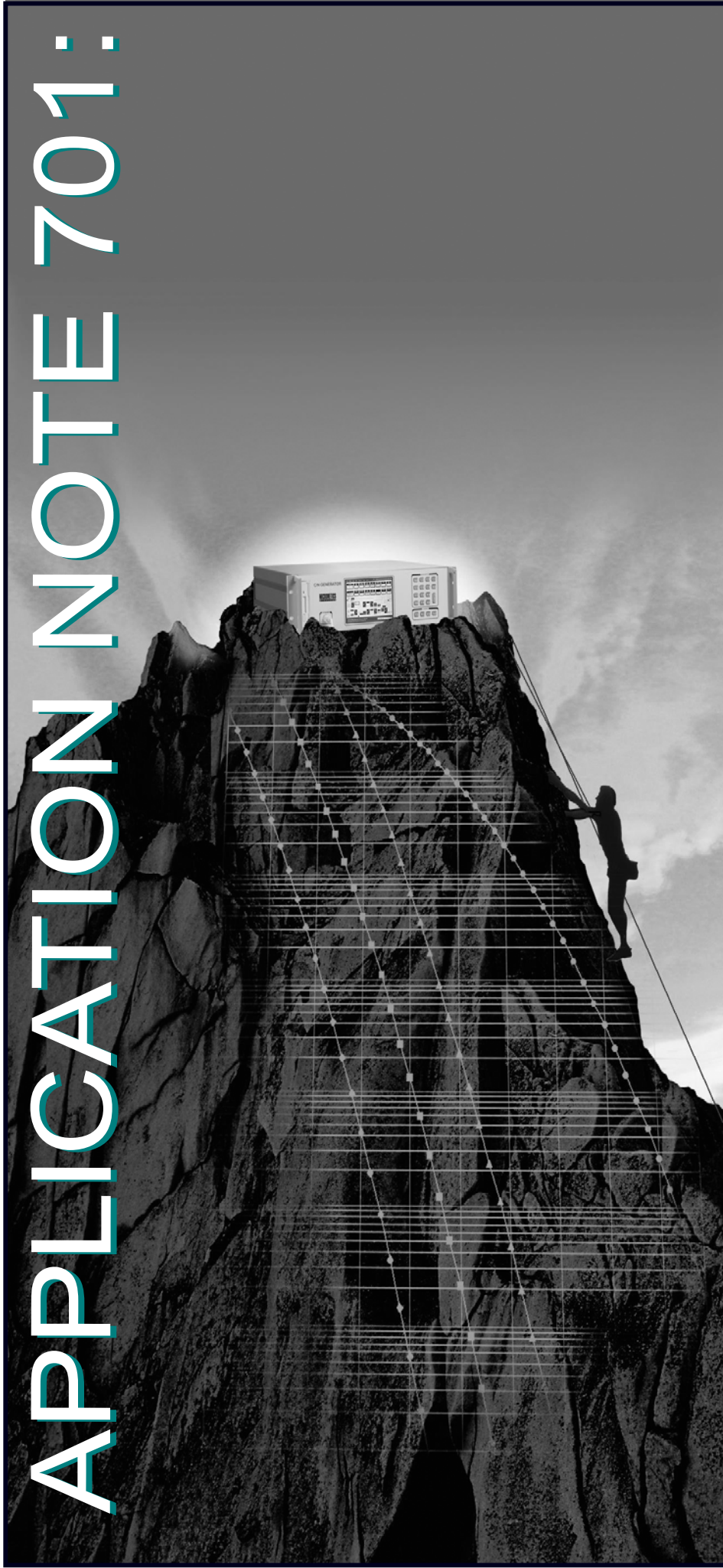


APPLICATION NOTE 701:



Achieving
High
Accuracy
 E_b/N_0 (C/N)
Ratios

MICRONETICS
TEST SOLUTIONS

Introduction:

The CNG series of generators have the ability to make highly accurate E_b/N_0 or C/N ratios when used correctly. The basic operation of the CNG is depicted by the RF test block diagram in *Figure 1*, which is on the front panel of the instrument, dynamically displayed with the internal power meter. Prior to entering a ratio, the instrument measures the test signal's amplitude with the embedded power measurement system. Once it has this value it prompts the user to enter in an E_b/N_0 ratio within the instruments allowable range. Upon entering the values, the instrument's CPU instructs the attenuator in the noise path to automatically actuate to the correct value to achieve the set ratio. In theory this sounds like an exceedingly simple device. The challenge is to be able to set this ratio accurately over multi-parameter variables such as frequency, bandwidth and test signal amplitude. As is clear from the BER vs. E_b/N_0 waterfall curve shown in *Figure 2*, any uncertainty in E_b/N_0 translates dramatically to uncertainty of BER. The combination of the CNG's hardware and software algorithms, calibration routines and user input parameters all work in unison to insure accurate E_b/N_0 . The three detailed items to follow are the most critical in this process.



