CDA 4253 FPGA System Design

The PicoBlaze Microcontroller

Hao Zheng
Comp Sci & Eng
U of South Florida

Overview of PicoBlaze

- Soft-core microcontroller in VHDL: portable to other platforms.
- Small: occupies ~20 CLBs.
- Respectable performance: 50 MIPS
- Predictable performance: every instruction takes 2 cycles.
- Suitable for simple data processing and control.

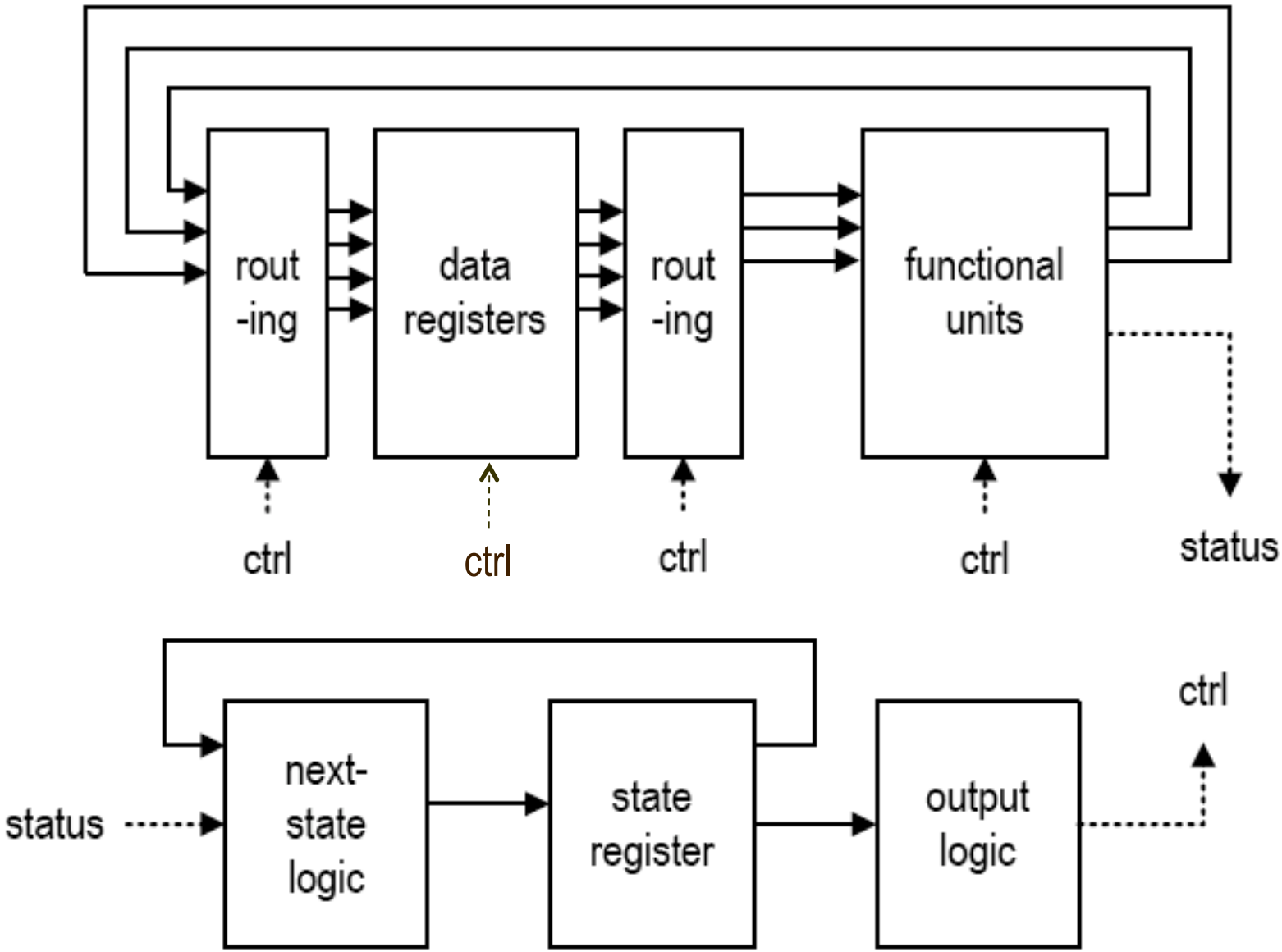
Required Reading

- P. Chu, FPGA Prototyping by VHDL Examples
  Chapter 14, PicoBlaze Overview

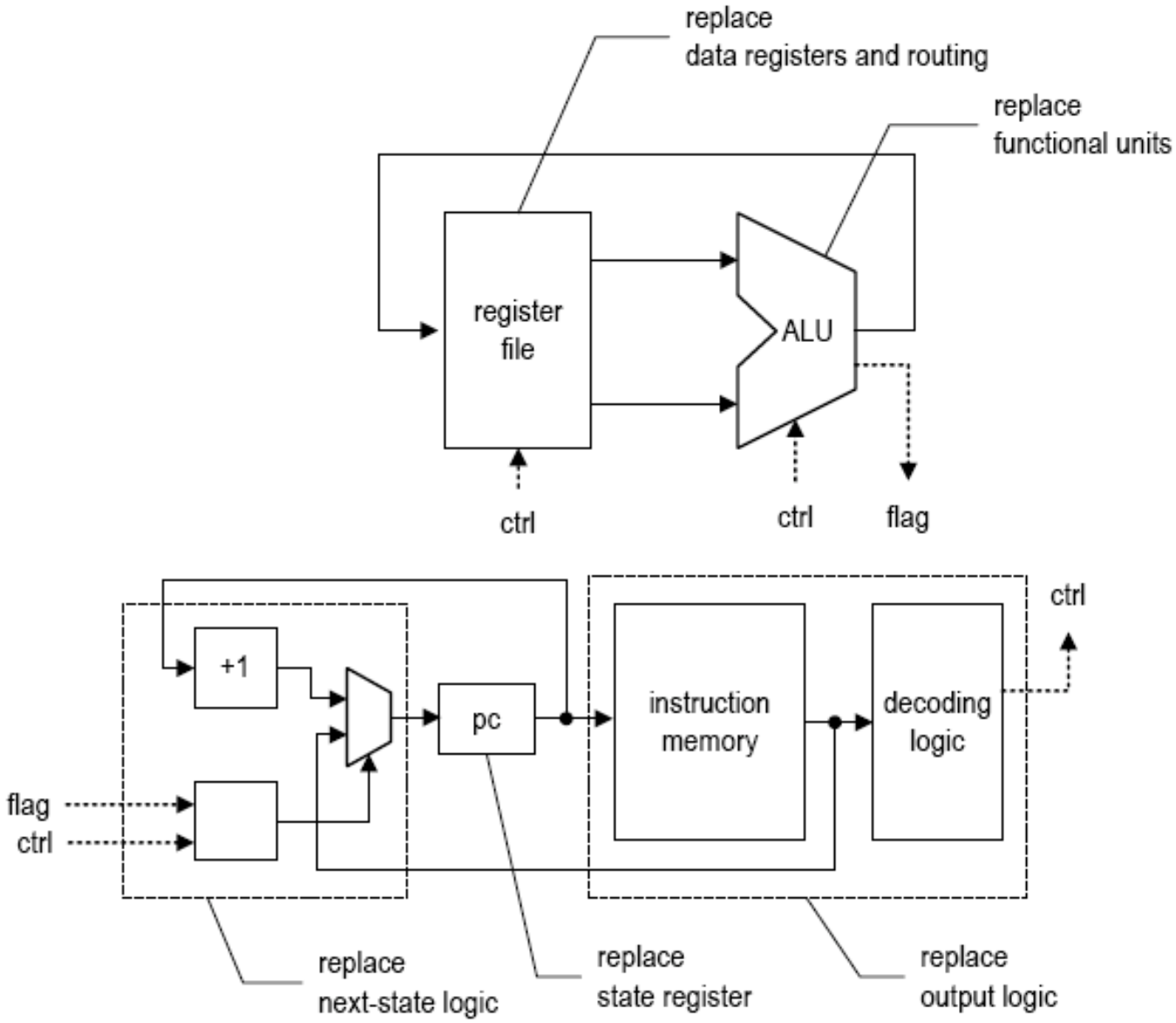
Recommended Reading

- *PicoBlaxe 8-bit Embedded Microcontroller User Guide (UG129)*
- *K. Chapman, PicoBlaze for Spartan-6, Virtex-6, and 7-Series (KCPSM6)*

