**SOFTWARE DEVELOPMENT FOR PARAMETER EXTRACTION OF GaN HEMT and GaAS MESFET**

A PROJECT REPORT

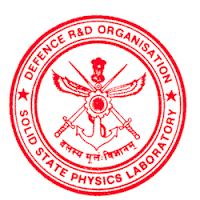
*Submitted By*

**PRANKUR TIWARI**

**(MASTER IN COMPUTER APPLICATION)**

**Under the guidance of Mr. A. K. Pandey, Scientist ‘G’**

**AT**



DRDO-SSPL

Lucknow Road, Banarsi Das Estate

Timarpur, New Delhi

Delhi - 110054

(27th May – 11th July, 2019)

**ACKNOWLEDGEMENT**

*I would like to express my deepest appreciation to all those who provided me the possibility to complete this report.  A special gratitude I give to our project head of the project,Mr. Atul Kumar Pandey Scientist ‘G’ at DRDO, whose contribution in stimulating suggestions and encouragement,  helped me to coordinate my project especially in writing this report.*

*Furthermore, I would also like to acknowledge with much appreciation the crucial role of the Mr. Tara Mishra and Ms. Shikha Sharma, who gave the permission to use all required equipment and the necessary material to complete the task “SOFTWARE DEVELOPMENT FOR PARAMETER EXTRACTION OF GaN HEMT”.  A special thanks goes to my team mate,Rahul Datta and Nishant Kaushik , who  help me to assemble the parts and gave their valuable suggestion.  I have to appreciate the guidance of all people who helped me in this project.*

*Prankur Tiwari*

**ABSTRACT**

A software based on ‘C’ programming language has been developed to determine the parameters for AlGaN/GaN HEMTs and GaAs MESFETs. This will enable the user to obtain the values for HEMT/MESFET electrical equivalent circuit model parameters such as extrinsic (bias independent) as well as intrinsic (bias dependent) for the given measured S-parameters of a particular MESFET or HEMT device. The presented method can successfully extract the values for both extrinsic and intrinsic parameters. Good agreement between measured and simulated data has also been achieved. To determine the parameters’ values using this method, only measured S-parameters at different bias conditions are required. Both methods of extraction i.e., first for extracting 19 element small signal HEMT model and second for extracting 14 element small signal MESFET model have been discussed and presented respectively. This presented method can extract data successfully with excellent correlation between the measured and modeled S-parameters up to 30 GHz.

LIST OF ABBREVIATIONS

* AFM – Atomic Force Microscopy
* AlGaAs – Aluminium gallium arsenide
* AlGaN – Aluminium gallium nitride
* Equiv. – Equivalent
* Freq. – Frequency
* GaAs – Gallium arsenide
* GaN – Gallium nitride
* HEMT – High Electron Mobility Transistor
* HRXRD – High Resolution X-Ray Diffraction of Epitaxial
* MESFET – Metal Semiconductor Field Effect Transistor
* MMIC – Microwave Monolithic Integrated Circuit
* MOVPE – Metal Organic Vapor Phase Epitaxy
* PVT – Physical Vapor Transport
* SiC – Silicon carbide
* SIMS – Secondary Ion Mass Spectrometry
* SPM – Scanning Probe Microscopy
* STM – Scanning Tunneling Microscope
* XPS – X-ray Photoelectron Spectroscopy

TABLE OF CONTENTS

ABSTRACT

LIST OF ABBREVIATION

1. *CHAPTER – 1 ORGANIZATION PROFILE*
   1. Historical Background
   2. Areas of Works
   3. Facilities Available
   4. Products
2. *CHAPTER – 2 INTRODUCTION*
   1. What is HEMT?
   2. Conceptual Explanation
   3. Conduction in HEMT
   4. Electrostatic Mechanism and Polarization Effects
   5. Software Requirements: Code Blocks
3. *CHAPTER – 3 PROJECT WORK PART I*
   1. Source Code
   2. Required Parameter as input
   3. Module Description

3.3.1 Reading of Text File using File handling

3.3.2 Used Data Structure : Struct num

3.3.3 num add( ),multi( ),sub( ),divi( ),mod( ),conju( )

3.3.4 num y2z(y parameter),z2y(z parameter),s2y(s parameter)

3.3.5 void extrinsiccap ( y parameter )

3.3.6 void modifiedy(y parameter)

3.3.7 void extrinsic (z parameter,frequency)

3.3.8 void modified(z parameter)

3.3.9 void extrinsicresist(z parameter)

3.3.10 void intrinsic (y parameter)

3.3.11 void slope (y parameter)

3.4 Output

CHAPTER 1

## ORGANIZATION PROFILE

**1.1 Historical Background**

Solid State Physics Laboratory (SSPL) is a premier laboratory under the Defence Research and Development Organization (DRDO), Ministry of Defence. It was established in 1962. The main purpose was developing an R&D based works in the area of solid-state materials, devices and sub-systems. The institute has an aim to be the center of excellence in research and development of solid-state materials, microelectronics devices and nanotechnology for defence applications.

**1.2 Areas of Work**

Their activities include development of semi-conductor materials, solid state devices, electronic components/sub-systems and investigation of solid-state materials/devices for futuristic defence applications. Improvement of semi-conducting materials, solid state semi-conducting devices, electrical chips, wafers and development of solid-state materials and devices are also done in this organization. Over the years, this lab has been working in the below mentioned fields: -

* RF Devices/MMICs
  1. III-As & III-N Devices
  2. MMIC Design & Fabrication
* Opto-electronic Devices
  1. High Power Laser diodes
  2. Infrared Detectors
* Advanced Materials
  1. III-As/N/Sb & II-VI MBE
  2. SiC PVT
  3. Bulk and LPE of II-VI semiconductors
  4. III-As /N MOVPE
* Sensors
  1. Micro Electro-Mechanical systems (MEMS)
  2. Surface Acoustic Wave (SAW) based sensors
  3. Acoustic sensors
  4. Carbon Nanotube (CNT) based sensors
* Cryogenic Technologies
  1. Stirling Cooler
  2. Joule-Thomson (J-T) Cooler and so on.

