GEX

SCALD Format Graphics Editor

User Manual & Description

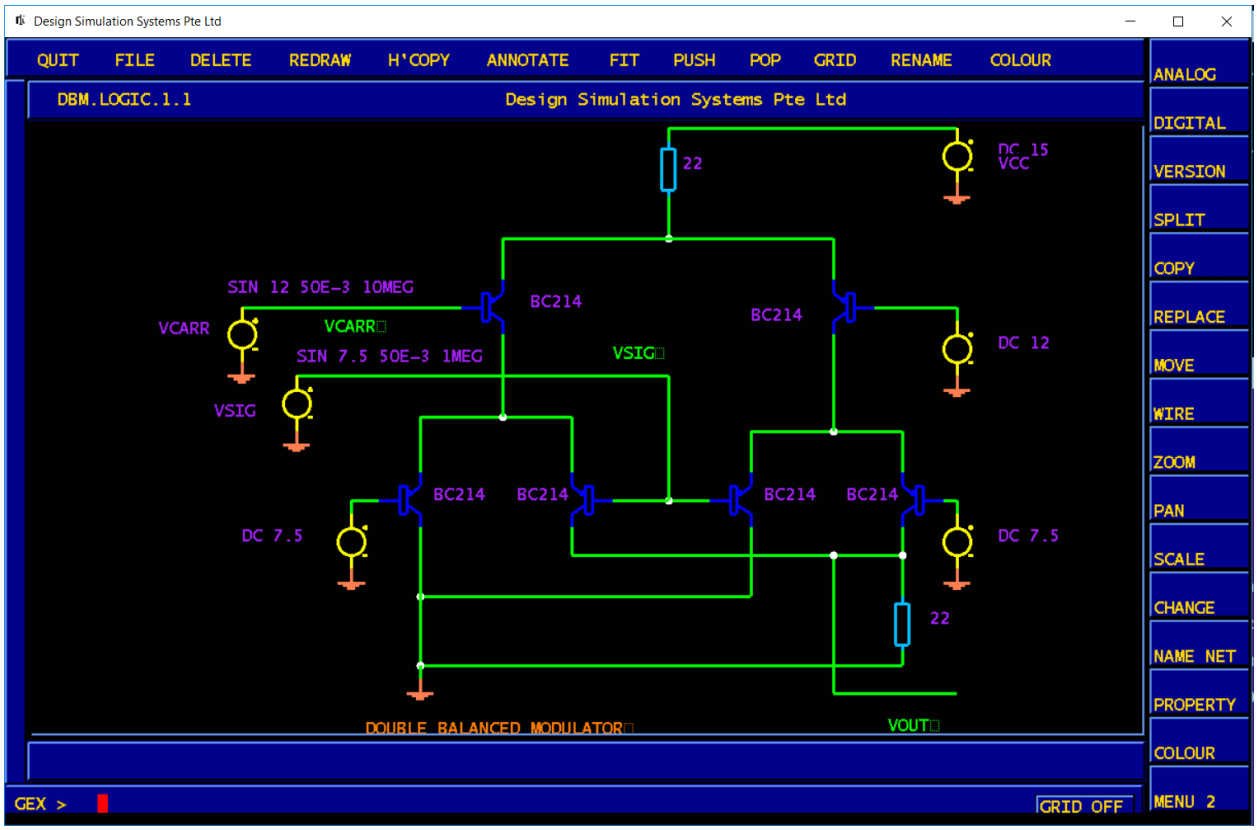


Table of Contents

GEX 1
SCALD Format Graphics Editor 1
User Manual & Description 1
Introduction 4
INTERNAL DATA STRUCTURES 4
ITEM SIZE LIMIT 4
THE SCALD STANDARD 5
OPERATION SUMMARY 10
INVOCATION 10
DRAWING TYPES 10
BODY DRAWINGS 11
CIRCUIT DIAGRAMS 13
COMMAND FILE gex.cmd 13
1. Libraries 13
2. Defaults 14
MENU COMMANDS 15
NOTE: 15
THE MENUBAR 16
* QUIT 16
* FILE 16
* DELETE 17
* REDRAW 17
* HCOPY 17
* ANNOTATE 17
* FIT 17
* PUSH 18
* POP 18
GRID 18
* RENAME 19
The Main Menu 20
ANALOG 21
..... 21
DIGITAL 22

VERSION 23
COPY 23
REPLACE 23
SPLIT 23
MOVE 23
WIRE 24
ZOOM 26
PAN 27
SCALE 27
CHANGE 27
NAME NET 27
PROPERTY 27
COLOUR 29
MENU 2 30
SEE NET 30
SEE PROP 30
SEE ATTS 30
SEE PNAME 30
SEE GROUP 30
SEE BODY 30
CENTRES 30
PASTE 30
GROUP 31
CHECK 31
ARC 32
CIRCLE 32
UNDELETE 32
ADD PIN 32
TRANSFORM 33
MENU 1 34
BUGS AND OMISSIONS 34

Introduction

GEX is a SCALD format Graphics Editor.
It may be used for the creation of circuit diagrams, simulation models and schematic symbols for devices. Its functions have been specifically geared towards the design of analogue circuitry and libraries, but digital circuits may also be created. Although the property definitions are obviously analogue, the VALUE property may be used to enter a delay value on a digital component, and the MODEL property will force the digital netlister to read a standard component definition from a file. Hierarchical designs may be created, to a maximum of 32 levels of hierarchy.

The designs produced by GEX may be used to drive the SPICE simulator by running the SPICE compiler, GSP, which will generate a netlist from a fairly complex circuit diagram in about 2ms on a medium-spec computer.

The digital compiler/netlister, GTL, produces a netlist for Dsim, the digital simulator.

INTERNAL DATA STRUCTURES

GEX stores information in arrays of data structures, each specific to the item stored. This sets the following limits on the numbers of the items listed below which may be included in a single drawing:

ITEM	SIZE LIMIT
components	[512]
wires, pins and dots	[2048]
properties and netnames	[100]
notes	[100]
arcs/circles	[100]
each of above in a group	[100]

The actual properties permitted to be associated with a body are fixed by the elements of the body structure, and are currently:
VALUE, MODEL, PARAMS, LOCATION, PART_NUMBER (for PCB netlisting) and COUPLED (for transformer windings), and these are also used for digital primitive properties. Delay is entered as a VALUE, while the type (7400, 7474, etc) is entered as MODEL. The other properties are the same.

The limit on string length for properties is 255 characters for the PARAMS property, and 30 characters for the rest. Library search paths for bodies are limited to 50 characters, and a body name must be no longer than 30 characters.
Currently, bodies may only have 16 pins which, for analogue devices, is reasonable.

Memory is pre-allocated by malloc at startup time, to make drawing construction and read-in faster, and structure-array length is maintained by malloc/free during

additions/deletions. The total core usage is small enough (10Megabytes) to make this practise acceptable.
There is no other reason why memory isn't allocated on demand.

THE SCALD STANDARD

The following brief description of the SCALD ("Structured Computer-Aided Logic Design") standard may aid the understanding of GEX, and how it applies this standard to analogue design.

SCALD was written to be independent of computer operating systems. It relies on the ability to group files together, in order to define a level of hierarchy, whether the operating system has a hierarchical file structure, or not. Under Unix, each level of hierarchy is a Unix directory, and all such directories are on the same level. Beneath each directory are the files necessary to fully reproduce that level of hierarchy. These comprise at least the following:

body.1.1	schematic symbol
logic.1.1	circuit diagram
logic_cn.1.1	connectivity
logic_dp.1.1	dependency
phys_dat	physical data

The dependency file contains a list of one of each component used in the circuit diagram, together with full Unix paths to the libraries where they may be found. GEX neither reads nor writes a dependency file, relying instead on the contents of its command file, gex.cmd.

If a non-Unix operating system is used, the files are associated by means of extensions. Thus, a design called "fred" would have its files named:

fred.body.1.1
fred.logic.1.1
fred.logic_cn.1.1
fred.logic_dp.1.1
fred.phys_dat

For consistency, the graphics editor always specifies drawings by means of extensions, irrespective of the operating system.
Therefore, to edit the logic.1.1 drawing of a design called "computer", the user types "GEX computer.logic.1.1" or, since ".1.1" is a default, it may be read-in by typing "GEX computer.logic". If it is known that only, say, a spice.1.1 drawing exists, it is permissible to invoke GEX: "GEX computer".
GEX is smart enough to try both files.

The basic idea of SCALD is that designs, and their associated libraries, should have a common format, containing all of the information necessary for all phases of the design, from drawing the block diagrams describing all levels of the hierarchy, to generating paperwork and PCB layout data.

