

IFP 用户手册

Product Name : IFP (ic flow platform)

Product Version : V1.0

Release Date : 2023.2.14

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

[@节喜](mailto:jiexi@bytedance.com) (jiexi@bytedance.com)

[@景富义](mailto:jingfuyi@bytedance.com) (jingfuyi@bytedance.com)

目录

前言.....	4
一、简介.....	4
1.1 IC 流程演化.....	4
1.2 IC 流程平台的主要作用.....	5
1.3 IFP 开发思路.....	5
1.4 IFP 基本概念和控制逻辑.....	7
1.4.1 任务拆解 (从 Block 到 Task)	7
1.4.2 单个任务的行为 (从 Task 到 Action)	8
1.4.3 单个行为的属性 (从 Action 到 Attribute)	8
1.4.4 IFP 控制逻辑	9
二、环境依赖.....	10
2.1 操作系统依赖.....	10
2.2 PYTHON 版本依赖.....	10
2.3 集群管理系统.....	10
2.4 共享存储.....	10
三、工具安装及配置.....	11
3.1 工具下载.....	11
3.2 工具安装.....	12
3.3 工具配置	14
3.3.1 config/config.py	14
3.3.2 config/default.yaml	14
3.3.3 config/env.*	17
四、工具使用.....	19
4.1 工具载入.....	19
4.2 帮助信息	19
4.3 图形界面介绍	20
4.3.1 菜单栏	21
4.3.2 功能区	25
4.3.3 工作区	26
4.4 配置文件	27
4.4.1 编辑用户配置	27
4.4.2 编辑 Task 行为和属性配置	30
4.4.3 降低配置工作量	31
4.5 IFP 运行	32
4.6 其它功能介绍	34
4.6.1 菜单栏功能介绍	34
4.6.2 主界面功能介绍	43
4.6.3 辅助工具	47
五、范例.....	48
5.1 用户环境准备	48
5.1.1 EDA 配置文件	48
5.1.2 ifp 用户配置文件	48
5.2 操作范例	49
5.2.1 Build	50
5.2.2 Run	51

5.2.3 Kill	52
5.2.4 Check	52
5.2.5 Summary	53
5.2.6 Release	54
附录	55
附一、变更历史	55
附二、IFP 的配置文件	55

前言

IFP 是 ByteDance 开源的芯片设计流程平台，全称为 ic flow platform，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

数字集成电路设计流程演进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环境依赖，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，因此对流程运行规范和行为一致性提出了更高的要求。

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 流程运行规范通过平台工具的方式固化，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

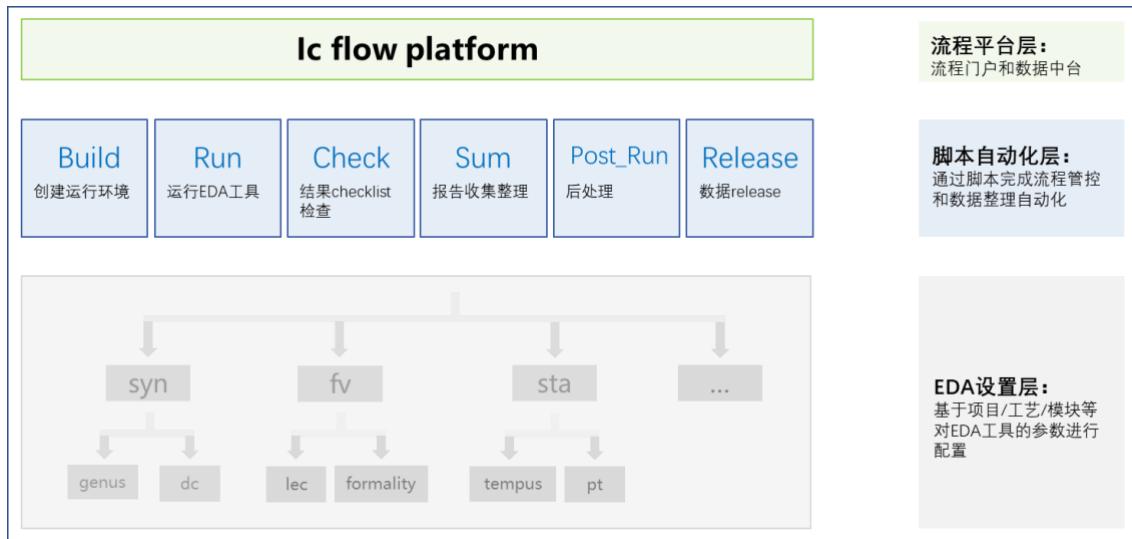
GUI 界面的流程平台简单易操作，可以大幅降低新人的流程运行门槛。同时“脚本自动化层”和“EDA 设置层”通过松耦合的方式嵌入，也不影响用户了解和配置单步自动化的实现方式以及具体的 EDA 设置项。

- 提升多模块运行效率

流程平台内嵌基于 multi-thread 或者 LSF/openlava 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 流程组织架构分为三个逻辑层次。



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 (cpu/gpu/ai/...), 项目所使用的工艺节点 (16/7/5/...), 自己所选用的 EDA 工具 (dc_shell/pt_shell/innovus/...), 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。

总地来说:

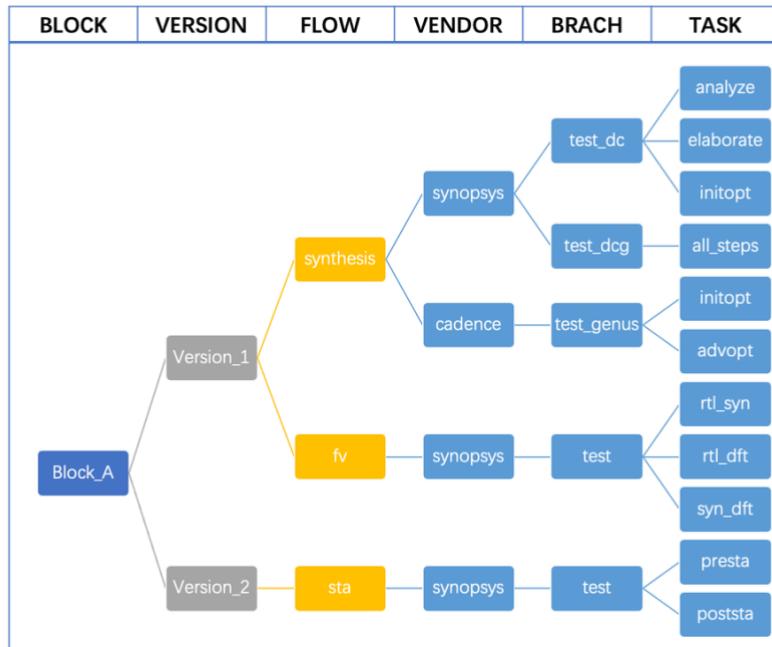
- IFP 主要涵盖流程平台层, 用来调度和管理用户任务。
- 自动化脚本仍然由每个 flow 的管理员自行维护和管理, 并非 IFP 内嵌的一部分。
- EDA 设置由每个 flow 的管理员和相关用户自行维护和管理, 并非 IFP 内嵌的一部分。
- 每个流程既可以采用流程平台运行, 也可以采用脚本运行, 只是流程规范和功能

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，下面用 mid-end 流程做例子。



- **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，大多按照 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 regression、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 等脚本的选择）
- **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（从 Task 到 Action）

每个 IC 设计流程的最小任务（task），都可以通过几个逻辑行为来实现，我们把每个逻辑行为称为一个 Action。

每一个 task 的实现步骤，都可以抽象为如下 6 种 Action。



每一个 Task 可能只用到其中的一种或者几种 Action，也可能用到全部 Action。

1.4.3 单个行为的属性（从 Action 到 Attribute）

对于每个 Action 而言，需要设定一些基本属性（Attribute）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。

也就是说，我们要完成一个任务（比如 synthesis 的 initopt），都可以通过指定的几个步骤完成（创建环境、运行工具、结果检查...），每个步骤的行为都是“**到指定 PATH 下面，执行执行 COMMAND, COMMAND 的运行方式为 RUN_METHOD**”。

针对 CHECK 和 SUMMARY 这两个 action，因为会生成报告，还需要查看报告，所以还有如下两个额外项目需要配置。

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。

这样，我们就把最小任务（task）的完成，拆解为几个行为（Action），又通过定义每个行为的属性来最终实现任务。

1.4.4 IFP 控制逻辑

IFP 本质上是一个带图形界面的逻辑控制机，输入复杂的用户需求，吐出统一的流程数据。



用户输入任务需求，任务需求包括三方面：

- Block/Version/Flow/Vendor/Branch/Task 信息，在用户配置文件（user config file）中定义。
- 每个任务（Task）的行为（Action）信息，可以在用户配置文件（user config file）中实现，也可以让流程管理员在默认配置文件（default config file）中预先定义。
- 每个行为（Action）的属性（Attribute）信息，可以在用户配置文件（user config file）中实现，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初始信息（用户配置文件）和操作指令（GUI 界面的点击操作），并行或者串行运行用户指定的任务，并最终汇总输出结果。

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 工具下载

IFP 的 github 路径位于 https://github.com/bytedance/ic_flow_platform。

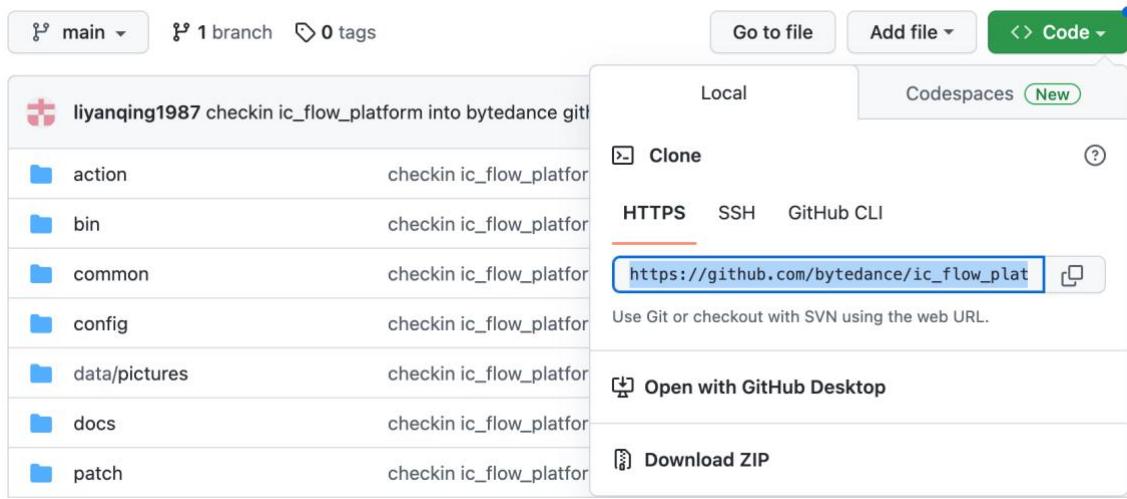
The screenshot shows the GitHub repository page for `bytedance/ic_flow_platform`. The repository is public and has 1 branch and 0 tags. The commit history shows a single commit from `liyanqing1987` at `b1735a1`, 47 minutes ago, which checks in the IC flow platform into the bytedance GitHub. The repository details include a description: "IFP (ic flow platform) is an integrated circuit design flow platform, mainly used for IC process specification management and data flow control." It also lists the README, GPL-2.0 license, 0 stars, 1 watching, and 0 forks. The releases section shows no releases published, and the packages section shows no packages published. The contributors section lists 2 contributors.

可以采用“`git clone https://github.com/bytedance/ic_flow_platform.git`”的方式拉取源代码。

Bash

```
bytedance@C02FT5LHMD6R Downloads % git clone  
https://github.com/bytedance/ic_flow_platform.git  
Cloning into 'ic_flow_platform'...  
remote: Enumerating objects: 111, done.  
remote: Counting objects: 100% (111/111), done.  
remote: Compressing objects: 100% (101/101), done.  
remote: Total 111 (delta 1), reused 108 (delta 1), pack-reused 0  
Receiving objects: 100% (111/111), 3.80 MiB | 1.17 MiB/s, done.  
Resolving deltas: 100% (1/1), done.
```

也可以在 IFP 的 github 页面上, Code -> Download ZIP 的方式拉取代码包。



3.2 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@n212-206-194 tools]$ cd ic_flow_platform/
[liyanqing.1987@n212-206-194 ic_flow_platform]$ ls
action  bin  common  config  data  docs  install.py  LICENSE.txt
Notice.txt  patch  README  requirements.txt  third_part  tools
```

首先确认当前 python 版本正确。

Bash

```
[liyanqing.1987@n212-206-194 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@ic-admin1 ic_flow_platform]# pip3 install -r requirements.txt
Looking in indexes: https://bytedpypi/byted.org/simple/
Collecting matplotlib==3.6.3
```

```
  Downloading
https://bytedpypi/byted.org/packages/pypi/matplotlib/matplotlib-
3.6.3-cp38-cp38-manylinux_2_12_x86_64.manylinux2010_x86_64.whl (9.4
MB)
----- 9.4/9.4 MB 174.4 MB/s
eta 0:00:00
...
Installing collected packages: matplotlib
  Attempting uninstall: matplotlib
    Found existing installation: matplotlib 3.3.4
    Uninstalling matplotlib-3.3.4:
      Successfully uninstalled matplotlib-3.3.4
Successfully installed matplotlib-3.6.3
```

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

```
Bash
[liyanqing.1987@n212-206-194 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!
```

3.3 工具配置

3.3.1 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 = ""
```

示例设置为：

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

# 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, 一般用于给 IFP 用户发送汇总结果, 也可以不设。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。

3.3.2 config/default.yaml

default.yaml 用于定义指定 Task 的默认 Action/Attribute 设置。

本来, 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，这样，用户配置文件 ifp.cfg.yaml 中缺失的信息，会自动来 config/default.yaml 中查找。

config/default.yaml 的原始设置为：

```
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
#
#
## Supportedd action attribute ##
# PATH/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}/action/check/syn/synopsys/syn_synopsys.syn_dc.py
-b ${BLOCK}
#           VIEWER:
${IFP_INSTALL_PATH}/action/check/scripts/view_checklist_report.py -i
#           REPORT_FILE: file_check/file_check.rpt
#           SUMMARY:
#               PATH: ${DEFAULT_PATH}/dc
#               COMMAND:
${IFP_INSTALL_PATH}/action/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

```

下面是一个示例：

```

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

TASK:
    gen_dir:common:build:
        BUILD:
            PATH: $CWD
            COMMAND: /ic/flow/scripts/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  
  ...
```

3.3.3 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  
  
##### Default EDA tool settings #####  
# Set default TESSENT 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/data/usr/liyanqing.1987/tools/ic_flow_platform/tools/lsfMonitor/monitor/bin:$PATH
```

```
# Set default soffice path.
```

示例设置为：

Bash

```
##### Default EDA tool settings #####
# Set default TESSENT 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/data/usr/liyanqing.1987/tools/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

IFP 的主程序是 ifp，位于 IFP 安装路径下的 bin/ifp，安装后可以直接引用。

如果配置了 modules，也可以通过 module load 的方式引用。

配置 ifp 的 alias 也是一种比较简便的方式。

Bash

```
[liyanqing.1987@n212-206-194 ic_flow_platform]$ alias  
ifp=/ic/data/usr/liyanqing.1987/tools/ic_flow_platform/bin/ifp  
[liyanqing.1987@n212-206-194 ic_flow_platform]$ which ifp  
alias  
ifp='/ic/data/usr/liyanqing.1987/tools/ic_flow_platform/bin/ifp'  
/ic/data/usr/liyanqing.1987/tools/ic_flow_platform/bin/ifp
```

