Programmable Logic Devices

Dr. Shubhajit Roy Chowdhury,

Centre for VLSI and Embedded Systems Technology,
IIIT Hyderabad, India

Email: src.vlsi@iiit.ac.in

PLD Definition

❑ Field Programmable Device (FPD):

— a general term that refers to any type of integrated circuit used for implementing digital hardware, where the chip can be configured by the end user to realize different designs. Programming of such a device often involves placing the chip into a special programming unit, but some chips can also be configured “in-system”. Another name for FPDs is *programmable logic devices* (PLDs).

Source: S. Brown and J. Rose, FPGA and CPLD Architectures: A Tutorial, IEEE Design and Test of Computer, 1996

Why Programmable Logic?

- Facts:
  - It is most economical to produce an IC in large volumes
  - Many designs required only small volumes of ICs
- Need an IC that can be:
  - Produced in large volumes
  - Handle many designs required in small volumes
- A programmable logic part can be:
  - made in large volumes
  - programmed to implement large numbers of different low-volume designs

Programmable Logic
- Additional Advantages

- Many programmable logic devices are *field- programmable*, i. e., can be programmed outside of the manufacturing environment
- Most programmable logic devices are *erasable* and *reprogrammable*.
  - Allows "updating" a device or correction of errors
  - Allows reuse the device for a different design - the ultimate in re-usability!
  - Ideal for course laboratories
- Programmable logic devices can be used to prototype design that will be implemented for sale in regular ICs.
  - Complete Intel Pentium designs were actually prototype with specialized systems based on large numbers of VLSI programmable devices!

Programming Technologies

- Programming technologies are used to:
  - Control connections
  - Build lookup tables
  - Control transistor switching
- The technologies
  - Control connections
    - Mask programming
    - Fuse
    - Antifuse
    - Single-bit storage element

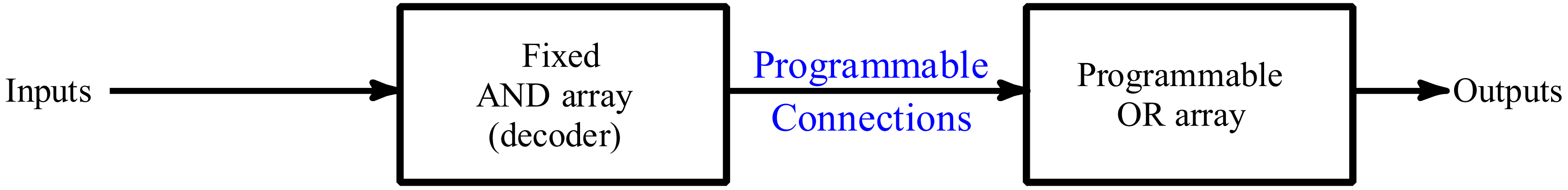
Programming Technologies (Cont’d)

- The technologies (continued)
  - Build lookup tables
    - Storage elements (as in a memory)
  - Transistor Switching Control
    - Stored charge on a floating transistor gate
      - Erasable
      - Electrically erasable
      - Flash (as in Flash Memory)
    - Storage elements (as in a memory)

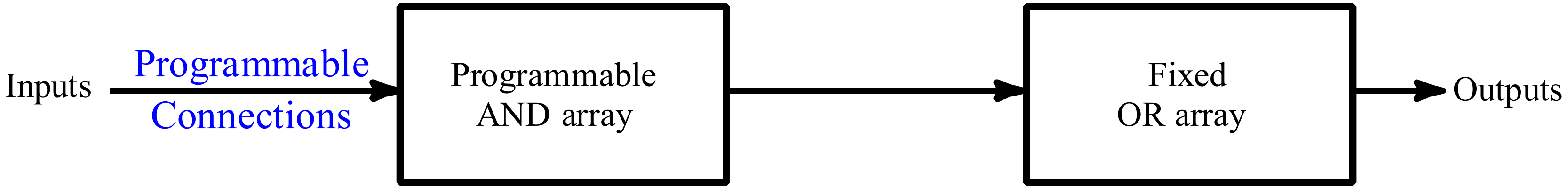
Programmable Configurations

- *Read Only Memory* (*ROM*) - a fixed array of AND gates and a programmable array of OR gates
- *Programmable Array Logic* (*PAL*)® - a programmable array of AND gates feeding a fixed array of OR gates.
- *Programmable Logic Array* (*PLA*) - a programmable array of AND gates feeding a programmable array of OR gates.
- *Complex Programmable Logic Device* (*CPLD*) /*Field-Programmable Gate Array* (*FPGA*) - complex enough to be called “architectures” - See VLSI Programmable Logic Devices reading supplement

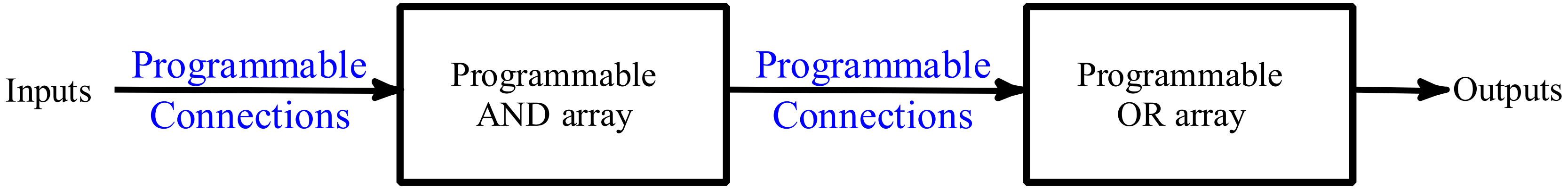
ROM, PAL and PLA Configurations



(a) Programmable read-only memory (PROM)



