Computer Structure

System and DRAM

Lecturer:
Roni Kupershtok

Slides prepared by Lihu Rappoport
6th Generation Intel Core™ – Skylake

- 14nm process
- Quad core die, with Intel HD Graphics 530
**SOC – System On a Chip**

- **Client SOC includes**
  - The System Agent, which handles Memory and I/O
    - PCI Express, Memory Controller, Display Engine, and more
  - The Graphics Unit
  - The Last Level Cache (LLC) Slices
  - A high bandwidth ring bus
    - Connects between the Cores, LLC slices, Graphics, and System Agent

- **Server SOC is different**
  - No Graphics Unit
  - Large capacity LLC with snooping capabilities that support multiple processors
  - Interfaces for supporting multi-socket platforms
  - Support for high BW traffic from memory and I/O
Last Level Cache – LLC

- The LLC consists of multiple cache slices
  - The number of slices is equal to the number of IA cores
  - Each slice can supply 32 bytes/cycle
  - Each slice may have 4/8/12/16 ways
    - Corresponding to 0.5M/1M/1.5M/2M cache size

- Physical addresses are uniformly distributed among cache slices using a hash function
  - The LLC acts as one shared cache
    - With multiple ports and BW that scales with the number of cores
    - Ring/LLC not likely to be a BW limiter
  - Prevents hot spots

- LLC is shared among all Cores, Graphics and Media
  - GFX/media may in compete with cores on LLC
Last Level Cache – LLC (cont.)

- **LLC hit latency is ~30 cycles**
  - Depends on Core location relative to the LLC Slice (how far the request and the data need to travel on the ring)

- **Fetching data from LLC when another core has the data (in E/M states)**
  - Clean hit – data is not modified in the other core – 43 cycles
  - Dirty hit – data is modified in the other core – 60 cycles

- **LLC is fully inclusive of all core internal caches**
  - Eliminates unnecessary snoops to cores
  - Per core “Core Valid bit” indicates if the internal core caches need to be snooped for a given cache line

- **Traffic that cannot be satisfied by the LLC**
  - LLC misses, dirty line write-back, non-cacheable operations, MMIO/IO operations
  - Travels through the ring to the IMC
Data Prefetch to MLC and LLC

- Two HW prefetchers fetch data from memory to MLC and LLC
  - Prefetch the data to the LLC
  - Typically data is brought also to the MLC
    - Unless the MLC is heavily loaded with missing demand requests

- Spatial Prefetcher
  - For every 128B-aligned cache line fetched to the MLC, prefetch the next sequential cache line, to complete a 128B-aligned chunk

- Streamer Prefetcher
  - Monitors read requests from the L1 caches for ascending and descending sequences of addresses
    - L1 D$ requests: loads, stores, and L1 D$ HW prefetcher
    - L1 I$ code fetch requests
  - When a forward or backward stream of requests is detected
    - Prefetch anticipated fwd/bwd sequential cache lines (no stride)
    - Cannot cross 4K page boundary (same physical page)
Data Prefetch to MLC and LLC

- Each MLC demand request may trigger 2 more prefetch requests
  - E.g., assume demand requests to addresses $n$, $n+64$, $n+64 \times 2$, ...
    - Demand to address $n$ triggers a prefetch to $n+64$, $n+64 \times 2$
    - Demand to address $n+64$ triggers a prefetch to $n+64 \times 3$, $n+64 \times 4$
  - The prefetch stream runs up to 20 lines ahead of the demand stream
    - Suspended prefetch stream

- Adjusts dynamically to the number of outstanding requests per core
  - Not many outstanding requests $\Rightarrow$ prefetch further ahead (20 lines)
  - Moderate num. of outstanding requests $\Rightarrow$ prefetch up to 10 lines ahead
  - Many outstanding requests $\Rightarrow$ prefetch to LLC only, and less far ahead

- When cache lines are far ahead
  - Prefetch to LLC only, and not to the MLC
  - Avoids replacement of useful cache lines in the MLC

- Detects and maintains up to 32 streams of data accesses
  - For each 4K byte page, can maintain one forward and one backward stream
Data Prefetch to MLC and LLC

Various condition \(\Rightarrow\) prefetch to LLC only
- Prefetch stream runs much ahead of the demand stream
- Many outstanding requests

