Beginners manual for Cadence

Starting the Cadence for the first time.

```
vahvel:/home/elmet>cd testcad
/home/elmet/testcad
vahvel:/home/elmet/testcad>ams_cds -mode fb -tech csx
ams_cds: Command not found.
vahvel:/home/elmet/testcad>cad
Select your CAD tools:

 1) CADENCE
 2) MENTOR
 3) SYNOPSYS
 4) CADENCE & SYNOPSYS
 5) MENTOR & SYNOPSYS
 0) no selection

----> 1


Environment ready for Cadence.

vahvel:/home/elmet/testcad>ams_cds -mode fb -tech csx
icfb : HIT-Kit=3.40 tech=csx tool=artist
```

Drawing 1: Cadence initialization (terminal window)

Follow these steps [Drawing 1]:

- create new subdirectory (use 'mkdir' command for this purpose)
- change working directory to this subdir (command 'cd <dirname>')
- run environment setup script (command 'cad') and choose the package you'll going to use
- run Cadence as follows: 'ams_cds -mode fb -tech csx'

As you can see from the picture, trying to run cadence without 'cad' command doesnt work.

For using newest version (if you feel like experimenting) then do this (until the newest installation becomes system default):

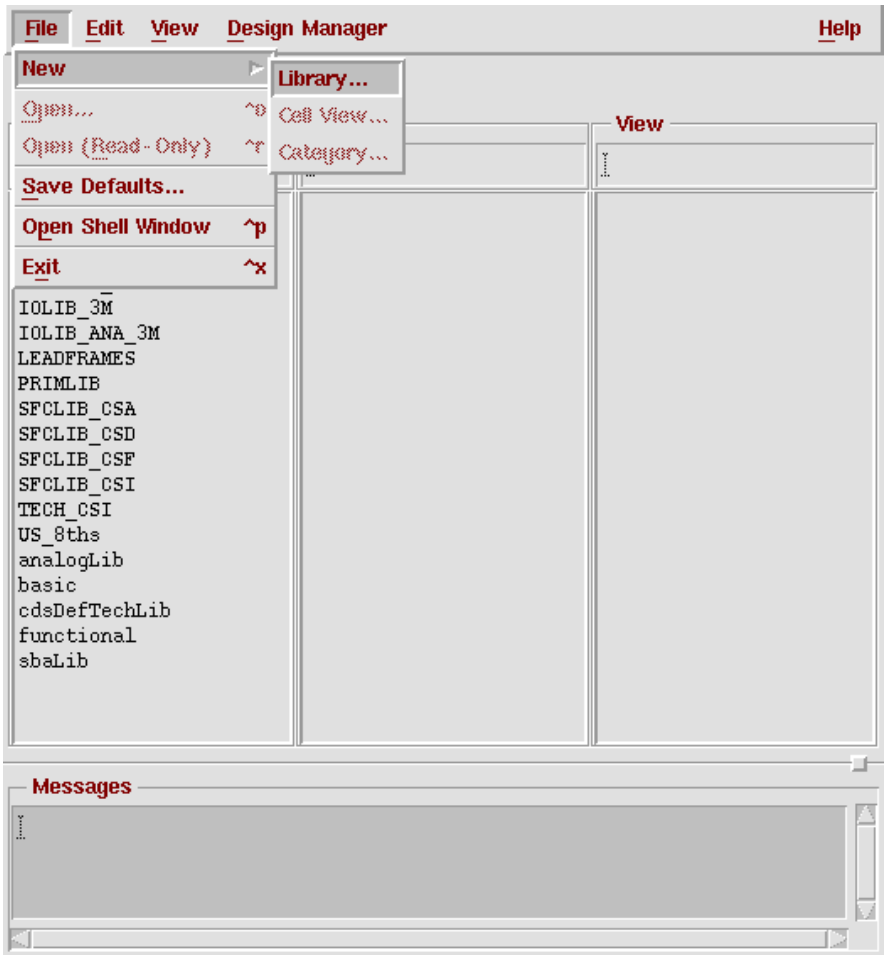
- first two steps are the same
- do 'setenv CADENCE_05' OR 'export CADENCE_05' (this depends on the shell you're using, if one doesnt work – try another)
- let system script set your environemnt variables for the use of Cadence (command 'source /cad/cadrc.include')
- run Cadence as follows: 'ams_cds -mode fb -tech cxq' (note the difference in technology selection option – HRDLIB may and will be different for each tech) and select 'CXQ' from the technology selection dialog.

If all is well - several windows will pop up, windows named 'icfb' and 'Library Manager' among those. Cadence Graphical User Interface (GUI) has been under very conservative development so there is'nt much difference between versions.

