Chapter 1

Introduction

Electronic devices around us, from a TV set and a mobile phone to a personal computer are all digital devices at least to some extent. Digital devices process signals like audio and video converted into a digital form, that is, into a set of numbers. These numbers, in turn, are represented inside the digital devices in a binary form which at the conceptual level consists of two symbols, namely, zero and one. Inside the digital devices these binary symbols will be typically represented by two levels of voltages, conventionally referred to as LOW and HIGH, or by two position of a basic electronic switch, namely, ON or OFF. These simple concepts form foundation stones of the enormous success of digital devices.

We start building up our knowledge of electronic digital devices from the fundamental principles up to the complex concepts. This knowledge build-up has its parallel in a design process which employs the **bottom-up approach**. Such an approach is convenient firstly when we study a new field, but it also delivers near-optimal circuity. Its drawback is that it is usually a time consuming design approach. Later on, we will find out that the **top-down approach**, possible when we have better understanding of the domain of digital devices, usually delivers a solution in a shorter period of time, although the solution obtained in such a way is not always optimal.

1.1 Fundamental concepts of Digital Design

As it might be easily predicted in a real-life situation a process of designing digital devices in an iterative one and involves a sequence of **top-down** and **bottom-up** steps. At the introductory level, we will examine first a concept of the top-down approach. In such a process

- at the **top**, we usually start with the **design specification** given initially in an un-formal language, and
- at the **bottom** in the physical domain we obtain a digital device which, in its most common form, is a **printed-circuit board** (PCB) populated with **integrated circuits** (IC).

The basic design steps are illustrated in Figure 1.1.

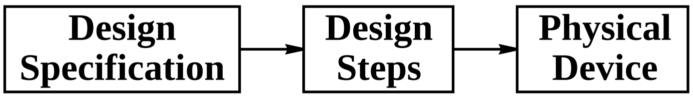

Figure 1.1: The design process

1.1.1 Physical domain

Normally the heart of a typical digital device is a printed circuit (PCB) populated with **Integrated Circuits**.
In an earlier generation of digital devices:

- such a printed-circuit board was populated with standard off-the-shelf components built in the Medium-Scale-Integration (MSI) technology. Typical representatives of this level of integration were: a 4-bit register, 4-bit adder, etc. On a typical PCB there were, say, $8 \times 12 = 96$ such components organized in rows and columns.

- The physical size of the MSI components was uniformed, a 16-pin DIL package being a typical representative, therefore, the PCB looked very regular.

Now, if you look at a motherboard of a typical PC you note that

- it is populated with relatively few integrated circuits of often different sizes and pin numbers.

- Typical components on the motherboard include a **general-purpose processor**, say Pentium X, **specialized processors**, say communication processor(s), video processor and other **application specific** integrated circuits (ASIC).

We will be studying ways of designing integrated circuits and we will not elaborate on the level of a PCB apart from introductory comments in this chapter. Our emphasis will be on the application specific integrated circuits rather than on a general purpose processors. However, at the lower level both types of circuitry share the common principles.

1.1.2 Implementation Options for Digital Devices

The name Application Specific Integrated Circuit (ASIC) implies that we design and build an integrated circuit which implements an **algorithm** for a specific application.
In general, in order to implement a required algorithm we can use

- A General Purpose Processor, e.g., Pentium X, for which a dedicated software package has been written,
- A Specialized Processor, e.g. DSP, also driven by a specialized software,
- a Dedicated Digital Device in which the algorithm is implemented at the hardware level.

These choices are illustrated in Figure 1.2.

