

Computer Hardware & Industry Perspective

커리큘럼

주차	날짜	이론	실습	토론 주제
1주차	4/20 (일)	오리엔테이션, 하드웨어 구성 요소 개요	데스크탑 전체 분해, 주요 부품 구조 관찰	컴퓨터를 이해한다는 건 무엇일까?
2주차	4/26 (토)	메모리 & 저장장치 구조	RAM 분리/장착, SSD 분해 관찰	저장 장치는 결국 사라질까?
3주차	5/3(토)	발열과 냉각 구조	CPU 쿨러 제거 + 써멀 재도포, 팬 속도 측정, 온도 로깅	열 관리가 성능을 결정한다면 우리는 어떻게 설계해야 할까?
4주차	미정	CPU 구조 & 최신 기술 + 분해 실습	CPU 쿨러 제거, 외부 구조 관찰, 발열 설계 확인	CPU는 왜 더 이상 클럭만으로 경쟁하지 않을까?
5주차	미정	GPU 구조 & 최신 기술 + 분해 실습	GPU 분해, 방열판 구조 분석, 팬 곡선 실험	AI와 게임 성능을 동시에 잡으려면 GPU는 어떻게 진화해야 할까?
6주차	미정	입출력 장치 & HCI	키보드/마우스 분해 및 센서 원리 학습	사람과 기계의 경계는 어디까지 사라질 수 있을까?
7주차	미정	반도체 제작: 웨이퍼~패키징 전체 흐름	공정 흐름도 도식화, 칩 구조 스케치, 패키지 비교	성능은 결국 패키징으로 결정되는가?
8주차	미정	미래 컴퓨팅 구조 - 포스트 폰 노이만 등	아키텍처 비교, 미래 예측	우리는 어떤 컴퓨팅 구조 위에서 살아가게 될까?

※ 일정은 변동 가능하며, 팀 내 논의를 통해 유동적으로 조정될 수 있습니다.

※ 주차별로 배운 내용을 다음주까지 리포트(선각)를 작성해 발표합니다.

3주차 - 발열과 냉각 구조

- 이론:
 - 전자제품에서 발생하는 발열의 원리
 - 소비전력, 클럭, TDP와 온도의 관계
 - 공랭, 수랭, 히트파이프, 베이퍼 챔버 냉각 방식 비교
 - 써멀그리스와 써멀패드의 역할
- 실습:
 - stress-ng: CPU 부하, 연산 방식별 발열 유도 실험
 - 3DMark: GPU 벤치마크로 실제 게임 시 발열 상황 재현
 - 써멀구리스 재도포 전후 온도 차이 측정
- 토론:
 - ARM SoC는 왜 발열에 강할까? - 구조적 이유와 냉각 방식 차이
 - 써멀구리스, 진짜 성능 차이를 만들까? - 전도율과 도포 방식 비교
 - 데이터센터 냉각 구조는 어디까지 진화했나? - 액침냉각, 해양냉각, AI 예측 쿨링 등
- 3주차 선각 주제:
 - ARM SoC는 왜 발열에 강할까? - 구조적 이유와 냉각 방식 차이

이론 → 실습 → 토론

- 이론: 전체 시스템의 흐름 감 잡기
- 실습: 손으로 구조 익히기
- 토론: 철학적 질문으로 인식 확장하기

이론

발열과 냉각 이해하기

성능이 좋은 컴퓨터를 샀는데
게임이 자꾸 끊겨요...

성능은
발열 한계 안에서만
의미가 있다



열은 왜 생길까?

1. 전력 소비 = 열 발생

- CPU나 GPU는 트랜지스터 수십억 개로 구성되어 있고, 이 트랜지스터들이 초당 수십억 번씩 on/off(스위칭) 하며 작동
- 이 과정에서 전류가 흐르고, 전자들이 실리콘 내부의 저항을 만나면서 에너지가 열의 형태로 방출
 - 마치 전기히터에서 발열이 생기는 원리와 유사

2. 전압과 클럭이 높을수록 발열도 증가

- 클럭이 높다는 건 1초에 더 많은 연산을 수행한다는 뜻이고,
 - 그만큼 더 자주 스위칭하며 더 많은 전력을 소비
-
- 전압이 높을수록 트랜지스터를 on/off시키는데 드는 에너지가 더 커지고,
 - 전력 = 전압² × 주파수 × 용량 공식에 따라 발열이 크게 증가