All of this data should be entirely in graphical and ASCII format, so the user can read, understand and be capable of modifying it.

When the user needs to perform any function, such as simulation, timing analysis, parts-list generation, or PCB layout, he runs a "compiler", or "netlister", which reads the graphical descriptions, and makes the appropriate netlist. Thus, no databases are stored, but are generated "on the fly", as they are needed.

Hierarchy is inherent in the format of the drawings, and is handled as follows:

The lowest level of hierarchy for a circuit design is a "simulation primitive".
This is a basic function that represents the level at which the simulator can simulate the circuit. The more functionality the primitive contains, the faster the simulator simulates complex circuits. Some digital simulators can only simulate at gate-level, like AND, OR, XOR etc, while others, like Dsim, have primitives at the level of flipflops and multiplexers.

Analogue simulators work with device-level primitives, such as NPN, MOSFET etc.

There is only one drawing associated with a primitive, and that is a drawing with a ".body" extension. This drawing will contain a graphical description of the shape representing this function on the circuit diagram so, if the simulator understood a primitive called and2 (two-input AND gate), the library entry for this would be a directory called "and2", under which would be a file called "body.1.1". Since the user might need both positive and negative logic representations of this gate, there would be two body drawings, "body.1.1" and "body.2.1". SCALD files always have two extension numbers and, in this case, the second number is used for different shapes of the same function. Thus, "body.1.1" could be used for the European representation of a positive AND function, while "body.1.2" might be the MIL standard shape.

The different body shapes are called "versions" and are selected on the schematic by use of the VERSION command.

Simulation primitives can be used for making higher-level functions, not in themselves understood by the simulator like, perhaps, RAM, ROM or FIFO.
The drawing representing this function would probably be a library model, and would have a ".logic" extension for the digital simulator, or ".spice" for the analogue simulator. This drawing could contain either primitives, or a mixture of primitives and higher-level devices.

Thus, a library entry for a RAM would comprise a directory called "ram" (or whatever its real name was), under which would be at least two files: "body.1.1" and "logic.1.1". The "logic.1.1" file would contain only simulator primitives, while the "body.1.1" file would be the logical symbol.

If a design is created using the RAM, it, too, must have a ".logic" extension.
This extension is used for all higher levels of hierarchy, whether the drawing contains real components, or primitives. A design of a memory board, therefore would be called "ramboard.logic.1.1" in the graphics editor, and would create a directory called "ramboard", under which would be a file "logic.1.1".

If, in turn, the memory board were a part of a larger design, it would need a "body.1.1" file, containing its graphical representation, so that it could be added to a drawing called, perhaps, "computer.logic.1.1"

It is possible that the design, "computer", might be too large to fit conveniently on one drawing. In such a case, SCALD permits the use of "pages" by merely indexing the second digit of the extension. Thus, "computer.logic.1.1" is the first page, "computer.logic.1.2" is the second page, and so on. The compiler uses all pages to create its netlist.

The graphics editor keeps track of hierarchy levels by means of the names associated with nets. When a wire is connected to a component pin, it is immediately assigned a SCALD unnamed net name, constructed as follows:

UN$<page>$<component>$<instance>$<pin>

The parameter "instance" is sometimes referred to as a "path property", for historical reasons, but is merely a linear numbering scheme, which indexes up with each component added.

Thus, UN1NAND2$5P$A, is a wire connected to pin A of the two-input NAND gate with path property 5P, on page 1 of the current hierarchy level. If the user wishes to connect wires across levels, they must have the suffix **"\I",** to indicate an "interface" signal, or one which leaves that level.
It is thus possible to have a signal called STROBE on one level, totally independent of all other signals called STROBE on other levels. However, if the name were STROBE\I, it would imply that all such names were common.

Having said that, the current set of compilers for analogue and digital simulators will take all user-defined names as being global. This means that a signal called STROBE on one level of hierarchy will be assumed to be joined to all other signals with the same name, on all levels, whether it is called "STROBE", or "STROBE\I". Also, it is not possible, with the current version of GEX, to name a drawing as anything but ".1.1" i.e, one page per level.

The SCALD standard caters for the case where there is more than one type of simulator (e.g analogue or digital) on a given system.
Simply by making drawings with different extensions, possibly containing the same schematic, it is possible to run the same design through SPICE, a timing verifier or a digital simulator, simply by invoking the appropriate compiler.
The key is in the fact that each compiler specialises in one simulator, and has built into it a list of primitives permitted for that simulator. It flattens the design down to primitive level, and anything not recognised as a primitive is assumed to be a hierarchical design and expanded.

In the limit, it is possible to have a common library, where this is appropriate, by making each library entry have a common body drawing and as many simulation models as are necessary to cater for all tools.
The ".spice" drawing is used for the default (SPICE) analogue simulator, with other extensions used for the additional simulators.

Hilo primitives can be used to create a ".hilo" drawing, digital primitives are added to a ".logic" drawing etc. The appropriate compiler can then take the information it needs from whichever drawing is associated with the target simulator.

Specifically with reference to SPICE, there is a library called "spice", containing the following primitives common to all SPICE simulators:

npn4 pnp4
npn pnp tnpn tpnp
pjfet njfet pmos nmos
d txline
r c l
resistor capacitor
vsource isource
isourcei isourcev vsourcei vsourcev

It also contains the primitives:

iswitch njf nmf pjf pmf rcline vswitch

which are specific to Spice3D2.

Note that Vswitch, Iswitch and Rcline all need .MODEL entries in a models file somewhere, and a MODEL PROPERTY, like a semiconductor device. These are of the form:

```
.MODEL VSWITCH1 SW VT=2v VH=1mV RON=0.1 ROFF=1MEG
.MODEL ISWITCH1 CSW IT=2a IH=1mA RON=0.1 ROFF=1MEG
.MODEL RCLINE1 K=1.2 FMAX=6.5MEG RPERL=10
```

Additionally, if it is desired to use the Spice3 silicon resistors and capacitors, these, too, must have .MODEL definitions. See Spice3 manual.

Each directory contains only a set of body drawings, since the compiler understands that these are primitives, and inserts the correct data into the netlist. There is significance in the names themselves, in that they alone are recognised by the compiler as primitives.

All higher-level library models are made with the above SPICE primitives which are added to a ".spice" drawing to create the next level of hierarchy.

For example, if it were necessary to have a model of a differential amplifier, a drawing "diffamp.spice" would be edited, to which would be added two "npn" primitives and three "resistor" primitives.

These would be wired up as a differential amplifier and, next, a circuit symbol called "diffamp.body" would be created by means of wires (lines), arcs and text.

This could then be added to any ".spice" drawing as a component.
In practise, the SPICE compiler only expands one level of hierarchy, because most analogue designs are not created hierarchically and, because it is not practical to

simulate excessively large analogue circuits with currently available simulators. The limit for all SPICE-based simulators is 1000 devices, or 1000 nodes, whichever applies.

With all SCALD implementations, attributes of devices which are variables, but which must appear in the netlist, are attached to the specific instance of the device on the circuit diagram, as "properties".

Thus a given resistor might have the property VALUE=1K, a transistor might have the property MODEL=BC109 and a voltage source might have the property PARAMS=DC 12, where PARAMS means "parameters".

A unique property "COUPLED" is applied to transformer windings, to indicate which windings are associated with a particular transformer.

If it is desired to add extra properties to a transistor, like "AREA=n", or "IC=v", these are added as "VALUE" or "PARAMS" properties, or both, with the entire string "AREA=xxx" entered. The same is true of delay line parameters, where a typical entry would be "ZO=xxx TD=yyy", again, as a PARAMS property.

In some cases, it is permissible to interchange VALUE with PARAMS, with no ill effect. However, the compiler, GSP, appends the properties in the order VALUE MODEL PARAMS, so if you need a temperature coefficient on a resistor, it is better to assign the value to VALUE, and the tempco to PARAMS, to keep the order correct.

Note, however, that MODEL is reserved for semiconductors, and cannot be used for anything else.

The LOCATION property is a user-defined name, such as "VIN", "Q201" etc and is important for defining the types of non-linear sources whose transfer functions are either polynomial expressions (SPICE2G6) or equations (Spice3D2).

The location property is also essential for naming inductors which are the windings of a transformer, since that is the only clue as to which inductors are part of the same transformer.

See the GSP manual for a detailed description of both the above.

The CHECK function tests for missing location properties on poly sources, but not on transformers or Spice3D2 sources.

In an analogue design, an NPN transistor primitive may well have the properties MODEL=BC108, LOCATION=Q1. The ability to attach these properties to the body obviates the need to clutter up the library with duplicate entries all of which would have exactly the same body shape, but a different name.
All NPN transistors can thus share a common symbol, while a single file is used to store the SPICE model information for all the different types.