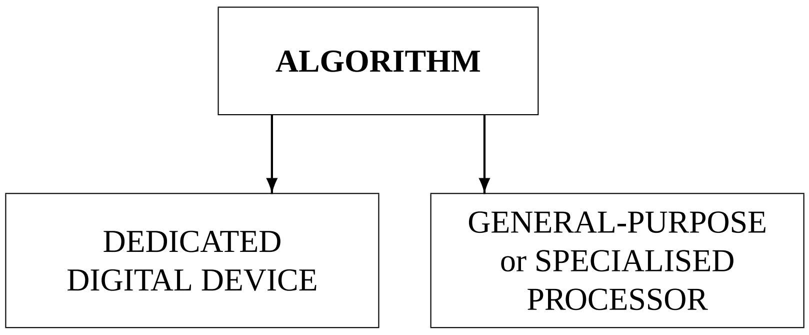

Figure 1.2: Two basic algorithm implementation options: hardware or software based.

Example of a specialized algorithm

In order to illustrate better the implementation options let us consider an algorithm for a one-dimensional linear digital filter (Finite Impulse Response (FIR) filter). Algorithmic specification of such a filter is presented in Figure 1.3. The algorithm for such an FIR filter requires

$$x(k) \longrightarrow \boxed{\begin{array}{c}\text{LINEAR}\\ \text{FIR}\\ \text{FILTER}\end{array}} \longrightarrow y(k) = \left[\begin{array}{cccc} a_1 & a_2 & \ldots & a_n \end{array}\right] \left[\begin{array}{c} x(k-1) \\ x(k-2) \\ \ldots \\ x(k-n) \end{array}\right]$$

Figure 1.3: Algorithmic specification of an FIR filter: a_i are constant coefficients, and $x(k-i)$ is the sample of the input signals measured at the time instant $k-i$.

that n multiplications of the signal samples $x(k-i)$ by filter coefficients a_i, and $n-1$ additions, to be performed every sampling instant, t_s. In such a case, we often refer to a multiply-accumulate (MAC) operation to be executed every sampling instance.
For the above FIR filter the choice of design options is based on the following considerations.

- For a relatively low sampling frequency, f_s, say 100KHz, and a small size of the filter n, a general purpose computer can deliver the required throughput.
- For a relatively high sampling frequency, say 10MHz, we will use a dedicated DSP, which is specialized in fast multiply-accumulate operations.
- Finally, for very high frequency, say 1Ghz, a **dedicated digital device** may be the only option do deliver the required processing speed.

Technologies for Specialized Digital Devices

At present, there are two prevailing technologies used to build specialized digital devices, typically referred to as

- **FPGAs** — Field Programmable Gate Arrays.

  A typical device, XILINX XC6200 series, consists of up to 100k logic gates organized in 16K cells which are arranged in hierarchical array structure. Each cell contains a 1-bit register and 2-bit universal function generator.

  Configuration of an FPGA device, that is, description of the cell functions and interconnections between them, is set up from an internal 262K-bit RAM. This configuration RAM is loaded from an external EPROM or a PC during the power-on operation.

- **ASICs** — Application Specific Integrated Circuits, which are built using two basic methodologies:

  - Full-custom design

    The layout of the chip is designed from the low-level geometric primitives, such as boxes and wires, using a graphics editor.
  - Standard-cell Design
    The layout of the chip is compiled from layouts of the standard cells. Library cells are automatically placed and routed using a software tool often referred to as a Silicon Compiler.

1.2 Description Levels for Digital Devices

Design process of digital devices can be considered in three basic **domains**, namely, behavioral, structural and physical domains. Each domain requires different approaches and dedicated design tools. In addition the complexity of digital devices can be considered on four basic **levels**, namely, circuit, logic, functional block and system levels. The relationships between the design objects for each domain and level are captured in Table 1.1 and, in a slightly different form in Figure 1.4.

Level	Domain		
	Behaviuoral	Structural	Physical
System Level	Algorithms, Spec. charts, flow-charts	Processors, Controllers, Memories, Buses, ...	Cabinets, Boards, Chips, ...
Functional block level	Register Transfer, State diagrams, ...	Registers, ALUs, Shifters, Multiplexers, ...	Floorplans, module placement, ...
Logic level	Boolean eqns, ...	Gates, Latches, Flip-Flops, ...	Cells, Modules
Circuit level	Current-voltage eqns, ...	Transistors, nets	Circuit layouts

Table 1.1: Design objects for different domains and levels.

