أتمتة النظم المنطقية الإلكترونية الثلاثية: دراسة مستحدثة مؤسسة على لغة VHDL
المرحلة الثانية: القطع المنطقية التجميعية الثلاثية

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

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

https://doi.org/10.20428/JST.23.2.3

أتمتة النظم المنطقية الإلكترونية الثلاثية: دراسة مستحدثة مؤسسة على لغة VHDL–المرحلة الثانية: القطع المنطقية التجميعية الثلاثية

الملخص:

سيتم في هذه الورقة العلمية بناء المرحلة الثانية من المكتبة البرمجية المؤسسة على لغة الـVHDL للقطع التجميعية الثلاثية الأساسية بدءاً بالقطعة TXOR (Ternary XOR gate) وختاماً بالقطعة TPA (Ternary Parallel Adder)، وهذه المرحلة الثانية امتداداً لمكتبة المرحلة الأولى من الدراسة والخاصة بالبوابات المنطقية الثلاثية الأساسية [1].

الكلمات المفتاحية : المنطق الثلاثي، القطع التجميعية الثلاثية، لغة الـVHDL.

Ternary Electronic Logic Systems Automation: A Novel Study Based on VHDL Language–Second Part: Ternary Combination Logic Components

Abstract:

In this paper, the second part of the software library for the Ternary combinational logic components will be built based on VHDL language starting by the TXOR (Ternary XOR gate) and ending by the TPA (Ternary Parallel Adder). This second part is an extension to the library given in the first part of the study which was about the basic Ternary Logic Gates [1].

Keywords: Ternary logic, Ternary combinational logic components, VHDL language.

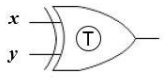
https://doi.org/10.20428/JST.23.2.3

1. المقدمة:

خصصت المرحلة الأولى من الدراسة للبوابات المنطقية الثلاثية [1]، وأشير في [1] إلى أن الدراسة ستكون على عدة مراحل وأن نتائج كل مرحلة ستعرض في ورقة علمية، وعليه فستكون هذه الورقة العلمية الثانية لنتائج المرحلة الثانية من الدراسة وذلك للقطع المنطقية التالية: TXOR – TDEC – TENC – TMUX – TDMUX – OTC – THA – TFA – OTM – TPA. نظمت هذه الورقة على النحو التالي: البند (2) للبوابة TXOR، البند (3) للقطعة TDEC، البند (4) للقطعة TENC، البند (5) للقطعة TMUX، البند (6) للقطعة TDMUX، البند (7) للقطعة OTC، البند (8) للقطعة THA، البند (9) للقطعة TFA، البند (10) للقطعة OTM، البند (11) للقطعة TPA، البند (12) لتنفيذ القطع السابقة بلغة الـ VHDL، البند (13) للخاتمة، ثم المراجع ثم الملحق المتضمن التصميم بلغة الـ VHDL للقطع سالفة الذكر.

2. البوابة TXOR [12]:

تعتبر هذه البوابة مكملة لمكتبة البوابات المنطقية الثلاثية والتي قدمت في المرحلة الأولى من الدراسة [1]، ورحلت هذه البوابة للمرحلة الثانية كونها معقدة ويدخل في تكوينها عدد من بوابات المرحلة الأولى، ويوضح شكل (1) الرمز الهندسي وجدول التشغيل للبوابة TXOR.



$z = TXOR(x, y)$

$x \text{Ⓣ} x = 2.x$

$x \text{Ⓣ} x \text{Ⓣ} x = 0$

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

شكل (1): الرمز الهندسي وجدول التشغيل للبوابة TXOR

أما الوصف اللغوي لعمل هذه البوابة يكون في الصورة التالية:

```
Tcomp: TXOR(2:1)
Inputs: x, y
Outputs: z
Begin
    Case x of
    Begin
      0: case y of
        Begin
          0: z=0
          1: z=1
          2: z=2
        End
      1: case y of
        Begin
          0: z=1
```

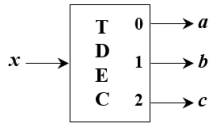
```
                1: z=2
                2: z=0
            End
        2: case y of
            Begin
                0: z=2
                1: z=0
                2: z=1
            End
    End
End
```

3.القطعة TDEC [2]:

الـ TDEC من العبارة Ternary Decoder، **وهذه القطعة على مقاسين:**

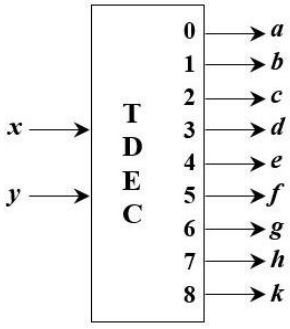
1. TDEC 1 **إلى** 3.
2. TDEC 2 **إلى** 9.

يوضح شكل (2) **الرمز الهندسي وجدول التشغيل للـ** TDEC **بالمقاس** 1 **إلى** 3، **بينما يوضح شكل** (3) **الرمز الهندسي وجدول التشغيل لـ** TDEC **بالمقاس** 2 **إلى** 9.



x	a	b	c
0	2	0	0
1	0	2	0
2	0	0	2

الشكل(2): الرمز الهندسي وجدول التشغيل لـ TDEC بالمقاس 1 إلى 3



x	y	a	b	c	d	e	f	g	h	k
0	0	2	0	0	0	0	0	0	0	0
0	1	0	2	0	0	0	0	0	0	0
0	2	0	0	2	0	0	0	0	0	0
1	0	0	0	0	2	0	0	0	0	0
1	1	0	0	0	0	2	0	0	0	0
1	2	0	0	0	0	0	2	0	0	0
2	0	0	0	0	0	0	0	2	0	0
2	1	0	0	0	0	0	0	0	2	0
2	2	0	0	0	0	0	0	0	0	2