(b) Programmable array logic (PAL) device

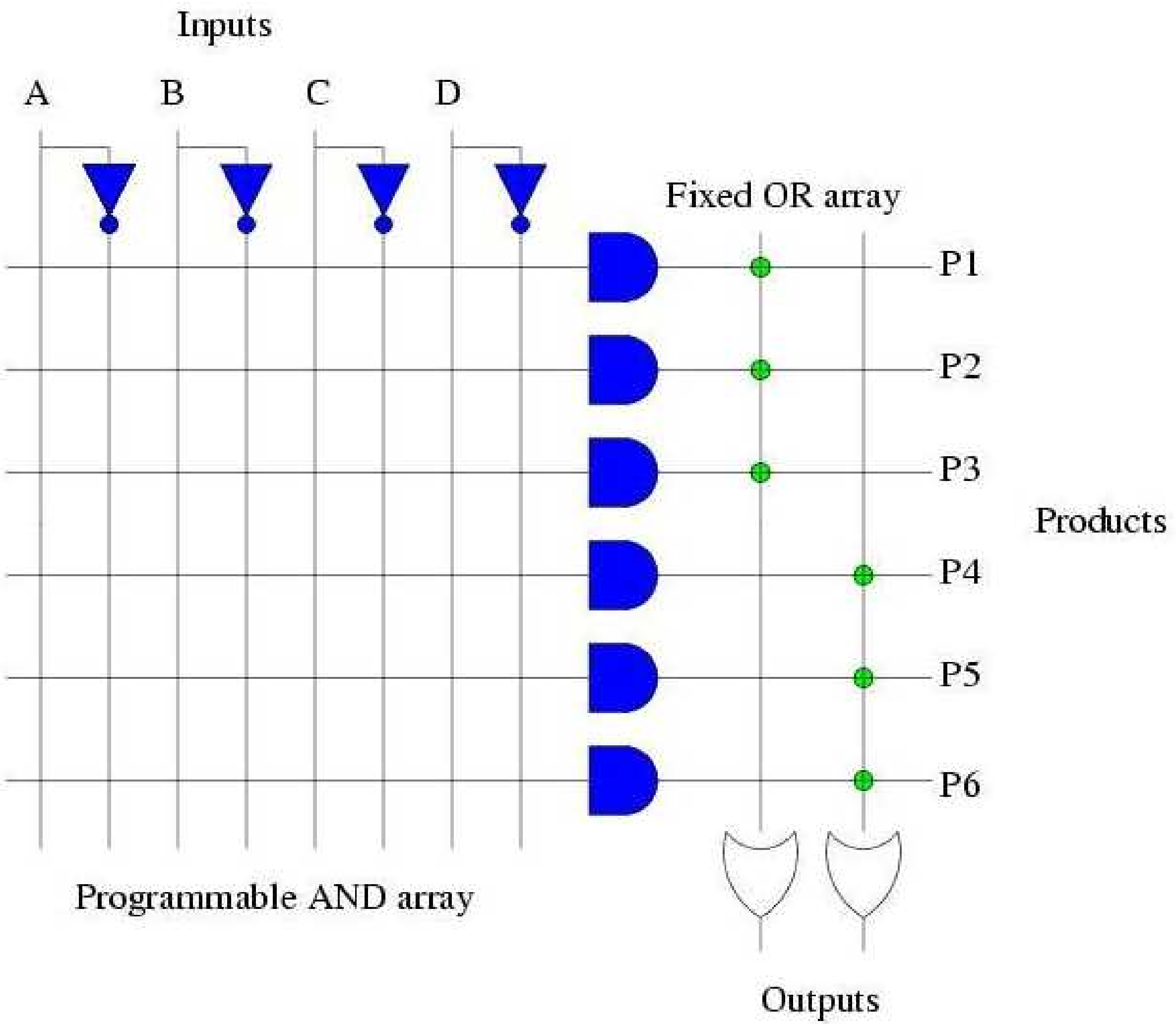


(c) Programmable logic array (PLA) device

Programmable Array Logic (PAL)

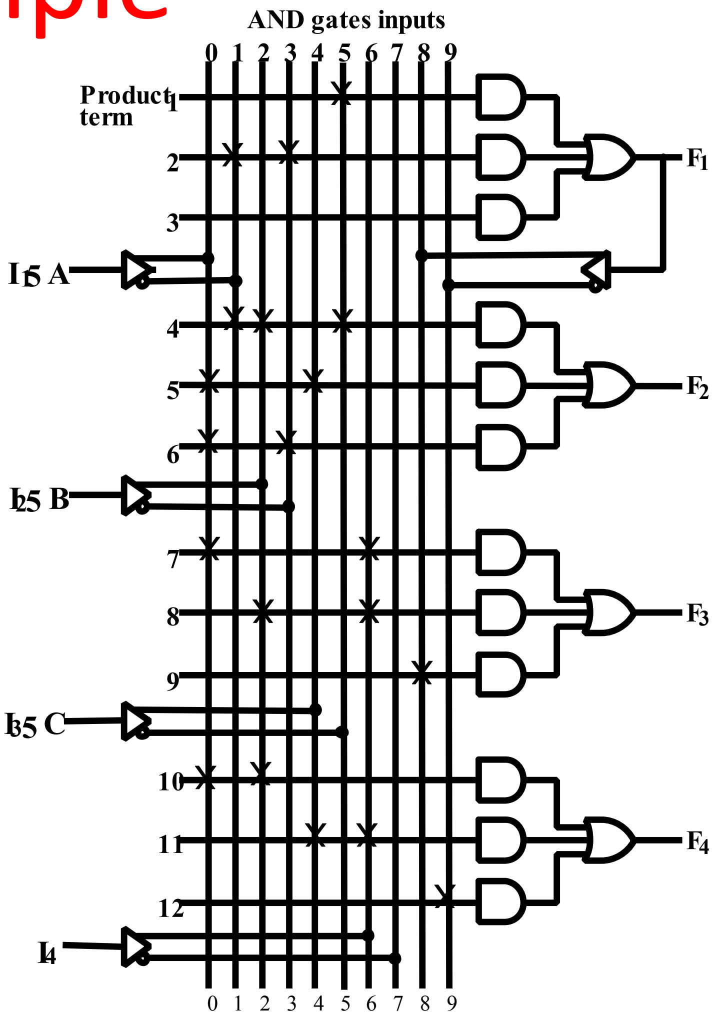
- The PAL is the opposite of the ROM, having a programmable set of ANDs combined with fixed ORs.
- Disadvantage
  - ROM guaranteed to implement any M functions of N inputs. PAL may have too few inputs to the OR gates.
- Advantages
  - For given internal complexity, a PAL can have larger N and M
  - Some PALs have outputs that can be complemented, adding POS functions
  - No multilevel circuit implementations in ROM (without external connections from output to input). PAL has outputs from OR terms as internal inputs to all AND terms, making implementation of multi-level circuits easier.

