KONDOR AX

Advanced System Development Board

BASIC DEMOS GUIDE

2.10.2015.

UM0028

Rev. 1.4

Table of contents

1 Introduction 1

2 Downloading FPGA designs 1

2.1 DOWNLOADING FPGA DESIGNS FROM PC USING JTAG 1
  2.1.1 TO ECP5 FPGA 1
  2.1.2 TO SPI FLASH 2
2.2 GENERATING FILES FOR EMBEDDED DESIGN 2
  2.2.1 USING SLAVE SPI 2
  2.2.2 USING JTAG 3
2.3 DOWNLOADING FPGA DESIGNS FROM I.MX6 4
  2.3.1 USING SLAVE SPI 4
  2.3.2 USING JTAG 4

3 Demos 5

3.1 DEMO 1 / RD0012 - DOWNLOAD BITSTREAM AND BLINK LEDS 5
  3.1.1 HARDWARE 6
  3.1.2 DOWNLOADING BITSTREAM USING SPI 6
  3.1.3 DOWNLOADING BITSTREAM USING JTAG 7
  3.1.4 LEDS 7
3.2 DEMO 2 / RD0012 - READ AND WRITE MEMORY OVER PCIE 8
  3.2.1 HARDWARE 8
  3.2.2 DESIGN 8
  3.2.3 DRIVER 8
  3.2.4 APPLICATION 8
3.3 DEMO 3 / RD0013 - LOOPBACK TEST ON SFP GIGE OVER PCIE 10
  3.3.1 HARDWARE 10
  3.3.2 DESIGN 11
  3.3.3 DRIVER 11
  3.3.4 TEST 12
3.4 DEMO 4 / RD0014 - LOOPBACK TEST ON RMII PHY OVER PCIE 13
  3.4.1 HARDWARE 13
  3.4.2 DESIGN 13
  3.4.3 DRIVER 14
  3.4.4 TEST 14
3.5 DEMO 5 / RD0015 - LOOPBACK TEST ON RMII PHY OVER EIM 15
  3.5.1 HARDWARE 15

3.5.2 DESIGN 15
3.5.3 DRIVER 16
3.5.4 TEST 16

3.6 DEMO 6 / RD0016 - LOOPBACK TEST ON SFP GIGE CONNECTED TO FMC CARD OVER EIM 17

3.6.1 HARDWARE 18
3.6.2 DESIGN 18
3.6.3 DRIVER 18
3.6.4 TEST 19

3.7 DEMO 7 / RD0017 - LOOPBACK LVDS, VERIFY CONNECTIVITY USING LEDS 20

3.7.1 HARDWARE 20
3.7.2 DESIGN 20
3.7.3 DRIVER 20
3.7.4 TEST 21

3.8 DEMO 8 / RD0018 - UART AND I2C COMMUNICATION BETWEEN ARM AND ECP5.. 22

3.8.1 HARDWARE 22
3.8.2 DESIGN 22
3.8.3 DRIVER 22
3.8.4 UART 23
3.8.5 I2C 23

Terms of use 25

Contact info 25

Revision History

Revision	Date	Author	Modification
1.0	3.3.2015.	DM	Initial
1.1	10.3.2015.	DM	Added demo 7
1.2	13.3.2015.	DM	Added demos 3 and 4
1.3	25.3.2015.	DM	All demos added
1.4	21.9.2015.	NDZ	Demos description edited

Related Documents

ID	Code	Description
1	UM0026	KONDOR AX – User Manual
2	UM0027	KONDOR AX – Linux BSP Build Setup Guide
4	UM0029	KONDOR AX – Basic Demos Reference Design Guide

1 Introduction

All source code for FPGA design, drivers and demo applications is provided separately. Compiling sources is described in Kondor Software User Guide.

2 Downloading FPGA designs

While following chapters describe four possible ways to transfer designs to FPGA, it suffices to use only one of them, most suitable for desired use.

Direct download to FPGA is useful when having multiple versions under development.

For verified and stable designs, best choice is downloading them into SPI flash.

For designs under development or when you want to be able to switch between several designs in short time, best choice is using slave SPI.

JTAG method (from i.MX6) is supported for completeness and seems to have no benefits over slave SPI.

2.1 Downloading FPGA designs from PC using JTAG

2.1.1 To ECP5 FPGA

1. Connect micro USB cable to U33 connector on the board, install necessary USB/serial port drivers on the PC if required.
2. Start ***Lattice Diamond Programmer*** tool and select matching communication port for transferring data to the board.
3. ***Device Family*** should be ***ECP5UM***; ***Device*** should be ***LFE5UM-85F***.
4. Double click on the cell in ***Operation*** column to open ***Device properties*** dialog box. Set ***Access mode*** to ***JTAG 1532 Mode***; set ***Operation*** to ***Fast Program***.
5. Under ***Programming file***, select the desired ***.bit*** file containing FPGA design.

6. Close the dialog box by clicking ***OK***.
7. ***Menu Design*** $\rightarrow$ ***Program***.

2.1.2 To SPI flash

1. Connect micro USB cable to U33 connector on the board, install necessary USB/serial port drivers on the PC if required.
2. Start ***Lattice Diamond Programmer*** tool and select matching communication port for transferring data to the board.
3. ***Device Family*** should be ***ECP5UM***; ***Device*** should be ***LFE5UM-85F***.
4. Double click on the cell in ***Operation*** column to open ***Device properties*** dialog box. Set ***Access mode*** to ***SPI Flash Background Programming***; set ***Operation*** to ***SPI Flash Erase, Program, Verify***.
5. Under ***Programming file***, select the desired ***.bit*** file containing FPGA design. ***Family*** should be ***SPI Serial Flash***. ***Vendor*** is ***STMicro***; ***Device*** is ***SPI-M25P64***; ***Package*** is ***8-lead VDFPN8***.
6. ***Load from File*** start and end addresses.
7. Close the dialog box by clicking ***OK***.
8. ***Menu Design*** $\rightarrow$ ***Program.***

Note: since the design is saved into flash memory, it will "survive" board resets and shutdowns.

2.2 Generating files for embedded design

2.2.1 Using Slave SPI

Start ***Lattice Diamond Programmer*** tool. The USB cable is not required.

1. Set ***Device Family*** to ***ECP5UM***; set ***Device*** to ***LFE5UM-85F***.
2. Double click on the cell in ***Operation*** column to open ***Device properties*** dialog box. Set ***Access mode*** to ***Slave SPI Interface Programming***; set ***Operation*** to ***Slave SPI Fast Program***.
3. Select desired ***.bit*** file as usual.
4. ***File*** $\rightarrow$ ***Save Untitled As***, save an ***.xcf*** file into a desired folder.

Start ***Lattice Diamond Deployment Tool***.

1. ***Create New Deployment***.
2. Set ***Function Type*** to ***Embedded System***; set ***Output File Type*** to ***Slave SPI Embedded***. Click ***OK***.
3. Turn on the check mark ***Input XCF File***. Select previously saved ***.xcf*** file. Click ***Next***.
4. ***Compress embedded files*** should be selected. ***Next***.
5. Optionally change folder, and give recognizable names to algorithm and data file. ***Next***.
6. ***Generate***.

2.2.2 Using JTAG

Start ***Lattice Diamond Programmer*** tool. The USB cable is not required.

1. Set ***Device Family*** to ***ECP5UM***; set ***Device*** to ***LFE5UM-85F***.
2. Double click on the cell in ***Operation*** column to open ***Device properties*** dialog box. Set ***Access mode*** to ***JTAG 1532 Mode***; set ***Operation*** to ***Fast Program***.
3. Select desired ***.bit*** file as usual.
4. ***Menu File*** $\rightarrow$ ***Save Untitled As***, save an ***.xcf*** into a desired folder.

Start ***Lattice Diamond Deployment Tool***.

1. ***Create New Deployment***.
2. Set ***Function Type*** to ***Embedded System***; set ***Output File Type*** to ***JTAG Full VME Embedded***. Click ***OK***.
3. Turn on the check mark ***Input XCF File***. Select previously saved ***.xcf*** file. Click ***Next***.
4. ***Compress VME File*** and ***Include Header*** should be selected. ***Next***.
5. Optionally change folder, and give recognizable name to output file. ***Next***.
6. ***Generate***.

2.3 Downloading FPGA designs from i.MX6

2.3.1 Using Slave SPI

Connect a micro USB cable to U24 connector. Board should be recognized as virtual port. Start a terminal application, choose appropriate virtual port and baud rate 115200 bps.

Transfer algorithm (.sea) and data (.sed) file to the board - see Kondor Hardware Instruction Manual document for description of download procedure. Let's assume files are named spi_algo.sea and spi_data.sed.

Navigate to /ecp5com/demos/demo_* (* indicates number of demo) folder:

```
> cd /ecp5com/demos/demo_*
```

Launch SPI programming application:

```
> ../../sspiem.lxe spi_algo.sea spi_data.sed
```

Programming should last a couple of seconds.

