**الجلسة الثالثة:**

**تصميم الدارات الالكترونية بلغة (VHDL)**

**2**

**العمليات الأساسية المتوفرة في لغة (VHDL)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **المعاملات المنطقية** | not |  |  |  |  |  |
| and | Or | nand | nor | xor | xnor |
| **معاملات الإسناد** | =:  يستخدم لإسناد قيمة لمتغير أو ثابت أو لتعيين قيمة ابتدائية | <=  يستخدم لإسناد قيمة لإشارة | | <=  يستخدم لإسناد القيم إلى عناصر شعاع مفردة , كما يستخدم مع الكلمة المحجوزة others | | |
| **معاملات الإزاحة** | Sll  Shift left logic | Srl  Shift Right logic | Sla  shift left arithmetic | Sra  shift right arithmetic | Rol  Rotate left | Ror  Rotate  right |
| **المعاملات الحسابية** | +  جمع | -  طرح |  |  |  |  |
| \*  ضرب | /  قسمة | Mod  باقي القسمة مع إشارة المقسوم عليه | Rem  باقي القسمة مع إشارة المقسوم |  |  |
| \*\*  رفع إلى قوة | Abs  القيمة المطلقة |  |  |  |  |

كما توجد معاملات أخرى مثل معامل الضم & ومعاملات المقارنة (= و =/ و >و < و >= و <=) .

حاول أن تكتب بعض العمليات باستخدام المعاملات السابقة:

**التصريح عن الإشارات :**

الإشارات عبارة عن المتحولات الداخلية التي تستخدم لإيصال خرج إحدى المكونات الداخلية إلى دخل مكونة داخلية أخرى, و طريقة التصريح عن الإشارات موضحة في المثال التالي :

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

ENTITY TEST IS

PORT (A,B,C,D : IN STD\_LOGIC;

E : OUT STD\_LOGIC);

END TEST;

ARCHITECTURE BEHAVIOR OF TEST IS

**SIGNAL X,Y : STD\_LOGIC;**



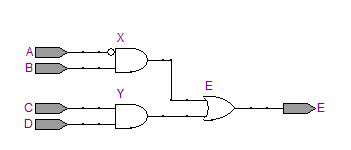
BEGIN

X <= (not A) AND B;

Y <= C AND D;

E <= X OR Y;

END BEHAVIOR;



هذا الشكل يوضح بنية التصميم السابق في برنامج Quartus و ذلك باستخدام RTL Viewer , و المطلوب انشاء ملف محاكاة و تطبيق نبضات على أقطاب الدخل و مراقبة إشارة الخرج .

**تعليمات استدعاء المكونات الداخلية اللازمة للتوصيف البنيوي :**

سنوضح هذه التعليمات من خلال المثال التالي :



LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

ENTITY **TEST** IS

PORT (A,B,C,D : IN STD\_LOGIC;

E : OUT STD\_LOGIC);

END **TEST**;

ARCHITECTURE **BEHAVIOR** OF **TEST** IS

SIGNAL X,Y : STD\_LOGIC;

**COMPONENT** Chip\_A

PORT (L,M,N : IN STD\_LOGIC;

O,P : OUT STD\_LOGIC);

**END COMPONENT;**

**COMPONENT** Chip\_B

PORT (Q,R,S : IN STD\_LOGIC;

T : OUT STD\_LOGIC);

END COMPONENT;

BEGIN

Chip1 : Chip\_A

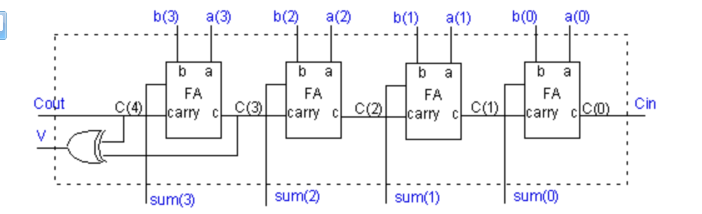
**PORT MAP (A,B,C,X,Y);**

Chip2 : Chip\_B

**PORT MAP (X,Y,D,E);**

END BEHAVIOR;

**مثال1 /دارة جامع بأربع خانات ثنائية/:**

****

في البداية نقوم بتصميم دارة جامع كامل لخانة واحدة

library ieee;

use ieee.std\_logic\_1164.all;

