# ReconOS:
# Extending OS Services Over FPGAs

Marco Platzner
Computer Engineering Group
University of Paderborn

marco.platzner@upb.de

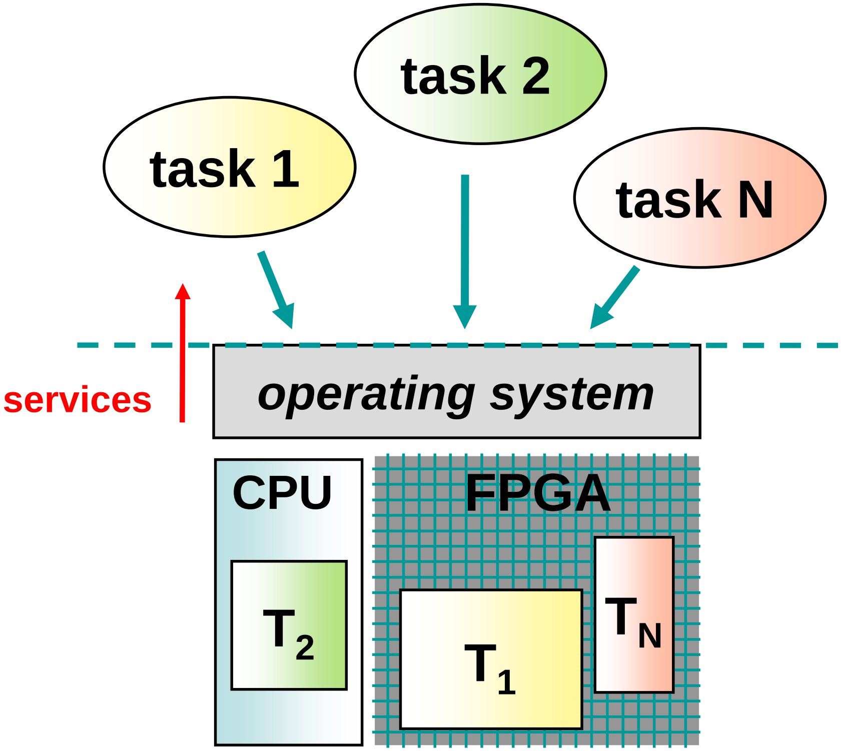
UNIVERSITÄT PADERBORN
*Die Universität der Informationsgesellschaft*

# Reconfigurable Hardware Operating Systems

- Introduce a new layer of abstraction
  - turn hardware accelerators into hardware tasks (threads)
  - rely on an operating system to schedule, place, and execute these tasks

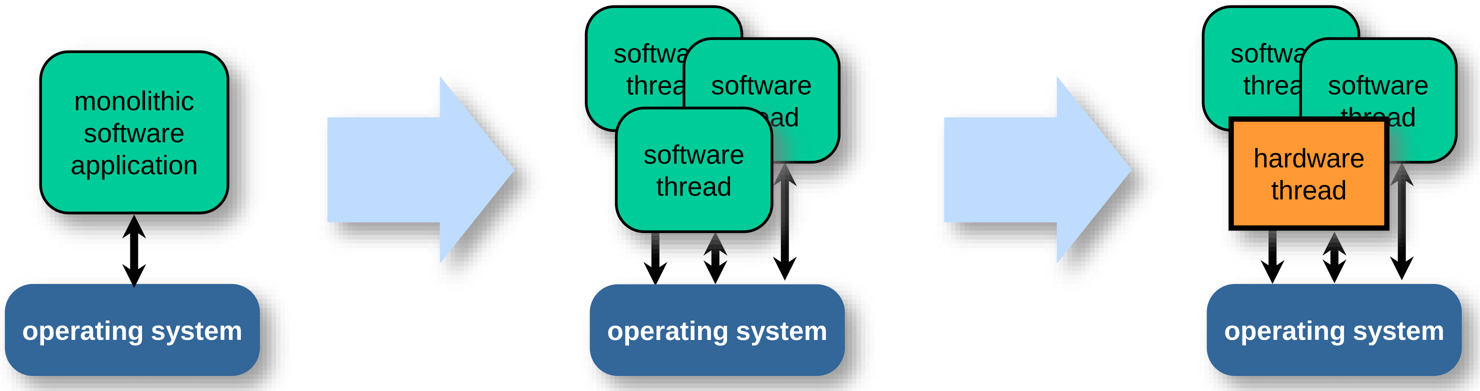
- Motivation
  - increase productivity and portability
  - exploit partial reconfigurability
  - use reconfigurable hardware for dynamic task sets

- Operating system services
  - task management
    - load/remove/preempt/resume
    - communication, synchronization
    - scheduling
  - resource management
  - time management



