## DEC RL01/RL02 DISK-DRIVE EMULATOR + Cloner + Reader + (Writer) User Manual for the DE10-Nano board <u>Version 2.8E</u>+



DE10-Nano board with interface SoC/HPS environment: Cyclone V FPGA + ARM Cortex-A9 CPU.

Emulates up to 4 RL01/RL02 drives simultaneously Supports mixed environment of emulated + real RL drives Access to 16 x 4 RL01/RL02 configurations sets Support .DSK data format Open FPGA-SoC-Linux environment SoC/HPS based disk emulator for the DEC RL01 and RL02 disk drives

## CONTENTS

Overview + Architecture	3
- Architecture	3
- Overview	4
Operating modes	5
Hardware	. 6
- Interface board	. 6
- Clone-mode, Power OK signal	. 7
- Reset/Config buttons	8
Environment and startup	9
Firmware and Software	12
- Ouick start	12
- Manual start	13
- Load FPGA	14
- Load/flash example	14
Software. rlemulator	16
- Select + init mode	16
- Example 1	18
- Example 2	19
Software, clonerl	23
- drive status + clone disk	24
- read one cylinder + seek_test	25
- Note: Write mode	25
- SIMH interface	23 26
- shift interface Annendiy Δ	20 27
Annendiv B	28
* PPVHULA DOMINION CONTROL CON	20

## **Overview & Architecture**

## Architecture: DE10-Nano based RL Emulator



WWW.PDP11GY.COM

# Secure the vintage software and preserve it on new technology

## Overview: 2021

Project Start was in 2009, details on my homepage <u>www.pdpl1gy.com</u>. The complete application is now ported to an open Linux SoC/HPS environment. In my case, it is the **DE10-Nano** board with Cyclon-V FPGA and 800MHz dual-core ARM Cortex-A9 processor. The performance increase is impressive. For example: Reading 4 RL images only takes about 5 seconds. With this project, much more is possible in addition to the RL-emulator and RL-cloner/reader/writer . It is possible to run the software **and** all SIMH emulators on a single board, in this case the **DE10-Nano** board. A PDP-11, PDP-8 and VAX emulator with all available RL-based software is running on this "one hand" large board. A Raspberry PI connected via network can be used for development purposes with the graphical interface. For example: The compiled programs like SIMH CPU-emulators can be copied it to the DE10-Nano board because it's binary compatible!

#### Architecture:

Basically, the design of my DEC RL02/RL01 disk drive emulator and reader 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 ARM Cortex-A9 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 CPU is 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. <u>https://www.youtube.com/watch?v=0i3ypBU39as</u>

## **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. Another disk format is the disk image structure .**DSK** which is used for CPU emulators.

This format is full supported and implemented inline. At write operation, the .DEC file and the .DSK file will be written. At read operation, first try is to read the .DEC file. If it does not exist, the .DSK file will be read. RL02: .DSK file = 10485760 byte .DEC file = 11796992 byte RL01: .DSK file = 5242880 byte .DEC file = 5898752 byte

The .DEC files are **compatible** to all my other RL-emulators.

## **Operating Modes**

**Emulator Mode** This mode supports a combination of real and emulated RL disk drives. Thus, it is possible to copy the data from a real disk drive into the emulated environment. However, there is still a computer system, e.g. a PDP-11 required. The interface is working in **Slave Mode** with the program **rlemulator** 

**Clone Mode** With this additional implementation, the need of a computer like a PDP-11 is no longer necessary. It is possible to read (and write) directly a RL disk and save it as a .DSK file. In addition, read errors will be corrected as good as possible before saving the data. The interface is working in **Master Mode** with the program **clonerl** 

Note: The clone/write mode is available, but I have never tested this mode on a real RL drive. In the end, it doesn't make any sense at all. It was just a personal challenge.

## For the hardware, this means that only the 40pin flat-cable has to be plugged into the RL interface board by 180. degrees as shown in the following overview:

FPGA+PINs		E	mulator-Mode	Clone-Mode		
-						
	AG25=13	B/ <mark>A</mark> =	01	Power_OK - +	39	
	AH26=14	D/C=	03	Drive-sel_0 - +	37	Drive-Error + -
	AH24=15	F/E=	05	Drive-sel_1 - +	35	Drive-ready + -
	AF25=16	J/H=	07	Write-gate - +	33	Sector + -
		L/K=	09		31	
	AG23=17	N/M=	11	System-Clk - +	29	Status-Clk + -
		R/P=	13		27	
	AF23=18	T/S=	15	Write-data - +	25	Status-IN + -
		V/U=	17		23	
	AG24=19	X/W=	19	Command - +	21	Read-data + -
	 АН22=20	7/Y=	21	Read-data - +	 19	Command + -
		BB/AA=	23		17	
	AH21=21	DD/CC=	25	Status-IN - +	15	Write-data +l-
		FF/EE=	27		13	
	AF22=25	, JJ/HH=	29	Status-Clk - +	11	System-Clk + -
		LL/KK=	31	·	09	
	AG20=27	NN/MM=	33	Sector - +	07	Write-gate + -
	AH19=32	RR/PP=	35	Drive-ready - +	05	Drive-sel 1 + -
	AH18=34	TT/SS=	37	Drive-Error - +	03	Drive-sel_0 + -
		<b>VV</b> /UU=	39		01	Power_OK + -
		1				-

