Comparison between HYPERSIGNAL® from Hyperception and MATLAB®/SIMULINK® from the MathWorks

Overview

There are many users of MATLAB, and potential users who are interested in finding out the respective merits of working with Hypersignal and MATLAB. This brief document summarizes how Hypersignal may be used along with MATLAB for some applications, and provides a table of differences between the two product lines.

Working with Hypersignal and MATLAB together

There are a number of existing MATLAB users who find that certain features of our Hypersignal products are desirable for some facets of their development. A common question is related to how Hypersignal may interoperate with MATLAB; the answer is that MATLAB is capable of using disk files for its input and output, and these files may be read and written from Hypersignal. So Hypersignal can read data produced by MATLAB, and operate/analyze, etc. this data, and may also produce output data files, with which MATLAB can deal. This allows for a relatively low risk in data transfer from one environment to another.

Comparison

Many prospective users of Hypersignal are curious as to the differences between the MATLAB product line and Hyperception's products. The following tables have been prepared to provide some insight into those differences. The information for this comparison between Hypersignal from Hyperception and MATLAB/SIMULINK from the MathWorks was contributed to Hyperception by two outside companies, and not created by Hyperception. These two companies were quite familiar with both product lines. The reader is encouraged to study and compare each product in more detail with information provided by the respective vendors.

Comparison Table

Capability	Hypersignal from Hyperception	MATLAB & SIMULINK from The MathWorks
Language & Formalism	Hypersignal uses block diagram as its formalism, and provides a complete set of functions to allow the modeling and simulation of any algorithm with this formalism. With its three types of links, (ie signal, conditional, and parameter), Hypersignal adds important components to the block diagram formalism. Also, the global variable block, directly available from the standard block library, allows the modification of any set of parameters directly from the block diagram representation.	Although SIMULINK can call MATLAB and the reverse, these two tools are well separated and require the learning of two formalisms instead of one. (MATLAB is a script/language based tool, and SIMULINK is a block diagram tool). SIMULINK provides only one type of link (signal), compared to the three in Hypersignal. Tasks like computing on a global parameter need to be specified in MATLAB, which forces SIMULINK users to work in a textual like tool, even if they would prefer to stay in a block diagram tool.
Execution	Hypersignal works at executable speed (not interpretive) and is exceptionally fast.	MATLAB & SIMULINK work at interpretive speeds and are far slower than Hypersignal.
Ease of use	Hypersignal is a very intuitive block diagram tool, providing all of it's math, modeling, simulation and graphics capabilities in the same environment and formalism. This can easily be seen even in the evaluation copy of Hypersignal Block Diagram.	MATLAB is a script language that needs to be learned. SIMULINK is easy enough to use, but the user will in many cases need to use MATLAB functions in his models (e.g.: use of global parameters and the defining of calculations on those parameters), and thus need to learn the MATLAB language anyway.
Data Typing	The type of data is simply selected in a parameter box between char, byte, short, int, long, float and double precision. Execution takes care of the data typing, allowing faster simulation where double precision is not needed.	MATLAB & SIMULINK give no natural access to data typing. This results in even more speed loss as compared to Hypersignal, when data does not need to be in double precision.