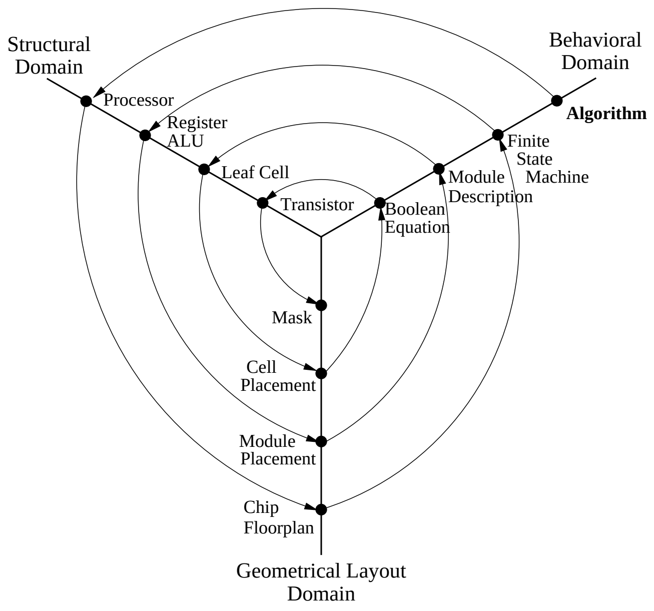

Figure 1.4: The Y-Chart (after Gajski) representing location of design objects for each domain.

We start our considerations at the circuit level and gradually progress toward the functional level and study the design methods in all three domains.
In order to further illustrate the concept of design levels and domains, in the following subsections we will provide an example for every entry in the Table 1.1.

1.2.1 System Level

We start at the top with a system level where an algorithm (behavioural domain) can be implemented as a processor (structural domain) in a form of an integrated circuit (physical domain). As an illustrative example let us consider a multiplication processor.

Behavioral domain

Multiplication of two digital numbers, $P = D * Q$, can be performed using many algorithm. A popular word-serial Booth's algorithm can be described by the following pseudo-code.

```
$P[0] = 0$ ;  $q_{-1} = 0$ ;
for ($i = 0$ ; $i < n$ ; $i{+}{+}$) {
    $\hat{q}_i = -q_i + q_{i-1}$ ;                         (1.1)
    $P[i+1] = (P[i] + \hat{q}_i \cdot D) \cdot 2^{-1}$ ;
}
```

Alternatively we could describe the above algorithm using a flow-chart. Details of the algorithm are not important at this stage, but from the above code we will conjuncture that the processor needs register to store variables used in the above program and arithmetic-logic unit(s) to perform required low level operations.

Structural domain

Structurally, a multiplication circuit implementing algorithm (1.1) is a typical example of a simple processor. Such a processor can be by definition decomposed into two basis parts, namely the **datapath** and the **control unit**. A relevant schematic with such decomposition generated by the Design Architect is presented in Figure 1.5.

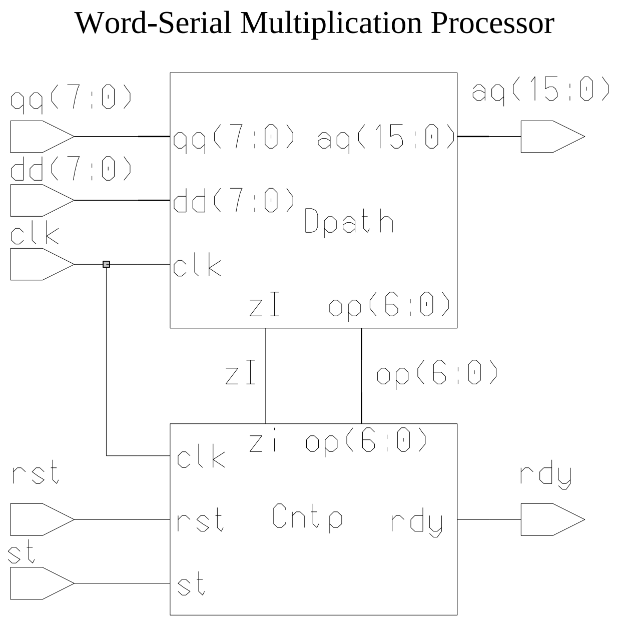

Figure 1.5: A possible decomposition of a multiplication processor into its datapath and control unit. Note the input/output ports and internal signal.