However, there is one exception, the Power\_OK signal. Details and how to modify the interface board on Page 7

Datei: RL-disk-emuclo.odt, Autor: R.Heuberger www.pdp11gy.com RL-Emulator Manual Date: 02/22/22 Seite 5/28



## Hardware: DE10-Nano + Interface board

slide switches	0-3 : select one of 16 disk set:	0 to F
----------------	----------------------------------	--------

- Button 1 Reset / Restart after Reset
- **Button 2** Reconfigure / Exit after Reset

### SW-0 (Nr.8) - SW-7(Nr.1):

- SW-0(Nr.8) Initialize a new disk subset, selected by the slide switches
- **SW-1**(Nr.7) **Force power OK**
- SW-2(Nr.6) Debug mode ON/OFF
- SW-3(Nr.5) RL drive type, RL01 or RL02 (ON)
- **SW-7 SW-4 :** 4 disk units, **DL3: DL0:** will be selected and configured. All 4 switches OFF = OFFLINE mode.

#### **Interface LED's** (from right to left):

- **LED 0** heartbeat (blinking)
- LED 1 Power OK
- LED 2 Read/Seek in progress
- **LED 3** Write 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

#### **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 implementation.

#### Micro-SD

Is not used and is replaced by the onboard Micro-SD from the DE10-Nano board.

#### Clone-Mode: Power\_OK Signal:

This signal is present at PIN B as input in slave mode and as output at pin VV in master mode. However, the pin VV is connected to ground on the interface board.

**Todo:** Cut the ground etch to pin VV and connect to +5 VCC via a resistor about 300 Ohm:



### 2.2 Reset/Reconfig buttons

Unfortunately, the reset and reconfig buttons 1 and 2 on the DE10 Nano board are very small and difficult to reach. Now it is possible to control the reset/reconfig function alternatively via 2 external button. These buttons must be connected to the Arduino connector as follows:

Arduino\_IO13 = AH12 (Button 1) = reset/exit Arduino\_IO12 = AH11 (Button 2) = reconfig/restart See also DE10 User Guide 3.6.3 Arduino Uno R3 Expansion Header , page 30

#### Design example:



If you also want to control the 4 slide switches externally then follow the instruction in get\_started folder: rlv28e/INFOs/extern\_switches/README.txt

## **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 <u>example</u>, NO external connection is required.

OFFLINE mode, DEBUG mode, drive-type=RL02, configured device: none, Force POK



## **Firmware and Software**

## NEW(JAN-2022) : Quick start

### A ready for use micro SD-Card image is available from my homepage: www.pdp11gy.com, chapter RL01/RL02/status: pdp11gv.com/neu/diskemu-de10.zip Load down the file diskemu-de10.zip, unzip this file, copy the image file to a micro SD-Card using a program like the Win32DiskImager. Power on the DE10-Nano board, login: root, password: pdp11gy.com **Everything else is self-explanatory :** DEC RL01/RL02 disk emulator running on DE10-Nano board quick start commands: RL-emulator: rle, RL-reader: rlr, PDP11-simulator: simh manual start: Select folder rlv28e using command: cd rlv28e ./loadrbf (heartbeat LED must flash) -load firmware: -start rlemulator: ./rlemulator or -start RL-cloner/reader: ./clonerl -start SIMH PDP-11 simulator V3.9-0 : ./pdp11 \_\_\_\_\_ MFM disk emulator running on DE10-Nano bard quick start commands: mfmemulator: mfme, mfmreader: mfmr manual start: Select folder mfmv1 using command: cd mfmv1 -load firmware: ./loadrbf (heartbeat LED must flash) ./mfmemulator or -start mfmemulator: -start MFM-cloner/reader: ./readc www.pdp11gy.com E-Mail: info@pdp11gy.com

## Manual start

## Both programs, rlemulator and clonerl, need the same firmware

## We recommend to download and install the Unix kernel de10 nano Linux Console (kernel 4.5) V 1.3

Details in the terasic manual Getting Started Guide, de10-nano.terasic.com/cd

Download file rlv28e.zip from my homepage, <u>http://www.pdp11gy.com</u> or from GitHub: <u>https://github.com/pdp11gy/DEC-RL02-01-disk-emulator-reader-cloner-writer</u>. Unzip this file and follow the instructions in the README.txt file.

The firmware can be loaded in 3 different ways.

