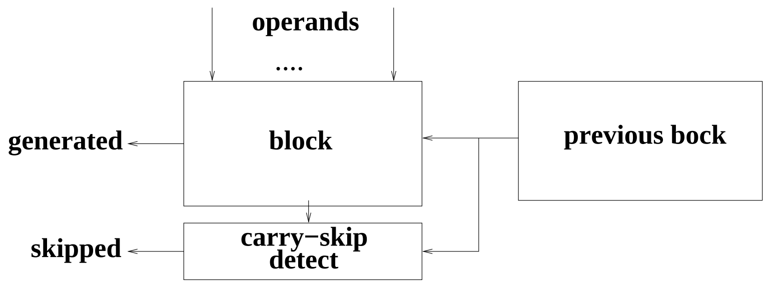
Quelle:Timo Hämäläinen, Computer Arithmetic: Carry-Skip Adder

Carry-Skip-Adder

- consider addition of the following numbers

  $\ldots a_{k+2}\ a_{k+1}\ a_k$ 010101 $a_{l+2}\ a_{l+1}\ a_l\ \ldots$
  $\ldots b_{k+2}\ b_{k+1}\ b_k$ 101010 $b_{l+2}\ b_{l+1}\ b_l\ \ldots$

- if c_{l+3} = 1 $\rightarrow$ carry will propagate to position k
- to speed-up operation, propagation is skipped to position i without waiting for rippling
- operation time varies according to operands as in carry-complete addition
- to implement carry-skip adder, stages are divided into blocks

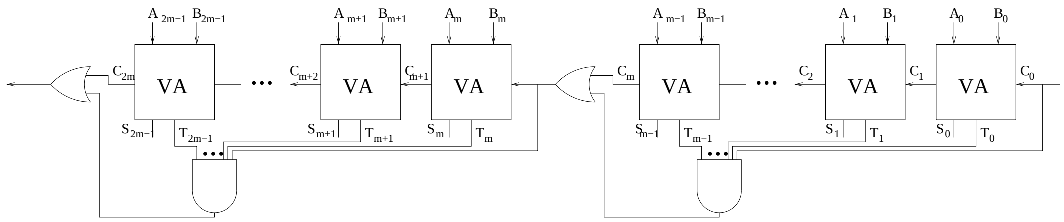


- carry-skip logic is added to each block to detect when carry-in the block can be passed directly to the next block
- define carry transfer $T_i = a_i + b_i$
- carry skipping can be detected for a block size of m as follows (carry propagates through all stages):

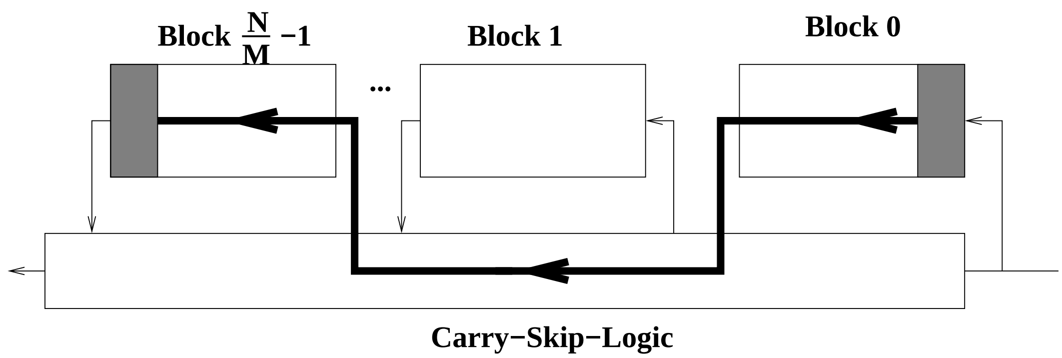
  $T_j \cdot T_{j+1} \ldots T_{j+m-1} = 1 \qquad (= (a_j + b_j) \cdot (a_{j+1} + b_{j+1}) \ldots)$

- note: this takes into account both propagated and generated carries!
- carry out from the block (m-bits in a block) is

$$\underbrace{T_j \cdot T_{j+1} \ldots T_{j+m-1} \cdot c_j}_{skipped} + \underbrace{c_{j+m}}_{generated}$$



- block size in carry-skip adder is very important
- worst case operation time takes place when
  - carry is generated in the first block
  - carry skips intermediate stages
  - carry is killed in the last block



;

- worst case addition time is $\left(\frac{2n}{m} + 4m - 4\right)\tau$ (n=adder width, m=block size)
- for optimal block size, minimize delay:
  $\frac{d}{dm}\left(\frac{2n}{m} + 4m - 4\right) = -2\left(\frac{n}{m^2} - 2\right) \equiv$
  $\Rightarrow m = \sqrt{\frac{n}{2}}$
- in practise, non-uniform block sizes gives the best performance
- in general, outer blocks should be smaller than middle blocks