فكرة عامة عن
الشرائح الالكترونية القابلة للبرمجة
(Programmable Logic Devices)

تأليف: جميل الخطيب
ترجمة: طارق الخولي و جميل الخطيب

1- مقدمة

أ- الغرض من هذا المقال هو تقديم فكرة عامة عن الشرائح الالكترونية القابلة للبرمجة والتعرف على تركيبها الداخلي وكيفية برمجتها حتى نتعرف على كيفية استخدامها في تصميم الدوائر الالكترونية

ب- من المستفيد من قراءة هذا المقال؟
هذا المقال مفيد لمهندسي الالكترونيات الجدد ولهواة تصميم الدوائر الالكترونية الذين يرغبون في تصميم وتنفيذ دوائرهم الخاصة بطريقة سهلة ومتطورة

جـ - ماهي المعلومات الأساسية التي يجب معرفتها قبل قراءة هذا المقال؟
يجب معرفة أساسيات الدوائر الالكترونية وأساسيات تصميم دوائر الديجيتال قبل قراءة هذا المقال

2- معلومات هامة عن تصميم الدوائر الالكترونية وتصميم دوائر الديجيتال

أ- اختراع الترانزستور(تأثيره)
أدى اختراع الترانزستور الى احداث أثر كبير في العالم بأسره بصفة عامة وفي صناعة الالكترونيات بصفة خاصة
وسبب حدوث هذا التاثير هو :-
1) حجم الترانزستور الصغير
2) تصميمه البسيط

ب- استخدامات الترانزستور
يستخدم الترانزستور في العديد من الدوائر الالكترونية بدءا من دوائر التكبير (Amplifiers) وحتى دوائر الأجهزة الموسيقية المعقدة ويقوم بتنفيذ وظائف مختلفة في هذه الدوائر
وأحد وظائف الترانزستور هو عمله في الدائرة الالكترونية كمفتاح(Switch) يعمل على فصل وتوصيل التيار الكهربي
وعمل الترانزستور كمفتاح (Switch) ساعد في تنفيذ دوائر الكترونية مثل دوائر التنبيه (Alarms) ودوائر التحكم (Control) ودوائر أخرى كثيرة تعتمد على فكرة فصل وتوصل التيار الكهربي (ON – OFF Technique)

جـ- تصميم دوائر الديجيتال يعتمد بشكل اساسي على فكرة الفصل والتوصيل (ON - OFF) ونرمز للفصل بالرقم صفر (0) ونرمز للتوصيل بالرقم واحد (1) وهذان الرمزان (الصفر والواحد) هما الأساس في عمليات ال (Boolean Arithmetic)
كما أن تصميم دوائر الديجيتال يعتمد على كل من :-

1) عمليات (Boolean Arithmetic)
والعمليات الرئيسية في الـ (Boolean Arithmetic) هي (And,Or,Not) حيث يمكن تكوين باقي العمليات من هذه العمليات الرئيسية

مثال (2 Bit Half Adder)

Sum = (x AND NOT y) OR (NOT x AND y)
Carry =(x AND y)

2) دوائر التخزين
كما يعتمد تصميم دوائر الديجيتال على دوائر الفليب فلوب (Flip Flop) والـ (Register) وهي دوائر تسمح بتنفيذ عمليات تخزين البيانات

هـ) أنواع دوائر الديجيتال (من حيث التزامن) :-
1) دوائر متزامنة (Synchronous)
وهي دوائر تعتمد في عملها على وجود اشارة تزامن (Clocked)
2) دوائر غير متزامنة (Asynchronous)
وهي دوائر لاتعتمد في عملها على وجود اشارة تزامن (Non-Clocked)

و) مزايا تصميم دوائر الديجيتال
1) السهولة والبساطة في التنفيذ
حيث يمكن بسهولة تصميم وتنفيذ دائرة ديجيتال معقدة عن ان تقوم بتصميم وتنفيذ دائرة أنالوج (Analog)
2) السهولة والبساطة في اكتشاف الاعطال واصلاحها
نظرا لان دوائر الديجيتال تعتمد على فكرة الفصل والتوصيل فانه من السهل اكتشاف الأعطال واصلاحها وكل خرج فيها ينبغي ان يكون اما صفر او واحد

3- تطور الدوائر المتكاملة (Integrated Circuits) والشرائح الالكترونية (Chips)

أ) الترانزستورات هي المكونات الرئيسية في دوائر الديجيتال وتستخدم بشكل مبسط في في الدوائر البسيطة كمفاتيح فصل وتوصيل للتيار الكهربي

