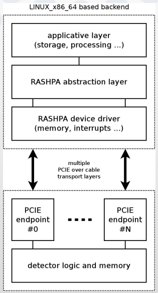
CRISP 2nd annual meeting

The use of hardware virtualization in RASHPA

Contents

- RASHPA context reminder
- Device virtualization
- The VPCIE framework

Context - RASHPA reminder


LINUX_x86_64 based backend
applicative layer (storage, processing ...)
RASHPA abstraction layer
RASHPA device driver (memory, interrupts ...)
multiple PCIE over cable transport layers
PCIE endpoint #0
....
PCIE endpoint #N
detector logic and memory

RASHPA basic goal is to transfer **detector** memory contents into a **backend** for further processing, visualisation or storage. To do so, RASHPA relies on low level hardware and software components

- a data transport layer, currently PCI Express over cable
- a data transmission engine implemented on XILINX FPGA
- LINUX kernel device driver and userland software

CRISP The Cluster of Research Infrastructures for Synergies in Physics

Context - RASHPA project issues

Still being prototyped

- how to simplify hardware software codesign ?

Aims at using multiple PCIE links. Yet, the first prototype is single link

- how to scale the prototype platform ?

Well positionned to interface with PCIE based accelerating technologies

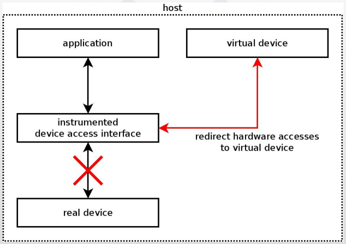
- how to test unavailable (too recent or expensive) hardware ?

CRISP
The Cluster of Research Infrastructures for Synergies in Physics

Context - solution

Solution: **device virtualization**

Device virtualization - main idea



Applications on a **host** machine access hardware via interfaces. By **instrumenting** these interfaces, one can **redirect** the accesses to a software implementing the device. The device is said to be **virtualized**.

Device virtualization - hardware access redirection (0)

Virtualization assumes device accesses are made using a known interface

- software library API (system calls)
- memory mapped registers
- CPU specific instructions (in, out)

CRISP
The Cluster of Research Infrastructures for Synergies in Physics

Device virtualization - hardware access redirection (1)

The interface is then instrumented to redirect access to the virtual device

- software hooking (code instrumentation, library replacing)
- instruction emulation (QEMU)
- hardware traps (page protection)
- architecture support (INTEL VT)
- paravirtualization (XEN)

CRISP
The Cluster of Research Infrastructures for Synergies in Physics

Device virtualization - applications

Device virtualization example applications

- support: application running on unmaintained platform
- security: sandboxing, reverse engineering
- quality: debugging, fault injection
- testing: milkymist, zinq, android

VPCIE - goals

VPCIE, a Virtual PCIE framework made for RASHPA

- virtualize PCIE endpoints
- generic enough to fit other projects
- the backend software must run **without any modification**
- should be possible to use VHDL for device simulation

VPCIE - opensource building blocks

QEMU

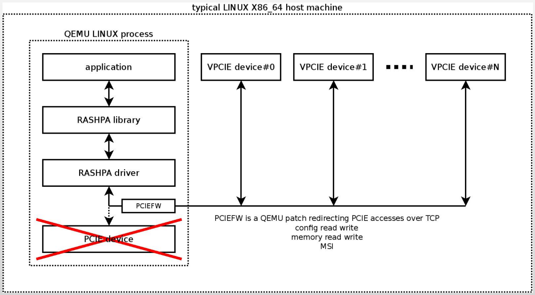
- http://wiki.qemu.org/Main_Page
- architecture emulator (X86_64, ARM ...)
- used to trap PCIE hardware accesses

GHDL

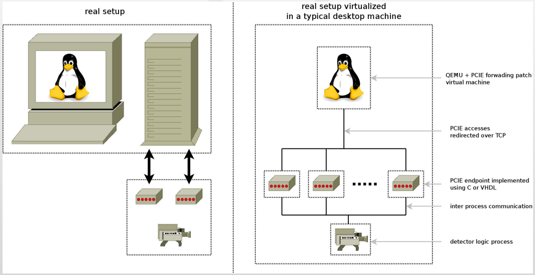
- http://ghdl.free.fr
- VHDL frontend for GCC
- used to implement device in VHDL

