# Part 1: Boolean Simplification

## Question 1:

EQ1: AB'C+A'BC'+A'B'C
EQ2: ABC+AB'C+A'BC'+A'B'C

**Simplifying EQ1: A'B + AB'C + ABC'**
B'C(A+A') + A'BC'
B'C1+A'BC'
B'C+A'BC'

**Simplifying EQ2: A'B + ABC + ACD' + BCD**
B(AC + A')+ACD'+BCD
B(C + A')+ACD'+BCD
BC + BA'+ACD'+BCD
BC + BA'+ACD'

## Question 2

| A | B | C | F |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

**Sum of Products (SOP) Form:**

**Product of Sums (POS) Form:**

# Part 2: 3-input Boolean Circuit Simplification using Laws

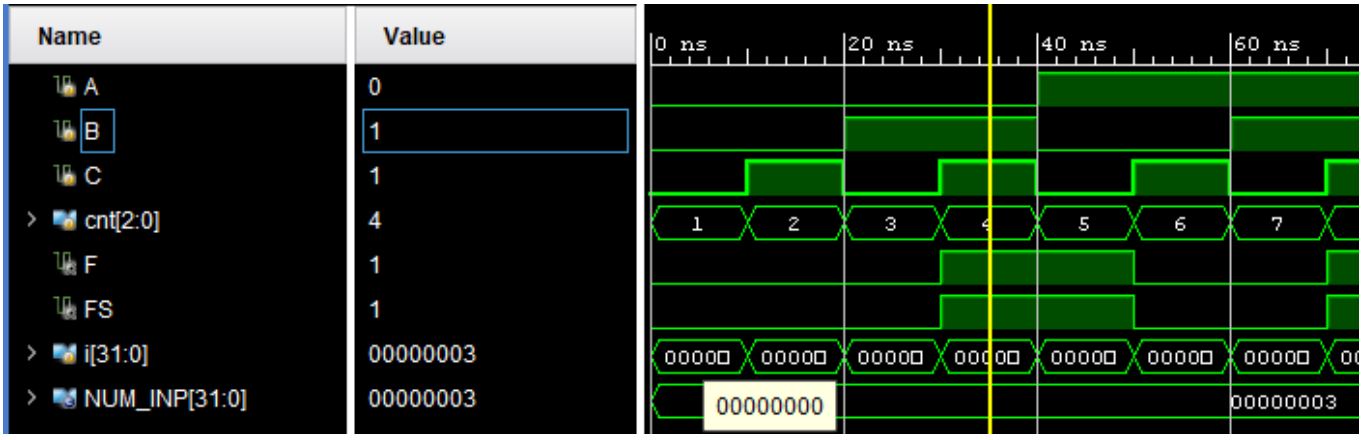
| A | B | C | F |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

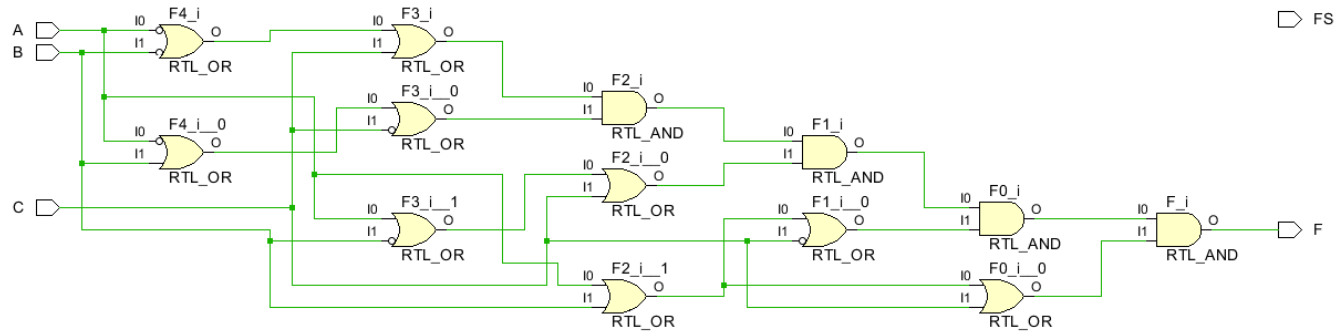
POS expression: (A+B+C)(A+B'+C')(A'+B+C)

