Microblaze Linux

Using an FPGA-based processor is:

- **Very intelligent** ☐
- **Very stupid** ☐
- **Don't know** ☐

Microblaze Linux

Using an FPGA-based processor is:

- Very intelligent √
- Very stupid √
- Don't know □

Leibniz: ***The current world is the best one possible***

... Let's improve it

Contents

Why?

- Processing power typically not an issue ...

- Test software
- Communication
- µC replacement

Why Linux?

- Linux widespread on standard server and desktop PC
  - Many developers familiar with tools and system
  - Robust system
  - Various distributions (Ubuntu, RedHat, SUSE, ...)
- Opensource kernel and tools
  - Already ported to many architectures
    - Standard kernel supports Microblaze (since 2.6.36.2)
  - Busybox toolset *(swiss-army knife of embedded systems)*
- Customisable (e.g. no graphics and disk) for small footprint
  - Special version (ucLinux) for non-MMU systems
  - We use the standard MMU kernel
- Loadable drivers (kernel modules)
- Built-in multi-threading, multi-processing
  - Preemptive kernel (optional)
  - Multiple task prioritites
- Networking
  - Simplifies development process (multiple terminals, ftp)
  - Time, WWW, mail, ...

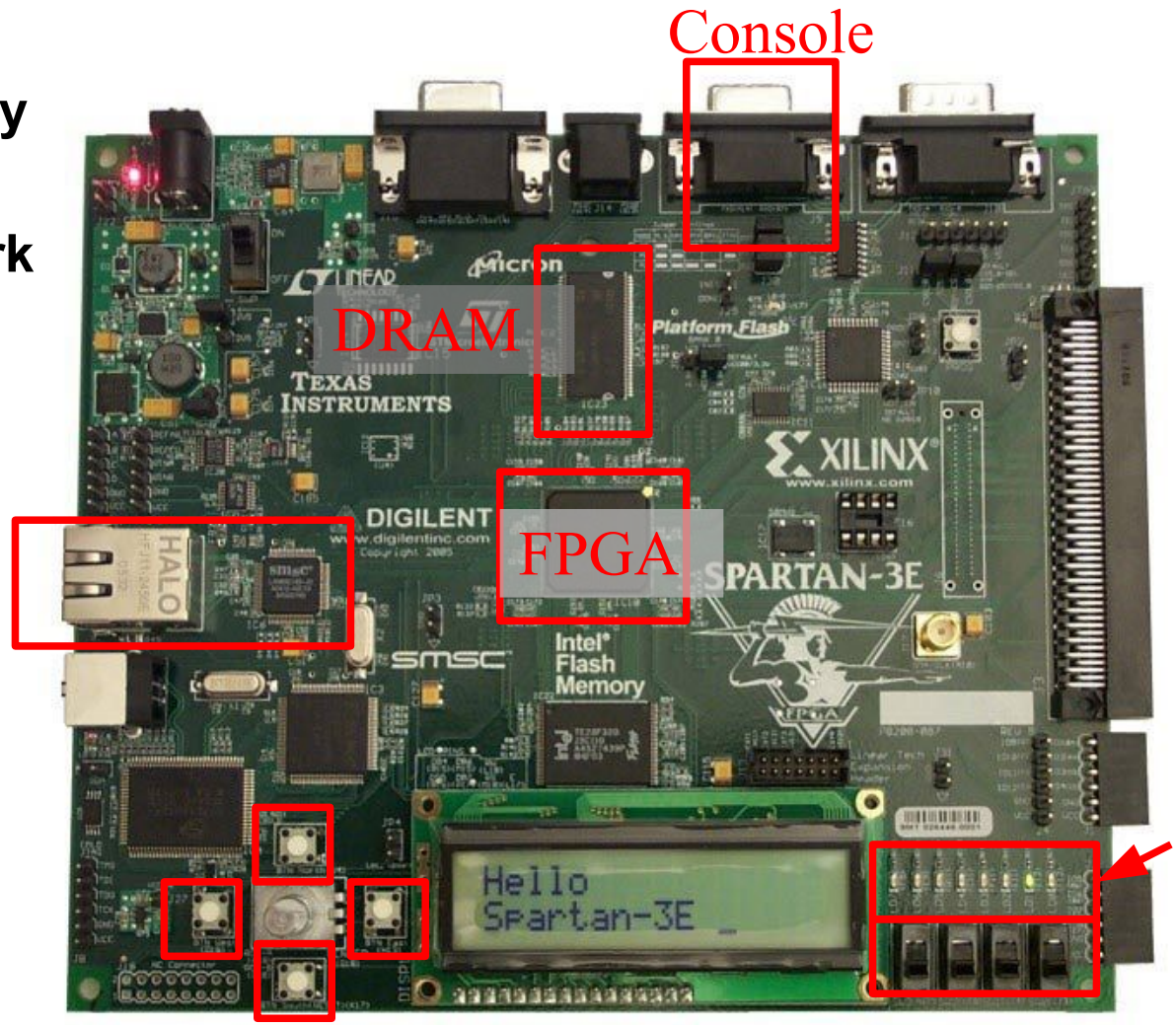
Target HW

- All modern FPGAs can host a processor
  - Openrisc etc
  - Nios (Altera), Microblaze (Xilinx)
  - Hard-IP: Arm (Altera), PowerPC (Xilinx)

- Minimal FPGA design (Mblaze, DRAM, RS232, Network, GPIO)
  - Spartan3, Spartan6, Virtex4, Virtex5, Virtex6
  - Mblaze speed 50 .. 125MHz
  - 32MB DRAM minimum?
  - Access to custom hardware (= FPGA fabric) via GPIO, EPC, FSL possible.

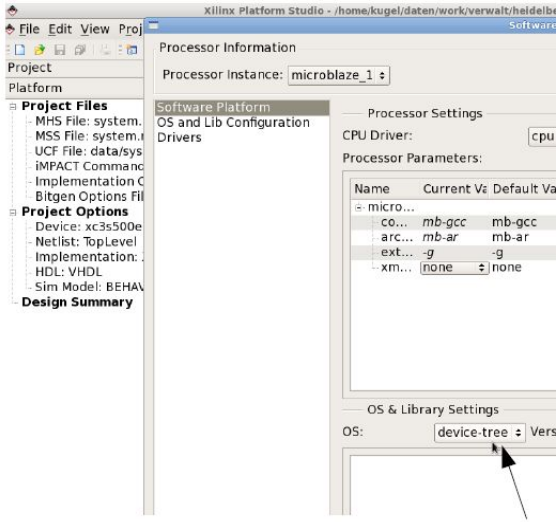
Test case: Linux on XILINX starter kit

- Elements
  - CPU
  - Memory
  - UART
  - Network
  - GPIO



Prerequisites (1)

- Hardware platform
  - ⇨ Define in EDK
  - ⇨ Minimal set of elements for reasonable system
    - Microblaze CPU with MMU and cache
    - DRAM memory (64MB)
    - Linux console (Uart lite with IRQ)
    - Dual timer (with IRQ)
    - Network (ethernet lite with IRQ)
    - GPIO for LEDs
    - GPIO for buttons and switches
    - Interrupt controller
    - Debug module (MDM)
  - ⇨ Set DRAM base address to 0xc0000000
  - ⇨ Select „device-tree“ in SW platform
  - ⇨ Generate addresses
  - ⇨ Compile libraries
    - *Don't select any app for BRAM init*
    - Produces device tree directory



Prerequisites (2)

- Xilinx tools
  - ⇨ Install ISE and EDK
  - ⇨ Gcc (mb-gcc) to compile kernel comes with EDK
  - ⇨ Toolset for Linux applications from Xilinx GIT server
    - http://git.xilinx.com/?p=mb_gnu.git;a=summary
      - Microblaze-unknown-linux-gnu-gcc (etc.)
      - Sample ram disk images
      - System include files and libraries for Linux
  - ⇨ Device-tree-generator for EDK from Xilinx GIT server
    - http://git.xilinx.com/?p=device-tree.git;a=summary
      - „bsp“ directory must be copied to root of EDK project
- Many useful Xilinx infos
  - ⇨ http://xilinx.wikidot.com
  - ⇨ Xilinx-version of Linux kernel available
    - Few more drivers and configurations
    - Less portable

Prerequisites (3)

- Download Linux application(s): Busybox
  - http://busybox.net/ (current version 1.18.1)
  - Busybox contains almost all basic Linux programs in a single binary (some with limited functionality)
- Configure Busybox
  - Copy arch/i386 to arch/microblaze
  - Edit arch/microblaze/Makefile
    - Add to CFLAGS: -L . -L *mbtooldir*/microblaze_v1.0/microblaze-unknown-linux-gnu/microblaze-unknown-linux-gnu/sys-root/usr/lib --sysroot=*linuxrootdir* -isystem *mbtooldir*/microblaze_v1.0/microblaze-unknown-linux-gnu/microblaze-unknown-linux-gnu/sys-root/usr/include
  - Add path to Linux GCC
    - export PATH=$PATH:*mbtooldir*/microblaze_v1.0/microblaze-unknown-linux-gnu/bin
  - Copy *mbtooldir/*microblaze_v1.0/microblaze-unknown-linux-gnu/microblaze-unknown-linux-gnu/sys-root/usr/lib/crt*.o to .
  - Make ARCH=microblaze CROSS_COMPILE=microblaze-unknown-linux-gnu- ; make install
    - Directory structure with busybox binary and soft links in _install/

