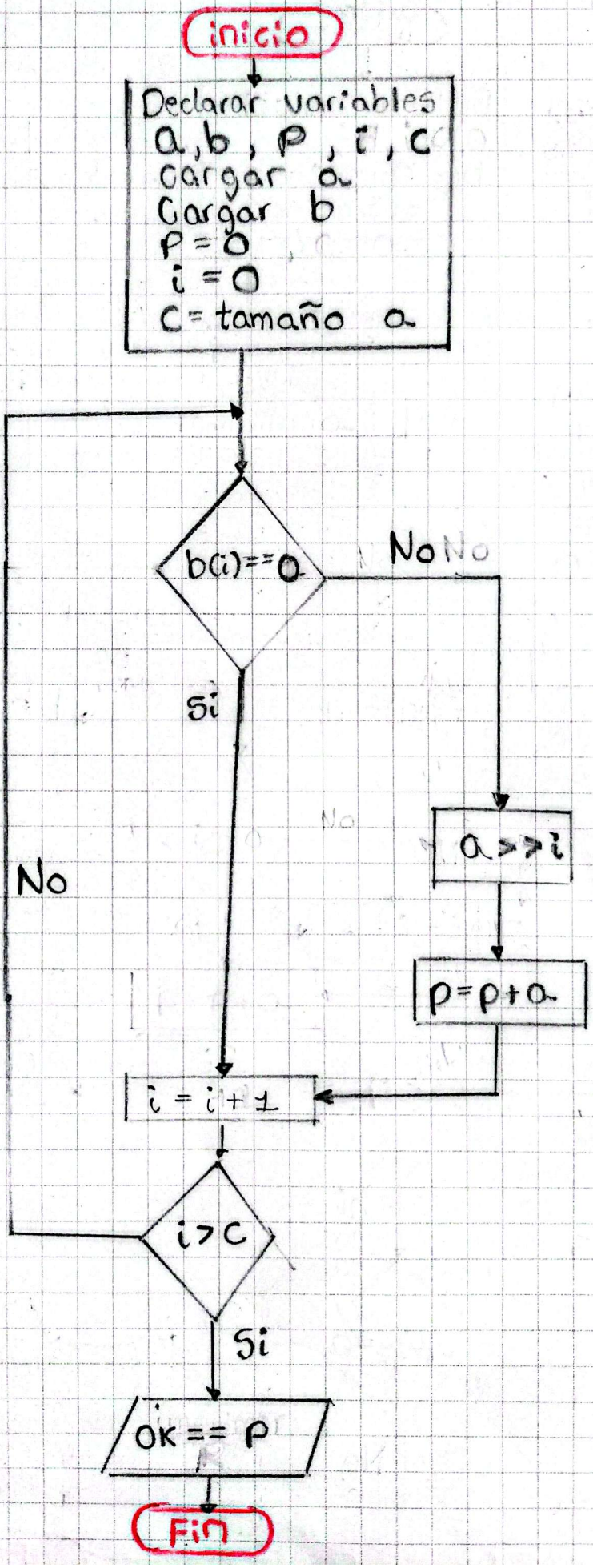
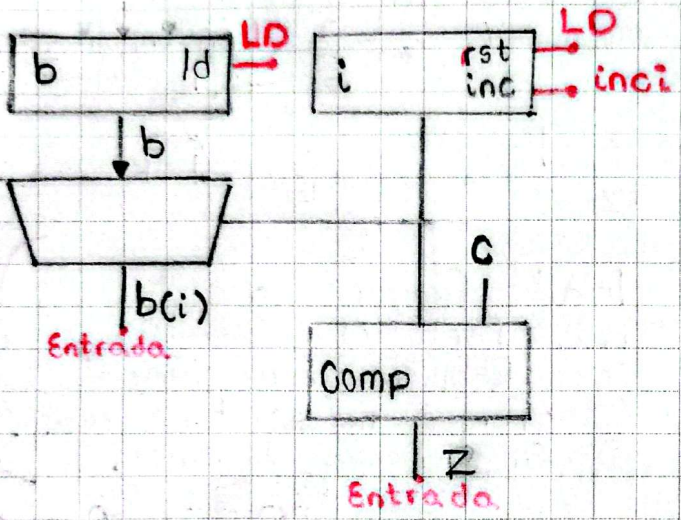
Tarea - Algoritmo multiplicación (Corregido).

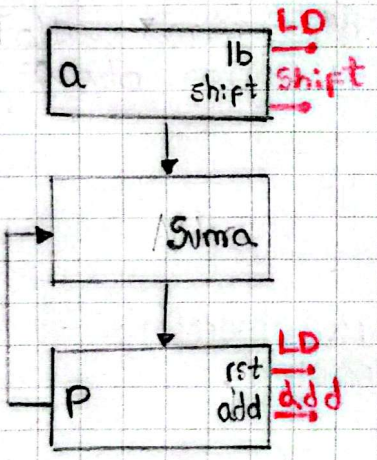


FECHA ... ___ ...

