Using FPGA for driver testing

Marek Vašut <marex@denx.de>

October 5, 2015

Marek Vasut

- Software engineer at DENX S.E. since 2011
  - Embedded and Real-Time Systems Services, Linux kernel and driver development, U-Boot development, consulting, training.
- Custodian at U-Boot bootloader
- Versatile Linux kernel hacker
- FPGA enthusiast

Table of content

- Introduction
- Simple cases
- Complex cases
- Conclusion

Motivation

Why this talk?

- Fuzz testing is successful in finding issues
- Kernel frameworks are easy to test via software
- Hardware drivers are harder to test via software
- Hardware itself is very hard to test via software

But there is another way . . .

Fuzz testing

- Feed the tested component with almost correct inputs
- Observe how the tested component behaves
- Look for crashes, misbehavior
- Tools: Trinity, AFL . . .

Fuzzing hardware

- Nicely applicable to busses
  - SPI, I2C, . . . – easy
  - Bus is almost working :-)
- Nicely applicable to endpoint devices
  - SD cards, PCIe cards, . . .
  - Device responds almost correctly :-)

But busses and devices are fast . . .

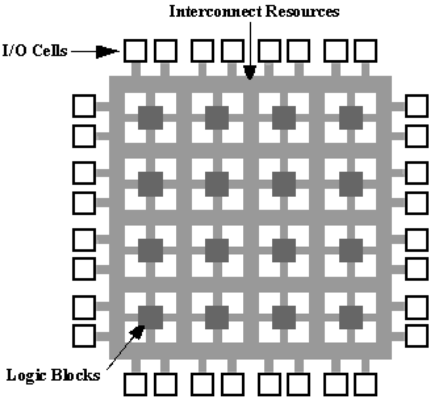
Introducing the PLD

- PLD – Programmable Logic Device
- Chip with programmable logic elements on the inside
- Also often contains DSP, Memory blocks . . .
- Usually have a lot of configurable I/O pins
- Allows implementing complex logic in the chip on demand

FPGA

- Abbr. for Field-Programmable Gate Array
- Flexible type of contemporary PLD
- Usually used for:
  - Digital Signal Processing (DSP)
  - Data crunching
  - Custom hardware interfaces
  - ASIC prototyping
  - ...
- Common vendors – Xilinx, Altera, Lattice, Microsemi...

Internal structure



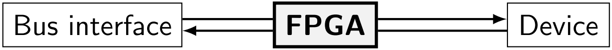
W.T.Freeman

http://www.vision.caltech.edu/CNS248/Fpga/fpga1a.gif

CC BY 2.5: http://creativecommons.org/licenses/by/2.5/

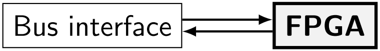
FPGA and testing

- Pass-through testing:



  - FPGA implements logic which ”understands” the bus protocol
  - FPGA inserts errors into the bus communication

- Endpoint testing:



  - FPGA implements logic which emulates the device
  - FPGA inserts errors into the device communication

Fuzzing simple busses

- I2C, SPI, . . .
- Frequency is either low or configurable
- Number of bus wires is limited
- Bus protocol is simple
- Attaching FPGA is easy, use bus buffers

I2C pass-through testing

- I2C uses very simple protocols
- Device address followed by a few bytes
- FPGA scans the bus for device address
- FPGA scans the bus for particular register I/O
- Upon a device response, modification is applied

I2C device testing

- FPGA implements the model of the EEPROM
- FPGA listens on the bus for the address
- FPGA responds on all requests
- FPGA introduces random bit errors

SPI bus

- FPGA implements SPI slave, chipselect is the trigger
- Transfer can have arbitrary length
- Bus frequency even over 100MHz
- Special case is the DSPI/QSPI for SPI NORs

Storage and FPGAs

In case we emulate storage devices, we need storage:

- FPGA has dedicated memory cells in the fabric
- FPGA supports fast external DRAM
- FPGA can interface slow permanent storage

Moving data:

- FPGA logic does direct access to storage (often RAM)
- FPGA interrupts a CPU, which does the transfer

CPUs and FPGAs

Modern FPGAs can contain a CPU or a dozen . . .

- SoC FPGA solutions – dedicated CPU cores in the package
- Softcores can be synthesised into the FPGA fabric
  - Many softcores available
  - J2 (see Jeff's talk!), RISC-V, . . .
- A dedicated CPU can implement complex fuzzing logic
- Instruction timing matters

Fuzzing SD cards

SD card is an excellent example where CPU is needed

- SD/MMC protocol is quite complex
- The protocol is stateful
- FPGA implements bus interface
- If a command happens on the bus, FPGA wakes CPU
- CPU handles the complex stateful protocol
- CPU sets up possible data transfer
- CPU instructs the FPGA to perform a response on the bus

Emulated SD card

Emulated SD card has more uses than just fuzzing

- Implement configurations otherwise unobtainable in shop
- Practical example . . .

Ethernet

Speaking of IoT . . .

- Fuzzing ethernet is also possible with FPGAs
- 10BaseT is very easy even without PHY
- 100BaseT needs PHY
- Anything faster needs proper PCB design
- Example – Stratix V can do 4x100G ethernet

Fuzzing ethernet

- The ethernet traffic passing through FPGA is a stream
- Stream is processed in real-time in the FPGA fabric
- Modifications are done to the stream in real-time
- Buffering must not happen at high link speeds

PCI express

- FPGAs can contain dedicated SerDes interfaces
- Some FPGAs contain dedicated PCIe EP/RC block
- Routing PCIe tracks requires proper PCB design
- PCIe is quite similar to ethernet
- PCIe is packet-based network architecture

The End

Thank you for your attention!

Contact: Marek Vasut <marex@denx.de>