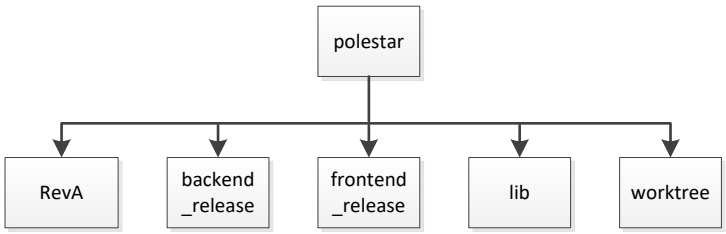
Polestar directory

Revision history

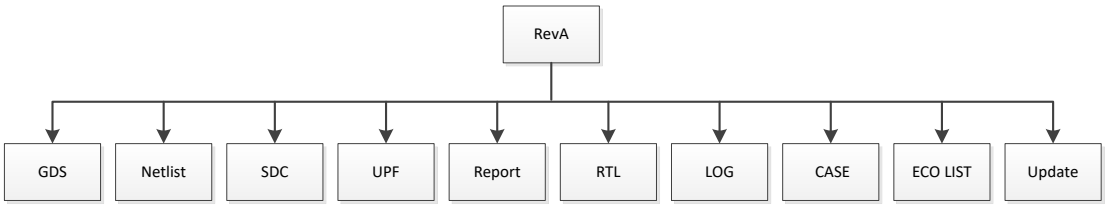
Revision	Date	Description	Author
0.1	2017-11-20	Initial	Peng Chen



Picture 1

The polestar directory is the root directory of polestar project, and it contains RevA, backend_release, frontend_release, lib and worktree directory.

1. RevA

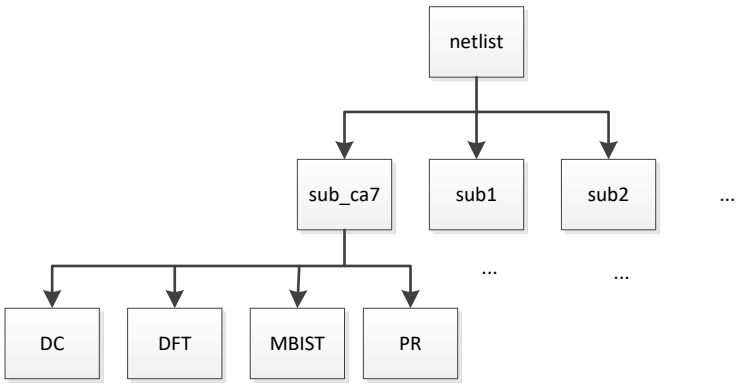


Picture 2

This directory is used to contain tap-out data. The upcoming polestar project is the first version, so it's called as RevA. If we do some ECO after the first tap-out, the version with ECO is called the RevB.

As shown in Picture 2, this directory contains GDS, Netlist, SDC, UPF, Report, RTL, LOG, CASE, ECO_LIST, Update. We can use this data to analyze problem and eco based on this data after the first tap-out. Every directory will have a log file which records all version information.

- **GDS** We will put our final GDS in this directory.
- **Netlist** We will put final DC/DFT/MBIST/PR netlist of all SUBs in this directory. We will put netlist of different stage into different directory like Picture 3.



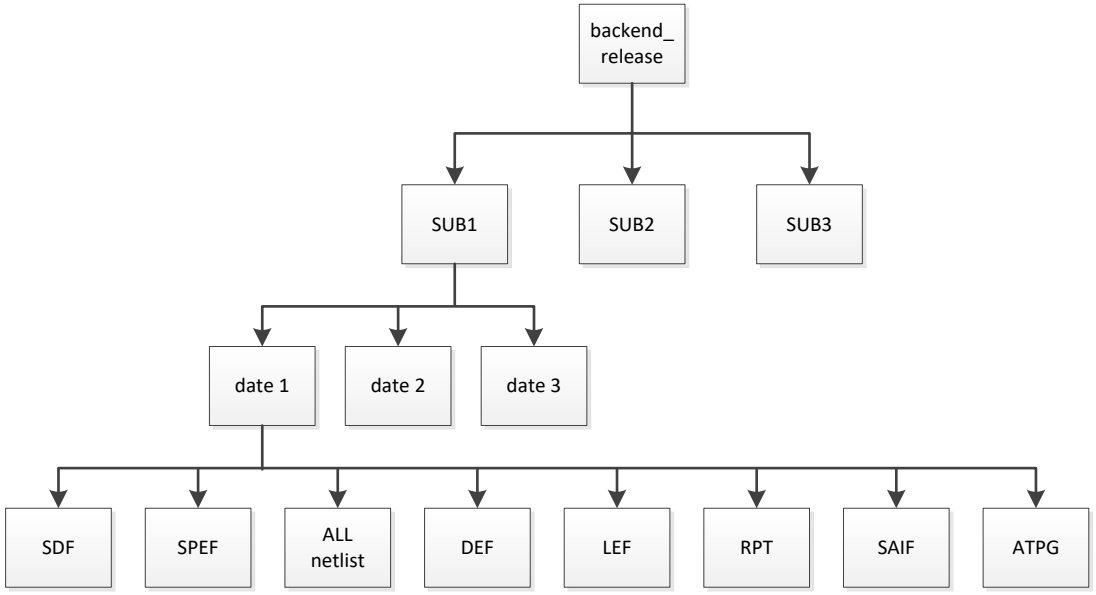
Picture 3

- **SDC** We will put final DC/DFT/MBIST/PR SDC of all SUBs in this directory. Because some new constraints may be added, so the SDC of different stage may be different. We will put SDC of different stage into different stage directory like Picture 3.
- **UPF** DC/DFT/MBIST UPF of all SUBs will be in this directory. New ports will be added at DFT and MBIST stage, so new constraints will be added into the UPF. We will put UPF of different state into different stage directory like Picture3.