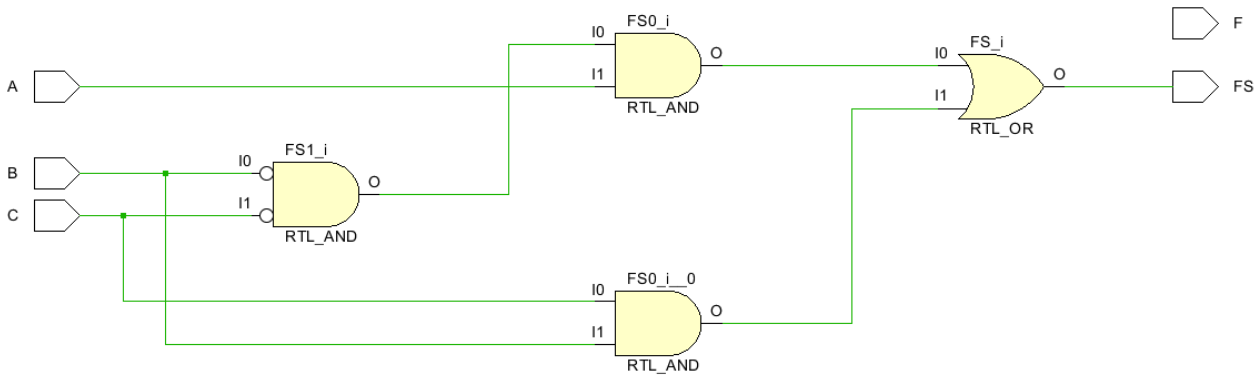
Simplified expression: AC+BC'+CB'

```
module Part_1(
    input A,
    input B,
    input C,
    input D,
    output F,
    output FS
    );

    assign F = (~A&~B&C&~D)|(~A&B&C&~D)|(~A&B&C&D)|(A&~B&C&~D);
    assign FS = (~A&C&B)|(C&~D&~B) |(C&~D&~A);
endmodule
```






# Part 3: 4-input Boolean Circuit Simplification using Laws

| A | B | C | D | F |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 |

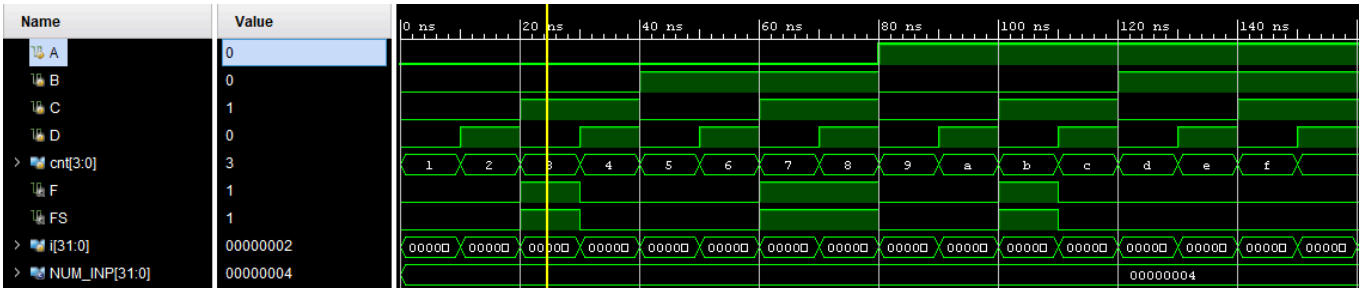
| A | B | C | D | F |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

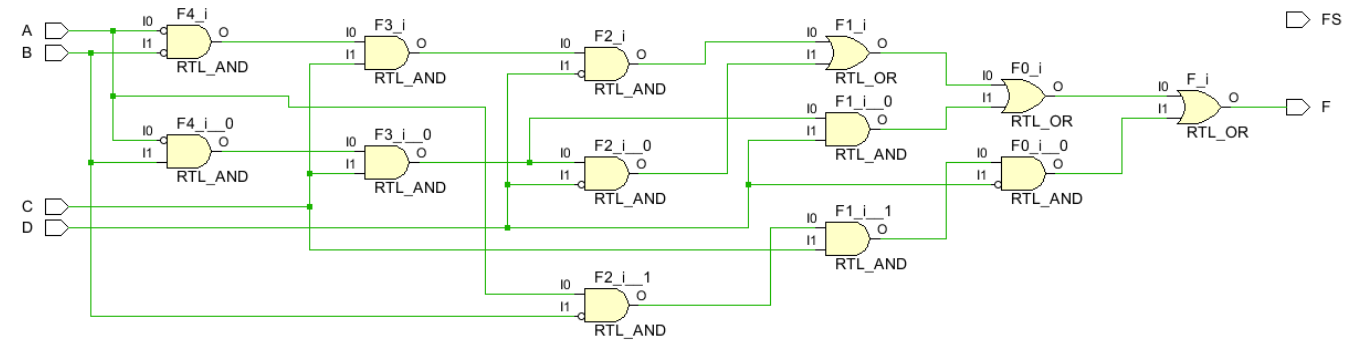
Original expression: A'B'CD'+A'BCD'+A'BCD+AB'CD'

