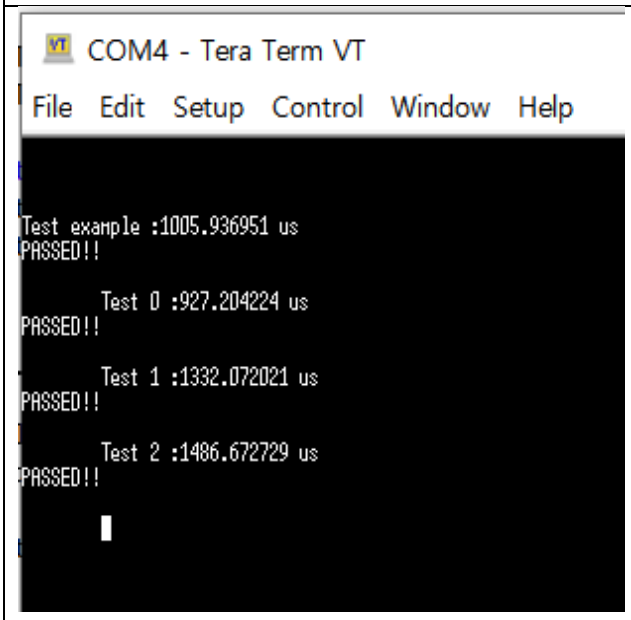
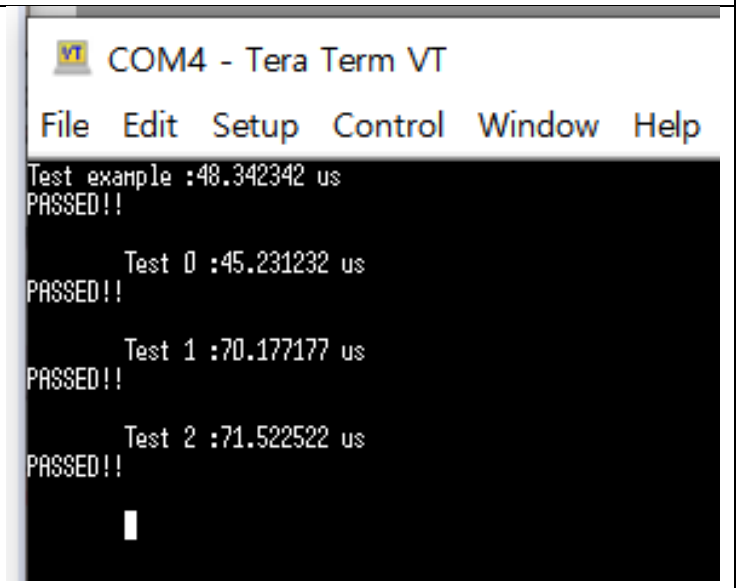
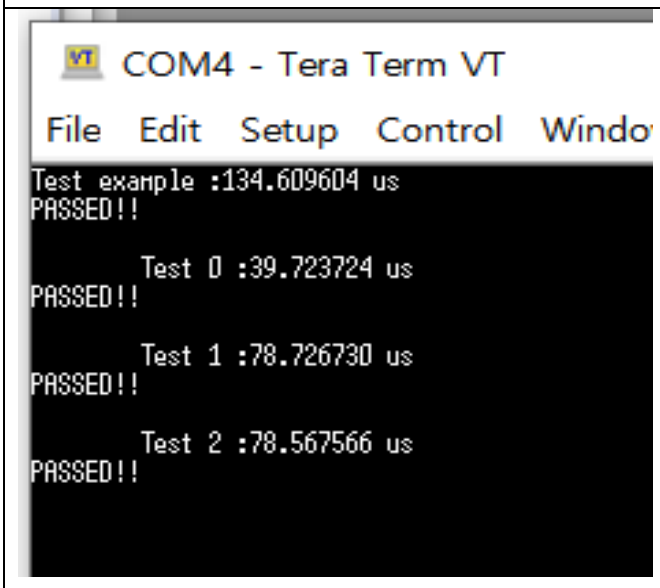
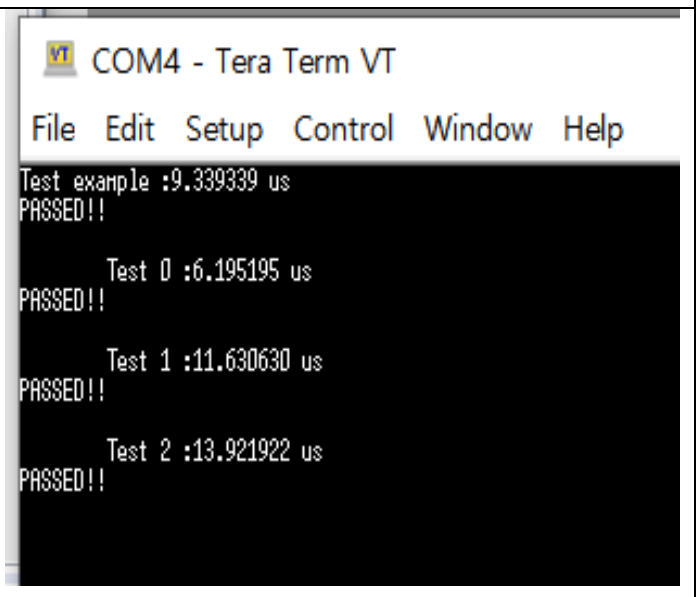
201810528 고려욱 Lab2 과제