- **Report** The final timing report and area report of DC/DFT/MBIST/PR/STA/LVS/DRC/POWER will be in this directory. We will put the reports of different stage into different state directory like Picture 3.
- **RTL** The final RTL of all designs will be in this directory. We will put the RTL according to IPs.
- **Log** The final log of DC/DFT/MBIST/PR/STA/LVS/DRC/POWER will be in this directory. We will put the reports of different stage into different state directory like Picture 3.
- **CASE** All test cases are here. I should be a regression which can cover most function of design. These test cases are mainly used to verify design after ECO.
- **ECO_LIST** If we find bug in our design after tap-out, we can record it in this directory. If we have an ECO opportunity, we can ECO our design according to this ECO list.
- **Update** It record all the updates of this version against the last version.

2 Backend_release

This is an exchange directory from backend team. This directory will contain DFT/MBIST/PR netlist, SDF files, SPEF files, DEF files, LEF files, report and saif files.



Picture 4

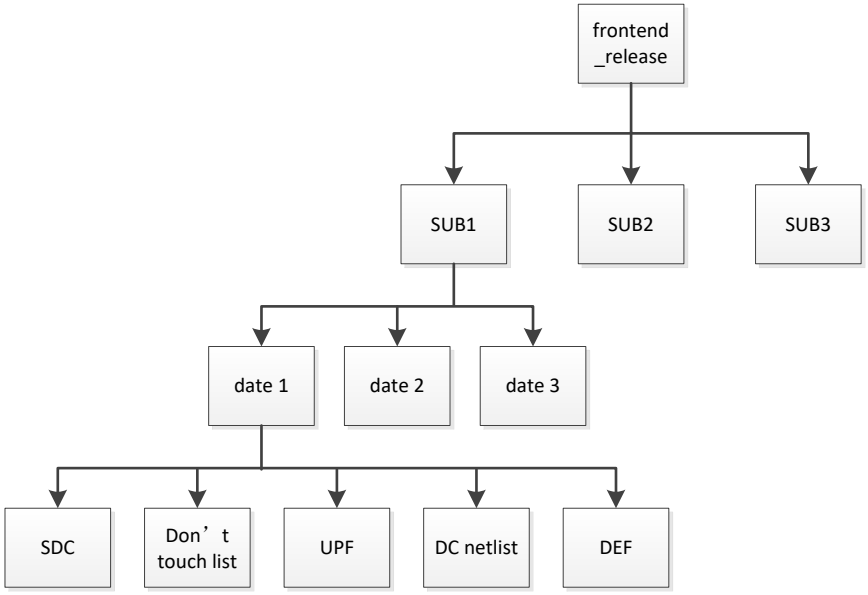
All files is managed according to sub and date like Picture 4. Let's see the function of every directory.

- **SDF** This directory contains the SDF files released by backend. This file is written out by PT and used to do post-simulation.
- **SPEF** This directory contains the SPEF files, this file can be used to do STA and POWER analyze.
- **ALL netlist** This directory contains the netlist of DC/DFT/MBIST/PR. Please pay attention to the DC netlist. The DC netlist is not from backend team. It's from frontend_release. The reason why put it here is that we should make sure the data from backend team is corresponding to which version of DC netlist.
- **DEF** This directory contains the DEF files from backend team. This file is used in DCG

synthesis. If you don't go DCG flow, you can ignore this file.

- **LEF** This directory contains the LEF files from backend team. This file contains the information of IP/SUB's block. This file is used in bottom up DCG flow.
- **RPT** This file contains the report from backend team. I think it should contains the flowing report.
  - Timing report of all corners in STA.
  - Physical report after Routing or DFM in layout tools (ICC or Innovus).
  - Power report including leakage power and peak power.
  - LVS report.
  - DRC report.
- **SAIF** This file contains the SAIF files written out by ICC. If your IP is go scan reorder flow in ICC, you should read in this file to do formality for DFT netlist ←→ PR netlist.
- **ATPG** This file contains the ATPG pattern. This files is generated by TMAX. If we do DFT test, we will need this files.

