Universitat de Girona

Escola Politècnica Superior

Dpt. Electrònica, Informàtica i Automàtica

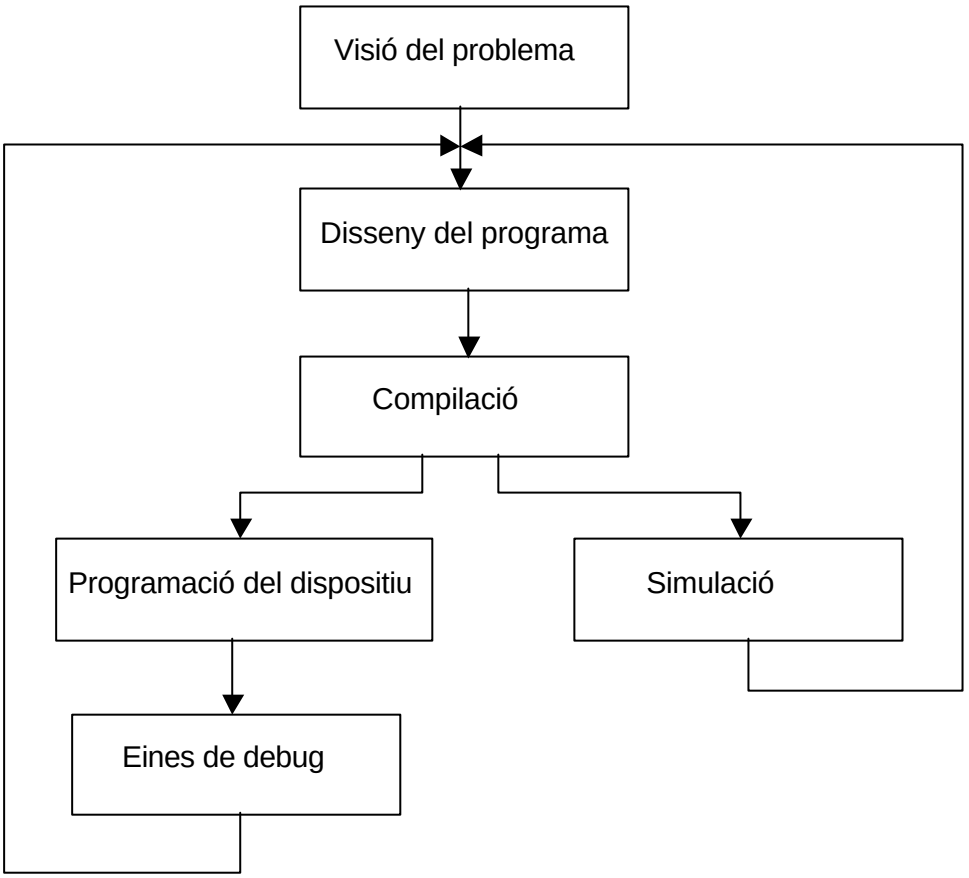
Grup de Robòtica i visió per Computador

Programació de dispositius d'ALTERA utilitzant el MAX+PLUS II

Document de:

Lluís Magí Carceller (**llmagi@eia.udg.es**)

PROGRAMACIÓ DE DISPOSITIUS FPGA's D'ALTERA



SOFTWARE PER A LA PROGRAMACIÓ DE DISPOSITIUS D'ALTERA

MAX+PLUS II

- **Entorn Windows 98, Windows NT, UNIX workstations.**

Eines de disseny

- **Editor d'esquemes**
- **Editor de text**
- ***Floorplan* editor**
- **Programació jeràrquica de dissenys**
- **Llibreria de mòduls parametrizables (LPM)**

Eines de compilació

- **Síntesis lògica i rutat automàtic del dispositiu**
- **Localització automàtica dels errors**
- **Ampliació de funcions genèriques (Megafunctions)**

Eines de verificació

- **Anàlisi temporal**
- **Creació de fitxers per a simuladors**

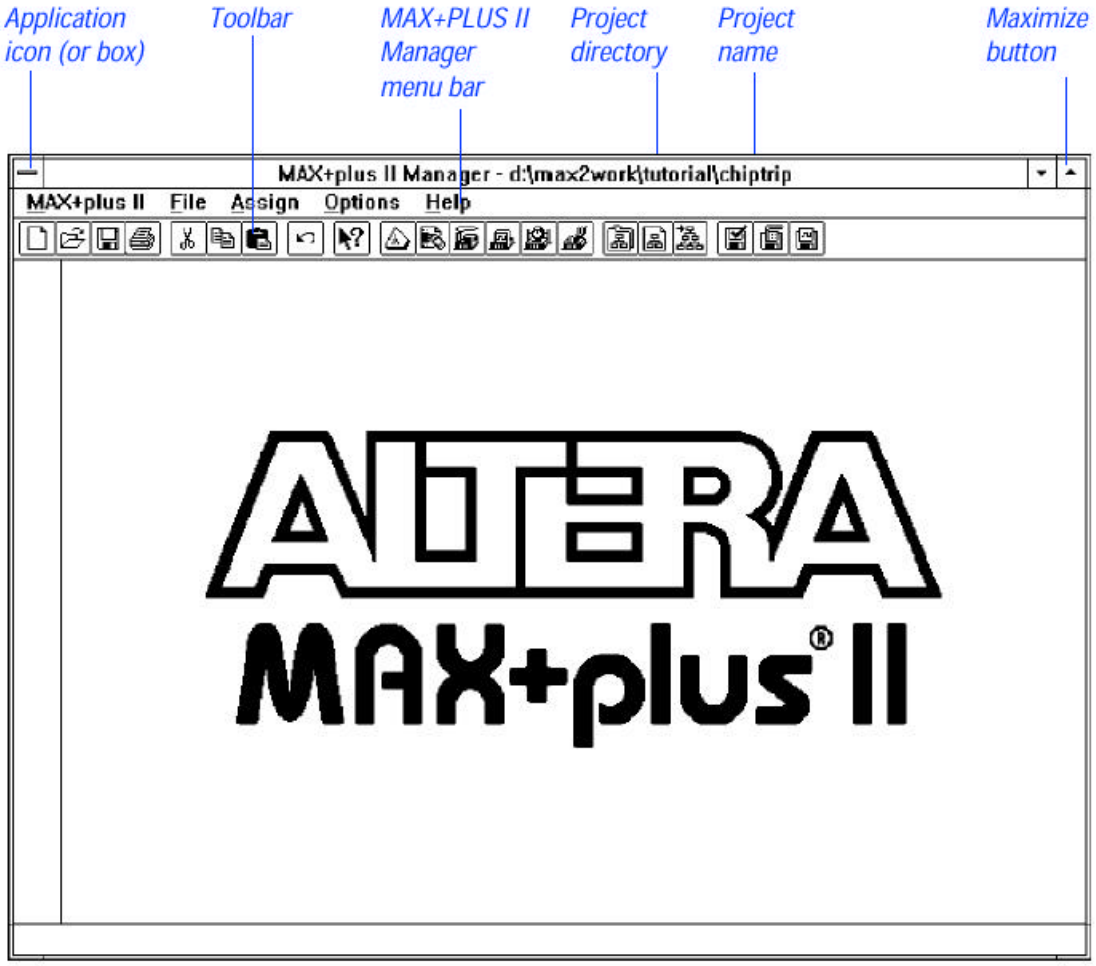
Eines de programació

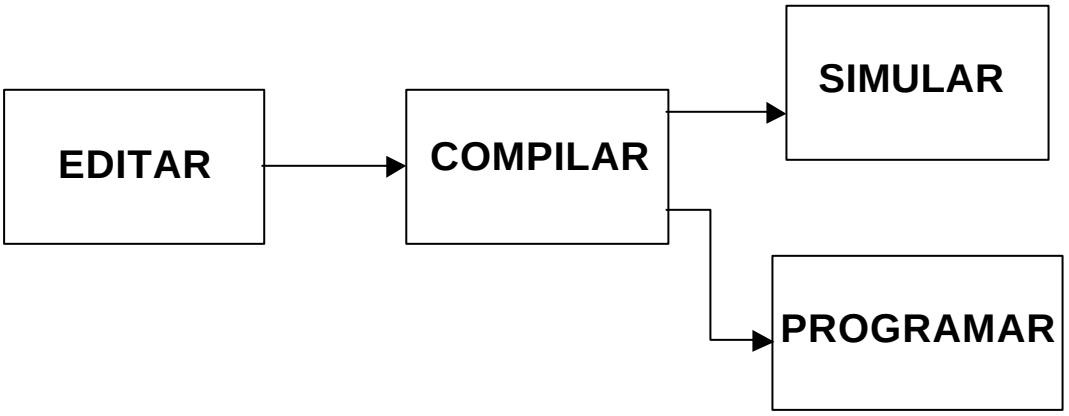
- **Programació i reprogramació dels dispositius**