Each library, thus, has a file in it called "models" which contains the specific model information for the components in that library.
The correct .MODEL card for a given device is subsequently extracted by the compiler and placed in the netlist file.
On the assumption that, at some time, the design will be implemented in a physical sense, with real components, the property 'PART_NUMBER' provides a reference to, for example, a carbon film 1/4 watt resistor, or a ceramic disk capacitor. The property is written to the connectivity file, and may be extracted by the PCB netlister, for instance, when it assembles footprint information for layout purposes.

OPERATION SUMMARY

INVOCATION

GEX **must** be started in the directory which contains all the drawings which you want to edit. It will not accept a path to a drawing. Each drawing is a subdirectory, containing body, logic or spice files.

GEX is started as follows:

GEX<CR>

Starts the editor and assumes it is editing a new drawing called "unnamed.spice.1.1"

GEX <drawing>

Starts the editor and reads in the drawing "drawing", which must be in one of the following formats:

- ***Drawing_name***
- ***Drawing_name.body***
- ***Drawing_name.spice***
- ***Drawing_name.logic***
- ***Drawing_name.body.m.n***
- ***Drawing_name.spice.m.n***
- ***Drawing_name.logic.m.n***

Where 'm' is the version number, and 'n' is the page number.

DRAWING TYPES

There are some freedoms associated with the exact format of "drawing_name". GEX is smart enough to realise that there are two principal kinds of drawing you might want to edit.

- If the name you type is of the form: <name>, with no extensions, it'll try to find the directory 'name', and look for logic.m.n or spice.m.n inside it. If that fails, it assumes that it's a new drawing and clears all data structures.

- If, on the other hand, the name is of the form <name.extension>, it tries to find name.extension.m.n and, again, if that fails, it assumes it's a new drawing.
- A name of the form <name.extension.m.n> is searched for unquestioningly, and as before, if it doesn"t exist, a new drawing is assumed.

Legal extensions are "body", "spice" and "logic".

BODY DRAWINGS

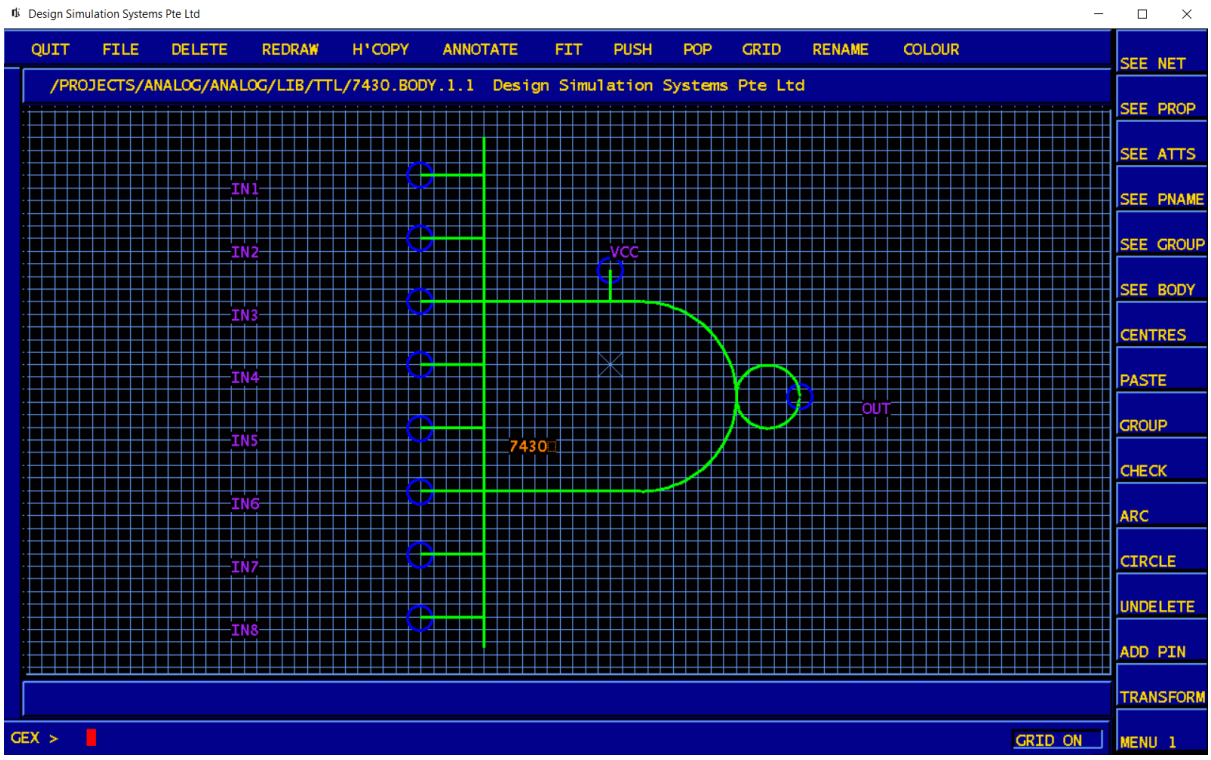
A body is an icon, or schematic symbol, representing a circuit function or hierarchical block. It may be any shape which can be made out of lines, arcs and text, and may subsequently be added, as a component, to a higher level of hierarchy. Each element of the body is stored as a line of text in a file, and is fairly easy to read and interpret. The syntax for each element is as follows:

1. Line
   L x1 y1 x2 y2 pattern colour

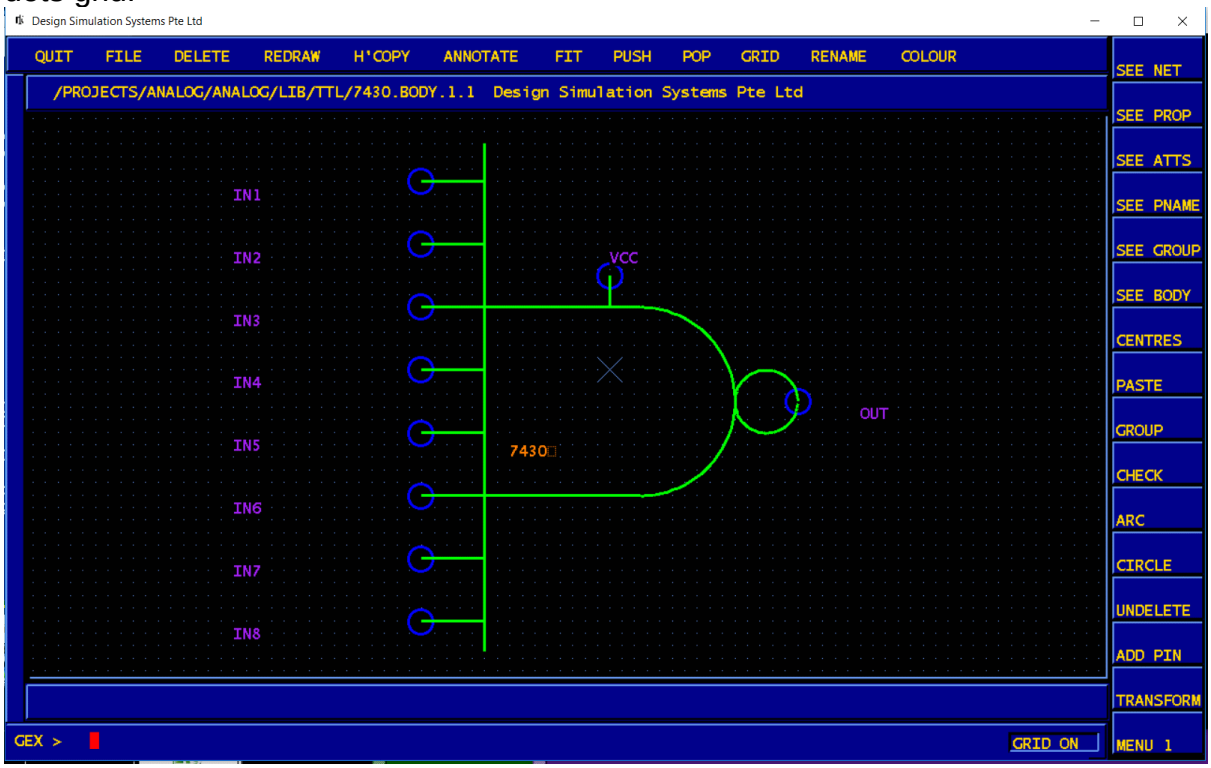
2. Arc and circle
   A xcentre ycentre radius start_angle stop_angle colour

3. Text
   T xstart ystart 0.00 0.00 size 0 0 0 0 number_chars colour
   <text_string>

4. Pins
   C xpin ypin "pin_name" xname yname 0 0 size colour justification



When a body is being edited, the grid is automatically displayed, with the point (0,0) in the centre of the screen, marked by a cross. The cross marks the point of attachment of the mouse cursor to the body, when it is moved across the screen, and the body should be created around it. If it is not easy to see, select CENTRES from Menu2, to highlight it. If the line grid is too distracting, it may be replaced by a dots grid.



