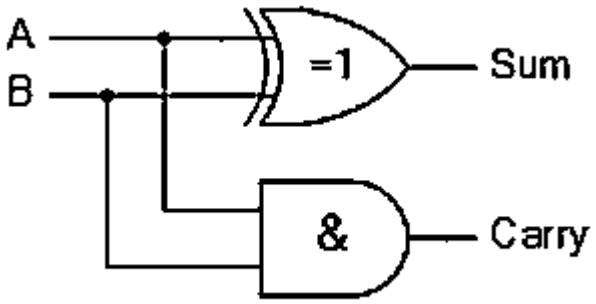
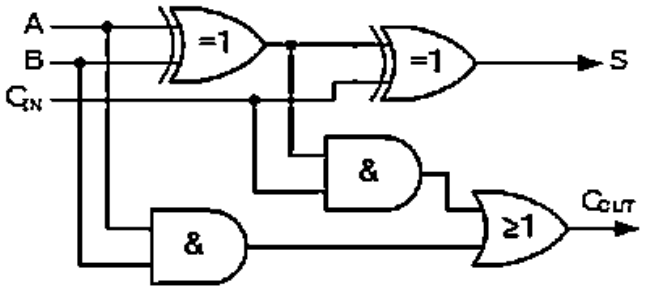


Truth Table			
B	A	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

(PRIOR ART)

Figure 1



Truth Table				
C-in	B	A	Sum	C-out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

(PRIOR ART)

Figure 2

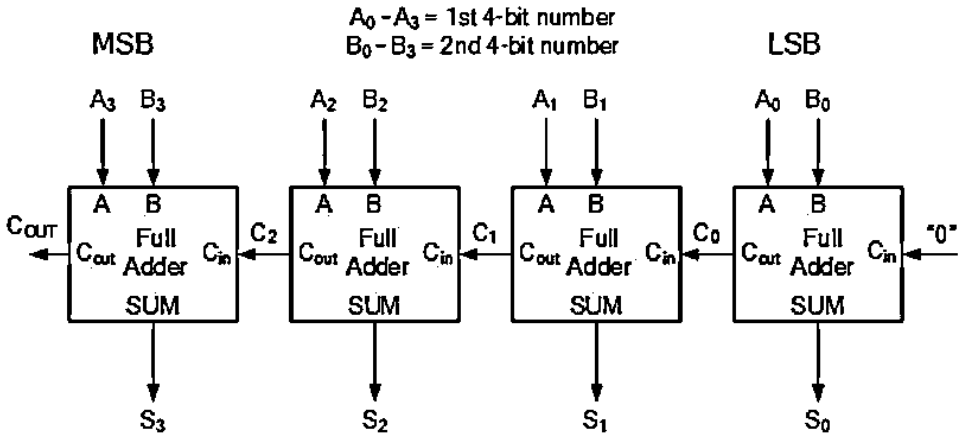


Figure 3 (PRIOR ART)

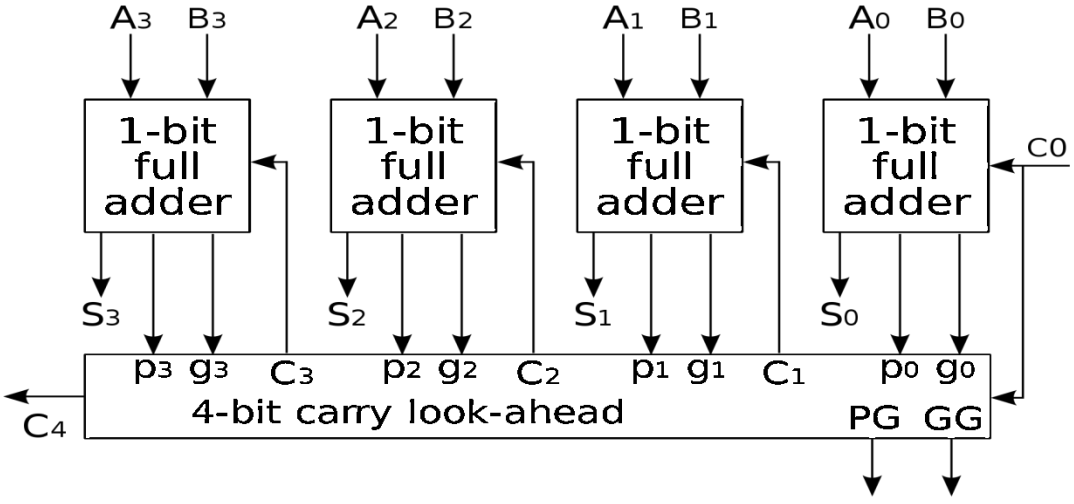


Figure 4 (PRIOR ART)

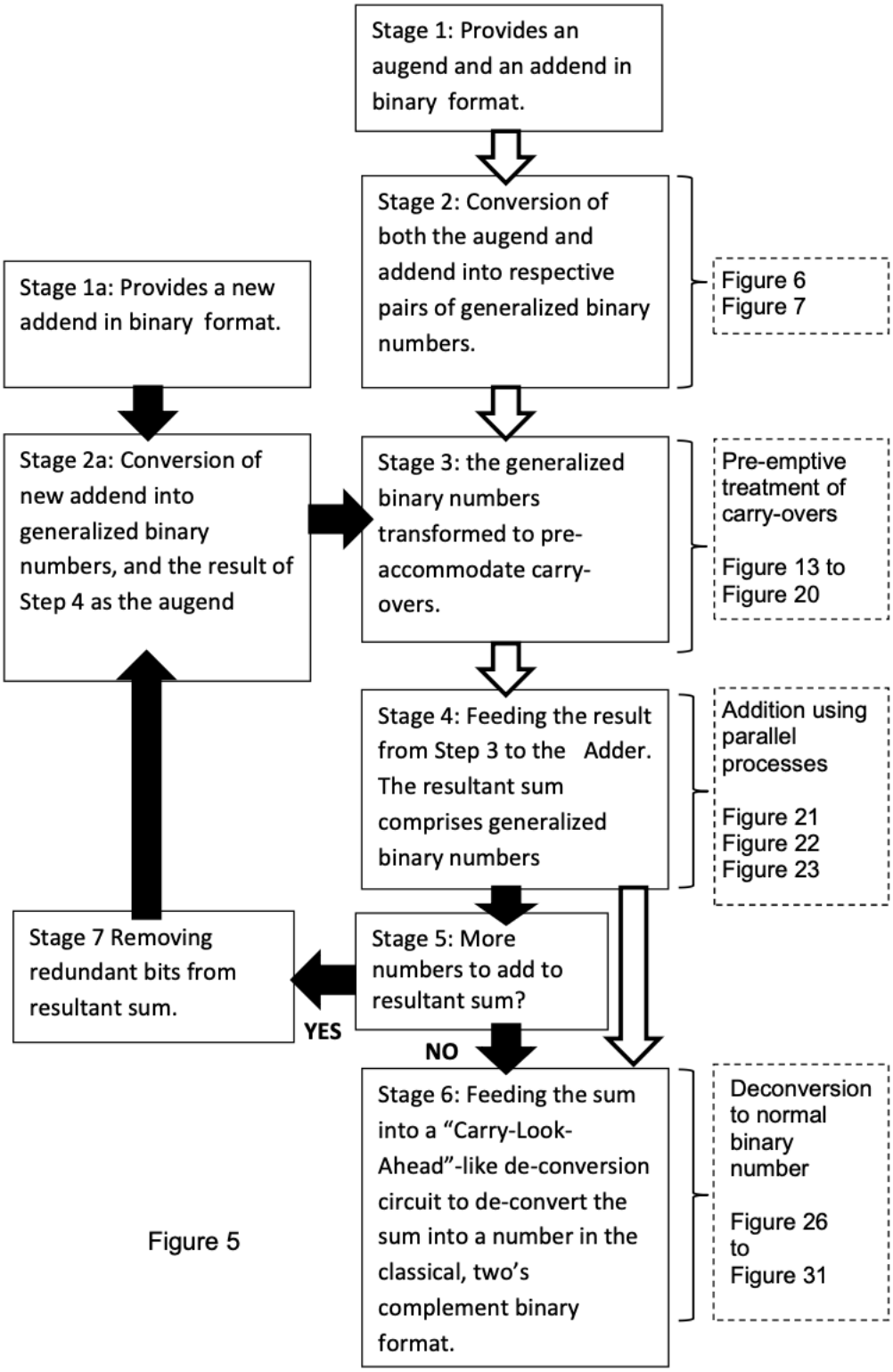

Stage 1: Provides an augend and an addend in binary format.
Stage 2: Conversion of both the augend and addend into respective pairs of generalized binary numbers.
Figure 6
Figure 7
Stage 1a: Provides a new addend in binary format.
Stage 2a: Conversion of new addend into generalized binary numbers, and the result of Step 4 as the augend
Stage 3: the generalized binary numbers transformed to pre-accommodate carry-overs.
Pre-emptive treatment of carry-overs
Figure 13 to Figure 20
Stage 4: Feeding the result from Step 3 to the Adder. The resultant sum comprises generalized binary numbers
Addition using parallel processes
Figure 21
Figure 22
Figure 23
Stage 7 Removing redundant bits from resultant sum.
Stage 5: More numbers to add to resultant sum?
YES
NO
Stage 6: Feeding the sum into a "Carry-Look-Ahead"-like de-conversion circuit to de-convert the sum into a number in the classical, two's complement binary format.
Deconversion to normal binary number
Figure 26
to
Figure 31

Figure 5

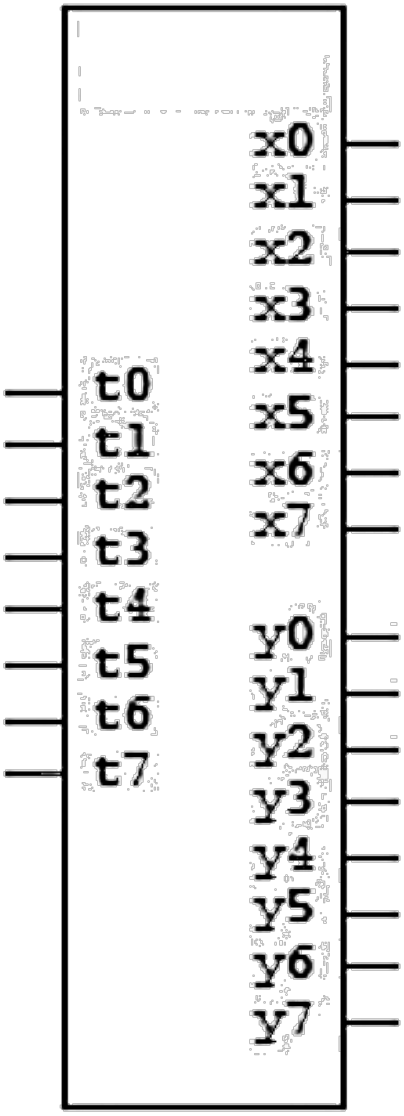


Figure 6

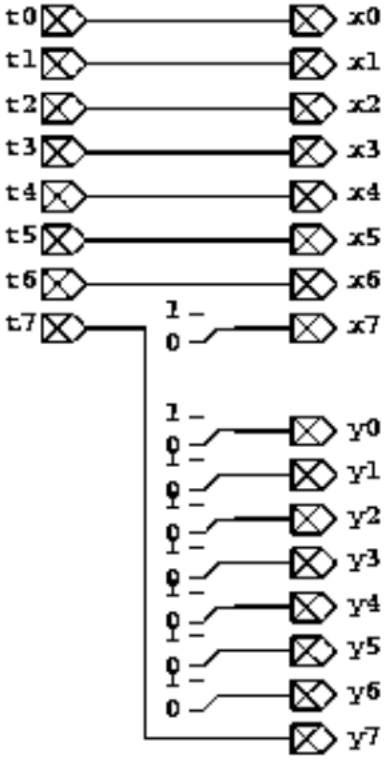


Figure 7

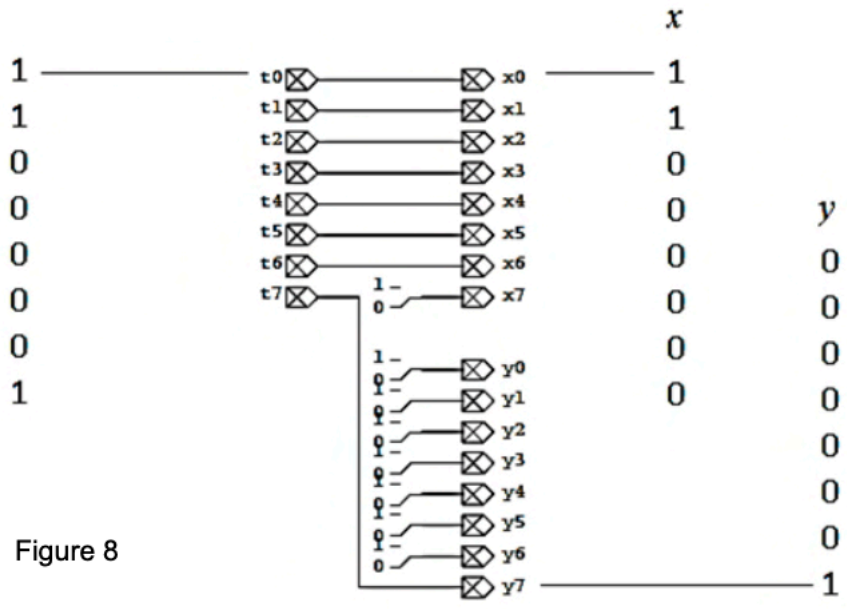


Figure 8

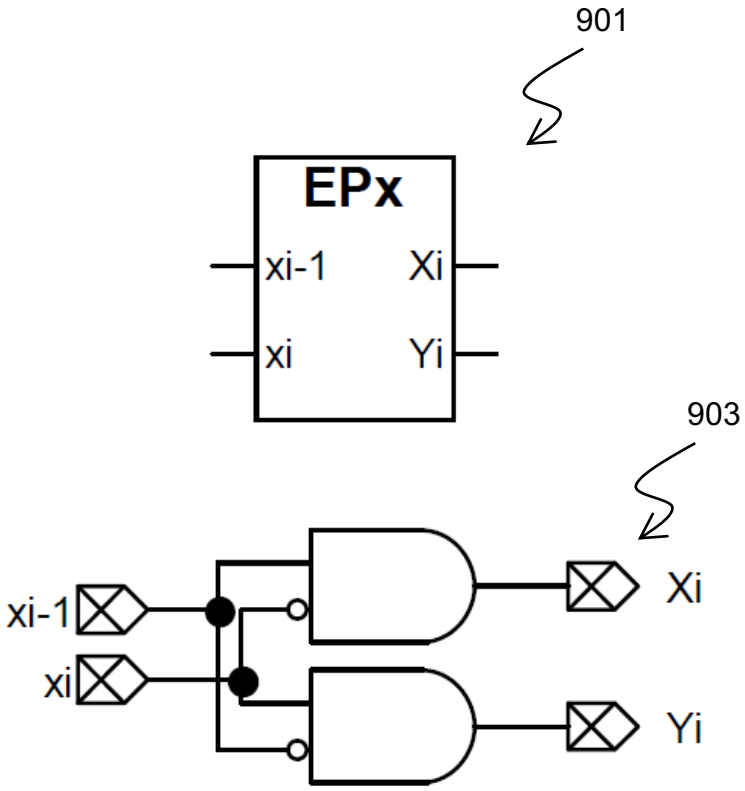


Figure 9

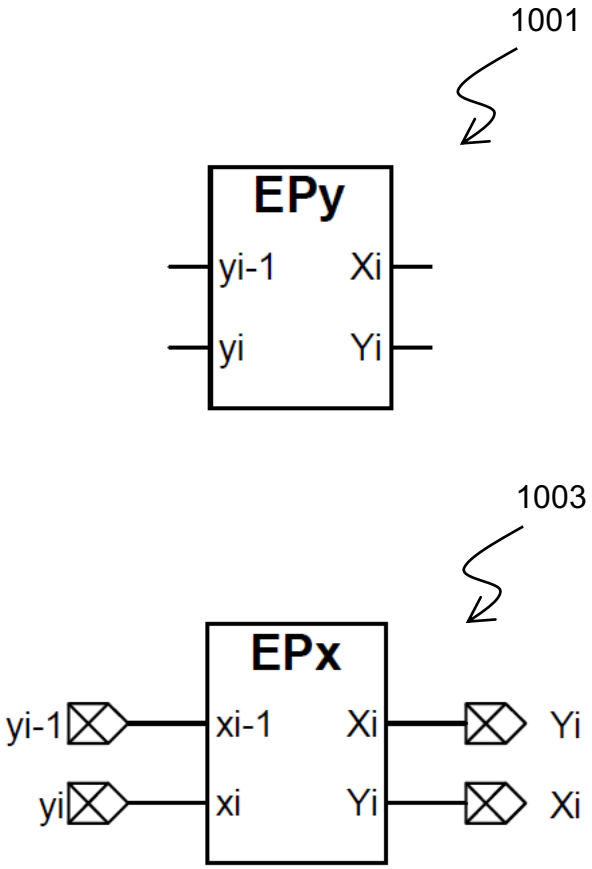


Figure 10

Substring of $m\ (\geq 1)$ 1's				Substring of $m\ (\geq 1)\ \overline{1}$'s			
x_i	x_{i-1}	X_i	Y_i	y_i	y_{i-1}	X_i	Y_i
0	0	0	0	0	0	0	0
0	1	1	0	0	$\overline{1}$	0	$\overline{1}$
1	0	0	$\overline{1}$	$\overline{1}$	0	1	0
1	1	0	0	$\overline{1}$	$\overline{1}$	0	0