الشكل (3): الرمز الهندسي وجدول التشغيل لـ TDEC بالمقاس 2 إلى 9

أما الوصف اللغوي لـ TDEC بالمقاس 1 إلى 3 يكون في الصورة التالية :

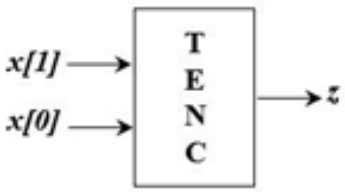
```
Tcomp: TDEC(1:3)
Inputs: x
Outputs: a, b, c
Begin
    if x= 0,    Then {a = 2, b = 0, c = 0}
    else if x = 1,   Then {a = 0, b = 2, c = 0}
    else if x = 2,   Then {a = 0, b = 0, c = 2}
      End
```

أو في الصورة التالية :

```
Tcomp: TDEC (1:3)
Inputs: x
Outputs: a, b, c
Begin
    Case x of
    Begin
      0: a = 2, b = 0, c = 0
      1: a = 0, b = 2, c = 0
      2: a = 0, b = 0, c = 2
    End
End
```

4. القطعة TENC [2]:

الـ TENC من العبارة Ternary Encoder، وهذه القطعة بالمقاس 2 إلى 1، ويوضح شكل (4) الرمز الهندسي وجدول التشغيل لـ TENC.



x[1]	x[0]	z
0	0	1
0	2	2
2	0	0
2	2	1

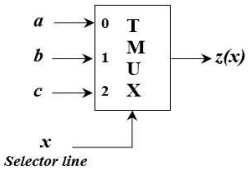
الشكل(4): الرمز الهندسي وجدول التشغيل لـ TENC

أما الوصف اللغوي للـ TENC يكون في الصورة التالية:

```
Tcomp: TENC (2:1)
Inputs: x [2]
Outputs: z
Begin
    if x = 00,  Then z = 1
    if x = 02,  Then z = 2
    if x = 20,  Then z = 0
    if x = 22,  Then z = 1
End
```

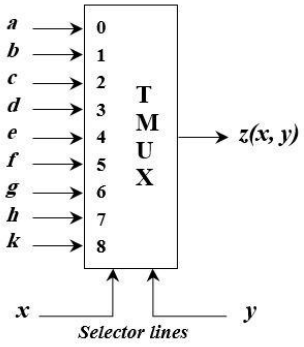
5. القطعة TMUX [2]:

الـ TMUX من العبارة Ternary Multiplexer، وتأتي هذه القطعة بالمقاس 3 إلى 1 والمقاس 9 إلى 1، ويوضح شكل (5) الرمز الهندسي وجدول التشغيل للـ TMUX بالمقاس 3 إلى 1، أما شكل (6) يوضح الرمز الهندسي وجدول التشغيل للـ TMUX بالمقاس 9 إلى 1، حيث Inputs $\in \{0,1,2\}$.



x	z
0	a
1	b
2	c

الشكل (5): الرمز الهندسي وجدول التشغيل للـ TMUX بالمقاس 3 إلى 1



x	y	z
0	0	a
0	1	b
0	2	c
1	0	d
1	1	e
1	2	f
2	0	g
2	1	h
2	2	k

الشكل (6): الرمز الهندسي وجدول التشغيل للـ TMUX بالمقاس 9 إلى 1

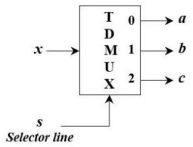
https://doi.org/10.20428/JST.23.2.3

أما الوصف اللغوي لـ TMUX بالمقاس 3 إلى 1 يكون في الصورة التالية :

```
Tcomp: TMUX(3:1)
Inputs: a, b, c, x
Outputs: z
Begin
    if x = 0, Then z = a
    if x = 1, Then z = b
    if x = 2, Then z = c
End
```

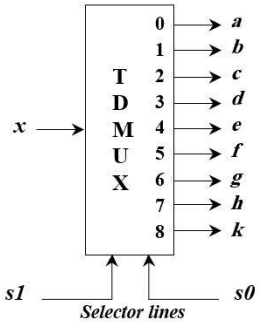
6. القطعة TDMUX:

الـ TDMUX من العبارة Ternary Demultiplexer**، وتأتي هذه القطعة بالمقاس 1 إلى 3 والمقاس 1 إلى 9، ويوضح شكل (7) الرمز الهندسي وجدول التشغيل لـ TDMUX بالمقاس 1 إلى 3، بينما يوضح شكل (8) الرمز الهندسي وجدول التشغيل لـ TMUX بالمقاس 1 إلى 9.**



s	a	b	c
0	x	0	0
1	0	x	0
2	0	0	x

الشكل (7): الرمز الهندسي وجدول التشغيل لـ TDMUX بالمقاس 1 إلى 3



s1	s0	a	b	c	d	e	f	g	h	k
0	0	x	0	0	0	0	0	0	0	0
0	1	0	x	0	0	0	0	0	0	0
0	2	0	0	x	0	0	0	0	0	0
1	0	0	0	0	x	0	0	0	0	0
1	1	0	0	0	0	x	0	0	0	0
1	2	0	0	0	0	0	x	0	0	0
2	0	0	0	0	0	0	0	x	0	0
2	1	0	0	0	0	0	0	0	x	0
2	2	0	0	0	0	0	0	0	0	x

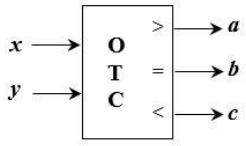
الشكل (8): الرمز الهندسي وجدول التشغيل لـ TDMUX بالمقاس 1 إلى 9

أما الوصف اللغوي لـ TDMUX بالمقاس 1 إلى 3 يكون في الصورة التالية :

```
Tcomp: TDMUX (1:3)
Inputs: x, s
Outputs: a, b, c
Begin
    if s = 0,  Then {a = x, b = 0, c = 0}
    if s = 1,  Then {a = 0, b = x, c = 0}
    if s = 2,  Then {a = 0, b = 0, c = x}
End
```

