rEDActor – кроссплатформенная среда разработки PDK-библиотек для полупроводниковых технологий

А.М. Дацук¹, В.П. Тимошенко²

¹IHP GmbH, г. Франкфурт-на-Одере, Германия, datsuk@ihp-microelectronics.com

²Национальный исследовательский университет «МИЭТ», г. Москва, Россия

Аннотация — В данной работе представлена кроссплатформенная интегрированная среда разработки PDK-библиотек – rEDActor. Эта среда включает в себя редактор исходного кода с расширенными возможностями, такими как автозавершение ключевых слов, свертывание и оптимизация кода, а также позволяет выполнить анализ и отладку программ. rEDActor встроен в различные системы автоматического проектирования, что увеличивает эффективность разработки параметрических ячеек, а также позволяет выполнить конвертацию SKILL-кода в AEL для обеспечения режима взаимодействия программ Cadence Virtuoso и Keysight ADS. rEDActor был успешно использован при разработке PDK-библиотек для кремнево-германиевых 0.13 мкм и 0.25 мкм БикМОП технологических процессов.

Ключевые слова — кроссплатформенная среда разработки, библиотека базовых компонентов (PDK).

I. ВВЕДЕНИЕ

Разработка библиотеки базовых компонентов и функций, или PDK (Process Design Kit), включает множество шагов, таких как разработка параметрических ячеек (PCells), моделей, правил проверки топологических норм (DRC), правил проверки соответствия топологии и принципиальной схемы (LVS), файлов конфигурации и т.д. Сложность разработки PDK с каждым годом возрастает из-за уменьшения топологических норм проектирования [1]. Более того, PDK необходимо разрабатывать для различных поставщиков программного обеспечения, таких как Cadence, Keysight, Synopsys и др.

Для разработки кода программисты PDK используют, как правило, различные текстовые редакторы, такие как Nedit, GNU Emacs, Eclipse, которые настраиваются путем подключения сценариев командной оболочки и пользовательских команд, создаваемых на встроенном макроязыке. Каждый из редакторов имеет свои недостатки. Так, например, в Nedit, несмотря на простоту установки, изучения и использования, отсутствует много возможностей, которые являются стандартом для профессиональных текстовых редакторов (макро- и прямоугольные блоки, базовые средства форматирования). GNU Emacs требует основательного изучения, а для расширения его возможностей придется программировать на языке LISP. Среда Eclipse – это кроссплатформенное

приложение, предлагающее широкий прикладной программный интерфейс, или API (Application Program Interface), для создания различных модулей. Однако данная программа требует продолжительного времени загрузки и использует много системных ресурсов, что значительно затрудняет ее использование для разработки PDK-библиотек.

Целью данной работы является создание кроссплатформенной интегрированной среды разработки ИСП (англ. Integrated Development Environment – IDE) rEDActor, которая значительно ускоряет этапы проектирования PDK за счет автоматического анализа и форматирования исходного кода на основе файлов конфигурации, заданных для конкретного языка программирования. ИСП rEDActor встроена в ведущие системы автоматического проектирования (САПР), такие как Cadence Virtuoso, Keysight ADS, что позволяет максимально увеличить производительность разработчика PDK благодаря тесно связанным компонентам с простыми пользовательскими интерфейсами. В отличие от дискретных программ разработки это позволяет программисту совершать меньшее количество действий при переходе от режима разработки, например, PCell к написанию правил проверки топологических норм проектирования. ИСП rEDActor позволяет конвертировать SKILL-код [2] в AEL [3], что значительно повышает эффективность разработки PDK-библиотек для САПР Keysight ADS.

II. РАСШИРЕННЫЕ ВОЗМОЖНОСТИ ИСП rEDActor ДЛЯ РАЗРАБОТКИ PDK

Для увеличения производительности разработки PDK в ИСП rEDActor были реализованы следующие функции.