For the body to be of any use at all, it must have at least one pin. Pins are attached to wires with the ADD PIN command from Menu2. The pins drawn on the body are open circles of fixed size, with a pre-attached name, of the form UNNAMEDn, where n is an automatically-incrementing integer. This name cannot be deleted, and must be changed with the CHANGE editor (q.v), to the desired name. Note that the auto numbering system resets each time a new drawing is edited, or each time the editor is restarted. This means that, if the pin names are not changed before a drawing is saved, any which are added when the drawing is re-read may have the same names.

In order to permit wiring to the pin, it must lie on a grid-line intersection.
Since the default schematic grid is 0.1 inch, pins should be added using this grid, not the default 0.02 inch grid of body drawings.
If this is not done, the grid will have to be reset to accommodate the non-standard spacing. This can be done in the command file, gex.cmd, but will confuse future users of the component.

Although the SCALD standard permits bodies to have properties (for passing to simulation models etc via the appropriate compiler) this facility hasn't been included. Certain functions are disabled when editing .body drawings. Specifically, the following: VERSION, REPLACE, PUSH, POP, SEE NET, SEE PROP, SEE PINS, SEE PIN NAMES.

CIRCUIT DIAGRAMS

The coordinate space available for the creation of bodies and circuits ranges from (-32000,-32000) to (32000,32000). This is known as "logical space", and an internal mapping algorithm relates this to the number of actual pixels available on the screen ("physical space"). When a body drawing is edited, the logical space origin is set at screen centre, and the grid displayed in four quadrants. For circuit diagrams, however, the logical space origin is set at the lower left corner of the screen, and the grid is only displayed in the first quadrant. The only reason for this is to encourage users to begin their designs in this quadrant.

Circuits are designed by adding components, from the ANALOG or DIGITAL menus, placing them in a suitable position, wiring them to other components, and giving the components appropriate properties. To save time, components and properties may be copied.

COMMAND FILE gex.cmd

1. Libraries

In order to access components from the libraries, whether to display a circuit or to put one together, the paths to the libraries must be included (one per line) in a file called "gex.cmd", as in the following example:

.

/analog/lib/generic
local/directory/lib/spice
/usr/local/lib/analog/alpha

Inclusion of the local directory on the first line (as ".") ensures finding any user-generated parts before searching all the libraries. If this file does not exist, only the local directory and adt/lib/spice will be searched for components. The library "alpha" contains upper-case letters and numbers, which may be used for placing scaleable text on body drawings.

Libraries are searched automatically, in the order specified in gex.cmd, and the first name matching the component name is taken to be the reqested component. The full path is stored internally, to obviate the need to repeat the search if the component is copied or moved.
Although the path is actually specified in the dependency file, this file is not used by GEX, since libraries sometimes get moved around. The file is not currently written either, although the information to do so is present in the internal data structures.

2. Defaults

The other function of gex.cmd is to permit the setting of default colours for various objects.

At present, the following defaults are recognised:

SET COLOR_WIRE <COLOUR>
SET COLOR_BODY <COLOUR>
SET COLOR_NOTE <COLOUR>
SET COLOR_PROP <COLOUR>
SET COLOR_DOT <COLOUR>
SET GRID <pitch> <line_pitch>

These should be entered one per line, with no leading spaces. The colours must be in upper case, and be those named in the colour menu, or the directive will be ignored. For obvious reasons, it is best to avoid colours like BLACK, and DGREY, since these will render the component difficult to see, at best or,at worst, totally invisible.

When a drawing is saved, the defaults are saved with it, in the first few lines of spice or logic.1.1. When it is read and displayed, these settings will override those currently in use. Note, that if the colours of individual objects are changed manually, these colours will also override the default settings, and be saved with the drawing.

The spelling is for backward compatibility with Valid's GED, and all the old Valid colour names are recognised, but mapped to existing colours. Thus, SALMON becomes PINK, AQUA becomes CYAN, etc.

MENU COMMANDS

NOTE:

Any menu command which actually or potentially changes the topology of the circuit, ends by calling a routine which deletes overlapping wires, repairs butt-joints in wires, and re-establishes which wires are connected to which components/wires. This means that the editor will visibly slow down when large schematics are edited, to the point where a command could take up to half a second to complete.

Commands are selected from the menubar, or one of three overlaid menus at the right side of the screen.

THE MENUBAR

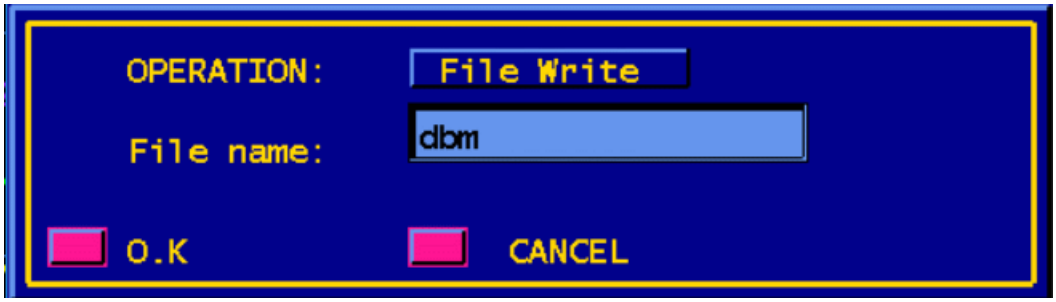
The menubar commands operate approximately as recommended by the Motif standard.

QUIT

Exits gracefully. Prior to exit, it deletes the undo log and the group log from /tmp, and wipes its X11 window ID from the X11 cut buffer. If GEX is crashed, this will not happen, and interactive programs, like Vspice3, will think it is still running, and not attempt to start it.

If this happens, start GEX by hand, since selecting SCHEMATIC from the SETUP menu in Vspice3 will not work.

FILE



The popup requires selection of either WRITE (the startup default) or READ, by pressing the button, and a text entry box contains the name of the drawing currently being edited.
If WRITE is selected and the drawing directory does not exist, it is created. Note that to save typing, it is only necessary to enter the generic drawing name, i.e, instead of WRITE "name.spice", or WRITE "name.body", just enter WRITE "name", since GEX will figure out the type of drawing from the active extension.

Thus, if you are editing "Darlington.body", you write it to the file "Darlington", not "Darlington.body", since this would create a directory "Darlington.body", with a file under it called "body.1.1". However, when reading the file, you need to specify "Darlington.body", since "Darlington" would get you "Darlington.spice.1.1".

DELETE

Deletes components, text or wires from the screen and from the internal data structures. To compensate for this, another structure is requested by "malloc" for each deletion, to keep the list the same length.
Prior to every deletion, the current state of the drawing is written to the undo log in /tmp. This permits a single-level undo to the time of the last delete.
If a wire is deleted which is internal to a net, thus splitting the net, both the remaining sections of the net are renamed. They are also highlighted in two different colours. If it was a user-named net, the original text of the name is left behind (attached to nothing) to remind the user of the original name.

REDRAW

Redraws the screen from the internal data structures. Useful for damage repair if some features become undrawn through moving, wiring etc.

HCOPY

This function creates a file called 'gex.plot' in the Cadence hpr format required by the psplot.gex PostScript driver. This file is converted by psplot.gex into PostScript format, and sent to the port /dev/ttya.
If it is required to preview the hardcopy, it may be saved by performing the conversion by hand, and redirecting the results to a file. This file may be displayed with the Sun utility 'pageview'.

Type 'psplot.gex gex.plot > file.ps' where 'file' is any name. Then type 'pageview -h 11.7 -w 8.27 -left', and adjust the shape of the resulting window to fit the A4 plot in it. There are two shell scripts in adt/tools/bin called 'psv' and 'pgv' which will save you some typing. Invoke them both with the file name as an argument.

There is an interface to HPGL in adt/tools/bin but, at the moment, this needs to be called by hand, to generate a file which may be fed to an HP pen plotter. Type 'hpplot gex.plot > /dev/ttya'

ANNOTATE

This command permits annotation of the drawing with up to 80 characters per note. The text is glued to the cursor, and may be placed anywhere on the drawing. No case conversion is performed on the text.

