Advanced Logic Synthesis for Electronics
http://www.alse-fr.com

Nina Extension Board
User's Manual

June 2014, ver 1.1b **USER's Manual**

Congratulations!

You are now the happy owner of a **NINA** *New Integrated Network Adapter* Extension Board!

This board has been designed to add Ethernet connectivity as well as other useful and exciting features to simple and low cost Altera FPGA Kits like the Terasic kits: **DE0, DE0-nano, DE1, DE2**...

Features

The following features have been implemented on Nina:

- **RJ45 Ethernet 100 Mbits/s PHY**.
  To connect your FPGA project to Ethernet or the Internet.
- **Temperature Sensor**: the LM71 located on the board measures the temperature and can be read using a simple SPI interface.
- **1 x USB - RS232 UART** adapter.
  Inserting a UART in your FPGA design is the simplest and most compact way to exchange data with a PC (through a USB cable).
- **2 x Rx + 2 x TX LVDS ports**. These ports traverse an LVDS driver thus protecting the FPGA from the link. These dual links can be used to interface directly with fast LVDS peripherals (eg for motor control).
- **Simple monaural audio output** (standard 6.35mm stereo jack).
- **1 x Rotating Encoder**.
- **1 x MEMS Motion Sensing Tri-Axis Accelerometer**.
- **2 x Digital Servo connectors** (robotics!)
- 1 x **iR (infraRed) LED** for transmission (shared with 1 x Servo).

Warranty

The Nina Extension Kits are carefully tested and are 100% functional when they are shipped. No other warranty is provided since it is quite easy to misuse and damage the Nina extension or the FPGA kit.

You will have to be careful while handling the Kit and the extension board, both mechanically and electrically. Our experience is that both are reasonably robust, but careless use can also easily be fatal.

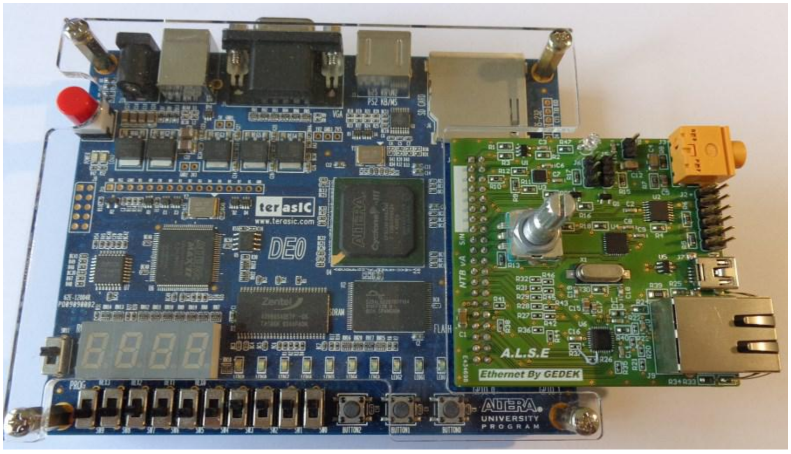
At any time, you can verify that the Nina extension is functioning properly by using the production test program provided (see later). If it fails you can contact ALSE and ask for a quotation to repair or replace the defective extension board. Again, no warranty applies.

Support files Archive

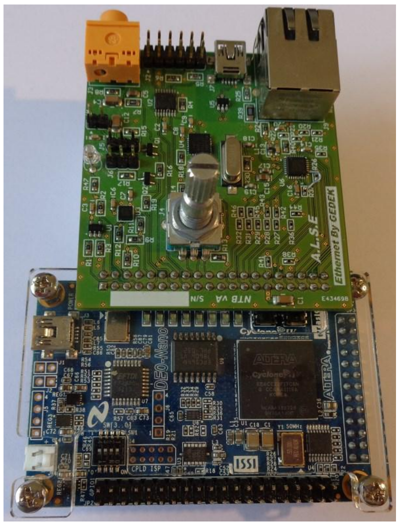
- You can download the latest archive of the support files for Nina from: http://www.alse-fr.com/archive/Nina_Archive.zip.
  The password is “MyNina!” (without the double quotes).
  This archive contains the latest documentation and support files, including the documentation for the on-board peripherals. The Production test bitstream is also included.