7. القطعة OTC [2]:

الـ OTC من العبارة One-Trit Comparator، ويوضح شكل (9) الرمز الهندسي وجدول التشغيل لـ OTC.



x	y	a	b	c
0	0	0	2	0
0	1	0	0	2
0	2	0	0	2
1	0	2	0	0
1	1	0	2	0
1	2	0	0	2
2	0	2	0	0
2	1	2	0	0
2	2	0	2	0

الشكل (9): الرمز الهندسي وجدول التشغيل لـ OTC

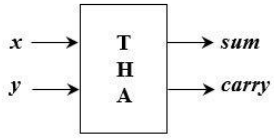
أما الوصف اللغوي لـ OTC يكون في الصورة التالية :

```
Tcomp: OTC (2:3)
Inputs: x، y
Outputs: a, b, c
Begin
   Case x of
   Begin
      0: case y of
         Begin
            0: a = 0, b = 2, c = 0
            1: a = 0, b = 0, c = 2
            2: a = 0, b = 0, c = 2
         End
```

```
        1: case y of
         Begin
            0: a = 2, b = 0, c = 0
            1: a = 0, b = 2, c = 0
            2: a = 0, b = 0, c = 2
         End
        2: case y of
         Begin
            0: a = 2, b = 0, c = 0
            1: a = 2, b = 0,c = 0
            2: a = 0, b = 2, c = 0
         End
      End
   End
```

8. القطعة THA [2]:

الـ THA من العبارة Ternary Half Adder، **ويوضح شكل** (10) **الرمز الهندسي وجدول التشغيل لـ** THA.



x	y	carry	sum
0	0	0	0
0	1	0	1
0	2	0	2
1	0	0	1
1	1	0	2
1	2	1	0
2	0	0	2
2	1	1	0
2	2	1	1

الشكل (10): الرمز الهندسي وجدول التشغيل لـ THA

أما الوصف اللغوي لـ THA **يكون في الصورة التالية:**

```
Tcomp: THA (2:2)
Inputs: x, y
Outputs: sum, carry
 Begin
    sum  = LST of x + y     / ternary addition
    carry = MST of x + y
 End
```

حيث: MST: = Most Significant Trit, LST: = Least Significant Trit

https://doi.org/10.20428/JST.23.2.3

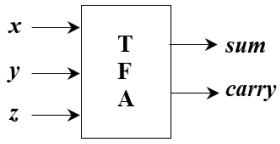
ويمكن أن يكتب الوصف اللغوي لـ THA في الصورة التالية :

```
Tcomp: THA (2:2)
Inputs: x, y
Outputs: sum، carry
Begin
    Case x of
    Begin
       0: case y of
            Begin
                0: sum = 0, carry = 0
                1: sum = 1, carry = 0
                2: sum = 2, carry = 0
            End
       1: case y of
            Begin
                0: sum = 1, carry = 0
                1: sum = 2, carry = 0
                2: sum = 0, carry = 1
            End
       2: case y of
            Begin
                0: sum = 2, carry = 0
                1: sum = 0, carry = 1
                2: sum = 1, carry = 1
            End
    End
End
```


https://doi.org/10.20428/JST.23.2.3

9. القطعة TFA [2]:

الـ TFA من العبارة Ternary Full Adder، **ويوضح شكل** (11) **الرمز الهندسي وجدول التشغيل لـ** TFA.



x	y	z	carry	sum
0	0	0	0	0
0	0	1	0	1
0	0	2	0	2
0	1	0	0	1
0	1	1	0	2
	. . .			. . .
1	1	1	1	0
	. . .			. . .
2	2	2	2	0

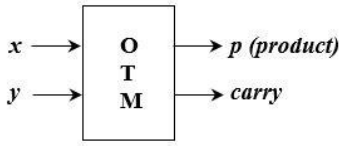
الشكل (11):الرمز الهندسي وجدول التشغيل لـ TFA

أما الوصف اللغوي لـ TFA **يكون في الصورة التالية :**

```
Tcomp: TFA (3:2)
Inputs: x, y, ci
Outputs: sum, carry
Begin
    sum   = LST of x + y + ci     / ternary addition
   carry = MST of x + y + ci
End
```

10. القطعة OTM [2]:

الـ OTM **من العبارة** One-Trit Multiplier، **ويوضح شكل** (12) **الرمز الهندسي وجدول التشغيل لـ** OTM.



x	y	carry	p
0	0	0	0
0	1	0	0
0	2	0	0
1	0	0	0
1	1	0	1
1	2	0	2
2	0	0	0
2	1	0	2
2	2	1	1

الشكل (12): الرمز الهندسي وجدول التشغيل لـ OTM

أما الوصف اللغوي لـ OTM يكون في الصورة التالية :

```
Tcomp: OTM(2,2)
Inputs: x, y
Outputs: p, carry
Begin
    Case x of
    Begin
      0: case y of
          Begin
              0: p = 0, carry = 0
              1: p = 0, carry = 0
              2: p = 0, carry = 0
          End
      1: case y of
          Begin
              0: p = 0, carry = 0
              1: p = 1, carry = 0
              2: p = 2, carry = 0
          End
      2: case y of
          Begin
              0: p = 0, carry = 0
              1: p = 2, carry = 0
              2: p = 1, carry = 1
          End
    End
End
```


https://doi.org/10.20428/JST.23.2.3

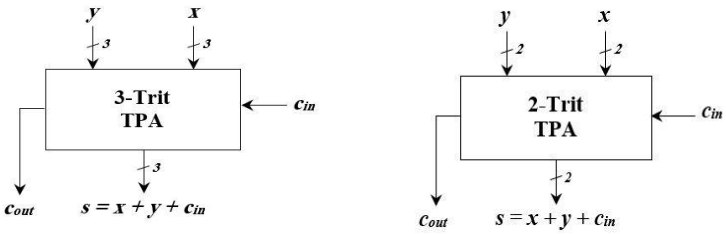
ويمكن أن يكتب الوصف اللغوي لـ OTM في الصورة التالية :

```
Tcomp: OTM (2:2)
Inputs: x[2]
Outputs: p, carry
Begin
    if x = 22,      Then {p = 1, carry = 1}
   else if x = 11, Then {p = 1, carry = 0}
   else if x = 12, Then {p = 2, carry = 0}
   else if x = 21, Then {p = 2, carry = 0}
    otherwise          {p = 0, carry = 0}
End
```