Starting to work on new circuit

You'll probably have to create new library (for storing circuit modules) and at least one cellview (selected circuit module). Follow these steps:

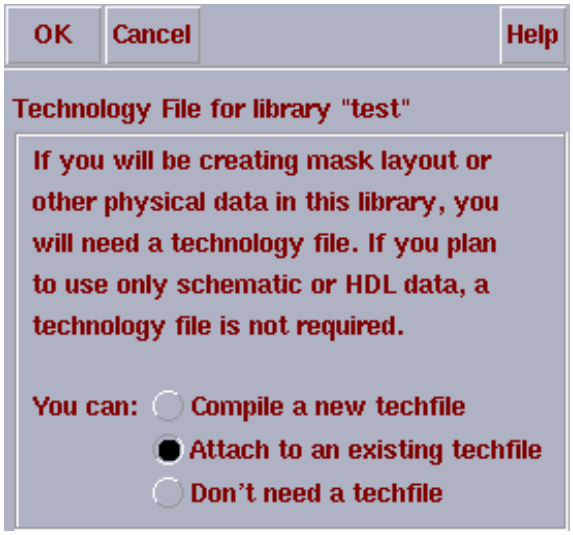
- create new library using 'Library Manager' window [Drawing 2]



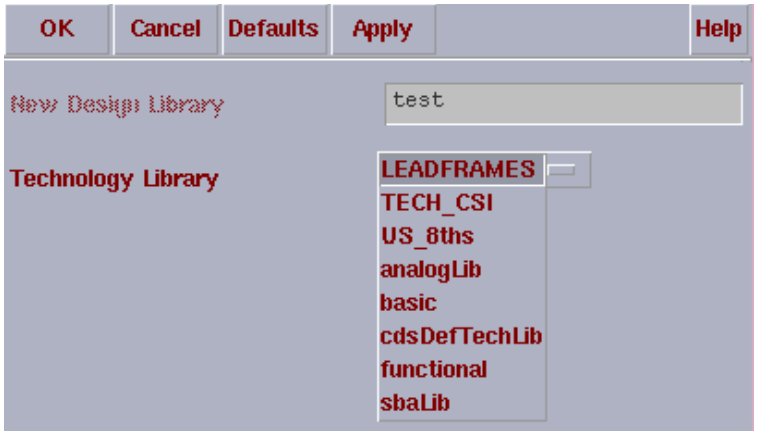
Drawing 2: Library manager - new library

- define new library name and choose techfile [Drawing 3, Drawing 4, Drawing 5]

Drawing 3: Create new library



Drawing 4: Technology selection, step 1

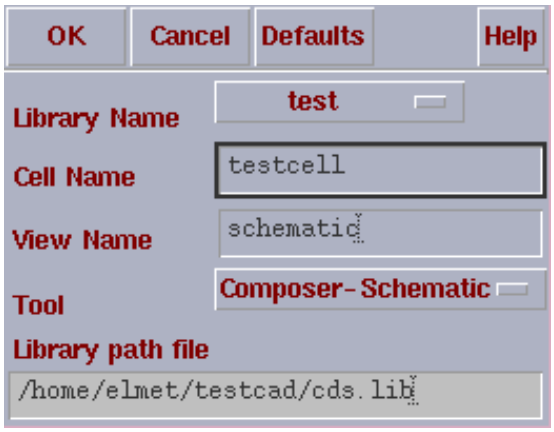


Drawing 5: Technology selection, step 2

Choose appropriate technology, TECH_CSI or cdsDefTechLib.

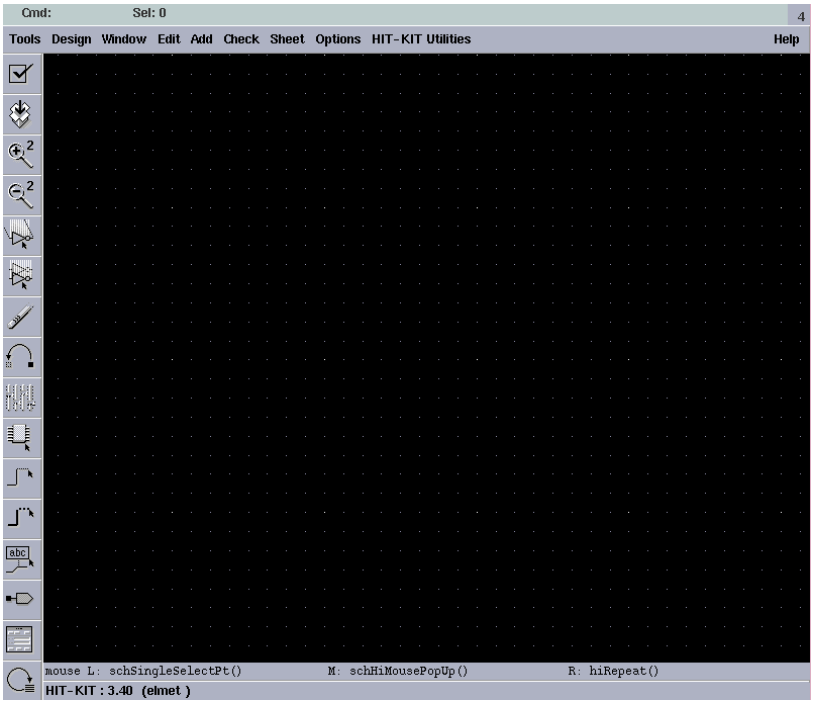
- create new cell view (open menu on the Library Manager window and enter cell name, as seen on [Drawing 6, Drawing 7]

