

온라인 게임 서버 환경

정내훈

2024년도 1학기

한국공학대학교 게임공학과

내용

- 게임서버를 위한 하드웨어와 소프트웨어

MMO 게임서버 특징

- 고사양을 필요로 한다.
 - 저사양 PC보다 고 사양 PC로 더 많은 동접을 받을 수 있다.
 - 가성비를 따져서 저 사양의 PC여러 대로 많은 동접을 올린다. (X)
- ⇒ 가능한 한 최고 사양의 PC를 사용해서 동접을 올린다.
- ⇒ 동접이 올라간다 => 게임이 재미있어 진다 => 많은 사용자가 몰려온다 => 게임이 재미있어 진다 => 많은 사용자가 몰려온다 => 많은 수익을 올린다.

MMO 게임서버 특징

- 고사양 PC (서버 PC)
 - CPU : Single Core로는 동접 1000도 어렵다.
 - 멀티 코어 사용 => 멀티쓰레딩 필요
 - 멀티 CPU 사용
 - Memory : 128GB이상 필요
 - World 정보, NPC/Monster 정보, 플레이어/아이템...
 - Network : 복수개의 Network Port 필요
 - 보조 기억장치 : 상관없음. (상관 없어야 함)

게임서버 하드웨어

- HW만 좋으면 되는가?
 - HW를 100%활용할 수 있는 프로그래밍이 필요
 - 특히 멀티 Core/멀티 CPU
 - 그리고, 네트워크 프로그래밍
- 하드웨어 지식이 필요한 이유
 - 서버에 걸리는 부하가 많기 때문에 하드웨어의 성능을 최대한 끌어내는 프로그래밍이 필요하다.
 - 하드웨어의 성능을 최대한 끌어내려면 하드웨어가 어떻게 성능에 영향을 미치는지 알아야 한다.
 - 작은 회사에서는 서버 프로그래머가 서버 머신 구입을 결정

게임서버 하드웨어

- 중요 하드웨어
 - CPU
 - Memory
 - Network

게임서버 하드웨어

● CPU

- X86 계열이 대세 이다. (~~인텔의 Itanium~~, IBM의 PowerPC 계열도 존재) ARM은 사용불가.
- AMD와 Intel의 2가지 계열이 있다.
 - 서버용 CPU가 따로 존재한다.
 - 예) Intel Xeon, AMD EPYC
- 64bit와 Multi-core
- 코어개수, 클럭 속도와 Cache 크기, 메모리 Bus대역폭이 중요하다.

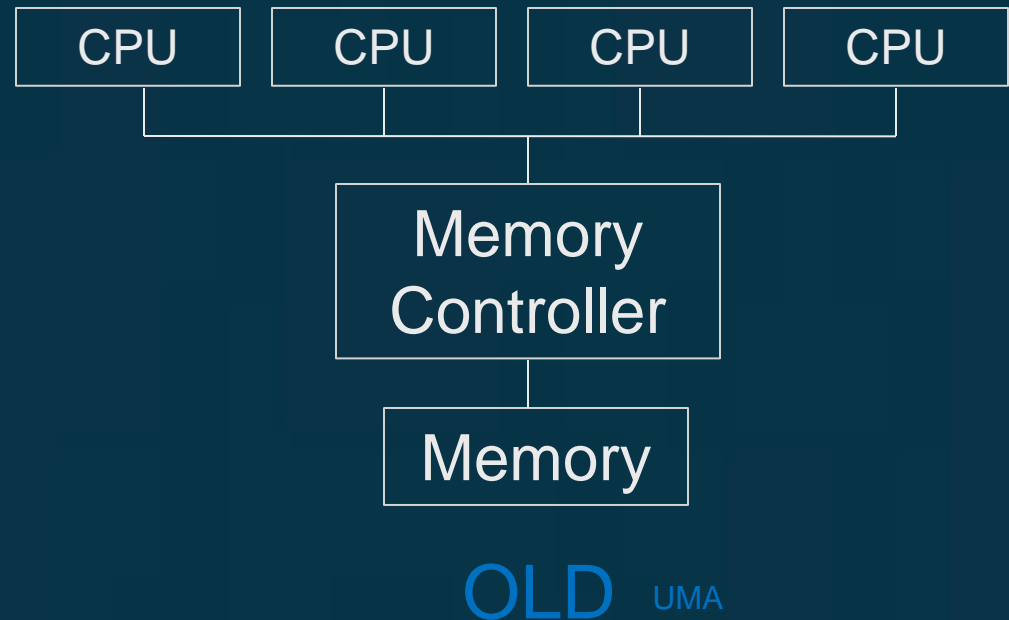
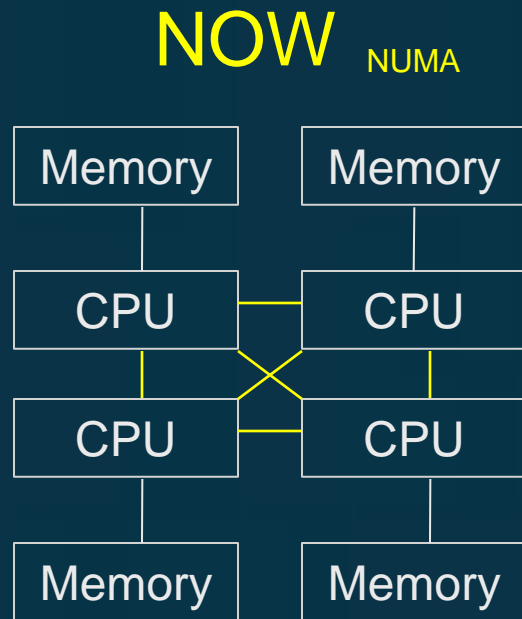
게임서버 하드웨어

