أتمتة النظم المنطقية الإلكترونية الثلاثية: دراسة مستحدثة مؤسسة على لغة VHDL
المرحلة الثالثة: القطع الثلاثية غير التجميعية (المتتابعة)

عبدالله علي قاسم قحطان الحميدي[1,*]
عبدالرقيب عبده أسعد العريقي[2,*]

[1]مدرس مساعد في هندسة الحاسوب، قسم الهندسة الإلكترونية، جامعة العلوم والتكنولوجيا، صنعاء، اليمن
[2]أستاذ الهندسة الإلكترونية والحاسوب، قسم الهندسة الإلكترونية، جامعة العلوم والتكنولوجيا، صنعاء، اليمن
*عناوين المراسلة: a.qahtan2@ust.edu , abdarraqib62@yahoo.com

https://doi.org/10.20428/JST.24.2.3

أتمتة النظم المنطقية الإلكترونية الثلاثية: دراسة مستحدثة مؤسسة على لغة VHDL المرحلة الثالثة: القطع الثلاثية غير التجميعية (المتتابعة)

الملخص:

سيتم في هذه الورقة العلمية بناء المرحلة الثالثة من المكتبة البرمجية المؤسسة على لغة الـVHDL للقطع الثلاثية غير التجميعية (المتتابعة)، أي القطع القادرة على حفظ المعلومة، بدءاً بالحابسة الثلاثية TDL (Ternary D Latch) وختاماً بالحافظة (Ternary RAM) RAM.

الكلمات المفتاحية : المنطق الثلاثي، عناصر حفظ المعلومات في المنطق الثلاثي، عدادات المنطق الثلاثي، الحافظة الثلاثية، لغة الـVHDL.

Ternary Electronic Logic Systems Automation: A Novel Study Based on VHDL Language Third Part: Ternary Non-Combinational (Sequential) Logic Components

Abstract:

In this scientific paper, the third part of the software library for the Ternary non-combinational (Sequential) logic components (the components that store information) will be built based on VHDL language starting by the Ternary D Latch (TDL) and ending by the Ternary RAM (TRAM).

Keywords: Ternary logic, Ternary memory elements, Ternary counters, Ternary memory, VHDL language.

https://doi.org/10.20428/JST.24.2.3

1. المقدمة:

خصصت المرحلتان الأولى والثانية من الدراسة لقطع المنطق الثلاثي التجميعية، أي التي لا تحتفظ بالمعلومة، بدءاً بالبوابة العاكسة TNOT بأنواعها الثلاثة وختاماً بالجامع المتوازي TPA (Ternary Parallel Adder).

هذه الورقة العلمية لنتائج المرحلة الثالثة من الدراسة لقطع المنطق الثلاثي القادرة على حفظ المعلومة على النحو التالي:

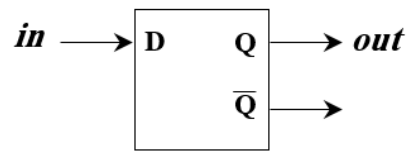
1) **الحابسة الثلاثية TDL (Ternary D Latch).**
2) **النطاطة الثلاثية TDF (Ternary D Flip-Flop).**
3) **السجلات TRE (Ternary Register).**
4) **العدادات TCN (Ternary Counter).**
5) **الحافظة TRAM (Ternary RAM).**

نظمت هذه الورقة العلمية على النحو التالي: البند (2) للـ TDL، البند (3) للـ TDF، البند (4) للـ TRE، البند (5) للـ TCN، البند (6) للـ TRAM، البند (7) لتنفيذ القطع السابقة بلغة الـ VHDL، البند (8) للخاتمة، ثم المراجع ثم الملحق المتضمن تنفيذ القطع سالفة الذكر بلغة الـ VHDL.

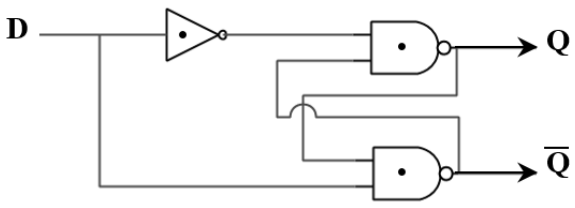
2. الحابسة TDL:

تعمل الحابسة من دون الحاجة إلى نبضات الإيقاع، وتقوم بحفظ معلومة ثلاثية واحدة تستقبلها على طرف الدخل D للحابسة، والخرج Q للحابسة يتبع دائماً ما هو على طرف الدخل D. ويوضح شكل (1) الرمز الهندسي للحابسة بثلاثية واحدة (One-Trit D Latch) وجدول التشغيل والدائرة المنطقية للحابسة والمكونة من البوابات STNAND وSTI.

D	Q	$\overline{Q}$
0	0	2
1	1	1
2	2	0



(ب) (أ)



(ج)

شكل (1): الحابسة TDL بثلاثية واحدة

(أ) الرمز الهندسي (ب) جدول التشغيل (ج) الدائرة المنطقية التجميعية المحققة للـ TDL

أما الوصف اللغوي لعمل الـ TDL فيمكن أن يكون في الصورة التالية :

```
Tcomp: TDL(1:1)
Inputs: in
Outputs: out
Begin
   out = in
End
```

أو في الصورة التالية :

```
Tcomp: TDL(1:1)
Inputs: in
Outputs: out
Begin
   Case in of
   Begin
      0: out = 0
      1: out = 1
 2: out = 2
   End
End
```

أما الوصف اللغوي لحابسة بطول 9 ثلاثيات (Nine-Trit D Latch) يكون في الصورة التالية :

```
Tcomp: TL (9:9)
Inputs: in [9]
Outputs: out [9]
Begin
  for ( i = 0 ; i < 9 ; i ++ )
  out [i] = in [i]
End
```

أو في الصورة التالية :

```
Tcomp: TL (9:9)
Inputs: in [9]
```


https://doi.org/10.20428/JST.24.2.3

```
Outputs: out [9]
Begin
  for ( i = 0 ; i < 9 ; i ++ )
  {
    Case in[i] of
    Begin
      0: out[i] = 0
      1: out[i] = 1
      2: out[i] = 2
    End
  }
  End
```

وبنفس الأسلوب يمكن كتابة الوصف اللغوي للحابسة بأي طول.