Figure 11

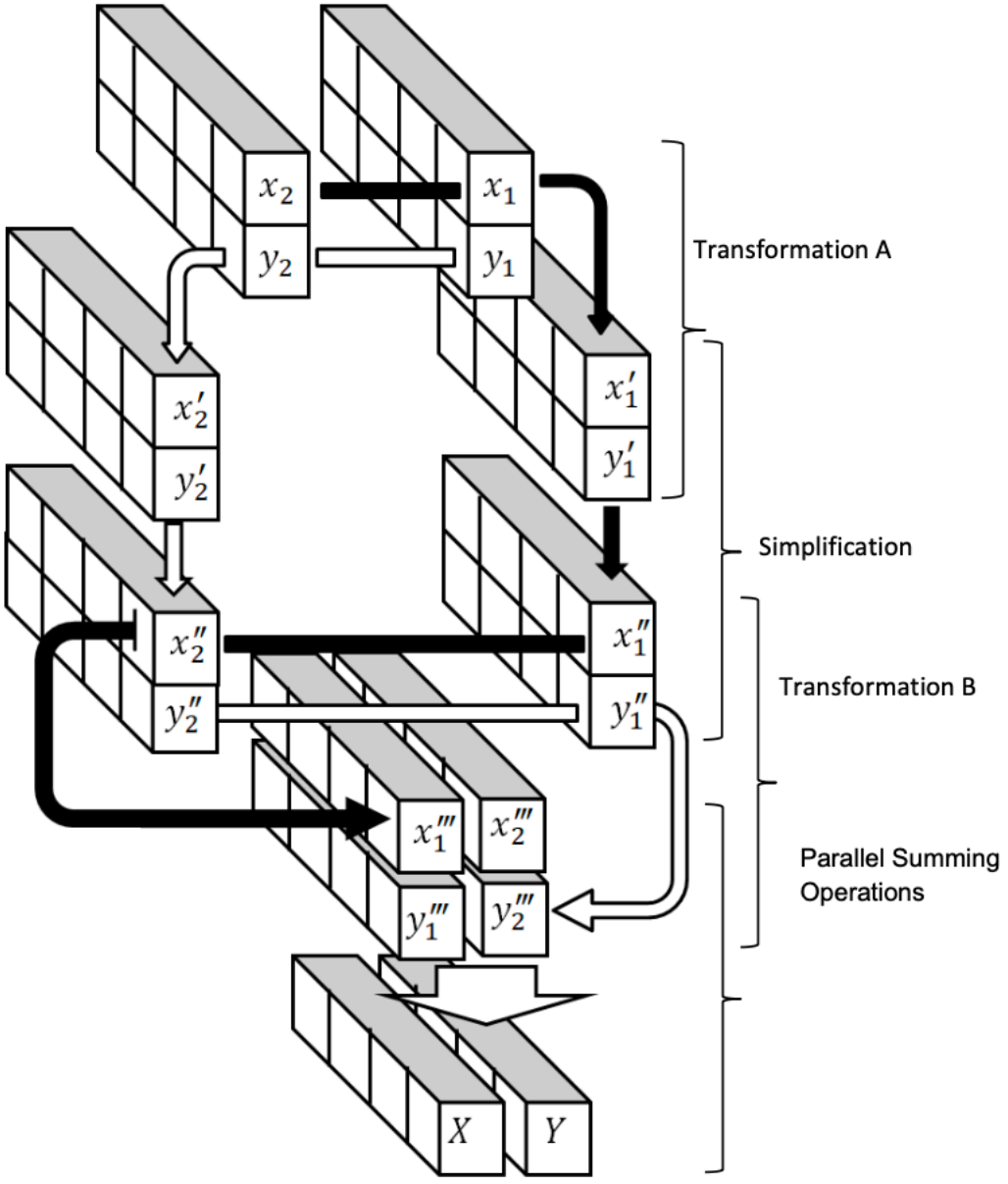

x_2
y_2
x_1
y_1
Transformation A
x_1'
y_1'
x_2'
y_2'
Simplification
x_1''
y_1''
x_2''
y_2''
Transformation B
x_1'''
x_2'''
y_1'''
y_2'''
Parallel Summing Operations
X
Y

Figure 12

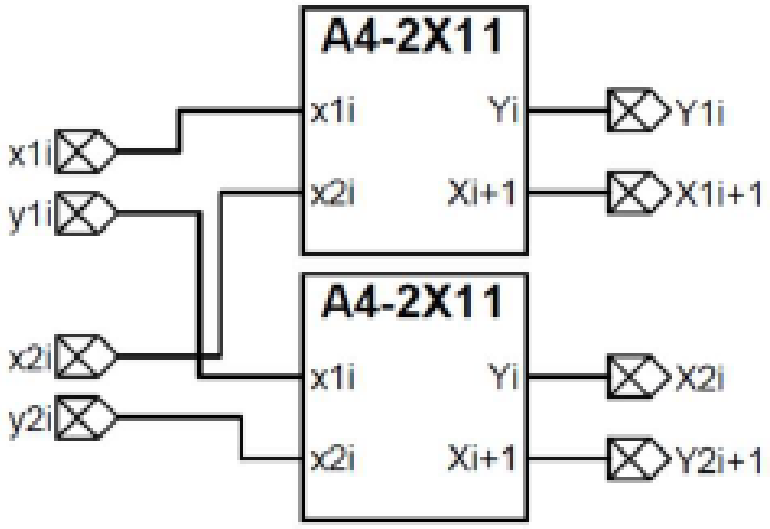


Figure 13

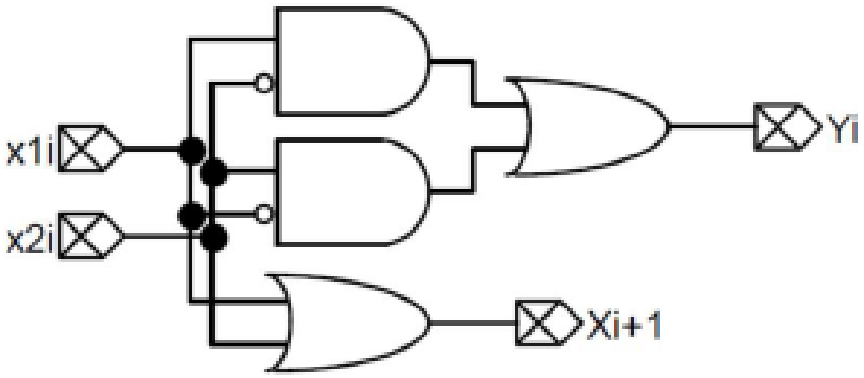


Figure 14

Positive input components				Negative input components			
Input n_w^+		Output $(n'_w)_1$		Input n_w^-		Output $(n'_w)_2$	
$(x_i)_1$	$(x_i)_2$	$(x'_{i+1})_1$ ‡	$(y'_i)_1$ §	$(y_i)_1$	$(y_i)_2$	$(y'_{i+1})_2$	$(x'_i)_2$
0	0	0	0	0	0	0	0
0	1	1	$\overline{1}$	0	$\overline{1}$	$\overline{1}$	1
1	0	1	$\overline{1}$	$\overline{1}$	0	$\overline{1}$	1
1	1	1	0	$\overline{1}$	$\overline{1}$	$\overline{1}$	0

Figure 15

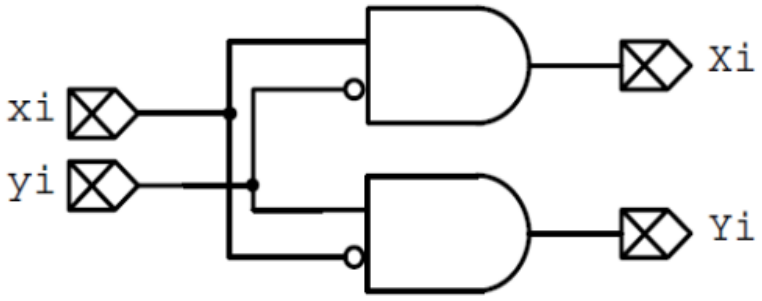


Figure 16

Input $\hat{z}_j$	Output z'_j
0^*	0
1	1
$\overline{1}$	$\overline{1}$

0^* implies $\hat{x}_j = \hat{y}_j = 0$ or $\hat{x}_j = \hat{y}_j = 1$

Figure 17

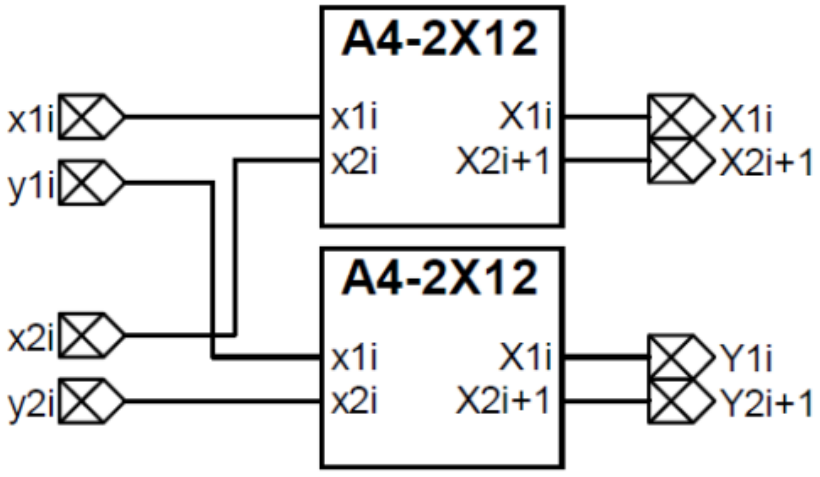


Figure 18

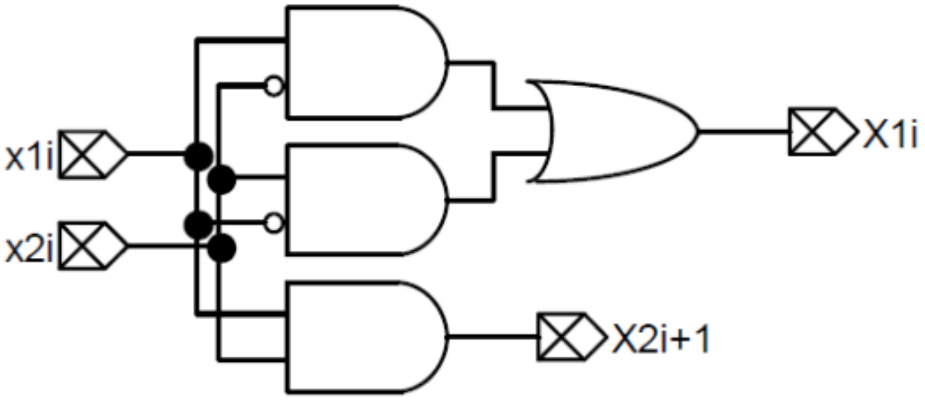


Figure 19

Positive input		Output		Negative input		Output	
$(x_i'')_1$	$(x_i'')_2$	$(x_{i+1}''')_2$	$(x_i''')_1$	$(y_i'')_1$	$(y_i'')_2$	$(y_{i+1}''')_2$	$(y_i''')_1$
0	0	0	0	0	0	0	0
0	1	0	1	0	$\overline{1}$	0	$\overline{1}$
1	0	0	1	$\overline{1}$	0	0	$\overline{1}$
1	1	1	0	$\overline{1}$	$\overline{1}$	$\overline{1}$	0

Figure 20

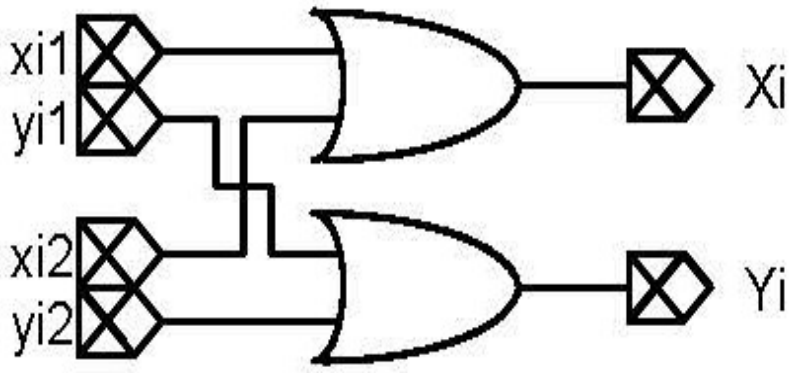


Figure 21

Input n'''_w		Output N_w
$(z'''_i)_1$	$(z'''_i)_2$	Z_i
$0^{\dagger}$	0	0
$0^{\dagger}$	1	1
$0^{\dagger}$	$\overline{1}$	$\overline{1}$
1	0	1
1	1	*
1	$\overline{1}$	0
$\overline{1}$	0	$\overline{1}$
$\overline{1}$	1	0
$\overline{1}$	$\overline{1}$	*

†: $(z'''_i)_1 = 0$ implies $(x'''_i)_1 = (y'''_i)_1 = 0$ or $(x'''_i)_1 = (y'''_i)_1 = 1$

*: impossible inputs

Figure 22

Inputs				Outputs		Simplified Outputs	
x'''_1	y'''_1	x'''_2	y'''_2	X'	Y'	X	Y
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0
0	0	0	$\overline{1}$	0	$\overline{1}$	0	$\overline{1}$
1	0	0	0	1	0	1	0
1	0	1	0	Prevented			
1	0	0	$\overline{1}$	1	$\overline{1}$	0	0
0	$\overline{1}$	0	0	0	$\overline{1}$	0	$\overline{1}$
0	$\overline{1}$	1	0	1	$\overline{1}$	0	0
0	$\overline{1}$	0	$\overline{1}$	Prevented			

Figure 23

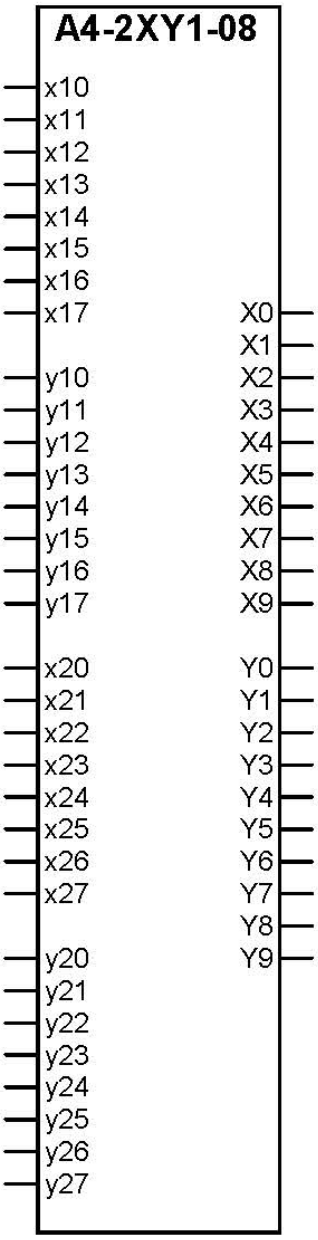

A4-2XY1-08
x10
x11
x12
x13
x14
x15
x16
x17
y10
y11
y12
y13
y14
y15
y16
y17
x20
x21
x22
x23
x24
x25
x26
x27
y20
y21
y22
y23
y24
y25
y26
y27
X0
X1
X2
X3
X4
X5
X6
X7
X8
X9
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
Y8
Y9

