
# *A software platform to support dynamically reconfigurable Systems-on-Chip under the GNU/Linux operating system*

26th July 2005

Alberto Donato
**donato@elet.polimi.it**

Relatore:
Prof. Fabrizio Ferrandi

Correlatore:
Ing. Marco Domenico Santambrogio

# Index

- Aim of the work
- The reconfigurable platform:
  - Hardware architecture
  - Partial dynamic reconfiguration flows
- Software architecture:
  - The ICAP kernel module
  - The IP-Core Manager
- Tests and results
- Future work

# Aim of the Work

FPGAs reconfiguration capabilities allow creation of Systems-on-Chip whose hardware components can be modified, added and removed at runtime.

Provide software support for **dynamic partial reconfiguration** on Systems-on-Chip running the **LINUX operating system**.

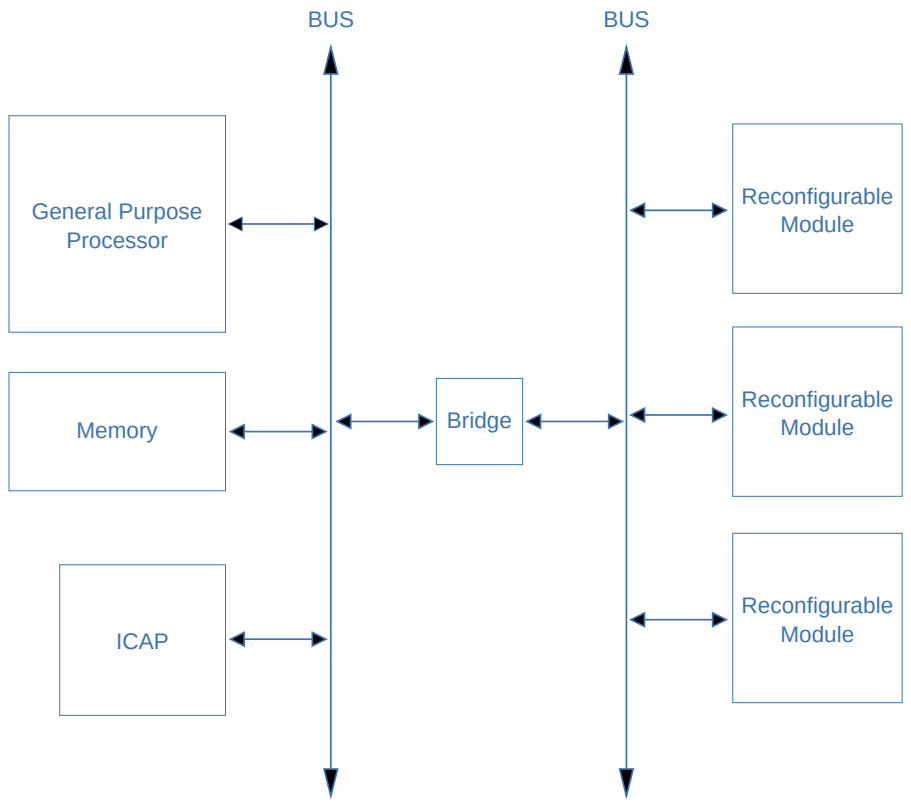
Issues:

- Partial reconfiguration process management from the OS
- Addition and removal of hardware reconfigurable components
- Automatic loading and unloading of specific drivers for the IP-Cores upon components configuration/deconfiguration
- Easier programming interface for specific drivers

# Hardware Architecture

Architecture derived from *Caronte*:

- General purpose processor
- Memory (BRAM, DDR)
- ICAP
- Reconfigurable components



