Проектиране на вградени автомобилни електронни системи
Лабораторно упражнение №1

Работа с Xilinx Vivado и Vitis. Синтезиране на микропроцесорна система върху FPGA. Работа с входно-изходен порт GPIO.

==========

1. Да се разучи вътрешната структура на програмируемата логическа матрица xc7z010-1clg400 от фамилията Zynq 7000 на фирмата Xilinx (базирана на два ARM Cortex-A9).

2. Да се разучи принципната схема на демо платката Zybo на фирмата Digilent.

3. Превключете джъмпера вдясно на платката на позиция JTAG. Свържете µUSB кабел към PROG/UART USB куплунга. Включете платката от ключа ON/OFF.

4. От страничния панел на Ubuntu изберете бутон “Show Applications”, след което в полето “Type to search” напишете Vivado и натиснете с ляв бутон на мишката иконката на програмата.

5. Create Project → Next → Project name: 01_gpio → Next → RTL Project + "Do not specify sources at this time" → Next → таб Boards: избира се Zybo (не Zybo Z7-10, не Zybo Z7-20, а само Zybo) → Next → Finish

ЗАБЕЛЕЖКА: работната маса с платка Zybo Z7-10 (вдясно на Етернет куплунга трябва да има 2 HDMI конектора; ако има един HDMI и един VGA значи, че е само Zybo) трябва да избере Zybo Z7-10 от това меню.

ВНИМАНИЕ: при първоначалната инсталация може да се наложи демо платката да се добави във Vivado. От прозореца “Boards” трябва да се натисне “Install/Update Boards” → Избира се производител → Избира се конкретна демо платка и се натиска бутон Install.

ВНИМАНИЕ: демо платката Zybo има три варианта: Zybo (класическа), Zybo Z7-10 и Zybo Z7-20. В настоящото лабораторно се използва класическата Zybo, която може да се познае по VGA конектора.

6. Вляво → Flow navigator → Create block design → OK

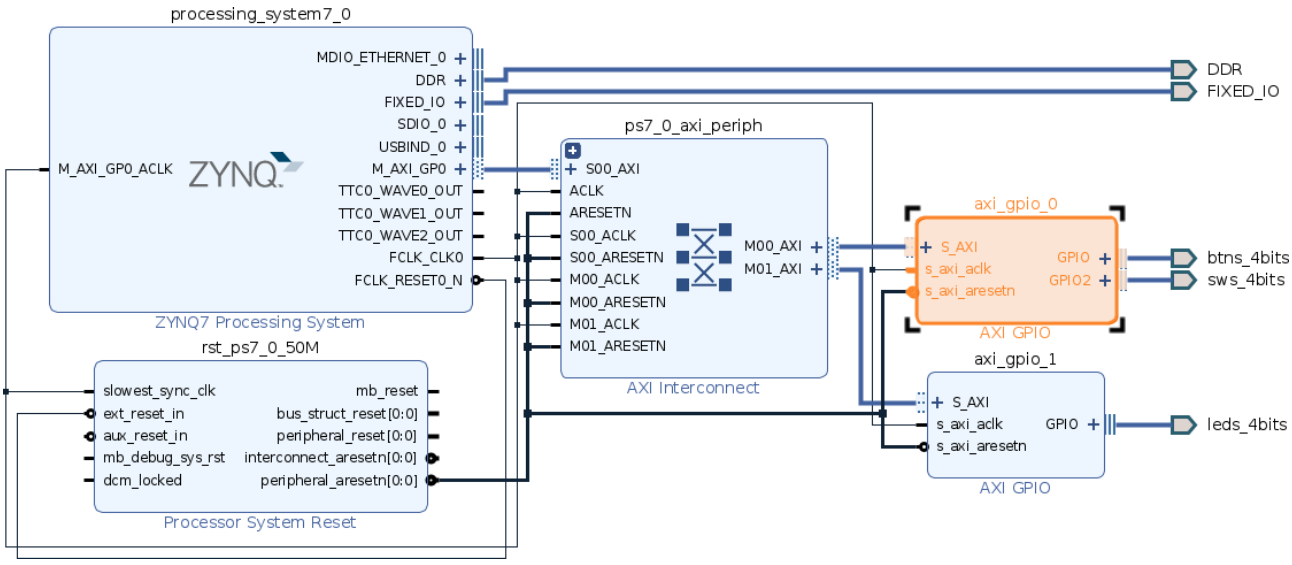
7. Вдясно → Diagram → right-click → Add IP → Search → ZYNQ7 Processing System → double click

8. Вдясно → Diagram → натиска се и се задържа ляв бутон върху FCLK_CLK0 сигнала и се свързва с M_AXI_GP0_ACLK, след това се пуска левия бутон.

9. Вдясно → Diagram → right-click → Add IP → Search → AXI GPIO → double click
10. Щраква се два пъти върху новопоставения блок axi_gpio_0 → IP Configuration → поставя се отметка на “Enable Dual Channel” → OK.
11. Добавя се още един GPIO модул → Вдясно → Diagram → right-click → Add IP → Search → AXI GPIO → double click. Този път се оставя блокът да е Single channel.