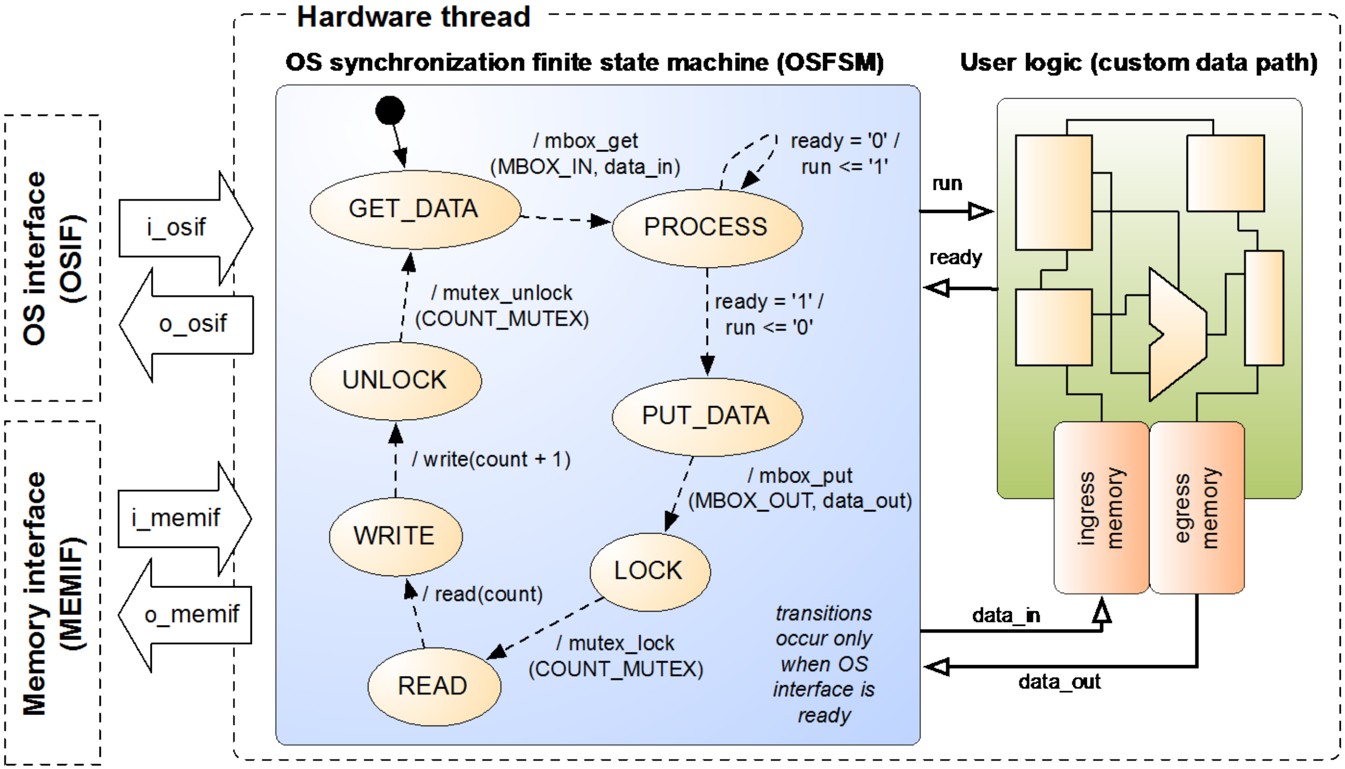
# ReconOS

- Main goal: extend the **multithreaded programming model** to reconfigurable hardware
  - threads communicate and synchronize using programming model primitives, e.g., semaphores, mutexes, mailboxes, shared memory
  - established model in software-based systems (e.g., POSIX pthreads)



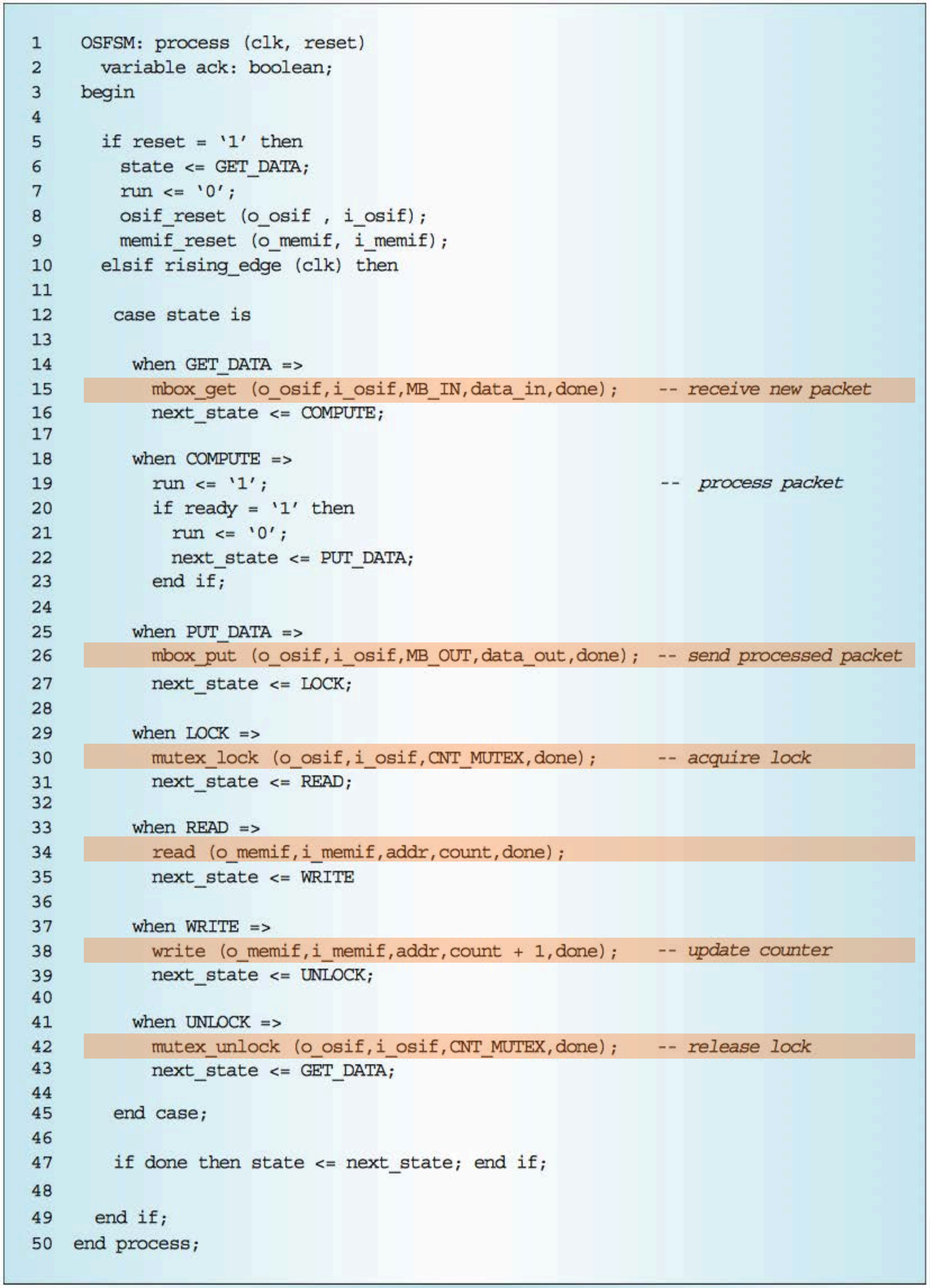
# Hardware Threads

- A hardware thread consists of two parts
  - OS synchronization finite state machine
  - user logic
- A hardware thread is connected to the
  - OS on the main CPU via the OSIF
  - main memory via the MEMIF



