CAPÍTULO 4. PLATAFORMA DE MEDIDA Y EQUIPOS

4.1 INTRODUCCIÓN

En este capítulo se pretende realizar una descripción de los equipos que componen la plataforma de medida automática que hemos utilizado en este proyecto, mostrada en la Figura 47. Como se puede observar, está compuesta por el generador de señal ***SMIQ02B de Rhode & Schwarz***, el analizador de espectros ***E4407B de Agilent,*** por el dispositivo bajo prueba (transistor funcionando como amplificador) y un PC.

La característica fundamental de estos equipos es que pueden ser controlados mediante el PC a través de conexiones GPIB, empleando secuencias de instrucciones en Matlab. Esto nos permite realizar medidas de forma automática o semiautomática.

Haciendo uso de las interfaces gráficas de usuario (GUIs) desarrolladas, se facilita aún más el entorno de trabajo, ya que se hace transparente al usuario.

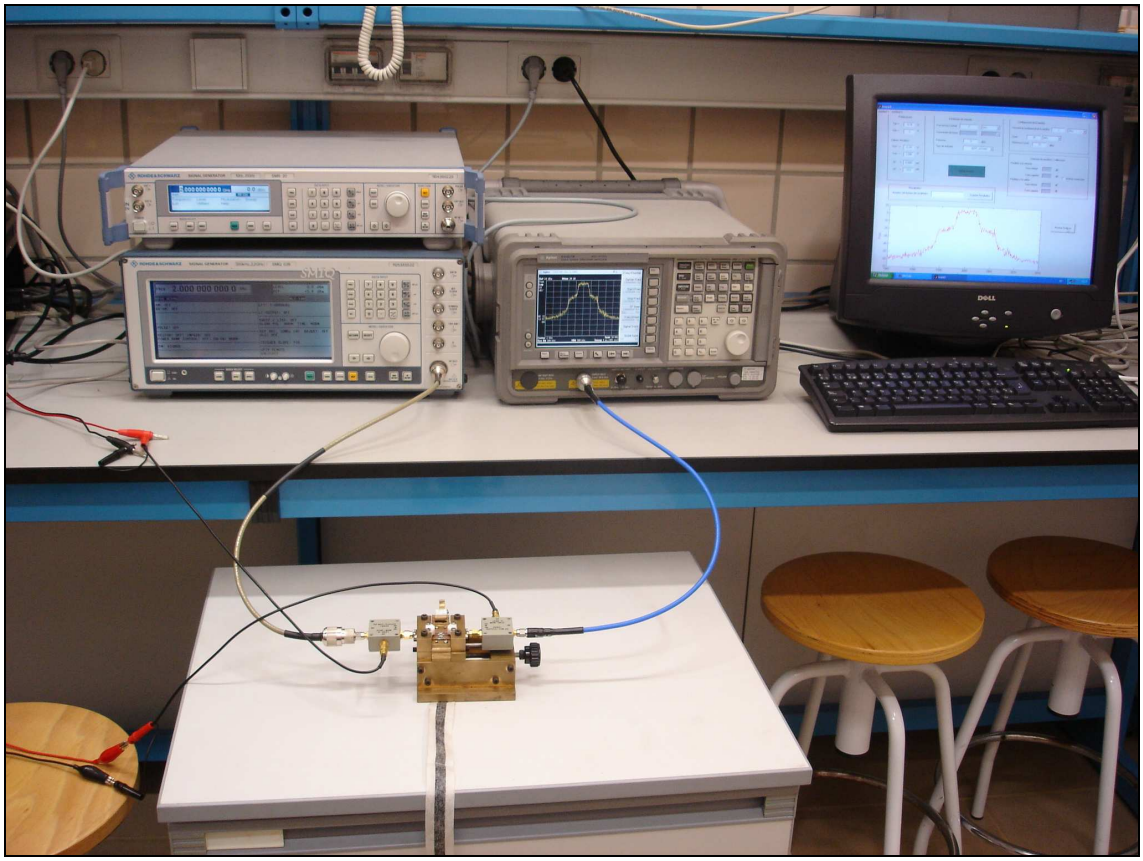

Figura 47. Plataforma de medida del laboratorio

4.2 EL GENERADOR VECTORIAL SMIQ02B DE RHODE & SCHWARZ

El generador de señal **SMIQ02B** (Vector Signal Generator 300 KHz to 2.2 GHz) permite generar un tono a frecuencia y potencia deseadas (que podrá ser utilizado por ejemplo como portadora en una modulación) y realizar modulaciones analógicas (AM y FM) y digitales (QPSK y FSK…) entre otras utilidades.

