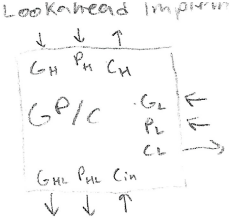
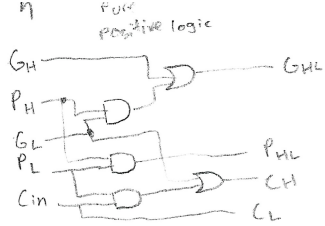
Carry-Lookahead Implementation



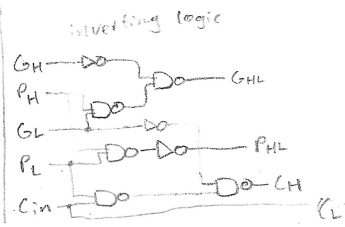
Pure positive logic



	tPD (ns)
GHL	0.27
PHL	0.12
CH	0.27
CL	0

size: 3·13 + 2·13 = 65 μ²

Inverting logic



	tPD (ns)
GHL	0.06
PHL	0.05
CH	0.06
CL	0

size: 5·10 + 3·10 = 80

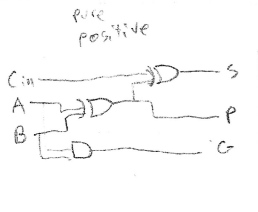
$G_{HL} = G_H \cdot P_H G_L = G_H \cdot (P_H G_L)$

$P_{HL} = P_H P_L = \overline{\overline{P_H P_L}} = \overline{\overline{P_H} + \overline{P_L}}$

$C_H = G_L + P_L C_{in} = \overline{\overline{G_L} \cdot (P_L C_{in})}$

← this is definitely the better approach

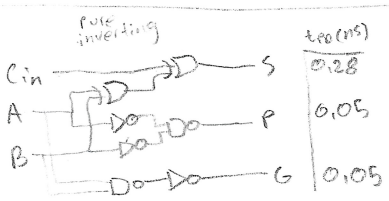
Pure positive



	tPD (ns)
S	0.28
P	0.14
G	0.12

size: 2·27 + 13 = 67

Pure inverting



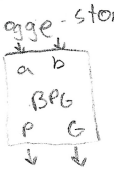
	tPD (ns)
S	0.28
P	0.05
G	0.05

size: 3·10 + 2·10 + 2·27 = 104

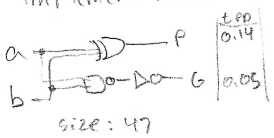
I did not know about the idea of alternating logic when I made the first version

← it might be worth the size cost to get P + G out faster

Kogge-stone implementation (alternative to carry-lookahead. Uses more size, but is faster)