## 1): Load FPGA from Linux My recommendation! see next page

The following 2 additional options require the additional software : Quartus programmer. It is intended for developers rather than pure users.

## 2): Load .sof file (NOT permanent) : Required: Quartus Lite Version 16.1

- De0-Nano-SoC DIP switch (SW10) to default configuration, see page 12 @ User\_manual
- unzip the file "rlv28e.zip"
- Start Quartus Lite Version 16.1
- Make sure, your USB connection to the DE10-Nano is working.
- Follow the instruction in the DE10-Nano\_User\_manual at page 15 and load the **RL EMULATOR SoC.sof** file.
- After download, the heartbeat LED schould be blinking.

## 3) Permanent (EPCS): Required: Quartus Lite Version 16.1

- De0-Nano-SoC DIP switch (SW10) to EPCS configuration, see page 12 @ User\_manual
- unzip the file "rlv28e.zip"
- Start Quartus Lite Version 16.1
- Make sure, your USB connection to the DE10-Nano is working.
- Follow the instruction in the DE10-Nano\_User\_manual at page 112 and flash the DE10-Nano board with the fil **RL\_EMULATOR\_SoC.jic** from folder /flash.
- After re-powering the DE10-Nano board, the heartbeat LED schould be blinking.

## **Load FPGA from Linux:**

To load the firmware from Linux, another software is used, see <a href="https://github.com/nhasbun/de10nano\_fpga\_linux\_config">https://github.com/nhasbun/de10nano\_fpga\_linux\_config</a>

This software was taken over unchanged, only the Makefile was modified and the executable file is called loadrbf. As a pure user, I recommend this method because there is no additional software required like Quartus. Here are the steps to load the firmware and start the RL emulator/cloner:

- First, copy the file "rlv28e.zip" to the DE0-Nano-SoC board, for example, using scp or winscp. Unpack the zip file and navigate to folder clonerl.

unzip rlv28e cd rlv28e chmod 777 \* The loadrbf program is using the filename fpga\_config\_file.rbf but the RL emulator is using the file RL\_EMULATOR\_SoC.rbf. Use a link to get this issue fixed as follow:

## In -s ./flash/RL\_EMULATOR\_SoC.rbf fpga\_config\_file.rbf

## That's all !

Directory listing:

```
root@mfmemu:~/rlv28e# ls -l
total 9984
drwxrwxrwx 2 root root
                          4096 Mar 30 10:48 INFOs
-rwxrwxrwx 1 root root
                          5514 Mar 30 10:48 README.txt
-rwxrwxrwx 1 root root 9537116 Mar 30 10:48 Subset 0.zip
-rwxrwxrwx 1 root root
                         40425 Mar 30 10:48 clonerl
drwxrwxrwx 2 root root
                          4096 Mar 30 10:48 flash
                            27 Mar 30 10:49 fpga config file.rbf ->
lrwxrwxrwx 1 root root
./flash/RL_EMULATOR_SoC.rbf
-rwxrwxrwx 1 root root 13795 Mar 30 10:48 loadrbf
-rwxrwxrwx 1 root root 559820 Mar 30 10:48 pdp11
                         45835 Mar 30 10:48 rlemulator
-rwxrwxrwx 1 root root
root@mfmemu:~/rlv28e#
```

One additional folder is available with 4 bootable RL02 images using subset 0: DL0: XXDP22 CHMDLD0 XXDP+ DL MONITOR DL1: XXDP25 CHMDLD0 XXDP+ DL MONITOR DL2: bootable, RT-11 V05.04 C with Macro-11, BASIC, Fortran, FOCAL + Kermit DL3: bootable, RT-11 V05.04 C with Macro-11, BASIC, Fortran, FOCAL + Kermit

#### **Extract the files:**

root@mfmemu:~/rlv28e# unzip subsets.zip

root@mfmemu:~/rlv28e# mv subsets/\*.\* .

Now you have to start first the **A**) firmware loader and then the **B**) rlemulator or **C**) clonerl:

- A) root@socfpga::~/rlv28e# ./loadrbf
- B) root@socfpga::~/rlv28e# ./rlemulator
- C) root@socfpga::~/rlv28e# ./clonerl

## loadrbf program output:

```
MSEL Pin Config..... 0xa
FPGA State..... Powered Off
cfgwdth Register.... 0x1
cdratio Register.... 0x0
axicfgen Register... 0x0
Nconfig pull reg.... 0x0
CONF DONE..... 0x0
Ctrl.en?..... 0x0
Turning FPGA Off.
Setting cdratio with 0x3.
Turning FPGA On.
Loading rbf file.
EOF reached.
MSEL Pin Config..... 0xa
FPGA State..... User Phase
cfgwdth Register.... 0x1
cdratio Register.... 0x3
axicfgen Register... 0x0
Nconfig pull reg.... 0x0
CONF DONE..... 0x0
Ctrl.en?..... 0x0
```

## The heartbeat LED is blinking.

It takes about 10 seconds to start the Linux system. After starting the rlemulator, the 8 LED's show a quick back and forth run which means the rlemulator has been started and the communication between FPGA and HPS is working fine.