Figure 1

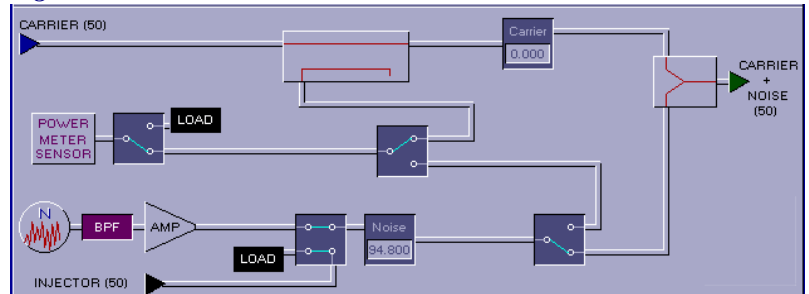
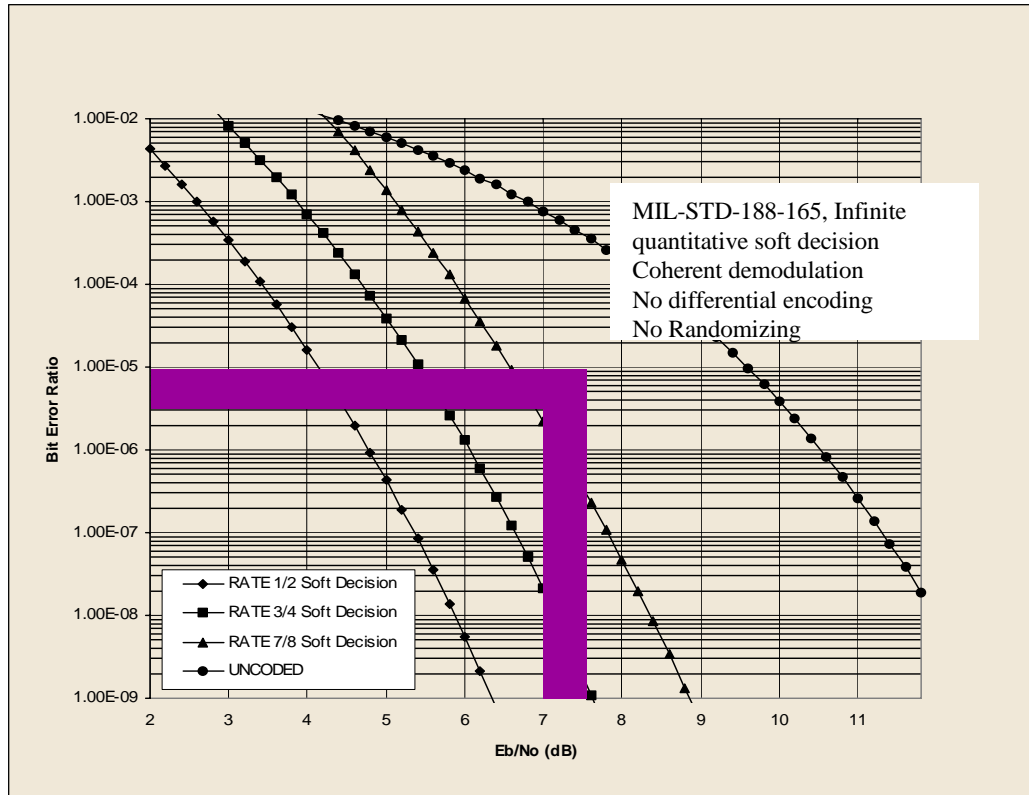