PAL Logic Diagram



Programmable Array Logic Example

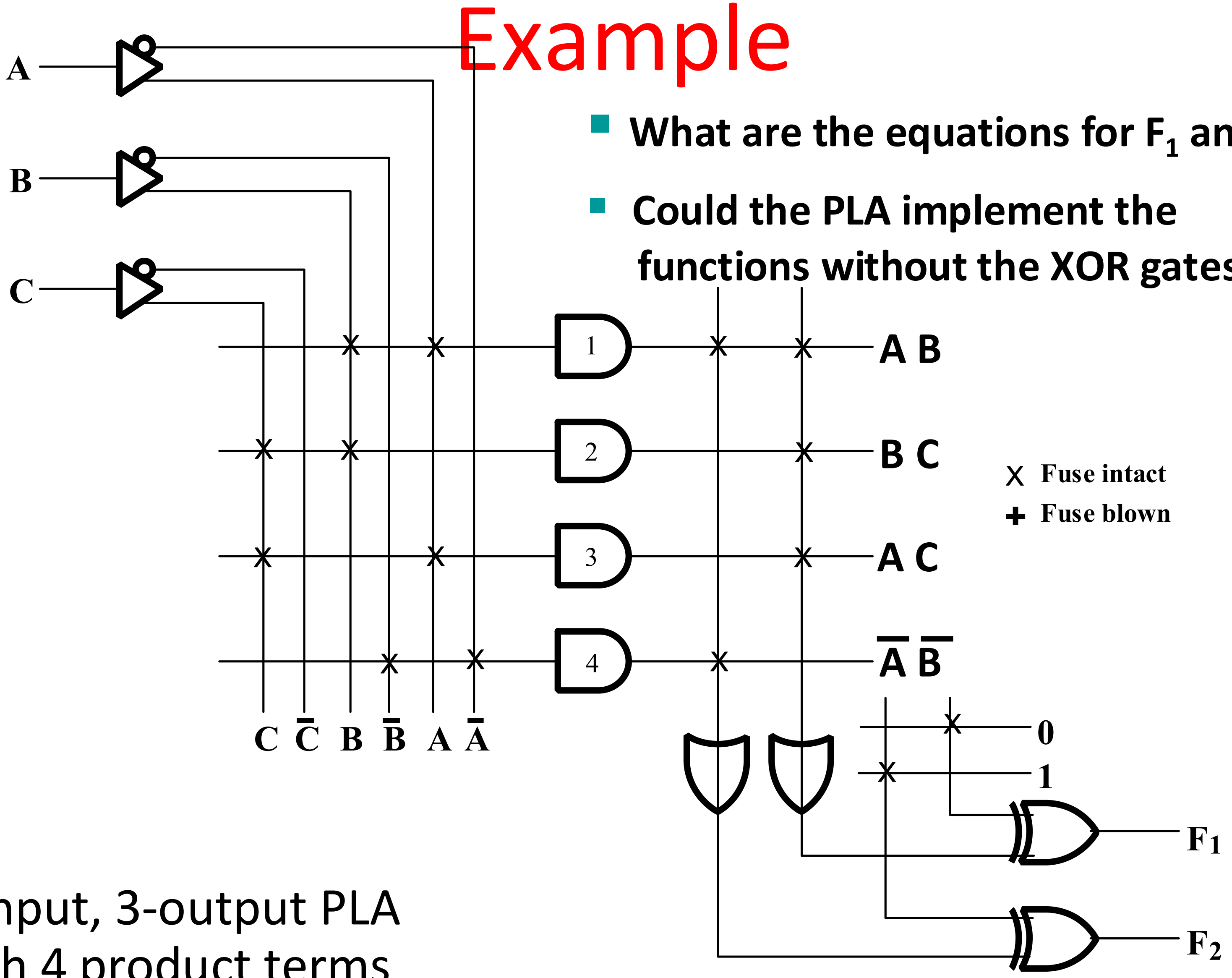
- 4-input, 3-output PAL with fixed, 3-input OR terms
- What are the equations for F1 through F4?



Programmable Logic Array (PLA)

- Compared to a ROM and a PAL, a PLA is the most flexible having a programmable set of ANDs combined with a programmable set of ORs.
- Advantages
  - A PLA can have large N and M permitting implementation of equations that are impractical for a ROM (because of the number of inputs, N, required
  - A PLA has all of its product terms connectable to all outputs, overcoming the problem of the limited inputs to the PAL Ors
  - Some PLAs have outputs that can be complemented, adding POS functions
- Disadvantage
  - Often, the product term count limits the application of a PLA. Two-level multiple-output optimization reduces the number of product terms in an implementation, helping to fit it into a PLA.

Programmable Logic Array Example



- What are the equations for F_1 and F_2?
- Could the PLA implement the functions without the XOR gates?

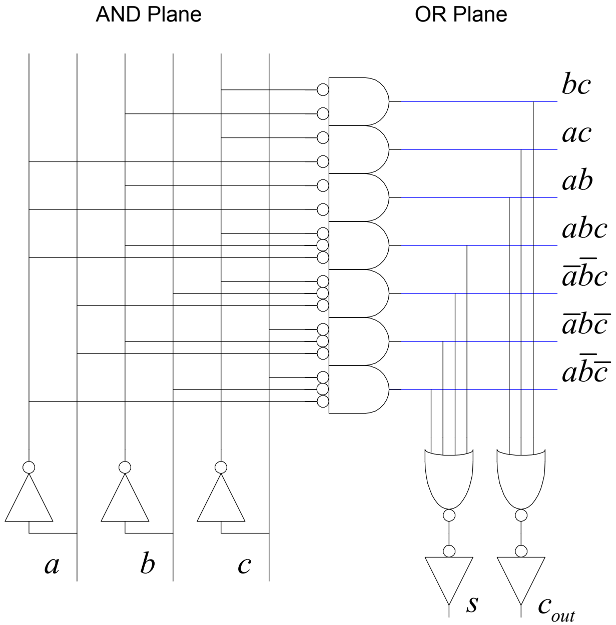
- 3-input, 3-output PLA with 4 product terms