In other conditions, stop prefetch also to LLC
Scalable Ring Interconnect

- **High bandwidth scalable ring bus**
  - Interconnects Cores, Graphics, LLC and SA
  - Composed of 4 bidirectional 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
  - Ring wires run over LLC without area impact
  - Distributed arbitration, coherency and ordering

- **Each Core/LLC can get data from two stations**
  - One connected to the “down“ direction,
  - and one connected to the “up”

- **Ring access always picks the shortest path**
  - E.g., Core3 to LLC2 data uses the “up” stream in 1 hop
    - Rather than from the "down" stream in 7 hops
  - Ave. trip from Core to LLC is \((0+1+2+3)/4 = 1.5\) hops
Ring Traffic Control

• Each stop can pull one request off the ring per clock
  – Either from the "up" or from the "down"

• To avoid a case of data arriving to a given stop from both "up" and "down" directions at the same time
  – Each ring stop has a given polarity ("even" or "odd")
    ➢ A ring stop can only pull data from the direction that matches its polarity
  – The ring flips polarity every clock
  – The sender knows the receiver and its polarity, and what is the hop count between them
  – By delaying sending the data for at most one cycle, a sender can assure that the receiver can read it
    ➢ and the case of two packets arriving at the same time never occurs
The GO and the data may be sent at different cycles: the GO is the outcome of a tag-hit, while the data comes from the data array.

LLC sends the second chunk to each one of the cores. The GO carries the MESI state: E/S.

Both Core's requests get a hit in the LLC, and the CVBs (core valid bits) indicate that no snoop is needed.
System Agent Components

- PCIe controllers that connect to external PCIe devices
  - Support different configurations: \( \times 16 + \times 4, \times 8 + \times 8 + \times 4, \times 8 + \times 4 + \times 4 + \times 4 \)

- DMI (Direct Media Interface) controller
  - Connects to the PCH (Platform Controller Hub)

- Integrated display engine
  - Handles delivering the pixels to the screen

- Flexible Display Interface (FDI)
  - Connects to the PCH, where the display connectors (HDMI, DVI) are attached

- DisplayPort (used for integrated display)
  - e.g., a laptop’s LCD

- Integrated Memory Controller (IMC)
  - Connects to and controls the DRAM

- An arbiter that handles accesses from Ring and from I/O (PCIe & DMI)
  - Routes the accesses to the right place
  - Routes main memory traffic to the IMC
DRAM
DRAM Operation

- DRAM cell consists of transistor + capacitor
  - Capacitor keeps the state; Transistor guards access to the state
  - Reading cell state: raise access line AL and sense DL
    - Capacitor charged $\Rightarrow$ current to flow on the data line DL
  - Writing cell state: set DL and raise AL to charge/drain capacitor
  - Charging and draining a capacitor is not instantaneous
- Leakage current drains the capacitor even when transistor is closed
  - The DRAM cell must be periodically refreshed (every 64ms)
• **DRAM access sequence**
  - Put Row on addr. bus
  - Assert RAS# (Row Addr. Strobe) to latch Row
  - Put Column on addr. bus
  - Wait RAS# to CAS# delay and assert CAS# (Column Addr. Strobe) to latch Col
  - Get data on address bus after CL (CAS latency)
Page Mode DRAM

- Allows Multiple accesses to different columns within the same row
  - Saves RAS, RAS to CAS delay, and Row pre-charge

- CAS-to-CAS delay: delay between two CASs within the same row
- tRP: Row pre-charge time:
  the time to close current row, and open a new row

- DRAM is not true random access
  - Accessing multiple columns within the same row is much faster than accessing addresses from different rows
DIMMs

- **DIMM**: Dual In-line Memory Module
  - A small circuit board that holds memory chips

- **64-bit wide data path (72 bit with parity)**
  - Single sided: 9 chips, each with 8 bit data bus
  - Dual sided: 18 chips, each with 4 bit data bus
  - Data BW: 64 bits on each rising and falling edge of the clock

- **Other pins**
  - Address – 14, RAS, CAS, chip select – 4, VDC – 17, Gnd – 18, clock – 4, serial address – 3, …
Synchronous DRAM – SDRAM