FIT

The contents of the drawing are scaled to fit the frame. In practise, the largest dimension is made an exact fit, while the other is made proportional to the frame"s current aspect ratio. Note that the algorithm only has information about a body's origin and pins. Thus, bodies should be created with the origin in the centre, and pins on both sides or, if this is impractical, the origin should be on the side opposite the pins.

PUSH

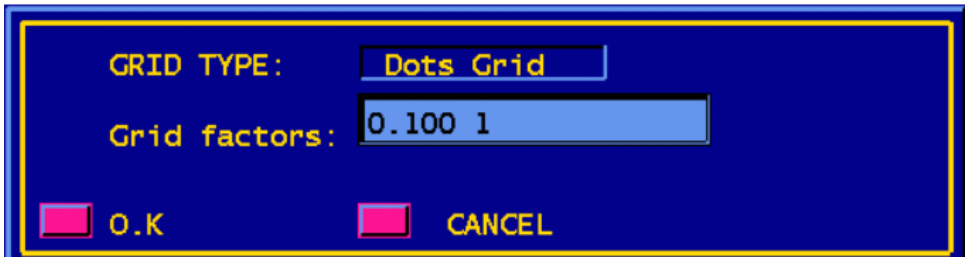
Followed by probing a body on the screen, will push the hierarchy stack and display the circuit diagram hierarchically associated with the body.
Note that if the current drawing is in another directory, and that directory does not contain the drawing of the part, nothing happens. It will be necessary to put a path to the part into gex.cmd.
Also note that the POP operation relies on the drawing being present on the disk. If a new drawing is being created, there will be nothing to return to.

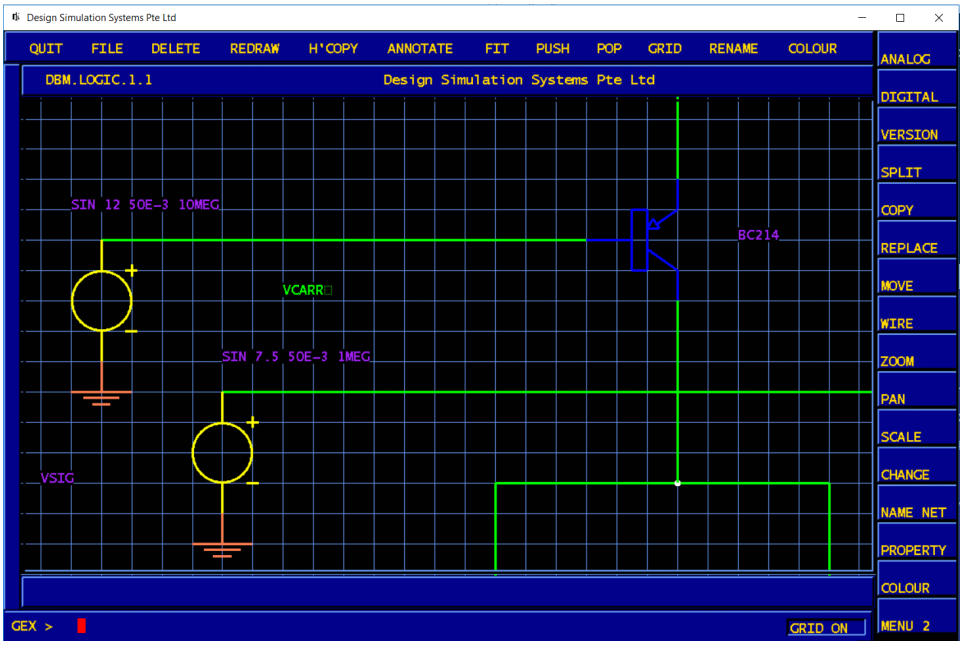
POP

Will pop the hierarchy stack and display the previous drawing. Note that if the current drawing is in another directory, and that directory does not contain the drawing of the part, nothing happens. It will be necessary to put a path to the part into gex.cmd.
Also note that the operation relies on the drawing being present on the disk. If a new drawing is being created, there will be nothing to return to.

GRID

A popup is displayed, offering a choice of LINE GRID or DOTS GRID, with the option of changing the density.





RENAME

Will rename the drawing currently displayed to the name typed in response to the prompt. Note that this will be the name that GEX will use to write the drawing to.

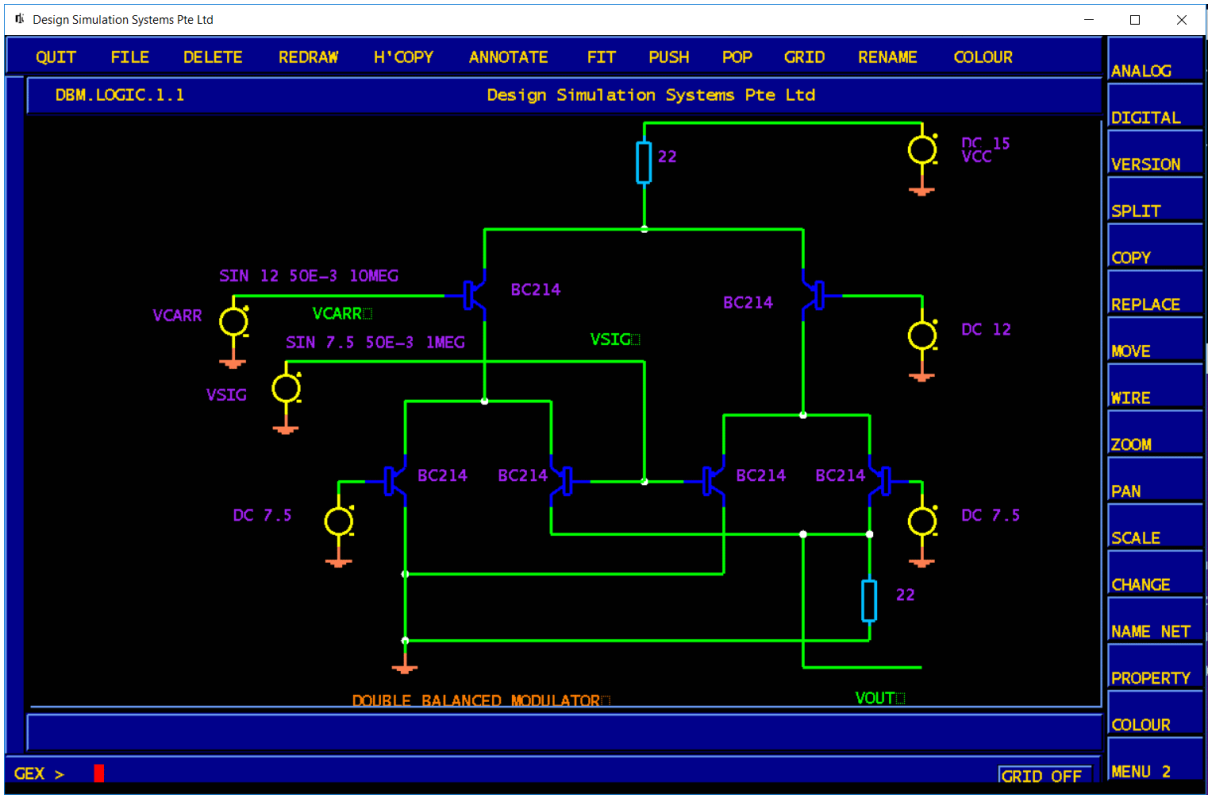
The Main Menu

The main menu is the default menu, situated at the right-hand edge of the screen. It gives access to three submenus:

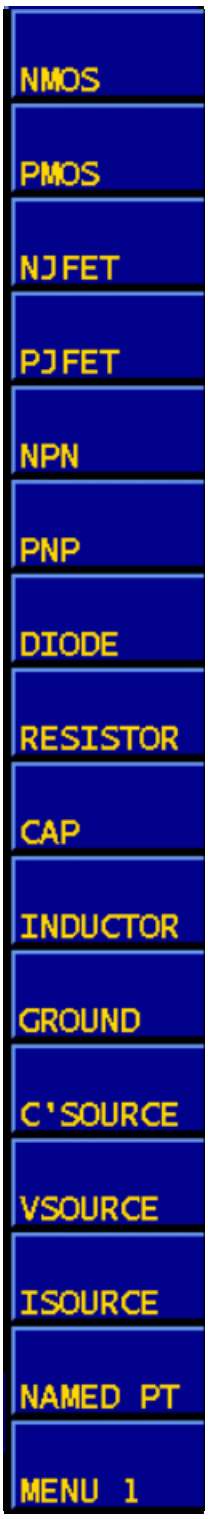
- ANALOG
  These are the default analog primitives supplied with the system

- DIGITAL
  These are the default digital primitives supplied with the system, which are only suitable for the Dsim digital simulator. Errors will result if these are added to an analogue circuit.