**1.3 Facilities Available**

* Crystal Growth Facility (e.g.: Czochralski Ge Puller, Bridgeman Growth etc.)
* Epitaxial Growth Facility (e.g.: III-N & III-As MOVPE, LPE Growth etc.)
* Central Characterization Facility (HRXRD, XPS, SIMS, SPM with AFM, STM etc.)
* Nano-Fabrication Facility (LPCVD Growth of CNT & Graphene, FIB, ALD etc.)
* Semiconductor Processing Facility (Class 100 clean room area, Mask Aligner etc.)
* Electron Beam Writing Tool Facility
* Laser Pattern Generator
* MOCVD System
* RF Probing station
* Ion-Implantation System
* Molecular Beam Epitaxy System
* FPA Testing Facility
* Low Temperature (77K) Four Probe Electrical Testing Facility
* Manual Load Pull Measurement Setup Facility

**1.4 Products**

RF Devices/MMICs

1. AlGaN/GaN HEMT Devices
2. 15 W GaN HEMT Devices
3. pHEMT based 40 GHz MMIC Technology
4. Power MESFET based 18 GHz MMIC Technology

Opto-electronic Devices

1. IRFPA
2. 1-15 W Pulsed Laser Diodes
3. 100 W QCW Laser Diode Array
4. Vertical Laser Diode Stack

Sensors

1. SAW e-NASIKA CWA Detector
2. Automatic Chemical Agent Detection Alarm (ACADA)
3. Acoustic Sensor

CHAPTER 2

## INTRODUCTION

**2.1 What is HEMT?**

A High Electron Mobility Transistor (HEMT), also known as Heterostructure FET (HFET) is a field-effect transistor (FET) incorporating a junction between two materials with different band gaps (i.e. a heterojunction) as the channel instead of a doped region (as is generally case for MOSFET). A commonly used material combination is GaAs with AlGaAs, though there is wide variationdepending on the application of the device. Heterojunction Bipolar Transistor (HBT) and HEMT are widely used in high frequency applications (such as cell phones, satellite television receivers, voltage converters)and high-power applications as they offer higher cutoff frequencies, high breakdown voltage, high power density, high operating temperature and large current driving capabilities.

**2.2 Conceptual Explanation**

HEMTs are heterojunction devices. This means that the semiconductors used have dissimilar band gaps. For instance, silicon has a band gap of 1.1 eV while germanium has a band gap of 0.74 eV. When a heterojunction is formed, the conduction band and valence band throughout the material must bend in order to form a continuous level.

The HEMTs' exceptional carrier mobility and switching speed come from the following conditions: The wide band element is doped with donor atoms.Thus it has excess electrons in its conduction band. These electrons will diffuse to the adjacent narrow band material’s conduction band due to the availability of states with lower energy. The movement of electrons will cause a change in potential and thus an electric field between the materials. The electric field will push electrons back to the wide band element’s conduction band. The diffusion process continues until electron diffusion and electron drift balance each other, creating a junction at equilibrium similar to a p-n junction. So that the undoped narrow band gap materialhas excess majority charge carriers. The fact that themajority charge carriers yields high switching speeds, and the fact that the low band gap semiconductor is undoped means that there are no donor atoms to cause scattering and thus yields high mobility.

An important aspect of HEMTs is that the band discontinuities across the conduction and valence bands can be modified separately. This allows the type of carriers in and out of the device to be controlled. As HEMTs require electrons to be the main carriers, a graded doping can be applied in one of the materials making the conduction band discontinuity smaller, and keeping the valence band discontinuity the same. This diffusion of carriers leads to the accumulation of electrons along the boundary of the two regions inside the narrow band gap material. The accumulation of electrons leads to a very high current in these devices. The accumulated electrons are also known as 2DEG or two-dimensional electron gas.

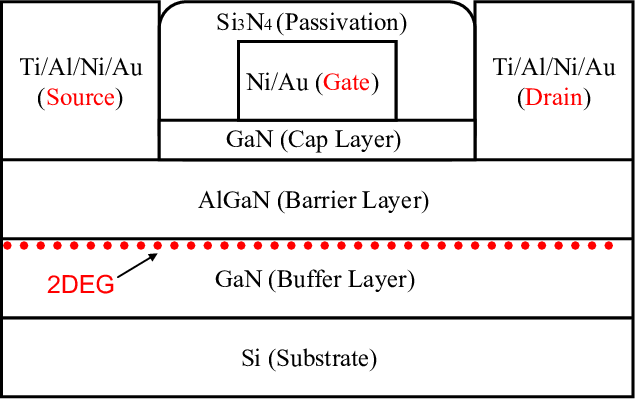


Fig-1: Cross-section view of AlGaN/GaN HEMT

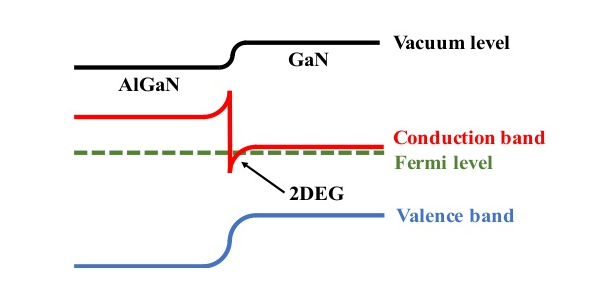


Fig-2: Band Diagram of AlGaN/GaN HEMT

The figure (fig-2) shows a wide bandgap semiconductor material (AlGaN) lying on a narrow band material (undoped GaN). The band diagram shown in above figure demonstrates the sharp dip in the conduction band edge at the AlGaN/GaN interface. A high carrier results at the dip within a narrow region (quantum well) in the source – draindirection. The distribution of electric states in the quantum well continues in the two dimensions parallel to the interface and discrete perpendicular to the interface due to the very small thickness of the quantum well in comparison to the width and length of the channel. The charge density is defined as a two dimensional electron gas (2DEG) amd quantified in terms of sheet carrier density, n­s­. An observation about the 2DEG for the AlGaN/GaN interface is how it can form even when there is no intentional doping of the AlGaN layer.