● CPU

- 서버용 CPU와 일반 Desktop용 CPU의 차이
 - 서버용 CPU는 Multi Processor를 지원한다.
 - Cache 동기화 지원
 - 서버용 CPU는 서로간의 데이터 전송 전용 통로를 제공한다.
 - AMD : HyperTransport (2001~) – 51GB/s, Infinity Fabric(2016~) – 30 ~ 512GB/s
 - 인텔 : QPI(Quick Path Interconnect) (2009~) – 26GB/s, Intel UltraPath Interconnect (2017~) – 6TB/s
 - 메모리 채널 증가 : 8개 까지

게임서버 하드웨어

● CPU



CPU의 개수가 많아질 수록
과거방식에서는 성능에 한계

게임서버 하드웨어(CPU)

- 64비트 이슈

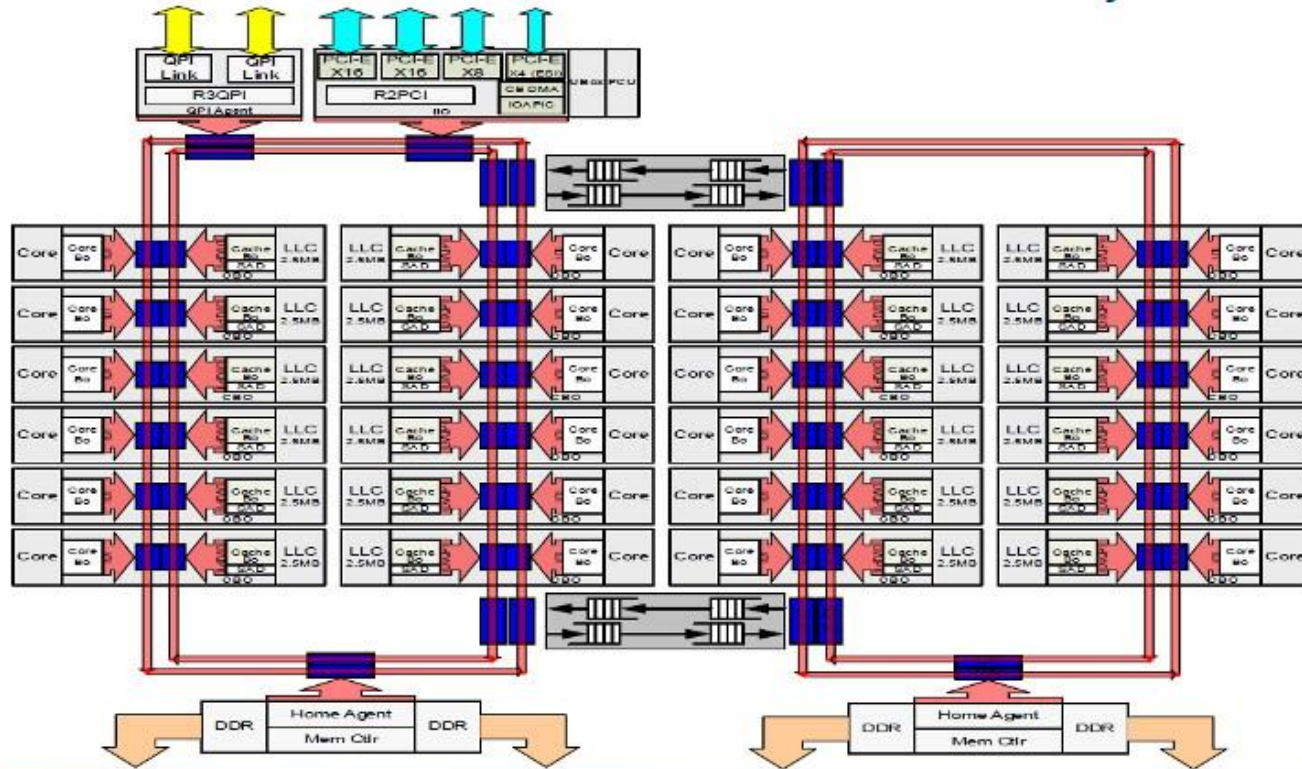
- 64 비트를 사용한다.
 - OS와 Compiler도 64비트 버전을 사용해야 한다.
- 기존의 32비트 CPU로는 최대 4GB의 메모리 밖에 쓸 수 없었기 때문에 서버 용량에 제한이 많았었다.
- 64 비트의 경우 16ExaByte의 메모리가 가능하다.
- 프로그래밍 시 int type과 Pointer type의 크기가 달라지는 것을 주의 해야 한다.
- Linux만 long이 64비트가 된다.

