

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
22 June 2006 (22.06.2006)

PCT

(10) International Publication Number
WO 2006/065620 A2

(51) International Patent Classification:
G06F 17/50 (2006.01) **G06F 9/44** (2006.01)

(21) International Application Number:
PCT/US2005/044455

(22) International Filing Date:
7 December 2005 (07.12.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/012,376 14 December 2004 (14.12.2004) US

(71) Applicant (for all designated States except US): **THE MATHWORKS, INC.** [US/US]; 3 Apple Hill Drive, Natick, Massachusetts 01760 (US).

(72) Inventor; and

(75) Inventor/Applicant (for US only): **GAUDETTE, Thomas** [US/US]; 27 Hampstead Road, Jamaica Plain, Massachusetts 02130 (US).

(74) Agents: **CANNING, Kevin, J.** et al.; Lahive & Cockfield, LLP, 28 State Street, Boston, Massachusetts 02109 (US).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

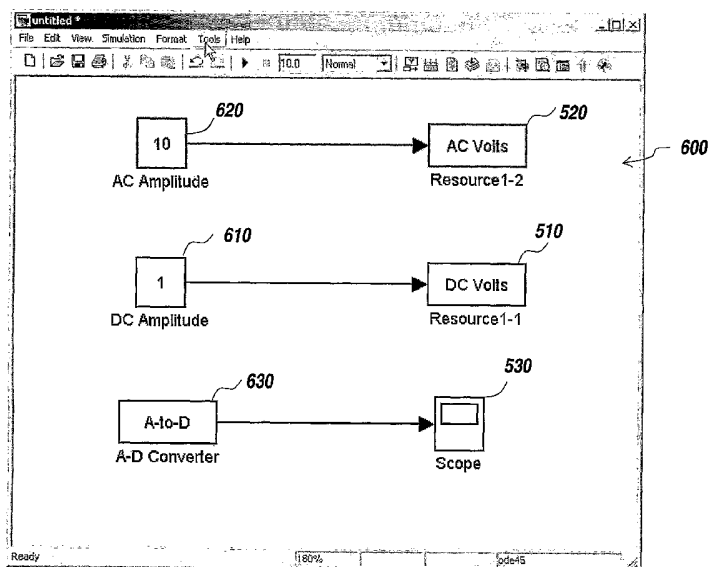
(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— without international search report and to be republished upon receipt of that report

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: SIGNAL DEFINITIONS OR DESCRIPTIONS IN GRAPHICAL MODELING ENVIRONMENTS



(57) Abstract: Methods, system and computer program products are disclosed for providing a graphical modeling environment in which a graphical model is generated and executed. In the graphical modeling environment, elements are provided to define or describe signals associated with resources that are coupled to the graphical modeling environment. The highlevel signal definition or description elements define or describe the signals associated with the resources regardless of the hardware of the resources. With the use of highlevel signal definition or description elements, the users have the capability to deal with the signals transmitted to/from the resources without the specific knowledge of the hardware of the resources.

SIGNAL DEFINITIONS OR DESCRIPTIONS IN GRAPHICAL MODELING ENVIRONMENTS

Related Applications

5 This application claims priority to U.S. Application Serial No. **11/012,376**, entitled “Signal Definitions or Descriptions in Graphical Modeling Environments,” Attorney Docket No. MWS-119, filed on December 14, 2004, the contents of which are hereby incorporated by reference.

Field of the Invention

10 The present invention relates generally to graphical modeling environments and more particularly to methods, systems and computer program products for providing signal
15 definitions or descriptions in the graphical modeling environments.

Background of the Invention

20 In an embedded control system, the interaction between the digital units, such as a controller, and the analog units, such as a controlled plant, of the system is critical. This interaction is accomplished generally by signal conditioning hardware, such as analog-to-digital (AD) and digital-to-analog (DA) converters. The signal conditioning hardware transforms low-power electrical signals in the operational range of logic signals (e.g., 0-5V) to high power signals to drive transducers, such as pumps, motors and valves, or vice versa.

25 ‘Smart sensors’ (such as the IEEE 1451 standard) alleviate the burden of signal communication between the digital and analog units of a system. The smart sensor includes a microprocessor that contains information about the measurement of the sensor, such as the unit and accuracy of the measurement. The microprocessor then communicates directly in
30 terms of signal values instead of having to convert them in binary representations that concur with the selected digitization and gain of the signal conditioning hardware.

In some text-based programming environments, users are allowed to use high-level signal definitions or descriptions instead of having to provide a sequence of hardware dependent commands to obtain a certain set of data. The high-level signal definitions or descriptions define or describe signals regardless of the hardware that generates or measures the signals. There are many standard signal definition or description languages available for traditional text-based programming languages that allow users to define or describe signals, apply signals, and measure signals. In those languages, users do not need to care about the hardware used to generate or measure the signals. For example, the Abbreviated Test Language for All Systems (ATLAS) allows standard signal operations, such as APPLY, MEASURE and VERIFY. Another example of the signal definition or description standard can be found in the "IEEE P1641" standard.

Recently, graphical programming environments are widely employed to model and simulate engineering and scientific systems, including embedded control systems. The graphical modeling environments provide tools for creating graphical models of the systems and for executing the graphical models. Exemplary graphical modeling environments can be found in time-based block diagram modeling environments, such as those found within Simulink® from The MathWorks, Inc. of Natick, MA, state-based and flow diagram modeling environments, such as those found within Stateflow® from The MathWorks, Inc. of Natick, MA, data-flow diagram modeling environments and Unified Modeling Language (UML) modeling environments. In these graphical modeling environments, the high-level definitions or descriptions of signals are also needed for the users to create and simulate the graphical models regardless of the hardware that generate or measure the signals.

Summary of the Invention

The present invention provides a graphical modeling environment in which a graphical model is generated and executed. In the graphical modeling environment, elements are provided to represent physical signals associated with resources that are coupled to the graphical modeling environment. The elements include information to define or describe the physical signals associated with the resources regardless of the hardware of the resources. Users are allowed to use the high-level signal definition or description elements to generate and execute the graphical model. Users are able to create and execute the graphical model using the elements describing physical signals instead of incorporating elements that represent

the hardware of the resources. With the use of high-level signal definition or description elements, the users have the capability to deal with the physical signals transmitted to/from the resources without the specific knowledge of the hardware of the resources.