Figure 2



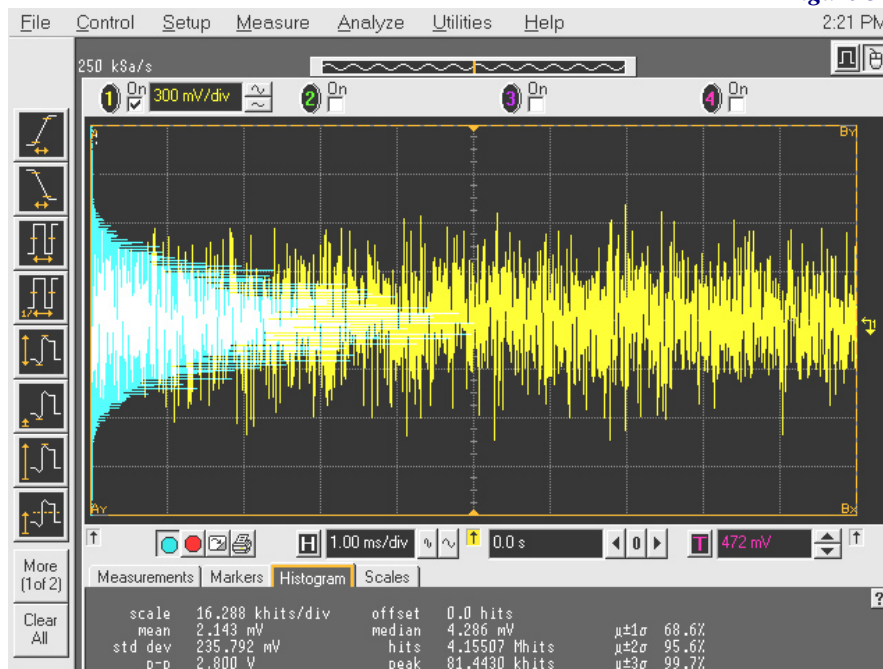
1. Power Measurement System: The heart of the instrument's accuracy starts with the power meter. Micronetics uses a sophisticated complex signal power meter to get a highly accurate reading of the modulated test signal. This power meter has a self-adjusting time constant requiring no external configuration. It is able to measure RMS signal amplitude (the value used to base the E_b/N_0 ratio on)

accurately including the very complex 7/8 coded QPSK signals or 256 QAM signals. The power measurement system uses a highly stable reference signal as a built-in power meter calibrator. This ensures long-term accuracy.

2. Amplified Noise Module:

- Amplitude Stability** - A stable amplified noise module is imperative for accuracy. Although the noise signal is Gaussian in probability density function and therefore is continuously jumping in amplitude, it averages to an RMS level over time. It is extremely important that this RMS power is stable. Any change in the noise RMS power will directly affect the E_b/N_0 ratio. The CNG unit calibrates this value upon command from the user and upon instrument power on. Some tests may require a lengthy period of time (sometimes weeks) if the bit error rate (BER) is very low. It can take that long to achieve a statistically accurate BER. It is impossible to recalibrate mid-stream during a test, which is why long-term stability is important. Micronetics has pioneered the use of amplified noise modules for these applications and others using state-of-the-art technology for RMS power stability needed for long term E_b/N_0 versus BER tests.
- Gaussianity** - Micronetics, the pioneer of solid-state noise technology, uses only analog Gaussian noise sources, not microprocessor generated pseudo-random noise (PN). The E_b/N_0 vs. BER relation assumes an ideal Gaussian source. The advantage of analog over digital is that all frequencies are contained as opposed to PN noise in which frequencies are stepped with the resolution being limited to the processing power of the PN generator. Secondly the extreme peaks at the tails of the bell curve are simply not generated by the PN chip so as to falsely make a test look falsely better. The extreme peaks, while statistically rare, are significant as they are apt to always produce bit errors. An analog source does not have infinite peaks but the extremes are at worst case compressed rather than simply not generated at all. Micronetics ensures that the compression point is significantly high so as not to affect the E_b/N_0 vs. BER test. Micronetics measures this as well as conformance to the bell shaped curve by making histogram measurements of the amplified noise module inside the CNG. *Fig 3* is a screen capture of such a measurement.

