


# Introduction to Programmable Logic Technologies

# Evolution of Implementation Technologies

- Logic gates (1950s-60s)
- Regular structures for two-level logic (1960s-70s)
  - muxes and decoders, PLAs
- Programmable gate arrays (1980s-90s)
  - PLDs, complex PLDs
- Programmable gate arrays (1980s-90s)
  - Densities high enough to permit entirely new class of application, e.g. prototyping, emulation, acceleration



**trend toward  
higher levels  
of integration**

# ASIC vs. FPGA

## ASIC

Application Specific  
Integrated Circuit

- designed all the way from behavioral description to **physical layout**
- designs must be sent for expensive and time consuming **fabrication** in semiconductor foundry

## FPGA

Field Programmable  
Gate Array

- no physical layout design; design ends with a **bitstream** used to configure a device
- bought **off the shelf** and reconfigured by designers themselves

# Which way to go?

## ASICs

High performance

Low power

Low cost in  
high volumes

## FPGAs

Off-the-shelf

Low development cost

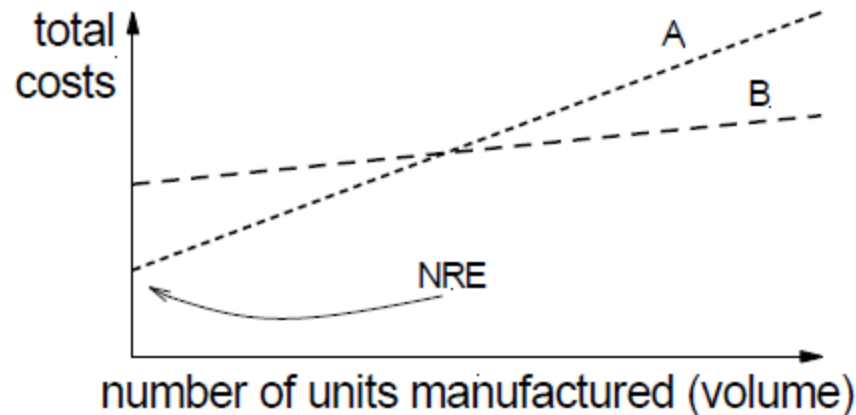
Short time to market

Reconfigurability



# Why FPGAs?

- Custom ICs sometimes designed to replace large amount of glue logic:
  - Reduced system complexity and manufacturing cost, improved performance.
  - However, custom ICs are very expensive to develop, and delay introduction of product to market (time to market) because of increased design time.
- Note: need to worry about two kinds of costs:
  - Cost of development, sometimes called non-recurring engineering (NRE)
  - Cost of manufacture
    - A tradeoff usually exist between NRE cost and manufacturing costs



# Why FPGAs?

- Custom IC approach viable for products that are...
  - Very high volume (where NRE could be amortized),
  - Not time-to-market sensitive.
- FPGAs introduced as an alternative to custom ICs for implementing glue logic:
  - Improved density relative to discrete SSI/MSI components (within around 10x of custom ICs)
  - With the aid of computer aided design (CAD) tools circuits could be implemented in a short amount of time (no physical layout process, no mask making, no IC manufacturing), relative to ASICs.
    - Lowers NREs
    - Shortens TTM
- Because of Moore's law the density (gates/areas) of FPGAs continued to grow through the 80's and 90's to point where major data processing functions can be implemented on a single FPGA.

# Applications of FPGAs

- Implementation of random logic
  - Easier changes at system-level (one device is modified)
  - Can eliminate need for full-custom chips
- Prototyping
  - Ensemble of gate arrays used to emulate a circuit to be manufactured
  - Get more/better/faster debugging done than possible with simulation
- Reconfigurable hardware
  - One hardware block used to implement more than one function
  - Functions must be mutually-exclusive in time
  - Can greatly reduce cost while enhancing flexibility
  - RAM-based only option
- Special-purpose computation engines
  - Hardware dedicated to solving one problem (or class of problems)
  - Accelerators attached to general-purpose computers

# Major FPGA Vendors

## **SRAM-based FPGAs**

- Xilinx, Inc.
  - Altera Corp.
  - Atmel
  - Lattice Semiconductor
- Share about 90% of the market

## **Flash & antifuse FPGAs**

- Actel Corp.
- Quick Logic Corp.



- **Old families**

- XC3000, XC4000, XC5200
- Old 0.5 $\mu$ m, 0.35 $\mu$ m and 0.25 $\mu$ m technology. Not recommended for modern designs.

- **High-performance families**

- Virtex (220 nm)
- Virtex-E, Virtex-EM (180 nm)
- Virtex-II, Virtex-II PRO (130 nm)
- Virtex-4 (90 nm)
- Virtex-5 (65 nm)
- Virtex-6



- **Low Cost Family**

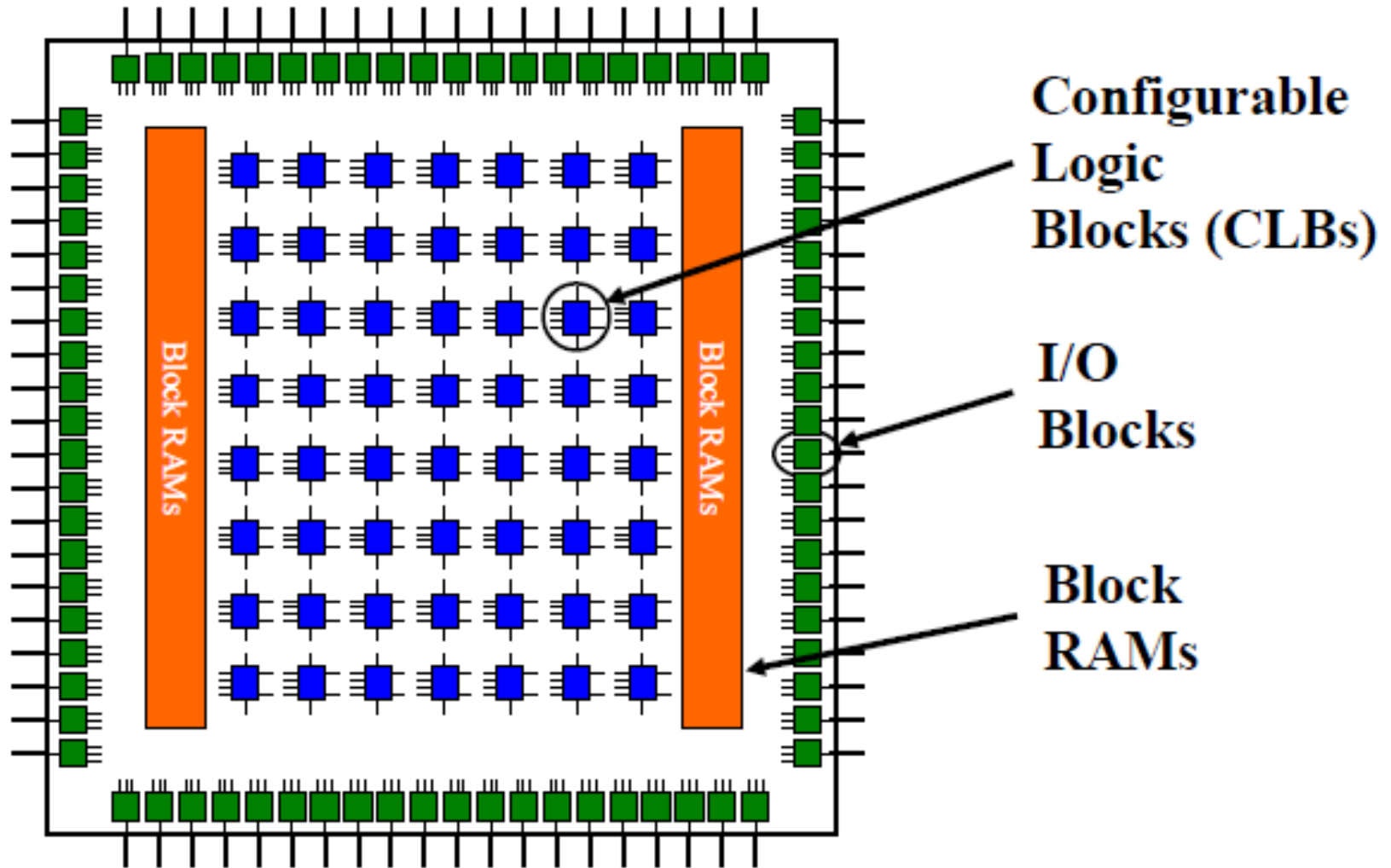
- Spartan/XL – derived from XC4000
- Spartan-II – derived from Virtex
- Spartan-II E – derived from Virtex-E
- Spartan-3 (90 nm)
- Spartan-3E (90 nm) – logic optimized
- Spartan-3A (90 nm) – I/O optimized
- Spartan-3AN (90 nm) – non-volatile
- Spartan-3A DSP (90 nm) – DSP optimized
- Spartan-6



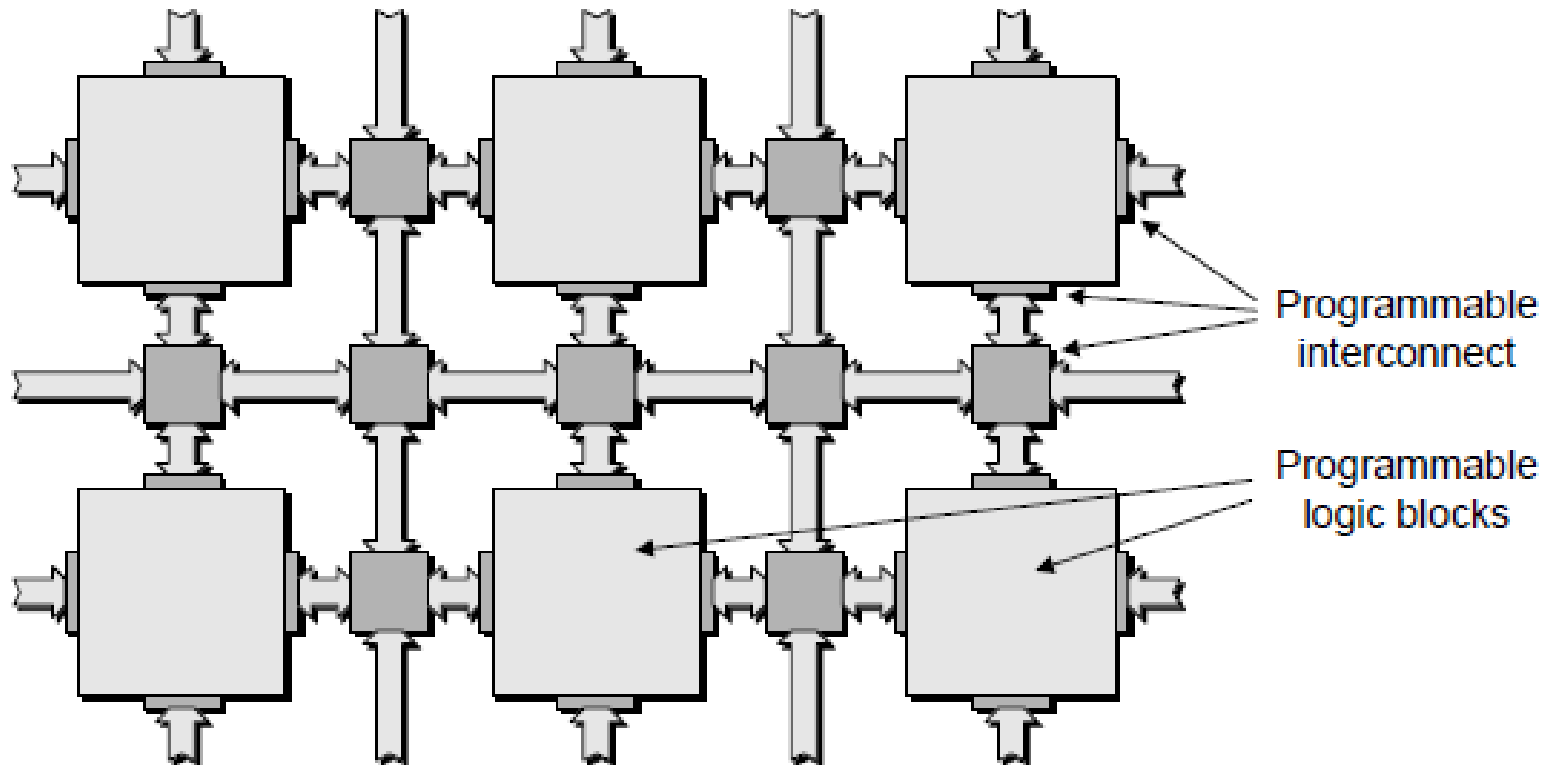
- High & Medium Density FPGAs
  - Stratix™ II, Stratix, APEX™ II, APEX 20K, & FLEX® 10K
- Low-Cost FPGAs
  - Cyclone™ & ACEX® 1K
- FPGAs with Clock Data Recovery
  - Stratix GX & Mercury™
- CPLDs
  - MAX® 7000 & MAX 3000
- Embedded Processor Solutions
  - Nios™, Excalibur™
- Configuration Devices
  - EPC



