МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ І СПОРТУ УКРАЇНИ
Тернопільський національний технічний університет
імені Івана Пулюя

Кафедра комп’ютерних систем та мереж

КОНСПЕКТ ЛЕКЦІЙ
з дисципліни
“***Проектування спеціалізованих засобів обробки даних з використанням сигнальних процесорів та ПЛІС***”
для студентів денної та заочної форми навчання
напряму підготовки 6.050102 “Комп'ютерна інженерія”
спеціальності 8.091501 «Комп'ютерні системи та мережі»

Частина 2

Проектування цифрових пристроїв на базі ПЛІС, FPGA

Тернопіль – 2015

М.І. Паламар "Проектування спеціалізованих засобів обробки даних з використанням сигнальних процесорів та ПЛІС" / Конспект лекцій з дисципліни для студентів денної та заочної форми навчання спеціальності 8.05010201 "Комп'ютерні системи та мережі" та 8.05100306 – «Інформаційні технології в приладобудуванні». Частина 2: "Проектування цифрових пристроїв на базі ПЛІС, FPGA".- Тернопіль: ТНТУ, 2015. – 57 с.

Укладачі: проф., д.т.н. Паламар М.І.

Рецензенти:

професор Євтух П. С.
професор Щербак Л. М.

Відповідальний за випуск: проф., д.т.н., Паламар М.І.

Затверджено на засіданні кафедри “Комп’ютерних систем та мереж”
Протокол № від 2015р.

Схвалено та рекомендовано до друку методичною комісією факультету комп'ютерно-інформаційних систем і програмної інженерії Тернопільського національного технічного університету імені Івана Пулюя.

Протокол № від 2015р.

ЗМІСТ

Вступ

Розвиток новітніх технологій у світі комп'ютерної техніки й мікроелектроніки та створення програмних логічних інтегральних схем (ПЛІС) високого ступеня інтеграції викликало підвищений інтерес до досліджень в області реконфігуровного комп'ютингу. Реконфігурована комп'ютерна система поєднує поняття реконфігурованої структури апаратного пристрою і процесу обробки даних, який виконується цим пристроєм. Відмінність реконфігуровного комп'ютера від традиційного полягає в тому, що структура реконфігуровних пристроїв не є фіксованою і змінюється в залежності від виконуваної задачі чи алгоритму, що дозволяє вносити зміни в готовий і функціонуючий продукт.

Gerald Estrin на початку 1960 років запропонував *«комп'ютер з елементами фіксованої та змінної структури»* [1]. Ця базова архітектура, яка підтримує програмовані пристрої і програмне забезпечення, лежить в основі реконфігуровних комп'ютерних систем. Дана концепція набагато випередила розвиток технологій у світі мікроелектроніки і, на жаль, він зміг представити тільки наближення своєї розробки. Множина понять, які зараз відкриті організацією з досліджень в області реконфігуовного комп'ютингу, не були прийняті до уваги в цих дослідженнях.

Втілення основних концепцій реконфігуровної комп'ютерної системи в реальні проекти стало можливим тільки з появою інтегральних схем типу FPGA (field programmable gate array), які містять масиви конфігуровних логічних блоків і програмованих взаємозв'язків між цими блоками [21. Логічні блоки можуть бути зконфігуровані на виконання простих чи складних логічних функцій і можуть змінюватися відповідно до заданої вимоги.

1. Механізми розробки комп’ютерних архітектур що базуються на інтегральних схемах типу FPGA

Ємність сучасних FPGA нараховує більше ніж 10 мільйонів логічних вентилів, крім того, FPGA також включають RAM (Read Access Memory), апаратні перемножувачі, які є стандартними компонентами сучасних мікропроцесорів та фіксоване мікропроцесорне ядро. Метод, за яким конфігуруються логічні блоки та взаємозв'язки, специфічний кожному виробнику FPGA. Логічні блоки та взаємозв'язки мають внутрішню структуру, яка зберігає поточну конфігурацію.

Час, потрібний для конфігурації FPGA, називається часом конфігурації. Час конфігурації залежить від серії, ємності FPGA та їх кількості. Для реконфігуровної комп'ютерної системи, яка включає декілька FPGA час конфігурації залежить не тільки від часу конфігурації окремої інтегральної схеми, а й від конфігурації всіх схем. Кристал FPGA може бути сконфігурований послідовно чи паралельно в залежності від розробки системи і, таким чином, час конфігурації може варіювати від сотень наносекунд до кількох секунд. Рисунок 1.1 ілюструє базові архітектури реконфігуровних систем, з яких походить більшість архітектур.

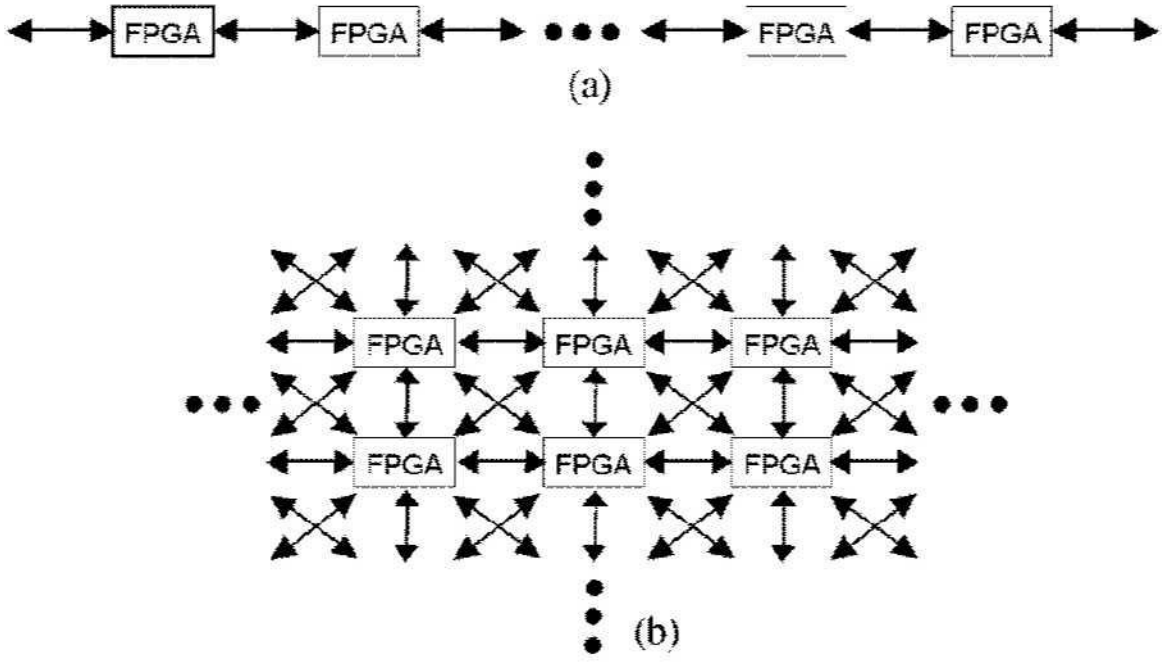

Рис.1.1. Базові архітектури реконфігурованих систем з множиною FPGA.

У Сполучених Штатах Америки ведуться інтенсивні роботи у напрямку досліджень та розробки комп'ютерних архітектур, які базуються на інтегральних схемах типу FPGA з метою прискорення обробки даних. Розробка структури реконфігурованого пристрою і її реалізація в кристалі на вентильному рівні дозволяє збільшити швидкодію комп'ютера на декілька порядків у порівнянні з традиційними рішеннями.

Не всі алгоритми цифрової обробки даних можуть бути ефективно реалізовані з використанням сучасних інтегральних схем. На Сьомій Міжнародній Конференції «IRTC-2001» обговорювалися такі проблеми як виконання обчислень з плаваючою комою та виконання числових операцій з високою точністю.

Процес проектування цифрових схем спрощується з появою апаратних мов описів, тому що використання машинних схематичних інструментальних засобів стає складним при наявності великої кількості інтегральних схем. Міжнародним стандартом є мова HDL (Hardware Description Language), спочатку розроблена для цілей проектування, моделювання і документування. Синтез з інструментальними засобами дозволив автоматизувати процес проектування.

До складу САПР ПЛІС вводиться **CORE Generator** -інструментальний засіб, який надає у розпорядження користувача параметричні логічні Cores, оптимізовані для ПЛІС, які описуються мовою VHDL. Система CORE дозволяє істотно зменшити час розробки нових проектів. У відповідності зі сформульованими технічними вимогами, проектувальник по мережі Internet може одержати оптимізоване для FPGA логічне Core і включити його у свій проект.