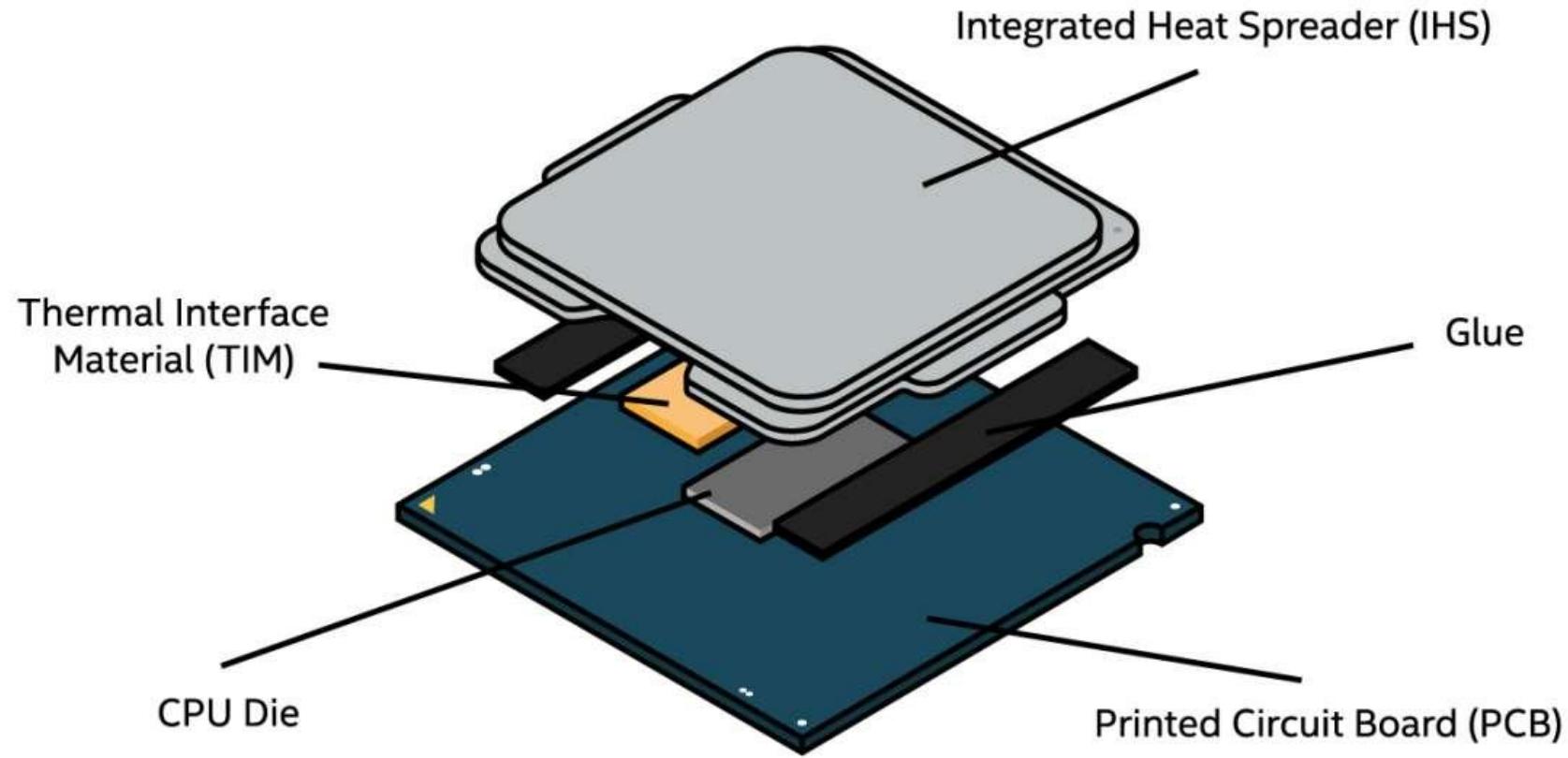
3. 누설 전류도 발열 원인

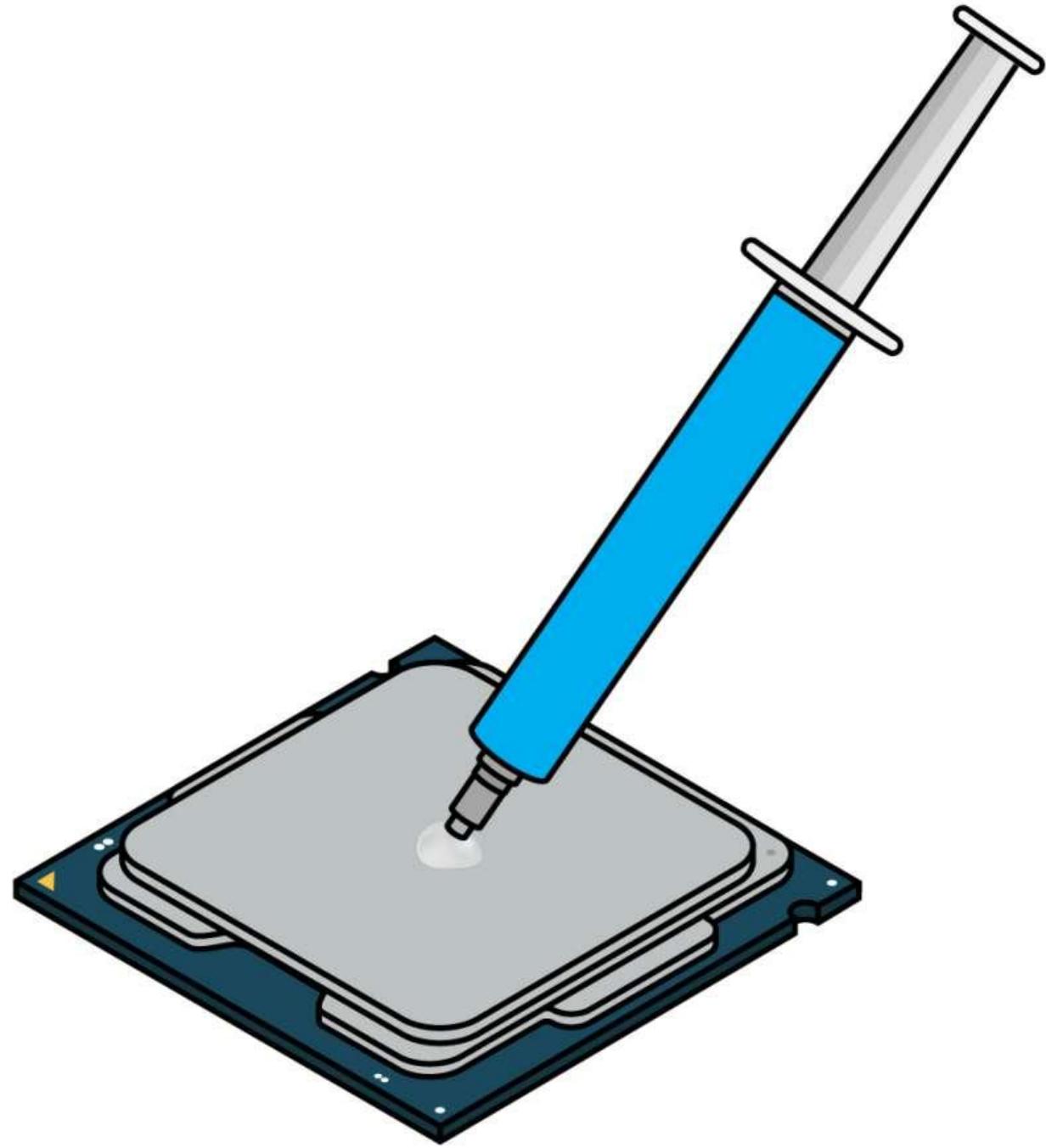
- 트랜지스터는 '꺼진 상태'에서도 아주 소량의 전류가 흐르는데,
- 이를 누설 전류 (leakage current)라고 한다
- 공정이 미세화될수록 이 전류가 무시할 수 없을 만큼 커지고, 이 역시 열로 변환된다