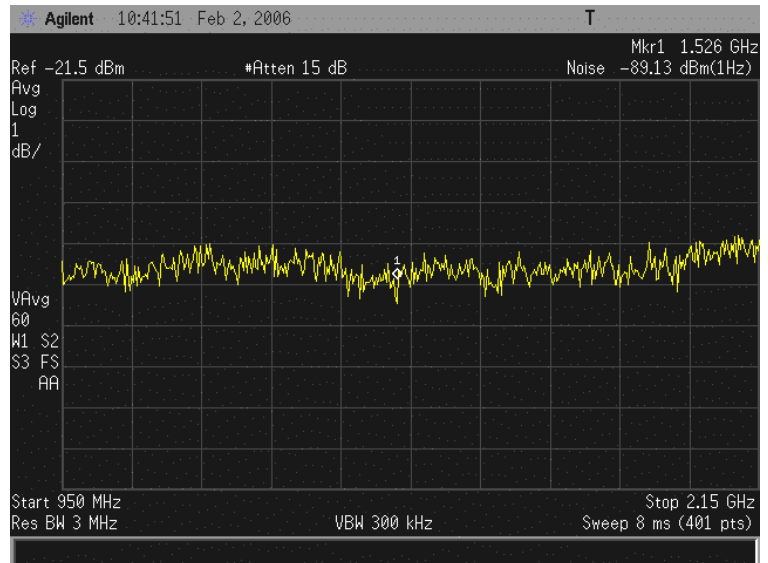
Figure 3



3. Spectral Calibration: The hardware of the CNG can only provide accuracy to a point. The spectral calibration is what ensures accuracy over frequency. This is most evident with the L-Band units, which operate from 950 MHz to 2250 MHz. Micronetics does everything that is practically possible to achieve a flat power spectrum with its noise module.

Figure 4

An equalizer is even used in the L-Band box to help. However, it is impossible to have perfect flatness. *Figure 4* shows actual data of the equalized noise module output of an L-band unit.

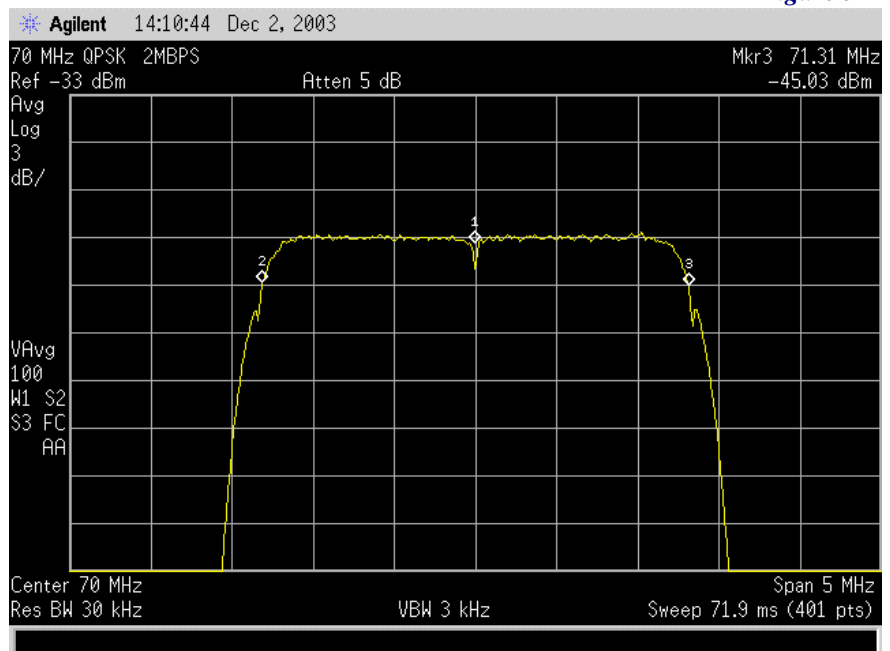


What is more, there are several other paths including the test signal path to the power meter, to the output, and the signal+noise combiner all of which are not perfectly flat. At 70 and 140 MHz, flatness is quite good, but in the L-band, cable loss is proportional to frequency. This all has to be accounted for. The difficult brute force way is to try to attempt to flatten all these paths with extensive equalizers, however, Micronetics

has pioneered the use of spectral calibration as a means to use spectral path characterization and software to achieve E_b/N_0 accuracy. The noise module and each of the RF paths are calibrated across the entire operating frequency at discrete frequency points with fine resolution. This data is stored in look up tables in the instruments memory. The CPU automatically chooses the correct frequency and bandwidth depending on the user's test. The user must enter into the CNG the signal center frequency and the noise bandwidth to invoke the spectral calibration. Without entering this critical information, the CNG simply uses the default setting which simply uses calibration table values corresponding to the instrument's mid band point which is around 1600 MHz for the L-band models. The center frequency entered should be that of the modulated test signal. The bandwidth may require some further analysis based on the modulation scheme and data rate being used. Some people use the symbol rate to loosely equate this with bandwidth. One way to help determine the BW is to look at the output of the modulator on a spectrum analyzer.