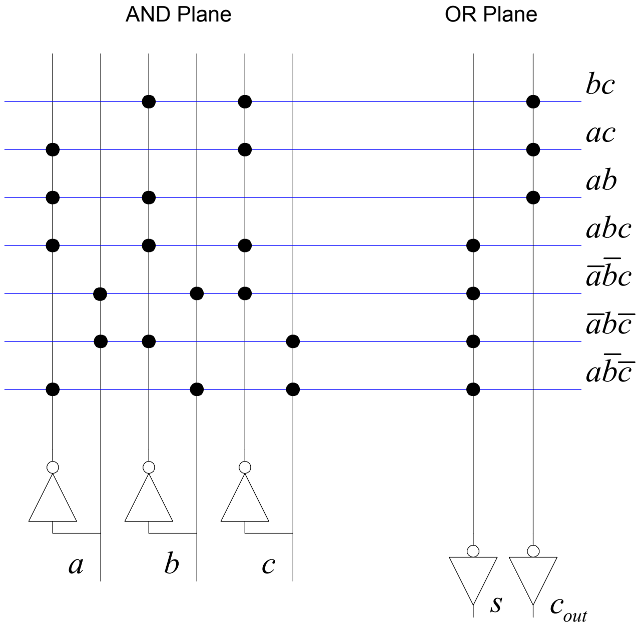
Logic Implementation with PLA

- Finite number of AND gates => simplify function to minimum number of product terms
- Number of literals in a product term is not important since we have all the input variables
- Sharing of product terms between outputs => multiple-output minimization

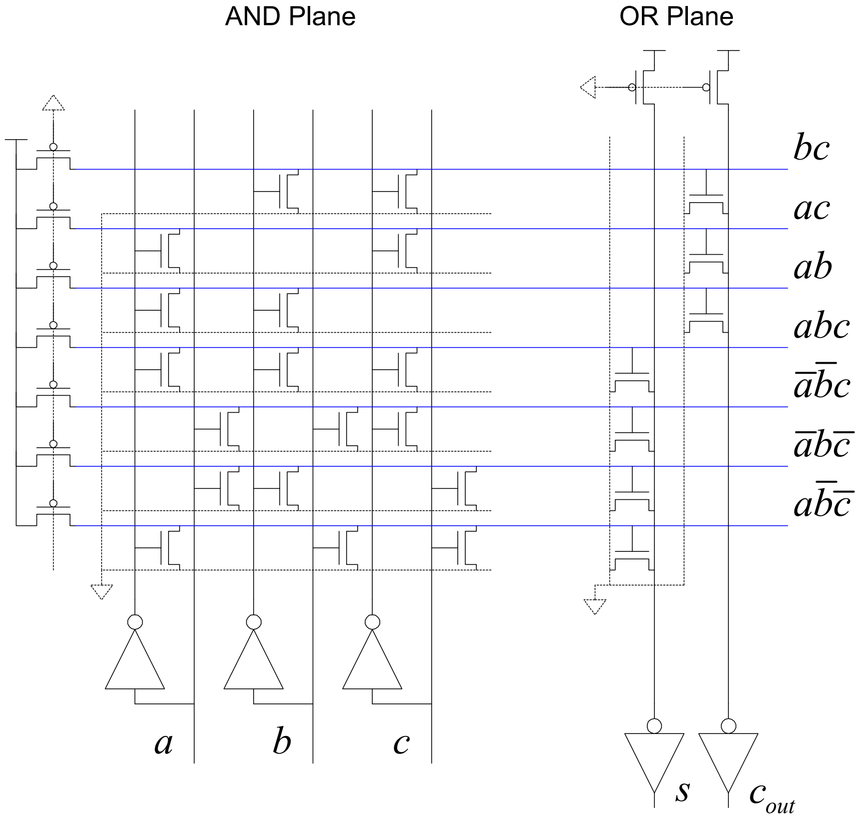
NOR-NOR PLAs

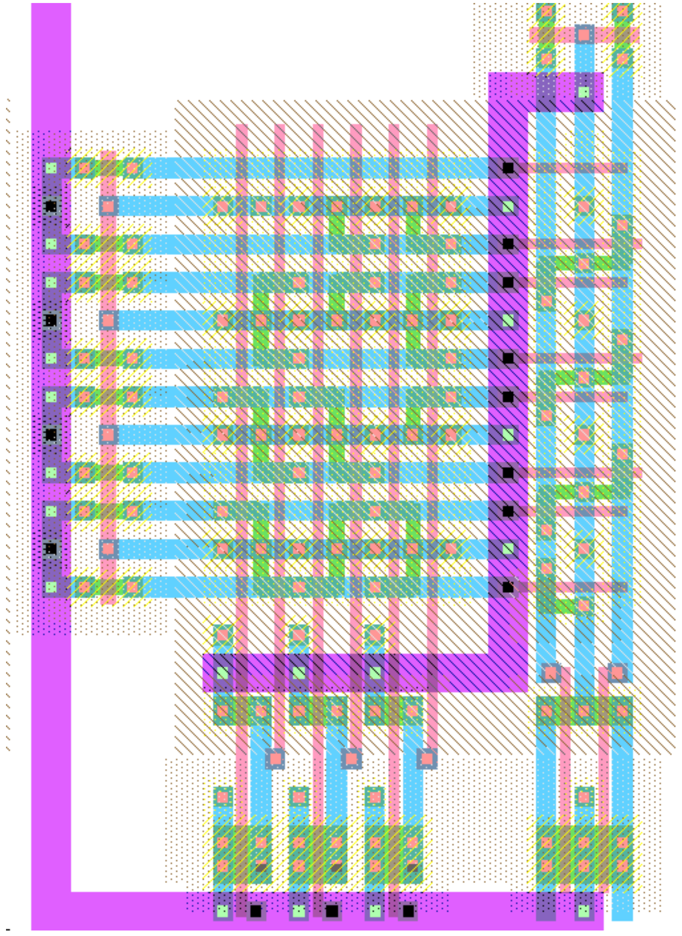
- ANDs and ORs are not very efficient in CMOS
- Dynamic or Pseudo-nMOS NORs are very efficient
- Use DeMorgan’s Law to convert to all NORs