Figure 24

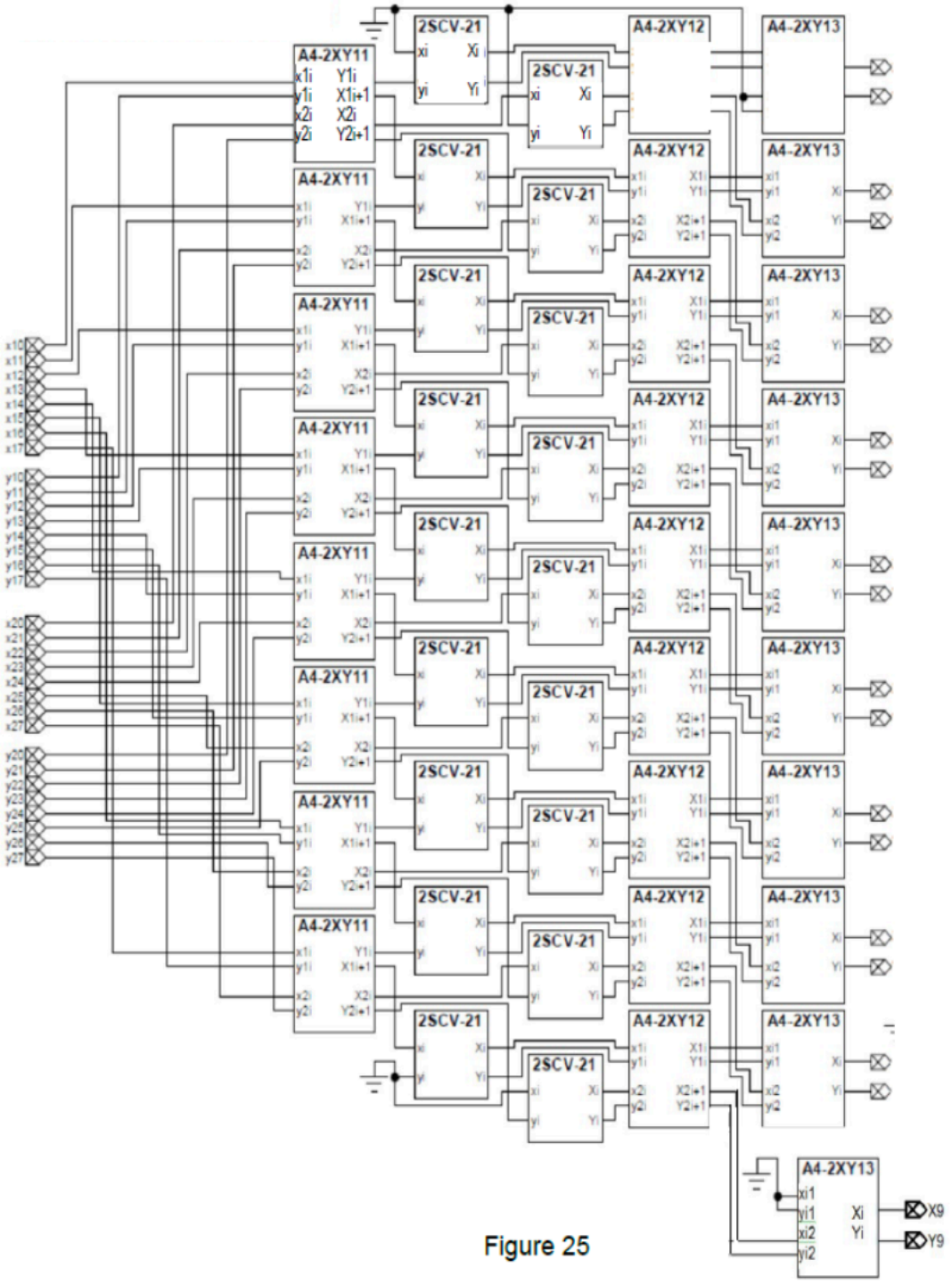

2SCV-21
A4-2XY11
A4-2XY12
A4-2XY13
x1i Y1i
y1i X1i+1
x2i X2i
y2i Y2i+1
xi Xi
yi Yi
xi1 Xi
yi1
xi2 Yi
yi2
X9
Y9

Figure 25

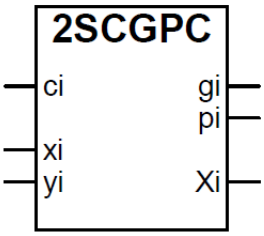


Figure 26

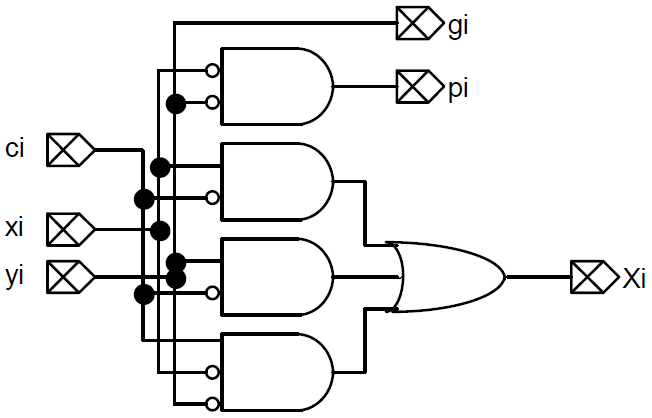


Figure 27

Input z_i	Generate g_i	Propagate p_i
0 ($x_i = y_i = 0$ only)	0	1
1	0	0
$\overline{1}$	1	0

Figure 28

"carry" c_i	General binary number z_i	2's complement number X_i
0	0	0
0	1	1
0	$\overline{1}$	1
1	0	1
1	1	0
1	$\overline{1}$	0

where $i = 0,1,2,\cdots,(n-1)$.

Figure 29

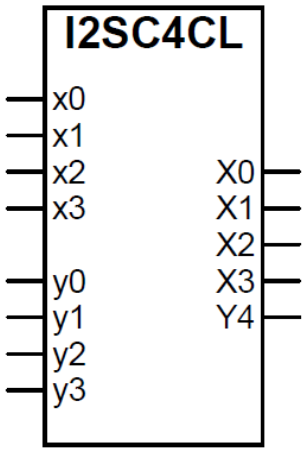


Figure 30

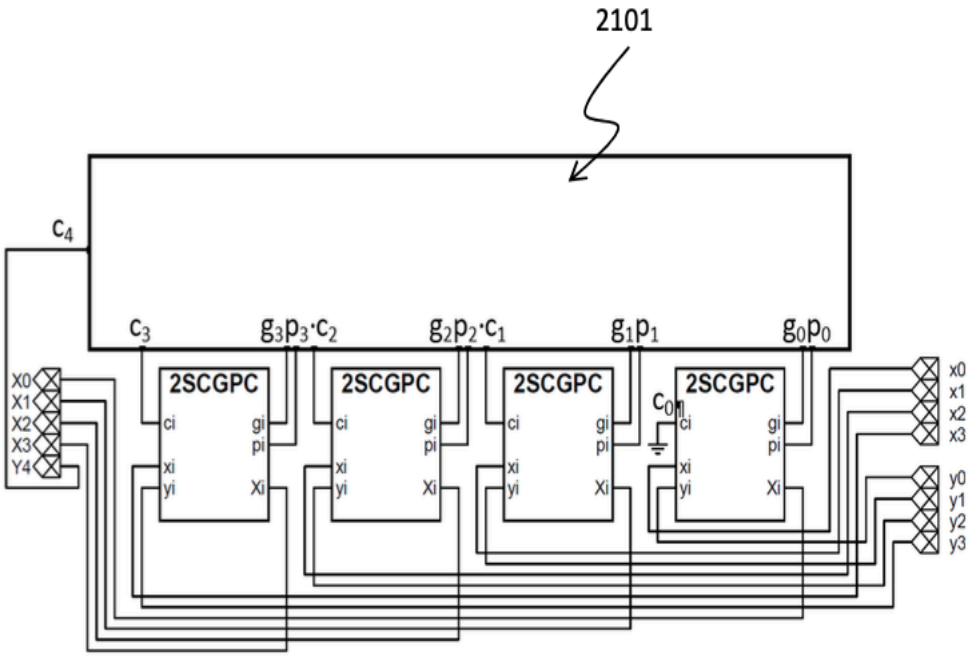


Figure 31

Input		Output		Input		Output	
$(x_i)_1$	$(x_i)_2$	$(x'_i)_1$	$(x'_i)_2$	$(y_i)_1$	$(y_i)_2$	$(y'_i)_1$	$(y'_i)_2$
0	0	0	0	0	0	0	0
0	1	1	0	0	$\overline{1}$	$\overline{1}$	0
1	0	1	0	$\overline{1}$	0	$\overline{1}$	0
1	1	1	1	$\overline{1}$	$\overline{1}$	$\overline{1}$	$\overline{1}$

Figure 32

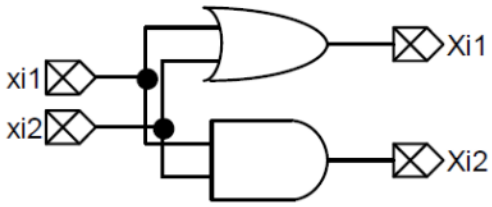


Figure 33

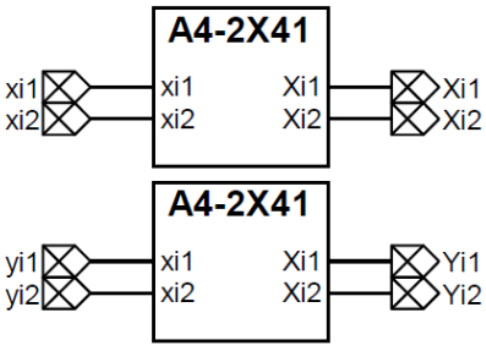


Figure 34

Input			Output			Input			Output		
$(x_i')_1$	$(x_i')_2$	$(x_{i-1}')_1$	$(x_i'')_1$	$(x_i'')_2$	$(y_i'')_3$	$(y_i')_1$	$(y_i')_2$	$(y_{i-1}')_1$	$(y_i'')_1$	$(y_i'')_2$	$(x_i'')_3$
0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	$\bar{1}$	0	$\bar{1}$	0
0	1	0	*	*	*	0	$\bar{1}$	0	*	*	*
0	1	1	*	*	*	0	$\bar{1}$	$\bar{1}$	*	*	*
1	0	0	0	0	$\bar{1}$	$\bar{1}$	0	0	0	0	1
1	0	1	0	0	0	$\bar{1}$	0	$\bar{1}$	0	0	0
1	1	0	0	0	0	$\bar{1}$	$\bar{1}$	0	0	0	0
1	1	1	1	0	0	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	0	0

Figure 35

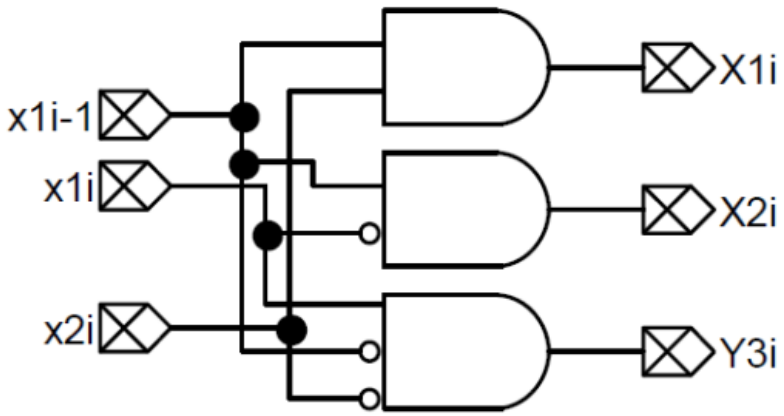


Figure 36

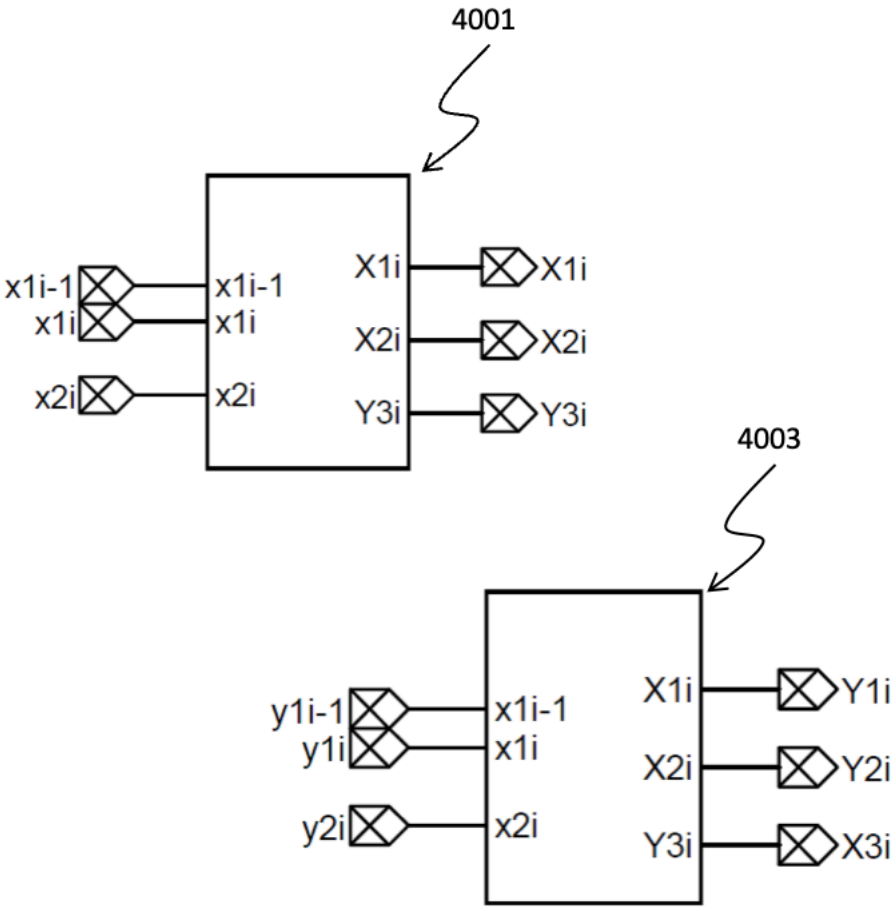

4001
x1i-1
x1i
x2i
x1i-1
x1i
x2i
X1i
X2i
Y3i
X1i
X2i
Y3i
4003
y1i-1
y1i
y2i
x1i-1
x1i
x2i
X1i
X2i
Y3i
Y1i
Y2i
X3i

Figure 37

Input*					Output
$(x_i'')_1$	$(x_i'')_2$	$(x_i'')_3$	$(x_{i-1}'')_2$	$(x_{i-1}'')_3$	x_i'''
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	1
0	0	1	0	0	1
0	0	1	1	0	1
0	1	0	0	0	1
0	1	0	0	1	1
0	1	1	0	0	0
1	0	0	0	0	1
1	0	0	0	1	1
Input					Output
$(y_i'')_1$	$(y_i'')_2$	$(y_i'')_3$	$(y_{i-1}'')_2$	$(y_{i-1}'')_3$	y_i'''
0	0	0	0	0	0
0	0	0	0	$\overline{1}$	0
0	0	0	$\overline{1}$	0	0
0	0	0	$\overline{1}$	$\overline{1}$	$\overline{1}$
0	0	$\overline{1}$	0	0	$\overline{1}$
0	0	$\overline{1}$	$\overline{1}$	0	$\overline{1}$
0	$\overline{1}$	0	0	0	$\overline{1}$
0	$\overline{1}$	0	0	$\overline{1}$	$\overline{1}$
0	$\overline{1}$	$\overline{1}$	0	0	0
$\overline{1}$	0	0	0	0	$\overline{1}$
$\overline{1}$	0	0	0	$\overline{1}$	$\overline{1}$

Figure 38

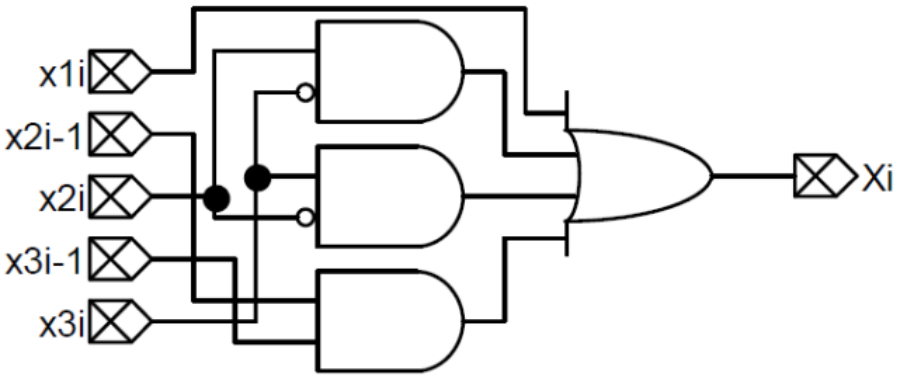


Figure 39

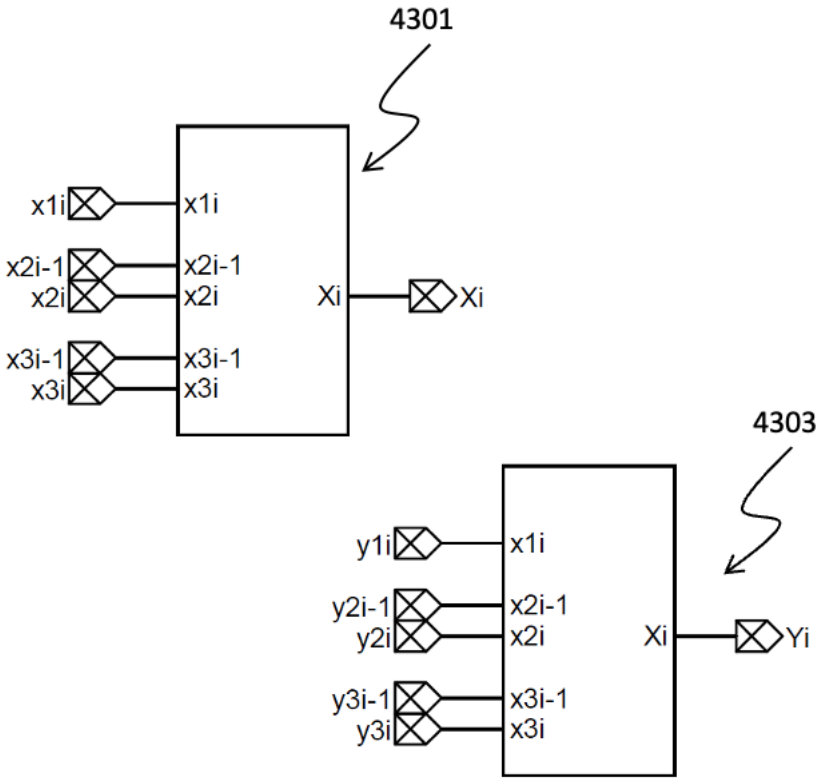


Figure 40

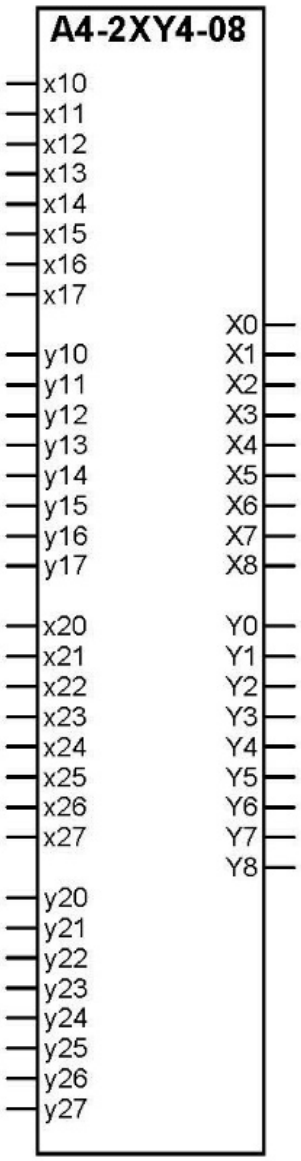

A4-2XY4-08
x10
x11
x12
x13
x14
x15
x16
x17
y10
y11
y12
y13
y14
y15
y16
y17
x20
x21
x22
x23
x24
x25
x26
x27
y20
y21
y22
y23
y24
y25
y26
y27
X0
X1
X2
X3
X4
X5
X6
X7
X8
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
Y8

