Sequencer

User Guide

January 29, 2020

Contents

1 Introduction **7**

2 Sequencer operation **7**

3 Implementation **7**

3.1 Block Diagram 10

3.2 Peripherals 10

3.2.1 General Purpose Register File 10

3.2.2 Debug Printer 11

3.2.3 Sequencer Loop Controller 11

3.2.4 LED Driver 11

3.2.5 Switch Driver 12

3.2.6 Push-Button Driver 12

4 Interface Signals **12**

5 Memory Map **14**

6 Implementation Results **15**

7 Conclusions **15**

List of Tables

1 Sequencer Loop Controller Inputs 8
2 Sequencer Loop Controller Outputs 8
3 Sequencer Loop Controller Inputs 11
4 Sequencer Loop Controller Outputs 11
5 Led driver inputs 12
6 Led driver outputs 12
7 Switch driver inputs 14
8 Switch driver outputs 14
9 Push-button driver inputs 14
10 Push-button driver outputs 14
11 Interface Signals 14
12 Memmory map 14

List of Figures

1 Basys 2 board peripherals . . . 7
2 Basys 2 board peripherals . . . 8
3 PicoVersat SoC with two peripherals . . . 8
4 Sequencer Loop controler fluxogram. . . . 9
5 Picoversat code fluxogram. . . . 9
6 Block Diagram . . . 10
7 PicoVersat SoC with two peripherals . . . 10
8 PicoVersat SoC with two peripherals . . . 11
9 PicoVersat SoC with two peripherals . . . 11
10 PicoVersat SoC with two peripherals . . . 12
11 PicoVersat SoC with two peripherals. . . . 12
12 Peripherals interconnections. . . . 13

1 Introduction

A sequencer is a device that can produce rythmic loops programmed by the user. The loop is devided into 8 equally spaced steps and each step can be activated in order to produce a sound. The sequencer will go trough the loop and will play a sound on the activated steps. The loop period and the sound frequency can be set by the user.

The sequencer hardware is implemented in Verilog and uses the Picoversat SoC as the basic processing unit. Refer to the Picoversat manual for more information. This sequencer implementation is meant to be used on the Basys 2 FPGA by Digilent, for that, custom-made peripherals are developed in order to use the board's features (e.g.LEDs, Switches, etc...).

2 Sequencer operation

In normal operation, the sequencer will loop through all the steps in a cyclic fashion, the current step can be seen by the indicated LED. In order to to create a rythm, the user can activate the desired steps by activating the correspondant switch. The activated steps are indicated by their correspondant LEDs being on.

The sound output of the signal is a DC biased square wave outputted by an IO pin of the Basys 2 Board, this should be connected to and external powered speaker in order to the sound to be apmplified and filtered.

Figure 1 shows the Basys 2 peripherals that the user can use in order to interact with the sequencer.

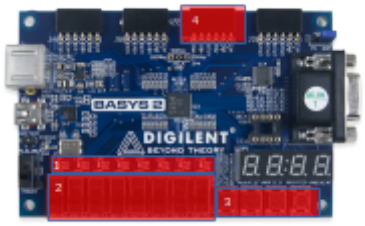

Figure 1: Basys 2 board peripherals

3 Implementation

Since the sequencer depends on multiple time-dependant routines and there is no trivial way to deal with this on the picoversat controller (because of the lack of interrupts), the main logic is divided into two routines: one for the main sequencer loop, implemented as a standalone peripheral, the "Sequencer loop controller", and other for the reading and debouncing of the pushbuttons and for keeping track of the frequency and loop values.

All the peripherals and modules are interconnected as described in the following picture:

The sequencer loop controller is used to generate the sequencer loop. The loop and note frequency can be set by using the *freq* input and by selecting the according selector signal. The Sequencer loop controller will output a square wave corresponding to the loop ouput. The led outputs are directly connected to the LED driver peripheral and send information about the current note.

