Chapter II
CPU

2 CPU

Objectives

The exercises in this chapter will demonstrate how the CPU provides support for multitasking operating systems and how its performance can be improved.

Exercise 11.

The following code snippet in the assembly language of the CT computer is part of the code of the task 1 shown in figure 2.1. The dotted lines show statements that are hidden.

```
 1  ...
 2  MOVL R0, 1
 3  MOVH R0, 0
 4  MOVL R1, 64h
 5  MOVH R1, 96h
 6  MOV  [R0], R1
 7  MOVL R2, 20h
 8  MOVH R2, 72h
 9  MOV  [R2], R0
10  MOV  [R1], R2
11  STI
12  ...
```

Figure 2.1 shows the address ranges used by the tasks and the operating system. If we want the CPU of the CT to support a modern operating system, which of the above instructions will generate problems? And why? Answer the number of the instruction and its reason.

```
Instruction 6.- Access to the IVT, which belongs to the OS.
Instruction 10.- Access to the memory range of another task.
Instruction 11.- Privileged instructions.
```

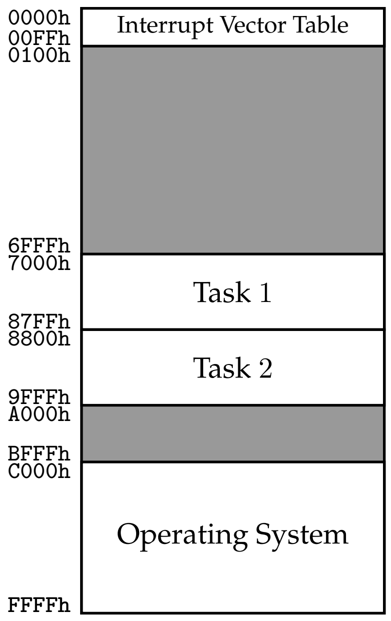

Figure 2.1: Tasks and operating system in memory.

Exercise 12.

The following programs, (a), (b), and (c), exemplify the deficiencies shown by the CPU of the CT to support multitasking operating systems.

Each snippet can be seen as a task that is loaded in memory from the address 4000h. In addition to the task there is another task loaded in memory from the address 2000h, as well as a multitasking operating system loaded from the address C000h.

(a)

```
ORIGEN 4000h
INICIO main
  .PILA 30h
  .CODIGO
main:
  CLI
  MOVL R1, '1'
  MOVH R1, 0
  STI
repeat:
  MOVL R2, 00h
  MOVH R2, 21h
  ADD  R1, R1, R2
  MOV  [R2], R1
  INT  10h
  JMP  repeat
FIN
```

(b)

```
ORIGEN 4000h
INICIO main
  .PILA 30h
  .CODIGO
main:
  MOVL R1, '1'
  MOVH R1, 0
repeat:
  XOR  R0,R0,R0
  MOVL R3, 80h
  MOVH R3, 0D8h
  MOV  [R3], R0
  MOVL R2, 00h
  MOVH R2, 01h
  ADD  R1,R1,R2
  INT  10h
  JMP  repeat
FIN
```

(c)

```
ORIGEN 4000h
INICIO main
  .PILA 30h
  .CODIGO
PROCEDIMIENTO min
  XOR  R7, R7, R7
  MOVH R7, 20h
  MOV  [R7], R5
  MOV  R5, R7
RET
FINP

main:
repeat:
  MOVL R5, 20h
  MOVL R7, 0CDh
  CALL min
  INT  10h
  JMP  repeat
FIN
```

What deficiencies of the CPU does each one of the above programs show? Answer the name of the program and its deficiency, if exists.

```
Program a.- Uses privileged instructions and writes in the mem-
ory range of another task.
Program b.- A memory location of the OS is accessed.
Program c.- A write operation is performed in the memory range
of another task.
```

Exercise 13.

The following listing is from a task executed on a PC running a Microsoft Windows OS. This code contains several instructions that raise exceptions when they are executed.

```
1  #include <stdio.h>
2  #include <conio.h>
3
4  void main() {
5
6    // Local variables
7    int num = -1;
8    unsigned int value = 500;
9    int result;
10         void* dir;
11
12   // Dissable interrupts
13   _asm {
14       cli
15   }
16
17   // Compute the result
18   result = value / (num+1);
19
20   // Read an I/O port
21   num = _inp(0x0240);
22
23   // Write the read value in the memory address 0xC0000000
24   dir = 0xC0000000;
25   *((unsigned int *)dir) = num;
26
27   // Enable interrupts
28   _asm {
29       sti
30   }
31 }
```

Which C or assembly instructions should be commented out in the above code for the program not to raise exceptions? Why? If there is no need to comment out any statement, answer NONE.

```
13 & 28.- (cli & sti) Assembly instructions allowed only for
the OS.
17.- result = value / (num+1); Division by zero; then, an ex-
ception is raised.
20.- _inp. I/O instruction allowed only for the OS.
24.- Attempt to access an address owned by the OS.
```

Exercise 14.

