ДОДАТОК А
Код управляючого скрипта

```
import os
import sys
import openpyxl
from TestModule import TestModule

if len(sys.argv) == 2 and sys.argv[1] == "help":
    print(f"{sys.argv[0]} \"path to input excel file\" \"path to folder with
variants files\"")
    sys.exit()
elif len(sys.argv) == 3 and isinstance(os.sys.argv[1], str) and
isinstance(os.sys.argv[2], str):
    script = sys.argv[0]
    xl_path = sys.argv[1]
    vars_path = sys.argv[2]
else:
    print("Invalid input parameters. Enter \"help\" for more information.")
    sys.exit()

xl_path_res = xl_path.replace(".xls", "_result.xls")

if not os.path.exists(xl_path) :
    print("Table path does not exist!")
    sys.exit()

if not os.path.exists(vars_path) :
    print("Vars path does not exist!")
    sys.exit()

testModule = TestModule(vars_path)

sheet = openpyxl.load_workbook(filename = xl_path).active

if os.path.exists(xl_path_res):
    xl = openpyxl.load_workbook(filename = xl_path_res)
    xls = xl.active
    xls.delete_cols(1, 3)
    xl.save(xl_path_res)
else:
    res_sheet = openpyxl.Workbook()
    res_sheet.save(xl_path_res)

res_book = openpyxl.load_workbook(filename = xl_path_res)
res_sheet = res_book.active

res_sheet.cell(row=1, column=1).value="Surname"
res_sheet.cell(row=1, column=2).value="Variant"
res_sheet.cell(row=1, column=3).value="Result"

i = 2
while i <= sheet.max_row:
    surname = sheet.cell(row=i, column=1).value
    variant = sheet.cell(row=i, column=2).value

    res_sheet.cell(row=i, column=1).value=surname
    res_sheet.cell(row=i, column=2).value=variant
```

```
    folder_path = sheet.cell(row=i, column=3).value

    if not (isinstance(folder_path, str) and isinstance(variant, int)):
        i += 1
        continue

    if not os.path.exists(folder_path):
        res_sheet.cell(row=i, column=3).value = "Path is wrong"
        res_err = "Path is wrong"
        i += 1
    else:
        res_err = testModule.CheckProject(folder_path, variant)
        res_sheet.cell(row=i, column=3).value = res_err
        i += 1
    print(f"{surname}\t\tvar_{variant}\t\t{res_err}")

print("All variants are checked!")
res_book.save(xl_path_res)
```

