Embedded Linux Combining the DE1-SoC Board with the DCDCbuck Board

Prof. Dr. Martin J. W. Schubert, Electronics Laboratory,

OTH Regensburg, Regensburg, Germany

Embedded Linux Combining the DE1-SoC Board with the DCDCbuck Board

Abstract. Employing some hardware/software codesign, today called "embedded".

1 Introduction

1.1 Goals

Goal is the Employment of some hardware/software codesign (embedded) to illustrate how it works and gain new possibilities.

1.2 Acknowledgements

The author would like to thank Alexander Forster [11] for implementing *Linux* into this practical training.

1.3 Outline

The organization of this communication is as follows:
Section 1: Introduction
Section 2: Using the *ARM* Core Embedded on the *FPGA*
Section 3: Preparations for the Embedded Exercises.
Section 4: Conclusion.
Section 5: References

2 Using the *ARM* Core Embedded on the *FPGA*

The *Cyclone V FPGA on the DE1-SoC* board *contains an 800MHz dual-core AMR Cortex-A9 MPCore* processor [35]. This chapter demonstrates how to use it for data transfer between the *FPGA* and a *PC*.

Acronyms:

ARM Advanced RISC Machine, a computer architecture family [27].
AXI Advance eXtensible Interface [28]. On-chip bus protocol developed by ARM.
HPS Hard Processor System [29]

Start the *Linux* [30] Server:

- Start *DE1-SoC* board
- Connect USB cable to Computer and *mini- USB B* [31] plug at the upper right corner of the *DE1-SoC* board
- Determine COM-Port number: Gerätemanager → Anschlüsse (COM & LPT) → USB serial Port (COM#)
- Start *PuTTY* [32]: set: serial, COM#, baud rate: 115200 [save session: hps], as shown in Fig. 2.
- Insert *microSD* [33] card into the SD card slot near the *USB mini-B* plug as shown in Fig. 2.2(c).
- Boot *Linux* system on *microSD* card by switching the *DE1-SoC* board's power OFF → ON or press the left or medium of the 3 little push buttons, which are located on the right hand side of the big push buttons.
- Program the *Cyclone-V FPGA* with the *Quartus* programmer
- **System prompt:** user input
  `socfpga login: root`
  <*ci_de1soc_DCDCbuck* software must run from here on the *DE1-SoC* board, on it set *sw*(9:0)="0…0">
  `root@socfpga:~# ls` // list directory
  `root@socfpga:~# ./hex_timer 5` // run hex timer for 5 seconds, observe 7-segment display!
  `root@socfpga:~# ./monitor 10` // read 10 data lines from *HPS* and print them
  <make *PuTTY* window COM# wider using the mouse to avoid undesired line feeds, as shown in Fig. 2.3.>
  `root@socfpga:~# ./monitor 10` // print again with matched window
  <On the *DE1-SoC* board set *sw*(9:6)="1111" to get *HPS* control over input *w*, the wanted output voltage>
  `root@socfpga:~# ./set_w 1234` // wanted input (→ sw(2)='1') is set to 1234
  `root@socfpga:~# ./monitor > target_file.log` // writes output to file *target_file.log*.
  `root@socfpga:~# cat target_file.log` // print the contents of *target_file.log*.
  `root@socfpga:~# rm target_file.log` // remove *target_file.log*.

Fig 2.1 illustrates how to figure out *COM#* of your serial port. In this figure it is *COM3*.

Hold your mouse on the *Windows* sign (), click right mouse button → *Device Manager* (dt.: *Geräte-Manager*) to get into this menu.