El SMIQ puede ser controlado desde el PC. Para que esto sea posible, es necesario establecer una conexión GPIB entre el ordenador y SMIQ.

Para comunicarnos con el equipo se emplean secuencias de instrucciones en Matlab. A cada uno de los botones del panel frontal, le corresponde una secuencia de instrucciones, de manera que el generador de señal pueda ser controlado de forma remota desde el PC.

4.2.1 Generador de forma de onda arbitraria (SMIQB60)

En este apartado se describe un método que permite obtener dos tonos usando un único generador, con la opción SMIQB60 del generador de señal SMIQB02, mediante modulación de la portadora con una forma de onda arbitraria definida externamente mediante Matlab.

La modulación que se realiza consiste en modular una portadora a frecuencia f_c, con una señal sinusoidal de frecuencia igual a la mitad de la separación entre los dos tonos $f_m = \frac{\Delta f}{2}$. La señal modulada resultante tiene la siguiente expresión:

$$s(t) = A\cos(\omega_m t)$$

$$c(t) = \cos(\omega_c t)$$

$$s(t)c(t) = A\cos(\omega_m t)\cos(\omega_c t) = A\frac{1}{2}[\cos[(\omega_c + \omega_m)t] + \cos[(\omega_c - \omega_m)t]]$$

Donde $f_c \pm f_m$, son las frecuencias correspondientes a los dos tonos. Por tanto, los dos tonos generados se encuentran separados una distancia dos veces la frecuencia de la señal moduladora.

Una vez que hemos generado las muestras de la señal moduladora de forma externa al generador, debemos pasarlas al SMIQ. Para ello utilizamos el software asociado al generador, el *IQWizard* y el *WinIQSIM*. Con el primero indicamos los ficheros donde se encuentran almacenadas las muestras y la frecuencia de muestreo con que esas muestras han sido obtenidas y con el segundo transmitimos estas muestras del PC al SMIQ con el puerto serie RS-232.

El proceso consiste básicamente en cargar previamente las formas de ondas para su posterior transmisión al SMIQ. Este paso se realiza a través de un programa denominado IQWizard. En este programa sólo necesitamos indicar cual es la componente en fase de la forma de onda generada, cual es la componente en cuadratura y por último cual es la frecuencia de muestreo, antes de cargar los datos en el WinIQSIM. Cuando hemos introducido todos estos valores, simplemente le damos a un botón que los carga para su envío al SMIQ.

A continuación, para enviarle al SMIQ las ondas generadas se utiliza el programa WinIQSIM. A través del menú *SMIQ(Arb)->Transmision* seleccionamos la forma de onda que queremos transmitir (que debe ser aquella que se había cargado previamente con el IQWizard).

El nombre con el que vamos a almacenar las formas de onda en el SMIQ de manera que nos sean fácilmente reconocibles cuando vayamos a utilizarlas es el siguiente: ‘x$\boxed{\Delta f \text{ en MHz}}$m’, es decir, la forma de onda de onda correspondiente a una separación de 1MHz se nombrará ‘x1m’ y cuando sea necesario poner un punto, añadiremos una ‘p’ del siguiente modo: ‘x0p5m’.

La ventaja de este procedimiento frente a otros que utilizan dos generadores sincronizados, no radica sólo en la reducción numérica de generadores, sino en que es posible controlar la diferencia de fases entre los dos tonos mediante definición por software de la modulación.

El principal problema de este método es que cuando aumenta la potencia de los tonos (potencia de portadora), aparecen nuevos tonos a frecuencias no deseadas, generados por el procesado digital de la señal realizado en el generador. En concreto, aparece un nuevo tono a la frecuencia de portadora y a las frecuencias de los productos de intermodulación, que afecta a la medida que se realiza a estas frecuencias. Sin embargo, estas respuestas no deseadas están 60dB por debajo de los tonos y permiten un margen suficiente para medir los productos de intermodulación en situaciones prácticas sin un error apreciable.