Installation

- Unpack carefully the Nina Extension board and protect it from ESD (Electro Static Discharges).
- Make sure the Kit (eg DE0) is turned OFF and that the power supply is <u>not</u> connected.
- Inspect the extension connector (**GPIO-0**) and make sure no pin is bent.
- Carefully insert the Nina Extension board to the selected extension connector (GPIO-0), as in the pictures below.
- Verify that the connector is fully inserted and is parallel to the board.



Example with DE0



Example with DE0-nano

- Apply the Power.
  The kit should continue to operate normally (if a functional design was previously loaded in the Flash).

Simple Verification with DE0

- Open the Quartus II Programmer. For example within a command shell :
  `%quartus_rootdir%/bin/quartus_pgmw`
- Verify that “**Hardware Setup**” displays the Blaster Interface:
  “*USB-Blaster [USB-0]*” eg. Else, install properly the Blaster drivers and verify the connection of your DE0 kit.
- Drag and drop the Production test programming file into the File window:
  `DE0_NINA_ProdTest.sof`
- Click on the “**Start**” button
- Make sure the DIP switches are all down :
  The temperature should be displayed !

Production Test with DE0

According to what you want to test, you can use your own design, or simply use a DE0 kit and load the Production test bitstream provided.

The following functions are implemented:

- ✔ DTMF tones cycled permanently, can be heard on the **Audio out** jack.
- ✔ If a 4.8V supply is connected to J8 and a **servo** on **J5**, it should sweep slowly and continuously in both directions (full swing).
- ✔ The **iR LED** (infrared) is toggled at 1 Hz. This can be observed with naked eyes (by looking directly in the axis, a pale read glow can be seen). This does also test J6 (shared funciton iR / servo).
- ✔ **Temperature** is displayed on the 7-seg LEDS (all SW down = “**00..00**”)
- ✔ Temperature is sent over **Ethernet** (use Temper.exe to read it).
  See details below.
- ✔ Positional Encoder counter on 7-seg LEDs when SW0 on (“0..**001**”).
  This should change when you turn the rotating button in either direction.
- ✔ RS232 characters sent/received are displayed on the 7-Seg when SW1 is on (“0..0**010**”). Use an RS232 Terminal Emulator @ 115200,N,8,1 no hs.
  Periodically, “*ALSE UART TEST - Echo Mode*” is sent by the FPGA.
- ✔ LVDS Loopback test display “A75E” with a decimal point when the test is successful. This test requires loop-back connections on J2 pins:
  1 → 10, 3 → 12 (D1 → R1), and 4 → 7, 6 → 9 (D2 → R2)
- ✔ Accelerometer: the 3 x axis are displayed on the 7-Seg when SW=”00.00**1000**”, “**10000**” and “**100000**” (SW3 through SW5).

Ethernet Test with DE0

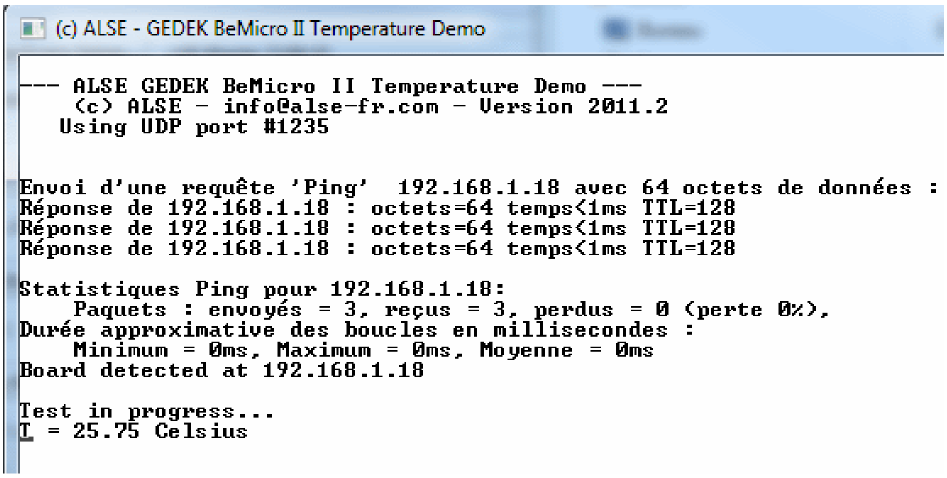
Set up your PC Ethernet interface as **Fixed IP** in the range **192.168.1.xx** with xx <u>not</u> being 00, 18, 254 nor 255 ! Try for example 192.168.1.238.