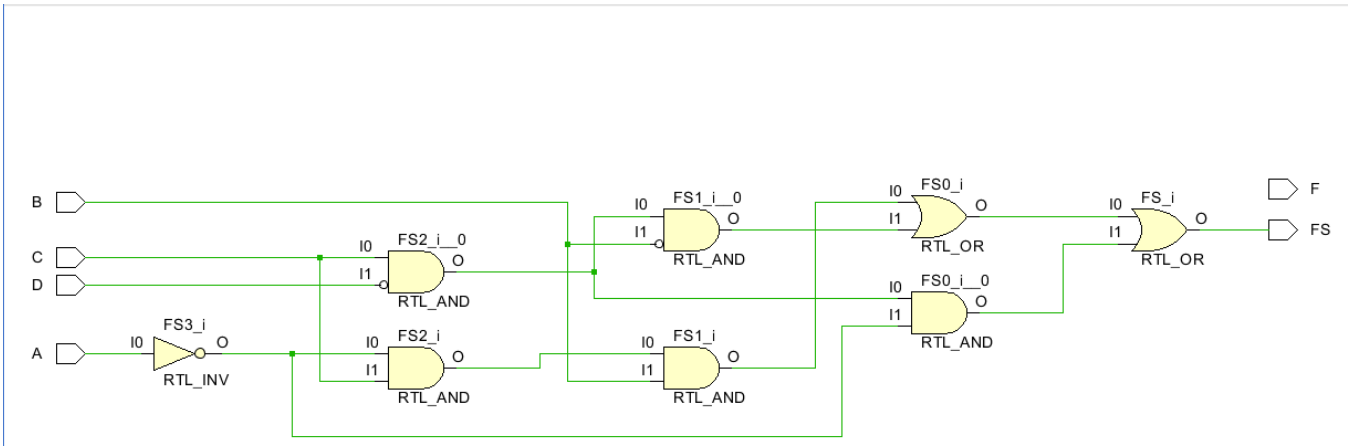
Simplified expression:

```
module Part_1(
    input A,
    input B,
    input C,
    output F,
    output FS
    );

    assign F = (~A|~B|C)&(~A|B|~C)&(A|~B|C)&(A|B|~C)&(A|B|C);
    assign FS = (~B&~C&A) |(C&B);
endmodule
```