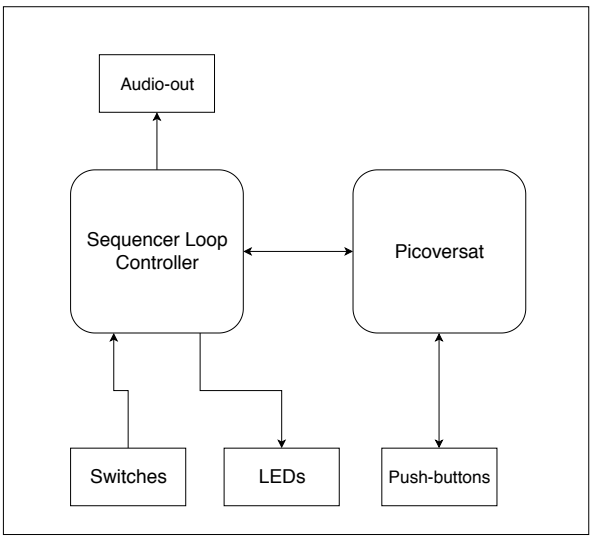

Figure 2: Basys 2 board peripherals

The sequencer loop controller operation is described by the fluxogram on Picture 5.

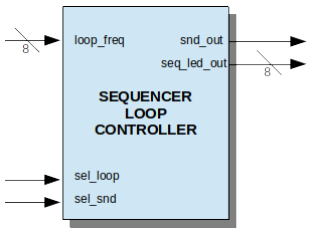

Figure 3: PicoVersat SoC with two peripherals

Table 1: Sequencer Loop Controller Inputs

Name	#bits	Description
freq	8	Loop Period / Note Frequency
sel_loop	1	Loop period select signal (address decoder)
sel_snd	1	Note frequency select signal (address decoder)

Table 2: Sequencer Loop Controller Outputs

Name	#bits	Description
snd_out	1	Audio Output
seq_led_out	8	Current note led output

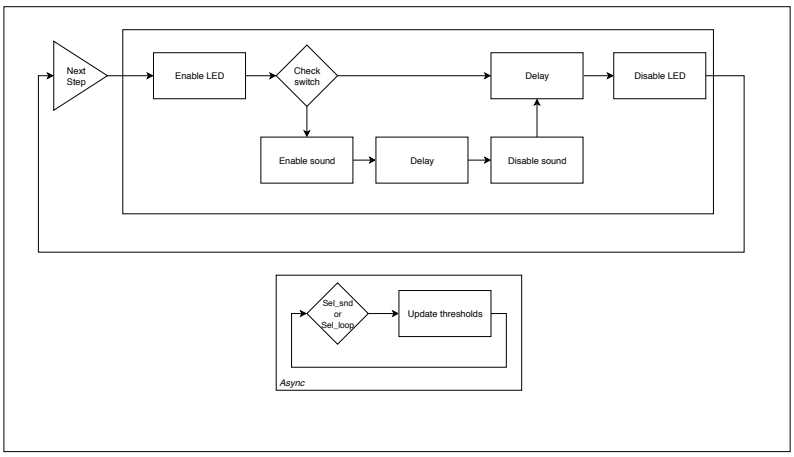

Figure 4: Sequencer Loop controler fluxogram.

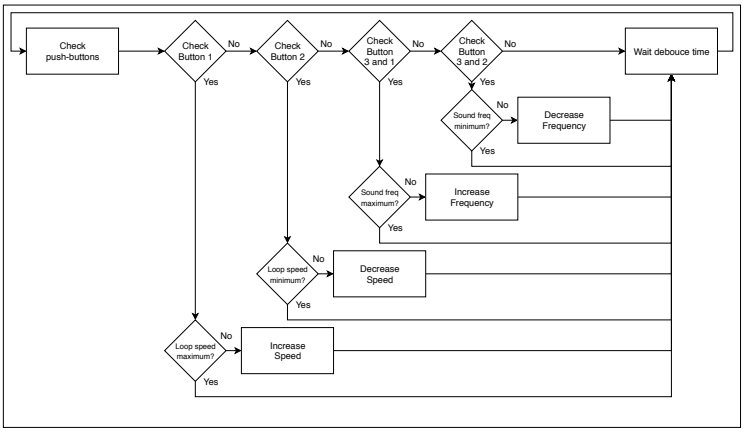

Figure 5: Picoversat code fluxogram.

