**DISSENY DIGITAL BÀSIC 2016-2017**

***PRÀCTICA 5: Implementació de sistemes seqüencials i aplicació a circuits lògics.***

L’objectiu d’aquesta pràctica és realitzar la implementació de sistemes seqüencials bàsics com són els Flips-Flops amb filosofia de disseny ‘**ifthen**’. A més a més, utilitzant aquests Flip-Flops i la filosofia de disseny estructural es poden realitzar exercicis del bloc 3.

Recordem que la metodologia del disseny ‘**ifthen**’ consisteix en construir clàusules del tipus ‘*si succeeix .... amb la variable de entrada, la sortida serà ...*’. Com a exemple d’aquesta metodologia, que ja s’ha introduït a la pràctica 4, implementarem a continuació un ‘**FF D**’ per flanc de baixada amb ‘**Preset**’ i ‘**Clear**’, i un ‘**Latch JK**’ amb ‘**Preset**’ i ‘**Clear**’:

ENTITY D\_Bajada\_PreClr IS

PORT(D,Clk,Pre,Clr: IN BIT; Q,NO\_Q: OUT BIT);

END D\_Bajada\_PreClr;

ARCHITECTURE ifthen OF D\_Bajada\_PreClr IS

SIGNAL qint: BIT;

-- Aquesta és la forma de definir sortides que es realimenten

-- a l’entrada. Recordem que en VHDL si una variable és de sortida no es

-- pot utilitzar com a entrada de nou. Por a això definim una variable interna

-- que després assignarem a la variable de sortida

BEGIN

PROCESS (Clk,Pre,Clr)

BEGIN

IF Clr='0' THEN qint<='0' AFTER 2 ns;

-- Aquí la funció Clear és Active-low, és a dir, quan Clr=0 la sortida es posa

-- a 0 de forma asíncrona; quan Clr=1 el circuit farà alguna altre funció.

ELSIF Pre='0' THEN qint<='1' AFTER 2 ns;

-- Hem imposat nosaltres que si es posen, simultàniament, Clear i Preset

-- a 0, el Clear és qui mana. Per això aquí la condició sobre el Preset

-- s’analitza només quan el Clear és 1.

ELSIF Clk'EVENT AND Clk='0' THEN

-- Clk’EVENT és una instrucció que dóna una sortida veritat, és a dir, un 1

-- quan es produeix un canvi en el valor de la variable Clk.

-- La instrucció, tal com està aquí, indica que si es produeix un canvi en

-- el senyal Clk i, a més, el valor posterior és 0, llavors …

qint <= D AFTER 6 ns;

END IF;

END PROCESS;

Q<=qint; NO\_Q<=NOT qint;

-- Aquí es on es fa l’assignació de les variables

-- internes a les variables de sortida

END ifthen;

ENTITY JK\_Latch\_PreClr IS

PORT(J,K,Clk,Pre,Clr: IN BIT; Q,NO\_Q: OUT BIT);

END JK\_Latch\_PreClr;

ARCHITECTURE ifthen OF JK\_Latch\_PreClr IS

SIGNAL qint: BIT;

BEGIN

PROCESS (J,K,Clk,Pre,Clr)

BEGIN

IF Clr='0' THEN qint<='0' AFTER 2 ns;

ELSE

IF Pre='0' THEN qint<='1' AFTER 2 ns;

ELSE

IF Clk='1' THEN

IF J='0' AND K='0' THEN qint<=qint AFTER 6 ns;

ELSIF J='0' AND K='1' THEN qint<='0' AFTER 6 ns;

ELSIF J='1' AND K='0' THEN qint<='1' AFTER 6 ns;

ELSIF J='1' AND K='1' THEN qint<= NOT qint AFTER 6 ns;

END IF;

END IF;

END IF;

END IF;

END PROCESS;

Q<=qint; NO\_Q<=NOT qint;

END ifthen;

-- Banco de Pruebas de ambos

ENTITY banco\_pruebas IS

END banco\_pruebas;

ARCHITECTURE test OF banco\_pruebas IS

COMPONENT mi\_D\_Bajada\_PreClr IS

PORT(D,Clk,Pre,Clr: IN BIT; Q,NO\_Q: OUT BIT);

END COMPONENT;

COMPONENT mi\_JK\_Latch\_PreClr IS

PORT(J,K,Clk,Pre,Clr: IN BIT; Q,NO\_Q: OUT BIT);

END COMPONENT;

SIGNAL ent1,ent2,clock,preset,clear,Dsort\_Q,Dsort\_noQ,JKsort\_Q,JKsort\_noQ: BIT;

FOR DUT1: mi\_D\_Bajada\_PreClr USE ENTITY WORK.D\_Bajada\_PreClr(ifthen);

FOR DUT2: mi\_JK\_Latch\_PreClr USE ENTITY WORK.JK\_Latch\_PreClr(ifthen);

BEGIN

DUT1: mi\_D\_Bajada\_PreClr PORT MAP (ent1,clock,preset,clear,Dsort\_Q,Dsort\_noQ);

DUT2: mi\_JK\_Latch\_PreClr PORT MAP (ent1,ent2,clock,preset,clear,JKsort\_Q,JKsort\_noQ);

ent1 <= NOT ent1 AFTER 800 ns;

ent2 <= NOT ent2 AFTER 400 ns;

clock <= NOT clock AFTER 500 ns;

preset <= '0', '1' AFTER 600 ns;

clear <= '1','0' AFTER 200 ns, '1' AFTER 400 ns;

