Simulation and Verification of Pulse Doppler Radar Systems

Application Note

Dingqing Lu, Agilent Technologies Inc., USA

Modern radar systems that operate in environments with strong clutter, noise and jamming require advanced digital signal processing techniques. Direct analysis techniques often fail when designing such complex systems. Although simulation is often used, most simulation tools do not have enough models and integration capability to handle modern radar systems.

Introduction

This application note proposes a solution to this dilemma, a system-design methodology that uses SystemVue. Examples will be used to illustrate how advanced Pulse Doppler (PD) surveillance radar with moving target detection (MTD) and a constant false alarm rate (CFAR) processor can be designed by using a platform in the SystemVue tool. To ensure the design works properly, the platform can be connected to instrumentation for system test and verification. This allows users to reduce their system development time and cost, while also decreasing their chances of unexpected system failures late in the system development process.

1.0 System Design Challenges

Advanced radar systems are very complex, necessitating sophisticated signal processing algorithms. Effective algorithm creation requires both a platform for simulation and for verification. Models for signal generation, transmission, antennas, T/R switching, clutter, noise, jamming, receiving, signal processing, and measurements are also needed to create advanced algorithms. Most simulation tools do not have enough models and the integration capability needed to design such complex algorithms. SystemVue provides an effective and efficient environment for algorithm creation.

Agilent Technologies

2.0 Signal Processing Algorithm Creation

The system-design methodology proposed below uses SystemVue. It concentrates on algorithm design for PD radar systems. The development of MTD and CFAR processors in a time-efficient manner are used as examples to better understand this methodology.

To begin, consider the tasks undertaken by the radar DSP algorithm designer, which typically breaks down into the following two stages:

Stage 1. Design the algorithm in software and verify it using a simulation tool.

To accomplish this task, the designer needs:

- A user-friendly algorithm modeling environment to easily create and debug the algorithm during algorithm development. The environment should support multiple languages, such as m-code*, C++ and HDL.
- Signal sources and measurements to verify the algorithm during verification, once it is created. Unfortunately, it is not a simple task for the DSP algorithm designer to create radar signal sources with radar cross section (RCS), clutters, noise, and jamming. The designer might also encounter difficulty creating measurements for the algorithm. Consequently, a tool that provides radar sources and measurements is desired.

Stage 2. Implement and test the algorithm in hardware.

After the algorithm testing and verification are finished, it needs to be implemented in hardware. To accomplish this task, a hardware test platform must be created.

To address these needs when creating the MTD and CFAR processor, a platform that provides the following functionality is required:

- An interface to a vector signal generator to generate test signals. The vector signal generator provides radar signal-generation test sources and models for RCS parameters, Clutter, Jamming, Doppler, and Frequency offset.
- An interface to a vector signal analyzer to verify the implemented hardware as compared to the original pure algorithm. The platform must support a wide range of measurements including waveforms, spectra, detection rate, and false alarm rate (FAR). Also it must provide an estimation of target distance, speed and angles for the detected target.

As will be shown, SystemVue provides a platform to meet all of these needs when creating radar algorithms.

*SystemVue has a built-in m-code compatible processor, called MathLang, which is available for use throughout the program. Existing m-code files can also be directly incorporated into a design. In addition, it has the capability of co-simulating with MATLAB™, from The MathWorks of Natick, MA.

3.0 SystemVue as a Platform for Simulation

The top-level system platform structure is shown in Figure 3-1. From the block diagram, the main models include signal source, transmitter, antenna, T/R switch, RCS, clutter, jammer, receiver, signal processors, and measurements. Sub libraries are also listed. The major task is how the designer creates their own algorithm using SystemVue.

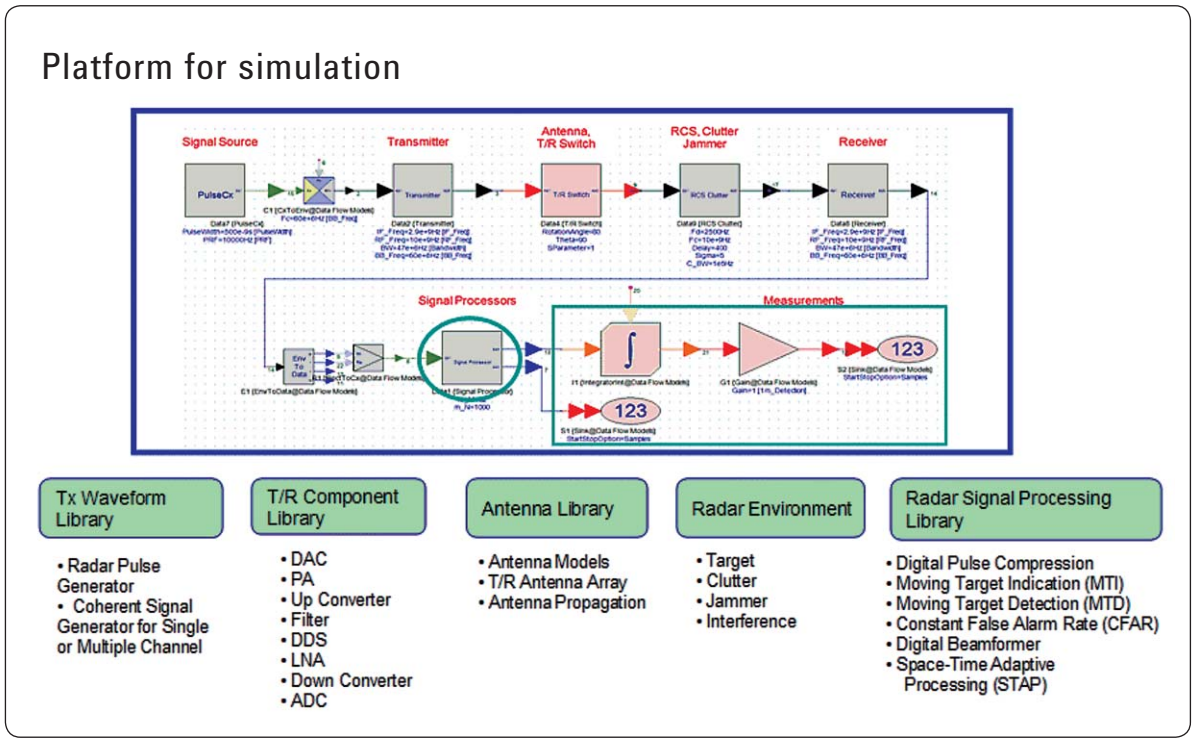

Figure 3-1. SystemVue as a platform for simulation

4.0 SystemVue as a Platform for Test

SystemVue can be used as a test platform for verification of the integrated system at each stage of development. This is done by connecting algorithms, instruments and test hardware together based on the created algorithm.

