СОДЕРЖАНИЕ

▼8

▼9

▼ 10

▼ 11

Введение

Автоматизация испытаний объектов или управления процессами в промышленности, равно как и автоматизация экспериментальных исследований, требует решения ряда задач, связанных с приобретением типового или с разработкой и изготовлением нестандартного оборудования, разработкой программ нижнего уровня для управления этим оборудованием. Кроме того, должно быть спроектировано и программное обеспечение верхнего уровня, предназначенное для объединения всех программных и технических компонентов в единую систему, которую может контролировать и которой должен управлять человек.

Решением каждой из этих достаточно специфичных и сложных задач занимаются специалисты соответствующего профиля – разработчики аналоговых или цифровых устройств, разработчики прикладных программ нижнего уровня или программисты системного программного обеспечения. Все они используют различные инструментальные средства, овладение которыми требует специальной подготовки, а для получения с их помощью качественных результатов необходимы определенный уровень квалификации и опыт. Нужны также специалисты, способные грамотно сформулировать задание для каждого из разработчиков, с тем чтобы минимизировать потери времени на интеграцию полученных результатов.

Заметно упрощается решение большинства этих задач, когда в качестве среды проектирования применяют все более широко распространяющуюся систему графического программирования LabVIEW, разработанную корпорацией National Instruments. Главные и принципиальные отличия LabVIEW от классических языков программирования – графический способ проектирования программ и потоковое их исполнение – позволяют реализовать совершенно новые возможности, а в ряде случаев получить качественно лучшие результаты. При этом удается существенно облегчить сам процесс разработки новых программно-технических комплексов.

LabVIEW – Laboratory Virtual Instrument Engineering Workbench – среда проектирования виртуальных измерительных приборов для инженеров – исключительно проста в освоении, не требует квалификации профессионального программиста для комплексного решения всей совокупности упомянутых задач, естественным образом реализуя интеграцию технических средств и программного обеспечения разного уровня.

Линейка предлагаемых вместе с базовой системой LabVIEW проблемно ориентированных библиотек функций высокого уровня, модулей, наборов и комплектов разработчика недавно пополнилась модулем программирования программируемых логических интегральных схем (ПЛИС), а точнее одной из самых сложных разновидностей ПЛИС – FPGA (Field Programmable Gate Array – матрицы логических вентилей, программируемых в условиях эксплуатации). Этот модуль – LabVIEW FPGA – предназначен для разработки систем, каналы ввода-вывода которых, а также встроенные специализированные устройства обработки данных выполняются реконфигурируемыми, причем алгоритм их функционирования реализуется на аппаратном уровне.

С появлением LabVIEW FPGA у инженеров появилась уникальная возможность – в одной среде и с помощью одних и тех же инструментов проектировать программное обеспечение, предварительно сконструировав необходимые электронные устройства. Эта концепция, названная специалистами National Instruments технологией реконфигурируемого ввода-вывода, позволяет на основе соответствующих технических средств быстро и эффективно создавать самые разнообразные и гибкие измерительные, тестовые и управляющие приложения с весьма высокими характеристиками по производительности, необходимыми при решении многих задач сбора и обработки данных, контроля и управления.

Главная цель настоящей книги – дать представление о принципах и технике проектирования реконфигурируемых систем. При этом для тех, кто незнаком с возможностями ПЛИС и средой графического программирования LabVIEW, автор счел необходимым привести некоторый минимальный объем сведений о программируемых логических интегральных схемах, об организации среды проектирования LabVIEW и основных приемах работы в этой среде.

Материал изложен в следующей последовательности.

В первой главе рассматриваются основные разновидности архитектур программируемых логических интегральных схем, сравниваются их характеристики, дается представление о средствах проектирования ПЛИС, приводятся некоторые примеры применения этих компонентов в специализированных устройствах обработки информации.

Принципы построения и функциональные возможности средств получения информации и выдачи управляющих воздействий – модулей ввода-вывода, предназначенных как для измерения, так и для генерации аналоговых электрических величин, сбора и формирования цифровых сигналов, – обсуждаются в главе 2. Здесь рассматриваются типовые, встраиваемые в персональные и промышленные компьютеры многофункциональные устройства, выполненные на основе традиционных схем, а также модули R-серии, реализующие новую концепцию реконфигурируемого ввода-вывода – Reconfigurable Input/Output (RIO). В этих устройствах, выпускаемых корпорацией National Instruments, используются ПЛИС, программируемые разработчиком на этапе создания прикладных систем, что обеспечивает возможность определять состав, функциональность и технические характеристики каналов ввода-вывода, а также устройств обработки данных не при изготовлении модулей на заводе, а в процессе их интеграции в системы.

