PLACE AND ROUTE LAYOUT DIAGRAMS

William Bricken
May 2002

This memo shows the Comesh block logic placement and routing layout for three functional circuits.

- 4-adder place and route
- 5-adder place and route
- two FSMs in the same block

4-ADDER PLACE AND ROUTE

William Bricken

April 2002

FEATURES

-- uses a 5x16 rectangular-comesh block
-- is carry look-ahead
-- is composable
-- quick and obvious layout
-- likely to be some parity errors, won't effect layout
-- page orientation only for convenience
-- ordering of cells and empty cells an artifact of convenience
-- draft visual only, not intended as a specification document

LEGEND

gi = ((ai)(bi)) = ai & bi

pi = ai bi = ai v bi

ci = carry

xi = ((ai bi)((ai)(bi))) = ai XOR bi

-name: select input inverter for this input

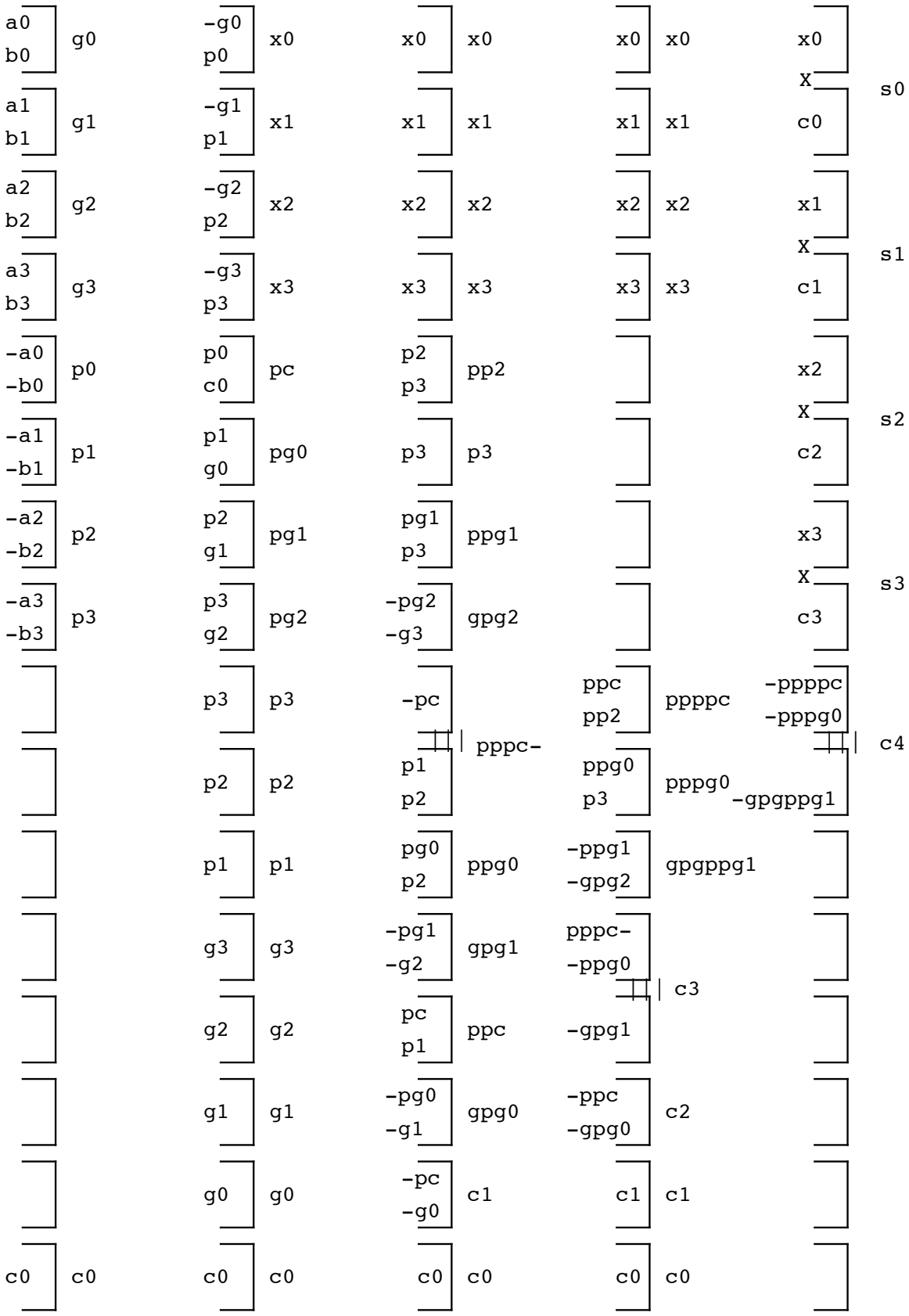
exotic-names: parts of the lookahead logic