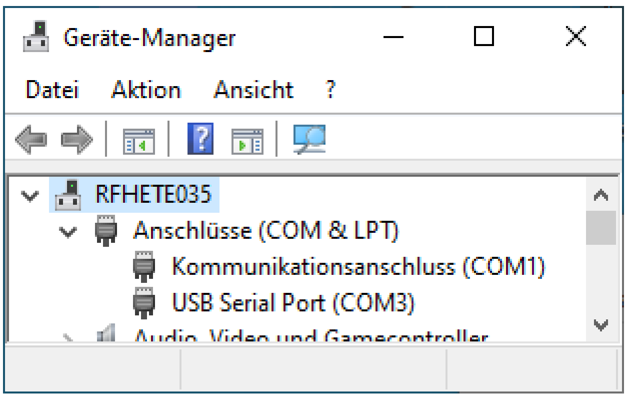


Fig. 2.1: Determine the COM port number of your serial connection via the *mini-USB B* plug.

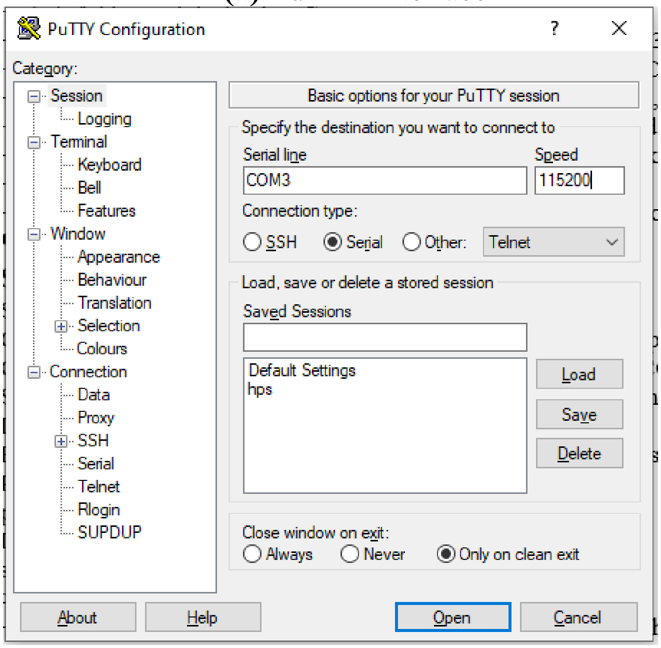
Fig. 2.2 illustrates how to connect the DE1-SoC board with a PC using a mini-USB-B [31] cable and where to insert the microSD card with the Linux.

Fig 2.3 illustrates the PuTTY window printing data.

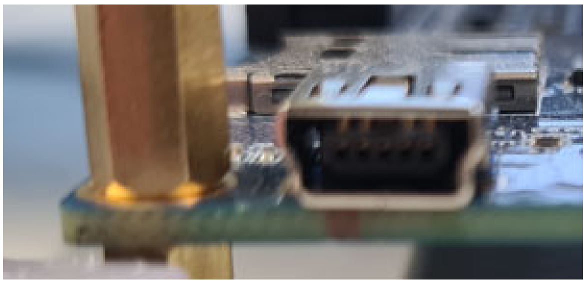
PuTTY interface (right):
Connection type: serial
Serial line: *COM#*
Speed: 115200 baud
[Save configuration as session *hps*]

To figure out #, use the *MS Window's* device manager (dt. Gerätemanager) > Anschlüsse > USB Serial Port (COM#).