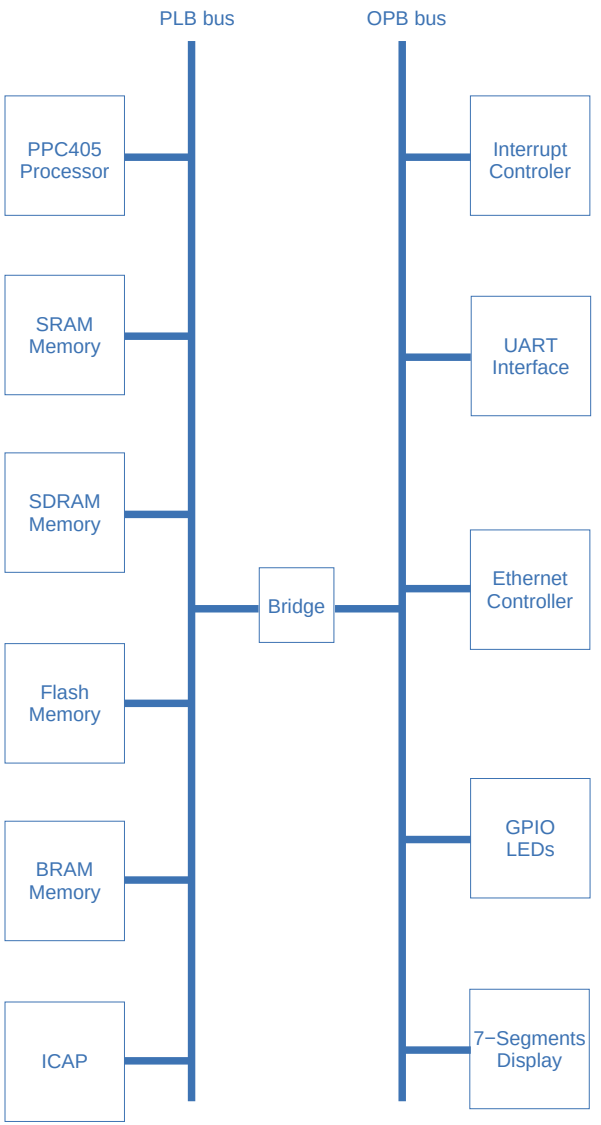
# Hardware Architecture (2)

Components of the hardware architecture:

- PPC405 processor
- SRAM and SDRAM memory
- Flash memory
- Xilinx ICAP IP-Core

I/O peripherals:

- Xilinx UARTLite interface
- Ethernet controller
- GPIO LEDS
- 7-Segments display



# Partial Dynamic Reconfiguration Flows

Reconfiguration of areas of the FPGA fabric without interfering with the rest of the system.

Two different approaches: small-bits and module-based reconfiguration.

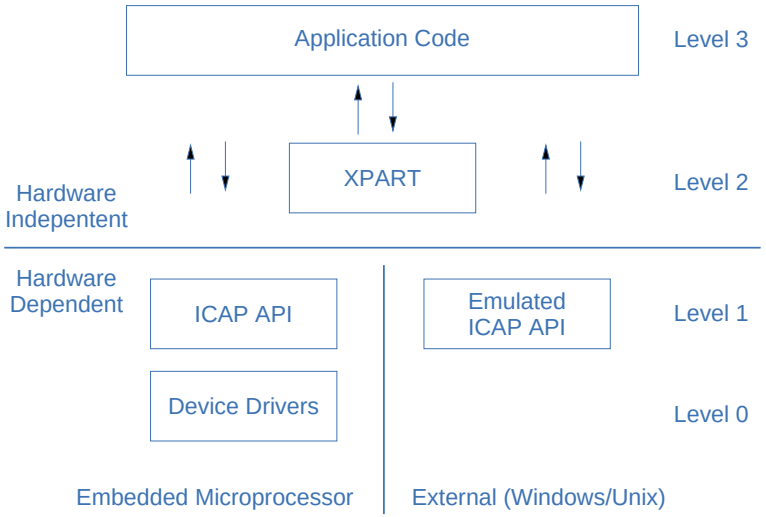
Small-bits manipulation:

- change single reconfigurable elements, such as slices
- useful to modify the configuration of hardware components of the system

Module-based reconfiguration:

- addition or removal of system components
- changes in the resources available to the system