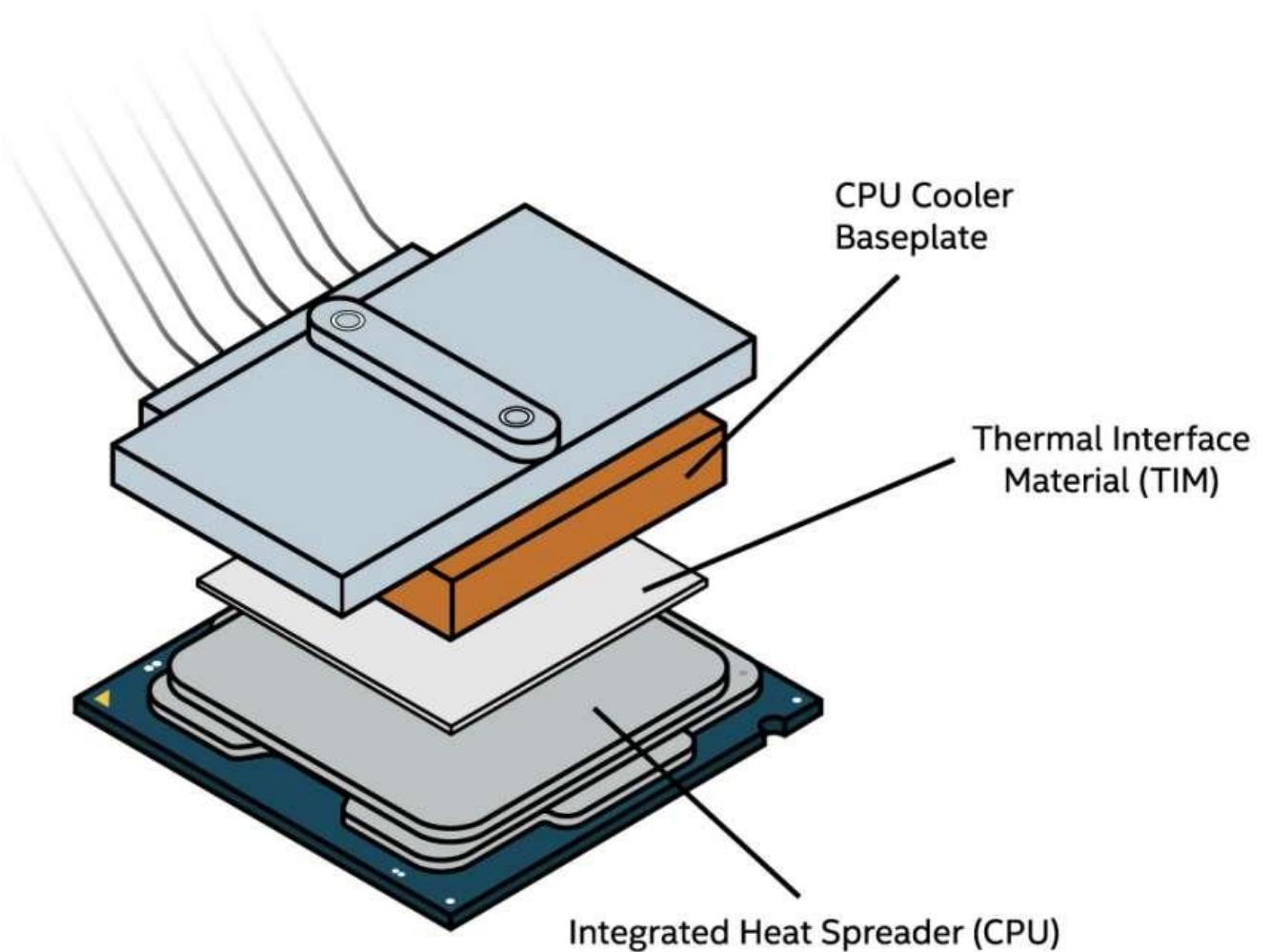
4. GPU가 더 뜨거운 이유는?

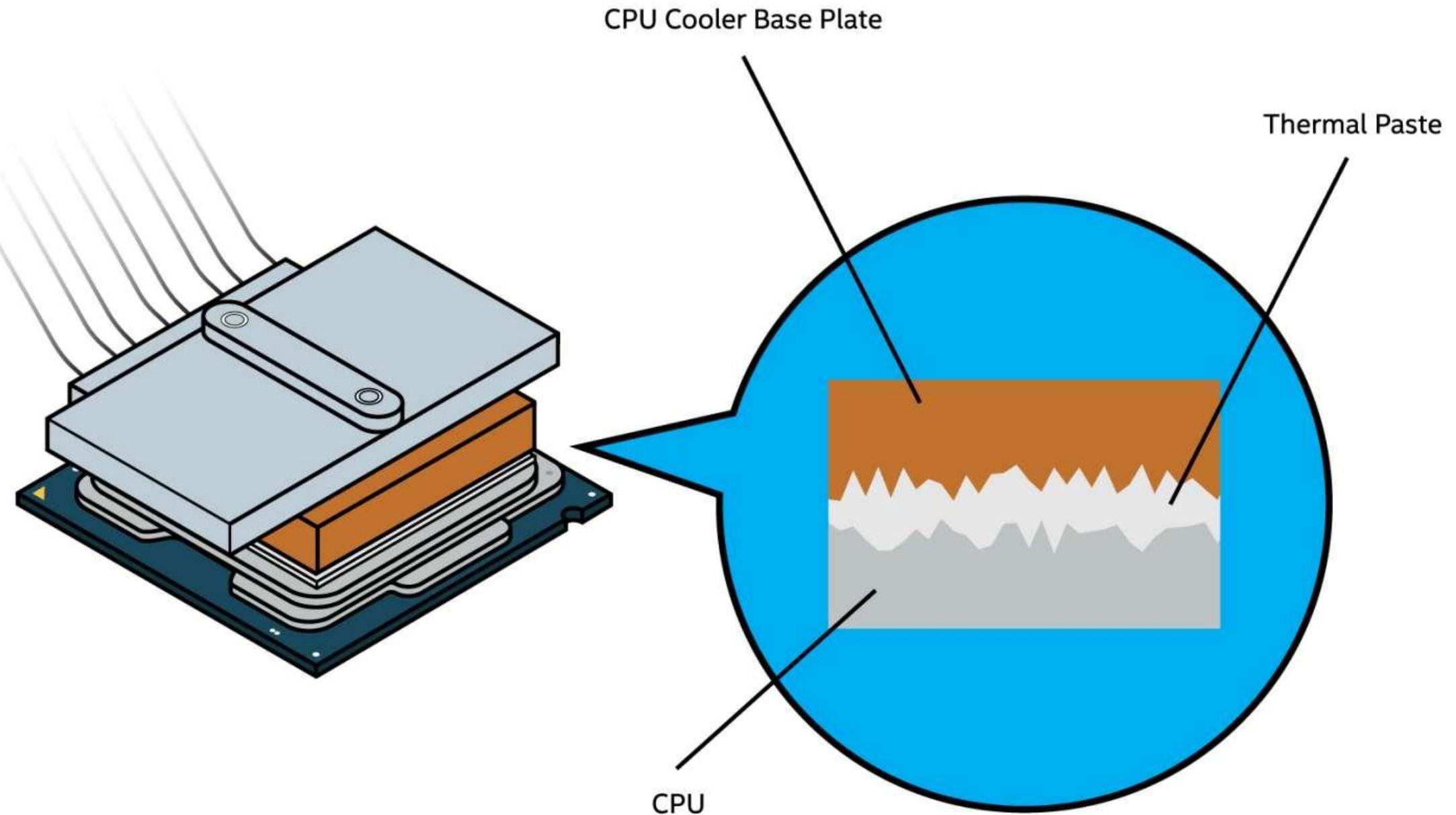
- GPU는 병렬 연산을 위해 더 많은 연산 유닛(코어)를 갖고 있어,
 - 일반적으로 CPU보다 전력 소비량과 발열이 큼
- 특히 고사양 게임, 딥러닝 연산 등에서 풀로드 상태가 지속되면
 - 수백 와트의 전력을 소모하고, 그만큼 열도 많이 발생한다