게임서버 하드웨어(CPU)

- Multi-processor (Multi-CPU)
 - SMP(Symmetric Multi Processing)
 - 빠른 네트워크 응답속도와 처리 속도 개선을 위해 Multi-processor을 사용
- Multi-core
 - 발열에 막힌 CPU의 성능향상 제한을 극복하기 위한 공여지책
 - 기존의 4 ~ 8개의 CPU를 활용하던 프로그램 방식에서 8 ~ 64개의 Core 를 활용하도록 변경필요. 앞으로도 계속 core 개수가 늘어날 예정
- Multi-Processor와 Multi-Core와의 차이점
 - SW적으로는 차이가 없음
 - HW적으로는 메모리 접근시 성능 차이가 존재
 - CPU의 개수 만큼 메모리 대역폭 증가
 - NUMA(Non Uniform Memory Access) 문제

게임서버 하드웨어(CPU)

Intel® Xeon® Processor E5 v4 Product Family HCC

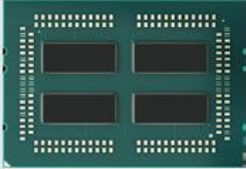
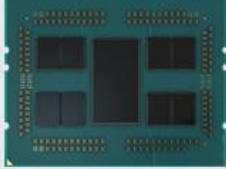
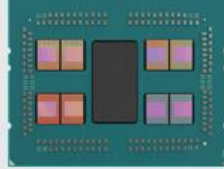



서버 CPU

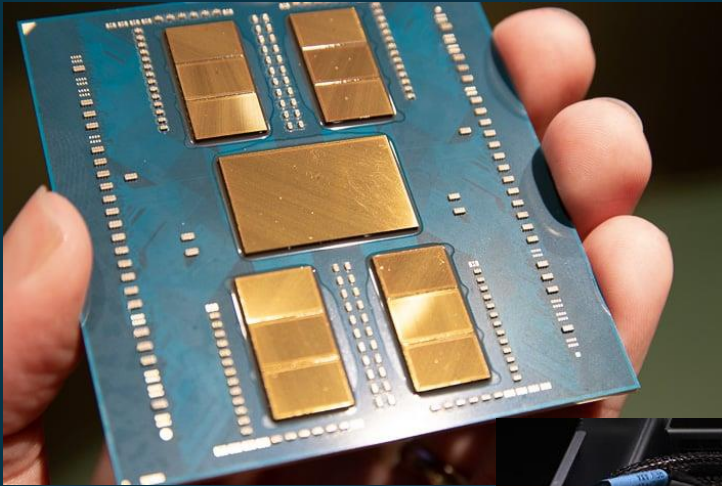
게임서버 하드웨어(CPU)

