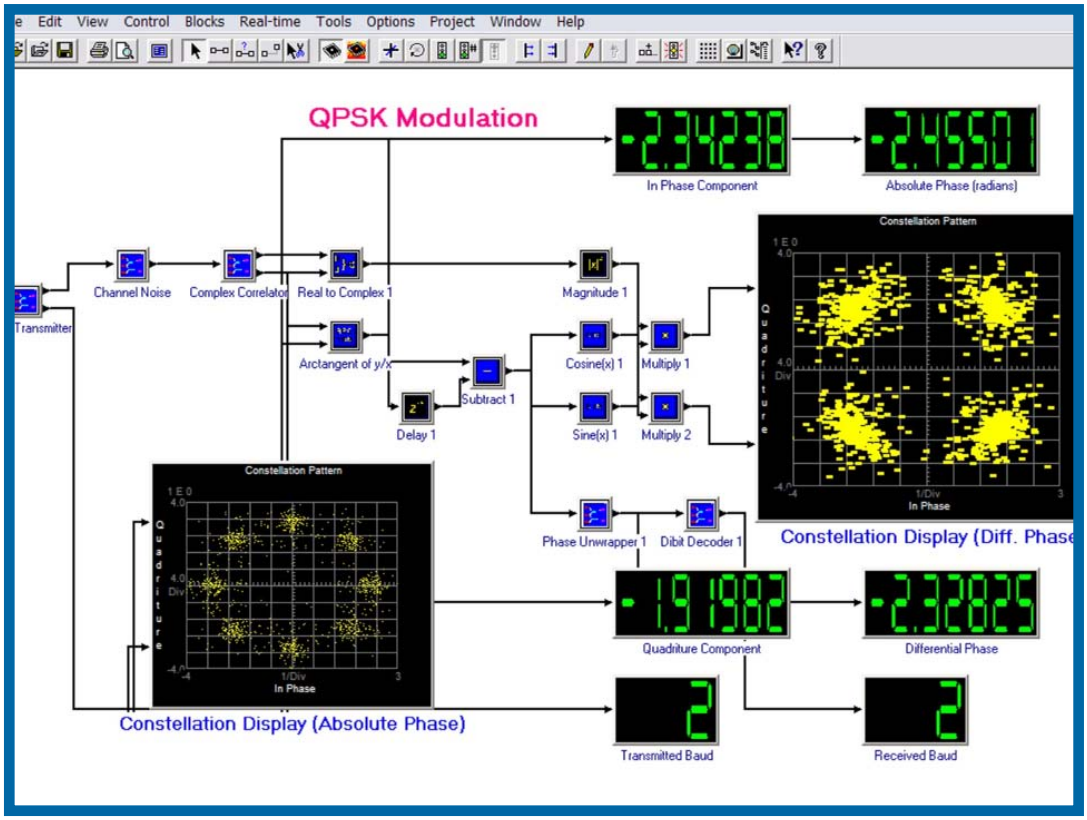


RIDE®

Graphical Programming for DSP

Document: HSMK8000

Real-time Integrated Development Environment for Digital Signal Processing (DSP)

- *Faster DSP code development*
- *Rapid prototyping of products and systems*
- *Scaleable platform with integration to leading DSP design tools*
- *Direct support for DSP development hardware*
- *Real-world integration to test with virtual instrumentation*

hyperception.com/RIDE

HYPERCEPTION
A NATIONAL INSTRUMENTS COMPANY

The RIDE Environment

Use a Better Tool for DSP Development

RIDE is a graphical programming environment that supports the design, implementation, and analysis of real-time DSP-based algorithms and systems. Developing with RIDE is simple - just create a block diagram of your DSP application. Your block diagram is your DSP code and can run on the DSP without need of any additional compilers. Block functions are selected from a menu and connected to establish data flow. Run-time parameters are adjusted from pop-up dialog boxes, and the entire algorithm is then executed on the DSP with the press of a single button. For supported target hardware RIDE even programs on-board FLASH memory directly from the graphical design. Thoughout development RIDE provides complete control and observation of DSP designs in an interactive fashion.

RIDE Simplifies Development

RIDE allows you to focus more on your ideas, and less on the numerous low-level details involved in implementing DSP code. Design/development iteration cycles are much faster and allow for more efficient overall design.

Rapid Prototyping

Use RIDE to quickly develop a complete DSP application by leveraging the extensive library of built-in functions. Have your application running in minutes on leading DSP evaluation hardware. You can change algorithm parameters on-the-fly, or view data results using a library of graphical controls and indicators.

Open Software Environment

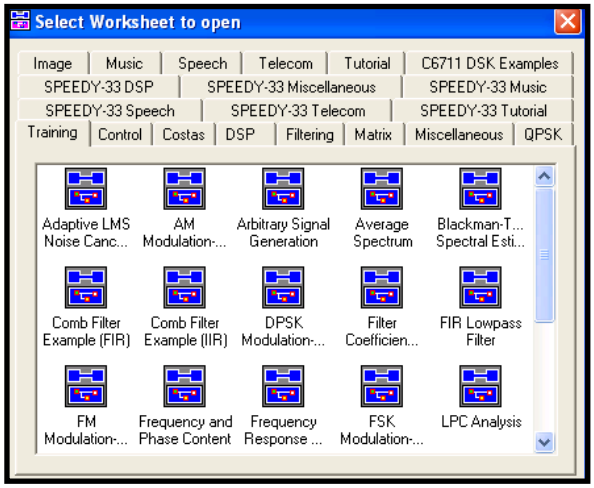
RIDE is based upon an open software architecture. In addition to the hundreds of included block functions, you can build your own functions from existing blocks using hierarchy or build blocks from your own C or Assembly source using the Block Wizard. RIDE is quite extensible, so customizing to your application is never a problem.