11. القطعة TPA:

الـ TPA من العبارة Ternary-Parallel Adder، ويتم الحصول على الـ TPA بربط اثنين أو أكثر من الـ TFA على التوالي كما في النظام الثنائي، ويكون مقاس الـ TPA على أساس عدد الـ TFA الموصلة على التوالي، فإذا كان عدد الـ TFA اثنان كان الناتج TPA Trit-2 وإذا كان عدد الـ TFA ثلاثة كان الناتج TPA Trit-3 وهكذا.

سيتم في هذا البند تقديم كل من الـ TPA بمقاس ثلاثيتين TPA Trit-2 والـ TPA بمقاس ثلاث ثلاثيات TPA Trit-3، ويوضح شكل (13 -أ) الرمز الهندسي للـ TPA بمقاس ثلاثيتين وشكل (13 -ب) الرمز الهندسي للـ TPA بمقاس ثلاث ثلاثيات.



(ب) TPA بمقاس ثلاث ثلاثيات (أ) TPA بمقاس ثلاثيتين

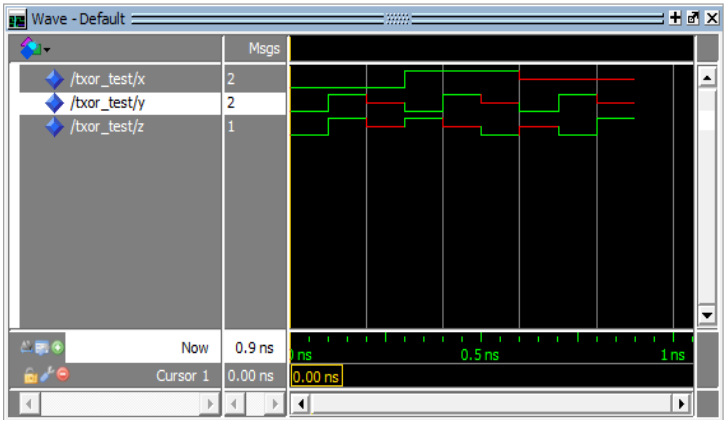
الشكل (6):الرمز الهندسي للـ TPA

أما الوصف اللغوي للـ TPA Trit-2 يكون في الصورة التالية :

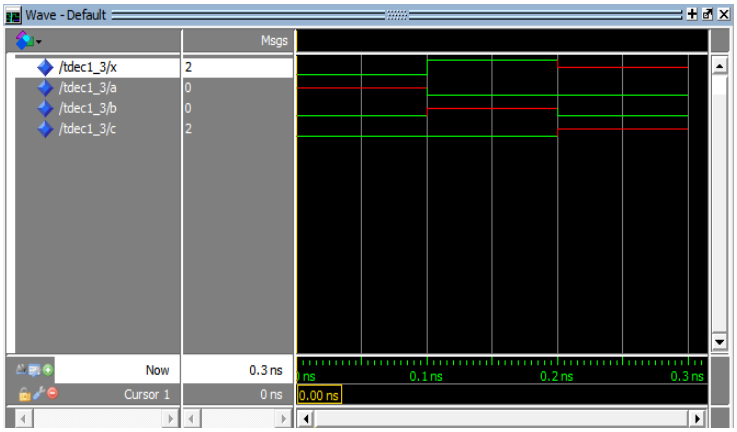
```
Tcomp: 2-Trit TPA (5:3)
Inputs:x[2], y[2], cin
Outputs: s[2], cout
Begin
    s[0] = LST of x[0] + y[0] + cin
    carry = MST of x[0] + y[0] + cin
    s[1] = LST of x[1] + y[1] + carry
    cout = MST of x[1] + y[1] + carry
End
```

12. التنفيذ بلغة الـ VHDL:

سيتم في هذا البند اختبار البرامج المكتوبة بلغة الـ VHDL لكل قطعة من القطع التي نوقشت في البنود من (2) إلى (11)، والمرفقة في ملحق هذه الورقة، وتوضح الأشكال من شكل (14) إلى شكل (26) نتائج التنفيذ للبرامج المكتوبة 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 من الدراسة [1]، وقد تم تحديثها بإضافة تعريف للبوابة TXOR.