Altres

- **Ajuda *on-line***

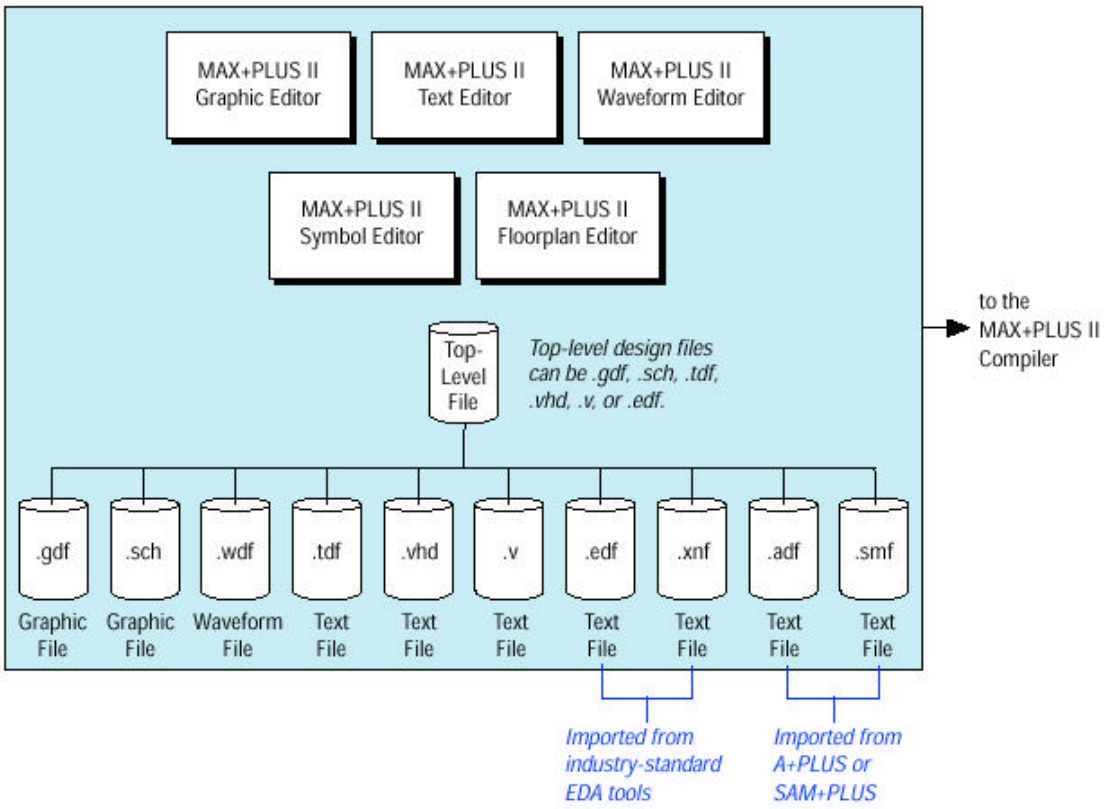
UTILITZACIÓ DEL SOFTWARE



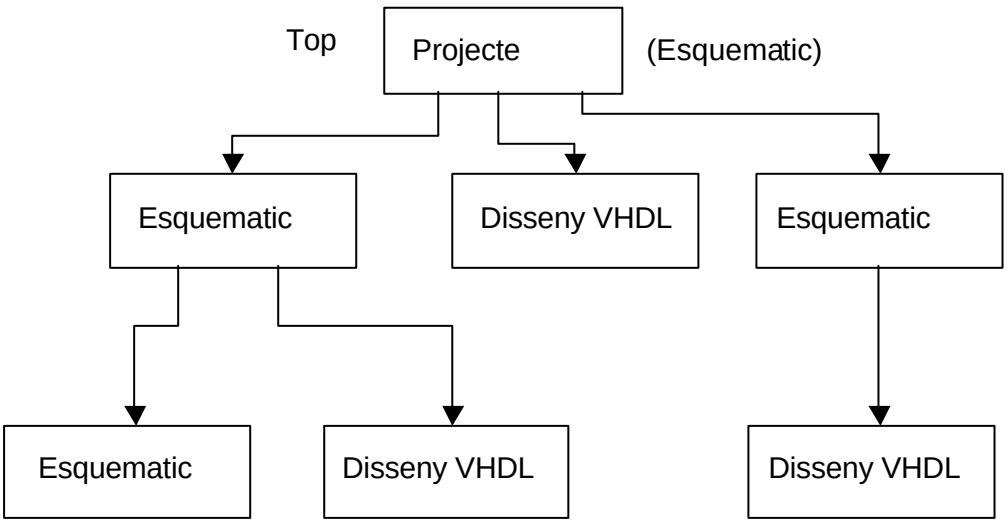


EDICIÓ

DISSENY DESCENDENT



Exemple de disseny



COMPILACIÓ

EL PROCÉSS DE COMPILACIÓ SEGUEIX LA JERARQUIA DEL DISSENY

Tipus de compilació

- **Funcional**
- **Completa**

Compilació Funcional:

Compila i crea les equacions funcionals

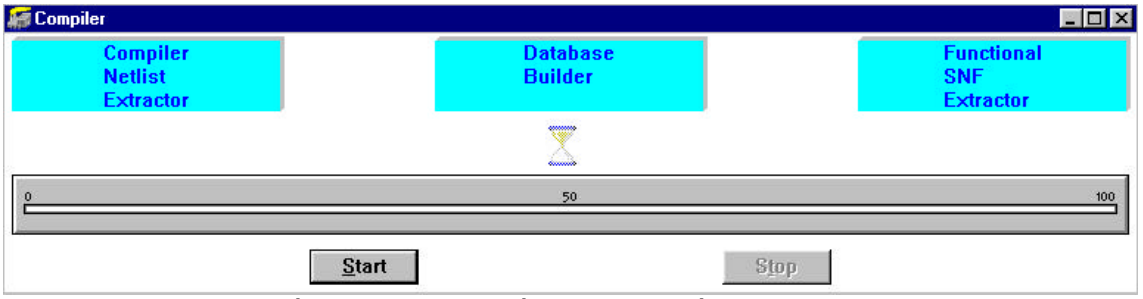
Utilització:

Compilat d'una part de projecte (Detecció d'errors i Simulació).

Realització:

Definir l'arxiu a compilar i fer click sobre l'icone.  A continuació activar la opció ***Functional SNF Extractor*** del menú ***Processing***.

La finestra de compilació quedaràde la següent manera:



Fent click sobre el botó **Start** començaràla compilació.

Si hi hagués algun error, apareix una segona finestra sota la de compilació amb la definició i localització dels errors.

Compilació Completa:

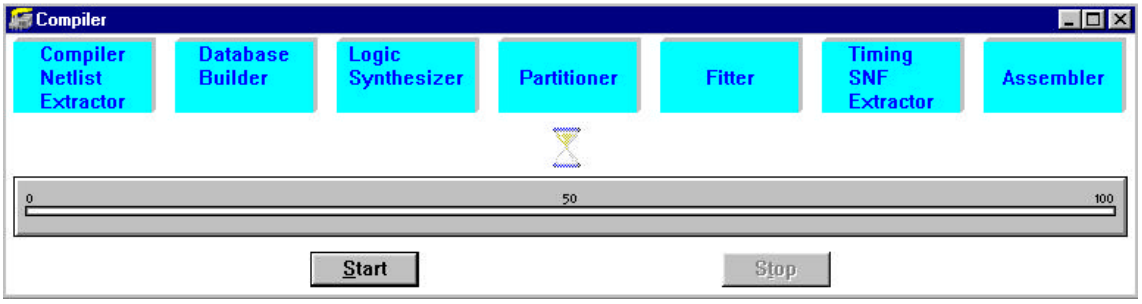
Bàsicament Compila, sintetitza, crea les equacions funcionals, realitza l'enrutat intern del dispositiu i crea els arxius de programació.