Figure 41

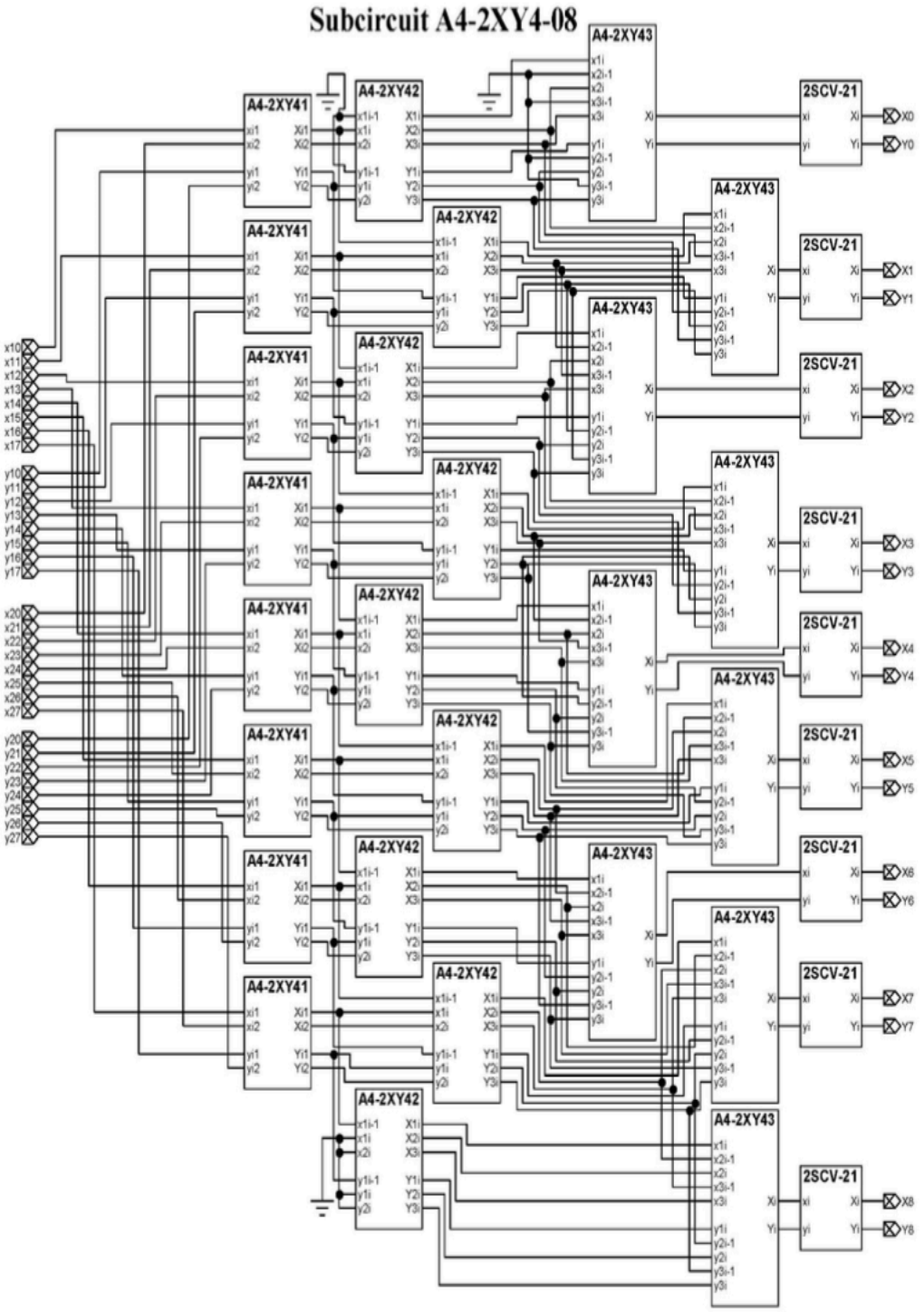


Figure 42

Input				Output	Input				Output
$(x_i'')_1$	$(x_{i-1}'')_1$	$(x_i'')_2$	$(x_{i-1}'')_2$	x_i'''	$(y_i'')_1$	$(y_{i-1}'')_1$	$(y_i'')_2$	$(y_{i-1}'')_2$	y_i'''
0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	$\overline{1}$	0
0	0	1	0	1	0	0	$\overline{1}$	0	$\overline{1}$
0	0	1	1	*	0	0	$\overline{1}$	$\overline{1}$	$\overline{1}$
0	1	0	0	0	0	$\overline{1}$	0	0	0
0	1	0	1	1	0	$\overline{1}$	0	$\overline{1}$	$\overline{1}$
0	1	1	0	1	0	$\overline{1}$	$\overline{1}$	0	$\overline{1}$
0	1	1	1	*	0	$\overline{1}$	$\overline{1}$	$\overline{1}$	*
1	0	0	0	1	$\overline{1}$	0	0	0	$\overline{1}$
1	0	0	1	1	$\overline{1}$	0	0	$\overline{1}$	$\overline{1}$
1	0	1	0	0	$\overline{1}$	0	$\overline{1}$	0	0
1	0	1	1	*	$\overline{1}$	0	$\overline{1}$	$\overline{1}$	0
1	1	0	0	1	$\overline{1}$	$\overline{1}$	0	0	*
1	1	0	1	*	$\overline{1}$	$\overline{1}$	0	$\overline{1}$	*
1	1	1	0	0	$\overline{1}$	$\overline{1}$	$\overline{1}$	0	*
1	1	1	1	*	$\overline{1}$	$\overline{1}$	$\overline{1}$	$\overline{1}$	*

*Impossible inputs

Figure 43

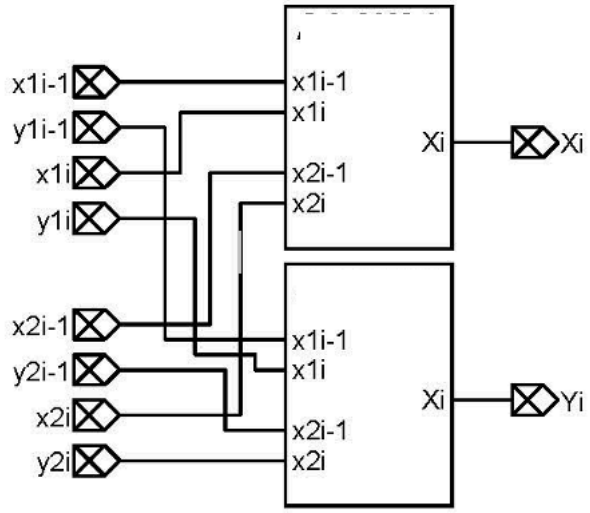


Figure 44

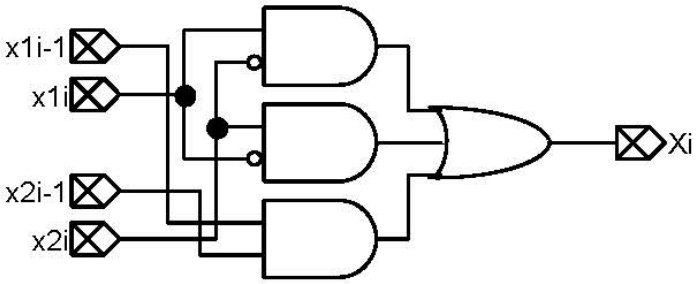


Figure 45

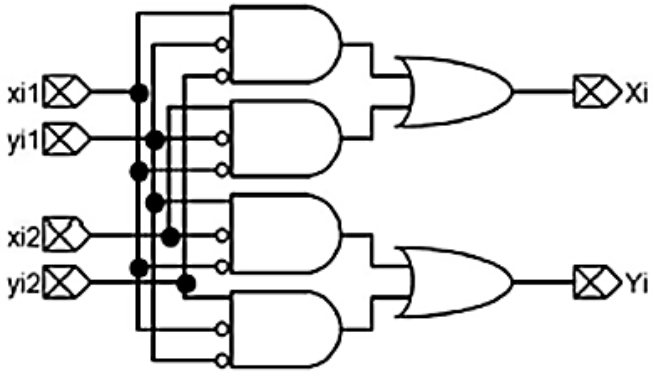


Figure 46

Box 1

The *Specific Equivalence Principle* in a generalized binary number system states that:

- *Any substring of* $m\ (\geq 1)$ *1's can be replaced by an 1 (more significant) and an* $\bar{1}$ *(less significant) separated by m-1 zeroes.*
- *Similarly, any substring of* $m(\geq 1)\ \bar{1}$*'s can be replaced by an* $\bar{1}$ *(more significant) and an 1 (less significant) separated by m-1 zeroes.*