The decomposition of the multiplication processor into its main subsystems specifies also its input/output (I/O) ports and signals exchanged between the subsystem.

Physical domain

The multiplication processor is a relatively simple circuit and can be easily built as a single ASIC or FPGA. The I/O ports are mapped onto the pins of the integrated circuit. We design such a circuit in sec. ? Example of a layout of an m 8-bit multiplication processor is given in Figure 1.6.

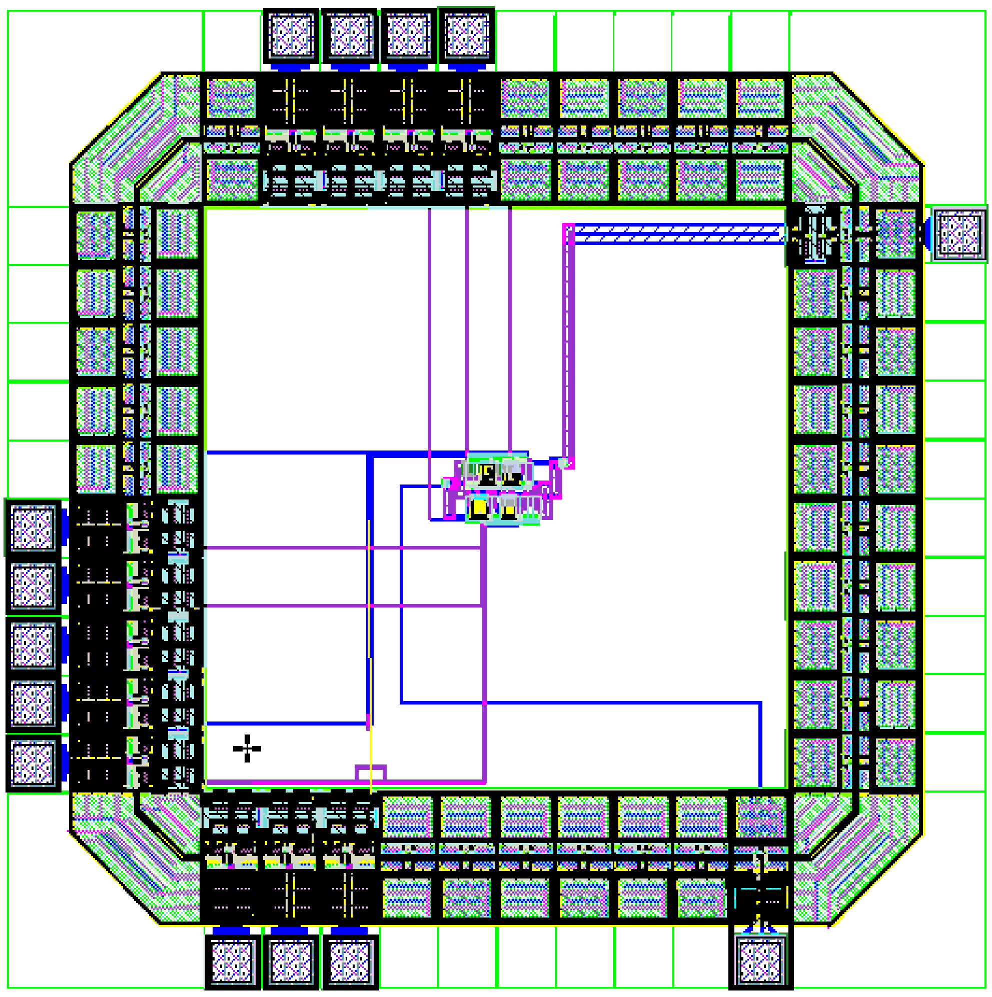

Figure 1.6: Layout of a 4-bit squaring circuit

1.2.2 Functional Block Level

The functional block level, also known as the micro-architectural level operates with objects like registers (to store variable from the previous level) and various blocks which perform required low-level operations, addition and shift being the most typical examples. In the behavioral domain we use a register-transfer language (RTL) and state diagram as basic tools. In the physical domain we will need tools to describe placement and interconnections of the structural objects.

