AMD's Vision for the Future: Fusion

Gage Mondok and Aaron Bishop

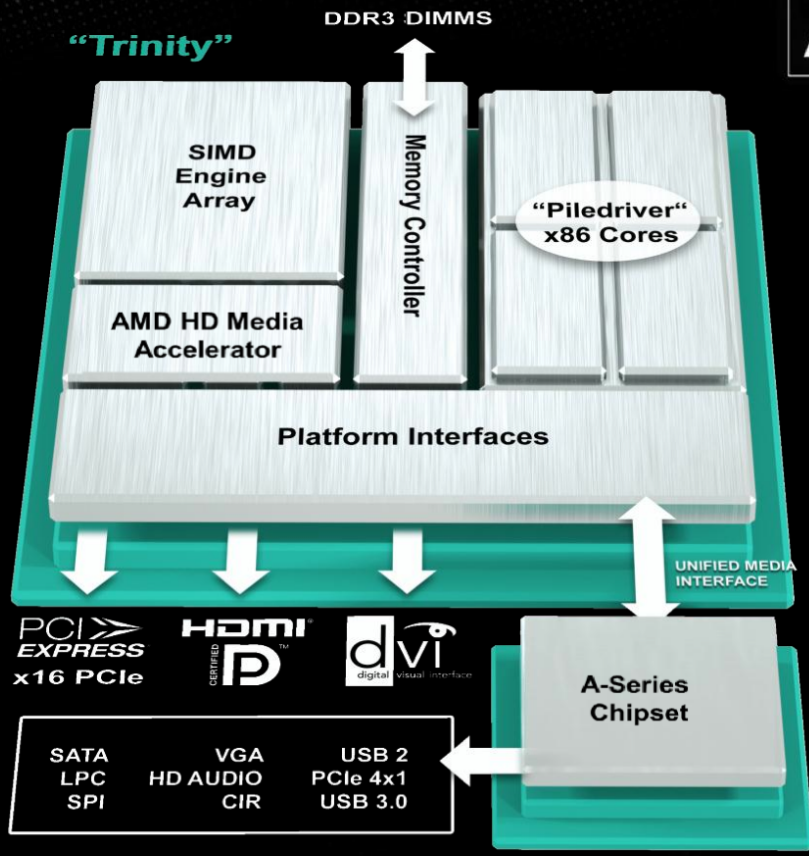
What is Fusion?

-Combination of GPU and CPU into a single package
-Ultimately to be recognized as a single entity
-Also known as Heterogeneous Systems Architecture or HSA
-Processors known as Accelerated Processing Units or APU

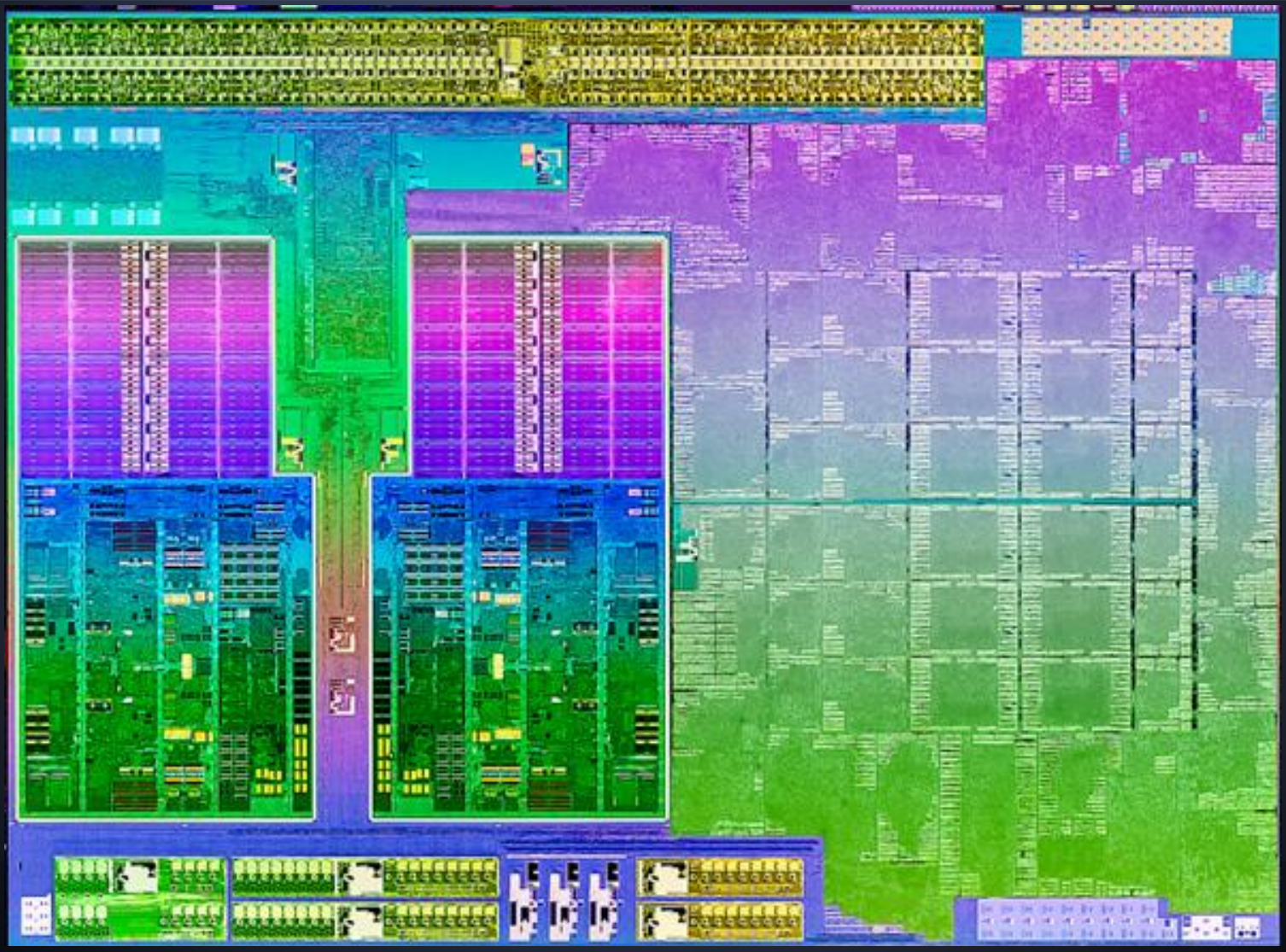
Today's Fusion: Trinity

"TRINITY" APU WITH AMD DISCRETE CLASS GRAPHICS

- **"Piledriver" Cores**
  - 2nd-Gen "Bulldozer" core ("Piledriver")
  - 3rd-Gen Turbo Core technology
- **Multiple Configurations**
  - Memory support up to DDR3-1866 (1600 for notebook)
  - Low power DDR3 (1.25V)
  - Up to quad core and 4MB L2
- **2nd-Gen AMD Radeon™ DirectX® 11**
  - Up to 384 Radeon™ Cores 2.0
- **HD Media Accelerator**
  - Accelerates and improves HD playback
  - Accelerates media conversion
  - Helps Improve streaming media
  - Allows for smooth wireless video
- **Enhanced Display Support**
  - AMD Eyefinity Technology[3]
  - DisplayPort 1.2