The formation of 2DEG at the heterointerface is different from other HEMTs like AlGaAs/GaAs. Due to the presence of a strong polarization field across the AlGaN/GaN heterojunction, a 2DEG with the sheet carrier density up to 1013 cm-2 can be achieved without intentional doping.

A two-dimensional electron gas (2DEG) is a scientific-model in solid-state physics. It is an electron gas that is free to move in two dimensions, but tightly confined in third. This tight confinement leads to quantized energy levels for motion in the third direction. Thus the electrons appear to be a 2D sheet embedded in a #D world. Most 2DEG are found in transistor-like structures made from semiconductors. When the transistor is in inversion mode, the electrons underneath the gate oxide are confined to the semiconductor-oxide interface, and thus occupy well defined energy levels. For thin-enough potential wells and temperatures not too high, only the lowest level is occupied and so the motion of electrons perpendicular to the interface can be ignored. However, the electron is free to move parallel to the interface, and so is quasi-two-dimensional state.

**2.3 Conduction in HEMT**

Semiconductors are doped with impurities which donate either mobile electrons or holes. However, these electron slowed down through collisions with the impurities (dopants) used to generate them in first place. HEMTs avoid this through the use of high mobility electrons generated using the heterojunction of a highly doped wide-bandgap n-type donor-supply layer (such as AlGaAs, AlGaN) and a non-doped narrow-bandgap channel layer with no dopant impurities (such as GaAs, GaN).

The electron generated in the thin n-type AlGaAs layer drop completely into the GaAs layer to form a depleted AlGaAs layer, because the heterojunction created by different band-gap materials forms a quantum well (a steep canyon) in the conduction band on the GaAs side where the electrons can move quickly without colliding with any impurities because the GaAs layer is undoped and from which the electrons cannot escape.

**2.4 Electrostatic Mechanism and Polarization Effects**

Since GaN has high electron affinity, free electrons in the AlGaAs layers are transferred to the undoped GaN layer where they form a two dimensional high mobility electron gas within 10 nm of the interface. The N-type AlGaN layer of the HEMT is depleted completely through two dimensional mechanisms:

* Trapping of free electrons by surface causes the surface depletion.
* Transfer of electrons into the undoped GaN layer brings about the interface depletion.

The fermi level of the gate metal is matched to the pinning point, about 1.2 eV below the conduction band. With the reduced AlGaN layer thickness, the electrons supplied by donors in the AlGaN layer are insufficient to pin the layer. As a result, band bending is moving upward and the two-dimensional electron gas does not appear. When a positive voltage greater than the threshold voltage is applied to the gate, electrons accumulate at the interface and form a two-dimensional electron gas.

AlGaN/GaN HEMT polarization effects are both spontaneous and piezoelectric. The spontaneous polarization refers to the built-in polarization field present in an unstrained crystal. The electric field exists because the crystal lacks inversion symmetry and bond between the two atoms is not purely covalent. This results in a displacement of the electron charge cloud towards one atom in the bond. In the direction along which the crystal lacks inversion asymmetry, the asymmetric electron results in a net positive charge located at one face of the crystal and a net negative charge at the other face. The piezoelectric polarization is due to the presence of a polarization field resulting from the distortion of the crystal lattice. Due to the large difference in lattice constant between AlGaN and GaN materials, the AlGaN layer which is grown on the GaN buffer layer is strained. Due to the large value of the piezoelectric coefficients of these materials, this strain results in a sheet charge at the two faces of AlGaN layer. The total polarization field entirely depends on the orientation of the GaN crystal.

**2.5 Software Requirement**

**2.5.1 Product Perspective**

The purpose of this project work is to develop a software, which will be able to determine the value of certain parameters based on the given inputs. So, the aim of this work is to provide a proper clustering methodology and an efficient algorithm for the purpose of the same. Here the main focus is given to the development of an algorithm which will prove to be more and more fruitful for this purpose. The users who are trying to make an evaluation need to be proficient in the following areas:

**2.5.2 C Programming Language**

C is a general-purpose, imperative computer programming language supporting structured programming, variable scope and recursion. C was originally developed at Bell Labs by Dennis Ritchie between 1972 and 1973. It is a robust language with rich set of built-in functions and operators that can be used to write any complex program. The C compiler combines the capabilities of an assembly language with features of a high-level language. Programs written in C are efficient and fast. This is due to its variety of data type and powerful operators. C is highly portable this means that programs once written can be run on other machines with little or no modification. A C program is basically a collection of functions that are supported by C library. We can create our own function easily and can add it to C library. C combines the power and capability of assembly language with the user-friendly features of high-level language. C is renowned for its simplicity and is easy to be use because of its structured approach. It has a vast collection of keywords, operators, built-in functions and data-types which make it efficient and powerful.

**2.5.3 Code Block and MinGW**

**Code::Blocks** is a free, open-source cross-platform IDE that supports multiple compilers including GCC, Clang and Visual C++. It is developed in C++ using [wxWidgets](https://en.wikipedia.org/wiki/WxWidgets) as the GUI toolkit. Using a plugin architecture, its capabilities and features are defined by the provided plugins. Currently, Code::Blocks is oriented towards [C](https://en.wikipedia.org/wiki/C_(programming_language)), [C++](https://en.wikipedia.org/wiki/C%2B%2B), and [FORTRAN](https://en.wikipedia.org/wiki/Fortran). It has a custom [build system](https://en.wikipedia.org/wiki/Build_automation) and optional [Make](https://en.wikipedia.org/wiki/Make_(software)) support.