En la Figura 48 se muestra una fotografía del generador.

Figura 48. Generador de señal SMIQ02B de RHODE & SCHWARZ

4.3 EL GENERADOR DE SEÑAL SMR20 DE RHODE & SCHWARZ

Para realizar determinadas medidas ha sido necesario utilizar el generador SMR20 de Rohde & Schwarz que se muestra en la Figura 49.

Este generador también se encuentra conectado al PC, y puede ser controlado desde el ordenador a través de conexiones GPIB, empleando secuencias de instrucciones en Matlab, aunque su empleo mediante control remoto no forma parte de las medidas automatizadas que son objeto de este proyecto. Las instrucciones Matlab necesarias para controlar el equipo de forma remota son muy similares a las que se utilizan en el caso del SMIQ.

En las medidas realizadas sólo hemos utilizado el SMR20 para generar tonos a frecuencias superiores a 2GHz (los armónicos superiores al fundamental) que es la frecuencia máxima que permite el SMIQ. No obstante, el SMR20 ofrece otras posibilidades, por ejemplo, podemos realizar modulaciones analógicas (AM y FM) y digitales (ASK y FSK).

Figura 49. Generador SMR20 de Rohde & Schwarz

4.4 EL ANALIZADOR DE ESPECTROS E4407B DE AGILENT

El analizador de espectros utilizado ha sido el E4407B de Agilent. Con él, se han realizado las medidas que se describen en el capítulo quinto. Al igual que el generador de señal, este equipo también puede ser controlado de forma remota desde el PC usando secuencias de instrucciones Matlab. Éste se encuentra conectado al ordenador a través de una conexión GPIB.

El analizador de espectros tienes dos modos de funcionamiento. El ***Spectrum Analysis*** y el ***Modulation Analysis***.

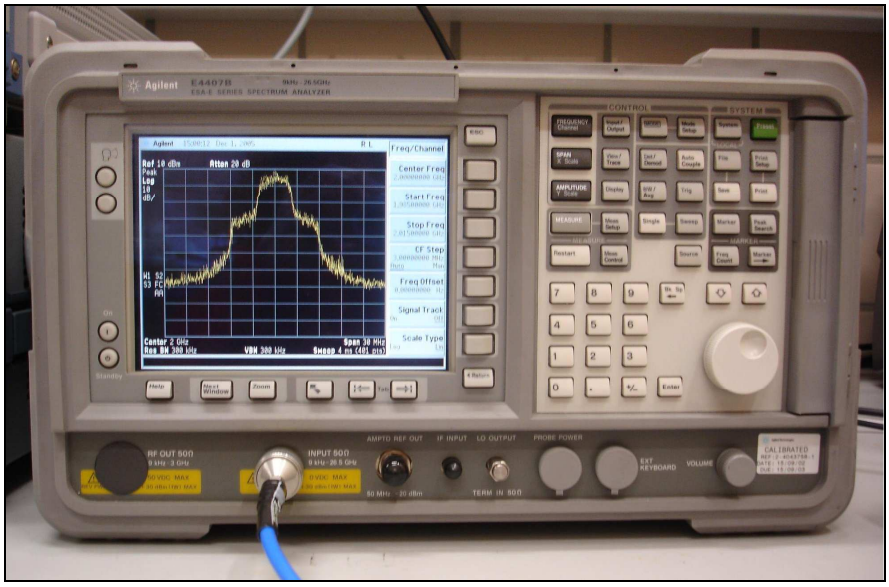

Figura 50. Analizador de espectros E4407B de Agilent

4.5 REALIZACIÓN DE LAS MEDIDAS

Como ya dijimos en el capítulo anterior, hemos tenido que hacer modificaciones a funciones ya existentes, para adaptarlas a nuestra plataforma de medida. Se trata de cuatro funciones que realizan las tres medidas que nos ocupan: barrido de potencia de señales de uno y dos tonos y la captura de la traza de una señal. A continuación se explica en qué consiste cada función.

