Chapter 3

Design Elements (ACC1 to BYPOSC)

This chapter describes design elements included in the Unified Libraries. The elements are organized in alphanumeric order with all numeric suffixes in ascending order.

Information on the specific architectures supported by each of the following libraries is contained under the Applicable Architectures section of the Unified Libraries Chapter.

- **XC3000 Library**
- **XC4000E Library**
- **XC4000X Library**
- **XC5200 Library**
- **XC9000 Library**
- **Spartan Library**
- **SpartanXL Library**
- **Virtex Library**

Note: Wherever *XC4000* is mentioned, the information applies to all architectures supported by the XC4000E and XC4000X libraries.

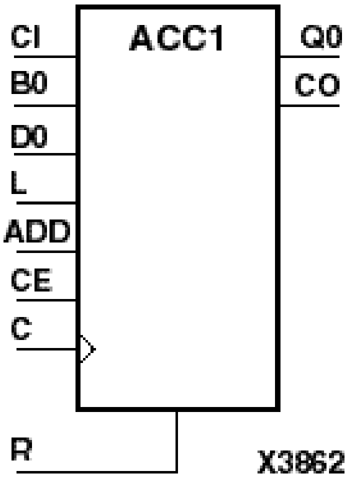
Note: Wherever *Spartans* or *Spartan series* is mentioned, the information applies to all architectures supported by the Spartan and SpartanXL libraries.

Schematics are included for each library if the implementation differs. Design elements with bused or multiple I/O pins (2-, 4-, 8-, 16-bit versions) typically include just one schematic — generally the 8-bit version. When only one schematic is included, implementation of the smaller and larger elements differs only in the number of sections. In cases where an 8-bit version is very large, an appropriate smaller element serves as the schematic example.

ACC1

1-Bit Loadable Cascadable Accumulator with Carry-In, Carry-Out, and Synchronous Reset

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	N/A	N/A	Macro	N/A	N/A	N/A



ACC1 can add or subtract a 1-bit unsigned-binary word to or from the contents of a 1-bit data register and store the results in the register. The register can be loaded with a 1-bit word. The synchronous reset (R) has priority over all other inputs and, when High, causes the output to go to logic level zero during the Low-to-High clock (C) transition. Clock (C) transitions are ignored when clock enable (CE) is Low.

The accumulator is asynchronously cleared, outputs Low, when power is applied. For CPLDs the power-on condition can be simulated by applying a High-level pulse on the PRLD global net.

Load

When the load input (L) is High, CE is ignored and the data on the input D0 is loaded into the 1-bit register during the Low-to-High clock (C) transition.

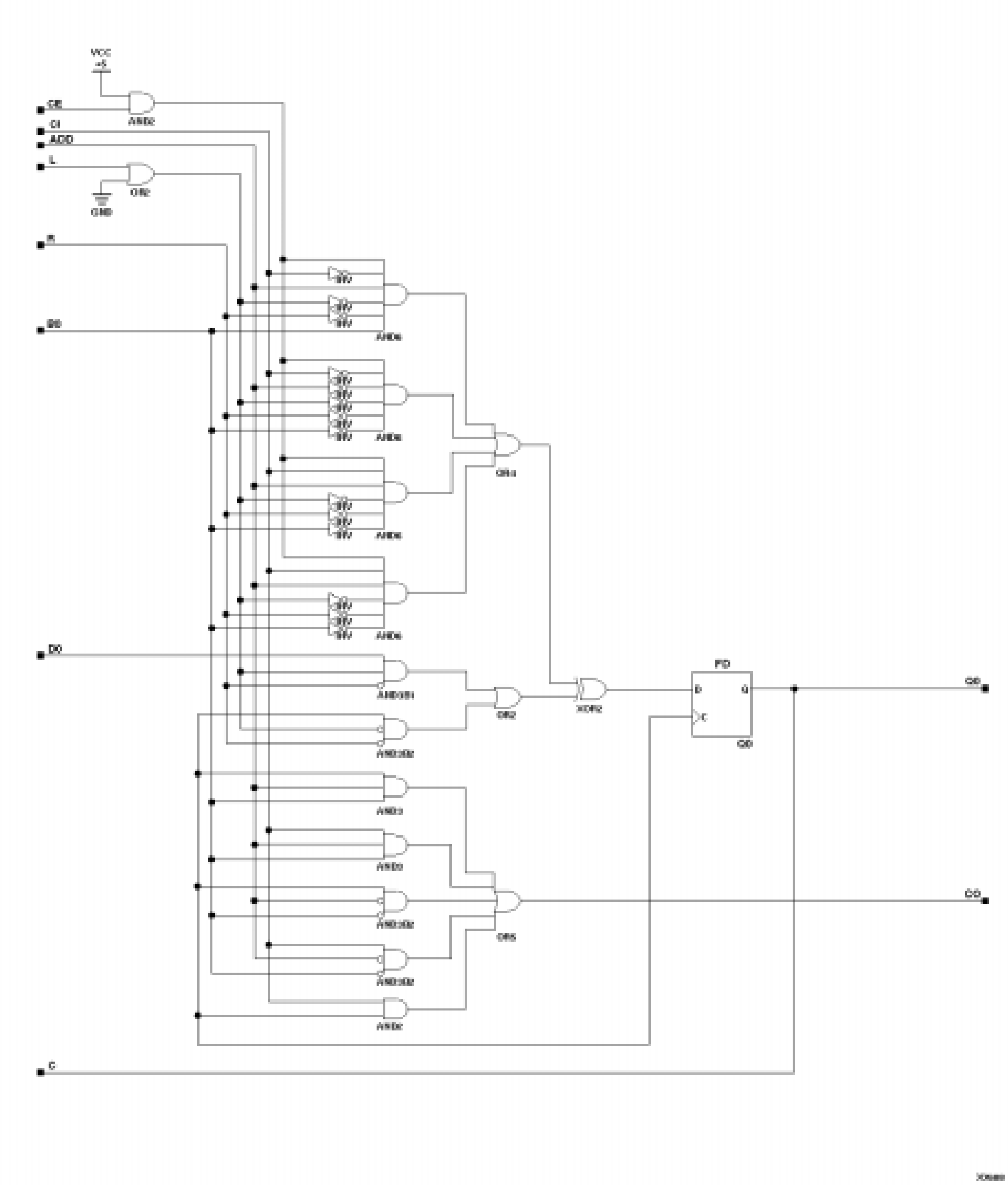
Add

When control inputs ADD and CE are both High, the accumulator adds a 1-bit word (B0) and carry-in (CI) to the contents of the 1-bit register. The result is stored in the register and appears on output Q0 during the Low-to-High clock transition. The carry-out (CO) is not registered synchronously with the data output. CO always reflects the accumulation of input B0 and the contents of the register, which allows cascading of ACC1s by connecting CO of one stage to CI of the next stage. In add mode, CO acts as a carry-out, and CO and CI are active-High.

Subtract

When ADD is Low and CE is High, the 1-bit word B0 and CI are subtracted from the contents of the register. The result is stored in the register and appears on output Q0 during the Low-to-High clock transition. The carry-out (CO) is not registered synchronously with the data output. CO always reflects the accumulation of input B0 and the contents of the register, which allows cascading of ACC1s by connecting CO of one stage to CI of the next stage. In subtract mode, CO acts as a borrow, and CO and CI are active-Low.

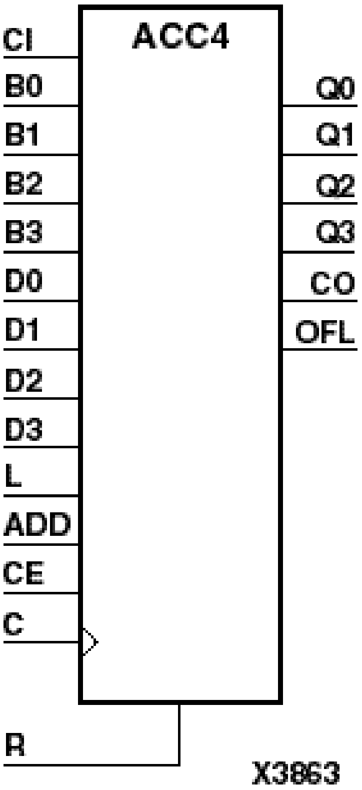
Figure 3-1ACC1 Implementation XC9000

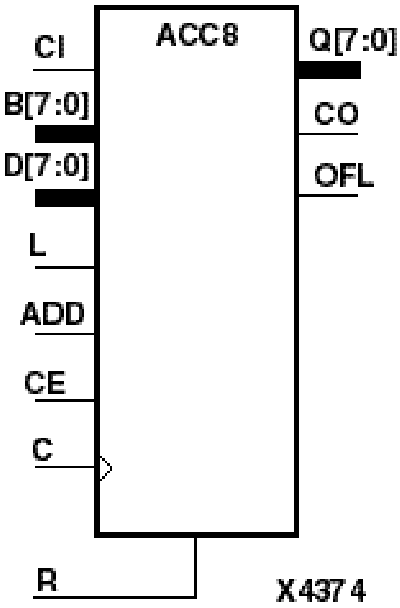


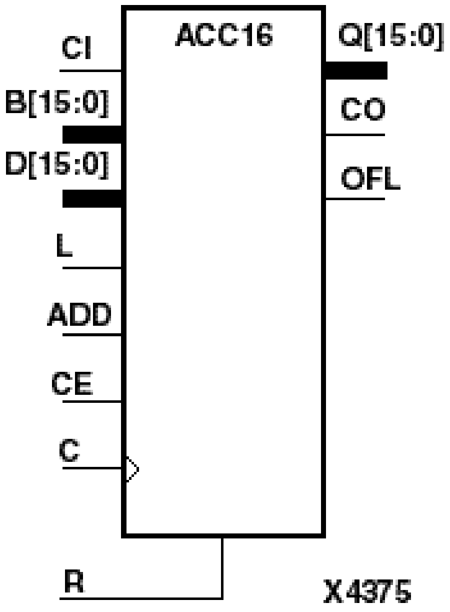
ACC4, 8, 16

4-, 8-, 16-Bit Loadable Cascadable Accumulators with Carry-In, Carry-Out, and Synchronous Reset

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
Macro	Macro	Macro	Macro	Macro	Macro	Macro	Macro







ACC4, ACC8, ACC16 can add or subtract a 4-, 8-, 16-bit unsigned-binary, respectively or twos-complement word to or from the contents of a 4-, 8-, 16-bit data register and store the results in the register. The register can be loaded with the 4-, 8-, 16-bit word.

In the XC4000 and Spartans, these accumulators are implemented using carry logic and relative location constraints, which assure most efficient logic placement.

The synchronous reset (R) has priority over all other inputs, and when High, causes all outputs to go to logic level zero during the Low-to-High clock (C) transition. Clock (C) transitions are ignored when clock enable (CE) is Low.

The accumulator is asynchronously cleared, outputs Low, when power is applied. For CPLDs, the power-on condition can be simulated by applying a High-level pulse on the PRLD global net. FPGAs simulate power-on when global reset (GR) or global set/reset (GSR) is active. GR for XC3000 is active-Low. GR for XC5200 and GSR (XC4000, Spartans, Virtex) default to active-High but can be inverted by adding an inverter in front of the GR/GSR input of the STARTUP or STARTUP_VIRTEX symbol.

Load

When the load input (L) is High, CE is ignored and the data on the D inputs is loaded into the register during the Low-to-High clock (C) transition. ACC4 loads the data on inputs D3 – D0 into the 4-bit register. ACC8 loads the data on D7 – D0 into the 8-bit register. ACC16 loads the data on inputs D15 – D0 into the 16-bit register.

Unsigned Binary Versus Twos Complement

ACC4, ACC8, ACC16 can operate, respectively, on either 4-, 8-, 16-bit unsigned binary numbers or 4-, 8-, 16-bit twos-complement numbers. If the inputs are interpreted as unsigned binary, the result can be interpreted as unsigned binary. If the inputs are interpreted as twos complement, the output can be interpreted as twos complement. The only functional difference between an unsigned binary operation and a twos-complement operation is how they determine when "overflow" occurs. Unsigned binary uses CO, while twos complement uses OFL to determine when "overflow" occurs.

Unsigned Binary Operation

For unsigned binary operation, ACC4 can represent numbers between 0 and 15, inclusive; ACC8 between 0 and 255, inclusive; and ACC16 between 0 and 65535, inclusive. In add mode, CO is active (High) when the sum exceeds the bounds of the adder/subtracter. In subtract mode, CO is an active-Low borrow-out and goes Low when the difference exceeds the bounds. The carry-out (CO) is not registered synchronously with the data outputs. CO always reflects the accumulation of the B inputs (B3 – B0 for ACC4, B7 – B0 for ACC8, B15 – B0 for ACC16) and the contents of the register, which allows cascading of ACC4s, ACC8s, or ACC16s by connecting CO of one stage to CI of the next stage. An unsigned binary "overflow" that is always active-High can be generated by gating the ADD signal and CO as follows.

```
unsigned overflow = CO XOR ADD
```

OFL should be ignored in unsigned binary operation.

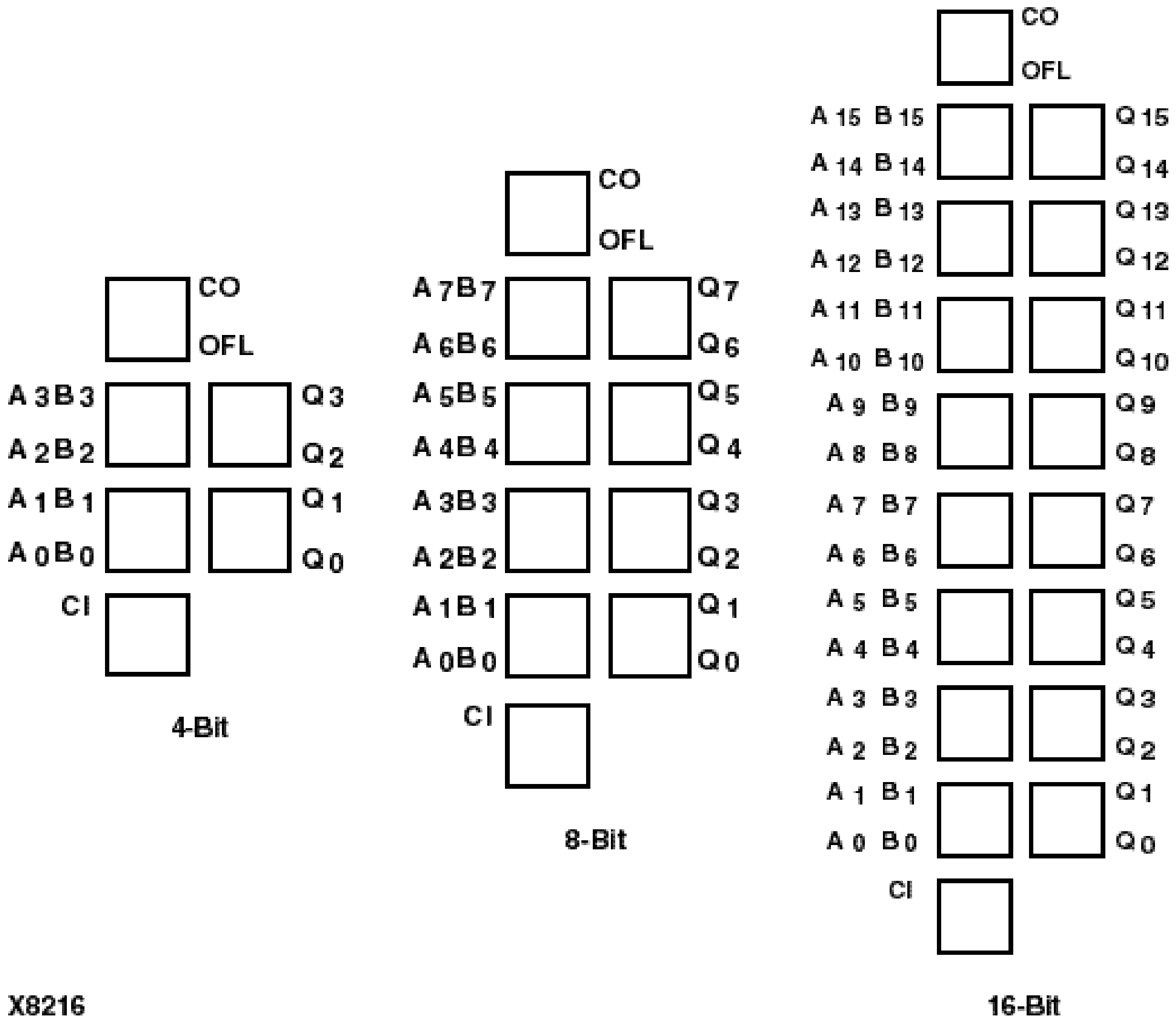
Twos-Complement Operation

For twos-complement operation, ACC4 can represent numbers between -8 and +7, inclusive; ACC8 between -128 and +127, inclusive; ACC16 between -32768 and +32767, inclusive. If an addition or subtraction operation result exceeds this range, the OFL output goes High. The overflow (OFL) is not registered synchronously with the data outputs. OFL always reflects the accumulation of the B inputs (B3 – B0 for ACC4, B7 – B0 for ACC8, B15 – B0 for ACC16) and the contents of the register, which allows cascading of ACC4s, ACC8s, or ACC16s by connecting OFL of one stage to CI of the next stage.

CO should be ignored in twos-complement operation.

Topology for XC4000 and Spartans

This is the ACC4 (4-bit), ACC8 (8-bit), and ACC16 (16-bit) topology for XC4000 and Spartan series devices.



Topology for XC5200

This is the ACC8 (8-bit) and ACC16 (16-bit) topology for XC5200 devices.

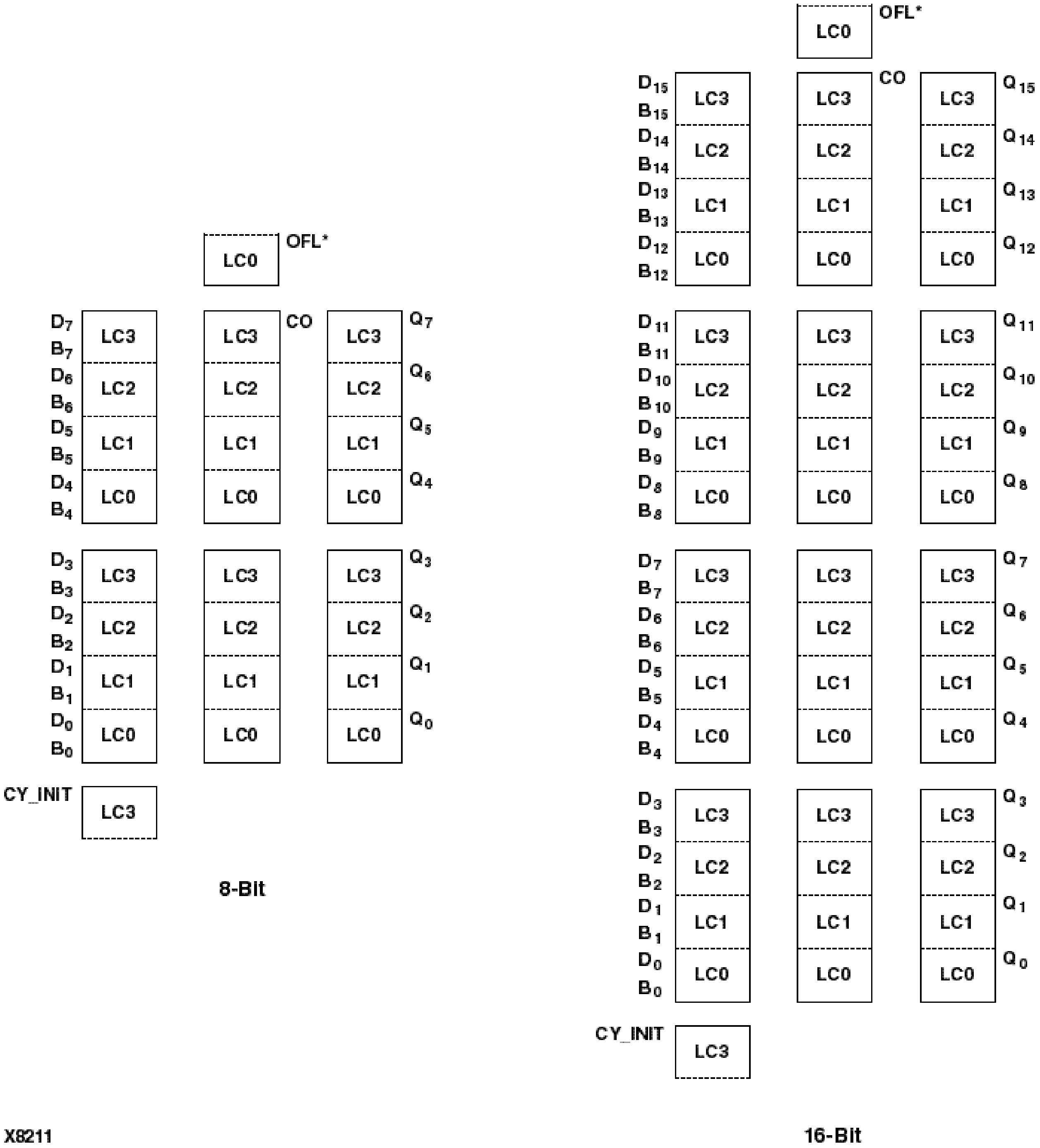


Figure 3-2ACC8 Implementation XC3000

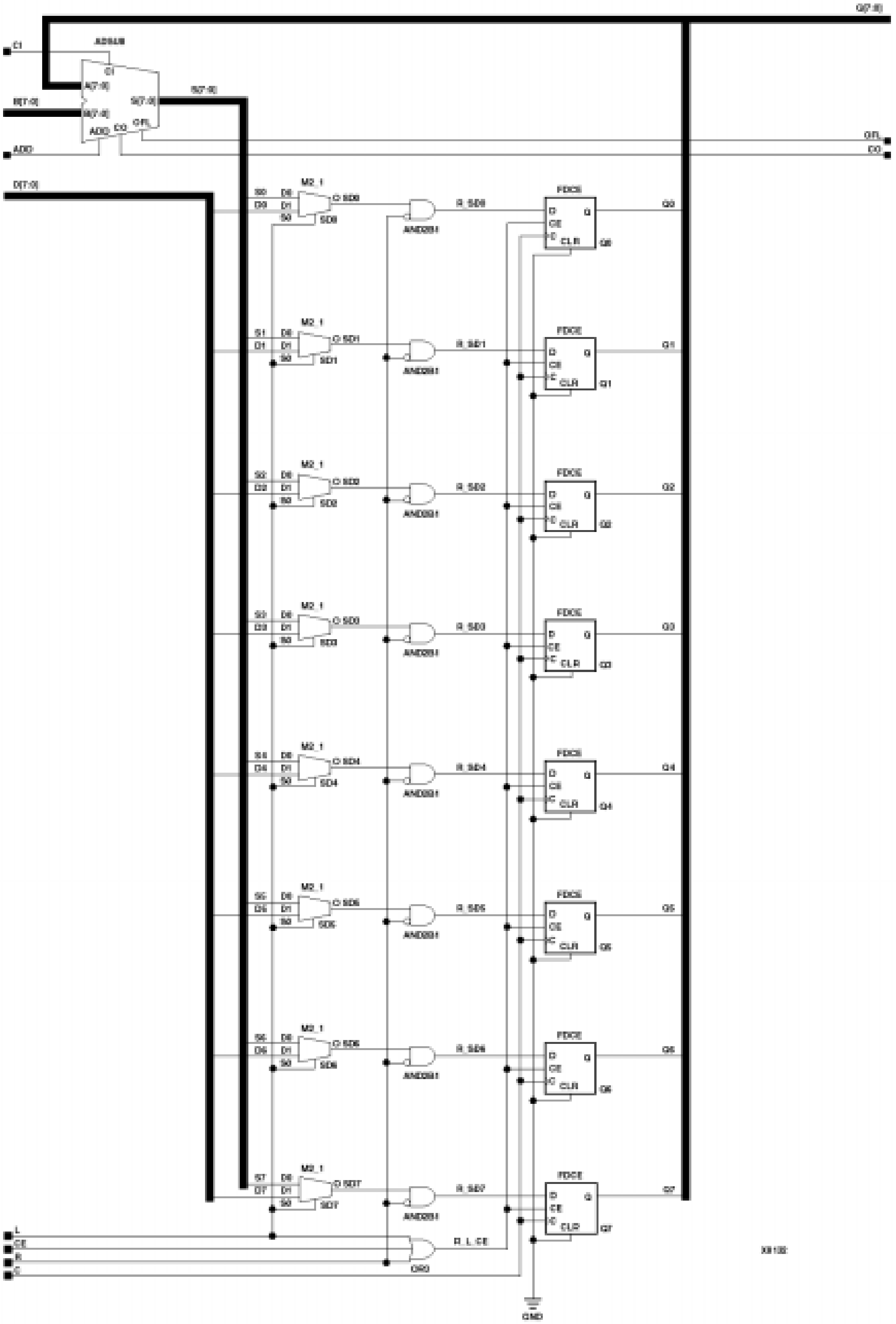

Q[7:0]
CI
ADSUB
CI
A[7:0]
B[7:0]
B[7:0]
S[7:0]
S[7:0]
ADD
CO
OFL
ADD
OFL
CO
D[7:0]
M2_1
S0
D0
D0
D1
S0
O SD0
SD0
AND2B1
R_SD0
FDCE
D
Q
CE
C
CLR
Q0
Q0
S1
D1
O SD1
SD1
R_SD1
Q1
S2
D2
O SD2
SD2
R_SD2
Q2
S3
D3
O SD3
SD3
R_SD3
Q3
S4
D4
O SD4
SD4
R_SD4
Q4
S5
D5
O SD5
SD5
R_SD5
Q5
S6
D6
O SD6
SD6
R_SD6
Q6
S7
D7
O SD7
SD7
R_SD7
Q7
L
CE
R
C
R_L_CE
OR3
GND