- Busybox files

- Init
- bin:
- [cryptpw fgrep length od seq traceroute [[cttyhack find less openvt setarch true addgroup cut fold linux32 passwd setkeycodes tty adduser date free linux64 patch setsid ttysize ar dc ftpget ln pgrep setuidgid udpsvd arping dd ftpput logger pidof sh umount ash deallocvt fuser login ping sha1sum uname awk delgroup getopt logname ping6 showkey uncompress basename deluser grep lpq pipe_progress sleep unexpand bbconfig df gunzip lpr pkill softlimit uniq bunzip2 diff gzip ls printenv sort unix2dos busybox dirname hd lsattr printf split unlzma bzcat dmesg head lzmacat ps stat unzip bzip2 dos2unix hexdump makemime pscan strings uptime cal dpkg hostid md5sum pwd stty usleep cat dpkg-deb hostname mesg readlink su uudecode catv du id microcom realpath sum uuencode chat dumpkmap install mkdir reformime sv vi chattr dumpleases ip mkfifo renice sync vlock chgrp echo ipaddr mknod reset tac watch chmod ed ipcalc mktemp resize tail wc chown egrep ipcrm more rm tar wget chpst eject ipcs mount rmdir tcpsvd which chrt env iplink mountpoint rpm tee who chvt envdir iproute mt rpm2cpio telnet whoami cksum envuidgid iprule mv rtcwake test xargs clear ether-wake iptunnel nc run-parts tftp yes cmp expand kbd_mode netstat runsv tftpd zcat comm expr kill nice runsvdir time cp false killall nmeter rx top cpio fdflush killall5 nohup script touch crontab fdformat last nslookup sed tr
- sbin:
- adjtimex dnsd hdparm loadfont nameif runlevel swapon arp fakeidentd httpd loadkmap pivot_root sendmail switch_root blkid fbset hwclock logread popmaildir setconsole sysctl brctl fbsplash ifconfig losetup poweroff setfont syslogd chpasswd findfs ifdown lpd raidautorun setlogcons telnetd chroot freeramdisk ifenslave makedevs rdate slattach udhcpc crond fsck ifup man rdev start-stop-daemon udhcpd devfsd fsck.minix inetd mdev readprofile sulogin vconfig devmem getty init mkfs.minix reboot svlogd watchdog dhcprelay halt klogd mkswap route swapoff zcip

Prerequisites (4)

- Create directory for initial ram disk, e.g. /tmp/ramdisk
- Extract template
  - cd /tmp/
  - gzip -d -c *mbtooldir*/microblaze_v1.0/initramfs_minimal.cpio.gz > ramfs.cpio
  - cd ramdisk
  - sudo cpio -i < ../ramfs.cpio
    - > ls
    - bin dev etc *init* mnt proc sbin sys tmp var
    - >

Start program

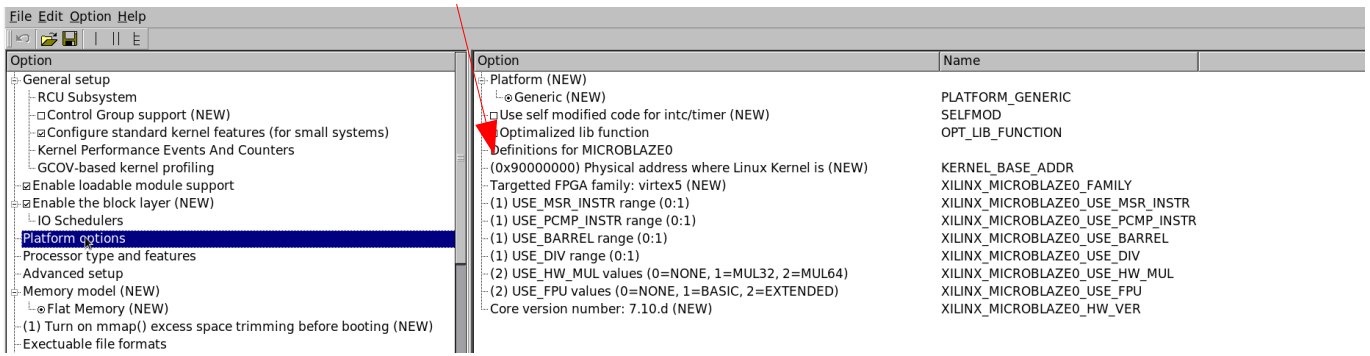
- Replace template binaries with new Busybox
  - sudo rm -rf bin sbin
  - sudo cp -r *busyboxdir*/_install/* .
- Set root owner of new binaries
  - sudo chown -R 0.0 init bin sbin
- Add user files, etc to /lib/modules, /src, /var/www
- Edit startup script /etc/init.d/rcS
  - Next slide

- Startup script /etc/init.d/rcS

```
#!/bin/sh
/bin/echo "Starting rcS"
/bin/echo "++ Creating device points"
/bin/mkdir /dev/pts
/bin/mount -t devpts devpts /dev/pts
/bin/echo "++ Mounting filesystem"
/bin/mount -t proc none /proc
/bin/mount -t sysfs none /sys
/bin/echo "++ Loading system loggers"
/sbin/syslogd
/sbin/klogd
/bin/echo "++ Starting telnet daemon"
/sbin/telnetd -l /bin/sh
/bin/echo "++ Starting network at 192.168.0.10"
/sbin/ifconfig eth0 192.168.0.10 up
/bin/echo "++ Starting HTTPD"
/sbin/httpd -h /var/www
/bin/echo "++ Creating user ftp"
/bin/cat /dev/null >> /etc/passwd
/bin/cat /dev/null >> /etc/group
/bin/adduser -D -H -h /tmp ftp
/bin/echo "++ Starting INETD (e.g.for ftp)"
/sbin/inetd -e /etc/inetd/inetd.conf
/sbin/route add 192.53.103.108 gw 192.168.0.1 eth0
/sbin/rdate 192.53.103.108
/bin/echo "rcS Complete"
```

Prerequisites (5)

- **Download Linux kernel**
  - ⇨ **http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.38.2.tar.bz2**
- **Configure kernel**
  - ⇨ **Copy arch/microblaze/configs/mmu_defconfig to .config**
  - ⇨ **Make ARCH=microblaze CROSS_COMPILE=mb- xconfig**
    - **Shows configuration options**
    - **Set /tmp/ramfs as directory for intial ram disk**
    - **Set memory to 0xc0000000 (problem with current linker script)**



Check .config after saving against tutorial .config to see all differences

  - ⇨ **Copy device tree from EDK microblaze0/libsrc/device-tree/xilinx.dts to arch/microblaze/boot/dts**
  - ⇨ **Make ARCH=microblaze CROSS_COMPILE=mb- simpleImage.xilinx**
    - **Kernel binary in arch/microblaze/boot/simpleImage.xilinx**

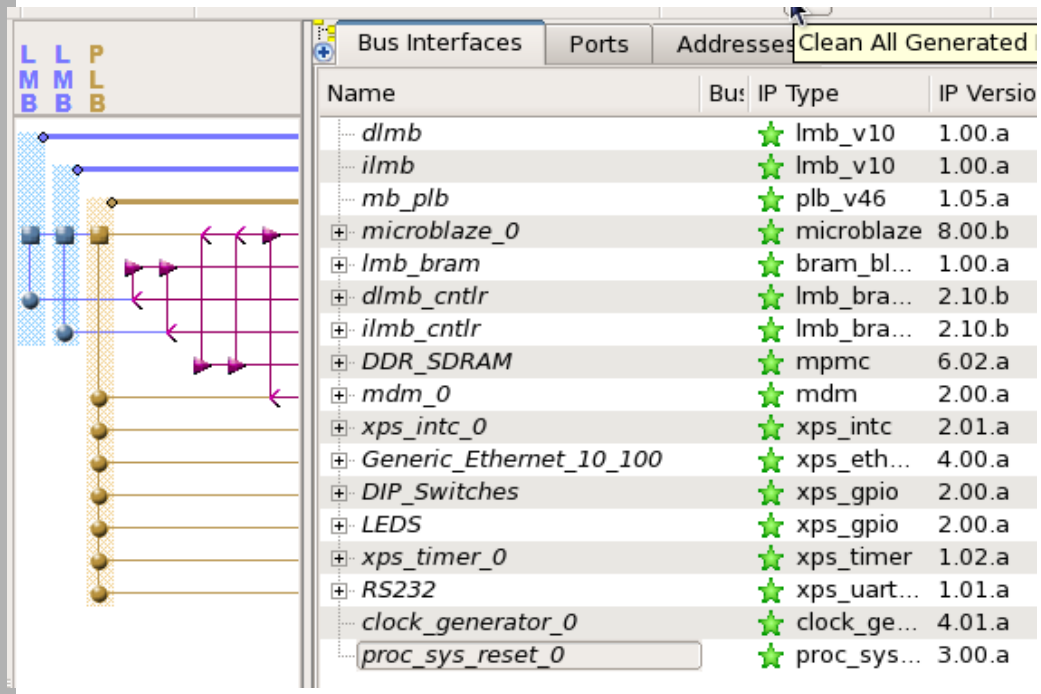
Prerequisites (6)