ДОДАТОК Б

Код модуля тестування

```
import os
import os
import re
import sys
import json
import subprocess


var_not_exist = "Vars path does not exist!"
proj_not_exist = "Some of project files does not exist!"

class TestModule:

    def __init__(self, vars_path):
        if not os.path.exists(vars_path) :
            print(var_not_exist)
            sys.exit()
        self.var_path = vars_path


    def CheckProject(self, proj_path, var):
        var_path = f"{self.var_path}\\var{var}.json"
        des_path = f"{proj_path}\\design.vhd"
        test_path = f"{proj_path}\\testbench.vhd"

        if not os.path.exists(var_path) :
            print(var_not_exist)
            return var_not_exist

        if not (os.path.exists(des_path) and os.path.exists(test_path)):
            print(proj_not_exist)
            return proj_not_exist

        if not self.__ParseForVariant(proj_path, var_path):
            return "Variant does not mach"

        if not self.__CompileProject(proj_path):
            return "Compiled with errors"

        return "All checks passed"


    def __ParseForVariant(self, proj_path, var_path):

        json_content = json.load(open(var_path))

        model = json_content["model"]

        model_content = open(f"{proj_path}/design.vhd").read()

        for word in model["keywords"]:
            if word not in model_content:
               return False
```

```
        if not (self.__CheckInOut(model_content, model["in"],
model["in_type"], "[ :]in ")
         and self.__CheckInOut(model_content, model["out"],
model["out_type"], "[ :]out ")
         and self.__CheckSignals(model_content, model["signals"],
model["signals_type"])):
            return False

        testbench = json_content["testbench"]

        testbench_content = open(f"{proj_path}/testbench.vhd").read()

        for word in testbench["keywords"]:
            if word not in testbench_content:
               return False

        if not (self.__CheckInOut(testbench_content, testbench["in"],
testbench["in_type"], "[ :]in ")
         and self.__CheckInOut(testbench_content, testbench["out"],
testbench["out_type"], "[ :]out ")
         and self.__CheckSignals(testbench_content, testbench["signals"],
testbench["signals_type"])):
           return False

        return True


    def __CheckInOut(self, file_content, in_list, types_list, pattern):
        i = len(re.findall(pattern, file_content))
        in_cpy = in_list.copy()
        find_index = 0

        while i:
            reg_exp_res = re.search(pattern, file_content[find_index:])
            type_str = file_content[reg_exp_res.end():file_content.find(";",
reg_exp_res.end())]
            in_str = file_content[file_content.rfind("\n", 0,
reg_exp_res.start()): reg_exp_res.start()]

            cnt_list = len(types_list)
            for t in types_list:
                if not type_str.upper().find(t.upper()) == -1:
                    types_list.remove(t)
                    break

            if len(types_list) == cnt_list:
                return False

            for v in in_list:
                if not re.search(("[ (,]" + v.upper() + "[ :,]"),
in_str.upper()) == None:
                    in_cpy.remove(v)

            find_index = reg_exp_res.end
            i -= 1

        if len(in_cpy) == 0:
            return True
        else:
            return False
```

```
    def __CheckSignals(self, file_content, sig_list, types_list):
        pattern = " signal "
        i = len(re.findall(pattern, file_content))
        sig_cpy = sig_list.copy()
        find_index = 0

        while i:
            find_index = file_content.find(pattern, find_index)
            type_str = file_content[file_content.find(":",
find_index):file_content.find(";", find_index)]
            sig_str = file_content[find_index + len(pattern) - 1 :
file_content.find(":", find_index) + 1]

            cnt_list = len(types_list)

            for t in types_list:
                if not type_str.upper().find(t.upper()) == -1:
                    types_list.remove(t)
                    break

            if len(types_list) == cnt_list:
                return False

            for v in sig_list:
                if not re.search(("[ ,]" + v.upper() + "[ :,]"),
sig_str.upper()) == None:
                    sig_cpy.remove(v)


            find_index += len(pattern)
            i -= 1
        if len(sig_cpy) == 0:
            return True
        else:
            return False


      def __CompileProject(self, proj_path):

        design_path = proj_path + r"\design.vhd"
        testbench_path = proj_path + r"\testbench.vhd"

        f = open(testbench_path, "r")
        testbench_cont = f.read()
        f.close()

        entity_name_beg = testbench_cont.find("entity ") + len("entity")
        entity_name_end = testbench_cont.find(" is", entity_name_beg)
        entity_name = testbench_cont[entity_name_beg:entity_name_end]

        sim_duration = 500

        macro_file = f"{proj_path}/macro.do"

        if os.path.exists(macro_file):
            os.remove(macro_file)

        f = open(macro_file, "x")
        f.write(f"alib  work\r\nset worklib work\r\nacom {design_path}
{testbench_pat}\r\nasim {entity_name}\r\nrun {sim_duration}ns\r\nendsim")
        f.close()
```

```
        cmd = f"vsimsa.bat -do {macro_file}"
        PIPE = subprocess.PIPE
        p = subprocess.Popen(cmd, shell=True, stdin=PIPE, stdout=PIPE,
stderr=subprocess.STDOUT)

        out = str(p.stdout.read(), "cp866")

        if out.find(" 0 Errors") < 0:
            return False

        if out.find("KERNEL: Simulation has finished.") < 0:
            return False
        else:
            return True
```

ДОДАТОК В

Графічний матеріал атестаційної роботи

Харківський національний університет радіоелектроніки

КАФЕДРА АПОТ

Система автоматизованого тестування великої кількості VHDL-проектів

Атестаційна робота
Другий (магістерський) рівень

Виконала
ст. гр. СКСм-19-1
Лопатіна А.О.

Керівник
проф. каф. АПОТ
Хаханова І.В.