A. Автоматическое форматирование исходного кода

При написании исходного кода важно соблюдать правила форматирования, поскольку это существенно упрощает его восприятие, что облегчает сопровождение кода в дальнейшем. В некоторых языках программирования, например, Python, форматирование необходимо для распознавания структуры кода компилятором (off-side rules) [4]. Как известно, язык Python широко применяется для разработки топологии параметрических ячеек (PyCells) [5]. В ИСП rEDActor были реализованы функции автоматического разбора

структуры кода, состоящей, например, из функций, циклов, if-блоков и т.д., и форматирования исходного кода в зависимости от настроек, заданных для данного языка программирования. Пользователь может использовать предустановленный стиль кода или определить свой.

Б. Режим свертки кода

Разработан режим свертки кода (code folding), позволяющий скрывать и показывать секции кода в зависимости от области видимости. Это позволяет пользователю управлять большими частями кода, отображая только те части, непосредственно с которыми ведется работа в данный момент времени. Идентификаторы свертки по умолчанию заданы для каждого языка программирования, поддерживаемого ИСР rEDActor, однако пользователь может их изменить или добавить новые в настройках редактора. Пример свертки AEL-кода представлен на рис. 1. Из рисунка видно, что код свернут для функции *texedaGetColor* (код скрыт с 35 по 42 строку) и цикла *for* (код скрыт с 50 по 55 строку).

Рис. 1. Автоматическая свертка AEL-кода

В. Автозавершение набираемых ключевых слов и переменных

Автозавершение набираемых слов – это способность программы предсказывать набираемые программистом ключевые слова, функции и переменные. Это позволяет разработчику значительно сократить как время набора кода, так и минимизировать время поиска необходимых ключевых слов в документации. При автоматическом завершении слова исключается возможность сделать орфографическую ошибку. В ИСР rEDActor предложен режим автозавершения набираемых слов, функций и переменных для языков программирования: SKILL, AEL, Python, Perl, Ruby, TCL, Cadence Assura и PVS, Mentor SVRF и др. Однако пользователь не ограничен только набором ключевых слов и функций по умолчанию и может самостоятельно добавлять новые или удалять существующие с помощью настроек ИСР rEDActor. Пример работы автозавершения SKILL-функций представлен на рис. 2. В ИСР rEDActor реализована поддержка проектов, в этом случае редактор автоматически находит все пользовательские функции и переменные данного проекта и добавляет их в словарь для автозавершения.

Рис. 2. Пример работы автозавершения SKILL-функций

Г. Синтаксическая проверка кода

В ИСР rEDActor реализован интерфейс для подключения сторонних программ проверки синтаксических ошибок, необходимых для языков программирования, таких как SKILL, AEL, Mentor SVRF и др. Пример работы программы синтаксического анализатора (skill [2]) для проверки SKILL-кода в ИСР rEDActor продемонстрирован на рис. 3.

Рис. 3. Проверка синтаксических ошибок SKILL-кода с помощью программы skill, встроенной в ИСР rEDActor

Данная функциональность позволяет программисту выполнить проверку синтаксических ошибок в любой момент времени, а также произвести отладку программы непосредственно в ИСР rEDActor. Редактор также имеет встроенный синтаксический анализатор формата XML для поиска ошибок. Данная функциональность была разработана для удобства программиста, поскольку с помощью XML-формата реализована поддержка конфигурационных файлов для

различных языков программирования, поддерживаемых ИСР gEDAActor.

Д. Автоинициализация глобальных переменных в SKILL

Глобальные переменные в SKILL имеют статическую область видимости: создаются при запуске программы и удаляются при ее завершении. Глобальные переменные можно использовать и изменять на протяжении всей работы программы Cadence Virtuoso, что может повлиять на работу блоков, в которых данная переменная используется. Также глобальные переменные делают программу менее модульной и гибкой. При написании SKILL-кода программисты сначала реализуют функционал блока без инициализации переменных, поскольку заранее неизвестно, какие переменные будут использованы при разработке функции. После того как функция реализована, происходит поиск глобальных переменных, которые вручную заносятся в блоки *let* или *prog*, чтобы скрыть их область видимости. В ИСР gEDAActor реализован алгоритм, который позволяет программисту автоматически находить глобальные переменные внутри функций и выполнять их инициализацию в блоках *let* и *prog*, сортируя переменные в алфавитном порядке. Данная функциональность сокращает время разработки программы, поскольку позволяет не задумываться о скрытии локальных переменных внутри области видимости функции во время ее написания. Инициализация глобальных переменных в данном случае производится на завершающем этапе, когда функционал блока реализован.