- All signals are referenced to an external clock (100MHz-200MHz)
  - Makes timing more precise with other system devices
- 4 banks – multiple pages (rows) open simultaneously (one per bank)
  - Accessing columns within each one of the 4 open rows is fast
- Command driven functionality instead of signal driven
  - ACTIVE: selects both the bank and the row to be activated
    - ACTIVE to a new bank can be issued while accessing current bank
  - READ/WRITE: select column
- Burst oriented read and write accesses
  - Successive column locations accessed in the given row
  - Burst length is programmable: 1, 2, 4, 8, full-page (may end by burst terminate)
- A user programmable Mode Register
  - CAS latency, burst length, burst type
- Auto pre-charge: may close row at last read/write in burst
- Auto refresh: internal counters generate refresh address
SDRAM Timing

- $t_{RCD}$: ACTIVE to READ/WRITE gap = $\left\lceil \frac{t_{RCD}(\text{MIN})}{\text{clock period}} \right\rceil$
- $t_{RC}$: successive ACTIVE to a different row in the same bank
- $t_{RRD}$: successive ACTIVE commands to different banks
**DDR-SDRAM**

- **2n-prefetch architecture**
  - DRAM cells are clocked at the same speed as SDR SDRAM cells
  - Internal data bus is twice the width of the external data bus
  - Data capture occurs twice per clock cycle
    - Lower half of the bus sampled at clock rise
    - Upper half of the bus sampled at clock fall

- **Uses 2.5V (vs. 3.3V in SDRAM)**
  - Reduced power consumption
DDR SDRAM Timing

200MHz clock

cmd

- ACT
- NOP
- NOP
- RD
- NOP
- ACT
- NOP
- NOP
- RD
- NOP
- ACT
- NOP
- NOP

- \( t_{RCD} > 20\text{ns} \)
- \( t_{RRD} > 20\text{ns} \)
- \( t_{RC} > 70\text{ns} \)

Bank

- Bank 0
- X
- X
- Bank 0
- X
- Bank 1
- X
- X
- Bank 1
- X
- Bank 0
- X
- X

Addr

- Row \( i \)
- X
- X
- Col \( j \)
- X
- Row \( m \)
- X
- X
- Col \( n \)
- X
- Row \( l \)
- X
- X

Data

- \( j \)
- +1
- +2
- +3
- \( n \)
- +1
- +2
- +3

- CL=2
**DDR2**

- DDR2 doubles the bandwidth
  - 4n pre-fetch: internally read/write 4× the amount of data as the external bus
  - DDR2-533 cell works at the same freq. as a DDR266 cell or a PC133 cell
  - Prefetching increases latency

- Smaller page size: 1KB vs. 2KB
  - Reduces activation power – ACTIVATE command reads all bits in the page

- 8 banks in 1Gb densities and above
  - Increases random accesses

- 1.8V (vs 2.5V) operation voltage
  - Significantly lower power
**DDR3**

- 30% power consumption reduction compared to DDR2
  - 1.5V supply voltage (vs. 1.8V in DDR2)
  - 90 nanometer fabrication technology

- **Higher bandwidth**
  - 8 bit deep prefetch buffer (vs. 4 bit in DDR2)

- 2× transfer data rate vs DDR2
  - Effective clock rate of 800–1600 MHz
    - using both rising and falling edges of a 400–800 MHz I/O clock
    - DDR2: 400–800 MHz using a 200–400 MHz I/O clock

- **DDR3 DIMMs**
  - 240 pins, the same number as DDR2, and are the same size
  - Electrically incompatible, and have a different key notch location
DDR4 – 4th Gen. DDR SDRAM

- Released to the market in 2014
  - Higher module density: up to 64GB DIMMs vs. 16GB in DDR3
  - Lower voltage requirements: 1.2V and 1.4V vs. 1.5V and 1.65V in DDR3
  - Higher data rate transfer speeds

- Prefetch has not been increased above the 8n used in DDR3
  - The basic burst size is eight words
  - Higher bandwidth achieved by more read/write commands per sec
    - To allow this, DRAM banks are divided into two or four selectable bank groups, where transfers to different bank groups may be done more rapidly

- The reduced power allows higher operation speeds
  - Without going to expensive power and cooling requirements
  - Power consumption increases with speed

