Towards OS kernel acceleration in heterogeneous systems

Alex Kroh | Oliver Diessel

UNSW AUSTRALIA

Never Stand Still | Engineering | School of Computer Science and Engineering

Accelerator candidates

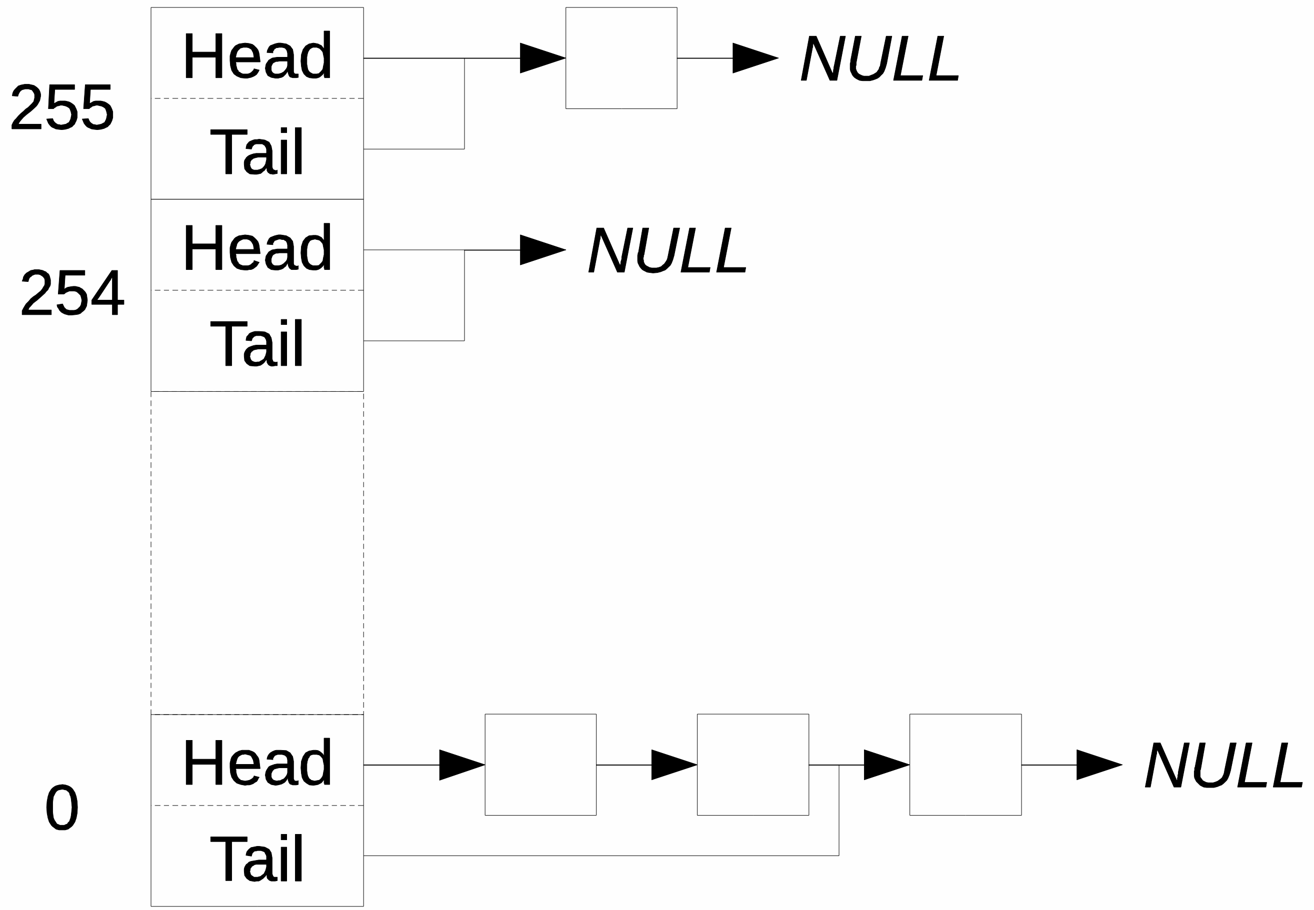
- Traditional candidates
  - Long running operations
  - Highly parallel algorithms