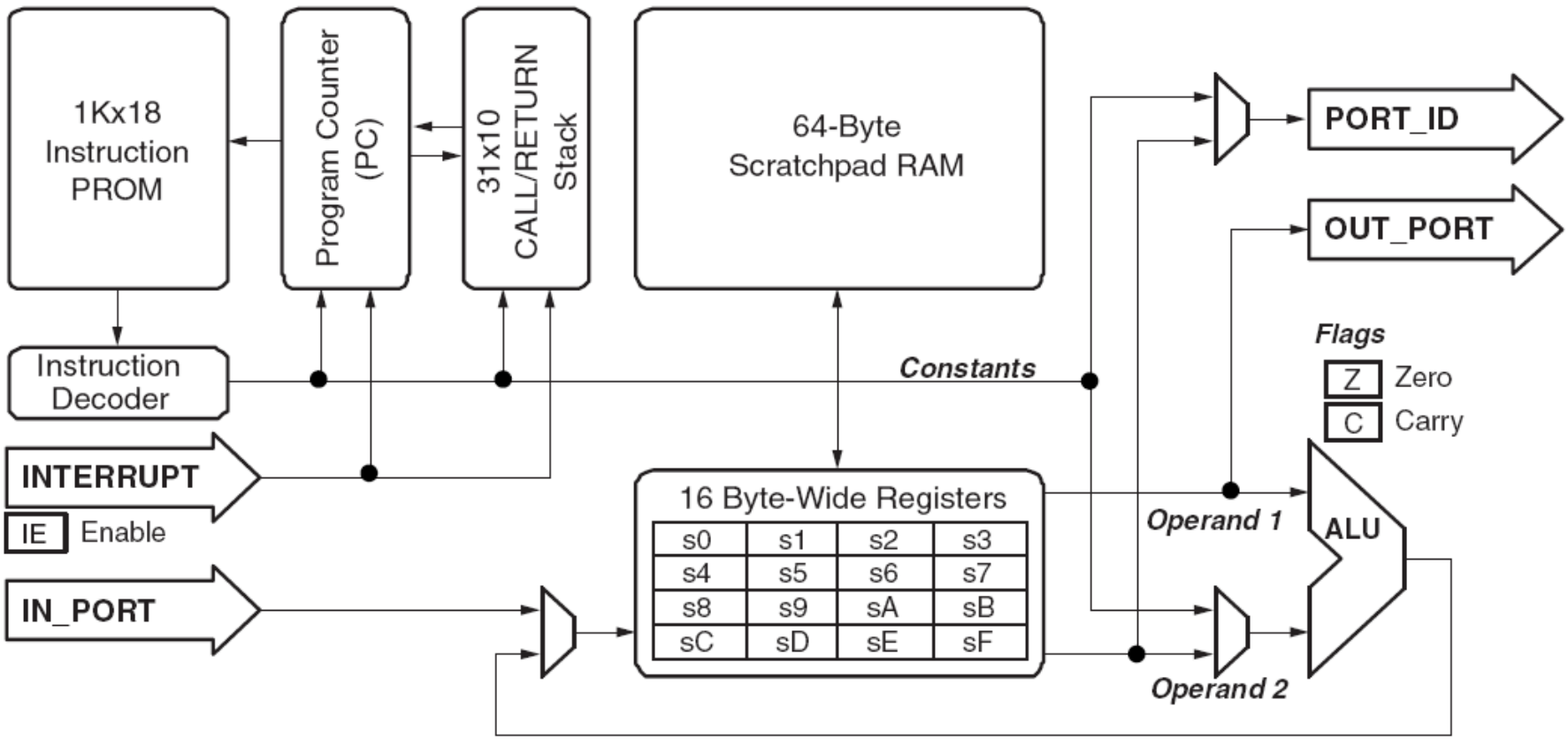
Block diagram of a General-Purpose Processor



Block diagram of a General-Purpose Processor (Microcontroller)



PicoBlaze Overview



8-bit data width, 18-bit instruction width, 10-bit program address

Size of PicoBlaze-6 in Spartan 6

1. Resource Utilization in CLB Slices
   - 26 CLB Slices
   - 1.1% of Spartan-6 used in Nexys3
2. Number of PicoBlaze-6 cores fitting inside of the Spartan-6

   FPGA (XC6SLX16) used in the Nexys3 FPGA board

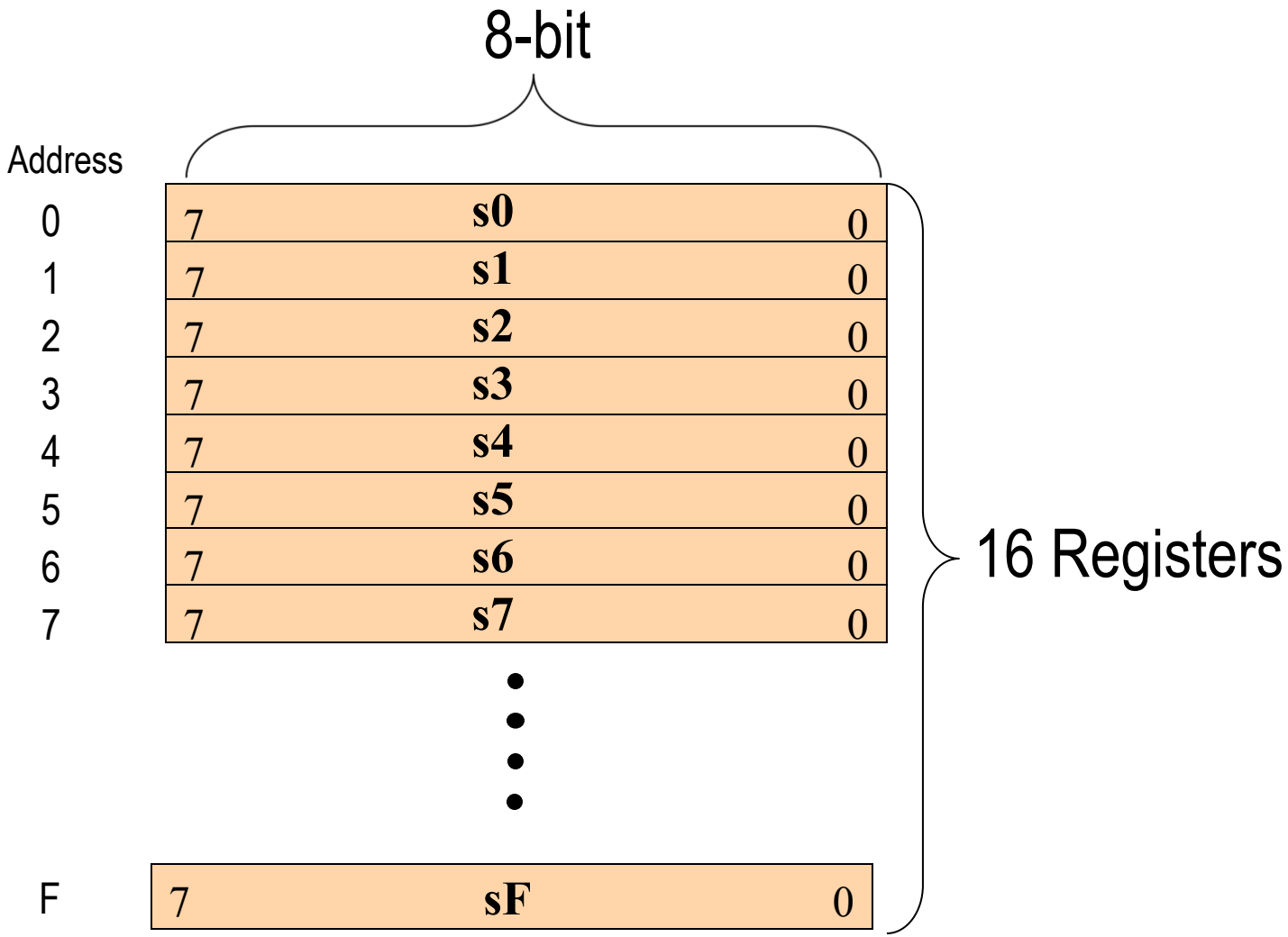
   - 87 PicoBlaze cores

Speed of PicoBlaze on Basys-3

1. Maximum Clock Frequency
   - 100 MHz
2. Maximum number of instructions per second
   - 50 millions of instructions per second (MIPS)

Fixed timing: ideal for real-time control applications, i.e. flight control, manufacturing process control, ...