Today's Fusion: Trinity

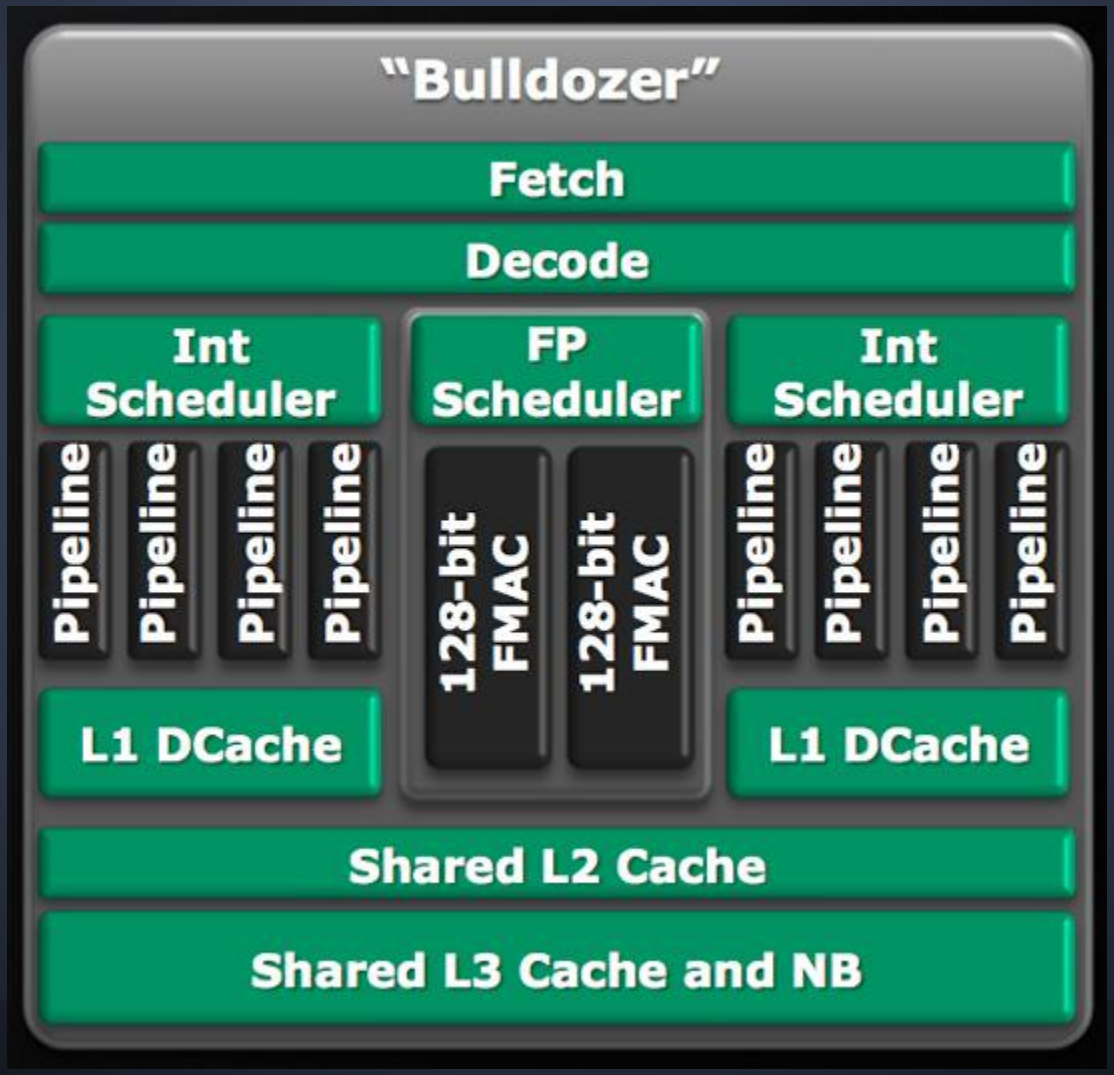


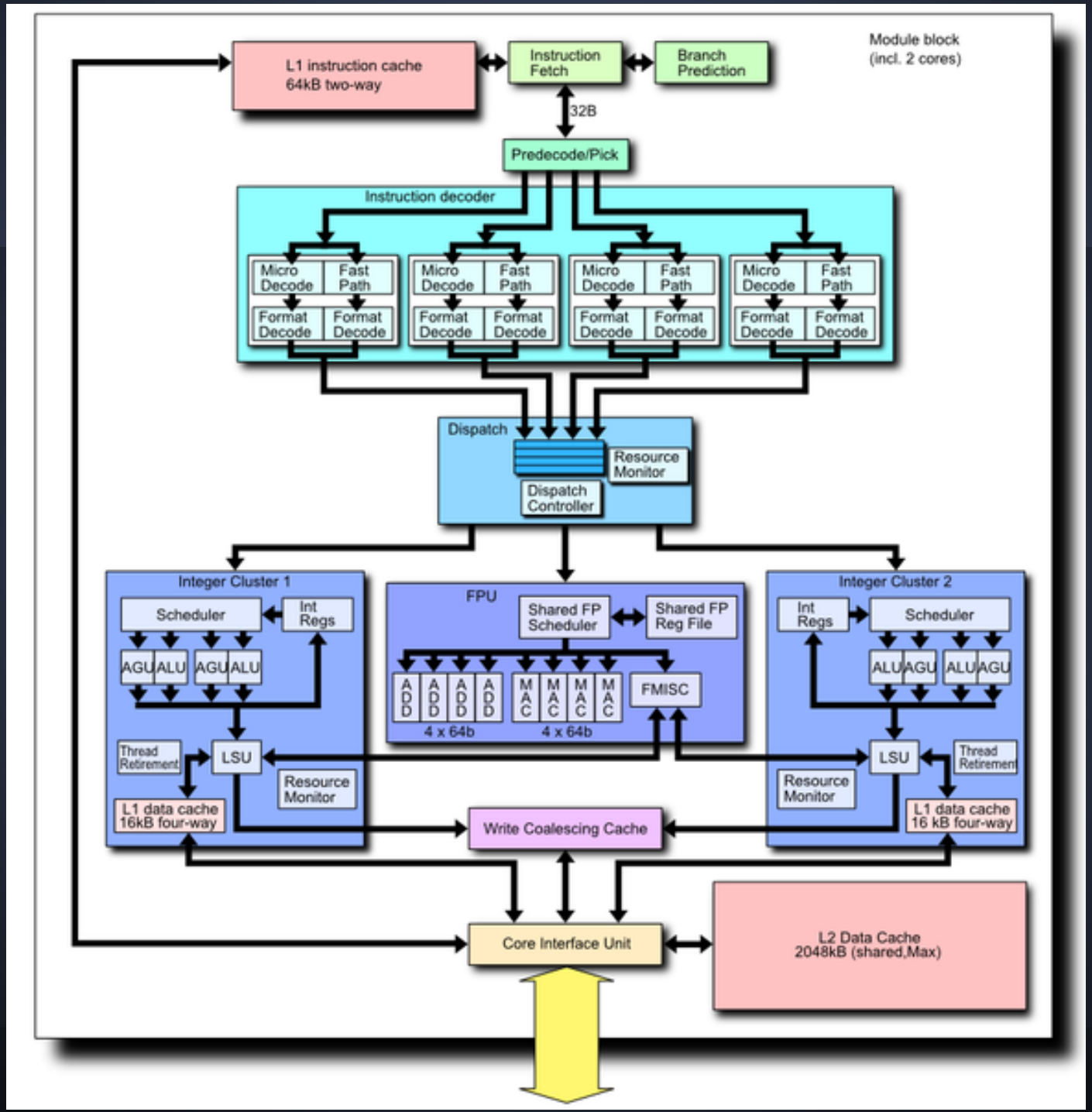
Beginning of Fusion

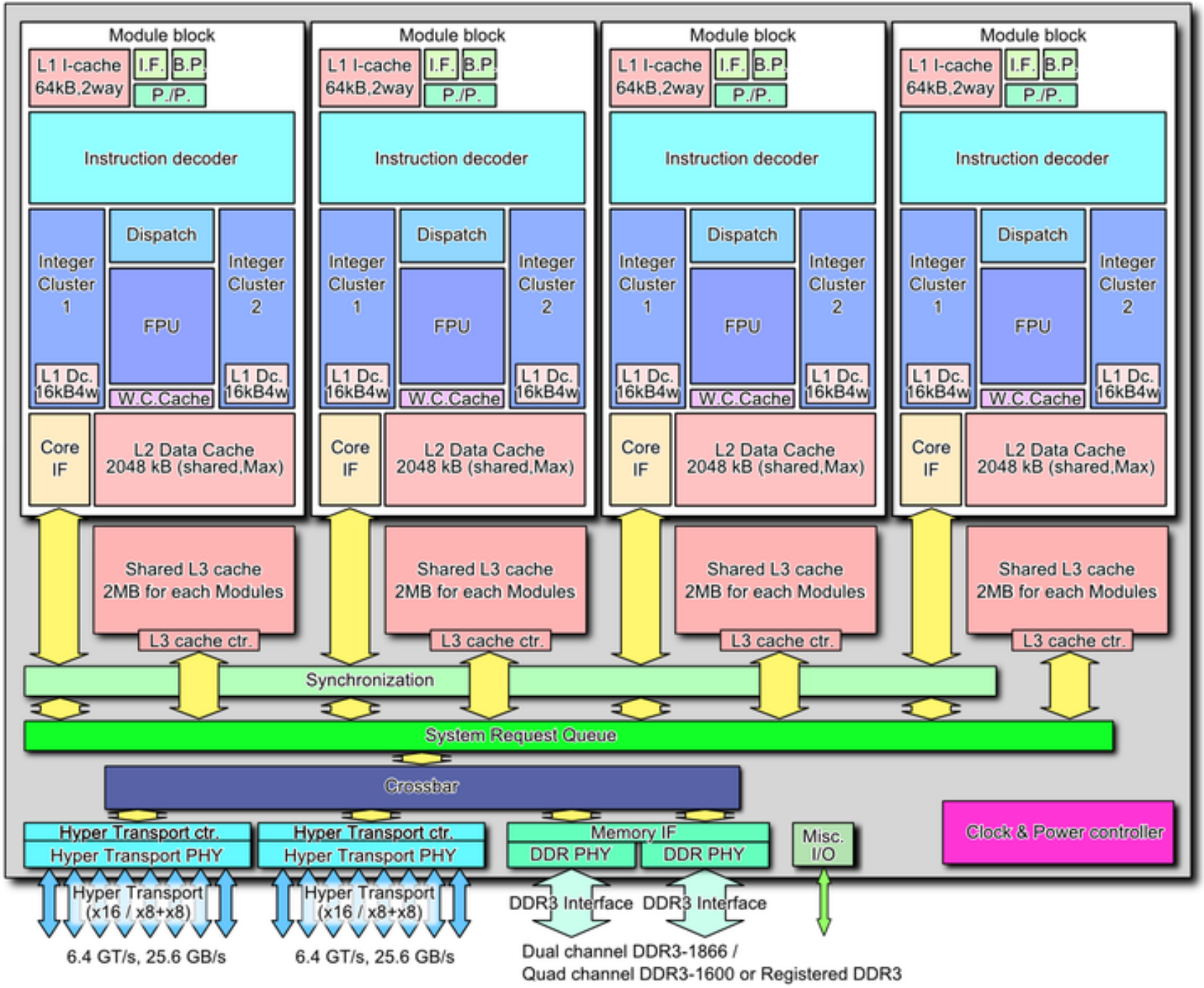
OpenGL®

Microsoft®
DirectX®10

The CPU Half: Bulldozer




Module block
(incl. 2 cores)
L1 instruction cache
64kB two-way
Instruction
Fetch
Branch
Prediction
32B
Predecode/Pick
Instruction decoder
Micro
Decode
Fast
Path
Format
Decode
Dispatch
Resource
Monitor
Dispatch
Controller
Integer Cluster 1
Scheduler
Int
Regs
AGU
ALU
Thread
Retirement
LSU
Resource
Monitor
L1 data cache
16kB four-way
FPU
Shared FP
Scheduler
Shared FP
Reg File
ADD
MAC
FMISC
4 x 64b
Integer Cluster 2
L1 data cache
16 kB four-way
Write Coalescing Cache
Core Interface Unit
L2 Data Cache
2048kB (shared,Max)


Module block
L1 I-cache 64kB,2way
I.F.
B.P.
P./P.
Instruction decoder
Integer Cluster 1
Dispatch
FPU
Integer Cluster 2
L1 Dc. 16kB4w
W.C.Cache
L1 Dc. 16kB4w
Core IF
L2 Data Cache 2048 kB (shared,Max)
Shared L3 cache 2MB for each Modules
L3 cache ctr.
Synchronization
System Request Queue
Crossbar
Hyper Transport ctr.
Hyper Transport PHY
Hyper Transport (x16 / x8+x8)
6.4 GT/s, 25.6 GB/s
Memory IF
DDR PHY
DDR3 Interface
Dual channel DDR3-1866 / Quad channel DDR3-1600 or Registered DDR3
Misc. I/O
Clock & Power controller

The CPU Half

-AMD analytics show that server workloads are, on average, 80% integer operations
-Bulldozer focuses on integer operations
-Only one FP scheduler vs two Int schedulers with many more Int pipelines
-Offload FP operations to GPU
-Reduced die size
-Less power consumption
-Greater performance

The CPU Half: Weaknesses

-Relies on a greater number of smaller cores for efficiency