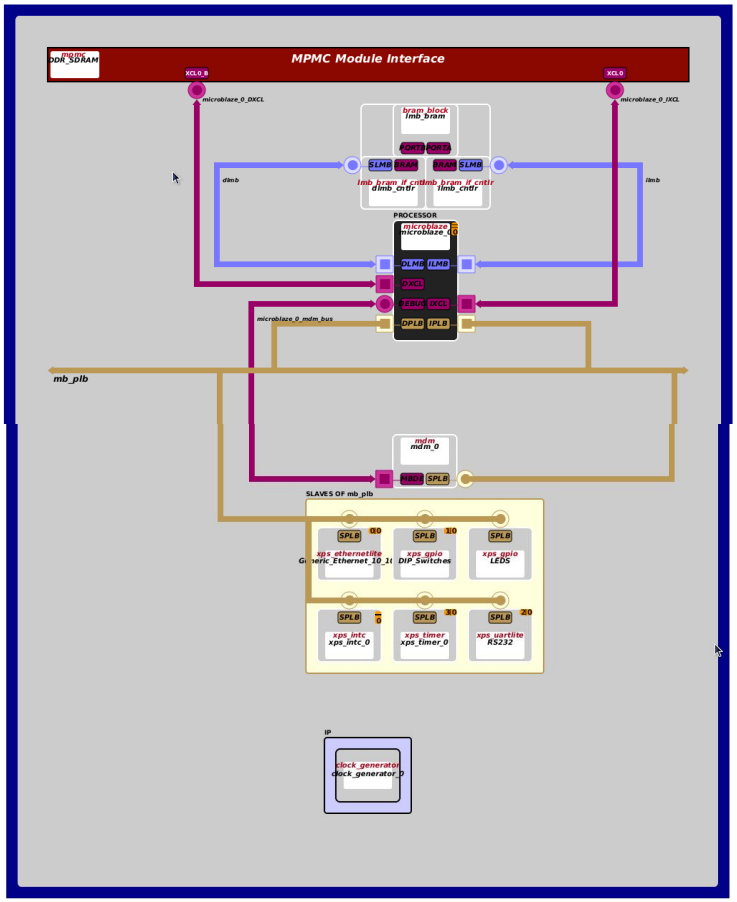
- More on kernel configuration
  - MMU enabled
  - XILINX uartlite (+ console), ethernetlite, GPIO enabled
  - Preemptive kernel enabled
  - Microblaze options and versions must match EDK
    - MMU: 3
    - Barrel shifter: 1, HW mult: 1 (32 bit), MSR: 1, PatCmp: 1
    - Divider: 0, Float: 0
  - DRAM address can be set in config dialog but seems to be ignored in linker script => use 0xc0000000
  - GPIO enabled

Prerequisites (7)

- Kernel compilation result
  - ...
  - SYSMAP System.map
  - SYSMAP .tmp_System.map
  - CP vmlinux arch/microblaze/boot/simpleImage.xilinx_sp3.unstrip
  - OBJCOPY arch/microblaze/boot/simpleImage.xilinx_sp3
  - UIMAGE arch/microblaze/boot/simpleImage.xilinx_sp3.ub
  - Image Name: Linux-2.6.36.2
  - Created: Sun Jan 9 18:29:22 2011
  - Image Type: MicroBlaze Linux Kernel Image (uncompressed)
  - Data Size: 5172900 Bytes = 5051.66 kB = 4.93 MB
  - Load Address: c0000000
  - Entry Point: c0000000
  - STRIP arch/microblaze/boot/simpleImage.xilinx_sp3
  - Kernel: arch/microblaze/boot/simpleImage.xilinx_sp3 is ready (#31)
  - [kugel@akudesk linux-2.6.36.2]$

Creating the HW





- Microblaze configuration

XPS Core Config - microblaze_0 - microblaze_v8_00_b

MicroBlaze™ Configuration Wizard

Select configuration:

- Current Settings
- Minimum Area
- Maximum Performance
- Maximum Frequency
- Linux with MMU
- Low-end Linux with MMU
- Typical

Welcome to MicroBlaze Configuration Wizard

- Select a predefined configuration in the list to the left. Information about the selected configuration is shown below. *Each predefined configuration completely changes the MicroBlaze parameters.*
- To modify the configuration, click on the *Next* button, click on *Advanced* to directly access parameters in a tabbed interface, or click on *OK* to accept the configuration and close the dialog.

- [ ] Select implementation to optimize area (with lower instruction throughput)
- [x] Enable Debug
- [x] Use Instruction and Data Caches
- [ ] Enable Exceptions
- [x] Use Memory Management

Enable Memory Management if planning to use an operating system - such as Linux - with support for virtual memory or memory protection.

Note that when area optimized MicroBlaze is enabled, the Memory Management Unit is not available.

Advanced

Next >

Frequency

Area

Performance

BRAM

MULT18

- Microblaze configuration

XPS Core Config - microblaze_0 - microblaze_v8_00_b

Page 2 of 6 - Performance and Instructions

Optimization

Enable Branch Target Cache

Branch Target Cache Size: DEFAULT

Instructions

Enable Barrel Shifter

Enable Floating Point Unit: NONE

Enable Integer Multiplier: MUL32

Enable Integer Divider

Enable Additional Machine Status Register Instructions

Enable Pattern Comparator

Advanced | < Back | Next >

Frequency

Area

Performance

BRAM

MULT18

- Microblaze configuration

XPS Core Config - microblaze_0 - microblaze_v8_00_b

Page 4 of 6 - Caches

Instruction Cache Feature

Size of the Instruction Cache in Bytes	256B
Instruction Cache Line Length	4 words
Instruction Cache Base Address	0xc0000000
Instruction Cache High Address	0xc3ffffff
Use Cache Links for All I-Cache Memory Accesses	☑
Number of I-Cache Streams	0
Number of I-Cache Victims	0

Data Cache Feature

Size of the Data Cache in Bytes	256B
Data Cache Line Length	4 words
Data Cache Base Address	0xc0000000
Data Cache High Address	0xc3ffffff
Use Cache Links for All D-Cache Memory Accesses	☑
Enable Write-back Storage Policy	☐
Number of D-Cache Victims	0

Advanced

< Back Next >

Frequency

Area

Performance

BRAM

MULT18

- Microblaze configuration

XPS Core Config - microblaze_0 - microblaze_v8_00_b

Page 5 of 6 - Memory Management Unit

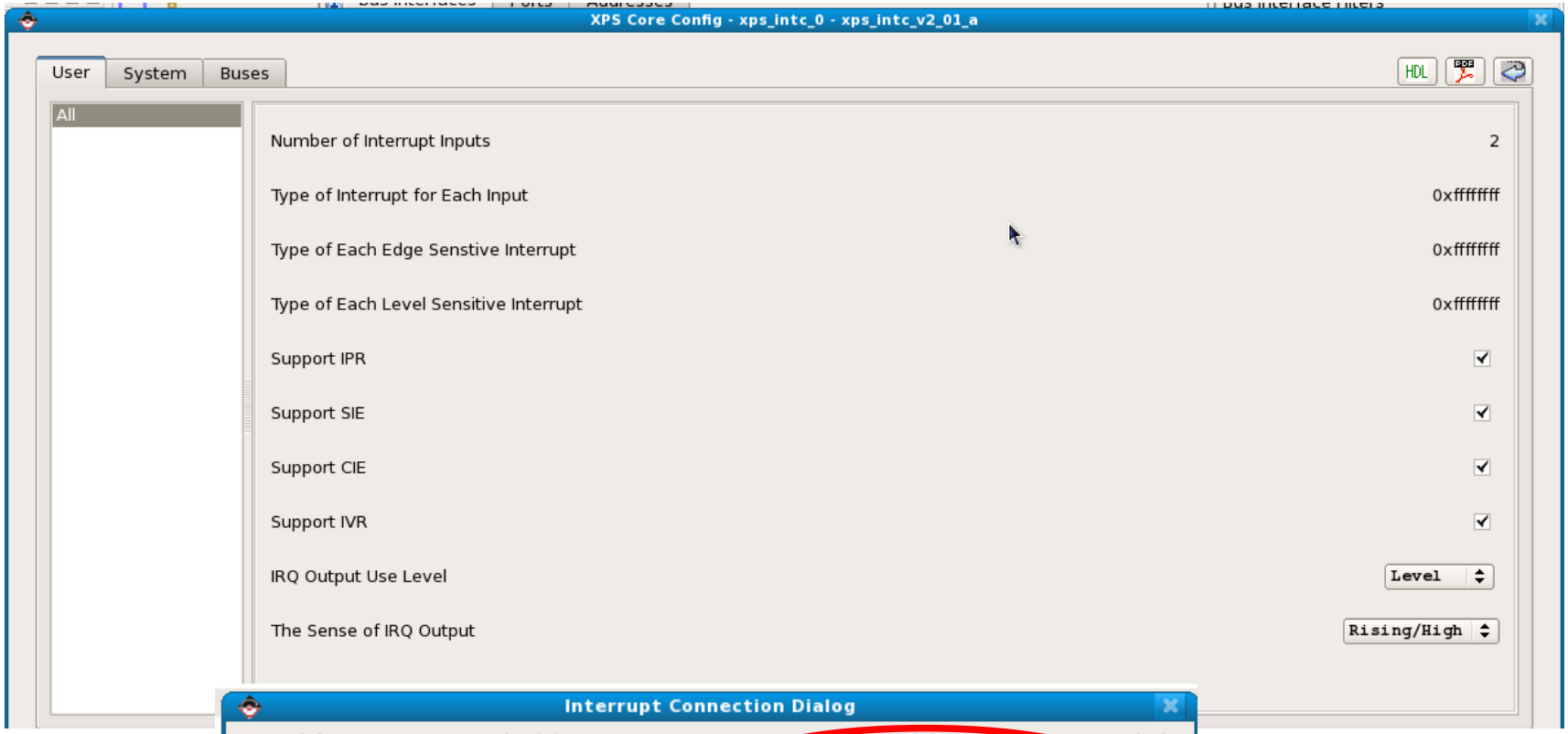
Memory Management	VIRTUAL
Data Shadow Translation Look Aside Buffer Size	4
Instruction Shadow Translation Look Aside Buffer Size	2
Enable Access to Memory Management Registers	FULL
Number of Memory Protection Zones	2