Depending on Online or Offline mode, a different LED pattern is started

## **Software: rlemulator**

#### **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 this version, the cartridge serial number can be set with the content of the file SNx.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 SNx.TXT is present, the serial number with the values from the file SNx.TXT will be always set after loading the RL images. If this is no longer necessary, then simply delete the file SNx.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

Please do not use a (hex) editor to change the SN. It would not build the new data CRC and would therefore cause problems, like boot/dup error.

#### Offline Mode: ( SW-4 - SW-7 is OFF)

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:** At leased one of the 4 SWITCHES SW-4 -SW-7 is ON: **Online mode is selected** 

## **SELECT + INIT mode**

With the implementation of the Select mode, 16 disk sets, each consisting of a maximum of 4 RL-images are supported. This results in a total of 16 sets and means that a maximum of 64 RL-images are available and accessable in sets of 4 RL-images. Of course you can extend this as you like because it is a Linux environment.

The SELECT + INIT mode is activated with SW-0 on the interface board. Please note: 4 files are always created for DL0: to DL3:

Assuming the **slide switches** are set to ON-OFF-OFF , disk set **8** will be used as in the following picture:

```
🚨 COM7 - Tera Term VT
Datei Bearbeiten Einstellungen Steuerung Fenster Hilfe
     ********> DEC RL01/RL02 EMULATOR <*********
      SoC/HPS DE10-Nano board based Version V.2.8E
                (c) WWW.PDP11GY.COM
            >>>>> Device Type = RL02 <<<<
            >>>>> DEBUG-MODE = ON <<<<<
            >>>>> Disk-subset: 8 <<<<<
    Configurated RL01/RL02 Unit(s): DL0: DL2: DL3:
            ******* ONLINE MODE *******
     Inizialize new disk set: 8
     To continue, set SW-0, (=Nr.8) to OFF position.
     SOC/HPC based V2.8E RL01/RL02 disk emulator
        developed with Quartus Version 16.1
       Copyright (C) by Reinhard Heuberger
        www.pdpllgy.com info@pdpllgy.com
Construct RL01/RL02 cartridge format in RAM
****************
   Clone DLO-RAM area to: DL1: DL2: DL3:
   Dump RAM to SD-Card into file:
   Unit number: 0 > Write to file RL02_0-8.DEC and RL02_0-8.DSK
   Unit number: 1 > Write to file RL02_1-8.DEC and RL02_1-8.DSK
   Unit number: 2 > Write to file RL02_2-8.DEC and RL02_2-8.DSK
                                                                   I
   Unit number: 3 > Write to file RL02_3-8.DEC and RL02_3-8.DSK
 RL cartridges Serial-Numbers(HEX), located in file SN8.TXT
    DL0: 0AF3.07A2
    DL1: not in use
    DL2: 08A2,077D
    DL3: 07D4,07A4
                 selected unit: 3
Started with operating mode: 0100 0000 1010 0001
```

## Example\_1:

Assuming, we have a real RL02 disk drive, unit 1 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 **dl1** and the emulator interface board is configured for RL02 units dl0, dl2 and dl3 : SWITCH 7, 6, 4 = ON, SWITCH 5 = OFF. Note: The file RL02\_3-8.DSK will be used instead of the .DEC file.

Starting the RL-emulator, the following messages appears on the screen:

```
********> DEC RL01/RL02 EMULATOR <*********
     SoC/HPS DE10-Nano board based Version V.2.8E
               (c) WWW.PDP11GY.COM
          >>>> Device Type = RL02 <<<<
          >>>>> DEBUG-MODE = ON <<<<<
          >>>>> Disk-subset: 8 <<<<<
   Configurated RL01/RL02 Unit(s): DL0: DL2: DL3:
          ******** ONLINE MODE *******
SOC/HPC based V2.8E RL01/RL02 disk emulator
               developed with Quartus Version 16.1
              (C) www.pdpllgy.com info@pdpllgy.com
                     info-file RL8.TXT
        <Edit the file RL8.TXT to change the info-message>
************
  Unit number: 0 > file RL02_0-8.DEC used
  Unit number: 1 = Not configured
  Unit number: 2 > file RL02_2-8.DEC used
  Unit number: 3 > file RL02_3-8.DEC not found, using file RL02_3-8.DSK
RL cartridges Serial-Numbers(HEX), located in file SN8.TXT
   DL0: 0AF3,07A2
    DL1: not in use
    DL2: 08A2,077D
   DL3: 07D4,07A4
               selected unit: 3
Started with operating mode: 0100 0000 1010 0001
```

Now, we can copy the data from the real RL02 disk drive unit 1 to the emulated RL02 disk drives, for example (RT-11): copy/device dl1: dl0: (dl2: / 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:

Reconfigurated RL01/RL02 Unit(s): DL0: DL1: DL2: DL3:

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

### Notes:

**Customize the disk-set environment:** Feel free to modify the File RL8.TXT according to your own needs.

**Emulated cartridge SERIAL NUMBER (SN) handling:** If file SN8.TXT exist, the content will be used to set the emulated cartridge SERIAL NUMBER. Feel free to modify the File SN8.TXT to change the SERIAL NUMBER.