# Hardware Threads

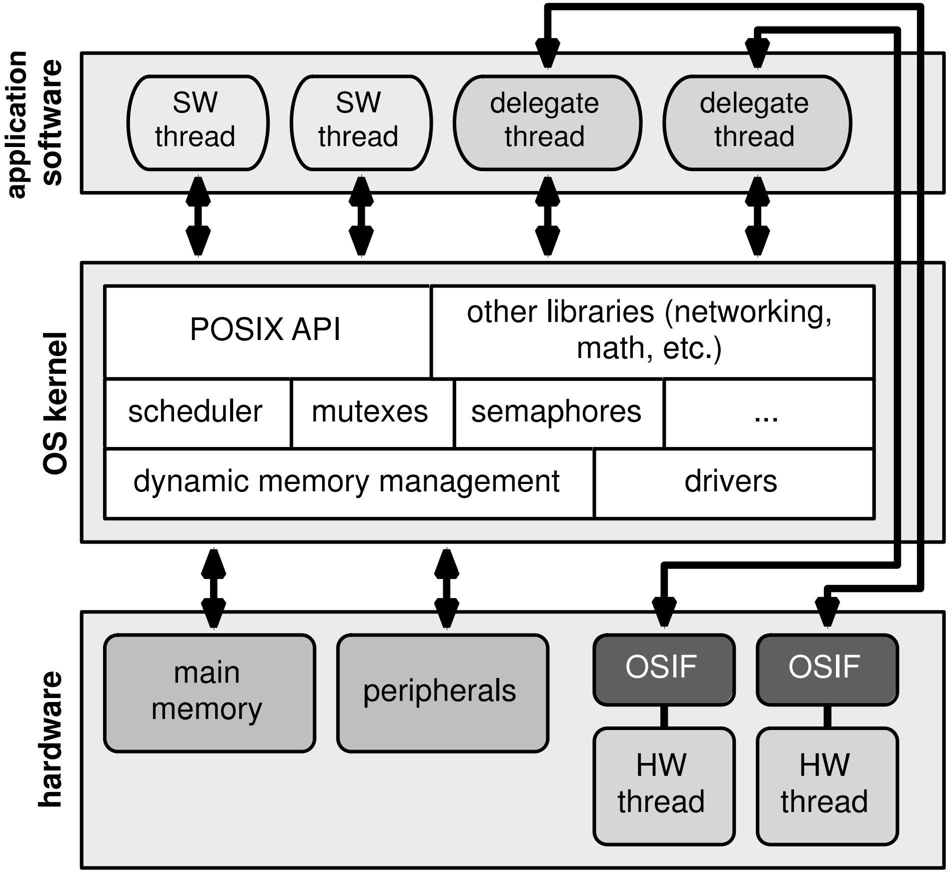
- Function library (VHDL) for implementing the OS synchronization FSM