ب) دوائر الديجيتال البدائية وتكنولوجيا الـ ((LSI(Large Scale Integration)
دوائر الديجيتال البدائية كانت قديما تستخدم دوائر أساسية بسيطة مثل دوائر (AND, OR, NOT)
تم تجميع هذه الدوائر في شريحة صغيرة تسمى الدائرة المتكاملة (Integrated Circuit)
واختصارا تسمى (IC)
ثم ظهر فيما بعد تكنولوجيا (LSI) والتي تقوم بتجميع دوائر معقدة داخل شريحة الكترونية صغيرة ومن أمثلتها (Decoder) و (Adder) و (Multiplier)

جـ) تكنولوجيا (VLSI) (Very Large Scale Integration)
وهي شريحة الكترونية اكثر تعقيدا من شريحة (LSI) تقوم بتجميع العديد من المكونات والدوائر في شريحة واحدة ومن أمثلتها (Processor) و (CPU) و (Communicaton Protocol)

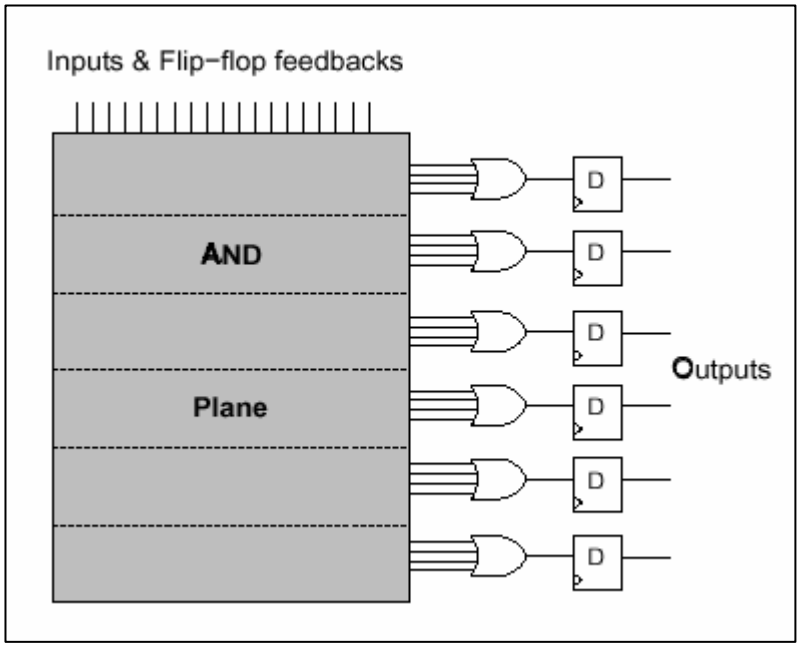
د) ظهرت تكنولوجيا (Programmable Logic Devices) (PLD)

وهي شريحة الكترونية يتم فيها تجميع العديد من الدوائر المتكاملة ويتم توصيل هذه الدوائر بواسطة فيوزات والوظيفة التي تقوم بها الشريحة النهائة تعتمد على عملية البرمجة التي تتم لهذه الشريحة
وعملية البرمجة هذه عبارة عن حرق بعض الفيوزات التي تربط الدوائر الداخلية الموجودة في الشريحة
عملية الحرق هذه تؤدي الى فصل بعض الفيوزات بين الدوائر وترك الفيوزات الاخرى متصلة ومن أمثلة شرائح (PLD) شريحة (ROM) وشريحة (PAL) وشريحة (PLAs)
والفرق الرئيسي بين هذه الشرائح هو مكان الفيوزات التي يمكن حرقها والتوصيلات الثابتة التي لايمكن حرقها

ملحوظة

نظرا لان هذه الدوائر الداخلية في شريحة (PLD) متصلة بواسطة فيوزات يتم حرقها عند البرمجة فانه لايمكن برمجة شريحة (PLD) الا مرة واحدة فقط

الشكل(1) الموضح بالاسفل يبين التركيب الداخلي لشريحة (PAL)
ومن الواضح في الشكل انها تتكون من دوائر (OR, AND) مرتبطة سويا بواسطة مجموعة فيوزات



الشكل (1) : تركيب شريحة (PAL)

هـ) الشرائح الالكترونية المصنعة لغرض محدد (ASIC)
وهي شرائح الكترونية يتم بناؤها لتنفيذ غرض محدد وهناك نوعان منها:-
1)النوع الأول شرائح تحتوي بداخلها على العديد من الدوائر الالكترونية

