Computer Architectures
Lab 2
Calculating the parity bit

1) Write an assembly program (**program.s**) for the *winMIPS64* architecture able to set the parity bit of a data array **X[]** allocated in memory.
    a. the data array **X[]** is composed of 100 elements
    b. every element **X[i]** is one byte long divided as follows:
        - $X[i]_{0-6} \rightarrow$data bits
        - $X[i]_7 \rightarrow$ parity bit
    c. consider *even parity*: the parity bit is set to 1 if the number of ones in a given set of bits (not including the parity bit) is odd
    d. the assembly program must be able to elaborate every data as presented by the following high level piece of code:

```
for (i = 0; i < 100; i++){
        if (parity_in [x[i]0..6 ])
            x[i]7 = 1;
        else
            x[i]7 = 0;
}
```

For example, if x[i] = X001101, then it becomes x[i] = $\underline{1}$001101; on the other side, if x[i] = X011101, then it becomes x[i] = $\underline{0}$011101.

2) Considering the classic *winMIPS64* architecture, enabling forwarding, and supposing that 50% of the data elements contain an odd number of ones in the data bits:
    a. calculate by hand, how many clock cycles take the program to execute?
    b. compute the same calculation using the *winMIPS64* simulator.
3) Compare the results obtained in the points 2.a and 2.b., and provide some explanation if the results are different.
4) Try to better organize the assembly instructions in the written program (i.e., reschedule the assembly instructions) in order to reduce at most the data hazards producing processor stalls.
    a. Compute again, by hand and using the simulator, the number of clock cycles required by the new version of the program.
5) Enable the *delay slot* in the *Configure* menu, is the program behavior the same? If it is necessary, try to fix the program and compare these results with the previous ones (point 4).