Raising energy modelling of embedded software to the multi-core network and system level

Steve Kerrison

University of Bristol

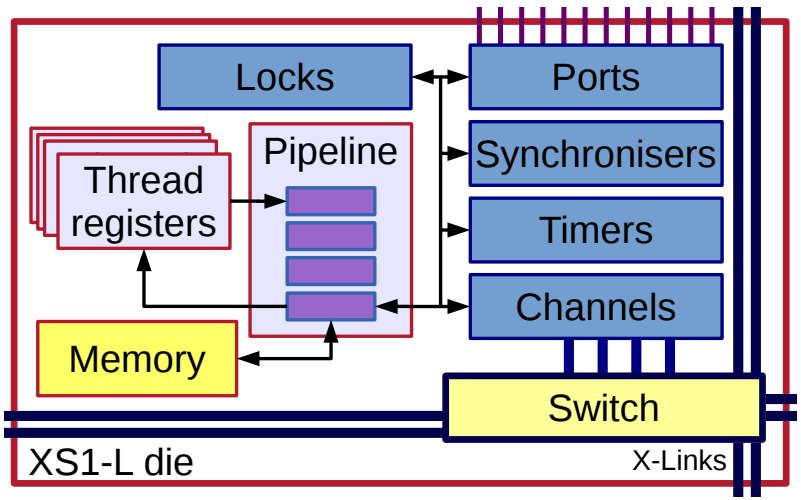
ENTRA workshop, Malaga, 6th May 2015

In this presentation

- **Xcore architecture refresher.**
- **A brief overview of our multi-threaded energy model.**
- Requirements for a multi-core model.
- Experimentation with the Swallow many-core platform.
- System level energy consumption view.
- Challenges and next steps.

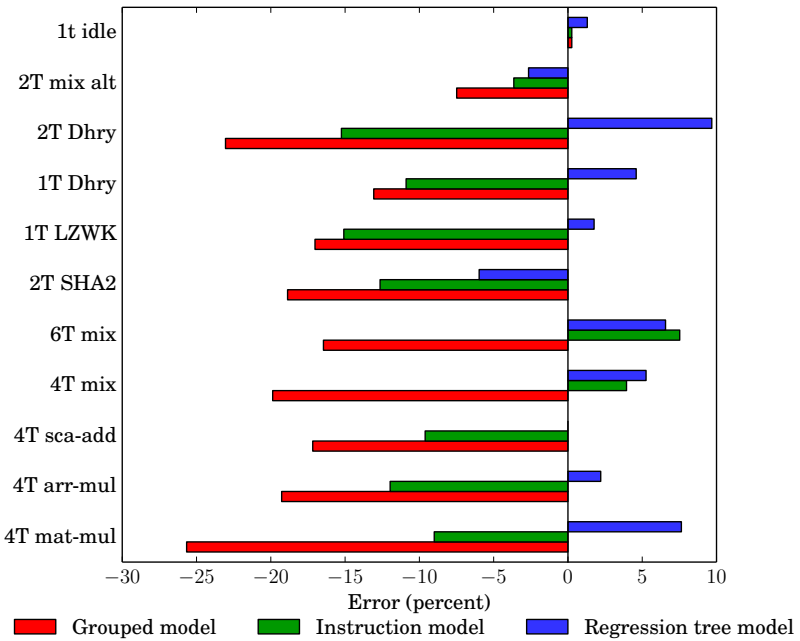
XS1-L Xcore: real-time, multi-threaded, embedded

Key features of the XS1-L:



- Four-stage pipeline, round-robin *hardware thread scheduling* with *no data hazards*.
- Single-cycle Static-RAM, predictable instruction fetching and execution time.
- Onboard peripherals (ports, timers, ...) accessible via ISA, *not memory mapped*.
- Channel communication paradigm baked into the architecture, reflected in the `XC` programming model.
- Operating frequency up to 500 MHz.

Multi-threaded energy model [1]



- Simulate a program to get instruction trace, or determine instruction trace statically.
- Consider concurrency (num. of active threads when an instruction is executed).
- Use Eq. 1, backed by energy data collected through profiling and measurement.