Істотно скорочується час розробки проекту з використанням System Generator, який є програмним інструментальним засобом для проектування, моделювання та створення систем Digital Signal Processing (DSP) на базі FPGA. Він дозволяє швидке застосовування алгоритмів DSP до пристроїв FPGA. Скорочення процесу розробки проекту, від його концепції до робочих апаратних засобів, досягається за допомогою використання множини компонентів одночасно. Потоки даних, традиційні мови апаратних описів (VHDL і Verilog) та функції, які одержані з мови програмування MATLAB, можуть використовуватися одночас-

но, моделюватися разом і синтезуватися в робочі апаратні засоби. На рис.1.2 зображено процес моделювання системи DSP.

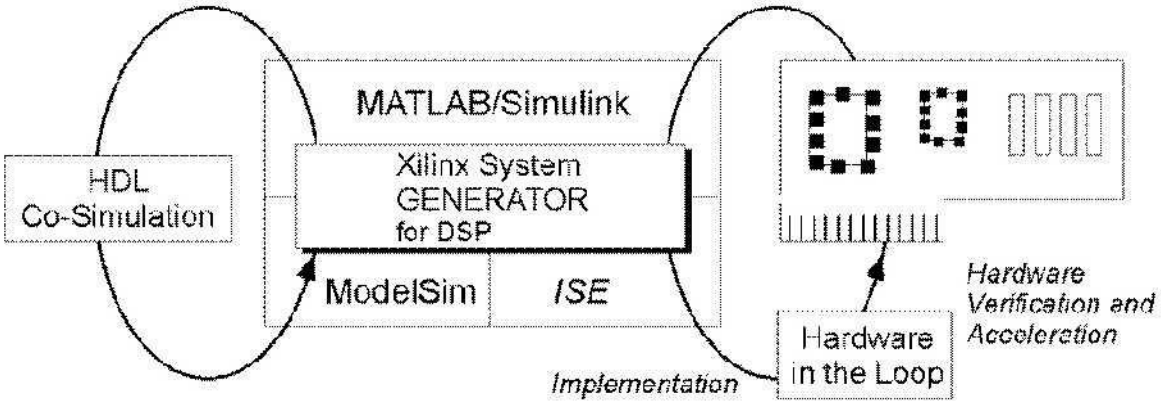

Рисунок 1.2. Моделювання системи DSP

MATLAB інтегрує математичні обчислення, візуалізацію і потужну мову, забезпечуючи гнучке середовище для технічних застосувань. Відкрита архітектура робить зручним використання MATLAB для дослідження даних, створення алгоритмів та користувальницьких інструментальних засобів.

Simulink - інтерактивний інструмент для моделювання, імітації та аналізу динамічних систем. Він дозволяє точно описувати, моделювати, оцінювати та уточнювати поведінку системи завдяки стандартним і користувальницьким бібліотекам. Simulink цілісно інтегрується з MATLAB, забезпечуючи швидкий доступ до множини інструментальних засобів проектування та аналізу.

System Generator розширює можливості Simulink моделювання на системному рівні за допомогою bit і cycle-true моделювання схем FPGA.

System Generator включає Simulink бібліотеку функціональних блоків для побудови DSP, арифметичних схем і схем цифрової логіки. Розробнику надається можливість комбінування блоків Xilinx з блоками MATLAB і Simulink для створення реалістичних test benches і аналізу даних. Високий рівень абстракції, який надає System Generator, спрощує розробку алгоритмів і їх перевірку.

Крім бібліотеки моделювання на системному рівні System Generator включає генератор коду, який автоматично генерує синтезований VHDL код з Sim-

ulink моделі. VHDL код включає IP (intellectual property) блоки, які розроблені для високої продуктивності та інтенсивності в пристроях Xilinx типу FPGA.

Вимоги до пристроїв на базі сучасних FPGA роблять майже неможливим налагодження та перевірку проектів з використанням традиційних логічних методів аналізу. Компанією Xilinx розроблений налагоджувач апаратних засобів ChipScope Pro, який дозволяє перевіряти функціонування схем на базі FPGA. ChipScope Pro включає losic analyzer (ILA) і bus analyzer (IBA). Ці Cores дозволяють переглядати всі внутрішні сигнали, вузли і системні шини в межах FPGA.

Створення ПЛІС високого ступеня інтеграції обумовило появу технології Internet Reconfigurable Logic (IRL). Ця технологія забезпечує дистанційну реконфігурацію структур апаратних пристроїв, включених до мережі і реалізованих на елементній базі ПЛІС, через мережу Internet. Для розробки продуктів на основі технології IRL необхідний розвиток трьох фундаментальних технологій - всеохоплюючого використання мережних технологій, набору програмних засобів на основі мови Java і реконфігурованих кристалів FPGA типу Virtex. Архітектура Virtex на даний час дозволяє розробляти пристрої з кількістю вентилів 10 мільйонів і може містити в собі апаратну реалізацію інтерпретатора Java програм для Java Virtual Machine (JVM).

Технологія Java дозволяє створювати програми для роботи з файловими системами і з локальними та глобальними комп'ютерними мережами. Java-застосування компілюються в байт-код, який виконується на JVM, яка представляє собою інтерпретатор байт-коду Java, що дозволяє динамічно розширювати систему. При першому запуску Java-програми JVM здійснює збір модулів і встановлює зв'язки між ними, при чому пошук відсутніх модулів здійснюється не тільки в локальній системі, а й у мережі.

Перед першим запуском нового застосування віртуальна машина перевіряє його код на належність до байт-коду, на небезпечність команд для системи і локальної мережі, на відповідність з дозволеними операціями, що свідчить про високий ступінь захисту та небезпеки мови Java для роботи в мережі. Java забезпечує швидкий цикл компіляції і налагодження програм. Засоби розробки, що міс-

тять JVM, забезпечують контроль програмних застосувань на стадії виконання. Істотною рисою архітектури Java є вбудована багатопотоковість, що дає можливість одночасно використовувати кожну функцію даної бібліотеки декількома потоками та виконувати декілька задач одночасно в межах одного застосування, що є актуальним для розподілених систем, коли процеси обміну по мережі можуть протікати одночасно і асинхронно. Обє'ктно-орієнтована парадигма мови Java зручна для організації розподілених обчислень, а також при створенні програмного забезпечення типу клієнт-сервер.

Завдяки наявності JVM мова Java є мобільною. JVM забезпечує абстрагованість скомпільованих Java-програм від апаратної платформи й операційної системи, яка досягається перш за все стандартизацією «бінарного формату коду». Байт-код не залежить від конкретної платформи чи системи, для роботи Java-програми на даній апаратно-програмній платформі достатньо наявності відповідної JVM. Повна специфікація віртуальної Java машини відкрита й загальнодоступна і віртуальна машина може бути реалізована на любій сучасній апаратно-програмній платформі. Система Java в силу своєї високої продуктивності, інтерпретованості і динамічної природи підходить для цілей швидкої розробки надійних програм.

Корпорацією Xilinx був представлений інструментальний пакет JBits (Java Based interface) для проектування систем за технологією IRL, призначений для створення спеціальних програмних застосувань мовою Java, які можуть використовуватися для модифікації апаратних засобів через мережу Internet.

JBits бере початок від робот, які пов'язані з кристалами серії XC6200 і базується на Java Environment for Re-configurable Computing (JERC). JERC надавав у розпорядження Application Programming Interface (API), який дозволяв маніпуляцію з бітами конфігурації низького рівня.

Мотивацією розробки JBits послужила підтримка динамічної реконфігурації для кристалів FPGA серії Virtex. Так як проектування пристрою було обмежене статичними методологіями розробки, інструментальними засобами проектування інтегральних схем - редактором схем або мовою HDL і не було забезпе-

чене програмною підтримкою конфігурації, тому динамічна реконфігурація не була можливою.

На зміну статичному способу проектування прийшло рішення, яке передбачало підтримку програмного забезпечення, яке б дало повний доступ до всіх архітектурних особливостей реконфігурованого пристрою. Цим забезпеченням стала бібліотека скомпільованих Java програм. Так як бібліотека являє собою скомпільовані програми - виходом проектування не є статичний двійковий потік, а є Java код, який виконується і забезпечує керування конфігурацією і даними реконфігурованої логіки.

Перевагою процесу проектування з використанням JBits (рис. 1.3) є те, що виконувана програма являється тільки довільною частиною скомпільованого Java коду, що робить її переносною на інші системи і дозволяє інтеграцію з іншими частинами системи.