Note: since the design is saved directly into FPGA, it will not "survive" shutdowns, but algorithm and data file will remain stored in file system. ARM CPU can be reset while preserving FPGA design by pressing reset switch on the board or telling it to:

```
> reboot
```

2.3.2 Using JTAG

Connect a micro USB cable to U24 connector. Board should be recognized as virtual port. Start a terminal application, choose appropriate virtual port and baud rate 115200 bps.

Transfer design file (.vme) to the board - see Kondor Hardware Instruction Manual document for description of download procedure. Let's assume file is named jtag.vme.

Navigate to /ecp5com/demos/demo_* (* indicates number of demo) folder:

```
> cd /ecp5com/demos/demo_*
```

Launch JTAG programming application:

```
> ../../ispvme.lxe jtag.vme
```

Programming should last about 3.5 minutes.

Note: since the design is saved directly into FPGA, it will not "survive" shutdowns, but algorithm and data file will remain stored in file system. ARM CPU can be reset while preserving FPGA design by pressing reset switch on the board or telling it to:

```
> reboot
```

3 Demos

Connect a micro USB cable to U24 connector. Board should be recognized as virtual port. Start a terminal application, choose appropriate virtual port and baud rate 115200 bps.

All files required to run demonstrations are in their corresponding folder within archives containing entire BSP or just demos. For proper positioning within folders, sometimes is needed to get back to upper folder using:

```
> cd ..
```

and then enter the desired demo folder:

```
> cd demo_*
```

(* indicates number of folder).

3.1 Demo 1 / RD0012 - download bitstream and blink LEDs

For this demo, use files from /ecp5com/demos/demo_1 folder.

3.1.1 Hardware

This demo does not have any special hardware prerequisites.

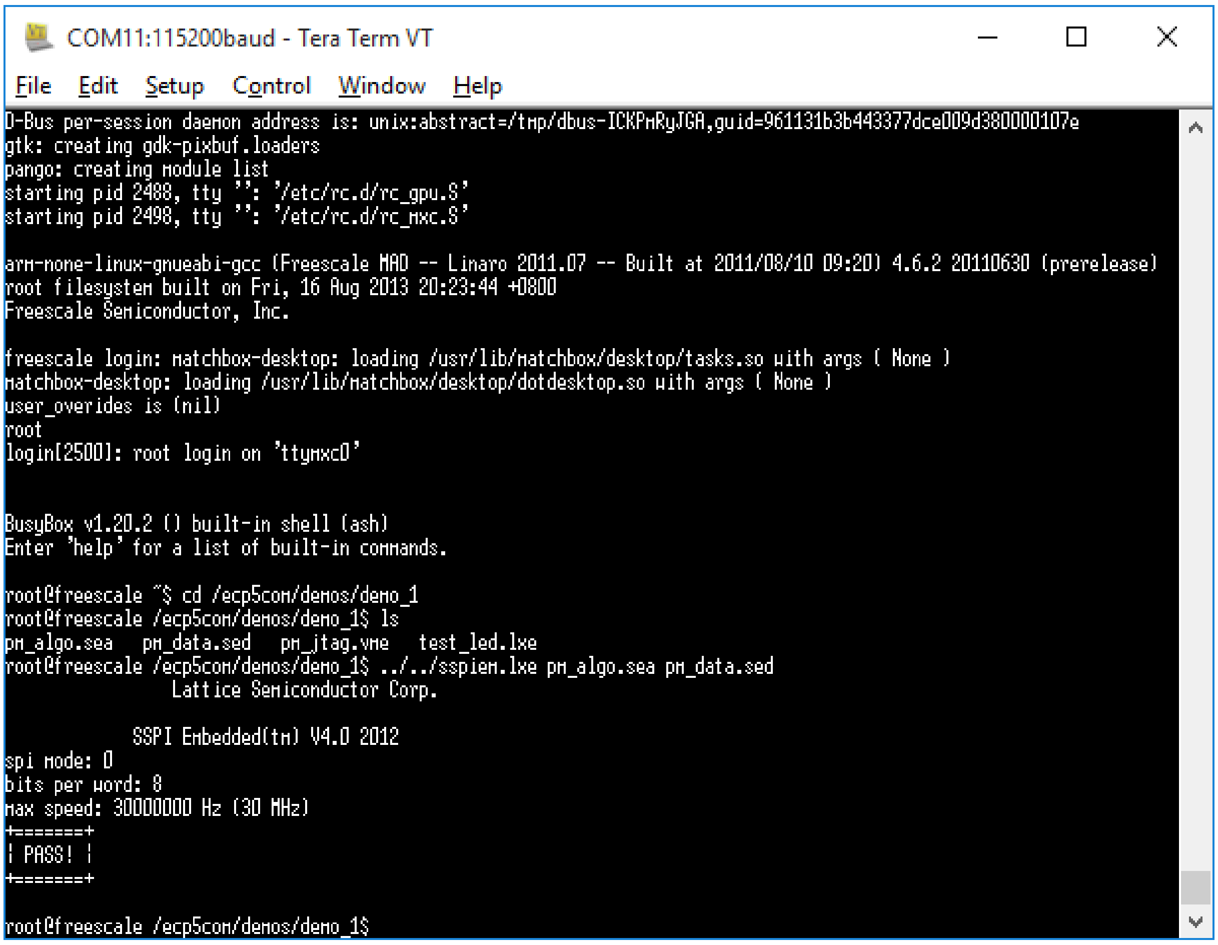
3.1.2 Downloading bitstream using SPI

FPGA design **ecp5com_pcie_mem** should be used; program it using any method from Downloading FPGA designs chapter. Slave SPI (.sea and .sed files) binaries are already provided in i.MX6 file system. Load them using following:

```
> cd /epc5com/demos/demo_1

> ../../sspiem.lxe pm_algo.sea pm_data.sed
```

This demo doesn't test design functionality, only that bitstream programming succeeded.


COM11:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
D-Bus per-session daemon address is: unix:abstract=/tmp/dbus-ICKPmRyJGA,guid=961131b3b443377dce009d380000107e
gtk: creating gdk-pixbuf.loaders
pango: creating module list
starting pid 2488, tty '': '/etc/rc.d/rc_gpu.S'
starting pid 2498, tty '': '/etc/rc.d/rc_mxc.S'

arm-none-linux-gnueabi-gcc (Freescale MAD -- Linaro 2011.07 -- Built at 2011/08/10 09:20) 4.6.2 20110630 (prerelease)
root filesystem built on Fri, 16 Aug 2013 20:23:44 +0800
Freescale Semiconductor, Inc.

freescale login: matchbox-desktop: loading /usr/lib/matchbox/desktop/tasks.so with args ( None )
matchbox-desktop: loading /usr/lib/matchbox/desktop/dotdesktop.so with args ( None )
user_overides is (nil)
root
login[2500]: root login on 'ttymxc0'


BusyBox v1.20.2 () built-in shell (ash)
Enter 'help' for a list of built-in commands.

root@freescale ~$ cd /ecp5com/demos/demo_1
root@freescale /ecp5com/demos/demo_1$ ls
pm_algo.sea   pm_data.sed   pm_jtag.vme   test_led.lxe
root@freescale /ecp5com/demos/demo_1$ ../../sspiem.lxe pm_algo.sea pm_data.sed
                Lattice Semiconductor Corp.

            SSPI Embedded(tm) V4.0 2012
spi mode: 0
bits per word: 8
max speed: 30000000 Hz (30 MHz)
+=======+
| PASS! |
+=======+