Model equation

$$E_{\mathsf{p}} = P_{\mathsf{base}} N_{\mathsf{idle}} T_{\mathsf{clk}} + \sum_{t=1}^{N_t} \sum_{i \in \mathsf{ISA}} \left(\left(M_t P_i O + P_{\mathsf{base}} \right) N_{i,t} T_{\mathsf{clk}} \right) \qquad (1)$$

What can our model... model?

The model works at multiple levels:

- Simulation trace
- Simulation statistics
- Static analysis [2]
- LLVM-IR (via mapping) [3]

What can our model... model?

The model works at multiple levels:

- Simulation trace
- Simulation statistics
- Static analysis [2]
- LLVM-IR (via mapping) [3]

And captures lots of program behaviours:

- Single-threaded
- Multi-threaded
- Shared memory & channel communication (core-local)
- Event-driven, supported by Xcore resources such as timers

In this presentation

- Xcore architecture refresher.
- A brief overview of our multi-threaded energy model.
- **Requirements for a multi-core model.**
- **Experimentation with the Swallow many-core platform.**
- **System level energy consumption view.**
- Challenges and next steps.

Requirements for a multi-core model

For multi-core, we must add the capability to:

- Represent the structure of the hardware platform and the software running upon it.
- Profile and model the cost of communication between Xcores, in terms of *time and energy*.
- Identify communication within the software that we want to analyse.

Requirements for a multi-core model

For multi-core, we must add the capability to:

- Represent the structure of the hardware platform and the software running upon it.
- Profile and model the cost of communication between Xcores, in terms of *time and energy*.
- Identify communication within the software that we want to analyse.

We achieve this with:

- Graph-based representation of system, including support for energy consumption visualisation.
- Device profiling with the Swallow many-core (16+) platform.
- Modifications to the `axe` simulator for the Xcore.
- The channel communication paradigms provided by `XC` and the XS1 ISA.

Channel communication explained (concept)

```
1 getr r0,2          #Get chanend
2 ldw  r1,cp[0]      #Load dst
3 setd res[r0],r1    #Set dst
4 out  res[r0],r0    #TX word
```

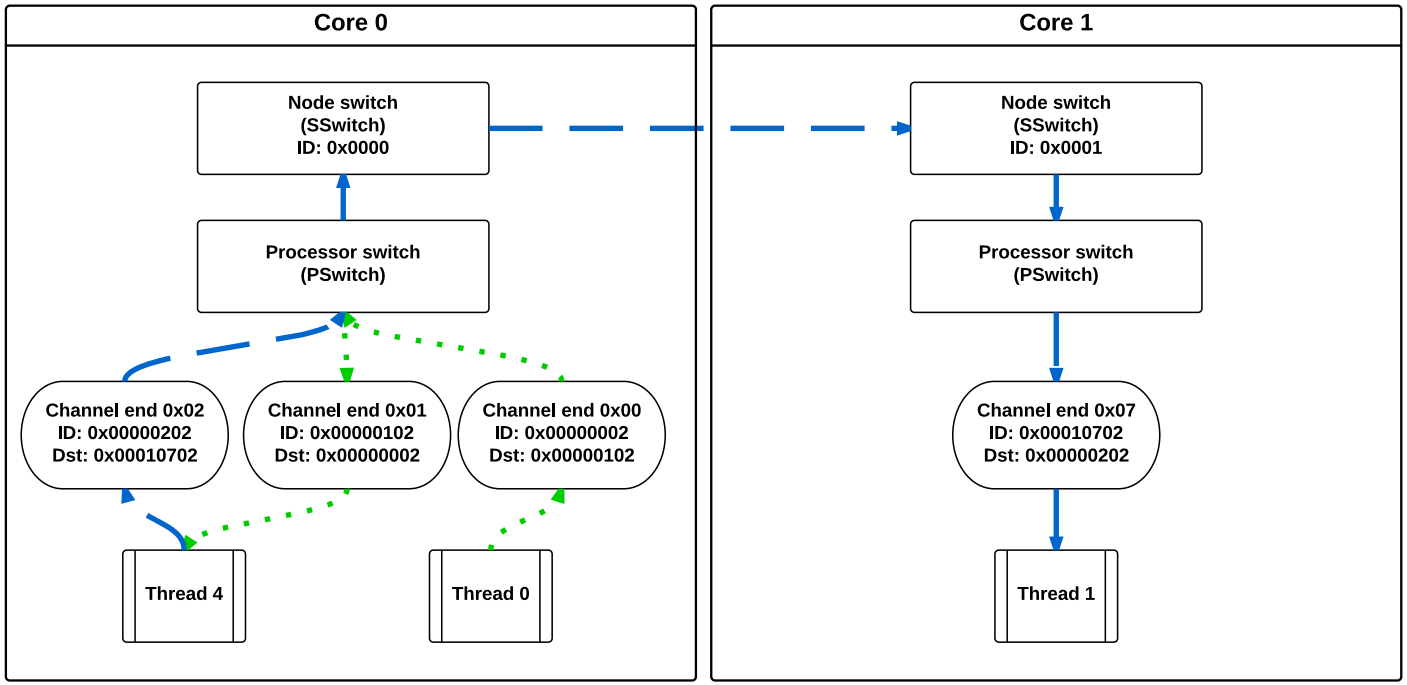
Listing 1: Sending on a channel.

```
1 getr r0,2          #Get chanend
2 ldw  r1,cp[0]      # Load dst
3 setd res[r0],r1    # Set dst
4 in   r0,res[r0]    # RX word
```