Structural domain

A possible structure of the datapath of the multiplication processor resulting from the code given in (1.1) is presented in Figure 1.7.

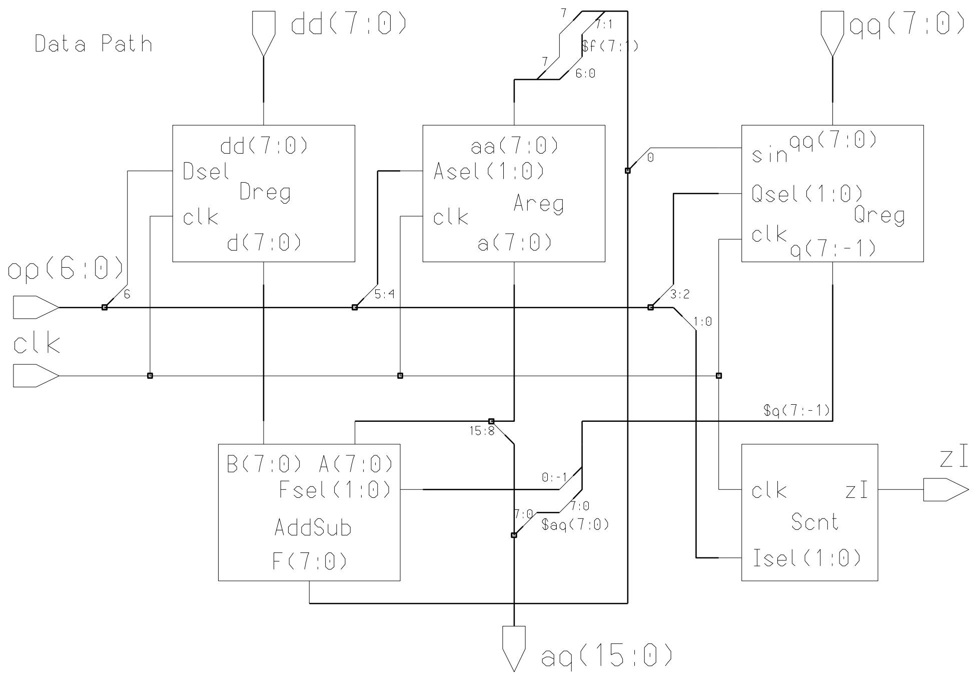

Figure 1.7: A possible internal structure of the datapath of the Booth’s multiplication processor

Main components of the datapath are three registers, Dreg, Areg, Qreg, a step counter, Scnt, and an arithmetic-logic unit AddSub. Shift operations are performed by appropriate wiring, as at the entry to the Areg register.

Behavioral domain

At this stage we have to specify the behavior, that is, a function of every component of the datapath. As an example we will specify the Q register, Qreg. Firstly, we can described elementary operations performed by this register in Table 1.2.

Multiplier register, Qreq		
Qop	operation	
0	$Q \Leftarrow Q$	nop
1	$Q \Leftarrow$ shr(F(0), Q)	shiftR
2,3	$Q \Leftarrow$ (qq & 0)	load

(‘&’ is the VHDL concatenation operator)

