gEDA/gaf

Here is a list of programs, libraries and other things that are part of gEDA/gaf (gschem and friends).

Schematic capture

gschem(1) is the schematic capture program/tool which is part of gEDA. Its sole purpose is to facilitate the graphical input of components/circuits. See the gschem User Guide for more information on the program.

Netlisting

gnetlist(1) is a netlist generation program. It takes as input gEDA/gaf .sch (schematic) files and the required .sym (symbol) files and converts them into netlists. See the gnetlist User Guide for more information on the program.

Attribute editing

gattrib(1) is gEDA's attribute editor. It reads a set of gschem .sch files (schematic files), and creates a spreadsheet showing all components in rows, with the associated component attributes listed in the columns.

See also the Master Attributes List document for more information on attributes used in gEDA/gaf.

Utilities

There are many utilities included as part of gEDA/gaf. See their manual pages and READMEs in the source distribution for more information on them:

- **gaf**(1) is a multipurpose command line utility implementing setting up the above programs, exporting schematics and symbols into various formats, and shell for command line processing of their data. See also this page on the utility.

Utilities for schematics

- **refdes_renum**(1) is a utility for renumbering reference designators in gschem schematic files.
- **grenum**(1) is an advanced refdes renumber utility.
- **garchive**(1) is a utility written in Python used to create and extract gEDA design archives. In archive mode it creates a project archive from a bunch of project files, and in extract mode it extracts the files from the archive and places them in the local directory.
- **gschlas**(1) is a utility which can embed the in-use schematic symbols and pictures into a schematic or set of schematics to put them, for instance, on a website or in an email without the recipient needing to download lots of custom symbols for the design. It can

also unembed symbol references from a schematic.

Utilities for symbols

- **gsymcheck**(1) is the symbol checker for the gEDA/gaf system. Give it a symbol file and it will go through and verify that the symbol will work in gschem and friends.
- **gsymfix**(1) is the utility to automatically fix common cut-and-paste issues with gEDA symbols and fix up the symbols so that they will pass gsymcheck with no errors or warnings.
- **gmk_sym**(1) is a program to create rectangular symbols for gschem from a file composed of comma separated lines.
- **tragesym**(1) is a small python script that creates gschem symbols out of structured textfiles. The aim of this script is to make it easier (and faster) to create symbols for gschem. See the tragesym tutorial and README in the distribution.

Tools to facilitate netlisting

- **gsch2pcb**(1) is a frontend to gnetlist(1) which aids in creating and updating pcb(1) printed circuit board layouts based on a set of electronic schematics created with gschem(1). See also Bill Wilson's gsch2pcb tutorial. Modern pcb also has an import function which uses import from gnetlist directly.
- **gnet_hier_verilog.sh**(1) is a simple shell script which gathers hierarchical information from a list of unique symbols/schematics originating from the top level schematic and produces a hierarchical Verilog netlist in a single file.

Utilities for printing and graphical output

- **schdiff**(1) is a graphical diff tool for gschem(1) schematics. It uses ImageMagick(1) and is most useful when combined with a revision control system (such as Git, Mercurial, and Subversion) so it can compare two revisions of the same file, review changes, etc.
- **gpstoimage** is a shell script that creates GIFs from PS files created by gschem using gs and ppmtogif. It has no documentation and is obsoleted by new gaf(1) and gschem(1) printing capabilities.

Format conversion tools

- **convert_sym**(1) converts a Viewlogic symbol/schematic to gEDA gschem format; there is also **convert_sym.awk** with almost the same functionality.
- **smash_megafile**(1) is a utility that takes a Viewlogic megafile and extracts its contents into a directory, where each element of the library will be represented with one file.
- **olib**(1) is a simple automated converter from OrCAD v4 ASCII parts library to gEDA symbols.
- Sarlacc is an OrCAD to gEDA format converter. It consists of **sarlacc_schem**(1) and **sarlacc_sym**(1). The first utility written in C converts OrCAD schematic files (in 16-bit format) to gEDA format. The second is a Perl script which converts OrCAD text

libraries to gEDA components.

Tools for interaction with other programs

- **gxyrs**(1) is a program written in Perl to batch process XYRS files. XYRS files are usually generated by PCB [http://pcb.geda-project.org] design programs, and are used by board assemblers.
- **sw2asc**(1) is a utility which converts a SWITCAP2 output file into ASCII data files that other tools can read. See the gEDA/gaf Switcap Symbols and Netlister for more information on the program.
- **pads_backannotate**(1) is a Perl program which backannotates changes from Pads PowerPCB board layouts to gschem(1) schematics. See the Forward/Backward Annotation Between gEDA and Pads PowerPCB document for more information.
- **pcb_backannotate**(1) is a program written in Perl which reads an engineering change order (ECO) files generated by the PCB [http://pcb.geda-project.org] program and backannotates the requested changes to a gschem(1) schematics. See the PCB manual [http://pcb.geda-project.org/manual.html] for a complete description of the ECO file format.

Update utilities

These are utilities for schematics and symbols for support of old schematic file formats:

- **gschupdate** and **gsymupdate** are programs written in Perl which update attributes in schematics and symbols of version 20020527 or earlier to use them in modern **gschem**; they are documented briefly in the source code.

Symbols

Symbols (on a schematic) are an abstract representation of the physical components used in electronic circuits. Initial gEDA installation provides you with default symbols contributed by many users. See also the gEDA/gaf Symbol Creation document on how to create your own symbols and http://gedasymbols.org [http://gedasymbols.org] to find already available symbols shared by other gEDA users.

Libraries

- **libgeda** is a main gEDA library of functions for manipulating gEDA schematics and symbols which is used by many of the above mentioned programs and utilities. See the gEDA Scheme Reference Manual (*info geda-scheme*) for more information on the library.
- **libgedacairo** is a library which provides a renderer for schematics and symbols based on the Cairo vector graphics library and the Pango font library. Data for rendering is loaded using libgeda. See the libgedacairo/README in the gEDA/gaf distribution for more information.

Examples and other documentation

gEDA/gaf contains some examples and other documentation which can be found in the documentation installation directory of your distribution.