Figure 5

Figure 5 shows a BPSK signal with markers at the 3 dB points. Using the 3 dB BW is a good approximation of the actual value to be used. It is not imperative to be spot on with respect to the bandwidth as long as it is in the ballpark.



MEASUREMENT RESOLUTION AND MAXIMUM ACCURACY

Measurement Resolution of 0.01 dB:

The CNG instrument has a measurement resolution of 0.01 dB, which means that the actual E_b/N_0 is stated on the front panel to 1/100 dB. However, the resolution the user can command the CNG to set is only 1/10 of a dB due to step size of the attenuators inside. An E_b/N_0 setting of 6.5 dB might actually yield a ratio of 6.47 for example. The CNG will automatically choose a setting that is closest to the commanded ratio. The actual ratio is displayed on the front panel and made available over the GPIB or RS/232 bus. When making communication system measurements of BER vs. E_b/N_0 it is most accurate to use the actual ratio (6.47 dB) as opposed to what was commanded for the instrument to set.

Algorithm for highest accuracy when making automated long-term measurements:

A typical automated program will automatically set the CNG to an E_b/N_0 ratio, measure the BER of the device under test, step and repeat using a different E_b/N_0 and then plot the E_b/N_0 vs. BER waterfall curve. For the most accurate results, it is best to use the actual ratio rather than the commanded ratio as described above. Depending on the bit error rate being measured, an E_b/N_0 vs. BER data point could take hours, days, weeks or even months to be statistically valid. During the course of this test, if there is any drift in amplitude to the test signal input coming into the CNG, this will affect the E_b value (E_b is proportional to signal power) and therefore the E_b/N_0 ratio. If the drift is significant in comparison to how accurate the measurement needs to be, it should be accounted for. The CNG displays the signal power (normalized to the output port of the CNG) dynamically so as to account for any change. An automated program should take signal power readings at regular intervals throughout the test and average the readings. The program should also poll the CNG for N_0 as well. Finally the actual E_b/N_0 can be calculated from the average carrier power, N_0 and data-rate using the formula $E_b = C - 10 \cdot \log(\text{data-rate})$ where C is the carrier (signal) power in dBm. In version 6.85 and above, the CNG has a built-in function that will automatically determine the E_b/N_0 real time, not just the signal power. In this instance, the program can be simplified by simply by polling the E_b/N_0 at regular intervals and then averaging the readings. It may be tempting for the programmer to use the “Signal Track” and “Ratio Track” features solely for the purpose of simplifying the code sequence. However, in reality this is not the case. When either of these features is enabled, the CNG will automatically compensate for any drift by actively changing the signal or noise power respectively to match the drift. It will do so to the closest 1/10 dB, again corresponding to the attenuator resolution. Therefore if it were desired to take advantage of the 1/100 of a dB measurement precision, one would have to poll the CNG for the actual ratio and then average the ratios over the test duration. Because in either case the program is of similar complication, if the signal or ratio track isn't a test requirement, it is generally better to eliminate the “active compensation loop” that is internally processed in the CNG when Signal or Ratio Track is enabled. These features are mainly built in for certain test applications such as those that require a stable signal amplitude for the device under test, or when a change in

E_b/N_0 due to fluctuating amplitude may render the BER measurement statistically insignificant. These features are also a handy way to deal with changing signal powers when using the CNG in front panel operation with good accuracy and without applying offset factors externally.

Lastly if one wants to go the extra mile in accuracy, one can convert to linear from the logarithmic reading prior to averaging than then convert back to logarithmic. Avoiding this step would typically produce an offset of a few thousandths of a decibel so it may be overkill for most test requirements.

Appendix A is a code sequence written in C corresponding to the procedure outlined in the CNG Quickstart guide with the averaging and E_b/N_0 correcting algorithm described above.