В главе 3 рассматриваются концепция виртуальных измерительных приборов и особенности графического языка программирования G, на котором базируется система LabVIEW, приводится краткий обзор программного обеспечения National Instruments, подробнее изложены вопросы конфигурирования и тестирования технических и программных средств с помощью утилиты Measurement and Automation eXplorer (MAX).

Вопросам организации среды проектирования LabVIEW, методике разработки и отладки программного обеспечения посвящены главы 4 и 5. Здесь дается информация об инструментах, предоставляемых разработчику графических программ, назначении и составе основных палитр объектов, используемых при «рисовании» программ, на конкретных примерах показываются приемы проектирования. Необходимо отметить, что затронуты далеко не все возможности LabVIEW, в частности поверхностно или совсем не рассмотрены средства визуализации данных в графических форматах, функции коммуникаций, управления приложениями и ряд других. При этом автор полагает, что, если потребуется, читатели смогут найти соответствующую информацию в книгах по общим принципам программирования в LabVIEW, их издано в России уже около десятка.

В то же время литературы о LabVIEW FPGA и системах с реконфигурируемыми каналами ввода-вывода, насколько известно автору, пока еще мало не только на русском языке, но и на языке разработчиков этого нового и, безусловно, перспективного направления системотехники. Можно добавить, что основными источниками информации, использованными при написании этой книги, послужила справочная система LabVIEW FPGA, а также ряд опубликованных в Интернете материалов, авторами которых являются сами разработчики LabVIEW FPGA и разработчики технических средств, созданных для реализации реконфигурируемых систем. Поэтому приведенный в конце книги список цитируемых источников содержит в основном ссылки на электронные документы, большинство из которых поставляется вместе с программным обеспечением или аппаратурой или опубликовано на сайтах корпорации National Instruments.

Основным содержанием главы 6 является описание особенностей архитектур реконфигурируемых систем, технологии их проектирования и тех специальных дополнительных средств LabVIEW, с помощью которых эти системы создаются. Затем, в главе 7, детально рассматривается процесс разработки FPGA VI – программ, код которых реализует на аппаратном уровне в FPGA типовые задач измерения и генерации сигналов. Особое внимание уделяется возникающим при этом проблемам синхронизации и взаимодействия параллельно исполняемых структур, корректного разрешения коллизий при использовании разделяемых ресурсов. Изложены также способы оптимизации кода FPGA VI, позволяющие повысить быстродействие приложений или/и уменьшить объем расходуемых логических элементов FPGA.

Вопросам разработки программ, исполняемых на главном компьютере и предназначенных для управления функционированием FPGA VI, посвящена глава 8. Сравниваются различные способы организации обмена данными между управляющей программой – Host VI и FPGA VI, обсуждаются их достоинства и недостатки, даются рекомендации по их применению.

В главе 9 приводится краткий обзор нового семейства модулей ввода-вывода Compact RIO (cRIO), описываются основные этапы программирования систем, скомпонованных из встраиваемых в компьютеры модулей R-серии, функциональные возможности которых расширены, а количество каналов увеличено путем подключения модулей cRIO.

Архитектура автономных (встраиваемых) и распределенных систем реального времени с реконфигурируемыми каналами ввода-вывода, а также особенности разработки программного обеспечения подобных систем изложены в главе 10.

Наконец, в 11-й главе рассматриваются примеры использования технологии реконфигурируемого ввода-вывода при решении ряда практически важных прикладных задач: цифровой фильтрации результатов измерений в темпе их получения, реализации различных, в том числе специализированных, интерфейсов и протоколов, тестирования и исследования современных систем связи, работающих в диапазоне СВЧ, разработки прототипов сложных систем управления.

Представляется, что информации, предлагаемой читателю, будет достаточно для ускоренного освоения новой перспективной технологии проектирования информационно-измерительных и управляющих систем.

Книга может быть рекомендована инженерам, начинающим применять LabVIEW FPGA в своей практической деятельности, а в качестве учебного пособия – студентам соответствующих специальностей.

