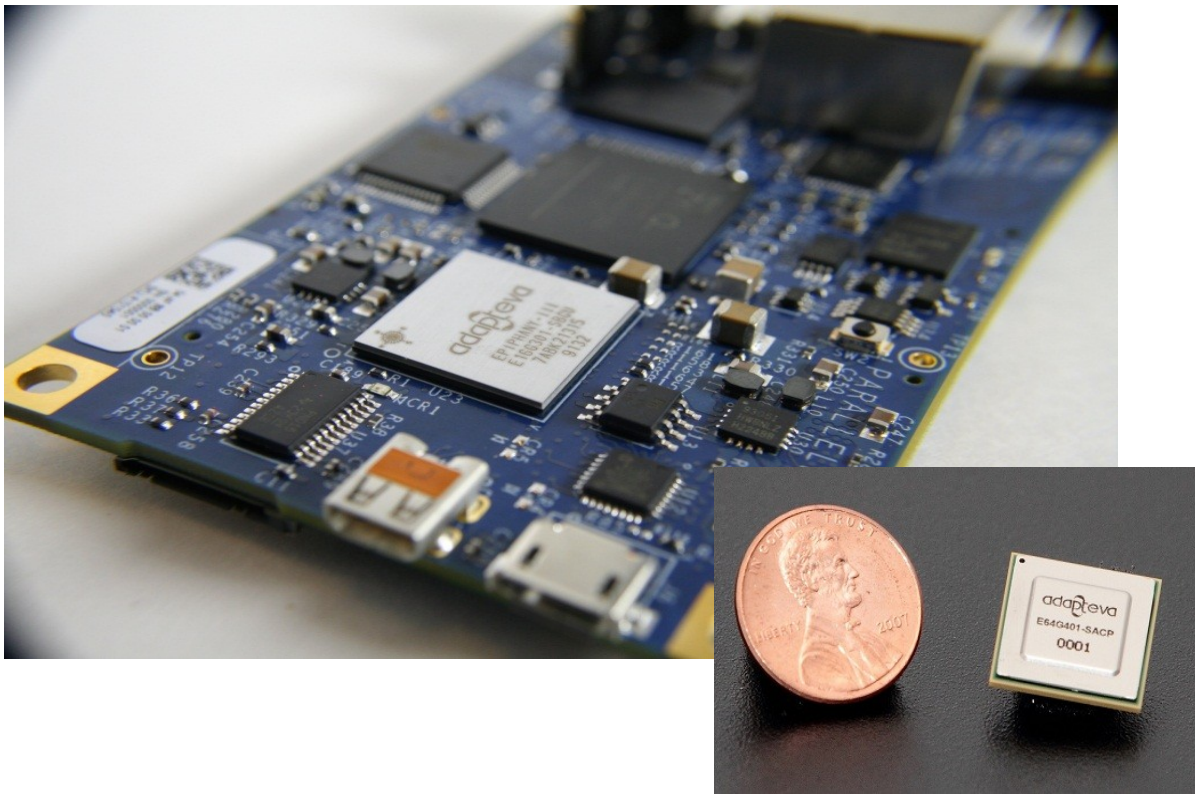
Things I learned while designing the Epiphany & Parallella

Presentation at

Chalmers University of Technology

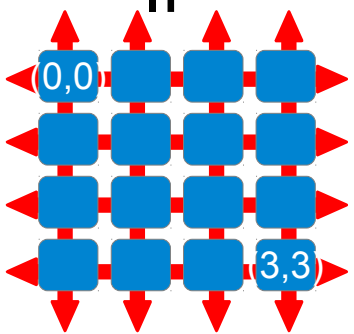
Feb 2, 2015



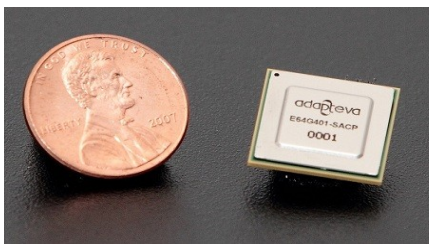
Adapteva Overview:

- Semiconductor company founded in 2008
- 64-core 28nm processor achieves 50 GFLOPS/W
- $900K “Parallella” Kickstarter funding 2012
- $3.6M Series-B in 2013 (Ericsson)
- 10,000 customers and $2M revenue in 2014

IP

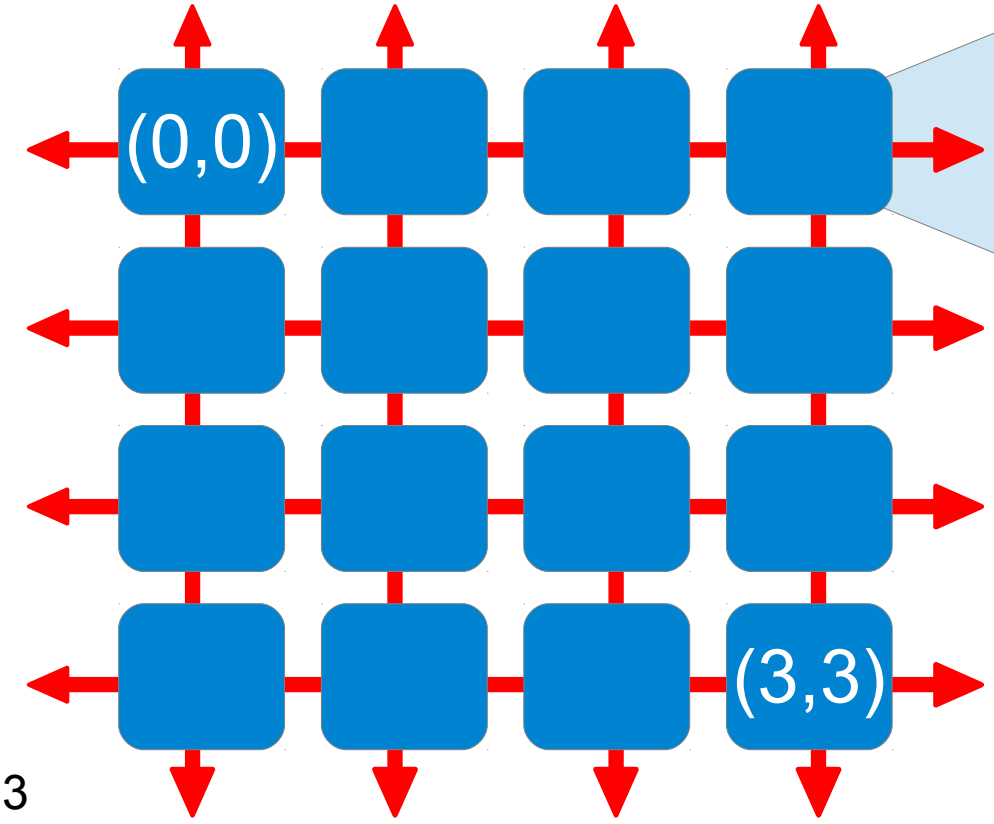


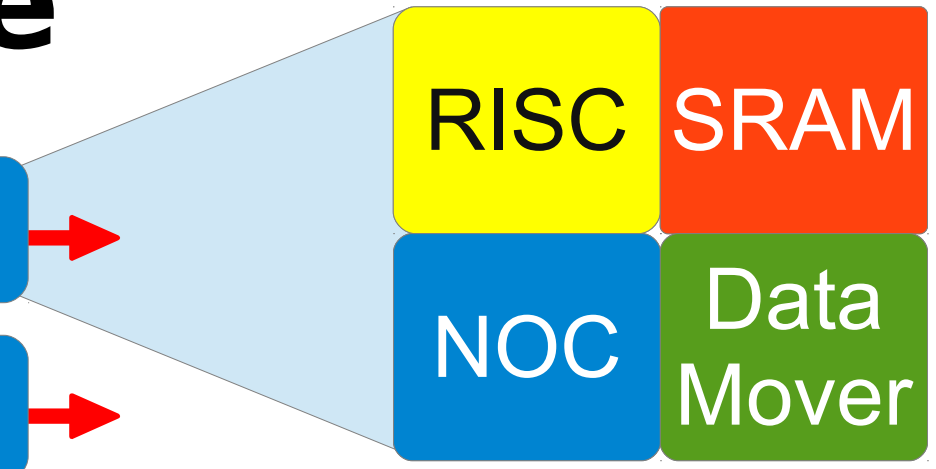
CHIPS



BOARDS

The Epiphany Multicore Architecture





- ANSI-C, OpenCL, MPI, OpenMP
- Scalabale to “infinity” (4096 cores/chip at 28nm)
- Distributed memory model
- 50 GFLOPS/W (25mW & 0.13mm^2 per core)

About Me: Before Adapteva

+

8 years designing DSPs

100 people, $100M R&D

TigerSHARC was a tech success, but a huge financial flop!

