**DISSENY DIGITAL BÀSIC 2019-2020**

***PRÀCTICA 3: Implementació de circuits combinacionals i seqüencials amb arquitectura “ifthen”***

***(dies 18, 19, 20, 21, 22 i 28 de novembre)***

L’objectiu d’aquesta tercera pràctica és doble: per un costat realitzarem la implementació de funcions lògiques utilitzant una nova arquitectura que anomenarem ***ifthen***, i, per l’altra, utilitzarem aquesta nova arquitectura per implementar sistemes seqüencials bàsics del tipus ***flip-flops***. De fet, utilitzant aquests ***flip-flops*** i la filosofia de disseny ***estructural*** es poden realitzar exercicis del **bloc 3**.

La metodologia del disseny ‘***ifthen***’ consisteix en construir clàusules del tipus ‘si succeeix .... amb la variable de entrada, la sortida serà ...’. Es basa, doncs, en la valoració de les diferents situacions que es podem donar en el sistema electrònic i, per tant, és equivalent a escriure la taula de veritat. Això ens servirà, entre altres coses, per poder entendre el comportament d’aquest nou tipus de dispositius i complementar la teoria, comprovant-la.

L’arquitectura ‘***ifthen***’ presenta un disseny molt més simple i directe que l’arquitectura lògica, on podem expressar de forma directa la taula de la veritat de la funció que es realitza. Aquí us exposem el codi per a la implementació de la funció **f=/a·b + a·/c**, que és la mateixa funció que us hem presentat com a exemple a l’arquitectura estructural de la pràctica 2.

-- Defineixo l’entitat

--ENTITY funcio\_logica is

--PORT ( a,b,c: IN BIT;

-- f: OUT BIT);

--END Funcio\_logica;

-- Ara definirem la nova arquitectura, tipus ‘ifthen’

ARCHITECTURE ifthen OF funcio\_logica IS

BEGIN

PROCESS (a, b, c)

-- Analitzem els canvis a les variables d’entrada a, b, c.

BEGIN

-- Iniciem el cos de l’arquitectura.

IF a='0' AND b='0' AND c='0' THEN

f<= '0' AFTER 3 ns;

ELSIF a='0' AND b='0' AND c='1' THEN

f<= '0' AFTER 3 ns;

ELSIF a='0' AND b='1' AND c='0' THEN

f<= '1' AFTER 3 ns;

ELSIF a='0' AND b='1' AND c='1' THEN

f<= '1' AFTER 3 ns;

ELSIF a='1' AND b='0' AND c='0' THEN

f<= '1' AFTER 3 ns;

ELSIF a='1' AND b='0' AND c='1' THEN

f<= '0' AFTER 3 ns;

ELSIF a='1' AND b='1' AND c='0' THEN

f<= '1' AFTER 3 ns;

ELSIF a='1' AND b='1' AND c='1' THEN

f<= '0' AFTER 3 ns;

END IF;

END PROCESS;

END ifthen;

-- Realitzem el banc de proves

-- Aquest banc de proves també contempla les arquitectures logica i estructural del banc de proves anterior (de fet, l’hem definit com a una entitat diferent).

ENTITY banc\_de\_proves IS

END banc\_de\_proves;

ARCHITECTURE test OF banc\_de\_proves IS

COMPONENT bloc\_que\_simulem IS

PORT ( a,b,c: IN BIT;

f: OUT BIT);

-- Recordeu que els noms de les variables del component han de ser

-- exactament les mateixes que a l’entitat a que es refereixen.

