

Національний університет "Львівська політехніка"
Кафедра електронних обчислювальних машин (ЕОМ)

Автоматизоване проектування комп'ютерних та кіберфізичних систем

спеціальність 123 "Комп'ютерна інженерія"
спеціалізація 123.04 "Кіберфізичні системи"
4-ий курс

Лекція 3. Високорівневі засоби системного проектування (1)

2021

Лекція 3. Високорівневі засоби системного проектування (1)

01. Системний рівень проектування(ESL)
та моделювання рівня транзакцій(TLM).
02. Засоби системного рівня проектування(ESL)
та моделювання рівня транзакцій(TLM).
03. Високорівневий синтез(HLS).
04. Засоби HLS. Використання Celoxica Agility Compiler.
05. Засоби HLS. Сумісне використання Celoxica Agility Compiler
та Xilinx ISE WebPack.
06. Сумісне проектування апаратури і програм
(*hardware/software co-design*)
07. Засоби сумісного проектування апаратури і програм.
08. Застосування MATLAB для системного проектування.

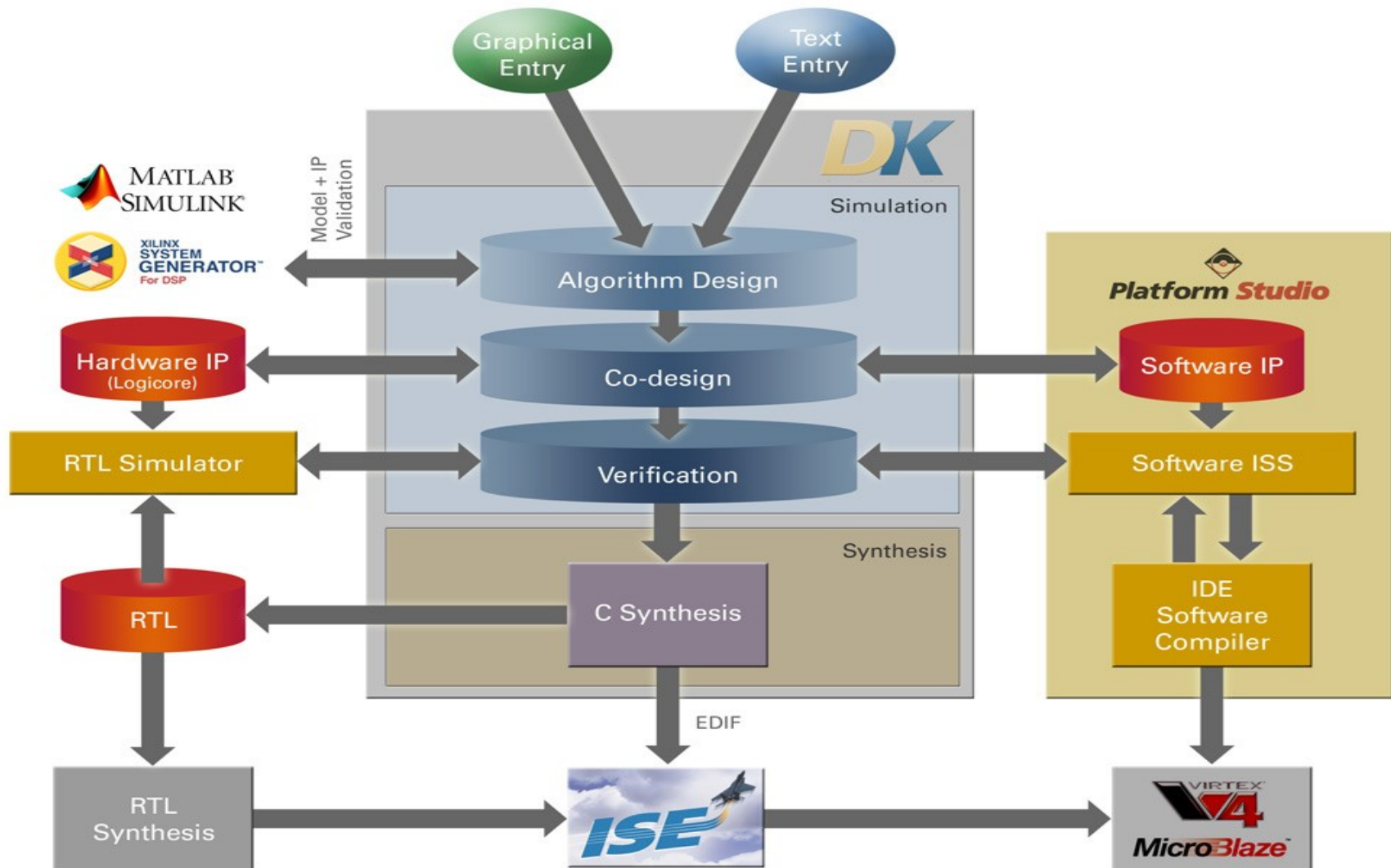
01.1. Системний рівень проектування(ESL) та моделювання рівня транзакцій(TLM).

Electronic System Level(ESL)

Засоби системного проектування електронних систем, які базуються на Сі-подібному синтаксисі, з можливістю моделювання і автоматизованим синтезом є поширеними при проектуванні сучасних цифрових систем. Вони дозволяють прискорити розробку складних електронних продуктів, якими наповнена сучасна економіка.

Популярними цільовими пристроями поведінкового дизайну і синтезу, окрім VLSI, є ПЛІС (Field Programmable Gate Array, FPGA), спеціалізовані по застосуваннях інтегральні схеми ASIC і системи на кристалі (System on Chip, SoC).

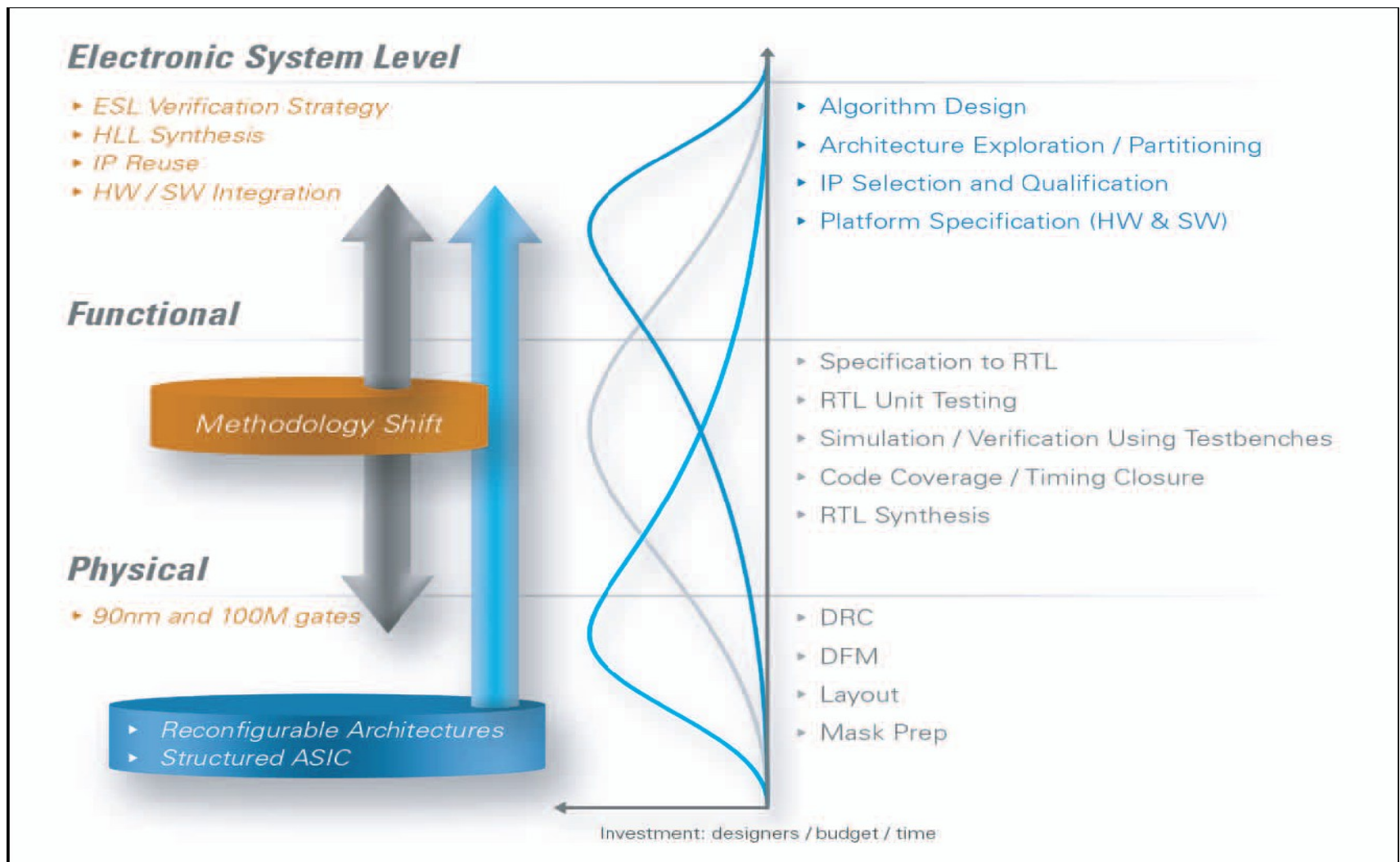
01.2. Системний рівень проектування(ESL) та моделювання рівня транзакцій(TLM).



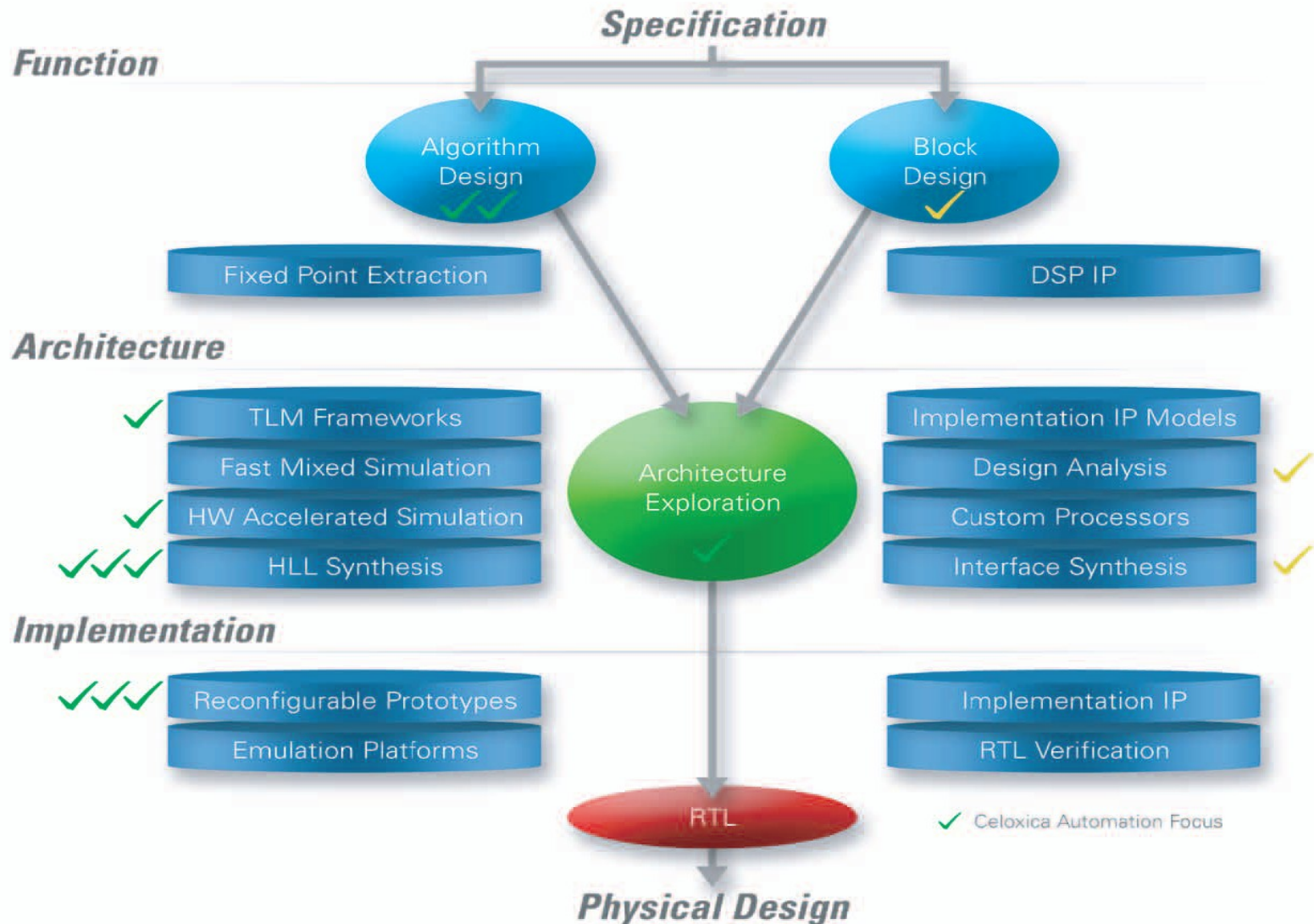
01.3. Системний рівень проектування(ESL) та моделювання рівня транзакцій(TLM).

ISS (Instruction Stream Simulator) це симулятор програмної частини проекту. RTL (Register Transfer Level) є симулятором апаратної частини проекту. Virtex-4 MicroBlaze є софт-ядром 32-х бітового процесора, що імплементується до ПЛІС Virtex-4. З імплементуванням до ПЛІС процесора MicroBlaze отримуємо комп'ютерну систему. Вона містить апаратну і програмну частини. Програмну частину опрацьовує апаратне обладнання на основі MicroBlaze (софт-процесор плюс пам'ять). Апаратна частина, як правило, містить апаратні прискорювачі (наприклад, апаратуру виконання операцій рухомої коми, цифрові фільтри тощо), що спільно працюють функціонують з програмою частиною.