- Frequency: 800 to 2133 MHz (DDR4-1600 through DDR4-4167)
  - compared 400 to 1067 MHz in DDR3

- Speeds are typically advertised as doubles of these numbers
  - DDR3-1600 and DDR4-2400 are common
  - DDR3-3200 and DDR4-4800 available at high cost
**DDR4 Standards**

- **Standard name vs. module name**
  - DDR4-xxxx denotes per-bit data transfer rate in MT/sec
  - PC4-xxxxx denotes overall module (DIMM) transfer rate in MByte/sec
    - DDR4 modules transfer data on a 8-byte data bus
      ⇒ module peak transfer rate in MByte/sec = transfer rate × 8

- **Memory clock frequency, I/O bus clock frequency, Data rate**
  - 8 wide prefetch ⇒ Data rate = Memory clock freq × 8
  - Data transferred on both clock edges ⇒ Data rate = I/O bus clock freq × 2

- **DRAM timing, measure in I/O bus cycles, specifies 3 numbers**
  - CAS Latency – RAS-to-CAS Delay – RAS Pre-charge Time

- **CAS latency (latency to get data in an open page) in nsec**
  - CAS Latency × I/O bus cycle time in the example: \[\frac{11}{(800 \times 10^6)} \times 10^9 = 13.75\]

<table>
<thead>
<tr>
<th>Standard name</th>
<th>Memory clock (MHz)</th>
<th>I/O bus clock (MHz)</th>
<th>Data rate (MT/s)</th>
<th>Module name</th>
<th>Peak transfer rate (MB/s)</th>
<th>Timings CL-tRCD-tRP</th>
<th>CAS latency (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDR4-1600K</td>
<td>200</td>
<td>800</td>
<td>1600</td>
<td>PC4-12800</td>
<td>12800</td>
<td>11-11-11</td>
<td>13.75</td>
</tr>
</tbody>
</table>
# DDR4 Standards

<table>
<thead>
<tr>
<th>Standard name</th>
<th>Memory clock (MHz)</th>
<th>I/O bus clock (MHz)</th>
<th>Data rate (MT/s)</th>
<th>Module name</th>
<th>Peak transfer rate (MB/s)</th>
<th>Timings</th>
<th>CAS latency (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDR4-1600J*</td>
<td>200</td>
<td>800</td>
<td>1600</td>
<td>PC4-12800</td>
<td>12800</td>
<td>10-10-10</td>
<td>12.5</td>
</tr>
<tr>
<td>DDR4-1600K</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>11-11-11</td>
<td>13.75</td>
</tr>
<tr>
<td>DDR4-1600L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>12-12-12</td>
<td>15</td>
</tr>
<tr>
<td>DDR4-1866L*</td>
<td>233.33</td>
<td>933.33</td>
<td>1866.67</td>
<td>PC4-14900</td>
<td>14933.33</td>
<td>12-12-12</td>
<td>12.857</td>
</tr>
<tr>
<td>DDR4-1866M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13-13-13</td>
<td>13.929</td>
</tr>
<tr>
<td>DDR4-1866N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14-14-14</td>
<td>15</td>
</tr>
<tr>
<td>DDR4-2133N*</td>
<td>266.67</td>
<td>1066.67</td>
<td>2133.33</td>
<td>PC4-17000</td>
<td>17066.67</td>
<td>14-14-14</td>
<td>13.125</td>
</tr>
<tr>
<td>DDR4-2133R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>16-16-16</td>
<td>15</td>
</tr>
<tr>
<td>DDR4-2400P*</td>
<td>300</td>
<td>1200</td>
<td>2400</td>
<td>PC4-19200</td>
<td>19200</td>
<td>15-15-15</td>
<td>12.5</td>
</tr>
<tr>
<td>DDR4-2400R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>16-16-16</td>
<td>13.32</td>
</tr>
<tr>
<td>DDR4-2400T</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>17-17-17</td>
<td>14.16</td>
</tr>
<tr>
<td>DDR4-2400U</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>18-18-18</td>
<td>15</td>
</tr>
<tr>
<td>DDR4-2666T</td>
<td>325</td>
<td>1333</td>
<td>2666</td>
<td>PC4-21333</td>
<td>21333</td>
<td>17-17-17</td>
<td>12.75</td>
</tr>
<tr>
<td>DDR4-2666U</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>18-18-18</td>
<td>13.50</td>
</tr>
<tr>
<td>DDR4-2666V</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>19-19-19</td>
<td>14.25</td>
</tr>
<tr>
<td>DDR4-2666W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>20-20-20</td>
<td>15</td>
</tr>
<tr>
<td>DDR4-2933V</td>
<td>366.6</td>
<td>1466.5</td>
<td>2933</td>
<td>PC4-23466</td>
<td>23466</td>
<td>19-19-19</td>
<td>12.96</td>
</tr>
<tr>
<td>DDR4-2933W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>20-20-20</td>
<td>13.64</td>
</tr>
<tr>
<td>DDR4-2933Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>21-21-21</td>
<td>14.32</td>
</tr>
<tr>
<td>DDR4-2933AA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>22-22-22</td>
<td>15</td>
</tr>
<tr>
<td>DDR4-3200W</td>
<td>400</td>
<td>1600</td>
<td>3200</td>
<td>PC4-25600</td>
<td>25600</td>
<td>20-20-20</td>
<td>12.50</td>
</tr>
<tr>
<td>DDR4-3200AA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>22-22-22</td>
<td>13.75</td>
</tr>
<tr>
<td>DDR4-3200AC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>24-24-24</td>
<td>15</td>
</tr>
</tbody>
</table>
DRAM Access Latency

