nano-tera.ch
swiss scientific initiative in health / security / environment systems

IcySoC RTD 2013

FNSNF

Ultra-Low-Power Processing Platform

Antonio Pullini[1], Frank K. Gürkaynak[1], Luca Benini[1],
Adam Teman[2], Jeremy Constantin[2], Andreas Burg[2]

[1]Integrated Systems Laboratory, ETH Zürich, [2]Telecommunications Circuits Lab, EPFL

ETH Zürich
EPFL ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

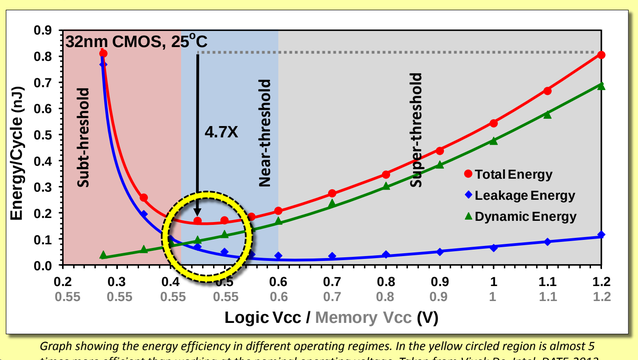
PULP Architecture

PULP stands for **P**arallel **U**ltra-**L**ow **P**ower Processor Architecture and is actively being developed by the Integrated Systems Laboratory of ETH Zürich. Our goal is to develop a system that has the same energy efficiency regardless of the computational load. We call this property **energy proportionality**. Our system works very well when there is little to do but is equally efficient when the work load increases. This is different form other processor systems which are optimized to work well at one corner, but do not scale well.

1. To allow us to scale efficiently, we have designed a many-core architecture organized in clusters.
2. Each cluster consists of several simple processor cores. Our current system is based on an open source architecture called OpenRISC
3. All processor cores within a cluster have access to a common data memory that we call **T**ightly-**C**oupled **D**ata-**M**emory (TCDM)

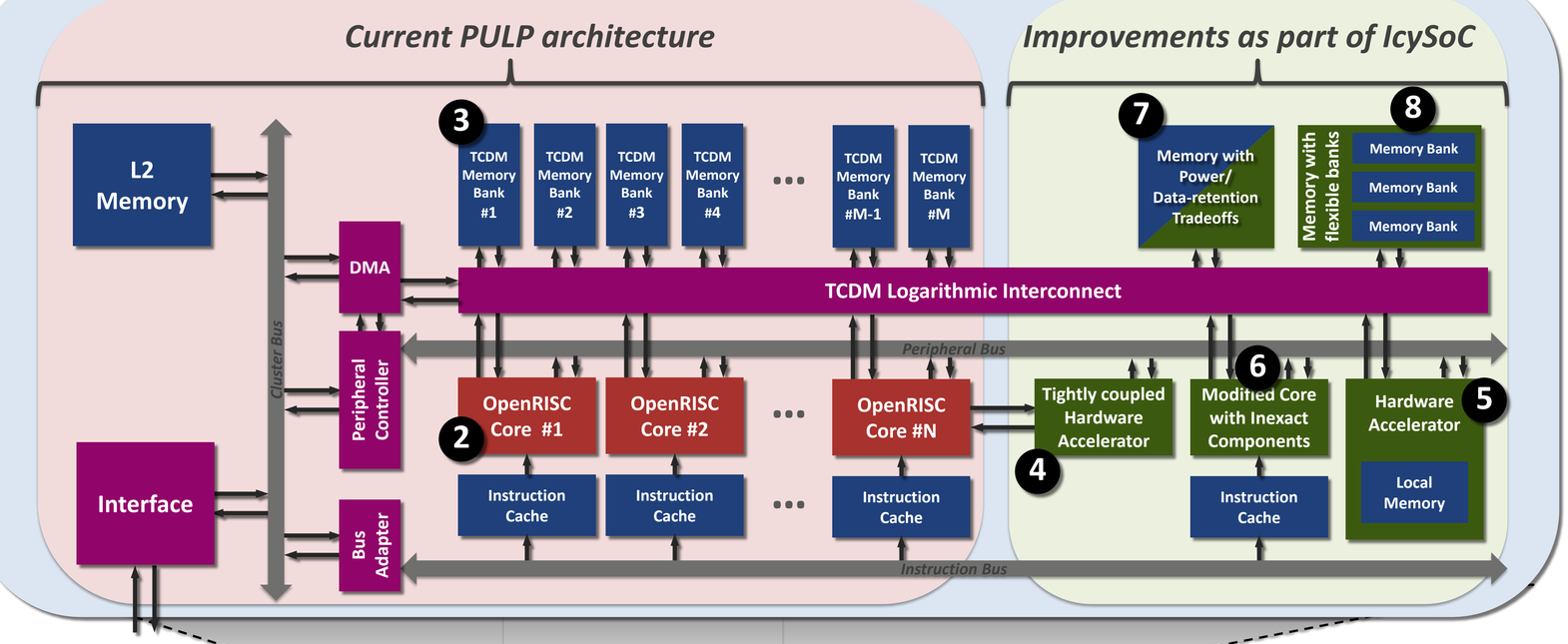
How do we achieve energy efficiency?