01.4. Системний рівень проектування(ESL) та моделювання рівня транзакцій(TLM).



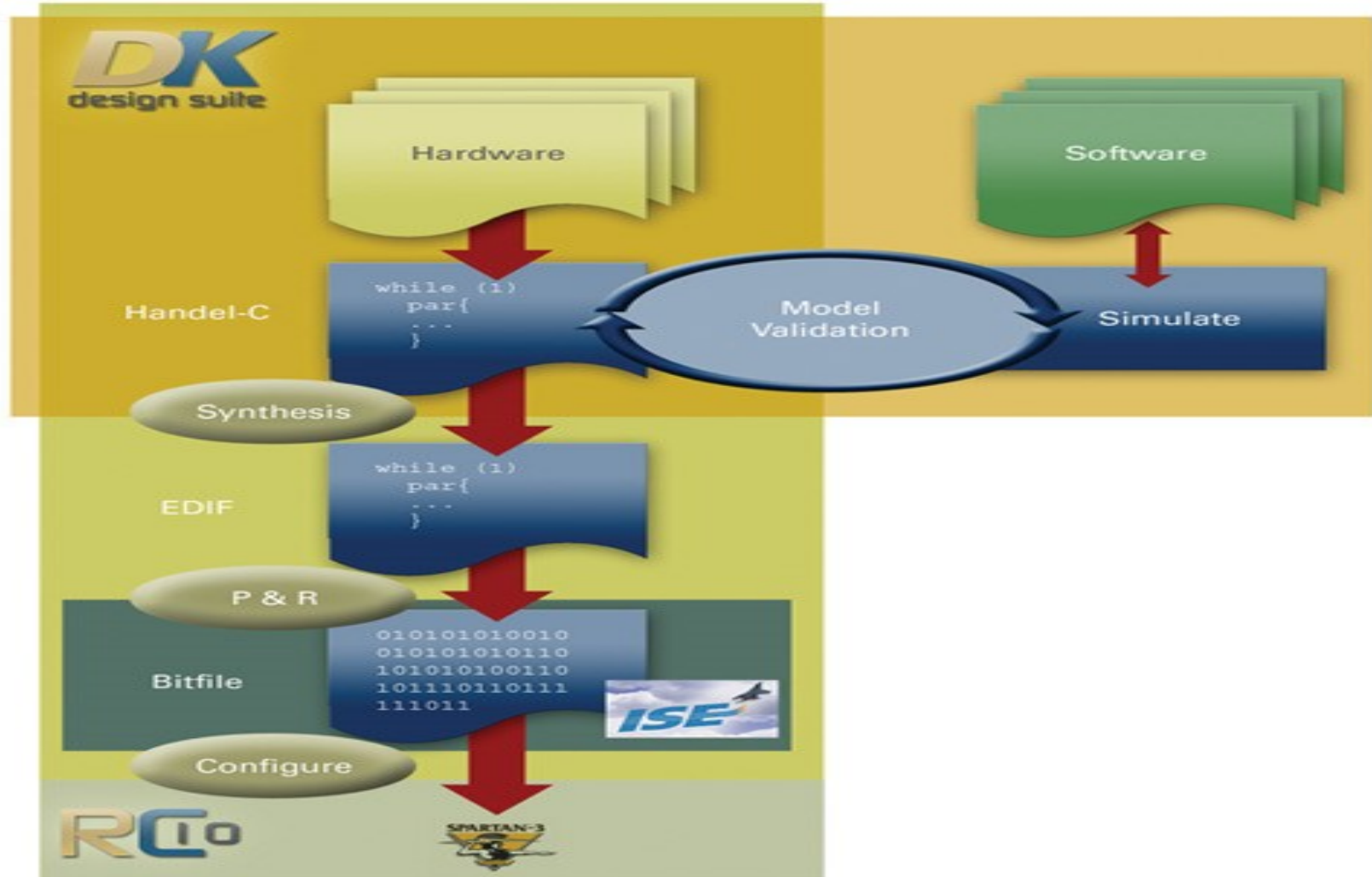
01.5. Системний рівень проектування(ESL) та моделювання рівня транзакцій(TLM).



01.6. Системний рівень проектування(ESL) та моделювання рівня транзакцій(TLM).

Складність, особливо на найвищому рівні проектування за допомогою незалежних від фізичного рівня специфікацій та моделювання, спонукала до появи методів моделювання рівня транзакцій (***Transaction Level Modeling — TLM***). TLM використовує код на основі C, щоб описати з достатньою кількістю деталей апаратне та програмне забезпечення. Це відокремлює функціональні одиниці від комунікації, абстрагує інтерфейси з акцентом на функціональність передачі даних.

01.7. Системний рівень проектування(ESL) та моделювання рівня транзакцій(TLM).



01.8. Системний рівень проектування(ESL) та моделювання рівня транзакцій(TLM).

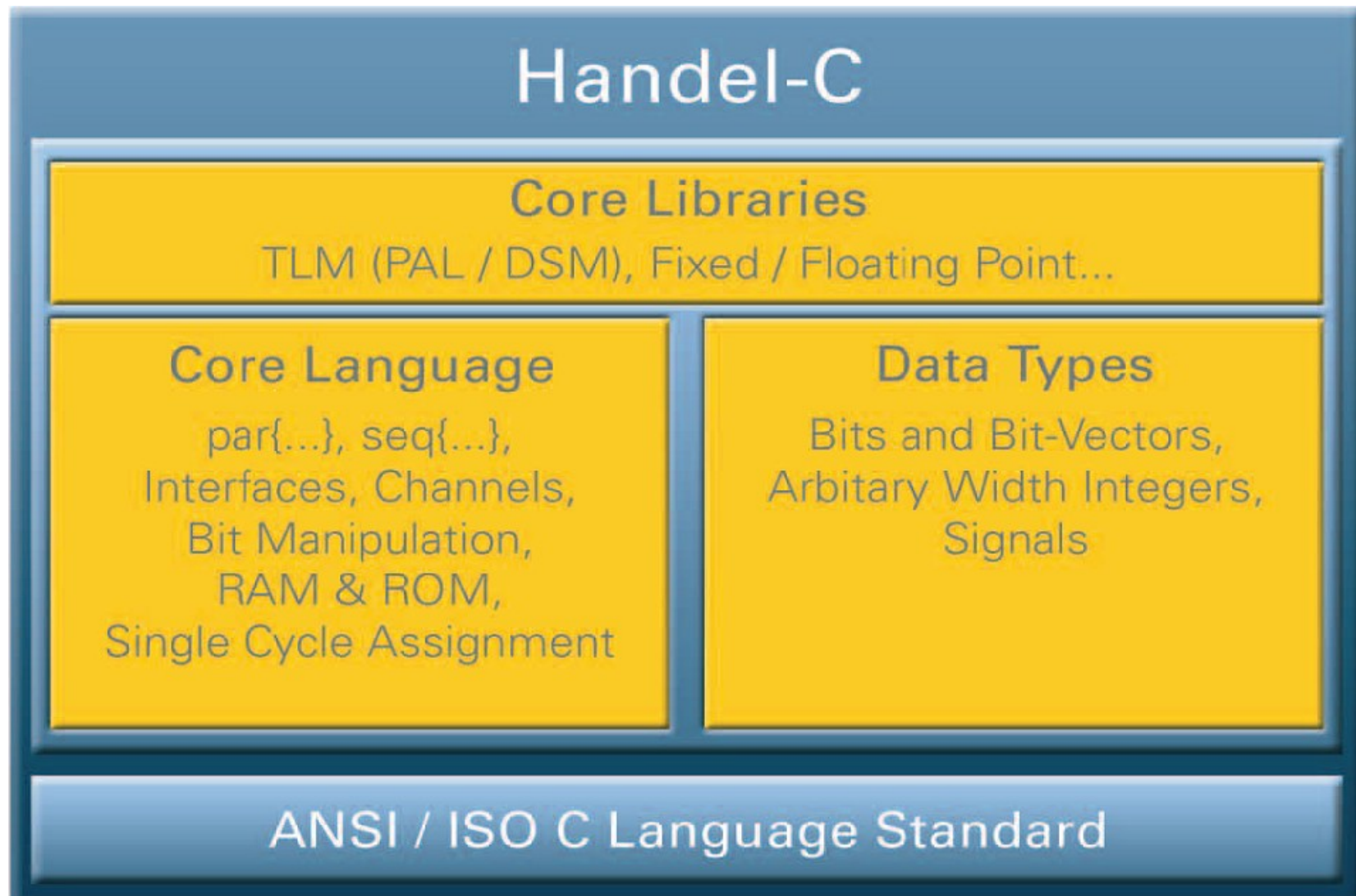
Мови на основі C у поєднанні з блоковими підходами є найпопулярнішим методом опису проектування ESL та моделей рівня транзакцій (TLM).

Handel-C та SystemC – популярні мови на основі C для проектування та синтезу ESL.

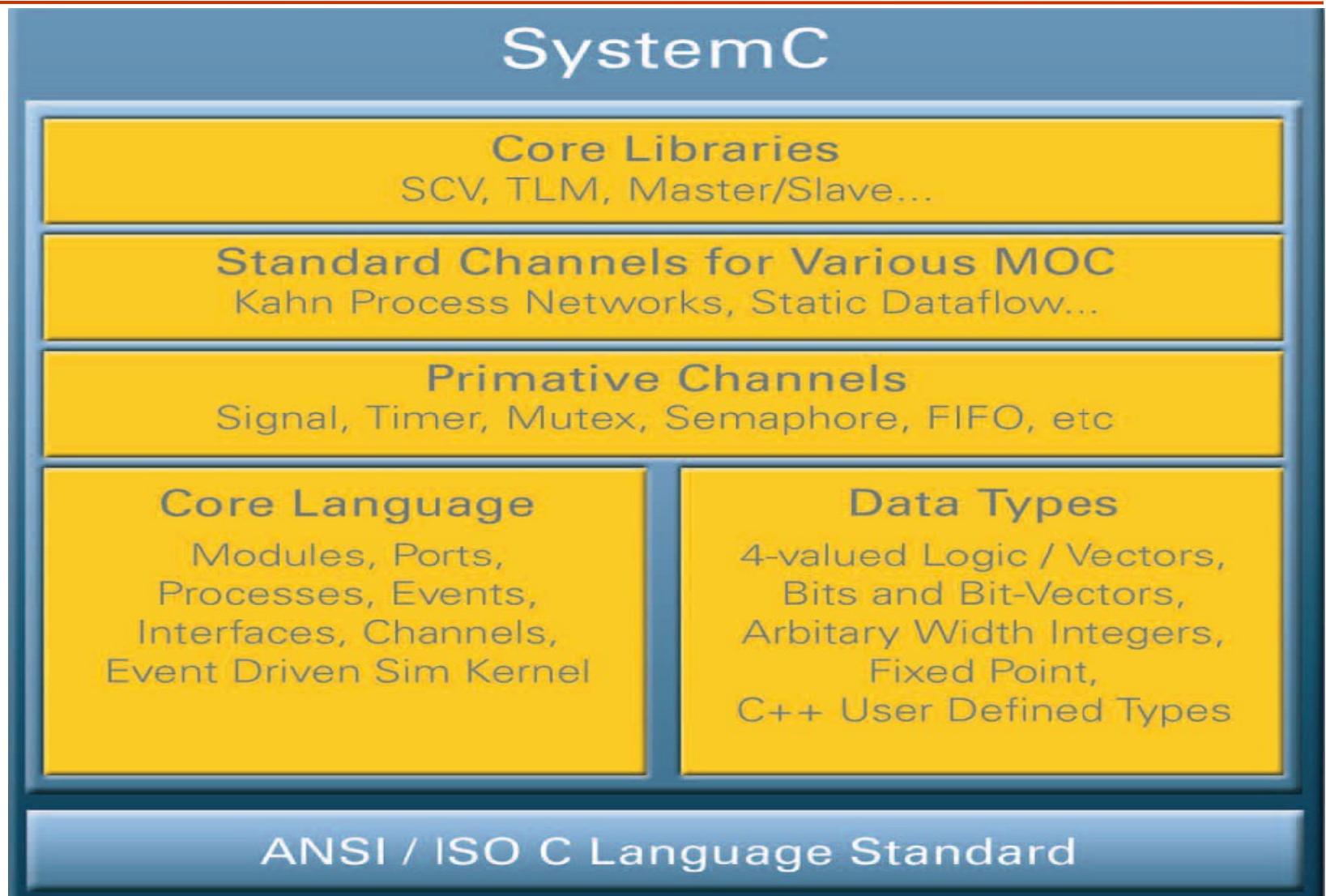
Handel-C, заснована на стандарті ANSI-C, – мова високого рівня, яка додає мінімальну кількість простих апаратно-орієнтованих конструкцій для реалізації дизайну.

SystemC – це бібліотека класів для C++, яка визначає синтаксис опису обладнання для моделей C++.

01.9. Системний рівень проектування(ESL) та моделювання рівня транзакцій(TLM).



01.10. Системний рівень проектування(ESL) та моделювання рівня транзакцій(TLM).