In SystemVue, an interface model (sink) allows for direct connections with various signal generators as shown in Figure 4-1. This allows software data to be downloaded to instruments for hardware test data. Figure 4-2 shows the details for using the platform to connect to instruments and test hardware based on the created algorithm.

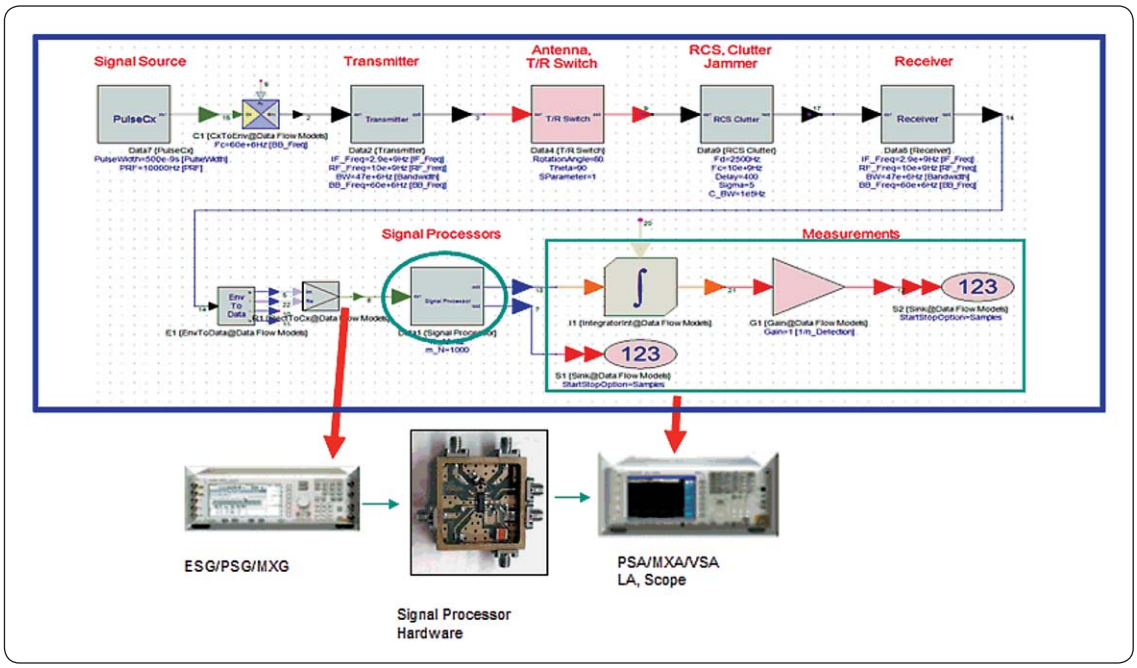

Figure 4-1. SystemVue as a platform for test

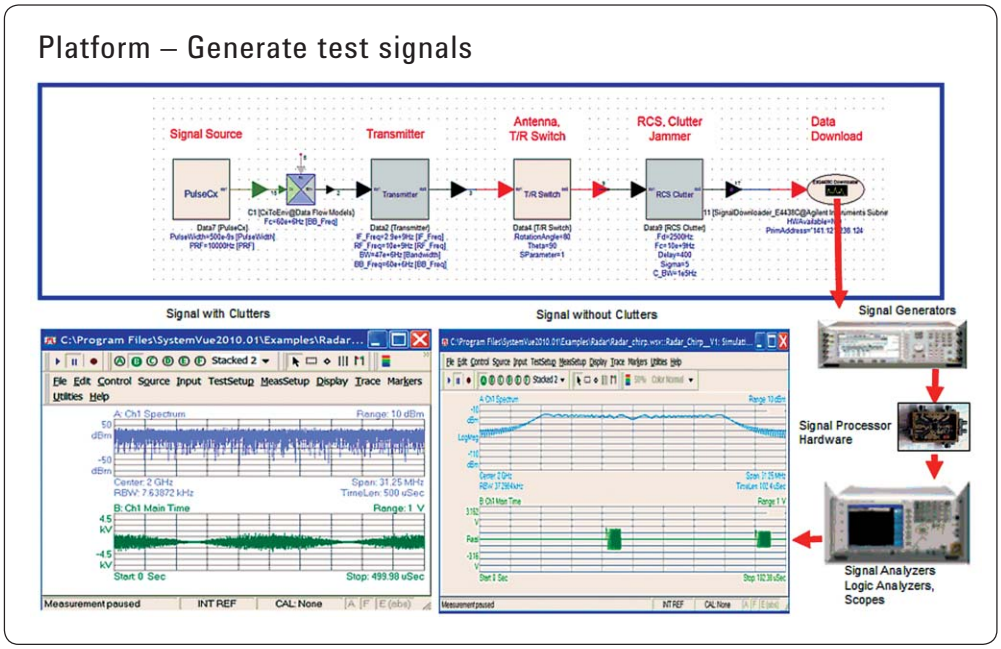

Figure 4-2. Generate test signals using SystemVue

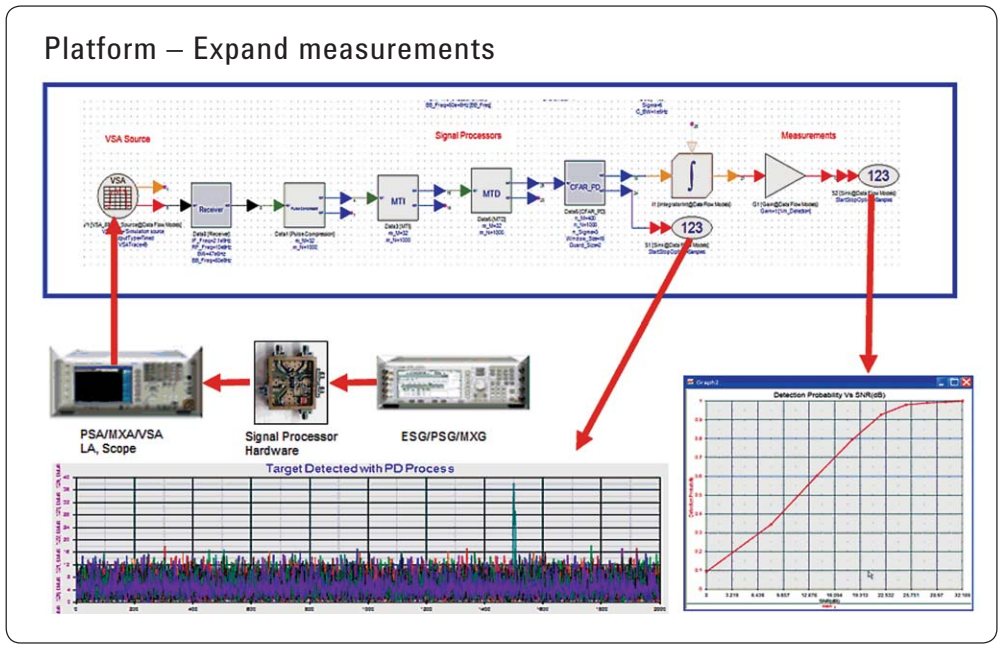

Figure 4-3. Expanding measurement capabilities using SystemVue

SystemVue can also connect to signal analyzers, logic analyzers or scopes to provide additional measurements and expand instrument capability according to the user's needs. As an example, Figure 4-3, shows the link between SystemVue and a signal analyzer. Here, test signals from the signal generator are sent to the device-under-test (DUT). The signal analyzer captures the DUT output waveforms and sends them to SystemVue, using the Vector Signal Analyzer (VSA) link model. In SystemVue, the waveform can be further processed using the radar signal processing function. SystemVue also provides additional meassurements such as Doppler frequency, detection probability and false alarm probability.

5.0 Principles of the PD Radar System

To better understand the proposed system design methodology, consider the design of PD radar algorithms. PD radars are extremely valuable for finding small moving targets hidden by heavily cluttered environments and are used in both military and commercial applications. Unlike continuous waveform (CW) radar, PD radar has the ability to detect angle, distance and also velocity. Typical examples include weather, low-flying aircraft and anti-ship missiles.

