Embedded Systems

Hardware Technologies for ICs

Prof. William Fornaciari

william.fornaciari@polimi.it

Outline

- Hardware Technologies for ICs
    - (AS)IC
        - Full custom
        - Standard cell
        - Gate array
    - Programmable logic
        - From ROMs to CPLDs
        - FPGAs

Hardware Technologies for ICs

Hardware

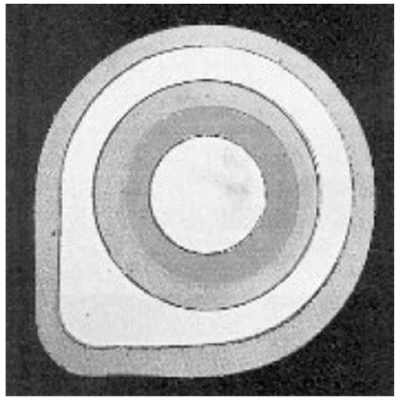
- (‘50): planar technology
  - Integration of Circuits, simultaneous design of replicas
- Possible flavors
  - COTS
    - Components, frequently integrated, tailored for specific function and available (ready) on the market. Not to be designed
  - ASIC
    - Circuits designed ad-hoc for a specific application for superior performance or for high volume
  - Programmable logic
    - ICs making available to the designer a number of logic functions, memory and programmable connections
  - Microprocessors, microcontrollers
    - Useful when the functions to be implemented are complex or not available as COTS

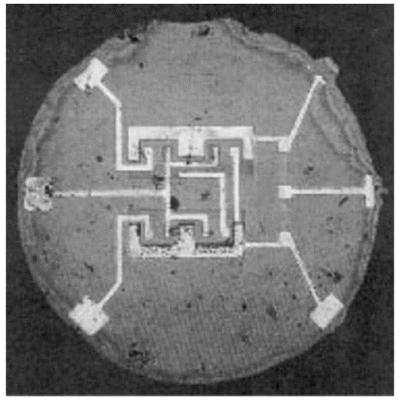
(AS) ICs

Application Specific Integrated Circuits

(AS)IC – a pill of history

- 1958 Jean Hoerni (and others) invented the planar process
  - In 1957 they founded Fairchild Semiconductor, today ST Microelectronics (Italy)
    - Others: Gordon Moore and Robert Noyce, in 1968 created Intel Corporation, Victor Grinich, Eugene Kleiner, Sheldon Roberts and Jay Last co-founders of the so called «Silicon Valley»
- Planar technology is at the beginning of the *digital era*
- First *transistor* and first *integrated circuit (*pictures)





(AS)IC

- Basics on the planar process
  - Use of a semiconductor (Ge, Si, GaAs) whose electro-magnetic characteristics of some regions can be modified in a controlled way
  - All the main components (transistors, diodes, resistors, capacitances) can be realized using Silicon after a proper processing
  - The connections among the components are made using metal (aluminum in the past, now copper) or polysilicon

(AS)IC

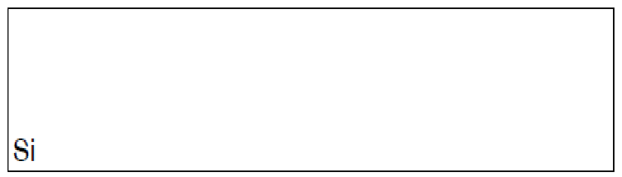
- Basics on the planar process: elements
  - Silicon n
    - N-doping, i.e. with more free electrons w.r.t. silicon
      - Diffusion and ion implantation
  - Silicon p
    - P-doping, i.e. with less electrons w.r.t. silicon
  - Insulator
    - SiO_2, obtained by exposing the silicon to oxygen
  - Conductor
    - Deposition of ALUMINIUM (Al) or COPPER (Cu)

(AS)IC

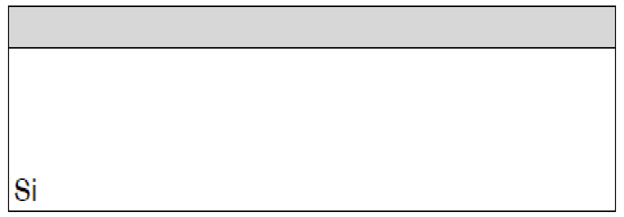
- Each of the elements is created during the planar process
  - All the regions are created at the same time on the entire slice of silicon
    - Each step is also called masking, since it requires the creation of a diffusion mask
  - Each phase has the role to prepare the silicon so that the part of the process considered (diffusion, oxidation, etc) takes place selectively in specific regions

(AS)IC

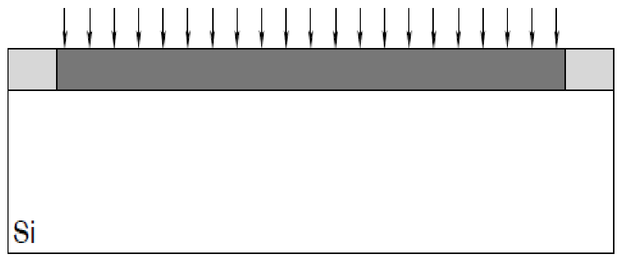
- Main phases of the planar process (1) - Etching



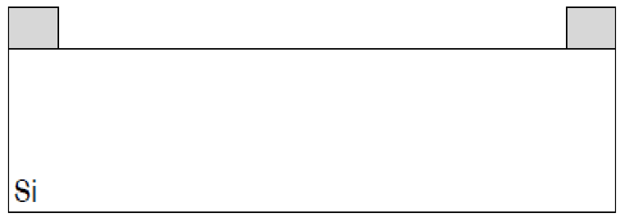
a. Wafer



b. Deposizione del photoresist



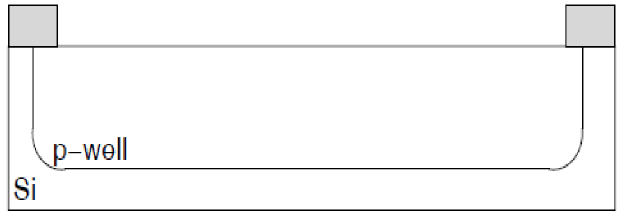
c. Mascheramento



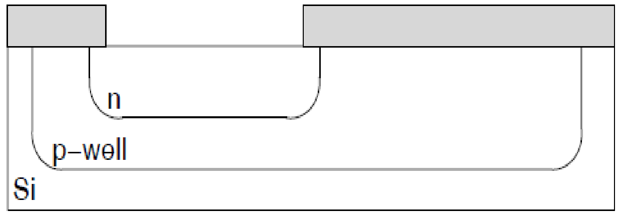
d. Lavaggio

(AS)IC

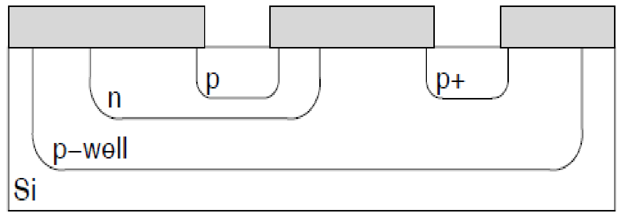
- Main phases of the planar process (2) - doping



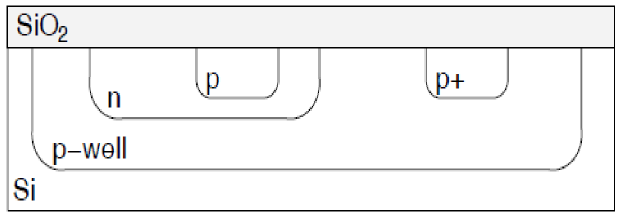
e. P-well



f. Base



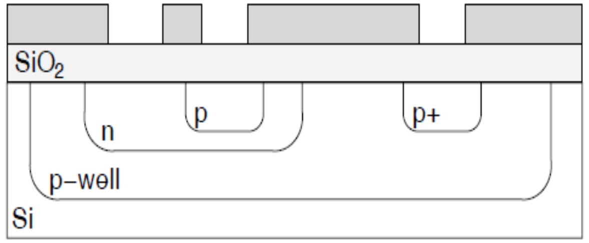
g. Collettore ed emettitore



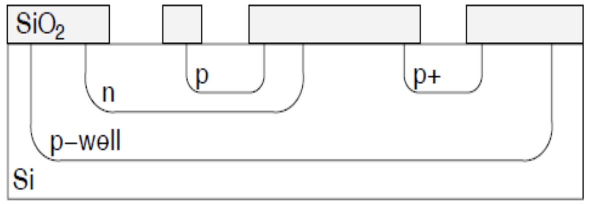
h. Ossidazione

(AS)IC

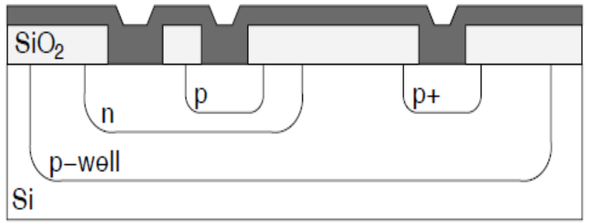
- Main phases of the planar process (3)



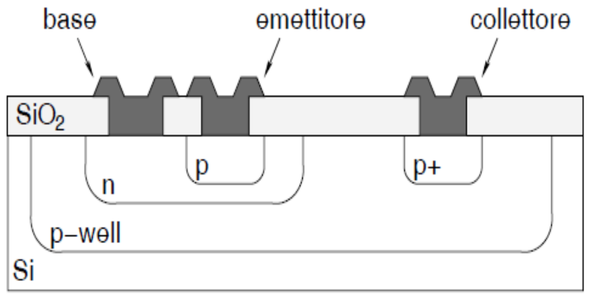
i. Mascheramento dell’isolante



l. Rimozione dell’isolante



m. Deposizione del conduttore

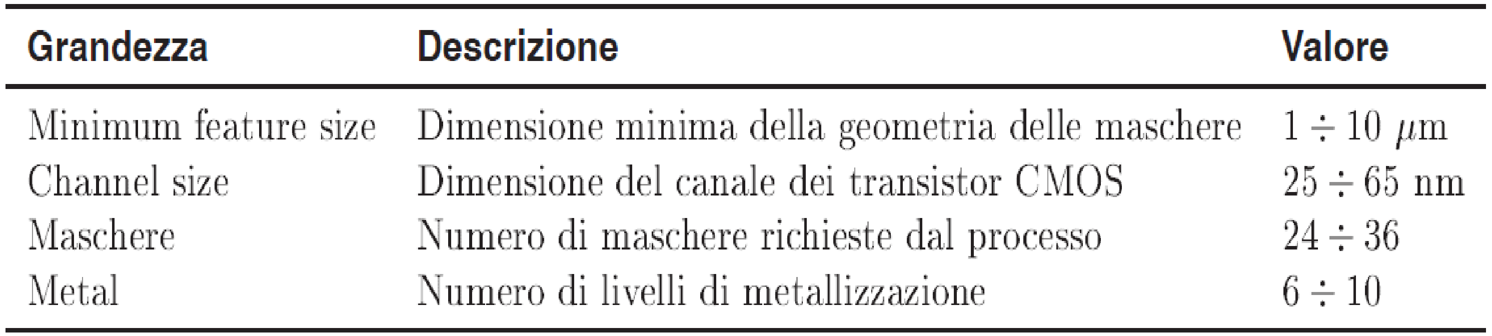


n. Transistor psn completo

(AS)IC

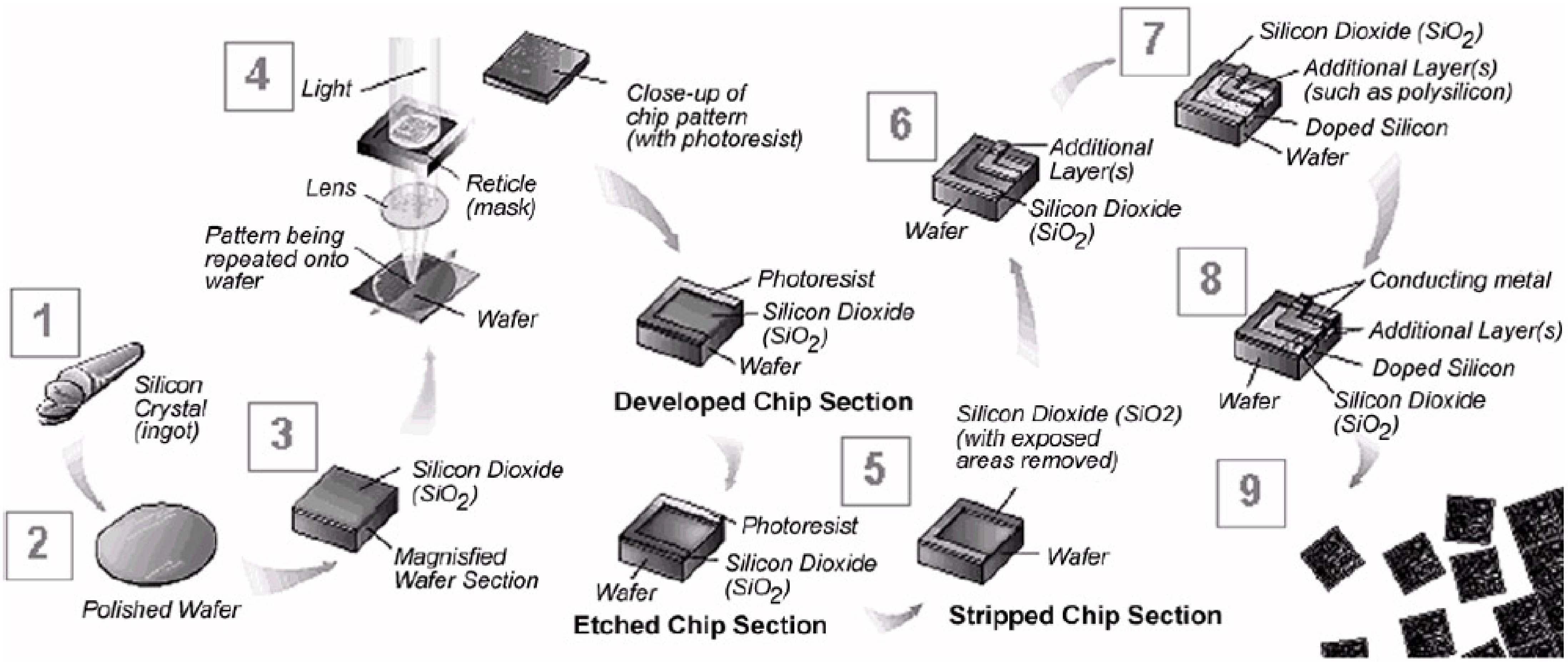
- Interconnections
  - It is not possible to separate the single networks (*net*)
    - Use of separate layers interposed with dielectric
      - *metal layer* or just called *metal*

- The previous process is for BJT, not replace by CMOS
  - Other steps with similar techniques



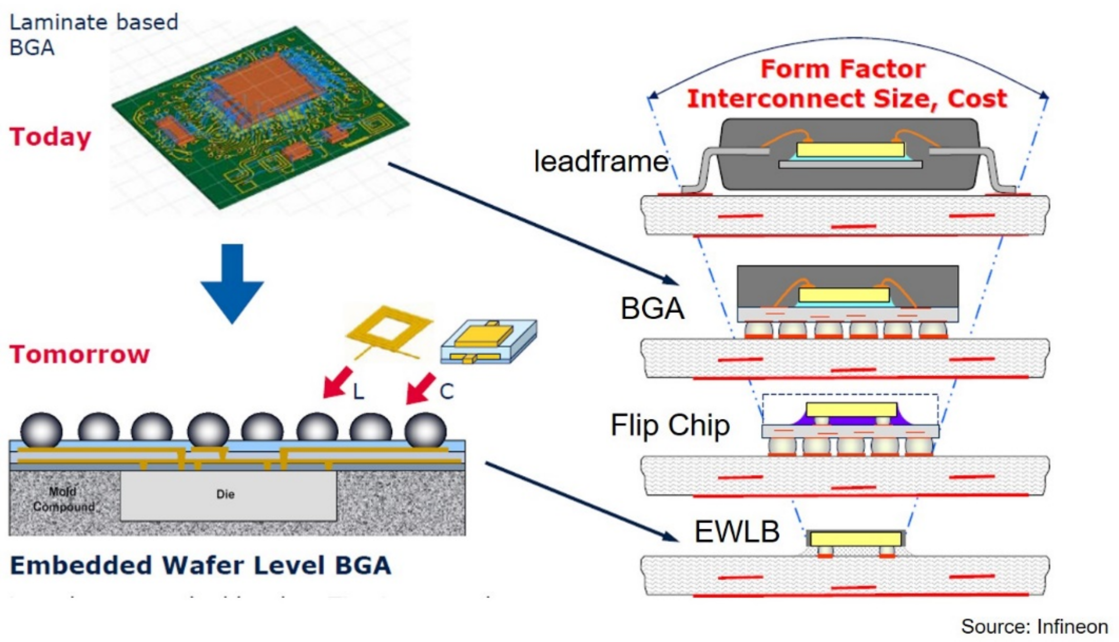
Grandezza	**Descrizione**	**Valore**
Minimum feature size	Dimensione minima della geometria delle maschere	$1 \div 10\ \mu$m
Channel size	Dimensione del canale dei transistor CMOS	$25 \div 65$ nm
Maschere	Numero di maschere richieste dal processo	$24 \div 36$
Metal	Numero di livelli di metallizzazione	$6 \div 10$

