ICS HW6

PB22111679 孙婧雯

T1

该程序的结束条件是 `R2=0`，`R2` 初始时保存了 ' A ' 的 ASCII 码值 65，每次循环都给 `R2` 减 1，故循环执行 65 次；`R3` 初始值为3052，则最终 $R_3 = (1+65)*65/2+3052 = 5197$。

T3

1. 该学生的本意是输出 $40+9=49$ 对应的 ASCII 字符，也就是 ' 1 '。
2. 程序不会输出正确的结果，因为在 `A` 函数中调用 `B` 之后，从 `B` return 只能返回到 `A` 内部，不能返回到主程序，`R7` 存储的地址并没有更新，程序会进入死循环。

T4 (?

PSR [15] = 1表示条件位 n 被置 1，地址寄存器

T5

1. 检查KBSR[15]，可以利用BRzp；
2. 键盘键入的值存入KBDR，处理器再将该ASCII值存入本地寄存器；
3. 在DSR[15]=1时将读取的字符存入DDR中；
4. 可能的代码如下：

```
START LDI R1, KBSR              ;检查 KBSR 的 ready 位
BRnp  START
LDI   R0, KBDR                  ;读取新字符
ECHO  LDI R1, DSR               ;检查 DSR 的 ready 位
BRzp  ECHO
STI   R0, DDR                   ;读取的字符存入输出数据寄存器
HALT
KBSR .FILL xFE00
KBDR .FILL xFE02
DSR  .FILL xFEO4
DDR  .FILL xFE06
```

T6

1. 补全：

(1) WAIT (2) LETTER (3) nzp CONTINUE

(4) GETCHAR (5) -65 (6) 17

2. `ADD` 写四遍的目的是将输入的字符转化为原来 ASCII 值的 16 倍以便运算 HEX。

3. 没有必要清零，因为 `GETC` 本身就将字符 ASCII 码存储到 `R0`。

T7

1. 输出为 H3ll0_W0r1d
2. 18 bytes

T8

1. 若KBSR [15] 已经为 0，可能会重复读取到原来已经读取过的字符；
2. 若KBSR [15] 为 1，上次的字符还没有被处理，直接写入新的字符可能会覆盖掉上一个字符，导致数据丢失；
3. 问题1更可能出现，KBSR [15] 为 0 的概率比为 1 的时间更大。

T9

输出为 F ！（F + 空格 + ！）（`Mem[x3009]  ASCII(32)  ASCII(33)`）