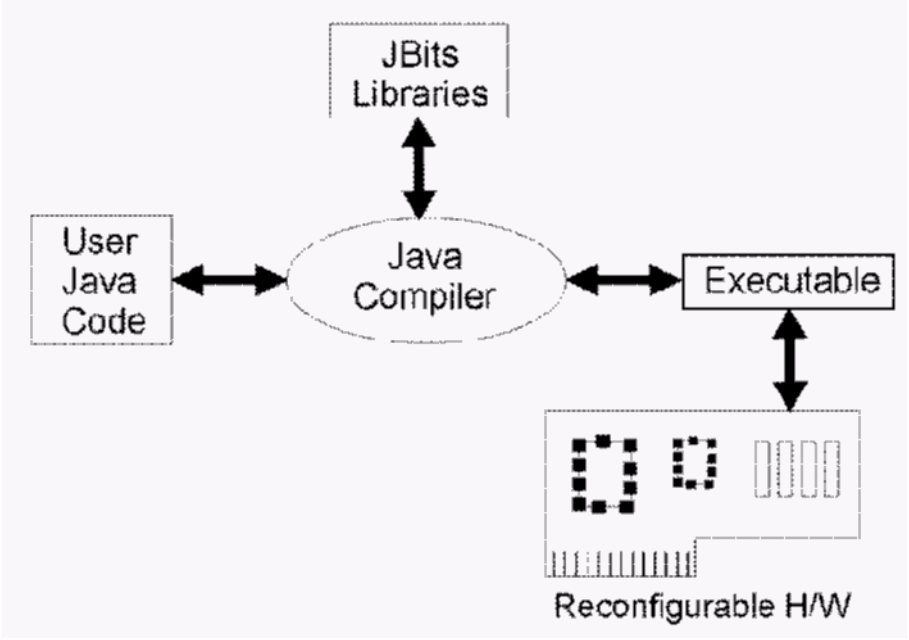

Рисунок 1.3. Процес проектування з використанням J Bits

JBits - це Java API, який забезпечує доступ до конфігурації пристроїв FPGA серії Virtex. Інтерфейс дозволяє доступ до всіх реконфігуровних ресурсів пристрою для їх швидкої модифікації під програмним керуванням, що, в свою чергу, забезпечує програмну підтримку нових можливостей, які не були реалізовані раніше в пристроях Xilinx.

В той час як оточення JBits було представлене вперше, воно вимагало налагодження проекту за допомогою використання фізичних апаратних засобів. Згодом був представлений Device Simulator, який працює на двійкових потоках і емулює пристрій FPGA в програмному забезпеченні. Це було важливим кроком у напрямку проектування під програмним керуванням, але не було враховано спосіб моделювання зовнішніх периферійних пристроїв, які взаємодіють з FPGA. Недоліком Virtex Device Simulator було те, що він не мав вбудованої схеми для перенесення даних до / від пристрою.

JBits забезпечує можливість швидкого створення і реконфігурації схем серії Virtex під час виконання програмного застосування за допомогою. Run-Time Reconfiguration (RTR).

Моделювання на рівні пристрою з використанням мови Java має переваги над традиційними методами статичного проектування і налагодження проекту. Перевагами є перевірка проекту на рівні двійкових потоків, підтримка RTR, гнучкість при роботі у середовищі моделювання проекту і середовищі налагодження та контролю.

Використання стандарту HDL для тестування проекту дозволяє моделювати апаратні засоби, включаючи моделювання поведінки. Але цей підхід працює лише на рівні проектування, і не обов'язково забезпечує точну модель поведінки під час запуску проекту на фізичних апаратних засобах. Перевірка правильності проекту з використанням традиційного методу вимагає щоб кожен крок компіляції був без помилок, що є часто неможливим. VHDL проект і середовища налагодження також не підтримують реконфігурацію під час виконання і не забезпечують гнучкість при роботі.

Система JBits. На рис. 1.4 зображена схема системи JBits. *User Java Application* являє собою написану програму мовою Java, яка використовує JBits інтерфейс для маніпуляції конфігуровними ресурсами FPGA, які включають пошукові таблиці, маршрутизацію та тригери. Кожна функція, яка викликається на рівні JBits інтерфейсу, робить один чи більше запитів до *Bit level Interface*, який відповідає за розміщення біта у двійковому потоці для будь-якого пристрою се-

рії Virtex. Bit-level Interface взаємодіє з файлом *Bitstream,* який керує двійковим потоком пристрою і підтримує зчитування та запис двійкових потоків з файлу / у файл. Можливість зчитування даних з пристрою і представлення їх у двійковому форматі є необхідністю для динамічної конфігурації.

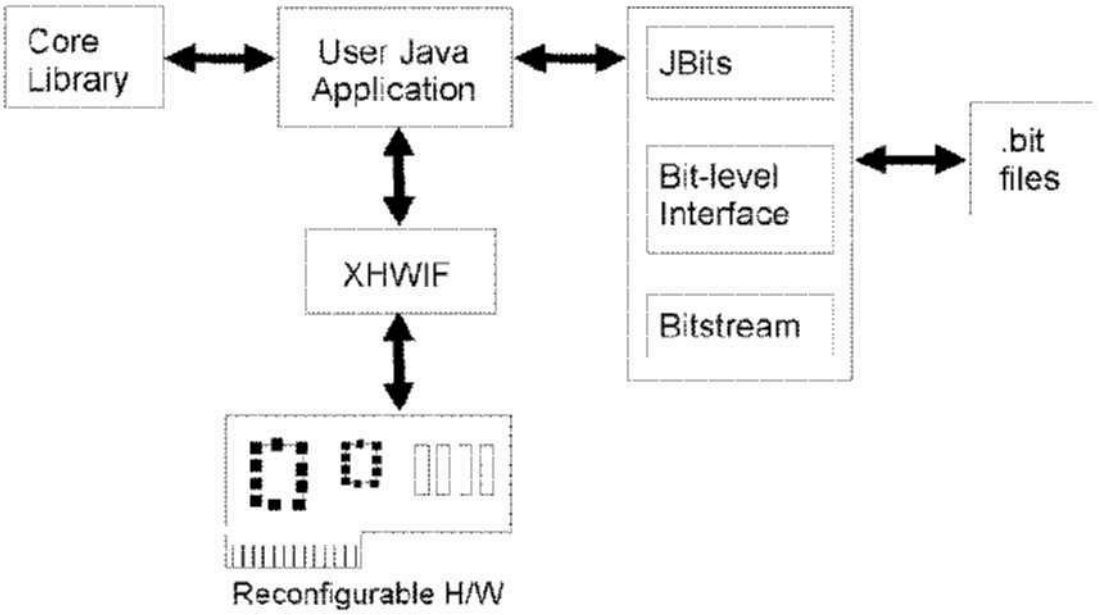

Рисунок 1.4. Система Jbits

JBits API використовує *Xilinx Hardware InterFace (XHWIF)* - Java інтерфейс, який розроблено для зв'язку зі схемами на базі FPGA і надає у розпорядження методи зчитування/запису цифрових потоків в/із FPGA, методи опису видів і кількості FPGA на платі. *XHWIF* також включає метод інкременту інтегрованого таймера та метод запису в блоки пам'яті, які розташовані на схемі, якщо вони доступні. Таким чином, інтерфейс описує схему і дозволяє пересилання даних на схему і зі схеми. XHWIF інтерфейс стандартизує спосіб, по якому застосування спілкуються з апаратними засобами таким чином, що вони, використовуючи той же самий інтерфейс, можуть взаємодіяти з множиною схем. *XHWIF server* дозволяє іншим застосуванням спілкуватися з реконфігуровними схемами, які знаходяться в межах мережі Internet. Ця можливість дозволяє налагоджувати проект без прямого доступу до апаратного забезпечення, крім того, дозволяє доступ до схеми множиною користувачів.

Core Library - це колекція Java програм, які визначають макрокомірки або Cores. Вони за звичай параметризовані і можуть переміщатися в межах пристрою. Прикладами Cores є лічильники, суматори, перемножувачі та інші стандартні логічні функції та функції обчислень.

Налагоджувач BoardScope. Корпорацією Xilinx був розроблений пакет BoardScope, подібний до старого інструменту WebScope для пристроїв серій XC6200, -інтерактивний налагоджувач апаратних засобів, який дозволяє слідкувати за функціонуванням схем на базі FPGA. BoardScope дає можливість налагоджувати схеми FPGA при їх взаємодії з іншими апаратними компонентами конфігурації, відслідковувати внутрішній стан систем, які включають декілька FPGA, контролювати рівні сигналів, а також стан багаторозрядних шин даних. Пакет може використовуватися для віддаленого налагодження пристроїв систем IRL або для спільної розробки автономних систем чи окремих кристалів через мережу Internet різними проектувальними групами.

