تصميم رقمي متقدم

Advanced Digital Design

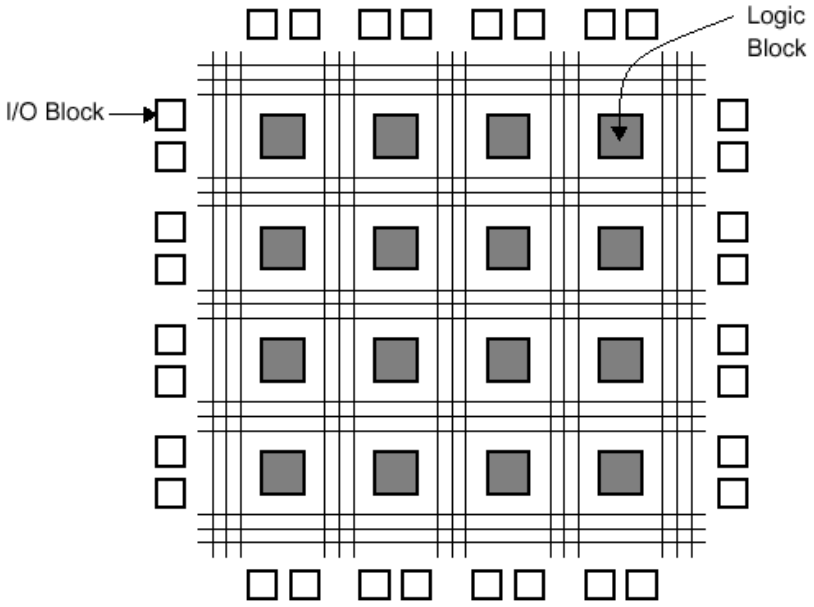
Dr.-Eng. Samer Sulaiman

2020-2021

مفردات المنهاج

- أساسيات التصميم الرقمي
- عناصر وتقنيات التصميم الرقمي التوافقي والتعاقبي (المتسلسل)
- **نمذجة التصميم الرقمي باستعمال لغة توصيف الكيان الصلب VHDL**
- المحاكاة الوظيفية والزمنية للأنظمة الرقمية

تصميم الأنظمة الرقمية باستخدام VHDL



- شرائح مصفوفات البوابات المنطقية القابلة للبرمجة حقلياً FPGA:
  - اختصار للعبارة التالية (Field Programmable Gate Arrays)
  - أي مصفوفات من البوابات المنطقية القابلة للبرمجة حقليًا
  - تحتوي على بلوكات قابلة للبرمجة مع مجموعة من الوصلات الداخلية القابلة للتعديل برمجياً.
  - تتكون البنية العامة لشرائح الـ FPGA داخليا :
    - كتل منطقية قابلة للبرمجة موزعة على هيئة مجموعة من الخلايا المنطقية تختلف بنيتها الداخلية تبعا للشركة الصانعة.
    - كتل التوصيلات البينية المسؤولة عن توصيل الكتل المبرمجة مع بعضها البعض ومع نقاط الدخل و الخرج
    - نقاط الدخل و الخرج
- أهمية شرائح الـ FPGA:
  - سهولة عملية تعديل التصميم حيث يمكن إعادة البرمجة ضمن النظام نفسه .
  - إمكانية بناء نظام كامل(hardware + software) على الشريحة دون الحاجة لدفع التكاليف الكبيرة المطلوبة لبناء نفس النظام باستخدام الشرائح المصنعة لتطبيقات مخصصة ASIC
  - معالجة عالية السرعة للإشارة حيث يمكن استخدام FPGA بدلا من الـ DSP، كذلك الـ FPGA أسرع من المتحكمات الصغرية microcontrollers لأنها تعتمد مبدأ البرمجة التفرعية، كما أن أبسط شريحة FPGA تملك أكثر بكثير من الإمكانيات المطلوبة لتقوم بعمل المتحكم الصغري بالإضافة إلى الموثوقية العالية في الأداء.

تصميم الأنظمة الرقمية باستخدام VHDL

- مراحل تصميم الأنظمة الرقمية باستخدام الـ FPGA:
  - وضع التصميم :
    - يعد الخطوة الأولى في التصميم، وهنا توجد طريقتان:
      - الطريقة الأولى رسم الهيكلية البنائية للدارة الرقمية Schematic Entry، وهذه الطريقة غير مجدية للتصميمات كبيرة الحجم التي تحتوي على العديد من المكونات و العناصر
      - الطريقة الثانية كتابة برنامج باستخدام إحد لغات الـ HDL و التي تصف ترتيب الدارة الرقمية باستخدام برامج مثل VHDL أو .... Verilog
  - المحاكاة (Simulation):
    - من الأفضل أن يتم اكتشاف الأعطال واصلاحها قبل أن تتم عملية البرمجة، وبهذه الطريقة يمكن توفير الوقت اللازم لبرمجة شرائح الـ FPGA لذلك نلجأ لعملية المحاكاة لاكتشاف الأعطال ومن ثم إصلاحها.
  - التشكيل (Synthesis):
    - هي عملية اكتشاف عناصر و مكونات الدارة التي تم تصميمها بواسطة لغة الـ VHDL لتحويل الوصف إلى دارة رقمية
  - وضع المكونات في أماكنها والربط بينها (Place and route):
    - هذه الخطوة تستخدم لمقابلة الدارات المصممة بالموارد المتاحة في الـ FPGA، و وضع المكونات في الأماكن المناسبة في الشريحة حيث يتم ربطهم سويا طبقا لتصميم الدارة باستخدام قنوات التوصيل والأسلاك الداخلية،
    - هذه الخطوة تربط كذلك بين أطراف التوصيل الخارجية للشريحة pins مع باقي أجزاء الدارة الداخلية التي سيتم توصيل الشريحة بها.
  - توليد ملف البرمجة (Programming File): يحتوي ملف البرمجة
    - على كل معلومات تصميم الدارة وكيف يتم مقابلة التصميم بالموارد الموجودة في FPGA
    - كيف ينبغي أن تتصل المفاتيح الداخلية فيها،
    - عبارة عن الملف الرقمي (bit stream) الذي يستعمل لبرمجة الشريحة
    - كل شريحة لها طريقة برمجة محددة ويتم تزويدها ببرنامج خاص لبرمجتها .

تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - مصطلح VHDL هو اختصار للعبارة : VHSIC HDL VHSIC (Very High Speed Integrated Circuits) Hardware Description Language
  - تعني لغة توصيف الكيان الصلب للدارات المتكاملة ذات السرعات المرتفعة جدا
  - تعدّ لغة الـVHDL لغة برمجة قياسية صممت من قبل وزارة الدفاع الأمريكية عام 1980، حيط تستعمل في توصيف سلوك و محاكاة عمل الأنظمة الرقمية (ابتداءً من البوابات البسيطة وانتهاءً بأعقد الأنظمة الرقمية)، و تصميم ومحاكاة دارات الـ VHSIC
  - تم اتخاذها لغة قياسية معتمدة من قبل IEEE في عام 1987، و تمت مراجعتها في الولايات المتحدة الأميركية عام 1993
  - في عام 1999 ظهر امتداد للغة VHDL وهو VHDL – AMS (Analog Mixed Signal)،
  - في عام 2008 تم إضافة خصائص جديدة ضمن المعيار IEEE Std 1076 – 2008
  - يوجد ن وعان من الأدوات التي تتعامل مع VHDL:
    - المحاكاة Simulation: لاختبار التصميم المنطقي باستخدام نماذج المحاكاة، ويمكن استخدام كل تعليمات VHDL
    - التركيب Synthesis: لتحويل الشيفرات إلى كيان صلب hardware

تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - مراحل توصيف الأنظمة الرقمية باستخدام اللغة VHDL:
    - التصريح عن المكتبات :
      - استدعاء مكتبات تحتوي العناصر و التعليمات الأساسية والقياسية المستخدمة في التصميم مثل . ieee , std , work
    - وحدة التوصيف الخارجي Entity:
      - فيها يتم توصيف المداخل و المخارج
    - وحدة التوصيف الداخلي Architecture:
      - تخصص هذ الوحدة للتعريف بسلوك أو بنية الدارة أو النظام المطلوب توصيفه انطلاقا من إشارات الدخل وصولا إلى إشارات الخرج وفق العلاقات الخاصة بالنظام
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - استدعاء المكتبات :
      - للتصريح عن مكتبة (أي لجعلها مرئية بالنسبة إلى التصميم) نكتب ما يلي :

- LIBRARY **library_name**;
  USE **library_name.package_name.package_parts**;

- مثال:

- library IEEE;
- use IEEE.STD_LOGIC_1164.ALL;

- بعض المكتبات تكون مرئية للتصميم افتراضيا default، لذلك لسنا بحاجة للتصريح عنها:
  - Std: مكتبة المصادر و تنسيق النصوص و بعض أنواع المعطيات، وذلك من أجل بيئة تصميم vhdl
  - Work: هي المكتبة التي نخزن فيها تصميمنا (ملف البرنامج) بالإضافة إلى كل الملفات المنشأة بوساطة برنامج الترجمة أو المحاكاة
- عادةً ما تتضمن المكتبات عدة أجزاء وأنماط معطيات وثوابت وعناصر جاهزة و توابع و إجرائيات ...

تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - استدعاء المكتبات :
      - تصنيف أنواع المعطيات المعرفة ضمن اللغة VHDL وفق المكتبات المتوفرة:
      - إن لغة VHDL تحتوي عدة أنماط للبيانات مسبقة التعريف و المحددة من خلال المعايير IEEE1076 وIEEE1164

القيمة المنطقية Logic Value	الوصف Description
U	Uninitialized
X	Forcing Unknown
0	Forcing Logic Low
1	Forcing Logic High
Z	Tri-state(High-Impedance)
W	Weak Unknown
L	Weak Low
H	Weak High
-	Don't Care

- مكتبة IEEE: تحتوي النظام المنطقي متعدد المستويات و أنواع المعطيات std _logic , std_ logic_vector
- مكتبة STD: تحتوي أنواع المعطيات Bit ,Boolean ,Integer ,Real ,Signed ,Unsigned والعمليات الحسابية و التوابع الأساسية للتحويل بين الأنماط.
- المنطق المعياري STD_LOGIC: نوع من المعطيات متعدد القيم ذو 9 مستويات موضحة بالجدول التالي :

	X	0	1	Z	W	L	H	-
X	X	X	X	X	X	X	X	X
0	X	0	X	0	0	0	0	X
1	X	X	1	1	1	1	1	X
Z	X	0	1	Z	W	L	H	X
W	X	0	1	W	W	W	W	X
L	X	0	1	L	W	L	W	X
H	X	0	1	H	W	W	H	X
-	X	X	X	X	X	X	X	X

- عندما يحدث تعرض بين إشارتين موصولتين إلى نفس العقدة سيتم حله آليا وفق الشكل التالي:

تصميم الأنظمة الرقمية باستخدام VHDL

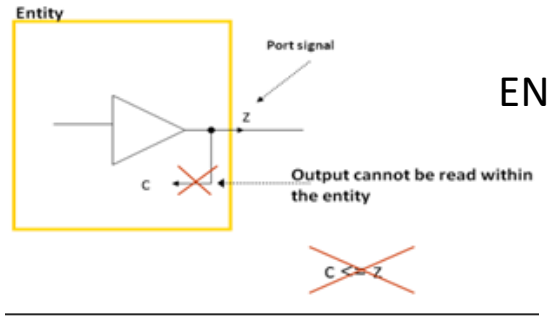
- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - استدعاء المكتبات :
    - بعض الملاحظات المهمة عند كتابة شيفرة VHDL:
      - يمكن استخدم دائما نوع المعطيات std_logic أو std_logic_vector من أجل كل منافذ الدخل والخرج، أما باقي أنواع المعطيات يمكن استخدامها داخل وحدة التوصيف الداخلي عند الحاجة أو مع الثوابت العامة.
    - وحدة التوصيف الخارجي Entity:
      - فيها يتم توصيف المداخل والمخارج وفق النمط التالي

