MPSoC '06

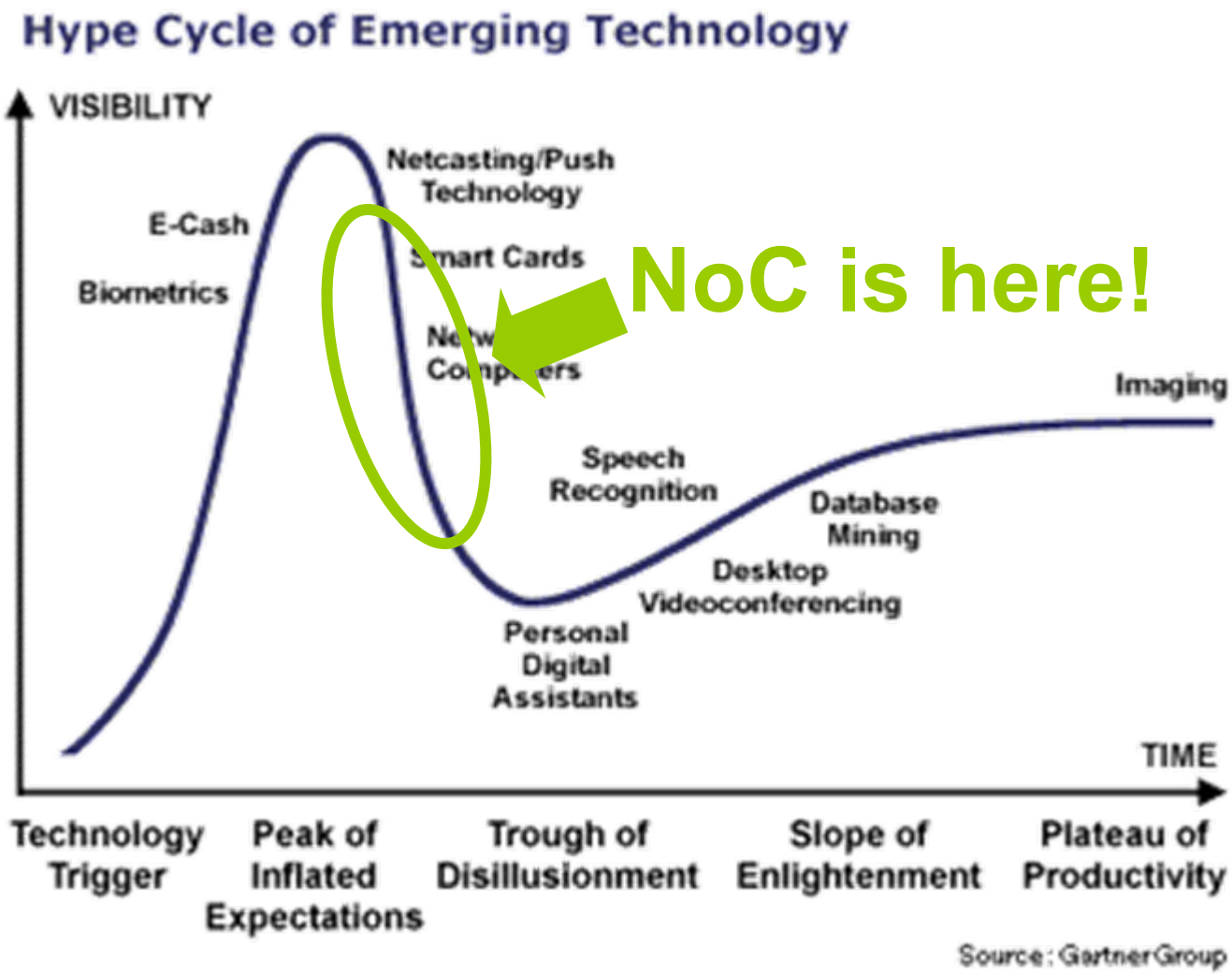
NoC is the Answer!
(What was the Question?)