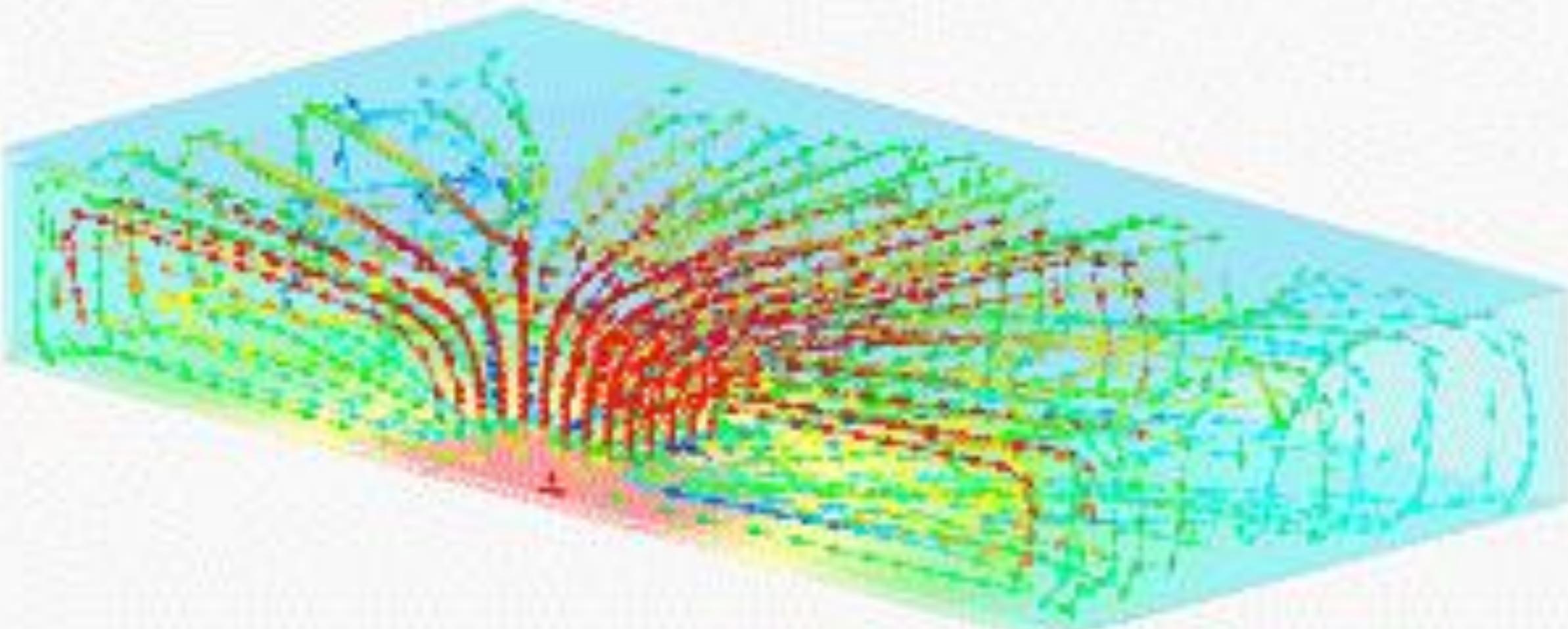
어디에서 열이 발생하나?











1. 발열 구조

(1) 발열의 시작: 실리콘 다이

- CPU나 GPU 내부의 연산이 이뤄지는 코어(core)와 캐시 메모리 등 트랜지스터가 밀집된 실리콘 다이에서 발열이 집중적으로 발생
- 특히 ALU, FPU, 컨트롤 유닛처럼 연산이 집중되는 부분에서 가장 심한 발열

(2) 히트 스팟(Hot Spot)

- 실리콘 전체가 고르게 뜨거워지는 게 아니라, 특정 연산이 집중되는 구역만 뜨거워지는 국소 발열 현상이 있음
- 이로 인해 히트 스팟 중심으로 불균형 발열 → 손상 위험이 커지기 때문에, 냉각 설계에서 이 부분을 고려해야함

2. 냉각 설계

(1) 열 전달 경로 구조

[실리콘 다이] → [IHS] → [서멀구리스] → [히트싱크 or 수냉 블록] → [팬 or 라디에이터] → [외부 공기]

(a) IHS (Integrated Heat Spreader)

실리콘 위에 덮여 있는 금속 뚜껑.

열을 넓은 면적으로 퍼뜨려서 방열판이 열을 더 잘 흡수할 수 있게 도와줌

(b) 서멀구리스 (Thermal Paste)

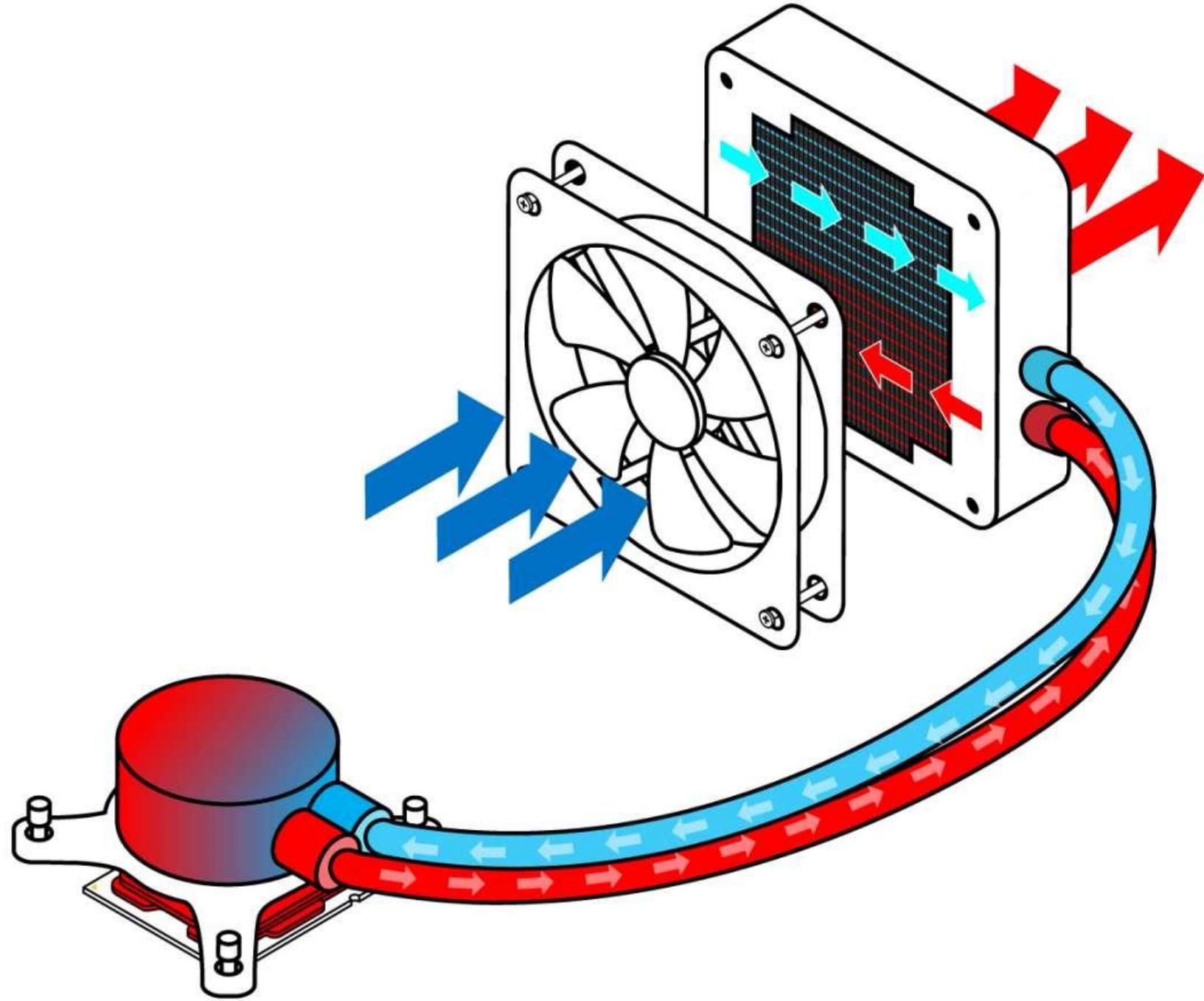
IHS와 히트싱크 사이의 미세한 틈을 메워 열 전달 효율을 높이는 전도성 물질

