

IFP 用户手册

Product Name : IFP (ic flow platform)

Product Version : V1.0

Release Date : 2022.12.12

Contact : @李艳青 (liyanqing.1987@bytedance.com)

@节喜 (jiexi@bytedance.com)

@景富义 (jingfuyi@bytedance.com)

目录

前言.....	3
一、简介.....	3
1.1 IC 流程演化	3
1.2 IC 流程平台的主要作用	3
1.3 IFP 开发思路	4
1.4 IFP 基本概念和控制逻辑.....	5
1.4.1 任务拆解 (从 Block 到 Task)	5
1.4.2 单个任务的行为逻辑 (从 Task 到 Action)	6
1.4.3 IFP 控制逻辑	7
二、环境依赖.....	8
2.1 操作系统依赖	8
2.2 PYTHON 版本依赖	8
2.3 集群管理系统	8
2.4 共享存储	8
三、工具安装及配置.....	9
3.1 工具安装	9
3.2 工具配置	10
3.2.1 config/config.py	10
3.2.2 config/default.yaml	11
3.2.3 config/env.*	13
四、工具使用.....	16
4.1 工具载入	16
4.2 配置文件	17
4.2.1 编辑用户配置	17
4.2.2 编辑默认 Task 属性配置	19
4.2.3 便捷 Config file 生成	20
4.3 界面介绍	21
4.3.1 菜单栏	22
4.3.2 功能区	25
4.3.3 工作区	25
4.4 工具运行	27
4.5 其它功能介绍	29
4.5.1 主界面功能	29
4.5.2 辅助工具	32
五、范例.....	33
5.1 用户环境准备	33
5.1.1 用户 configure 文件	33
5.1.2 ifp 配置文件	34
5.2 操作范例	35
附录.....	39
附一、变更历史	39
附二、IFP 的配置文件	39

前言

IFP 是 ByteDance 芯片业务自研的芯片设计流程平台，主要用于超大规模数字集成电路设计的流程规范管理和数据流转控制。

一、简介

1.1 IC 流程演化

集成电路设计流程建设并非一蹴而就，在很多公司和团队的流程演化都可以笼统地分为三个阶段。

手工阶段：

集成电路设计流程以手工为主，包括手工创建运行目录并解决文件依赖，手工运行 EDA 工具并检查运行结果，手工收集报告并 **release** 数据。

这阶段的主要瓶颈是，在集成电路设计规模较大，并且人力资源不够充足的情况下，手工操作效率较低，工作量巨大。

脚本阶段：

通过脚本化，包括但不限于 **shell/Makefile/perl/python** 等，将设计流程步骤中的手工操作转化为脚本执行，这样不但可以极大地提升运行效率，而且可以将流程步骤固化。

这阶段的主要瓶颈是，在超大规模集成电路设计中，每个流程往往需要多人协作，多个流程之间数据流转也更加紧密，以方便快速迭代，因此对流程运行规范提出了更高的要求。

平台阶段：

通过一个统一的流程平台，在一个公司（或团队）范围内统一流程运行规范，数据集约化管理和展示，可以大幅降低流程运行成本（包括时间成本和交互成本）。

这个阶段的主要制约因素是，大家的操作对象从分立脚本切换到统一平台，有一个一次性的学习成本；部分流程管理员失去了对操作平台功能的直接修改权限会有一些不适应。

1.2 IC 流程平台的主要作用

- **统一流程运行规范**

超大规模集成电路设计需要多流程参与，每个流程内部也需要多人协同工作，流程平台可以将 IC 流程运行规范通过工具的方式固化，确保所有的人采用统一的运行模式和规则，降低交互成本。

同分立的自动化脚本相比，流程平台是多流程唯一入口且无法随意篡改的。

- **数据集约化管理和展示**

流程平台本身承载数据检查及展示（checklist/summary）和数据管理（release）的作用，同时可以嵌入更长维度的数据保存及检索能力，可以对多项目/多流程/多用户的数据进行集约化管理和展示，对多流程间的数据交互更友好。

- **降低流程运行门槛**

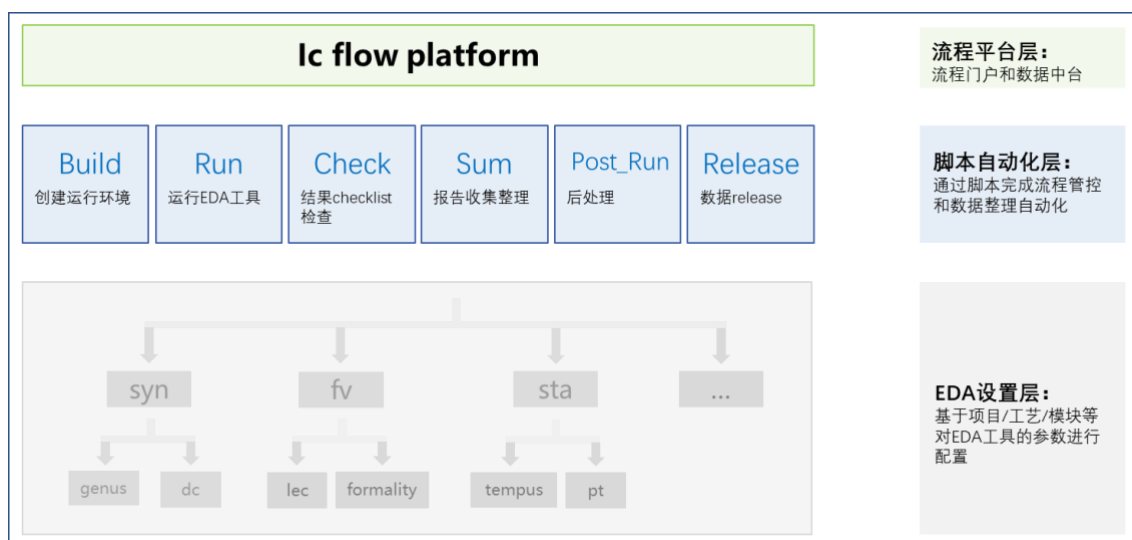
GUI 界面的流程平台简单易操作，可以大幅降低新人的流程运行门槛。同时“脚本自动化层”和“EDA 设置层”通过松耦合的方式嵌入，也不影响用户了解和配置单步自动化的实现方式以及具体的 EDA 设置项。

- **提升多模块运行效率**

流程平台内嵌基于 LSF 或者 multi-thread 的多模块并行运行控制，能够在 block/version/flow/vendor/branch/task 不同的层面实现串并行精准管理。

1.3 IFP 开发思路

IFP 采用层次解耦的设计思路，将整个 IC 流程平台分为三个逻辑层次。



底层是 **EDA 设置层**，也是绝大多数 project engineer 日常接触到的部分。我们根据自己所参加的项目（ada/comet/vivian/...），项目所使用的工艺节点（16/7/5/...），自

己所选用的 EDA 工具 (genus/dc/...), 自己所负责的模块 (PCIE/DDR/...), 来决定如何设置 EDA 工具的配置文件, 以保证 EDA 工具能够根据自己的合理配置运行得到期望的结果。

中间是**脚本自动化层**, 也是每个 flow administrator 所维护的部分。针对单个的 block 而言, 实际的 IC 流程运行并非单纯运行 EDA 工具, 还需要预先准备运行环境 (目录结构, input 文件, 环境配置等), 按照 checklist 检查 EDA 运行结果, 从运行目录收集和整理重要报告及数据, 结果符合预期的情况下进行数据 release, 有时候还有一些后处理的工作要做, 而通过脚本, 无论是 shell/Makefile 还是 perl/python, 都可以大幅降低这部分工作的操作难度。除此之外, 多模块/多流程同时运行也可以借助脚本实现并行控制和顺序管控。

顶层是**流程平台层**, 是 IC 用户 EDA 流程的入口。比较知名的 IC 流程平台有 AMD 的 TileBuilder/达索的 Altair FlowTracer/synopsys 的 Lynx 等, 一般采用 GUI 界面的方式, 支持多 block/version/flow/vendor/branch/task 的并行运行, 并集成自动化脚本实现各个流程的自动化运行。

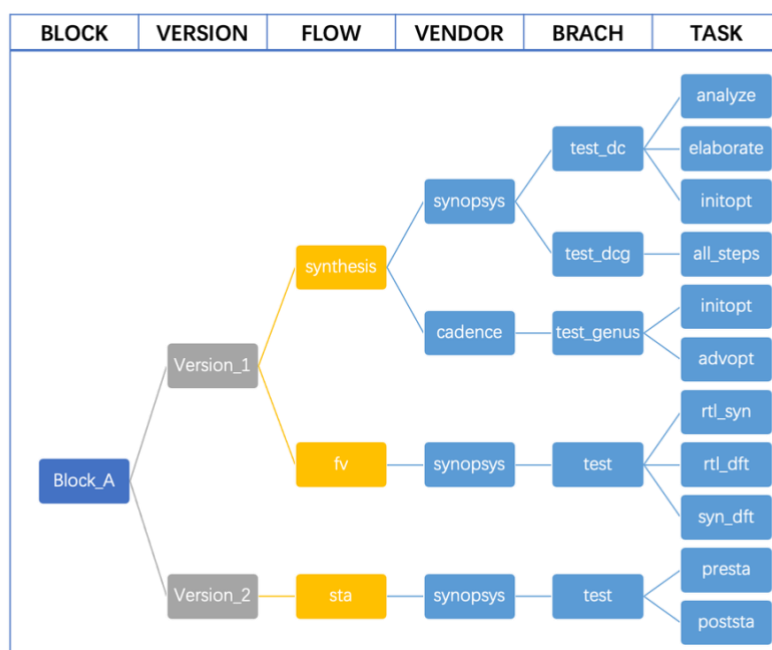
汇总来说:

- IFP 只是流程平台层, 用来调度和管理用户指定的任务。
- 自动化脚本由每个 flow 的管理员自行维护和管理, 并非 IFP 的一部分。
- EDA 设置由每个 flow 的管理员和相关用户自行维护和管理, 并非 IFP 的一部分。
- 每个流程既可以采用流程平台运行, 也可以采用脚本运行, 只是流程规范和功能完备性上的区别。

1.4 IFP 基本概念和控制逻辑

1.4.1 任务拆解 (从 Block 到 Task)

IFP 的任务拆解逻辑如下。



- Block，即集成电路模块。之所以把 Block 放到任务划分的顶层，是因为一般自顶向下的数字集成电路设计流程，大多采用 Block 来划分任务。
- Version，即 Block 不同设计阶段的版本信息。比如 PN85、PN95_v1 等。
- Flow，即集成电路设计的流程环节。比如 DFT、synthesis、fv、sta 等，不同公司的划分和命名方式会有些许不同。
- Vendor，即运行某个流程所用到的哪家 EDA 厂商的工具。比如针对 synthesis 流程，常用的工具有 cadence 的 genus 和 synopsys 的 dc_shell，所以存在 cadence 和 synopsys 两个 Vendor 可选。（不同 vendor 会影响 checklist/summary 脚本，建议区分）
- Branch，用于同一 Vendor 工具的不同模式区分。比如用 synopsys 的 dc_shell 来跑 synthesis 流程，也可能存在 dc 和 dcg 两种不同的运行模式。
- Task，每一个运行环节的最小的运行逻辑，即一个任务。比如 dc_shell 跑 synthesis，可以细分为 analyze/elaborate/initopt/advopt/finalopt 等不同的步骤，每个步骤都是一个 Task。

1.4.2 单个任务的行为逻辑（从 Task 到 Action）

我们把 IC 设计流程步骤（task）的实现抽象成几个基本的行为逻辑，每一个 task 所有可能的操作都可以包含到这几个基本行为逻辑中。



每一个 Task 可能只用到其中的一个或者几个 Action。

对于每个 Action 而言，需要设定一些基本属性以帮助 ifp 确定如何实现这一 Action，其基本的属性如下：

PATH：在哪个路径下运行这个行为。

COMMAND：如何运行这个行为（执行的命令）。

RUN_METHOD：执行这个行为的时候，是否需要 bsub 出去（指定 bsub 的命令），如果没设定，默认是本地运行。

针对 CHECK 和 SUMMARY 这两个 action，因为会生成报告，还需要查看报告，所以还有如下两个可选项需要配置。

VIEWER：是用什么工具或者命令来查看报告，可以带参数。

REPORT_FILE：报告文件的具体位置。

1.4.3 IFP 控制逻辑

用户需要运行哪些任务（Block/Version/Flow/Vendor/Branch/Task），需要用户每次运行的时候，在用户配置文件（user config file）中定义。

每个任务（Task）的行为（Action）如何执行，用户可以在用户配置文件（user config file）中实现，方便起见，也可以让流程管理员在默认配置文件（default config file）中预先定义。

IFP 主要负责按照用户的意图（IFP 的 GUI 界面操作），并行或者串行运行用户指定的任务（user config file），并通过 IFP 内置的功能处理数据展示和流转事宜。

二、环境依赖

2.1 操作系统依赖

IFP 的开发和测试 OS 为 **CentOS Linux release 7.9.2009 (Core)**。

centos6/centos7/centos8，及对应的 redhat 版本应该都可以运行，主要的潜在风险在于系统库版本差异可能会影响部分组件的运行。

建议使用 centos7.9 操作系统。

2.2 python 版本依赖

IFP 基于 python 开发，其开发和测试的 python 版本为 **python3.8.8**，实际使用 **Anaconda3-2021.05**。

不同版本的 python 会有 python 库版本问题。

建议优先使用 Anaconda3-2021.05 版本，或者根据当前使用版本安装好依赖的 python 库。

2.3 集群管理系统

IFP 的多任务并行控制主要通过集群管理系统来实现，所以需要安装 LSF 或者 openlava 等集群管理软件。

2.4 共享存储

由于 IFP 可以借助集群管理系统实现多服务器的资源使用，因此依赖共享存储以保证多服务器上所访问到的数据一致性。

三、工具安装及配置

3.1 工具安装

工具安装之前，首先参照第二章“环境依赖”满足 IFP 的环境依赖关系。

将安装包拷贝到安装目录，并给与合适的目录名。

安装包下的文件和目录如下。

```
Bash
[liyanqing.1987@n212-206-218 tools]$ cd ic_flow_platform/
[liyanqing.1987@n212-206-218 ic_flow_platform]$ ls
bin common config data function install.py README
requirements.txt scripts tools
```

首先确认当前 python 版本正确。

```
Bash
[liyanqing.1987@n212-206-218 ic_flow_platform]$ which python3
/ic/software/tools/python3/3.8.8/bin/python3
```