Code::Blocks is being developed for [Windows](https://en.wikipedia.org/wiki/Microsoft_Windows) and [Linux](https://en.wikipedia.org/wiki/Linux)  and has been ported to [FreeBSD](https://en.wikipedia.org/wiki/FreeBSD)[[2]](https://en.wikipedia.org/wiki/Code::Blocks#cite_note-FreeBSD-2), [OpenBSD](https://en.wikipedia.org/wiki/OpenBSD)[[3]](https://en.wikipedia.org/wiki/Code::Blocks#cite_note-3) and [Solaris](https://en.wikipedia.org/wiki/Solaris_(operating_system))[[4]](https://en.wikipedia.org/wiki/Code::Blocks#cite_note-4).

MinGW, formerly mingw32, is a free and open source software development environment to create Microsoft Windows applications. The development of the MinGW project has slowed down since the creation in 2013 of an alternative project called MinGW-w64 by a different author.

CHAPTER 3

## PROJECT WORK PART I

* 1. **Source Code**

#include <stdio.h>

#include <conio.h>

float freq[30];

float cb[30],rg[30],rs[30],rd[30],w[30],rgd[30]; //Extrinsic Parameter

float lg[30],ls[30],ld[30];

float cgd[30],cgs[30],ri[30],gm[30],tau[30],cds[30],cdc[30],rds[30]; //Intrinsic Parameter

float vgs,vth,rch;

float cs[30],cg[30],cd[30];

typedef struct num

{

float x,y;

}num;

num s[30][4]; //Global Declaration

num y[30][4]; //Global Declaration

num z[30][4]; //Global Declaration

num Zo; //Characteristics Impedance

num Yo; //Characteristics Admittance

num add(num ca, num cb)

{

num cc;

cc.x=ca.x+cb.x;

cc.y=ca.y+cb.y;

return cc;

}

num mult(num ca, num cb)

{

num cc;

cc.x=ca.x\*cb.x-ca.y\*cb.y;

cc.y=ca.x\*cb.y+ca.y\*cb.x;

return cc;

}

num sub(num ca,num cb)

{

num cc;

cc.x=ca.x-cb.x;

cc.y=ca.y-cb.y;

return cc;

}

num conj(num ca)

{

num cc;

cc.x=ca.x;

cc.y=-ca.y;

return cc;

}

num divi(num ca,num cb)

{

num cc,cc1,cc2,pro;

cc1=conj(cb);

cc2=mult(cc1,ca);

pro=mult(cb,cc1);

cc.x=cc2.x/pro.x;

cc.y=cc2.y/pro.x;

return cc;

}

float mod (num number)

{

float temp;

temp=(number.x\*number.x)+(number.y\*number.y);

temp=sqrt(temp);

return temp;

}

/\*\*\*\*\*\*\*\*\* y to z \*\*\*\*\*\*\*\*\*/

num y2z(num y[30][4])

{

num t0,t1,t2,t3,c1;

num delta;

int i;

c1.x=(1.000000);

c1.y=0.000000;

c1.x=(-1)\*c1.x;

for(i=0;i<30;i++)

{

t0=y[i][0];

t1=y[i][1];

t2=y[i][2];

t3=y[i][3];

delta=sub(mult(t0,t3),mult(t1,t2));

z[i][0]=divi(t3,delta);

z[i][1]=mult(c1,divi(t1,delta));

z[i][2]=mult(c1,divi(t2,delta));

z[i][3]=divi(t0,delta);

}

}

/\*\*\*\*\*\*\*\*\* z to y \*\*\*\*\*\*\*\*\*/

num z2y(num z[30][4])

{

num t0,t1,t2,t3,c1;

num delta;

int i;

c1.x=(1.000000);

c1.y=0.000000;

c1.x=(-1)\*c1.x;

for(i=0;i<30;i++)

{

t0=z[i][0];

t1=z[i][1];

t2=z[i][2];

t3=z[i][3];

delta=sub(mult(t0,t3),mult(t1,t2));

y[i][0]=divi(t3,delta);

y[i][1]=mult(c1,divi(t1,delta));

y[i][2]=mult(c1,divi(t2,delta));

y[i][3]=divi(t0,delta);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* s to y \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

num \*s2y(num s[30][4])

{

num c1,delta,two;

num d1,d2,d3,ds1,ds2,ay11,ay12,ay21,ay22,s21,h21,h22;

num t1,t2,t3,t0;

int i;

//num y[30][4];

/\*for(i=0;i<30;i++)

{

printf("\n");

for(j=0;j<4;j++)

printf("%f\t",s[i][j].x);

}\*/

c1.x=1.000000;

c1.y=0.000000;

printf("\nEnter Value for Characteristics Impedance : ");

scanf("%f",&Zo.x);

Yo.x=(1/Zo.x);

Zo.y=0.000000;

Yo.y=0.000000;

for(i=0;i<30;i++)

{

t0=s[i][0];

t1=s[i][1];

t2=s[i][2];

t3=s[i][3];

d1=add(c1,t0);

d2=add(c1,t3);

d3=mult(t1,t2);

delta=sub(mult(d1,d2),d3);

ds1=sub(c1,t0);

ds2=sub(c1,t3);

ay11=add(mult(d2,ds1),d3);

two=add(c1,c1);

two.x=((-1)\*two.x);

ay12=mult(two,t1);

ay21=mult(two,t2);

ay22=add(mult(d1,ds2),d3);

y[i][0]=mult(divi(ay11,delta),Yo);

y[i][1]=mult(divi(ay12,delta),Yo);

y[i][2]=mult(divi(ay21,delta),Yo);

y[i][3]=mult(divi(ay22,delta),Yo);