(a) PuTTY interface

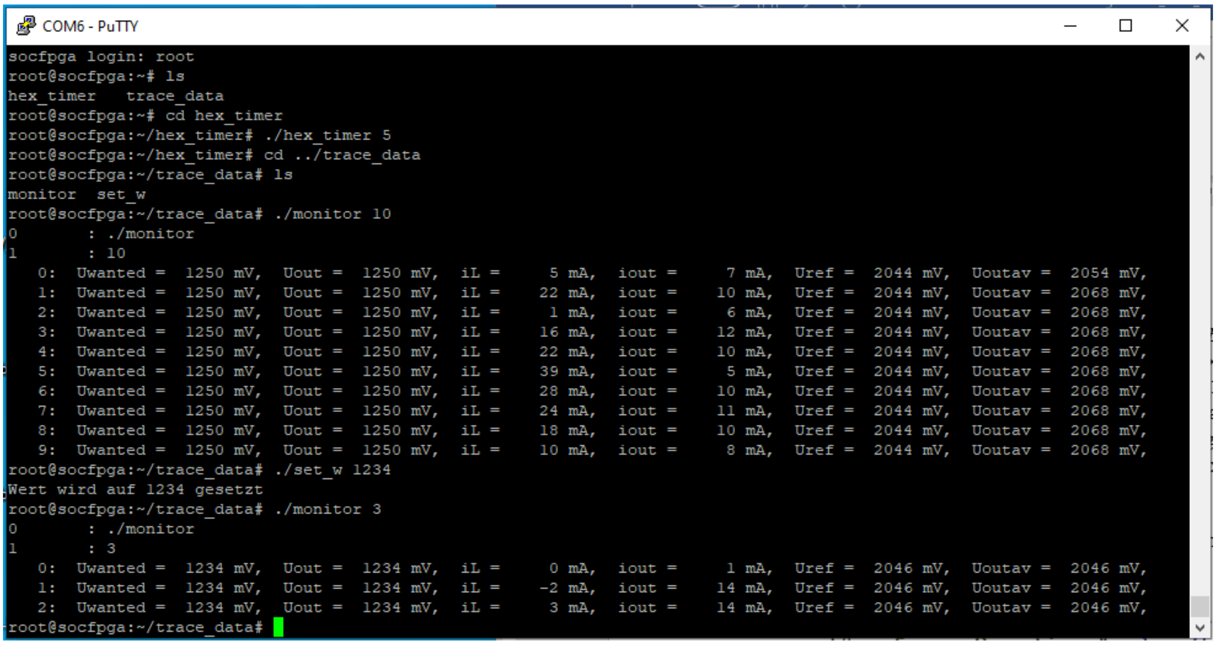


(b) *mini-USB B* [30] plug at *DE1-SoC* board



(c) *mini-USB B* and *microSD* [32] card

Fig. 2.2: *USB* connection between PC and *DE1-SoC* board:


COM6 - PuTTY

```
socfpga login: root
root@socfpga:~# ls
hex_timer   trace_data
root@socfpga:~# cd hex_timer
root@socfpga:~/hex_timer# ./hex_timer 5
root@socfpga:~/hex_timer# cd ../trace_data
root@socfpga:~/trace_data# ls
monitor  set_w
root@socfpga:~/trace_data# ./monitor 10
0        : ./monitor
1        : 10
   0:  Uwanted =  1250 mV,  Uout =  1250 mV,  iL =     5 mA,  iout =     7 mA,  Uref =  2044 mV,  Uoutav =  2054 mV,
   1:  Uwanted =  1250 mV,  Uout =  1250 mV,  iL =    22 mA,  iout =    10 mA,  Uref =  2044 mV,  Uoutav =  2068 mV,
   2:  Uwanted =  1250 mV,  Uout =  1250 mV,  iL =     1 mA,  iout =     6 mA,  Uref =  2044 mV,  Uoutav =  2068 mV,
   3:  Uwanted =  1250 mV,  Uout =  1250 mV,  iL =    16 mA,  iout =    12 mA,  Uref =  2044 mV,  Uoutav =  2068 mV,
   4:  Uwanted =  1250 mV,  Uout =  1250 mV,  iL =    22 mA,  iout =    10 mA,  Uref =  2044 mV,  Uoutav =  2068 mV,
   5:  Uwanted =  1250 mV,  Uout =  1250 mV,  iL =    39 mA,  iout =     5 mA,  Uref =  2044 mV,  Uoutav =  2068 mV,
   6:  Uwanted =  1250 mV,  Uout =  1250 mV,  iL =    28 mA,  iout =    10 mA,  Uref =  2044 mV,  Uoutav =  2068 mV,
   7:  Uwanted =  1250 mV,  Uout =  1250 mV,  iL =    24 mA,  iout =    11 mA,  Uref =  2044 mV,  Uoutav =  2068 mV,
   8:  Uwanted =  1250 mV,  Uout =  1250 mV,  iL =    18 mA,  iout =    10 mA,  Uref =  2044 mV,  Uoutav =  2068 mV,
   9:  Uwanted =  1250 mV,  Uout =  1250 mV,  iL =    10 mA,  iout =     8 mA,  Uref =  2044 mV,  Uoutav =  2068 mV,
root@socfpga:~/trace_data# ./set_w 1234
Wert wird auf 1234 gesetzt
root@socfpga:~/trace_data# ./monitor 3
0        : ./monitor
1        : 3
   0:  Uwanted =  1234 mV,  Uout =  1234 mV,  iL =     0 mA,  iout =     1 mA,  Uref =  2046 mV,  Uoutav =  2046 mV,
   1:  Uwanted =  1234 mV,  Uout =  1234 mV,  iL =    -2 mA,  iout =    14 mA,  Uref =  2046 mV,  Uoutav =  2046 mV,
   2:  Uwanted =  1234 mV,  Uout =  1234 mV,  iL =     3 mA,  iout =    14 mA,  Uref =  2046 mV,  Uoutav =  2046 mV,
root@socfpga:~/trace_data#
```