A new version of the CT has been implemented. In this new version the execution of every instruction is divided into two stages. Each stage consumes 5 ns. The first stage fetches the instruction from memory and increases in one unit the program counter (PC), whereas the second stage executes the instruction. It must be taken into account the fact that increasing the PC requires the use of the ALU. Furthermore, you must assume that the CPU has been rebooted and no instruction has been executed yet.

This CPU is connected to a memory device storing both code and data.

```
ADD R4, R3, R2
MOV R2, R4
AND R5, R2, R3
MOV [R6], R2
MOV R5, R2
XOR R6, R6, R6
```

❑ **14.1** Assuming that the two stages are run sequentially, how much time is needed to run the previous snippet?

2 stages × 5 ns × 6 instructions = 60 ns

In order to improve the performance of the CPU, its control unit is modified. Now this CPU allows the two stages to run concurrently if possible.

❑ **14.2** Draw the schedule of the execution of instructions taking the new operation mode into account. Draw the fetch stage in the first row and the execution stage in the second row.

ADD		MOV	AND		MOV		MOV	XOR	
	ADD		MOV	AND		MOV		MOV	XOR

❑ **14.3** How long does the pipelined CPU require for running the above instructions?

10 stages × 5 ns = 50 ns

The CPU is modified again to further improve its performance. An increment unit is added avoiding the ALU to perform the task of increasing the PC in the fetching stage of every instruction.

❑ **14.4** How long does the CPU require to run the above program taking this modification into account?

8 stages × 5 ns = 40 ns

Exercise 15.

The organization of the CPU of the CT has been redesigned. The execution of every instruction is divided into two stages that can run concurrently. The first stage is the instruction fetch and consumes 4 ns. The second stage is the instruction execution and also consumes 4 ns. The CPU is designed with a PC increment unit, avoiding the ALU to do this task. Thus, the ALU is only in charge of the arithmetic and logic operations.

The CPU has just been rebooted and no instruction has been executed yet.

```
MOVL R2, 05h
MOVH R2, F4h
XOR R4, R4, R4
ADD R4, R2, R1
MOV R3, [R4]
MOV [R3], R1
XOR R1, R2, R3
MOV R1, [R6]
```

❑ **15.1** Assuming that the two stages are running in a sequential mode, how long does the CPU need to run the above snippet?

$2 \times 4\,\text{ns} \times 8 = 64\,\text{ns}$

❑ **15.2** Assuming that the CPU is running now in a concurrent mode, what is the percentage of reduction in the execution time of the above snippet in the two following scenarios?

- One unique memory device for both data and code (Von Neumann architecture).
- Two memory devices for separate data and code (harvard architecture).

Von Neumann: $4\,\text{ns} \times 11$ cycles $= 44\,\text{ns}$
[(64-44)/64]$\times$100 $\approx$ 31.25% Harvard: $4\,\text{ns} \times 9$ cycles $= 36\,\text{ns}$
[(64-36)/64]$\times$100 $\approx$ 43.75%

❑ **15.3** Now, assume that the CPU is running in the concurrent mode but the PC increment unit has been disabled. Which is the percentage of reduction in the execution time of the above snippet in the two following scenarios?

- One unique memory device for both data and code (Von Neumann architecture).
- Two memory devices for separate data and code (harvard architecture).

Von Neumann: $4\,\text{ns} \times 14$ cycles $= 56\,\text{ns}$
[(64-56)/64]$\times$100 $\approx$ 12.5% Harvard: $4\,\text{ns} \times 12$ cycles $= 48\,\text{ns}$
[(64-48)/64]$\times$100 $\approx$ 25%

❏ **15.4** Which is the option that produces a lower reduction in the execution time? Why?

The CPU without PC increment unit implementing a harvard architecture. In this scenario structural hazards appear while fetching the instruction.

Think of a program where 35% of its instructions perform memory accesses for both reading and writing, 25% perform arithmetic instructions and 5% logic instructions.

❏ **15.5** Which of the following design options will provide a better performance in the aforementioned CPU?

- One unique memory device for both data and code with PC increment unit.
- Two memory devices for separate data and code without PC increment unit.