(c) 히트싱크 / 수냉 블록

열을 빠르게 흡수해 공기나 냉각수로 전달

공랭: 알루미늄이나 구리 핀에 팬을 달아 냉각

수냉: 냉각수가 순환하면서 블록 → 튜브 → 라디에이터를 통해 열 전달



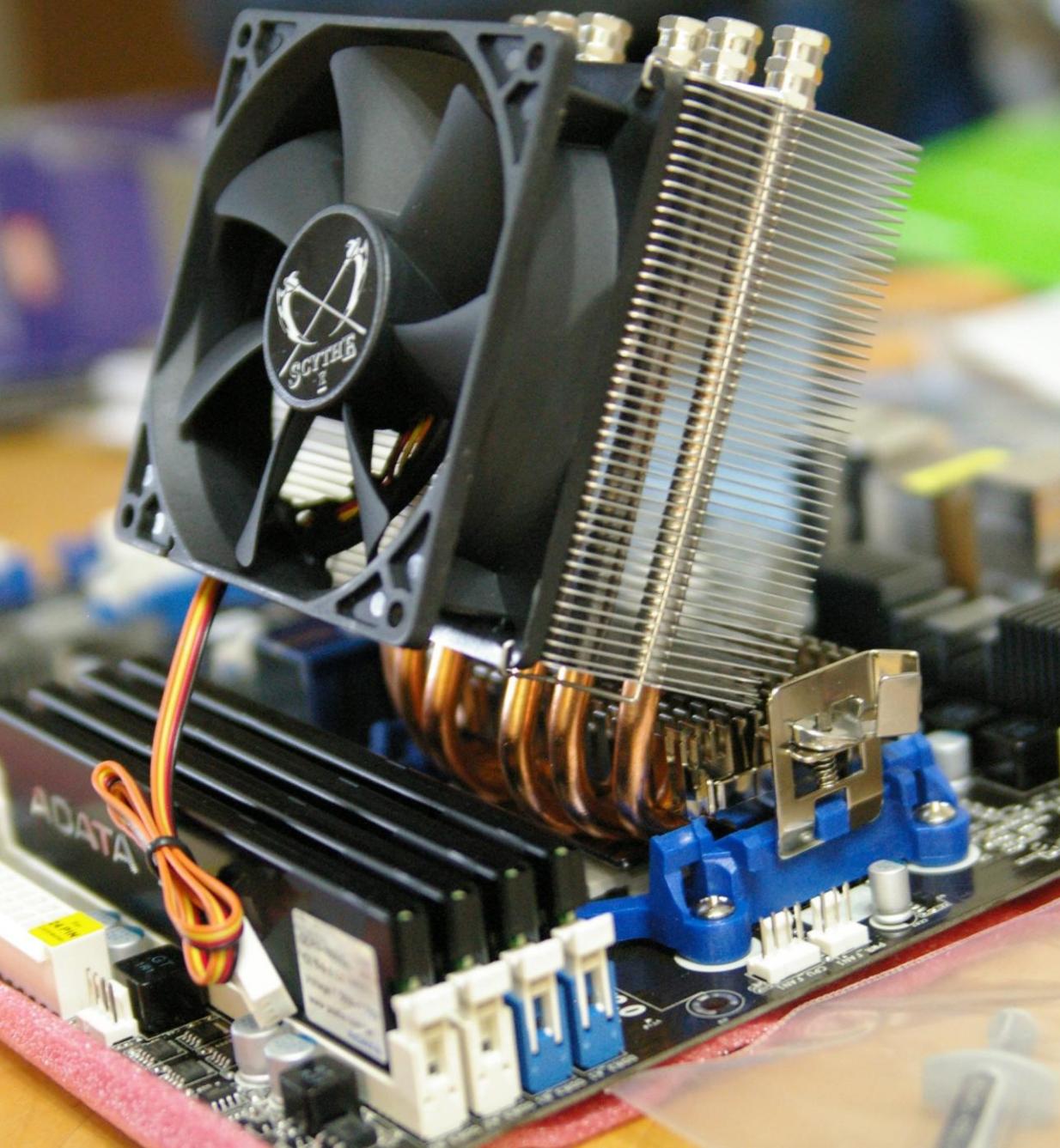
3. 냉각 방식 비교

항목	공랭 쿨러	수랭 쿨러
구조	히트싱크 + 팬	워터블록 + 펌프 + 라디에이터
장점	구조 간단, 유지보수 쉬움	고성능, 정숙성 뛰어남
단점	고열에서 한계, 공간 차지	설치 복잡, 누수 위험 존재
가격	상대적으로 저렴	상대적으로 고가

4. 추가 냉각 기술

- **히트파이프 (Heat Pipe)**
 - 구리 관에 냉매가 들어 있어 증발/응축을 통해 빠르게 열을 이동시킴
 - 대부분의 고급 공랭 쿨러에 채택
- **베이퍼 챔버 (Vapor Chamber)**
 - 고가 제품에 들어가는 평면 히트파이프 구조
 - 고른 열 분산 효과가 뛰어나 고성능 GPU에 자주 사용
- **제로팬(Zero Fan)**
 - 부하가 적을 때 팬을 멈추는 기술로, 소음과 전력을 줄임
 - GPU, 노트북에서 많이 사용