1. Delay in LLC request buffer
2. Request sent from LLC to IMC
3. IMC delay
4. IMC sends command to DRAM
5. Row pre-charge
6. RAS-to-CAS
7. CAS latency
8. Data returns to IMC
9. IMC delay
10. Data sent from IMC to LLC
11. Delay in LLC fill buffer
DRAM Access Latency

- Assume CPU clock cycle of 4GHz = 250ps = 0.25ns

- Single access to a new page
  - In the DRAM
    - Row pre-charge 13ns
    - RAS-to-CAS 13ns
    - CAS latency 13ns
  - Rest of the system 25ns
  - Total 64ns = 256 CPU clock cycles

- Single access to an open page
  - In the DRAM
    - CAS-to-CAS 6ns
    - CAS latency 13ns
  - Rest of the system 25ns
  - Total 44ns = 176 CPU clock cycles
The Memory Controller
The Memory Controller

- A typical memory controller supports two channels of DDR3

- Each channel has its own resources
  - Handles memory requests independently
  - Contains a 32 cache-line write-data-buffer
  - Supports 8 bytes per cycle

- A hash function distributes addresses between channels
  - Attempts to balance the load between the channels in order to achieve maximum bandwidth and minimum hotspot collisions

- An out-of-order scheduler maximizes BW and minimize latency
  - Writes to the memory controller are considered completed when they are written to the write-data-buffer
  - The write-data-buffer is flushed out to main memory at a later time, not impacting write latency
Partial Memory Writes

- Memory reads and writes are done at full Cache Line granularity

- **Partial Write** transactions are writes to a subset of a Cache Line
  - Non-cacheable writes, e.g., write requests driven by IO devices
  - Would require adding more pins to indicate to memory which bytes to write
  - Not supported with ECC

- Partial writes are not common enough to justify the extra cost
  - The MC supports a partial write, by a RMW operation
    - Reads the current value of the full CL
    - Replaces the bytes that have to be modified
    - Writes back the full CL to memory

- Software should avoid creating partial write transactions whenever possible
  - E.g., buffer the partial writes into full cache line writes
How to get the most of Memory?

• **For best performance**
  – Populate both channels with equal amounts of memory
    ➢ Preferably the exact same types of DIMMs
  – Use highest supported speed DRAM, with the best DRAM timings

• **Each DIMM supports 4 open pages simultaneously**
  – The more open pages, the more random access
  – It is better to have more DIMMs: \( n \) DIMMs \( \Rightarrow 4n \) open pages
  – Dual sided DIMMs *may* have separate CS of each side
    ➢ Support 8 open pages
    ➢ Dual sided DIMMs may also have a common CS
Motherboard Layout (Sandy Bridge)