Utilització:

Compilat final del projecte.

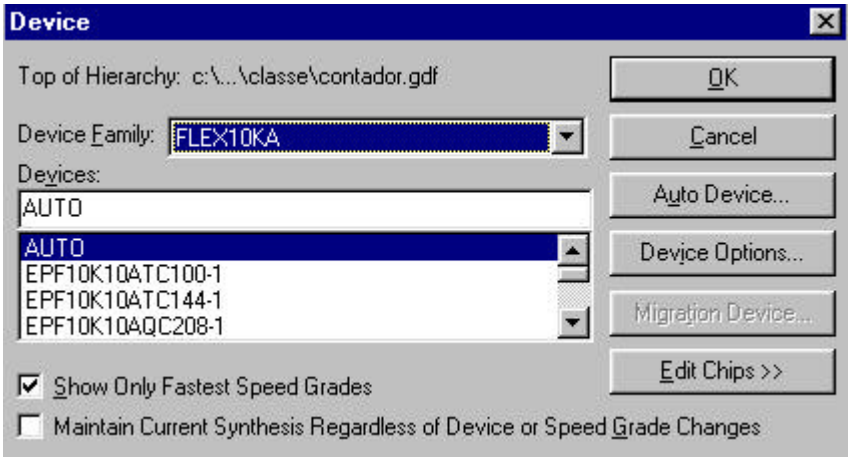
Realització:

Activar el mode de compilació utilitzant l'icone  i la opció ***Timing SNF Extractor*** del menú ***Processing***. La finestra del compilador presenta el següent aspecte:



Abans de realitzar la compilació, i per tal de completar-la amb èxit, cal triar el dispositiu que volem programar.

Ho farem amb el menú ***Assign → Device...*** amb el que es presenta la següent finestra:



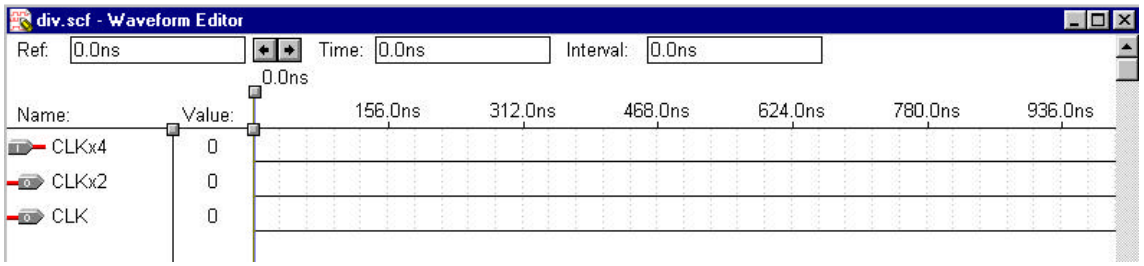
En aquesta podem triar el tipus de família i el dispositiu en concret.

Un cop triat podem fer click sobre el botó **Start** del compilador. Els errors es presenten com s'ha explicat anteriorment.

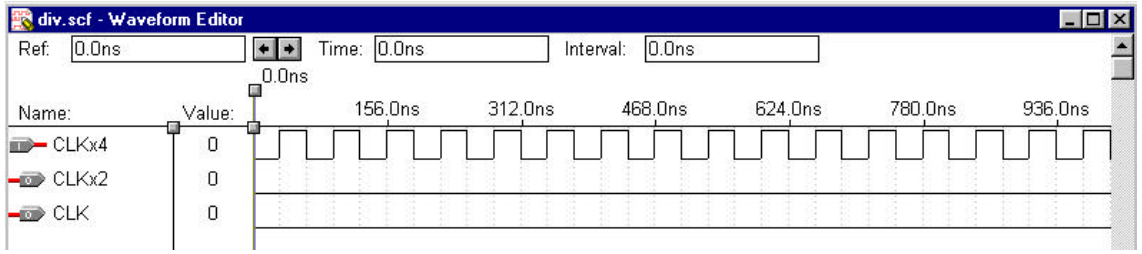
SIMULACIÓ

Per a simular un disseny, cal haver-lo compilat.

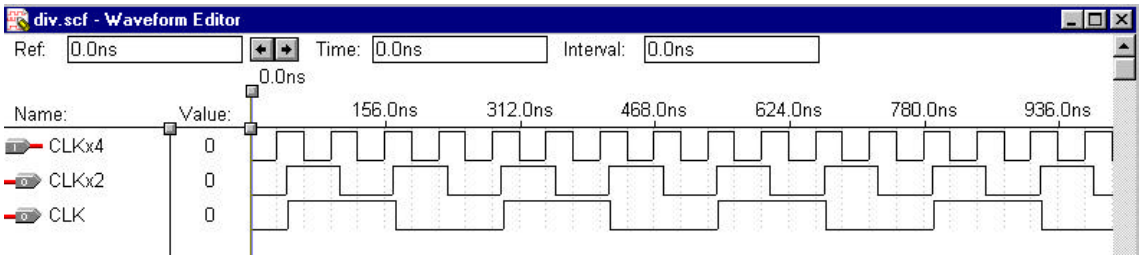
S'utilitza l'editor de formes d'ona ***Waveform editor***.



Es dibuixen les formes d'ona dels senyals d'entrada.



Es realitza la simulació.



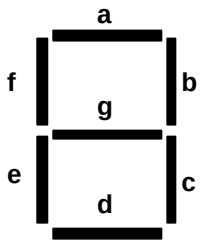
EXEMPLE D'APLICACIÓ

COMPTADOR/DESCOMPTADOR AMB SORTIDA PER DISPLAY 7 SEGMENTS

1. PLANTEIG DEL PROBLEMA

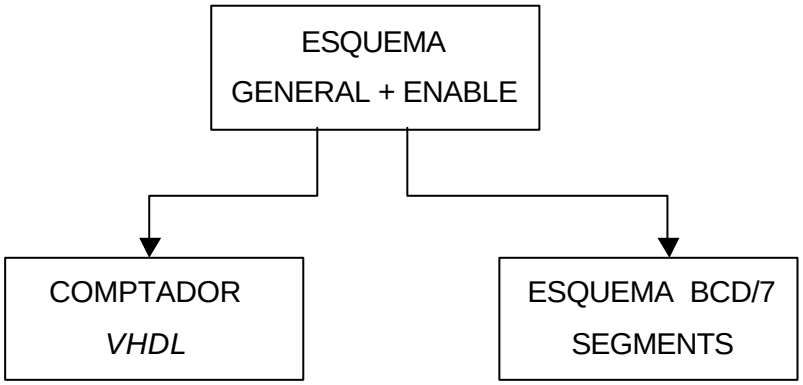
Realitzar un comptador/descomptador amb entrades *enable a nivell alt, enable a nivell baix*, *reset, clock i up/down.*

Sortida directe per a display 7 segments.



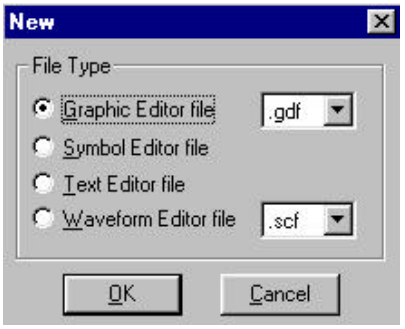
El disseny el separem en tres blocs. Funció *enable,* Comptador/descomptador BCD i conversor BCD a 7 segments.

Farem el disseny amb la següent jerarquia:



2. DISSENY DEL CONVERSOR BCD/7 SEGMENTS

Crear un nou arxiu amb **FILE → NEW...** i seleccionar **Graphic editor File (.gdf)**.