З використанням XHWIF, двійкові потоки завантажуються для конфігурації FPGAs, або зчитуються - для їх аналізу (рис. 1.5).

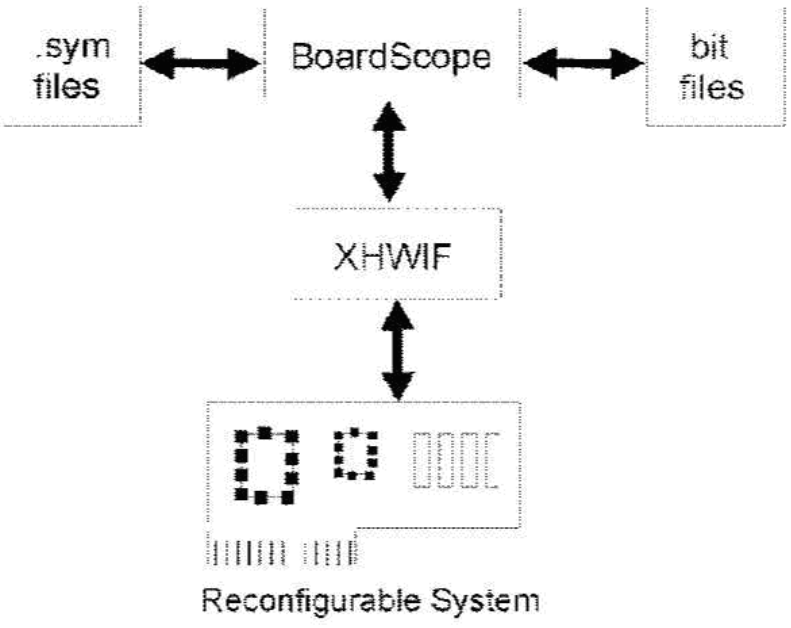

Рисунок 1.5. Структура налагоджувача BoardScope

Крім того що BoardScope може взаємодіяти з локальними апаратними засобами та апаратними засобами, які включені до мережі, він може використовувати Device Simulator.

Device Simulator виконує моделювання апаратних засобів, яке засновано на параметрах конфігурації, і може використовуватися коли апаратні засоби недоступні. У такому випадку налагодження з використанням Device Simulator потребує порівняно менше часу, а ніж з прямим використанням апаратних засобів, так як не потребує виконання операції завантаження та зчитування двійкових потоків. На даний час BoardScope підтримує пристрої серії Virtex.

Run-Time Reconfiguration. JBits, на відміну від традиційних мов апаратних описів, створює нові можливості в області RTR, забезпечуючи можливість швидкого створення і динамічної модифікації схем серії Virtex під час виконання програмного застосування за допомогою прямого доступу до конфігурації двійкового потоку.

RTR визначається як динамічна модифікація апаратних схем FPGA під час виконання. RTR системи розрізняють за визначенням логіки схем і маршрутизації до виконання операції чи під час виконання. RTR системи модифікують схему декілька разів на протязі виконання програмного застосування. Так як не виключається можливість проектування низького рівня, JBits надає у розпорядження необхідні інструментальні засоби для ефективного модифікування чи створення проекту.

RTR досягається за допомогою інтеграції JBits і XH-WIF API у JBits технології. Тобто, RTR застосування може робити запити до JBits інтерфейсу для модифікації даних конфігурації у двійковому потоці і явні запити XHWIF для взаємодії з апаратним забезпеченням. Наприклад, RTR застосування виконує запити setConfiguration (пристрій, дані) і getConfiguration () для виконання завантаження і зчитування даних конфігурації.

На даний час JBits 3.0 для Virtex-II містить файли класу для створення Run Time Reconfigurable застосувань і інструментальні засоби, які використовують

архітектуру Virtex-II. JBits API дозволяє Java застосуванням динамічно модифікувати Xiiipx Virtex-II двійкові потоки.

Обмеження JBits. Поряд з усіма перевагами способу проектування інтегральних схем з використанням JBits існують і деякі обмеження JBits API. Програмне забезпечення вимагає повної специфікації деталей, на відміну від інструментів статичного проектування. JBits інтерфейс підтримує тільки структуровані схеми. Не-структуровані схеми, такі як схеми з випадковою логікою, не задовольняють пряме виконання в JBits застосуваннях.

JBits API вимагає від користувача повного ознайомлення з архітектурою пристроїв. В той час як архітектура пристроїв повністю задокументована фірмою Xilinx, більшість користувачів не мали необхідності ознайомитися з її особливостями. Очікується, що необхідність розуміння архітектури пристроїв буде самим великим бар'єром для схвалення і широкого поширення JBits інтерфейсу чи інших йому подібних інструментів.

В той час як JBits API дозволяє використовувати схеми, які створено стандартними інструментальними засобами розробки, зміною чи реконфігурацією схеми, на рівні JBits виключається можливість використання будь-яких інструментальних засобів аналізу, доступних проектувальникам схем. В JBits відсутня можливість аналізу синхронізації. Навіть невеликі зміни в конфігурації схеми можуть вплинути на її функціональні можливості, такі як синхронізація. Лише одним засобом, який нещодавно був розроблений для налагоджування апаратних засобів, є BoardScope.

2. Огляд елементної бази.

2.1 Програмовані логічні інтегральні схеми: огляд архітектур й особливості застосування.

Програмовані логічні інтегральні схеми стають останнім часом усе більше розповсюдженою й звичною елементною базою для розробників цифрових пристроїв. Останні роки характеризуються різким ростом щільності упакування елементів на кристалі, багато провідних виробників або почали серійне виробництво, або анонсували ПЛІС із еквівалентною ємністю більше 1 мільйона логічних вентилів. Ціни на ПЛІС (на жаль, тільки лише в доларовому еквіваленті) неухильно падають. Так, ще рік - півтора назад ПЛІС ємністю 100 000 вентилів коштувала залежно від виробника, прийомки, швидкодії від 1500 до 3000 у.о., то зараз така мікросхема коштує від 50 до 350 у.о., тобто ціни впали практично на порядок і ця тенденція стійка. Що стосується ПЛІС ємністю 10 000 - 30 000 логічних вентилів, то з'явилися мікросхеми вартістю менш 10 у.о.

У таблиці 1 наведена динаміка розвитку ринку ПЛІС.

Таблиця 1. Обсяг ринку ПЛІС, млн $.

Область проджу	1994	1995	1996	1997	1998	1999
Військово-промислова й космічна	43	68	92	119	150	188
Цивільна	684	1125	1598	2146	2823	3678
Разом	727	1193	1690	2265	2973	3866

У зв'язку з цим з'являється ряд питань, пов'язаних з тим, яку елементну базу і як використати в нових розробках, а також при проведенні модернізації існуючих систем.

Розглянемо особливості вибору елементної бази на прикладі проектування пристроїв цифрової обробки сигналів.

Сучасні алгоритми обробки сигналів функціонально можна розділити на наступні основні класи.

1. Алгоритми цифрової фільтрації (у т.ч. алгоритми нелінійної, оптимальної, адаптивної фільтрації, евристичні алгоритми, поліноміальні фільтри, алгоритми фільтрації зображень й ін.).
2. Алгоритми, засновані на застосуванні ортогональних перетворень (швидкі перетворення Фур'є, Хартлі, Уолша, Адамара, перетворення Карунена – Лоєва й ін.)
3. Алгоритми, що реалізують кодування й декодування, модулятори й демодулятори, у тому числі складних сигналів (псевдовипадкових, хаотичних й ін.).
4. Алгоритми інтерфейсів і стандартних протоколів обміну й передачі даних.

Далі розглянемо перспективи тих або інших шляхів реалізації алгоритмів ЦОС.

2.2. Реалізація алгоритмів ЦОС на основі спеціалізованих БІС.

Існує цілий ряд пристроїв й алгоритмів, які практично є стандартними, і в великій кількості повторюються від розробки до розробки. Прикладом таких пристроїв можуть служити вузли масових комунікаційних засобів, мікросхеми складних інтерфейсів (pCI, pCMCI і т.д.), компоненти систем мультимедіа й відео-обробки для масових комп'ютерів і т.п. На жаль, ці БІС володіють рядом недоліків, що стримують їхнє застосування в розробках вітчизняного виробника.