III. АВТОМАТИЧЕСКАЯ КОНВЕРТАЦИЯ SKILL-КОДА В AEL

Компания Keysight в среде разработки интегральных схем (ИС) ADS реализовала режим взаимодействия (interoperability) с Cadence Virtuoso. Режим позволяет пользователю открывать и редактировать в ADS электрические схемы и топологии, созданные в Cadence Virtuoso, а также проводить различные виды моделирования, что дает возможность разработчикам использовать преимущества обеих платформ при проектировании ИС [6].

Режим взаимодействия значительно упрощает разработку PDK под ADS, поскольку поддерживает формат базы данных OpenAccess и синтаксический анализатор LISP, необходимый для чтения процедурных вызовов (callbacks). Это позволяет производить чтение, редактирование и моделирование электрических схем в ADS с использованием PDK, созданной для среды Cadence Virtuoso. Однако PCells для топологического проектирования в Cadence Virtuoso создаются с помощью языка программирования SKILL. Поддержка этого языка программой Keysight ADS невозможна, поскольку SKILL является собственностью компании Cadence Design Systems.

Для автоматизации режима взаимодействия для отображения топологической схемы в ИСР gEDAActor был разработан конвертер, позволяющий транслировать язык программирования SKILL в AEL. После конвертации пользователю необходимо проверить правильность полученного кода, поскольку именно программист является ответственным за работу требуемой функциональности.

На рис. 4 показан пример части SKILL-кода для емкости (cmim), реализованного для кремниевогерманиевого 0.13 мкм БиКМОП технологического процесса. Топологическое представление данного прибора в среде разработки Cadence Virtuoso представлено на рис. 5. Поскольку данный код не может быть использован в программе Keysight ADS напрямую, его необходимо реализовать на языке AEL для обеспечения режима взаимодействия.

```
when( metallization=="M4" || metallization=="M4M5"
      bmlayer="Metal3")

; check minvalues
when(w < wmin-epsilon
      w=wmin;
      hiGetAttention()
      printf("Width < %L\n" wmin)
)
when(l < lmin-epsilon
      l=lmin;
      hiGetAttention()
      printf("Height < %L\n" lmin)
)

; how many vias?
xanz=fix((w-cont_over-cont_over+cont_dist)/(cont_size+cont_dist)+epsilon)
; width for vias
wl=xanz*(cont_size+cont_dist)-cont_dist+cont_over+cont_over
; offset to first via
xoffset=(w-wl)/2
xoffset=GridFix(xoffset)

yanz=fix((l-cont_over-cont_over+cont_dist)/(cont_size+cont_dist)+epsilon)
ll=yanz*(cont_size+cont_dist)-cont_dist+cont_over+cont_over
yoffset=(l-ll)/2
yoffset=GridFix(yoffset)

ycont_cnt=cont_over+yoffset
; draw vias
while(ycont_cnt+cont_size+cont_over <= l+epsilon
      xcont_cnt=cont_over+xoffset
      while(xcont_cnt+cont_size+cont_over <= w+epsilon
            dbCreateRect(pcCellView vialayer list(
                          list(xcont_cnt ycont_cnt)
                          list(xcont_cnt+cont_size ycont_cnt+cont_size))
            )
            xcont_cnt=xcont_cnt+cont_size+cont_dist
            )
            ycont_cnt=ycont_cnt+cont_size+cont_dist
            )
```

Рис. 4. Пример части реализации SKILL-кода для емкости (cmim)

С помощью ИСР gEDAActor вышеупомянутый SKILL-код был сконвертирован в AEL-код (рис. 6), который без внесения изменений программистом PDK был загружен в среду разработки Keysight ADS (рис. 7). Данный подход позволил значительно сократить время разработки топологического представления данного устройства, а в дальнейшем данный конвертер был применен для автоматической конвертации всех устройств, реализованных с помощью языка программирования SKILL для кремниевогерманиевых 0.13 мкм и 0.25 мкм БиКМОП технологических процессов.

