

Laboratory
8

Expected delivery of lab_08.zip must include:

- zipped project folders for Exercise1, Exercise2
- this lab track completed and converted to pdf format.

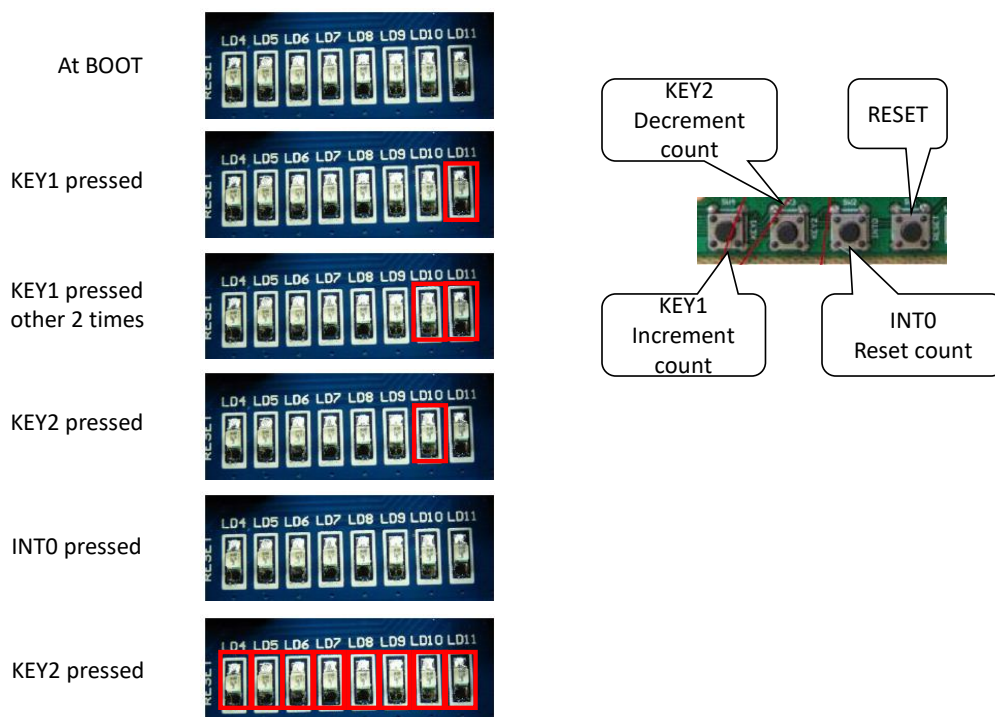
Exercise 1)

- Download the **template project** for Keil μ Vision “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 is 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, il bouncing, anche se premo una volta il tasto mi fa molte più esecuzioni di una sola (anche 5 o 6 in un colpo)

Andiamo a ricevere più di un'interruzione quando incrementiamo/decrementiamo solamente di 1.

Exercise 2) Experiment the SVC instruction.

- Download the **template project** for Keil μ Vision “**01 SVC**” from the course material.
- You must execute the debug of the project on the LandTiger Board.

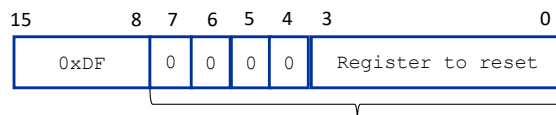
2.1) Write, compile, and execute a code that invokes an SVC instruction in the reset handler.

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

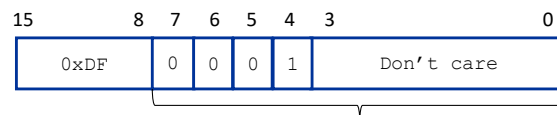
By means of invoking a SuperVisor Call, we want to implement a RESET, a NOP and a MEMCPY functions. The MEMCPY function is used to copy a block of data from a source address to a destination address and return information about the data transfer execution.

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

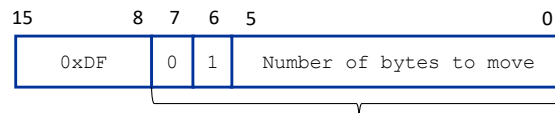
1. 0 to 7: RESET the content of register R?, where ? can assume values from 0 to 7
2. 8 to 15: NOP (no operation)
3. 64 to 127: the SVC call must implement a MEMCPY operation, with the following input parameters and return values:
 - the 6 least significant bits of the SVC number indicates the number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

Lo stack è utilizzato all'inizio del reset_handler, lo SP è inizializzato per puntare allo Stack_Mem. Con SVC_Handler, l'istruzione STMFD è stata usata per pushare in descending order il contenuto dei registri da r0 a r12 e il LR. Una volta finita l'esecuzione dell'SVC LDMFD per fare il restore di tutti i registri prima dell'esecuzione dell'SVC.

Q2: What needs to be changed in the SVC handler if the access level of the caller is privileged? Please report code chunk that solves this request.

Bisognerebbe scegliere il MSP invece del PSP, e handler mode anziché thread mode.

```
STMFD SP!, {R0-R12, LR}
```

```
MSR R1,msp
```

```
LDR R0, [R1, #24]
```

```
LDR R0, [R0, #-4]
```

```
BIC R0, #0xFF000000
```

```
LSR R0, #16
```

Q3: Is the encoding of the SVC numbers complete? Please comment.

No, non è completo.

Manca la parte che gestisce tra 16 e 63 e tra 128 a 255 (siccome abbiamo 8 bit, quindi 256 numeri rappresentabili)