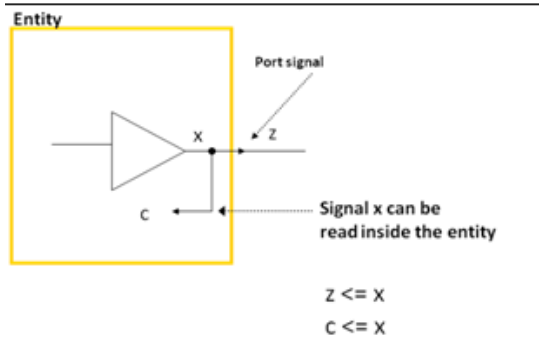
```
ENTITY entity_name IS
PORT (
        port_name : port_mode signal_type;
        port_name : port_mode signal_type;
        ……
        port_name : port_mode signal_type);
END entity_name;
```

تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - وحدة التوصيف الخارجي Entity:
      - مثال: بوابة NAND بمدخلين

```
ENTITY nand_gate IS
        PORT( a : IN STD_LOGIC;
        b : IN STD_LOGIC;
        z : OUT STD_LOGIC );
END nand_gate;
```

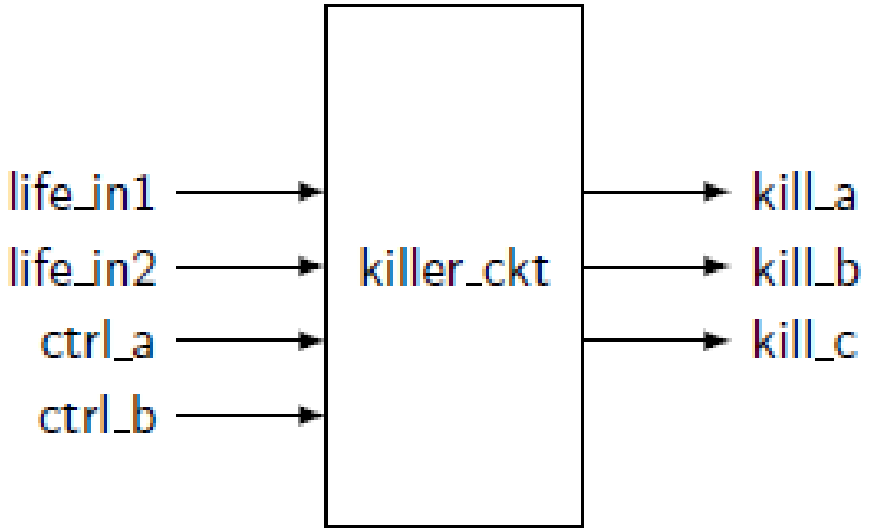




- أنواع أنماط الإشارة التي تصف اتجاه نقل البيانات بالنسبة إلى العنصر الموصوف
  - دخل IN: يشير إلى أن الإشارة المطبقة على هذا القطب تتجه إلى داخل النظام
  - خرج OUT: يشير إلى أن الإشارة الناتجة من هذا القطب تتجه إلى خارج النظام
    - لا يمكن قراءة الخرج ضمن الـ Entity لذلك يتم استخدام تعريف الإشارات كما هو موضح بالشكل
  - ثنائي الاتجاه INOUT: يشير إلى أن الإشارة المطبقة على هذا القطب ثنائية الاتجاه يمكن القراءة منها أو الكتابة عليها
  - عازل Buffer: يشير إلى أن الإشارة المطبقة على هذا القطب تشكل إشارة يمكن قرائتها من قبل وحدات توصيف خارجي اخر (أي توصيل بين الوحدات وليس من الدخل إلى الخرج او بالعكس)
    - هنا يمكن قراءة العازل ضمن الـ Entity

تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - وحدة التوصيف الخارجي Entity:
    - مثال: أكتب شيفرة الـ VHDL الخاصة بالدارة التالية:



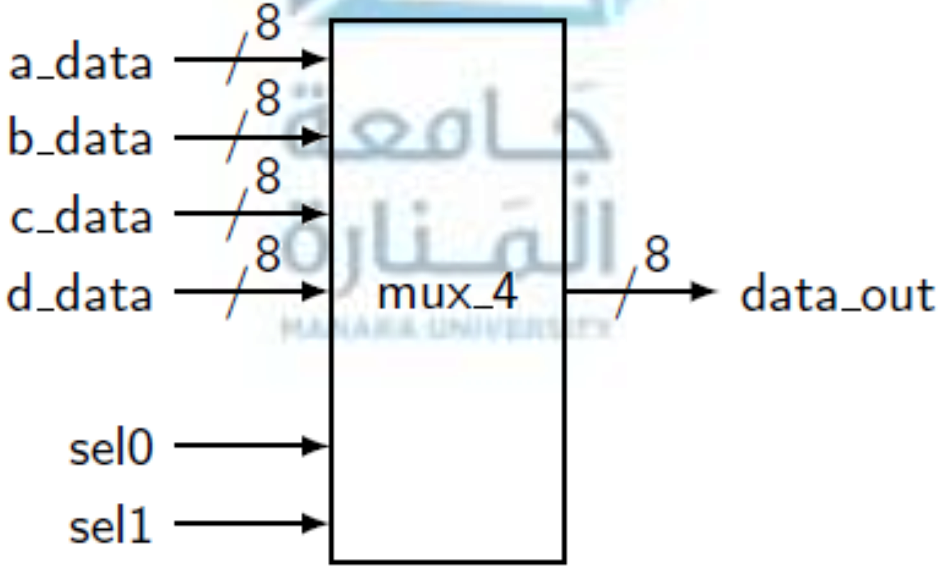
تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - وحدة التوصيف الخارجي Entity:
      - مثال: أكتب شيفرة الـ VHDL الخاصة بالدارة التالية:
        - الحل:

```
entity killer_ckt is
    port (
                life_in1 : in std_logic;
                life_in2 : in std_logic;
                crtl_a, ctrl_b : in std_logic;
                kill_a : out std_logic;
                kill_b, kill_c : out std_logic);
end killer_ckt;
```

تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - وحدة التوصيف الخارجي Entity:
      - مثال: أكتب شيفرة الـ VHDL الخاصة بالدارة التالية:



تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - وحدة التوصيف الخارجي Entity:
      - مثال: أكتب شيفرة الـ VHDL الخاصة بالدارة التالية:
      - الحل:

```
entity mux4 is
    port ( a_data : in std_logic_vector(0 to 7);
                b_data : in std_logic_vector(0 to 7);
                c_data : in std_logic_vector(0 to 7);
                d_data : in std_logic_vector(0 to 7);
                sel1,sel0 : in std_logic;
                data_out : out std_logic_vector(0 to 7));
end mux4;
```

تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - وحدة التوصيف الداخلي Architecture:
      - عبارة عن وصف يعبر عن كيفية عمل الدارة أي توضيح لسلوك الدارة
      - طرق توصيف النظام داخليا
        - توصيف دارة ما من خال استدعاء عدد من المكونات الأولية والربط فيما بينها لتشكيل البنية المطلوبة
        - توصيف البنية الداخلية لهذه الدارة من خلال اتباع مخطط منهجي يوضح آلية العمل المطلوبة بغض النظر عن البنية
        - توصيف نظام منطقي ما من خال جملة معادلات منطقية،
        - كما يمكن المزج بين الطرق السابقة جميعها .
      - إذا يمكن توصيف النظام داخليا بعدة طرق :
        - توصيف بنيوي .
        - توصيف سلوتي .
        - توصيف رياضي منطقي .
        - توصيف مختلط .
      - يتم التصريح عنه بالشتل التالي :

- ARCHITECTURE **architecture_name** OF **entity_name** IS
  [declarations]
  BEGIN
  code
  END **architecture_name**;

تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - وحدة التوصيف الداخلي Architecture:
      - مثال: بوابة NAND:

-
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY nand_gate IS
PORT(
        a : IN STD_LOGIC;
        b : IN STD_LOGIC;
        z : OUT STD_LOGIC);
END nand_gate;
ARCHITECTURE dataflow OF nand_gate IS
BEGIN
        z <= a NAND b;
END dataflow;
```

تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - في لغة VHDL إذا وضع (- -) قبل سطر التعليمة فهذا يدل على أن السطر هو تعليقات أو ملاحظات.
    - يجب أن تنتهي جميع أسطر التعليمات والتصريحات بفاصلة منقوطة (;)
    - عند التصريح عن أكثر من متحول تستخدم الفاصلة للفصل بينهم (,)
    - إسناد قيمة ما إلى إشارة رمزها(=>)، بينما إسناد قيمة ما إلى متحول والتصريح عن قيمة ثابتة ولتعيين قيمة ابتدائية
    - رمزها (=:)
    - يمكن استخدام الأرقام و الحروف و(_) للتصريح عن أسماء المتحولات و الإشارات
      - لا يمكن أن يبدأ الاسم برقم ، وكذلك لا يمكن استخدام كلمة محجوزة في الاسم .
    - لغة VHDL غير حساسة لحالة الحروف (صغيرة أو كبيرة)
    - خلافاً لبرامج الحاسوب العادية التي تكون متسلسلة التنفيذ فإن لغة الـ VHDL تنفذ التعليمات على التفرع لذلك نقول VHDL code بدلا من VHDL program
    - العمليات الأساسية المتوفرة في لغة VHDL

	Operator Class	Operator
Highest precedence	Miscellaneous	**, ABS, NOT
	Multiplying	*, /, MOD, REM
	Sign	+, −
	Adding	+, −, &
	Shift	SLL, SRL, SLA, SRA, ROL, ROR
	Relational	=, /=, <, <=, >, >=
Lowest precedence	Logical	AND, OR, NAND, NOR, XOR, XNOR

تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - التصريح عن الإشارات ضمن لغة الـ VHDL:
      - الإشارات عبارة عن المتحولات الداخلية التي تستخدم لإيصال خرج إحد المكونات الداخلية إلى دخل مكون داخلي أخر
      - يتم ذلك كما يلي:

```
SIGNAL a : STD_LOGIC;
SIGNAL b : STD_LOGIC_VECTOR(7 DOWNTO 0);
```

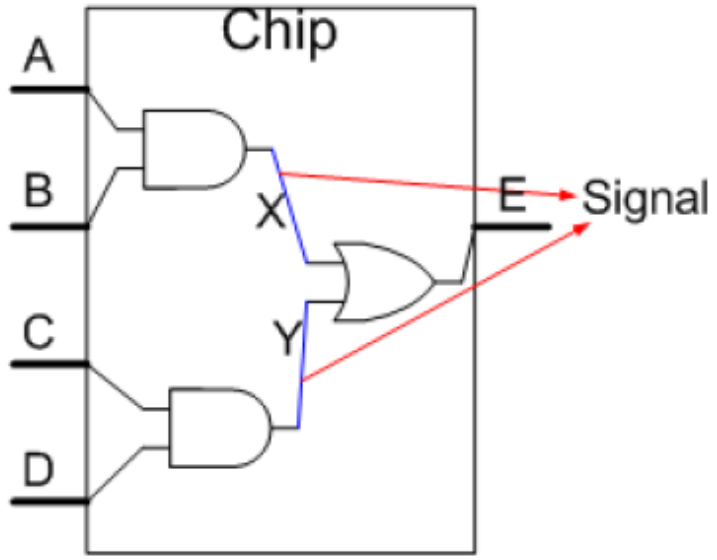
      - أمثلة عن التصريح عن إشارات مختلفة:

```
SIGNAL a: STD_LOGIC;
SIGNAL b: STD_LOGIC_VECTOR(3 DOWNTO 0);
SIGNAL c: STD_LOGIC_VECTOR(3 DOWNTO 0);
SIGNAL d: STD_LOGIC_VECTOR(15 DOWNTO 0);
SIGNAL e: STD_LOGIC_VECTOR(8 DOWNTO 0);
a <= '1';
b <= "0000"; -- Binary base assumed by default
c <= B"0000"; -- Binary base explicitly specified
d <= X"AF67"; -- Hexadecimal base
e <= O"723"; -- Octal base
```

تصميم الأنظمة الرقمية باستخدام VHDL

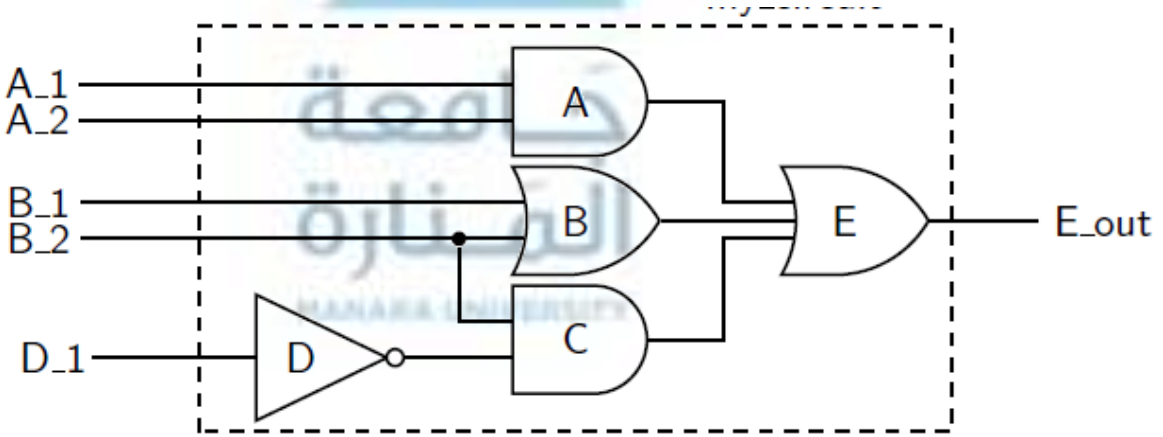
- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - التصريح عن الإشارات ضمن لغة الـ VHDL:
    - مثال أكتب شيفرة الـ VHDL للدارة المنطقية التالية:

```
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;
```



تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - التصريح عن الإشارات ضمن لغة الـ VHDL:
    - مثال أكتب شيفرة الـ VHDL للدارة المنطقية التالية:



تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - التصريح عن الإشارات ضمن لغة الـ VHDL:
    - مثال أكتب شيفرة الـ VHDL للدارة المنطقية التالية:

```
entity my_circuit is
        port (
                A_1,A_2,B_1,B_2,D_1 : in std_logic;
                E_out : out std_logic);
end my_circuit;
architecture my_circuit_arc of my_circuit is
        signal A_out, B_out, C_out : std_logic;
begin
        A_out <= A_1 and A_2;
        B_out <= B_1 or B_2;
        C_out <= (not D_1) and B_2;
        E_out <= A_out or B_out or C_out;
end my_circuit_arc;
```

يمكن كتابتها ايضاً بالشكل التالي بدون الحاجة إلى التصريح عن الاشارات:

```
E_out <= (A_1 and A_2)or (B_1 or B_2)
      or ((not D_1) and B_2);
```

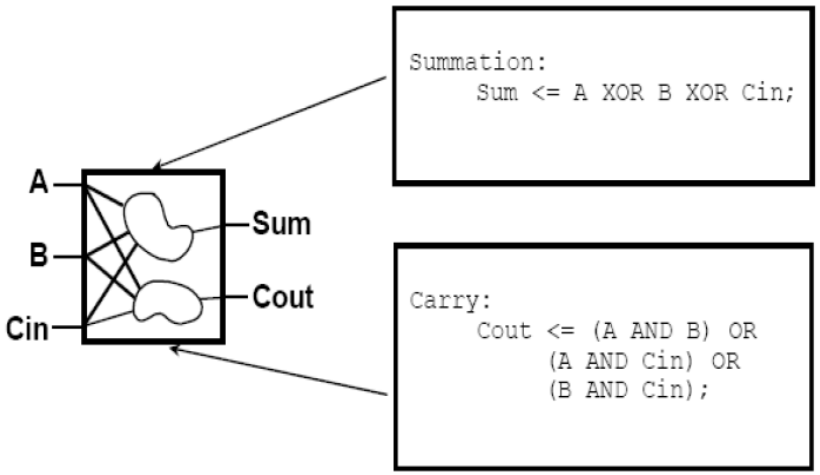
تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - مثال: أكتب شيفرة الـ VHDL لدارة جامع كامل بخانة واحدة
    - المرحلة الأولى:
      - توصيف النظام على أنه صندوق أسود يمتلك عدد ا من المداخل والمخارج بمواصفات محددة .
      - إن الدارة الرقمية للجامع الكامل عبارة عن دارة ذات ثلاثة مداخل ومخرجين:
        - دخل أول: A، دخل ثاني: B، دخل ثالث: الحمل . Cin
        - خرج أول : يمثل المجموع A + B = Sum، خرج ثاني : يمثل الحمل الناتج . Cout
        - جدول الحقيقة للجامع الكامل:

A	B	Cin	Sum	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

    - المرحلة الثانية :
      - توصيف النظام من الداخل من خال توصيف العلاقة التي تربط المداخل بالمخارج اعتماداً على العلاقات المنطقية المستنتجة من جدول الحقيقة

تصميم الأنظمة الرقمية باستخدام VHDL

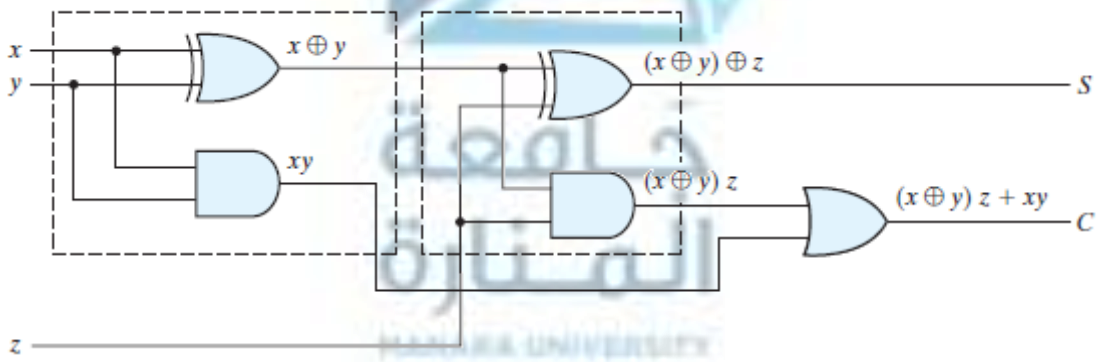


- اساسيات البرمجة بلغة الـ VHDL:
- طريقة كتابة شيفرة (كود) الـ VHDL:
  - مثال: أكتب شيفرة الـ VHDL لدارة جامع كامل بخانة واحدة
  - المرحلة الثانية :
  - الشيفرة المطلوبة:

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
ENTITY full_adder IS
        PORT ( A,B,cin: IN STD_LOGIC ;
                Sum, cout : OUT STD_LOGIC ) ;
END full_adder;
ARCHITECTURE dataflow OF full_adder IS
BEGIN
        Sum <= A XOR B XOR cin ;
        cout <= (A AND B) OR (cin AND A) OR (cin AND B) ;
END dataflow ;
```