Interoperate with Standard DSP Tools

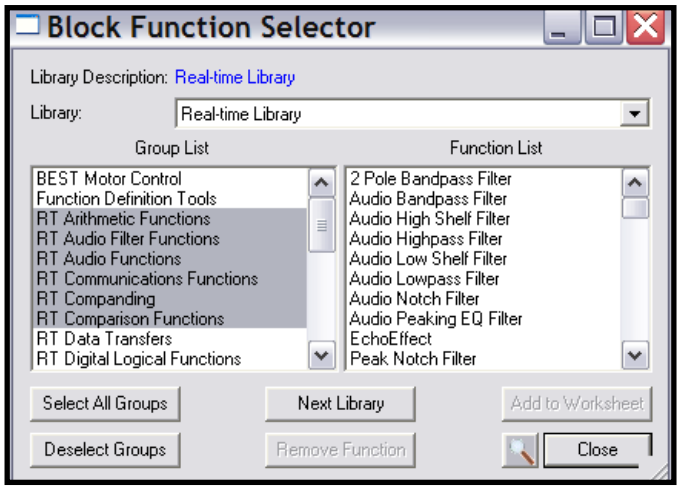
While RIDE does not require other development tools to create DSP applications, interoperability with conventional DSP tools can play an important role for advanced DSP developers. RIDE supports DSP tools such as Texas Instruments Code Composer Studio™ (CCS) and Analog Device's VisualDSP++™.

Getting Started

RIDE includes many reference block diagram design templates to assist you with development of DSP applications. These time-saving templates for applications such as telecom, speech, control, communications, and other DSP areas shorten the development process and serve as a starting-point for your own DSP-based application.



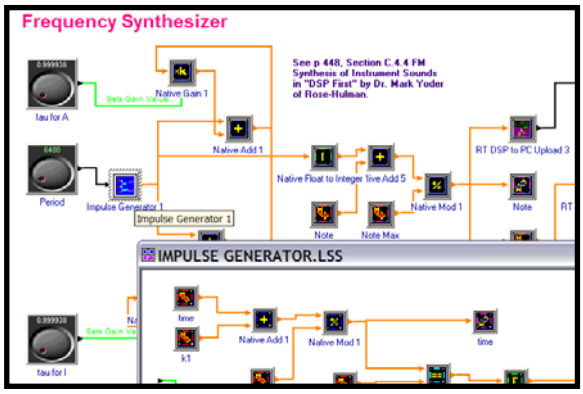
Reference Templates – Choose from a variety of relevant templates and examples in many engineering areas.



Block Selector – Search from hundreds of block functions included with RIDE.

Creating Your Application

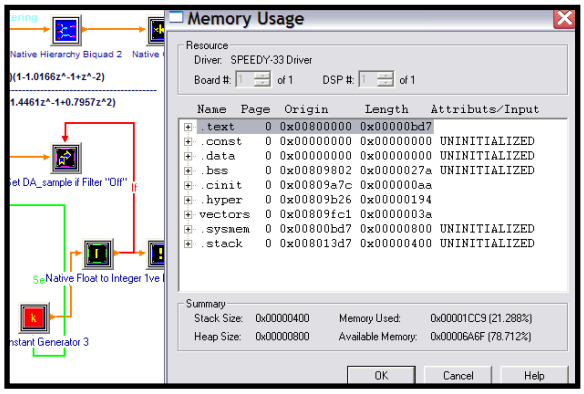
RIDE simplifies DSP code development. To develop your application select block functions from a pull-down menu and connect with the data flow wiring tool. The blocks are linked and downloaded on your DSP platform for immediate execution.



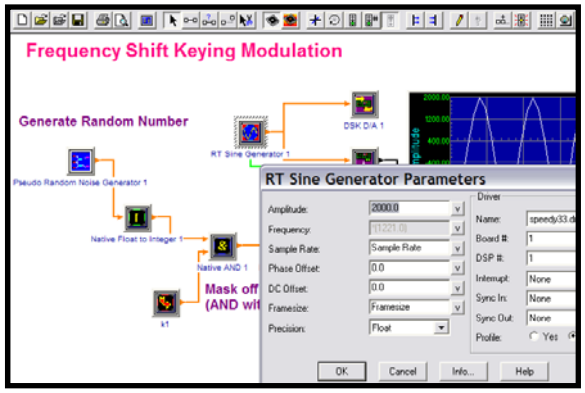
Hierarchy – Create new components from built-in blocks or your own blocks - true N-level hierarchy.

Debugging Your Application

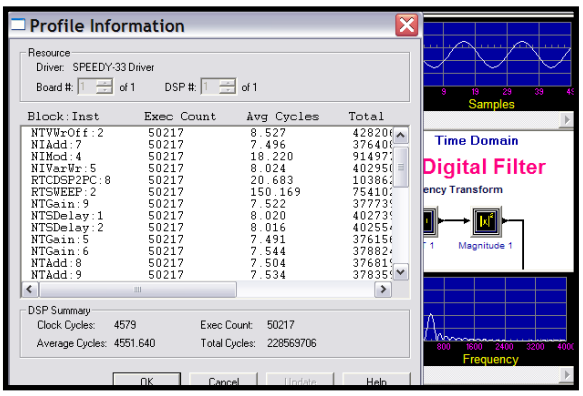
RIDE features built-in debugging tools to easily verify and debug your DSP application.



