안녕하세요, 저는 임준상이라고 합니다. 다음에는 과제 설명하도록하겠습니다.

먼저 실행 결과를 보여드리겠습니다. 제가 쓴 debugging macro는 debugM이라고 합니다. sorting program에서 마음대로 넣어서 실행하겠습니다. (设置breakpoint 然后看register) 여기 register의 값은 이렇게 되어있고 이 debugM을 실행한 후에 teraterm에서 register의 정보를 print했습니다. 서로 비교해서 완전 같습니다. 그리고 program이 sorting을 끝난 후 결과도 debugM의 영향을 얻지 않았습니다. debugM을 다른 곳에서 넣어도 같습니다.(再来一次)

다음은 코드 설명입니다. 먼저 r0-r12를 stack에 두 번 push합니다. 첫 번째 push는 이 macro 끝이 부분에 각 register을 회복하기 위해 하고 두 번째 push는 다음 코드에서 register의 값을 하나씩 하나씩 pop해서 print하기 위해 하는 작업입니다. 다음은 sp, lr, pc의 값을 저장해서 register을 print합니다. 일단 string을 print하고 register을 pop해서 r0에 저장하고 regPrint에(400 행) 갑니다. regPrint에서 print해야 할 register의 값을 얻어서 먼저 오론쪽으로 자리 이동하고 숫자 15와 and해서 28에서 31까지의 bit를 얻습니다. 이 숫자가 character을 전환하기 위해 48을 더하고 58보다 크면 숫자가 아니라 자모인 것을 알 수가 있어서 39를 더해서 자모로 바뀝니다. 오른쪽으로 자리 이동하는 bit 설치를 통해 이 register의 다른 부분의 값도 hex 숫자로 전환할 수 있습니다. 이 register의 값을 hex 숫자로 모두 print를 해서 위에 return합니다(23행). r1도 마찬가지로 먼저 string을 계속 print하고 pop된 값을 r0에 저장하고 자기 값을 hex 숫자로 print합니다.(91행) 원래 cpsr, pc 등 register의 값을 저장하고 있는 register을 print하기 전에 그냥 이미 print한 register에 저장하면 됩니다.(128행 sp부분) sp, lr, pc를 print 하기 전에 macro 후출하기 전의 값을 print해야 해서 먼저 전에 값으로 계산해서 print합니다.

마지막은 cpsr입니다. 마찬가지로 먼저 string을 print하고 NZCV의 값을 계산합니다.(到cpsrPrint1) 전에 오른쪽으로 이동해서 register의 각 부분의 bit을 얻는 방법과 같습니다. 값이 1이면 capital character로 print합니다. (回到上面157행) cpsrPrint1을 끝난 후에 IF를 print합니다. 또 마찬가지로 I bit와 F bit를 얻어서 값이 1이면 capital character로 print합니다. 다음은 cpsr instruction set state를 print합니다. (239행) J bit와 T bit를 얻고 숫자와 비교해서 어느 mode인지 식별해서 print합니다. 다음 부분은 processor mode입니다.(308행) 대응 bit를 읽어서 mode를 식별하며, 또 위에 코드와 마찬가지입니다. 마지막은 regPrint를 통해 cpsr의 값을 hex 숫자로 print한후에 cpsr를 회복합니다. endmacro에서 lr과 push된 register를 모두 pop해서 macro이 끝이다.

네, 제 설명은 여기까지입니다. 감사합니다.