Introducing Sandy Bridge

Bob Valentine
Senior Principal Engineer
Sandy Bridge: Overview

Integrates CPU, Graphics, MC, PCI Express® On Single Chip

Next Generation Intel® Turbo Boost Technology
High Bandwidth Last Level Cache
Next Generation Processor Graphics and Media
Embedded Display Port
Discrete Graphics Support: 1x16 or 2x8

High BW/low-latency modular core/GFX interconnect
Substantial performance improvement
Intel® Advanced Vector Extension (Intel® AVX)
Integrated Memory Controller 2ch DDR3
Intel® Hyper-Threading Technology
4 Cores / 8 Threads
2 Cores / 4 Threads

Energy Efficiency
Stunning Performance
Agenda

• Innovation in the Processor core

• System Agent, Ring Architecture and Other Innovations

• Innovation in Power Management
Introduction to Sandy Bridge Processor Core Microarchitecture
Outline

• Sandy Bridge Processor Core Summary

• Core Major Microarchitecture Enhancements

• Core Architectural Enhancements

• Processor Core Summary
Sandy Bridge Processor Core Summary

• Build upon the successful Nehalem microarchitecture processor core
  – Converged building block for mobile, desktop, and server

• Add “Cool” microarchitecture enhancements
  – Features that are better than linear performance/power

• Add “Really Cool” microarchitecture enhancements
  – Features which gain performance while saving power

• Extend the architecture for important new applications
  – Floating Point and Throughput
    ▪ Intel® Advanced Vector Extensions (Intel® AVX) - Significant boost for selected compute intensive applications
  – Security
    ▪ AES (Advanced Encryption Standard) throughput enhancements
    ▪ Large Integer RSA speedups
  – OS/VMM and server related features
    ▪ State save/restore optimizations
Processor Core Tutorial
Microarchitecture Block Diagram

Front End
(IA instructions → Uops)

In Order Allocation, Rename, Retirement

Out of Order “Uop” Scheduling

Six Execution Ports

L2 Data Cache (MLC)

Data Cache Unit

In order

Out-of-order
Front End Microarchitecture

Instruction Decode in Processor Core
- 32 Kilo-byte 8-way Associative ICache
- 4 Decoders, up to 4 instructions / cycle
- Micro-Fusion
  - Bundle multiple instruction events into a single “Uops”
- Macro-Fusion
  - Fuse instruction pairs into a complex “Uop”
- Decode Pipeline supports 16 bytes per cycle
Add a Decoded Uop Cache

- An L0 Instruction Cache for Uops instead of Instruction Bytes
  - ~80% hit rate for most applications
- Higher Instruction Bandwidth and Lower Latency
  - Decoded Uop Cache can represent 32-byte / cycle
    - More Cycles sustaining 4 instruction/cycle
  - Able to ‘stitch’ across taken branches in the control flow
New Branch Prediction Unit

Do a ‘Ground Up’ Rebuild of Branch Predictor
- Twice as many targets
- Much more effective storage for history
- Much longer history for data dependent behaviors
“Really Cool” features in the front end
- Decoded Uop Cache lets the normal front end sleep
  - Decode one time instead of many times
- Branch-Mispredictions reduced substantially

“Really Cool” Features
Save Power while Increasing Performance
Power is fungible...
...give it to other units in this core, or other units on die
“Out of Order” Cluster

In Order Allocation, Rename, Retirement

Out of Order “Uop” Scheduling

- Receives Uops from the Front End
- Sends them to Execution Units when they are ready
- Retires them in Program Order

- Goal: Increase Performance by finding more Instruction Level Parallelism
  - Increasing Depth and Width of machine implies larger buffers
    - More Data Storage, More Data Movement, More Power

Challenge to the OoO Architects:
Increase the ILP while keeping the power available for Execution
Sandy Bridge Out-of-Order (OOO) Cluster

• Method: Physical Reg File (PRF) instead of centralized Retirement Register File
  – Single copy of every data
  – No movement after calculation
• Allow significant increase in buffer sizes
  – Dataflow window ~33% larger

PRF is a “Cool” feature better than linear performance/power

Key enabler for Intel® Advanced Vector Extensions (Intel® AVX)

<table>
<thead>
<tr>
<th></th>
<th>Nehalem</th>
<th>Sandy Bridge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load Buffers</td>
<td>48</td>
<td>64</td>
</tr>
<tr>
<td>Store Buffers</td>
<td>32</td>
<td>36</td>
</tr>
<tr>
<td>RS - Scheduler Entries</td>
<td>36</td>
<td>54</td>
</tr>
<tr>
<td>PRF integer</td>
<td>N/A</td>
<td>160</td>
</tr>
<tr>
<td>PRF float-point</td>
<td>N/A</td>
<td>144</td>
</tr>
<tr>
<td>ROB Entries</td>
<td>128</td>
<td>168</td>
</tr>
</tbody>
</table>
Execution Cluster