Drawing 6: Creating new cellview, step 1



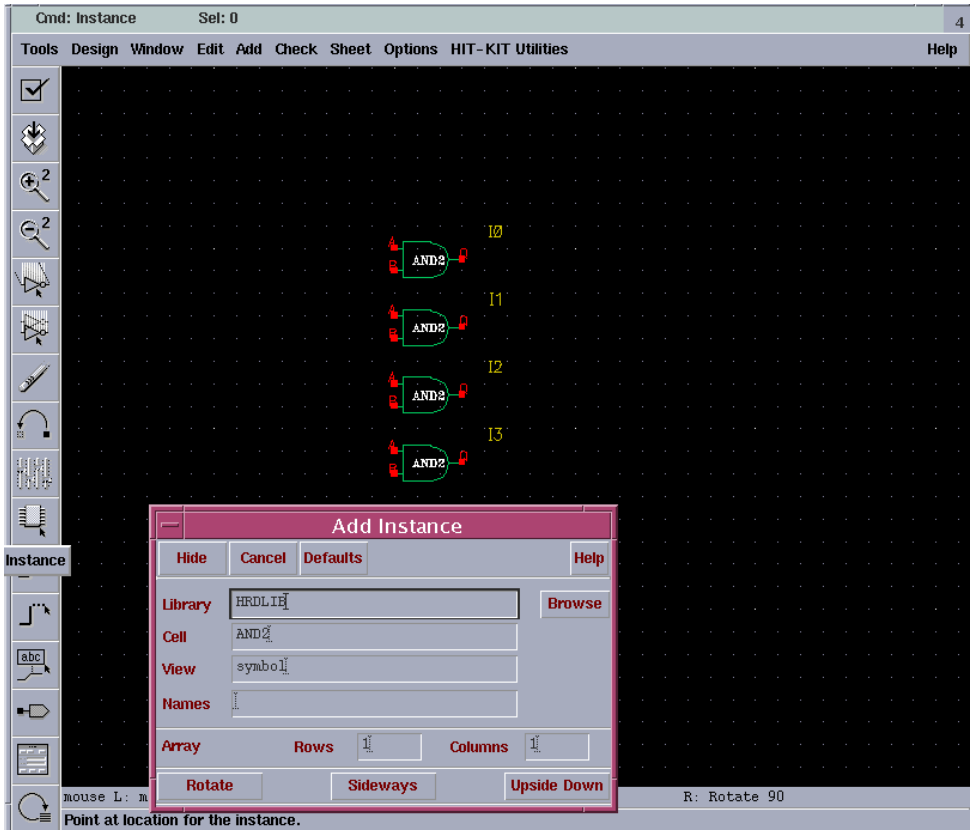
Drawing 7: Creating new cellview, step 2

After this, circuit editor window appears [Drawing 8]. You can now start working on your design. Preferred work order should be following:



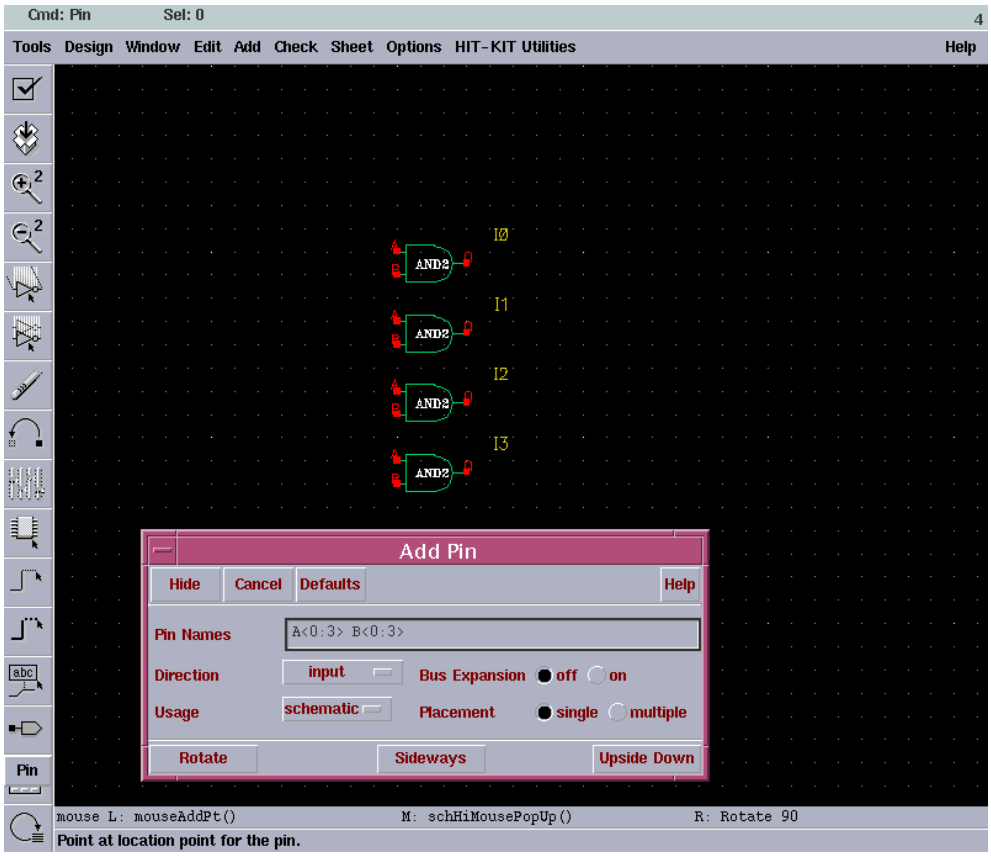
Drawing 8: Editor window

- place instances [Drawing 9], instances should all stem from library 'HRDLIB'. Keyboard shortcut for this is 'i'.



Drawing 9: Instances

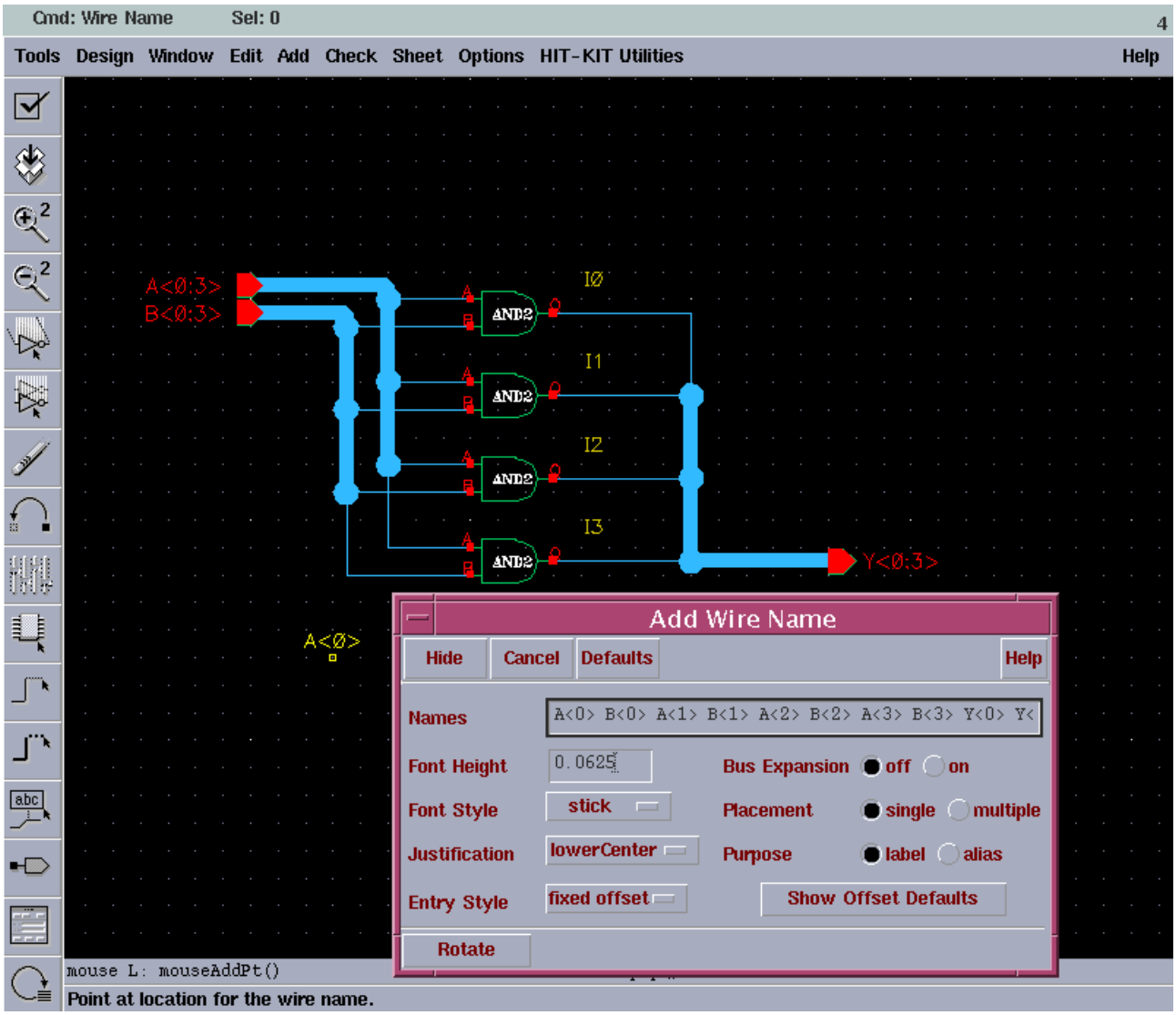
- place I/O pins. These pins could be single wires or buses. Bus notation is written as 'pin_name<start_index:end_index>'. See [Drawing 10].