Figure 47

Box 2

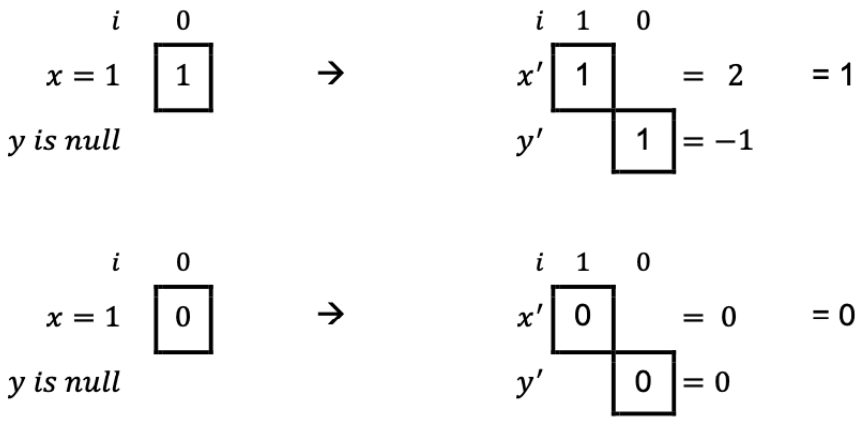


Figure 48

Box 3

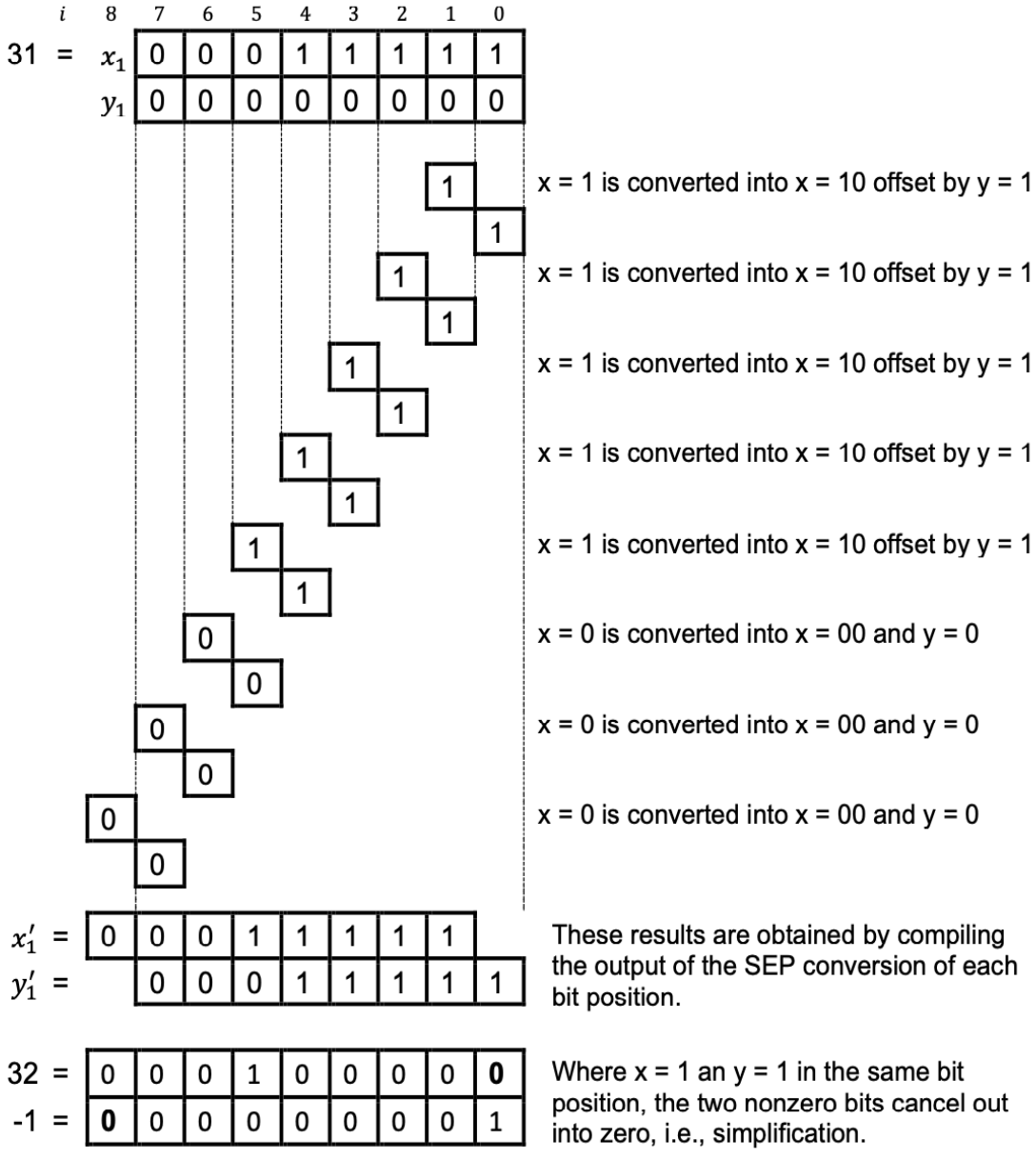


Figure 49

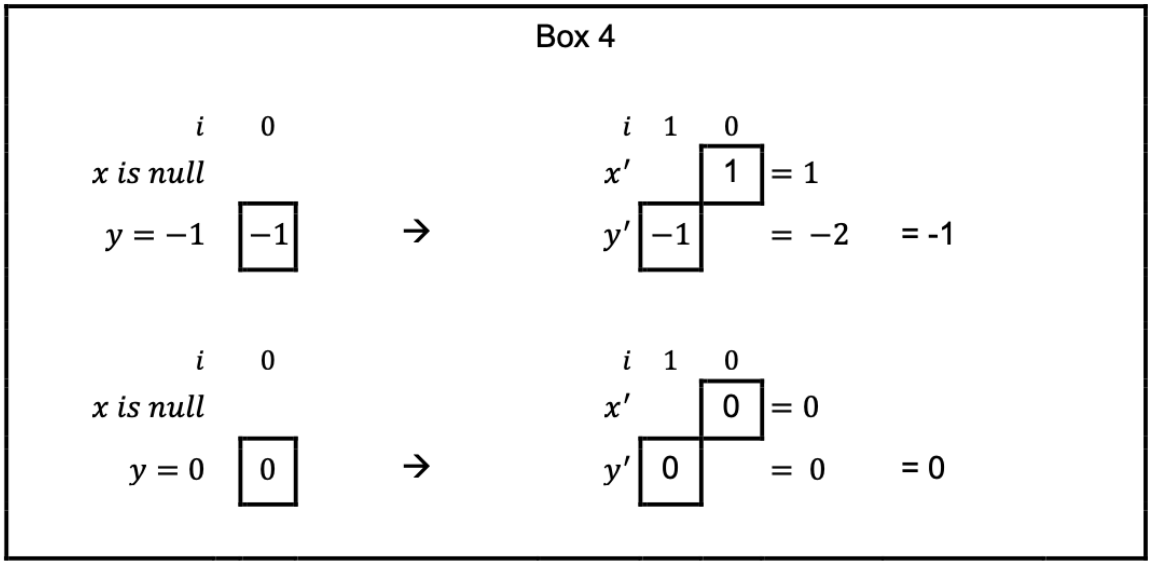


Figure 50

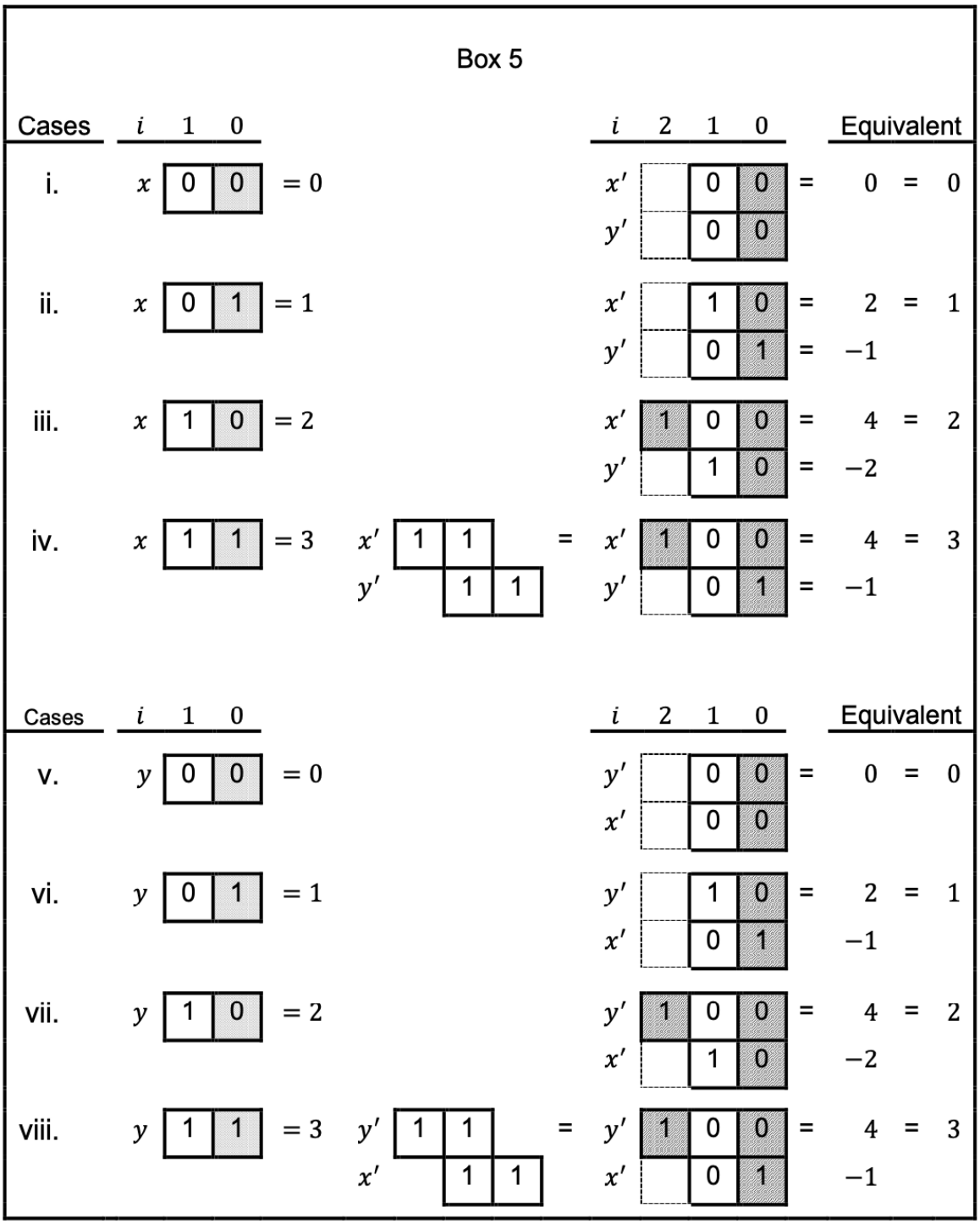

Box 5
Cases
i 1 0
i 2 1 0
Equivalent
i.
ii.
iii.
iv.
Cases
i 1 0
i 2 1 0
Equivalent
v.
vi.
vii.
viii.

Figure 51

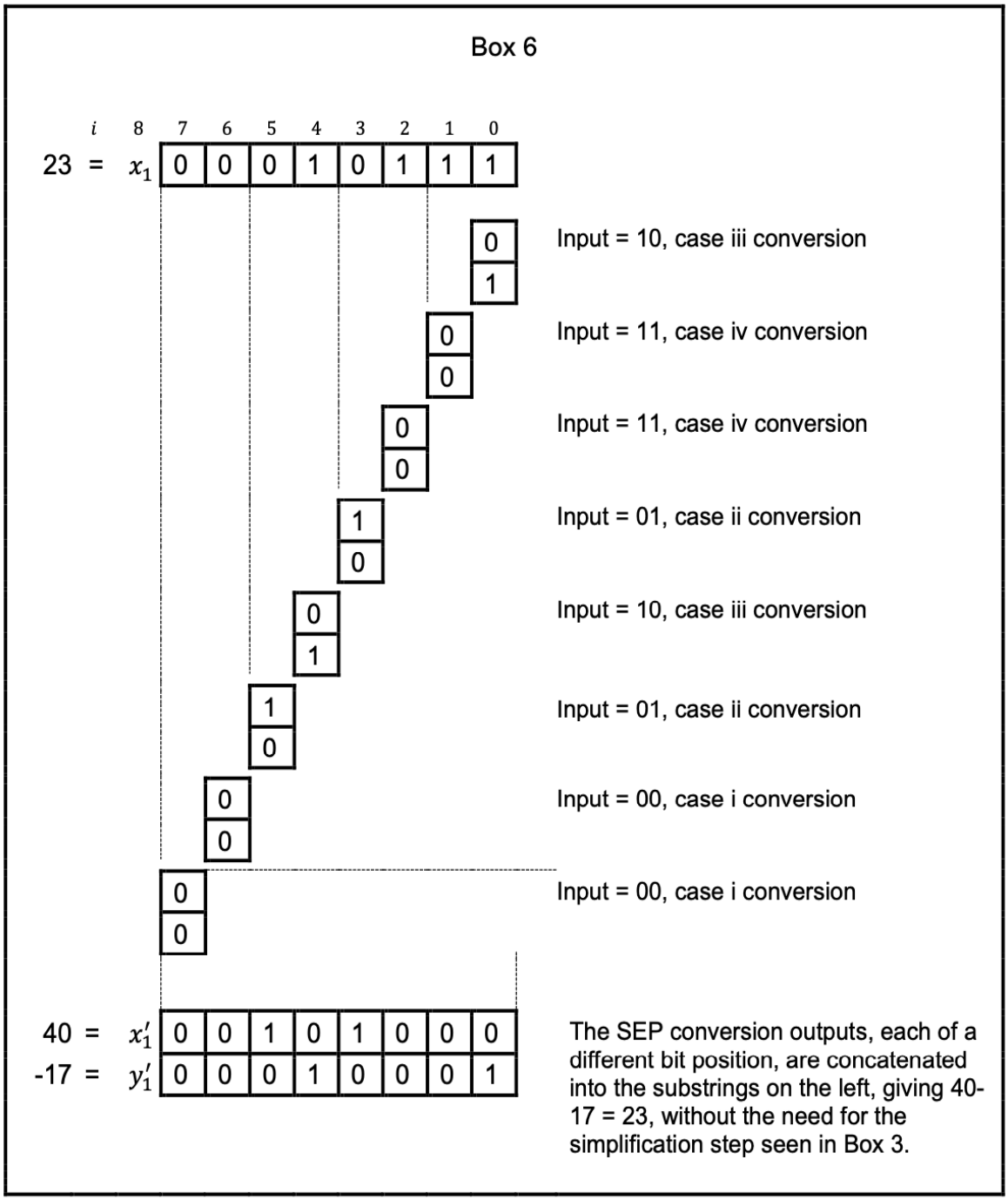

Box 6
i 8 7 6 5 4 3 2 1 0
23 = x1 0 0 0 1 0 1 1 1
Input = 10, case iii conversion
Input = 11, case iv conversion
Input = 11, case iv conversion
Input = 01, case ii conversion
Input = 10, case iii conversion
Input = 01, case ii conversion
Input = 00, case i conversion
Input = 00, case i conversion
40 = x'1 0 0 1 0 1 0 0 0
-17 = y'1 0 0 0 1 0 0 0 1
The SEP conversion outputs, each of a different bit position, are concatenated into the substrings on the left, giving 40-17 = 23, without the need for the simplification step seen in Box 3.

Figure 52

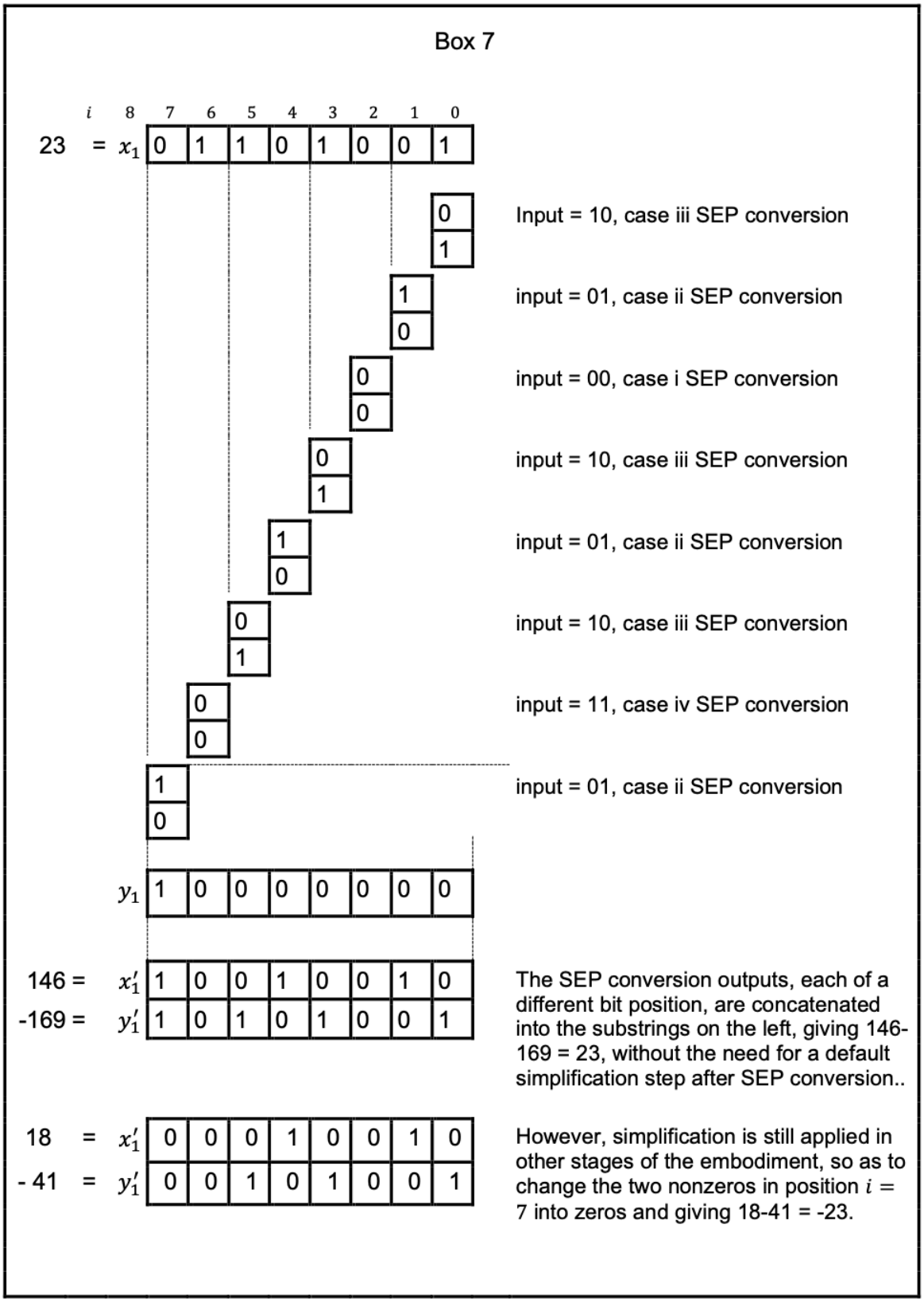

Box 7
i 8 7 6 5 4 3 2 1 0
23 = x1 0 1 1 0 1 0 0 1
Input = 10, case iii SEP conversion
input = 01, case ii SEP conversion
input = 00, case i SEP conversion
input = 10, case iii SEP conversion
input = 01, case ii SEP conversion
input = 10, case iii SEP conversion
input = 11, case iv SEP conversion
input = 01, case ii SEP conversion
y1 1 0 0 0 0 0 0 0
146 = x'1 1 0 0 1 0 0 1 0
-169 = y'1 1 0 1 0 1 0 0 1
The SEP conversion outputs, each of a different bit position, are concatenated into the substrings on the left, giving 146-169 = 23, without the need for a default simplification step after SEP conversion..
18 = x'1 0 0 0 1 0 0 1 0
- 41 = y'1 0 0 1 0 1 0 0 1
However, simplification is still applied in other stages of the embodiment, so as to change the two nonzeros in position i = 7 into zeros and giving 18-41 = -23.

Figure 53

Box 8

										i	8	7	6	5	4	3	2	1
13	=	0	0	0	0	1	1	0	1	= x_1	0	0	0	0	1	1	0	1
										y_1	0	0	0	0	0	0	0	0
14	=	0	0	0	0	1	1	1	0	= x_2	0	0	0	0	1	1	1	0
										y_2	0	0	0	0	0	0	0	0

Figure 54

Box 9

Derivation of Transformation A (positive components)

Case	(a) Input			(b) Sum		SEP	(c) Output			(d) Transformation
		i						$i+1$	i	
i.	x_1	0	+	=	0		x_1'	0		$0-0=0$
	x_2	0					y_1'		0	
ii.	x_1	0	+	=	1	SEP →	x_1'	1		$2-1=1$
	x_2	1					y_1'		1	
iii.	x_1	1	+	=	1	SEP →	x_1'	1		$2-1=1$
	x_2	0					y_1'		1	
iv.	x_1	1	+	=	1 0		x_1'	1		$2-0=2$
	x_2	1					y_1'		0	

Figure 55

Box 10

Derivation of Transformation A (negative components)

Case	(a) Input	(b) Sum	SEP	(c) Output	(d) Transformation
v.	i: y_1 0, y_2 0 +	= 0		$i+1$, i: x'_2 0, y'_2 0	$0 - 0 = 0$
vi.	y_1 0, y_2 $\bar{1}$ +	= -1	SEP →	x'_2 1, y'_2 $\bar{1}$	$1 - 2 = -1$
vii.	y_1 $\bar{1}$, y_2 0 +	= -1	SEP →	x'_2 1, y'_2 $\bar{1}$	$1 - 2 = -1$
viii.	y_1 $\bar{1}$, y_2 $\bar{1}$ +	= -1 0		x'_2 0, y'_2 $\bar{1}$	$0 - 2 = -2$

Note: -1 is only illustrative as the y substring has only absolute value of 1, as explained in a preceding paragraph.