X-between-cells: pair of cells used to form XOR (tier 5 only)

|||-between-cells: neighbor 2NAND used to create breadth

name-: only on pppc-, to remind that it is in reversed polarity

same-name-on-each-side-of-cell: pass-through


a0
b0
g0
a1
b1
g1
a2
b2
g2
a3
b3
g3
-a0
-b0
p0
-a1
-b1
p1
-a2
-b2
p2
-a3
-b3
p3
c0
c0
-g0
p0
x0
-g1
p1
x1
-g2
p2
x2
-g3
p3
x3
p0
c0
pc
p1
g0
pg0
p2
g1
pg1
p3
g2
pg2
p3
p3
p2
p2
p1
p1
g3
g3
g2
g2
g1
g1
g0
g0
c0
c0
x0
x0
x1
x1
x2
x2
x3
x3
p2
p3
pp2
p3
p3
pg1
p3
ppg1
-pg2
-g3
gpg2
-pc
pppc-
p1
p2
pg0
p2
ppg0
-pg1
-g2
gpg1
pc
p1
ppc
-pg0
-g1
gpg0
-pc
-g0
c1
c0
c0
x0
x0
x1
x1
x2
x2
x3
x3
ppc
pp2
pppc
ppg0
p3
pppg0
-ppg1
-gpg2
gpgppg1
pppc-
-ppg0
c3
-gpg1
-ppc
-gpg0
c2
c1
c1
c0
c0
x0
x
s0
c0
x1
x
s1
c1
x2
x
s2
c2
x3
x
s3
c3
-ppppc
-pppg0
c4
-gpgppg1

5-ADDER PLACE AND ROUTE

William Bricken

May 2002

PAGE 1: block-5add simply routes the 5bit adder logic through one block, using some neighbor sharing (5 cases) and some pass-through (7 cases). No feed-up. Carry from lookahead logic must go from lower logic block into upper bit block, presumably though superblock interconnect.

PAGE 2: block-5add-w includes the input line mux constraints, giving the following line usage:

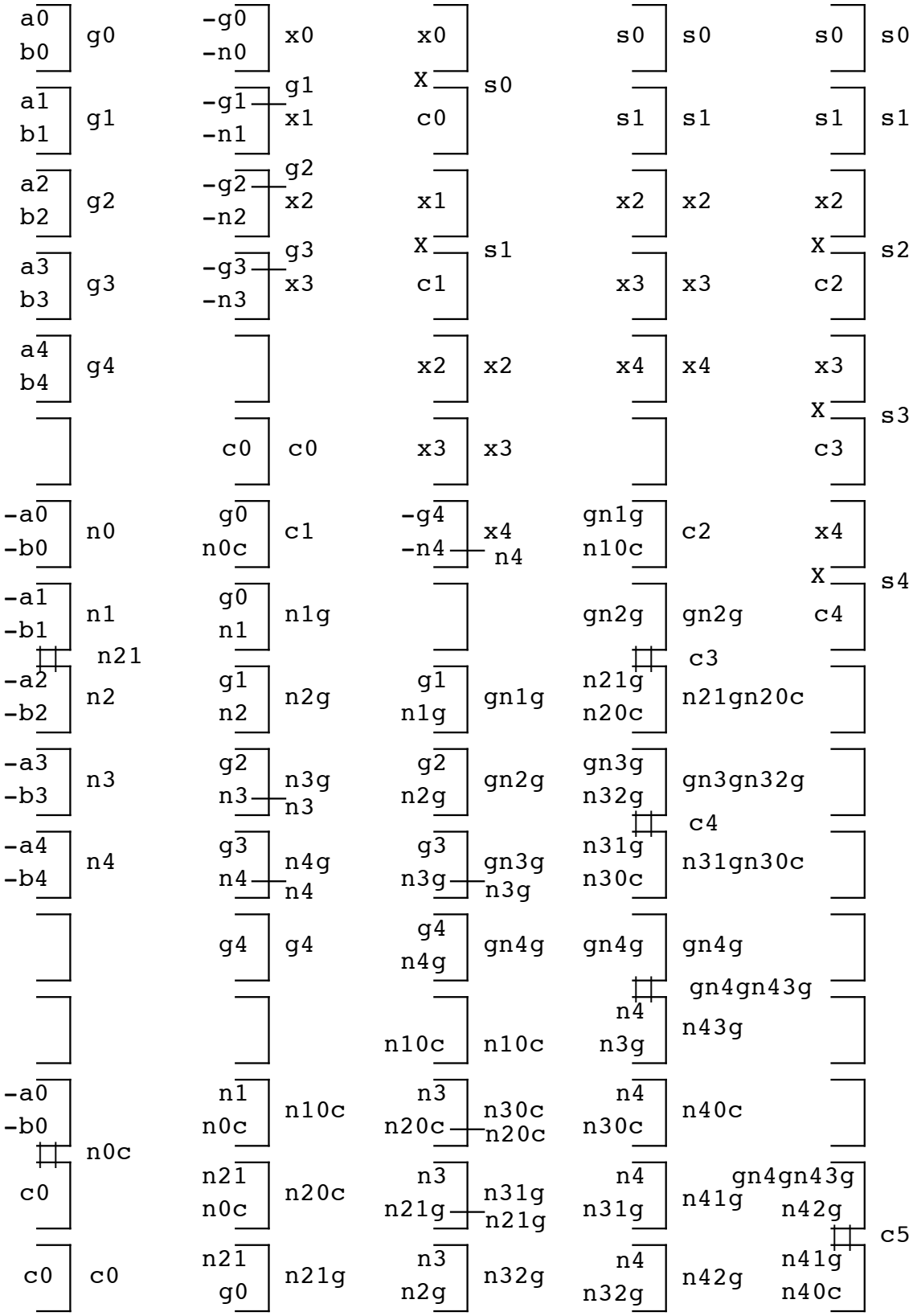
incoming	11
tier 1-2	20
tier 2-3	18
tier 3-4	18
tier 4-5	18
out	6

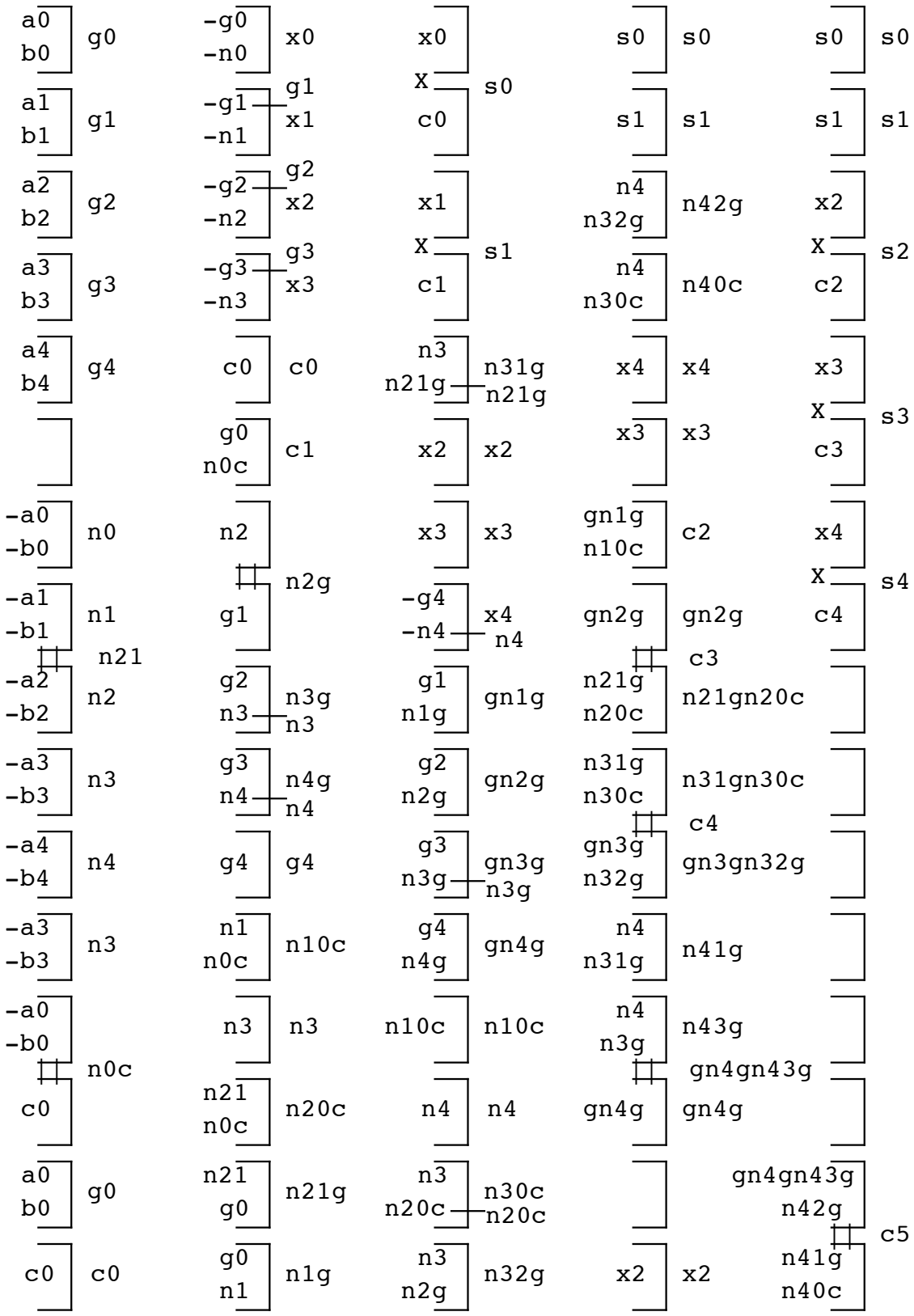
PAGE 3: block-5add-wp adjusts the polarities of signals, giving this inversion usage:

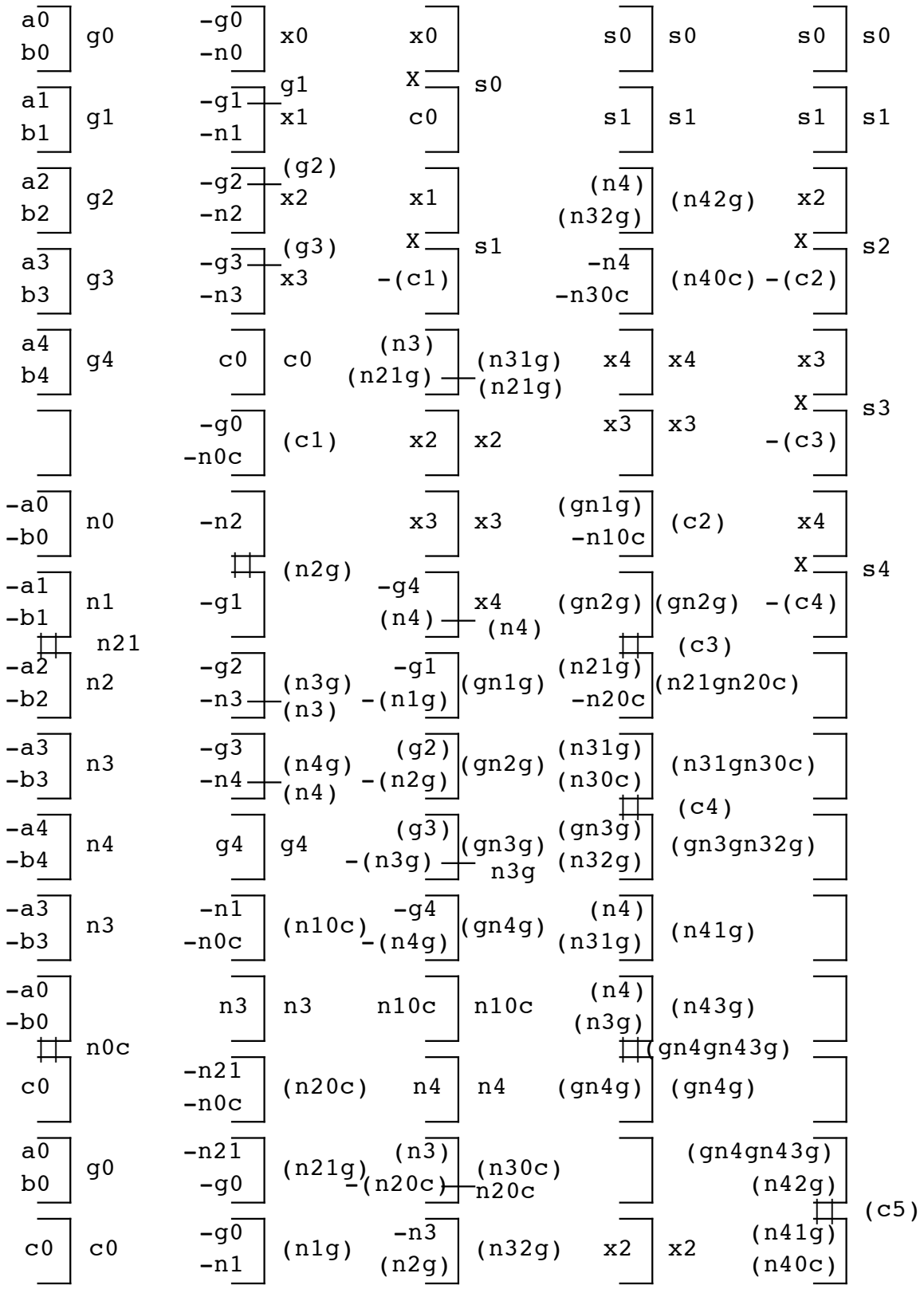
incoming	14
tier 1-2	24
tier 2-3	10
tier 3-4	4
tier 4-5	3
out	1