تصميم الأنظمة الرقمية باستخدام VHDL

- اساسيات البرمجة بلغة الـ VHDL:
  - طريقة كتابة شيفرة (كود) الـ VHDL:
    - وظيفة: أكتب شيفرة الـ VHDL لدارة جامع كامل بخانة واحدة علماً أن الدارة المنطقية المكافئة له موضحة بالشكل التالي:



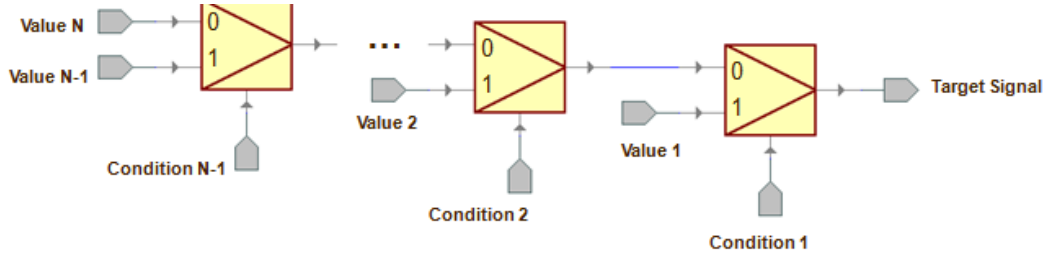
تصميم الأنظمة الرقمية باستخدام VHDL

- طريقة التنفيذ في اللغة VHDL:
  - على خلاف لغات البرمجة التقليدية يتم تنفيذ التعليمات في لغة VHDL على التوازي Concurrent (parallel)
    - يتم تنفيذ الكود البرمجي بنفس الوقت تقريبا لكافة التعليمات
    - يمكن القول ان إشارات الدخل تنقل إلى الخرج بزمن مهمل تقريباً باستثناء تأخير الانتشار
  - نظراً لهذ الخصوصية في تنفيذ التعليمات ونظراً للحاجة في بعض الحالات من تحويل التنفيذ إلى النمط التقليدي أي ترتيب تنفيذ التعليمات بحسب حاجة المبرمج وما يراه من ضرورة في عمل النظام
    - تم إيجاد مجموعة تعليمات كتعليمة Process .. End process التي يمكن إضافتها في وحدة التوصيف الداخلي ليتم تنفيذ التعليمات التي تذكر بعدها بترتيب ورودها من الأعلى إلى الأسفل
      - مما يؤدي إلى تحويل آلية التنفيذ إلى النمط التتابعي Sequential
  - من أنماط التعليمات التي يتم تنفيذ ا بصورة تسلسلية نذكرPROCESS, FUNCTION, PROCEDURE، حيث يتم تنفيذ ما بداخل هذه الكتل بشكل تسلسلي
    - لكن على مستو الكتلة نفسها فإنها تنفذ تفرعيا شأنها شأن أي تعليمة في VHDL،
      - وبالتالي تكون مهمتها تحقيق الترتيب أو التزامن في تنفيذ التعليمات
    - تسمى بـ GUARDED BLOCK، أو محطات الانتظار التي يكون من مهامها الأساسية منع الانتقال إلى كتلة برمجية أخر حتى يتحقق شرط ما يكون شرط التزامن مثلا

تصميم الأنظمة الرقمية باستخدام VHDL

- تعليمات البنية التفرعية في اللغة VHDL:
  - عمليات الإسناد (<=)
  - العمليات المنطقية.
  - العمليات الحسابية.
  - تعليمة when بأشكالها (when-else وwith-select-when)
  - تعليمة Generate
  - تعليمة when:
    - تأخذ عدة أشكال:
    - الشكل الأول When – Else:

- target_signal <= value1 when condition1 else
  value2 when condition2 else
  . . . valueN-1 when conditionN-1 else
  valueN;

