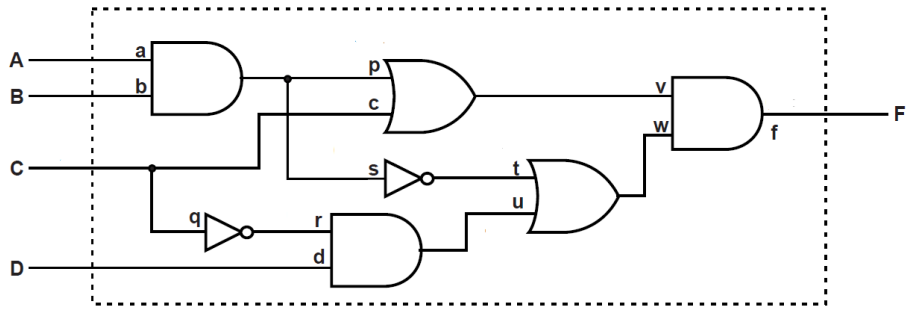
# 3. vaja: Odkrivanje napak s testnimi vektorji

## 3.1 Kombinacijsko vezje




Naredi model majhnega kombinacijskega vezja, ki ima štiri vhode in en izhod z logičnimi vrati. Deklariraj notranje povezave in opiši vezje na nivoju logičnih vrat: `and v1 (p, a, b); ...`

Ugotovi s katero kombinacijo vhodov bi ugotovil ali je vhod b zataknjen na 1 (SA1). Kombinacija mora biti takšna, da se sprememba izhoda prvih vrat (p,a,b) propagira na izhod vezja: _____

## 3.2 Testna struktura

```
for (i=0; i<16; i=i+1) begin
  {a, b, c, d} = i; #10;
end
```

Naredi testno strukturo, ki nastavi na vhode izbiralnika vse kombinacije z zanko **for**. Naredi simulacijo in ugotovi pri katerih kombinacijah vhodov se izhod postavi na ena: i = _____

V testni strukturi naredi vzporedni model kombinacijskega vezja, ki s stavkom **case** določi vrednost referenčnega izhoda f0. Stavek **case** je zapisan v svojem proceduralnem bloku:

```
always @* begin
  case ({a,b,c,d})
    2,3... : f0=1;
    default: f0=0;
  endcase
end
```

Dodaj v zanko za nastavljanje vhodov še preverjanje ali sta izhod vezja in referenčni izhod različna in izpis vhodnih kombinacij, kot pri prejšnji vaji. Preverjanje naj se izvede po zakasnitvi (#10). V simulatorju z nastavitvami signalov simuliraj napako znotraj testiranje komponente in preglej izpis po simulaciji. Npr. napako b SA1 nastavimo z ukazom: `force dut/b 1`

## 3.3 Modeliranje napak

Modeliraj napake na vhodih notranjih logičnih vrat **and, or** in zapiši testne vektorje, ki odkrijejo napako:

| | |
|---|---|
| r SA0 | |
| r SA1 | |
| p SA0 | |
| p SA1 | |
| t SA0 | |
| t SA1 | |
| u SA0 | |
| u SA1 | |

Ugotovi koliko in katere testne vektorje je potrebno poslati na vhod vezja, da odkrijemo vse te napake? _____