Drawing 10: Adding pins

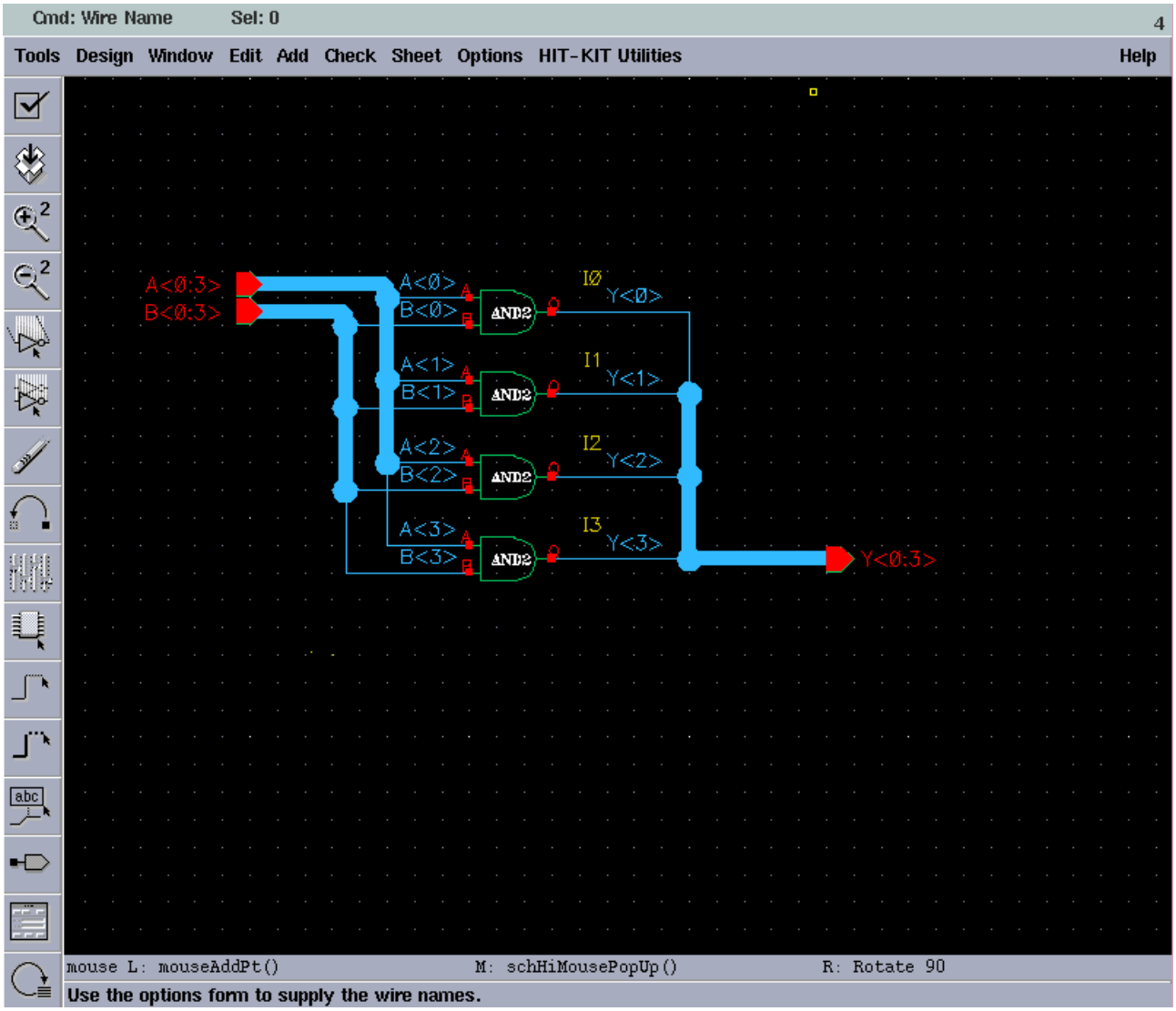
Direction field determines signal direction. Menu shorcut 'p'.

- create connections [Drawing 11] Note there are thin and thick wires (single signals and buses). For separating single signal or group of signals from bus, the wires have to have names. This solves the problem what signal should go where.



Drawing 11: Wires and names

– finished circuit [Drawing 12]. Check and save it before doing any further operations.



