أتمتة النظم المنطقية الإلكترونية الثلاثية: دراسة مستحدثة مؤسسة على لغة VHDL - المرحلة الأولى: البوابات المنطقية الثلاثية

زكريا محمد حسن الشيخي [1،*]
عبدالله علي قاسم قحطان الحميدي [1،*]
عبدالرقيب عبده أسعد [1،*]

[1] قسم الهندسة الإلكترونية، كلية الهندسة ، جامعة العلوم والتكنولوجيا، صنعاء، اليمن
*عناوين المراسلة: zakariaalsheikh1993@gmail.com . a.qahtan2@ust.edu . abdarraqib62@yahoo.com

https://doi.org/10.20428/JST.22.2.2

أتمتة النظم المنطقية الإلكترونية الثلاثية: دراسة مستحدثة مؤسسة على لغة VHDL – المرحلة الأولى: البوابات المنطقية الثلاثية

الملخص:

سيتم في هذه الورقة العلمية بناء مكتبة برامجية على أساس لغة الـ VHDL لبوابات المنطق الثلاثي لتستخدم في تنفيذ الدوائر المنطقية التجميعية الثلاثية. وحيث أن لغة الـ VHDL صممت أساساً لأتمتة تصميم النظم الإلكترونية الثنائية، فقد برزت عدة مشاكل في استخدام الـ VHDL مع النظم المنطقية الإلكترونية الثلاثية. غير أنه تم حل العديد من المشاكل وكانت النتائج مشجعه لمواصلة الدراسة للمراحل الأخرى وصولاً إلى بناء مكتبة برامجية بلغة الـ VHDL لقطع المنطق الثلاثي المختلفة لتستخدم في تصميم الدوائر والنظم المنطقية الإلكترونية الثلاثية ومحاكاة عملها.

الكلمات المفتاحية : المنطق الثلاثي، بوابات المنطق الثلاثي، لغة الـ VHDL.

Ternary Electronic Logic Systems Automation: A Novel Study Based on VHDL Language – First Part: Ternary Logic Gates

Abstract:

In this scientific paper, a software library for Ternary logic gates will be built based on VHDL language to be used in the implementation of combinational Ternary logic circuits. The VHDL language had been designed to be used to automate and design Binary electronic logic systems. Therefore, several problems have been arised in adapting the VHDL language to be used with Ternary electronic logic systems. However, these problems have been solved and the result were encouragement to continue the study with the other Ternary components, so that a complete software library based on VHDL language for different Ternary components will be used as a tool in the design and simulation of Ternary electronic logic circuits and systems.

Keywords: Ternary logic, Ternary logic gates, VHDL language.

https://doi.org/10.20428/JST.22.2.2

ا.المقدمة:

النظام الحسابي العشري نظام رياضي أساسه الرقم "10" وأرقامه (0، 1، 2، 3،....9)، وهي عشرة أرقام تُعرف بالأرقام العشرية، ويستخدم هذا النظام في المعاملات اليومية بين البشر. أما النظام الحسابي الثنائي فهو نظام رياضي أساسه الرقم "2" وأرقامه (0، 1)، وهما رقمان ثنائيان، وعلى أساس هذا النظام الثنائي صممت وأنتجت الأجهزة والنظم المنطقية الإلكترونية المتواجدة في الأسواق والمستخدمة في كثير من مجالات الحياة وأهم هذه الأجهزة هو جهاز الحاسوب الرقمي (Digital Computer) بأنواعه وأشكاله وأحجامه المختلفة. فالرقم الثنائي "0" أو "1" هو أصغر معلومة رقمية تُحمل على خط نقل (سلك) واحد أو تُحفظ في عنصر الحفظ المعروف بالنطاطة (Flip-Flop). لو أعتبر الرقم العشري $(8)_{10}$ وهو رقم عشري واحد، فعندما يحول هذا الرقم العشري إلى النظام الثنائي يكون في الصورة $(1000)_2$ أي أن $(8)_{10} = (1000)_2$، أي أن الرقم العشري "8" يحول إلى أربعة أرقام ثنائية.

لو أن النظام العشري أستخدم لإنتاج قطع إلكترونية رقمية عشرية، فإن الرقم "8" يحمل على خط نقل واحد فقط ويحفظ في عنصر حفظ عشري واحد فقط، بينما العدد $(1000)_2$ يتطلب أربعة خطوط لنقله كل خط يحمل رقماً ثنائياً واحداً فقط ويتطلب حفظه في أربعة عناصر حفظ رقمية ثنائية.