Listing 2: Receiving on a channel.

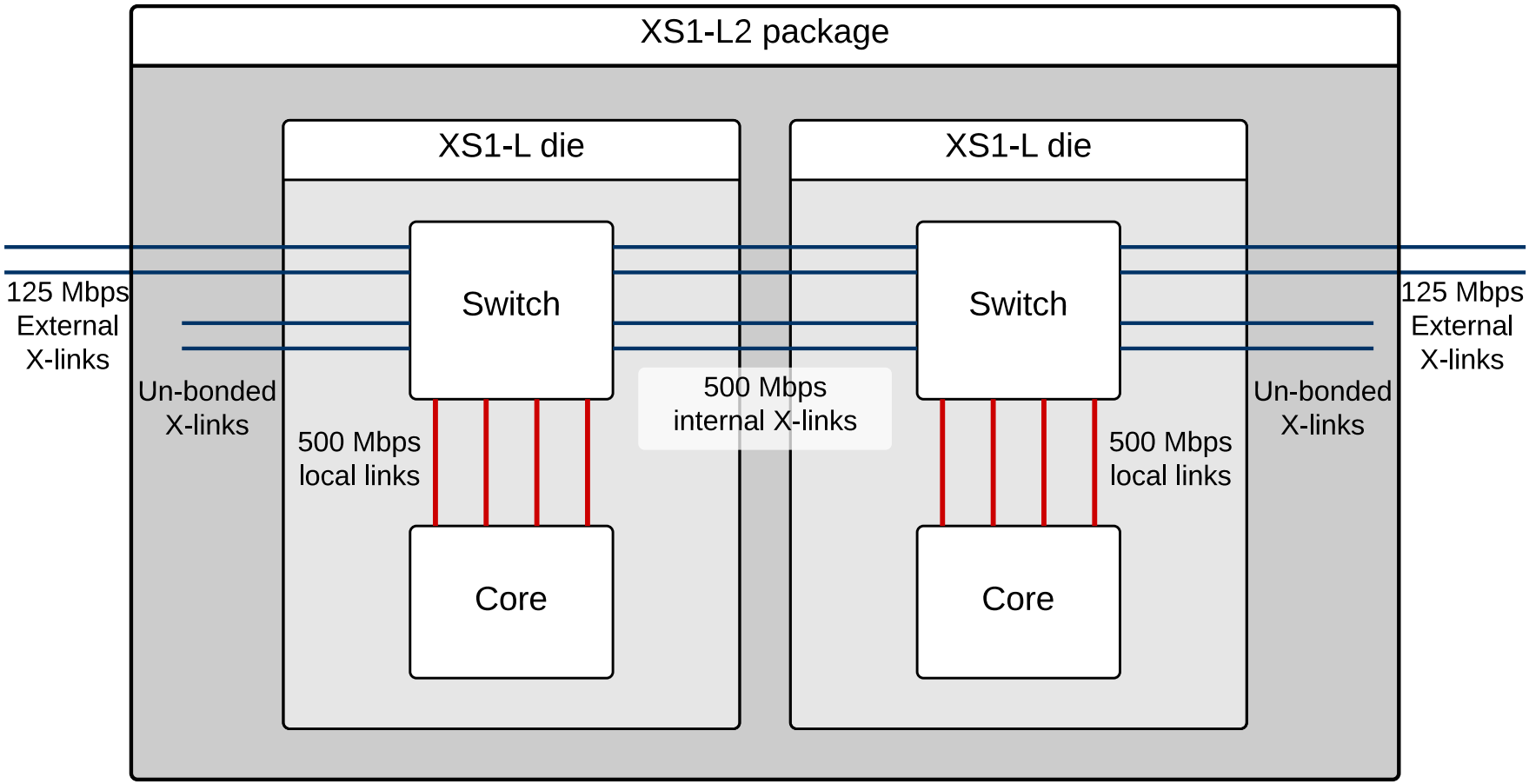
- Channel ends are a *resource* obtainable in the ISA.
- Destination is configurable.
- 32 channel ends per core.
- Enables on-core multi-threaded communication...
- ...as well as communication with other cores.