基于安装包中的 requirements.txt 安装 python 依赖库。（可能需要 root 权限）

```
Bash
[root@n212-206-218 ic_flow_platform]# pip3 install -r
requirements.txt
Requirement already satisfied: certifi==2022.9.24 in
/ic/software/tools/python3/3.8.8/lib/python3.8/site-packages (from -
r requirements.txt (line 1)) (2022.9.24)
Requirement already satisfied: charset-normalizer==2.1.1 in
/ic/software/tools/python3/3.8.8/lib/python3.8/site-packages (from -
r requirements.txt (line 2)) (2.1.1)
...
```

在安装目录下，使用命令“python3 install.py”安装 IFP。

```
Bash
[liyanqing.1987@n212-206-218 ic_flow_platform]$ python3 install.py
>>> Generate wrapper script "bin/ifp" ...
```

```
>>> Generate wrapper script
"action/check/scripts/gen_checklist_scripts" ...
>>> Generate wrapper script
"action/check/scripts/gen_checklist_summary" ...
>>> Generate wrapper script "action/check/scripts/ic_check" ...
>>> Generate wrapper script
"action/check/scripts/view_checklist_report" ...
>>> Generate config file
"/ic/data/usr/liyanqing.1987/tools/ic_flow_platform/config/config.py"
..
>>> Generate top sh environment file
"/ic/data/usr/liyanqing.1987/tools/ic_flow_platform/config/env.sh" .
..
>>> Generate top csh environment file
"/ic/data/usr/liyanqing.1987/tools/ic_flow_platform/config/env.csh"
...
>>> Update EXPECTED_PYTHON/IFP_INSTALL_PATH settings for specified
tools ...
>>> Install tool "lsfMonitor" ...

Install successfully!
```

3.2 工具配置

有如下文件需要配置。

3.2.1 config/config.py

config.py 是 ifp 的主配置文件。

原始设置为：

```
Bash
# Only default_yaml_administrators can edit default.yaml on ifp GUI
directory.
default_yaml_administrators = ""

# send result command
send_result_command = ""
```

当前设置为：

```
Bash
# Only default_yaml_administrators can edit default.yaml on ifp GUI
directory.
default_yaml_administrators = "liyanqing.1987 jiexi jingfuyi
weikangliang"

# send result command
send_result_command = "/ic/software/cad_tools/bin/send_lark -c
RESULT -r USER"
```

default_yaml_administrators，用于指定谁可以在 ifp 的 GUI 界面中修改 default.yaml，多用户之间用空格隔开。

send_result_command，在 RUN 这个行为后执行什么命令，可以不设，如果设置的话，支持 RESULT 和 USER 两个变量，其中 RESULT 会被 Task 当前状态的文本取代，USER 会被当前用户名取代。

3.2.2 config/default.yaml

default.yaml 用于定义指定 Task 的默认 Action 属性。

原始设置为：

```
Bash
## Format ##
# VAR:
#     key: value
#
# TASK:
#     flow:vendor:task:
#         action:
#             attribute: value
#
#
## Supported Variables (default) ##
# CWD: <ifp start directory>
# IFP_INSTALL_PATH: <ifp install path>
# BLOCK: <block name>
# VERSION: <version name>
# FLOW: <flow name>
```

```

# VENDOR: <vendor name>
# BRANCH: <branch name>
# TASK: <task name>
#
#
## Supported action ##
# BUILD/RUN/CHECK/SUMMARY/POST_RUN/RELEASE
#
#
## Supporteddd action attribute ##
# PATH/CMD/COMMAND/RUN_METHOD/VIEWER/REPORT_FILE
#
#
## Example ##
# VAR:
#     BSUB_QUEUE: ai_syn
#     DEFAULT_PATH:
${CWD}/${BLOCK}/<VERSION>/<FLOW>/<VENDOR>/<BRANCH>
#
# TASK:
#     synthesis:synopsys:intopt:
#         BUILD:
#             PATH: $DEFAULT_PATH
#             COMMAND: make build
#         RUN:
#             PATH: ${DEFAULT_PATH}/dc
#             COMMAND: make run_initopt
#             RUN_METHOD: bsub -q $BSUB_QUEUE -n 8 -R
"rusage[mem=50000]"
#         CHECK:
#             PATH: ${DEFAULT_PATH}/dc
#             COMMAND:
${IFP_INSTALL_PATH}/function/check/syn/synopsys/syn_synopsys.syn_dc.
py -b ${BLOCK}
#             VIEWER:
${IFP_INSTALL_PATH}/function/check/tools/view_checklist_report.py -i
#             REPORT_FILE: file_check/file_check.rpt
#         SUMMARY:
#             PATH: ${DEFAULT_PATH}/dc
#             COMMAND:
${IFP_INSTALL_PATH}/function/summary/collect_syn_qor.py
#             VIEWER: /bin/soffice
#             REPORT_FILE: syn_qor.xlsx
#         POST_RUN:

```

```
#          PATH: ${DEFAULT_PATH}/dc
#          COMMAND: make post_run
#          RUN_METHOD: bsub -q $BSUB_QUEUE -n 8 -R
"rusage[mem=50000]"
#          RELEASE:
#          PATH: ${DEFAULT_PATH}/dc
#          COMMAND: make release
```

当前设置为：

```
Bash
...
VAR:
  DEFAULT_PATH: ${CWD}/${BLOCK}/${BLOCK}_${VERSION}_${BRANCH}

TASK:
  gen_dir:common:build:
    BUILD:
      PATH: $CWD
      COMMAND: ./gen_block_run_dir.pl -c
${BLOCK}.block_flow.configure

  syn:synopsys:lint:
    RUN:
      PATH: $DEFAULT_PATH
      COMMAND: make lint run_local=1
      RUN_METHOD: bsub -q comet -n 8 -R "rusage[mem=80000]" -Is
...
```

对于指定的 Task，比如 Flow(syn)/Vendor(synopsys)/Task(lint)，它的 BUILD/RUN/CHECK/SUMMARY/POST_RUN/RELEASE 分别干什么，用户可以在用户配置文件 ifp.cfg.yaml 中定义，但是这样会造成 ifp.cfg.yaml 的配置过于复杂，为了简化逻辑，可以让 syn 的流程管理员把相关信息提前定义到 default.yaml 中。

3.2.3 config/env.*

env.csh 和 env.sh 分别用于定义 c/tcsh 和 sh/bash 中断中的用户默认环境变量设置，一般用于指定 EDA 工具版本，下面以 env.sh 为例。

原始设置为：

```
Bash
# Set python3 path.
export PATH=/ic/software/tools/anaconda/Anaconda3-2021.05/python_bin:$PATH

# Set ic_flow_platform install path.
export
IFP_INSTALL_PATH=/ic/software/cad_tools/flows/ic_flow_platform

#### Default EDA tool settings ####
# Set default TESSANT setting.

# Set default DC setting.

# Set default GENUS setting.

# Set default FORMALITY setting.

# Set default LEC setting.

# Set default PT setting.

# Set default TEMPUS setting.

# Set default ICC2 setting.

# Set default INNOVUS setting.

#####

# Set lsfMonitor path.
export
PATH=/ic/software/cad_tools/flows/ic_flow_platform/tools/lsfMonitor/monitor/bin:$PATH

# Set default soffice path.
```

当前设置为：

```
Bash
# Set python3 path.
```

```

export PATH=/ic/software/tools/anaconda/Anaconda3-
2021.05/python_bin:$PATH

# Set ic_flow_platform install path.
export
IFP_INSTALL_PATH=/ic/software/cad_tools/flows/ic_flow_platform

#### Default EDA tool settings ####
# Set default TESSANT setting.

# Set default DC setting.
module load dc_shell/T-2022.03-SP3

# Set default GENUS setting.
module load genus/21.14-s082_1

# Set default FORMALITY setting.

# Set default LEC setting.

# Set default PT setting.

# Set default TEMPUS setting.

# Set default ICC2 setting.

# Set default INNOVUS setting.

#####

# Set lsfMonitor path.
export
PATH=/ic/software/cad_tools/flows/ic_flow_platform/tools/lsfMonitor/
monitor/bin:$PATH

# Set default soffice path.

```

我们只是增加了 dc 和 genus 的默认设置，即可在 ifp 的 ENV 页的 PATH 项中看到 dc 和 genus 的默认路径信息。

四、工具使用

4.1 工具载入

在字节芯片设计环境中，IFP 已经被安装到 /ic/software/cad_tools/flows/ic_flow_platform 中，并且已经配置了 module files 以方便用户引用。

IFP 的主程序为 ifp，可以通过 module load cad 获取。

Bash

```
[liyanqing.1987@n212-206-218 ~]$ module load cad
[liyanqing.1987@n212-206-218 ~]$ which ifp
/ic/software/cad_tools/bin/ifp
```

工具的帮助信息如下。

Bash

```
[liyanqing.1987@n212-206-218 test]$ ifp -h
usage: ifp.py [-h] [-config_file CONFIG_FILE] [-build] [-run] [-check] [-summary] [-post_run] [-release] [-d]

optional arguments:
  -h, --help                show this help message and exit
  -config_file CONFIG_FILE, --config_file CONFIG_FILE
                           Specify the configure file, default is
                           "<CWD>/ifp.cfg.yaml".
  -build, --build            Enable build function, create run
                           directories/files.
  -run, --run                Enable run function, run tasks/corners.
  -check, --check            Enable check function, check results of
                           tasks/corners with specified checklist.
  -summary, --summary        Enable summary function, get summary report
                           with specified information requirement.
  -post_run, --post_run      Enable post run function, execute user
                           specified command.
  -release, --release        Enable release function, release current
                           result to release directory.
  -d, --debug                Enable debug mode, will print more useful
                           messages.
```


--help: 打印帮助信息。

--config_file: 指定用户配置文件，用来声明跑什么任务，默认为“./ifp.cfg.yaml”。

--build: 功能项，开启 build 功能。

--run: 功能项，开启 run 功能。

--check: 功能项，开启 check 功能。

--summary: 功能项，开启 check 功能。

--post_run: 功能项，开启 post_run 功能。

--release: 功能项，开启 release 功能。

--debug: 开启 debug 功能。打印更多的中间执行过程到桌面上。

其中所有的功能项都可以直接在 GUI 界面上操作，此处只是为了增加非交互式的 command_line 接口。

4.2 配置文件

用户通过 user config file 跟 ifp 交互，告诉 ifp：

- 我想要跑什么任务。(Block/Version/Flow/Vendor/Branch/Task)
- 这些任务有哪些行为是可执行的。(Build/Run/Check/Summay/Post_run/Release)
- 任务行为如何执行。(比如 Run，那么需要在什么 PATH 下面执行什么 COMMAND。)

4.2.1 编辑用户配置

默认用户配置为运行目录下的 ifp.cfg.yaml 文件，样式如下。

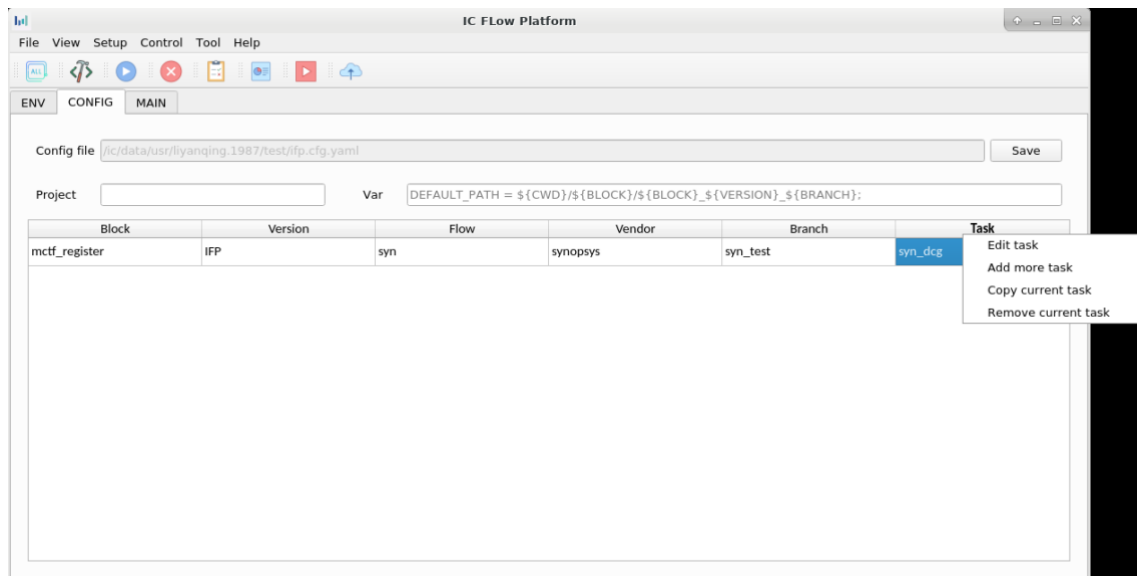
```
YAML
PROJECT: comet
VAR: {}
BLOCK:
  mctf_register:
    IFP(RUN_ORDER=gen_dir,syn,fv|sta):
      gen_dir:
```

```

common:
  test(RUN_TYPE=serial):
    build: {}
syn:
  synopsys:
    test(RUN_TYPE=serial):
      syn_dcg: {}
      dataout: {}
    test2(RUN_TYPE=serial):
      syn_dcg: {}
      dataout: {}
  cadence:
    test(RUN_TYPE=serial):
      syn_genus: {}
    test2(RUN_TYPE=serial):
      syn_genus: {}

```

ifp 主页面的 CONFIG 页支持便捷设置 user config file。



如果 ifp 启动的时候没有特别指定，Config file 默认为当前路径下的 ifp.cfg.yaml 文件，Save 的时候可以重新指定。

Project 是项目信息。

Var 用于指定环境变量信息。

CONFIG 页面主界面可以直接编辑 Block/Version/Flow/Vendor/Branch/Task 信息，Task 具体的行为 (Build/Run/Check/Summay/Post_run/Release)，以及具体行为的属性 (PATH/COMMAND/RUN_METHOD/VIEWER/REPORT_FILE)，则可以通过右键 Edit task 功能编辑。

Detailed setting for syn/synopsys/syn_dcg

Env setting :

	Item	Value
1	\$CWD	/ic/data/usr/liyanqing.1987/test
2	\$IFP_INSTALL_PATH	/ic/software/cad_tools/flows/ic_flow_platform
3	\$BLOCK	mctf_register
4	\$VERSION	IFP
5	\$FLOW	syn

Flow setting :

	Item	Value
1	* BUILD	
2	PATH	
3	COMMAND	
4	* RUN	
5	PATH	\$DEFAULT_PATH
6	COMMAND	make syn_dcg run_local=1
7	RUN_METHOD	bsub -q comet -n 8 -R "rusage[mem=80000]" -ls

save cancel

上图我们可以看到，每个 Task 都有很多行为及行为属性需要编辑，这样对用户来说包含较大的工作量，因为同一类型 Task 的行为及行为属性配置都是固定的，所以 flow owner 可以提前将这些设置放置到一个默认的配置文件中，ifp 在读取用户配置的时候直接引用（如果用户没有配置），这样可以极大减轻配置工作量。