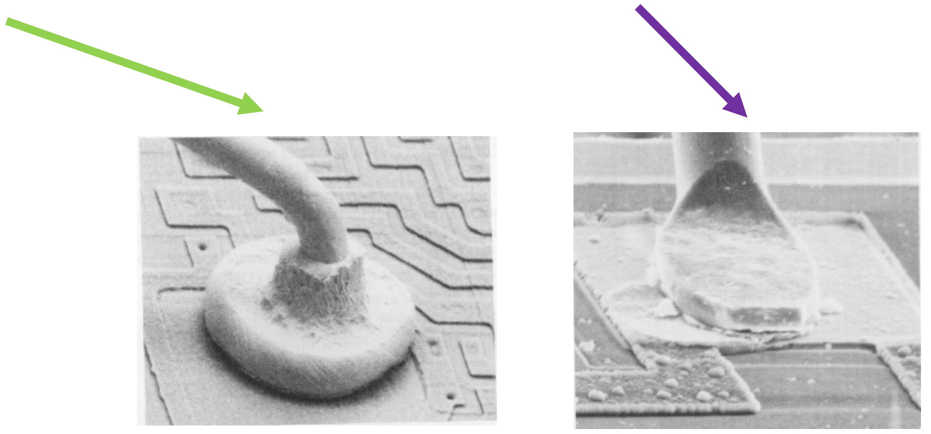
Complete view of the chain to produce a *die*

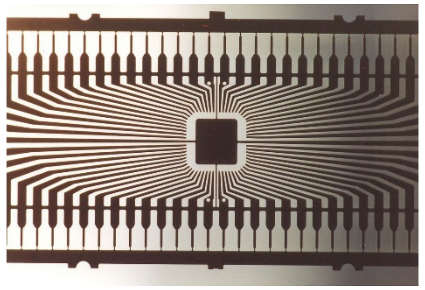

1
Silicon Crystal (ingot)
2
Polished Wafer
3
Silicon Dioxide (SiO_2)
Magnisfied Wafer Section
4
Light
Lens
Reticle (mask)
Pattern being repeated onto wafer
Wafer
Close-up of chip pattern (with photoresist)
Photoresist
Silicon Dioxide (SiO_2)
Wafer
Developed Chip Section
Photoresist
Silicon Dioxide (SiO_2)
Wafer
Etched Chip Section
5
Silicon Dioxide (SiO2) (with exposed areas removed)
Wafer
Stripped Chip Section
6
Additional Layer(s)
Silicon Dioxide (SiO_2)
Wafer
7
Silicon Dioxide (SiO_2)
Additional Layer(s) (such as polysilicon)
Doped Silicon
Wafer
8
Conducting metal
Additional Layer(s)
Doped Silicon
Silicon Dioxide (SiO_2)
Wafer
9

Bonding and packaging

- The bonding and packaging of an integrated chip provide connections to the outside world
- Usually, an IC can be packaged individually or incorporated into a hybrid circuit
- Packaging the circuit give it some mechanical and environmental stability
- Additional tests are carried out for chip electrical and mechanical behaviors after it is packaged
- The two common bonding techniques are thermocompression bonding and ultrasonic bonding



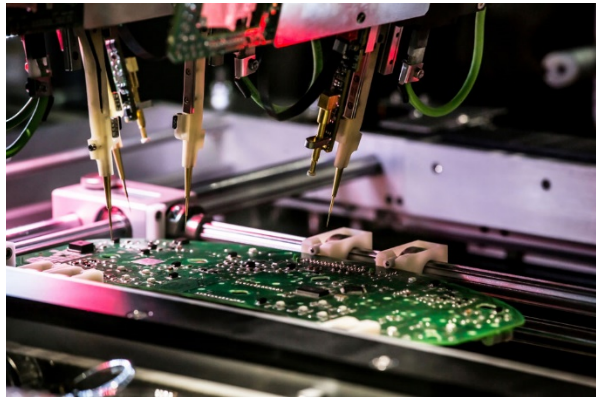


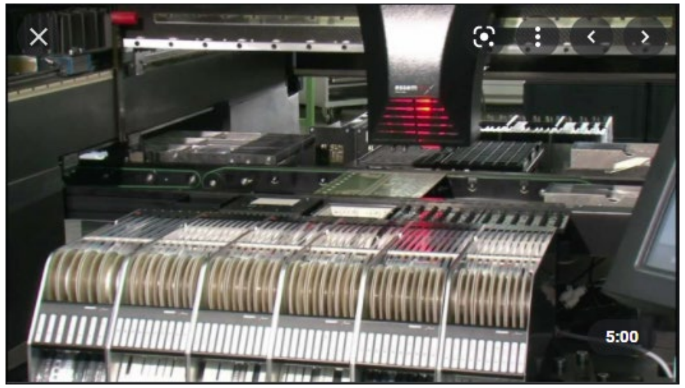


Most ICs are bonded to small IC packages

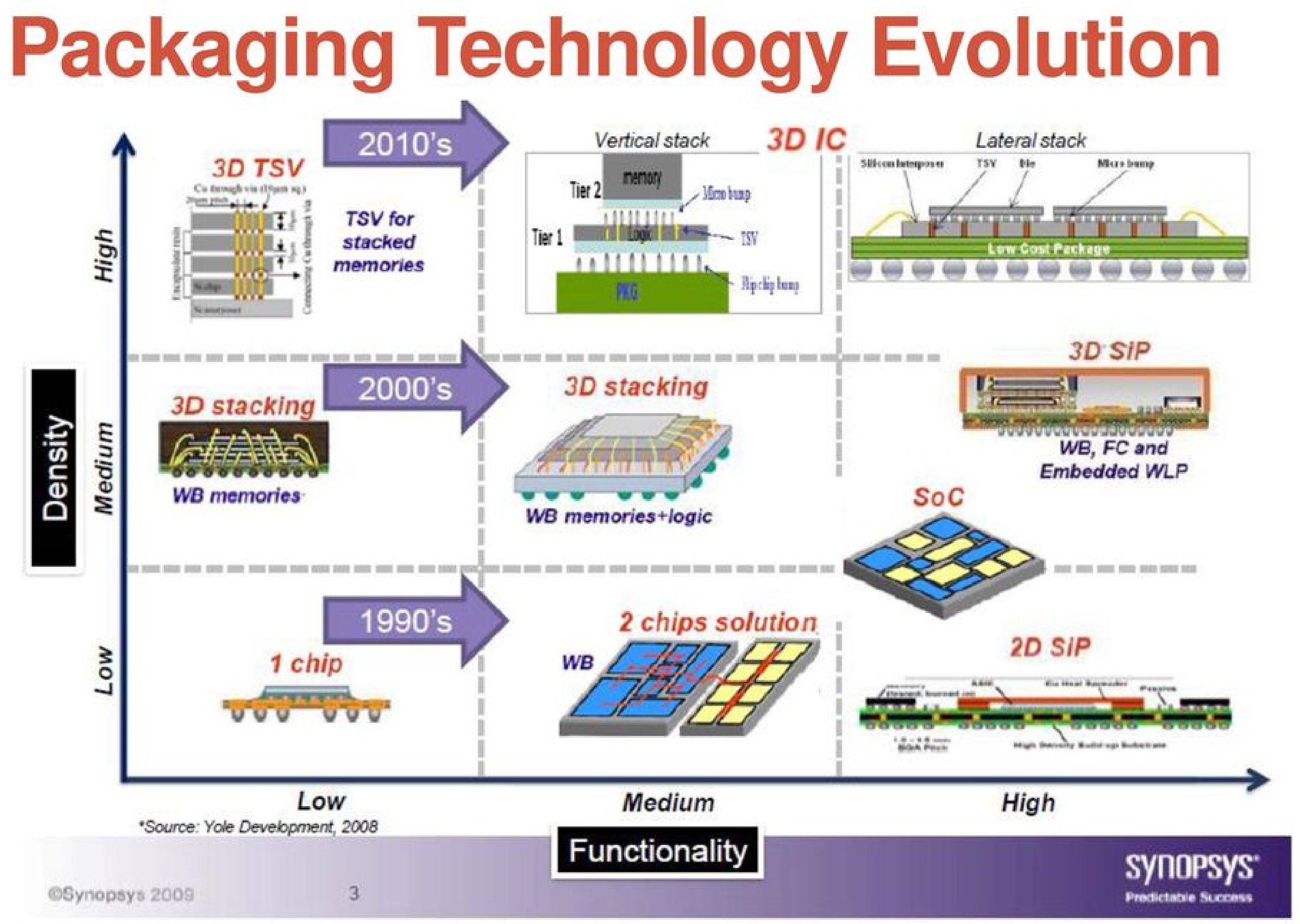
- It is possible to attach chips directly to boards
  - Method used extensively in low-cost consumer electronics
- Placing chips in packages enables independent testing of packaged parts, and eases requirements on board pitch and P&P (**pick-and-place**) equipment
- IC Packages- inexpensive plastic packages:
  - <200 pins- packages with >1000 pins available(e.g. Xilinx FF1704: 1704-ball flip-chip BGA)
- IC Packaging Materials- Plastic, ceramic, laminates (fiberglass, epoxy resin), metalDMT 243

Pick and place



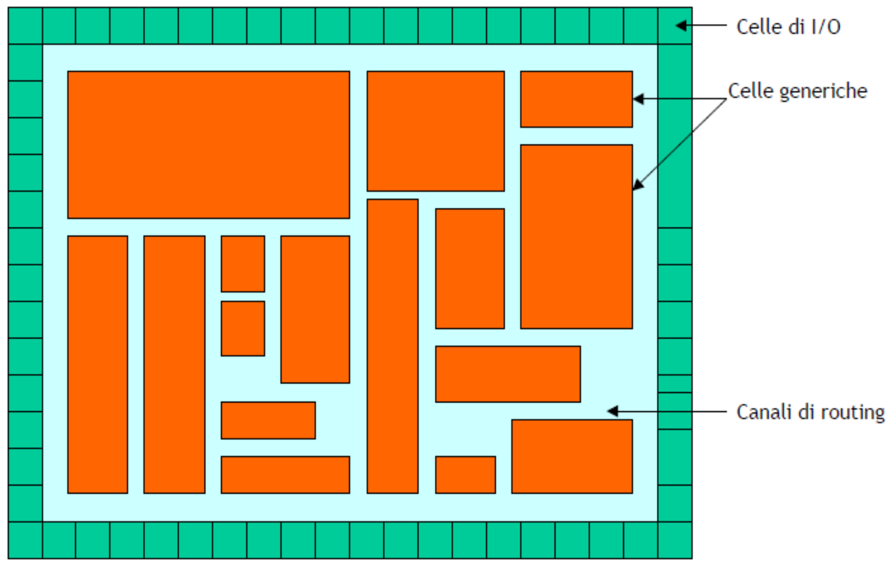


Never-ending evolution of IC packages



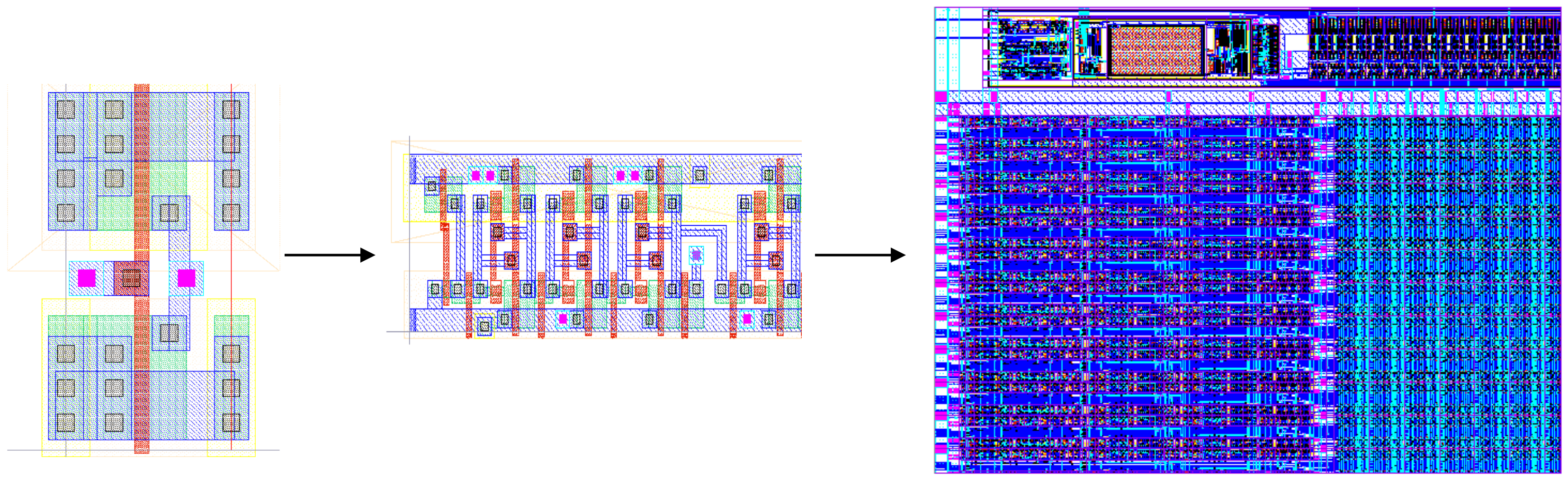
Full custom design of IC

- By exploiting the CMOS process it is possible to flexibly realize different configurations of transistors, in terms of size, positioning, etc
- Such flexibility is the base of the ***full custom*** design
  - Experienced designers
  - Hand-made geometry
  - Digital + Analog
  - Transistor-level simulation
  - High density
  - Superior performance
  - Long design times



Use of regular structures

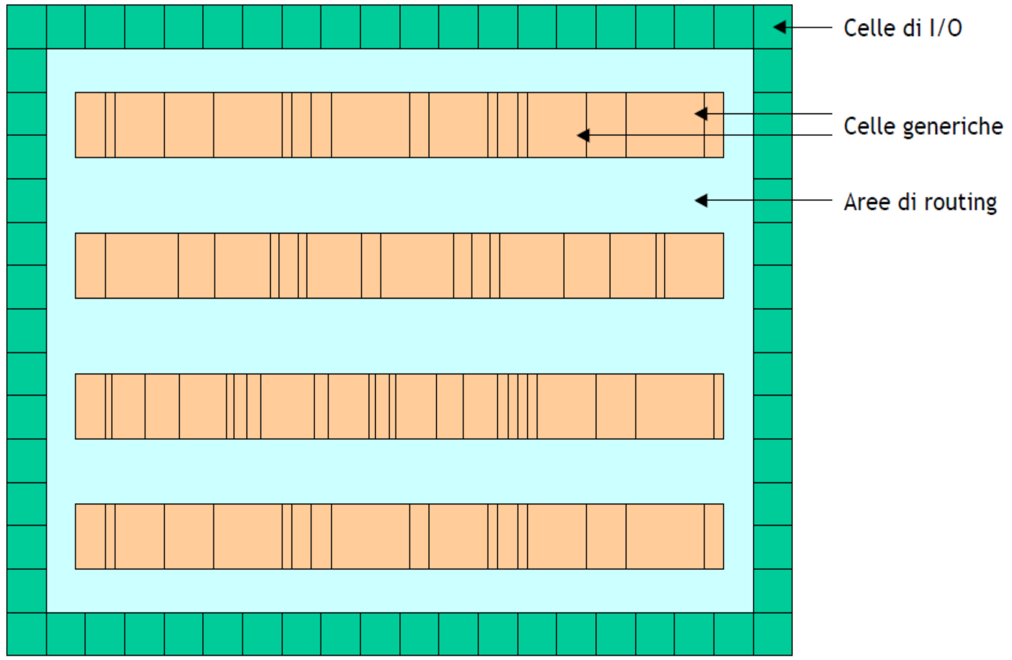
- To reduce the complexity of a full custom design
  - ***Standard cell:*** *e*xploitation fo regular and fixed structures