Product Name	Launch Date	# of Cores	Max Turbo Frequency	Processor Base Frequency	Cache	TDP
<input type="checkbox"/> Intel® Xeon® Platinum 8571N Processor (300M Cache, 2.40 GHz)	Q4'23	52	4 GHz	2.4 GHz	300 MB	300 W
<input type="checkbox"/> Intel® Xeon® Platinum 8568Y+ Processor (300M Cache, 2.30 GHz)	Q4'23	48	4 GHz	2.3 GHz	300 MB	350 W
<input type="checkbox"/> Intel® Xeon® Platinum 8580 Processor (300M Cache, 2.00 GHz)	Q4'23	60	4 GHz	2 GHz	300 MB	350 W
<input type="checkbox"/> Intel® Xeon® Platinum 8593Q Processor (320M Cache, 2.20 GHz)	Q4'23	64	3.9 GHz	2.2 GHz	320 MB	385 W
<input type="checkbox"/> Intel® Xeon® Platinum 8558 Processor (260M Cache, 2.10 GHz)	Q4'23	48	4 GHz	2.1 GHz	260 MB	330 W
<input type="checkbox"/> Intel® Xeon® Platinum 8558P Processor (260M Cache, 2.70 GHz)	Q4'23	48	4 GHz	2.7 GHz	260 MB	350 W
<input type="checkbox"/> Intel® Xeon® Platinum 8592+ Processor (320M Cache, 1.90 GHz)	Q4'23	64	3.9 GHz	1.9 GHz	320 MB	350 W
<input type="checkbox"/> Intel® Xeon® Platinum 8570 Processor (300M Cache, 2.10 GHz)	Q4'23	56	4 GHz	2.1 GHz	300 MB	350 W
<input type="checkbox"/> Intel® Xeon® Platinum 8558U Processor (260M Cache, 2.00 GHz)	Q4'23	48	4 GHz	2 GHz	260 MB	300 W
<input type="checkbox"/> Intel® Xeon® Platinum 8581V Processor (300M Cache, 2.00 GHz)	Q4'23	60	3.9 GHz	2 GHz	300 MB	270 W
<input type="checkbox"/> Intel® Xeon® Platinum 8592V Processor (320M Cache, 2.00 GHz)	Q4'23	64	3.9 GHz	2 GHz	320 MB	330 W

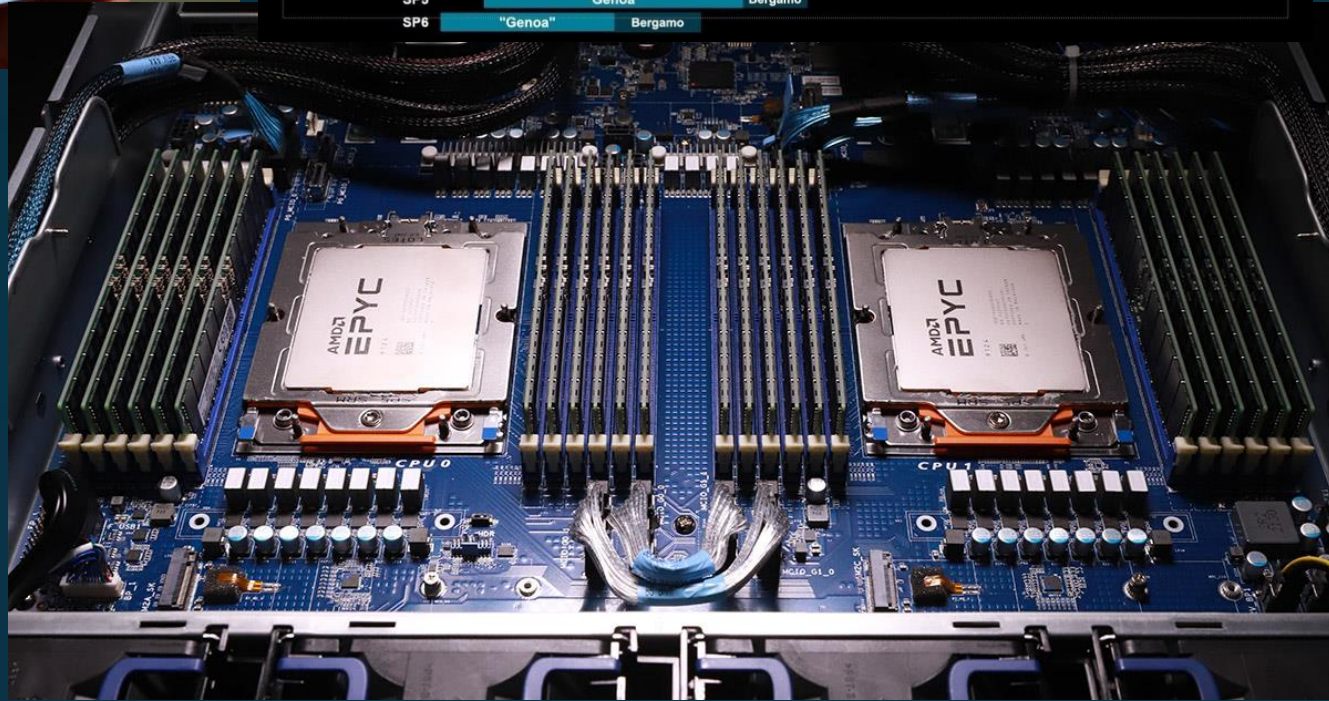
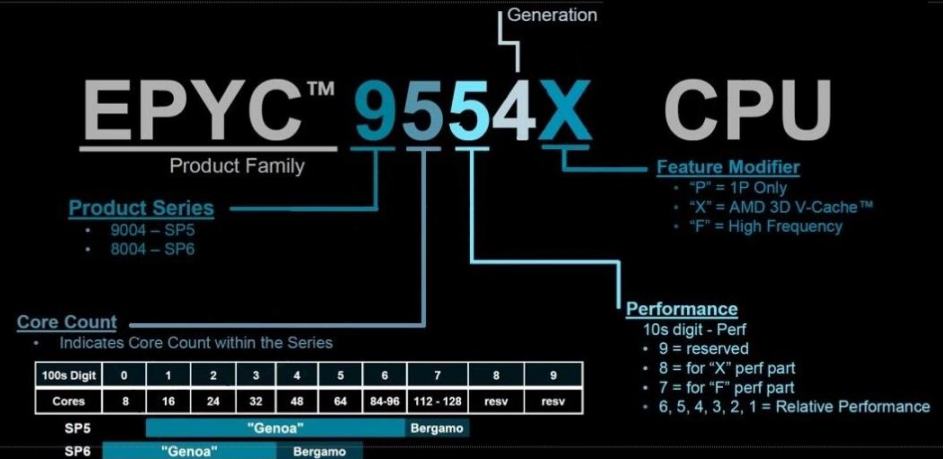
게임서버 하드웨어(CPU)