# **<u>Example\_2</u>**: Convert .DEC file to .DSK file inline with rlemulator and start the PDP-11 emulator using SIMH.

A test RL02 image file is included in the folder socv2\_1/RL/RL02\_0-9.DEC . It's bootable : RT-11 V05.04 C with Macro-11, BASIC, Fortran, FOCAL + Kermit

```
Requirement: (see also page 13)
- set slide switches to disk-set hex 9
                                 1-0-0-1
- configure unit DL0: only, set SW to 0-0-0-1 (Nr. 5 = ON)
- copy the file rlv28e.zip to DE10-Nano board using scp.
- Extract the zip file:
 root@socfpga:~# unzip r1v28e.zip
// Steps:
root@socfpga:~# cd rlv28e
root@socfpga:~/rlv28e# ls
INFOs README.txt clonerl flash
                                     loadrbf pdp11 rlemulator
subsets.zip
root@socfpga:~/rlv28e# unzip subsets
root@socfpga:~/rlv28e# chmod 777 *
root@socfpga:~/rlv28e# mv subsets/*.* .
root@socfpga:~/rlv28e# in -s ./flash/RL EMULATOR SoC.rbf fpga config file.rbf
root@socfpga:~/rlv28e# ls -l
total 68304
drwxrwxrwx 3 root root
                            4096 Mar 30 18:05 INFOs
                             196 Mar 30 18:10 PDP11GY.INF
-rw-r--r-- 1 root root
-rwxrwxrwx 1 root root
                            5514 Mar 30 18:05 README.txt
                             367 Mar 30 18:10 RL0.TXT
-rw-r--r-- 1 root root
-rw-r--r-- 1 root root 11796992 Mar 30 18:10 RL02 0-0.DEC
-rw-r--r-- 1 root root 11796992 Mar 30 18:10 RL02 0-9.DEC
-rw-r--r-- 1 root root 11796992 Mar 30 18:10 RL02 1-0.DEC
-rw-r--r-- 1 root root 11796992 Mar 30 18:10 RL02 2-0.DEC
-rw-r--r-- 1 root root 11796992 Mar 30 18:10 RL02 3-0.DEC
```

-rw-r--r-- 1 root root 431 Mar 30 18:10 RL9.TXT -rw-r--r-- 1 root root 34 Mar 30 18:10 SN0.TXT -rw-r--r-- 1 root root 36 Mar 30 18:10 SN9.TXT -rwxrwxrwx 1 root root 40425 Mar 30 18:05 clonerl drwxrwxrwx 2 root root 4096 Mar 30 18:05 flash lrwxrwxrwx 1 root root 27 Mar 30 18:19 fpga config file.rbf -> ./flash/RL\_EMULATOR\_SoC.rbf -rwxrwxrwx 1 root root 13795 Mar 30 18:05 loadrbf -rwxrwxrwx 1 root root 559820 Mar 30 18:05 pdp11 45835 Mar 30 18:05 rlemulator -rwxrwxrwx 1 root root drwxrwxrwx 2 root root 4096 Mar 30 18:14 subsets -rwxrwxrwx 1 root root 10228403 Mar 30 18:05 subsets.zip // Note: the .DEC file always has 11796992 byte // load fimware root@socfpga:~/rlv28e# ./loadrbf // start the rlemulator root@socfpga:~/rlv28e# ./rlemulator \*\*\*\*\*\*\*> DEC RL01/RL02 EMULATOR <\*\*\*\*\*\*\*\* SoC/HPS DE10-Nano board based Version V.2.8E (c) WWW.PDP11GY.COM >>>> Device Type = RL02 <<<<</pre> >>>>> DEBUG-MODE = ON <<<<< >>>>> Disk-subset: 9 <<<<<<</pre> Configurated RL01/RL02 Unit(s): DL0: \*\*\*\*\*\*\* ONLINE MODE \*\*\*\*\*\*\* SOC/HPC based V2.2 RL01/RL02 disk emulator developed with Quartus Version 16.1 PCB design in cooperation with www.GfhR.de (C) www.pdp11gy.com info@pdp11gy.com info-file RL9.TXT <Edit the file RL9.TXT to change the info-message> DL0: bootable, RT-11 V05.04 C with Macro-11, BASIC, Fortran, FOCAL + Kermit DL1: not configured DL2: not configured not configured DL3: +.....+ Unit number: 0 > file RL02 0-9.DEC used +....+