Розглянемо їх докладніше:

По-перше, практично повністю відсутнє власне виробництво й розробка масових високотехнологічних засобів зв'язку й компонентів обчислювальної техніки. У зв'язку з цим придбати подібні БІС для апробації практично неможливо.

По-друге, адаптація стандартних компонентів для обробки сигналів під спеціалізоване завдання вимагає застосування додаткових схем сполучення, об-в'язки, найчастіше нестандартних і знову розроблювальних, що практично зводить нанівець всі переваги спеціалізованих БІС. Виготовлення ж БІС за замовленням практично неможливо через високу вартість.

У зв'язку з цим досить цікавими й перспективними до застосування у вітчизняних умовах доступні БІС, що реалізують, з одного боку, деякі стандартні протоколи передачі даних, з іншої сторони наділені достатньою гнучкістю й сумісністю на рівні програм зі стандартними керуючими або сигнальними процесорами. Подібні БІС роблять такі компанії як Giga, Mitel, Teltone, Motorola, Siemens, plessey, Zilog, Harris і ряд інших (особливо японських і корейських) компаній. Так, фірма Zilog пропонує досить широку номенклатуру БІС для реалізації систем передачі й обробки даних, що володіють, з одного боку, підтримкою специфічних функцій, характерних для комунікаційних завдань, з іншого боку, сумісних по програмному коді й інтерфейсу зі стандартними апаратними засобами. Дана програма створення БІС одержала назву Zilog superintegration TM , і в її рамках розроблені наступні кристали:

- Z89C25 на 80% сполучимо по кодах із процесором TMS320C25, містить ряд додаткових інструкцій для забезпечення функціонування додаткових

пристроїв, має 32 розрядні АЛУ й акумулятор, 16 розрядний перемножувач. На цьому ж кристалі інтегровані напівдуплексний кодек, контролер протоколу V.24, таймери.

- Z01701 крім ядра цифрового сигнального процесора містить контролери протоколів V.17, V.29, V.27, V.21, сполучимо зі специфікаціями T.30 і T.4, має інтегровані фільтри.
- Z80382 забезпечує підтримку шини pCMCI, інтерфейсів GCI, plug-and-play, асинхронного послідовного адаптера, має убудоване процесорне ядро.

Крім вищезгаданих БІС, Zilog випускає досить широку номенклатуру БІС контролерів різноманітної периферії з вбудованим процесорним ядром. Застосування подібних кристалів у розробках експериментальних, дрібно- і середньосерійних пристроїв дозволить досягти високого результату за прийнятну ціну.
Інший клас спеціалізованих БІС, що заслуговує уваги розробників - БІС для реалізації специфічних алгоритмів, такі як нейрочіпи, процесори для розподілених обчислень, обробки радіолокаційної інформації й інших. Незважаючи на те, що їхнє виробництво й застосування перебувають у зародковому стані, багато сучасних алгоритмів реалізувати іншим шляхом практично неможливо.
Таким чином, застосування спеціалізованих БІС в сучасних російських умовах, на жаль, майже не поширене й обмежується в основному реалізацією протоколів передачі даних.

2.3.Реалізація алгоритмів ЦОС на основі цифрових сигнальних процесорів загального призначення.

У цей час велика кількість розробників вибирають як засіб реалізації алгоритмів цифрові сигнальні процесори (ЦСП) загального призначення. У цьому є певний глузд, пов'язаний з тим, що ЦСП досить поширено й доступні на ринку, мають привабливі ціни. Головною перевагою систем обробки сигналів на ЦСП є гнучкість системи, можливість реалізації адаптивних й алгоритмів, що навчаються. Крім того, відладочні засоби початкового рівня недорогі, достатня інформаційна підтримка, випущена література по застосуванню російською мовою.

Лідером по розробці й виробництву ЦСП є компанія Texas Instruments (TI). Далі по обсязі виробництва (1996-1997 роки) випливають Motorola й Lucent (AT&T). Як не дивно, що займає на російському ринку провідні позиції Analog devices перебуває на 4 місці. Проте, ЦСП цієї компанії, мабуть, щонайкраще пристосовані до реалізації широкого кола завдань ЦОС. Їхня основна перевага - широка номенклатура програмно повністю сумісних пристроїв з різною швидкодією й додатковими периферійними елементами, з фіксованою й плаваючою крапкою. Наявність недорогих засобів налагодження дозволяє використати ці ЦСП у малобюджетних проектах.

Разом з тим, ЦСП мають ряд недоліків, які безумовно доводиться враховувати при розробці нових виробів. По-перше, поки тактова частота портів обміну даними ЦСП не перевищує 100 Мгц, що обмежує область застосування в системах радіодіапазону. По-друге, кожне сімейство ЦСП має власні коди команд, що робить практично неможливим перенос реалізованого алгоритму на ЦСП інших сімейств або створення універсальних бібліотек алгоритмів. Існуючі ж компілятори з мов високого рівня, наприклад із Сі, також орієнтовані на конкретні ЦСП і не вирішують дану проблему. По-третє, при реалізації складних паралельних структур доводиться збільшувати число процесорів і забезпечувати їхню нормальну роботу в мультипроцесорному режимі. Нарешті, по-четверте, ЦСП, як правило, вимагають зовнішніх навічних елементів для реалізації спряжувального інтерфейсу із джерелами й приймачами даних.

2.4 Реалізація алгоритмів ЦОС на базі ПЛІС.

Основними перевагами ПЛІС при застосуванні в засобах обробки сигналів є:

- висока швидкодія;
- можливість реалізації складних паралельних алгоритмів;
- наявність засобів САПР, що дозволяють провести повне моделювання системи;
- можливість програмування або зміни конфігурації безпосередньо в системі;
- сумісність при перекладі алгоритмів на рівні мов опису апаратури (VHDL, AHDL, Verilog й ін.);
- сумісність по рівнях і можливість реалізації стандартного інтерфейсу;
- наявність бібліотек мегафункцій, що описують складні алгоритми;
- архітектурні особливості ПЛІС якнайкраще пристосовані для реалізації таких операцій, як множення, згортка й т.п.

У даний час швидкодія ПЛІС досягла величин порядку 250 - 300 Мгц, що дозволяє реалізовувати багато алгоритмів у радіодіапазоні.

Розглянемо історію розвитку архітектур ПЛІС. Наприкінці 1970 років на ринку з'явилися ПЛІС, що мають програмувальні матриці "І" й "АБО". У закордонній літературі ці архітектури FpLA (Field programmable Logic Array) і FpLS (Field programmable Logic Sequensers). У ті часи вітчизняна електронна промисловість була ще "на плаву" і незабаром з'явилися вітчизняні схеми К556р1,р2,р21. Недолік такої архітектури - слабке використання ресурсів програмувальної матриці "АБО".

Ідучи по шляху вдосконалювання такої архітектури, розроблювачі ПЛІС запропонували більш просту й витончену архітектуру - архітектуру програмованої матричної логіки (pAL - programmable Array Logic й GAL - Gate Array Logic) - це ПЛІС, що мають програмувальну матрицю "И" і фіксовану матрицю "АБО", у ПЛІС GAL на виході є тригер. До цього класу відноситься широка номенклатура ПЛІС невеликої ступені інтеграції. Як приклади можна привести вітчизняні

ИС КМ1556ХП4, ХП6, ХП8, ХЛ8, ранні розробки (середина -кінець 1980-х років) ПЛІС фірм INTEL, ALTERA, AMD, LATTICE й ін. Крім pAL й GAL архітектур, були розроблені ПМЛ, що мають тільки одну програмувальну матрицю "І", наприклад, схема 85С508 фірми INTEL. Іншим підходом до зменшення надмірності програмованої матриці "АБО" є програмована макрологіка. ПЛІС, побудовані по даній архітектурі містять єдину програмовану матрицю "І-НІ" або "АБО-НІ", але за рахунок численних інверсних зворотних зв'язків здатні формувати складні логічні функції. До цього класу ставляться, наприклад, ПЛІС pLHS501 й pLHS502 фірми SIGNETICS, що мають матрицю "І-НІ", а також схема XL78C800 фірми EXEL, заснована на матриці "АБО"-НЕ.

Вище перераховані архітектури ПЛІС містять невелике число комірок, до теперішнього часу морально застаріли й застосовуються для реалізації простих пристроїв, для яких не існує готових ІС середнього ступеня інтеграції. Природно, для реалізації серйозних алгоритмів керування або ЦОС вони не придатні.