	AMD EPYC 7001 'NAPLES'	AMD EPYC 7002 'ROME'	AMD EPYC 7003 'MILAN'	AMD EPYC 9004, 8004 'GENOA', 'SIENA'
				
Core Architecture	'Zen'	'Zen 2'	'Zen 3'	'Zen 4' and 'Zen 4c'
Cores	8 to 32	8 to 64	8 to 64	8 to 128
IPC Improvement Over Prior Generation	N/A	~24% ^{ROM-236}	~19% ^{MLN-003}	~14% ^{EPYC-038}
Max L3 Cache	Up to 64 MB	Up to 256 MB	Up to 256 MB	Up to 384 MB (EPYC 9004) Up to 128 MB (EPYC 8004)
Max L3 Cache with 3D V-Cache™ technology			768 MB	Up to 1152 MB
PCIe® Lanes	Up to 128 Gen 3	Up to 128 Gen 3	Up to 128 Gen 4	Up to 128 Gen 5 8 bonus lanes Gen 3
CPU Process Technology	14nm	7nm	7nm	5nm
I/O Die Process Technology	N/A	14nm	14nm	6nm
Power (Configurable TDP [cTDP])	120-200W	120-280W	155-280W	70-400W
Max Memory Capacity	2 TB DDR3-2400/2666	4 TB DDR4-3200	4 TB DDR4-3200	6 TB DDR5-4800

게임서버 하드웨어(CPU)



AMD EPYC™ Zen4 Processor Naming Convention



게임서버 하드웨어(CPU)