For this discussion, we focus on airborne PD radar that is trying to detect a moving target near the ground or the sea. In this case, the moving target returned signal is much weaker than the clutter from the ground or sea. As shown in Figure 5-1, the target signal is hidden by a heavily cluttered environment. Consequently, it is almost impossible to detect the target in the time domain using regular radar processing.

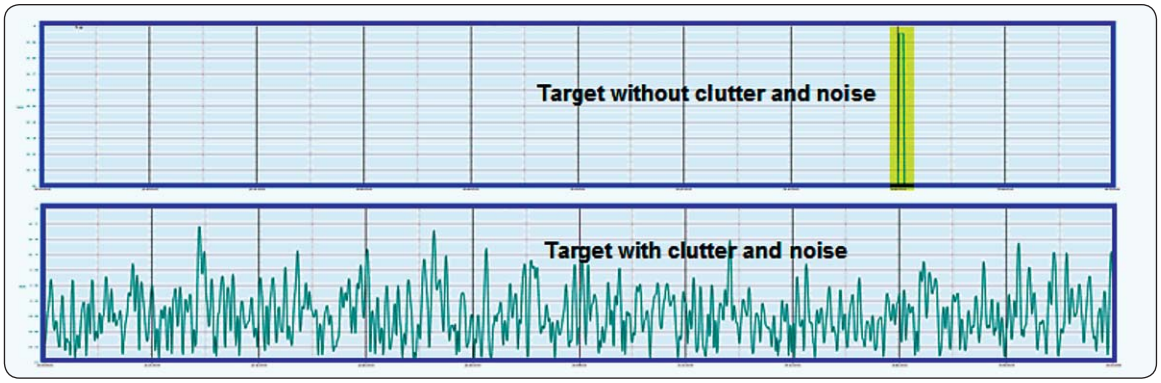

Figure 5-1. Target signal hidden by heavily cluttered environment

The radar transmission signal and returned signal [1] can be expressed as:

$$S(t) = A(t)Cos(2\pi j_c t)$$

$$S(t - \tau) = A(t)Cos(2\pi(f_c + f_d)t - \tau) + N_c(t) + N_n(t) + N_j(t)$$

where f_d is the Doppler frequency, τ is the delay, and Nc, Nn and Nj refer to clutter, noise and jamming, respectively. To detect target speed and distance, f_d and τ must be estimated.

Because the small moving targets are hidden by the heavily cluttered environments, they can not be detected in the time domain. Instead, the signal must be detected in the frequency domain using Doppler frequency analysis. To do this, return data must be collected and processed using two dimensional (2D) signal analysis for both target speed and distance. 2D signal processing is required for moving target indicator (MTI) and MTD. CFAR processing is needed for auto-detection in PD signal processing. Without CFAR, auto-detection will likely fail.

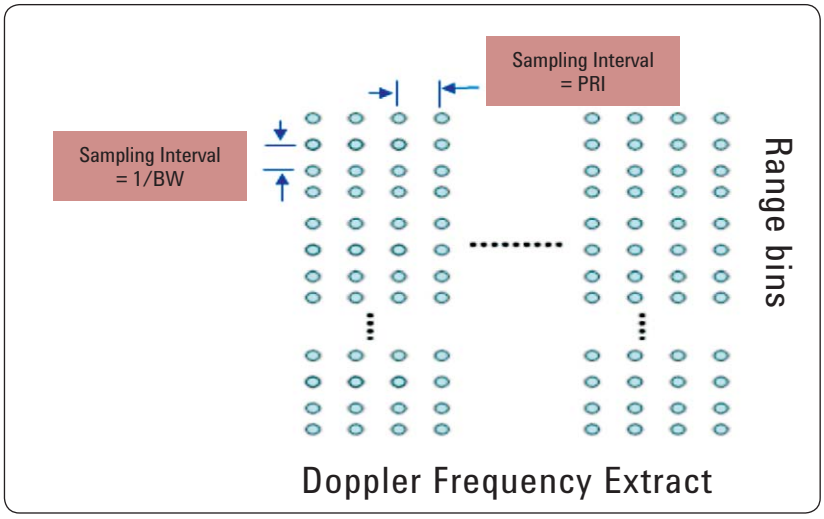

Figure 5-2. 2D signal processing for moving target detection

The first step in PD signal processor development is to design a data bank to store received timed signals, as shown in Figure 5-2. The received data is entered into the data bank point-by-point from one column to another until the bank is full.