5 In one aspect of the present invention, a method is provided for describing physical signals in a graphical modeling environment. The method includes the step of providing a graphical element for describing physical signals associated with a hardware component that transmits or receives the physical signals. The method also includes the step of incorporating the element in the model.

10 In another aspect of the present invention, a method is provided for defining signals for a model in a graphical modeling environment. The method includes the step of determining resources coupled to the modeling environment. The method also includes the step of generating graphical elements that describe signals associated with the resources. The graphical elements describe the signals associated with the resources regardless of hardware of the resources.

15 In another aspect of the present invention, a system is provided for describing signals for a model in a graphical modeling environment. The system includes resources coupled to the modeling environment. The system also includes a data structure for graphical elements that describe signals associated with the resources. The data structure describes the signals associated with the resources regardless of hardware of the resources.

20 In another aspect of the present invention, a computer program product is provided that holds instructions executable in a computer that provides an environment for modeling a system. The computer program product provides a graphical element that describes signals associated with a hardware component that transmits or receives the signals. The computer program product enables users to incorporate the element in the model.

25 In another aspect of the present invention, a computer program product is provided that holds instructions executable in a computer that provides an environment for modeling a system. The computer program product determines resources coupled to the modeling environment. The computer program product generates graphical elements that describe

signals associated with the resources. The graphical elements describe the signals associated with the resources regardless of hardware of the resources.

Brief Description of the Drawings

5

The aforementioned features and advantages, and other features and aspects of the present invention, will become better understood with regard to the following description and accompanying drawings, wherein:

10 FIGURE 1 shows an exemplary computational device suitable for practicing the illustrative embodiment of the present invention;

FIGURE 2 depicts an exemplary block diagram modeling environment provided in the illustrative embodiment of the present invention;

15 FIGURE 3 is a flow chart showing an exemplary operation of the illustrative embodiment of the present invention for creating and simulating a model in the block diagram modeling environment;

FIGURE 4 is a flow chart showing an exemplary operation of the illustrative embodiment of the present invention for generating signal definition or description elements in the block diagram modeling environment;

20 FIGURE 5 displays a library that contains exemplary signal definition or description elements generated in the block diagram modeling environment; and

FIGURE 6 is an exemplary model created in the block diagram modeling environment using the signal definition or description elements.

Detailed Description

25

Certain embodiments of the present invention are described below. It is, however, expressly noted that the present invention is not limited to these embodiments, but rather the intention is that additions and modifications to what is expressly described herein also are
30 included within the scope of the invention. Moreover, it is to be understood that the features of the various embodiments described herein are not mutually exclusive and can exist in various combinations and permutations, even if such combinations or permutations are not made express herein, without departing from the spirit and scope of the invention.

The illustrative embodiment of the present invention provides a graphical modeling environment in which a graphical model is generated and simulated/executed. In the description of the illustrative embodiment, execution of the graphical model is also referred to as simulation. The illustrative embodiment will be described solely for illustrative purposes relative to a block diagram modeling environment. Although the illustrative embodiment will be described relative to the block diagram modeling environment, one of skill in the art will appreciate that the present invention may apply to other graphical modeling environments including data flow modeling environments and Unified Modeling Language (UML) modeling environments. One of skill in the art will also appreciate that the present invention may apply to other graphing tools including Microsoft Graph from Microsoft Corporation of Redmond, WA.

An exemplary block diagram modeling environment can be found in Simulink[®] from The MathWorks, Inc. of Natick, MA. Simulink[®] provides tools for modeling and simulating a variety of dynamic systems in one integrated, graphical environment. Simulink[®] enables users to design a block diagram for a target system, simulate the system's behavior, analyze the performance of the system, and refine the design of the system. Simulink[®] allows users to design target systems through a user interface that allows drafting of block diagram models of the target systems. All of the blocks in a block library provided by Simulink[®] and other programs are available to users when the users are building the block diagram of the target systems. Individual users may be able to customize this model block to: (a) reorganize blocks in some custom format, (b) delete blocks they do not use, and (c) add custom blocks they have designed. The blocks may be dragged through some human-machine interface (such as a mouse or keyboard) from the block library on to the window (i.e., model canvas). In Simulink[®], there is also a textual interface with a set of commands that allow interaction with the graphical editor, such as the textual interface provided in MATLAB[®]. Using this textual interface, users may write special scripts that perform automatic editing operations on the block diagram. Simulink[®] also allows users to simulate the designed target systems to determine the behavior of the systems.

The illustrative embodiment will be described below relative a block diagram model in Simulink[®]. Nevertheless, those of skill in the art will appreciate that the present invention may be practiced relative to models implemented in other graphical modeling environments,

including but not limited to LabVIEW from National Instruments Corporation of Austin, Texas, and Rational Rose from IBM of White Plains, New York.

In the illustrative embodiment, block elements are provided to define or describe signals associated with resources that are coupled to the block diagram modeling environment. Users are allowed to create and execute the block diagram model using the signal definition or description block elements. This high-level signal definition or description block element defines or describes the signals associated with the resources regardless of the hardware of the resources. Therefore, users are able to create and execute the block diagram model without incorporating block elements representing the hardware of the resources. With the use of high-level signal definition or description elements, the users can deal with the signals associated with the resources without the specific knowledge about the hardware of the resources.

FIGURE 1 is an exemplary computational device 100 suitable for practicing the illustrative embodiment of the present invention, which provides a block diagram modeling environment. One of ordinary skill in the art will appreciate that the computational device 100 is intended to be illustrative and not limiting of the present invention. The computational device 100 may take many forms, including but not limited to a workstation, server, network computer, quantum computer, optical computer, bio computer, Internet appliance, mobile device, a pager, a tablet computer, and the like.

The computational device 100 may be electronic and include a Central Processing Unit (CPU) 110, memory 120, storage 130, an input control 140, a modem 150, a network interface 160, a display 170, etc. The CPU 110 controls each component of the computational device 100 to provide the block diagram modeling environment. The memory 120 temporarily stores instruction and data and provides them to the CPU 110 so that the CPU 110 operates the computational device 100 and runs the block diagram modeling environment. The storage 130 usually contains software tools for applications. The storage 130 includes, in particular, code 131 for the operating system (OS) of the device, code 132 for applications running on the operation system including applications for the block diagram modeling environment, and data 133 for the models created in the block diagram modeling environment.