Table 1.2: A table of operation of the multiplier register

This description is often referred to as the Register-Transfer-Language (RTL). In addition to the tabular specification, we can give the following description of the register in VHDL:

```
--    qreg - multiplier register
library IEEE ;
use IEEE.std_logic_1164.all ;
ENTITY qreg IS
  PORT (clk, sin : IN  STD_LOGIC ;
             Qop : IN  STD_LOGIC_VECTOR(1 DOWNTO 0) ;
              qq : IN  STD_LOGIC_VECTOR(7 DOWNTO 0) ;
               Q : OUT STD_LOGIC_VECTOR(8 DOWNTO 0) );
END qreg ;

ARCHITECTURE behv OF qreg IS
  SIGNAL P, Din : STD_LOGIC_VECTOR(8 DOWNTO 0) ;
BEGIN
-- input multiplexer of the qreg
  WITH  Qop  SELECT
  Din <=  P                   WHEN  "00" ,
         (sin & P(8 DOWNTO 1)) WHEN  "01" ,
         (qq  & ’0’)           WHEN  OTHERS ;
```

```
-- positive-edge triggered flip-flops:
 clkd: PROCESS (clk)
 BEGIN
    IF ( clk’EVENT AND clk = ’1’ ) THEN
       P <= Din ;
    END IF ;
 END PROCESS clkd ;

-- assigning internal signal to the external output port.
 Q <= P ;
END behv ;
```

In the above VHDL code we can identify two basic blocks of the qreg, namely the input multiplexer and the flip-flops.

Physical domain

A layout of a 9-bit register qreg is given in Figure 1.8.

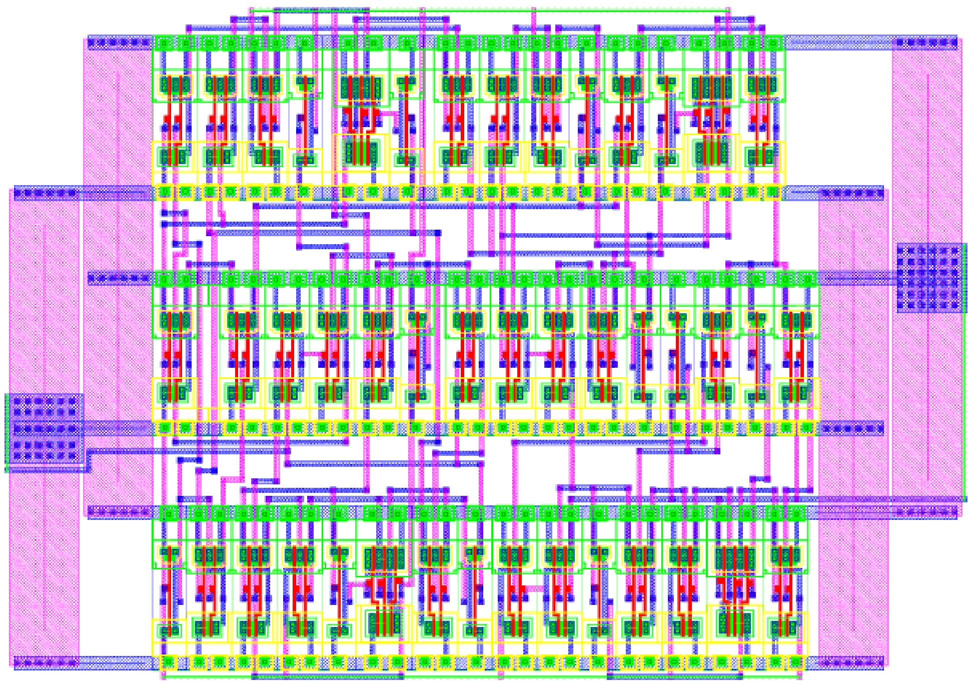

Figure 1.8: A layout of a division-by-3 circuit

1.2.3 Logic Level

At the logic level we design basic building blocks to be used in the next, functional level. Typical building blocks are: gates, multiplexers, 1-bit adders, latches and flip-flops.
As an example let us consider a simple logic circuit represented first by its logic diagram.

Structural domain

A simple logic circuit can be represented by interconnection of simple gates as in Figure 1.9.