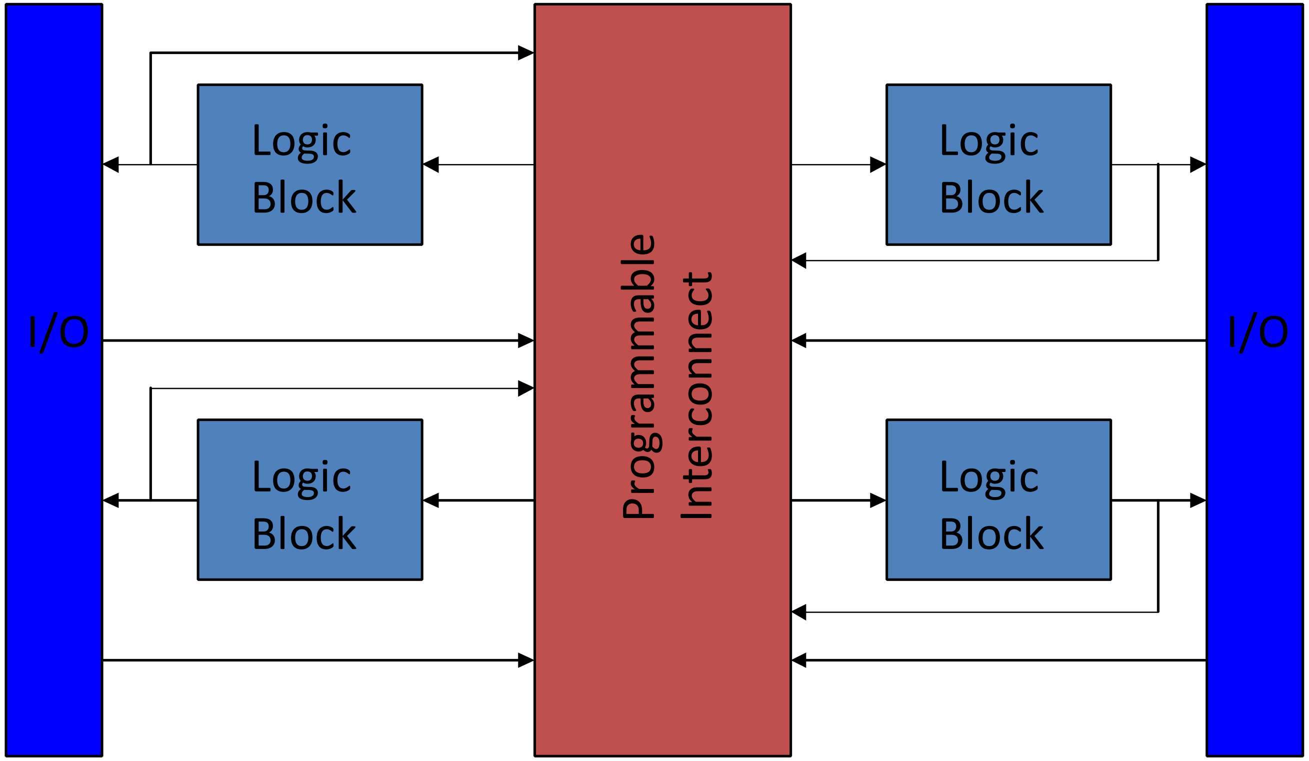


PLA Schematic & Layout





CPLD

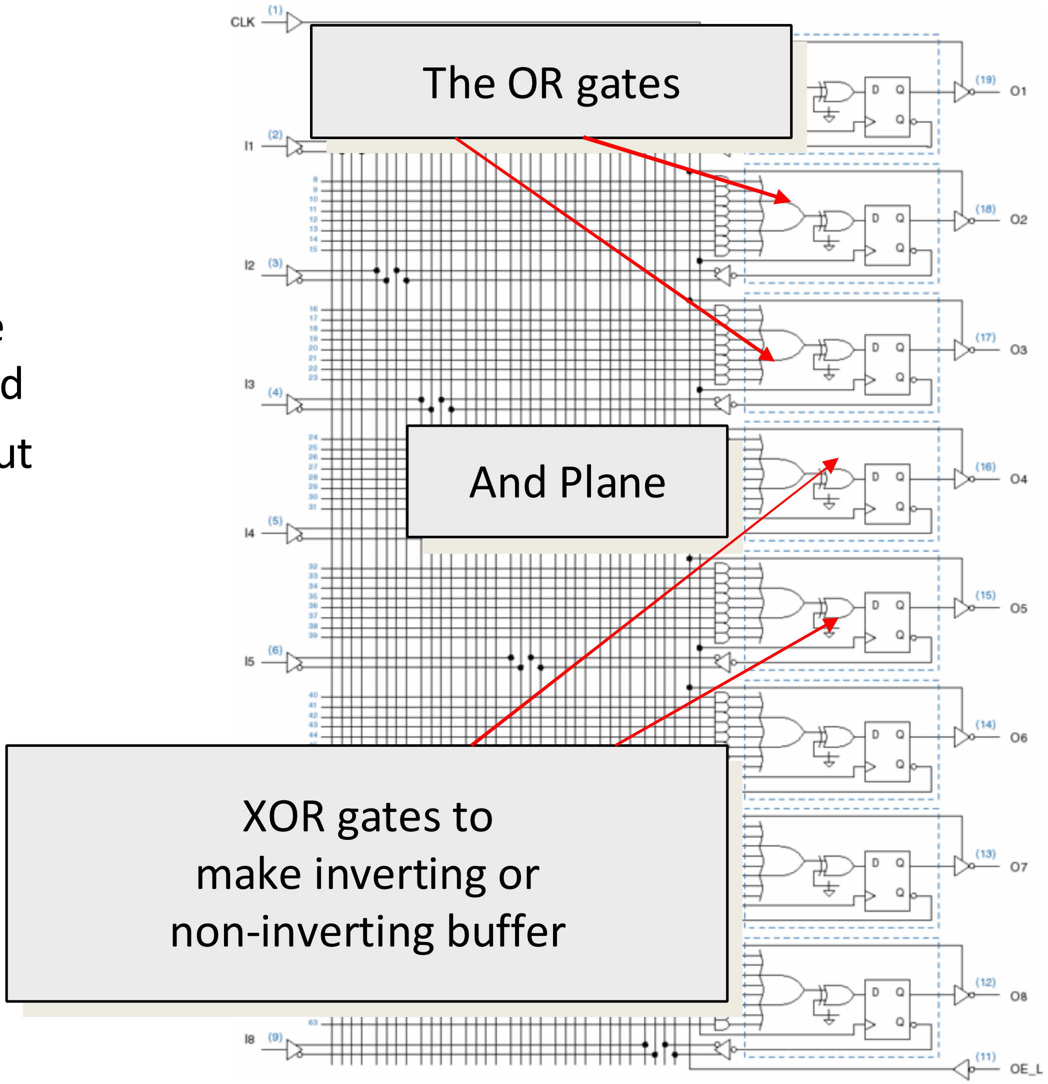

I/O
Logic Block
Logic Block
Programmable Interconnect
Logic Block
Logic Block
I/O