Fig. 2.3: Window "*COM# - PuTTY*" showing measured data of signals *w* (u_{wanted}), *v* (u_{ADC}, sampled ADC out), *i_L* (sampled inductor current i_L), *i_out* (sampled output current i_{out}), U_{ref} (reference voltage for current +/- measurement), $U_{out,av}$ (average output voltage measured as output of the RC lowpass (R_a, C_a) on the *DCDCbuck_Rev.10.02* board.)

3 Preparations for the Embedded Exercises

This part is typically not done during the standard practical training.

3.1 Modifications to be done at the *VHDL* code to include the *HPS*:

(This is done for files of the practical training)

1. Copy file AxiInterface.vhd to directory ...\Models_DCDCbuck\VHDL\Quartus\ci_de1soc_DCDCbuck_hps
2. Copy entity instantiation i_AxiInterface: entity work.AxiInterface(arch_AxiInterface) PORT MAP(...) into ci_de1soc_DCDCbuck.vhd
3. Add file hps.vhd to Quartus project (Project > Add/Remove File in Project > hps.vhd > Apply > OK
4. Add file AxiInterface.vhd to Quartus project (Project > Add/Remove File in Project > AxiInterface.vhd > Apply > OK
   Sequence in Window of: Project > Add/Remove Files in Project >
   1. AxiInterface.vhd, 2. hps/ip/hps/synthesis/hps.vhd, 3. hps/ip/hps/synthesis/hps.qip
   (a) Clear entry in library properties of hps/ip/hps/synthesis/hps.vhd
   (b) compile -> error within fitter
   (c) run tcl script *hps_sdram_p0_pin_assignments.tcl* using Tools > Tcl Scripts >
   (d) compile again (should work now)
5. signals reg_w, reg_v, reg_iL, reg_iout, reg_vref, reg_voutav into the system:
   (a) Within directory ...\Models_DCDCbuck\VHDL\Quartus\ci_de1soc_DCDCbuck_hps,
   architecture rtl_ci_de1soc_DCDCbuck OF ci_de1soc_DCDCbuck, add
   (i) component declaration of de1soc_DCDCbuck, add the 4 port signals:
   reg_w,reg_v,reg_iL,reg_iout: OUT std_logic_vector(31 downto 0));
   (ii) Add signal declaration:
   "SIGNAL reg_w,reg_v,reg_iL,reg_iout,reg_vref,reg_voutav: std_logic_vector(31 DOWNTO 0);" to ci_de1soc_DCDCbuck
   (iii) Add the 4 signal in component instantiation of de1soc_DCDCbuck
   i_fpga:de1soc_DCDCbuck ... PORT MAP(...,reg_w,reg_v,reg_iL,reg_iout);
   (b) Within directory ...\Models_DCDCbuck\VHDL\VHDL\rtl\de1soc
   add the 4 signals into the PORT of entity de1soc_DCDCbuck:
   ENTITY de1soc_DCDCbuck IS ...
   PORT(...,reg_w,reg_v,reg_iL,reg_iout:OUT std_logic_vector(31 downto 0));