Decíamos que existía en cada interfaz, un botón para iniciar la medida correspondiente. Este capítulo parte del momento en que ese botón es pulsado por el usuario, en el que para cada medida se realizará una llamada a una función distinta. Y concluye con el final de su *callback* correspondiente.

★ Caso de la interfaz para la medida de un tono:

```
[Pin, medida_1t,Vgsm, Vdsm, Igm, Idm] = medida_1t_potRe(Vgs, Vds, frec_in, frec_fund,
Pot_start, Pot_inc, Pot_stop, numarmonicos, correc_activa, correc_in, correc_out)
```

medida_1t_potRe , es la función encargada de transmitir las órdenes necesarias a los equipos de la plataforma de medida para que se realice el barrido de potencia de una señal de un tono con los valores de medida escritos por el usuario en la interfaz. Lo primero que haremos será aplicar la polarización al transistor.

```
% Se aplica la polarizacion al transistor
[Vgsm, Vdsm, Igm, Idm]=fuenteDC(Vds, Vgs);          % Dar valores a los valores medidos
```

Para ello usamos la función **fuenteDC,** que a su vez, nos devolverá los valores de polarización medidos reales.

```
fuente=visa('agilent','GPIB0::5::0::INSTR');
fopen(fuente);

% Polarizacion drenador-fuente
fprintf(fuente, 'VSET 2, %6.2f\n', Vds);
fprintf(fuente,'VOUT? 2');
vout2=fscanf(fuente,'%s');
Vdsm=str2num(vout2);                          % Vds medida expresada en V
fprintf(fuente,'IOUT? 2');
iout2=fscanf(fuente,'%s');
Idm=str2num(iout2)*1e3;                       % Id medida expresada en mA

% Polarizacion puerta-fuente.
% En este caso se han colocado los cables de polarizacion con la
% referencia de tierra cambiada para poder obtener una tension negativa,
% pero a la fuente hay que indicarle una tension positiva.
fprintf(fuente, 'VSET 1, %6.2f\n', -Vgs);
fprintf(fuente,'VOUT? 1');
vout1=fscanf(fuente,'%s');
Vgsm=-str2num(vout1);                         % Vgs medida expresada en V
fprintf(fuente,'IOUT? 1');
iout1=fscanf(fuente,'%s');
Igm=str2num(iout1);                           % Ig medida expresada en mA

fclose(fuente)
delete(fuente)
clear fuente
```

Una vez que tenemos polarizado el transistor, abrimos el generador de señal SMIQ y generamos el tono de entrada.

```
% Se especifica la señal de excitacion de entrada
smiq=visa('agilent','GPIB0::28::0::INSTR');          % Abrir el SMIQ
fopen(smiq);
fprintf(smiq,'*RST');
fprintf(smiq,'*CLS');

% Generamos la excitacion, una señal de 1 tono:
frec_str = num2str(frec_in);                         % Paso frecuencia de entrada a cadena
fprintf(smiq,'FREQ %s MHz',frec_str);                % Frecuencia de la portadora
Pot_str = num2str(Pot_start);                        % Paso potencia inicial a cadena
fprintf(smiq,'POW %s dBm',Pot_str);    % Potencia de la portadora. Para empezar la potencia inicial
fprintf(smiq,'OUTP:STAT ON')                         % Se genera el tono

fclose(smiq)                                         % Cerramos el SMIQ
delete(smiq)
clear smiq
```