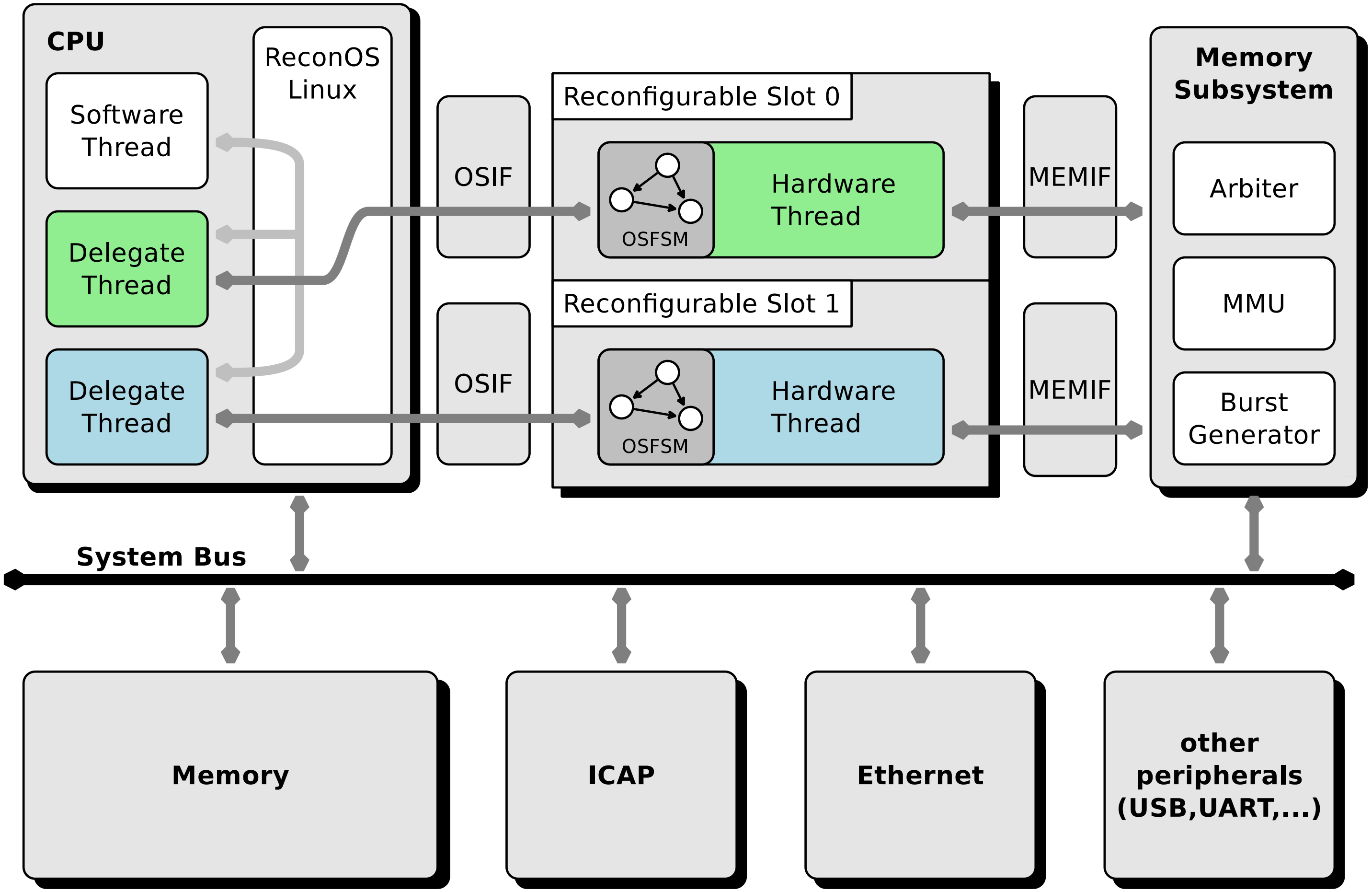
```
OSFSM: process (clk, reset)
  variable ack: boolean;
begin

  if reset = ‘1’ then
    state <= GET_DATA;
    run <= ‘0’;
    osif_reset (o_osif , i_osif);
    memif_reset (o_memif, i_memif);
  elsif rising_edge (clk) then

   case state is

     when GET_DATA =>
       mbox_get (o_osif,i_osif,MB_IN,data_in,done);     -- receive new packet
       next_state <= COMPUTE;

     when COMPUTE =>
       run <= ‘1’;                                       --  process packet
       if ready = ‘1’ then
         run <= ‘0’;
         next_state <= PUT_DATA;
       end if;

     when PUT_DATA =>
       mbox_put (o_osif,i_osif,MB_OUT,data_out,done);  -- send processed packet
       next_state <= LOCK;

     when LOCK =>
       mutex_lock (o_osif,i_osif,CNT_MUTEX,done);       -- acquire lock
       next_state <= READ;

     when READ =>
       read (o_memif,i_memif,addr,count,done);
       next_state <= WRITE

     when WRITE =>
       write (o_memif,i_memif,addr,count + 1,done);     -- update counter
       next_state <= UNLOCK;

     when UNLOCK =>
       mutex_unlock (o_osif,i_osif,CNT_MUTEX,done);     -- release lock
       next_state <= GET_DATA;

   end case;

   if done then state <= next_state; end if;

  end if;
end process;
```

# Delegate Threads

- A SW delegate thread is associated with every hardware thread
  - calls the OS kernel on behalf of the hardware thread