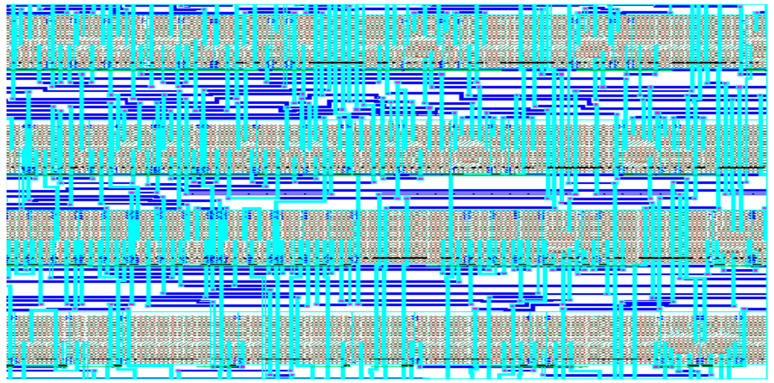


Standard cells: *higher abstraction of the design*

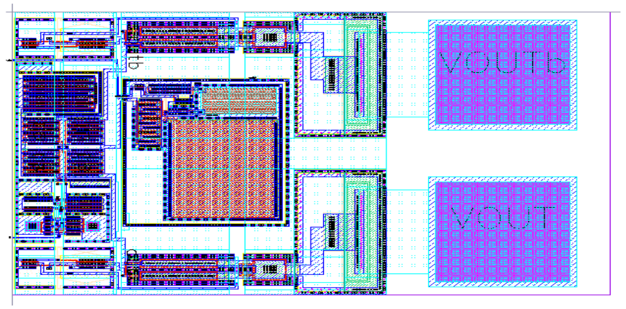
- A cell is pre-optimized, the designer does not know the details
    - Technology libraries of the different *silicon vendor*
        - 500-2000 types of cells to realize a wide range of functions: logic gates, complex combinatorial, flip-flops, memories, etc
    - Information for the designer
        - Reference name
        - Functional model
        - Non functional properties (low power, low noise, high density, …)
        - Number, name and position of the pins
        - Axis of symmetry or *flip*
        - Angles or rotation of *orientation*
- Simplified back-end stage: all the cells have the same headroom
    - Place and routing is simplified
        - Routing, is based on a grid containing the path of the nets
            - *Manhattan Geometry*
    - *Static timing analysis* is also simplified

Standard cells

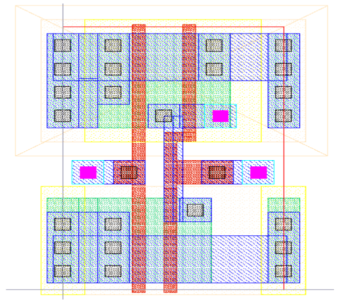




Routing



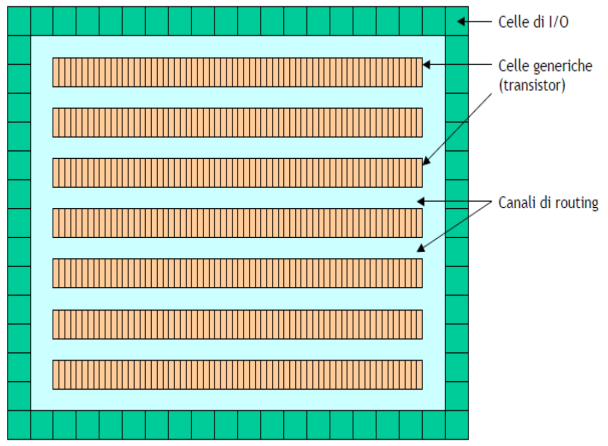
I/O cell



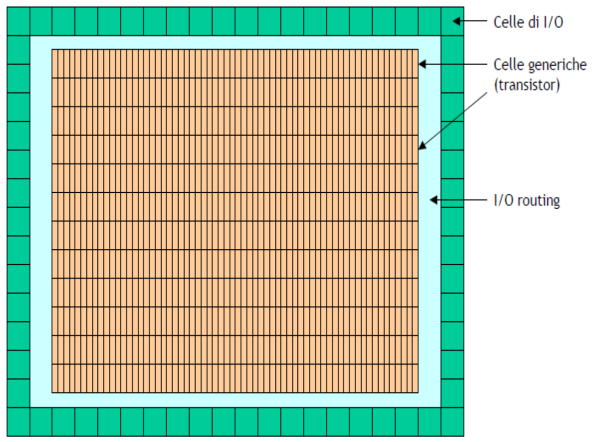
Cell

Use of regular structures

- It is possible to realize generic circuits, using very regular structures
  - ***gate array*** or ***sea of gates***
    - Pre-fabricated part of the active part of transistors, the contact are left disconnected. Cells are transistor or NAND gates
    - The designer has only to map the interconnections and the planar process is only the phases to create metal layers
  - Still a cell-based approach



Channel based



Sea of gates

Programmable and Reprogrammable Technologies

Programmable logic

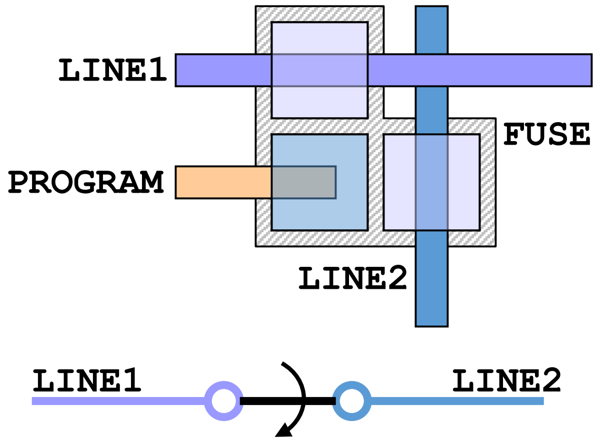
- Availability of a wide range of hw devices (gates, flip-flops, buffers, …) where the designers can decide only how to connect, based on the needs of the application
- Types of programmable devices
  - ROM, PLA, PAL
  - GAL, CPLD
  - FPGAs

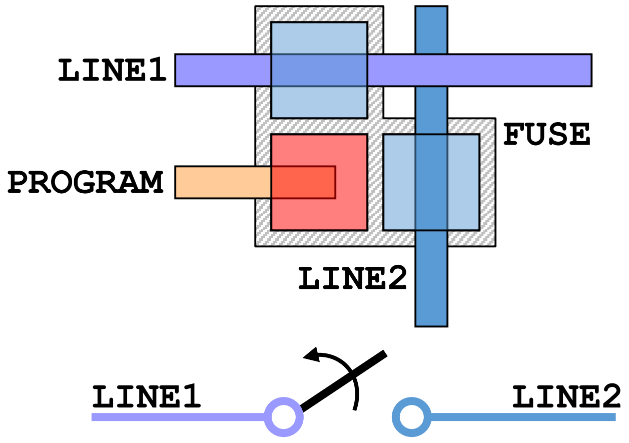
Programmable logic

- Classification
  - *Programming modes*
    - *One-Time Programmable* - OTP
      - Fuse, Antifuse
    - *Reprogrammable*
      - E^2PROM, SRAM, Flash
    - *Reprogrammable* & *Reconfigurable*
      - SRAM, Flash
  - *Connections*
    - Global
    - Local and distributed

Programming

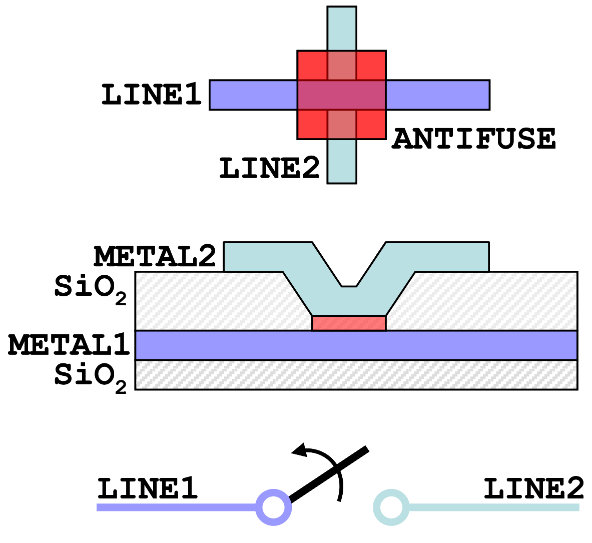
- Fuse (*OTP*)
  - Lines are always connected
  - The programming burns (fuse) some connections to preserve only those that are necessary
  - Programming requires high voltage w.r.t. the power supply of the logic

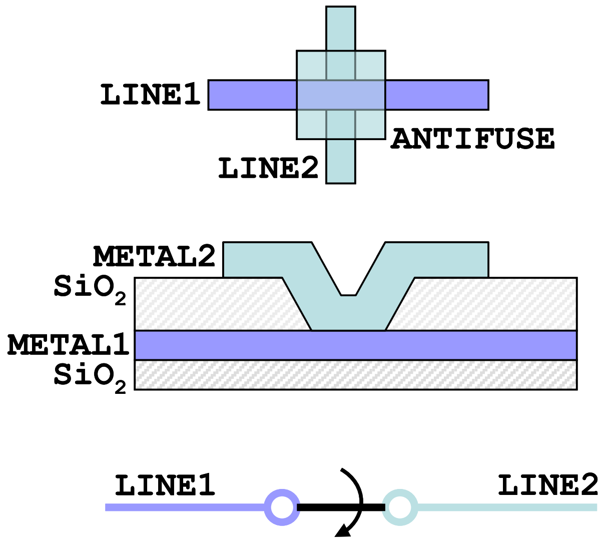




Programming

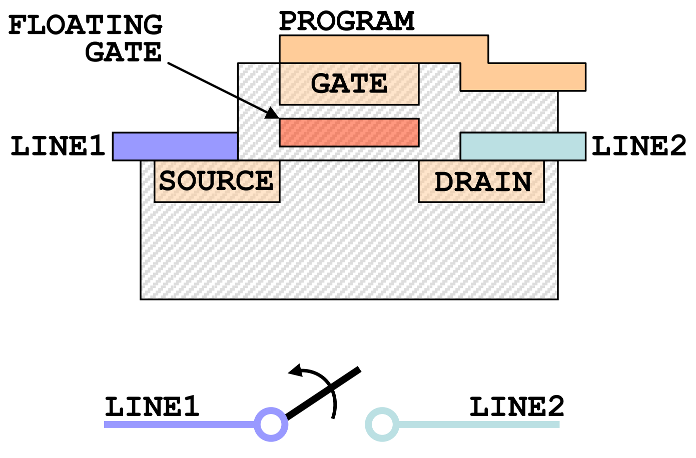
- Antifuse (*OTP*)
  - Lines are always disconnected
  - Programming «creates» (*antifuse*) the connections needed
  - Programming requires high voltage w.r.t. the power supply of the logic

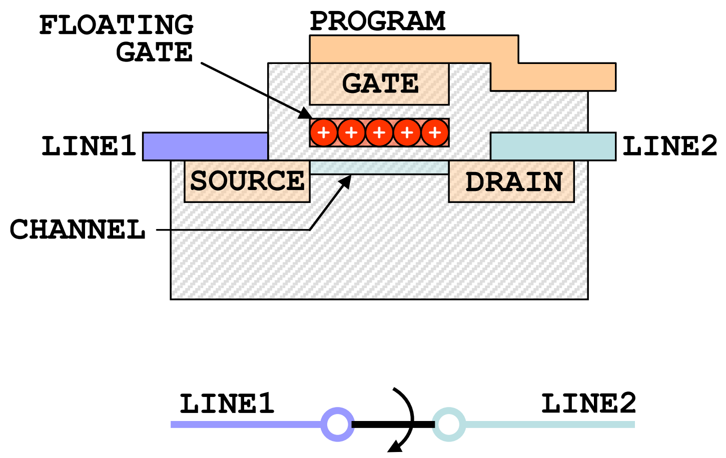




Programming modes

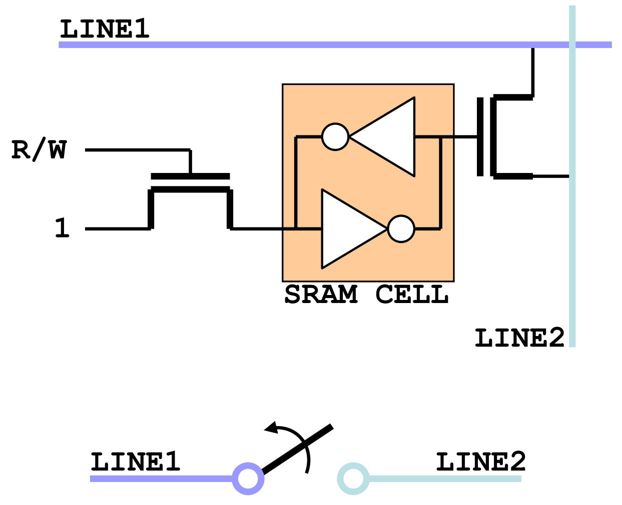
- E²PROM (*Reprogrammable*)
  - Not disruptive connections among lines, which can be modified when the systems is not in operation
    - Programming consist in placing a charge in the *floating gate* of a transistor to create/destroy the conductive channel

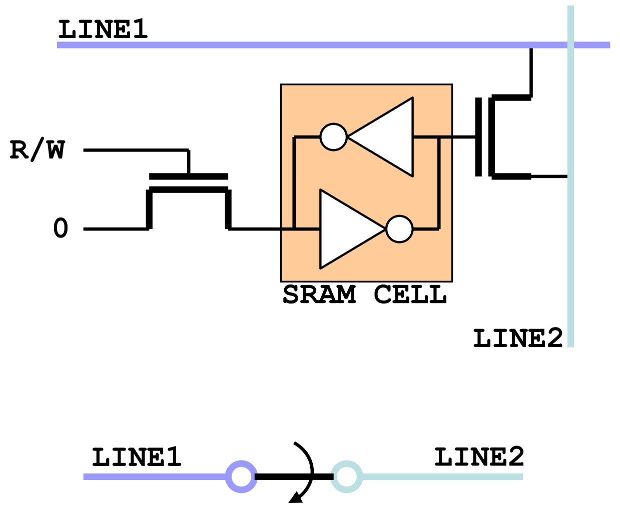




Programming modes

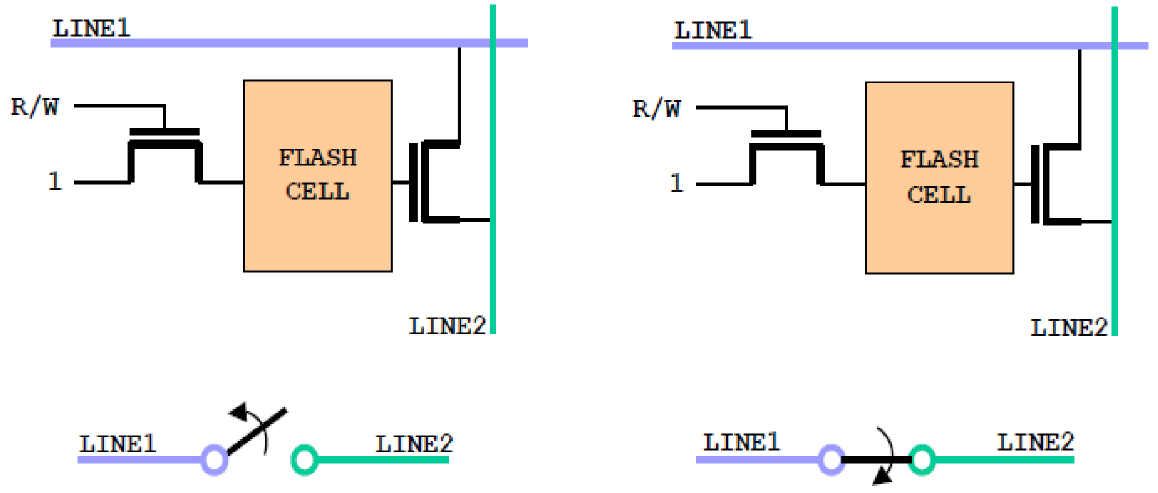
- SRAM (*Reprogrammable* e *Reconfigurable*)
  - Not disruptive connections among lines, which can be modified electrically
    - Programming is storing a logic value (0 or 1) in a static RAM cell
      - volatile
    - Depending on the technology, it can happen
      - When the system in not in operation (off line) (*Reprogrammable*)
      - When the system is working (*Reconfigurable*)