- MENU2
  The menu of utility commands for the creation of body symbols, and for circuit topography checking,



ANALOG



Displays the menu of SPICE primitives which may be used to create a
schematic. The list is:

- NMOS
- PMOS
- NJFET
- PJFET
- NPN
- PNP
- DIODE
- RESISTOR
- CAP
- INDUCTOR
- GROUND - every drawing must contain at least one ground symbol
- CSOURCE - popup-selectable controlled sources
- VSOURCE
- ISOURCE
- NAMED PT - enter path to component in text entry box

When using NAMED PT, it is unnecessary to include a path to any library already specified in gex.cmd - typing the part name is sufficient, since GEX will search all the libraries of which it is aware.

DIGITAL

Displays the menu of digital primitives which may be used to create a schematic for Dsim, the digital simulator. The list is:

- INVERTER
- BUFFER
- TS INV - tristate inverter
- TS BUF - tristate buffer
- AND - type the number of inputs, from two to eight
- OR - type the number of inputs, from two to eight
- NAND - type the number of inputs, from two to eight
- NOR - type the number of inputs, from two to eight
- XOR
- XNOR
- JK - JK edge-triggered flipflop with preset and clear
- D-TYPE - D-type edge-triggered flipflop with preset and clear
- LATCH - transparent latch with preset and clear
- MEMORY - 16 bit memory element with enable
- NAMED PT - enter path to component in text entry box

When using NAMED PT, it is unnecessary to include a path to any library already specified in gex.cmd - typing the part name is sufficient, since GEX will search all the libraries of which it is aware.

VERSION

Redraws the component using the next body drawing in the library, cycling from body.1.1 to body.n.1 in numerical order.
When there are no more versions, the sequence starts again at body.1.1
The current library contains flips and rotations, but these may be deleted, and different styles may be included.

COPY

Copies bodies, wires, circles, arcs and text, or groups of components,together with their properties to the spot pointed to by the mouse.
If a property is copied, the cross-hairs follow the cursor to the next button-press, and the body nearest to the button-press point acquires the copied property. If a netname is copied, the target net must be selected.
If groups are copied, properties are always copied with bodies, but never by themselves. Pins on .body drawings are also not group copied.

Copied groups are isolated from the parent circuit, and inherit no connectivity from it. They must be explicitly connected to the parent circuit, NOT merely placed in apparent electrical contact. Use SEE NET to check connectivity.

REPLACE

Replaces the component pointed to by the mouse with the component whose name is typed in the text entry box. All the original properties are deleted, and the replacement component is always version 1, irrespective of the version of the original. Wires attached to the pins of the original component will still have the original names, unless a command is executed which alters circuit topology.

SPLIT

Moves components or text around the screen, without moving the associated properties or wires.

MOVE

Moves bodies, wires, circles, arcs and text, or groups of components around the drawing area.

If the object is a wire, it will remain attached to any other objects to which it is connected. If necessary, additional wires will appear, to create an orthogonal connection (if the wire itself is orthogonal). If the wire is angled, it will move such that a rubberband connection is maintained to its fixed end. Because the method used to draw and undraw the wire prevents it from erasing other drawing elements, the wire may occasionally appear to vanish. This is not a problem, and the wire will reappear when it has been placed.

Circles and arcs move with their geometric centre attached to the cursor. It is worth remembering this if it is necessary to move a small segment of a large-radius circle, and making sure that there is enough screen available to move the arc to its destination.

In the case of bodies, these are moved with their attached wires maintaining orthogonal connections. Associated properties and netnames stay put. If the wires are joined to other wires, occasionally, the interconnecting wire may seem to disappear, but this is only due to the method used for moving the wire.
It reappears when placed.

NOTE: There is only provision for moving three wires with each pin. If orthogonal wiring is used, it is impossible to have more than three wires per pin (one up, one down, one out). However, if angled wires are used, it is perfectly easy to have many of these emitting radially from the pin.

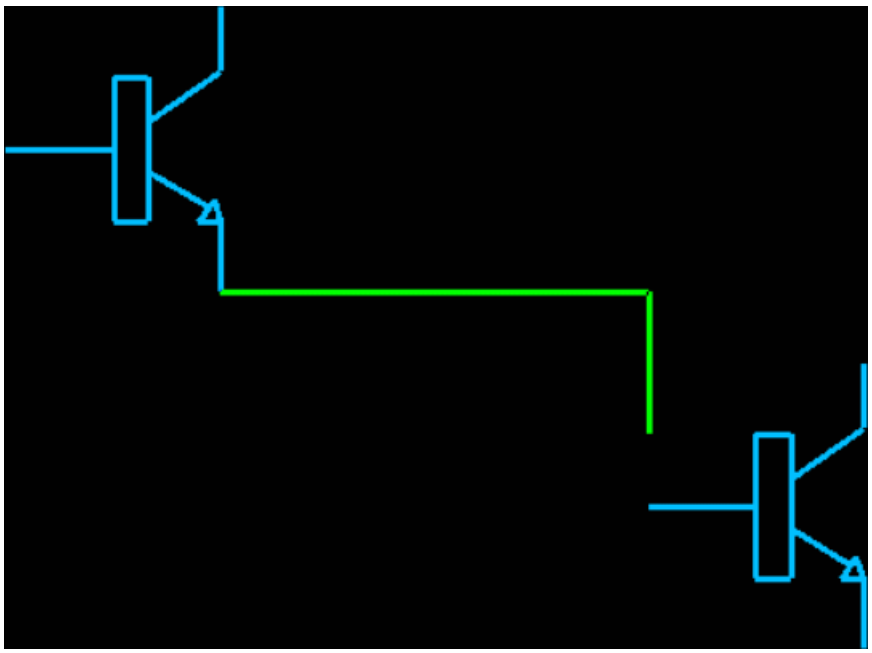
In such a case, only the first three wires found will move with the pin, the rest would remain in their original places, and would have to be moved by hand.

When groups are moved, any wires protruding from the group are disconnected from the end remote from the group and, when the group is placed, must be physically reconnected, or they will become isolated.

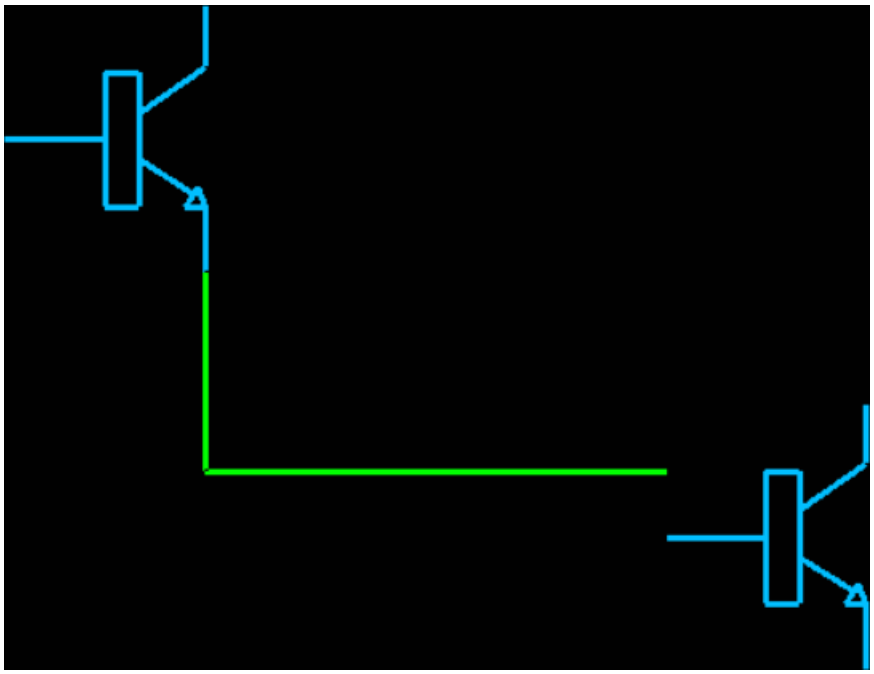
WIRE

This command has three modes. The left mouse button starts the wire, the right button ends it, while the middle button selects the wiring modes.