4.2 帮助信息

ifp 的帮助信息如下。

Bash

```
[liyanqing.1987@n212-206-194 ic_flow_platform]$ 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。

4.3 图形界面介绍

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@n212-206-194 ifp_test]$ ifp  
>>> Generating empty configure file  
"/ic/data/usr/liyanqing.1987/ifp_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。

菜单栏囊括了完整的功能和配置选项。

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，分别用来做环境信息查看、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。

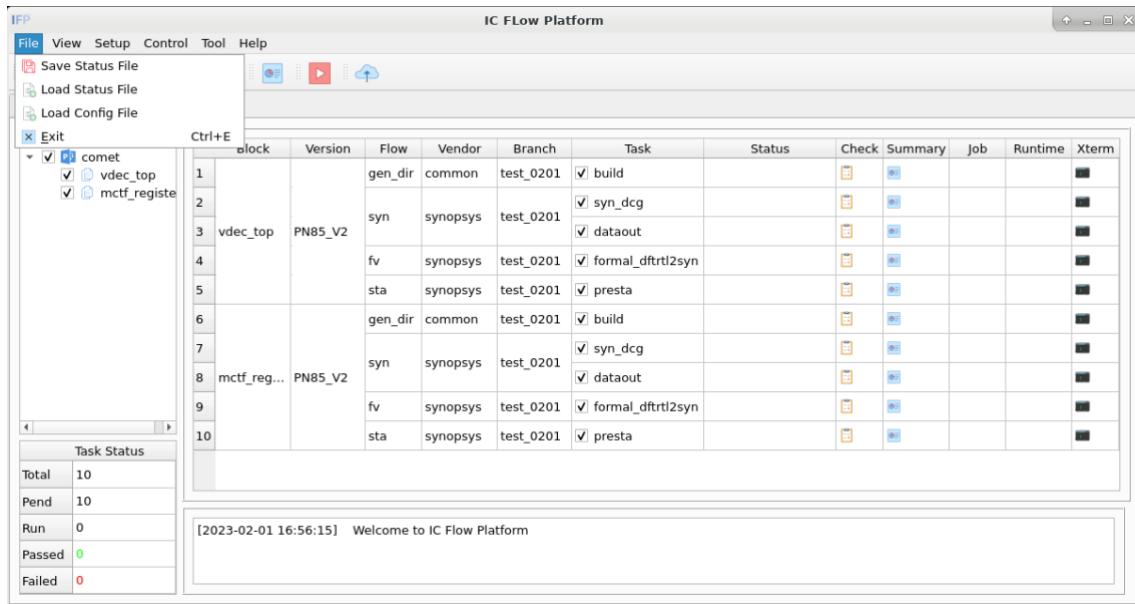
菜单 File

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。

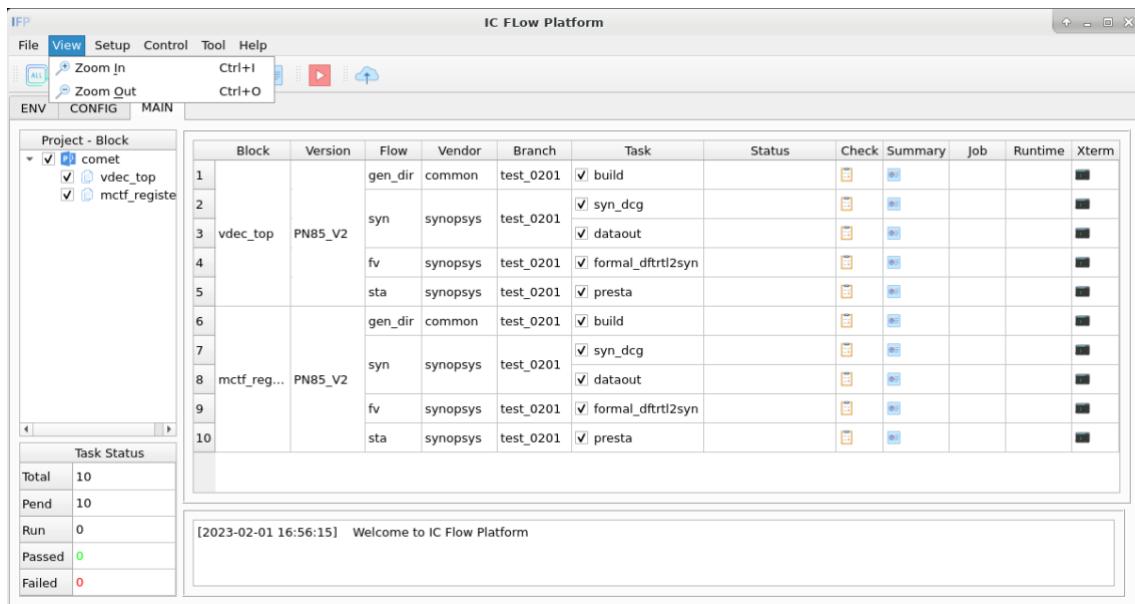
Exit: 退出。



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

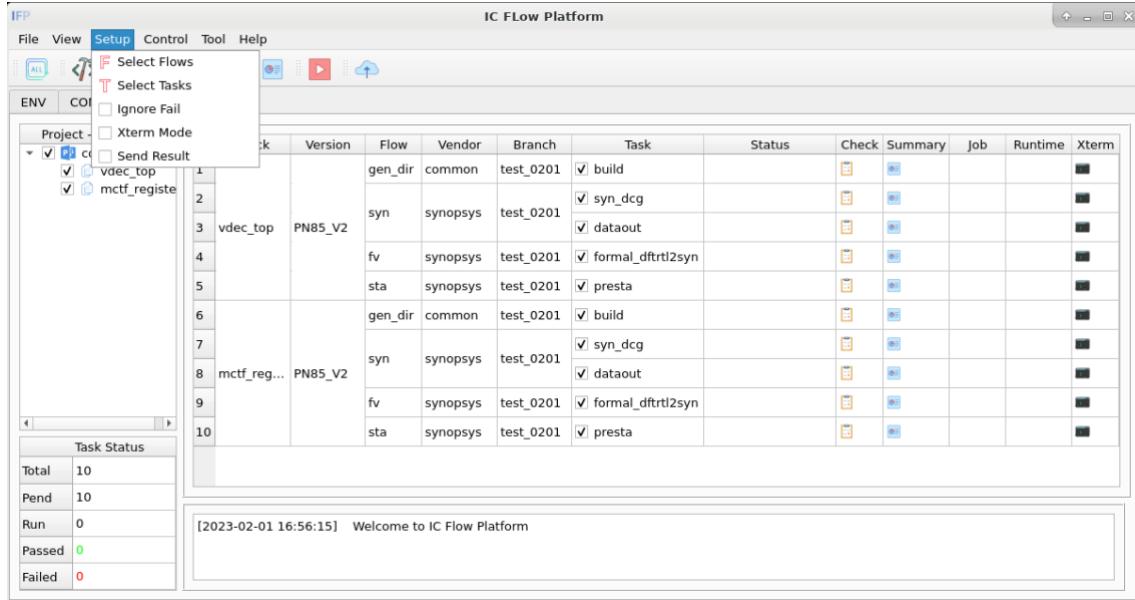
Select Flows: 批量选中主页中的 Flow 项。

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。

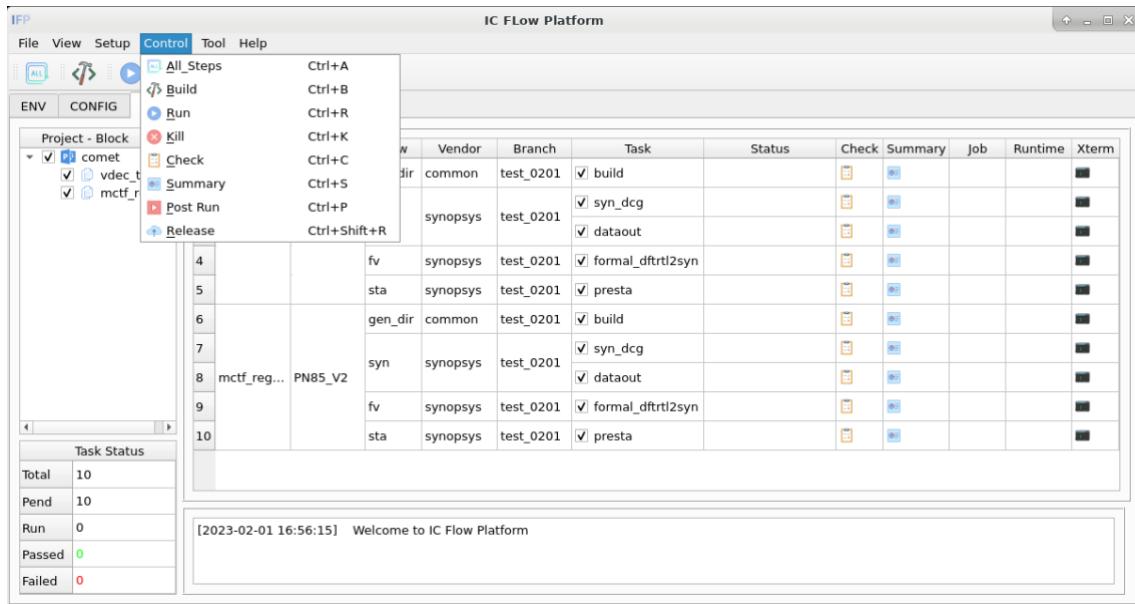
Kill: 如果 job 是 bsub 出去的， kill 所选中的任务。

Check: 执行 Check 行为。

Summary: 执行 Summary 行为。

Post Run: 执行 Post_run 行为。

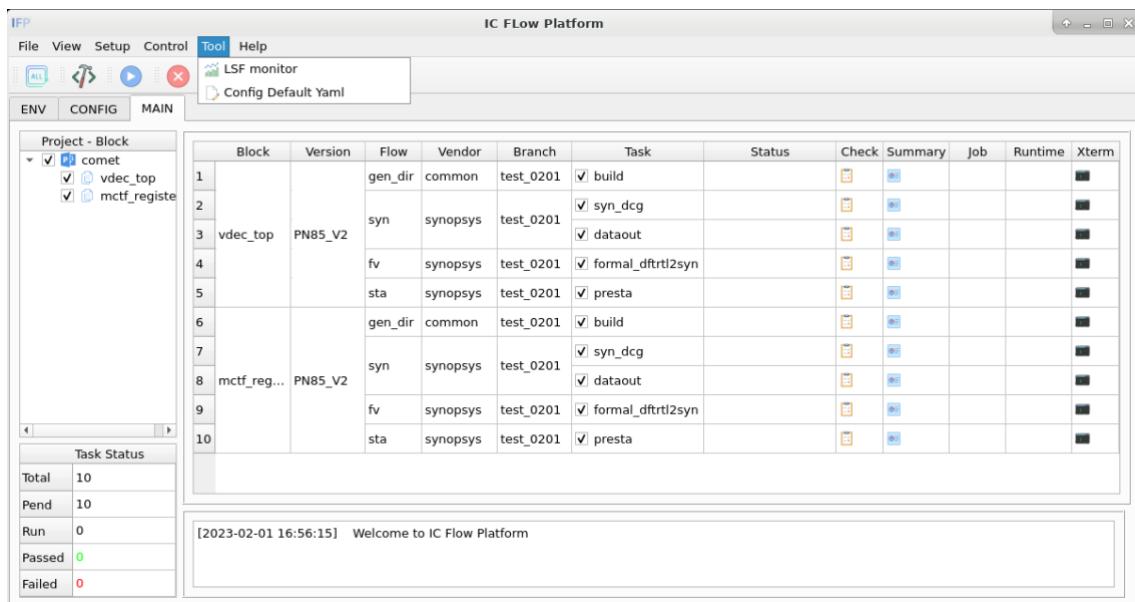
Release: 执行 Release 行为。



菜单 Tool

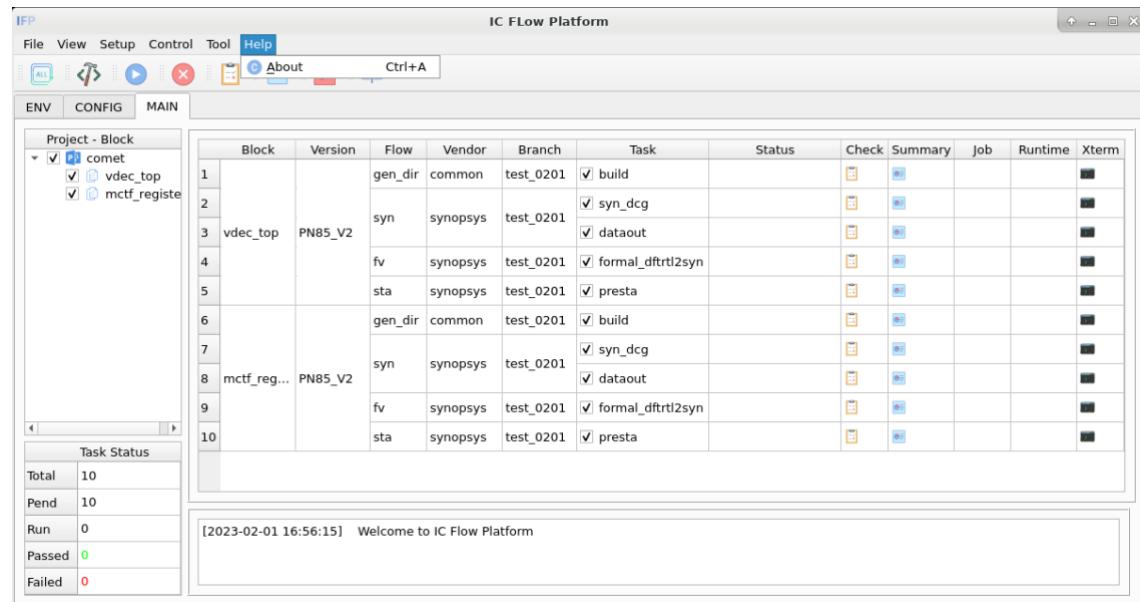
LSF monitor: 执行开源工具 lsfMonitor，用于获取 LSF 的基本信息。

Config Default Yaml: 执行 config_default_yaml 工具，用于编辑默认 Task 属性。
(只有 IFP 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。



从左到右依次是：

Run All Steps : 运行后面所有步骤

Build : 环境运行创建

Run : 运行 EDA 工具

Kill : 终止选定的任务

Check : checklist 检查

Summary : summary 报告收集

Post Run : 结果后处理

Release : 数据 release

4.3.3 工作区

工作区是用户主要的操作区域和信息获取区域，分为 ENV/CONFIG/MAIN 三个子页面。

ENV 页

查看当前的环境变量信息，可以用户环境设置确认。

Env_Variable	Value
MANPATH	/ic/software/tools/lsf/10.1/man:
SSH_AGENT_PID	125534
HOSTNAME	n212-206-194
_LMFILES_modshare	/ic/software/modules/config/EDA/cadence/genus/21.14-s082_1:1:/ic/software/modules/config/tools/cad:1:/ic/software/modules/config/...
MODULEPATH_modshare	/ic/software/modules/modules-4.7.1/modulefiles:1
GPG_AGENT_INFO	/home/liyanqing.1987/.gnupg/S.gpg-agent:125536:1
MODULES_CMD	/ic/software/modules/modules-4.7.1/libexec/modulecmd.tcl
SHELL	/bin/bash
VTE_VERSION	5204
TERM	xterm-256color
HISTSIZE	1000
PYTHONUNBUFFERED	1
WINDOWID	48240248
QTDIR	/usr/lib64/qt-3.3
IICF_SREVDIR	/ic/software/tools/lef/10.1/linux3.10/libc2.17-v26.64/etc

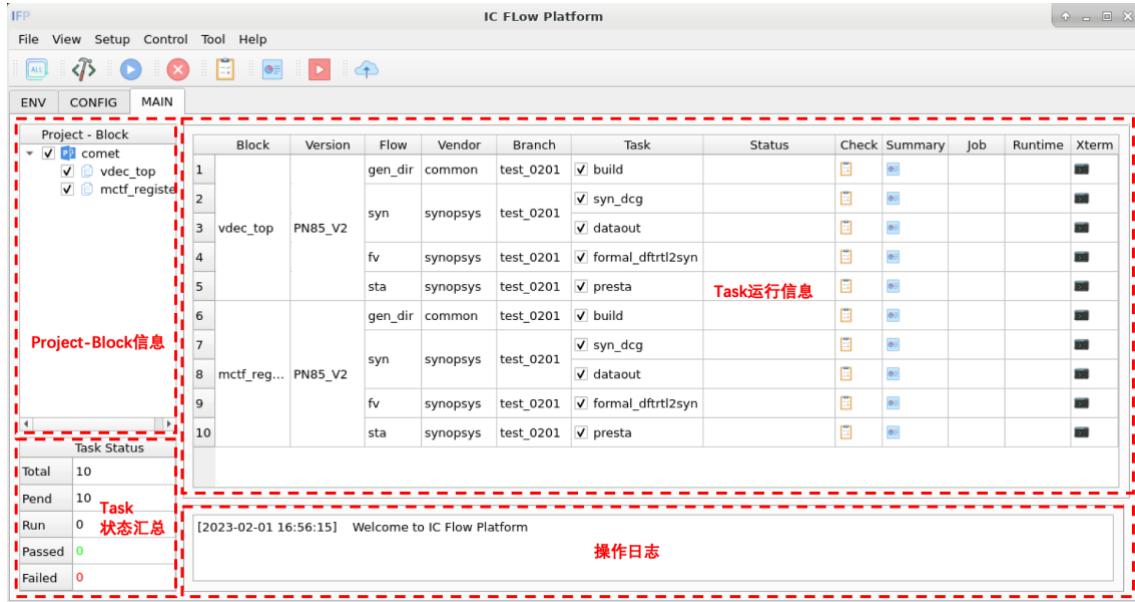
CONFIG 页

配置和查看用户配置文件，详情请参照附二。

Project		Var	Config file: /ic/data/usr/liyanqing.1987/ifp_test/ifp.cfg.yaml						
vdec_top	comet	DEFAULT_PATH = \${CWD}/\${BLOCK}/\${BLOCK}_\${VERSION}_\${BRANCH};	Block	Version	Flow	Vendor	Branch		
	gen_dir	common			test_0201				
	syn	synopsys			test_0201				
	fv	synopsys			test_0201				
mctf_register	comet	DEFAULT_PATH = \${CWD}/\${BLOCK}/\${BLOCK}_\${VERSION}_\${BRANCH};			sta	synopsys	test_0201		
					gen_dir	common	build		
					syn	synopsys	syn_dcg		
					fv	synopsys	dataout		

MAIN 页

Task 的运行信息展示区域。



4.4 配置文件

用户通过用户配置文件 (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, 用什么 RUN_METHOD, 执行什么 COMMAND。)

PS. 用户配置文件的设置略微复杂，手工配置量大，所以请仔细阅读本节，有助于用户后续将此环节自动化。

4.4.1 编辑用户配置

默认的用户配置文件为当前运行目录下的 ifp.cfg.yaml，其数据格式如下所示。

YAML

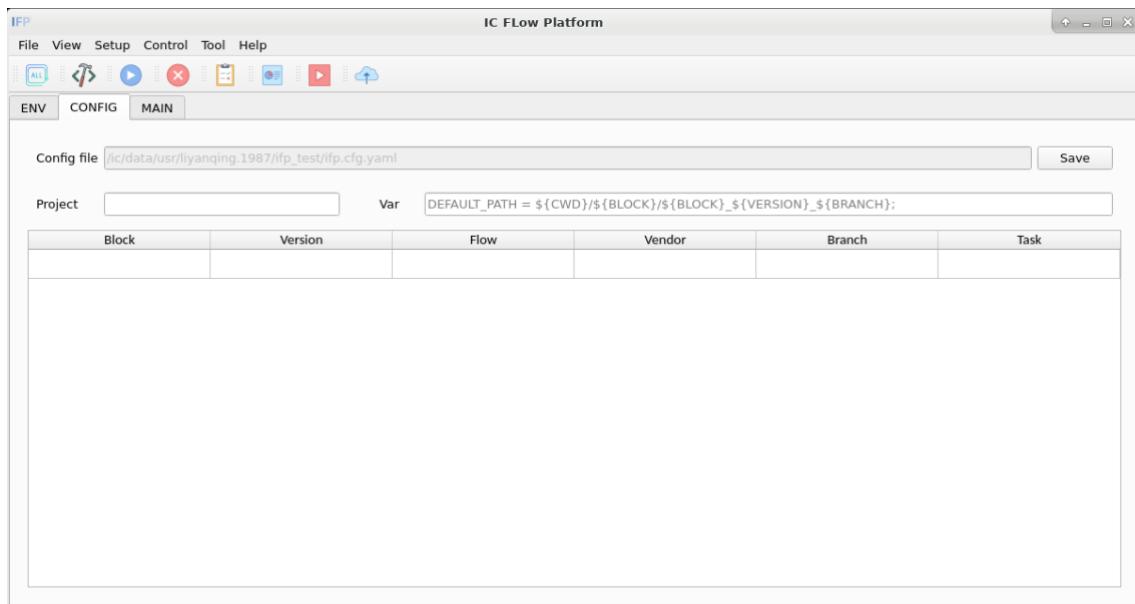
```
PROJECT: comet
VAR: {}
BLOCK:
```