- CPU 발전의 Trend

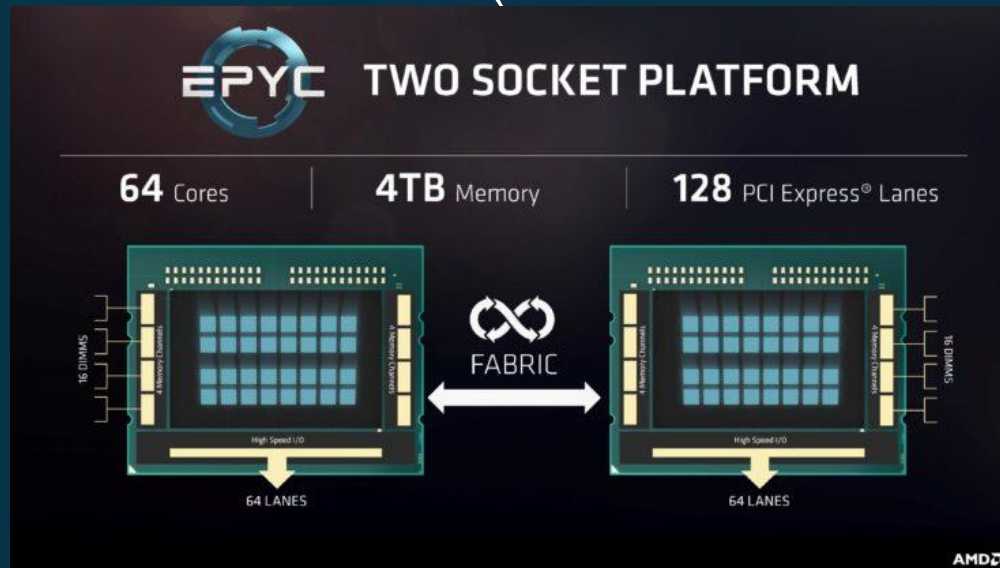
- 클럭속도 증가 (~ 2000년 까지)
 - 반도체 생산 공정 개선
 - 물리법칙 한계에 부딪침 : 발열
- Clock당 수행되는 명령어의 개수 증가 (IPC)
 - 아키텍처의 개선
 - Pipeline, SuperScalar, SuperPipeline, Out-of-order
 - 캐시용량 증가.
 - 한계에 부딪침 : 한계효용의 법칙

게임서버 하드웨어(CPU)

- CPU 발전의 Trend

- Core 개수 증가 (2005년 ~)

- 현재의 방식, 프로그램 작성 방식이 바뀌어야 함
 - 발열로 인한 클럭속도 향상의 한계가 주 원인
 - 현재 128-Core까지 (앞으로 계속 늘어날 예정)



게임서버 하드웨어(CPU)

ISSCC 70th Anniversary 1954-2023



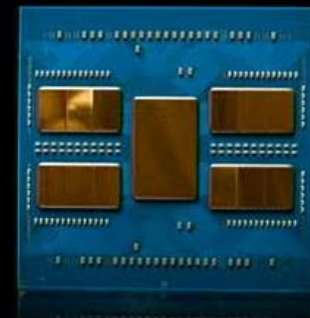
1954

- Tr-1
- 4 NPN Germanium Transistors



2013

- AMD mobile APU
- 1.3B transistors
- 4 cores/4 threads
- Monolithic 32nm SOI
- 4MB total cache



2023

- 4th gen AMD EPYC CPU
- 90 billion transistors
- 96 cores/192 threads
- 13 5nm/6nm FinFET chiplets
- 386MB total cache

게임서버 하드웨어(CPU)

- 파이프라인의 발전

- “어떤 명령어를 수행하는데 몇 cycle이 걸리는가?”라는 질문은 의미가 없음
- 파이프라인이 무효화 되지 않는 한 프로그램의 실행 속도는 메모리 Read에 종속됨
 - 명령어가 더 많아도 메모리 Read가 적으면 더 빠름

- SIMD 명령어의 발전

- 하나의 명령어로 여러 개의 실수를 동시에 연산
- MMX(MultiMedia eXtension) -> SSE(Streaming SIMD Extension) -> AVX (Advanced Vector Extension) -> AVX2

게임서버 하드웨어(CPU)

- Pipeline의 고도화에 따른 주의
 - 파이프 라인을 리셋 시키면 손해가 너무 크다
 - Pentium 4의 경우 31단계
 - 리셋의 원인
 - 시스템 콜
 - 분기 예측 오류
 - Interrupt, Exception
 - Stall의 원인 : Cache read miss
 - 대책
 - 시스템 콜을 될 수 있으면 하지 말 것
 - If, switch등을 자제한다

게임서버 하드웨어(CPU)

● CACHE

- 프로그램 실행속도에 가장 큰 영향을 미치는 요소
- Cache가 큰 CPU일 수록 속도가 빠르다

● Tip

- 같이 쓰이게 되는 데이터는 묶어 놓는다.
- 루프 안에서 사용하는 데이터는 캐쉬에 다 올라올 수 있도록 한다.
- Int대신에 short나 char을 사용한다.

