Lecture 16

PicoBlaze I/O & Interrupt Interface
Required reading

• P. Chu, *FPGA Prototyping by VHDL Examples*

  Chapter 16, *PicoBlaze I/O Interface*

  Chapter 17, *PicoBlaze Interrupt Interface*
## Syntax and Terminology

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Example</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>sX</td>
<td>s7</td>
<td>Value at register 7</td>
</tr>
<tr>
<td>KK</td>
<td>ab</td>
<td>Value ab (in hex)</td>
</tr>
<tr>
<td>PORT(KK)</td>
<td>PORT(2)</td>
<td>Input value from port 2</td>
</tr>
<tr>
<td>PORT((sX))</td>
<td>PORT((s1))</td>
<td>Input value from the port specified by register s1</td>
</tr>
<tr>
<td>RAM(KK)</td>
<td>RAM(4)</td>
<td>Value from the RAM location 4</td>
</tr>
</tbody>
</table>
Addressing modes

Immediate mode

SUB s7, 07

ADDCY s2, 08

Direct mode

ADD sa, sf

INPUT s5, 2a

Indirect mode

STORE s3, (sa)

INPUT s9, (s2)
Output Decoding of Four Output Registers

in_port  | out_port  | en_d(0)  | out_data0
reset  | port_id  | en_d(1)  | out_data1
instruction  | read_strobe  | en_d(2)  | out_data2
interrupt  | write_strobe  | en_d(3)  | out_data3
            | interrupt_ack  |         |
            | address        |         |
Output Instructions

OUTPUT sX, KK

PORT(KK) <= sX

OUTPUT sX, (sY)

PORT((sY)) <= sX

DIR

IND

C Z

_ _
Timing Diagram of an Output Instruction

- clk
- instruction
- port_id
- out_port
- write_strobe
- en_d(0)
- en_d(1)
- en_d(2)
- en_d(3)
- out_data2

from PicoBlaze

decoded signals

content of s0

content of s0 sampled and stored
Truth Table of a Decoding Circuit

<table>
<thead>
<tr>
<th>write_strobe</th>
<th>port_id(1)</th>
<th>port_id(0)</th>
<th>output</th>
<th>en_d</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>–</td>
<td>–</td>
<td>0000</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0001</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0010</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0100</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1000</td>
<td></td>
</tr>
</tbody>
</table>
Input Instructions

INPUT sX, KK
sX <= PORT(KK)

INPUT sX, (sY)
sX <= PORT((sY))
Block Diagram of Four Continuous-Access Ports
Timing Diagram of an Input Instruction

```
clk
instruction
input s0, 02
port_id
02
in_port
read_strobe
register s0
sampled data
data is sampled
```
Block Diagram of Four Single-Access Ports
FIFO Interface

clk  rst
clk  rst
FIFO

din  dout
full  empty
write  read

8
8
Operation of the “Standard” FIFO

clk
write
read
din
A B C D
A B C D
dout
empty
write write read write read write read
FIFO operation
write write read read
FIFO data
A B A B C D
Operation of the First-Word Fall-Through FIFO

clk
write
read
din
A  B  C  D

dout
A  B  C  D
empty
FIFO operation
write  write  read  write  read  read
FIFO data
A  B  A  B  C  D
This diagram represents one possible arrangement in which KCPSM6 can be used to service 4 input ports and 4 general purpose output ports. Each port is up to 8-bits and could be connected directly to pins on the device package or may connect to some of your other logic within the device. All KCPSM6 designs build on variations of this fundamental arrangement. Suitable PSM code for this circuit is shown on page 51.
Timing Diagram of an Input Instruction

 clk

 instruction

 port_id

 in_port

 read_strobe

 register s0

 sampled data

data is sampled
Interrupt Flow

; ======= main loop =======
forever:
    ... 
    enable interrupt 
    ... 
    add s0, s3 
    sub s5, 01 
    ... 
    call critical_timing 
    ... 
    jump forever 