1 Программируемые логические интегральные схемы

По мере развития микроэлектронных технологий и возрастания степени интеграции элементов на кристалле все очевиднее становилось противоречие между потребностями создавать компактные специализированные устройства из наименьшего количества компонентов и нерентабельностью расширения номенклатуры выпускаемых массовым тиражом типовых логических элементов и блоков. Действительно, уже с конца 60-х гг. прошлого века стало возможным на одном кристалле размещать десятки и сотни логических вентилей, но чем насыщеннее становился каждый новый кристалл, тем больше возрастала его специализация и тем меньше становилась относительная доля объема его выпуска.

Появление универсальных логических устройств, микропроцессоров, конечная функция которых определялась загружаемой в них программой, казалось, сблизило возможности изготовителей микросхем, владеющих высокотехнологичными средствами производства, с одной стороны, потребности и возможности разработчиков прикладных устройств и систем – с другой. Однако инженеры попрежнему вынуждены были для решения своих задач использовать компоненты высокой (по меркам того времени) степени интеграции – в качестве основного функционального блока (например, процессора) и десятки логических элементов малой степени интеграции – для того, чтобы состыковать между собой микросхемы разных классов и типов, реализовать на аппаратном уровне специфические для каждого конкретного случая узлы.

Например, контроллер выполнялся на основе 4- или 8-разрядного микропроцессора и нескольких микросхем памяти, содержащих тысячи логических вентилей, и до десятка и более интегральных схем низкой степени интеграции – дешифраторов, регистров, простейших логических элементов, с помощью которых реализовывалась системная шина, каналы ввода-вывода и т. п. При этом для изготовления такого контроллера требовалась печатная плата с площадью, достаточной для установки нескольких микросхем повышенной степени интеграции (микропроцессор, память) и десятка, а то и более, микросхем малой степени интеграции, размещения всех необходимых печатных проводников. Сборка относительно несложных логических узлов контроллера проводилась с помощью паяльника, в то время как существенно более сложные микропроцессор и блоки памяти

поступали «в собранном виде», готовые. Очевидно, что и стоимость монтажа оказывалась непропорциональной сложности узлов, а надежность устройства в целом определялась надежностью элементов малой степени интеграции, количеством их выводов и соединительных проводников на печатной плате.

Таким образом, разработчики цифровой аппаратуры, получив возможность использовать достаточно мощные и универсальные процессорные компоненты, вынуждены были по-прежнему применять простые и тоже универсальные элементарные логические схемы, собирая их в специализированные блоки, без которых невозможно было спроектировать и изготовить устройство, решающее специальную задачу измерений и обработки информации, управления или тестирования.

Назревала необходимость усовершенствования технологии проектирования и производства конечных изделий, приближения ее к технологии проектирования и производства микроэлектронных компонентов. Безусловно, передовой являлось массовое производство микросхем, где связи между элементами в кристалле создавались с помощью масок. Однако использовать эту технологию в многочисленных лабораториях разработчиков цифровых устройств невозможно, так как изготовление масок – весьма трудоемкий и дорогой процесс. Необходимо было придумать более доступный механизм произвольного соединения элементов кристалла между собой и с внешними выводами микросхемы, а разместить на кристалле необходимый набор типовых логических элементов (десятки или сотни) к тому времени проблемой уже не являлось.

Другими словами, разработчику надо было предоставить возможность более простыми и в то же время более эффективными средствами **создавать собственную микросхему**, которая могла бы заменить горсть универсальных микросхем малой степени интеграции, объединяемых в устройстве в специализированный узел. Такая возможность была реализована в программируемых логических интегральных структурах – ПЛИС (PLD – Programmable Logic Device). На кристалле ПЛИС размещались простые логические вентили различных типов и объединяющие их регулярные шины проводников. Подобная микросхема представляла собой некоторую универсальную заготовку, в которой разработчик доступными средствами может удалить ненужные связи между вентилями, изменяя тем самым логическую функцию ПЛИС и, по существу, создавая уникальное, необходимое только ему логическое устройство, сохраняя почти все качества интегрального исполнения.

1.1. Простые программируемые логические устройства – SPLD

Первые кристаллы, структуру и функции которых мог определять пользователь, – программируемые постоянные запоминающие устройства ППЗУ (PROM – Programmable Read Only Memory). ППЗУ состоит из двух матриц логических элементов – матрицы элементов «И» и матрицы элементов «ИЛИ». Входы элементов «И» соединяются с внешними входами микросхемы через повторитель, или ин-

вертор, а входы элементов «ИЛИ» – с выходами элементов «И» [1]. Таким образом, на выходах элементов «ИЛИ», являющихся внешними выходами микросхемы, формируется логическая сумма произведений входных переменных.