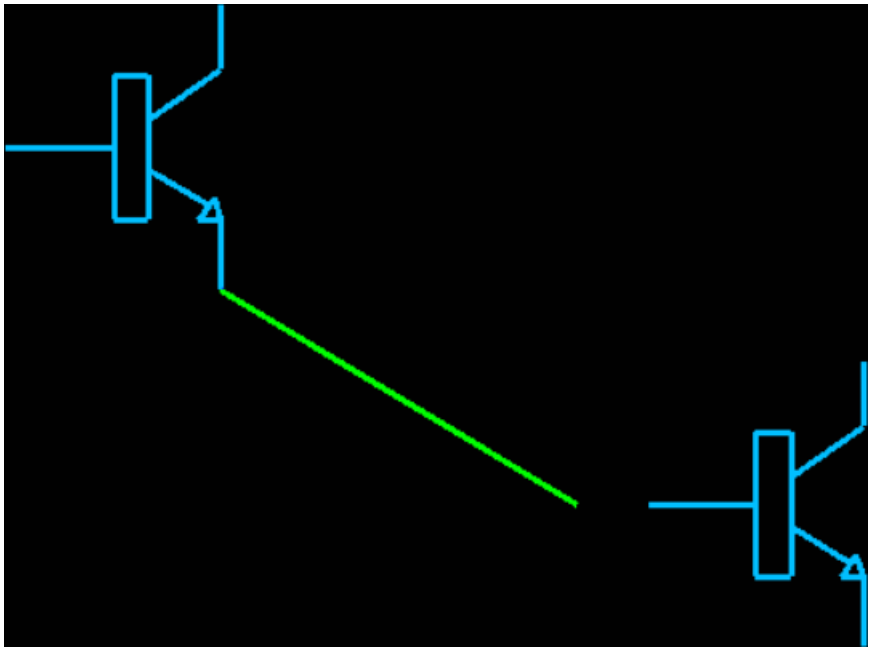
The default is for fixed-vertical-free-horizontal corners.



Pressing the middle button once selects fixed-horizontal-free-vertical corners



while pressing it twice selects rubber-band point-to-point wiring.



The orthogonal wires will always follow a path which is on grid lines, while the rubber-band wires always join two grid points. This means that it is possible to join a rubber-band wire to the mid-point of an orthogonal wire, but not to another rubber-band wire since, unless this latter is at 45 degrees, it will not pass through any grid intersections along its length.

If it is desired to join two angled wires, the same procedure should be followed as when creating a 4-way junction: wire towards the junction, then away from the junction.

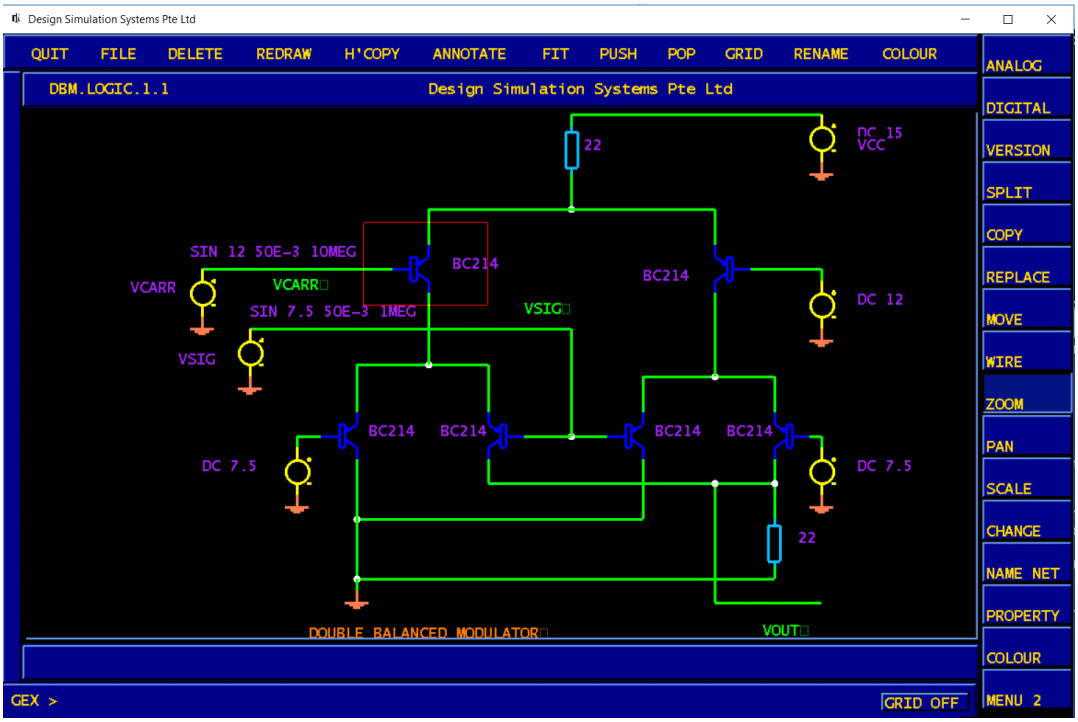
Component pins join wire ends and junctions - nowhere else. If a wire is drawn past a pin, the two will not be connected. Similarly, if a wire is moved towards a pin (or vice versa), such that the pin contacts the wire at a point between the two ends of the wire, the two will not be joined.

After each wiring operation, the internal structures are searched for overlapping wire segments and butt-joints, which are resolved, and T-junctions, which are broken up into separate wires. Then, wire end-coordinates and pin coordinates are matched and netnames propagated through the netlist, to identify separate nets.

This same operation places a dot at each wire intersection point. The dot is a fair indication of whether there is a true connection or not. SEE NET provides absolute confirmation.

Note that, because of these processes, any nets with default names are constantly being renamed, so any name beginning with 'UN$...' is only temporary.

ZOOM



The red cross-hairs follow the cursor to the first button-press, where a rectangle begins to grow. It follows the cursor to the next right-hand mouse button-press. The contents of this rectangle are expanded to fill the frame. In practise, the largest dimension is made an exact fit, while the other is made proportional to the frame"s current aspect ratio.

PAN

The cross-hairs follow the cursor to the first button-press, and the point selected on the screen becomes the new centre. A two-dimensional pan is performed.

SCALE

If a number is typed in the text entry box of the popup, e.g 0.5, the scale of the displayed drawing is multiplied by that factor.

In the event that the scale factor is too high, all text display is turned off. X11 dot-matrix fonts are unscaleable, and the drawing is too cluttered if the text is left on. When nets are being selected for plotting in Vspice3, the GEX window is frequently made very small, resulting in the schematic being totally obliterated by text.

CHANGE

This is a single-line screen text editor for editing properties, netnames and notes which are displayed in the prompt bar, at the bottom of the screen..
It roughly follows the commands used by DEC's edt. It has the following commands:

- **^A cursor to end of line**
- **^B cursor back one character**
- **^C abort edit**
- **^D delete character under cursor**
- **^E cursor to end of line**
- **^K kill line from cursor to end**

- **Typed text is inserted before the cursor position**
- **<CR> terminates the edit and writes the edited text to the screen in the same position as the original.**

NAME NET

Prompts for a text string with which to name a wire. The typed string then glues itself to the cursor, and may be placed anywhere convenient with any mouse button. All netnames are converted internally to upper case.

Note that, with a hierarchical drawing, all non-default names (specifically, those not containing a '$') are treated by the netlister as global names, and are assumed to be the same net. Thus, a signal called 'CLOCK' will be common with all other signals called 'CLOCK' throughout the hierarchy. On the other hand, a signal '$CLOCK' will be treated as purely local, irrespective of the use of the same name on another level.

PROPERTY

The popup has pushbutton-selectable properties, and a text entry box.
The cross-hairs follow the cursor to the first button-press, and the property is associated with the nearest body. The property then glues itself to the cursor, and may be placed anywhere convenient with any mouse button.
All properties are converted internally to upper case.

Selectable properties are:

- **model**: <semiconductor> e.g BC109, IN4148 etc.

- **value**: <passive component value> e.g 1K, 100uF, 2mH, etc.
  Also source value, as in 'DC 12' 'AC 1' etc

- **location**: <part name> e.g RLOAD, Q1, CFILT, VCC, HMULT, EOUT, BDIV.
  This property is essential for non-linear sources, and
  its omission is an error.

- **coupled**: <L1 L2 L3 L4...LN> where L1-LN are the location properties
  of inductors which form the transformer windings

- **parms**: <parameter list> e.g "SIN 0 1v 50Meg", "AREA=2",
  "POLY(2) V1 0 V2 0 0 1 1"

- **part_no**: <physical device type> e.g "CR25" "123-456" "TO92" etc

See the GSP manual for details of location properties on transformers and
non-linear dependent sources.

COLOUR

Selecting COLOUR from the menubar displays the colour menu at the right-hand edge of the screen. Selecting it again returns to the default menu..
The colours of all drawing elements may be changed, but body drawings may need special attention, as no nets are formed from the component wires.
Currently available colours are:

VIOLET
RED
GREEN
BLUE
YELLOW
ORANGE
BEIGE
CHARTREUSE
MAGENTA
SGREEN
CYAN
CORAL
PINK
PURPLE
SKYBLUE
WHITE

MENU 2

SEE NET

Will print the name of the wire pointed to by the mouse, and highlight it, in magenta, on the circuit.