2)النوع الثاني شرائح تحتوي بداخلها على دائرة واحدة

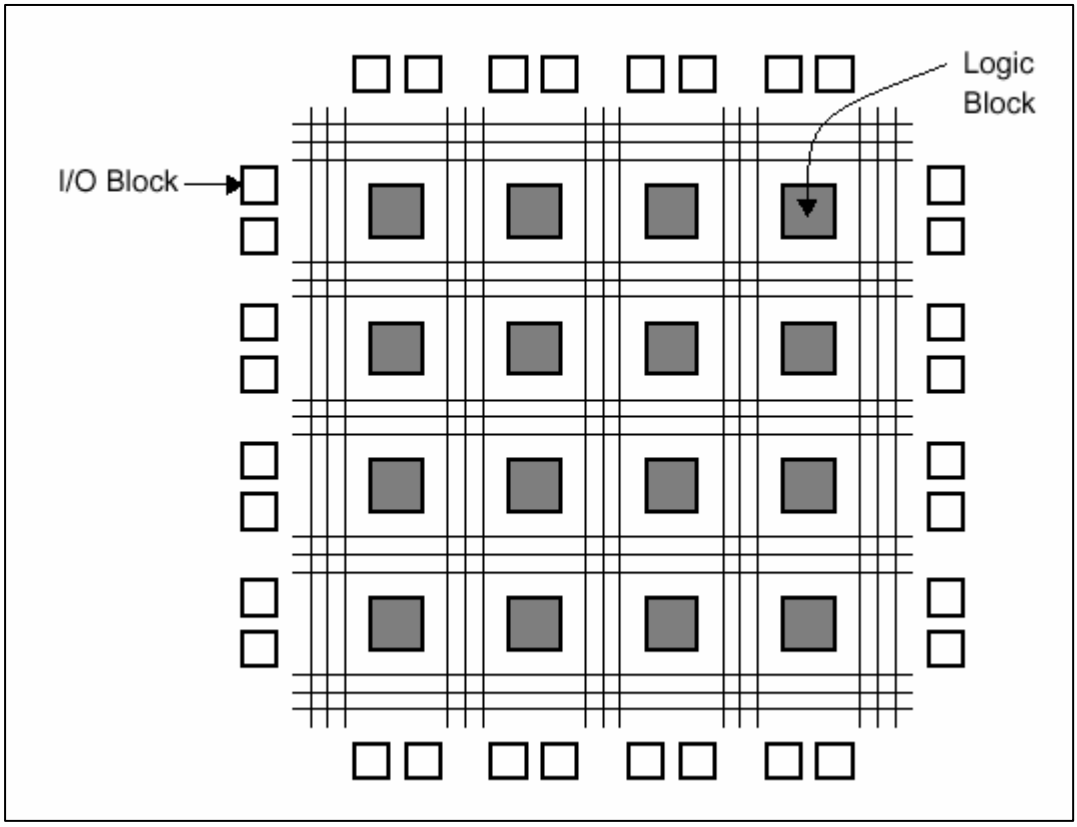
وهذه الشرائح يتم ضبطها لتنفيذ غرض محدد أثناء عملية التصنيع

و) الشرائح الالكترونية المعقدة القابلة للبرمجة(CPLD)

هي شرائح الكترونية أكثر تعقيدا من شرائح(PLD) وتحتوي بداخلها على بلوكات من شرائح (PLD) وهي تشبه الى حد كبير في تعقيدها شرائح (PAL) التي سبق شرحها ولكن شرائح (CPLD) عدد الدوائر بها أكثر

ز) الشرائح الالكترونية Field Programmable Gate Arrays (FPGA)

هي شرائح الكترونية تركيبها العام يسمح باستيعاب عدد كبير جدا من الشرائح الالكترونية وفي حين ان شرائح(CPLD) بها امكانية تواجد عدد كبير من الشرائح الالكترونية بداخلها فان شرائح (FPGA) تسمح بعدد محدود من الدوائر بداخلها ولكنها يتوافر بها عدد اكبر من دوائر الفليب فلوب (Flip-Flop) نسبة الى عدد الدوائر المسموح بها في شرائح (CPLD)



الشكل (2) : تركيب شريحة (FPGA)

4- التركيب االداخلي لشرائح (FPGA) راجع الشكل (2)

أ) تتكون شرائح(FPGA) داخليا من بلوكات من الدوائر الالكترونيةوكل بلوك يتكون من دوائر صغيرة موزعة على هيئة مجموعة من الخلايا(LOGIC CELLS) وتتكون كل خلية عادة من دائرة فليب فلوب(Flip-Flop) وبعض الدوائر الاخرى التي تختلف حسب كل من الشركة المصنعة(Vendor) والـ(Family) التي تنتمي لها شريحة الـ (FPGA)

ملحوظة
كل شركة مصنعة لشرائحFPGA تستخدم اسم خاص لوصف الخلايا وكيفية بنائها ومن أمثلة هذه الأسماء(Logic Block) و(Logic Element)

ب) كما يتواجد داخل الخلايا(Logic Cells) دائرة(LUT (Look up Tables وهي تشبه الـ(ROMs) بعض أنواع شرائحFPGA يتواجد بها دوائر ذاكرة أخرى مثل SRAMو(Dual Port Memory) و(CAM)وهي دوائر يتم استخدامها بشكل خاص إما في لغة(HDL) او باستخدام دوائر خاصة في عملية رسم الدوائر بالبرامج الخاصة schematic entry

