# Описание языка файла правил

Скрипт верификации содержит алгоритмы проверки норм конструктивно-технологических ограничений. В основе языка скрипта верификации лежит язык программирования Lua, расширенный специализированными функциями для проверки КТО: логические, топологические и геометрические операции над слоями и пр.

Ниже приведено описание функций, доступных для использования при разработке скрипта верификации.

## Объявление слоёв и наборов слоёв

Для удобства в пределах скрипта верификации для идентификации первичного слоя возможно использовать не только его числовой идентификатор (номер слоя) из файла GDSII, но и произвольное строковое обозначение. Для сопоставления номера слоя и строкового обозначения необходимо выполнить команду присвоения значения:

*имя\_переменной = номер\_слоя*

где *имя\_переменной* – произвольное строковое обозначение; задаётся пользователем;

*номер\_слоя* – номер первичного слоя из файла GDSII, которому присваивается строковое обозначение.

Пример присвоения строковых обозначений «l1», «l2» слоям с номерами «9», «32»:

l1 = 9

l2 = 32

Для объявления производных слоёв необходимо выполнить команду присвоения значения:

*имя\_переменной = операция*

где *имя\_переменной* – произвольное строковое обозначение; задаётся пользователем;

*операция* – операция (набор операций), в результате которой генерируется новый слой.

Пример:

l3 = intersection(l1, l2)

Для дублирования содержимого уже существующего слоя в новый слой допустим вариант:

*имя\_переменной = идентификатор\_слоя*

где *имя\_переменной* – произвольное строковое обозначение; задаётся пользователем;

где *идентификатор\_слоя* – числовой или строковый идентификатор другого слоя.

Пример:

l4 = 32

Для объявления набора слоёв необходимо выполнить команду присвоения значения:

*имя\_переменной = {слой1, слой2, …, слойN}*

где *имя\_переменной* – произвольное строковое обозначение; задаётся пользователем;

*{слой1, слой2, …, слойN}* – заключенный в фигурные скобки список числовых или строковых идентификаторов слоёв (наборов слоев), из которых формируется набор слоев *имя\_переменной*.

Пример объявления набора слоев «l3», состоящего из слоя с номером «9» и слоя «l2»:

l3 = {9, l2}

## Описание функций

### setOutputFile

Выходным файлом работы программы является база данных слоев. Для задания имени выходного файла необходимо вызвать функцию *setOutputFile*. В противном случае программа начнет работу, но не сможет сохранить результаты вычислений и выведет в лог-файл сообщение об ошибке:

Не удалось сохранить слой. База данных не открыта

При использовании в скрипте верификации функции *setOutputFile* более 1 раза программа проигнорирует все её вызовы, кроме первого, и выведет в лог‑файл сообщение об ошибке:

База данных уже открыта

Сигнатура функции:

*setOutputFile("имя\_выходного\_файла.dblite")*

где *имя\_выходного\_файла.dblite* – имя файла, в который будут помещены результаты работы программы. Если файла с указанным именем не существует в директории с исполняемым файлом проекта, он будет создан. Допустимо не указывать расширение файла при вызове функции.

Пример вызова функции:

setOutputFile("example.dblite")

### openGDS

Для выбора файла исходных данных необходимо вызвать функцию *openGDS*.

При указании некорректного файла исходных данных программа выведет в лог‑файл сообщение об ошибке:

Не удалось открыть файл имя*\_*файла для чтения

Сигнатура функции:

*openGDS("путь\_к\_файлу", {слои})*

где *путь\_к\_файлу* – путь к файлу входных данных;

*слои* – заключенный в фигурные скобки список слоёв входного файла, которые необходимы для выполнения текущего скрипта верификации и будут считаны программой из файла GDSII.

Пример вызова функции:

openGDS("/home/user/Documents/test.gds", {9, 32})

### setUnitSizeInMeters

Для явного задания единиц измерения, в которых указываются ограничения пространственно-геометрических операций проверки норм КТО необходимо вызвать функцию *setUnitSizeInMeters*.

