Appendix A

Arduino IDE code:

```
#include <LiquidCrystal.h>

LiquidCrystallcd(12, 11, 5, 4, 3, 2);

constintsensorPin = A0;

constint Heater=9;

constint Cooling=8;

int reading;

float voltage;

floattemperatureC;

void setup( )

{ Serial.begin(9600);

lcd.begin(16, 2);

pinMode (Heater, OUTPUT);

pinMode (Cooling, OUTPUT);}

void loop ( )

{ digitalWrite (Heater, HIGH);

digitalWrite (Cooling, HIGH);

reading = analogRead(sensorPin);

voltage = reading * 5.0/1024;
```

```
temperatureC = voltage* 100 ;

if (temperatureC> 100)

{digitalWrite (Heater, LOW);

Serial.println("========= WARNING ========== ");

Serial.println("Temperature is: ");

Serial.print(temperatureC);

Serial.println(" degrees C");

lcd.print(" ===WARNING===");

lcd.setCursor(0, 1);

lcd.print(temperatureC);

lcd.print("' C");

lcd.setCursor(0, 0);

delay(1000);}

else if(temperatureC< -20)

{digitalWrite (Cooling, LOW);

Serial.println("========= WARNING ========== ");

Serial.println("Temperature is: ");

Serial.print(temperatureC);

Serial.println(" degrees C");

lcd.print(" ===WARNING===");

lcd.setCursor(0, 1);
```

```
lcd.print(temperatureC);

lcd.print("' C");

lcd.setCursor(0, 0);

delay(1000);}

else

{Serial.println("Temperature is: ");

Serial.print(temperatureC);

Serial.println(" degrees C");

lcd.print("Temprature is");

lcd.setCursor(0, 1);

lcd.print(temperatureC);

lcd.print("'  C");

lcd.setCursor(0, 0);

delay(2000);}}
```

Appendix B:

ANSI C Code by Atmel Studio for thermal control system test:

```
/*
 * GRADUATE_PROJECT.c
 *
 * Created: 10/19/2016 8:08:14 PM
*  Author: Muhanned
 */
```

```
#defineF_CPU      1000000UL
#include<avr/delay.h>
#include<avr/io.h>
#include<string.h>
#include<avr/interrupt.h>
/*LCD function declarations */
voidLCD_send_command(unsignedcharcmnd);
voidLCD_send_data(unsignedchardata);
voidLCD_init();
voidLCD_goto(unsignedchary,unsignedcharx);
voidLCD_print(char*string);
#defineLCD_DATA_PORT     PORTB
#defineLCD_DATA_DDR      DDRB
#defineLCD_DATA_PIN      PINB
#defineLCD_CNTRL_PORT    PORTC
#defineLCD_CNTRL_DDR     DDRC
#defineLCD_CNTRL_PIN     PINC
#defineLCD_RS_PIN        0
#defineLCD_RW_PIN        1
#defineLCD_ENABLE_PIN    2
#defineSET_HOUR          3
#defineSET_MINUTE        4
intmain(void)
{
        volatileuint8_tadcValue;

        DDRA=0x00;
        DDRD=0xff;
        PORTD=0b00000011;
```

```
ADCSRA|=(1<<ADEN);
ADCSRA|=(1<<ADPS0)|(1<<ADPS1);
ADMUX|=(1<<ADLAR);
ADMUX|=(1<<MUX0);
LCD_init();
LCD_goto(1,2);
LCD_print("Temperature is");
charLCD[8];
	while(1)
{
	ADCSRA|=(1<<ADSC);
	while(ADCSRA&(1<<ADSC));
	adcValue=ADCH;

	floatvoltage=adcValue*5.0/256;
	inttemperatureC=voltage*100;

itoa(temperatureC,LCD,10);
LCD_goto(2,2);
LCD_print(LCD);

LCD_goto(2,5);
LCD_print("' C");
if(temperatureC>=100)
{
	PORTD=0b00000010;
	LCD_goto(1,2);
	LCD_print("====WARNING====");
```

```
        }
        elseif(temperatureC<=-20)