The input control 140 may interface with a keyboard 180, a mouse 190, and other input devices. The computational device 100 may receive through the input control 140 input data necessary for creating models in the block diagram modeling environment, such as the selection of the attributes and operations of component blocks in the models. The
5 computational device 100 may display in the display 170 user interfaces for the users to edit the models in the block diagram modeling environment. The network interface 160 and the MODEM 150 enable the computational device 100 to communicate with other electronic devices through communication networks, such as Internet, intranet, LAN (Local Area Network), WAN (Wide Area Network) and MAN (Metropolitan Area Network). The
10 communication facilities may support the distributed implementations of the present invention.

The computational device 100, which runs the block diagram model, may be coupled to various resources 195. The resources 195 may include signal source devices from which
15 data or signals are provided, such as function generators. The signal source devices may provide DC signals, AC signals, and many other waveforms of signals, for example, step signals, square wave signals, triangular signals, etc. The resources 195 may also include signal sink devices to which data or signals are provided, such as transducers including pumps, motors and valves. The resources 195 may include signal conditioning devices, such
20 as an analog-to-digital (AD) converter that transforms high power signals to low-power logic signals in the range of, for example, 0-5V, and a digital-to-analog (DA) converter that transforms low-power logic signals to high power signals to drive the transducers. The signal conditioning devices can provide interfaces between a digital controller and an analog plant in an embedded control system. One of ordinary skill in the art will appreciate that the resources
25 195 described above are illustrative and any other resources can be coupled to the computational device 100.

FIGURE 2 depicts an exemplary block diagram modeling environment 200 provided in the illustrative embodiment. The block diagram modeling environment 200 may include at
30 least a model editor 210, a model processing engine 220 and a model execution engine 230. FIGURE 3 is a flow chart showing an exemplary operation of the block diagram modeling environment 200 in the illustrative embodiment. Users may create their block diagram model 240 using the model editor 210 (step 310). The editor 210 allows users to perform such actions as draw, edit, annotate, save, and print out block diagram representations of systems.

The block diagram model editor 210 may provide a graphical user interface (GUI) component that allows drafting of block diagram model 240 by the users. Methods used in editing are called by the block diagram editor 210 in order to render the block appropriately in the GUI of its parent block diagram. The block diagram model editor 210 may allow users to specify the parameters for the block when they use it in their models.

In creating the block diagram model 240, the users may use block elements provided in the block diagram modeling environment 200. The block elements may be contained in the library 250. An exemplary block library can be found in Blocksets from The MathWorks, Inc. of Natick, MA, including Gauges Blockset, Aerospace Blockset, Signal Processing Blockset, etc. The users may also define their own block elements with desired functions and use them in the block diagram model 240. The users may define their own block elements using, for example, the S-function provided in Simulink®.

In the illustrative embodiment, the block diagram modeling environment 200 provides signal definition or description block elements to define or describe signals associated with the resources 195 that are coupled to the block diagram modeling environment 200. Users can create the block diagram model 240 using the signal definition or description block elements. The high-level signal definition or description block elements define or describe the signals associated with the resources 195 regardless of the hardware of the resources 195. Therefore, users are able to create and execute the block diagram model 240 without incorporating block elements representing the hardware of the resources. With the use of high-level signal definition or description elements, the users can deal with the signals associated with the resources without the specific knowledge about the hardware of the resources.

FIGURE 4 is a flow chart showing an exemplary operation of the illustrative embodiment of the present invention for generating signal definition or description block elements in the block diagram modeling environment 200. Users may input a request for determining resources 195 coupled to the block diagram modeling environment 200 (step 410). For example, users may input a “getresources” command in a MATLAB® command line, which provides a foundation for Simulink®. In response to the request from the users, the block diagram modeling environment 200 determines the resources 195 coupled to the block diagram modeling environment 200 (step 420). The resources 195 coupled to the block

diagram modeling environment 200 can be determined using many kinds of tools provided by various vendors, such as Instrument Control Toolbox and Data Acquisition Tools provided by The MathWorks, Inc. of Natick, MA. Although the resources 195 are determined in response to the request from the users in the illustrative embodiment, one of skill in the art will appreciate that the determination can be made in a different manner, for example, when the block diagram modeling environment 200 is initialized or each time when a new resource is coupled to the block diagram modeling environment 200. It may be preferable to have the resources 195 determined during initialization time and not force the user to enter the “getresource” command.

If the resources 195 are determined, the block diagram modeling environment 200 looks up the resources 195 in the database of the block diagram modeling environment 200 (step 430). The block diagram modeling environment 200 searches a directory, or other suitable database, such as files or look up tables, for the resources 195. One of ordinary skill in the art will appreciate that users may download information for the resources 195 from a remote data center that retains data for the resources 195. Using available information for the resources 195, the block diagram modeling environment 200 can build the block elements for defining or describing signals associated with the resources 195 and store the block elements in the library 250 so that users can use the signal definition or description block elements in creating or editing the model 240 (step 440). XML can be used to map the resources 195 to the corresponding block elements for defining or describing signals associated with the resources 195. For example, the Instrument Definition XML from IEEE can help map the resource instrument to the corresponding block element.

FIGURE 5 shows an exemplary library 500 that contains exemplary signal definition or description block elements 510 and 520 for the resources 195. The exemplary library 500 contains a DC voltage definition or description block 510 and an AC voltage definition or description block 520 for Resource 1-1 and Resource 1-2, respectively, which are coupled to the block diagram modeling environment 200. One of ordinary skill in the art will also appreciate that the block elements 510 and 520 are illustrative and the present invention may generate many other kinds of block elements describing different signals, such as an analog-to-digital converter block 530. The analog-to-digital (AD) converter block 530 may transform high power analog signals to their digital equivalent in the range of, for example, 0-

5V. The analog-to-digital (AD) converter block 530 may define or describe the transformation necessary for the mapping from the high power signals.