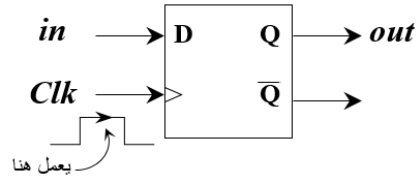
3. النطاطة الثلاثية TDF:

تعتبر النطاطة عنصر حفظ المعلومة الرقمية بثلاثية واحدة (One-Trit Memory Element)، حيث تستقبل المعلومة على طرف الدخل ثم تظهر هذه المعلومة على طرف الخرج ويتحكم بظهورها نبضة الإيقاع، والنطاطة على نوعين من حيث تأثير نبضة الإيقاع:

(1) النوع الأول (Level-triggered FF)

يعمل هذا النوع عند المستوى المرتفع لنبضة الإيقاع وليس عند الحافة الموجبة أو الحافة السالبة للنبضة، هذا يعني أن الإشارة على طرف الدخل D (النطاطة D مثلاً) يجب أن تبقى مستقرة خلال الفترة الزمنية للمستوى المرتفع لنبضة الإيقاع (عرض النبضة أو عمر النبضة). هذا النوع من النطاطات لا يستخدم في الحاسبات الرقمية والنظم الرقمية المشابهة ويستخدم في نظم رقمية أخرى، ويوضح شكل (2) الرمز الهندسي وجدول التشغيل لهذا النوع من النطاطات.

Clk	D	Q	$\overline{Q}$
0	-	NC	NC
2	0	0	2
2	1	1	1
2	2	2	0



NC := No Change

(ب) **(أ)**

شكل (2): النطاطة TDF التي تعمل عند المستوى المرتفع للنبضة
(أ) الرمز الهندسي (ب) جدول التشغيل

أما الوصف اللغوي لعمل هذه النطاطة يكون في الصورة التالية :

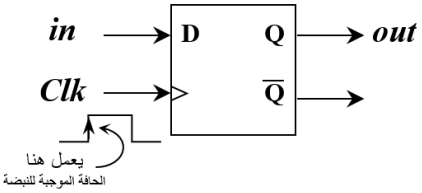
```
Tcomp: TDF(2:1)
Inputs: in, clk
Outputs: out
Begin
  Case clk(t) of
  Begin
    0: case in(t) of
    Begin
       0،1،2: out(t) = out (t-1)
    End
    2: case in(t) of
    Begin
       0: out(t) = 0
       1: out(t) = 1
       2: out(t) = 2
    End
  End
End
```

(2) النوع الثاني (Edge-triggered TDF)

يعمل هذا النوع عند حافة النبضة (تغير النبضة من 0 إلى 2 [+] أو عند تغير النبضة من 2 إلى 0 [-])، وهذا النوع هو الذي يستخدم في الحاسبات الرقمية والنظم الرقمية المشابهة، ويوضح شكل (3) الرمز الهندسي وجدول التشغيل لهذا النوع من النطاطات.

Clk	D	Q	$\overline{Q}$
0	-	NC	NC
2	-	NC	NC
↓	-	NC	NC
↑	**0**	0	2
↑	**1**	1	1
↑	**2**	2	0

(ب)



(أ)

شكل (3): النطاطة TDF التي تعمل عند حافة النبضة
(أ) الرمز الهندسي لـ TDF التي تعمل عند الحافة الموجبة أو الصاعدة (Rising)
(ب) جدول التشغيل لـ TDF التي تعمل عند الحافة الموجبة

https://doi.org/10.20428/JST.24.2.3

أما الوصف اللغوي لعمل هذا النوع من النطاطات وفقاً لجدول التشغيل يكون في الصورة التالية :

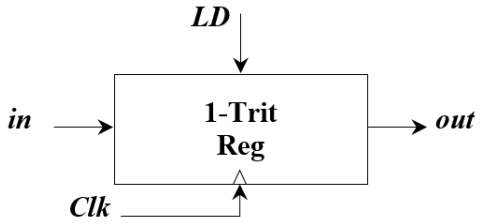
```
Tcomp: TDF(2:1)
Inputs: in, clk
Outputs: out
Begin
   Case clk(t) of
   Begin
     0: case in(t) of
     Begin
       0،1،2: out(t) = out (t-1)
     End
     2: case in(t) of
     Begin
       0،1،2: out(t) = out (t-1)
     End
     2 to 0: case in(t) of
     Begin
       0،1،2: out(t) = out (t-1)
     End
      0 to 2: case in(t) of
     Begin
       0: out(t) = 0
       1: out(t) = 1
       2: out(t) = 2
     End
   End
End
```

4. السجل TRE:

يعتبر السجل جهاز رقمي لخزن المعلومة التي وضعت فيه دون تغيير حتى يتم وضع معلومة جديدة. يتحكم بوظيفة السجل إشارة تعرف بإشارة التحميل LD (Load Signal)، فإن كانت إشارة التحميل غير فعالة لا يحدث تغيير في محتوى السجل بالإشارات المتواجدة على مداخله حتى وإن كانت نبضة الإيقاع فعالة. أما إذا كانت إشارة التحميل فعالة ومع وجود نبضة الإيقاع يحدث استبدال لمحتوى السجل بالإشارات المتواجدة على مداخل السجل (مثل النطاطة تماماً). ويوضح شكل (4) الرمز الهندسي وجدول التشغيل لسجل بثلاثية واحدة (One-Trit Register)، بينما يوضح شكل (5) الرمز الهندسي لسجل بطول 9 ثلاثيات (Nine-Trit Register).