# What is an FPGA ?

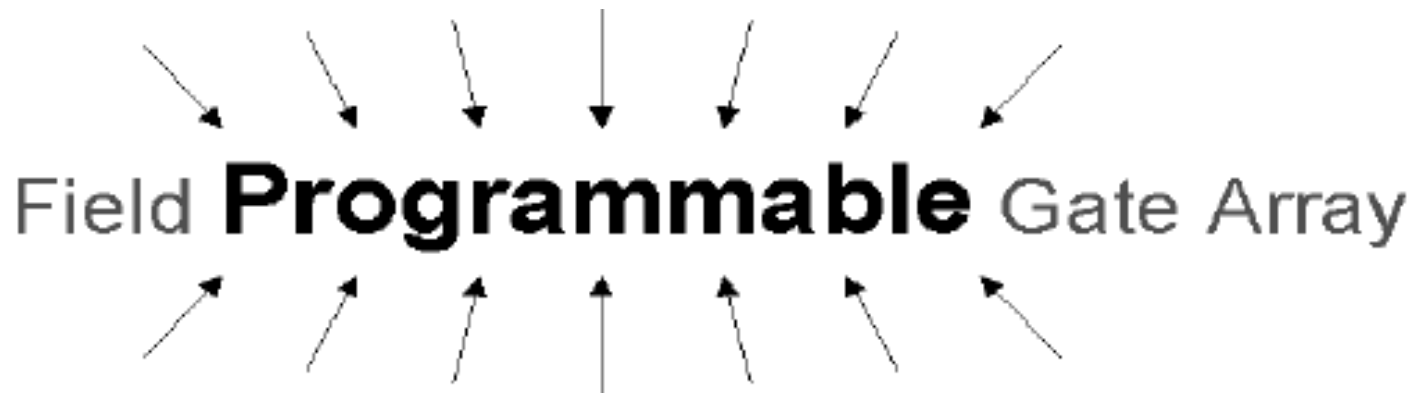


# What is an FPGA?



# The key thing about FPGAs

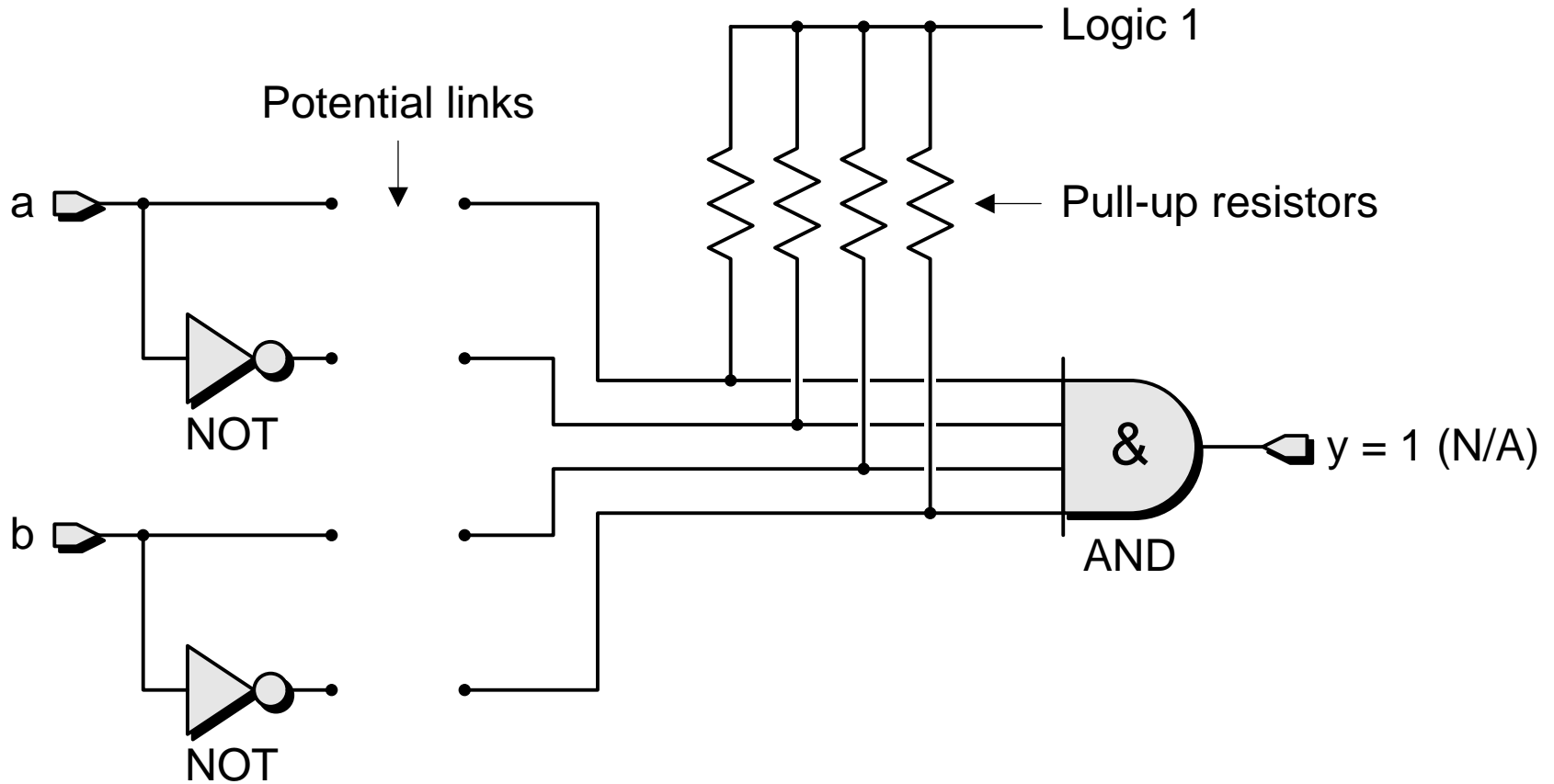
- The thing that really distinguishes an FPGA from an ASIC is
  - The crucial aspect that resides at the core of their reason for being is
    - Embodied in their name



Field **Programmable** Gate Array

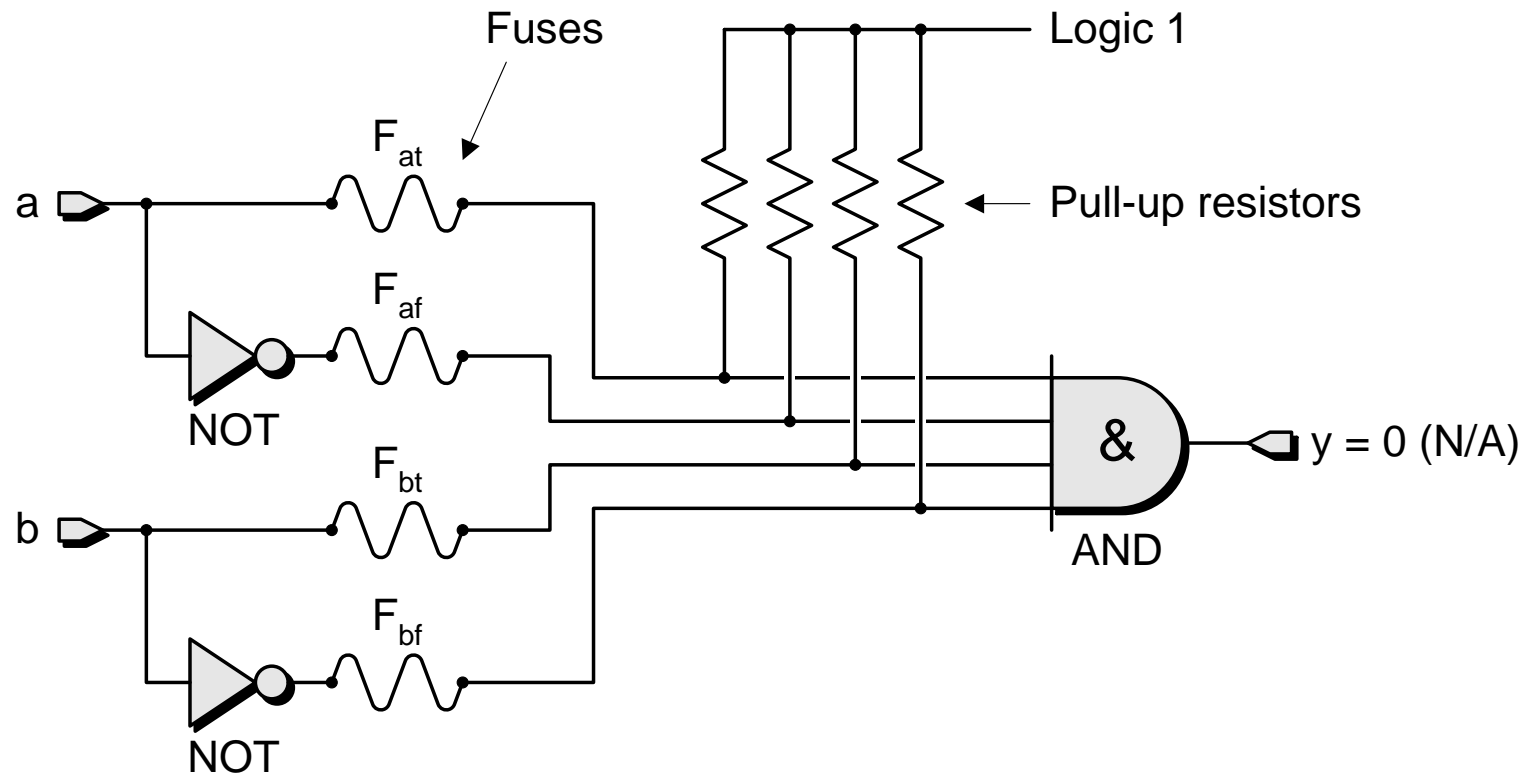
A diagram illustrating the structure of the name 'Field Programmable Gate Array'. The word 'Programmable' is written in a large, bold, black font. The words 'Field' and 'Gate Array' are written in a smaller, regular, grey font on either side of 'Programmable'. There are seven arrows pointing towards the word 'Programmable': three from above and four from below. The arrows from above are positioned over the 'P', 'r', 'o', 'g', 'r', 'a', and 'm' of 'Programmable'. The arrows from below are positioned under the 'P', 'r', 'o', 'g', 'r', 'a', and 'm' of 'Programmable'.