Abrimos también el analizador de espectro y configuramos la medida.

```
% Se especifican las condiciones de medida
ade=visa('agilent','GPIB0::18::0::INSTR');                    % Abrimos el ADE
ade.InputBufferSize=8000;
fopen(ade);
fprintf(ade,'*RST');

frecfund_str = num2str(frec_fund, 10);                        % Para verlo con precision
fprintf(ade,'SENS:FREQ:CENT %s MHz', frecfund_str);   % Frecuencia central en pantalla del
                                                                 analizador
span=frec_fund/100;

fprintf(ade,'UNIT:POW DBM');                                  % Medida de potencia en dBm
fprintf(ade,'INIT:CONT 0');                                % Establece el ADE en modo barrido único
fprintf(ade,':DISP:WIND:TRAC:Y:LOG:RANG:AUTO OFF');    % Preamplificador desconectado

P = Pot_start:Pot_inc:Pot_stop;                       % Vector con los valores de potencia a barrer
Np = length(P);                                       % Longitud del barrido de potencias

for i=1:Np,                                           % Iniciamos el barrido de potencia

  smiq=visa('agilent','GPIB0::28::0::INSTR');         % Abrir el SMIQ
  fopen(smiq);
  P_string=num2str(P(i));
  fprintf(smiq,'POW %s dBm',P_string);                % Potencia de la portadora generada por el SMIQ
  fprintf(smiq,'OUTP:STAT ON');

  fclose(smiq)                                        % Cerramos el SMIQ
  delete(smiq)
  clear smiq

  rlevel = P(i) + 25;
  RL = num2str(rlevel);
  fprintf(ade,':DISP:WIND:TRAC:Y:RLEV %s DBM', RL);   % Ref level en dBm
  BWres1 = span;
  BWres2 = num2str(BWres1);
  fprintf(ade,'BAND %s KHz', BWres2);
```

```
% Voy rellenando fila(i) con columnas 1,2,3... hasta numarmonicos
   switch numarmonicos
      case 1
         medida_1t(i,1) = medida(frec_fund,span,rlevel,BWres1,ade);
      case 2
         medida_1t(i,1) = medida(frec_fund,span,rlevel,BWres1,ade);
         medida_1t(i,2) = medida(2*frec_fund,span,rlevel-10,BWres1,ade);
      case 3
         medida_1t(i,1) = medida(frec_fund,span,rlevel,BWres1,ade);
         medida_1t(i,2) = medida(2*frec_fund,span,rlevel-10,BWres1,ade);
         medida_1t(i,3) = medida(3*frec_fund,span,rlevel-15,BWres1,ade);
      case 4
         medida_1t(i,1) = medida(frec_fund,span,rlevel,BWres1,ade);
         medida_1t(i,2) = medida(2*frec_fund,span,rlevel-10,BWres1,ade);
         medida_1t(i,3) = medida(3*frec_fund,span,rlevel-15,BWres1,ade);
         medida_1t(i,4) = medida(4*frec_fund,span,rlevel-20,BWres1,ade);
      case 5
         medida_1t(i,1) = medida(frec_fund,span,rlevel,BWres1,ade);
         medida_1t(i,2) = medida(2*frec_fund,span,rlevel-10,BWres1,ade);
         medida_1t(i,3) = medida(3*frec_fund,span,rlevel-15,BWres1,ade);
         medida_1t(i,4) = medida(4*frec_fund,span,rlevel-20,BWres1,ade);
         medida_1t(i,5) = medida(5*frec_fund,span,rlevel-25,BWres1,ade);
      otherwise
         text(0,1,'{numero de armonicos erroneo}')
         hold off
         medida_1t=[ ];
   end

end

fclose(ade)                          % Cerramos el analizador
delete(ade)
clear ade
```

El barrido de potencia para cada armónico se efectúa con la función **medida** que va capturando la potencia máxima mostrada en la pantalla del analizador de espectros para cada uno de ellos. La variable resultante *medida_1t*, será una matriz con tantas columnas como número de armónicos haya seleccionado el usuario.

```
frec2=num2str(frec1,10);                          % Frecuencia central en pantalla del analizador
fprintf(ade,':SENS:FREQ:CENT %s MHZ',frec2);
span2=num2str(span);                              % Frecuencia central en pantalla del analizador
fprintf(ade,':SENS:FREQ:SPAN %s MHZ',span2);    % SPAN en el analizador
fprintf(ade,'UNIT:POW DBM');                      % Medida de potencia en dBm
fprintf(ade,'INIT:CONT 1');                       % Modo de barrido continuo
fprintf(ade,'INIT:IMM');
rlevel2=num2str(rlevel);                          % Frecuencia central en pantalla del analizador
fprintf(ade,':DISP:WIND:TRAC:Y:RLEV %s DBM',rlevel2);        % Ref level en dBm
fprintf(ade,':DISP:WIND:TRAC:Y:LOG:RANG:AUTO OFF');       %  IF GAIN fixed
BWres2=num2str(BWres1);
fprintf(ade,'BAND %s KHz',BWres2);

fprintf(ade,'CALC:MARK:PEAK:EXC dB');             % Indicamos que se reconozcan como señales
fprintf(ade,'CALC:MARK:PEAK:THR -90 \n');         % aquellas con un nivel superior a -90dB
fprintf(ade,'*WAI');                              % Esperar a que se estabilice la señal
fprintf(ade,'CALC:MARK:MAX');                     % Situamos el cursor en el pico máximo
fprintf(ade,'CALC:MARK:CPE ON');
fprintf(ade,'INIT:CONT 0');                       % Modo de barrido único

fprintf(ade,'CALC:MARK:Y?');                      % Obtenemos el valor del máximo
potencia=fscanf(ade,'%lf');                       % Potencia del máximo en pantalla del ADE

ade.ValuesReceived;
```

