DEC RL01/RL02 DISK-DRIVE EMULATOR

User Manual for the DE10-Lite board

Version 2.2

DE10-Lite board with emulator interface
Embedded environment based on the Altera MAX 10 FPGA.

Emulates up to 4 RL01/RL02 drives simultaneously
Supports mixed environment of emulated + real RL drives
Access to 17 x 4 RL01/RL02 configurations sets
WLAN basics implemented (based on ESP8266)

FPGA based disk emulator for the DEC RL01 and RL02 disk drives

CONTENTS

Secure the vintage software and preserve it on new technology

Overview:

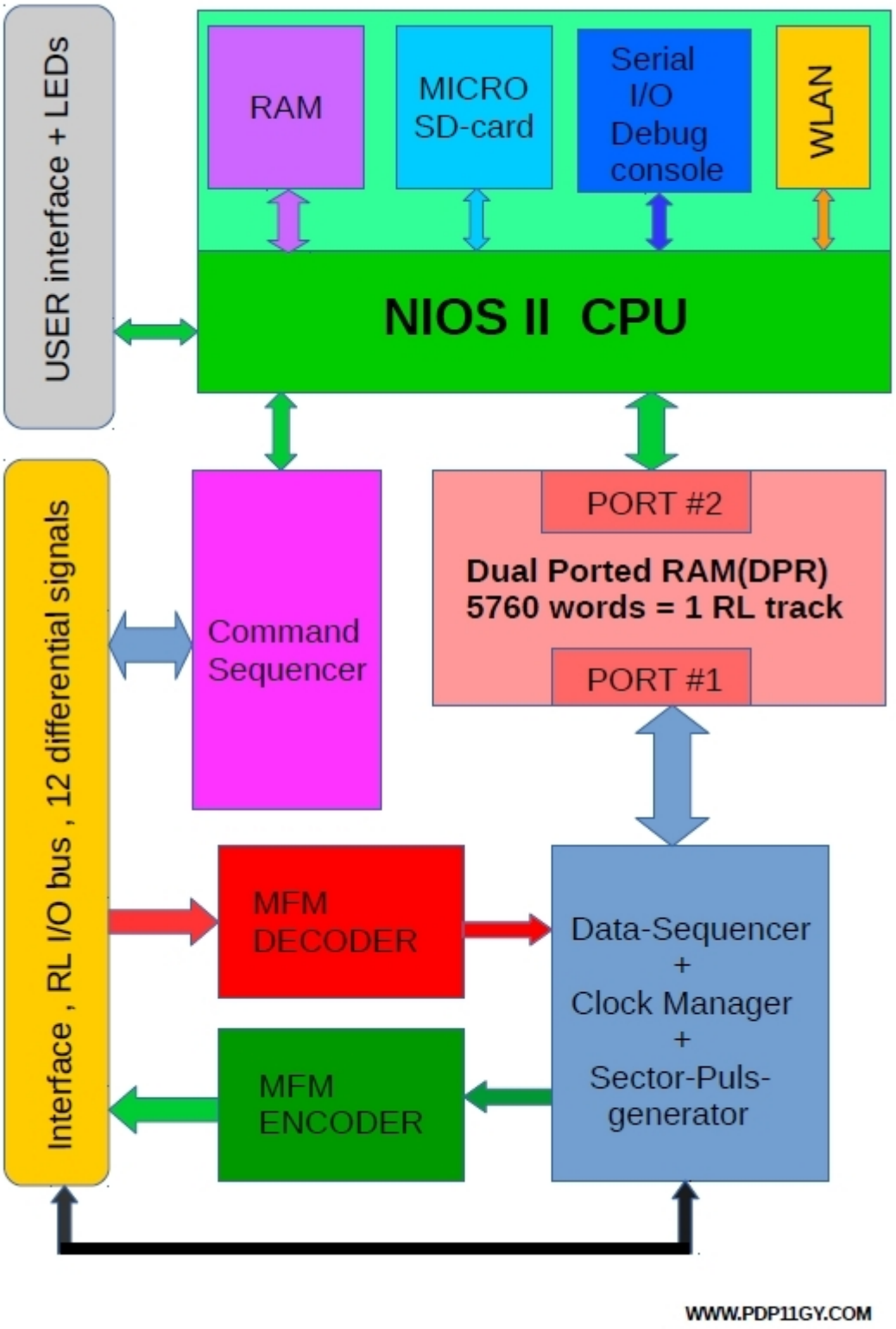
Project Start was in 2009. In the initial phase, a PIC processor and shortly afterwards an ARM SBC was used. This idea quickly turned out to be unrealizable and I walked step by step into the FPGA world. First, I had worked on the MAX-2 CPLD. The realization failed due to the non-existent onchip-memory. Then the DE1 board was used with the CYCLON 2 FPGA. The RL01 emulator in the first version was completed in **2012** (see video). Then, the next versions were realized with the DE0-Nano, BeMicro CV board and it was now possible to emulate up to 4 RL02 disk drives simultaneously. Unfortunately, the BeMicro CV board is no longer available until now (JAN 2017), to bad and it was a big setback.
The current version has been ported to the DE10-Lite board and **many** new options have been developed, such as basics of WLAN support.

Architecture:

Basically, the design of my DEC RL02/RL01 disk drive emulator works like a Solid-State-Disk(SSD), interfacing the DEC RL-disk serial bus signals (1980) to the current FPGA technology. The heart of my design is a DPR (Dual Ported RAM) which can hold one RL-track. DPR-Port #1 is responsible for the firmware communication like MFM De/En-coding, provides the complete data transfer to/from DPR-Port #1 based on a data sequencer and runs completely automatically. DPR-Port #2 is responsible for the data transfer to/from the (NIOSII) CPU. Sounds easy, but it was very difficult to construct the right data format emulating the cartridge format with CRC and all the servo information. The (NIOSII) CPU is also responsible for the data transfer in the memory with up to 4 emulated RL drives and finally also for the transfer to/from the SD card. The operation of the RL02/RL01 emulator is best viewed with a VIDEO via YouTube, however in the first version from **2012**, based on the DE1-Board.
https://www.youtube.com/watch?v=0i3ypBU39as

Next page: Block diagram architecture overview

Architecture: DE10-Lite/MAX10 based RL Emulator



Data format

The DEC RL01/RL02 disk drive did have a capacity of 5.2MB/10.4MB, 2 Heads(surfaces), 256/512 cylinder, 40 sectors/track. 1 sector contains 128 16-bit words (256 Byte) of Data + 12 16-Bit words for Servo/Header/CRC Data = 140 words(280 Byte)/sector. The emulater is using the .DEC format which contains all the information plus a serial number and the bad sector file. The size of the .DEC file for the RL02 is 11.521 KB and for the RL01 5.761 KB. Another disk format is the disk image structure .DSK which is often used for CPU emulators. To convert this data, the necessary programs are available on my homepage or direct by: www.2jo.de/pdp11/rlutils/rlutils.zip

Release notes

New Features & Enhancements

Firmware: MFM decoder schematic has been improved and merged.

Emulated cartridge SERIAL NUMBER (SN) handling:
Up to the version 1.5, the handling of the cartridge serial numbers was static, and by default, always the same serial number was used. This can result in errors by some DEC operating systems. In version 2.0 or higher , the cartridge serial number can be set with the content of the file **SN**x**.TXT** and can be changed individually for each subset at any time with a text editor. It contains the serial numbers for each 4 cartridges per disk-subset (DL0: to DL3:) in the form of 2 16-bit integer values (HEX-notation). As long as the file **SN**x**.TXT** is present, the serial number with the values from the file **SN**x**.TXT** will be always set after loading the RL images. If this is no longer necessary, then simply delete the file **SN**x**.TXT.** Now, the serial numbers of disk image are used. The cartridge serial number is located on the last cylinder, RL01=256, RL02=512. You can also check the serial number with a HEX editor by opening a RL02 emulator image file and navigating to the offset (h) B3A610. For example, if serial number is 1234 and 5678
00B3A610 00 00 00 80 34 12 78 56 00 00 00 00 FF FF FF FF