Taking a closer look at each column, the time interval for each data point is 1/bandwidth. Each data point is a return signal from different distances. The sampling interval between each column is the pulse signal repetition interval. All data points in the same row are returned from the same distance with different timing. Doppler frequency can be extracted from data in the rows. Either a filter bank or a group of fast Fourier transform (FFT) operations can be used for all data points in the data bank. In software design, a group of FFTs is always used. Once the Doppler frequency is detected, the location of the row can be mapped to the return target distance from the range bins. Then, the distance can be detected.

6.0 PD Radar System Structure

Figure 6-1 depicts the top-level structure of a PD radar system in SystemVue. Since we are interested in creating an algorithm for PD signal processing, this discussion focuses on the PD signal processing, including MTD and CFAR. For other blocks, just a brief introduction is provided.

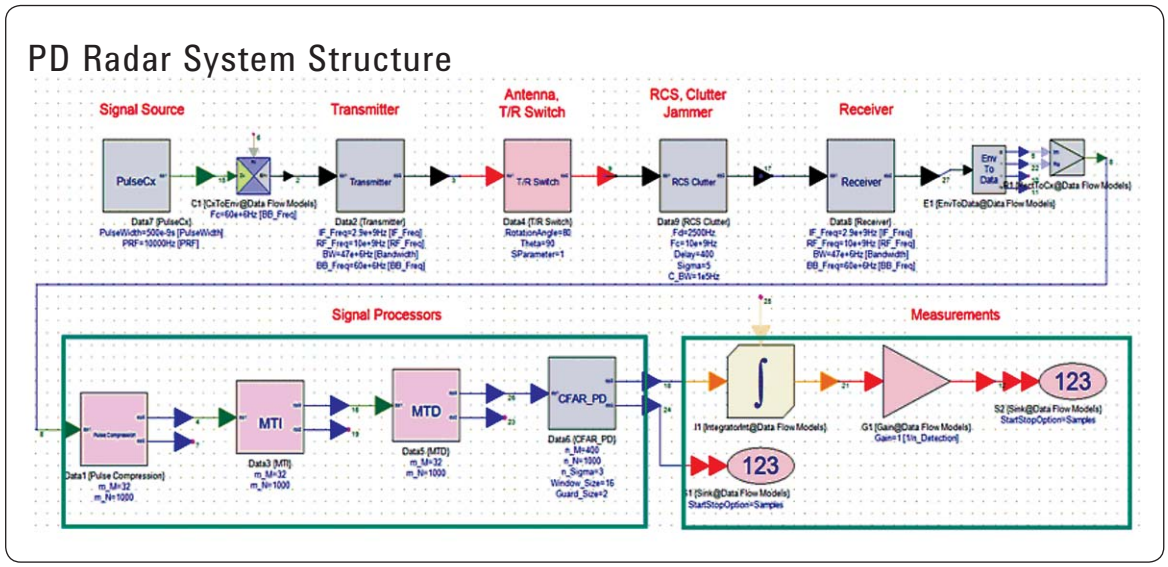

Figure 6-1. PD radar structure

6.1 Signal Sources

For this example, signal sources include:

- Pulse signal generator
- Linear FM pulse signal generator
- Nonlinear FM signal generator
- Polyphase code generator

6.2 Target Return Model

The target return model includes:

- RCS, Doppler effect, delay and attenuation
- Fluctuant RCS types: Swirling 0, I, II, III, and IV linear FM pulse signal generator

$$S_r(t - t_r) = kA(t - t_r)\cos[2\pi(f_0 \pm f_d)t - 4\pi R_0/\lambda + \varphi] \bullet U(t - t_r)$$

where R_0 is the distance between radar and the target, and t_r is the path delay, so $t_r = 2\mathrm{R}/\mathrm{c}$.

6.3 Clutter Models

Since this is the system-level simulation, behavioral models can be used to describe the functionality. We focus on the probability distribution functions (PDFs) and power spectrum densities (PSDs) that are good enough to model the system performance, but the physical-level model is not provided here. If required by the user, this service can be provided. SystemVue offers a choice of four PDFs and three PSDs, which include:

- PDF
  - Rayleigh
  - Log-Normal
  - Weibull
  - K–Distribution
- PSD
  - Gaussian
  - Cauchy
  - All Pole

The user can also define any distribution using SystemVue's built-in MathLang capabilities.

6.4 Radar RF Transmitters and Receivers

Here we provide the behavioral model's structure. If desired, the SpectraSys RF link enables the user to go down to the circuit level. Another way to model complex frequency dependent behavior is to import S-parameters using the SData model in SystemVue. An example of using the SData model is shown in Figure 6-5.

As shown in Figure 6-2, the RF transmitter features local oscillators, which can include phase noise, modulators and mixers with non-ideal behavior, and amplifiers which can include complex nonlinear behaviors and filters. Figure 6-3 depicts the RF receiver's oscillators, demodulator, amplifiers, and filters.

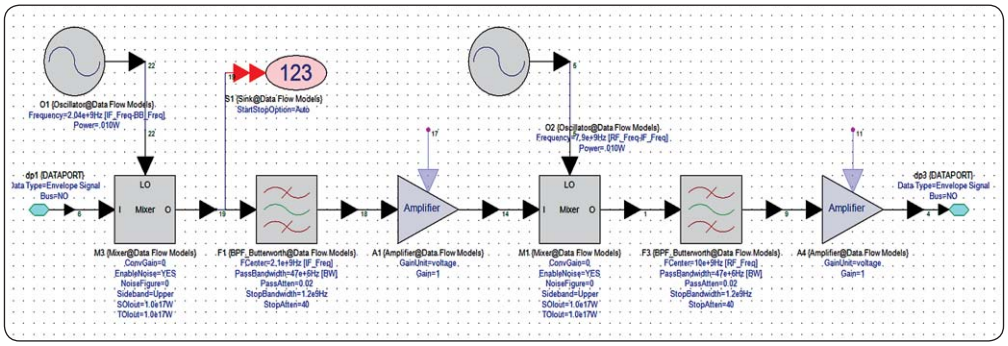

Figure 6-2. RF transmitter

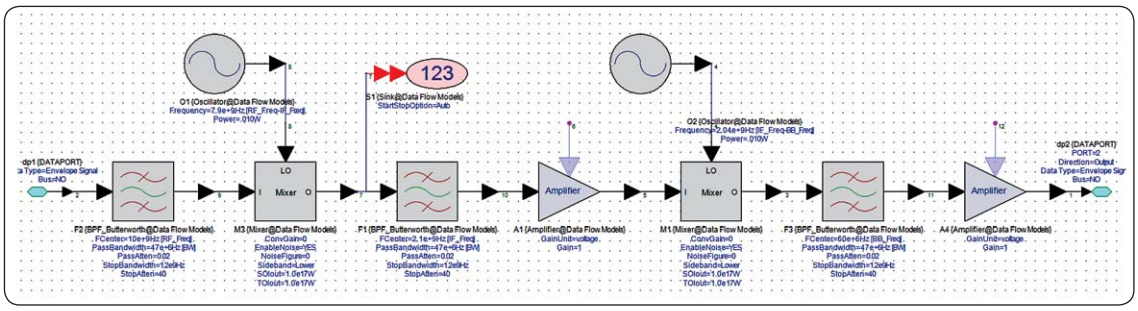

Figure 6-3. RF receiver

6.5 Digital Up/Down Converter (DUC/DDC) for Digital IF

The two models depicted in Figure 6-4 are very useful for creating new DSP models with certain algorithms.