        {PORTD=0b00000001;
        LCD_goto(1,2);
        LCD_print("===WARNING===");}
        }
}

voidLCD_send_command(unsignedcharcmnd)
{
        LCD_DATA_PORT=cmnd;
        LCD_CNTRL_PORT&=~(1<<LCD_RW_PIN);
        LCD_CNTRL_PORT&=~(1<<LCD_RS_PIN);

        LCD_CNTRL_PORT|=(1<<LCD_ENABLE_PIN);
        _delay_us(2);
        LCD_CNTRL_PORT&=~(1<<LCD_ENABLE_PIN);
        _delay_us(100);
}

/* This function sends the data 'data' to the LCD module*/
voidLCD_send_data(unsignedchardata)
{
        LCD_DATA_PORT=data;
        LCD_CNTRL_PORT&=~(1<<LCD_RW_PIN);
        LCD_CNTRL_PORT|=(1<<LCD_RS_PIN);

        LCD_CNTRL_PORT|=(1<<LCD_ENABLE_PIN);
        _delay_us(2);
```

```c
	LCD_CNTRL_PORT&=~(1<<LCD_ENABLE_PIN);
	_delay_us(100);
}
voidLCD_init()
{
	LCD_CNTRL_DDR=0xFF;
	LCD_CNTRL_PORT=0x00;
	LCD_DATA_DDR=0xFF;
	LCD_DATA_PORT=0x00;
	_delay_ms(10);
	LCD_send_command(0x38);
	LCD_send_command(0x0C);
	LCD_send_command(0x01);
	_delay_ms(10);
	LCD_send_command(0x06);
}

/* This function moves the cursor the line y column x on the LCD module*/
voidLCD_goto(unsignedchary,unsignedcharx)
{
unsignedcharfirstAddress[]={0x80,0xC0,0x94,0xD4};

LCD_send_command(firstAddress[y-1]+x-1);
_delay_ms(10);
}
voidLCD_print(char*string)
{
	unsignedchari=0;
```

```
        while(string[i]!=0)
        {
                LCD_send_data(string[i]);
                i++;
        }
}
```

Appendix C:

EMI SHIELDING CHARACTERISTICS OF METALS

METAL	SPECIFIC ELECTRIC CONDUCTIVITY σ_r	SPECIFIC PERMEABILITY μ_r (≤ 10 kHz)	SPECIFIC ABSORPTION LOSS $A = k_1 \sqrt{\sigma_r \mu_r}$	SPECIFIC REFLECTION LOSS $R = k\sqrt{\sigma_r / \mu_r}$	SPECIFIC REFLECTION LOSS R (dB)	DENSITY ρ (g/cm^3)
Silver	1.064	1	1.03	1.3	0.3	10.501
Copper (solid)	1	1	1	1	0	8.96
Copper (flame spray)	0.1	1	0.32	0.32	-10	N/A
Gold	0.7	1	0.88	0.88	-1.1	19.282
Chromium	0.664	1	0.81	0.81	-1.8	7.19
Aluminum (soft)	0.63	1	0.78	0.78	-2.1	2.6
Aluminum (tempered)	0.4	1	0.63	0.63	-4	N/A
Aluminum (household foil, 1 mil)	0.53	1	0.73	0.73	-2.8	2.698
Aluminum (flame spray)	0.036	1	0.19	0.19	-14.4	N/A
Brass (91% Cu, 9% Zn)	0.47	1	0.69	0.69	-3.3	8.7
Brass (66% Cu, 34% Zn)	0.35	1	0.52	0.52	-5.7	8.5
Zinc	0.305	1	0.57	0.57	-4.9	7.134
Tin	0.151	1	0.39	0.39	-8.2	7.287
Superalloy	0.023	100,000	53.7	0.0005	-65.4	8.9
78 Permalloy	0.108	8,000	29.4	0.0037	-48.7	8.6
Purified Iron	0.17	5,000	29.2	0.0058	-44.7	7.85
Conetic AA	0.031	20,000	28.7	0.0011	-58.8	N/A
4-79 Permalloy	0.0314	20,000	25.1	0.0013	-58	N/A
Mumetal	0.0289	20,000	24	0.0012	-58.4	8.75
Permedur (50 Cu, 1-2 V, % Fe)	0.247	800	14.1	0.0018	-35.1	N/A
Hypernick	0.0345	4,500	12.5	0.0028	-51.1	N/A
45 Permalloy (1200 anneal)	0.0384	4,000	12.4	0.0031	-50.2	8.25
45 Permalloy (1050 anneal)	0.0384	2,500	9.8	0.0039	-48.1	8.25
Hot-Rolled Silicon Steel	0.0384	1,500	7.59	0.0051	-45.9	3.58
Sinimax	0.0192	3,000	7.59	0.0025	-51.9	1.04
4% Silicon Iron (grain oriented)	0.037	1,500	7.45	0.005	-46.1	N/A