• 3 Execution Ports
• Maximum throughput of 8 floating point operations* per cycle
  – Port 0: packed SP multiply
  – Port 1: packed SP add

*FLOPS = Floating Point Operations / Second
Doubling the FLOPs in a “Cool” Manner

- Intel® Advanced Vector Extensions (Intel® AVX)

- Extend SSE FP instruction set to 256 bits operand size
  - Intel AVX extends all 16 XMM registers to 256 bits

- New, non-destructive source syntax
  - VADDPS ymm1, ymm2, ymm3

- New Operations to enhance vectorization
  - Broadcasts
  - Masked load & store
Intel® Advanced Vector Extensions (Intel® AVX)

• Doubling the FLOPs in a cool manner

• Extend SSE FP instruction set to 256 bits operand size
  – Intel AVX extends all 16 XMM registers to 256bits

• New, non-destructive source syntax
  – VADDPS ymm1, ymm2, ymm3

• New Operations to enhance vectorization
  – Broadcasts
  – Masked load & store

Intel® AVX is a “Cool” Architecture
Vectors are a natural data-type for many applications
Wider vectors and non-destructive source specify more work with fewer instructions
Extending the existing state is area and power efficient
Execution Cluster – A Look Inside

Scheduler sees matrix:

- 3 “ports” to 3 “stacks” of execution units
- General Purpose Integer
  - SIMD (Vector) Integer
  - SIMD Floating Point
- The challenge is to double the output of one of these stacks in a manner that is invisible to the others
Execution Cluster

**Solution:**
- Repurpose existing datapaths to dual-use
- SIMD integer and legacy SIMD FP use legacy stack style
- Intel® AVX utilizes *both* 128-bit execution stacks
Execution Cluster

Solution:

- Repurpose existing datapaths to dual-use
- SIMD integer and legacy SIMD FP use legacy stack style
- Intel® AVX utilizes both 128-bit execution stacks

“Cool” Implementation of Intel AVX
256-bit Multiply + 256-bit ADD + 256-bit Load per clock...
Double your FLOPs with great energy efficiency
Memory Cluster

- Memory Unit can service two memory requests per cycle
  - 16 bytes load and 16 bytes store per cycle

Challenge to the Memory Cluster Architects

Maintain the historic bytes/flop ratio of SSE for Intel® AVX

...and do so in a “cool” manner
Memory Cluster in Sandy Bridge

- **Solution**: Dual-Use the existing connections
  - Make load/store pipes symmetric
- **Memory Unit** services **three** data accesses per cycle
  - 2 read requests of up to 16 bytes AND 1 store of up to 16 bytes
  - Internal sequencer deals with queued requests
Memory Cluster in Sandy Bridge

• Solution: Dual-Use the existing connections
  – Make load/store pipes symmetric
• Memory Unit services **three** data accesses per cycle
  – 2 read requests of up to 16 bytes AND 1 store of up to 16 bytes
  – Internal sequencer deals with queued requests

Second Load Port is one of highest performance features
Required to keep Intel® Advanced Vector Extensions (Intel® AVX)
Instruction Set fed linear power/performance means its “Cool”
Putting it together
Sandy Bridge Microarchitecture

32k L1 Instruction Cache

Branch Pred

Load Buffers
Store Buffers
Reorder Buffers

Pre decode

Instruction Queue

Decoders

1.5k uOP cache

Allocate/Rename/Reti
Zeroing Idioms

32k L1 Data Cache

48 bytes/cycle

L2 Data Cache (MLC)

Store Address

Load

In order

Out-of-order

Port 0
Port 1
Port 5
Port 2
Port 3
Port 4

ALU
VI MUL
VI Shuffle
DIV
AVX FP MUL
AVX FP Blend

ALU
VI ADD
VI Shuffle
AVX FP Shuf
AVX FP Bnd
AVX FP Blend

ALU
JMP

Memory Control

Load
Store Data

32k L1 Data Cache

Load

Fill Buffers

Sandy Bridge - Intel® Next Generation Microarchitecture
AVX = Intel® Advanced Vector Extensions (Intel® AVX)
Other Architectural Extensions

• Cryptography Instruction Throughput Enhancements
  – throughput for AES instructions introduced in Westmere

• Large Number Arithmetic Throughput Enhancements
  – ADC (Add with Carry) throughput doubled
  – Multiply (64-bit multiplicands with 128-bit product)
    ▪ ~25% speedup on existing RSA binaries!