Clk	LD	In	Out
0	-	-	NC
2	-	-	NC
↓	-	-	NC
↑	0	-	NC
↑	2	0	0
↑	2	1	1
↑	2	2	2

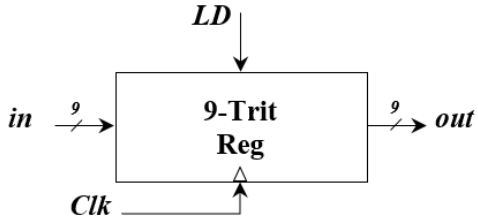
(ب)



(أ)

شكل (4): سجل بثلاثية واحدة TRE

(أ) الرمز الهندسي (ب) جدول التشغيل



شكل (5): الرمز الهندسي لسجل بطول 9 ثلاثيات

أما الوصف اللغوي لعمل السجل في شكل (4 -أ) يكون في الصورة التالية:

```
Tcomp: TRE(2:1)
Inputs: in, LD
Outputs: out
Begin
    if LD(t) Then
      out(t) = in (t)
```


https://doi.org/10.20428/JST.24.2.3

```
    else
        out(t) = out (t-1)
End
```

والوصف اللغوي لعمل السجل في شكل (5) يكون في الصورة التالية :

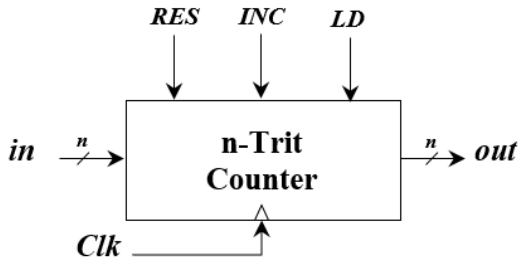
```
Tcomp: TRE(10:9)
Inputs: in[9], LD
Outputs: out[9]
Begin
    if LD(t) Then
        for ( i = 0 ; i <9 ; i++ )
          out(t)[i] = in (t)[i]
    else
        for ( i = 0 ; i <9 ; i++)
          out(t)[i] = out (t-1)[i]
End
```

وبالمثل يمكن كتابة الوصف اللغوي لسجل بأي طول.

5. العدادات TCN:

من أهم استخدامات العدادات في الحاسوب توليد عنوان الأمر الذي سيتم احضاره من الحافظة لتنفيذه، ويقوم العداد في أغلب الحالات بزيادة محتواه مع كل نبضة إيقاع حتى يكون محتواه يشير إلى موقع الأمر التالي في البرنامج تحت التنفيذ، بينما في حالات أخرى يتم تحميل العداد بقيمة جديدة لا علاقة لها بمحتوى العداد السابق مثلاً في حالة أمر القفز Jump للانتقال إلى أمر في البرنامج ليس هو الأمر التالي للأمر الأول الذي تم تنفيذه. فعلى سبيل المثال الأمر المراد تنفيذه له العنوان n، في هذا الوضع يجب أولاً وضع العنوان n في العداد ليبدأ المعالج بتنفيذ البرنامج بدءاً بالأمر في الموقع n ثم بعد إحضار الأمر الذي عنوانه n يزيد محتوى العداد ليشير إلى الأمر الذي عنوانه n+1 ثم إلى الأمر الذي عنوانه n+2 ....الخ. أيضاً قد يحدث إعادة لتنفيذ البرنامج في أي وقت بعد أن يكون العداد قد صُفر، أي أن محتوى العداد يساوي صفر، على فرض أن الصفر في العداد يشير إلى عنوان أول أمر في البرنامج، وبناءً على ما تقدم لا بد أن يكون للعداد طرف تحميل (Load, LD) وطرف تصفير (Reset, RES) وطرف زيادة (Increment, INC)، لهذا فالعدادات تشبه السجلات فيما عدا أن العداد له طرفان RES وINC ولا يتواجد هذان الطرفان في السجل. كذلك يمكن جعل العداد يقوم بوظيفة السجل من خلال إلغاء التفعيل للطرف INC والعكس غير ممكن.

يوضح شكل (6) الرمز الهندسي للعداد على أساس ما تقدم.



شكل (6):الرمز الهندسي لعداد ثلاثي بـ n ثلاثية

الوصف اللغوي لعمل عداد بثلاثيتين (للعد من 0 إلى 8) يكون في الصورة التالية :

```
Tcomp: TCN (5:2)
Inputs: in[2], LD, INC, RES
Outputs: out[2]
Begin
   for ( i = 0 ; i <2 ; i++ )
  {
      if RES(t) Then
            out(t)[i] = 0
      else if LD(t) Then
            out(t)[i] = in(t)[i]
      else if INC(t) Then
            out(t)[i] = out(t-1)[i] + 1
      else
            out (t)[i] = out(t-1)[i]
    }
End
```

6. الحافظة TRAM :

الحافظة TRAM عبارة عن عدد N من السجلات وكل سجل بطول m ثلاثية، ويكون حجم أو سعة الـ TRAM يساوي N*m ثلاثية حيث $N = 3^n$ و n عدد خطوط حافلة العناوين و m تمثل عدد خطوط حافلة البيانات.