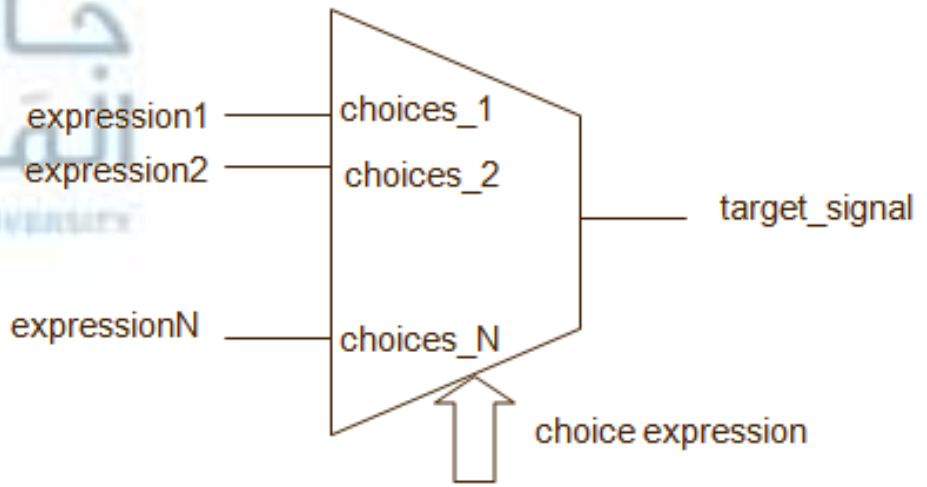


- .

تصميم الأنظمة الرقمية باستخدام VHDL

- تعليمات البنية التفرعية في اللغة VHDL:
  - تعليمة when:
    - تأخذ عدة أشكال:
      - الشكل الثاني With –Select-When:

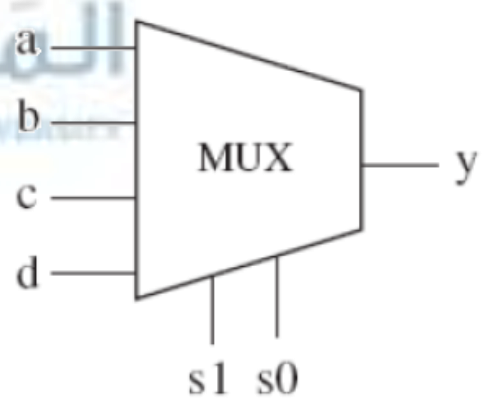
- with choice_expression select
  target_signal <= expression1 when choices_1,
  expression2 when choices_2,
  . . . expressionN when choices_N;



تصميم الأنظمة الرقمية باستخدام VHDL

- تعليمات البنية التفرعية في اللغة VHDL:
  - تعليمة when:
    - مثال: أكتب شيفرة الـ VHDL لدارة الناخب المبينة في الشكل:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY example IS
PORT ( a, b, c, d, s0, s1: IN STD_LOGIC;
y: OUT STD_LOGIC);
END example;
ARCHITECTURE Mux OF example IS
BEGIN
y <= (a AND NOT s1 AND NOT s0) OR
(b AND NOT s1 AND s0) OR
(c AND s1 AND NOT s0) OR
(d AND s1 AND s0);
END Mux;
```



input s	output x
0 0	a
0 1	b
1 0	c
1 1	d

تصميم الأنظمة الرقمية باستخدام VHDL

- تعليمات البنية التفرعية في اللغة VHDL:
  - تعليمة when:
    - مثال: أكتب شيفرة الـ VHDL لدارة الناخب المبينة في الشكل:

- الشكل الأول

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY mul IS
PORT ( a, b, c, d: IN STD_LOGIC;
s: IN STD_LOGIC_VECTOR (1 DOWNTO 0);
y: OUT STD_LOGIC);
END mul;
ARCHITECTURE mux OF mul IS
BEGIN
y <= a WHEN s="00" ELSE
b WHEN s="01" ELSE
c WHEN s="10" ELSE
d ;
END mux;
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY example IS
PORT ( a, b, c, d, s0, s1: IN STD_LOGIC;
y: OUT STD_LOGIC);
END example;
ARCHITECTURE Mux OF example IS
BEGIN
y <= (a AND NOT s1 AND NOT s0) OR
(b AND NOT s1 AND s0) OR
(c AND s1 AND NOT s0) OR
(d AND s1 AND s0);
END Mux;
```

تصميم الأنظمة الرقمية باستخدام VHDL

- تعليمات البنية التفرعية في اللغة VHDL:
  - تعليمة when:
    - مثال: أكتب شيفرة الـ VHDL لدارة الناخب المبينة في الشكل:
      - الشكل الثاني

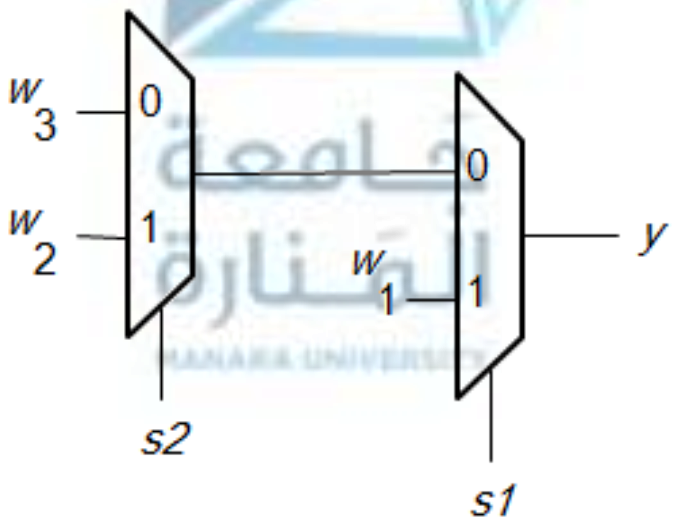
```
LIBRARY ieee ;
USE ieee.std_logic_1164.all;
ENTITY mul IS
PORT ( a, b, c, d: IN STD_LOGIC;
s: IN STD_LOGIC_VECTOR (1 DOWNTO 0);
y: OUT STD_LOGIC);
END mul;
ARCHITECTURE mux OF mul IS
BEGIN
WITH s SELECT
y <= a WHEN "00", -- notice "," instead of ";"
b WHEN "01",
c WHEN "10",
d WHEN OTHERS; -- cannot be "d WHEN "11" "
END mux;
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY example IS
PORT ( a, b, c, d, s0, s1: IN STD_LOGIC;
y: OUT STD_LOGIC);
END example;
ARCHITECTURE Mux OF example IS
BEGIN
y <= (a AND NOT s1 AND NOT s0) OR
(b AND NOT s1 AND s0) OR
(c AND s1 AND NOT s0) OR
(d AND s1 AND s0);
END Mux;
```

- .