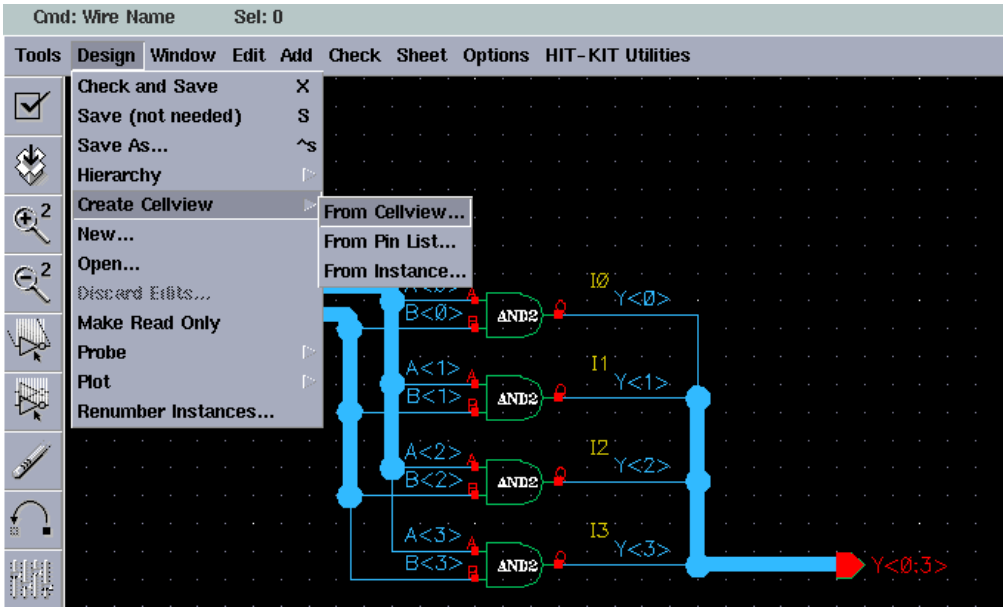
Drawing 12: Finished circuit

Now there are several possibilities. You can use this circuit as building block for even more complex circuit (as macro), run simulation in Cadence or export it for external tools like Turbo Tester. First, marco creation.

Generating macro (symbol) from cellview.

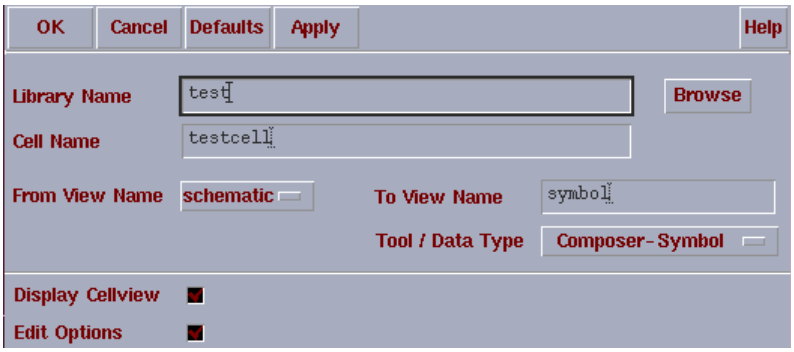
This operation takes several steps:

- make cellview from cellview



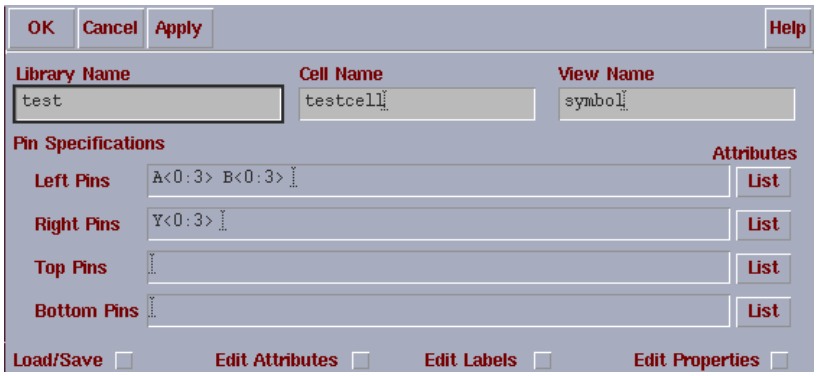
Drawing 13: Create symbol, step 1

- Cellview generation confirmation, step 2 [Drawing 14]. You can select different design here.



