Introduction

PCB Editor has the ability to be customized to suit user preference. Many commands and groups of commands can be grouped together using a programming language called skill. Many users write skill programs to aid during their design process. To run skill files, you normally load the skill file and then run it by entering the name at the command line in PCB Editor. You can auto load the skill files using an allegro.ilinit file. Please see "How to auto load Skill" for further information. You can also define a "User Menu" so that these commands run the same way that a standard function will run e.g. Edit > Move. You also have the ability to add keyboard shortcuts to activate commonly used commands e.g. instead of Edit > Rotate, you can define a key so that you can press "sr" and invoke the Rotate command. Details and examples of function keys are described in this document.

How to define function keys (shortcut keys) in PCB Editor

You have the ability to define keyboard "shortcuts" to run commands in PCB Editor. To define these, locate a file called env located in your pcbenv directory. (If you do not know where this is located type %HOME% into the address bar in Windows Explorer followed by a return). The takes you to your HOME folder. In here is a pcbenv folder and in here is the env file. Open the env file in a text editor and add any of the following (changing the shortcut keys if required) above the section below because any edits below this section will not be kept.

```
### User Preferences section
### This section is computer generated.
### Please do not modify to the end of the file.
### Place your hand edits above this section.
```

To understand the "actual" command that PCB Editor uses, at the command line in PCB Editor type scriptmode +e. Any command invoked after this will be listed (printed) on the command line.

Note # anywhere in the env file define that any items after this are comments.

```
funckey n pop neck # during add connect n will go into neck mode.
funckey r iangle 90 # during a command that supports rotate r will rotate 90 degrees.
funckey d "delete; pick_to_grid -cursor; done" # delete using the d key.
funckey f "prepopup;pop dyn_option_select 'Snap pick to@:@Figure'" # RMB snap to function for figure.
funckey i "prepopup;pop dyn_option_select 'Snap pick to@:@Intersection'" # RMB snap to function for I/S.

funckey a "prepopup;pop dyn_option_select 'Snap pick to@:@Arc/Circle Center'" # RMB snap to function for C
funckey v "prepopup;pop dyn_option_select 'Snap pick to@:@Via'" # RMB snap to function for via.
funckey + subclass -+ # use + to change active subclass
funckey - subclass -- # use - to change active subclass
funckey m "pop mirror" # mirrors symbol during place manual
funckey " " "pick_to_grid -cursor;pick_to_grid -cursor" # use spacebar to minimize connections
funckey c "zoom center;pick_to_grid -cursor" # centres screen on cursor position
funckey t toggle # toggles direction of route
```

For the next 8 funckeys the layer names MUST match for the functions to work.

```
funckey 1 options subclass TOP # assigning numeric keys to change active subclass to TOP.
funckey 2 options subclass SIGNAL_2 # assigning numeric keys to change active subclass to SIGNAL_2.
funckey 3 options subclass SIGNAL_3 # assigning numeric keys to change active subclass to SIGNAL_3.
funckey 4 options subclass SIGNAL_4 # assigning numeric keys to change active subclass to SIGNAL_4.
funckey 5 options subclass SIGNAL_5 # assigning numeric keys to change active subclass to SIGNAL_5.
funckey 6 options subclass SIGNAL_6 # assigning numeric keys to change active subclass to SIGNAL_6.
funckey 7 options subclass SIGNAL_7 # assigning numeric keys to change active subclass to SIGNAL_7.
funckey 8 options subclass BOTTOM # assigning numeric keys to change active subclass to BOTTOM.
```

The next two are really quick ways to find Components by RefDes or Nets by netname. Use CTRL+F for Components and CTRL+N for nets, then simply type the refdes or netname required. The component or net is highlighted and the item is zoomed to and centred.

```
alias ~F "prepopup ; pop dyn_option_select 'Selection set@:@Clear all selections';set prompt;prompt 'Find Ref Des';refdes $prompt;zoom selection"

alias ~N "prepopup ; pop dyn_option_select 'Selection set@:@Clear all selections';set prompt;prompt 'Find Net Name';net $prompt;zoom selection"
```

How to Define a User Menu in PCB Editor (preferred method using skill)

Below is some sample skill that will build a custom menu in PCB Editor regardless of version, so you do not need to worry about menu changes made by Cadence between releases. This skill can be pasted into a skill file (e.g. menu.il) and stored in a directory specified by the allegro.ilinit file. The skill is then auto-loaded by PCB Editor and the menu is built. The menu items contain additional skill files that can be located either by contacting Parallel Systems support or from the following website: - http://cadence.com/community. The Menu items can be modified to suit. This menu is called User-Defined; the next 8 items are skill commands.

```
; The following demos additional Allegro script capabilities:
;   - command chaining via the ";" allows multiple commands on 1 line
;   - "scriptmode +i" to make forms invisible; scriptmode value restored at
;     end of command chain
;   - "toggle" value for checkbox form fields will switch the checkbox state
;   - "form_next" for enum fields (or checkboxes) will switch to next enum
;     item (form_prev also available). Change "FORM" to form_next and
;     remove value
custMenu = '(
      (popup "User-Defi&ned")
            ("&LogoMaker" "logomaker")
            ("&Height" "upd_fe_height")
            ("&SMD Pin Count" "smd_count")
            (separator)
            ("&Missing Paste/Solder Pad" "smd pad check")
            ("Fin&d Component" "find component")
            ("Find &Net" "find net")
            ("Change Net of Vias" "change_vianet")
            ("Assign Dummy Nets" "cns_dummy_net")
            )

procedure( menuTrig1()
    ; add a new menu item before the help menu
    res = axlUIMenuRegister(-1 custMenu)
)

; To see axlUIMenuRegister uncomment the following line
menuTrig1()
```

Alternative method for User Defined menu (not recommended)

IMPORTANT – Because the software is using a User Defined menu structure any new Hotfixes or releases will NOT be shown until the menu file from the install directory is copied and any user edits re-applied.

Make a directory on your computer i.e. c:\local_cadence which you can use to copy any installed files to. Inside this directory generate a folder called menu. To generate a user defined version of the menu go to the following

directory on your computer <your_inst_dir>\share\pcb\text\cuimenus and copy the allegro.men (Allegro) or orcad.men (OrCAD) to the c:\local_cadence\menu folder.

Once this is done you can add your user edits to this new menu file. Open the allegro.men or the orcad.men using a text editor. The example below adds a user menu called User-Defined and has 4 skill commands as a menu pick. The actual skill command is the last part of the menu item i.e. logomaker.

```
POPUP "User-Defi&ned"
BEGIN
MENUITEM "&X Section Block",   "XSection2"
MENUITEM "&LogoMaker",         "logomaker"
MENUITEM "&Height",            "upd_fe_height"
MENUITEM "&SMD Pin Count",     "smd_count"
END
```

The & before a letter acts as an ALT Keyboard shortcut i.e. ALT N L would invoke the Logomaker command. Once you have finished with your edits save the file and then launch The PCB Editor tool. Under Setup>User Preferences>Paths>Editor as shown below. Add a new menupath to your location above (c:\local_cadence\menu). Using the arrows move your new menu to the top of the list. Restart the software and your menus will be there.