# A Simple Programmable Function

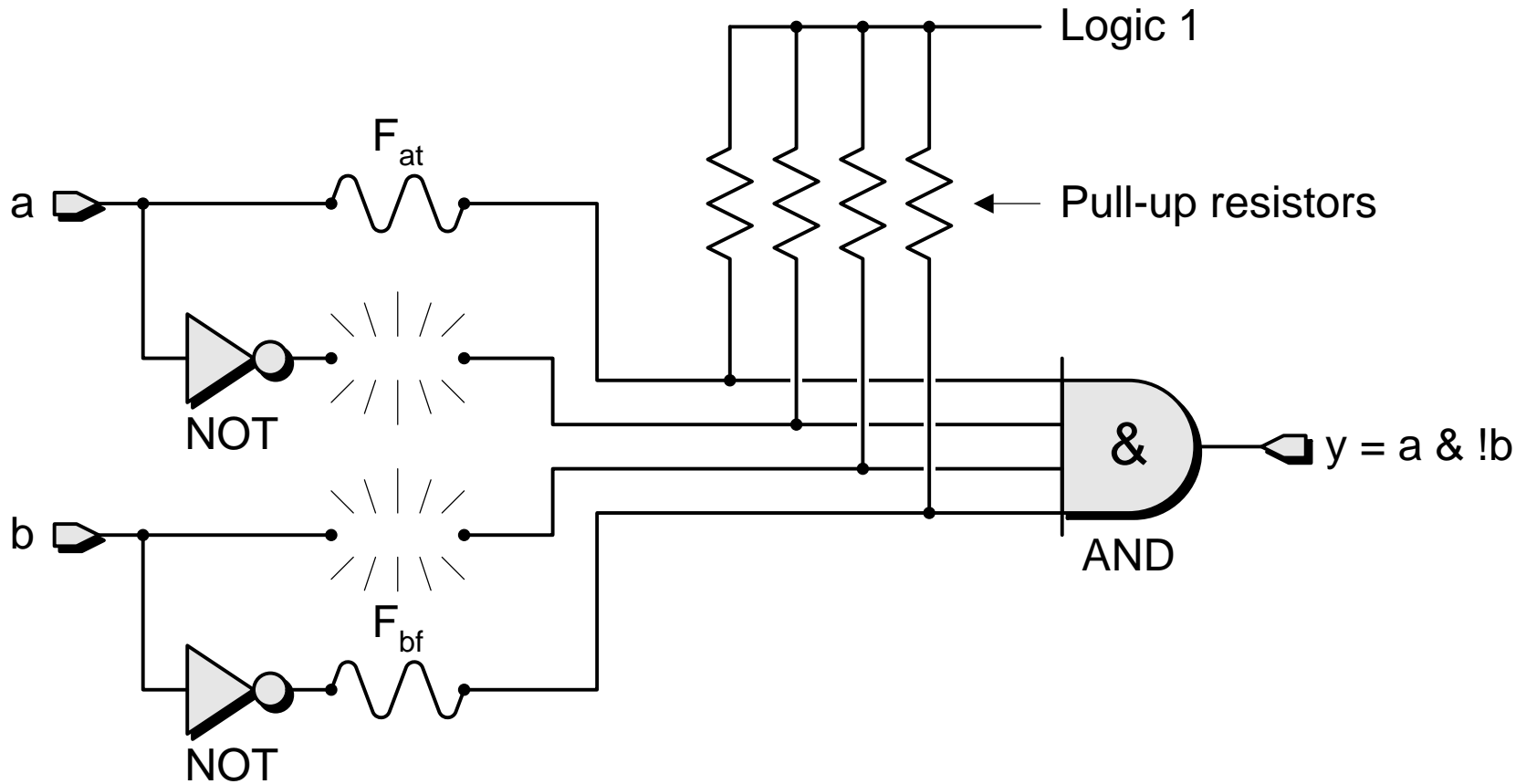


# Fusible link Technologies

- The device is manufactured with all of the links in place, where each link is referred to as a *fuse*.
- Devices based on fusible-link technologies are said to be one-time programmable, or OTP, because once a fuse has been blown up, it can not be replaced and there's no going back.



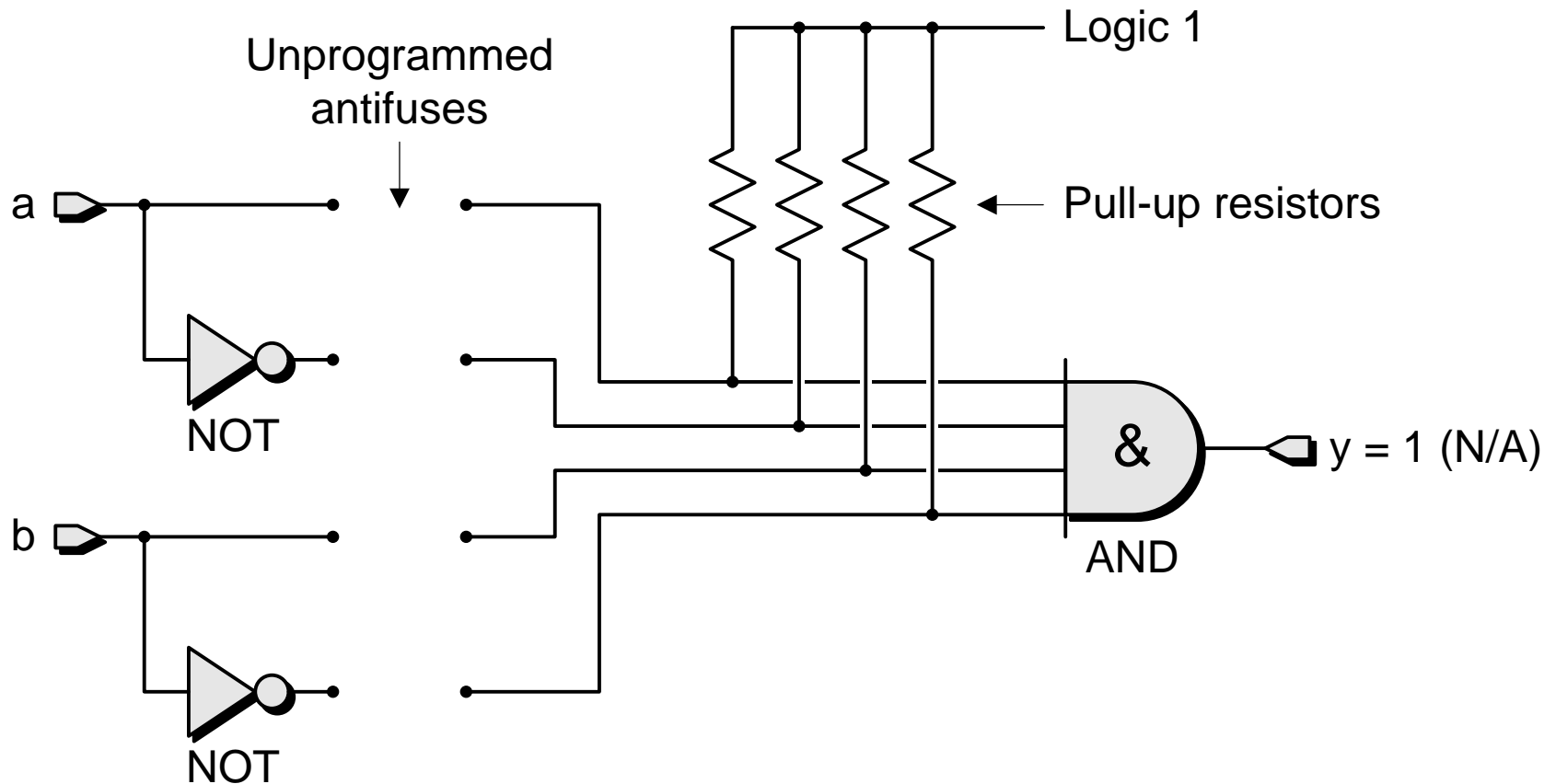
# Programmed Fusible links



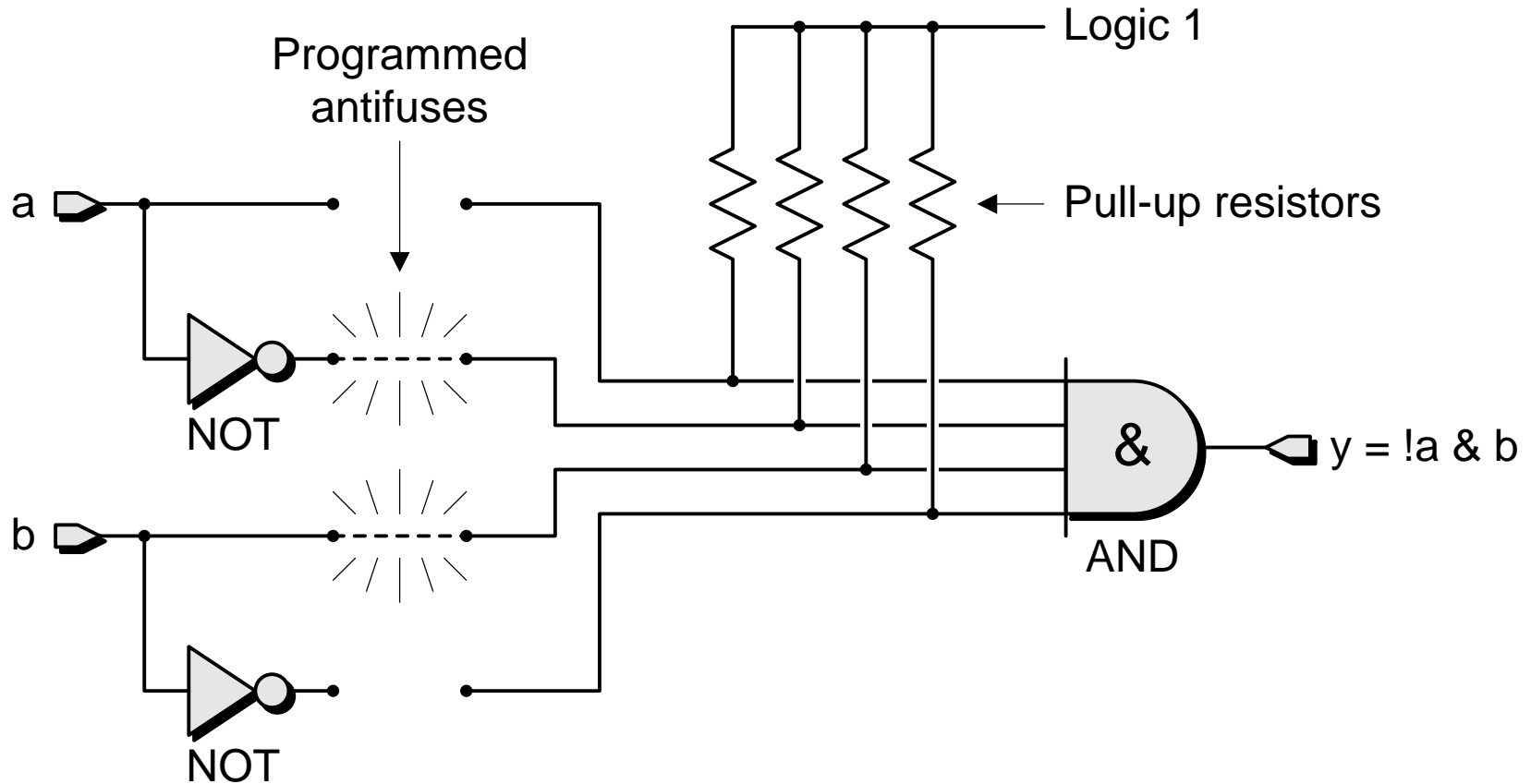


# Antifuse Technologies

- In unprogrammed state, an antifuse such a high resistance that may be considered an open circuit.