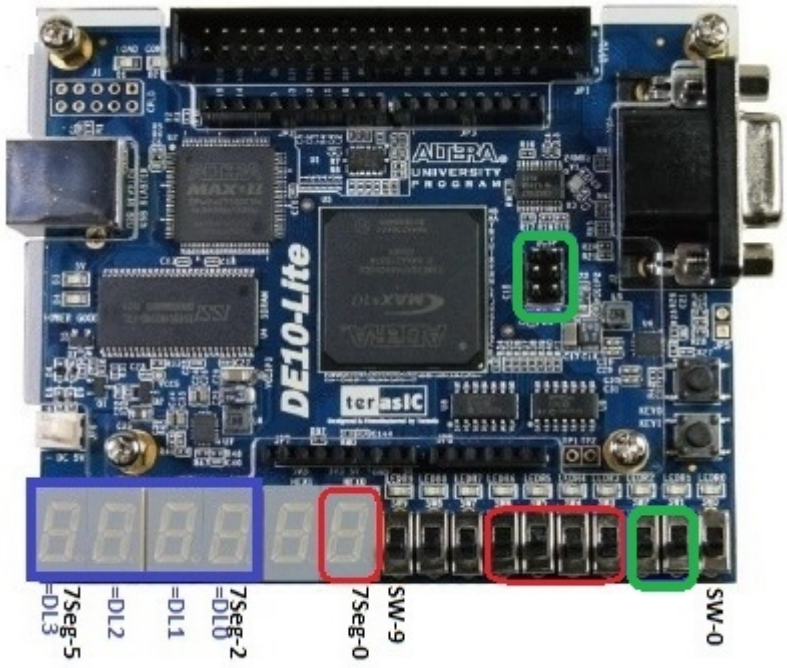
Customize the disk-subset environment:
To make the operation and documentation easier, a configuration file **RL**x**.TXT** for each disk subset is now implemented. In this file, you can write your own notes with a standard text editor.

WLAN support:
The current implementation only supports the output of some debug messages with WLAN (UDP transmission UART - WiFi passthrough). A further WLAN development is not planned. For questions, please send an e-mail.

Configuration & Jumper-Settings

1. DE10-Lite board

http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=english&No=1021



SW-0: No functional meaning , only for LED expansion/test purpose.

SW-1 - SW-2 : **WLAN** operating mode, details in chapter WLAN mode.

SW-3 – SW-6 : **SELECT** mode. With these 4 switches, 16 sets (0 - F) with 4 units each can be selected. Current used set is displayed with 7seg-0. Details in chapter SELECT mode.

SW-7: **ONLINE / OFFLINE :** ON = ONLINE. Depending on the switch position, a different LED pattern is displayed.

SW-8: **DEBUG** mode : ON = DEBUG mode on

SW-9: **Format/Initialize** the micro SD-Card at the next restart. (= **Switch-1**)

7Seg-0: Shows the currently used disk set (0 – F) in SELECT mode.

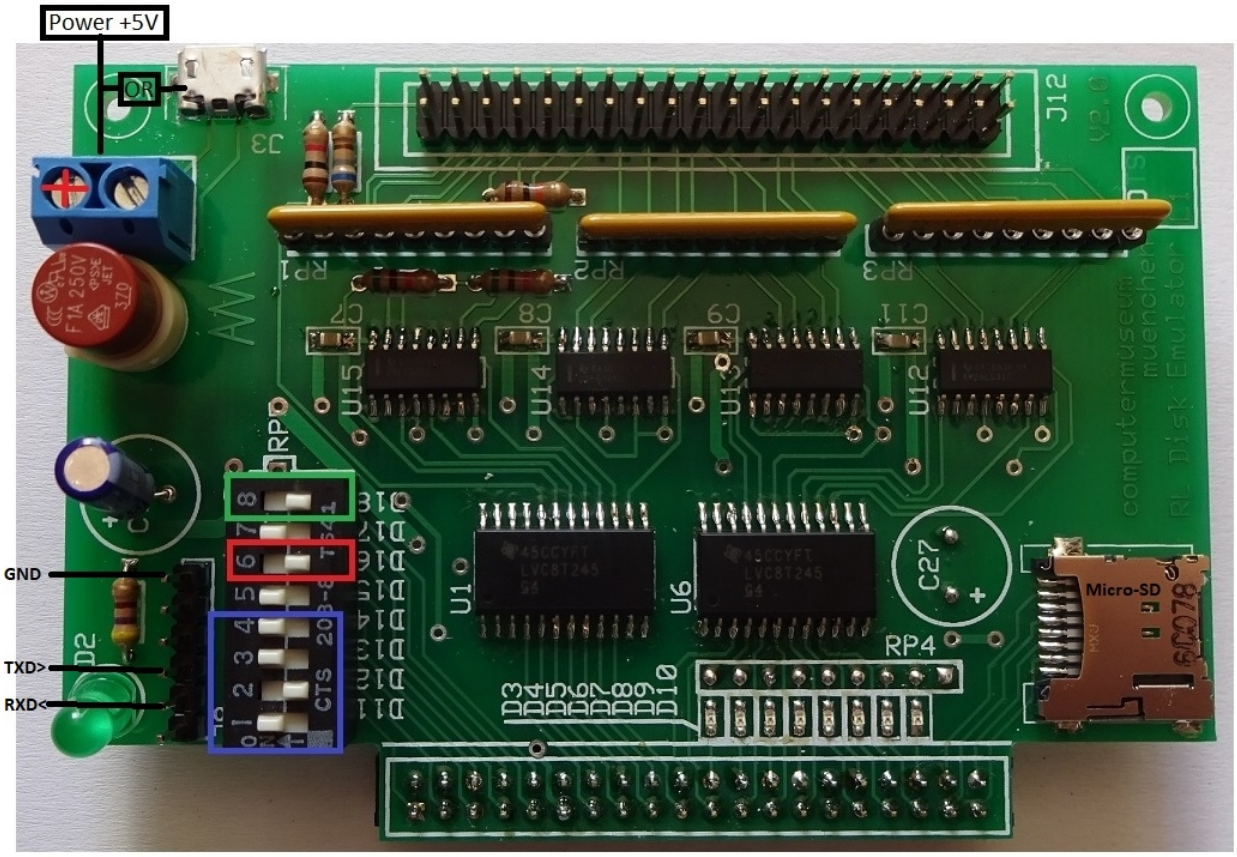
7Seg-1: Displays the RL unit-nr, which is currently active, also the external disks if available.

7Seg-2 -5: Displays the configured/emulated disk drives DL0(7seg-2) to DL3. A configured disk is displayed with a circular cursor in the 7-segment display, otherwise the "-" character appears.

Button and Switches

Button 0	Reset / Restart
Button 1	Reconfigure. More details in section example.

2. Emulator interface, see also: http://fafner.dyndns.org/~heuberger/rlfertig/Readme.pdf



Implementation/architecture of the Interface board:

The interface board consists of the following components:

- **2** LVC8T245 = level converter
- **2** AM26LS31 = Transmitter
- **2** AM26LS32 = Receiver
- **8** LEDs
- **1** 8 pin DIP-switch
- **3** pluggable resistor networks
- **1** holder for a micro SD card
- **2** connectors (40 pin)
- **1** 6 pin connector for serial connection with 19200 baud based on + 3.3 Volt. A “RoHS TTL-232R-3V3” USB converter will provide PC-connection.
- **2** 5 Volt Power distributen connectors micro-USB connector: This is a simple and inexpensive way for a battery backup implementation with a standard handy-power bank.

Interface LED's (from right to left):

LED 0	heartbeat (blinking)
LED 1	Power OK
LED 2	Write in progress
LED 3	Read/Seek in progress
LED 4	Configured Unit dl3 active
LED 5	Configured Unit dl2 active
LED 6	Configured Unit dl1 active
LED 7	Configured Unit dl0 active

Interface switches (from top to bottom):

SWITCH 8	WLAN mode enable
SWITCH 7	Force Power OK
SWITCH 6	Select mode enable
SWITCH 5	RL drive type, RL01 or RL02
SWITCH 4	configure emulated RL drive dl0
SWITCH 3	configure emulated RL drive dl1
SWITCH 2	configure emulated RL drive dl2
SWITCH 1	configure emulated RL drive dl3

Pluggable resistor networks:
Necessary if the interface board is connected directly to the RL controller.

Serial Interface:
The serial interface is configured for **19200** baud based on a 6 pin connector with + 3.3 Volt.
A "RoHS TTL-232R-3V3" USB converter will provide PC-connection.

Battery Backup:
The additional micro-USB connector is available for connecting a standard Handy Power Bank.
This is a very simple and cost-effective Battery Backup implementatio.