2. Frontend_release

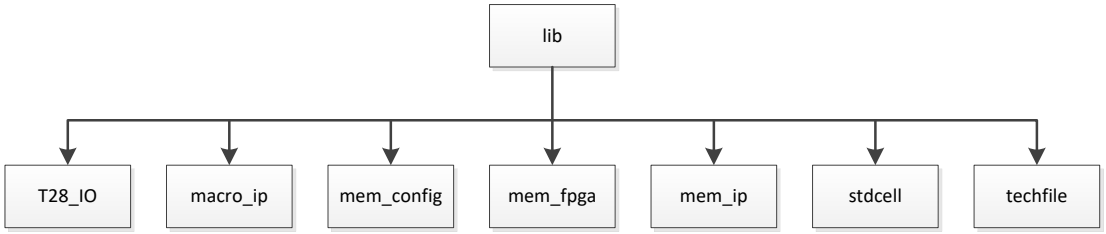


Picture 5

Frontend team will use this directory to release data to backend team. All files are managed according to sub and date like Picture 5. The following is the instruction for the function of every directory.

- **SDC** This directory contains the SDC files written out by DC or written by IP owner.
- **Don't touch list** This file will contain the don't touch list for special design. For example, clock gate, design with STDCELL and so on.
- **UPF** If there is low power design in your design. You need to supply the UPF file. This may be written out by DC or written by IP owner. This is decided by talking with backend team.

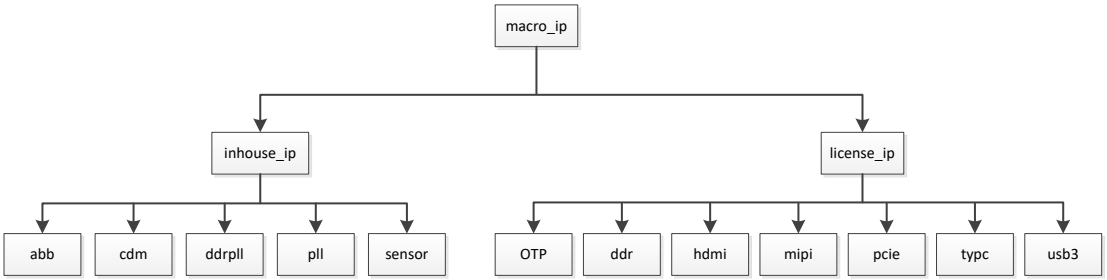
3. Lib



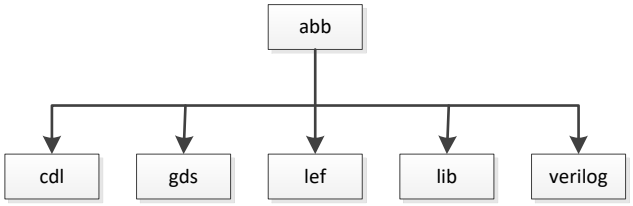
Picture 6

As shown in Picture6, the lib directory contains T28_IO, macro_ip, mem_config, mem_fpga, mem_ip, stdcell and techfile. The following is the introduction of these directories.

- **T28_IO** This the TSMC IO pad library. It contains all the data of TSMC IO pad.
  - If you do simulation using pad. Please point to this directory. The sim models of IO pad without PG are tphn28hpcgv18od33_rgm2.v, tphn28hpcgv18e.v and tphn28hpcgv18od33.v. The sim models of IO pad with PG are tphn28hpcgv18od33_rgm2_pwr.v, tphn28hpcgv18e_pwr.v and tphn28hpcgv18od33_pwr.v. The sim models with PG pins are used when you do PG netlist simulation. You can use "find . –name "*.v"" at this directory to find these simulation models location.
  - If you do synthesis using pad. Please point to this directory. You can use "find . –name "*.lib"" at this directory to find all libs' detail path. Please use the NLDM model to synthesis. I don't find the CCS lib in this library.
- **Macro_ip** This directory contains inhouse_ip and license_ip like Picture 7. The IPs designed by ourselves are in inhouse_ip. There are abb, cdm, ddrpll, pll and sensor in this directory. The licensed IPs are in license_ip. There are OTP, DDR, HDMI, MIPI, PCIE, TYPC and USB3.0 PHY in this directory. In order to use easily and make sure the version is consistent, I merge all data of one IP into one directory. For example abb, I add cdl, gds, lef, lib and Verilog directory under its direcroty like Picture 9. Other in-house IPs have the same directory structure.