Сигнатура функции:

*setUnitSizeInMeters(unit)*

где *unit* – отношение устанавливаемых единиц измерения к метрам в экспоненциальном представлении (например, *unit* = 10е-6 означает, что в качестве единиц измерения задаются микрометры).

Использование функции в скрипте верификации не является обязательным. Единицы измерения ограничений пространственно-геометрических операций проверки норм КТО по умолчанию – микрометры.

### saveLayer

Для размещения слоя в БД выходных данных необходимо вызвать функцию *saveLayer*.

Сигнатура функции:

*saveLayer(слой, “комментарий”)*

где *слой* – числовой или строковый идентификатор слоя, который необходимо разместить в БД выходных данных;

*комментарий* – комментарий, с которым необходимо сохранить слой в БД результатов (используется для интерпретации результатов пользователем).

Пример вызова функции:

saveLayer(l3, "3.7.2. Minimum PWELL width = 4.0 ")

### intersection

Функция выполняет пересечение многоугольников входных слоёв. Результат: слой многоугольников. В результирующий слой попадают пересекающиеся области многоугольников входных слоев.

Сигнатура функции:

*intersection({слой1, слой2, …, слойN}, {слойN+1, слойN+2, …, слойN+M})*

где *{слой1, слой2, …, слойN}, {слойN+1, слойN+2, …, слойN+M}* – заключенные в фигурные скобки списки числовых или строковых идентификаторов слоев (наборов слоев), для которых выполняется операция пересечения;

Пример вызова функции для объединения слоев l1, l2 и l3:

intersection({l1}, {l2, l3})

### subtraction

Функция выполняет вычитание многоугольников входных слоев. Результат: слой многоугольников. Многоугольники результирующего слоя формируются из элементов первого входного слоя путем исключения областей пересечения со вторым входным слоем.

Сигнатура функции:

*subtraction({слой1}, {слой2})*

где *слой1*, *слой2* – числовой или строковый идентификатор слоя (набора слоев), для которых выполняется операция вычитания.

Пример вызова функции для вычитания слоя l1 из слоя l2:

subtraction({l2}, {l1})

### union

Функция выполняет объединение многоугольников входных слоев. Результат: слой многоугольников. Многоугольники результирующего слоя формируются из пересекающихся и соприкасающихся более чем в одной точке многоугольников входных слоёв.

Сигнатура функции:

*union({слой1, слой2, …, слойN}, {слойN+1, слойN+2, …, слойN+M})*

где *{слой1, слой2, …, слойN}, {слойN+1, слойN+2, …, слойN+M}* – заключенные в фигурные скобки списки числовых или строковых идентификаторов слоёв (наборов слоев), для которых выполняется операция объединения.

Пример вызова функции для объединения слоев l1, l2 и l3:

union({l1, l2}, {l3})

### XOR

Функция выполняет логическую операцию XOR для многоугольников входных слоев. Результат: слой многоугольников.

Сигнатура функции:

*XOR({слой1}, {слой2})*

где слой1, слой2 – числовой или строковый идентификатор слоя, для которых выполняется операция XOR.

Пример вызова функции XOR для слоев l1 и l2:

XOR({l1}, {l2})

### size

Функция выполняет изменение длины ребер (увеличение/уменьшение) многоугольников входного слоя на заданную величину. Результат: слой многоугольников.

Сигнатура функции:

*size({слой1}, size\_value)*

где *слой1* – числовой или строковый идентификатор слоя (набора слоев), для которого выполняется операция изменения размера;

*size\_value* – число, на которое необходимо изменить размеры многоугольников слоя *слой1*. Положительное число приведет к увеличению размеров многоугольников на указанную величину, отрицательное – к уменьшению.