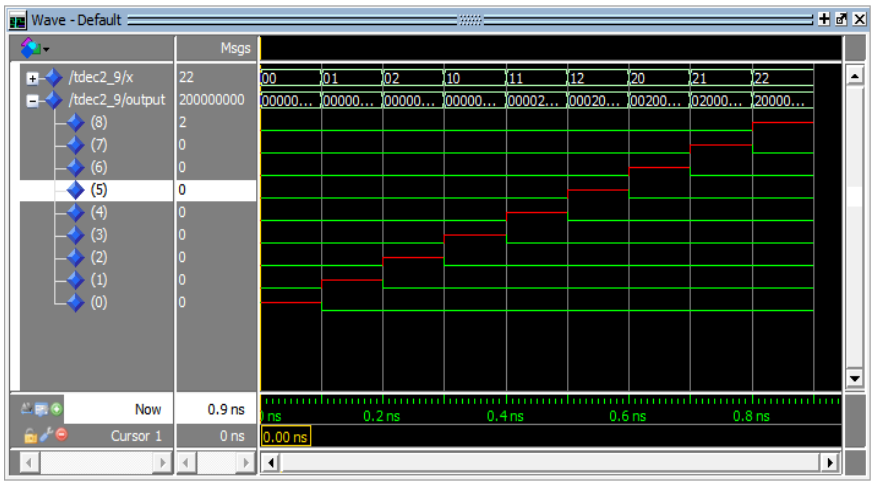


شكل(14): نتائج التنفيذ للبوابة TXOR

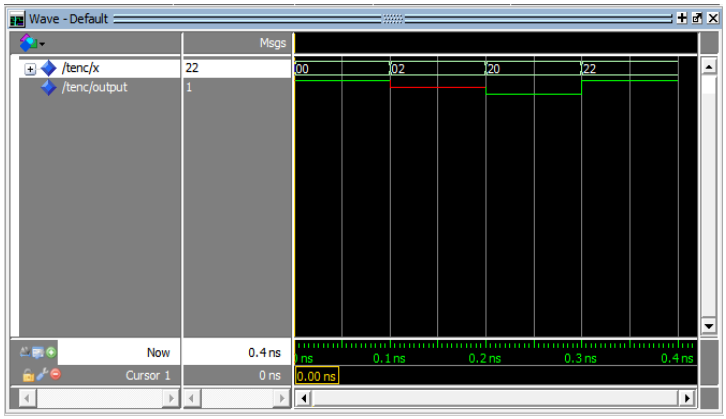


شكل(15): نتائج التنفيذ للقطعة TDEC بالمقاس 1 إلى 3

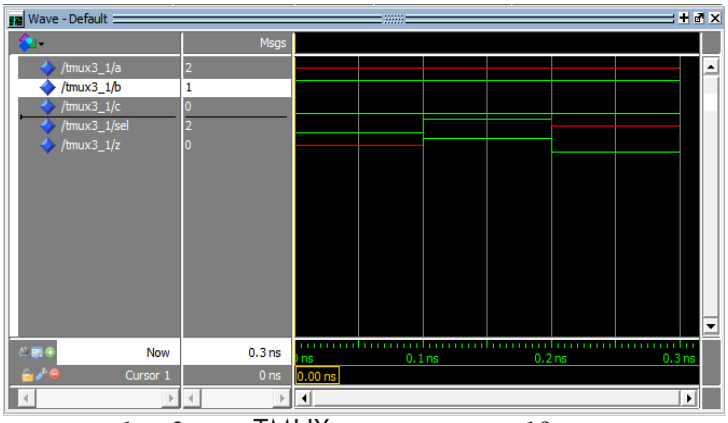
https://doi.org/10.20428/JST.23.2.3



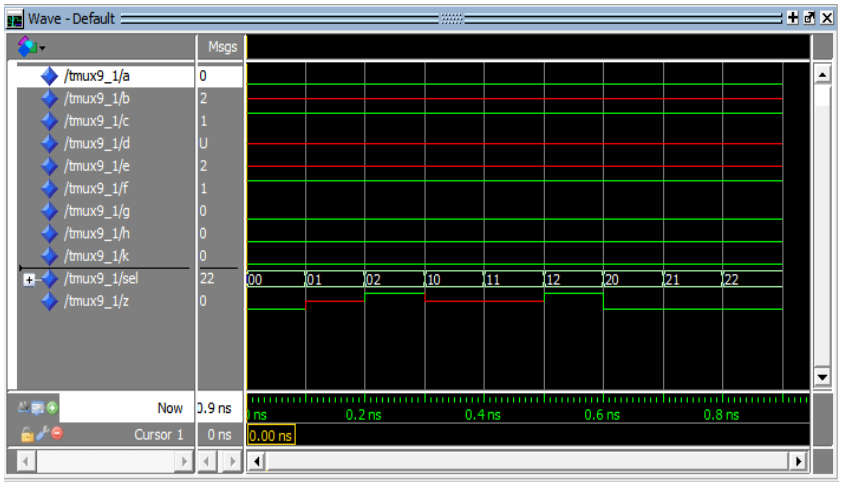
شكل(16): نتائج التنفيذ للقطعة TDEC بالمقاس 2 إلى 9



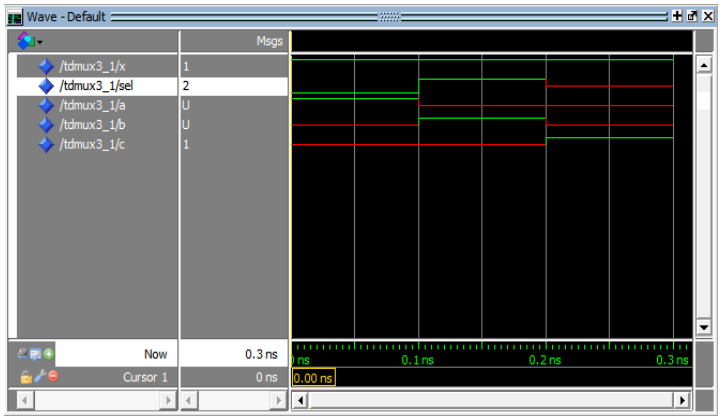
شكل(17): نتائج التنفيذ للقطعة TENC



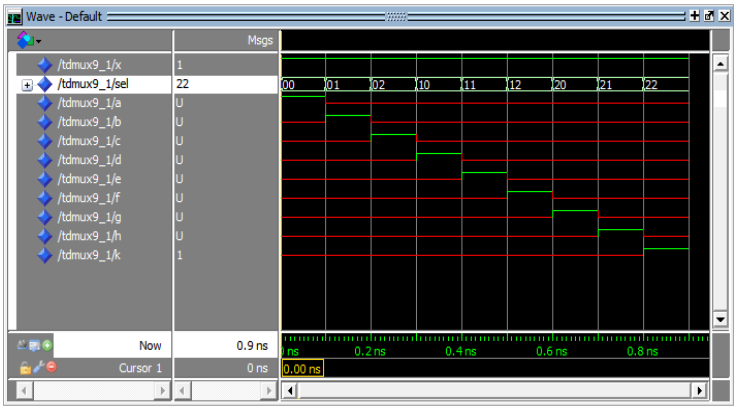
شكل(18): نتائج التنفيذ للقطعة TMUX بالمقاس 3 إلى 1



