Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра ЕОМ



Звіт

з лабораторної роботи № 2

з дисципліни: «Тестування та діагностика програмно-апаратних засобів»

на тему: «ТЕСТУВАННЯ ОЗП»

Варіант 29

Виконав:

ст. гр. КІ-303

Порубайміх О.Є.

Перевірив:

Ст.викл. Хомуляк М.О.

Львів – 2024

Зміст

[Зміст 2](#__RefHeading___Toc55759_1171611487)

[Перелік рисунків 2](#__RefHeading___Toc55761_1171611487)

[РОЗДІЛ 1. Мета 4](#__RefHeading___Toc2946_552797710)

[РОЗДІЛ 2. Теоретичні відомості 5](#__RefHeading___Toc2952_552797710)

[РОЗДІЛ 3. Індивідуальне завдання 8](#__RefHeading___Toc2950_552797710)

[РОЗДІЛ 4. Виконання завдання 9](#__RefHeading___Toc2948_552797710)

[4.1. Створення проєкту та файлів модулів. 9](#__RefHeading___Toc2954_552797710)

[4.2. Еталонний вузол ОЗП. 9](#__RefHeading___Toc2966_552797710)

[4.3. Вузол ОЗП з помилкою. 10](#__RefHeading___Toc2964_552797710)

[4.4. Вузол порівняння. 11](#__RefHeading___Toc2962_552797710)

[4.5. Генератор тестових послідовностей. 12](#__RefHeading___Toc2960_552797710)

[4.6. Схема. 16](#__RefHeading___Toc2958_552797710)

[4.7. Процес тестування 17](#__RefHeading___Toc2956_552797710)

[Висновок 20](#__RefHeading___Toc2944_552797710)

[Список використаних джерел 21](#__RefHeading___Toc2942_552797710)

[Додаток А. Код вузла RAM. 22](#__RefHeading___Toc2930_552797710)

[Додаток Б. Код вузла RAM\_Err. 23](#__RefHeading___Toc2940_552797710)

[Додаток В. Код вузла CMP. 24](#__RefHeading___Toc2938_552797710)

[Додаток Г. Код компонента TEST\_BENCH. 25](#__RefHeading___Toc2936_552797710)

[Додаток Д. Код архітектури компонента TEST\_BENCH для синтезу. 26](#__RefHeading___Toc2934_552797710)

[Додаток Е. Код архітектури компонента TEST\_BENCH для симуляції. 27](#__RefHeading___Toc2932_552797710)

Перелік рисунків

[Рис. 4.1. Створення проєкту. 9](#%25D0%25A0%25D0%25B8%25D1%2581._4.!0|seq)

[Рис. 4.2. Створення файлу RAM.vhd. 10](#%25D0%25A0%25D0%25B8%25D1%2581._4.!1|seq)

[Рис. 4.3. Опис еталонного ОЗП. 10](#%25D0%25A0%25D0%25B8%25D1%2581._4.!2|seq)

[Рис. 4.4. Створення файлу RAM\_Err.vhd. 11](#%25D0%25A0%25D0%25B8%25D1%2581._4.!3|seq)

[Рис. 4.5. Опис ОЗП з розривом сигналу. 11](#%25D0%25A0%25D0%25B8%25D1%2581._4.!4|seq)

[Рис. 4.6. Створення файлу CMP.vhd. 12](#%25D0%25A0%25D0%25B8%25D1%2581._4.!5|seq)

[Рис. 4.7. Опис взула порівняння. 12](#%25D0%25A0%25D0%25B8%25D1%2581._4.!6|seq)

[Рис. 4.8. Створення файлу TEST\_BENCH.vhd. 13](#%25D0%25A0%25D0%25B8%25D1%2581._4.!7|seq)

[Рис. 4.9. Створення файлу TEST\_BENCH\_ARCH.vhd. 14](#%25D0%25A0%25D0%25B8%25D1%2581._4.!8|seq)

[Рис. 4.10. Створення файлу TEST\_BENCH\_ARCH\_SIM.vhd. 14](#%25D0%25A0%25D0%25B8%25D1%2581._4.!9|seq)

[Рис. 4.11. Опис компонента TEST\_BENCH. 15](#%25D0%25A0%25D0%25B8%25D1%2581._4.!10|se)

[Рис. 4.12. Опис архітектури компонента TEST\_BENCH для синтезу. 15](#%25D0%25A0%25D0%25B8%25D1%2581._4.!11|se)

[Рис. 4.13. Опис архітектури компонента TEST\_BENCH для симуляції. 16](#%25D0%25A0%25D0%25B8%25D1%2581._4.!12|se)

[Рис. 4.14. Створення файлу схеми. 17](#%25D0%25A0%25D0%25B8%25D1%2581._4.!13|se)

[Рис. 4.15. Схема. 17](#%25D0%25A0%25D0%25B8%25D1%2581._4.!14|se)

[Рис. 4.16. Часова діаграма тестування шини адреси. 18](#%25D0%25A0%25D0%25B8%25D1%2581._4.!15|se)

[Рис. 4.17. Часова діаграма тестування шини даних. 19](#%25D0%25A0%25D0%25B8%25D1%2581._4.!16|se)

1. Мета

Ознайомлення із загальною схемою тестування цифрової техніки. Засвоєння методів та засобів тестування комбінаційних схем з пам’яттю на прикладі оперативного запам’ятовувального пристрою (ОЗП).

1. Теоретичні відомості

Тестування схем з пам'яттю є значно складнішим процесом порівняно з тестуванням комбінаційних схем. Основна відмінність полягає в тому, що результати роботи цифрових вузлів з пам'яттю залежать не тільки від їх поточного стану, а й від попередніх станів. Це додає додаткові виклики у процес тестування, оскільки потрібно враховувати історію станів вузла.

Один з найпоширеніших типів схем з пам'яттю - це оперативний запам'ятовуючий пристрій (ОЗП). Він використовується для тимчасового зберігання даних, які можуть бути швидко зчитані або записані процесором. Ця характеристика робить ОЗП критично важливим компонентом у більшості комп'ютерних систем.

Оперативний запам'ятовуючий пристрій (ОЗП) має різноманітні інтерфейсні входи, кожен з яких відіграє ключову роль у функціонуванні пристрою:

1. Адреса (A): Цей вхід важливий для ідентифікації конкретної комірки пам'яті, з якої потрібно зчитати інформацію або в яку треба записати дані. Адреса представляє собою двійковий код, де кожен біт відповідає за окрему комірку.
2. Вхідні дані (DI - Data Input): Цей вхід призначений для подання інформації, яка буде записана в пам'ять. Так само як і адреса, дані кодуються у двійковій формі.
3. Дозвіл видачі інформації на вихід (OE - Output Enable): Використовується для управління виведенням даних з ОЗП. При активації OE дозволяє передачу даних з вибраної комірки на вихід DO; без активації даних на виході не з'являється.
4. Запис (WR - Write): Цей сигнал керує процесом запису в пам'ять. Коли вхід WR активований, інформація з DI переноситься у комірку, адреса якої задана на вході A.
5. Читання (RD - Read): Відповідає за читання даних з пам'яті. При активації RD інформація з комірки, обраної адресою, виводиться через DO. Без активації читання не відбувається.
6. Нарощування об'єму пам'яті (CS - Chip Select): Цей вхід необхідний для вибору конкретної мікросхеми ОЗП у конфігураціях, де кілька мікросхем об'єднані для розширення обсягу доступної пам'яті.

Вихід ОЗП:

1. Вихід даних (DO - Data Output): Використовується для виводу даних з ОЗП. Дані, що з'являються на цьому виході, відповідають інформації, збереженій у вибраній комірці.

Кількісні характеристики ОЗП:

1. Кількість слів (N): Вказує на загальну кількість комірок пам'яті в ОЗП. Значення N визначається як 2^n, де n — кількість бітів адреси.
2. Об'єм пам'яті (V): Визначає загальний обсяг пам'яті в бітах. Об'єм обчислюється за формулою V = m \* N, де m — розмір одного слова пам'яті в бітах.

Процес тестування ОЗП можна розділити на три основні етапи: перевірка шини даних, перевірка шини адреси, та перевірка на збереження інформації. Перевірка шини даних і адреси включає тести на виявлення можливих закороток або обривів, що виконуються за допомогою спеціально розроблених тестових послідовностей кодів. Тест на збереження інформації перевіряє, чи здатний ОЗП надійно зберігати дані, шляхом запису та подальшого читання даних з усіх комірок пам'яті.

Кількість необхідних циклів запису та читання для тестування ОЗП залежить від розміру пам'яті та типу проведеного тесту. Наприклад, для тесту шини даних потрібно виконати певну кількість циклів запису та читання, пропорційну розміру ОЗП.

Тестування репрограмованого постійного запам'ятовуючого пристрою (РПЗП) має схожість з тестуванням ОЗП, але існують ключові відмінності. Основна з них полягає у тому, що РПЗП може бути перепрограмований лише обмежену кількість разів, через що його тестування слід проводити з особливою обережністю. Відмінною особливістю тестування РПЗП є використання контрольних сум для перевірки цілісності даних.

Узагальнюючи, тестування схем з пам'яттю представляє собою виклик, який вимагає особливих методів та підходів, відмінних від тих, що використовуються при тестуванні комбінаційних схем. Важливість такого тестування зростає з ускладненням схемотехніки та підвищенням вимог до надійності сучасних електронних пристроїв.

1. Індивідуальне завдання

Змоделювати ОЗП та протестувати його при різних помилках.

Варіант № 29: Обрив вихідного сигналу.

1. Виконання завдання
   1. Створення проєкту та файлів модулів.

Створюю проєкт (Рис. 4.1) та встановлюю його налаштування згідно методичних вказівок.

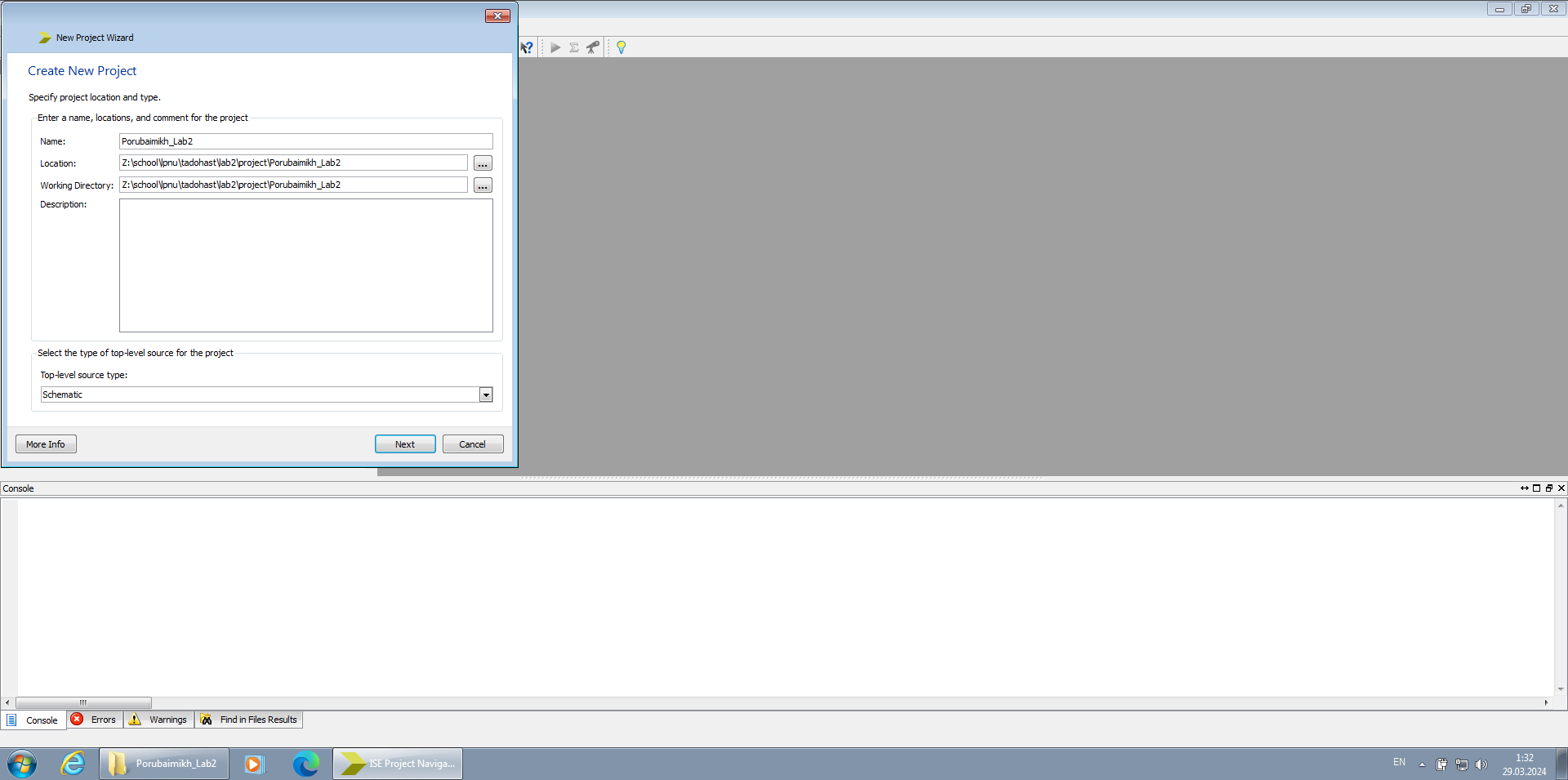


Рис. 4.1. Створення проєкту.