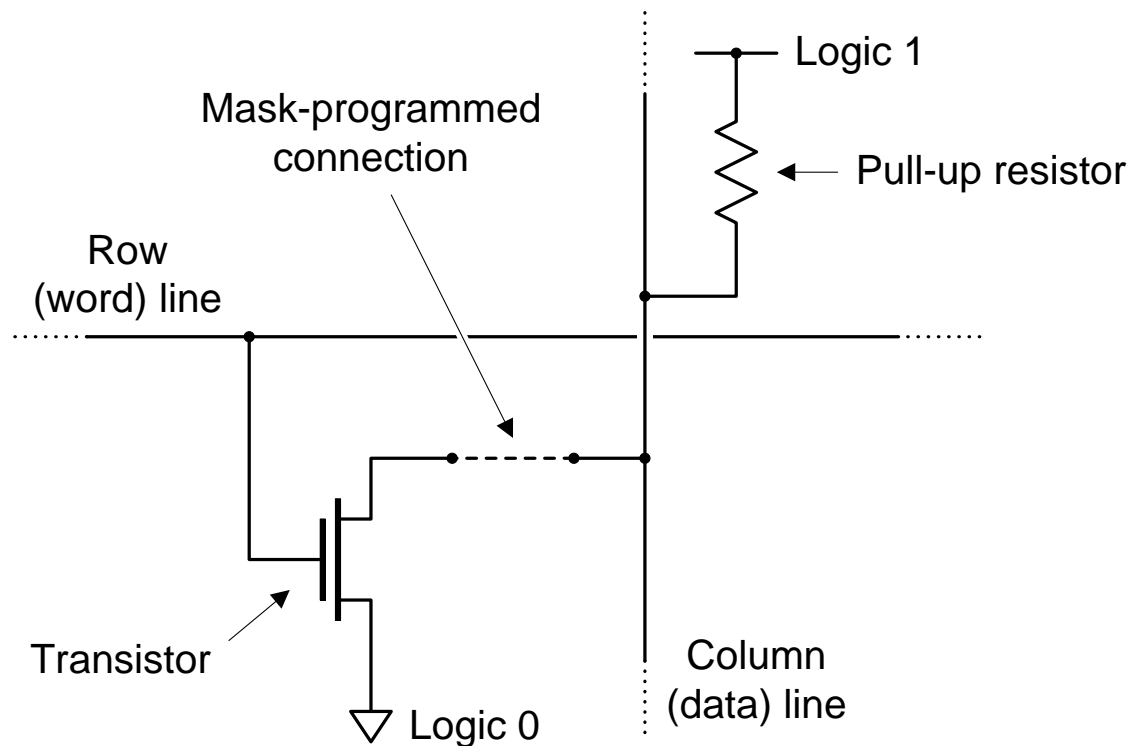
# Programmed antifuse links



# Mask-programmed devices

- Electronic systems in general – and computers in particular – make use of two major classes of memory devices:
  - Read-Only Memory (ROM)
  - Random-Access Memory (RAM)
- ROMs are said to be *nonvolatile* because their data remains when power is removed from the system.
  - Other components in the system can read data from ROM devices, but they can not write data into them.
- By comparison, data can be both written into and read out of RAM devices, which are said to be *volatile* because any data they contain is lost when the system is powered down.

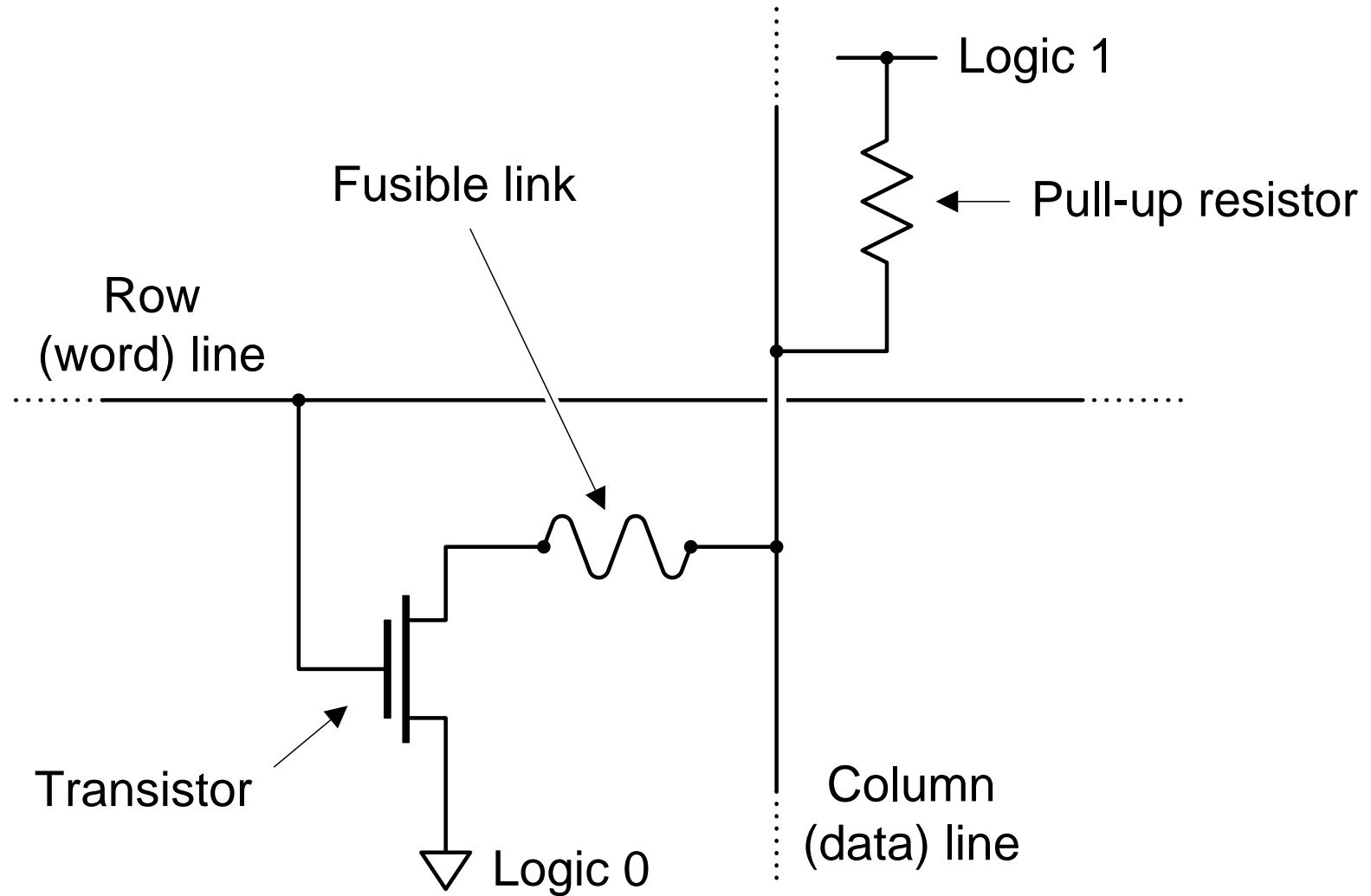
- Basic ROMs are also said to be *mask-programmable*.
- Because any data they contain is hard-coded into them during their construction by means of the photo-masks.
  - Photo masks are used to create the transistors and the metal tracks (referred to as the *metallization layers*) connecting them together on the silicon chip.



# PROMs

- The problem with mask-programmed devices is that creating them is a very expensive pastime unless you intend to produce them in extremely large quantities.
- Such components are of little use in a development environment in which you often need to modify their contents.
- For this reason, the first programmable read-only memory (PROM) devices were developed at Harris Semiconductor in 1970.
- These devices were created using a nichrome-based fusible-link technology.

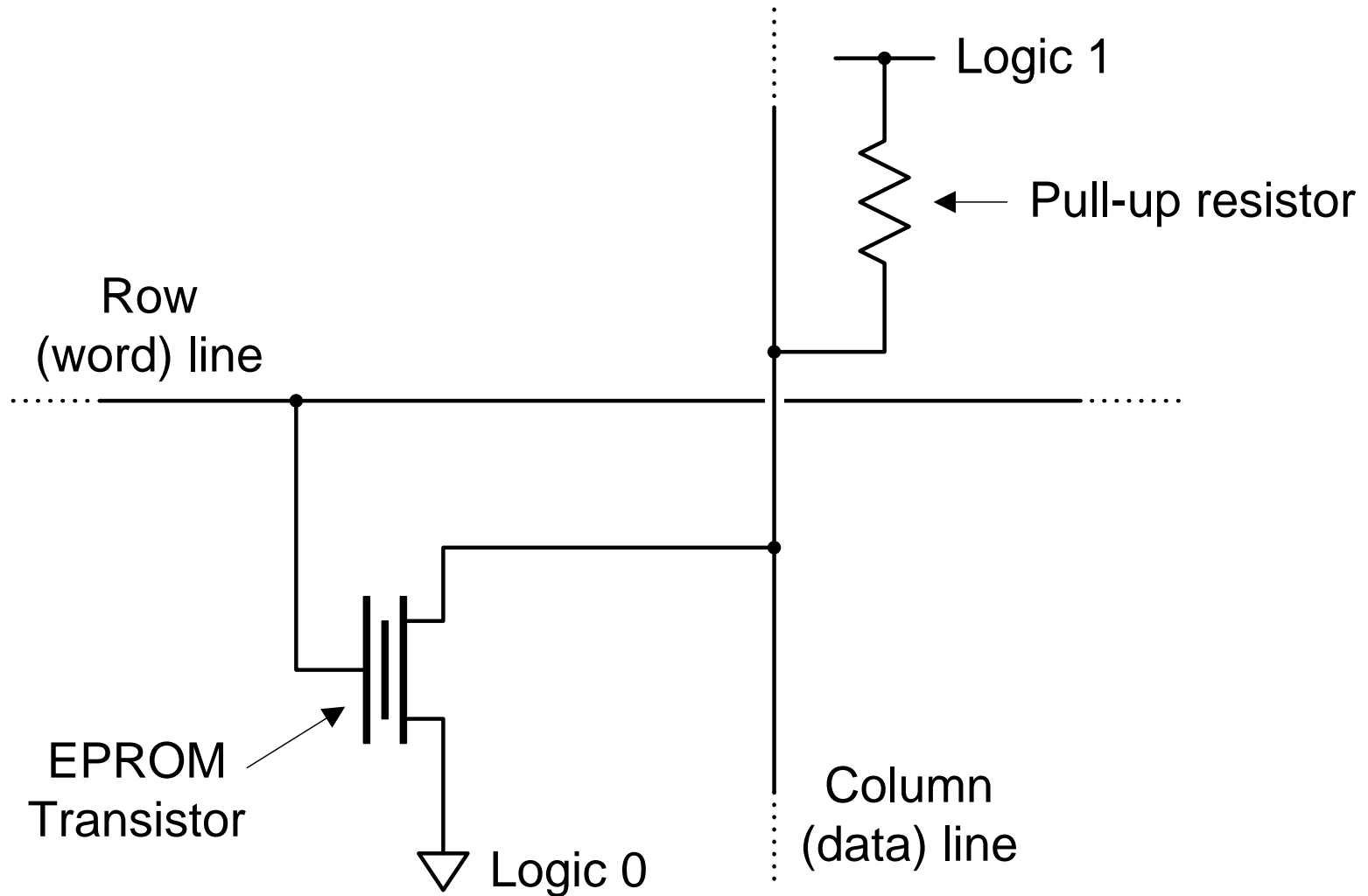
# A transistor-and-fusible-link-based PROM cell



# EPR0M-based Technologies

- Devices based on fusible-link or antifuse technologies can only be programmed a single time – Once you've blown a fuse, it's too late to change your mind.
- For this reason, people started to think that it would be nice if there were some way to create devices that could be programmed, erased, and reprogrammed with new data.
- One alternative is a technology known as erasable programmable read-only memory (EPR0M), with the first such device – the 1702 – being introduced by Intel in 1971.

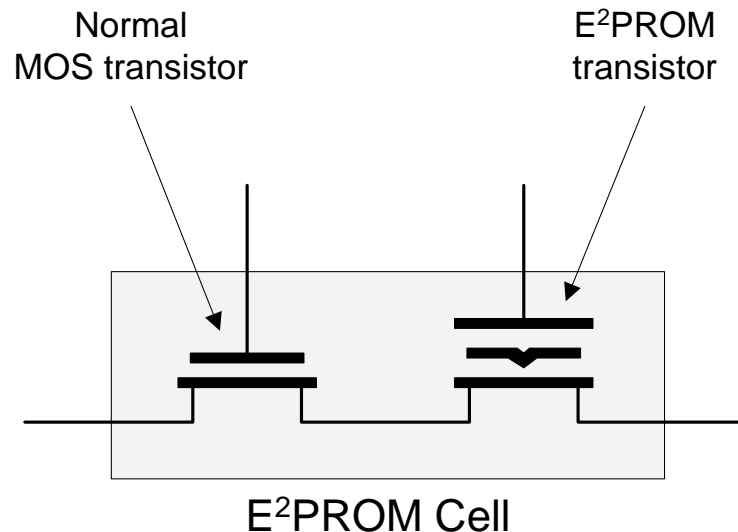
# An EPROM transistor-based memory cell





# EEPROM-based technologies

- An EEPROM cell is approximately 2.5 times larger than an equivalent EPROM cell because it comprises two transistors and the space between them.
- The EEPROM transistor is similar to that of an EPROM transistor in that it contains a floating gate, but the insulating oxide layers surrounding this gate are very much thinner.
- The second transistor can be used to erase the cell electrically.