Resources - View a system summary of DSP resources.



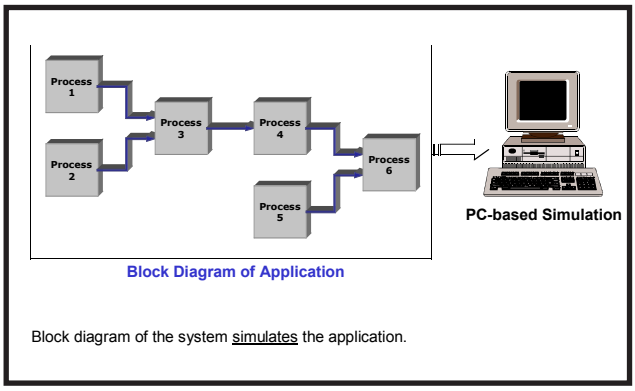
Targeting Your Diagram - Adjust runtime parameters from pop-up dialog boxes, then press the compile icon.



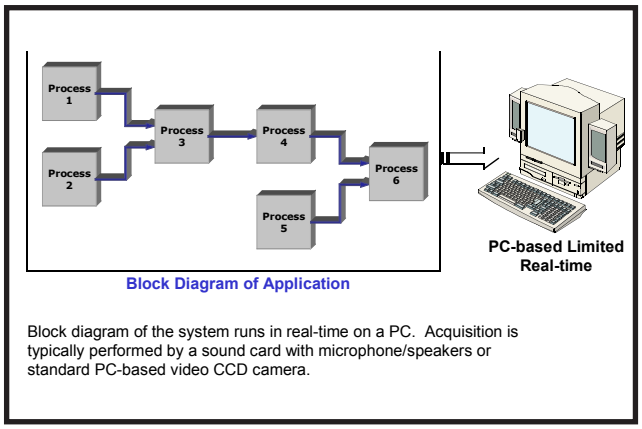
Code Profiling – Monitor algorithm execution time for the DSP using the block profile function.

The RIDE Environment

Simulate and perform limited real-time DSP on your PC

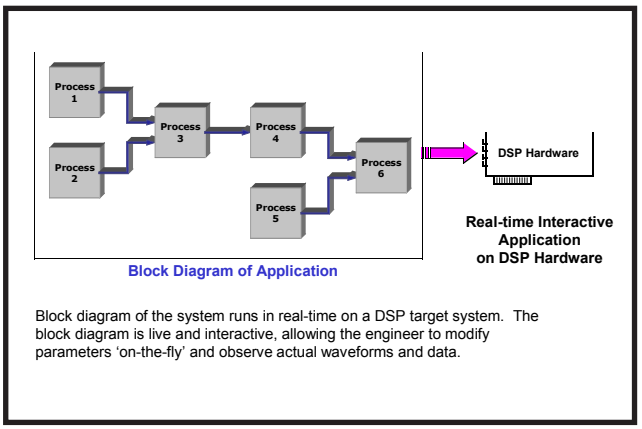


Simulate and Model - Use the PC to simulate or model your DSP design

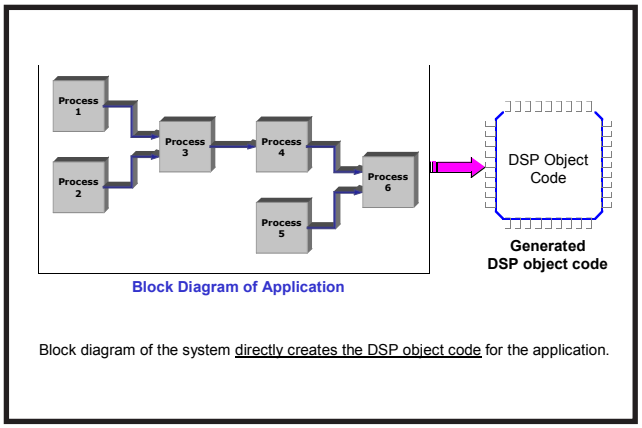


PC based Real-time Development - Use your PC sound card for limited real-time DSP processing and algorithm development

Program your entire DSP application graphically

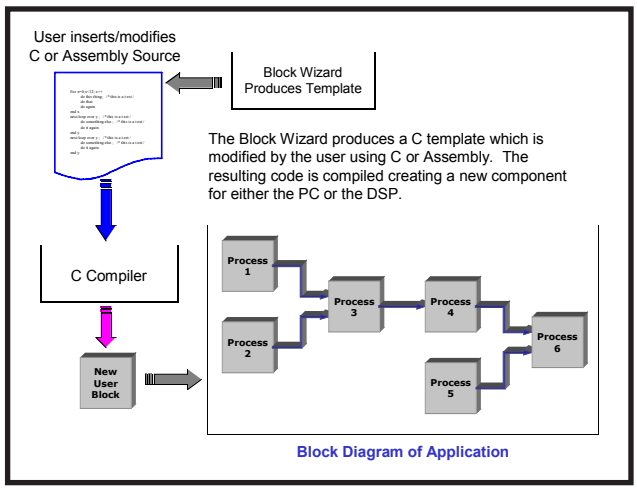


Direct Programming of Off-the-shelf DSP Hardware - Use standard low-cost DSP development boards for rapid prototyping of your DSP application.