يأتي بعد النظام اثنائي النظام الثلاثي وهو نظام رياضي أساسه الرقم "3" وأرقامه (0، 1، 2) وهي ثلاثة أرقام، ولو رجعنا إلى المثال السابق فإن $(8)_{10} = (1000)_2 = (22)_3$. أي أن الرقم العشري "8" يمثل في النظام الثنائي بأربعة أرقام ثنائية ويمثل في النظام الثلاثي برقمين ثلاثيين. هذا يعني أن الرقم العشري "8" يتطلب حمله على خطين إثنين في النظام الثلاثي ويتطلب حفظه في عنصري حفظ في النظام الثلاثي، وجميع هذه المتطلبات أقل من متطلبات النظام الثنائي (4 خطوط نقل و4 عناصر حفظ).

يتضح مما تقدم أنه كلما أستخدم نظام رياضي أساسه أكبر من "2" كلما تم نقل معلومة كبيرة على خطوط نقل أقل عدداً من الخطوط المطلوبة لنقل نفس المعلومة في حالة النظام الثنائي، وكلما تطلب حفظ المعلومة في عناصر حفظ عددها أقل من عدد عناصر الحفظ المطلوبة لحفظ نفس المعلومة في النظام الثنائي.

إن الذي يهمنا في هذه الدراسة هو النظام الثلاثي والذي قد أشبعه الباحثون دراسة وتحليلاً وتطبيقاً منذ ستينات القرن العشرين [1-7] ومن زوايا مختلفة. أما دراستنا للنظام المنطقي الثلاثي ستكون من زاوية أخرى تقوم على أساس لغة الـVHDL بهدف بناء مكتبة برامجية لتستخدم في تصميم ومحاكاة النظم المنطقية الإلكترونية الثلاثية، وبحسب علم الباحثين فإنه لم يتطرق لمثل هذه الدراسة من قبل حتى الآن لاسيما وأن لغة الـVHDL صممت أساساً للتعامل مع المنطق الثنائي وليس مع المنطق الثلاثي.

سيتم تقسيم هذه الدراسة إلى عدة مراحل وسيتم عرض نتائج كل مرحلة في ورقة علمية مستقلة، وباكتمال كل مراحل الدراسة يكون قد تم إنشاء أداة برامجية على أساس لغة الـ VHDL لتستخدم في التصميم والمحاكاة للدوائر والنظم المنطقية الإلكترونية الثلاثية.

تمثل هذه الورقة العلمية المرحلة الأولى من الدراسة والتي خصصت للبوابات المنطقية الثلاثية والتي قسمت لتحتوي على البنود التالية: البند (2) البوابات المنطقية الأساسية في النظام الثلاثي، البند (3) البوابات المنطقية الثلاثية من تقنية الـMOS، البند (4) تنفيذ البوابات المنطقية الثلاثية بلغة الـ VHDL، البند (5) تطبيق، البند (6) الخاتمة.

2.البوابات المنطقية الأساسية في النظام الثلاثي:

يوجد للنظام الثنائي مستويان للجهد يناظرهما مستويان منطقيان هما المستوى المنطقي "0" والمستوى المنطقي "1"، وكان لهذا النظام جبر المنطق الثنائي وقامت على أساس هذا النظام صناعة القطع المنطقية الإلكترونية المختلفة مثل البوابات المنطقية، والنطاطات، والمداولات الرقمية.....الخ، واستخدمت في إنتاج مختلف أنواع الأجهزة والنظم المنطقية الإلكترونية (تسمى أيضاً الأجهزة والنظم الرقمية) البسيطة والمعقدة مثل المعالج الدقيق (Microprocessor). أما النظام الثلاثي فله ثلاثة مستويات للجهد يناظرهم ثلاثة مستويات منطقية هي: المستوى المنطقي "0" والمستوى المنطقي "1" والمستوى المنطقي "2"، وتعرف بمستوى الجهد المنخفض (Vcc-) ومستوى الجهد المتوسط (0 V) ومستوى الجهد المرتفع (Vcc+) على الترتيب.

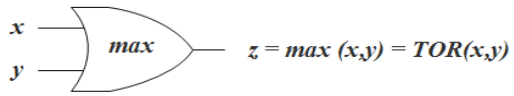
وكما أن للمنطق الثنائي ثلاث عمليات أساسية أو ثلاث بوابات منطقية أساسية هي: OR وAND وNOT، فإن للمنطق الثلاثي ثلاث عمليات أساسية أو ثلاث بوابات منطقية أساسية وعلى النحو التالي:

(1) البوابة TOR (Ternary OR Gate)

وتعرف أيضاً بالبوابة max (Maximum Gate)

$$\max(x , y) = \text{TOR}(x , y) = x + y$$

ويوضح شكل (1) الرمز الهندسي لهذه البوابة، ويكون جدول التشغيل للبوابة TOR كما هو موضح في جدول (1).