Mixed signal SOCs for CCDs

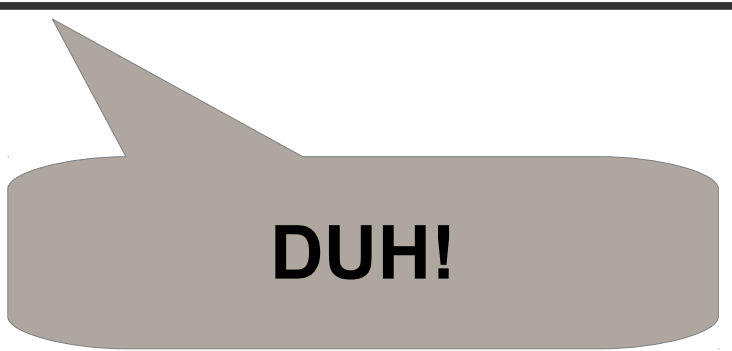
Invented new ISA in 2 weeks

3-4 designers per derivative

Huge success (>$100M)

“My Epiphany” (2007)

For DSP it's much more efficient to design a pipeline of specialized processors than having a single massive processor that runs a complete application.



Goals for Epiphany (2008)

- Floating point (industrial + medical markets)
- ANSI-C programmable
- Scalabale
- 10X boost in GFLOPS/Watt (otherwise, who would buy?)
- Realizable by a team < 5 engineers

Old Architecture Process (Waterfall model)

1) Application/system engineer creates a strawman
2) Performance tested in simulation
3) Microarchitecture development
4) RTL development
5) Synthesis and place and route
6) Build chips, wait for feedback, go back to “1” (12-24 months)

Still the norm today to have different people for each task and not to feedback information to previous step. (successful SOC companies are smarter than this)

My Architecture Process

- One person owns the whole path from arch to layout
- Spreadsheet (triangulating architectures)
- Emacs (does code look clean on paper?)
- Manually “count” gates in code. “Feel complexity”
- Verilog Simulation (checks for logical/mental mistakes)
- Synthesize design (in FPGA or EDA to **measure cost!!**)
- Use advisors (peer review to look for gotchas)
- Application development (**No** but I should have!!)
  (*considering the lack of benchmarks, it's a small miracle Epiphany works as well as it does)

Step 1: Study History

My inspiration of what to do (and NOT to do)

- Henessy and Patterson's classic books. (Re)Read them!
- Blackfin, TigerSHARC, and c64x (DSP features)
- ARM and MIPS (standard features)
- Tilera and Ambric (manycore features)

My rule: Watch out for “clever” features! The really useful features will be common to all archs in a class.

Step 2: Create a baseline

- How much area goes to computation?
- How expensive is a floating point unit?
- How expensive is SRAM and cache?
- How expensive is the Network?
- What is the performance gain of instruction X?
- How often is instruction X used?
- What does a typical program look like

  These numbers are your anchor, get them right!

Step 3: Verify assumptions

- Design and simulate (to verify cost assumptions)
- Build prototypes (to verify simulation assumptions)
- Talk to customers (to verify application and product feature assumptions)

Summary of Choices

Dilemma	Choice	Reason	Outcome
Programming Lang	C	No brainer	Great (common)
Memory size	32KB	Tapeout cost	Poor (needed 64KB)
Load-stores	64-bit single LD/ST	Compromise	Great
# Memory banks	4	Compromise	Great
# Registers	64	Compromise	Great
2nd Integer Instr?	Yes	Hedge	Great (see bcrypt..)
NOC flits	Yes	Efficiency	Great
NOC buffers	Yes	Efficiency	Great
Byte addressing	Yes	General purpose	Great
Debug, interrupts	Yes	General purpose	Great
Hardware caching	No	Power	Great

Summary of Choices

Dilemma	Choice	Reason	Outcome
No special instr	No	Efficiency	50 GFLOPS/W
64bit floating point	No	overkill	Didn't matter
Branch target buffer	No	power	Great
Push/pop	No	Limit function calls	Good enough
Dual issue	Yes	2x perf (!2x power)	Great
Stall scoreboard	Yes	Ease of use	Great
LVDS	Yes	Performance	Great
PLL on chip	No	No $$	Great
SERDES	No	No $$	Great
Integer multiply	No	Power	Poor (very common)

Summary of Choices

Dilemma	Choice	Reason	Outcome
Signed byte load	No	power	Not good (limits apps)
Streaming IO protocol	Autoburst	Simplicity/latency	OK (random only at 25% peak)
Pipeline depth	5+3	Compromise	Great
Autoincrement LDR	Yes	Efficiency	OK (no compiler support)
Hardware loops	Yes	Benchmarks	OK (no compiler support)

Some Hardware Lessons

- Architects should be vertically integrated
- Tiled architectures are MAGICAL
- Symmetry is a very powerful design principle with sometimes unexpected rewards
- Designing leading edge chips is fairly easy today for a small team with the right experience (<<$100M)
- Designing boards is easier than designing chips but not trivial
- Never push the tools, process, or capabilities of your vendor (they will break).

Some Software Lessons

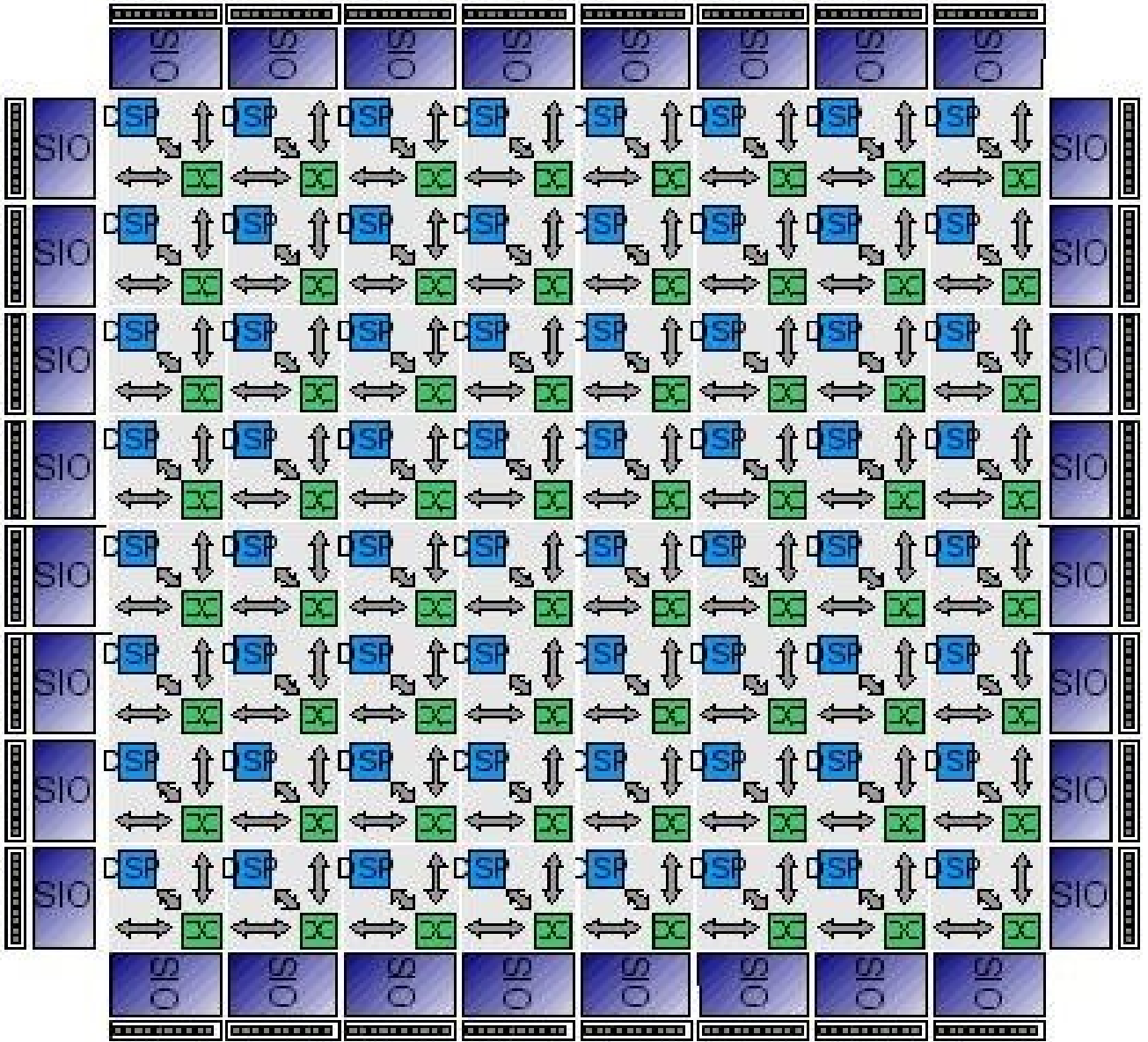
- Modern compilers are incredible (given enough registers and a small instruction set)
- Customers take the path of least resistance
- Developers take the path of least resistance
- Surprisingly hard to find applications that need more performance....