Power up the DE0 and load the Production test bitstream (as above)

Connect Nina to the Network using a standard Ethernet cable. You can have a switch inserted between your PC and Nina, this is not a problem.
A direct connection will also work.

Instantly, the two LEDs on the RJ45 connector should turn on (and one should blink) indicating a successful 100 MBits/s negotiation and the exchanges of frames over the network.

Launch the provided utility “Temper.exe”. The ping operation should succeed and the temperature should be displayed (and refreshed) in the utility window:



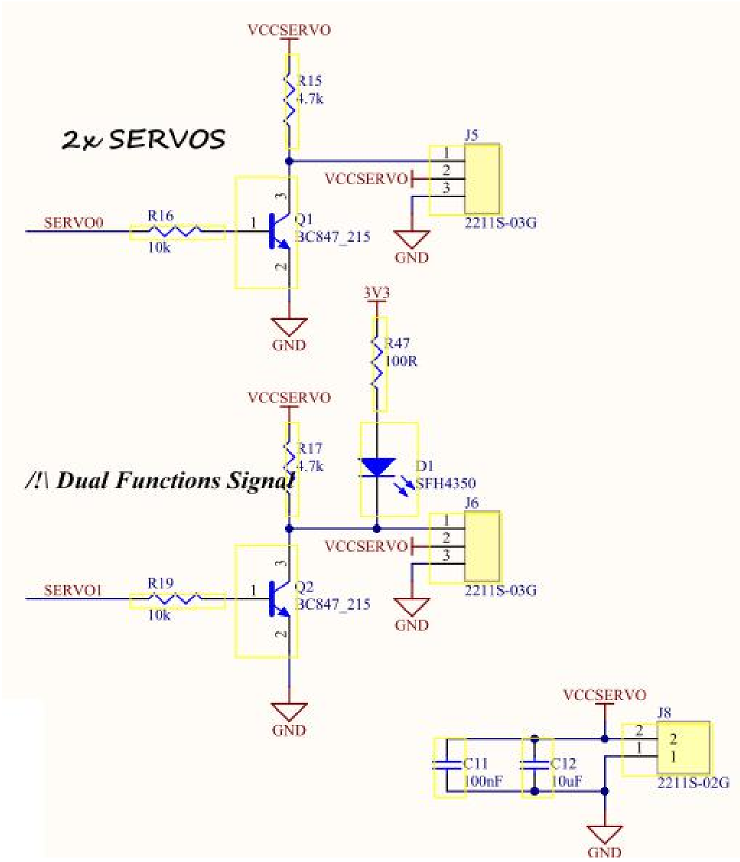
Nina Features

We are going to review the different features.

RC Servo Connectors

J8 is used to connect an external 4.8 Volts power supply to drive up to 2 x RC servo engines, to be attached to J5 and J6.

The following schematics illustrates this :



Note that pin 1 of the RC servo connectors (pin 1 has a triangle pointing to it) is for the command, pin 2 (center) is +4.8V positive power supply, and pin 3 is ground.

Keep in mind that there is an inversion between your design and the servo: a '1' on the FPGA output drives the transistor and brings a 0 Volt on the servo's command pin.

The **infraRed LED** us shared with the second RC servo, and these two features are mutually exclusive. This function sharing was required by the limited number of signals available on the 40 pin extension connector.

Be careful: an inversion of the power connector will probably destroy the servo and harm the Nina board.

J2 - LVDS Inputs & Outputs

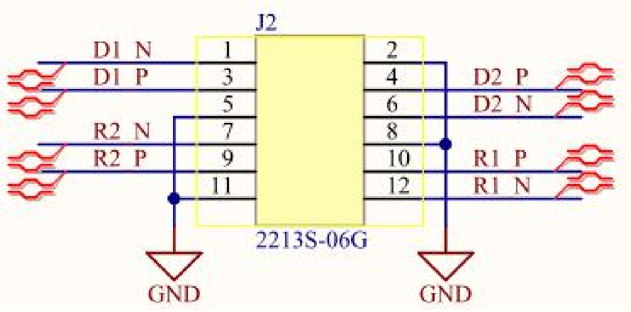
The 12 pins J2 connector offers 2 x differential inputs pairs R1 & R2, as well as 2 x differential output pairs D1 & D2. Outputs. Pins are grounded.