شكل(19): نتائج التنفيذ للقطعة TMUX بالمقاس 9 إلى 1

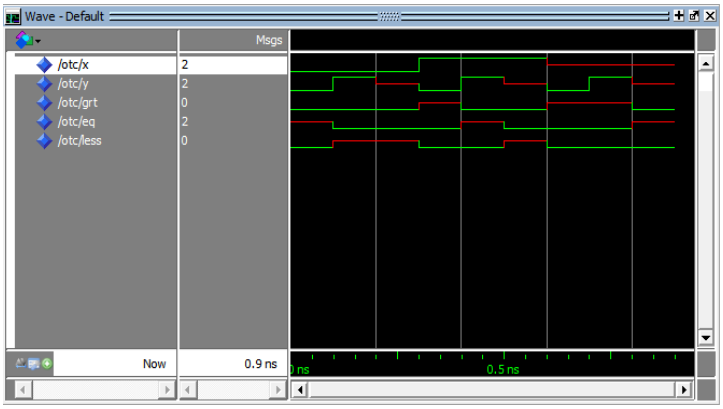


شكل(20): نتائج التنفيذ للقطعة TDMUX بالمقاس 1 إلى 3

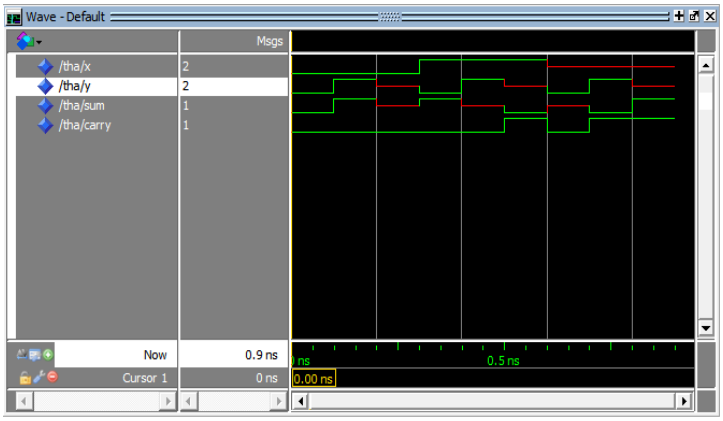


شكل(21): نتائج التنفيذ للقطعة TDMUX بالمقاس 1 إلى9

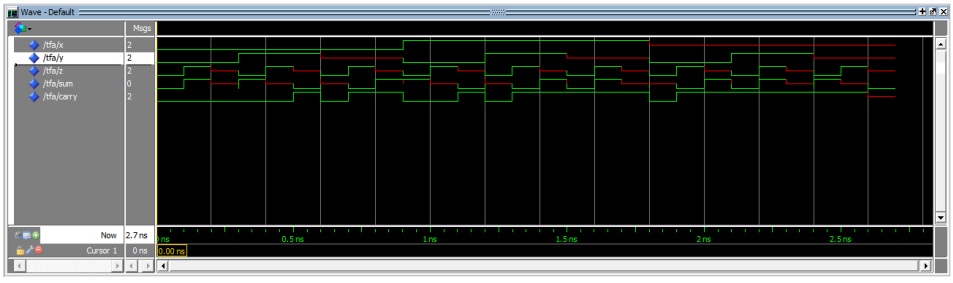
https://doi.org/10.20428/JST.23.2.3



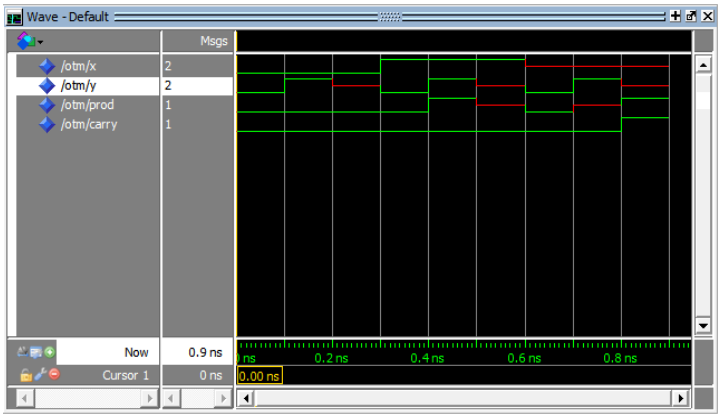
شكل(22): نتائج التنفيذ للقطعة OTC



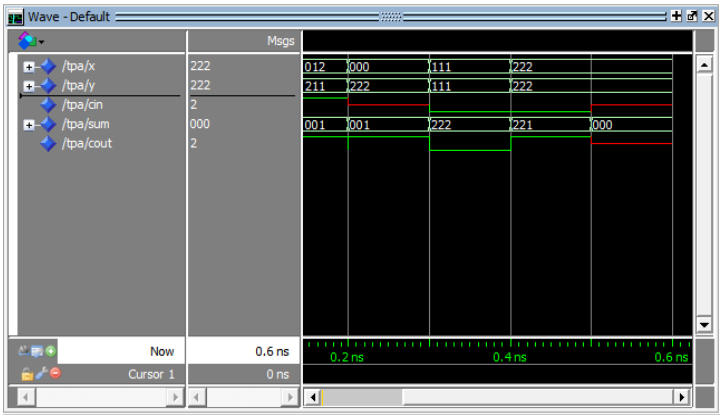
شكل(23): نتائج التنفيذ للقطعة THA



شكل(24): نتائج التنفيذ للقطعة TFA



شكل(25) نتائج التنفيذ للقطعة OTM



شكل(26) نتائج التنفيذ للقطعة TPA

13. الخاتمة:

تم في هذه المرحلة من الدراسة تطوير المكتبة البرامجية المؤسسة على لغة الـVHDL للمنطق الثلاثي حيث تم إضافة البوابة TXOR والقطع التجميعية الثلاثية بدءاً بالقطعة TDEC وانتهاءً بالقطعة TPA.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.**

https://doi.org/10.20428/JST.23.2.3

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

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

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

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

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

[8] 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.