Drew Wingard, Sonics

NoC's are ***GREAT!!!***

- Apply *Networking* techniques to on-chip communications
  - Packetization and Serialization
  - Routers, Links, and Network Interfaces
  - 7-layer OSI model
- Benefits
  - Higher Performance!
  - Lower Power!
  - Lower Routing and Gate Area!
  - Higher Scalability!
  - Higher Productivity and Quality!

Gartner Hype Cycle Model



NoC is the Answer! (What was the Question?)

- Problem Statement:
  - "MPSoC's cost too much, take too long, and usually encounter timing convergence and application performance issues"
- Assumed Root Cause:
  - On-chip busses (and bus matrices) don't scale
    - Both due to total connectivity and wire scaling
- Proposed Answer:
  - NoC
    - In all of its glory…

My Thesis:
Tightly-coupled Design Doesn't Scale
(Computer Busses are Symptom)

My Reference Point

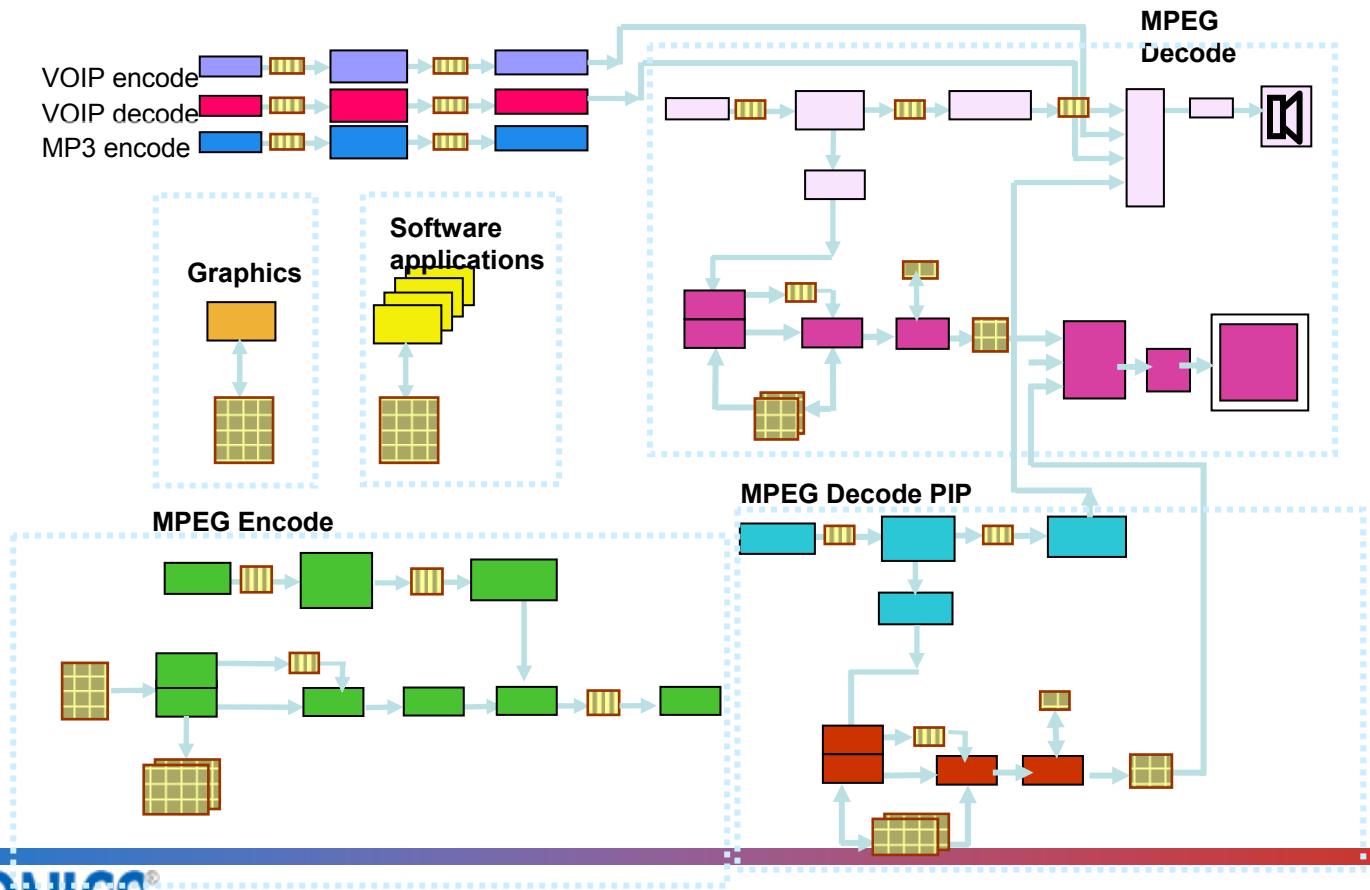
- MPSoC design projects are more expensive than uni-processor SoC
  - Should consider NoC in context of high-volume applications
- Favorite MPSoC applications:
  - Wireless phones
  - HDTV/STB
  - Gaming New!
  - Printers New!
- NoC benefits should be measured against requirements of above applications

