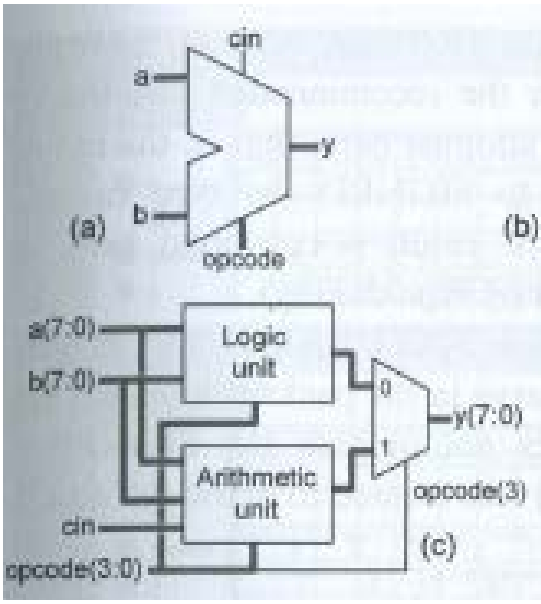
# Combinational Logic with VHDL

Example. Consider the circuit below which shows a simple computer Arithmetic Logic Unit (ALU). The inputs are *a* and *b*, and the output is *y*. Note that *a*, *b* and *y* are each one byte.




| Unit | Instruction | Operation | opcode |
|---|---|---|---|
| Logic | Complement a | y = NOT a | 0000 |
| | Complement b | y = NOT b | 0001 |
| | AND | y = a AND b | 0010 |
| | OR | y = a OR b | 0011 |
| | NAND | y = a NAND b | 0100 |
| | NOR | y = a NOR b | 0101 |
| | XOR | y = a XOR b | 0110 |
| | XNOR | y = a XNOR b | 0111 |
| Arithmetic | Transfer a | y = a | 1000 |
| | Transfer b | y = b | 1001 |
| | Increment a | y = a+1 | 1010 |
| | Increment b | y = b+1 | 1011 |
| | Decrement a | y = a–1 | 1100 |
| | Decrement b | y = b–1 | 1101 |
| | Add a and b | y = a+b | 1110 |
| | Add a and b with carry | y = a+b+cin | 1111 |

From Pedroni, *Circuit Design and Simulation with VHDL*, MIT Press, 2010

The 4-bit *opcode* input decides what the ALU does. For example, if the opcode is `0011`, the output is the OR-ing of *a* and *b*. The additional input to the ALU, *cin*, is used only in the addition operation.

Write VHDL code to implement the ALU using WITH/SELECT/WHEN statements. Assume that all arithmetic is unsigned.

**Library:**

```
--Library declaration
library IEEE;                      -- Declare which VHDL library
use IEEE.std_logic_1164.all;       -- and packages to use
use IEEE.std_logic_unsigned.all; -- for +, - operators
```

**Entity:**

```
-- Entity declaration
ENTITY alu IS
PORT(
    a, b      :  IN STD_LOGIC_VECTOR ( 7 DOWNTO 0 );
    cin       :  IN STD_LOGIC;
    opcode    :  IN STD_LOGIC_VECTOR ( 3 DOWNTO 0 );

    y         : OUT STD_LOGIC_VECTOR ( 7 DOWNTO 0 )
    );
END alu;
```

**Architecture:**

```
--Architecture body
ARCHITECTURE alu_arch OF alu IS

SIGNAL y_logical  :  STD_LOGIC_VECTOR ( 7 DOWNTO 0 );
SIGNAL y_arith    :  STD_LOGIC_VECTOR ( 7 DOWNTO 0 );

BEGIN
   WITH opcode( 2 downto 0 ) SELECT
      y_logical <=   NOT a WHEN "000" ,
                     NOT b WHEN "001" ,
                     a AND b WHEN "010" ,
                     a OR b WHEN "011" ,
                     a NAND b WHEN "100" ,
                     a NOR b WHEN "101" ,
                     a XOR b WHEN "110" ,
                     a XNOR b WHEN OTHERS;

WITH opcode( 2 downto 0 ) SELECT
      y_arith  <= a WHEN "000" ,
                  b WHEN "001" ,
                  a + 1 WHEN "010" ,
                  b + 1 WHEN "011" ,
                  a - 1 WHEN "100" ,
                  b - 1 WHEN "101" ,
                  a + b WHEN "110" ,
                  a + b + cin  WHEN OTHERS;

WITH opcode(3) SELECT
   y <=  y_logical WHEN '0',
         y_arith WHEN OTHERS;

END alu_arch;
```