Cache Behavior Affects Performance

Cost of Data Access increases with Distance from CPU

Programming Tips:

- Maximize work done on cached data
- Work with Hardware Prefetch (arrays vs linked lists)

Cost of accessing data

Where Data Is Resident	Time to fetch data
Register	1 cycle
L1 Cache	4 cycles
L2 Cache	10 cycles
L3 Cache	40-75 cycles
Memory	60-100 ns

http://software.intel.com/sites/products/collateral/hpc/vtune/performance_analysis_guide.pdf

게임서버 하드웨어(CPU)

● Multi Processor Programming

- 멀티쓰레드 프로그래밍이다.
- 잘하면 N배 성능향상, 못하면 성능하락
- Lock을 줄여라
 - Lock으로 보호 받는 코드는 N배 성능향상에서 예외
 - Lock 자체 부하 : 버스 locking
 - Semaphore, Condition 변수는 시스템 Call
- Cache Thrashing에 주의 하라 <실습 필요>
 - Cache는 line단위로 움직임

게임서버 하드웨어(CPU)

- 최근 CPU 구조를 알고 싶으면
 - <http://udteam.tistory.com/57>

게임서버 하드웨어

● Memory

- 서버가 요구하는 용량을 제공하면 된다.
- 일반적인 Desktop용 메모리가 아니라 Error수정기능이 있는 특수 메모리를 사용한다.
- 대역폭이 크지만 반응 속도는 느리다.



게임서버 하드웨어

● Network

- 10M -> 100M -> 1G -> 10G 까지 발전
- 서버간의 연결을 위해 한 서버에 여러 개의 네트워크 카드를 꽂아서 사용하기도 한다.
 - 확장 버스의 대역폭을 고려해야 한다.
 - PCI (32bit, 33MHz) : 133MB/s
 - PCI-Express 2.0 (1 lane) : 500MB/s
- 속도가 더 필요하면 Infiniband사용

Effective unidirectional theoretical throughput (actual data rate, not signaling rate)						
	SDR	DDR	QDR	FDR-10	FDR	EDR
1X	2 Gbit/s	4 Gbit/s	8 Gbit/s	10 Gbit/s	13.64 Gbit/s	25 Gbit/s
4X	8 Gbit/s	16 Gbit/s	32 Gbit/s	40 Gbit/s	54.54 Gbit/s	100 Gbit/s
12X	24 Gbit/s	48 Gbit/s	96 Gbit/s	120 Gbit/s	163.64 Gbit/s	300 Gbit/s

게임서버 운영체제

- 서버용 운영체제의 종류
 - Unix계열
 - 리눅스, FreeBSD, Solaris, OSx
 - 가격이 저렴하다.
 - 유지 보수 관리가 어렵다. (잘 아는 사람이 필요하다)
 - Windows계열
 - Windows 2016, Windows 2019, Windows 2022
 - 비싸다.
 - 유지 보수 관리가 비교적 쉽다. (배우기 쉽다. 신경쓸 것이 적다.)

프로그램 최적화

- 첫번째 : 꼭 필요한 일만 하기
 - 시스템 호출 최소화 (new/delete 포함)
- 두번째 : 좋은 알고리즘 사용하기 **O()**
- 세번째 : 메모리 복사 줄이기
 - Call by Value 대신 Call by Reference
 - Copy Constructor 사용 회피
- 네번째 : **HW 영향 고려**
 - 캐시, 파이프라인
- 다섯번째 : 멀티쓰레드 프로그래밍

성능 향상

- 하드웨어가 프로그래밍 성능에 미치는 영향
실습
 - 시스템 Call
 - Cache
 - Pipelining

실제 예제

● 시스템 Call

```
#include <iostream>
#include <chrono>
#include <thread>

using namespace std;
using namespace chrono;

int main()
{
    volatile long long tmp = 0;
    auto start = high_resolution_clock::now();
    for (int j=0; j<10000000; ++j) {
        tmp += j;
        this_thread::yield();
    }
    auto duration = high_resolution_clock::now() - start;
    cout << "Time " << duration_cast<milliseconds>(duration).count();
    cout << " msec\n";
    cout << "RESULT " << tmp << endl;
}
```

실제 예제

● Cache Miss

```
for (int i=0; i<20; ++i) {  
    const int size = 1024 << i;  
    long long *a = (long long *) malloc(size);  
    unsigned int index = 0;  
    long long tmp=0;  
    const int num_data = size / 8;  
    auto start = high_resolution_clock::now();  
    for (int j=0; j<100000000; ++j) {  
        tmp += a[index % num_data];  
        index += CACHE_LINE_SIZE * 11;  
    }  
    auto dur = high_resolution_clock::now() - start;
```

Cache_Line_Size??? * 11 ????