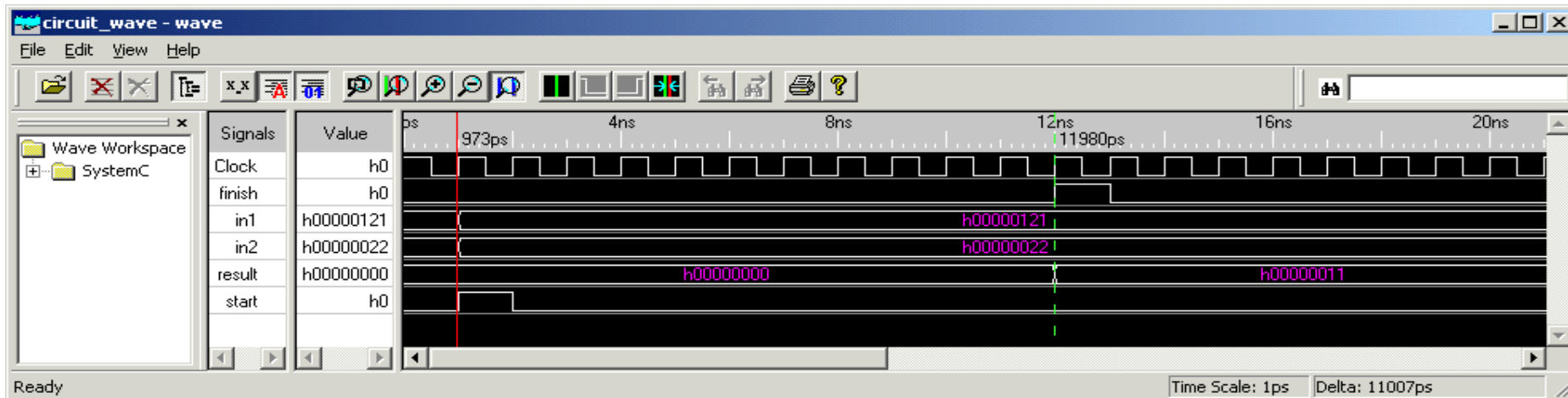


02.1. Засоби системного рівня проектування(ESL) та моделювання рівня транзакцій(TLM).

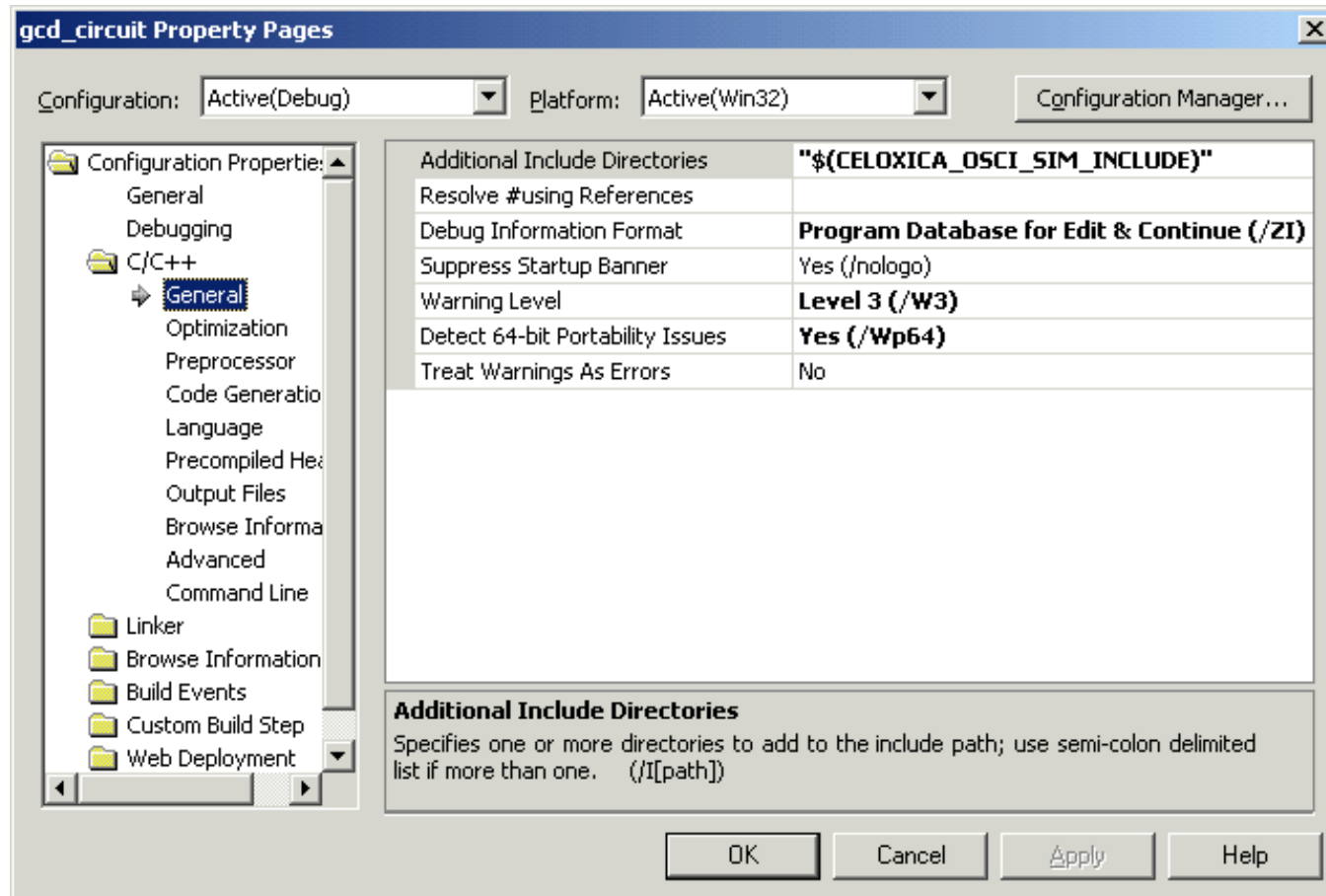
При ELS-проектуванні для TLM-моделювання часто використовують спеціалізовані засоби розробки, але(приміром у випадку використання SystemC) можна застосовувати довільне середовище розробки програмного забезпечення, яке підтримує відповідну мову програмування(у випадку SystemC цією мовою є C++).

Для налаштування середовища програмування потрібно підключити відповідні бібліотеки.

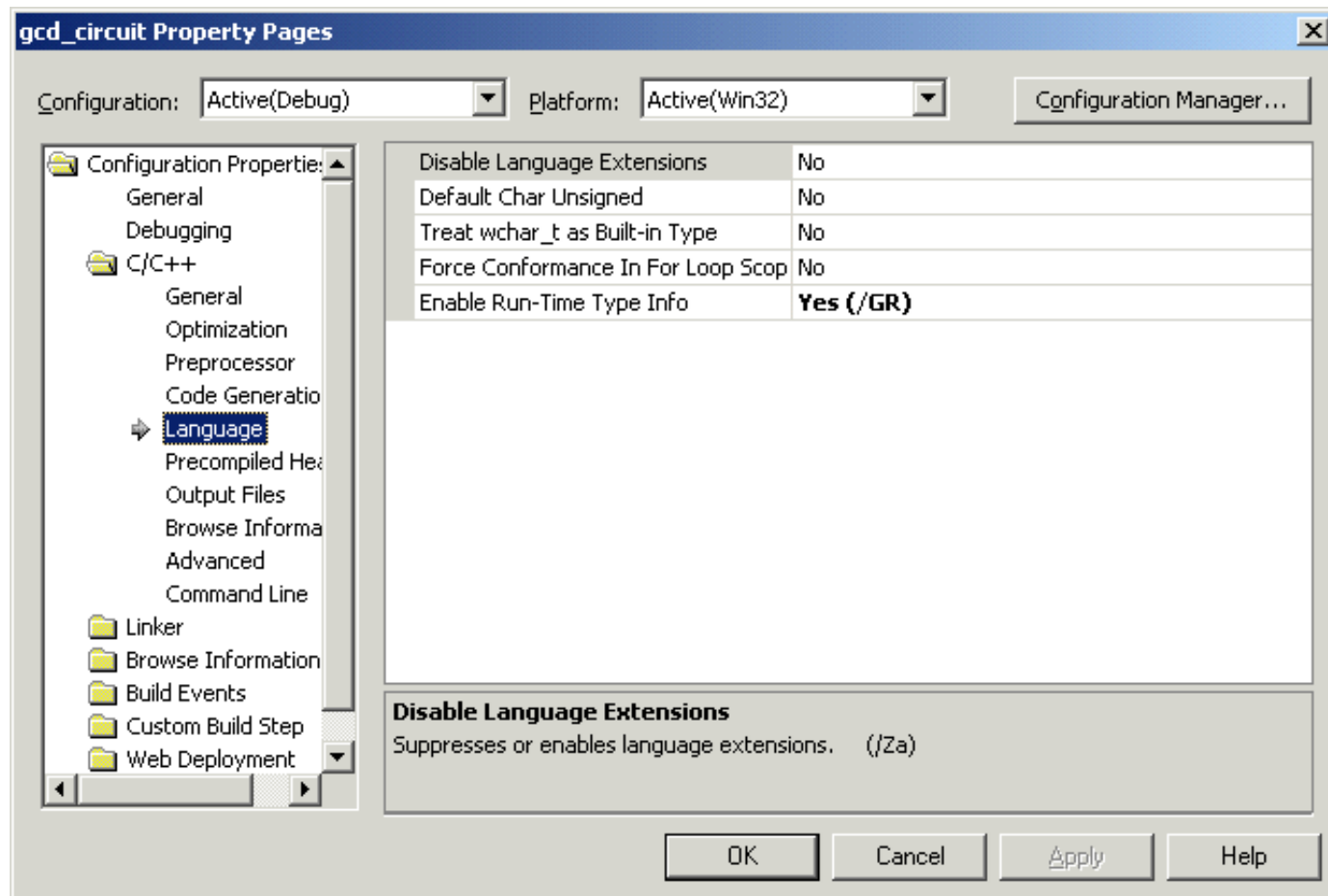
Для більшої зручності часто використовується додаткове програмне забезпечення(наприклад для відображення часових діаграм сигналів).



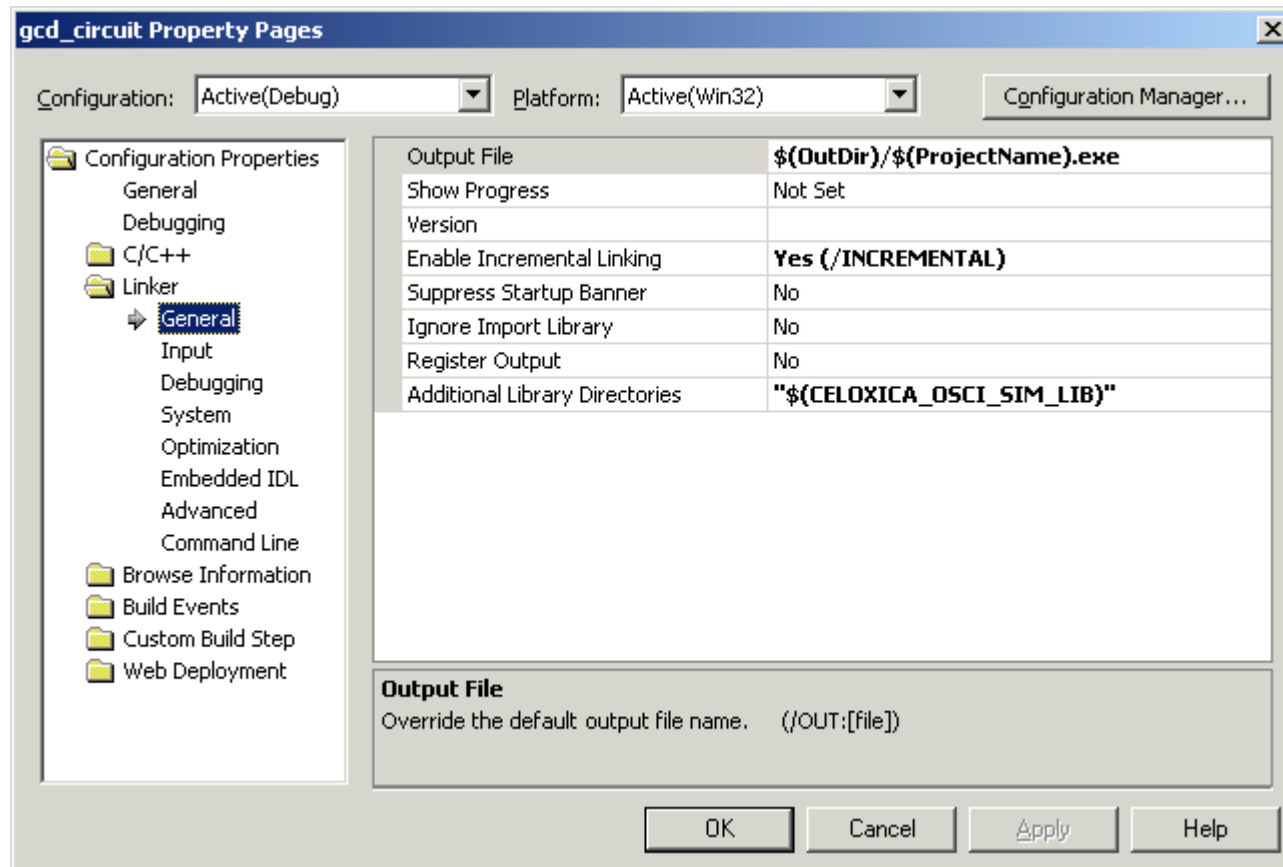
02.2. Засоби системного рівня проектування(ESL) та моделювання рівня транзакцій(TLM).



