**ALCOHOLIMETRO**

Isabella Caldas e Isabela Acevedo

**Profesor**: Maribell Sacanamboy y José Oliden Sanchéz

Pontificia Universidad Javeriana Cali

Arquitectura de computadores

Santiago de Cali

2019-1

Contenido

[**1.** **Alcoholímetro** 1](#_Toc10185964)

[1.1 Especificación: 1](#_Toc10185965)

[1.2 Multiciclo 1](#_Toc10185966)

[2. Diagrama de flujo y máquina de estados 2](#_Toc10185967)

[2.1 Diagrama de flujo 2](#_Toc10185968)

[2.2 Máquina de estados detallada hecha a mano. 3](#_Toc10185969)

[3. Diagrama estructural 9](#_Toc10185970)

[3.1 Estándar (Modificado según nuestra multiciclo) 9](#_Toc10185971)

[4. Descripción y formato de instrucciones y registros 9](#_Toc10185972)

# **Alcoholímetro**

## Especificación:

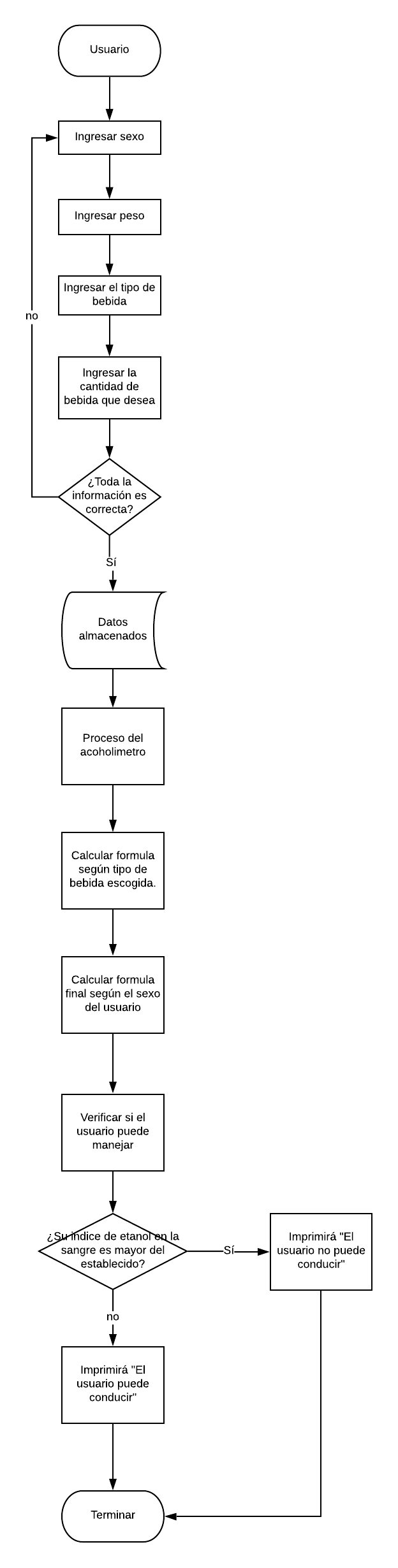
El sistema de alcoholímetro es un sistema en el cuál se puede medir la cantidad de alcohol en la sangre, el alcoholímetro indica si una persona es apta de conducir o no. El alcoholímetro determina esto dependiendo de la cantidad de etanol que tiene en el cuerpo, si este supera un nivel establecido de alcohol que debe tener en la sangre para no poder conducir, se le anuncia que no es apto para conducir. El alcoholímetro mide de manera distinta dependiendo el género de la persona, si es mujer la fórmula para medir es la siguiente: gramosAlcohol/(pesoPersona\*0.6) y si es hombre, la fórmula es: gramosAlcohol/(pesoPersona\*0.7) si el resultado al aplicar la fórmula es mayor a 0.4, se le recomendará a la persona no conducir por seguridad.

