```
    1       AREA    LAB2, CODE, READWRITE
    2       EXPORT  div_and_mod
    3
    4   div_and_mod
    5       STMFD r13!, {r2-r12, r14}
    6
    7       ; Your code for the signed division/mod routine goes here.
    8       ; The dividend is passed in r0 and the divisor in r1.
    9       ; The quotient is returned in r0 and the remainder in r1.
   10
   11       ; r0 = dividend
   12       ; r1 = divisor
   13       ; r2 = counter
   14       ; r3 = quotient
   15       ; r4 = remainder
   16       ; r5 = dividend sign
   17       ; r6 = divisor sign
   18       ; r7 = r5 XOR r6
   19       ; RETURN r0 = quotient
   20       ;        r1 = remainder
   21
   22       ; check sign of dividend
   23       CMP     r0, #0
   24       MOV     r5, #0
   25       MOVMI   r5, #1
   26       ; if dividend < 0, convert to two's comp
   27       MVNMI   r0, r0
   28       ADDMI   r0, r0, #1
   29
   30       ; check sign of divisor
   31       CMP     r1, #0
   32       MOV     r6, #0
   33       MOVMI   r6, #1
   34       ; if divisor < 0, convert to two's comp
   35       MVNMI   r1, r1
   36       ADDMI   r1, r1, #1
   37
   38       MOV     r2, #15         ; Init counter to 15
   39       MOV     r3, #0          ; Init quotient to 0
   40       LSL     r1, r1, #15         ; lsl divisor by 15
   41       ADD     r4, r0, #0          ; Set remainder to dividend
   42
   43   loop
   44       SUBS    r4, r4, r1          ; rem = rem - divis
   45
   46       ; if(remainder < 0)
   47       ADDLT   r4, r4, r1          ; rem = rem + divis
   48       LSLLT   r3, #1              ; lsl quotient
   49       ; else
   50       LSLGE   r3, #1              ; lsl quotient
   51       ORRGE   r3, r3, #1          ; set LSB of quot = 1
   52
   53       LSR     r1, r1, #1          ; right shift divis
   54       SUBS    r2, r2, #1          ; decrement counter
   55       BPL     loop                ; branch if count >= 0
   56
   57       ADD     r0, r3, #0          ; set quot to r0
   58       ADD     r1, r4, #0          ; set remain to r1
   59
   60       EOR     r7, r5, r6
   61       CMP     r7, #1
   62       ; if dvnd != dvsr, convert answer to two's comp
   63       MVNEQ   r0, r0
   64       ADDEQ   r0, r0, #1
   65
   66       LDMFD r13!, {r2-r12, r14}
   67       BX lr       ; Return to the C program
   68
   69
   70       END
   71
   72
```