Architetture dei Sistemi Di Elaborazione **Laboratory 8**	Delivery date: 14th December 2023 Expected delivery of lab_08.zip must include: - zipped project folders for Exercise1, Exercise2 - this lab track completed and converted to pdf format.

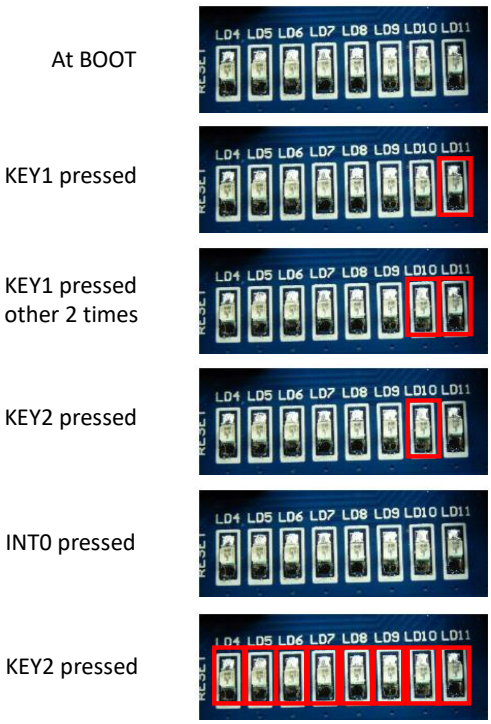
Exercise 1)

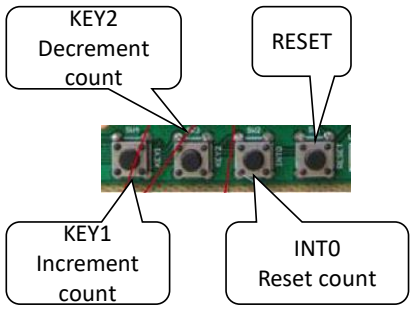
- Download the **template project** for Keil µVision "***03_sample_BUTTON_LED***" from the course material.

Implement an 8-bit "signed counter" by using LANDTIGER board; the software permits to use buttons to update a counting value which could be either positive or negative, and the LEDs to show the current value. By first using emulation capabilities (later, move your firmware on the board), please implement the following functionalities:

- increment a variable every time the button KEY1 is pressed
- decrement when KEY2 if pressed (in case, go to negative number)
- reset the count when INT0 is pressed.

LEDs are showing the current count in a binary, 2's complement representation.





HINT: It could be useful to use a global variable to keep the information about turned ON LEDs. For example, using a variable called "`char led_value`", already available in the project.

Q1: Do you observe any unexpected behaviour on the board with respect to SW emulation? Please describe.

Sulla board fisica possiamo notare che certe volte la funzione LED_Out, alla pressione singola del bottone KEY1 o KEY2, viene eseguita più di una sola volta, portando a risultati non coerenti a quelli desiderati

Exercise 2) Experiment the SVC instruction.

- Download the **template project** for Keil µVision "**01_SVC**" from the course material.
- You must execute the debug of the project on the LandTiger Board.

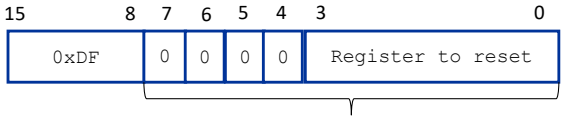
2.1) Write, compile, and execute a code that invokes an `SVC` instruction in the reset handler.

You must set the control to `user mode(unprivileged)`.

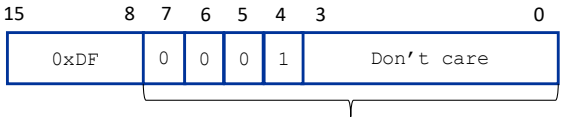
By means of invoking a SuperVisor Call, we want to implement a RESET, a NOP and a MEMCPY functions. The MEMCPY function is used to copy a block of data from a source address to a destination address and return information about the data transfer execution.

In the handler of `SVC`, the following functionalities are implemented according to the **`SVC` number**:

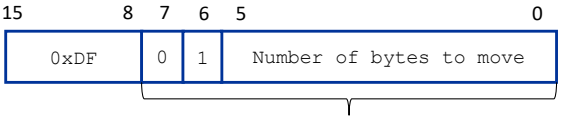
1. 0 to 7: RESET the content of register R?, where ? can assume values from 0 to 7
2. 8 to 15: NOP (no operation)
3. 64 to 127: the `SVC` call must implement a MEMCPY operation, with the following input parameters and return values:
    - the 6 least significant bits of the SVC number indicates the number of bytes to move.
    - source and destination start addresses of the areas to copy are 32 bits values passed through stack.
    - by again using the stack, it returns the number of transferred bytes.



SVC number for Register RESET instruction (0 to 7)



SVC number for NOP (8 and 15)



SVC number for MEMCPY (>= 64 and <=127)

Example: the following `SVC` invokes MEMCPY from a given source to a destination

```
LDR  R0, SourceStartAddress
LDR  R1, DestinationStartAddress
PUSH R0
PUSH R1
SVC  0x48; 2_01001000 binary value of the SVC number
POP  R0
```

Q1: Describe how the stack structure is used by your project.
All'inzio essendo in user mode useremo lo stack pointer psp, dopo la chiamata dell' svc lo stack utilizzato diventa il msp, per poi tornare al psp una volta che la chiamata svc si è conclusa

Q2: What need to be changed in the `SVC` handler if the access level of the caller is privileged? Please report code chunk that solves this request.
Se il caller avesse avuto i privilegi lo stack pointer avrebbe puntato direttamente al msp e di conseguenza non sarebbe servito richiamare il psp tramite il comndo MSR per ritrovare le informazioni che dovevano essere passate dal chiamante alla chiamata svc

Q3: Is the encoding of the SVC numbers complete? Please comment.
No, non è completa dato che mancano i numeri dal 16 al 63 e dal 128 al 255