}

return y;

}

void extrinsiccap(num y[30][4],float freq1[30])

{

int i,j;

float temp1,temp2;

num t1,t2,t3;

float cpd = 0.012838,cgsi=0.193435,cgdi=0.1152382978,cdsi=0.009828,cpg=0.0084347;

for(i=0;i<30;i++)

w[i]=6.2832\*freq1[i];

for(i=1;i<30;i++)

{

cgd[i]=(((-1)\*y[i][1].y)/w[i]);

cds[i]=(y[i][3].y/w[i]);

//cds[i]=cds[i];

cgs[i]=(y[i][0].y/w[i]);

printf("\n\n Cgd = %f \t\t Cds = %f \t\t Cgs = %f\n",cgd[i],cds[i],cgs[i]);

}

}

void modifiedy(num y[30][4],float freq1[30])

{

int i,j;

float cpd = 0.012838,cgsi=0.193435,cgdi=0.1152382978,cdsi=0.009828,cpg=0.0084347;

for(i=0;i<30;i++)

w[i]=6.2832\*freq1[i];

for(i=0;i<30;i++)

{

y[i][0].y=y[i][0].y-(w[i]\*0.296\*0.001);

// y[i][1].y=y[i][1].y+(w[i]\*cgdi);

// y[i][2].y=y[i][2].y+(w[i]\*cgdi);

y[i][3].y=y[i][3].y-(w[i]\*0.01712\*0.001);

}

}

void modifiedy1(num y[30][4],float freq1[30])

{

int i,j;

float cpd = 0.012838,cgsi=0.193435,cgdi=0.1152382978,cdsi=0.009828,cpg=0.0084347;

for(i=0;i<30;i++)

w[i]=6.2832\*freq1[i];

for(i=0;i<30;i++)

{

y[i][0].y=y[i][0].y-(w[i]\*0.0674\*0.001);

y[i][1].y=y[i][1].y+(w[i]\*0.057\*0.001);

y[i][2].y=y[i][2].y+(w[i]\*0.057\*0.001);

y[i][3].y=y[i][3].y-(w[i]\*0.1981\*0.001);

}

}

void extrinsic(num z[30][4],float freq1[30])

{

int i,j;

float temp1,temp2;

float slopee[30];

for(i=0;i<30;i++)

w[i]=6.2832\*freq1[i];

for(i=1;i<30;i++)

{

temp1=((z[i][1].y)/w[i])-(z[i-1][1].y/w[i-1]);

temp2=(1/(w[i]\*w[i]))-(1/(w[i-1]\*w[i-1]));

slopee[i]=temp2/temp1;

cs[i]=(-1)\*slopee[i];

temp1=((z[i][0].y)/w[i])-(z[i-1][0].y/w[i-1]);

temp2=(1/(w[i]\*w[i]))-(1/(w[i-1]\*w[i-1]));

cg[i]=(-1)\*(temp1/temp2);

cg[i]=cg[i]-(1/cs[i]);

cg[i]=(1/cg[i]);

temp1=((z[i][3].y)/w[i])-(z[i-1][3].y/w[i-1]);

temp2=(1/(w[i]\*w[i]))-(1/(w[i-1]\*w[i-1]));

cd[i]=(-1)\*(temp1/temp2);

cd[i]=cd[i]-(1/cs[i]);

cd[i]=(-1)\*(1/cd[i]);

temp1=((z[i][1].y)\*w[i])-(z[i-1][1].y\*w[i-1]);

temp2=((w[i]\*w[i]))-((w[i-1]\*w[i-1]));

ls[i]=temp1/temp2;

temp1=((z[i][0].y)\*w[i])-(z[i-1][0].y\*w[i-1]);

temp2=((w[i]\*w[i]))-((w[i-1]\*w[i-1]));

lg[i]=(temp1/temp2)-(ls[i]);

temp1=((z[i][3].y)\*w[i])-(z[i-1][3].y\*w[i-1]);

temp2=((w[i]\*w[i]))-((w[i-1]\*w[i-1]));

ld[i]=(temp1/temp2)-(ls[i]);

printf("\n\n Cs[%d] = %f\tCg[%d] = %f \t Cd[%d] = %f\tLs[%d] = %.10lf\t Lg[%d] = %.8lf \t Ld[%d] = %.8lf\n",i,cs[i],i,cg[i],i,cd[i],i,ls[i],i,lg[i],i,ld[i]);

//printf("Cs[%d] = %f\tCg[%d] = %f\tCd[%d] = %f\tLs[%d] = %0.7lf\n",i,cs[i],i,cg[i],i,cd[i],i,ls[i]);

}

}

void modifiedz(num z[30][4])

{

int i,j;

for(i=0;i<30;i++)

{

//z[i][0].x=z[i][0].x-rs[i]-rg[i];

z[i][0].y=z[i][0].y-(w[i]\*(lg[i]+ls[i]));

//z[i][1].x=z[i][1].x-rs[i];

z[i][1].y=z[i][1].y-(w[i]\*ls[i]);

//z[i][2].x=z[i][2].x-rs[i];

z[i][2].y=z[i][2].y-(w[i]\*ls[i]);

//z[i][3].x=z[i][3].x-rs[i]-rd[i];

z[i][3].y=z[i][3].y-(w[i]\*(ls[i]+lg[i]));

}

// printf("\n\nZ Matrix for Intrinsic Parameter :\n");

for(i=0;i<30;i++)

{

printf("\n");

for(j=0;j<4;j++)

printf("%f + i %f\t\t",z[i][j].x,z[i][j].y);