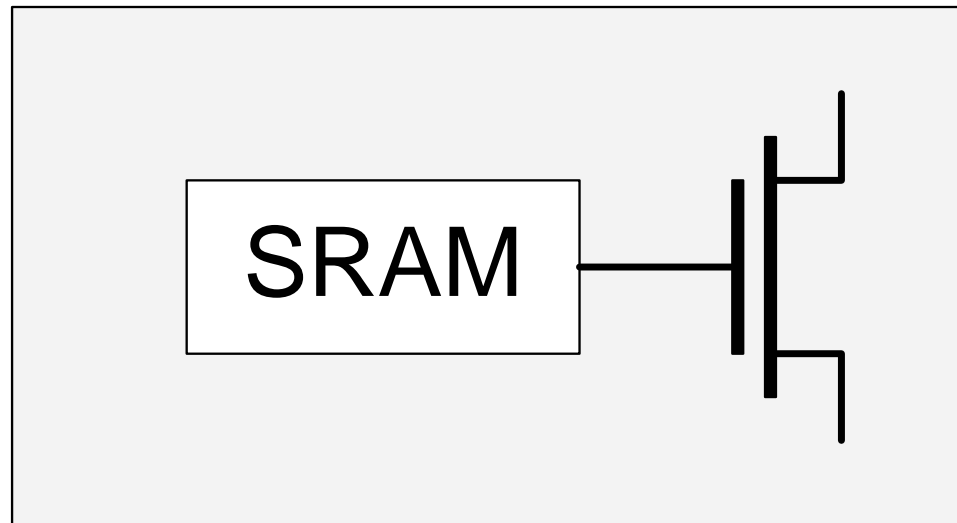
# Flash-based Technologies

- A development known as FLASH can trace its ancestry to both the EPROM and EEPROM technologies.
- The name “FLASH” was originally coined to reflect this technology’s rapid erasure times compared to EPROM.
- Components based on FLASH can employ a variety of architectures.
- Some have a single floating gate transistor cell with the same area as an EPROM cell.
- Other architectures feature a two-transistor cell similar to that of an EEPROM cell, thereby allowing them to be erased and reprogrammed on a word-by-word basis.



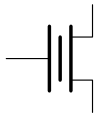
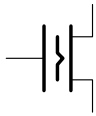
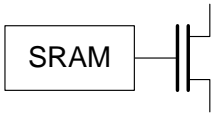
# SRAM-based technologies

- There are two main versions of semiconductor RAM devices: Dynamic RAM (DRAM) and static RAM (SRAM).
- In the case of DRAMs, each cell is formed from a transistor-capacitor pair that consumes very little silicon real state.
  - The “dynamic” qualifier is used because the capacitor loses its charge over time, so each cell must be periodically recharged if it is to retain its data.
  - This operation is known as refreshing.
  - However, DRAM technology is of little interest with regard to programmable logic.

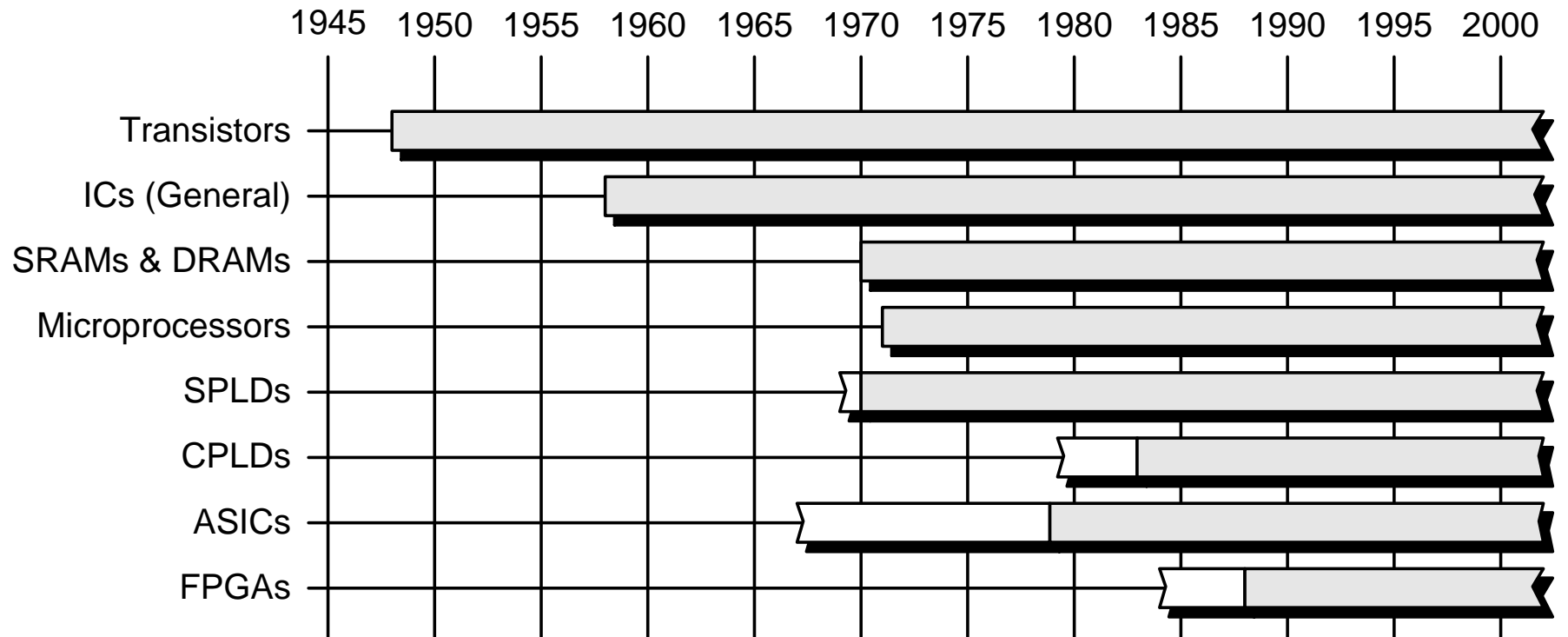
- By comparison, the ‘static’ qualifier associated with SRAM is employed because – once a value has been loaded into an SRAM cell – it will remain unchanged unless it is specifically altered or until power is removed from the system.



# Summary of Programming Technologies

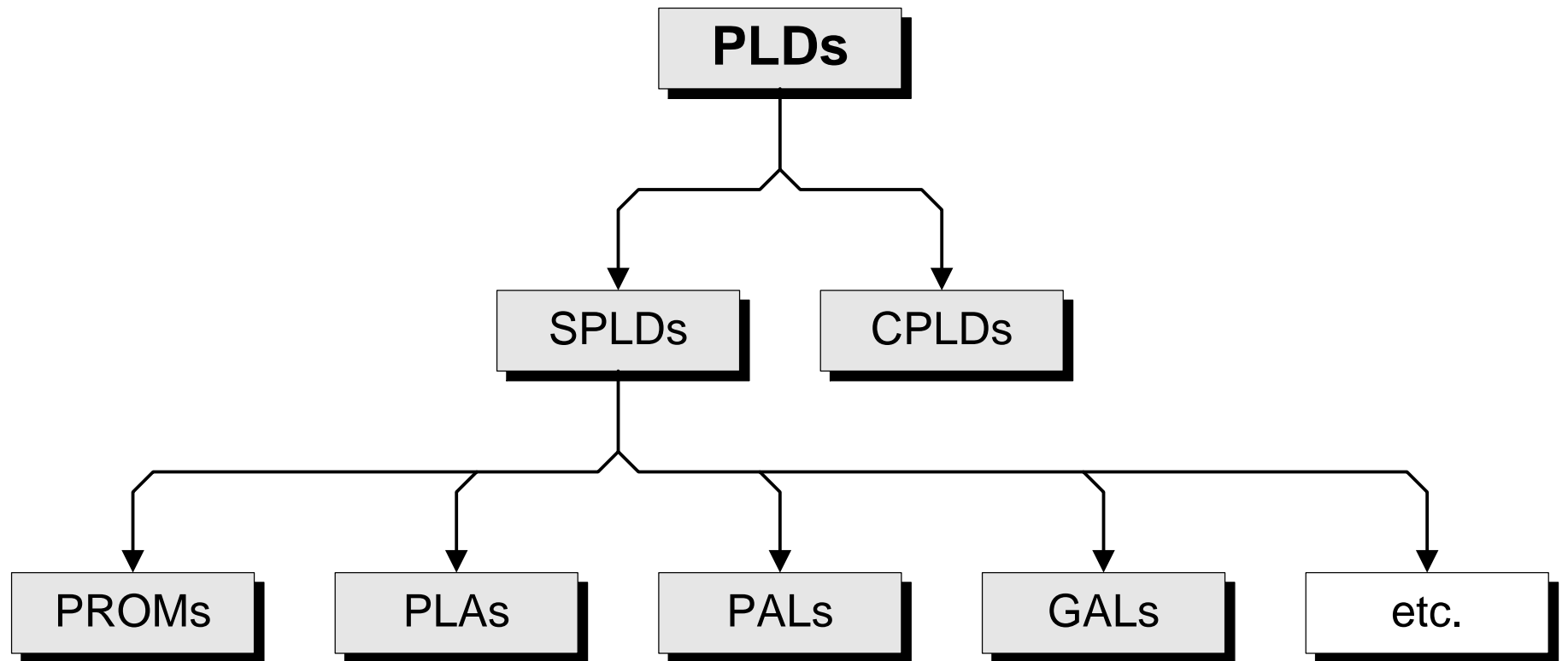
Technology	Symbol	Predominantly associated with ...
Fusible-link		SPLDs
Antifuse		FPGAs
EPROM		SPLDs and CPLDs
E <sup>2</sup> PROM/ FLASH		SPLDs and CPLDs (some FPGAs)
SRAM		FPGAs (some CPLDs)

# Related Technologies



# SPLDs and CPLDs

- The first programmable ICs were generically referred to as *programmable logic devices (PLDs)*.
- The original components, which started arriving on the scene in 1970's in the form of PROMs, were rather simple, but everyone was too polite to mention it.
- It was only toward the end of the 1970s that significantly more complex versions became available.
- In order to distinguish them from their less-sophisticated ancestors, which still find use to this day, these new devices were referred to as *complex PLDs (CPLDs)*.