Figure 3-3ACC8 Implementation XC4000, Spartans

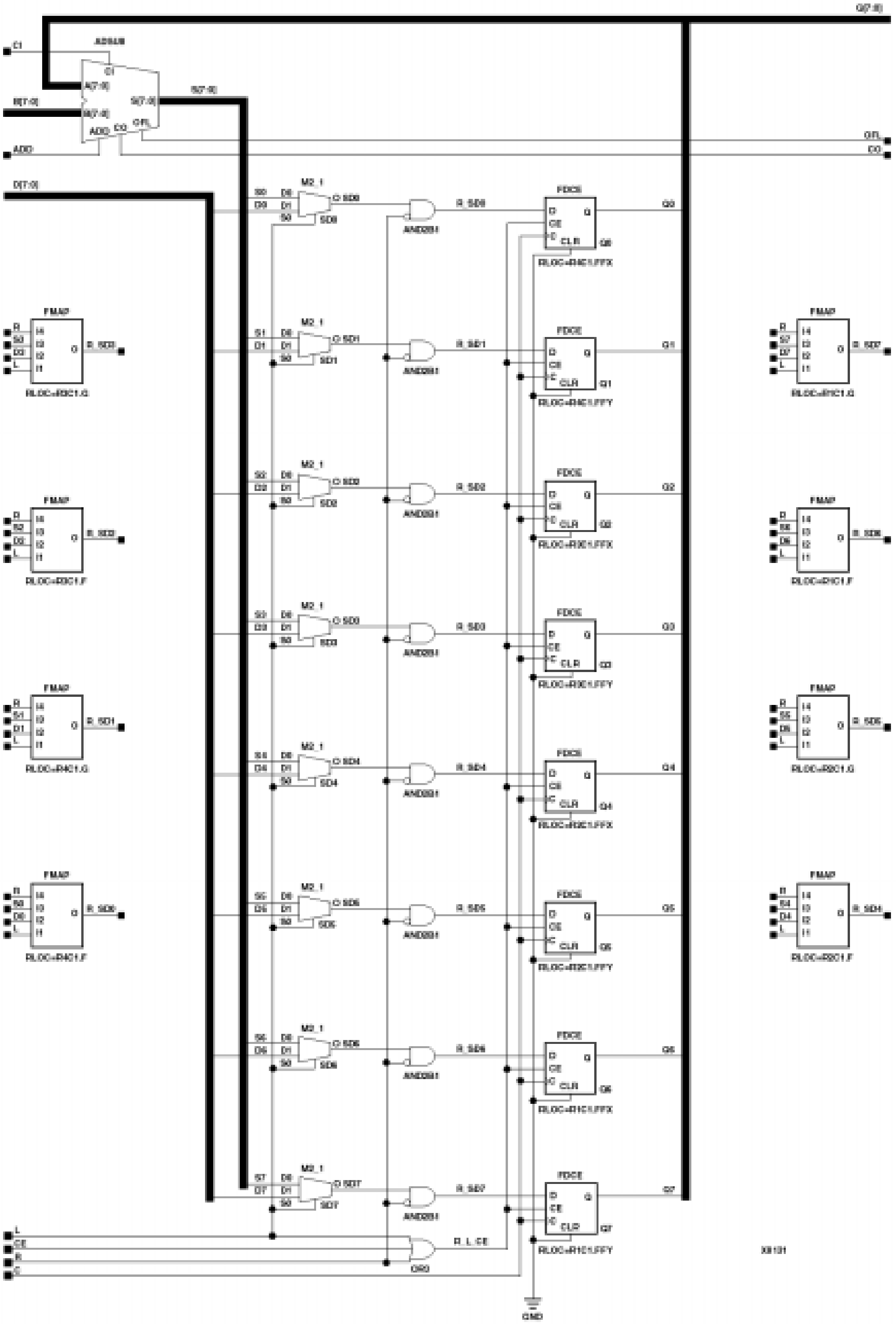

Q[7:0]
CI
ADSU8
CI
A[7:0]
B[7:0]
B[7:0]
S[7:0]
S[7:0]
ADD
CO
OFL
ADD
OFL
CO
D[7:0]
M2_1
S0
D0
D0
D1
S0
O SD0
SD0
AND2B1
R_SD0
FDCE
D
Q
CE
C
CLR
Q0
Q0
RLOC=R4C1.FFX
FMAP
R
S3
D3
L
I4
I3
I2
I1
O
R_SD3
RLOC=R3C1.G
M2_1
S1
D1
O SD1
SD1
R_SD1
FDCE
Q1
RLOC=R4C1.FFY
FMAP
R
S7
D7
L
R_SD7
RLOC=R1C1.G
M2_1
S2
D2
O SD2
SD2
R_SD2
FDCE
Q2
RLOC=R3C1.FFX
FMAP
R
S2
D2
L
R_SD2
RLOC=R3C1.F
FMAP
R
S6
D6
L
R_SD6
RLOC=R1C1.F
M2_1
S3
D3
O SD3
SD3
R_SD3
FDCE
Q3
RLOC=R3C1.FFY
FMAP
R
S1
D1
L
R_SD1
RLOC=R4C1.G
FMAP
R
S5
D5
L
R_SD5
RLOC=R2C1.G
M2_1
S4
D4
O SD4
SD4
R_SD4
FDCE
Q4
RLOC=R2C1.FFX
FMAP
R
S0
D0
L
R_SD0
RLOC=R4C1.F
FMAP
R
S4
D4
L
R_SD4
RLOC=R2C1.F
M2_1
S5
D5
O SD5
SD5
R_SD5
FDCE
Q5
RLOC=R2C1.FFY
M2_1
S6
D6
O SD6
SD6
R_SD6
FDCE
Q6
RLOC=R1C1.FFX
M2_1
S7
D7
O SD7
SD7
R_SD7
FDCE
Q7
RLOC=R1C1.FFY
L
CE
R
C
R_L_CE
OR3
GND
X8101

Figure 3-4ACC8 Implementation XC5200

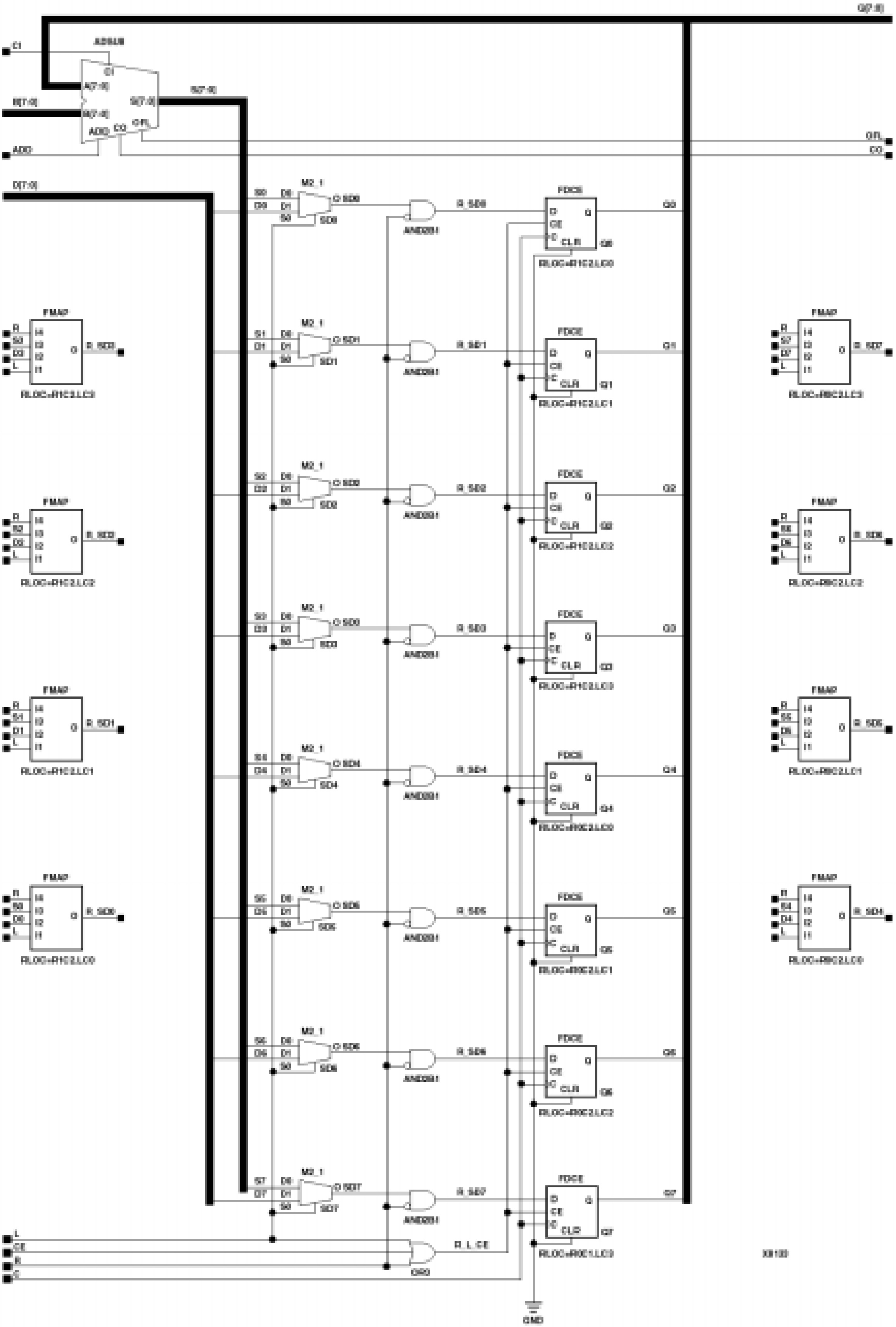

Q[7:0]
CI
ADSU8
A[7:0]
B[7:0]
S[7:0]
ADD
CO
OFL
D[7:0]
M2_1
D0
D1
S0
O SD0
AND2B1
R_SD0
FDCE
D
Q
CE
C
CLR
Q0
RLOC=R1C2.LC0
FMAP
I4
I3
I2
I1
O
R_SD3
RLOC=R1C2.LC3
R_SD2
RLOC=R1C2.LC2
R_SD1
RLOC=R1C2.LC1
R_SD0
RLOC=R1C2.LC0
R_SD7
RLOC=R0C2.LC3
R_SD6
RLOC=R0C2.LC2
R_SD5
RLOC=R0C2.LC1
R_SD4
RLOC=R0C2.LC0
Q1
RLOC=R1C2.LC1
Q2
RLOC=R1C2.LC2
Q3
RLOC=R1C2.LC3
Q4
RLOC=R0C2.LC0
Q5
RLOC=R0C2.LC1
Q6
RLOC=R0C2.LC2
Q7
RLOC=R0C1.LC3
L
CE
R
C
R_L_CE
OR3
GND
X8100

Figure 3-5ACC8 Implementation Virtex

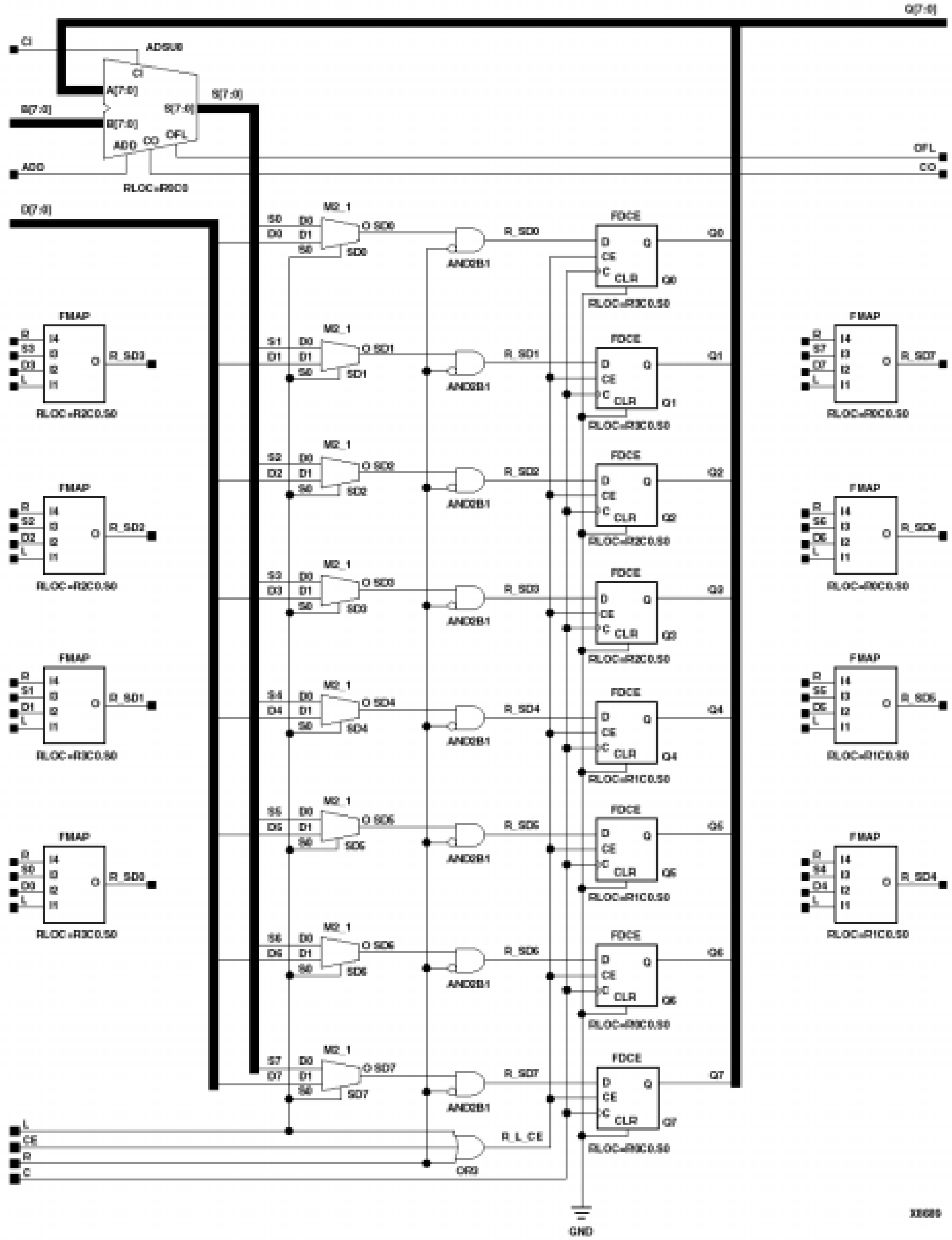


Figure 3-6ACC4 Implementation XC9000

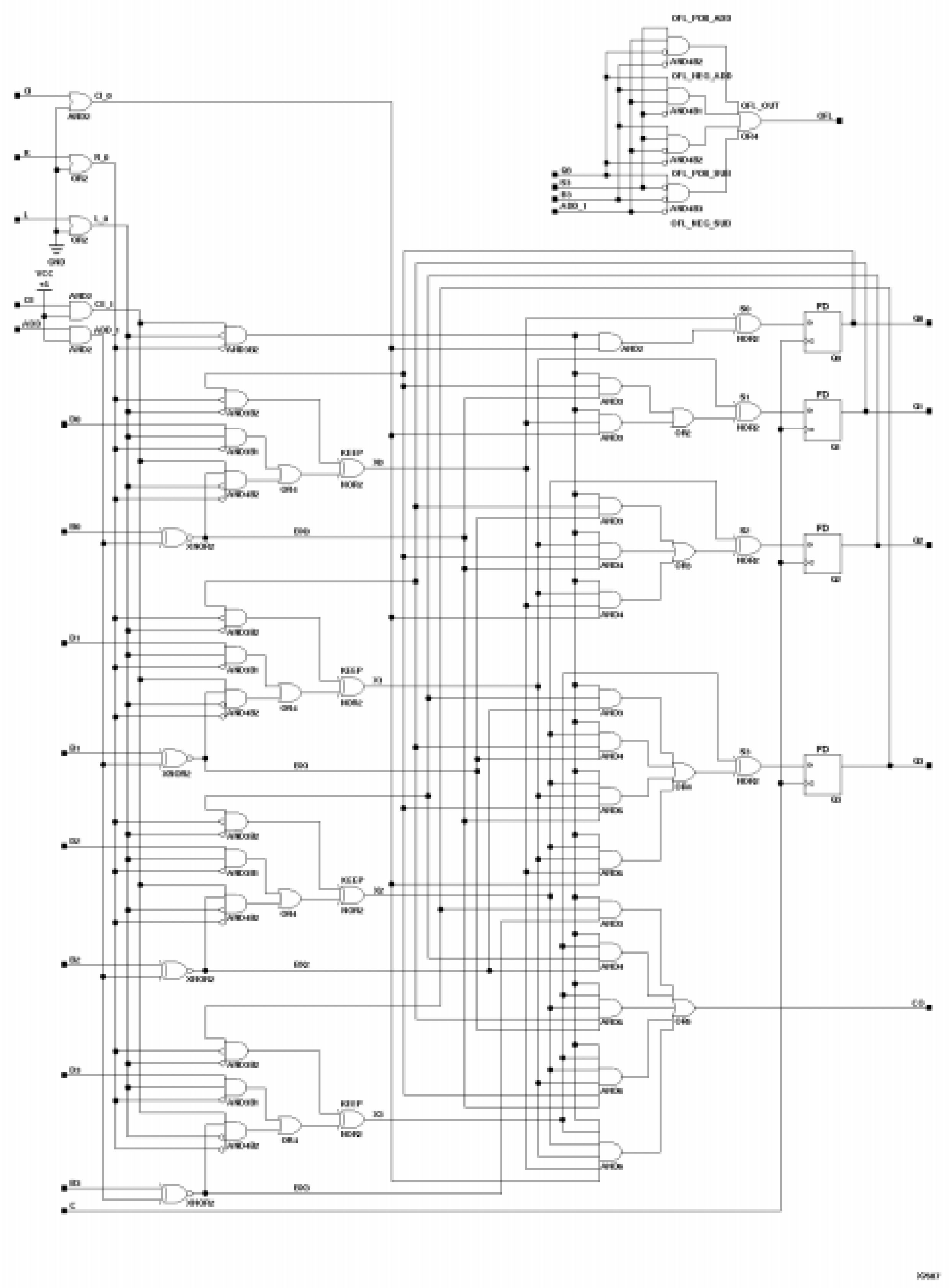
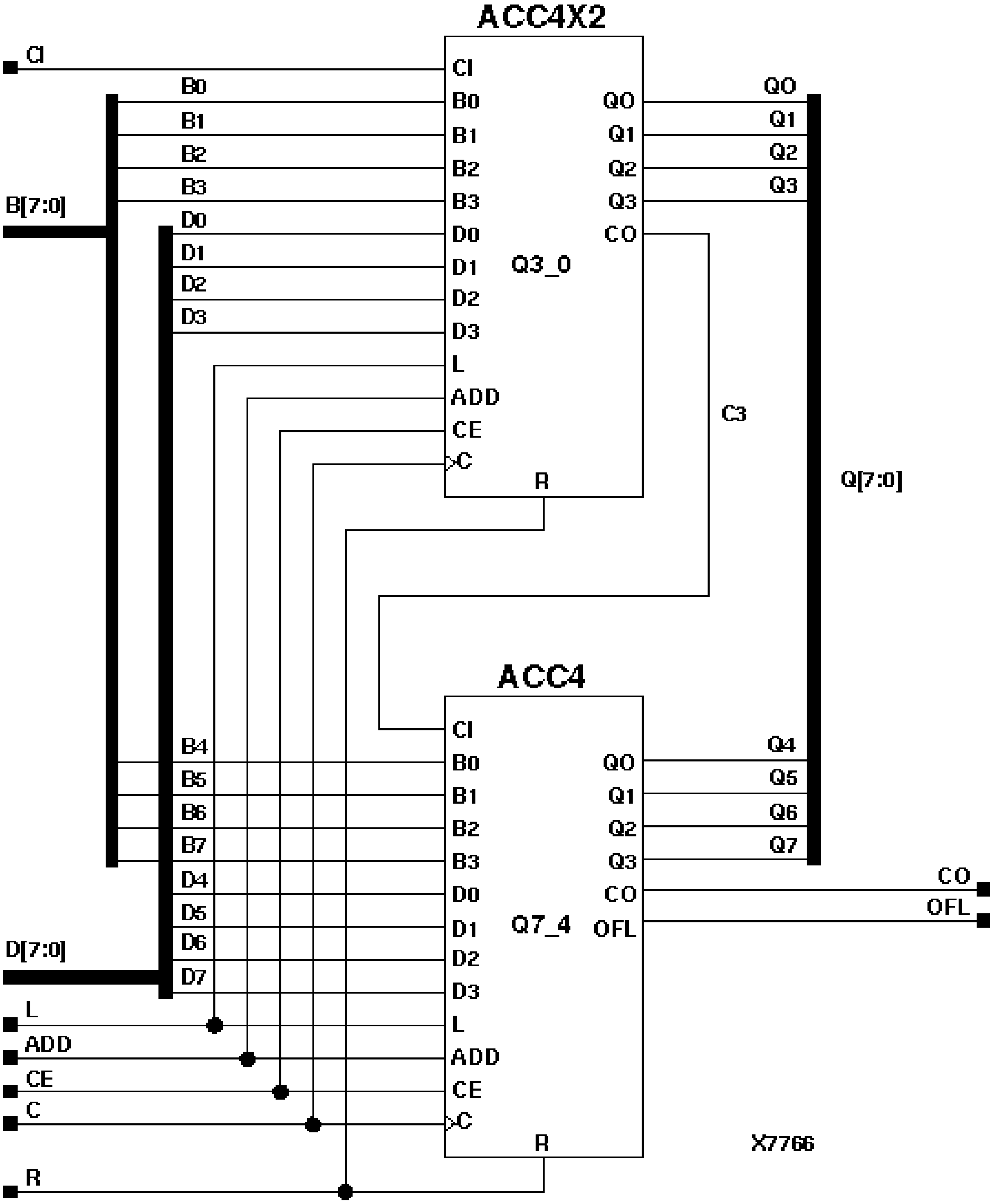


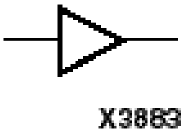
Figure 3-7ACC8 Implementation XC9000



ACLK

Alternate Clock Buffer

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
Primitive	N/A	N/A	N/A	N/A	N/A	N/A	N/A



ACLK, the alternate clock buffer, is used to distribute high fan-out clock signals throughout a PLD device. One ACLK buffer on each device provides direct access to every Configurable Logic Block (CLB) and Input Output Block (IOB) clock pin. The ACLK buffer is slightly slower than the global clock buffer (GCLK) but otherwise similar. Unlike GCLK, the routing resources used for the ACLK network can be used to route other signals if it is not used. For this reason, if only one of the GCLK and ACLK buffers is used, GCLK is preferred. The ACLK input (I) can come from one of the following sources.

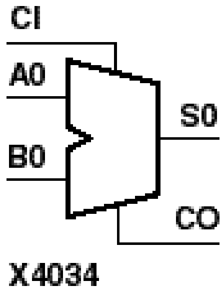
- A CMOS-level signal on the dedicated BCLKIN pin. BCLKIN is a direct CMOS-only input to the ACLK buffer. To use the BCLKIN pin, connect the input of the ACLK element to IBUF and IPAD elements.
- A CMOS- or TTL-level external signal. To connect an external input to the ACLK buffer, connect the input of the ACLK element to the output of the IBUF for that signal. Unless the corresponding IPAD element is constrained otherwise, PAR typically places that IOB directly adjacent to the ACLK buffer.
- The on-chip crystal oscillator. The output of the XTAL oscillator on XC3000 devices is directly adjacent to the ACLK buffer input. If the GXTL element is used, the output of the XTAL oscillator is automatically connected to the ACLK buffer; do not use the ACLK element for anything else.
- An internal signal. To drive the ACLK buffer with an internal signal, connect that signal directly to the input of the ACLK element.

For a negative-edge clock, insert an INV (inverter) element between the ACLK output and the clock input. Inversion is performed inside the CLB, or in the case of IOB clock pins, on the IOB clock line (that controls the clock sense for the IOBs on an entire edge of the chip).

ADD1

1-Bit Full Adder with Carry-In and Carry-Out

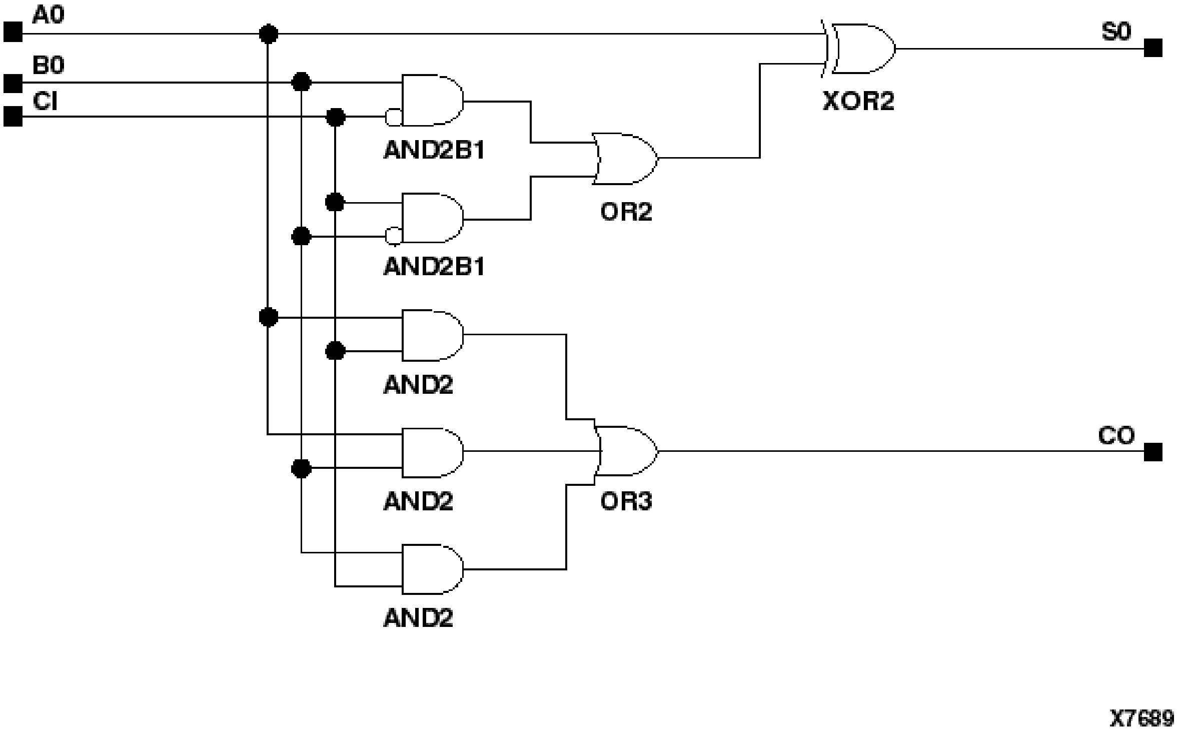
XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	N/A	N/A	Macro	N/A	N/A	N/A



ADD1, a cascadable 1-bit full adder with carry-in and carry-out, adds two 1-bit words (A and B) and a carry-in (CI), producing a binary sum (S0) output and a carry-out (CO).

Inputs			**Outputs**	
A0	**B0**	**CI**	**S0**	**CO**
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	0	1
0	0	1	1	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

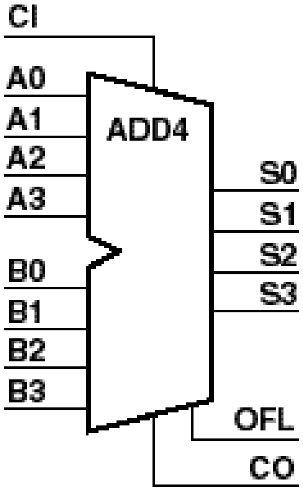
Figure 3-8ADD1 Implementation XC9000



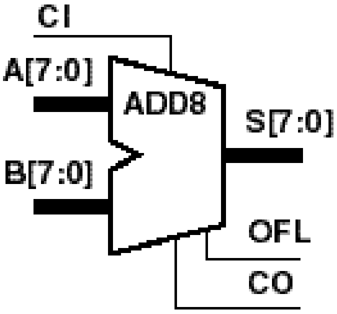
ADD4, 8, 16

4-, 8-, 16-Bit Cascadable Full Adders with Carry-In, Carry-Out, and Overflow

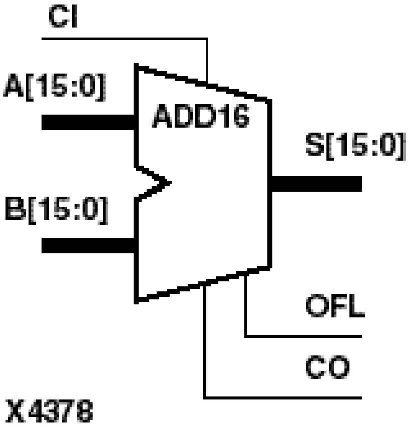
XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
Macro	Macro	Macro	Macro	Macro	Macro	Macro	Macro



X4376



X4377



X4378

ADD4, ADD8, and ADD16 add two words and a carry-in (CI), producing a sum output and carry-out (CO) or overflow (OFL). ADD4 adds A3 – A0, B3 – B0, and CI producing the sum output S3 – S0 and CO (or OFL). ADD8 adds A7 – A0, B7 – B0, and CI, producing the sum output S7 – S0 and CO (or OFL). ADD16 adds A15 – A0, B15 – B0 and CI, producing the sum output S15 – S0 and CO (or OFL).

ADD4, ADD8, and ADD16 are implemented in the XC4000 and Spartans using carry logic and relative location constraints, which assure most efficient logic placement.

Unsigned Binary Versus Twos Complement

ADD4, ADD8, ADD16 can operate on either 4-, 8-, 16-bit unsigned binary numbers or 4-, 8-, 16-bit twos-complement numbers, respectively. If the inputs are interpreted as unsigned binary, the result can be interpreted as unsigned binary. If the inputs are interpreted as twos complement, the output can be interpreted as twos complement. The only functional difference between an unsigned binary operation and a twos-complement operation is how they determine when "overflow" occurs. Unsigned binary uses CO, while twos-complement uses OFL to determine when "overflow" occurs.