IEEE Electrical Insulation Magazine Nov./Dec 1990-Vol.6, No. 6

Table (No.1) IEEE Electrical Insulation

Appendix D

MATLAB code for Interference Immunity test:

```
% Frequency Range
Freq=1e9:1e6:300*1e9;
% Absorption Loss for all Fields
K1 = 3.34; %Constant = 131.4 if l is meters; 3.34 if l is inches
%l = .00035; %Thickness in inches
% Superalloy Parameters
SA_ur = 1e5; %Permability
SA_gr = 0.023; %Conductivity
% Aluminum Parameters
Al_ur = 1; %Permability
Al_gr = 0.53; %Conductivity
% Copper Parameters
Co_ur = 1; %Permability
Co_gr = 0.1; %Conductivity
% Hot-polled silicon steel Parameters
Hpss_ur = 1500; %Permability
Hpss_gr = 0.0384; %Conductivity
% Hypernick Parameters
Hy_ur = 4500; %Permability
Hy_gr = 0.0345; %Conductivity
% Reflection Loss
C1 = 0.0117; %Coefficient for Magnetic = 0.0117 if r is meters
% = 0.462 if r is inches
C2 = 5.35; %Coefficient for Magnetic = 5.35 if r is meters
% = 0.136 if r is inches
```

```
C3 = 322;
r = 1; %Distance from EM Source to Shield in meters
% Reflection Loss for Magnetic Field
Rm_SA = 20 * log10((C1 ./ (r .* sqrt((Freq .* SA_gr) ./ SA_ur))) +...
(C2 .* (r .* sqrt((Freq .* SA_gr) ./ SA_ur))) + 0.354);
Rm_Al = 20 * log10((C1 ./ (r .* sqrt((Freq .* Al_gr) ./ Al_ur))) +...
(C2 .* (r .* sqrt((Freq .* Al_gr) ./ Al_ur))) + 0.354);
Rm_Co = 20 * log10((C1 ./ (r .* sqrt((Freq .* Co_gr) ./ Co_ur))) +...
(C2 .* (r .* sqrt((Freq .* Co_gr) ./ Co_ur))) + 0.354);
Rm_Hpss = 20 * log10((C1 ./ (r .* sqrt((Freq .* Hpss_gr) ./ Hpss_ur)))
+...
(C2 .* (r .* sqrt((Freq .* Hpss_gr) ./ Hpss_ur))) + 0.354);
Rm_Hy = 20 * log10((C1 ./ (r .* sqrt((Freq .* Hy_gr) ./ Hy_ur))) +...
(C2 .* (r .* sqrt((Freq .* Hy_gr) ./ Hy_ur))) + 0.354);
% Reflection Loss For Electric Field
Re_SA = C3 - (10 * log10((SA_ur * Freq.^3 * r.^2) / SA_gr));
Re_Al = C3 - (10 * log10((Al_ur * Freq.^3 * r.^2) / Al_gr));
Re_Co = C3 - (10 * log10((Co_ur * Freq.^3 * r.^2) / Co_gr));
Re_Hpss = C3 - (10 * log10((Hpss_ur * Freq.^3 * r.^2) / Hpss_gr));
Re_Hy = C3 - (10 * log10((Hy_ur * Freq.^3 * r.^2) / Hy_gr));
% Reflection Loss For Plane Wave
Rp_SA = 168 - 20 * log10(sqrt((Freq * SA_ur) / SA_gr));
Rp_Al = 168 - 20 * log10(sqrt((Freq * Al_ur) / Al_gr));
Rp_Co = 168 - 20 * log10(sqrt((Freq * Co_ur) / Co_gr));
Rp_Hpss = 168 - 20 * log10(sqrt((Freq * Hpss_ur) / Hpss_gr));
Rp_Hy = 168 - 20 * log10(sqrt((Freq * Hy_ur) / Hy_gr));
%%%%%%%%%%%%%%%%% Re-reflection correction factors
parameters %%%%
% Parameter m for r in meters for Magnetic Field
```

```
mM_SA = (4.7e-2 ./ r) .* sqrt(SA_ur ./ (Freq .* SA_gr));
mM_Al = (4.7e-2 ./ r) .* sqrt(Al_ur ./ (Freq .* Al_gr));
mM_Co = (4.7e-2 ./ r) .* sqrt(Co_ur ./ (Freq .* Co_gr));
mM_Hpss = (4.7e-2 ./ r) .* sqrt(Hpss_ur ./ (Freq .* Hpss_gr));
mM_Hy = (4.7e-2 ./ r) .* sqrt(Hy_ur ./ (Freq .* Hy_gr));
% Reflection Coefficient for Magnetic Field
GammaM_SA = 4 .* (((1 - (mM_SA.^2)).^2 - (2 .* (mM_SA.^2)) +...