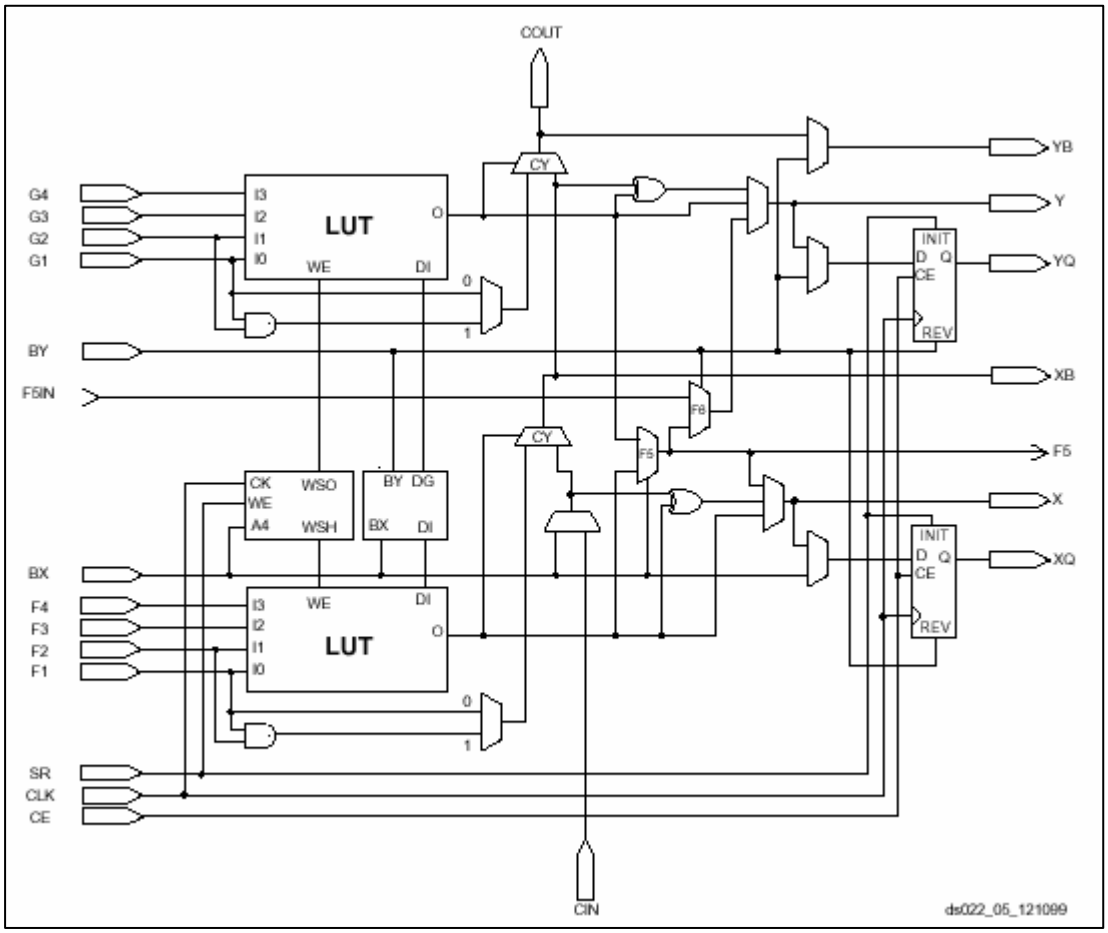
جـ) وسائل الربط والتوصيل(Routing Resources) بين دوائر شريحة(FPGA) وسائل الربط والتوصيل في شريحة(FPGA) هي قنوات توصيل (Routing Channels) وأسلاك ومفاتيح تربط بين الدوائر الداخلية مثل دوائر Memory, LUT & Logic Cells

د) pinsأطراف التوصيل الخارجية
وبهذه الوسائل يمكن الربط بين الدوائر الداخلية في الشريحة أطراف التوصيل الخارجية (PINs) لشرائح (FPGA)
تختلف في كيفية توزيعها وترتيبها فبعض الشرائح يتم توزيع ترتيبها بطريقة (TTL)أو(CMOS) أو(PCI) أو (AGP) او اي طريقة اخرى. لذلك فان شرائح (FPGA)يمكن ان تستخدم للربط بين تكنولوجيات مختلفة من تكنولوجيات الدوائر