The information obtained for the resources 195 from the database of the block diagram modeling environment 200 or from a remote data center is used to build the data structure of the block elements 510 and 520 for the resources 195. A typical base data-structure for the block elements 510 and 520 for the resources 195 may be represented as follows:

```

class Block {
10   public:
        // Access methods for setting/getting block data
        ...
        // Methods for block editing
        virtual ErrorStatus BlockDrawIcon();
15   virtual BlockParameterData BlockGetParameterData();
        ...
        // Methods for block compilation
        ...
        // Methods for block execution
20   .....
        virtual ErrorStatus BlockOutput() = 0;
        virtual ErrorStatus BlockDerivative() = 0;
        virtual ErrorStatus BlockUpdate() = 0;
        ...
25   private:
        BlockGraphicalData blkGraphicalAttributes;
        BlockFunctionalData blkFunctionalAttributes;
        BlockCompiledData blkCompiledAttributes;
        BlockExecutionData blkExecutionData;
30   ...
};

```

Although the example of the data structure above is written in C++, those skilled in the art will recognize that equivalent data structures written in other languages may also be used.

The block data structure has a set of associated methods that may be categorized as access methods to data fields, methods used in editing, methods used in compilation and methods used in execution. Access methods to data fields help in setting and getting the various data fields of the block. Methods used in editing are called by the block diagram editor in order to render the block appropriately in the GUI of its parent block diagram. For instance, this set of methods may include a BlockDrawIcon method that determines the shape the block icon has on the GUI. Methods used in compilation are methods that are called by the block diagram compilation engine. They help validate the connections of the block to other blocks on the block diagram. The methods used in execution include a number of different run-time methods that are required for execution. These include the BlockOutput, BlockUpdate, BlockDerivative methods that realize the Output, Update, and Derivative equations discussed earlier in the context of dynamic systems. In addition, to these methods Simulink includes several other run-time methods, such as the Jacobian, Projection, ZeroCrossings, Enable, Disable, Initialize, EvalParams (check and process parameters), and GetTimeOfNextHit methods. It should be noted that there is no explicit method for algebraic equations because these are represented and processed

The data fields of the block data structure may include a graphical attributes field, a functional attributes field, a compiled attributes field and an execution data field.

The graphical attributes field is responsible for storing information relevant for graphical rendering of the block within its parent block diagram's GUI. Attributes specific to the block icon, such as font, color, name and icon-image, are stored in this field. It should be noted that modifying these attributes does not affect the dynamics of the model using this block.

The functional attributes field is responsible for specifying block attributes that may potentially affect the dynamics of the model using this block. These attributes are specified for the block as a whole and the input and output ports of the block. Examples of block attributes include block sample times and restrictive flags. Block sample times specify if the block corresponds to an elemental, continuous, discrete, or hybrid dynamic system. If the block is an elemental discrete-time system, then the attribute specifies the spacing between time instants at which the block response should be traced. A restrictive flag disallows the

use of blocks in certain modeling contexts. For example, one may impose the restriction that there may only be one instance of given block in a model.

Attributes of block ports specify properties of the data that is either available or produced at that port. Block port attributes include dimensions, datatypes, sample rates, and direct feedthrough. Dimension attributes are individual dimensions of a multi-dimensional matrix that is used as a container for data elements. Datatype attributes are the datatype of each element of data in the data container. A complexity attribute is a flag to specify if each data element is real or complex. A sample rate attribute specifies how when the signal corresponding to an input or output port will be used. The port sample times may sometimes be used to implicitly infer the block's sample time. The direct feedthrough attribute is specified only for input ports and indicates whether or not the Output and/or GetTimeOfNextHit equations of the block are a function of the given input. This attribute helps in determining the sequence in which block methods should be executed while executing the block diagram.

The data structure of the block elements 510 and 520 also include attributes for describing signals associated with the resources 195 coupled to the block diagram modeling environment 200. The attributes of the signals may include the type of signals, such as voltage signals, defined with respect to time. One of ordinary skill in the art will appreciate that the attributes of the signals may define or describe other types of signals, such as signals representing events.

The DC voltage description block 510 may include following exemplary attributes depending on the waveform of the signal.

WAVEFORM	ATTRIBUTES
Constant	Amplitude
Step	Amplitude, Start Time
Single Trapezoid	Amplitude, Start Time, Rise Time, Pulse Width, Fall Time
Single Ramp	Amplitude, Start Time, Rise Time

For example, the constant signal has one property, the amplitude of the signal. The step signal makes transition from zero to a give level. Thus, the step signal has two properties,

the start time of the transition and the final amplitude level. Before the start time, the value is zero and after the art time, the value is the amplitude. One of ordinary skill in the art will appreciate that the signals described above are illustrative and many other types of DC signals can be defined or described in other embodiments.

5

The AC voltage resource 520 may include following exemplary attributes depending on the shape of the signal.

WAVEFORM	ATTRIBUTES
Sinusoidal	Amplitude, Frequency, Phase
Trapezoid	Amplitude, Period, Rise Time, Pulse Width, Fall Time
Ramp	Amplitude, Period, Rise Time
Triangle	Amplitude, Period, Duty Cycle
Square Wave	Amplitude, Period, Duty Cycle
Waveform Ramp	Amplitude, Period, Sampling Interval, Points
Waveform Step	Amplitude, Period, Sampling Interval, Points

10 For example, a sinusoidal is a signal whereby the amplitude of the dependent variable is given by the formula:

$$a=A\sin(\omega t+\phi),$$

where A is the amplitude, ω is $2\pi \times$ frequency, and ϕ is the initial phase angle. The sinusoidal signal has amplitude, frequency and phase as parameters. One of ordinary skill in the art will appreciate that the signals described above are illustrative and many other types of AC signals can be defined or described in other embodiments.

15

The block elements 510 and 520 may include values, such as a physical value and an associated dimension described by the units. The value property assigns the complete physical value to the attributes of the signals. Changing property values may not affect other property values. The following shows exemplary quantities and units.

20

QUANTITY	UNIT
Current	Ampere (A)
Frequency	Hertz (Hz)

Power	Watt (W)
Time	Second/Minute/Hour (S/M/H)
Voltage	Volt (V)

One of ordinary skill in the art will appreciate that the quantities and units of signals described above are illustrative and many other quantities and their units can be used to describe signals in other embodiments.

