VHDL, FPGA's and softcores

“Koen Martens” <gmc@metro.cx>

eth0:winter 2014

http://corpsmoderne.itch.io/flappy-space-program

OUTLINE

- VHDL
- SIMULATION
- SYNTHESIS: FPGA's and ASIC's
- SOFT-CORES
  - opencores.org
  - wishbone
  - zpu
  - gmzpu

VHDL

VHDL : VHSIC Hardware Description Language

VHSIC : Very High Speed Integrated Circuit

www.freerangefactory.org/dl/free_range_vhdl.pdf

http://it-ebooks.info/book/2757/

VHDL>1+1=10

0 + 0 = 00
0 + 1 = 01
1 + 0 = 01
1 + 1 = 10

A + B = C

VHDL>boilerplate

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
```

VHDL>entity

```
entity adder is
    -- simple adder, adds two 1-bit inputs and
outputs the 2-bit sum
    port(
        a   : in std_logic;
        b   : in std_logic;
        c   : out std_logic_vector(1 downto 0)
    );
end entity adder;
```

VHDL>architecture

```
architecture rtl of adder is
    signal a2   : std_logic_vector(1 downto 0);
    signal b2   : std_logic_vector(1 downto 0);
begin
    a2(0) <= a;
    a2(1) <= '0';
    b2(0) <= b;
    b2(1) <= '0';

    c <= std_logic_vector(unsigned(a2) +
                          unsigned(b2));

end architecture rtl;
```

VHDL>testbench [1/2]

```
architecture rtl of adder_TB is
    signal bit0 : std_logic;
    signal bit1 : std_logic;
    signal result :
              std_logic_vector(1 downto 0);

    component adder is
        port (
            a   : in std_logic;
            b   : in std_logic;
            c   : out
                std_logic_vector(1 downto 0)
        );
    end component adder;
```

VHDL>testbench [2/2]

```
    adder0 : adder
        port map (
         a => bit0, b => bit1, c => result);

    process
    begin
        bit0 <= '0'; bit1 <= '0';
        wait for 1 us;
        bit0 <= '1';
        wait for 1 us;
        bit0 <= '0'; bit1 <= '1';
        wait for 1 us;
        bit0 <= '1';
        wait for 1 us;
    end process;
end architecture rtl;
```

SIMULATION>tools

> vim
> ModelSim
  > Altera
> GHDL
> IDE's (Altera, Xilinx, ..)

SIMULATION>tools

DEMO: adder

VHDL>1+1=10

0 + 0 = 00
0 + 1 = 01
1 + 0 = 01
1 + 1 = 10

A + B = C

A	B	C(1)	C(0)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

VHDL>1+1=10

0 + 0 = 00
0 + 1 = 01
1 + 0 = 01
1 + 1 = 10

A + B = C

A	B	C(1)	C(0)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

C(1) = A AND B
C(0) = A XOR B

VHDL>better architecture

```
architecture rtl of adder is
    signal a2   : std_logic_vector(1 downto 0);
    signal b2   : std_logic_vector(1 downto 0);
begin
    a2(0) <= a;
    a2(1) <= '0';
    b2(0) <= b;
    b2(1) <= '0';

    c <= std_logic_vector(unsigned(a2) +
                          unsigned(b2));

end architecture rtl;
```

SIMULATION>better adder

DEMO: better adder

VHDL>clock divider