(i * (2 .* sqrt(2)) .* mM_SA .* (1 - (mM_SA.^2)))) ./...
(((1 + (sqrt(2) .* mM_SA)).^2 + 1).^2));
GammaM_Al = 4 .* (((1 - (mM_Al.^2)).^2 - (2 .* (mM_Al.^2)) +...
(i * (2 .* sqrt(2)) .* mM_Al .* (1 - (mM_Al.^2)))) ./...
(((1 + (sqrt(2) .* mM_Al)).^2 + 1).^2));
GammaM_Co = 4 .* (((1 - (mM_Co.^2)).^2 - (2 .* (mM_Co.^2)) +...
(i * (2 .* sqrt(2)) .* mM_Co .* (1 - (mM_Co.^2)))) ./...
(((1 + (sqrt(2) .* mM_Co)).^2 + 1).^2));
GammaM_Hpss = 4 .* (((1 - (mM_Hpss.^2)).^2 - (2 .* (mM_Hpss.^2))
+...
(i * (2 .* sqrt(2)) .* mM_Hpss .* (1 - (mM_Hpss.^2)))) ./...
(((1 + (sqrt(2) .* mM_Hpss)).^2 + 1).^2));
GammaM_Hy = 4 .* (((1 - (mM_Hy.^2)).^2 - (2 .* (mM_Hy.^2)) +...
(i * (2 .* sqrt(2)) .* mM_Hy .* (1 - (mM_Hy.^2)))) ./...
(((1 + (sqrt(2) .* mM_Hy)).^2 + 1).^2));
%Parameter m for r in meters for Electric Field
mE_SA = 0.205e-16 * r * sqrt((SA_ur * Freq.^3) / SA_gr);
mE_Al = 0.205e-16 * r * sqrt((Al_ur * Freq.^3) / Al_gr);
mE_Co = 0.205e-16 * r * sqrt((Co_ur * Freq.^3) / Co_gr);
mE_Hpss = 0.205e-16 * r * sqrt((Hpss_ur * Freq.^3) / Hpss_gr);
mE_Hy = 0.205e-16 * r * sqrt((Hy_ur * Freq.^3) / Hy_gr);
%Reflection Coefficient for Electric Field
```

```
GammaE_SA = 4 .* (((1 - (mE_SA.^2)).^2 - (2 .* (mE_SA.^2)) -...
(i * (2 .* sqrt(2)) .* mE_SA .* (1 - (mE_SA.^2)))) ./...
(((1 - (sqrt(2) .* mE_SA)).^2 + 1).^2));
GammaE_Al = 4 .* (((1 - (mE_Al.^2)).^2 - (2 .* (mE_Al.^2)) -...
(i * (2 .* sqrt(2)) .* mE_Al .* (1 - (mE_Al.^2)))) ./...
(((1 - (sqrt(2) .* mE_Al)).^2 + 1).^2));
GammaE_Co = 4 .* (((1 - (mE_Co.^2)).^2 - (2 .* (mE_Co.^2)) -...
(i * (2 .* sqrt(2)) .* mE_Co .* (1 - (mE_Co.^2)))) ./...
(((1 - (sqrt(2) .* mE_Co)).^2 + 1).^2));
GammaE_Hpss = 4 .* (((1 - (mE_Hpss.^2)).^2 - (2 .* (mE_Hpss.^2)) -...
(i * (2 .* sqrt(2)) .* mE_Hpss .* (1 - (mE_Hpss.^2)))) ./...
(((1 - (sqrt(2) .* mE_Hpss)).^2 + 1).^2));
GammaE_Hy = 4 .* (((1 - (mE_Hy.^2)).^2 - (2 .* (mE_Hy.^2)) -...
(i * (2 .* sqrt(2)) .* mE_Hy .* (1 - (mE_Hy.^2)))) ./...