IEEE-1394a header
PCI add-in card connector
PCI express x1 connector
PCI express x16 connector
Back panel connectors

audio header

High Def. Audio header
S/PDIF

PCH

Front panel USB headers

Bios setup config jumper

Reset, power, Disk LED
SATA connectors

speaker
Battery

Main Power connector

Processor core power connector

Rear chassis fan header
LGA775 processor socket

Processor fan header
DIMM Channel A sockets
DIMM Channel B sockets
Front chassis fan header
Chassis intrusion header

Serial port header
Rear chassis fan header
Backup
- Put row address on address bus and assert RAS#
- Wait for RAS# to CAS# delay (tRCD) between asserting RAS and CAS
- Put column address on address bus and assert CAS#
- Wait for CAS latency (CL) between time CAS# asserted and data ready
- Row pre-charge time: time to close current row, and open a new row
DRAM controller

- DRAM controller gets address and command
  - Splits address to Row and Column
  - Generates DRAM control signals at the proper timing

- DRAM data must be periodically refreshed
  - DRAM controller performs DRAM refresh, using refresh counter
The Memory Controller

- The memory controller supports high-priority isochronous requests
  - E.g., USB isochronous, and Display isochronous requests
  - High bandwidth of memory requests from the integrated display engine takes up some of the memory bandwidth
  - Impacts core access latency to some degree
### SRAM vs. DRAM

- Random Access: access time is the same for all locations

<table>
<thead>
<tr>
<th></th>
<th>DRAM — Dynamic RAM</th>
<th>SRAM — Static RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Refresh</strong></td>
<td>Refresh needed</td>
<td>No refresh needed</td>
</tr>
<tr>
<td><strong>Address</strong></td>
<td>Address muxed: row+ column</td>
<td>Address not multiplexed</td>
</tr>
<tr>
<td><strong>Access</strong></td>
<td>Not true “Random Access”</td>
<td>True “Random Access”</td>
</tr>
<tr>
<td><strong>Density</strong></td>
<td>High (1 Transistor/bit)</td>
<td>Low (6 Transistor/bit)</td>
</tr>
<tr>
<td><strong>Power</strong></td>
<td>low</td>
<td>high</td>
</tr>
<tr>
<td><strong>Speed</strong></td>
<td>slow</td>
<td>fast</td>
</tr>
<tr>
<td><strong>Price/bit</strong></td>
<td>low</td>
<td>high</td>
</tr>
<tr>
<td><strong>Typical usage</strong></td>
<td>Main memory</td>
<td>cache</td>
</tr>
</tbody>
</table>
## DDR4 vs DDR3

<table>
<thead>
<tr>
<th>Feature/Option</th>
<th>DDR3</th>
<th>DDR4</th>
<th>DDR4 Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voltage (core and I/O)</td>
<td>1.5V</td>
<td>1.2V</td>
<td>Reduces memory power demand</td>
</tr>
<tr>
<td>$V_{REF}$ inputs</td>
<td>2 – DQs and CMD/ADDR</td>
<td>1 – CMD/ADDR</td>
<td>$V_{REFDQ}$ now internal</td>
</tr>
<tr>
<td>Low voltage standard</td>
<td>Yes (DDR3L at 1.35V)</td>
<td>No</td>
<td>Memory power reductions</td>
</tr>
<tr>
<td>Data rate (Mb/s)</td>
<td>800, 1066, 1333, 1600, 1866, 2133</td>
<td>1600, 1866, 2133, 2400, 2666, 3200</td>
<td>Migration to higher-speed I/O</td>
</tr>
<tr>
<td>Densities</td>
<td>512Mb–8Gb</td>
<td>2Gb–16Gb</td>
<td>Better enablement for large-capacity memory systems</td>
</tr>
<tr>
<td>Internal banks</td>
<td>8</td>
<td>16</td>
<td>More banks</td>
</tr>
<tr>
<td>Bank groups (BG)</td>
<td>0</td>
<td>4</td>
<td>Faster burst accesses</td>
</tr>
<tr>
<td>$t_{CK}$ – DLL enabled</td>
<td>300 MHz to 800 MHz</td>
<td>667 MHz to 1.6 GHz</td>
<td>Higher data rates</td>
</tr>
<tr>
<td>$t_{CK}$ – DLL disabled</td>
<td>10 MHz to 125 MHz (optional)</td>
<td>Undefined to 125 MHz</td>
<td>DLL-off now fully supported</td>
</tr>
<tr>
<td>Read latency</td>
<td>$AL + CL$</td>
<td>$AL + CL$</td>
<td>Expanded values</td>
</tr>
<tr>
<td>Write latency</td>
<td>$AL + CWL$</td>
<td>$AL + CWL$</td>
<td>Expanded values</td>
</tr>
<tr>
<td>DQ driver (ALT)</td>
<td>40Ω</td>
<td>48Ω</td>
<td>Optimized for PtP (point-to-point) applications</td>
</tr>
<tr>
<td>DQ bus</td>
<td>SSTL15</td>
<td>POD12</td>
<td>Mitigate I/O noise and power</td>
</tr>
<tr>
<td>$R_{TT}$ values (in Ω)</td>
<td>120, 60, 40, 30, 20</td>
<td>240, 120, 80, 60, 48, 40, 34</td>
<td>Support higher data rates</td>
</tr>
<tr>
<td>$R_{TT}$ not allowed</td>
<td>READ bursts</td>
<td>Enables during READ bursts</td>
<td>Ease-of-use</td>
</tr>
<tr>
<td>ODT modes</td>
<td>Nominal, dynamic</td>
<td>Nominal, dynamic, park</td>
<td>Additional control mode; supports OTF value change</td>
</tr>
<tr>
<td>ODT control</td>
<td>ODT signaling required</td>
<td>ODT signaling not required</td>
<td>Ease of ODT control, allows non-ODT routing on PtP applications</td>
</tr>
<tr>
<td>Multipurpose register (MPR)</td>
<td>Four registers – 1 defined, 3 RFU</td>
<td>Four registers – 3 defined, 1 RFU</td>
<td>Provides additional specialty readout</td>
</tr>
</tbody>
</table>