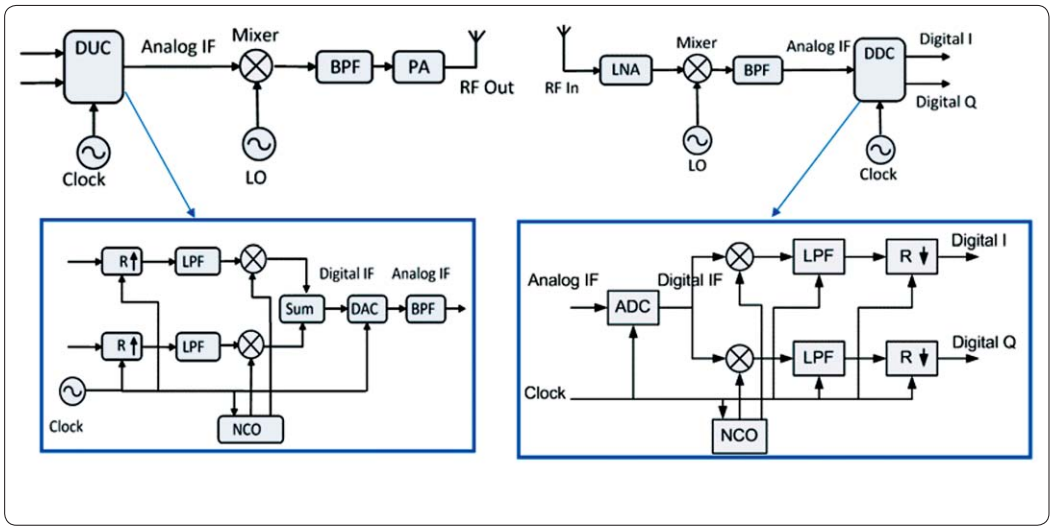

Figure 6-4. Digital up/down converters

6.6 Antenna Models

The antenna model structure shown in Figure 6-5 was created using a SystemVue SData model. If the user knows the antenna's Gain as a function of Deflection Angle, Modeled with S-parameters, the simulation model can be easily structured. Figure 6-5 also shows the antenna measurements.

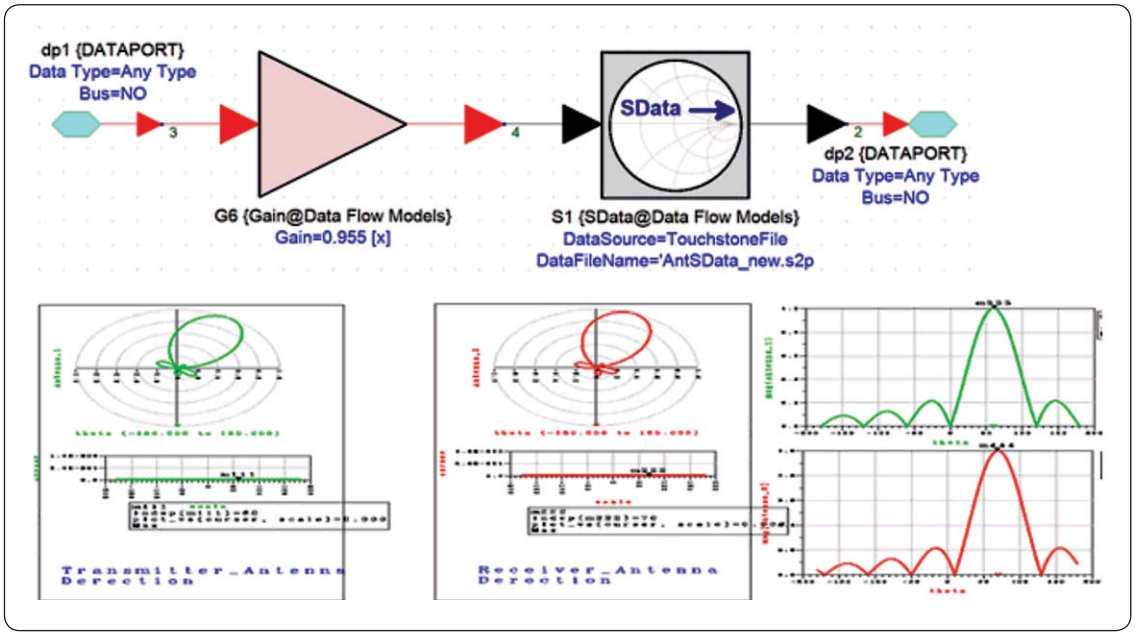

Figure 6-5. Antenna structure

6.7 Pulse Compression

Figure 6-6 depicts the PD processing pulse compression.

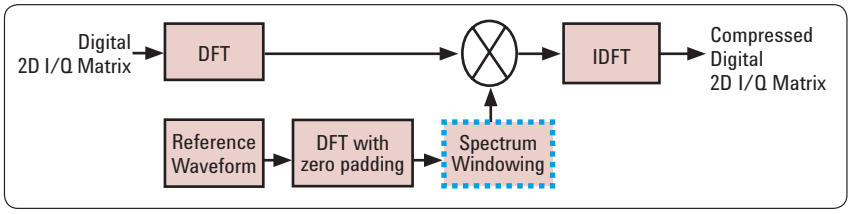

Figure 6-6. Pulse compression in frequency domain for PD processing

6.8 Moving Target Indicator

The basic idea behind the MTI is to filter the clutter at or very near DC, while keeping the other spectrum region flat. As shown in Figure 6-7, a three-pulse (double, second-order) canceller can be formed by cascading two first-order sections using a transfer function.