(((1 - (sqrt(2) .* mE_Hy)).^2 + 1).^2));
%Parameter m for r in meters for Plane Wave
mP_SA = 9.77e-10 .* sqrt((Freq .* SA_ur) / SA_gr);
mP_Al = 9.77e-10 .* sqrt((Freq .* Al_ur) / Al_gr);
mP_Co = 9.77e-10 .* sqrt((Freq .* Co_ur) / Co_gr);
mP_Hpss = 9.77e-10 .* sqrt((Freq .* Hpss_ur) / Hpss_gr);
mP_Hy = 9.77e-10 .* sqrt((Freq .* Hy_ur) / Hy_gr);
%Reflection Coefficient for Plane Wave
GammaP_SA = 4 .* (((1 - (mP_SA.^2)).^2 - (2 .* (mP_SA.^2)) -...
(i * (2 .* sqrt(2)) .* mP_SA .* (1 - (mP_SA.^2)))) ./...
(((1 + (sqrt(2) .* mP_SA)).^2 + 1).^2));
GammaP_Al = 4 .* (((1 - (mP_Al.^2)).^2 - (2 .* (mP_Al.^2)) -...
(i * (2 .* sqrt(2)) .* mP_Al .* (1 - (mP_Al.^2)))) ./...
(((1 + (sqrt(2) .* mP_Al)).^2 + 1).^2));
GammaP_Co = 4 .* (((1 - (mP_Co.^2)).^2 - (2 .* (mP_Co.^2)) -...
```

```
(i * (2 .* sqrt(2)) .* mP_Co .* (1 - (mP_Co.^2)))) ./...
(((1 + (sqrt(2) .* mP_Co)).^2 + 1).^2));
GammaP_Hpss = 4 .* (((1 - (mP_Hpss.^2)).^2 - (2 .* (mP_Hpss.^2)) -...
(i * (2 .* sqrt(2)) .* mP_Hpss .* (1 - (mP_Hpss.^2)))) ./...
(((1 + (sqrt(2) .* mP_Hpss)).^2 + 1).^2));
GammaP_Hy = 4 .* (((1 - (mP_Hy.^2)).^2 - (2 .* (mP_Hy.^2)) -...
(i * (2 .* sqrt(2)) .* mP_Hy .* (1 - (mP_Hy.^2)))) ./...
(((1 + (sqrt(2) .* mP_Hy)).^2 + 1).^2));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
option=input('This PROGRAM can help you to test existent SHIELD or
to design new one, just press the number of your situation:\n  1. Design
new Shield\n  2. Testing existent shield\n\n your selction is:  ');
if (option==1)
   l=input('Enter the expected Thickness (in INCHES) of the shield:  ');
% Absorption Loss Equations
A_SA = K1 * l * sqrt(Freq * SA_ur * SA_gr);
A_Al = K1 * l * sqrt(Freq * Al_ur * Al_gr);
A_Co = K1 * l * sqrt(Freq * Co_ur * Co_gr);
A_Hpss = K1 * l * sqrt(Freq * Hpss_ur * Hpss_gr);
A_Hy = K1 * l * sqrt(Freq * Hy_ur * Hy_gr);
% Re-Reflection Correction Factor
%Re-Reflection Correction Factor for Magnetic Field
CM_SA = 20 .* log(1 - (GammaM_SA .* (10.^(-A_SA ./ 10)) .*...
(cos(0.23 .* A_SA) - (i .* sin(0.23 .* A_SA)))));
CM_Al = 20 .* log(1 - (GammaM_Al .* (10.^(-A_Al ./ 10)) .*...
(cos(0.23 .* A_Al) - (i .* sin(0.23 .* A_Al)))));
CM_Co = 20 .* log(1 - (GammaM_Co .* (10.^(-A_Co ./ 10)) .*...
```

```
(cos(0.23 .* A_Co) - (i .* sin(0.23 .* A_Co)))));
CM_Hpss = 20 .* log(1 - (GammaM_Hpss .* (10.^(-A_Hpss ./ 10)) .*...
(cos(0.23 .* A_Hpss) - (i .* sin(0.23 .* A_Hpss)))));
CM_Hy = 20 .* log(1 - (GammaM_Hy .* (10.^(-A_Hy ./ 10)) .*...
(cos(0.23 .* A_Hy) - (i .* sin(0.23 .* A_Hy)))));
% Magnitude of Correction Factor for Magnetic Field
magCM_SA = abs(CM_SA);
magCM_Al = abs(CM_Al);
magCM_Co = abs(CM_Co);
magCM_Hpss = abs(CM_Hpss);
magCM_Hy = abs(CM_Hy);
%----------------------------------------------------------------------
%Re-Reflection Correction Factor for Electric Field
CE_SA = 20 .* log(1 - (GammaE_SA .* (10.^(-A_SA ./ 10)) .*...