5

The compiled attributes field of the block data structure holds the attributes of the block and its ports that mirror the functional attributes listed above. This field is filled in during block diagram compilation by utilizing the functional attributes of the block in conjunction with the functional and compiled attributes of the blocks that are connected to it.

10 This process of determining the compiled attributes from the functional attributes is termed attribute propagation. The execution data field is mainly responsible for storing the memory locations that are going to serve as sources for block inputs, outputs, states, parameters, and other work areas during execution of blocks.

15 The block elements 510 and 520 can be built using basic signal components that cannot be decomposed into simpler components. The block elements 510 and 520 may contain a single basic signal component or combined basic signal components to define more complex signals. The basic signal components can define static signals where the signals do not change over time. The signal component can also define dynamic signal where the value
20 of the attributes or the signal changes while the signal is being used. The basic signal components are described in "IEEE P1641" standard in more detail with reference to a text-based programming language.

FIGURE 6 is an exemplary model 600 created using the signal definition or
25 description elements in the block diagram modeling environment 200. This simple model 600 includes the DC voltage definition or description block 510, the AC voltage definition or description block 520, and the analog-to-digital converter block 530, which are depicted in FIGURE 5. The model 600 also includes DC Amplitude block 610, an AC Amplitude block 620 and a Scope block 630. The DC Amplitude block 610 may set the output level (1 in the
30 model 600) of the DC power generator represented by the DC voltage definition or description block 510. In a like manner, the AC Amplitude block 620 may set the output

level (10 in the model 600) of the AC power generator represented by the AC voltage definition or description block 520. The output signal or data of the analog-to-digital converter represented by the analog-to-digital converter block 530 is displayed in the Scope block 630.

5

The DC voltage definition or description block 510, the AC voltage definition or description block 520 and the analog-to-digital converter block 530 describe signals associated with resources, such as a DC signal generator, an AC signal generator and an analog-to-digital converter, that are coupled to the block diagram modeling environment 200.

10 The DC voltage definition or description block 510, the AC voltage definition or description block 520 and the analog-to-digital converter block 530 describe the signals in a manner that is independent of the hardware of the DC signal generator, the AC signal generator and the analog-to-digital converter, respectively. For example, the DC voltage description block 510 may describe signals produced by 33220A power supply from Agilent Technologies or by
15 DAS 16/16-AO board from Measurement Computing Corp. Because the DC voltage description block 510, the AC voltage definition or description block 520 and the analog-to-digital converter block 530 can describe signals regardless of the hardware of the resources, the users can use the DC voltage description block 510, the AC voltage definition or description block 520 and the analog-to-digital converter block 530 without any knowledge
20 about the hardware of the resources associated with the signals.

Referring back to Figure 3, when the block diagram model 240 is created or edited, the block diagram model 240 is compiled and executed by the model processing engine 220 and the model execution engine 230, respectively. The model processing engine 220 carries out
25 the task of compiling and linking the block diagram to produce an “in-memory executable” version of the model that is used for generating code and/or simulating or linearizing a block diagram model (step 320). The compilation stage involves preparing data structures and evaluating parameters, configuring and propagating block characteristics, determining block connectivity, and performing block reduction and block insertion. In the link stage, the model
30 processing engine 220 uses the result of the compiled stage to allocate memory needed for the execution of the various components of the block diagram. The linking stage also produces block method execution lists which are used by the simulation or linearization of the block diagram. After linking has been performed, the model processing engine 220 may generate code. If this stage is performed, then the stages that follow use the generated code during the

execution of the block diagram. If this stage is skipped completely, then the execution engine 230 uses an interpretive mode of execution for the block diagram (step 330).

5 It will thus be seen that the invention attains the objectives stated in the previous description. Since certain changes may be made without departing from the scope of the present invention, it is intended that all matter contained in the above description or shown in the accompanying drawings be interpreted as illustrative and not in a literal sense. For example, the illustrative embodiment of the present invention may be practiced in any other graphical modeling environments, such as a state-based modeling environment. Practitioners
10 of the art will realize that the sequence of steps and architectures depicted in the figures may be altered without departing from the scope of the present invention and that the illustrations contained herein are singular examples of a multitude of possible depictions of the present invention.

We Claim:

1. In a computational device, a method for providing signal attributes for a hardware component in a modeling environment, the method comprising:
 - 5 providing a graphical element that represents attributes of signals transmitted to or received from the hardware component; and
 - enabling users to incorporate the graphical element in a model.
2. The method of claim 1, wherein the model comprises a block diagram model and the
10 graphical element comprises a block element.
3. The method of claim 1, wherein the graphical element comprises attributes of the signals that are associated with the hardware component.
- 15 4. The method of claim 1, wherein the graphical element describes the signal with respect to time.
5. The method of claim 1, wherein the graphical element describes the signal with respect to frequency.
20
6. The method of claim 4, wherein the graphical element represents static signals that do not change with respect to time.
7. The method of claim 4, wherein the graphical element represents dynamic signals that
25 change with respect to time.
8. The method of claim 1, wherein the graphical element represents the signals regardless of a characteristic of the hardware component.
- 30 9. In a computational device that provides a modeling environment, a method for providing a signal attribute in a model, the method comprising:
 - determining resources coupled to the modeling environment; and

generating graphical elements that represent attributes of signals associated with the resources, wherein the graphical elements represent attributes of the signals associated with the resources regardless of hardware of the resources.

- 5 10. The method of claim 9, wherein the model comprises a block diagram model and the graphical elements comprise block elements.
11. The method of claim 9, further comprising:
searching for data of the resources provided in the modeling environment, wherein the
10 data is used to generate the graphical elements.
12. The method of claim 9, further comprising:
receiving a request for determining the resources coupled to the modeling environment.
- 15 13. The method of claim 9, wherein the graphical elements comprise attributes of the signals associated with the resources.
14. The method of claim 13, wherein the attributes of the signals are used to generate the graphical elements.
20
15. The method of claim 9, wherein the graphical elements enable users to create the model without information on the hardware of the resources.
16. A system for providing signal attributes in a graphical modeling environment, the system
25 comprising;
resources coupled to the modeling environment; and
a data structure for graphical elements that describes attributes of signals associated with the resources, wherein the data structure describes the attributes of the signals associated with the resources regardless of hardware of the resources.
30
17. The system of claim 16, wherein the model comprises a block diagram model and the graphical elements comprise block representation elements.