CPLD Logic Block

- Simple PLD
  - Inputs
  - Product-term array
  - Product term allocation function
  - Macro-cells (registers)
- Logic blocks executes sum-of-product expressions and stores the results in micro-cell registers
- Programmable interconnects route signals to and from logic blocks

Typical CPLD Architecture

- Each output is programmable as combinational or registered
- Also has programmable output polarity

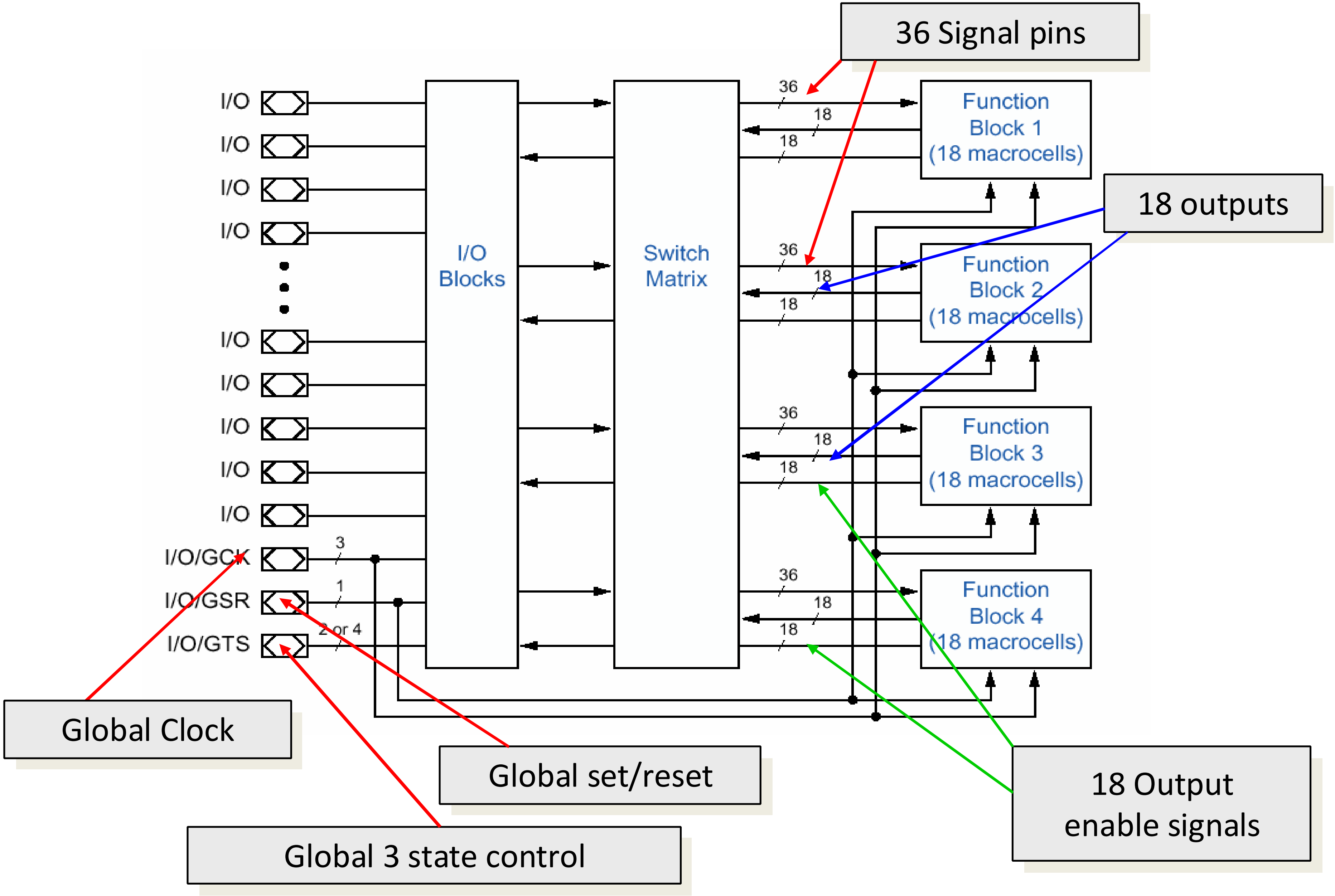


The OR gates

And Plane

XOR gates to make inverting or non-inverting buffer

Architecture of Xilinx 9500-family CPLD


I/O
I/O
I/O
I/O
I/O
I/O
I/O
I/O
I/O
I/O/GCK
I/O/GSR
I/O/GTS
3
1
2 or 4
I/O Blocks
Switch Matrix
36
18
18
Function Block 1 (18 macrocells)
Function Block 2 (18 macrocells)
Function Block 3 (18 macrocells)
Function Block 4 (18 macrocells)