Unsigned Binary Operation

For unsigned binary operation, ADD4 can represent numbers between 0 and 15, inclusive; ADD8 between 0 and 255, inclusive; ADD16 between 0 and 65535, inclusive. CO is active (High) when the sum exceeds the bounds of the adder.

OFL is ignored in unsigned binary operation.

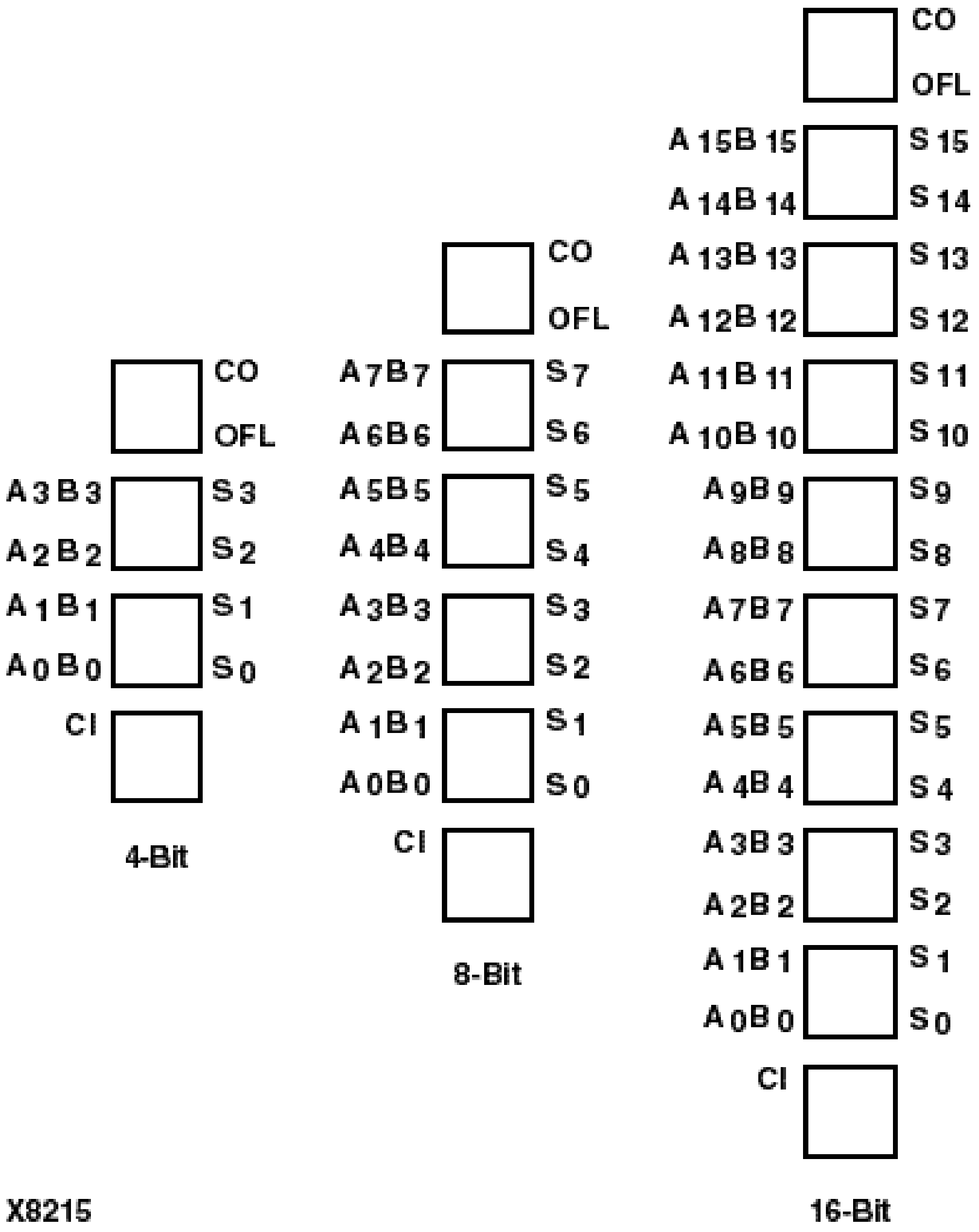
Twos-Complement Operation

For twos-complement operation, ADD4 can represent numbers between -8 and +7, inclusive; ADD8 between -128 and +127, inclusive; ADD16 between -32768 and +32767, inclusive. OFL is active (High) when the sum exceeds the bounds of the adder.

CO is ignored in twos-complement operation.

Topology for XC4000 and Spartans

This is the ADD4 (4-bit), ADD8 (8-bit), and ADD16 (16-bit) topology for XC4000 and Spartan series devices.



Topology for XC5200

This is the ADD8 (8-bit) and ADD16 (16-bit) topology for XC5200 devices.

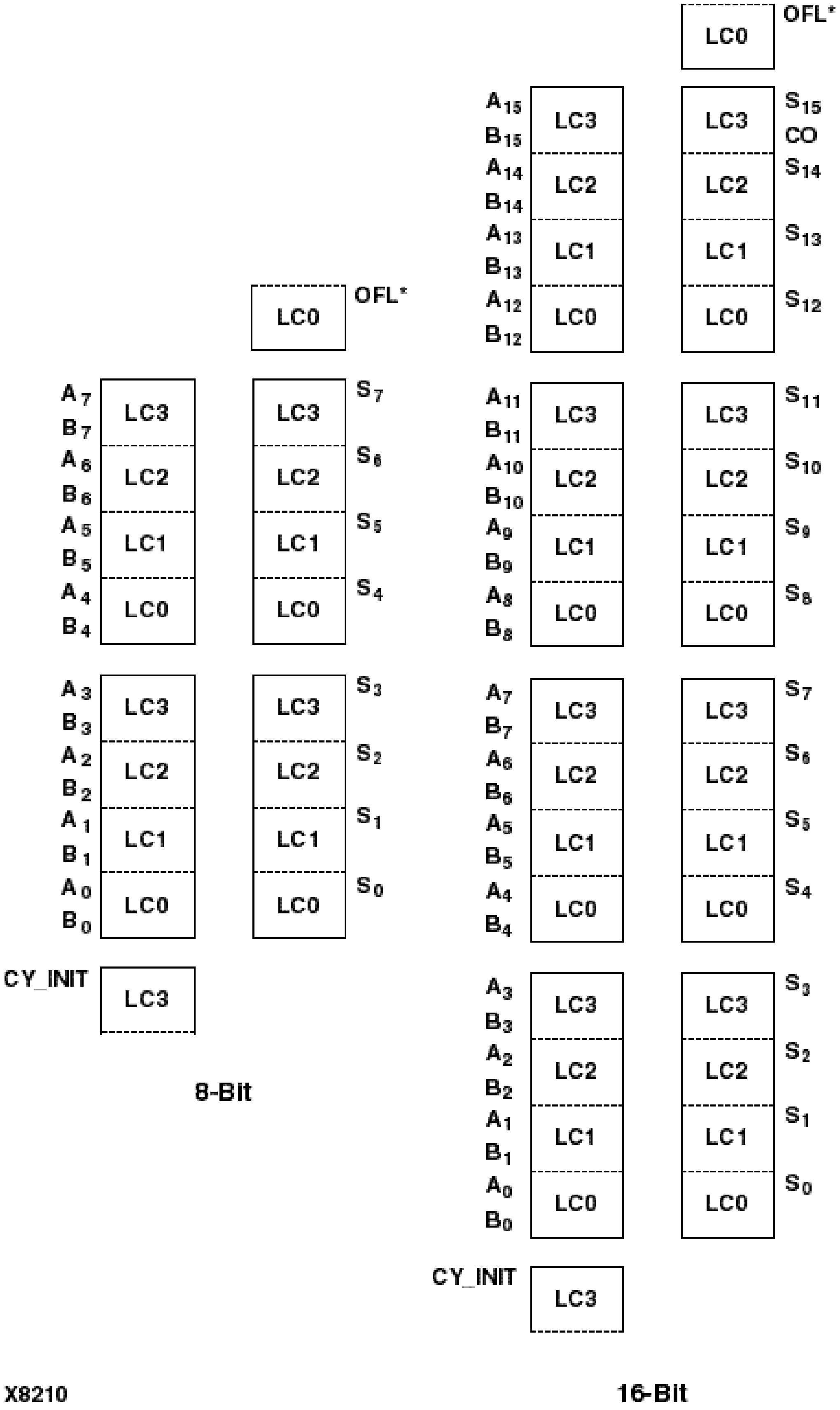

LC0
OFL*
A_{7}
B_{7}
A_{6}
B_{6}
A_{5}
B_{5}
A_{4}
B_{4}
LC3
LC2
LC1
LC0
S_{7}
S_{6}
S_{5}
S_{4}
A_{3}
B_{3}
A_{2}
B_{2}
A_{1}
B_{1}
A_{0}
B_{0}
S_{3}
S_{2}
S_{1}
S_{0}
CY_INIT
LC3
8-Bit
LC0
OFL*
A_{15}
B_{15}
A_{14}
B_{14}
A_{13}
B_{13}
A_{12}
B_{12}
S_{15}
CO
S_{14}
S_{13}
S_{12}
A_{11}
B_{11}
A_{10}
B_{10}
A_{9}
B_{9}
A_{8}
B_{8}
S_{11}
S_{10}
S_{9}
S_{8}
A_{7}
B_{7}
A_{6}
B_{6}
A_{5}
B_{5}
A_{4}
B_{4}
S_{7}
S_{6}
S_{5}
S_{4}
A_{3}
B_{3}
A_{2}
B_{2}
A_{1}
B_{1}
A_{0}
B_{0}
S_{3}
S_{2}
S_{1}
S_{0}
CY_INIT
LC3
X8210
16-Bit

Figure 3-9ADD8 Implementation XC3000

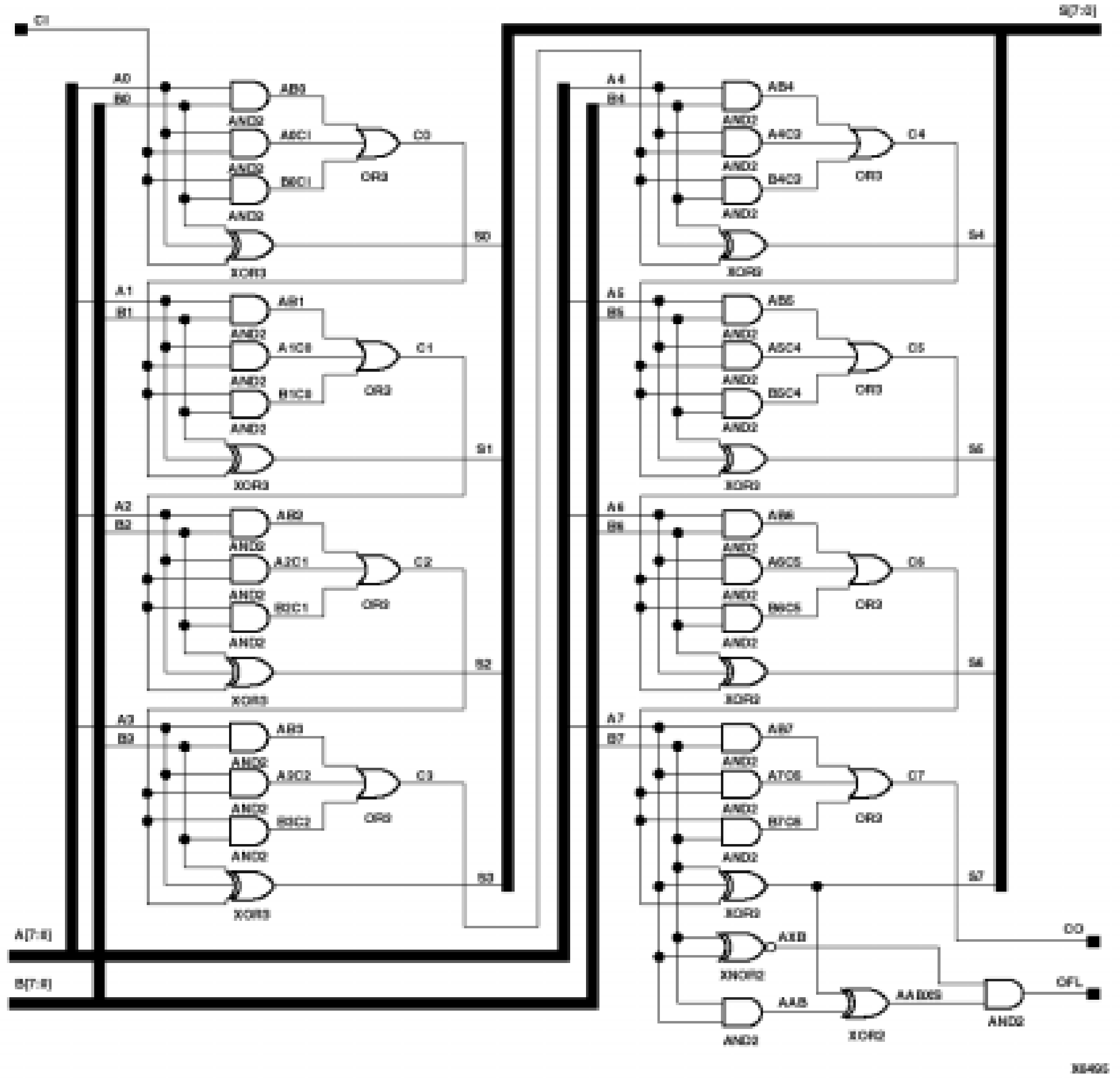


Figure 3-10ADD8 Implementation XC4000, Spartans

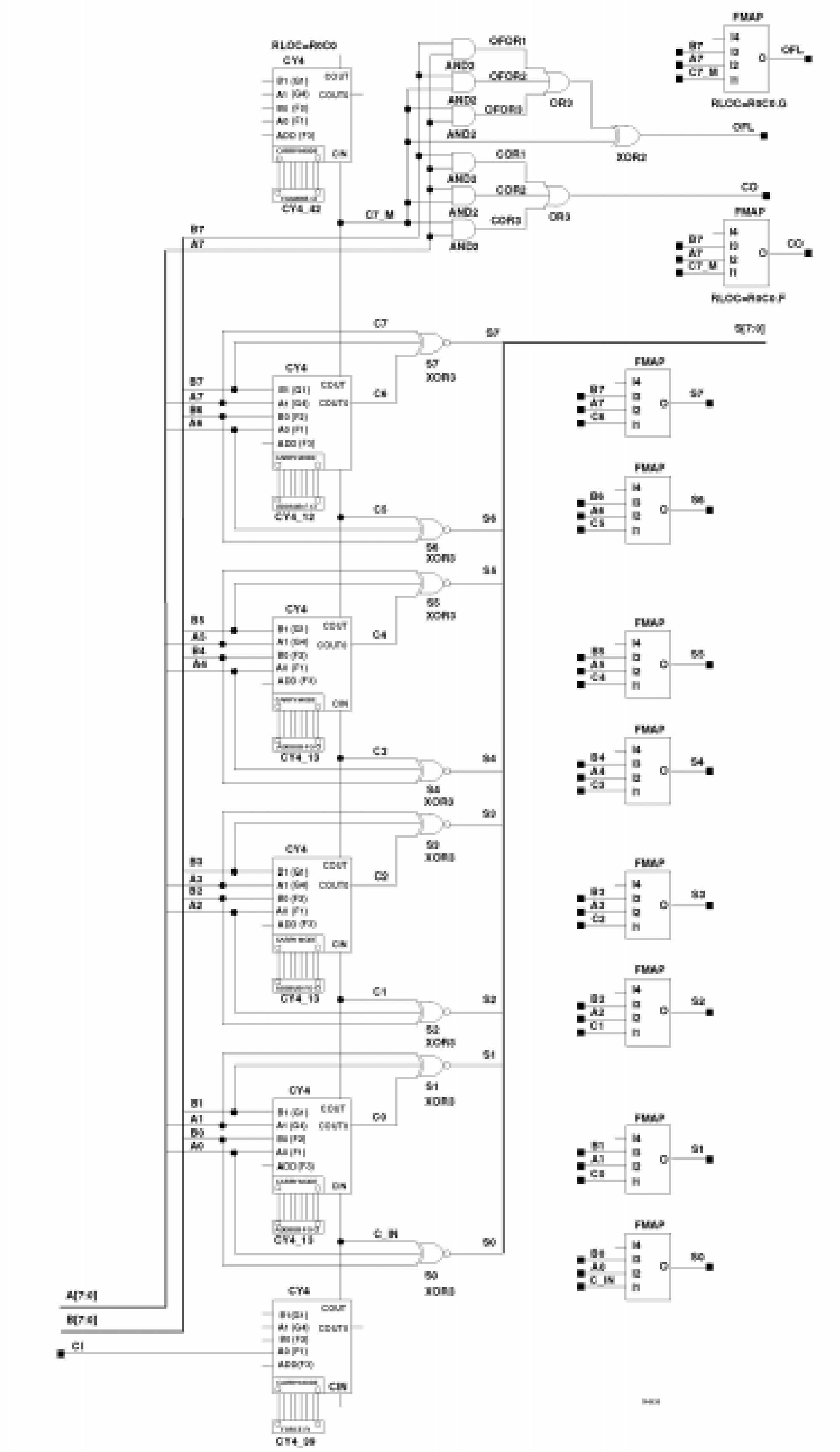

RLOC=R0C0
CY4
COUT
COUT0
CIN
CY4_42
OFOR1
AND3
OFOR2
AND2
OFOR3
AND2
OR3
XOR2
OFL
COR1
AND3
COR2
AND2
COR3
AND3
OR3
CO
C7_M
B7
A7
FMAP
I4
I3
I2
I1
O
OFL
RLOC=R0C0.G
FMAP
CO
RLOC=R0C0.F
C7
S7
S[7:0]
S7
XOR3
CY4
B1 (G1)
A1 (G4)
B0 (F3)
A0 (F1)
ADD (F3)
C6
B7
A7
B6
A6
CY4_12
C5
S6
S6
XOR3
S5
S5
XOR3
B5
A5
B4
A4
C4
CY4_13
C3
S4
S4
XOR3
S3
S3
XOR3
B3
A3
B2
A2
C2
CY4_13
C1
S2
S2
XOR3
S1
S1
XOR3
B1
A1
B0
A0
C0
CY4_13
C_IN
S0
S0
XOR3
A[7:0]
B[7:0]
CI
CY4
CY4_09
B6
A6
C5
B5
A5
C4
B4
A4
C3
B3
A3
C2
B2
A2
C1
B1
A1
C0
B0
A0
C_IN

Figure 3-11ADD8 Implementation XC5200

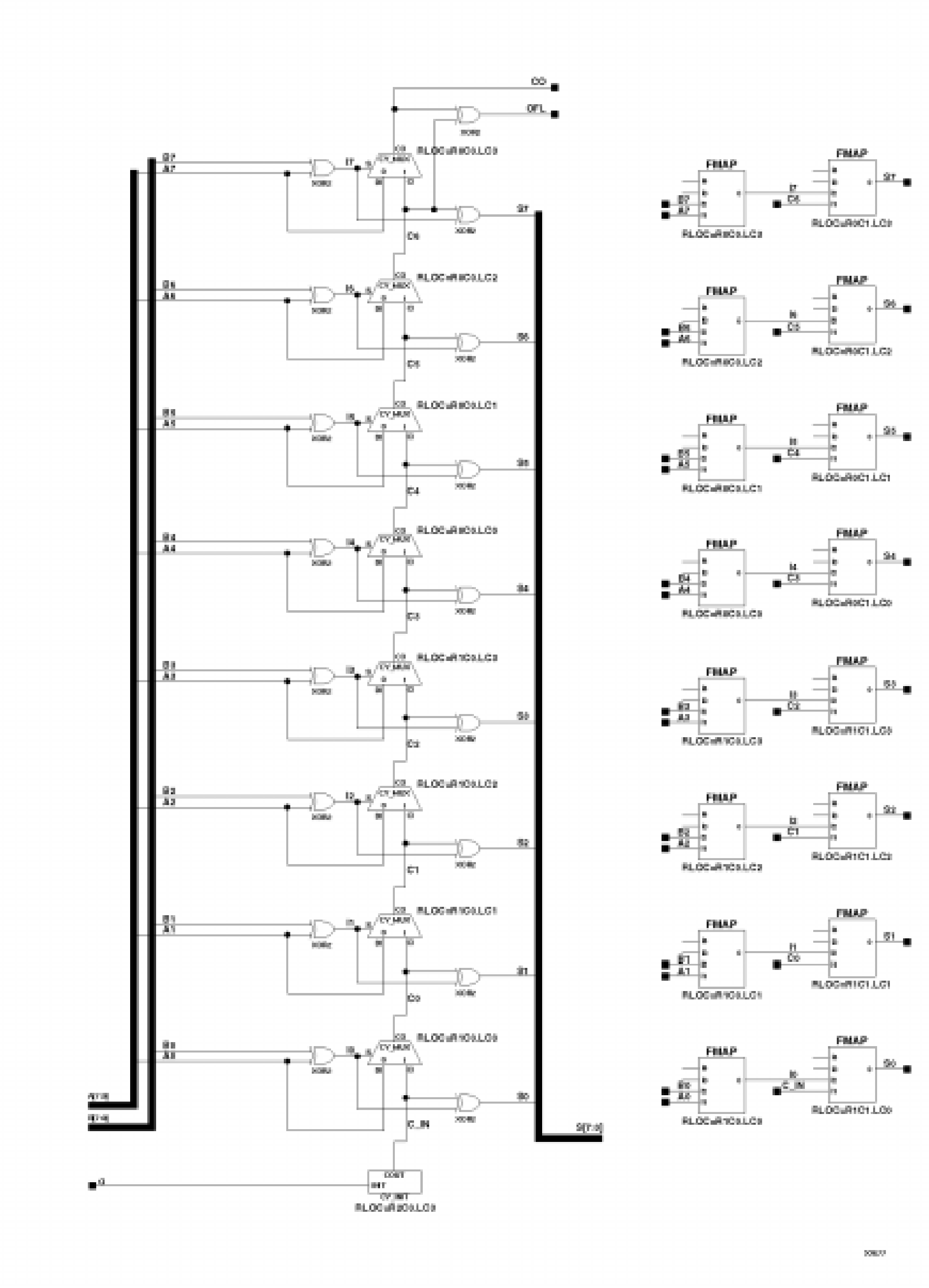


Figure 3-12ADD8 Implementation Virtex

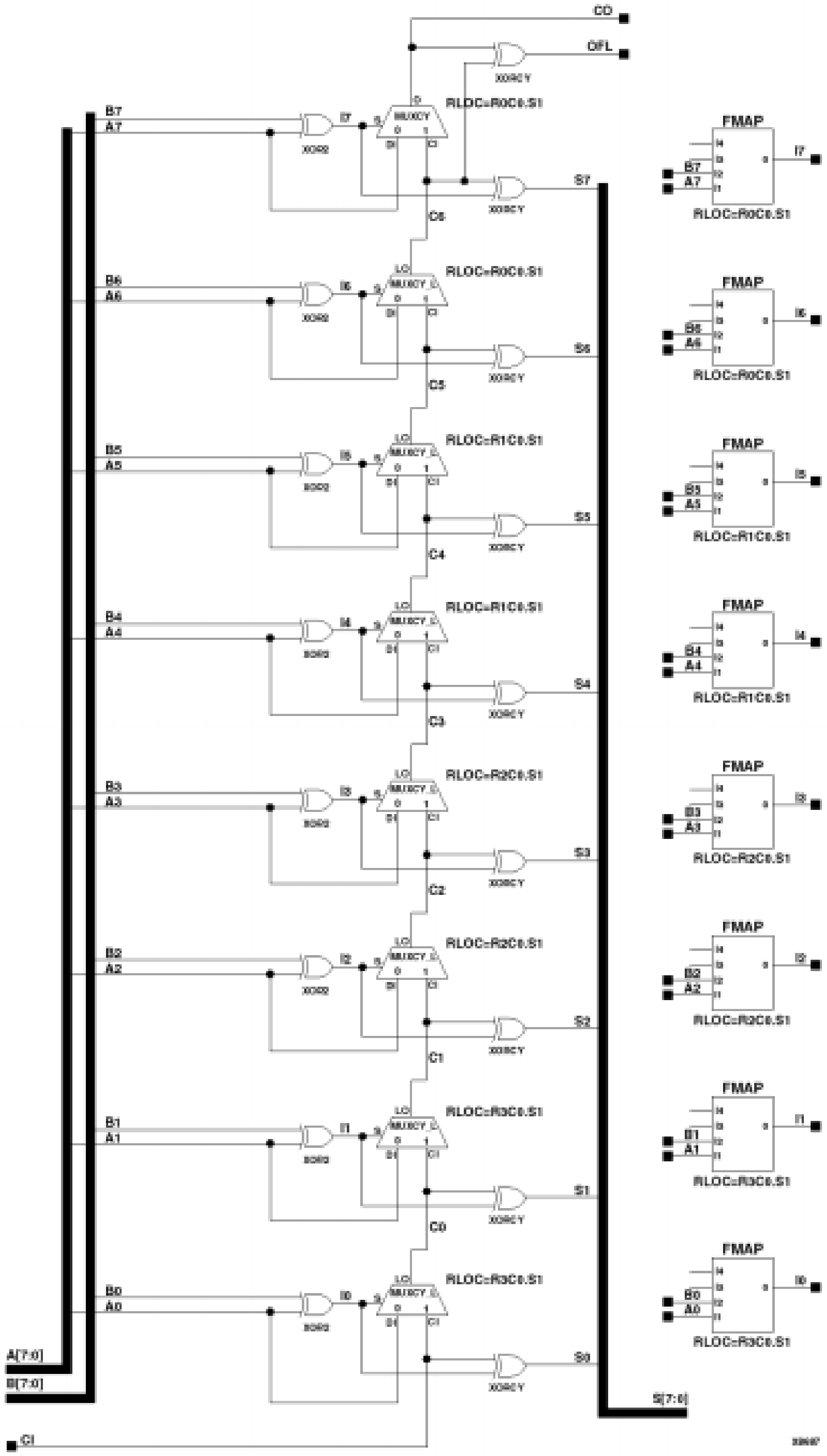

CO
OFL
XORCY
RLOC=R0C0.S1
B7
A7
I7
XOR2
MUXCY
S7
C6
FMAP
B6
A6
I6
S6
C5
RLOC=R1C0.S1
B5
A5
I5
S5
C4
B4
A4
I4
S4
C3
RLOC=R2C0.S1
B3
A3
I3
S3
C2
B2
A2
I2
S2
C1
RLOC=R3C0.S1
B1
A1
I1
S1
C0
B0
A0
I0
S0
A[7:0]
B[7:0]
S[7:0]
CI