3.1 Block Diagram

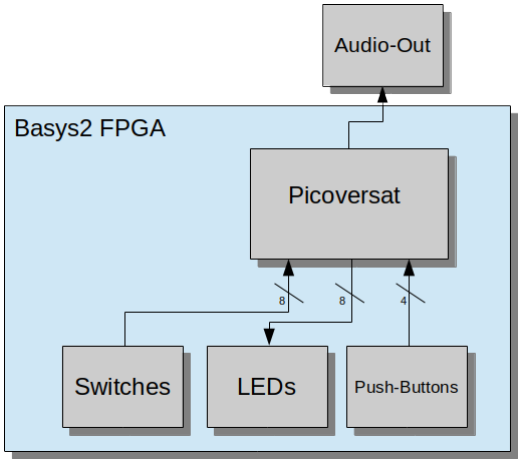

Figure 6: Block Diagram

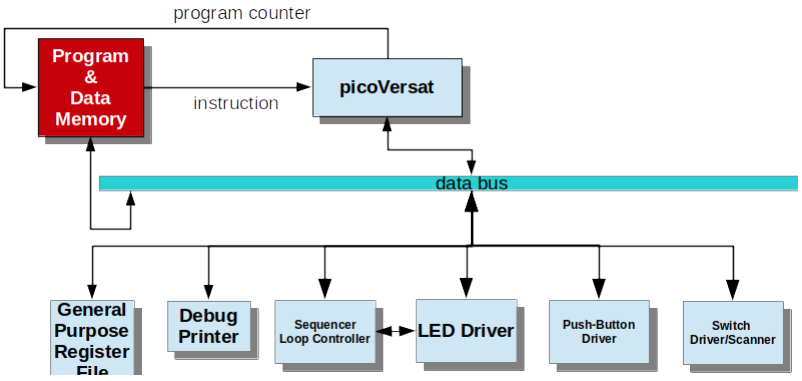

Figure 7: PicoVersat SoC with two peripherals

3.2 Peripherals

In order to controll the Basys 2 board peripherals with the picoversat controller, peripheral drivers are used. The peripherals are mapped in memmory, refer to the memory map in subsection 5 to check the peripheral's addresses.

3.2.1 General Purpose Register File

This peripheral contains a 16x32bit register file that can be used by user programs. Refer to the picoversat manual for more information about this peripheral.

3.2.2 Debug Printer

This peripheral can be used by user programs to print characters, mainly for debug purposes. Refer to the picoversat manual for more information about this peripheral.

3.2.3 Sequencer Loop Controller

This peripheral is used to generate the sequencer loop. The loop and note frequency can be set by using the *freq* input and by selecting the according selector signal. The Sequencer loop controller will output a square wave corresponding to the loop ouput. The led outputs are directly connected to the LED driver peripheral and send information about the current note.

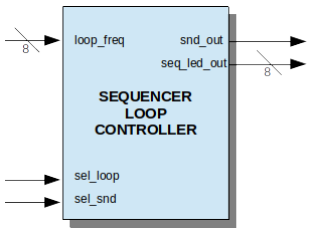

Figure 8: PicoVersat SoC with two peripherals

Table 3: Sequencer Loop Controller Inputs

Name	#bits	Description
freq	8	Loop Period / Note Frequency
sel_loop	1	Loop period select signal (address decoder)
sel_snd	1	Note frequency select signal (address decoder)

Table 4: Sequencer Loop Controller Outputs

Name	#bits	Description
snd_out	1	Audio Output
seq_led_out	8	Current note led output

3.2.4 LED Driver

The LED driver will display the current note. The input of the driver is directly controlled by the sequencer loop controller.

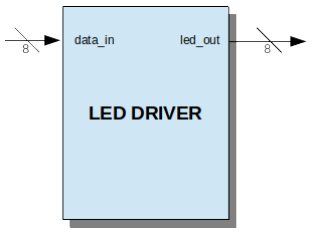

Figure 9: PicoVersat SoC with two peripherals

Table 5: Led driver inputs

Name	#bits	Description
data_in	8	Led display information

Table 6: Led driver outputs

Name	#bits	Description
led_out	8	Led display output

3.2.5 Switch Driver

This driver reads the state of the slide-switches of the Basys 2 board.