Channel communication explained (concept)

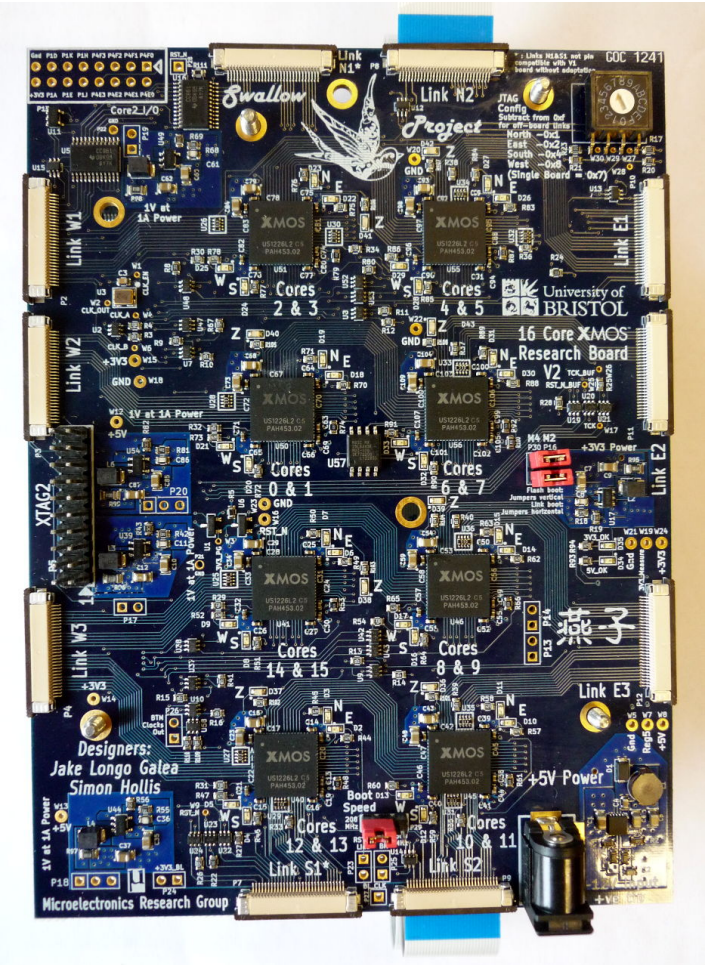


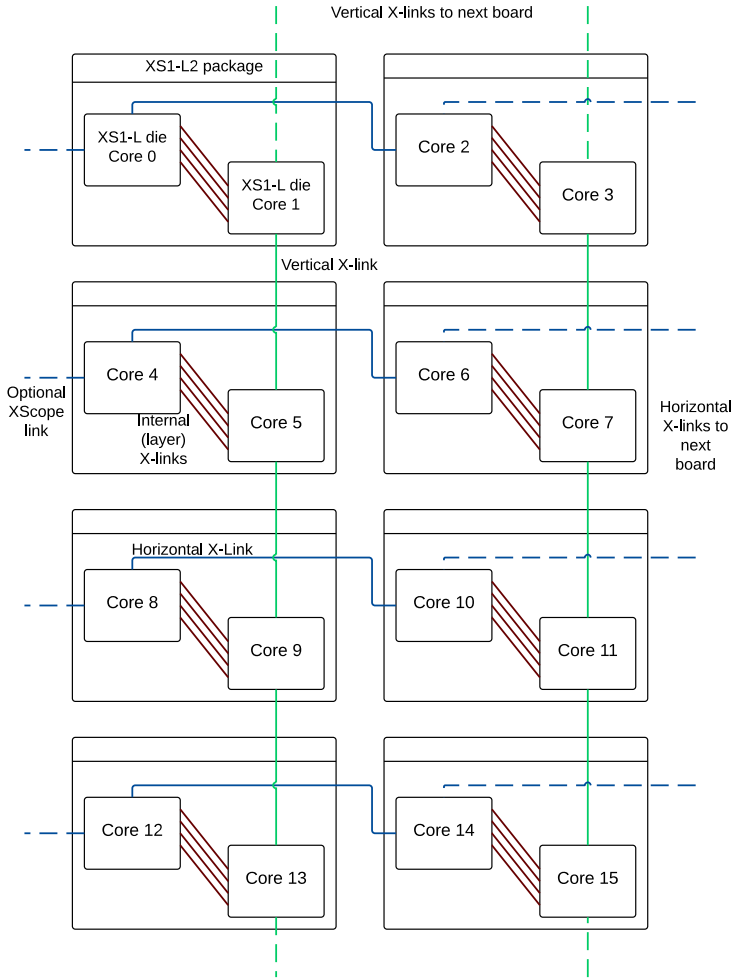
- `out` and `in` instructions in ISA used for communication.
- `outct` and `chkct` used for protocol.
- Header sent implicity with first `out`.
- Links held open until an appropriate `outct` instruction is sent.

Channel communication explained (physical)