Register File of PicoBlaze-3



Definition of Flags

Flags are set or reset after ALU operations

Zero flag - Z *zero condition*

Z = 1 if result = 0
0 otherwise

Carry flag - C *overflow, underflow, or various conditions*

<u>Example</u>*

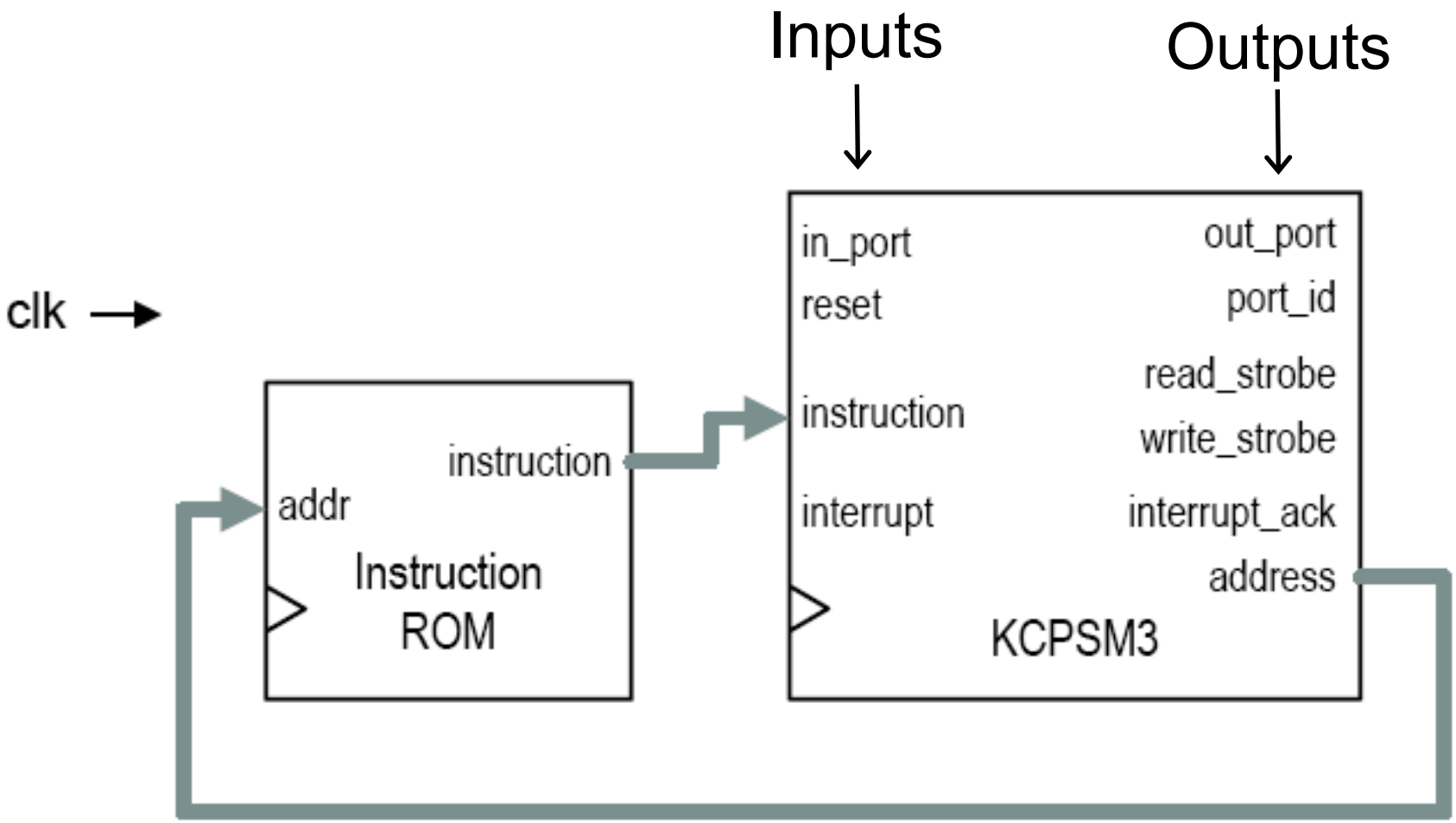
C = 1 if result > 2^8**-1 (for addition) or**

result < 0 (for subtraction)

0 otherwise

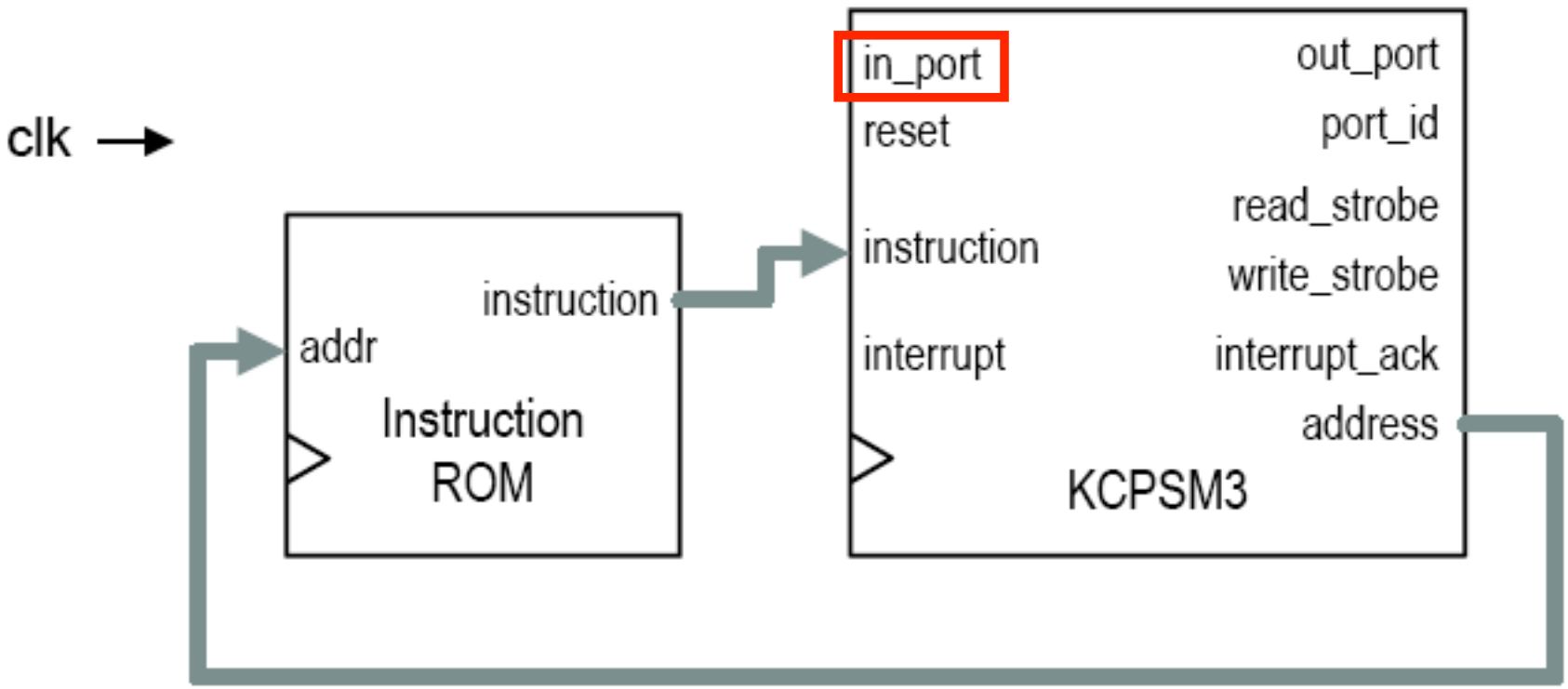
***Applies only to addition or subtraction related instructions, refer to the following slides otherwise**