* 1. Еталонний вузол ОЗП.

Процес створення VHDL файлу для опису еталонного ОЗП наведено на Рис. 4.2, опис еталонного ОЗП наведено на Рис. 4.3, повний опис вузла RAM містить ДОДАТОК А. Справний ОЗП, при надходженні даних на вхід DI (вхідні дані) і логічної одиниці на вхід WR (дозвіл запису), записує значення DI за адресою A. При надходженні логічної одиниці на вхід OE (дозвіл читання), виводить дані за адресою A на вихід DO (вихідні дані). Для того, щоб ОЗП функціонував, на вхід CS треба занести логічний нуль. Інакше – читання і запис не відбуваються.

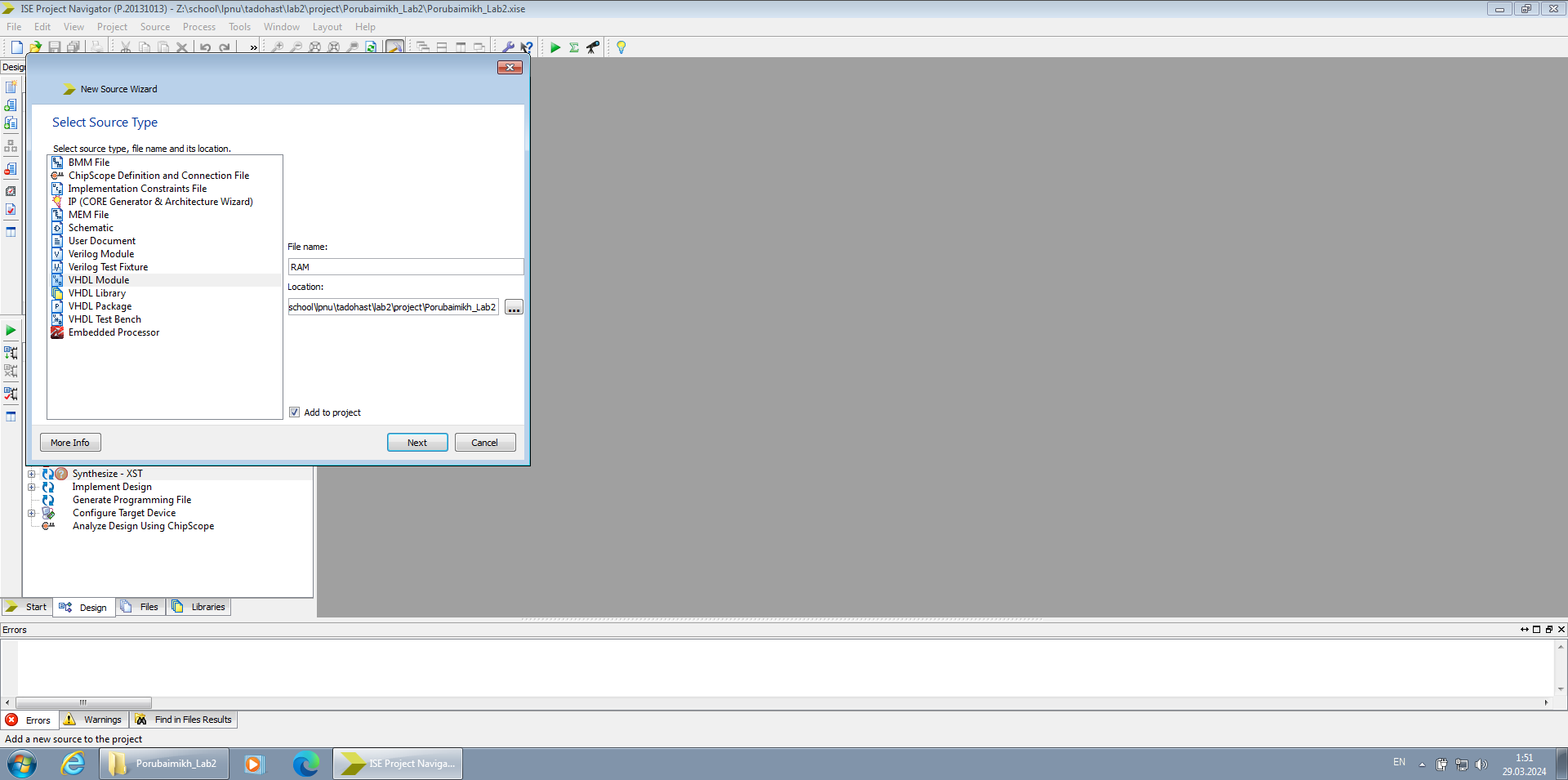


Рис. 4.2. Створення файлу RAM.vhd.

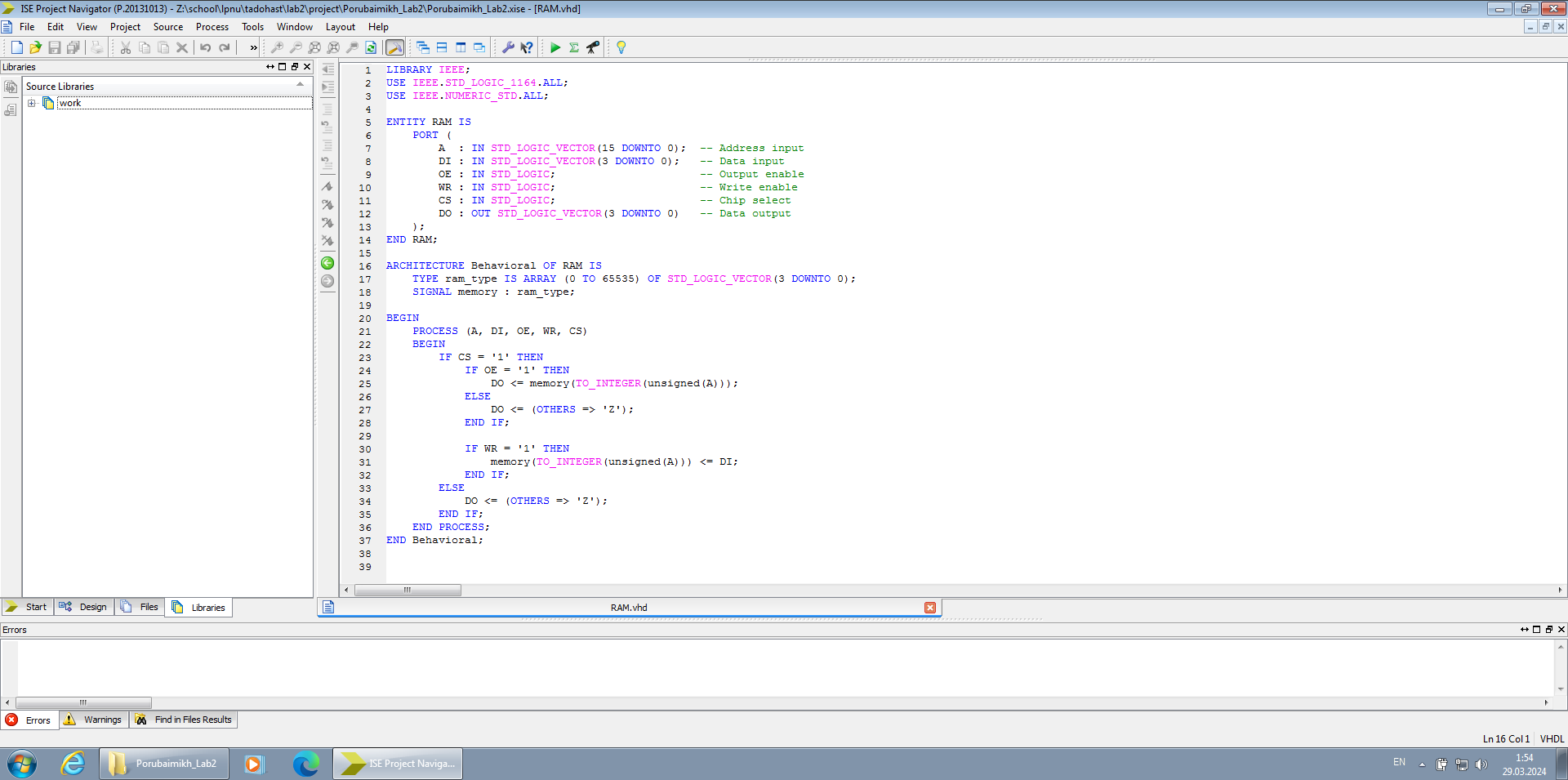


Рис. 4.3. Опис еталонного ОЗП.

* 1. Вузол ОЗП з помилкою.

Процес створення VHDL файлу для опису вузла ОЗП з помилкою наведено на Рис. 4.3. Опис вузла ОЗП з помилкою наведено на Рис. 4.5, опис вузла RAM\_Err містить ДОДАТОК Б. Несправний ОЗП повторює функціонал, але на етапі виводу даних симулює обрив сигналу 1 шини виводу даних. Симуляцію розриву виконує сегмент коду, названий «Error injection».