4.2.2 编辑默认 Task 属性配置

默认 Task 属性配置位于 ifp 安装目录下的 config/default.yaml 中，可以通过 ifp 自带的 config default yaml 工具编辑。（菜单栏 Tool -> Config Default Yaml）

Default Config

Config path :

Save

Var setting :

Item	Value
1 DEFAULT_PATH	\${CWD}/\${BLOCK}/\${BLOCK}_\${VERSION}_\${BRANCH}
2	

Flow setting :

All

gen_dir:common:build

Flow

gen_dir

Vendor

common

Task

build

Item	Value
1 * BUILD	
2 PATH	\$CWD
3 COMMAND	./gen_block_run_dir.pl -c \${BLOCK}.block_flow.configure
4 * RUN	
5 PATH	

Update setting for gen_dir/common/build

delete

cancel

只有指定的流程管理员可以直接编辑 default.yaml，以防止数据错乱。

4.2.3 便捷 Config file 生成

为了减轻大家的 config file 编辑工作量，针对 ByteDance IC 当前在用的设计流程（主要是 syn/fv/sta），我们开发了 pre_ifp 来自动生成 ifp 的用户配置文件。

工具名为 pre_ifp，在“module load cad”后可以直接载入。

```

Bash
[liyanqing.1987@n212-206-194 ~]$ module load cad
[liyanqing.1987@n212-206-194 ~]$ which pre_ifp
/ic/software/cad_tools/bin/pre_ifp

```

pre_ifp 的帮助信息如下。

```

Bash
[liyanqing.1987@n212-206-194 ~]$ pre_ifp -h

```

```
usage: pre_ifp [-h] [-p PROJECT] [-d DESIGN [DESIGN ...]]

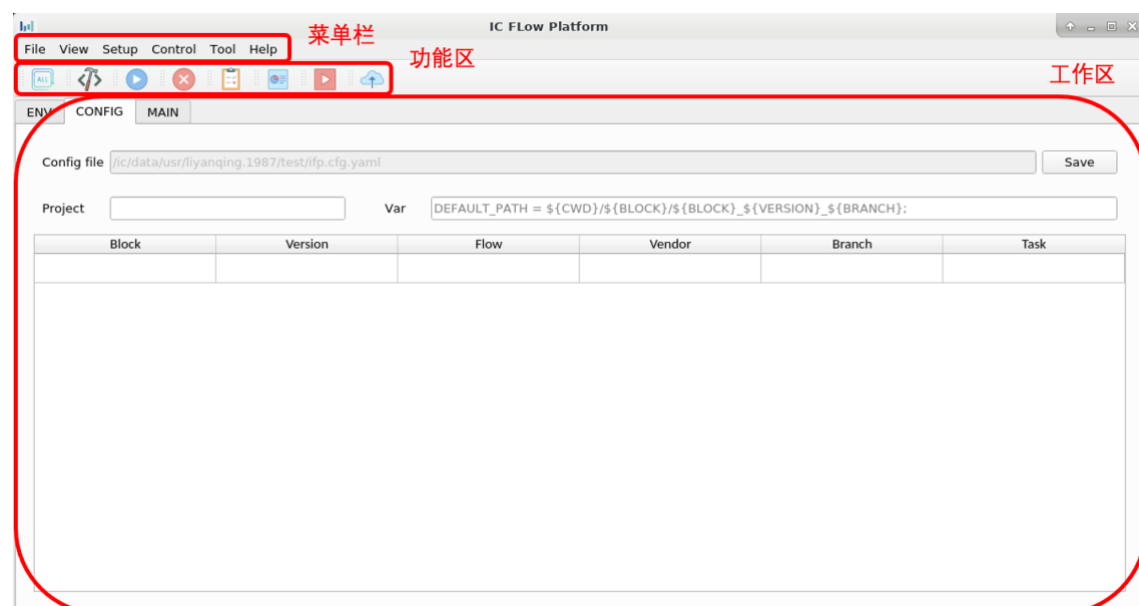
optional arguments:
  -h, --help            show this help message and exit
  -p PROJECT, --project PROJECT
                        Specify project.
  -d DESIGN [DESIGN ...], --design DESIGN [DESIGN ...]
                        Specify design (block) name(s), generate
                        related configure file(s).
```

关于配置文件的格式及编辑方式，更详细的内容请参见“附二、ifp 的配置文件”。

4.3 界面介绍

在流程运行目录下执行命令“ifp”，可以启动 ifp 的 GUI 界面。

```
Bash
[liyanqing.1987@n212-206-218 test]$ ifp
>>> Generating empty configure file
"/ic/data/usr/liyanqing.1987/test/ifp.cfg.yaml" ...
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to
'/tmp/runtime-liyanqing.1987'
```



默认的启动界面包括三个区域，分为菜单栏、功能区和工作区。其中菜单栏囊括了绝大多数功能和配置选项，功能区则将最常用的任务控制功能放到主界面方便调用，

工作区包括 ENV/CONFIG/MAIN 三个子页，分别用来做环境查看、用户任务配置和运行信息展示。

4.3.1 菜单栏

菜单栏包括 File/View/Setup/Contral/Tool/Help 几部分。

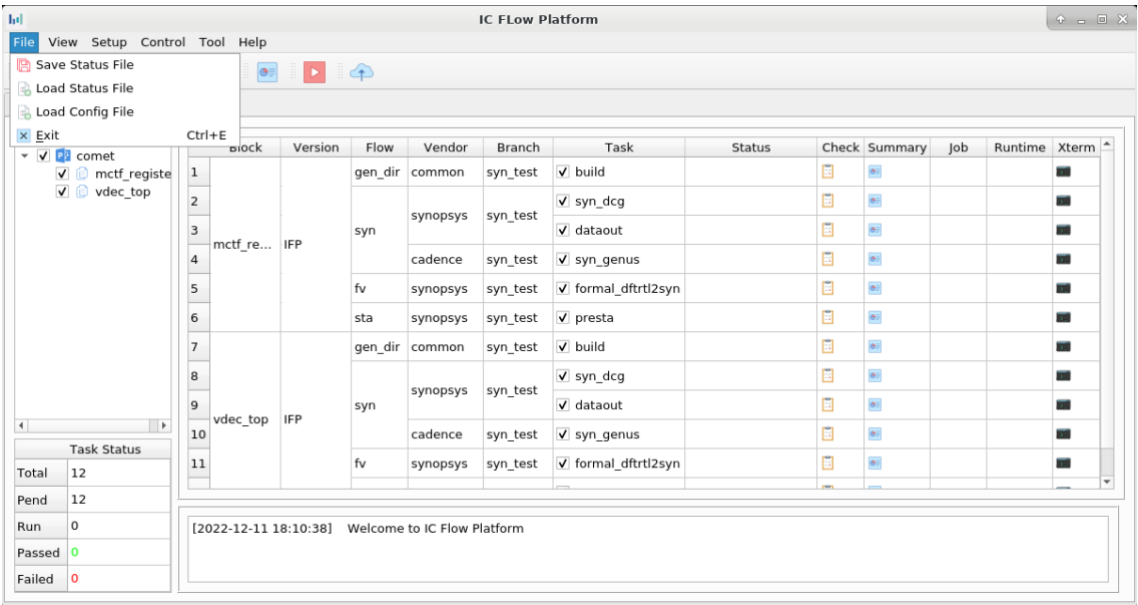
菜单 File

Save Status File: 保存当前任务的 Status/Job/Runtime 等信息到指定的 yaml 文件中。

Load Status File: 载入之前保存的 status file，将 Status/Job/Runtime 等信息直接展开到当前主界面中。

Load Config File: 载入用户配置文件。

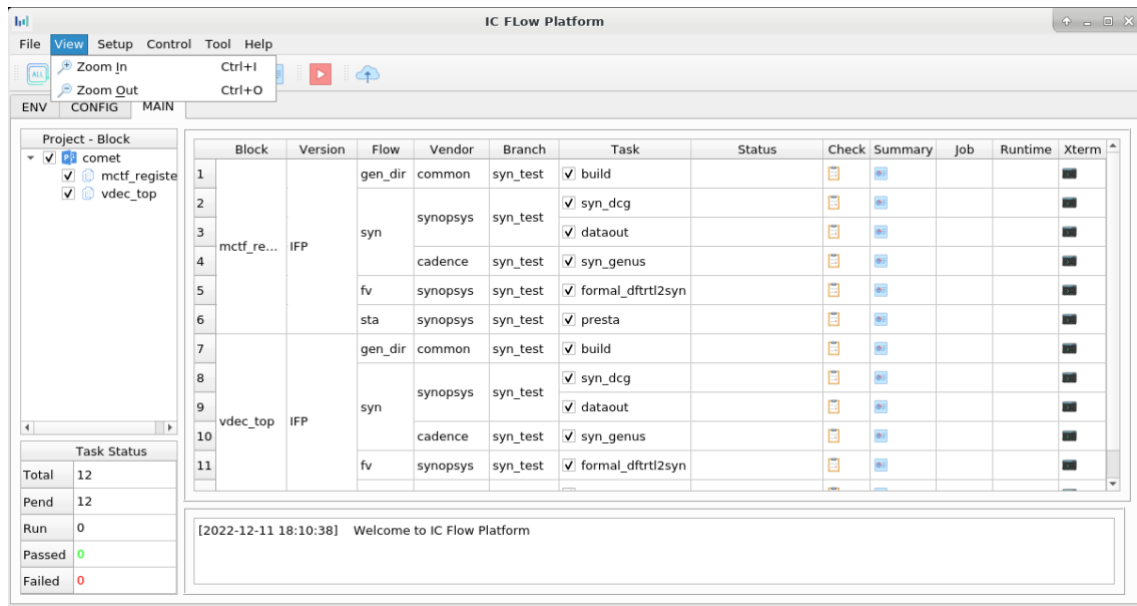
Exit: 退出。



菜单 View

Zoom In: 增加 ifp 主界面的高度。

Zoom Out: 减少 ifp 主界面的高度。



菜单 Setup

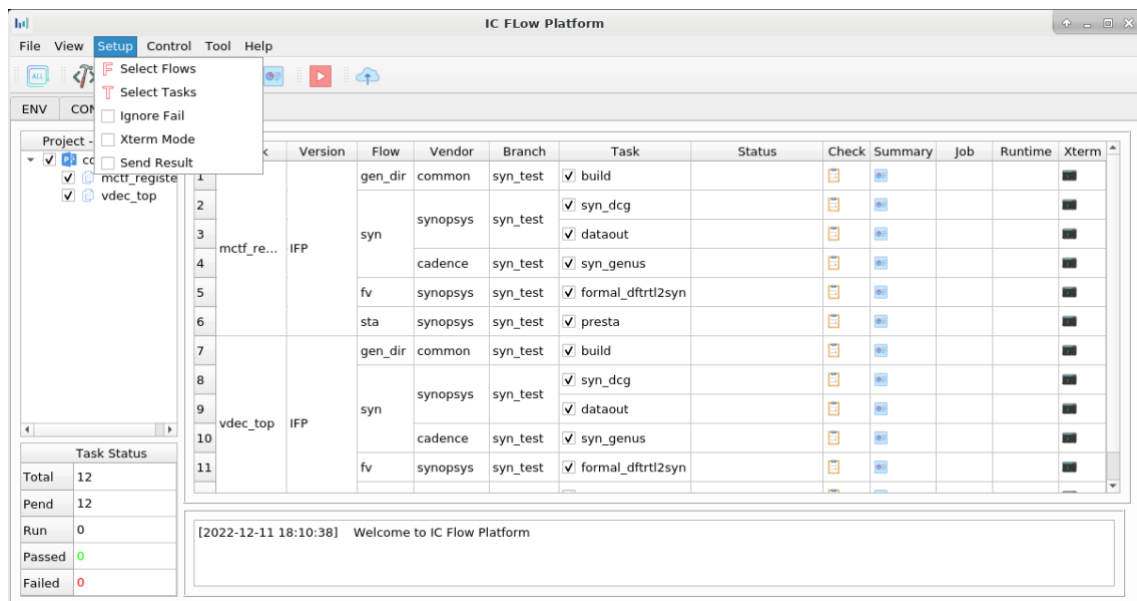
Select Flows: 批量选中主页中的 Flow 项。

Select Tasks: 批量选中主页中的 Task 项。

Ignore Fail: ifp 默认的行为是，串行 Task 中，前者失败后者自动取消，如果选中了 Ignore Fail 选项，则前者失败后后者仍然运行。

Xterm Mode: ifp 中配置的 Task 运行默认是在后台执行的，如果选中了 Xterm Mode，则会在新开 Xterm 执行。（副作用是会在桌面上弹出一堆 Xterm）

Send Result: ifp 的结果默认展示在 GUI 界面左下角的 Task Status 上，如果选中了 Send Result，会把汇总结果信息通过管理员指定的方式发送给用户。（当前是通过 send_lark 发送到用户的飞书上）