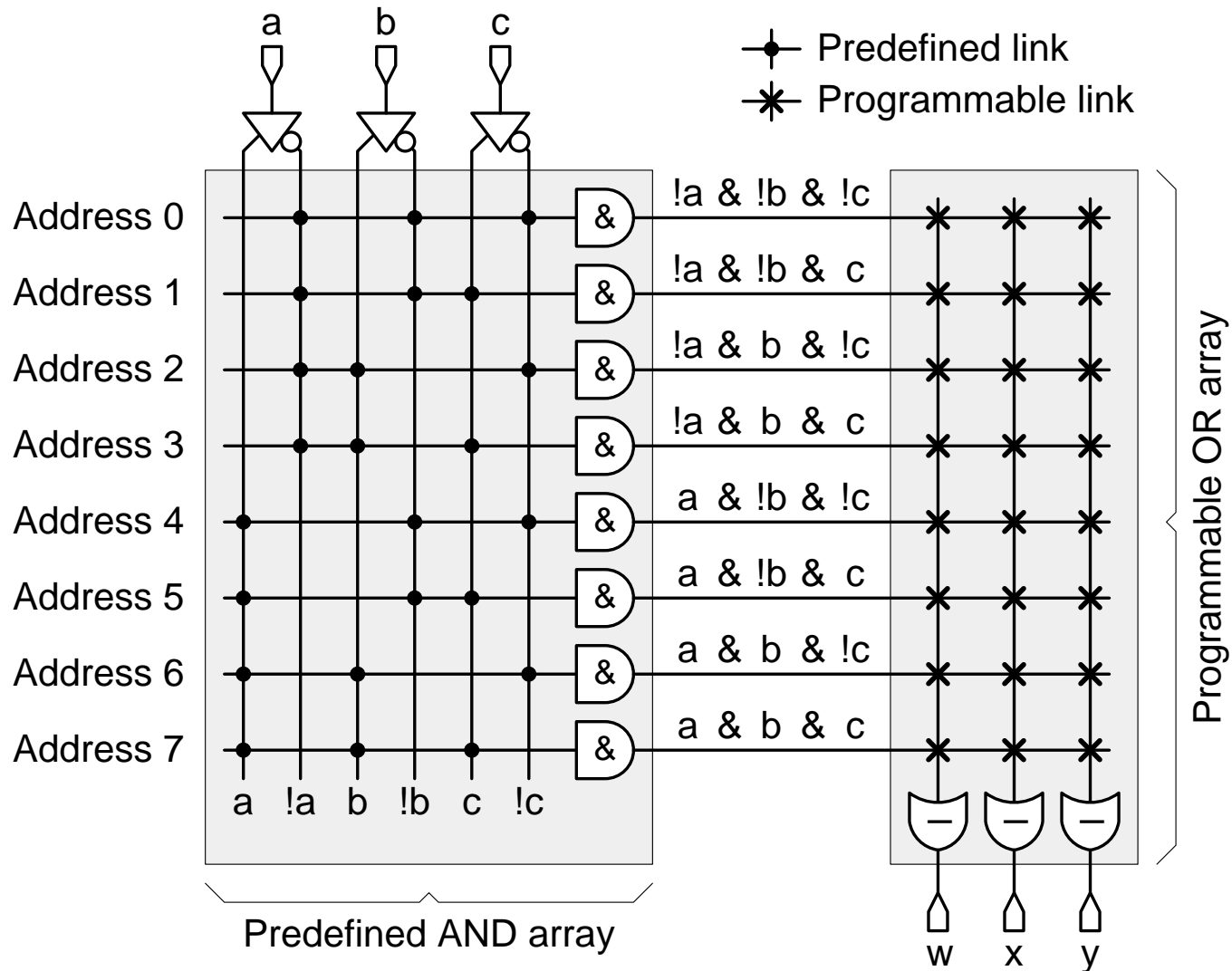




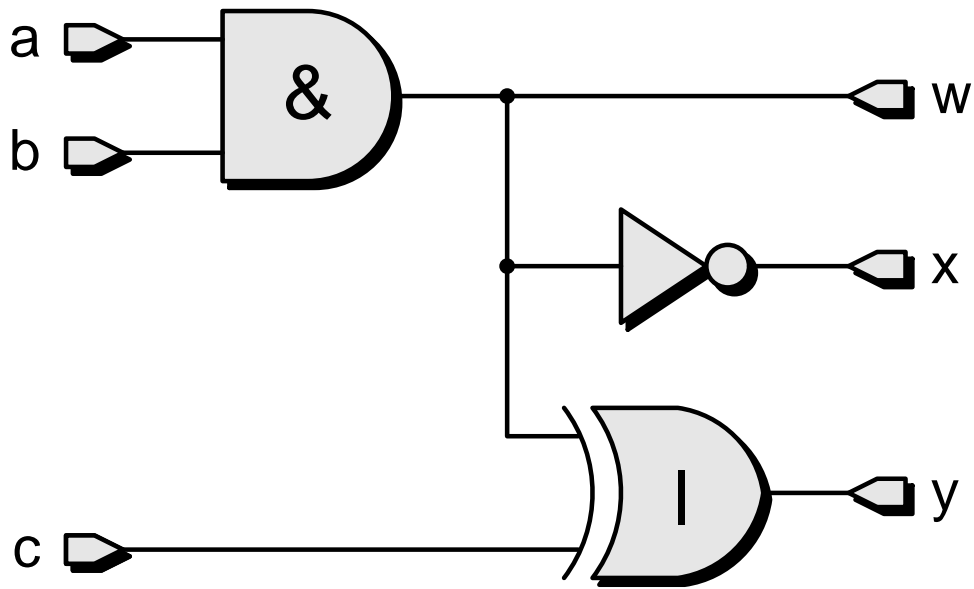
# PROMs

- The first of the simple PLDs were PROMs, which appeared on the scene in 1970.
- One way to visualize how these devices perform their magic is to consider them as consisting of a fixed array of AND functions driving a programmable array OR functions.
- The programmable link in the OR array can be implemented as fusible links, or as EPROM transistors and EEPROM cells in the case of EPROM and EEPROM devices, respectively.

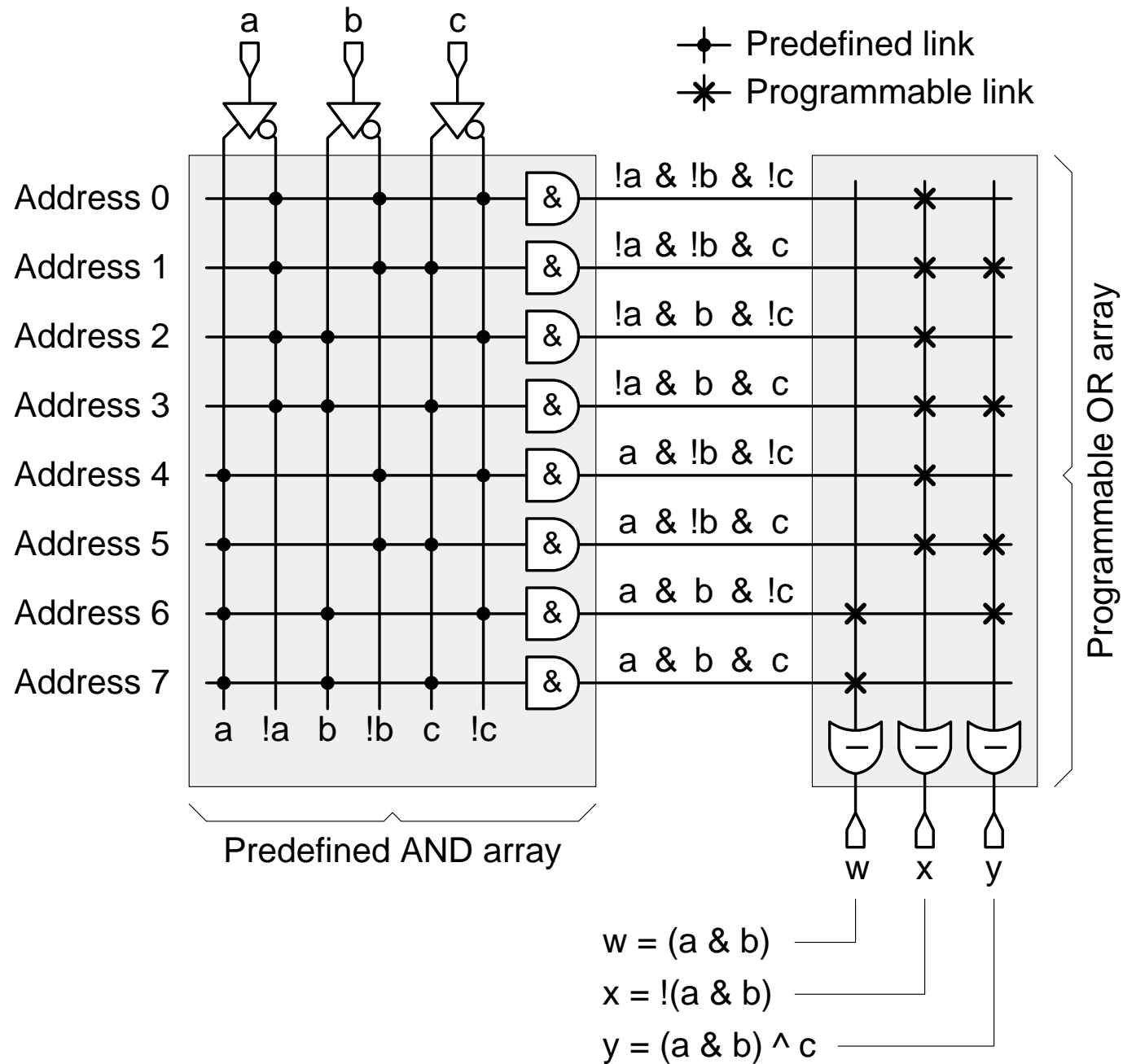
# Unprogrammed PROM



# Example

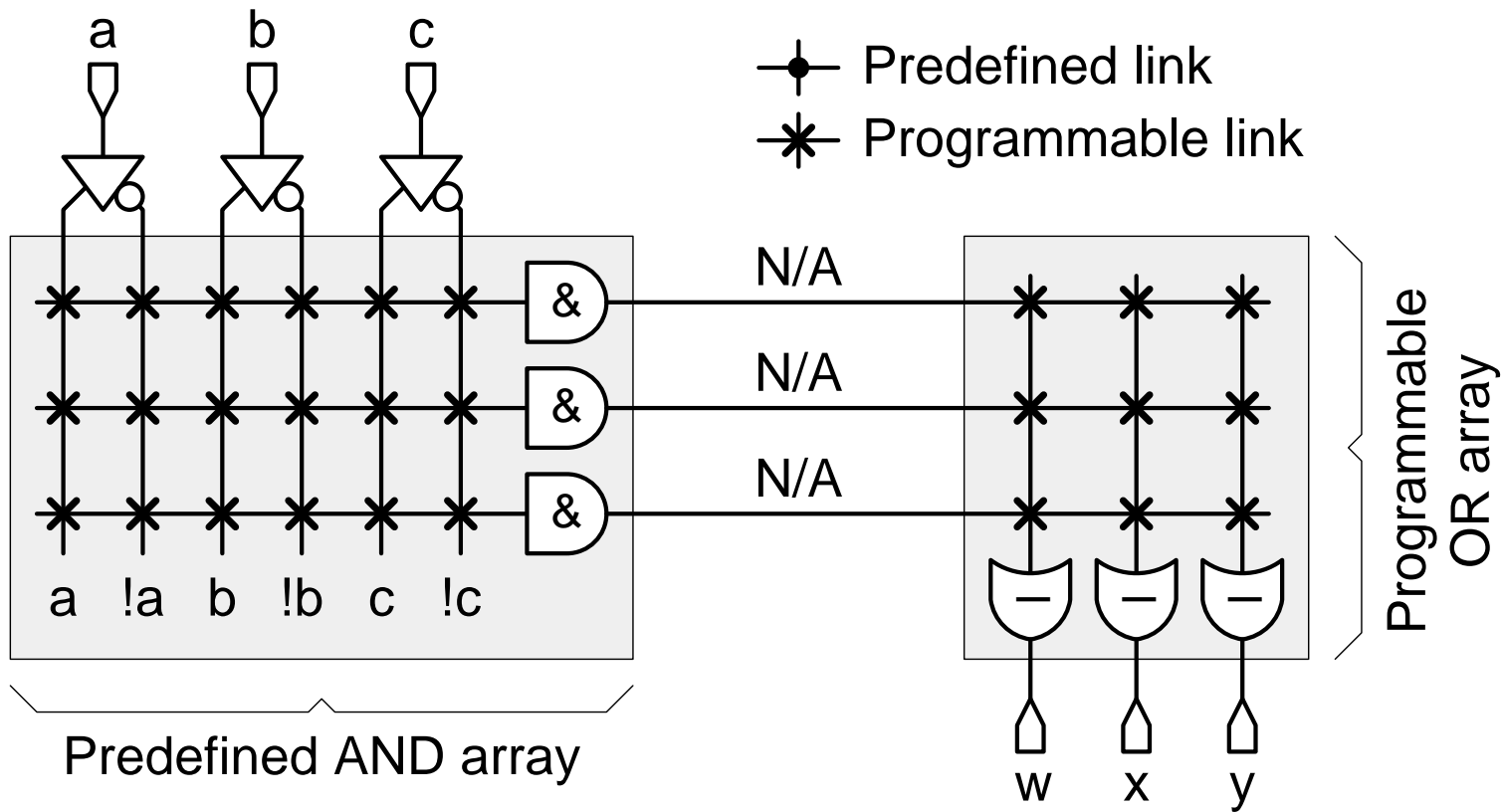


a	b	c	w	x	y
0	0	0	0	1	0
0	0	1	0	1	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	0	1	0
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	0	0