}

}

void modifiedz1(num z[30][4])

{

int i,j;

for(i=0;i<30;i++)

{

z[i][0].x=z[i][0].x-(rs[i]+rg[i]);

//z[i][0].y=z[i][0].y-(w[i]\*(0.00811));

z[i][1].x=z[i][1].x-rs[i];

//z[i][1].y=z[i][1].y-(w[i]\*0.00811);

z[i][2].x=z[i][2].x-rs[i];

//z[i][2].y=z[i][2].y-(w[i]\*0.00811);

z[i][3].x=z[i][3].x-(rd[i]+rs[i]);

//z[i][3].y=z[i][3].y-(w[i]\*(0.00811));

}

printf("\n\nZ Matrix for Intrinsic Parameter :\n");

for(i=0;i<30;i++)

{

printf("\n");

for(j=0;j<4;j++)

printf("%f + i %f\t\t",z[i][j].x,z[i][j].y);

}

}

void extrinsicresist(num z[30][4],float freq1[30])

{

int i,j;

float temp1,temp2;

float slopee[30];

for(i=0;i<30;i++)

w[i]=6.2832\*freq1[i];

for(i=10;i<30;i++)

{

rs[i]=z[i][1].x;

rg[i]=6.9;

rd[i]=2.2159418+rs[i];

printf("\nRs[%d] = %f\tRg[%d] = %f\tRd[%d] = %f",i,rs[i],i,rg[i],i,rd[i]);

}

}

void intrinsic(num y[17][4])

{

int i;

float cgsarg,denri,gmt,gds[17];

printf("\n\nIntrinsic Elements : \n");

for(i=1;i<10;i++)

{

cgd[i]=(-1)\*(y[i][1].y/w[i]);

cgsarg=pow((y[i][0].y-(w[i]\*cgd[i])),2);

cgs[i]=(y[i][0].y-(w[i]\*cgd[i]))\*((1+((y[i][0].x\*y[i][0].x)/cgsarg))/w[i]);

denri=(y[i][0].y-(w[i]\*cgd[i]))\*(y[i][0].y-w[i]\*cgd[i])+y[i][0].x\*y[i][0].x;

ri[i]=(y[i][0].x/denri);

gmt=y[i][2].y+(w[i]\*cgd[i]);

gm[i]=sqrt((y[i][2].x\*y[i][2].x+gmt\*gmt)\*(1.0+w[i]\*w[i]\*cgs[i]\*cgs[i]\*ri[i]\*ri[i]));

tau[i]=asin((-w[i]\*cgd[i]-y[i][2].y-w[i]\*cgs[i]\*ri[i]\*y[i][2].x)/gm[i])/w[i];

cds[i]=(y[i][3].y-(w[i]\*cgd[i]))/w[i];

cdc[i]=cgs[i]\*y[i][1].x/y[i][0].x;

cds[i]=cds[i]-cdc[i];

rds[i]=(1/y[i][3].x);

//rds[i]=(1/gds[i]);

printf("\n\n Cgd(pF) = %f\t\tCgs(pF) = %f\t\tRi(ohm) = %f\t\tgm(S) = %f\t\tTau = %f\t\tCds(pF) = %f\t\tRds(ohm) = %f",cgd[i]\*1000,cgs[i]\*1000,ri[i],gm[i],tau[i],cds[i]\*1000,rds[i]);

}

}

void slope(num y[30][4],float freq[30])

{

int i,j;

float w[30];

float temp1,temp2;

FILE \*fp1,\*fp2,\*fp3,\*fp4;

fp1=fopen("Slope1.txt","w+");

fp2=fopen("Slope2.txt","w+");

fp3=fopen("Slope3.txt","w+");

fp4=fopen("Slope4.txt","w+");

for(i=0;i<30;i++)

w[i]=(6.2832\*freq[i]);

for(i=0;i<30;i++)

{

temp1=(y[i][0].x);

temp2=w[i];

fprintf(fp1,"%f\t%f\n",y[i][0].y,w[i]);

fprintf(fp2,"%f\t%f\n",y[i][3].y,w[i]);

fprintf(fp3,"%f\t%f\n",y[i][1].y,w[i]);

fprintf(fp4,"%f\t%f\n",temp1,temp2);

}

}

int main()

{

int i,j,k;

float arr[30][9];

FILE \*fp;

/\*\*\*\*\*\*\*\*\* Section 1 \*\*\*\*\*\*\*\*\*/

fp=fopen("coldfet2.txt","r");

for(i=0;i<30;i++)

{

for(j=0;j<9;j++)

fscanf(fp,"%f",&arr[i][j]);

}

printf("COLDFET TEXT READ DATA\n");

for(i=0;i<30;i++)

{

printf("\n");

for(j=0;j<9;j++)

printf("%f\t",arr[i][j]);

}

for(i=0;i<30;i++)

{

freq[i]=arr[i][0];

}

for(i=0;i<30;i++)

{

s[i][0].x=arr[i][1];s[i][0].y=arr[i][2];

s[i][1].x=arr[i][3];s[i][1].y=arr[i][4];

s[i][2].x=arr[i][5];s[i][2].y=arr[i][6];

s[i][3].x=arr[i][7];s[i][3].y=arr[i][8];

}

s2y(s);

printf("\n\nY-matrix\n");

for(i=0;i<30;i++)

{

printf("\n");

for(j=0;j<4;j++)

printf("%f + i %f\t\t",y[i][j].x,y[i][j].y);

}

slope(y,freq);

extrinsiccap(y,freq);

fclose(fp);

//modifiedy(y,freq);

/\*\*\*\*\*\*\*\*\* Section 2 \*\*\*\*\*\*\*\*\*/

fp=fopen("unbiased.txt","r");

for(i=0;i<30;i++)