Diagrama de caja negra

Camino de datos





Maquina de estados

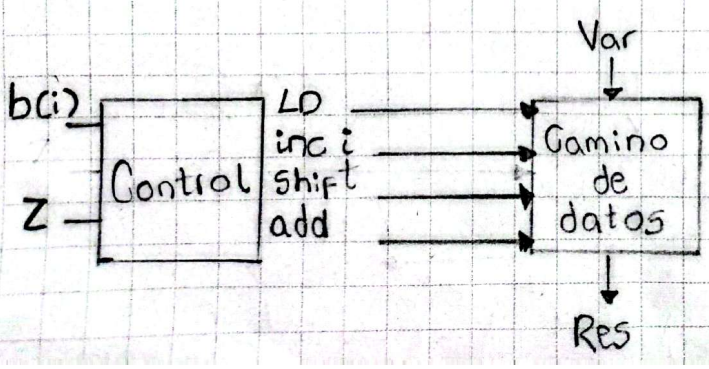
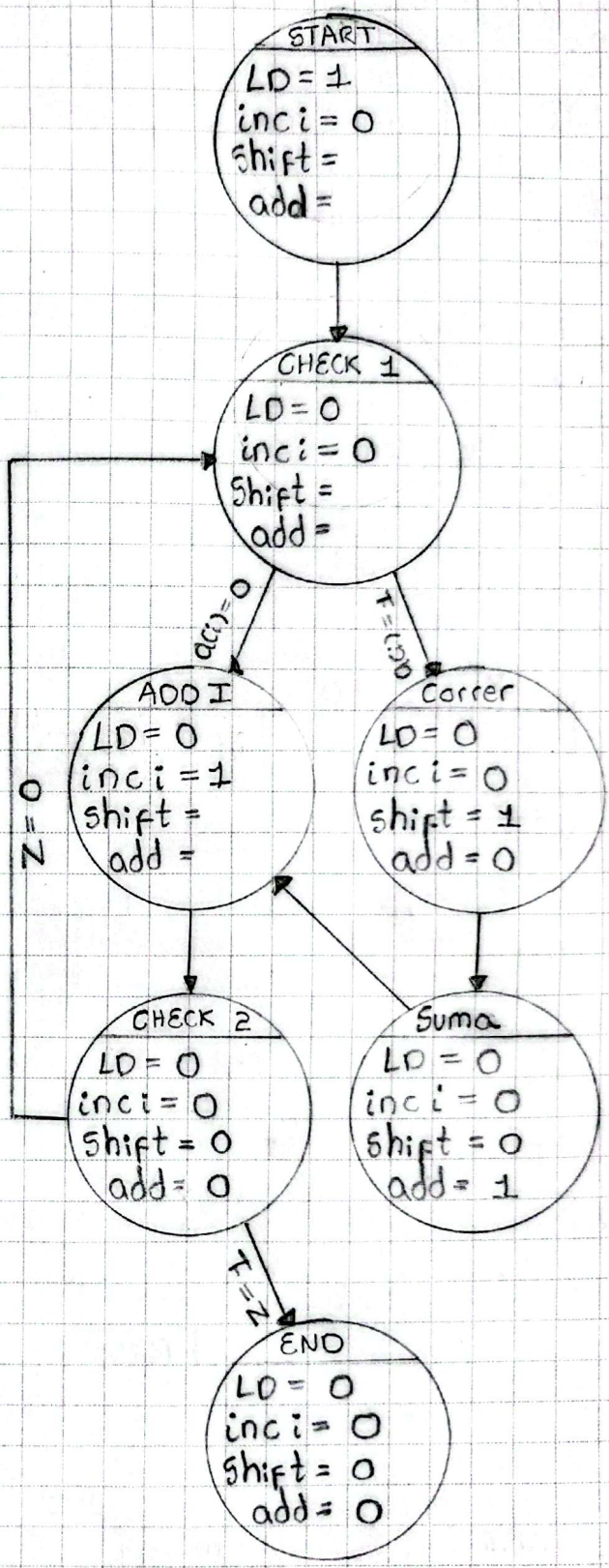
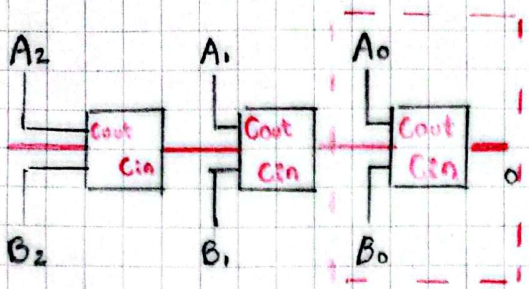


Diagrama de transición



Fecha 25-02-2026

- Se corrige la maquina de algoritmo de multiplicación.
- Estructura de sumadores



Entradas			Salidas	
Cin	A	B	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Traducción de la tabla

Suma booleana

① Sumar los S = 1 ;

$S = \bar{C}in\bar{A}B + \bar{C}in A\bar{B} + Cin\bar{A}\bar{B} + CinAB$

② Ahora mirar miniterminos para mirar negación $\bar{0}$ = Negar ;

$S = \bar{C}in\bar{A}B + \bar{C}in A\bar{B} + Cin\bar{A}\bar{B} + CinAB$

③ Factorizar:

$S = \bar{C}in(\bar{A}B + A\bar{B}) + Cin(\bar{A}\bar{B} + AB)$

④ Mirar que es cada termino

$S = \bar{C}in\,(\bar{A}B + A\bar{B}) + Cin\,(\bar{A}\bar{B} + AB)$

xor — xor negado

$S = Cin \oplus A \oplus B$ ← Equivalencia

xor xor

lenguaje que describe Hardware

```
module hadd (Cin, A, B, S)
input Cin ;
input A ;
input B ;
output C ;
always @*
  begin
    S = ~Cin ^ (A ^ B);
  end
```

se puede reemplazar

```
Case ({Cin, A, B})
  3'b000 : S = 0 ;
  3'b 001 : S = 1 ;
  3'b 010 : S = 1 ;
endcase
```

Sumador

```
endmodule
```

```
module acc (clk, A, add, rst, pp);
  input clk ;
  input [31:0] A ;
  input add ;
  input rst ;
  output reg [31:0] pp ;

  initial pp = 0 ;
```

acumulador

=>

```
always @(negedge clk)
  if (rst)
    pp = 32'h00000000
  else
    begin
      if (add) pp = pp + A
      else pp = pp
    end

endmodule
```