Interface of PicoBlaze



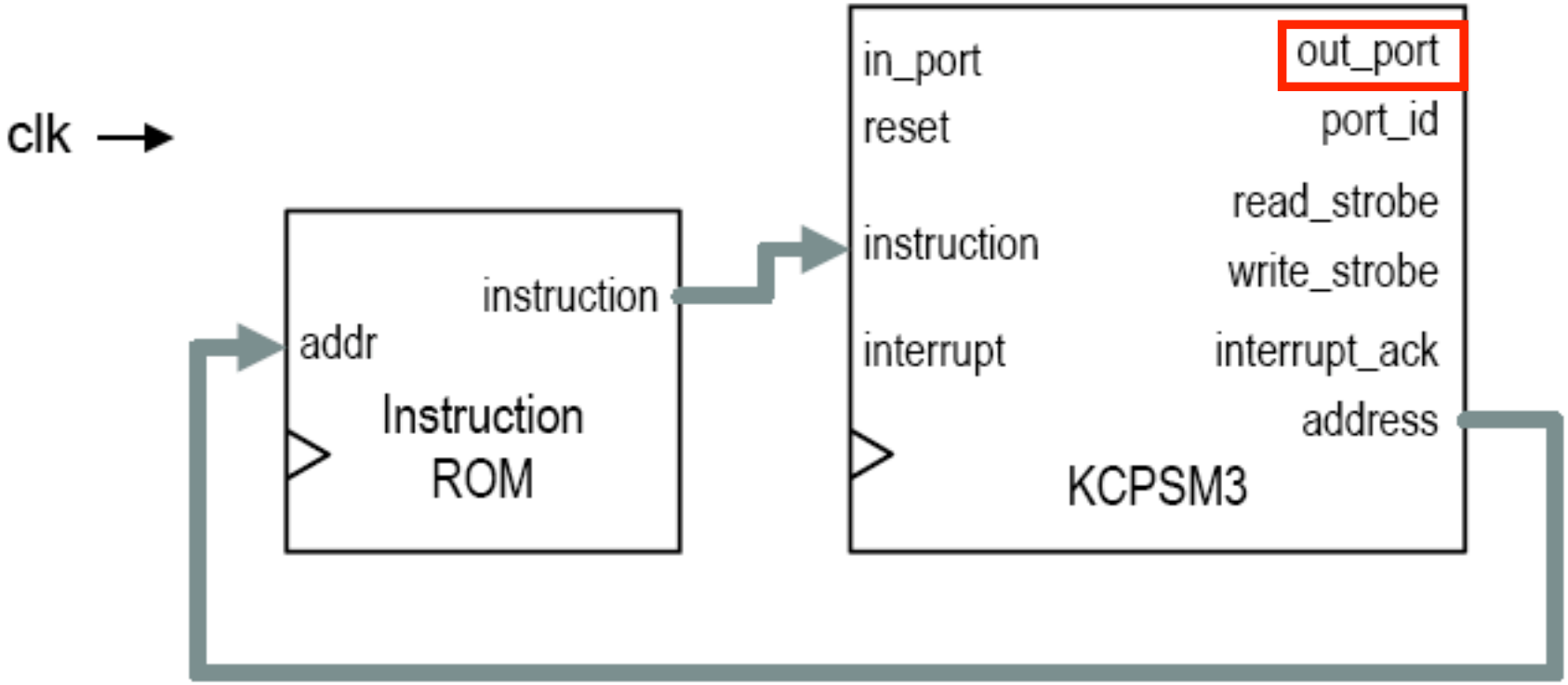
KCPSM = constant (K) coded programmable state machine

Interface of PicoBlaze



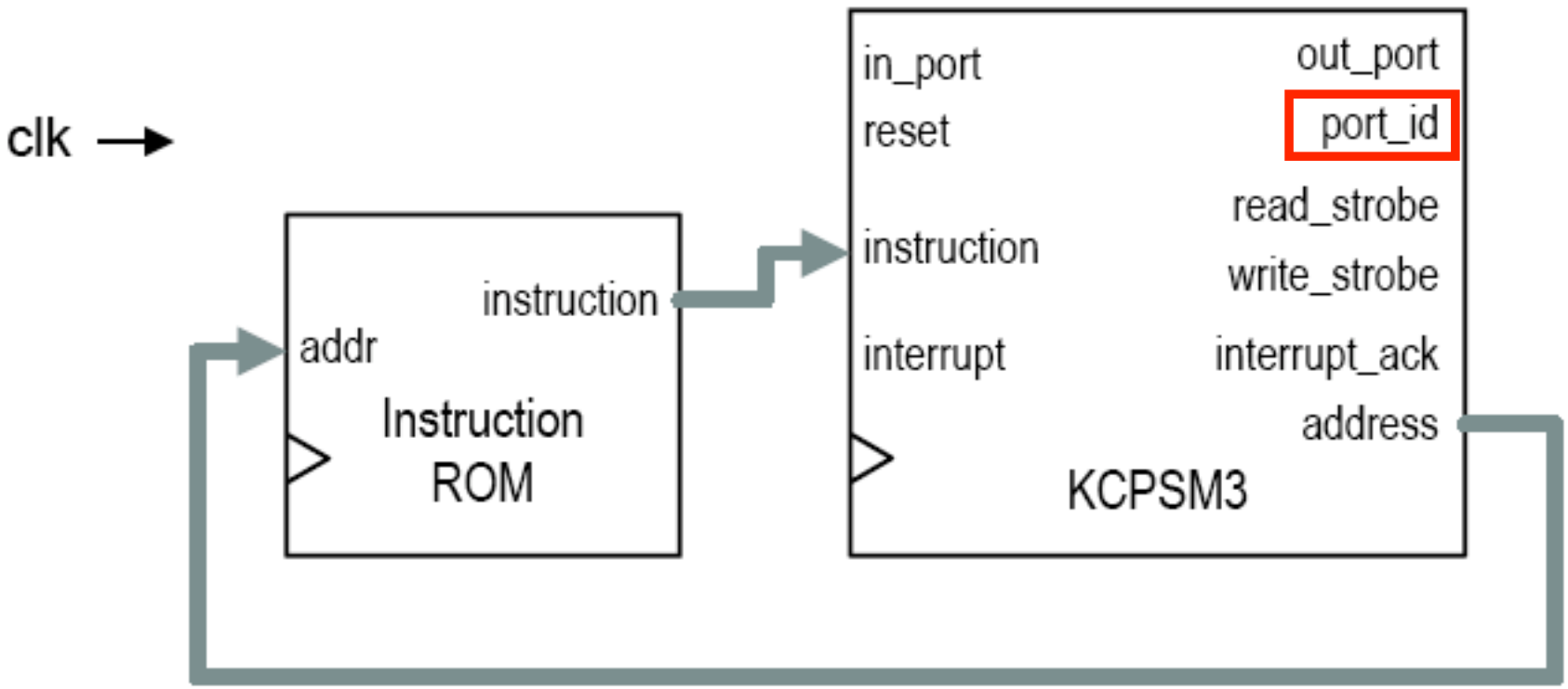
in_port[7:0] – input data port that carries the data for the INPUT instruction.

Interface of PicoBlaze



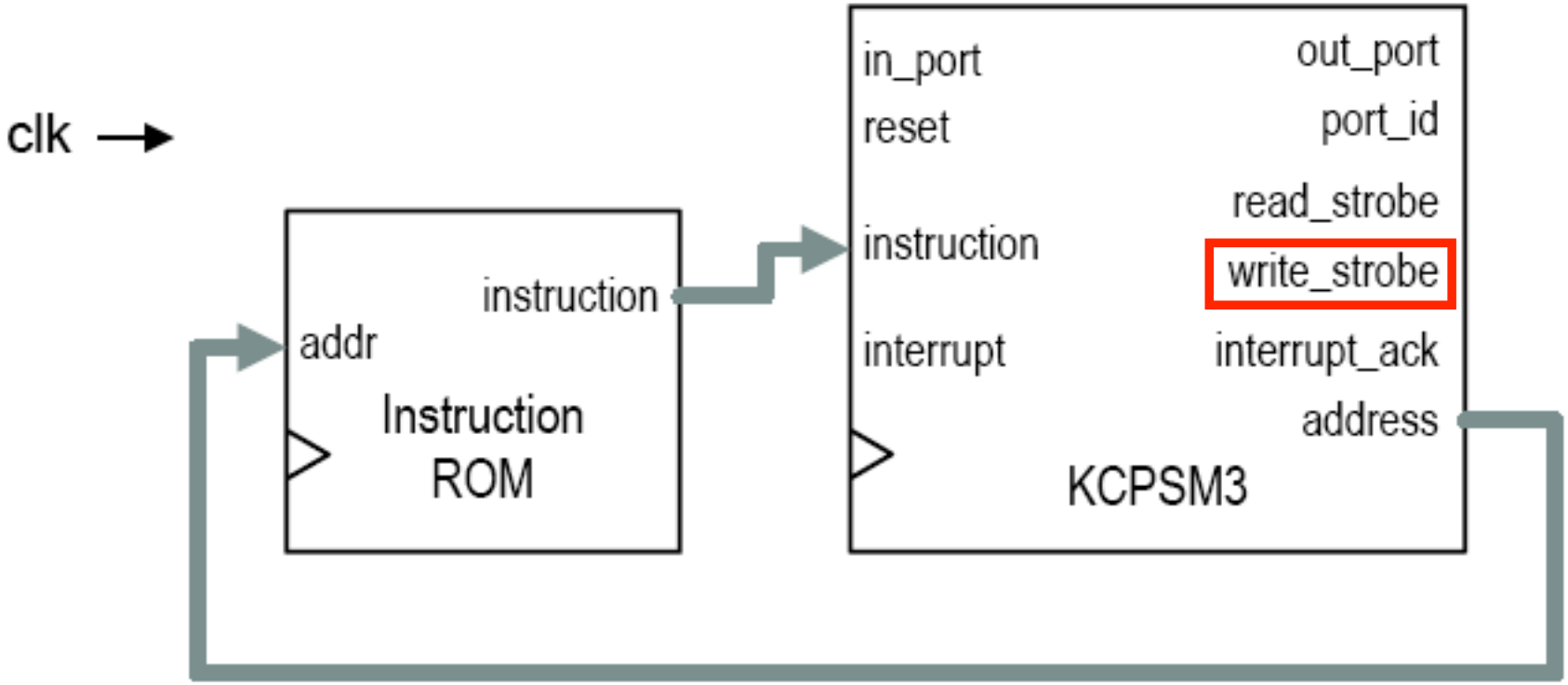
out_port[7:0] – carries the output data for an OUTPUT instruction.