-- definition of a full adder

entity **FULLADDER** is

port (a, b, c: in std\_logic;

sum, carry: out std\_logic);

end **FULLADDER**;

architecture **fulladder\_behav** of **FULLADDER** is

begin

sum <= (a xor b) xor c ;

carry <= (a and b) or (c and (a xor b));

end **fulladder\_behav**;

ثم نقوم بتصميم دارة جامع لأربع خانات وذلك باستدعاء الجامع الذي قمنا بتصميمه سابقاً :

**library** ieee;

**use** ieee.std\_logic\_1164.**all**;

**entity** **FOURBITADD** **is**

**port** (a, b: **in** std\_logic\_vector(3 **downto** 0);

Cin : **in** std\_logic;

sum: **out** std\_logic\_vector (3 **downto** 0);

Cout, V: **out** std\_logic);

**end** **FOURBITADD**;

**architecture** **fouradder\_structure of** **FOURBITADD** **is**

**signal** c: std\_logic\_vector (4 **downto** 0);

**component** FULLADDER

**port**(a, b, c: **in** std\_logic;

sum, carry: **out** std\_logic);

**end** component;

**begin**

FA0: FULLADDER

**port map** (a(0), b(0), Cin, sum(0), c(1));

FA1: FULLADDER

**port map** (a(1), b(1), C(1), sum(1), c(2));

FA2: FULLADDER

**port map** (a(2), b(2), C(2), sum(2), c(3));

FA3: FULLADDER

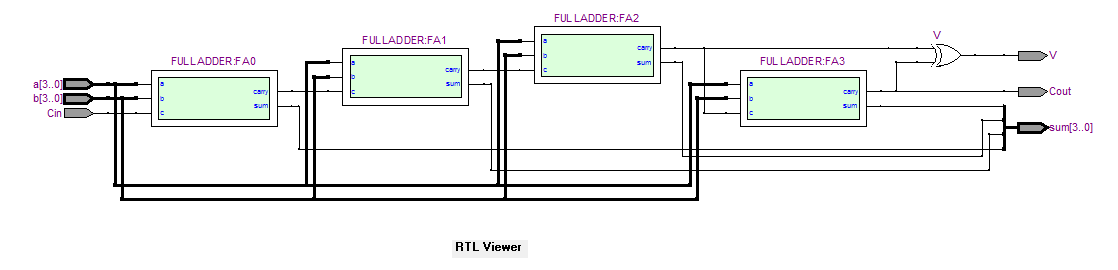
**port map** (a(3), b(3), C(3), sum(3), c(4));

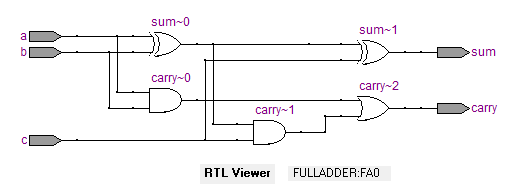
V <= c(3) **xor** c(4);

Cout <= c(4);

**end** **fouradder\_structure**;

هذا الشكل يوضح بنية التصميم السابق في برنامج Quartus و ذلك باستخدام RTL Viewer , و المطلوب انشاء ملف محاكاة و تطبيق نبضات على أقطاب الدخل و مراقبة إشارة الخرج .



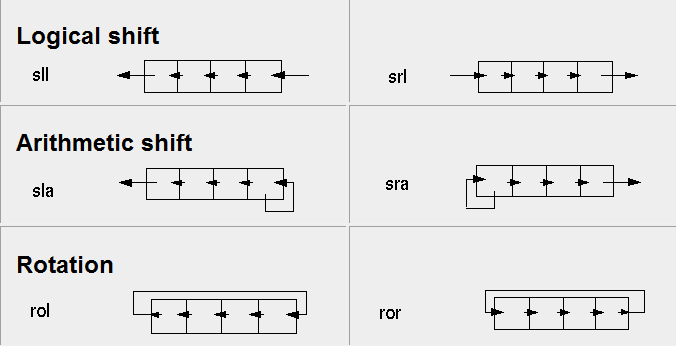
****

**الوظيفة**

توصيف دارة جامع كامل بنيوياً و محاكاتها عملياً ضمن برنامج Quartus .

**ملحقات:**

****

****

Reserved VHDL keywords