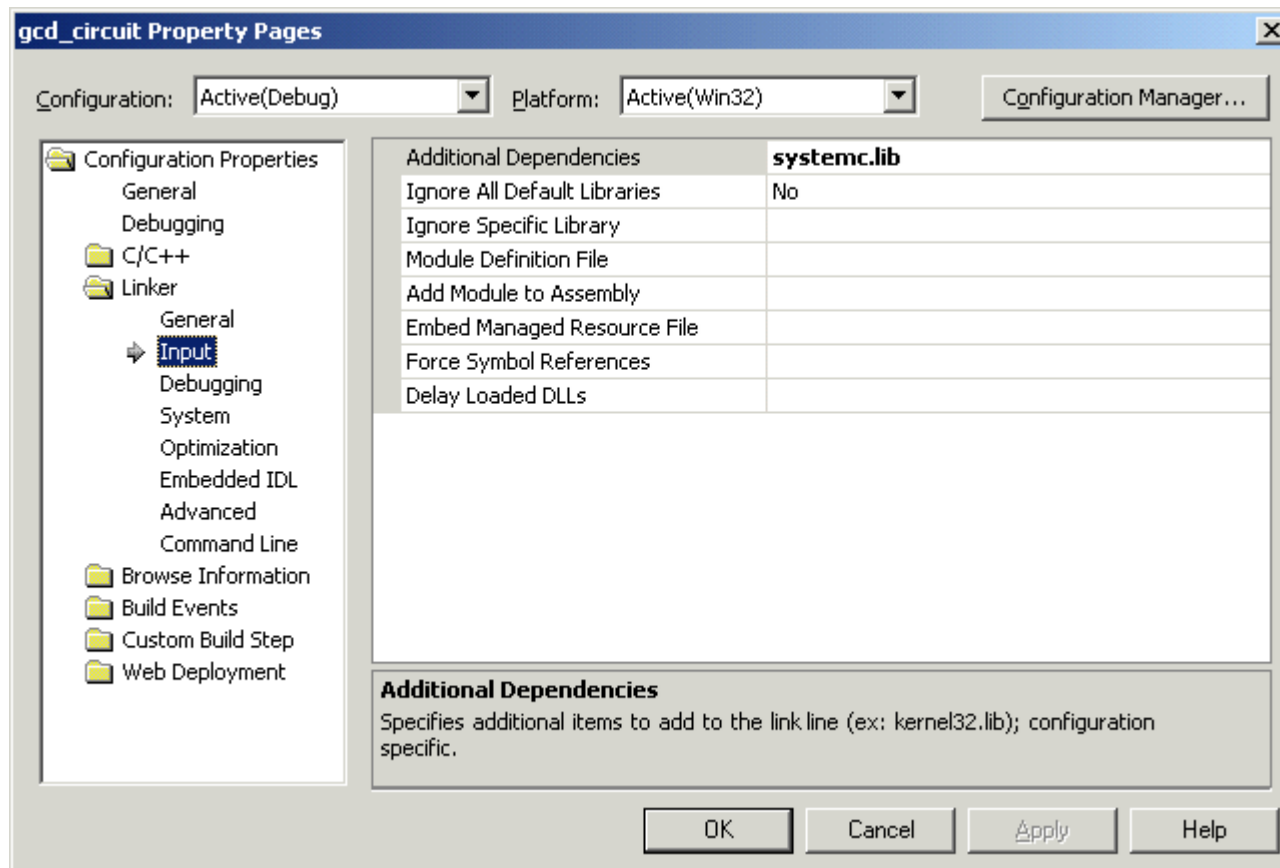
02.3. Засоби системного рівня проектування(ESL) та моделювання рівня транзакцій(TLM).



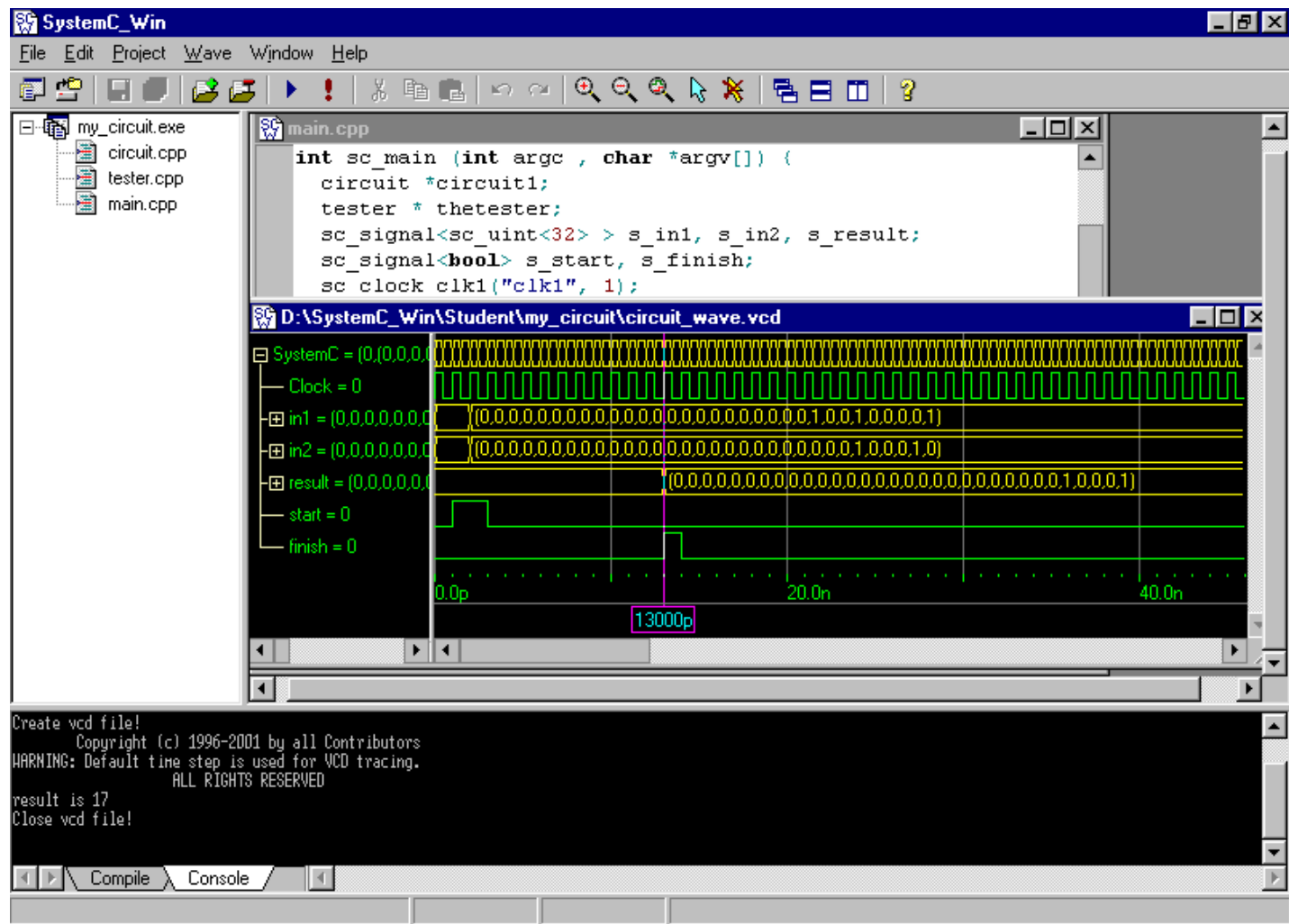
02.4. Засоби системного рівня проектування(ESL) та моделювання рівня транзакцій(TLM).



02.5. Засоби системного рівня проектування(ESL) та моделювання рівня транзакцій(TLM).



02.6. Засоби системного рівня проектування(ESL) та моделювання рівня транзакцій(TLM).



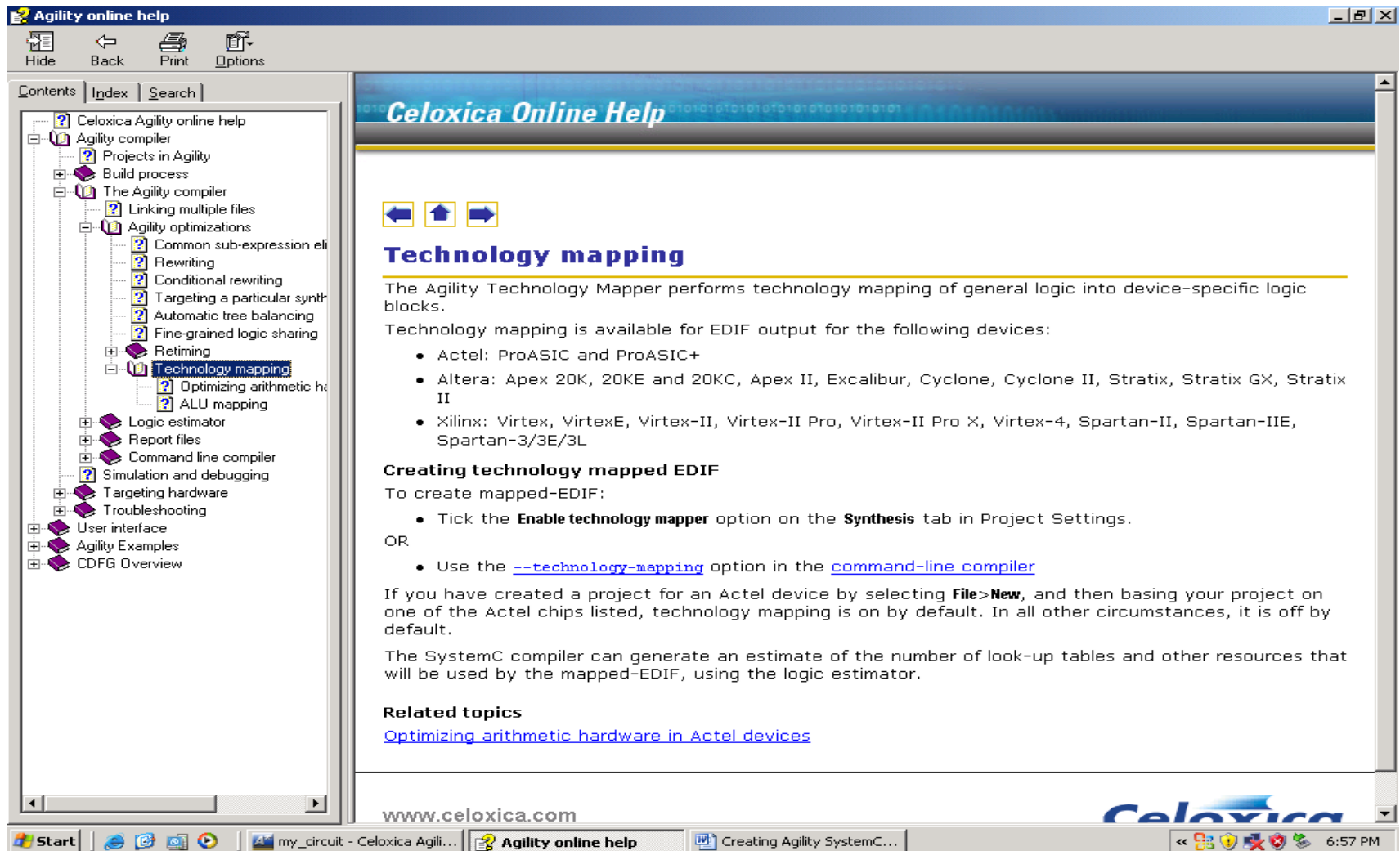
03.1. Високорівневий синтез(HLS)

Синтез високого рівня (HLS) – автоматизований процес проектування, який приймає абстрактну поведінкову специфікацію цифрової системи та будує RTL-опис системи. Синтез високого рівня працює на більш високому рівні абстракції(ніж RTL), починаючи з алгоритмічного опису на мовах високого рівня, таких як SystemC та ANSI C/C++. Дизайнер зазвичай розробляє функціональні можливості модуля та протокол взаємозв'язку. Інструменти синтезу перетворюють з високого рівня функціональний код у повністю синхронізовані RTL-моделі, які потім використовуються безпосередньо у логічному синтезі для створення реалізації на рівні логічних вентилів.

04.1. Засоби HLS. Застосування Celoxica Agility Compiler.

Agility Compiler для SystemC є рішенням для проектування рівня електронної системи (ESL). Він пов'язує ESL із імплементацією у FPGA та ефективно відображає модель при проектуванні SoC/ASIC. Компілятор Agility дозволяє моделям рівня транзакцій (TLM), описаним у SystemC, впроваджуватись у реальний кристал на самому початку проектування. Компілятор Agility дозволяє проектувати складні системи швидше, з меншим ризиком та з меншими витратами.

04.2. Засоби HLS. Застосування Celoxica Agility Compiler.



The screenshot shows the 'Agility online help' window. The left sidebar contains a tree view with the following structure:

- Celoxica Agility online help
 - Agility compiler
 - Projects in Agility
 - Build process
 - The Agility compiler
 - Linking multiple files
 - Agility optimizations
 - Common sub-expression eli
 - Rewriting
 - Conditional rewriting
 - Targeting a particular synth
 - Automatic tree balancing
 - Fine-grained logic sharing
 - Retiming
 - Technology mapping**
 - Optimizing arithmetic h
 - ALU mapping
 - Logic estimator
 - Report files
 - Command line compiler
 - Simulation and debugging
 - Targeting hardware
 - Troubleshooting
- User interface
- Agility Examples
- CDFG Overview

The main content area displays the 'Technology mapping' section. It includes navigation arrows, a title, a description, a list of supported devices, and instructions on how to create technology mapped EDIF.

Technology mapping

The Agility Technology Mapper performs technology mapping of general logic into device-specific logic blocks.

Technology mapping is available for EDIF output for the following devices:

- Actel: ProASIC and ProASIC+
- Altera: Apex 20K, 20KE and 20KC, Apex II, Excalibur, Cyclone, Cyclone II, Stratix, Stratix GX, Stratix II
- Xilinx: Virtex, VirtexE, Virtex-II, Virtex-II Pro, Virtex-II Pro X, Virtex-4, Spartan-II, Spartan-IIE, Spartan-3/3E/3L

Creating technology mapped EDIF

To create mapped-EDIF:

- Tick the **Enable technology mapper** option on the **Synthesis** tab in Project Settings.