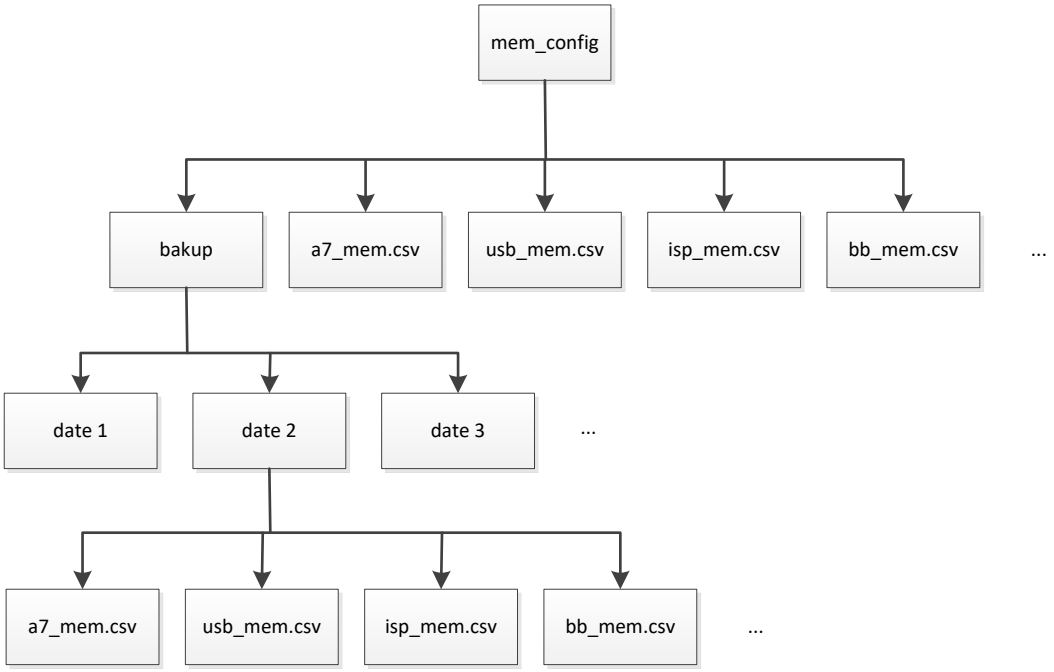
Picture 7



Picture 8

- Cdl This directory contains the spice model of abb which is used to do LVS check after layout.
- Gds This directory contains the GDS file of abb which is the real circuit and will be merged with design GDS after layout design.
- Lef This directory contains the LEF file of abb which is abstract view of GDS and will be used to generate Milkyway.
- Lib This directory contains the lib and corresponding db file, which is the logical library of abb and will be used in all eda flow except phycial verification.
- Verilog This directory contains the simulation model of abb, this simulation model can't be used in synthesis. Please pay attention at it.

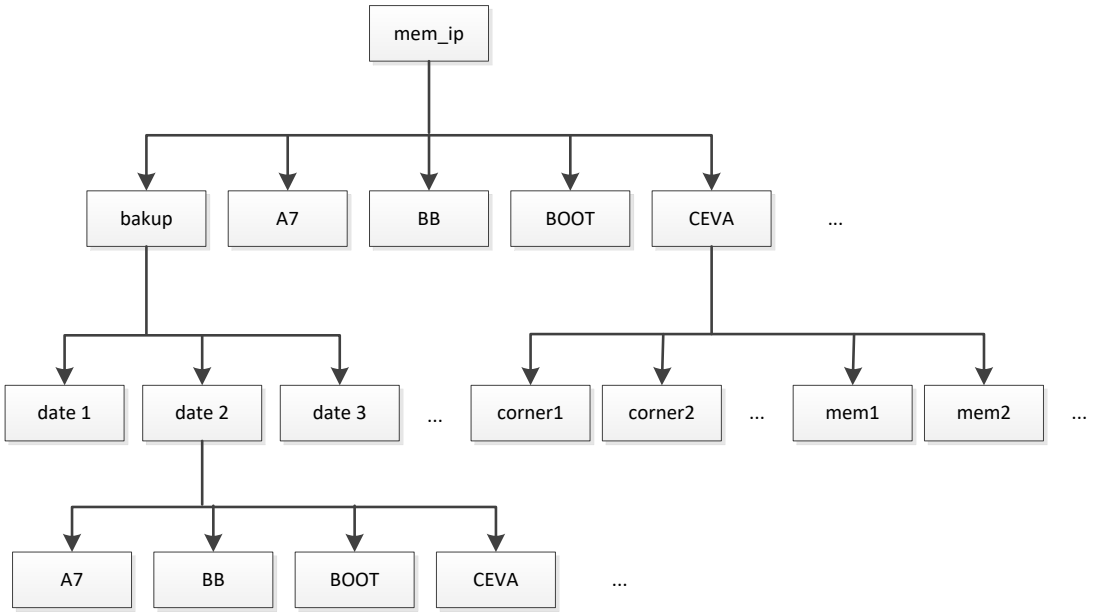
- **Mem_config** As shown in Picture 9, it's the structure of mem_config. All the csv files under mem_config are the current version mem configure files. The mem generate script will use these mem configuration files to generate ASIC memories and FPGA memories. When we update the mem configuration files, we will make a new directory marked with current date under bakup directory, and copy all the CSV files to this directory. And then update the CSV files under mem_config directory. There is a demo mem configuration CSV file at ${polestar_doc_git}/memory/Memory_Configuration_Form_sample.xls. You can update your mem configuration files at this GIT path, I'll update them at our server. I'll introduce how to choose the mem configuration latter.



Picture 9

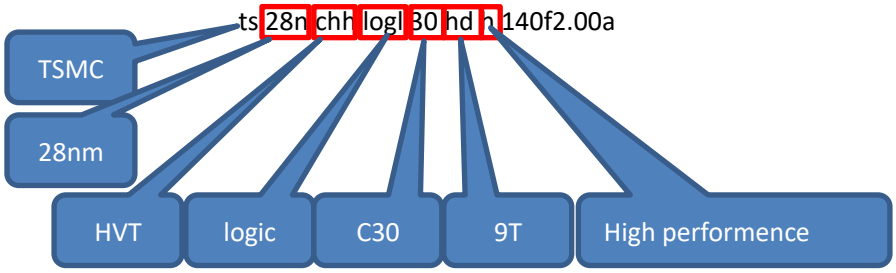
- **Mem_ip** As shown in Picture 10, it's the structure of mem_ip. All memories are arranged by IPs. All IP directories under mem_ip are the current used memories. If we update the memories, we will back-up current memories under bakup directory marked with the date. There are two kinds of directory under every IP directory and take CEVA as an example. There are corner directories and memory directory. Corner directory is named like ssg0p81vn40c, it contains all the LIB and DB files of all the memories at this corner. When you synthesis, you can point the search path to this path to use these library at this corner.

