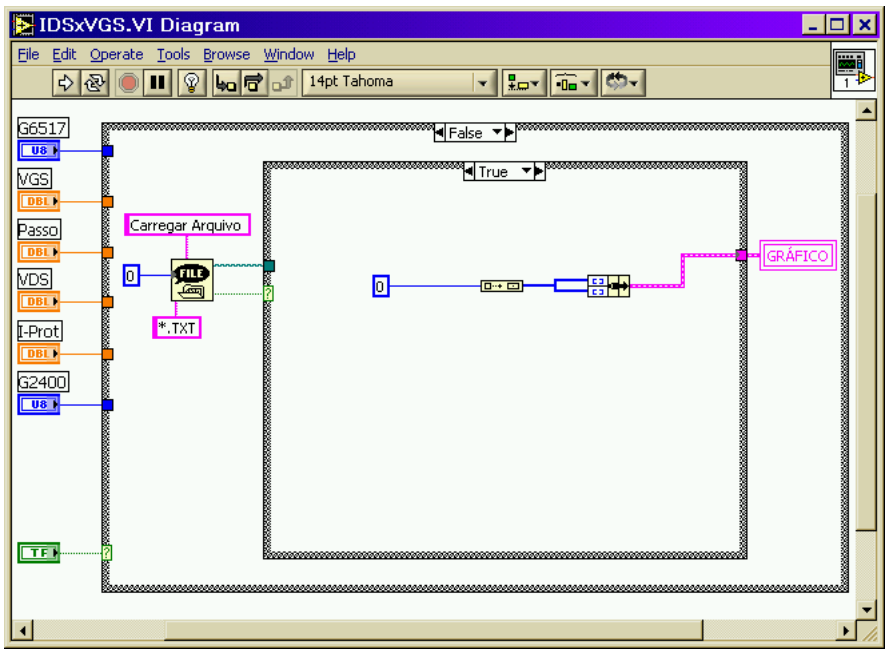


Figura E.12 - Programa IDSxVGS zerar o gráfico.

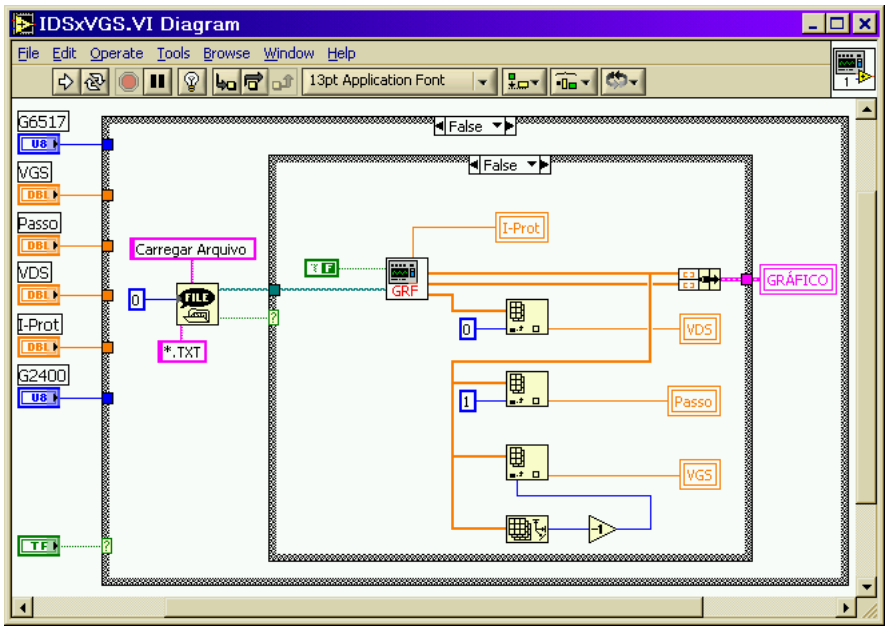


Figura E.13 - Programa IDSxVGS no modo Carregar Arquivo.

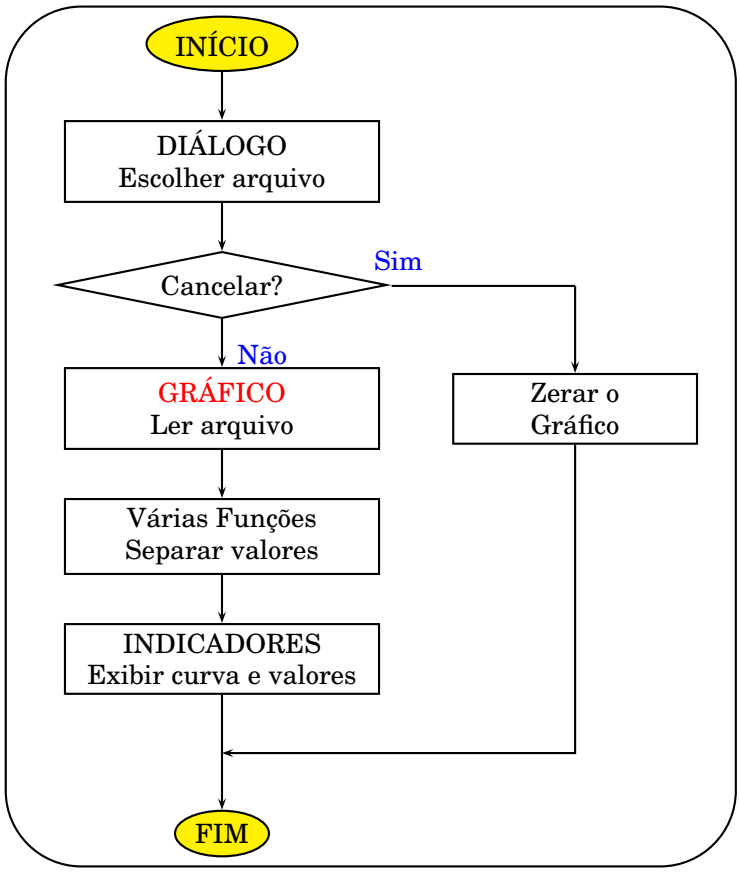


Figura E.14 - Fluxograma do modo Carregar Arquivo

A seguir os códigos dos SubVI's serão estudados separadamente.

SubVI INICIAR

Tem a função de resetar e configurar os equipamentos Keithley 2400 e 6517A para gerar tensão e medir corrente. A figura E.15 mostra o código fonte e a figura E.16 o fluxograma. O primeiro conjunto de comandos SCPI é destinado ao Fonte-Medidor Keithley 2400. É explicado na tabela E.1. Configura para gerar tensão e medir a corrente consumida pelo DUT que é no nosso caso entre dreno e fonte do transistor. A entrada I-Prot recebe o valor da corrente de proteção definido pelo usuário. É a corrente máxima que a fonte irá fornecer para não danificar o DUT. Esse valor é transformado em *string* pela função *Number to Exponential String* e inserido nos comandos pela função *Search and Replace String*, substituindo a palavra [II]. A função GPIB *Send* envia os comandos ao instrumento.

*RST	Reset - Coloca na configuração padrão.
:SOUR:FUNC VOLT	Configura para fornecer tensão.
:SOUR:VOLT:MODE FIX	Fonte de tensão no modo fixo.
:SOUR:VOLT:RANG 21	Escala de 21 V
:SOUR:VOLT 0	Zerar - Fonte inicia em 0 V
:SENS:FUNC "CURR"	Configura para medir corrente.
:SENS:CURR:PROT [II]	Define a corrente de proteção.
:SENS:CURR:RANG:AUTO ON	Escala de corrente no modo automático.
:OUTPUT ON	Ativar a saída

Tabela E.1 - Configuração do Fonte-Medidor Keithley 2400

O segundo bloco de comandos configura o Keithley 6517A, que está ligado entre porta e fonte do transistor para as mesmas funções. O comando :CONF:CURR precisa de uma operação de leitura. O *loop* com o temporizador gera um tempo para que os instrumentos se configurem. Em seguida é feita uma leitura.

:SYSTEM:PRESET	Coloca na configuração padrão.
:SOUR:VOLT:RANG 100	Escala da fonte de tensão em 100 V.
:SOUR:VOLT:LIM:AMPL 6	Tensão gerada limitada em 6 V.
:SOUR:VOLT:LIM:STAT ON	Ativa o limite de tensão.
:SOUR:VOLT 0	Zerar - Fonte inicia em 0 V.
:OUTP ON	Ativa a fonte de tensão.
:SENS:FUNC "CURR"	Configura para medir corrente.
:SENS:CURR:AVER:TCON MOV	Configura o filtro de ruído.
:SENS:CURR:AVER:STAT ON	Habilita o filtro de ruído.
:CONF:CURR	Configura para medir corrente.
:SENS:CURR:RANG:AUTO OFF	Desabilita a escala automática.
:SENS:CURR:RANG 20E-6	Escala do medidor em 20μA.
:SYSTEM:ZCH OFF	Desabilita a checagem de zero.

Tabela E.2 - Configuração do Eletrômetro Keithley 6517A

Os comandos da tabela E.3 são enviados pela função GPIB *Send* e o resultado recebido pela função GPIB *Receive*. Esse resultado não é usado pelo programa.

*WAI	Aguardar o fim da execução dos outros comandos.
:READ?	Realizar uma medida e enviar o resultado.

Tabela E.3 - Comandos SCPI para realizar uma medida.

Os últimos comandos fazem o ajuste de zero no amperímetro do Eletrômetro. São descritos na tabela E.4.

:SENS:CURRent:REFerence:STAT Off	Desabilita a correção.
:SENS:CURRent:REFerence:ACQuire	Memoriza a corrente de fuga.
:SENS:CURRent:REFerence:STAT ON	Habilita a correção.
:INIT:CONT ON;	Instrumento mede continuamente.

Tabela E.4 - Comandos SCPI para calibrar o eletrômetro.

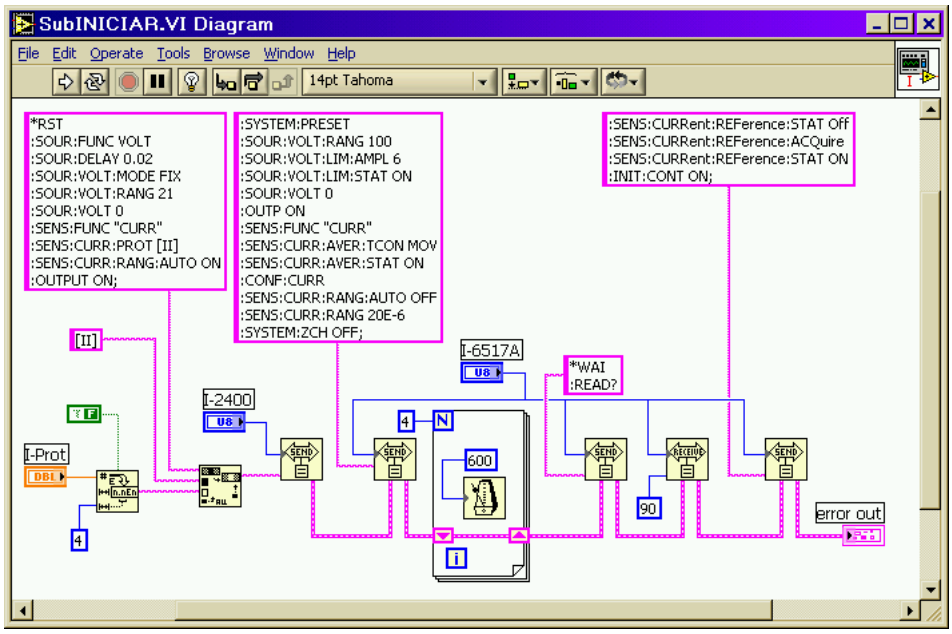


Figura E.15 - SubVI **INICIAR**.

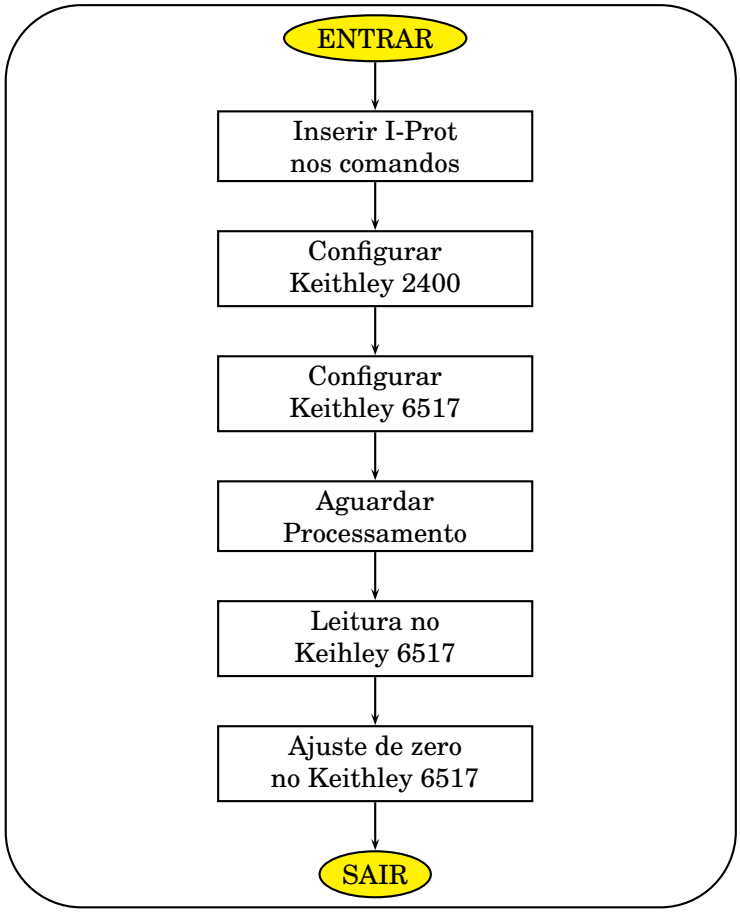


Figura E.16 - Fluxograma do SubVI **INICIAR**

SubVI ESCALA.