18. The system of claim 16, wherein the data structure comprises attributes of the signals associated with the resources.

19. The system of claim 18, wherein the data structure comprises attributes of the signals that describe the signal with respect to time.

20. The system of claim 19, wherein data structure comprises attributes of the signals that do not change with respect to time.

21. The method of claim 19, wherein data structure comprises attributes of the signals that change with respect to time.

22. A computer program product holding instructions executable in a computer that provides an environment for modeling a system, comprising:

providing a graphical element that represents attributes of signals transmitted to or received from a hardware component ; and
incorporating the element in a model.

23. The computer program product of claim 22, wherein the model comprises a block diagram model and the graphical element comprises a block element.

24. The computer program product of claim 22, wherein the graphical element comprises attributes of the signals that are associated with the hardware component.

25. The computer program product of claim 22, wherein the graphical element represents the signal with respect to time.

26. The computer program product of claim 25, wherein the graphical element represents static signals that do not change with respect to time.

27. The computer program product of claim 25, wherein the graphical element represents dynamic signals that change with respect to time.

28. The computer program product of claim 22, wherein the graphical element represents the signals regardless of a characteristic of the hardware component.

29. A computer program product holding instructions executable in a computer that provides an environment for modeling a system, comprising:

determining resources coupled to the modeling environment; and

generating graphical elements that represent attributes of signals associated with the resources, wherein the graphical elements represent the attributes of the signals associated with the resources regardless of hardware of the resources.

30. The computer program product of claim 29, wherein the model comprises a block diagram model and the graphical elements comprise block elements.

31. The computer program product of claim 29, further comprising:

searching for data of the resources provided in the modeling environment, wherein the data is used to generate the graphical elements.

32. The computer program product of claim 29, further comprising:

receiving a request for determining the resources coupled to the modeling environment.

33. The computer program product of claim 29, wherein the graphical elements comprise attributes of the signals associated with the resources.

34. The computer program product of claim 33, wherein the attributes of the signals are used to generate the graphical elements.

35. The computer program product of claim 29, wherein the graphical elements enable users to create the model without information on the hardware of the resources.

36. A method for creating and executing a graphical model, the method comprising the step of:

providing a graphical modeling environment for creating and executing the graphical model; and

enabling users to create and execute the graphical model in the graphical modeling environment regardless of hardware of resources that are coupled to the graphical modeling environment.

- 5 37. A method for creating and executing a graphical model in a graphical modeling environment, the method comprising the steps of:
- the graphical model including at least an element associated with resources that are coupled to the graphical modeling environment; and
- enabling the graphical model to be executed in the graphical modeling environment
- 10 regardless of hardware of the resources.

38. A system for creating and executing a graphical model in a graphical modeling environment, the system comprising;
- a graphical modeling tool for providing the graphical modeling environment; and
- 15 resources coupled to the graphical modeling environment,
- wherein the graphical modeling tool enables users to create and execute the graphical model in the graphical modeling environment regardless of hardware of the resources.

39. A system for creating and executing a graphical model in a graphical modeling
- 20 environment, the system comprising;
- an editor for creating the graphical model that includes at least an element associated with resources coupled to the graphical modeling environment; and
- an execution engine for executing the graphical model in the graphical modeling environment regardless of hardware of the resources.