Hypersignal & MATLAB/SIMULINK Comparison Table

Capability	Hypersignal from Hyperception	MATLAB™ & SIMULINK from The MathWorks
Simulation Strategy	Hypersignal uses a data driven technique, far better suited to discrete systems modeling and simulation, to signal processing, and image processing applications. The use of this technique contributes to the high simulation speed of Hypersignal.	SIMULINK is a dynamic simulation tool that is based on a time driven approach, well adapted to control applications. No natural support for frames is provided in SIMULINK, and signal processing simulation is very slow. No image processing is included in SIMULINK. MATLAB, is a math tool, allowing to process one frame of signal or one image at a time. Its mode of simulation may be compared to the simulation on one frame in Hypersignal.
Number of Functions	Block Diagram (Core module in the Hypersignal environment) has many blocks that are standard. In many cases there is no need to purchase any optional libraries. The easy use of the Block Wizard can rapidly expand the function set as well.	MATLAB alone is a very complete Math Tool, but for any application in signal processing or image processing it is necessary, most of the time, to purchase a specialized toolbox.
Intrinsic Function Creation	Hypersignal allows creation of new blocks from other existing blocks, using a hierarchical design approach, with graphical programming of blocks.	SIMULINK allows graphical programming of hierarchy blocks, and MATLAB allows the creation of MATLAB based functions.
External Function Creation using C functions	Hypersignal provides the Block Wizard, which automatically generates the C source code structure necessary to produce a new block in the Hypersignal environment. The newly generated block, once compiled, is a simple DLL. In the generated C code structure, the user does not even need to make any variable or parameter declaration. This is done for the user as the Block Wizard prompts the user for the variables to be changed when the block's parameter box is opened. He only has to write his algorithm and produce the corresponding DLL with a C/C++ compiler such as Microsoft™ Visual C/C++™. Advantages of Block Wizard: • The user only needs to fill in the code describing the algorithm (i.e.: no need to be an expert C programmer at all); • The generated block executes at execution speed; • The user can include calls to any library or assembly code he likes very easily.	SIMULINK and MATLAB allow the creation of C user defined functions. However, no automated generation tool exists for the creation of the structure of the needed code. The development of C based functions in MATLAB & SIMULINK, is far less easy than with the Block Wizard in Hypersignal. Moreover, even though they are compiled, the C functions in MATLAB and SIMULINK still work in an interpreted environment (This results in a loss of speed).
Project Security and Management	Hypersignal provides a tool to define rights on a project (Write, Read & Write, etc.), a tool to visualize a project, its structure, and to see the history of each block in the project.	MATLAB & SIMULINK have no tool for project management and security.

Hypersignal & MATLAB/SIMULINK Comparison Table

Capability	Hypersignal from Hyperception	MATLAB™ & SIMULINK from The MathWorks
Graphics	Hypersignal graphics are provided in well integrated blocks. All of the functions in the graphics blocks are accessible via dialog boxes that allow the interactive change of any parameter during a simulation. To see examples of Hypersignal Graphics, ask for product brochures or the evaluation version.	MATLAB provides a complete set of graphic capabilities, but those are accessible through interpreted commands that need to be learned. SIMULINK provides several graphic blocks that are less powerful than those in Hypersignal.
User interfaces	In Hypersignal, a C user defined block may have its own auto-generated user interface. Hypersignal provides also a full set of control blocks (Cursors, keypads, LED's, sliders, etc.) that allow the very easy creation of user interfaces.	MATLAB provides a user interface generator, that will work on MATLAB based functions. No automatic tool is provided for the creation of user interfaces associated to a SIMULINK model.
Stand alone applications	Hypersignal provides as an option a very powerful ANSI C source code generator, that can be used either for a real-time implementation (see points on Real-time and Pegasus™ comparison), or to export an algorithm to be cross-compiled for any platform. The code generated by the C code generator is both platform independent and royalty free. HAppl™, the optional application builder in Hypersignal, creates with a simple mouse click independent applications based on any Hypersignal Model whether it is simulation or real-time. The generated application includes: the user interface, the algorithms, the drivers of any supported I/O devices or DSP boards used in the model. The generated application can run on any PC platform under Windows 95, 98 or NT. Not all applications built with HAppl are royalty free. Regarding royalties for HAppl applications, please contact Hypercetion for more details.	MATLAB, provides as an option, the MATLAB compiler, that turns calculation .m files only (no graphics supported), into compiled C MATLAB functions. To run the generated C file outside MATLAB, the C math library is required. Regardless the royalties that need to be paid in case the code is duplicated, it is not possible to run it on any platform on which the C math library is not available. The Real Time Workshop generates C code from a SIMULINK model. In cases where calls to MATLAB files are made, the Real Time Workshop does not translate the code corresponding to the MATLAB based files. In many cases it is necessary to add MATLAB based files to build a complete SIMULINK model. As a consequence, to generate the C code of a SIMULINK model, the user often needs to turn all his MATLAB based calls into C based functions in order to export his complete model. (This may represent a large amount of work!)
Acquisition and online (Real- time) processing on a PC	Hypersignal has blocks for the acquisition of real-time signals with devices spanning from **sound cards up to PCI acquisition boards sampling at 500 MHz**! On a typical PC, many signals can be acquired and processed online. With the Image Processing Library (or ImPro Lab™), it is also possible to acquire images with low cost standard video cameras, and process the image on line (e.g. Track an object in real time).	Although MATLAB (like Hypersignal does as well) reads Wave files, or can read a data file retrieved from an acquisition device, it is too slow for the online processing of acquired signals on a PC. SIMULINK is also too slow for online processing of acquired signals on a PC. The same as above applies to video devices. (SIMULINK has no Image processing capability).
Cost	Hypersignal Block Diagram or RIDE are generally considerably lower cost solutions	MATLAB/SIMULINK generally are higher cost solutions than Hypersignal

