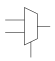
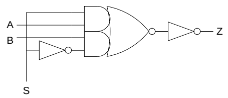
Alternative Cell Design Strategy

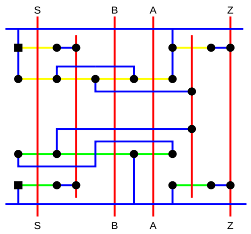
Gate Matrix Style

The circuit is created as a matrix of intersecting transistor diffusion rows and polysilicon columns.

A simple example is the two input multiplexor we saw earlier. We align transistors on their common gate connections (as for *line of diffusion* designs), but here we allow multiple transistors to use the same polysilicon column.

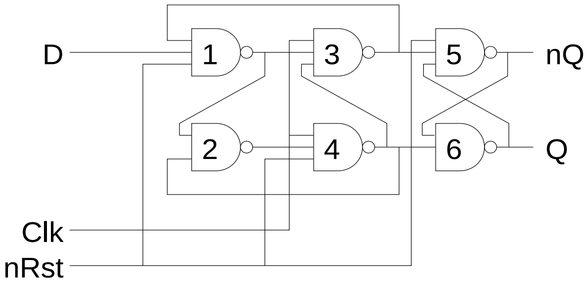




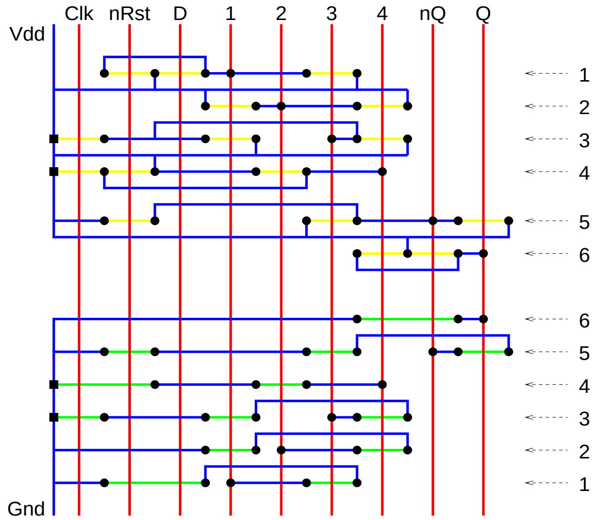


Gate Matrix Style

The layout of a standard six NAND gate d-type should illustrate the style more completely.



Here each NAND gate is allocated a different pair of rows in the layout (further optimization of this circuit is possible).



Cell Design Choices

- Line of Diffusion – Euler Path

  The line of diffusion approach to cell design, backed up by investigation of Euler paths, leads to efficient layout of small cells.

- Gate Matrix[1]

  Where cells are more complex and in particular when there are multiple transistors sharing a common gate connection, gate matrix design will often give more efficient use of area.

  – Complex Standard Cells

  - - Mux, D-Type, Full Adder etc

    - consider impact of tall gate matrix cells on space efficiency in simple cells.

  – Full Custom Cells

  - - Efficient sub-circuit design.

    e.g. Full custom ALU bitslice.

[1]note that partial Euler paths play an important part in efficient Gate Matrix design.

Bit Slicing

Repetitive Logic

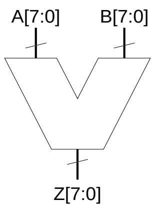
Where logic blocks are duplicated within a system, there is much to gain from optimization.

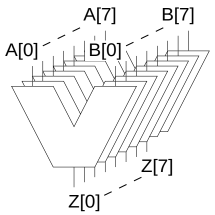
- Optimize single block for size.
  The effect is amplified by the number of identical blocks.

- Optimize interconnect.
  With careful design, the requirement for routing channels between the blocks can be eliminated.
  Interconnection is by butting in 2 dimensions.