These differential I/Os are converted in both directions by the SN65LVDS049PWR chip into single-ended signals for the FPGA with the advantage of offering some protection between the LVDS signals and the FPGA. The driver can operate up to 400 Mbps ! The data sheet is provided.

The connector's twelve points are :

Pin Signal - Function

1 .... D1 output Pos - This pin is identified by a small triangle.
2 .... Gnd - 0V
3 .... D1 output Neg
4 .... D2 output Pos
5 .... Gnd - 0V
6 .... D2 output Neg
7 .... R2 input Neg
8 .... Gnd - 0V
9 .... R2 input Pos
10 ...... R1 input Pos
11 ...... Gnd - 0V
12....... R1 input Neg

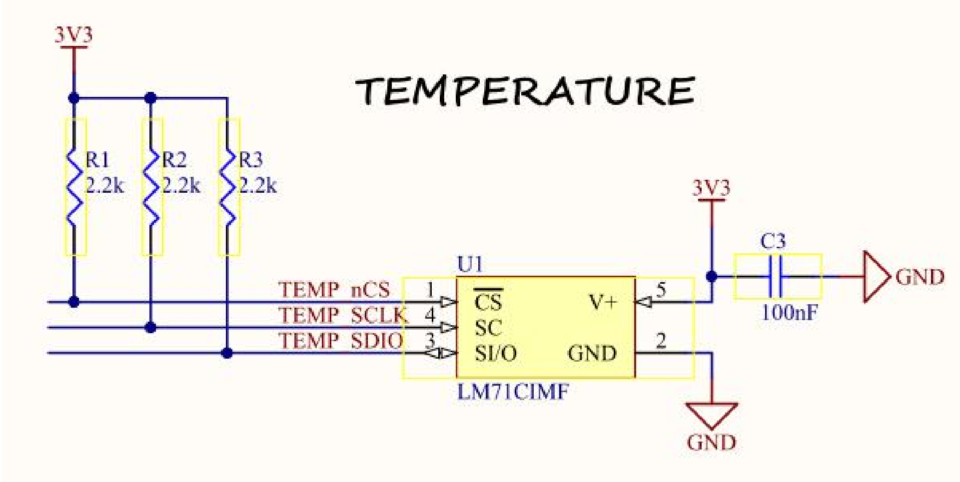


Temperature Sensor

The LM71 is a 13-bits plus Sign Temperature sensor chip.

The three signals connected to the FPGA are noted :
* **TEMP_nCS** (Device Selected, active low)
* **TEMP_sCLK** (SPI Serial Clock)
* **TEMP_SDIO** (SPI data, bidirectional)

Please refer to the sensor's Data Sheet (available in the archive).
Note that SDIO is truly bidirectional ! (you can write into the sensor, though this function is not very useful).

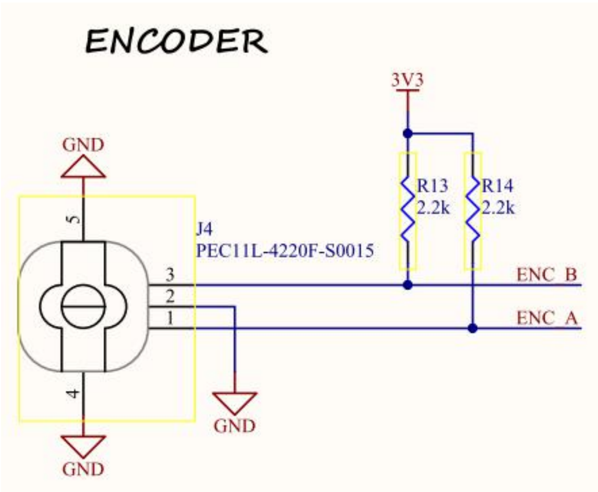


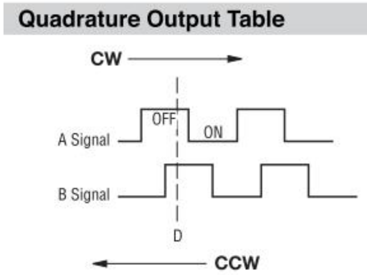
Reading the sensor from the FPGA is easy : a simple FSM with 3 or 4 states can easily do the job (his is one of our Training Course exercises).