Пример вызова функции size для слоя l1:

size({l1}, 7)

### extent

Функция вычисляет минимальный ограничивающий прямоугольник для многоугольников входного слоя. Результат: слой многоугольников, содержащий 1 прямоугольник.

Сигнатура функции:

*extent({слой1})*

где *слой1* – числовой или строковый идентификатор слоя (набора слоев), для которого выполняется операция вычисления минимального ограничивающего прямоугольника.

Пример вызова функции extent для слоя l1:

extent({l1})

### rectangle

Функция выполняет выборку прямоугольников, размеры которых удовлетворяют заданному ограничению, среди многоугольников входного слоя. Результат: слой многоугольников.

Сигнатура функции:

*rectangle({слой1}, ограничение)*

где *слой1* – числовой или строковый идентификатор слоя (набора слоев), для которого выполняется операция выборки прямоугольников;

*ограничение* – ограничение на размеры прямоугольников, которые должны попасть в выборку.

Пример вызова функции rectangle для слоя l1:

rectangle({l1}, “>=4”)

### notRectangle

Функция выполняет выборку многоугольников, не являющихся прямоугольниками, среди многоугольников входного слоя. Результат: слой многоугольников.

Сигнатура функции:

*notRectangle({слой1}, ограничение)*

где *слой1* – числовой или строковый идентификатор слоя (набора слоев), для которого выполняется операция;

*ограничение* – ограничение на размеры прямоугольников, которые должны попасть в выборку.

Пример вызова функции notRectangle для слоя l1:

notRectangle({l1}, “>=4”)

### touch

Функция выполняет выборку многоугольников первого входного слоя, которые удовлетворяют следующим условиям:

* имеют общее ребро или сегмент ребра (соприкасаются) с многоугольниками второго входного слоя;
* не пересекаются (частично или полностью) с многоугольниками второго входного слоя.

Результат: слой многоугольников.

Сигнатура функции:

*touch({слой1, слой2, …, слойN}, {слойN+1, слойN+2, …, слойN+M})*

где *{слой1, слой2, …, слойN}, {слойN+1, слойN+2, …, слойN+M}* – заключенные в фигурные скобки списки числовых или строковых идентификаторов слоёв (наборов слоев), для которых выполняется операция поиска соприкасающихся ребер.

Пример вызова функции touch для слоев l1 и l2:

touch({l1}, {l2})

### notTouch

Функция выполняет выборку многоугольников первого входного слоя, которые не имеют общих ребер или общих сегментов ребер (не соприкасаются) с многоугольниками второго входного слоя или пересекаются (частично или полностью) с многоугольниками второго входного слоя.

Сигнатура функции:

*notTouch({слой1, слой2, …, слойN}, {слойN+1, слойN+2, …, слойN+M})*

где *{слой1, слой2, …, слойN}, {слойN+1, слойN+2, …, слойN+M}* – заключенные в фигурные скобки списки числовых или строковых идентификаторов слоёв (наборов слоев), для которых выполняется операция поиска несоприкасающихся ребер.

Результат: слой многоугольников.

Пример вызова функции notTouch для слоев l1 и l2:

notTouch({l1}, {l2})

### offGrid

Функция выполняет поиск вершин многоугольников входного слоя, которые не попадают в узлы сетки проектирования. Результат: слой ребер, примыкающих к вершинам, не попадающим в узлы сетки проектирования.

Сигнатура функции:

*offGrid({слой1}, шаг)*

где *слой1* – числовой или строковый идентификатор слоя (набора слоев), для которого выполняется операция поиска вершин, не попадающих в узлы сетки проектирования;

*шаг* – шаг сетки проектирования.

Пример вызова функции offGrid для слоя l1:

offGrid({l1}, 50)

### length

Функция выполняет выборку ребер многоугольников входного слоя, которые удовлетворяют заданному ограничению на длину. Результат: слой ребер.