بالإضافة إلى حافلة العناوين وحافلة البيانات توجد أطراف أخرى للـ TRAM أهمها طرف الكتابة (Write, WR) أو التحميل (Load,LD) وطرف القراءة (Read, RD). فعندما يراد حفظ بيانات جديدة في موقع ما في الحافظة يتم وضع عنوان الموقع على حافلة العناوين ثم توضع البيانات على حافلة

https://doi.org/10.20428/JST.24.2.3

البيانات ثم تفعل إشارة الكتابة WR (أو التحميل LD) ليحدث عند تفعيلها وضع البيانات المحمولة على حافلة البيانات في الموقع الذي عنوانه على حافلة العناوين، ويلخص عملية الكتابة العبارة التالية:

WR: M[ABUS] = DBUS

أو في الصورة التالية:

if WR Then M[ABUS] = DBUS

أما عندما يراد قراءة أو إخراج محتوى موقع ما في الحافظة يوضع عنوان الموقع على حافلة العناوين ثم تفعل إشارة القراءة RD ليحدث عند تفعيلها وضع محتوى الموقع الذي عنوانه على حافلة العناوين على حافلة البيانات، ويلخص عملية القراءة العبارة التالية:

RD: DBUS = M[ABUS]

أو في الصورة التالية:

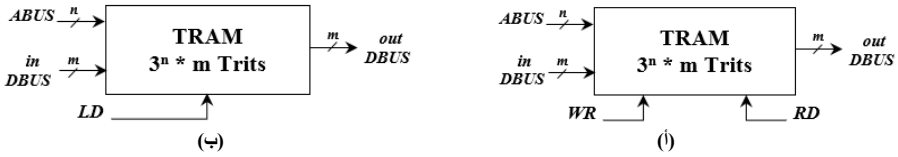
if RD Then DBUS = M[ABUS]

هناك حافظات يكون لها طرف واحد فقط لتفعيل عمليتي الكتابة والقراءة ويعطى لهذا الطرف التسمية (مثلاً Load, LD) فعندما تكون الإشارة على هذا الطرف في المستوى المرتفع تحدث عملية الكتابة في الحافظة وعندما تكون الإشارة على هذا الطرف في المستوى المنخفض تكون عملية القراءة، وتوضح العبارة التالية عمليتي الكتابة والقراءة:

if LD Then M[ABUS] = DBUS

else DBUS = M[ABUS]

يوضح شكل (7) الرمز الهندسي لـ TRAM بنوعيها.



شكل (7):الرمز الهندسي لـ TRAM

(أ) تواجد الطرفان WR, RD (ب) تواجد طرف واحد هو LD

أما الوصف اللغوي لعمل الـ TRAM والتي لها n=18, m=18, WR, RD يكون في الصورة التالية:

```
Tcomp: TRAM (38:18)
Inputs: in[18], ABUS[18], RD, WR
Outputs: out[18]
Begin
    //Read Operation
    if RD Then
```

```
    for ( i = 0 ; i <18 ; i++ )
    out [i] = TRAM [ABUS][i]
//Write Operation
if WR Then
    for ( i = 0 ; i <18 ; i++ )
      TRAM [ABUS][i] = in [i]
End
```

أما الوصف اللغوي لعمل الـ TRAM والتي لها LD ,m=18 ,n=18 يكون في الصورة التالية :

```
Tcomp: TRAM (37:18)
Inputs: in[18], ABUS[18], LD
Outputs: out[18]
Begin
  //Read Operation
  for ( i = 0 ; i <18 ; i++ )
     out [i] = TRAM [ABUS][i]
//Write Operation
if LD Then
    for ( i = 0 ; i <18 ; i++ )
      TRAM [ABUS][i] = in [i]
End
```

أو في الصورة التالية :

```
Tcomp: TRAM (37:18)
Inputs: in[18], ABUS[18], LD
Outputs: out[18]
Begin
    if LD Then
    for ( i = 0 ; i <18 ; i++ )
      TRAM [ABUS][i] = in [i]
else
```

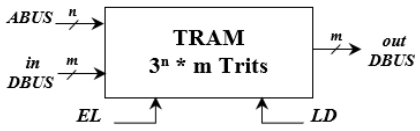

https://doi.org/10.20428/JST.24.2.3

```
    for ( i = 0 ; i <18 ; i++ )
      out [i] = TRAM [ABUS][i]
End
```

يتضح مما تقدم أن الـ TRAM التي لها طرف LD للتحكم بعمليتي القراءة والكتابة تظهر فيها مشكلة وهي أنه عندما تكون الإشارة على الطرف LD في المنطق "0" لتهيئة عملية القراءة، تكون محتويات الموقع في الحافظة والذي عنوانه على ABUS على حافلة البيانات حتى وإن كان الـ ABUS يحمل العنوان "0" تكون محتويات الموقع "0" على حافلة البيانات (out) وهذا غير مقبول، ولهذا لا بد من إضافة طرف أخر يتحكم في عملية القراءة يمكن أن يسمى بخط التمكين (Enable Line, EL)، الإشارة على هذا الخط هي التي تسمح بوضع محتوى الموقع الذي عنوانه على الـ ABUS على حافلة البيانات عندما تكون الإشارة فعالة وعندما تكون الإشارة على الـ EL غير فعالة تمنع خروج البيانات من الموقع الذي عنوانه على الـ ABUS.

يوضح شكل (8) الرمز الهندسي للـ TRAM بوجود الخط EL وجدول التشغيل. الحالة الأولى في جدول التشغيل تحدث عندما يكون النظام خاملاً، أما الحالة الرابعة لا يمكن أن تحدث لأن وحدة التحكم التي تصدر إشارة الـ LD وإشارة الـ EL تُفعل إشارة واحدة فقط إما LD يكون في المستوى المرتفع و EL في المستوى المنخفض أو العكس.