菜单 Control

All Steps: 执行如下所有的步骤。

Build: 执行 Build 行为。

Run: 执行 Run 行为。

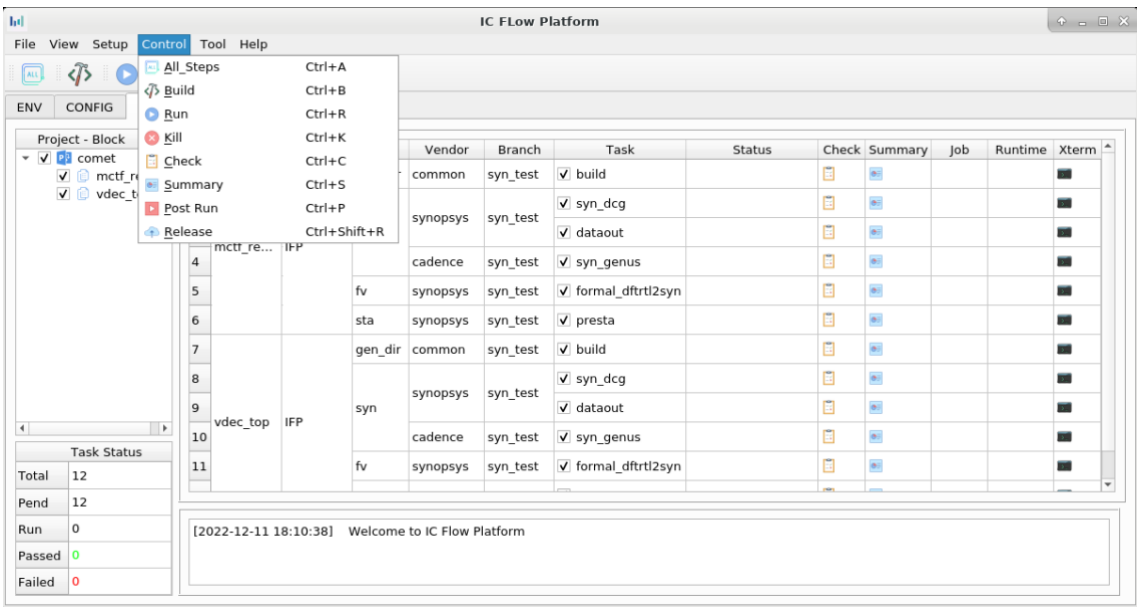
Kill: 如果 job 是 bsub 出去的，kill 所选中的任务。

Check: 执行 Check 行为。

Summary: 执行 Summary 行为。

Post Run: 执行 Post_run 行为。

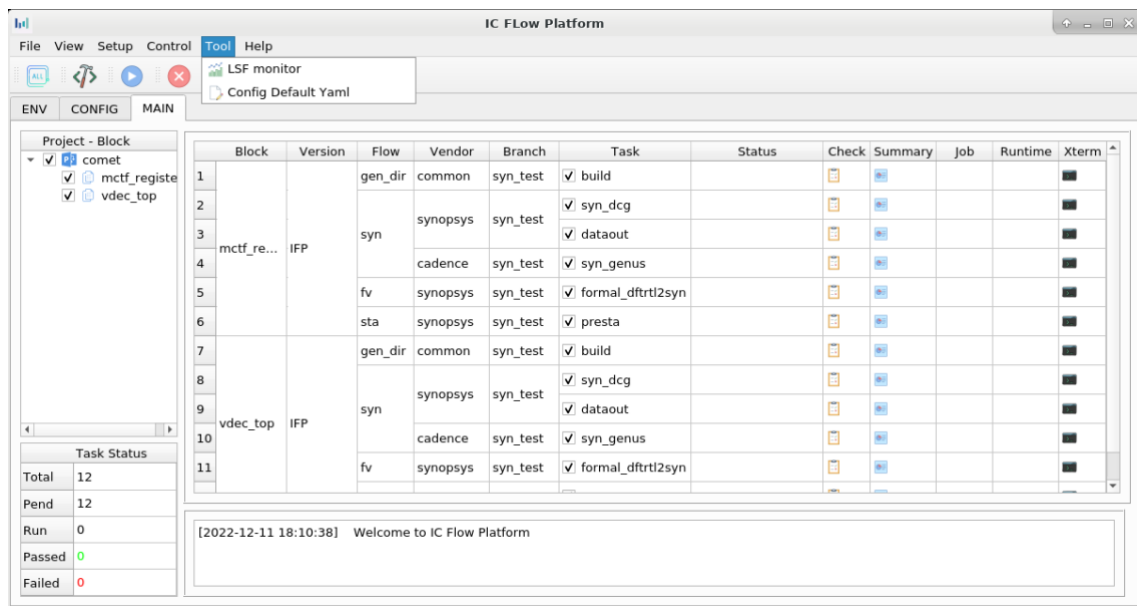
Release: 执行 Release 行为。



菜单 Tool

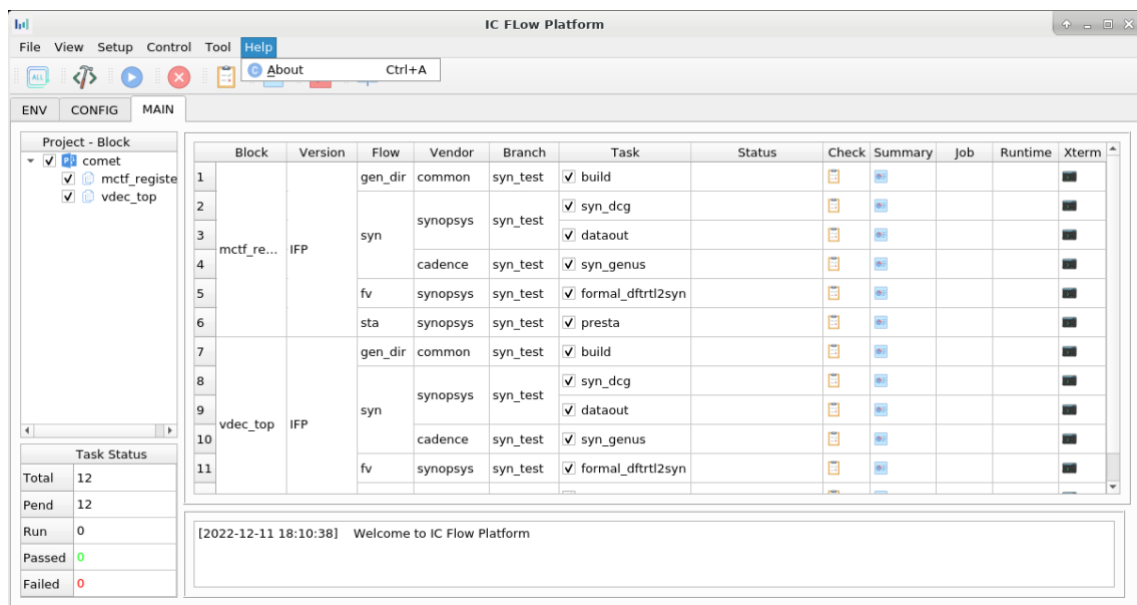
LSF monitor: 执行开源工具 IsfMonitor，用于获取 LSF 的基本信息。

Config Default Yaml: 执行 config_default_yaml 工具，用于编辑默认 Task 属性。
(只有制定管理员有权限)



菜单 Help

About: 工具的基本信息。



4.3.2 功能区

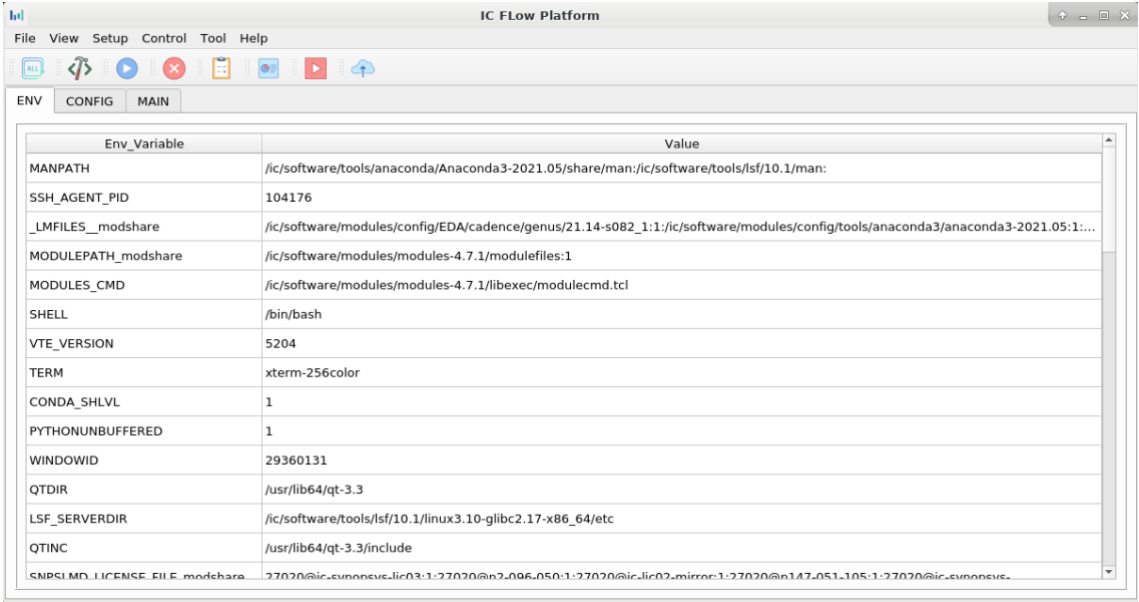
功能区同菜单栏的 Control 菜单内容相同，将这些常用的控制功能直接挪到了功能区。

4.3.3 工作区

工作区是用户的主工作区，主要用于配置和获取各类信息。

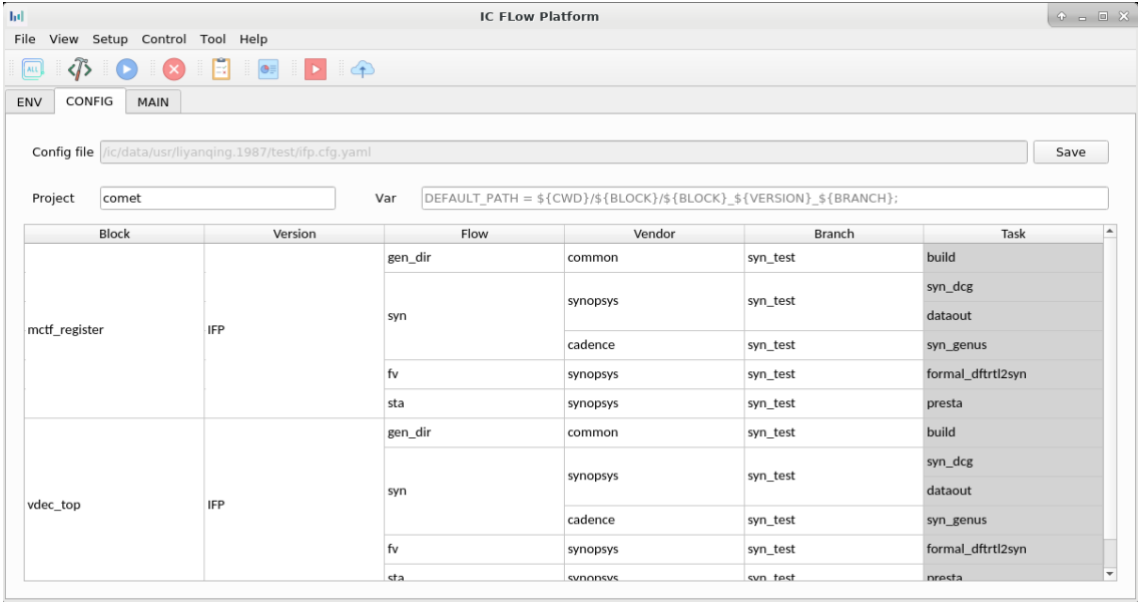
ENV 页

查看当前的环境变量信息，可以用用户环境设置确认。



CONFIG 页

配置和查看用户配置文件，详情参照#4.2 或者附二。



MAIN 页

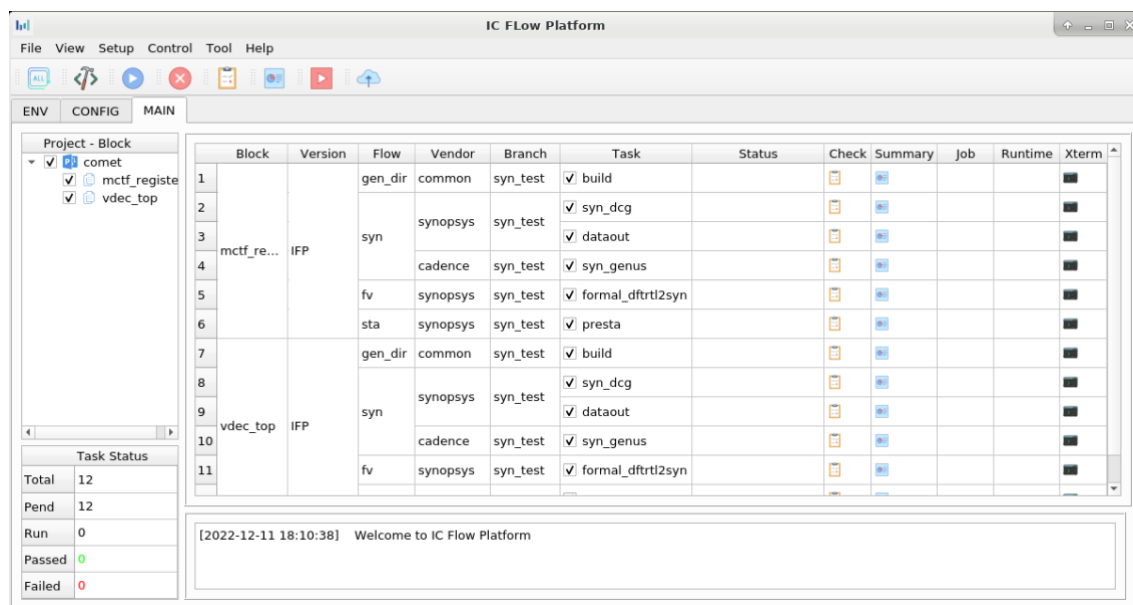
查看用户运行任务的实时信息。

左上角是 project - block 的信息。

左下角是 Task 状态汇总。

右上角是所有 Task 的状态详情。

右下角是用户操作记录。



4.4 工具运行

工具运行的具体步骤如下

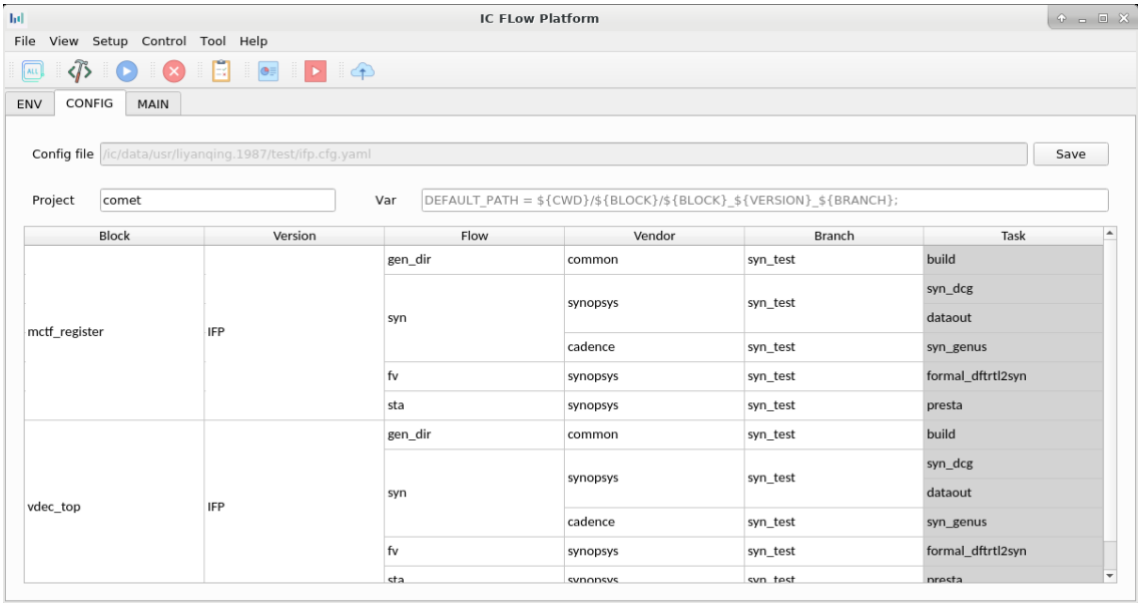
- 进入运行目录。
- 配置用户配置文件。
- 在 ifp 图形界面上执行用户指定的行为。（比如 Build/Run/Check）
- 结果检查和数据核对。