NoC Myths

1. MPSoC applications offer lots of network-level concurrency
2. Packetization and serialization are the best way to minimize implementation costs
3. NoC latency is acceptable

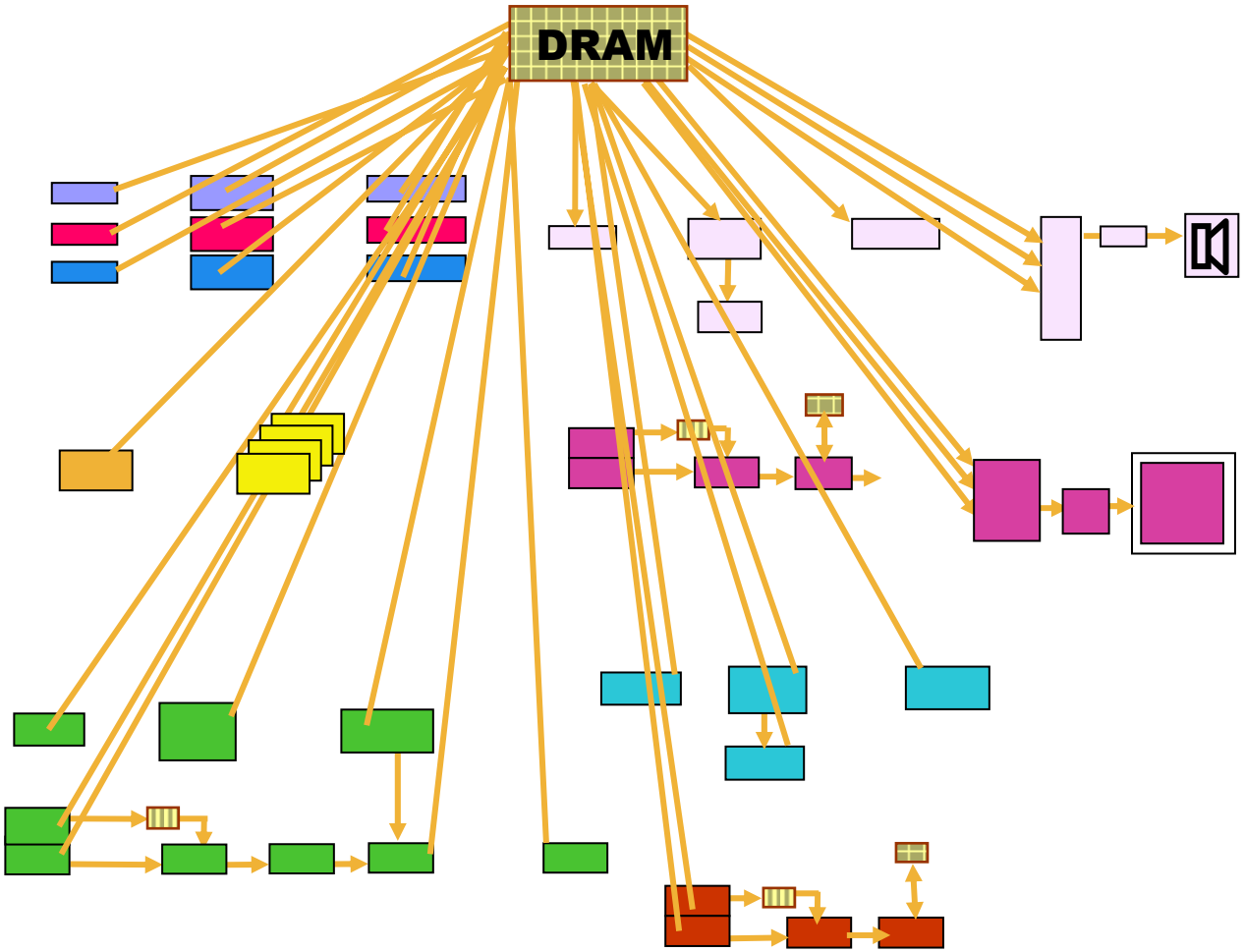
Myth 1: Network-level Concurrency

- Assertion: reference MPSoC apps have >> 50% of traffic to DRAM



Myth 1: Network-level Concurrency

- Assertion: reference MPSoC apps have >> 50% of traffic to DRAM



Myth 1: Network-level Concurrency

- Assertion: reference MPSoC apps have >> 50% of traffic to DRAM
- Implications:
  - Most of network is a fanin tree to a single DRAM
  - Maximizing delivered DRAM efficiency is key

Myth 2: Packetization

- Segmentation/Re-assembly
  - Data network background:
    - Break long payloads into smaller ones to cope with physical link capabilities and minimize intermediate buffering
      - Trades off higher complexity at endpoints for simpler core
  - NoC situation:
    - Physical fabric is under control; should minimize total cost function
    - Segmentation sacrifices DRAM performance
- Payload encapsulation (add headers, etc.)
  - Data network background:
    - Required to add routing information, checksums, etc.
  - NoC situation:
    - Already operates in a strongly-addressed world
    - Should not pay latency overhead for this on chip

Myth 2: Serialization

- Bit-level
  - Data network background:
    - Have no choice but to serialize, parallel physical links are expensive
  - NoC situation:
    - Little benefit in serialization (wires relatively cheap) – costs latency!
- Packet interleaving
  - Data network background:
    - Normally do not interleave except at packet boundaries
  - NoC situation:
    - Most NoC's same as data networks
    - Non-blocking fabrics (aka virtual channels with virtual flit flow control) allow interleaving at link-word boundaries
      - Enables ***multiplexing*** of links in path to DRAM

