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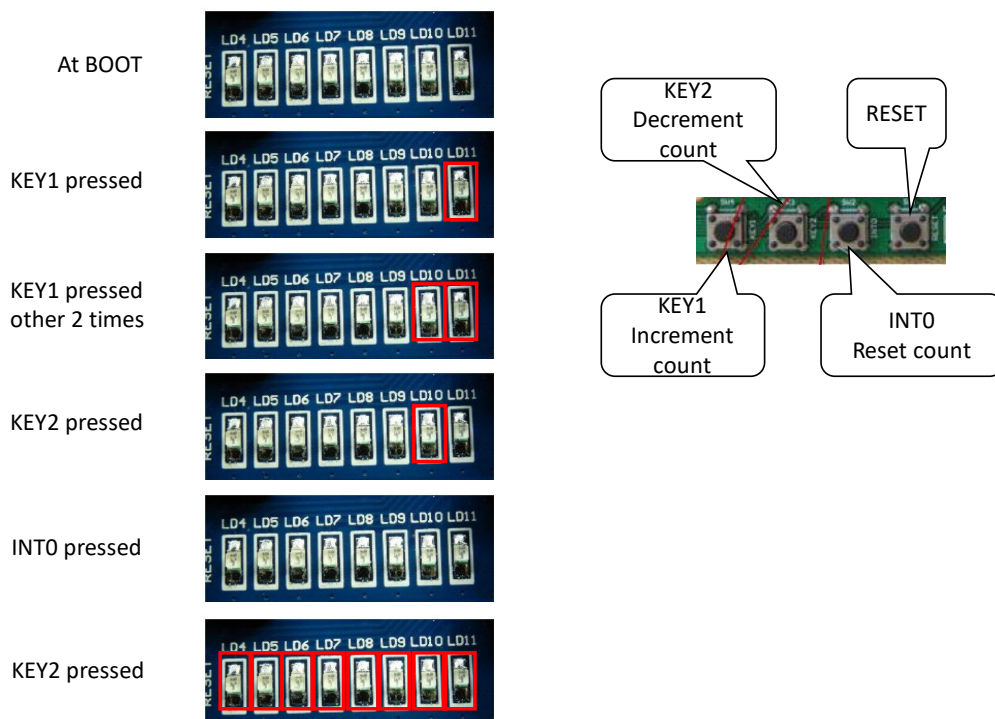
HINT: It could be useful to use a global variable to keep the information about turned ON LEDs. For example, using a variable called "`char led_value`", already available in the project.

Q1: Do you observe any unexpected behaviour on the board with respect to SW emulation? Please describe.
Si, nella scheda possiamo osservare un comportamento anomalo dei led quando vengono premuti i bottoni, questo è chiamato buttons bouncing. Il bouncing dei pulsanti è il fenomeno di oscillazione che si verifica quando si preme o rilascia un pulsante, il contatto elettrico può oscillare o rimbalzare per un breve periodo di tempo prima di stabilizzarsi. Questo fenomeno può causare letture instabili o multiple per un singolo premere o rilasciare del pulsante.

Exercise 2) Experiment the SVC instruction.
- <u>Download the **template project** for Keil µVision "**01_SVC**" from the course material.</u>
- You must execute the debug of the project on the LandTiger Board.
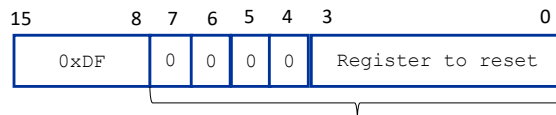
2.1) Write, compile, and execute a code that invokes an SVC instruction <u>in the reset handler</u>.

<mark>You must set the control to user mode(unprivileged)</mark>.

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**:
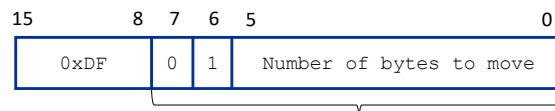1. 0 to 7: RESET the content of register R?, where ? can assume values from 0 to 7
2. 8 to 15: NOP (no operation)
3. 64 to 127: the SVC call must implement a MEMCPY operation, with the following input parameters and return values:
   - the 6 least significant bits of the SVC number indicates the number 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.

15	8	7	6	5	4	3	0
0xDF		0	0	0	0	Register to reset	

SVC number for Register RESET instruction (0 to 7)

15	8	7	6	5	4	3	0
0xDF		0	0	0	1	Don't care	
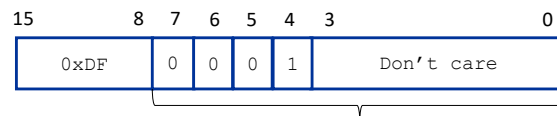
SVC number for NOP (8 and 15)

15	8	7	6	5	0
0xDF		0	1	Number of bytes to move	

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.
Lo stack utilizzato di default dalla chiamata SVC per salvare i registri R0-R3, R12, LR, PC e xPSR è il PSP. Nella procedura SVC_Handler invece l'istruzione: STMFD SP!, {R0-R7, LR} salva i registri nello stack MSP. E successivamente verrà fatta la LDMFD SP!, {R0-R7, LR}.

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.
I registry R0-R3, R12, LR, PC nel momento della chiamata SVC in modalità unprivileged vengono salvati nello stack puntato da PSP, altrimenti sarebbero salvati nello stack del MSP.

Q3: Is the encoding of the SVC numbers complete? Please comment.
No, poichè manca la codifica dei numeri da 16 a 64,