Captura o valor de VMAX e o passo de tensão fornecido pelo usuário e calcula o número de interações "N" para a estrutura *For Loop*. Basicamente o número pode ser calculado simplesmente dividindo VMAX pelo passo. Entretanto é necessário adicionar alguns cálculos para resolver problemas de arredondamento e conversão. A estrutura trabalha apenas com números inteiros e positivos. O código pode ser conferido na figura E.17.

Se o usuário fornecer os valores: VMAX = 3,0 V; e Passo = 0,6 V deve ser geradas as seguintes tensões:

0,0 V : 0,6 V : 1,2 V : 1,8 V : 2,4 V : 3,0 V

São gerados 6 valores de tensões. Logo, o SubVI deve calcular "N=6" . Trabalha-se com o valor absoluto dos dados de entrada. A multiplicação por 1000 permite valores de entrada fracionários com até 3 casas decimais. Soma-se 0,6 antes de converter para inteiro garantindo o arredondamento sempre para cima.O cálculo fica assim:

|3,0| x 1000 = 3000 |0,6| x 1000 = 600 3000 / 600 + 0,6 = 5,6

Arredondando para inteiro: **N = 6**

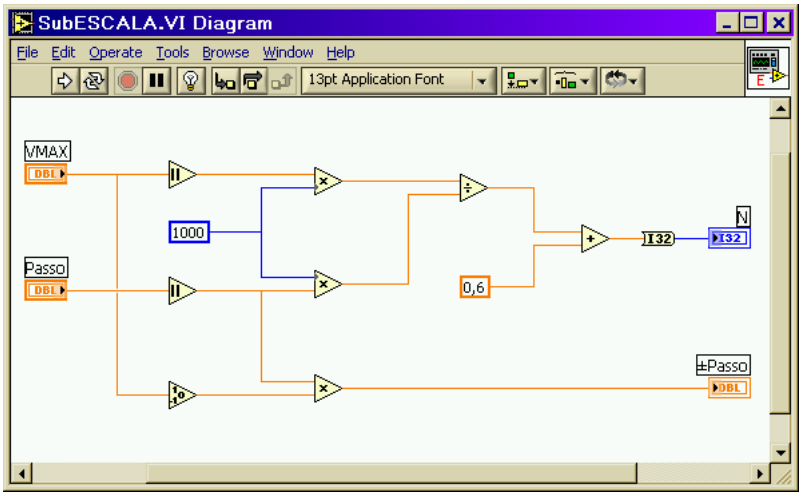


Figura E.17 - SubVI **ESCALA**.

SubVI VOLT

Fornece os comandos SCPI para o instrumento especificado no endereço GPIB gerar as tensões. O instrumento endereçado pode ser o Keithley 2400 ou o Keithley 6517A dependendo do código principal. As saídas *VNUM* e *VSTR* fornecem ao código principal o valor da tensão gerada em *string* e número. O funcionamento pode ser verificado no código mostrado na figura E.18 no fluxograma da figura E.19. Os comandos SCPI enviados estão descritos na tabela E.5.

O valor numérico do passo de tensão fornecido pelo usuário é multiplicado pelo índice da estrutura *For Loop*. O resultado é o valor de tensão que o instrumento deverá gerar. Esse valor é transformado em *string* pela função *Format Value*. O código de formatação "%.;%+08.3" gera números de 8 dígitos contando o sinal e o ponto que separa as casas decimais. A função *Search and Replace String* substitui nos comandos SCPI os caracteres "[VV]" pela tensão calculada. Os comandos são enviados ao instrumento pela função GPIB *Send*.

:SOUR:VOLT [VV]	Determina o valor da tensão fornecida.
:SOUR:VOLT?	Pergunta qual a tensão fornecida.

Tabela E.5 - Comandos do SubVI VOLT

Para uma tensão de 2,4 V a primeira linha de comando será:

:SOUR:VOLT +002.400

Em seguida pergunta-se ao instrumento a tensão que ele está fornecendo. Tal procedimento torna o programa mais confiável pois um eventual problema de comunicação será detectado imediatamente. A função GPIB *Receive* envia a resposta do instrumento para a função *Frac/Exp String To Number* que transforma o valor da tensão em numérico e fornece à saída *VNUM*. O valor é também enviado à outra função *Format Value* que coloca na saída *VSTR* o valor da tensão em *String* no formato desejado.

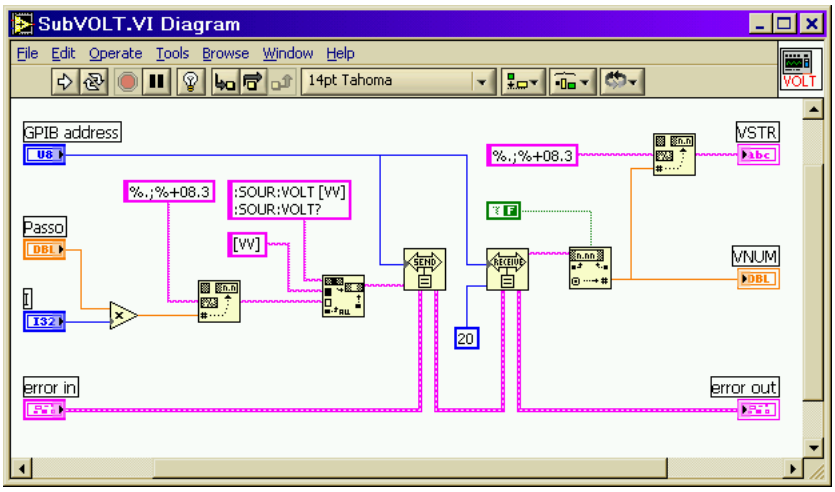


Figura E.18 - SubVI **VOLT**.

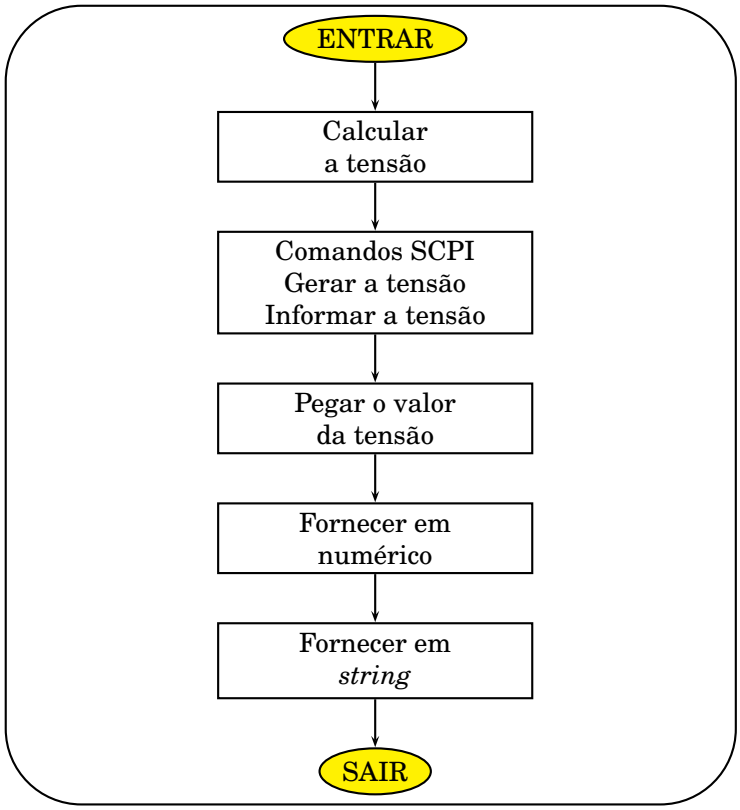


Figura E.19 - Fluxograma do SubVI **VOLT**

O código simplificado e o fluxograma das figuras E.20 e E.21 mostram como as tensões são geradas. Considere o exemplo em que VMAX = 3,0 e Passo = 0,6. O SubVI **ESCALA** calcula o número de execuções N = 6. O índice da estrutura varia de 0 a 5. Os valores das tensões geradas são calculados multiplicando o valor do passo pelo índice.

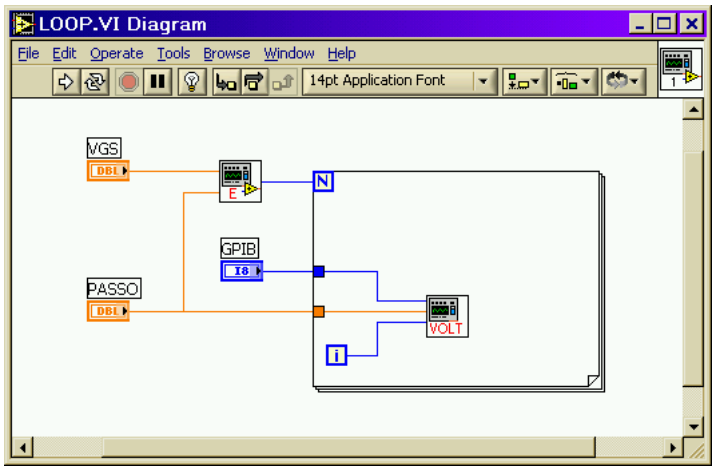


Figura E.20 - Gerar sequência de tensão

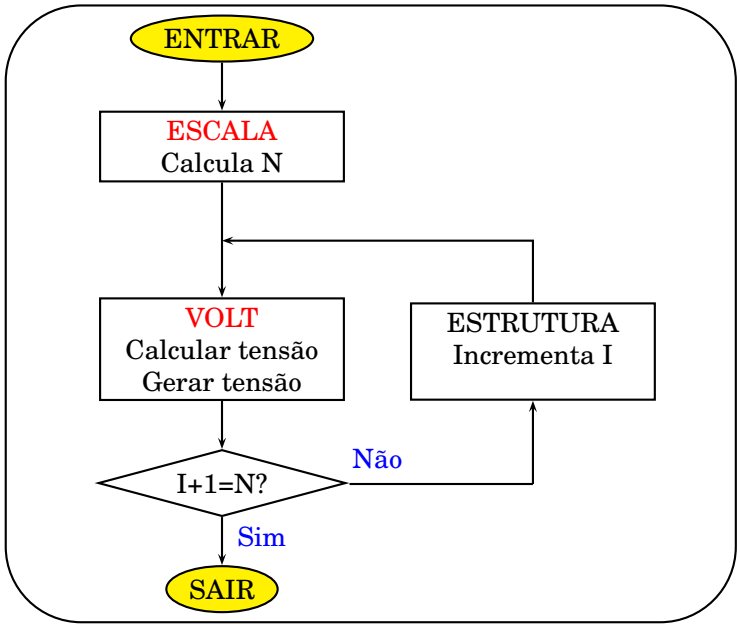


Figura E.21 - Fluxograma do exemplo

Serão feitas as seguintes operações:

$0{,}6 \times 0 = 0{,}0$ $0{,}6 \times 1 = 0{,}6$ $0{,}6 \times 2 = 1{,}2$

$0{,}6 \times 3 = 1{,}8$ $0{,}6 \times 4 = 2{,}4$ $0{,}6 \times 5 = 3{,}0$

Os comandos SCPI da tabela E.6 serão enviados:

Índice **I**	COMANDOS SCPI
0	:SOUR:VOLT +000.000 :SOUR:VOLT?
1	:SOUR:VOLT +000.600 :SOUR:VOLT?
2	:SOUR:VOLT +001.200 :SOUR:VOLT?
3	:SOUR:VOLT +001.800 :SOUR:VOLT?
4	:SOUR:VOLT +002.400 :SOUR:VOLT?
5	:SOUR:VOLT +003.000 :SOUR:VOLT?

Tabela E.6 - Comandos para gerar tensão

Sub-vi LER

Captura os valores medidos pelos instrumentos e organiza para ser exibido no gráfico e também para gravação em arquivo. Primeiramente o comando de medida READ? é enviado ao Keithley 2400, cujo endereço é obtido na entrada *GPIB 2400*. Em seguida a função GPIB *Receive* coleta o resultado. O instrumento fornece uma palavra com 5 números com 13 caracteres separados por vírgulas. Veja o exemplo abaixo. O primeiro valor é a tensão gerada V_{DS}, o segundo mostra a corrente consumida IDS, o terceiro é a resistência da carga, o quarto informa o tempo e o quinto o estado do instrumento.

+6.000000E-01,+1.000236E-04,+5.998584E+03,+7.282600E+01,+4.813200E+04

As funções *String Subset* separam os valores de tensão e corrente. Os valores de IDS e V_{DS} são transformados em numérico para ser enviado ao gráfico.
O comando de leitura FETCH? é enviado ao Eletrômetro , cujo endereço é obtido na entrada *GPIB 6517*. A resposta fornecida à função GPIB *Receive* possui o seguinte formato:

+05.18619E-06NADC,+0009733.920165secs,+56773rdng#

O valor da corrente na porta do transistor é separado pela função *String Subset*. O valor da tensão de polarização da porta, fornecido pelo SubVI **VOLT** é pego na entrada *Vg*. Com a constante *string* e as funções *Search and Replace String* forma-se a palavra mostrada abaixo que será gravada em arquivo.