## Multiciclo

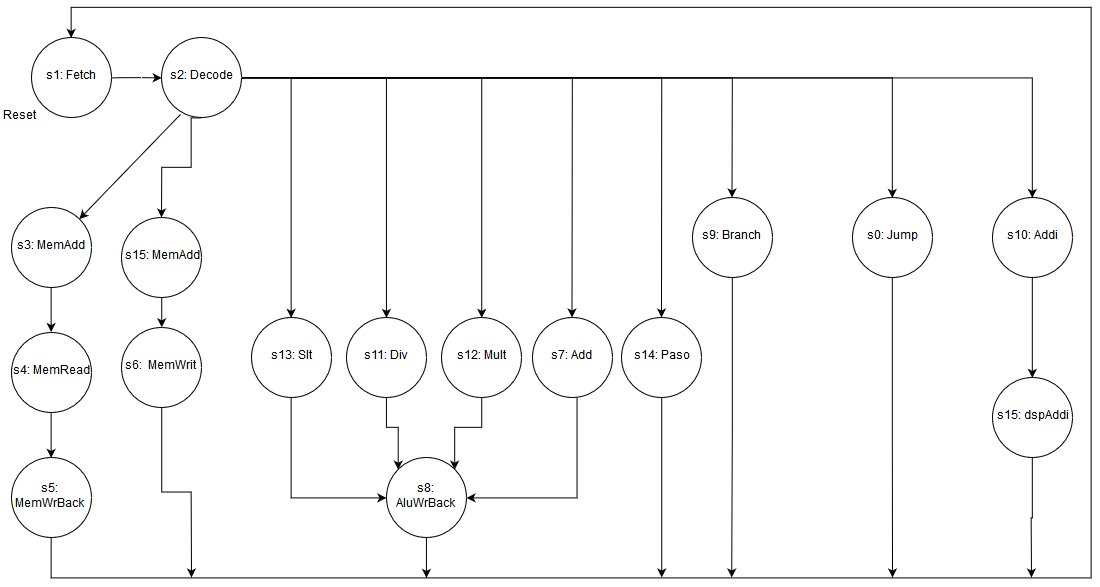
Personalmente, preferimos mantener la estructura de la multiciclo debido a que, al cambiarla, nos podría generar muchos problemas. Además, que nos recomendaron manejarla con 32 bits.

# Diagrama de flujo y máquina de estados

## 2.1 Diagrama de flujo



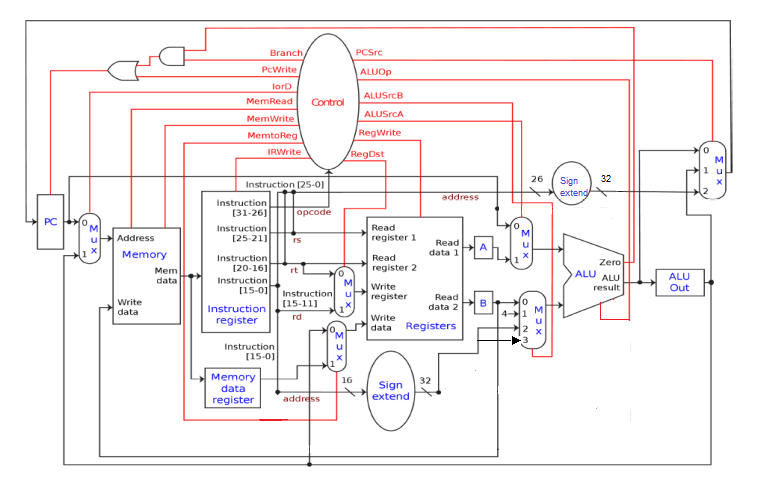
## 2.2 Máquina de estados detallada hecha a mano.