Рис. 5. Топологическое представление емкости (cmim) в среде Cadence Virtuoso

```
if(metallization == "M4" || metallization == "M4M5") {
    bmlayer = "Metal3";
}
if(w < wmin-epsilon) {
    w = wmin;
    hiGetAttention();
    de_info(sprintf("Width < %L\n", wmin));
}
if(l < lmin-epsilon) {
    l = lmin;
    hiGetAttention();
    de_info(sprintf("Height < %L\n", lmin));
}
xanz = fix((w-cont_over-cont_over+cont_dist)/(cont_size+cont_dist)+epsilon);
wl = xanz*(cont_size+cont_dist)-cont_dist+cont_over+cont_over;

xoffset = (w-wl)/2;
xoffset = GridFix(xoffset);

yanz = fix((l-cont_over-cont_over+cont_dist)/(cont_size+cont_dist)+epsilon);
ll = yanz*(cont_size+cont_dist)-cont_dist+cont_over+cont_over;
yoffset = (l-ll)/2;
yoffset = GridFix(yoffset);

ycont_cnt = cont_over+yoffset;

while(ycont_cnt+cont_size+cont_over <= l+epsilon) {
    xcont_cnt = cont_over+xoffset;
    while(xcont_cnt+cont_size+cont_over <= w+epsilon) {
        dbCreateRect(pcCellView, vialayer,
            list(xcont_cnt, ycont_cnt,
                xcont_cnt+cont_size, ycont_cnt+cont_size));
        xcont_cnt = xcont_cnt+cont_size+cont_dist;
    }
    ycont_cnt = ycont_cnt+cont_size+cont_dist;
}
```

Рис. 6. Пример части сконвертированного AEL-кода для емкости (cmim)

Рис. 7. Проверка синтаксических ошибок SKILL-кода с помощью программы skill, встроенной в ИСР rEDActor

Конвертер работает на основе программного расширения (plug-in). Данное программное расширение представляет собой файл с описанием основных SKILL-функций, реализованных на языке AEL, для работы с топологией. Пример файла plug-in представлен на рис. 8. Данный пример иллюстрирует реализацию SKILL-функций `dbCreateRect` и `dbCreatePolygon` на языке AEL. Программное расширение, содержащее основные функции для работы с геометрией для языка AEL, поставляется вместе с ИСР rEDActor. При необходимости пользователь может самостоятельно расширить функционал программного расширения, добавив требуемые SKILL-функции, реализованные на AEL.

ИСР rEDActor был внедрен в САПР Cadence Virtuoso и Keysight ADS, поэтому выполнение и отладка кода возможны напрямую из редактора. Таким способом пользователь может разрабатывать параметрическую ячейку в Cadence Virtuoso, а также осуществлять контроль выполнения SKILL-кода, не прибегая к постоянной загрузке файла в командном окне Cadence (Command Interpreter Window). При необходимости можно воспользоваться отладчиком SKILL, устанавливая точки останова (break point), и выполнять анализ кода в ИСР rEDActor. После установки точки останова код будет выполняться до тех пор, пока не достигнет строки, где установлена данная точка, после чего управление будет передано SKILL-отладчику. При этом пользователь может выводить информацию о локальных переменных процедуры, глобальных переменных, а также проанализировать стек вызовов, предшествующих вызову данной процедуры. Дальнейший анализ кода может быть выполнен путем трассировки (пошаговой отладки). Пример, иллюстрирующий отладку SKILL-программы с помощью ИСР rEDActor в САПР Cadence Virtuoso, показан на рис. 9.

Рис. 8. Пример реализации SKILL-функций на языке AEL

Рис. 9. Пример отладки SKILL-кода в САПР Cadence Virtuoso, иллюстрирующий вывод локальных и глобальных значений переменных в точке останова 899

После того как разработка параметрической ячейки в САПР Cadence Virtuoso завершена, пользователь может сконвертировать написанный и отлаженный код в AEL. ИСР rEDActor может быть запущен из САПР Keysight ADS для окончательной отладки AEL-кода и генерации топологического представления параметрической ячейки.

