Проектиране на вградени автомобилни електронни системи
Лабораторно упражнение №2

Работа с Xilinx Vivado и Vitis. Синтезиране на микропроцесорна система върху FPGA. Работа с контролер за прекъсвания GIC.

==========

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

2. 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.

3. 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.

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

5. 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

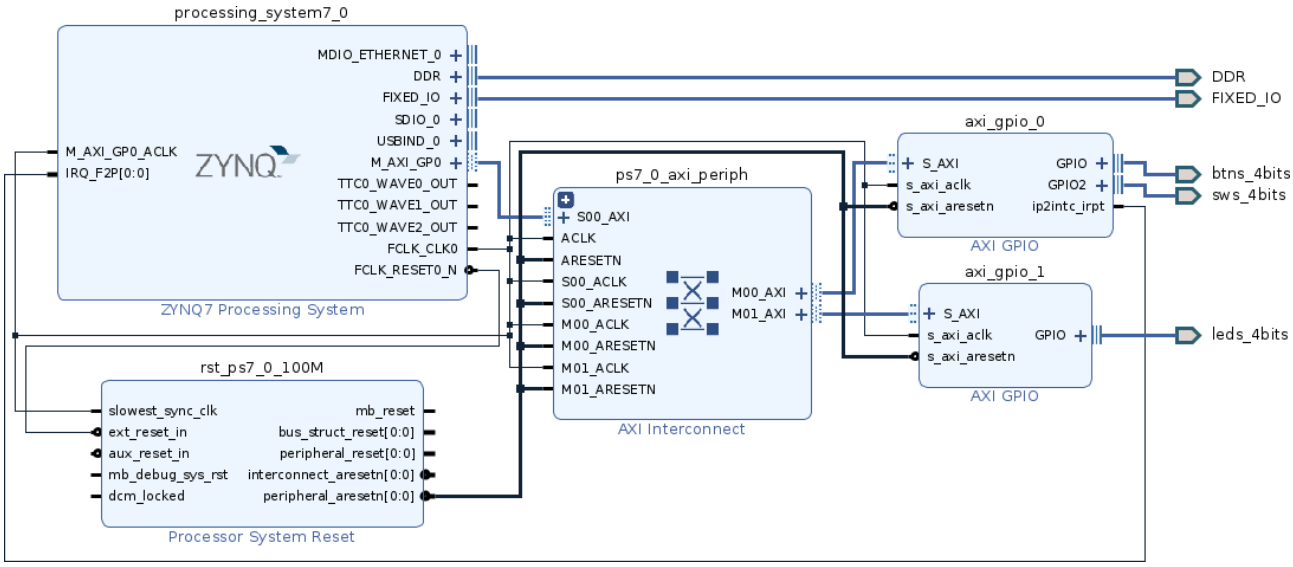
16. Щраква се два пъти върху блока “ZYNQ7 Processing System” → в “Page navigator” се отива на раздел “Interrupts” → слага се отметка на “Fabric interrupts” → в подраздела “PL-PS Interrupt Ports” се слага отметка на “IRQ_F2P” → OK

17. Щраква се два пъти върху блока “axi_gpio_0” (GPIO с бутони и ключове) → слага се отметка “Enable interrupt” → OK

#18. Щраква се два пъти върху блока “axi_gpio_1” (GPIO със светодиоди) → #слага се отметка “Enable interrupt” → OK

18. В блока “axi_gpio_0” → натиска се и се задържа ляв бутон върху сигнала “ip2intc_irpt”, след което се свързва с “IRQ_F2P” на блока “ZYNQ7 Processing System” и бутона се пуска.

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



20. 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

21. Вляво → Flow navigator → Generate bitstream → Yes → OK → изчаква се няколко минути (докато завърши синтеза) → View reports → 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.

22. File → Export → Export hardware → Next → Include bitstream → Next → Next → Finish

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

23. 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.

24. 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.