شكل (1): الرمز الهندسي للبوابة TOR

جدول (1): التشغيل للبوابة TOR

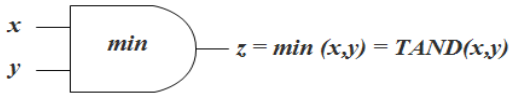
x \ y	0	1	2
0	0	1	2
1	1	1	2
2	2	2	2

(2) البوابة TAND (Ternary AND Gate)

وتعرف أيضاً بالبوابة min (Minimum Gate)

$$\min(x,y) = TAND(x,y) = x.y$$

ويوضح شكل (2) الرمز الهندسي لهذه البوابة، ويكون جدول التشغيل للبوابة TAND كما هو موضح في جدول (2).



شكل (2):الرمز الهندسي للبوابة TAND

جدول(2): التشغيل للبوابة TAND

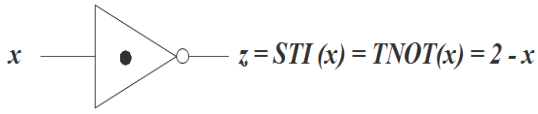
x \ y	0	1	2
0	0	0	0
1	0	1	1
2	0	1	2

(3) البوابة TNOT (Ternary NOT Gate)

وتعرف كذلك بالـ STI (Standard Ternary Inverter)

$$STI(x) = TNOT(x) = 2-x$$

ويوضح شكل (3) الرمز الهندسي لهذه البوابة، ويكون جدول التشغيل للبوابة TNOT كما هو موضح في جدول (3).



شكل (3):الرمز الهندسي للبوابة TNOT

جدول(3):جدول التشغيل للبوابة TNOT

x	z
0	2
1	1
2	0

3.البوابات المنطقية الثلاثية من تقنية الـ MOS:

بينت الورقة العلمية [10] كيفية الحصول على البوابات المنطقية TNOR وTNAND وTNOT على أساس تقنية الـ MOS، ولن نتعرض للدوائر الإلكتروني الرقمية للبوابات ولكن سيتم التعامل مع الرموز الهندسية للبوابات ومسمياتها وجداول التشغيل وعلى النحو التالي:

أولاً: البوابات TNOT

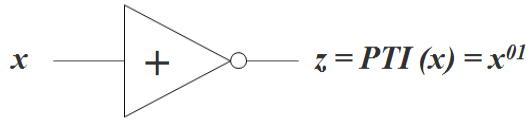
وهذه البوابة على ثلاثة أنواع وعلى النحو التالي:

(1) STI (Standard Ternary Inverter) أو STNOT

وهي البوابة الموضحة في شكل (3) السابق والمبين جدول التشغيل لها في جدول (3).

(1) PTI (Positive Ternary Inverter) أو PTNOT

ويوضح شكل (4) الرمز الهندسي للبوابة PTI، ويكون جدول التشغيل للبوابة كما هو موضح في جدول (4).



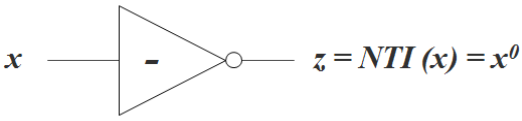
شكل (4):الرمز الهندسي للبوابة PTI

جدول(4): التشغيل للبوابة PTI

x	z
0	2
1	2
2	0

(2) NTI (Negative Ternary Inverter) أو NTNOT

ويوضح شكل (5) الرمز الهندسي للبوابة NTI، ويكون جدول التشغيل للبوابة كما هو موضح في جدول (5).



شكل (5):الرمز الهندسي للبوابة NTI

جدول(5):التشغيل للبوابة NTI

x	z
0	2
1	0
2	0

https://doi.org/10.20428/JST.22.2.2

ثانياً: البوابات TNOR

وهي البوابات المكونة من تطبيق البوابة TNOT على خرج البوابة TOR وبحسب المعادلة التالية:

$$TNOR(x,y) = TNOT\,[TOR(x,y)] = TNOT\,[\max(x,y)]$$

وحيث أن هناك ثلاثة أنواع من البوابة TNOT، فسيكون هناك ثلاثة أنواع من البوابة TNOR وعلى النحو التالي:

(1) STNOR (Standard TNOR)

ويوضح شكل (6) الرمز الهندسي للبوابة STNOR مع المعادلة الخاصة بها.