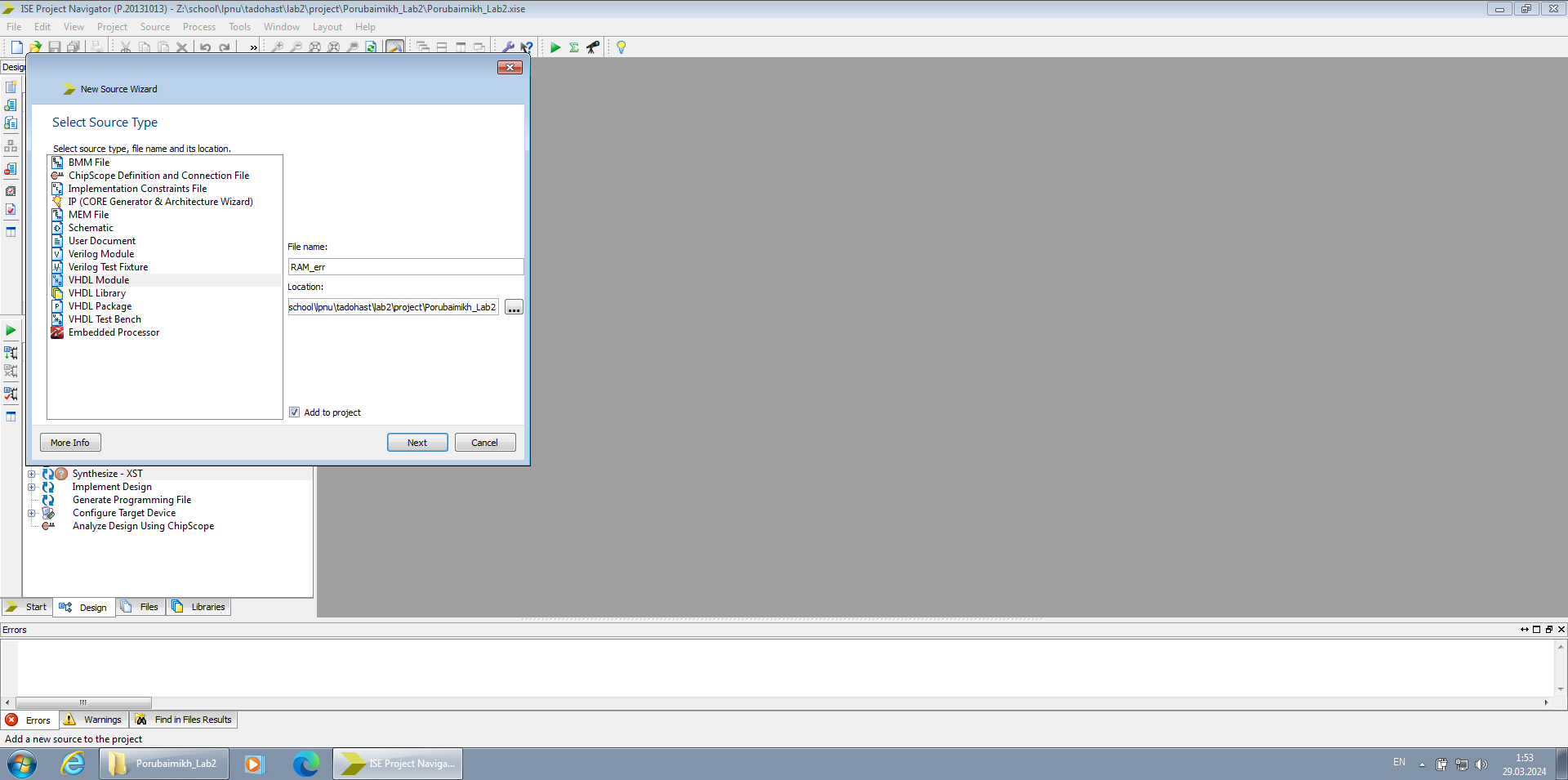


Рис. 4.4. Створення файлу RAM\_Err.vhd.

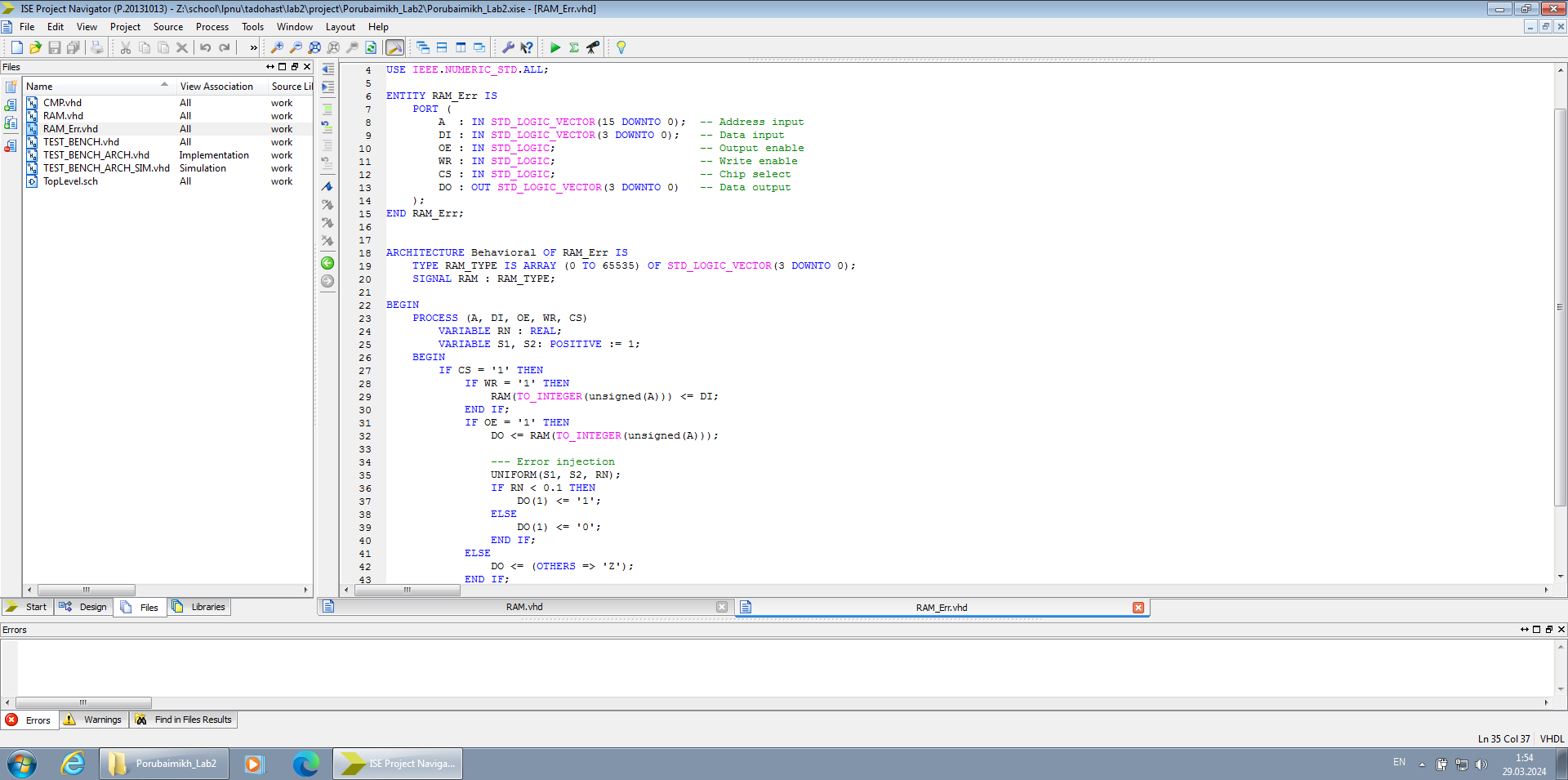


Рис. 4.5. Опис ОЗП з розривом сигналу.

* 1. Вузол порівняння.

Процес створення VHDL файлу для опису вузла порівняння наведено на Рис. 4.6. Опис вузла порівняння наведено на Рис. 4.7, опис вузла CMP містить ДОДАТОК В. При надходженні сигналів на входи A та B, модуль порівняння робить перевірку на рівність даних сигналів. Якщо значення сигналів співпадає, на виході Error буде 0, інакше – 1.

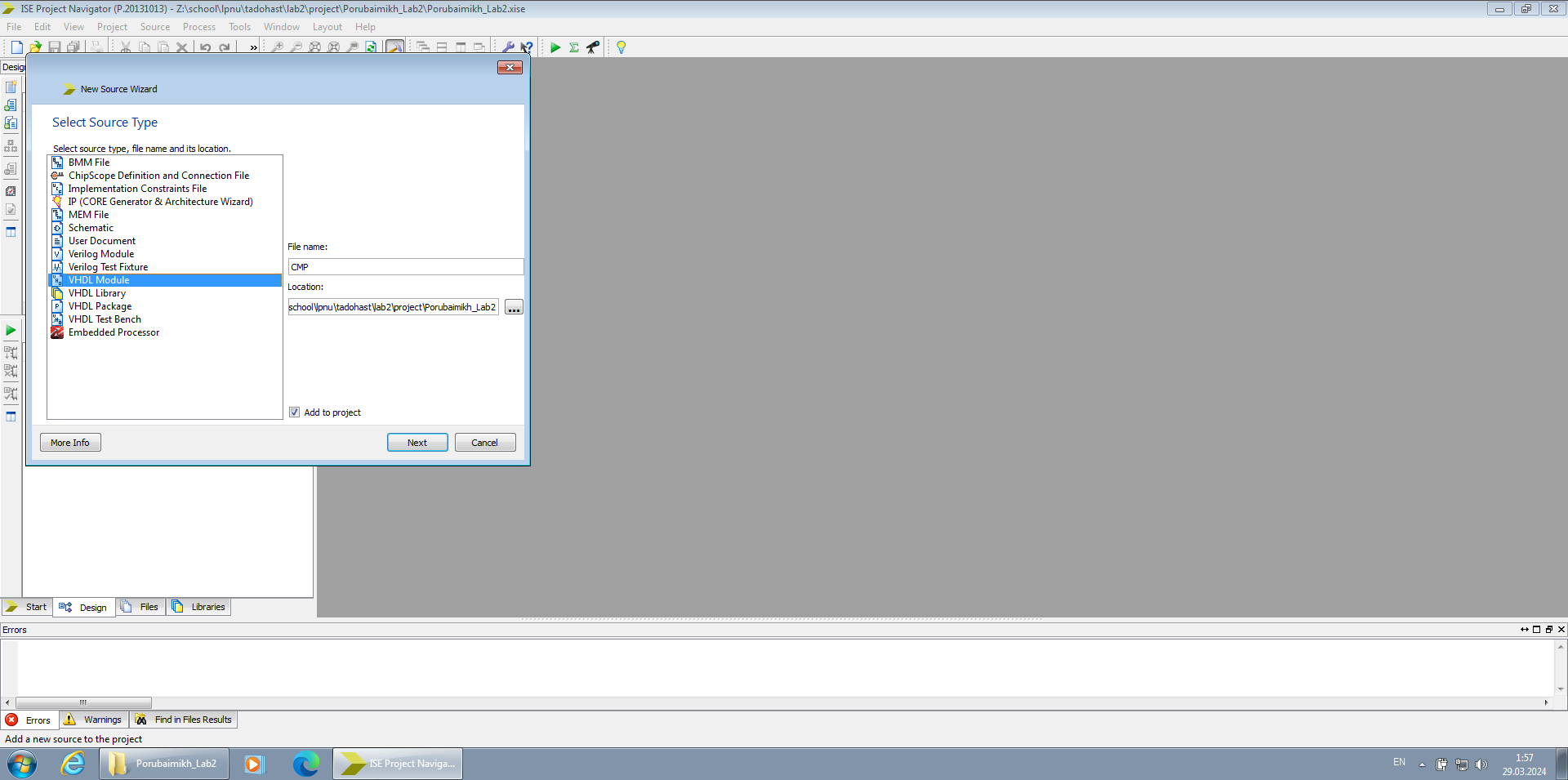


Рис. 4.6. Створення файлу CMP.vhd.