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Fetch: | |  | | --- | | pcWrite = 1 | | branch = 0 | | IorD = 0 | | memRead = 1 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 1 | | PCsrc = 00 | | aluOP = 000 | | aluSrcA = 0 | | aluSrcB = 01 | | regWrite = 0 | | regDst = 0 | |  | |
| Decode: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 000 | | aluSrcA = 0 | | aluSrcB = 11 | | regWrite = 0 | | regDst = 0 | |
| MemAdd: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 000 | | aluSrcA = 1 | | aluSrcB = 10 | | regWrite = 0 | | regDst = 0 | |
| memAddrSW: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 000 | | aluSrcA = 1 | | aluSrcB = 10 | | regWrite = 0 | | regDst = 0 | |
| MemRead: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 1 | | memRead = 1 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 000 | | aluSrcA = 0 | | aluSrcB = 00 | | regWrite = 0 | | regDst = 0 | |  | |
| MemWrBack: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 1 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 000 | | aluSrcA = 0 | | aluSrcB = 00 | | regWrite = 1 | | regDst = 0 | |
| MemWrite: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 1 | | memRead = 0 | | memWrite = 1 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 000 | | aluSrcA = 0 | | aluSrcB = 00 | | regWrite = 0 | | regDst = 0 | |
| Add: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 000 | | aluSrcA = 1 | | aluSrcB = 00 | | regWrite = 0 | | regDst = 0 | |
| Branch: | |  | | --- | | pcWrite = 0 | | branch = 1 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 01 | | aluOP = 001 | | aluSrcA = 1 | | aluSrcB = 00 | | regWrite = 0 | | regDst = 0 | |
| Jump: | |  | | --- | | pcWrite = 1 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 10 | | aluOP = 000 | | aluSrcA = 0 | | aluSrcB = 00 | | regWrite = 0 | | regDst = '0'; | |
| Addi: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 000 | | aluSrcA = 1 | | aluSrcB = 10 | | regWrite = 0 | | regDst = 0 | |
| Div: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 011 | | aluSrcA = 1 | | aluSrcB = 00 | | regWrite = 0 | | regDst = 0 | |
| Mult: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 010 | | aluSrcA = 1 | | aluSrcB = 10 | | regWrite = 0 | | regDst = 0 | |
| Slt: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 110 | | aluSrcA = 1 | | aluSrcB = 00 | | regWrite = 0 | | regDst = 0 | |
| Paso: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 111 | | aluSrcA = 1 | | aluSrcB = 00 | | regWrite = 0 | | regDst = 0 | |
| DspAddi: | |  | | --- | | pcWrite = 0 | | branch = 0 | | IorD = 0 | | memRead = 0 | | memWrite = 0 | | memToReg = 0 | | IRwrite = 0 | | PCsrc = 00 | | aluOP = 000 | | aluSrcA = 0 | | aluSrcB = 00 | | regWrite = 1 | | regDst = 0 | |

# Diagrama estructural

## 3.1 Estándar (Modificado según nuestra multiciclo)



# Descripción y formato de instrucciones y registros

A continuación, se expresarán las instrucciones y registros usados en MIPS para resolver el alcoholímetro.

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **MNEMONIC** | **FORMAT** | **OPERATION** |
| **Add** | add | R | R[rd]=R[rs]+R[rt] |
| **Add Inmediate** | addi | I | R[rd]=R[rs]+SignExtImm |
| **Branch On Equal** | beq | I | if (R[rs]=R[rt])  PC=PC+4+BranchAddr |
| **Set Less Than** | slt | R | R[rd] = R[rs] < R[rt] |
| **Load Word** | lw | I | R[rt]=M[R[rs]+SignExtImm] |
| **Store Word** | sw | I | M[R[rs]+SignExtImm]=R[rt] |
| **Jump** | j | J | PC=JumpAddr |
| **Move From Lo** | mflo | R | R[rd] = Lo |

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **NUMBER** | **USE** | **PRESERVED ACROSS A CALL?** |
| $zero | 0 | The constant value 0 | N.A |
| $v0-$v1 | 2-3 | Values for Function Results and Expression Evalution | No |
| $a0-$a3 | 4-7 | Arguments | No |
| $t0-$t7 | 8-15 | Temporaries | No |
| $t8-$t9 | 24-25 | Temporaries | No |
| $S0- $S7 | 16-23 | Saved Temporaries | Yes |