Сигнатура функции:

*length({слой1}, ограничение)*

где *слой1* – числовой или строковый идентификатор слоя (набора слоев), для которого выполняется операция поиска ребер заданной длины;

*ограничение* – ограничение на длину ребер, которые должны попасть в выборку.

Пример вызова функции length для слоя l1:

length({l1}, “>2”)

### notLength

Функция выполняет выборку ребер многоугольников входного слоя, которые не удовлетворяют заданному ограничению на длину. Результат: слой ребер.

Сигнатура функции:

*notLength({слой1}, ограничение)*

где *слой1* – числовой или строковый идентификатор слоя (набора слоев), для которого выполняется операция поиска ребер заданной длины;

*ограничение* – ограничение на длину ребер, которые должны попасть в выборку.

Пример вызова функции notLength для слоя l1:

notLength({l1}, “<5”)

### angle

Функция выполняет выборку ребер многоугольников исходного слоя, которые удовлетворяют заданному ограничению на величину меньшего из углов, образующихся между ребром и осью Х. Результат: слой ребер.

Сигнатура функции:

*angle({слой1}, ограничение)*

где *слой1* – числовой или строковый идентификатор слоя (набора слоев), для которого выполняется операция поиска углов, удовлетворяющих ограничению;

*ограничение* – ограничение на величину углов, которые должны попасть в выборку.

Пример вызова функции angle для слоя l1:

angle({l1}, “>45 <90”)

### coincidentEdge

Функция выполняет поиск ребер или сегментов ребер многоугольников первого входного слоя, которые соприкасаются с ребрами многоугольников второго входного слоя. Результат: слой ребер.

Сигнатура функции:

*coincidentEdge({слой1}, {слой2})*

где *слой1, слой2* – числовой или строковый идентификатор слоя, для которых выполняется операция поиска совпадающих ребер.

Пример вызова функции coincidentEdge для слоев l1 и l2:

coincidentEdge({l1}, {l2})

### notCoincidentEdge

Функция выполняет поиск ребер или сегментов ребер многоугольников первого входного слоя, которые не соприкасаются с ребрами многоугольников второго входного слоя. Результат: слой ребер.

Сигнатура функции:

*notCoincidentEdge({слой1}, {слой2})*

где *слой1, слой2* – числовой или строковый идентификатор слоя, для которых выполняется операция поиска несовпадающих ребер.

Пример вызова функции notCoincidentEdge для слоев l1 и l2:

notCoincidentEdge({l1}, {l2})

### cut

Функция выполняет выборку многоугольников первого входного слоя, которые частично пересекаются с многоугольниками второго входного слоя. Результат: слой многоугольников.

Сигнатура функции:

*cut({слой1}, {слой2})*

где *слой1*, *слой2* – числовой или строковый идентификатор слоя, для которых выполняется операция выборки пересекающихся многоугольников. Результат: слой многоугольников.

Пример вызова функции cut для слоев l1 и l2:

cut({l1}, {l2})

### notCut

Функция выполняет выборку многоугольников первого входного слоя, которые полностью пересекаются или не пересекаются с многоугольниками второго входного слоя. Результат: слой многоугольников.

Сигнатура функции:

*notCut({слой1}, {слой2})*

где *слой1*, *слой2* – числовой или строковый идентификатор слоя, для которых выполняется операция выборки не пересекающихся многоугольников. Результат: слой многоугольников.

Пример вызова функции notCut для слоев l1 и l2:

notCut({l1}, {l2})

### density

Функция выполняет поиск геометрических областей входного слоя, плотность которых удовлетворяет заданному ограничению.