Datei: RL-disk-emuclo .odt, Autor: R.Heuberger www.pdp11gy.com RL-Emulator Manual Date: 02/22/22 Seite 20/28

```
User Manual RL01/02 DISK-Emulator/Reader, SoC/HPS - DE10-Nano
   Unit number: 1 = Not configured
   Unit number: 2 = Not configured
   Unit number: 3 = Not configured
 RL cartridges Serial-Numbers(HEX), located in file SN9.TXT
     DL0: 0AF3,07A2
     DL1: not in use
     DL2: not in use
     DL3: not in use
                selected unit: 0
Started with operating mode: 0100000010100001
//******* Press the RESET Button 1 or force a power fail ********
..... Shutting down system .....
+-----
    Unit number: 0 > Write to file RL02_0-9.DEC and RL02_0-9.DSK
+-----
   Unit number: 1 not configured, will be skipped
   Unit number: 2 not configured, will be skipped
   Unit number: 3 not configured, will be skipped
 Press RESET/Button-1 for exit, Reconfig/Button-2 for restart
// the .DSK file is now available, always 10485760 byte
root@socfpga:~/socv2 1/RL# ls -l RL02_0-9.*
-rw-r--r-- 1 root root 11796992 Mar 30 18:32 RL02 0-9.DEC
-rw-r--r-- 1 root root 10485760 Mar 30 18:32 RL02_0-9.DSK
// Start the PDP-11 simulator
root@socfpga:~/socv2_1/RL# ./pdp11
PDP-11 simulator V3.9-0
sim> set CPU 11/23 512k
Disabling CR
Disabling RK
Disabling HK
Disabling TM
sim> attach RL0 ./RL02_0-9.DSK
sim> boot RL0
```

RT-11SJ V05.04 C

.SET USR NOSWAP

.SET TT SCOPE

.SET EDIT KED

.INIT/NOQUE VM:

.dir

DL0DL0.INF	59		SWAP	.SYS	27P	02-Sep-87
RT11SJ.SYS	79P	15-Jan-88	DD	.SYS	5P	02-Sep-87
DY .SYS	4P	02-Sep-87	LS	.SYS	5P	02-Sep-87
SL .SYS	17P	02-Sep-87	TT	.SYS	2P	02-Sep-87
VM .SYS	3P	02-Sep-87	DU	.SYS	8P	02-Sep-87
LD .SYS	8P	02-Sep-87	DL	.SYS	5	17-0ct-84
STARTS.COM	1	28-Mar-99	DIR	.SAV	19	02-Sep-87
PIP .SAV	30	02-Sep-87	DUP	.SAV	49	02-Sep-87
RESORC.SAV	25	02-Sep-87	KED	.SAV	58	02-Sep-87
UCL .SAV	16	02-Sep-87	CREF	.SAV	6	02-Sep-87
SRCCOM.SAV	26	02-Sep-87	BASIC	.SAV	53	04-Apr-83
MACRO .SAV	61	02-Sep-87	DUMP	.SAV	9	02-Sep-87
MKDL0 .BAS	1		MKDL2	.BAS	1	
MKDL3 .BAS	1		MKDL1	.BAS	1	
SYSLIB.OBJ	216	24-Mar-87	FORTRA	A.SAV	206	24-Mar-87
KERMIT.INI	1	28-Jul-87	KERMIT	SAV.	182	02-Apr-86
KERMIT.HLP	148	13-Apr-86	FOCAL	.SAV	36	30-Nov-84
FOCALD.SAV	38	30-Nov-84				
35 Files,	1406 BI	locks				

18976 Free blocks

// Let's start BASIC

.r basic

BASIC-11/RT-11 V2.1 OPTIONAL FUNCTIONS (ALL, NONE, OR INDIVIDUAL)? ALL

READY

// If you want, try to run this small program:

10 FOR E=1 TO 30 STEP .1 20 Y=INT((SIN(E)\*20)+30) 40 PRINT TAB(Y);"HELLO-1980" 100 NEXT E

## Software: clonerl

**Background**: I developed this program for the purpose if it is necessary to save the contents of an RL cartridge without a DEC system such as a PDP-11.

The **clonerl** program runs with the same firmware. How to load this firmware is described in detail on pages 12 to 14. Before you start the **clonerl** program, make sure that the RL-bus cable is plugged in correctly, i.e. reversed compared when using the rlemulator program. See also page 5. In the appendix, the topic of cable length, termination and grounding is addressed and discussed.

