CSC364/CSCM64 Lab 4

To be solved in groups of two or three.
Last day for lab sign-off: 28^{th} March 2022

Task 1. Consider the boolean formula

$$(\textbf{not } A \textbf{ and } B) \textbf{ and } (C \textbf{ or } D). \tag{1}$$

Assume that each logical connective in the formula 1 is implemented using a suitable logic gate. The following test suite is designed to test the implementation:

Case	A	B	C	D	Expected Output
1	T	T	T	F	F
2	F	T	F	F	F
3	T	F	T	T	F
4	F	F	F	F	F

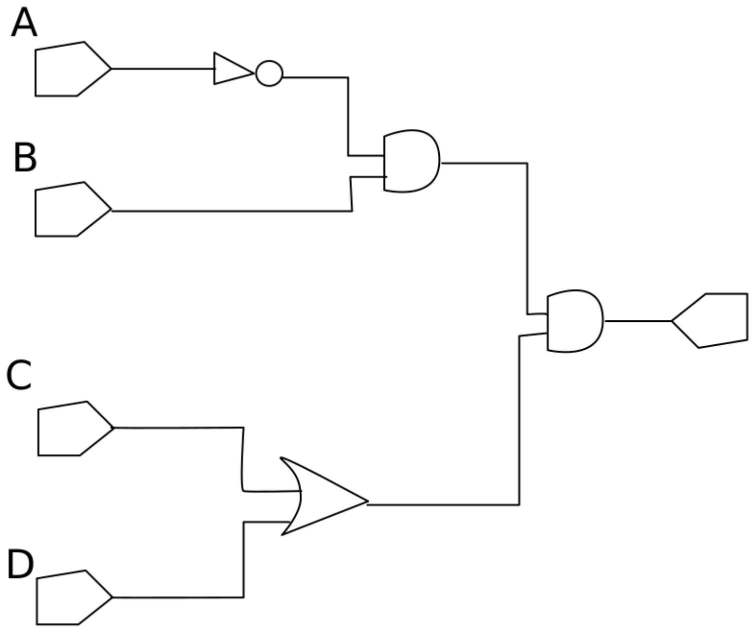
It is obvious that this test suite does not satisfy the MC/DC criterion (why?).

1. Use the first four steps of the five-step evaluation process introduced in the lectures to decide for each individual gate whether the test suite provides MC/DC for that gate (in the sense of the masking approach). Where the test suite does not provide MC/DC for a gate, list all missing test cases. Make all four steps clearly visible.

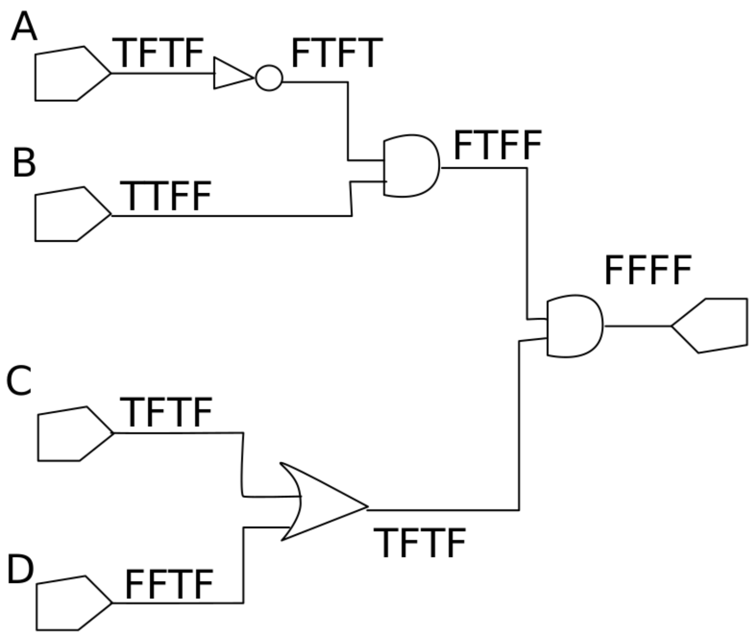
2. Add further test cases to the above test suite to make it satisfy the MC/DC criterion (in the sense of the masking approach). What is the smallest number of test cases one needs to add?

