C-4 BombMark

ARM Cortex-A 기반
Multi-Processing 커널 및 응용프로그램 설계

{ 성능을 테스트 해보세요! };

// with various Open source Benchmark :)

멀티 프로세싱을 통한
시스템의 성능 측정

구현 내용
- Context Switching
- System Call
- Demand Paging
- Porting Opensource Benchmark on Embedded System

/* C반 4팀 김동영/이연주/정경환/조재용 */

What is the C-4 BombMark?

아무개 연구원은 입사 초기 시스템을 측정할 수 있는 임베디드 시스템 벤치마크를 만들어달라는 니즈 (aka. VOE)를 듣게 되며 벤치마크에 대한 생각에 사로잡히게 되었다...
마침 아무개 연구원은 SW Bootcamp 임베디드 과정에 참여하게 되며 그 꿈을 펼칠 수 있는 기회를 얻게 된다.
이러한 비하인드 스토리로 탄생하게 된 C-4 BombMark는 Cortex-A9 기반의 임베디드에 오픈소스 벤치마크를 포팅 하여 Cache 및 Branch prediction 비/활성화에 따른 성능을 측정할 수 있다.
이로써 아무개 연구원은 '벤치 마스터'로 진화할 수 있었다.

포팅한 Benchmark 중 하나인 embench

embench: 오픈소스 임베디드 시스템 벤치마크
(github.com/embench/embench-iot)

Baseline Data

Name	Comments	Orig Source	C LOC	code size	data size	time (ms)	branch	memory	compute
aha-mont64	Montgomery multiplication	AHA	162	1,052	0	4,000	low	low	high
crc32	CRC error checking 32b	MiBench	101	230	1,024	4,013	high	med	low
cubic	Cubic root solver	MiBench	125	2,472	0	4,140	low	med	med
edn	More general filter	WCET	285	1,452	1,600	3,984	low	high	med
huffbench	Compress/Decompress	Scott Ladd	309	1,628	1,004	4,109	med	med	med
matmult-int	Integer matrix multiply	WCET	175	420	1,600	4,020	med	med	med
minver	Matrix inversion	WCET	187	1,076	144	4,003	high	low	med
nbody	Satellite N body, large data	CLBG	172	708	640	3,774	med	low	high
nettle-aes	Encrypt/decrypt	Nettle	1,018	2,880	10,566	3,988	med	high	low
nettle-sha256	Crytographic hash	Nettle	349	5,564	536	4,000	low	med	med
nsichneu	Large - Petri net	WCET	2,676	15,042	0	4,001	med	high	low
picojpeg	JPEG	MiBench2	2,182	8,036	1,196	3,748	med	med	high
qrduino	QR codes	Github	936	6,074	1,540	4,210	low	med	med
sglib-combined	Simple Generic Library for C	SGLIB	1,844	2,324	800	4,028	high	high	low
slre	Regex	SLRE	506	2,428	126	3,994	high	med	med
st	Statistics	WCET	117	880	0	4,151	med	low	high
statemate	State machine (car window)	C-LAB	1,301	3,692	64	4,000	high	high	low
ud	LUD composition Int	WCET	95	702	0	4,002	med	low	high
wikisort	Merge sort	Github	866	4,214	3236	4,226	med	med	med

What to Measure?