На початку 1980 років на світовий ринок мікроелектронних виробів виходять три провідні фірми-виробники ПЛІС. У червні 1983 року заснована фірма Altera Corporation, (101 Innovation Drive, San Jose, CA 95134, USA, www.altera.com), у лютому 1984 компанія Xilinx, Inc. (2100 Logic Drive, San Jose, CA 95124-3400, USA, www.xilinx.com), в 1985 році - Actel Corporation (955 East Arques Avenue, Sunnyvale, CA 94086-4533, USA, www.actel.com). Ці три компанії займають до 80% усього ринку ПЛІС й є основними розроблювачами ідеології їхнього застосування. Якщо раніше ПЛІС були одним з безлічі продуктів, що випускають такими гігантами, як Intel, AMD й ін., то починаючи із середини 1980 років на ринку ПЛІС відбувається спеціалізація й законодавцями мод є фірми, що спеціалізуються тільки на розробці й виробництві ПЛІІС.

З появою нових виробників з'явилися й нові архітектури. ІС ПМЛ мають архітектуру, досить зручну для реалізації цифрових автоматів. Розвиток цієї архітектури - CpLD (Complex programmable Logic Devices) - ПЛІС, що містять кілька логічних блоків (ЛБ), об'єднаних комутаційною матрицею. Кожен ЛБ являє собою структуру типу ПМЛ, тобто програмувальну матрицю "И" і фіксовану

матрицю "АБО". ПЛІС типу CpLD, як правило, мають досить високий ступінь інтеграції (до 10000 еквівалентних вентилів, до 256 макроосередків). До цього класу ставляться ПЛІС сімейства MAX5000 й MAX7000 фірми ALTERA, схеми XC7000 й XC9500 фірми XILINX, а також велика кількість мікросхем інших виробників (Atmel, Vantis, Lucent й ін.). Розглянемо цю архітектуру на прикладі ПЛІС сімейства MAX3000 фірми Altera.. Їхня архітектура близька до архітектури сімейства MAX7000, однак є ряд невеликих відмінностей. У таблиці наведені основні параметри ПЛІС MAX3000.

Таблиця 2.

	Ep3032A	Ep3064A	Ep3128A	Ep3256A
Логічна ємність, еквівалентних вентилів	600	1250	2500	5000
Число макроосередків	32	64	128	256
Число логічних блоків	2	4	8	16
Число програмованих користувачем виводів	34	66	96	158
Затримка поширення сигналу вхід-вихід, tp, нс	4.5	4.5	5	6
Час установки глобального тактового сигналу, t_{SU}, нс	3.0	3.0	3.2	3.7
Затримка глобального тактового сигналу до виходу, t_{CO1}, нс	2.8	2.8	3.0	3.3
Максимальна глобальна так-	192.3	192.3	181.8	156.3

това частота, f_{CNT}, МГц				

Мікросхеми сімейства MAX3000 виконані по CMOS EpROM технології, при дотриманні технологічних норм 0.35 мкм, що дозволило істотно здешевити їх у порівнянні із сімейством MAX7000S. Усі ПЛІС MAX3000 підтримують технологію програмування в системі (ISp, In-system programmability) і периферійного сканування (boundary scan) у відповідності зі стандартом IEEE Std. 1149.1 JTAG. Елементи вводу-виводу (EBB) дозволяють працювати в системах з рівнями сигналів 5В, 3.3В, 2.5В. Матриця з'єднань має безперервну структуру, що дозволяє реалізувати час затримки поширення сигналу до 4.5 нс. ПЛІС MAX3000 мають можливість апаратної емуляції виходів з відкритим колектором (open - drains pin) і задовольняють вимогам стандарту pCI. Є можливість індивідуального програмування кіл скидання, установки й тактування тригерів, що входять у макроосередок. Передбачено режим зниженого енергоспоживання. Програмувальний логічний додаток дозволяє реалізувати на одному макроосередку функції до 32 змінних. Є можливість задання біта таємності (security bit) для захисту від несанкціонованого тиражування розробки.

Реалізація функції програмування в системі підтримується з використанням стандартних засобів завантаження, таких як ByteBlasterMV, BitBlaster, MasterBlaster, а також підтримується формат JAM.

ПЛІС MAX3000 випускаються в корпусах від 44 до 208 виводів.

На мал. 1 представлена функціональна схема ПЛІС сімейства MAX3000.

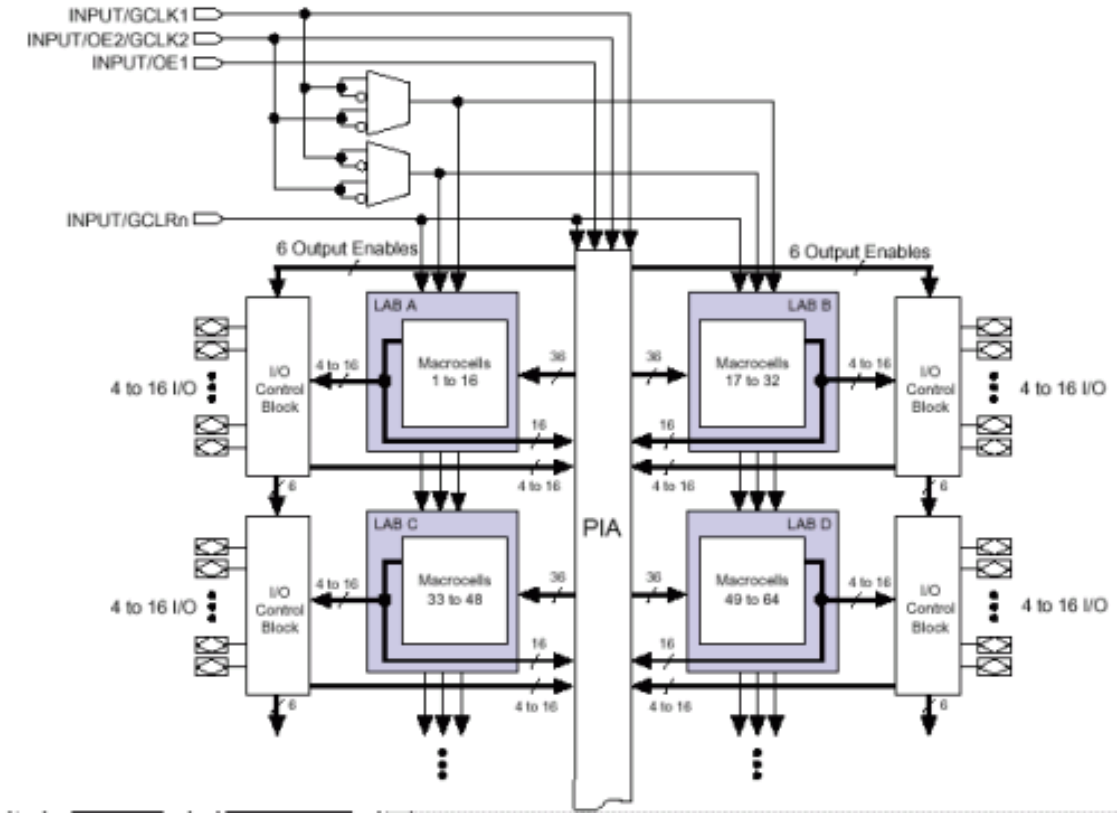

Рис. 2.1. Функціональна схема ПЛІС сімейства MAX3000.

Основними елементами структури ПЛІС сімейства MAX3000 є:

- логічні блоки (ЛБ) (LAB, Logic array blocks);
- макроядра (МЯ) (macrocells);
- логічні розширники (expanders) (паралельний (parallel) і поділюваний (shareble));
- програмувальна матриця з'єднань (ПМС) (programmable interconnect array, pIA);
- елементи вводу-виводу (ЭВВ) (I/O control block).

ПЛІС сімейства MAX3000 мають чотири виводи, закріплених за глобальними ланцюгами (dedicated inputs). Це глобальні ланцюги синхронізації скидання й установки в третій стан кожного макроядра. Крім того, ці виводи можна використати як входи або виходи користувача для "швидких" сигналів, оброблюваних у ПЛІС.

Як видно з рис.2.1 в основі архітектури ПЛІС сімейства MAX3000 лежать логічні блоки складаються з 16 макроосередків кожний. Логічні блоки з'єдну-

ються за допомогою програмувальної матриці з'єднань. Кожен логічний блок має 36 входів із ПМС.

На рис 2. наведено структурну схему макроядра ПЛІС сімейства MAX3000.