# Software Side: Previous Work



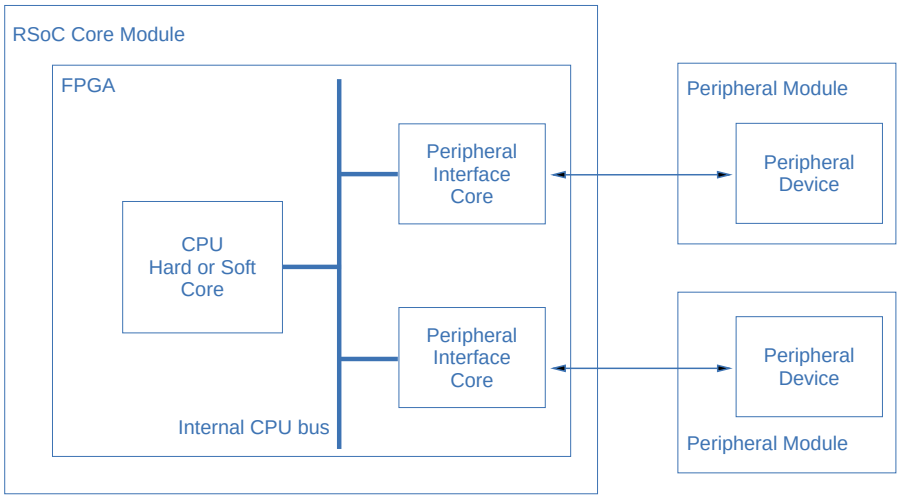

Self-reconfigurable platform:

- PowerPc/MicroBlaze processor
- ICAP component
- Stand-alone application using the XPART API

Supports small-bits reconfiguration.

Egret architecture:

- Modular hardware/software architecture
- Based on the LINUX OS
- Dynamic configuration of hard-ware modules and loading of drivers



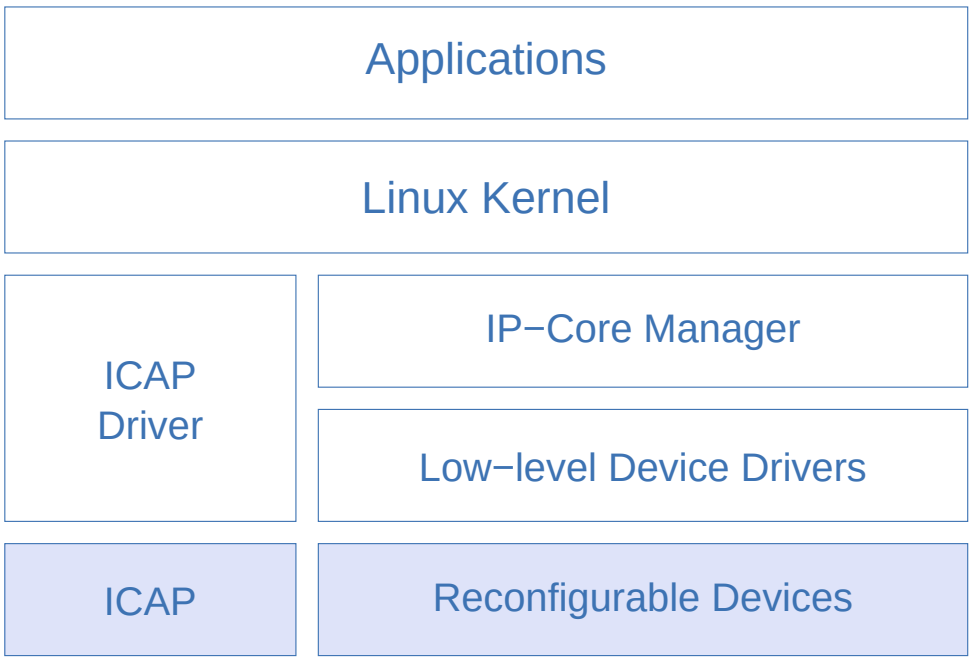
# Software Architecture

Composed of two main elements:

- Driver to support partial reconfiguration
- Manager for the IP-Cores devices

The software architecture provides:

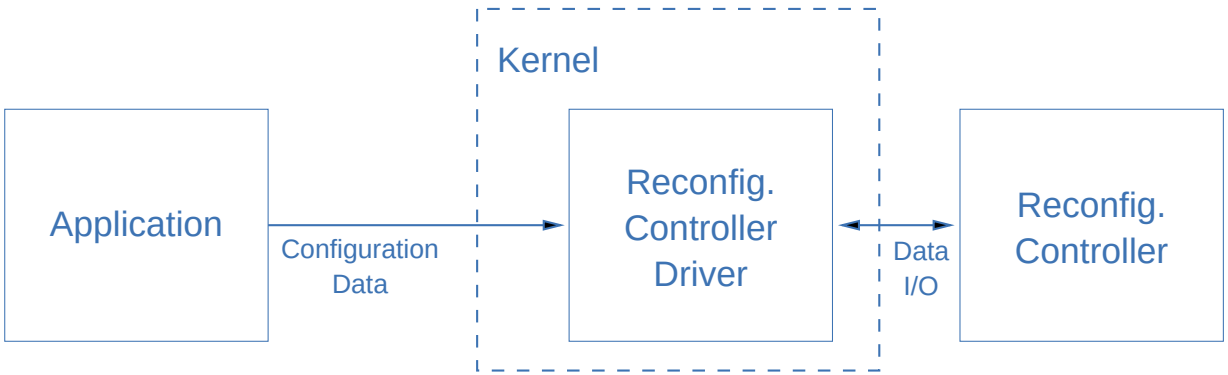
- Access to ICAP component from userspace
- Interface between IP-Cores low-level drivers and kernel
- Access to reconfigurable devices from userspace processes



# The ICAP Kernel Module

Implements a device driver, adds kernel support for the Xilinx ICAP component.

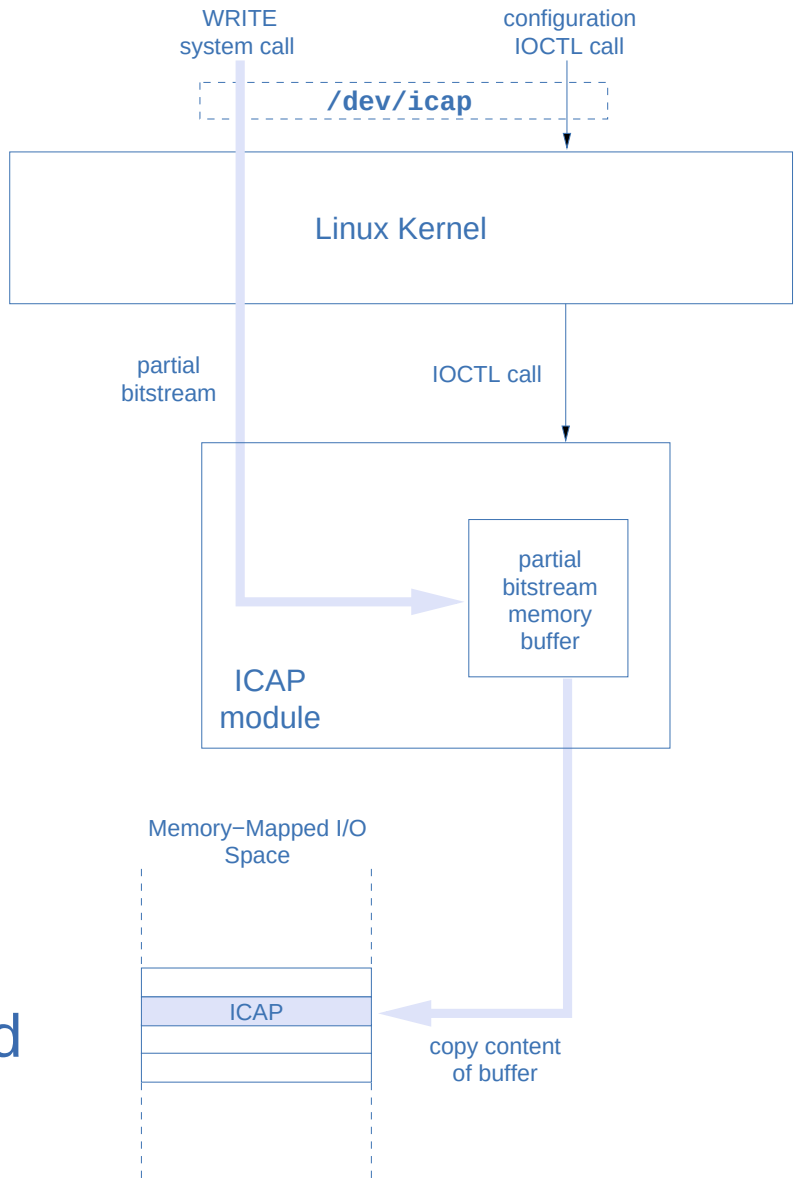
- Access from userspace via standard *device node* mechanism (i.e. `/dev/icap`)
- Masks hardware details
- Reconfiguration data provided in the form of *partial bitstream files*