Programming modes

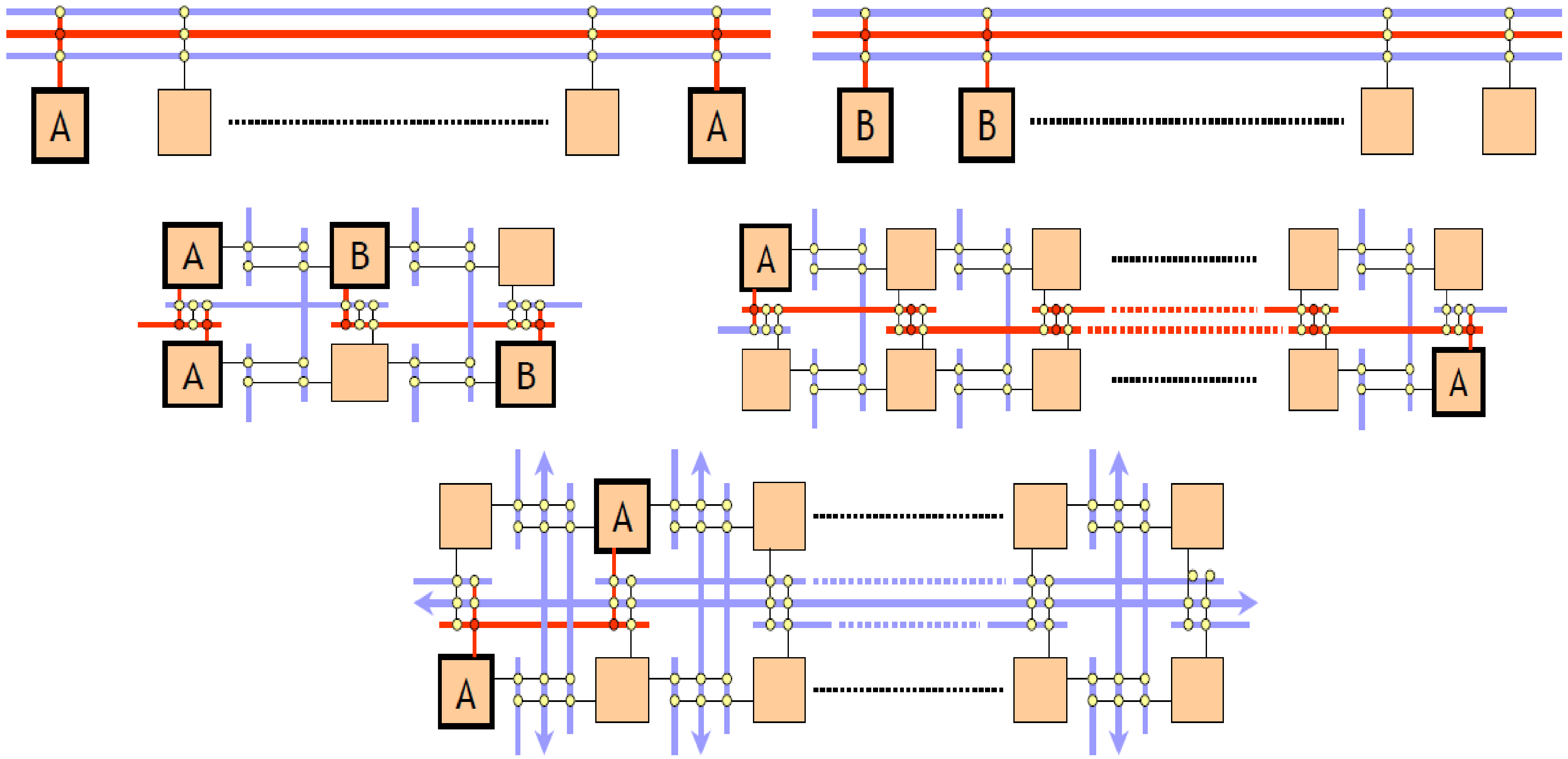
- Flash (*Reprogrammable* e *Reconfigurable*)
  - The connections among lines can be activate/deactivate electrically and in a not disruptive way
    - Programming is storing 0 or 1 in a cell of a FLASH memory
      - Not volatile



Connections

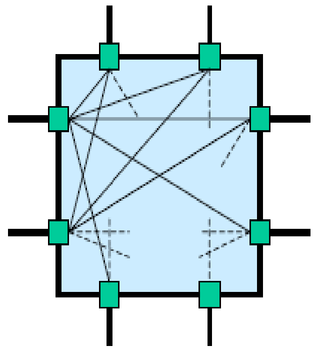
- Global connection
  - Can be shared by many components of the device
    - Long delays
    - Low flexibility: can be the output of only a logic element
- Local connection
  - Shared by a few logic elements
    - Reduced delay
    - Several local lines can coexists inside the same device with different lengths
      - Routing flexible while complex
- Hierarchical connectivity

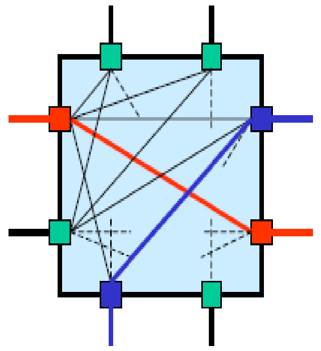
Connections

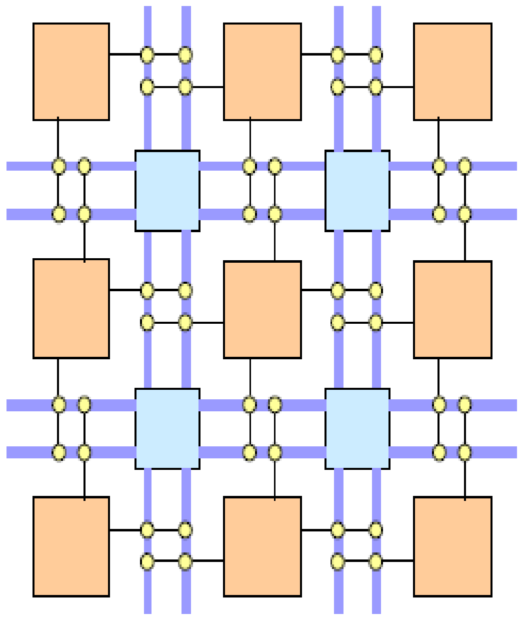


Connections

- Other types
  - *Programmable Switch Matrix*







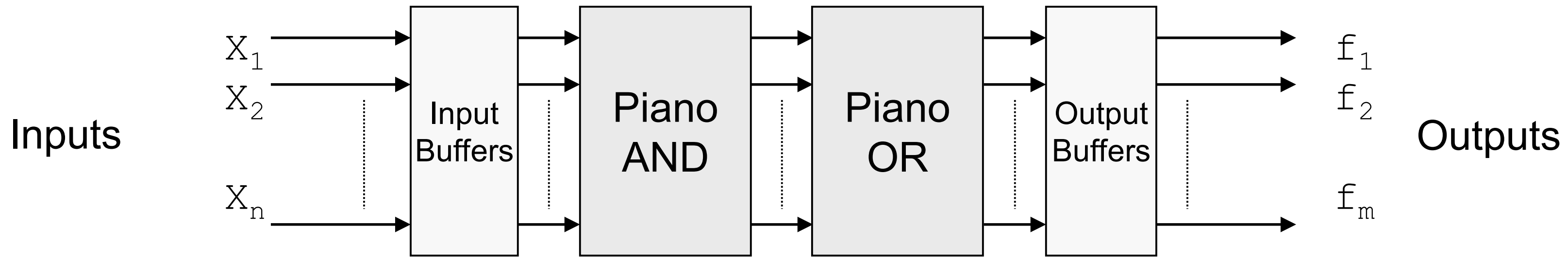
Connections

- Global connections
    - 2-level programmable logic devices
        - Read-Only Memory (ROM/PROM)
        - Programmable Logic Array (PAL)
        - Programmable Array Logic (PLA)
    - Advanced PAL and PLA
    - Generic Array Logic (GAL)
    - Complex Programmable Logic Devices (CPLD)
- Local and distributed connections (short and long)
    - Field Programmable Gate Array (FPGA)

2-level programmable logic

General structure

- Conceived for generic functions: `n` inputs and `m` outputs
  - `f`$_i$ `= f`$_i$`(x`$_1$`, x`$_2$`, ..., x`$_n$`) con i={1, 2, ..., m}`
    - Possibility to create multi-level logic by exploiting feedback lines, or FSMs by adding memory elements
  - Elements
    - Fixed number of in and out (see family of devices)
    - AND plane - to create minterms
    - OR plane – to select the minterms
    - Buffers – to avoid fan-in and fan-out problems

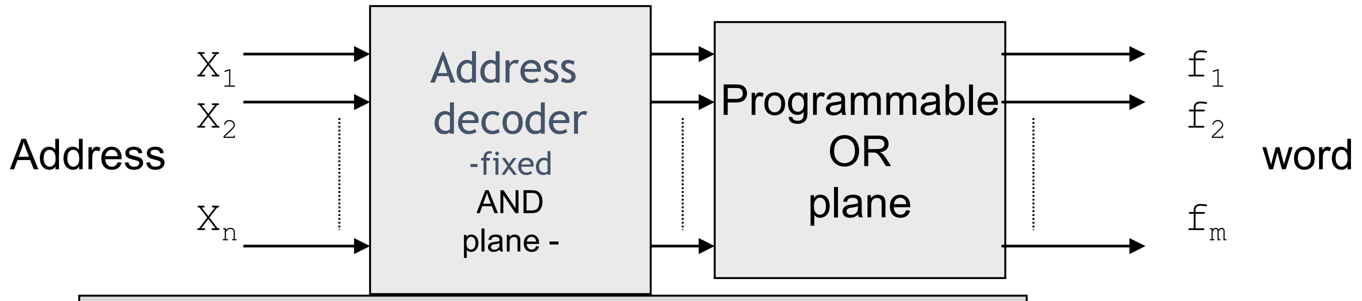


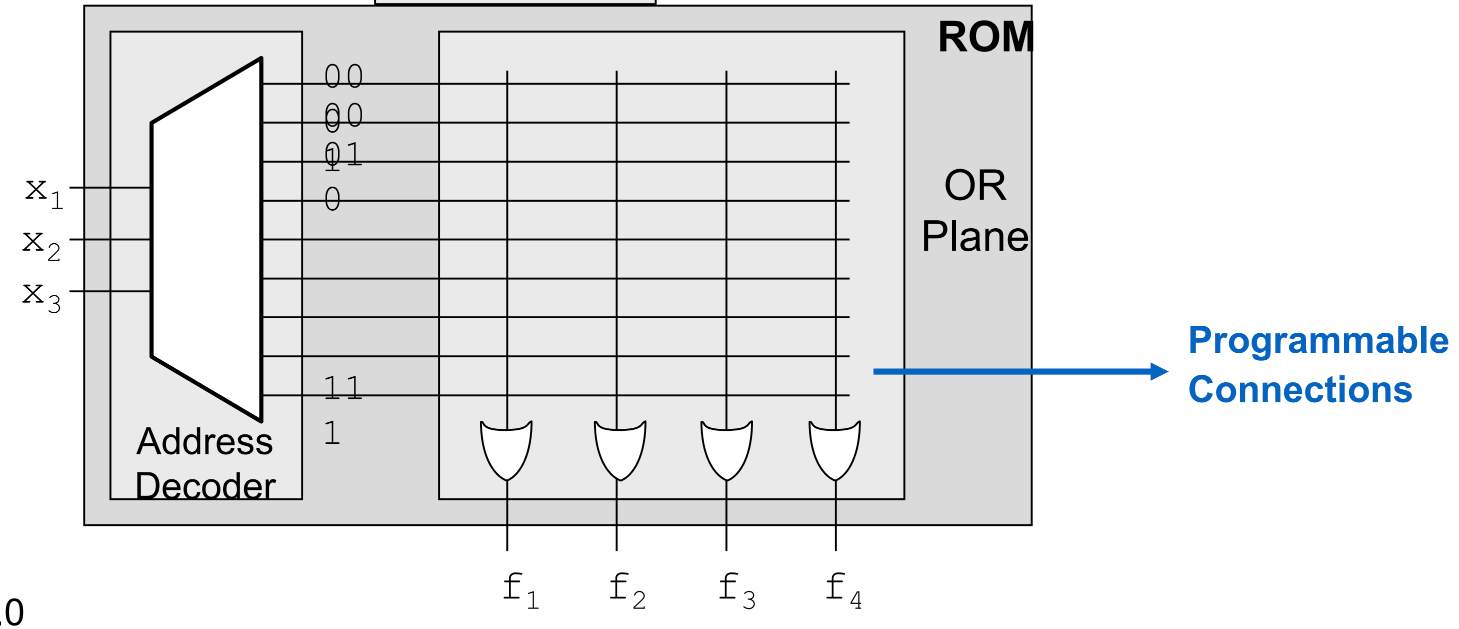
Main types

- *Read-Only Memory (ROM/PROM)*
  - Fixed AND Plane
    - Implements all the possible minterms (decoder)
  - OR plane, permanent modification
- *Programmable Logic Array (PLA)*
  - Programmable AND plane
    - Implement only the necessary minterms
  - Programmable OR plane
- *Programmable Array Logic (PAL)*
  - Programmable AND plane
    - Implement only the necessary minterms
  - Fixed OR plane
    - Constraint on the number of outputs

Read-Only Memory (ROM/PROM)

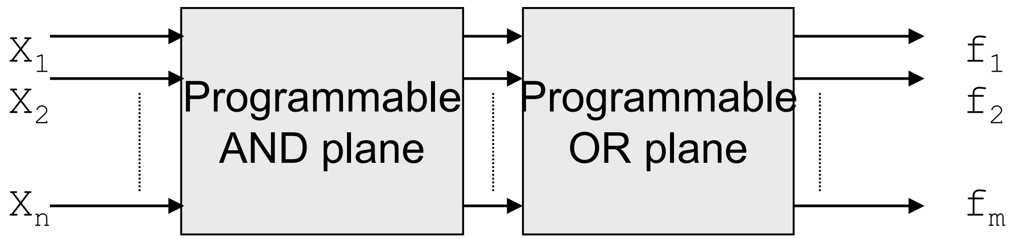
- Sum of products (SoP), multiple outputs (word size)