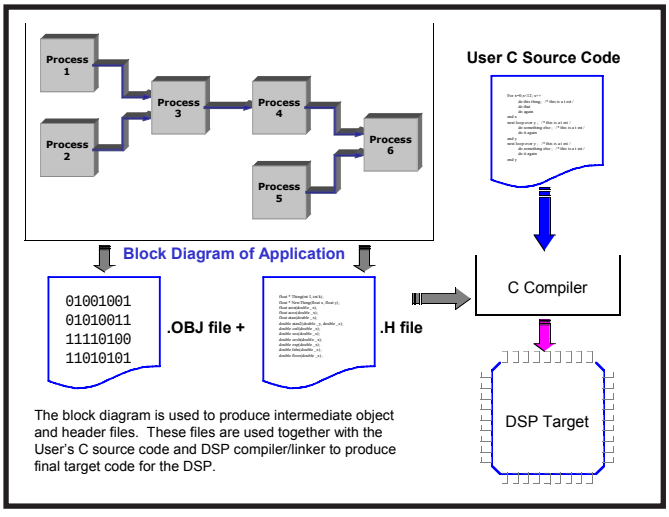
Direct Programming of Embedded DSP Hardware - Use RIDE to create your entire DSP application's object code for use with your own embedded DSP target hardware.

Create new blocks using C or assembly

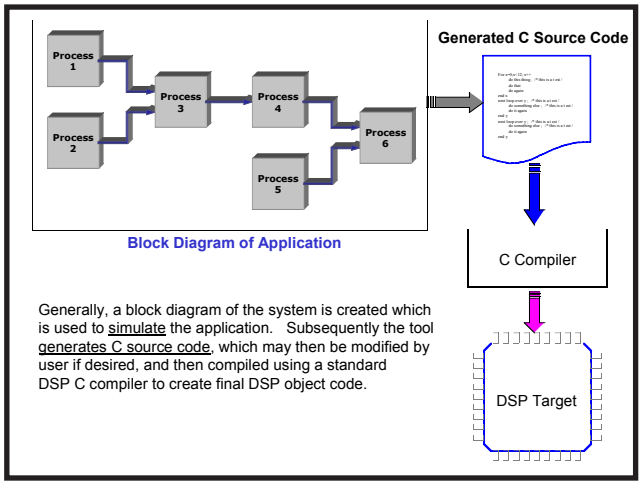


Open Platform - Add new blocks using standard C or Assembly source code.

Complement your conventional C or assembly development process



Programming Support for DSP Applications in C - Create part of your application graphically, then export the .H and .OBJ files to use with conventional DSP development tools.

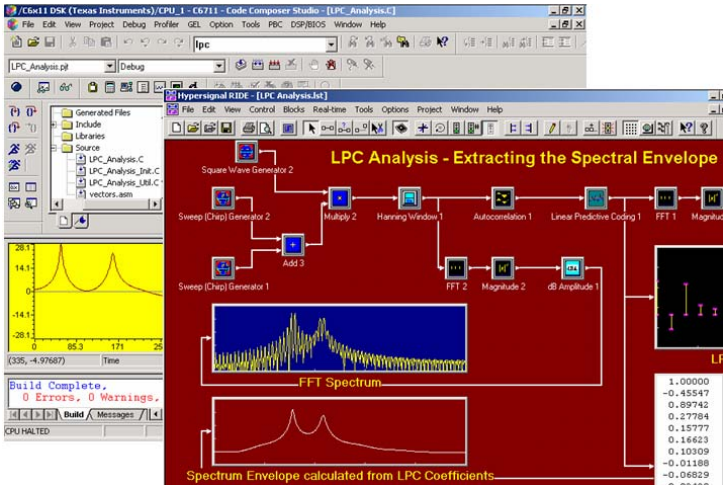


C Source Code Generation for DSP Hardware - Automatically generate ANSI C source code from graphical block diagrams for use with conventional DSP compilers.

Target Embedded DSP Applications

Device Independence

Most DSP applications can be broken into three parts: interfacing with real-world data; processing data to extract or analyze meaningful information; and, presenting the information back in a form expected by a user.



Connectivity - Real-time interface to Texas Instruments and Analog Devices evaluation hardware and software development tools.

RIDE is an open software environment designed to simplify interfacing with target hardware. With interactive, real-time communication to DSP evaluation hardware platforms from Texas Instruments and Analog Devices, RIDE makes algorithm coding simple and fast.

Complete block function components for analog-to-digital converters and evaluation hardware codecs are included in RIDE. A powerful feature of RIDE is its device independence; after designing your block diagram in the RIDE environment using standard blocks, you can target many DSPs.

RIDE supports high-speed real-time communication to your DSP target code, without halting the DSP while moving data in most applications. With RIDE graphical controls, indicators and powerful memory commands you can easily view and debug application code.

Target many DSP Processors directly with RIDE - algorithms can be quickly constructed and immediately executed on a number of leading DSP's. The following popular DSP families are supported:

- Texas Instruments C2000™, C3x™, C5000™ and C6000™ families
- Analog Devices SHARC™ and Blackfin™ families

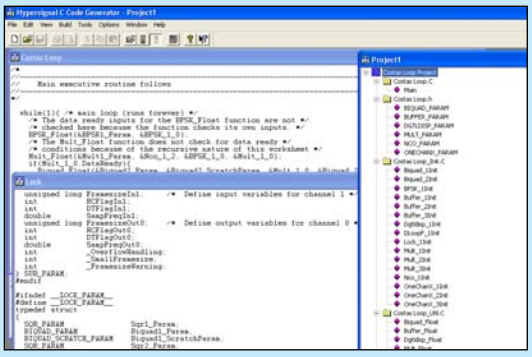
Interoperability With Leading Design Tools

Conventional development tools play an important role for advanced DSP developers. RIDE operates seamlessly with textual-based tools such as Texas Instruments' Code Composer Studio (CCS), and Analog Device's VisualDSP++.