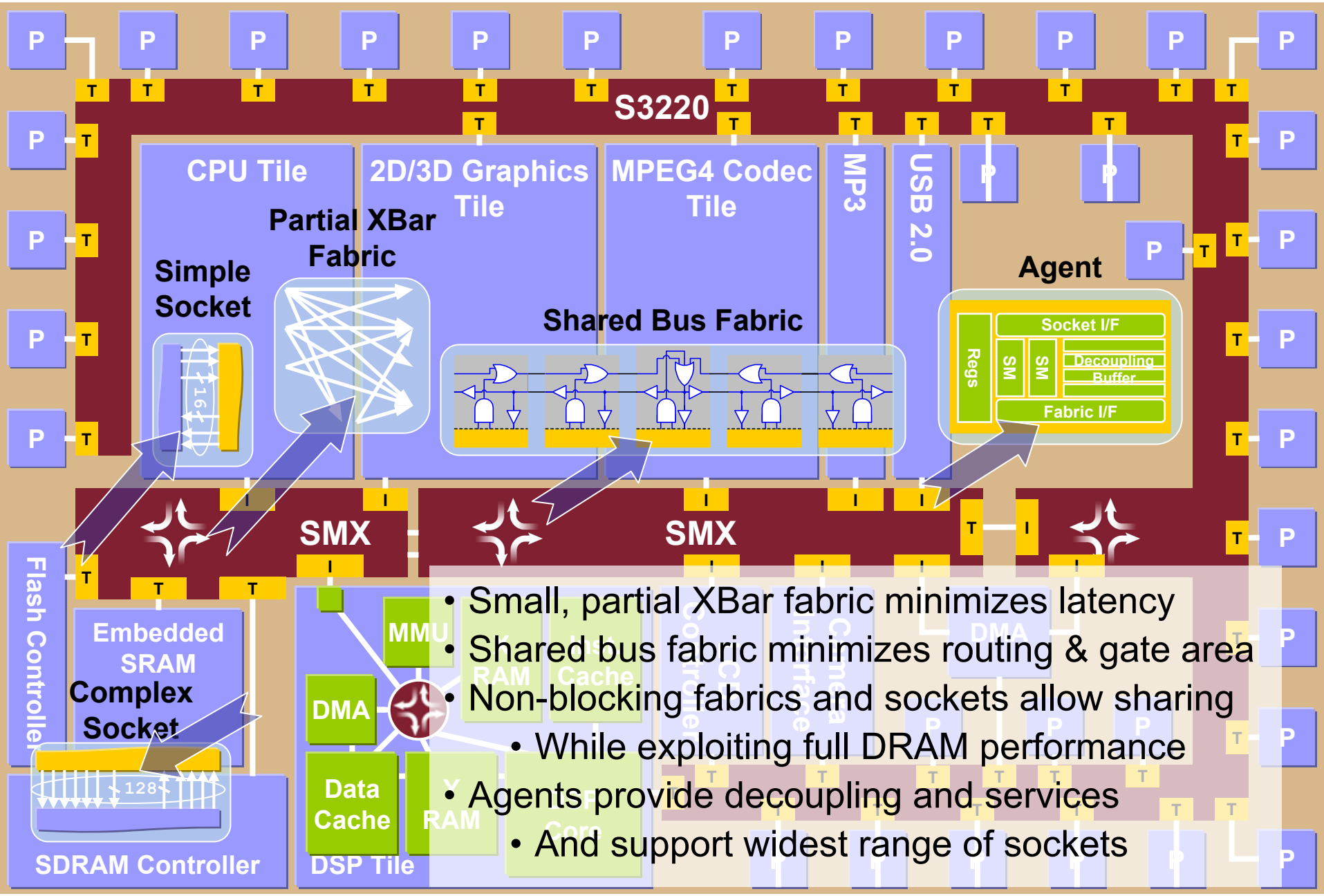
Myth 3: NoC Latency is OK

- Latency-tolerant design approaches not well practiced
- Extra latency in CPU-memory path still expensive (Amdahl's Law)
- Extra latency in pipelined accelerator costs:
  - More outstanding requests to hide latency
  - Larger FIFO's to store pending requests in accelerator
  - More storage in NoC to cover pipeline length
- Maintaining performance with extra latency costs area and power
- Running NoC clock faster than sockets not practical
  - Increases link delay (flip-flop overhead)
  - DRAM freq increases faster than ASIC

NoC Truths

- Isolating fabric protocol from socket is important
  - Fabric typically much more advanced
- Decoupling at edges has many design benefits
  - Independent design and re-use
  - Better physical implementation
  - Structured verification
- Tooling to deliver automated views is helpful
  - Particularly validation
- Centralized network services are important
  - QoS, security, error mgmt., power mgmt., …

What is Practical Today?



What Could Change

- Embedded memory
  - Cost must decrease ***faster*** than data set size increases
  - Shared scratchpad and/or direct peer-peer transfers
- 3D memory
  - Massive increase in BW & more concurrency
  - Looks like more targets
- True latency-tolerant design put to practice
  - Even the inevitable takes time…

Thank you!

Sonics View on NoC

- NoC adapts key abstractions from networking
  - Layering (orthogonalization of concerns)
    - Socket-based design
    - Optimized fabric protocols
    - Higher level services (security, QoS, error mgmt.)
  - Distributed design & implementation
- We call this *decoupling*

Sonics NoC Implementations

Product	SiliconBackplane	Sonics3220	SonicsMX
Introduction	1999	2002	2004
Connects	Processors, memories	Peripherals, registers	Processors, memories
Socket(s)	OCP 1	OCP 1/2, APB	OCP 1/2, AHB, AXI
Decoupling	High	Medium	Very High
Fabric	Distributed, shared bus	Multi-branch shared bus	Hybrid cross-bar / shared bus
Applications	WLAN, HDTV, PVR, …	Handsets, consumer, …	Handsets, HDTV, DSC, OA, …
Volume Production	Over 150 million IC's shipped	2005	2005