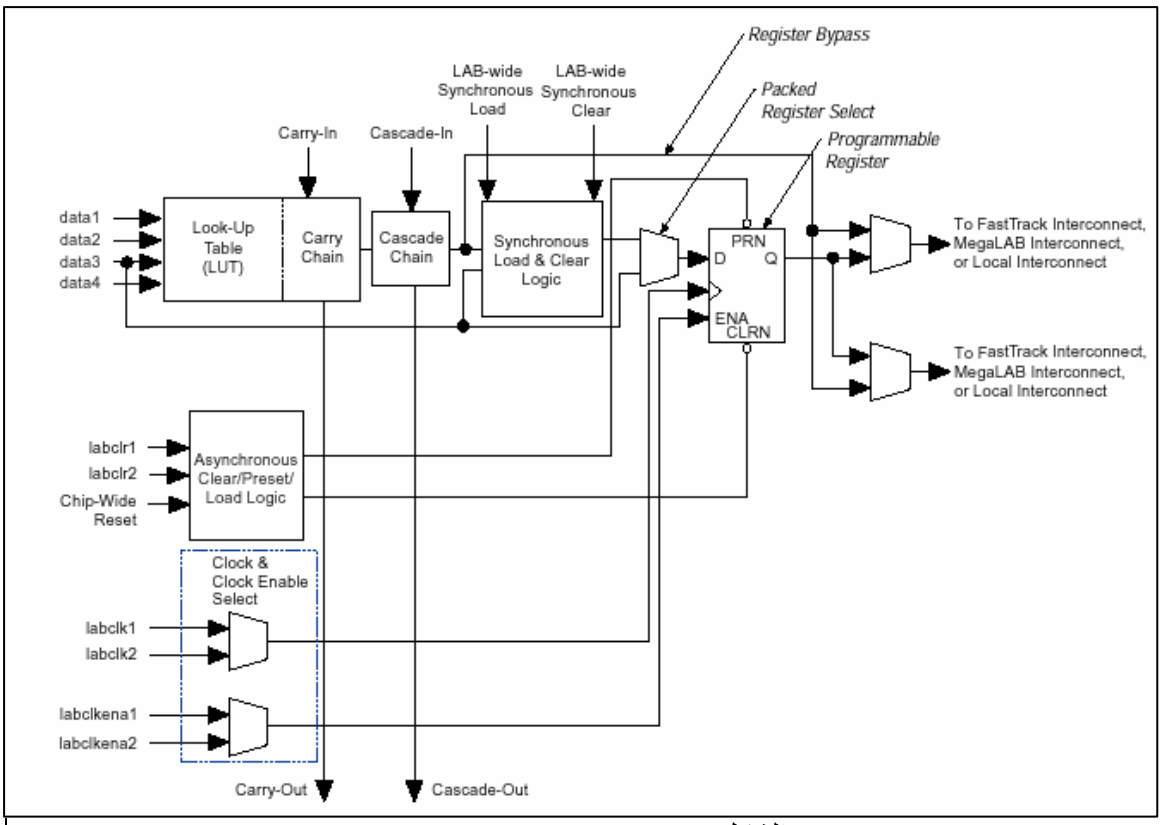
هـ) pins أطراف التوصيل الخارجية (Clock) و (PLL)
بعض شرائح (FPGA) تخصص أطراف توصيل ذات سرعة عالية للـ (Clock) و (Reset) وبعض شرائح (FPGA) يمكن ان يكون لها بلوكات للتحكم في ال (Clock) و (PLL) مثل (ClockMultiplier) و (Divider)

5- أمثلة عن شرائح (FPGA)(من شركات مختلفة)

أ. (Xilinx Spartan and Virtex Families)
ب. (Altera Apex and Acex Families)
ج. (Actel)
د. (Lucent Orca Devices)



الشكل(3) Xilinx's Virtex Slice



الشكل (4) Altera's Apex Logic Element

6- اللوحات الالكترونية المخصصة لشرائح (FPGA)

هناك العديد من اللوحات الالكترونية المخصصة لشرائح (FPGA) ومنها

أ. لوحات عرض (Demo) للشرائح

ب. ولوحات اخرى لتطبيقات معينة (Applications)

هذه اللوحات تتراوح من :-

أ. لوحات مخصصة لشرائح (FPGA) الصغيرة يمكن ان تكون مزودة بـ (External Headers) ومجموعة توصيلات

ب. وحتى لوحات كبيرة ومعقدة مزودة بشرائح ربط أخرى (Interface Chips) وكذلك بشرائح (FPGA)

وفيما يلي بعض مصادر اللوحات الالكترونية المخصصة لشرائح(FPGA)

أ. Xess boards (some of them are suitable for beginners)

ب. Altera SOPC boards

ج. http://www.burched.com.au/B5Spartan2.html

7- أساليب تصميم برنامج شريحة (FPGA)

تصميم برنامج الشرائح الالكترونية واحد لكل المهندسين المتخصصين والطلبة والهواة والفرق الوحيد هو مدى تعقيد التصاميم والادوات التي يستخدمها المصمم والتي يمكن ان تتوفر بها امكانيات افضل لمعالجة التصاميم وتحليلها