• State Save/Restore Enhancements
  – New state added in Intel® Advanced Vector Extensions (Intel® AVX)
  – HW monitors features used by applications
    ▪ Only saves/restores state that is used
Sandy Bridge Processor Core Summary

• Build upon the successful Nehalem processor core
  – Converged building block for mobile, desktop, and server
  – “Cool” and “Really Cool” features
    ▪ Improve performance/power and performance/area

• Extends the architecture for important new applications
  – Floating Point and Throughput Applications
    ▪ Intel® Advanced Vector Extensions (Intel® AVX) - Significant boost for selected compute intensive apps
  – Security
    ▪ AES (Advanced Encryption Standard) Instructions speedup
    ▪ Large Integer RSA and SHA speedups
  – OS/VMM and server related features
    ▪ State save/ restore optimizations
System Agent, Ring Architecture and Other Innovations
Integration: Optimization Opportunities

- **Dynamically redistribute** power between Cores & Graphics
- **Tight power management** control of all components, providing better granularity and deeper idle/sleep states
- Three **separate power/frequency domains**: System Agent *(Fixed)*, Cores+Ring, Graphics *(Variable)*

- High BW Last Level Cache, **shared** among Cores and Graphics
  - Significant performance boost, saves memory bandwidth and power
- Integrated **Memory Controller** and **PCI Express*** ports
  - Tightly integrated with Core/Graphics/LLC domain
  - Provides low latency & low power – remove intermediate busses

- **Bandwidth is balanced** across the whole machine, from Core/Graphics all the way to Memory Controller
- Modular uArch for **optimal cost/power/performance**
  - Derivative products done with minimal effort/time
Scalable Ring On-die Interconnect

- **Ring-based** interconnect between Cores, Graphics, Last Level Cache (LLC) and System Agent domain
- Composed of **4 rings**
  - 32 Byte *Data* ring, *Request* ring, *Acknowledge* ring and *Snoop* ring
  - Fully pipelined at core frequency/voltage: bandwidth, latency and power scale with cores
- Massive ring **wire routing** runs over the LLC with no area impact
- Access on ring always picks the **shortest path** – minimize latency
- **Distributed arbitration**, sophisticated ring protocol to handle coherency, ordering, and core interface
- **Scalable to servers** with large number of processors

*High Bandwidth, Low Latency, Modular*
**Cache Box**

- **Interface block**
  - Between **Core/Graphics/Media** and the **Ring**
  - Between **Cache controller** and the **Ring**
  - Implements the **ring logic, arbitration, cache controller**
  - Communicates with **System Agent** for LLC misses, external snoops, non-cacheable accesses

- **Full cache pipeline** in each cache box
  - Physical Addresses are **hashed at the source** to **prevent hot spots** and increase bandwidth
  - Maintains **coherency and ordering** for the addresses that are mapped to it
  - LLC is fully inclusive with “**Core Valid Bits**” – eliminates unnecessary snoops to cores

- Runs at **core voltage/frequency**, scales with Cores

---

*Block Diagram Illustrative only. Number of processor cores will vary with different processor models based on the Sandy Bridge Microarchitecture. Represents client processor implementation.*
Ring Illustration: Clean LLC Hit
Ring Illustration: Clean LLC Hit
Ring Illustration: Clean LLC Hit

- System Agent
- iMC
- DDR
- Core
- MLC
- LLC
- Core
- MLC
- LLC
- Graphics
- DRd

Request
Data
Global Observation
Ring Illustration: Clean LLC Hit

- Core
  - MLC
  - DRd
  - LLC

- Graphics
  - Ring IFC
  - DRd

System Agent

iMC

DDR

Request
Data
Global Observation
Ring Illustration: Clean LLC Hit

System Agent

iMC

DDR

Core

MLC

Core

MLC

Core

MLC

Graphics

Ring IFC

Request

Data

Global Observation
Ring Illustration: Clean LLC Hit

- System Agent
- iMC
- DDR

Core
- MLC
- Hit (No Snoop)

Core
- MLC
- Hit (No Snoop)

Graphics
- Ring IFC

Request  Data  Global Observation
Ring Illustration: Clean LLC Hit

System Agent

DDR

Core

MLC

Core

MLC

Graphics

Ring IFC

Request  Data  Global Observation
Ring Illustration: Clean LLC Hit

System Agent

Core
MLC
Core
MLC
Graphics
Ring IFC

iMC
DDR

GO

No Snoop

D1
D2

Request
Data
Global Observation
Ring Illustration: Clean LLC Hit

System Agent

iMC

DDR

Core

MLC

GO

D1

Core

MLC

D1

GO

Graphics

Ring IFC

Request

Data

Global Observation
Ring Illustration: Clean LLC Hit

- System Agent
- iMC
- DDR
- Core (GO_LC)
- Core (GO_LC)
- Graphics (Ring IFC)