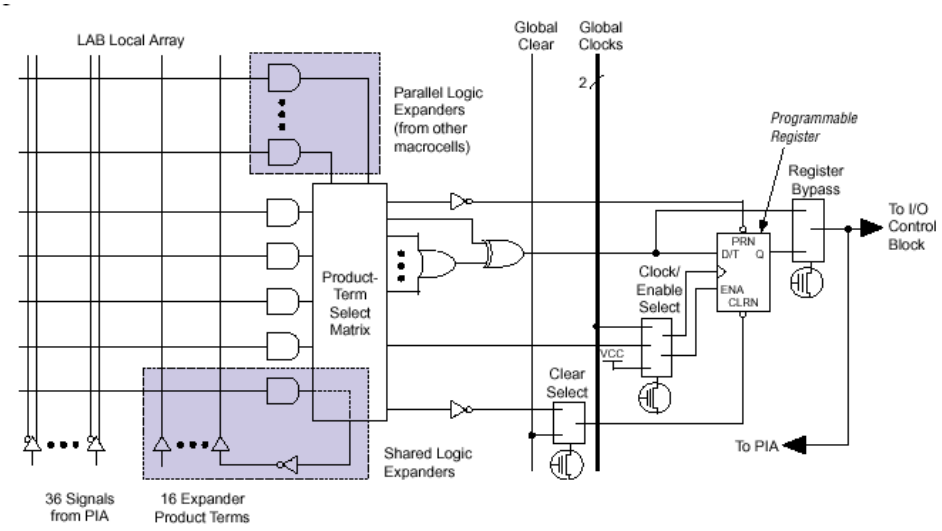

Рис. 2.2. Структурна схема макроядра ПЛІС сімейства MAX3000.

Математичне ядро ПЛІС сімейства MAX3000 складається із трьох основних вузлів:

- локальної програмувальної матриці (LAB local array);
- матриці розподілу термів (product-term select matrix);
- програмувального регістра (programmable register).

Комбінаційні функції реалізуються на локальній програмувальній матриці й матриці розподілу термів, що дозволяє поєднувати логічні добутки або по АБО (OR), або по що виключає АБО (XOR). Крім того, матриця розподілу термів дозволяє зкомутувати коло керування тригером МЯ.

Режим тактування й конфігурація тригера вибираються автоматично під час синтезу проекту в САПР MAX+PLUS II залежно від обраного розроблювачем типу тригера при описі проекту.

У ПЛІС сімейства MAX3000 доступно 2 глобальних тактових сигнали, що дозволяє проектувати схеми із двофазною синхронізацією.

Для реалізації логічних функцій великого числа змінних використаються логічні розширники

Поділюваний логічний розширник (рис. 2.3) дозволяє реалізувати логічну функцію з більшим числом входів, дозволяючи об'єднати МЯ, що входять до складу одного ЛБ. Таким чином, поділюваний розширник формує терм, інверсне значення якого передається матрицею розподілу термів у локальну програмувальну матрицю й може бути використане в будь-який МЯ даного ЛБ. Як видно з мал. 3, є 36 сигналів локальної ПМС, а також 16 інверсних сигналів з поділюваних логічних розширників, що дозволяє в межах одного ЛБ реалізувати функцію до 52 термів рангу 1.

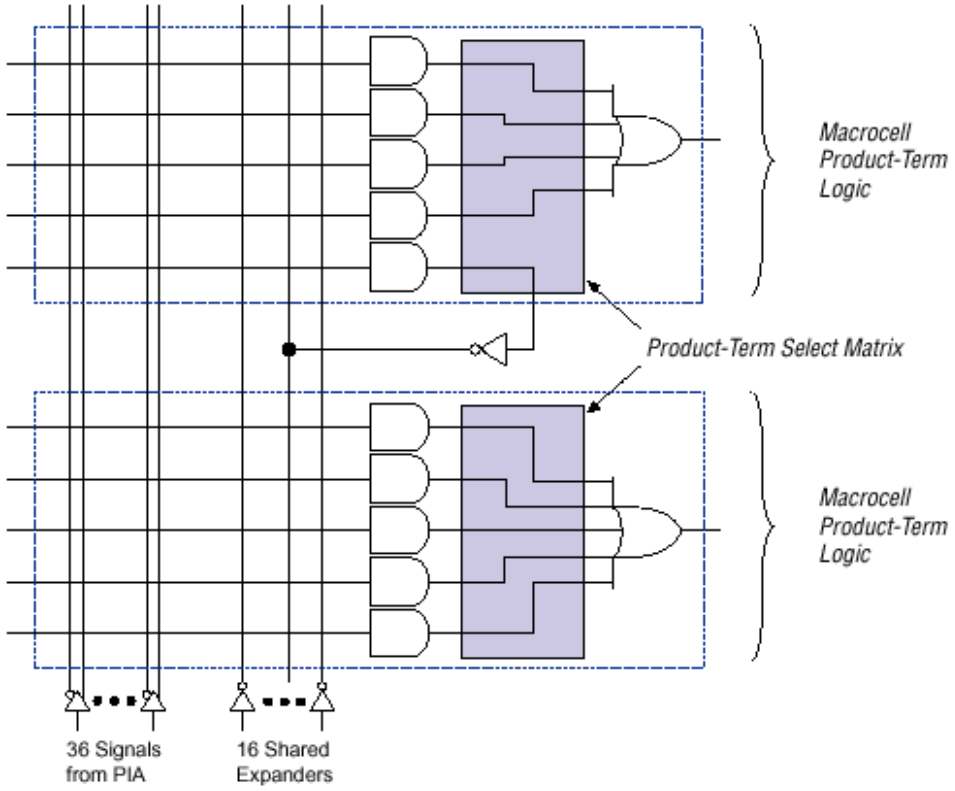

Рис. 2.3. Поділюваний логічний розширювач.

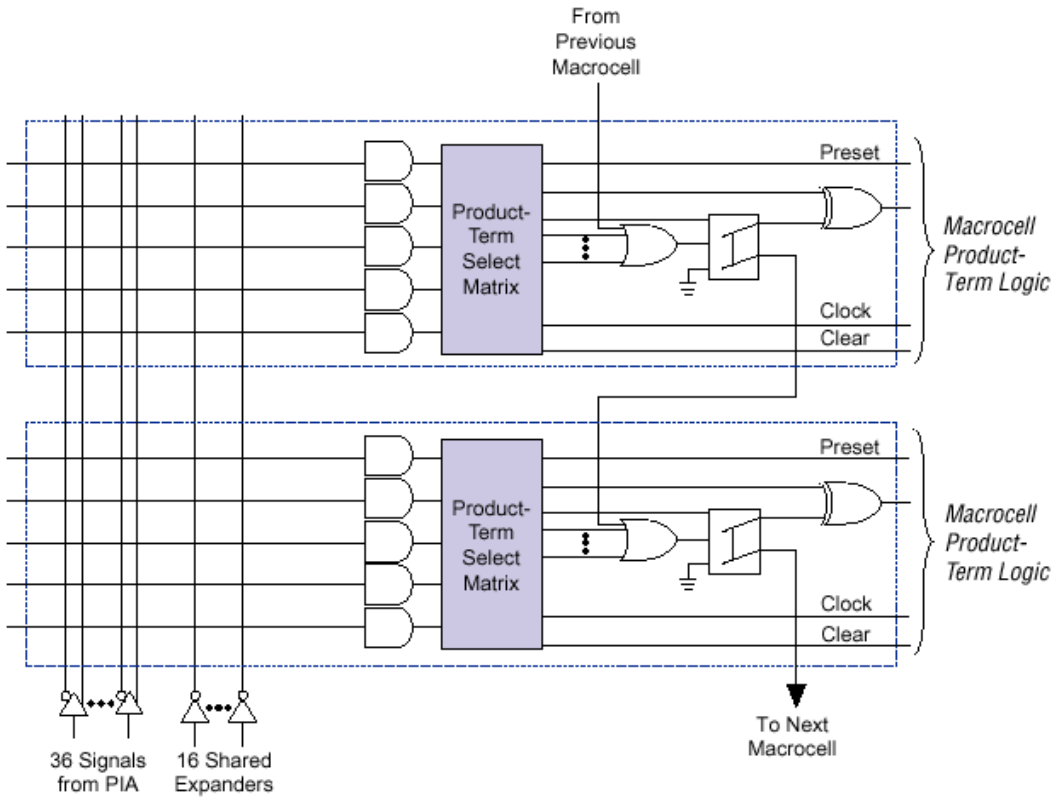

Рис. 2.4. Паралельний логічний розширювач.