We have designed the architecture so that it can work at the near/sub-threshold operation mode. At this mode, the circuits work slower, but they are more energy efficient (see below). We can make up the speed deficit by having multiple parallel cores. In addition we also have the capability of switching cores on/off, and using body biasing techniques to improve energy efficiency.

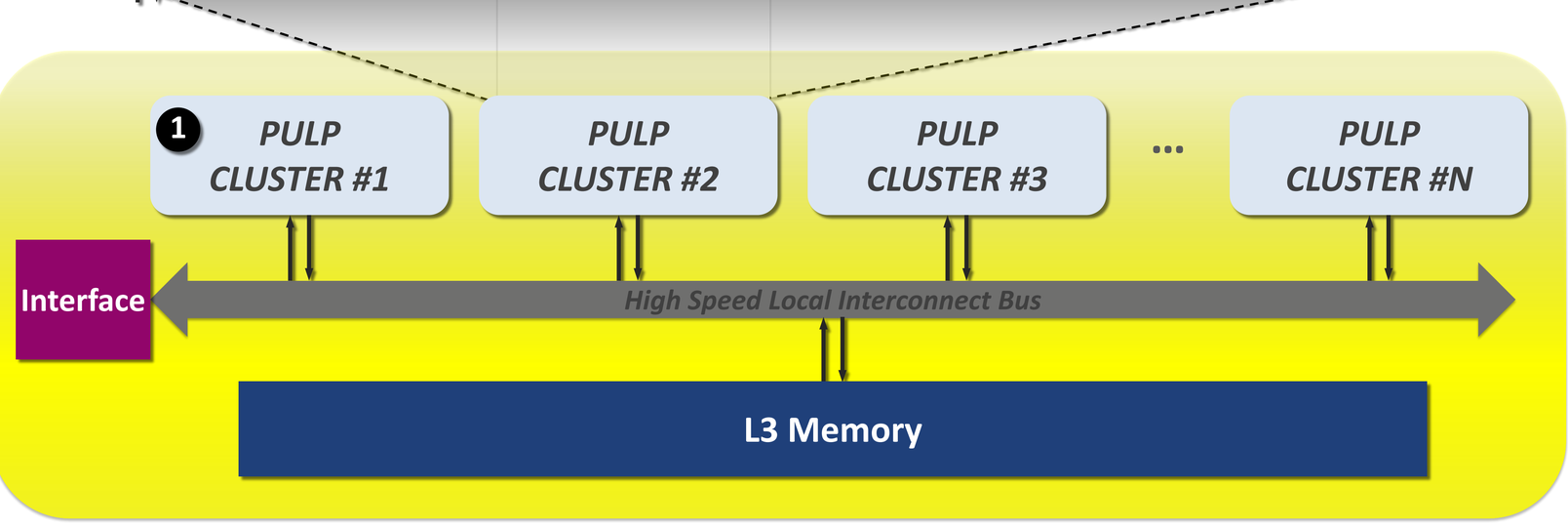


Graph showing the energy efficiency in different operating regimes. In the yellow circled region is almost 5 times more efficient than working at the nominal operating voltage. Taken from Vivek De, Intel, DATE-2013

Current PULP architecture



Improvements as part of IcySoC



What will we do in IcySoC?

In the IcySoC project we will combine two exciting ideas for more efficient processing platforms. Operating at the near/sub threshold and using inexact computing. We will use PULP based systems for our investigations. As part of the project we envision several modifications to the original PULP architecture.