OR

- Use the `--technology-mapping` option in the [command-line compiler](#)

If you have created a project for an Actel device by selecting **File>New**, and then basing your project on one of the Actel chips listed, technology mapping is on by default. In all other circumstances, it is off by default.

The SystemC compiler can generate an estimate of the number of look-up tables and other resources that will be used by the mapped-EDIF, using the logic estimator.

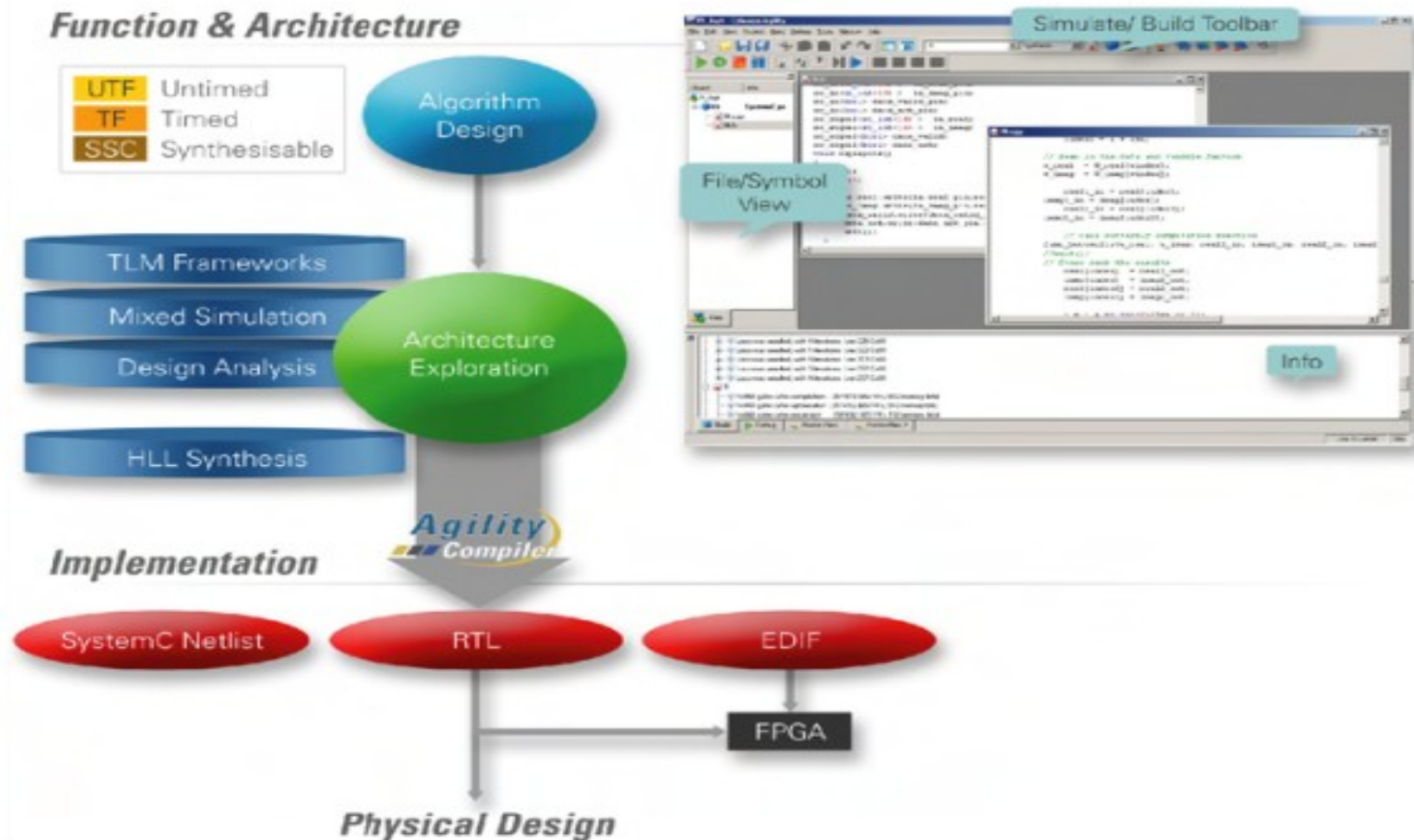
Related topics

[Optimizing arithmetic hardware in Actel devices](#)

www.celoxica.com

Start | my_circuit - Celoxica Agili... | Agility online help | Creating Agility SystemC... | 6:57 PM

04.3. Засоби HLS. Застосування Celoxica Agility Compiler.



05.1. Засоби HLS. Сумісне використання Celoxica Agility Compiler та Xilinx ISE WebPack.

Існує можливість “перехоплення” процесу проектування на себе від засобів автоматичного синтезу. Після цього відбувається керований людиною, неавтоматичний синтез ПЛІС засобами WebPack відповідно до цільового проекту. Перехоплення дозволяє ще раз верифікувати проект, але вже не на рівні мови Сі, а на звичному нам рівні апаратного проектування. В ручному запуску САПР WebPack до САПР завантажують згенерований Agility Compiler EDIF-файл, для якого знадобиться ще “обгортка” (wrapper) у даному середовищі.

05.2. Засоби HLS. Сумісне використання Celoxica Agility Compiler та Xilinx ISE WebPack.

Вікно навігатора проектів Xilinx ISE WebPack з завантаженим вручну результатом роботи Celoxica Agility Compiler

The screenshot shows the Xilinx ISE WebPack interface with the 'FPGA Design Summary' window open. The design is 'agility_probe_2' for target device 'xc2v40-4cs144'. The summary includes project status, partition information, and device utilization details.

AGILITY_PROBE_2 Project Status

| | | | |
|-------------------------|---------------------|-----------------------|----------------------------|
| Project File: | agility_probe_2.isc | Current State: | Programming File Generated |
| Module Name: | circuit_8 | Errors: | No Errors |
| Target Device: | xc2v40-4cs144 | Warnings: | 3 Warnings |
| Product Version: | ISE 8.2i | Updated: | Sat Dec 9 16:48:32 2006 |

AGILITY_PROBE_2 Partition Summary

No partition information was found.

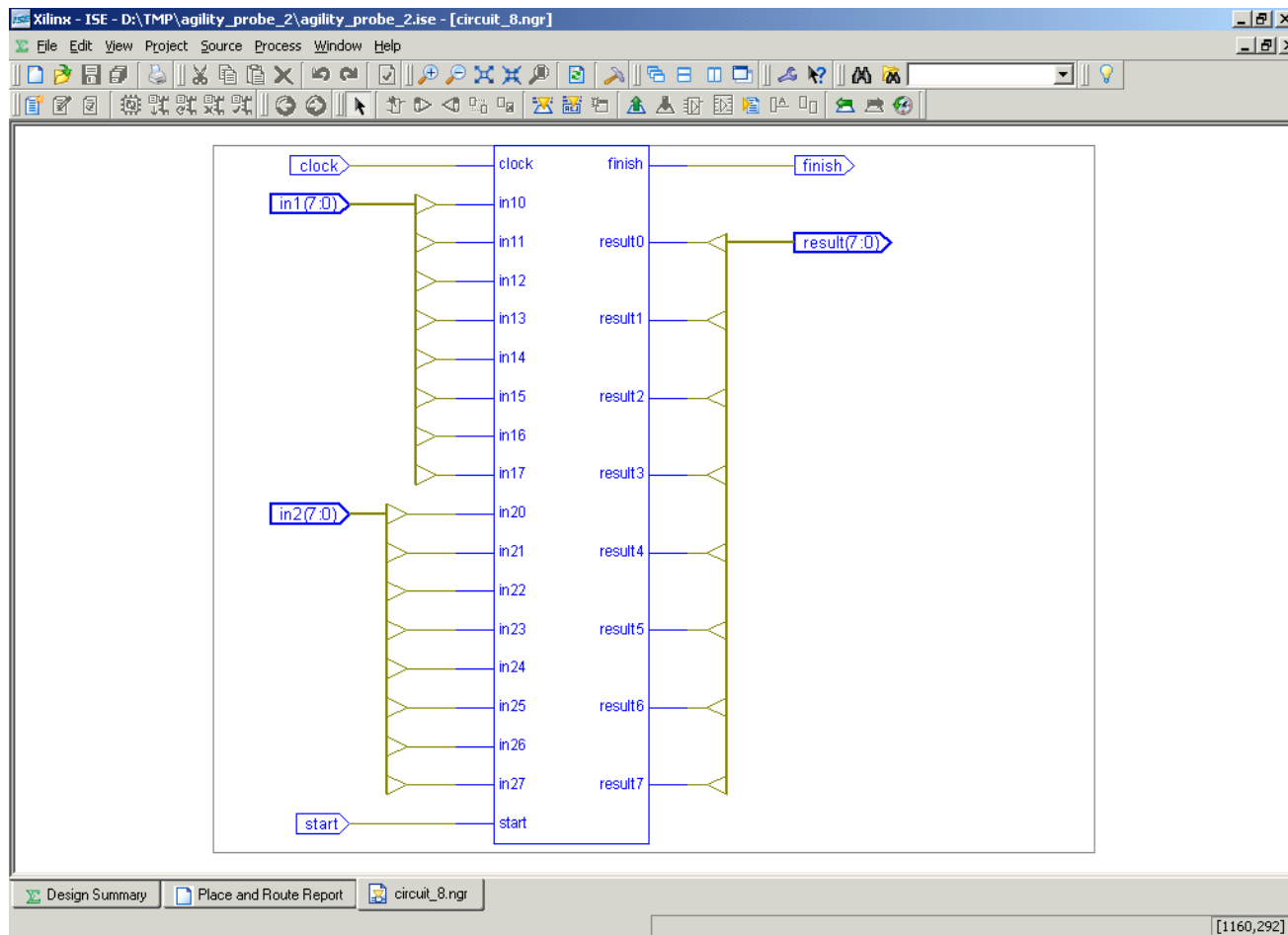
Device Utilization Summary

| Logic Utilization | Used | Available | Utilization | Note(s) |
|--|-----------|------------|-------------|---------|
| Number of Slice Flip Flops | 33 | 512 | 6% | |
| Number of 4 input LUTs | 78 | 512 | 15% | |
| Logic Distribution | | | | |
| Number of occupied Slices | 61 | 256 | 23% | |
| Number of Slices containing only related logic | 61 | 61 | 100% | |
| Number of Slices containing unrelated logic | 0 | 61 | 0% | |
| Total Number 4 input LUTs | 92 | 512 | 17% | |

WARNING: HDLParsers:3215 - Unit work/glbl1 is now defined in a different file: was D:/TMP/agility_probe_2/netgen/par/circuit_8

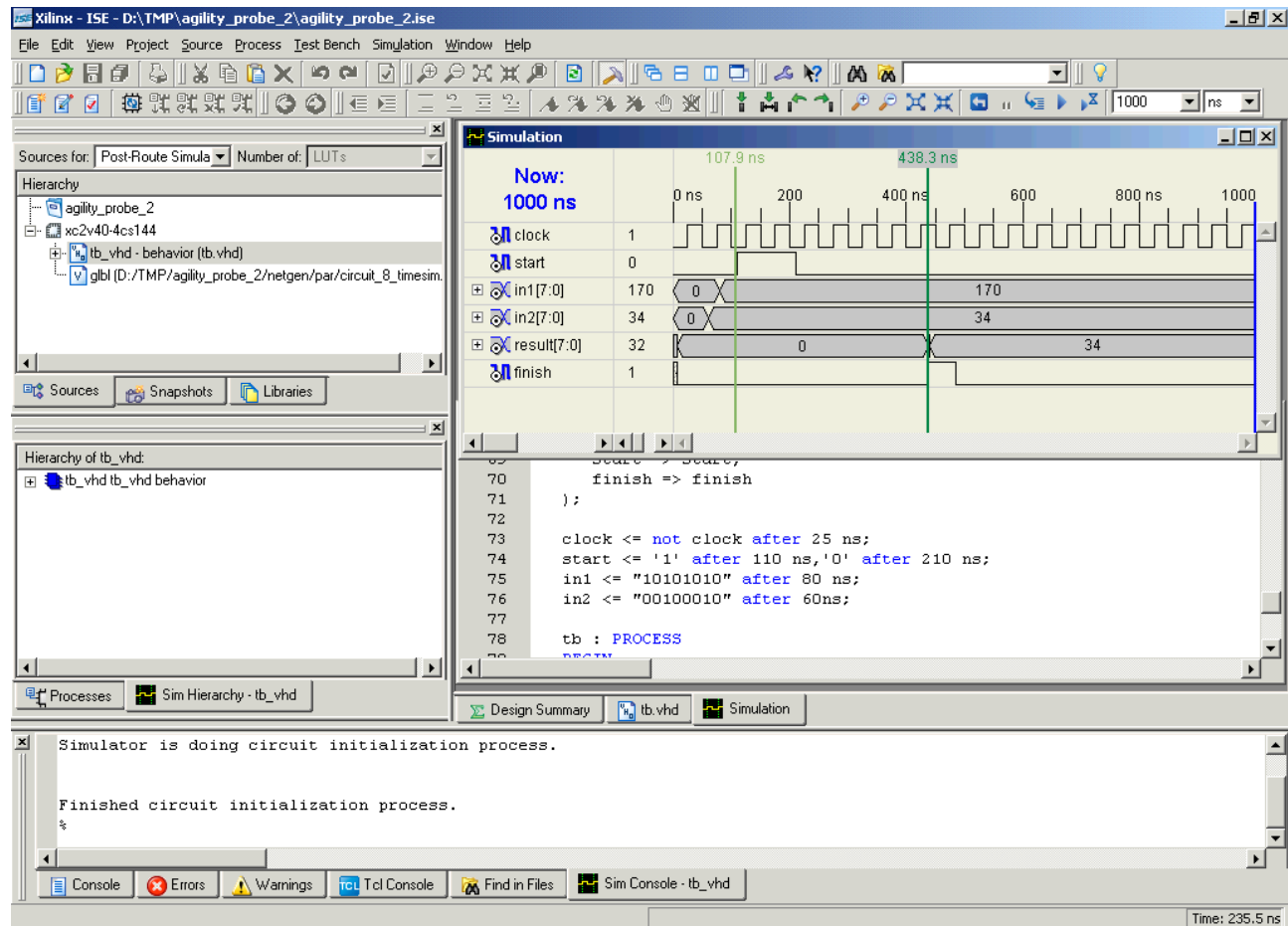
05.3. Засоби HLS. Сумісне використання Celoxica Agility Compiler та Xilinx ISE WebPack.