root@freescale /ecp5com/demos/demo_1$
```

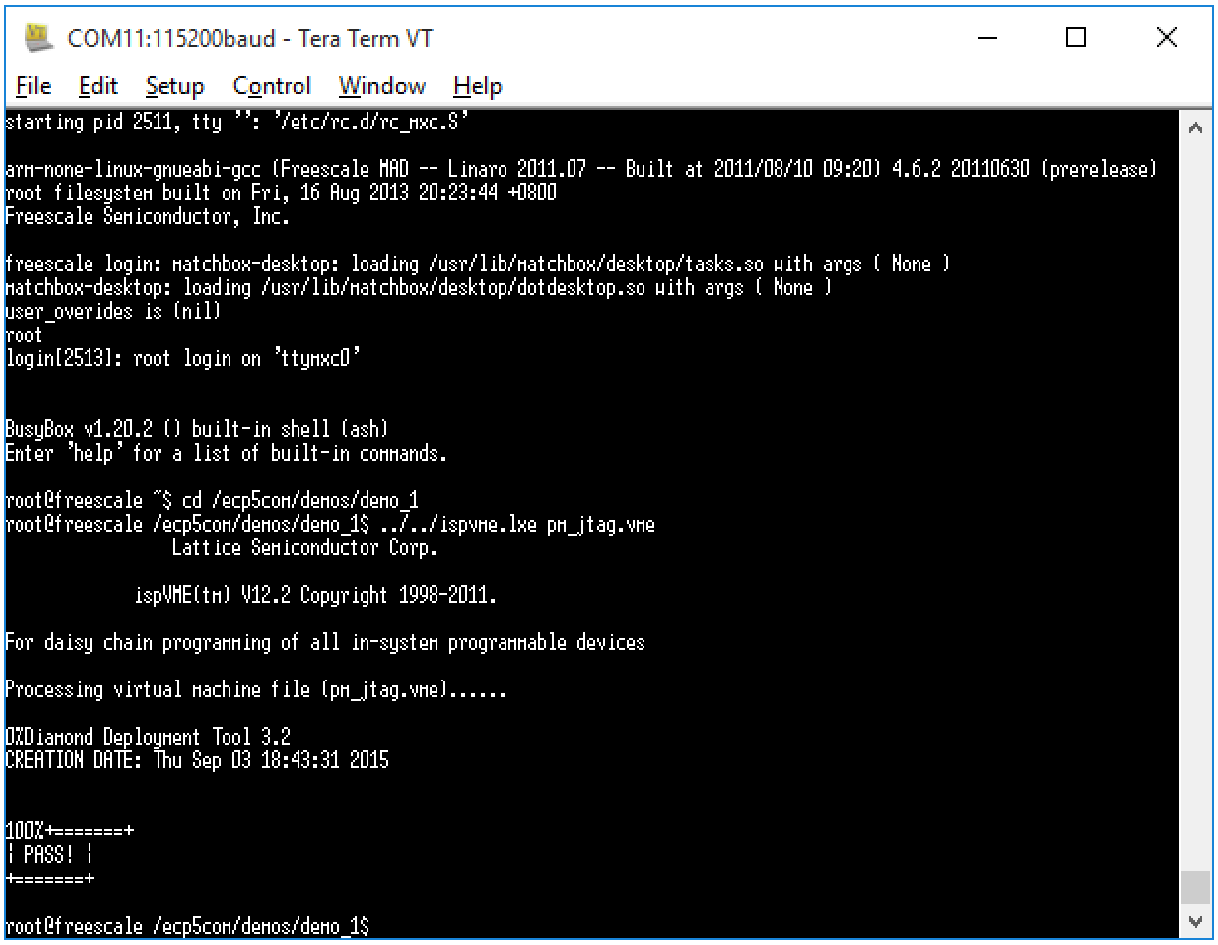
Figure 1: Downloading bitstream using SPI

3.1.3 Downloading bitstream using JTAG

FPGA design **ecp5com_pcie_mem** should be used; program it using any method from Downloading FPGA designs chapter. JTAG (.vme) binary file is already provided in i.MX6 file system. Load it using following:

```
> cd /epc5com/demos/demo_1

> ../../ispvme.lxe pm_jtag.vme
```


COM11:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
starting pid 2511, tty '': '/etc/rc.d/rc_mxc.S'

arm-none-linux-gnueabi-gcc (Freescale MAD -- Linaro 2011.07 -- Built at 2011/08/10 09:20) 4.6.2 20110630 (prerelease)
root filesystem built on Fri, 16 Aug 2013 20:23:44 +0800
Freescale Semiconductor, Inc.

freescale login: matchbox-desktop: loading /usr/lib/matchbox/desktop/tasks.so with args ( None )
matchbox-desktop: loading /usr/lib/matchbox/desktop/dotdesktop.so with args ( None )
user_overides is (nil)
root
login[2513]: root login on 'ttymxc0'


BusyBox v1.20.2 () built-in shell (ash)
Enter 'help' for a list of built-in commands.

root@freescale ~$ cd /ecp5com/demos/demo_1
root@freescale /ecp5com/demos/demo_1$ ../../ispvme.lxe pm_jtag.vme
                 Lattice Semiconductor Corp.

             ispVME(tm) V12.2 Copyright 1998-2011.

For daisy chain programming of all in-system programmable devices

Processing virtual machine file (pm_jtag.vme)......

DXDiamond Deployment Tool 3.2
CREATION DATE: Thu Sep 03 18:43:31 2015


100%+=======+
| PASS! |
+=======+

root@freescale /ecp5com/demos/demo_1$
```

Figure 2: Downloading bitstream using JTAG

3.1.4 LEDs

Compiled executable file test_led.lxe is provided and needs to be run by:

```
> ./test_led.lxe
```

It may be necessary to adjust file permissions by running

```
> chmod +x ./test_led.lxe
```

3.2 Demo 2 / RD0012 - read and write memory over PCIe

For this demo, use files from /ecp5com/demos/demo_2 folder.

3.2.1 Hardware

This demo does not have any special hardware prerequisites.

3.2.2 Design

FPGA design **ecp5com_pcie_mem** should be used; program it using any method from Downloading FPGA designs chapter. Both, Slave SPI (.sea and .sed files) and JTAG (.vme) binaries are already provided in i.MX6 file system. Load Slave SPI binaries using following:

```
> cd /epc5com/demos/demo_2

> ../../sspiem.lxe pm_algo.sea pm_data.sed
```

Reboot ARM afterwards because otherwise kernel won't be aware of a new PCIe peripheral device. Type:

```
> reboot
```

3.2.3 Driver

Load compiled ecp5com_pcie.ko binary using:

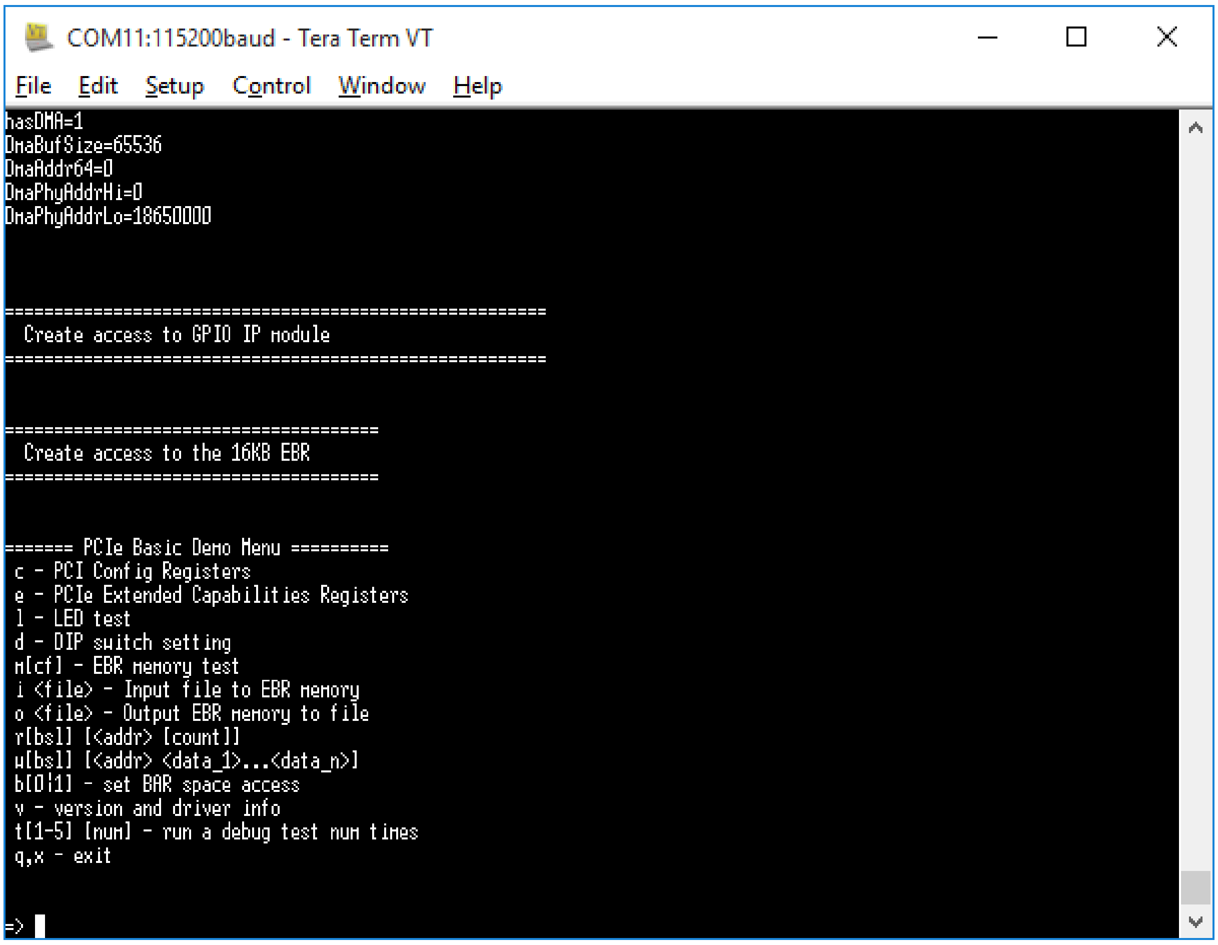
```
> insmod ecp5com_pcie.ko
```

3.2.4 Application

Compiled test application is named test_pcie.lxe. Start it with:

```
> ./test_pcie.lxe
```

List of options should be shown on the screen.