В показанной на рис. 1-1 схеме переменные, подаваемые на входы адреса, в дешифраторах строк образуют все возможные конъюнкции – функции «И» от всех входных переменных. Объединением выходов конъюнкторов в столбцах матрицы элементов памяти реализуются логические функции «ИЛИ». Именно отключением определенных строк от столбцов определяются конечные логические функции ППЗУ.

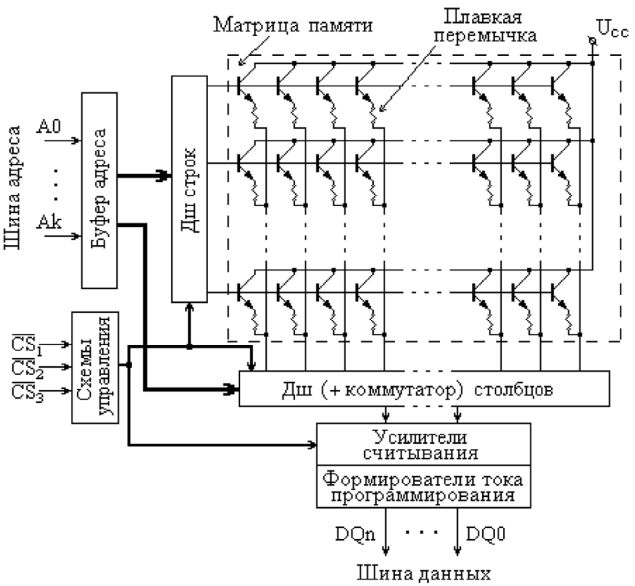


Рис. 1-1. Структура ППЗУ

От изготовителя ППЗУ поступало «чистым» – подключены все входы всех элементов «И» и «ИЛИ», однако пользователь мог удалять ненужные соединения, адресно разрушая их в специальном режиме. В качестве разрушаемых соединений (перемычек) использовались выжигаемые током проводники или пробиваемые p-n-переходы (диоды), сохраняемые же связи и являлись собственно запоминающими элементами и определяли логическую функцию для каждого выхода ППЗУ. Следует отметить, что использовались и иные способы программирования ППЗУ – не удалением ненужных из предварительно созданных межсоединений, а наоборот – созданием нужных соединений. Однако подобные технологические нюансы для разработчика прикладных устройств непринципиальны.

Различают ППЗУ двух основных типов:

- в микросхемах, получивших название PROM, изменять можно только подключения входов элементов «ИЛИ», матрица связей элементов «И» фиксирована (пример на рис. 1-1);
- в микросхемах типа PAL/GAL (Programmable/Generic Array Logic) – программируются связи элементов «И», а соединения элементов «ИЛИ» изменению не подлежат.

Набор из N произвольных логических функций от k входных переменных может быть реализован в одной микросхеме ППЗУ с N выходами емкостью 2^k вентилей, достаточно записать эти функции в совершенной дизъюнктивной нормальной форме и ввести их таблицы истинности в программатор.

Очевидно, что в программируемых ПЗУ площадь кристалла расходуется неэкономно – для всех возможных конъюнкций и дизъюнкций входных и промежуточных переменных должны быть зарезервированы входы элементов «И» и «ИЛИ», значительная часть которых при программировании отключается.

Более рационально логические функции реализуются на базе ПЛМ – программируемых логических матриц (PLA – Programmable Logic Array) [2, 3]. В ПЛМ могут быть запрограммированы обе матрицы логических элементов – «И» и «ИЛИ», при этом количество входов конъюнкторов и дизъюнкторов может быть уменьшено без существенных потерь сложности реализуемых логических функций (рис. 1-2).

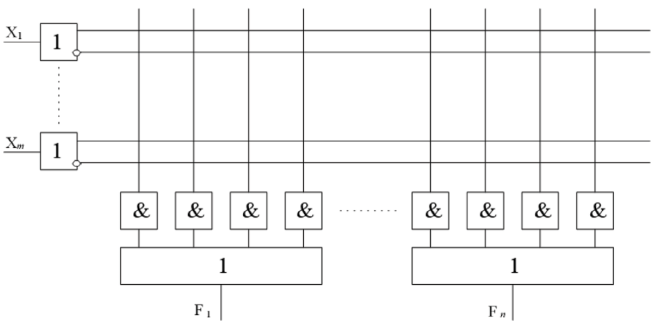


Рис. 1-2. Структура ПЛМ

ПЛИС, рассмотренные выше, – PROM, PAL, GAL, PLA – предоставили разработчикам определенную свободу в реализации проектов, обеспечили возможность уменьшения количества компонентов, повышения надежности изделий и снижения их стоимости. Стало возможным создавать вычислительные устройства не только на основе микропроцессоров с жесткой системой команд, но и с использованием микропроцессорных секций с микропрограммным управлением, причем набор команд уже определялся самим разработчиком, который запи-