شكل (6): الرمز الهندسي للبوابة STNOR

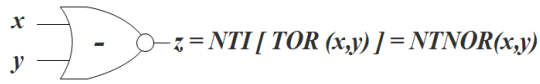
(2) PTNOR (Positive TNOR)

ويوضح شكل (7) الرمز الهندسي للبوابة PTNOR مع المعادلة الخاصة بها.

شكل (7): الرمز الهندسي للبوابة PTNOR

(3) NTNOR (Negative TNOR)

ويوضح شكل (8) الرمز الهندسي للبوابة NTNOR مع المعادلة الخاصة بها.



شكل (8): الرمز الهندسي للبوابة NTNOR

أما جدول التشغيل لهذه البوابات فيمكن تلخيصه كما في جدول (6).

جدول(5): جدول التشغيل للبوابات STNOR وPTNOR وNTNOR

x	y	STNOR	PTNOR	NTNOR
0	0	2	2	2
0	1	1	2	0
0	2	0	0	0
1	0	1	2	0
1	1	1	2	0
1	2	0	0	0
2	0	0	0	0
2	1	0	0	0
2	2	0	0	0

ثالثاً: البوابات TNAND

وهي البوابات المكونة من تطبيق البوابة TNOT على خرج البوابة TAND وبحسب المعادلة التالية:

$$TNAND\ (x,y) = TNOT[TAND\ (x,y)] = TNOT\ [\min(x,y)]$$

وحيث أن هناك ثلاثة أنواع من البوابة TNOT، فسيكون هناك ثلاثة أنواع من البوابة TNAND أيضاً وعلى النحو التالي:

(1) STNAND (Standard TNAND)

ويوضح شكل (9) الرمز الهندسي للبوابة STNAND مع المعادلة الخاصة بها.

شكل (9):الرمز الهندسي للبوابة STNAND

(2) PTNAND (Positive TNAND)

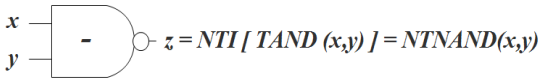
ويوضح شكل (10) الرمز الهندسي للبوابة PTNAND مع المعادلة الخاصة بها.

شكل (10):الرمز الهندسي للبوابة PTNAND

https://doi.org/10.20428/JST.22.2.2

(3) NTNAND (Negative TNAND)

ويوضح شكل (11) الرمز الهندسي للبوابة NTNAND مع المعادلة الخاصة بها.



شكل (11):الرمز الهندسي للبوابة NTNAND

أما جدول التشغيل لهذه البوابات فيمكن تلخيصه كما في جدول (7).

جدول(5): جدول التشغيل للبوابات STNAND وPTNAND وNTNAND

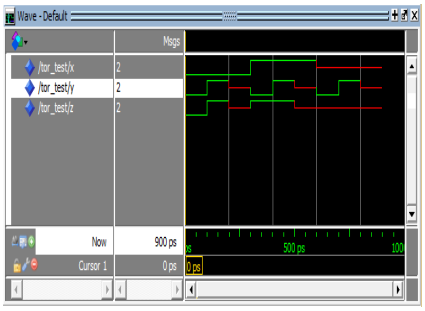
x	y	STNAND	PTNAND	NTNAND
0	0	2	2	2
0	1	2	2	2
0	2	2	2	2
1	0	2	2	2
1	1	1	2	0
1	2	1	2	0
2	0	2	2	2
2	1	1	2	0
2	2	0	0	0

4.تنفيذ البوابات المنطقية الثلاثية بلغة الـ VHDL:

سيتم في هذا البند كتابة برنامج بلغة الـ VHDL لاختبار عمل كل بوابة من البوابات التي عرضت في البندين 2 و3، ومرفق مع برنامج الـ VHDL لكل بوابة نتائج التنفيذ على برنامج المحاكاة Modelsim من شركة Mentor Graphics والذي يعرض نتائج المحاكاة على شكل موجات Waveforms وهي من أسهل الطرق لاستخلاص وإدراك النتائج. أما المكتبة التي تحتوي على تعريف عمل البوابات فتم إلحاقها في الملحق المرفق لهذه الورقة.