Конвертация параметрических ячеек значительно упростила разработку PDK-библиотек, поддерживающих режим взаимодействия между САПР Cadence Virtuoso и Keysight ADS, который позволяет проектировать ИС одновременно в обеих САПР. Это позволяет разработчикам ИС использовать функциональные возможности обеих программ, как, например, электротепловое моделирование с помощью программы Keysight Electrothermal [7], которое отсутствует в САПР Cadence Virtuoso.

IV. ЗАКЛЮЧЕНИЕ

В работе представлена реализация кроссплатформенной интегрированной среды разработки rEDActor, которая увеличивает эффективность разработки PDK-библиотек. Продemonстрировано, что ИСР rEDActor обладает как необходимыми, так и расширенными функциональными возможностями, требуемыми современными программистами PDK-библиотек. ИСР rEDActor интегрирован в Cadence Virtuoso и Keysight ADS, что позволяет разработчику PDK-библиотек выполнять анализ и отладку программ непосредственно в вышеперечисленных САПР. Редактор успешно применен для разработки PDK-библиотек для

кремниво-германиевых 0.13 мкм и 0.25 мкм БИМОП технологических процессов.

За счет разработанного конвертера ИСР rEDActor позволил выполнить трансляцию всех топологических представлений параметрических ячеек, разработанных с помощью языка SKILL в AEL. Необходимо отметить, что некоторые параметрические ячейки (примерно 10 % от общего числа) откорректированы вручную. Тем не менее временные затраты, требуемые для корректировки конечного AEL-кода, несравнимы с затратами, необходимыми для разработки аналогичной функциональности с нуля.

Применение ИСР rEDActor не ограничивается разработкой только параметрических ячеек, а позволяет также программировать правила DRC и LVS для различных САПР (Cadence Assura, PVS), Mentor (Calibre), PolyEDA (Power), TexEDA (LayVer). ИСР rEDActor использован для разработки технологического стека, требуемого для электротеплового моделирования, а также написания алгоритмов оптимизаций значений теплопроводностей материалов [8].

ЛИТЕРАТУРА

- [1] Y. Li, M. Li, and W. Wong, "A complete process design kit verification flow and platform for 28 nm technology and beyond," in Proc. IEEE Int. Conf. Solid-State and Integrated Circ. Technology, Oct. 2012, pp. 1-4.
- [2] T. J. Barnes, "SKILL: A CAD system extension language," in 27th ACM/IEEE Design Automation Conference. IEEE, 1990, pp. 266-271.
- [3] Keysight Technologies. (2020) Advanced Design System Documentation - AEL. [Online]. Available: http://literature.cdn.keysight.com/litweb/pdf/ads2008/ael/-ads2008/Introduction_to_AEL.html
- [4] P. J. Landin, "The next 700 programming languages," Communications of the ACM, vol. 9, no. 3, pp. 157-166, 1966.
- [5] R. Goldman, K. Bartleson, T. Wood, K. Kranen et al., "32/28nm educational design kit: Capabilities, deployment and future," in 2013 IEEE Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics (PrimeAsia). IEEE, 2013, pp. 284-288.
- [6] Keysight. (2019) Silicon RFIC interoperability w/Virtuoso element. [Online]. Available: <https://www.keysight.com/en/-pd-2370313-pn-W2319EP/silicon-rfic-interoperability-w-virtuoso-element>
- [7] W2349EP. (2017) Keysight ADS Electro-thermal simulator. [Online]. Available: <http://www.keysight.com/en/-pd-2145993-pn-W2349EP/ads-electro-thermal-simulator-element?cc=DE&lc=ger>
- [8] F. Korndörfer, A. Datsuk, and M. Kaynak, "Layout based electro-thermal simulation setup," in Proc. IEEE 18th Topical Meeting Silicon Monolithic Integrated Circuits in RF Systems (SiRF), Jan. 2018, pp. 68-70.

rEDActor – A PDK Cross-platform Integrated Development Environment for Semiconductor Technologies

A. Datsuk¹, V. Timoshenkov²

¹IHP GmbH, Im Technologiepark 25, 15236 Frankfurt (Oder), Germany,
datsuk@ihp-microelectronics.com.