{

for(j=0;j<9;j++)

fscanf(fp,"%f",&arr[i][j]);

}

printf("\nUNBAISED TEXT READ DATA\n");

for(i=0;i<30;i++)

{

printf("\n");

for(j=0;j<9;j++)

printf("%f\t",arr[i][j]);

}

for(i=0;i<30;i++)

{

freq[i]=arr[i][0];

}

for(i=0;i<30;i++)

{

s[i][0].x=arr[i][1];s[i][0].y=arr[i][2];

s[i][1].x=arr[i][3];s[i][1].y=arr[i][4];

s[i][2].x=arr[i][5];s[i][2].y=arr[i][6];

s[i][3].x=arr[i][7];s[i][3].y=arr[i][8];

}

s2y(s);

y2z(y);

extrinsic(z,freq);

extrinsicresist(z,freq);

fclose(fp);

/\*\*\*\*\*\*\*\*\* Section 3 \*\*\*\*\*\*\*\*\*/

fp=fopen("hotfet2.txt","r");

for(i=0;i<30;i++)

{

for(j=0;j<9;j++)

fscanf(fp,"%f",&arr[i][j]);

}

printf("\n\nHOTFET TEXT READ DATA\n");

for(i=0;i<30;i++)

{

printf("\n");

for(j=0;j<9;j++)

printf("%f\t",arr[i][j]);

}

for(i=0;i<30;i++)

{

freq[i]=arr[i][0];

}

for(i=0;i<30;i++)

{

s[i][0].x=arr[i][1];s[i][0].y=arr[i][2];

s[i][1].x=arr[i][3];s[i][1].y=arr[i][4];

s[i][2].x=arr[i][5];s[i][2].y=arr[i][6];

s[i][3].x=arr[i][7];s[i][3].y=arr[i][8];

}

s2y(s);

printf("\nS Matrix (HOTFET)\n\n");

for(i=0;i<4;i++)

{

printf("%f+i%f\n",s[4][i].x,s[4][i].y);

}

printf("\nY Matrix (HOTFET)\n\n");

for(i=0;i<4;i++)

{

printf("%f+i%f\n",y[4][i].x,y[4][i].y);

}

modifiedy(y,freq);

printf("\nY Matrix (HOTFET)\n\n");

for(i=0;i<4;i++)

{

printf("%f+i%f\n",y[4][i].x,y[4][i].y);

}

y2z(y);

printf("\nZ Matrix (HOTFET)\n\n");

for(i=0;i<4;i++)

{

printf("%f+i%f\n",z[4][i].x,z[4][i].y);

}

modifiedz(z);

z2y(z);

modifiedy1(y,freq);

y2z(y);

modifiedz1(z);

z2y(z);

intrinsic(y);

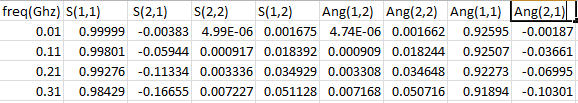
return 0;

}

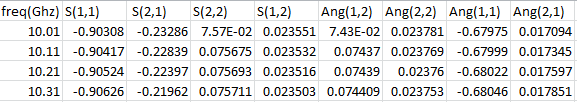
* 1. **Required file as Input**

We have to have three text file named coldfe2.txt, unbiased.txt and hotfet.txt as our input data, which contain frequency, and S parameter magnitude and angle in certain order the snippet of all files are:

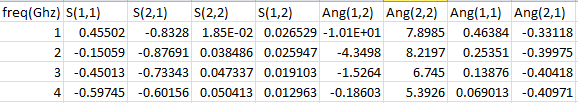
* Coldfet2.txt



* Unbaised.txt



* Hotfet.txt



* 1. **Model description**
* **Reading of text file using file handling**

In this module we have read all the input file accordingly and store values in different variables in order to use it in later module.

This file handling part is done with the help of an inbuilt function *fscanf( ).*

* **Data Structure:Structure num**

A sturucure is created using struct keyword and named num, in that structure there is two floating type variable x and y where x contain the real valur and yu conatain the imaginary part.

struct num

{

float x,y;

}

* **Add( ),Multi( ),Sub( ),Divi( ),Conj( ),Mod( )**

This module will add,subtract,divide,conjugate the complex number given to them as a parameter .

* **Y2Z( ), Z2Y( ), S2Y( )**

In this module the parameter conversion occur as the name indicate itself Y2Z convert Y parameter to Z with y parameter as formal arguments, similarly Z22 convert Z parameter to Y with z parameter as formal argument and S2Y convert S parameter to Y with s parameter as formal argument.

* **Extrinsiccap( )**

In this module, the extrinsic capacitance is calculated with the help of certain formula. This module takes Y parameter and frequency as formal argument.

* **Modifiedy( )**

In this module the Y parameter, which has been calculated earlier, is modified accordingly.

* **Extrinsic** ( )

Under this function all other parameter such as Ld,Lg,Ls,Cs,Cd,Cg are calculated and printed in the order. This module takes Z parameter and frequency as formal argument.

* **Modifiedz( )**

As the name indicate this module, modify the Z parameter which has been calculated earlier and that modified Z parameter is used in further module.

* **Extrinsicresist**( )

Under this module, the extrinsic resistance is calculated with the help of Z parameter and frequency as formal argument.

* **Intrinsic( )**

In this module, the intrinsic elements of the device is calculated. Intrinsic elements like Cgd(nF),Cgs(nF) ,Ri(ohm),Gm(S),Tau ,Cds(nF),Rds(ohm).