Актуальність розробки

Необхідність автоматизації перевірки великої кількості завдань студентів та прискорення отримання результатів перевірки

Мета і задачі роботи

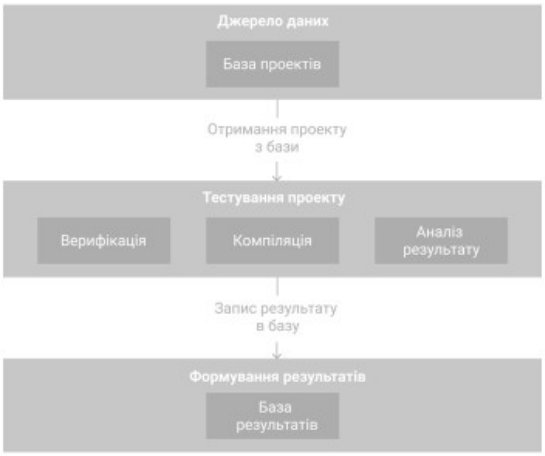
- Метою атестаційної роботи є розробка системи автоматизованого тестування великої кількості VHDL-проектів.
- Для досягнення мети були поставлені такі задачі, як:
  - аналіз методів перевірки проектів;
  - розгляд структури системи та основних етапів автоматизації;
  - розробка алгоритму автоматизованого тестування;
  - аналіз засобів розробки автоматизованої системи;
  - розробка програмного забезпечення для автоматизованої системи.

Методи перевірки проекту

- Аналіз коду на відповідність варіанту
- Компіляція проекту та аналіз результатів

Структура автоматизованої системи

5



6

Опис моделі

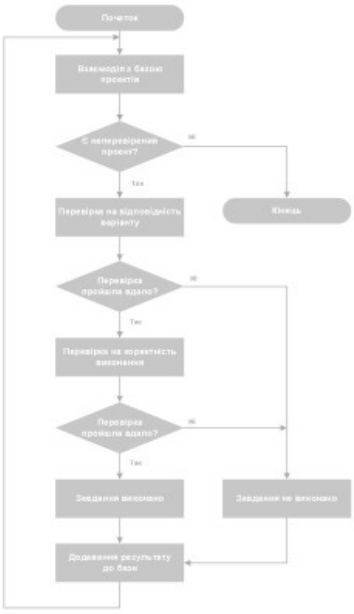
```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity schema is
port (x: in std_logic;
      y: in std_logic;
      z: in std_logic;
      f: out std_logic);
end schema;

architecture arch of schema is
signal a,b: std_logic;
begin
  a <= x and not y after 10 ns;
  b <= y and z after 10 ns;
  f <= a or b after 3 ns;
end arch;
```

Testbench

```
library ieee;
use ieee.std_logic_1164.all;

entity schema_tb is
end schema_tb;

architecture arch_tb of schema_tb is
  component schema is
  port (x, y, z: in std_logic;
  f: out std_logic);
end component;

signal test : std_logic_vector(2 downto 0);
signal Res : std_logic;

begin
UUT: schema
port map (x => test(2),y => test(1), z => test(0), f => Res);
test <= "000",
    "001" after 15 ns,
    "010" after 30 ns,
    "011" after 45 ns,
    "100" after 60 ns,
    "101" after 75 ns,
    "110" after 90 ns,
    "111" after 105 ns;
end arch_tb;
```

Структура VHDL-моделі та VHDL-Testbench

Алгоритм роботи



Приклад бази проектів та результатів

	A	B	C
1	Surname	Variant	Path
2	Karnitskaya	1	D:\Testing\Task1\Karnitskaya _var1
3	Hvorostovsky	3	D:\Testing\Task1\Hvorostovsky_var3
4	Hasymov	6	D:\Testing\Task1\Hasymov_var6
5	Tatishev	11	D:\Testing\Task1\Tatishev_var11
6	Dzhumayev	16	D:\Testing\Task1\Dzhumayev_var16
7	Tsvetaeva	4	D:\Testing\Task1\Tsvetaeva_var4
8	Chistovich	7	D:\Testing\Task1\Chistovich_var7
9	Vasilyev	2	D:\Testing\Task1\Vasilyev _var2
10	Georgiyev	8	D:\Testing\Task1\Georgiyev_var8
11	Dudinsky	10	D:\Testing\Task1\Dudinsky_var10
12	Yeremeyeva	9	D:\Testing\Task1\Yeremeyeva_var9
13	Vorobyov	5	D:\Testing\Task1\Vorobyov_var5
14	Zaytsev	13	D:\Testing\Task1\Zaytsev_var13
15	Lapshin	12	D:\Testing\Task1\Lapshin_var12