36 Signal pins

18 outputs

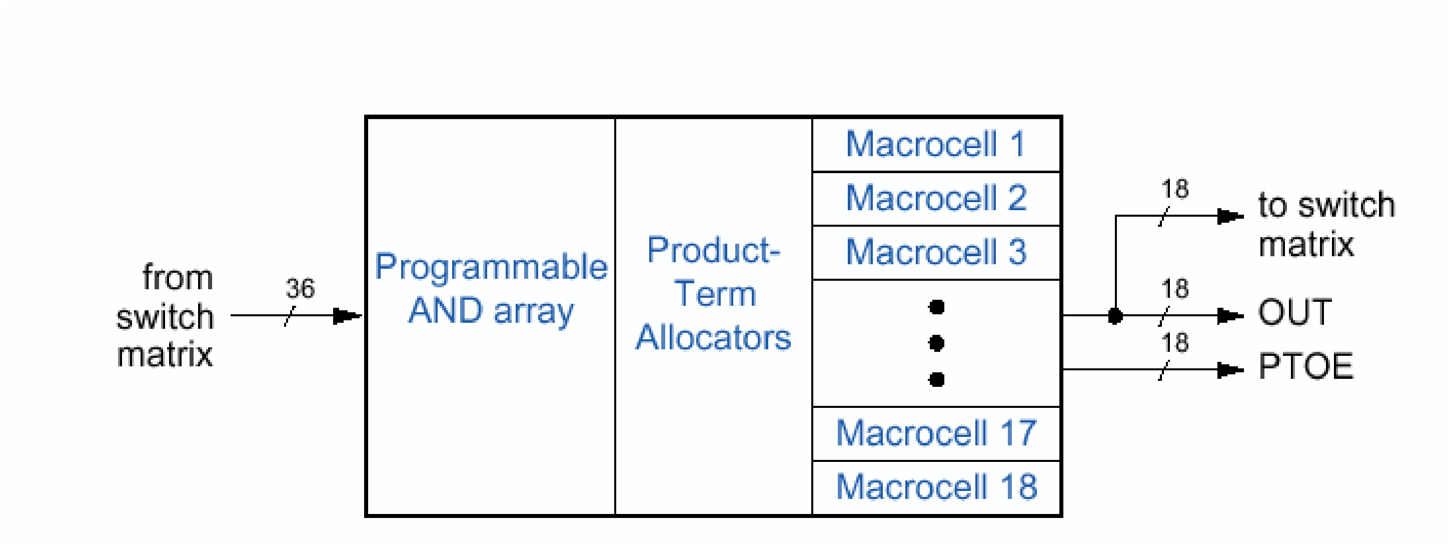
Global Clock

Global set/reset

Global 3 state control

18 Output enable signals

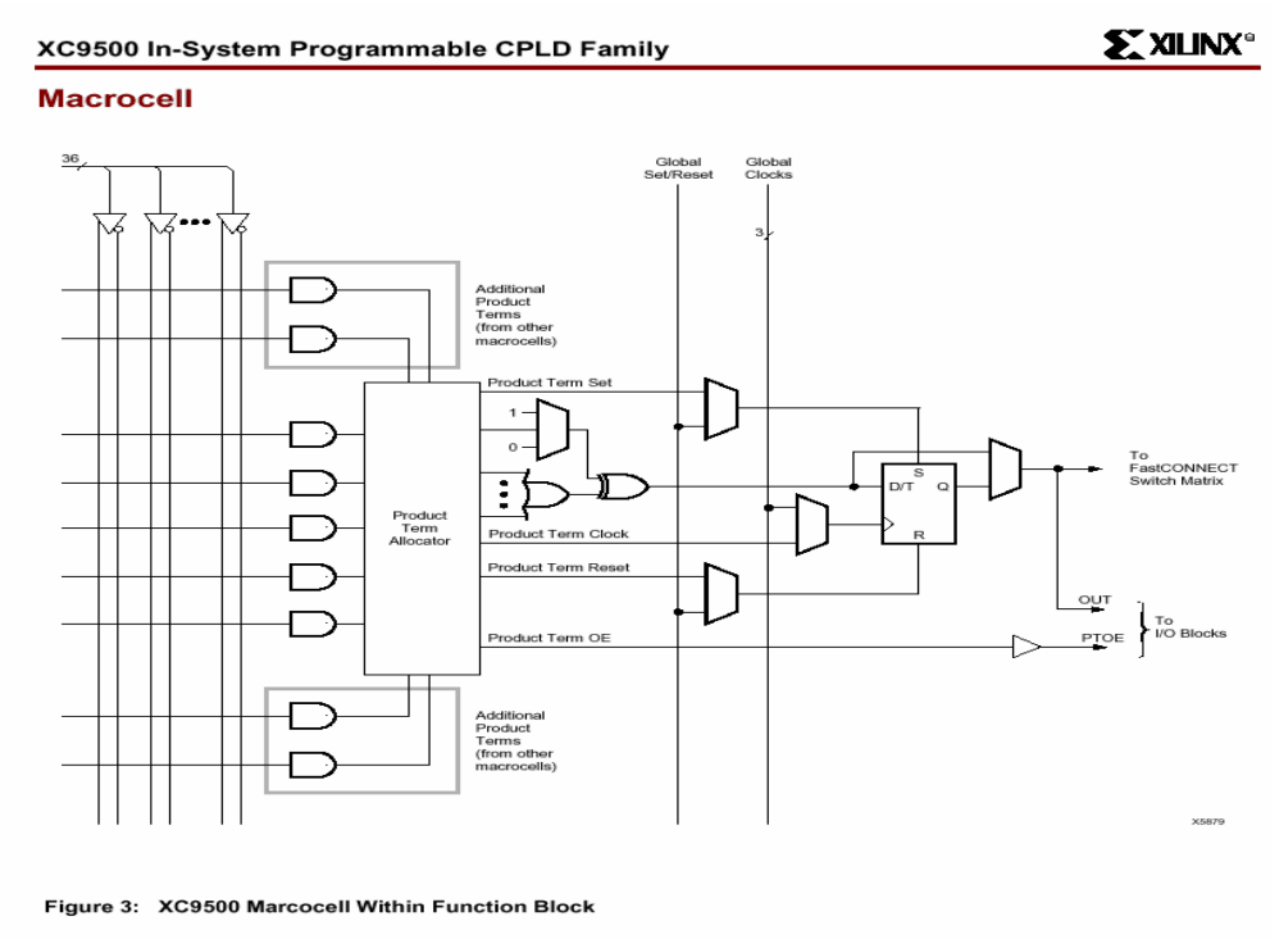
Architecture of Xilinx FB



Most CLPDs have fewer AND terms per macrocell
XC9500 has 5 whereas 16V8 has 8 and 22V10 has 8-16

But…each macrocell can use unused ANDs froms its neighboring macrocells using the *"product-term-allocators"*