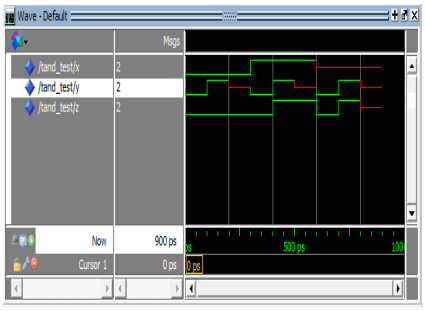
(1) البوابة TOR

```
1   USE WORK.ternary_type.ALL;
2
3   entity TOR_TEST is
4       PORT (x,y : IN  t_logic;
5             z   : OUT t_logic);
6   end entity;
7
8   architecture arc of TOR_TEST is
9   begin
10            z <= tor (x,y);
11  end arc;
```



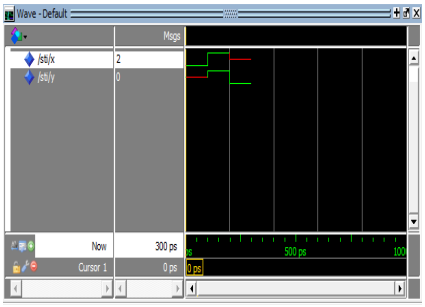
(2)البوابة TAND

```
1   USE WORK.ternary_type.ALL;
2
3   entity TAND_TEST is
4      PORT (x,y : IN  t_logic;
5            z   : OUT t_logic);
6   end entity;
7
8   architecture arc of TAND_TEST is
9   begin
10           z <= tand (x,y);
11  end arc;
```



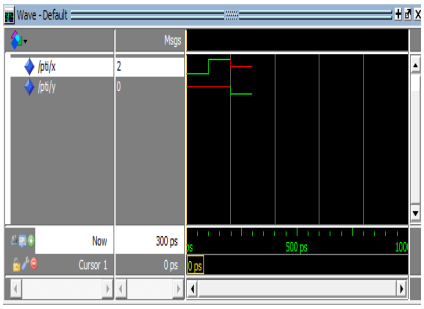
(3)البوابة STI

```
1   USE WORK.ternary_type.ALL;
2
3   entity STI is
4      PORT (x : IN  t_logic;
5            y : OUT t_logic);
6   end entity;
7
8   architecture arc of STI is
9   begin
10           y <= stnot (x);
11  end arc;
```



(4)البوابة PTI

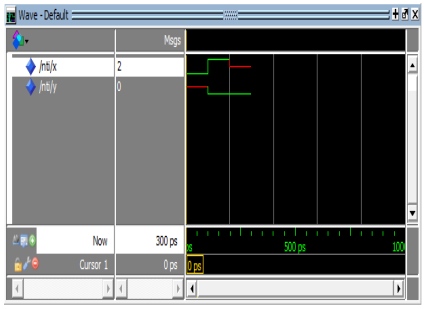
```
1   USE WORK.ternary_type.ALL;
2
3   entity PTI is
4      PORT (x : IN  t_logic;
5            y : OUT t_logic);
6   end entity;
7
8   architecture arc of PTI is
9   begin
10           y <= ptnot (x);
11  end arc;
```



https://doi.org/10.20428/JST.22.2.2

(5)البوابة NTI

```
1   USE WORK.ternary_type.ALL;
2
3   entity NTI is
4      PORT (x : IN  t_logic;
5            y : OUT t_logic);
6   end entity;
7
8   architecture arc of NTI is
9   begin
10           y <= ntnot (x);
11  end arc;
```



(6)البوابة STNOR

```
1   USE WORK.ternary_type.ALL;
2
3   entity STNOR_TEST is
4      PORT (x,y : IN  t_logic;
5            z   : OUT t_logic);
6   end entity;
7
8   architecture arc of STNOR_TEST is
9   begin
10           z <= stnor (x,y);
11  end arc;
```

(7)البوابة PTNOR

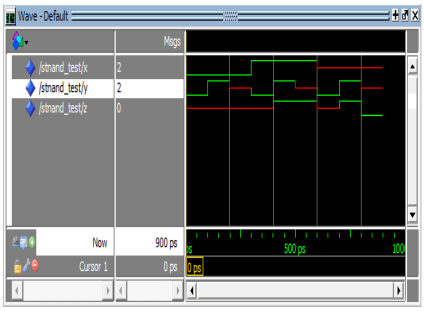
```
1   USE WORK.ternary_type.ALL;
2
3   entity PTNOR_TEST is
4      PORT (x,y : IN  t_logic;
5            z   : OUT t_logic);
6   end entity;
7
8   architecture arc of PTNOR_TEST is
9   begin
10           z <= ptnor (x,y);
11  end arc;
```

(8)البوابة NTNOR

```
1   USE WORK.ternary_type.ALL;
2
3   entity NTNOR_TEST is
4      PORT (x,y : IN  t_logic;
5            z   : OUT t_logic);
6   end entity;
7
8   architecture arc of NTNOR_TEST is
9   begin
10           z <= ntnor (x,y);
11  end arc;
```