Apareix una finestra en blanc amb l'editor gràfic.

A continuació inserirem els components necessaris per a implementar el conversor fent un doble click sobre qualsevol lloc de la finestra i triant un component.

A exemple pràctic:

Les entrades es denominen ***input***

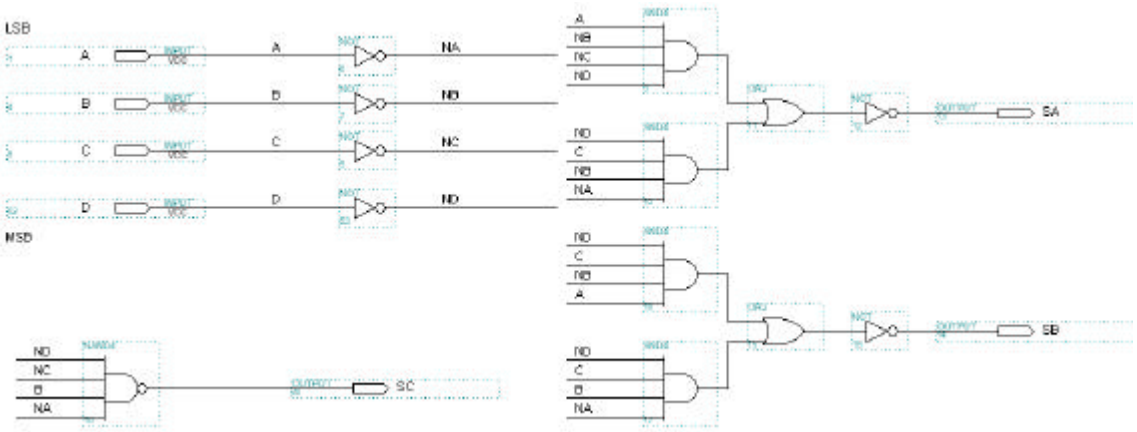
Les sortides ***output***

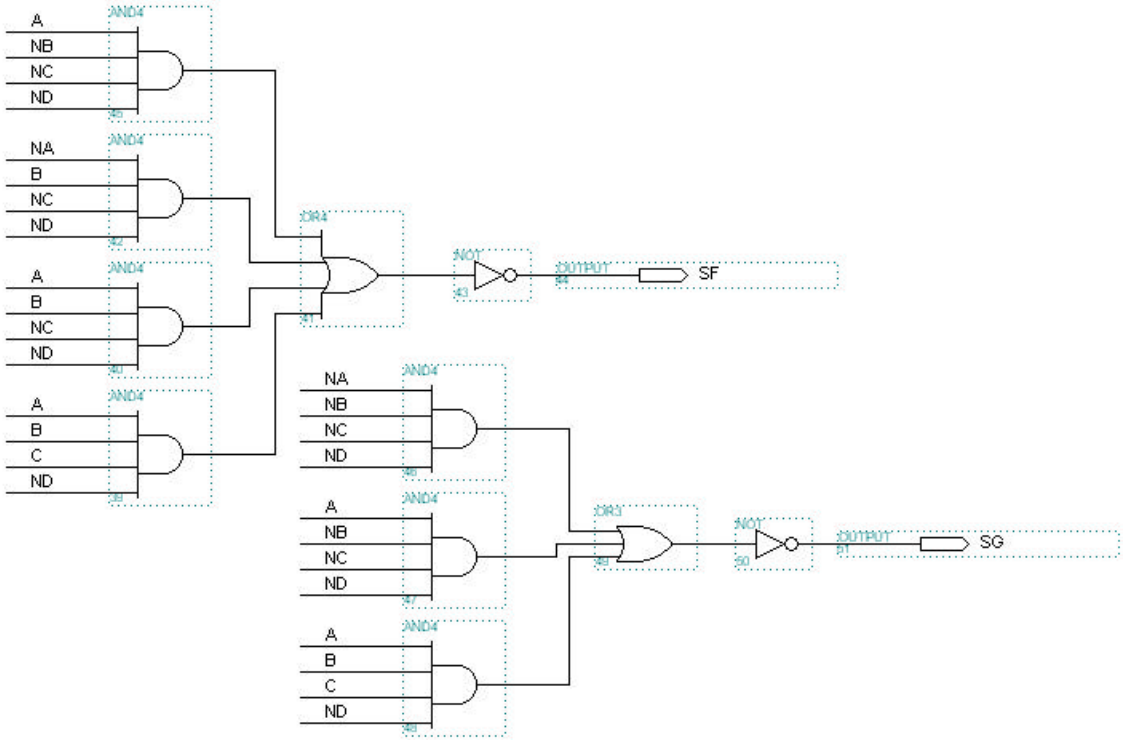
Una AND de dues entrades és ***AND2***

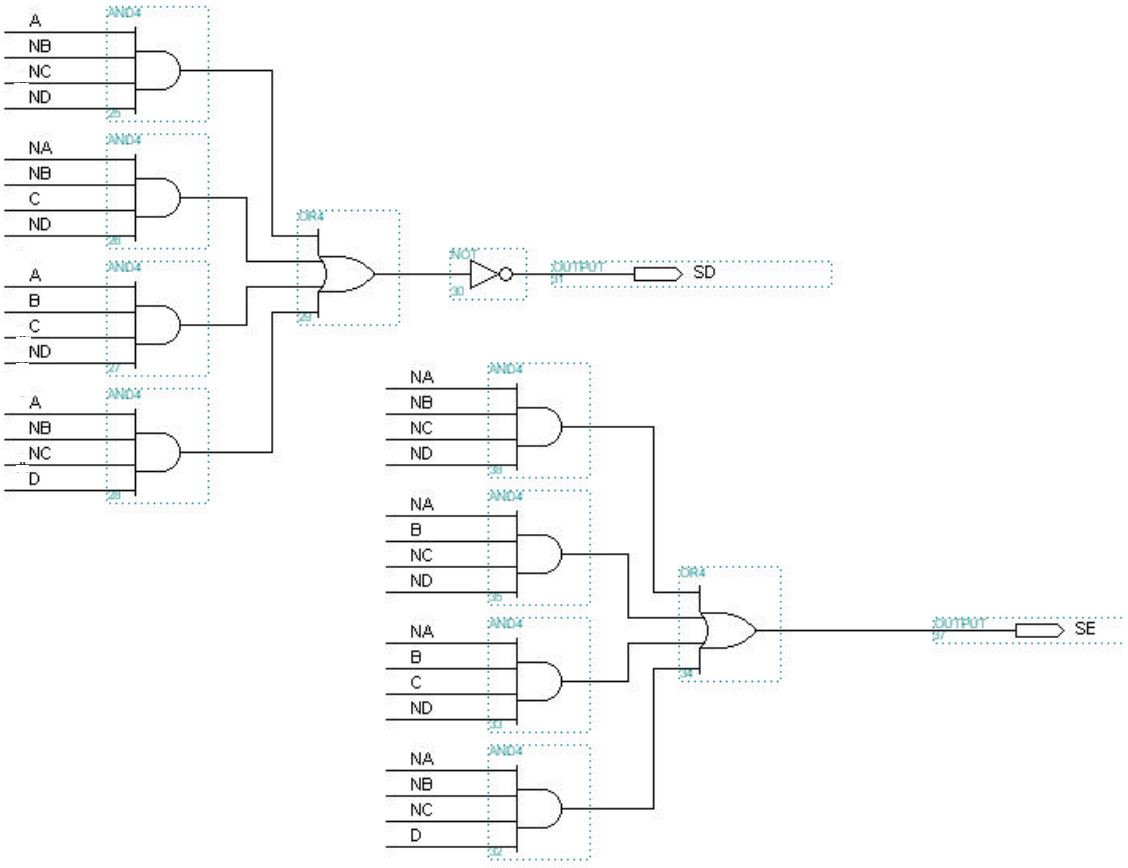
Etc.

Situant el mouse sobre una entrada o sortida d'un símbol, podrem dibuixar les línies d'unió.