Jumper setting, see page 6. SW-7 – SW-4 : Select the available real RL01/RL02 disk drives. If more than one disk drive is configured then the disk drive with the lowest unit number is used. Load the fimware with ./loadrbf and start the program: ./clonerl

The clonerl-program first checks whether the drive is ready. Then the program tries to read the drive status. If these 2 points are met, the RL drive is positioned on track 0 and the following startup message appears:

\*\*\*\*\*\*> DEC RL01/RL02 Cloner/Reader <\*\*\*\*\*\*\* SoC/HPS DE10-Nano board V.2.8E (c) WWW.PDP11GY.COM >>>> Device Type = RL02 <<<<</pre> >>>>> DEBUG-MODE = ON <<<<< \*\*\*\* Cloning disk-drive: DL0: \*\*\*\*\*\*\* preset memory \*\*\*\*\* Started with operating mode: 1100 0000 1010 0001 Drive READY signal = TRUE Reset/Init drive Request drive status Drive status: 29D = 0000 0010 1001 1101 Drive at cylinder position: 0 Drive positioned to cylinder position : 0 \*\*\*\*\* Select Mode \*\*\*\*\* 0=exit, 1=get status, 2=clone disk, 3=read one Cylinder, 4=seek-test, 5=write :

If the RL drive does not respond, the program loops and prints a "." every second until the RL drive answers by sending the drive status.

1) get drive status. Example:

get drive status: 29D = 0000 0010 1001 1101 Load cartridge state Spin up Brush cycle Load Heads Seek-Track Counting > Seek-Linear Mode (Lock ON) Unload Heads Spin down > Brush Home (BH) > Heads Out (HO) Cover OPEN (CO) Head Selected (HS) > -reserved-Drive Select Error (DSE) > Volume Check (VC) Write Gate Error (WGE) Spin Error (SPE) Seek Time Out (SKTO) Drive Write Locked Head Current Error (HCE) Write Data Error (WDE)

2) clone disk. Example

**0**=exit, **1**=get status, **2**=clone disk, **3**=read one Cylinder, **4**=seek-test, 5=write :2 Data will be saved in file: RL02\_0-clone.dsk set\_track-0: 0000 0000 0000 0101 cylinder: 0 RAM\_Address: 0 set\_track-1: 0000 0000 0001 0101 set track-0: 0000 0000 1000 0101 cylinder: 1 RAM Address: 11520 set track-1: 0000 0000 0001 0101 set track-0: 0000 0000 1000 0101 cylinder: 2 RAM Address: 23040 set\_track-1: 0000 0000 0001 0101 set\_track-0: 0000 0000 1000 0101 cylinder: 3 RAM\_Address: 34560 set\_track-1: 0000 0000 0001 0101 set\_track-0: 0000 0000 1000 0101 cylinder: 4 RAM\_Address: 46080 set track-1: 0000 0000 0001 0101 set\_track-0: 0000 0000 1000 0101 cylinder: 509 RAM\_Address: 5863680 set\_track-1: 0000 0000 0001 0101 set track-0: 0000 0000 1000 0101 cylinder: 510 RAM Address: 5875200 set track-1: 0000 0000 0001 0101 set track-0: 0000 0000 1000 0101 cylinder: 511 RAM Address: 5886720 set track-1: 0000 0000 0001 0101 Saved data in file: RL02 0-clone.dsk \*\*\*\*\* Select Mode \*\*\*\*\* 0=exit, 1=get status, 2=clone disk, 3=read one Cylinder, 4=seek-test, 5=write :

<sup>3)</sup> read one Cylinder. Example:

Drive command received: 0100 1011 0000 0101 Cylinder difference : 150 Using head 0 to: Cylinder-Nr. 150 = RAM-address: 1728000 Drive command received: 0000 0000 0001 0101 Cylinder difference : 0 Using head 1 to: Cylinder-Nr. 150 = RAM-address: 1733760 Drive command received: 0100 1011 0000 0001 Cylinder difference : 150 Using head 0 to: Cylinder-Nr. 0 = RAM-address: 0

4) seek-test. Example:

0=exit, 1=get status, 2=clone disk, 3=read one Cylinder, 4=seek-test, 5=write:4
Seek to cylinder (max = 511) : 511
Number of loops: 3
1time: seek to cylinder 511 .....back to cylinder 0
2time: seek to cylinder 511 .....back to cylinder 0
3time: seek to cylinder 511 .....back to cylinder 0

5) write Note :

If this write mode is selected, the following menu will be available: 1=write pattern, 2=write bad sector, 3=write .dsk-file

But !!

Implementing this write mode was a purely personal challenge. It does not make any sense to write on a real old RL disk drive. The idea behind it was to get the possibility to control an original RL-drive **directly** from the SIMH project and/or from a PiDP-11 running SIMH on it. Unfortunately, no cooperation with other people was possible and I no longer want to develop the product further with the write option. I also no longer have the option of accessing an original RL drive to test the software. Note, this part of the software was never tested on a real RL drive !! The program code is available on GitHub. The whole code is written in verilog. Maybe someone else has the time and motivation to expand and verify this option. ( I am now out of motivation and also "too old" to continue here ) Example: clonerl and SIMH interface root@socfpga:~/clonerl# ./pdp11 PDP-11 simulator V3.9-0 sim> set CPU 11/23 512k Disabling CR Disabling RK Disabling HK Disabling TM sim> attach RL0 ./RL02\_0-clone.dsk sim> boot RL0 RT-11SJ V05.04 C .SET USR NOSWAP .SET TT SCOPE .SET EDIT KED .INIT/NOQUE VM:

> For comments and questions, please contact me. INFO@pdp11gy.com

References: <u>http://www.pdp11gy.com</u> <u>https://github.com/pdp11gy/DEC-RL02-01-disk-emulator-reader-cloner-writer</u>

https://github.com/pdp11gy/SoC-HPS-based-RL-disk-emulator https://github.com/pdp11gy/DEC-RL02-RL01-disk-emulator https://github.com/pdp11gy/SoC-HPS-based-MFM-disk-emulator

## Appendix A

It was difficult to make everything runable because many things in the documentation and in the examples were not correct. Here is a step by step explamation to rebuild the RL-emulator if necessary or if you want to design some add-on application.

\*1 : error You must define soc\_cv\_av or soc\_a10 before compiling with HwLibs Go to intelFPGA/16.1/embedded/ip/altera/hps/altera\_hps/hwlib/include Copy all .h files in the folder soc cv av and soc a10

\*2 : generate\_hps\_qsys\_header.sh : PATH is not set correct: correct as following: #!/bin/sh PATH=/cygdrive/C/altera\_lite/16.1/quartus/sopc\_builder/bin:\$PATH sopc-create-header-files \ "\$PWD/RL\_system.sopcinfo" \ --single hps\_0.h \ --module hps\_0

\*3: Modify the Makefiles, here the RL-emulator make file software/RL\_emulator/Makefile

```
#
TARGET = rlemulator
ALT_DEVICE_FAMILY ?= soc_cv_av
ALT_DEVICE_FAMILY ?= soc_al0
#
CROSS_COMPILE = arm-linux-gnueabihf-
#CFLAGS = -static -g -Wall -I$
{SOCEDS_DEST_ROOT}/ip/altera/hps/altera_hps/hwlib/include
CFLAGS = -g -Wall -I${SOCEDS_DEST_ROOT}/ip/altera/hps/altera_hps/hwlib/include/$
{ALT_DEVICE_FAMILY} -Dsoc_cv_av -Dsoc_al0
LDFLAGS = -g -Wall
CC = $(CROSS_COMPILE)gcc
ARCH= arm
```

```
build: $(TARGET)
$(TARGET): main.o
    $(CC) $(LDFLAGS) $^ -o $@
%.o : %.c
    $(CC) $(CFLAGS) -c $< -o $@</pre>
```

.PHONY: clean clean: rm -f \$(TARGET) \*.a \*.o \*~

## **Appendix B**

### Notes concerning hardware, cable length and RL-Bus termination

### **RL-Bus chips:**

The RL disk drive and the controller used the Line Receiver chips SN75107A and Line Drivers SN75113. These chips are out of date (from 1980) and no longer available. In addition, the receiver chip SN75107A requires a negative voltage of -5 volts and is therefore difficult to handle. I replaced these chips as follows:

Line Receiver SN75107A  $\rightarrow$  AM26LS32ACDR

Line Drivers SN75113  $\rightarrow$  AM26LS31CDR

It's also necessary to implement a level converter and I decided to use the chip SN74LVC8T245DWR. At least, these new chips are available in SMD technology.

## cable length:

The RL-bus works with a frequency of 4.1 Mhz. For proper operation, the cable length should be calculated according to the **lambda**/wavelength rule. This results in a cable length of lambda 1/256 = 29.25 cm. Put simply, the cable should be a multiple of about ~29cm. However, I am not a high frequency specialist and would be grateful for any advice.

## **Termination :**

In general, the new chips are less critical in terms of termination, but an RL disk drive or RL disk controller has still the old driver and receiver chips. The termination is not critical in emulator mode. As described on page 9, the termination on the RL drive is completely sufficient with a 2m long ribbon cable. Termination is more critical in clone/ read mode. In addition, the receiver chips get relatively hot when cloning a complete RL disk. I had the best results with an approx.1m flat cable directly connected to the RLV12 controller with 330 OHM termination on the interface board.

## Grounding

Very, very important. Please make sure that ground with the same potential is available on all involved components. Line filters are built into an RL drive or in a computer like PDP-11. If the grounding is bad, there are reflections which result in data transfer problems.

Finally: The LOM data for the interface are available on my homepage.

Also: I had the idea to merge both interfaces together, i.e. the RL01/RL02-disk emulator and the MFM-disk emulator. Everything is already prepared, but I'm not a CAD specialist. In addition, there is hardly any demand. A cost estimate from an external company was too expensive for me. If anyone is interested in continuing, please contact me.

Info, MFM disk emulator: https://github.com/pdp11gy/SoC-HPS-based-MFM-disk-emulator

## All firmware & software is open source