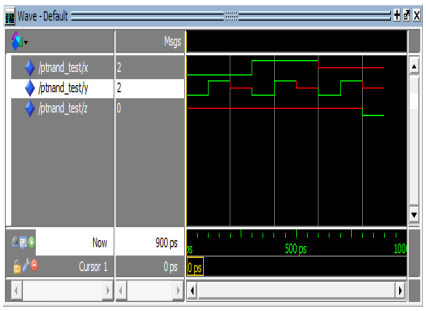
(9)البوابة STNAND

```
1   USE WORK.ternary_type.ALL;
2
3   entity STNAND_TEST is
4      PORT (x,y : IN  t_logic;
5            z   : OUT t_logic);
6   end entity;
7
8   architecture arc of STNAND_TEST is
9   begin
10           z <= stnand (x,y);
11  end arc;
```



(10)البوابة PTNAND

```
1   USE WORK.ternary_type.ALL;
2
3   entity PTNAND_TEST is
4      PORT (x,y : IN  t_logic;
5            z   : OUT t_logic);
6   end entity;
7
8   architecture arc of PTNAND_TEST is
9   begin
10           z <= ptnand (x,y);
11  end arc;
```



https://doi.org/10.20428/JST.22.2.2

(11)البوابة NTNAND

```
1   USE WORK.ternary_type.ALL;
2
3   entity NTNAND_TEST is
4      PORT (x,y : IN  t_logic;
5            z   : OUT t_logic);
6   end entity;
7
8   architecture arc of NTNAND_TEST is
9   begin
10           z <= ntnand (x,y);
11  end arc;
```

5.تطبيق:

سيتم في هذا البند عرض تطبيق لكيفية استخدام مكتبة البوابات الثلاثية في تحقيق الدوائر المنطقية الثلاثية من خلال تصميم الدائرة المنطقية المحققة للدالة :

$$F(A,B)=1 \quad at \ \sum(0,1,3)$$

$$=2 \quad at \ \sum(4,5,7,8)$$

وسيتم بناء الدائرة المحققة للدالة F(A,B) أولاً على أساس البوابات الأساسية TOR وTAND والبوابات العاكسة STI وPTI وNTI ، ثم ثانياً على أساس البوابات STNAND والبوابات العاكسة، والخطوة الأولى تكون بوضع التعبير الجبري للدالة F(A,B) في أبسط صورة بإتباع الطريقة الجبرية [1] أو الطريقة التصويرية [1] فنحصل على:

$$F(A,B)=A^{12}B^{12}+1.A^{01}B^{01}$$

أولاً: على أساس البوابات الأساسية TOR وTAND والبوابات العاكسة

يمكن كتابة الدالة F(A،B) بالصورة التالية :

$$F(A,B)=TOR\left[A^{12}B^{12},1.A^{01}B^{01}\right]$$

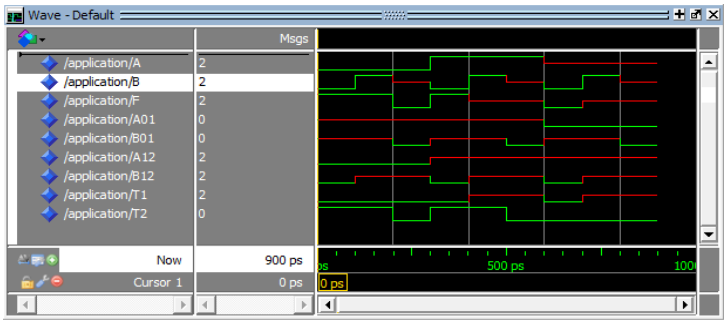
$$F(A,B)=TOR\left[TAND\ (A^{12},B^{12}) \ ,TAND\ (1,TAND\ (A^{01},B^{01}))\right]$$

وبرنامج الـ VHDL للدالة F(A,B) يكون كما هو موضح في شكل (12).

```
1   USE WORK.ternary_type.ALL;
2
3   entity Application is
4      PORT (A,B : IN  t_logic;
5            F   : OUT t_logic);
6   end entity;
7
8   architecture arc of Application is
9   signal A01, B01, A12, B12, T1, T2: t_logic;
10  begin
11           A01 <= ptnot(A);
12           B01 <= ptnot(B);
13           A12 <= ptnot(stnot (A));
14           B12 <= ptnot(stnot (B));
15           T1  <= tand (A12,B12);
16           T2  <= tand ('1',tand (A01,B01));
17           F   <= tor  (T1,T2);
18  end arc;
```

شكل (12): برنامج الـ VHDL لتحقيق الدالة F(A,B) على أساس البوابات الأساسية TOR وTAND والبوابات العاكسة

ويوضح شكل (13) ناتج المحاكاة للدائرة المحققة للدالة F(A,B) على أساس البوابات الأساسية TOR و TAND والبوابات العاكسة STI وPTI وNTI.



