To witness o2 = 1, since g9 is an and gate, o2 = 1 is the output-controlling.

- ⇨ f = 1 and g8 = 1
- ⇨ build f = 1 (bdd node count = 1)
- ⇨ build witness bdd g8 = 1, first try to witness g4 = 1 (care set is g7 = 0)
  - =>build g4 = 1, since g4 is an and gate, g4 = 1 is output controlling
  - =>build c = 1
  - =>build g1 = 1, since g1 is an and gate, g1 = 1 is output-sontrolling
  - =>build g1
  - =>That is, A is exactly the whole bdd of g4
  - =>build g7 = 0, since its “iscare” = true, we also have to build the whole bdd
  - =>That is, B is exactly the bdd of ~g7
- ⇨ R = restrict(A, B) = restrict(g4, ~g7)
- ⇨ And R with f gives
- ⇨ The output is:

```
78
79      BddNode R = bm.restrict(g4, ~g7);
80      cout << "R = " << R << endl;
81
82      BddNode witness_bdd = R & f;
83      cout << "winess_bdd = " << witness_bdd << endl;
84      // BddNode g = c | d;
85      // cout << g << endl;
86
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

> building: testBdd
(base) yenlu_mepu@cthulhu:~/RicBDD$ ./testBdd
R = [4](+) 0x18d5c40 (2)
  [3](+) 0x18d5bf0 (1)
    [2](+) 0x18d5970 (6)
      [0](+) 0x18d4310 (26)
      [0](-) 0x18d4310 (26) (*)
    [0](-) 0x18d4310 (26) (*)
  [0](-) 0x18d4310 (26) (*)

==> Total #BddNodes : 4

winess_bdd = [6](+) 0x18d64e0 (1)
  [4](+) 0x18d5c40 (3)
    [3](+) 0x18d5bf0 (1)
      [2](+) 0x18d5970 (6)
        [0](+) 0x18d4310 (27)
        [0](-) 0x18d4310 (27) (*)
      [0](-) 0x18d4310 (27) (*)
    [0](-) 0x18d4310 (27) (*)
  [0](-) 0x18d4310 (27) (*)

==> Total #BddNodes : 5
```

- ⇨ The total bdd node = 5, which is less than the original 10 nodes