```
library IEEE;
  use IEEE.STD_LOGIC_1164.all;

entity tb_fulladder is -- testbenkentiteter er normalt tomme.
end entity tb_fulladder;

architecture behavioral of tb_fulladder is
  -- en komponent er en entitet definert i en annen fil, og som vi vil bruke.
  -- komponentdeklarasjonen må matche entiteten.
  component fulladder is
    port(
      a, b : in std_logic;
      cin  : in std_logic;
      s    : out std_logic;
      cout : out std_logic
    );
  end component;

  -- Tilordning av startverdi ved deklarasjon gjøres med :=
  signal tb_a, tb_b, tb_cin : std_logic := '0';

  -- outputs bør ikke få en startverdi i testbenken, da det kan maskere feil.
  signal tb_s, tb_cout : std_logic;

begin
  -- instansiering:
  DUT: fulladder     -- Merkelappen DUT betyr «device under test» som er en av
mange
  port map(          -- vanlige betegnelser på simuleringsobjektet.
    a    => tb_a,    -- Mappinger gjøres med =>, til forskjell fra tilordninger som
    b    => tb_b,    -- bruker <= eller :=
    cin  => tb_cin,  -- Mappinger kan ses på en ren sammenkobling av ledninger
    s    => tb_s,    -- Vi mapper alltid testenhetens porter til testbenkens
signaler
    cout => tb_cout  -- Siste informasjon før parantes-slutt har ikke ',' eller ';'
  );

  -- I testbenker kan vi ha prosesser uten sensitivitetsliste..
  -- i slike prosesser kan vi angi tid med «wait» statements, og
  -- vi kan sette signaler flere ganger etter hverandre uten å gi konflikter.
  -- NB: Prosessen vil trigges om og om igjen om vi ikke hindrer det.
  process
  begin
    wait for 10 ns;
    tb_a <= '0';
    tb_b <= '0';
    tb_cin <= '0';
    wait for 10 ns;
    tb_a <= '0';
    tb_b <= '0';
    tb_cin <= '1';
    wait for 10 ns;
    tb_a <= '0';
    tb_b <= '1';
    tb_cin <= '0';
    wait for 10 ns;
    tb_a <= '0';
    tb_b <= '1';
    tb_cin <= '1';
```

```
    wait for 10 ns;
    tb_a <= '1';
    tb_b <= '0';
    tb_cin <= '0';
    wait for 10 ns;
    tb_a <= '1';
    tb_b <= '0';
    tb_cin <= '1';
    wait for 10 ns;
    tb_a <= '1';
    tb_b <= '1';
    tb_cin <= '0';
    wait for 10 ns;
    tb_a <= '1';
    tb_b <= '1';
    tb_cin <= '1';
    wait for 10 ns;

    report("Ferdig!") severity note;
    std.env.stop; -- stopper simuleringen
  end process;
end architecture behavioral;
```