下面是一个简单示例：

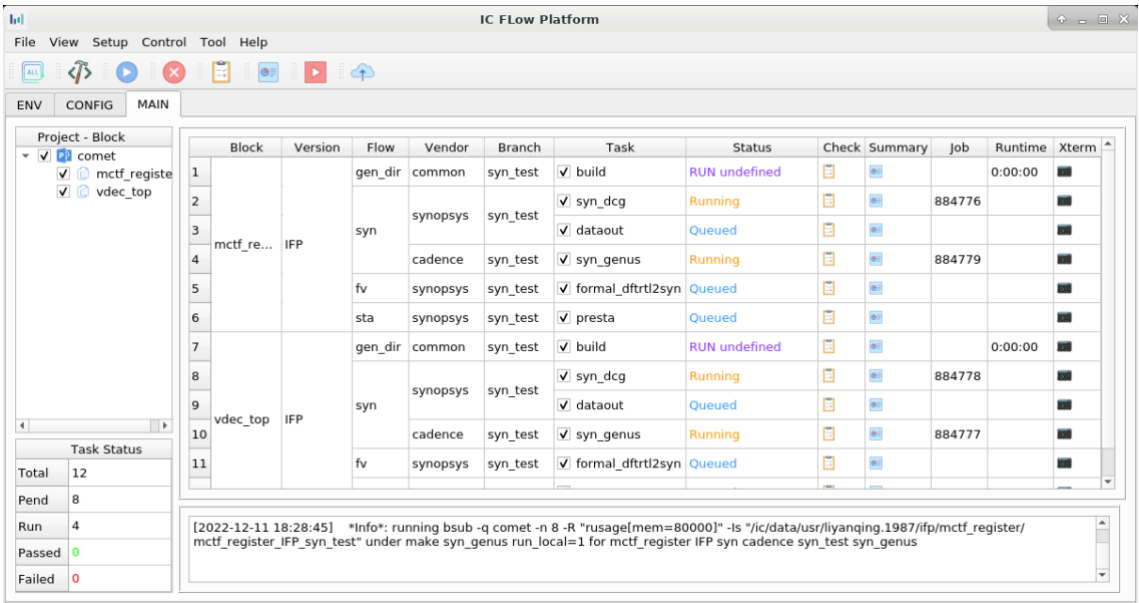
进入运行目录

```
Bash
[liyanqing.1987@n212-206-218 test]$ cd
/ic/data/usr/liyanqing.1987/test
[liyanqing.1987@n212-206-218 test]$ ifp
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to
'/tmp/runtime-liyanqing.1987'
```

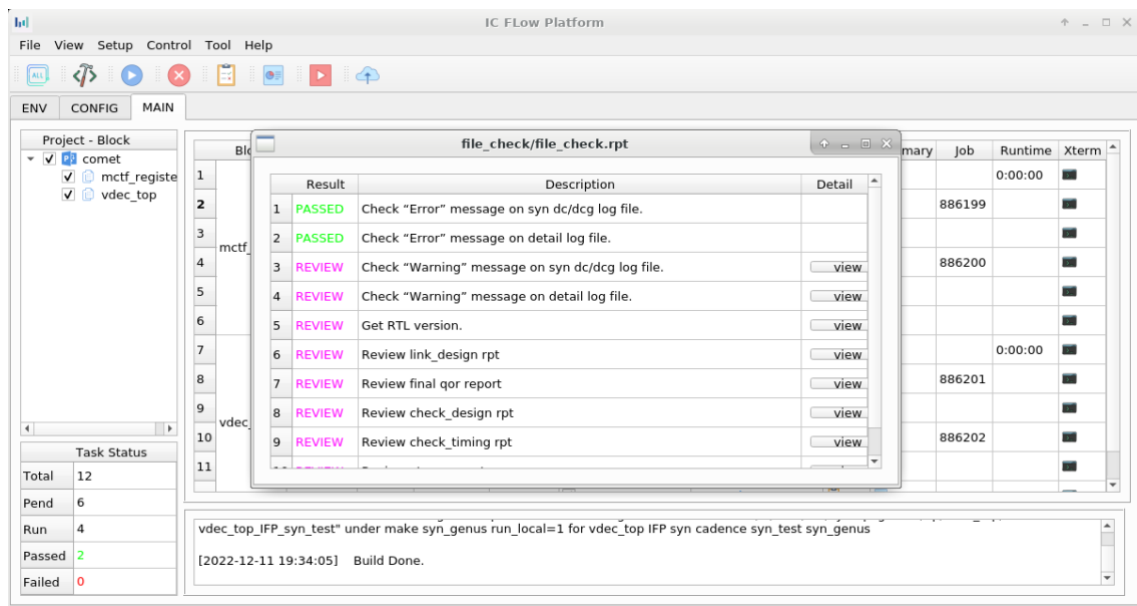
配置用户配置文件



执行指定行为

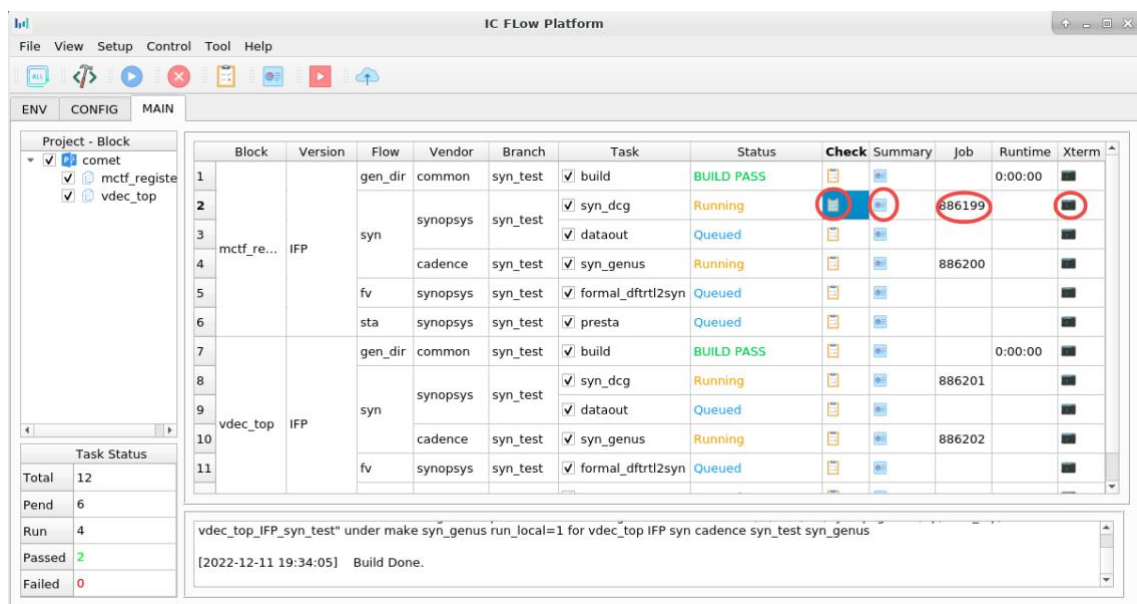


结果检查和数据核对



4.5 其它功能介绍

4.5.1 主界面功能



Check

点击 Task 行的 Check 图标，会对当前 Task 做 checklist 检查，并弹出 checklist 检查报告。（如果已设置）

file_check/file_check.rpt			
	Result	Description	Detail
1	PASSED	Check "Error" message on syn dc/dcg log file.	
2	PASSED	Check "Error" message on detail log file.	
3	REVIEW	Check "Warning" message on syn dc/dcg log file.	view
4	REVIEW	Check "Warning" message on detail log file.	view
5	REVIEW	Get RTL version.	view
6	REVIEW	Review link_design rpt	view
7	REVIEW	Review final qor report	view
8	REVIEW	Review check_design rpt	view
9	REVIEW	Review check_timing rpt	view

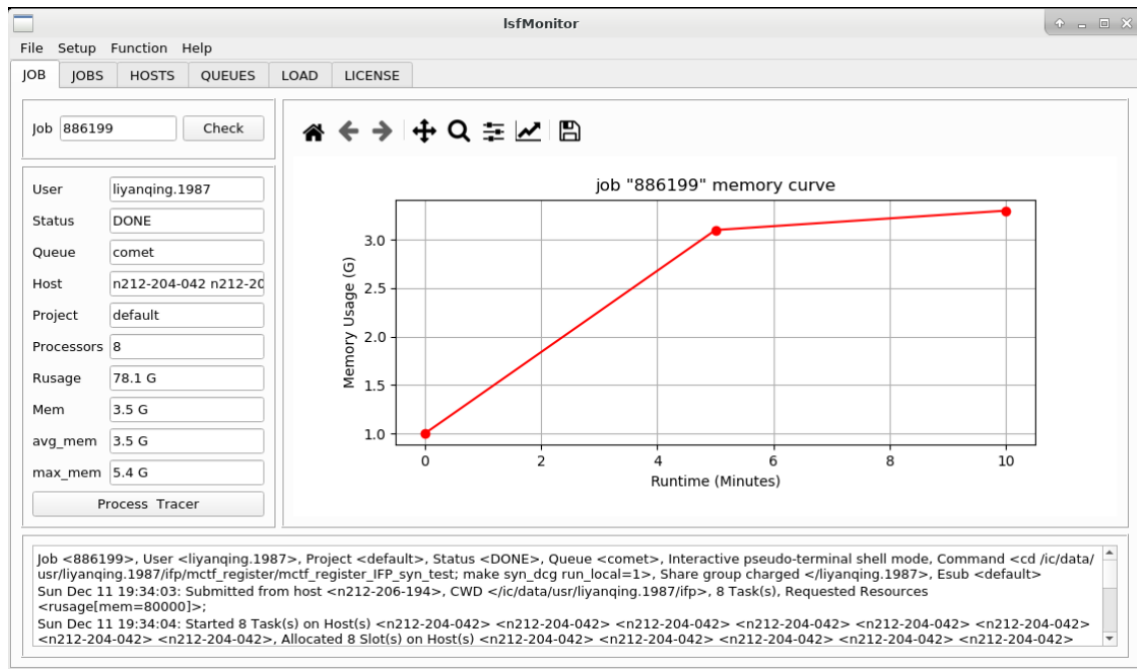
Summary

点击 Task 行的 Summary 图标，会对当前 Task 做 Summary 检查，并弹出 summary 汇总报告。（如果已设置）

Untitled 1 - LibreOffice Calc									
File Edit View Insert Format Sheet Data Tools Window Help									
Arial 10									
A1									
1	Block	Version	Task	Tool Mode	Total Cell Count	Memory Count	Seq Cell Count	BUF/INV Cell Count	C2C WNS
2	mctf_register	mctf_register_IFP_syn_test	syn_dc	DCG	536	0	41	185	0.70

Job

点击 Task 行的 Job 信息，会调用 IsfMonitor 来展示 job 的基本信息。



Xterm

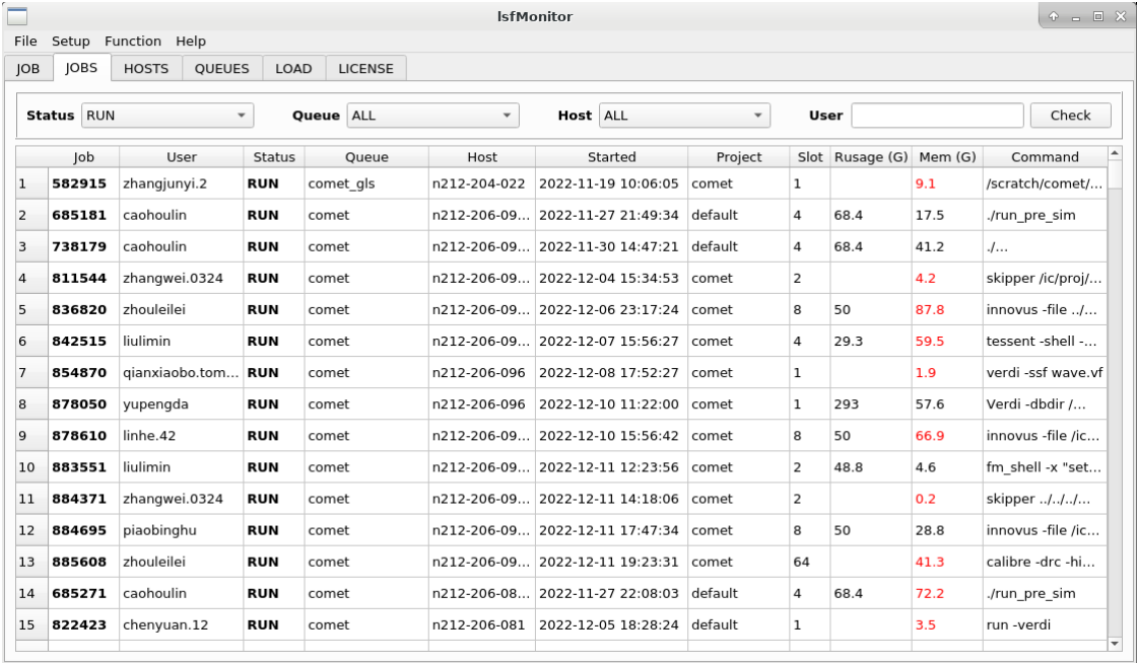
点击 Task 行的 Xterm 图标，会弹出一个 Xterm，并自动进入 Task 的路径。(Run 路径或者 Build 路径，默认是当前运行路径)

The screenshot shows an Xterm terminal window. The title bar reads 'liyanqing.1987@n212-206-194:/ic/data/usr/liyanqing'. The terminal content shows a message about access control being disabled, followed by a prompt '[liyanqing.1987@n212-206-194 mctf_register_IFP_syn_test]\$' with a cursor.