Micro-SD:
The emulator software supports FAT32 for the Micro-SD card.
A **RL02** emulator-image file has a size of **11.521 KB** with file extension ".**DEC**"
A **RL01** emulator-image file has a size of **5.761 KB** with file extension ".**DEC**"

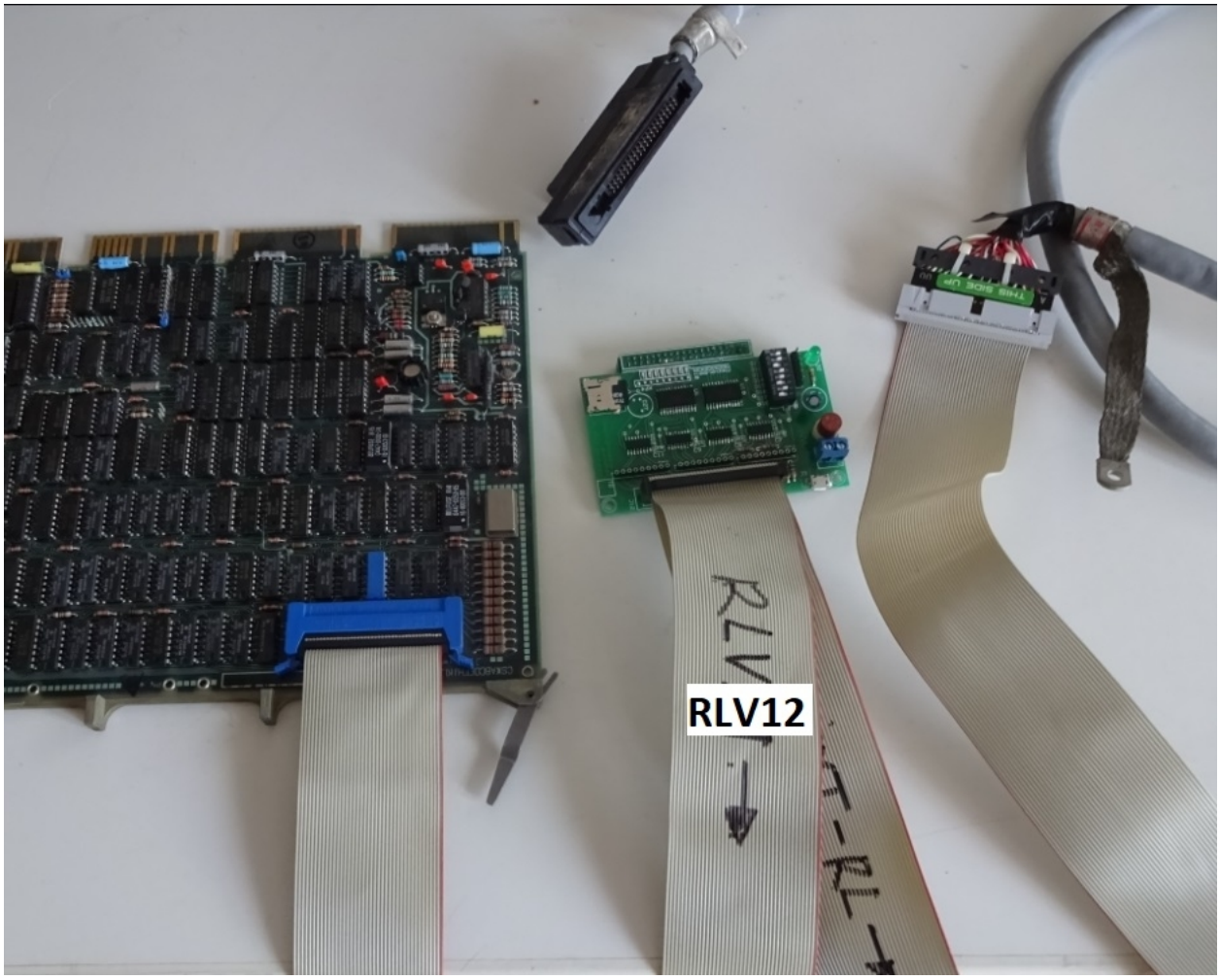
With the additional programs you can convert files: .DSK <==> .DEC
These programs are available at: www.2jo.de/pdp11/rlutils/rlutils.zip

3. Environment and Startup

Overview of the hardware and software setup including step-by-step procedures from installing the necessary software tools to use the DE10-Lite board.

This example shows a Q-BUS implementation with RLV12 controller

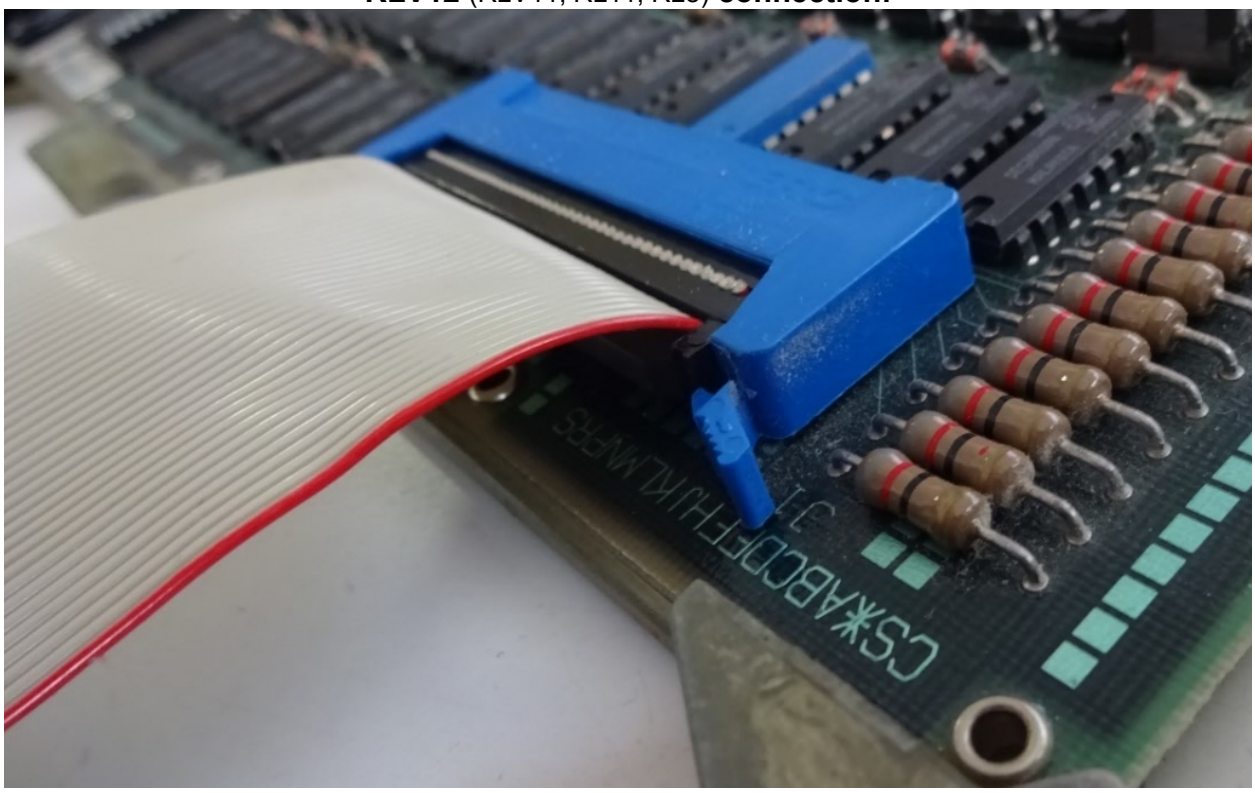
The following figure shows the connections based on a RLV12 Q-BUS controller-board to the emulator board and to an external RL disk drive.