Figure 56

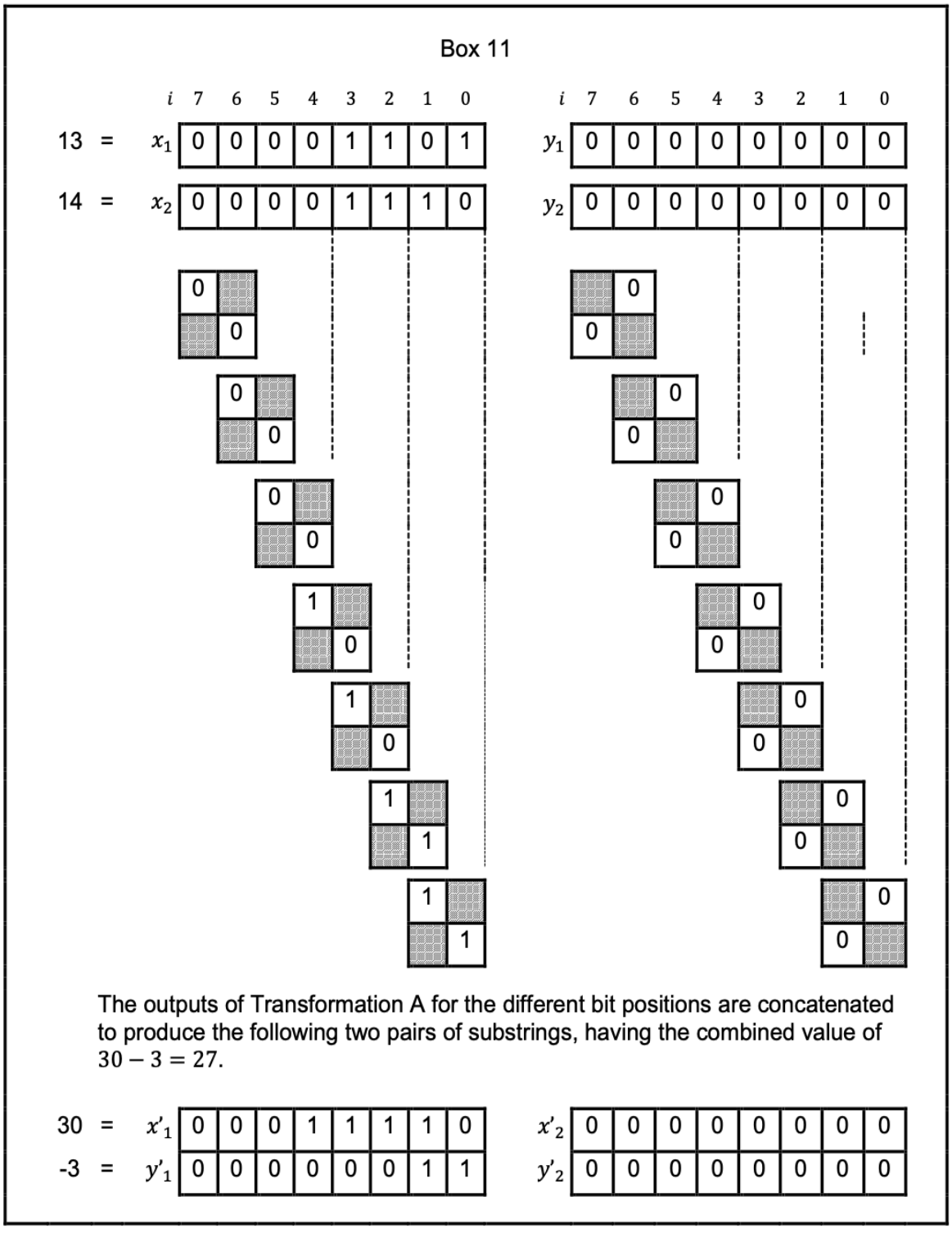

Box 11
i 7 6 5 4 3 2 1 0
13 = x1 0 0 0 0 1 1 0 1
14 = x2 0 0 0 0 1 1 1 0
y1 0 0 0 0 0 0 0 0
y2 0 0 0 0 0 0 0 0
The outputs of Transformation A for the different bit positions are concatenated to produce the following two pairs of substrings, having the combined value of 30 − 3 = 27.
30 = x'1 0 0 0 1 1 1 1 0
-3 = y'1 0 0 0 0 0 0 1 1
x'2 0 0 0 0 0 0 0 0
y'2 0 0 0 0 0 0 0 0

Figure 57

Box 12

		i	7	6	5	4	3	2	1	0		i	7	6	5	4	3	2	1	0	
30	=	x'_1	0	0	0	1	1	1	**1**	0	x'_2		0	0	0	0	0	0	0	0	=0
-3	=	y'_1	0	0	0	0	0	0	**1**	1	y'_2		0	0	0	0	0	0	0	0	=0

Simplification of the above substring pairs give the substring pairs below.

28	=	x''_1	0	0	0	1	1	1	**0**	0	x''_2		0	0	0	0	0	0	0	0	=0
-1	=	y''_1	0	0	0	0	0	0	**0**	1	y''_2		0	0	0	0	0	0	0	0	=0

Figure 58

Box 13

Derivation of Transformation B (positive components)

Case	(a) Input	Sum (bin)	(b) Output	(c) Transformation
1)	x_1'' 0 + x_2'' 0	= 0	x_2''' 0 (at $i+1$) x_1''' 0 (at i)	$0+0=0$
2)	x_1'' 0 + x_2'' 1	= 1	x_2''' 0 (at $i+1$) x_1''' 1 (at i)	$0+1=1$
3)	x_1'' 1 + x_2'' 0	= 1	x_2''' 0 (at $i+1$) x_1''' 1 (at i)	$1+0=1$
4)	x_1'' 1 + x_2'' 1	= 10	x_2''' 1 (at $i+1$) x_1''' 0 (at i)	$1+1=10$

Figure 59

Box 14

Derivation of Transformation B (negative components)

Case	(a) Input	Sum (bin)	(b) Output	(c) Transformation
5)	y_1'' 0 + y_2'' 0	= 0	y_2''' 0 (at $i+1$) y_1''' 0 (at i)	$0+0=0$
6)	y_1'' 0 + y_2'' $\bar{1}$	= -1	y_2''' 0 (at $i+1$) y_1''' $\bar{1}$ (at i)	$0-1=-1$
7)	y_1'' $\bar{1}$ + y_2'' 0	= -1	y_2''' 0 (at $i+1$) y_1''' $\bar{1}$ (at i)	$-1+0=-1$
8)	y_1'' $\bar{1}$ + y_2'' $\bar{1}$	= -10	y_2''' $\bar{1}$ (at $i+1$) y_1''' 0 (at i)	$-1-1=-10$

Figure 60

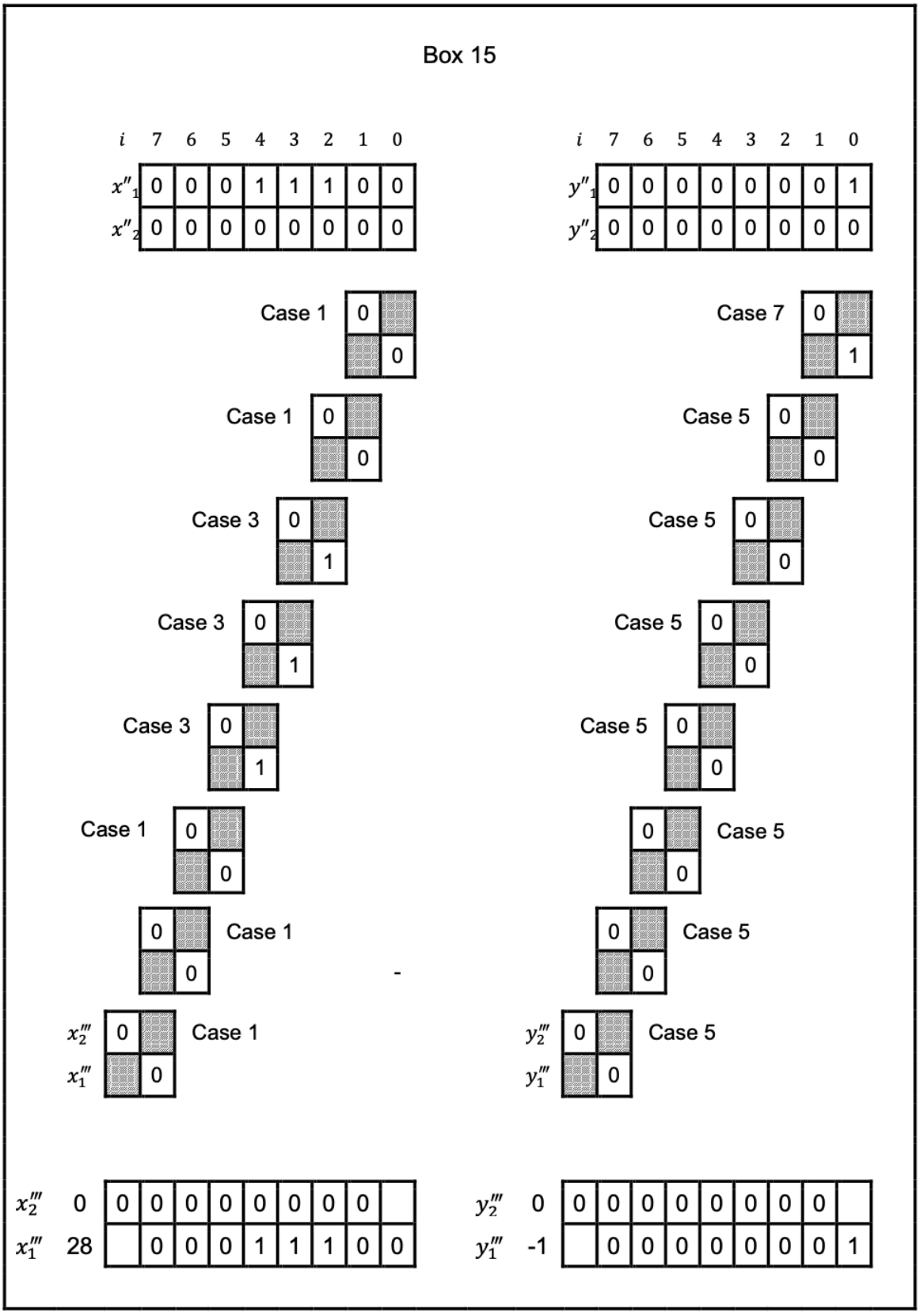

Box 15
i 7 6 5 4 3 2 1 0
x''1 0 0 0 1 1 1 0 0
x''2 0 0 0 0 0 0 0 0
i 7 6 5 4 3 2 1 0
y''1 0 0 0 0 0 0 0 1
y''2 0 0 0 0 0 0 0 0
Case 1
Case 1
Case 3
Case 3
Case 3
Case 1
Case 1
x'''2
x'''1
Case 1
Case 7
Case 5
Case 5
Case 5
Case 5
Case 5
Case 5
y'''2
y'''1
Case 5
-
x'''2 0 0 0 0 0 0 0 0 0
x'''1 28 0 0 0 1 1 1 0 0
y'''2 0 0 0 0 0 0 0 0 0
y'''1 -1 0 0 0 0 0 0 0 1

Figure 61

Box 16

i	7	6	5	4	3	2	1	0
x_2'''	0	0	0	0	0	0	0	0
x_1'''	0	0	0	1	1	1	0	0
X	0	0	0	1	1	1	0	0

i	7	6	5	4	3	2	1	0
y_2'''	0	0	0	0	0	0	0	0
y_1'''	0	0	0	0	0	0	0	1
Y	0	0	0	0	0	0	0	1

For each bit position i added up the values of the bits; and all the adding in the different positions are done at the same time as there is no carry-over required in any downstream addition to wait for the carry-over input.

Figure 62

Box 17		
Generalization of augend $(\tilde{n}_w)_1$ and addend $(\tilde{n}_w)_2$	$(\tilde{n}_w)_1^+ = (x_i)_1$	1 0 0 1 0 1 0 1
	$(\tilde{n}_w)_2^+ = (x_i)_2$	0 1 0 0 0 1 0 0
	$(\tilde{n}_w)_2^- = (y_i)_2$	0 0 0 0 $\overline{1}$ 0 0 $\overline{1}$
	$(\tilde{n}_w)_1^- = (y_i)_1$	0 0 $\overline{1}$ 0 $\overline{1}$ 0 0 0
Transformation A	$(x'_{i+1})_1 = (x_i)_1 + (x_i)_2$	1 1 0 1 0 1 0 1 0
	$(y'_i)_1 = (x_i)_1(\overline{x}_i)_2 + (\overline{x}_i)_1(x_i)_2$	0 $\overline{1}$ $\overline{1}$ 0 $\overline{1}$ 0 0 0 $\overline{1}$
	$(x'_i)_2 = (y_i)_1(\overline{y}_i)_2 + (\overline{y}_i)_1(y_i)_2$	0 0 0 1 0 0 0 0 1
	$(y'_{i+1})_2 = (y_i)_1 + (y_i)_2$	0 0 $\overline{1}$ 0 $\overline{1}$ 0 0 $\overline{1}$ 0
Simplification	$(x''_i)_1 = (x'_i)_1(\overline{y}'_i)_1$	1 0 0 1 0 1 0 1 0
	$(y''_i)_1 = (\overline{x}'_i)_1(y'_i)_1$	0 0 $\overline{1}$ 0 $\overline{1}$ 0 0 0 $\overline{1}$
	$(x''_i)_2 = (x'_i)_2(\overline{y}'_i)_2$	0 0 0 1 0 0 0 0 1
	$(y''_i)_2 = (\overline{x}'_i)_2(y'_i)_2$	0 0 $\overline{1}$ 0 $\overline{1}$ 0 0 $\overline{1}$ 0
Transformation B	$(x'''_i)_1 = (x''_i)_1(\overline{x}''_i)_2 + (\overline{x}''_i)_1(x''_i)_2$	1 0 0 0 0 1 0 1 1
	$(x'''_{i+1})_2 = (x''_i)_1(x''_i)_2$	0 0 0 1 0 0 0 0 0 0
	$(y'''_i)_1 = (y''_i)_1(\overline{y}''_i)_2 + (\overline{y}''_i)_1(y''_i)_2$	0 0 0 0 0 0 0 $\overline{1}$ $\overline{1}$
	$(y'''_{i+1})_2 = (y''_i)_1(y''_i)_2$	0 0 $\overline{1}$ 0 $\overline{1}$ 0 0 0 0 0
Parallel Bit Addition Operations	$X_i = (x'''_i)_1(\overline{y}'''_i)_1(\overline{y}'''_i)_2 + (\overline{x}'''_i)_1(\overline{y}'''_i)_1(x'''_i)_2$	0 1 0 1 0 0 1 0 0 0
	$Y_i = (\overline{x}'''_i)_1(y'''_i)_1(\overline{x}'''_i)_2 + (\overline{x}'''_i)_1(\overline{y}'''_i)_1(y'''_i)_2$	0 0 $\overline{1}$ 0 $\overline{1}$ 0 0 0 0 0
	N_w	0 1 $\overline{1}$ 1 $\overline{1}$ 0 1 0 0 0