وتمر عملية التصميم بمراحل هي:-

أ - تحديد الغرض من الشريحة

تحديد المشكلة المراد حلها والغرض من الشريحة ومانحتاج لتنفيذه والتي تعد خطوة مهمة قبل البدأ بالتصميم

ب - وضع التصميم

هذه الخطوة هي مانفعله عادة بواسطة الورقة والقلم ولكن يتم تنفيذها هنا بطريقة منظمة تعتمد على الكمبيوتر وهي اكثر الخطوات اهمية منذ بدء التصميم وهناك طريقتان لوضع التصميم هما:-

1) طريقة التصميم الأولى (رسم الهيكل البنائي للدائرة الكهرية) (schematic Entry)

ويعتبر ذلك مماثلا لرسم الهيكل البنائي لاي دائرة الكترونية وهي خطوة يتم فيها رسم المكونات الالكترونية على لوحة من الورق أومباشرة على صفحة (Sheet) ببرنامج الحاسوب ويتم توصيلها سويا. وهذه الطريقة غير مجدية للتصميمات كبيرة الحجم التي تحتوي على العديد من المكونات والدوائر

2) طريقة التصميم الثانية (كتابة برنامج باستخدام احدى لغات HDL)

هو الأسلوب الذي يصف تركيب الدوائر الالكترونية باستخدام برنامج كمبيوترواحدى لغات (HDL) مثل VHDL & Verilog هي لغات كمبيوتر مثل لغة الـ (C) والـ (C++) والـ (Pascal) والـ (Basic) ولمزيد من المعلومات عن هذه اللغة يمكنك مراجعة الجزء الخاص بشرح هذه اللغة باخر الملخص

جـ - المحاكاة (Simulation)

بالرغم من أن شرائح (FPGA) يمكن برمجتها ومسحها بسهولة في حالة حدوث خطأ ولكن في حالة التصميمات الكبيرة فانه من الافضل ان يتم اكتشاف الاعطال واصلاحها قبل ان تتم عملية البرمجة وبهذه الطريقة يمكن توفير الوقت اللازم لبرمجة شرائح (FPGA) لذلك نلجأ لعملية المحاكاة لاكتشاف الأعطال ومن ثم اصلاحها

د - تحليل التصميم (Synthesis)

هي عملية استنباط مكونات الدائرة التي تم تصميمها بواسطة لغة (HDL) لتحويل الوصف الى دوائر الكترونية وهذه الخطوة لاتستخدم في حالة استخدام اسلوب رسم الهيكل البنائي للدائرة schematic Entry

هـ- وضع المكونات في أماكنها والربط بينها(Place and route)

هذه الخطوة تستخدم لمقابلة الدوائر المصممة بالدوائر والموارد المتاحة بالFPGA ووضع الدوائر بالاماكن المناسبة بالشريحة وبعد وضع المكونات في أماكانها يتم ربطهم سويا طبقا لتصميم الدائرة باستخدام قنوات التوصيل والأسلاك الداخلية. هذه الخطوة تربط كذلك بين أطراف التوصيل الخارجية للشريحة (pins) مع باقي أجزاء الدائرة الداخلية التي سيتم توصيل الشريحة بها

و- توليد الـ(Bit Stream)

عملية توليد الـ (Bit Stream) أو ملف البرمجة (Programming File) الذي يحتوي على كل المعلومات عن تصميم الدائرة وكيف يتم مقابلة التصميم بالمواردالموجودة بال(FPGA) وكيف ينبغي ان تتصل المفاتيح الداخلية للـ(FPGA). وهو الملف الذي يستعمل لبرمجة الشريحة.

وبرمجة ملف الـ (Bit Stream) يعتمد على الشريحة والشركة المصنعة لها وكل شريحة لها طريقة برمجة محددة ويتم تزويدها ببرنامج خاص لبرمجتها

ملحوظة الخطوات الثلاث الخيرة تنفذ عادة بواسطة برامج مخصصة لشرائح (FPGA) من قبل الشركة المصنعة للشريحة

– طقنية (Hardware Description Languages) (HDL) اللغات الوصفية للالكترونيات

كما تم الشرح مسبقا فان (HDL) هي طريقة من طرق وضع تصميم شرائح (FPGA) وفي الحقيقة فانها يمكن ان تكون طريقة لتصميم الهاردوير (Hardware Design) وهي ببساطة طريقة لوصف تصميم الهاردوير بطريقة لغة شبيهة باللغة التي يتحدث بها البشر (Human -Like) والتي تشبه لحد كبير لغات برمجة الحاسوب

وهناك العديد من لغات (HDL) بعضها بسيط وبعضها معقد ومعظم هذه اللغات يمكن ان تصف عمليات ديجيتال أساسية مثل (AND,OR,NOT) واللغات الأخرى الأكثر تعقيدا والأكثر تقدما للمستخدم ان يصف التصميم بطريقة يمكن ان يتم قراءتها بواسطة البشر علي سبيل المثال فانها تسمح للمستخدم ان يستخدم أوامر (Loops,Case,If) والتي تجعله يركز على التصميم نفسه (كبرنامج) وليس على الهاردوير و معظم لغات (HDL) تعرف أطراف الدخل والخرج للشريحة ووظائفها الداخلية ومن أمثلتها(Verilog) و (VHDL) و (AHDL)

ويمكنك مراجعة الأمثلة الخاصة بلغة (VHDL) في القسم الخاص بذلك

9 – الأدوات(البرامج المحوسبة) (Tools)

Simulation: Modelsim, Active-HDL, NC-SIM
Synthesis: Leonardo, Synplify, Synopsys
Place and Route: Altera Quartus, Xilinx Foundation.