```

<BLOCK>:
  <VERSION>(RUN_ORDER=<FLOW1>, <FLOW2>|<FLOW3>):
    <FLOW>:
      <VENDOR>:
        <TASK>(RUN_TYPE=serial):
          <FUNCTION>: {}
          <FUNCTION>: {}

```

可以图形化方式编辑用户配置文件 ifp.cfg.yaml, ifp 的 CONFIG 页即为用户配置文件编辑器。

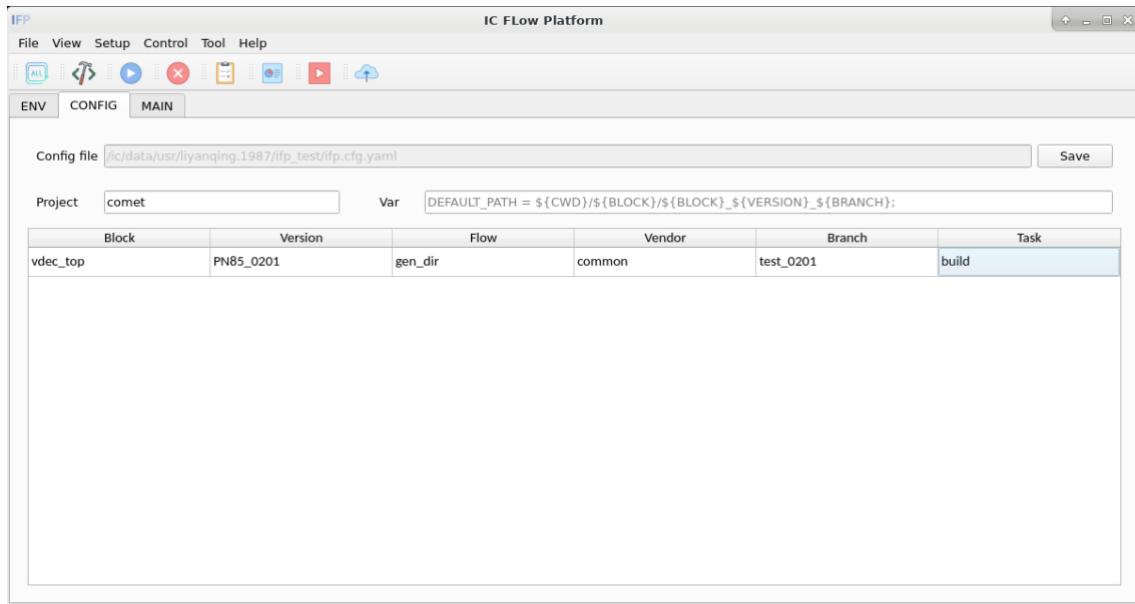


图形化编辑方式中，如果 ifp 启动的时候没有特别指定，用户配置文件默认为当前路径下的 ifp.cfg.yaml 文件，Save 的时候可以重新指定其路径和文件名。

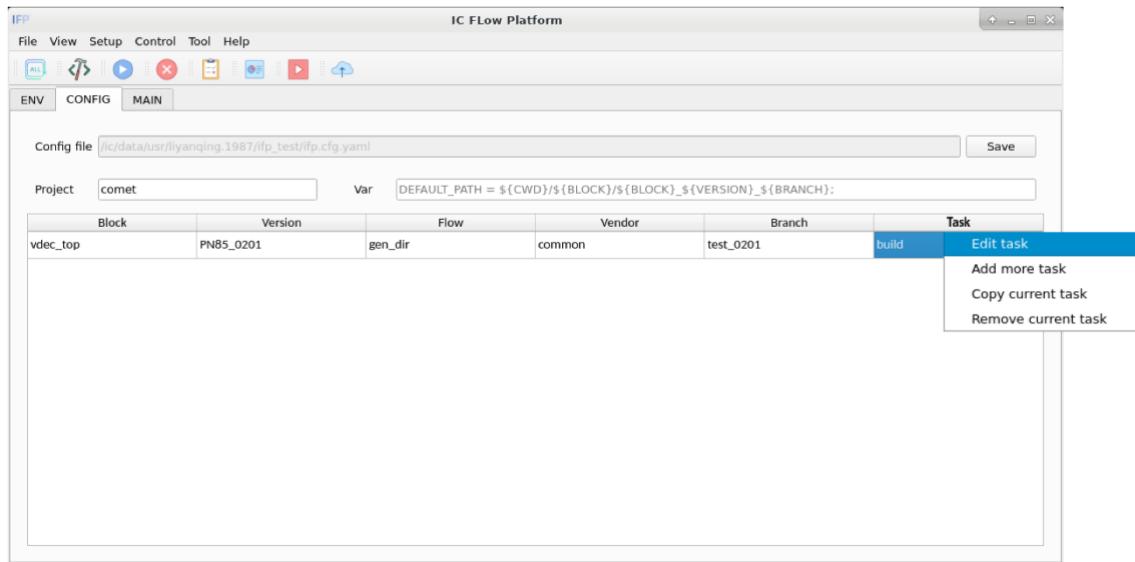
"Project"是项目信息。

"Var"用于指定环境变量信息。

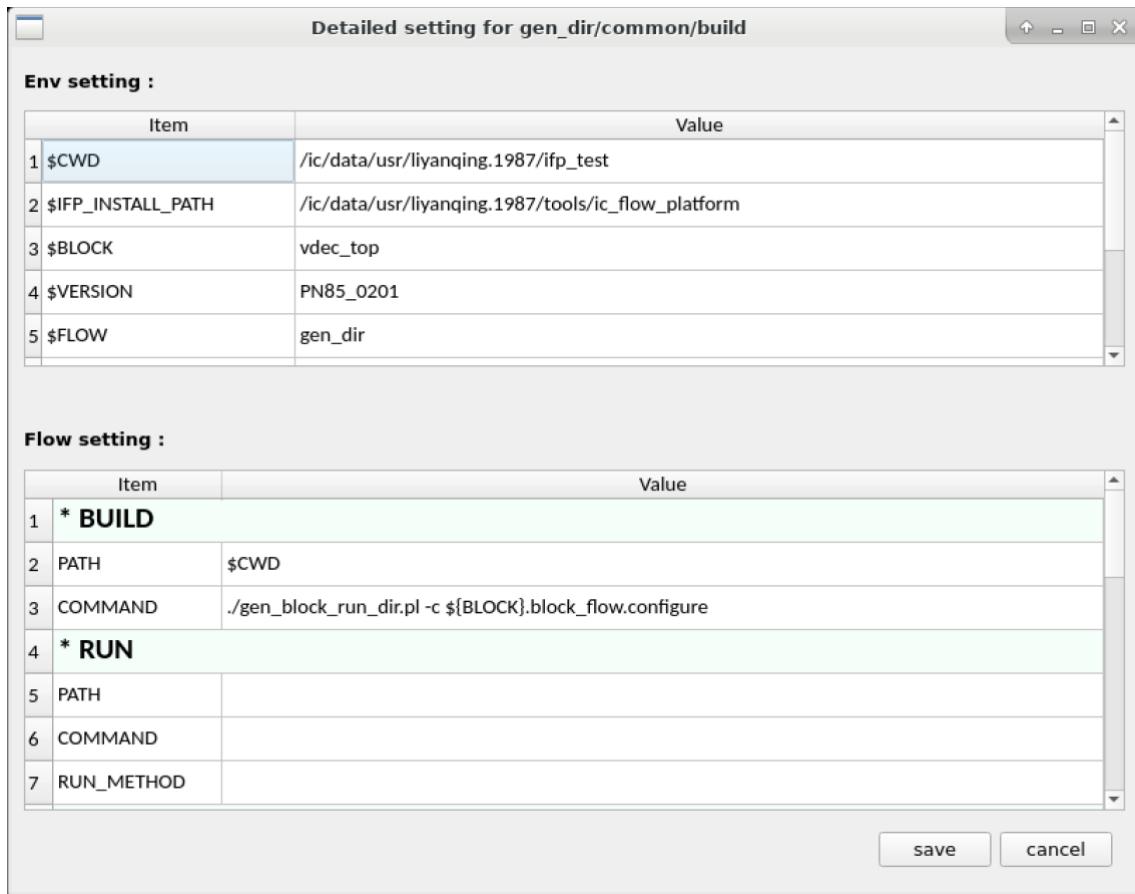
CONFIG 页面主界面可以直接编辑 Block/Version/Flow/Vendor/Branch/Task 信息。
(单元格均可编辑)



Task 具体的行为 (Build/Run/Check/Summary/Post_run/Release)，以及具体行为的属性 (PATH/COMMAND/RUN_METHOD/VIEWER/REPORT_FILE)，则可以通过 Task 单元格的右键 Edit task 功能编辑。(每个单元格均可编辑)



调出的 Task 属性编辑界面如下图所示。

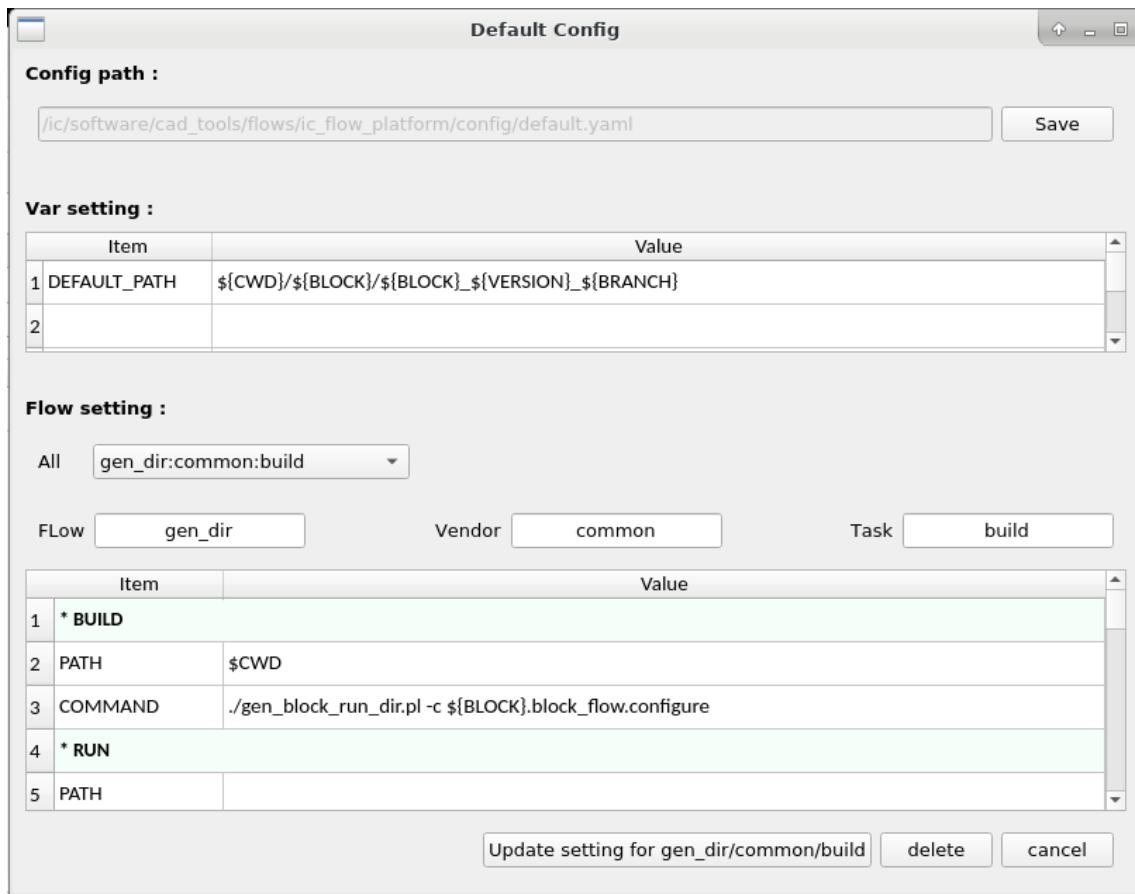


上图我们可以看到，每个 Task 都有很多行为及行为属性需要编辑（Flow setting），如果当前 Task 某些行为及属性不会用到，则可以不用编辑。

4.4.2 编辑 Task 行为和属性配置

每个 Task 行为及属性信息的编辑，会带来较大的工作量。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 安装目录下的 config/default.yaml），让 ifp 在读取用户配置的时候直接引用这些默认的 Task 属性，这样可以极大减轻配置工作量。

可以通过 ifp 自带的 config default yaml 工具编辑（菜单栏 Tool -> Config Default Yaml），工具样式如下。



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。

4.4.3 降低配置工作量

用户配置文件的编辑工作比较麻烦，有几个途径可以降低配置工作量。（我们实际使用过程中也是这样操作的）

- 由于每个流程的 Task 的行为和属性相对固定，可以配置到默认配置文件 config/default.yaml 中。
- 同一流程的用户运行模式往往一致，不同点在于运行模块（Block）有差异。

同一用户的运行模块往往也一致，每次运行的不同点在于 version 等信息。

因此，可以准备好一个配置文件模板，用脚本基于模板自动生成当次需要的用户配置文件。

如上最佳实践的实现如果存在疑问，可以联系 IFP 的 Contact 联系人咨询具体方法。

4.5 IFP 运行

IFP 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。

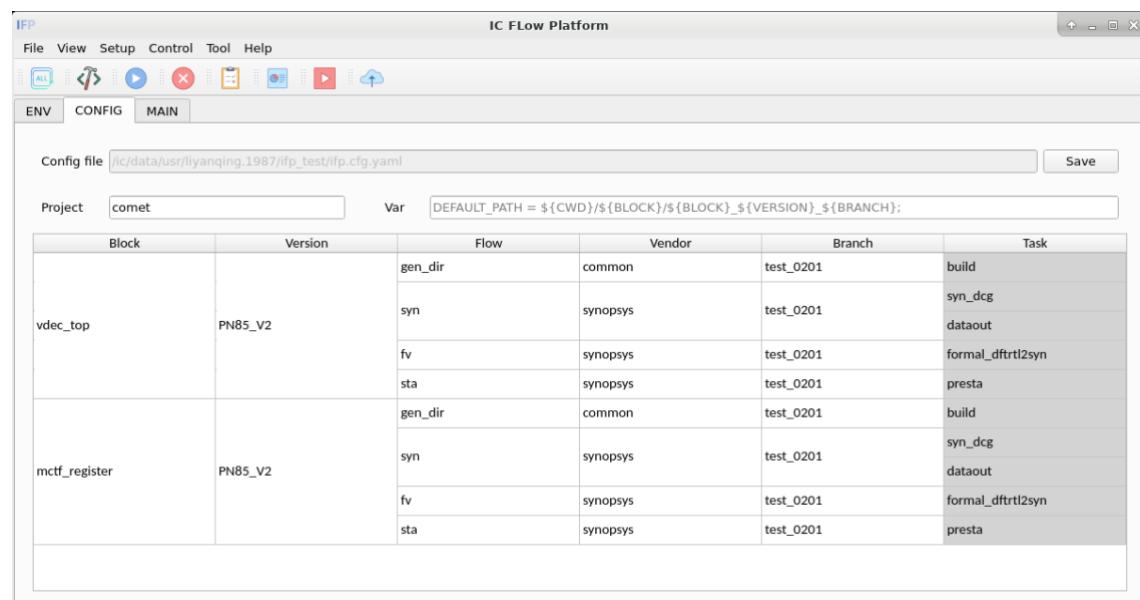
下面是一个 IFP 使用示例:

进入运行目录