LD	EL	Operation	Out
0	0	Hold	Floating
0	2	Read	Connected
2	0	Write	Floating
2	2	Hold	Floating



شكل (8): الرمز الهندسي وجدول التشغيل لـ TRAM بوجود الـ LD والـ EL

الوصف اللغوي لعمل الـ TRAM بالمقاس $3^{18}*18$ ثلاثية بوجود الـ LD والـ EL يكون في الصورة التالية :

```
Tcomp: TRAM (38:18)
Inputs: in[18], ABUS[5], LD, EL
Outputs: out[18]
Begin
if LD Then
    for ( i = 0 ; i <18 ; i++ )
      TRAM [ABUS][i] = in [i] // Write operation
else if EL Then
    for ( i = 0 ; i <18 ; i++ )
      out [i] = TRAM [ABUS][i] //Read operation
End
```

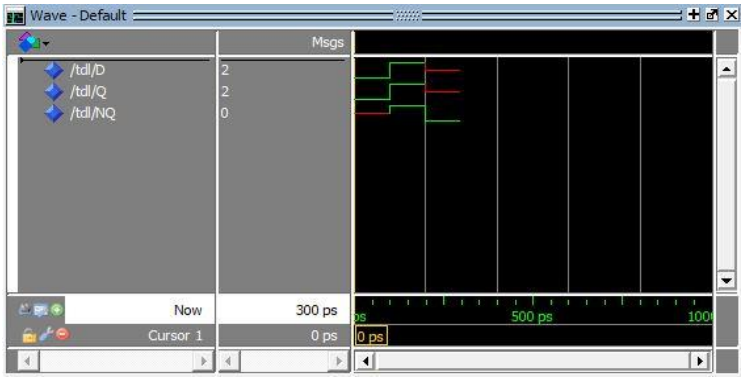
أو في الصورة التالية التي تحاكي جدول التشغيل في شكل (8):

```
Tcomp: TRAM (38:18)
Inputs: in[18], ABUS[18], LD, EL
Outputs: out[18]
Begin
    Case LD of
    Begin
     0: case EL of
    Begin
     0: for ( i = 0 ; i <18 ; i++ )
     Out[i] = HI-IMP  // floating
     2: for ( i = 0 ; i <18 ; i++ )
     Out[i] = TRAM [ABUS][i] //Read
End
2: case EL of
Begin
  0: for ( i = 0 ; i <18 ; i++ )
       TRAM [ABUS][i] = in [i] // Write
       2: for ( i = 0 ; i <18 ; i++ )
       Out[i] = HI-IMP  // floating
      End
    End
End
```

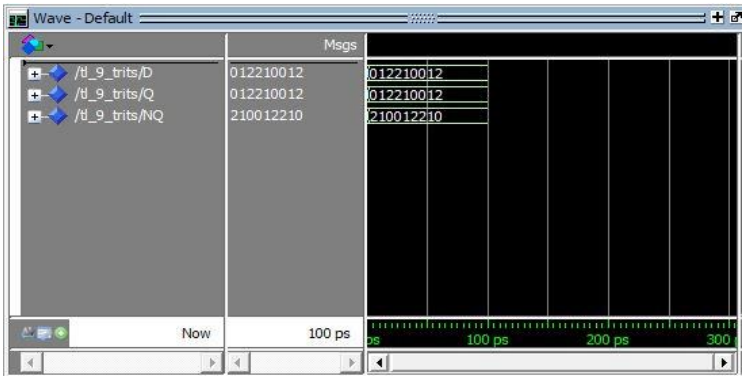
7.التنفيذ بلغة الـ VHDL:

سيتم في هذا البند تنفيذ كل القطع التي عرضت في البنود السابقة بلغة الـVHDL، وبرنامج الـVHDL المحقق لكل قطعة من القطع السابقة مرفق في الملحق لهذه الورقة العلمية، وتوضح الأشكال من شكل (9) إلى شكل (18) نتائج التنفيذ للبرامج المكتوبة بلغة الـVHDL باستخدام برنامج المحاكاة Modelsim من شركة Mentor Graphics والذي يعرض نتائج المحاكاة على صورة موجات Waveforms وهي من أسهل الطرق لاستخلاص وإدراك النتائج. أما المكتبة التي تم استخدامها كأساس لتنفيذ القطع فهي نفس المكتبة المستخدمة في المرحلة الأولى والثانية من الدراسة [1,3]. مع الأخذ بعين الاعتبار أن نبضات الإيقاع تم تنفيذها للعمل بين المنطق 0 و1 للتبسيط وإظهار النتائج بشكل أوضح.

