DISSENY DIGITAL BÀSIC 2021-2022

PRÀCTICA 1: Implementació de funcions lògiques (dies 18,19,20,21,22 d'Octubre)

L'objectiu d'aquesta pràctica és familiaritzar-se amb el programa de simulació *QuestaSim*/*ModelSim* i començar a fer servir les seves propietats per realitzar funcions lògiques tal com les descrites a les classes de teoria. En concret, volem realitzar les funcions bàsiques:

- Inversió (ja fet a la pràctica 0)
- Producte lògic (AND)
- Suma lògica (OR)

A més, volem realitzar també

- Implementació de funcions lògiques

En aquesta pràctica heu de generar l'entitat, les arquitectures i el banc de proves corresponents a les portes AND i OR de dues, tres i quatre entrades. Caldrà, doncs, que genereu 7 entitats, que anomenareu **inversor** (la mateixa de la pràctica inicial), **and2**, **and3**, **and4**, **or2**, **or3** i **or4**, cadascuna amb la seva arquitectura **logica**. Recordeu que el nom de l'arquitectura fa referència sempre a una entitat concreta, que utilitzarà les definicions de les funcions lògiques AND, OR i NOT que s'han donat a teoria i que podeu trobar a qualsevol dels llibres de l'assignatura.
Feu servir com a variables d'entrada **a**, **b** (per a les portes de 2 entrades); **a**, **b**, **c** (per a les portes de 3 entrades) o **a**, **b**, **c** i **d** (per a les de 4). La sortida serà **z** per a cadascuna de les entitats. A mode d'exemple, us presentem l'entitat i l'arquitectura de la porta **and2**:

```
ENTITY and2 IS
PORT(a, b: IN BIT; z: OUT BIT);
END and2;

ARCHITECTURE logica OF and2 IS
BEGIN
Z <= a AND b;
END logica;
```

Genereu un banc de proves per *testar cada dispositiu de forma individual* (**inversor**, **and2**, **or2**, **and3**, **or3**, **and4** i **or4**), i així comprovar el seu correcte funcionament. Després genereu un banc de proves que us permeti *testar totes les entitats simultàniament*. Heu de tenir en compte que els dispositius podran compartir entrades, però cadascun d'ells haurà de tenir una sortida diferent.

Un altre aspecte molt important serà que el banc de proves permeti visualitzar tots els casos possibles de les variables d'entrada, és a dir, que les variables d'entrada recorrin totes les combinacions de valors possibles. A la *pràctica 0* ja s'ha mostrat una forma de fer-ho, però en el cas de senyals periòdics, és a dir, que tornen a repetir-se amb durada determinada, hi ha una manera més compacta de fer-ho. A continuació detallem com fer-ho en el cas de la porta **and2**, fent servir l'entitat i l'arquitectura de dalt:

```
ENTITY banc_de_proves IS
END banc_de_proves;

ARCHITECTURE test OF banc_de_proves IS
-- Aquí definim els components el comportament dels quals volem comprovar.
-- Tot i que encara no hem relacionat amb l'entitat que utilitzarem,
```

```
-- el component ha de tenir EXACTAMENT els mateixos noms dels senyals d'entrada i
-- sortida que l'entitat a la que lligarem.
COMPONENT la_porta_and2
PORT (a,b: IN BIT; z: OUT BIT);
END COMPONENT;
-- Declarem els senyals externs que hi han i de quin tipus són. Recordeu que
-- encara no es determina quins seràn d'entrada i quins de sortida.
SIGNAL ent1, ent2, sortida: BIT;
-- Relacionem el component que volem testejar amb una entitat i arquitectura
-- que abans ja haurem generat i compilat. Cal que ambdues estiguin en un fitxer .VHD
-- en el directori de treball, si no estan incorporades al mateix fitxer que estem
-- generant.
FOR DUT1: la_porta_and2 USE ENTITY WORK.and2(logica);

-- Hem acabat la declaració i ara comença el cos de l'arquitectura
BEGIN
-- Aquí establim la relació entre els senyals abans definits i els
-- terminals del component a testejar. El programa els relaciona
-- en el mateix ordre que estan a l'entitat: el 1r senyal extern serà el 1r
-- senyal de l'entitat
DUT1: la_porta_and2 PORT MAP (ent1,ent2,sortida);

-- Comença l'anàlisi de la variació de les variables, que cal especificar
-- explícitament. Indiquem al PROCESS la variació de totes les variables
-- que volem que el programa analitzi, aquí les d'entrada ent1 i ent2.
PROCESS (ent1,ent2)
BEGIN
ent1 <= NOT ent1 AFTER 50 ns;
ent2 <= NOT ent2 AFTER 100 ns;
END PROCESS;
END test;
```