Amb això podem realitzar un esquema com el següent:




A
NB
NC
ND
NA
B
NC
ND
A
B
NC
ND
A
B
C
ND
NA
NB
NC
ND
A
NB
NC
ND
A
B
C
ND
SF
SG


A
NB
NC
ND
NA
NB
C
ND
A
B
C
ND
A
NB
NC
D
NA
NB
NC
ND
NA
B
NC
ND
NA
B
C
ND
NA
NB
NC
D
SD
SE

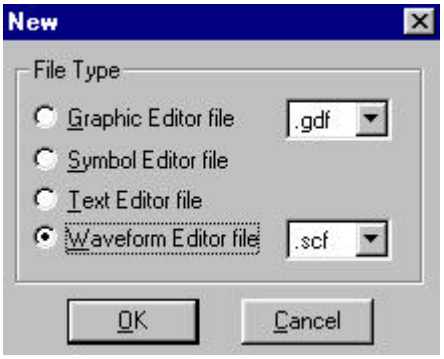
Un cop acabat, es **compila** aquest esquema per a verificar el seu funcionament i trobar possibles errors.

Primer el guardarem . Tot seguit, i recordant el seu sistema de compilació, farem que aquest arxiu sigui el més alt de la jerarquia utilitzant l'icone i finalment el compilarem **en mode funcional** com s'ha explicat anteriorment.

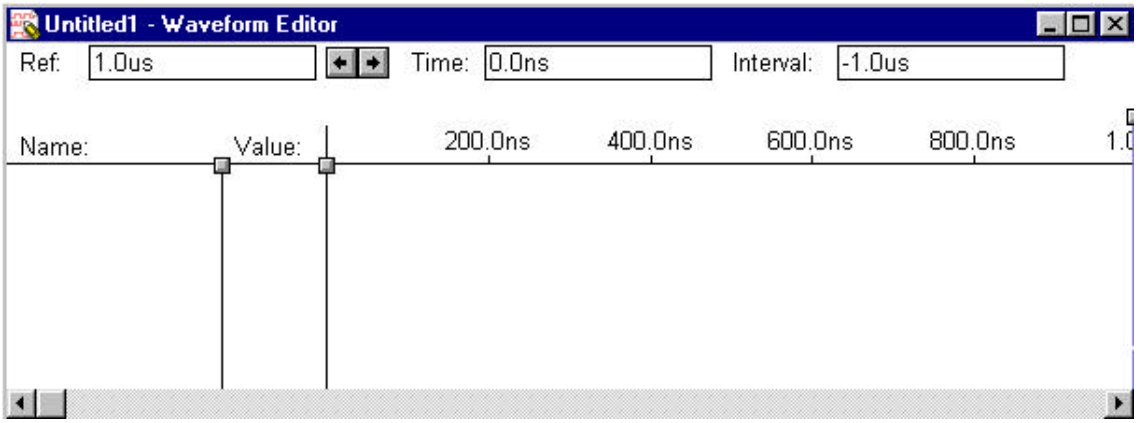
Un cop compilat sense cap error, passarem a la seva simulació per comprovar el correcte funcionament.

SIMULACIÓ

Creem un nou arxiu de tipus **Editor de forma d'ones** amb la opció ***New...*** del menú ***File***.

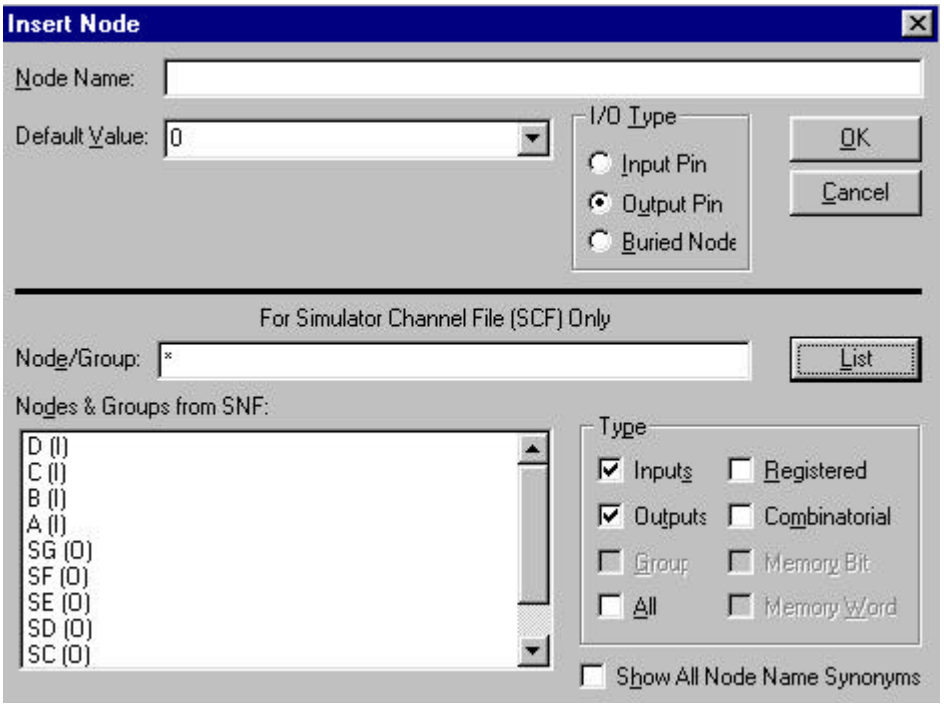


A continuació apareix una finestra com la següent:



En aquest editor podem inserir els senyals que volem veure, de la següent manera:

- Fem un doble click sobre la columna ***Name:*** i apareix la següent finestra:

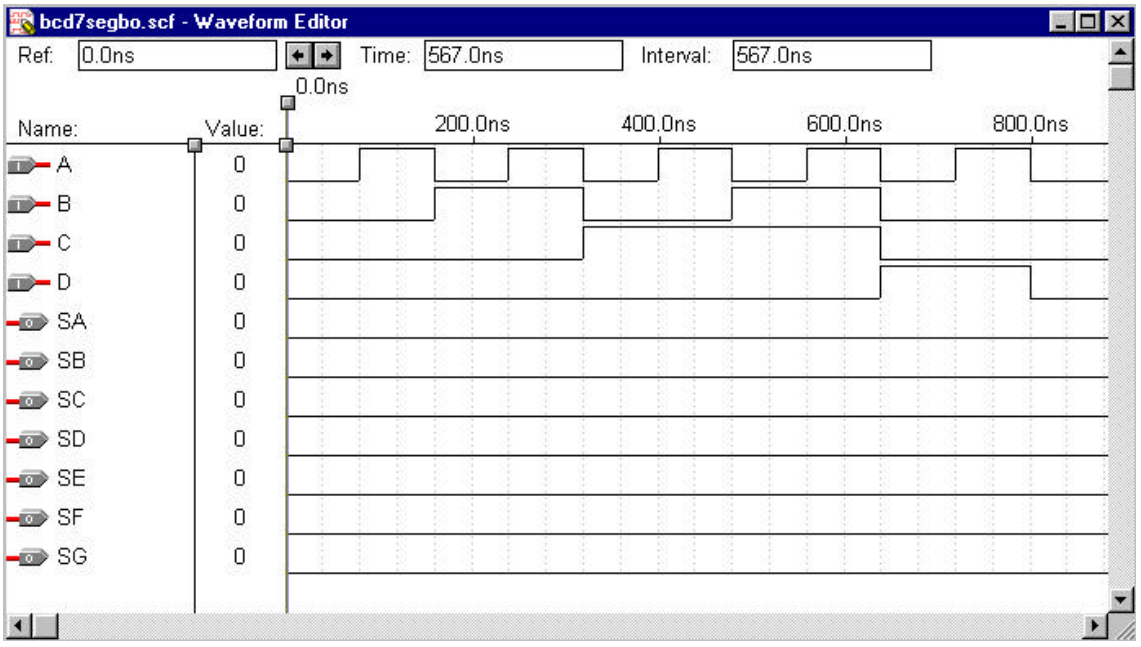


Fent un click sobre el botó **List** es presenten tots els senyals visibles i el seu tipus. Per exemple, en la figura anterior podem veure un senyal anomenat **A** del tipus *input* **(I)**.