Una vez que tenemos todas las potencias en *medida_1t*, sólo nos queda aplicar las correcciones por pérdidas si procede. *Correc_out(k)* será el valor de pérdida para el armónico k y *correc_in*, el valor de pérdida a la entrada introducido.

```
% Correcciones a la entrada y a la salida
for k=1:numarmonicos,
   medida_1t(:,k) = medida_1t(:,k) + correc_out(k);   % Potencia a la salida del transistor corregida.
end
Pin = P' - correc_in;                                 % Potencia a la entrada del transistor
```

En este punto concluirían las funciones del botón iniciar medida, por tanto el objetivo de este capítulo para la medida de un tono.

★ Caso de la interfaz para el barrido de potencia de una señal de dos tonos.

```
[Pin, medida_2t,Vgsm, Vdsm, Igm, Idm] = medida_2t_potRe(Vgs, Vds, frec_in, Df, frec_fund, Pot_start, Pot_inc, Pot_stop, Intermod, correc_activa, correccion_IN, correc_out)
```

medida_2t_potRe, al igual que antes, es la que se va a encargar de introducir, mediante instrucciones específicas, los valores dados por el usuario en la interfaz en los equipos de medida. Para ello, seguiremos los mismos pasos con alguna modificación que se será indicada.

Primero polarizamos al transistor con la llamada a la función **fuenteDC.**

```
% Se aplica la polarizacion al transistor
[Vgsm, Vdsm, Igm, Idm]=fuenteDC(Vds, Vgs);
```

Abrimos el generador de señal exactamente de la misma manera que antes, pero introduciendo el siguiente código necesario para generar dos tonos con el mismo generador. Según la forma en que generamos los tonos, la potencia de la portadora estará 3dB por encima de la potencia de estos (los tonos tienen la mitad de amplitud que la portadora), es decir, si en la interfaz escribimos una potencia de -10dBm, en el SMIQ tendrá que aparecer una potencia de portadora de -7 dBm, de ahí la explicación de las siguientes líneas de código.

```
% Para generar una excitacion de 2 tonos:
onda = formadeonda(Df);                  % Hay que cargar previamente la forma de onda
fprintf(smiq,':ARB:WAV:SEL %s', onda)  %Fichero almacenado en el SMIQ en el que se encuentran
                                         % las muestras necesarias para generar los dos tonos
fprintf(smiq,':ARB:STAT ON')
Pot_stop=Pot_stop+3;
Pot_start=Pot_start+3;                   % se pierden 3 dB de potencia
Pot_str = num2str(Pot_start);            % Paso Potencia a cadena
fprintf(smiq,'POW %s dBm',Pot_str);
```

La función **formadeonda** obtiene el nombre de la forma de onda adecuada previamente cargada en función de la separación de frecuencia *Df* que se le pase.

```
wave1 = strcat('x', num2str(Df), 'm');
wave2 = strrep(wave1, '.', 'p');
onda = strcat(char(39), wave2, char(39));
```