Interface of PicoBlaze



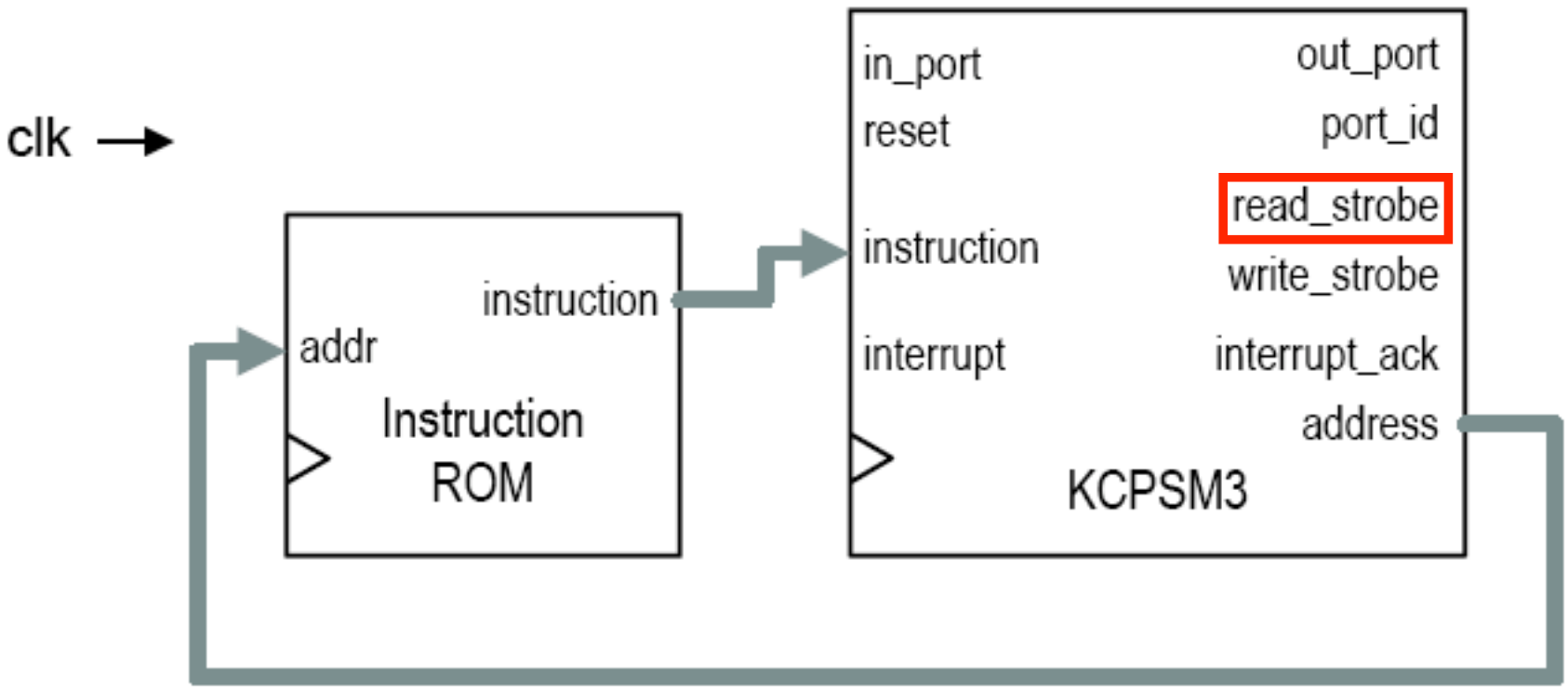
port_id[7:0] – addresses of components connected to PicoBlaze. Holds for two cycles during an INPUT/OUTPUT instruction.

Interface of PicoBlaze



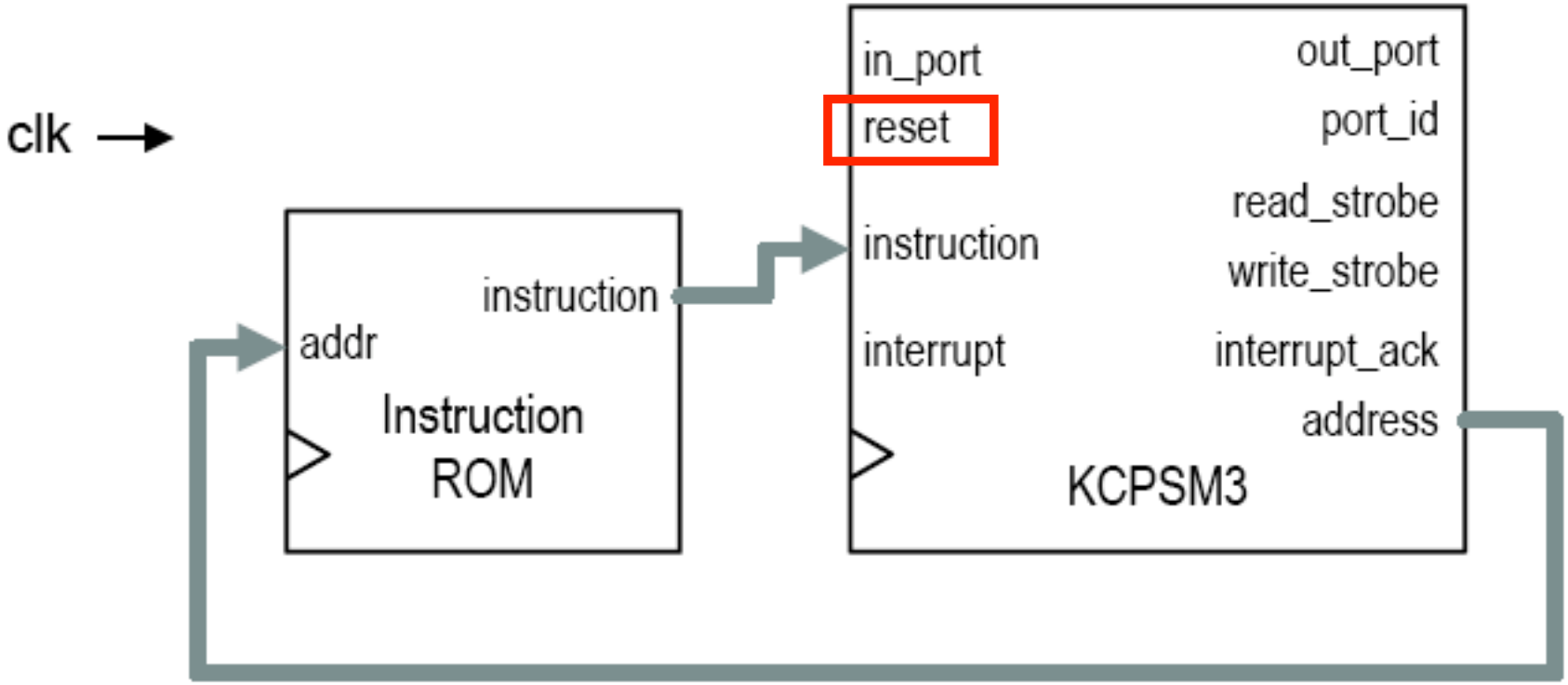
write_strobe – being asserted ‘1’ validates the data on the **output_port[7:0]**.