```
Bash
[liyanqing.1987@n212-206-194 ~]$ cd
/ic/data/usr/liyanqing.1987/ifp_test/
[liyanqing.1987@n212-206-194 ifp_test]$ ifp
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to
'/tmp/runtime-liyanqing.1987'
```

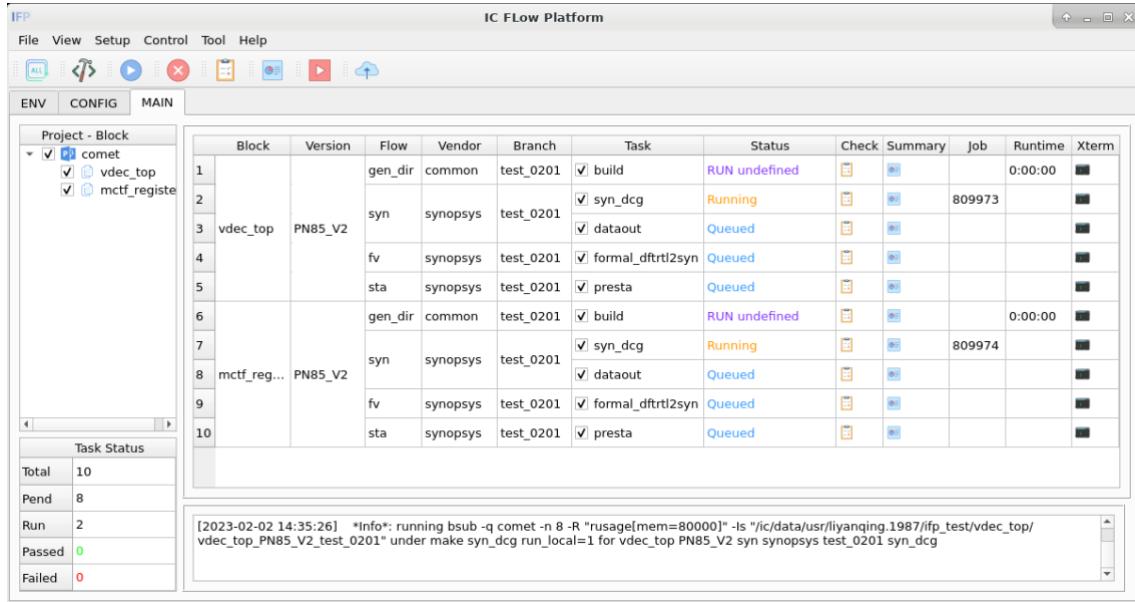
配置用户配置文件

通过 CONFIG 页编辑用户配置文件 (提前准备好亦可)。



执行指定行为

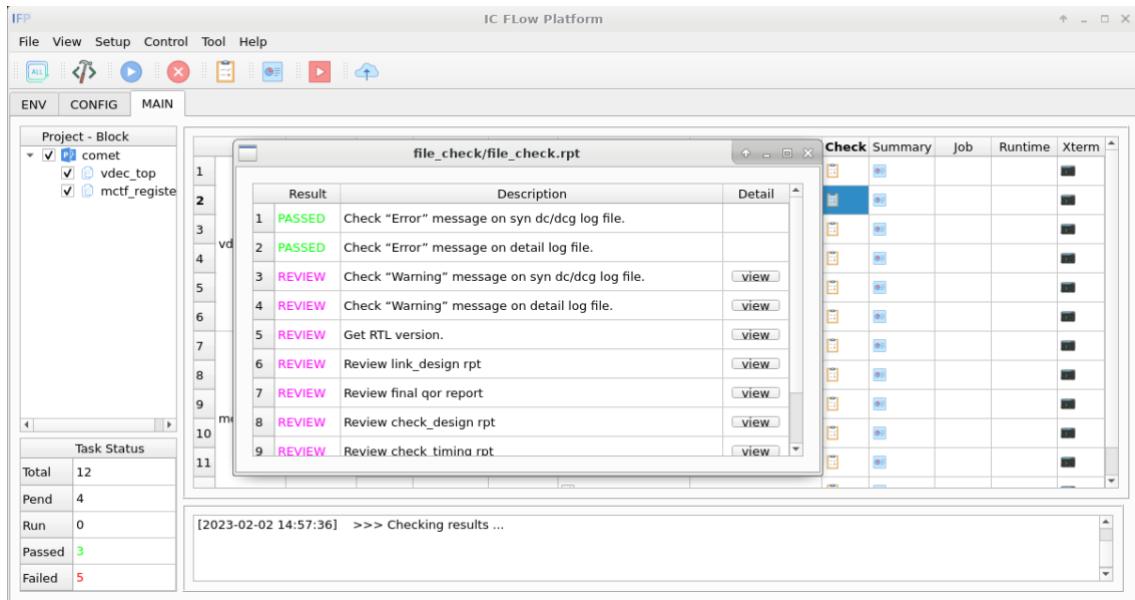
到 MAIN 界面，执行 BUILD、RUN 等步骤。



结果检查和数据核对

IFP 平台完成 check 和 summary 操作后，会生成 checklist 和 summary（一般是 QoR）汇报报告，可以点击对应的单元格查看。

下面是一个利用 IFP 自带的 checklist 检查机制生成 checklist report 的展示。



下面是一个 Excel 格式的 summary report 的展示。

The screenshot shows the IC FFlow Platform interface with several windows open:

- IFP** window: Shows a tree view under "Project - Block" with nodes like "comet", "vdec_top", and "mctf_register".
- LibreOffice Calc** window: An untitled spreadsheet titled "Untitled 1 - LibreOffice Calc". It has a header row with columns: Block, Version, Flow, Vendor, Branch, Task, Status, Check, Summary, Job, Runtime, Xterm. Below this, two rows of data are shown:

Block	Version	Flow	Vendor	Branch	Task	Status	Check	Summary	Job	Runtime	Xterm
1		gen_dir	common	test	<input checked="" type="checkbox"/> build	SUMMARY ...					
2					<input checked="" type="checkbox"/> syn_dcg	SUM PASS					
- Task Status** window: A table showing task statistics:

Total	12
Pend	4
Run	0
Passed	6
Failed	2

At the bottom of the Calc window, there is a message log:

```
[2023-02-02 14:59:06] >>> Summarizing results ...
[2023-02-02 14:59:06] >>> Summarizing Done
```

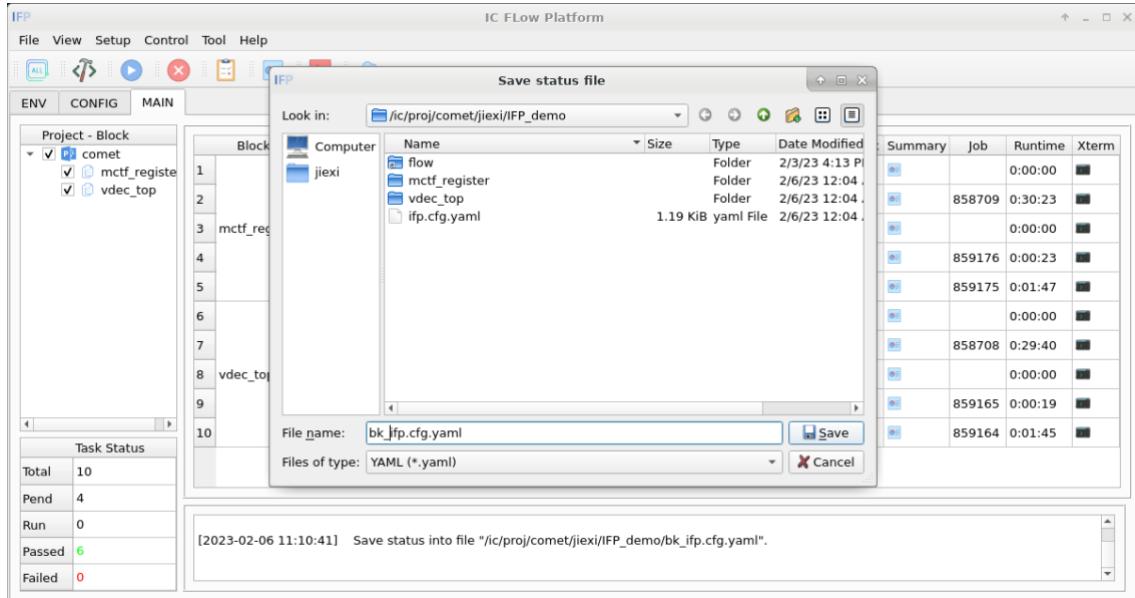
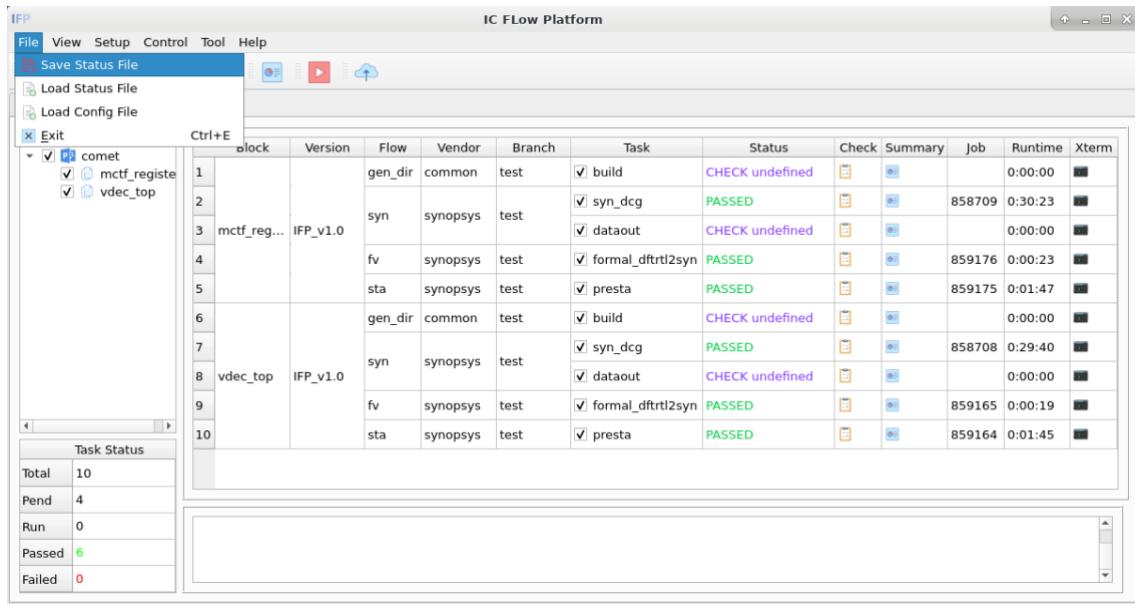
4.6 其它功能介绍

4.6.1 菜单栏功能介绍

- **Save Status File**

保存当前任务的 Status/Job/Runtime 等信息到指定的 bk_ifp.cfg.yaml 文件中，操作如下：

点击“File”->“Save Status File”，然后在 File_name 处命名为“bk_ifp.cfg.yaml”，
点击“Save”进行保存，Gui 界面下方会显示保存信息“Save status into file
"/ic/proj/comet/jiexi/IFP_demo/bk_ifp.cfg.yaml"。”



• Load Status File

载入之前保存的 status file(如 bk_ifp.cfg.yaml), 将 Status/Job/Runtime 等信息直接展开到当前主界面中。

点击“File”->“Load Status File”，弹出界面框后选择“bk_ifp.cfg.yaml”，然后点击“Open”，Gui 界面将 Status/Job/Runtime 等信息直接展开到当前主界面中。下方会显示加载信息“Load status with file ”/ic/proj/comet/jiexi/IFP_demo/bk_ifp.cfg.yaml”。”

IC FFlow Platform

File View Setup Control Tool Help

Load Status File

Load Config File

Exit

Block Version Flow Vendor Branch Task Status Check Summary Job Runtime Xterm

Block	Version	Flow	Vendor	Branch	Task	Status	Check	Summary	Job	Runtime	Xterm
1		gen_dir	common	test	✓ build	PASSED				0:00:00	
2		syn	synopsys	test	✓ syn_dcg	PASSED			858709	0:30:23	
3	mctf_reg...	IFP_v1.0			✓ dataout	CHECK undefined				0:00:00	
4		fv	synopsys	test	✓ formal_dftrtl2syn	PASSED			859176	0:00:23	
5		sta	synopsys	test	✓ presta	PASSED			859175	0:01:47	
6		gen_dir	common	test	✓ build	CHECK undefined				0:00:00	
7		syn	synopsys	test	✓ syn_dcg	PASSED			858708	0:29:40	
8	vdec_top	IFP_v1.0			✓ dataout	CHECK undefined				0:00:00	
9		fv	synopsys	test	✓ formal_dftrtl2syn	PASSED			859165	0:00:19	
10		sta	synopsys	test	✓ presta	PASSED			859164	0:01:45	

Task Status

Total	10
Pend	10
Run	0
Passed	0
Failed	0

[2023-02-06 11:12:01] Welcome to IC Flow Platform

IC FFlow Platform

File View Setup Control Tool Help

ENV CONFIG MAIN

Project - Block

- comet
 - ✓ mctf_register
 - ✓ vdec_top

Block Version Flow Vendor Branch Task Status Check Summary Job Runtime Xterm

Block	Version	Flow	Vendor	Branch	Task	Status	Check	Summary	Job	Runtime	Xterm
1		gen_dir	common	test	✓ build	PASSED				0:00:00	
2		syn	synopsys	test	✓ syn_dcg	PASSED			858709	0:30:23	
3	mctf_reg...	IFP_v1.0			✓ dataout	CHECK undefined				0:00:00	
4		fv	synopsys	test	✓ formal_dftrtl2syn	PASSED			859176	0:00:23	
5		sta	synopsys	test	✓ presta	PASSED			859175	0:01:47	
6		gen_dir	common	test	✓ build	CHECK undefined				0:00:00	
7		syn	synopsys	test	✓ syn_dcg	PASSED			858708	0:29:40	
8	vdec_top	IFP_v1.0			✓ dataout	CHECK undefined				0:00:00	
9		fv	synopsys	test	✓ formal_dftrtl2syn	PASSED			859165	0:00:19	
10		sta	synopsys	test	✓ presta	PASSED			859164	0:01:45	

Task Status

Total	10
Pend	10
Run	0
Passed	0
Failed	0

[2023-02-06 11:12:01] Welcome to IC Flow Platform

Load status file

Look in: /ic/proj/comet/jiexi/IFP_demo

Name	Size	Type	Date Modified
flow	1.75 KiB	yaml File	2/6/23 12:04
mctf_register	1.19 KiB	yaml File	2/6/23 12:04
vdec_top		Folder	2/6/23 12:04
bk_ifp.cfg.yaml	1.75 KiB	yaml File	2/6/23 11:11
ifp.cfg.yaml	1.19 KiB	yaml File	2/6/23 12:04

File name: bk_ifp.cfg.yaml

Files of type: YAML (*.yaml)

Summary Job Runtime Xterm

IC FFlow Platform

File View Setup Control Tool Help

ENV CONFIG MAIN

Project - Block

- comet
 - ✓ mctf_register
 - ✓ vdec_top

Block Version Flow Vendor Branch Task Status Check Summary Job Runtime Xterm

Block	Version	Flow	Vendor	Branch	Task	Status	Check	Summary	Job	Runtime	Xterm
1		gen_dir	common	test	✓ build	PASSED				0:00:00	
2		syn	synopsys	test	✓ syn_dcg	PASSED			858709	0:30:23	
3	mctf_reg...	IFP_v1.0			✓ dataout	CHECK undefined				0:00:00	
4		fv	synopsys	test	✓ formal_dftrtl2syn	PASSED			859176	0:00:23	
5		sta	synopsys	test	✓ presta	PASSED			859175	0:01:47	
6		gen_dir	common	test	✓ build	CHECK undefined				0:00:00	
7		syn	synopsys	test	✓ syn_dcg	PASSED			858708	0:29:40	
8	vdec_top	IFP_v1.0			✓ dataout	CHECK undefined				0:00:00	
9		fv	synopsys	test	✓ formal_dftrtl2syn	PASSED			859165	0:00:19	
10		sta	synopsys	test	✓ presta	PASSED			859164	0:01:45	

Task Status

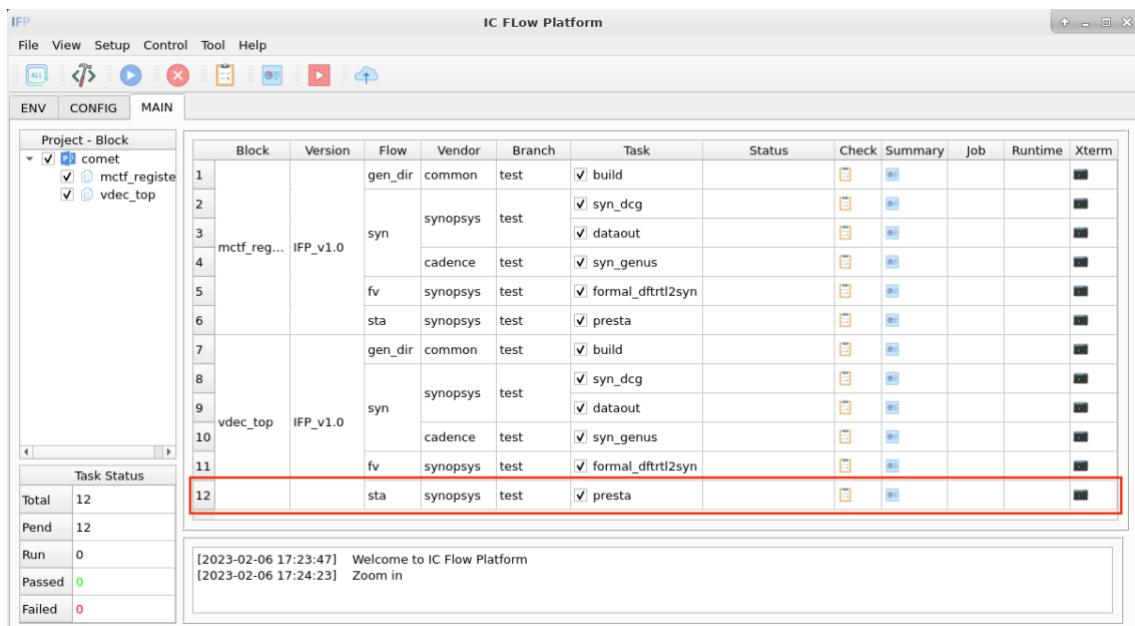
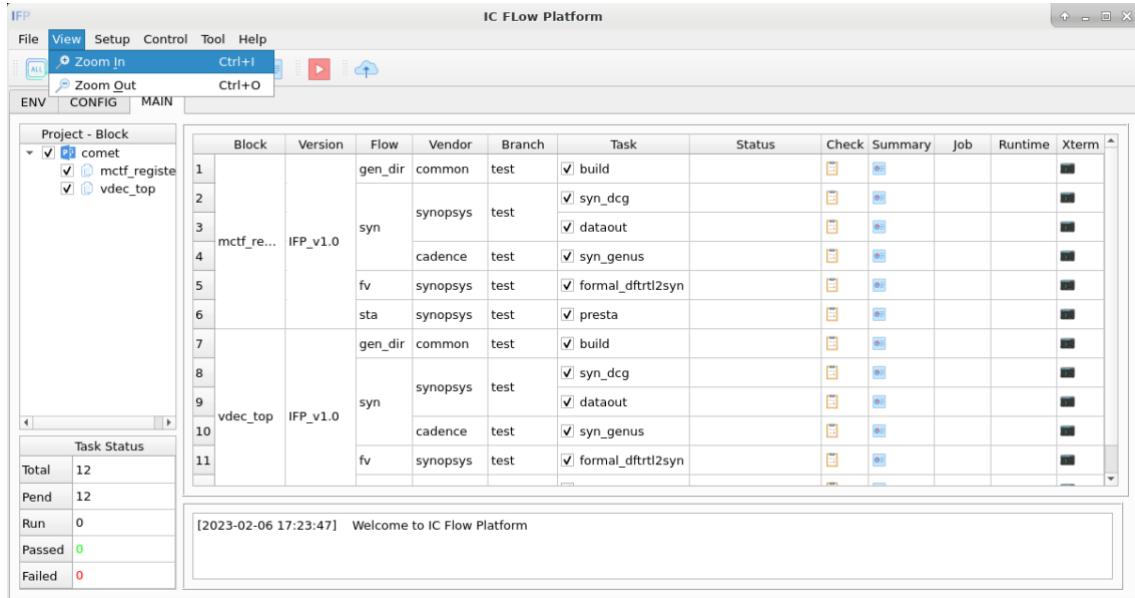
Total	10
Pend	4
Run	0
Passed	6
Failed	0

[2023-02-06 11:12:01] Welcome to IC Flow Platform

[2023-02-06 11:13:19] Load status with file "/ic/proj/comet/jiexi/IFP_demo/bk_ifp.cfg.yaml".

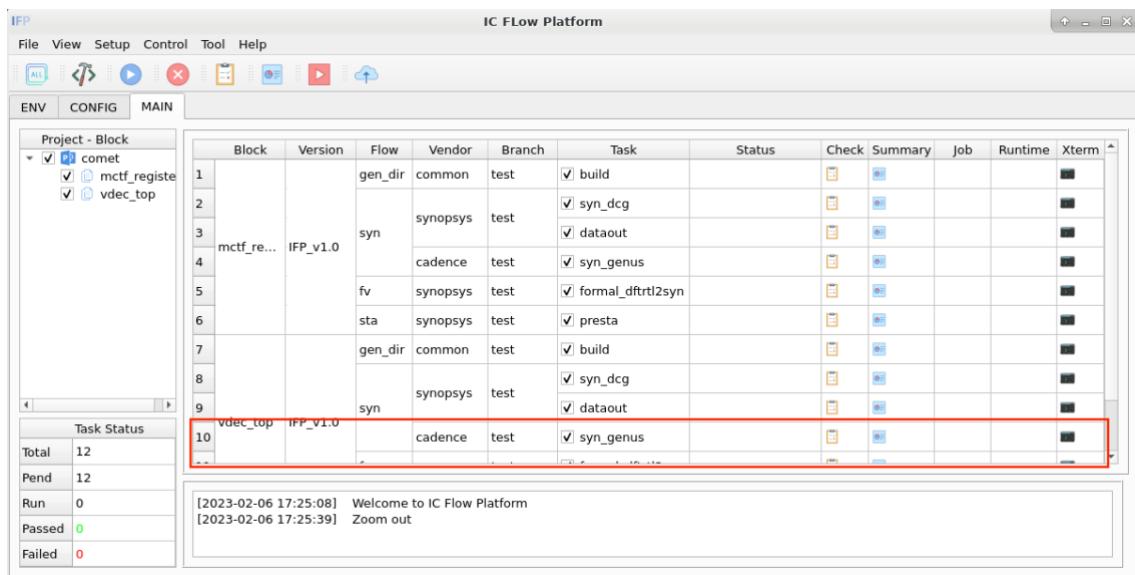
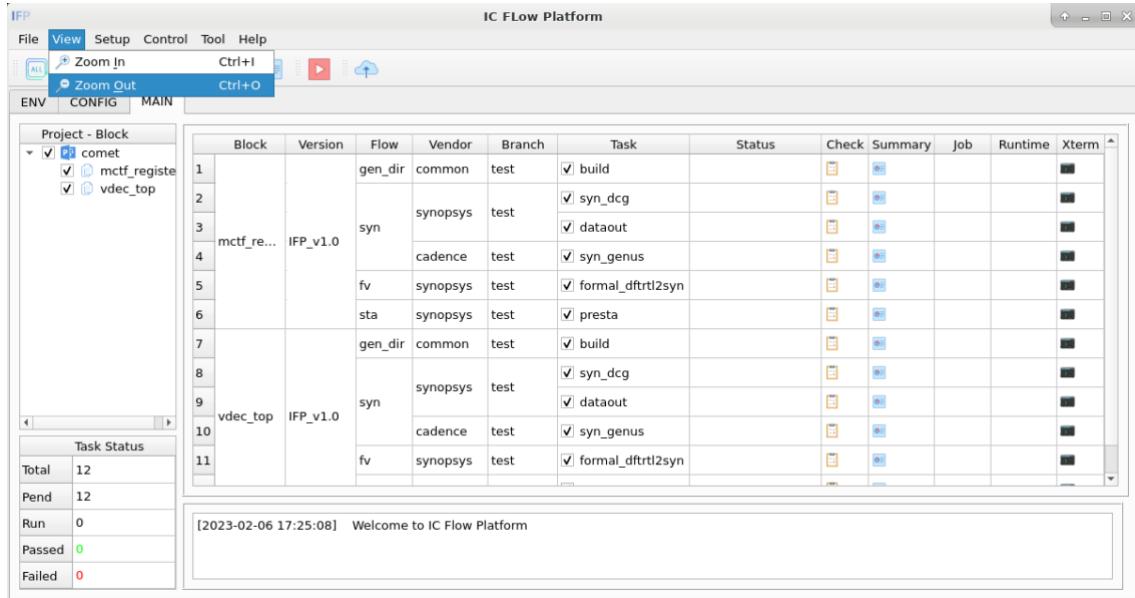
- **Zoom In**

增加 ifp 主界面的高度，点击一次放大一行。



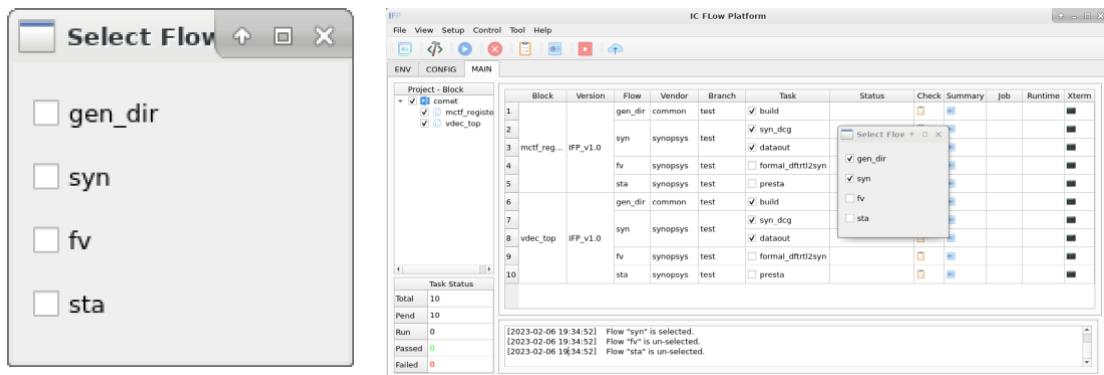
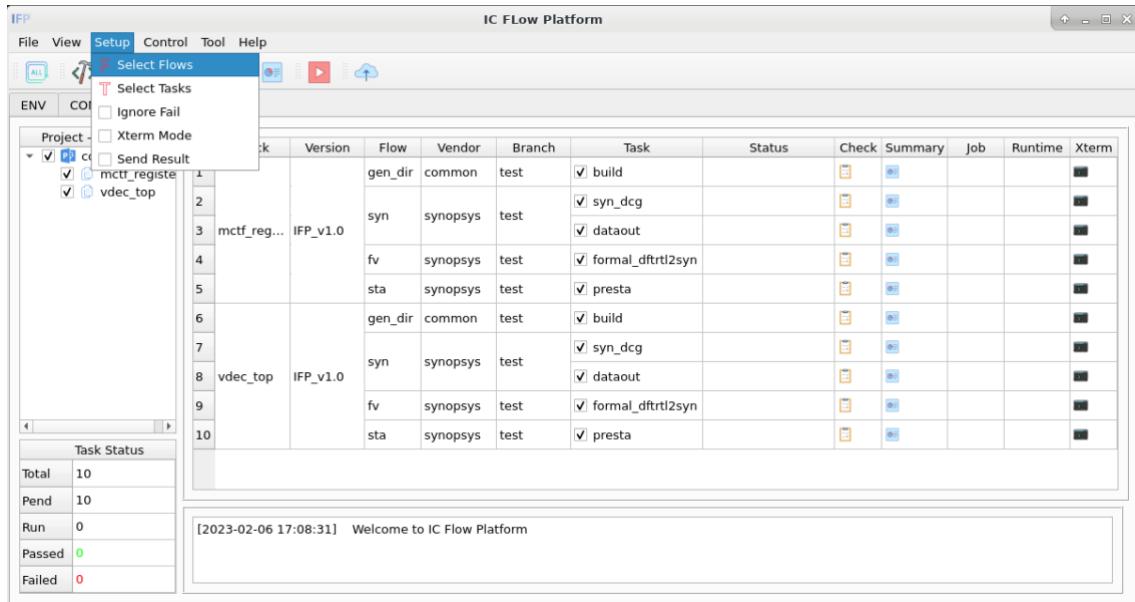
- **Zoom Out**

减少 ifp 主界面的高度，点击一次缩短一行。



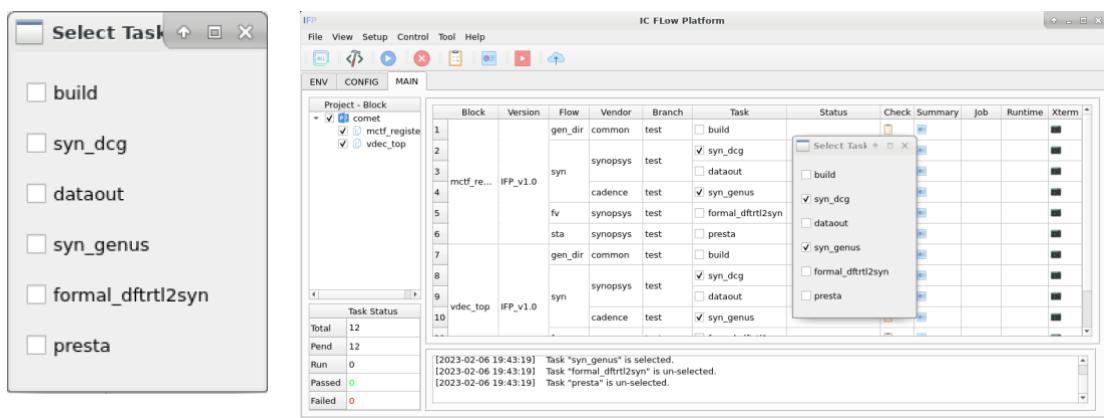
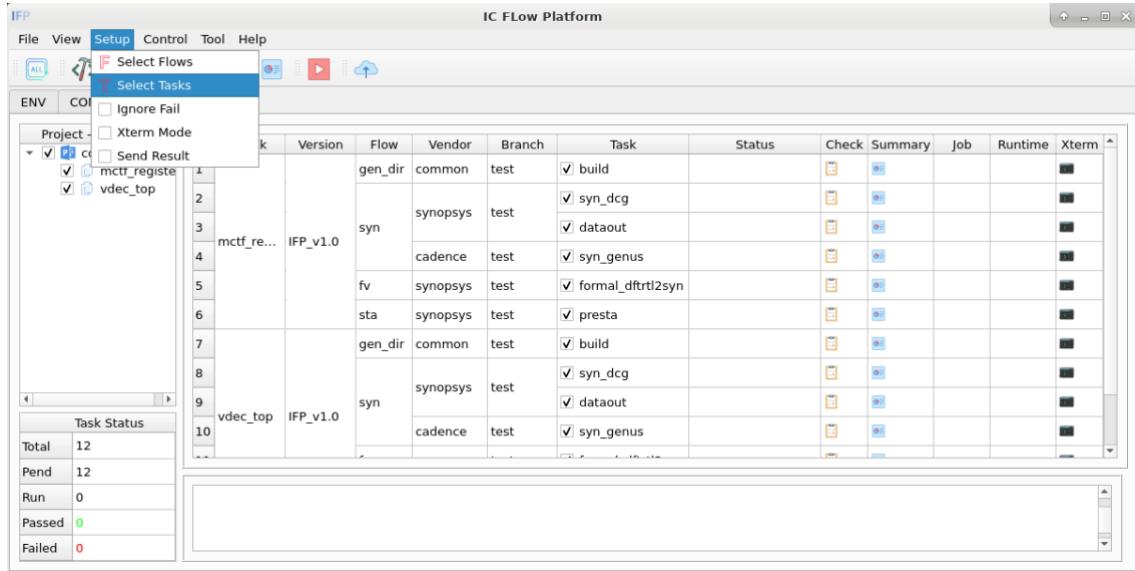
- **Select Flows**

批量选中主页中的 Flow 项，例如选中 flow 为 gen_dir 和 syn 项。



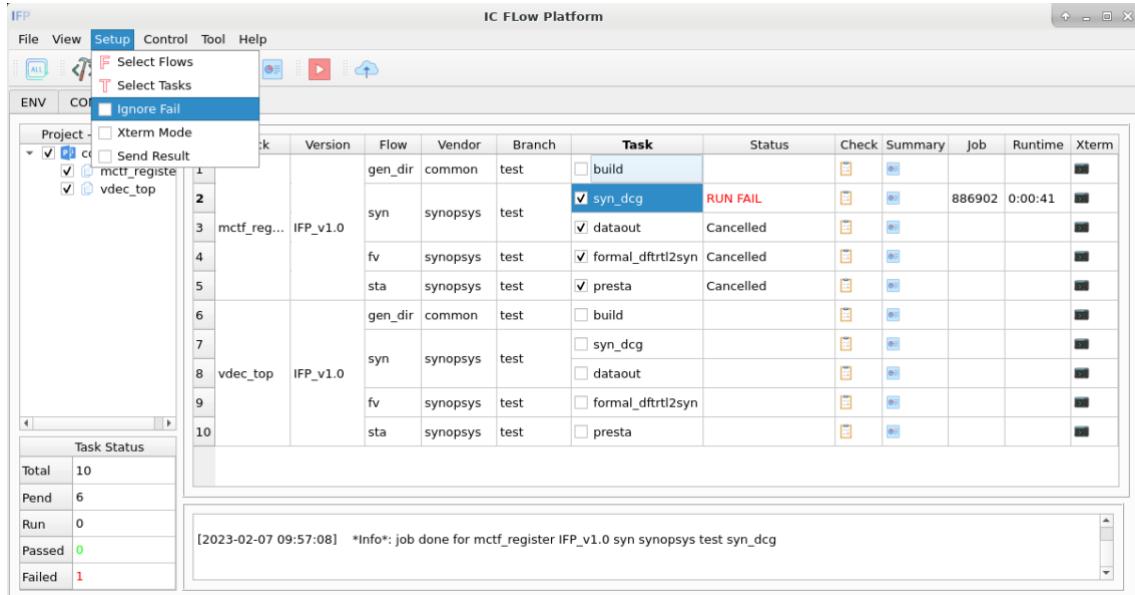
- **Select Tasks**

批量选中主页中的 Task 项，例如选中 syn_dcg 和 syn_genus 两个 task 项。

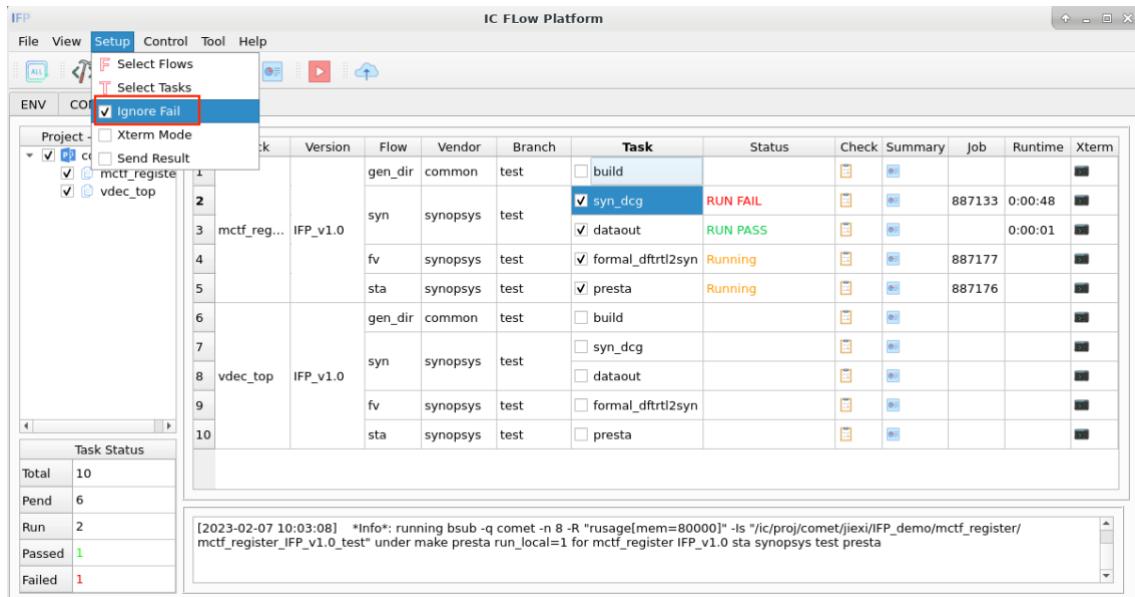


- **Ignore Fail**

若不选中 Ignore Fail, 串行 Task 中, 前者 task 失败后后者 task 会自动取消, 如下:

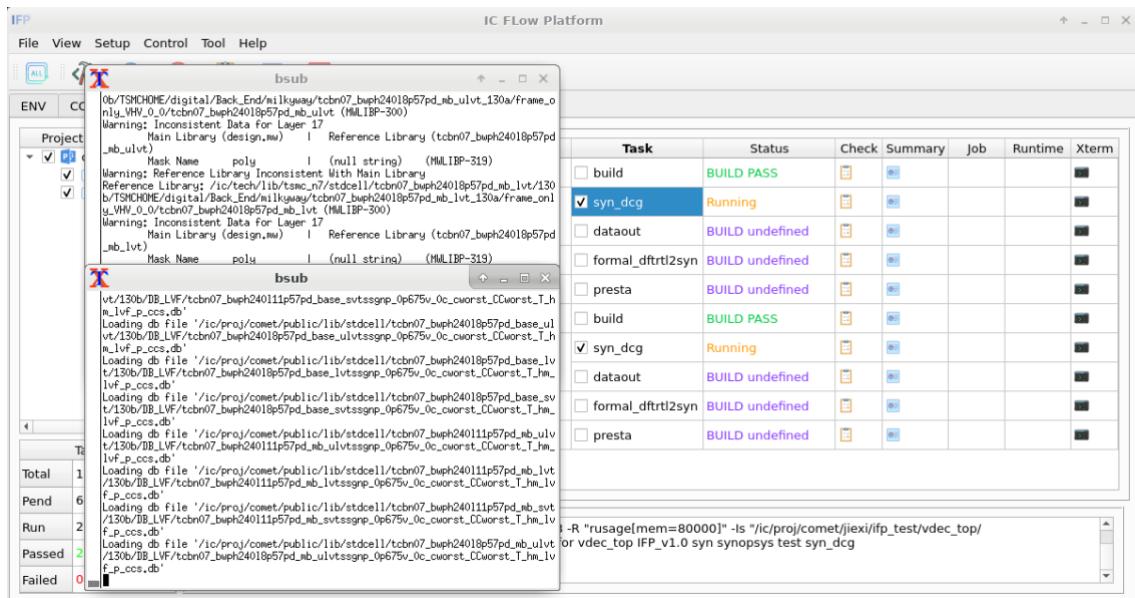
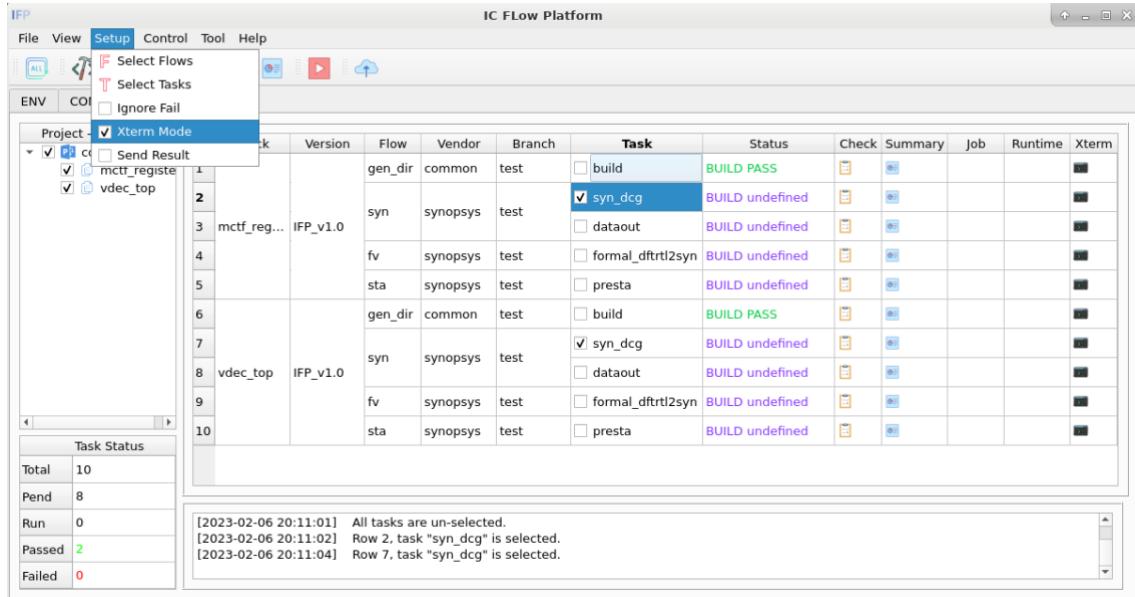


如果选中了 Ignore Fail 选项, 则前者失败后后者 task 仍然运行。



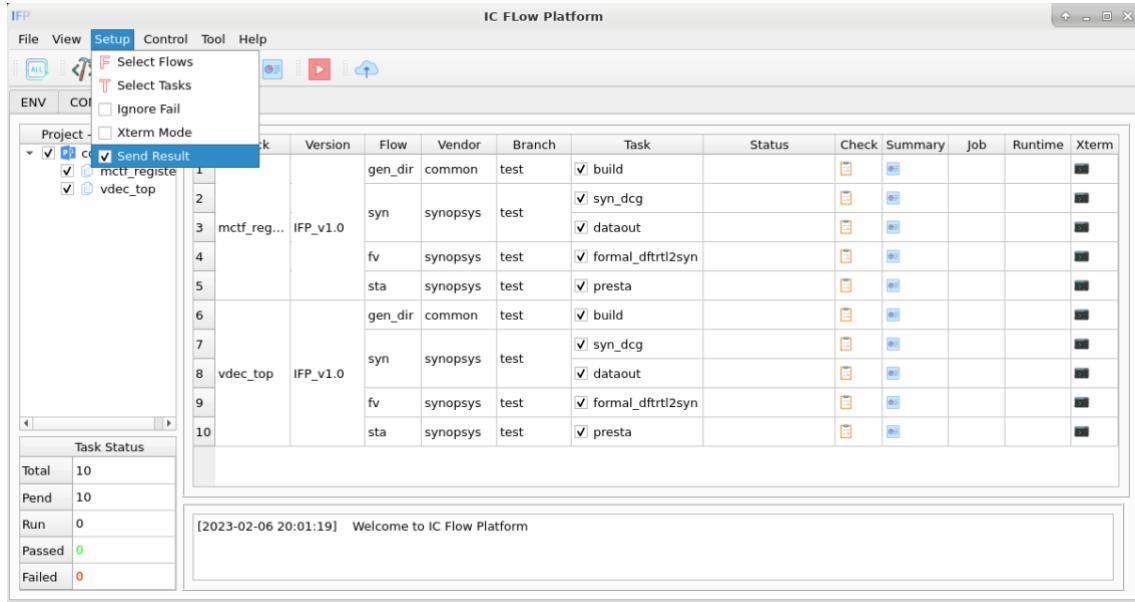
- **Xterm Mode**

针对需要运行的 job，选中 Xterm Mode 后点击 Run 按钮运行任务，则会在新开的 Xterm 执行，如下。



- **Send Result**

如下选中 Send Result, 然后点击 All Steps 运行 job, 会将 ifp 的汇总结果信息发送给用户。



发送信息及结果如下：

The screenshot shows the IC CAD interface. On the left, there's a message box with the title 'API' and 'IC CAD' containing the following text:

IFP RESULT
User : jiexi
Directory : /ic/proj/comet/jiexi/IFP_demo
Config : /ic/proj/comet/jiexi/IFP_demo/ifp.cfg.yaml
Total 10 tasks, 6 pass.

To the right of the message box is a 'Task Status' table:

Total	10
Pend	4
Run	0
Passed	6
Failed	0

Below the table, a message box shows: [2023-02-06 22:35:50] Send result.

4.6.2 主界面功能介绍

主界面上，我们可以看到 Task 的如下信息（或者访问如下功能）。

- **Check**：如果存在文本图标，可以查看 checklist 报告。
- **Summary**：如果存在文本图标，可以查看 summary 报告。
- **Job**：如果显示数字，可以得知采用 LSF 分布式并行运行的 Task 任务的 jobid，点击数字，可以使用自带的工具 lsfMonitor 查看任务信息。
- **Runtime**：如果 Task 是 LSF 任务并且已完成，此处可以看到其 runtime。
- **Xterm**：点击黑色按钮，弹出 Xterm 供用户操作。

IC Flow Platform

Block	Version	Flow	Vendor	Branch	Task	Status	Check	Summary	Job	Runtime	Xterm
1		gen_dir	common	test	<input type="checkbox"/> build	BUILD PASS					
2					<input checked="" type="checkbox"/> syn_dcg	Running			810976		
3	mctf_re...	IFP_v1.0	syn		<input checked="" type="checkbox"/> dataout	Queued					
4			cadence	test	<input checked="" type="checkbox"/> syn_genus	Running			810973		
5			fv	synopsys	<input checked="" type="checkbox"/> formal_dfrtl2syn	Queued					
6			sta	synopsys	<input checked="" type="checkbox"/> presta	Queued					
7			gen_dir	common	<input type="checkbox"/> build	BUILD PASS					
8					<input checked="" type="checkbox"/> syn_dcg	Running			810974		
9	vdec_top	IFP_v1.0	syn		<input checked="" type="checkbox"/> dataout	Queued					
10			cadence	test	<input checked="" type="checkbox"/> syn_genus	Running			810975		
11			fv	synopsys	<input checked="" type="checkbox"/> formal_dfrtl2syn	Queued					

Project - Block

- comet
 - mctf_register
 - vdec_top

Task Status

Total	12
Pend	6
Run	4
Passed	2
Failed	0

Check

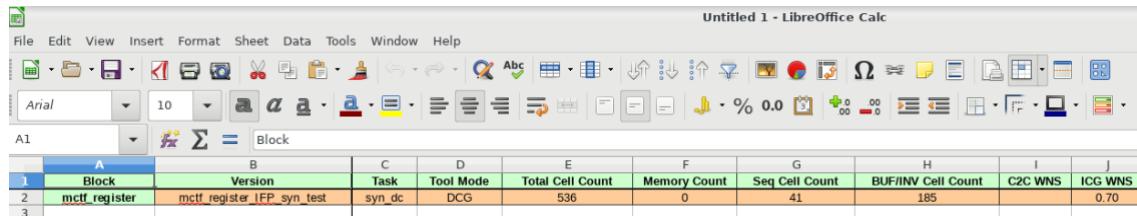
点击 Task 行的 Check 图标，会对当前 Task 做 checklist 检查，并弹出 checklist 检查报告。(如果已设置)

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.	
4 REVIEW	Check "Warning" message on detail log file.	
5 REVIEW	Get RTL version.	
6 REVIEW	Review link_design rpt	
7 REVIEW	Review final qor report	
8 REVIEW	Review check_design rpt	
9 REVIEW	Review check_timing rpt	

Summary

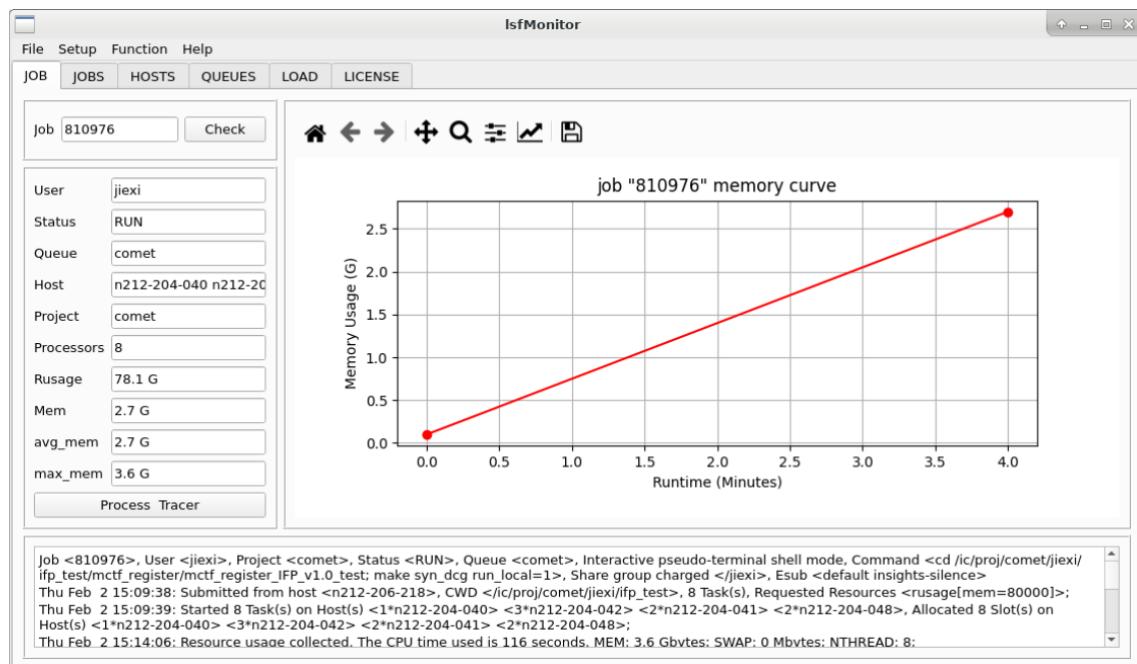
点击 Task 行的 Summary 图标，会对当前 Task 做 Summary 检查，并弹出 summary 汇总报告。(如果已设置)



A	B	C	D	E	F	G	H	I	J
Block	Version	Task	Tool Mode	Total Cell Count	Memory Count	Sed Cell Count	BUF/INV Cell Count	C2C WNS	ICG WNS
1	mctf_register	mctf_register(IFP_syn_test)	syn_dc	DCG	536	0	41	185	0.70
2									
3									

Job

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



Xterm

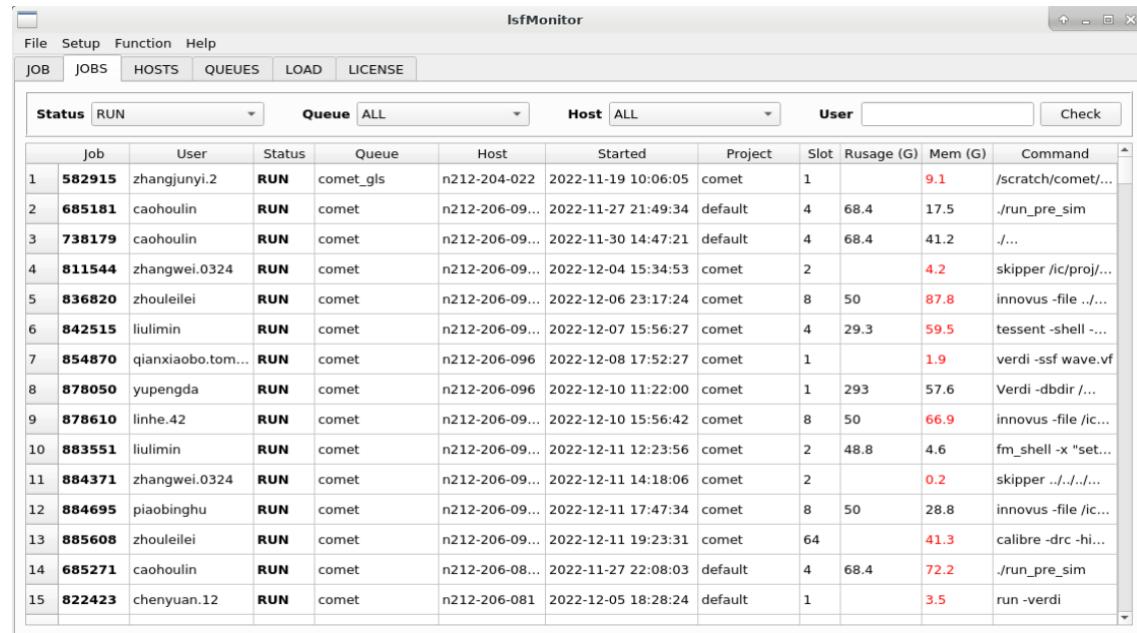
点击 Task 行的 Xterm 图标，会弹出一个 Xterm，并自动进入 Task 的路径。（Run 路径或者 Build 路径，默认是当前运行路径）



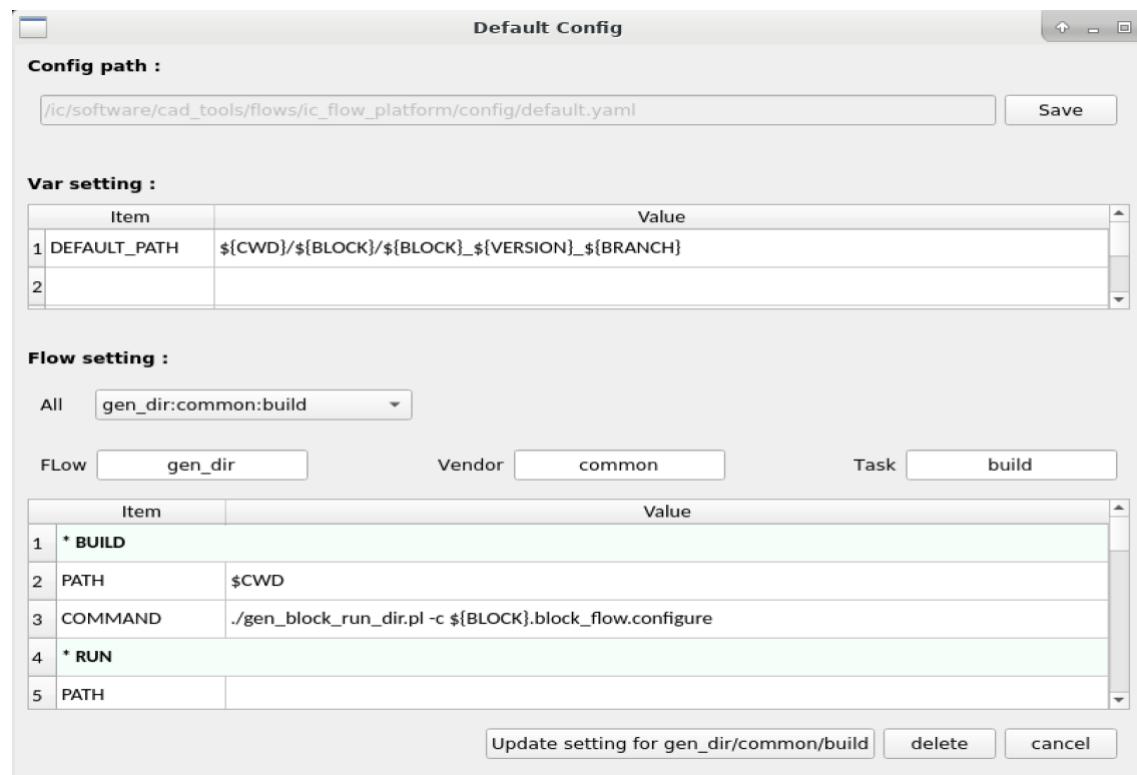
4.6.3 辅助工具

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

lsfMonitor 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。



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。



五、范例

5.1 用户环境准备

5.1.1 EDA 配置文件

EDA 配置文件由 project engineer 自行准备。

5.1.2 ifp 用户配置文件

ifp 用户配置文件，即 ifp.cfg.yaml，用于配置用户任务信息。由于每个用户运行的任务，以及任务运行方式相对固定，因此我们可以用脚本的方式辅助便捷生成 ifp.cfg.yaml，此处以自用脚本 pre_ifp 为例，帮助用户为指定的 block 自动生成 ifp.cfg.yaml 文件：