Una vez introducida la excitación de entrada, se pasará a la configuración de la medida, abriendo el analizador de espectro, como se comentó anteriormente, pero realizando el barrido de potencia de la forma que sigue:

```
P = Pot_start:Pot_inc:Pot_stop;                    % Vector con los valores de potencia a barrer
Np = length(P);                                    % Longitud del barrido de potencias

for i=1:Np,                                        % Iniciamos el barrido de potencia

   smiq=visa('agilent','GPIB0::28::0::INSTR');     % Abrir el SMIQ
   fopen(smiq);
   P_string=num2str(P(i));
   fprintf(smiq,'POW %s dBm',P_string);            % Potencia de la portadora generada por el SMIQ
   fprintf(smiq,'OUTP:STAT ON');

   fclose(smiq)                                    % Cerramos el SMIQ
   delete(smiq)
   clear smiq

   rlevel = P(i) + 25;
   RL = num2str(rlevel);
   fprintf(ade,':DISP:WIND:TRAC:Y:RLEV %s DBM', RL);        % Ref level en dBm
   BWres1 = span;
   BWres2 = num2str(BWres1);
   fprintf(ade,'BAND %s KHz', BWres2);

   f2= frec_in +(Df/2);                            % Frecuencia superior
   f1= frec_in -(Df/2);                            % Frecuencia inferior

   if Intermod == 1
      medida_2t(i,1) = medida((2*f1)-f2,span,rlevel-20,BWres1,ade);      %Col 1: IM3 Inf
      medida_2t(i,2) = medida(f1,span,rlevel,BWres1,ade);                %Col 2: Tono Inf
      medida_2t(i,3) = medida(f2,span,rlevel,BWres1,ade);                %Col 3: Tono Sup
      medida_2t(i,4) = medida((2*f2)-f1,span,rlevel-20,BWres1,ade);      %Col 4: IM3 Sup
   else
      medida_2t(i,1) = medida((3*f1-2*f2),span,rlevel-30,BWres1,ade);    %Col 1: IM5 Inf
      medida_2t(i,2) = medida((2*f1)-f2,span,rlevel-20,BWres1,ade);      %Col 2: IM3 Inf
      medida_2t(i,3) = medida(f1,span,rlevel,BWres1,ade);                %Col 3: Tono Inf
      medida_2t(i,4) = medida(f2,span,rlevel,BWres1,ade);                %Col 4: Tono Sup
      medida_2t(i,5) = medida((2*f2)-f1,span,rlevel-20,BWres1,ade);      %Col 5: IM3 Sup
      medida_2t(i,6) = medida((3*f2-2*f1),span,rlevel-30,BWres1,ade);    %Col 6: IM5 Sup
     end
end
```

Donde la función **medida** como ya se ha explicado, devuelve la potencia a la frecuencia que se pasa como parámetro. La variable resultado *medida_2t* será una matriz con cuatro o seis columnas, dependiendo del orden máximo de los productos de intermodulación que se hayan especificado en la interfaz y con el orden descrito en el código.

Por último, realizamos la corrección de las pérdidas si es preciso a *medida_2t.* La variable ***correc_out(k)*** contendrá el valor de la pérdida a la salida para el tono k y ***correccion_IN*** es un vector de dos elementos de pérdidas para el tono inferior y superior.

```
% Correcciones a la salida
if Intermod==1

   for k=1:4,
      medida_2t(:,k) = medida_2t(:,k) + correc_out(k);   %Potencia a la salida del transistor corregida.
   end
else
   for k=1:6,
      medida_2t(:,k) = medida_2t(:,k) + correc_out(k);   %Potencia a la salida del transistor corregida.
   end
end

Pin_inf = P'-3-correccion_IN(1);                          %Potencia a la entrada del transistor corregida
Pin_sup = P'-3-correccion_IN(2);
Pin = [Pin_inf, Pin_sup];
```

★ Interfaz para la captura de la traza de una señal.

```
[f,tr1,Pin,Vgsm, Vdsm, Igm, Idm] = captura_trazaR(Vgs, Vds, frec_in, Df, frec_fund, Pot,...
Potencia, Span, RefLevel, correc_activa, correccion_IN, correc_out);
```