The memory directory is named like CEVA_SR_1P_1024X32, and it contains all the data of this memory.



Picture 10

- **Stdcell** This directory contains all the library of the stdcell. Let's talk about the naming rule of STDCELL library. Take ts28nchhlogl30hdh140f2.00a as an example.



chh →HVT, chs→SVT, chl→LVT;
30 →c30, 35→C35, 40→C40;
hs→12T, hd→9T, ud→7T;
h→high performance, l→logic library(basic library), p→power library.
In polestar, we will use 9T and 7T C35 SVT stdcell to synthesis our design, maybe we will use a litte LVT C35 stdcell to synthesis. Please don't use any HVT stdcell to synthesis.

- **Techfile** This directory contains all data about technology file. We will use tluplus file in this directory in DCG flow. Please don't care about other files. They will be used in backend flow.

4. Worktree

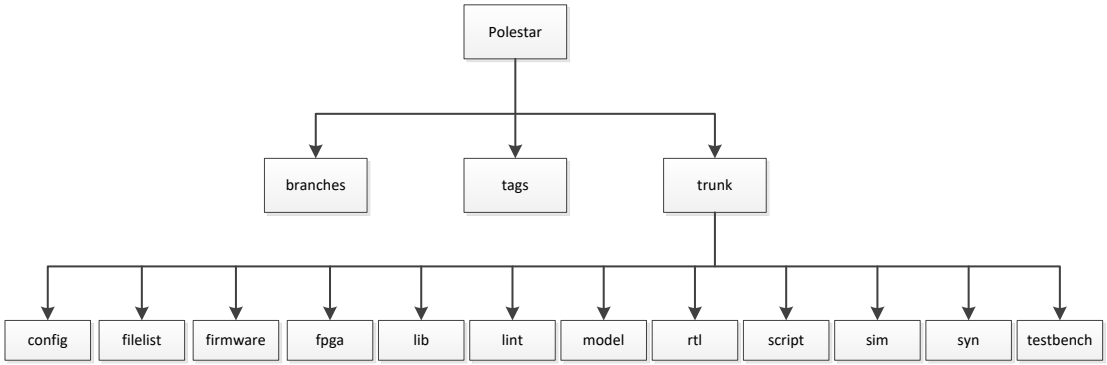
This is our work space. Everyone has a project home directory. In order to use these directory conveniently, I advise you to link your project home directory to your home. For example, use "ln –s /project/polestar/worktree/pchen ~/polestar_home", so I can use the polestar_home upder my home to get into my project home directory.
The information of Polestar SVN server is like Picture 11.

```
Working Copy Root Path: /projects/polestar/worktree/pchen/Polestar
URL: https://192.168.200.210/svn/Projects/Polestar
Repository Root: https://192.168.200.210/svn/Projects
Repository UUID: a2c8865f-f9d5-d249-8d55-a1f78a06f2fb
Revision: 18922
Node Kind: directory
Schedule: normal
Last Changed Author: VisualSVN Server
Last Changed Rev: 18922
Last Changed Date: 2017-11-15 17:09:26 +0800 (Wed, 15 Nov 2017)
```

Picture 11

- **Check out** If you want to check out this database, you can change your path to where you want to create it, and then use "svn co https://192.168.200.210/svn/Projects/Polestar" to check out this database.
- **Check in** If you have modified something, and want to check in it. You can use "svn ci –m "the comment"" command to check in it. ***Please pay attention at the comment. The comment must descript your modification, and don't just ignore it.***
- delete If you want to delete any directory or file. You can use "svn del file/directory name" and then use "svn ci –m "comment"" to check in it.
- update Use "svn update" to update the latest database.

4.1 Polestar database



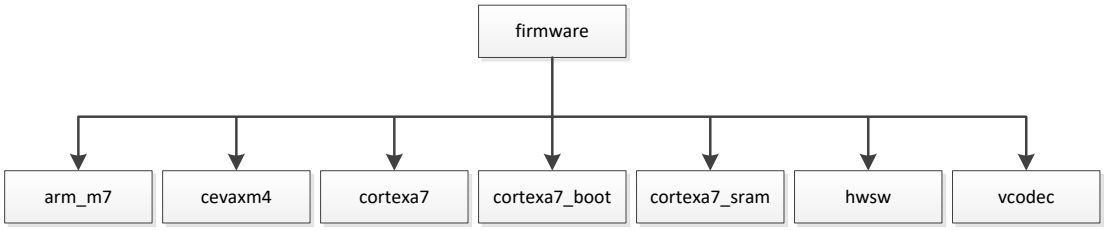
Picture 12

As shown in the Picture 12, it's the directory structure of polestar database. The following will descript all the directories in detail.