- OS kernel operations
  - Typically short running
  - Little parallelism (any thread will usually be blocked on IO)
  - ***Shared between all applications***
  - ***Always on critical path***
  - ***CPU execution time is non-deterministic***
    - ***Difficult WCET analysis of real-time systems***

Case study: Kernel scheduler

- Zynq-7000 APSoC
  - Dual core ARM Cortex-A9 CPU
  - On-chip FPGA fabric accessible via MMIO over ARM AXI bus
- seL4 micro-kernel
  - Minimal code executing in privileged mode
  - High performance inter-process communication (IPC)
- Fixed-priority scheduler
  - Self contained
  - Scheduling is the most frequent kernel operation
  - Trivial to implement in hardware

Case study: OS kernel scheduler (SW)



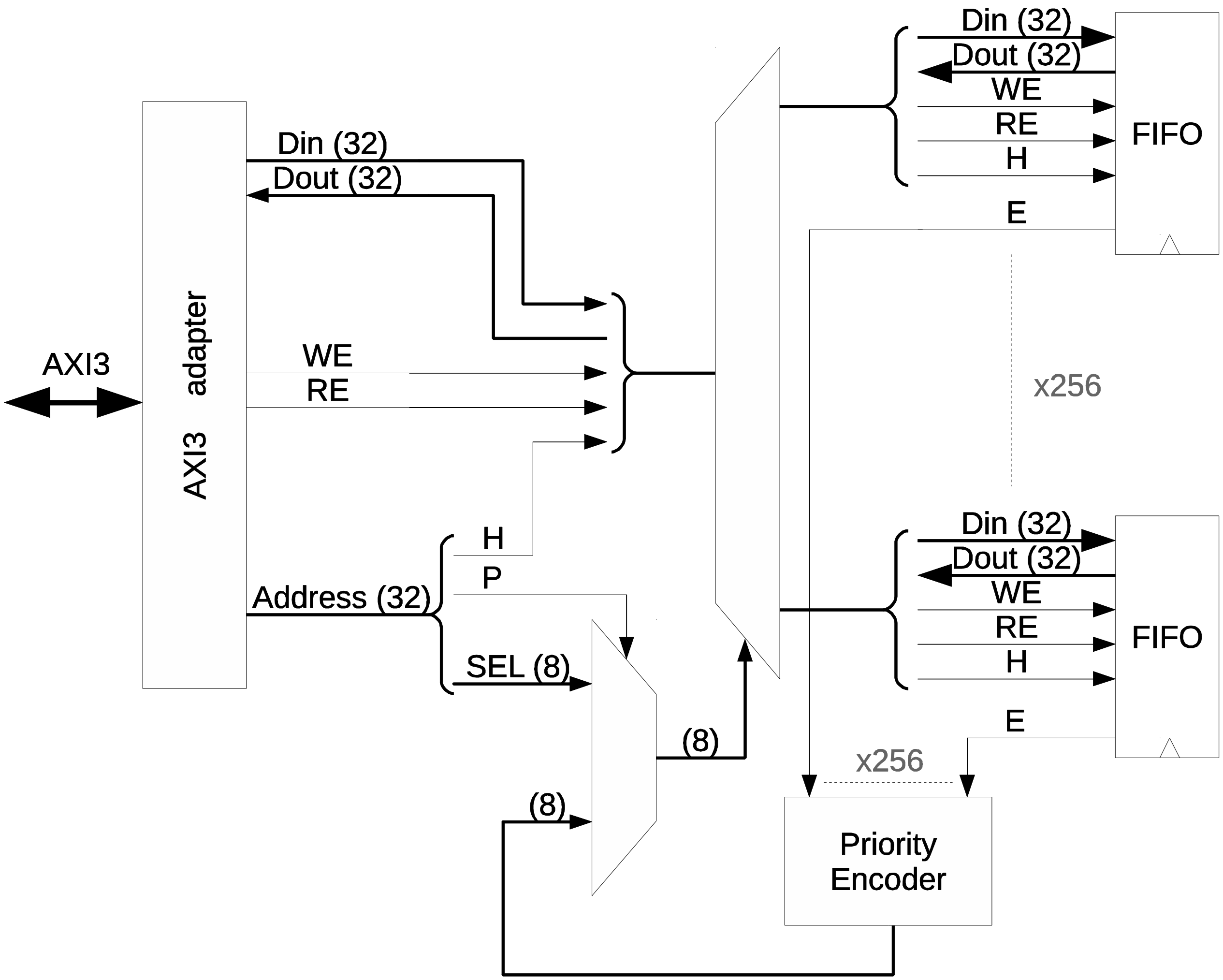
Case study: OS kernel scheduler (HW)