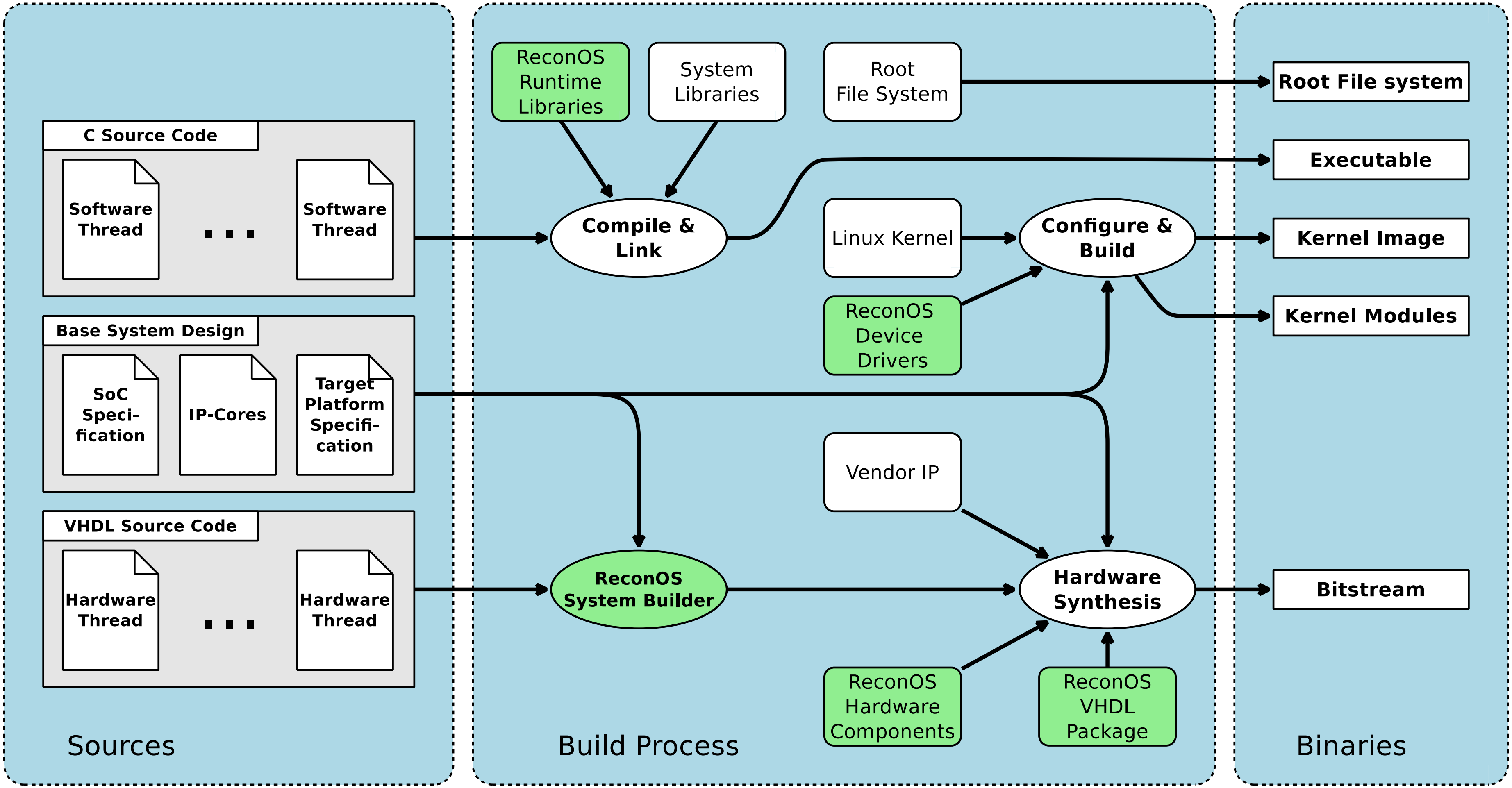


# Example ReconOS Architecture

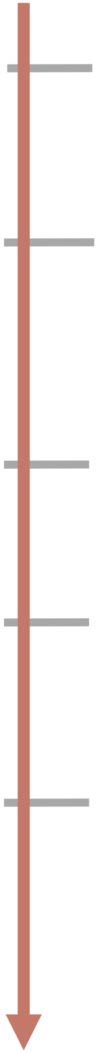


- Hardware threads can be loaded / removed by partial reconfiguration
- Hardware threads use cooperative multitasking

# ReconOS Toolflow

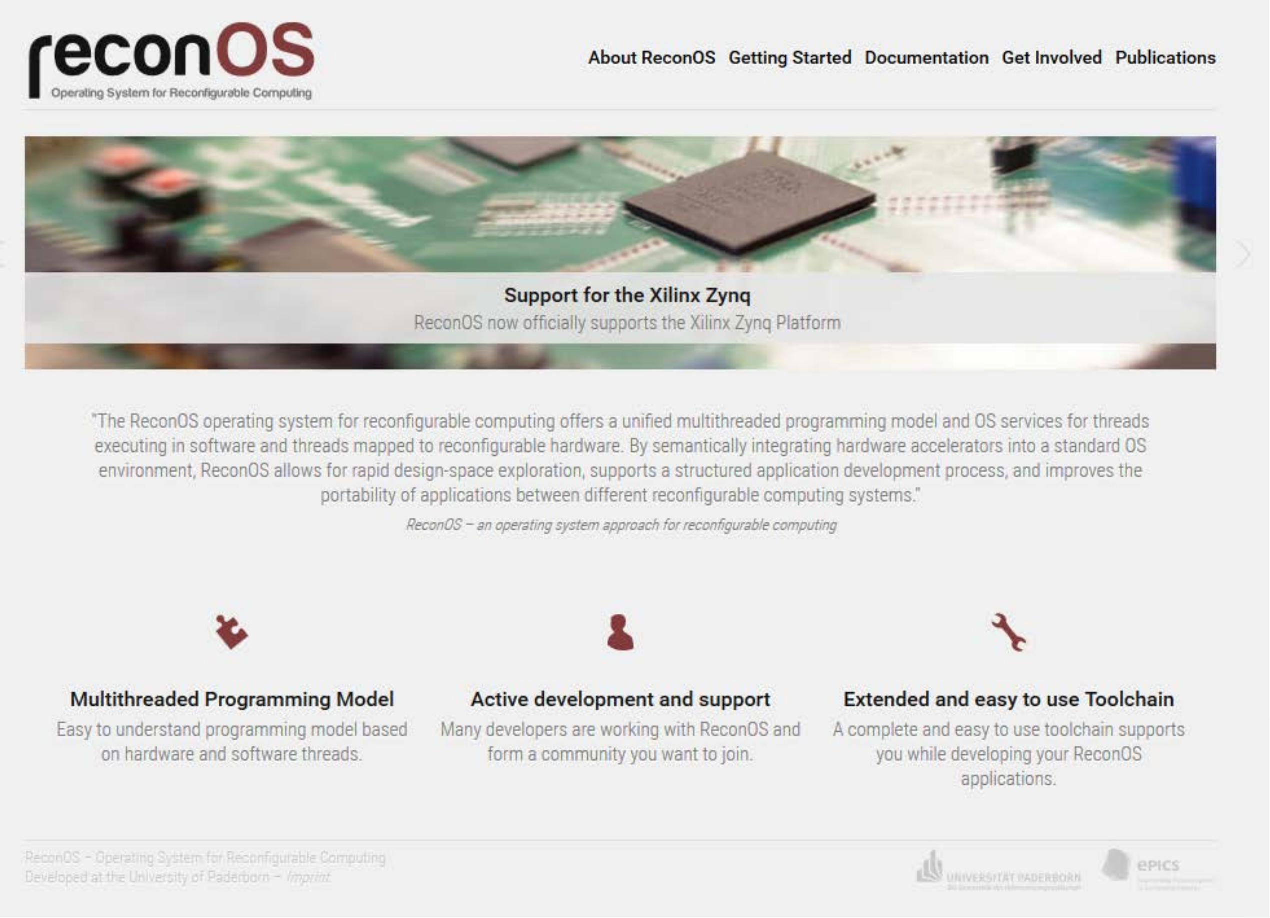


# ReconOS Versions



| | |
|---|---|
| Version 1.0 | eCos/PowerPC, Virtex-2Pro (XUPV2P), Virtex-2 (Erlangen Slot Machine) and Virtex-4 (Avnet Virtex-4 PCIe Kit, ML403) |
| Version 2.0 | Linux, eCos / PowerPC, Virtex-2Pro (XUPV2P) and Virtex-4 (ML403)<br>virtual memory support, FIFO interconnect |
| Version 3.0 | Linux, xilkernel / MicroBlaze, Virtex-6 (ML605) |
| Version 3.1 | **Linux / ARM, Xilinx Zynq (Zedboard)** |
| Version ? | Linux / ARM, Xilinx Zynq (Zedboard)<br>**Vivado HLS for hardware thread design,**<br>**direct communication between hardware threads** |