Figure 3-13ADD4 Implementation XC9000

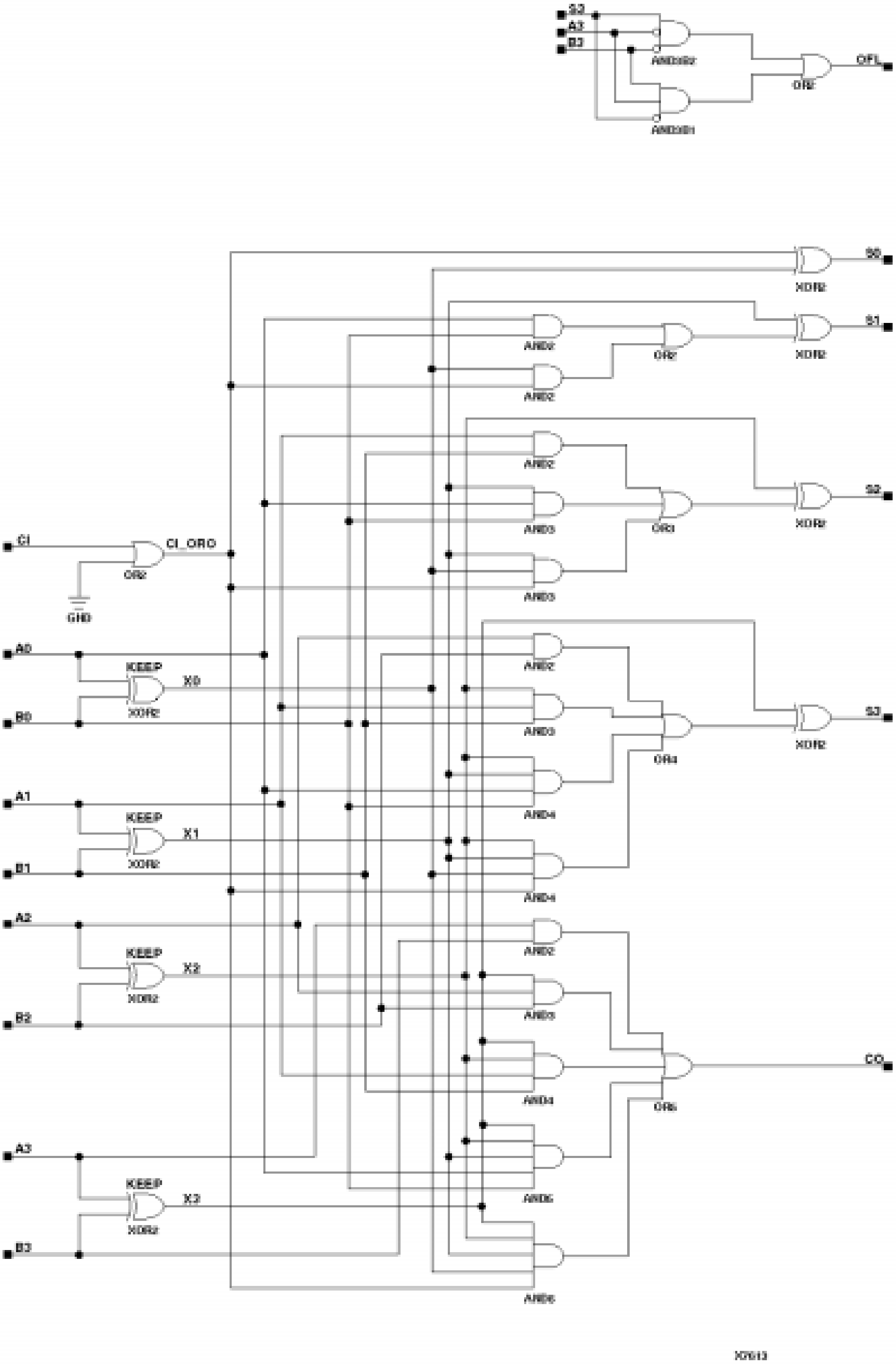
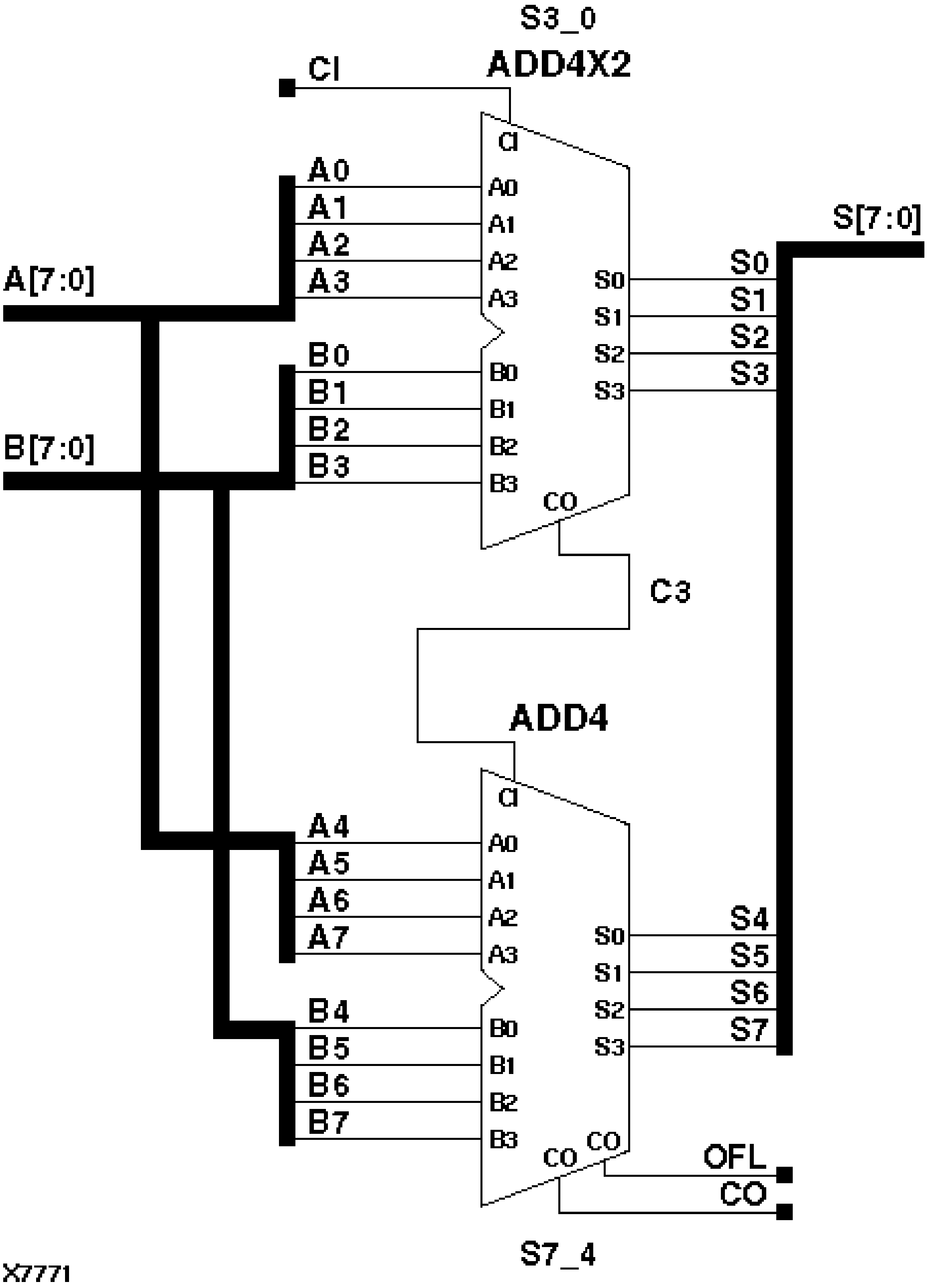

S3
A3
B3
AND3B2
AND3B1
OR2
OFL
S0
XOR2
S1
AND2
OR2
XOR2
AND2
S2
AND2
AND3
OR3
XOR2
AND3
CI
CI_ORO
OR2
GND
A0
KEEP
X0
XOR2
B0
S3
AND2
AND3
OR4
XOR2
A1
AND4
KEEP
X1
XOR2
B1
AND4
A2
AND2
KEEP
X2
XOR2
AND3
B2
CO
AND4
OR5
A3
AND5
KEEP
X3
XOR2
B3
AND5
X7613

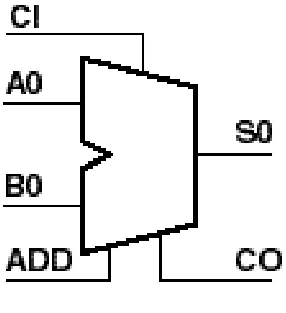
Figure 3-14ADD8 Implementation XC9000



ADSU1

1-Bit Cascadable Adder/Subtracter with Carry-In and Carry-Out

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	N/A	N/A	Macro	N/A	N/A	N/A



X4035

When the ADD input is High, two 1-bit words (A0 and B0) are added with a carry-in (CI), producing a 1-bit output (S0) and a carry-out (CO). When the ADD input is Low, B0 is subtracted from A0, producing a result (S0) and borrow (CO). In add mode, CO represents a carry-out, and CO and CI are active-High. In subtract mode, CO represents a borrow, and CO and CI are active-Low.

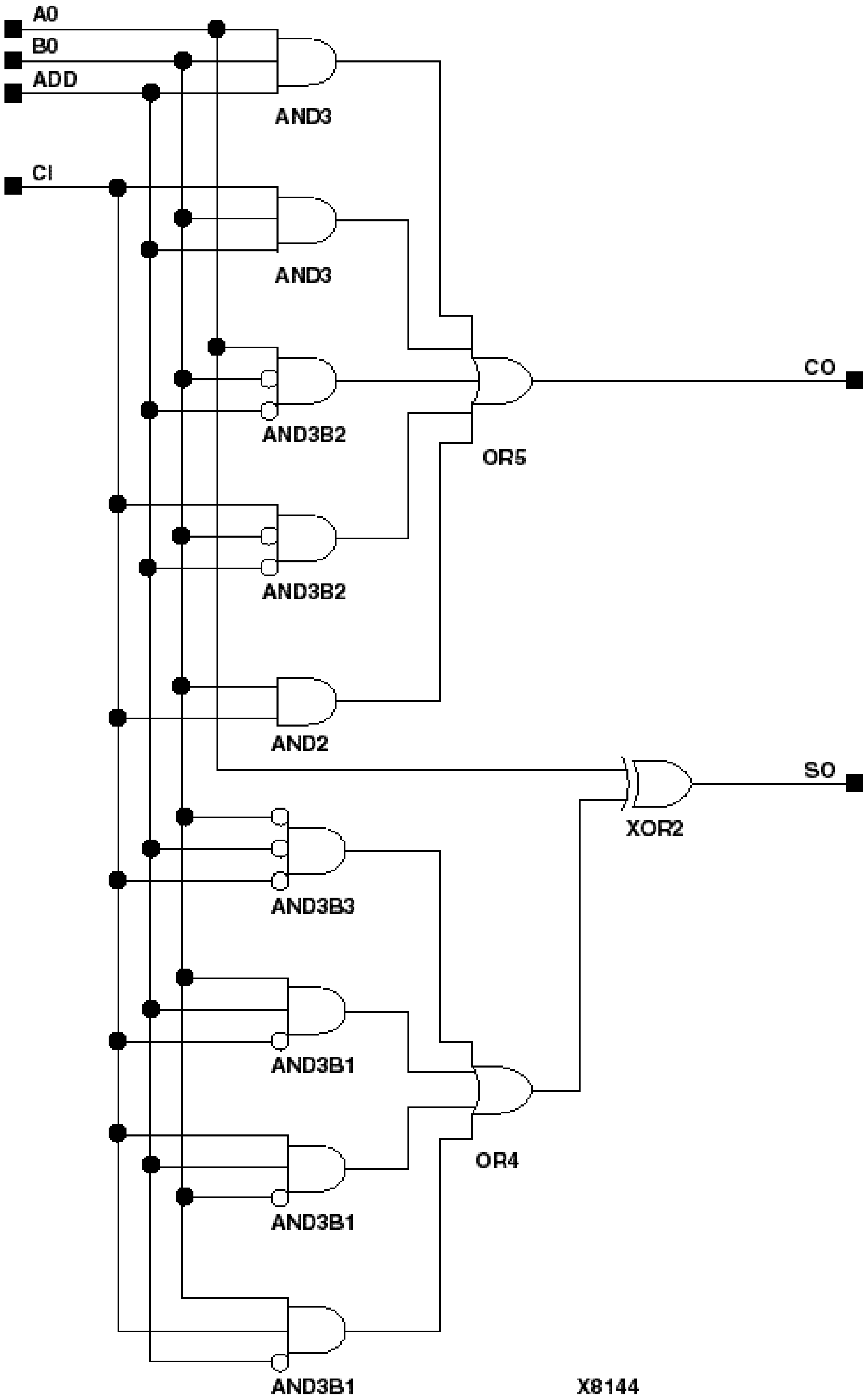
Inputs			Outputs	
A0	**B0**	**CI**	**S0**	**CO**
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Add Function, ADD=1

Inputs			**Outputs**	
A0	**B0**	**CI**	**S0**	**CO**
0	0	0	1	0
0	1	0	0	0
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	1
1	1	1	0	1

Subtract Function, ADD=0

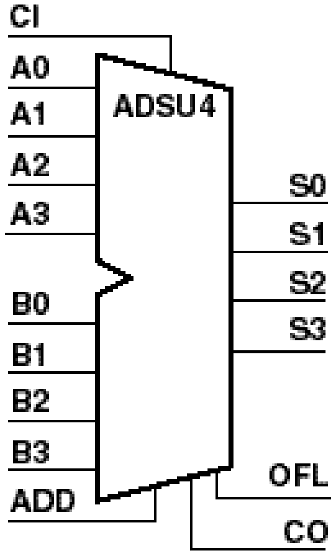
Figure 3-15ADSU1 Implementation XC9000


A0
B0
ADD
CI
AND3
AND3
AND3B2
AND3B2
AND2
AND3B3
AND3B1
AND3B1
AND3B1
OR5
CO
XOR2
SO
OR4
X8144

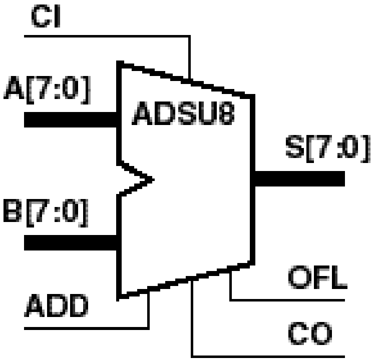
ADSU4, 8, 16

4-, 8-, 16-Bit Cascadable Adders/Subtracters with Carry-In, Carry-Out, and Overflow

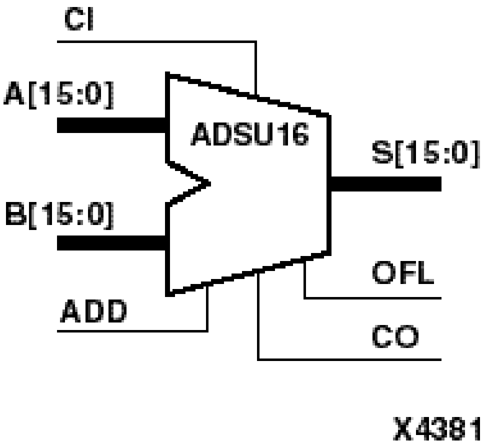
XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
Macro	Macro	Macro	Macro	Macro	Macro	Macro	Macro



X4379



X4380



When the ADD input is High, ADSU4, ADSU8, and ADSU16 add two words and a carry-in (CI), producing a sum output and carry-out (CO) or overflow (OFL). ADSU4 adds two 4-bit words (A3 – A0 and B3 – B0) and a CI, producing a 4-bit sum output (S3 – S0) and CO or OFL. ADSU8 adds two 8-bit words (A7 – A0 and B7 – B0) and a CI producing, an 8-bit sum output (S7 – S0) and CO or OFL. ADSU16 adds two 16-bit words (A15 – A0 and B15 – B0) and a CI, producing a 16-bit sum output (S15 – S0) and CO or OFL.

When the ADD input is Low, ADSU4, ADSU8, and ADSU16 subtract Bz – B0 from Az– A0, producing a difference output and CO or OFL. ADSU4 subtracts B3 – B0 from A3 – A0, producing a 4-bit difference (S3 – S0) and CO or OFL. ADSU8 subtracts B7 – B0 from A7 – A0, producing an 8-bit difference (S7 – S0) and CO or OFL. ADSU16 subtracts B15 – B0 from A15 – A0, producing a 16-bit difference (S15 – S0) and CO or OFL.

In add mode, CO and CI are active-High. In subtract mode, CO and CI are active-Low. OFL is active-High in add and subtract modes.

ADSU4, ADSU8, and ADU16 are implemented in the XC4000 and Spartans using carry logic and relative location constraints, which assure most efficient logic placement.

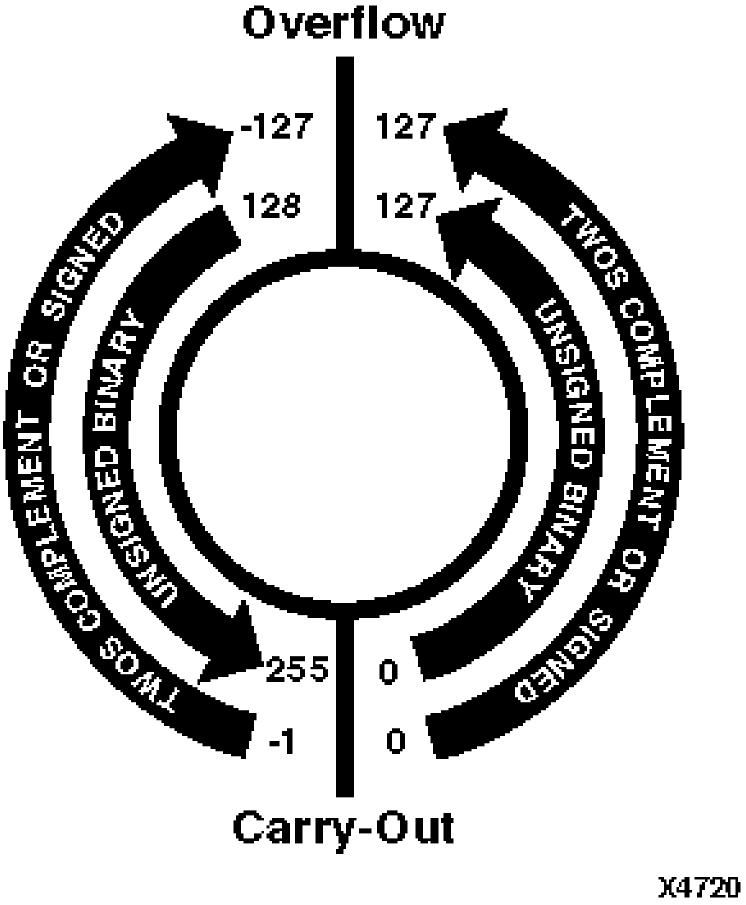
ADSU4, ADSU8, and ADSU16 CI and CO pins do not use the CPLD carry chain.

Unsigned Binary Versus Twos Complement

ADSU4, ADSU8, ADSU16 can operate, respectively, on either 4-, 8-, 16-bit unsigned binary numbers or 4-, 8-, 16-bit twos-complement numbers. If the inputs are interpreted as unsigned binary, the result can be interpreted as unsigned binary. If the inputs are interpreted as twos complement, the output can be interpreted as twos complement. The only functional difference between an unsigned binary operation and a twos-complement operation is how they determine when "overflow" occurs. Unsigned binary uses CO, while twos complement uses OFL to determine when "overflow" occurs.

With adder/subtracters, either unsigned binary or twos-complement operations cause an overflow. If the result crosses the overflow boundary, an overflow is generated. Similarly, when the result crosses the carry-out boundary, a carry-out is generated. The following figure shows the ADSU carry-out and overflow boundaries.

Figure 3-16ADSU Carry-Out and Overflow Boundaries



Unsigned Binary Operation

For unsigned binary operation, ADSU4 can represent numbers between 0 and 15, inclusive; ADSU8 between 0 and 255, inclusive; ADSU16 between 0 and 65535, inclusive. In add mode, CO is active (High) when the sum exceeds the bounds of the adder/subtracter. In subtract mode, CO is an active-Low borrow-out and goes Low when the difference exceeds the bounds.

An unsigned binary "overflow" that is always active-High can be generated by gating the ADD signal and CO as follows.

```
unsigned overflow = CO XNOR ADD
```

OFL is ignored in unsigned binary operation.

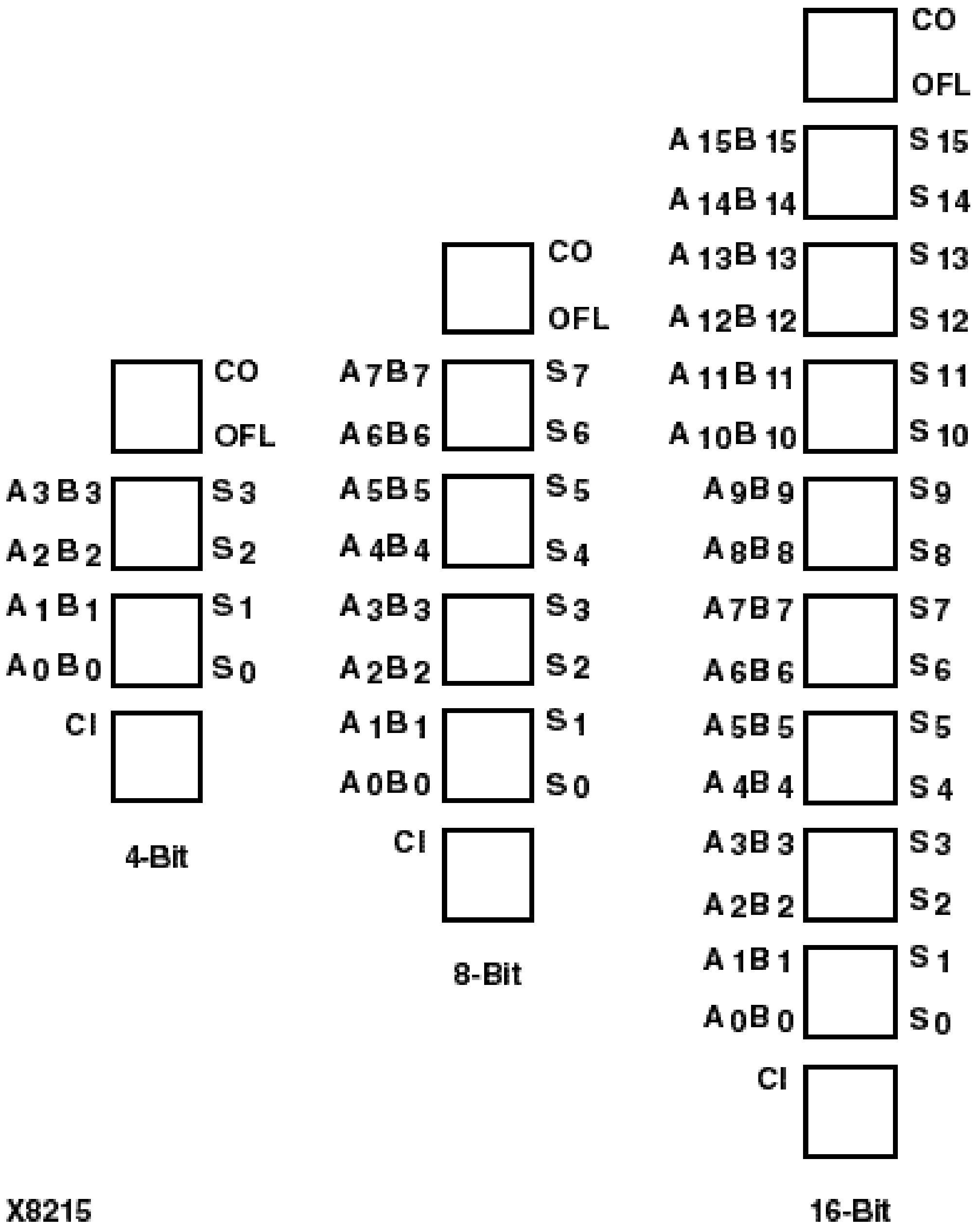
Twos-Complement Operation

For twos-complement operation, ADSU4 can represent numbers between -8 and +7, inclusive; ADSU8 between -128 and +127, inclusive; ADSU16 between -32768 and +32767, inclusive. If an addition or subtraction operation result exceeds this range, the OFL output goes High.

CO is ignored in twos-complement operation.

Topology for XC4000 and Spartans

This is the ADSU4 (4-bit), ADSU8 (8-bit), and ADSU16 (16-bit) topology for XC4000 and Spartan series devices.



XC5200 Topology

This is the ADSU8 (8-bit) and ADSU16 (16-bit) topology for XC5200 devices.