- 256 FIFOs (1 per priority)
  - FIFO empty signals aggregated at a priority encoder
- MMIO via ARM AXI
- Address mapping matches SW scheduler
  - Additional bit 11 selects highest priority non-empty FIFO

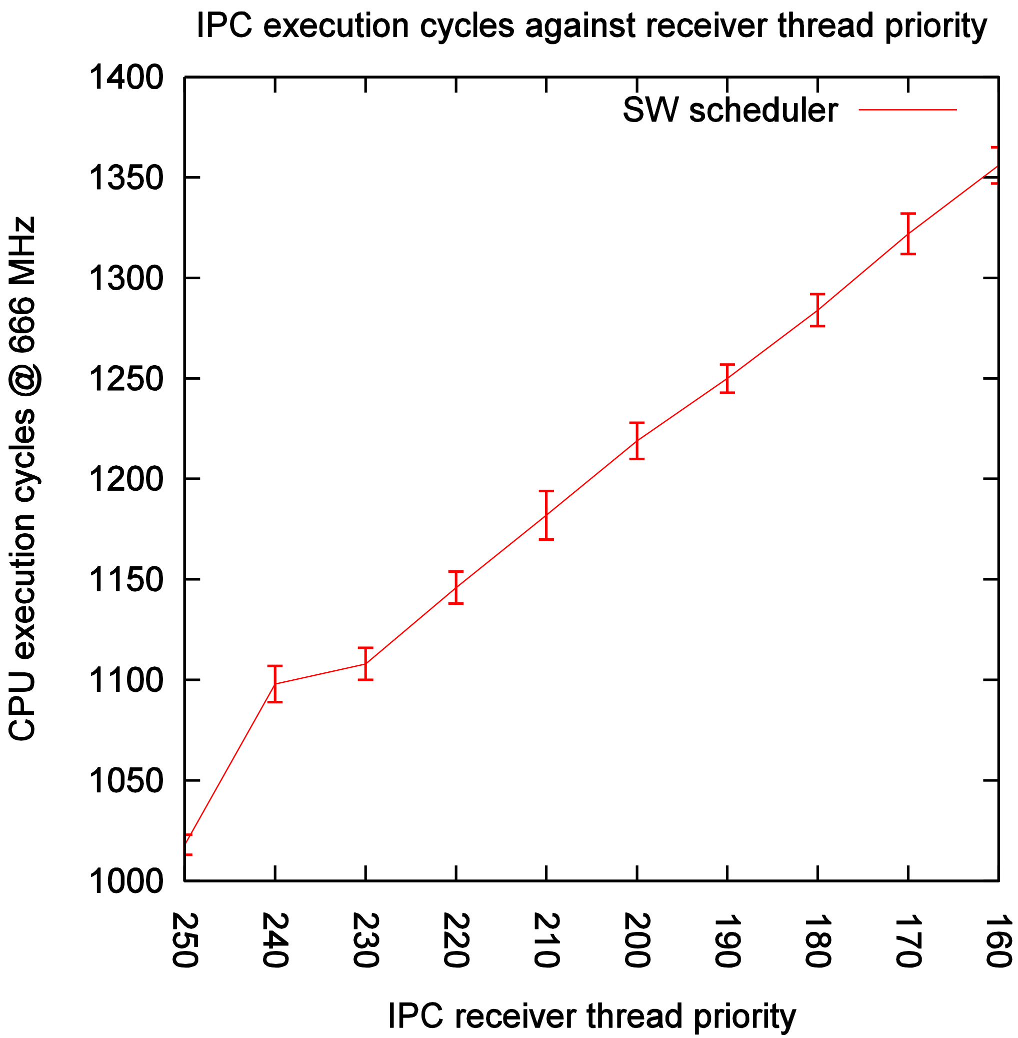
11	10	9	8	7	6	5	4	3	2	1	0
P = 0	Priority								H	0	0
P = 1	X	X	X	X	X	X	X	X	H	0	0