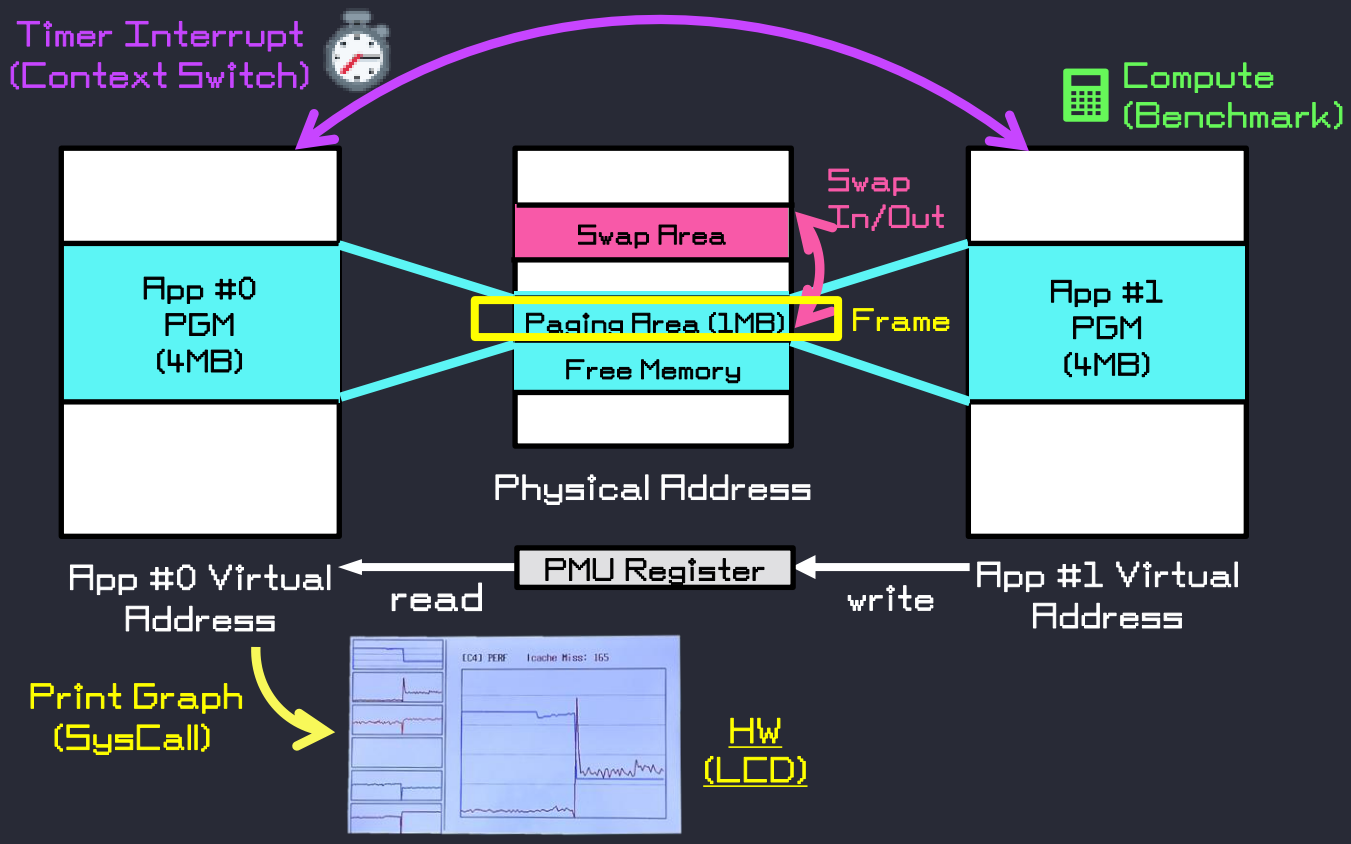
측정 대상	저장 대상/역할	특징
I-Cache	명령어	명령어 전용 L1 Cache, 코어 전용
D-Cache	데이터	데이터 전용 L1 Cache, 코어 전용
L2 Cache	명령어 + 데이터	L1 Cache 용량 ↑, 속도는 다소 ↓, 공유 Cache
Branch Prediction	분기 예측	Pipeline 효율 극대화

Supported Benchmark 무려 12개 지원 !

DhryStone: 프로세서와 컴파일러의 정수 계산 성능 측정 Benchmark
aha-mont64: embench의 *Montgomery 곱셈 연산 성능 측정 Benchmark
CRC32: 오류 검출 방식 중 하나인 CRC32 알고리즘 성능을 측정하는 Benchmark (embench)
and so on ···

* 공개키 암호 알고리즘에서 핵심적으로 사용되는 모듈러 곱셈 기법

System Layout.



1. OS 영역

- Context Switching :
  일정시간 마다 Interrupt 를 발생시켜 App#0 과 App#1이 전환되어 거의 실시간으로 동작
- Demand Paging :
  Paging Area(1MB) 에 App#0, App#1 의 데이터를 Swap In/Out 하여 작은 메모리 환경에서도 동작 되도록 관리

2. APP #0: Graph Printer

- SVC exception을 활용한 Device driver 등의 OS 자원 접근 (LCD)
- Context switch를 통한 Performance graph 실시간 update 및 출력