- Link with RIDE executable DSP object code directly by using the companion environments.
- Export a portion of your RIDE diagram as a C callable function* to use with conventional C compilers.
- Easily move your C or Assembly routines into RIDE using the included Block Wizard.
- Protect your algorithm intellectual property (IP) and use RIDE as a platform for customer evaluation.

Automated Source Code Generation

Advanced users select the C Source Code Generator to use RIDE in support of traditional C development for virtually any DSP. Modular, commented source code provides for fast project turnaround and sharing of code.



* Currently Texas Instruments C6000 family only

Partial List of Block Functions

Arithmetic Functions

Absolute Value - *Outputs the absolute value of the input signal*

Add - *Adds two input signals*

Complex Conjugate - *Takes the complex conjugate of the input signal*

Complex to Real - *Converts the complex input values to the corresponding real and imaginary values*

dBAmplitude - *Performs the function of 20*log10(x)*

dBPower - *Perfoms the function of 10*log10(x)*

Differentiate - *Differentiates the input signal*

Divide - *Divides one signal by another*

Exponential - *Calculates the exponential function of the input signal*

Five Input Add - *Adds the corresponding elements of five input signals*

Four Input Add - *Adds the corresponding elements of four input signals*

Integrate - *Integrates the input signal*

Log - *Calculates the natural logarithm (base e) value of each element of the input signal*

Log 10 - *Calculates the common logarithm (base 10) value of each element of the input signal*

Logx(y) - *Calculates the logarithm (base x) value of each element of the input signal*

MOD (Modulo) - *Divides the input frame of data with the Modulo value*

Multiply - *Multiplies the corresponding elements of two input signals*

Polar to Rectangular - *Converts polar to rectangular*

Power - *Calculates the user specified exponent (power) value of the input signal*

Product - *Outputs the product of all input frame samples*

Real to Complex - *Converts real and imaginary values into the corresponding complex values*

Reciprocal - *Calculates the 1/x value for a data sample x*

Rectangular to Polar - *Converts rectangular to polar*

Square - *Calculates the square of the input signal*

Square Root - *Calculates the square root of the input signal*

Subtract - *Subtracts the corresponding elements of channel 1 input from channel 0 input*

Sum - *Outputs the sum of the input data frame values*

Three Input Add - *Adds the corresponding elements of three input signals*

Bit Conversion Functions

Bit Mask - *Masks off user-specified bits in the input signal*

Ones Complement - *Performs ones complement bit conversion on each element of the input signal*

Pack - *Packs 8, 4, 2,or 1-bit data to become 16-bit integer data*

Shift Left - *Applies a user-specified left shift to each data value of the input frame*

Shift Right - *Applies a user-specified right shift to each data value of the input frame*

Unpack - *Unpacks 16-bit integer data to become 8,4,2, or 1-bit data*

Communications Functions

1st Order Butterworth Filter - *Performs 1st order Butterworth filtering on the input data based on some conditions*

AGC - *Automatic Gain Control*

Averaged Periodogram - *Calculates the averaged periodogram of the input signal*

BER - *Compares the two input channels using the exclusive-or function to count bit errors and calculate bit error rate*

Fixed Offset - *Adds a specified offset value to each element of the input signal*

Gain - *Multiplies each element of the input signal by a constant*

Integrate and Dump - *Performs integrate and dump*

Leaky LMS Adaptive Filter - *Performs a leaky least mean square adaptive filter on the input signal*

LMS Adaptive Filter - *Performs a Least Mean Square adaptive filter on the input signal*

Loop Filter - *Performs first order loop filtering on the input data*

NCO - *Numerically Controlled Oscillator - generates sinusoidal signals with frequencies governed by the amplitude of the input signal*

Periodogram - *Calculates the periodgram of the input signal*

Phase Decoder - *Decodes the phase angle into data bits*

Phase Locked Loop - *Used to effectively discriminate between a feedback frequency and the input signal*

PSK (Phase Modulation) - *Creates a phase modulated signal using 2(BPSK), 4(QPSK), or 8 symbols per baud*

Rectify - *Performs either half or full wave rectification on the input signal*

Spectral Inversion - *Performs a spectral inversion (flipping spectrum) in the time domain*

Strobe - *Finds the amplitude of the signal at the given index (zero-based) within the frame*

Companding

u-255 Decode - *Performs the u-255 decode function on the input signal*

u-255 Encode - *Performs the u-255 encode function*

A-87.6 Decode - *Performs A law expansion on the input signal*

A-87.6 Encode - *Performs A law compression on the input signal*

Conditional Operators

Equal - *Checks for Equal condition and controls the block connected to this block*

Greater Than - *Checks for Greater Than condition to control the block connected to this block*

Greater Than/Equal - *Checks for Greater Than/Equal condition to control the block connected to this block*

Less Than - *Checks for Less Than condition to control the block connected to this block*

Less Than/Equal - *Checks for Less Than/Equal condition to control the block connected to this block*

Loop Counter - *Performs the Loop Counter operation to control the block connected to this block*