-However, this only works as desired if programs are multi-threaded
-This leads to low single threaded performance
-Unfortunately, many programs are yet to take advantage of more than 1-2 threads
-Scheduling is poor since the architecture is vastly different
-Technology disadvantage compared to Intel: 32nm vs 22nm

The CPU Half: Weaknesses

Branch Prediction	
Architecture	**Branch Misprediction Penalty**
AMD K10 (Barcelona, Magny-Cours)	12 cycles
AMD Bulldozer	20 cycles
Pentium 4 (NetBurst)	20 cycles
Core 2 (Conroe, Penryn)	15 cycles
Nehalem	17 cycles
Sandy Bridge	14-17 cycles

The CPU Half: Weaknesses

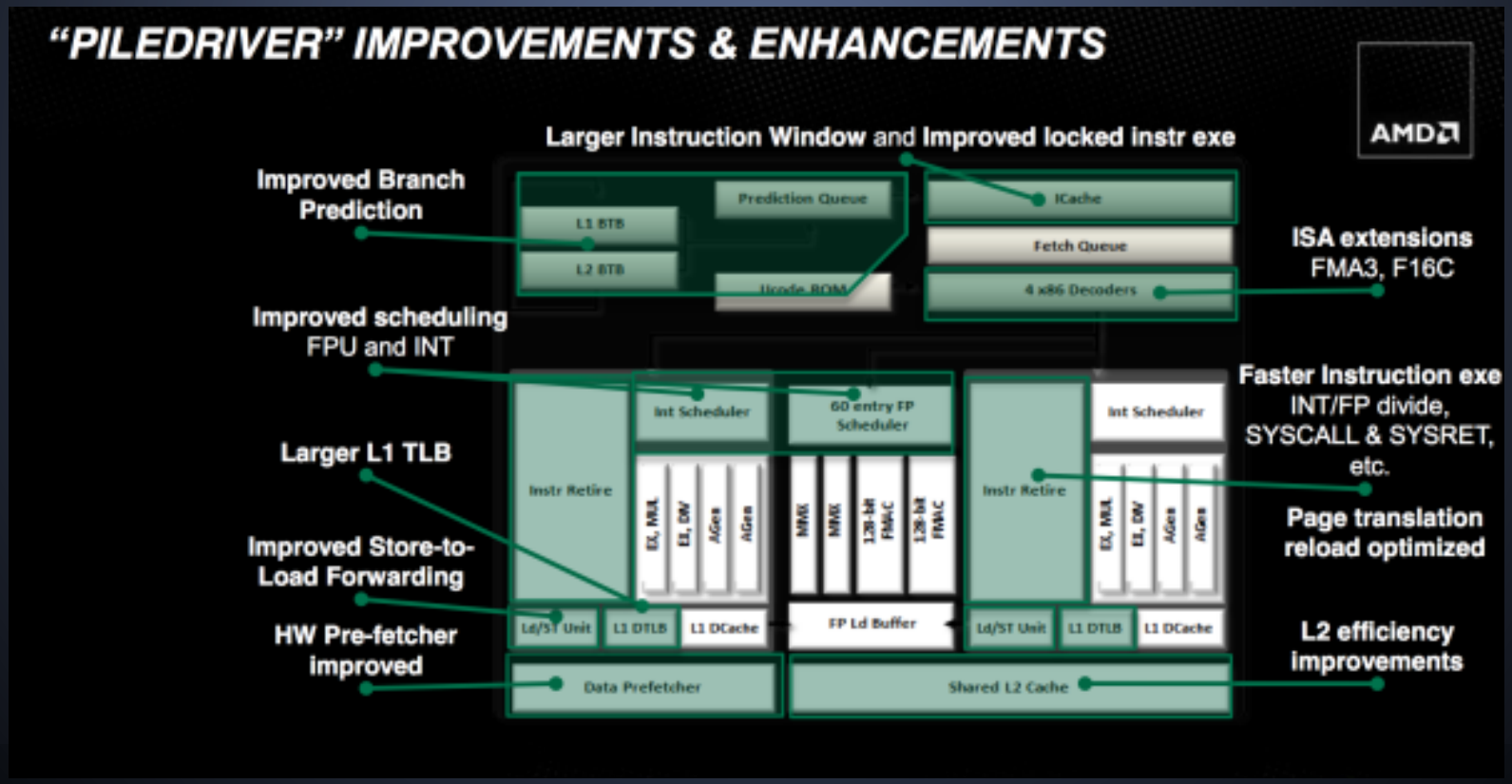
Front End Comparison			
	AMD Phenom II	AMD FX	Intel Core i7
Instruction Decode Width	3-wide	4-wide	4-wide
Single Core Peak Decode Rate	3 instructions	4 instructions	4 instructions
Dual Core Peak Decode Rate	6 instructions	4 instructions	8 instructions
Quad Core Peak Decode Rate	12 instructions	8 instructions	16 instructions
Six/Eight Core Peak Decode Rate	18 instructions (6C)	16 instructions	24 instructions (6C)

The CPU Half: Strengths

-The server world is highly parallel, leading to full utilization of Bulldozer modules
-Server applications are often compiled for a specific archetecture
-Bulldozer contains many advanced extensions to the x86 ISA like AVX, SSE4.1, FMA4, and XOP
-When these extensions are leveraged, the architecture can be fully utilized
-Example: AVX allows the two 128 bit FP units to act as a single 256 bit unit
-High Clock Speed: World record of 8.7GHz
-Scalable and in line with Fusion vision

Tweaking Bulldozer: Piledriver

-Hard edged flip-flops for lower power consumption and higher clocks along with architectural tweaks