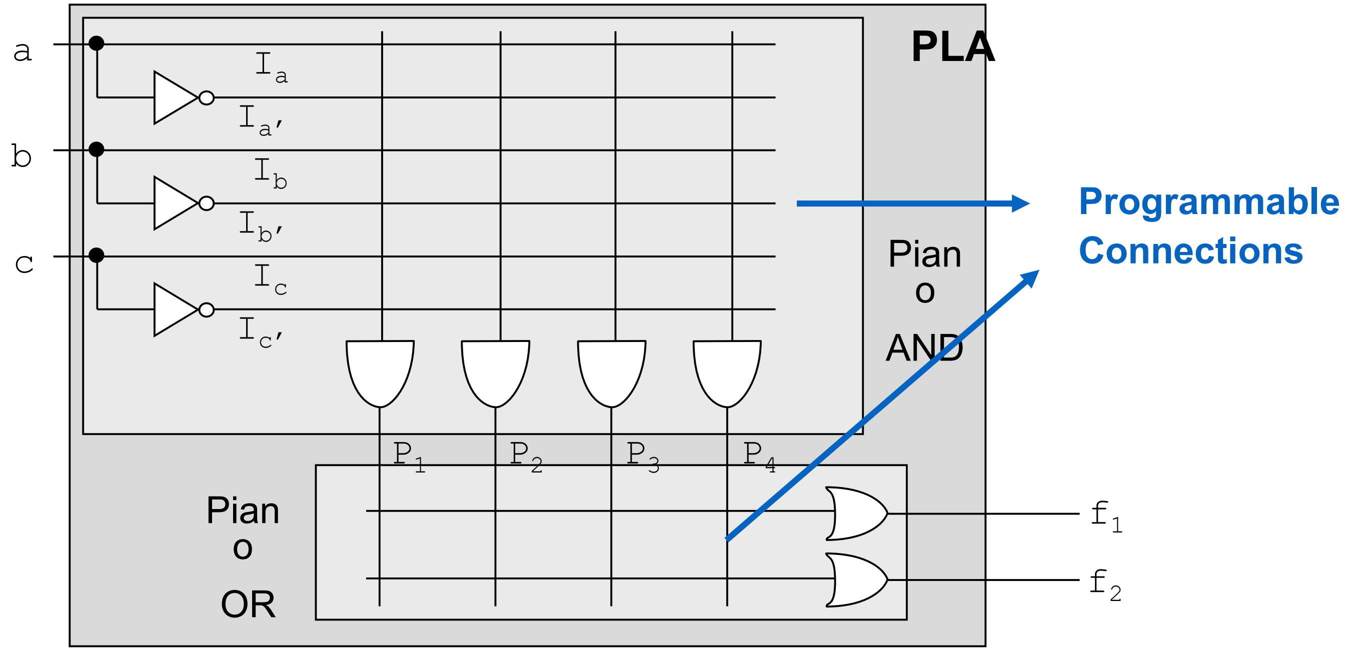




Programmable Logic Array (PLA)

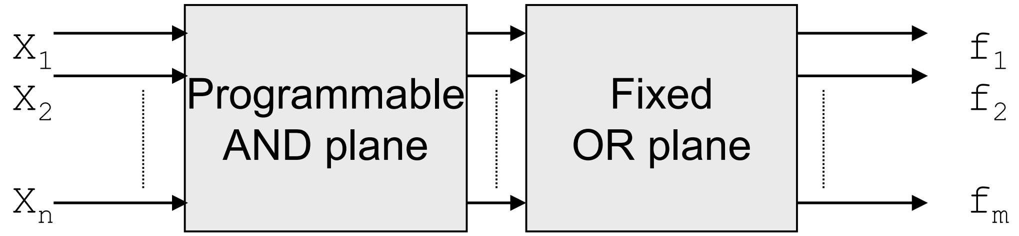
- PLA implements a 2-level sum of products

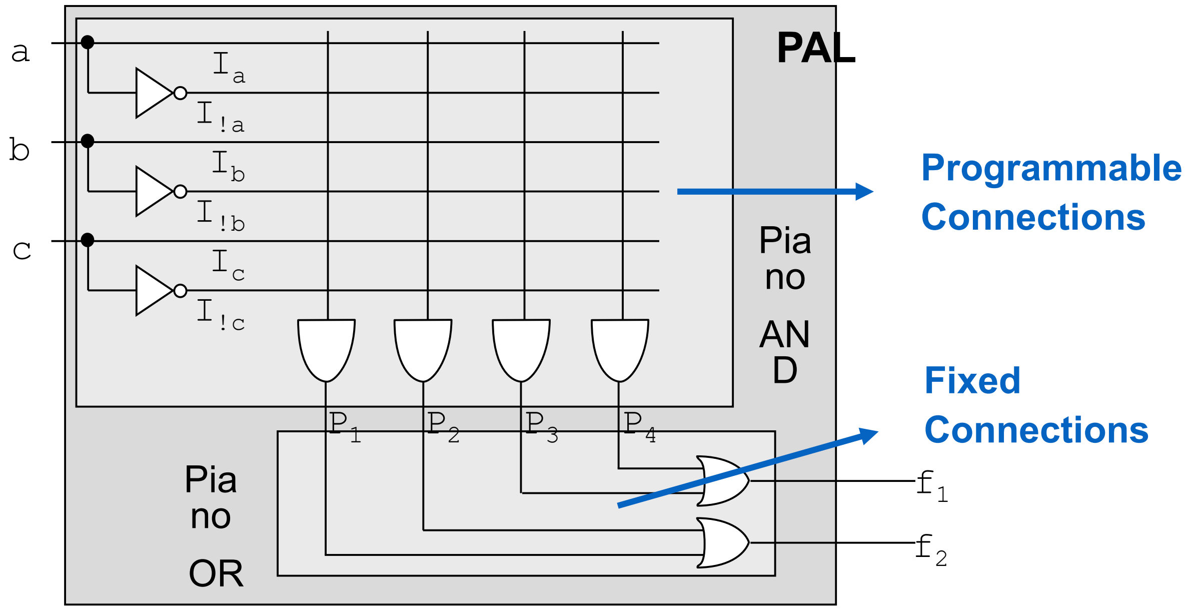




Programmable Array Logic (PAL)

- PAL implements a 2-level form, properly minimized
  - PLA and PAL are used in the same application field

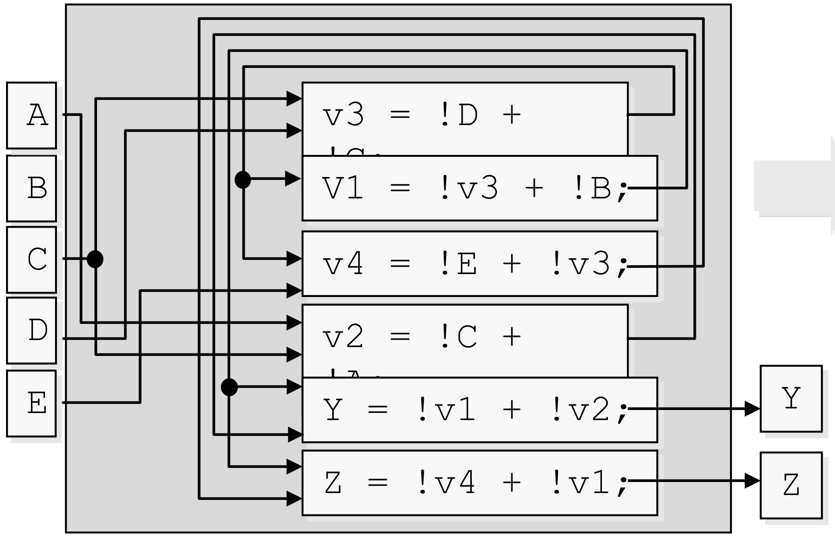


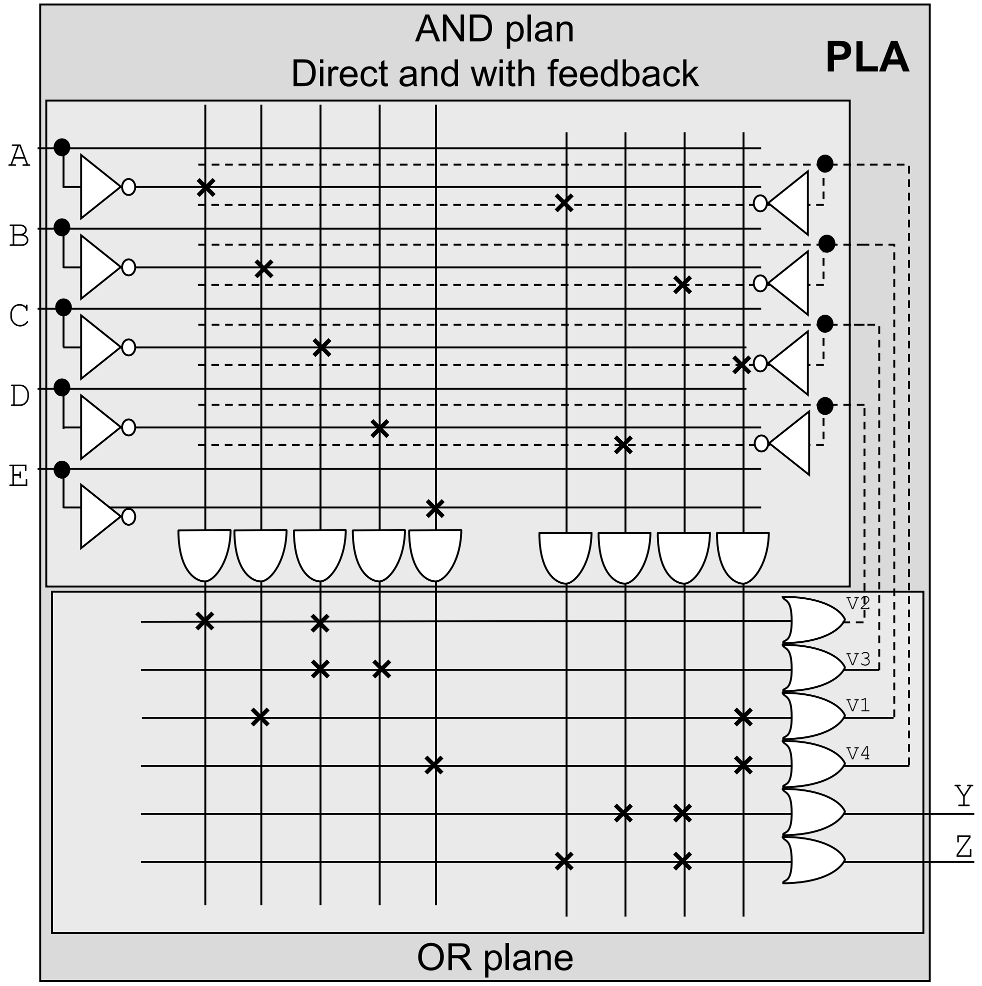


Multi-level programmable logic

Multi-level logic

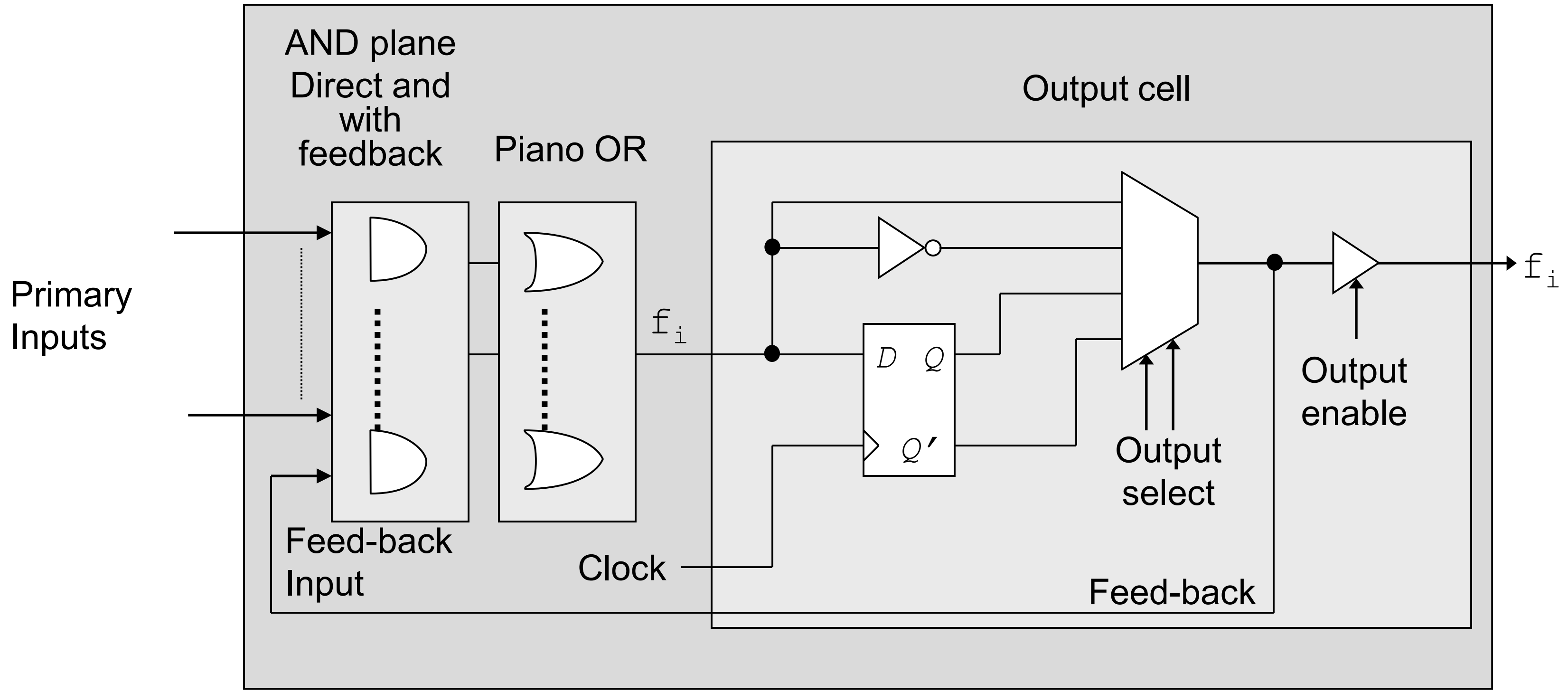
- Advanced PLA and PAL
  - Feedback lines





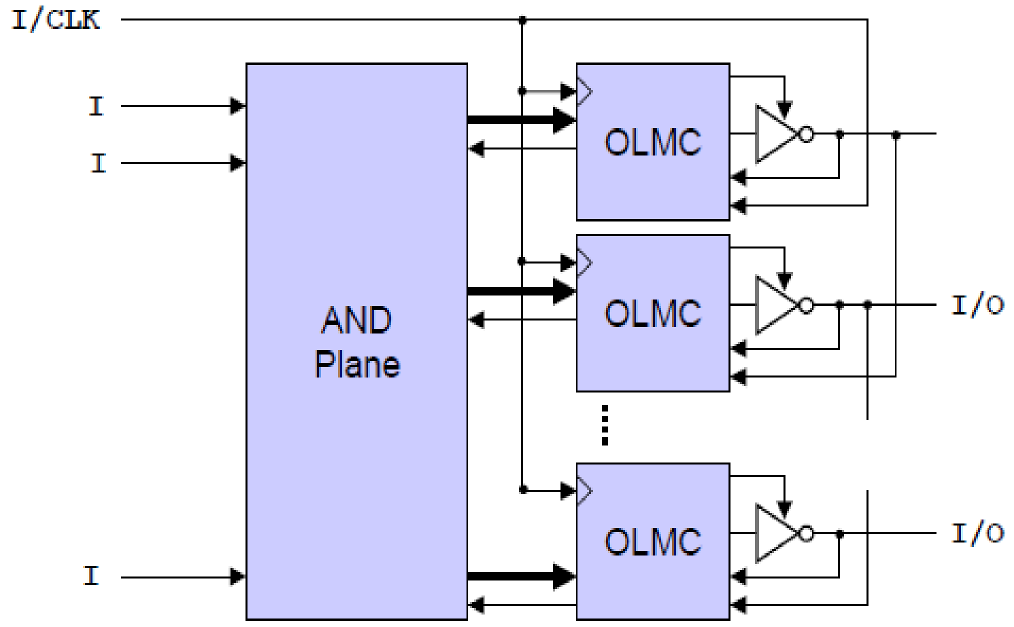
Advanced PLA and PAL

- Presence of memory elements to implement FSMs



Generic Array Logic (GAL)

- GAL generalized the structure of PAL/PLA
    - The output cells OLMC or *Output Logic Macro Cell,* increase the flexibility of the design