Figure 63

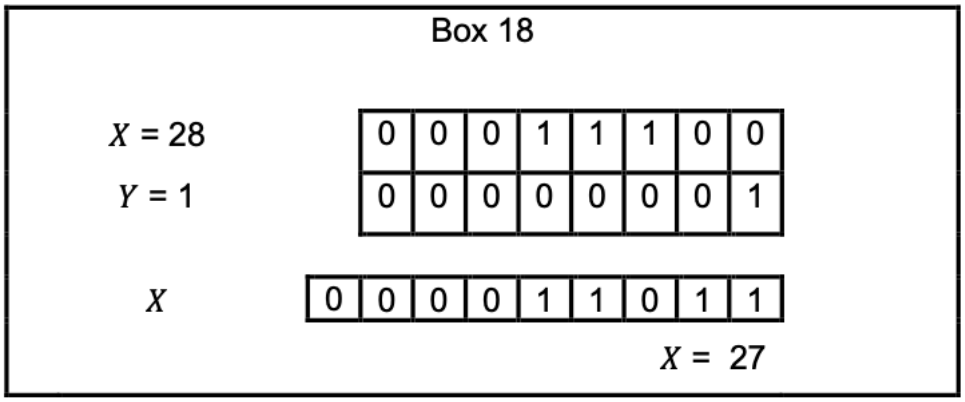


Figure 64

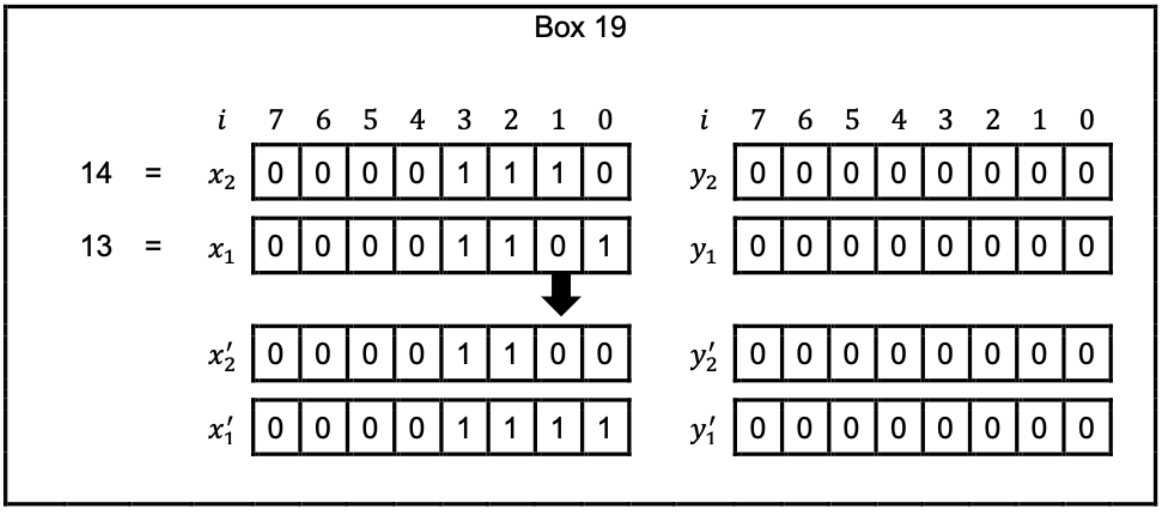


Figure 65

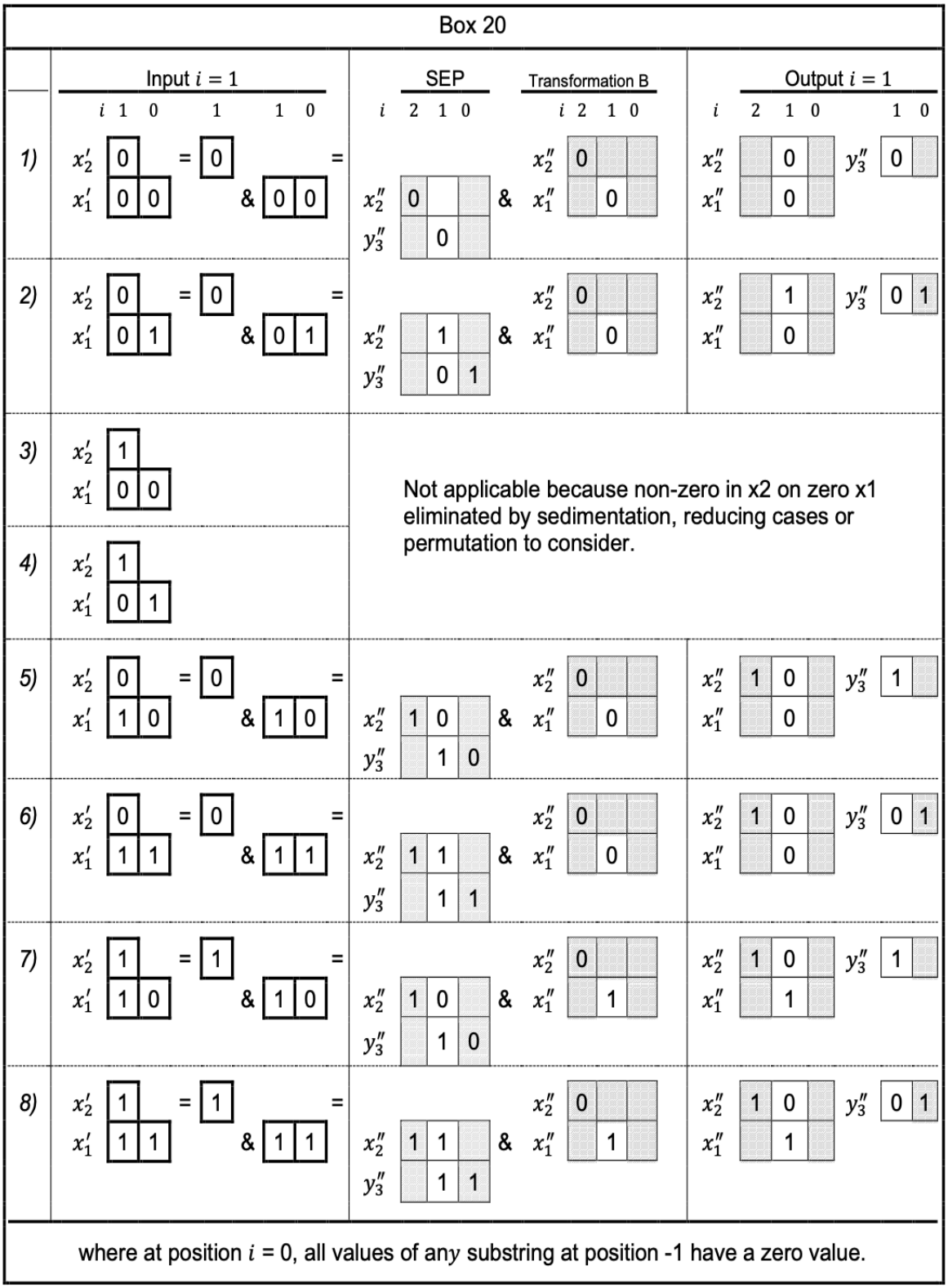

Box 20
Input $i = 1$
SEP
Transformation B
Output $i = 1$
1)
2)
3)
4)
5)
6)
7)
8)
Not applicable because non-zero in x2 on zero x1 eliminated by sedimentation, reducing cases or permutation to consider.
where at position i = 0, all values of any substring at position -1 have a zero value.

Figure 66

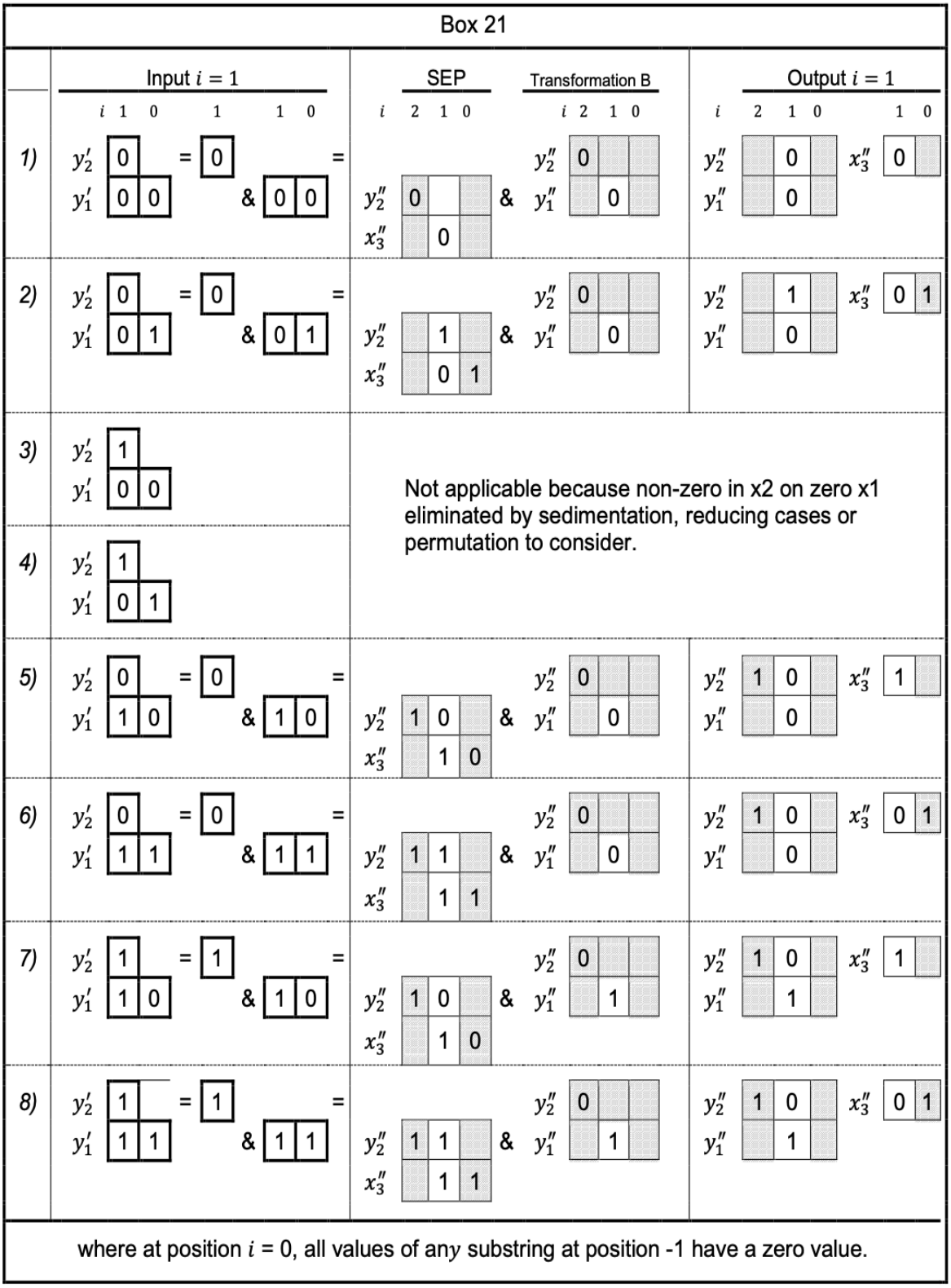

Box 21
Input $i = 1$
SEP
Transformation B
Output $i = 1$
1)
2)
3)
4)
5)
6)
7)
8)
Not applicable because non-zero in x2 on zero x1 eliminated by sedimentation, reducing cases or permutation to consider.
where at position i = 0, all values of any substring at position -1 have a zero value.

Figure 67

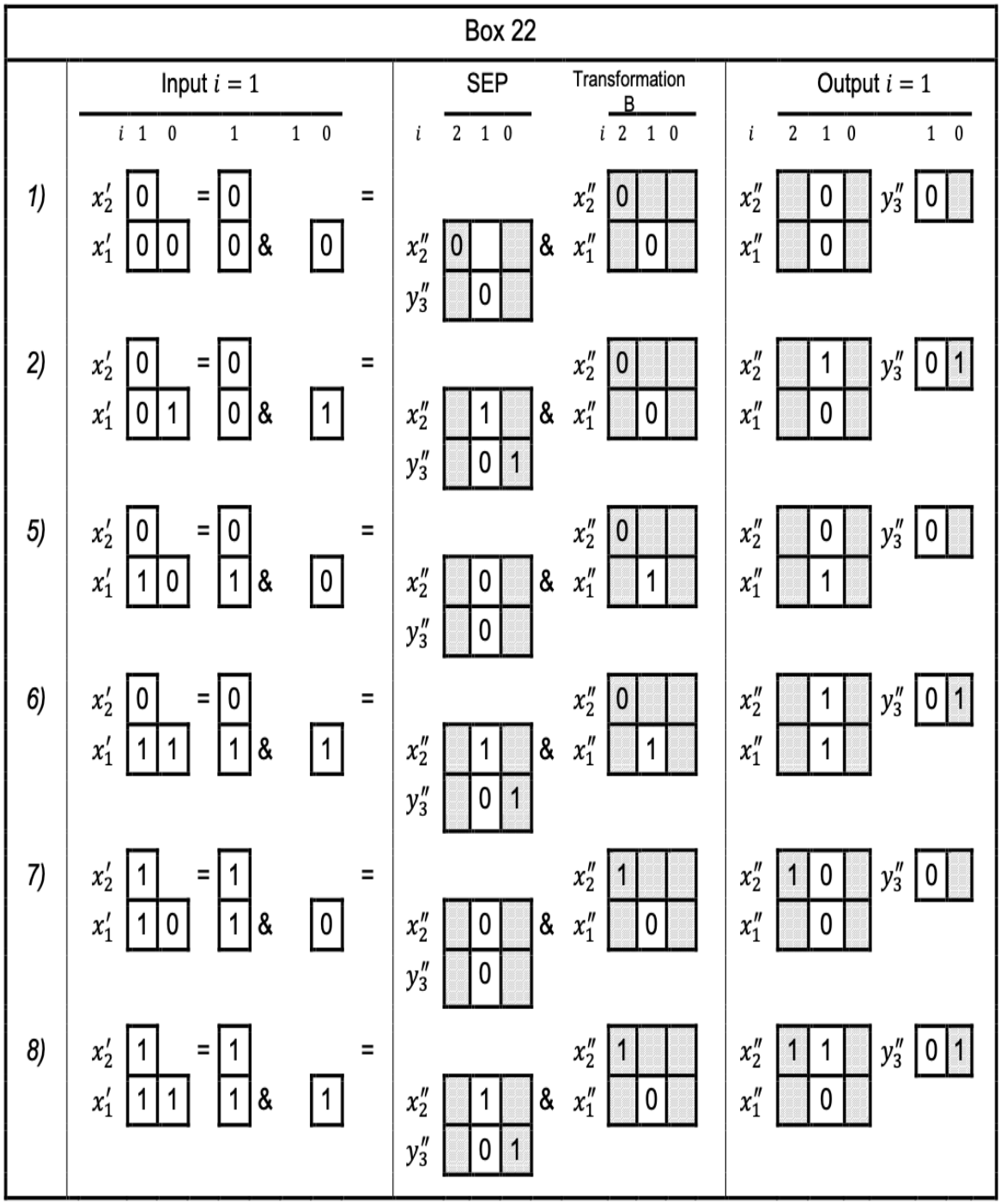


Figure 68

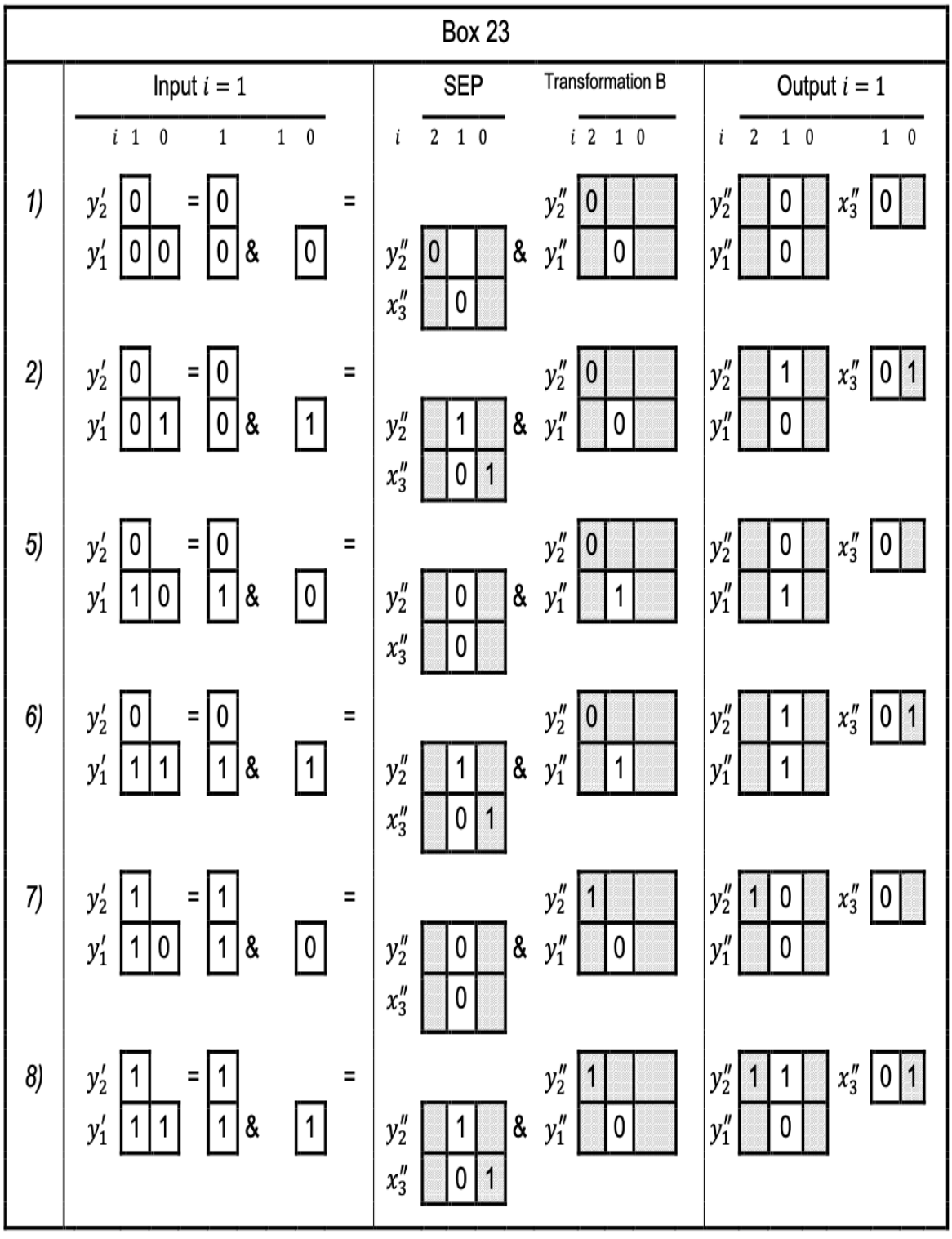


Figure 69

Box 24									
Generalising augend $(\tilde{n}_w)_1$ and addend $(\tilde{n}_w)_2$	$(\tilde{n}_w)_1^+ = (x_i)_1$	1	0	0	1	0	1	0	1
	$(\tilde{n}_w)_2^+ = (x_i)_2$	0	1	0	0	0	1	0	0
	$(\tilde{n}_w)_2^- = (y_i)_2$	0	0	0	0	$\bar{1}$	0	0	$\bar{1}$
	$(\tilde{n}_w)_1^- = (y_i)_1$	0	0	$\bar{1}$	0	$\bar{1}$	0	0	0
Sedimentation	$(x'_i)_2 = (x_i)_1 (x_i)_2$	0	0	0	0	0	1	0	0
	$(x'_i)_1 = (x_i)_1 + (x_i)_2$	1	1	0	1	0	1	0	1
	$(y'_i)_1 = (y_i)_1 + (y_i)_2$	0	0	$\bar{1}$	0	$\bar{1}$	0	0	$\bar{1}$
	$(y'_i)_2 = (y_i)_1 (y_i)_2$	0	0	0	0	$\bar{1}$	0	0	0
Transformation AB	$(x''_i)_1 = (x'_i)_2 (x'_{i-1})_1$	0	0	0	0	0	0	0	0	0
	$(x''_i)_2 = (\bar{x}'_i)_1 (x'_{i-1})_1$	1	0	0	1	0	1	0	1	0
	$(y''_i)_3 = (x'_i)_1 (\bar{x}'_i)_2 (\bar{x}'_{i-1})_1$	0	0	$\bar{1}$	0	$\bar{1}$	0	0	0	$\bar{1}$
	$(y''_i)_1 = (y'_i)_2 (y'_{i-1})_1$	0	0	0	0	0	0	0	0	0
	$(y''_i)_2 = (\bar{y}'_i)_1 (y'_{i-1})_1$	0	0	$\bar{1}$	0	$\bar{1}$	0	0	$\bar{1}$	0
	$(x''_i)_3 = (y'_i)_1 (\bar{y}'_i)_2 (\bar{y}'_{i-1})_1$	0	0	0	1	0	0	0	0	1
Parallel Bit Addition Operations	Eq. (33)	1	0	1	0	0	1	0	1	1
	Eq. (34)	0	$\bar{1}$	0	$\bar{1}$	0	0	0	$\bar{1}$	$\bar{1}$
Simplification of the sum	N_w	1	$\bar{1}$	1	$\bar{1}$	0	1	0	0	0