Both memory access and arithmetic-logic instructions involve structural hazards in the pipeline. Therefore, the best choice would be the one minimizing these hazards. In the first option, the structural hazards will be reduced $35\,\%$, whereas in the second option this reduction reaches $(25\,\%{+}5\,\%) = 30\,\%$. Thus, in this case, using one unique memory device for both data and code without a PC increment unit provides the best performance option.

Exercise 16.

A pipelined version of the CPU of the CT computer will be used. This CPU provides one ALU, which is used for arithmetic and logic operations, and a PC increment unit. The execution of every instruction is divided into two stages of the same duration (one clock cycle) that run in parallel:

1. Instruction fetch and PC increment.
2. Instruction execution.

The computer implements a Von Neumann architecture design: data and code are stored in one memory device. Furthermore, it is known that the clock frequency of the CPU is $2.5\,\text{GHz}$. In this CPU the following snippet is executed, where ?R4? register is properly initialized.

```
XOR  R3, R3, R3
XOR  R2, R2, R2
MOVL R2, 2
PUSH R2
repeat:
CMP  R2, R3
BRZ  exit
MOV  [R4], R2
INC  R4
DEC  R2
JMP  repeat
exit:
INC  R7
NOP
```

❑ **16.1** How long does it take the execution of any instruction in this computer?

```
Every instruction requires two clock cycles for the two stages
to be run: 2 ×1/2.5 = 0.8 ns.
```

❑ **16.2** In the long term, and depending on pipeline conditions, what is the minimum and the maximum time taken by the CPU to complete the execution of an instruction?

```
The time taken by the CPU to execute instructions is always the
same: 2 clock cycles. When the pipeline is not stalled the CPU
finishes one instruction per clock cycle, 0.4 ns. Conversely, if
the pipeline is stalled the CPU finishes an instruction each
two clock cycles, 0.8 ns.
```

❑ **16.3** How many instructions per second can this CPU execute in the worst working conditions?

```
1250 MIPS.
```

❑ **16.4** If no restriction applies to this CPU, which will be the maximum theoretical number of instructions per second?

```
2500 MIPS.
```

❑ **16.5** How much time is required to execute the above code fragment?

```
1/2,5 × 27 = 10.8 ns. Twenty instructions are executed (21 stages);
the pipeline is stalled 6 times, 3 due to memory accesses (PUSH
and MOV), and 3 due to control flow (JMP and BRZ).
```

❑ **16.6** What is the CPI of this CPU for the above snippet?

```
20 instructions are executed (4 + 6 × 2 + 2 + 2). Furthermore, the
snippet requires 25 clock cycles to be executed (as computed in
the previous question). Thus: CPI= 27/20 = 1.35.
```

❑ **16.7** What will be the number of clock cycles consumed in the execution of the above snippet if the architecture of the computer provided two separated memory devices for data and instructions?

```
24 clock cycles. ?PUSH R2? and ?MOV [R4], R2? instructions would
not stall the pipeline due to their memory accesses.
```

Exercise 17.

A CPU implementing the AMD-64 architecture will be designed. First, a non-pipelined version of this CPU is designed. This CPU is able to execute 250 million instructions per second when running at its highest clock frequency. The execution of every instruction is divided into twelve stages, all of them with the same duration: one clock cycle. These twelve stages run in sequential mode.

❏ **17.1** Which is the highest clock frequency that can be used with this CPU?

```
3 GHz
```

❏ **17.2** Which is the minimum time required to execute any instruction in this CPU? Give your answer in nanoseconds.

```
4 ns
```

❏ **17.3** How many instructions can this CPU execute per clock cycle? Example of answer: 1.2 instructions per clock cycle.

```
1/12 instructions/cycle
```

Once the initial version of the CPU has been designed it has been modified to use the pipeline technique. The execution of the twelve stages of the pipeline can now run concurrently.

❏ **17.4** Which is the minimum required time to execute any instruction? How many instructions, at most, can be executed per second?

```
4 ns
3000 MIPS
```

❏ **17.5** What modifications —without becoming a superscalar CPU— could be done in the internal organization of the CPU in order to achieve 4000 MIPS?

```
The number of stages in the pipeline should be increased up to
16.

4000 MIPS (target) / 250 MIPS (sequential) = 16 stages
```

Exercise 18.

Let A and B be two CPU manufacturers. A has designed the $A10$ CPU that is able of executing 3000 MIPS at a clock frequency of 1 GHz. B will release the *B20-PRO* CPU to compete with A. *B20-PRO* runs at 2 GHz, implements the same instruction set of $A10$ and is an enhanced version of the non-pipelined *B20* CPU, which runs at 200 MHz and requires 2 clock cycles to execute any instruction.