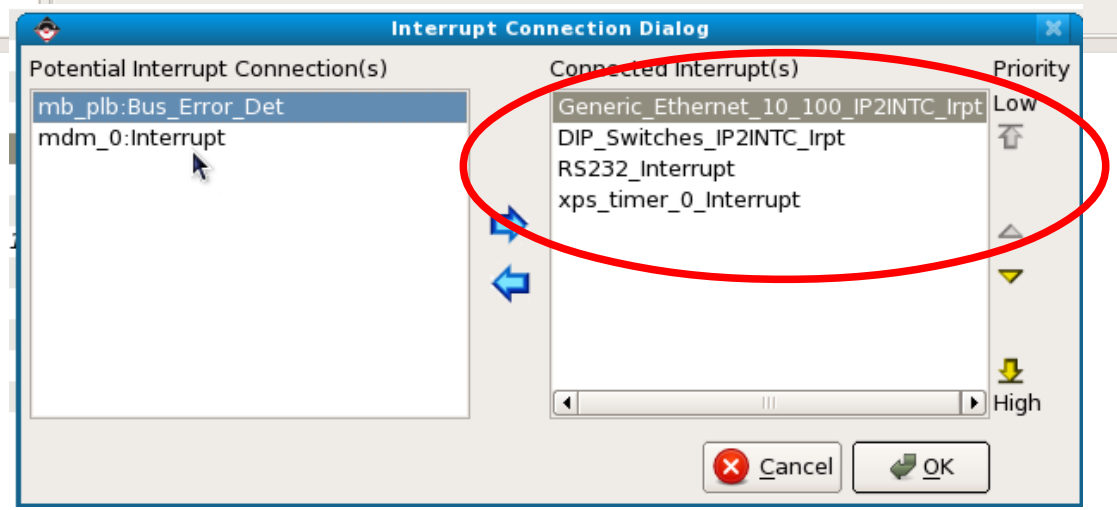
Advanced < Back Next >

Frequency
Area
Performance

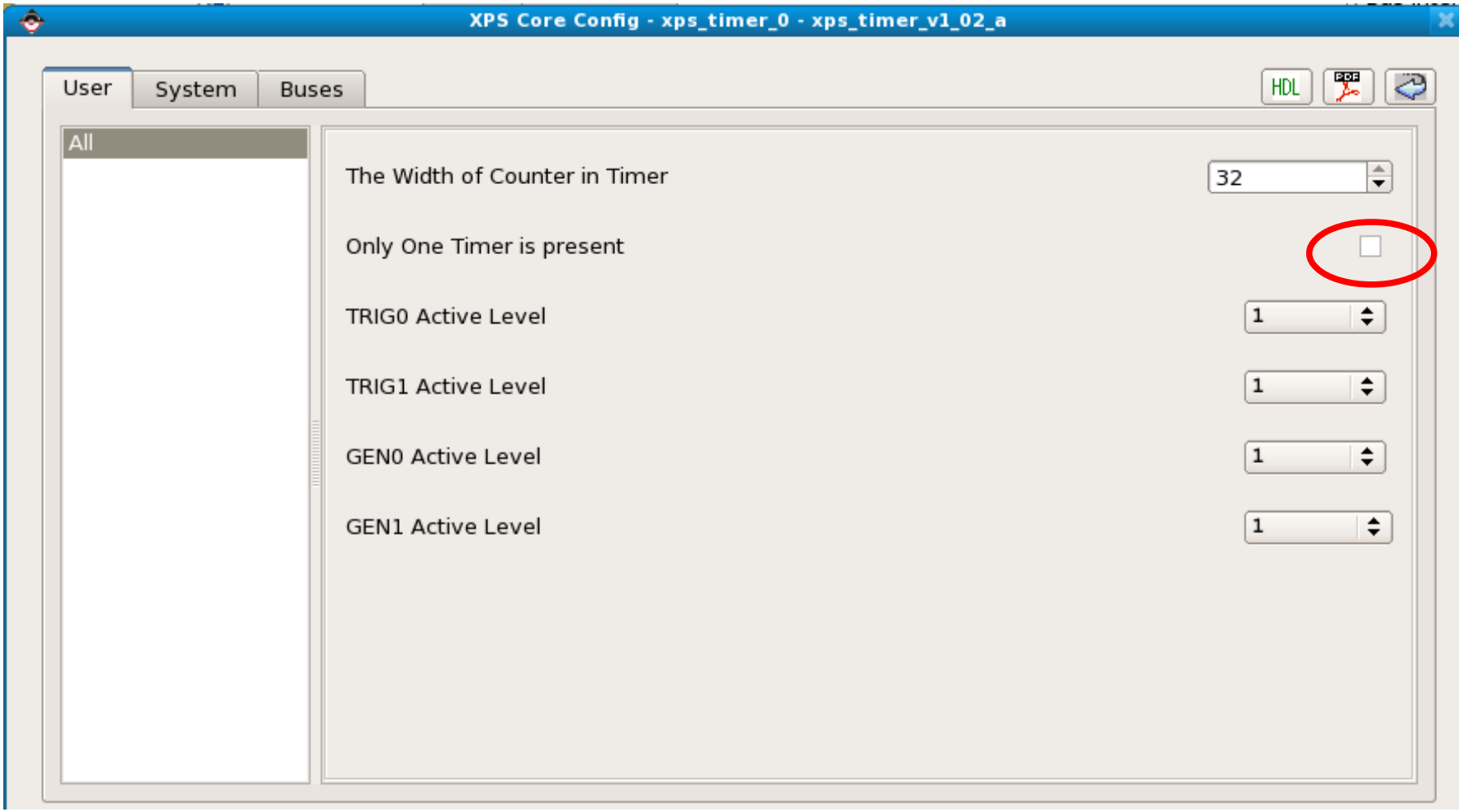
BRAM
MULT18

INTC

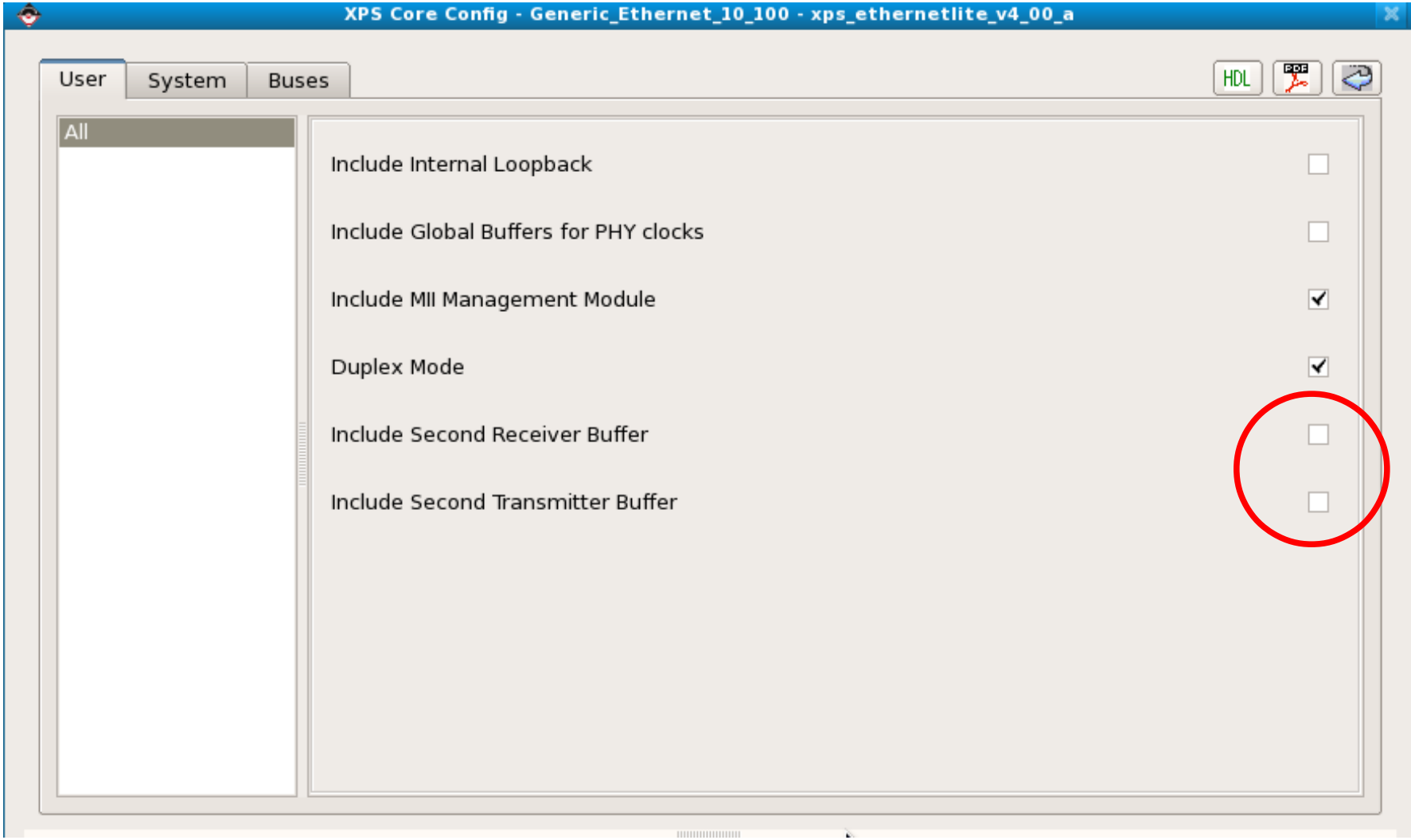

XPS Core Config - xps_intc_0 - xps_intc_v2_01_a
User
System
Buses
HDL
All
Number of Interrupt Inputs
2
Type of Interrupt for Each Input
0xffffffff
Type of Each Edge Senstive Interrupt
0xffffffff
Type of Each Level Sensitive Interrupt
0xffffffff
Support IPR
Support SIE
Support CIE
Support IVR
IRQ Output Use Level
Level
The Sense of IRQ Output
Rising/High