VDS=+6.000000E-01V IDS=+1.000000E-04A Ig=+05.18619E-06A Vg=+002.000V

O código fonte pode ser estudado na figura E.22. A figura E.23 mostra o fluxograma.

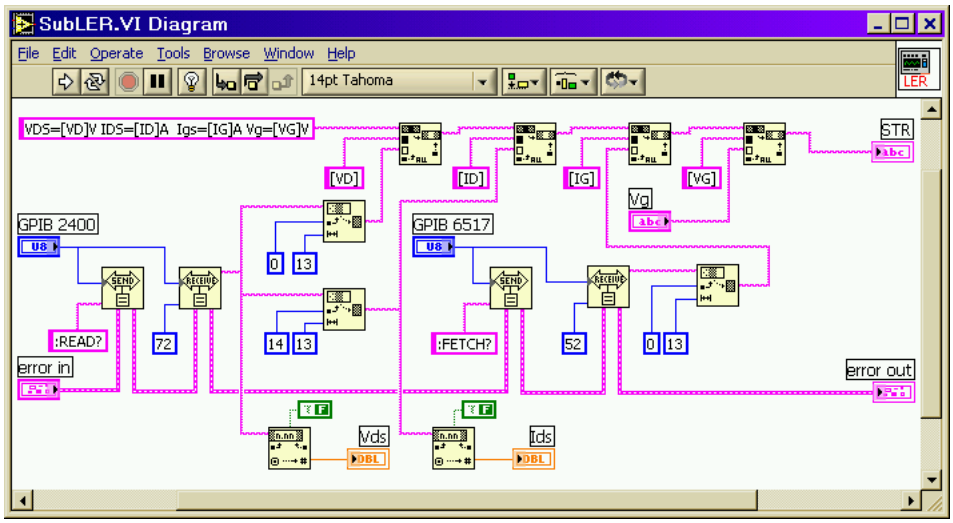


Figura E.22 - SubVI **LER**.

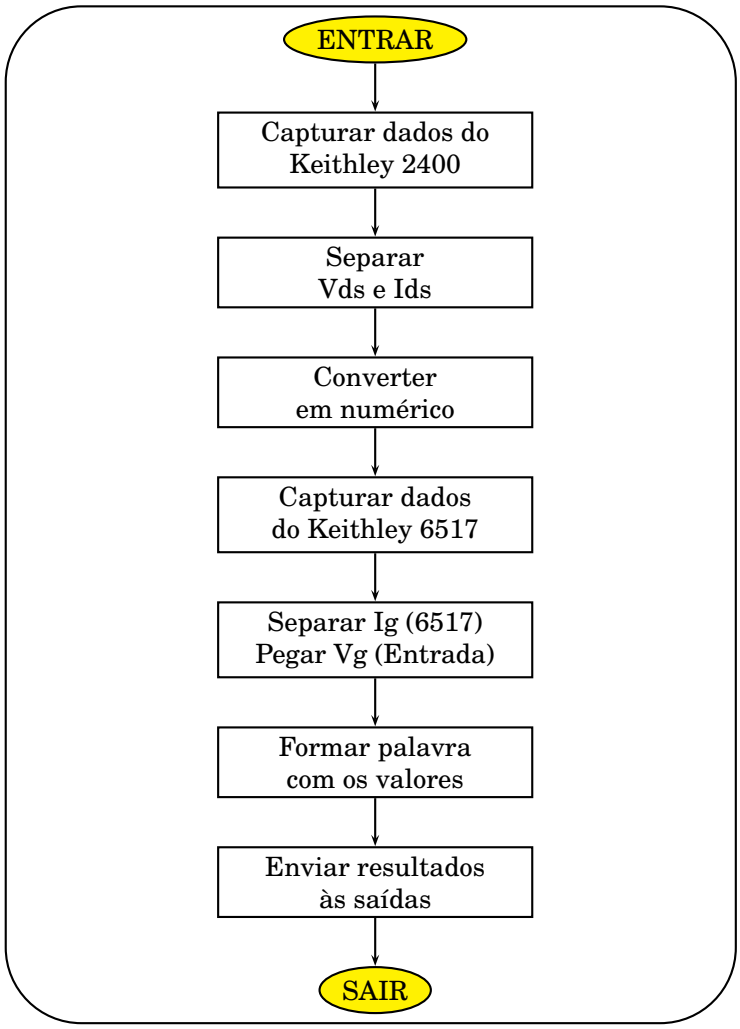


Figura E.23 - Fluxograma do SubVI **LER**

Sub-vi SALVAR

Desliga as fontes de tensão dos instrumentos. Anexa aos valores medidos a corrente de proteção. Finalmente grava tudo em um arquivo de texto. Na figura E.24 temos o código fonte. A figura E.25 mostra o fluxograma. A entrada *Caminho* recebe o nome do arquivo fornecido pelo usuário. O programa principal fornece um índice que é incrementado e enviado à saída *Incremento*. Com as funções *Path To String*, *Format Value* e *String To Path* formata-se o nome do arquivo que é fornecido à função *Write Characters To File* que cria o arquivo para gravação. Se o usuário fornecer o nome PMOS e fizer quatro medidas, elas serão gravadas nos arquivos PMOS-01.TXT, PMOS-02.TXT, PMOS-03.TXT, PMOS-04.TXT.

Na entrada *Arquivar* é fornecido o *array* com os resultados obtidos pelo SubVI **LER**. Os dados são formatados pela função *Array to Spreadsheet String* antes de serem enviados ao arquivo. A constante "\n" na entrada *delimiter* significa que o retorno é usado para separar as palavras. O comando :OUTP OFF é enviado simultaneamente aos dois instrumentos para que eles desliguem suas fontes de tensão. O valor da corrente de proteção IPROT é obtido do Keithley 2400 e anexado aos resultados, que são gravados no arquivo.

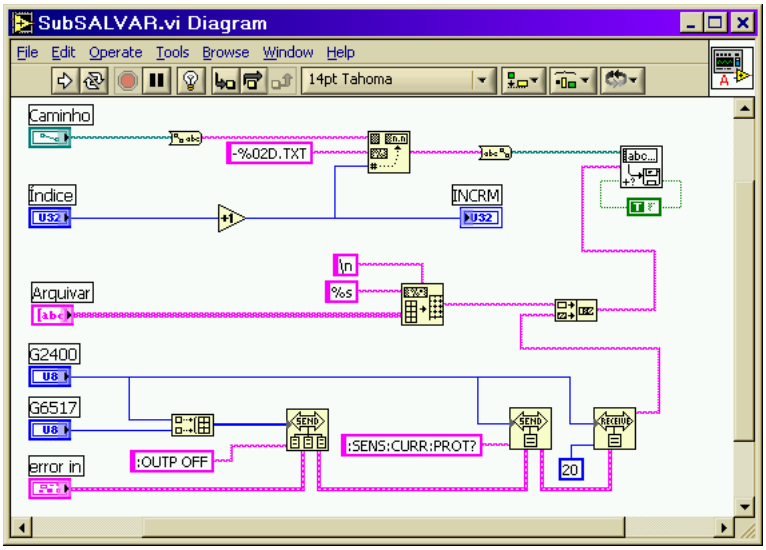


Figura E.24 - SubVI **SALVAR**.

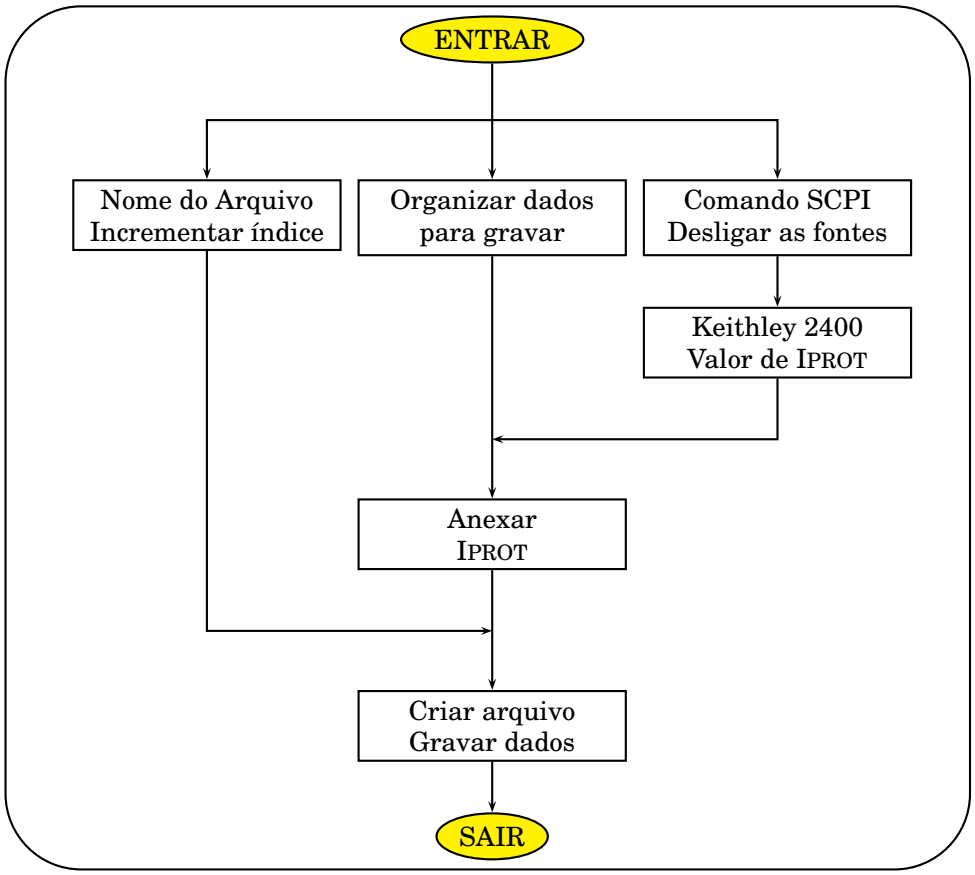


Figura E.25 - Fluxograma do SubVI **SALVAR**

SubVI GRÁFICO

Usado quando os programas estão no modo Carregar Arquivo. Lê o arquivo de texto e fornece os dados para o código principal. O programa IDSxVDS usa as saídas *I-Prot*, *GRF*, *VDS* e *VGS1*. Quando o usuário escolhe um arquivo, a saída *cancelled* da função *File Dialog* fornece *False* e a saída *path* fornece o caminho do arquivo ao SubVI. A entrada *MULT* é colocada em *True*. Nas figuras E.26 temos o código do programa IDSxVDS e na figura E.27 o SubVI.

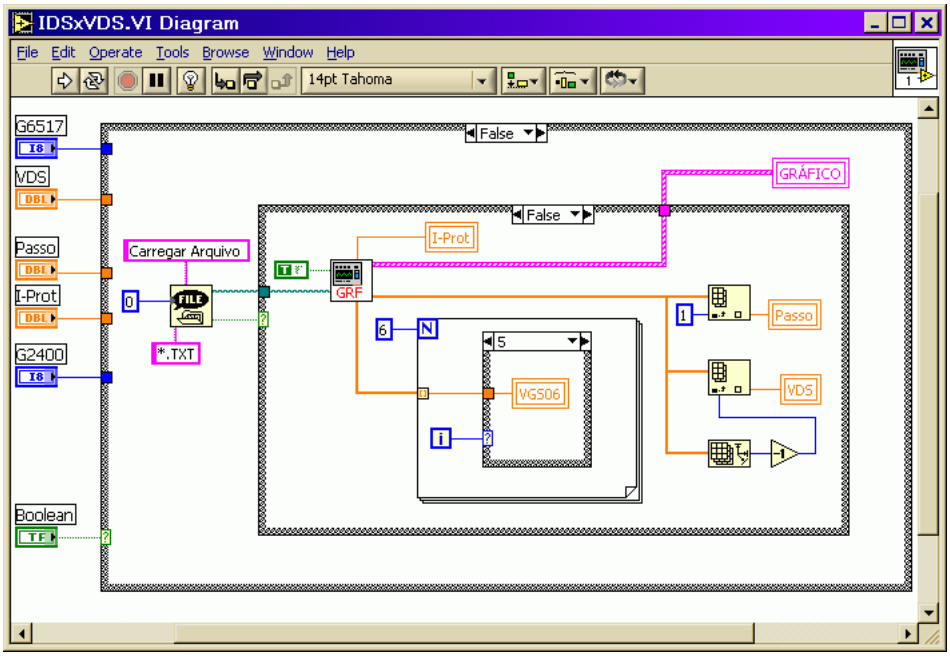


Figura E.26 - Programa IDSxVDS no modo Carregar Arquivo.