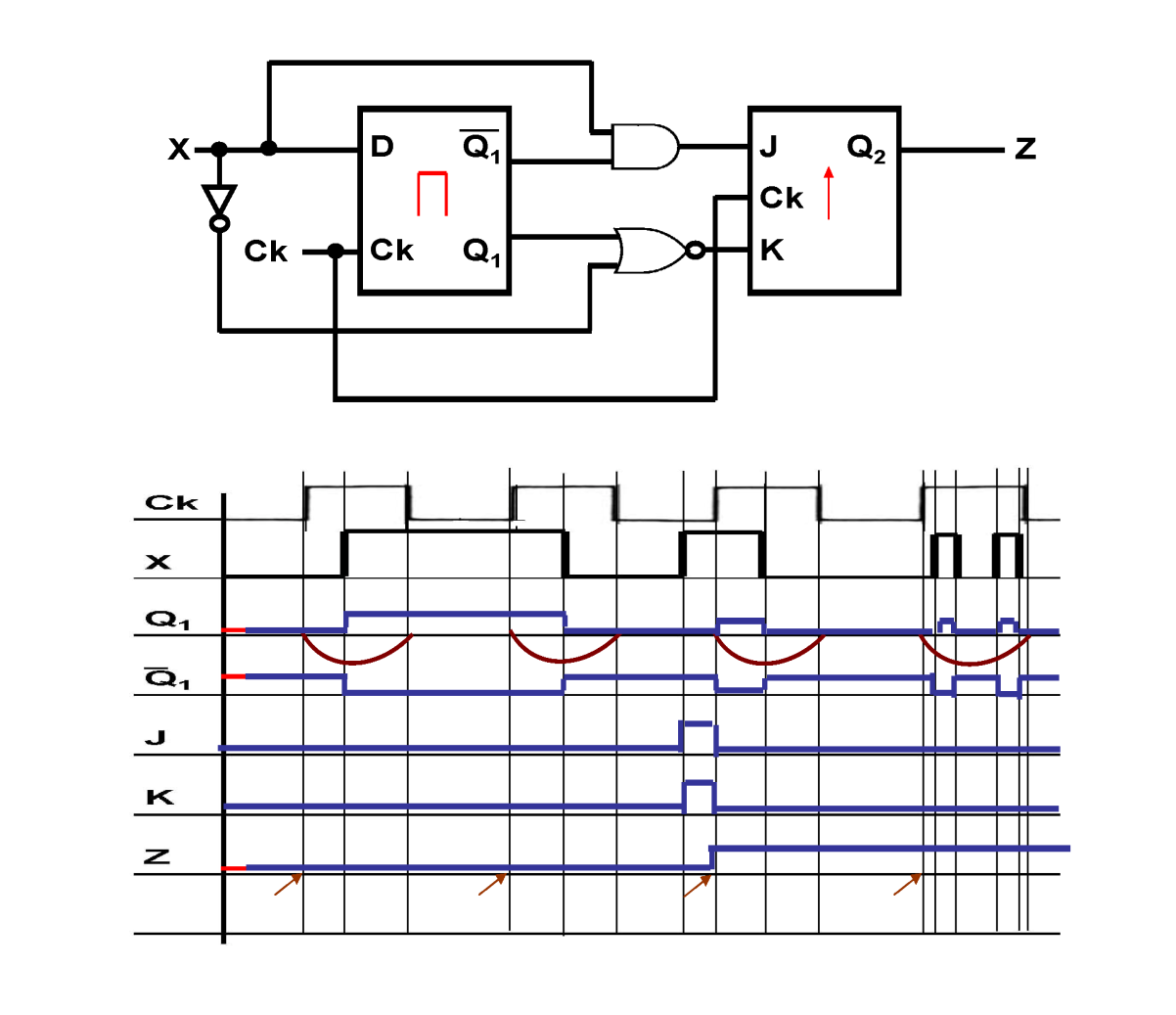
-- simular hasta 15000 ns

END test;

**Treball a desenvolupar de forma autònoma:**

**(a entregar abans de la vostra corresponent sessió de pràctiques a través de CampusVirtual)**

1. Comproveu el funcionament dels FFs descrits anteriorment i compareu-los amb la teoria.
2. Realitzeu amb arquitectures de tipus ‘**ifthen**’ un Latch D i un Flip-Flop JK per flanc de pujada, ambdós amb PRESET i CLEAR (feu que el PRESET tingui prioritat).
3. Utilitzant els Flips-Flops del punt 2 i les portes necessàries, realitzeu amb arquitectures de tipus estructural el següent circuit que té dues entrades ‘**x**’ i ‘**Ck**’, i una sortida ‘**z**’.
4. Simuleu el comportament per les entades representades al cronograma inferior.



**Treball a desenvolupar al laboratori:**

* Recupereu el codi del sumador modificat (de 1 i 3 bits) i comproveu el seu funcionament.
* Feu una ALU amb 8 funcions (les que es defineixen a la teoria) utilitzant el sumador modificat. Teniu el codi del sumador modificat de 3 bits a la pràctica 3 i la ALU la teniu definida a la teoria (transparències del tema 6).
* Intenteu afegir també un registre de 3 bits a la sortida de la ALU (descrit al tema 8).

Haureu de pujar un fitxer 24 hores abans de la vostra sessió de pràctiques que continguin les següents informacions:

* Un fitxer amb les entitats i arquitectures de les portes, Lach\_D, FF\_JK i banc de proves. El fitxer es dirà **P5\_Zxx\_Cognom1\_Cognom2\_Nom\_portes.vhd** (**Zxx** serà el vostre número del grup de pràctiques).