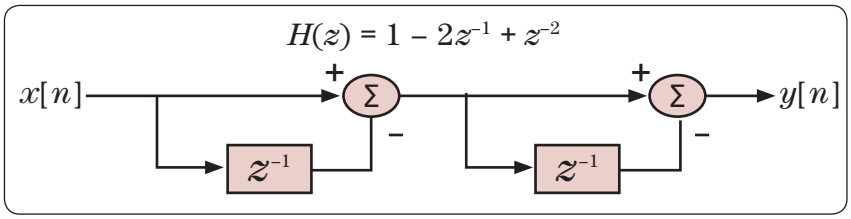

Figure 6-7. Moving target indicator

6.9 Moving Target Detector

The MTD is a key processor for PD radar [2]. A bank of Doppler filters or FFT operators cover all possible expected target Doppler shifts (Figure 6-8). The input data is collected in a repetition period by using a data bank. Data points within the same range are then correlated and processed until all data in the data bank is processed.

There are two ways to do the 2D signal processing, either using a filter bank or group of FFT. In the example, a group of FFTs is used to operate on each row in the data bank to detect f_d. Delay is then detected by looking at the detected data point location for range bins.

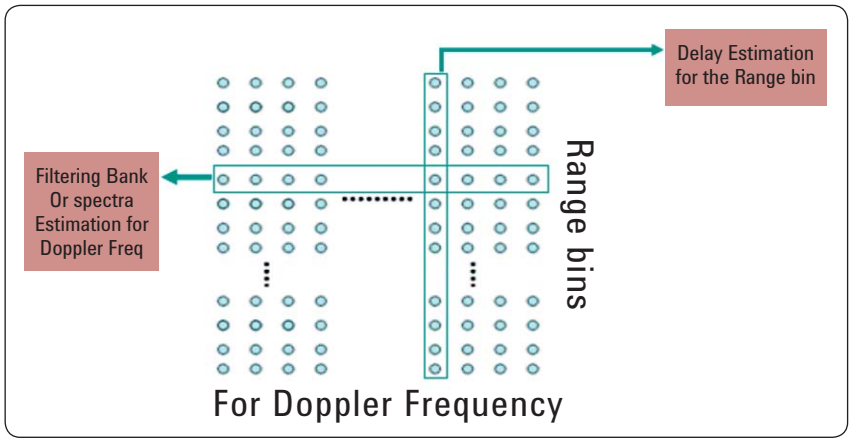

Figure 6-8. Signal Processing in MTD

Once the algorithm is understood, the code can be derived using C++ or MathLang. During its development, the code can be easily debugged for either C++ or MathLang as shown from Figure 6-9. Users can modify the code or insert their own in the code window to implement their own MTD algorithm.

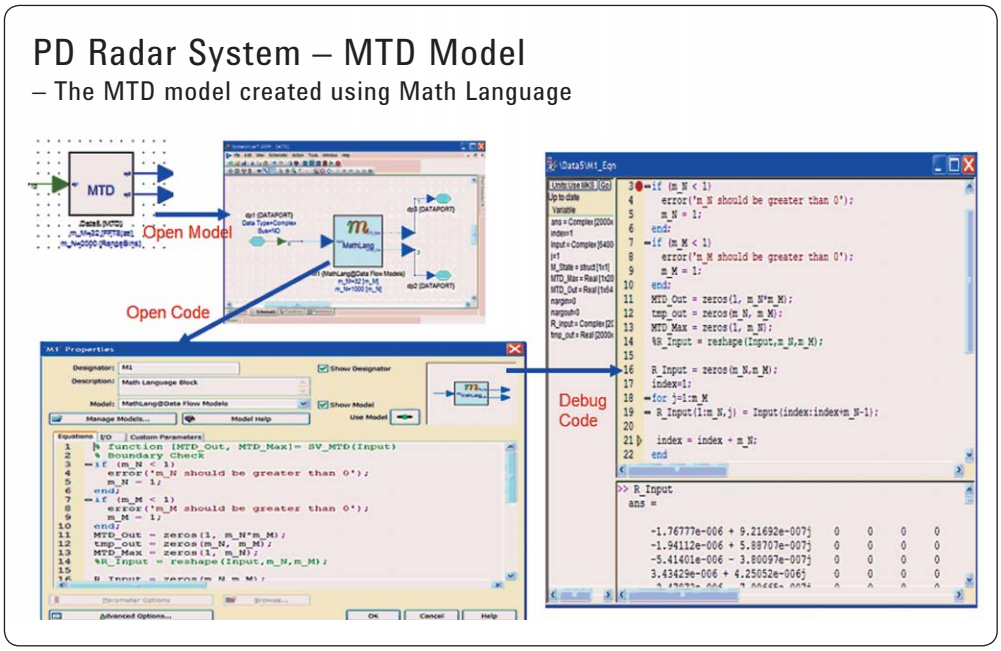

Figure 6-9. MTD implemented by using MathLang

6.10 CFAR Processor

Since modern radar requires auto detection, PD radar must use CFAR to control the false alarm rate. Otherwise, the radar won't work. The CFAR can be done in time or frequency domains or both. Instead of the fixed detection threshold, the averaging amplitude value of reference cells is used as the threshold in order to prevent false alarms from happening too frequently (Figure 6-10). This CFAR system is called a cell averaging (CA) CFAR system.

In the PD radar example in this application note, CFAR was done in the frequency domain. Cell averaging CFAR was used.