- **config** This directory contains some Linux environment configuration including alias configuration and tool path. Because everyone has different habit, I modify my own .cshrc to configure my alias and tool path environment. I advise to set your own alias, it can speed up your access different directories.
  - **alias.bash** It's a bash alias configuration file. The alias is just about some directory of polestar project directory. The default shell of EDA-20~25 is C shell. If you use this script, you need to change your shell to bash.
  - **alias.csh** It's a C shell alias configuration file. The alias is just about some directory of polestar project directory. The default shell of EDA-20~25 is C shell. You can just source this script to configure these alias.

- **bashrc.cadence** It's a bash configuration which configure cadence tool path.

- **filelist** This directory contains the filelist of all IPs. All the filelist is named as ${ip_name}.f. This directory is named as filelist2 in Sirius, I remove the old filelist and rename filelist2 as filelist and change all path using filelist2 to filelist in polestar.
- **firmware** This directory is software environment. It contains arm_m7, cevaxm4, cortexa7, cortexa7_boot, cortexa7_sram, hwsw, and vcodec software environment.

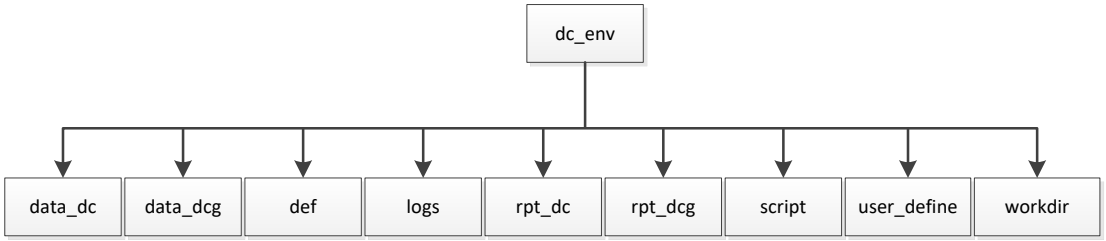


Picture 13

  - **arm_m7** This is a software environment for cortexM7. It has two link script – reset_flash.ld and reset_itcm.ld. This environment use reset_itcm.ld which constraints the M7 boot from 0x0000_0000 in default mode. Reset_flash.ld constraints the M7 boot from 0x1000_0000, which function need to change the vector_bass of M7 in hardware environment.
  - **cevaxm4** This is a ceva software convert environment. The code of ceva need to be generated by ceva tool-chain. It's ***.a file. This environment converts this ***.a to a hex file which can be read into mem.
  - **cortexa7** This is a software environment for cortexa7, which require the system has a 8MB ROM at address 0x0000_0000. Its data is merged with code, but cortexa7 need the data is in SRAM. The boot of this software can move data from ROM to SRAM automatically, configure the stack point to SRAM, and then jump to the main function.
  - **cortexa7_boot** This is a first level boot software environment for cortexa7. It can make core 1~3 get into sleep mode and just permit core 0 get into the second level boot. The second level boot's address is 0x3000_0000.
  - **cortexa7_sram** This is second level boot which runs in SRAM. It finish initializing general register of a7, points the vector table to the SRAM, setup the stack and then jump to main function to run your test case.
  - **hwsw** This is a hardware and software environment which is used the hwsw function of Verdi. Hwsw can convert the tarmac log to fsdb file. Verdi can read in this fsdb file and simulation fsdb file, and enable the hwsw debug to debug software and hardware. It's like the codelink of velcc.
  - **Vcodec** This is a software for cortexa7, it contains the driver of vcodec and test case.
- **Fpga** This directory is for synplify and vivado compile. I reserved all the database of Sirius, if some database is useless, please remove it from svn server. The synplify environment is to synthesis the RTL to gate level netlist, and Vivao use this netlist to do PR.
- **Lib** This directory is used to contain the data of in-house IPs. The analog team can release their IPs's data in this directory. If they feel the IPs are stable, I'll copy them to /projects/polestare/lib/inhouse_ip. The data of in-house IP should include lib file, lef file,

GDS file, spice model and simulation model. Analog team must make sure the version of all data should be consistent.

- **Lint** This is the SpyGlass environment which is used to check CDC design based on RTL. All IP owners must check their design using this environment to make sure there is no CDC problem.
- **model** This directory contains the simulation model. This directory should just contains the device model like the flash model, amba_vip and so on. Please don't put the simulation model of STDCELL, memory and MACRO IP in this directory, these simulation model should be in /project/polestar/lib.
- **rtl** Please put all RTL design in this directory according to the IP and make sure don't put simulation files into this directory.
- **script** This directory contains some tool script, for example, bin2hex, lib2db and so on. If you have any widely used script, you can share it in this directory.
- **syn** This directory contains a dc_env. This is a general dc environment. If you want to evaluate the area and timing of your IP, you can copy it and rename it according to your IP name. Its directory structure is like Picture 14.