(cos(0.23 .* A_SA) - (i .* sin(0.23 .* A_SA)))));
CE_Al = 20 .* log(1 - (GammaE_Al .* (10.^(-A_Al ./ 10)) .*...
(cos(0.23 .* A_Al) - (i .* sin(0.23 .* A_Al)))));
CE_Co = 20 .* log(1 - (GammaE_Co .* (10.^(-A_Co ./ 10)) .*...
(cos(0.23 .* A_Co) - (i .* sin(0.23 .* A_Co)))));
CE_Hpss = 20 .* log(1 - (GammaE_Hpss .* (10.^(-A_Hpss ./ 10)) .*...
(cos(0.23 .* A_Hpss) - (i .* sin(0.23 .* A_Hpss)))));
CE_Hy = 20 .* log(1 - (GammaE_Hy .* (10.^(-A_Hy ./ 10)) .*...
(cos(0.23 .* A_Hy) - (i .* sin(0.23 .* A_Hy)))));
% Magnitude of Correction Factor for Electric Field
magCE_SA = abs(CE_SA);
magCE_Al = abs(CE_Al);
magCE_Co = abs(CE_Co);
magCE_Hpss = abs(CE_Hpss);
magCE_Hy = abs(CE_Hy);
```

```
%------------------------------------------------------------------------
%Re-Reflection Correction Factor for Plane Wave
CP_SA = 20 .* log(1 - (GammaP_SA .* (10.^(-A_SA ./ 10)).*...
(cos(0.23 .* A_SA) - (i .* sin(0.23 .* A_SA)))));
CP_Al = 20 .* log(1 - (GammaP_Al .* (10.^(-A_Al ./ 10)).*...
(cos(0.23 .* A_Al) - (i .* sin(0.23 .* A_Al)))));
CP_Co = 20 .* log(1 - (GammaP_Co .* (10.^(-A_Co ./ 10)).*...
(cos(0.23 .* A_Co) - (i .* sin(0.23 .* A_Co)))));
CP_Hpss = 20 .* log(1 - (GammaP_Hpss .* (10.^(-A_Hpss ./ 10)).*...
(cos(0.23 .* A_Hpss) - (i .* sin(0.23 .* A_Hpss)))));
CP_Hy = 20 .* log(1 - (GammaP_Hy .* (10.^(-A_Hy ./ 10)).*...
(cos(0.23 .* A_Hy) - (i .* sin(0.23 .* A_Hy)))));
% Magnitude of Correction Factor for Plane Wave
magCP_SA = abs(CP_SA);
magCP_Al = abs(CP_Al);
magCP_Co = abs(CP_Co);
magCP_Hpss = abs(CP_Hpss);
magCP_Hy = abs(CP_Hy);
%Shielding Effectiveness For Magnetic Field
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%
SEm_SA = A_SA + Rm_SA - magCM_SA;
SEm_Al = A_Al + Rm_Al - magCM_Al;
SEm_Co = A_Co + Rm_Co - magCM_Co;
SEm_Hpss = A_Hpss + Rm_Hpss - magCM_Hpss;
SEm_Hy = A_Hy + Rm_Hy - magCM_Hy;
%Shielding Effectiveness For Electric Field
SEe_SA = A_SA + Re_SA - magCE_SA;
SEe_Al = A_Al + Re_Al - magCE_Al;
```

```
SEe_Co = A_Co + Re_Co - magCE_Co;
SEe_Hpss = A_Hpss + Re_Hpss - magCE_Hpss;
SEe_Hy = A_Hy + Re_Hy - magCE_Hy;
%Shielding Effectiveness For Plane Wave
SEp_SA = A_SA + Rp_SA - magCP_SA;
SEp_Al = A_Al + Rp_Al - magCP_Al;
SEp_Co = A_Co + Rp_Co - magCP_Co;
SEp_Hpss = A_Hpss + Rp_Hpss - magCP_Hpss;
SEp_Hy = A_Hy + Rp_Hy - magCP_Hy;
% Plot Shielding Effectiveness For Magnetic Field
subplot(2,2,1)
semilogx(Freq, SEm_SA, Freq, SEm_Al, Freq, SEm_Co, Freq,
SEm_Hpss, Freq, SEm_Hy);
gridon;
title('Shielding Effectiveness For Magnetic Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,2)
semilogx(Freq, SEe_SA, Freq, SEe_Al, Freq, SEe_Co, Freq, SEe_Hpss,
Freq, SEe_Hy);
gridon;
title('Shielding Effectiveness For Electric Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,[3,4])
semilogx(Freq, SEp_SA, Freq, SEp_Al, Freq, SEp_Co, Freq, SEp_Hpss,
Freq, SEp_Hy);
gridon;