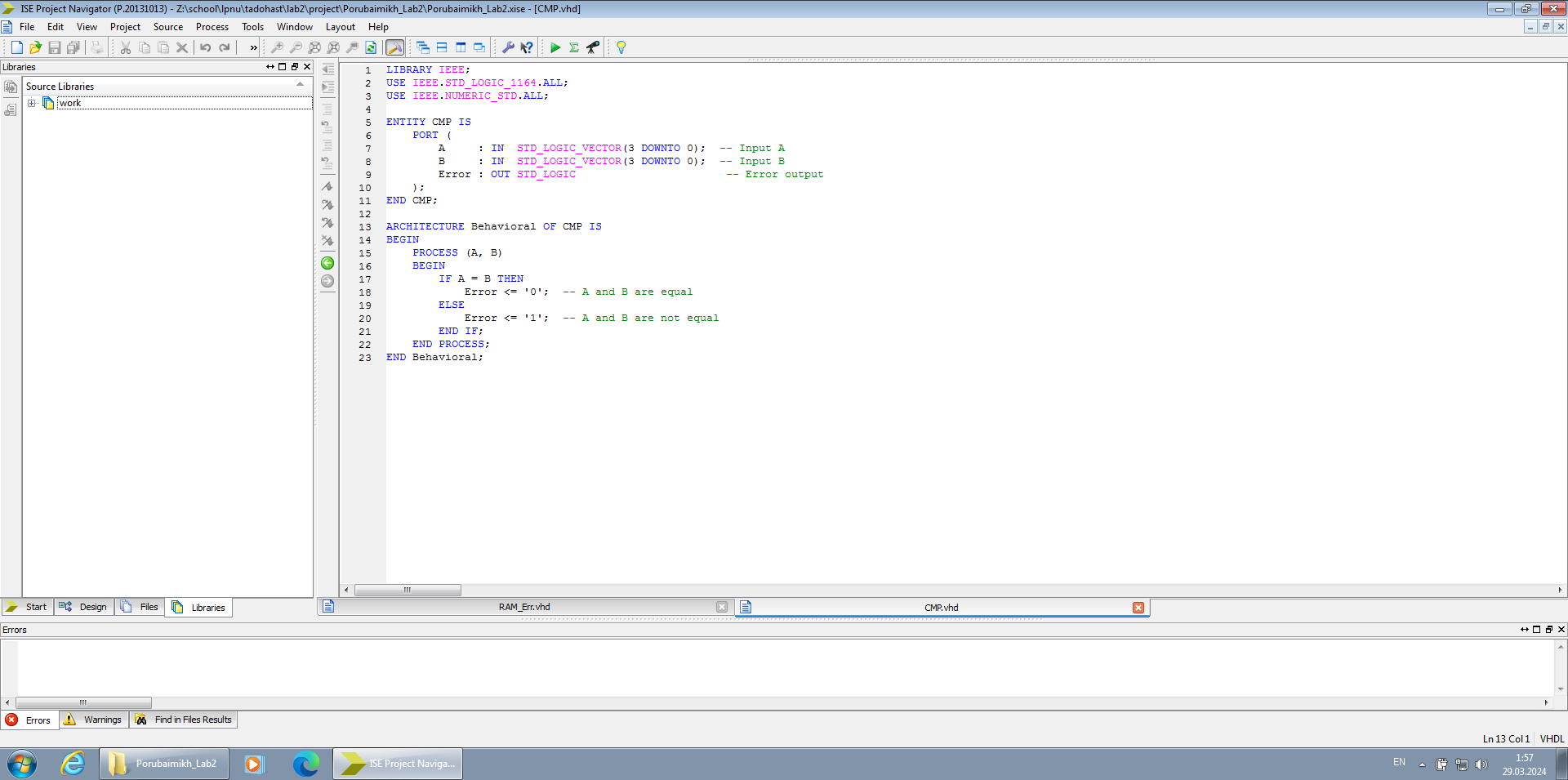


Рис. 4.7. Опис взула порівняння.

* 1. Генератор тестових послідовностей.

Для опису генератора тестових послідовностей, необхідно створити 3 файли:

1. файл з описом компонента TEST\_BENCH;
2. файл з описом архітектури, що буде використана під час синтезу;
3. файл з описом архітектури, що буде використана під час симуляції.

Процес створення файлу для опису компонента TEST\_BENCH наведено на Рис. 4.8. Процес створення файлу для опису архітектури компонента TEST\_BENCH для синтезу наведено на Рис. 4.9. Процес створення файлу для опису архітектури компонента TEST\_BENCH для симуляції наведено на Рис. 4.10.

Опис компонента TEST\_BENCH наведено на Рис. 4.11. Повний опис компонента TEST\_BENCH містить ДОДАТОК Г.

Опис архітектури компонента TEST\_BENCH для синтезу наведено на Рис. 4.12. Повний опис компонента TEST\_BENCH для синтезу містить ДОДАТОК Д.

Опис архітектури компонента TEST\_BENCH для симуляції наведено на Рис. 4.13. Повний опис компонента TEST\_BENCH для симуляції містить ДОДАТОК Е.

Архітектура TEST\_BENCH для симуляції підтримує 3 режими: режим тестування адресної шини, шини даних і тестування збереження інформації. Переключення режиму відбувається за допомогою зміни значення сигналу move у вихідному коді опису архітектури.

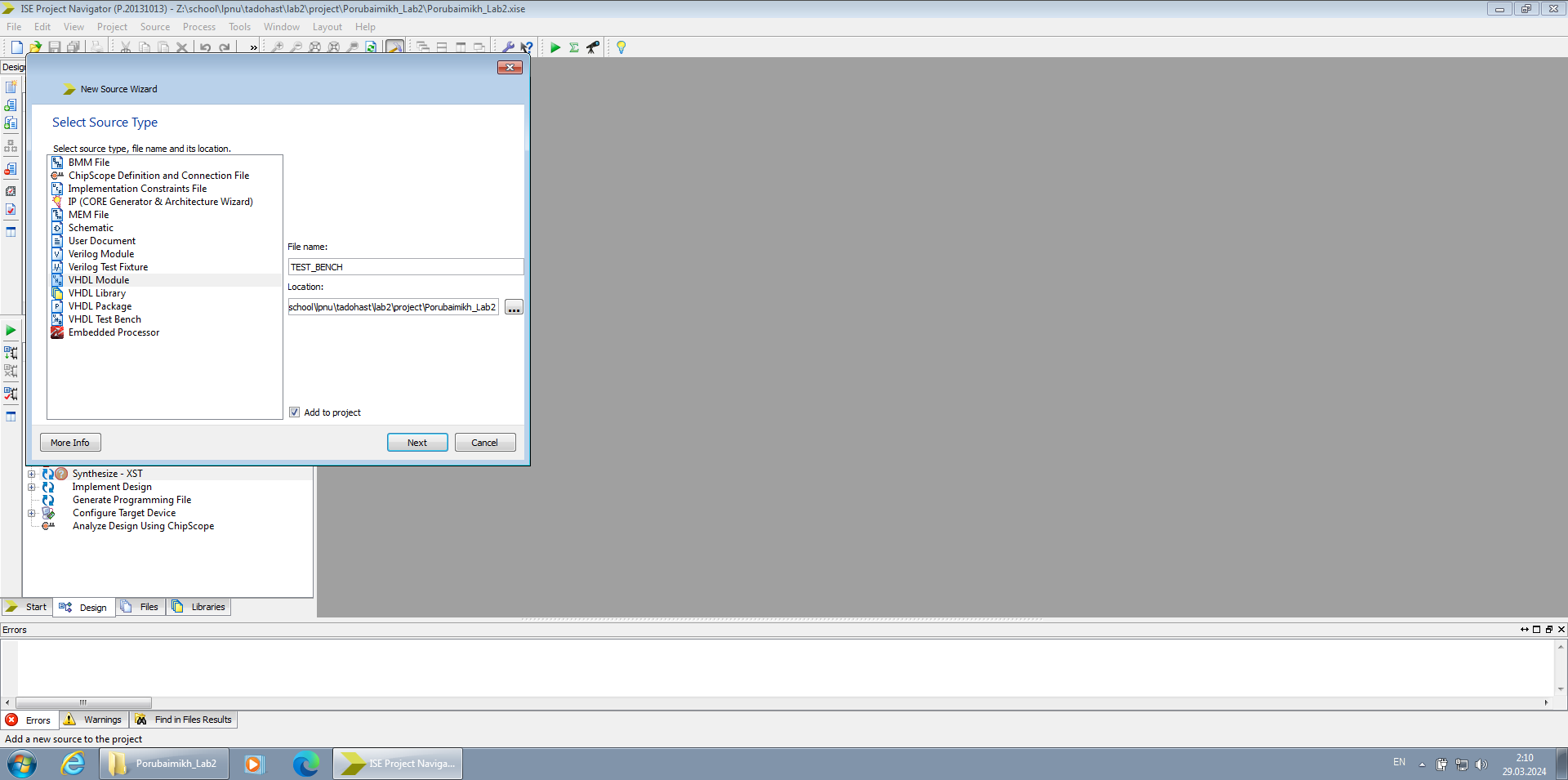


Рис. 4.8. Створення файлу TEST\_BENCH.vhd.

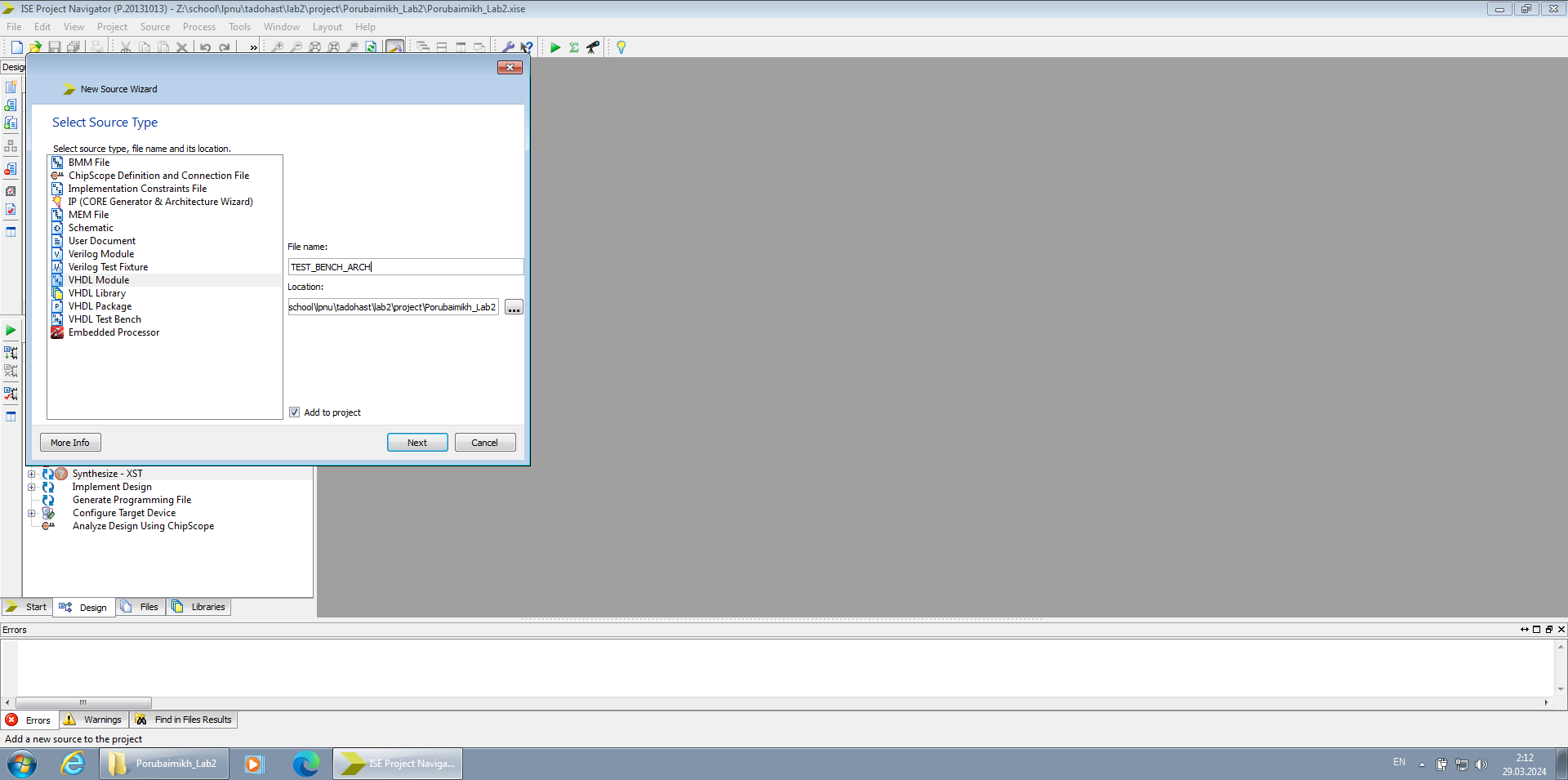


Рис. 4.9. Створення файлу TEST\_BENCH\_ARCH.vhd.