Position Encoder

The Bourns PEC11L Encoder (rotating button) is a mechanical position encoder with two outputs noted ENC_A & ENC_B which are in quadrature, thus allowing to determine the direction of turn. One “click” goes through the 4 phases.

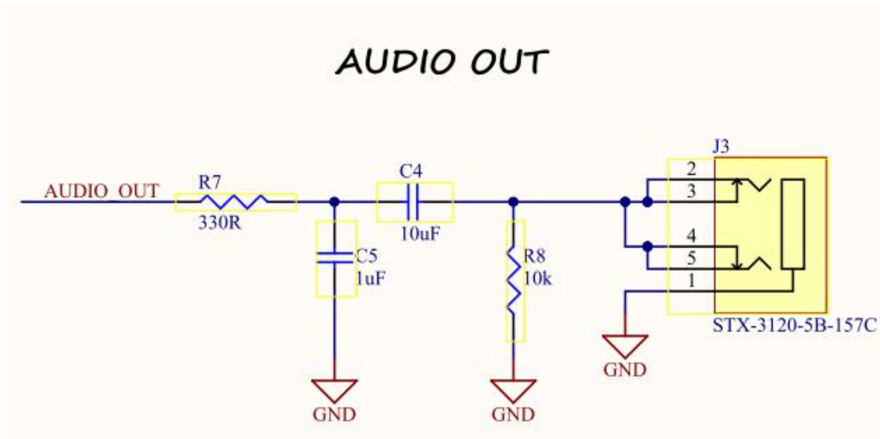
Since the outputs are mechanical switches, you can expect a lot of bounces, and if turned too fast, the switches can even skip pulses.





J3 Audio Out

This outputs simply a Low pass Filter with DC removal from the Audio_out output from the FPGA, driving bot the Left & Right signals of the standard audio jack. You can connect amplified speakers or just headphones.



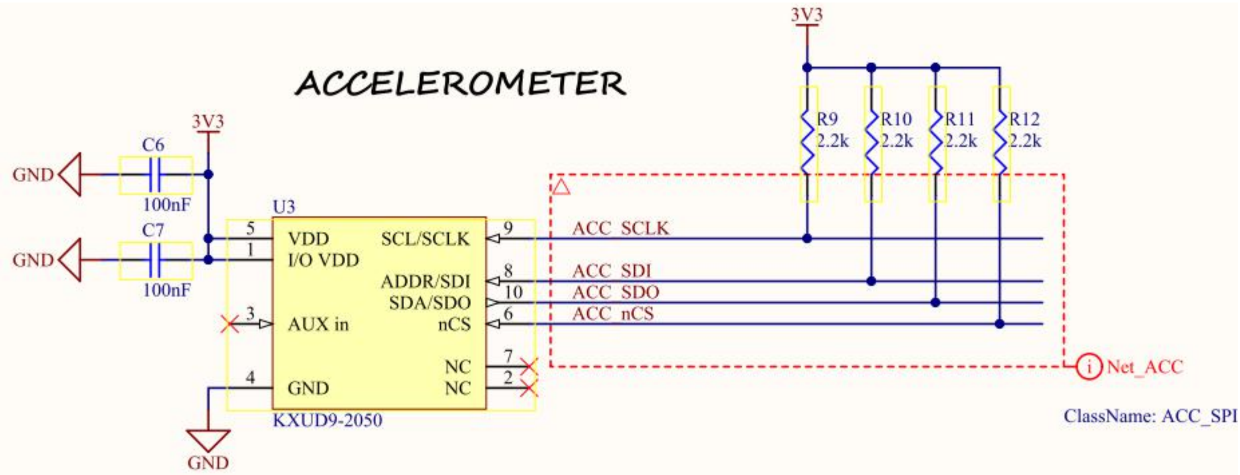
A simple PWM (Delta Sigma) conversion will allow to generate or reproduce sounds. The production test generates DTMF dialing tones.

Accelerometer

The **KXUD9-2050** chip (U3) is a tri-axis motion sensor / accelerometer.

For ease of use, we have connected it to the FPGA using the 4-wires SPI protocol C (without requiring an I2C controller).