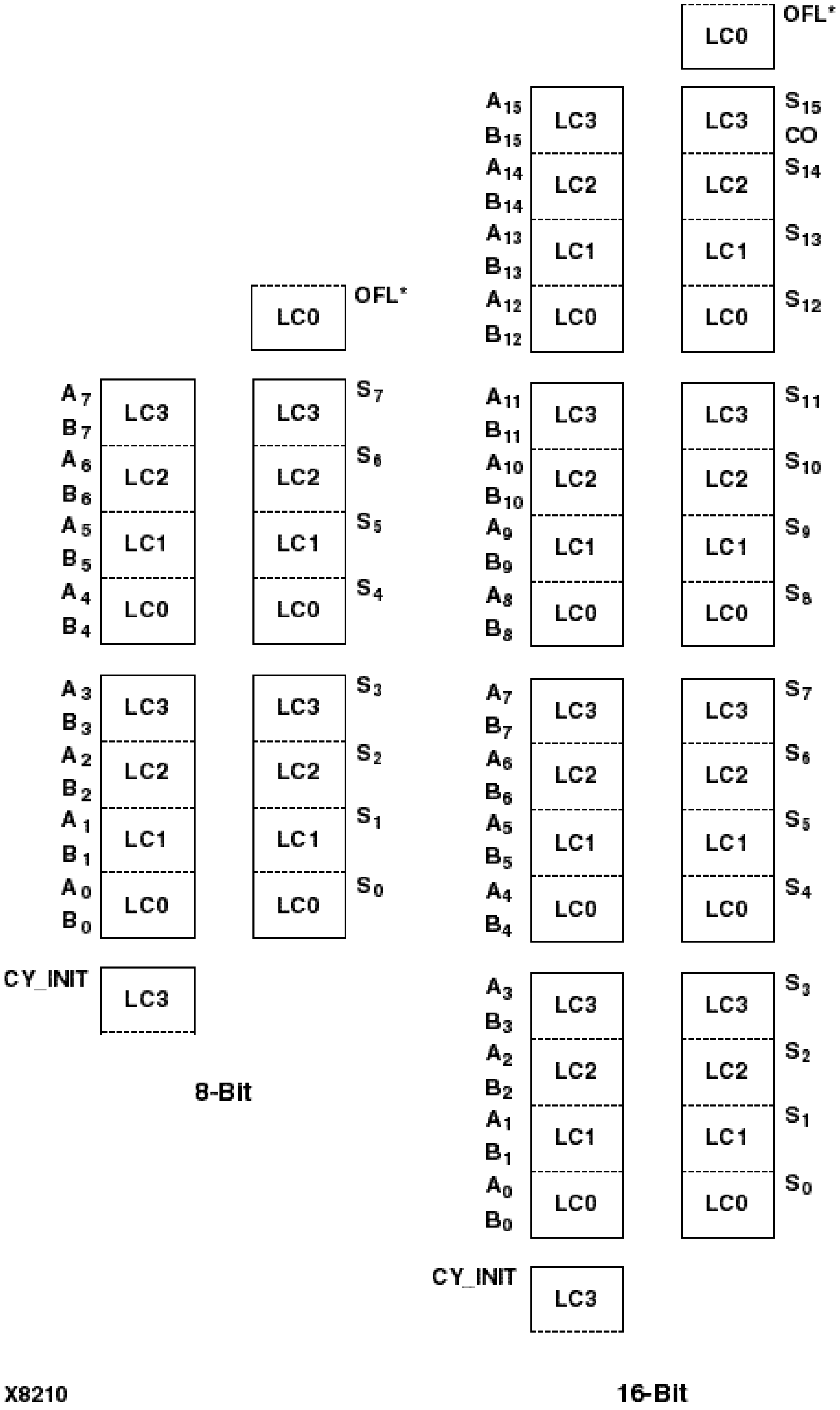

OFL*
LC0
A15
B15
A14
B14
A13
B13
A12
B12
LC3
LC2
LC1
LC0
S15
CO
S14
S13
S12
OFL*
A7
B7
A6
B6
A5
B5
A4
B4
S7
S6
S5
S4
A11
B11
A10
B10
A9
B9
A8
B8
S11
S10
S9
S8
A3
B3
A2
B2
A1
B1
A0
B0
S3
S2
S1
S0
CY_INIT
8-Bit
16-Bit
X8210

Figure 3-17ADSU8 Implementation XC3000

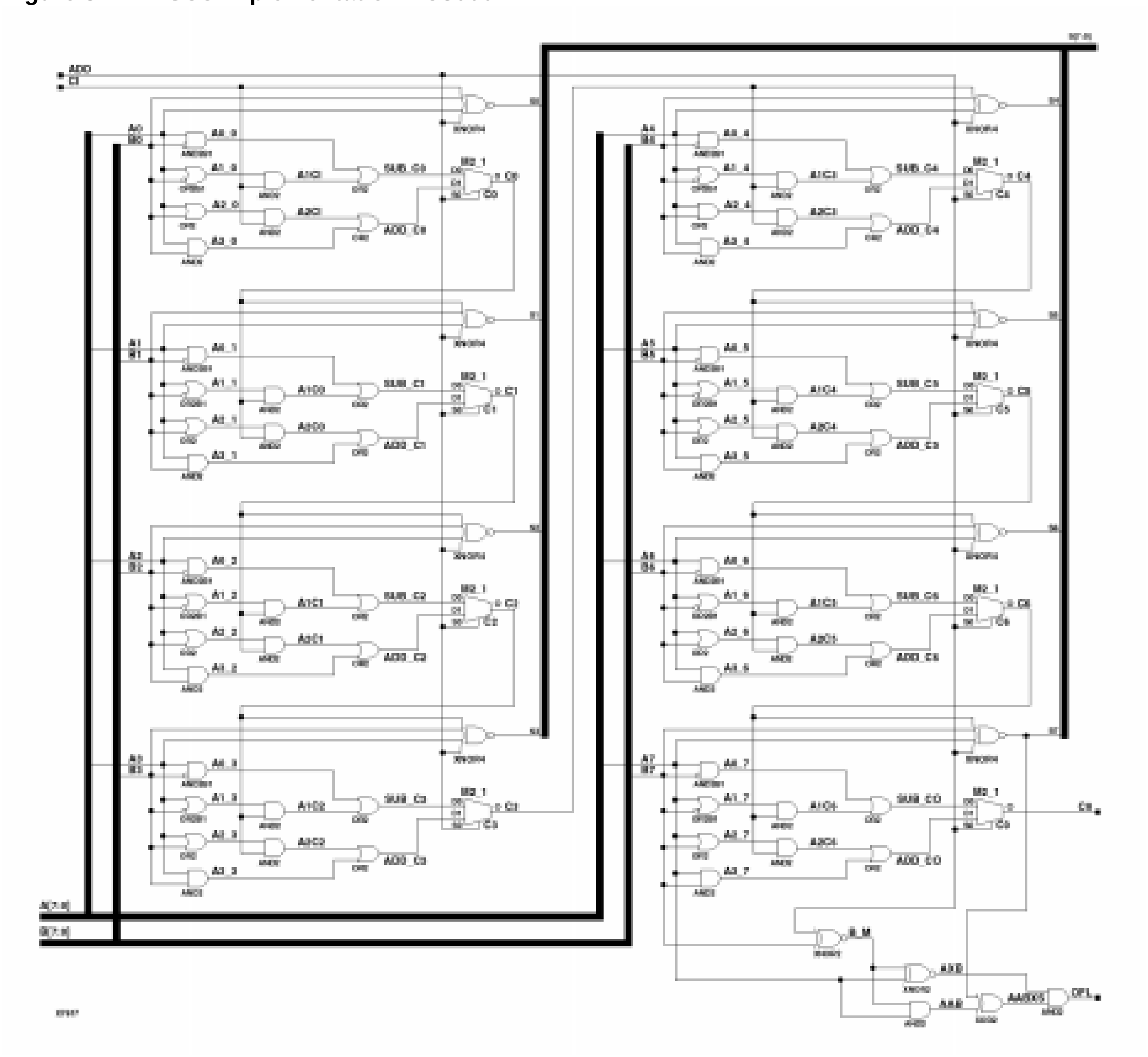


Figure 3-18ADSU8 Implementation XC4000, Spartans

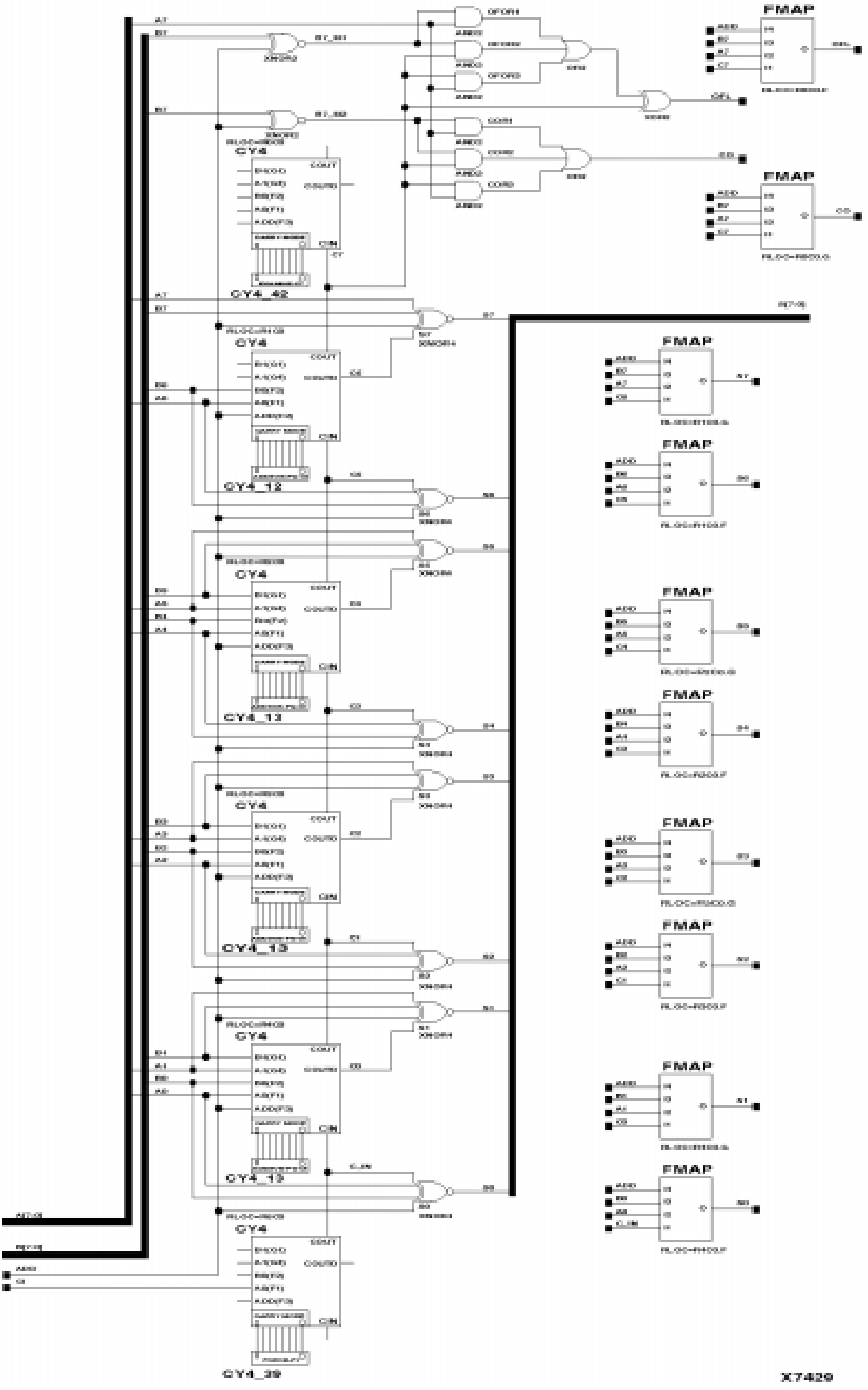
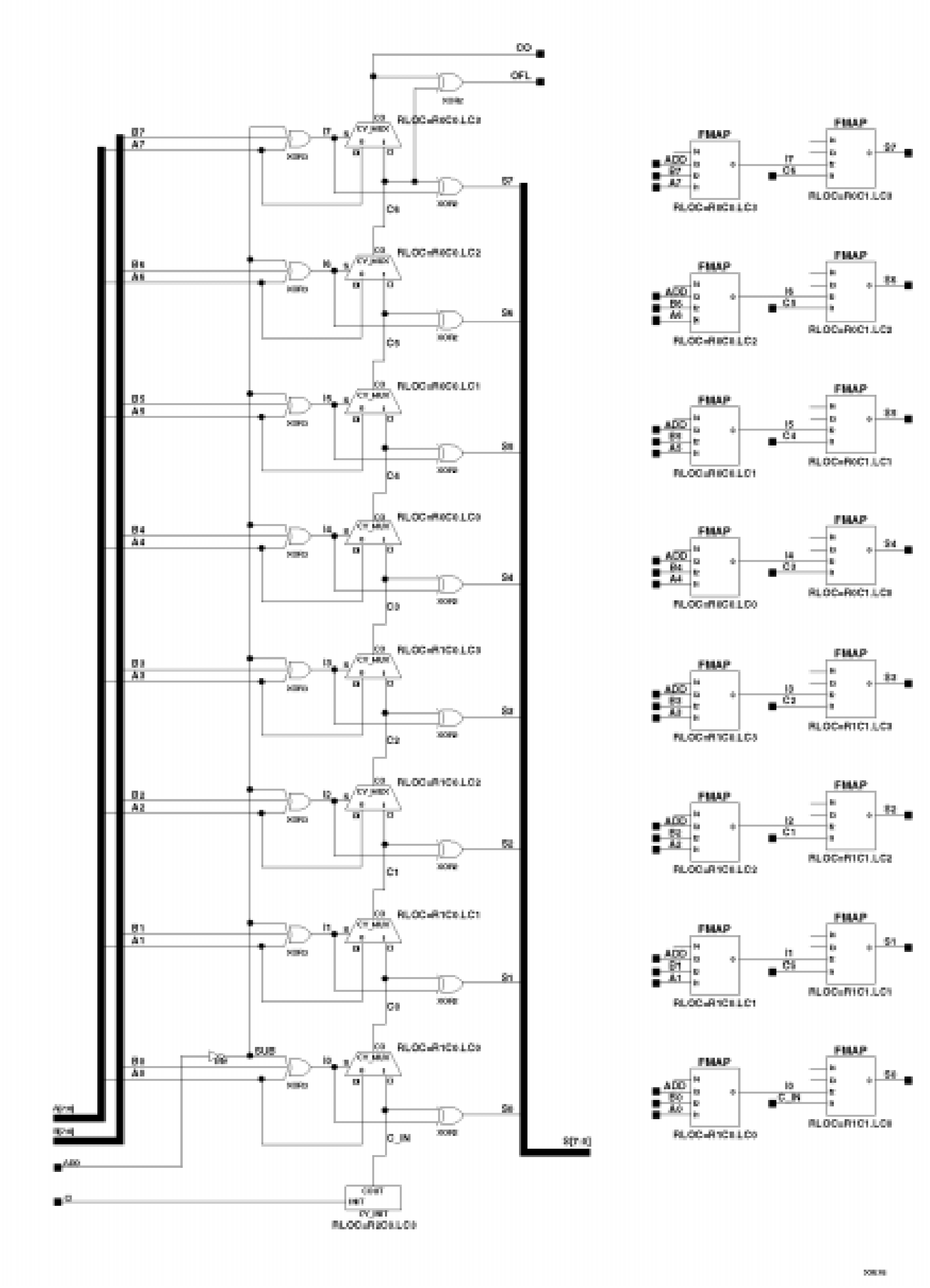

FMAP
FMAP
FMAP
FMAP
FMAP
FMAP
FMAP
FMAP
FMAP
FMAP
CY4
CY4_42
CY4
CY4_12
CY4
CY4_13
CY4
CY4_13
CY4
CY4_13
CY4
CY4_39
COUT
CIN
X7429

Figure 3-19ADSU8 Implementation XC5200



]

Figure 3-20ADSU8 Implementation Virtex

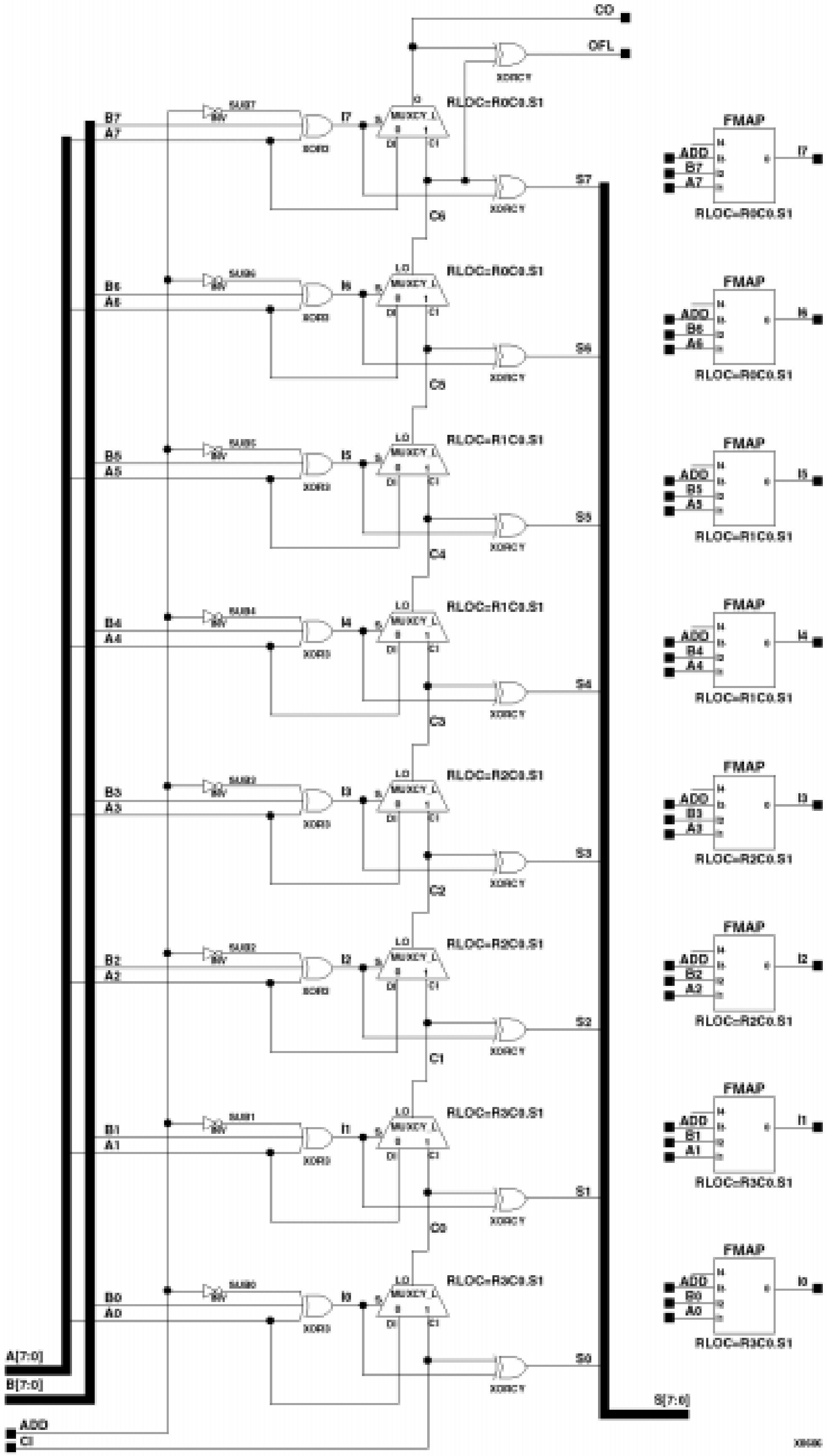

CO
OFL
XORCY
RLOC=R0C0.S1
MUXCY_L
SUB7
INV
B7
A7
XOR3
I7
S7
C6
B6
A6
I6
S6
C5
RLOC=R1C0.S1
B5
A5
I5
S5
C4
B4
A4
I4
S4
C3
RLOC=R2C0.S1
B3
A3
I3
S3
C2
B2
A2
I2
S2
C1
RLOC=R3C0.S1
B1
A1
I1
S1
C0
B0
A0
I0
S0
FMAP
ADD
A[7:0]
B[7:0]
CI
S[7:0]

Figure 3-21ADSU4 Implementation XC9000

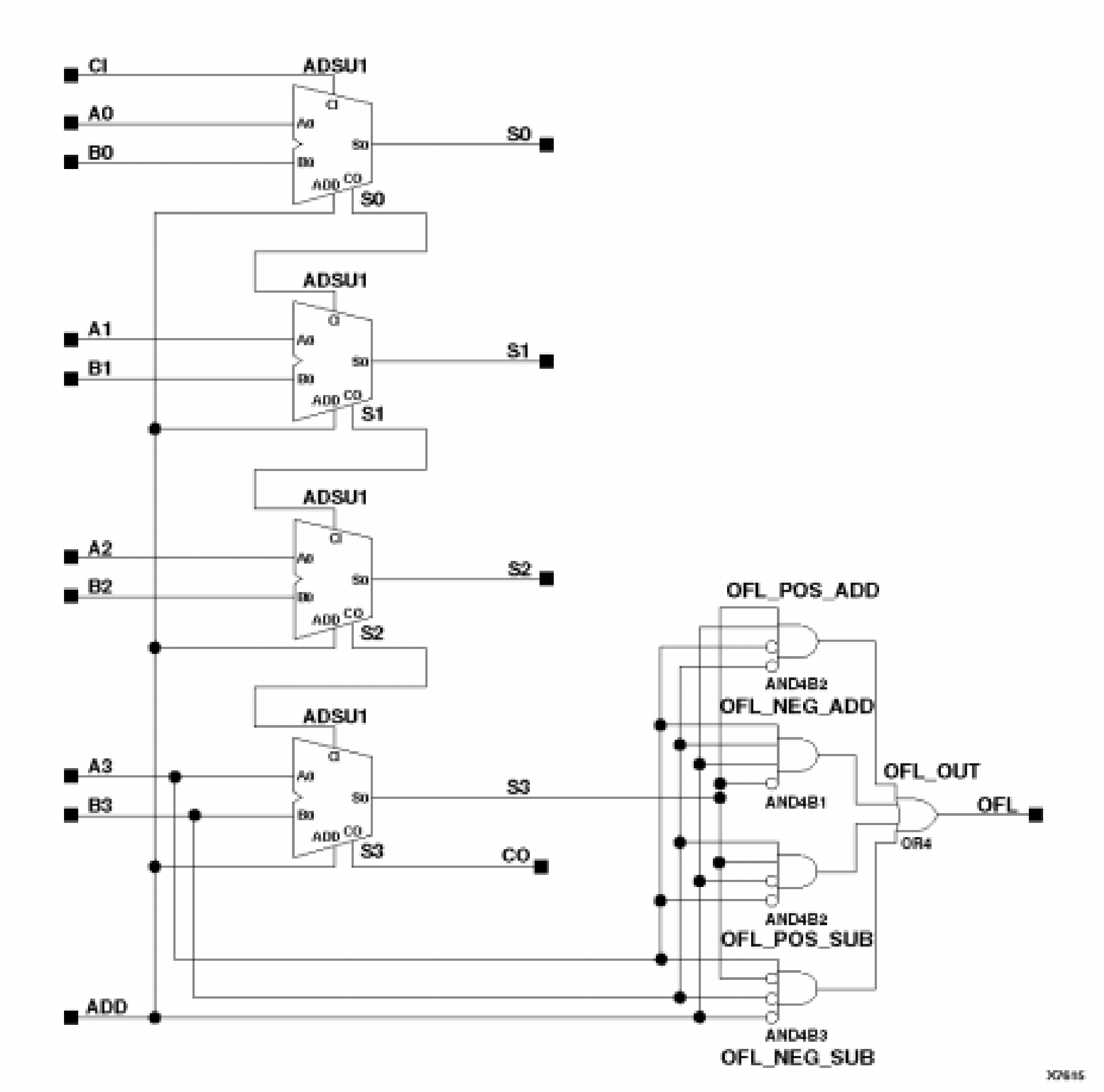
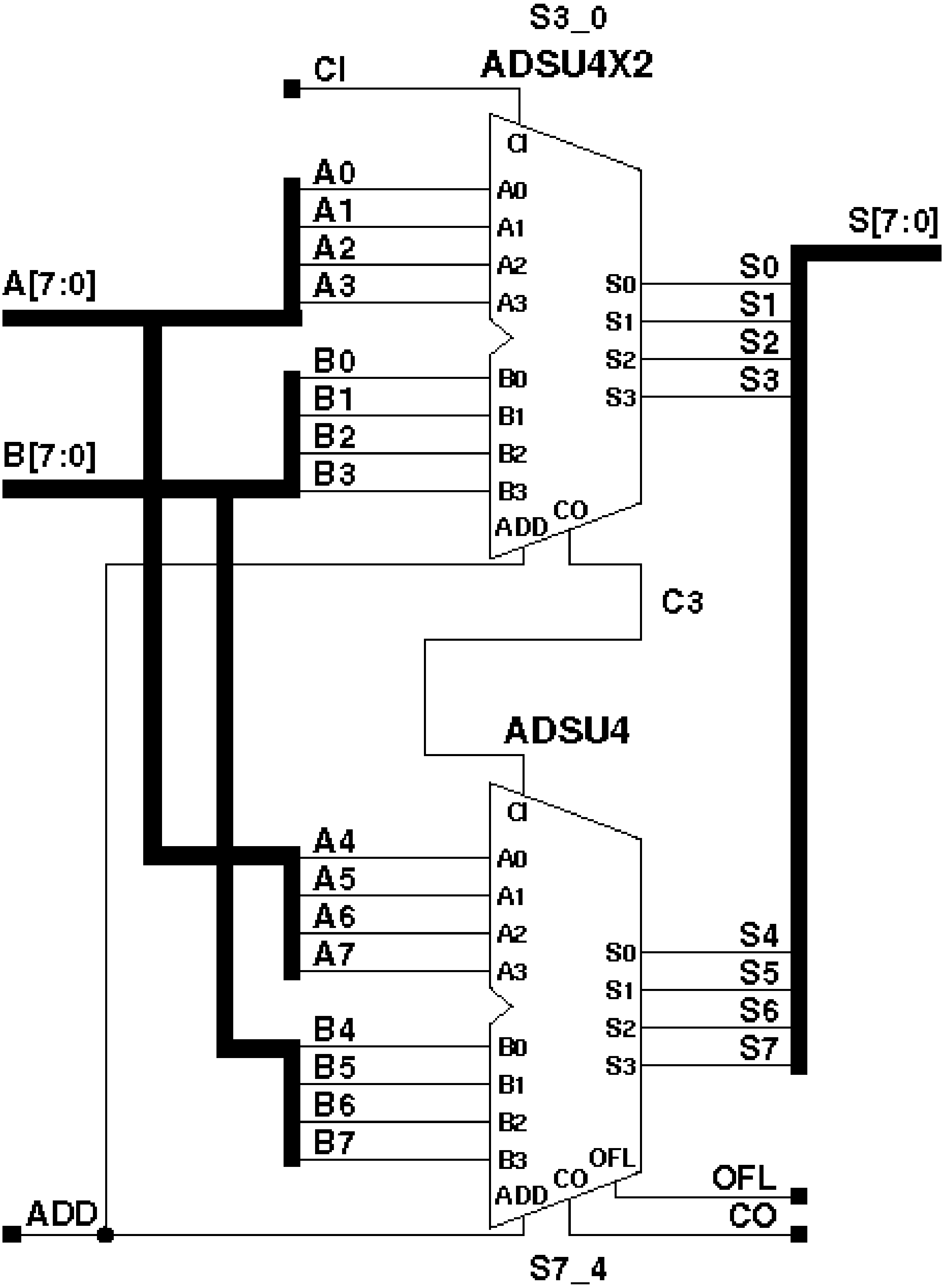


Figure 3-22ADSU8 Implementation XC9000


S3_0
ADSU4X2
CI
CI
A0
A1
A2
A3
A0
A1
A2
A3
A[7:0]
S[7:0]
S0
S1
S2
S3
S0
S1
S2
S3
B0
B1
B2
B3
B0
B1
B2
B3
B[7:0]
ADD
CO
C3
ADSU4
CI
A4
A5
A6
A7
A0
A1
A2
A3
S0
S1
S2
S3
S4
S5
S6
S7
B4
B5
B6
B7
B0
B1
B2
B3
ADD
CO
OFL
OFL
CO
ADD
S7_4

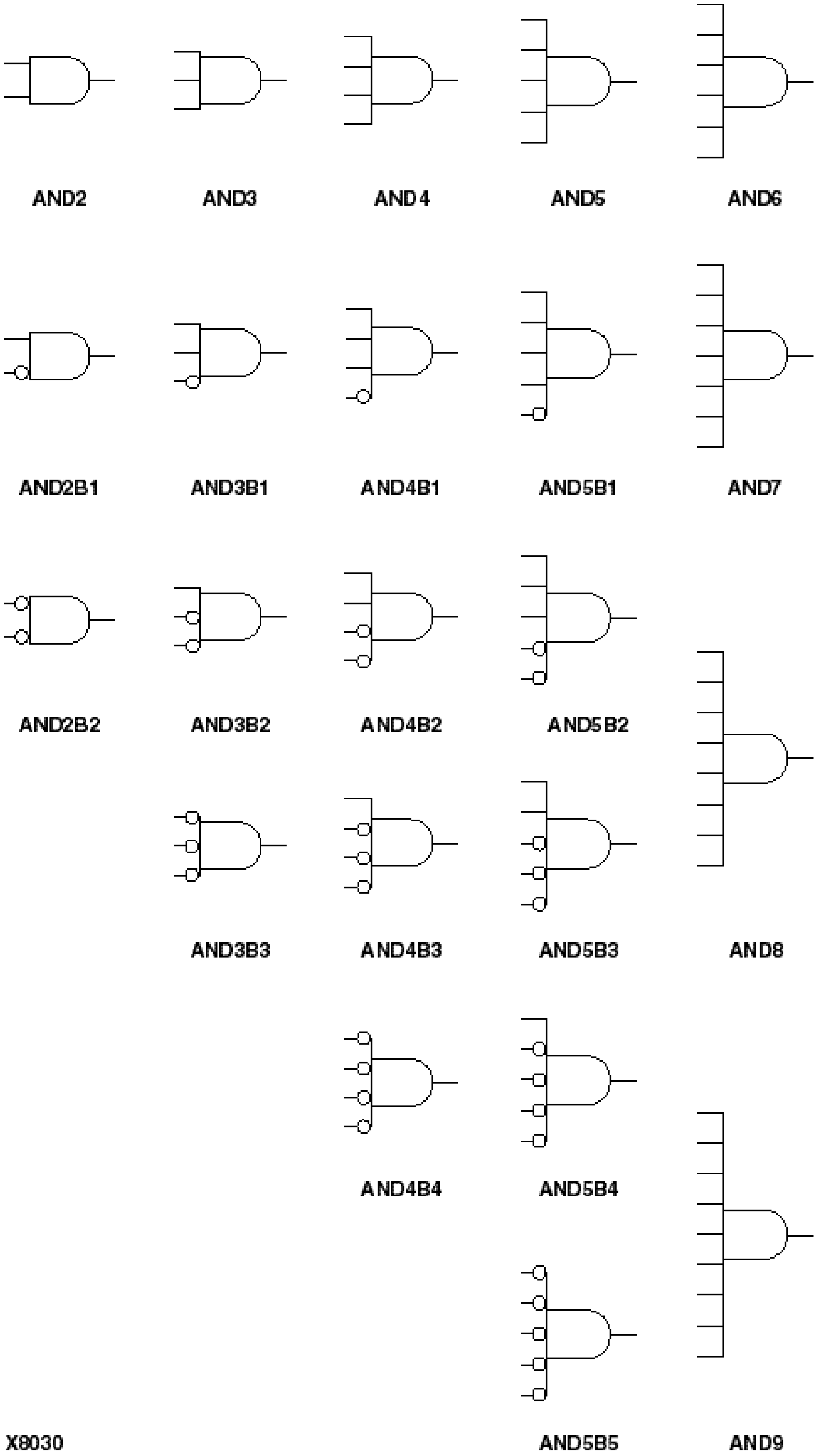
X7774

AND2-9

2- to 9-Input AND Gates with Inverted and Non-Inverted Inputs

Element	XC3000	XC4000E	XC4000X	XC5200	XC9000	Spartan	SpartanXL	Virtex
AND2, AND2B1, AND2B2, AND3, AND3B1, AND3B2, AND3B3, AND4 AND4B1, AND4B2, AND4B3, AND4B4	Primitive	Primitive	Primitive	Primitive	Primitive	Primitive	Primitive	Primitive
AND5, AND5B1, AND5B2, AND5B3, AND5B4, AND5B5	Primitive	Primitive	Primitive	Macro	Primitive	Primitive	Primitive	Primitive
AND6, AND7, AND8, AND9	Macro	Macro	Macro	Macro	Primitive	Macro	Macro	Macro

Figure 3-23AND Gate Representations


AND2
AND3
AND4
AND5
AND6
AND2B1
AND3B1
AND4B1
AND5B1
AND7
AND2B2
AND3B2
AND4B2
AND5B2
AND3B3
AND4B3
AND5B3
AND8
AND4B4
AND5B4
X8030
AND5B5
AND9

The AND function is performed in the Configurable Logic Block (CLB) function generators for XC3000, XC4000, XC5200, and Spartans.