Request
Data
Global Observation
Ring Illustration: Clean LLC Hit

System Agent

Core

M-LC

D2

LLC

Core

M-LC

D2

LLC

Graphics

Ring IFC

GO

Request

Data

Global Observation

DDR

iMC
Ring Illustration: Clean LLC Hit

- System Agent
- DDR
- Core MLC
- LLC
- Core MLC
- LLC D2
- Graphics Ring IFC

Request Data Global Observation

31
Ring Illustration: Clean LLC Hit

- System Agent
- iMC
- DDR
- Core MLC
- Core MLC
- Graphics Ring IFC

Request  Data  Global Observation
Ring Illustration: Clean LLC Hit

System Agent

Core
MLC

Core
MLC

Graphics
Ring IFC

Resource Agent
iMC

Request

Data

Global Observation

DDR
Ring Illustration: Clean LLC Hit

- System Agent
- iMC
- DDR
- Core
  - MLC
  - LLC
- Core
  - MLC
  - LLC
- Graphics
  - Ring IFC

Request
Data
Global Observation
Sandy Bridge LLC Sharing

- **LLC shared** among all Cores, Graphics and Media
  - Graphics driver controls **which streams** are cached/coherent
  - **Any agent** can access all data in the LLC, independent of who allocated the line, after **memory range checks**
- Controlled LLC **way allocation** mechanism to prevent thrashing between Core/graphics
- Multiple coherency domains
  - **IA Domain** (**Fully coherent via cross-snoops**)  
  - **Graphic domain** (**Graphics virtual caches, flushed to IA domain by graphics engine**)  
  - **Non-Coherent domain** (**Display data, flushed to memory by graphics engine**)  

**Much higher Graphics performance, DRAM power savings, more DRAM BW available for Cores**

Block Diagram Illustrative only. Number of processor cores will vary with different processor models based on the Sandy Bridge Microarchitecture. Represents client processor implementation. Sandy Bridge - Intel® Next Generation Microarchitecture.
Lean and Mean System Agent

- Contains PCI Express*, DMI, Memory Controller, Display Engine...
- Contains **Power Control Unit**
  - Programmable uController, handles all power management and reset functions in the chip
- **Smart integration** with the ring
  - Provides cores/Graphics/Media with high BW, low latency to DRAM/IO for best performance
  - Handles IO-to-cache coherency
Lean and Mean System Agent

• Contains PCI Express*, DMI, Memory Controller, Display Engine...

• Contains **Power Control Unit**
  – Programmable uController, handles all power management and reset functions in the chip

• **Smart integration** with the ring
  – Provides cores/Graphics /Media with high BW, low latency to DRAM/IO for best performance
  – Handles IO-to-cache coherency

• Separate **voltage and frequency** from ring/cores, **Display integration** for better battery life

• Extensive **power and thermal management** for PCI Express* and DDR

*Smart I/O Integration*

Block Diagram Illustrative only. Number of processor cores will vary with different processor models based on the Sandy Bridge Microarchitecture. Represents client processor implementation.
Power and Thermal Management
Usage Scenario: Responsive Behavior

- Interactive work benefits from Next Generation Intel® Turbo Boost
- Idle periods intermixed with user actions

Example – Photo editing

- Open image
- View
- Process
  - Rotate
  - Balance colors
  - Contrast
  - Corp
  - Zoom
  - Etc.
Innovative Concept: Thermal Capacitance

**Classic Model**
Steady-State Thermal Resistance
Design guide for steady state

**New Model**
Steady-State Thermal Resistance
AND
Dynamic Thermal Capacitance

Temperature rises as energy is delivered to thermal solution
Thermal solution response is calculated at real-time

- Classic model response
- More realistic response to power changes
Next Generation Intel® Turbo Boost Benefit

After idle periods, the system accumulates “energy budget” and can accommodate high power/performance for a few seconds.

In Steady State conditions, the power stabilizes on TDP.

Use accumulated energy budget to enhance user experience.
Core and Graphic Power Budgeting

- Cores and Graphics integrated on the same die with separate voltage/frequency controls; tight HW control
- Full package power specifications available for sharing
- Power budget can shift between Cores and Graphics
Core and Graphic Power Budgeting

- Cores and Graphics integrated on the same die with separate voltage/frequency controls; tight HW control
- Full package power specifications available for sharing
- Power budget can shift between Cores and Graphics

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Heavy CPU workload</td>
<td>Heavy Graphics workload</td>
<td>Sum of max</td>
<td>Realistic concurrent max power</td>
<td>Sandy Bridge Next Gen Turbo for short periods</td>
<td></td>
</tr>
</tbody>
</table>
Summary

32nm Next Generation Microarchitecture

Processor Graphics

System Agent, Ring Architecture and Other Innovations

Performance and Power Efficiency