;===time critical segment ===
critical_timing:
    disable interrupt 
    ... 
    enable interrupt 
    return 

;===interrupt service routine===
 isr:
    test s2, 01 
    ... 
    returni enable 

;===interrupt vector ===
address 3FF 
jump isr
Timing Diagram of an Interrupt Event

The instruction is preempted and "call 3FF" is implicitly executed.
Interrupt recognized

5 clock cycles

Begin executing interrupt service routine

CLK

INPUT s1, 01

ADD s0, s1

JUMP isr

TEST s7, 02

PREEMPTED

Address of

ADD s0, s1

3FF

ISR

Interrupt enabled.

CALL/RETURN Stack

Preserved

ZERO Flag

Preserved

CARRY Flag

Preserved

ZERO Flag

Preserved

CARRY Flag

Interrupt disabled.

Interrupt service routine

Interrupt service routine

Interrupt service routine

Interrupt service routine

Interrupt service routine

Interrupt service routine
Interrupt Related Instructions

RETURNI ENABLE

PC <= STACK[TOS] ; TOS <= TOS – 1;
I <= 1;   C<= PRESERVED C;   Z<= PRESERVED Z

RETURNI DISABLE

PC <= STACK[TOS] ; TOS <= TOS – 1;
I <= 0;   C<= PRESERVED C;   Z<= PRESERVED Z

ENABLE INTERRUPT

I <=1;

DISABLE INTERRUPT

I <=0;
Interrupt Interface with a Single Event
Interrupt Interface with Two Requests
PicoBlaze Input/Output Class Exercise
Task

Draw a detailed block diagram of the digital system including:

1. PicoBlaze-6, KCPSM6
2. 4k x 18 instruction ROM required for the basic operation of PicoBlaze
3. 64 x 8 external data RAM visible under addresses 0x00-0x3F
4. 64 x 8 external data ROM visible under addresses 0x40-0x7F
5. two input registers with the virtual addresses 0x80 and 0xCF
6. two output registers with the virtual addresses 0x80 and 0xC0
7. D flip-flop with the output Q connected to the interrupt input of the PicoBlaze core, input SET connected to the external port INT, and input CLR connected to an appropriate output of the PicoBlaze core
Assumptions

- input register with the address 0x80 is the same as the output register with the address 0x80
- the input and output registers, data RAM, and data ROM specified above are the only i/o devices that the PicoBlaze core is communicating with
- your system needs to be able to allow the PicoBlaze core to write to all aforementioned output registers and data RAM, and read from all the aforementioned input registers, data RAM, and data ROM, using instructions OUTPUT and INPUT, respectively
- you need to provide all details of the address decoder, and build it out of basic logic components you are familiar with
- all registers and flip-flops have a reset input connected to the external port RESET.
Please clearly mark on your schematic:
- sizes of all memories and registers
- sizes and directions of all buses.
Interface of PicoBlaze-6

```plaintext
kcpsm6

[17:0]
- instruction
- bram_enable
- address

[7:0]
- in_port
- out_port
- write_strobe
- read_strobe
- k_write_strobe
- port_id

interrupt
- interrupt_ack

sleep
reset

clk

your_program

[11:0]
- enable
- instruction
- address

[7:0]
- clk

C_FAMILY => "S6"
C_RAM_SIZE_KWORDS => 1
C_JTAG_LOADER_ENABLE => 1

Connections to input and output ports
Connection of control signals

hwbuild => X"00"
interrupt_vector => X"3FF"
scratch_pad_memory_size => 64
```
rdl – reset during load

The program memory has the option to include the JTAG Loader utility which facilitates rapid development of your KCPSM6 program.

‘rdl’ is a ‘reset during load’ signal associated with the loader, which needs to be connected to the reset input of the processor.
Development
Flow of a System
with PicoBlaze
PicoBlaze Input/Output
Class Exercise
Solutions