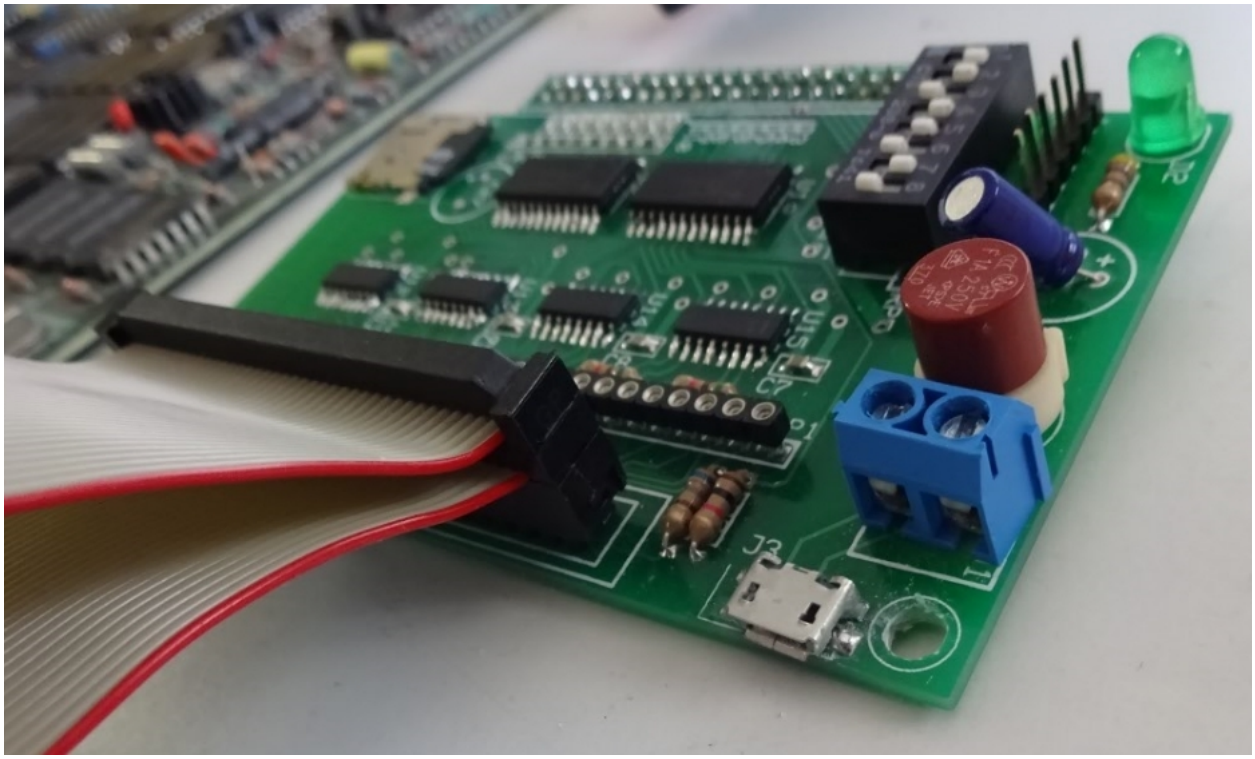
A 40-pin flat ribbon cable is required for the connections

The details of the connections are shown in the next 3 pictures

RLV12 (RLV11, RL11, RL8) **connection:**



Emulator board connection:

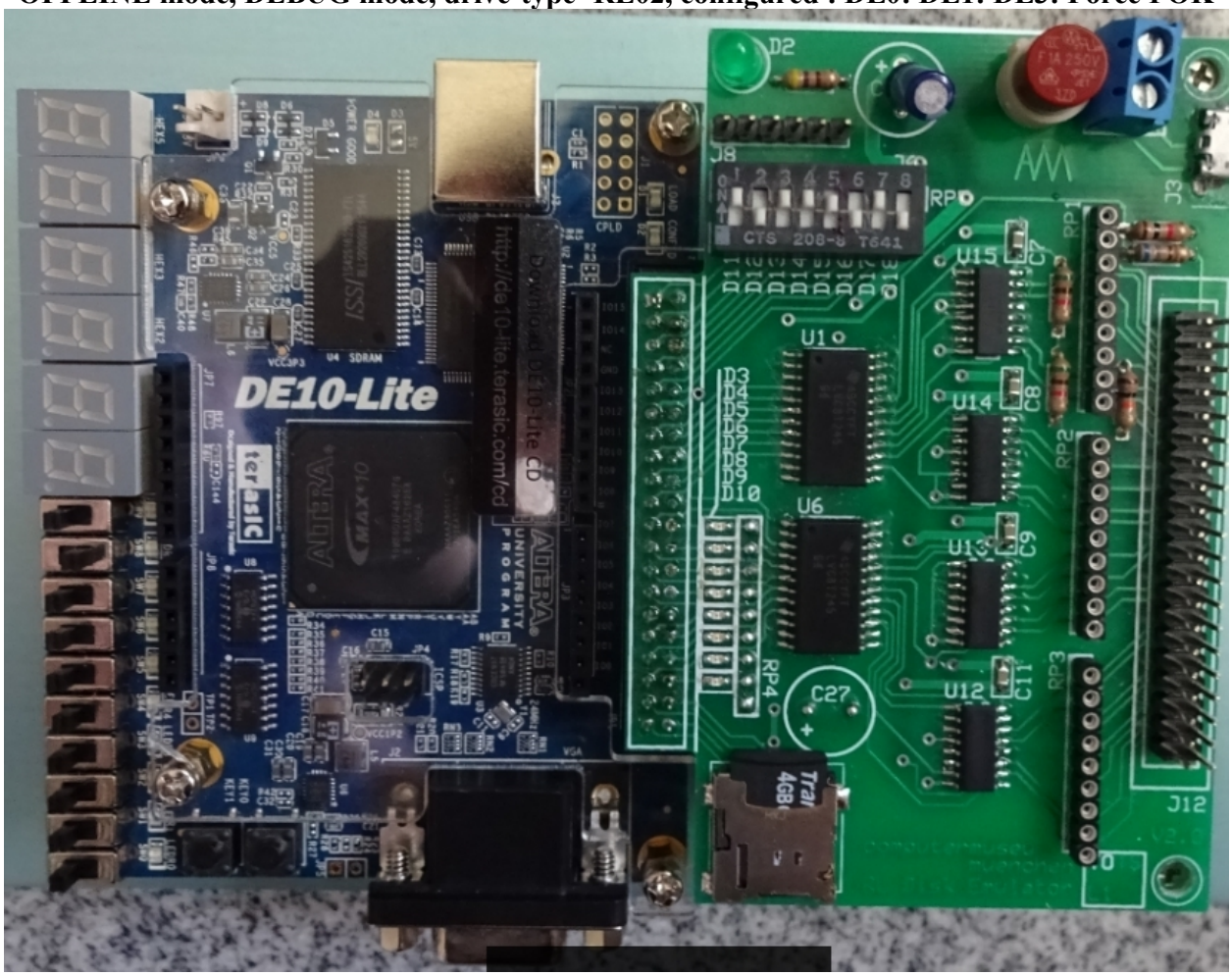


RL-BUS connection:

Disconnect if no external RL-drive is used , but install the 3 terminator resistor-networks on the emulator board.

Jumper settings for start-up/test example, NO external connection is required.

OFFLINE mode, DEBUG mode, drive-type=RL02, configured : DL0: DL1: DL3: Force POK



Jumper Settings details in the User Guide

Steps to bring up the emulator interface board up and running

Installing Quartus II Software Version 16.1

The Altera Complete Design Suite provides the necessary tools used for developing hardware and software solutions for Altera FPGAs. The Quartus II software is the primary FPGA development tool used to create reference designs along with the Nios II soft-core embedded processor integrated development environment.
User can download the latest software from https://www.altera.com/download/dnl-index.jsp
Use the Software Selector to get Quartus Version Prime 16.1 lite Edition downloaded and select MAX 10 FPGA support.