https://doi.org/10.20428/JST.24.2.3

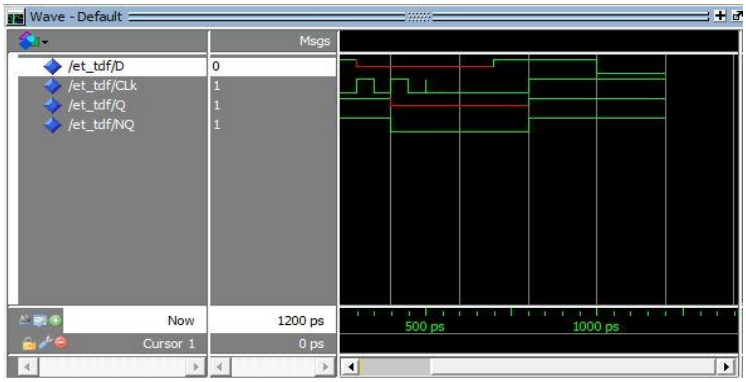


شكل (9): نتائج التنفيذ للحابسة TDL بثلاثية واحدة

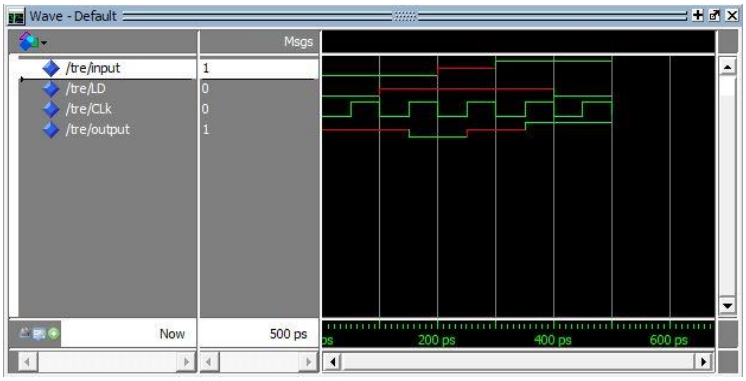


شكل (10): نتائج التنفيذ للحابسة TL بطول 9 ثلاثيات

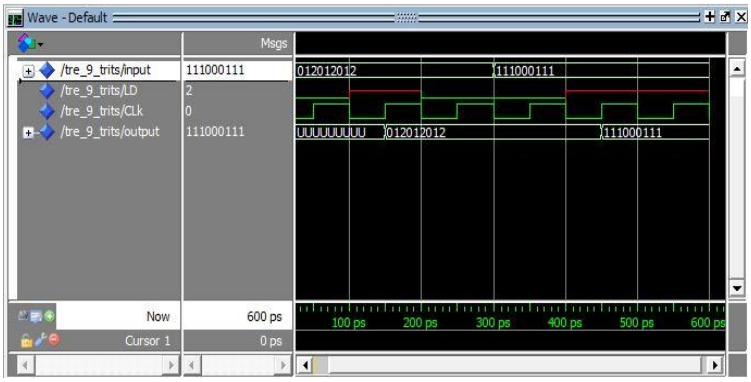
شكل (11): نتائج التنفيذ للنطاطة TDF التي تعمل عند المستوى المرتفع للنبضة



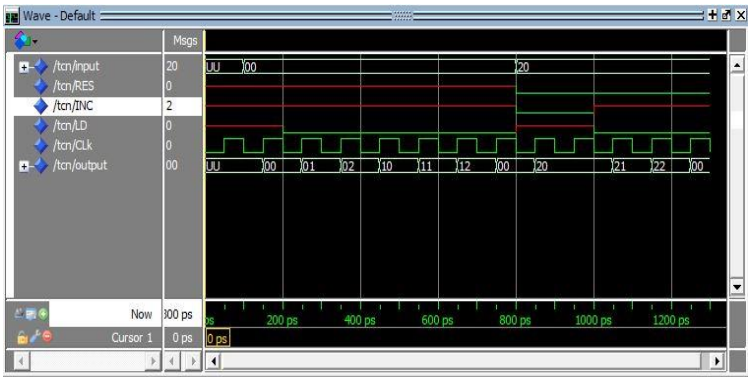
شكل (12): نتائج التنفيذ للنطاطة TDF التي تعمل عند الحافة الموجبة أو الصاعدة للنبضة



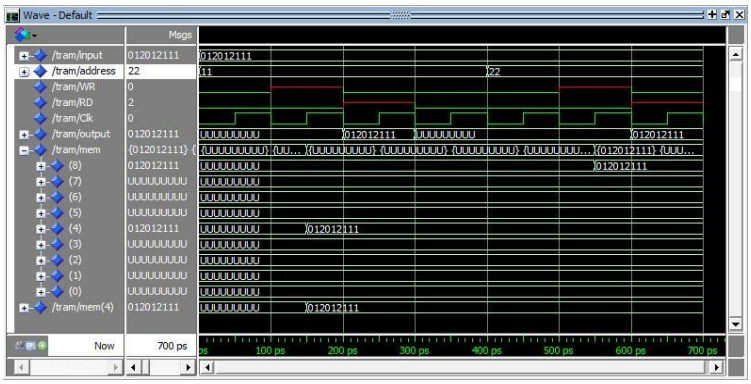
شكل (13): نتائج التنفيذ للسجل TRE بثلاثية واحدة



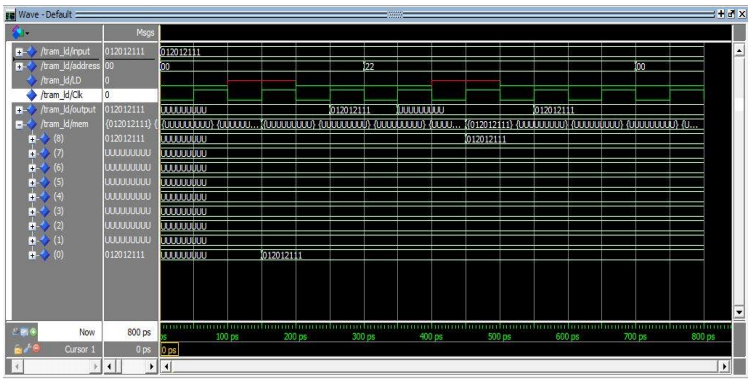
شكل (14): نتائج التنفيذ للسجل TRE بطول 9 ثلاثيات



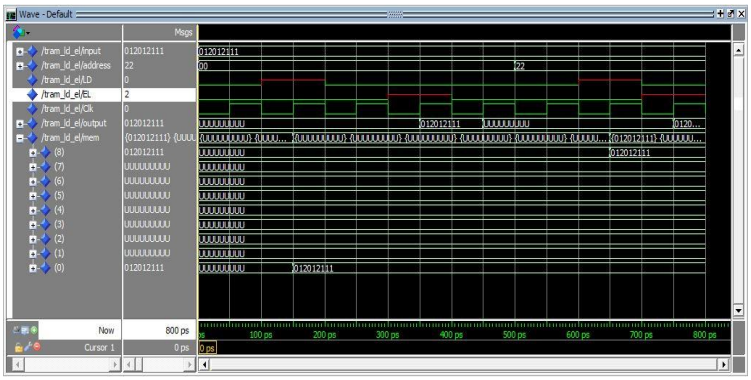
شكل (15): نتائج التنفيذ للعداد TCN بطول ثلاثيتين