Seleccionem l'apropiat i tot seguit el botó **OK**.

Repetim la operació fins a tenir els senyals desitjats.

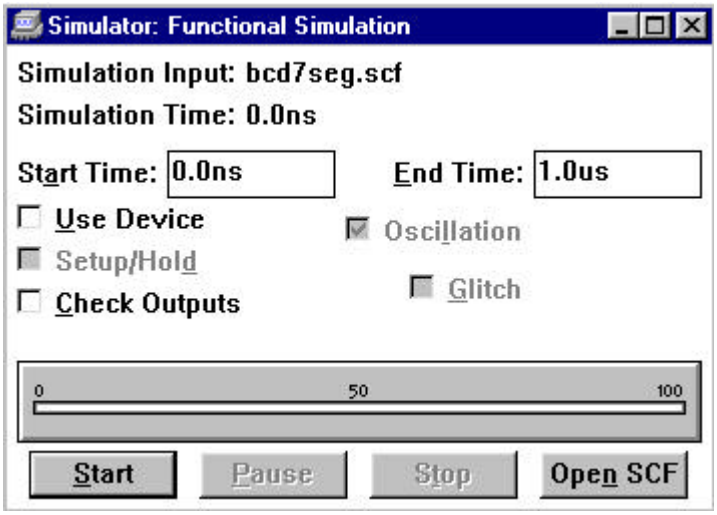
En el nostre exemple:



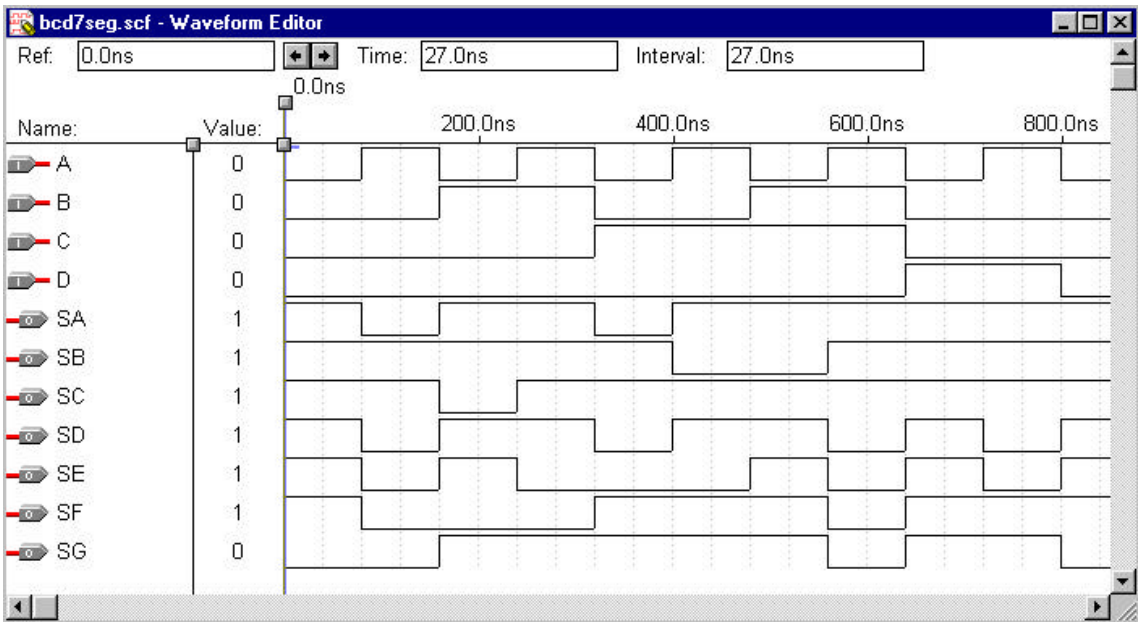
La modificació de la forma d'ona es realitza seleccionant la zona a modificar i utilitzant la barra d'eines que apareix a l'esquerra.

Si volem modificar tot el marge de temps podem seleccionar-ho directament fent click sobre el nom del senyal.

Un cop realitzada aquesta operació passarem a comprovar el seu funcionament mitjançant l'icone  , amb la qual apareix la finestra del simulador:



Finalment fem click sobre **Start** i es realitza la simulació. Finalitzada fem click sobre **Open SCF** i passem directament a veure els resultats:



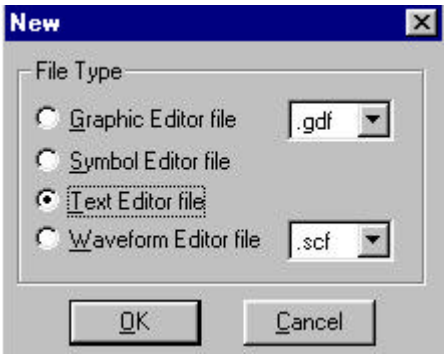
Un cop comprovat el seu perfecte funcionament, passarem a crear el símbol per a poder-lo introduir dins el disseny general fent: ***File → Create Default Simbol***

Fins aquí hem creat el símbol perfectament testejat del conversor.

3. Creació del comptador/descomptador.

Aquest bloc el crearem utilitzant el llenguatge VHDL.

Fem un arxiu nou del tipus texte:

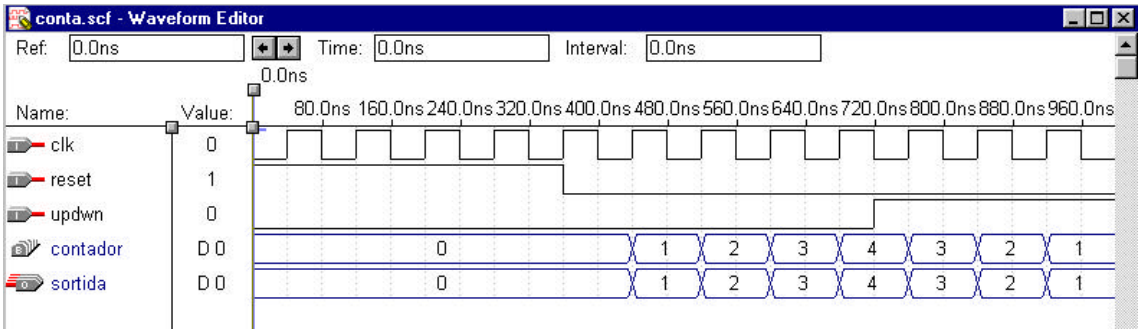


A continuació entrem el codi que realitza la funció que volem:

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;

ENTITY conta IS
   PORT(
      clk: in std_logic;
      updwn: in std_logic;    -- '0' incrementa
      reset: in std_logic;
      sortida: buffer integer range 0 to 9
   );

END conta;


ARCHITECTURE arch_conta OF conta IS

   signal contador: integer range 0 to 9;

BEGIN
   process(clk)
   begin
      if rising_edge(clk) then
         if reset='0' then
            case updwn is
            when '0' =>  -- incrementa
               if contador=9 then
                  contador <= 0;
               else
```

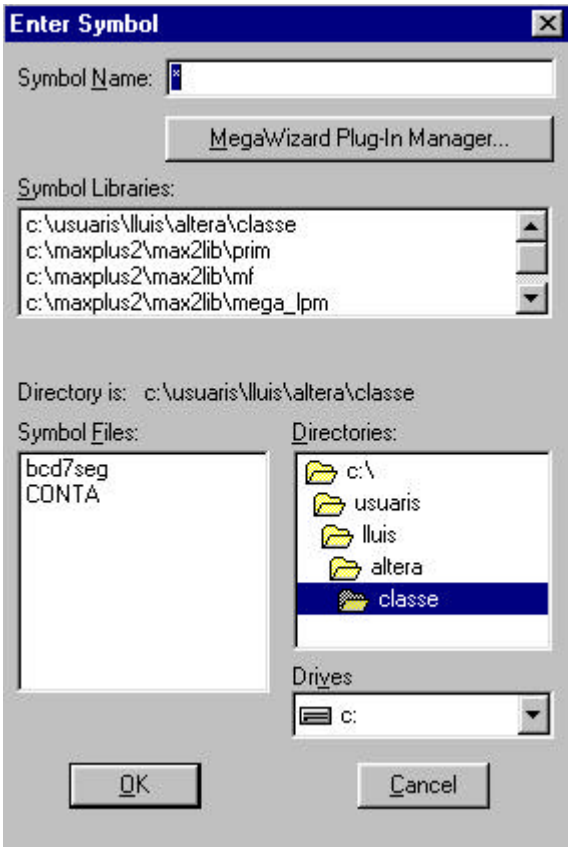
```
              contador <= contador + 1;
            end if;
          when others =>  -- decrementa
            if contador=0 then
              contador <= 9;
            else
              contador <= contador - 1;
            end if;
          end case;
        else
          contador <= 0;
        end if;
      end if;
    end process;