5. 실제로 발열이 문제를 일으키는 경우

- 열 쓰로틀링(Thermal Throttling) 발생
 - → 온도가 너무 높으면 시스템이 자동으로 클럭을 낮춰 성능 저하
 - → 장기적으로는 부품 수명 단축, 시스템 불안정, 다운 가능성

요약: 성능 = 발열 관리

- CPU/GPU는 전기적 연산의 부산물로 열을 생성
- 이 열을 어떻게 빠르게 빼내고 확산시키느냐에 따라, 시스템의 성능·안정성·수명이 달라짐
- 그래서 고성능 컴퓨터일수록 정교한 냉각 솔루션이 필수

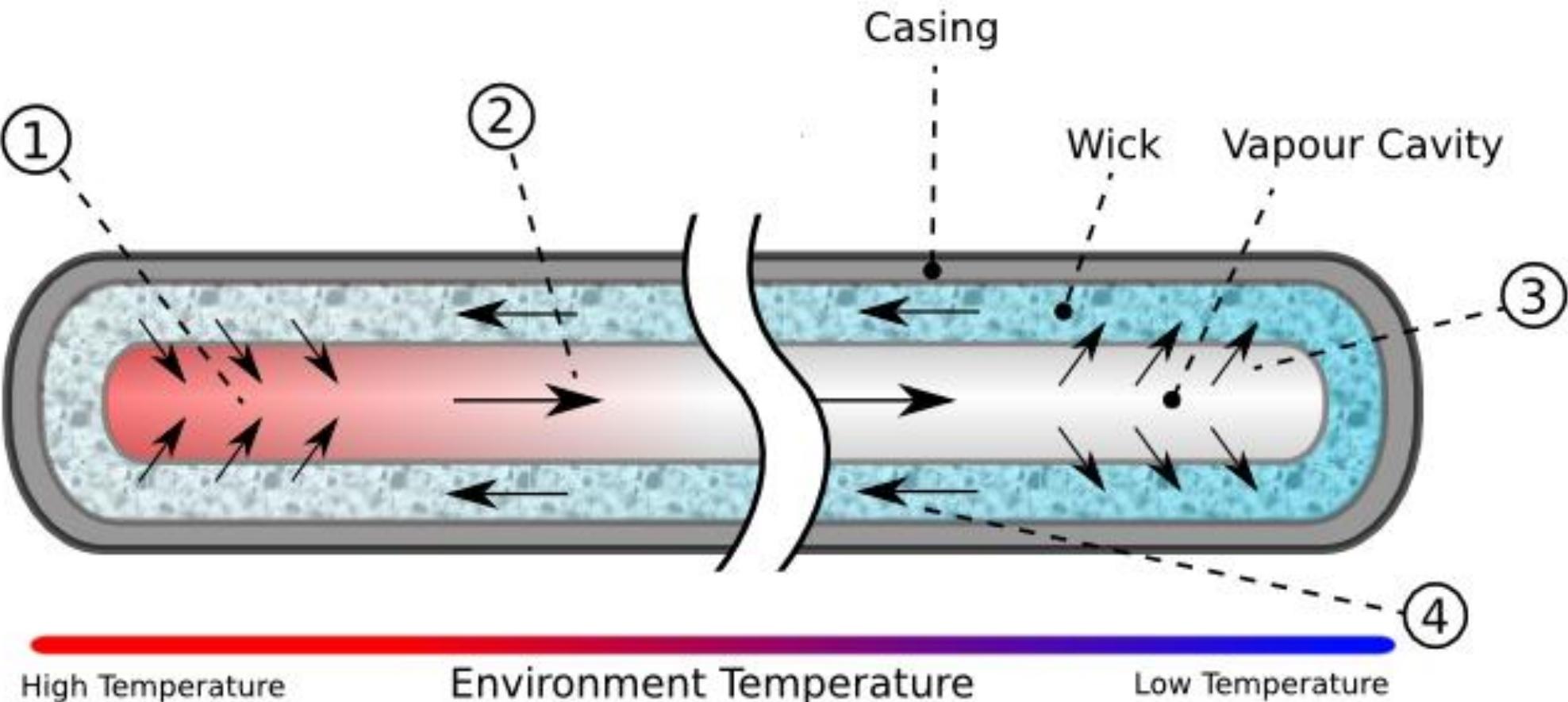
히트파이프

금속관(구리) 내부에

→ **작은 압력의 냉매(물, 암모니아 등)**

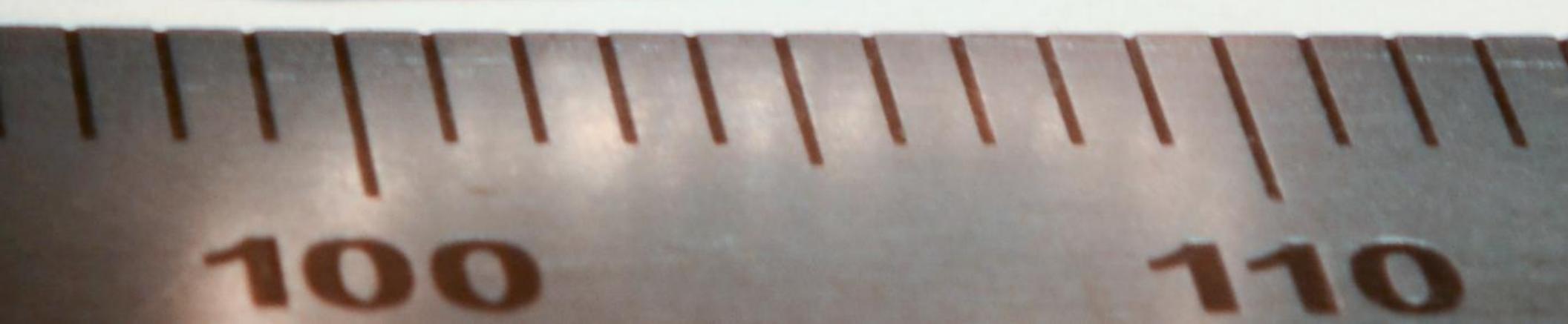
→ **모세관 구조의 심지(wick structure)가 내벽에 부착**

- 외벽: 구리나 알루미늄 등 열전도율이 높은 금속
- 심지(wick): 미세한 구멍이나 망사 구조 → 모세관 현상 유도
- 냉매: 증발과 응축이 반복될 수 있는 휘발성 액체



Heat pipe thermal cycle

- 1) Working fluid evaporates to vapour absorbing thermal energy.
- 2) Vapour migrates along cavity to lower temperature end.
- 3) Vapour condenses back to fluid and is absorbed by the wick, releasing thermal energy
- 4) Working fluid flows back to higher temperature end.