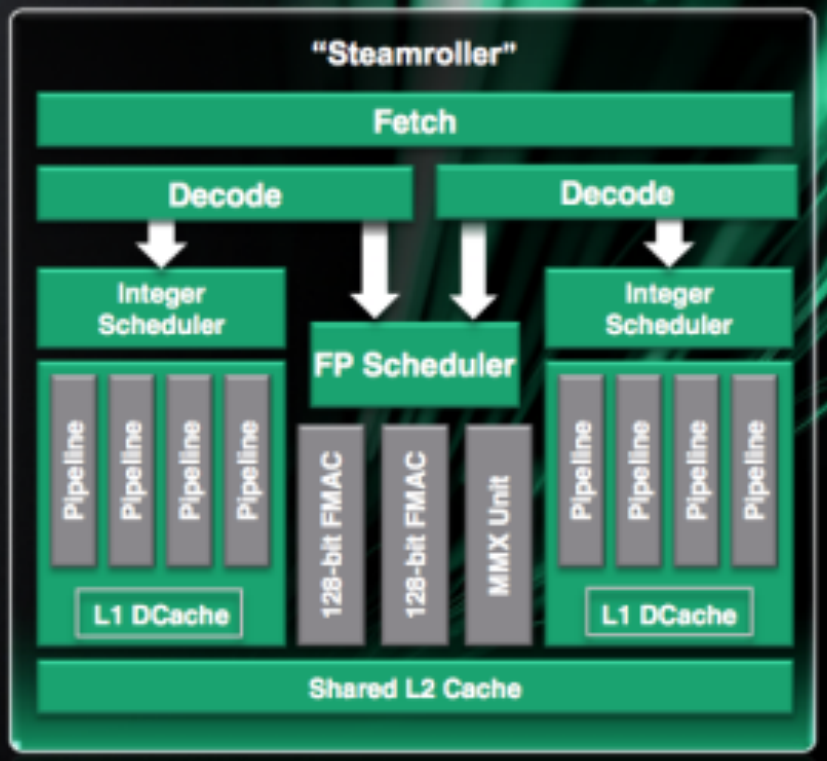
Fixing Bulldozer: Steamroller

AMD'S "STEAMROLLER" CORE

AMD's next-generation, multi-threaded "Steamroller" Microarchitecture

Expand computation efficiency across design
- Target "real-world" client and server applications
- Maintain high-frequency engine

- **Feed the cores faster**
- **Improve single-core execution**
- **Push on performance/watt**



Fixing Bulldozer: Steamroller

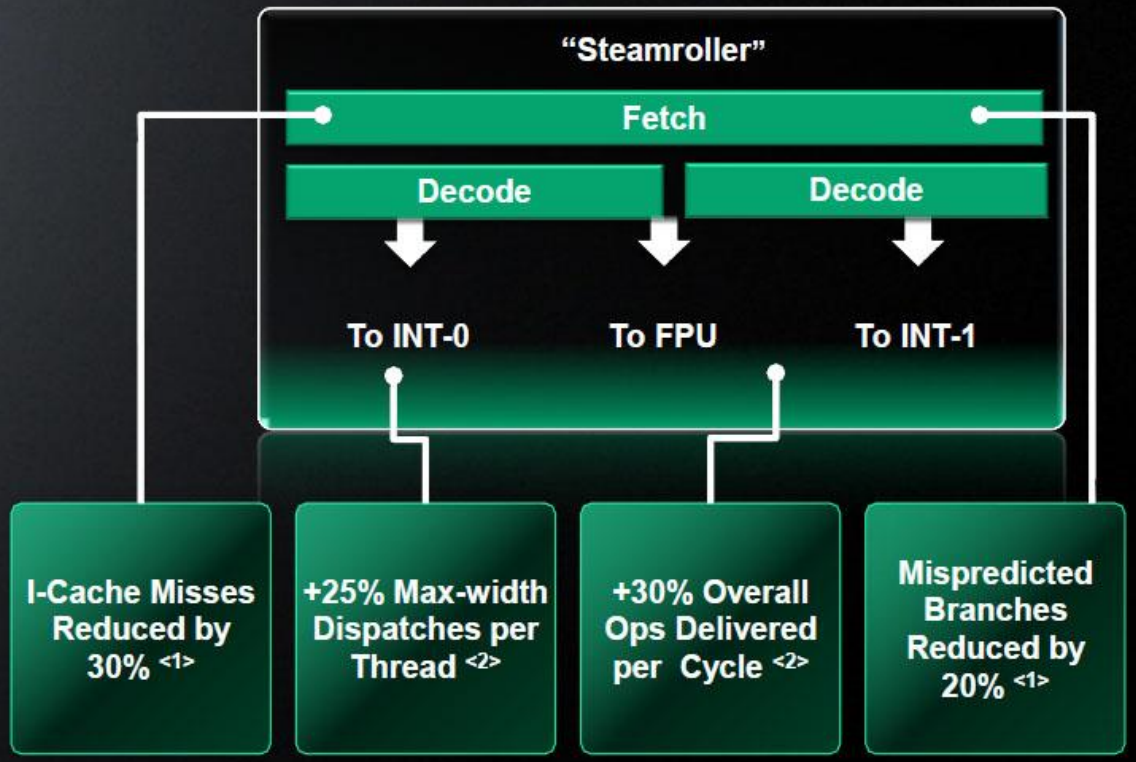
"STEAMROLLER": FEED THE CORES FASTER

Server-focused

- Better containment of code footprint
- Increase instruction cache size
- Enhance instruction prefetch
- Expand branch detection

Client-focused

- Improve dispatch bandwidth
- Wider decode
- Simultaneous dispatch for MT
  - Ops to both cores each cycle

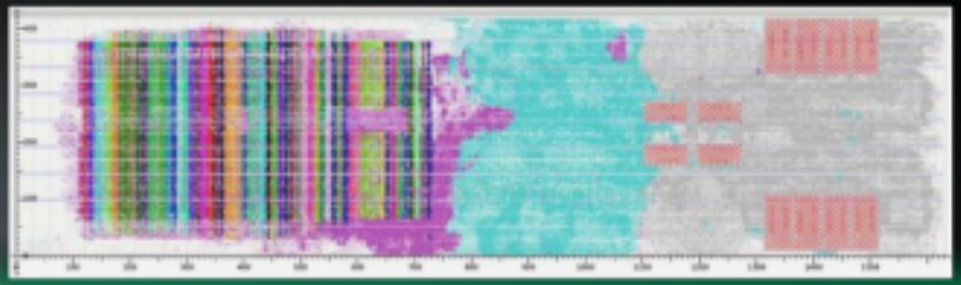


<1>: Multithreaded Server mix: Based on simulation average workloads of simulated performance on a number of tests, including those testing transactional processing (Systems have to be publicly available to publish SPEC CPU Rate.)
<2>: Multithreaded Client mixBased on simulation average workloads of simulated performance on a number of tests, includindigital media, productivity, gaming apps

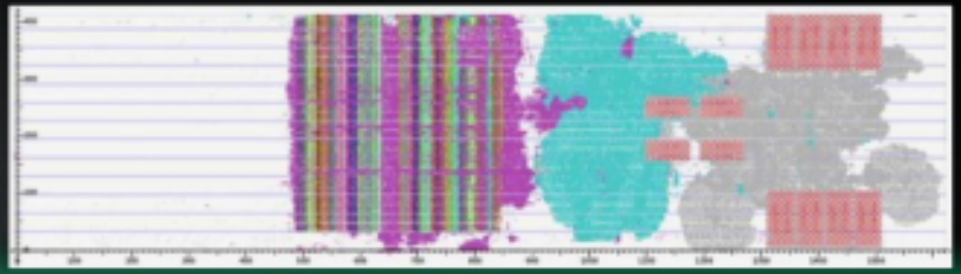
AMD

Fixing Bulldozer: The Future

POWER EFFICIENCY GAINS FROM IMPROVED DESIGN METHODS



"Bulldozer"
Part of the Floating Point Unit. Hand-drawn for maximum speed and density in 32nm.



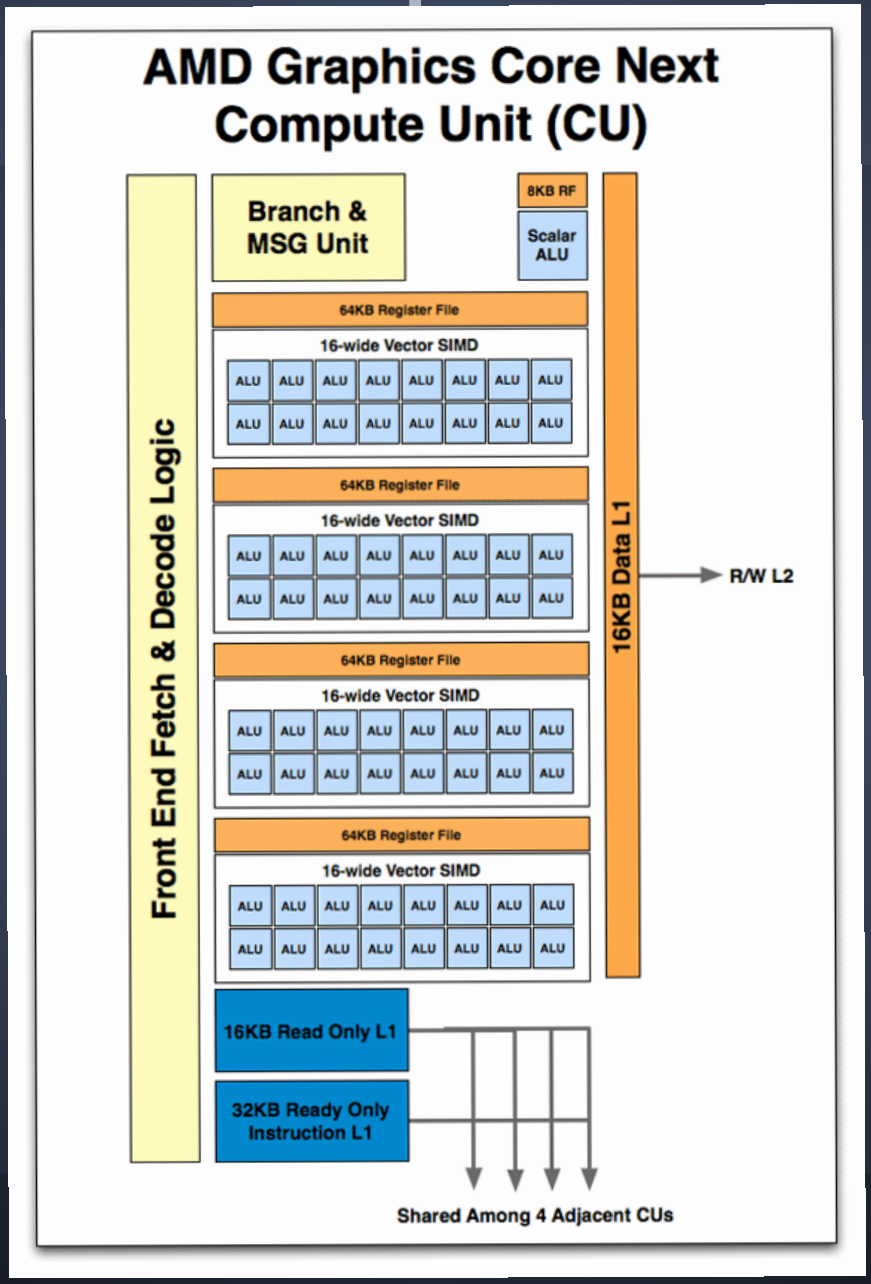
With High Density Library
The same blocks again, but rebuilt using a **High-Density** cell library to achieve **30% area and power reductions.**