3.2 Preparation of a *microSD* Card.

We have to copy a *Linux* [30] image on the *microSD* card [33]. First download the file *DE1_SoC_SD.zip* (66 495 KB) from [34] und unpack the required image named *DE1_SoC_SD.iso* (1 899 724 KB) as detailed in the "*DE1-SoC Getting Started Guide*" [35]. Then a Software like *Rufus* [36] can be used on the *Microsoft Windows 10* [37] operating system (OS) to copy the image bit-accurate on the *microSD* card as illustrated in Fig. 3.2. Use button "*Auswahl*" to select the image *DE1_SoC_SC.img*. Click on button *START* to write the image on the *microSD* card, click on *SCHLIESSEN* to quit the *Rufus* Software.

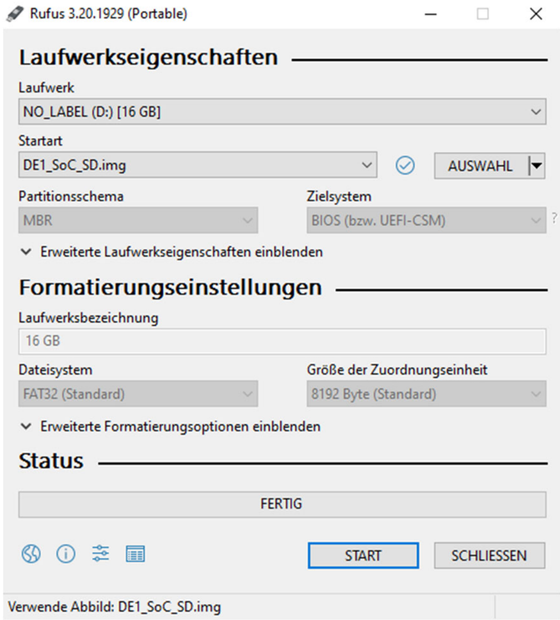


Fig. 3.2: Using *Rufus* [34] under Windows 10 to copy an image on a *microSD* card.

3.3 First *Linux* Operations on the *ARM* Processor

We assume that the *monitor* and *set_w* executable programs are not yet on the *microSD* card.

Listing 3.3: first *Linux* operations on the *ARM* processor

```
root@socfpga:~# ls                  // list directory
root@socfpga:~# ls -al ls           // list directory with all properties
root@socfpga:~# mkdir trace_data    // create directory trace_data/
root@socfpga:~# cd trace_data       // go into directory trace_data/
root@socfpga:~# ls -al              // go into directory trace_data/
root@socfpga:~# mkdir bin           // create directory bin/
```

3.4 Preparing the *microSD* card on a *Linux* Computer

Checking the *microSD* card on a *Linux* computer, we will see that it contains 2 partitions now with a size of 589 MB and 1100 MB. The latter contains the directories *home/* and *home/root/*. Within the latter we find our selfmade directory and file *home/root/trace_data/monitor*.

Some basic knowledge for operation within Linux environment

- Getting the terminal window path: *right mouse button* in window → *open in Terminal*
- "~/" is root directory, "./" is this and "../" is parent directory
- `mkdir` and `rmdir` are make and remove directory.
- Permission problems: precede a command by `sudo` (*super-user do*)
- Rename something: `$ mv <name1> <name2>` or `$ sudo mv <name1> <name2>`
  Example with permission: `$ sudo mv meas trace_data`

- Grant to all directory contents read/write/exe rights: **`$ sudo chmod 777 ./`**
- Remove recursively (i.e. also directory contents): **`$ sudo rm -r <filename>`**
- Make directory: $ **`mkdir trace_data`**
- Change directory: **`$ cd trace_data`**

Create executable monitor.