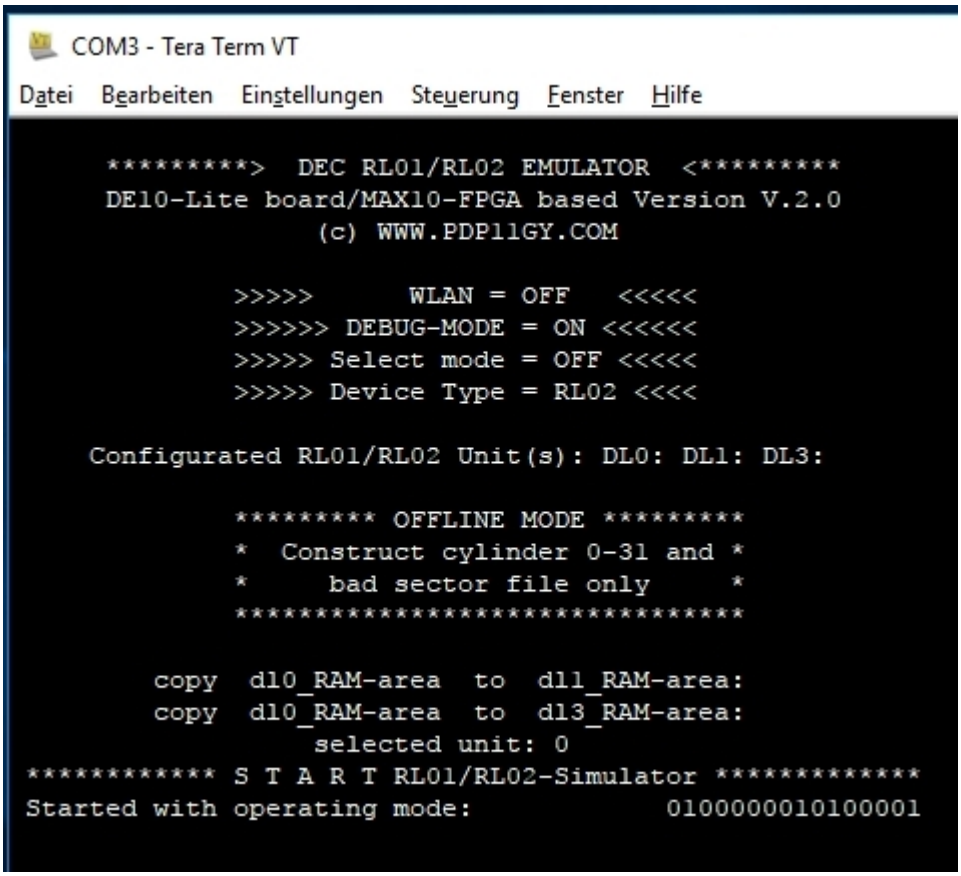
Now, Quartus can be started. Please follow the intructions in the flash folder if necessary.
If you want to make the design permanent, booting from the onchip_flash:

- **start the Quartus (Prime 16.1 lite Edition)**
- **Navigate to Tools/Programmer**
- **start Auto Detect and select 10M50DA**
- **right mouse click to "10M50DA" and select Change file**
- **select the .pof-file, in this case: flash/in_onchip_flash/MAX10_RL_emulator.pof**
- **Select the file @ Program/Configure and Press Start**

..... flashing .....

- **re-power the MAX10 DE10-Lite board.**

The system is ready for use and should start as follows:



Power up :

The heartbeat LED is blinking.
The 7-Segment displays HEX2 to HEX5 shows a circular cursor for the configured RL drives.

It takes about 10 seconds to start the NIOS processor and SD-RAM. When that's done,
the 8 LED's show a quick back and forth run which means that the NIOS II CPU has been started

As shown in the last picture, the system is now configured
Depending on Online or **Offline** mode, a different LED pattern is started

Offline Mode:

In this operating mode, no complete RL drives are emulated, access to the SD card is not possible and the emulator can be started without external connections, primary for verify purpose. **BUT,** if you connect the RL-Bus to the emulator board :
Access to an external "real" RL drives is possible (for test/verify purpose the external cable)
Limited access to cylinder 0-31 only is also possible. (about 0.3 MB)

Assuming RT-11 runs from another drive, such as RX01, RX02 or RX50, alternatively, my bootable RT-11 image files are available from my homepage, then the following commands can be used without problems (in this hardware example) :

dump/term dl0: (or dl2: , dl3:)
dump/term/only:23730 **// get the cartridge SN**
init dl0: (or dl2: , dl3:)
copy/sys *.* dl0: (or dl2: , dl3:) (**cancellation after 0.3 MB**)
dir dl1: (external , "real" RL02)

Online Mode:

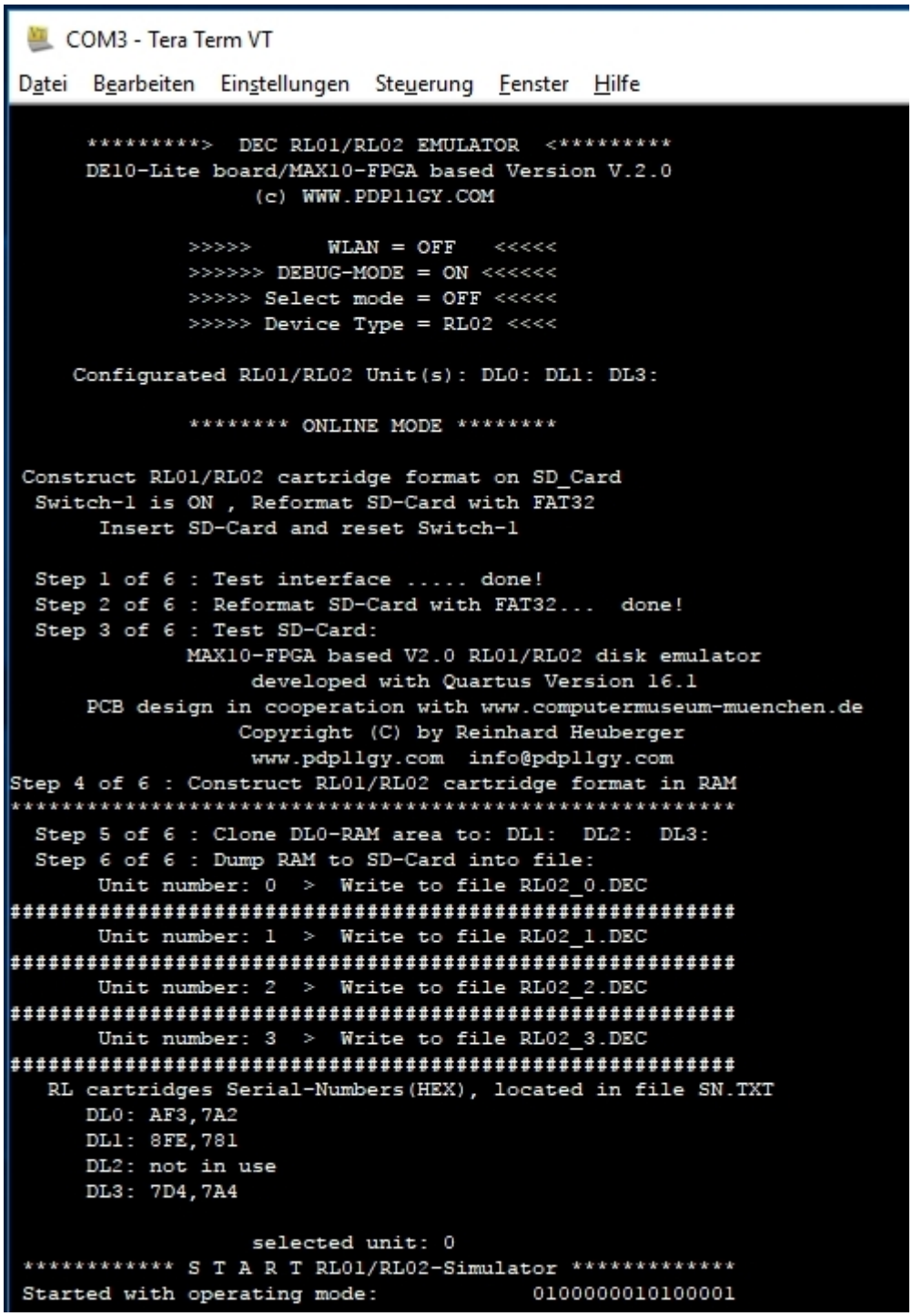
SWITCH 7 is responsible to select Online mode.

5. Example

Assuming, we have a real RL02 disk drive, unit 2 and we want to copy the data from the real RL02 to the emulated RL02 disk drives. First, we have to remove the terminator from the emulator board and cabling the real RL02 to be at the end of the RL-bus with connected RL-bus terminator. The real RL02 disk drive is configured as unit dl2 and the emulator board is configured for RL02 units dl0, dl1 and dl3 : SWITCH 4, 5, 7 = ON , **SWITCH 6 = OFF**.
(See also picture on page 11)

Note: SWITCH 6 off , the SD card will be **formatted**, images RL02_0.DEC to RL02_3.DEC will be generated.