- WRITE := enqueue
- READ := dequeue

Case study: OS kernel scheduler (HW)



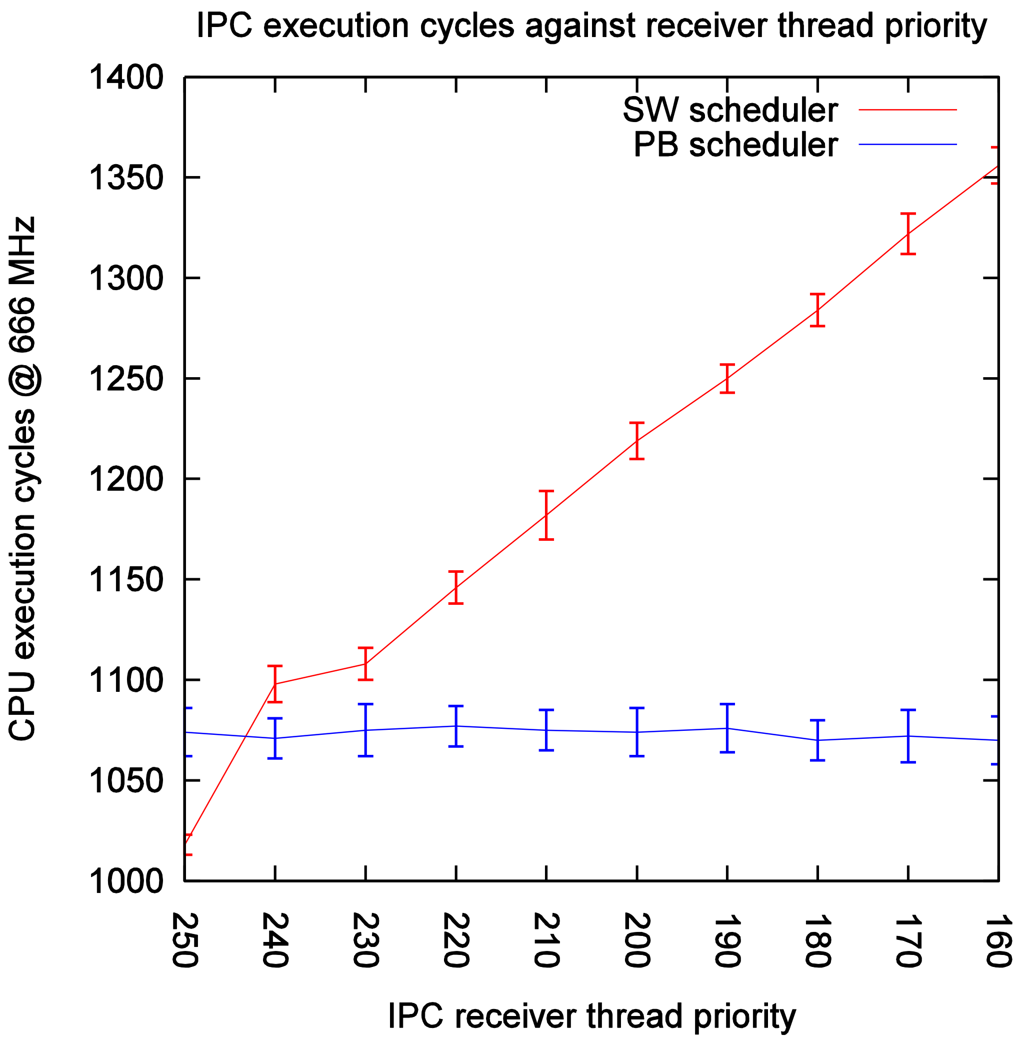
Case study: OS kernel scheduler (SW)