APPENDIX A:

```
// CNG_EXP.C

/*****
*****
*      Sample Programming Code for control of Micronetics Carrier-to-
Noise Generator      *
*
*
*      This example configures the CNG for a typical Eb/No measurement
and then polls the CNG at roughly *
*      5 second intervals for carrier, noise and ratio values.  An
average Eb/No is then computed. *
*
*      CNGOS Version 5.85 or later
*
*      A CNG with CNGOS with earlier revision may not provide realtime
updates for Eb/No during the *
*      5 second polling intervals.  Average Eb/No can be determined
from the Carrier Level and Density *
*
*
*****
*****/

// include files
#include <stdlib.h>           // exit function
#include <stdio.h>           // printing functions
#include <conio.h>           // the _getch function
#include <time.h>            // functions for delay
#include <string.h>          // string functions
#include "windecl.h"        // National Instrument GPIB
Driver Functions

// constants & definitions
#define ACTIVATE_LOG      1           // set to 1 to print
communications to screen
#define PLAY_BY_PLAY      0           // set to 1 to pause after
each programming out/in pair

#define GPIB_CARD         0           // gpib card number
#define CNG_ADDR          1           // primary address of CNG
#define CNG_ADDR_SEC      0           // secondary address of CNG
#define CNG_TIMEOUT       T30s       // timeout for CNG -> 30
seconds
#define CNG_EOT           1           // assert EOI on last data
byte
#define CNG_EOS           0           // end-of-string termination
mode

// function prototypes
int dev_init();             // initialize a gpib
device session
```

```

int dev_out(int iDev, char *sOut);           // send a command string
out to a gpib device
int dev_in(int iDev, char *sIn);           // get a command string
back from a gpib device
void delay(long lngDlymsec);               // delay for a number of
milliseconds
double get_avg( double dData[], int iPts); // compute the average
of a data array

// global variables
int iCNG;                                  // handle for CNG gpib session

/*****
*****/
void main(void)
{
    int iPoll;
    char sInput[128];
    double dCarr[25];                       // Carrier Power Level readings
    double dBitEng[25];                     // Carrier Bit Energy readings
    double dDens[25];                       // Noise Density readings
    double dRatio[25];                     // Eb/No readings
    double dBitEngAvg;                      // Overall Average Bit Energy
    double dDensAvg;                        // Overall Average Noise Density
    double dSetRatio;                       // Initial Eb/No set by CNG

    // open communication path with CNG
    iCNG = dev_init();
    if( iCNG < 0 )
    {
        printf( "Error in communication with CNG\n" );
        printf( "Program is terminating!\n" );
        exit(1);
    }

    // query the device to ensure it is the CNG!
    dev_out( iCNG, "*IDN?" );
    dev_in( iCNG, sInput );

    // set System Impedance to 50 ohms
    dev_out( iCNG, "SYST:IMPED 50" );
    dev_in( iCNG, sInput );

    // turn OFF Signal Tracking
    dev_out( iCNG, "OPT:SIGTRACK OFF" );
    dev_in( iCNG, sInput );

    // turn OFF Ratio Tracking
    dev_out( iCNG, "OPT:TRACK OFF" );
    dev_in( iCNG, sInput );
}

```

```

        // Perform a full system calibration - power meter and noise
reference
        dev_out( iCNG, "SYST:CAL ALL" );
        delay( 4000 ); // Cal process does take time,
wait before querying completion
        dev_in( iCNG, sInput );

        // confirm that the CNG is now fully calibrated
        dev_out( iCNG, "SYST:CAL?" );
        dev_in( iCNG, sInput ); // response should be 'CAL: 2'
indicating fully cal-ed

        // lets perform an Eb/No set-up
        dev_out( iCNG, "MODE:RAT EBNO" );
        delay( 500 ); // Initial mode set-up takes a
little time
        dev_in( iCNG, sInput );

        // set Noise Density as our constant parameter
        dev_out( iCNG, "MODE:CONST NOISE" );
        dev_in( iCNG, sInput );

        // set out desired Center Frequency to 1.25 GHz
        dev_out( iCNG, "CARR:FREQ 1.25E09" );
        dev_in( iCNG, sInput );

        // set DUT BW to 21.56 MHz
        dev_out( iCNG, "NOISE:BW 21.56E06" );
        dev_in( iCNG, sInput );

        // set Noise Density (constant parameter) to -104.5 dBm/Hz
        dev_out( iCNG, "NOISE:DENS -104.5" );
        dev_in( iCNG, sInput );

        // set Carrier Signal Bit Rate to 9600 BPS
        dev_out( iCNG, "CARR:BITR 9600" );
        dev_in( iCNG, sInput );

        // set Eb/No ratio to desired level
        dev_out( iCNG, "RAT:SET 11.35" );
        delay( 500 ); // setting ratio takes a
little time
        dev_in( iCNG, sInput );

        // turn on the Noise at the C + N Output
        dev_out( iCNG, "NOISE:STAT ON" );
        dev_in( iCNG, sInput );
        delay( 2000 );

        // query the CNG for the actual Eb/No setting
        dev_out( iCNG, "RAT:SET?" );
        delay( 100 );
        dev_in( iCNG, sInput );
        dSetRatio = atof( sInput );