# The ICAP Kernel Module: Reconfiguration

Reconfiguration process using the ICAP kernel module:

1. partial bitstream is copied into ICAP module buffer from a userspace process
2. reconfiguration `ioctl` call is performed from userspace
3. the kernel module sends partial configuration data to the ICAP component

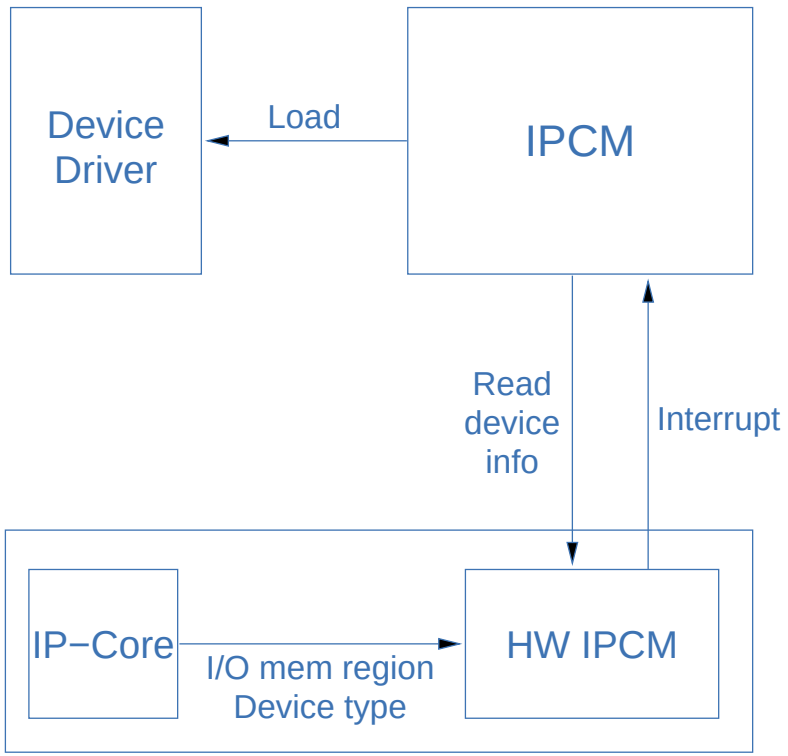



The hardware ICAP component is accessed through the *memory mapping* mechanism.

# The IP-Core Manager

A LINUX kernel module which implements a unified infrastructure for the management of the IP-Cores.

- IP-Cores *Plug-and-Play*
- Runtime loading of specific IP-Cores drivers
- Management of operations common to all drivers
- Access to reconfigurable components from userspace
- Standardize and simplify writing of specific drivers