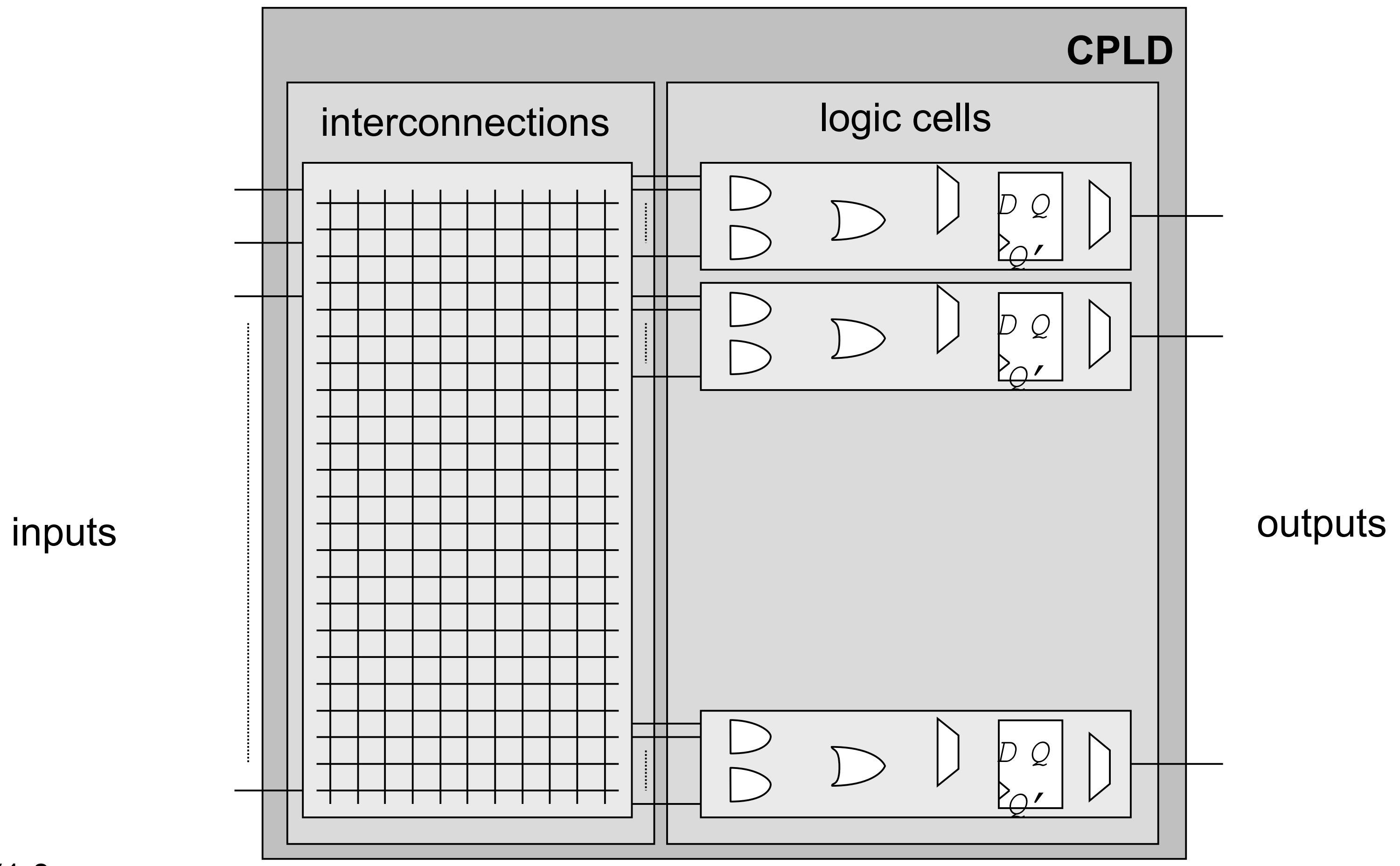


Complex Programmable Logic Devices

- *Complex Programmable Logic Devices – CPLD,* are an evolution of PLA, PAL and GAL
  - Characteristics
    - Global connections
    - Logic grouped
  - With respect to PAL, PLA and GAL
    - Higher level of integration
    - More complex cells
    - Higher performance
    - Structure more regular and easy to program

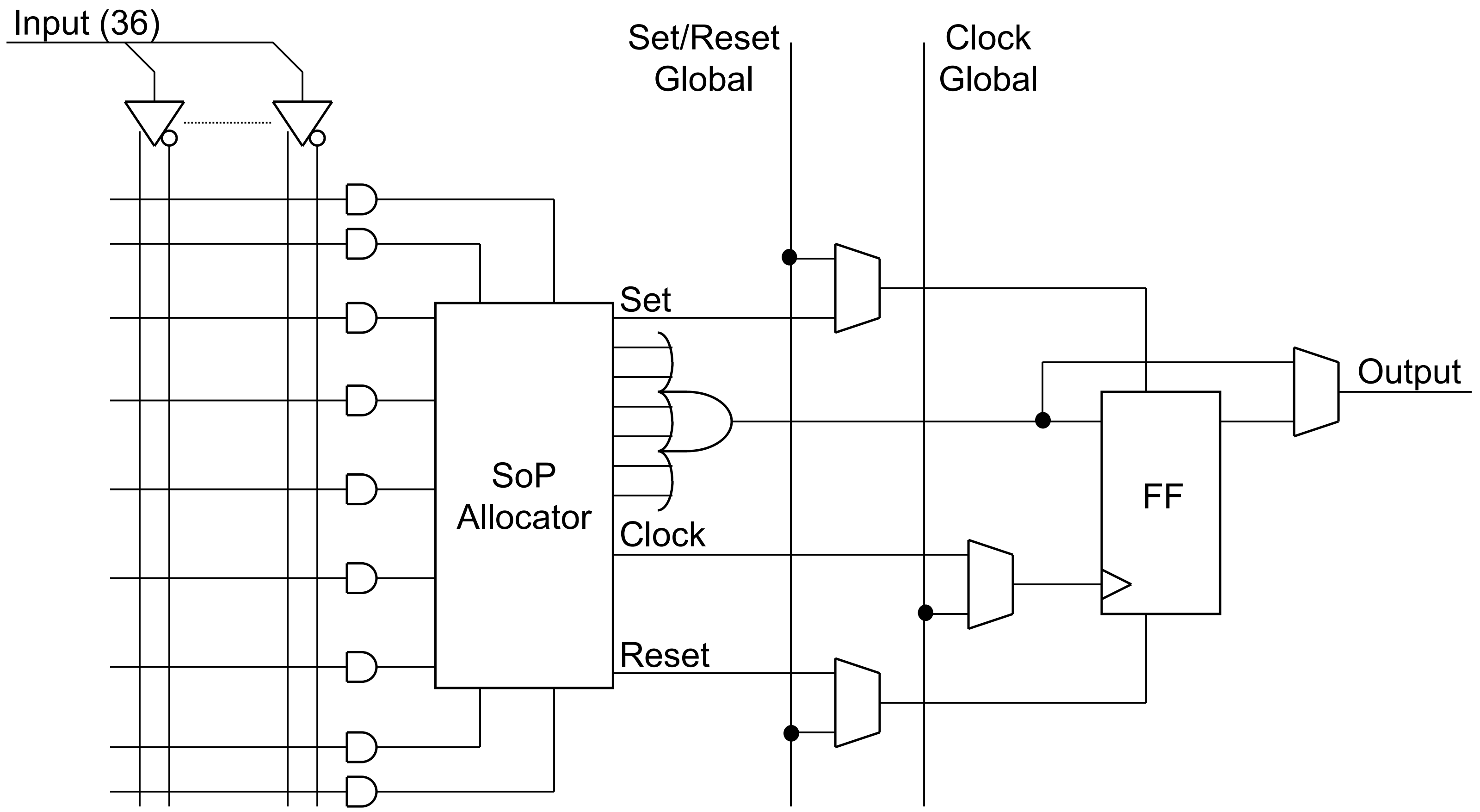
Complex Programmable Logic Devices

- General architecture



Complex Programmable Logic Devices

- Example: logic cell of *Xilinx XC9500*



Field Programmable Gate Arrays

FPGA

FPGA

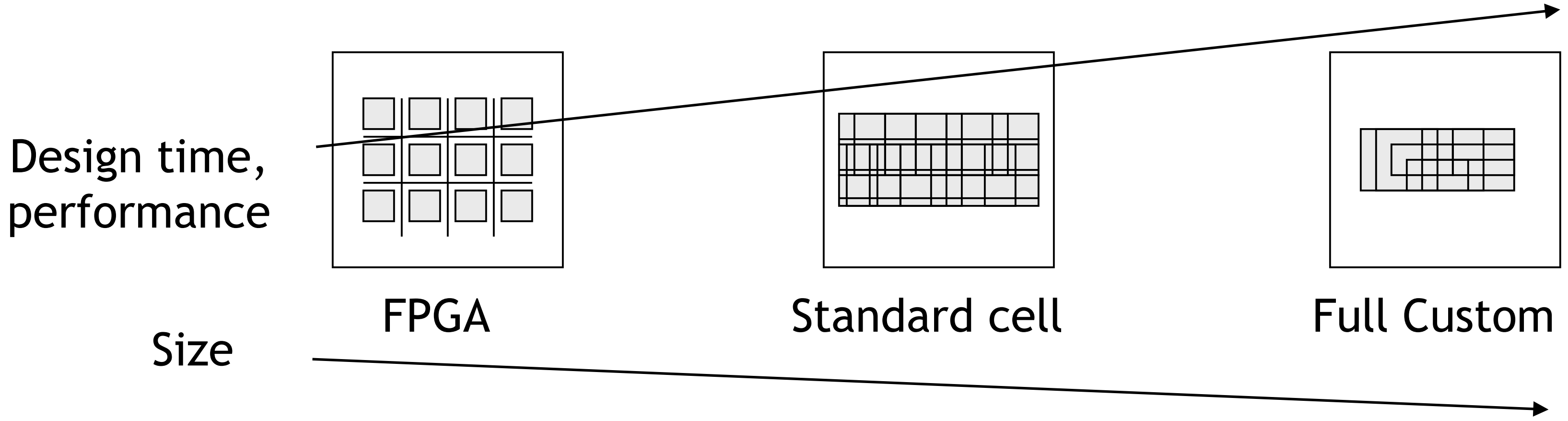
- FPGA - programmable device where an array of components can be connected
  - *Logic components* (gates, Flip-flop, Buffer…)
    - Very complex
      - Possibility to realize complex functionality at very high speed, even with low resource utilization
    - Low complexity
      - Better resource exploitation, paid in terms of connectivity resources
  - *Connections local and distributed*
      - Short local lines
        - Delay and power consumption limited
      - Flexibility. Inside a device several types of lines coexists

FPGA – main aspects

- Compromise between flexibility and performance
  - PLD $\Rightarrow$ FPGA $\Leftarrow$ MPGA
- Balance between cost and performance
  - FPGA vs. ASIC
- Good for verification of complex systems before volume
  - Fast prototyping…(it was actually the initial purpose)
- Compromise between general purpose and custom attitude
  - SW (GPP, DSP) $\Rightarrow$ FPGA $\Leftarrow$ HW (ASIC)

FPGA – other aspects

- Possibility to modify dynamically the configuration aka the application
    - Hardware Multi-modal platform (*off-line*)
        - Configuration are stored in a ROM and copied on a RAM before use
    - Reconfigurable platform (*on-line*)
        - The configurations (or part of them) are available in RAM and swapped or copied during operation
- Comparison between technologies

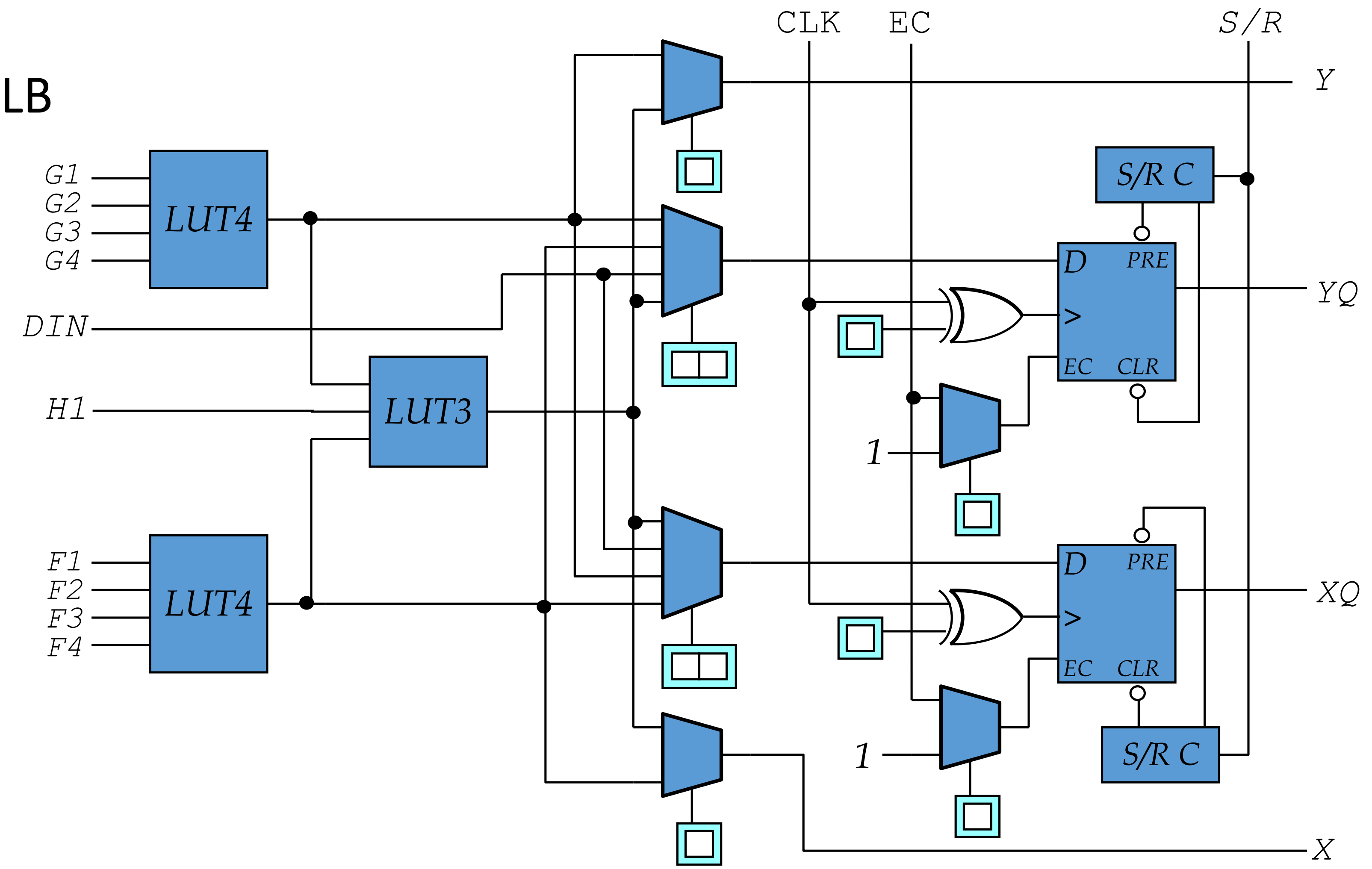


Comparisons

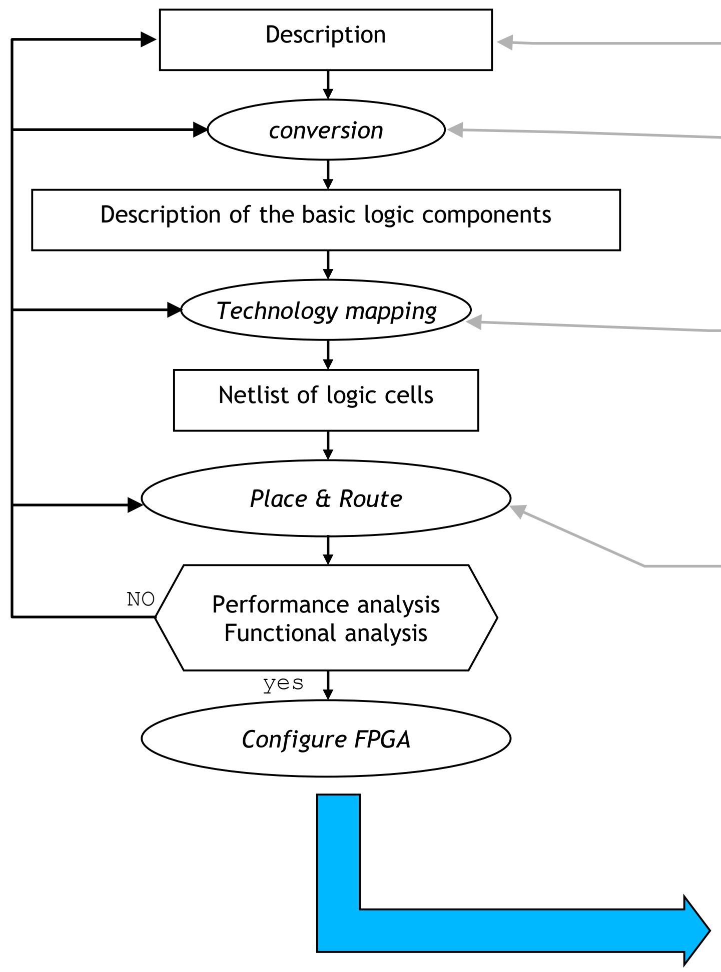
	FPGA	Gate array	Standard cell	Full custom
Densità	Basso	Medio	Medio	Alto
Flessibilità	bassa (alta)	Basso	Medio	Alto
Analogico	No	No	No	Si
Prestazioni	Basso	Medio	Alto	Molto alto
Tempo progetto	Basso	Medio	Medio	Alto
Costo progetto	Basso	Medio	Medio	Alto
Tools	Semplice	Complesso	Complesso	Molto complesso
Volume	Basso	Medio	Alto	Alto