Answer 1. The test suite obviously does not provide MC/DC because the whole expression has not taken all possible outcomes (true and false).

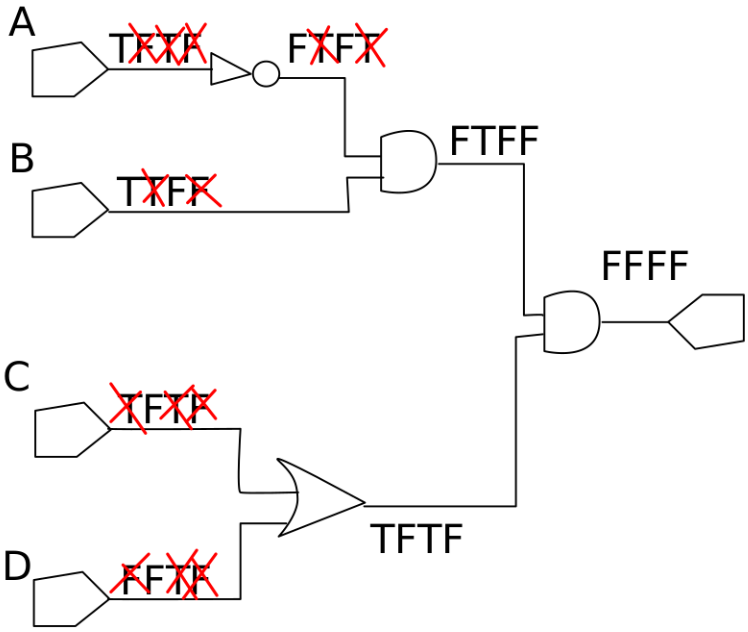
1. **Step 1.** Write down the formula schematically:



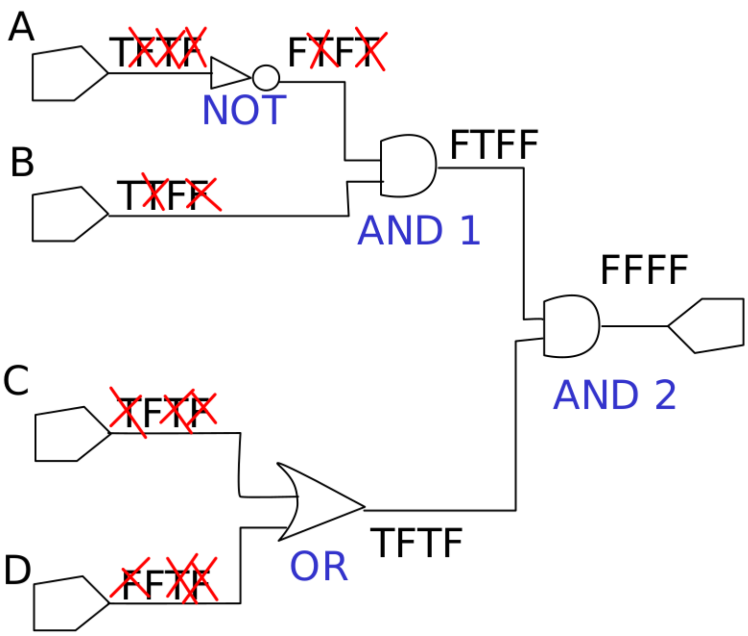
Step 2. Insert test values:



Step 3. Eliminate masked test cases:



Step 4. Verify MC/DC for each gate individually. To this end, let us introduce names for the gates:



The following table lists valid and missing test cases for all gates:

Gate	Valid Test Cases	Missing Test Cases	MC/DC?
NOT	T	F	No.
OR	FF	TF, FT	No.
AND1	FT, FF	TF, TT	No.
AND2	FT, TF, FF	TT	No.