- create directory *trace_data* and go into this directory
- copy *main.c* and *MakeFile* into *trace_data/* and create there an empty folder *bin/*
- in window *trace_data/* click *right mouse button → open in terminal*
  type command **`make`**.
  If **`make`** does not work properly, install *C* compiler for *ARM* using
  `$ sudo apt install gcc-arm-linux-gnueabihf`
- Make should create in *bin/* files *monitor* and *main.o*; whereas *main.o* may be deleted
- **`$ ls -al`** // do my files have sufficient rights to be copied?
- **`$ chmod 777 *`** // grant all rights to all files in the actual directory.
- Copy

3.5 Testing the Own *C* Program *microSD* card on *Linux* on *ARM*

Listing 3.5: Run *Linux* on the *ARM* processor to monitor data

… *DCDCbuck_Rev10* board must be connected and run at this point to deliver the required monitoring data
`#  ./monitor` // print 50 measurement lines, *DCDCbuck_Rev10* must run at this point!
<make *PuTTY* window COM# wider using the mouse to avoid undesired line feeds>
`# ./monitor 10` // example 10 is the number of desired measurement lines
`# ./monitor 10 > monitor.log` // write output into file *monitor.log*.
`# ./cat monitor.log` // print the contents of *monitor.log* in the PuTTY window.
`# ./set_w 1234` to set *w*=1234mV, set *sw*(9:6) ="1111" to make that value of *w* active.

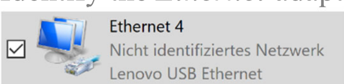
3.6 Install *Ethernet* Connection to *DE1-SoC* Board

Linux operations in the PuTTY window

- Open Putty: **`cd /etc/network`**
- **`vi interfaces`** // open file *interfaces* with editor **`vi`**
- **`d`** <to delete the 2 lines containing string *eth0*>
- **`i`** → *Enter* to insert a line // key 'i' for insert
- insert the following 3 lines:
  `iface eth0 inet static`
  <tab> **`address 10.0.0.2`** //region 10, board addressed by trailing 2, PC by 1
  <tab> **`netmask 255.255.255.0`**
- *ESC* **`:wq`** → *Enter* // write (save) file and quit
- **`cat interfaces`** // print file interfaces to check its contents
- **`ifup eth0`** // interface upload

Windows operations

- Hit *Windows key* ⊞, type **`Ethernet`** → *Return* → *click on Adapteroptionen ändern*
- Identify the *Ethernet* adaptor connected to the *DE1-SoC* board, left-click on it



- Window *Status of Ethernet#* opens →click on *Eigenschaften* (*Properties*)
- Click on *Internet-Protokoll, Version 4* →
- Set *IP-Adresse 10.0.0.1*, *Subnetz-Maske 255.255.255.4*.