15%-30% lower energy per operation[1] for power constrained designs – same order as a full process node improvement

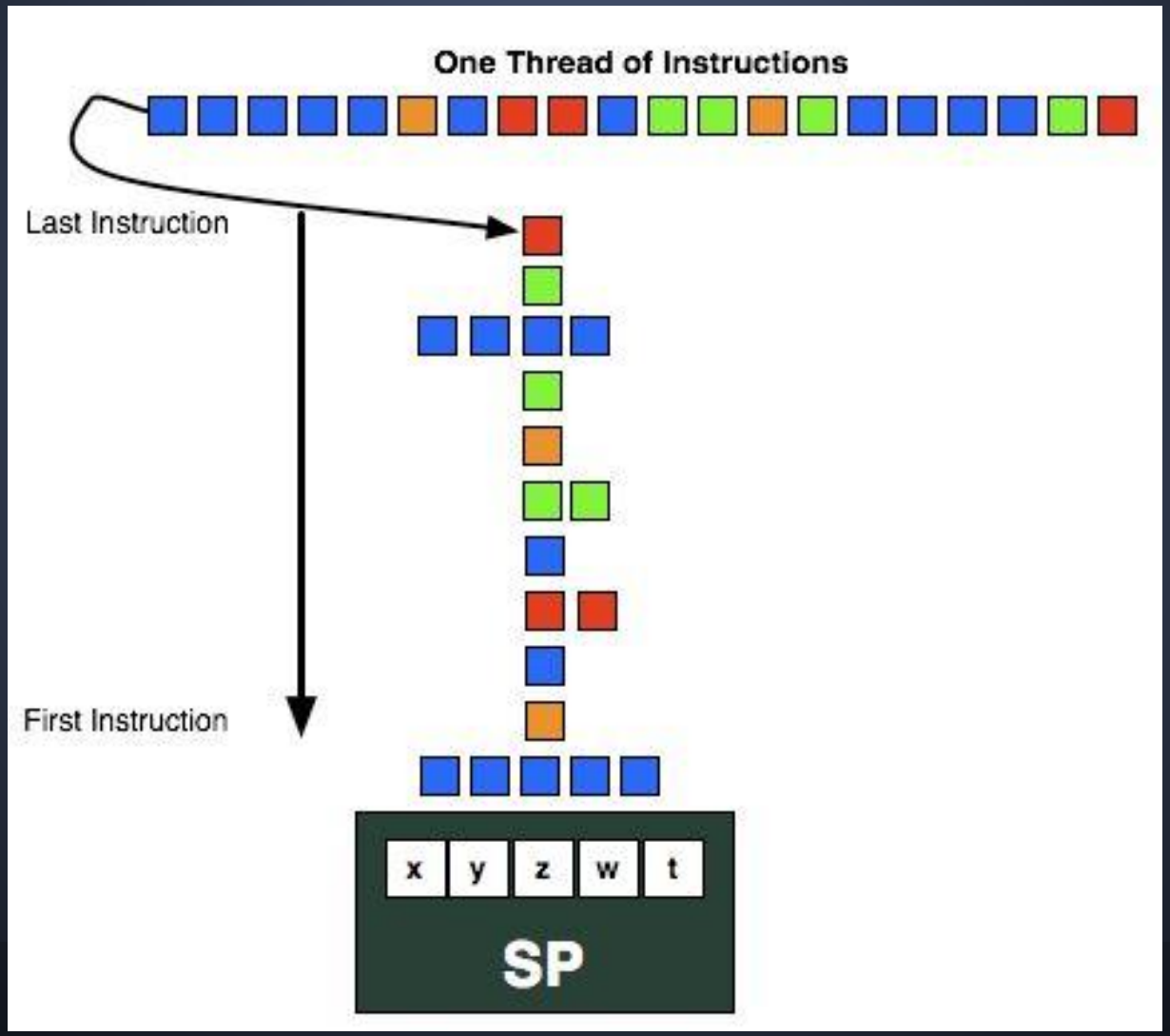
AMD

1. Based on AMD's internal analysis

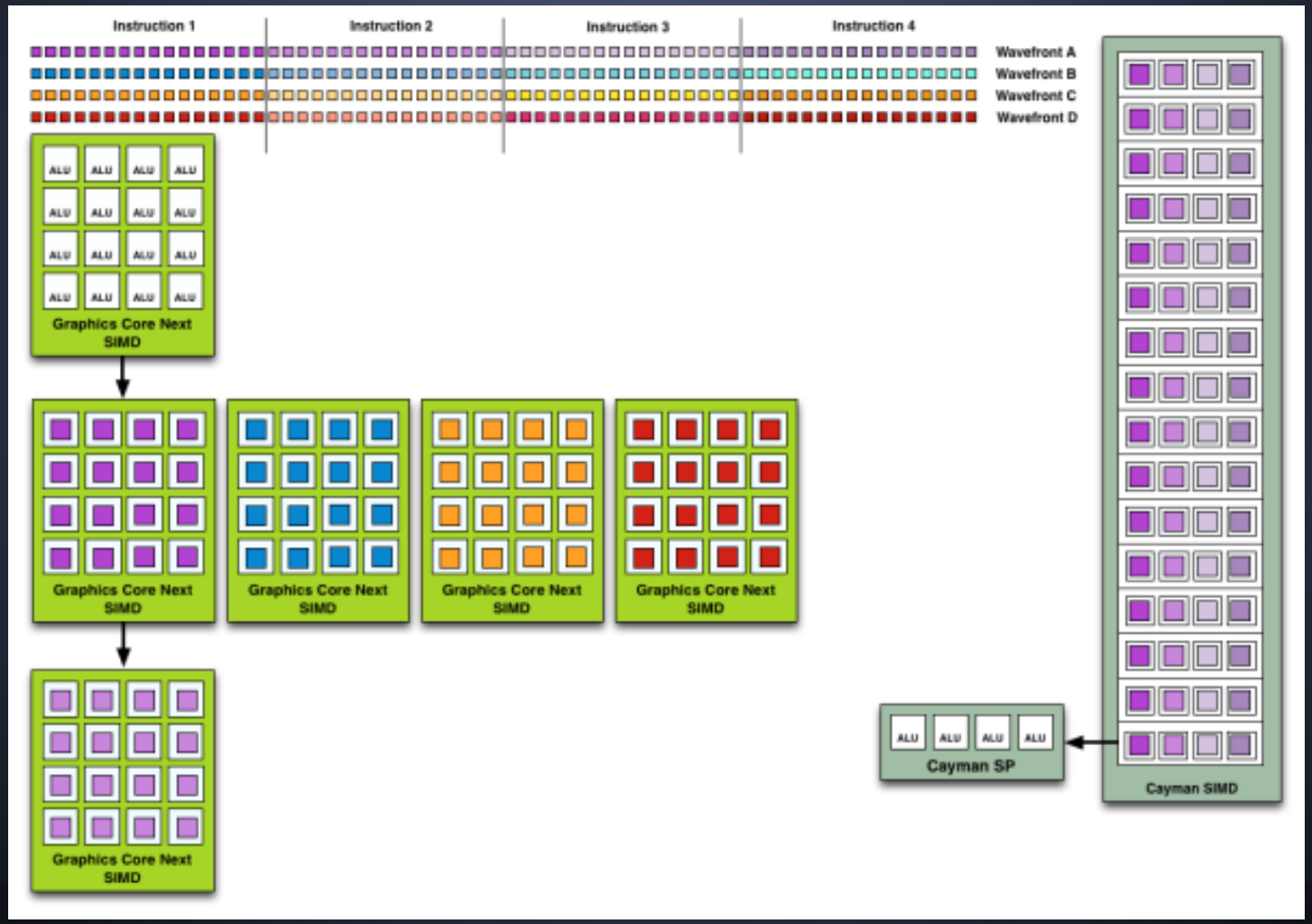
The GPU Half: Graphics Core Next



The GPU Half: VLIW vs SIMD



The GPU Half: VLIW vs SIMD



The GPU Half: VLIW vs SIMD

VLIW

```
// Registers r0 contains "a", r1 contains "b"
// Value is returned in r2
00   ALU_PUSH_BEFORE
       1  x: PREDGT      ____, R0.x,  R1.x
             UPDATE_EXEC_MASK UPDATE PRED
01 JUMP   ADDR(3)
02 ALU
       2  x: SUB         ____, R0.x,  R1.x
       3  x: MUL_e       R2.x, PV2.x, R0.x
03 ELSE POP_CNT(1) ADDR(5)
04 ALU_POP_AFTER
       4  x: SUB         ____, R1.x,  R0.x
       5  x: MUL_e       R2.x, PV4.x, R1.x
05 POP(1) ADDR(6)
```