4. The performance of the system can be improved by developing dedicated **hardware accelerators** that can calculate certain operations faster than what can be done in a standard processor. These can be **tightly coupled** to the processor.
5. It is also possible to design hardware accelerators that work more independently and even have access to their own local memory.
6. **Inexact computing** allows a trade-off between the accuracy of calculated results and the performance metrics, such as energy. It is possible to design more energy efficient processor cores if the calculations are allowed to have an occasional small mistake. We will also investigate how reliable operations can be performed while using such inexact components throughout the system.
7. Similar **energy-accuracy trade-offs** can also be made for memories. Memories that consume less power can be designed if they are occasionally allowed to forget certain bits.
8. We also will investigate **fine-grained memory banking** methods that will allow us to configure the TCDM memory depending on the actual workload, reducing the contribution from idle memories.

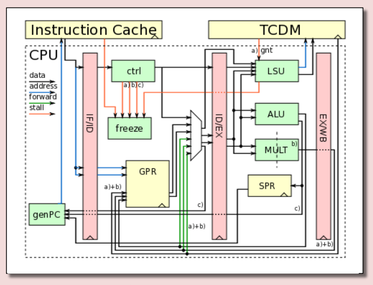
A better processing core

In the current PULP architecture we are using the OpenRISC architecture. As this project is open source, it allows us to share our platform without problems to all project partners.

There is also a publicly available HDL implementation of the OpenRISC architecture. Under ideal conditions such a processor is expected to execute 1 **i**nstruction **p**er **c**lock cycle (IPC). We soon realized that this is not the case for this implementation.

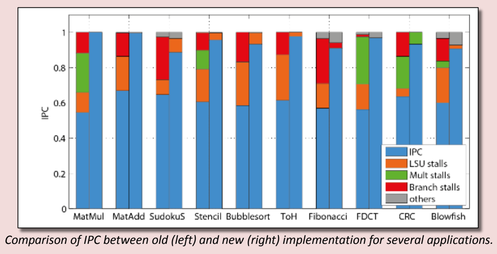
The original architecture had three problems, effectively reducing its IPC:

- Reading and writing to memory was slow (LSU Stalls).
- Multiplications took 3 cycles
- Time was lost during branches



Block diagram of the improved OpenRISC processor

We improved this implementation and the new micro-architecture has much better performance. The graph below shows a comparison of the IPC between the old (left) and new (right) implementation. The IPC is very close to the ideal 1 in our new design.



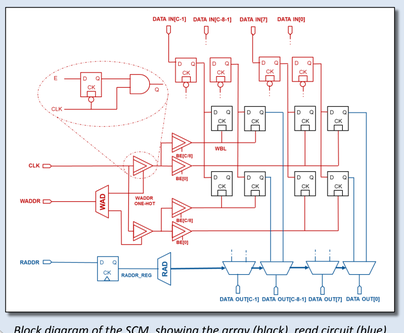
Comparison of IPC between old (left) and new (right) implementation for several applications.

New Memory Styles

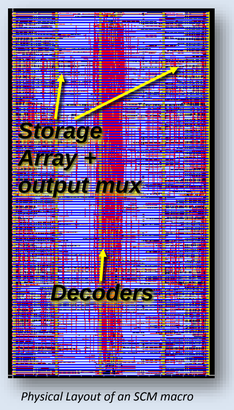
Memories occupy a large part of any processing system. They not only occupy circuit area, but also contribute significantly to the power consumption. This is why it is important to optimize the memory hierarchy in a system as much as possible.

Typically local memories in such systems are built using standard SRAMs (**S**tatic **R**andom **A**ccess **M**emory). For large memory sizes these SRAM blocks offer high density, however when smaller memory blocks are used, they have some overhead. Most importantly, it is not very easy to scale down the operating voltage of the SRAM macros, which prevents us from operating in the near-threshold region.

One of the solutions we are investigating is the use of **S**tandard **C**ell based **M**emories (SCM) to help us with these problems in applications where smaller memories are needed. For a 64x64 bit memory, our SCM is operational at lower voltages and consumes up to 5 times less energy than comparable SRAMs.



Block diagram of the SCM, showing the array (black), read circuit (blue) and write circuit (red)



Physical Layout of an SCM macro

Timeline for ASICs

as part of the IcySoC project ...