The following messages appears on the screen :



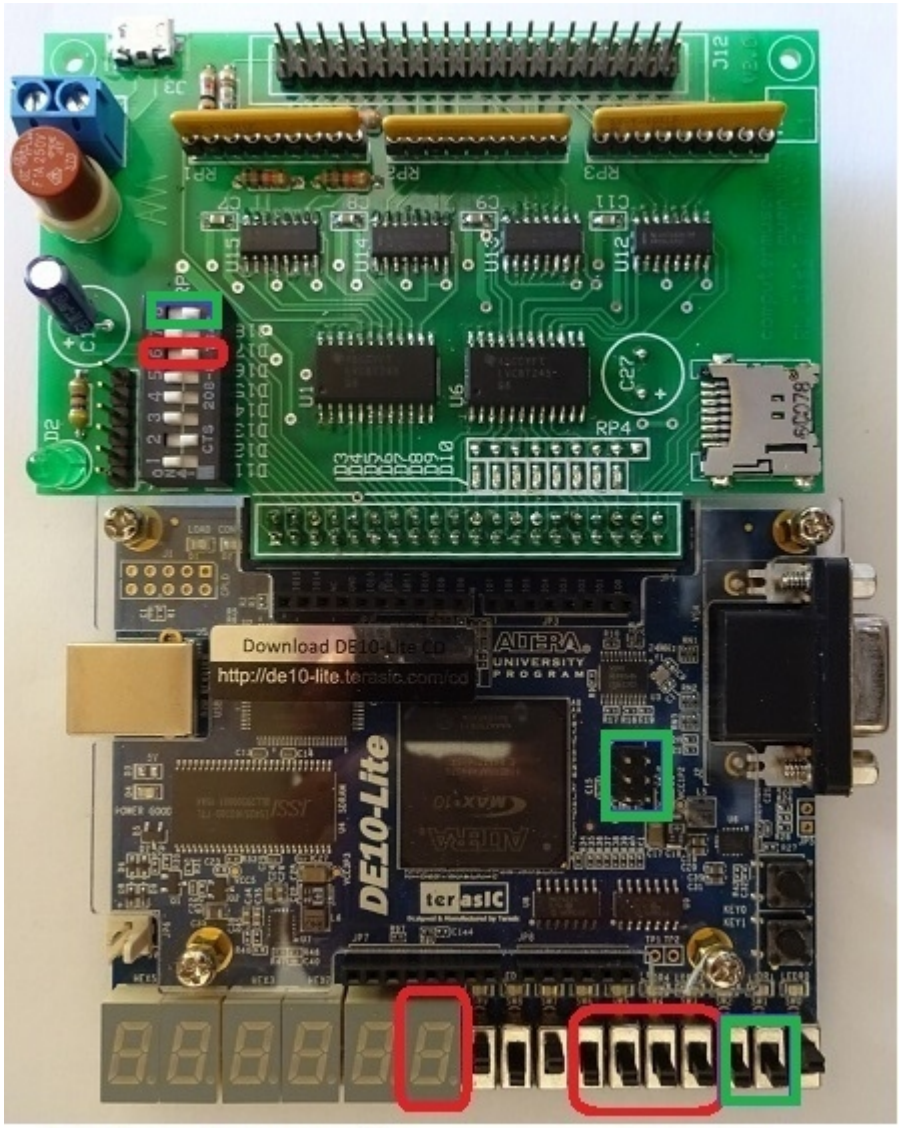
Now, we can copy the data from the real RL02 disk drive unit 2 to the emulated RL02 disk drives, for example (RT-11): copy/device dl2: dl0: (dl1: / dl3:)

Here comes a special feature:

- Switch down the real RL02 disk drive
- Set SWITCH 2 = ON (DL2)
- Press button 2 on DE10-Lite board and following message will appear:

From now on, 4 RL02 units will be emulated with full access to the dl2 unit.

6. SELECT and WLAN mode

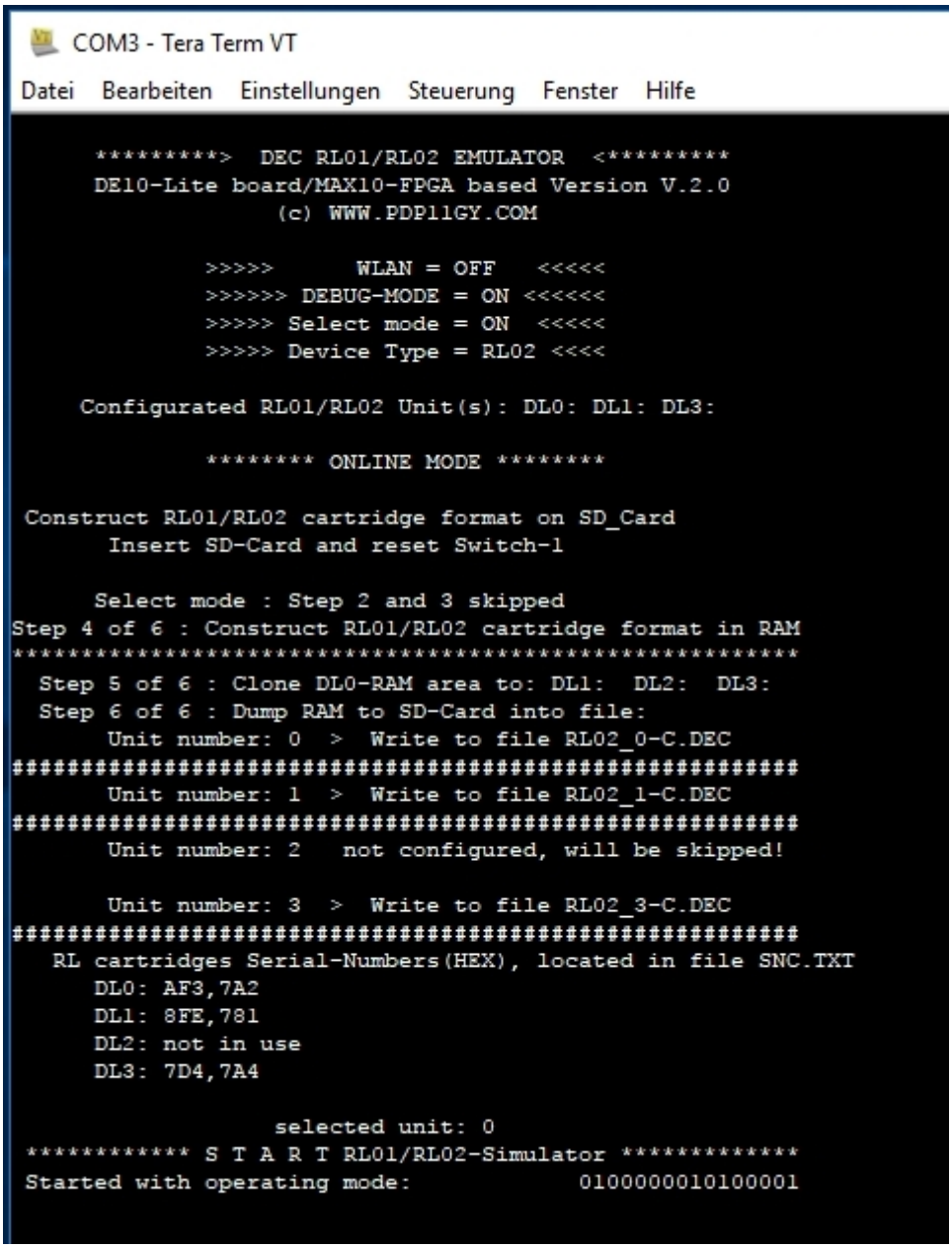


5.a SELECT mode , implemented with version V1.4 or higher

Without the SELECT mode, only one set consisting of 4 RL-images was supported. With the implementation of the Select mode, 16 additional sets, each consisting of a maximum of 4 RL-images are supported. This results in a total of 17 sets and means that a maximum of 68 RL-images are available and accessable in sets of 4 RL-images. **The SELECT mode is activated with SWITCH 6 on the interface board.**
Assuming that SW-3 = off, SW-4 = off, SW-5 = on and SW-6 = on, the hex-code C appears in the display 7seg-0. Now the RL-images **RL02_0-C**.DEC to **RL02_0-C**.DEC can be created.

With reference to the example on page 9 an initialization with SELECT mode is made **without** formatting the SD card: Step 2 and 3 are skipped.