  ...and those applications are really complex..

  ..meaning they have tons of legacy stack software...
- SW trumps energy efficiency in 99.99% of applications

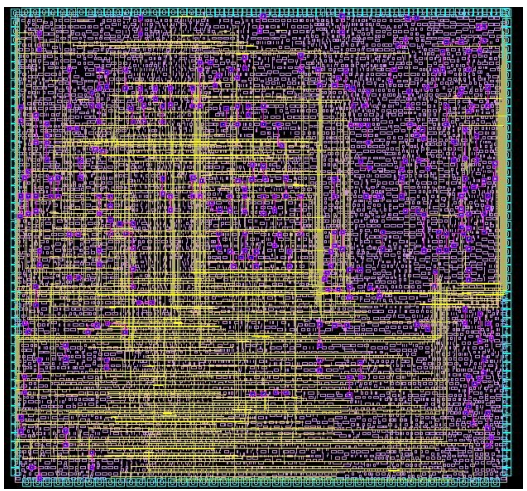
Epiphany (-1)

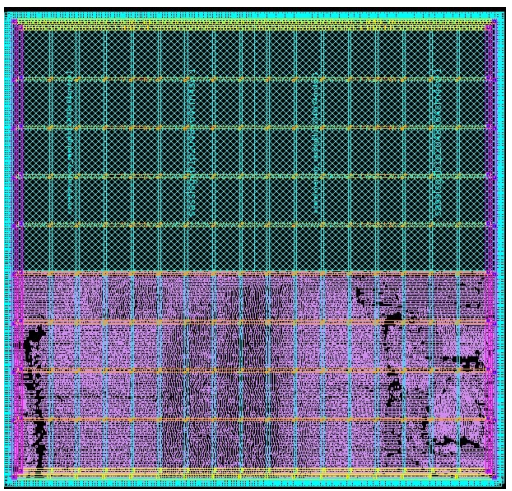
- May 2008
- Naive but close...
- Could have had 64 cores in 2008!
- Epiphany arch has only changed by “delta” since 2008
- Submitted many grant proposals (all rejected)

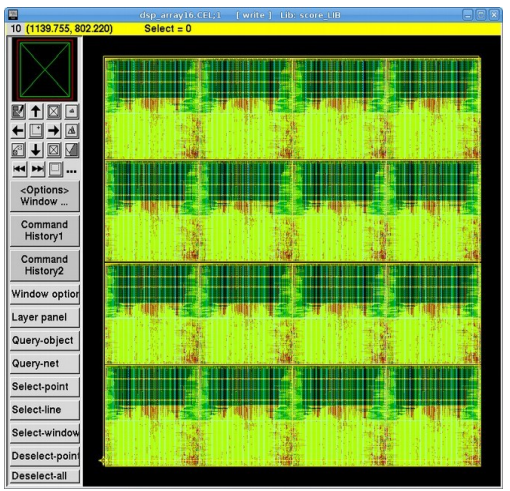


Epiphany-0

- Oct 2008 synthesis and layout prototype
- Virtual prototypes (synthesis and layout of cores)
- Showed that 1 Ghz operation and 50GFLOPS/W was possible
- Gave me the courage to ask F&F for money ($200K seed)