сывал микропрограммы в ПЛИС. Таким же образом проектировались специализированные комбинационные устройства – преобразователи кодов, шифраторы и дешифраторы, а применение ПЛИС совместно с элементами памяти – триггерами, регистрами, счетчиками – позволяло разрабатывать быстродействующие устройства микропрограммного управления различными объектами.

Дальнейшее развитие технологии ПЛИС позволило включить в них и элементы памяти, так что разработчик смог перейти на качественно новый уровень, создавая не простые комбинационные устройства, а законченный последовательностный автомат, полностью реализующий требуемую диаграмму состояний-переходов без использования дополнительных элементов малой и средней степени интеграции. Достаточно наглядное представление о возможностях, предоставляемых ПЛИС с памятью, дает приведенная на рис. 1-3 схема популярной микросхемы PAL22V10, клоны которой выпускали и выпускают многие компании [4, 5].

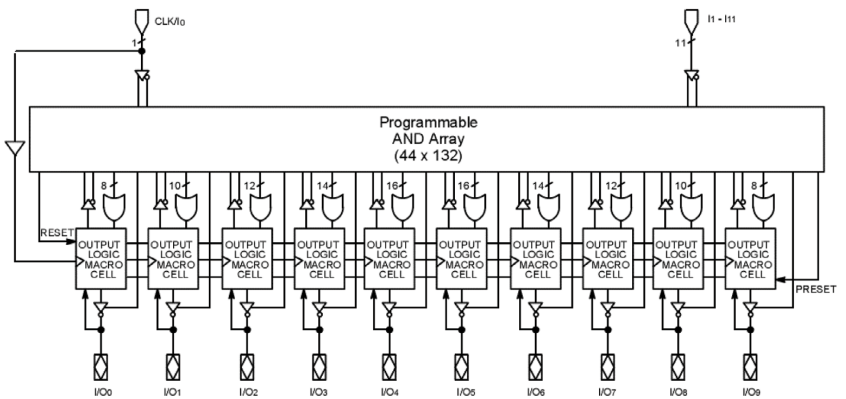
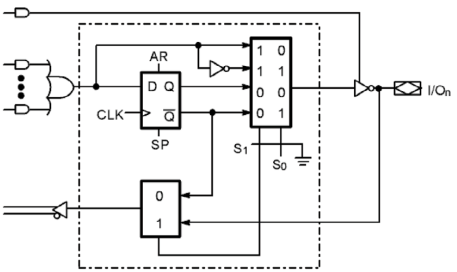


Рис. 1-3. Структура ПЛИС с памятью

Эта микросхема, относящаяся к классу PAL, содержит программируемую матрицу элементов «И» (Programmable AND Array), на входы которых I_1–I_{11} поданы 11 переменных (или их инверсии) с внешних входов ПЛИС и 10 внутренних переменных (или их инверсии), ассоциируемых с внешними входами/выходами ПЛИС. Выходы элементов «И» объединяются в линейке из 10 непрограммируемых элементов «ИЛИ» с фиксированным количеством входов – от 8 до 16. Сигналами с выходов элементов «ИЛИ» управляются выходные макроячейки (OUTPUT LOGIC MACRO CELL), каждая из которых содержит тактируемый D-триггер с входами сброса и установки.

Каждая из 10 макроячеек может быть сконфигурирована для работы в одном из 4 режимов, выбираемых с помощью программируемых перемычек S0 и S1 (рис. 1-4).



S0	S1	Выходной сигнал
0	0	Триггер, инверсный
0	1	Триггер, прямой
1	0	Комбинационный, инверсный
1	1	Комбинационный, прямой

Рис. 1-4. Макроячейка PAL22V10

Этими перемычками основной мультиплексор ячейки настраивается таким образом, что на выходной контакт ПЛИС поступает сигнал, реализуемый комбинационной логикой (матрицей «И» и «ИЛИ»), или тот же сигнал, зафиксированный D-триггером по приходу синхроимпульса CLK. Выходной сигнал (I/O_n) с помощью дополнительного мультиплексора может быть возвращен во входную комбинационную схему в качестве сигнала обратной связи для текущей макроячейки или в качестве дополнительной логической переменной – для остальных макроячеек.