❏ **18.1** Which is the average time required to execute an instruction in the *B20* CPU?

```
2 cycles × 5 ns/cycle = 10 ns
```

❏ **18.2** The *B20-PRO* CPU is obtained by dividing the execution of the instructions in N stages of one cycle length (2 GHz) running concurrently. Which is the value of N?

```
N = 10 ns / 0,5 ns = 20
```

❑ **18.3** Which CPU does provide a better performance, $A10$ @ 1 GHz or *B20-PRO* @ 2 GHz? Why?

```
The B20-PRO CPU can execute one instruction per cycle since
it is a pipelined CPU. Its clock frequency is 2 Ghz; thus, it
achieves 2000 MIPS, less than the 3000 MIPS achieved by A10.
Therefore, the A10 CPU provides a better performance even
though its clock frequency is lower than the B20-PRO clock
frequency.
```

❑ **18.4** Is the $A10$ CPU only a pipelined CPU or is it also a superscalar CPU? Why?

```
It is a superscalar CPU since it can execute more than one in-
struction per clock cycle. It can execute (3000/1000) = 3 in-
structions per clock cycle.
```

Exercise 19.

A CPU running in a non-pipelined mode can execute 100 million instructions per second. The execution of every instruction requires completing 5 stages sequentially. Each stage consumes one clock cycle. Giving these features of the CPU, answer the following questions.

❑ **19.1** Which is the maximum frequency of this CPU in the non-pipelined mode?

```
100 MIPS × 5 cycles/instruction = 500 MHz
```

❑ **19.2** How much time does the execution of an instruction require? Answer in nanoseconds.

```
1 s / 100 MIPS = 10 ns/instruction
```

❑ **19.3** How many instructions can be executed per clock cycle?

```
1 instruction / 5 clock cycles = 0,2 instructions/cycle
```

A new CPU with the same instruction set but with a clock frequency of 1 GHz is used. This CPU uses a 12-stage pipeline, where the stages can run in parallel. Each stage consumes one clock cycle.

❑ **19.4** Which is the maximum amount of instructions per second that this CPU is able to execute?

```
1000 MIPS
```

❑ **19.5** How long does the execution of any instruction in this new CPU require? Answer in nanoseconds.

12 clock cycles $\times(1/(1 \times 10^9))$ ns/clock cycle = 12 ns

❏ **19.6** How many instructions per clock cycle can this CPU execute?

```
1 instruction per clock cycle.
```

The above values computed for the new CPU are valid for the most favorable conditions. However, when real working conditions appear, such as conditional branches, interrupts and exceptions, the pipeline is stalled. In these examples the following instruction to be executed is not the following instruction coded in memory and previously pointed by the program counter.

❏ **19.7** Assuming that when a hazard appears in the pipeline all the stages must be cleared, how long does the execution of the instruction after the branch of a service call require?

12 cycles $\Rightarrow$ 12 ns

❏ **19.8** Assuming that $20\,\%$ of the instructions executed by the CPU correspond to the above instructions, how many instructions (in average) would this CPU execute in each clock cycle?

$1 \times 0,8 + 1/12 \times 0,2 = 0,8167$ instructions/clock cycle

❏ **19.9** How many instructions could the CPU execute taking these working conditions into account? Answer in MIPS using two decimal digits.

0,8167 instructions/clock cycle $\times$ 1 GHz = 816.7 MIPS

❏ **19.10** Repeat the computations for the last two questions taking into account that the percentage of instructions that require "clearing" the pipeline is now $30\,\%$.

```
0,725 instructions/clock cycle
725 MIPS
```

Exercise 20.

The operating system takes the control of the system when a system service is called, an interrupt is requested, or an exception is raised. Identify which one of the above operations occurs in the following situations: Writing a file in disk; mouse movement; missing machine code of an instruction in memory; dividing by zero; receiving a packet from the network; reading the system time.

```
System service: Writing a file in disk; reading the system
time. Interrupt: Mouse movement; receiving a packet from the
network. Exception: Missing machine code of an instruction in
memory; dividing by zero.
```

Exercise 21.

Which of the following statements are TRUE? You may answer NONE if you think all of them are false.

A) The pipeline technique divides the execution of instructions in several stages, and these stages run in a parallel mode to increase the performance.

B) When a service call, an interrupt or an exception occurs, a handler stored in the address space of the task is executed.

C) Non-maskable interrupts are managed regardless of the value of the interrupt *flag*.

D) Superscalar is a term used to describe CPUs that replicate internal components, and thus, are able to work over several instructions in each stage of the pipeline.

E) A CPU running in user level privilege can gain access to any memory addresses of the computer.

```
A, C & D
```