4.5.2 辅助工具

菜单栏 Tool 可以载入 IsfMonitor 和 config_default_yaml 等辅助工具。

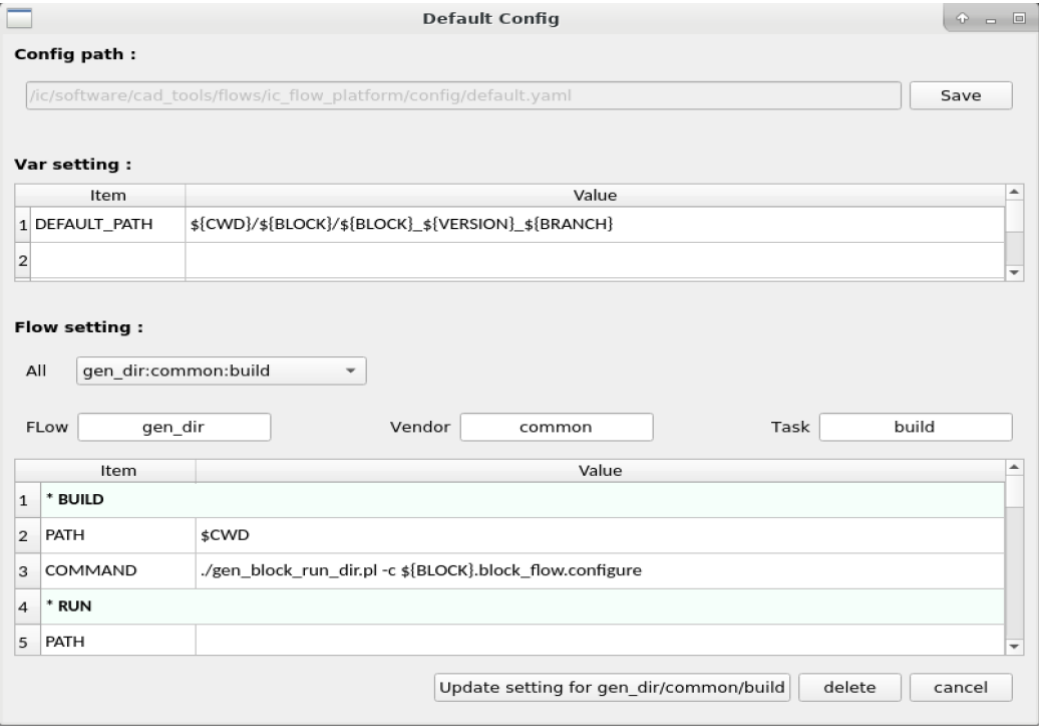
IsfMonitor 是一个 LSF 信息查看的开源工具，可以汇总地查看 LSF 的集群、机器、任务等信息。



The screenshot shows the IsfMonitor application window. It has a menu bar (File, Setup, Function, Help) and a toolbar (JOB, JOBS, HOSTS, QUEUES, LOAD, LICENSE). Below the toolbar are filters for Status (RUN), Queue (ALL), Host (ALL), and User. A 'Check' button is also present. The main area displays a table of jobs with the following columns: Job, User, Status, Queue, Host, Started, Project, Slot, Rusage (G), Mem (G), and Command. The table contains 15 rows of job data.

Job	User	Status	Queue	Host	Started	Project	Slot	Rusage (G)	Mem (G)	Command
1	582915	zhangjunyi.2	RUN	comet_gls	n212-204-022	2022-11-19 10:06:05	comet	1	9.1	/scratch/comet/...
2	685181	caohoulin	RUN	comet	n212-206-09...	2022-11-27 21:49:34	default	4	68.4	./run_pre_sim
3	738179	caohoulin	RUN	comet	n212-206-09...	2022-11-30 14:47:21	default	4	68.4	./...
4	811544	zhangwei.0324	RUN	comet	n212-206-09...	2022-12-04 15:34:53	comet	2	4.2	skipper /ic/proj/...
5	836820	zhouleilei	RUN	comet	n212-206-09...	2022-12-06 23:17:24	comet	8	50	innovus -file ./...
6	842515	liulimin	RUN	comet	n212-206-09...	2022-12-07 15:56:27	comet	4	29.3	tessent -shell -...
7	854870	qianxiaobo.tom...	RUN	comet	n212-206-096	2022-12-08 17:52:27	comet	1	1.9	verdi -ssf wave.vf
8	878050	yupengda	RUN	comet	n212-206-096	2022-12-10 11:22:00	comet	1	293	Verdi -dbdir /...
9	878610	linhe.42	RUN	comet	n212-206-09...	2022-12-10 15:56:42	comet	8	50	innovus -file /ic...
10	883551	liulimin	RUN	comet	n212-206-09...	2022-12-11 12:23:56	comet	2	48.8	fm_shell -x "set...
11	884371	zhangwei.0324	RUN	comet	n212-206-09...	2022-12-11 14:18:06	comet	2	0.2	skipper ./././...
12	884695	piaobinghu	RUN	comet	n212-206-09...	2022-12-11 17:47:34	comet	8	50	innovus -file /ic...
13	885608	zhouleilei	RUN	comet	n212-206-09...	2022-12-11 19:23:31	comet	64	41.3	calibre -drc -hi...
14	685271	caohoulin	RUN	comet	n212-206-08...	2022-11-27 22:08:03	default	4	68.4	./run_pre_sim
15	822423	chenyuan.12	RUN	comet	n212-206-081	2022-12-05 18:28:24	default	1	3.5	run -verdi

config_default_yaml 则是一款默认 Task 属性配置工具，用于修改 IFP 的 default.yaml 文件，详情可以参照#4.2.2 或者附二。



The screenshot shows the Default Config application window. It has a menu bar (File, Setup, Function, Help) and a toolbar (JOB, JOBS, HOSTS, QUEUES, LOAD, LICENSE). Below the toolbar are filters for Status (RUN), Queue (ALL), Host (ALL), and User. A 'Check' button is also present. The main area displays a table of jobs with the following columns: Job, User, Status, Queue, Host, Started, Project, Slot, Rusage (G), Mem (G), and Command. The table contains 15 rows of job data.

Config path :
/ic/software/cad_tools/flows/ic_flow_platform/config/default.yaml Save

Var setting :

Item	Value
1	DEFAULT_PATH \${CWD}/\${BLOCK}/\${BLOCK}_\${VERSION}_\${BRANCH}
2	

Flow setting :

All gen_dir:common:build

Flow gen_dir Vendor common Task build

Item	Value
1	* BUILD
2	PATH \$CWD
3	COMMAND ./gen_block_run_dir.pl -c \${BLOCK}.block_flow.configure
4	* RUN
5	PATH

Update setting for gen_dir/common/build delete cancel

五、范例

5.1 用户环境准备

5.1.1 用户 configure 文件

用户 configure 文件属于 EDA 设置层部分，由 project engineer 自行配置，用来定义需要运行的任务及 EDA 工具配置。以 mid-end flow 为例：

- configure 文件名称：<design>.block_flow.configure
- 文件内容：

```
Bash
#####
#####
### Block Mid End flow configure file
### 2022
#####
#####
# top design name of the block
proj_name: comet
process: TSMC_N7
DESIGN: vdec_top
DESIGN_RENAME:
# version control
RTL_VERSION: IFP_v1.0
SYN_VERSION: test
# data in
RTL_FILELIST:
/ic/proj/comet/jiexi/ifp_test/flist/vdec_top.v
SDC_func:
/ic/proj/comet/jiexi/ifp_test/flist/vdec_top_cons.tcl
SDC_func_add:
SDC_ac_cap_add:
SDC_dc_cap:
SDC_shift:
DFT_CONS:
DFT_SCAN_INSERT:      # scan insert parameter file
DEF_FILE:              # def for physcial synthesis
UPF_FILE:              # upf for low power
# optional flow inputs or user vars:
EXTRA_OPT:
  #PROJ_PATH:
```

```

FM_DFRTL_FILELIST:
$vars(root_dir)/datain/RTL/$vars(DESIGN).list
FM_SYN_NETLIST:
$vars(root_dir)/dataout/NETLIST/$vars(DESIGN).syn.prescan.v.gz
FM_SVF_LIST:
- $vars(root_dir)/dataout/CONS/$vars(DESIGN).compile1.svf
- $vars(root_dir)/dataout/CONS/$vars(DESIGN).compile2.svf
# optional user setting files for each sub-flow
USER_FILE:
#syn_dc:
# specify std cell library name
STD_CELL_LIBRARY:
- tcbn07_bwph240l11p57pd_base_ulvt
# specify hard macro used in current design:
MACRO_LIST:
- sarels0g4s1p184x160m2b1w0c1p0d0l1rm3sdmg0rw01zh
# delay corner
DELAY_CORNER:
- ssgnp_0p675v_0c
# RC corner
RC_CORNER:
- cworst_CCworst_T

```

5.1.2 ifp 配置文件

ifp 需要的配置文件需要 5.1.1 节提到的用户 configure 文件作为交互，以此来生成 ifp.cfg.yaml，方法如下：

```

Bash
[n212-206-218]: pre_ifp -d vdec_top
>>> Generate ifp config file
"/ic/proj/comet/jiexi/ifp_test/ifp.cfg.yaml"
    /ic/software/cad_tools/flows/ic_flow_platform/tools/gen_yaml.py
vdec_top.block_flow.configure

```

生成的 ifp 配置文件(ifp.cfg.yaml)格式如下，可以看出配置中 project/block/version/branch 等相关信息均与用户 configure 文件配置保持一致：

```

Bash

```

```

PROJECT: comet
VAR:
BLOCK:
  vdec_top:
    IFP_v1.0(RUN_ORDER=gen_dir,syn,fv|sta):
      gen_dir:
        common:
          test:
            build:
      syn:
        synopsys:
          test(RUN_TYPE=serial):
            syn_dcg:
            dataout:
        cadence:
          test:
            syn_genus:
      fv:
        synopsys:
          test:
            formal_dftrtl2syn:
      sta:
        synopsys:
          test:
            presta:

```

注意：ifp 配置文件 ifg.cfg.yaml 有两种生成方式：

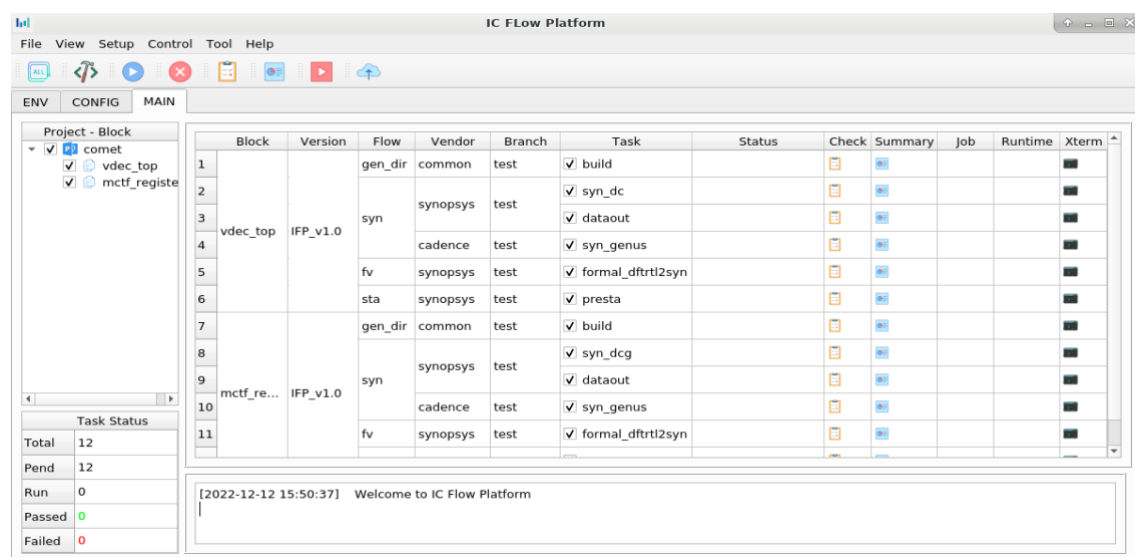
- a. 采用 pre_ifp 脚本生成：基于 ifp 自带的 config default yaml，包含 mid-end 的所有 flow 及对应 task，针对用户不需要进行的任务可直接“vim ifg.cfg.yaml”删除对应行。
- b. 通过 ifp 主页面的 CONFIG 页编辑生成：可根据实际任务需求进行编辑，具体格式及编辑方式，请参见“附二、ifp 的配置文件”。

5.2 操作范例

<design>.block_flow.configure 和 ifg.cfg.yaml 准备完成后，在当前路径执行“ifp”调出 gui 界面：

Bash

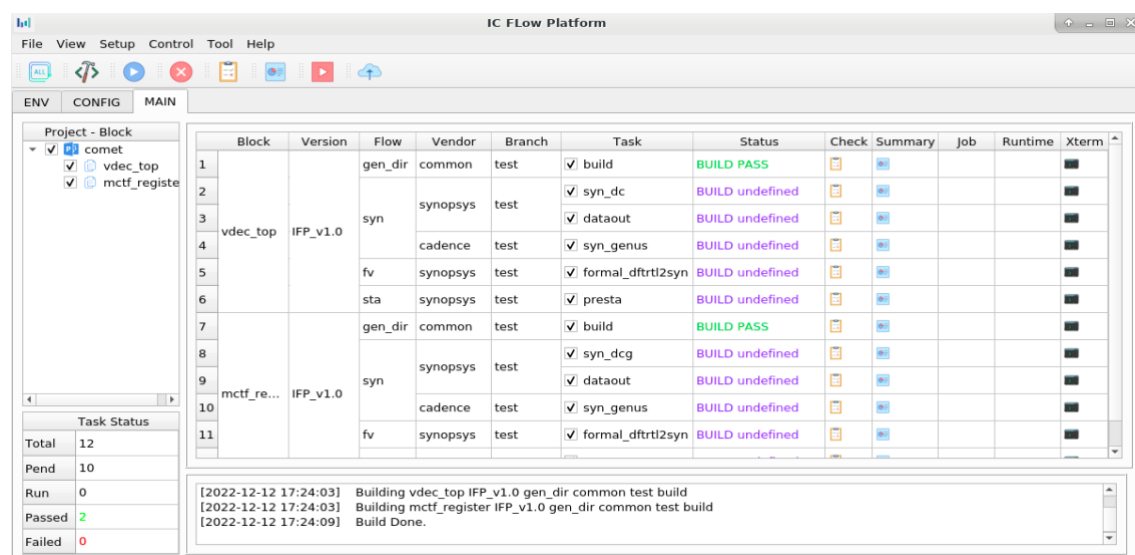
```
[n212-206-218]: /ic/proj/comet/jiexi/ifp_test/ls
mctf_register.block_flow.configure      vdec_top.block_flow.configure
flow                                    gen_block_run_dir.pl
[n212-206-218]: /ic/proj/comet/jiexi/ifp_test/ifp
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to
'/tmp/runtime-jiexi'
```