# www.reconos.de



reconOS

Operating System for Reconfigurable Computing

About ReconOS Getting Started Documentation Get Involved Publications

**Support for the Xilinx Zynq**

ReconOS now officially supports the Xilinx Zynq Platform

"The ReconOS operating system for reconfigurable computing offers a unified multithreaded programming model and OS services for threads executing in software and threads mapped to reconfigurable hardware. By semantically integrating hardware accelerators into a standard OS environment, ReconOS allows for rapid design-space exploration, supports a structured application development process, and improves the portability of applications between different reconfigurable computing systems."

*ReconOS – an operating system approach for reconfigurable computing*

**Multithreaded Programming Model**

Easy to understand programming model based on hardware and software threads.

**Active development and support**

Many developers are working with ReconOS and form a community you want to join.

**Extended and easy to use Toolchain**

A complete and easy to use toolchain supports you while developing your ReconOS applications.

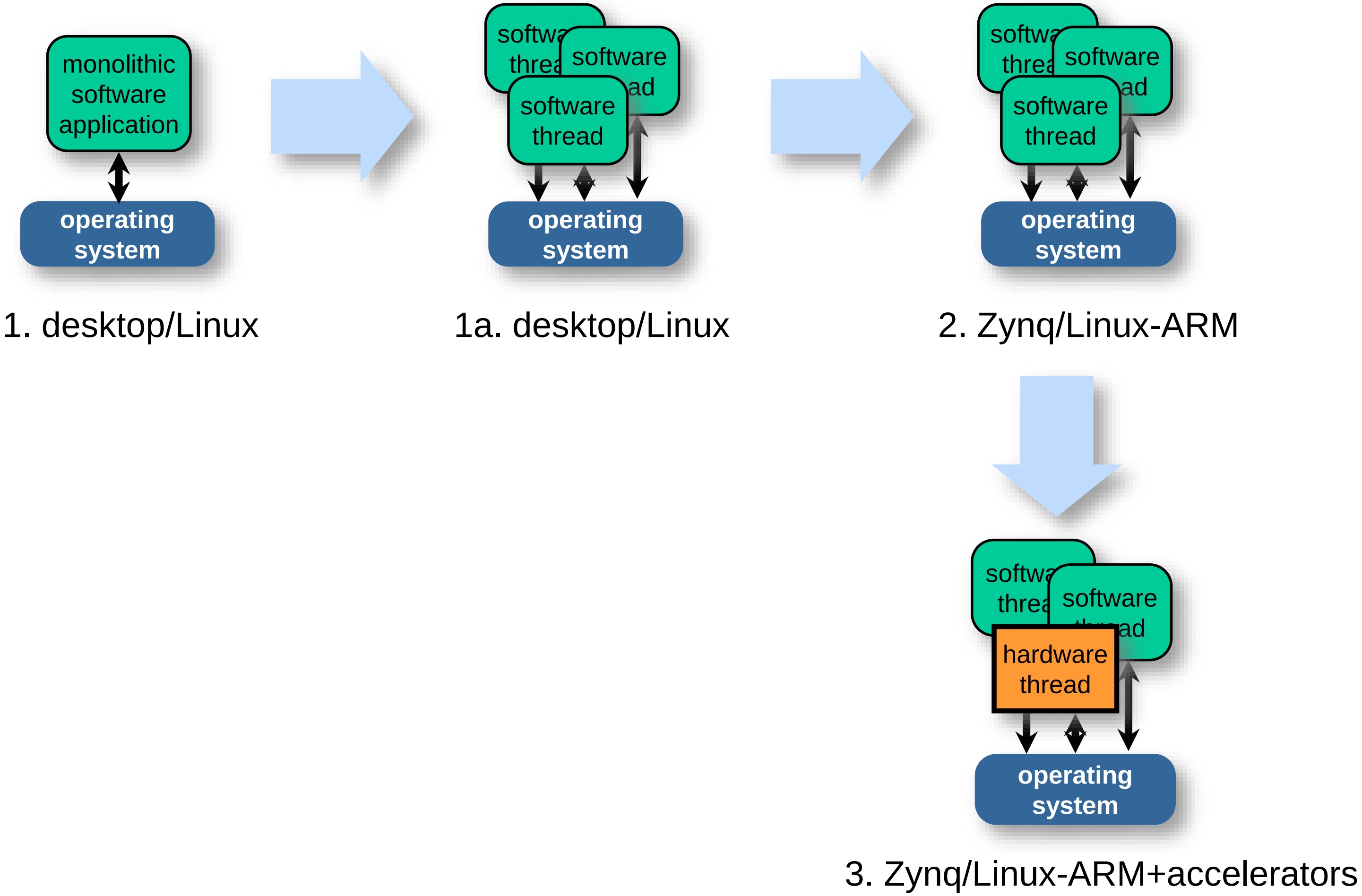
ReconOS – Operating System for Reconfigurable Computing
Developed at the University of Paderborn – *Imprint*