```

```

        // CNG now configured to perform desired measurement, setup to
poll for readings

        /* Signal Tracking is NOT enabled so if carrier level drifts CNG
will update its
            reading for output level but will not take any action to
correct.
        Ratio Tracking is NOT enabled so if carrier level drifts CNG
will only update
            the reading for Eb/No, but not try to maintain the initial
setting
        */
        for( iPoll = 0; iPoll < 25; iPoll++ )
        {
            // query the CNG for the actual settings of...
            printf( "Current CNG settings....\n" );
            // Carrier Output Level
            dev_out( iCNG, "CARR:POW?" );
            delay( 100 );
            dev_in( iCNG, sInput );
            dCarr[iPoll] = atof( sInput );

            // Carrier Bit Energy
            dev_out( iCNG, "CARR:BITENG?" );
            delay( 100 );
            dev_in( iCNG, sInput );
            dBitEng[iPoll] = atof( sInput );

            // Noise Density Level
            dev_out( iCNG, "NOISE:DENS?" );
            delay( 100 );
            dev_in( iCNG, sInput );
            dDens[iPoll] = atof( sInput );

            // Eb/No Level
            dev_out( iCNG, "RAT:SET?" );
            delay( 100 );
            dev_in( iCNG, sInput );
            dRatio[iPoll] = atof( sInput );

            delay( 4600 ); // wait remainder of 5 seconds
        }
    for next set of readings

        printf( "Average Carrier Output Level: %.3f dBm\n", get_avg(dCarr,
25) );
        dBitEngAvg = get_avg(dBitEng, 25);
        printf( "Average Bit Energy: %.3f dBm/Hz\n", dBitEngAvg );
        dDensAvg = get_avg(dDens, 25);
        printf( "Average Noise Density Level: %.3f dBm/Hz\n", dDensAvg );
        printf( "Reported Average Eb/No Ratio: %.3f dB\n", get_avg(dRatio,
25) );
        printf( "Computed Average Eb/No Ratio: %.3f dB\n", dBitEngAvg -
dDensAvg );
        printf( "Initial Reported Eb/No: %.3f\n", dSetRatio );

```

```

    printf( "Programming steps have completed.  Press ANY KEY to
end\n" );
    _getch();

    // Preset the CNG to default settings
    dev_out( iCNG, "*RST" );
    dev_in( iCNG, sInput );

    printf( "\n\n" );
} // end of Main

/*****
*****/
int dev_init(void)
{
    // open a gpib session for the device
    int iHandle;

    iHandle = ibdev(GPIB_CARD, CNG_ADDR, CNG_ADDR_SEC, CNG_TIMEOUT,
CNG_EOT, CNG_EOS);

    return iHandle;
}

int dev_out(int iDev, char *sOut)
{
    // output a string to the gpib device
    int iRtn = 0;
    long lngLen;

    lngLen = strlen( sOut );

    ibwrt( iDev, sOut, lngLen );

    if( (ibsta & ERR) != 0 )
        iRtn = -1;

    if( ACTIVATE_LOG )
        printf( "GPIB Out: %s\n", sOut );

    return iRtn;
}

int dev_in(int iDev, char *sIn)
{
    // input a string from the gpib device
    int iRtn = 0;
    char sResponse[128];

    memset( sResponse, 0, 128 );           // clear the response string

    ibrd( iDev, sResponse, 128L );
    if( ibcnt > 0 )
        strcpy( sIn, sResponse );

    if( (ibsta & ERR) != 0 )
        iRtn = -1;
}

```

```

    if( ACTIVATE_LOG )
        printf( "GPIB In: %s\n", sIn );

    if( PLAY_BY_PLAY )
    {
        printf( "Paused. Press ANY KEY to continue....\n" );
        _getch();
    }

    return iRtn;
}

void delay(long lngDlymsec)
{
    // perform a delay of lngDlymsec milliseconds
    clock_t goal;

    goal = lngDlymsec + clock();

    while( goal > clock() );
}

double get_avg( double dData[], int iPts)
{
    // compute the average of a data array
    int i;
    double dAvg = 0;

    for( i = 0; i < iPts; i++ )
        dAvg += dData[i];

    dAvg /= (double)iPts;

    return dAvg;
}