```
entity clkdiv is
    port (
        clk_i  : in std_logic;
        clk_o  : out std_logic
    );
end entity clkdiv;
```

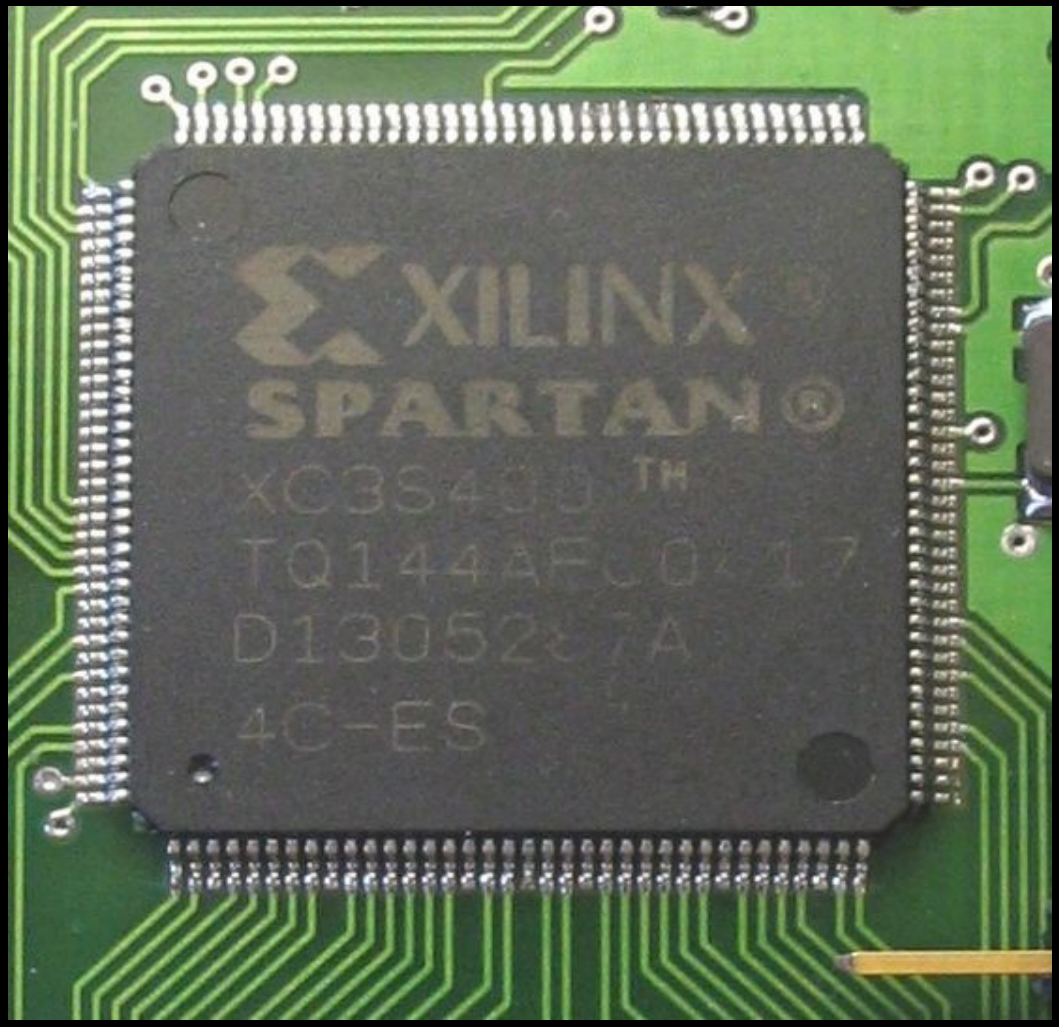
VHDL>clock divider