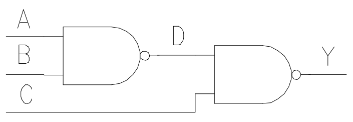

Figure 1.9: A logic diagram

Note that

- structural representation at the logic level uses **logic symbols**.
- Interconnections represent **logic signals**

Behavioral domain A the logic level the behavior can be conveniently described by Boolean expressions. The logic diagram from Figure 1.9 is represented by the following equations:

$$D = \overline{A} + \overline{B}, \quad Y = \overline{C \cdot D} = A \cdot B + \overline{C}$$

Physical domain In the physical domain of the logic level we typically work with circuit layouts of cells from the library of standard cell. We will discuss details of such cells later on. At this stage we present a layout of an inverted sum-of-products gates, aoi221, from a library of standard cells, AMI05.

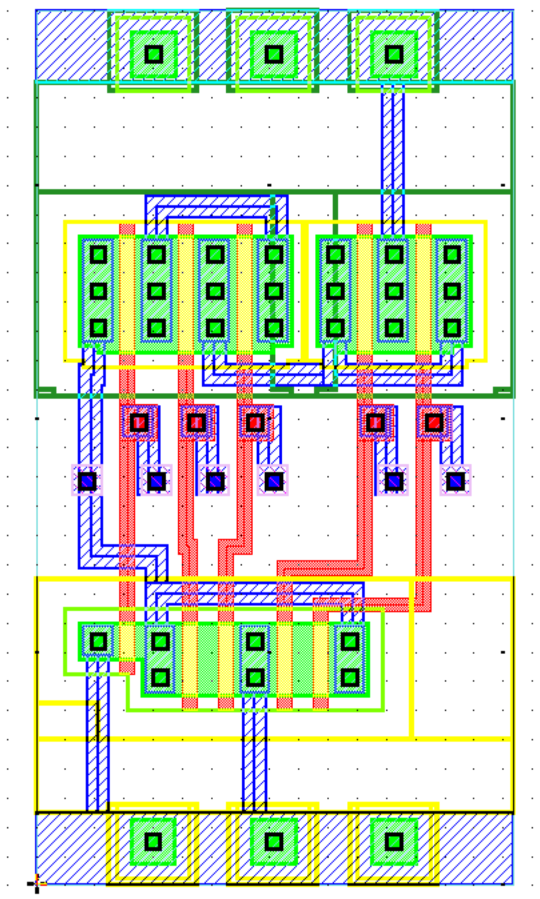

Figure 1.10: A layout of an aoi221 gate from the AMI05 library

The aoi221 cell performs operation

$$Y = \overline{a1 \cdot b1 + a2 \cdot b2 + c}$$

1.2.4 Circuit Level

The class of digital circuits we will be preoccupied in this book, namely, the CMOS circuits consist almost exclusively of MOS Field-Effect Transistors (FET). The simplest example we will consider at the circuit level is a CMOS inverter.

Structural domain

A structural representation of a CMOS inverter, that is, its schematic is given in Figure 1.11.

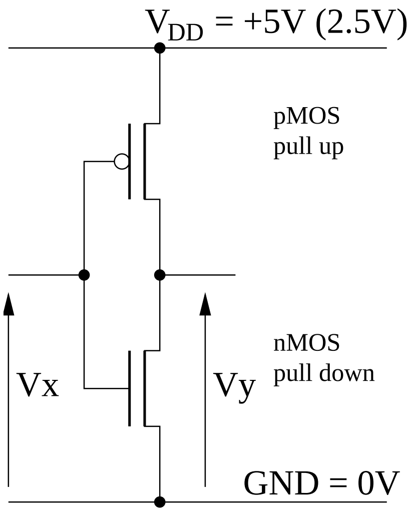

Figure 1.11: A layout of an aoi221 gate from the AMI05 library

- Structural representation uses **circuit symbols**, namely, transistors, power and ground rails/tetminals.
- Interconnections represent **electric wires** through which currents and voltages are applied to the circuit

The CMOS (Complementary Metal-Oxide-Semiconductor) inverter is built from two MOS Field-Effect Transistors (FET), nMOS and pMOS. The charge carriers are negative (electrons) in the nMOS, and positive (‘holes’) in the pMOS transistor.