SEE PROP

Will print the properties of the component pointed to by the mouse.

SEE ATTS

Draws a line from every property to the body with which it is associated.
In the case of netnames, the line doesn't necessarily go to the nearest wire of the net, merely to the first wire found bearing that name.

SEE PNAME

Will print the netnames associated with each pin of the selected component.

SEE GROUP

Prompts for a group number. Groups are numbered sequentially as they are created, and this command will highlight (in white) the last group created.
This command is useful if the group is disturbed, by deleting components etc. In such a case, it is best to redefine the group.

SEE BODY

Will print the name, version number and path property of the component pointed to by the mouse.

CENTRES

Draws an "x" at the coordinates of each pin of all components, and at the coordinates of the origins of all components.

PASTE

Permits the pasting into a drawing of a circuit previously cut from another drawing. Note that the cut-buffer is cleared when the editor terminates, so take care not to quit in between cutting and pasting.

Unlike a group COPY, there is no need to select anything. The cut-buffer only contains the last item cut, which glues itself to the cursor as soon as the menu button is pressed.

GROUP

The next two LEFT mouse-button presses define the corners of a rectangle.
All objects within the rectangle become grouped, for the purpose of MOVE, COPY and DELETE operations. In the case of wires, it is only necessary for one end of the wire to be within the rectangle, to be part of the group, while components only need to have their origins included. Properties cannot be included in a group COPY in isolation from their component but, on the other hand, a component is always copied with all its properties.

If the corners of the rectangle are defined with the RIGHT mouse button, (no other will do) then a CUT operation is performed on the contents of the rectangle. This group may now be pasted to any other drawing edited before the editor terminates.

Note that a group stays grouped until a new one is defined, whatever topological changes are made to the circuit. Defining a group will automatically supercede all previous groups.

It is also worth noting that a copied or cut group is an isolated circuit.
It inherits none of its old connectivity, and placing it in contact with a net or a pin does NOT join it to the net or pin. If it is necessary to connect a copied circuit to another, they must be explicitly connected, with a wire, or by placing the wires in contact, then moving one of the wires. GEX always performs a connectivity check after a wire is moved, so this will connect any wires in physical contact.

On the other hand, a moved group maintains all the original connectivity, even if there is no wire connection, until circuit topology is disturbed, e.g a wire is moved. It then becomes isolated. To maintain connectivity, a moved group must be reconnected to any circuitry it should be a part of. Use of SEE NET will confirm connectivity.

To activate a group operation, select MOVE, COPY or DELETE, in the usual way, then select any object within the group with the RIGHT mouse button - not the left one.

CHECK

This function will check your drawing for the following:

- Dangling wires (where one end goes nowhere)
- Unconnected pins
- Bodies with no properties
- Missing GND body
- Missing or incorrect location properties on poly sources

All wires are repainted in white, so as to provide a contrast for highlighting the dangling wires in red. A simple REDRAW restores the original colour.
Any bodies or pins containing errors are indicated with a large yellow 'X', and a summary of the errors found is printed on the prompt bar.

Note that this function is for information only, and the lack of ground, or dangling wires are not necessarily errors. Some models have no internal ground, and most engineers attach dangling wires to inputs and outputs.

On the other hand, the location property is crucial on a polynomial-defined source. For a voltage source, it must begin with 'E' or 'H', and for a current source, it must begin with 'G' or 'F', depending on whether it is voltage or current controlled. No check is made on the syntax of the actual polynomial expression (see GSP manual).

Note that the Spice3 non-linear controlled source is not checked for a location property, since the equation defining the transfer function is inconsistent. However, if the location property of this does not begin with 'B', Spice3 will not recognise it.

The significance of checking wires and pins is to cater for the situation where a wire just fails to connect to a pin, possibly because of different grids used to create the component and the schematic.

ARC

Cross-hairs follow the cursor to the mouse-button entry, which is marked with a cross, and taken as the centre of the circle. The second mouse button entry both defines the radius of the circle, and the start angle of the arc. The last mouse-button entry defines the end angle of the arc.
The angle is calculated anti-clockwise from the second mouse point.

CIRCLE

Will draw a circle with its centre at the first mouse point, and its radius equal to the distance between this and the second mouse point.

UNDELETE

Returns the circuit to the condition before the last DELETE operation.
An undo log is kept in /tmp in the form of the last known state of the circuit. This is updated with each DELETE operation, and selecting UNDELETE will read in the drawing in the log. Note that, since this log is always there, selecting UNDELETE at any time will undo (irrevocably) all changes made to the circuit since the last DELETE. Avoid selecting it accidentally.

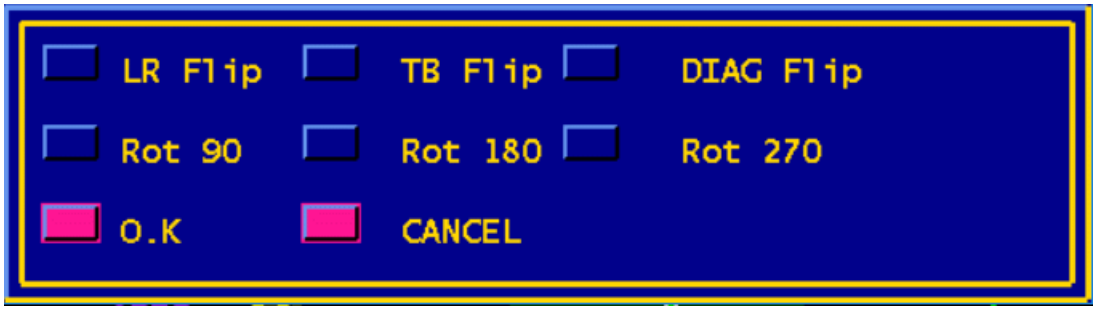
ADD PIN

If a body is being edited, ADD PIN will draw a fixed size circle, to create the component pin. This circle does not appear on the body when it is in use, but is the point to which wires connect. If the pin is created off-grid, or if the grid spacing is such that the pin does not lie on a grid intersection, it will be impossible to make a connection to the pin.

It is best to use a 0.1 inch grid for creating pins, and then to increase the grid density to whatever is necessary to make the fine detail of the body.

At the high magnifications used for creating body shapes, the adding of pins makes apparent the fact that objects 'snap' to grid squares. Frequently, if care is not taken, the pins (and anything else) will appear to go to the wrong point. The placement algorithm ALWAYS snaps an object to either the top or bottom right-hand corner of a grid square, whichever is nearer. It will NEVER go to either of the left-hand corners.

TRANSFORM



The popup allows the selection of all the principal geometric transforms, which may only be applied to bodies. Dot-matrix text on bodies is not transformed, but line text (which is made of wires) is.

Caution:

Transforming a body with wires connected will cause the wires to become disconnected, but the old connectivity to be maintained. Care should be taken when performing this operation, since it can lead to scrambled netlists which are difficult to debug. It is best not to do it but, if you must, check that the net has the same name as the pin, using SEE NET and SEE PNAME.
It won't, so delete the wire and rewire it. Then check again.
Alternatively, use SPLIT to take the component away from the wires, then TRANSFORM, then MOVE to replace it. MOVE will reassign the correct net names to the pins.

The transforms comprise:

- Left-right flip
- Top-bottom flip
- Diagonal (left-right and top-bottom) flip.
- Rotate 90 deg
- Rotate 180 deg
- Rotate 270 deg

Only one transform may be selected at a time, but transforms may be applied sequentially to a body.
If a .body drawing is being edited, applying a transform will cause the entire drawing to be transformed, with the exception of any dot-mtrix text.

MENU 1

Displays Menu 1.

BUGS AND OMISSIONS

Frame doesn't always draw fully on startup. This seems to be a window manager problem, depending on what processes are running. Moving the window forces a redraw and solves the problem.

Text entry boxes on popups don't ignore leading spaces. Take care when entering numbers, which are sometimes stored as strings.

Text may look better with a line font - esp. on hardcopy

There should be a browser for part names (analog and digital)

There should be a browser for existing user drawings

H'COPY needs a popup to select HPGL

Copied wires maintain old net identity if left unconnected (is this a bug?)

A popup would be better for pin numbers of gates

If you use the CHANGE editor for Spice3 transfer functions with an exponent, this gets zapped, since it looks like a '^'. Maybe we'll change the edit cursor to a '|' or something.

Moving an orthogonal wire with an angled wire joined to one end causes the two to part company sometimes. Connectivity is maintained.

Saving a drawing full of wires and no bodies results in scrambled connectivity when it is re-read. The assumption is that wires not connected to anything have no connectivity requirement.