UNIVERSITÄT PADERBORN

EPICS

# Experience with ReconOS (1)

## 1. ReconOS supports a step-by-step application design process



1. desktop/Linux

1a. desktop/Linux

2. Zynq/Linux-ARM

3. Zynq/Linux-ARM+accelerators

# Experience with ReconOS (2)

## 2. ReconOS facilitates design space exploration

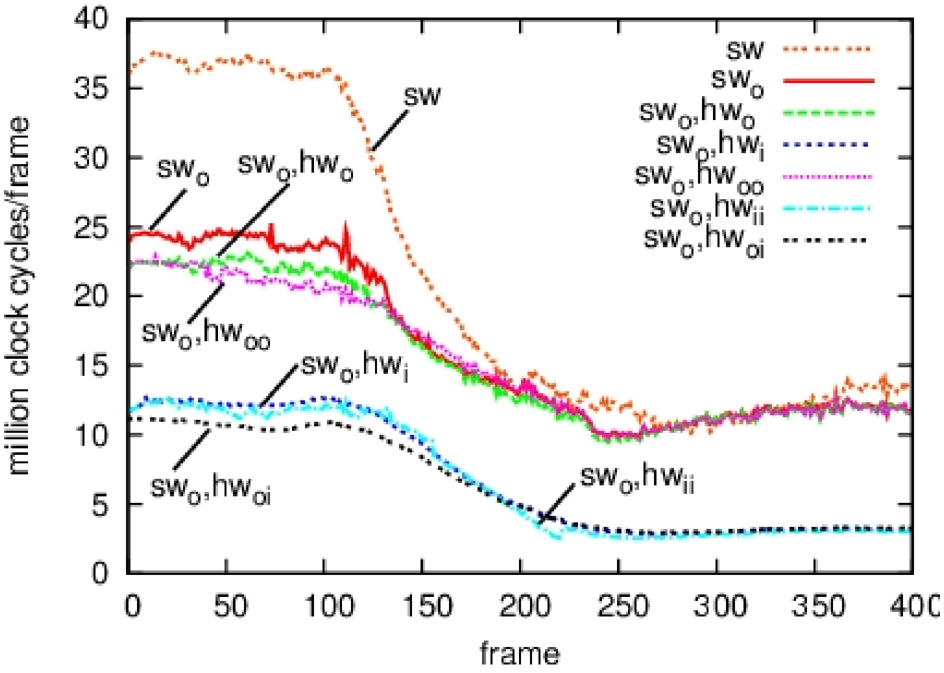
- Example: video object tracker
  - Virtex-4 FPGA (2 x PPC 405)
  - sw: all threads run in software
  - hw$_*$: a number of threads run in hardware
  - sw$_*$: a number of threads run on second (worker) CPU

frame 5

frame 90

frame 150

frame 260



# Experience with ReconOS (3)

## 3. ReconOS enables (self-)adaptive systems

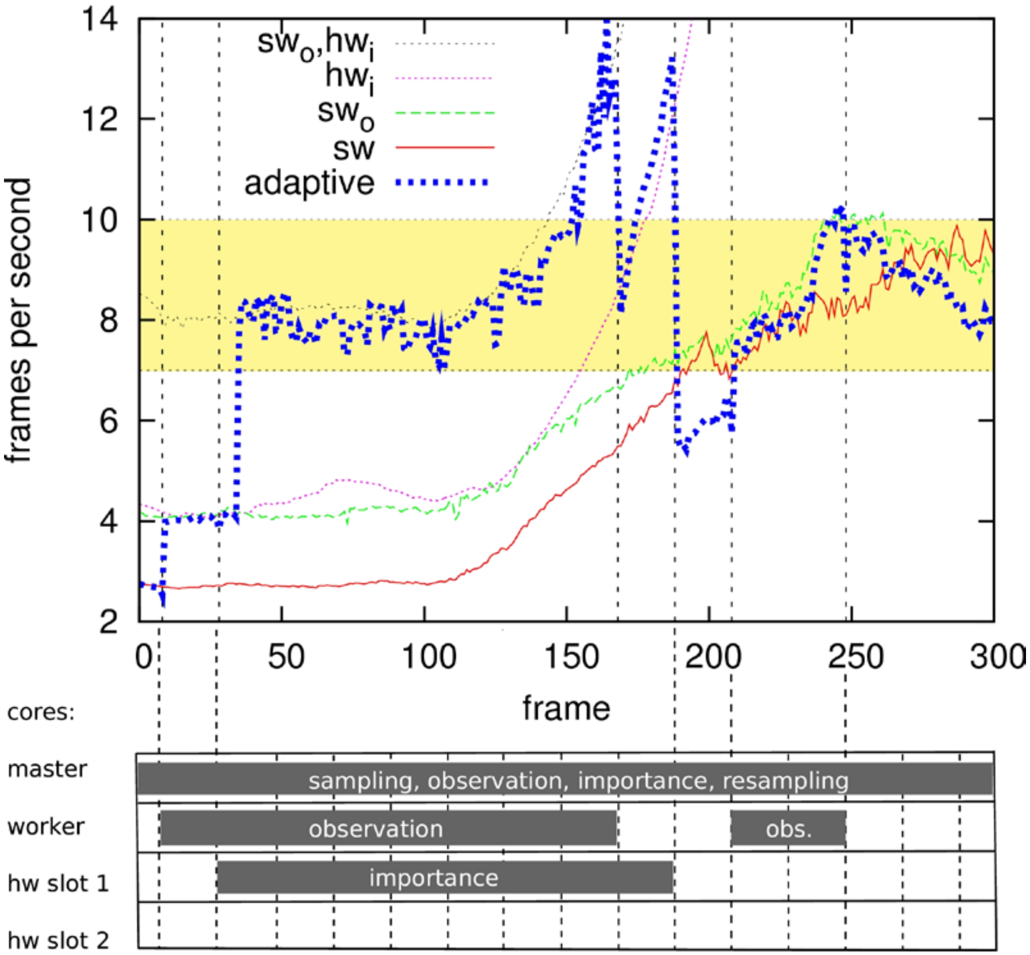
- Example: video object tracker
  - performance in [7,10] fps
  - minimize number of cores

frame 5

frame 90

frame 150

frame 260



# Why isn't ReconOS used more?

- ReconOS is an academic project
  - good as playground for research ideas, but we have limited resources for making it easily usable for others
  - out of the box only a few platforms are supported
  - more tutorials and examples needed on the website