END COMPONENT;

SIGNAL senyalA,senyalB,senyalC: BIT;

SIGNAL sortida\_f\_logica,sortida\_f\_estructural, sortida\_f\_ifthen: BIT;

FOR DUT1: bloc\_que\_simulem USE ENTITY WORK.Funcio\_logica(logica);

FOR DUT2: bloc\_que\_simulem USE ENTITY WORK.Funcio\_logica(estructural);

FOR DUT3: bloc\_que\_simulem USE ENTITY WORK.Funcio\_logica(ifthen);

BEGIN

DUT1: bloc\_que\_simulem PORT MAP(senyalA, senyalB, senyalC, sortida\_f\_logica);

DUT2: bloc\_que\_simulem PORT MAP(senyalA, senyalB, senyalC, sortida\_f\_estructural);

DUT3: bloc\_que\_simulem PORT MAP(senyalA, senyalB, senyalC, sortida\_f\_ifthen);

PROCESS (senyalA, senyalB, senyalC)

BEGIN

senyalA <= NOT senyalA AFTER 200 ns;

senyalB <= NOT senyalB AFTER 100 ns;

senyalC <= NOT senyalC AFTER 50 ns;

END PROCESS;

END test;

Aquesta nova arquitectura és molt útil per definir sistemes seqüencials, en els que la seva sortida depèn de l’estat anterior, com són els biestables. Com a exemple d’aquesta metodologia implementarem primer un ‘**FF D**’ per flanc de baixada amb ‘**Preset**’ i ‘**Clear**’, i després 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. Per tant definim una variable interna

-- que després assignarem a la variable de sortida.

BEGIN

PROCESS (D,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 altra 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 té prioritat. Per això aquí la condició sobre el Preset

-- s’analitza només quan el Clear està posat a 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 (baixada), llavors …

qint <= D AFTER 2 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,qint)

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 2 ns;

ELSIF J='0' AND K='1' THEN qint<='0' AFTER 2 ns;

ELSIF J='1' AND K='0' THEN qint<='1' AFTER 2 ns;

ELSIF J='1' AND K='1' THEN qint<= NOT qint AFTER 2 ns;

END IF;

END IF;

END IF;

END IF;

END PROCESS;

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

END ifthen;

-- Banc de Proves d’ambos

ENTITY banc\_proves IS

END banc\_proves;

ARCHITECTURE test OF banc\_proves 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;

-- simuleu fins a 15000 ns

END test;

**Treball a desenvolupar de forma autònoma:**

**(a entregar 48 hores 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. Definiu les entitats amb les seves corresponents arquitectures de tipus ‘**ifthen**’ (amb retard) dels següents sistemes seqüencials:
   1. Latch D i Flip-Flop D, tots dos amb els següents ports d’entrada i sortida:

IN BIT: D,Clk,Pre,Clr

OUT BIT: Q,NO\_Q

* 1. Latch J-K i Flip-Flop J-K, tots dos amb els següents ports d’entrada i sortida:

IN BIT: J,K,Clk,Pre,Clr

OUT BIT: Q,NO\_Q

* 1. Latch T i Flip-Flop T, tots dos amb els següents ports d’entrada i sortida:

IN BIT: T,Clk,Pre,Clr

OUT BIT: Q,NO\_Q

1. Escribiu un banc de proves ***bdp*** amb la seva arquitectura ***test*** per tal de comprobar el correcte funcionament d’aquests biestables.
2. Quina diferència hi ha entre un Latch i un FF? (ja sigui del tipus D, JK o T). Afegeix un comentari raonant la teva resposta.

Recordeu de fer córrer la simulació un temps prou llarg per tal que es puguin veure totes les combinacions possibles de les entrades. Amb l’ajuda del cursor, comproveu que el comportament és el que s’espera.

**Aquest és el treball que haureu de pujar a través del campus virtual abans de la vostra corresponent sessió de pràctiques. El termini d’aquesta tasca és 48 hores abans de la vostra sessió de pràctiques, que serà diferent per als diferents grups. Un cop passat aquest temps ja no serà possible pujar els fitxers.**

* Pujeu un fitxer VHD amb les entitats i arquitectures dels biestables i el seu banc de proves. El fitxer es dirà **P3\_Cognom1\_Cognom2\_Nom.vhd**.