AND functions of up to five inputs are available in any combination of inverting and non-inverting inputs. AND functions of six to nine inputs are available with only non-inverting inputs. To make some or all inputs inverting, use external inverters. Because each input uses a CLB resource in FPGAs, replace functions with unused inputs with functions having the appropriate number of inputs.

Refer to **"AND12, 16"** for information on additional AND functions for the XC5200 and Virtex.

Figure 3-24AND5 Implementation XC5200

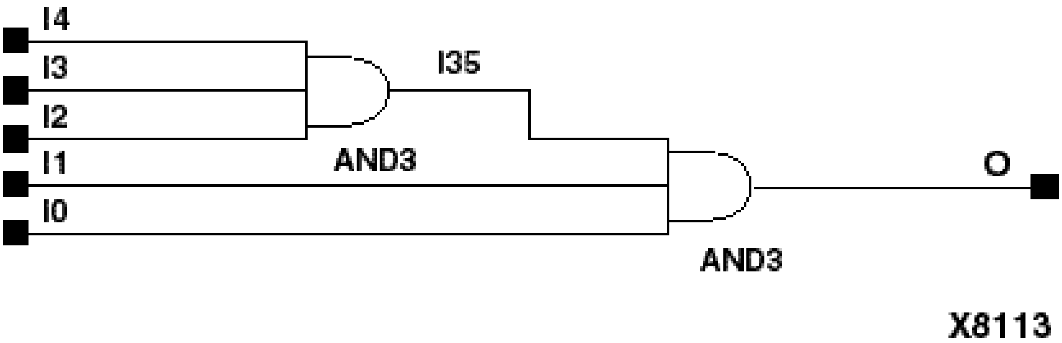


Figure 3-25AND8 Implementation XC3000

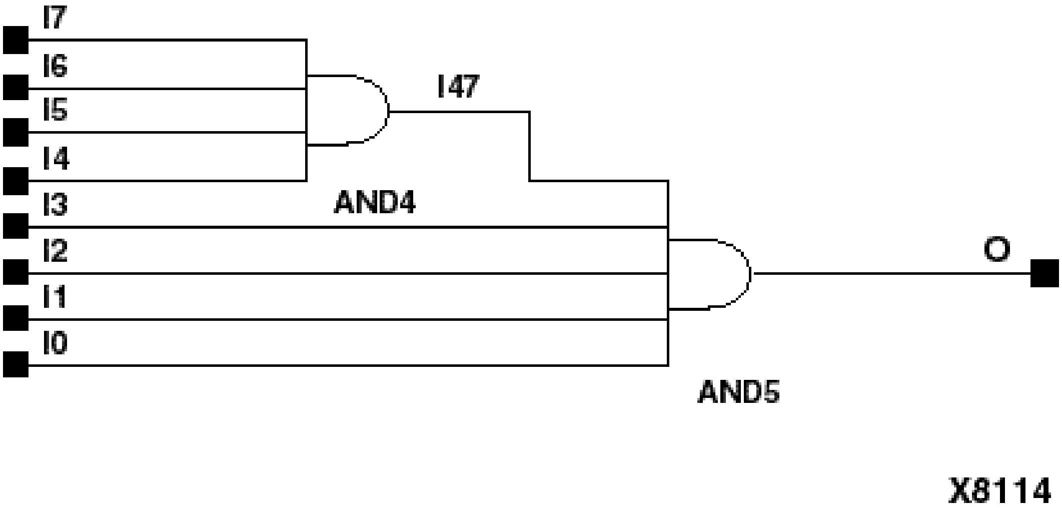


Figure 3-26AND8 Implementation XC4000, Spartans

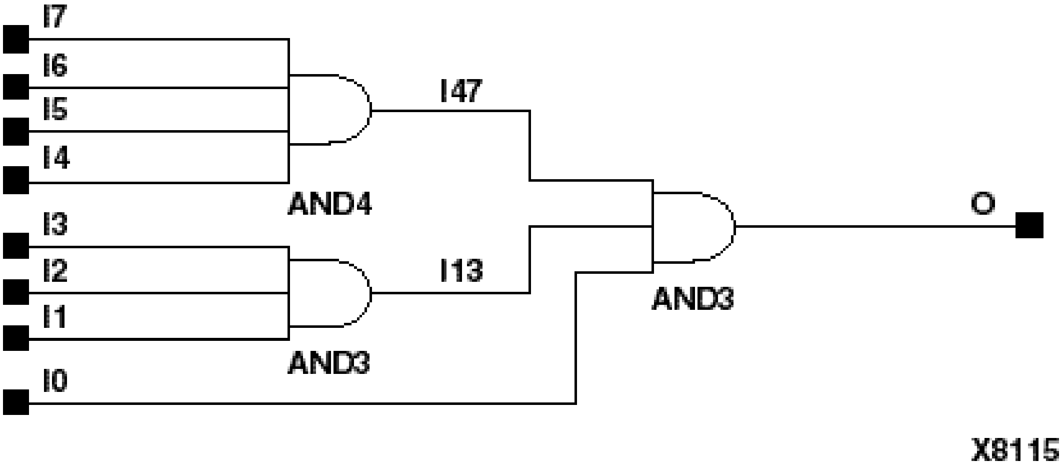


Figure 3-27AND8 Implementation XC5200

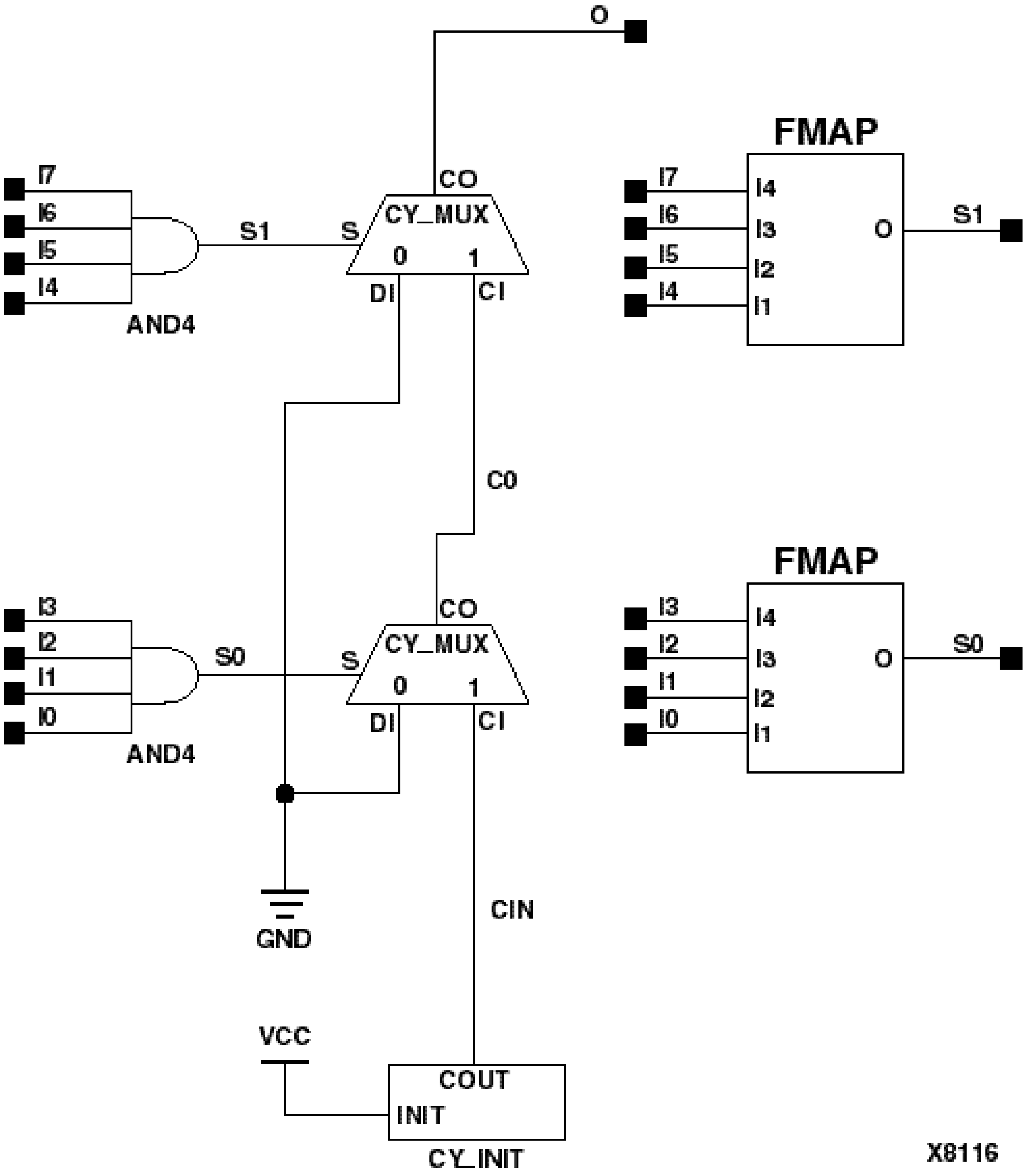
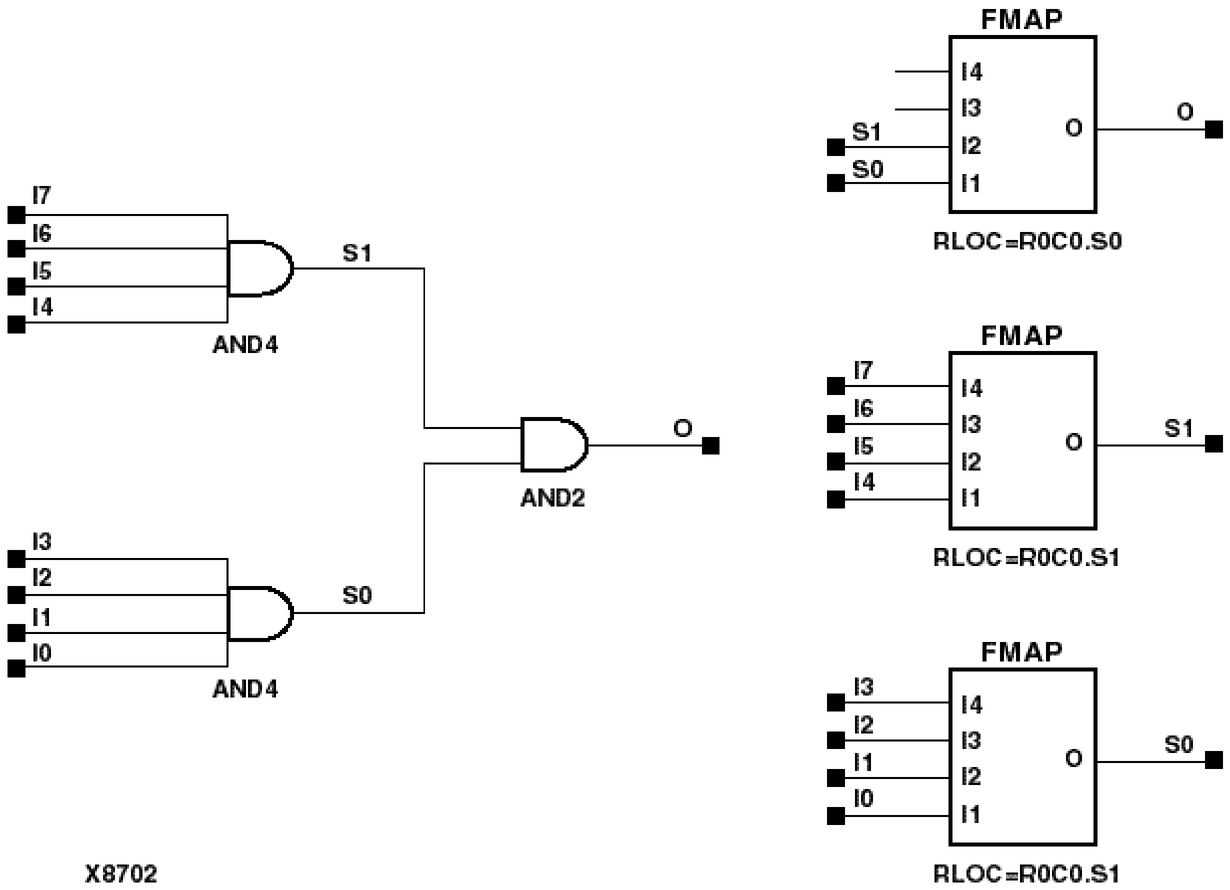


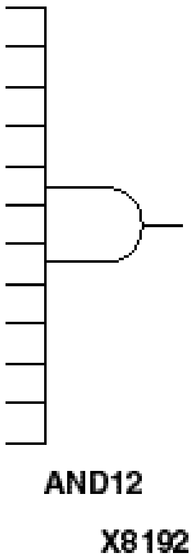
Figure 3-28AND8 Implementation Virtex

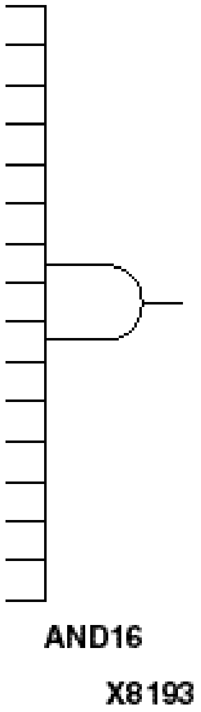


AND12, 16

12- and 16-Input AND Gates with Non-Inverted Inputs

Element	XC3000	XC4000E	XC4000X	XC5200	XC9000	Spartan	Spartan XL	Virtex
AND12, AND16	N/A	N/A	N/A	Macro	N/A	N/A	N/A	Macro





AND12 and AND16 functions are performed in the Configurable Logic Block (CLB) function generator.

The 12- and 16-input AND functions are available only with non-inverting inputs. To invert all of some inputs, use external inverters.

Refer to **"AND2-9"** for information on more AND functions.

Figure 3-29AND12 Implementation XC5200

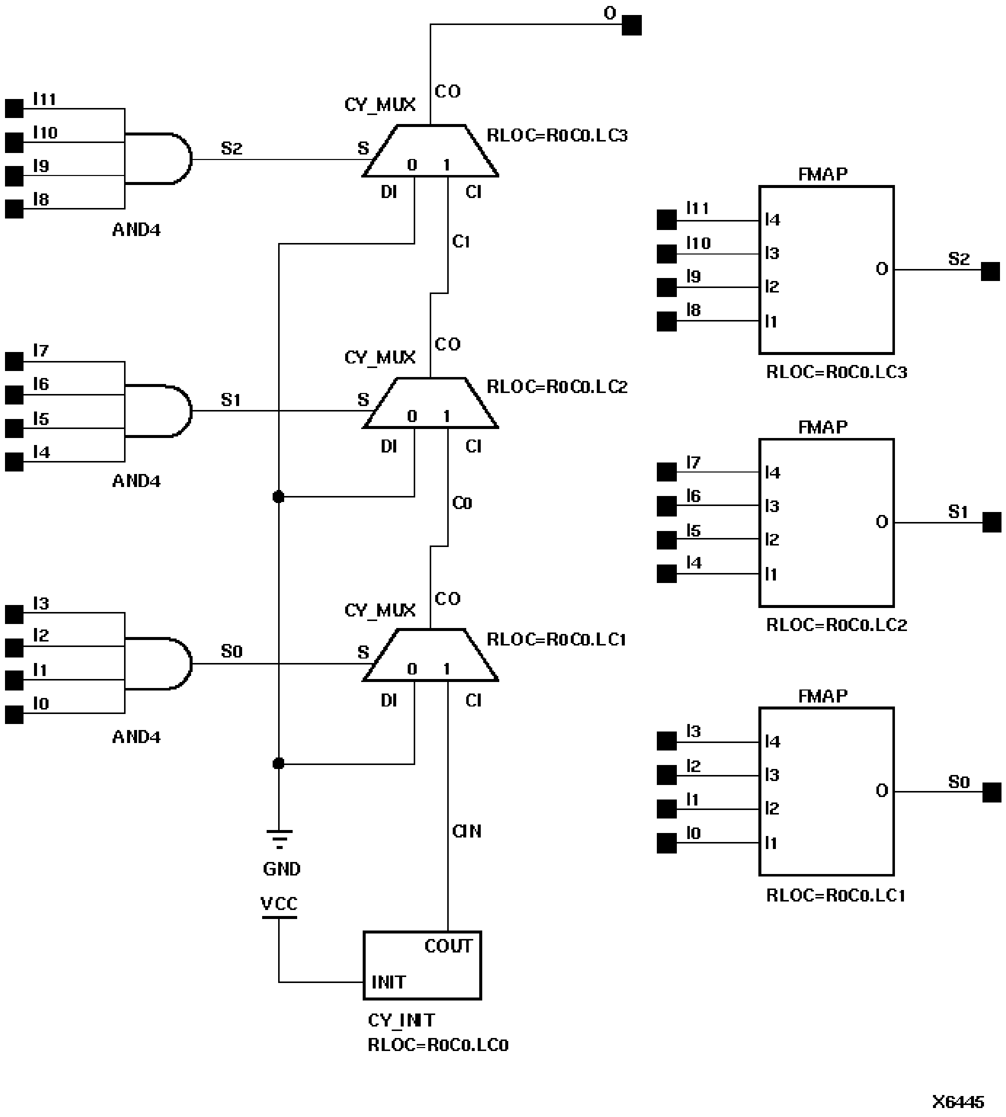


Figure 3-30AND12 Implementation Virtex

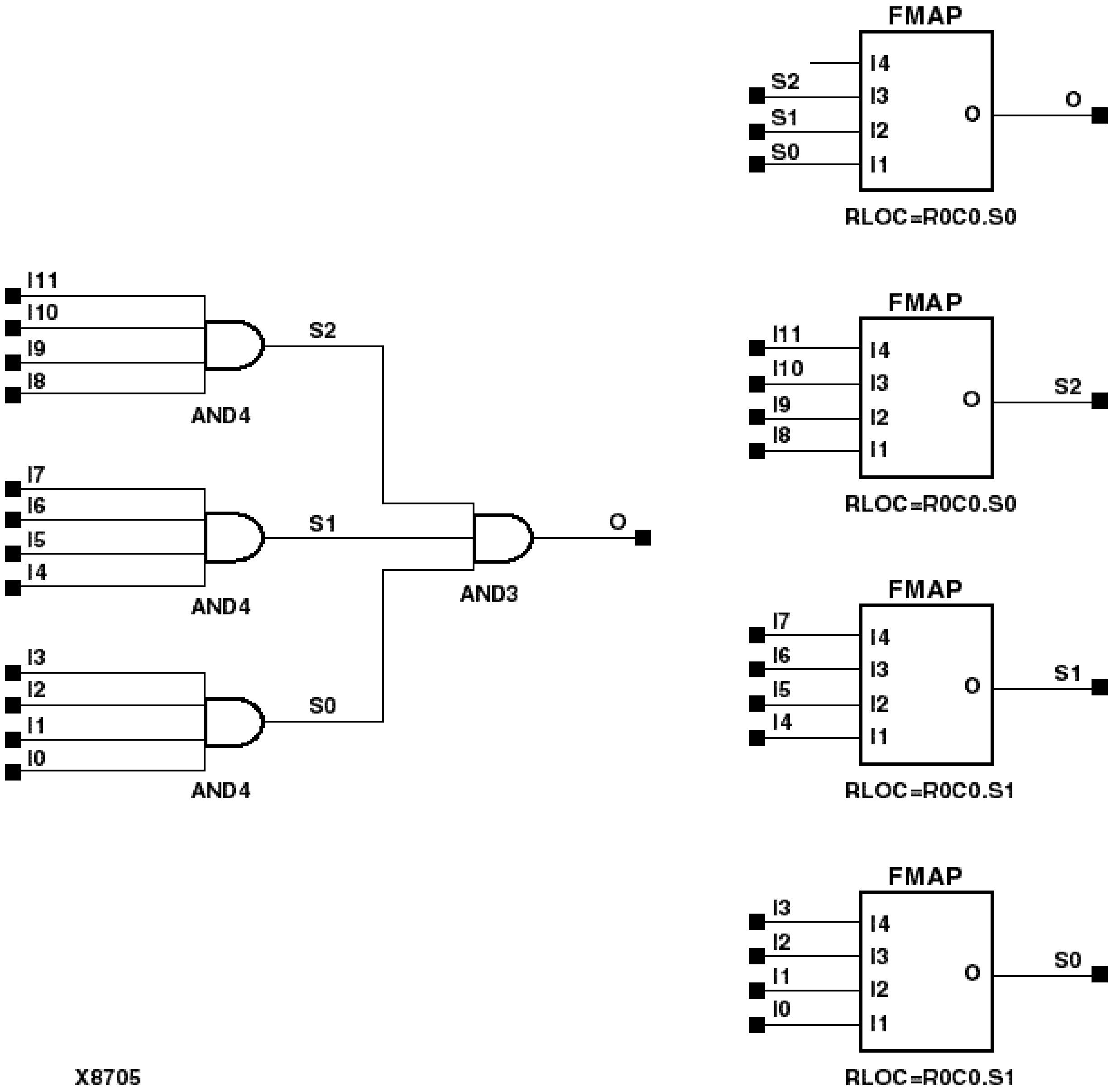


Figure 3-31AND16 Implementation XC5200

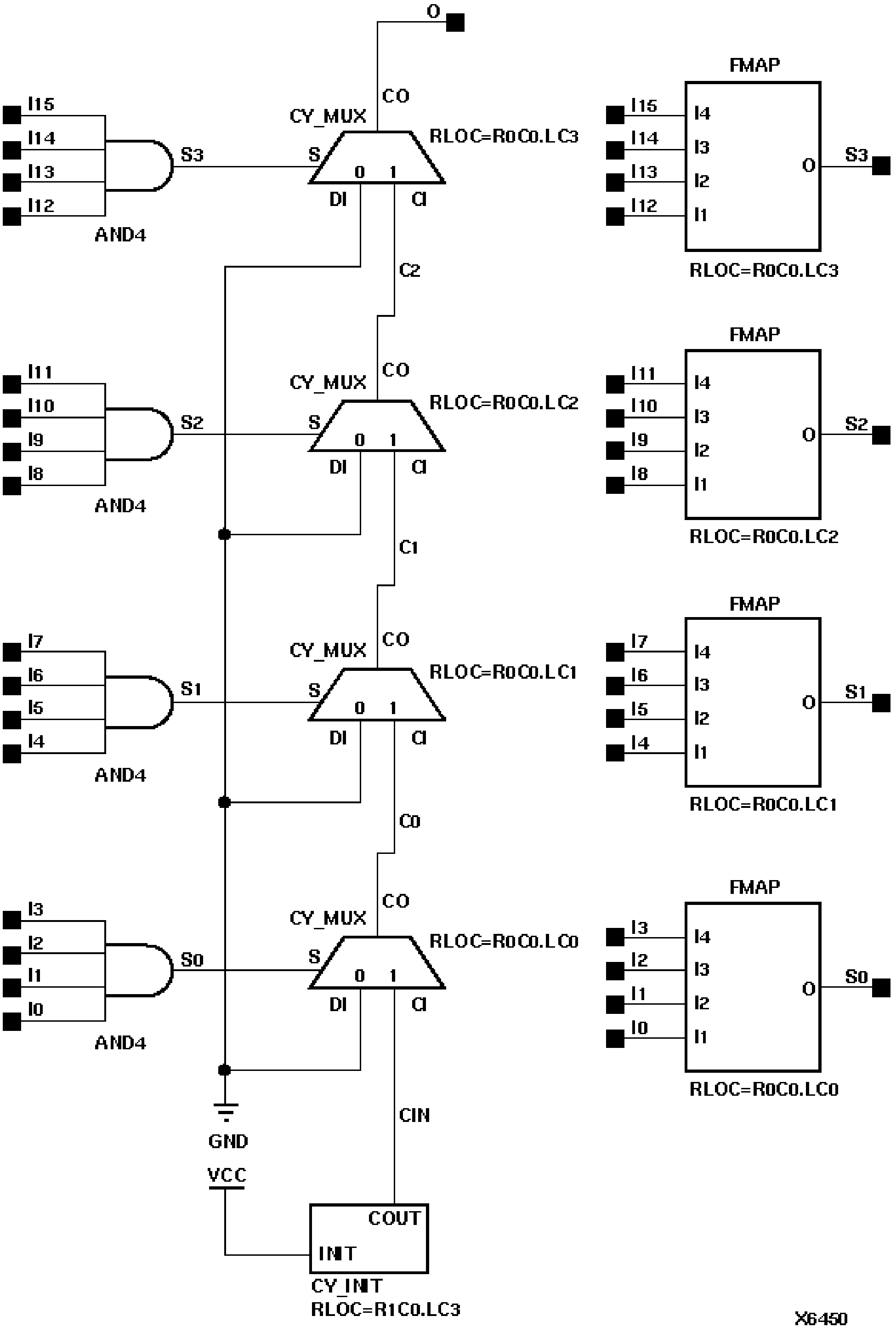
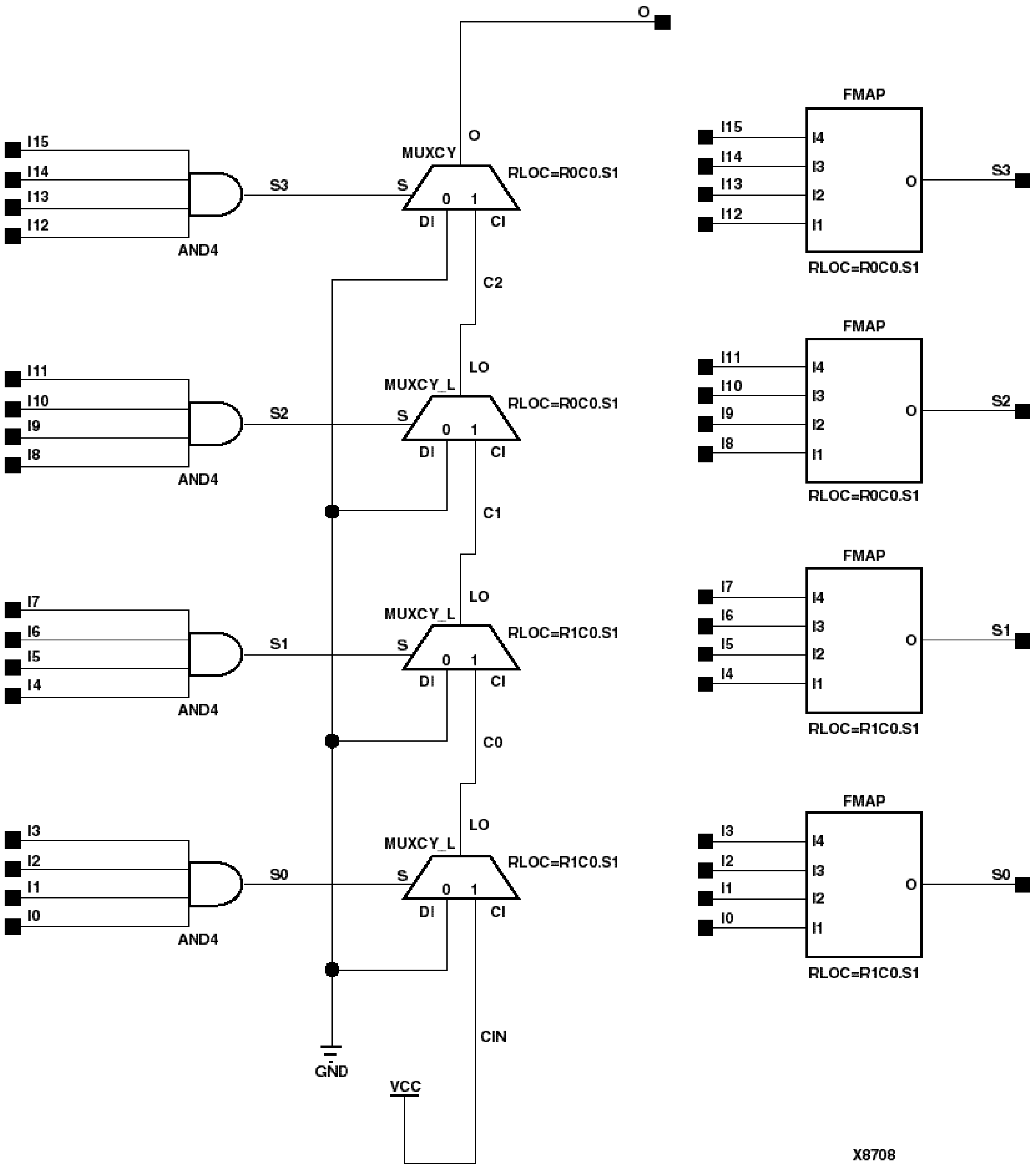


Figure 3-32AND16 Implementation Virtex

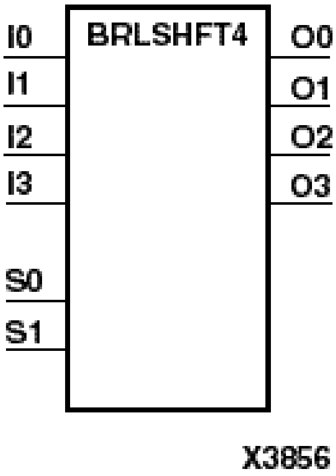


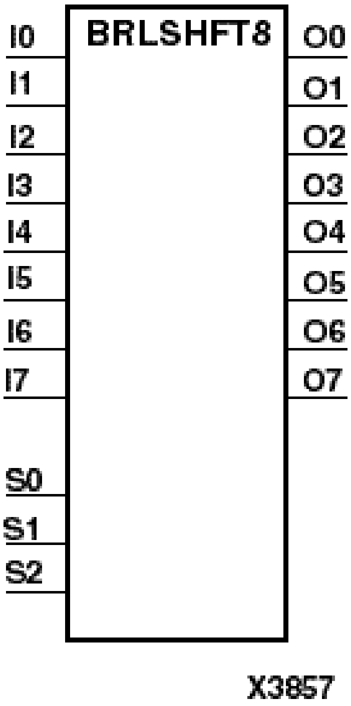
BRLSHFT4, 8

4-, 8-Bit Barrel Shifters

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex

Macro	Macro	Macro	Macro	Macro	Macro	Macro	Macro





BRLSHFT4, a 4-bit barrel shifter, can rotate four inputs (I3 – I0) up to four places. The control inputs (S1 and S0) determine the number of positions, from one to four, that the data is rotated. The four outputs (O3 – O0) reflect the shifted data inputs.