6.2 SELECT mode example #1, generate disk-subset C:

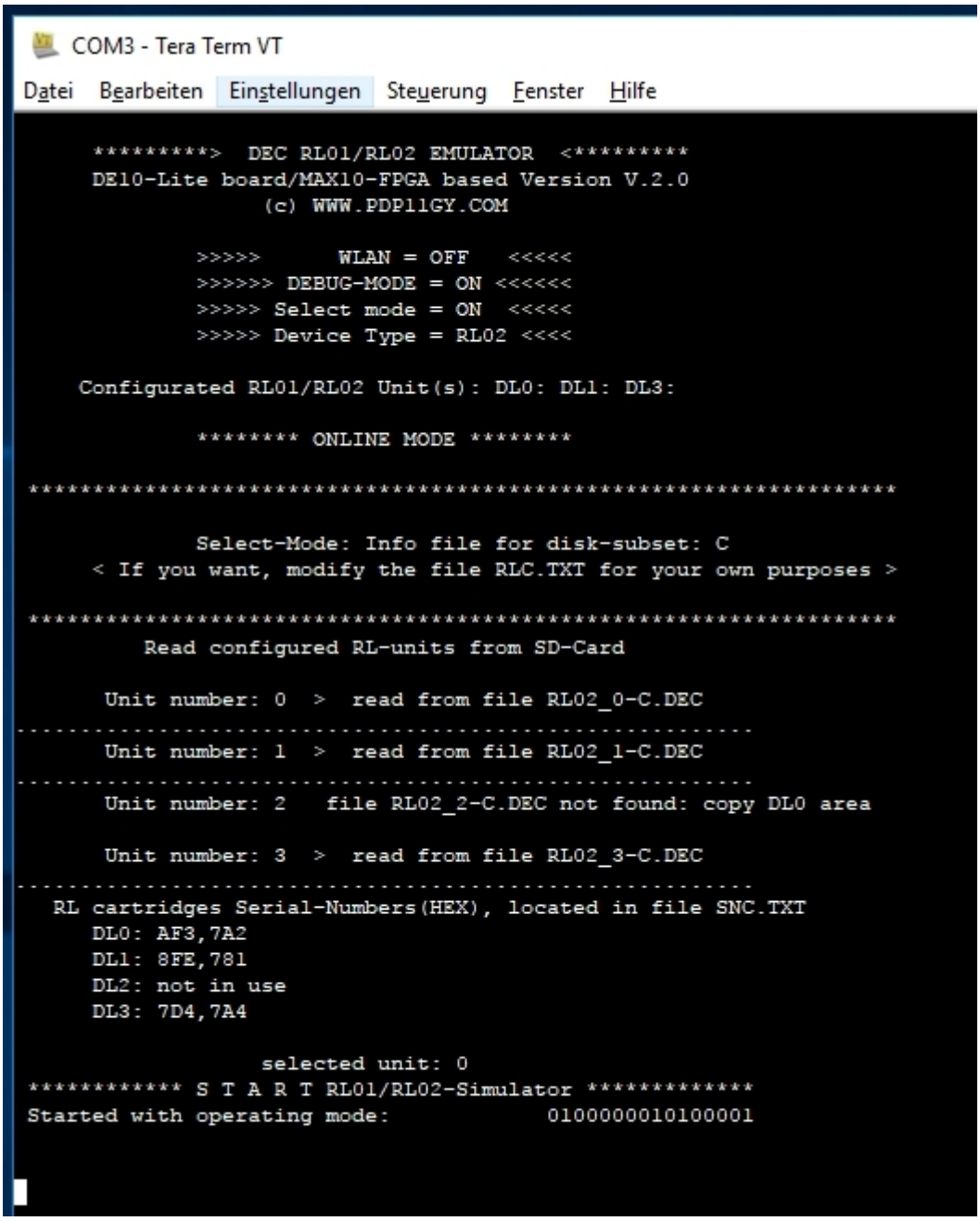


Note: If the SELECT mode is **not** switched on (SWITCH 6 off) the SD card will be **formatted** and the images RL02_0.DEC to RL02_3.DEC will be generated.

Summary: With this implementation you can create different disk images Sets of operating systems and/or data, selectable via different Set letters, 0 to F. For example, Set 0 is for RT11, set 1 for RSX etc….

To use a different disk Set, a restart is always required.

6.3 SELECT mode example #2, read and start **disk-subset C:**



New in Version 2.0 (or higher):

Customize the disk-subset environment: Feel free to modify the File RLC.TXT according to your own needs

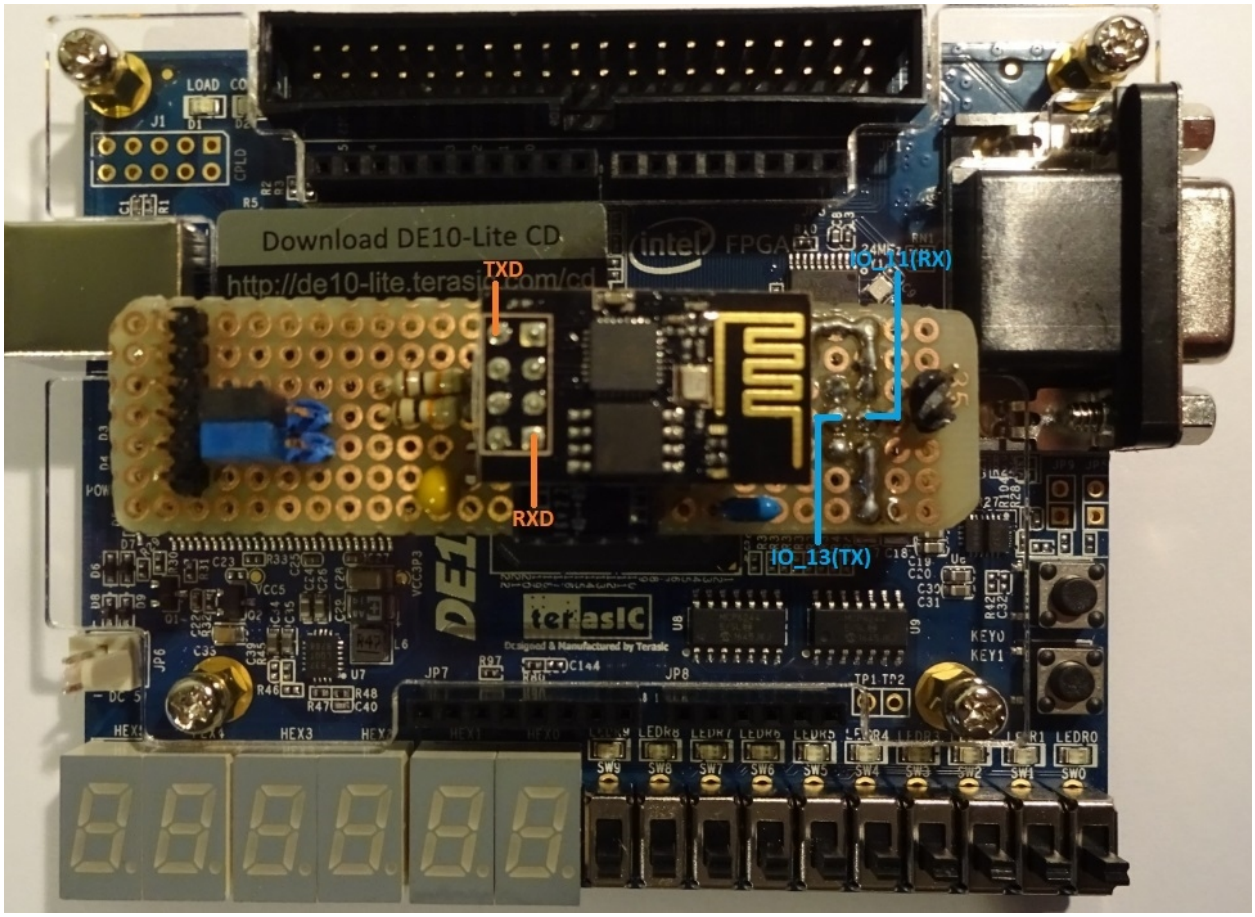
Emulated cartridge SERIAL NUMBER (SN) handling: If file SNC.TXT exist, the content will be used to set the emulated cartridge SERIAL NUMBER. Feel free to modify the File SNC.TXT to change the SERIAL NUMBER.

6.b WLAN mode, implemented with version V1.5 or higher

In general, the question arises whether it is useful for a disk emulator or not.
Does it make sense?
...to get the debug messages on your Handy / LapTop?
….to upload / download RL images to / from your Handy / LapTop ?