[9] 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.

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

[11] 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.

[12] 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.

https://doi.org/10.20428/JST.23.2.3

الملحق

الإضافة على مكتبة تعريف المنطق الثلاثي :

```
PACKAGE ternary_type IS
        TYPE t_logic IS ('U','0','1','2');
        TYPE t_logic_vector is array (natural range <>) of t_logic;
        FUNCTION txor          (a,b: 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 txor_table: t_logic_table := (
-- ----------------------
-- |  U    0    1    2  |
-- ----------------------
      ( 'U', 'U', 'U', 'U' ),  -- | U |
      ( 'U', '0', '1', '2' ),  -- | 0 |
      ( 'U', '1', '2', '0' ),  -- | 1 |
      ( 'U', '2', '0', '1' ));-- | 2 |

------------------------------------------------------------
-- TXOR
------------------------------------------------------------
FUNCTION txor ( a,b :t_logic) RETURN t_logic IS
BEGIN
      RETURN (txor_table(a,b)) ;
END txor;
```

برنامج لغة الـ VHDL للقطعة TDEC بالمقاس 1 إلى 3

```
USE WORK.ternary_type.ALL;
entity TDEC1_3 is
```


https://doi.org/10.20428/JST.23.2.3

```
PORT (x          : IN  t_logic;
      a,b,c : OUT t_logic);
end entity;

architecture arc of TDEC1_3 is
begin
    process(x)
    begin
        if x = '0'      then a<='2'; b<='0'; c<='0';
        elsif x = '1'   then a<='0'; b<='2'; c<='0';
        elsif x = '2'   then a<='0'; b<='0'; c<='2';
        else            a<='U';      b<='U';      c<='U';
        end if;
    end process;
end arc;
```

برنامج لغة الـ VHDL للقطعة TDEC بالمقاس 2 إلى 9

```
USE WORK.ternary_type.ALL;
entity TDEC2_9 is
  PORT (x          : IN  t_logic_vector(1 downto 0);
        output: OUT t_logic_vector(8 downto 0));
end entity;
architecture arc of TDEC2_9 is
begin
    output<= "000000002" when x="00" else
             "000000020" when x="01" else
             "000000200" when x="02" else
             "000002000" when x="10" else
             "000020000" when x="11" else
             "000200000" when x="12" else
```

```
                "002000000" when x="20" else
                "020000000" when x="21" else
                "200000000" when x="22" else
                "UUUUUUUUU" ;
end arc;
```

برنامج لغة الـ VHDL للقطعة TENC

```
USE WORK.ternary_type.ALL;
entity TENC is
  PORT (x       : IN  t_logic_vector(1 downto 0);
        output: OUT t_logic);
end entity;
architecture arc of TENC is
begin
      with x select
      output <= '1' when "00",
                '2' when "02",
                '0' when "20",
                '1' when "22",
                'U' when others;
end arc;
```

برنامج لغة الـ VHDL للقطعة TMUX بالمقاس 3 إلى 1

```
USE WORK.ternary_type.ALL;
entity TMUX3_1 is
  PORT (a,b,c : IN  t_logic;
        sel   : in  t_logic;
        z     : OUT t_logic);
end entity;
architecture arc of TMUX3_1 is
begin
```


https://doi.org/10.20428/JST.23.2.3

```
        z<= a when sel='0' else
             b when sel='1' else
             c when sel='2' else
             'U' ;
end arc;
```

برنامج لغة الـ VHDL للقطعة TMUX بالمقاس 9 إلى 1

```
USE WORK.ternary_type.ALL;
entity TMUX9_1 is
  PORT (a,b,c : IN  t_logic;
        d,e,f : IN  t_logic;
        g,h,k : IN  t_logic;
         sel   : in  t_logic_vector(1 downto 0);
         z     : OUT t_logic);
end entity;
architecture arc of TMUX9_1 is
begin
        with sel select
        z <= a when "00",
             b when "01",
             c when "02",
             d when "10",
             e when "11",
             f when "12",
             g when "20",
             h when "21",
             k when "22",
             'U' when others;
end arc;
```

برنامج لغة الـ VHDL للقطعة TDMUX بالمقاس 1 إلى 3

```
USE WORK.ternary_type.ALL;
entity TDMUX1_3 is
  PORT (x     : IN  t_logic;
        sel   : IN  t_logic;
        a,b,c : OUT t_logic);
end entity;
architecture arc of TDMUX1_3 is
begin
        process(sel)
        begin
                if sel = '0'      then a <= x; b <= 'U'; c <= 'U';
                elsif sel = '1' then a <= 'U'; b <= x; c <= 'U';
                elsif sel = '2' then a <= 'U'; b <= 'U'; c <= x;
                else                 a <= 'U'; b <= 'U'; c <= 'U';
                end if;
        end process;
end arc;
```