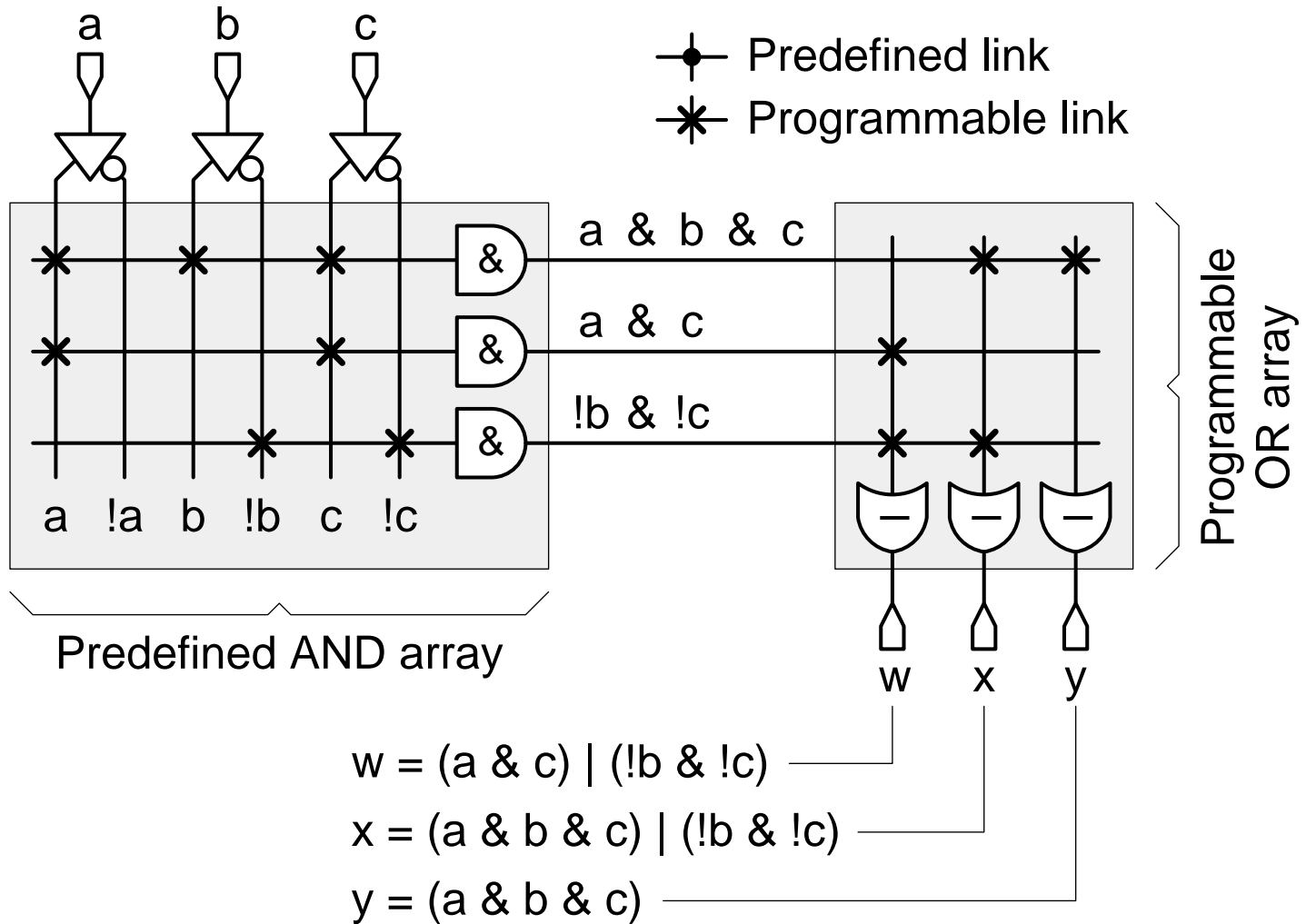


# PLAs

- In order to address the limitations imposed by the PROM architecture, the next step up the PLD evolutionary ladder was that of *programmable logic arrays (PLAs)*.
- These were the most user configurable of the simple PLDs because both the AND and OR arrays were programmable.



# Example

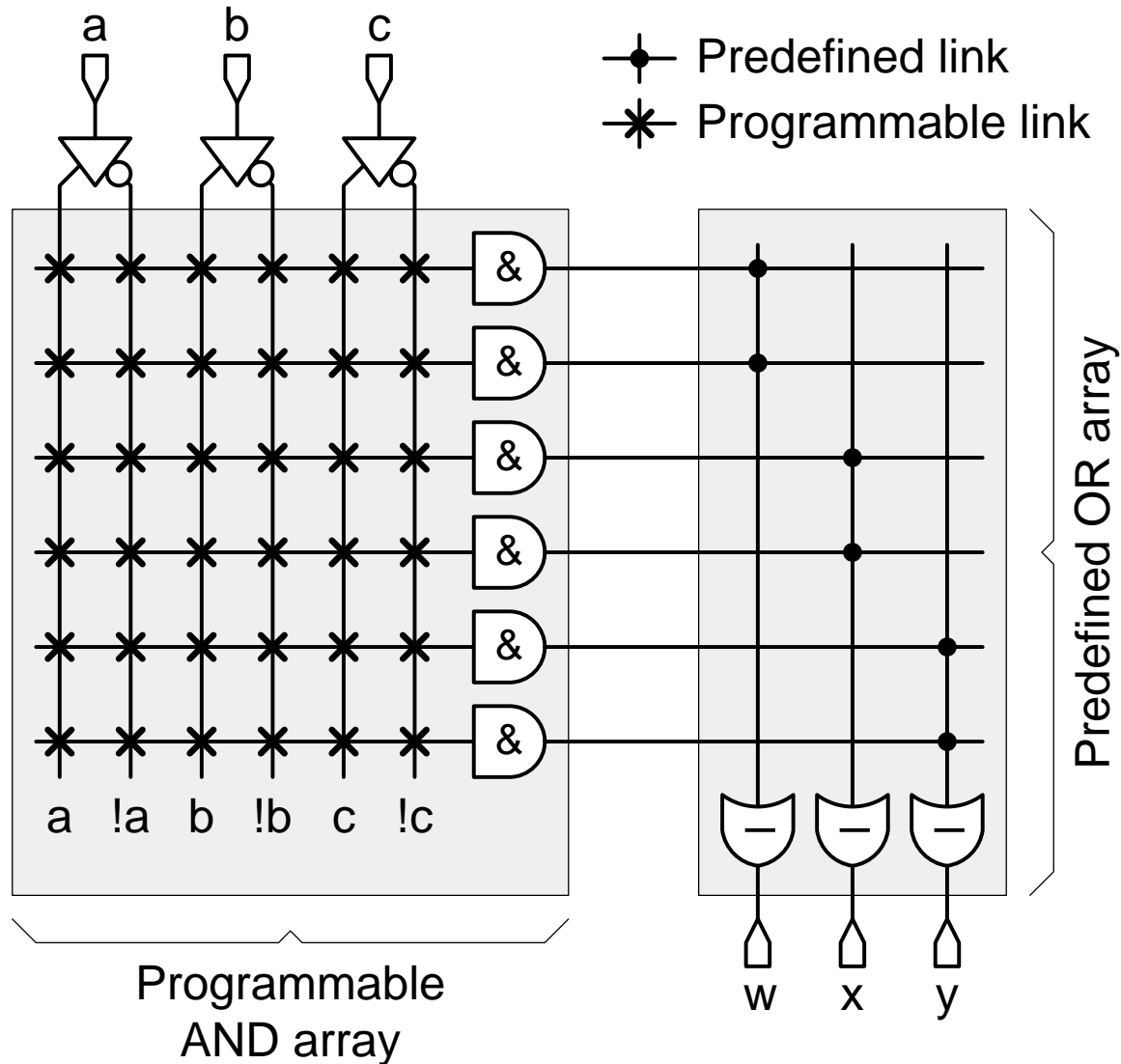


# PALs

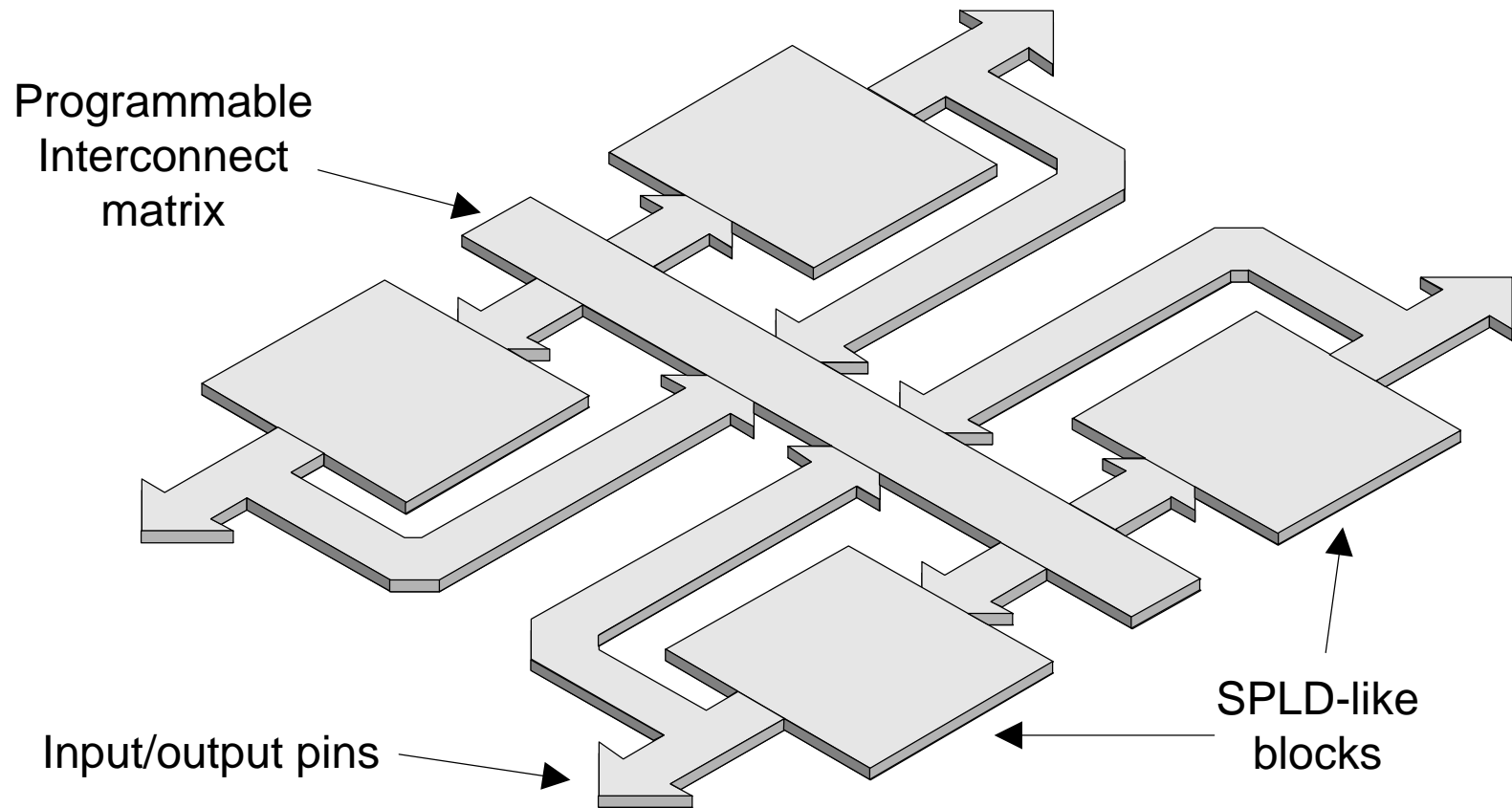
- In order to address the speed problems posed by PLAs, a new class of devices called *programmable array logic (PAL)* was introduced in the late 1970s.
- Conceptually, a PAL is almost the exact opposite of a PROM because it has a programmable AND array and a predefined OR array.



