|  |  |
| --- | --- |
| **Computer Architectures**  **02LSEOV** | Delivery date:  **By 2:00 AM on October, 16 2024** |
| **Laboratory**  **2** | The expected delivery of lab\_02.zip must include:   * **program\_1.s** * This file, filled with information and possibly compiled in a pdf format. |

Please configure the WinMIPS64 simulator with the *Initial Configuration* provided below c):

* *Integer ALU: 1 clock cycle*
* *Data memory: 1 clock cycle*
* Code address bus: 12
* Data address bus: 12
* FP arithmetic unit: pipelined, 4 clock cycles
* FP multiplier unit: pipelined, 6 clock cycles
* FP divider unit: not pipelined, 30 clock cycles

1. Write an assembly program (**program\_1.s**) for the *WinMIPS64* architecture described before being able to implement the following high-level code:

for (i = 31; i >= 0; i--){

v4[i] = v1[i]\*v1[i] – v2[i];

v5[i] = v4[i]/v3[i] – v2[i];

v6[i] = (v4[i]-v1[i])\*v5[i];

}

Assume that the vectors v1[], v2[], and v3[] have been previously allocated in memory and contain 32 double-precision **floating-point values;** also assume that v3[] doesnot contain 0 values. Additionally, the vectors v4[], v5[], v6[] are empty vectors also allocated in memory.

**Calculate** the data memory footprint of your program:

|  |  |
| --- | --- |
| Data | Number of bytes |
| V1 | 256 |
| V2 | 256 |
| V3 | 256 |
| V4 | 256 |
| V5 | 256 |
| V6 | 256 |
| Total | 1536 |

Are there any issues? Yes, where and why? No? Do you need to change something?

|  |
| --- |
| Your answer: No non ci sono issues. |

ATTENTION: WinMIPS64 has a limitation regarding the maximum length of the string when declaring a vector. It is therefore recommended to split the elements of the vectors into multiple lines: this also increases readability.

Example: my\_fancy\_vector: .byte 8, 12 ,2, 9

.byte 49,77, 28

.byte ……

* + - Calculate the CPU performance equation (CPU time) of the above program by assuming a clock frequency of 15 MHz:

By definition:

* + - * CPI is equal to the number of clock cycles required by the related functional unit to execute the instruction (EX stage).
      * IC*i* is the number of times an instruction is repeated in the referenced source code.
    - Recalculate the CPU performance equation assuming that you can triple the speed by just one unit of your choice between the FP multiplier or the FP divider:
      * FP multiplier unit: 6 🡪 2 clock cycles

*or*

* + - * FP divider unit: 30 🡪 10 clock cycles

Table 1: CPU time by hand

|  |  |  |  |
| --- | --- | --- | --- |
|  | Initial CPU time (a) | CPU time  (b – MUL speeded up) | CPU time  (b – DIV speeded up) |
| program\_1.s | 0.16943 ms | 0.14945 ms | 0.1225 ms |

* + - Using the simulator, calculate the CPU time again and fill in the following table:

Table 2: CPU time using the simulator

|  |  |  |  |
| --- | --- | --- | --- |
|  | Initial CPU time (a) | CPU time  (b – MUL speeded up) | CPU time  (b – DIV speeded up) |
| program\_1.s | 0.16253 ms | 0.1455 ms | 0.1199 ms |

Are there any differences? If so, where and why? If not, please provide some comments in the box below:

|  |
| --- |
| Your answer: Ci sono differenze perchè nel calcolo a mano usiamo la formula teorica del CPU time che non tiene conto dell’architettura effettiva. Con il calcolo |

* + - Using the simulator and the *Initial Configuration*, enable the Forwarding option and compute how many clock cycles the program takes to execute.

Table 3: forwarding enabled

|  |  |  |
| --- | --- | --- |
|  | Number of clock cycles | IPC (Instructions Per Clock) |
| program\_1.s | 1924 | 0.2495 |

Enable one at a time the *optimization features* that were initially disabled and collect statistics to fill the following table (fill all required data in the table before exporting this file to pdf format to be delivered).

Table 4: **Program performance for different processor configurations**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Program | Forwarding | | Branch Target Buffer | | Delay Slot | | Forwarding + Branch Target Buffer | |
|  | IPC | CC | IPC | CC | IPC | CC | IPC | CC |
| program\_1.s | 0.2495 | 1924 | 0.197 | 2440 | 0.205 | 83 | 0.253 | 1896 |

1. Using the WinMIPS64 simulator, validate experimentally the Amdahl’s law, defined as follows:



1. Using the program developed before: **program\_1.s**
2. Modify the processor architectural parameters related to multicycle instructions (Menu🡪Configure🡪Architecture) in the following way:
   * + - 1. Configuration 1

Starting from the *Initial Configuration*, change the FP addition latency to 3

* + - * 1. Configuration 2

Starting from the *Initial Configuration*, change the FP multiplier latency to 4

* + - * 1. Configuration 3

Starting from the *Initial Configuration*, change the FP division latency to 10

Compute both manually (using the Amdahl’s Law) and with the simulator the speed-up for any one of the previous processor configurations. Compare the obtained results and complete the following table.

Table 5: **program\_1.s speed-up computed by hand and by simulation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Proc. Config.**    **Speed-up comp.** | Initial config.  [c.c.] | Config. 1 | Config. 2 | Config. 3 |
| **By hand** | 2243 | 1.0424 | 1.0463 | 1.33433 |
| **By simulation** | 2438 | 1.0435 | 1.0554 | 1.35595 |