Swallow [4] project experiments

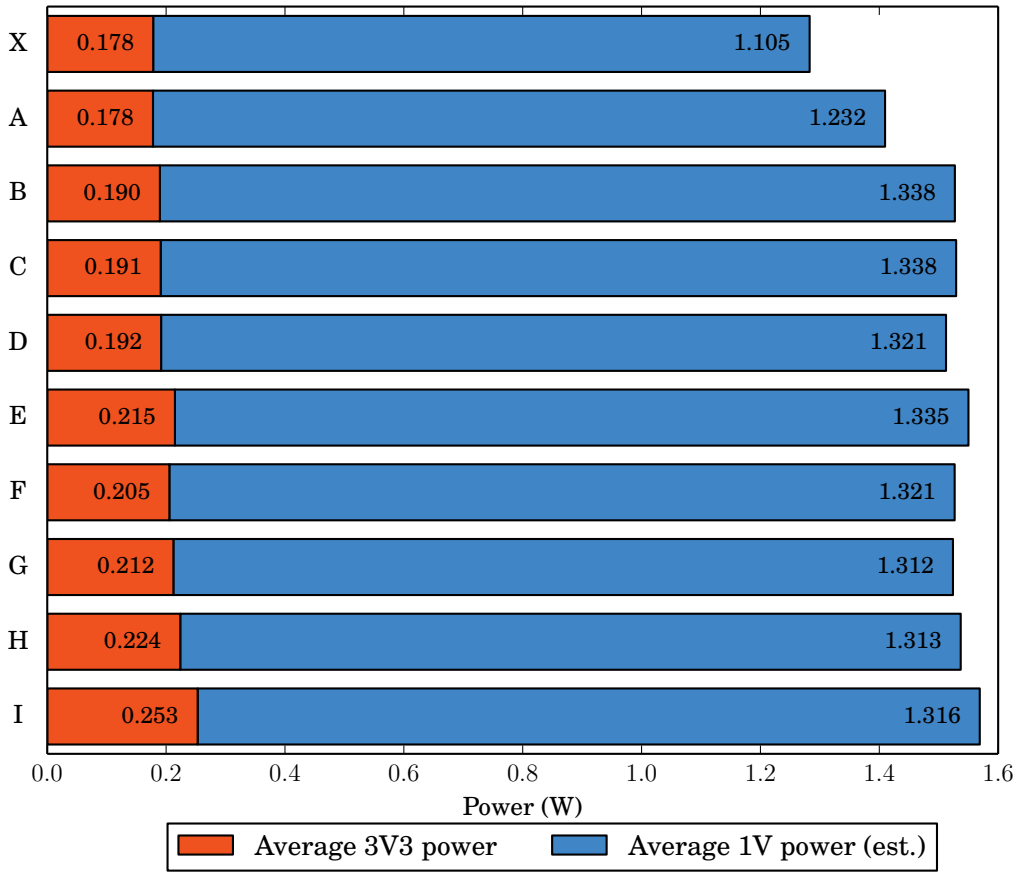




Swallow project experiments

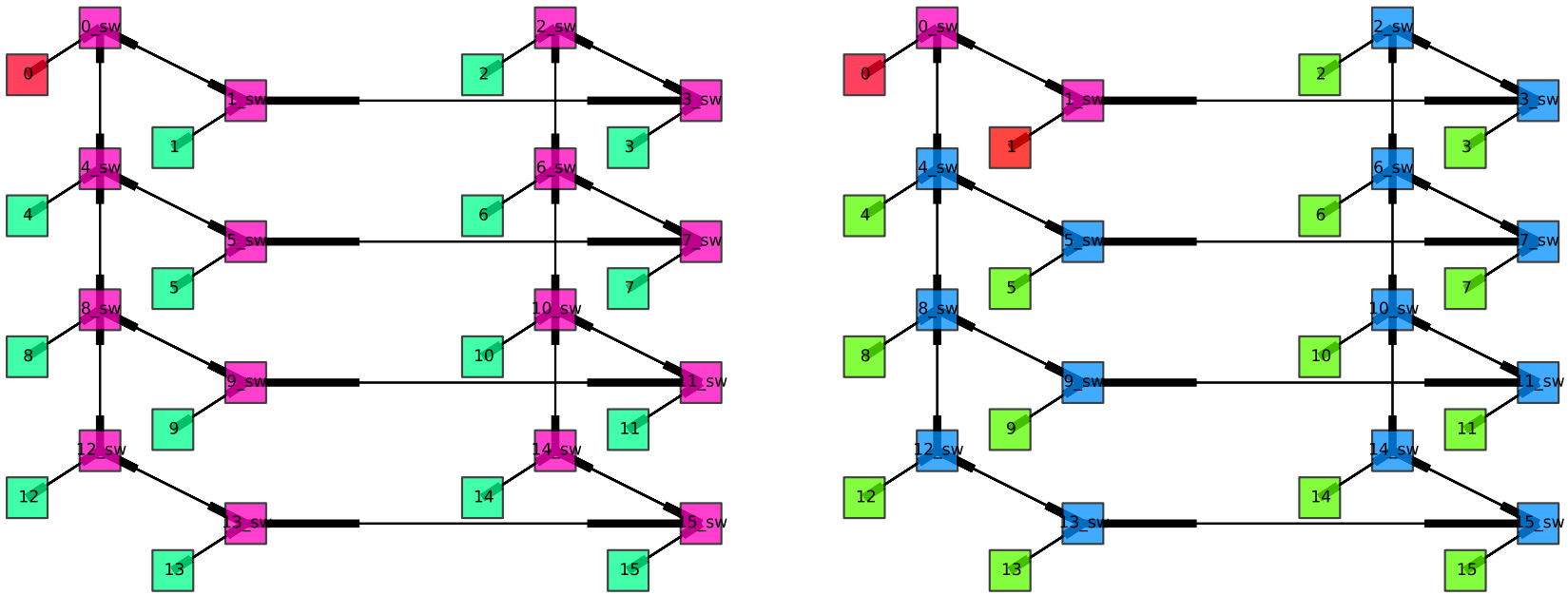
- Tests for different communication distances.
- 3.3 V power in orange (leftmost), captures link communication cost.
- 1.0 V power in blue (rightmost), captures routing and computation cost.

The cost to communicate can be determined, but what about the static + dynamic power dissipated by cores that are *waiting* for data? Time is very important!