# Unprogrammed PAL

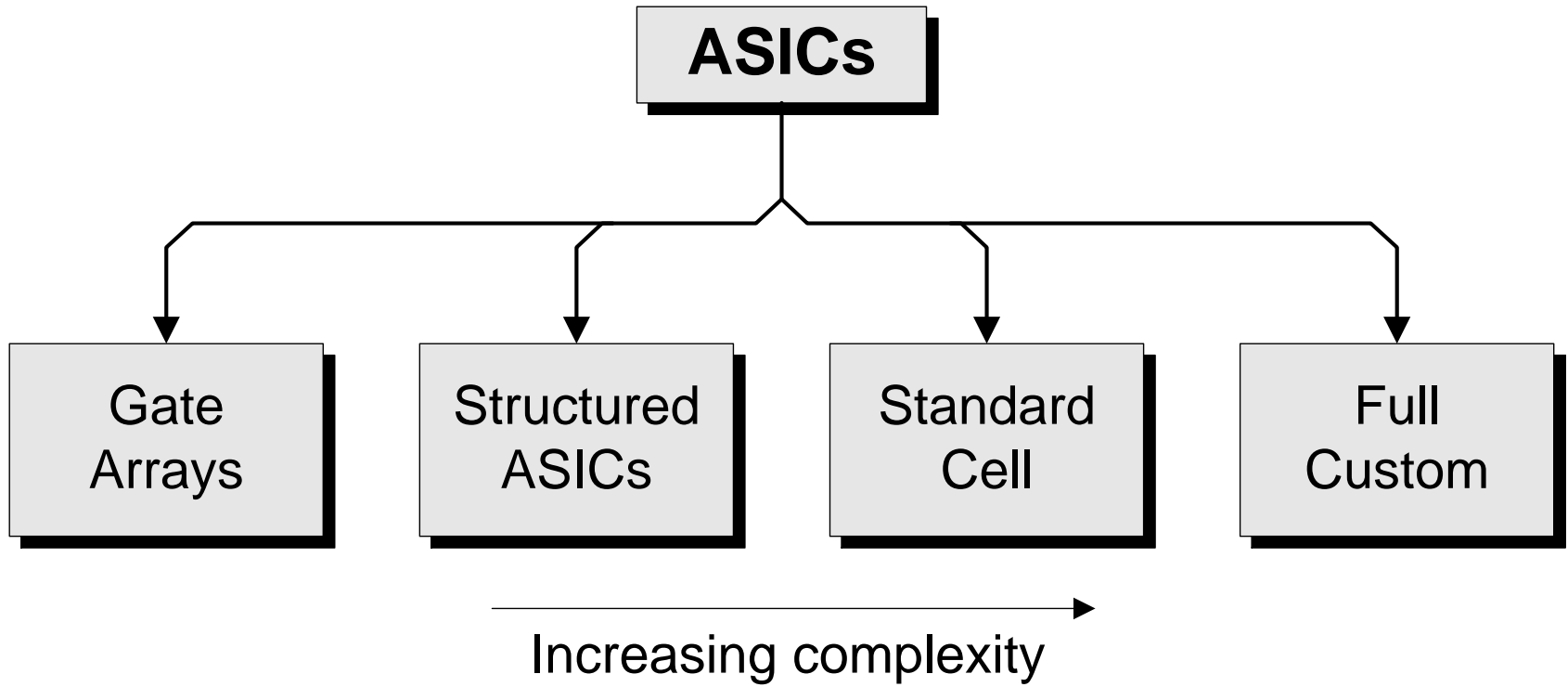


# CPLDs



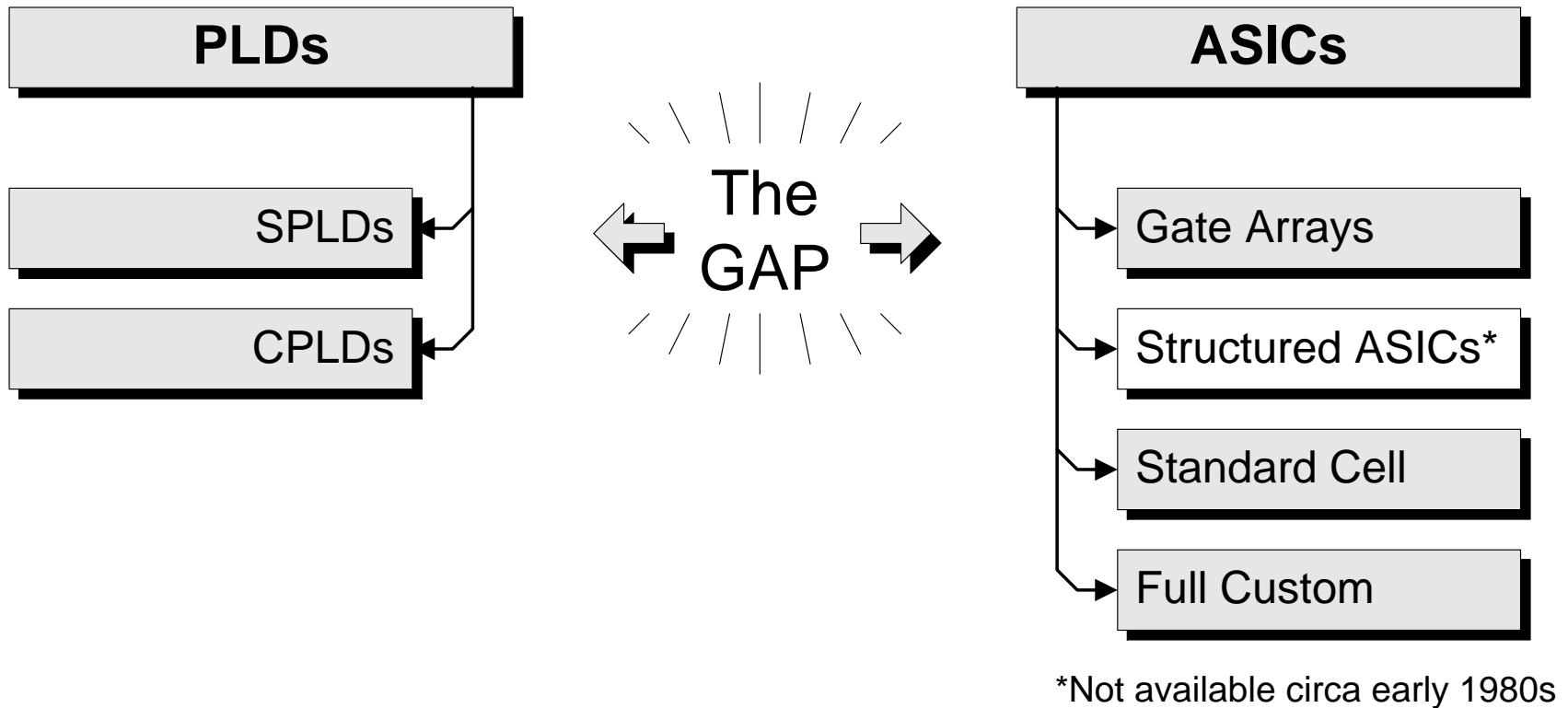
- Depending on the manufacturer and the device family, the CPLD's programmable switches may be based on EPROM, EEPROM, FLASH, or SRAM cells.

# ASICs



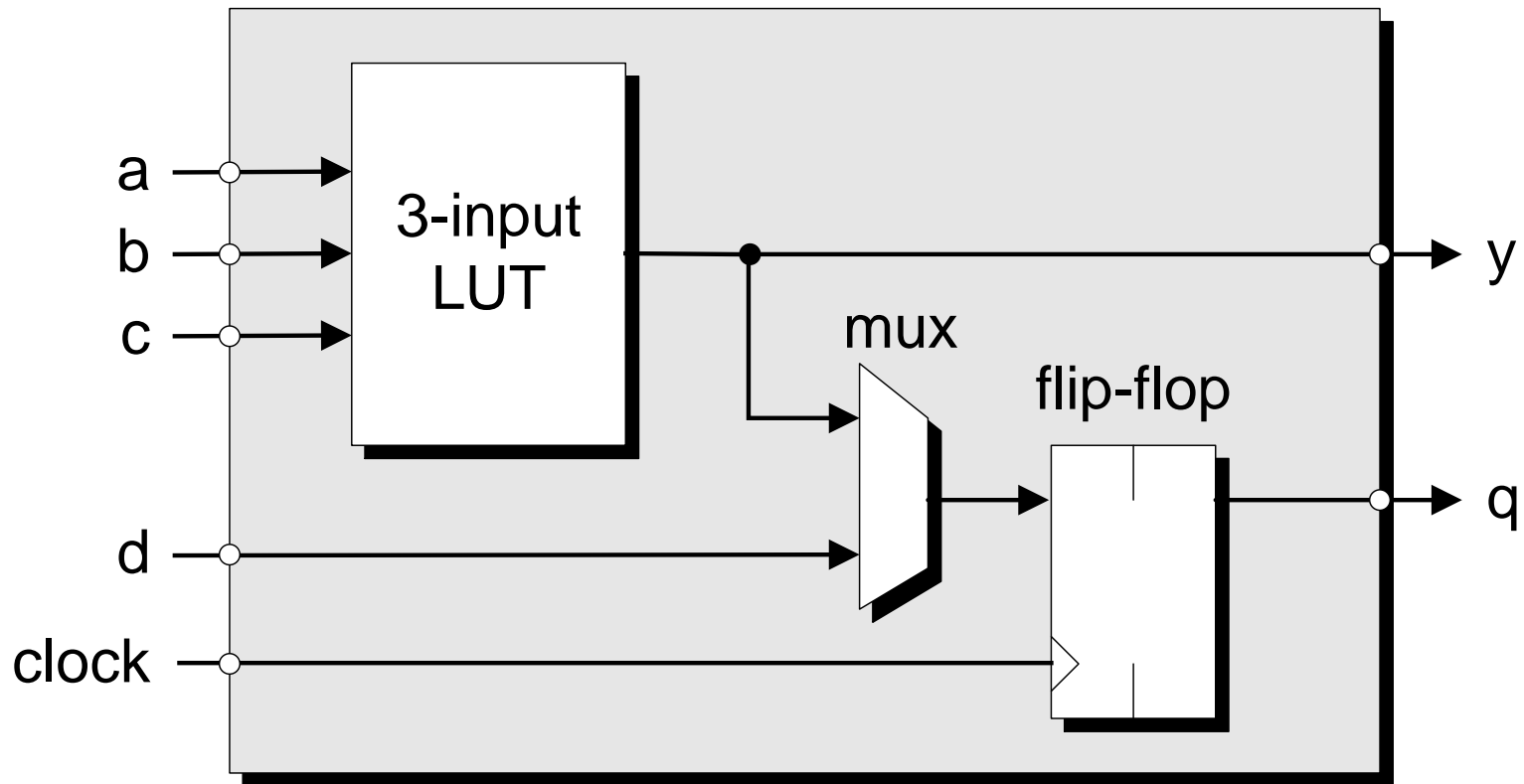
# FPGAs

- Round the beginning of the 1980s, it became apparent that there was a gap in the digital IC continuum.
- At one end, there were programmable devices like SPLDs and CPLDs, which were highly configurable and had fast design and modification times, but which couldn't support large or complex functions.
- At the other end of the spectrum were ASICs. These could support extremely large and complex functions, but they were painfully expensive and time-consuming to design.



- In order to address this gap, Xilinx developed a new class of IC called a *field-programmable gate array*, or FPGA, which they made available to the market in 1984.

# The key elements forming a simple programmable logic block



# Example: Configuring a LUT