Non-VLIW SIMD

```
// Registers r0 contains "a", r1 contains "b"
// Value is returned in r2
v_cmp_gt_f32        r0,r1          //a > b, establish VCC
s_mov_b64           s0,exec        //Save current exec mask
s_and_b64           exec,vcc,exec  //Do "if"
s_cbranch_vccz      label0         //Branch if all lanes fail
v_sub_f32           r2,r0,r1       //result = a - b
v_mul_f32           r2,r2,r0       //result=result * a


s_andn2_b64         exec,s0,exec   //Do "else" (s0 & !exec)
s_cbranch_execz     label1         //Branch if all lanes fail
v_sub_f32           r2,r1,r0       //result = b - a
v_mul_f32           r2,r2,r1       //result = result * b

s_mov_b64           exec,s0        //Restore exec mask
```

What GCN Means for Fusion

-Incredibly high compute ability approaching 10 TFLOPS on the high-end 7990
-High end CPU's manage only 0.1TFLOPS
-GPU architecture now more closely resembles CPU architecture.
-Easier scheduling of general purpose computations
-Native C++ support including pointers, virtual functions, exceptions, and recursion
-Paves the way for shared resources between CPU/GPU with its I/O memory mapping unit that maps in the x86-64 space.
-Shared resources will be realized this year with the "Kaveri" APU

What Fusion Means for Computing

-Many times higher FLOPS with the GPU

-No coding/time expense to leverage GPU

-Cheaper devices

-Less power consumption

-Less heat

-Resource intensive interfaces like Kinect become commonplace

Ultimate Goal for Fusion

-Seamless integration of GPU and CPU
-No special coding necessary
-APU will automatically context switch between the GPU's stream units and the CPU's ALU/FPU
-True realization of a heterogeneous system

What is Necessary for a True HSA?

Achieved with GCN:

-GPU C++ compute support

-GPU uses pageable memory via CPU pointers

-Unified address space for GPU/CPU

Coming with Kaveri:

-HSA memory management

-Fully coherent memory between GPU/CPU

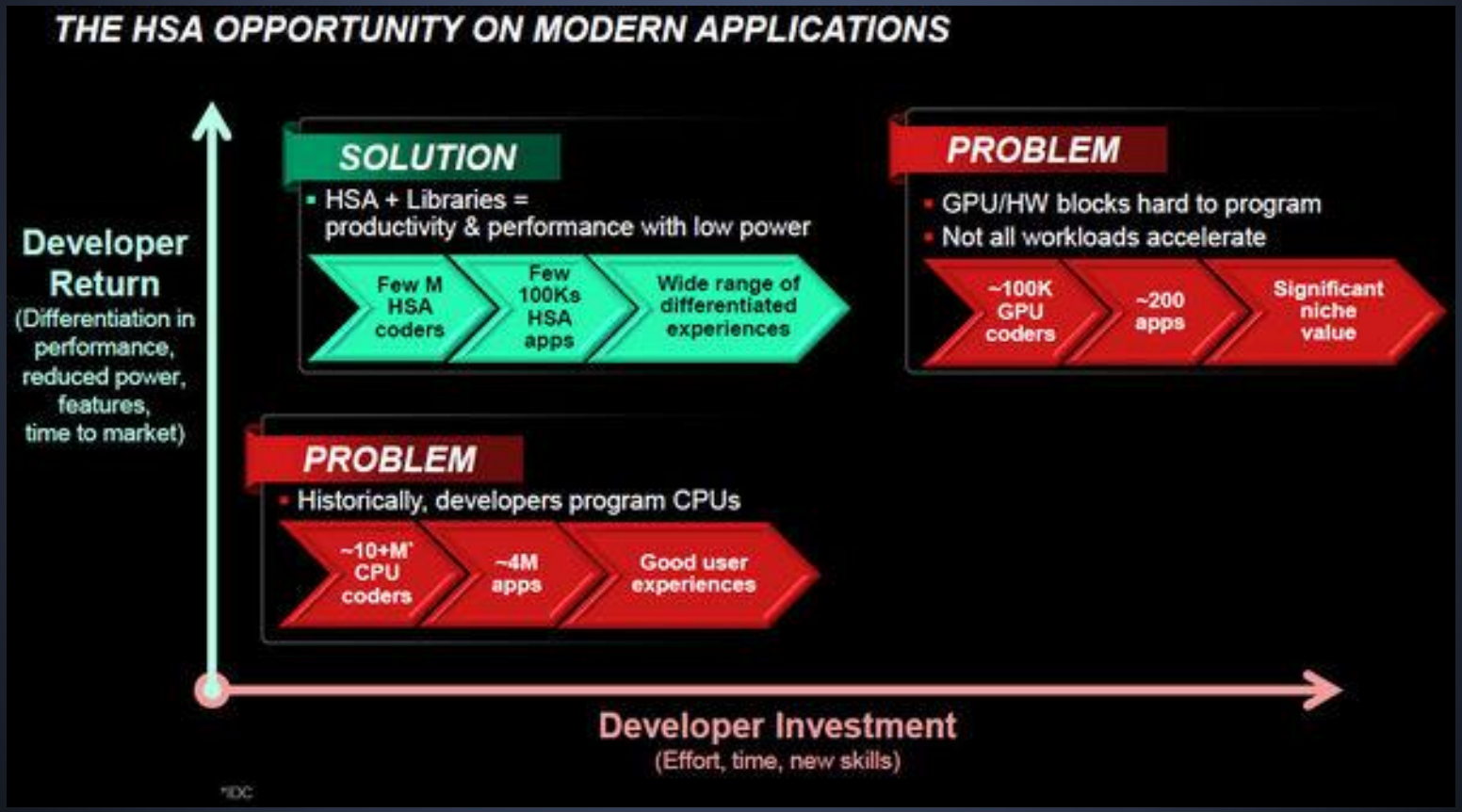
Coming with future Excavator/Sea Island APUs:

-GPU context switching and graphics pre-emption

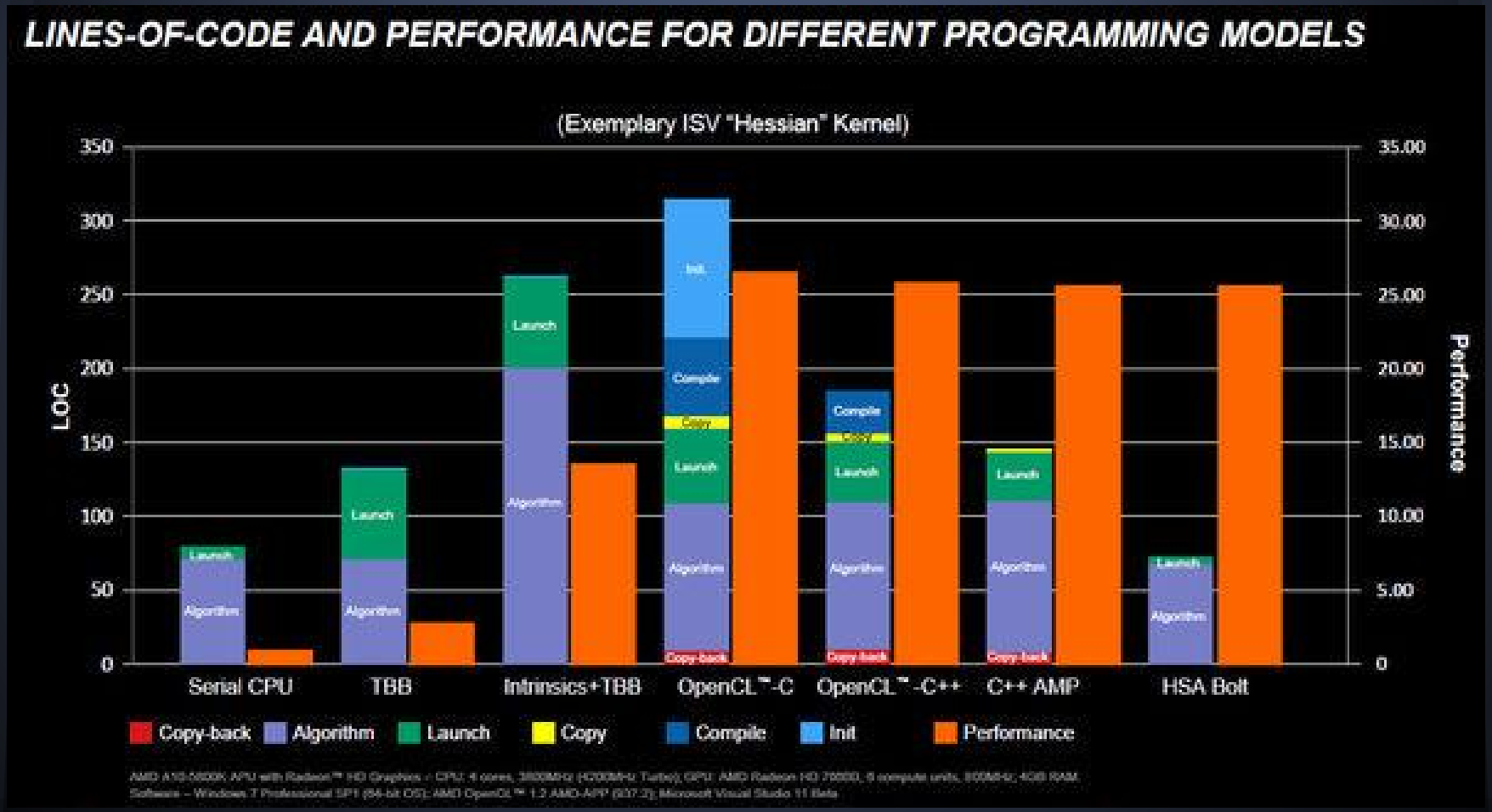
Software Support

Component Name	AMD Specific	Rationale
HSA Bolt Library	No	Enable understanding and debug
OpenCL HSAIL Code Generator	No	Enable research
LLVM Contributions	No	Industry and academic collaboration
HSA Assembler	No	Enable understanding and debug
HSA Runtime	No	Standardize on a single runtime
HSA Finalizer	Yes	Enable research and debug
HSA Kernel Driver	Yes	For inclusion in linux distros

