May 1, 2006

CSE2306 Digital Logic CSE1308

Practical 7: Asynchronous sequential circuits

7.1 About this practical

The objective of this practical is to design and test three implementations of a D flip-flop as an asynchronous sequential circuit.

Contents

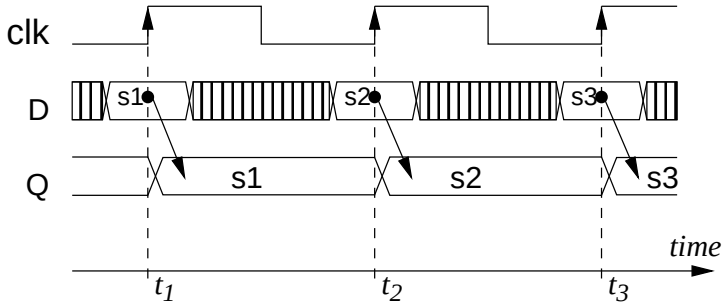
7.2 A two-state-signals implementation of a D flip flop

The first implementation is based on the generic description of a D flip-flop in VHDL which is based on the **process** statement and the **event** attribute.

- Start the Design manager and create a new project: `P7`**you**
- Create a VHDL design view with the
  - Entity name: `DFF`**you**
  - Architecture name: e.g., **ahdl**
- In the **VHDL editor window** enter an appropriate entity specification for the D flip-flop and
- the architecture to be as follows (watch the single quotes!)

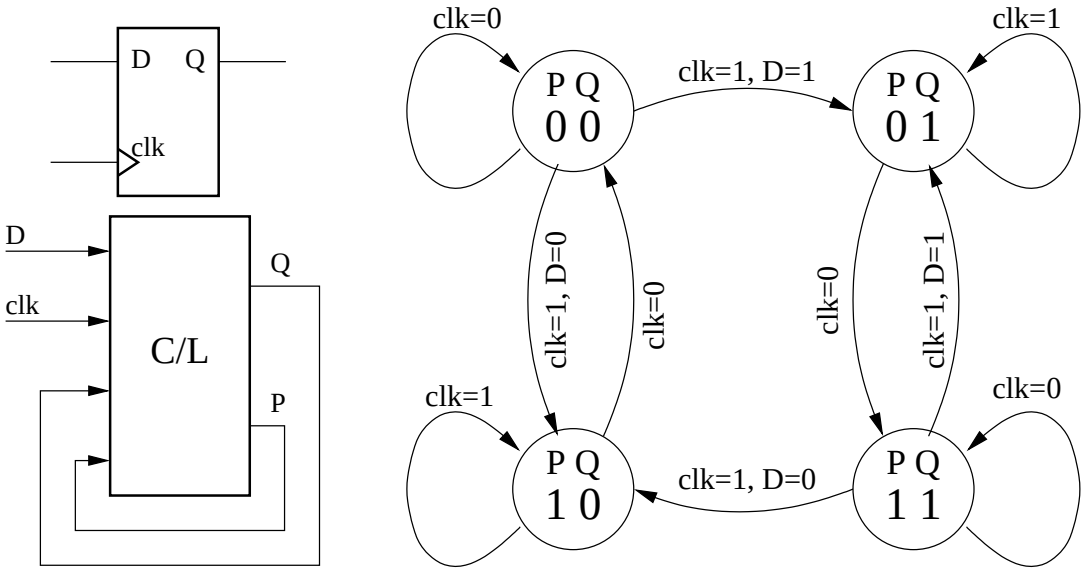
```
ARCHITECTURE ahdl OF DFFyou IS
BEGIN
 PROCESS (clk)
 BEGIN
   IF (clk'EVENT AND clk = '1') THEN
     Q <= D ;
   END IF ;
 END PROCESS ;
END ARCHITECTURE ahdl;
```

- Save the VHDL file.
- Compile the VHDL specification selecting in the VHDL editor window:
  Tasks → ModelSim flow → Run single
- At the conclusion of the successful compilation a window **Start ModelSim** appears. If you accept its default settings, it will open ModelSim simulation window.
- Write and execute an appropriate simulation script to generate waveforms conceptually similar to the following:

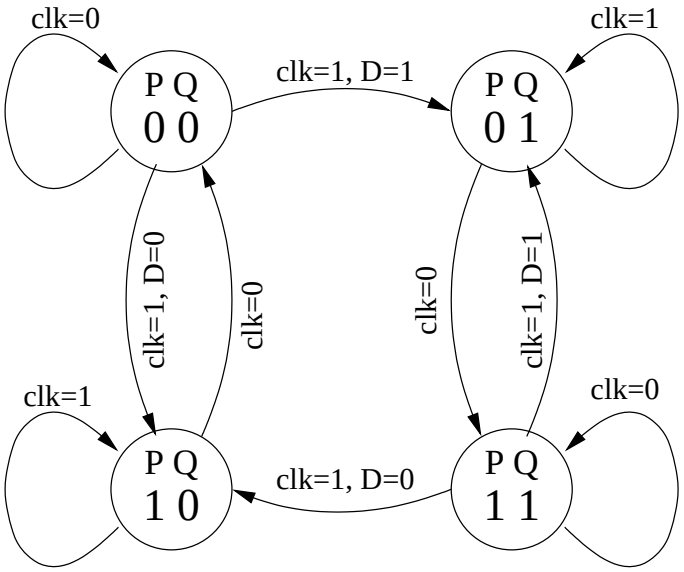


7.3 A two-state-signals implementation of a D flip flop

- In the second implementation of the D flip-flop we start with the state diagram with two state signals P and Q as discussed in the lecture notes:



- Subsequently, we convert the **state diagram** into the **state transition** table.