Picture 14

  - **data_dc** SDC, DDC, UPF and SAIF files written out by DC will be in this directory.
  - **data_dcg** SDC, DDC, UPF and SAIF files written out by DCG will be in this directory.
  - **def** The DEF file released by backend will be put in this directory. The DEF file will be used in DCG flow.
  - **rpt_dc** All reports written out by DC is in this directory. You can check the timing report, link report, check design report and so on in this directory.
  - **logs** The logs of DC and DCG will be in this directory. You can open the log of DC or DCG in this directory to check if there is any error in the log to debug DC/DCG flow.
  - **rpt_dcg** All reports written out by DCG is in this directory. You can check the timing report, link report, check design report and so on in this directory.
  - **scripts** The main flow scripts of DC/DCG/FM is in this directory is in this directory. You don't need to modify anything in this directory.
  - **user_define** All the files you need to modify is in this directory.
    - **0_lib_setup:** You just need to modify the mem_db list in this file like Picture 14, you can add your memories and Macros into this list. This file will be merge into ".synopsys_dc.setup" file.

```
set mem_db "\
                                RF_1P_256X22_lib.db \
                                RF_1P_128X26_lib.db \
                                RF_1P_512X32B1_lib.db \
                                SR_1P_4096X32_lib.db \
                                SR_1P_1024X64_lib.db \
                                RF_1P_32X24_lib.db \
                                RF_1P_32X32_lib.db \
                                SR_1P_1024X32B1_lib.db \
                                SR_1P_8192X32B1_lib.db \
                    "
```

Picture 14

- **1_design_setup**: This file define your IP top. Take CXSOC as an example. Its design setup is like Picture 15. Just rename CXSOC according to your top name.

```
set design_name "CXSOC"

set lib_name "lib_CX_SOC"
```

Picture 15

- **2_rtl_list**: This is the file-list of your design. Just add all your verilog designs path into this file. If your design contains the VHDL design, please tell me. Please pay attention, there is “include” in your design file, you must add its path like Picture 16. It has the same grammar as VCS.

```
+incdir+../../../rtl/CoreSight_SoC/cxapbasyncbridge/logical/cxapbasyncbridge_i/verilog
+incdir+../../../rtl/CoreSight_SoC/cxapbic2_7/logical/cxapbic_2x7/verilog
+incdir+../../../rtl/CoreSight_SoC/cxatbasyncbridge/logical/cxatbasyncbridge_i/verilog
+incdir+../../../rtl/CoreSight_SoC/cxatbdownsizer/logical/cxatbdownsizer_64_32/verilog
+incdir+../../../rtl/CoreSight_SoC/cxatbfunnel/logical/cxatbfunnel_64/verilog
+incdir+../../../rtl/CoreSight_SoC/cxcti/verilog
+incdir+../../../rtl/CoreSight_SoC/cxctm/verilog
+incdir+../../../rtl/CoreSight_SoC/cxdapahbap/verilog
+incdir+../../../rtl/CoreSight_SoC/cxdapapbap/verilog
+incdir+../../../rtl/CoreSight_SoC/cxdapaxiap/logical/cxdapaxiap_i/verilog/
+incdir+../../../rtl/CoreSight_SoC/cxdapbusic/logical/cxdapbusic_3/verilog
+incdir+../../../rtl/CoreSight_SoC/cxdapswjdp/verilog
+incdir+../../../rtl/CoreSight_SoC/cxetb/verilog
+incdir+../../../rtl/CoreSight_SoC/cxntsasyncbridge/logical/cxntsasyncbridge_mcu3/verilog
+incdir+../../../rtl/CoreSight_SoC/cxtpiu/verilog
+incdir+../../../rtl/CoreSight_SoC/cxtsgen/verilog


../../../rtl/CoreSight_SoC/cxsoc/CXSOC.v
../../../rtl/CoreSight_SoC/cxapbasyncbridge/logical/cxapbasyncbridge_i/verilog/cxapbasyncbridge_cdc_capt.v
../../../rtl/CoreSight_SoC/cxapbasyncbridge/logical/cxapbasyncbridge_i/verilog/cxapbasyncbridge_cdc_corrupt_gry.v
../../../rtl/CoreSight_SoC/cxapbasyncbridge/logical/cxapbasyncbridge_i/verilog/cxapbasyncbridge_cdc_corrupt.v
../../../rtl/CoreSight_SoC/cxapbasyncbridge/logical/cxapbasyncbridge_i/verilog/cxapbasyncbridge_cdc_launch_data.v
```

Picture 16

- **3_constraints**: This file contains all your constraints for your design. It contains create_clock, create_generate_clock, max_delay, max_fanout, max_transition, set_load, multicycle and so on. If you add false path in this file, please make sure the constraint is reasonable.
- **4_power**: If your design is a low-power design, please add your UPF in this file. If your design is not a lower-power design, just keep it empty.
- **5_dft**: This file is used during DFT compile. Please just keep it empty.

