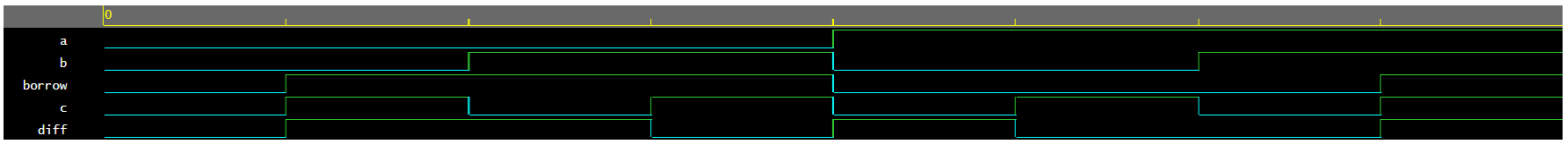
Código full-substractor usando 2 half-substractors:

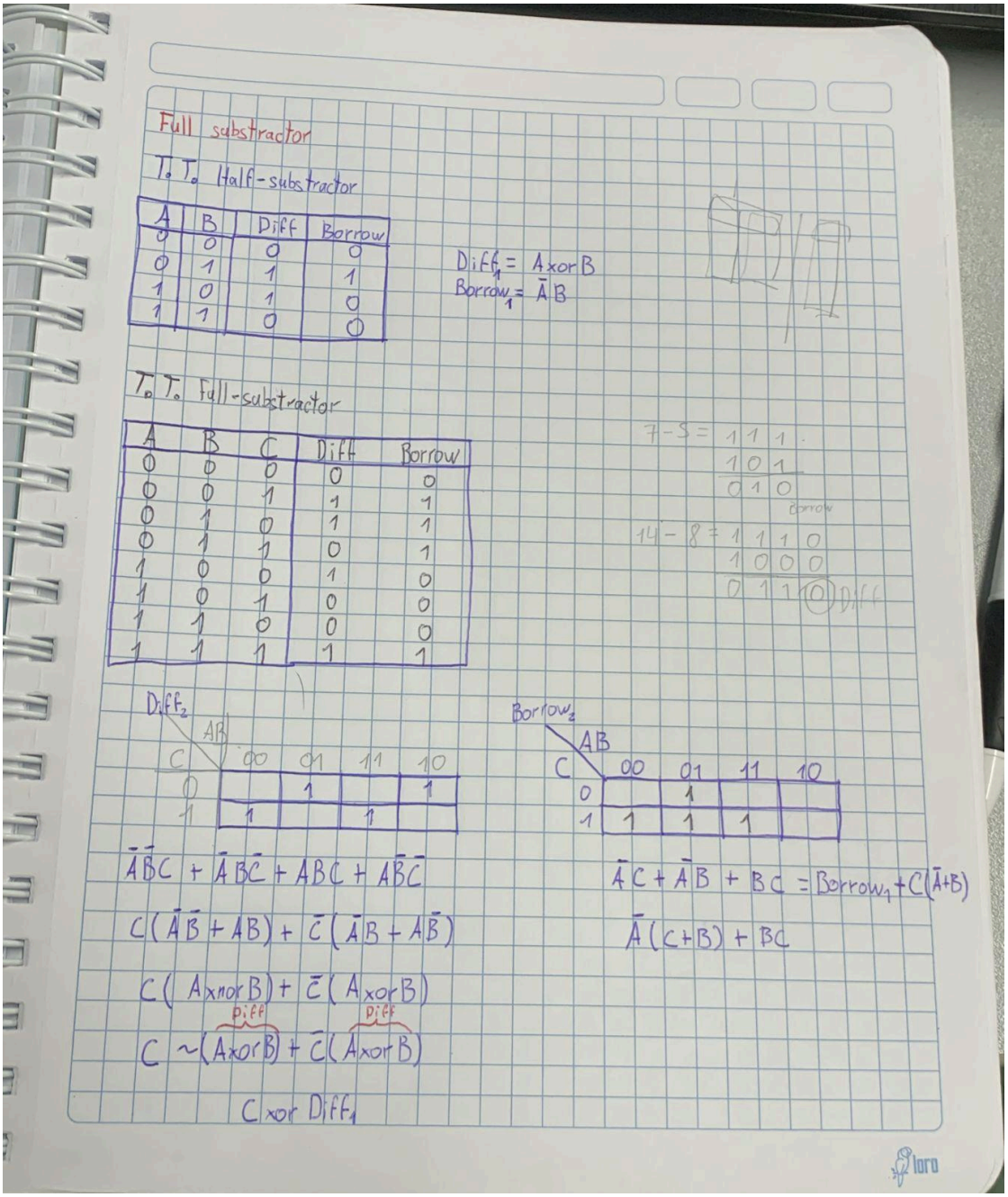
```
`timescale 1ns/1ns
module halfsubstractor_tb();
  reg a, b, c;
  wire diff, borrow;
  wire n1, n2, n3;

  fullsubstractor half_test(.a(a), .b(b), .c(c), .diff(diff),
.borrow(borrow));
  initial begin
    a = 0;
    b = 0;
    c = 0;
    #1

    a = 0;
    b = 0;
    c = 1;
    #1

    a = 0;
    b = 1;
    c = 0;
    #1
```

```
module halfsubstractor(a, b, diff, borrow);
  input a, b;
  output diff, borrow;

  assign diff = a ^ b;
  assign borrow = ~a & b;
endmodule

module fullsubstractor(a, b, c, diff, borrow);
  input a, b, c;
  output diff, borrow;
  wire n1, n2, n3;

  halfsubstractor first(.a(a), .b(b), .diff(n1), .borrow(n2));
  halfsubstractor second(.a(n1), .b(c), .diff(diff), .borrow(n3));
  assign borrow = n3 | n2;


endmodule
```

Diff= Vale 1 cuando la diferencia entre a, b y c es 1 (a - b - c = abs(1))
Borrow = Vale 1 en el momento que detecte una resta negativa (a-b <0 | b-c<0)



Apuntes:



Full substractor

T. T. Half-substractor

A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$Diff_1 = A \text{ xor } B$

$Borrow_1 = \bar{A}B$

T. T. Full-substractor

A	B	C	Diff	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

7 - 5 = 1 1 1 . / 1 0 1 / 0 1 0 (borrow)

14 - 8 = 1 1 1 0 / 1 0 0 0 / 0 1 1 0 (0) Diff

$Diff_2$

C \ AB	00	01	11	10
0		1		1
1	1		1	

$\bar{A}\bar{B}C + \bar{A}B\bar{C} + AB\bar{C} + A\bar{B}\bar{C}$

$C(\bar{A}\bar{B} + AB) + \bar{C}(\bar{A}B + A\bar{B})$

$C(A \text{ xnor } B) + \bar{C}(A \text{ xor } B)$

$C \sim(A \text{ xor } B) + \bar{C}(A \text{ xor } B)$ (Diff, Diff)

$C \text{ xor } Diff_1$

$Borrow_2$

C \ AB	00	01	11	10
0		1		
1	1	1	1	

$\bar{A}C + \bar{A}B + BC = Borrow_1 + C(\bar{A}+B)$

$\bar{A}(C+B) + BC$