FPGA

- Logic Block
  - Example of a CLB



FPGA – Design flow



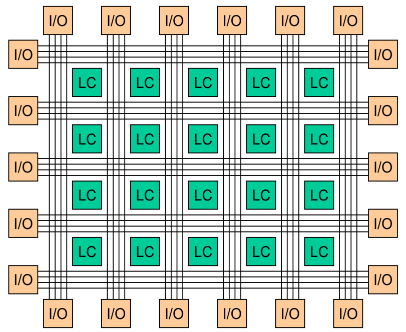
Schematic, FSMs, logical expressions, HDLs

Unified description of the design
It can include some optimizations

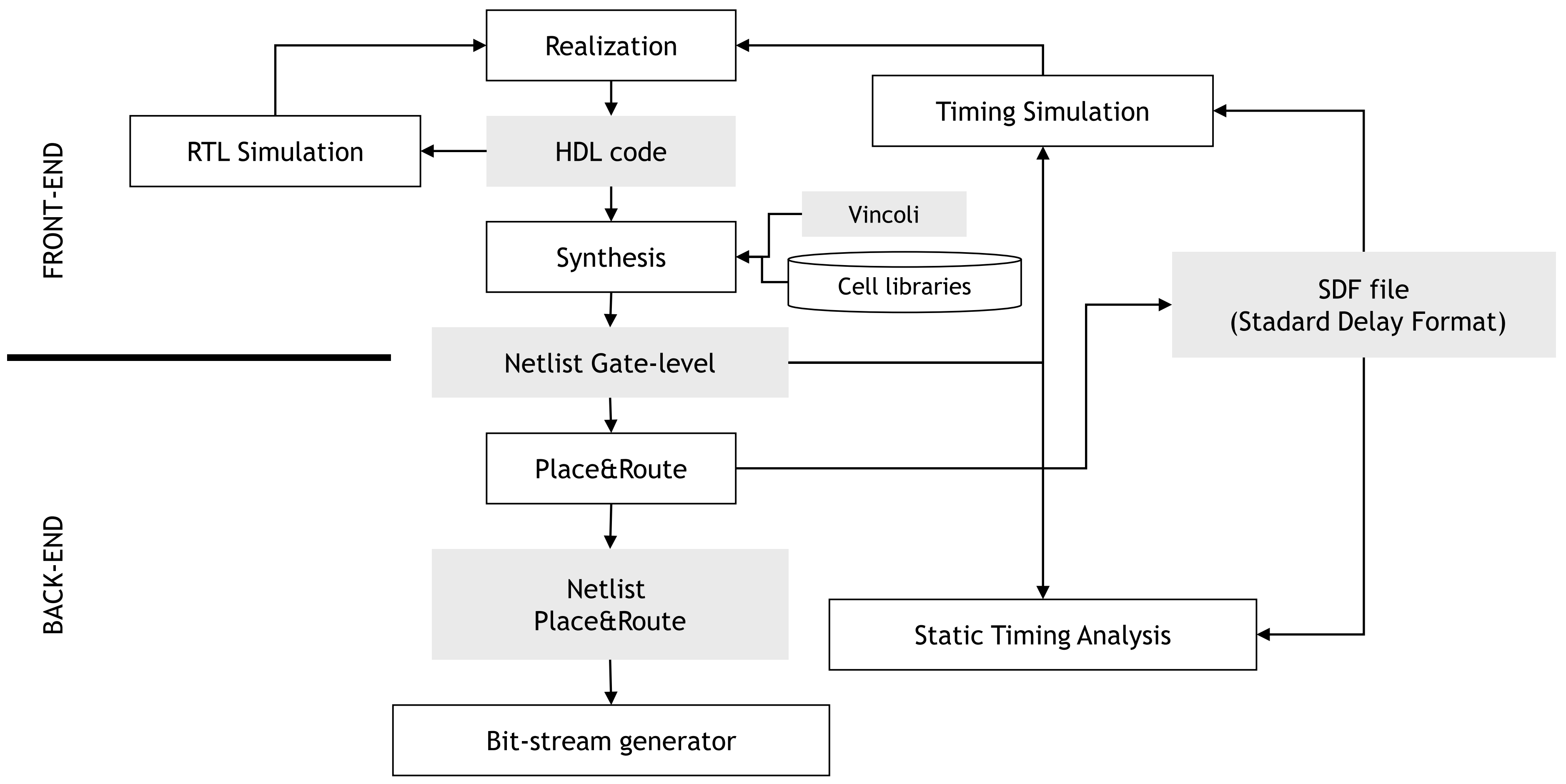
Conversion of the base components into a list of logic cells.
The *Mapping* depends on the type of device employed
This procedure optimizes the circuit to reduce the area (total number of logic cells or to improve the performance (timing)

Placement: assign the logic cells on a specific location on the FPGA

Routing: Selections of the connection wires and their binding to the cells. Configuration of the switch matrix (if present)

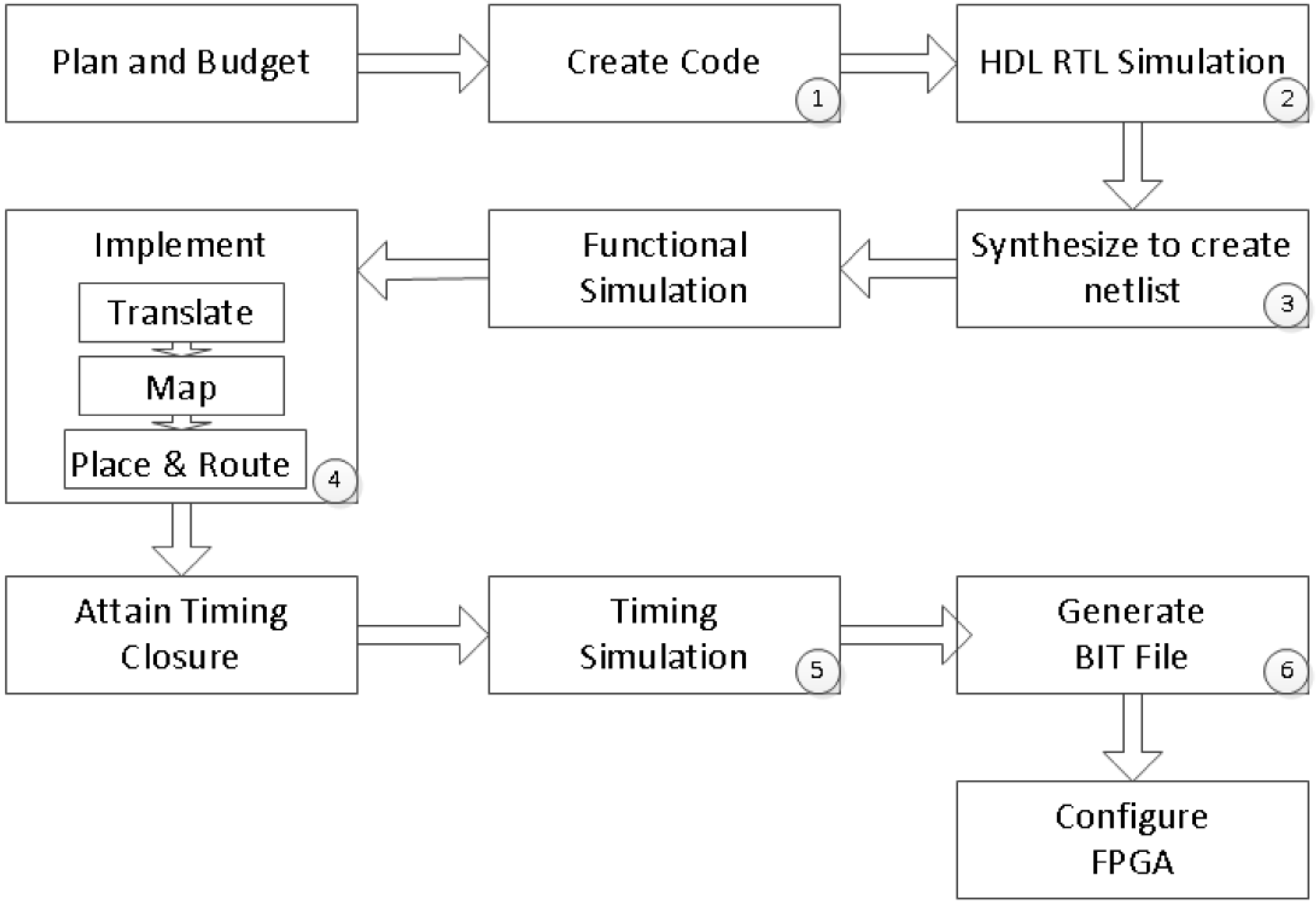


FPGA – Design flow, typical split



For FPGAs, the overall desigh flow and in particular the back-end is much simpler than the ASICs one

VIVADO integrated support of the design phases



CLB in Xilinx 7 Series - example

- LUTs in 7 series FPGAs can be configured as either a 6-input LUT with one output, or as two 5-input LUTs with separate outputs but common addresses or logic inputs
- Each 5-input LUT output can optionally be registered in a flip-flop
- Four such 6-input LUTs and their eight flip-flops as well as multiplexers and arithmetic carry logic form a slice, and two slices form a CLB
- Four flip-flops per slice (one per LUT) can optionally be configured as latches. In that case, the remaining four flip-flops in that slice must remain unused
- Approximately two-thirds of the slices are SLICEL logic slices and the rest are SLICEM, which can also use their LUTs as distributed 64-bit RAM or as 32-bit shift registers (SRL32) or as two SRL16s
- Modern synthesis tools take advantage of these highly efficient logic, arithmetic, and memory features
- “Expert” designers can also instantiate them

7 Series CLB main features

- Columnar architecture
  - Scales easily to higher densities
  - More routing between CLBs
- Only SLICEL and SLICEM
  - All slices support carry logic
- The number of CLBs and the ratio between CLBs and other device resources differentiates the 7 series families. Migration between the 7 series families does not require any design changes for the CLBs
- Device capacity is often measured in terms of logic cells, which are the logical equivalent of a classic *four-input LUT and a flip-flop*
- The 7 series FPGA CLB six-input LUT, abundant flip-flops and latches, carry logic, and the ability to create distributed RAM or shift registers in the SLICEM, increase the effective capacity
- The ratio between the number of logic cells and 6-input LUTs is 1.6:1.

Artix and Spartan, the popular ones

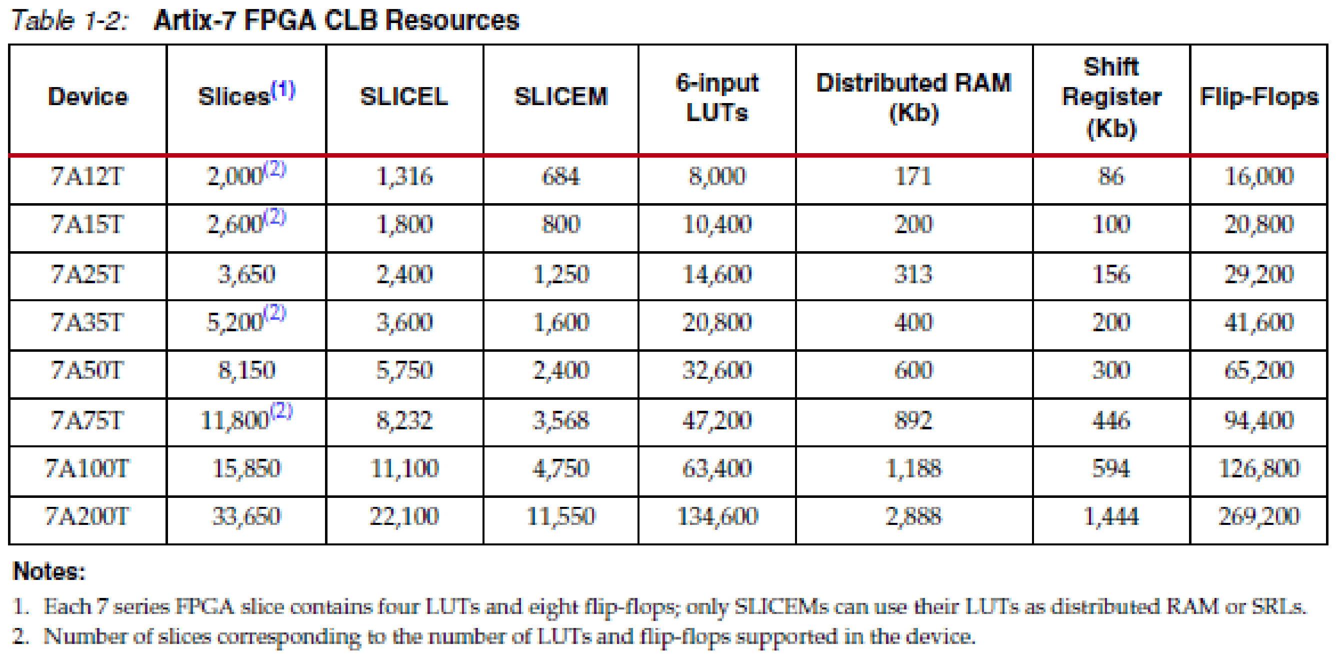


Table 1-2: **Artix-7 FPGA CLB Resources**

Device	Slices[1]	SLICEL	SLICEM	6-input LUTs	Distributed RAM (Kb)	Shift Register (Kb)	Flip-Flops
7A12T	2,000[2]	1,316	684	8,000	171	86	16,000
7A15T	2,600[2]	1,800	800	10,400	200	100	20,800
7A25T	3,650	2,400	1,250	14,600	313	156	29,200
7A35T	5,200[2]	3,600	1,600	20,800	400	200	41,600
7A50T	8,150	5,750	2,400	32,600	600	300	65,200
7A75T	11,800[2]	8,232	3,568	47,200	892	446	94,400
7A100T	15,850	11,100	4,750	63,400	1,188	594	126,800
7A200T	33,650	22,100	11,550	134,600	2,888	1,444	269,200

Notes:

1. Each 7 series FPGA slice contains four LUTs and eight flip-flops; only SLICEMs can use their LUTs as distributed RAM or SRLs.
2. Number of slices corresponding to the number of LUTs and flip-flops supported in the device.

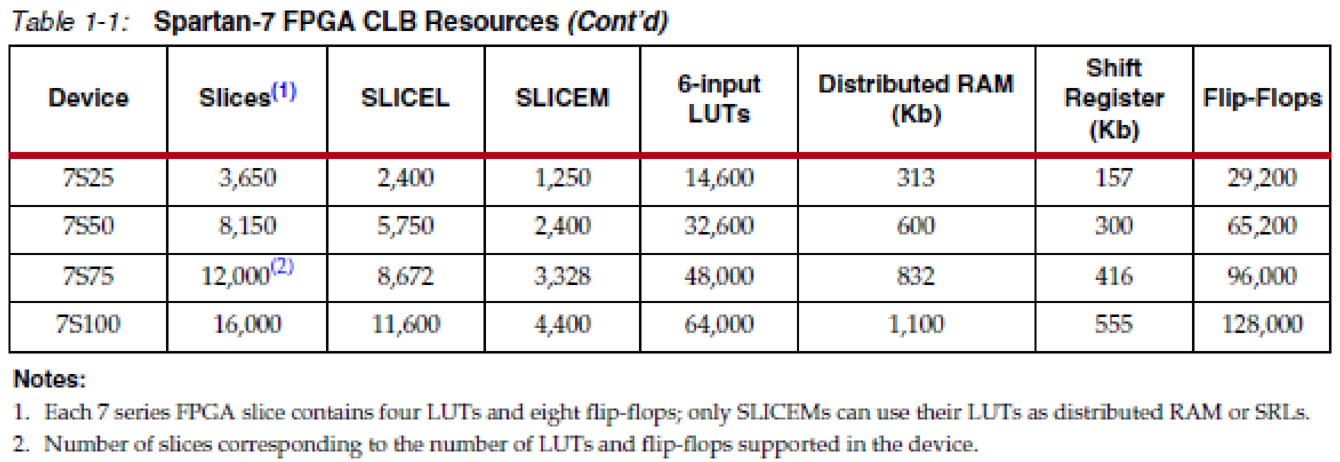


Table 1-1: **Spartan-7 FPGA CLB Resources *(Cont'd)***

Device	Slices[1]	SLICEL	SLICEM	6-input LUTs	Distributed RAM (Kb)	Shift Register (Kb)	Flip-Flops
7S25	3,650	2,400	1,250	14,600	313	157	29,200
7S50	8,150	5,750	2,400	32,600	600	300	65,200
7S75	12,000[2]	8,672	3,328	48,000	832	416	96,000
7S100	16,000	11,600	4,400	64,000	1,100	555	128,000

Notes:

1. Each 7 series FPGA slice contains four LUTs and eight flip-flops; only SLICEMs can use their LUTs as distributed RAM or SRLs.
2. Number of slices corresponding to the number of LUTs and flip-flops supported in the device.