Bit Slicing

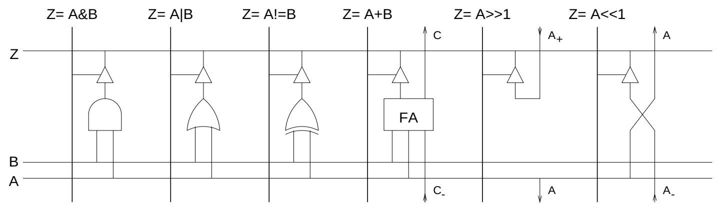
Instead of creating an ALU function by function, we create it slice by slice.





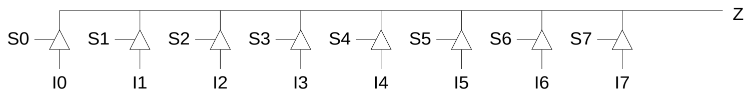
- Each *bit slice* is a full 1-bit ALU.
- *N* are used to create an *N*–bit ALU.

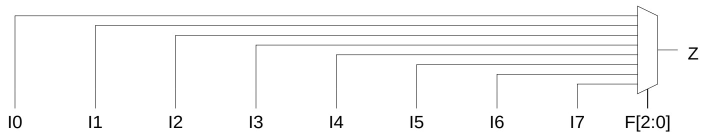
Bit Slicing



- Simple 1–bit ALU.
- The control lines select which function of A and B is fed to Z.
- Some functions, e.g. Add, require extra data I/O.

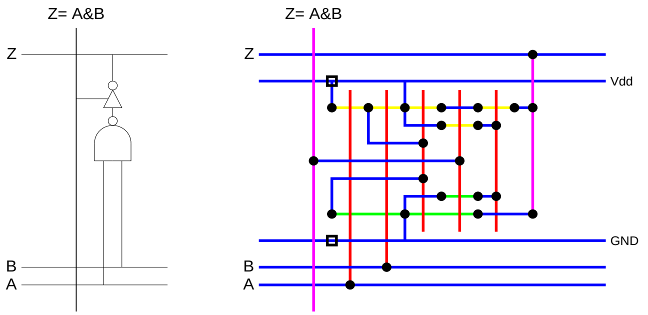
Distributed Multiplexing





- The tri-state buffers act as a multiplexor.
- This *distributed multiplexing* reduces data wiring at the expense of increased control wiring, potentially saving space in a multi-bit bitslice system.
- For on chip buses we must ensure that in each clock cycle the bus is driven by exactly one source.

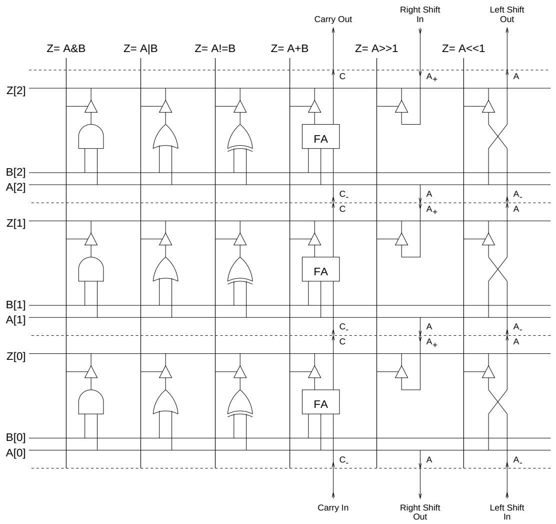
Bit Slicing



- Data busses horizontal, control lines vertical.
- Compact gate matrix implementation[2].

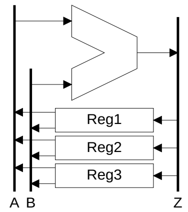
[2]bit slice designs can also be built around standard cells although the full custom approach used here gives better results

- Bit Sliced ALU (3-bits, scalable).