NOT - *Creates the logical opposite of the input signal.*

Not Equal - *Checks for Not Equal condition to control the block connected to this block*

Digital Logic Functions

1 to 2 Demultiplexer - *Demultiplexes one input channel to two channel output based on a select input*

1 to 4 Demultiplexer - *Demultiplexes one channel input to four channel outputs based on two control inputs*

2 to 1 Multiplexer - *Multiplexes two channel input to one channel output based on a select input*

4 to 1 Multiplexer - *Multiplexes four channel input to one channel output based on two control inputs*

Displays

2-Channel Display - *Displays input from two channels in several display modes.*

Digital Display - *Displays the input signal using digital numbers*

Single Channel Display - *Displays input from a single channel in several different display modes*

Text Display - *Text display block*

XY Display - *Diplays two channel signals using two orthogonal axes*

DSP Functions

1/3 Octave Band Analyzer - *Performs 1/3 octave band analysis on the input signal*

2-Sided Magnitude - *Calculates the double-sided magnitude of the complex input signal*

2-Sided Phase - *Calculates double-sided ArcTangent of both the real and imaginary components of the complex input*

Accumulate - *Accumulates data values of all frames*

For a complete list, visit hyperception.com/RIDE

Capabilities for Advanced DSP Development

RIDE includes specialized capabilities to address requirements of advanced DSP design and result in faster overall application development. These capabilities include:

Integrated Object Linker- DSP object linker for efficient direct generation of executable code.

Symbolic Debugging- RIDE creates and maintains a symbol table of DSP resources. Symbol tables provide important information for low-level application analysis and debug or integration with third-party debuggers.

DSP Code Profiling- Profile your code at the block and DSP level. Get access to block execution number, number of clock cycles for a single execution and total clock cycles in execution after reset.

DSP Memory Operations - Control the memory map of each DSP resource for algorithm development on custom DSP hardware.

System-Level Statistics - You have visibility into your real-time design including memory analysis and processing performance.

Multiprocessor Support - RIDE can support systems with multiple processors; each block function allows for a different DSP resource. The graphical nature of the RIDE design environment offers an efficient method to program multiple DSP processors.

Synchronization Support - Synchronize real-time blocks that execute at different rates. This synchronization is sometimes required in multi-rate applications for two or more processes needing to synchronize data.

DSP Interrupt Support - RIDE includes DSP interrupt support of any real-time block, or complete block diagram. Link blocks to any DSP interrupt for visual interrupt handler creation.

DSP Applications

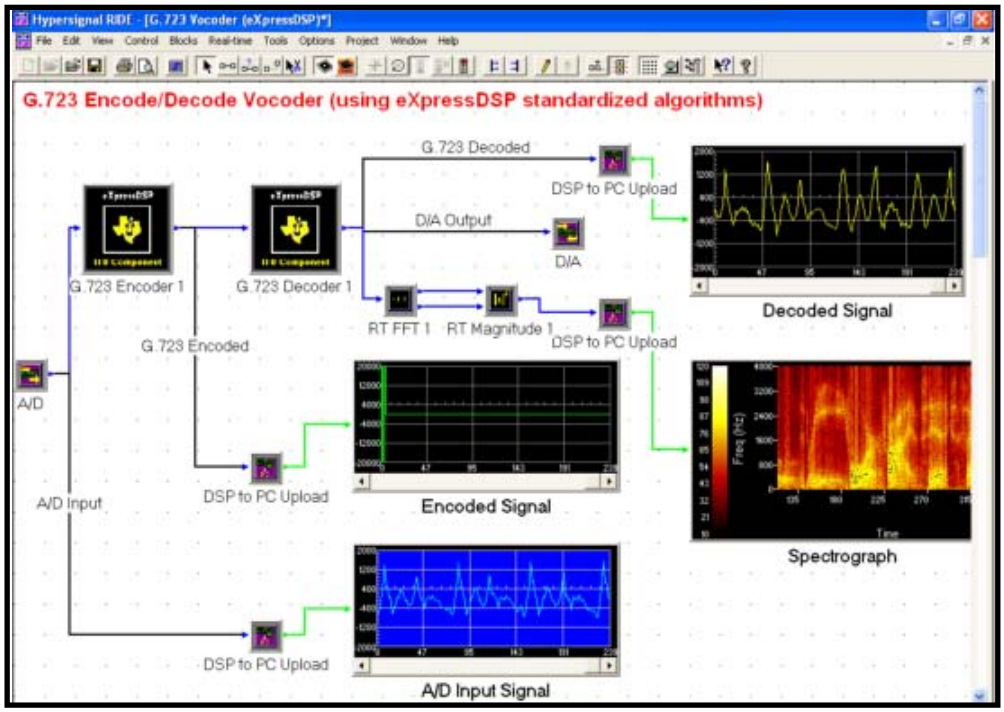
Support for Virtually All DSP Applications

RIDE is a general purpose DSP development tool akin to an assembler or C compiler, but without the complexity, and is used in many signal processing applications. The graphical methodology employed by RIDE software is ideal for a wide variety of signal processing applications including communications, image processing, medical-related technology, automotive applications, military applications, and many others.

RIDE includes a multitude of specialized components to address the depth required by many DSP product areas, and results in quicker overall application development. Efficient inline scaler functions as well as powerful vector based functions within RIDE allow both point-by-point processing and array processing to be handled well.

RIDE packages many design templates in important DSP application areas - communications, control, pro-audio, telecom, image processing, research, teaching or education, and more - which get you quickly started with your application.