Interrupt Connection Dialog
Potential Interrupt Connection(s)
mb_plb:Bus_Error_Det
mdm_0:Interrupt
Connected Interrupt(s)
Generic_Ethernet_10_100_IP2INTC_Irpt
DIP_Switches_IP2INTC_Irpt
RS232_Interrupt
xps_timer_0_Interrupt
Priority
Low
High
Cancel
OK

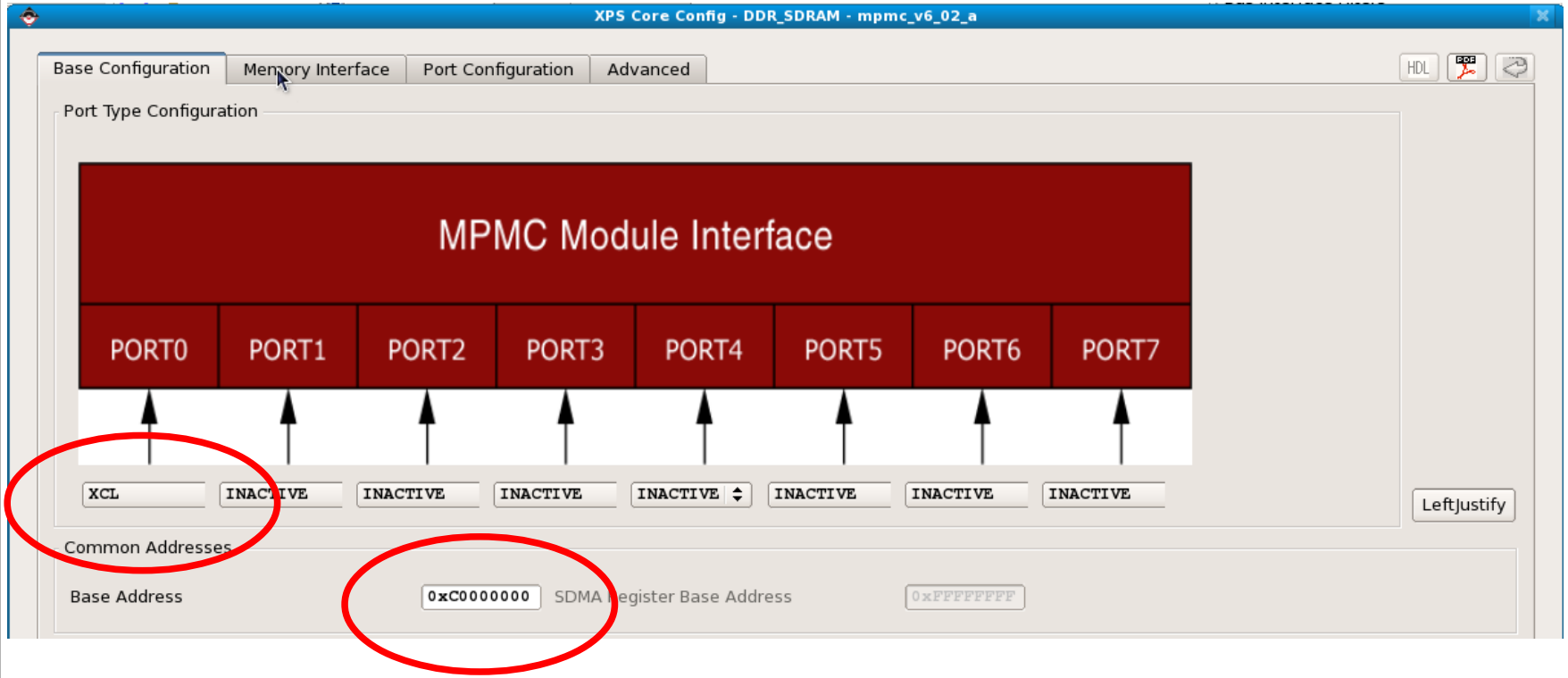
- Timer


XPS Core Config - xps_timer_0 - xps_timer_v1_02_a
User
System
Buses
HDL
All
The Width of Counter in Timer
32
Only One Timer is present
TRIG0 Active Level
1
TRIG1 Active Level
1
GEN0 Active Level
1
GEN1 Active Level
1

- Network


XPS Core Config - Generic_Ethernet_10_100 - xps_ethernetlite_v4_00_a
User
System
Buses
HDL
All
Include Internal Loopback
Include Global Buffers for PHY clocks
Include MII Management Module
Duplex Mode
Include Second Receiver Buffer
Include Second Transmitter Buffer

- Memory (MPMC)


XPS Core Config - DDR_SDRAM - mpmc_v6_02_a
Base Configuration
Memory Interface
Port Configuration
Advanced
HDL
Port Type Configuration
MPMC Module Interface
PORT0
PORT1
PORT2
PORT3
PORT4
PORT5
PORT6
PORT7
XCL
INACTIVE
INACTIVE
INACTIVE
INACTIVE
INACTIVE
INACTIVE
INACTIVE
LeftJustify
Common Addresses
Base Address
0xC0000000
SDMA Register Base Address
0xFFFFFFFF

- Memory (MPMC)

XPS Core Config - DDR_SDRAM - mpmc_v6_02_a

Base Configuration | Memory Interface | Port Configuration | Advanced

HDL

Memory Part Selector

Type: DDR | Manufacturer: * | Style: * | Density: * | Width: * | Part No.: Select a part (209)

Selected Memory Info

Part No. MT46V32M16-6 Size 64MB Type DDR Base Address 0xC0000000 High Address 0xC3FFFFFF

Memory/DIMM Settings | Memory Part Settings | MIG Settings | MCB

Settings

Number of DIMMs	1	Dynamic Write ODT Setting	OFF
Memory Data Width	16	Partial Array Self Refresh	FULL
ODT Setting	Disabled/Disable	Auto Self Refresh	ENABLED
Enable DQSN in DDR2	☑	High Temp Self Refresh	NORMAL
Reduced Drive Output	FULL	X Memory Clock Period (ps)	10,000
Enable Write Leveling	☑		

Configuration

X CE Width	1
X ODT Width	1
X Clock Width	1
X CSn Width	1
X No. of Ranks	1
X Registered Memory	☐

Info

Memory DM Width	2
Memory Address Width	13
Memory Bank Address Width	2
Memory DQS Width	2
CAS Write Latency	5

- Device tree generator

- Device tree generator

Software Platform Settings -- Deprecated

Processor Information

Processor Instance: microblaze_0

Software Platform
OS and Lib Configuration
Drivers

Processor Settings

CPU Driver: cpu CPU Driver Version: 1.13.a

Processor Parameters:

Name	Current Value	Default Value	Type	Description
microbla...				
compi...	*mb-gcc*	mb-gcc	string	Compiler used to compile bo...
archiver	*mb-ar*	mb-ar	string	Archiver used to archive libr...
extra...	*-g*	-g	string	Extra compiler flags used in ...
xmds...	mdm_0	none	peripheral_i...	Debug peripheral to be used ...

Visible only when bsp subdir present

OS & Library Settings

OS: device-tree Version: 0.00.x Generate flat device tree

Software Platform Settings -- Deprecated

Processor Information

Processor Instance: microblaze_0

Console: /ttyUL0 -> kernel config!

Software Platform
OS and Lib Configuration
Drivers

Configuration for OS: device-tree v0.00.x

Name	Current Value	Default Value	Type
device-tree			
bootargs	*console=ttyUL0 root=/dev/ram ip=192.168.0.10*	console=tty...	string
console device			string

- HW implementation result

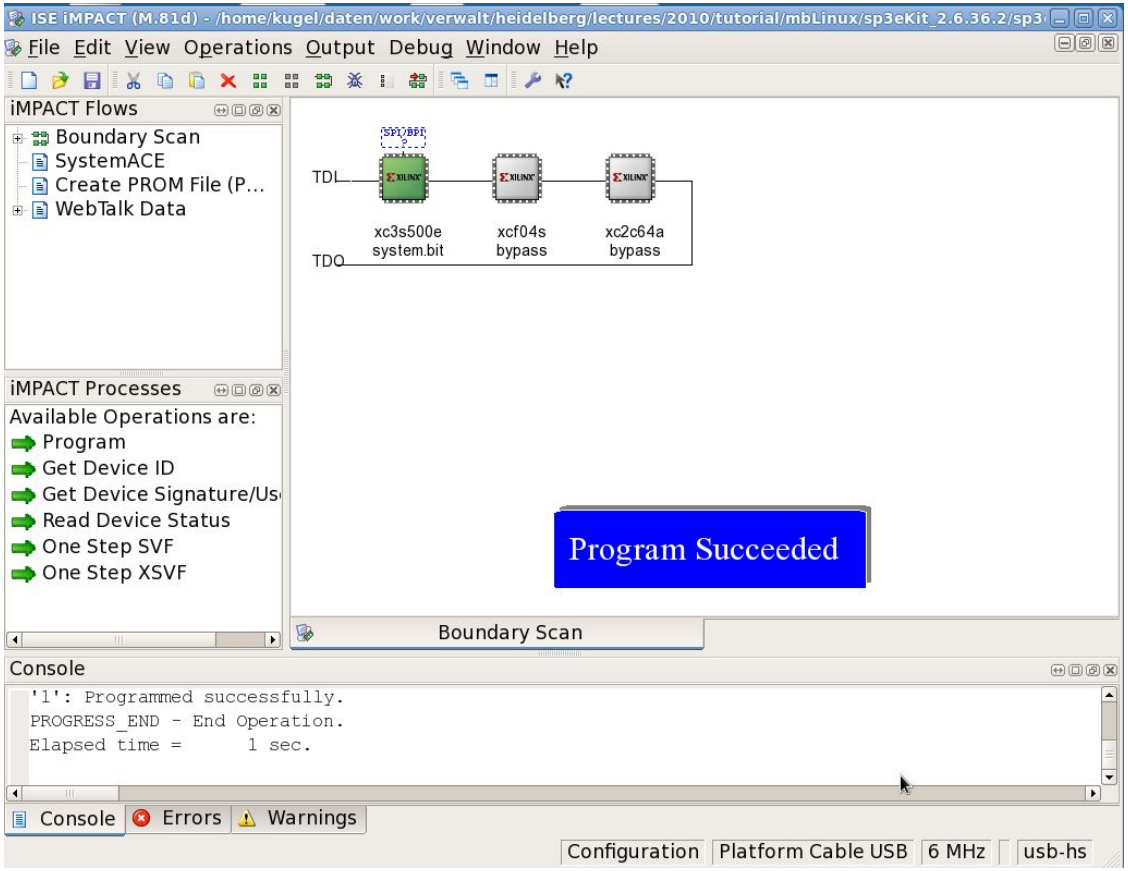
```
Design Summary Report (XC3S500e):

 Number of External IOBs                          79 out of 232    34%
   Number of External Input IOBs                 19
      Number of External Input IBUFs             19
   Number of External Output IOBs                40
      Number of External Output DIFFMs            1
      Number of External Output DIFFSs            1
      Number of External Output IOBs             38
   Number of External Bidir IOBs                 20
      Number of External Bidir IOBs              20
   Number of BSCANs                            1 out of 1     100%
   Number of BUFGMUXs                          5 out of 24     20%
   Number of DCMs                              2 out of 4      50%
   Number of MULT18X18SIOs                     3 out of 20     15%
   Number of RAMB16s                          10 out of 20     50%
   Number of Slices                         4612 out of 4656   99%
      Number of SLICEMs                      777 out of 2328   33%
```