CPU execution cycles :=

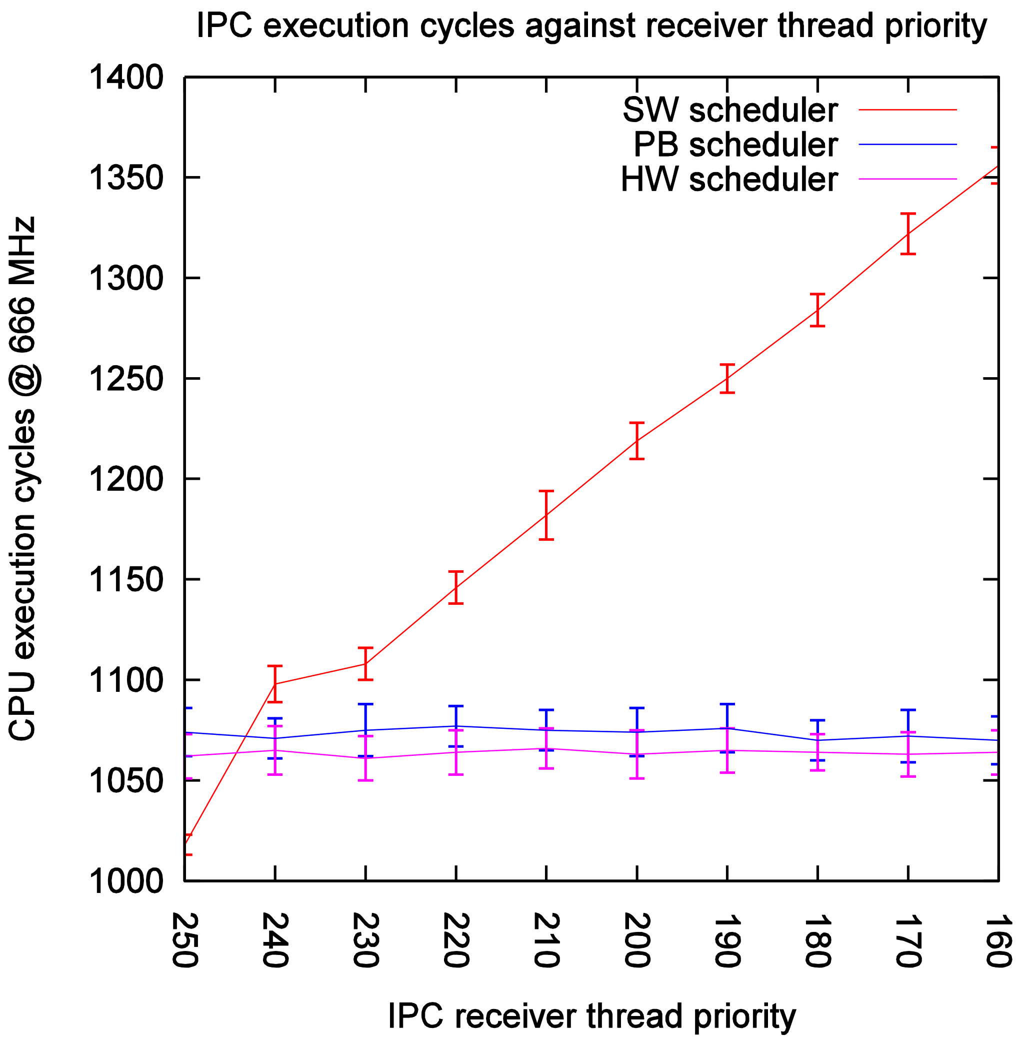
- Kernel invocation +
- Kernel scheduling +
- Kernel reply to sender

Case study: OS kernel scheduler (SW)



