1. 流程分析
2. 命令行解析接口：AR\_ReadArgs(int argc, char \*argv[])

class ArgRead {

public:

ArgRead();

~ArgRead();

void AR\_ReadArgs(int argc, char \*argv[]); //命令行解析接口

void AR\_Usage(); //命令行参数说明

string cnfFile; //配置文件

string primFile; //原语文件

string hsvFile; //待生成hsv文件

string ar\_commandLine; //命令行参数

string logFile; //日志文件

}

1. 配置文件解析

① 配置文件解析类

class LineParser {

public:

LineParser(const char \*filename,const char \*commentPrefix="#",const char \*continuationSuffix="\\",bool skipWhiteLines=1);

~LineParser();

bool ReadNextSection(string &line, bool &linesSkipped); //读取section

bool ReadSectionLine(string &line);

bool ReadKey(string &key, string &value); //读取key和value

bool ReadLine(string &line); //读取配置项里的一行

void SplitIntoWords(string &line,list<string> &words); //将value里面的各项存入list

int LineNumber(); //行数

private:

bool IsSection(string &line);

ifstream in; //输入文件流

string currentFilename; //当前配置文件名

string prefix;

string suffix;

bool skipWhite;

string lastSection;

int lineNumber;

regex\_t whitespaceExpr;

regex\_t sectionExpr;

regex\_t keyValuePairExpr;

regex\_t wordExpr;

};

② Library 配置项

i. Library里基本配置项的存储类

class Cell{

public:

Cell(string &n): name(n) {}

int I(); //输入端口个数

int O(); //输出端口个数

int T(); //终端个数(输入端口+输出端口)

double g(); //输出端口占比

const string &Name(); //取出基本单元名

virtual ~Cell() {}

protected:

string name; //基本单元(module)名

list<string>numInputs; //输入端口名

list<string>numOutputs; //输出端口名

}

ii. 基本存储类扩展

class Librarycell: public Cell{} //暂定

iii. Library 配置项存储

class Library{

public:

list<Cell \*> cells; //基本单元存储列表

~Library();

}

iv. Library 全局配置项存储

map<string,Library> Globals::libraries;

string:library名字,可以根据名字来区别不同的Library

Library:代表Library里所有的配置项

③ module配置项

i.

class ModuleType : public Cell{

public:

string instanceName; //module名

list<string> libraries; //存储库名

list<int> distribution; //基本单元的实例个数

map<int, Region> regions; //Region区间规则(用于连线)

int numBlocks; //包含基本单元的个数

}

1. 生成合法hsv文件