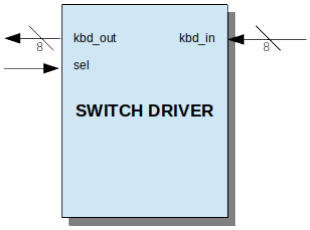

Figure 10: PicoVersat SoC with two peripherals

3.2.6 Push-Button Driver

This driver reads the state of the push-buttons of the Basys 2 board.

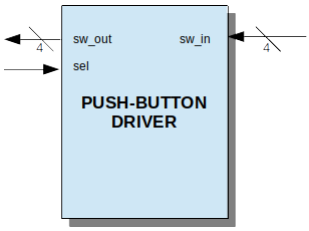

Figure 11: PicoVersat SoC with two peripherals.

The peripheral are interconnected according to the following picture:

4 Interface Signals

The interface signals of the sequencer controller peripherals Table 11..

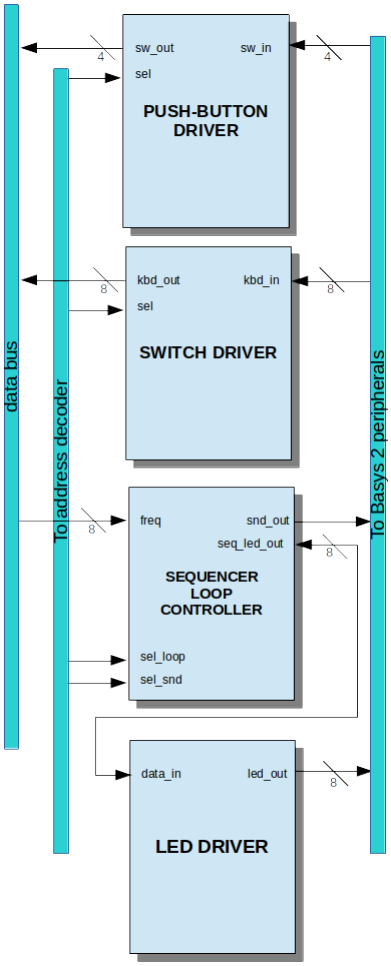

Figure 12: Peripherals interconnections.

Table 7: Switch driver inputs

Name	#bits	Description
sw_in	8	Switch state inputs
sel	1	driver selector (address decoder)

Table 8: Switch driver outputs

Name	#bits	Description
kbd_out	8	Switch state outputs

Table 9: Push-button driver inputs

Name	#bits	Description
sw_in	4	Push-button state inputs
sel	1	Driver selector (address decoder)

Table 10: Push-button driver outputs

Name	#bits	Description
sw_out	4	Push-button state outputs

Table 11: Interface Signals

Name	#bits	Direction	Peripheral	Description
clk	1	IN	FPGA clock	Clock Signal
led_out	8	OUT	LED driver	Reset Signal
sw_in	4	IN	Push-button driver	Push Buttons
kbd_in	8	IN	Switch driver	Slide Switches
snd_out	1	OUT	Sequencer loop controller	Sound output

5 Memory Map

The sequencer peripherals are mapped in memmory according to the following table:

Table 12: Memmory map

Mnemonic	Address	Read/Write	Read/Write Latency	Description
SND_BASE	0x25D	Write only	0	Sound frequency address for sequencer loop controller
LOOP_BASE	0x25C	Write only	0	Loop frequency address for sequencer loop controller
SW_BASE	0x25B	Read only	0	Slide switch peripheral
PUSH_BASE	0x25A	Read only	0	Push-button peripheral
CPRT_BASE	0x258	Read only	N/A	Debug printer peripheral
REGF_BASE	0x200	Read+Write	0	Register file peripheral
PROG_BASE	0x0	Read+Write	1	User program and data

6 Implementation Results

7 Conclusions