Visit **www.hyperception.com** for application notes and additional application examples.

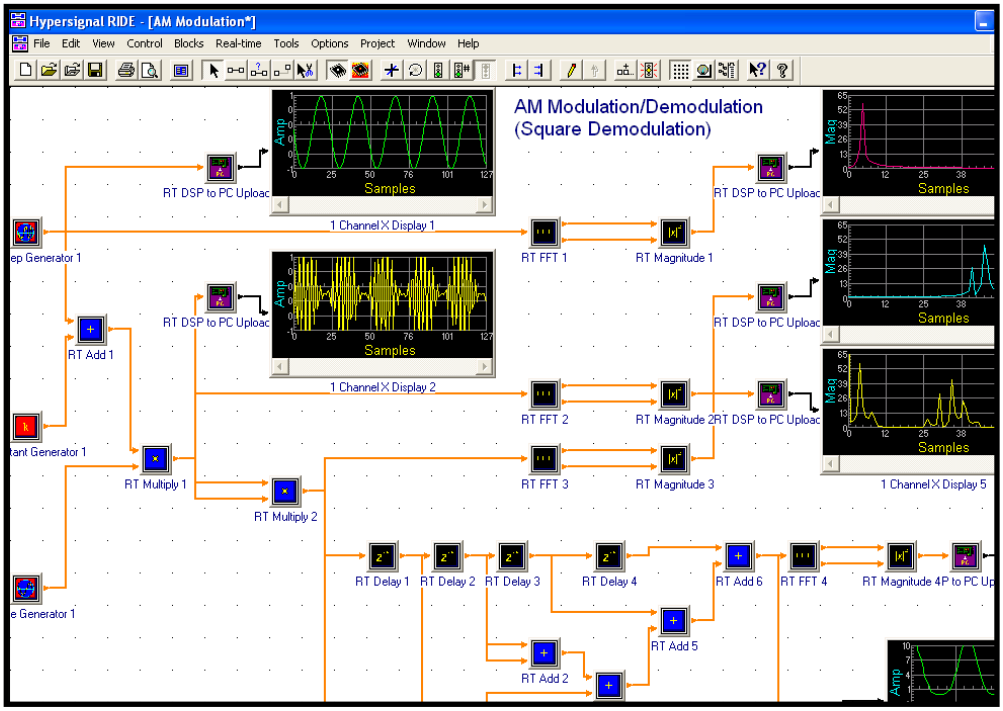


Speech Processing - Program the DSP and observe the interactive programming experience of RIDE by speaking into a microphone and hearing processed results live.

Telecom

Design your DSP algorithms graphically. RIDE graphically uses the capabilities provided by the Texas Instruments' TMS320 DSP Algorithm Standard technology. In this case a G.723 Encoder/Decoder for Texas Instruments C6000 is used in design. This target DSP application makes use of eXpressDSP™-compliant algorithms and is executed in real-time on a Texas Instruments C6000-based DSP hardware target.

DSP Applications



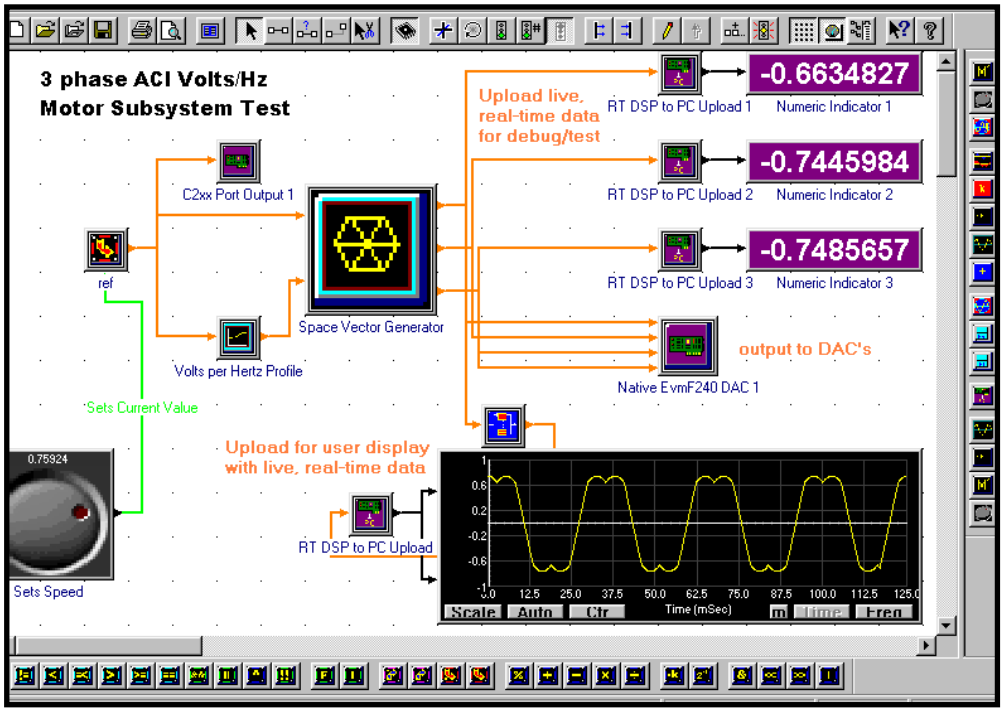
DSP-based Communications - Quickly build, modify, and test your real-time DSP application graphically with RIDE.