1. Optimization level : O0

D-Cache Disabled	D-Cache Enabled
	

2. Optimize level -O3

D-Cache Disabled	D-Cache Enabled
	

Speed-up 이유

1) D-Cache Disable/ Enable

각 Optimize level에서의 speed-up 차이에는 D-Cache에 접근이 허락되었는가 아닌가에 있다. Intro.pdf 5p.를 참고하면 Processor Unit에는 I-Cache와 D-Cache가 각각 32KB로 존재한다. 프로그램 실행시에 기본값은 D-Cache는 Disable 즉, data cache를 사용할 수 없게 설정되어 있다. 즉, 단순히 core 내부에서는 Instrcution Cache만을 사용하여야 하므로 Instruction fetch와 Branch prediction에 관련된 정보만 참조할 수 있다. Xill_DCacheDisable()함수의 선언을 보면, 만약 chip 내부에 L2Cache가 존재하게 된다면, L2 Cache에 flush하여 대체적으로 D-Cache대신 L2 Cache를 사용할 수 있다. 그러나 L2캐시는

보조캐시의 역할으로, APU내부에서 먼저 L1캐시를 거치고 L1 Cache와의 호환성을 맞춰주기 위한 SCU 패널을 지나 L2 Cache에 도달하게 되므로, L1캐시에서 직접 데이터를 가져오는 것 보다 더 많은 시간이 걸리게 된다.

따라서, L1 Cache안의 D-Cache를 직접 사용할 수 있는 D-Cache Enable 상황에서 더 고속의 동작이 가능하다.

2) Optimize level change

1과 2의 차이는 Optimize level이 O1에서 O3로 변화된 것에 있다.

먼저, O1의 경우 전혀 Optimize되지 않은 코드이다. SDK의 Disassembly를 통해 assembly코드를 자세히 보면, memory Write의 경우 1iteration 당 load명령어를 총 5번, store명령어를 총 2번 진행한다. 또한, read&check 부분역시 1iteration 당 load 및 store명령어를 6번, 2번 진행한다. 따라서, 해당 코드를 실행시키면 각각 1023번 iteration을 하므로 최종적으로 1023*(7+8)에 해당하는 load/store명령어가 진행되게 된다. Cortex A9은 {LDR reg,[reg,imm]}, {STR reg,[reg,imm]}의 경우 각각 2cycle이 걸리기 때문에 많은 load/store명령어가 존재 할 경우 속도가 저하되게 된다. 또한, 각 명령어에서 참조하는 메모리의 주소들이 연속적이지 않을 수 있다. 예를 들어, pattern을 저장해 둔 주소와 pattern을 저장해야할 DDR, OCM의 주소가 연속적이지 않기 때문에 cache에서 miss가 발생될 확률이 높다. 만일, cache에서 miss가 발생하게 된다면 data를 main_memory에서 write를 해야하므로 추가적으로 더 많은 시간이 소요되게 된다.

반면에, O3의 경우 movw, movt를 이용해 r1~r3까지의 register에 DDR, OCM에 해당하는 시작주소, 끝주소의 immediate값을 넣어주고 해당하는 값에 load 혹은 store을 1번 진행하므로 write시에 1번의 store, read&check시에 load를 1번 진행한다. 해당코드 역시 각각 1023번 iteration을 진행하므로 최종적으로 1023*2번의 load/store Inst.가 실행된다. O1과 비교하여 많은 수의 Instrution 수가 줄어들게 되어 속도가 빨라지게 된다. 또한, 참조하는 메모리 영역이 계속 순차적으로 증가하여 cache에서 hit될 확률이 높아져 이 역시 속도 증가에 영향을 끼쳤다고 생각한다.

3) OCM/ DDR의 차이

좀 더 자세히 들여다 보면, memtext_example과 memtest_0의 경우 DDR, OCM에 load/store의 차이가 있다. OCM은 APU 내부에 존재하고, DDR은 APU외부에 존재하여 data를 주고받는데 시간이 덜 걸리게 된다. 즉, 같은 1), 2) 조건하에서 조금 더 빠르게 동작한다.

하지만, memtest1, memtest2의 경우는 word단위가 아닌 half_word단위로 OCM에 저장되게 되며, 특히, memtest2의 경우 word단위로 주어진 offset을 halfword 단위로 변경하여 각 메모리에 저장하게 되므로 시간이 더 오래걸리게 된다.