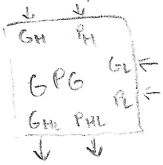


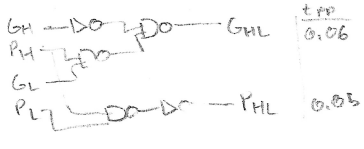
bit propagate & generate block



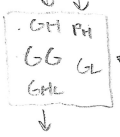
	tPD
P	0.14
G	0.05

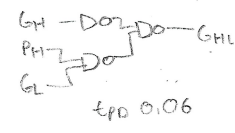
size: 47





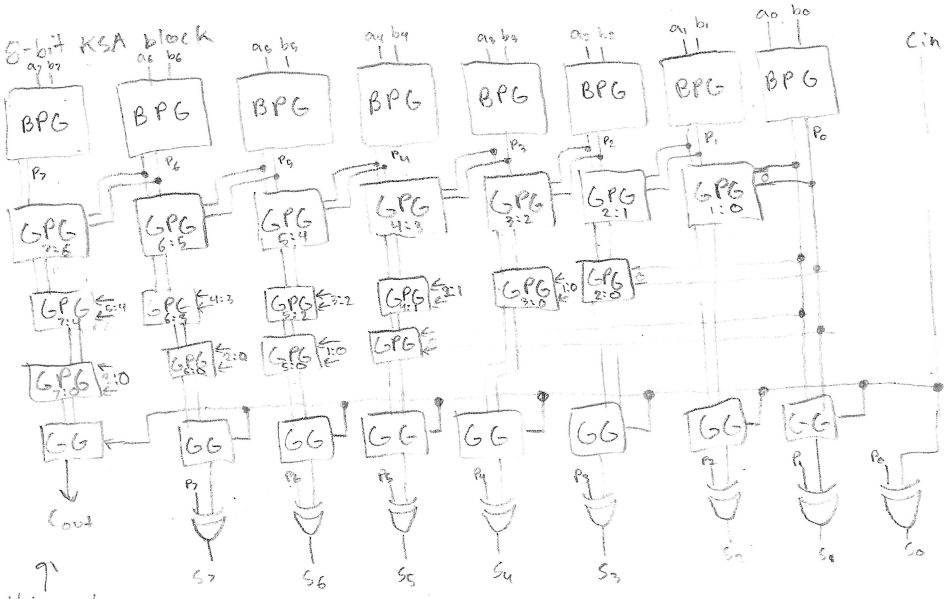
	tPD
GHL	0.06
PHL	0.05





tPD 0.06

8-bit KSA block



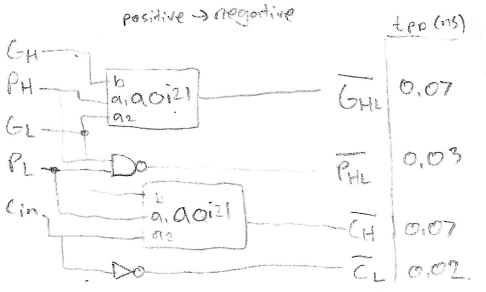
• Good idea to buffer $P_{0:0}$, $G_{0:0}$

this column is unnecessary for 32 bit since we do not check overflow

Carry-lookahead with alternating logic

GPC-inv PH

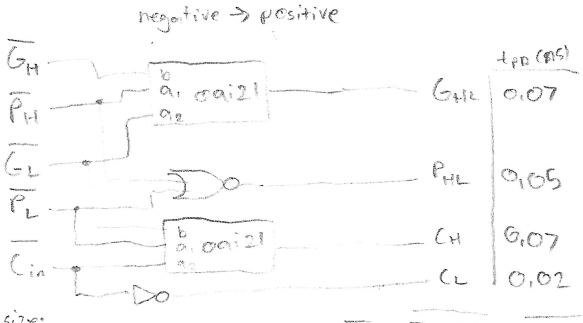
positive → negative



	tPD (ns)
$\overline{G_{HL}}$	0.07
$\overline{P_{HL}}$	0.03
$\overline{C_H}$	0.07
$\overline{C_L}$	0.02

size: 13·2 + 10 + 10 = 46 μ²

negative → positive



	tPD (ns)
GHL	0.07
PHL	0.05
CH	0.07
CL	0.02

size: 17·2 + 10 + 10 = 54 μ²

GPG-inv NP

oai21: $\overline{(A_1 + A_2) \cdot B}$

aoi21: $\overline{(A_1 \cdot A_2) + B}$

$\overline{G_{HL}} = \overline{G_H + P_H G_L}$

$\overline{P_{HL}} = \overline{P_H P_L}$

$\overline{C_H} = \overline{G_L + P_L C_{in}}$

$\overline{C_L} = \overline{C_{in}}$

$G_{HL} = \overline{\overline{G_H} + \overline{P_H} \cdot \overline{G_L}} = \overline{\overline{G_H} + \overline{P_H} + G_L} = \overline{\overline{G_H} \cdot (\overline{P_H} + G_L)}$

$P_{HL} = \overline{\overline{P_H} \cdot \overline{P_L}} = \overline{\overline{P_H} + \overline{P_L}}$

$C_H = \overline{\overline{G_L} \cdot (\overline{P_L} + \overline{C_{in}})}$

$C_L = \overline{\overline{C_{in}}}$

At an average of 62.5% c an only barely slower, the alternating logic makes sen The lowest layer may nee to be perform positive→positiv for 2^n n odd bit addition.