Lecture 17

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((sa))</td>
<td>Input value from port specified by register a</td>
</tr>
<tr>
<td>RAM(KK)</td>
<td>RAM(4)</td>
<td>Value from RAM location 4</td>
</tr>
</tbody>
</table>
# Addressing modes

## Immediate mode

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDCY s2, 08</td>
<td>s2 + 08 + C → s2</td>
</tr>
<tr>
<td>SUB s7, 7</td>
<td>s7 - 7 → s7</td>
</tr>
</tbody>
</table>

## Direct mode

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT s5, 2a</td>
<td>PORT(2a) → s5</td>
</tr>
<tr>
<td>ADD sa, sf</td>
<td>sa + sf → sa</td>
</tr>
</tbody>
</table>

## Indirect mode

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT s9, (s2)</td>
<td>PORT((s2)) → s9</td>
</tr>
<tr>
<td>STORE s3, (sa)</td>
<td>s3 → RAM((sa))</td>
</tr>
</tbody>
</table>
Output Decoding of Four Output Registers
Output Instructions

OUTPUT

OUTPUT sX, KK
PORT(KK) <= sX

OUTPUT sX, (sY)
PORT((sY)) <= sX
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

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>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>–</td>
<td>–</td>
<td>0000</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0001</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0010</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0100</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1000</td>
</tr>
</tbody>
</table>
Input Instructions

INPUT

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

port_id

in_port

read_strobe

register s0

input s0, 02

02

sampled data

data is sampled
Block Diagram of Four Single-Access Ports
FIFO Interface
Operation of the First-Word Fall-Through FIFO

The diagram illustrates the operation of the First-Word Fall-Through FIFO under different clock cycles. The diagram shows the following:

- **clk**: The clock signal indicating the time for each operation.
- **write**: The write operation signals for placing data into the FIFO.
- **read**: The read operation signals for retrieving data from the FIFO.
- **din**: The data input to the FIFO.
- **dout**: The data output from the FIFO.
- **empty**: The status of the FIFO indicating whether it is empty or not.
- **FIFO operation**: The sequence of write and read operations.
- **FIFO data**: The data flow through the FIFO, demonstrating the fall-through mechanism.

At each cycle, data is written into the FIFO, and after an empty status, data is read out, continuing the cycle.
Operation of the “Standard” FIFO
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
    ...
    return enable

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

```
clk

instruction
add s0, s3
sub s5, 01
jump isr
test s2, 01

address
addr of add s0, s3
addr of sub s5, 01
3FF
isr

interrupt

interrupt_ack

The instruction is preempted and "call 3FF" is implicitly executed
```
Interrupt Related Instructions

RETURNI ENABLE

\[
\begin{align*}
PC & \leq \text{STACK}[\text{TOS}] ; \quad \text{TOS} \leq \text{TOS} – 1; \\
I & \leq 1; \quad C \leq \text{PRESERVED C}; \quad Z \leq \text{PRESERVED Z}
\end{align*}
\]

RETURNI DISABLE

\[
\begin{align*}
PC & \leq \text{STACK}[\text{TOS}] ; \quad \text{TOS} \leq \text{TOS} – 1; \\
I & \leq 0; \quad C \leq \text{PRESERVED C}; \quad Z \leq \text{PRESERVED Z}
\end{align*}
\]

ENABLE INTERRUPT

\[
I \leq 1;
\]

DISABLE INTERRUPT

\[
I \leq 0;
\]
Interrupt Interface with a Single Event

![Diagram showing interrupt interface with a single event]
Interrupt Interface with Two Requests

```
int request 0
set flag
clr
flag FF

int request 1
set flag
clr
flag FF

in_port
reset
instruction
interrupt

out_port
port_id
read_strobe
write_strobe
interrupt_ack
address

KCPSM3

output decoding
```
Time-Multiplexed Seven Segment Display
Block Diagram of the Hexadecimal Time-Multiplexing Circuit
Hexadecimal Multiplexing Circuit Based on PicoBlaze and mod-500 Counter