25. File → New → Platform project → Platform project name: 02_ints → Next → таб "Create new platform from hardware" → Browse → избира се пътя до проекта 02_ints, създаден от Vivado → design_1_wrapper.xsa → Open → Finish

26. Вляво → Project explorer → избира се 02_ints → right-click → Build Project

27. File → New → Application project → Next → "Select a platform from repository" → Избира се 02_ints → Next → Application project name: 02_ints_app → Next → Next → Empty application → Finish

28. Щраква се с десен бутон върху директорията src в проекта 02_ints_app_system/02_ints_app → New → Other → C/C++ → Source File → Next → Source file: дава се име main.c → Finish

29. В текстовия редактор на Vitis въведете следната програма [1]:

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

XScuGic INTC0;
XGpio GPIO1;
XGpio GPIO0;

void xgpio_int_handler(void *InstancePtr){
	u32 int_status;

	int_status = XGpio_InterruptGetStatus(&GPIO0);

	XGpio_InterruptClear(&GPIO0, int_status);
}


int main(void){
	XScuGic_Config *intc_config;

	intc_config = XScuGic_LookupConfig(XPAR_PS7_SCUGIC_0_DEVICE_ID);
	XScuGic_CfgInitialize(&INTC0, intc_config, intc_config->CpuBaseAddress);
	XScuGic_Connect(&INTC0, XPAR_FABRIC_AXI_GPIO_0_IP2INTC_IRPT_INTR,
(Xil_ExceptionHandler) xgpio_int_handler, &GPIO0);
	XScuGic_Enable(&INTC0, XPAR_FABRIC_AXI_GPIO_0_IP2INTC_IRPT_INTR);
	Xil_ExceptionRegisterHandler(XIL_EXCEPTION_ID_INT, (Xil_ExceptionHandler)
XScuGic_InterruptHandler, &INTC0);
	Xil_ExceptionEnable();

	XGpio_Initialize(&GPIO1, XPAR_AXI_GPIO_1_DEVICE_ID);
	XGpio_SetDataDirection(&GPIO1, 1, 0x0);
	XGpio_Initialize(&GPIO0, XPAR_AXI_GPIO_0_DEVICE_ID);
```

```
	XGpio_SetDataDirection(&GPIO0, 1, 0xF);
	XGpio_SetDataDirection(&GPIO0, 2, 0xF);

	XGpio_InterruptEnable(&GPIO0, 0xFF);
	XGpio_InterruptGlobalEnable(&GPIO0);

	while(1){
		XGpio_DiscreteWrite(&GPIO1, 1, 0x00);
		usleep(200000);
		XGpio_DiscreteWrite(&GPIO1, 1, 0x01);
		usleep(200000);
	}

	return 0;
}
```

30. Вляво, Project explorer -> избира се 02_ints_app -> right-click -> Build project

31. Щракнете два пъти в текстовия редактор вляво на реда:

```
int_status = Xgpio_InterruptGetStatus(&GPIO0);
```

за да поставите точка на прекъсване.

32. Вляво, Project explorer -> избира се 02_ints_app_system -> right-click -> Debug as -> Launch Hardware

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

34. Натиснете някой от бутоните BTN0 ÷ BTN3 на платката Zybo. Ако всичко е минало успешно, би трябвало да влезете в хендлера на прекъсванията xgpio_int_handler( ).

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

36. 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 чрез прекъсване и преобръща (toggle) логическото състояние на съответните светодиоди LD0 ÷ LD3.

Използвайте битовия оператор XOR.

Използвайте функцията за задаване на прекъсване по нарастващ фронт[2]:

```
XScuGic_SetPriorityTriggerType(&INTC0, XIL_EXCEPTION_ID_INT, 0xA0, 0x3);
```

37. Напишете програма, която чете ключове SW0 ÷ SW3 чрез прекъсване и преобръща (toggle) логическото състояние на съответните светодиоди LD0 ÷ LD3. Не забравяйте, че трябва да върнете ключът в изходна позиция след задействането на светодиода.

* * *

[1] Kris Gaj, “Software/Hardware Codesign”, Lecture 5, Spring 2016, online, Oct 17, 2021.
https://people-ece.vse.gmu.edu/coursewebpages/ECE/ECE699_SW_HW/S16/viewgraphs/ECE699_lecture_5.pdf

[2] Srinivas Neeli, “Xilinx AXI GPIO Standalone driver”, online, Oct 17, 2021.
https://github.com/Xilinx/embeddedsw/blob/master/XilinxProcessorIPLib/drivers/gpio/examples/xgpio_intr_tapp_example.c

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