确认 ifp.cfg.yaml 后，即可通过 ifp 流程平台进行任务运行：

- 可直接点击 gui 界面功能区的 ALL 图标，工具会按照 Build/Run/Check/Summary 顺序自动执行
- 分步骤执行：

1. Build：点击 build 图标，完成目录创建。



Bash

```
[n212-206-218]: /ic/proj/comet/jiexi/ifp_test/ls */*/  
mctf_register/mctf_register_IFP_v1.0_test/:  
clp conformal_eco datain dataout eco formal gen_etm lec  
Makefile ptc setup spyglass sta syn_dc syn_genus vclp  
  
vdec_top/vdec_top_IFP_v1.0_test/:  
clp conformal_eco datain dataout eco formal gen_etm lec  
Makefile ptc setup spyglass sta syn_dc syn_genus vclp
```

1. Run: 点击 Run 图标，调用 EDA 工具提交 job。

IC FLOW Platform

File View Setup Control Tool Help

ENV CONFIG MAIN

Project - Block

- comet
- vdec_top
- mctf_register

Block	Version	Flow	Vendor	Branch	Task	Status	Check	Summary	Job	Runtime	Xterm
1	vdec_top	gen_dir	common	test	✓ build	RUN undefined				0:00:00	
2		syn	synopsys	test	✓ syn_dcg	Running			898386		
3					✓ dataout	Queued					
4			cadence	test	✓ syn_genus	RUN PASS			898387	0:00:03	
5		fv	synopsys	test	✓ formal_dfrt2syn	Queued					
6	mctf_re...	sta	synopsys	test	✓ presta	Queued					
7		gen_dir	common	test	✓ build	RUN undefined				0:00:00	
8		syn	synopsys	test	✓ syn_dcg	Running			898388		
9					✓ dataout	Queued					
10			cadence	test	✓ syn_genus	Running			898385		
11		fv	synopsys	test	✓ formal_dfrt2syn	Queued					

Task Status

Total 12

Pend 8

Run 3

Passed 1

Failed 0

1. Check: 点击 Check 图标，进行 checklist 项目检查。

IC FLOW Platform

File View Setup Control Tool Help

ENV CONFIG MAIN

Project - Block

- comet
- vdec_top
- mctf_register

Block	Version	Flow	Vendor	Branch	Task	Status	Check	Summary	Job	Runtime	Xterm
1	vdec_top	gen_dir	common	test	✓ build	CHECK undefined				0:00:00	
2		syn	synopsys	test	✓ syn_dcg	FAILED			898386	0:21:47	
3					✓ dataout	CHECK undefined				0:00:00	
4			cadence	test	✓ syn_genus	FAILED			898387	0:00:03	
5		fv	synopsys	test	✓ formal_dfrt2syn	PASSED			898652	0:00:17	
6	mctf_re...	sta	synopsys	test	✓ presta	PASSED			899379	0:01:27	
7		gen_dir	common	test	✓ build	CHECK undefined				0:00:00	
8		syn	synopsys	test	✓ syn_dcg	FAILED			898388	0:21:57	
9					✓ dataout	CHECK undefined				0:00:00	
10			cadence	test	✓ syn_genus	FAILED			898385	2:04:38	
11		fv	synopsys	test	✓ formal_dfrt2syn	PASSED			899384	0:00:29	

Task Status

Total 12

Pend 4

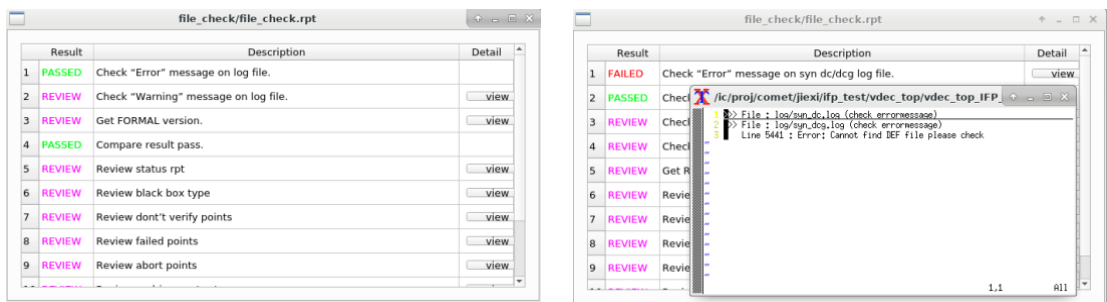
Run 0

Passed 4

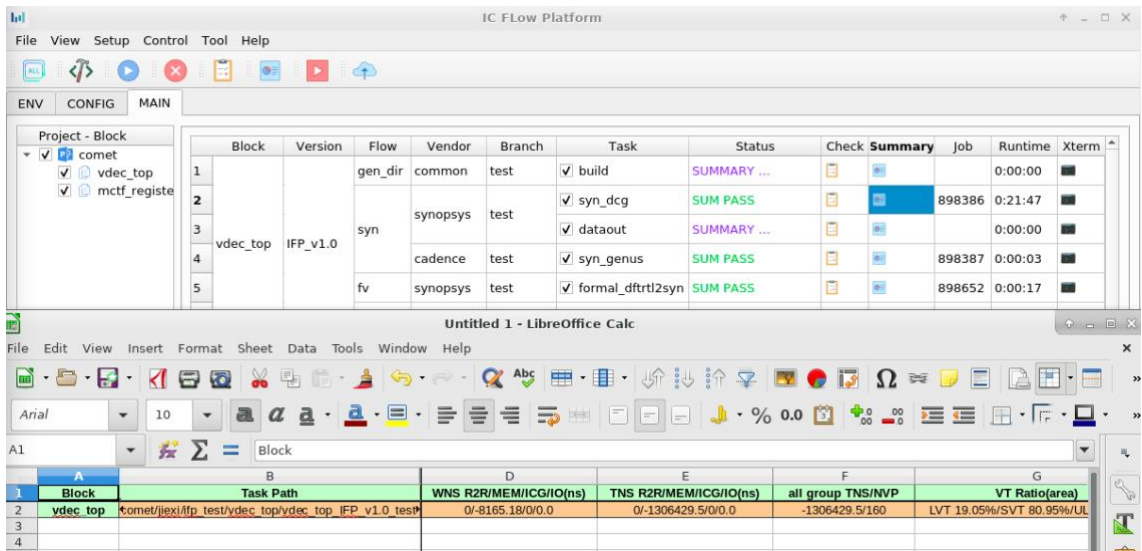
Failed 4

[2022-12-12 20:55:08] >>> Checking results ...

点击对应 task check 列的图标，查看具体 checklist 检查项。



1. Summary: 点击 Summary 图标，调用 qor 脚本自动收集 PPA 数据信息。



附录

附一、变更历史

日期	版本	变更描述	源代码变更
2022.1 2.12	1.0	发布第一个正式 release 版本	无

附二、ifp 的配置文件

1、配置文件采取 **yaml** 文件格式，包含内容具体如下：

```
Bash
PROJECT: <$project>
VAR: <$环境变量>
BLOCK:
  -<design1>:
    <version>:
      <flow>:
        <vendor>:
          <branch>:
            <task>: {}
  -<design2>:
    <version>:
      <flow>:
        <vendor>:
          <branch>:
            <task>: {}
```

```
Bash
PROJECT: comet      => 项目名称
VAR: {}            => 设置环境变量
BLOCK:
  vdec_top:        => design_name
    IFP_v1.0(RUN_ORDER=gen_dir,syn,fv|sta):  => version 信息
```

```

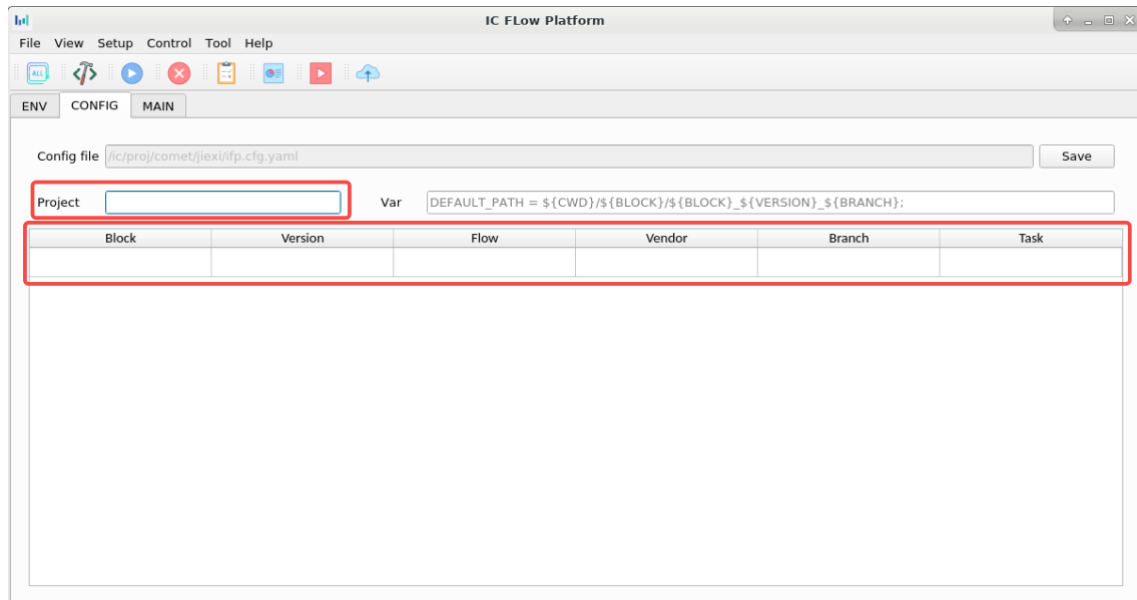
gen_dir:      ⇒ flow 信息
  common:
    test(RUN_TYPE=serial):    ⇒ branch 信息
    build: {}                ⇒ task 信息
syn:          ⇒ flow 信息
  synopsys:
    test(RUN_TYPE=serial):    ⇒ branch 信息
    syn_dcg: {}              ⇒ task 信息
    dataout: {}              ⇒ task 信息
  cadence:
    test(RUN_TYPE=serial):    ⇒ branch 信息
    syn_genus: {}            ⇒ task 信息
fv:           ⇒ flow 信息
  synopsys:
    test(RUN_TYPE=serial):    ⇒ branch 信息
    formal_dftrtl2syn: {}    ⇒ task 信息
sta:          ⇒ flow 信息
  synopsys:      ⇒ vendor 信息
    test(RUN_TYPE=serial):
      presta: {}          ⇒ task 信息
mctf_register: ⇒ design_name
  IFP_v1.0(RUN_ORDER=gen_dir,syn,fv|sta):    ⇒ version 信息
  gen_dir:      ⇒ flow 信息
    common:
      test(RUN_TYPE=serial):
        build: {}        ⇒ task 信息
syn:          ⇒ flow 信息
  synopsys:      ⇒ vendor 信息
    test(RUN_TYPE=serial):    ⇒ branch 信息
    syn_dcg: {}              ⇒ task 信息
    dataout: {}              ⇒ task 信息
  cadence:      ⇒ vendor 信息
    test(RUN_TYPE=serial):    ⇒ branch 信息
    syn_genus: {}            ⇒ task 信息
fv:           ⇒ flow 信息
  synopsys:      ⇒ vendor 信息
    test(RUN_TYPE=serial):    ⇒ branch 信息
    formal_dftrtl2syn: {}    ⇒ task 信息
sta:          ⇒ flow 信息
  synopsys:      ⇒ vendor 信息
    test(RUN_TYPE=serial):    ⇒ branch 信息

```


presta: {} ⇒ task 信息

2、配置文件编辑方式：

如果 ifp 启动路径不存在 ifp.cfg.yaml，可直接“ifp”启动 gui 界面进行编辑。

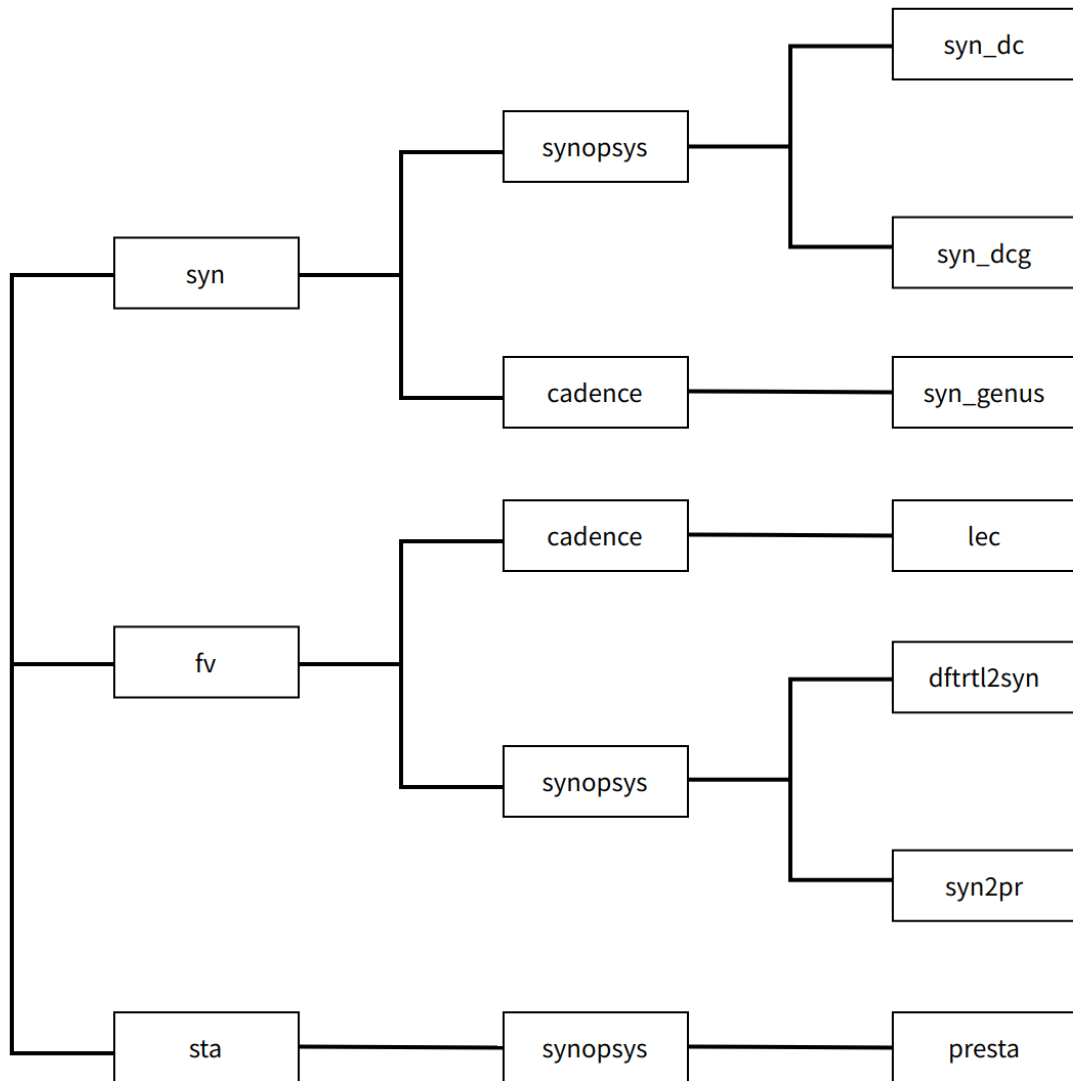


其中 Project 指定项目名称，单击编辑框直接输入；Block/Version/Branch 需要双击点击进行输入；

填写规范与用户 configure 文件（详见 5.1.1）中的对应关系如下：

```
Bash
Project ⇔ proj_name
Block   ⇔ DESIGN
Version ⇔ RTL_VERSION
Branch  ⇔ SYN_VERSION
```