COM11:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
hasDMA=1
DmaBufSize=65536
DmaAddr64=0
DmaPhyAddrHi=0
DmaPhyAddrLo=18650000



=======================================================
  Create access to GPIO IP module
=======================================================



======================================
  Create access to the 16KB EBR
======================================



======= PCIe Basic Demo Menu ==========
 c - PCI Config Registers
 e - PCIe Extended Capabilities Registers
 l - LED test
 d - DIP switch setting
 m[cf] - EBR memory test
 i <file> - Input file to EBR memory
 o <file> - Output EBR memory to file
 r[bsl] [<addr> [count]]
 w[bsl] [<addr> <data_1>...<data_n>]
 b[0|1] - set BAR space access
 v - version and driver info
 t[1-5] [num] - run a debug test num times
 q,x - exit


=>
```

Figure 3: Memory test options

To test the memory, first output current values from some memory locations:

```
> rl 0 4
```

(some more or less random values should be shown)

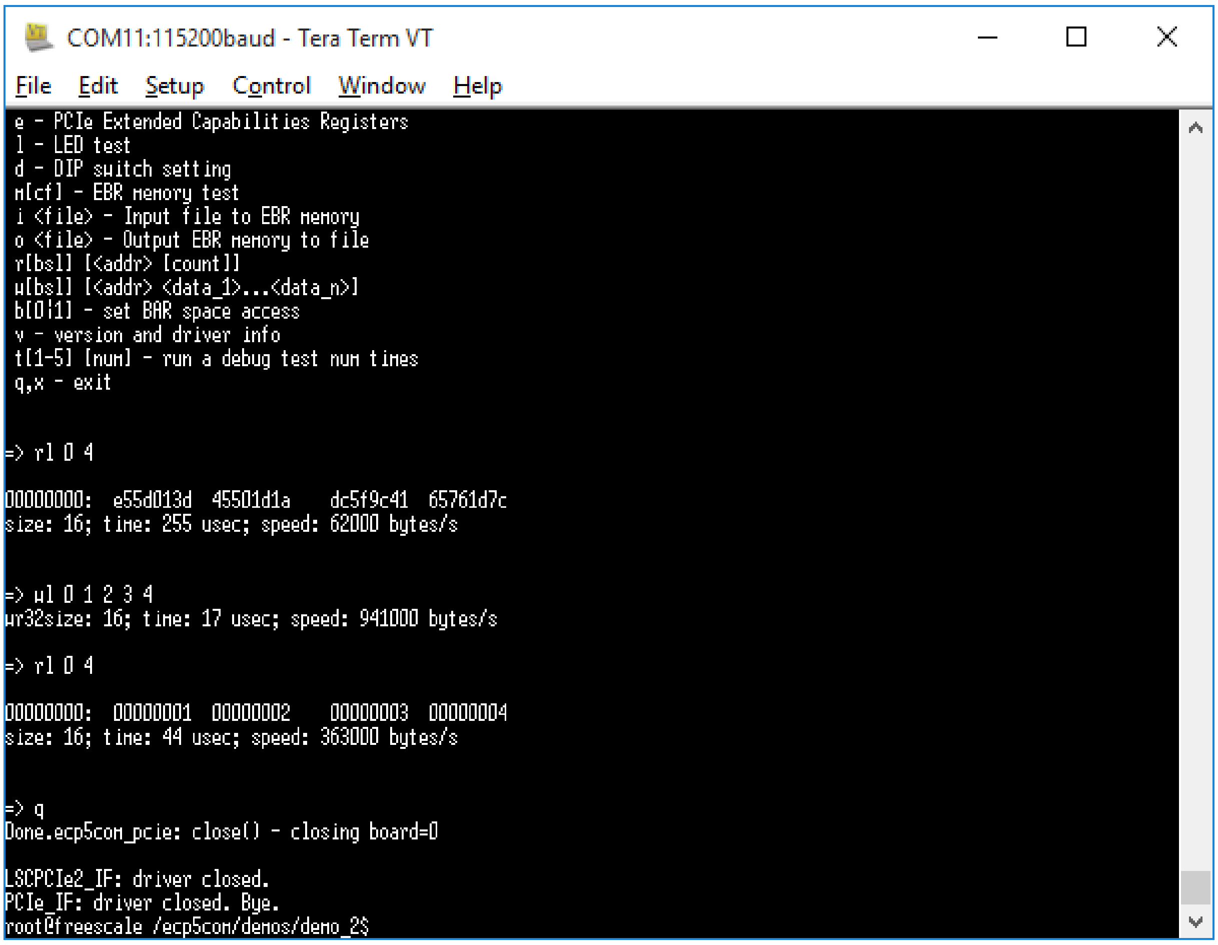
Then write some new values:

```
> wl 0 1 2 3 4
```

Verify new values were written:

```
> rl 0 4
```

(00000001 00000002 00000003 00000004 should be new contents)


COM11:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
e - PCIe Extended Capabilities Registers
l - LED test
d - DIP switch setting
m[cf] - EBR memory test
i <file> - Input file to EBR memory
o <file> - Output EBR memory to file
r[bsl] [<addr> [count]]
w[bsl] [<addr> <data_1>...<data_n>]
b[0|1] - set BAR space access
v - version and driver info
t[1-5] [num] - run a debug test num times
q,x - exit


=> rl 0 4

00000000:  e55d013d  45501d1a    dc5f9c41  65761d7c
size: 16; time: 255 usec; speed: 62000 bytes/s


=> wl 0 1 2 3 4
wr32size: 16; time: 17 usec; speed: 941000 bytes/s

=> rl 0 4

00000000:  00000001  00000002    00000003  00000004
size: 16; time: 44 usec; speed: 363000 bytes/s


=> q
Done.ecp5com_pcie: close() - closing board=0

LSCPCIe2_IF: driver closed.
PCIe_IF: driver closed. Bye.
root@freescale /ecp5com/demos/demo_2$
```

Figure 4: Memory reading and writing

Tests other than accessing memory may not work; this is behavior per design.

3.3 Demo 3 / RD0013 - loopback test on SFP GigE over PCIe

For this demo, use files from /ecp5com/demos/demo_3 folder.

3.3.1 Hardware

This demo needs an SGMII network module (Cisco GLC-T) inserted into outer SFP cage (J9/J10). A network cable is optional since the primary goal of the demo is to test loopback functionality.

3.3.2 Design

FPGA design **ecp5com_pcie_sgmii** should be used; program it using any method from Downloading FPGA designs chapter. Both, Slave SPI (.sea and .sed files) and JTAG (.vme) binaries are already provided in i.MX6 file system. Load Slave SPI binaries using following:

```
> cd /epc5com/demos/demo_3

> ../../sspiem.lxe ps_algo.sea ps_data.sed
```

ARM must be rebooted afterwards. Run:

```
> reboot
```

3.3.3 Driver

Compiled driver binary file ecp5pcie.ko is already provided. Load it using:

```
> insmod ecp5pcie.ko sgmii=1 loopback=1
```

'sgmii=1' option activates SGMII, instead of RMII, functionality within the driver. The driver can also automatically detect SGMII or RMII design from design's reported PCIe subsystem code.

'loopback=1' means driver will not send and receive packets from the connected network cable (it will activate internal loopback within the PHY).

To activate the driver, run:

```
> ifconfig eth1 up 192.168.2.222
```

Optionally, if network connection (must be Gigabit Ethernet) is wanted to test, loopback should be disabled. If this is done after loading above mentioned driver with loopback enabled, it must be unloaded first:

```
> rmmod ecp5pcie.ko
```

and then loaded again with different loopback value:

```
> insmod ecp5pcie.ko sgmii=1 loopback=0
```

For activating driver IP address might need to be changed to match address range used in your LAN. Assuming SFP cage is inserted and a cable connects the board to a working LAN, you should see a message signaling "Link UP" on serial terminal.