شكل (13): ناتج المحاكاة للدائرة المحققة للدالة F(A,B) على أساس البوابات الأساسية TOR وTAND والبوابات العاكسة

ثانياً: على أساس البوابات STNAND والبوابات العاكسة

يمكن إعادة كتابة الدالة F(A,B) بالصورة التالية:

$$F(A,B) = A^{12}\,B^{12} + 1.A^{01}\,B^{01}$$

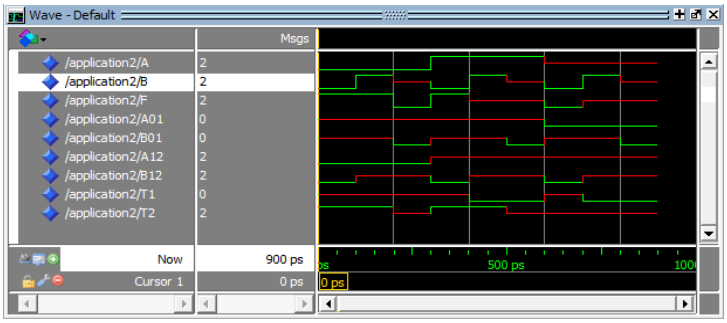
$$F(A,B) = STNAND\,[STNAND\,(A^{12},B^{12}),\,STNAND\,(1,A^{01},B^{01})]$$

وبرنامج الـ VHDL للدالة F(A,B) يكون كما هو موضح في شكل (14).

```
1   USE WORK.ternary_type.ALL;
2
3   entity Application2 is
4      PORT (A,B : IN  t_logic;
5            F   : OUT t_logic);
6   end entity;
7
8   architecture arc of Application2 is
9   signal A01, B01, A12, B12, T1, T2: t_logic;
10  begin
11          A01 <= ptnot(A);
12          B01 <= ptnot(B);
13          A12 <= ptnot(stnot (A));
14          B12 <= ptnot(stnot (B));
15          T1  <= stnand(A12,B12);
16          T2  <= stnand('1',A01,B01);
17          F   <= stnand(T1,T2);
18  end arc;
```

شكل (14):برنامج الـ VHDL لتحقيق الدالة F(A,B) على أساس البوابات STNAND والبوابات العاكسة

ويوضح شكل (15) ناتج المحاكاة للدائرة المحققة للدالة F(A,B) على أساس البوابات STNAND والبوابات العاكسة STI وPTI وNTI.



شكل (15):ناتج المحاكاة للدائرة المحققة للدالة F(A,B) على أساس البوابات الأساسية STNAND والبوابات العاكسة

https://doi.org/10.20428/JST.22.2.2

6. الخاتمة:

تم في هذه المرحلة من الدراسة المستحدثة بناء مكتبة برامجية بلغة الـ VHDL للبوابات الأساسية في المنطق الثلاثي وهي TOR أو تسمى max وTAND أو تسمى min وبوابات المنطق الثلاثي من تقنية الـ MOS وهي: TNOT بأنواعها الثلاثة والـ TNOR بأنواعها الثلاثة و الـ TNAND بأنواعها الثلاثة، ومن ثم تم استخدام هذه المكتبة في التصميم والمحاكاة للدوائر المنطقية الإلكترونية الثلاثية (Ternary Electronic Logic Circuits)**. ووضح البند** (5) **استخدام المكتبة من خلال مثال عددي كتطبيق.**

7. المراجع:

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

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

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

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

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

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

[7] 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.

[8] 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.

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

[10] 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.

ملحق:

مكتبة تعريف المنطق الثلاثي وبواباته الأساسية

```
PACKAGE ternary_type IS
        TYPE t_logic IS ('U','0','1','2');
        TYPE t_logic_vector is array (natural range <>) of t_logic;

        function stnot  (l: t_logic) return t_logic;
        function ptnot  (l: t_logic) return t_logic;
        function ntnot  (l: t_logic) return t_logic;
        FUNCTION tor    (l,r: t_logic) RETURN t_logic;
        FUNCTION tand   (l,r: t_logic) RETURN t_logic;
        FUNCTION stnor  (l,r: t_logic) RETURN t_logic;
        FUNCTION ptnor  (l,r: t_logic) RETURN t_logic;
        FUNCTION ntnor  (l,r: t_logic) RETURN t_logic;
        FUNCTION stnand (l,r: t_logic) RETURN t_logic;
FUNCTION stnand (l,r,k: t_logic) RETURN t_logic;
        FUNCTION ptnand (l,r: t_logic) RETURN t_logic;
        FUNCTION ntnand (l,r: t_logic) RETURN t_logic;
END ternary_type;

PACKAGE BODY ternary_type is
        type t_logic_ld is array (t_logic) of t_logic;
        type t_logic_table is array (t_logic, t_logic) of t_logic;

CONSTANT stnot_table: t_logic_ld :=
   --  ------------------------
   --  |   U    0    1    2    |
   --  ------------------------
       ( 'U', '2', '1', '0');
```


