Using LVDS Pins in FPGA-Based Designs

Notes:
1) Basically all current FPGAs allow their pins to operate in LVDS mode (see an example of LVDS usage in chapter 17). In such a case, all remaining pins in the same group must operate with 2.5V, because that is the (nominal) value of VDD for LVDS (therefore, the default option for the remaining pads, which normally is LVTTL3.3V, must be changed to 2.5V).
2) Usually, an LVDS pin is required to be a certain number of pads away from other, regular pins.
3) A macrofunction is not needed to use LVDS pins (even though there are macrofunctions that can be helpful when designing bidirectional LVDS buses).
4) Any LVDS input or output must be specified as a single (non-differential) signal (a single bit) in the ENTITY. This pin will be called the positive LVDS pin by the compiler. The accompanying negative LVDS pin will be assigned automatically by the compiler.

Example:
Say that we want to implement the functions \( x = a' \) and \( y = b' \) depicted in the figure below, where \( b \) and \( x \) are regular signals (3.3V LVCMOS or 3.3V LVTTL—the latter is usually the default) and \( a \) and \( y \) are differential signals (hence with two wires) of LVDS type. Note that internally any differential pair becomes just a single wire (the 2-to-1 and 1-to-2 conversions are provided during pin assignments).

Part 1: VHDL code
Note in the code below that \( a \) is just one wire, because that is the signal that VHDL will deal with (this is the internal \( a \) shown in the figure — after the LVDS receiver). Neither pad (input or output) is ever part of the code. The instantiation of the proper pin type is made during the pin assignment phase (next).

```vhdl
ENTITY using_lvds IS
  PORT (a, b: IN BIT; x, y: OUT BIT);
END ENTITY;
ARCHITECTURE using_lvds OF using_lvds IS
BEGIN
  x <= NOT a;
  y <= NOT b;
```

Part 2: Pin assignment

In what follows, compilation with Quartus II is used as an example, in which the Cyclone II EP2C35F672C6 FPGA available on Altera’s DE2 board is employed.

a) After compiling the code above, select Assignment > Pins. The screen below will be opened, but yet without the pins a(n) (negative a, that is, negative LVDS input) and y(n) (negative y, that is, negative LVDS output).

b) In the I/O Standard column, select LVDS for a and y.

c) Assign a pin that has the LVDSp (positive terminal) option to a (pin U20, of block 5, for example). Immediately, the line with a(n) is automatically created by the software (pin U21 is automatically selected, which is the LVDSn pair of U20).

d) Do the same for y. Say that pin U23 (LVDSp pin) is chosen; then the software will automatically include U24 in the list (LVDSn pair of U23).

e) In the I/O Standard column, select 2.5V for b and x.

f) Recompile the code.

g) Finally, check whether the correct pins were assigned. Select Assignments > Pins (see figure below) and click on each signal name (under Node Name) and observe that the corresponding pin is highlighted in the Top View - Wire Bond layout.
Part 3: Simulation

a) Open the simulator’s waveform editor and include in it the project’s signals (see figure below). Note that it includes $a(n)$ and $y(n)$.

b) Next, prepare the input waveforms, as in the next figure.

c) Run the simulation. The simulator will eliminate the negative LVDS signals, as shown below, because those signals are no longer important once you are inside the FPGA.