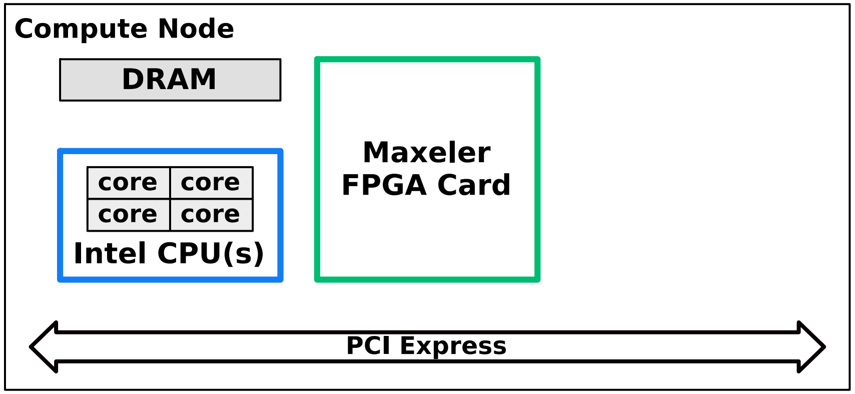
- ReconOS is (still) a complex environment
  - requires understanding of platform FPGA architectures and tool flows
  - requires some hardware design skills (for creating hardware threads)

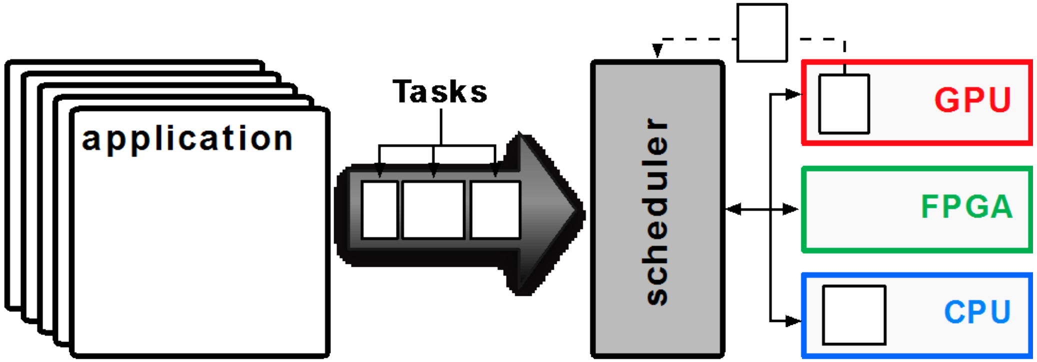
- Performance more important than productivity / flexibility
  - designers tend to optimize to the max, at the end they often have one big hardware thread and thrown away the OS abstractions

- The multi-threading model of ReconOS is obviously not suitable for all types of applications

# OS Abstractions for Heterogeneous Nodes

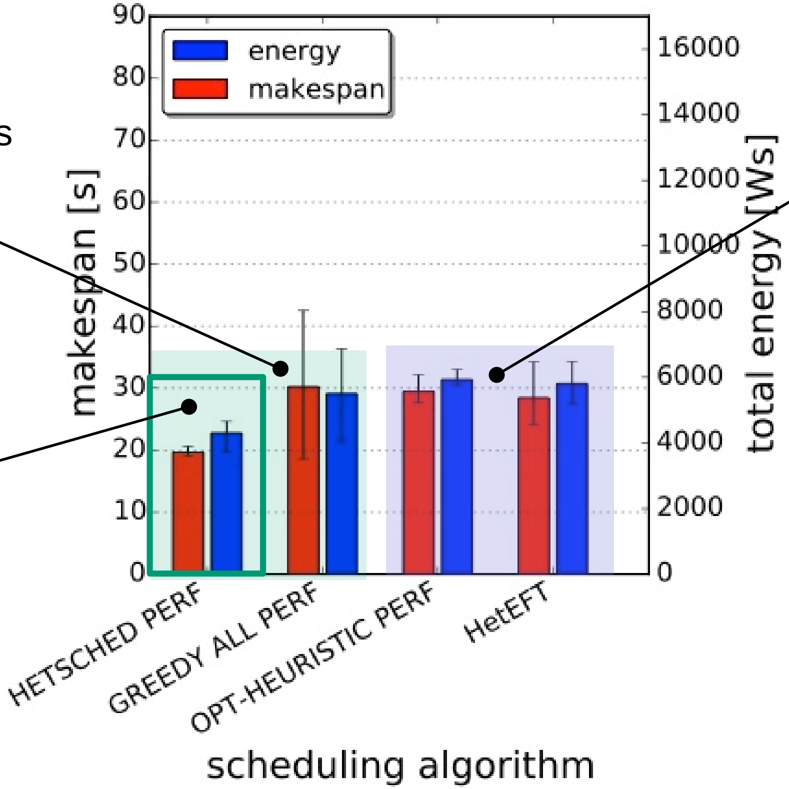
- Delegate threads, cooperative multitasking (like in ReconOS) for tasks on FPGA and GPU
- Allows for preemption and heterogeneous migration of tasks
  - based on a programming pattern with check-pointing and strip-mining





# Scheduling for Heterogeneous Nodes: HETSCHED

- Experiment
  - sets of 32 tasks: Heat Distribution, Correlation Matrix, Gauss Blur, Markov Chain
  - all tasks implemented on CPU, FPGA, GPU
  - all schedulers are work-conserving



based on task affinities

based on task runtimes

preemption and
heterogeneous
migration

# Summary: OS Services for FPGAs

- ReconOS: multithreaded programming for software and hardware
- Heterogeneous node: preemption and heterogeneous migration
- Does ReconOS get software programmers on FPGAs?
- Which OS services are useful for FPGAs in …
    - embedded systems
    - high-performance computing
    - warehouse scale computers

Thank You!

Questions?