شكل (16): نتائج التنفيذ للحافظة TRAM بالحجم 9×3^2 ذات الأطراف RD، WR



شكل (17): نتائج التنفيذ للحافظة TRAM بالحجم 9×3^2 ذات الطرف LD فقط



شكل (18): نتائج التنفيذ للحافظة TRAM بالحجم 9×3^2 ذات الأطراف LD، EL

8. الخاتمة:

تم في هذه المرحلة من الدراسة تنفيذ القطع المنطقية الثلاثية بلغة الـ VHDL بدءاً بالحابسة (Latch) وختاماً بالحافظة TRAM (Ternary Memory) وأضيفت جميعها إلى المكتبة البرامجية للمنطق الثلاثي والمؤسسة على لغة الـ VHDL. وقد تم تنفيذ المحاكاة لجميع القطع وإظهار النتائج باستخدام برنامج Modelsim.

المراجع:

[1] **زكريا محمد حسن، عبدالله علي قاسم وعبدالرقيب عبده أسعد، "أتمتة النظم المنطقية الإلكترونية الثلاثية: دراسة مستحدثة مؤسسة على لغة VHDL - المرحلة الأولى: البوابات المنطقية الثلاثية"، مجلة العلوم والتكنولوجيا، المجلد 22، العدد 2، 2017، ص 28 - 47.**

[2] **عبدالرقيب عبده أسعد، "المنطق الرقمي الثلاثي وأسس تصميم الحاسوب"، مطبوعات جامعة العلوم والتكنولوجيا، الطبعة الأولى، صنعاء، 2001.**

[3] **عبدالله علي قاسم الحميدي وعبدالرقيب عبده أسعد، "أتمتة النظم المنطقية الإلكترونية الثلاثية: دراسة مستحدثة مؤسسة على لغة VHDL - المرحلة الثانية: القطع المنطقية التجميعية الثلاثية"، مجلة العلوم والتكنولوجيا، المجلد 23، العدد 2، 2018، ص 1 - 30.**

[4] R. Vacca, "A three-valued system of logic and its application to base three digital circuits," in IFIP Congress, 1959, pp. 407-413.

[5] M. Yoeli and G. Rosenfeld, "Logical design of ternary switching circuits," IEEE Transactions on Electronic Computers, vol. EC-14, pp. 19-29, 1965.

[6] R. D. Merrill Jr, "Ternary logic in digital computers," in Proceedings of the SHARE design automation project, 1965, pp. 6.1-6.17.

[7] I. Halpern and M. Yoeli, "Ternary arithmetic unit," in Proceedings of the Institution of Electrical Engineers, 1968, pp. 1385-1388.

[8] D. Porat, "Three-valued digital systems," in Proceedings of the Institution of Electrical Engineers, 1969, pp. 947-954.

https://doi.org/10.20428/JST.24.2.3

[9] H. Mouftah, "A study on the implementation of three-valued logic," in Proceedings of the sixth international symposium on Multiple-valued logic, 1976, pp. 123-126.

[10] C.-Y. Wu and H.-Y. Huang, "Design and application of pipelined dynamic CMOS ternary logic and simple ternary differential logic," IEEE journal of solid-state circuits, vol. 28, pp. 895-906, 1993.

[11] A. Srivastava and K. Venkatapathy, "Design and implementation of a low power ternary full adder," VLSI Design, vol. 4, pp. 75-81, 1996.

[12] H. Mouftah and K. Smith, "Design and implementation of three-valued logic systems with MOS integrated circuits," in IEE Proceedings G-Electronic Circuits and Systems, 1980, pp. 165-168.

[13] A.R.A. Asaad, "Minimization of Ternary Reed-Muller Switching Fuctions with Fixed Polarity," Journal of Engineering, vol. 2, No. 2, 1992, pp. 37-54.

ملحق

برنامج لغة الـ VHDL للحابسة TDL بثلاثية واحدة

```
USE WORK.ternary_type.ALL;
entity TDL is
   PORT (D    : IN  t_logic;
            Q,NQ : OUT t_logic);
end entity;
architecture arc of TDl is
begin
        Q  <= D;
        NQ <= stnot(D);
end arc;
```

برنامج لغة الـ VHDL للحابسة TL بطول 9 ثلاثيات

```
USE WORK.ternary_type.ALL;
entity TL_9_trits is
   PORT (D    : IN  t_logic_vector(8 downto 0);
            Q,NQ : OUT t_logic_vector(8 downto 0));
end entity;
architecture arc of TL_9_trits is
```

```
begin
U1: for i in 8 downto 0 generate
                Q(i)  <= D(i);
        NQ(i) <= stnot(D(i));
        end generate;
end arc;
```

برنامج لغة الـ VHDL للنطاطة TDF التي تعمل عند المستوى المرتفع للنبضة

```
IIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE WORK.ternary_type.ALL;
entity LT_TDF is
   PORT (D    : IN  t_logic;
            CLk  : IN  std_logic;
            Q,NQ : OUT t_logic);
end entity;
architecture arc of LT_TDF is
begin
        process (Clk, D)
        begin
                if Clk = '1' then
                   Q <= D;
                   NQ<= stnot(D);
                end if;
        end process;
end arc;
```


https://doi.org/10.20428/JST.24.2.3

برنامج لغة الـ VHDL للنطاطة TDF التي تعمل عند الحافة الموجبة أو الصاعدة للنبضة

```
lIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE WORK.ternary_type.ALL;
entity ET_TDF is
  PORT (D    : IN  t_logic;
        CLk  : IN  std_logic;
        Q,NQ : OUT t_logic);
end entity;
architecture arc of ET_TDF is
begin
      process (Clk, D)
      begin
            if Clk'event and Clk = '1' then
            Q <= D;
              NQ<= stnot(D);
            end if;
      end process;
end arc;
```