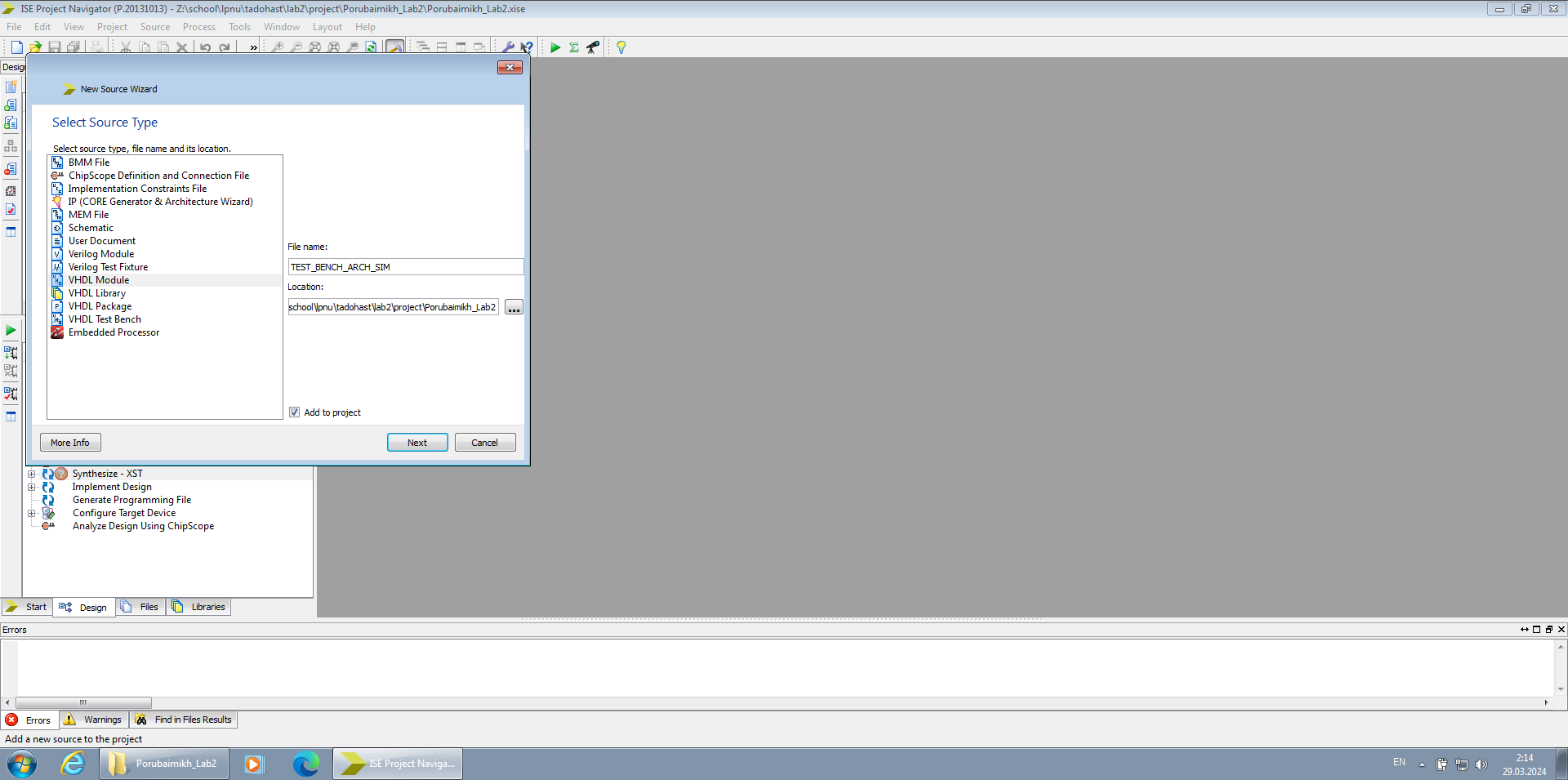


Рис. 4.10. Створення файлу TEST\_BENCH\_ARCH\_SIM.vhd.

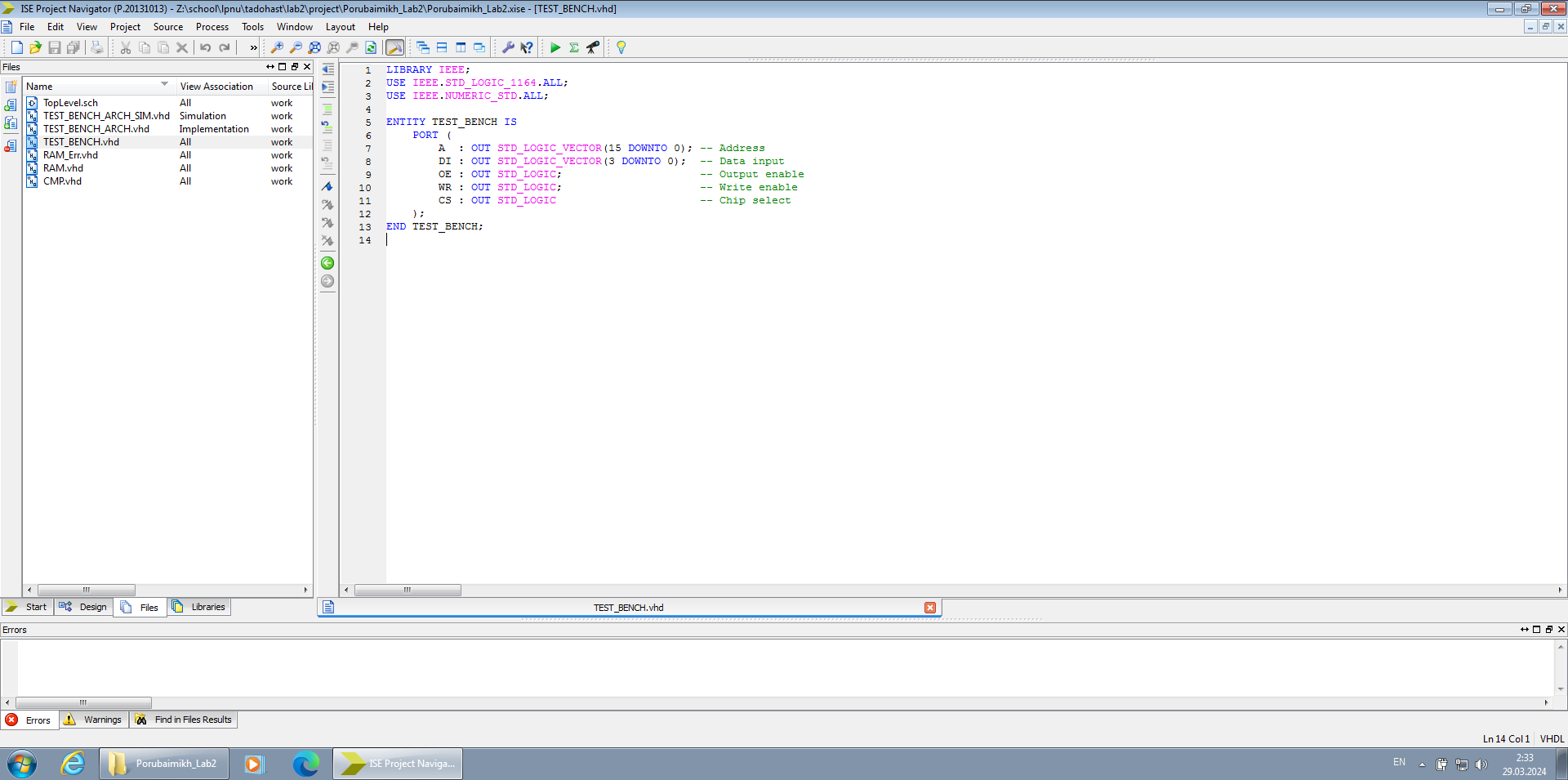


Рис. 4.11. Опис компонента TEST\_BENCH.

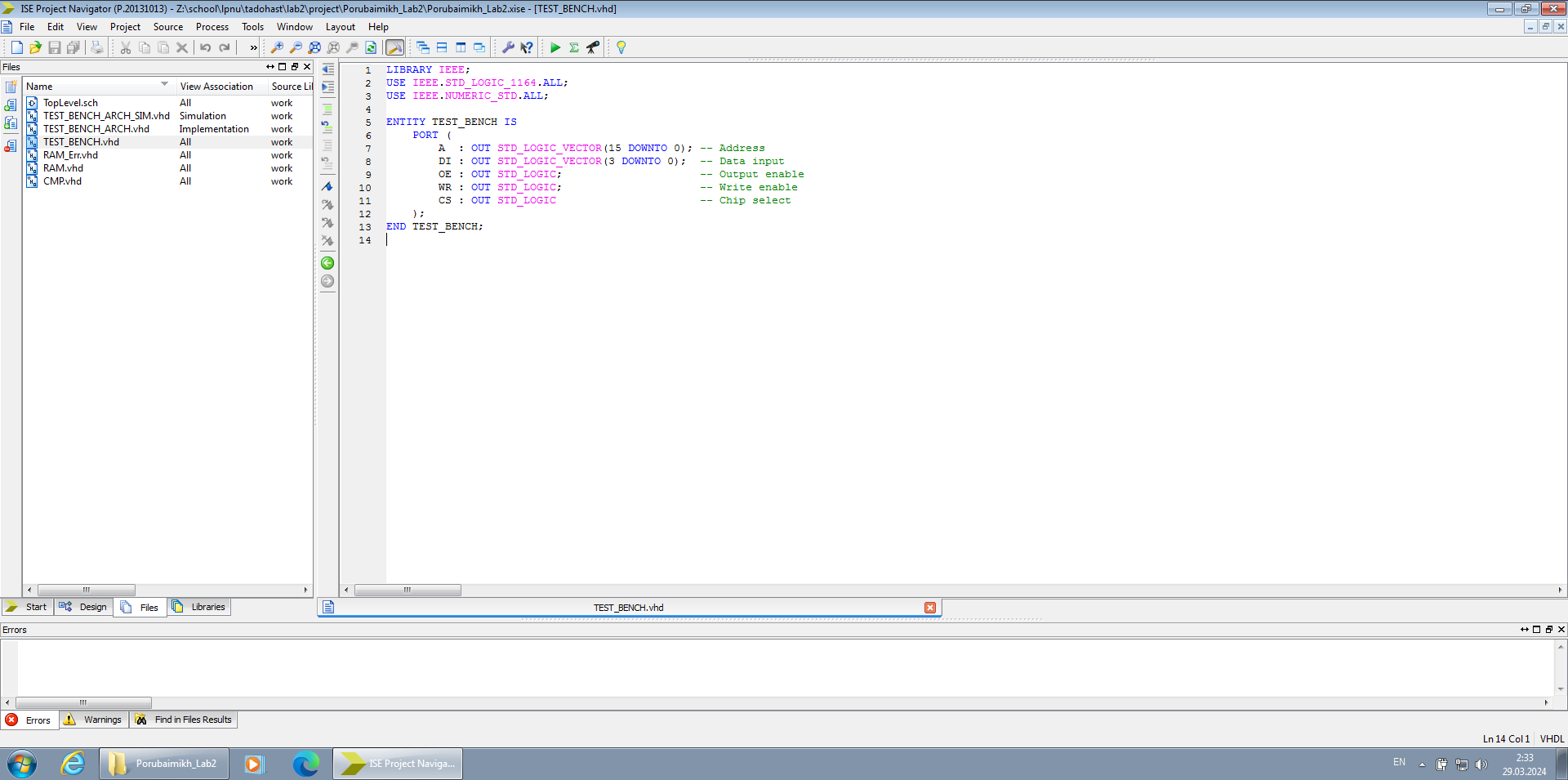


Рис. 4.12. Опис архітектури компонента TEST\_BENCH для синтезу.