# The IP-Core Manager: Hierarchy

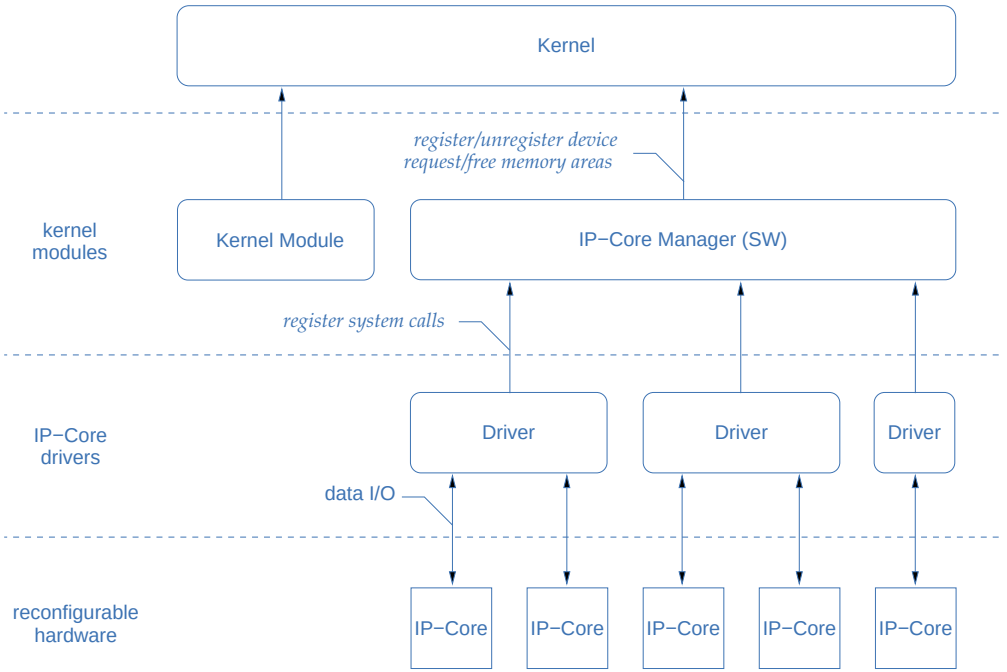
The IP-Core Manager acts as a *layer* between the operating system kernel and the low-level device drivers.

The low-level drivers contain:

- system calls implementation
- devices initialization and shutdown functions

The drivers also contain a *stub*:

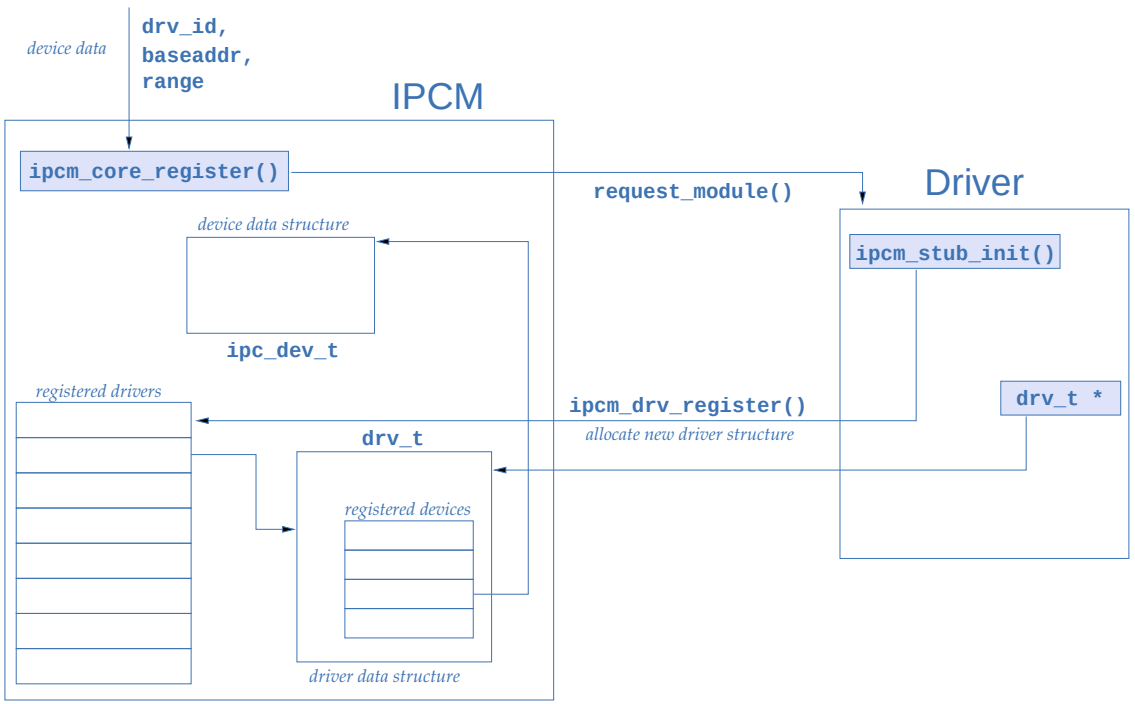
- provides standard kernel module interface
- provides module initialization and shutdown functions