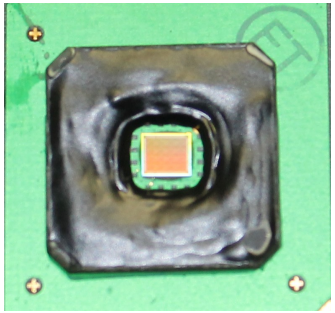


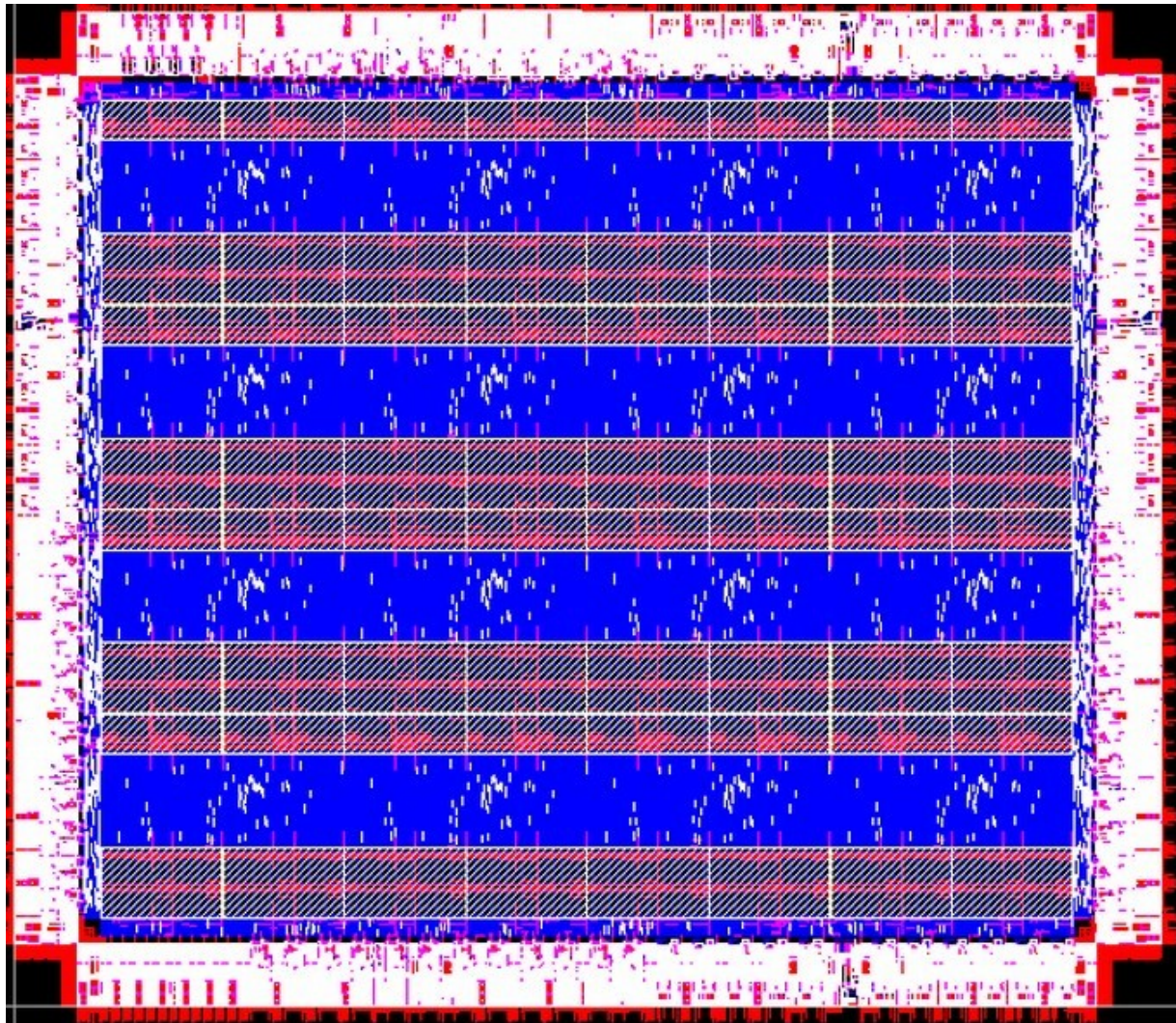




Epiphany-I

- Jun 2009 tapeout
- 16 cores, 65nm prototype
- ~1 person, 12 weeks
- Changes same day as TO!
- Barely worked but proved it was possible
- ~12mm^2





Building a Team (Dec, 2009)

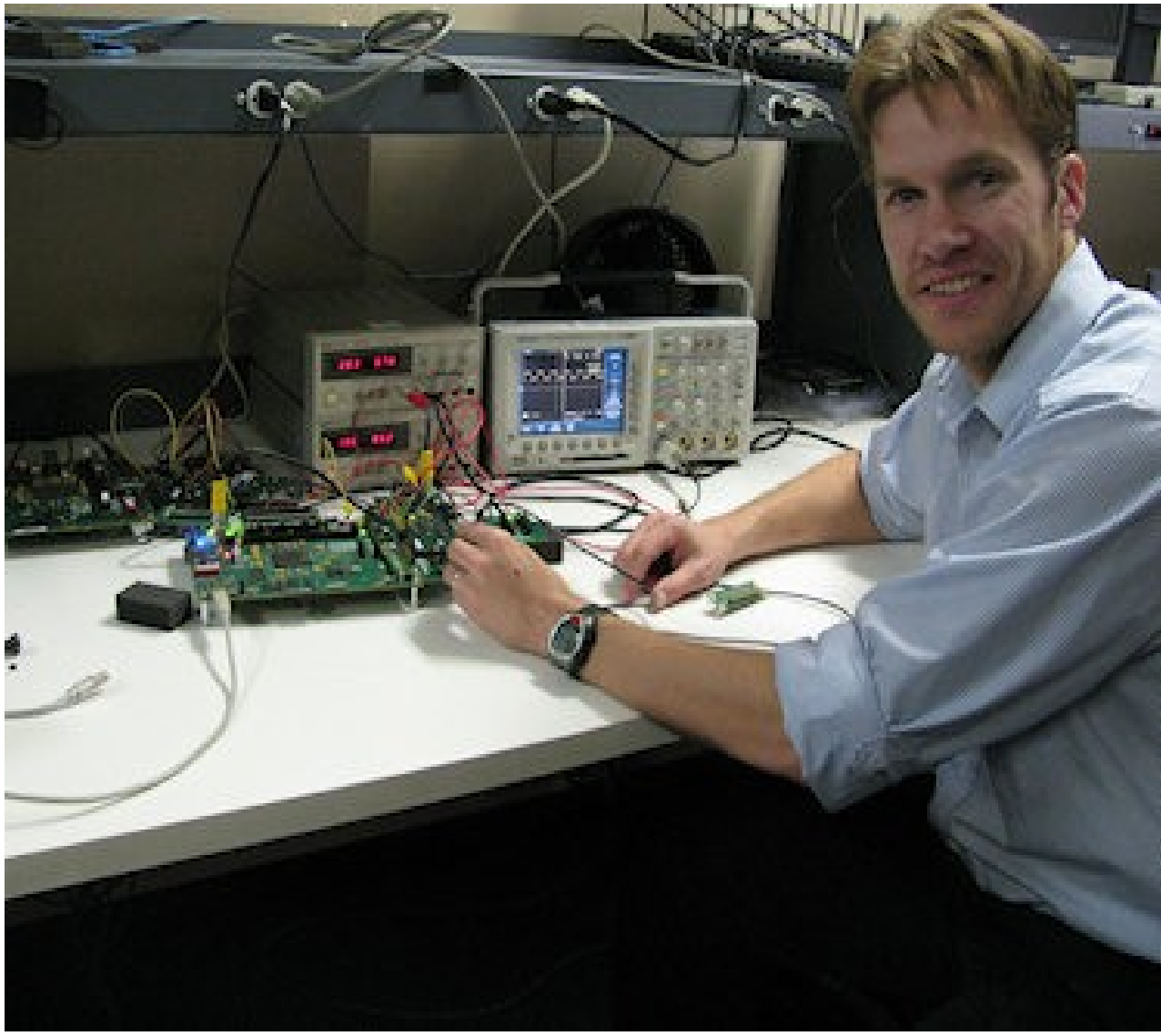
Lesson: Team makeup is the #1 determinant for success. You must have it in place before you set out on journey.

Epiphany-II

- May 2010 tapeout
- 16 cores, 65nm
- ~3 person, ~6 weeks
- Changes 1 day before TO
- A vendor screwed up (lucky for us...)
- Should have been a product
- ~12mm^2

It Works!

- Sept 2010 bringup
- Ran floating point program!
- Provied efficiency!
- Packaging yield was less than 10% on eLink (vendor).
- SPI backdoor was key to debugging vendor packaging issue