No compromise: Virtex-7

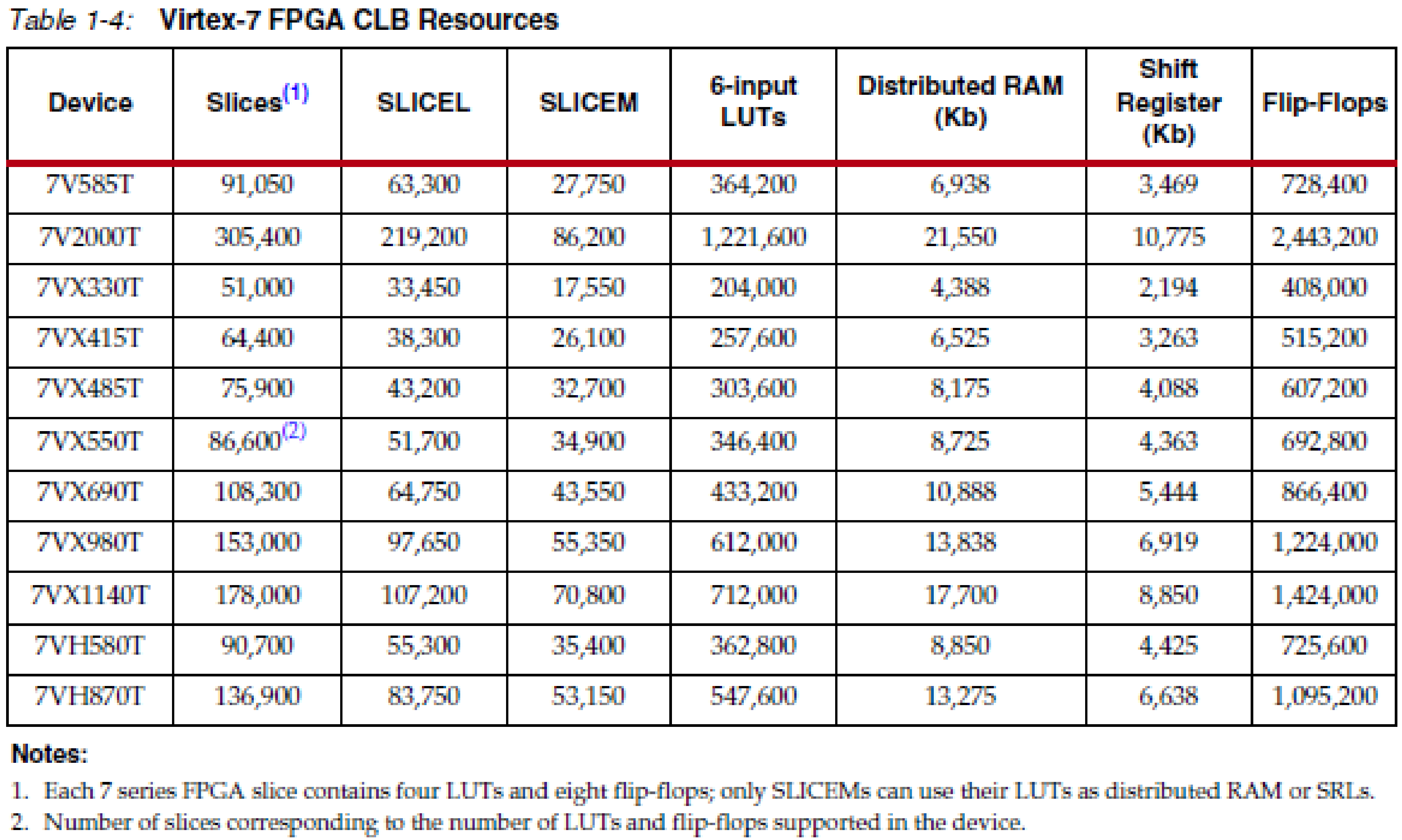


Table 1-4: **Virtex-7 FPGA CLB Resources**

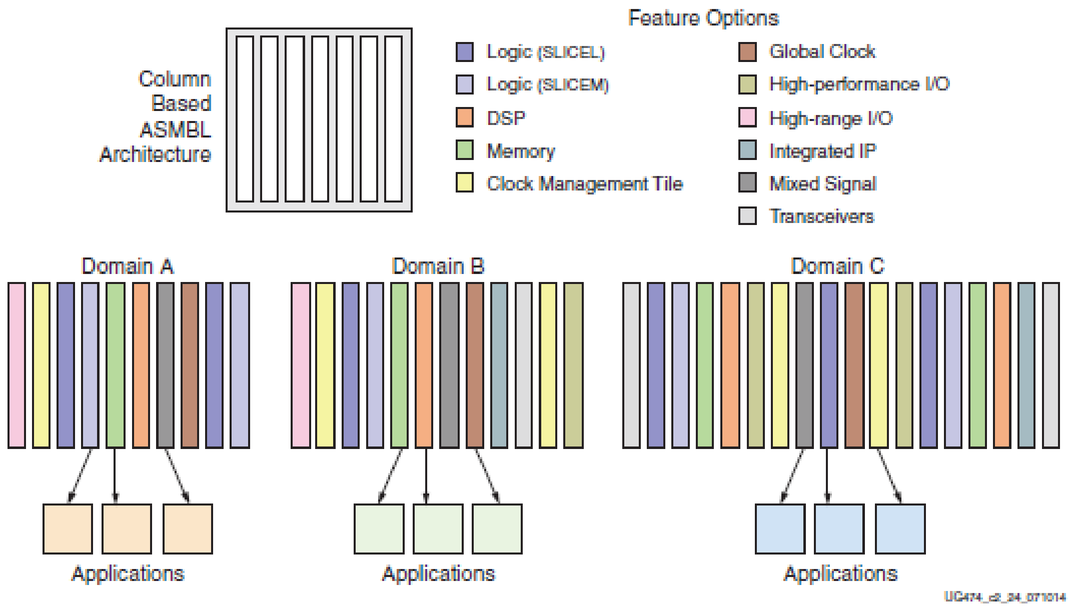
Device	Slices[1]	SLICEL	SLICEM	6-input LUTs	Distributed RAM (Kb)	Shift Register (Kb)	Flip-Flops
7V585T	91,050	63,300	27,750	364,200	6,938	3,469	728,400
7V2000T	305,400	219,200	86,200	1,221,600	21,550	10,775	2,443,200
7VX330T	51,000	33,450	17,550	204,000	4,388	2,194	408,000
7VX415T	64,400	38,300	26,100	257,600	6,525	3,263	515,200
7VX485T	75,900	43,200	32,700	303,600	8,175	4,088	607,200
7VX550T	86,600[2]	51,700	34,900	346,400	8,725	4,363	692,800
7VX690T	108,300	64,750	43,550	433,200	10,888	5,444	866,400
7VX980T	153,000	97,650	55,350	612,000	13,838	6,919	1,224,000
7VX1140T	178,000	107,200	70,800	712,000	17,700	8,850	1,424,000
7VH580T	90,700	55,300	35,400	362,800	8,850	4,425	725,600
7VH870T	136,900	83,750	53,150	547,600	13,275	6,638	1,095,200

Notes:

1. Each 7 series FPGA slice contains four LUTs and eight flip-flops; only SLICEMs can use their LUTs as distributed RAM or SRLs.
2. Number of slices corresponding to the number of LUTs and flip-flops supported in the device.

ASMBL architecture

- Xilinx created the Advanced Silicon Modular Block (ASMBL) architecture to enable FPGA platforms with varying feature mixes optimized for different application domains
- Through this innovation Xilinx offers a greater selection of devices, enabling customers to select the FPGA with the right mix of features and capabilities for their specific design



Simplified view of Virtex 7 series FPGA slice cell

- Each SLICEM LUT can be configured as a look-up table, distributed RAM, or a shift register. Every slice contains:
- Four logic-function generators (or look-up tables)
- Eight storage elements
- Wide-function multiplexers
- Carry logic

•These elements are used by all slices to provide logic, arithmetic, and ROM functions. In addition, some slices support two additional functions: storing data using distributed RAM and shifting data with 32-bit registers

- Slices that support these additional functions are called SLICEM; others are called SLICEL (Figure 2-4)
- SLICEM (Figure 2-3) represents a superset of elements and connections found in all slices
- Each CLB can contain two SLICEL or a SLICEL and a SLICEM

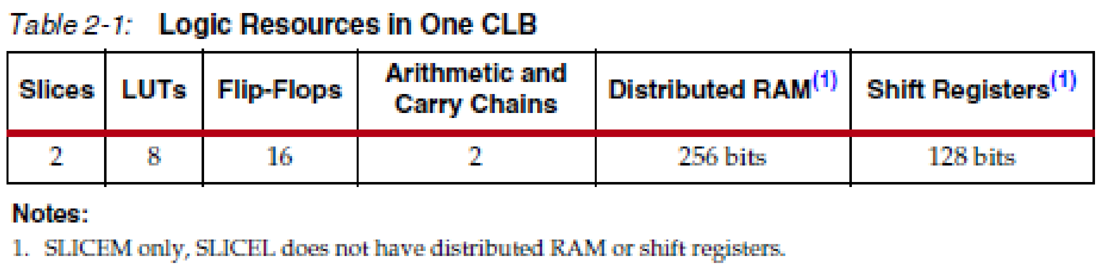


Table 2-1: **Logic Resources in One CLB**

Slices	LUTs	Flip-Flops	Arithmetic and Carry Chains	Distributed RAM(1)	Shift Registers(1)
2	8	16	2	256 bits	128 bits

Notes:
1. SLICEM only, SLICEL does not have distributed RAM or shift registers.

“Simplified” Virtex 7 series FPGA slice cells

Relax!... tools will do the configuration task for you

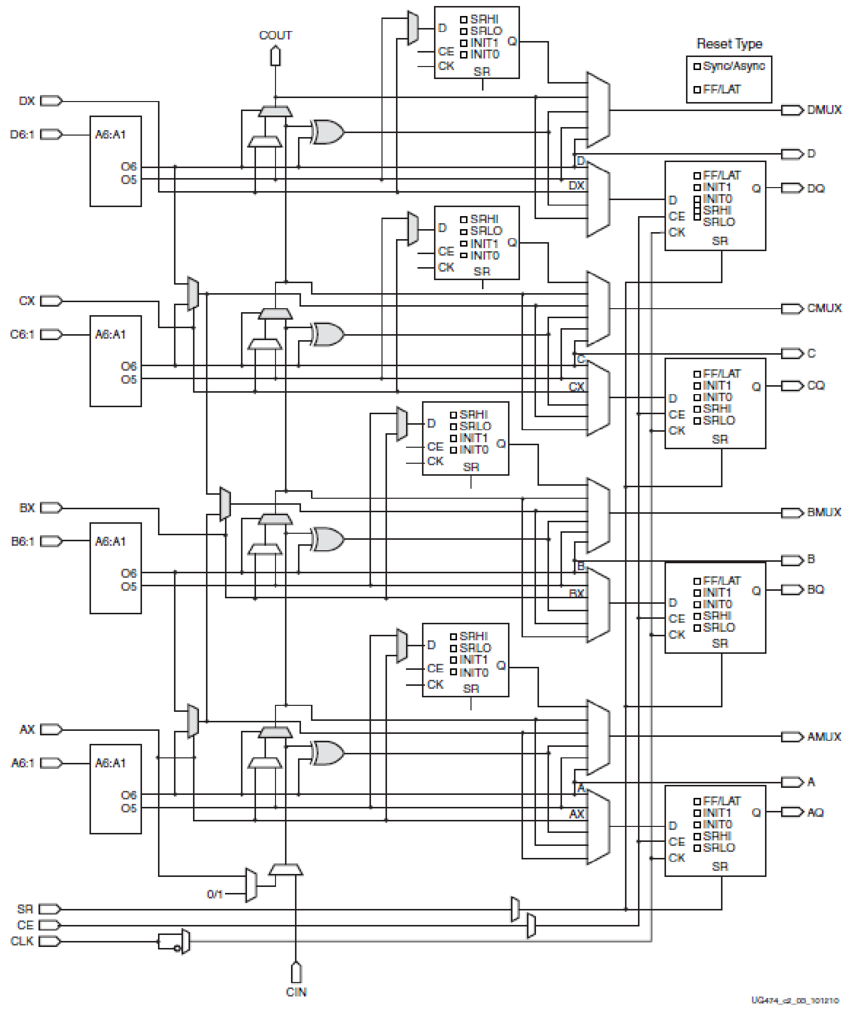


Figure 2-4: **Diagram of SLICEL**

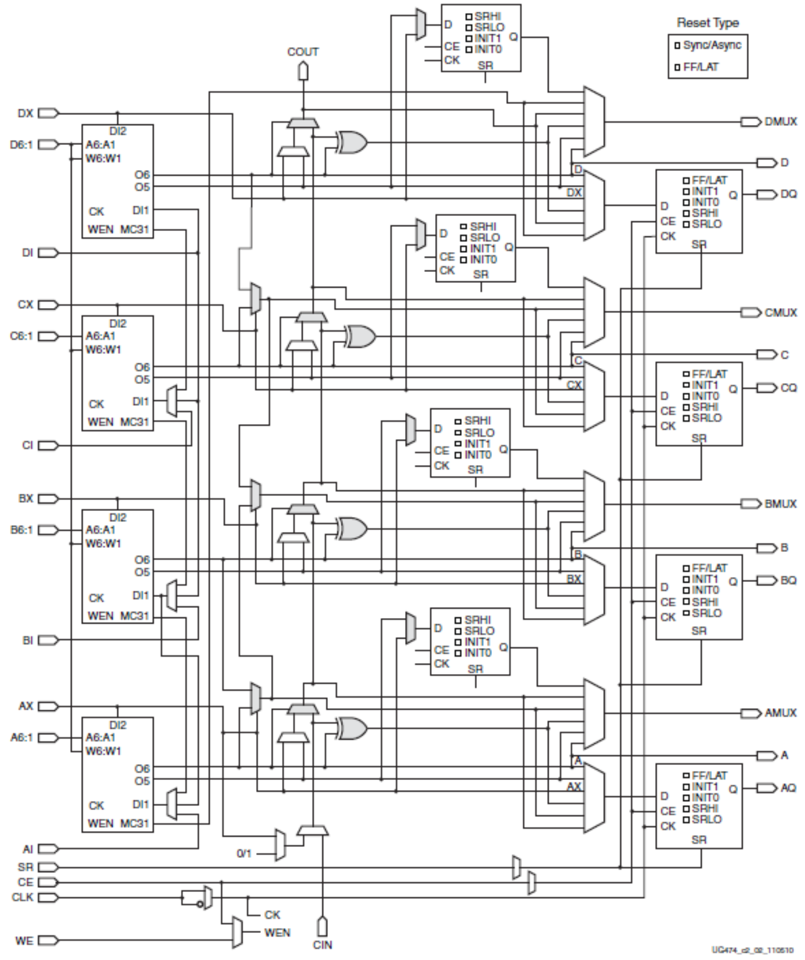


Figure 2-3: **Diagram of SLICEM**