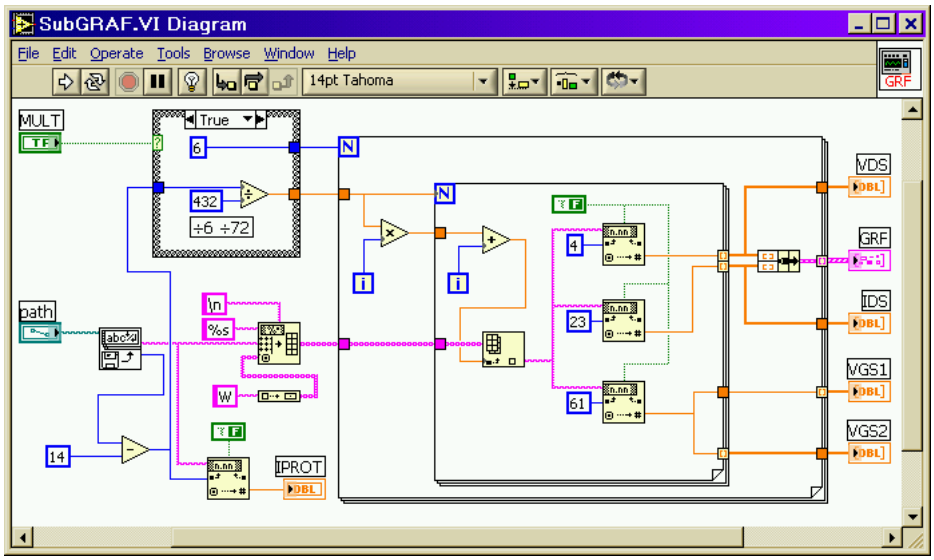


Figura E.27 - SubVI **GRÁFICO**.

A entrada *MULT* comanda uma estrutura *Case* que controla duas estruturas *For Loop*. É colocada em *True* pelo programa IDSxVDS com seis valores de V_{GS} e em *FALSE* pelo programa IDSxVGS para V_{DS} fixo através da entrada *MULT*. Pode ser observada na figura E.28.

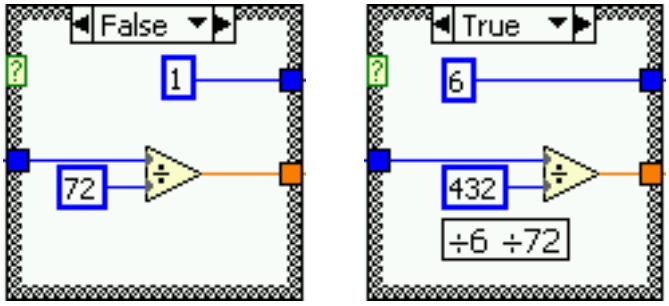


Figura E.28 - Estrutura CASE.

Dentro do SubVI, a função *Read Lines From File* lê arquivo e fornece seus dados na saída *line string* e o número total de caracteres na saída *mark after read*. Desse número é subtraído 14 que corresponde aos caracteres da corrente de proteção, colocada ao final do arquivo. Com o resultado é obtido o valor da corrente de proteção através da função *Frac/Exp String To Number* e também o número de medidas salvas no arquivo. Dividir por 432, dentro da estrutura *Case*, é o mesmo que dividir por 6 que é o número de valores de V_{GS} e depois dividir por 72 que é a quantidade de caracteres de cada linha. O resultado é o número de valores de V_{DS}'s para cada VGS.

A estrutura *For Loop* interna recebe o número de V_{DS}'s e a estrutura externa recebe o número seis. O número de V_{DS}'s também é multiplicado pelo índice da estrutura externa que varia de 0 a 5. O resultado é somado ao índice da estrutura interna e levado à entrada *index* da função *Index Array*. Ela fornece as linhas do arquivo. As funções *Frac/Exp String To Number* separam os valores de V_{DS}, IDS e VGS. Esses valores são entregues ao programa principal para serem exibidos no gráfico.

Para o programa IDSxVGS a estrutura *Case* é colocada em *False*. A estrutura *For Loop* é executada uma vez. A outra estrutura é executada o número correspondente ao de medidas gravadas no arquivo. As saídas usadas são *VDS*, *IDS*, *VGS2* e *I-Prot*. Nas figuras E.29 e E.30 podemos ver os códigos do programa e do SubVI.

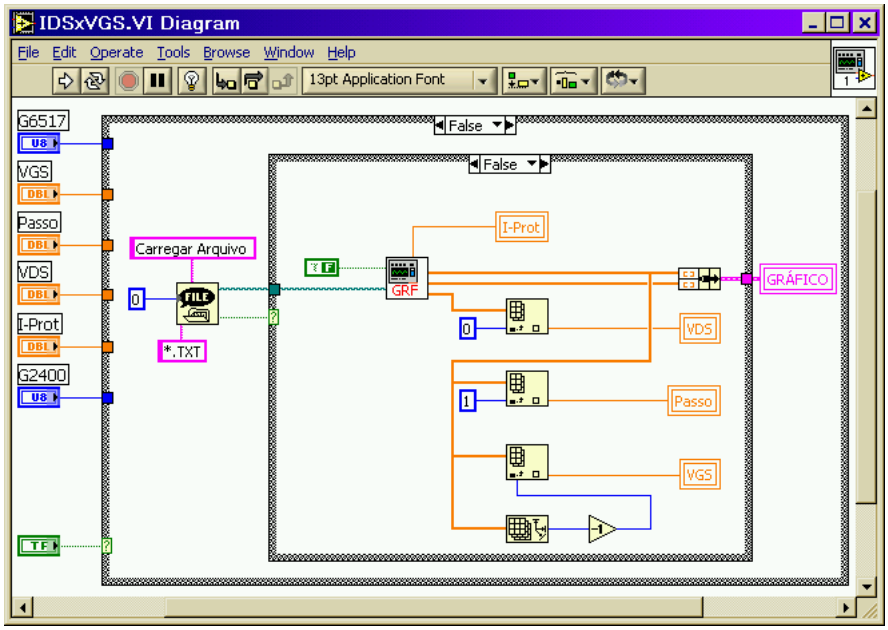


Figura E.29 - Programa IDSxVGS no modo Carregar Arquivo.

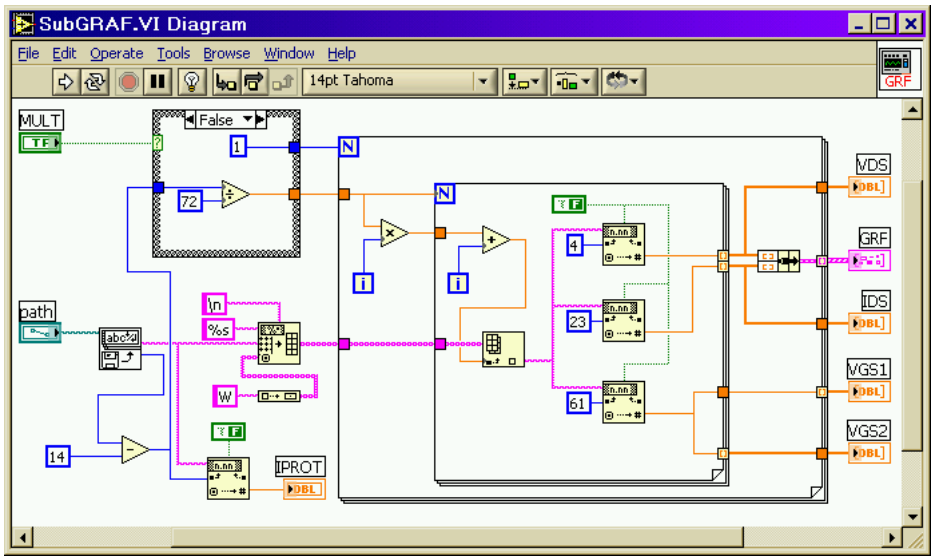


Figura E.30 - SubVI **GRÁFICO**.

Apêndice F - Parâmetros S

Usa-se dois programas, um que traça as curvas S_{11} e S_{21} e outro que traça as curvas S_{22} e S_{12}. Ambos controlam o Analisador de redes Agilent 9714ES através da interface GPIB.

Os programas possuem duas possibilidades de funcionamento. O modo de medida e o modo de arquivo. No modo de medida, os programas controlam os instrumentos, colhem os resultados das medidas, gravam em arquivo e traçam os gráficos. No modo de arquivo os programas carregam arquivos previamente gravados e traçam o gráfico para análise.

Os códigos dos programas usam no total sete SubVI's. Algums são usados apenas em um programa enquanto que outros são compartilhados. O SubVI **INICIAR S11** configura o instrumento para medir S_{11} e S_{21}. O SubVI **INICIAR S22** configura o instrumento para medir S_{22} e S_{12}. O SubVI **MHZ** determina a frequência do sinal de saída. Os dois SubVI's **LER** capturam os resultados medidos e formatam para serem exibidos na tela e gravado no arquivo. A única diferença está na *string* gerada. Um é usado pelo programa **S11S21** e o outro pelo programa **S22S12**. O SubVI **ARQUIVO** salva os dados em forma de texto e o SubVI **GRÁFICO** mostra os dados do arquivo no gráfico em forma de curvas. A figura F.1 identifica os SubVI's mostrando seus ícones.

Figura F.1 - SubVI's dos programas de parâmetros S

Programa S11S21

Tem como finalidade mostrar as curvas da potência refletida na entrada S_{11} e transmitida S_{21}. Na figura F.2 podemos ver a tela do programa. No campo *Endereço* esta o endereço GPIB do instrumento. Os campos *Inicial [kHz]* e *Final [kHz]* informa-se as frequências para que o instrumento faça a varredura. O campo *Passo* determina o saltos durante a varredura. No campo *Nível [DBm]* o usuário determina a potência de saída em dBm. Pode-se também escolher o nome e o diretório em que o arquivo será gravado.

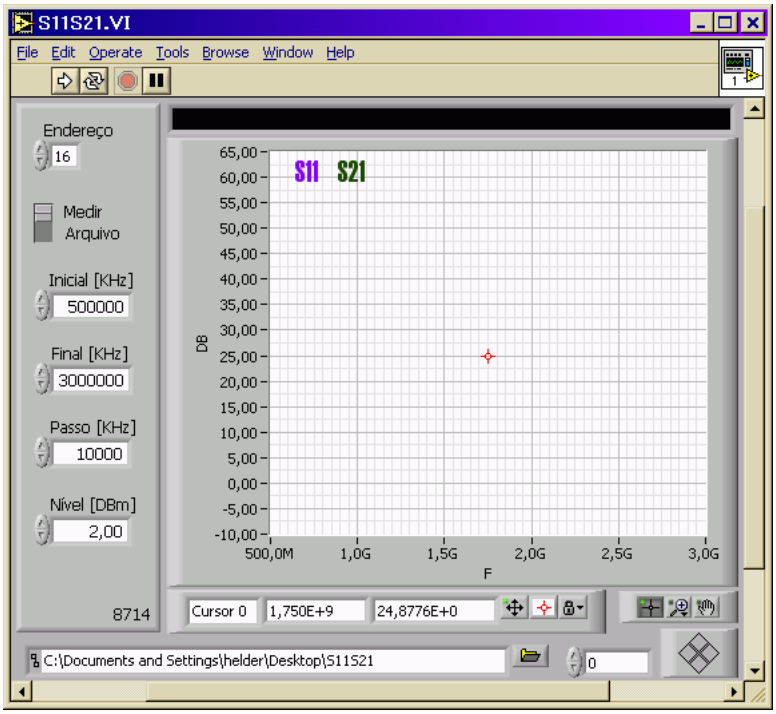


Figura F.2 - Programa **S11S21**

A figura F.3 mostra o código do programa feito em LabVIEW no modo Medir. A figura F.4 mostra o fluxograma. Primeiramente o SubVI **INICIAR S11** configura o instrumento para traçar as curvas de S_{11} e S_{21} usando os dados fornecidos pelo usuário. O número de execuções para a estrutura *For Loop* fazer a varredura também é calculado. Dentro da estrutura o SubVI **MHz** calcula e gera a frequência de saída e o SubVI **LER** captura os resultados das medidas, formata e exibe na tela preta que é o indicador *Str*. Os resultados são transformados em *array* pela estrutura. A função *Wait(ms)* determina um tempo de 20 ms entre uma medida e outra.

Ao sair da estrutura *For Loop* o programa envia o *array* com os resultados ao SubVI **ARQUIVO** que os salva em arquivo.