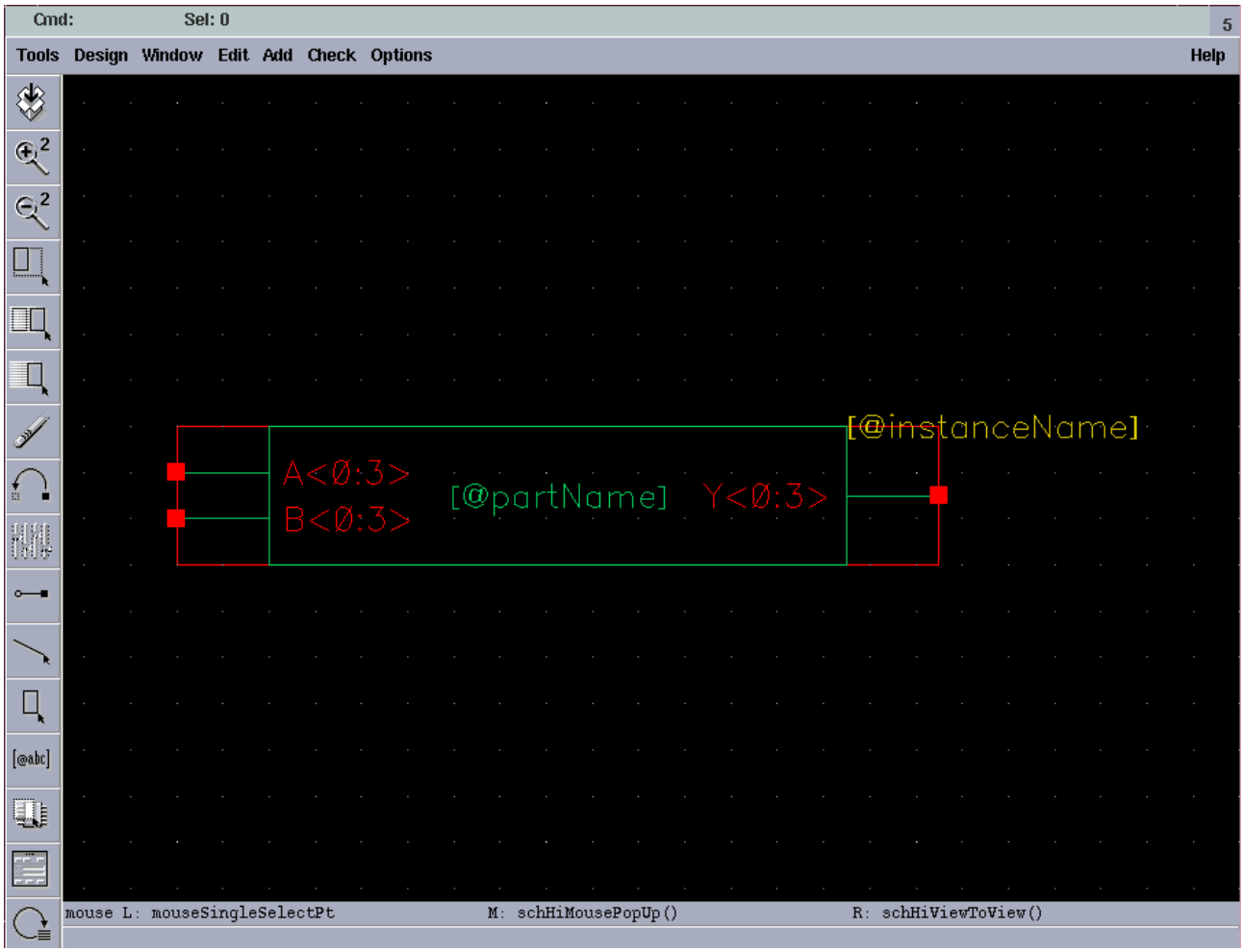
Drawing 14: Cellview from cellview

- Symbol options, step 3 [Drawing 15]. Set pin locations for macro box.



Drawing 15: Symbol options

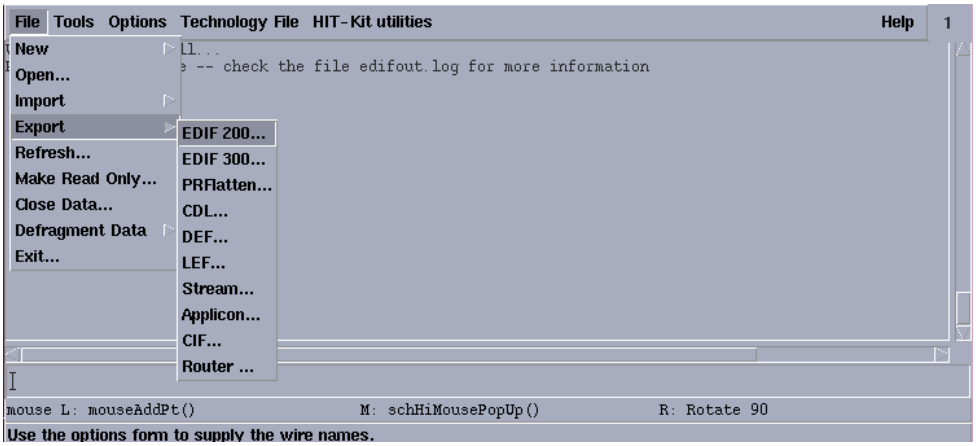
- Symbol editing, step 4 [Drawing 16]. You can rearrange pins and other macro objects in case you dont like defaults