```

Available Models

Model	Frequency Range	Applications
CNG-055	1 to 10 MHz	Baseband
CNG70/140	50 to 90 MHz; 100 to 180 MHz	70 and 140 MHz Modem Testing, Satellite IF Loopback, HP3708A Replacement
CNG225	50 to 400 MHz	General purpose IF
CNG1600	950 to 2250 MHz	L-Band Modem and Satellite Loopback Testing
CNG70/140-L	50 to 90; 100 to 180; 850 to 2250 MHz	70/140, L-band Modem & Satellite IF Loopback Testing
CNG892/1850	822 to 962; 1710 to 1990 MHz	CDMA Mobile Phone and Base Station Receiver Testing
CNG1015	30 to 2000 MHz	General Purpose IF/RX
CNG2105	1710 to 2500 MHz	3G Mobile Telecom, CDMA Wireless Local Loop
CNG2442	2400 to 2484 MHz	ISM, Wifi, Bluetooth

Chart 1

All values in dBm

CNG70/140 and CNG225		Range Total Signal In	Range Ideal Signal In	Range Total Signal Out	Range Ideal Signal Out	In _{max} (No Damage)	Insertion Loss
notes:	Plain	-37 to 0	-30 to -7	-44 to -7	-37 to -37	+20	7 dB
<i>Output range @ 0 dB carrier Attenuation</i>	Opt001/002	-37 to 0	-30 to -7	-46 to -9	-39 to -39	+20	9 dB
	Opt003	-37 to 0	-30 to -7	-37 to -0	-30 to -7	+12	0 dB
	Opt004	n/a	n/a	n/a	n/a	n/a	n/a
	Opt004A	n/a	n/a	n/a	n/a	n/a	n/a
CNG892/1850, CNG1600 and CNG70/140-L, CNG1015		Range Total Signal In	Range Ideal Signal In	Range Total Signal Out	Range Ideal Signal Out	In _{max} (No Damage)	Insertion Loss
notes:	Plain	-37 to 0	-30 to -7	-46 to -9	-39 to -16	+20	9 dB
<i>Output range @ 0 dB carrier Attenuation</i>	Opt001/002	-37 to 0	-30 to -7	-48 to -11	-41 to -18	+20	11 dB
	Opt003	-37 to 0	-30 to -7	-37 to -0	-30 to -7	+12	0 dB
<i>Opt004/4A apply only to CNG892/1580</i>	Opt004	-67 to -30	-60 to -37	-76 to -39	-51 to -46	0	9 dB
	Opt004A	-67 to 0	-60 to -7	-76 to -9	-69 to -9	+12	9 dB
CNG2105 and CNG2442		Range Total Signal In	Range Ideal Signal In	Range Total Signal Out	Range Ideal Signal Out	In _{max} (No Damage)	Insertion Loss
notes:	Plain	-37 to 0	-30 to -7	-46 to -9	-39 to -16	+20	9 dB
<i>Output range @ 0 dB carrier Attenuation</i>	Opt001/002	-37 to 0	-30 to -7	-49 to -12	-42 to -19	+20	12 dB
	Opt003	-37 to 0	-30 to -7	-37 to -0	-30 to -7	+12	0 dB
	Opt004	-67 to -30	-60 to -37	-76 to -39	-51 to -46	0	9 dB
	Opt004A	-67 to 0	-60 to -7	-76 to -9	-69 to -9	+12	9 dB

Interpreting Chart 1:

The matrix above gives both the input and output ranges in dBm for each CNG model with each of the standard options. There are two sets of ranges, one is the ideal range for highest accuracy and the other is the total or functional range. The reason for the two sets is that the very complex signals such as 256 QAM, Faded CDMA base station receive signals, and 7/8 coded QPSK have a very high peak factor and large time constant. The built-in power meter can accurately measure these, but over a narrower dynamic range than less complex signals. The built-in power meter does have powerful signal processing and curve fitting algorithms to accurately measure complex signals. Micronetics chose these limited ideal ranges based on extensive data taken over many modulation schemes and data rates. We chose to be conservative because the CNG instruments are universal to any modulation type, the output corresponds to the input signal amplitude range less any insertion loss from input to output. If the carrier signal attenuator option is ordered, The output signal range corresponds to the 0 dB attenuation setting of the carrier attenuator. Obviously the overall output range is much greater corresponding to the range at the 0 dB setting plus 99 dB.