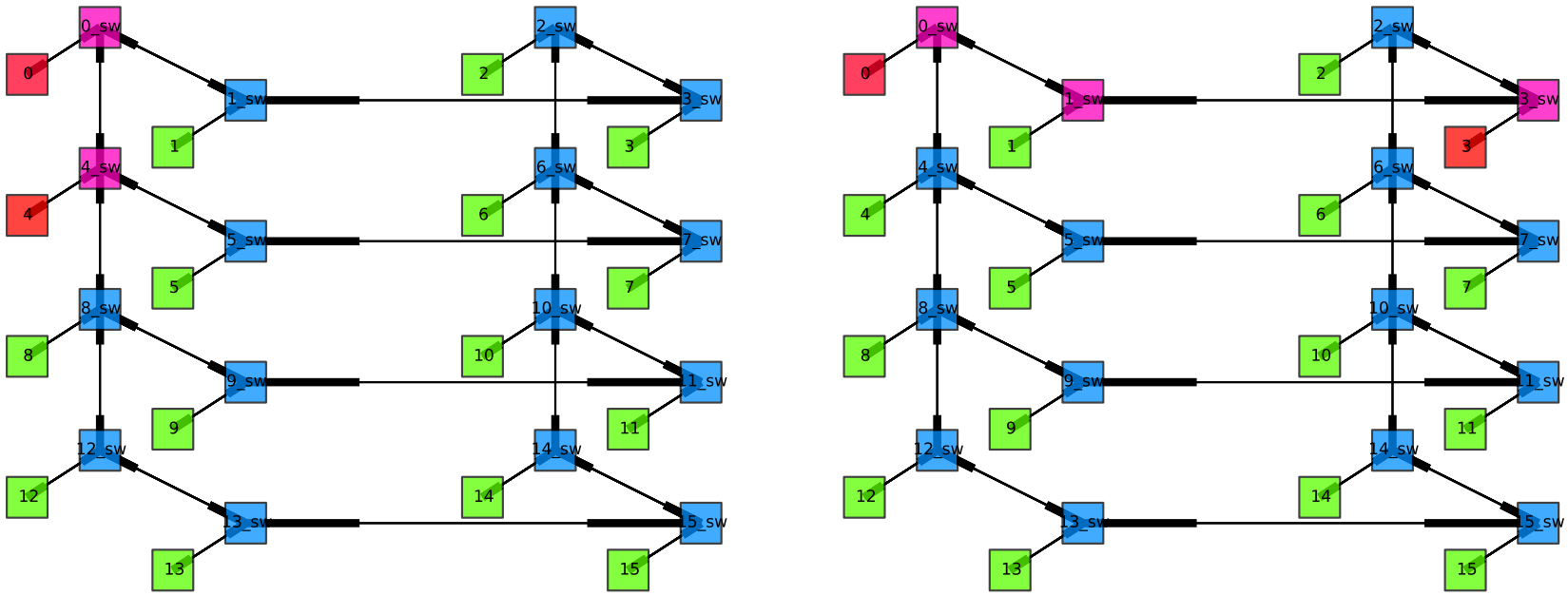
Visualising multi-core energy consumption

Communication between threads on same or different cores.



Visualising multi-core energy consumption

Communication between threads on different cores.