# The IP-Core Manager: Structure

Registration process of a new IP-Core with the IPCM:

1. interrupt is received from the HW-IPCM
2. read device data (base address, device id, address range)
3. load low-level driver if not already loaded
4. the low-level driver initialization function registers driver data structures
5. data structures for the new device is initialized



## Comparison between original *Caronte* architecture and the one supporting LINUX

| Resource | Orignal Caronte | | Caronte LINUX | | Total |
|---|---|---|---|---|---|
| | Elem. | Perc. | Elem. | Perc. | available |
| Slice Flip Flops | 1843 | 18% | 2369 | 24% | 9856 |
| 4-input LUTS | 1727 | 17% | 2173 | 22% | 9856 |
| Occupied Slices | 1818 | 36% | 2262 | 45% | 4928 |
| Bonded IOBS | 107 | 27% | 168 | 42% | 396 |
| Block RAM | 32 | 72% | 32 | 72% | 44 |
| DCMS | 2 | 50% | 2 | 50% | 4 |

## Resources usage in different hardware architectures supporting LINUX

| Resource | Base Arch. | | No Ethernet | | No Flash | | Total |
|---|---|---|---|---|---|---|---|
| | Elem. | Perc. | Elem. | Perc. | Elem. | Perc. | available |
| Slice Flip Flops | 5079 | 51% | 2369 | 24% | 4668 | 47% | 9856 |
| 4-input LUTS | 5883 | 59% | 2173 | 22% | 5598 | 56% | 9856 |
| Occupied Slices | 4926 | 99% | 2262 | 45% | 4711 | 95% | 4928 |
| Bonded IOBS | 205 | 51% | 168 | 42% | 36 | 81% | 396 |
| Tbufs | 64 | 2% | 64 | 2% | 64 | 2% | 2464 |
| Block RAM | 36 | 81% | 32 | 72% | 36 | 81% | 44 |
| GCLKS | 6 | 37% | 4 | 25% | 6 | 37% | 16 |
| DCMS | 2 | 50% | 2 | 50% | 2 | 50% | 4 |

# Tests and Results (2)

## Small-bits manipulation partial reconfiguration tests

| Label | Reconfig. frames | Bitstream size (Byte) | Configuration time (msec) | Throughput (MByte/sec) |
|---|---|---|---|---|
| RC 0 | 33 | 24179 | 15.509 | 1.487 |
| RC 1 | 46 | 40171 | 25.674 | 1.492 |
| RC 2 | 50 | 44907 | 28.628 | 1.496 |
| RC 3 | 60 | 51347 | 32.700 | 1.497 |
| RC 4 | 68 | 53619 | 34.203 | 1.495 |
| RC 5 | 88 | 67099 | 42.814 | 1.495 |
| RC 6 | 104 | 60567 | 38.598 | 1.459 |
| RC 7 | 132 | 100595 | 64.140 | 1.496 |
| RC 8 | 148 | 94063 | 60.045 | 1.494 |
| RC 9 | 182 | 119319 | 76.049 | 1.496 |

Reconfiguration times have been measured on the *Avnet Virtex-II Pro Development Board*:

- with 66 Mhz processor: throughput ~1.5 Mbyte/s
- with 100 Mhz processor: throughput > 3.0 Mbyte/s

# Future Work

Possible extensions of the described software architecture:

- extension of the ICAP driver to allow creation of reconfiguration command from reconfiguration data (without pre-synthesized bitstreams)
- runtime calculation of difference bitstreams
- implementation of the HW-IPCM
- extension of the ICAP component to support DMA transfers