Name Cunyang Liu ID u201811446 Due date: Friday, August 20, 2021

S&H Co-Design Homework #2 Handout Summer 2021

(Zynq-7010, xil_printf(), Zybo Board Interface Files)

Refer to the Zynq Book, Zybo Reference Manual (ZYBO_RM_B_V6.pdf), and Zynq-7000 Technical Reference Manual (ug585) and the other related Xilinx user guide (ugxxx) and data sheet (ds187) documents to answer the following questions.

1 Define the following terms as used in Zynq-7000 (Most definitions are copied from the Zynq Book.)

1.1 Processing System (PS)

The part of the Zynq device that includes an ARM processor and associated facil ities, and which is capable of running software applications.

1.2 Programmable Logic (PL)

The section of the Zynq device which comprises reconfigurable logic and interconnects. It has the same architecture as 7-series FPGA devices.

1.3 Multiplexed IO (MIO)

The set of peripheral outputs from the Zynq processing system, which are multiplexed to a dedicated IO bank.

1.4 Extended Multiplexed IO (EMIO)

The mechanism for extending the fixed Multiplexed Input / Output(MIO) facilities of the processing system, by extending into the programmable logic.

1.5 AXI 3.0

A high performance interconnect for use in high clock rate system designs. It is a member of the AMBA family.

1.6 SelectIO

The general purpose input/output facilities(IOBs) on the Zynq are collectively referred to as SelectIO Resources, and these are organised into banks of 50 IOBs each. Each IOB contains one pad, which provides the physical connection to the outside world for a single input or output signal.

1.7 GPIO

A collective term for the basic user input and output facil ities on a development board, usually including slide switches, push buttons, and LEDs.

2 The Zybo's Zynq-7010 chip bears the following marking: "XC7Z010-1CLG400C". It uses the "Ball Grid Array" (BGA) pinout for maximum packaging density. Answer the following questions about the chip. Include sources and evidences to support your answers such as document names and page numbers etc.

2.1 What is the maximum number of physical pin connections available with this particular BGA layout? (Hint: see ug865)

*400pins (20*20)*
ug865 page78

2.2 What is the vertical and horizontal spacing (pitch) between adjacent ball grid pins?

0.8mm
ug865 page10

2.3 What are the overall physical dimensions of the Zybo's Zynq chip?

*17mm*17mm*1.47mm*
ug865 page9

2.4 How many processing system (PS) I/O pins are available?

128
ug865 page11

2.5 How many programmable logic (PL) ("Select I/O") single-ended I/O pins are available?

100
ug865 page11

2.6 How many programmable logic (PL) ("Select I/O") differential I/O pairs pin pairs are available?

48
ug865 page11

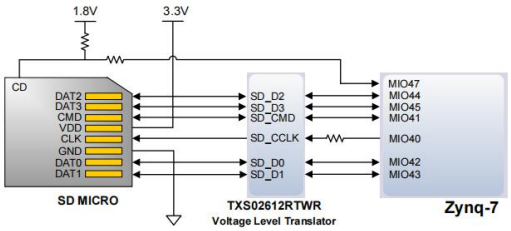
3 *Find out if I/O pins on the Zybo board are mapped to either MIO or Select I/O pins from ZYBO FPGA Board Reference Manual 2016 available from https://reference.digilentinc.com/_media/zybo:zybo_rm.pdf.*

3.1 *Circle the components on Zybo below that are connected to MIO pins.*

- LED4
- HDMI
- SPI
- LEDs LD3-0
- UART 1
- USB 0
- SDIO 0
- LED4
- BTN4
- Push Buttons BTN3-0
- BTN5
- Pmod JA

3.2 *Find where MicroSD interface pins are connected, Select I/O or MIO, and their pin numbers? You can include screen shots as your answer.*

Signal Name	Description	Zynq Pin	SD Slot Pin
SD_D0	Data[0]	MIO42	7
SD_D1	Data[1]	MIO43	8
SD_D2	Data[2]	MIO44	1
SD_D3	Data[3]	MIO45	2
SD_CCLK	Clock	MIO40	5
SD_CMD	Command	MIO41	3
SD_CD	Card Detect	MIO47	9



3.3 *Find where HDMI interface pins are connected, Select I/O or MIO, and their pin numbers? Include screen shots to support your answer.*

HDMI is PL pin
J8 Select I/O
pin numbers:

HDMI_HPD:	*E18*
HDMI_SDA:	*G18*
HDMI_SCL:	*G17*
HDMI_CLK_N:	*H17*
HDMI_CLK_P:	*H16*
HDMI_D0_N:	*D20*
HDMI_D0_P:	*D19*
HDMI_D1_N:	*B20*
HDMI_D1_P:	*C20*
HDMI_D2_N:	*A20*
HDMI_D2_P:	*B19*

4 *Describe major differences between printf() and xil_printf() and why and when xil_printf() is better than printf(). (Hint: read the header of xil_printf().c source code.) Include screen shots to support your conclusion.*

```
/*----------------------------------------------------*/
/*                                                    */
/* This routine operates just like a printf/sprintf   */
/* routine. It outputs a set of data under the        */
/* control of a formatting string. Not all of the     */
/* standard C format control are supported. The ones  */
/* provided are primarily those needed for embedded   */
/* systems work. Primarily the floating point         */
/* routines are omitted. Other formats could be       */
/* added easily by following the examples shown for   */
/* the supported formats.                             */
/*                                                    */
```

Printf() cause many programs from libraries and the ELFs because containing printf() will be larger than containing Xil_printf().

5 Read the source code for xil_printf(), describe what it does and how it is implemented to accomplish its function. Include screen shot or reference pages to support your conclusion.