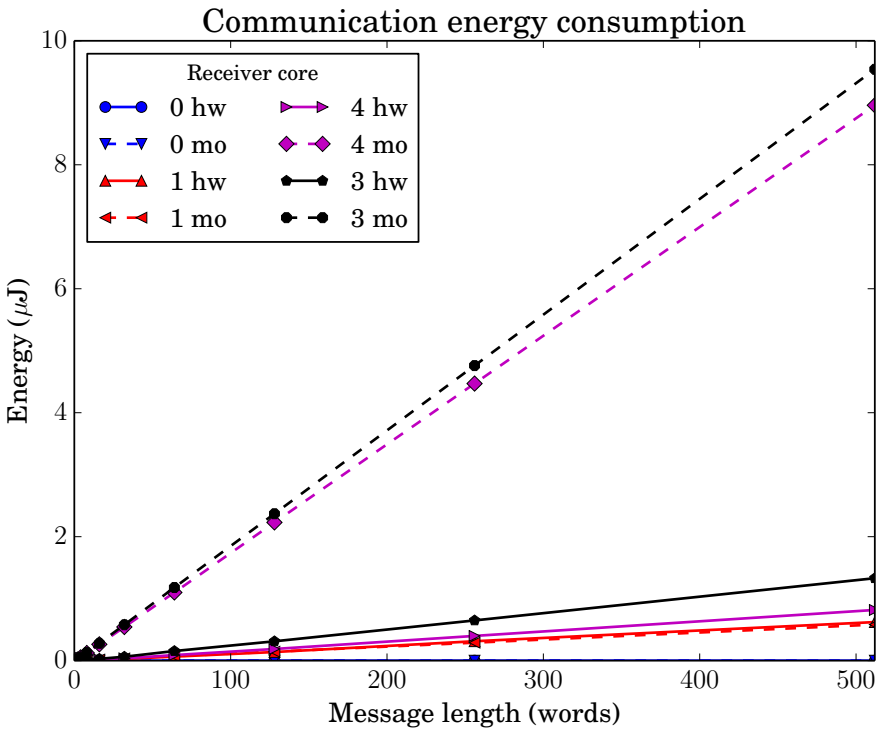


Textual reporting also possible, including static/dynamic power, communication, and breakdown by core.

In this presentation

- Xcore architecture refresher.
- A brief overview of our multi-threaded energy model.
- Requirements for a multi-core model.
- Experimentation with the Swallow many-core platform.
- System level energy consumption view.
- **Challenges, current work, next steps**.

Challenges



- Swallow is a very large system.
- Network modelling accuracy varies.
- Creates a gap between system utilisation and analysable software.

Lesson: Don't jump from 1 core to lots of cores.

Fortunately, we have smaller systems that we can work with more easily.

Current work/next steps, other/future work

Currently:

- Analyse programs running on multiple (2) cores.
- Analyse balanced pipelined communicating programs across multiple cores.

Then:

- Unbalanced pipelined programs.
- Different program structures (client-server, complex compositions, etc).
- Consider voltage-frequency scaling.

Other opportunities:

- Static optimisation of task placement based on communication costs available from model.
- Consider other peripheral devices in the graph-based energy model, not just compute nodes.
- System level costs, such as power supplies.

References

[1] S. Kerrison and K. Eder, "Energy Modeling of Software for a Hardware Multithreaded Embedded Microprocessor," *ACM Transactions on Embedded Computing Systems*, vol. 14, pp. 56:1–56:25, Apr. 2015.

[2] U. Liqat, S. Kerrison, A. Serrano, K. Georgiou, P. Lopez-Garcia, N. Grech, M. Hermenegildo, and K. Eder, "Energy Consumption Analysis of Programs based on XMOS ISA-level Models," in *Proceedings of the 23rd International Symposium on Logic-Based Program Synthesis and Transformation (LOPSTR'13)*, 2014.

[3] N. Grech, K. Georgiou, J. Pallister, S. Kerrison, J. Morse, and K. Eder, "Static analysis of energy consumption for llvm ir programs," in *18th International Workshop on Software and Compilers for Embedded Systems (SCOPES'15)*, 2015.

To appear, accepted for publication.

[4] S. J. Hollis and S. Kerrison, "Overview of swallow—a scalable 480-core system for investigating the performance and energy efficiency of many-core applications and operating systems," *arXiv preprint arXiv:1504.06357*, 2015.

Thank you

Questions?

Pipeline schedule

Time-step	1 thread	2 threads	3 threads	4 threads	5 threads
1	$T_{0,0}$	$T_{0,0}$	$T_{0,0}$	$T_{0,0}$	$T_{0,0}$
2	—	—	$T_{1,0}$	$T_{1,0}$	$T_{1,0}$
3	—	$T_{1,0}$	$T_{2,0}$	$T_{2,0}$	$T_{2,0}$
4	—	—	—	$T_{3,0}$	$T_{3,0}$
5	$T_{0,1}$	$T_{0,1}$	$T_{0,1}$	$T_{0,1}$	$T_{4,0}$
6	—	—	$T_{1,1}$	$T_{1,1}$	$T_{0,1}$

Table: Representation of instruction sequence for various *active* thread counts, with threads represented as $T_{n,i}$, for thread number n and instruction number i.