It is a highly versatile and configurable device, and using it will require to read and understand the 23 pages data sheet that is also provided in the archive.



USB - UART

The **Silicon Labs** **CP2102** chip can be attached to a PC through J7, a standard mini-USB connector, and is then recognized as a COM Port: "***Silicon Labs CP210x USB to UART Bridge***".
If your Windows Device Manager doesn't recognize it as such, you need to install the proper driver from Silicon Labs.

Once properly recognized, the Device Manager will also report which RS232 port it does emulate (for example "COM3"). If you double-click on the device, you can specify the default RS232 speed and transmission format, with baud rates up to 115,200 or 128,000 bauds.
We recommend **115200 bauds, 8 bits data, No Parity, no flow control**.
The applications will be able to modify these default settings, but the above are what we recommend using.

The Advanced parameters settings allow you to emulate a different Serial COM Port number (if not in use).

To communicate with your FPGA, you can use any Terminal Emulator, like the free **PuTTY**. If you load the Production test bitstream, and run the terminal Emulator for this port at 115,200,N,8,1 no handshake, you will see a welcome string every few seconds and your characters will be echoed (and the ASCII code will appear on the 7 segments).

The Data Sheet of the CP2102 is provided in the archive.

Ethernet

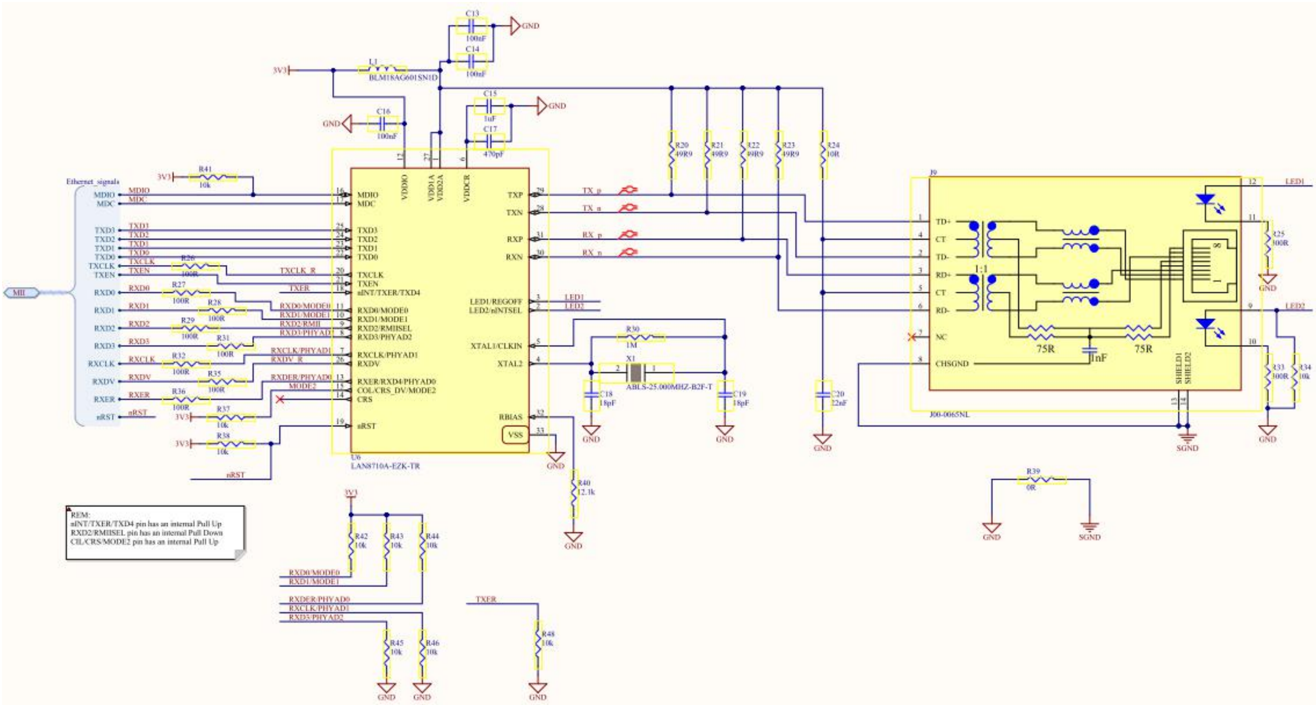
The Ethernet interface uses an SMSC/MicroChip 100M PHY: the **LAN8710a**.