La función **captura_trazaR** es la encargada de capturar la traza, como su nombre indica, de señales de un tono, dos tonos o una modulación WCDMA. Primero, por tanto, se pasarán los valores de polarización mediante la llamada a **fuenteDC**, como ya hemos indicado. Luego, abriremos el SMIQ para poder transferir los datos de excitación de la entrada del mismo modo que ya conocemos, pero habrá que incluir un código que diferencie el tipo señal de entrada que ha especificado el usuario.

```
% Si se trata de una excitación de 2 tonos o una señal 3GPP-WCDMA:
if Pot==2                                          % Es una excitación de 2 tonos:
   onda = formadeonda(Df);                         % Hay que cargar previamente la forma de onda
   fprintf(smiq.':ARB:WAV:SEL %s', onda)   %Fichero almacenado en el SMIQ en el que se
                                          %encuentran las muestras necesarias para generar los tonos.
   fprintf(smiq.':ARB:STAT ON')
   Potencia=Potencia+3;                            % En este caso se pierden 3 dB de potencia
   Potencia_str = num2str(Potencia);                                % Paso Potencia a cadena
   fprintf(smiq,'POW %s dBm',Potencia_str);
   correccion_entrada=((correccion_IN(1)+correccion_IN(2))/2)+3;       %Tomo el valor medio
   correccion_salida=(correc_out(1)+correc_out(2))/2 ;
elseif Pot==3                                      % Es una excitación 3GPP-WCDMA
   fprintf(smiq.':DM:STAN QWCD');                  % Se indica modulación WCDMA
   fprintf(smiq.':DM:STAT ON');
   correccion_entrada=correccion_IN(1);    % Me quedo con el primer termino ya que el segundo era 0
   correccion_salida=correc_out(1);
else
   correccion_entrada=correccion_IN(1);    % Me quedo con el primer termino ya que el segundo era 0
   correccion_salida=correc_out(1);
end
```

De esta manera, si la excitación de entrada (*Pot*) es una señal de dos tonos, habrá que llamar a la función **formadeonda** como ya explicamos, y si es una señal con modulación WCDMA se le indicará al generador de señal con las correspondientes instrucciones.

Seguidamente se abrirá el ADE para que sea controlado desde el ordenador y se le transmitirán los valores que configuran la medida. Los resultados que se obtendrán serán la traza de la señal (*tr1*) y la potencia de entrada, (*Pin*), corregidas, así como el rango de frecuencias apropiado para la representación del espectro. Las variables ***correccion_entrada*** y ***correccion_salida*** corresponden a dos números creados en el bucle anterior.

```
fprintf(ade,'*IDN?');
idn=fscanf(ade);
fprintf(ade,'*RST');
fprintf(ade,'SENS:SWE:POIN?');                    % Por defecto, si no se modifica este
Npoin=fscanf(ade,'%d');                           % parámetro, se toman 401 puntos

frecfund_str = num2str(frec_fund, 10);              % Para fijar la frecuencia fundamental
fprintf(ade,'SENS:FREQ:CENT %s MHz', frecfund_str);

span_str = num2str(Span, 10);                             % Para fijar el Span
fprintf(ade,'SENS:FREQ:SPAN %s MHZ', span_str);

Rf_str = num2str(RefLevel, 10);                           % Para fijar el Ref.Level
fprintf(ade,':DISP:WIND:TRAC:Y:RLEV %s DBM',Rf_str);
fprintf(ade,'INIT:CONT 0');                               % Modo de barrido único. La
                                                          % medida se realiza una sola vez.
fprintf(ade,'INIT:IMM; *WAI');                  % Espera a que se estabilice la señal
fprintf(ade,'UNIT:POW DBM');                    % Se garantiza que las unidades serán dBm
fprintf(ade,'FORM:TRAC:DATA ASC');
fprintf(ade,'INIT:IMM; *WAI');                  % Espera a que se estabilice la señal
fprintf(ade,'TRAC:DATA? TRACE1');
data=fscanf(ade,'%s');
                                                  % tr1 contiene las muestras de la traza
tr1=str2num(data) + correccion_salida;            % procedentes del ADE
Pin = Potencia - correccion_entrada;
f=linspace((frec_fund -Span/2),(frec_fund +Span/2),Npoin);
fprintf(ade,'INIT:CONT 1');                        %Modo de barrido continuo.
```