Version 2.0 or higher: The current implementation only supports the output of some debug messages. Further development is not planned.

Well, I've tried it and in principle it works. I have not implemented everything yet. Then I will consider how I continue. Perhaps there is a collaboration. Also, at the moment there is no PCB board available, only one prototype as shown in the following pictures.



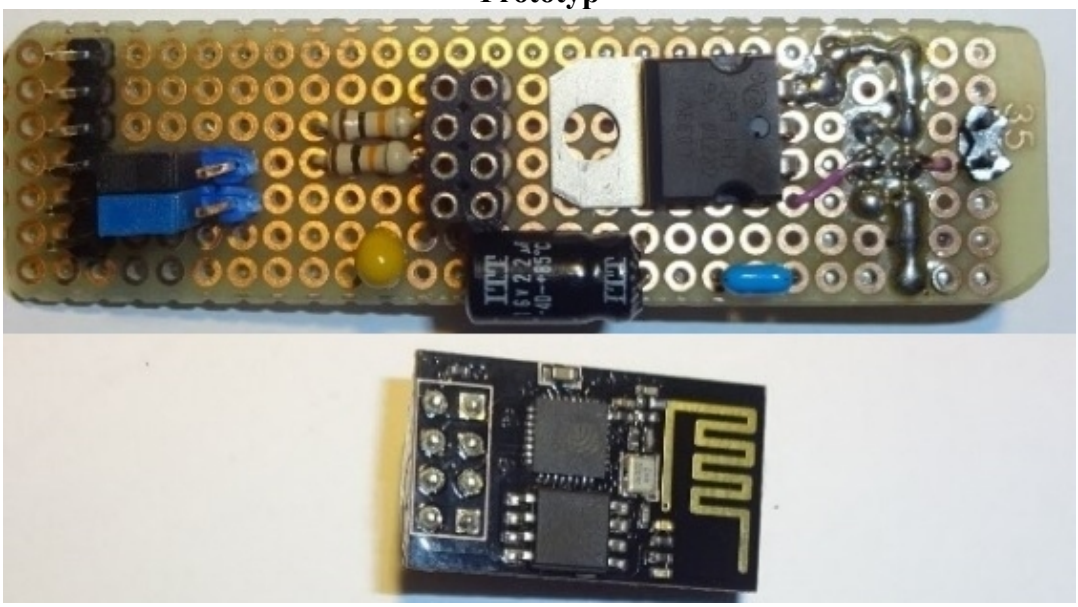
The WLAN implementation is based on the **ESP8266.**

It would not be much effort to design a PCB board which holds the ESP8266 as in the picture above. In addition, only one DC converter +5V to +3.3V, one tantalum capacitor and two resistors are required. With reference to the DE10-Lite_User_Manual, page **32** :
I had decided that I use the Arduino Connecter as interface to the ESP8266 as follows:

Arduino_IO_13 (TX) ----------> ESP8266 (RXD)
Arduino_IO_11 (RX) <---------- ESP8266 (TXD)

The serial connection runs at 115200 Baud

Prototyp

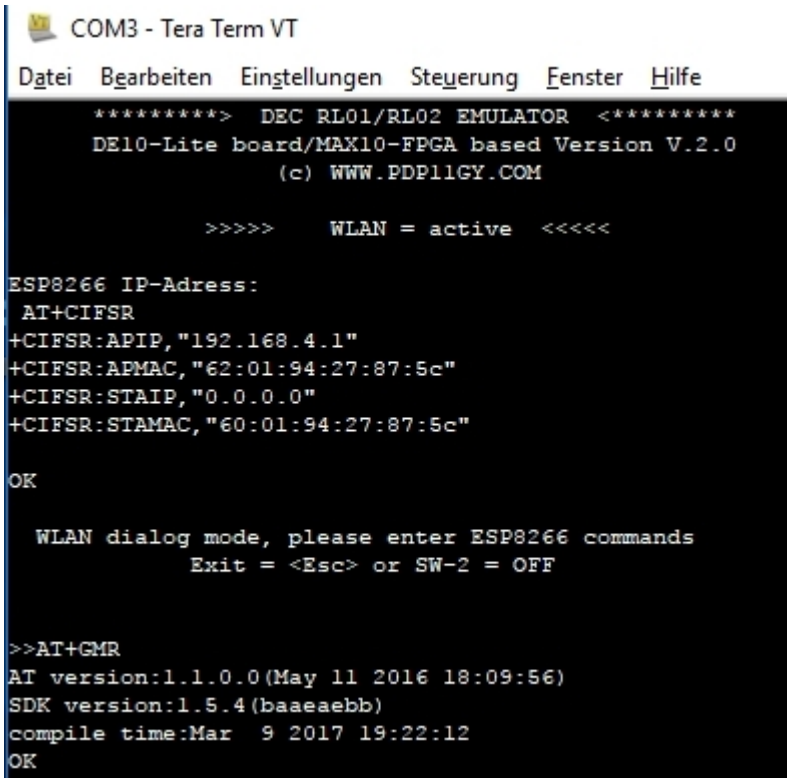


On the left is a 6-pin connector for direct serial connection and two jumpers. This may be necessary for debugging purposes or for an ESP8266 FW upgrade. On the right are 2 Pins for a HW reset. The two 10k resistors should be connected to + 3.3V and to the pins RST and CH_PD. Many further details and information, as well as programming, can be found on the Internet

Software:

Currently 2 WLAN modes are implemented. If WLAN is activated, the two switches SW-1 and SW2 will be checked after startup.
SW-1=OFF , SW-2=ON : WLAN dialog mode
SW-1=ON , SW-2=OFF : WLAN config mode via file ESP8266.CFG
The remaining 2 switch combinations are not implemented yet.

6.5 Example 1 , WLAN dialog mode:



In the WLAN dialog mode, the RL emulator responds with the >> .Now, it is possible to enter an AT-command and send it directly to the ESP8266. In this example 1, the AT+GMR command was used to receive the current FW version.

There are many ESP8266 programming examples available in the internet. To receive the debug messages from the RL emulator I would like to introduce the following 2 possibilities:

# Connection as TCP Client # Transparent transmission AT+CWMODE=3 // set WIFI mode # ssid and password from your router AT+CWJAP “ssid”,”password” AT+CIFSR // Query device IP # Use a network tool on the computer. For # example server IP = 192.168.178.63 using # port 8080 AT+CIPSTART=”TCP”,”192.168.178.63”, 8080 # Enable transparent transmission mode AT+CIPMODE=1 # Start send AT+CIPSEND # Response : > # From now on, data received from the RL # emulator will be transparent transmitted to # the Server.	# Configuration of softAP mode , acting as a # WiFi Access Point AT+CWMODE=3 // set WIFI mode AT+CWSAP=”RLEMULATOR”,”12345678”,7,3 # WLAN "RLEMULATOR" should now be # visible. Please connect to your PC / laptop or # mobile phone. The password is "12345678" AT+CIFSR // Query device IP #AT+CIPSTART="TCP","192.168.4.1",8080 #or AT+CIPSTART="UDP","192.168.4.1",8080,1112,2 AT+CIPMUX=1 AT+CIPSERVER=1,8080 AT+CIPSEND # # Now, open Browser: http://192.168.4.1:8080 #

If the dialog mode is left, the configuration is saved in the ESP8266. Restarting the emulator with SW-1 and SW-2 is OFF, the emulator starts the ESP8266 with the last configuration

6.6 Example 2 , WLAN config mode via file ESP8266.CFG

In this mode, the file ESP8266.CFG will be read from the micro SD card and the commands will be executed. This file can be modified with every editor and looks as follows:

```
# RL02/RL02 disk emulator with WLAN support receiving
#
# WLAN config file for ESP8266 working as access point.
# Used WLAN channel = 7  with WPA2 encryption ,= 3
#
AT+CWMODE=3
AT+CWSAP=”RLEMULATOR”,”12345678”,7,3
# To continue, press button 2
* User action required , press button 2
```

The following applies:

If there is a # sign at the beginning of a line, then it is a comment.
If there is a * at the beginning of a line, the execution is paused.
- All LEDs on the DE10-Lite board are now on.
- To continue, press button 2.

In principle, the basic things are up and running. I will not continue developing Wi-Fi support

For further questions, please contact me.
INFO@pdp11gy.com