Сигнатура функции:

*density({слой1}, ограничение, {ширина, высота}, имя\_файла)*

где *слой1, слой2* – числовой или строковый идентификатор слоя, для которых выполняется операция;

*ограничение* – ограничение на плотность слоя;

*{ширина, высота}* – ширина и высота окна. Необязательный параметр;

*имя\_файла* – имя файла для записи вычисленных значений плотности в каждом окне. Необязательный параметр

Пример вызова функции density для слоев l1 и l2:

density({l1}, “<0.25”)

density({l1}, “<0.25”, {400, 400})

density({l1}, “<0.25”, “results.txt”)

density({l1}, “<0.25”, {400, 400}, “results.txt”)

### enclosure

Функция выполняет поиск ребер или сегментов ребер многоугольников первого входного слоя, которые охвачены многоугольниками второго входного слоя, и величина охвата при этом удовлетворяет заданному ограничению.

Сигнатура функции:

*enclosure({слой1}, {слой2}, ограничение)*

где *слой1, слой2* – числовой или строковый идентификатор слоя, для которых выполняется операция;

*ограничение* – ограничение на величину охвата.

Пример вызова функции XOR для слоев l1 и l2:

enclosure({l1}, {l2}, “<=3”)

### external

Функция выполняет поиск ребер или сегментов ребер многоугольников первого входного слоя, расстояние от которых (по направлению наружу) до многоугольников второго входного слоя удовлетворяет заданному ограничению.

Сигнатура функции:

*external({слой1}, {слой2}, ограничение)*

где *слой1, слой2* – числовой или строковый идентификатор слоя, для которых выполняется операция;

*ограничение* – ограничение на расстояние от многоугольников *слоя1* до многоугольников *слоя2*.

Пример вызова функции XOR для слоев l1 и l2:

external({l1}, {l2}, “<1”)

### internal

Функция выполняет поиск ребер или сегментов ребер многоугольников первого входного слоя, расстояние от которых (по направлению вовнутрь) до многоугольников второго входного слоя удовлетворяет заданному ограничению.

Сигнатура функции:

*internal({слой1}, {слой2}, ограничение)*

где *слой1, слой2* – числовой или строковый идентификатор слоя, для которых выполняется операция;

*ограничение* – ограничение на расстояние от многоугольников *слоя1* до многоугольников *слоя2*.

Пример вызова функции XOR для слоев l1 и l2:

internal({l1}, {l2}, “>2 <4”)

## **Пример скрипта верификации**

setOutputFile("results.dblite")

openGDS("/path/to/input/data/input.gds")

n\_well = 1

N\_Guard = 2

Locos = 4

n\_plus = 8

maskd = 36

active = subtraction({Locos}, {maskd})

nwell = subtraction({n\_well}, {maskd})

nguard = subtraction({N\_Guard}, {maskd})

nplus = subtraction({n\_plus}, {maskd})

nimpl = union({nguard}, {nplus})

ncont = intersection({cont}, {nsd})

active\_offgrid\_err = offGrid({activ}, 100)

saveLayer(active\_offgrid\_err, “active OFFGRID errors”)

active\_angle\_err\_1 = angle({active}, “>0 <45”)

saveLayer(active\_angle\_err\_1, “active not allowed angles >0 <45”)

active\_angle\_err\_2 = angle({active}, “>45 <90”)

saveLayer(active\_angle\_err\_2, “active not allowed angles >45 <90”)

rule\_1.1\_err = internal({nwell}, {nwell}, “<6”)

saveLayer(nwell\_internal\_err, “Rule 1.1 Minimum NWELL width = 6”)

rule\_1.2\_err = internal({nwell}, {nwell}, “<20”)

saveLayer(nwell\_external\_err, “Rule 1.2 Minimum spacing of NWEL = 20”)

rule\_2.3\_err = enclosure({nguard}, {nwell}, “<2”)

saveLayer(nguard\_enclosure\_of\_nwell\_err, “Rule 2.3 Minimum NGUARD enclosure of NWELL = 2”)

rule\_9.1\_err = notRectangle({cont}, “==3.2 BY ==3.2”)

saveLayer(cont\_width\_err, “Rule 9.1 Fixed CONT width = 3.2 x 3.2”)