실제 예제

● Cache Miss

선택 C:\WINDOWS\system32\cmd.exe

```

SIZE : 1K, Time : 223msecs
SIZE : 2K, Time : 222msecs
SIZE : 4K, Time : 227msecs
SIZE : 8K, Time : 226msecs
SIZE : 16K, Time : 222msecs
SIZE : 32K, Time : 220msecs
SIZE : 64K, Time : 219msecs
SIZE : 128K, Time : 218msecs
SIZE : 256K, Time : 218msecs
SIZE : 512K, Time : 264msecs
SIZE : 1024K, Time : 308msecs
SIZE : 2048K, Time : 288msecs
SIZE : 4096K, Time : 248msecs
SIZE : 8192K, Time : 705msecs
SIZE : 16384K, Time : 829msecs
SIZE : 32768K, Time : 838msecs
SIZE : 65536K, Time : 853msecs
SIZE : 131072K, Time : 905msecs
SIZE : 262144K, Time : 1000msecs
SIZE : 524288K, Time : 1018msecs
SIZE : 1048576K, Time : 1107msecs
계속하려면 아무 키나 누르십시오 . . .

```

CPU-Z

CPU | Caches | Mainboard | Memory | SPD | Graphics | Bench | About

Processor

Name	Intel Core i7 6700		
Code Name	Skylake	Max TDP	65.0 W
Package	Socket 1151 LGA		
Technology	14 nm	Core Voltage	1.020 V

Specification

Intel®Core™i7-6700 CPU @ 3.40GHz

Family	6	Model	E	Stepping	3
Ext. Family	6	Ext. Model	5E	Revision	R0

Instructions MMX, SSE, SSE2, SSE3, SSSE3, SSE4.1, SSE4.2, EM64T, VT-x, AES, AVX, AVX2, FMA3, TSX

Clocks (Core #0)

Core Speed	800.19 MHz
Multiplier	x 8.0 (8 - 40)
Bus Speed	100.02 MHz
Rated FSB	

Cache

L1 Data	4 x 32 KBytes	8-way
L1 Inst.	4 x 32 KBytes	8-way
Level 2	4 x 256 KBytes	4-way
Level 3	8 MBytes	16-way

Selection: Socket #1 Cores: 4 Threads: 8

CPU-Z Ver. 1.79.0.x64 Tools Validate Close

실제 예제

- Pipeline stall

- conditional branch miss
 - 조건부 분기와 분기 예측 실패

```
#define abs(x) ((x)>0)?(x):- (x))
```

VS

```
#define abs2(x) (((x>>31)^x)-(x>>31))
```


실제 예제

- 절대 값?

- $y = x \gg 31$???

- x 가 양수이면 y 는 0

- x 가 음수이면 y 는 -1 (111111111...11)

- \gg 연산은 모든 비트를 오른쪽으로 이동하고 MSB는 복사한다. (나누기 연산을 shift로 대체하기 위해서)

- $(x \wedge y) - y$???

- x 가 양수이면 계산결과는 x

- x 가 음수이면 $x \wedge (1111111111) + 1$

- 모든 비트를 반전시키고 1을 더한다 => 부호를 바꾼다!!!

실제 예제

● Pipeline stall

```
int main()
{
    int sum;

    for (int i = 0; i < T_SIZE; ++i) rand_arr[i] = rand() - 16384;

    sum = 0;
    auto start_t = high_resolution_clock::now();

    for (int i = 0; i < T_SIZE; ++i) sum += abs(rand_arr[i]);

    auto du = high_resolution_clock::now() - start_t;
    cout << "[abs] Time " << duration_cast<milliseconds>(du).count() << " ms\n" ;
    cout << "Result : " << sum << endl;

    sum = 0;
    start_t = high_resolution_clock::now();

    for (int i = 0; i < T_SIZE; ++i) sum += abs2(rand_arr[i]);

    du = high_resolution_clock::now() - start_t;
    cout << "[abs2] Time " << duration_cast<milliseconds>(du).count() << " ms\n";
    cout << "Result : " << sum << endl;
}
```

실제 예제

- Pipeline stall
 - branch miss
 - 조건부 분기와 분기 예측 실패
 - 만일 루프에서 같은 값만을 사용한다면?

정리

- 게임서버의 최적화는 HW도 고려해야 한다.
 - 컴퓨터 구조에서 배운 것을 실제로 고려해야 한다.
 - System Call 하지 않기
 - Cache 잘 사용하기
 - 가능하면 메모리 적게 사용하기
 - Multi-Thread Programming (고급멀티쓰레드 프로그래밍, 멀티코어프로그래밍 과목에서)