In plenty of cases it is necessary to add MATLAB based files to build a complete SIMULINK model. As a consequence, to generate the C code of a SIMULINK model, the user often needs to

Hypersignal & MATLAB/SIMULINK Comparison Table

Capability	Hypersignal from Hyperception	MATLAB™ & SIMULINK from The MathWorks
Real Time Support using a DSP board	Hypersignal RIDE™ (Real Time Integrated Development Environment), works directly on real time blocks associated to optimized object code for the DSP target. For each real time function, one object code for each supported DSP board is provided. RIDE uses the unique graphical compiling technique that produces the executable for the DSP, directly from the block diagram. (No Cross compiler needed). **RIDE supports a very large number of DSP boards, from a variety of DSP hardware manufacturers**. To target a board the user needs only to select the appropriate board driver in RIDE. This way an algorithm can be ported from one DSP board to another by a simple mouse click. Each real time block of RIDE has a real time parameter user interface, where the user can select interactively the DSP board driver, activate an interrupt or set a synchronization flag. In the same worksheet any number of DSP board drivers may be used, allowing heterogeneous multi-processing (i.e. for multiple DSP boards applications). After the real time model has been automatically loaded the user can access the number of cycles needed for the execution of each block, also, the address of the code and its size, the address of the data and its size. With the profiling capability it is possible to see at a glance how the implemented real time code is performing globally and on a block per block basis. Also the user has access to the memory usage and can look at any memory zone in the DSP, either graphically or as a text. Also the table of symbols is provided to allow advanced optimization of the real time application. Prior to compile the user can modify interactively the DSP memory mapping to adapt it to his actual need and the actual memory setting of his real time DSP board. RIDE can be used also to export the executable DSP file (COFF output file), on any DSP chip supported. For those whom only whish to work on implementation aspects, Hyperception has built the OORVL™ graphical compiler, using the same unique graphical compiling technique. **This bridges the gap between rapid prototyping and implementation**.	SIMULINK's optional, Real Time Workshop™ works with dSPACE hardware and environment, but does not provide true support to other hardware. Although SIMULINK/RTW/dSPACE is adapted to control applications, it is not as well-suited for conventional signal processing applications. The use of heterogeneous DSP systems (multi DSP boards applications), is not supported by SIMULINK/RTW. The only working choice would be to program a multi DSP system from dSPACE (ie adapted only to control and one type of hardware). The only info the user gets in dSPACE after the implementation has been done is the execution time (No profiler, no memory usage, no DSP memory plot or dump, no table of symbols) . The code produced by the Real Time Workshop, can be compiled using only one compiling option for all the code. (No possibility to set a different optimization option for each block). No interactive tool is provided to change the memory mapping of the target DSP. To export a code the user is mainly left with the C code generated with the Real Time Workshop, and his cross compiler. SIMULINK has no graphical compiling capability. The design cycle in MATLAB/ SIMULINK stops at the rapid prototyping phase. The prototyping phase is limited itself mainly to the use of the dSPACE hardware and to control applications.
Use of an RTOS	see PEGASUS Comparison	see PEGASUS Comparison