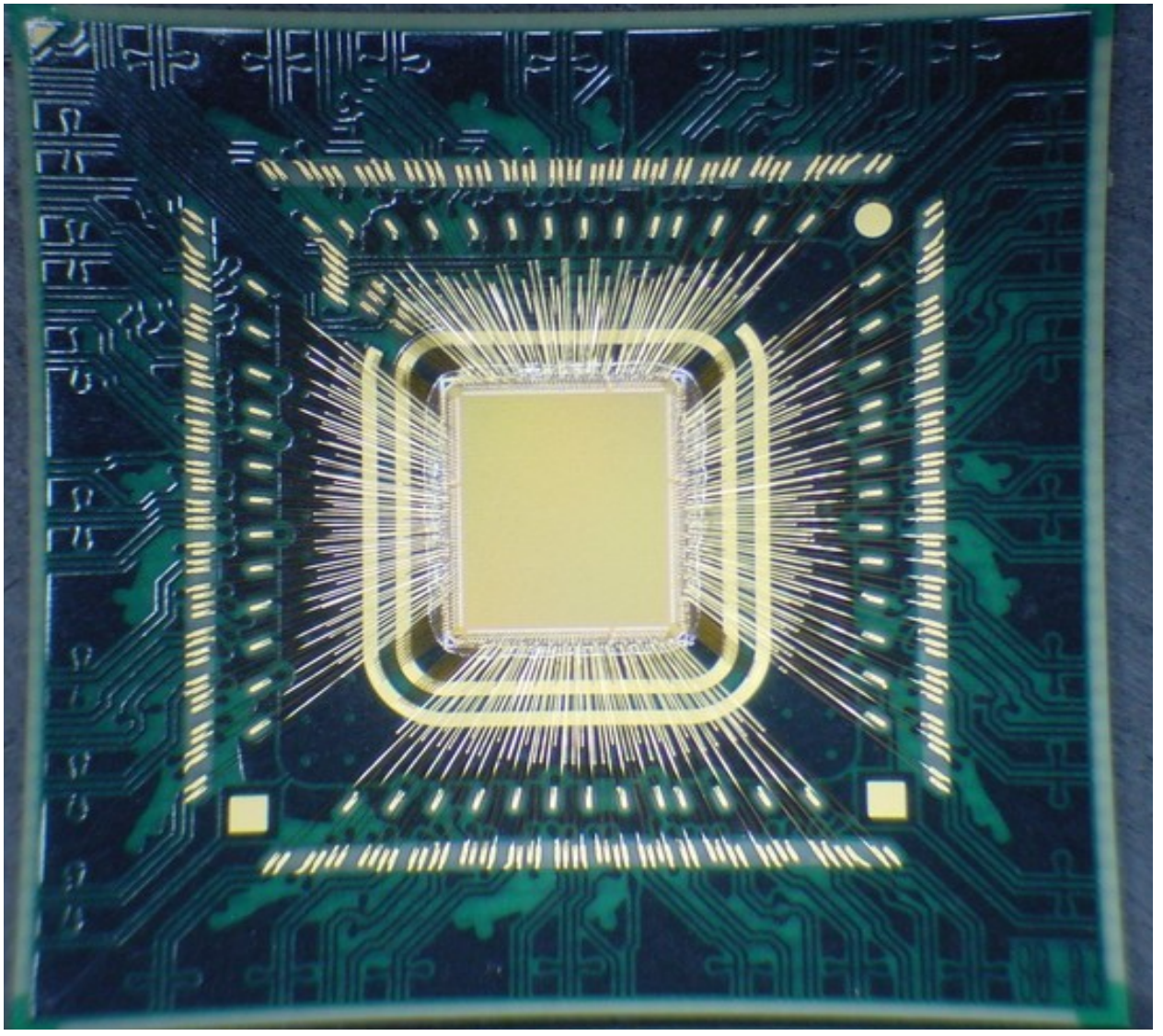


Epiphany-III

- Dec 2010 tapeout
- 16 cores, 65nm
- This is a product!
- Kept improving with every iteration
- RTL changes 1 day before TO
- ~25 GFLOPS/W
- ~12mm^2

New Packaging

- 60um staggered pitch is not trivial!
- Vendor failure on E-1 and E-2 almost broke company!
- New vendor (Kyocera)
- You can't be on the bleeding edge without the right partners!
- Better results!



First Delivery

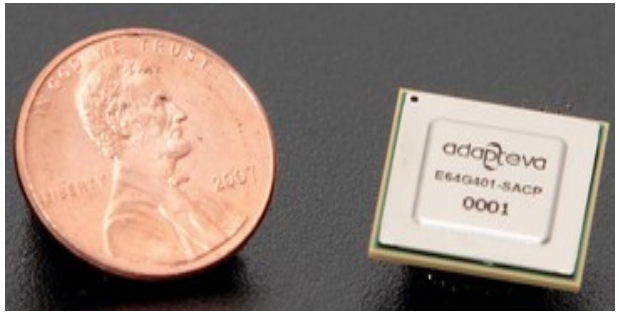
- May 2011
- Chip worked perfectly out of the box!
- This silicon became the E16G301
- Felt at peace
- Happy moment, but only the beginning...

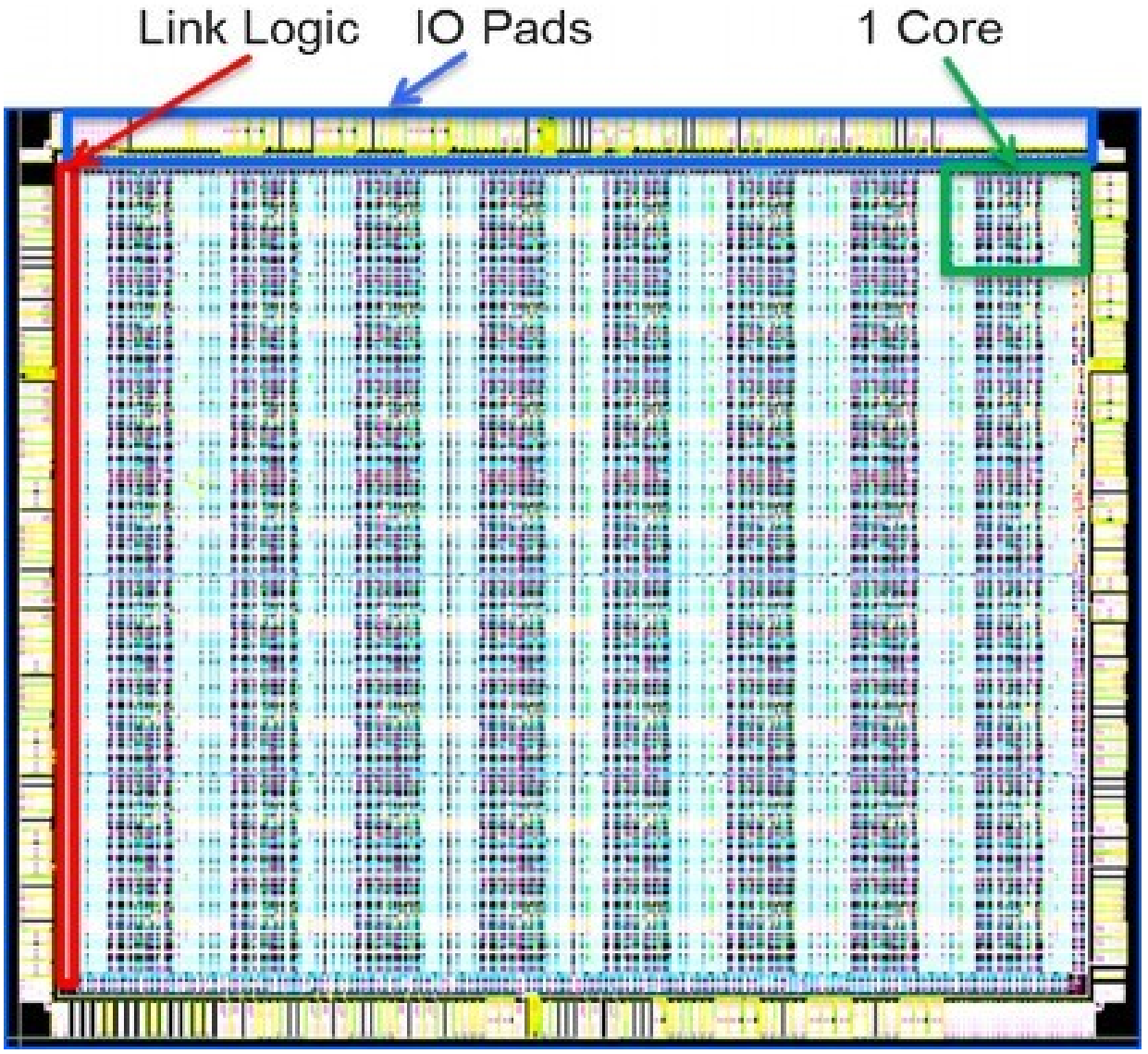
ESC-SV 2011

- May, 2011
- Launched Epiphany-III
- Lots of press!
- Fun...
- But where were the customers!!???