```
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:
```

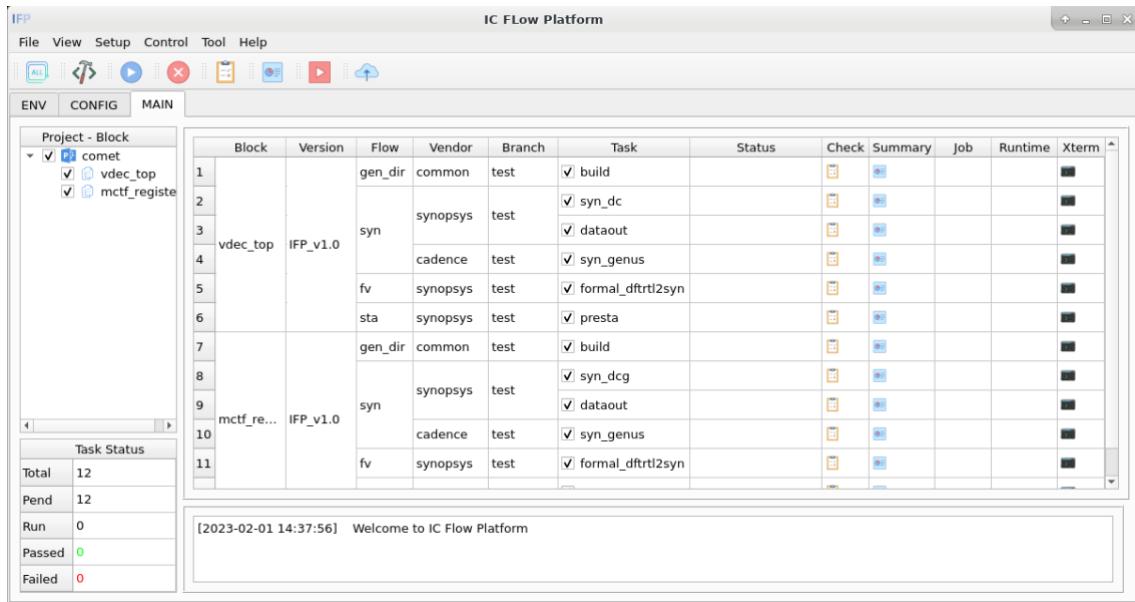
注意：ifp 配置文件 ifg.cfg.yaml 有两种生成方式：

- 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'
```