Interface of PicoBlaze



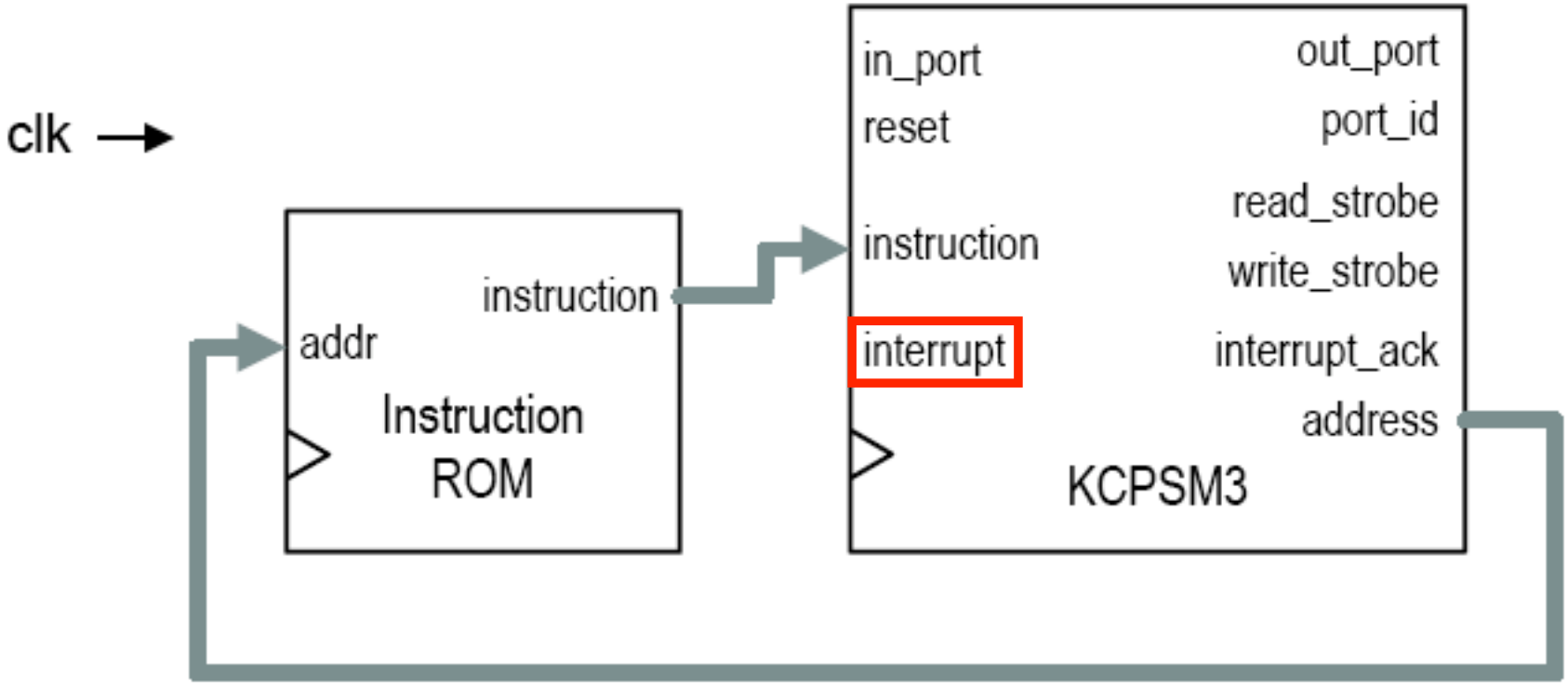
read_strobe – being asserted '1' indicates the capture of the data on the **input_port[7:0]** during an INPUT instruction.

Interface of PicoBlaze



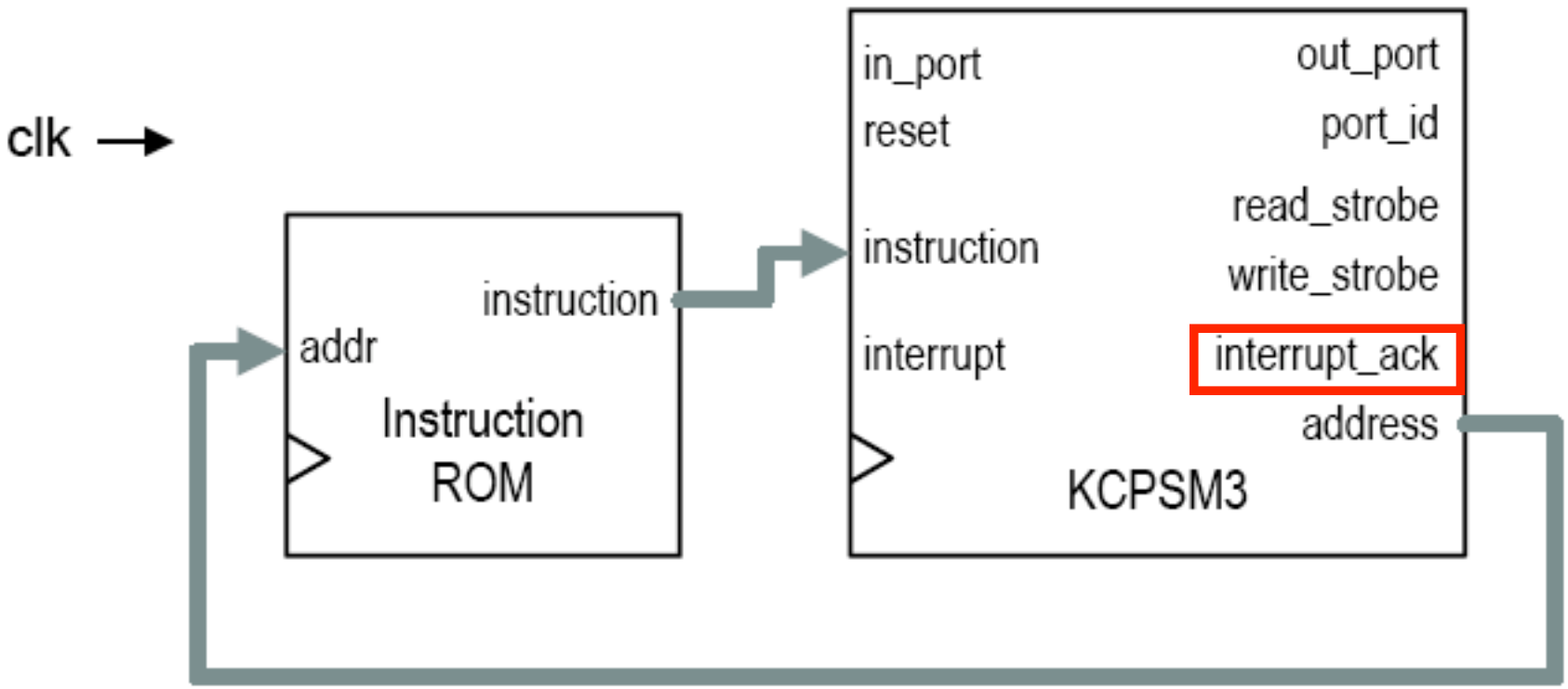
reset – needs to be asserted for at least one cycle.

Interface of PicoBlaze



interrupt – assert it for at least two cycles to trigger an interrupt event in PicoBlaze.

Interface of PicoBlaze



interrupt_ack – acknowledges the current interrupt has been recognized by PicoBlaze. Used to clear the current interrupt.

Interface of PicoBlaze – Summary

Name	Direction	Size	Function
clk	input	1	System clock signal.
reset	input	1	Reset signal.
address	output	10	Address of the instruction memory. Specifies address of the instruction to be retrieved.
instruction	input	18	Fetched instruction.
port_id	output	8	Address of the input or output port.
in_port	input	8	Input data from I/O peripherals.
read_strobe	output	1	Strobe associated with the input operation.
out_port	output	8	Output data to I/O peripherals.
write_strobe	output	1	Strobe associated with the output operation.
interrupt	input	1	Interrupt request from I/O peripherals.
interrupt_ack	output	1	Interrupt acknowledgment to I/O peripherals

Use of PicoBlaze in VHDL Design

```
component KCPSM3
port (
      address        : out std_logic_vector( 9 downto 0);
      instruction    : in  std_logic_vector(17 downto 0);
      port_id        : out std_logic_vector( 7 downto 0);
      write_strobe   : out std_logic;
      out_port       : out std_logic_vector( 7 downto 0);
      read_strobe    : out std_logic;
      in_port        : in  std_logic_vector( 7 downto 0);
      interrupt      : in  std_logic;
      interrupt_ack  : out std_logic;
      reset          : in  std_logic;
      clk            : in  std_logic
     );
end component;
```

PicoBlaze Component Declaration

Use of PicoBlaze in VHDL Design

```
processor: kcpsm3
    port map(
                    address => address_signal,
                instruction => instruction_signal,
                    port_id => port_id_signal,
               write_strobe => write_strobe_signal,
                   out_port => out_port_signal,
                read_strobe => read_strobe_signal,
                    in_port => in_port_signal,
                  interrupt => interrupt_signal,
              interrupt_ack => interrupt_ack_signal,
                      reset => reset_signal,
                        clk => clk_signal
            );
```