How to use it in workdir directory:

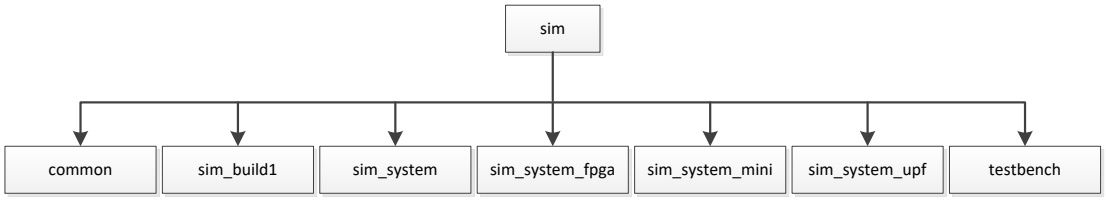
- **./run dc -7t**: DC mode using 7T LVT and SVT stdcell to synthesis.

- **./run dc -9t –svt**: DC mode using 9T SVT stdcell to synthesis.
- **./run dcg -9t –svt –tt**: DCG mode using 9T SVT TT corner STDCELL to synthesis.
- **./run fm**: Do formality check.

The above are just some example. For general case, **./run <mode> <track>[cell_type][corner]**. "[]" is optional option, "<>" is mandatory option.

- **mode**: DC DCG and FM
- **track**: 7T, 9T and 12T. Please don't use 12T in polestar.
- **cell_type**: LVT, SVT and HVT. Please don't use HVT cell in polestar.
- **corner**: TT, typical corner. Default is ssg0p81vn40c.

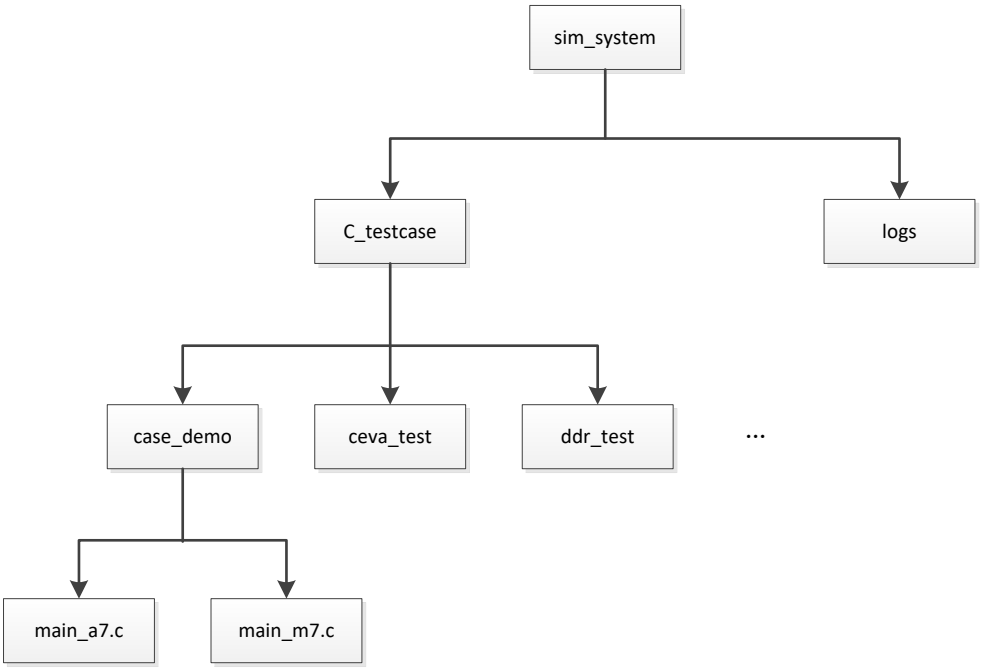
- **sim** This is the simulation directory. It's the same with sim2 in Sirius. I remove the old sim directory and rename the sim2 to sim, and change all paths about sim2. The directory structure of sim is like Picture 17.



Picture 17

- common This directory is some script used in simulation.
- sim_build1 This is a veloce environment.
- sim_system This is RTL simulation environment.
- sim_system_fpga This is a FPGA simulation.
- sim_system_mini This is a post-simulation environment.
- sim_system_upf This is an UPF simulation environment.
- testbench This directory contains testbench and other simulation model.

Take sim_sytem as an example.

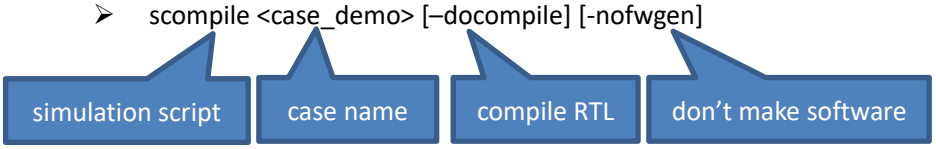


Picture 18

The test case is under C_testcase directory. There is a directory for every test case. There is main_a7.c and main_m7.c. If you want to create a test case, you can copy a test case and name it as your test case. When you run the test case, these two C file will be linked to the firmware/cortexa7 and firmware/arm_m7 and compile them.

Sim step: (“<>” is mandatory option, “[]“ is optional option)

- source “common/simsetup” at sim directory.
- Get into the simulation environment. Take sim_system as an example.
- scompile <case_demo> [–docompile] [-nofwgen]



- get into simulation directory “tmpdir/case_demo_VCS”.
- To generate Verdi database, “make com”.
- Start Verdi, use “make Verdi”.

For the eda_flow directory, I’ll create them in my project home. I’ll introduce them when I introduce the EDA flow.