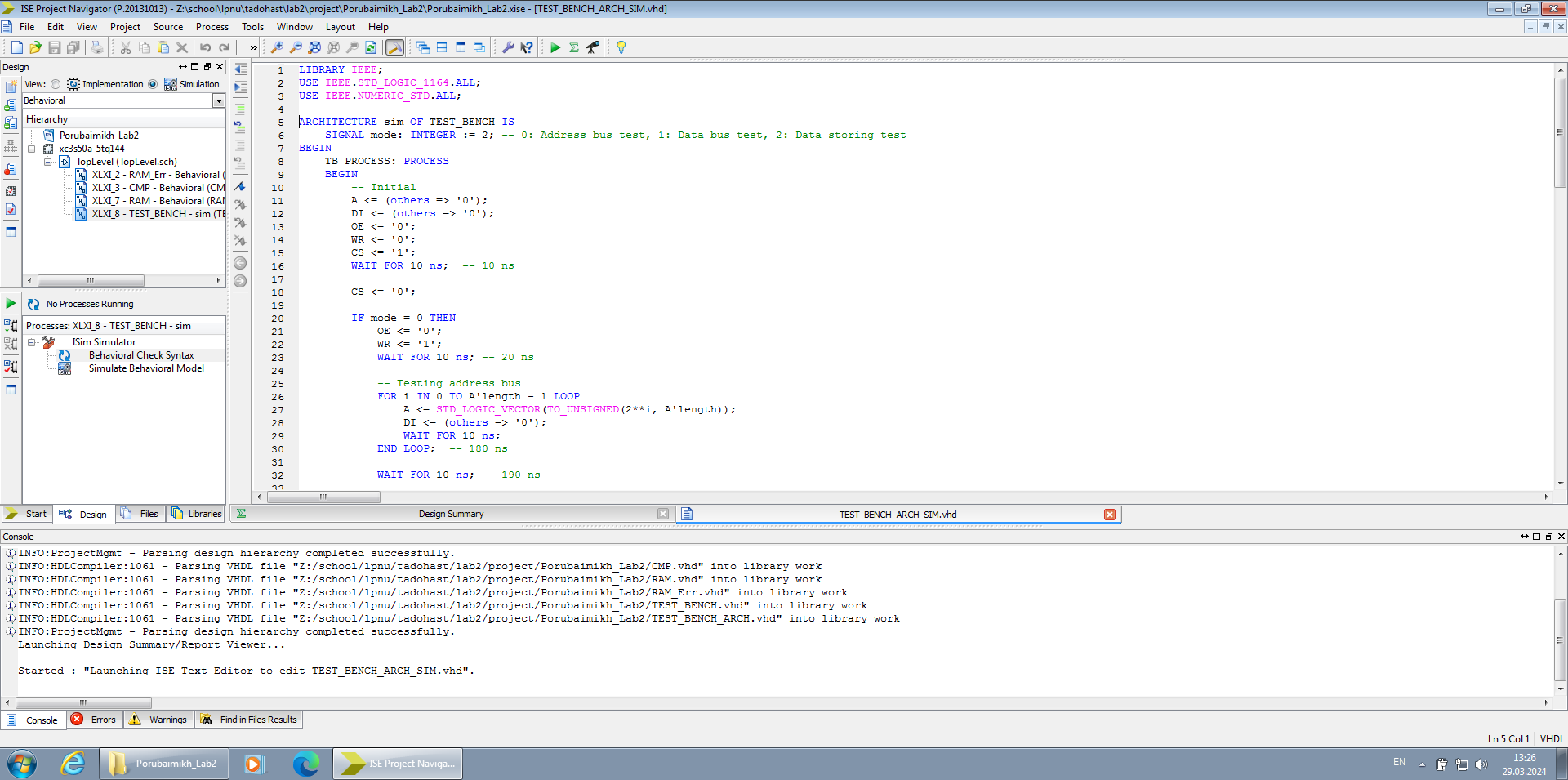


Рис. 4.13. Опис архітектури компонента TEST\_BENCH для симуляції.

* 1. Схема.

Процес створення файлу схеми наведено на Рис. 4.14. Готову схему наведено на рис. 4.15.

Як видно, на схемі присутній генератор тествоих послідовностей, що генераує вхідні значення для схем ОЗП. Кожний вихід генератора підключенно до відповідого маркеру виводу для зручного тестування. На схемі присутні дві схеми ОЗП: еталонний (RAM) і ОЗП з помилкою (RAM\_Err). Виходи кожного з них під’єднано до вузла порівняння, а також до маркерів виводу для зручного тестування. Вузол порівняння порівнює два вхідних сигнали, і, якщо вони не співпадають, виводить логічну одиницю на виході Error.

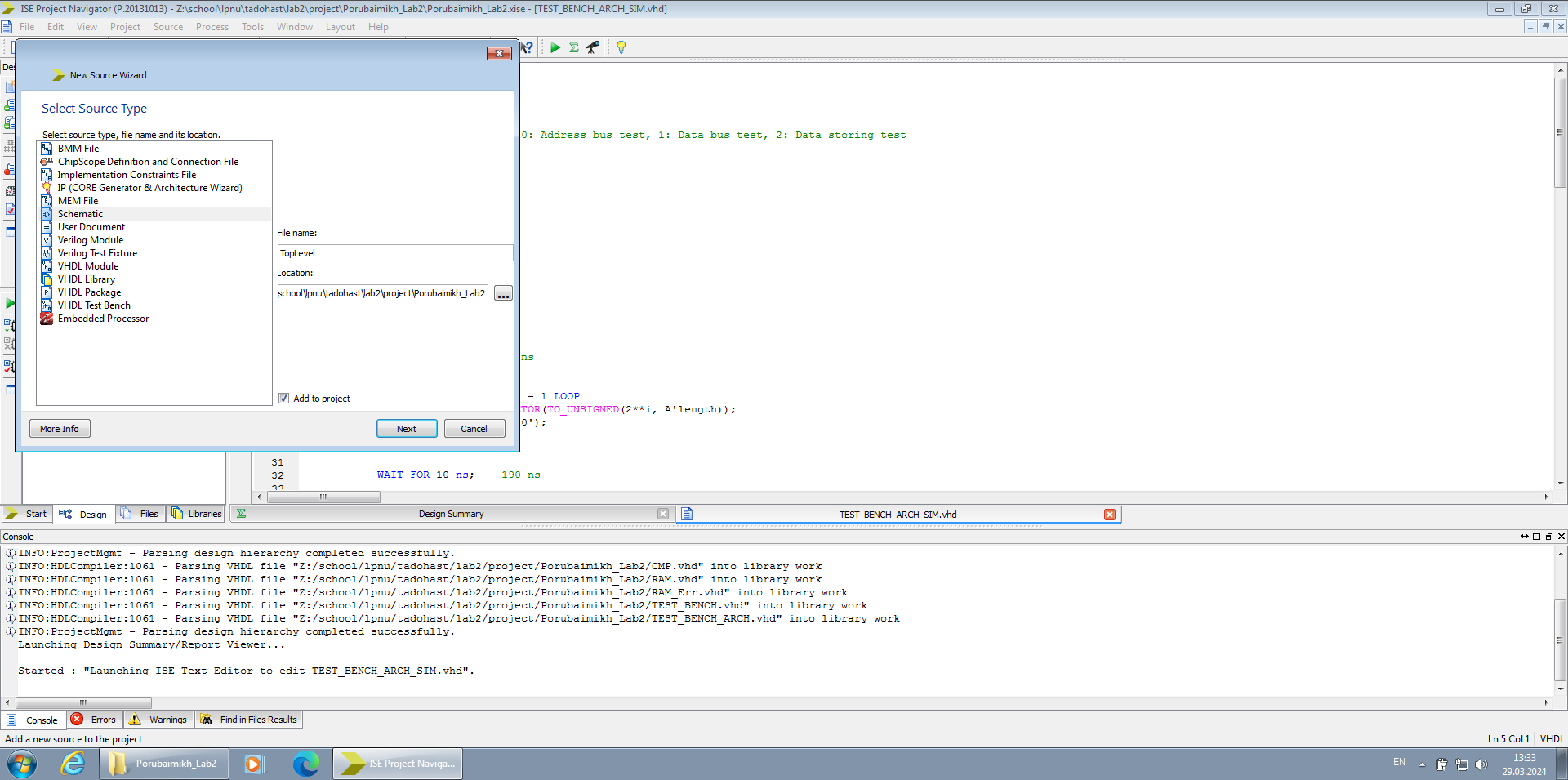


Рис. 4.14. Створення файлу схеми.

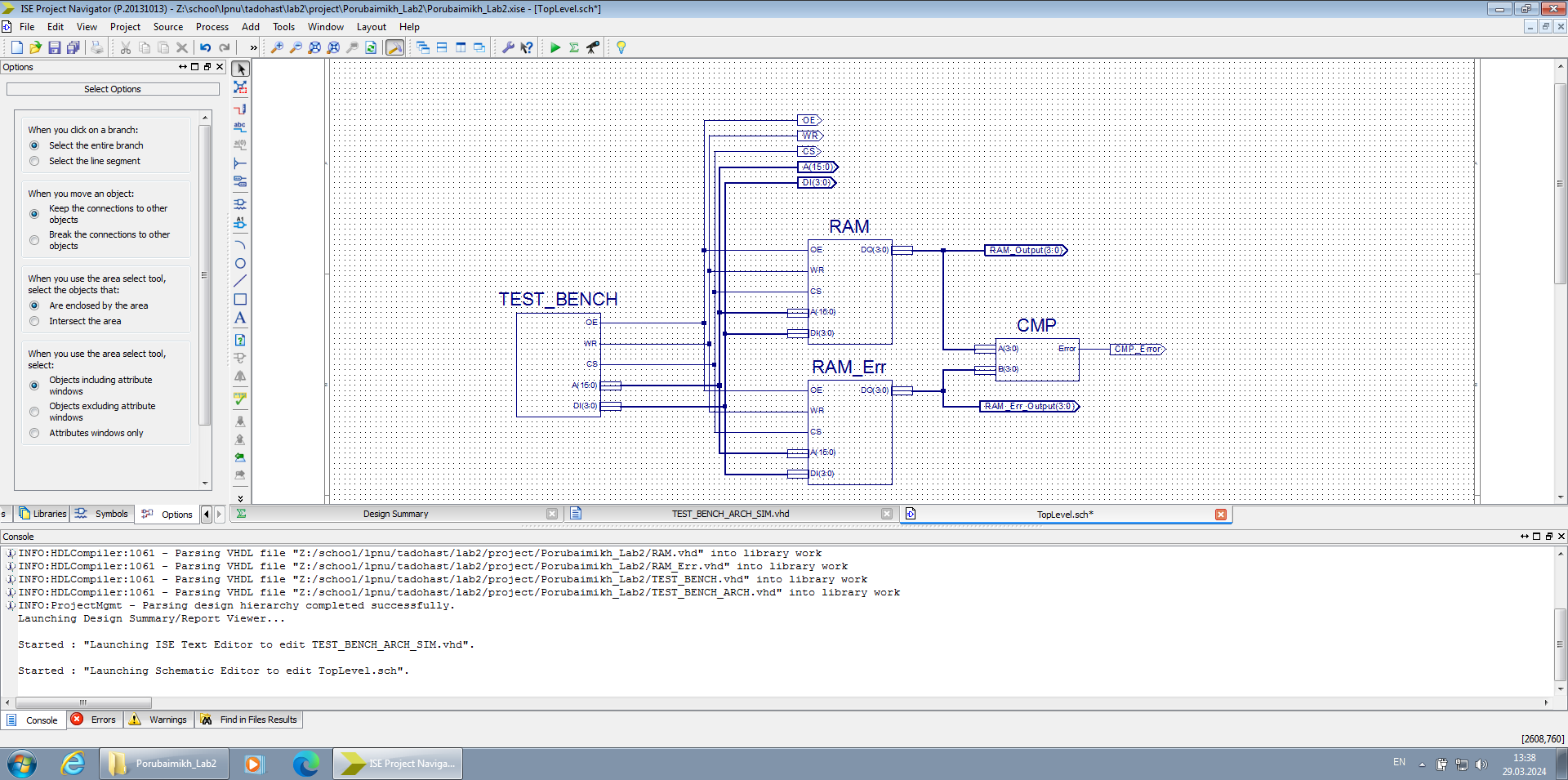


Рис. 4.15. Схема.