PicoBlaze Component Instantiation

Use of PicoBlaze in VHDL Design

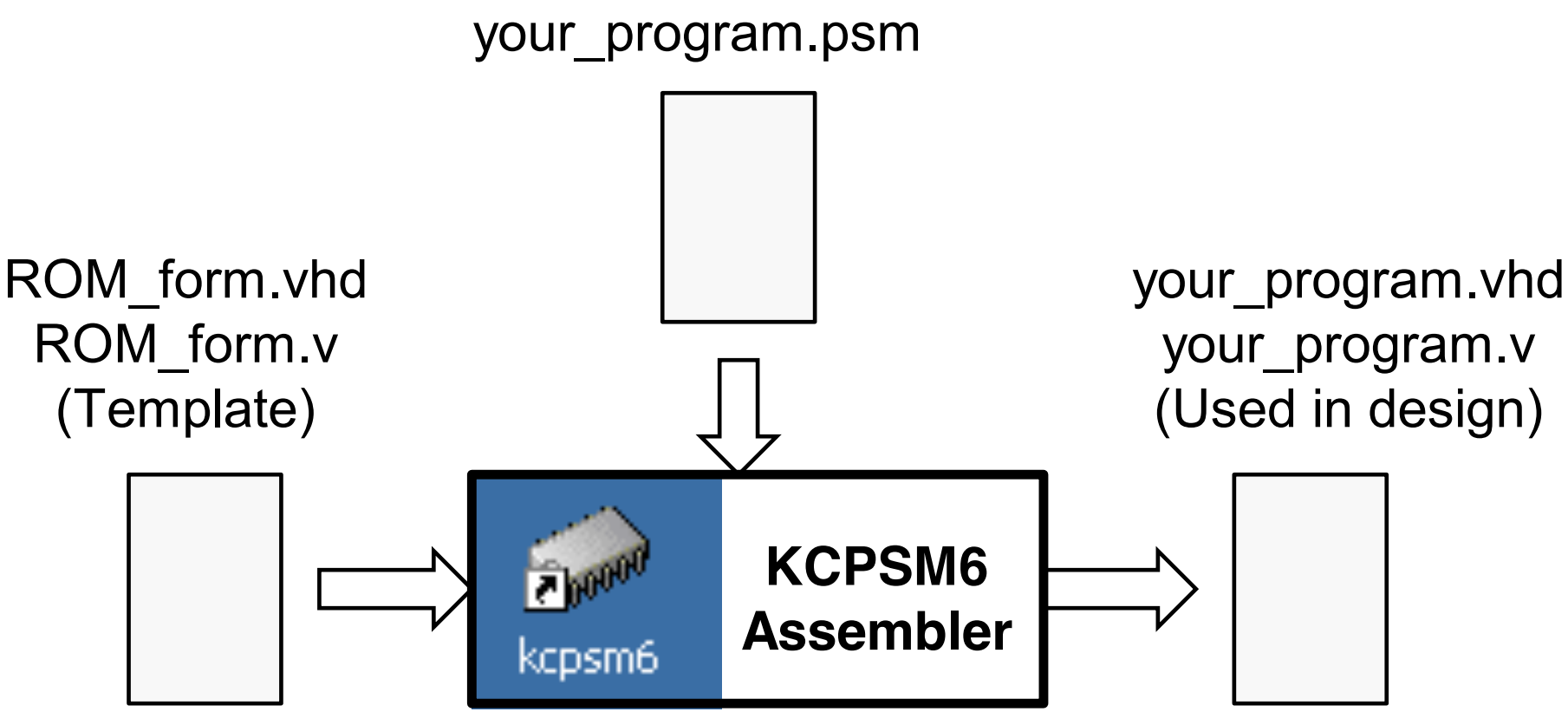
```
component prog_rom
port (
      address     : in  std_logic_vector( 9 downto 0);
      instruction : out std_logic_vector(17 downto 0);
      clk         : in  std_logic
     );
end component;
```

```
program: prog_rom
port map(    address => address_signal,
         instruction => instruction_signal,
                 clk => clk_signal
        );
```

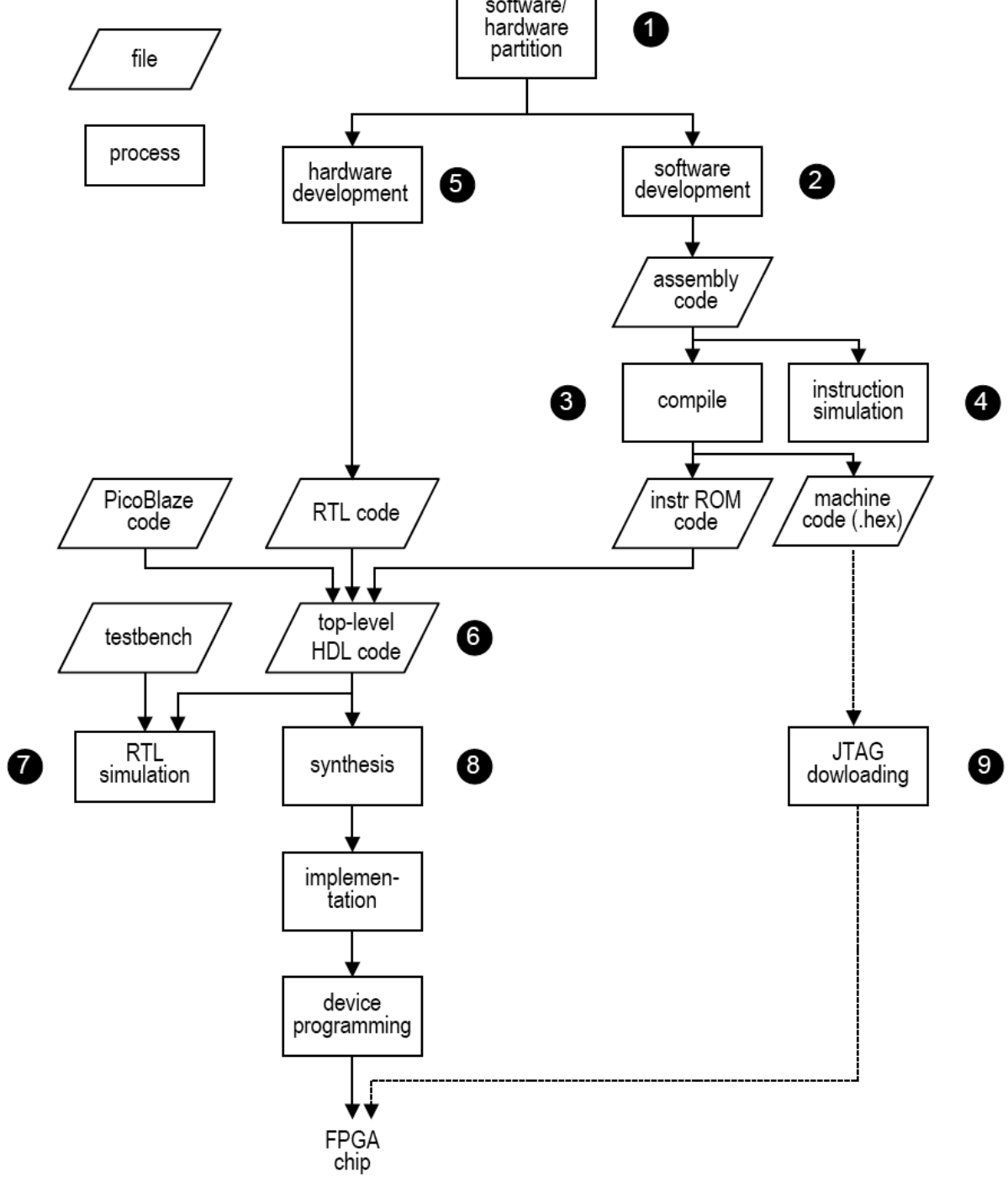
PicoBlaze Program ROM Component Declaration/ Instantiation

KCPSM3 and prog_rom are generated automatically by the assembler.