Macrocell of Xilinx Family CPLD



XC9500 In-System Programmable CPLD Family

XILINX

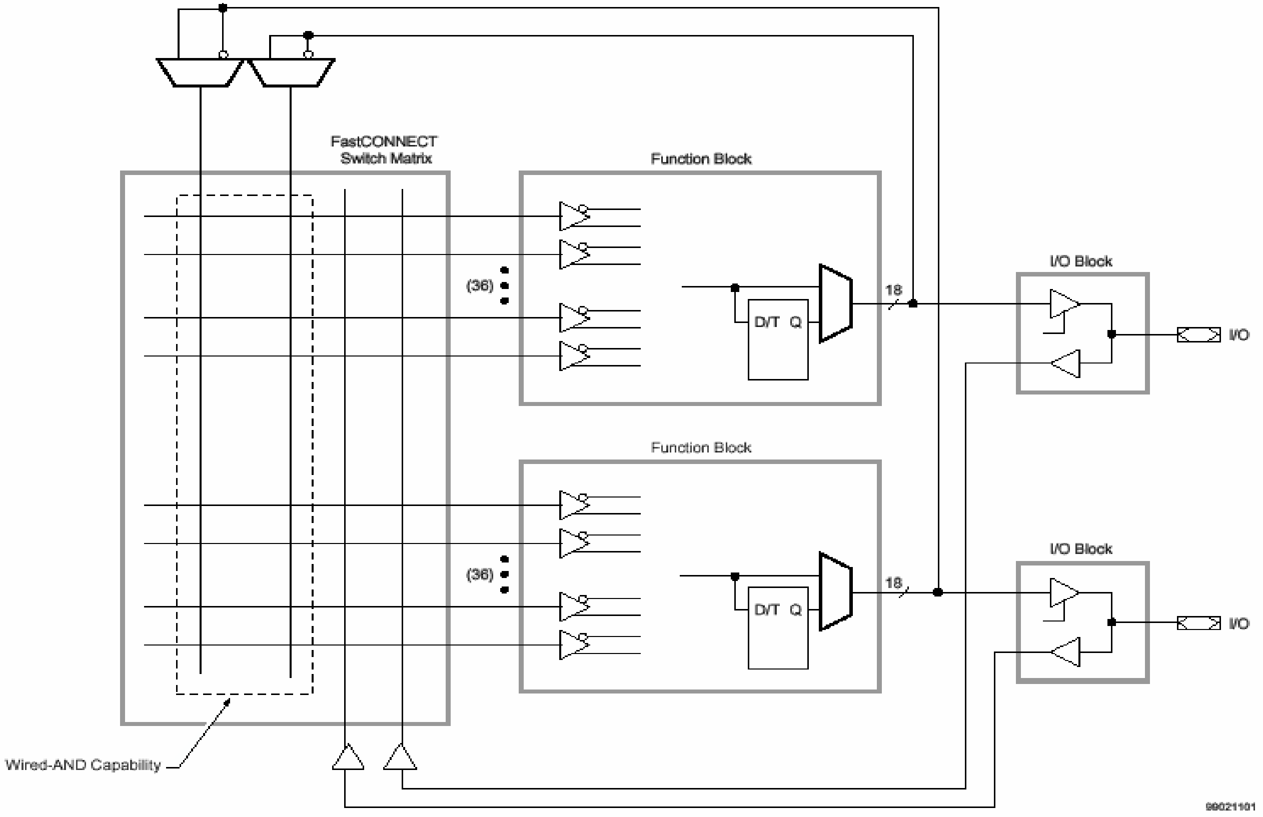
Macrocell

Figure 3: XC9500 Marcocell Within Function Block

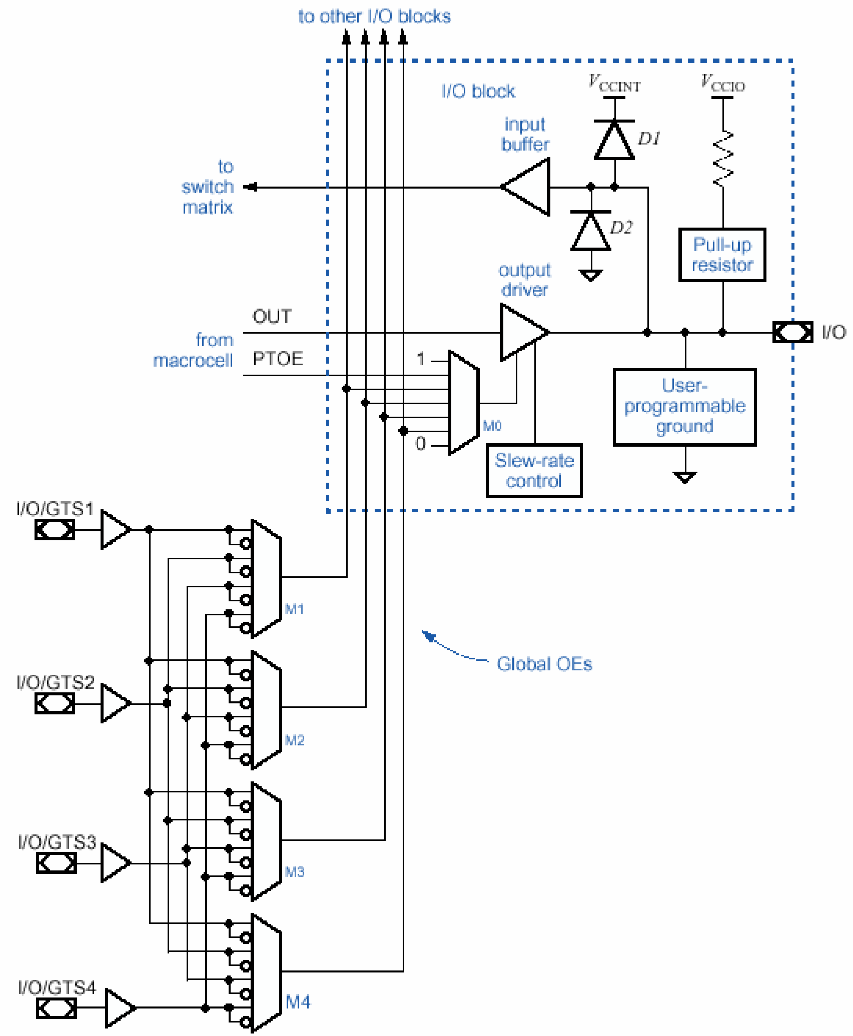
Switch Matrix

XILINX® XC9500 In-System Programmable CPLD Family

FastCONNECT Switch Matrix



Xilinx I/O Block



FPGA *v.s.* CPLD

- Capacitance

	SPLDs	CPLDs	FPGAs
Equivalent gates	0 ~ 200	200 ~ 12,000	1000 ~ 1,000,000

- Applications

CPLDs	FPGAs
1. Implement random glue logics or Replace circuits previously implemented by multiple SPLDs 2. Circuits that can exploit wide AND/OR gates, and do not need a very large number of flip-flops are good candidates for implementation in CPLDs.	1. FPGAs can be used in various applications: prototyping, FPGA-based computers, on-site hardware re-configuration, DSP, logic emulation, network components, etc.

?