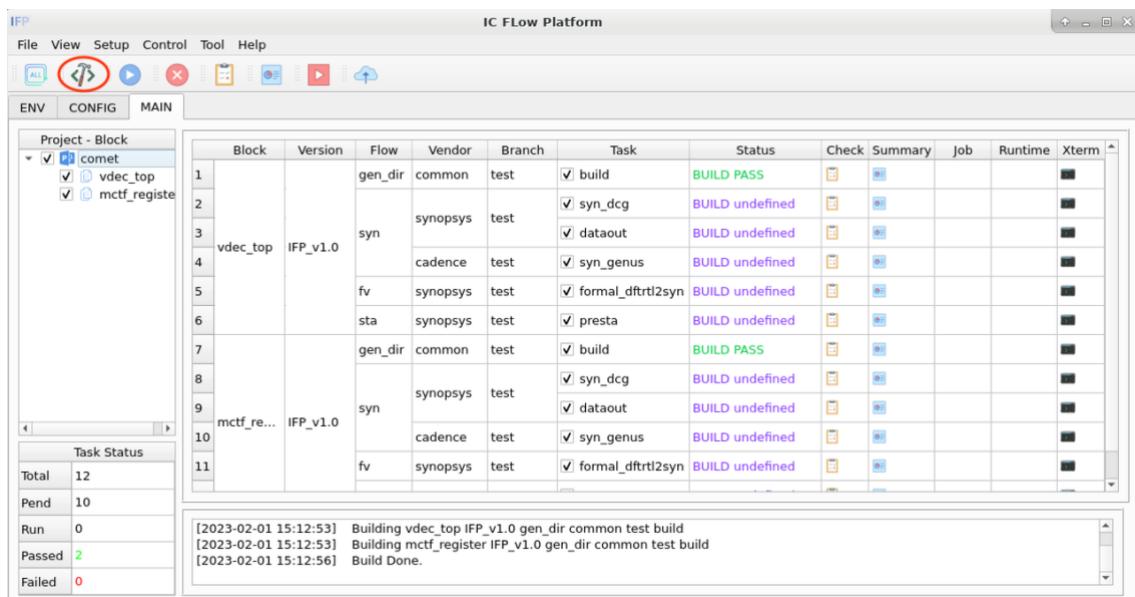
确认 ifp.cfg.yaml 后，即可通过 ifp 流程平台进行任务运行：

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，

也可以分步骤执行：

5.2.1 Build

点击 build 图标，完成目录创建。



Bash

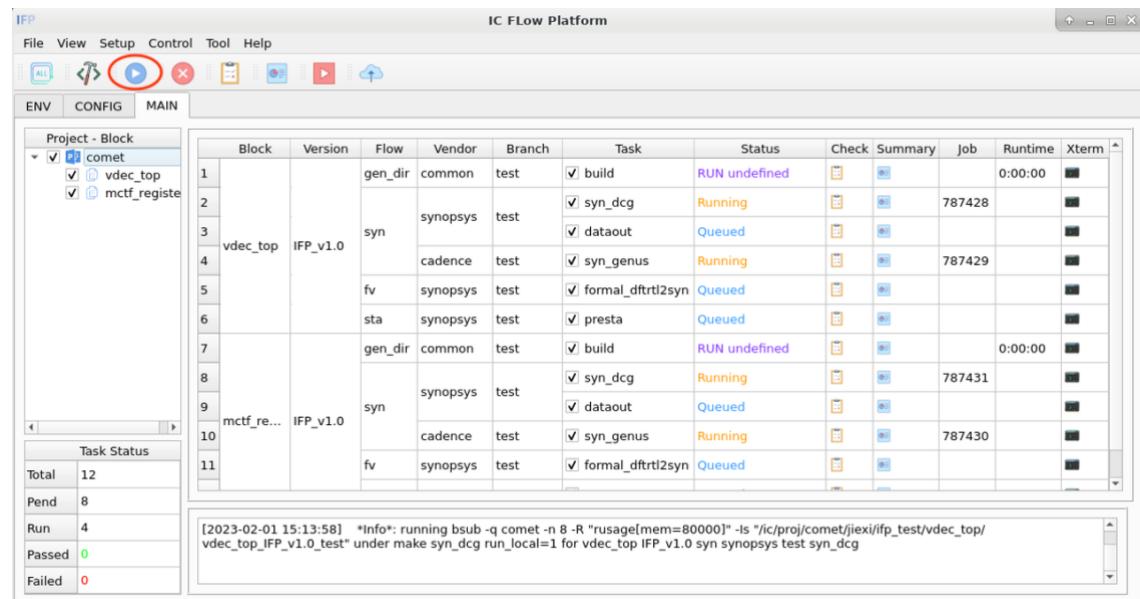
```
[n212-206-218]: /ic/proj/comet/jiexi/ifp_test/ls /*/
mctf_register/mctf_register_IFP_v1.0_test/:
```

```
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
```

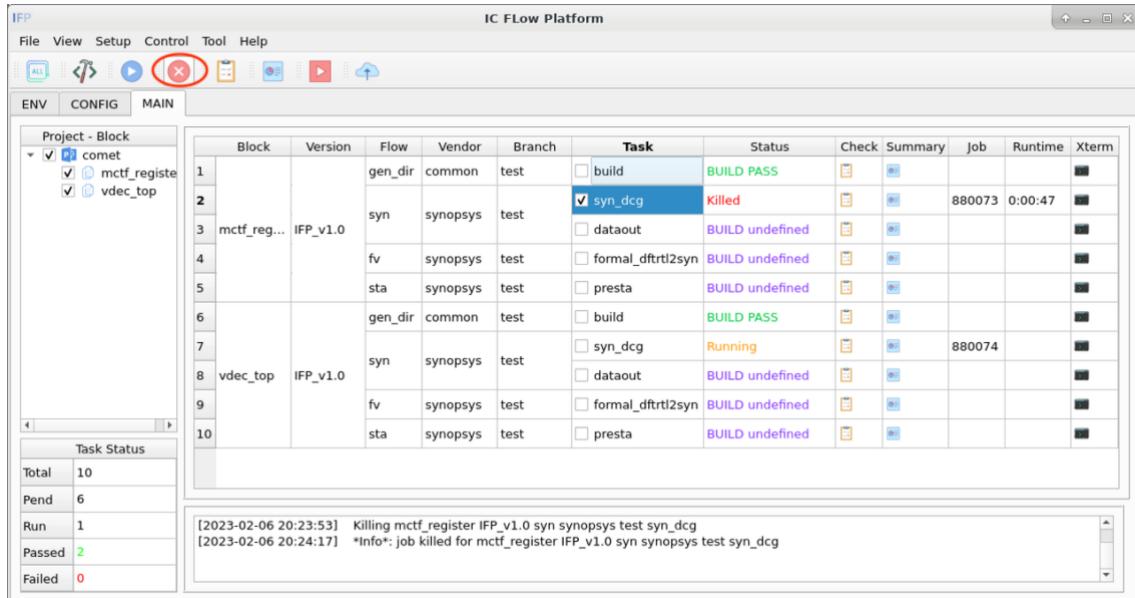
5.2.2 Run

点击 Run 图标，调用 EDA 工具提交 job。



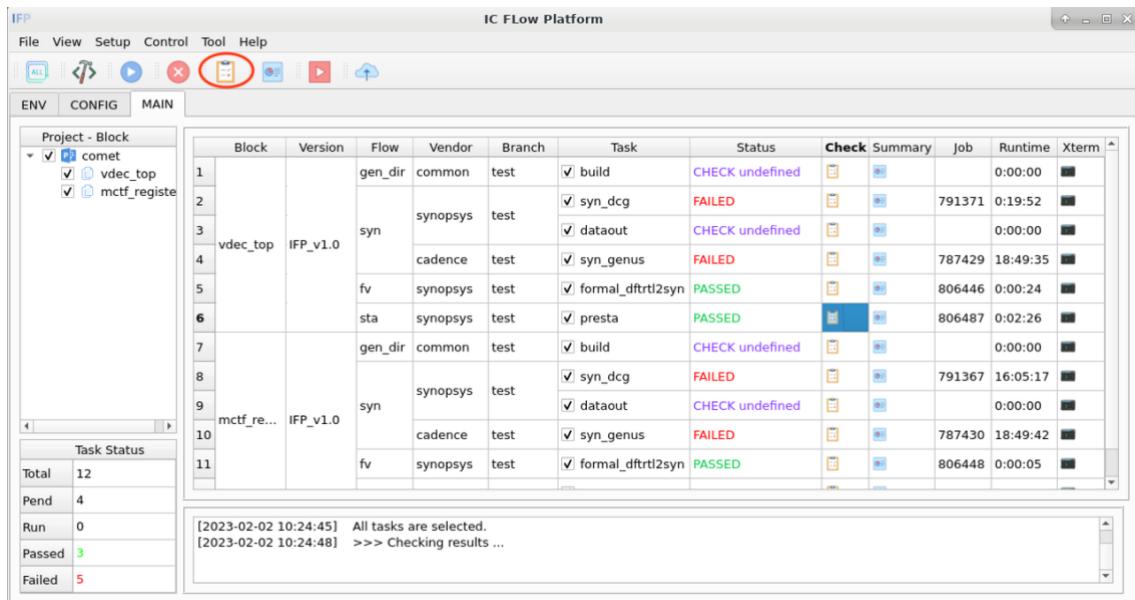
5.2.3 Kill

点击 Kill 图标，终止选中的正在运行的 job，Status 从“Killing”变为“Killed”，则证明 job 完成终止，如下：



5.2.4 Check

点击 Check 图标，进行 checklist 项目检查。



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

Result	Description	Detail
PASSED	Check "Error" message on log file.	<input type="button" value="view"/>
REVIEW	Check "Warning" message on log file.	<input type="button" value="view"/>
REVIEW	Get FORMAL version.	<input type="button" value="view"/>
PASSED	Compare result pass.	
REVIEW	Review status rpt	<input type="button" value="view"/>
REVIEW	Review black box type	<input type="button" value="view"/>
REVIEW	Review don't verify points	<input type="button" value="view"/>
REVIEW	Review failed points	<input type="button" value="view"/>
REVIEW	Review abort points	<input type="button" value="view"/>

Result	Description	Detail
FAILED	Check "Error" message on syn_dc/dcg log file.	<input type="button" value="view"/>
PASSED	Check /ic/proj/comet/jexi/ifp_test/vdec_top/vdec_top_IFP.log file.	<input type="button" value="view"/>
REVIEW	Check File : /log/syn_dc.log (check errormessage)	<input type="button" value="view"/>
REVIEW	Check File : /log/syn_dsp.log (check errormessage)	<input type="button" value="view"/>
REVIEW	Check Line 5441 : Error! Cannot find DEF file please check	<input type="button" value="view"/>
REVIEW	Get R	<input type="button" value="view"/>
REVIEW	Revie	<input type="button" value="view"/>
REVIEW	Revie	<input type="button" value="view"/>
REVIEW	Revie	<input type="button" value="view"/>
REVIEW	Revie	<input type="button" value="view"/>

5.2.5 Summary

点击 Summary 图标，调用 qor 脚本自动收集 PPA 数据信息。

Block	Version	Flow	Vendor	Branch	Task	Status	Check	Summary	Job	Runtime	Xterm
1		gen_dir	common	test	<input checked="" type="checkbox"/> build	SUMMARY ...	<input type="button" value=""/>	<input type="button" value=""/>		0:00:00	
2					<input checked="" type="checkbox"/> syn_dcg	SUM PASS	<input type="button" value=""/>	<input checked="" type="checkbox"/>	791371	0:19:52	
3		syn	synopsys	test	<input checked="" type="checkbox"/> dataout	SUMMARY ...	<input type="button" value=""/>	<input type="button" value=""/>		0:00:00	
4	vdec_top	IFP_v1.0	cadence	test	<input checked="" type="checkbox"/> syn_genus	SUM PASS	<input type="button" value=""/>	<input type="button" value=""/>	787429	18:49:35	
5		fv	synopsys	test	<input checked="" type="checkbox"/> formal_dftrtl2syn	SUM PASS	<input type="button" value=""/>	<input type="button" value=""/>	806446	0:00:24	

A	B	G	H	I	Memory
Block	Task Path	VT Ratio(area)	Gating Ratio	Multibit Ratio	
ydec_top	/ic/proj/comet/jexi/ifp_test/vdec_top/vdec_top_IFP_v1.0_test/syn_dc	LVT 19.05%/SVT 80.95%/ULVT 0.00%	0.00%	0.00%	1
4					

5.2.6 Release

选中需要 release 的 task，点击 Release 图标，进行数据信息的交付。

The screenshot shows the IC FLOW Platform software interface. The main window title is "IC FLOW Platform". The menu bar includes File, View, Setup, Control, Tool, and Help. The toolbar contains various icons, with the last one being a blue cloud icon with an upward arrow, which is highlighted with a red circle. The tabs at the top are ENV, CONFIG, and MAIN, with MAIN selected. On the left, there's a tree view under "Project - Block" showing "comet" with sub-items "mctf_register" and "vdec_top". The central area is a table titled "Task Status" with columns: Block, Version, Flow, Vendor, Branch, Task, Status, Check, Summary, Job, Runtime, and Xterm. Rows 1 through 10 show tasks for "gen_dir", "syn_dcg", "dataout", "formal_dftrtl2syn", and "presta" across different blocks and versions. Row 7, "syn_dcg", has a checked checkbox in the "Task" column and is highlighted with a blue background. The status for this row is "RELEASE PASS". A message box at the bottom displays two log entries: "[2023-02-07 00:22:03] >>> Releasing..." and "[2023-02-07 00:22:29] >>> Release Done". On the far left, a "Task Status" panel shows the total count of 10 tasks, with 4 pending, 0 running, 6 passed, and 0 failed.

release 数据结果如下：

```
00:28 [n212-206-218]: ~/ll /ic/proj/comet/public/release/syn/vdec_top/IPF_v1.0_S0207A
total 48
drwxr-xr-x  2 ic_admin proj_bluewhale 4096 Mar  8  2022 SAIF
drwxr-xr-x  2 ic_admin proj_bluewhale 4096 Mar  8  2022 DB
drwxr-xr-x  2 ic_admin proj_bluewhale 4096 Mar  8  2022 UPF
drwxr-xr-x  2 ic_admin proj_bluewhale 4096 Mar  8  2022 DEF
drwxr-xr-x  2 ic_admin proj_bluewhale 4096 Feb  6 00:34 SDC
drwxr-xr-x  2 ic_admin proj_bluewhale 4096 Feb  6 00:34 RPT
drwxr-xr-x  2 ic_admin proj_bluewhale 4096 Feb  6 00:34 CONS
drwxr-xr-x  2 ic_admin proj_bluewhale 4096 Feb  6 20:18 NETLIST
drwxr-xr-x  4 ic_admin ic_work_group 4096 Feb  7 00:26 ..
drwxr-xr-x 10 ic_admin ic_work_group 4096 Feb  7 00:26 .
-rw-r--r--  1 ic_admin ic_work_group   171 Feb  7 00:26 .release
-rwxr-xr-x  1 ic_admin ic_work_group   743 Feb  7 00:26 release.20230207_002603_jiexi.log
```

附录

附一、变更历史

日期	版本	变更描述	源代码变更
2023.2.2	1.0	代码上传 https://github.com/bytedance/ic_flow_platform	
2022.12.14	1.0	按照 PEP8 规范修正所有 python 代码。	所有 python 脚本。