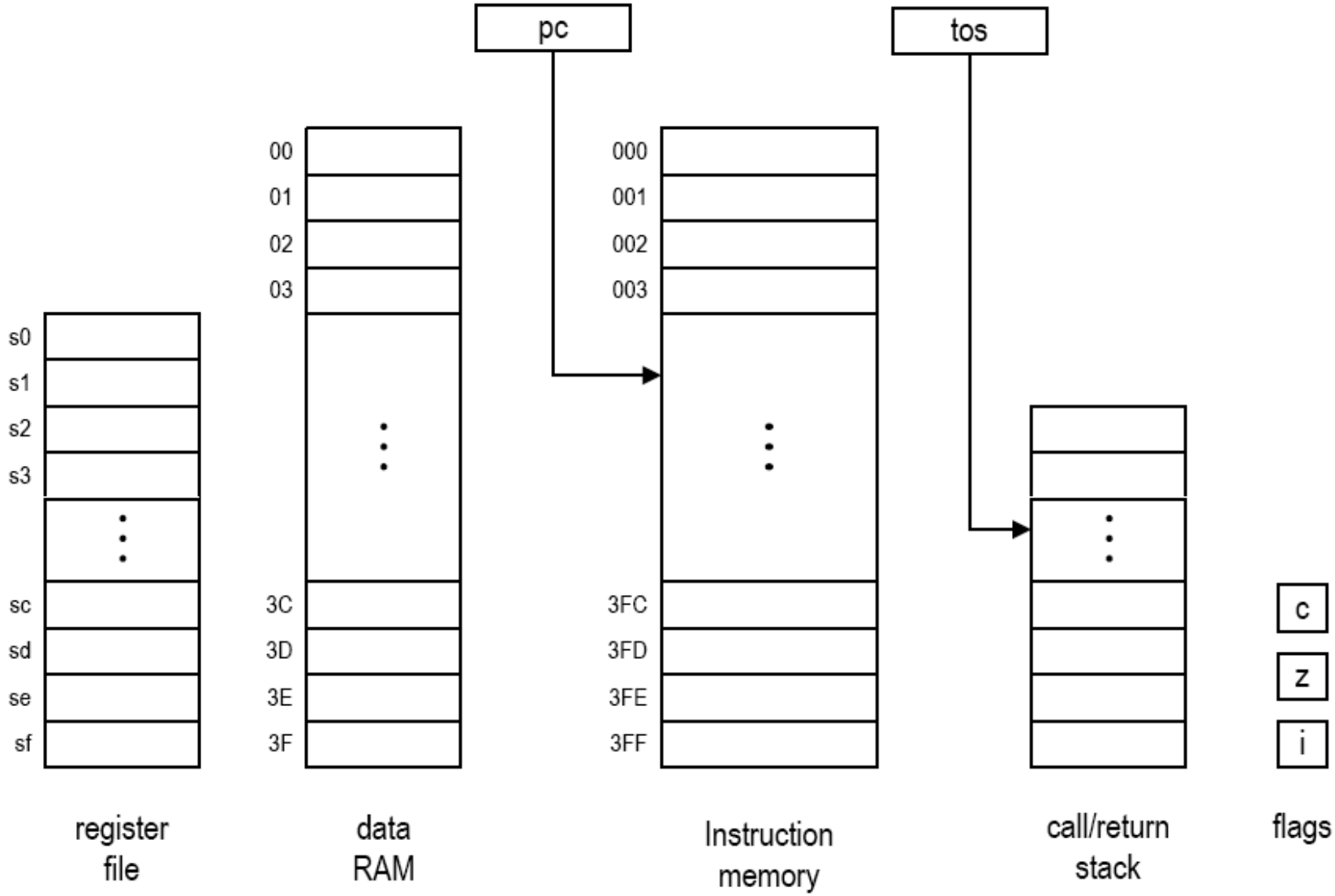
PicoBlaze Design Flow



Development Flow of a System with PicoBlaze



PicoBlaze Programming Model



Addressing Modes

Immediate mode

SUB s7, 07	s7 ← s7 – 07
ADDCY s2, 08	s2 ← s2 + 08 + C

Direct mode

ADD sa, sf	sa ← sa + sf
INPUT s5, 2a	PORT_ID ← 2a s5 ← IN_PORT

Indirect mode

STORE s3, (sa)	RAM[sa] ← s3
INPUT s9, (s2)	PORT_ID ← s2 s9 ← IN_PORT

PicoBlaze Instruction Set Summary (1)

Instruction	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD sX,kk	0	1	1	0	0	0	x	x	x	x	k	k	k	k	k	k	k	k
ADD sX,sY	0	1	1	0	0	1	x	x	x	x	y	y	y	y	0	0	0	0
ADDCY sX,kk	0	1	1	0	1	0	x	x	x	x	k	k	k	k	k	k	k	k
ADDCY sX,sY	0	1	1	0	1	1	x	x	x	x	y	y	y	y	0	0	0	0
AND sX,kk	0	0	1	0	1	0	x	x	x	x	k	k	k	k	k	k	k	k
AND sX,sY	0	0	1	0	1	1	x	x	x	x	y	y	y	y	0	0	0	0
CALL	1	1	0	0	0	0	0	0	a	a	a	a	a	a	a	a	a	a
CALL C	1	1	0	0	0	1	1	0	a	a	a	a	a	a	a	a	a	a
CALL NC	1	1	0	0	0	1	1	1	a	a	a	a	a	a	a	a	a	a
CALL NZ	1	1	0	0	0	1	0	1	a	a	a	a	a	a	a	a	a	a
CALL Z	1	1	0	0	0	1	0	0	a	a	a	a	a	a	a	a	a	a
COMPARE sX,kk	0	1	0	1	0	0	x	x	x	x	k	k	k	k	k	k	k	k
COMPARE sX,sY	0	1	0	1	0	1	x	x	x	x	y	y	y	y	0	0	0	0
DISABLE INTERRUPT	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ENABLE INTERRUPT	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
FETCH sX, ss	0	0	0	1	1	0	x	x	x	x	0	0	s	s	s	s	s	s
FETCH sX,(sY)	0	0	0	1	1	1	x	x	x	x	y	y	y	y	0	0	0	0
INPUT sX,(sY)	0	0	0	1	0	1	x	x	x	x	y	y	y	y	0	0	0	0
INPUT sX,pp	0	0	0	1	0	0	x	x	x	x	p	p	p	p	p	p	p	p

PicoBlaze Instruction Set Summary (2)

JUMP	1	1	0	1	0	0	0	0	a	a	a	a	a	a	a	a	a	a
JUMP C	1	1	0	1	0	1	1	0	a	a	a	a	a	a	a	a	a	a
JUMP NC	1	1	0	1	0	1	1	1	a	a	a	a	a	a	a	a	a	a
JUMP NZ	1	1	0	1	0	1	0	1	a	a	a	a	a	a	a	a	a	a
JUMP Z	1	1	0	1	0	1	0	0	a	a	a	a	a	a	a	a	a	a
LOAD sX,kk	0	0	0	0	0	0	x	x	x	x	k	k	k	k	k	k	k	k
LOAD sX,sY	0	0	0	0	0	1	x	x	x	x	y	y	y	y	0	0	0	0
OR sX,kk	0	0	1	1	0	0	x	x	x	x	k	k	k	k	k	k	k	k
OR sX,sY	0	0	1	1	0	1	x	x	x	x	y	y	y	y	0	0	0	0
OUTPUT sX,(sY)	1	0	1	1	0	1	x	x	x	x	y	y	y	y	0	0	0	0
OUTPUT sX,pp	1	0	1	1	0	0	x	x	x	x	p	p	p	p	p	p	p	p
RETURN	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
RETURN C	1	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
RETURN NC	1	0	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0
RETURN NZ	1	0	1	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0
RETURN Z	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
RETURNI DISABLE	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RETURNI ENABLE	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

PicoBlaze Instruction Set Summary (3)

RL sX	1	0	0	0	0	0	x	x	x	x	0	0	0	0	0	0	1	0
RR sX	1	0	0	0	0	0	x	x	x	x	0	0	0	0	1	1	0	0
SL0 sX	1	0	0	0	0	0	x	x	x	x	0	0	0	0	0	1	1	0
SL1 sX	1	0	0	0	0	0	x	x	x	x	0	0	0	0	0	1	1	1
SLA sX	1	0	0	0	0	0	x	x	x	x	0	0	0	0	0	0	0	0
SLX sX	1	0	0	0	0	0	x	x	x	x	0	0	0	0	0	1	0	0
SR0 sX	1	0	0	0	0	0	x	x	x	x	0	0	0	0	1	1	1	0
SR1 sX	1	0	0	0	0	0	x	x	x	x	0	0	0	0	1	1	1	1
SRA sX	1	0	0	0	0	0	x	x	x	x	0	0	0	0	1	0	0	0
SRX sX	1	0	0	0	0	0	x	x	x	x	0	0	0	0	1	0	1	0
STORE sX, ss	1	0	1	1	1	0	x	x	x	x	0	0	s	s	s	s	s	s
STORE sX,(sY)	1	0	1	1	1	1	x	x	x	x	y	y	y	y	0	0	0	0
SUB sX,kk	0	1	1	1	0	0	x	x	x	x	k	k	k	k	k	k	k	k
SUB sX,sY	0	1	1	1	0	1	x	x	x	x	y	y	y	y	0	0	0	0
SUBCY sX,kk	0	1	1	1	1	0	x	x	x	x	k	k	k	k	k	k	k	k
SUBCY sX,sY	0	1	1	1	1	1	x	x	x	x	y	y	y	y	0	0	0	0
TEST sX,kk	0	1	0	0	1	0	x	x	x	x	k	k	k	k	k	k	k	k
TEST sX,sY	0	1	0	0	1	1	x	x	x	x	y	y	y	y	0	0	0	0
XOR sX,kk	0	0	1	1	1	0	x	x	x	x	k	k	k	k	k	k	k	k
XOR sX,sY	0	0	1	1	1	1	x	x	x	x	y	y	y	y	0	0	0	0