Notation:

id	names which hint at lookahead logic gi = ai AND bi = ((ai)(bi)) ni = ai NOR bi = (ai bi) xi = ai XOR bi = ((ai bi)((ai)(bi))) rest are OR combinations
-id	use inverter
||	use neighbor sharing
X	xor using neighbor
(id)	output is inverted wrt id-semantics


a0
b0
g0
a1
b1
g1
a2
b2
g2
a3
b3
g3
a4
b4
g4
-a0
-b0
n0
-a1
-b1
n1
n21
-a2
-b2
n2
-a3
-b3
n3
-a4
-b4
n4
-a0
-b0
n0c
c0
c0
c0
-g0
-n0
x0
-g1
-n1
g1
x1
-g2
-n2
g2
x2
-g3
-n3
g3
x3
c0
c0
g0
n0c
c1
g0
n1
n1g
g1
n2
g1
n2g
g2
n3
g2
n3g
n3
g3
n4
g3
n4g
n4
g4
g4
n1
n0c
n10c
n21
n0c
n20c
n21
g0
n21g
x0
X
s0
c0
x1
X
s1
c1
x2
x2
x3
x3
-g4
-n4
x4
n4
g1
n1g
gn1g
g2
n2g
gn2g
g3
n3g
gn3g
g4
n4g
gn4g
n10c
n10c
n3
n20c
n30c
n20c
n3
n21g
n31g
n21g
n3
n2g
n32g
gn1g
n10c
c2
gn2g
gn2g
c3
n21g
n20c
n21gn20c
gn3g
n32g
gn3gn32g
c4
n31g
n30c
n31gn30c
gn4g
gn4g
gn4gn43g
n4
n3g
n43g
n4
n30c
n40c
n4
n31g
n41g
gn4gn43g
n42g
c5
n4
n32g
n42g
n41g
n40c
s0
s0
s0
s0
s1
s1
s1
s1
x2
x2
x2
X
s2
x3
x3
c2
x3
X
s3
c3
x4
X
s4
c4


a0
b0
g0
a1
b1
g1
a2
b2
g2
a3
b3
g3
a4
b4
g4
-a0
-b0
n0
-a1
-b1
n1
n21
-a2
-b2
n2
-a3
-b3
n3
-a4
-b4
n4
-a3
-b3
n3
-a0
-b0
n0c
c0
a0
b0
g0
c0
c0
-g0
-n0
x0
g1
-g1
-n1
x1
g2
-g2
-n2
x2
g3
-g3
-n3
x3
c0
c0
g0
n0c
c1
n2
n2g
g1
g2
n3
g3
n4
g4
g4
n1
n0c
n10c
n3
n3
n21
n0c
n20c
n21
g0
n21g
g0
n1
g0
n1g
x0
X
s0
c0
x1
X
s1
c1
n3
n21g
n31g
n21g
x2
x2
x3
x3
-g4
-n4
x4
n4
g1
n1g
gn1g
g2
n2g
gn2g
g3
n3g
gn3g
n3g
g4
n4g
gn4g
n4
n3
n10c
n10c
n4
n4
n3
n20c
n3
n20c
n3
n2g
n32g
s0
s0
s1
s1
n4
n32g
n42g
n4
n30c
n40c
x4
x4
x3
x3
gn1g
n10c
c2
gn2g
gn2g
c3
n21g
n20c
n21gn20c
n31g
n30c
n31gn30c
c4
gn3g
n32g
gn3gn32g
n4
n31g
n41g
n4
n3g
n43g
gn4gn43g
gn4g
gn4g
x2
x2
s0
s0
s1
s1
x2
X
s2
c2
x4
X
s3
c3
x4
X
s4
c4
gn4gn43g
n42g
c5
n41g
n40c