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
```

```

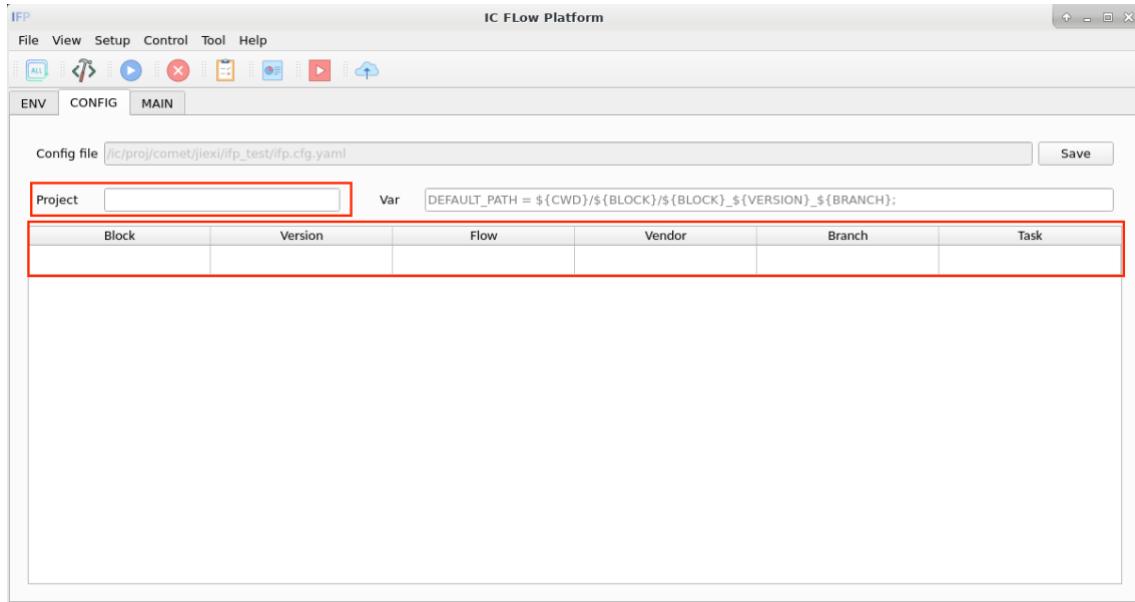
IFP_v1.0(RUN_ORDER=gen_dir,syn,fv|sta):      ⇒ version 信息
    gen_dir:      ⇒ flow 信息
        common:
            test(RUN_TYPE=serial):      ⇒ branch 信息
            build: {}      ⇒ task 信息
    syn:      ⇒ flow 信息
        synopsys:
            test(RUN_TYPE=serial):      ⇒ branch 信息
            syn_dcg: {}      ⇒ 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。

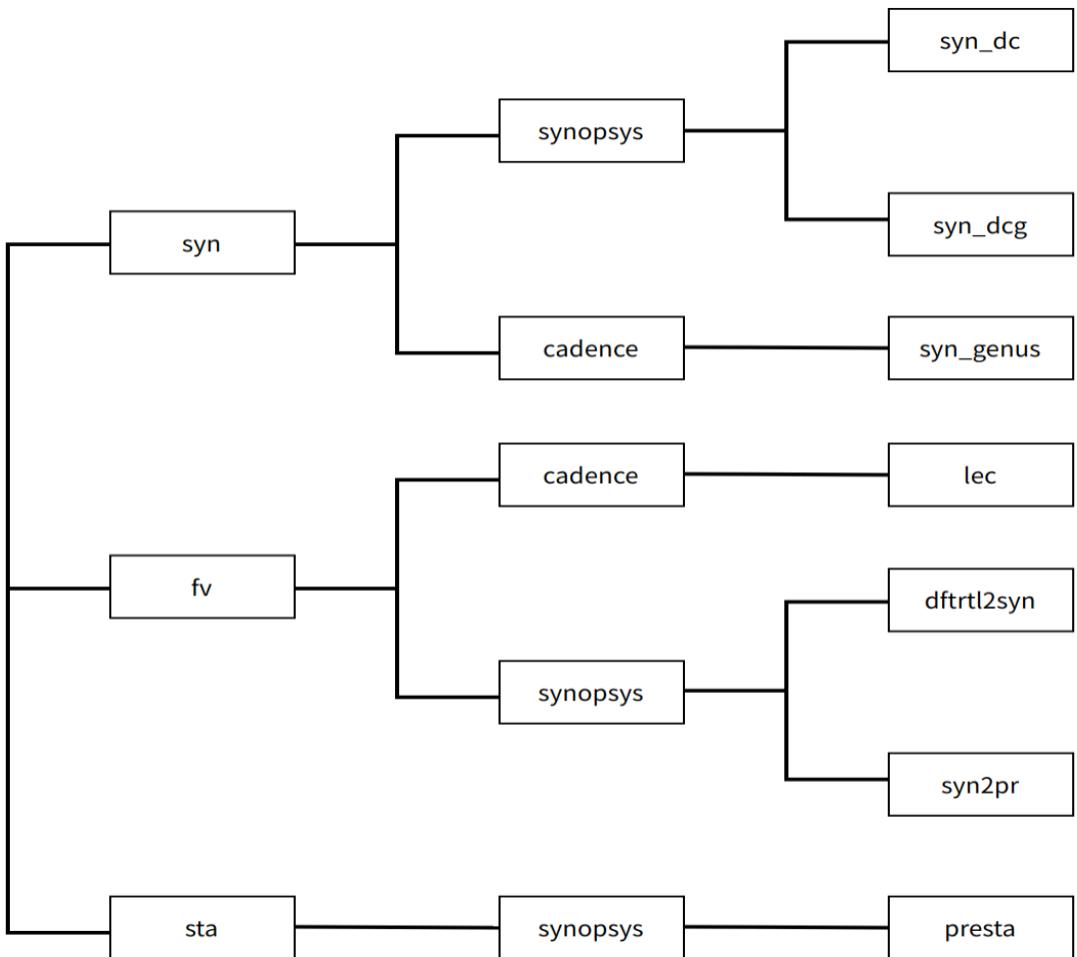


其中 Project 指定项目名称，单击编辑框直接输入；Block/Version/Branch 需要双击点击进行输入；

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 \leftrightarrow proj_name
Block \leftrightarrow DESIGN
Version \leftrightarrow RTL_VERSION
Branch \leftrightarrow SYN_VERSION

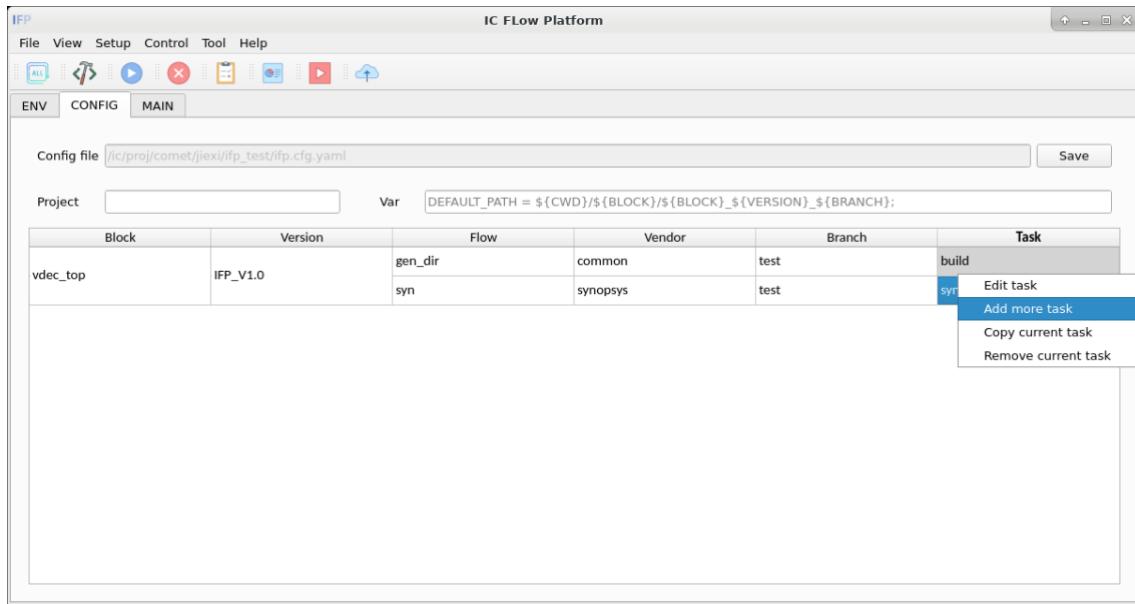
Flow/Vendor/Task 则根据需要运行的 task 进行编辑，如下：



根据以上注意事项编辑 config 界面信息，如下：

- **Task 右键编辑**

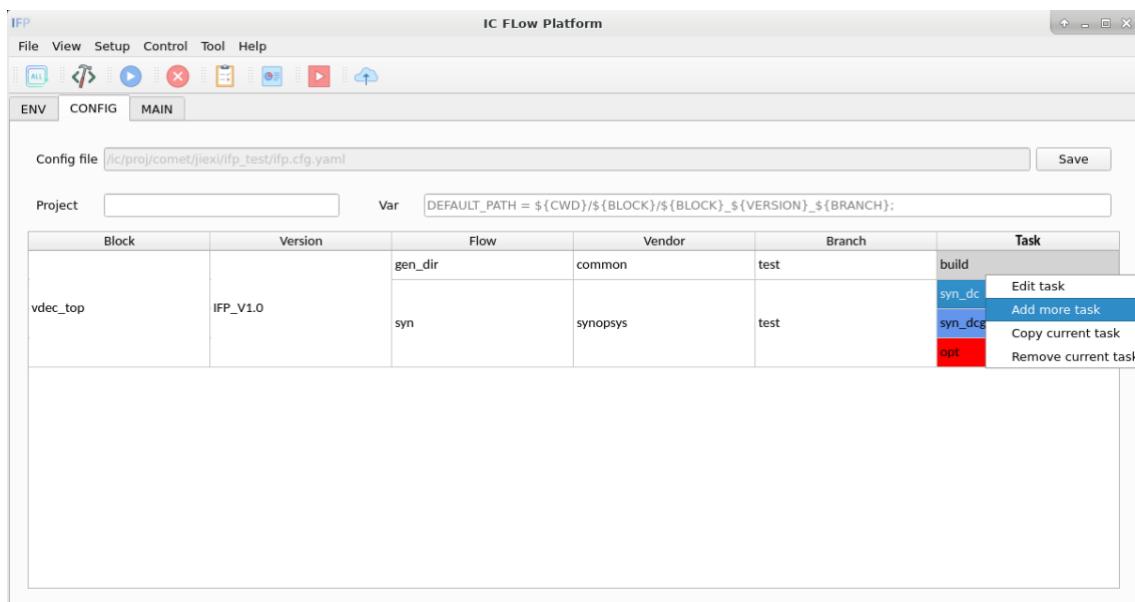
包含“Edit task”，“Add more task”，“Copy current branch”和“Remove current branch”四部分。



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；



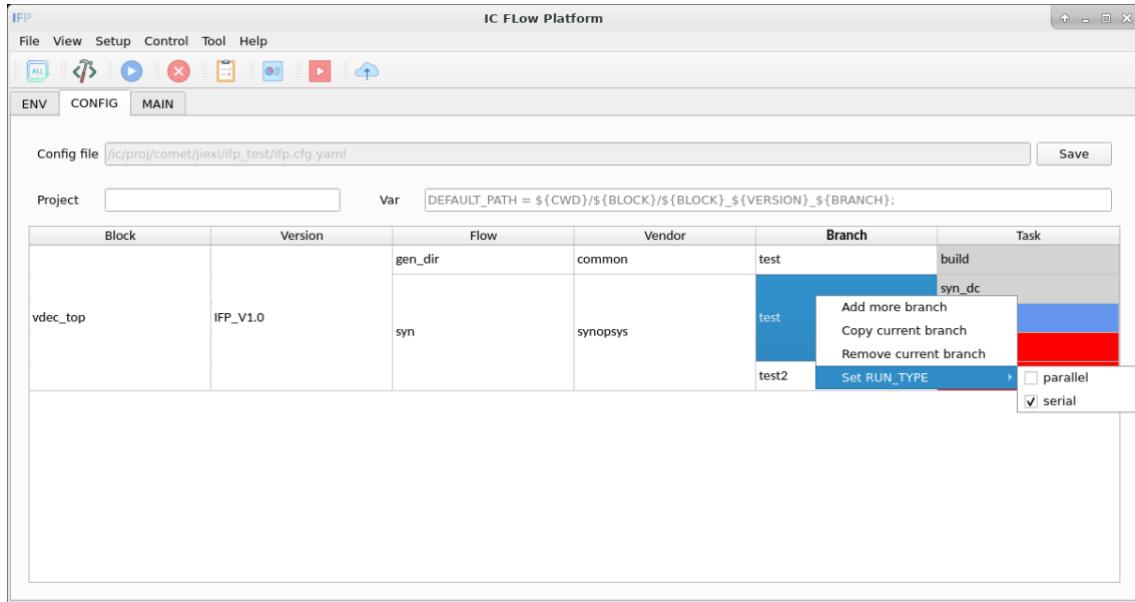
Copy current task: 复制当前 task 项目。

Remove current task: 删除当前 task 项目。

- **Branch 右键编辑**

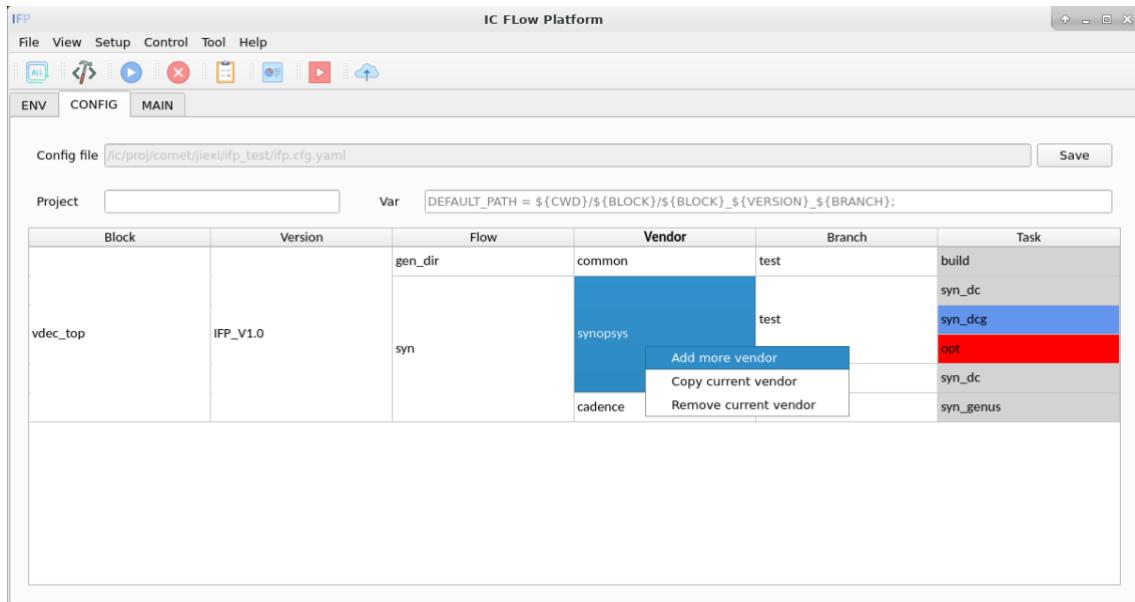
包含“Add more branch”, “Copy current branch”, “Remove current branch”和“Set RUN_TYPE”四部分。

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。



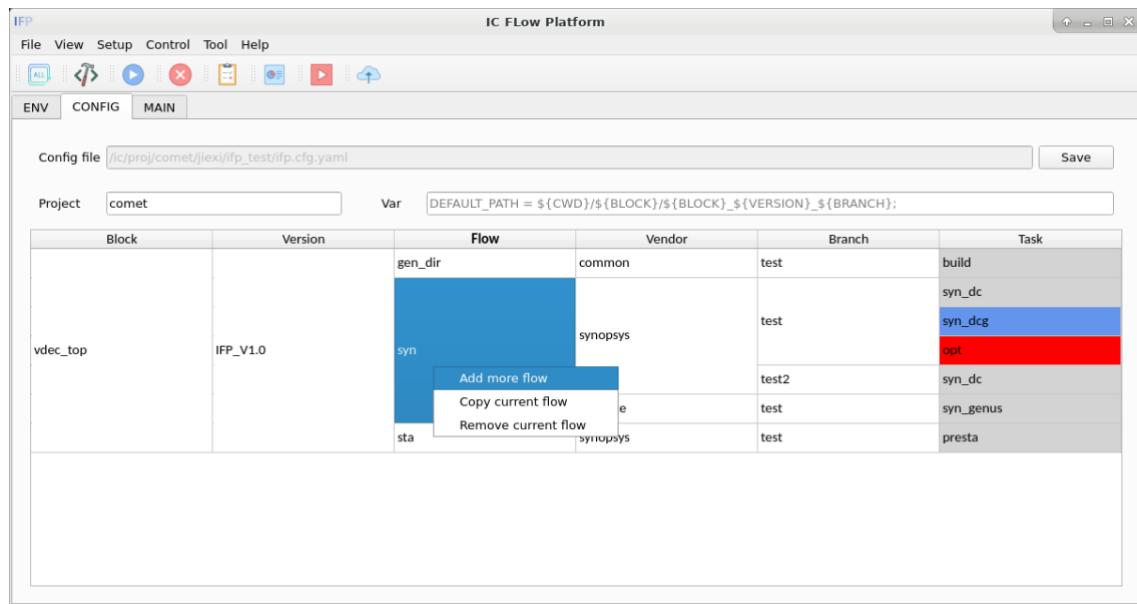
- **Vendor 右键编辑**

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**

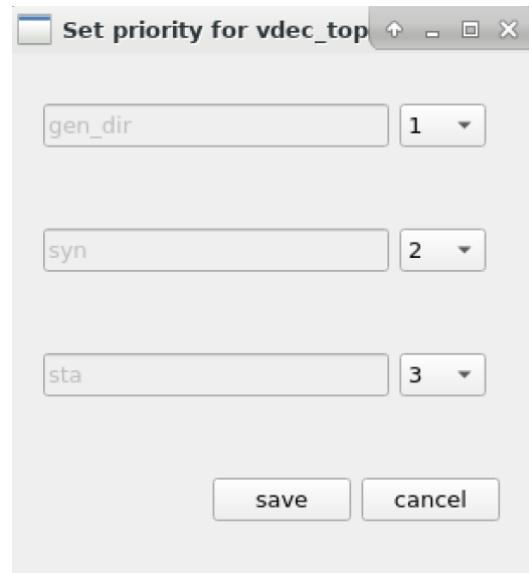
包含“Add more flow”, “Copy current flow”和“Remove current flow”三部分。如下新增 flow 为 sta 相关的 task 信息。



- **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。