²National Research University of Electronic Technology,
Shokin Square, Zelenograd, Moscow, Russia, 124498.

Abstract – The paper presents a cross-platform integrated development environment (IDE) for PDKs. A PDK development includes many steps such as coding, building, releasing, configuring, setting up runtime environment, testing, troubleshooting and final deployment. The modern editors and IDEs are largely non-optimal for a few vital reasons. Firstly, they do not provide such functionalities as word autocompletion, code folding and code refactoring integrated into one editor. Secondly, these editors do not have PDK specific features by default. These features have to be implemented additionally by PDK developers which costs time and requires appropriate programmer skills. The IDE rEDActor has been developed to address these problems and challenges and it includes a code editor which supports various languages required for PDK development. rEDActor allows performing a syntax check verification and source code debug by plugging in a third-party tools through an appropriate embedded interface. The IDE has been incorporated into the leading-edge software tools increasing efficiency of PDK development significantly. This functionality gives PDK developer an opportunity to implement and debug the code within both Cadence Virtuoso and Keysight ADS environments. The presented IDE is not limited to parametrical cells (PCells) development only and it is applicable for programming any PDK routines and rule deck scripts (DRC, LVS) for different vendors: Mentor (Calibre), Cadence Assura and PVS, Polyteda (Power), TexEDA (LayVer) as well. The IDE rEDActor has been enhanced to convert SKILL code for layout PCells into AEL one. The implemented functionality allows significantly reduce the effort of layout PCells development for interoperable PDKs which enable users to edit and simulate designs created in Cadence Design System's Virtuoso software within Keysight ADS framework and visa versa. The IDE rEDActor has been successfully applied for development of PDKs based on 0.13 um and 0.25 um SiGe BiCMOS technologies. The result was the significant reduction of programming effort, and consequently, highly reliable and robust PDK releases.

Keywords – rEDActor, Integrated Development Environment (IDE), Process Design Kit (PDK), Code Editor, SKILL, AEL, Cadence Virtuoso, Mentor Calibre, Keysight ADS.

REFERENCES

- [1] Y. Li, M. Li, and W. Wong, "A complete process design kit verification flow and platform for 28 nm technology and beyond," in Proc. IEEE Int. Conf. Solid-State and Integrated Circ. Technology, Oct. 2012, pp. 1-4.
- [2] T. J. Barnes, "SKILL: A CAD system extension language," in 27th ACM/IEEE Design Automation Conference. IEEE, 1990, pp. 266-271.
- [3] Keysight Technologies. (2020) Advanced Design System Documentation - AEL. [Online]. Available: http://literature.cdn.keysight.com/litweb/pdf/ads2008/ael/-ads2008/Introduction_to_AEL.html
- [4] P. J. Landin, "The next 700 programming languages," Communications of the ACM, vol. 9, no. 3, pp. 157-166, 1966.
- [5] R. Goldman, K. Bartleson, T. Wood, K. Kranen et al., "32/28nm educational design kit: Capabilities, deployment and future," in 2013 IEEE Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics (PrimeAsia). IEEE, 2013, pp. 284-288.
- [6] Keysight. (2019) Silicon RFIC interoperability w/Virtuoso element. [Online]. Available: <https://www.keysight.com/en/-pd-2370313-pn-W2319EP/silicon-rfic-interoperability-w-virtuoso-element>
- [7] W2349EP. (2017) Keysight ADS Electro-thermal simulator. [Online]. Available: <http://www.keysight.com/en/-pd-2145993-pn-W2349EP/ads-electro-thermal-simulator-element?cc=DE&lc=ger>
- [8] F. Korndörfer, A. Datsuk, and M. Kaynak, "Layout based electro-thermal simulation setup," in Proc. IEEE 18th Topical Meeting Silicon Monolithic Integrated Circuits in RF Systems (SiRF), Jan. 2018, pp. 68-70.