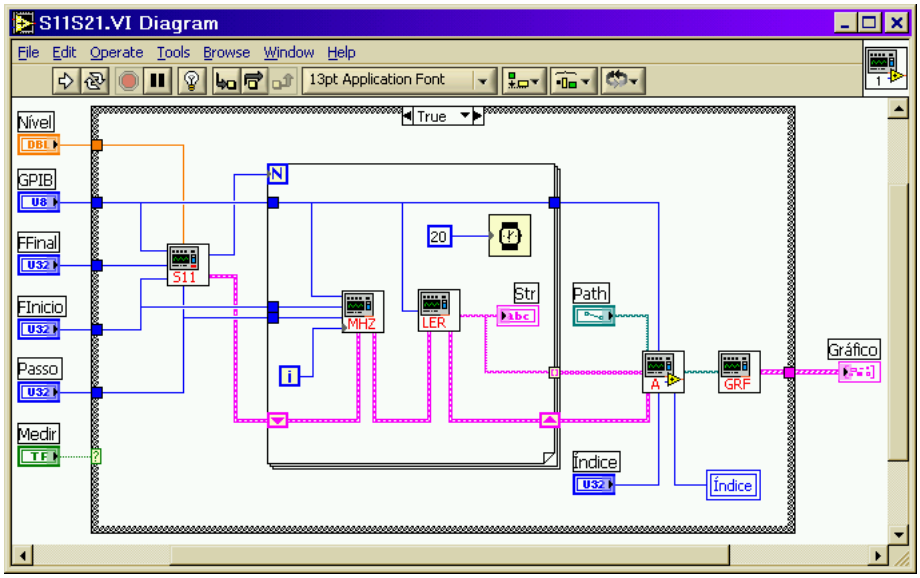


Figura F.3 - Código do programa **S11S21**

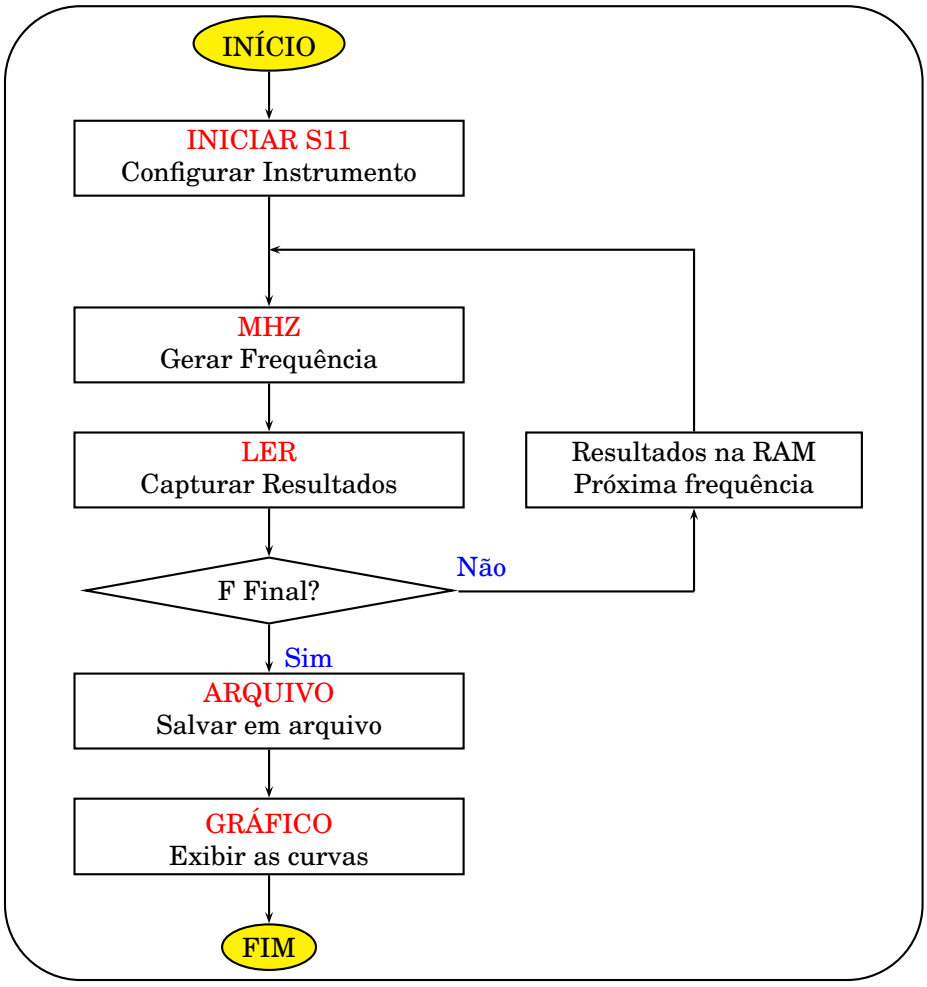


Figura F.4 - Processo para obtenção de parâmetros S

O indice é incrementado. Se o usuário fornecer o nome FiltroRF e fizer quatro medidas elas serão salvas como FiltroRF-01.TXT, FiltroRF-02.TXT, FiltroRF-03.TXT e FiltroRF-04.TXT. Em seguida o SubVI **GRÁFICO** lê o arquivo e exibe as curvas no gráfico.

Quando executado no modo Carregar Arquivo, a função *File Dialog* abre a caixa de diálogo do Windows vista na figura F.5. O usuário pode escolher um arquivo previamente gravado para ser exibido no gráfico.

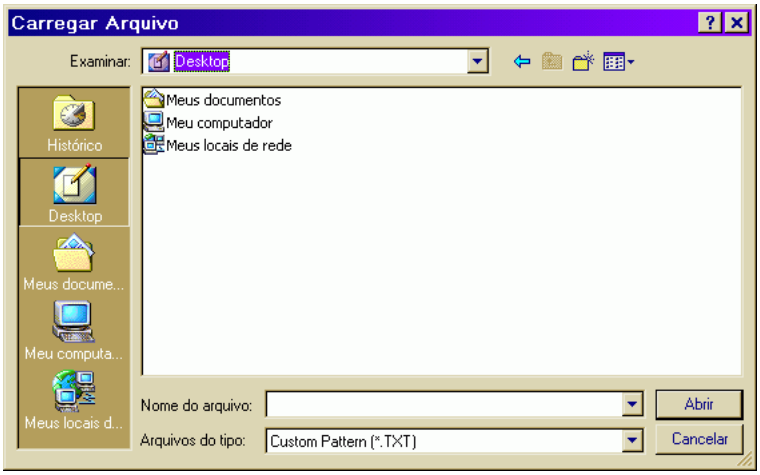


Figura F.5 - Escolher um arquivo

Quando o usuário clica em *Cancelar* a saída *Cancelled* coloca a estrutura em *True*. O gráfico é zerado, como mostra a figura F.6.

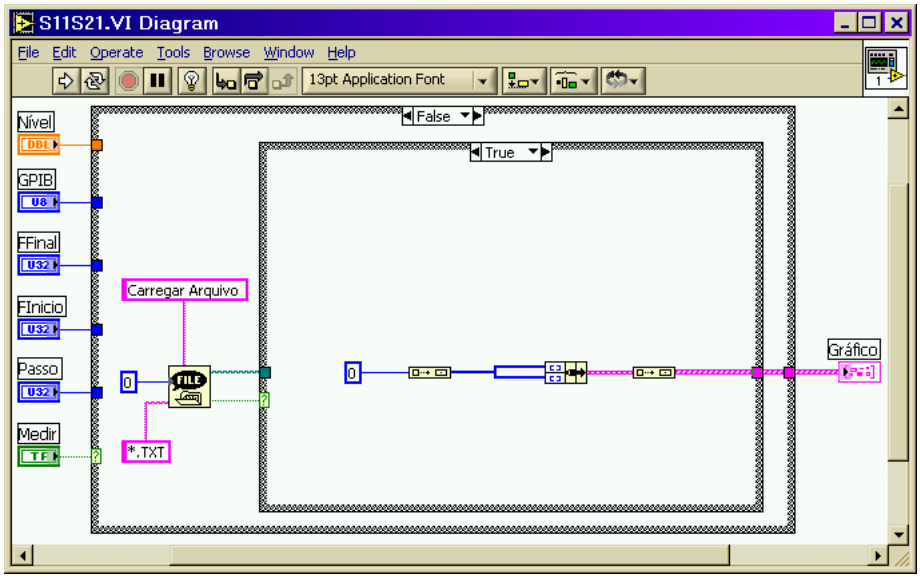


Figura F.6 - Programa **S11S21** zera o gráfico.

Quando um arquivo é escolhido o caminho é levado ao SubVI **GRÁFICO**, como mostra a figura F.7. Na figura F.8 temos o fluxograma para o modo Carregar Arquivos.

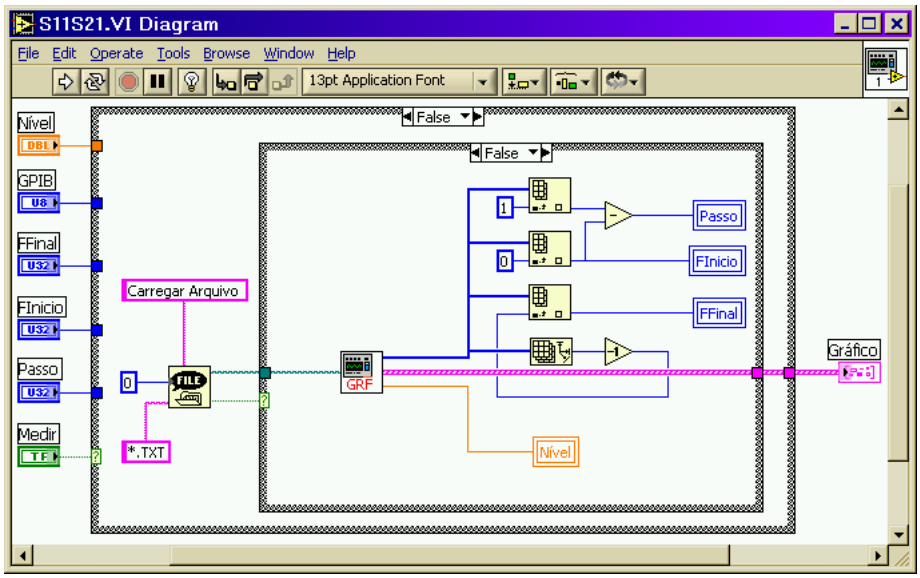


Figura F.7 - Programa **S11S21** no modo Carregar Arquivo

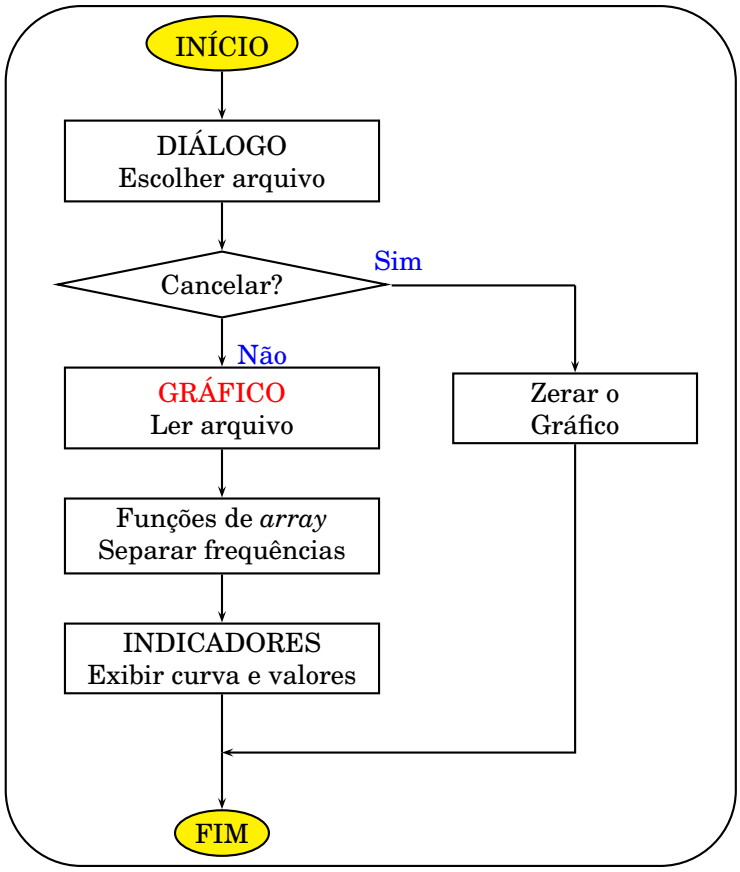


Figura F.8 - Fluxograma do modo Carregar Arquivo

O SubVI fornece um *array* com os valores de todas as frequências geradas. O valor do *Passo* é obtido subtraindo o valor da segunda posição, índice 1 do valor da frequência de início no índice zero. O a posição de *FFinal*, N-1, é obtido subtraindo um da saída da função *Array Size*. Os valores das frequências e do nível de potência são escritos nos campos através de variáveis locais.

Programa S22S12

Tem como finalidade mostrar as curvas da potência refletida na saída S_{22} e transmitida reversa S_{12}. Na figura F.9 podemos ver a tela do programa. No campo *Endereço* esta o endereço GPIB do instrumento. Os campos *Inicial [kHz]* e *Final [kHz]* informa-se as frequências para que o instrumento faça a varredura. O campo *Passo* determina o saltos durante a varredura. No campo *Nível [DBm]* o usuário determina a potência de saída em dBm. Pode-se também escolher o nome e o diretório em que o arquivo será gravado.

