# 水滴计划学习总结

赵正杰
zj.zhao.work@foxmail.com

## I. 个人背景

本科专业是电子科学与技术，主要项目经历中使用的语言是 C 语言，日常学习中不怎么写代码。

硕士学习的专业是软件工程专业，主要研究方向为 EDA 领域内的静态时序分析。

C++方面，在本科时自学过 C++基础，但是缺乏软件体系,大型软件管理，vscode 跨语言 debug 的知识。

## II. C++标准化训练

### A. 学习过程

因为本科阶段自学 C++的时候看过《C++ primer plus》当时就觉得这本书写的太详细了，也是自己做了笔记，不过考研以后就没写代码了，也是借助这个机会重新复习了一遍 C++的基础知识。

此外还通过水滴计划学习了以前只是范范了解的东西，比如 log 文件的编写，Cmake 对项目的关系，软件的体系架构，C++新特性，跨语言的编译。

### B. C++工具掌握

由于之前都是使用 Visual Studio 进行单一语言的开发，所以我想当然的认为跨语言的代码无法进行 debug，于是当我需要对 OpenROAD 和 iEDA 这种跨语言的大型项目进行 debug 的时候只知道使用最古老的输出的方式去调试，但是这样效率底下且很难对整体的结构进行全面的了解（每个类中都包含什么，类与类之间的关系是什么）。一次偶然的机会，在与龙老师的交流中他发给我一份 launch.json 文件，从此开启新大门，我突然发现 vscode 竟然这么好用，它竟然可以通过配置文件的方式直接调试代码，而且具有很高的扩展性。

在后续的学习中我陆陆续续纠正了过去一年里走过的弯路:

1）我开始配置 launch.json,tasks.json 等相关 vscode 的 json 文件，并通过 gpt 理解了他们每一行命令是在干什么，而不是每次遇到问题的时候直接从网上薅一个过来，不行就换一个继续试的愚蠢方式。

2）我开始在完成一个项目的时候尝试构思如何做到像大型软件一样文件之间条理清晰，而不是让软件体系结构的知识仅仅停留在纸面。比如 src，bin，header，third-party，readme 等文件的收纳。这让我意识到大项目都是从小项目做起的，没有人能一天搞定数十万行代码的开发。

3）与架构对应的是如何使得将一个大型项目拆分成小块以后，整体仍然能够轻松编译，并且还要在修改完成后加速编译（Makefile），于是就引入了 CMake 管理方式，我开始试着使用 CMake 生成 makefile 而不是从网上 git 下来别人的库以后，直接一套丝滑小连招

```
(
    mkdir   build && cd  build
    cmake   ..
    make
),
```

通过自己编写细分到每一个子模块的 CMakelists.txt 我开始真正了解为什么要使用这么一个玩意对整体项目进行构建。

4)在完成上面重要内容之外,还学习了 Doxygen，GoogleTest 等知识，其中 test 这个步骤，我知道他很重要，但是苦于没有应用空间，我到现在仍然不太懂这个玩意在小型开发中有啥意义（可能能让测试变的更方便？）。至于 Doxygen 这个玩意，确实能够让别人快速了解你的代码的架构和类中属性，而且也不

需要什么学习成本，在注释前面加个’!’即可。

## C. C++训练项目

对于 A*的实训其实我觉得写出来这个代码是不难的，但是要在这个过程中实现对上述 4 点知识的应用，尤其是对于刚接触这些知识的人来说还是有点烦，毕竟几百行的代码能用一个 main 函数解决，为什么要大费周章的拆分成几个模块呢？但是实际上，如果坚持用上述提到的工具的话其实是可以更快了解如何开发 EDA 软件的基础知识的。

# III. EDA 业务训练

## A. 知识学习

在 eda 的学习过程中，主要是依托于那几本经典图书，ieda 平台和各位老师开设的讲座。我个人认为这个模式是很好的,通过书本学习基础知识并在 ieda 上学习 flow 是怎么运行的，然后在讲座的时候听老师答疑解惑。可惜的是，在这段时间里还有实验室的任务要完成，所以也只能做到浮于表面，但也确实让我更加理解其他的点工具都在干什么。

此外，我认为整个学习流程其实最大的收获是，除了 STA 以为，我知道了其余点工具的入门方法，这样在后续做多方向结合的时候可以借助这一经历，快速入门，而不是再从 0 开始探索。

## B. 学习心得

虽然在学习水滴计划之前我已经了解了 STA 的基本知识，并且做了一些小型项目。但是我一直有个疑惑，在当初我学习那几本书的时候，里面的概念我都清楚,但是组合在一起我就不明所以了。并且一个 EDA 流程有这么长，基本处于看了 B 忘了 A 的状态。以至于在很长一段时间都怀疑我这水平真的适合这个方向吗？但是后来跟同行的学生接触多了就发现，这个方向本身确实不是那么好做的，需要大量时间沉淀下去，急于求成往往只能使得自己越来越焦虑。

于是在这次培训中，我尝试慢下来，没有必要做到对每一个知识点都熟悉。而是着重理解他干了一件什么事。我可能不用知道他的底层逻辑是什么，但是我知道他输入了什么，经过什么约束，输出了什么就可以了。等到后续深入这个领域内的时候，日积月累自然就能理解为什么底层要以某种形式展开。

# IV. 培训计划建议

EDA 方面，由于 eda 点工具的数量众多，加上水滴计划也是面向初学者的课程,所以我感觉讲座的内容可以更加面向基础一些，比如可以加上与当前讲座内容相关点工具在 ieda 里实现的算法，这样经过老师对算法的讲解学生可以直接在后续实践中接触到已经听过的算法，而不是又重新接触一个全新的算法，这样就可以加深印象，我觉得可以在取得更好的效果的同时减少前期学习过程中反感的情绪。

除此之外我觉得EDA学习部分对于每个点工具的时间分配其实可以多一点，目前是一天一个点工具，但是我觉得时间有点紧，除非之前学过否则一天快速入门一个点工具，我觉得难度还是有点大。

对于 C++部分，我觉得可以多设置一些框架方面的题给同学们，因为现在 Ai 发展这么迅速，个人感觉对算法实现的要求也逐渐在下降。而选择什么样的算法，以什么形式，什么结构展现出来，这是 Ai 无法代替人类完成的。