title('Shielding Effectiveness For Plane Wave');
```

```
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
legend('Superalloy', 'Aluminum', 'Copper', 'Hot-polled silicon steel',
'Hypernick', -1);
else s=input('Please select the material of your shield from beloww
options, and enter the Thickness:\n 1. Superalloy\n 2.Aluminum\n
3.Copper\n 4.Hot-polled silicon steel\n 5.Hypernick\n  your material
number is:  ');
if (s<0)|(s>5)
display ('SORRY... We can not help you ^_^');
else
    l=input('Enter the Thickness (INCHES): ');
end
if (s==1)
% Absorption Loss Equations
A_SA = K1 * l * sqrt(Freq * SA_ur * SA_gr);
CM_SA = 20 .* log(1 - (GammaM_SA .* (10.^(-A_SA ./ 10)) .*...
(cos(0.23 .* A_SA) - (i .* sin(0.23 .* A_SA)))));
magCM_SA = abs(CM_SA);
CE_SA = 20 .* log(1 - (GammaE_SA .* (10.^(-A_SA ./ 10)) .*...
(cos(0.23 .* A_SA) - (i .* sin(0.23 .* A_SA)))));
magCE_SA = abs(CE_SA);
CP_SA = 20 .* log(1 - (GammaP_SA .* (10.^(-A_SA ./ 10)).*...
(cos(0.23 .* A_SA) - (i .* sin(0.23 .* A_SA)))));
magCP_SA = abs(CP_SA);
SEm_SA = A_SA + Rm_SA - magCM_SA;
SEe_SA = A_SA + Re_SA - magCE_SA;
SEp_SA = A_SA + Rp_SA - magCP_SA;
subplot(2,2,1)
```

```
semilogx(Freq, SEm_SA);
gridon;
title('Shielding Effectiveness For Magnetic Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,2)
semilogx(Freq, SEe_SA);
gridon;
title('Shielding Effectiveness For Electric Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,[3,4])
semilogx(Freq, SEp_SA);
gridon;
title('Shielding Effectiveness For Plane Wave');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
elseif (s==2)
% Absorption Loss Equations
A_Al = K1 * l * sqrt(Freq * Al_ur * Al_gr);
CM_Al = 20 .* log(1 - (GammaM_Al .* (10.^(-A_Al ./ 10)) .*...
(cos(0.23 .* A_Al) - (i .* sin(0.23 .* A_Al)))));
magCM_Al = abs(CM_Al);
CE_Al = 20 .* log(1 - (GammaE_Al .* (10.^(-A_Al ./ 10)) .*...
(cos(0.23 .* A_Al) - (i .* sin(0.23 .* A_Al)))));
magCE_Al = abs(CE_Al);
CP_Al = 20 .* log(1 - (GammaP_Al .* (10.^(-A_Al ./ 10)).*...
(cos(0.23 .* A_Al) - (i .* sin(0.23 .* A_Al)))));
magCP_Al = abs(CP_Al);
```

```
SEm_Al = A_Al + Rm_Al - magCM_Al;
SEe_Al = A_Al + Re_Al - magCE_Al;
SEp_Al = A_Al + Rp_Al - magCP_Al;
subplot(2,2,1)
semilogx(Freq, SEm_Al);
gridon;
title('Shielding Effectiveness For Magnetic Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,2)
semilogx(Freq, SEe_Al);
gridon;
title('Shielding Effectiveness For Electric Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,[3,4])
semilogx(Freq, SEp_Al);
gridon;
title('Shielding Effectiveness For Plane Wave');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
elseif (s==3)
% Absorption Loss Equations
A_Co = K1 * l * sqrt(Freq * Co_ur * Co_gr);
CM_Co = 20 .* log(1 - (GammaM_Co .* (10.^(-A_Co ./ 10)) .*...
(cos(0.23 .* A_Co) - (i .* sin(0.23 .* A_Co)))));
magCM_Co = abs(CM_Co);
CE_Co = 20 .* log(1 - (GammaE_Co .* (10.^(-A_Co ./ 10)) .*...