Drawing 16: Symbol editing

Exporting circuit (EDIF 200) to Turbo Tester

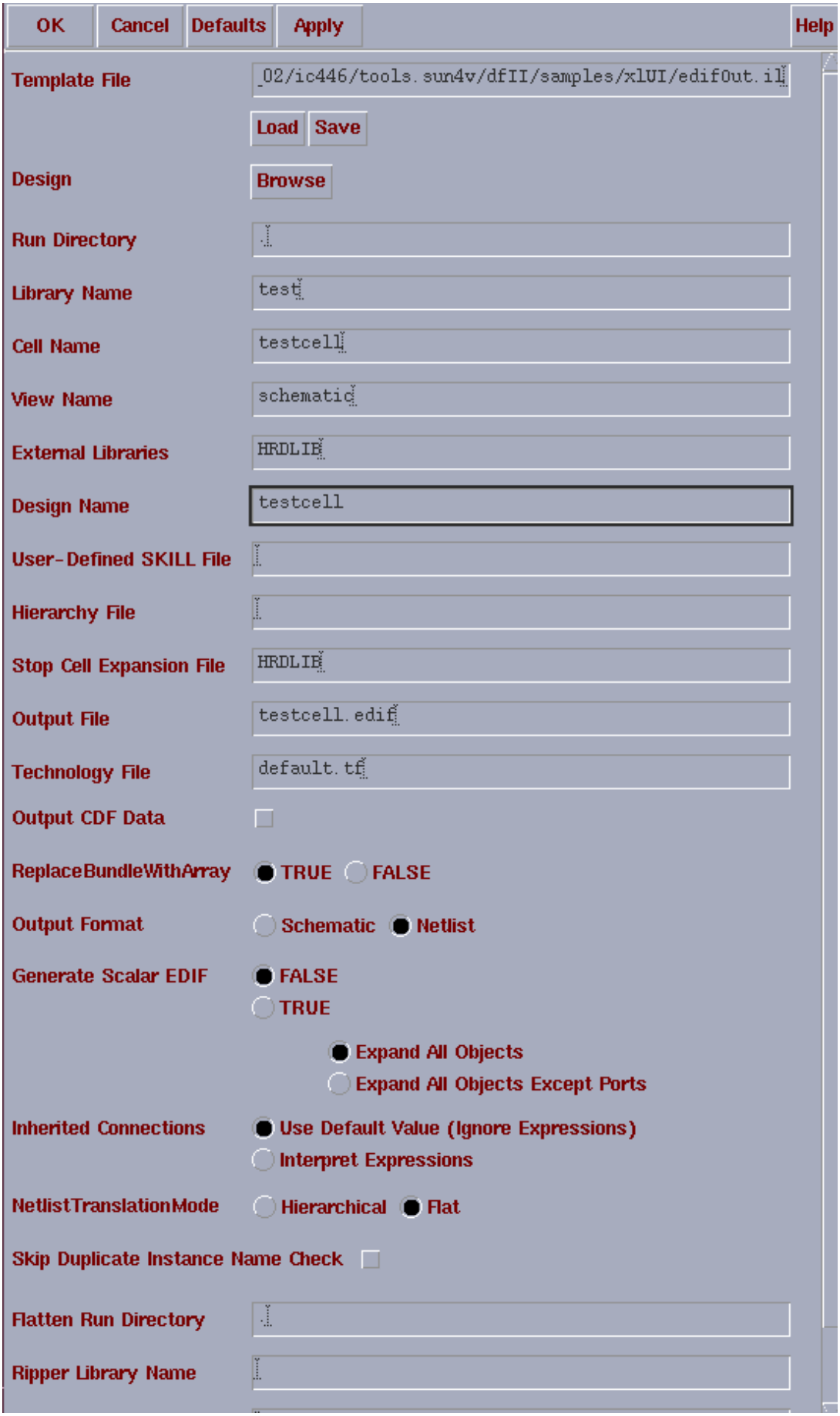
Locate window named 'icfb' and open menu as shown on [Drawing 17].



Drawing 17: Exporting desing, step 1

A large dialog window appears [Drawing 18]. Filling it takes some steps:

- select circuit to be exported. This is done via 'Browse' button opening 'Library Manager' window where you can select your design (and schematic view!)
- specify 'HRDLIB' for fields 'External Libraries' and 'Stop Cell Expansion File'. The latter is most important as this wont let transistor level data into your export (Turbo Tester importer cannot handle that).
- specify design name. Turbo Tester importer need it, although leaving it blank wont otherwise affect export process in any way.
- specify EDIF output file. Your circuit will be written there. When importing to Turbo Tester, this file name is used for resulting AGM file.
- set 'Output format' to 'Netlist' as we need the circuit only.
- set 'NetlistTranslationMode' to 'Flat', it will flatten (ie. unwrap) all macros.



Drawing 18: Exporting, step 2

Importing EDIF to Turbo Tester

Turbo Tester has special import library for 'HRDLIB' called 'ams.lib'. This library defines all HRDLIB elements (at least it should). Issue following command for the import:

'import -tool cadence <edif file name> ams.lib'.