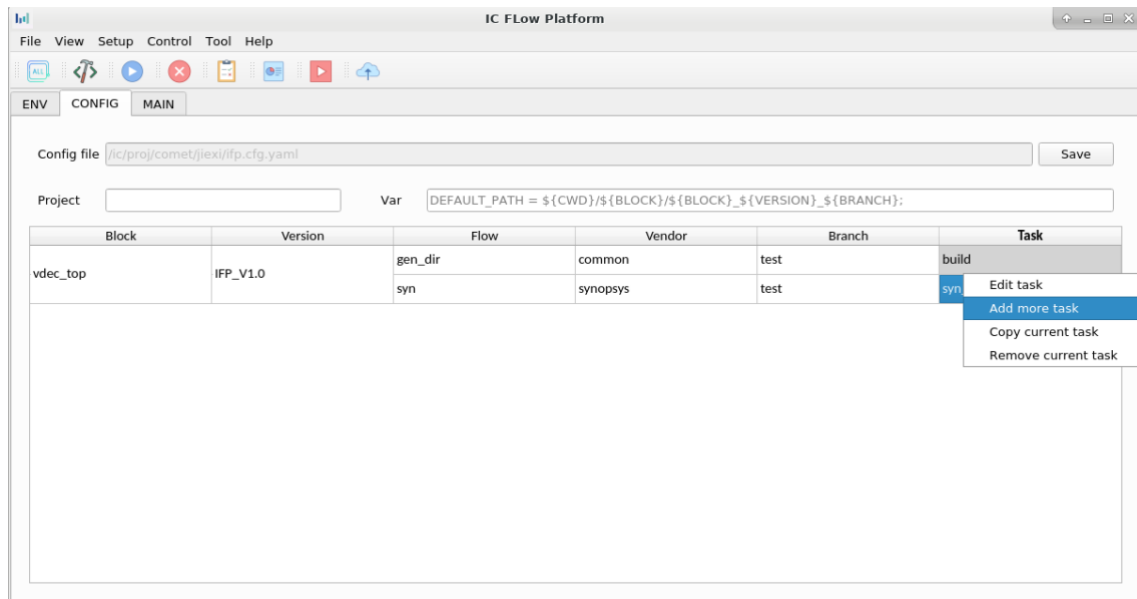
Flow/Vendor/Task 则根据需要运行的 task 进行编辑，如下：



根据以上注意事项编辑 config 界面信息，如下：

- **Task 右键编辑**

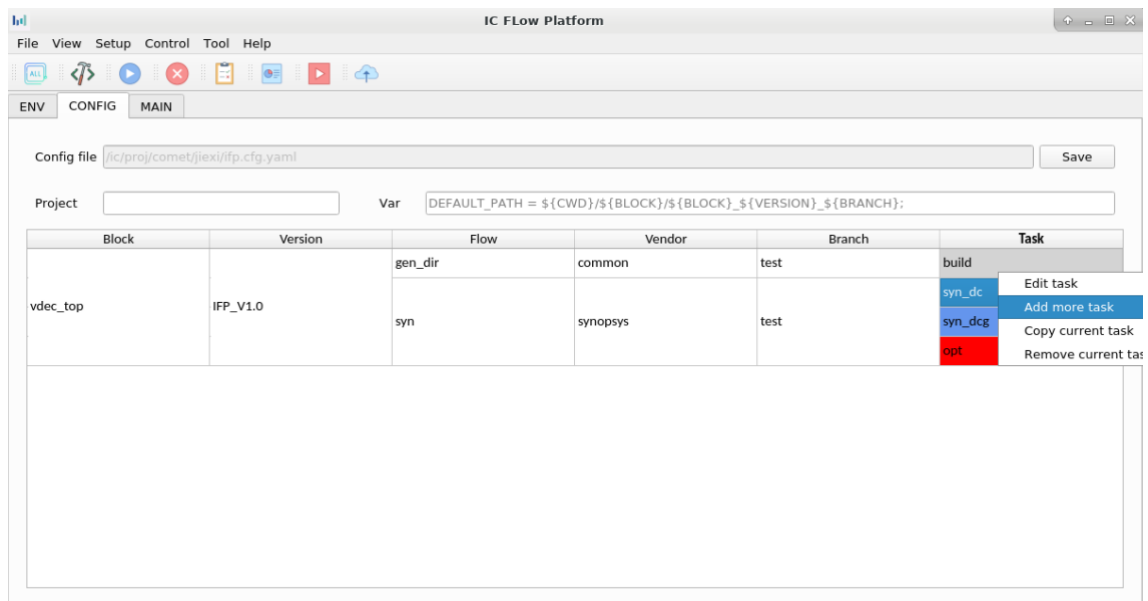
包含“Edit task”，“Add more task”，“Copy current branch”和“Remove current branch”四部分。



Edit task: 用来编辑 Task 的属性，包含“Env setting”和“Flow setting”两项。

Add more task: 新增 task 项目，在 block/version/flow/vendor/branch 一致的情况下，快速增加 task。

- 若新增 task 项在 default config file 预先有定义且用户不修改 task 属性时，界面显灰色背景；
- 若新增 task 项在 default config file 预先有定义但用户修改 task 属性时，界面显蓝色背景；
- 若新增 task 项未在 default config file 预先有定义，界面显红色背景，且 task 对应的 flow setting 均为空，需要用户自行定义；



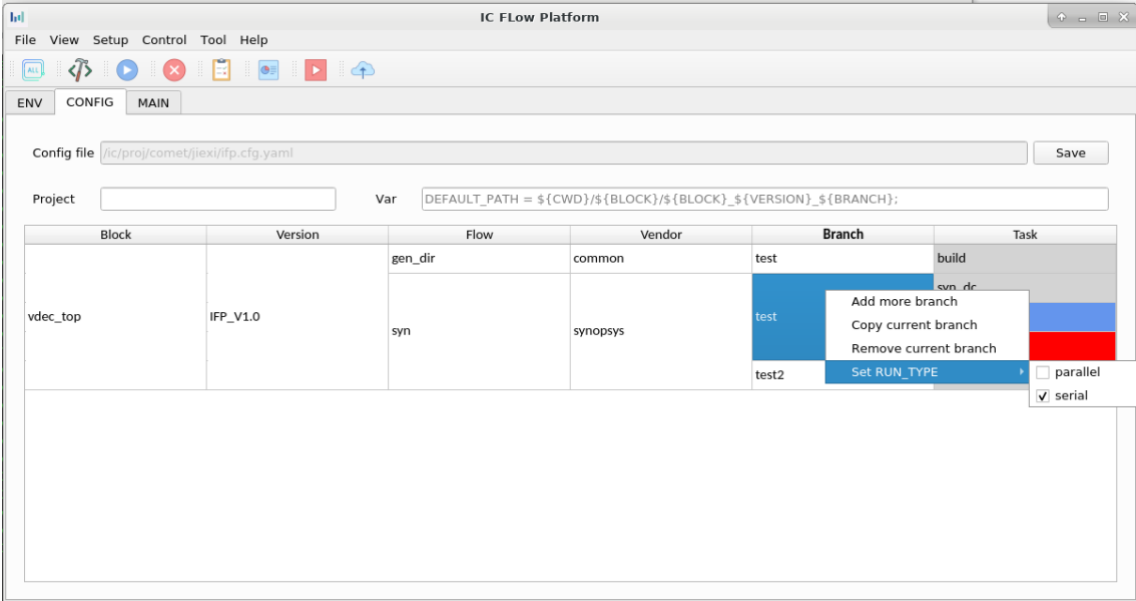
Copy current task: 复制当前 task 项目。

Remove current task: 删除当前 task 项目。

- **Branch 右键编辑**

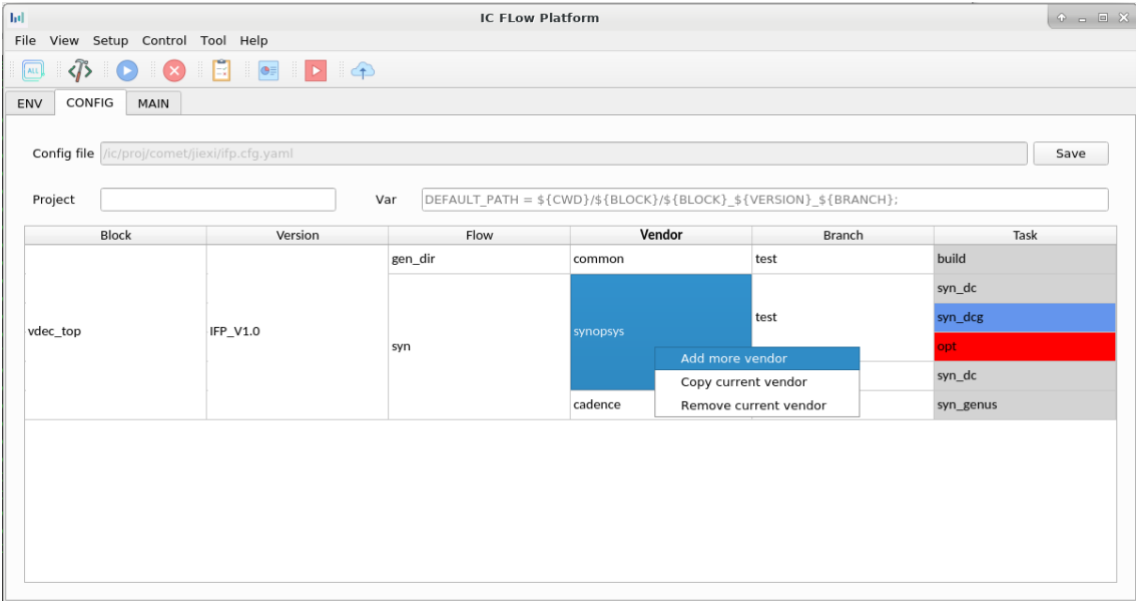
包含“Add more branch”，“Copy current branch”，“Remove current branch”和“Set RUN_TYPE”四部分。

前三部分同 task 项，“Set RUN_TYPE”是针对 branch 后对应的多个 task 设置运行方式，即并行或者串行。



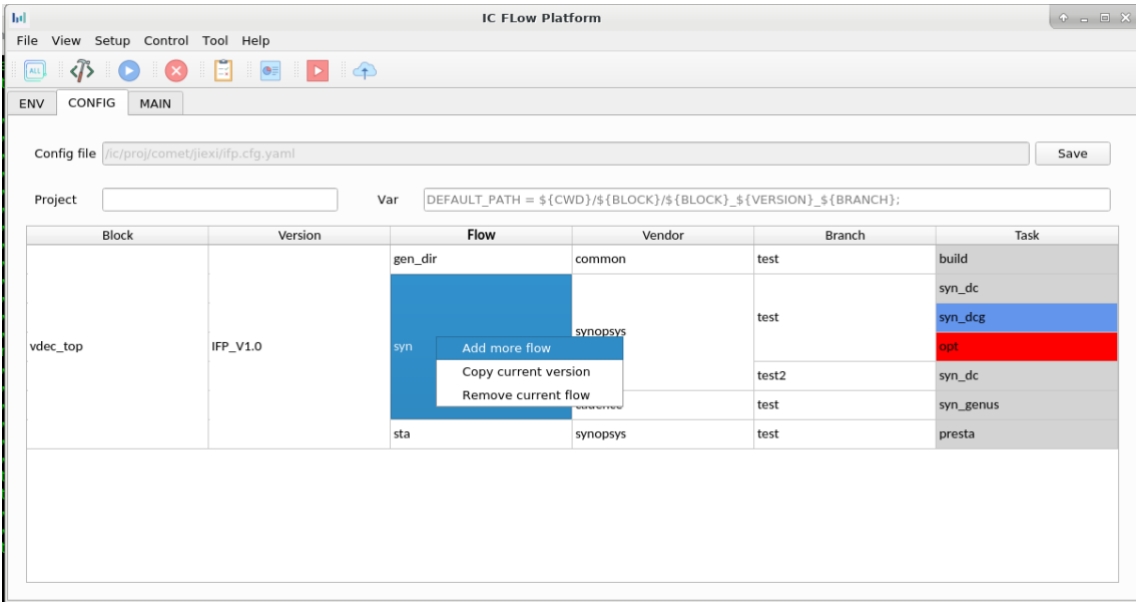
- **Vendor 右键编辑**

包含“Add more vendor”，“Copy current vendor”和“Remove current vendor”三部分。 如下为新增 vendor 为 cadence 的相关 task 信息：



- **Flow 右键编辑**

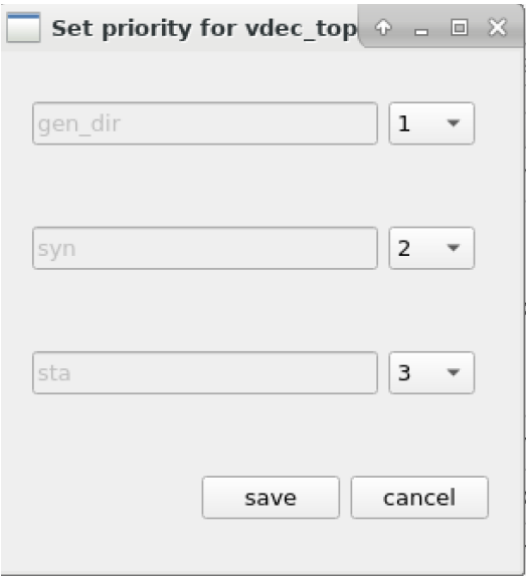
包含“Add more flow”，“Copy current flow”和“Remove current flow”三部分。如下新增 flow 为 sta 相关的 task 信息。



- **Version 右键编辑**

包含“Add more version”，“Copy current version”，“Remove current version”和“Set RUN_ORDER”四部分。

注：Set RUN_ORDER 定义该 block/version 下不同 flow 的运行顺序。



- **Block 右键编辑**

包含“Add more block”，“Copy current block”和“Remove current block”三部分。
如果新增 block 需要运行的任务与上个 block 一致，则直接点击“copy current block”，
然后只修改 block 名称即可，如下。