CRISP
The Cluster of Research Infrastructures for Synergies in Physics

VPCIE - implementation



VPCIE - RASHPA possible setup


real setup
real setup virtualized in a typical desktop machine
QEMU + PCIE forwading patch virtual machine
PCIE accesses redirected over TCP
PCIE endpoint implemented using C or VHDL
inter process communication
detector logic process

VPCIE - RASHPA backend

RASHPA backend is a full featured LINUX system

- runs in a QEMU virtual machine
- PCIE accesses are trapped and sent over TCP to the devices
- PCIE forwarder is available as a QEMU patch

CRISP The Cluster of Research Infrastructures for Synergies in Physics

VPCIE - RASHPA devices

RASHPA devices

- run as a LINUX processes
- can be implemented in C or VHDL
- can be duplicated at will

VPCIE - VHDL interface

```
entity endpoint is
 port
 (
  rst: in std_ulogic;
  clk: in std_ulogic;

  req_en: out std_ulogic;
  req_wr: out std_ulogic;
  req_bar: out std_ulogic_vector(pcie.BAR_WIDTH - 1 downto 0);
  req_addr: out std_ulogic_vector(pcie.ADDR_WIDTH - 1 downto 0);
  req_data: out std_ulogic_vector(pcie.DATA_WIDTH - 1 downto 0);

  rep_en: in std_ulogic;
  rep_data: in std_ulogic_vector(pcie.DATA_WIDTH - 1 downto 0);

  mwr_en: in std_ulogic;
  mwr_addr: in std_ulogic_vector(pcie.ADDR_WIDTH - 1 downto 0);
  mwr_data: in std_ulogic_vector(pcie.PAYLOAD_WIDTH - 1 downto 0);
  mwr_size: in std_ulogic_vector(pcie.SIZE_WIDTH - 1 downto 0);

  msi_en: in std_ulogic
 );
end entity;
```

VPCIE - C interface

```
/* runtime initialization */
int pcie_init_net(pcie_dev_t*, const char*, const char*, const char*, const char*);
int pcie_fini(pcie_dev_t*);
int pcie_loop(pcie_dev_t*);

/* misc config byte accessors */
int pcie_set_deviceid(pcie_dev_t*, uint16_t);
int pcie_set_vendorid(pcie_dev_t*, uint16_t);

/* PCIE BAR access handlers */
typedef void (*pcie_readfn_t)(uint64_t, void*, size_t, void*);
typedef void (*pcie_writefn_t)(uint64_t, const void*, size_t, void*);
int pcie_set_bar
(pcie_dev_t*, unsigned long, size_t, pcie_readfn_t, pcie_writefn_t, void*);

/* host memory read write operations */
int pcie_write_host_mem(pcie_dev_t*, uint64_t, size_t*);
int pcie_read_host_mem(pcie_dev_t*, uint64_t, size_t*);

/* send an MSI */
int pcie_send_msi(pcie_dev_t*);
```

CRISP
The Cluster of Research Infrastructures for Synergies in Physics

VPCIE - benefits

Hardware software codesign

- reduce development time
- no **modification** on the backend software (esp. driver)
- act as a stimuli generator for VHDL simulations

Platform scaling

- one PCIE endpoint per LINUX process

Investigate unavailable technologies

- NVM Express support for QEMU is available

CRISP
The Cluster of Research Infrastructures for Synergies in Physics

VPCIE - more benefits

Virtual machine

- resources easily changed (RAM amount ...)
- test with different CPU architectures (x86, x86_64, ARM ...)
- small LINUX disk images reduce reboot times (few seconds)

Fault injection

- LINUX and software response to PCIE device disconnection

VPCIE - availability

https://github.com/texane/vpcie

VPCIE - demo

demo

The Cluster of Research Infrastructures for Synergies in Physics