(a) status window (b) adaptor properties (c) TCP/IPv4 settings

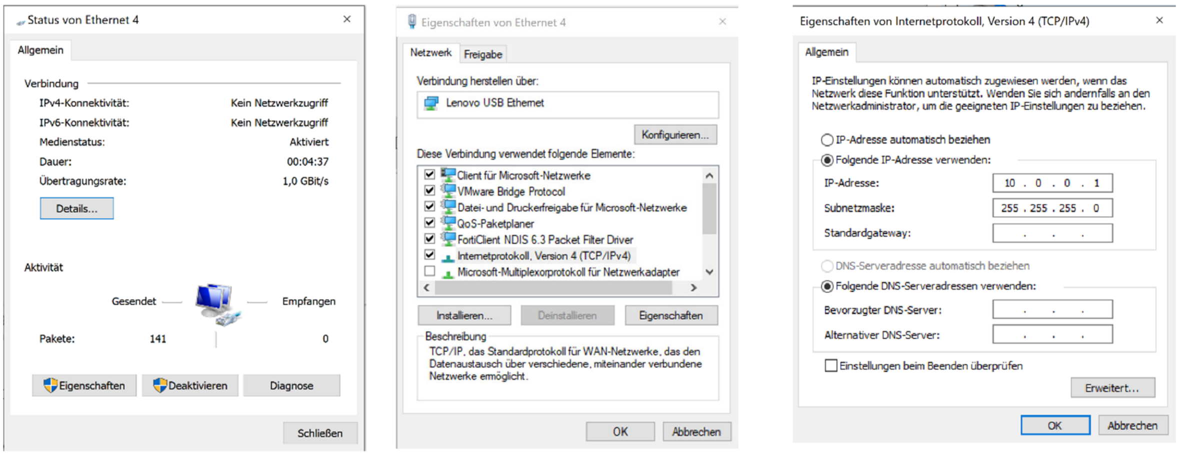


Fig. 7.6: *Ethernet* setting windows

***Linux* operations in *PuTTY* window, after every start on the *microSD* card:**

- **`ifup eth0`** // upload interface setup
- **`ping 10.0.0.1`** → *CTRL+C* // ping must reach target, *CTRL+C* stops ringing
- User: **`root`** <enter> → Password: **`root`** <enter>

Window operations to see the microSD card:

- Open *WinScp* → *neues Verbindungsziel*:
  SFTP, *Serveradresse 10.0.0.2*, *Port 22*, *User=root*, *Pwd=root*, *Verbindung vertrauen* → *ja*
- Now you can exchange data between *Windows* ⇔ *microSD* card using *WinSCP*.

3.7 Save and Duplicate an Image-File Using *Linux Ubuntu*

- Select *Open in Terminal* on *Linux* desktop background
- **`ls /dev`** // list devics
  sda = memory of the virtual machine (VM) with *Linux*
  sdb = memory of *microSD* card
- Commands used below: **`dd`**: disc dump, **`if`**: input file, **`of`**: output file, **`bs`**: block size, **`count=`**#: number of block to be copied, **`status=progress`** displays operation progress.
- Copy an image of the *microSD* card to a file in Ubuntu (will be 2500 MB big!)
  `sudo dd if=/dev/sdb of=microSD_ref_0 bs=1M count=2500 status=progress`
- Copy the image in Ubuntu to the *microSD* card (2500 MB = all → may be omitted)
  `sudo dd if=microSD_ref_0 of=/dev/sdb bs=1M count=2500 status=progress`
- Attention: if accidently sda is used, the *Linux* within the virtual machine is overwritten!

4 Conclusions

DCDCbuck board was operated as daughterboard of *DE1-SoC* board. Some embedded (hardware / software codesign) aspects were demonstrated, reading measured data from and writing data to *VHDL* signals inside the *Cyclone-V PFGA* to an external *PC*.