Communications

Use RIDE to build a basic modem application running on a DSP from Texas Instruments or Analog Devices. The user may run directly in real-time (with live interchange of real-time data) within the environment, or use RIDE to create the actual DSP object code for final production use.

Control

Use RIDE to design complex motor control applications. RIDE includes primitive and application specific blocks to support a variety of control applications. RIDE is used to graphically create an ACI motor control application designed for Texas Instruments C2000-based DSP. The design process is performed with the user selecting the software components desired for the real-time application and connecting them together with a mouse.



DSP-based Motor Control - The interactive nature of RIDE allows for dynamically changing parameters on the DSP in real-time while observing the results in the form of waveforms, graphs, or indicators.

RIDE Integration with Education and Test

DSP in Education

Graphical programming for DSP is revolutionizing industry. A RIDE based laboratory makes researchers more productive. VAB®, a subset of RIDE, improves the way students learn. Educators and students focus on concepts and results rather than on tedious implementation details. Students still learn methodology, but spend more time learning from experiments instead of programming them.

Hyperception is a partner with The Infinity Project℠ and produces the DSP Education Kits used by that successful program (for more info on The Infinity Project, go to www.infinity-project.org). DSP Education Kits include real-time DSP based hardware board, speakers, microphone and software to get your DSP Lab up and running quickly. You will benefit from example applications that work right out of the box and technology that readily adapts to your changing requirements.

Through the strong connection to National Instruments and LabVIEW™, Hyperception understands your needs as an educator and continuously develops resources to increase your effectiveness in the lab and classroom.

Student Benefits

- Focus on the engineering, not the tools
- Gain exposure to leading industry software
- Prepare for a professional career

Instructor Benefits

- Take advantage of low-cost pricing and licensing
- Update your laboratory with leading software tools
- Integrate into existing National Instruments measurement based classrooms or laboratories
- Use the available textbook from Prentice Hall to easily provide a world class curriculum to your students

Textbook

Exciting new textbook from Prentice Hall is available and supports the DSP Education Kit

Test with Virtual Instrumentation

RIDE interfaces with National Instruments LabVIEW and PXI modular instrumentation for rapid design and test of DSP based products. Share data from your target between RIDE and LabVIEW while your DSP application is running. Use virtual instrumentation to provide stimulus/response types of testing and to characterize your algorithm.

National Instruments LabVIEW and modular instruments are a powerful combination of sophisticated measurement and analysis software, and high-performance measurement hardware. Use modular instruments to interface your prototype board with test signals - video, audio, analog, and digital signals. Throughout the design flow, NI virtual instrumention helps build versatile, cost-effective rapid prototyping test systems. Reduce your time to market by investing in a user defined instrumentation platform that employs:

Graphical measurement and analysis software
High-speed digitizers
Signal generators
High-speed digital I/O
Digital multimeters
Dynamic signal acquisition

DSP based System Test - Plug-in measurement I/O for desktop and portable devices.

Choosing the Right Development System

Hyperception offers RIDE software in number of configurations to help you build professional DSP applications. Choose from three different software editions: Standard, Professional, and Enterprise.

RIDE Enterprise Edition

Designed for the advanced DSP developer, the RIDE Enterprise includes all the functionality of the RIDE Professional Edition, with the additional capability for automated C source code generation to support virtually any target processor.

RIDE Professional Edition

The RIDE Professional Edition is the ideal solution for those needing both block diagram programming support as well as run time applications. It contains all the features of RIDE Standard Edition and adds the run time Application Builder to create stand alone host executables.

RIDE Standard Edition

Use RIDE Standard Edition for direct graphical programming of DSP using block diagram methodology. RIDE Standard supports DSPs from leading vendors such as Texas Instruments and Analog Devices. RIDE Standard includes DSP-based function blocks to build complete target DSP applications as well as PC-based function blocks for simulation, modeling and limited real-time DSP using only a PC and sound card.

Available RIDE Packages

P/N	Description	DSP HW Support	Embedded Support	Block Wizard	Run-time Application	C Code Generation
HSWN8000	RIDE Standard Edition	X	X	X		
HSWN8100	RIDE Professional Edition	X	X	X	X	
HSWN8200	RIDE Enterprise Edition	X	X	X	X	X

Optional Libraries

RIDE supports several additional libraries for advanced DSP applications. The Image Processing Library, Speech Library, and Advanced Transmission Library are available for users with interest in these areas. Contact Hyperception for more details on these libraries or visit www.hyperception.com to download data sheets.

Services and Support

Hyperception has the services and support to meet your needs- from planning and development through deployment and ongoing maintenance.

RIDE Training

Hyperception offers Basic and Advanced RIDE Instructor-led training. We can also work with you to plan a course customized to your needs and can additionally provide training onsite at your facility.

Software Maintenance with Automatic Upgrades

For convenient software maintenance RIDE allows for updates by using an automated update utility to make sure your software is always current with the latest block functions and RIDE improvements.

DSP Engineering Services

Hyperception has supplied leading edge DSP engineering software since 1984. With our leading edge DSP development tools, and our signal processing experience, Hyperception is an excellent choice when you need semi-custom software development for DSP-related applications.

Visit **www.hyperception.com/support** for more information related to our services and support.

HYPERCEPTION
A NATIONAL INSTRUMENTS COMPANY

For more detailed information, please contact:

Hyperception, Inc.
9550 Skillman LB 125
Dallas, TX 75243
Voice: (214) 343-8525 Fax: (214) 343-2457
Toll Free: (800) 450-8999

www.hyperception.com * info@hyperception.com

Printed in USA