Figure 70

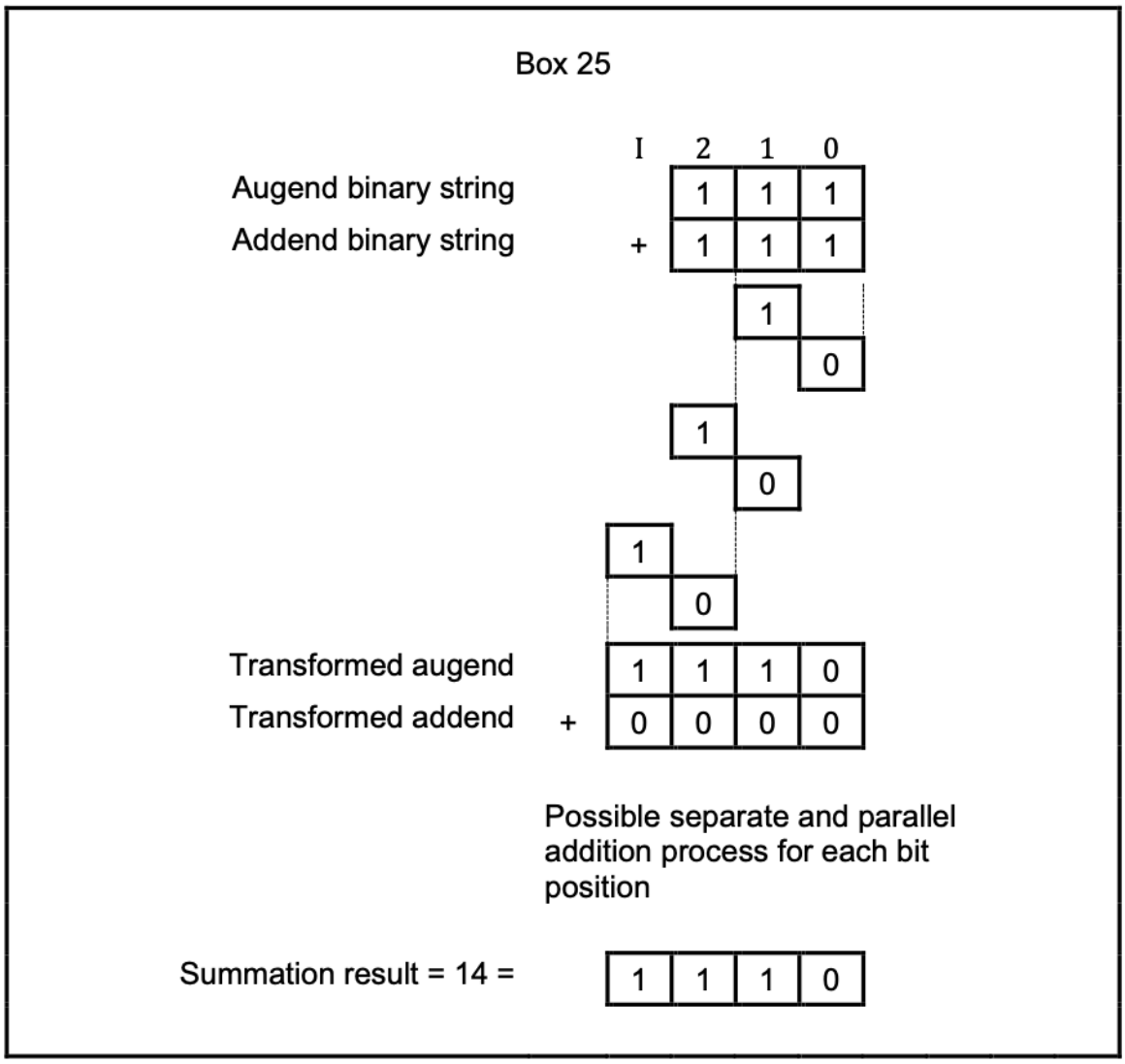

Box 25
I 2 1 0
Augend binary string
Addend binary string
+
1 1 1
1 1 1
1
0
1
0
1
0
Transformed augend
Transformed addend
+
1 1 1 0
0 0 0 0
Possible separate and parallel addition process for each bit position
Summation result = 14 =
1 1 1 0

Figure 71

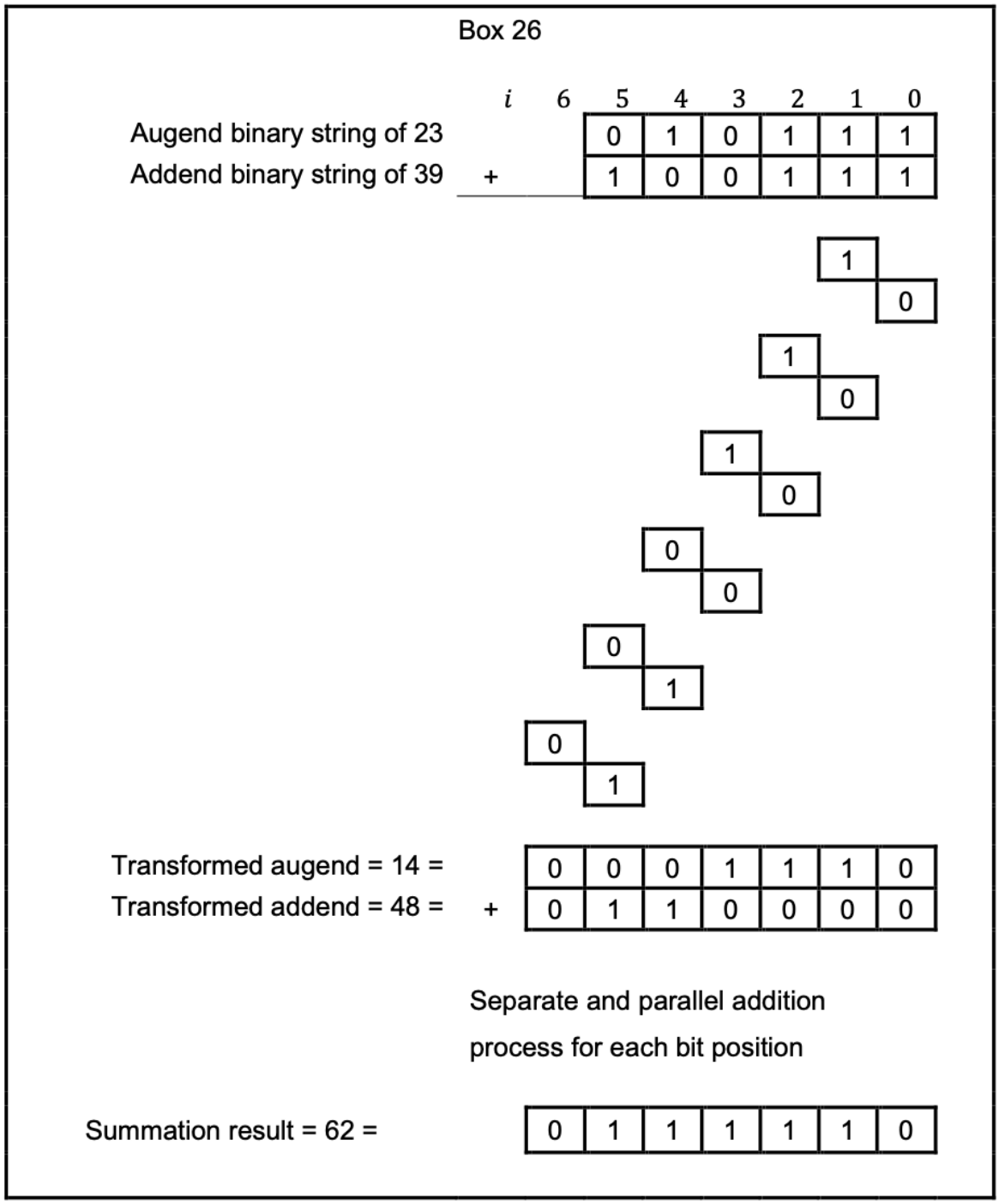

Box 26
i 6 5 4 3 2 1 0
Augend binary string of 23
0 1 0 1 1 1
Addend binary string of 39
+
1 0 0 1 1 1
1 0
1 0
1 0
0 0
0 1
0 1
Transformed augend = 14 =
Transformed addend = 48 =
+
0 0 0 1 1 1 0
0 1 1 0 0 0 0
Separate and parallel addition
process for each bit position
Summation result = 62 =
0 1 1 1 1 1 0

Figure 72

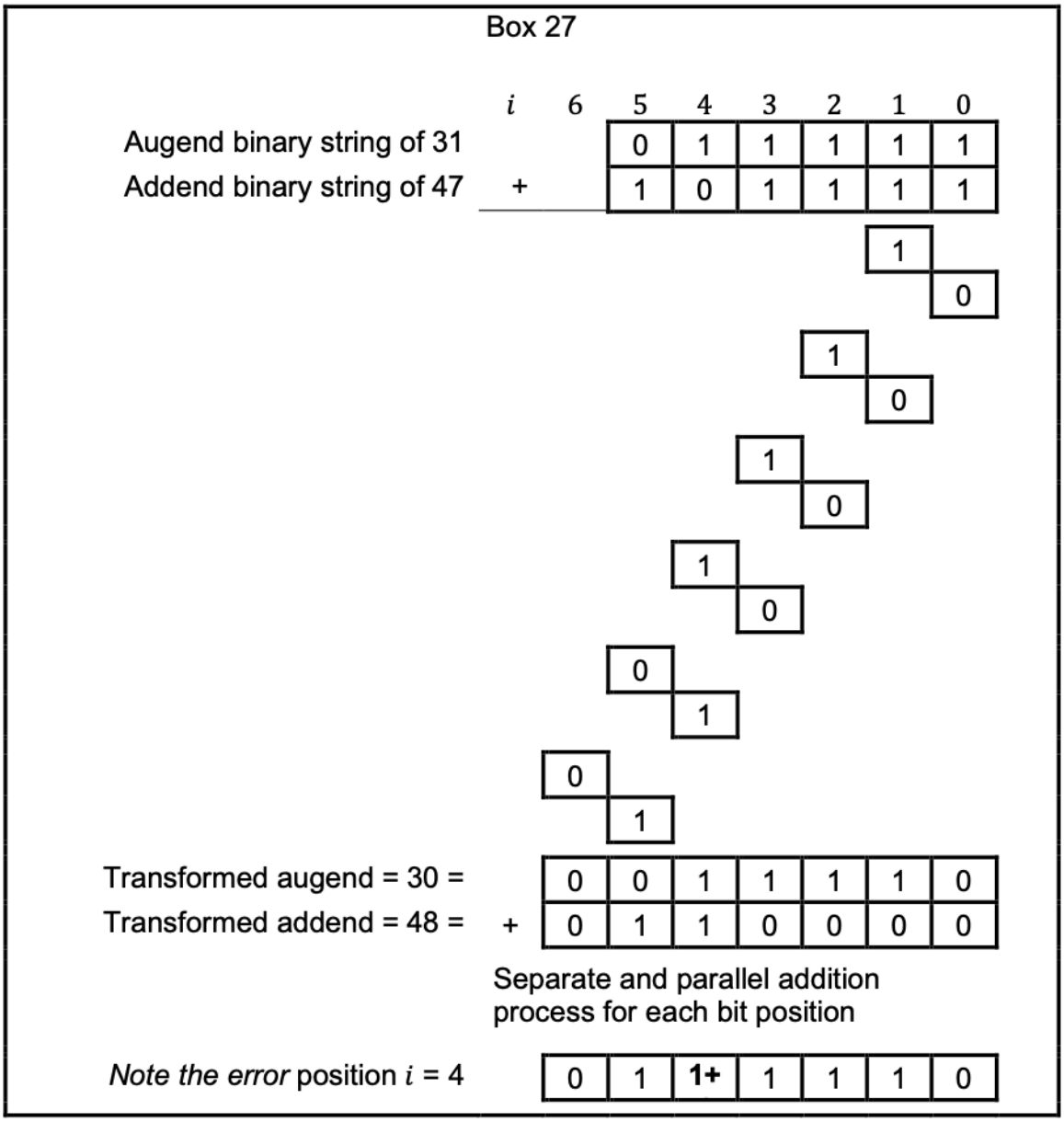

Box 27
i 6 5 4 3 2 1 0
Augend binary string of 31 0 1 1 1 1 1
Addend binary string of 47 + 1 0 1 1 1 1
1
0
1
0
1
0
1
0
0
1
0
1
Transformed augend = 30 = 0 0 1 1 1 1 0
Transformed addend = 48 = + 0 1 1 0 0 0 0
Separate and parallel addition process for each bit position
Note the error position i = 4 0 1 1+ 1 1 1 0

Figure 73

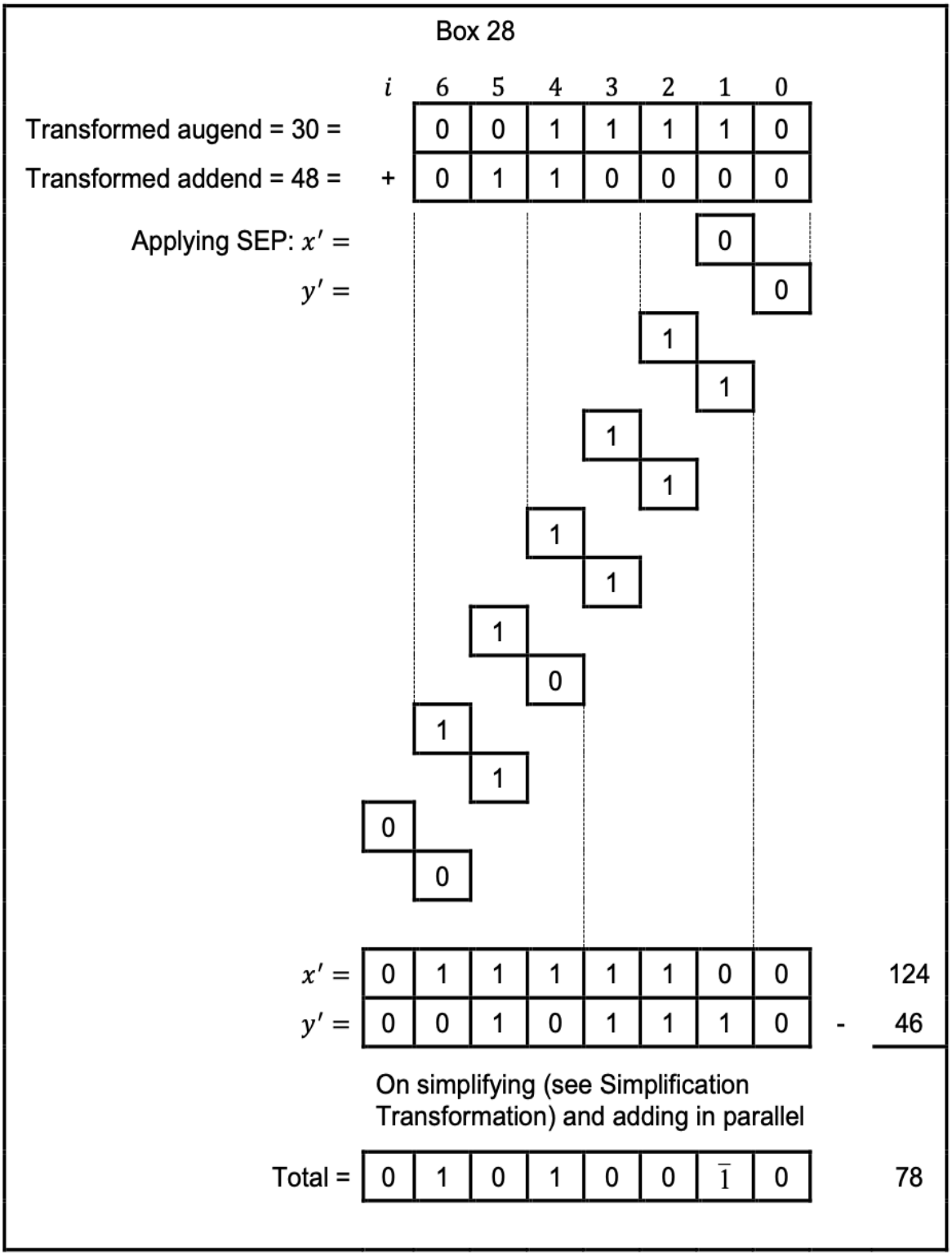

Box 28
i 6 5 4 3 2 1 0
Transformed augend = 30 = 0 0 1 1 1 1 0
Transformed addend = 48 = + 0 1 1 0 0 0 0
Applying SEP: x′ =
y′ =
x′ = 0 1 1 1 1 1 0 0 124
y′ = 0 0 1 0 1 1 1 0 - 46
On simplifying (see Simplification Transformation) and adding in parallel
Total = 0 1 0 1 0 0 1̄ 0 78

Figure 74