تصميم الأنظمة الرقمية باستخدام VHDL

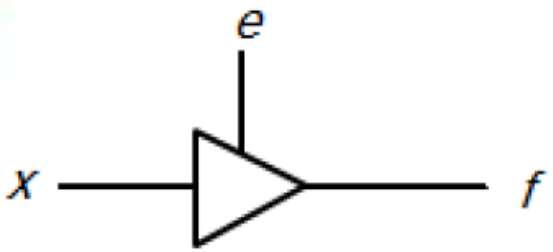
- تعليمات البنية التفرعية في اللغة VHDL:
  - تعليمة when:
    - وظيفة: أكتب شيفرة الـ VHDL لتوصيف النظام المبين بالشتل التالي:



تصميم الأنظمة الرقمية باستخدام VHDL

- تعليمات البنية التفرعية في اللغة VHDL:
  - تعليمة when:
    - مثال: أكتب شيفرة الـ VHDL لتوصيف دارة عازل ثلاثي الحالة بمدخل تمكين واحد:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY tri_state IS
PORT ( ena: IN STD_LOGIC;
   input: IN STD_LOGIC;
   output: OUT STD_LOGIC );
END tri_state;
ARCHITECTURE dataflow OF tri_state IS
BEGIN
   output <= input WHEN (ena = '1') ELSE 'Z';
END dataflow;
```



A tri-state buffer

e	x	f
0	0	Z
0	1	Z
1	0	0
1	1	1

تصميم الأنظمة الرقمية باستخدام VHDL

- تعليمات البنية التفرعية في اللغة VHDL:
  - تعليمة when:
    - مثال: المطلوب تصميم دارة عازل لثمانية خانات بمدخلي تمكين متعاكسيين:

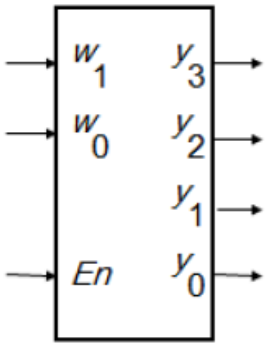
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY tri IS
PORT ( E: IN STD_LOGIC_VECTOR(1 downto 0);
   input: IN STD_LOGIC_VECTOR (7 DOWNTO 0);
   Output: OUT STD_LOGIC_VECTOR (7 DOWNTO 0));
END tri ;
ARCHITECTURE tri_state OF tri IS
BEGIN
   with E select
   output <= input WHEN "01" | "10",
   "ZZZZZZZZ" when others;
END tri_state;
```

تصميم الأنظمة الرقمية باستخدام VHDL

- تعليمات البنية التفرعية في اللغة VHDL:
  - تعليمة when:
    - مثال: أكتب شيفرة الـ VHDL لتوصيف دارة المشفر الموضحة بالشكل:

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
ENTITY dec2to4 IS
PORT (w: IN STD_LOGIC_VECTOR(1 DOWNTO 0) ;
    En: IN STD_LOGIC ;
    y: OUT STD_LOGIC_VECTOR(3 DOWNTO 0) ) ;
END dec2to4 ;
ARCHITECTURE dataflow OF dec2to4 IS
    SIGNAL Enw : STD_LOGIC_VECTOR(2 DOWNTO 0) ;
BEGIN
    Enw <= En & w ;
    WITH Enw SELECT
    y <= “0001" WHEN "100",
    "0010" WHEN "101",
    "0100" WHEN "110",
    “1000" WHEN "111",
    "0000" WHEN OTHERS ;
END dataflow ;
```

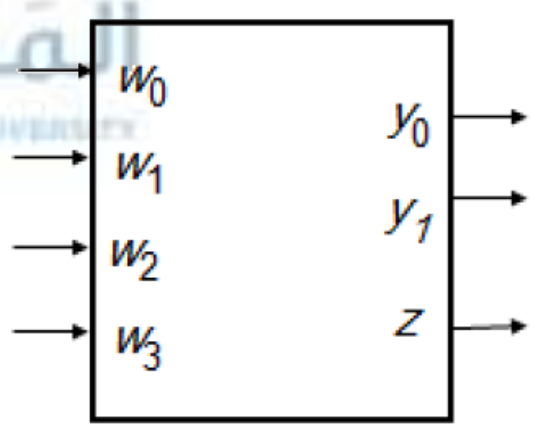
En	w_1	w_0	y_3	y_2	y_1	y_0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0
0	x	x	0	0	0	0



تصميم الأنظمة الرقمية باستخدام VHDL

- تعليمات البنية التفرعية في اللغة VHDL:
  - تعليمة when:
    - مثال: أكتب شيفرة الـ VHDL لتوصيف دارة فاك تشفير الأولوية الموضحة بالشكل:

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
ENTITY priority IS
PORT (w : IN STD_LOGIC_VECTOR(3 DOWNTO 0) ;
   y : OUT STD_LOGIC_VECTOR(1 DOWNTO 0) ;
   z : OUT STD_LOGIC ) ;
END priority ;
ARCHITECTURE dataflow OF priority IS
BEGIN
   y <= "11" WHEN w(3) = '1' ELSE
   "10" WHEN w(2) = '1' ELSE
   "01" WHEN w(1) = '1' ELSE "00" ;
   z <= '0' WHEN w = "0000" ELSE '1' ;
END dataflow
```



w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1