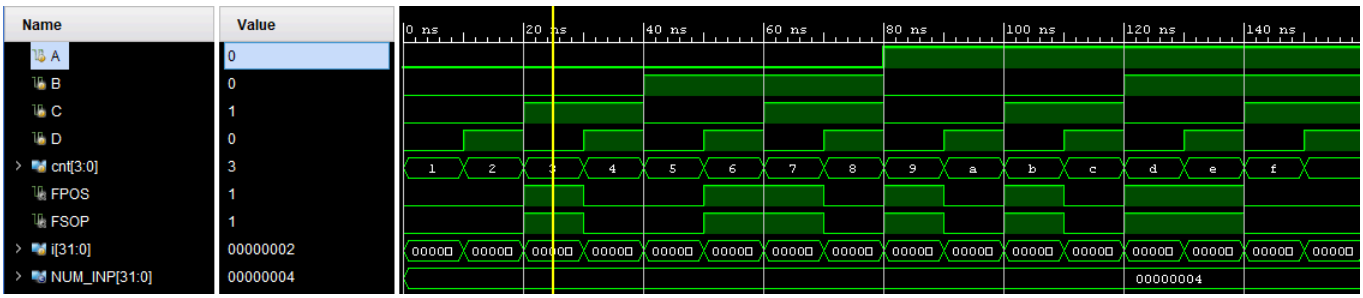

## Part 4: Boolean Simplification using K-Maps (Get equation -> Make circuit)

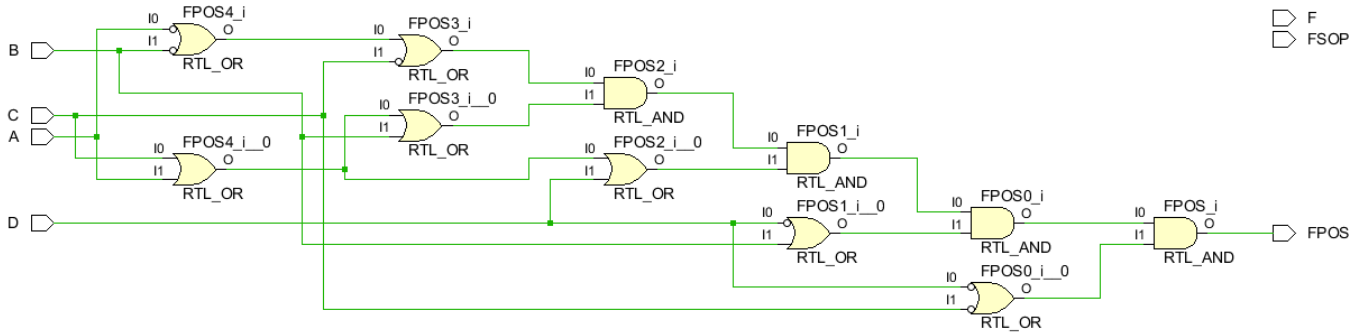
| A | B | C | D | F |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |

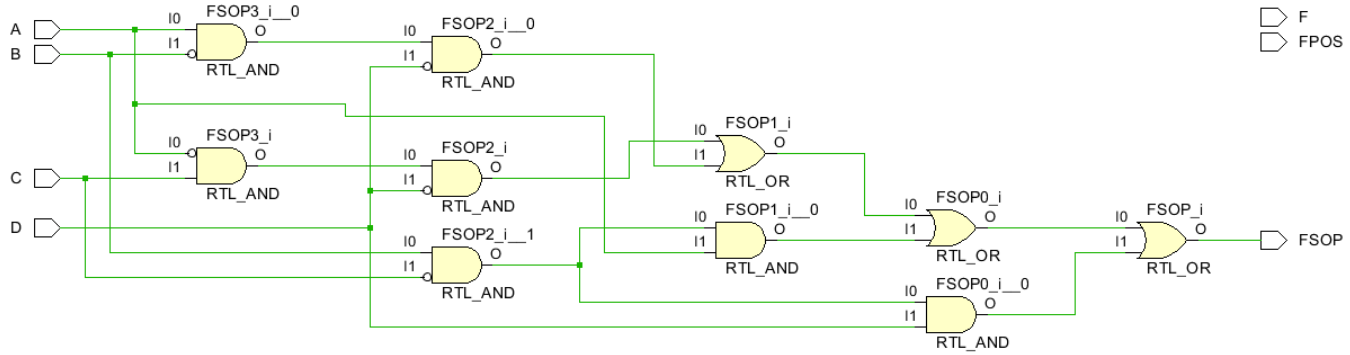
| A | B | C | D | F |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

```
module Part_1(
    input A,
    input B,
    input C,
    input D,
    output FPOS,
    output FSOP
    );

    assign FPOS = (~A|~B|~C)&(C|A|B)&(C|A|D)&(~D|B)&(~D|~C);
    assign FSOP = (~A&C&~D)|(A&~B&~D)|(B&~C&A)|(B&~C&D);
endmodule
```






K-Map:

| CD\AB | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 1 ⋁ |
| 01 | 0 | 1 ⋁ | 0 | 1 ⋀ |
| 11 | 1 -> | <- 1 ⋀ | 0 | 0 |
| 10 | <- 1 | 0 | 0 | 1 -> |

## Overall Deliverables and Q&A

What form of Boolean equations is an ANDs of ORs?

- Product of Sums (POS)
  What form of Boolean equations is an ORs of ANDs?
- Sum of Products (SOP)
  How many logic gates comprise a 3-variable term?
- A minimum of 2 gates (AND, OR, or NOT) depending on the term.
  What does a logic gate represent within the physical realm of computers?
- A logic gate represents a fundamental building block of digital circuits that perform logical operations on binary inputs.