```
architecture rtl of clkdiv is
    signal dclk : std_logic:='0';
begin
    clock_divide:
    process(clk_i)
    begin
        if rising_edge(clk_i) then
            dclk <= not dclk;
        end if;
    end process;
    clk_o <= dclk;
end architecture rtl;
```

SIMULATION>clock divider

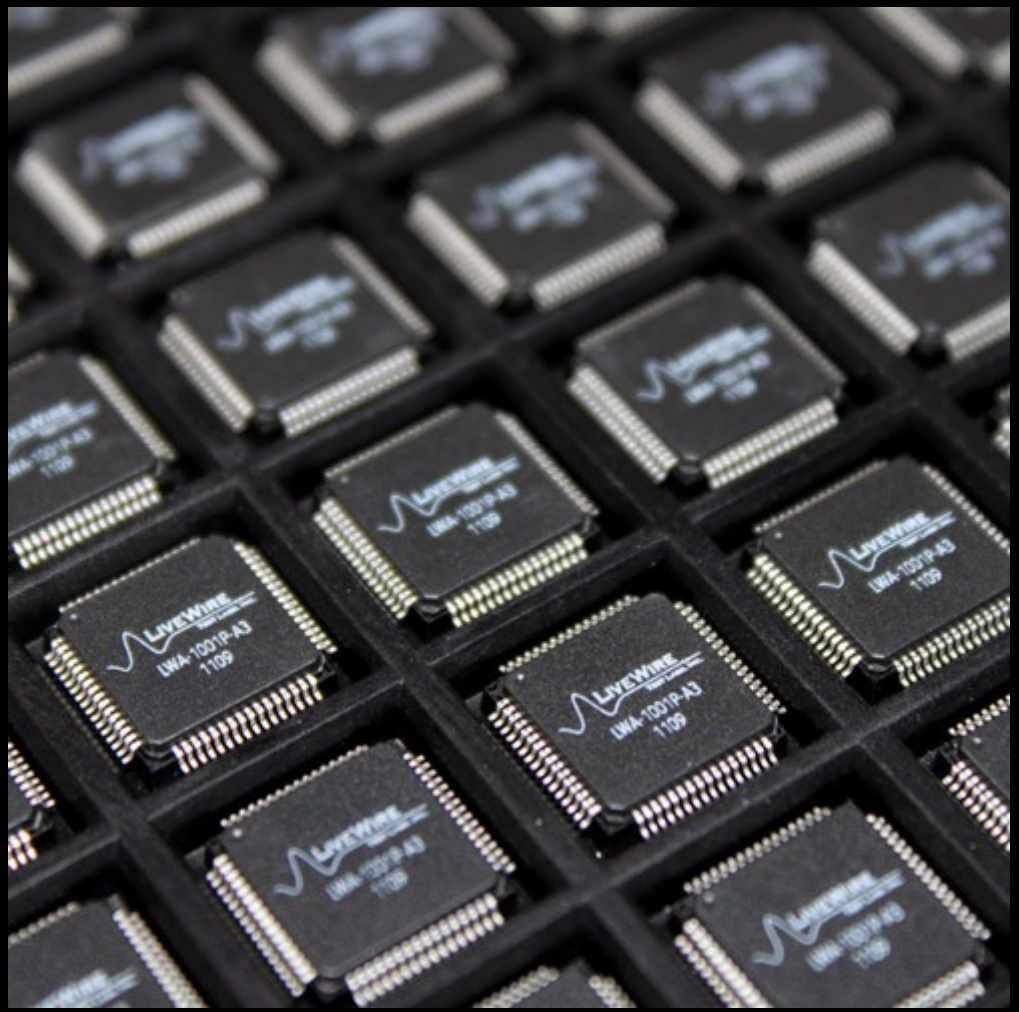
DEMO: Clock Divider

SYNTHESIS: FPGA's