**Note:** DDR4 Advantage is a summary of benefits over DDR3, focusing on key performance and efficiency improvements.
Server Mesh Architecture

- Interconnects between CPU cores, memory hierarchy, and I/O
  - Increase bandwidth, reduce latency, reduce power
## DDR Comparison

<table>
<thead>
<tr>
<th></th>
<th>DDR</th>
<th>DDR2</th>
<th>DDR3</th>
<th>DDR4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefetch depth</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Memory Clock (MHz)</td>
<td>100-200</td>
<td>100-200</td>
<td>100-266(\frac{2}{3})</td>
<td>200-400</td>
</tr>
<tr>
<td>I/O bus clock (MHz)</td>
<td>100-200</td>
<td>200-400</td>
<td>400-1066(\frac{2}{3})</td>
<td>800-1600</td>
</tr>
<tr>
<td>Data rate (MT/s)</td>
<td>200-400</td>
<td>400-800</td>
<td>800-2133</td>
<td>1600-3200</td>
</tr>
<tr>
<td>Module rate (GB/s)</td>
<td>1.6-3.2</td>
<td>3.2-6.4</td>
<td>6.4-17.1</td>
<td>12.8-25.6</td>
</tr>
<tr>
<td>CAS latency (ns)</td>
<td>9.4-12.5</td>
<td>11.2-15</td>
<td>11-15</td>
<td>12.5-15</td>
</tr>
<tr>
<td>DRAM timing (lowest)</td>
<td>2.5-3-3</td>
<td>3-3-3</td>
<td>5-5-5</td>
<td>10-10-10</td>
</tr>
<tr>
<td>Voltage (standard/low)</td>
<td>2.5V / 1.8V</td>
<td>1.8V</td>
<td>1.5V / 1.35V</td>
<td>1.2V / 1.05V</td>
</tr>
<tr>
<td>Power (mW)</td>
<td>399</td>
<td>217</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max DIMM size</td>
<td>1GB</td>
<td>4GB</td>
<td>16GB</td>
<td>64GB</td>
</tr>
<tr>
<td>Internal banks</td>
<td>4</td>
<td>4 / 8</td>
<td>8</td>
<td>16</td>
</tr>
<tr>
<td>Banks Groups</td>
<td>n.a.</td>
<td>n.a.</td>
<td>n.a.</td>
<td>4</td>
</tr>
<tr>
<td>Year released</td>
<td>2000</td>
<td>2003</td>
<td>2007</td>
<td>2014</td>
</tr>
</tbody>
</table>