* **Main( )**

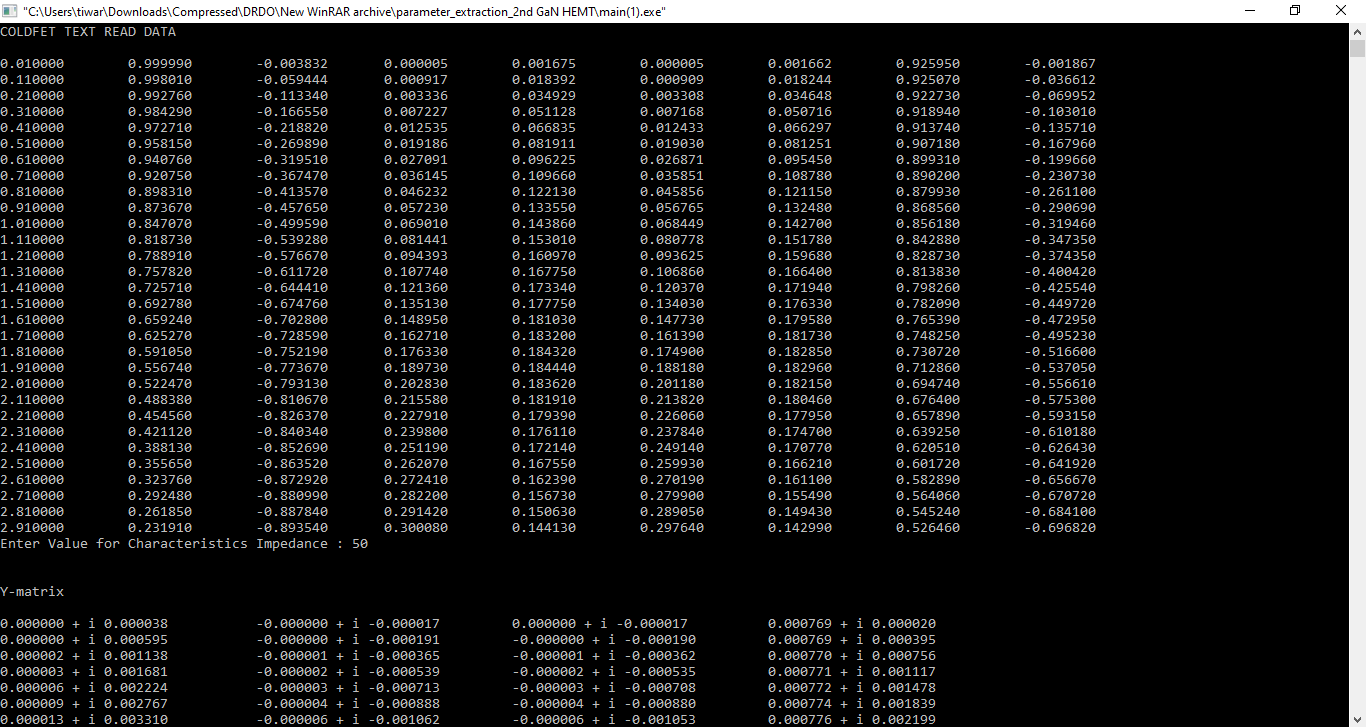
This is the most important module of the program as it is the driver’s code of the program, which determines the order of calling of all the other modules.

**Conclusion**

Modeling of AlGaN/GaN HEMTs is a research area gaining momentum for application in understanding device behavior. The modeling of the device depends on the accuracy of the parameter extraction process. The parameter extraction process strongly depends on the biasing condition of device, device’s physical condition, doping concentration, doping material, channel width, channel length and so on which can only be measured physically. So, the accuracy of the result obtained from this software entirely depends on the measured S-parameter. Errors in S-parameter values will result in defected outputs.

The parasitic resistance at gate terminal depends on the gate current, which can only measure experimentally. Due to some difficulties, this type of experiment can’t be carried out. Therefore, gate resistance (Rg) cannot measured using this software. The 19-elements small-signal model can reproduce DC, small-signal RF, S-parameter and large-signal RF for AlGaN/GaN accurately.

**OUTPUT**



**References**

1. R. G. Brady, C. H. Oxley and T. J. Brazil, “An improved small-signal parameter-extraction algorithm for GaN HEMT devices,” IEEE Transactions on Microwave Theory and Techniques, vol. 56, no. 7, July 2008.
2. J. Michael Golio, Microwave MESFETs & HEMTs, Artech House, Massachusetts,1991
3. E. Chigeva, W. Walthes, D. Wiegner, M. Grӧzing, F. Schaich, N. Wieser and M. Berroth, “Determination of Small-signal Parameters of GaN-based HEMTs.”
4. L. Sang, Y. Xu, Y. Chen, Y. Guo and R. Xu, “Large Signal Equivalent Model For Package AlGaN/GaN HEMT.”
5. Gilles Dambrine, Alain Cappy, Frederic Heliodore and Edouard Playez, “A New Method for Determining the FET Small-signal Equivalent Circuit.”
6. Chen G, Kumar V, Schwindt R S, Adesida I, “A low gate bias model extraction technique for AlGaN/GaN HEMTs.”
7. Du J f, Xu P, Wang K, “Small signal modeling of AlGaN/GaN HEMTs with consideration of CPW capacitances,” Journal of semiconductors, 2015.
8. S. Yanagawa, H. Ishihara and M. Ohtomo, “Analytical Method for Determining Equivalent Circuit Parameters of GaAs FET’s,” IEEE Trans. Microwave Theory Tech., vol. 44, no 10, pp.1637-1641, October 1996.
9. A. Jarndal, G. Kompa, “A new small-signal mdeling approach applied to GaN Devices,” IEEE Transactions on Microwave Theory and Techniques, vol. 53, no. 11, pp. 3440-3448, November 2005.
10. F. Lin, G. Kompa, “Efficient FET Model Parameter Extraction Using Multi-Plane Data-Fitting and Bidirectional Search Technique,”IEEE MTT-S Int. Microwave Symp. Dig., pp. 1021-1024, 1993.