http://en.wikipedia.org/wiki/File:Fpga_xilinx_spartan.jpg

SYNTHESIS: ASIC's



http://en.wikipedia.org/wiki/File:SSDTR-ASIC_technology.jpg

SYNTHESIS>difference w/ sim

- Default values for signals
- Type conversions
- Timing
- Layout / resource limitations

http://en.wikipedia.org/wiki/File:Fpga_xilinx_spartan.jpg

SOFTCORES

- VHDL
- SIMULATION
- SYNTHESIS: FPGA's and ASIC's
- SOFT-CORES
  - opencores.org
  - wishbone
  - zpu
  - gmzpu

SOFTCORES>opencores.org

the #1 community within open source hardware IP-cores

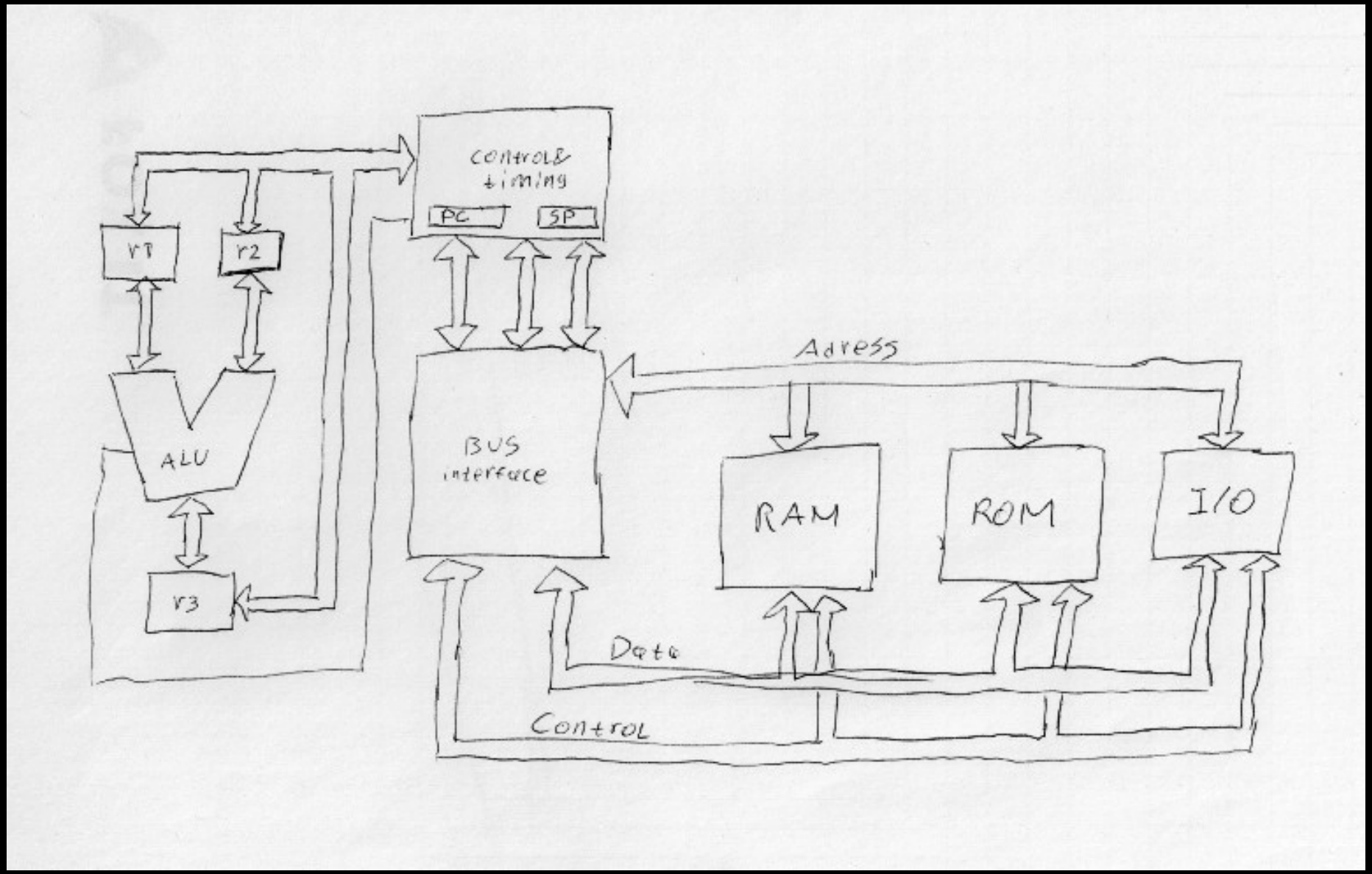
OpenCores is the world’s largest site/community for development of hardware IP cores as open source.

OpenCores.org host the source code for different digital HW projects (IP-cores, SoC, boards, etc) and support the users with different tools, platforms, forums and other useful information.

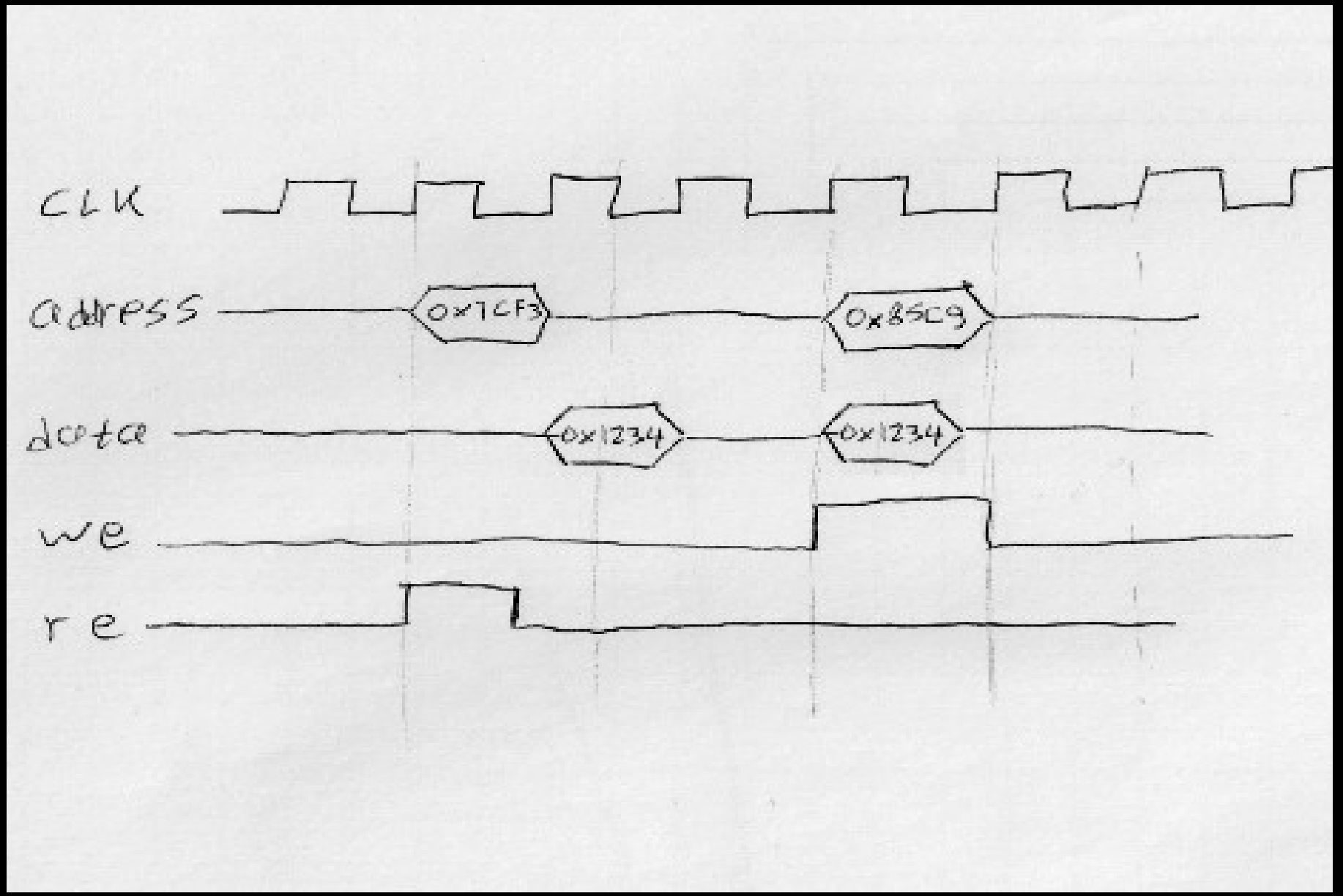
SOFTCORES>opencores.org

DEMO: opencores.org

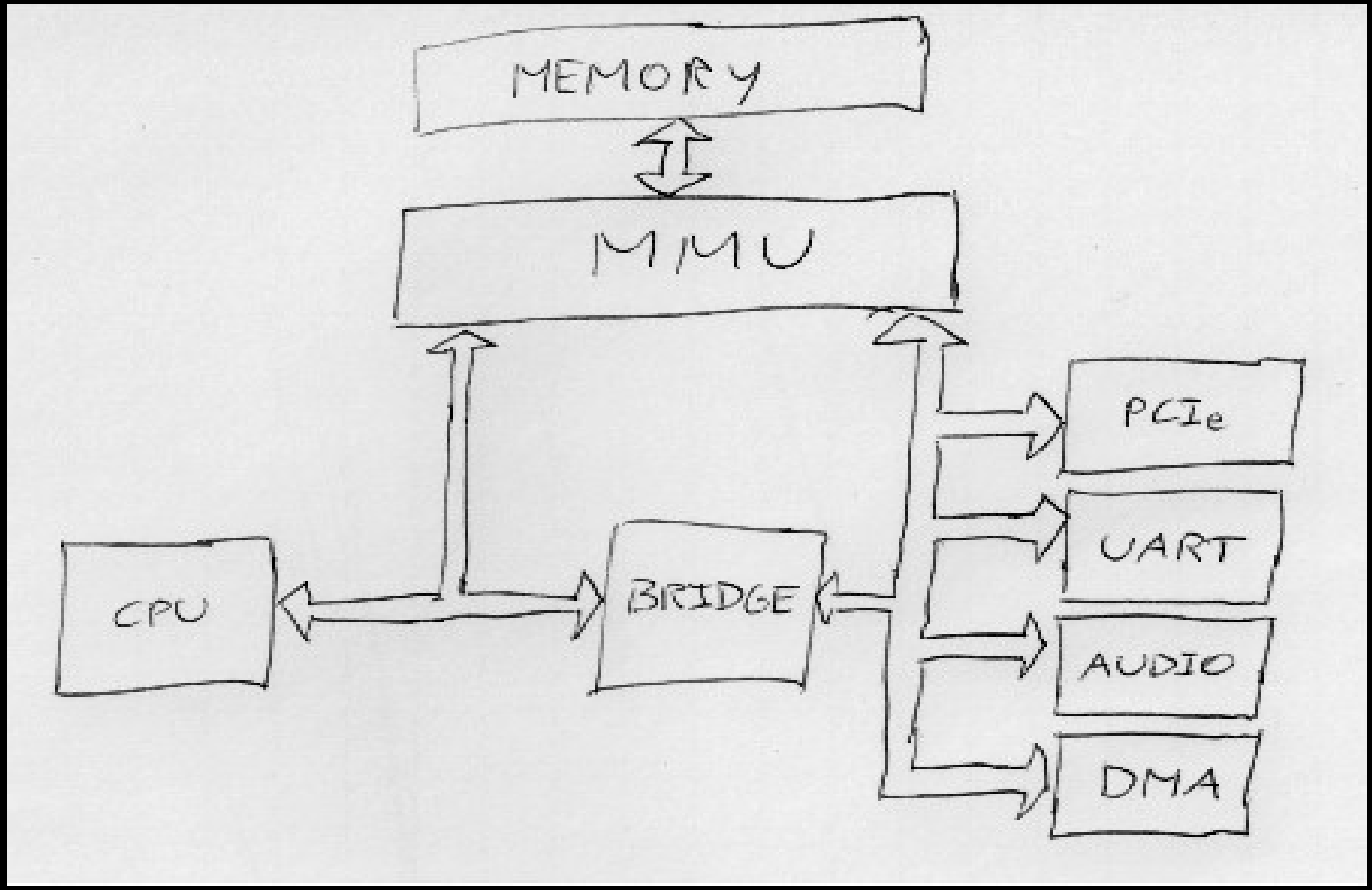
SOFTCORES>cpu design



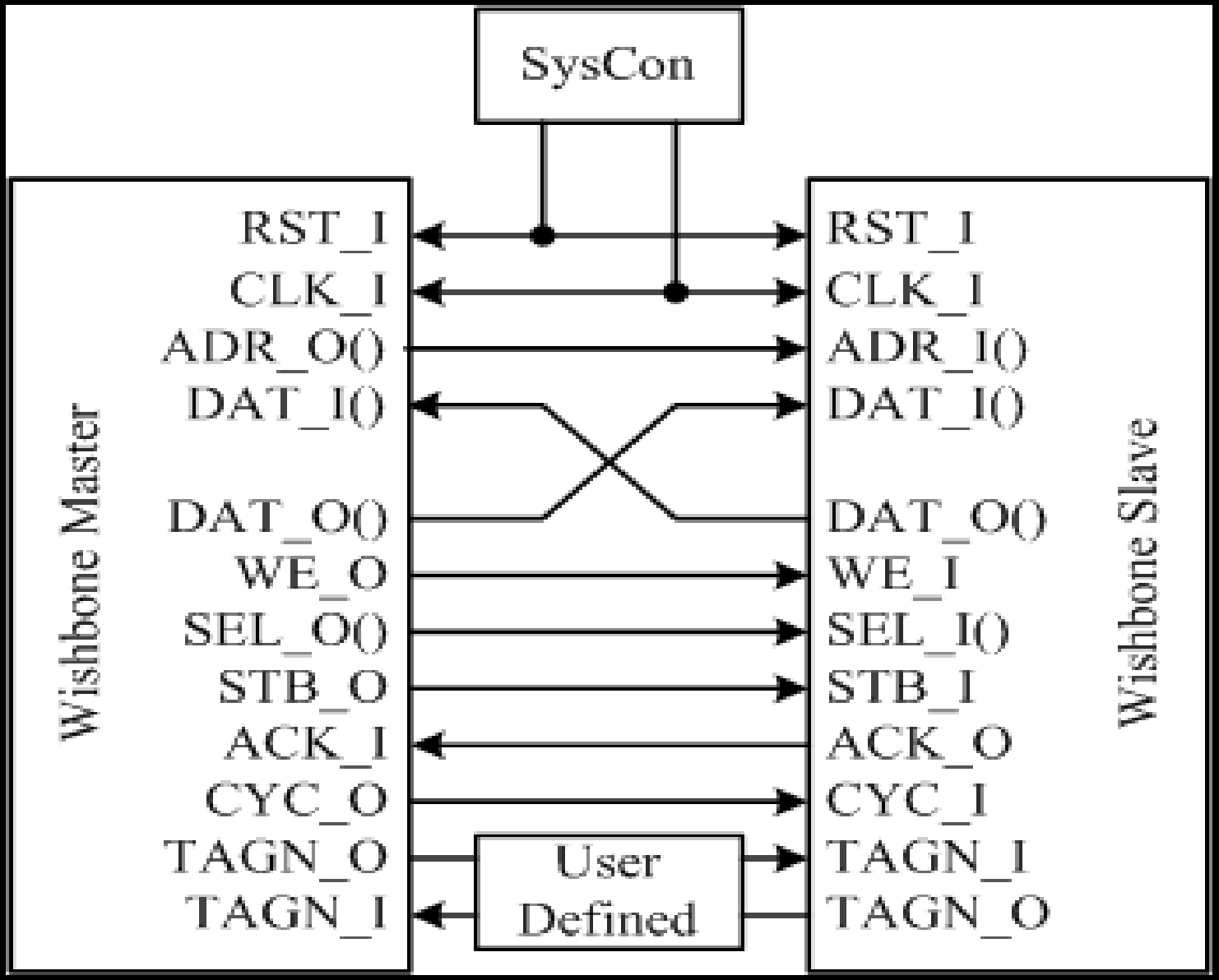
SOFTCORES>timing



SOFTCORES>SoC design

