Status of WR software support (2010-12-10)

- The Linux kernel for the White Rabbit Switch V2
- The build scripts
- The Interrupt Controller in the FPGA
- The NIC device driver
- Other development plans
- The testing environment

The build scripts

The wr-switch-software package has no build scripts
- The current build scripts are Tomasz's work
- They work awfully well but are suboptimal in some ways

The package offers some post-factum fixup
- Mounting initramfs instead of ext2 ramdisk
- Using the new drivers instead of old ones

More integration is needed. These are the plans:
- Move build scripts inside wr-switch-software
- Add more developer-oriented kernel configs (e.g.: NFSROOT)
- Have a mechanism to avoid downloading stuff every time
- Document the procedure to ease new users

Linux Kernel for White Rabbit Switch V2

The current kernel is 2.6.35, ported by Tomasz
- The original tree has been put into a git repository
- The patch-set is ready to be rebased to more recent kernels

I plans to keep rebasing to the latest upstream kernel
- We must check the level of integration of Atmel patches
- The rebase and testing should be done at a non-hot time lapse
- The upgrade to 2.6.36 won’t happen immediately

The VIC Interrupt Controller in the FPGA

The WR-VIC driver has been rewritten as an irq_chip
- Irq chips hide to driver writers the level of multiplexers
- It’s the standard mechanism by which every irq is handler in Linux

Unfortunately, there is no support (yet) for pluggable irq muxes
- We needed to export two more internal kernel symbols
- We decided to disable the upper irq line when no clients are there
  - The multiplexer disappears while reprogramming the FPGA
  - An undriven IRQ line with a registered mux would lock the system

We need better integration in the kernel
- The VIC code can be moved in the board-specific kernel patch
- The kernel can be extended to support pluggable IRQ multiplexers
**The NIC device driver**

**The NIC driver has been written from scratch**
- The code is heavily based on the Minic achievements (Tom/Emilio)
- The driving point of this driver is cleanliness and maintainability

**Currently, the driver is mostly working**
- Packet transmission and reception is working
- Timestamping is being worked on

**Tomasz already fixed a number of bugs in the initial code**
- I hope this validates the design goals

---

**The testing environment**

**We are working on a test suite for switches**
- The target is mainly the WR switch
- We are currently testing a cheap commercial switch
- The plan is comparing WR switch with serious commercial switches

**The testing environment is designed to be**
- Easily portable: plain C, no external libraries
- Extensible: each test is a separate file
  - Each file includes both client and server
  - A linker script trick is used to select code sections
  - To add a test, just plug a file and add it to Makefile

---

**Other development fields (personal wishlist)**

**The u-boot port has not yet been cleaned up**
- The patch should be checked and possibly rebased
- A short git branch over upstream might increase acceptance
  - This is not relevant to switch users, only to developers
  - Developers are one of the targets of this project, anyways

**We may consider switching to barebox**
- It’s the project that originally was born as “U-Boot V2”
- Barebox is designed much better than U-Boot is
- The AT91SAM9 family is already very well supported

**I’d also love to clean up the ARM7 code base**
- Again, it should be integrated in the software package
- Integration and documentation makes the package stronger and better
- Its role in the switch is simple enough to be a testbed for ...

---

**RFC-1242, RFC-2544, RFC-2889**

**These RFC documents suggest a set of tests for switches**
- We plan to implement those tests
- A test suite can be “conditionally compliant”
  - It implements the mandatory tests
  - Or it can be “unconditionally compliant”
  - It implements also the optional tests

**Our aim is being unconditionally compliant**
- However we didn’t evaluate the overall effort for that

**The tests are mainly concerned with**
- Throughput
- Latency
- Frame loss rate
- Back-to-back capability
Testing equipments

Our initial tests have been made with cheap cards
  - Two RTC8169 (PCI) on a quad-core system
    - We tested loopback with and without a switch
    - The payload was raw Ethernet frames

Next step is using hardware timestamping
  - Use of two Intel 82576 devices on PCI-Express
  - HW-TS-aware tests are different tests
    - People without hw-timestamp can still run some tests
    - People with hw-timestamp can exploit the extra precision

The WR Switch is a perfect testing equipment
  - We have complete control of what is happening inside
  - The final FPGA might have special test-oriented features
  - A WR-Switch user can thus test both WR switches and the overall network

Results of a preliminary latency test

![Graph showing latency results over different packet sizes.](image)