Alternativament, i fent servir ara el que ja vàrem aprendre a la *pràctica 0*, la variació dels senyals també la podríem fer de forma manual, indicant tots els canvis. Per tant, podríem escriure al codi del banc de proves el següent, tenint en compte que **substitueix** el text en **verd** al codi anterior:

```
PROCESS
BEGIN
ent1 <= '0';
ent2 <= '0';
WAIT FOR 50 ns;
ent1 <= '1';
ent2 <= '0';
WAIT FOR 50 ns;
ent2 <= '0';
ent1 <= '1';
WAIT FOR 50 ns;
ent1 <= '1';
ent2 <= '1';
WAIT FOR 50 ns;
ent1 <= '0';
ent2 <= '0';
WAIT FOR 50 ns;
END PROCESS;
```

És evident que en aquest darrer codi s'han d'escriure de forma manual totes les combinacions possibles que adquireixen les entrades. Pel cas de dos senyals d'entrada, el codi no és massa llarg, però la longitud del codi creix exponencialment amb el número de senyals d'entrada (en cas de tenir 8 senyals d'entrada, hauríem d'escriure 256 combinacions diferents).

Treball a desenvolupar de forma autònoma:

1. En un fitxer .vhd genereu les 7 entitats **inversor**, **and2**, **or2**, **and3**, **or3**, **and4** i **or4**, fent servir l'arquitectura **logica**, tal i com es mostra a l'exemple al principi del guió: senyals d'entrada **a**, **b** (per a les portes de 2 entrades); **a**, **b**, **c** (per a les portes de 3 entrades) i **a**, **b**, **c** i **d** (per a les de 4). La sortida serà **z** per a cadascuna de les entitats.

2. En un segon fitxer .vhd genereu ara un banc de proves **bdp_portes** (amb la seva arquitectura **test**) que us permeti de testar i mostrar el comportament individual de cada porta. A més, haureu de fer un altre banc de proves que permeti testar simultàniament TOTES les portes. Per fer-ho, utilitzeu quatre senyals **ent1**, **ent2**, **ent3** i **ent4** a les portes amb quatre entrades; **ent1**, **ent2** i **ent3** com a senyals d'entrada a les portes de tres entrades; i **ent1** i **ent2** com a senyals d'entrada a les portes de dues entrades i **ent1**, a l'inversor. Els senyals de sortida seran **sort_and2_logica** per a la porta AND de 2 entrades, realitzada amb l'arquitectura **logica**; **sort_or2_logica** per a la porta OR de 2 entrades, realitzada amb l'arquitectura **logica**; ... i així successivament. Això us donarà un total de 7 senyals de sortida, un per cada tipus de porta (**inversor**, **and2**, **or2**, **and3**, **or3**, **and4** i **or4**).

   ACLARIMENT: els senyals d'entrada de les *entitats* no tenen perquè tenir el mateix nom que els senyals al *banc de proves*.

Aquestes entitats, arquitectures i bancs de proves es faran servir en properes pràctiques.

En el banc de proves feu que el senyal **ent1** canviï cada 50ns, **ent2** cada 100ns, **ent3** cada 200ns, etc... (de manera semblant al exemple del banc de proves de l'inici del guió).

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 al menys 48 hores abans de la vostra sessió de pràctiques. Un cop passat aquest temps ja no serà possible pujar els fitxers. Els codis de la part autònoma s'avaluen. Els codis enviats FORA DEL TERMINI de les 48 hores prèvies a les sessions pràctiques es corregiran però NO S'AVALUARAN.

Recordeu que totes les trameses de fitxers es faran a través del campus virtual. NO ENVIEU ELS CODIS PER AVALUAR PER CORREU ELECTRÒNIC.

Els fitxers que haureu de pujar a través del campusvirtual, tal com us ho hem ensenyat a la pràctica 0, seran 2:

1) Un primer fitxer amb totes les entitats i arquitectures de les portes **inversor**, **and2**, **and3**, **and4**, **or2**, **or3** i **or4**. El nom del fitxer serà: **P1a_Cognom1_Cognom2_Nom_01.vhd**.
2) Un segon fitxer en el que posareu el banc de proves d'aquestes portes lògiques, amb el nom **P1a_Cognom1_Cognom2_Nom_02.vhd**.
3) Un cop pujats els fitxers, no oblideu de clicar "Enviar per avaluar".