a0
b0
g0
-g0
-n0
x0
x0
s0
s0
s0
s0
a1
b1
g1
-g1
-n1
g1
x1
x
s0
c0
s1
s1
s1
s1
a2
b2
g2
-g2
-n2
(g2)
x2
x1
(n4)
(n32g)
(n42g)
x2
a3
b3
g3
-g3
-n3
(g3)
x3
x
s1
-(c1)
-n4
-n30c
(n40c)
x
-(c2)
s2
a4
b4
g4
c0
c0
(n3)
(n21g)
(n31g)
(n21g)
x4
x4
x3
x
s3
-g0
-n0c
(c1)
x2
x2
x3
x3
-(c3)
-a0
-b0
n0
-n2
x3
x3
(gn1g)
-n10c
(c2)
x4
-a1
-b1
n1
-g1
(n2g)
-g4
(n4)
x4
(n4)
(gn2g)
(gn2g)
x
-(c4)
s4
-a2
-b2
n2
-g2
-n3
(n3g)
(n3)
-g1
-(n1g)
(gn1g)
(n21g)
-n20c
(c3)
(n21gn20c)
-a3
-b3
n3
-g3
-n4
(n4g)
(n4)
(g2)
-(n2g)
(gn2g)
(n31g)
(n30c)
(n31gn30c)
-a4
-b4
n4
g4
g4
(g3)
-(n3g)
(gn3g)
n3g
(gn3g)
(n32g)
(c4)
(gn3gn32g)
-a3
-b3
n3
-n1
-n0c
(n10c)
-g4
-(n4g)
(gn4g)
(n4)
(n31g)
(n41g)
-a0
-b0
n0c
n3
n3
n10c
n10c
(n4)
(n3g)
(n43g)
(gn4gn43g)
c0
-n21
-n0c
(n20c)
n4
n4
(gn4g)
(gn4g)
a0
b0
g0
-n21
-g0
(n21g)
(n3)
-(n20c)
(n30c)
n20c
(gn4gn43g)
(n42g)
(c5)
c0
c0
-g0
-n1
(n1g)
-n3
(n2g)
(n32g)
x2
x2
(n41g)
(n40c)

TWO FSMS IN THE SAME BLOCK

William Bricken
May 2002

Two small FSMs (MCNC dk27 and s27) sharing the same input in the same block.

Not intended to illustrate dense packing of block resources.

Feedback from registers is illustrated. 6 registers are being used, feed-back to several tiers.