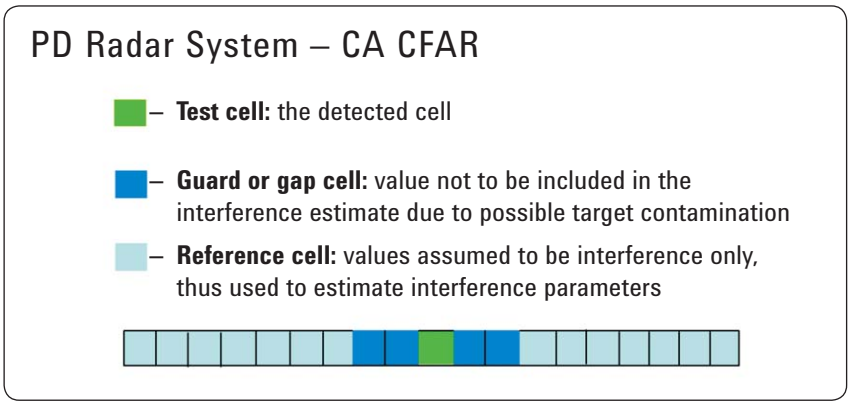

Figure 6-11. CA CFAR processor code implementation

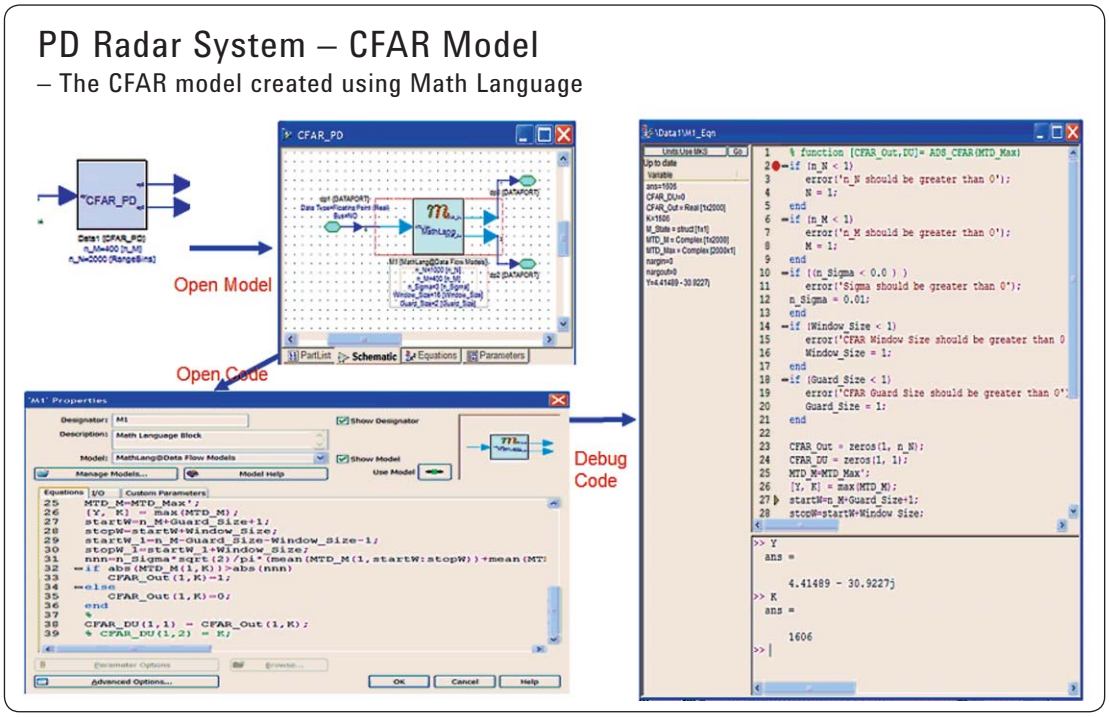

Figure 6-10. Cell averaging CFAR processor

Once the algorithm is understood, the code can be derived using C++ or MathLang. During its development, the code can be easily debugged for either C++ or MathLang as shown from Figure 6-11. Users can modify the code or insert their own in the code window to implement their own CFAR algorithm.

6.11 Measurements

Models have been implemented to do the following measurements:

- Basic Measurements
  - Waveform
  - Spectrum
  - Signal noise ratio
- Advanced Measurements
  - Estimation of distances and speeds
  - Detection probability, Pd = number of successful detection/total number of tests
  - False alarm probability, Pf = number of false errors/ total number of tests
  - Importance sampling will be implemented to speed up the Pf simulation [3]

If the user wants more, custom measurement models can be created using a combination of existing models.

7.0 Simulation of PD Radar System

The PD design shown in Figure 6-1 is simulated. All key parameters can be set at the Parameter table defined for the PD simulation system as shown in Figure 7-1. The user can very easily edit, add or delete any parameter.

7.1 Simulation Results

Before PD processing, at the receiver input, the target signal cannot be recognized and the target is hidden by strong clutter environment (Figure 7-2). However, after the 2D PD processing, the target is detected. After PD and CFAR, clean target detection is achieved.

PD_Radar

Name	Description	Default Value	Units	Tune	Show	Initially Use Default
PulseWidth	Pulse Width	500e-9	s	☐	☑	☐
PRF	Pulse Repeat Frequency	10000	Hz	☐	☑	☐
Fd	Doppler Frequncy	8125	Hz	☐	☑	☐
Range	Target Range	12000	M	☐	☑	☐
RF_Freq	RF Frequency	10e9	Hz	☐	☑	☐
IF_Freq	IF Frequency	2.9e9	Hz	☐	☑	☐
BB_Freq	Baseband Frequency	60e6	Hz	☐	☑	☐
Bandwidth	Band Width	47e6	Hz	☐	☑	☐
SamplingFreq	Simulation Sampling Frequency	120e6	Hz	☐	☑	☐
FFTSize	FFT Size	32	( )	☐	☑	☐
n_M	Detection Cell	400	( )	☐	☑	☐
Window_Size	CFAR Window Size	32	( )	☐	☑	☐
Guard_Size	CFAR Guard Size	6	( )	☐	☑	☐
Sigma	Clutter Standard Diviation	2.5	( )	☐	☑	☐
C_BW	Clutter Bandwidth	1e6	Hz	☐	☑	☐
RotationAngle	Antenna Rotation Angle	60	deg	☐	☑	☐
Theta	Antenna Angle	90	deg	☐	☑	☐
n_Sigma	Noise Standard Diviation	3	( )	☐	☑	☐
n_Detection	Number of statistics for the detec	1	( )	☐	☑	☐