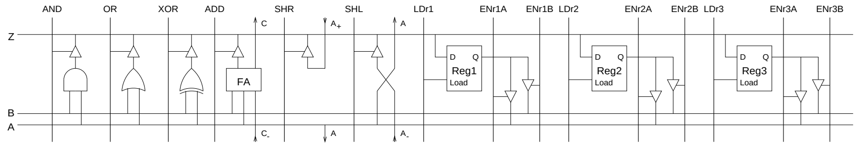


Bit Slicing

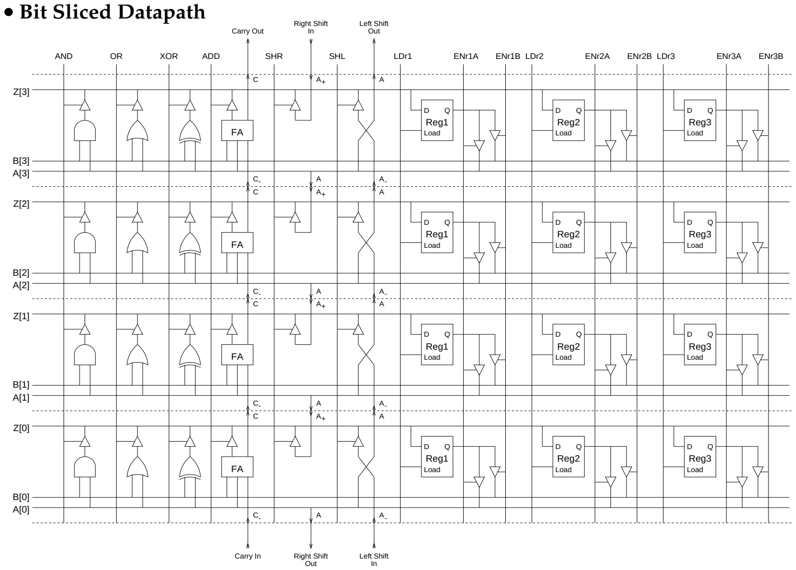
We can extend this principle to the whole *datapath*.



- 1–bit datapath.

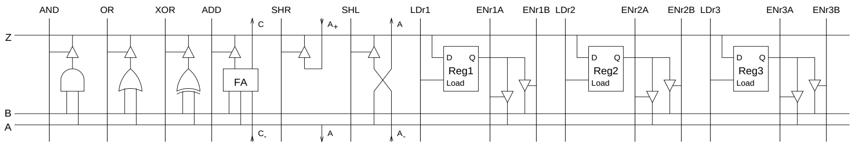


- **Bit Sliced Datapath**

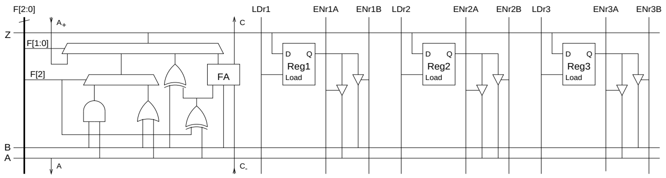


Bit Slicing

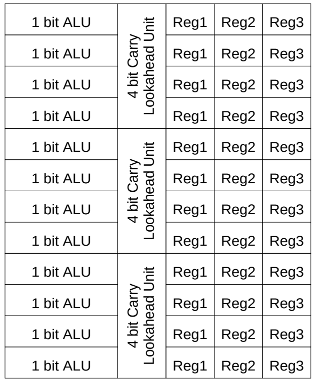
- Distributed Multiplexing

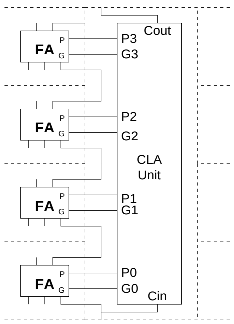


- Local Multiplexing



Bit Slicing





- Bitslice Exceptions

  Where full bitslicing is not suitable we attempt to disrupt the bitslice as little as possible.