CPU execution cycles :=
- Kernel invocation +
- Kernel scheduling +
- Kernel reply to sender

Case study: OS kernel scheduler



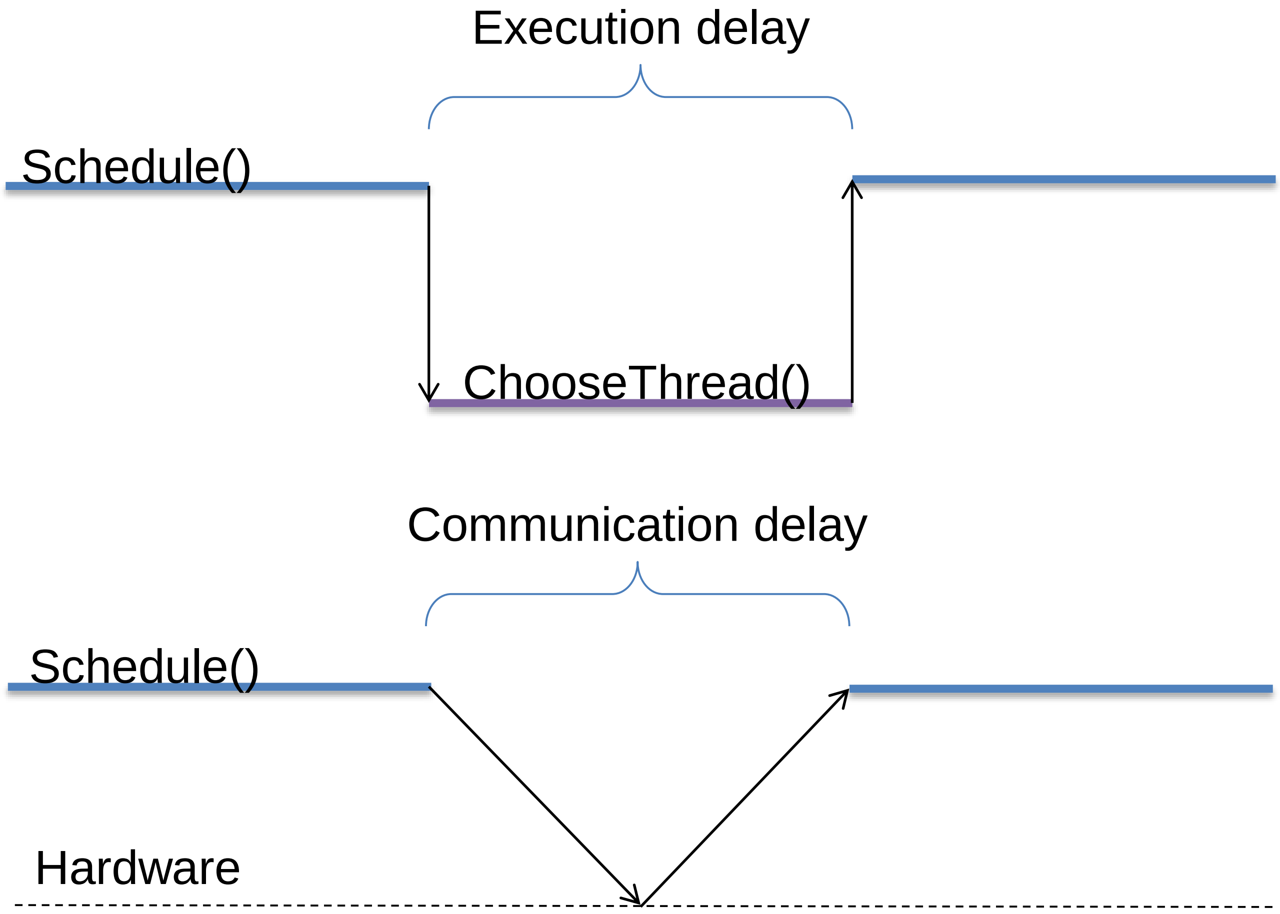
CPU execution cycles :=

Kernel invocation +

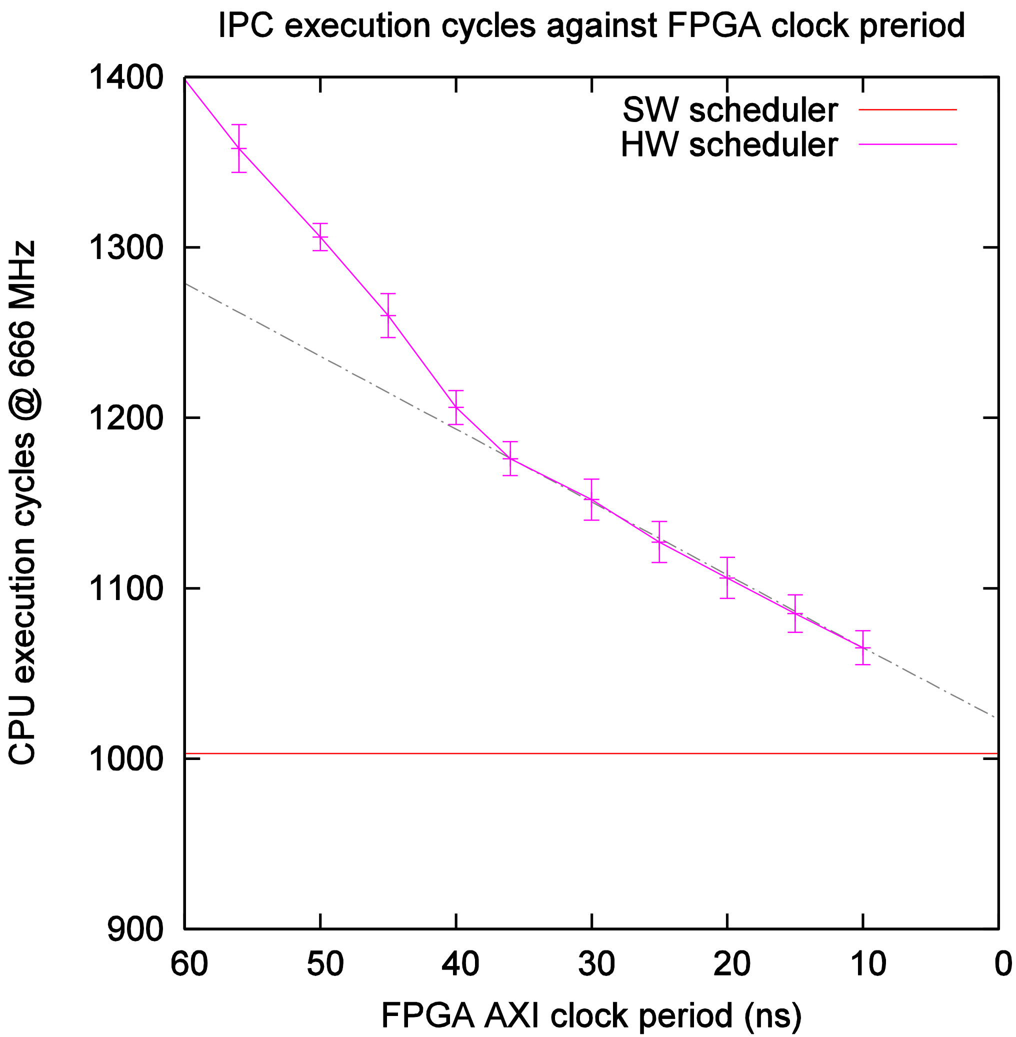
Kernel scheduling +

Kernel reply to sender

Case study: OS kernel scheduler



Theoretical limits



CPU execution cycles :=

Kernel invocation +

Kernel scheduling +

Kernel reply to sender

Future work

- Investigation of cache coherent AXI port (ACP) to reduce delay
- Acceleration of other kernel functions
- Zynq Ultrascale+
  - OS kernel acceleration for Cortex-R
  - FPGA resource virtualisation for use by virtual machines on Cortex-A53

Branch predictor anomalies