25

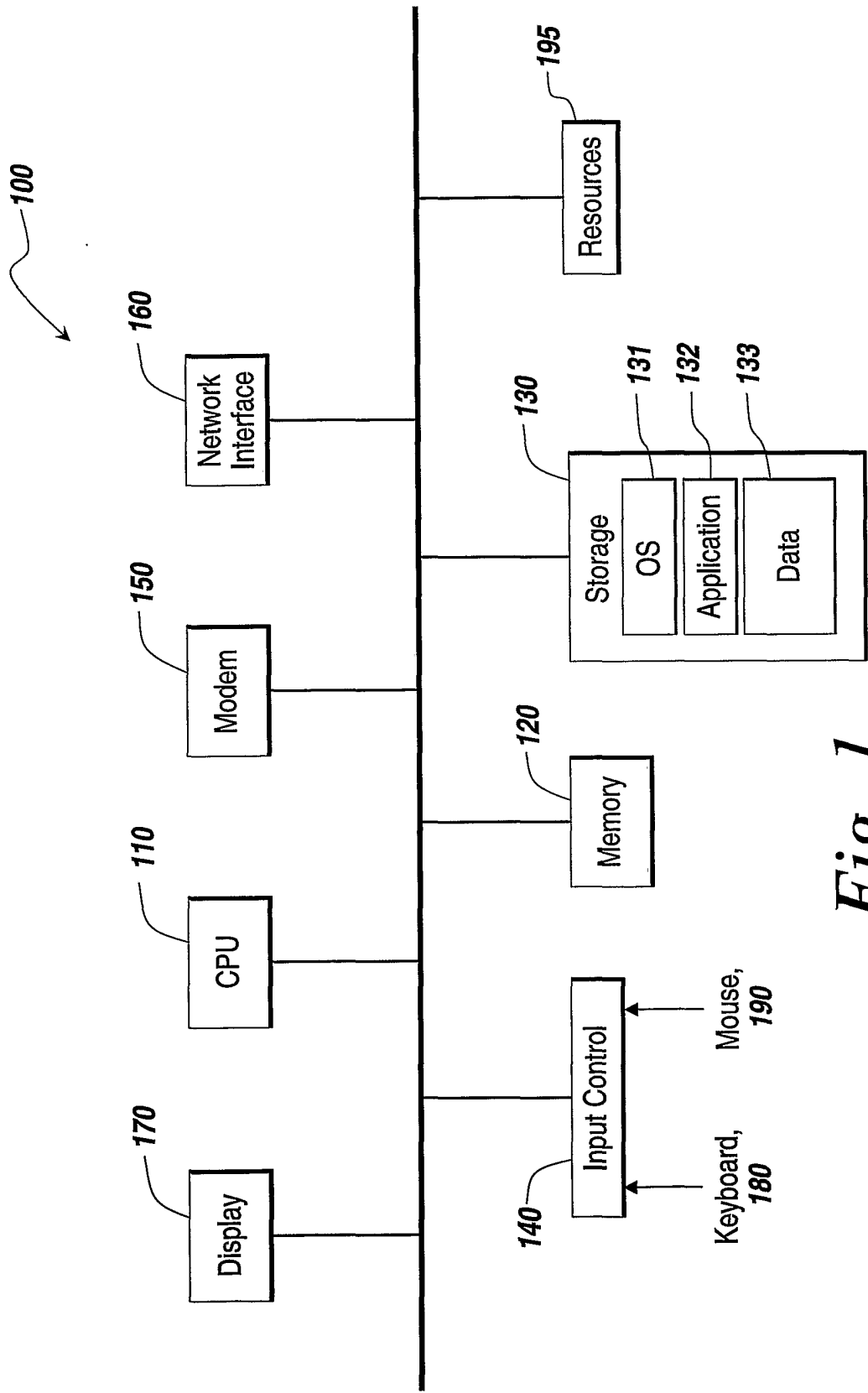
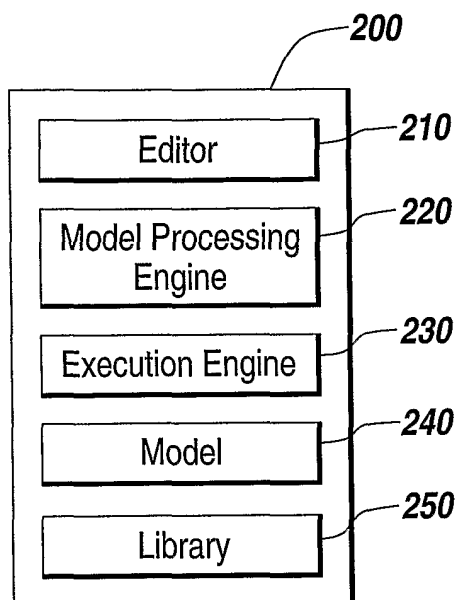
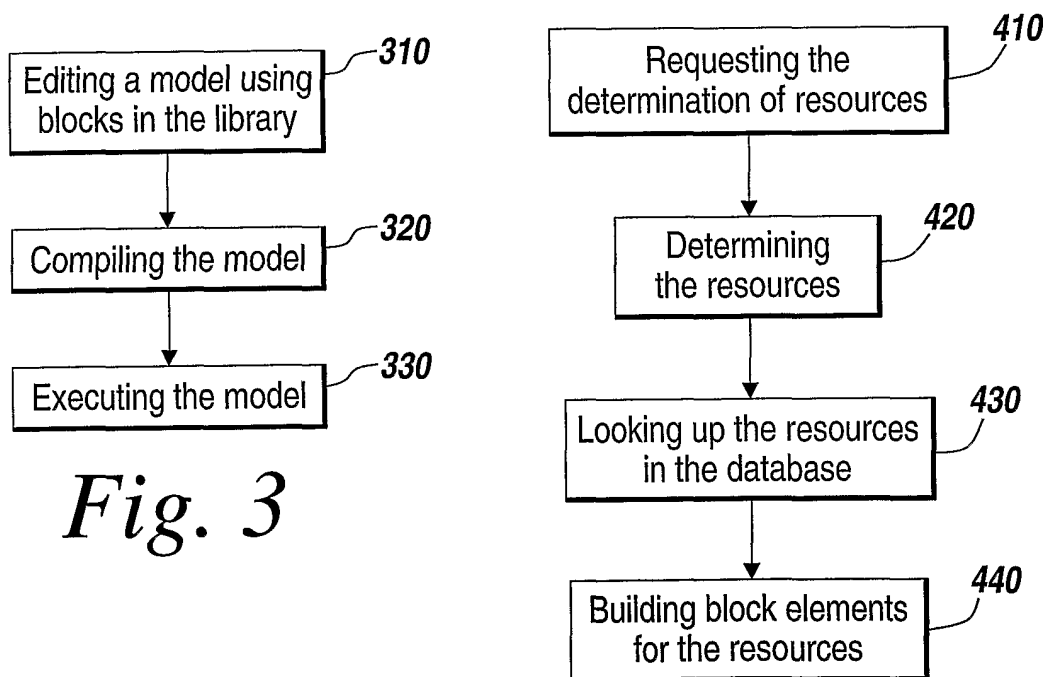