* 1. Процес тестування

Для виявлення помилок у схемі RAM\_Err проведено три сесії тестування: по одній для кожного з режимів тестування.

Перевірка шини адреси необхідна для того, щоб перевірити, чи немає на шині адреси закороток і обривів. Алгоритм тестування наступний: для тесту обирається певна група адрес, в кожну з яких записується «фон». Потім за першою адресою з групи записується «не фон», і виконується перевірка інших адрес з цієї групи. Якщо «не фон» знайдено в відмінній від поточної комірці, це означає, що шина адреси несправна.

В якості групи адрес обрано адреси, що відповідають набору «біжуча 1». В якості «фону» обрано значення «0000».

Часову діаграму тестування шини адреси наведено на Рис. 4.16. Як видно, під час тестування виникли певні неспівпадіння виводів, але послідовності «не фон» не знайдено в жодній з комірок.

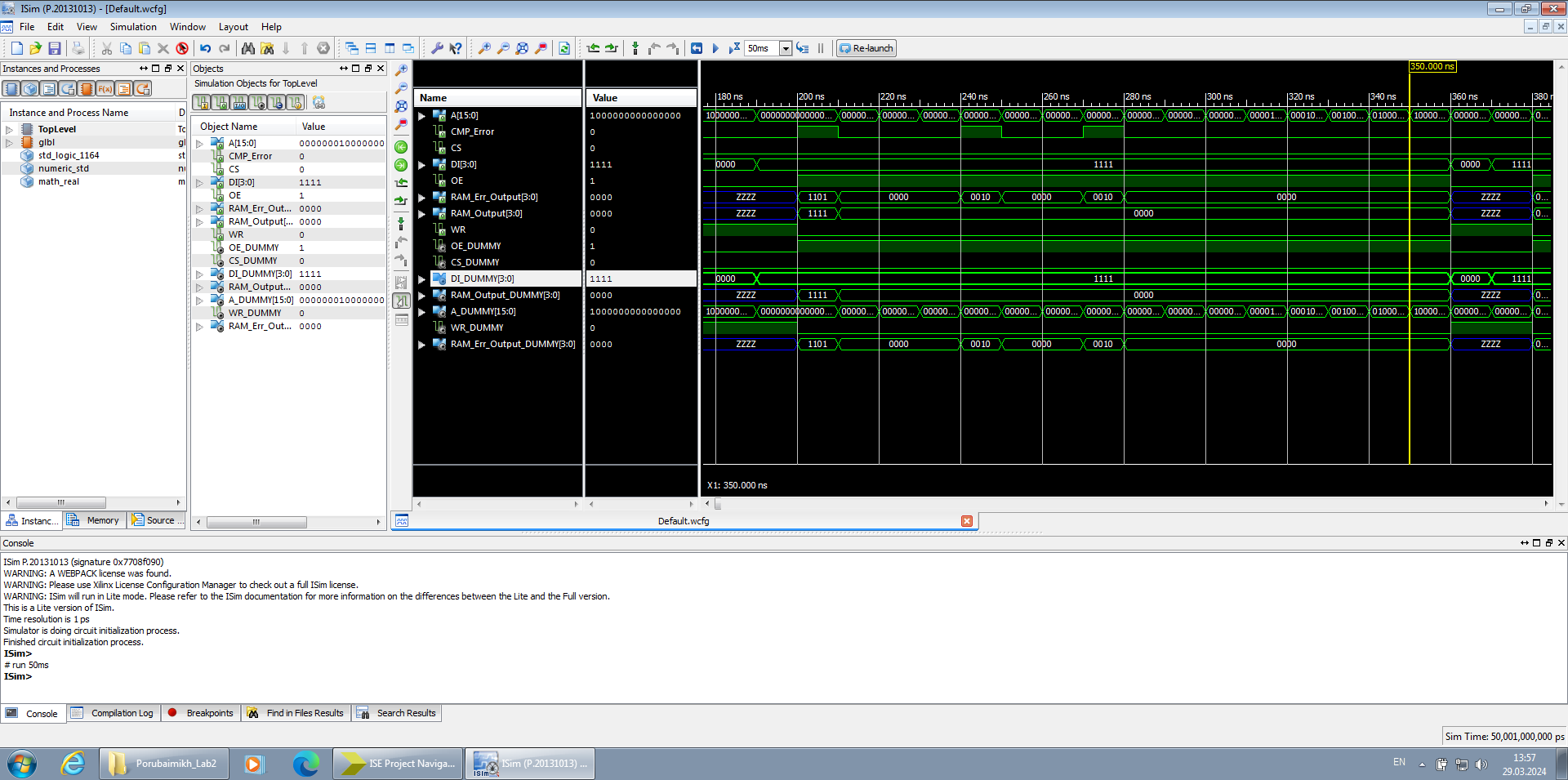


Рис. 4.16. Часова діаграма тестування шини адреси.

Для тестування шини даних необхідно, необхідно обрати одну адресу, і послідовно записати і зчитати одну з тестових послідовностей.

В якості досліджуваної адреси обрано адресу «0x0000», в яку послідовно будуть записані послідовності «біжуча 1» і «біжучий 0».

Часову діаграму тестування шини даних наведено на Рис. 4.17. Як видно, під час тестування шини даних було виявлено помилку: біт DO(1) вузла ОЗП з помилкою є несправним і за певних умов не співпадає зі значенням, що повертає еталонний ОЗП.

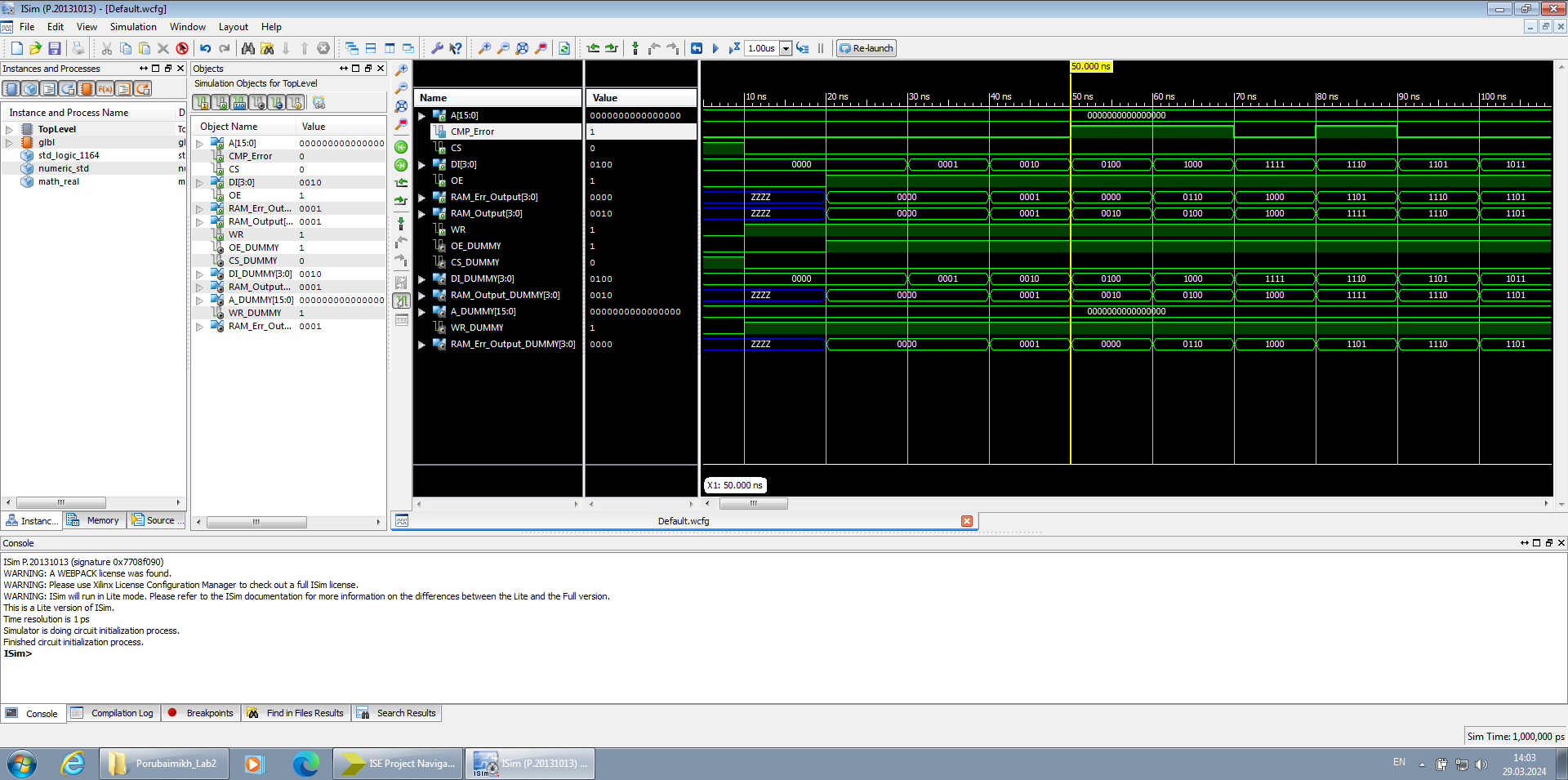


Рис. 4.17. Часова діаграма тестування шини даних.

Для тестування збереження даних необхідно до кожної з комірок ОЗП записати двійковий «0» і перевірити, що за всіма адресами зберігся саме двійковий «0». Якщо в комірці знайдено «0», тест на збереження даних не пройдено.

Часову діаграму тестування збереження даних наведено на Рис. 4.18. Як видно, результат тестування шини даних повторюється: біт DO(1) є дійсно несправним.