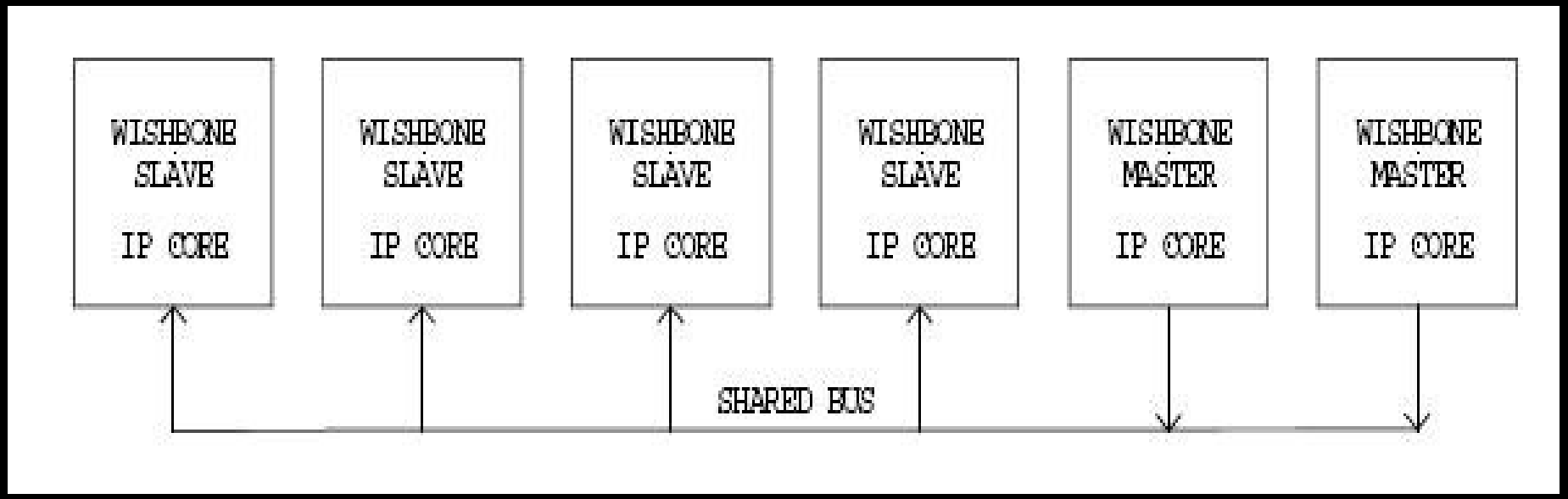


SOFTCORES>wishbone



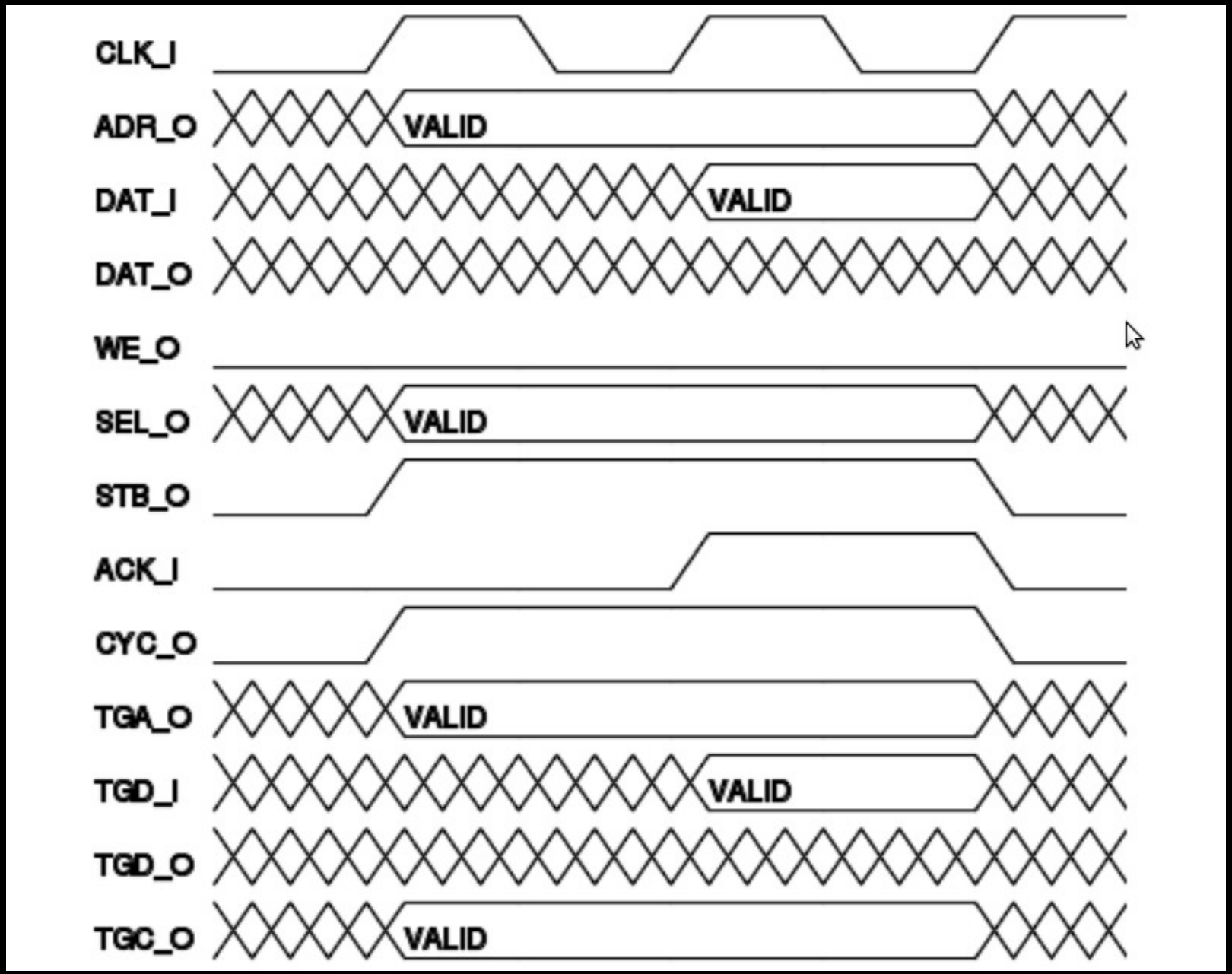
http://en.wikipedia.org/wiki/File:Wishbone_Interface.svg

SOFTCORES>wishbone



http://en.wikipedia.org/wiki/File:Wishbone_shared_bus.jpg

SOFTCORES>wishbone



SOFTCORES>zpu

http://opensource.zylin.com/zpu.htm

DEMO: a brief tour through the zpu VHDL

SOFTCORES>zpu

DEMO: zpugcc

SOFTCORES>gmzpu

- My own playground
- Based on zpu medium
- Goals:
  - Learn VHDL
  - Experiment with secure cpu design
  - Experiment with secure os design
- Short term:
  - Wishbone bus controller
  - Interrupt controller
  - Expand address space
  - Memory management

https://github.com/sonologic/gmzpu

VHDL, FPGA's and softcores

“Koen Martens” <gmc@metro.cx>