vtags user guide

vtags is a gvim plugin, it's function similar like verdi's trace verilog signal function, this plugin writed by python, currently support: trace signal， macro define trace，show module's topology， quick access， save and opend a snapshort, add checkpiont etc.

support function and shortcut key:

Key	function
	Build vtags.db
vtags	Build vtags data base at current dir and include design at current dir and it's sub dir
-v <design_file>	Specify signal design file
-f <design_file_list>	Specify design file list and direct use vcs and verdi file list
+vtags_incdir+ <dir_path>	Add dir and it's sub dir's design files into vtags data base
+incdir+ <dir_path>	Same like vcs/verdi , and include file search path
	Function in Gvim/VIM window
gi	go into submodule
gu	go upper module
mt	Print the module call trace from top to current module 【3.00 +】
ct	Clear vtags module call trace(used when gi/gu ...) 【3.00 +】
<Space><Left>	trace source
<Space><Right>	trace destination
<Space><Down>	roll back
<Space><Up>	go forward
<Space> + v	Show the sidebar, fold or unfold module's topt Dynamic build vtags data base for current file and file dir 【3.00+】
<Space> + c	Add check point
<Space> + b	Add base module
<Space> + d	Delete check point or base module
<Space> +h	Hold current window
<Space>	Fast access: 1. Sidebar module 2. Trace Report 3. mt command report 【3.00+】

<Space> + s	Save snap short
<Space> +r	Reload snap short at current window【3.00+】
<Space> + q	Close all open gvim windows【3.00+】

Function when Gvim/VIM opening

gvim/vim <vtags.db path>	Reload snap short saved at <vtags.db path>【3.00+】

Offline function base on vtags.db

vtags -db <vtags.db path> “mtrace(<module_name>)”

At *<vtags.db path> corresponding design, list <module_name> all the call trace from most top module to <module_name>*

vtags -db <vtags.db path> “mfilelist(<module_name>)”

At *<vtags.db path> corresponding design, list <module_name>'s used file list*

vtags -db <vtags.db path> “mtopo(<module_name>, depth, mask)”

At *<vtags.db path> corresponding design, list module call instance topo*

depth optional parm, topo depth

mask optional parm, when sub module call more than mask times not list it one by one

vtags -db <vtags.db path> “mopen(<module_name>)”

At *<vtags.db path> corresponding design, open <module_name> file*

vtags install

(1) tar -zxvf vtags-3.00.tar.gz // unpack the file to the path you want to install. we assuming to "~/"

(2) in "~/.cshrc" add line: alias vtags 'python ~/vtags-3.00/vtags.py'

or "~/.bashrc" add line: alias=vtags 'python ~/vtags-3.00/vtags.py'

(3) "source ~/.cshrc" or "source ~/.bashrc"

(4) in "~/.vimrc" add line: source ~/vtags-3.00/vtags_vim_api.vim

(5) config vtags-3.00/vim_glb_config.py as it decribe in vim_glb_config.py. (can do it in local_config.py see below)

vtags use

1 generate vtags.db database

when finish install vtags, first need generate vtags database for code you want use.

(1) "cd" to code dir

(2) use command "vtags"

then will generate a vtags.db dir, in the dir include module database , log file, and local config files.

3 vtags vim config

config file：

vtags's config file at two position, global config and local config. when gvim open file and valid open vtags, will use local config first, if not local config then will use the global config.

(1) global config：

at the vtags's install dir, we can config vtags in vtags_glb_config.py.

(2) local config：

at code dir use cmd: "vtags" will generate vtags.db folder, in this folder has a file vtags_local_config.py we can config vtags in this.

config parms：

frame_window_width = 30

set the gvim left sidebar width, this sidebar used to show topo checkpoint and base module.

report_window_height = 10

set gvim bottom report window height, this window used to show the report and signal trace result.

max_open_work_window_number = 1

when vtags need open new windows(such as : quick access, trace crose module ...), this number is the max window num opened, if opened window already reach this number, vtags will close a old window and then open new file.

max_roll_trace_depth = 1000

use <Space> <Down/Up> max history depth.

max_his_check_point_num = 1000

max checkpoint can rember, added by <Space> + c

base_module_threshold = 200

when code generate vtags.db, if some module instantiate times bigger than

base_module_threshold will auto set it to be base module, when show the topo will not show the base module instance one by one.

support_verilog_postfix = ['v']

all the verilog code file postfix used, default only "v", you can add your self postfix like support_verilog_postfix = ['v', 'your_postfix0', 'your_postfix1' ...], all the file below code dir has postfix in support_verilog_postfix will treat as verilog code.

debug_mode = False

this mode used for develop， will print lots of report, suggest set to False.

dynamic_update_vtags_db = True

When file modify dynamic refresh vtags db