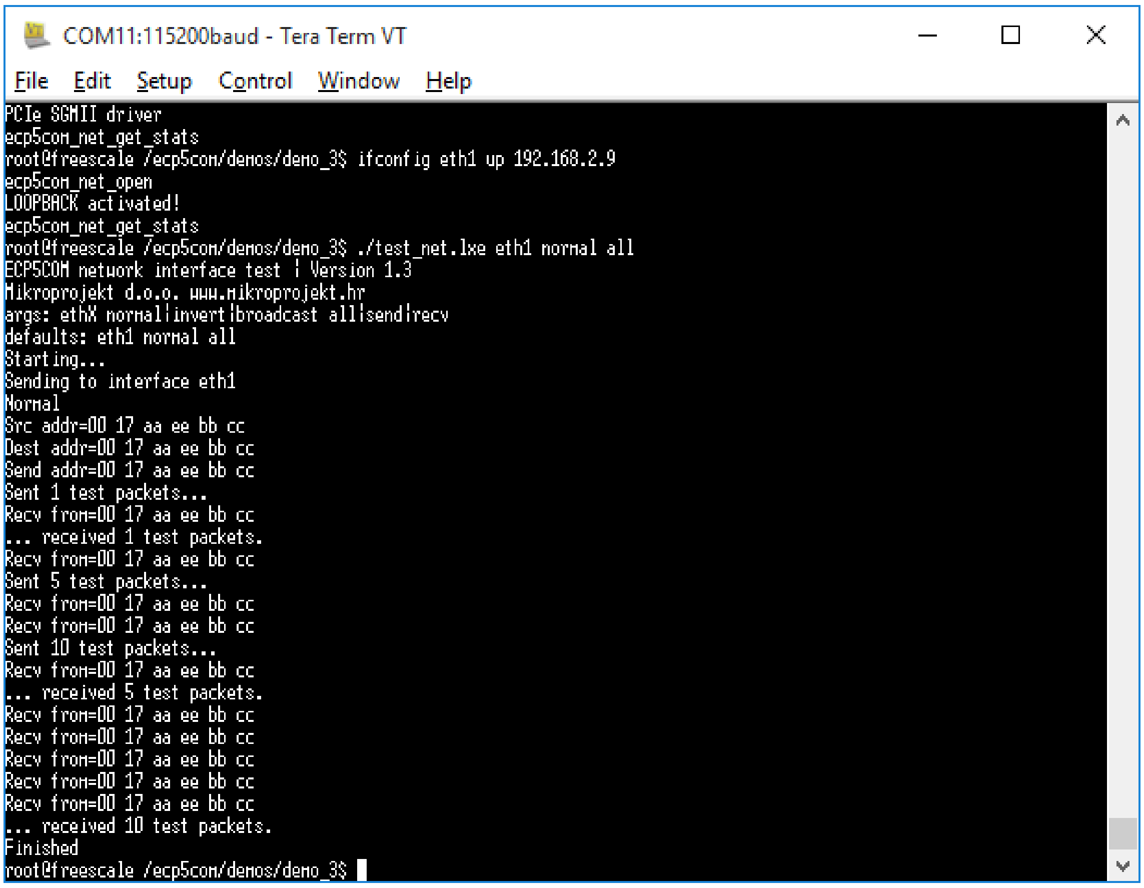
3.3.4 Test

Compiled test application test_net.lxe is provided. This application verifies only loopback functionality.

Start it with:

```
> ./test_net.lxe eth1 normal all
```

Application should print out that it sent and received some number of test network packets. Number of sent and received packets must match and packet contents is checked too.


COM11:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
PCIe SGMII driver
ecp5com_net_get_stats
root@freescale /ecp5com/demos/demo_3$ ifconfig eth1 up 192.168.2.9
ecp5com_net_open
LOOPBACK activated!
ecp5com_net_get_stats
root@freescale /ecp5com/demos/demo_3$ ./test_net.lxe eth1 normal all
ECP5COM network interface test | Version 1.3
Mikroprojekt d.o.o. www.mikroprojekt.hr
args: ethX normal|invert|broadcast all|send|recv
defaults: eth1 normal all
Starting...
Sending to interface eth1
Normal
Src addr=00 17 aa ee bb cc
Dest addr=00 17 aa ee bb cc
Send addr=00 17 aa ee bb cc
Sent 1 test packets...
Recv from=00 17 aa ee bb cc
... received 1 test packets.
Recv from=00 17 aa ee bb cc
Sent 5 test packets...
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Sent 10 test packets...
Recv from=00 17 aa ee bb cc
... received 5 test packets.
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
... received 10 test packets.
Finished
root@freescale /ecp5com/demos/demo_3$
```

Figure 5: Loopback test on SFP GigE over PCIe

The application supports following options:

- argument 1 = interface name to use for executing tests; normally you'd want to always use 'eth1' here since 'eth0' is already reserved by ARMs built-in network interface
- argument 2 defines test packets destination address:
    - 'normal' means use test interfaces own MAC address as destination address of test packets (since we want the packets to return to us)
    - 'invert' reverses all bytes in destination MAC address - one usage for this to verify that interface will not receive packets not addressed to it
    - 'broadcast' uses 0xFF ... 0xFF broadcast address
- argument 3 defines what this instance of the application should do:
    - 'all' sends and receives test packets
    - 'send' only sends
    - 'recv' only receives

3.4 Demo 4 / RD0014 - loopback test on RMII PHY over PCIe

For this demo, use files from /ecp5com/demos/demo_4 folder.

3.4.1 Hardware

This demo does not have any special hardware prerequisites.

3.4.2 Design

FPGA design **ecp5com_pcie_rmii** should be used; program it using any method from Downloading FPGA designs chapter. Both, Slave SPI (.sea and .sed files) and JTAG (.vme) binaries are already provided in i.MX6 file system. Load Slave SPI binaries using following:

```
> cd /epc5com/demos/demo_4

> ../../sspiem.lxe pr_algo.sea pr_data.sed
```

ARM must be rebooted afterwards. Run:

```
> reboot
```

3.4.3 Driver

Compiled driver binary file ecp5pcie.ko is already provided. Load it using:

```
> insmod ecp5pcie.ko loopback=1
```

'sgmii=0' (omitted as it is default parameter value) option activates RMII, instead of SGMII, functionality within the driver.

'loopback=1' means driver will not send and receive packets from the connected network cable (it will activate internal loopback within the PHY).

To activate the driver, run:

```
> ifconfig eth1 up 192.168.2.222
```

Optionally, if network cable is wanted to test, see example in chapter 3.3.3.

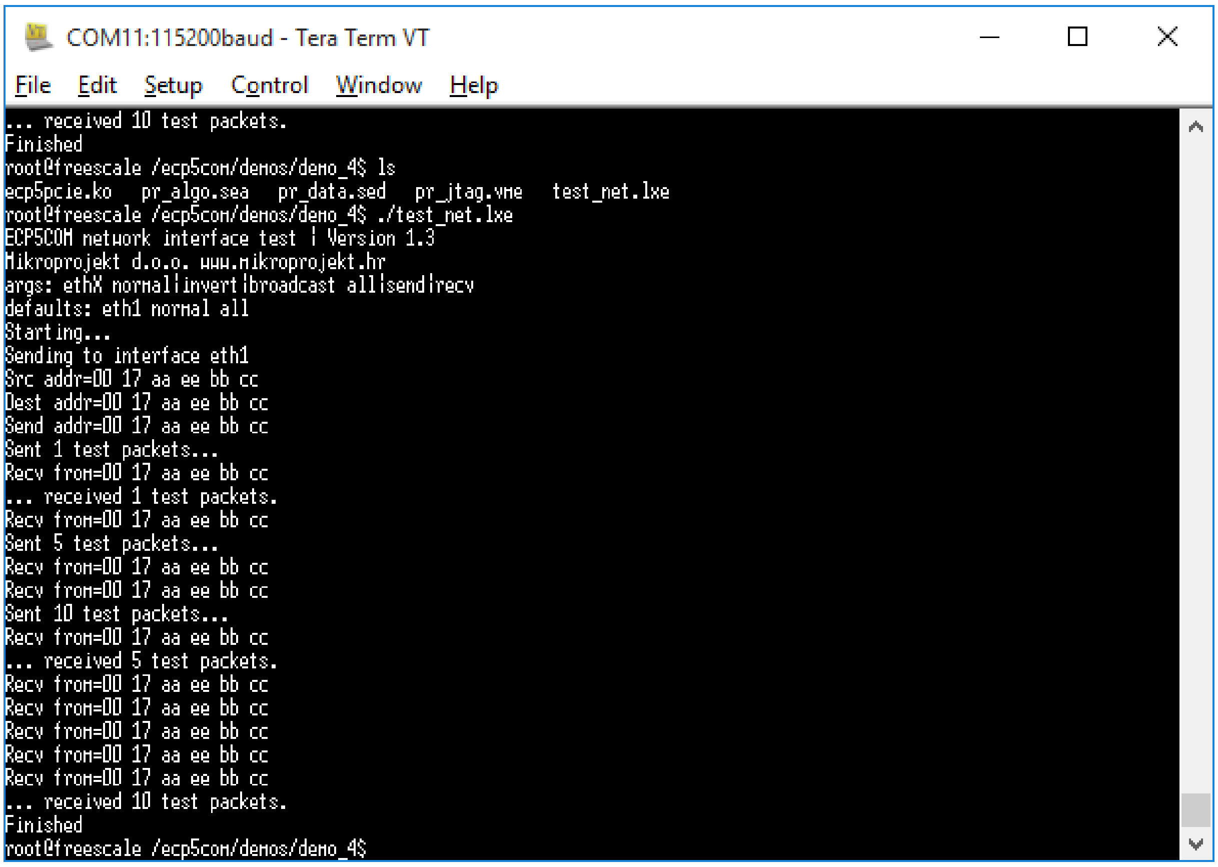
3.4.4 Test

Compiled test application test_net.lxe is provided. It verifies only loopback functionality.

Start the application using:

```
> ./test_net.lxe eth1 normal all
```

Application should print out that it sent and received some number of test network packets. Number of sent and received packets must match and packet contents is checked too.


COM11:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
... received 10 test packets.
Finished
root@freescale /ecp5com/demos/demo_4$ ls
ecp5pcie.ko   pr_algo.sea   pr_data.sed   pr_jtag.vme   test_net.lxe
root@freescale /ecp5com/demos/demo_4$ ./test_net.lxe
ECP5COM network interface test | Version 1.3
Mikroprojekt d.o.o. www.mikroprojekt.hr
args: ethX normal|invert|broadcast all|send|recv
defaults: eth1 normal all
Starting...
Sending to interface eth1
Src addr=00 17 aa ee bb cc
Dest addr=00 17 aa ee bb cc
Send addr=00 17 aa ee bb cc
Sent 1 test packets...
Recv from=00 17 aa ee bb cc
... received 1 test packets.
Recv from=00 17 aa ee bb cc
Sent 5 test packets...
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Sent 10 test packets...
Recv from=00 17 aa ee bb cc
... received 5 test packets.
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
... received 10 test packets.
Finished
root@freescale /ecp5com/demos/demo_4$
```

Figure 6: Loopback test on RMII PHY over PCIe

Application supported options are provided in previous demo (Demo 3 – chapter 2.3.4).