BRLSHFT8, an 8-bit barrel shifter, can rotate the eight inputs (I7 – I0) up to eight places. The control inputs (S2 – S0) determine the number of positions, from one to eight, that the data is rotated. The eight outputs (O7 – O0) reflect the shifted data inputs.

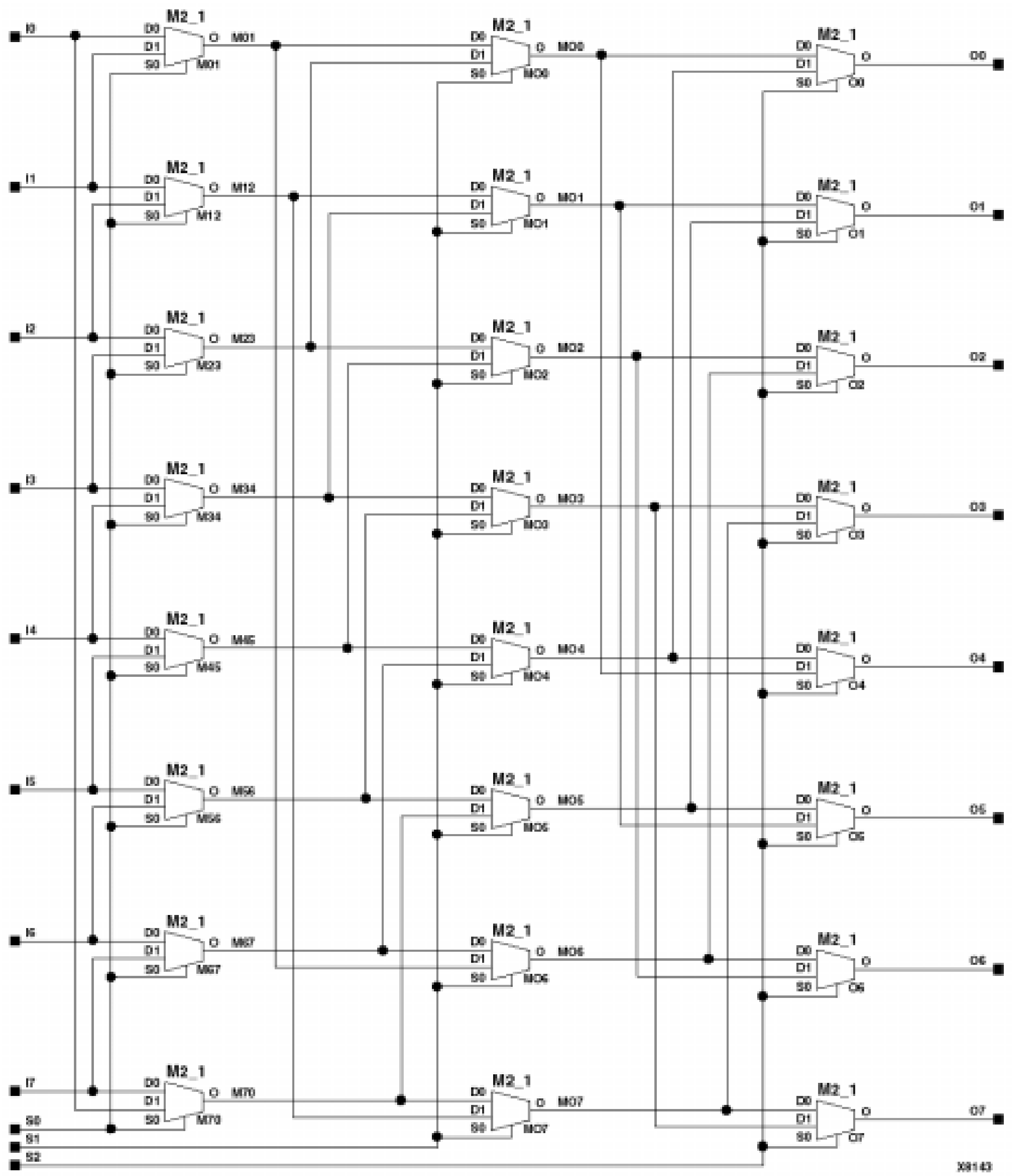
Inputs						Outputs			
S1	**S0**	**I0**	**I1**	**I2**	**I3**	**O0**	**O1**	**O2**	**O3**
0	0	a	b	c	d	a	b	c	d
0	1	a	b	c	d	b	c	d	a
1	0	a	b	c	d	c	d	a	b
1	1	a	b	c	d	d	a	b	c

BRLSHFT4 Truth Table

Inputs											Output							
S2	**S1**	**S0**	**I0**	**I1**	**I2**	**I3**	**I4**	**I5**	**I6**	**I7**	**O0**	**O1**	**O2**	**O3**	**O4**	**O5**	**O6**	**O7**
0	0	0	a	b	c	d	e	f	g	h	a	b	c	d	e	f	g	h
0	0	1	a	b	c	d	e	f	g	h	b	c	d	e	f	g	h	a
0	1	0	a	b	c	d	e	f	g	h	c	d	e	f	g	h	a	b
0	1	1	a	b	c	d	e	f	g	h	d	e	f	g	h	a	b	c
1	0	0	a	b	c	d	e	f	g	h	e	f	g	h	a	b	c	d
1	0	1	a	b	c	d	e	f	g	h	f	g	h	a	b	c	d	e
1	1	0	a	b	c	d	e	f	g	h	g	h	a	b	c	d	e	f
1	1	1	a	b	c	d	e	f	g	h	h	a	b	c	d	e	f	g

BRLSHFT8 Truth Table

Figure 3-33BRLSHFT8 Implementation XC3000, XC4000, XC5200, XC9000, Spartans, Virtex

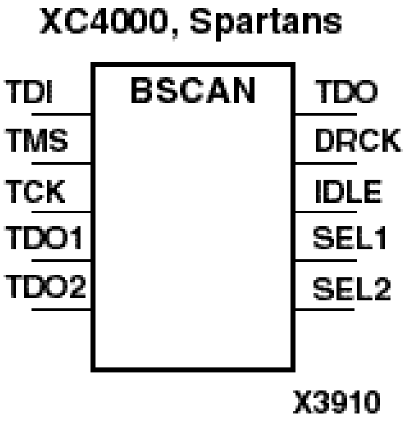


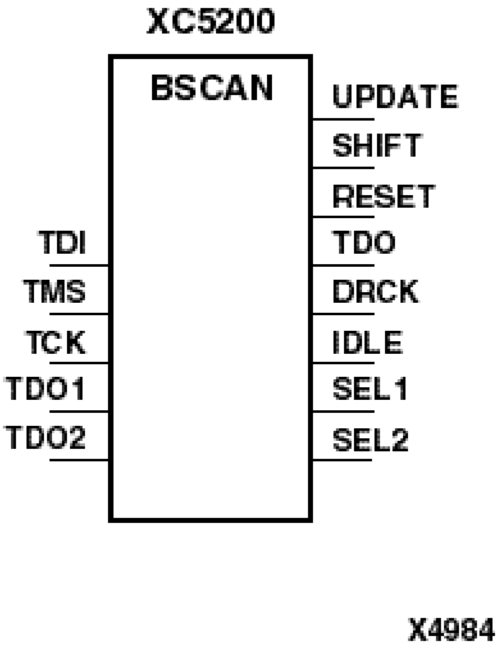
BSCAN

Boundary Scan Logic Control Circuit

XC3000	XC4000	XC4000	XC5200	XC9000	Spartan	Spartan	Virtex

	E	X				XL	
N/A	Primitive	Primitive	Primitive	N/A	Primitive	Primitive	N/A



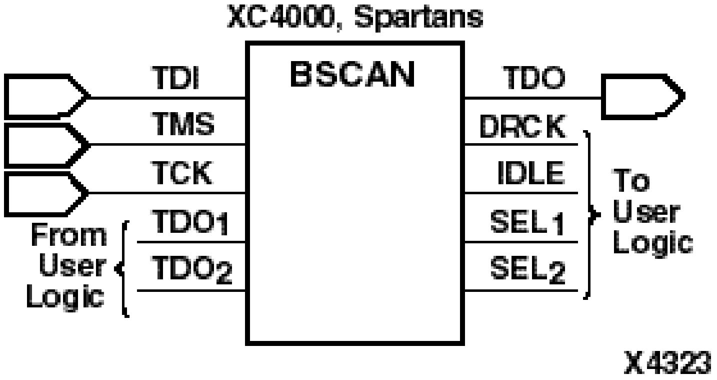


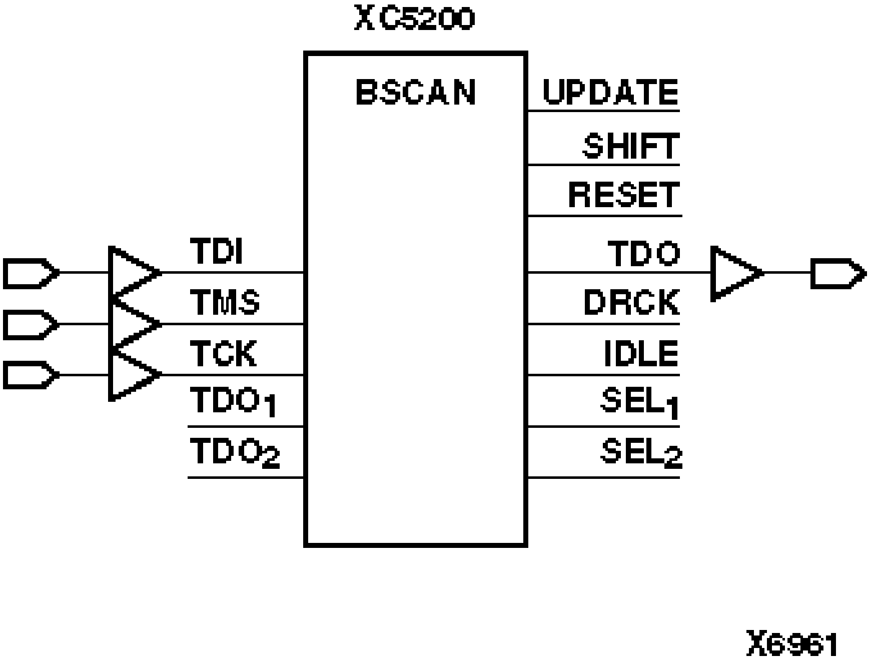
The BSCAN symbol indicates that boundary scan logic should be enabled after the programmable logic device (PLD) configuration is complete. It also provides optional access to some special features of the XC5200 boundary scan logic.

Note: For specific information on boundary scan for each architecture, refer to *The Programmable Logic Data Book*.

To indicate that boundary scan remains enabled after configuration, connect the BSCAN symbol to the TDI, TMS, TCK, and TDO pads. The other pins on BSCAN do not need to be connected, unless those special functions are needed. A signal on the TDO1 input is passed to the external TDO output when the USER1 instruction is executed; the SEL1 output goes High to indicate that the USER1 instruction is active. The TDO2 and SEL2 pins perform a similar function for the USER2 instruction. The DRCK output provides access to the data register clock (generated by the TAP controller). The IDLE output provides access to a version of the TCK input, which is only active while the TAP controller is in the Run-Test-Idle state. The RESET, UPDATE, and SHIFT pins of the XC5200 BSCAN symbol represent the decoding of the corresponding state of the boundary scan internal state machine. These functions are not available in the XC4000 and Spartans.

If boundary scan is used only before configuration is complete, do not include the BSCAN symbol in the design. The TDI, TMS, TCK, and TDO pins can be reserved for user functions.

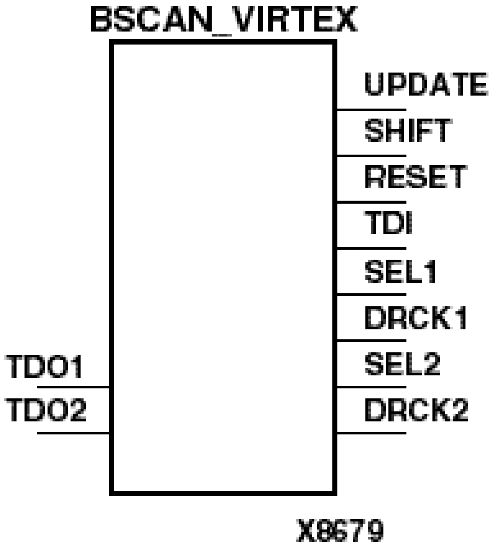




BSCAN_VIRTEX

Virtex Boundary Scan Logic Control Circuit

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	N/A	N/A	N/A	N/A	N/A	Primitive



The BSCAN_VIRTEX symbol indicates that boundary scan logic should be enabled after the programmable logic device (PLD) configuration is complete. The 4-pin JTAG interface (TDI, TDO, TCK, and TMS) are dedicated pins in Virtex. To use normal JTAG for boundary scan purposes, just hook up the JTAG pins to the port and go. The pins on the BSCAN_VIRTEX symbol do not need to be connected, unless those special functions are needed to drive an internal scan chain.

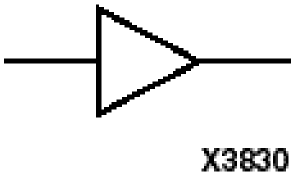
A signal on the TDO1 input is passed to the external TDO output when the USER1 instruction is executed; the SEL1 output goes High to indicate that the USER1 instruction is active.The DRCK1 output provides USER1 access to the data register clock (generated by the TAP controller). The TDO2 and SEL2 pins perform a similar function for the USER2 instruction and the DRCK2 output provides USER2 access to the data register clock (generated by the TAP controller). The RESET, UPDATE, and SHIFT pins represent the decoding of the corresponding state of the boundary scan internal state machine. The TDI pin provides access to the TDI signal of the JTAG port in order to shift data into an internal scan chain.

Note: For detailed information on boundary scan for Virtex, refer to the Xilinx web site, http://www.xilinx.com.

BUF

General-Purpose Buffer

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
Primitive	Primitive	Primitive	Primitive	Primitive	Primitive	Primitive	Primitive



BUF is a general purpose, non-inverting buffer.

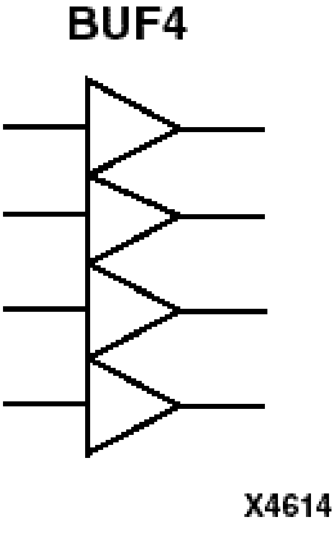
In FPGA architectures, BUF is usually not necessary and is removed by the partitioning software (MAP). The BUF element can be preserved for reducing the delay on a high fan-out net, for example, by splitting the net and reducing capacitive loading. In this case, the buffer is preserved by attaching an X (explicit) attribute to both the input and output nets of the BUF.

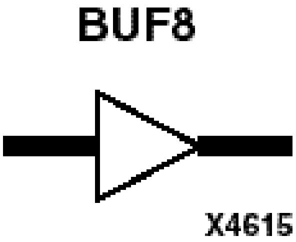
In CPLD architecture, BUF is usually removed, unless you inhibit optimization by applying the OPT=OFF attribute to the BUF symbol or by using the LOGIC_OPT=OFF global attribute.

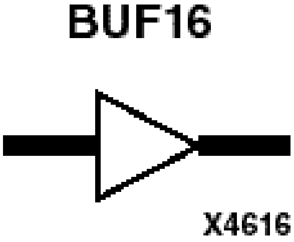
BUF4, 8, 16

General-Purpose Buffers

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	N/A	N/A	Macro	N/A	N/A	N/A



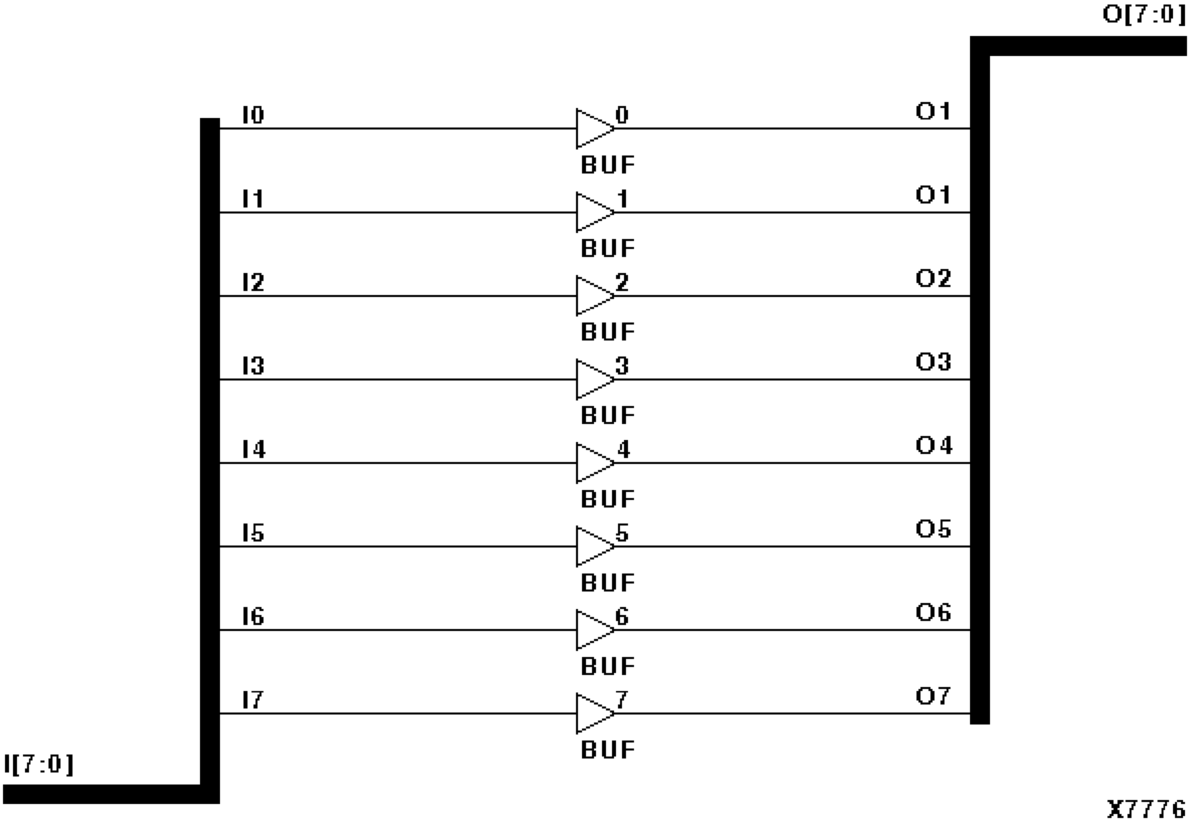




BUF4, 8, 16 are general purpose, non-inverting buffers.

In CPLD architecture, BUF4, BUF8, and BUF16 are usually removed, unless you inhibit optimization by applying the OPT=OFF attribute to the BUF4, BUF8, or BUF16 symbol or by using the LOGIC_OPT=OFF global attribute.

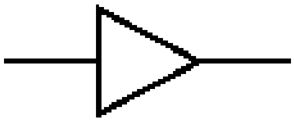
Figure 3-34BUF8 Implementation XC9000



BUFCF

Fast Connect Buffer

XC3000	XC4000E	XC4000X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	N/A	N/A	N/A	N/A	N/A	Primitive



X3830

BUFCF is a single fast connect buffer used to connect the outputs of the LUTs and some dedicated logic directly to the input of another LUT. Using this buffer implies CLB packing. No more than four LUTs may be connected together as a group.

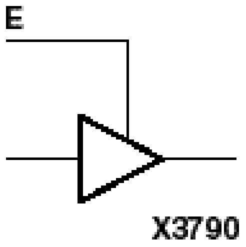
BUFE, 4, 8, 16

Internal 3-State Buffers with Active High Enable

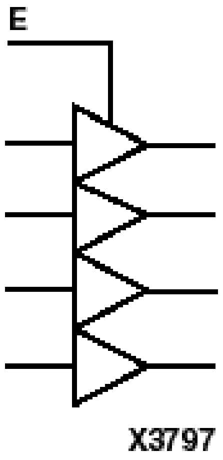
Element	XC3000	XC4000E	XC4000X	XC5200	XC9000	Spartan	Spartan XL	Virtex
BUFE	Macro	Macro	Macro	Macro	Primitive*	Macro	Macro	Primitive
BUFE4, BUFE8, BUFE16	Macro	Macro	Macro	Macro	Macro*	Macro	Macro	Macro

* not supported for XC9500XL devices

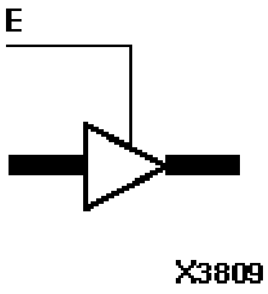
BUFE

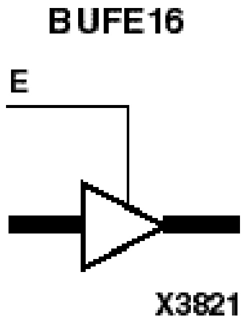


BUFE4



BUFE8





BUFE, BUFE4, BUFE8, and BUFE16 are single or multiple tristate buffers with inputs I, I3 – I0, I7 – I0, and I15 – I0, respectively; outputs O, O3 – O0, O7 – O0, and O15 – O0, respectively; and active-High output enable (E). When E is High, data on the inputs of the buffers is transferred to the corresponding outputs. When E is Low, the output is high impedance (Z state or Off). The outputs of the buffers are connected to horizontal longlines in FPGA architectures.

The outputs of separate BUFE symbols can be tied together to form a bus or a multiplexer. Make sure that only one E is High at any one time. If none of the E inputs is active-High, a "weak-keeper" circuit (FPGAs) keeps the output bus from floating but does not guarantee that the bus remains at the last value driven onto it.

In XC3000, XC4000, and Spartans, the E signal in BUFE macros is implemented by using a BUFT with an inverter on the active-Low enable (T) pin. This inverter can add an extra level of logic to the data path. Pull-up resistors can be used to establish a High logic level if all BUFE elements are Off.

In the XC5200 architecture, pull-ups cannot be used in conjunction with BUFT or BUFE macros because there are no pull-ups available at the ends of the horizontal longlines.

For XC9500 devices, BUFE output nets assume the High logic level when all connected BUFE/BUFT buffers are disabled. On-chip 3-state multiplexing is not available in XC9500XL devices.

For Virtex, BUFE elements need a PULLUP element connected to their output. NGDBuild inserts a PULLUP element if one is not connected.