5.1 Where is the base address of the STDOUT and STDIN. Describe the call structure and how the driver knows the base address of UART 1.

STDIN_BASEADDRESS : 0xE0001000
STDOUT_BASEADDRESS : 0xE0001000

5.2 Describe the call structure and how

*Call structure is void xil_printf(const char8 *ctrl1, ...)*
How: include the "xparameters.h" to get it since the base address of UART1 is defined in it.

5.3 5.3 How does xil_printf() know where or which com port, to send its formatted character string?

It uses XUartPs_SendByte() to choose com port, which depends on the base address of STDOUT and UART1

5.4 How are data received and transmitted?

Write the data to TX FIFO,receive it from RX FIFO

5.5 Are the sending and transmitting subroutines interrupt-driven or blocking calls?

Blocking calls.

5.6 Indicate how Level 1 driver of xil_printf() relates to Level 0 driver of uart device. Hint: xil_printf() uses outbyte() from uartps_3_0 library, which sends data to UART 1. Include screen captures to support your answer.

xil_printf() call outbyte(), outbyte() call XUartPs_SendByte(),XUartPs_SendByte() call XUL_Out32(): level 1 driver.

6 Zybo Board Interface Files. Use XML Notepad from Microsoft to describe how Pmod Connector JC is defined in Board Support File A.0 for zybo-z7-10. Include screen captures to support your description. (zybo-z7-10 board files can be downloaded from https://github.com/Digilent/vivado-boards/archive/master.zip. XML Notepad is available from http://www.lovettsoftware.com/downloads/xmlnotepad/readme.htm.)

6.1 How is the component defined?

The component is named jc.

```
<component name="jc" display_name="Connector JC" type="chip" sub_type="chip" major_group="Pmod">
<description>Pmod Connector JC</description>
```

6.2 How is connection defined?

The connection is defined as part0_jc. The two components are part0 and jc.

```
<connection name="part0_jc" component1="part0" component2="jc">
  <connection_map name="part0_jc_1" c1_st_index="55" c1_end_index="62" c2_st_index="0" c2_end_index="7"/>
```

6.3 Where in the board file and how is the interface defined?

All interfaces are defined under the first component of part0, the chip definition.

```
</compatible_board_revisions>
<file_version>1.0</file_version>
<description>Zybo Z7-10</description>
<components>
<component name="part0" display_name="Zybo Z7-10" type="fpga" part_name="xc7z010clg400-1" pin_map_file="part0_pins.xml" vendor="xilinx" s
<interfaces>
<interface mode="master" name="btns_4bits" type="xilinx.com:interface:gpio_rtl:1.0" of_component="btns_4bits" preset_proc="push_butto
</interface>
<interface mode="master" name="leds_4bits" type="xilinx.com:interface:gpio_rtl:1.0" of_component="leds_4bits" preset_proc="led_4bits_
</interface>
<interface mode="master" name="ps7_fixedio" type="xilinx.com:display_processing_system7:fixedio_rtl:1.0" of_component="ps7_fixedio" p
</interface>
<interface mode="master" name="sws_4bits" type="xilinx.com:interface:gpio_rtl:1.0" of_component="sws_4bits" preset_proc="dip_switches
<port_maps>
<port_map logical_port="TRI_I" physical_port="sws_4bits_tri_i" dir="in" left="3" right="0"> ···
</port_map>
</port_maps>
</interface>
<interface mode="slave" name="sys_clock" type="xilinx.com:signal:clock_rtl:1.0" of_component="sys_clock" preset_proc="sys_clock_prese
</interface>···
<preferred_ip vendor="digilentinc.com" library="ip" name="dvi2rgb" order="0"/>
</preferred_ips>
<port_maps>···
<preferred_ips>···
</preferred_ips>
<port_maps>···
<preferred_ips>
<preferred_ip vendor="digilentinc.com" library="ip" name="rgb2dvi" order="0"/>
</preferred_ips>
<port_maps>
<port_map logical_port="CLK_P" physical_port="TMDS_OUT_clk_p" dir="out">
<pin_maps>
<pin_map port_index="0" component_pin="TMDS_OUT_clk_p"/>
</pin_maps>
</port_map>
<port_map logical_port="CLK_N" physical_port="TMDS_OUT_clk_n" dir="out">
<pin_maps>
<pin_map port_index="0" component_pin="TMDS_OUT_clk_n"/>
</pin_maps>
</port_map>
<port_map logical_port="DATA_P" physical_port="TMDS_OUT_D_P" dir="out" left="2" right="0">
<pin_maps>
```

6.4 *How are the pins described and mapped between logical and physical pins?*

The pins are described and mapped in part0_pins.xml file between logical and physical pins. However, pins 55 to 62 are not included in zybo-z7-10 A.0 part0_pins.xml file. They are given in zybo-z7-20 A.0 part0_pins.xml file as follows. JC1 to JC10 are connected to pins 55 to 62.each one is connected to a physical pin such as JC1 on V15, JC2 on W15, etc.

```
<pin index="55" name ="JC1" iostandard="LVCMOS33" loc="V15"/>
<pin index="56" name ="JC2" iostandard="LVCMOS33" loc="W15"/>
<pin index="57" name ="JC3" iostandard="LVCMOS33" loc="T11"/>
<pin index="58" name ="JC4" iostandard="LVCMOS33" loc="T10"/>
<pin index="59" name ="JC7" iostandard="LVCMOS33" loc="W14"/>
<pin index="60" name ="JC8" iostandard="LVCMOS33" loc="Y14"/>
<pin index="61" name ="JC9" iostandard="LVCMOS33" loc="T12"/>
<pin index="62" name ="JC10" iostandard="LVCMOS33" loc="U12"/>
```