Epiphany-IV

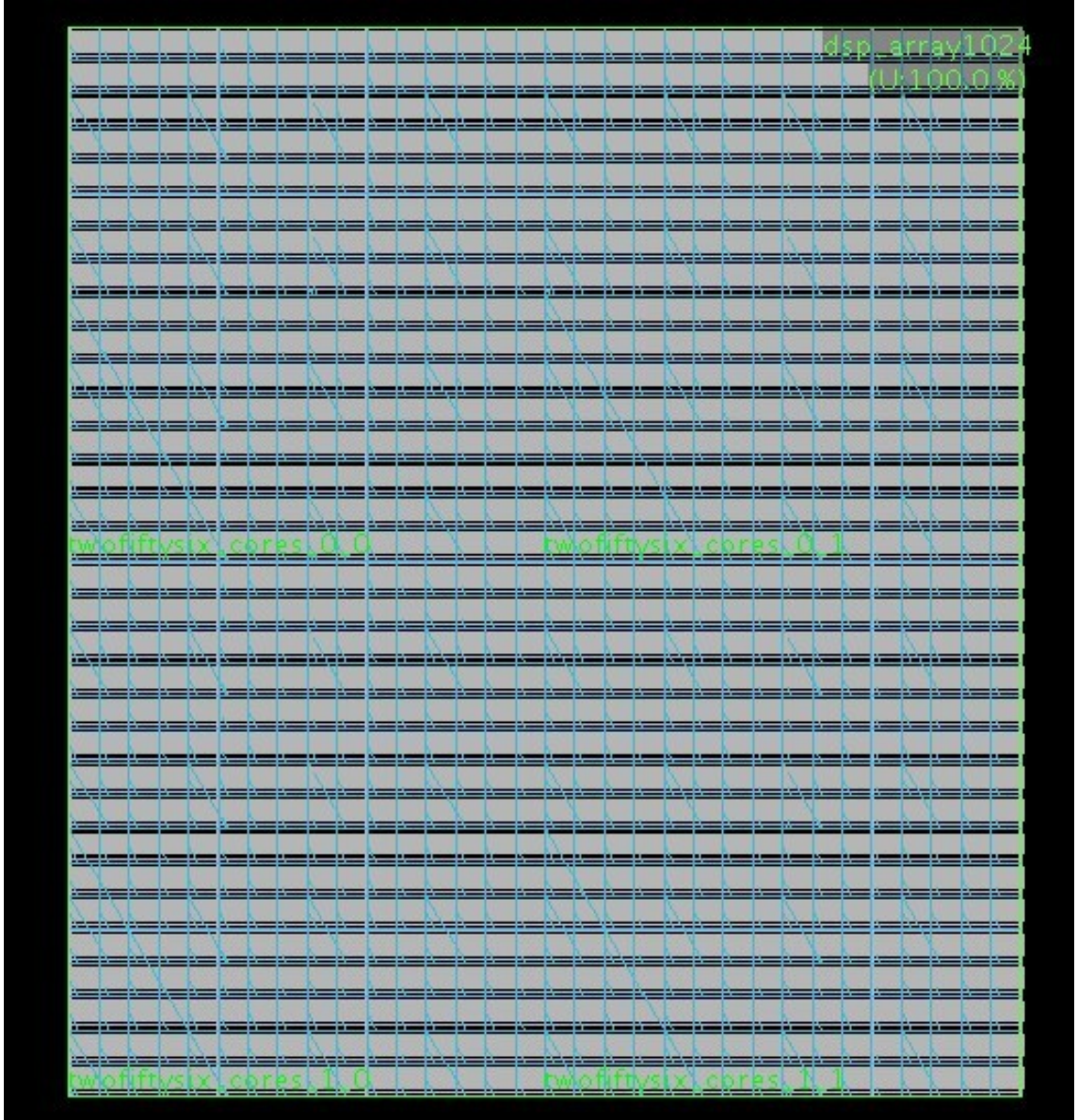
- Aug 2011 tapeout
- (Jul 2012 samples)
- 64 cores, 28nm
- 50 GFLOPS/W
- RTL changes 2 days before TO
- Done in 12 weeks!





More HW...

- Feb 2012 IP demo
- IP demo of 1024 cores
- IP demo of DPF
- IP demo of 128KB SRAM
- 1 day to layout each

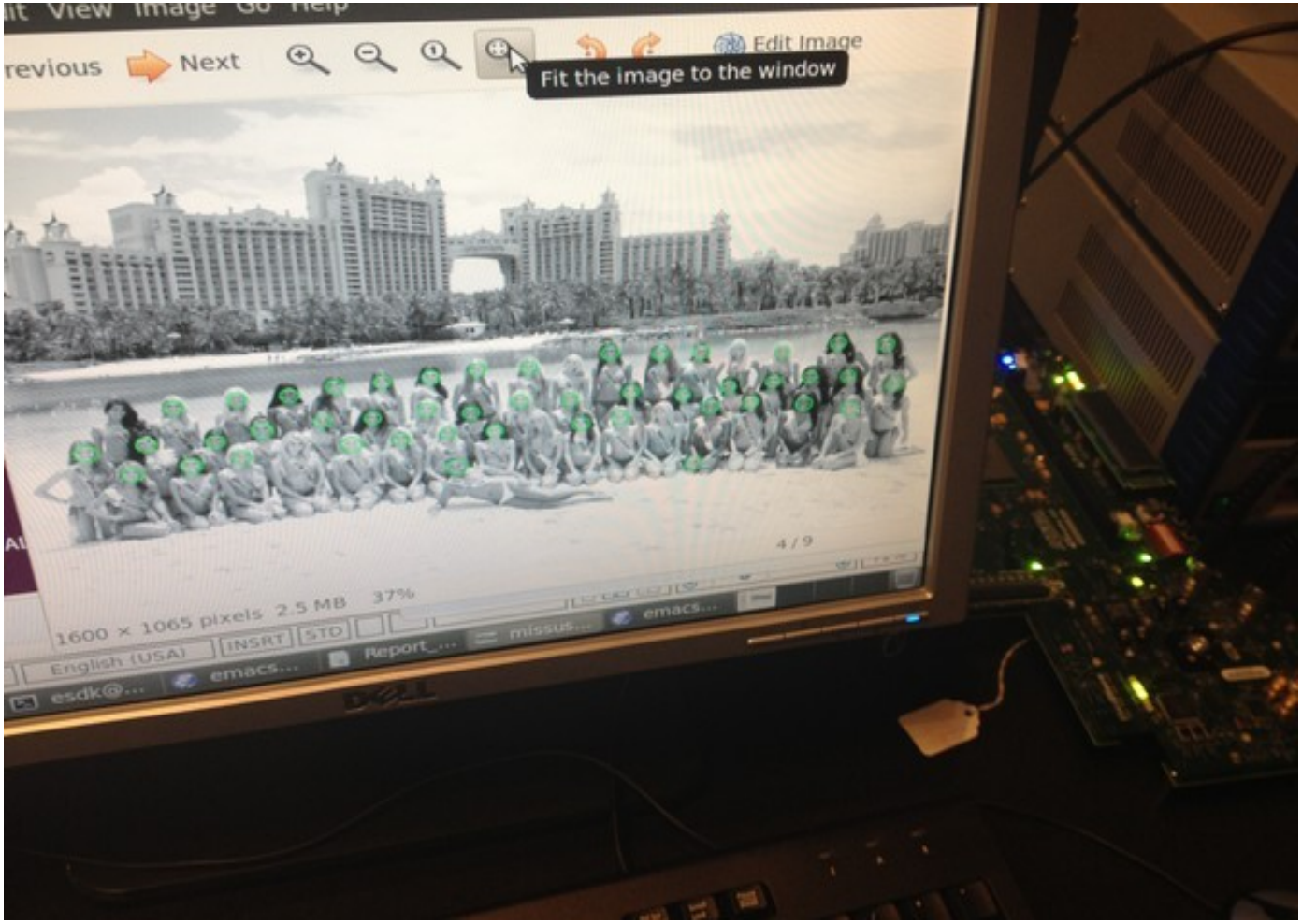


First Product

- May, 2012 ship
- FMC board from Bittware
- 4 chips in array
- Not cheap!
- Never shipped in volume