Inputs		**Outputs**
E	**I**	**O**
0	X	Z
1	1	1
1	0	0

Figure 3-35BUFE Implementation XC3000, XC4000, XC5200, Spartans

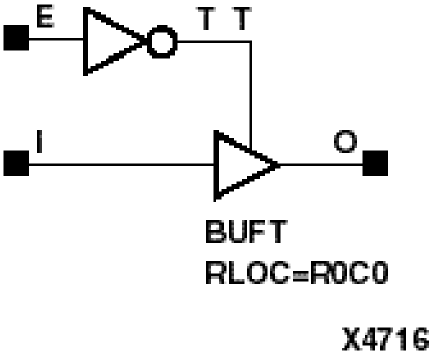


Figure 3-36BUFE8 Implementation XC3000, XC4000, XC5200, Spartans

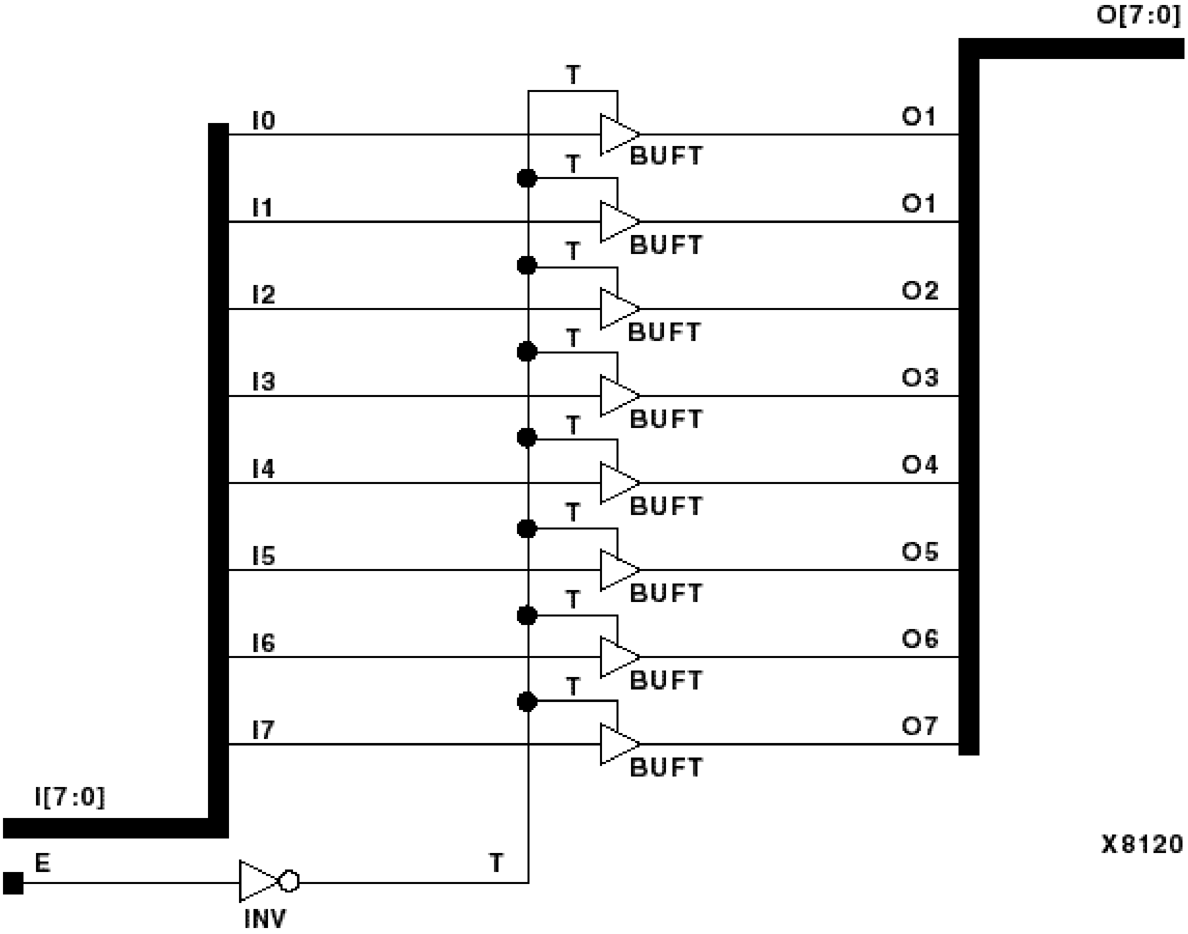
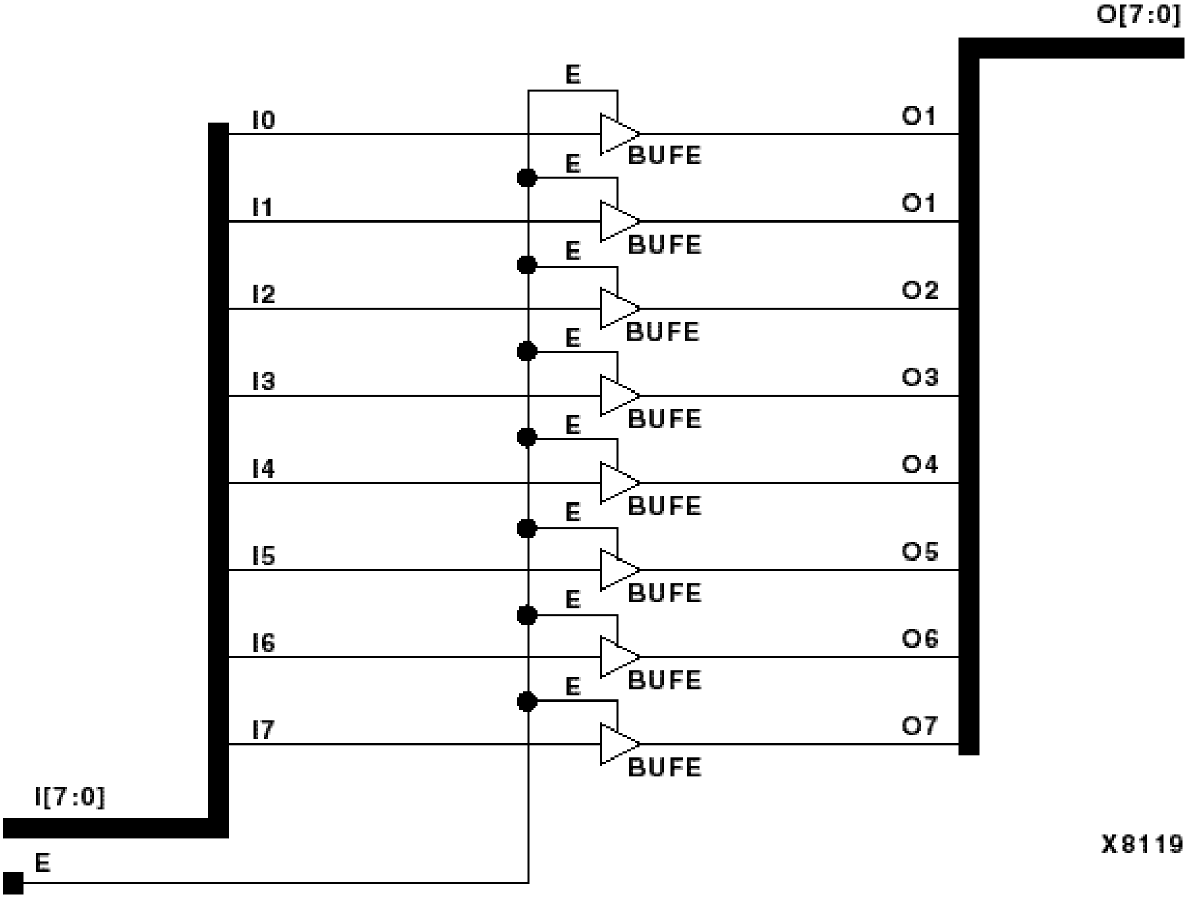


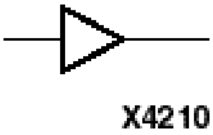
Figure 3-37BUFE8 Implementation XC9000, Virtex



BUFFCLK

Global Fast Clock Buffer

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	Primitive	N/A	N/A	N/A	N/A	N/A



BUFFCLK (FastCLK buffer) provides the fastest way to bring a clock into the XC4000X device. Four of these buffers are present on those devices — two on the left edge of the die and two on the right edge.

Using BUFFCLK, you can create a very fast pin-to-pin path by driving the F input of the CLB function generator with BUFFCLK output.

You can use BUFFCLK to minimize the setup time of input devices if positive hold time is acceptable. Use BUFFCLK to clock the Fast Capture latch and a slower clock buffer to capture the standard IOB flip-flop or latch. Either the Global Early buffer (**BUFGE**) or the Global Low-Skew buffer (**BUFGLS**) can be used for the second storage element (the one used should be the same clock as the internal related logic).

You can also use BUFFCLK to provide a fast Clock-to-Out on device output pins.

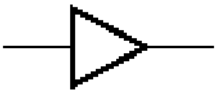
These buffers can access IOBs on one half of the die edge only. They can each drive two of the four vertical lines accessing the IOBs on the left edge of the device or two of the eight vertical lines accessing the IOBs on the right edge of the device. They can only access the CLB array through single and double-length lines.

BUFFCLKs must be driven by the semi-dedicated IOBs. They are not accessible from internal nets.

BUFG

Global Clock Buffer

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
Primitive	Primitive	Primitive	Primitive	Primitive	Primitive	Primitive	Primitive



X3831

BUFG, an architecture-independent global buffer, distributes high fan-out clock signals throughout a PLD device. The Xilinx implementation software converts each BUFG to an appropriate type of global buffer for the target PLD device.

For XC9000 designs, consult the device data sheet for the number of available global pins.

For an XC3000 design, you can use a maximum of two BUFG symbols (assuming that no specific GCLK or ACLK buffer is specified). For an XC4000 or Spartan series design, you can use a maximum of eight BUFG symbols (assuming that no specific BUFGP or BUFGS buffers are specified). For XC3000 designs, MAP always selects an ACLK, then a GCLK. For XC4000 or Spartan series designs, it always selects a BUFGS before a BUFGP. If you want to use a specific type of buffer, instantiate it manually.

To use a BUFG in a schematic, connect the input of the BUFG symbol to the clock source. Depending on the target PLD family, the clock source can be an external PAD symbol, an IBUF symbol, or internal logic. In Virtex, the BUFG cannot be driven directly from a pad; it can be driven from an IBUFG instead. For a negative-edge clock input, insert an INV (inverter) symbol between the BUFG output and the clock input. The inversion is implemented at the Configurable Logic Block (CLB) or Input Output Block (IOB) clock pin.

For XC9000 designs, BUFG is always implemented using an IOB. Connect the input of BUFG to an IPAD or an

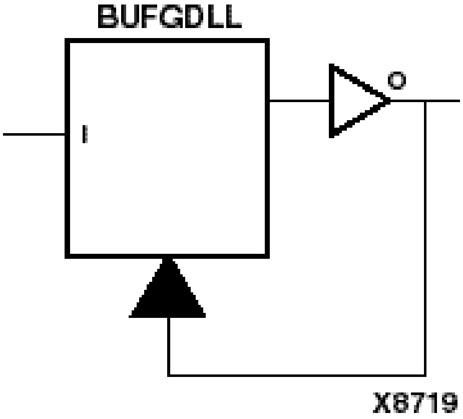
IOPAD that represents an external signal source. Each BUFG can drive any number of register clocks in a designs.

For XC9000 designs, the output of a BUFG may also be used as an ordinary input signal to other logic elsewhere in the design.

BUFGDLL

Clock Delay Locked Loop Buffer

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	N/A	N/A	N/A	N/A	N/A	Primitive

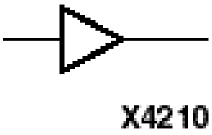


BUFGDLL is a special purpose clock delay locked loop buffer for clock skew management. It is provided as a user convenience for the most frequently used configuration of elements for clock skew management. It consists of an IBUFG followed by a CLKDLL followed by a BUFG.

BUFGE

Global Low Early Clock Buffer

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	Primitive	N/A	N/A	N/A	N/A	N/A



Eight Global Early buffers (BUFGE), two on each corner of the device, provide an earlier clock access than the potentially heavily loaded Global Low-Skew buffers (**BUFGLS**).

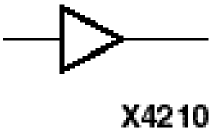
BUFGE can facilitate the fast capture of device inputs using the Fast Capture latches **ILFFX** and **ILFLX**. For fast capture, take a single clock signal and route it through both a BUFGE and a BUFGLS. Use the BUFGE to clock the fast capture latch and the BUFGLS to clock the normal input flip-flop or latch.

You can also use BUFGE to provide a fast Clock-to-Out on device output pins.

BUFGLS

Global Low Skew Clock Buffer

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	Primitive	N/A	N/A	N/A	Primitive	N/A



Each corner of the XC4000X or SpartanXL device has two Global Low-Skew buffers (BUFGLS). Any of the eight buffers can drive any of the eight vertical global lines in a column of CLBs. In addition, any of the buffers can drive any of the four vertical lines accessing the IOBs on the left edge of the device and any of the eight vertical lines accessing the IOBs on the right edge of the device.

IOBs at the top and bottom edges of the device are accessed through the vertical global lines in the CLB array. Any global low-skew buffer can, therefore, access every IOB and CLB in the device.

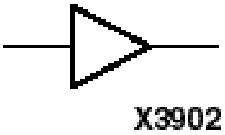
The global low-skew buffers can be driven by either semi-dedicated pads or internal logic.

BUFGP

Primary Global Buffer for Driving Clocks or Longlines (Four per PLD Device)

XC3000	XC4000	XC4000	XC5200	XC9000	Spartan	Spartan	Virtex

	E	X				XL	
N/A	Primitive	N/A	Macro	N/A	Primitive	N/A	Primitive



BUFGP, a primary global buffer, is used to distribute high fan-out clock or control signals throughout PLD devices. In Virtex, BUFGP is equivalent to an IBUFG driving a BUFG. In CPLD designs, BUFGP is treated like BUFG. A BUFGP provides direct access to Configurable Logic Block (CLB) and Input Output Block (IOB) clock pins and limited access to other CLB inputs. Four BUFGPs are available on each XC4000E and Spartan device, one in each corner. The input to a BUFGP comes only from a dedicated IOB.

Alongside each column of CLBs in an XC4000E or Spartan device are four global vertical lines, which are in addition to the standard vertical longlines. Each one of the four global vertical lines can drive the CLB clock (K) pin directly. In addition, one of the four lines can drive the F3 pin, a second line can drive the G1 pin, a third can drive the C3 pin, and a fourth can drive the C1 pin. Each of the four BUFGPs drives one of these global vertical lines. These same vertical lines are also used for the secondary global buffers (refer to the **"BUFGS"** section for more information).

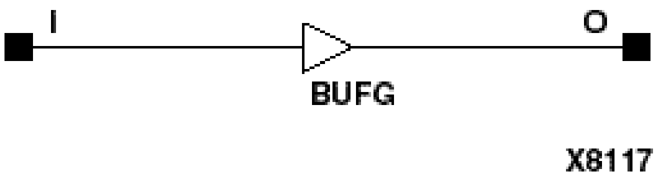
Because of its structure, a BUFGP can always access a clock pin directly. However, it can access only one of the F3, G1, C3, or C1 pins, depending on the corner in which the BUFGP is placed. When the required pin cannot be accessed directly from the vertical line, PAR feeds the signal through another CLB and uses general purpose routing to access the load pin.

To use a BUFGP in a schematic, connect the input of the BUFGP element directly to the PAD symbol. Do not use any IBUFs, because the signal comes directly from a dedicated IOB. The output of the BUFGP is then used throughout the schematic. For a negative-edge clock, insert an INV (inverter) element between the output of the BUFGP and the clock input. This inversion is performed inside each CLB or IOB.

A Virtex BUFGP must be sourced by an external signal. Other BUFGPs can be sourced by an internal signal, but PAR must use the dedicated IOB to drive the BUFGP, which means that the IOB is not available for use by other signals. If possible, use a BUFGS instead, because they can be sourced internally without using an IOB.

The dedicated inputs for BUFGPs are identified by the names PGCK1 through PGCK4 in pinouts in XC4000E and Spartan. The package pin that drives the BUFGP depends on which corner the BUFGP is placed by
PAR.

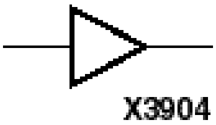
Figure 3-38BUFGP Implementation XC5200



BUFGS

Secondary Global Buffer for Driving Clocks or Longlines (Four per PLD Device)

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	Primitive	N/A	Macro	N/A	Primitive	N/A	N/A



BUFGS, a secondary global buffer, distributes high fan-out clock or control signals throughout a PLD device. In CPLD designs, BUFGS is treated like BUFG. BUFGS provides direct access to Configurable Logic Block (CLB) clock pins and limited access to other CLB inputs. Four BUFGSs are available on each XC4000E and Spartan device, one in each corner. The input to a BUFGS comes either from a dedicated Input Output Block (IOB) or from an internal signal.

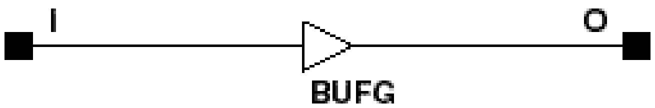
Alongside each column of CLBs in an XC4000E or Spartan device are four global vertical lines, which are in addition to the standard vertical longlines. Each one of the four global vertical lines can drive the CLB clock (K) pin directly. In addition, one of the four lines can drive the F3 pin, a second line can drive the G1 pin, a third can drive the C3 pin, and a fourth can drive the C1 pin. Each of the four BUFGSs can drive any of these global vertical lines and are also used as the primary global buffers (refer also to the **"BUFGP"** section for more information).

Because of its structure, a BUFGS can always access a clock pin directly. Because the BUFGS is more flexible than the BUFGP, it can use additional global vertical lines to access the F3, G1, C3, and C1 pins but requires multiple vertical lines in the same column. If the vertical lines in a given column are already used for BUFGPs or another BUFGS, PAR might have to feed signals through other CLBs to reach the load pins.

To use a BUFGS in a schematic, connect the input of the BUFGS element either directly to the PAD symbol (for an external input) or to an internally sourced net. For an external signal, do not use any IBUFs, because the signal comes directly from the dedicated IOB. The output of the BUFGS is then used throughout the schematic. For a negative-edge clock, insert an INV (inverter) element between the output of the BUFGS and the clock input. This inversion is performed inside each CLB or IOB.

The dedicated inputs for BUFGSs are identified by the names SGCK1 through SGCK4 in pinouts in XC4000E and Spartan. The package pin that drives the BUFGS depends on which corner the BUFGS is placed by PAR.

Figure 3-39BUFGS Implementation XC5200

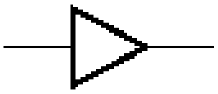


X8117

BUFGSR

Global Set/Reset Input Buffer

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	N/A	N/A	Primitive	N/A	N/A	N/A



X3831

BUFGSR, an XC9000-specific global buffer, distributes global set/reset signals throughout selected flip-flops of an XC9000 device. Global Set/Reset (GSR) control pins are available on XC9000 devices; consult device data sheets for availability.

BUFGSR always acts as an input buffer. To use it in a schematic, connect the input of the BUFGSR symbol to an IPAD or an IOPAD representing the GSR signal source. GSR signals generated on-chip must be passed through an OBUF-type buffer before they are connected to BUFGSR.

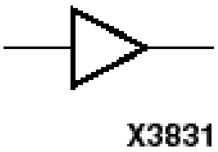
For global set/reset control, the output of BUFGSR normally connects to the CLR or PRE input of a flip-flop symbol, like FDCP, or any registered symbol with asynchronous clear or preset. The global set/reset control signal may pass through an inverter to perform an active-low set/reset. The output of BUFGSR may also be used as an ordinary input signal to other logic elsewhere in the design. Each BUFGSR can control any number of flip-flops in a design.

BUFGTS

Global Three-State Input Buffer

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex

N/A	N/A	N/A	N/A	Primitive	N/A	N/A	N/A



BUFGTS, an XC9000-specific global buffer, distributes global output-enable signals throughout the output pad drivers of an XC9000 device. Global Three-State (GTS) control pins are available on XC9000 devices; consult device data sheets for availability.

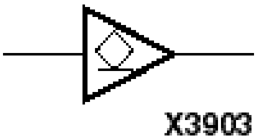
BUFGTS always acts as an input buffer. To use it in a schematic, connect the input of the BUFGTS symbol to an IPAD or an IOPAD representing the GTS signal source. GTS signals generated on-chip must be passed through an OBUF-type buffer before they are connected to BUFGTS.

For global 3-state control, the output of BUFGTS normally connects to the E input of a 3-state output buffer symbol, OBUFE. The global 3-state control signal may pass through an inverter or control an OBUFT symbol to perform an active-low output-enable. The same 3-state control signal may even be used both inverted and non-inverted to enable alternate groups of device outputs. The output of BUFGTS may also be used as an ordinary input signal to other logic elsewhere in the design. Each BUFGTS can control any number of output buffers in a design.

BUFOD

Open-Drain Buffer

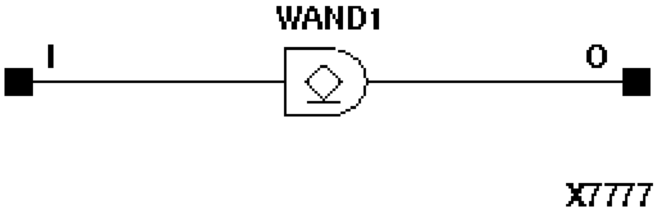
XC3000	**XC4000 E**	**XC4000 X**	**XC5200**	**XC9000**	**Spartan**	**Spartan XL**	**Virtex**
N/A	Macro	Macro	N/A	N/A	N/A	N/A	N/A



BUFOD is a buffer with input (I) and open-drain output (O). When the input is Low, the output is Low. When the input is High, the output is off. To establish an output High level, a pull-up resistor is tied to output O. One pull-up resistor uses the least power; two pull-up resistors achieve the fastest Low-to-High speed.

To indicate two pull-up resistors, append a DOUBLE parameter to the pull-up symbol attached to the output (O) node. Refer to the appropriate CAE tool interface user guide for details.

Figure 3-40BUFOD Implementation XC4000

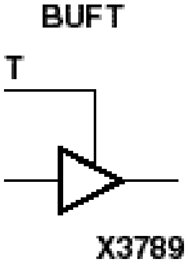


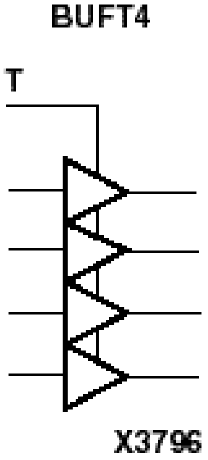
BUFT, 4, 8, 16

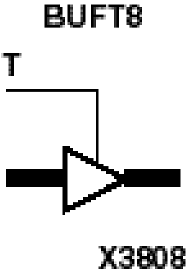
Internal 3-State Buffers with Active-Low Enable

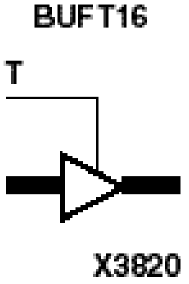
Element	XC3000	XC4000E	XC4000X	XC5200	XC9000	Spartan	Spartan XL	Virtex
BUFT	Primitive	Primitive	Primitive	Primitive	Primitive*	Primitive	Primitive	Primitive
BUFT4, BUFT8, BUFT16	Macro	Macro	Macro	Macro	Macro*	Macro	Macro	Macro

* not supported for XC9500XL devices









BUFT, BUFT4, BUFT8, and BUFT16 are single or multiple 3-state buffers with inputs I, I3 – I0, I7 – I0, and I15 – 10, respectively; outputs O, O3 – O0, O7 – O0, and O15 – O0, respectively; and active-Low output enable (T). When T is Low, data on the inputs of the buffers is transferred to the corresponding outputs. When T is High, the output is high impedance (Z state or off). The outputs of the buffers are connected to horizontal longlines in FPGA architectures.

The outputs of separate BUFT symbols can be tied together to form a bus or a multiplexer. Make sure that only one T is Low at one time. If none of the T inputs is active (Low), a "weak-keeper" circuit (FPGAs) prevents the output bus from floating but does not guarantee that the bus remains at the last value driven onto it.

Pull-up resistors can be used to establish a High logic level if all BUFT elements are off in XC3000, XC4000, and Spartans.

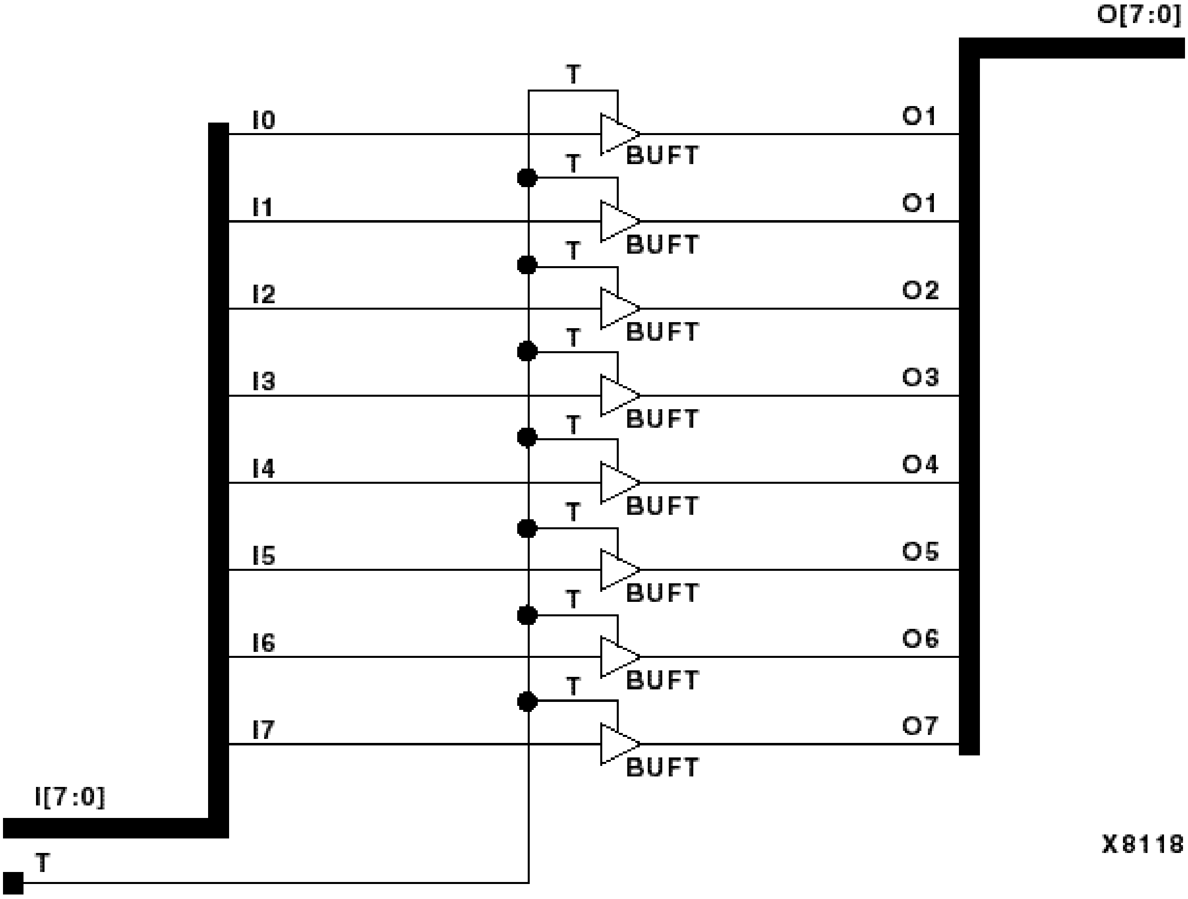
In the XC5200 architecture, pull-ups cannot be used in conjunction with BUFT or BUFE macros because there are no pull-ups available at the ends of the horizontal longlines.

For XC9500 devices, BUFT output nets assume the High logic level when all connected BUFE/BUFT buffers are disabled. On-chip 3-state multiplexing is not available in XC9500XL devices.

For Virtex, when all BUFTs on a net are disabled, the net is High. For correct simulation of this effect, a PULLUP element must be connected to the net. NGDBuild inserts a PULLUP element if one is not connected so that back-annotation simulation reflects the true state of the Virtex chip.

Inputs		**Outputs**
T	**I**	**O**
1	X	Z
0	1	1
0	0	0

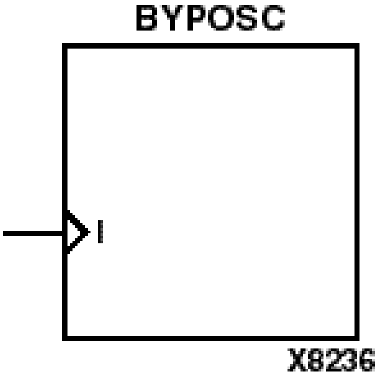
Figure 3-41BUFT8 Implementation XC3000, XC4000, XC5200, XC9000, Spartans, Virtex



BYPOSC

Bypass Oscillator

XC3000	XC4000 E	XC4000 X	XC5200	XC9000	Spartan	Spartan XL	Virtex
N/A	N/A	N/A	Primitive	N/A	N/A	N/A	N/A



BYPOSC provides for definition of a user clock for the charge pump via its I pin. When the BYPOSC symbol is not used or its I pin is not connected, the charge pump uses an internal clock.