- **No space left in device**

Download to Starterkit (FPGA config)

- Download FPGA configuration
- Sonnect JTAG cable
- Start Impact
- Select .bit file for FPGA, no SPI, other devices in BYPASS



Download to Starterkit (XMD connect)

Download kernel via XMD

```
[kugel@pcakulap sp3eKit_2.6.36.2]$ xmd
Xilinx Microprocessor Debugger (XMD) Engine
Xilinx EDK 12.4 Build EDK_MS4.81d
XMD%
XMD% connect mb mdm
JTAG chain configuration
-----------------------------------------------
-
Device    ID Code        IR Length    Part Name
 1        41c22093           6        XC3S500E
 2        f5046093           8        XCF04S
 3        06e5e093           8        XC2C64A
```

FPGA Type

CPU Config

```
MicroBlaze Processor Configuration :
-------------------------------------
Version.............................8.00.b
Optimization........................Performance
Interconnect........................PLB_v46
MMU Type............................Full_MMU
No of PC Breakpoints................1
No of Read Addr/Data Watchpoints...0
No of Write Addr/Data Watchpoints..0
Instruction Cache Support...........on
Instruction Cache Base Address.....0xc0000000
Instruction Cache High Address ....0xc3ffffff
Data Cache Support..................on
Data Cache Base Address.............0xc0000000
Data Cache High Address.............0xc3ffffff
Exceptions  Support.................on
FPU  Support........................off
Hard Divider Support................off
Hard Multiplier Support.............on - (Mul32)
Barrel Shifter Support..............on
MSR clr/set Instruction Support....on
Compare Instruction Support........on
Data Cache Write-back Support......off
Connected to "mb" target. id = 0
Starting GDB server for "mb" target (id = 0) at
TCP port no 1234
```

Download to Starterkit (Linux boot)

```
XMD% dow ./simpleImage.xilinx_sp3
Downloading Program -- ./simpleImage.xilinx_sp3
     section, .text: 0xc0000000-0xc027cadf
     section, .init.text: 0xc0346000-0xc035e1e3
     section, .init.ivt: 0xc035fe50-0xc035fe77
     section, __fdt_blob: 0xc027cae0-0xc0280adf
     section, .rodata: 0xc0281000-0xc031892f
     section, __ksymtab: 0xc0318930-0xc031ceff
     section, __ksymtab_gpl: 0xc031cf00-0xc031efff
     section, __ksymtab_strings: 0xc031f000-0xc032cdf7
     section, __param: 0xc032cdf8-0xc032dfff
     section, __ex_table: 0xc032e000-0xc032ecd7
     section, .sdata2: 0xc032ecd8-0xc032efff
     section, .data: 0xc032f000-0xc03442df
     section, .data..shared_aligned: 0xc03442e0-0xc034436b
     section, .init.data: 0xc035e1e4-0xc035fe4f
     section, .init.setup: 0xc035fe78-0xc03600e7
     section, .initcall.init: 0xc03600e8-0xc036032b
     section, .con_initcall.init: 0xc036032c-0xc036032f
     section, .init.ramfs: 0xc0361000-0xc04ef45f
     section, .bss: 0xc04f0000-0xc051cf07
Setting PC with Program Start Address 0xc0000000
System Reset .... DONE
XMD% con
Processor started. Type "stop" to stop processor
```

Linux boot console output

```
Ramdisk addr 0x00000003, Compiled-in FDT at 0xc027cae0
Linux version 2.6.36.2 (kugel@pcakulap) (gcc version 4.1.2 20070214 (Xilinx 12.3 Build EDK_MS3.66
14 Jul 2010)) #43 PREEMPT Sat Jan 8 19:28:40 CET 2011
...
setup_memory: max_mapnr: 0x4000
setup_memory: min_low_pfn: 0xc0000
setup_memory: max_low_pfn: 0xc4000
On node 0 totalpages: 16384
...
Kernel command line: console=ttyUL0 root=/dev/ram ip=192.168.0.10
...
Memory: 59664k/65536k available
xlnx,xps-intc-1.00.a #0 at 0xc4000000, num_irq=4, edge=0xb
xlnx,xps-timer-1.00.a #0 at 0xc4004000, irq=0
Calibrating delay loop... 23.75 BogoMIPS (lpj=118784)
XGpio: /plb@0/gpio@81420000: registered
XGpio: /plb@0/gpio@81400000: registered
...
io scheduler cfq registered (default)
84000000.serial: ttyUL0 at MMIO 0x84000000 (irq = 1) is a uartlite
console [ttyUL0] enabled
...
++ Starting INETD (e.g.for ftp)
Sun Jan  9 19:15:09 2011
rcS Complete
/ #
```

Login prompt

Running Linux on Starterkit

- Access
- Commands
- Development
  - WWW
  - Applications
  - Modules

- Access

- Console and telnet
  - Serial port is primary Linux terminal
  - With Microblaze serial port can be run over JTAG (not tested)
  - Several telnet terminals (default: 5) available
    - [kugel@akudesk linux-2.6.36.2]$ telnet 192.168.0.10
    - Trying 192.168.0.10...
    - Connected to 192.168.0.10.
    - Escape character is '^]'.
    - / # uname -a
    - Linux sp3eStartKit 2.6.36.2 #43 PREEMPT Sat Jan 8 19:28:40 CET 2011 microblaze GNU/Linux
    - / #
  - Per default root access without password
  - Additional users can be added via adduser

- Access (2)

- Ftp
  - FTP daemon started via inetd
  - Configuration in /etc/inetd/inetd.conf
    - / # cat /etc/inetd/inetd.conf
    - 21 stream tcp nowait ftp ftpd ftpd -w /tmp
      - Default user ftp, created by startup script in /etc/init.d/rcS
      - Default transfer directory /tmp
  - Cross-compiled binaries (and other files) can be uploaded via ftp
  - Tar and gzip work as usual to simplify uploads

- Linux commands

- Basic tools
  - ls, cp, mv, mkdir, chmod, find, date, ...
- System tools
  - Dmesg, ps, kill, ifconfig, udhcpc, ...
- System files
  - /dev
    - Device nodes
  - /etc
    - Scripts, configuration files
  - /proc, e.g.
    - /proc/cpuinfo
    - /proc/interrupts
  - /sys/class, e.g.
    - /sys/class/gpio

```
> cat /proc/cpuinfo
CPU-Family:     MicroBlaze
FPGA-Arch:      virtex5
CPU-Ver:        Unknown
CPU-MHz:        50.00
BogoMips:       23.75
HW:
 Shift:         yes
 MSR:           yes
 PCMP:          yes
 DIV:           no
 MMU:           3
 MUL:           v1
 FPU:           no
 Exc:
Icache:         0kB     line length:    16B
Dcache:         0kB     line length:    16B
                write-through
HW-Debug:       yes
PVR-USR1:       00
PVR-USR2:       00000000
Page size:      4096
/ # cat /proc/interrupts
                CPU0
  0:    2960701      edge Xilinx INTC  timer
  1:        120      edge Xilinx INTC  uartlite
  3:       4191      edge Xilinx INTC  eth0
/ #
```

- Development cycle (user land)