https://doi.org/10.20428/JST.22.2.2

```
CONSTANT ptnot_table: t_logic_ld :=
   --  ------------------------
   --  |   U    0    1    2    |
   --  ------------------------
       ( 'U', '2', '2', '0');

CONSTANT ntnot_table: t_logic_ld:=
   --  ------------------------
   --  |   U    0    1    2    |
   --  ------------------------
       ( 'U', '2', '0', '0');

CONSTANT tor_table: t_logic_table:= (
   -- ------------------------
   -- |  U    0    1    2  |
   -- ----------------------
     ( 'U', 'U', 'U', 'U' ),  -- | U |
     ( 'U', '0', '1', '2' ),  -- | 0 |
     ( 'U', '1', '1', '2' ),  -- | 1 |
     ( 'U', '2', '2', '2' ));  -- | 2 |

CONSTANT tand_table: t_logic_table := (
   -- ------------------------
   -- |  U    0    1    2  |
   -- ----------------------
     ( 'U', 'U', 'U', 'U' ),  -- | U |
     ( 'U', '0', '0', '0' ),  -- | 0 |
     ( 'U', '0', '1', '1' ),  -- | 1 |
     ( 'U', '0', '1', '2' ));  -- | 2 |
```

```
-----------------------------------------------
--Standard Ternary Not
-----------------------------------------------
FUNCTION stnot (l: t_logic)    RETURN t_logic is
BEGIN
RETURN (stnot_table(l));
END stnot;
-----------------------------------------------------------
-- TOR
-----------------------------------------------------------
FUNCTION tor ( l,r :t_logic) RETURN t_logic IS
BEGIN
RETURN (tor_table(l,r)) ;
END tor;
-----------------------------------------------------------
-- TAND
-----------------------------------------------------------
FUNCTION tand ( l,r :t_logic) RETURN t_logic IS
BEGIN
RETURN (tand_table(l,r)) ;
END tand;
-----------------------------------------------
--Positive Ternary Not
-----------------------------------------------
FUNCTION ptnot (l: t_logic) RETURN t_logic is
BEGIN
RETURN (ptnot_table(l));
END ptnot;
-----------------------------------------------
```


https://doi.org/10.20428/JST.22.2.2

```
--Negative Ternary Not
-------------------------------------------
FUNCTION ntnot (l: t_logic) RETURN t_logic is
BEGIN
RETURN (ntnot_table(l));
END ntnot;
-----------------------------------------------------------
-- Standard TNOR
-----------------------------------------------------------
FUNCTION stnor ( l,r :t_logic) RETURN t_logic IS
BEGIN
RETURN stnot(tor_table(l,r)) ;
END stnor;
-----------------------------------------------------------
-- Positive TNOR
-----------------------------------------------------------
FUNCTION ptnor ( l,r :t_logic) RETURN t_logic IS
BEGIN
RETURN ptnot(tor_table(l,r)) ;
END ptnor;
-----------------------------------------------------------
-- Negative TNOR
-----------------------------------------------------------
FUNCTION ntnor ( l,r :t_logic) RETURN t_logic IS
BEGIN
RETURN ntnot(tor_table(l,r)) ;
END ntnor;
-----------------------------------------------------------
-- Standard TNAND
```

```
------------------------------------------------------------
FUNCTION stnand ( l,r :t_logic) RETURN t_logic IS
BEGIN
RETURN stnot(tand_table(l,r)) ;
END stnand;

FUNCTION stnand ( l,r,k :t_logic) RETURN t_logic IS
BEGIN
RETURN stnot(tand_table(l,tand_table(r,k)));
END stnand;
------------------------------------------------------------
-- Positive TNAND
------------------------------------------------------------
FUNCTION ptnand ( l,r :t_logic) RETURN t_logic IS
BEGIN
RETURN ptnot(tand_table(l,r)) ;
END ptnand;
------------------------------------------------------------
-- Negative TNAND
------------------------------------------------------------
FUNCTION ntnand ( l,r :t_logic) RETURN t_logic IS
BEGIN
RETURN ntnot(tand_table(l,r));
END ntnand;
------------------------------------------------------------
END ternary_type;
```


https://doi.org/10.20428/JST.22.2.2