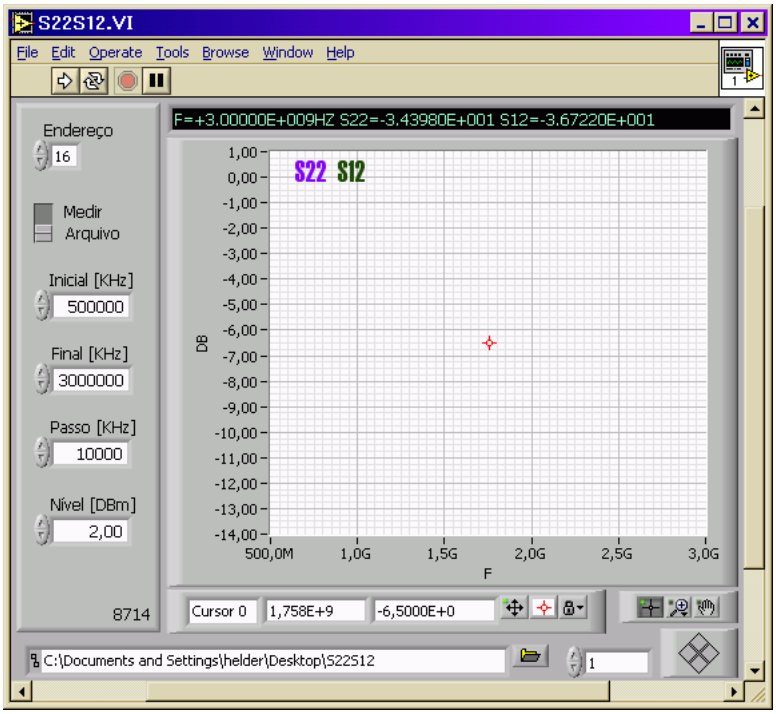


Figura F.9 - Programa **S22S12**

A figura F.10 mostra o código do programa feito em LabVIEW no modo Medir. A figura F.11 mostra o fluxograma. Primeiramente o SubVI **INICIAR S22** configura o instrumento para traçar as curvas de S_{22} e S_{12} usando os dados fornecidos pelo usuário. O número de execuções para a estrutura *For Loop* fazer a varredura também é calculado. Dentro da estrutura o SubVI **MHz** calcula e gera a frequência de saída e o SubVI **LER** captura os resultados das medidas, formata e exibe na tela preta que é o indicador *Str*. Os resultados são transformados em *array* pela estrutura. A função *Wait(ms)* determina um tempo de 20 ms entre uma medida e outra.

Ao sair da estrutura *For Loop* o programa envia o *array* com os resultados ao SubVI **ARQUIVO** que os salva em arquivo.

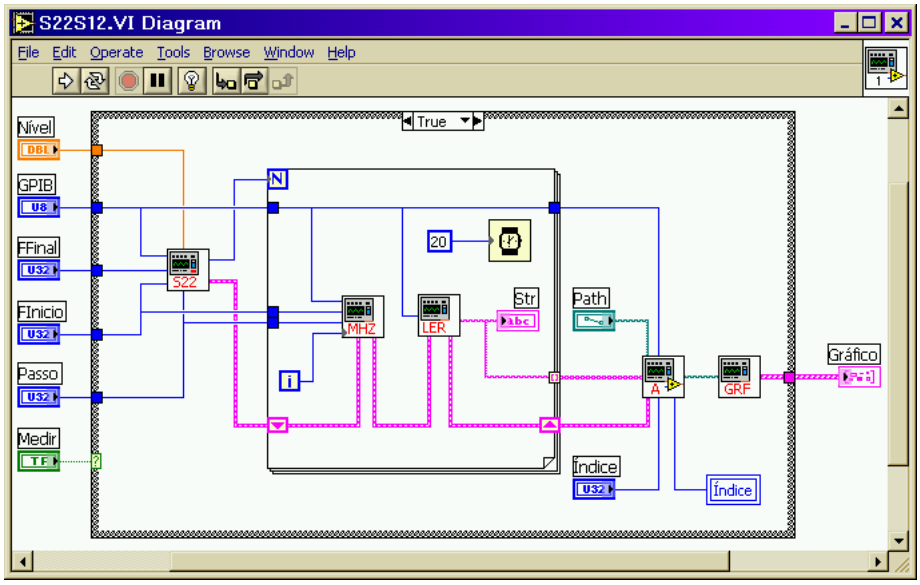


Figura F.10 - Código do programa **S22S12**

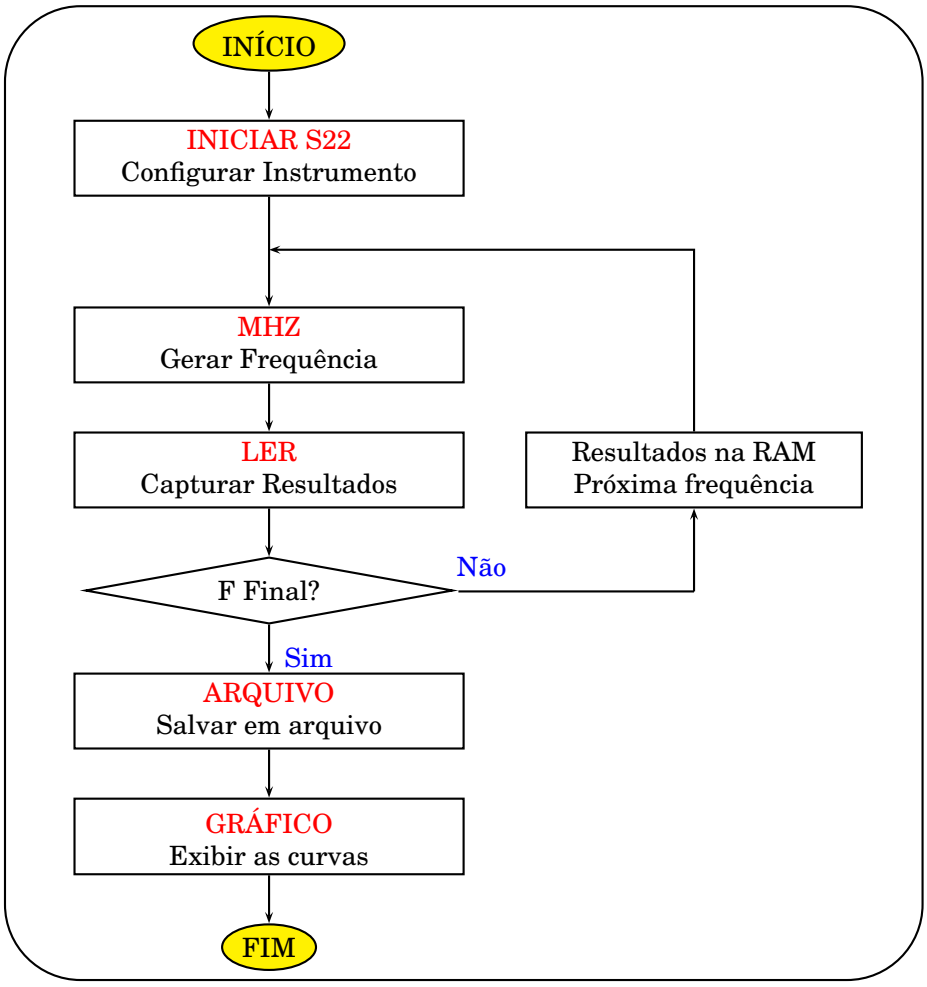


Figura F.11 - Processo para obtenção de parâmetros S

O indice é incrementado. Se o usuário fornecer o nome FiltroRF e fizer quatro medidas elas serão salvas como FiltroRF-01.TXT, FiltroRF-02.TXT, FiltroRF-03.TXT e FiltroRF-04.TXT. Em seguida o SubVI **GRÁFICO** lê o arquivo e exibe as curvas no gráfico.

Quando o usuário clica em *Cancelar* a saída *Cancelled* coloca a estrutura em *True*. O gráfico é zerado, como mostra a figura F.12. Quando um arquivo é escolhido o caminho é levado ao SubVI **GRÁFICO**, como mostra a figura F.13. Na figura F.14 temos o fluxograma para o modo Carregar Arquivos.

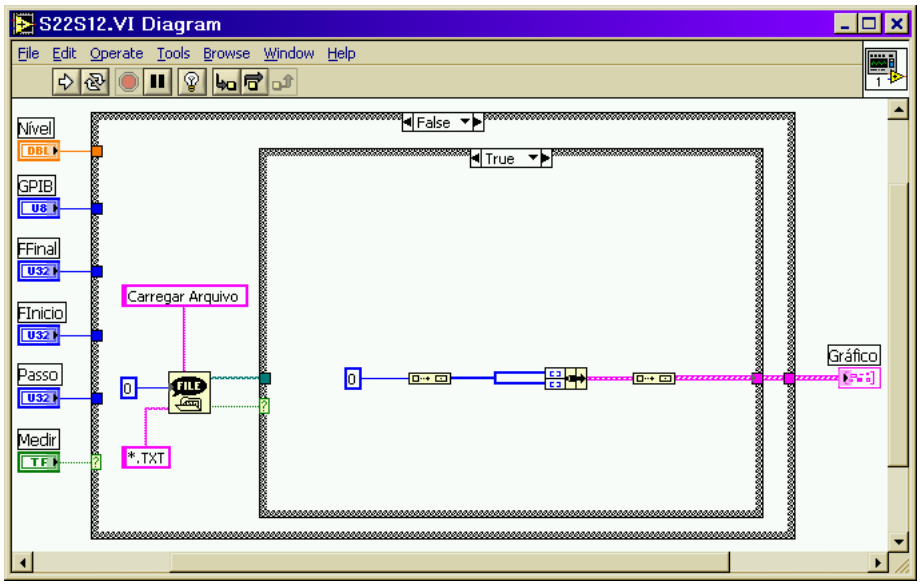


Figura F.12 - Programa **S22S12** zera o gráfico

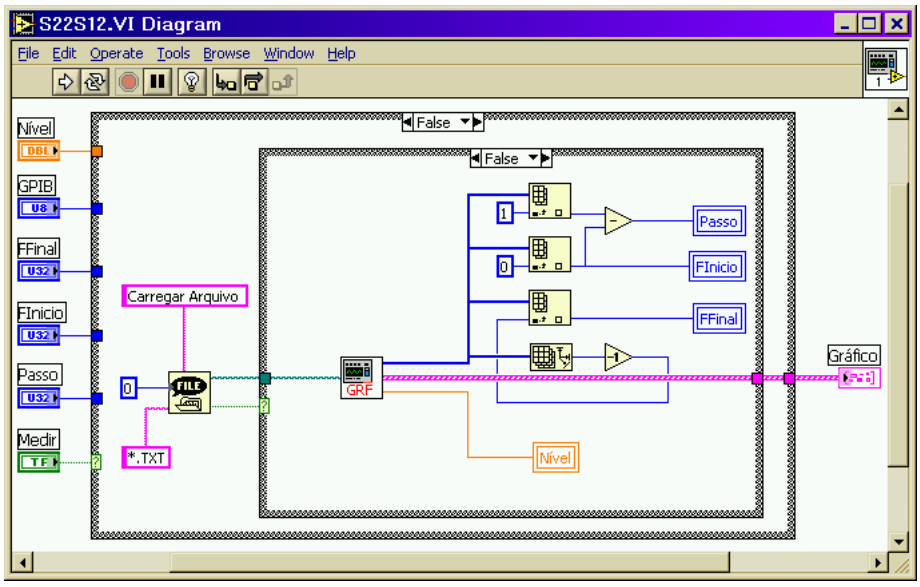


Figura F.13 - Programa **S11S21** no modo Carregar Arquivo

O SubVI fornece um *array* com os valores de todas as frequências geradas. O valor do *Passo* é obtido subtraindo o valor da segunda posição, índice 1 do valor da frequência de início no índice zero. O a posição de *FFinal*, N-1, é obtido subtraindo um da saída da função *Array Size*. Os valores das frequências e do nível de potência são escritos nos campos através de variáveis locais.

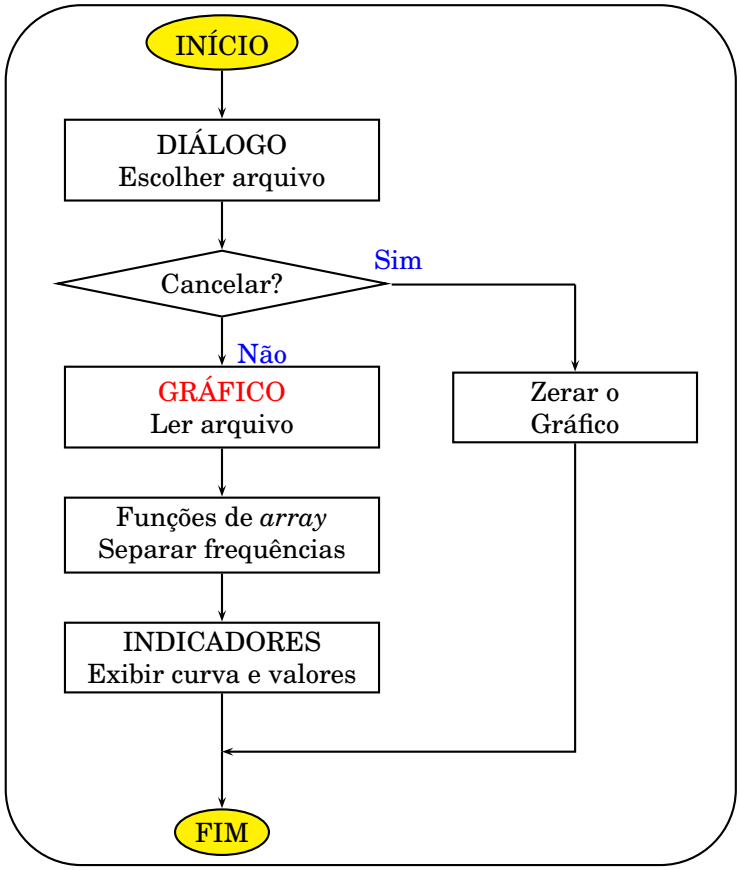


Figura F.14 - Fluxograma do modo Carregar Arquivo

SubVI's INICIAR

São dois, um é usado pelo programa **S11S21** e configura o instrumento para medir os parâmetros S_{11} e S_{21}. O outro é usado pelo programa **S22S12** e configura o instrumento para medir os parâmetros S_{22} e S_{12}. As figuras F.15 e F.16 mostram os códigos em LabVIEW. Os valores nas entradas *NÍVEL*, *FFINAL*, *FINÍCIO*, e *PASSO*, fornecidos pelo usuário são transformados em *string* e inseridos nos comandos SCPI pelas funções *Search and Replace String*. Os comandos são enviados pela função GPIB *Send*. Antes do programa sair do SubVI, a estrutura *For Loop* com a função de tempo *Wait(ms)* faz com que ele espere um tempo para que o instrumento possa se configurar.