2. The not-gate needs a valid **false** test input. The gate *AND1* needs the inputs TT and TF. We hence need to add at least two test inputs, where A is assigned **false** and B is assigned **true** and **false** respectively.

   The output of the or-gate needs to be **true** for both test inputs to prevent masking. When B is assigned **false**, the output of the or-gate will be masked. When B is assigned **true**, the output of the or-gate will not be masked. In the latter case we can add a missing test case to the or-gate. This suggests to add the following two inputs:

Case	A	B	C	D	Expected Output
5	F	F	T	T	F
6	F	T	T	F	T

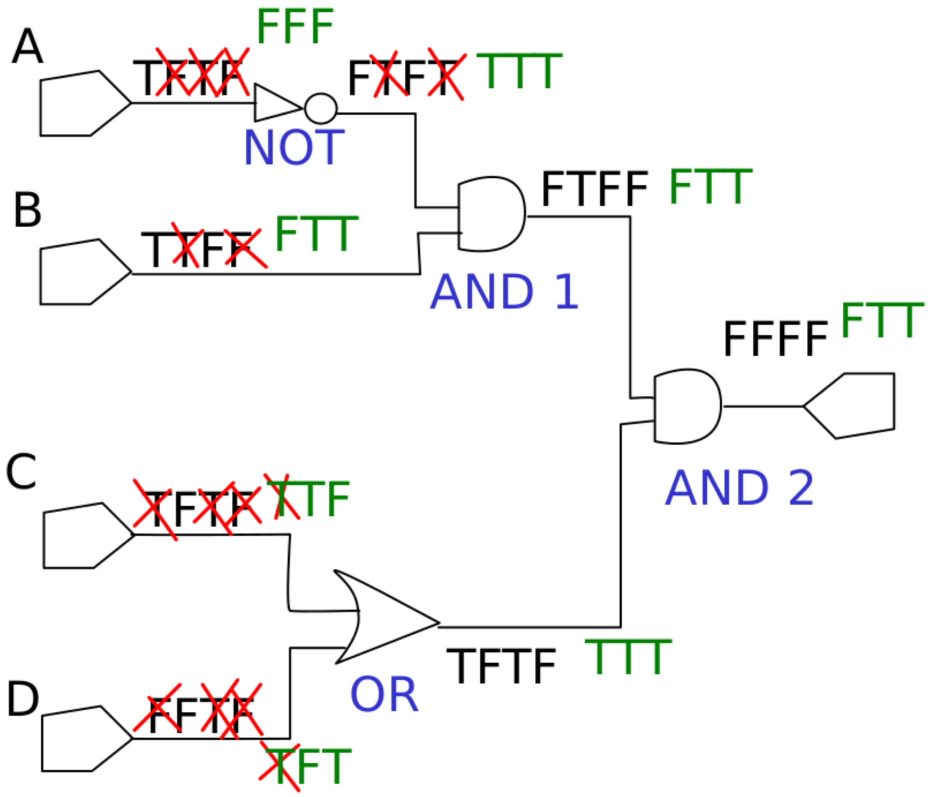
   For test case 5 we could have also chosen TF or FT for the assignment to C and D to prevent masking. For test case 6 we could have also chosen FT for the assignment to C and D to add this missing test case.

   After adding these inputs, the test case FT is still missing for the or-gate. We hence need to add another test input. The variable C needs to be assigned the value **false**. The variable D needs to be assigned the value **true**. The result of the *AND1*-gate has to be **true** to prevent masking. This only leaves us one choice for the assignment to A and B:

Case	A	B	C	D	Expected Output
7	F	T	F	T	T

   With these three new test cases, we have also added the missing test cases for the gate *AND2*. Hence, the new test suite provides MC/DC. It is clear from our discussion that it was necessary to add three test cases.

   A graphical representation of the new test suite is given on the next page.



A
TFTF
FFF
NOT
FTFT
TTT
B
TTFF
FTT
AND 1
FTFF
FTT
FFFF
FTT
AND 2
C
TFTF
TTF
OR
TFTF
TTT
D
FFTF
TFT