The Data Sheet for the LAN8710a is available in the Archive.

The RJ45 connector includes the magnetics (transformer).

For ease of use, we have connected the LAN8710a to the FPGA using the MII interface (4 data bits, Rx + Tx).

If you use ALSE's Ethernet Communication kit ("**GEDEK**"), like we have done in the Production Tester bitstream, exchanging data between the PC and the FPGA through Ethernet is extremely simple and straightforward.
GEDEK fits in ~2k Logic Elements only.



The default settings used set up the PHY in auto-negotiation mode.
The value of PHY_AD is fixed to "001".

The MDIO interface is connected to the FPGA, which therefore can read and write into the LAN8710a for diagnostic or mode control purposes.

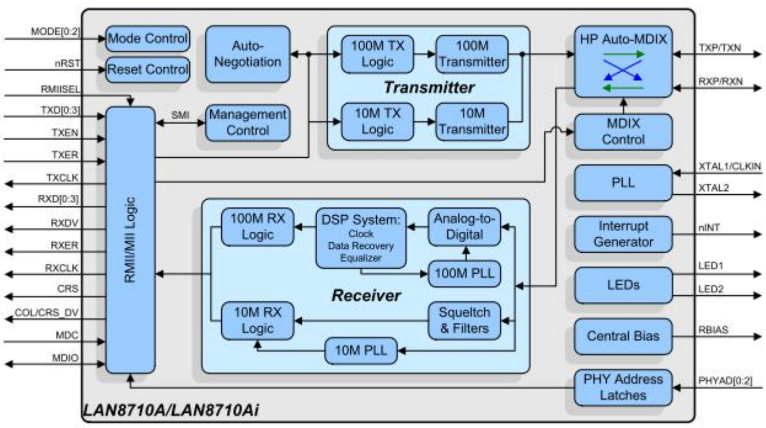


Figure 1.2 Architectural Overview

Appendix A

ALSE Nina Extension Board pinout for DE0

J4 Pin	DE0 GPIO-0	NINA signal	DE0 FPGA Pin
40	GPIO0_D[31]	MII_RXDV	PIN_U7
39	GPIO0_D[30]	MII_TXD3	PIN_V5
38	GPIO0_D[29]	MII_TXD2	PIN_W6
37	GPIO0_D[28]	MII_TXD1	PIN_W7
36	GPIO0_D[27]	MII_TXD0	PIN_V8
35	GPIO0_D[26]	MII_TXEN	PIN_T8
34	GPIO0_D[25]	MII_TXCLK	PIN_W10
33	GPIO0_D[24]	MII_nRST	PIN_Y10
32	GPIO0_D[23]	MII_MDC	PIN_V11
31	GPIO0_D[22]	MII_MDIO	PIN_R10
30	**GND**		
29	**VCC33**		
28	GPIO0_D[21]	MII_RXER	PIN_V12
27	GPIO0_D[20]	MII_RXD0	PIN_U13
26	GPIO0_D[19]	MII_RXD1	PIN_W13
25	GPIO0_D[18]	MII_RXD2	PIN_Y13
24	GPIO0_D[17]	MII_RXD3	PIN_U14
23	GPIO0_D[16]	MII_RXCLK	PIN_V14
22	GPIO0_D[15]	USBUART_RXD	PIN_AA4
20	GPIO0_D[14]	ENC_B	PIN_AB4
18	GPIO0_D[13]	R1	PIN_AA5
17	GPIO0_D[12]	R2	PIN_AB5
16	GPIO0_D[11]	D2	PIN_AA8
15	GPIO0_D[10]	D1	PIN_AB8
14	GPIO0_D[9]	AUDIO_OUT	PIN_AA10
13	GPIO0_D[8]	SERVO0	PIN_AB10
12	**GND**		
11	**VCC50**		
10	GPIO0_D[7]	ACC_SDO	PIN_AA13
9	GPIO0_D[6]	SERVO1	PIN_AB13
8	GPIO0_D[5]	ACC_SCLK	PIN_AB14
7	GPIO0_D[4]	ACC_nCS	PIN_AA14
6	GPIO0_D[3]	ACC_SDI	PIN_AB15
5	GPIO0_D[2]	TEMP_nCS	PIN_AA15
4	GPIO0_D[1]	TEMP_SCLK	PIN_AA16
2	GPIO0_D[0]	TEMP_SDIO	PIN_AB16
1	GPIO0_CLKIN[0]	NC	PIN_AB12
3	GPIO0_CLKIN[1]	NC	PIN_AA12
19	GPIO0_CLKOUT[0]	ENC_A	PIN_AB3
21	GPIO0_CLKOUT[1]	USBUART_TXD	PIN_AA3