3.5 Demo 5 / RD0015 - loopback test on RMII PHY over EIM

For this demo, use files from /ecp5com/demos/demo_5 folder.

3.5.1 Hardware

This demo does not have any special hardware prerequisites.

3.5.2 Design

FPGA design **ecp5com_eim_rmii** should be used; program it using any method from Downloading FPGA designs chapter. Both, Slave SPI (.sea and .sed files) and

JTAG (.vme) binaries are already provided in i.MX6 file system. Load Slave SPI binaries using following:

```
> cd /epc5com/demos/demo_5

> ../../sspiem.lxe er_algo.sea er_data.sed
```

ARM does not need to be rebooted afterwards.

3.5.3 Driver

Compiled driver binary file ecp5eim.ko is already provided. As this demo doesn't require reboot, there is a possibility some other driver is already loaded. Check it with command:

```
> lsmod
```

If there is one loaded, unload it first using:

```
> rmmod <name of file>
```

(see example of command in chapter 3.3.3).

Driver for this demo should be loaded using:

```
> insmod ecp5eim.ko loopback=1
```

'sgmii=0' (omitted as it is default parameter value) option activates RMII, instead of SGMII, functionality within the driver.

'loopback=1' means driver will not send and receive packets from the connected network cable (it will activate internal loopback within the PHY).

To activate the driver, run:

```
> ifconfig eth1 up 192.168.2.222
```

Optionally, if network cable is wanted to test, see example in chapter 3.3.3.

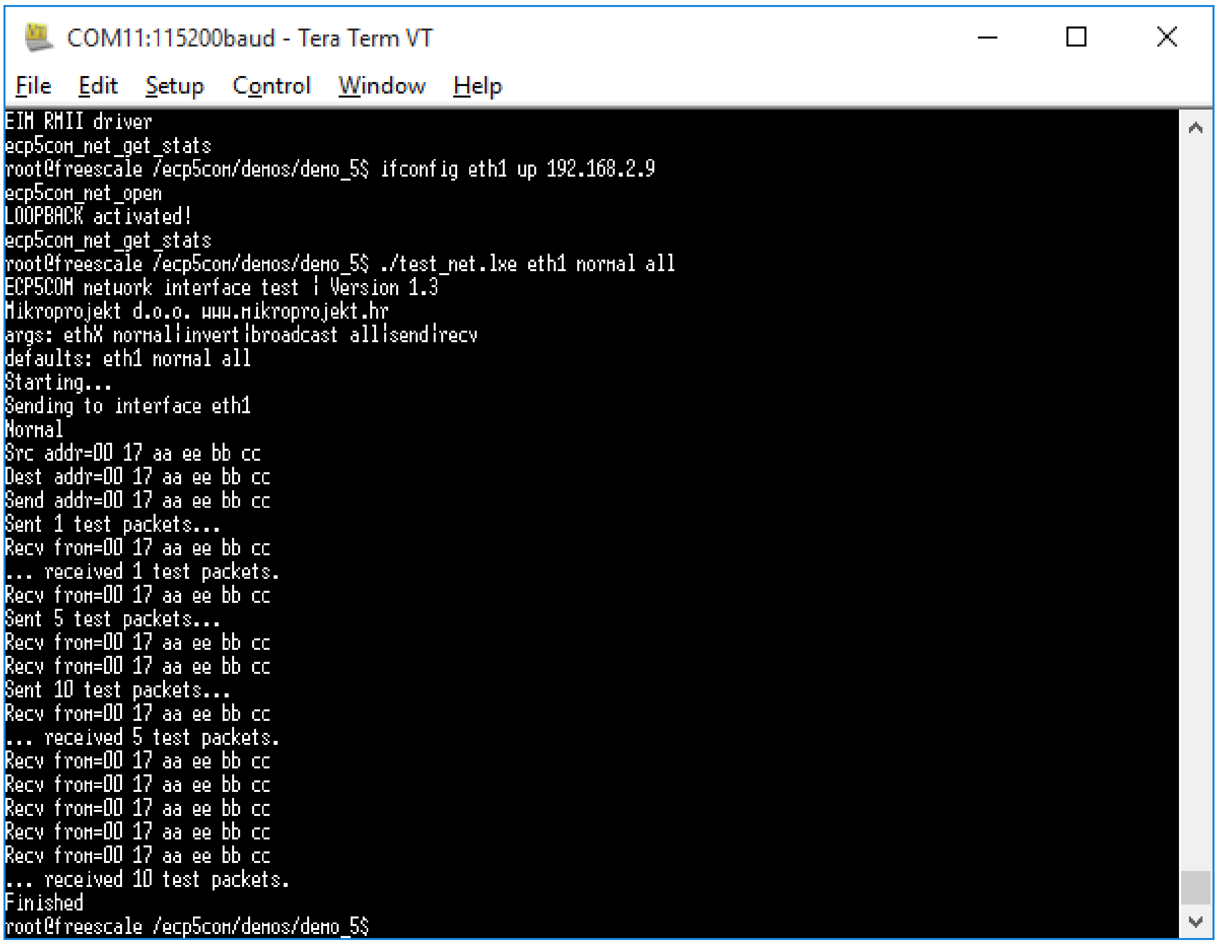
3.5.4 Test

Compiled test application test_net.lxe is provided. It verifies only loopback functionality.

Start the application using:

```
> ./test_net.lxe eth1 normal all
```

Application should print out that it sent and received some number of test network packets. Number of sent and received packets must match and packet contents is checked too.


COM11:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
EIM RMII driver
ecp5com_net_get_stats
root@freescale /ecp5com/demos/demo_5$ ifconfig eth1 up 192.168.2.9
ecp5com_net_open
LOOPBACK activated!
ecp5com_net_get_stats
root@freescale /ecp5com/demos/demo_5$ ./test_net.lxe eth1 normal all
ECP5COM network interface test | Version 1.3
Mikroprojekt d.o.o. www.mikroprojekt.hr
args: ethX normal|invert|broadcast all|send|recv
defaults: eth1 normal all
Starting...
Sending to interface eth1
Normal
Src addr=00 17 aa ee bb cc
Dest addr=00 17 aa ee bb cc
Send addr=00 17 aa ee bb cc
Sent 1 test packets...
Recv from=00 17 aa ee bb cc
... received 1 test packets.
Recv from=00 17 aa ee bb cc
Sent 5 test packets...
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Sent 10 test packets...
Recv from=00 17 aa ee bb cc
... received 5 test packets.
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
... received 10 test packets.
Finished
root@freescale /ecp5com/demos/demo_5$
```

Figure 7: Loopback test on RMII PHY over EIM

Application supported options are provided in chapter 2.3.4.

3.6 Demo 6 / RD0016 - loopback test on SFP GigE connected to FMC card over EIM

For this demo, use files from /ecp5com/demos/demo_6 folder.

3.6.1 Hardware

This demo needs a FMC - SFP card (HTG-FMC-x4-SFP www.HiTechGlobal.com) and SFP module (Cisco GLC-T) inserted in DP2 SFP Cage on the card.

3.6.2 Design

FPGA design **ecp5com_eim_fmc** should be used; program it using any method from Downloading FPGA designs chapter. Both, Slave SPI (.sea and .sed files) and JTAG (.vme) binaries are already provided in i.MX6 file system. Load Slave SPI binaries using following:

```
> cd /epc5com/demos/demo_6

> ../../sspiem.lxe es_algo.sea es_data.sed
```

ARM does not need to be rebooted afterwards.

3.6.3 Driver

Compiled driver binary file ecp5eim.ko is already provided. As this demo doesn't require reboot, there is a possibility some other driver is already loaded. Check it with command:

```
> lsmod
```

If there is one loaded, unload it first using:

```
> rmmod <name of file>
```

(see example of command in chapter 3.3.3).

Driver for this demo should be loaded using:

```
> insmod ecp5eim.ko sgmii=1 loopback=1
```

'sgmii=1' option activates SGMII, instead of RMII, functionality within the driver.

'loopback=1' means driver will not send and receive packets from the connected network cable (it will activate internal loopback within the PHY).

To activate the driver, run:

```
> ifconfig eth1 up 192.168.2.222
```

Optionally, if network cable is wanted to test, see example in chapter 3.3.3.

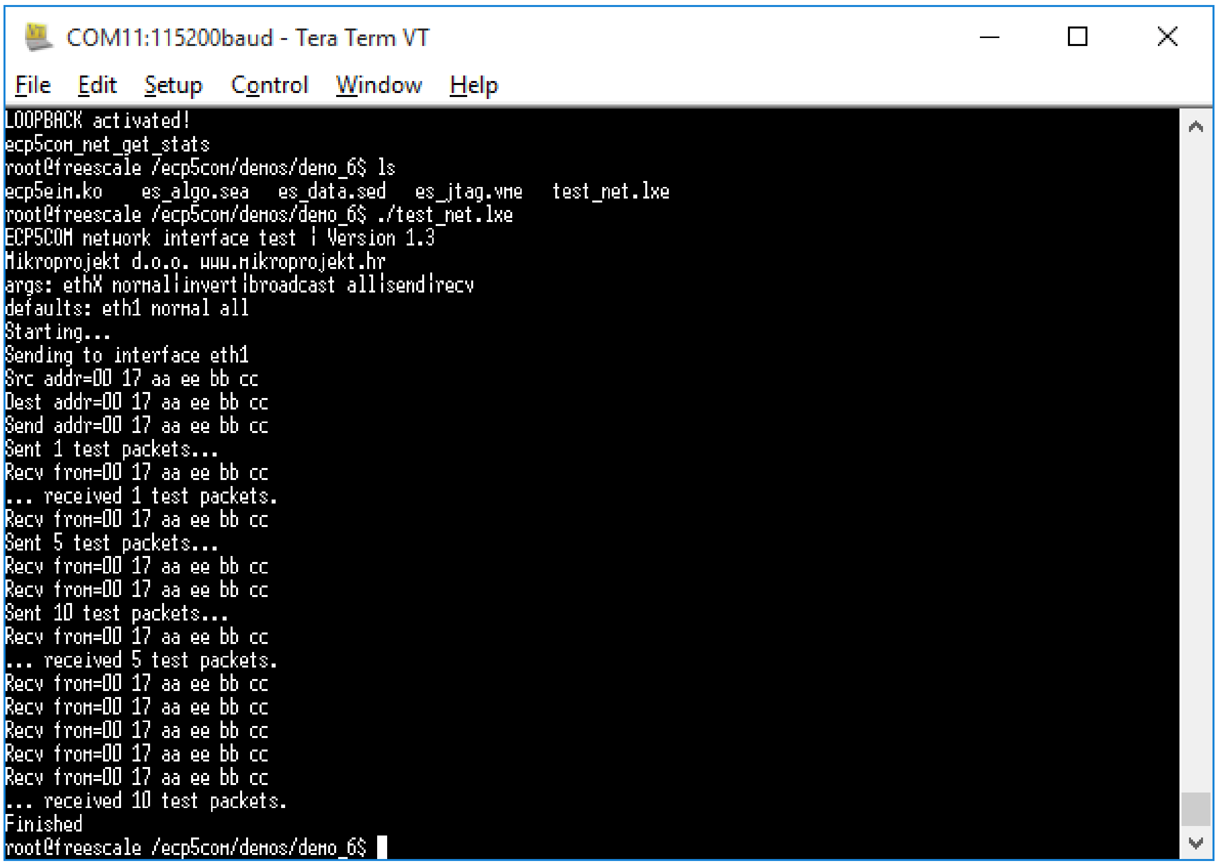
3.6.4 Test

Compiled test application test_net.lxe is provided. It verifies only loopback functionality.

Start the application using:

```
> ./test_net.lxe eth1 normal all
```

Application should print out that it sent and received some number of test network packets. Number of sent and received packets must match and packet contents is checked too.


COM11:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
LOOPBACK activated!
ecp5com_net_get_stats
root@freescale /ecp5com/demos/demo_6$ ls
ecp5eim.ko    es_algo.sea   es_data.sed   es_jtag.vme   test_net.lxe
root@freescale /ecp5com/demos/demo_6$ ./test_net.lxe
ECP5COM network interface test | Version 1.3
Mikroprojekt d.o.o. www.mikroprojekt.hr
args: ethX normal|invert|broadcast all|send|recv
defaults: eth1 normal all
Starting...
Sending to interface eth1
Src addr=00 17 aa ee bb cc
Dest addr=00 17 aa ee bb cc
Send addr=00 17 aa ee bb cc
Sent 1 test packets...
Recv from=00 17 aa ee bb cc
... received 1 test packets.
Recv from=00 17 aa ee bb cc
Sent 5 test packets...
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Sent 10 test packets...
Recv from=00 17 aa ee bb cc
... received 5 test packets.
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
Recv from=00 17 aa ee bb cc
... received 10 test packets.
Finished
root@freescale /ecp5com/demos/demo_6$
```

Figure 8: Loopback test on SFP GigE connected to FMC card over EIM

Application supported options are provided in chapter 2.3.4.

3.7 Demo 7 / RD0017 - loopback LVDS, verify connectivity using LEDs

For this demo, use files from /ecp5com/demos/demo_7 folder.

3.7.1 Hardware

This demo needs appropriately connected LVDS lines. According to schematic, next pin connections should be done on connector J18:

- 3 – 19 and 4 – 20
- 7 – 23 and 8 – 24
- 11 – 27 and 12 – 28
- 15 – 35 and 16 – 36.

3.7.2 Design

FPGA design **ecp5com_lvds** should be used; program it using any method from Downloading FPGA designs chapter. Both, Slave SPI (.sea and .sed files) and JTAG (.vme) binaries are already provided in i.MX6 file system. Load Slave SPI binaries using following:

```
> cd /epc5com/demos/demo_7

> ../../sspiem.lxe lvds_algo.sea lvds_data.sed
```

ARM does not need to be rebooted afterwards.

3.7.3 Driver

This demo uses EIM interface for communication between CPU and FPGA. Compiled driver binary file ecp5com_eim.ko is already provided. As this demo doesn’t require reboot, there is a possibility some other driver is already loaded. Check it with command:

```
> lsmod
```

If there is one loaded, unload it first using:

```
> rmmod <name of file>
```

(see example of command in chapter 3.3.3).

Load EIM driver using:

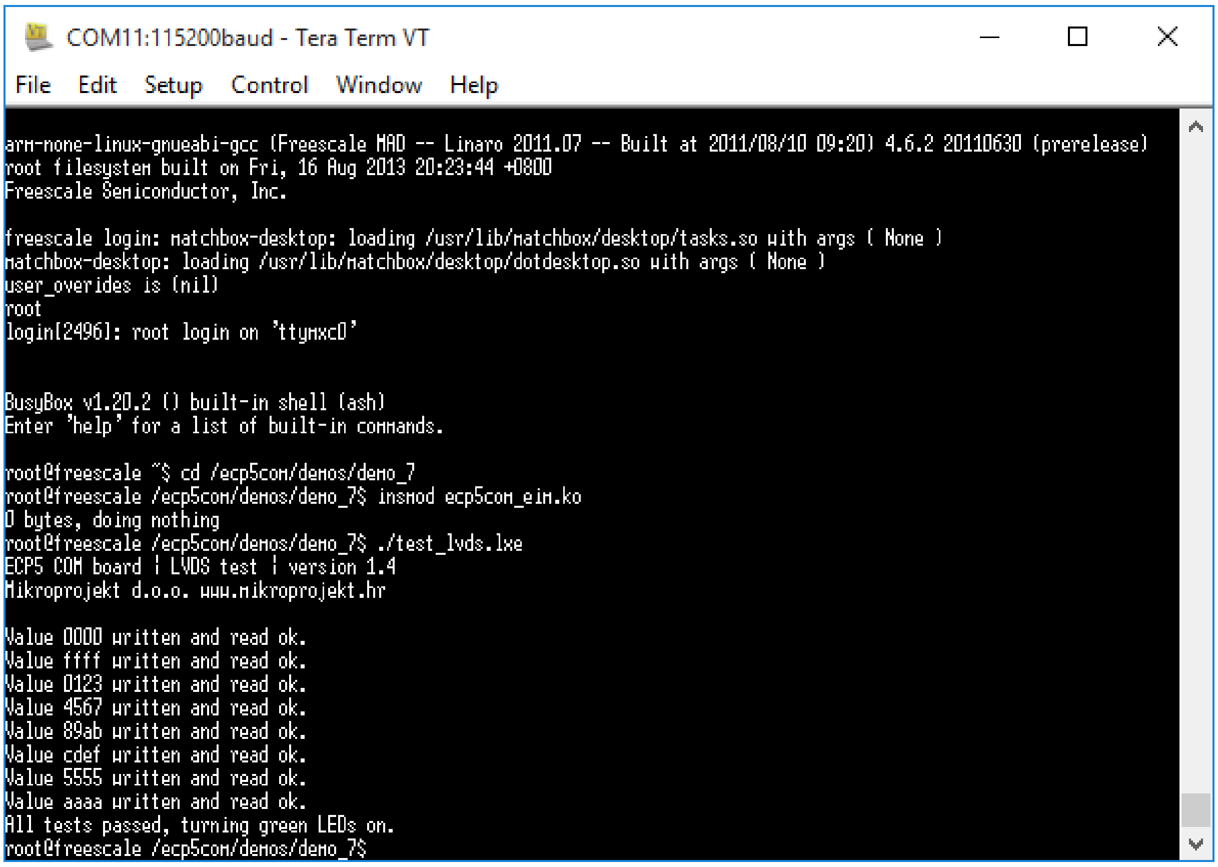
```
> insmod ecp5com_eim.ko
```

3.7.4 Test

Start test application test_lvds.lxe using:

```
> ./test_lvds.lxe
```

It will try sending several values over EIM and over LVDS and receiving them back. In case everything goes well, two green LEDs will be lit, otherwise two red LEDs will be lit.