100

110

히트파이프의 동작 원리: 상변화 이용

1. 열 흡수부(열원): 증발

- 히트파이프 한쪽이 열원(CPU 등)에 닿으면,
 - 그 부분의 냉매가 액체 → 기체로 증발
- 이때 잠열(Latent Heat)을 흡수하며 주변의 열을 가져감
 - → 즉, 기화하면서 열을 흡수

2. 열 방출부(방열판 방향): 응축

- 증기로 변한 냉매가 히트파이프 내부를 따라
 - 상대적으로 차가운 쪽으로 이동
- 그곳에서 열을 방출하며
 - 다시 기체 → 액체로 응축 → 응축하면서
 - 열을 외부(히트싱크, 라디에이터)에 전달

3. 액체 복귀: 모세관 작용

- 다시 액체로 변한 냉매는 심지(wick)를 따라
 - 모세관 현상으로 원래 열원이 있는 쪽으로 되돌아옴
- 이렇게 자연 순환 시스템이 완성됨 (별도의 펌프 없이도 순환이 이루어짐)

[CPU에서 열 발생]

- [히트파이프 내 냉매 증발]
- [증기가 반대쪽으로 이동]
- [냉각되어 응축]
- [액체 상태로 심지를 따라 복귀]
- [무한 루프]

왜 히트파이프를 구리로 만드나

- 구리는 열전도율이 높고, 가격이 저렴함
- 또한 냉매와의 화학적 안정성이 높아 장기간 사용할 수 있음

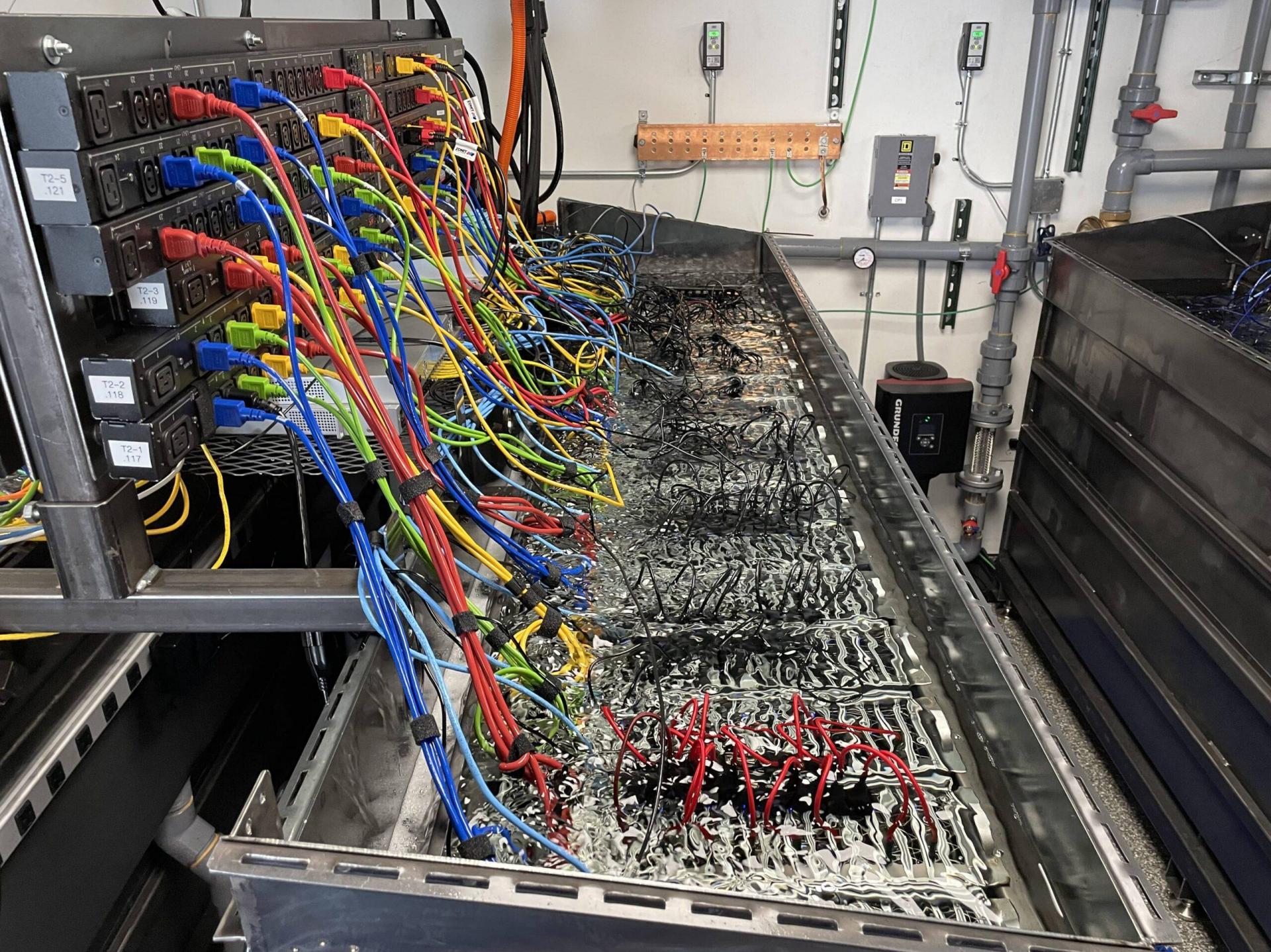
데이터 센터 냉각 시스템

서버 컴퓨터보다 냉각에 더 많은 전력을 쓰기도 한다

- 서버는 24시간 고성능 연산을 수행하며, 엄청난 열을 발생시킴
- 이 열을 식히지 못하면 장비 손상 → 서비스 중단으로 이어짐
- 그래서 냉각 시스템은 데이터 센터 운영의 핵심 인프라

주요 냉각 방식

냉각 방식	원리	특징
공랭(Air Cooling)	찬 공기로 열 제거	저렴, 유지 쉬움, 효율 낮음
수랭(Water Cooling)	냉각수 순환으로 열 흡수	높은 효율, 복잡한 설비
액침냉각(Immersion Cooling)	서버를 절연 냉각액에 담금	차세대 기술, 발열 완벽 대응
냉각타워(Cooling Tower)	증발열 이용해 대기 중으로 방열	수랭 시스템과 함께 사용
열 교환기(Heat Exchanger)	냉각 루프 사이의 열만 전달	내부/외부 분리 가능







1. 공랭 시스템 예시

- **CRAC (Computer Room Air Conditioner)**
 - 서버 룸 바닥 아래로 차가운 공기를 불어넣고, 더운 공기는 위로 빠져나감
 - 냉방기기가 서버 열을 흡수해서 다시 순환
 - 전통적인 방식이지만, 최근 고밀도 서버에는 부족함

