

# Computer Architectures

Programming part T1.1 – June 19, 2019

PLEASE FILL THIS FORM

Student name\_\_\_\_\_

ID\_\_\_\_\_ Signature\_\_\_\_\_

Delivery time \_\_\_\_\_

Code compiles: yes ☐ no ☐

Code works: yes ☐ no ☐ partly ☐

Please read accurately:

- 1) The ARM programming part of the exam has a duration of 2 hours
- 2) You have to develop an ARM project using the KEIL  $\mu$ Vision IDE
- 3) Login in your LABINF area and use the available installation (v4.74) to edit, compile and SW debug your code
- 4) Use the provided LANDTIGER board and HW debugger to prototype your project
- 5) You are allowed to access the teaching portal page; this access will be granted by the LABINF infrastructure and any other web page access will be denied and all attempts will provoke the immediate ejection from the exam: LABINF personnel will monitor the network usage along the exam.
- 6) You can bring a single USB key and use your personal projects, material and notes.
- 7) Before the exam time ends you **MUST** upload a zipped folder of the developed project called **20190619.zip** of your project including your project in the “elaborates” section of your Computer Architecture account, in the POLITO teaching portal. Late delivery will not be considered valid and always lead rejection.
- 8) The professors will reject delivered projects that produce errors during the compiling phase; make sure your project compilation is free of errors.

## Exercise 1 (max 30 points)

You are required to implement the following functionalities on the LANDTIGER board equipped with the LPC1768 chip.

- 1) Define and populate a constant vector called **DATA\_IN** of **N** elements as a literal pool (**N** is defined as a symbolic constant and is higher than 3), every element in the vector is an integer value in the following range [-100,100] composed of 8 bits (1 byte). The literal pool should be allocated in the code memory, thus in a read only memory zone. The vector is manually initialized with values in the admissible range.
- 2) Create an empty vector in the RAM memory called **BEST\_3** composed by 3 8-bits elements.
- 3) Once BUTTON EINT1 is pressed, an assembly function that finds the highest 3 elements in the DATA\_IN vector and saves them in the in the BEST\_3 vector allocated in RAM.
  1. The prototype of the function is:  

```
int find_best_3 (int DATA_IN[], int N, int BEST_3[]);
```

which returns N at the end of the process.
- 4) Show the found values using the board LEDs in a cyclical way:
  1. Show every value *i* in the BEST\_3 vector starting from the first one (i=0), and then, replace this for the next one every 0.8s.
  2. TIMER 2 should be configured to count 0.8s and is used to drive the period of the showing process.
- 5) The showing process should be circular, then, once the last element (i=2) in the BEST\_3 vector is shown, it should start again from the first one (i=0).