Driver synthesis

- Formal OS interface spec
- Formal device spec

Developed once for a class of devices
Use existing hardware specs
Hardware design workflow

- Informal specification
- High-level model
- Register-transfer-level description
- netlist

**Informal specification**

**High-level model**

**Register-transfer-level description**

**netlist**

- Low-level description: registers, gates, wires.
- Cycle-accurate
- Precisely models internal device architecture and interfaces

bus_write(u32 addr, u32 val) {
    ...
}

Driver synthesis as controller synthesis

- OS requests = control objective

Inputs from the environment

Driver = controller

Device

Packet has been sent

send() = send a network packet
Driver synthesis as controller synthesis

Driver = controller
OS requests = control objective
Packet has been sent
Inputs from the environment

Example: GPIO controller

GPIO controller registers
ctrl  data  stat
0=off  1=on  0=busy  1=done

Example: GPIO controller

GPIO line

Example: GPIO controller

GPIO line

Example: GPIO controller

GPIO line

Example: GPIO controller

GPIO line
Example: GPIO controller

- **OS interface spec**
  - changeLevel
  - isetComplete

- **Device spec**
  - isWrite(1)
  - isWrite(0)
  - dataWrite
  - changeLevel/stat=0
  - changeLevel/stat=1

**GPIO controller registers**
- ctrl
- data
- stat

- **GPIO line**
  - 0=off
  - 1=on

**Fairness**
- The device is infinitely often not busy

**Liveness**
- The driver must make sure that the system leaves state 2 eventually
Example: GPIO controller

Challenges

- Challenge #1: State Explosion
  - Real devices contain dozens of registers, internal FIFOs, support multiple modes of operation and failure modes etc.
  - An efficient synthesis algorithm must rely on abstraction

- Challenge #2: Partial Information
  - The driver cannot directly observe device states and state transitions
  - Partial observations can be obtained by reading register values
  - Device state can change between two register accesses

Results

Proof-of-concept implementation:

- Successfully synthesised drivers for real devices based on manually written specifications:
  - Asix AX88772 USB-to-Ethernet adapter
  - Ricoh R5C822 SD host controller
- Performance on par with manually developed drivers

Conclusions

- Automatic driver synthesis is a radical approach to improving OS reliability
- Driver synthesis poses interesting challenges in the area of controller synthesis algorithms