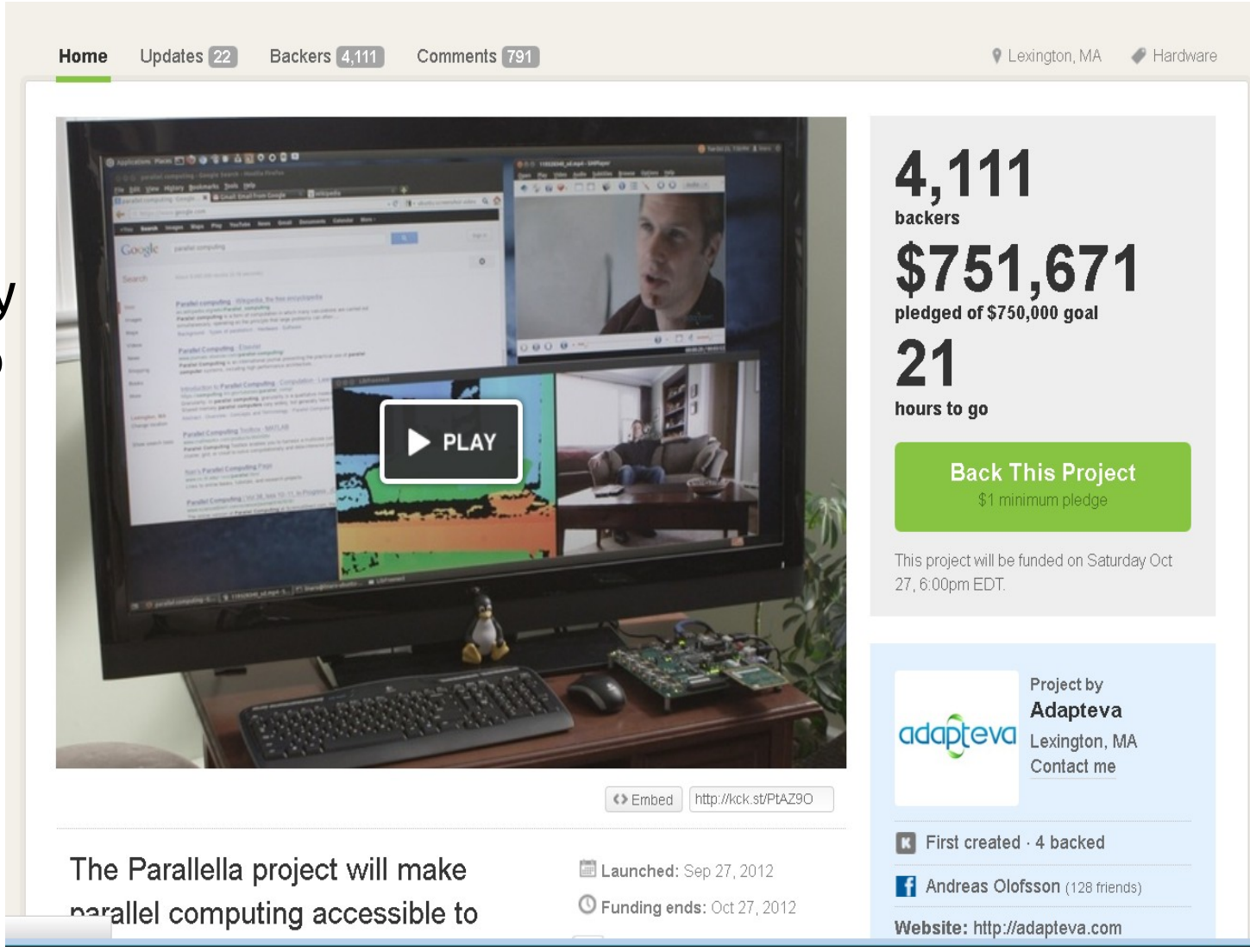
Software

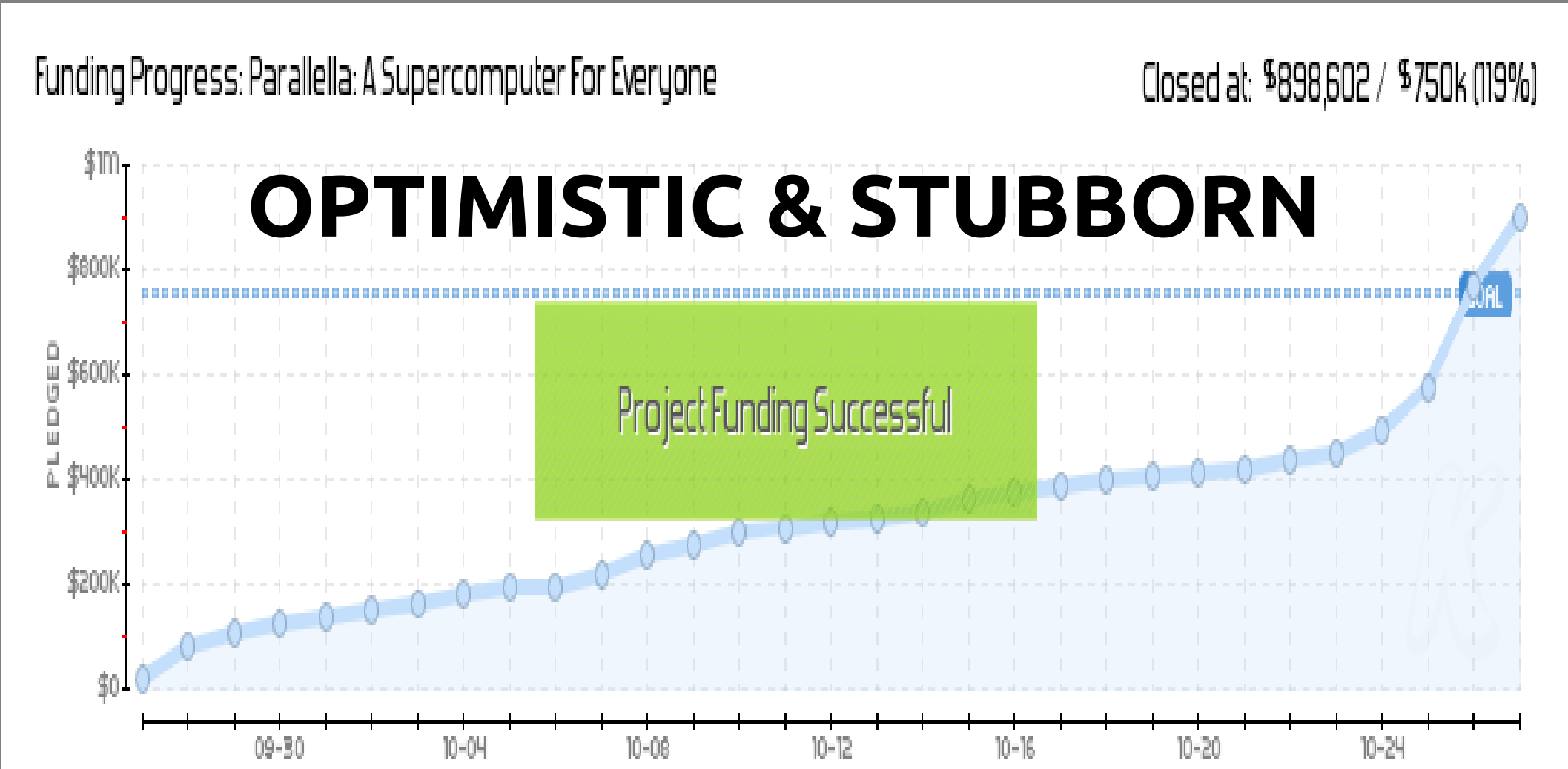
- 2012 focus
- Invested heavily in software tools + demos
- Face recognition, FFT, Matmul
- Beat ARM by 10X on demo for BIG smartphone company!
- Response: “Meh...”



Parallella

- Sept, 2012
- We needed a parallel community and a better kit, so the $99 Parallella was born, Sept 2012
- KS combines pre-purchase, community, and marketing.



