|  |  |
| --- | --- |
| **Architetture dei Sistemi**  **Di Elaborazione** | Delivery date:  14th December 2023 |
| **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 µVision “***03\_sample\_BUTTON\_LED***” from the course material.

Implement an 8-bit “signed counter” by usingLANDTIGER 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.

Yes, sometimes the value reported by the LED lights is not correct, this is caused by the jumping phenomenon

**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.



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\_**01**001000 binary value of the SVC number

POP R0

Q1: Describe how the stack structure is used by your project.

In the Reset\_handler R0 and R1 are pushed into the psp (in caso of MEMCPY). A call to the SVC causes R0-R3, R12, LR, PC, xPSR to be pushed onto PSP, from the SVC\_handler R0-R12, LR are pushed into MSP and popped at the end of it, after exiting the SVC\_handler R0-R3, R12, LR, PC, xPSR are popped out of PSP and R0 is popped to save the result from the MEMCPY

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.

We would need to always use the MSP instead of the PSP

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

No, the range from 16 to 63 and 128 to 255 is unused.