Інтерфейс синтезованої вручну мікросхеми (САПР ISE WebPack)



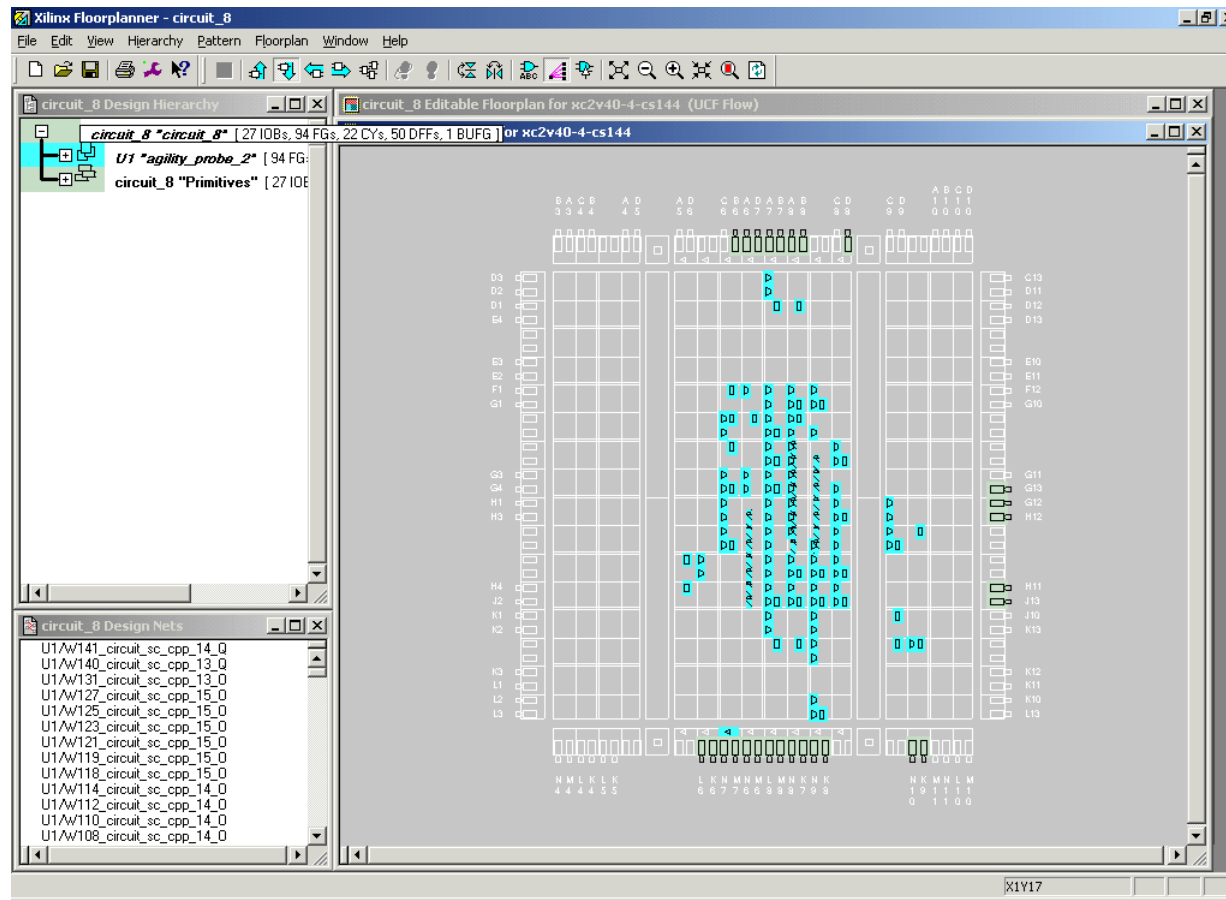
05.4. Засоби HLS. Сумісне використання Celoxica Agility Compiler та Xilinx ISE WebPack.

Результати часового симулювання поведінки синтезованої вручну мікросхеми



05.5. Засоби HLS. Сумісне використання Celoxica Agility Compiler та Xilinx ISE WebPack.

Топологія синтезованого вручну проекту для ПЛІС Virtex-II (САПР ISE WebPack)



05.6. Засоби HLS. Сумісне використання Celoxica Agility Compiler та Xilinx ISE WebPack.

САПР ISE WebPack: звіт з споживання електроенергії для ПЛІС Virtex-II (80 тис. вентилів)

The screenshot displays the Xilinx XPower - [circuit_8] application window. The interface includes a menu bar (File, Edit, View, Tools, Window, Help), a toolbar, and a main workspace divided into several panes.

Left Pane: Data and Report Views

| | Voltage (V) | Current (mA) | Power (mW) |
|-----------------------------|-------------|--------------|------------|
| Vccint | 1.5 | | |
| Dynamic | | 0.00 | 0.00 |
| Quiescent | | 3.00 | 4.50 |
| Vccaux | 3.3 | | |
| Dynamic | | 0.00 | 0.00 |
| Quiescent | | 100.00 | 330.00 |
| Vcco33 | 3.3 | | |
| Dynamic | | 0.00 | 0.00 |
| Quiescent | | 1.00 | 3.30 |
| Total Power | | | 337.80 |
| Startup Curr | | 200.00 | |
| Battery Capacity (mA Hours) | | | 0.00 |
| Battery Life (Hours) | | | 0.00 |

Bottom Left: Report Views

- Data Views
- Report Views
 - Power Report (HTML)
 - Power Report

Main Workspace: Power summary table

| Power summary: | | I(mA) | P(mW) |
|------------------------------------|--|-------|-------|
| Total estimated power consumption: | | | 338 |
| Vccint 1.50V: | | 3 | 5 |
| Vccaux 3.30V: | | 100 | 330 |
| Vcco33 3.30V: | | 1 | 3 |
| Clocks: | | 0 | 0 |
| Inputs: | | 0 | 0 |
| Logic: | | 0 | 0 |
| Outputs: | | | |
| Vcco33 | | 0 | 0 |
| Signals: | | 0 | 0 |
| Quiescent Vccint 1.50V: | | 3 | 5 |
| Quiescent Vccaux 3.30V: | | 100 | 330 |
| Quiescent Vcco33 3.30V: | | 1 | 3 |

Bottom Right: Thermal summary table

| Thermal summary: | |
|---------------------------------|-------|
| Estimated junction temperature: | 36C |
| Ambient temp: | 25C |
| Case temp: | 36C |
| Theta J-A: | 34C/W |

Status Bar: Done | 2v40cs144-4

Footer Message: The power estimate will be calculated using ADVANCED data.

06.1. Сумісне проектування апаратури і програм.

Сумісне проектування апаратури і програм (hardware/software co-design) є важливим елементом процесу проектування для сучасних електронних систем, які містять як мікропроцесори, керовані програмним забезпеченням, так і програмоване обладнання. За їх допомогою компоненти можна розділити між апаратним та програмним забезпеченням, функціонально перевірити у різних засобах моделювання, а потім ітеративно перенести до засобів імплементації.

06.2. Сумісне проектування апаратури і програм.

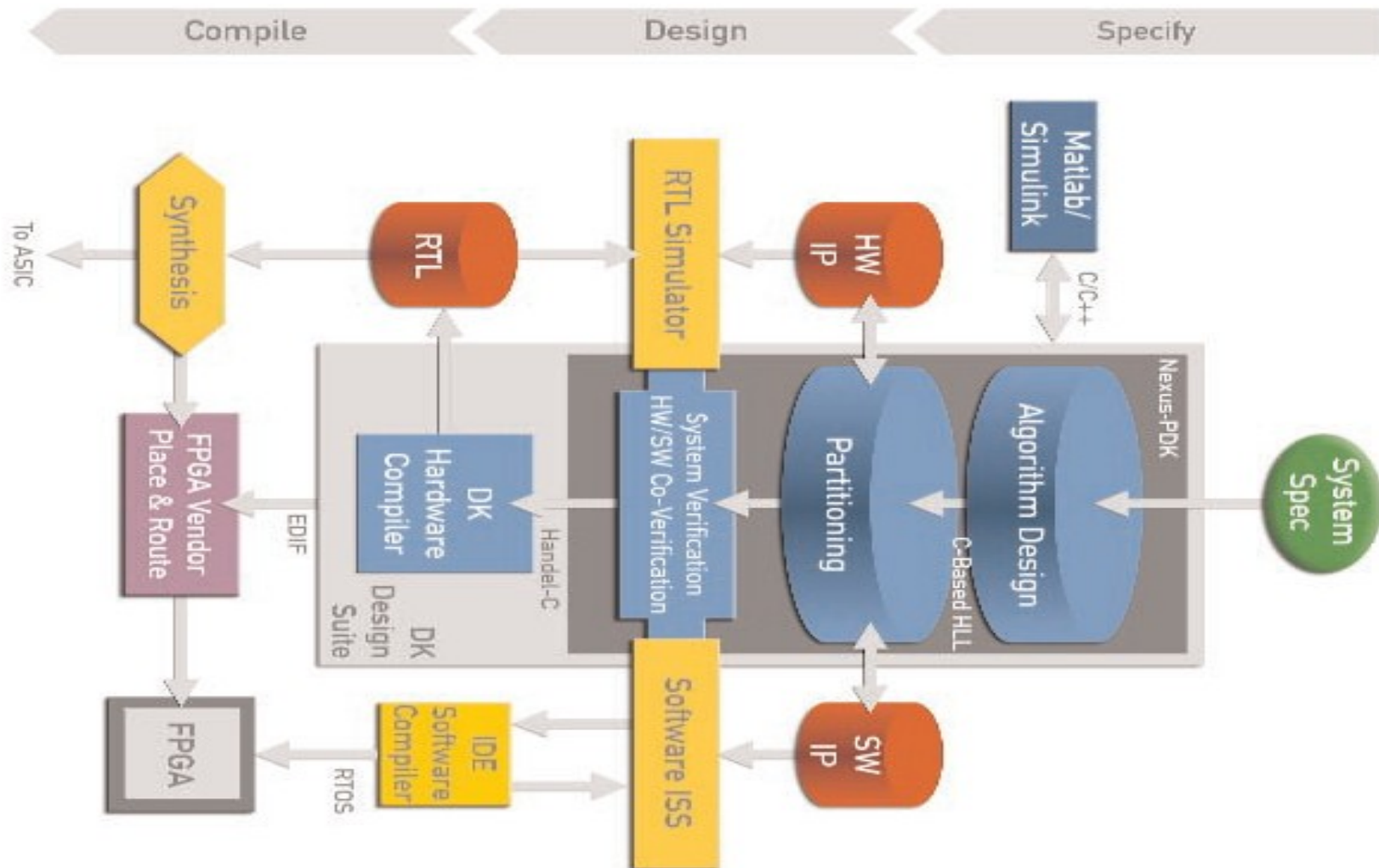
Технологія Platform Abstraction Layer (PAL) абстрагує деталі апаратних периферійних пристроїв та пристроїв введення/виведення, щоб зробити можливою розробку незалежного від платформи опису системи.

PAL абстрагує інженерів-розробників від апаратних інтерфейсів низького рівня, щоб полегшити інтеграцію прототипів ПЛІС з фізичними засобами. Це робиться шляхом розробки бібліотеки низькорівневих інтерфейсів для певних ресурсів платформи, таких як введення/виведення або пам'ять. Потім до цієї бібліотеки, яка називається бібліотекою підтримки платформи (PSL), можна отримати доступ із програм за допомогою простого інтерфейсу програмування API PAL.

07.1. Засоби сумісного проектування апаратури і програм

Celoxica DK Design Suite – повнофункціональне середовище розробки для програмно-компільованого проектування систем. Це дозволяє всім членам команди, від системних архітекторів та розробників апаратного забезпечення до розробників системного програмного забезпечення та прикладного програмного забезпечення, ділитися кодом від специфікації системи до реалізації. Ця методологія призначена для проектування сучасних електронних систем, які містять як мікропроцесори, керовані програмним кодом, так і ПЛІС.