O número de execuções para a estrutura *For Loop* do código principal também é calculado. É obtido com a operação

N = ((FFINAL - FINÍCIO) / PASSO) + 0,6

O resultado é arredondado para inteiro quando passa pela saída *NLOOPS*. Suponha que o usuário forneça as frequências Inicial = 600 kHz, Final = 1600 kHz e Passo = 200 kHz. As frequências geradas serão:

600 kHz : 800 kHz : 1000 kHz

1200 kHz : 1400 kHz : 1600 kHz

Como são seis frequências, será calculado N = 6. O cálculo ficará assim:

1600 - 600 = 1000 1000 / 200 = 5 5 + 0,6 = 5,6

Arredondando para inteiro, **N = 6**

As tabelas F.1 e F.2 explicam os comandos enviados ao instrumento. As palavras **[DB]**, **[F2]**, e **[F1]** são substituídas pelos valores fornecidos pelo usuário. São muito semelhantes. A diferença está nas linhas 2, 6 e 8. É importante notar que uma das funções do ponto e vírgula é usado para inserir mais de um comando na mesma linha. Nas tabelas os comandos estão separados para facilitar o entendimento.

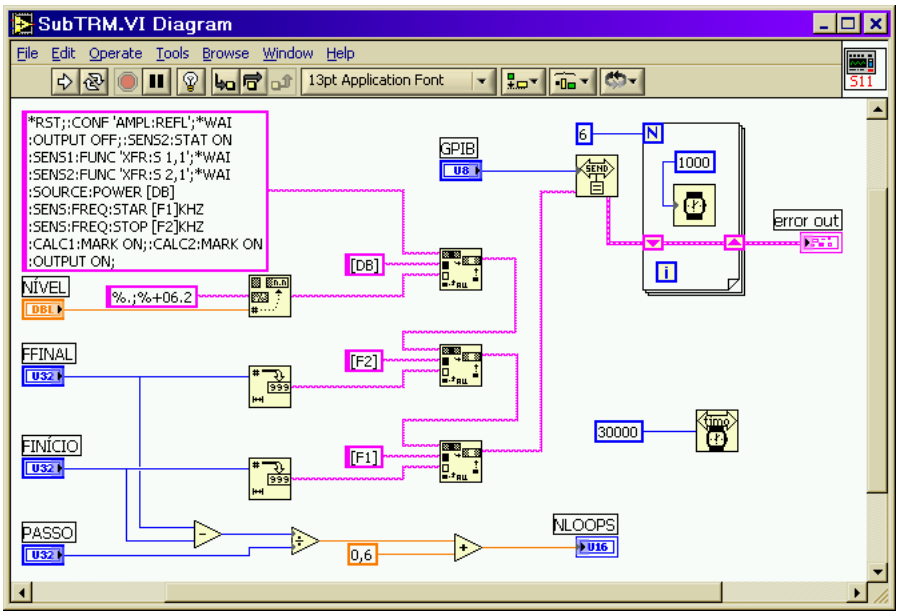


Figura F.15 - SubVI **INICIAR 1**.

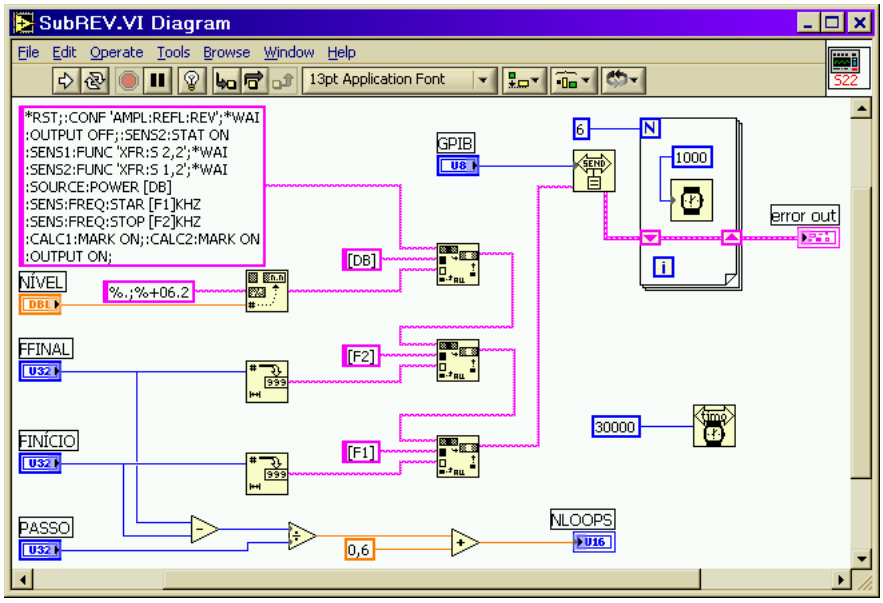


Figura F.16 - SubVI **INICIAR 2**

*RST	Reset - Instrumento na configuração padrão.
:CONF 'AMPL:REFL'	Modo amplificador, potência refletida na porta 1.
*WAI	Aguardar execução dos comandos pendentes.
:OUTPUT OFF	Desligar a saída de sinal.
:SENS2:STAT ON	Ligar o canal 2.
:SENS1:FUNC 'XFR:S 1,1'	O canal 1 irá medir S_{11}.
*WAI	Aguardar execução dos comandos pendentes.
:SENS2:FUNC 'XFR:S 2,1'	O canal 2 irá medir S_{21}.
*WAI	Aguardar execução dos comandos pendentes.
:SOURCE:POWER [DB]	Definir a potência do sinal da saída.
:SENS:FREQ:STAR [F1] kHz	Definir a frequência de início da varredura.
:SENS:FREQ:STOP [F2] kHz	Definir a frequência final para a varredura.
:CALC1:MARK ON	Ativar o marcador no canal 1.
:CALC2:MARK ON	Ativar o marcador no canal 2.
:OUTPUT ON	Ativar a saída de sinal.

Tabela F.1 - Configurar Analisador para medir S_{11} e S_{21}

*RST	Reset - Instrumento na configuração padrão.
:CONF 'AMPL:REFL:REV'	Modo amplificador, potência refletida na porta 2.
*WAI	Aguardar execução dos comandos pendentes.
:OUTPUT OFF	Desligar a saída de sinal.
:SENS2:STAT ON	Ligar o canal 2.
:SENS1:FUNC 'XFR:S 2,2'	O canal 1 irá medir S_{22}.
*WAI	Aguardar execução dos comandos pendentes.
:SENS2:FUNC 'XFR:S 1,2'	O canal 2 irá medir S_{12}.
*WAI	Aguardar execução dos comandos pendentes.
:SOURCE:POWER [DB]	Defir da potência do sinal da saída.
:SENS:FREQ:STAR [F1] kHz	Definir a frequência de início da varredura.
:SENS:FREQ:STOP [F2] kHz	Definir a frequência final para a varredura.
:CALC1:MARK ON	Ativar o marcador no canal 1.
:CALC2:MARK ON	Ativar o marcador no canal 2.
:OUTPUT ON	Ativar a saída de sinal.

Tabela F.2 - Configurar Analisador para medir S_{22} e S_{12}.

SubVI MHz

Trabalha dentro da estrutura *For Loop* do código principal. Observe as figuras F.3 e F.10. Calcula a frequência em que serão medidos os parâmetros e envia o comando SCPI que posiciona o marcador nessa frequência. O cálculo é feito usando o índice da estrutura da seguinte forma:

$$F = FInício + (Índice \times Passo)$$

O resultado é transformado em *string* e inserido no comando SCPI substituindo a palavra [FF]. O processo pode ser visto na figura F.17, que mostra o código.

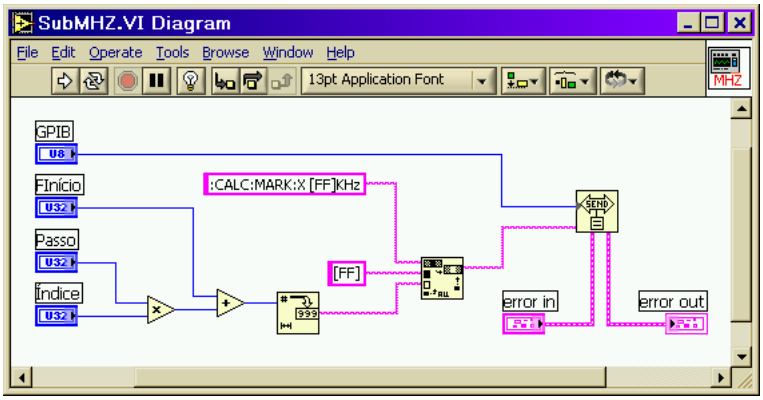


Figura F.17 - SubVI **MHZ**.

Suponha que o usuário forneça as seguintes frequências: Inicial = 600 kHz, Final = 1600 kHz e Passo = 200 kHz. O SubVI **INICIAR** calcula **N = 6**. Como o índice da estrutura varia de 0 a N-1, serão gerados os comandos vistos na tabela F.3.

Índice **I**	COMANDOS SCPI
0	:CALC:MARK:X 600 KHz
1	:CALC:MARK:X 800 KHz
2	:CALC:MARK:X 1000 KHz
3	:CALC:MARK:X 1200 KHz
4	:CALC:MARK:X 1400 KHz
5	:CALC:MARK:X 1600 KHz

Tabela F.3 - Comandos para posicionar o marcador

SubVI LER

São duas uma para o **S11S21** e a outra para o **S22S12**. Capturam os valores medidos pelo instrumento e organiza para ser exibido no gráfico e também para gravação em arquivo. A única diferença está nas *strings* de saída, que identificam os parâmetros medidos.

O comando ":CALC1:MARK:X?;Y?" é primeiramente enviado ao analisador. Obtém-se a frequência e o valor do parâmetro S medido no canal 1 com a função GPIB *RECEIVE*. Em seguida, com o comando ":CALC2:MARK:Y?" obtém-se o valor do parâmetro medido no canal 2. Os valores são separados pelas funções *String Subset* e inseridos na *string* de saída pelas funções *Search and Replace String*. Note nas figuras F.18 e F.19 que próximo à saída *DB* a última função *Search and Replace String* que retira o excesso de zeros. Os resultados entregues às saídas das dois SubVI's possuem o seguinte formato:

F=+7.05000E+008HZ S11=-4.99170E+000 S21=-3.25652E-002

F=+7.05000E+008HZ S22=+5.36617E+000 S12=+6.40435E-002

A figura F.20 mostra o fluxograma, que é válido para os dois códigos.

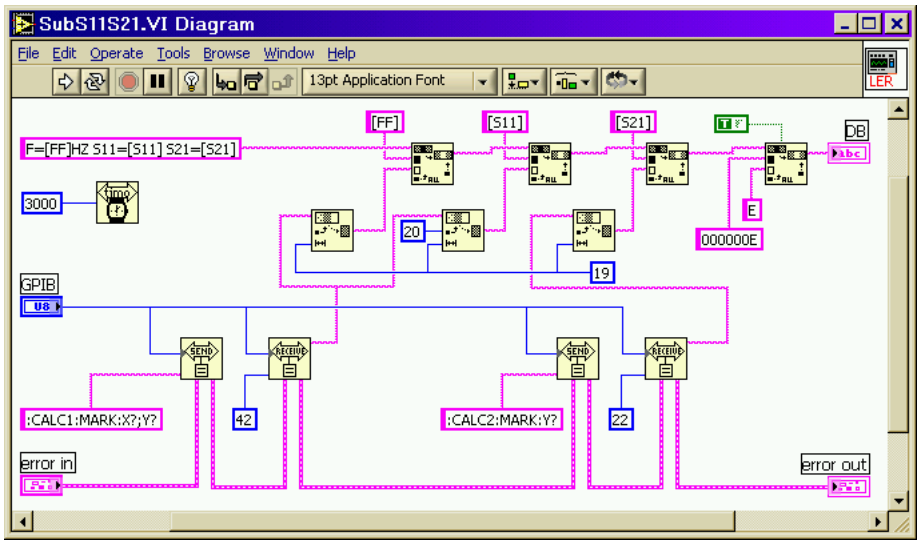


Figura F.18 - SubVI **LER S11S21**.