2. 수랭 시스템 예시

- **Rear Door Heat Exchanger (RDHx)**
 - 서버 랙 뒷면에 물 기반 방열기 설치
 - 열을 물이 직접 흡수하여 외부로 전달
- **Direct-to-Chip Liquid Cooling**
 - CPU/GPU 위에 수냉 블록을 직접 설치해 열을 빼냄
 - 서버 내부에 수로(튜브)가 구성되어 있어 정교한 설계 필요

3. 액침 냉각 (Immersion Cooling)

- 서버 전체를 절연성 냉각액에 담금
- 열이 즉시 액체로 전달되어 팬 없이도 완전 냉각 가능
- 냉각 효율 매우 높음 → AI 서버, 고밀도 클러스터에 적합

4. 미래 냉각 기술 동향

- 자연 냉각(Free Cooling): 외부 차가운 공기를 직접 활용
- 지하 데이터센터 / 북유럽 설립: 낮은 외부 온도를 활용
- AI 기반 온도 제어: Google, Meta 등은 냉각 효율화를 위해 AI 도입

발열 처리를 위해 아키텍처가 변하고 있다

더 빨라지기 위해서가 아니라, 더 뜨거워지지 않기 위해 구조가 바뀌고 있다

문제의 시작: 발열 병목

미세공정(5nm, 3nm) 시대에도 성능이 무한정 늘지 않는 이유는?

→ 발열이 병목(Bottleneck)이 되기 때문

칩 면적은 줄어드는데, 연산 밀도는 커지고, 트랜지스터는 더 빨리 스위칭

→ 전력 밀도(Power Density) 폭증

→ 한 지점에 열이 집중되며 냉각으로 제어 불가

그래서 구조가 바뀌었다

1. 칩렛(Chiplet): 열을 나눠버리자

문제	해결
단일 칩에 연산 집중 → 국소 발열 폭발	칩을 여러 개로 나눠 열을 분산
발열이 집중된 곳은 냉각도 어려움	칩마다 열 특성 따라 별도 냉각 가능

2. 3D 패키징: 위로 쌓고, 열은 아래로 분산

문제: 수평 면적 한계로 인해 더 이상 칩 확장 불가

해결: 칩을 수직으로 적층, TSV로 연결

→ 하지만 위층에 열이 쌓이면 아래로 빠지지 않음

→ 그래서 발열 많은 연산 유닛은 하단에 배치, 메모리 등은 상단에 배치

3. CXL(Compute Express Link): 발열의 원인을 나눠놓자

문제: 연산이 특정 칩에 집중될수록 발열도 집중

해결: CPU, GPU, 메모리 등 서로 다른 장치로 연산 분산

- 각 장치는 자신만의 냉각 시스템 사용 가능
- 발열을 물리적으로 떨어뜨려 제어
- CXL은 ‘구조적 발열 분산’을 위한 설계

[전력 밀도 증가] → [국소 발열] → [냉각 불가]



[칩렛] → 열 분산

[3D 적층] → 열 하향 배치

[CXL] → 열의 공간적 분리

**이제 컴퓨터 아키텍처는 계산을 잘하기 위해서가 아니라,
‘열을 잘 흘려보내기 위해’ 다시 설계되고 있다.**

실험

발열 유도와 발열 측정 소프트웨어

실험 번호	실험 명	사용 명령어 / 코드	실험 목적	관찰 포인트
①	기본 CPU 부하	<code>stress-ng --cpu 4 --timeout 30s</code>	단순 CPU 점유에 의한 발열 유도	부하 시 온도 상승 속도, 팬 반응
②	CPU 연산 종류별 발열 비교	<code>--cpu-method sqrt, memcpy, io</code>	연산 방식에 따른 발열 차이 비교	연산별 온도/점유율 차이
③	고밀도 행렬 곱 연산 + 메모리 부하	<code>--cpu-method matrixprod --vm 2 --vm-bytes 1G</code>	CPU + 메모리 복합 부하 상황	시스템 전체 발열 패턴
④	멀티코어 발열 한계 실험	<code>--cpu 8 --timeout 60s</code>	모든 논리 코어 점유 → 최대 발열 유도	열 누적 속도, 스로틀링 발생 여부
⑤	파이썬 무한 루프	<code>while True: pass</code>	CPU 한 코어를 지속 점유하는 최소 단위 코드	팬 반응 속도, 클럭 변화
⑥	써멀구리스 재도포 전후 비교	동일 부하 + 재도포 후 비교	쿨링 접촉 개선 전후 효과 측정	최고 온도, 회복 속도 비교
⑩	스로틀링 발생 시점 측정	<code>stress-ng + watch -n 1 sensors + lscpu</code>	부하 상태에서 클럭이 감소하는 시점 관찰	TJMax 도달 타이밍, 보호 회로 동작

stress-ng

- 커널 인터페이스에 부하를 가해 시스템에 스트레스를 유도하는 리눅스 기반 스트레스 테스트 도구
- 이 도구는 CPU, 메모리, 디스크 등 다양한 자원에 부하를 줄 수 있는 수백 가지의 스트레스 메커니즘을 제공
- 스트레스 테스트는 일반적으로 시스템이 과도하게 작동하거나 발열이 심할 때 발생할 수 있는 하드웨어 문제(과열, 스로틀링 등)를 유도해, 시스템의 안정성과 한계점을 실험하는 데 사용

3DMark

- GPU와 CPU의 성능을 측정하고 시각적으로 시뮬레이션해 주는 그래픽 중심의 벤치마크 테스트 도구
- 고성능 게임 그래픽과 물리 연산, 연속적인 렌더링 장면을 통해 실제 게이밍/고부하 환경에서 시스템이 얼마나 잘 버티는지를 시각적으로 보여줌

- **stress-ng는 코드 기반 부하로 열을 유도했다면,**
- **3DMark는 시각적 연산 부하로 GPU 중심의 실험을 진행**

- 즉, **stress-ng는 계산기 실험, 3DMark는 게임기 실험이라고 이해해도 좋음**

실습

CPU쿨러 분해와 GPU분해를 통한
쿨링 시스템 관찰
써멀 그리스 도포 체험

[과제 안내] 3주차 선각 리포트

주제 : Arm SoC는 왜 발열에 더 강할까?

주제 해설 : 왜 모바일 칩은 팬도 없는데 성능이 괜찮고, 발열도 심하지 않을까?

반면 인텔 기반 노트북은 팬이 미친 듯이 돌고, 몇 분이면 뜨거워진다. 둘의 차이는 무엇일까?

제출 방법

- 파일명: `홍길동_3주차_선각리포트.pdf`
- 형식: PDF 변환 후 슬랙 `CHIP 톡방`에 제출
- 마감: 다음 모임 전까지
- 다음 모임 : 5월 10일 토요일 22시

다음 시간엔?

우리는 CPU 속 수십억 개 트랜지스터가 어떻게 계산을 수행하고, 어떻게 발열을 유발하며, 그 구조가 어떻게 성능과 효율을 결정하는지 파고듭니다.

- CPU 구조와 최신 아키텍처의 세계로 -