07.2. Засоби сумісного проектування апаратури і програм

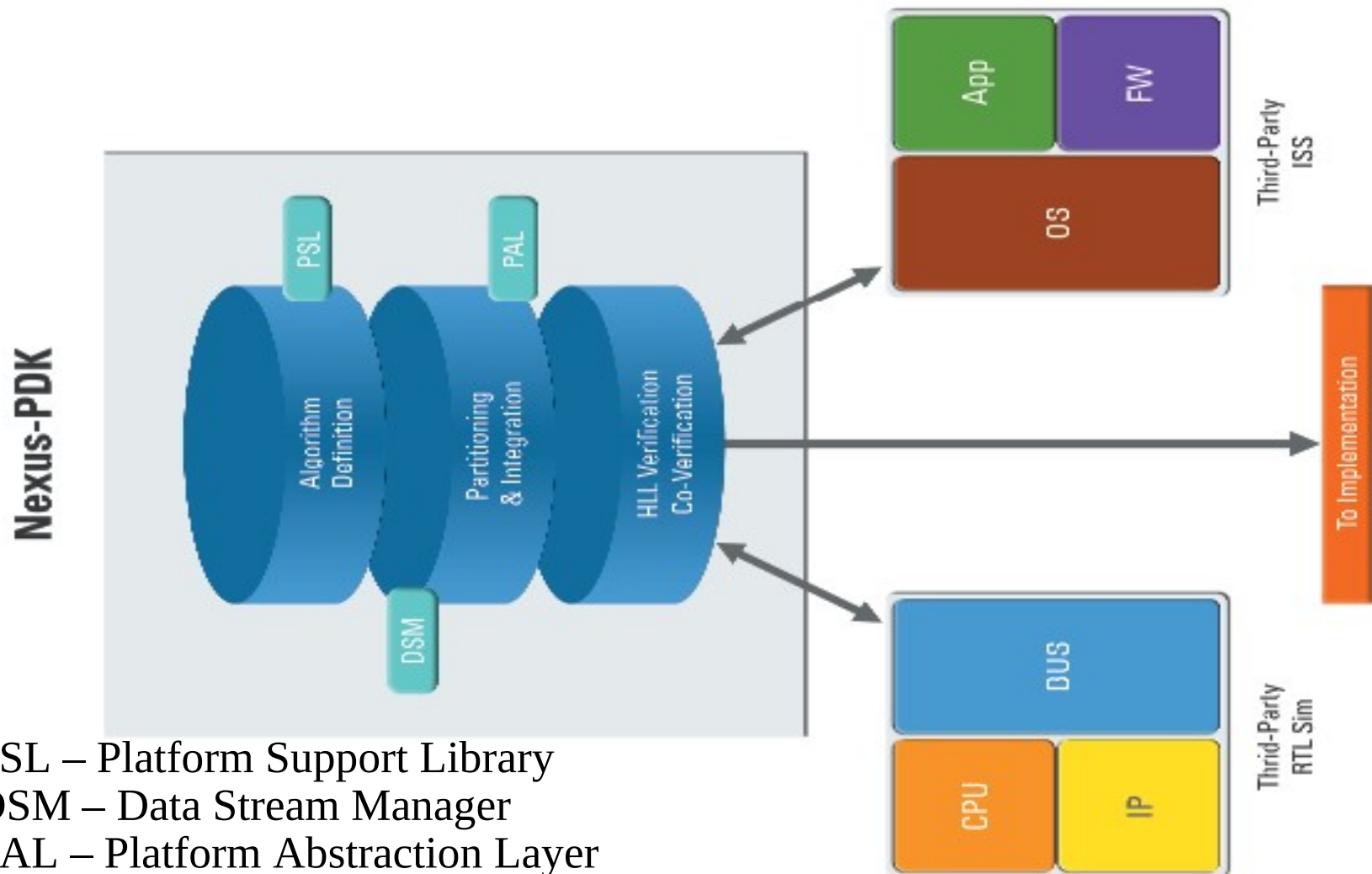


07.3. Засоби сумісного проектування апаратури і програм

Nexus-PDK — інструмент спільного проектування (HW/SW co-design) для розробки, HW/SW-розподілу та моделювання систем за допомогою мов моделювання високого рівня на основі C.

Комплект для розробників платформи (PDK), що входить до складу продукту Nexus-PDK, містить ключові технології системної інтеграції: PAL (Platform Abstraction Layer) та DSM (Data Stream Manager).

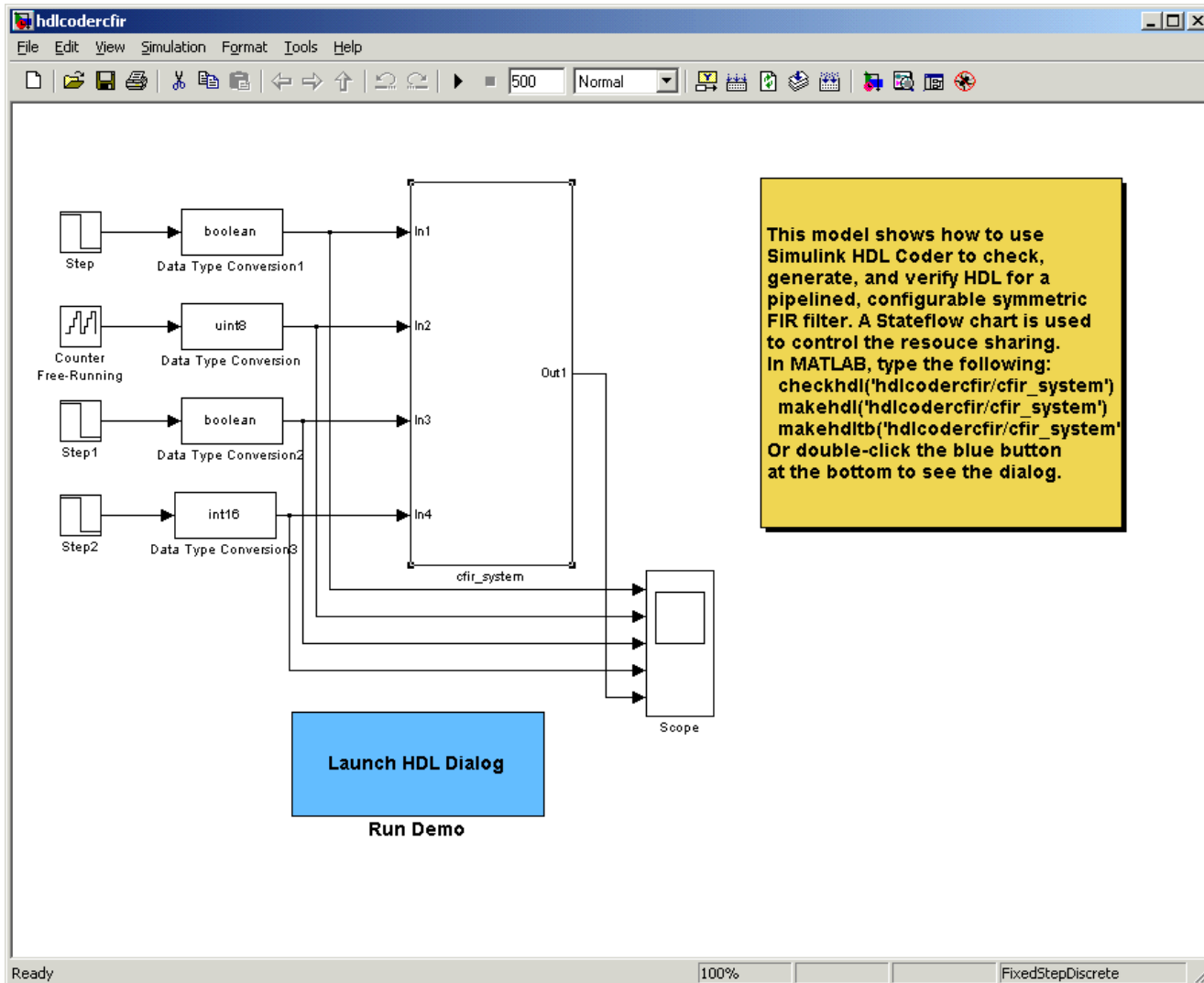
07.4. Засоби сумісного проектування апаратури і програм



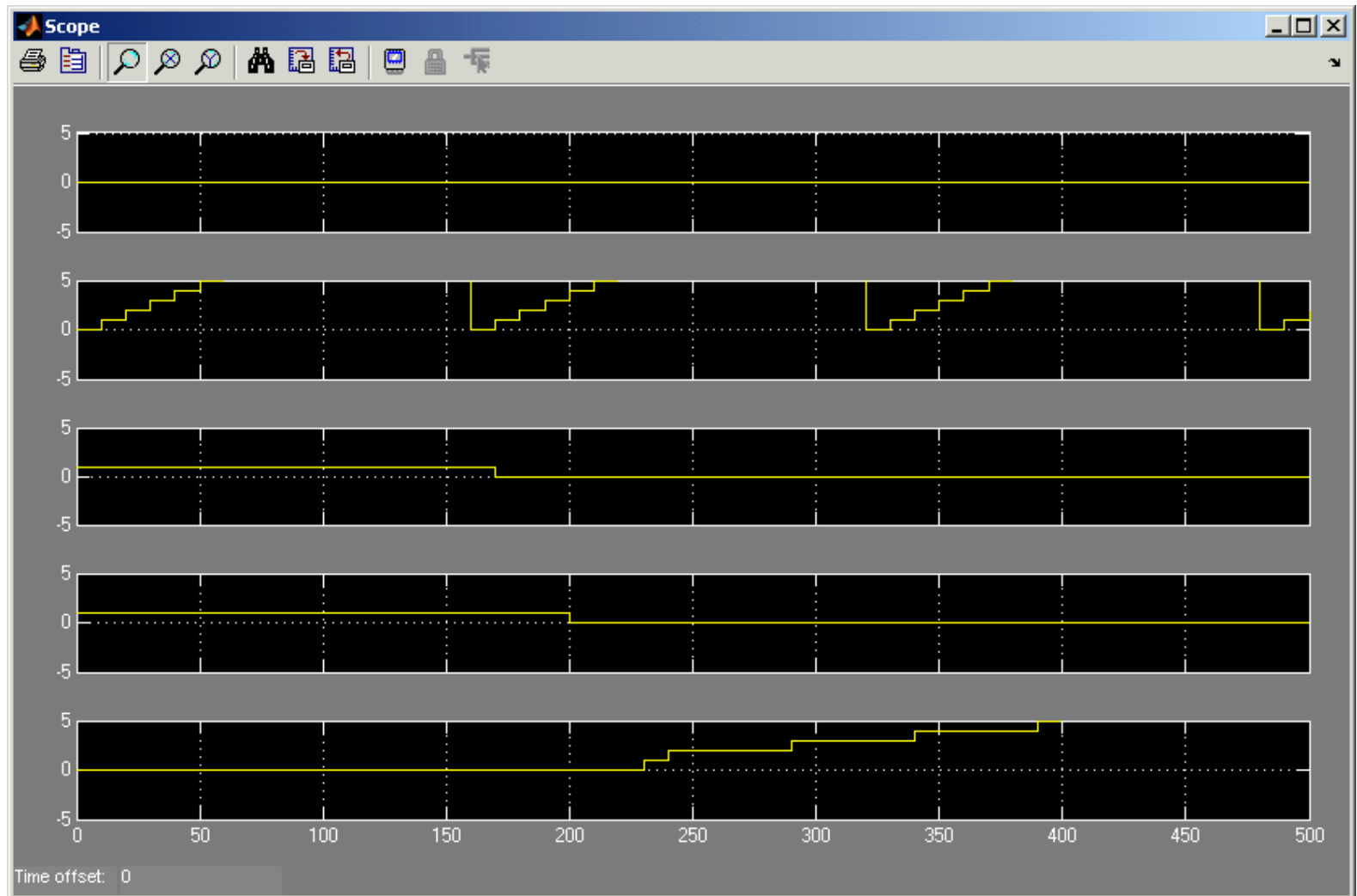
08.1. Застосування MATLAB для системного проектування.

САПР MATLAB містить засіб візуального програмування системних моделей, що має назву Simulink. В свою чергу, існує так званий Simulink HDL Coder, за допомогою якого візуально складену і змодельовану систему можна автоматично перетворити на поведінкову VHDL або Verilog модель з подальшим втіленням останньої до ПЛІС (FPGA) або до спеціалізованої мікросхеми класу ASIC. Отже, MatLab Simulink створює ефективний міст над прірвою, що розділяє результати моделювання на системному рівні і низькорівневу розробку відповідних моделі апаратних засобів. Simulink HDL Coder автоматично генерує поведінковий VHDL опис налаштованої попереднім високорівневим моделюванням Simulink моделі системи. Це дозволяє розробникам швидко і акуратно перетворювати високорівневі абстракції візуальних моделей Simulink models на VHDL або Verilog імплементації, що далі синтезуються до ПЛІС або ASIC.

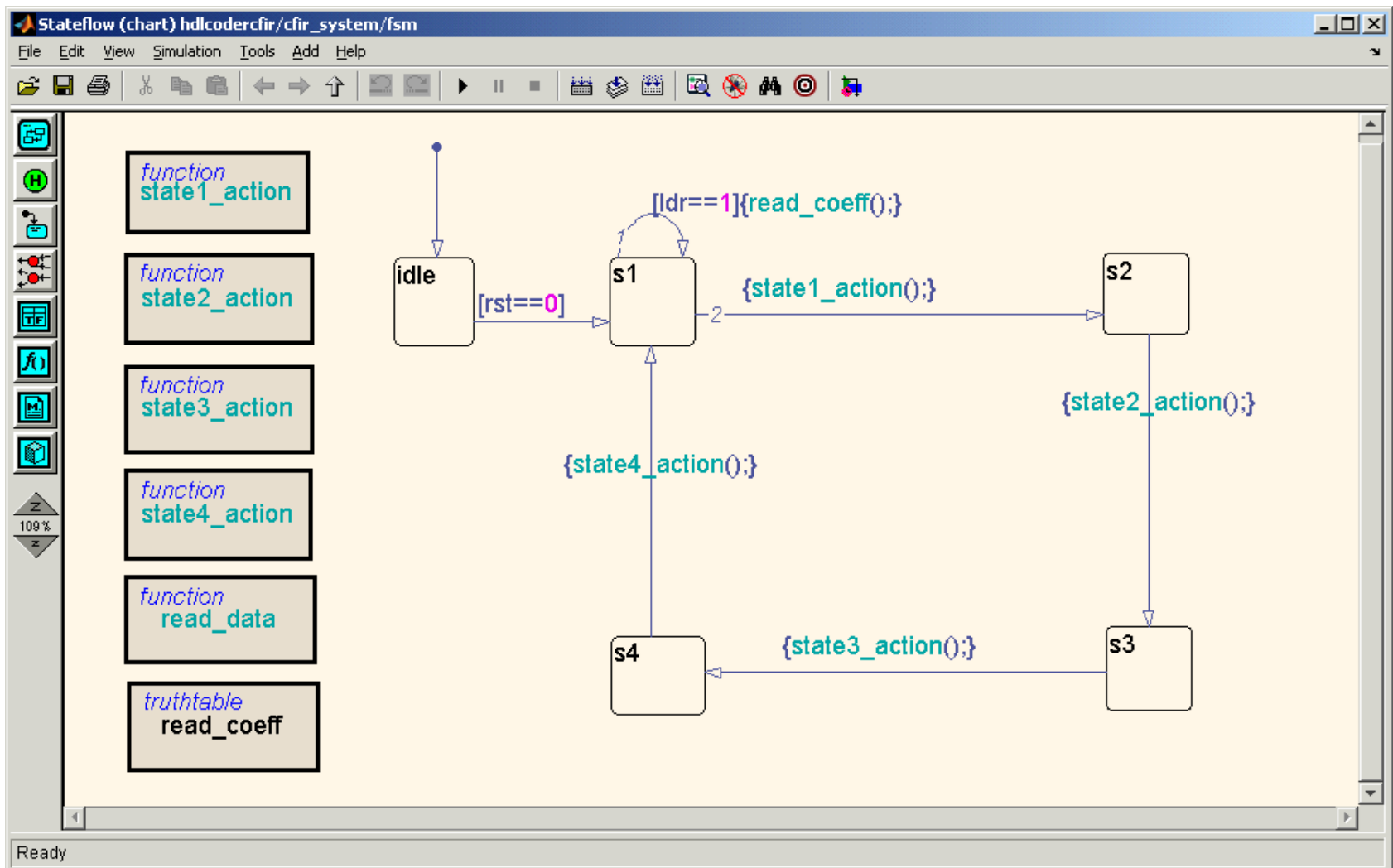
08.2. Застосування MATLAB для системного проектування.