Figure 7-1. Simulation setup table

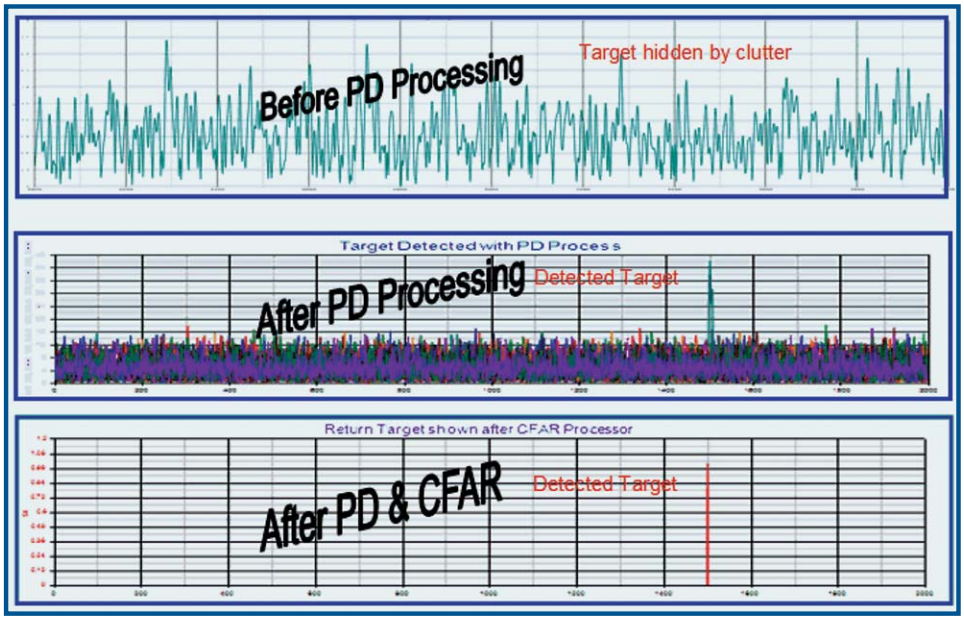

Figure 7-2. PD detection of a moving target

In Figure 7-3, the detection probability of the system is displayed. The detection rate is obtained by running a number of tests and coming up with the detect rate using the Pd definition. The target distance and speed are estimated using the detected Doppler frequency and the detected range bin location.

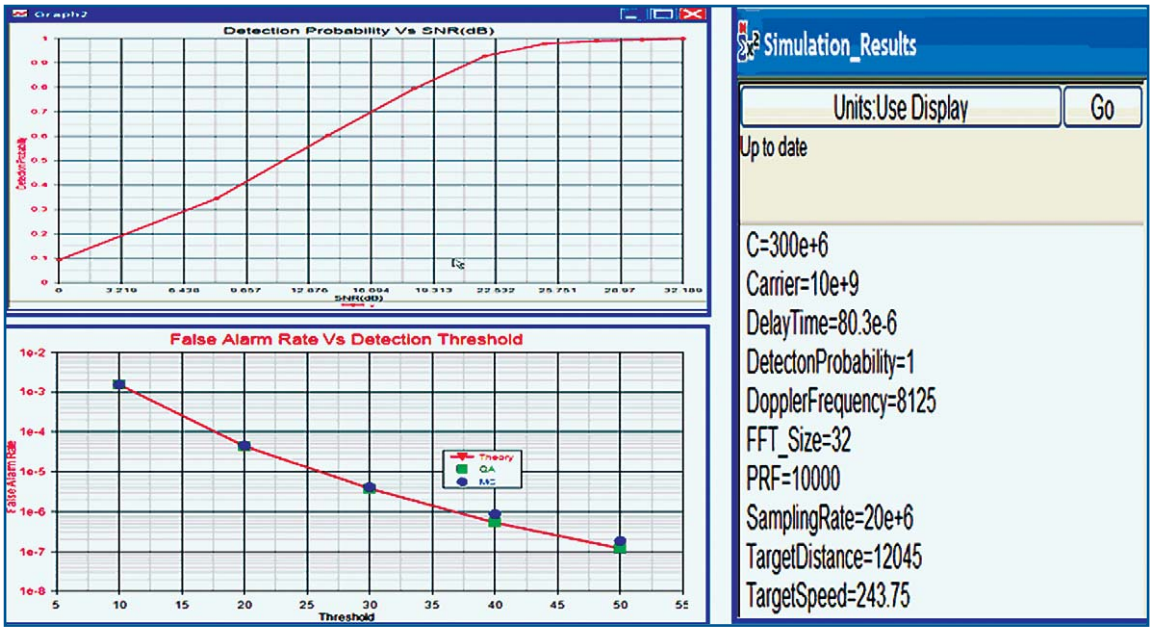

Figure 7-3. System detection rate and estimated target distance and speed

8.0 Summary

Algorithms are critical for high-performing advanced radar systems. A unified approach to radar system design that relies on SystemVue now offers designers a viable means of creating effective algorithms. It provides a user-friendly environment for algorithm development, while also integrating software and hardware to verify the algorithms. SystemVue can even be used to develop algorithms for digital array radar plus space-time adaptive processing (STAP) and multiple-input multiple-output (MIMO) radar. Moreover, it allows the system development team to quickly and easily try new and innovative ideas and to evaluate the effects of jamming and interference sources on radar performance.

9.0 References

1. I. Skolnik, *Radar Handbook*, 2nd ed. McGraw-Hill Inc., 1990.
2. D. Curtis Schleher, *MTI and Pulse Doppler Radar*, Artech House, Inc., 1991.
3. Dingqing Lu and Kong Yao, *Importance Sampling Simulation Techniques Applied to Estimating False Alarm Probabilities*, Proc. IEEE ISCAS, 1989, pp. 598-601.

Additional radar resources from Agilent:

- **www.agilent.com/find/radar**
- SystemVue home page: **www.agilent.com/find/eesof-systemvue**
- SystemVue videos: **www.agilent.com/find/eesof-systemvue-videos**

www.agilent.com

www.agilent.com/find/eesof-systemvue

For more information on Agilent Technologies' products, applications or services, please contact your local Agilent office. The complete list is available at:

www.agilent.com/find/contactus

Americas

Canada	(877) 894 4414
Latin America	305 269 7500
United States	(800) 829 4444

Asia Pacific

Australia	1 800 629 485
China	800 810 0189
Hong Kong	800 938 693
India	1 800 112 929
Japan	0120 (421) 345
Korea	080 769 0800
Malaysia	1 800 888 848
Singapore	1 800 375 8100
Taiwan	0800 047 866
Thailand	1 800 226 008

Europe & Middle East

Austria	43 (0) 1 360 277 1571
Belgium	32 (0) 2 404 93 40
Denmark	45 70 13 15 15
Finland	358 (0) 10 855 2100
France	0825 010 700*
	*0.125 €/minute
Germany	49 (0) 7031 464 6333
Ireland	1890 924 204
Israel	972-3-9288-504/544
Italy	39 02 92 60 8484
Netherlands	31 (0) 20 547 2111
Spain	34 (91) 631 3300
Sweden	0200-88 22 55
Switzerland	0800 80 53 53
United Kingdom	44 (0) 118 9276201

Other European Countries:
www.agilent.com/find/contactus

Revised: October 1, 2009

Product specifications and descriptions in this document subject to change without notice.

Printed in USA, April 12, 2010
5990-5392EN

Agilent Technologies