5 References

[1] Available: https://www.terasic.com.tw.
[2] Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=836
[3] Available: http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=886
[4] Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=165&No=836&PartNo=4
[5] CD ROM *DE1-SoC CD-ROM (rev.F Board) Version 5.1.2 of 2910-01-28* from [4]
[6] *DE1*-SoC User Manual, Ref. F, taken from [5], available: https://hps.hs-regensburg.de/~scm39115/homepage/education/labs/Lab_ElectronicBoards/DE1-SoC_UserManual.pdf
[7] *DE1*-SoC Schematic, Ref. F, taken from [5], available: https://hps.hs-regensburg.de/~scm39115/homepage/education/labs/Lab_ElectronicBoards/DE1-SoC_Schematic_revF.pdf
[8] K:\SB\Sources\EDA\Terasic\Hardware\
[9] Bode 100 Vector Network Analyzer, Omicron Labs, Available: https://www.omicron-lab.com/products/vector-network-analysis/bode-100/
[10] Florian Schwankner, *Hardware-Erweiterung einer DC/DC-Buck-Konverter-Platine zur Messung des mittleren Spulen- und Ausgangs-Stromes mittels eines ADC*, MEI-Master Projektarbeit, Elektroniklabor der OTH Regensburg, Januar 2022.
[11] Alexander Forster, *Development of a Python to VHDL Compiler Applied to Interface an ARM Processor to an FPGA*, MEI-Master Thesis, Elektroniklabor der OTH Regensburg, September 2022.
[12] Available: https://en.wikipedia.org/wiki/Intel_Quartus_Prime
[13] Available: https://en.wikipedia.org/wiki/ModelSim
[14] Av. https://www.intel.com/content/www/us/en/programmable/downloads/download-center.html
[15] Linear Technology, *LTC2308, Low Noise, 500ksps, 8-Channel, 12-Bit ADC*, available Aug. 2022 : https://www.analog.com/media/en/technical-documentation/data-sheets/2308fc.pdf
[16] Ribbon cable plugs: Sullins Headers SFH11-PBPC-D05-ST-BK, availble Aug. 2022 : https://www.digikey.de/en/products/detail/sullins-connector-solutions/SFH11-PBPC-D05-ST-BK/1990087
[17] MP770803, BNC breakout to pin, available Aug. 2022: https://de.farnell.com/multicomp-pro/mp770803/bnc-plug-sq-pin-socket-x-2-blk/dp/3703639?ost=bnc+to+pin&cfm=true
[18] BNC coaxial cable, available Aug 2022 : https://www.farnell.com/datasheets/179130.pdf
[19] Jumper *M50-1900005*, available : https://cdn.harwin.com/pdfs/M50-19X.pdf
[20] Eagle 7.2, Autodesk, available Aug. 2022 : http://eagle.autodesk.com/eagle/software-versions/7
[21] Eagle 9.6, Autodesk, availble Aug. 2022 : https://www.autodesk.com/products/eagle/blog/autodesk-eagle-9/
[22] Available: https://edg.uchicago.edu/~tang/VHDLref.pdf
[23] Available: https://www.mimuw.edu.pl/~marpe/pul/card_vhdl.pdf
[24] Available: https://www.mimuw.edu.pl/~marpe/pul/card_1164.pdf
[25] Texas Instruments, *LM27222 High-Speed 4.5A Synchronous MOSFET Driver*, available: http://www.ti.com/lit/ds/symlink/lm27222.pdf
[26] Homepage Schubert, Available Aug. 2022: https://hps.hs-regensburg.de/~scm39115/index.htm
[27] ARM: Advanced RISC Machine, avail.: https://en.wikipedia.org/wiki/ARM_architecture_family
[28] AXI: Advanced eXtensible Interface, available: https://en.wikipedia.org/wiki/Advanced_eXtensible_Interface
[29] HPS: Hard Processor System, available: https://www.intel.com/content/www/us/en/docs/programmable/683717/current/hard-processor-system-hps.html
[30] *Linux*, available: https://de.wikipedia.org/wiki/Linux

[31] *Mini-USB B*, Avalable: https://www.hb-digital.de/USB-20-Kabel-USB-A-Stecker-auf-Mini-USB-Stecker

[32] [putty] PuTTY, avaiable: https://www.putty.org/

[33] *microSC* Card, Avalable: https://de.wikipedia.org/wiki/MicroSD

[34] *Linux* image*, Altera, DE1_SoC_SD.img*, Available : http://www.terasic.com/downloads/cd-rom/de1-soc/linux_BSP/DE1_SoC_SD.zip

[35] *Altera*, *DE1-SoC Getting Started Guide*, Available: http://www.ee.ic.ac.uk/pcheung/teaching/E2_experiment/DE1-SoC_Getting_Started_Guide.pdf

[36] Rufus, available: https://rufus.ie/de/

[37] *Microsoft, Windows 10* Operating System Available: https://de.wikipedia.org/wiki/Microsoft_Windows_10

[38] Standard C++ Library reference, available*:* https://cplusplus.com/reference/

[39] *Texas Instruments, "REF5020, 2.048-V, 3-µVpp/V noise, 3-ppm/°C drift precision series voltage reference"*, Available Sep. 2022: https://www.ti.com/product/REF5020

[40] Kwang Liang Chong, "*Design, Manufacturing, Optimization and Characterization of a DC/DC Buck Converter Board Driving Up to 30V/4.5A for FPGA and µC Control*", BA Thesis, *OTH Regensburg, Electronics Lab*., Feb. 2019.

[41] Christoph Mayer, " *Reduzierung von Noise bei einer DC/DC-Buck-Konverter-Platine*", MEI-Master Projektarbeit, Elektroniklabor der OTH Regensburg, April 2022.