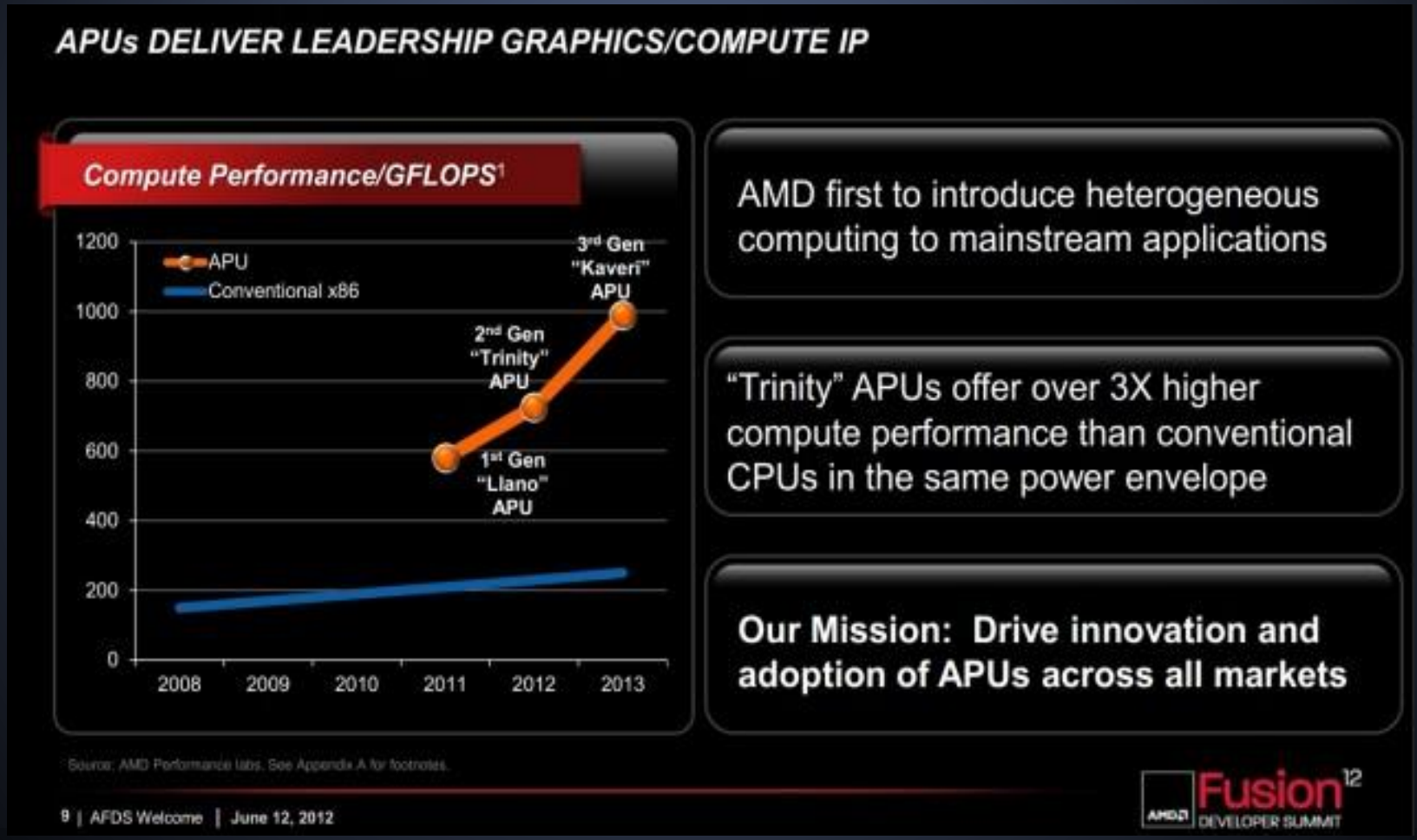
Software Support Goal



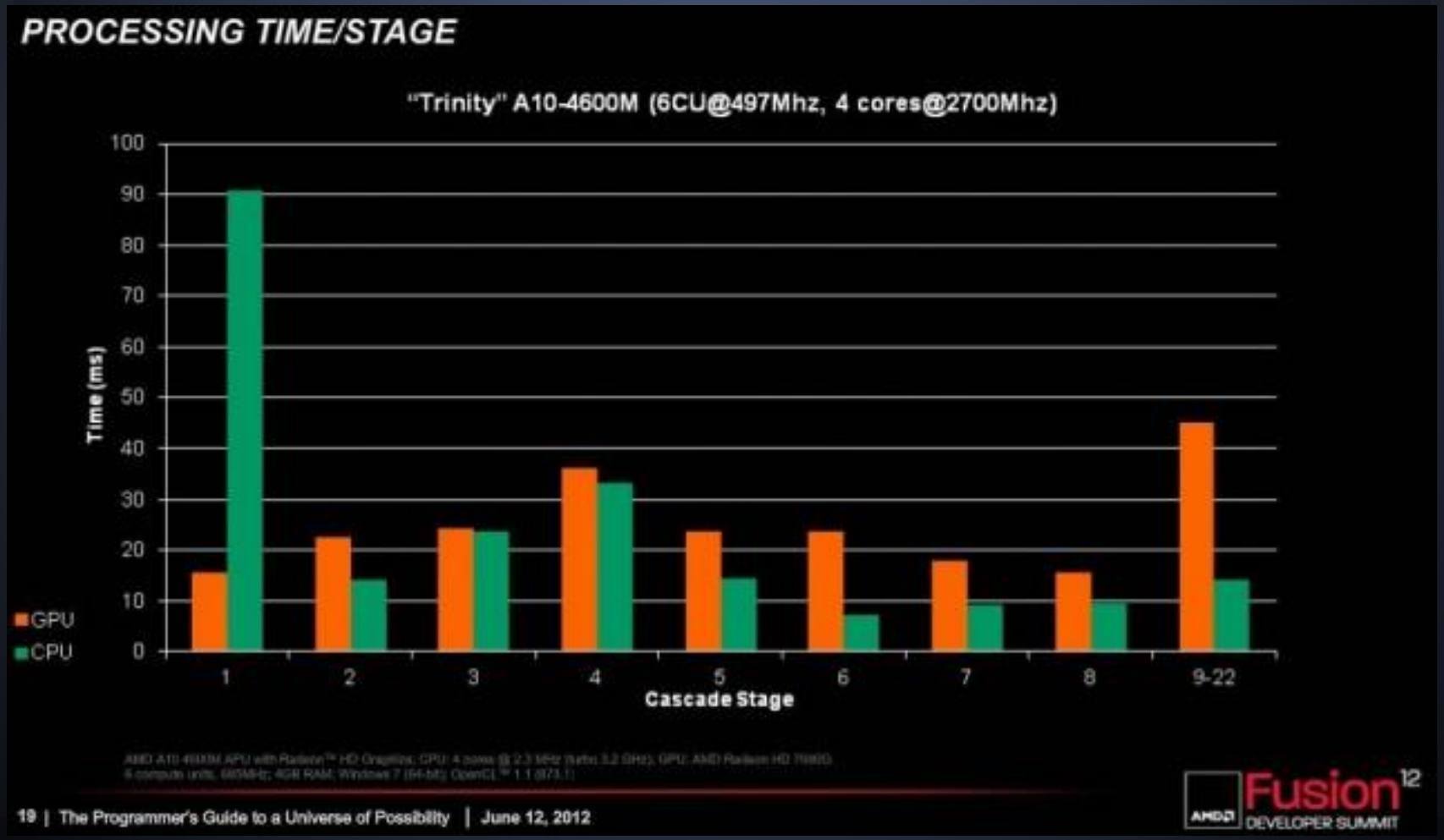
Software Support Advantage



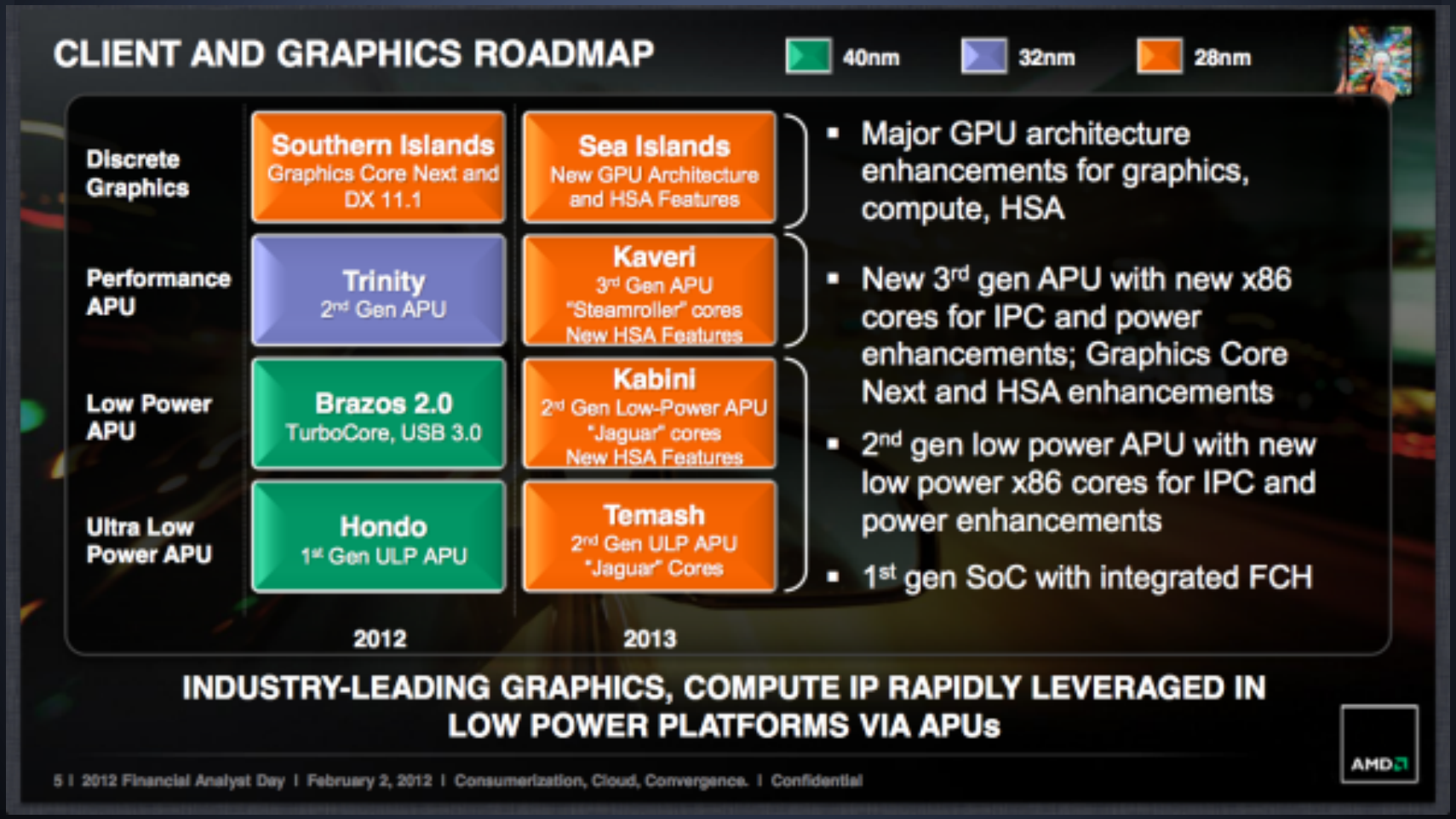
Fusion Performance



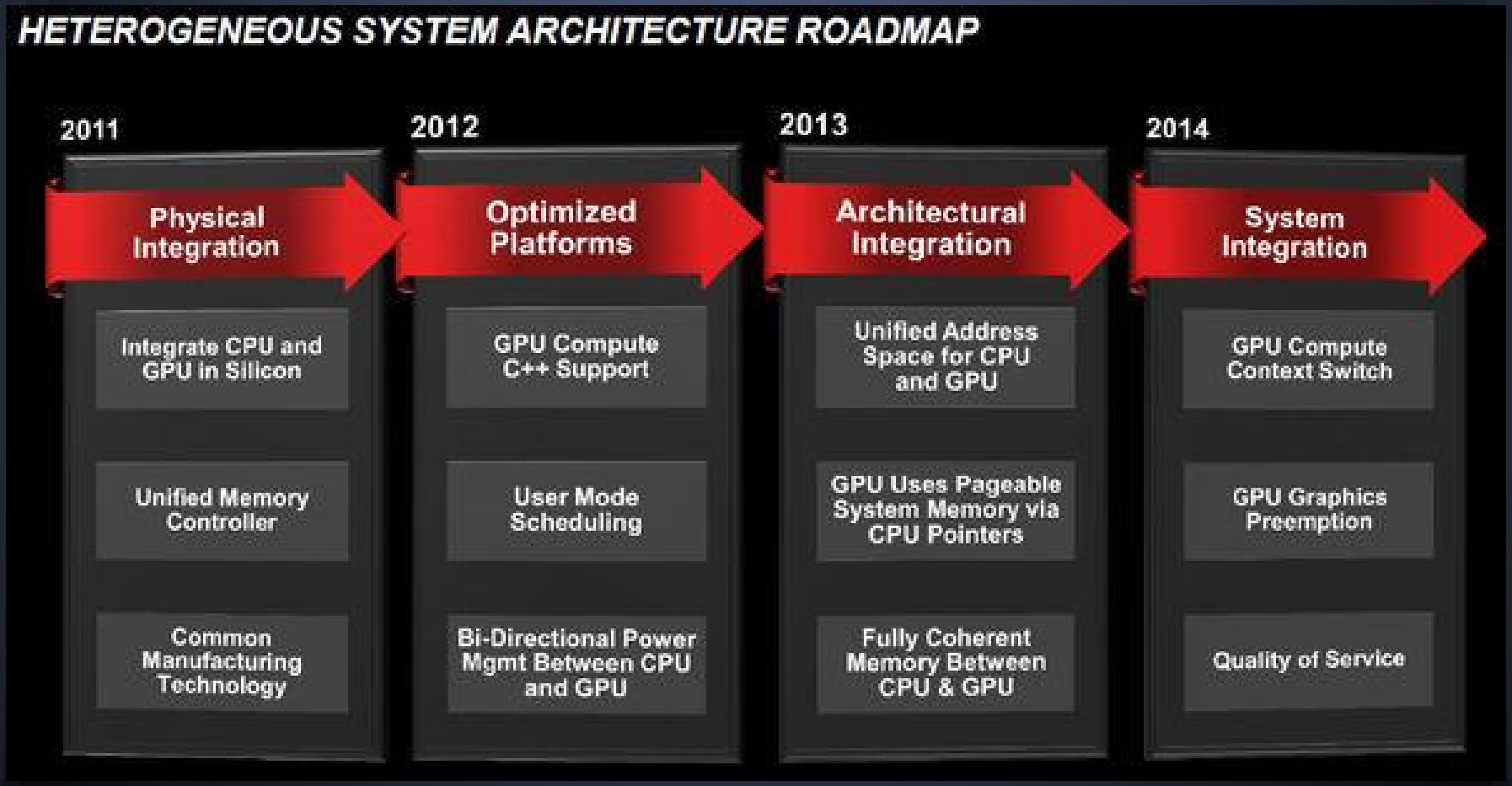
Performance Example: Face Detection



AMD's Fusion Roadmap



AMD's Fusion Roadmap



Conclusion

-Since AMD's $5.4 Billion purchase of ATI in 2006, they have been working toward creating a processor that is a true combination of GPU and CPU

-Design decisions made on both the CPU and GPU separately in order to prepare for their convergence

-This point of convergence is coming next year

-APU's will transform from a traditional CPU with an integrated GPU into a new entity that treats the two as the same processor, seamlessly scheduling code between the CPU and GPU components