    process(reset, clk)
    begin
      if reset='1' then
        sortida <= 0;
      else
        sortida <= contador;
      end if;
    end process;
end arch_conta;
```

Fem aquest arxiu com a **arxiu més alt** tal i com s'ha fet anteriorment i el compilem i simulem. Un cop funcioni, creem el símbol.

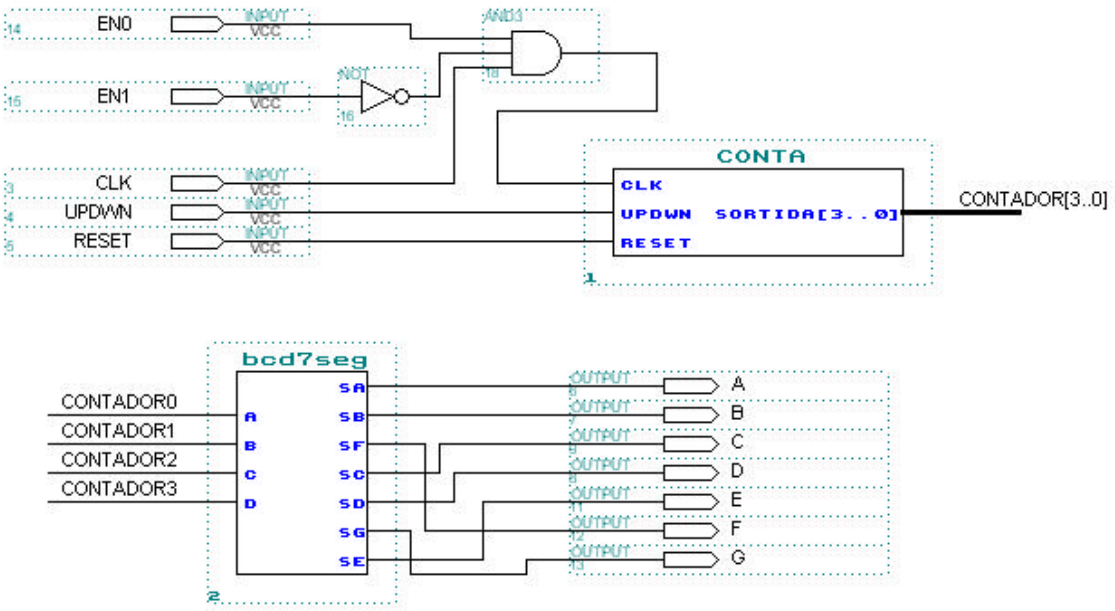


Finalment només queda crear l'esquema general del projecte.

Fem un nou arxiu de tipus gràfic i al fer un doble click per insertar components podrem triar els blocs fets anteriorment:

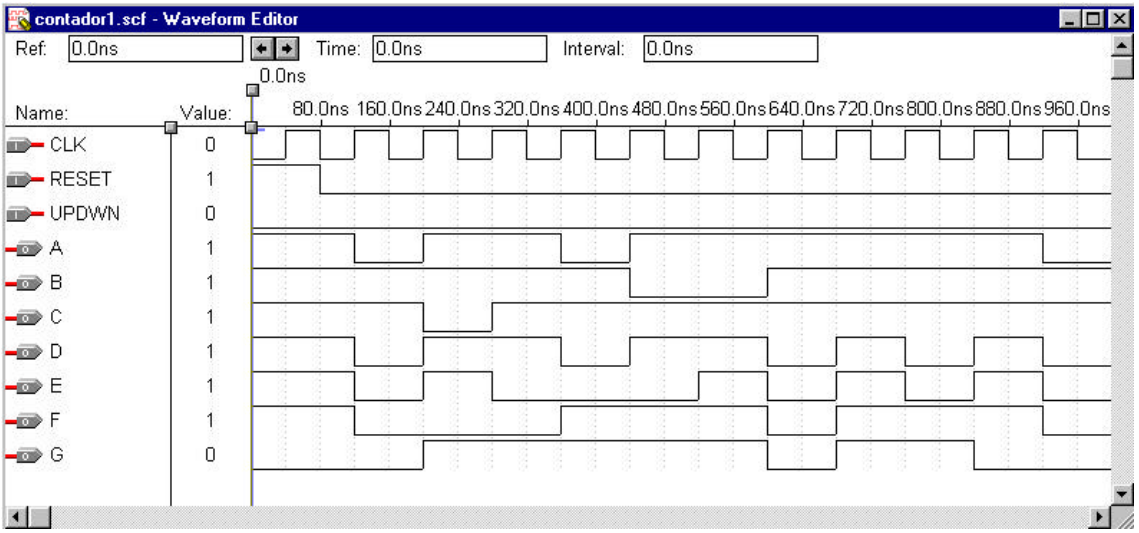


L'esquema general quedaràde la següent manera:

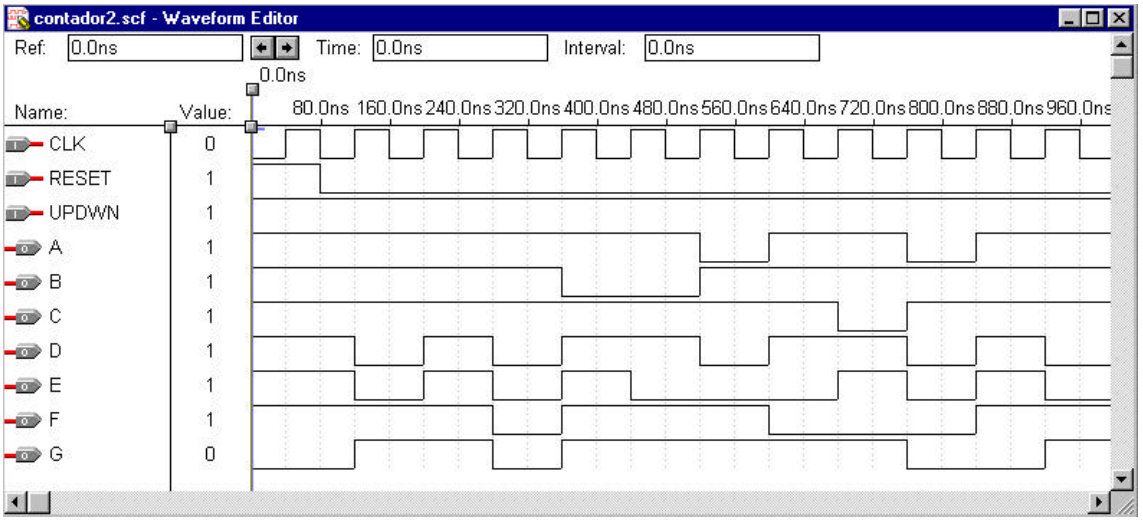


Només falta compilar **totalment** aquest esquema i ja estarà a punt per a ser volcat en un dispositiu.

La **simulació funcional** final quedarà



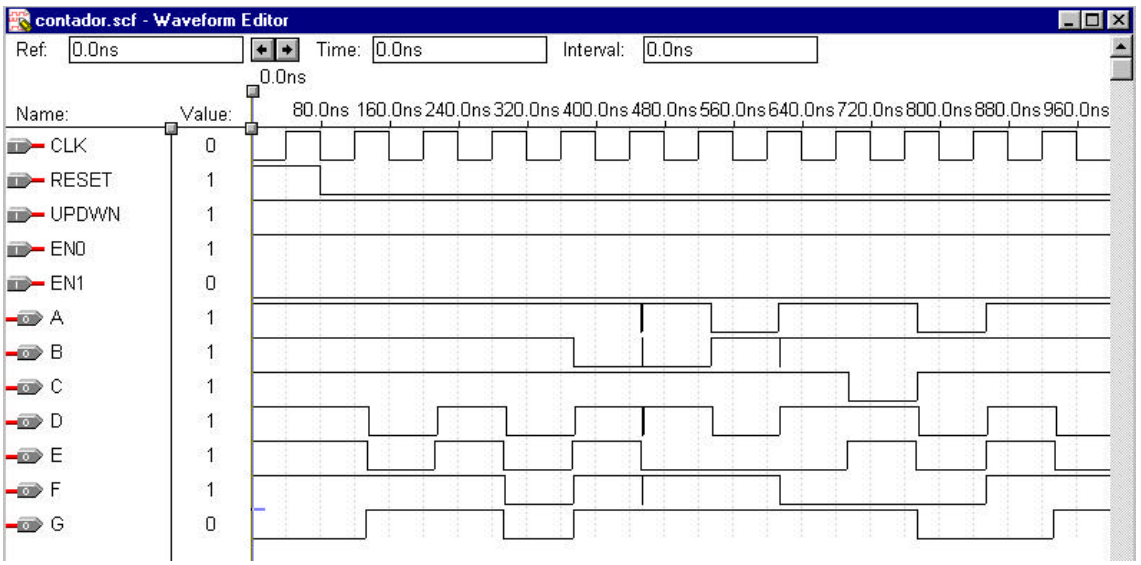
Simulació comptador



Simulació descomptador