База проектів

	A	B	C
1	Surname	Variant	Result
2	Karnitskaya	1	All checks passed
3	Hvorostovsky	3	All checks passed
4	Hasymov	6	Variant does not match
5	Tatishev	11	All checks passed
6	Dzhumayev	16	All checks passed
7	Tsvetaeva	4	Compiled with errors
8	Chistovich	7	All checks passed
9	Vasilyev	2	Variant does not match
10	Georgiyev	8	Variant does not match
11	Dudinsky	10	Compiled with errors
12	Yeremeyeva	9	All checks passed
13	Vorobyov	5	All checks passed
14	Zaytsev	13	Compiled with errors
15	Lapshin	12	All checks passed

База результатів

Шаблони перевірки

```
{
  "model": {
    "keywords": [
      "library",
      "use",
      "entity",
      "port",
      "architecture",
      "signal",
      "begin",
      "process"
    ],
    "in_type": "std_logic",
    "in": [ "a", "b", "c" ],
    "out_type": "std_logic",
    "out": [ "f" ],
    "signals_type": "std_logic",
    "signals": [ "a0", "b0", "c0" ]
  },
  "testbench": {
    "keywords": [
      "library",
      "use",
      "entity",
      "port",
      "architecture",
      "component",
      "signal",
      "begin",
      "process",
      "map"
    ],
    "in_type": "std_logic",
    "in": [ "a", "b", "c" ],
    "out_type": "std_logic",
    "out": [ "f" ],
    "signals_type": "std_logic",
    "signals": [ "a0", "b0", "c0" ]
  }
}
```


Тестування програмного забезпечення

	A	B	C
1	Surname	Variant	Path
2	Karnitskaya	1	D:\Testing\Task1\Karnitskaya _var1
3	Hvorostovsky	3	D:\Testing\Task1\Hvorostovsky_var3
4	Hasymov	6	D:\Testing\Task1\Hasymov_var6
5	Tatishev	11	D:\Testing\Task1\Tatishev_var11
6	Dzhumayev	16	D:\Testing\Task1\Dzhumayev_var16
7	Tsvetaeva	4	D:\Testing\Task1\Tsvetaeva_var4
8	Chistovich	7	D:\Testing\Task1\Chistovich_var7

```
C:\Users\Admin\Project>main.py D:\Testing\task1.xlsx D:\Testing\Task1_Vars
Karnitskaya       var_1       All checks passed
Hvorostovsky      var_3       All checks passed
Hasymov           var_6       Variant does not match
Tatishev          var_11      Variant does not match
Dzhumayev         var_16      All checks passed
Tsvetaeva         var_4       Compiled with errors
Chistovich        var_7       All checks passed
All variants are checked!
```

	A	B	C
1	Surname	Variant	Result
2	Karnitskaya	1	All checks passed
3	Hvorostovsky	3	All checks passed
4	Hasymov	6	Variant does not match
5	Tatishev	11	Variant does not match
6	Dzhumayev	16	All checks passed
7	Tsvetaeva	4	Compiled with errors
8	Chistovich	7	All checks passed

Висновки

На основі поставлених вимог і проведених досліджень була розроблена система, що дозволяє автоматизувати процес перевірки проектів, написаних на мові VHDL.

Розроблена система має такі переваги, як:
- автоматизування перевірки великої кількості завдань;
- прискорення отримання результатів перевірки;
- можливість масштабування системи залежно від вимог викладача.

В майбутньому пропонується удосконалити систему шляхом додавання можливості автоматизованого збереження результатів в онлайн сервісах (наприклад, Google Таблиці), що підвищить ефективність системи та дозволить студентом мати доступ до результатів тестування відразу після перевірки.

Відомість атестаційної роботи

№	Позначення	Найменування	Дод. відомості
		Текстові документи	
1	ГЮІК. ХХХХХХ.011 ПЗ	Пояснювальна записка	82 с.
		Графічні документи	
2		Слайд-презентація	11 слайдів
		Носії інформації	
3		Компакт-диск (CD)	1 шт.

Змін.	Арк.	Номер докум.	Підп.	Дата	ГЮІК. ХХХХХХ.011 ВД			
Розроб.		Лопатина А.О.		16.12	Система автоматизованого тестування великої кількості VHDL-проектів	Літ	Аркуш	Аркушів
Перевір.		Хаханова І.В.		16.12		У		
Н.контр.		Рожнова Т.Г..		18.12.2020	Відомість атестаційної роботи	ХНУРЕ Кафедра АПОТ		
Затв.		Чумаченко С.В.						