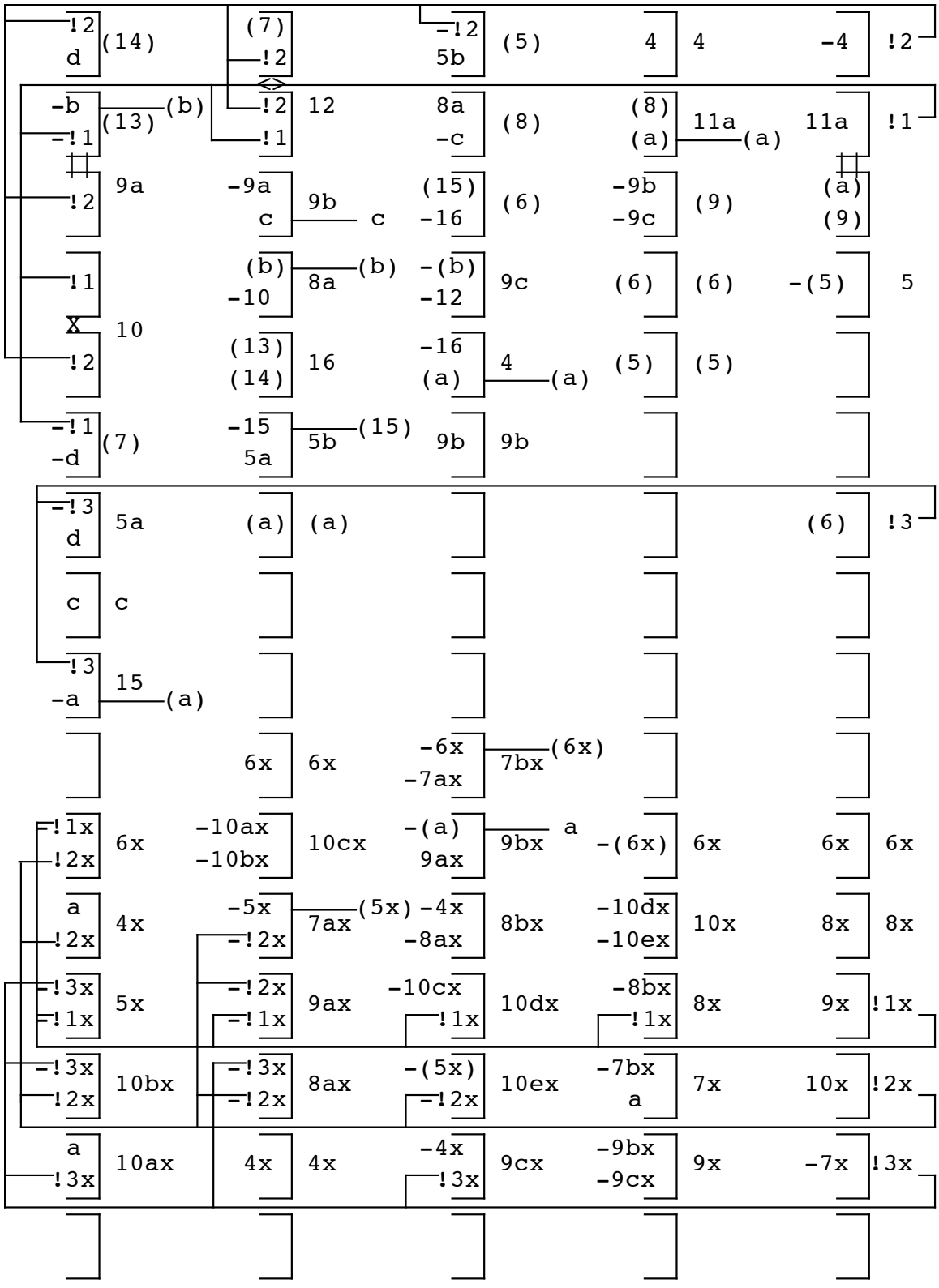
Line usage:

incoming	14*
tier 1-2	15
tier 2-3	16
tier 3-4	18
tier 4-5	11
out	3

Some inputs use to line to meet input mux constraint.
Several signals are passed through to lower tiers.
Registers are connected to a long line, not only to their bit slice.

Notation:

id	names are integers
!id	register
-id	use inverter
||	use neighbor sharing
X	xor using neighbor
<>	mux using neighbor
(id)	output is inverted wrt id-semantics


!2 d (14)
(7) !2
-!2 5b (5)
4 4
-4 !2
-b (13) (b)
-!1
!2 12
!1
8a -c (8)
(8) (a) 11a (a)
11a !1
9a
-9a c 9b c
(15) -16 (6)
-9b -9c (9)
(a) (9)
!1
(b) -10 8a (b)
-(b) -12 9c
(6) (6)
-(5) 5
10
!2
(13) (14) 16
-16 (a) 4 (a)
(5) (5)
-!1 -d (7)
-15 5a 5b (15)
9b 9b
-!3 d 5a
(a) (a)
(6) !3
c c
!3 -a 15 (a)
6x 6x
-6x -7ax 7bx (6x)
-!1x -!2x 6x
-10ax -10bx 10cx
-(a) 9ax 9bx a
-(6x) 6x
6x 6x
6x
a -!2x 4x
-5x 7ax (5x)
-!2x
-4x -8ax 8bx
-10dx -10ex 10x
8x 8x
-!3x -!1x 5x
-!2x -!1x 9ax
-10cx !1x 10dx
-8bx !1x 8x
9x !1x
-!3x -!2x 10bx
-!3x -!2x 8ax
-(5x) -!2x 10ex
-7bx a 7x
10x !2x
a -!3x 10ax
4x 4x
-4x !3x 9cx
-9bx -9cx 9x
-7x !3x