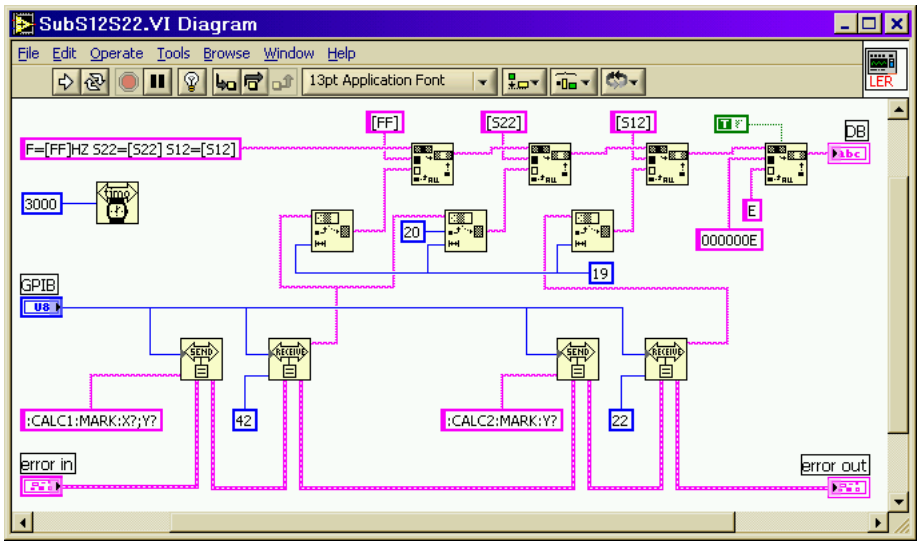


Figura F.19 - SubVI **LER S22S12**.

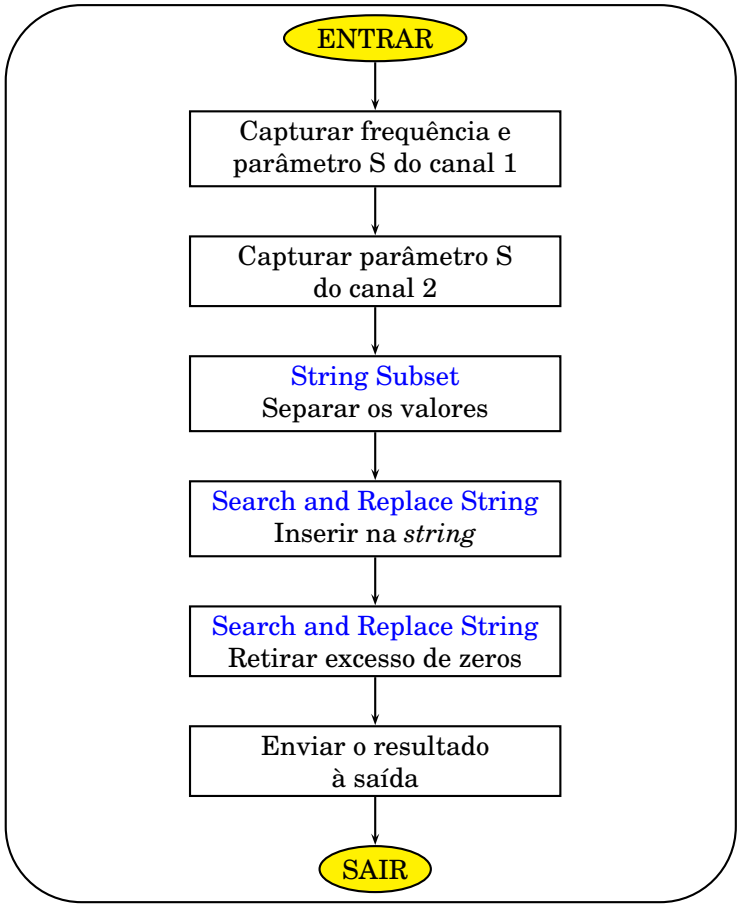


Figura F.20 - Fluxograma do SubVI **LER**

SubVI ARQUIVO

Grava os valores medidos o nível do sinal de saída em um arquivo de texto. Na figura F.21 temos o código fonte e na figura F.22 temos o fluxograma.

A entrada *Caminho* recebe o nome do arquivo fornecido pelo usuário. O programa principal fornece um índice que é incrementado e enviado à saída *Inc*. Com as funções *Path To String*, *Format Value* e *String To Path* formata-se o nome do arquivo que é fornecido à função *Write Characters To File* que cria o arquivo para gravação. Se o usuário fornecer o nome FILTRO e fizer quatro medidas, elas serão gravadas nos arquivos FILTRO-01.TXT, FILTRO-02.TXT, FILTRO-03.TXT, FILTRO-04.TXT.

Na entrada *Dados* é fornecido o *array* com os resultados obtidos pelo SubVI **LER**. Os dados são formatados pela função *Array to Spreadsheet String* antes de serem enviados ao arquivo. A constante "\n" na entrada *delimiter* significa que o retorno é usado para separar as palavras. O comando ":ABORT" coloca o instrumento no estado ocioso. Com o comando ":SOURCE:POWER?" obtém-se o nivel do sinal fornecido na saída. Esse valor é anexado aos resultados, que são gravados no arquivo.

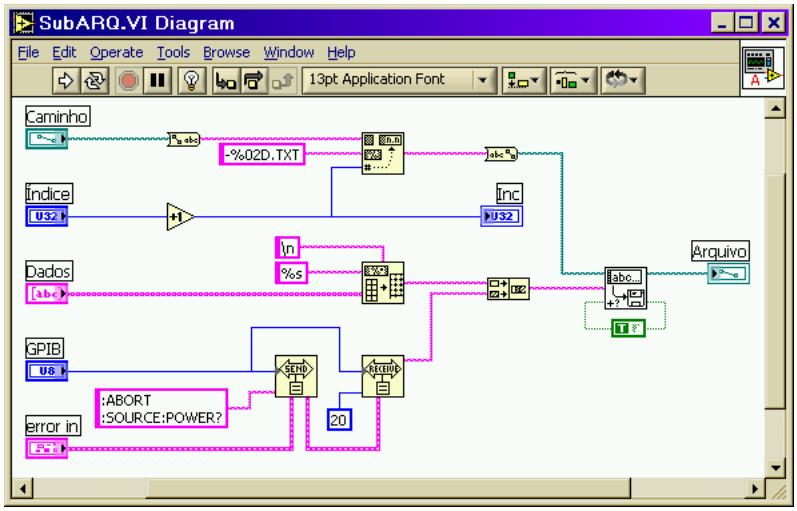


Figura F.21 - SubVI **ARQUIVO**.

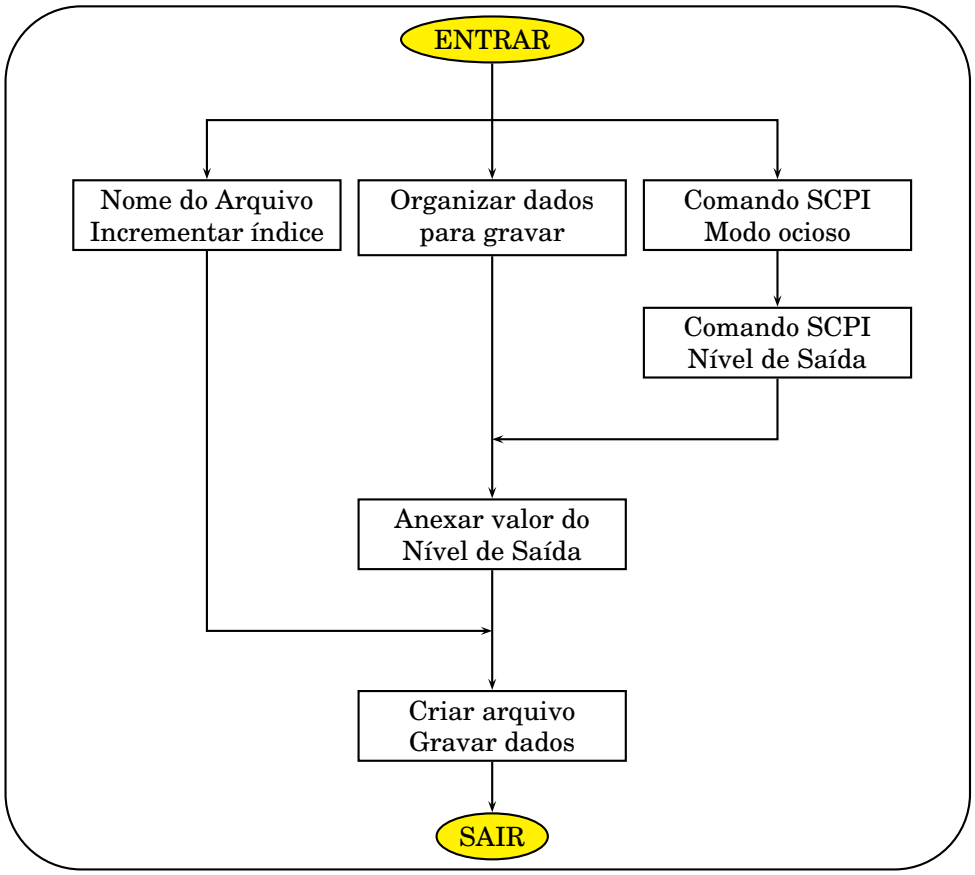


Figura F.22 - Fluxograma do SubVI **SALVAR**

SubVI GRÁFICO

O arquivo de texto gerado possui o formato mostrado abaixo. O valor na última linha é a potência da saída. O arquivo do programa **S22S12** possui o mesmo formato. Muda apenas a identificação dos parâmetros para "S_{22}=" e "S_{12}=".

F=+2.76500E+009HZ S11=-7.30091E+000 S21=-1.65609E-001
F=+2.77000E+009HZ S11=-7.54135E+000 S21=-1.38826E-001
F=+2.77500E+009HZ S11=-7.79396E+000 S21=-1.07522E-001
F=+2.78000E+009HZ S11=-7.94352E+000 S21=-6.52609E-002
F=+2.78500E+009HZ S11=-8.02439E+000 S21=-1.56956E-002
F=+2.79000E+009HZ S11=-8.00730E+000 S21=+1.93913E-002
F=+2.79500E+009HZ S11=-7.76165E+000 S21=+2.06957E-002
F=+2.80000E+009HZ S11=-7.51600E+000 S21=+2.20000E-002
+4.02000000000E+000

A estrutura *While Loop* controla a função *Read Characters From File*. Cada linha é fornecida às funções *Frac/Exp String To Number* que separam e convertem para numérico os valores da frequência, do parâmetro de espalhamento e do nível de potência.

Todas as linhas exceto a última possuem 55 caracteres, incluindo o retorno que não é visível. Dentro da estrutura *While Loop* o índice é multiplicado por 55 e ligado na entrada *offset* da função *Read Characters From File* (RCFF). Por isso a cada execução da estrutura (*Loop*), a leitura começa na próxima linha. São lidos 77 caracteres por execução o que inclui parte da linha posterior. Quando chega na penúltima linha a função detecta o fim do arquivo (eof) e para a estrutura. Apesar disso, o nível de potência na última linha é lido.

São feitas duas leituras, uma para a coluna de S_{11} e outra para a coluna de S_{22}. Isso é executato pela estrutura *For Loop* com "N=2". O índice é multiplicado por 18, que é o número de caracteres das colunas, incluindo o espaço e o retorno na última. Em seguida, dentro da estrutura *While Loop* é somado 22 e levado à entrada *offset* da função *Frac/Exp String To Number*.

Na primeira execução o índice é zero. A função começa a ler apartir do vigésimo segundo caracter (0+22), o que corresponde ao valor de S_{11}. Na segunda execução o índice é igual a dois. A função começa a ler apartir do quadragésimo caracter (18+22), o que corresponde ao valor de S_{21}.

A frequência é transformada de Hz para kHz multiplicando por 1E-3. Os valores são indexados e o *array* é entregue à saída *Freq*. Os valores das frequências e parâmetros de espalhamento são organizados em *array* de *clusters* com o auxílio da função *Bundle*. Esse formato é o padrão de entrada do gráfico e é transferido ao código principal pela saída *GRF*. O valor da potência de saída é fornecido à saída *Nível*. Repare que o fio não é indexado em nenhuma estrutura, de maneira que apenas o valor da última linha do arquivo é fornecido à saída. Na figura F.23 temos o código.

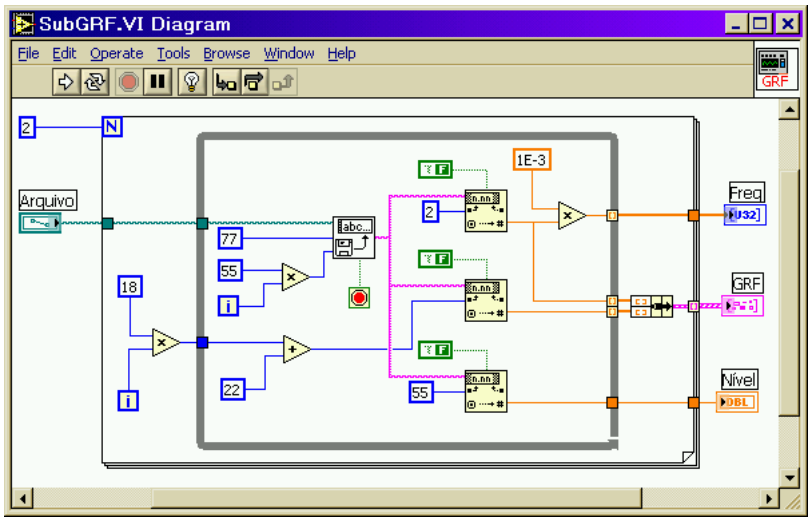


Figura F.23 - SubVI **GRÁFICO**.