Fig. 1

2/4

*Fig. 2**Fig. 3**Fig. 4*

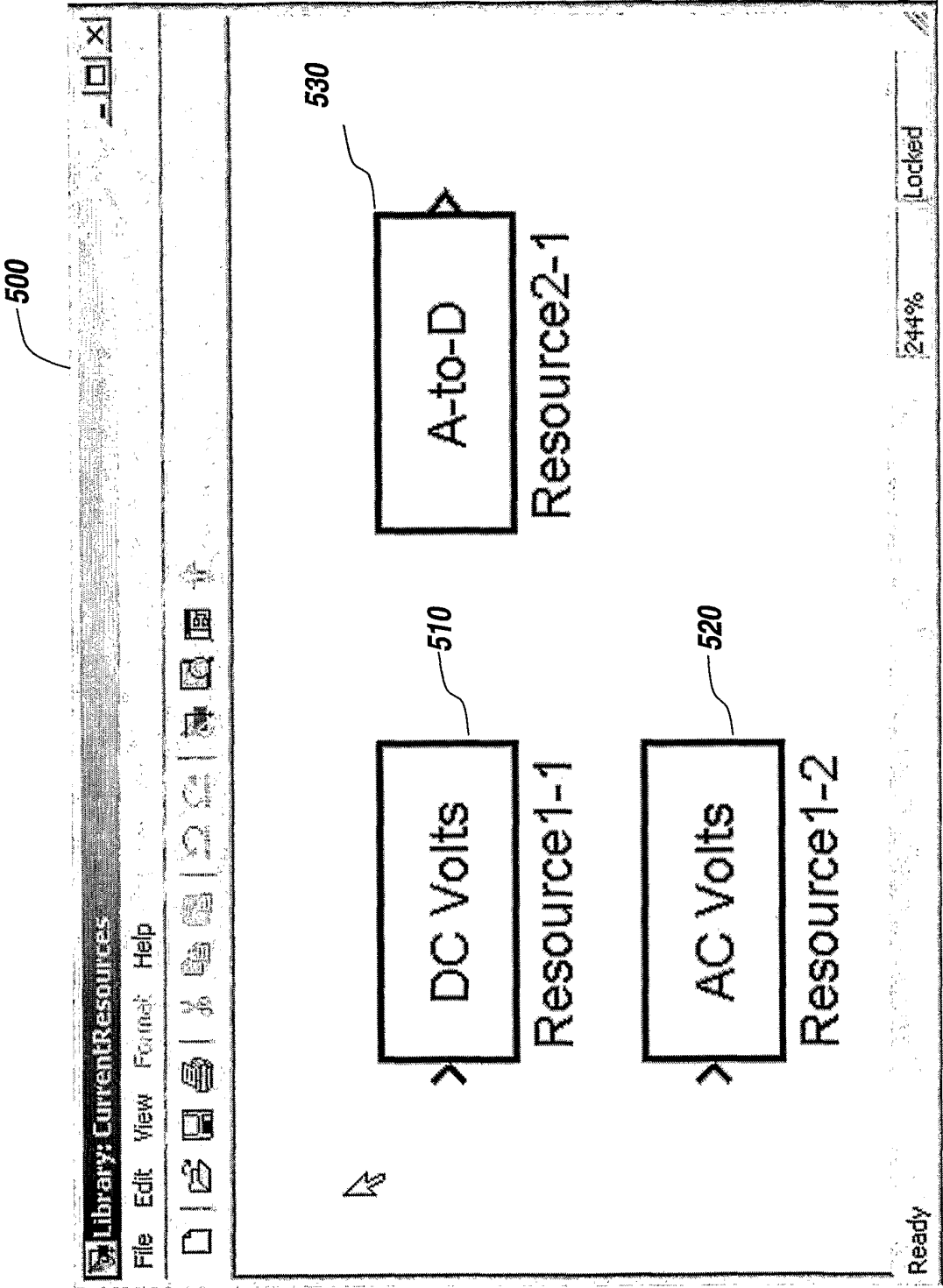


Fig. 5

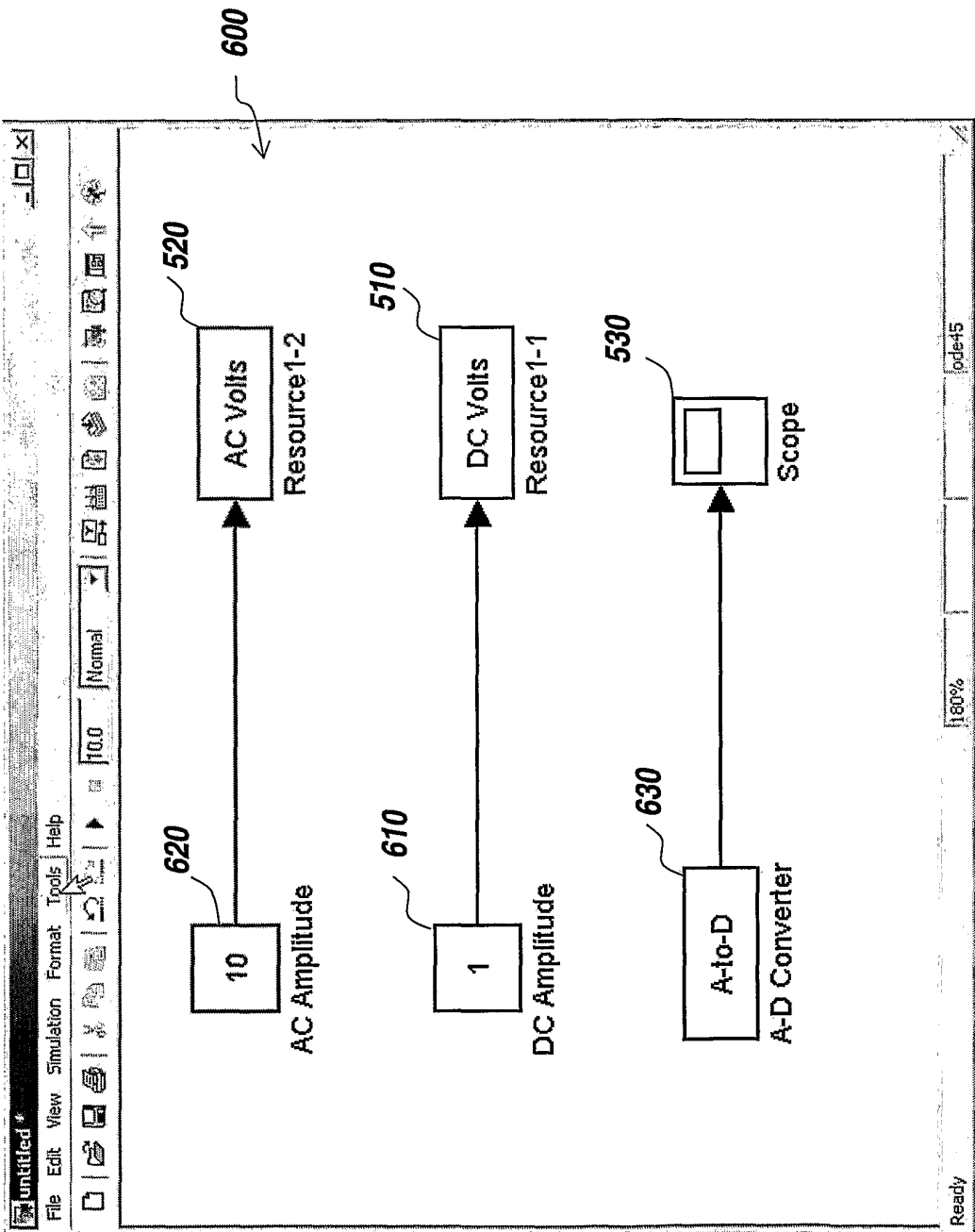


Fig. 6