معظم الشركات المنتجة لشرائح (FPGA) تنتج برامج بسيطة للمحاكاة والتركيب والربط وهي تفيد في التصميمات الصغيرة والمتوسطة الحجم هذه الأدوات مناسبة لهواة تصميم الدوائر الالكترونية وللطلبة وحتى للشركات التي تحتاج الى تنفيذ دوائر بسيطة وعادة ما تكون هذه البرامج مجانية من مواقع الشركات على الانترنت.

10 - أمثلة التصميم للـ (VHDL)

- **Half adder**

```
LIBRARY ieee;
```

```
USE ieee.std_logic_1164.ALL;

ENTITY Adder_ent IS

  PORT (
    Op1    : IN  std_logic;            -- Operand 1
    op2    : IN  std_logic;            -- Operand 2
    carry  : OUT std_logic;              -- Output carry
    Result : OUT std_logic);            -- Result

END Adder_ent;

ARCHITECTURE behavior OF Adder_ent IS

BEGIN  -- behavior

  Result <= (Op1 AND NOT Op2) OR (NOT Op1 AND Op2);
  Carry  <= Op1 AND Op2;

END behavior;
```

- **8 bit Adder**

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.ALL;

ENTITY Adder_ent IS

  PORT (
    clk    : IN  std_logic;                    -- System clock
    rst_n  : IN  std_logic;                     -- System reset
    Op1    : IN  std_logic_vector(7 DOWNTO 0);   -- Operand 1
    op2    : IN  std_logic_vector(7 DOWNTO 0);   -- Operand 2
    Result : OUT std_logic_vector(7 DOWNTO 0));  -- Result

END Adder_ent;

ARCHITECTURE behavior OF Adder_ent IS

BEGIN  -- behavior

  PROCESS (clk, rst_n)
  BEGIN  -- PROCESS
    IF rst_n = '0' THEN                 -- asynchronous reset (active low)
      Result <= (OTHERS => '0');
    ELSIF clk'event AND clk = '1' THEN  -- rising clock edge
      Result <= Op1 + op2;
    END IF;
  END PROCESS;

END behavior;
```

- **Counter**

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.ALL;
```

```
ENTITY Adder_ent IS

  PORT (
    clk   : IN  std_logic;                    -- System clock
    rst_n : IN  std_logic;                     -- System reset
    Count : OUT std_logic_vector(7 DOWNTO 0));  -- Count

END Adder_ent;

ARCHITECTURE behavior OF Adder_ent IS
  SIGNAL counter : std_logic_vector(7 DOWNTO 0);  -- internal counter
BEGIN  -- behavior

  PROCESS (clk, rst_n)
  BEGIN  -- PROCESS
    IF rst_n = '0' THEN                 -- asynchronous reset (active low)
      Counter <= (OTHERS => '0');
    ELSIF clk'event AND clk = '1' THEN  -- rising clock edge
      Counter <= counter + 1;
    END IF;
  END PROCESS;

  count <= counter;

END behavior;
```

- 7-Segment decoder

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY Decoder IS
  PORT (
    InBin   : IN  std_logic_vector (3 DOWNTO 0);
    Display : OUT std_logic_vector (6 DOWNTO 0));
END Decoder;

ARCHITECTURE rtl OF decoder IS
  SIGNAL t : std_logic_vector (6 DOWNTO 0);
BEGIN

  seg_process : PROCESS (InBin)
  BEGIN

    CASE InBin IS
      WHEN "0000" => t <= "1111110";
      WHEN "0001" => t <= "0110000";
      WHEN "0010" => t <= "1101101";
      WHEN "0011" => t <= "1111001";
      WHEN "0100" => t <= "0110011";
      WHEN "0101" => t <= "1011011";
      WHEN "0110" => t <= "0011111";
      WHEN "0111" => t <= "1110000";
      WHEN "1000" => t <= "1111111";
      WHEN "1001" => t <= "1110011";
      WHEN "1010" => t <= "1110111";
      WHEN "1011" => t <= "0011111";
      WHEN "1100" => t <= "1001110";
      WHEN "1101" => t <= "0111101";
```