Si volem volcar aquest disseny dins un dispositiu, l'haurem de compilar en mode total tal i com s'ha explicat anteriorment.

Un cop realitzada aquesta operació es poden veure els **resultats reals** en les simulacions:

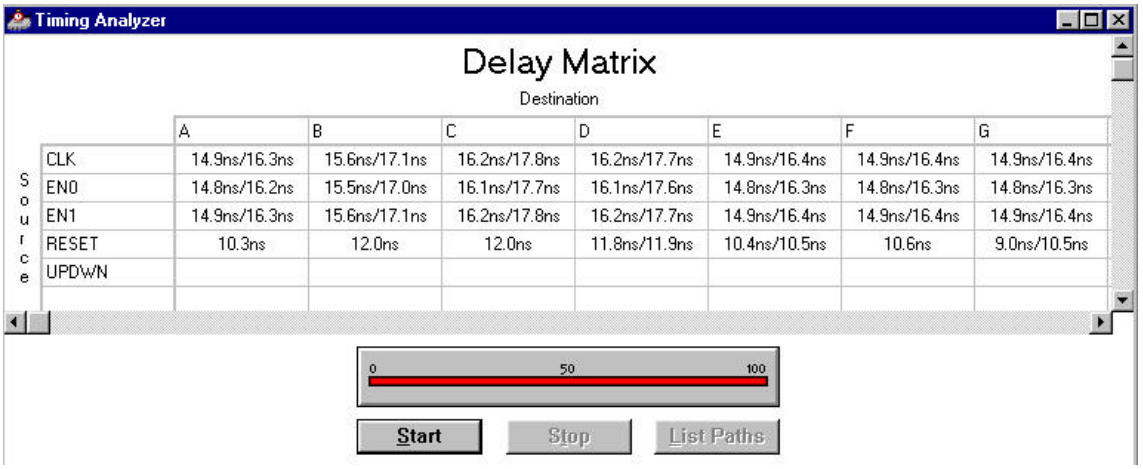


Es poden comprovar els retards que hi hauran a la sortida del dispositiu.

Altres eines

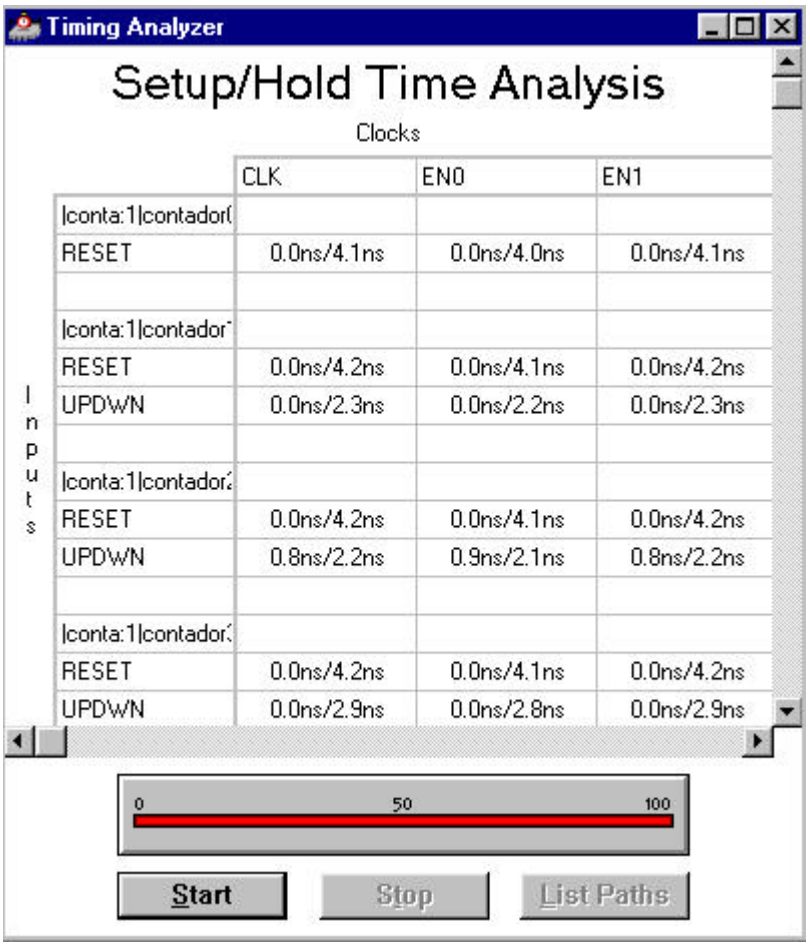
Després de la compilació total, podem realitzar vàries comprovacions com per exemple:

Matriu de retard



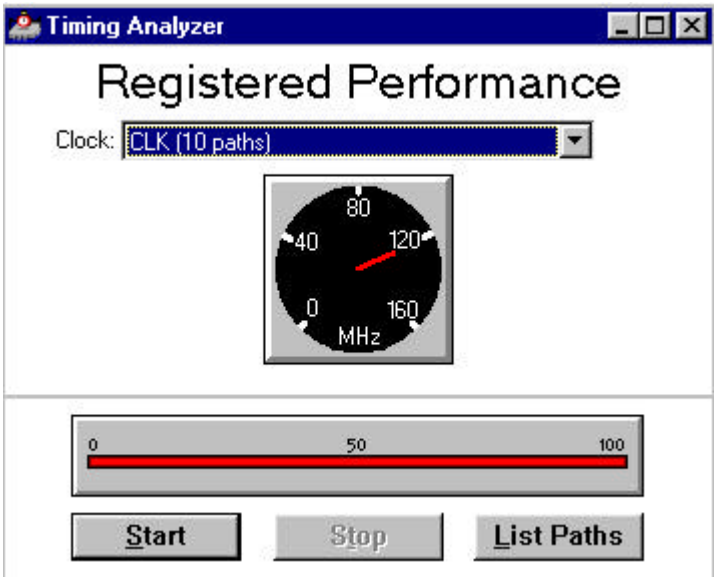
Ens presenta el retard de totes les sortides respecte la variació en una de les entrades.

Temps de Setup i Hold



Ens mostra el temps que una dada ha de ser estable abans d'un clock per tal que sigui considerada bona (*Setup Time*) i el temps que ha de ser estable per a poder realitzar les operacions correctament (*Hold time*).

Analitzador de velocitat de clock



Podem saber la velocitat màxima a la que podem fer anar el nostre disseny.