Appendix B
Top Level Entity for use with DE0

```
-- DE0_top_nina.vhd
----------------------------------------------------------------
-- Copyright    : ALSE - http://www.alse-fr.com
-- Contact      : info@alse-fr.com
-- Project Name : DE0_TOP_NINA (GOLDEN TOP)
-- Block Name   : DE0_TOP_NINA
-- Description  : Top Level
----------------------------------------------------------------
Library IEEE;
  use IEEE.std_logic_1164.all;
  use IEEE.numeric_std.all;

-- ------------------------
    Entity DE0_TOP_NINA is
-- ------------------------
  port (
          CLK50MHZ   : in     std_logic;  - System Clock
          BUTTON     : in     std_logic_vector(2 downto 0); --  Push Buttons 0 .. 2
          SW         : in     std_logic_vector(9 downto 0); --  DPDT Switches (0..9)
          --  7-SEG Display, active low, 0 to 6 = A,B,C,D,E,--
          HEX0_D     : out    std_logic_vector(6 downto 0); --  Digit 0
          HEX0_DP    : out    std_logic;                    --  Digit DP 0
          HEX1_D     : out    std_logic_vector(6 downto 0); --  Digit 1
          HEX1_DP    : out    std_logic;                    --  Digit DP 1
          HEX2_D     : out    std_logic_vector(6 downto 0); --  Digit 2
          HEX2_DP    : out    std_logic;                    --  Digit DP 2
          HEX3_D     : out    std_logic_vector(6 downto 0); --  Digit 3
          HEX3_DP    : out    std_logic;                    --  Digit DP 3

          LEDG       : out    std_logic_vector(9 downto 0);   --  10 x LEDs (Green)

    -- --------------------------------------------------------
    --  ALSE NINA Extension
    -- --------------------------------------------------------
        -- TEMPERATURE SENSOR
          TEMP_SCLK   : out   std_logic;
          TEMP_nCS    : out   std_logic;
          TEMP_SDIO   : in    std_logic;
        -- ACCELEROMETER
          ACC_SCLK    : out   std_logic;
          ACC_nCS     : out   std_logic;
          ACC_SDI     : out   std_logic;
          ACC_SDO     : in    std_logic;
        -- SERVO or TX IR (on SERVO1)
          SERVO_S     : out   std_logic_vector(2 downto 1);
        -- AUDIO
          AUDIO_OUT   : out   std_logic;
        -- UNIPOLAR SIGNALS to LVDS INTERFACE
          LVDS_D      : out   std_logic_vector(1 to 2);
          LVDS_R      : in    std_logic_vector(1 to 2);
        -- ENCODER
          ENC_A       : in    std_logic;
          ENC_B       : in    std_logic;
        -- USB/UART
          USBUART_TXD : out   std_logic;
          USBUART_RXD : in    std_logic;
        -- ETHERNET
          MII_MDIO    : inout std_logic;
          MII_MDC     : out   std_logic;
          MII_RXD     : IN    std_logic_vector(3 downto 0);
          MII_RXCLK   : IN    std_logic;
          MII_RXER    : INout std_logic;
          MII_RXDV    : IN    std_logic;
          MII_nRST    : OUT   std_logic;
          MII_TXEN    : OUT   std_logic;
          MII_TXCLK   : IN    std_logic;
          MII_TXD     : OUT   std_logic_vector(3 downto 0)
        );
    End Entity DE0_TOP_NINA;
-- ------------------------
```

Conclusion

You should now have all the information needed to use all the features included in the Nina Extension Board.

LET US KNOW what you do with this board !

Suggestions and any kind of feedback are always welcome.

Best regards,

Bertrand CUZEAU
Technical Manager A.L.S.E.
info@alse-fr.com