برنامج لغة الـ VHDL للقطعة TDMUX بالمقاس 1 إلى 9

```
USE WORK.ternary_type.ALL;
entity TDMUX1_9 is
  PORT (x     : IN  t_logic;
        sel   : IN  t_logic_vector(1 downto 0);
        a,b,c : OUT t_logic;
        d,e,f : OUT t_logic;
        g,h,k : OUT t_logic);
end entity;
architecture arc of TDMUX1_9 is
begin
```


https://doi.org/10.20428/JST.23.2.3

```
process(sel)
begin
        case sel is
                when "00" =>  a <= x;   b <= 'U'; c <= 'U';
                              d <= 'U'; e <= 'U'; f <= 'U';
                              g <= 'U'; h <= 'U'; k <= 'U';
                when "01" => a <= 'U'; b <= x;   c <= 'U';
                             d <= 'U'; e <= 'U'; f <= 'U';
                             g <= 'U'; h <= 'U'; k <= 'U';
                when "02" => a <= 'U'; b <= 'U'; c <= x;
                             d <= 'U'; e <= 'U'; f <= 'U';
                             g <= 'U'; h <= 'U'; k <= 'U';
                when "10" => a <= 'U'; b <= 'U'; c <= 'U';
                             d <= x;   e <= 'U'; f <= 'U';
                             g <= 'U'; h <= 'U'; k <= 'U';
                when "11" => a <= 'U'; b <= 'U'; c <= 'U';
                             d <= 'U'; e <= x;   f <= 'U';
                             g <= 'U'; h <= 'U'; k <= 'U';
                when "12" => a <= 'U'; b <= 'U'; c <= 'U';
                             d <= 'U'; e <= 'U'; f <= x;
                             g <= 'U'; h <= 'U'; k <= 'U';
                when "20" => a <= 'U'; b <= 'U'; c <= 'U';
                             d <= 'U'; e <= 'U'; f <= 'U';
                             g <= x;   h <= 'U'; k <= 'U';
                when "21" => a <= 'U'; b <= 'U'; c <= 'U';
                             d <= 'U'; e <= 'U'; f <= 'U';
                             g <= 'U'; h <= x;   k <= 'U';
                when "22" => a <= 'U'; b <= 'U'; c <= 'U';
                             d <= 'U'; e <= 'U'; f <= 'U';
```

```
                                    g <= 'U';  h <= ‹U›;  k <= x;
                        when others =>a <= 'U';  b <= 'U';  c <= 'U';
                                      d <= 'U';  e <= 'U';  f <= 'U';
                                      g <= 'U';  h <= 'U';  k <= 'U';
                        end case;
            end process;
end arc;
```

برنامج لغة الـ VHDL للقطعة OTC

```
USE WORK.ternary_type.ALL;
--One Trit Comprator
entity OTC is
  PORT (x,y    : IN  t_logic;
        grt    : OUT t_logic;
        eq     : OUT t_logic;
        less   : OUT t_logic);
end entity;
architecture arc of OTC is
begin
      process(x,y)
      variable a,b : integer;
      begin
            a := to_dec (x);
            b := to_dec (y);
            if a = b     then grt  <= '0'; eq   <= '2'; less <= '0';
            elsif a > b then grt  <= '2'; eq   <= '0'; less <= '0';
            elsif a < b then grt  <= '0'; eq   <= '0'; less <= '2';
            else                grt  <= 'U'; eq   <= 'U'; less <= 'U';
            end if;
```


https://doi.org/10.20428/JST.23.2.3

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

برنامج لغة الـ VHDL للقطعة THA

```
USE WORK.ternary_type.ALL;
--Ternary Half Adder
entity THA is
  PORT (x,y    : IN  t_logic;
        sum    : OUT t_logic;
        carry  : OUT t_logic);
end entity;

architecture arc of THA is
begin
      sum   <= txor (x,y);
      carry <= '1' when x = '1' and y = '2' else
               '1' when x = '2' and y = '1' else
               '1' when x = '2' and y = '2' else
               '0';
end arc;
```

برنامج لغة الـ VHDL للقطعة TFA

```
USE WORK.ternary_type.ALL;
--Ternary Full Adder
entity TFA is
  PORT (x,y,z  : IN  t_logic;
        sum    : OUT t_logic;
        carry  : OUT t_logic);
end entity;
architecture arc of TFA is
component THA
```

```
        PORT (x,y       : IN  t_logic;
                sum     : OUT t_logic;
                carry   : OUT t_logic);
end component;
signal s1,c1,s2,c2,c3: t_logic;
begin
        U1: THA         port map (x, y, s1, c1);
        U2: THA         port map (s1, z, s2, c2);
        --to find the final carry
        U3: THA         port map (c1, c2, carry, c3);
        sum   <= s2 ;
end arc;
```

برنامج لغة الـ VHDL للقطعة OTM

```
USE WORK.ternary_type.ALL;
--One Trit Multiplier
entity OTM is
  PORT (x,y     : IN  t_logic;
          prod  : OUT t_logic;
          carry : OUT t_logic);
end entity;
architecture arc of OTM is
begin
        prod  <= '1' when x = '1' and y = '1'else
                  '1' when x = '2' and y = '2' else
                  '2' when x = '1' and y = '2' else
                  '2' when x = '2' and y = '1' else
                  '0';
        carry <= '1' when x = '2' and y = '2' else
                 '0';
end arc;
```


https://doi.org/10.20428/JST.23.2.3

برنامج لغة الـ VHDL للقطعة TPA

```
USE WORK.ternary_type.ALL;
--Ternary parallel Adder
entity TPA is
        generic (width: integer:= 3);
  PORT (x,y     : IN  t_logic_vector(width-1 downto 0);
          cin   : IN  t_logic;
          sum   : OUT t_logic_vector(width-1 downto 0);
          cout  : OUT t_logic);
end entity;
architecture arc of TPA is
component TFA
  PORT (x,y,z   : IN  t_logic;
          sum   : OUT t_logic;
          carry : OUT t_logic);
end component;
signal carry: t_logic_vector(width downto 0);
begin
        carry(0) <= cin;
        F1: for i in 0 to width-1 generate
                U1: TFA port map (x(i), y(i), carry(i), sum(i), carry(i+1));
        end generate;
        cout <= carry(width);
end arc;
```