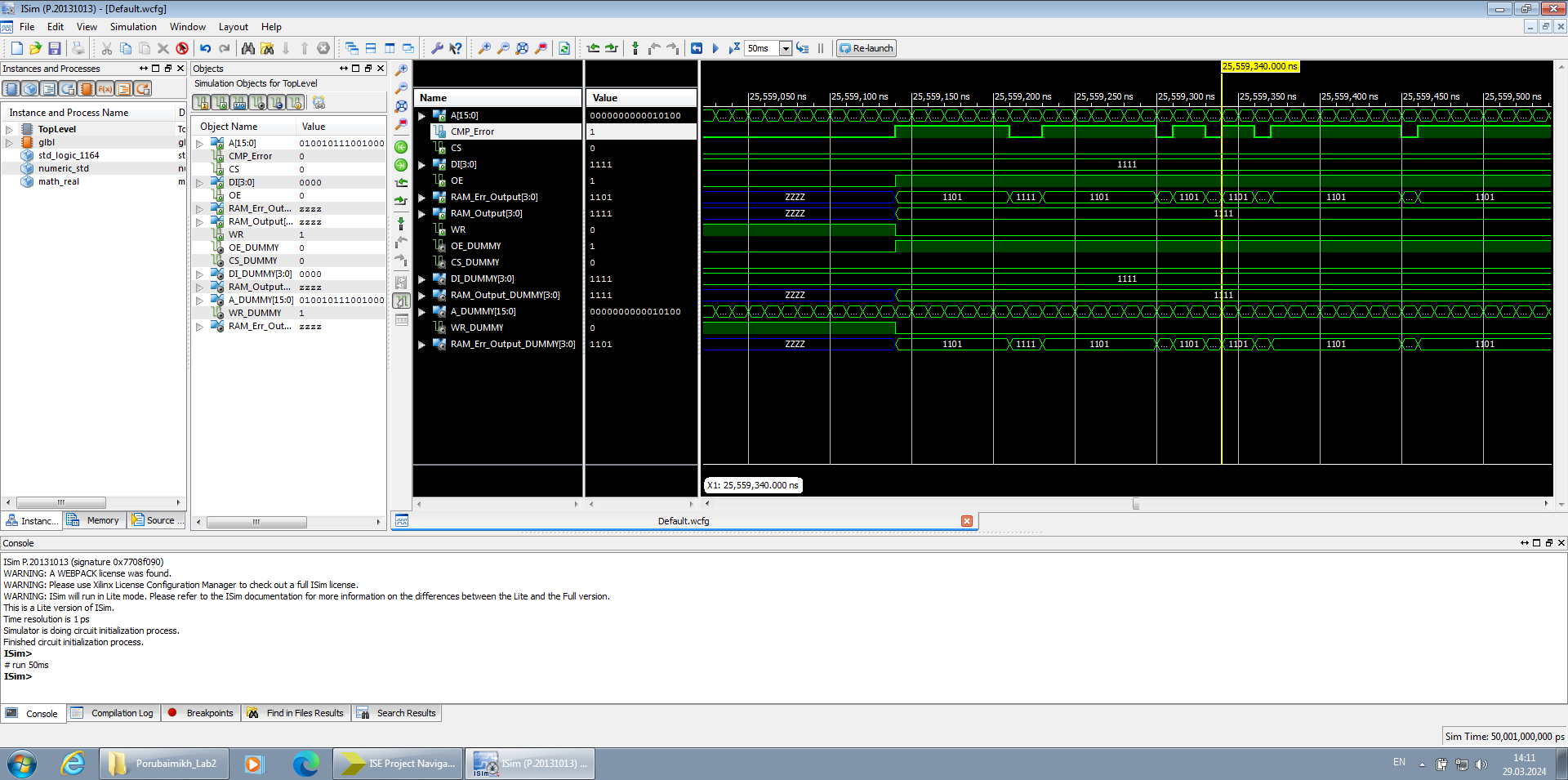


Рис. 4.18. Часова діаграма тестування збереження даних.

Висновок

Я ознайомився із загальною схемою тестування цифрової техніки. Засвоїв методи та засоби тестування цифрових схем з пам’яттю на прикладі оперативного запам’ятовуючого пристрою, використавши тестові алгоритми тестуваня шини адреси, шини даних і збереження інформації.

Список використаних джерел

1. Тестування і діагностика програмно-апаратних засобів : лабораторний практикум для студентів спеціальності 123 “Комп’ютерна інженерія” / В. С. Глухов, М. О. Хомуляк, Г. В. Бойко, І. М. Жолубак. – Львів : Видавництво Національного університету “Львівська політехніка”, 2021. – 120 с
2. ISE In-Depth Tutorial <https://docs.xilinx.com/v/u/en-US/ise_tutorial_ug695> 25.10.2023

Додаток А. Код вузла RAM.

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.NUMERIC\_STD.ALL;

ENTITY RAM IS

PORT (

A : IN STD\_LOGIC\_VECTOR(15 DOWNTO 0); -- Address input

DI : IN STD\_LOGIC\_VECTOR(3 DOWNTO 0); -- Data input

OE : IN STD\_LOGIC; -- Output enable

WR : IN STD\_LOGIC; -- Write enable

CS : IN STD\_LOGIC; -- Chip select

DO : OUT STD\_LOGIC\_VECTOR(3 DOWNTO 0) -- Data output

);

END RAM;

ARCHITECTURE Behavioral OF RAM IS

TYPE ram\_type IS ARRAY (0 TO 65535) OF STD\_LOGIC\_VECTOR(3 DOWNTO 0);

SIGNAL memory : ram\_type;

BEGIN

PROCESS (A, DI, OE, WR, CS)

BEGIN

IF CS = '0' THEN

IF OE = '1' THEN

DO <= memory(TO\_INTEGER(unsigned(A)));

ELSE

DO <= (OTHERS => 'Z');

END IF;

IF WR = '1' THEN

memory(TO\_INTEGER(unsigned(A))) <= DI;

END IF;

ELSE

DO <= (OTHERS => 'Z');

END IF;

END PROCESS;

END Behavioral;

Додаток Б. Код вузла RAM\_Err.

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.MATH\_REAL.ALL;

USE IEEE.NUMERIC\_STD.ALL;

ENTITY RAM\_Err IS

PORT (

A : IN STD\_LOGIC\_VECTOR(15 DOWNTO 0); -- Address input

DI : IN STD\_LOGIC\_VECTOR(3 DOWNTO 0); -- Data input

OE : IN STD\_LOGIC; -- Output enable

WR : IN STD\_LOGIC; -- Write enable

CS : IN STD\_LOGIC; -- Chip select

DO : OUT STD\_LOGIC\_VECTOR(3 DOWNTO 0) -- Data output

);

END RAM\_Err;

ARCHITECTURE Behavioral OF RAM\_Err IS

TYPE RAM\_TYPE IS ARRAY (0 TO 65535) OF STD\_LOGIC\_VECTOR(3 DOWNTO 0);

SIGNAL RAM : RAM\_TYPE;

BEGIN

PROCESS (A, DI, OE, WR, CS)

VARIABLE RN : REAL;

VARIABLE S1, S2: POSITIVE := 1;

BEGIN

IF CS = '0' THEN

IF WR = '1' THEN

RAM(TO\_INTEGER(unsigned(A))) <= DI;

END IF;

IF OE = '1' THEN

DO <= RAM(TO\_INTEGER(unsigned(A)));

--- Error injection

UNIFORM(S1, S2, RN);

IF RN < 0.1 THEN

DO(1) <= '1';

ELSE

DO(1) <= '0';

END IF;

ELSE

DO <= (OTHERS => 'Z');

END IF;

ELSE

DO <= (OTHERS => 'Z');

END IF;

END PROCESS;

END Behavioral;

Додаток В. Код вузла CMP.

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.NUMERIC\_STD.ALL;

ENTITY CMP IS

PORT (

A : IN STD\_LOGIC\_VECTOR(3 DOWNTO 0); -- Input A

B : IN STD\_LOGIC\_VECTOR(3 DOWNTO 0); -- Input B

Error : OUT STD\_LOGIC -- Error output

);

END CMP;

ARCHITECTURE Behavioral OF CMP IS

BEGIN

PROCESS (A, B)

BEGIN

IF A = B THEN

Error <= '0'; -- A and B are equal

ELSE

Error <= '1'; -- A and B are not equal

END IF;

END PROCESS;

END Behavioral;

Додаток Г. Код компонента TEST\_BENCH.

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.NUMERIC\_STD.ALL;

ENTITY TEST\_BENCH IS

PORT (

A : OUT STD\_LOGIC\_VECTOR(15 DOWNTO 0); -- Address

DI : OUT STD\_LOGIC\_VECTOR(3 DOWNTO 0); -- Data input

OE : OUT STD\_LOGIC; -- Output enable

WR : OUT STD\_LOGIC; -- Write enable

CS : OUT STD\_LOGIC -- Chip select

);

END TEST\_BENCH;

Додаток Д. Код архітектури компонента TEST\_BENCH для синтезу.

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.NUMERIC\_STD.ALL;

ARCHITECTURE real OF TEST\_BENCH IS

BEGIN

END real;

Додаток Е. Код архітектури компонента TEST\_BENCH для симуляції.

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.NUMERIC\_STD.ALL;

ARCHITECTURE sim OF TEST\_BENCH IS

SIGNAL mode: INTEGER := 0; -- 0: Address bus test, 1: Data bus test, 2: Data storing test

BEGIN

TB\_PROCESS: PROCESS

BEGIN

-- Initial

A <= (others => '0');

DI <= (others => '0');

OE <= '0';

WR <= '0';

CS <= '1';

WAIT FOR 10 ns; -- 10 ns

CS <= '0';

IF mode = 0 THEN

OE <= '0';

WR <= '1';

WAIT FOR 10 ns; -- 20 ns

-- Testing address bus

FOR i IN 0 TO A'length - 1 LOOP

A <= STD\_LOGIC\_VECTOR(TO\_UNSIGNED(2\*\*i, A'length));

DI <= (others => '0');

WAIT FOR 10 ns;

END LOOP; -- 180 ns

WAIT FOR 10 ns; -- 190 ns

FOR i IN 0 TO A'length - 1 LOOP

A <= STD\_LOGIC\_VECTOR(TO\_UNSIGNED(2\*\*i, A'length));

DI <= (others => '1');

WAIT FOR 10 ns;

OE <= '1';

WR <= '0';

FOR j IN 0 TO A'length - 1 LOOP

A <= STD\_LOGIC\_VECTOR(TO\_UNSIGNED(2\*\*j, A'length));

WAIT FOR 10 ns;

END LOOP;

OE <= '0';

WR <= '1';

A <= STD\_LOGIC\_VECTOR(TO\_UNSIGNED(2\*\*i, A'length));

DI <= (others => '0');

WAIT FOR 10 ns;

END LOOP;

ELSIF mode = 1 THEN

-- Testing data bus (Address = 0x0000)

A <= "0000000000000000";

OE <= '0';

WR <= '1';

CS <= '0';

WAIT FOR 10 ns;

OE <= '1';

DI <= "0000"; WAIT FOR 10 ns;

DI <= "0001"; WAIT FOR 10 ns;

DI <= "0010"; WAIT FOR 10 ns;

DI <= "0100"; WAIT FOR 10 ns;

DI <= "1000"; WAIT FOR 10 ns;

DI <= "1111"; WAIT FOR 10 ns;

DI <= "1110"; WAIT FOR 10 ns;

DI <= "1101"; WAIT FOR 10 ns;

DI <= "1011"; WAIT FOR 10 ns;

DI <= "0111"; WAIT FOR 10 ns;

ELSIF mode = 2 THEN

-- Data storing test

-- Write zeros

OE <= '0';

WR <= '1';

FOR i IN 0 TO 65535 LOOP

A <= STD\_LOGIC\_VECTOR(TO\_UNSIGNED(i, A'length));

DI <= (others => '0');

WAIT FOR 10 ns;

END LOOP;

WR <= '0';

OE <= '1';

FOR i IN 0 TO 65535 LOOP

A <= STD\_LOGIC\_VECTOR(TO\_UNSIGNED(i, A'length));

WAIT FOR 10 ns;

END LOOP;

-- Write ones

WR <= '1';

OE <= '0';

FOR i IN 0 TO 65535 LOOP

A <= STD\_LOGIC\_VECTOR(TO\_UNSIGNED(i, A'length));

DI <= (others => '1');

WAIT FOR 10 ns;

END LOOP;

WR <= '0';

OE <= '1';

FOR i IN 0 TO 65535 LOOP

A <= STD\_LOGIC\_VECTOR(TO\_UNSIGNED(i, A'length));

WAIT FOR 10 ns;

END LOOP;

END IF;

WAIT;

END PROCESS TB\_PROCESS;

END sim;