Любой выход может быть переведен в высокоимпедансное состояние и использован как вход или двунаправленный вход/выход ПЛИС. Перечень полезных свойств этой микросхемы расширяют возможности сброса элементов памяти в исходное состояние по включению питания, загрузки в регистр произвольного кода через выходные контакты (что улучшает тестопригодность создаваемых на основе этой ПЛИС устройств), а также возможность установки защиты от несанкционированного копирования внутренней структуры ПЛИС – интеллектуальной собственности (Intellectual Property) разработчика.

Микросхемы типа PAL с элементами памяти вместе с ПЛИС комбинационного типа PROM, PAL, GAL в последующем были отнесены к классу простых программируемых логических устройств – SPLD (Simple Programmable Logic Device).

Отличительные признаки SPLD:

- каждая макроячейка имеет свой внешний выход и свой собственный триггер;
- в микросхеме SPLD реализовано не менее двух макроячеек;
- обычно все макроячейки выполнены одинаковыми;
- логическая функция макроячейки описывается одним логическим термом;
- логическая функция макроячейки реализуется матрицами элементов «И» и «ИЛИ».

К достоинствам SPLD обычно относят простоту проектирования специализированных устройств, постоянное и, как правило, одинаковое время прохождения сигналов со входов на выходы, возможность замены одной или несколькими микросхемами SPLD достаточно большого количества типовых микросхем малой и средней степени интеграции.

Основные недостатки SPLD – неэффективное использование ресурсов (логических вентилей) и, как следствие, проблематичность создания на их основе сложных цифровых устройств.

1.2. Технологии программирования ПЛИС

Последовательное увеличение степени интеграции микросхем и функциональной насыщенности реализуемых в кристалле ячеек, которые мог сконфигурировать разработчик, сопровождалось совершенствованием технологии программирования ПЛИС. И прежде чем продолжить обзор архитектур ПЛИС, целесообразно обсудить параллельно развивающиеся технологии программирования.

В первых семействах SPLD (PROM, PAL/GAL, PLA), выпускаемых на основе биполярных полупроводниковых элементов, однажды реализованная пользователем функция не могла быть изменена, структура соединений логических вентилей в ПЛИС записывалась однократно [5, 6]. Из-за этого если в процессе отладки вновь разработанного устройства обнаруживались ошибки, то для их исправления необходимо было извлечь из печатной платы ПЛИС, «прошитую» с ошибками, и заменить на исправленную. То есть по-прежнему разработчик был вынужден использовать паяльник (хоть и в меньшей степени, чем раньше) или, если это допускалось с точки зрения надежности изделия, монтировать на печатной плате гнезда (сокеты), позволяющие оперативно заменять микросхемы ПЛИС.

Совершенно новое качество было достигнуто после перехода на кристаллы с униполярными полупроводниковыми компонентами, в которых вместо однократно разрушаемых (создаваемых) связей-перемычек применялись запоминающие элементы на МОП-транзисторах с изолированным затвором. Заряд, создаваемый на затворе МОП-транзистора при программировании таких структур, сохранялся годами и не исчезал при отключении питания. Заряд этот можно снять, восстановив тем самым исходное состояние ячейки памяти, а при необходимости вновь записать – то есть появилась возможность **неоднократного** перепрограммирования ПЛИС. Технологии производства таких изделий разработаны в 1971 г. фирмой Intel, а в 1974 г. – фирмой Toshiba и реализованы в репрограммируемых ПЗУ (РППЗУ). Запись информации в РППЗУ производится в специальном режиме импульсами напряжения повышенной амплитуды, а стирание осуществляется двумя способами:

- ультрафиолетовым излучением – такие микросхемы назывались UV-EPROM (Ultraviolet Erasable PROM), УФ РППЗУ;
- электрическим напряжением противоположной полярности – микросхемы EEPROM (Electrically Erasable ROM), в отечественной литературе их обычно называют ЭСППЗУ – электрически стираемые программируемые ПЗУ.

Для стирания информации ультрафиолетовым излучением в корпусах микросхем РППЗУ создается закрытое кварцевым стеклом окошко, через которое облучается кристалл.

Теперь инженеру не нужно было иметь запас однократно программируемых устройств, чтобы заменить неправильно спроектированную микросхему на ис-