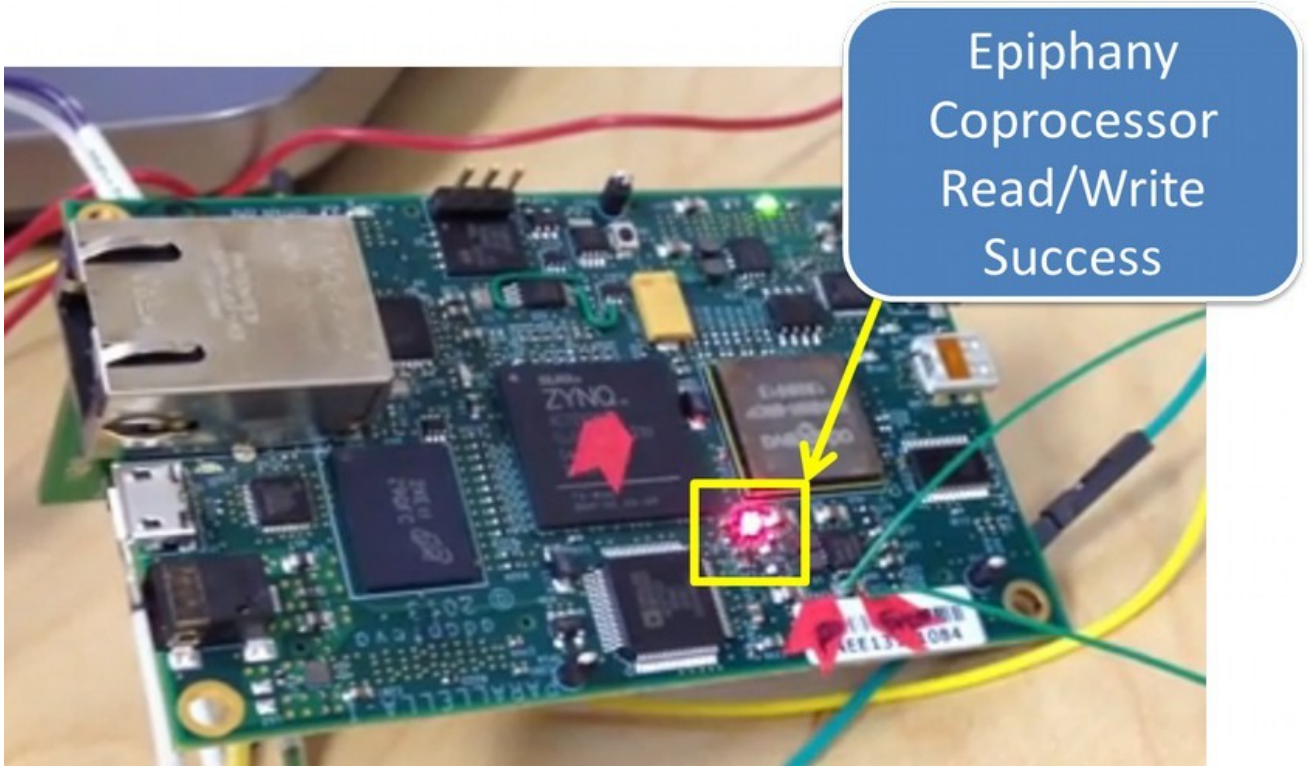


Success?

- Nov 2012
- Woke up next morning stressed and behind schedule.
- Only sell what you have!!

Powerup (Gen0)

- May 2013
- My first board and it worked!
- Only minor issues
- Challenge was super aggressive deisgn targets (credit card, features, cost, schedule)



Gen0 Release

- July 2013
- Gen0 board released!
- We build working cluster with 42 boards!
- Sent out 50 boards to KS backers, only ~1 (NOTZED) got significant use!

Chips are back!

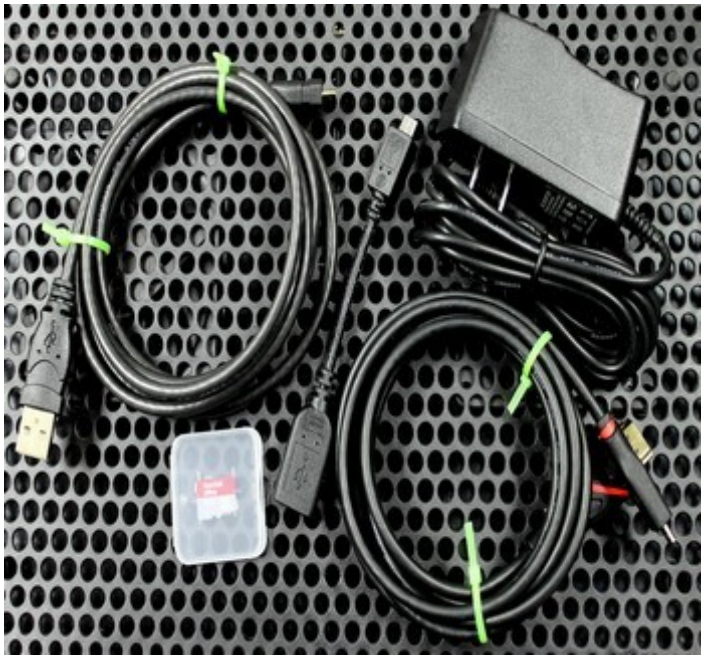
- Aug 2013
- Full mask
- New Tier-1 package vendor (ASE). Awesome company!
- 90% yield!
- Good thermals
- 100/100 grade
- Now we need to test 10,000 chips!?

New Investment

- Dec 2013 closing
- $3.6M from Ericsson + VC
- Published WP showing 25X edge over Intel
- Great...but we were still buried in KS comittments and logistics issues...

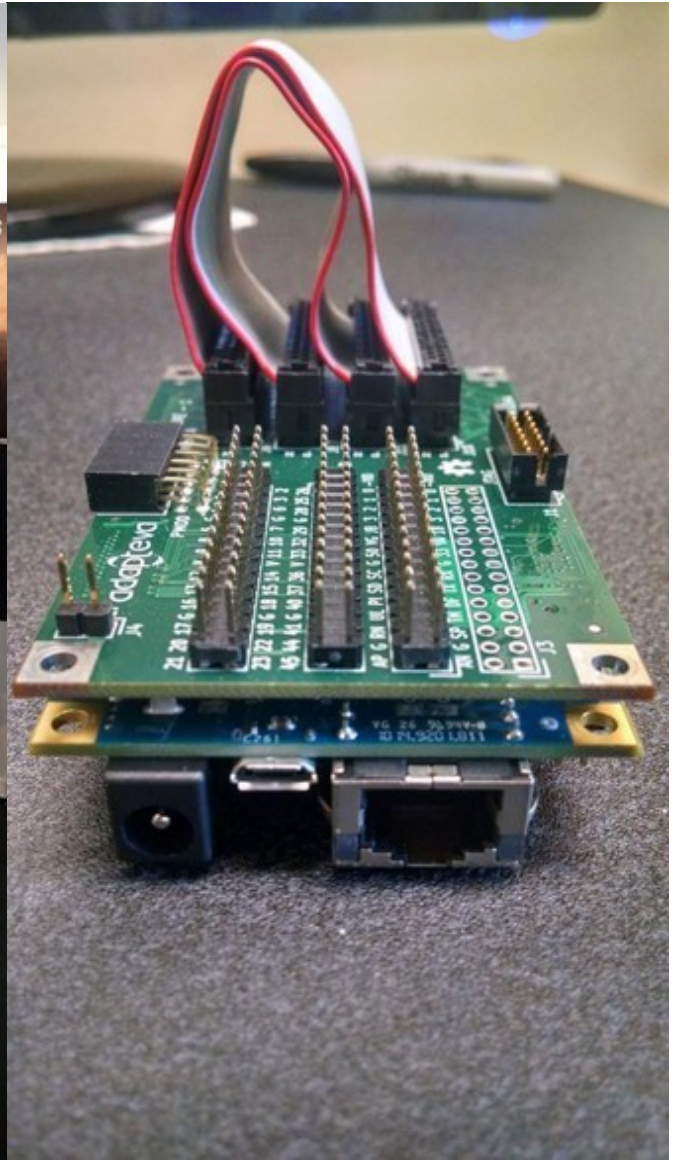
ERICSSON

2014 Distractions: T-shirts, cases, logistics, accessories... (sourcing industrial design is not us!) What is Adapteva again?



More HW...

- June 2014
- Shipped to 200 Universities & 10,000 developers
- Built the “A1”, the world's densest cluster.
- Where are the BIG customers????!!!!!!!



MY FINAL LESSON:
(took me 7 years to learn)

IT'S THE SOFTWARE STUPID!!!!

..but we are still going!!

The Parallel Architectures Library

- Compact C library with optimized routines for vector math, synchronization, and multi-processor communication.
- Designed for parallel and memory constrained hardware
- Designed to be portable across multiple ISAs
- Open source (apache 2.0 permissive license)
- Open invitation to participate!!
- https://github.com/parallella/pal

What I (still) Know:

- Moore's law WILL come to an end
- Parallel computing is inevitable
- Architectures like Epiphany are the future
- CPUs, FPGAs, and manycore will coexist
- The world will continue to be driven by $$