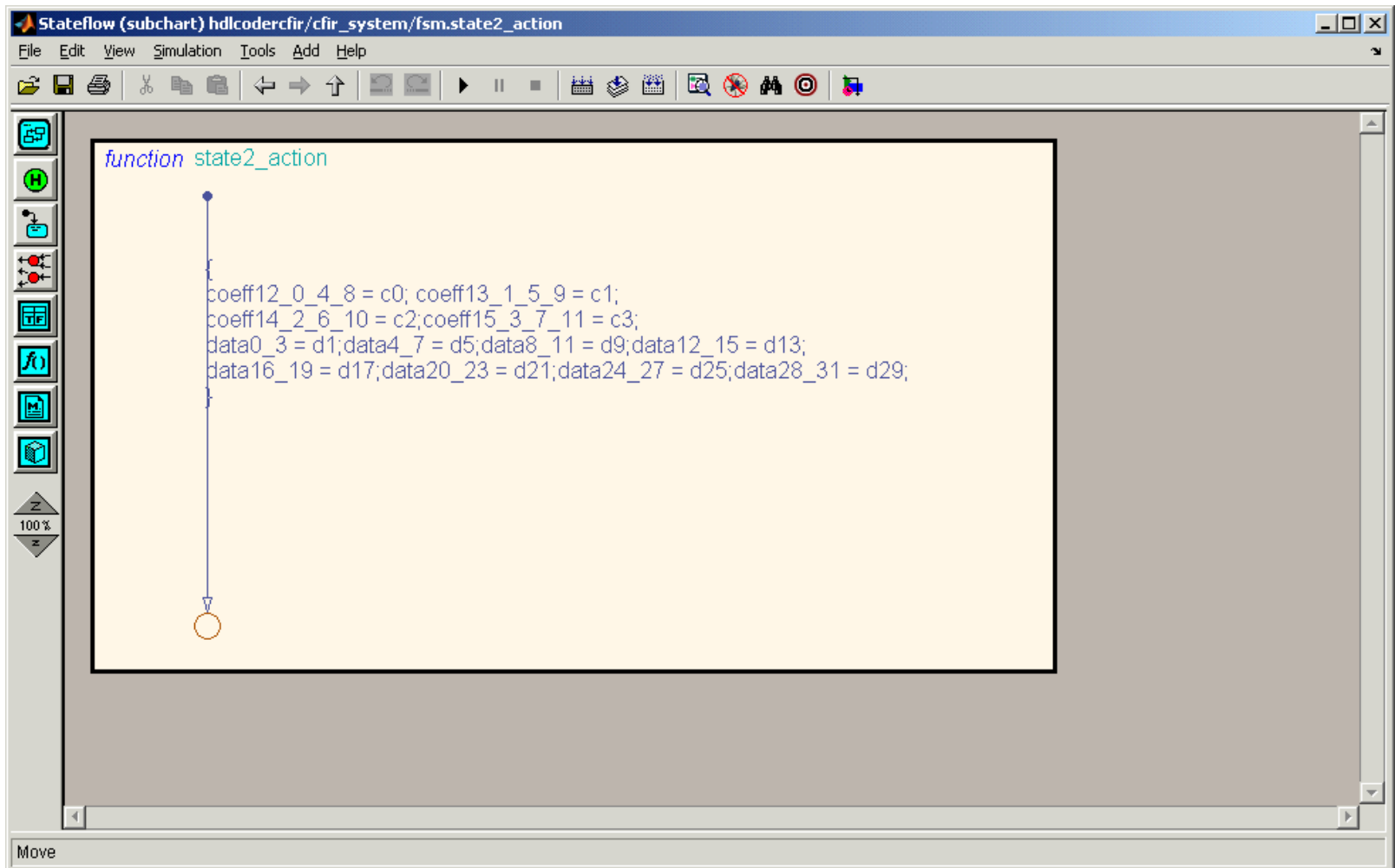
08.3. Застосування MATLAB для системного проектування.



08.4. Застосування MATLAB для системного проектування.



08.5. Застосування MATLAB для системного проектування.



08.6. Застосування MATLAB для системного проектування.

Stateflow (truth table) hdlcoderfir/cfir_system/fsm.read_coeff

File Edit Settings Add Help

Condition Table

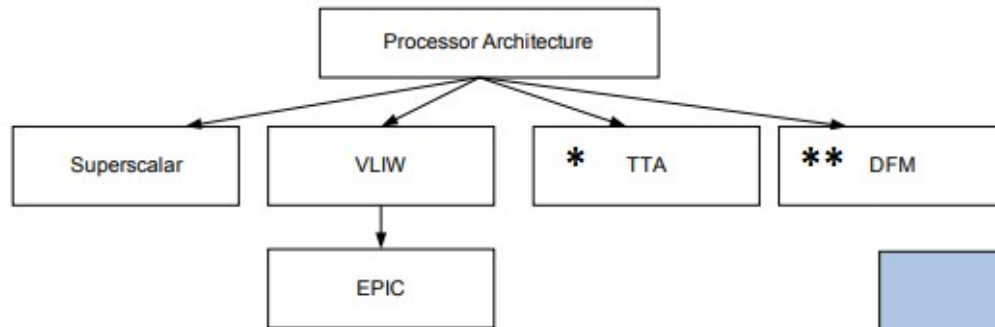
| | Description | Condition | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 | D10 | D11 | D12 | D13 | D14 | D15 | D16 |
|---|-------------|-----------|----|----|----|----|----|----|----|----|----|-----|-----|-----|-----|-----|-----|-----|
| 1 | | addr==0 | T | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 2 | | addr==1 | - | T | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 3 | | addr==2 | - | - | T | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 4 | | addr==3 | - | - | - | T | - | - | - | - | - | - | - | - | - | - | - | - |
| 5 | | addr==4 | - | - | - | - | T | - | - | - | - | - | - | - | - | - | - | - |
| 6 | | addr==5 | - | - | - | - | - | T | - | - | - | - | - | - | - | - | - | - |
| 7 | | addr==6 | - | - | - | - | - | - | T | - | - | - | - | - | - | - | - | - |
| 8 | | addr==7 | - | - | - | - | - | - | - | T | - | - | - | - | - | - | - | - |

Action Table

| # | Description | Action |
|---|-------------|----------------|
| 1 | D1 | A1:c0=data_in; |
| 2 | D2 | A2:c1=data_in; |



Selection of Configurable Processor Extendable Architecture

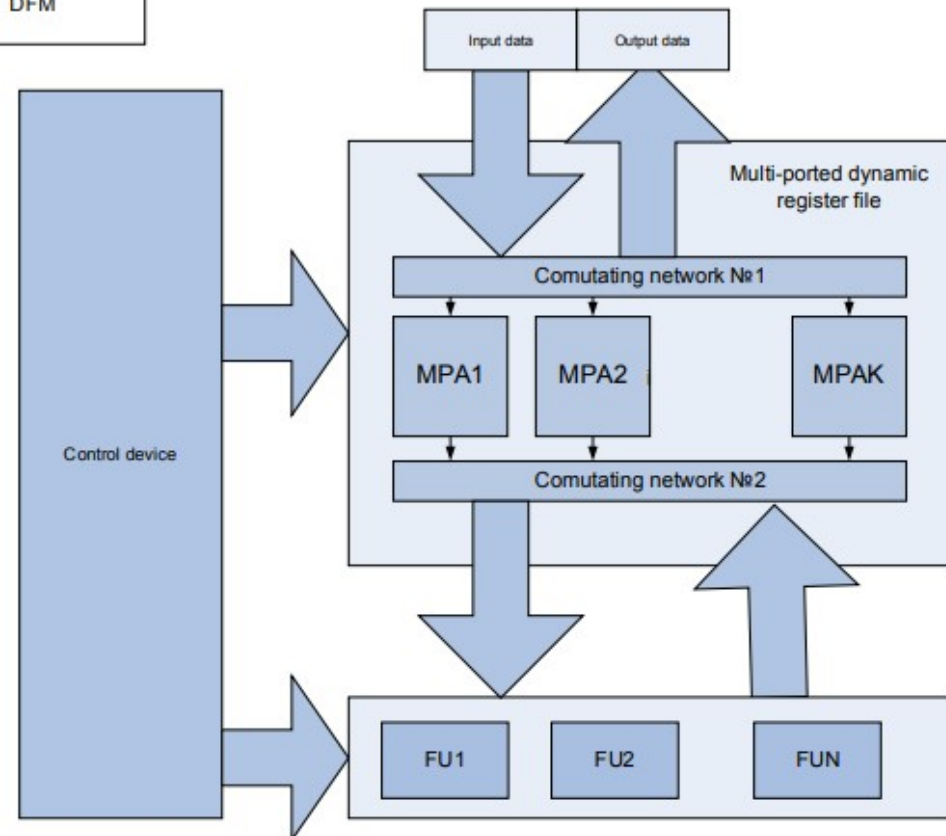


Main principles:

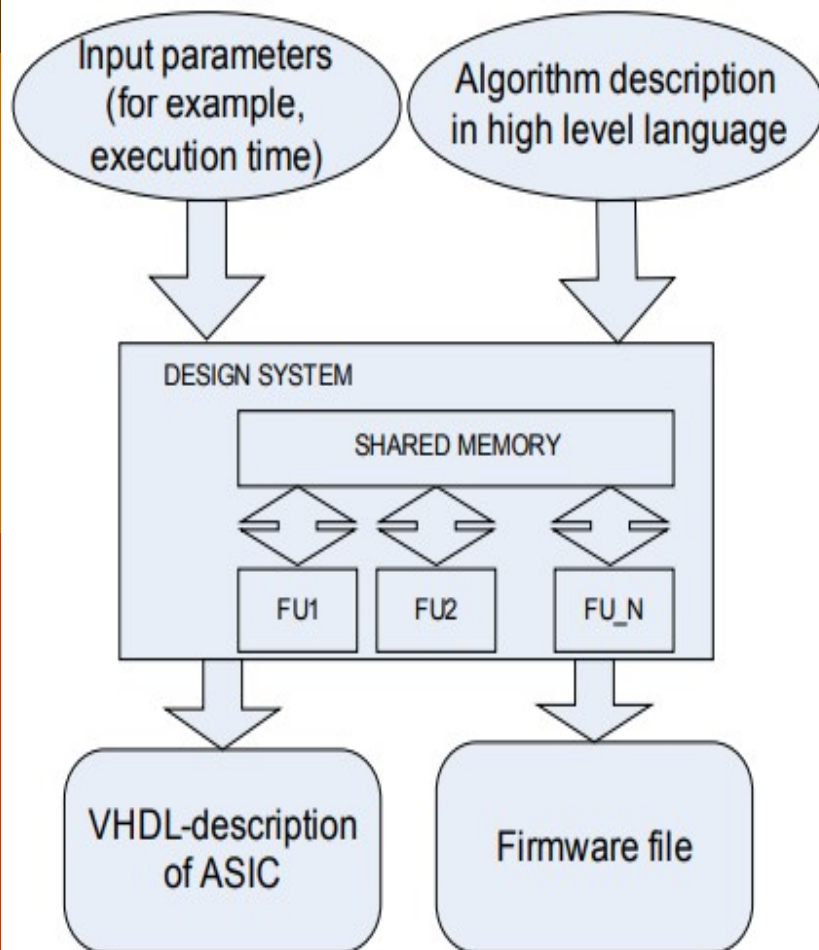
- Instruction level parallelism
- Parallel multichannel instruction execution
- Scalability
- Configurability
- Dynamic principle of data storing

* Transport Triggered Architecture (TTA)

** Dataflow machine (DFM)



Intron's system for automatic generation of programming models



The design flow of system for automatic generation of programming models is divided into hardware and software subsystems development:

One of the main hardware subsystem elements is expandable configurable processor architecture which would afford to create specialized processors for executing given algorithm, accordingly to given configuration parameters. The result of designing system software part is specialization of configurable processor parameters and instruction ROM, which allows generating of optimal certain algorithm oriented VHDL device model.