```
        WHEN "1110" => t <= "1001111";
        WHEN OTHERS => t <= "1000111";
      END CASE;
      Display       <= NOT t;
    END PROCESS seg_process;
  END rtl;
```

11- لماذا نستخدم الشرائح الالكترونية القابلة للبرمجة؟

أ. ليس هناك حاجة لعملية تصنيع معقدة لتنفيذ الدوائر الالكترونية باتخدام شرائح (FPGA) اذا كانت الدائة الالكترونية موجودة أصلا ضمن مكونات شريحة (FPGA) ملف (Bit Stream) يتم الاحتياج اليه عند برمجة شريحة (FPGA) وتغير الوظيفة الخاصة بها

ب. وانه من الاسهل تصميم واكتشاف العيوب وتعديلها في شرائح (FPGA) بالنسبة لشرائح (ASIC)

12 - بناء وحدة معالجة مركزية لغرض خاص

نظرا لان شرائح (FPGA) يسهل برمجتها وتصميمها فان الشركات يمكنها بناء وتنفيذ وحدات معالجة مركزية خاصة ولهذا السبب فان هناك العديد من وحدات المعالجة المركزية المجانية على صفحات الانترنت مثل www.opencores.org

13 - الدوائر الالكترونية القابلة لتغيير وظائفها أثناء تشغيلها

أحد مزايا الشرائح الالكترونية القابلة للبرمجة انها يمكن اعادة تغيير وظيفتها أثناء عملها، وهذا يعني أن وظيفة الشريحة الالكترونية (FPGA) يمكن ان يتم تغييرها خلال تشغيلها ، ميزة هذه الخاصية هي أنه يمكن لشخص ما وضع العديد من التصميمات داخل شريحة (FPGA) وهذه التصميمات لاتعمل في شريحة (FPGA) في نفس الوقت ولكن يتم تشغيل بعضها فقط عندما يتم الحاجة اليهم وفي هذه الحالة يمكننا وضع العديد من التصميمات في شريحة (FPGA) واحدة أكثر بكثير من الحجم الحقيقي لها

على سبيل المثال هناك بعض التصميمات تحتاج ان يتم تنفيذها فقط عند بداية تشغيل النظام الالكتروني ثم لايتم استخدامها بعد ذلك اما باقي التصميمات فيمكنها أن تعمل فقط عندما يحتاج المستخدم منها أن تعمل لذلك اذا قمنا بوضع كل هذه التصميمات معا على شريحة (FPGA) فاننا سنحتاج شريحة (FPGA) كبيرة جدا ولكن بفضل خاصية امكانية تغيير وظيفة الشريحة اثناء تشغيلها فاننا يمكننا وضع العديد من التصميمات على شريحة (FPGA) واحدة ثم نقوم بتشغيل التصميم الذي نرغب أن تعمل الشريحة طبقا له موضع التنفيذ ثم نقوم بفصل هذا التصميم وتشغيل تصميم آخرطبقا لاحتياجاتنا وبذلك نقلل الحجم الذي تحتاجه شريحة (FPGA)

14 - تصميمات الشرائح الالكترونية المعروضة بشكل مجاني

كما هو متبع في مجال تصميمات برامج الكمبيوتر المعروضة بشكل مجاني فان تصميمات الهاردوير لشرائح (FPGA) يتم عرضها بشكل مجاني لاستخدامها في برمجة شرائح (FPGA) ولغات (HDL) ، المصممون والهواة يمكنهم كتابة كود (HDL) ثم نبرمج هذا الكود على شرائح (FPGA) ثم يتم تبادل هذا الكود بين المصممين عبر العالم ويمكن للمصممين مراجعة وتعديل هذا الكود

باختصار فان كود (HDL) وشرائح (FPGA) يشبهون كود تصميمات برامج الكمبيوتر المعروضة مجانا مثل ماهو موجود للنظام (Linux) وفي يوم ما سيكون لدينا تصميمات معروضة مجانا لوحدات المعالجة المركزية وحاواسيب

15- المراجع

- Digital design by Morris Mano.
- Field Programmable Gates Array by Oldfield and Dorf.
- http://klabs.org/richcontent/Tutorial/fpga/Toronto_tutorial.pdf
  Architecture of FPGAs and CPLDs: A Tutorial (The best document)
- http://klabs.org/richcontent/Tutorial/PLD_Definitions.htm
- http://hobbes.inesc.pt/~hcn/cid/xprojecto2.pdf
- www.optimagic.com
- www.xilinx.com
- www.altera.com
- www.xess.com
- http://www.eecg.toronto.edu/~lewis/ece451/vhdl_examples.html
- http://www.mcmanis.com/chuck/robotics/fpga/
-

برجاء ارسال أي تعليقات على هذا المقال او أي اضافات الى

jamilkhatib75@yahoo.com

أو

Tarek_ah@yahoo.com