3. APP #1: Benchmark test

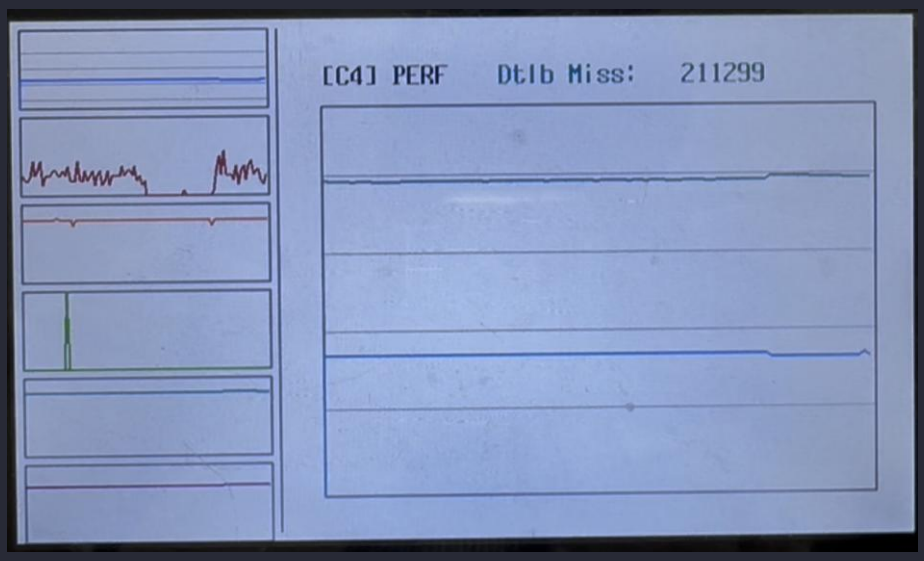
- Performance Monitoring Unit (PMU)을 활용해 benchmark 측정
- Dtlb, Itlb, Icache, Dcache 그리고 branch prediction miss 측정
- 이미지 압축 등 작업 진행 시 하드웨어 자원 사용에 대한 정보 PMU update
- Data cache 등 HW 자원 on/off 따른 performance 변동 측정

Benchmark Info.

TEST 조건

- L2, D-Cache, I-Cache 모두 켠 상태에서 benchmark 측정
- Branch prediction (BP) 여부에 따른 Dhrysteon test 진행

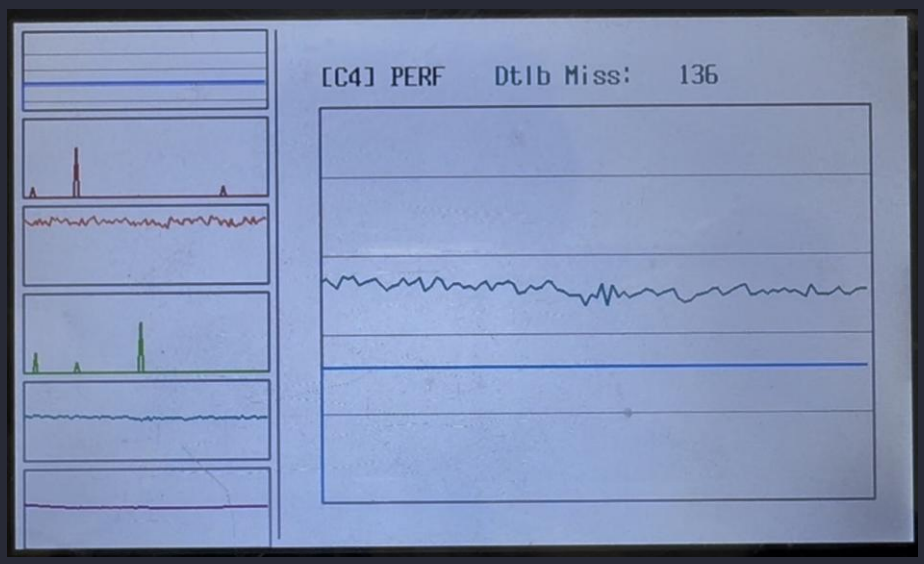
CPU 사용량
ICache Miss
Itlb Miss
DCache Miss
Dtlb Miss
BP Miss



Overall

조건 1: L2, D-Cache, I-Cache ON, Prediction OFF

CPU 사용량
ICache Miss
Itlb Miss
DCache Miss
Dtlb Miss
BP Miss



Overall

조건 2: L2, D-Cache, I-Cache ON, Prediction ON

TEST 결과

- BP 기능 실행 시 데이터 접근 패턴의 Locality 개선에 따른
- 실험 결과 Dtlm miss 개수가 현저하게 감소