(cos(0.23 .* A_Co) - (i .* sin(0.23 .* A_Co)))));
```

```
magCE_Co = abs(CE_Co);
CP_Co = 20 .* log(1 - (GammaP_Co .* (10.^(-A_Co ./ 10)).*...
(cos(0.23 .* A_Co) - (i .* sin(0.23 .* A_Co)))));
magCP_Co = abs(CP_Co);
SEm_Co = A_Co + Rm_Co - magCM_Co;
SEe_Co = A_Co + Re_Co - magCE_Co;
SEp_Co = A_Co + Rp_Co - magCP_Co;
subplot(2,2,1)
semilogx(Freq, SEm_Co);
gridon;
title('Shielding Effectiveness For Magnetic Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,2)
semilogx(Freq, SEe_Co);
gridon;
title('Shielding Effectiveness For Electric Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,[3,4])
semilogx(Freq, SEp_Co);
gridon;
title('Shielding Effectiveness For Plane Wave');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
elseif (s==4)
% Absorption Loss Equations
A_Hpss = K1 * l * sqrt(Freq * Hpss_ur * Hpss_gr);
CM_Hpss = 20 .* log(1 - (GammaM_Hpss .* (10.^(-A_Hpss ./ 10)) .*...
```

```
(cos(0.23 .* A_Hpss) - (i .* sin(0.23 .* A_Hpss)))));
magCM_Hpss = abs(CM_Hpss);
CE_Hpss = 20 .* log(1 - (GammaE_Hpss .* (10.^(-A_Hpss ./ 10)) .*...
(cos(0.23 .* A_Hpss) - (i .* sin(0.23 .* A_Hpss)))));
magCE_Hpss = abs(CE_Hpss);
CP_Hpss = 20 .* log(1 - (GammaP_Hpss .* (10.^(-A_Hpss ./ 10)).*...
(cos(0.23 .* A_Hpss) - (i .* sin(0.23 .* A_Hpss)))));
magCP_Hpss = abs(CP_Hpss);
SEm_Hpss = A_Hpss + Rm_Hpss - magCM_Hpss;
SEe_Hpss = A_Hpss + Re_Hpss - magCE_Hpss;
SEp_Hpss = A_Hpss + Rp_Hpss - magCP_Hpss;
subplot(2,2,1)
semilogx(Freq, SEm_Hpss);
gridon;
title('Shielding Effectiveness For Magnetic Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,2)
semilogx(Freq, SEe_Hpss);
gridon;
title('Shielding Effectiveness For Electric Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,[3,4])
semilogx(Freq, SEp_Hpss);
gridon;
title('Shielding Effectiveness For Plane Wave');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
```

```
elseif (s==5)
% Absorption Loss Equations
A_Hy = K1 * l * sqrt(Freq * Hy_ur * Hy_gr);
CM_Hy = 20 .* log(1 - (GammaM_Hy .* (10.^(-A_Hy ./ 10)) .*...
(cos(0.23 .* A_Hy) - (i .* sin(0.23 .* A_Hy)))));
magCM_Hy = abs(CM_Hy);
CE_Hy = 20 .* log(1 - (GammaE_Hy .* (10.^(-A_Hy ./ 10)) .*...
(cos(0.23 .* A_Hy) - (i .* sin(0.23 .* A_Hy)))));
magCE_Hy = abs(CE_Hy);
CP_Hy = 20 .* log(1 - (GammaP_Hy .* (10.^(-A_Hy ./ 10)).*...
(cos(0.23 .* A_Hy) - (i .* sin(0.23 .* A_Hy)))));
magCP_Hy = abs(CP_Hy);
SEm_Hy = A_Hy + Rm_Hy - magCM_Hy;
SEe_Hy = A_Hy + Re_Hy - magCE_Hy;
SEp_Hy = A_Hy + Rp_Hy - magCP_Hy;
subplot(2,2,1)
semilogx(Freq, SEm_Hy);
gridon;
title('Shielding Effectiveness For Magnetic Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,2)
semilogx(Freq, SEe_Hy);
gridon;
title('Shielding Effectiveness For Electric Field');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
subplot(2,2,[3,4])
semilogx(Freq, SEp_Hy);
```

```
gridon;
title('Shielding Effectiveness For Plane Wave');
xlabel('Freqency (Hz)');
ylabel('Shielding Effectiveness (dB)');
end
end
```