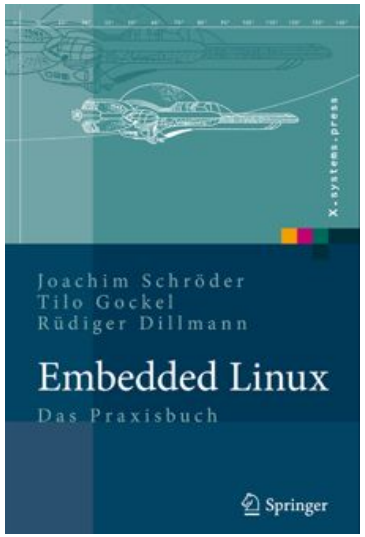
- Host based
  - Linux gcc, g++ cross-compilers
    - mb-linux-gcc/g++ (alias microblaze-unknown-linux-gnu-gcc/g++)
    - E.g. mb-linux-gcc -g -O0 -L *mbtooldir*/microblaze_v1.0/microblaze-unknown-linux-gnu/microblaze-unknown-linux-gnu/sys-root/usr/lib --sysroot=*linuxdir*/ -isystem *mbtooldir*/microblaze_v1.0/microblaze-unknown-linux-gnu/microblaze-unknown-linux-gnu/sys-root/usr/include -o hello hello.c -static -lc
  - Limited functionality due to static linking and fewer libs
    - Potentially can be improved by cross-compiling full GLIBC
  - Debugging on host
    - Linux GDB and GDBSERVER not (yet) operational on Microblaze target
    - EDK gdb not suitable for Linux applications
  - Advantage: identical code on host and target in many cases
- Transfer via ftp
- Serious error messages potentially go to system log
- On crash: kill process from other terminal

- Development cycle (kernel)

- Linux: User land (regular applications) vs. Kernel (drivers/modules)
  - ⇨ Static (boot time loaded) drivers
    - For ES: UART, timer, network, IRQ
    - Disk, VGA not relevant
  - ⇨ Loadable kernel modules
    - insmod
    - rmmod
    - Lsmod
    - Big advantage: no need to recompile kernel
- Same host based development, but different includes and libs
  - ⇨ Limited debugging on host
  - ⇨ Debugging via printk() and kernel log

Applications

- Simple applications (... hello world ...)
- Simple Timing
- Simple HW access via GPIO
- Advanced apps (mostly from Embedded Linux book)
  - ⇨ Threads, Mutexes
  - ⇨ Sockets
  - ⇨ Periodic threads
  - ⇨ IPC



Many useful code samples at

http://www.praxisbuch.net/embedded-linux/downloads/embedded-linux-source-1.0.zip

- Simple applications

- Simple „hello world“-type applications straightforward
  - ⇨ Just different compiler, include and lib directories
- Simple timing
  - ⇨ Standard Linux interval timer
    - Install signal handler: signal(SIGALRM, sigHandler);
    - Set time: setitimer(timer, &tval, 0);
    - Wait for signal: sigwait(&sigList, &theSig);
  - ⇨ Runs on both host and target
- HW access, e.g. to LEDs and switches
  - ⇨ User apps cannot directly access physical memory
  - ⇨ GPIO via sysfs
    - GPIOs are availabe via /sys/class/gpio
    - Use from shell (script) via echo and cat
    - Use from program via fread/fwrite to sysfs files
    - Very simple, slow
  - ⇨ GPIO via mmap
    - Map physical memory via /dev/mem file to virtual address
    - Much faster, only for programs

Hardware access via /dev/mem

Example mbFastIo

```
// open memory access
memfd = open("/dev/mem", O_RDWR | O_SYNC);

// Map physical device memory (1 page) into user space
mapped_base = mmap(0, MAP_SIZE, PROT_READ | PROT_WRITE, MAP_SHARED,
memfd, dev_base & ~MAP_MASK);

// get the address of the device in user space (offset from base address)
mapped_dev_base = mapped_base + (dev_base & MAP_MASK);

// write to the direction register so all the GPIOs are on output to drive LEDs
*((unsigned long *) (mapped_dev_base + GPIO_DIRECTION_OFFSET)) = 0;

// set output value
*((unsigned long *) (mapped_dev_base + GPIO_DATA_OFFSET)) = 0x1234;

// unmap and close
munmap(mapped_base, MAP_SIZE);
close(memfd);
```

- Advanced applications

- Multi-threading
  - Single program, multiple threads
  - Threads share address space
    - Data exchange via global variables (structures) possible
    - Issue: synchronisation and resource sharing
  - Posix thread, mutex, semaphore, condition variables
    - pthread_create(...);
    - pthread_join(...);
    - pthread_mutex_lock(&mutex);
    - pthread_cond_signal(&cond);
    - pthread_mutex_unlock(&mutex);

```
$ ./threads_basic0
./threads_basic0: condition demo
  Thread 1: reading value
  Thread 2: incrementing value
Exit Program with Ctrl+C
Initial value is 4
(2) Incrementing value, is now 5
(1) Read value 5
(2) Incrementing value, is now 6
(1) Read value 6

$ ftp 192.168.0.10
ftp> put threads_basic1
226 Operation successful
683159 bytes sent in 3,05 secs
(223,94 Kbytes/sec)
ftp> quit

$ telnet 192.168.0.10
Trying 192.168.0.10...
Connected to 192.168.0.10.
/ # chmod +x /tmp/threads_basic1
/ # /tmp/threads_basic1
/tmp/threads_basic1: condition demo
  Thread 1: reading value
  Thread 2: incrementing value
Exit Program with Ctrl+C
Initial value is 4
(2) Incrementing value, is now 5
(1) Read value 5
(2) Incrementing value, is now 6
(1) Read value 6
```

- Advanced applications (3)

- **Performance**
  - ⇨ **Embedded Linux book „Servo“ example**
    - **Periodic thread, 20 ms, 1.6GHz ATOM**
    - **Kernel 2.6, RT prio: max 297µs deviation**
  - ⇨ **MB on Spartan3: max +/-7ms deviation**

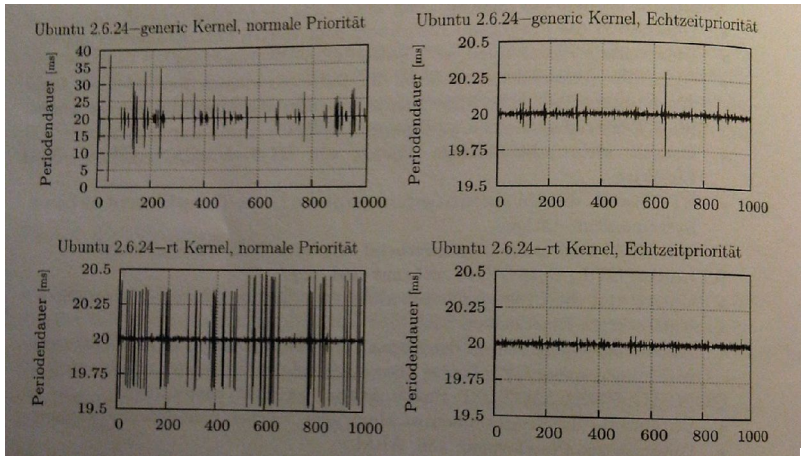
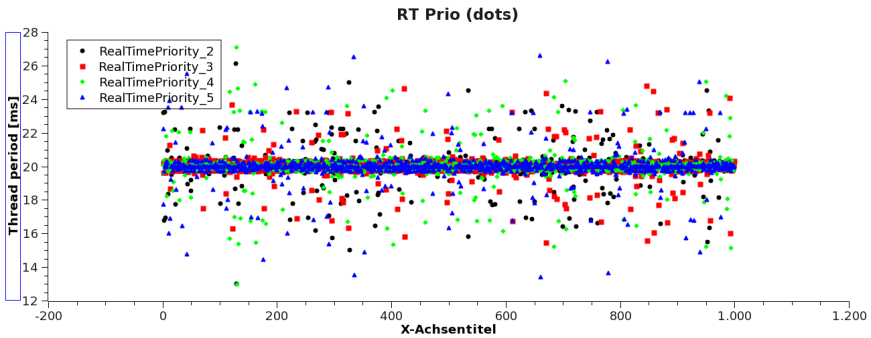


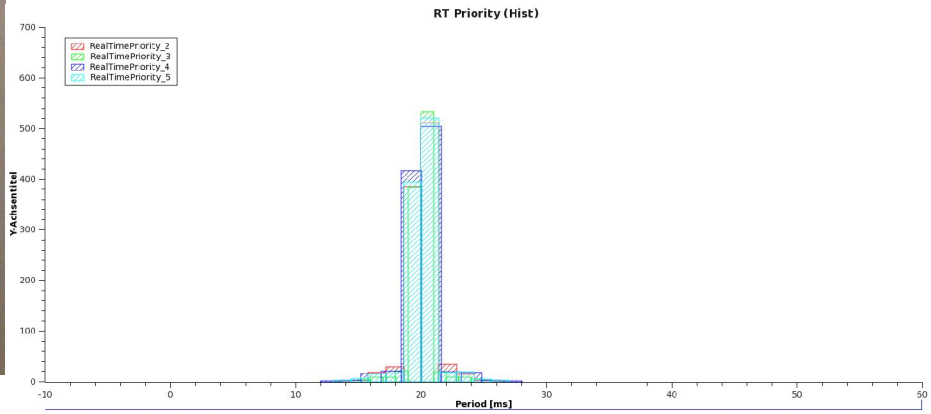
Abb. 12.7. Grafische Darstellung der Periodendauer von 1 000 aufeinanderfolgenden Ausführungen des `ServoThreads` für die vier untersuchten Konfigurationen.

Konfiguration	Min	Avg	Max
Ubuntu 2.6.24-generic Kernel, normale Priorität	0,01 µs	24,71 µs	18695,85 µs
Ubuntu 2.6.24-generic Kernel, Echtzeitpriorität	0,01 µs	9,35 µs	297,90 µs
Ubuntu 2.6.24-rt Kernel, normale Priorität	0,0 µs	6,56 µs	484,37 µs
Ubuntu 2.6.24-rt Kernel, Echtzeitpriorität	0,01 µs	6,78 µs	116,88 µs

Tabelle 12.2. Die minimale (Min), durchschnittliche (Avg) und maximale (Max) Abweichung von der Periode für die vier untersuchten Konfigurationen bei jeweils 1 000 aufeinanderfolgenden Ausführungen.