COM11:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
arm-none-linux-gnueabi-gcc (Freescale MAD -- Linaro 2011.07 -- Built at 2011/08/10 09:20) 4.6.2 20110630 (prerelease)
root filesystem built on Fri, 16 Aug 2013 20:23:44 +0800
Freescale Semiconductor, Inc.

freescale login: matchbox-desktop: loading /usr/lib/matchbox/desktop/tasks.so with args ( None )
matchbox-desktop: loading /usr/lib/matchbox/desktop/dotdesktop.so with args ( None )
user_overides is (nil)
root
login[2496]: root login on 'ttymxc0'


BusyBox v1.20.2 () built-in shell (ash)
Enter 'help' for a list of built-in commands.

root@freescale ~$ cd /ecp5com/demos/demo_7
root@freescale /ecp5com/demos/demo_7$ insmod ecp5com_eim.ko
0 bytes, doing nothing
root@freescale /ecp5com/demos/demo_7$ ./test_lvds.lxe
ECP5 COM board | LVDS test | version 1.4
Mikroprojekt d.o.o. www.mikroprojekt.hr

Value 0000 written and read ok.
Value ffff written and read ok.
Value 0123 written and read ok.
Value 4567 written and read ok.
Value 89ab written and read ok.
Value cdef written and read ok.
Value 5555 written and read ok.
Value aaaa written and read ok.
All tests passed, turning green LEDs on.
root@freescale /ecp5com/demos/demo_7$
```

Figure 9: Loopback LVDS, verify connectivity using LEDs

3.8 Demo 8 / RD0018 - UART and I2C communication between ARM and ECP5

For this demo, use files from /ecp5com/demos/demo_8 folder.

3.8.1 Hardware

This demo does not have any special hardware prerequisites.

3.8.2 Design

File arm_sdk_i2c_ram.bit

FPGA design **ecp5com_uart_i2c** should be used; program it using any method from Downloading FPGA designs chapter. Both, Slave SPI (.sea and .sed files) and JTAG (.vme) binaries are already provided in i.MX6 file system. Load Slave SPI binaries using following:

```
> cd /epc5com/demos/demo_8

> ../../sspiem.lxe ui_algo.sea ui_data.sed
```

ARM does not need to be rebooted afterwards.

3.8.3 Driver

Compiled driver binary file ecp5com_eim.ko is already provided. As this demo doesn't require reboot, there is a possibility some other driver is already loaded. Check it with command:

```
> lsmod
```

If there is one loaded, unload it first using:

```
> rmmod <name of file>
```

(see example of command in chapter 3.3.3).

Driver for this demo should be loaded using:

```
> insmod ecp5com_eim.ko
```

3.8.4 UART

Start compiled binary:

```
> ./test_uart.lxe
```

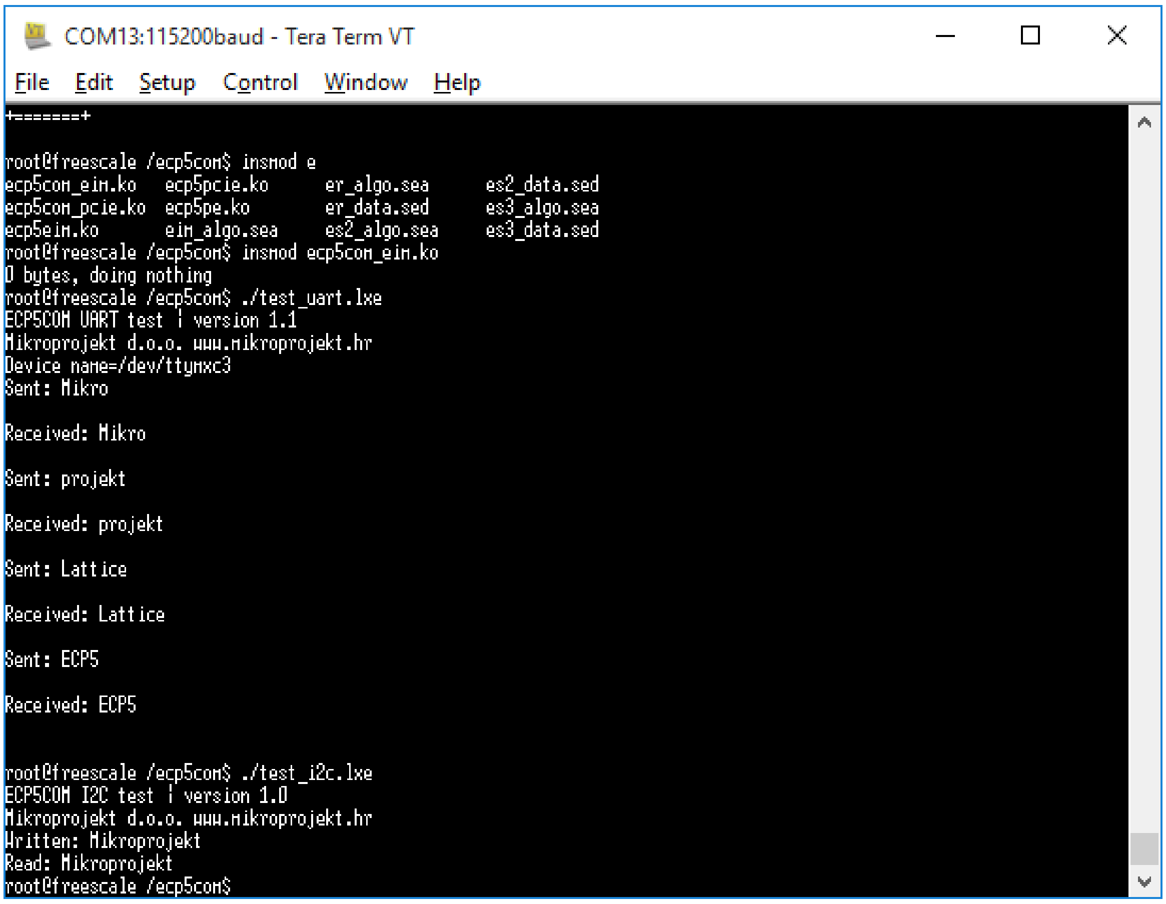
It will send several strings over one of ARM's serial port connected to the FPGA; the FPGA will receive them and send them back; finally ARM will print out the strings it received (result shown along with I2C test).

3.8.5 I2C

Start compiled binary:

```
> ./test_i2c.lxe
```

It sends one string to an I2C device on I2C bus shared between CPU and FPGA, and reads it back.


COM13:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
+=======+

root@freescale /ecp5com$ insmod e
ecp5com_eim.ko   ecp5pcie.ko      er_algo.sea      es2_data.sed
ecp5com_pcie.ko  ecp5pe.ko        er_data.sed      es3_algo.sea
ecp5eim.ko       eim_algo.sea     es2_algo.sea     es3_data.sed
root@freescale /ecp5com$ insmod ecp5com_eim.ko
0 bytes, doing nothing
root@freescale /ecp5com$ ./test_uart.lxe
ECP5COM UART test | version 1.1
Mikroprojekt d.o.o. www.mikroprojekt.hr
Device name=/dev/ttymxc3
Sent: Mikro

Received: Mikro

Sent: projekt

Received: projekt

Sent: Lattice

Received: Lattice

Sent: ECP5

Received: ECP5


root@freescale /ecp5com$ ./test_i2c.lxe
ECP5COM I2C test | version 1.0
Mikroprojekt d.o.o. www.mikroprojekt.hr
Written: Mikroprojekt
Read: Mikroprojekt
root@freescale /ecp5com$
```

Figure 10: UART and I2C communication between ARM and ECP5

Note: since Lattice I2C Slave IP Core is not supported on ECP5 device to our knowledge, we have provided a working design suitable for this test by Mikroprojekt.

Terms of use

Contact info

Mikroprojekt d.o.o.

Aleja Blaža Jurišića 9,
HR-10040 Zagreb, Croatia
T/F: +385 1 2455 659

contact@mikroprojekt.hr
http://www.mikroprojekt.hr