Паралельний логічний розширювач (рис. 2.4), дозволяє використати локальні матриці суміжних МЯ для реалізації функцій, у які входять більше 5 термів. Один ланцюжок паралельних розширювачів може включати до 4 МЯ, реалізуючи функцію 20 термів. Компілятор системи MAX+PLUS II підтримує розміщення до 3-х наборів не більш ніж по 5 паралельних розширювачів.

На рис. 2.5 наведена структура програмувальної матриці з'єднань.

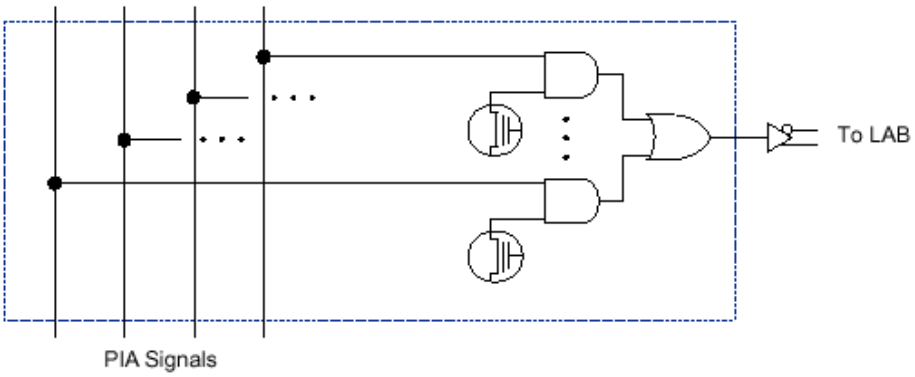

Рис. 2.5. Структура ПМС ПЛІС сімейства MAX3000.

На ПМС виводяться сигнали від всіх можливих джерел: ЗВВ, сигналів зворотного зв'язку ЛБ, спеціалізованих виділених виводів. У процесі програмування тільки необхідні сигнали "заводяться" на кожен ЛБ. На рис. 2.5 наведена структурна схема формування сигналів ЛБ.

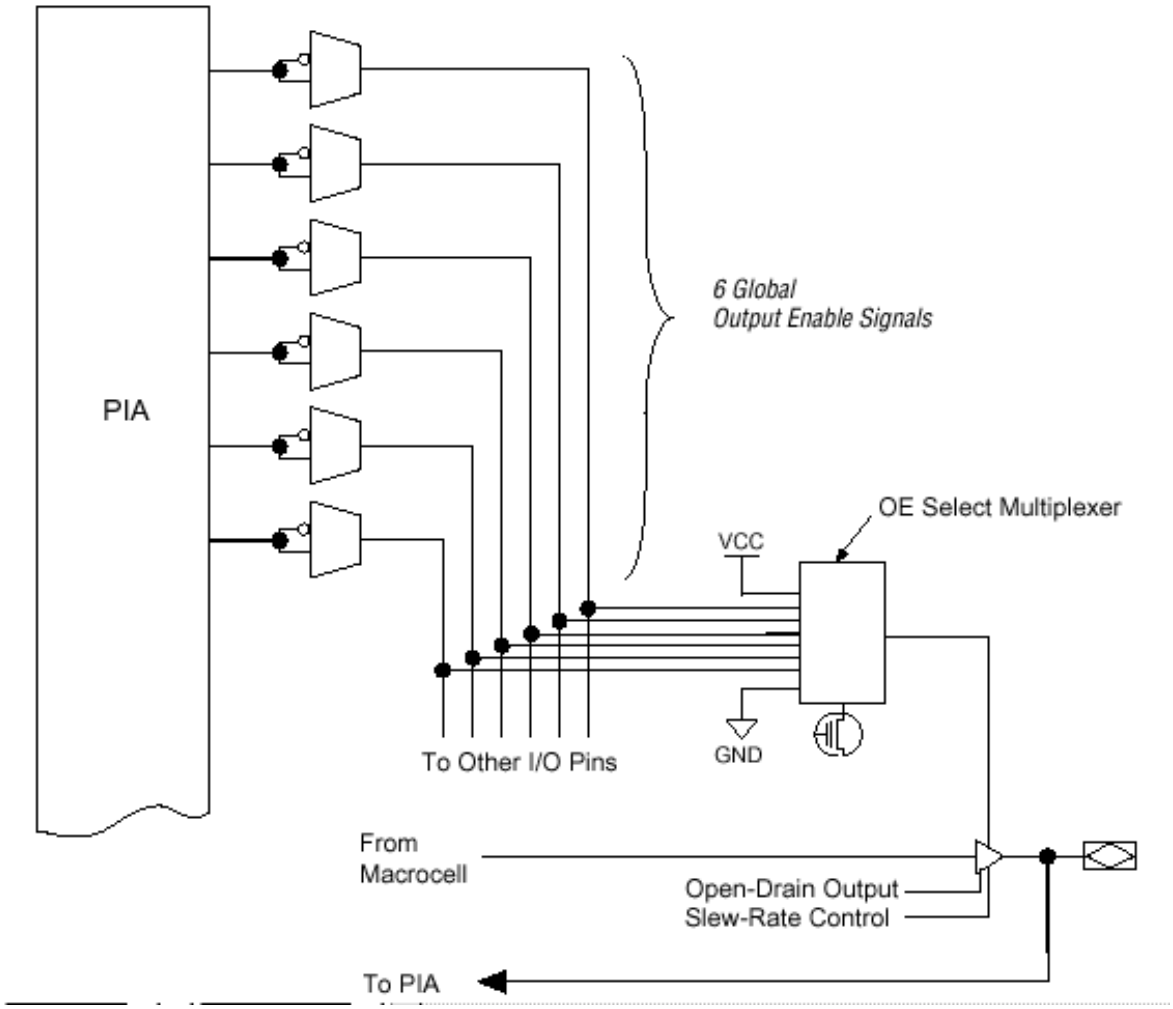

Рис. 2.6. Елемент вводу-виводу.

На рис 6 наведена схема елемента вводу-виводу (ЕВВ) ПЛІС сімейства MAX3000. ЕВВ дозволяє організувати режими роботи з відкритим колектором і третім станом.

Висновки

Поява інтегральних схем типу FPGA зробила можливим втілення основних концепцій Reconfigurable computing в реальні проекти.
Синтез мови апаратних засобів HDL з інструментальними засобами дозволив автоматизувати процес проектування.

Процес розробки проекту істотно скорочується з використанням System Generator для DSP, який є програмним інструментальним засобом для проектування, моделювання та створення систем DSP на базі FPGA.

Технології IRL є основою для створення нових продуктів, які можуть динамічно модифікуватися як за допомогою програмного забезпечення, так і за допомогою алгоритму, заданого замовником. Комбінація трьох фундаментальних технологій - всеохоплююче використання мережних технологій, технологія Java та кристали Virtex дозволяє виконувати проектування та розробку зовсім нових IRL-продуктів. Такі продукти можуть поновлюватися та модифікуватися через мережу Internet і розширюватися новими можливостями і особливостями.

Здатність розробляти інтегральні схеми в середовищі програмування, з можливістю швидкого компілювання і налагодження змінює традиційний спосіб проектування FPGA з використанням інструментальних засобів автоматизованого проектування. Технологія Java забезпечує швидкий цикл компіляції, виконання та перевірки програм. Засоби розробки, що містять JVM, забезпечують контроль програмних застосувань на стадії виконання. Завдяки наявності JVM досягається мобільність мови Java, забезпечується абстрагованість компільованих Java-програм від апаратної платформи й операційної системи.

Використання мови Java має багато переваг над традиційними статичними методологіями проектування. Серед них - перевірка проекту на рівні двійкових потоків, гнучкість при роботі у середовищі моделювання і налагодження проекту, взаємодія з JBits API [13].

JBits API, на відміну від традиційних мов апаратних описів, підтримує RTR, забезпечуючи можливість швидкого створення і реконфігурації схем Xilinx

типу FPGA серії Virtex під час виконання програмного застосування за допомогою прямого доступу до конфігурації двійкового потоку.

JBits API та програмне забезпечення, яке входить до його складу, BoardScope, XHWIF та Device Simulator надають нову методологію для проектування, розробки і перевірки інтегральних схем. Можливості й особливості XHWIFServer і BoardScope полегшують віддалену розробку і налагодження проекту. Крім того, система JBits може використовуватись як автономний інструмент або як база для створення інших інструментів.

Спираючись на виконаний вище аналіз сучасного стану ключових розділів Reconfigurable computing, можна сказати, що реконфігурована обробка даних, сучасні ПЛІС і розроблені на їх базі засоби обчислювальної техніки представляють собою останні досягнення в області комп'ютерної техніки та мікроелектроніки.