J. Schröder, Embedded Linux





- Advanced applications (4)

- Network programming
  - Simple UDP sockets (unreliable)
  - Standard TCP sockets
  - Client-server applications
  - Remote display (QT, ...)
- Watch endiannes
  - Use htonl, htons, ntohl, ntohs
  - Carefull with structs containing non-aligned members
    - Strings, short, ...
- Check firewall

```
$ ./receiver0 8095
Receiving UDP-packets at port 8095..
Received string: sdasdk from
192.168.0.10:46414
Received string: 123456 from
192.168.0.10:46414
Received string: sdqd from
192.168.0.10:46414
^C
$
```

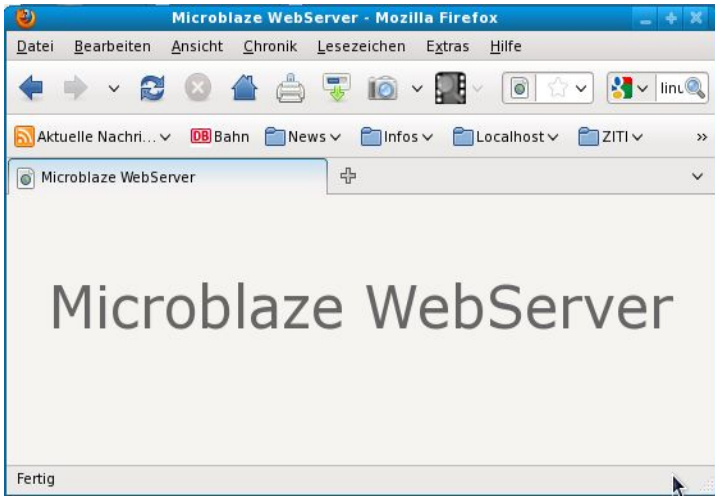
```
# sender1 192.168.0.134 8095
Sending UDP-packets to
192.168.0.134:8095..
Enter your string now, ENTER to send
sdasdk
Enter your string now, ENTER to send
123456
Enter your string now, ENTER to send
sdqd
Enter your string now, ENTER to send
^C
/ #
```

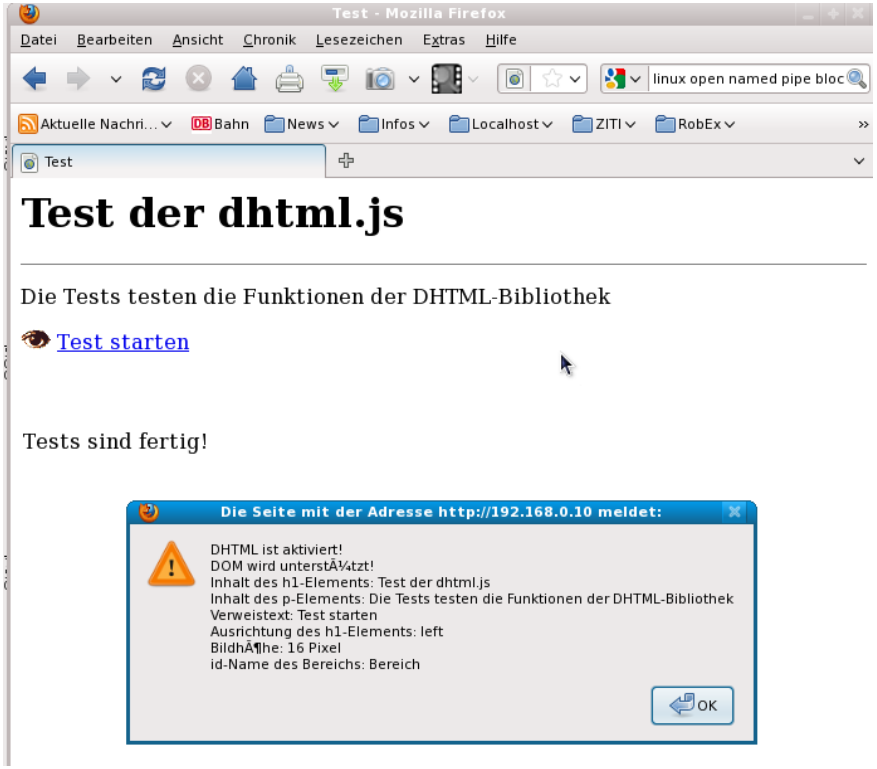
- Kernel modules

- Device classes: character vs. block
  - Simple load/unload via insmod, rmmod
    - Module version must match kernel version
    - Modules to be placed in /lib/modules/<kernel-version>/
  - Modules can access phyical memory
  - Modules cannot use user-land libraries and files
  - Modules can interact to user-land via /proc filesystem
- Samples from „Embedded Linux“ book
  - Module template
    - init, exit
    - open, close, read, write
    - ioctl (device control)
    - Minimal module < 20 lines of code (no real function)
  - Kernel - Userspace transfers
    - E.g. character device
- UIO
  - User space IO/ drivers: exists, but not information

Webserver

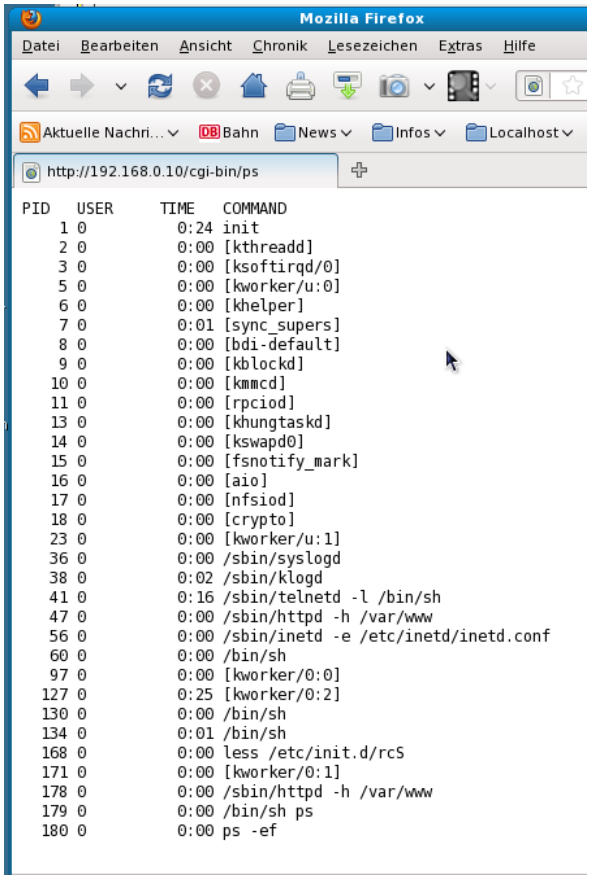
- Busybox comes with httpd daemon (web server)
  - Simple configuration: start in /etc/init.d/rcS
    - /sbin/httpd -h /var/www
  - Plain html and DHTML
    - Simple display
    - Downloads (e.g. log files)
    - Interactions via Javascript





- Advanced WWW

- **CGI**
  - ⇨ **Access to scripts and programs on server**
  - ⇨ **Full access possible (e.g. measurements, control)**



  - ⇨ **Information transfer to and from CGI application**

SAN (Storage Area Network) – host preps

Check VBLADE service on host

- $ cat /etc/vblade.conf

  # network_device shelf slot file/disk/partition mac[,mac[,mac]]

  #eth3 0 0 /home/kugel/temp/aoe/aoe.dat 00:0A:35:08:21:00

Edit proper inferface, filename and MAC address

Start service

$ sudo service vblade restart

vblade starten:

.../aoe.dat (e0.0@wlan0) [pid 24785]
[OK]

SAN - discovery

- **/ # aoe-sancheck**

```
Probing... etherd/e0.0: unknown partition table
done.
==========================================
INTERFACE SUMMARY
==========================================
Name      Status        MTU  PCI ID
eth0      UP     1500
==========================================
DEVICE SUMMARY
==========================================
Device    Macs Payload      Local Interfaces
e0.0        1    1024  eth0
```

SAN – mount

- **/ # n_partitions=1 n_shelves=1 aoe-mkdevs /dev/etherd**

  sh: set: -e: invalid option

  sh: set: -e: invalid option
- **/ # mkdir /usr/local**
- **/ # mount /dev/etherd/e0.0 /usr/local**

  EXT2-fs (etherd/e0.0): warning: mounting unchecked fs, running e2fsck is recommended
- **/ # df**

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/etherd/e0.0	1032088	271788	707872	28%	/usr/local

SAN – use

- **/ # ls /usr/local/**

  bin lost+found
  etc microblaze-unknown-linux-gnu
  lib mist.txt
  libs.tgz sbin
  libsrc usr

- ***... use disk ...***
- **/ # cp /usr/local/mist.txt /tmp/**
- **/ # cat /tmp/mist.txt**

  export HOME='/'
  export TERM='vt102'
  ...

- **/ # umount /usr/local/**

To Do's

- Boot loader for FLASH (replace XMD)
- Network fixes
  - DNS resolution (nslookup not working)
  - SSH (*dropbear* not compiling)
  - NFS/CIFS mount (no connection yet)
- Tools: gcc4.6 should support Mblaze natively. Verify!
- Libraries
  - Test Xilinx shared libs
  - Port standard GLIBC
  - Libs need much space in ramfs
    - Usefull only with network mount working
- Debugging
  - Port gdb or gdbserver
- Remote graphics
  - Nano-X, QT client
- http://li5.ziti.uni-heidelberg.de/lectures/2010hws/embedded/