12. Вдясно →Diagram → зелена лента → Designer Assitance available -> Run Connection Automation → Слага се отметка на "All Automation":
→ Избира се GPIO подраздела на axi_gpio_0 → Options: btns_4bits
→ Избира се GPIO2 подраздела на axi_gpio_0 → Options: sws_4bits
→ Избира се GPIO подраздела на axi_gpio_1 → Options: leds_4bits
Натиска се OK.

13. Подрежда се блоковата схема с бутон Regenerate Layout



14. Вдясно →Diagram → зелена лента → Designer Assitance available -> Run Block Automation → Слага се отметка на "All Automation":

15. Вдясно→Diagram→лента с бутони → Validate Design (F6) -> "Validation successful. There are no errors or critical warnings in this design." -> OK

16. Централно→в Block design прозореца, натиска се таб-а Sources → Design sources → right-click на design_1.bd → Create HDL Wrapper (създава VHDL описание на новосъздадената система) → Let Vivado manage wrapper and auto-

update → OK

17. Вляво → Flow navigator → Generate bitstream → Yes → OK → изчаква се няколко минути (докато завърши синтеза) → View reports → OK →Ако излезе прозорец Xilinx survey - No → OK

ВНИМАНИЕ: долу, централно, в таб Log може да наблюдавата съобщенията от синтеза. Най-горе, вдясно на Vivado прозореца ще видите иконка на въртящ се зелен часовник. Докато тя е видима, значи трябва да се изчака.
18. File → Export → Export hardware → Next → Include bitstream → Next → Next → Finish

==========
==========

19. От страничния панел на Ubuntu изберете бутон “Show Applications”, след което в полето “Type to search” напишете Vitis Classic и натиснете с ляв бутон на мишката иконката на програмата.

20. Избира се път до workspace за фърмуерния проект→ Launch

ВНИМАНИЕ: възможно е да има останали фърмуерни проекти от минали групи. В таб-а Explorer на средата Vitis със задържане на CTRL от клавиатурата изберете с ляв бутон на мишката всички проекти, след което натиснете десен бутон на мишката и Delete. Ако проектите ще се използват, махнете отметката от “Delete project contents on disk (cannot be undone)” и натиснете OK.

21. File → New → Platform project → Platform project name: 01_gpio_platform → Next → таб "Create new platform from hardware" → Browse → избира се пътя до проекта 01_gpio, създаден от Vivado → design_1_wrapper.xsa → Open → Finish

22. Вляво→Project explorer → избира се 01_gpio_platform → right-click → Build Project

23. File → New → Application project → Next → "Select a platform from repository" → Избира се 01_gpio_platform → Next → Application project name: 01_gpio_app → Next → Next → Empty application → Finish

24. Щраква се с десен бутон върху директорията src в проекта 01_gpio_app_system/01_gpio_app → New → Other → C/C++ → Source File → Next → Source file: дава се име main.c → Finish

25. В текстовия редактор на Vitis въведете следната програма:

```c
#include <stdio.h>
#include <xgpio.h>
#include "xparameters.h"
#include "sleep.h"

int main(void){
	XGpio output;

	XGpio_Initialize(&output, XPAR_AXI_GPIO_1_DEVICE_ID);

	XGpio_SetDataDirection(&output, 1, 0x0);

	while(1){
		XGpio_DiscreteWrite(&output, 1, 0x00);
		usleep(200000);
		XGpio_DiscreteWrite(&output, 1, 0x01);
		usleep(200000);
	}

	return 0;
}
```

26. Вляво, Project explorer -> избира се 01_gpio_app -> right-click -> Build project

27. Вляво, Project explorer -> избира се 01_gpio_app_system -> right-click -> Debug as -> Launch Hardware

28. Натиска се бутон Resume (F8), за да се стартира безкрайното изпълнение на програмата за мигане на светодиод (blinky).

29. За да спрете debug сесията във Vitis, натиснете бутон Disconnect.

30. Напишете програма, която мига и четирите светодиода.

31. Засветете името на някоя от XGpio функциите. Натиснете бутон F3 от клавиатурата. Така ще влезете в GPIO библиотеката на системата. Използвайки функциите оттам, напишете програма, която чете бутони BTN0 ÷ BTN3 и отразява тяхното логическо състояние чрез светодиоди LD0 ÷ LD3 (логическа 1 – светодиодът свети, логическа 0 – светодиодът е изгасен).

31. Напишете програма, която чете ключове SW0 ÷ SW3 и отразява тяхното логическо състояние чрез светодиоди LD0 ÷ LD3 (логическа 1 – светодиодът свети, логическа 0 – светодиодът е изгасен).

* * *

[1] W. Marshall, “Getting started with Zynq”, online, Oct 5, 2021.

https://digilent.com/reference/learn/programmable-logic/tutorials/zybo-getting-started-with-zynq/start

доц. д-р инж. Любомир Богданов, 2024 г.