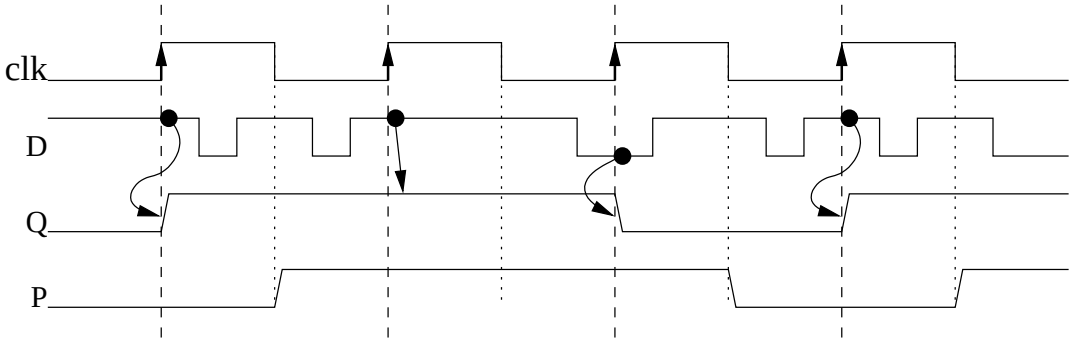


clk	D	P	Q	P+	Q+
0	–	0	0	0	0
1	0	0	0	1	0
1	1	0	0	0	1
0	–	0	1	1	1
1	–	0	1	0	1
0	–	1	0	0	0
1	–	1	0	1	0
0	–	1	1	1	1
1	0	1	1	1	0
1	1	1	1	0	1

- Create a new VHDL architecture flowing from the above state table similar to the following:

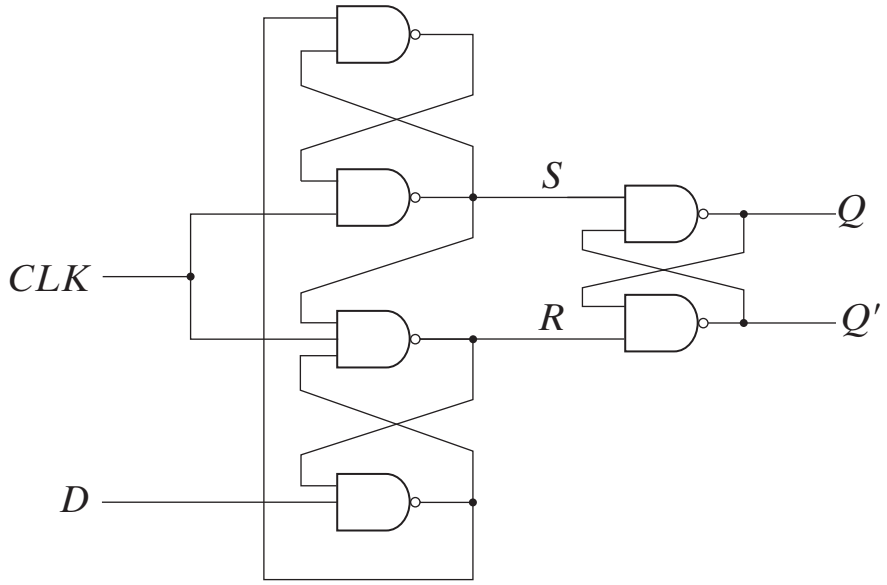
```
ARCHITECTURE  ttbl  OF  DFFyou IS
  TYPE arr2d IS ARRAY (natural range <>,
                       natural range <>) OF std_logic;
  CONSTANT ttDff : arr2d(1 to 2, 0 to 15) := (
   --    truth table for P+  Q+
   --0123456789abcdef  (P, Q, clk, D)
    "0010110000111110" ,  --1  P
    "0001111100001101" ); --2  Q
   SIGNAL P, QQ : std_logic ;
   SIGNAL     S : std_logic_vector (3 downto 0) ;
   SIGNAL    Si : integer range 0 to 15 ;
BEGIN
    S <= (P, QQ, clk, D) ;
   Si <= conv_integer(unsigned(S)) ;
   --    reading from the truth table
    P <= ttDff (1, Si) ;
   QQ <= ttDff (2, Si) ;
    Q <= QQ ;
END ttbl ;
```

- Simulate the VHDL design to obtain waveforms similar to the following:



7.4 A three-state-signals implementation of a D flip flop

- The third implementation of the D flip-flop is based on the following NAND-based logic diagram.



- Use the **graphical entry** to specify the D flip-flop given by the above logic diagram.
- Simulate the design using an appropriate simulation script. Show the internal R and S signals.

7.5 The report

In your report (due after prac 8) include the results in the form of:

- Relevant state diagrams, state tables, state equations, other logic equations,
- block/logic diagrams,
- VHDL programs,
- simulation scripts,
- simulation waveforms,
- **short** description of the above.

Wherever possible publish the results selecting in the **Block Diagram** window
File → **HTML Export ...** . Specify the export target directory to be `...\DigDes\Reports`.