برنامج لغة الـ VHDL للسجل TRE بثلاثية واحدة

```
lIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE WORK.ternary_type.ALL;
entity TRE is
  PORT (input : IN  t_logic;
        LD    : IN  t_logic;
        CLk   : IN  std_logic;
        output: OUT t_logic);
end entity;
```

```
architecture arc of TRE is
begin
        process (Clk, LD)
        begin
                if Clk'event and Clk = '1' then
                if LD = '2' then
                                output <= input;
                   end if;
                end if;
        end process;
end arc;
```

برنامج لغة الـ VHDL للسجل TRE بطول 9 ثلاثيات

```
lIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE WORK.ternary_type.ALL;
entity TRE_9_trits is
   PORT (input : IN  t_logic_vector(8 downto 0);
            LD    : IN  t_logic;
            CLk   : IN  std_logic;
            output: OUT t_logic_vector(8 downto 0));
end entity;
architecture arc of TRE_9_trits is
begin
        process (Clk, LD)
        begin
                if Clk'event and Clk = '1' then
                if LD = '2' then
                        output <= input;
                   end if;
```


https://doi.org/10.20428/JST.24.2.3

```
            end if;
        end process;
end arc;
```

برنامج لغة الـ VHDL للعداد TCN بطول ثلاثيتين

```
lIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE WORK.ternary_type.ALL;
entity TCN is
  PORT (input : IN  t_logic_vector(1 downto 0);
        RES   : IN  t_logic;
        INC   : IN  t_logic;
        LD    : IN  t_logic;
        CLk   : IN  std_logic;
        output: OUT t_logic_vector(1 downto 0));
end entity;
architecture arc of TCN is
begin
      process (Clk, RES, INC, LD)
      variable v : integer;
      variable out_buf : t_logic_vector(1 downto 0);
      begin
        v:= to_dec(out_buf);
            if Clk'event and Clk = '1' then
            if RES = '2' then
                        out_buf := "00";
                elsif LD = '2' then
                        out_buf := input;
                elsif INC = '2' then
                  v := v+1;
```

```
                    out_buf := to_ternary (v,2);
                end if;
            end if;
            output <= out_buf;
        end process;
end arc;
```

برنامج لغة الـ VHDL للحافظة TRAM بالحجم 9×3^2 ذات الأطراف RD، WR

```
lIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE WORK.ternary_type.ALL;
entity TRAM is
  GENERIC(data_width   : integer := 9;
          address_width: integer := 2);
       PORT (input  : IN  t_logic_vector(data_width-1 downto 0);
             address: IN  t_logic_vector(address_width-1 downto 0);
             WR     : IN  t_logic;
             RD     : IN  t_logic;
             Clk    : IN  std_logic;
             output : OUT t_logic_vector(data_width-1 downto 0));
end entity;
architecture arc of TRAM is
type memtype is array ((3**address_width)-1 downto 0) of
t_logic_vector(data_width-1 downto 0);

signal mem : memtype;
begin
  --read
       output <= mem(to_dec(address)) when RD='2'
        else "UUUUUUUUU";
```


https://doi.org/10.20428/JST.24.2.3

```
    process (Clk, WR)
    begin
        if Clk'event and Clk='1' then
            --write
            if WR = '2' THEN
                mem(to_dec(address)) <= input;
            end if;
        end if;
    end process;
end arc;
```

برنامج لغة الـ VHDL للحافظة TRAM بالحجم 9×3^2 ذات الطرف LD فقط

```
lIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE WORK.ternary_type.ALL;
entity TRAM_LD is
  GENERIC(data_width   : integer := 9;
        address_width: integer := 2);
      PORT (input  : IN  t_logic_vector(data_width-1 downto 0);
          address: IN  t_logic_vector(address_width-1 downto 0);
          LD     : IN  t_logic;
          Clk    : IN  std_logic;
          output : OUT t_logic_vector(data_width-1 downto 0));
end entity;
architecture arc of TRAM_LD is
type memtype is array ((3**address_width)-1 downto 0) of
t_logic_vector(data_width-1 downto 0);

signal mem : memtype;
begin
```

```
        process (Clk, LD)
        begin
                if Clk'event and Clk='1' then
                        if LD = '2' THEN
                                mem(to_dec(address)) <= input;
                        elsif LD = '0' THEN
                                output <= mem(to_dec(address));
                        else
                                output <=  "UUUUUUUUU";
                        end if;
                end if;
        end process;
end arc;
```

برنامج لغة الـ VHDL للحافظة TRAM بالحجم 9×3^2 ذات الأطراف LD، EL

```
IIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE WORK.ternary_type.ALL;
entity TRAM_LD_EL is
   GENERIC(data_width   : integer := 9;
           address_width: integer := 2);
        PORT (input  : IN  t_logic_vector(data_width-1 downto 0);
              address: IN  t_logic_vector(address_width-1 downto 0);
              LD     : IN  t_logic;
              EL     : IN  t_logic;
              Clk    : IN  std_logic;
              output : OUT t_logic_vector(data_width-1 downto 0));
end entity;
architecture arc of TRAM_LD_EL is
type memtype is array ((3**address_width)-1 downto 0) of
```


https://doi.org/10.20428/JST.24.2.3

```
t_logic_vector(data_width-1 downto 0);
signal mem : memtype;
begin
        process (Clk, LD, EL)
        begin
                if Clk'event and Clk='1' then
                        if LD = '2' THEN
                                mem(to_dec(address)) <= input;
                        elsif LD = '0' and EL = '2' THEN
                                output <= mem(to_dec(address));
                        else
                                output <=  "UUUUUUUUU";
                        end if;
                end if;
        end process;
end arc;
```