Behavioral domain

At the circuit level, the partial static behavioral description of the CMOS inverter can be given by its transfer characteristics:

- The output voltage V_y versus the input voltage V_x (top right of Figure 1.12), and
- The current I_D between V_{DD} and GND versus the input voltage V_x (bottom right of Figure 1.12).

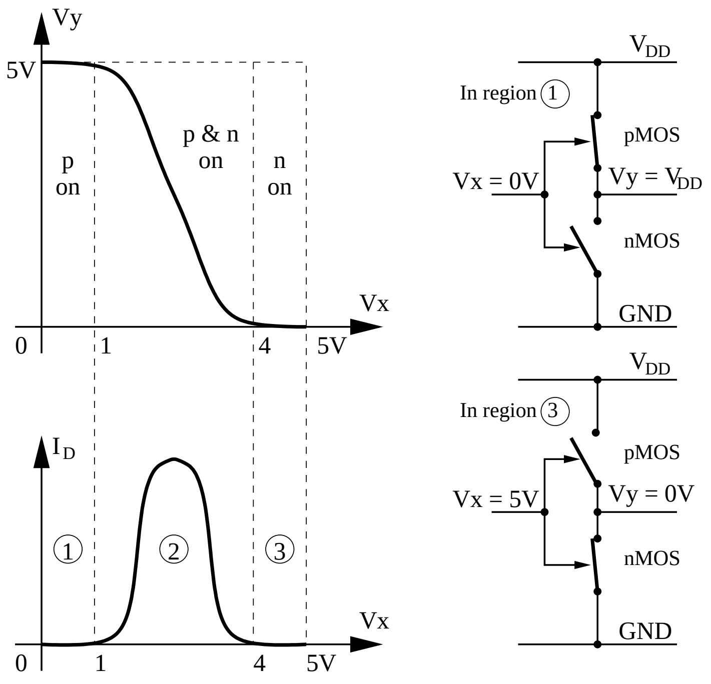

Figure 1.12: Transfer characteristics and a simplified switch model of a CMOS inverter.

The function of the inverter can be approximated by two complementary switches as illustrated in Figure 1.12:

- In regions '1' and '3', when the input voltage V_x is either close to 0V or to V_{DD}, no current flows between V_{DD} and

GND, and the output voltage is either V_{DD}, or 0V, respectively.

- During transition, there is a current pulse occurring as illustrated in the current characteristic.

Physical domain

A circuit layout of a CMOS inverter is presented in Figure 1.13.

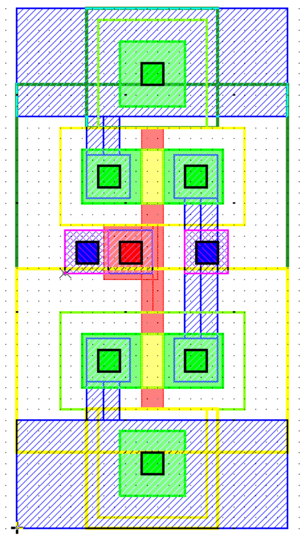

Figure 1.13: A layout of an inverter

1.3 CMOS technology — basic facts

CMOS (Complementary Metal-Oxide-Semiconductor) integrated circuits are built on the surface of a **silicon wafer** and are composed of layers of three basic materials:

- Silicon, Si, — semiconductor
- Oxide — Silicon Dioxide, SiO_2 — perfect insulator,
- Metal — Aluminium, Al — very good conductor.

These three materials are used to fabricate two types of complementary field-effect transistors, namely, nMOS and pMOS FETs, and interconnections between them.
All digital devices, from a single inverter to a powerful processor of a personal computer, with the exception of dynamic memories, are built from these two transistors.
We will consider details of the CMOS technology in Ch. **??**. Here we present some fundamental details needed to understand working of a MOS transistor.
The substrate on which integrated circuits are build, a **silicon wafer**, is a polished disk cut out of a single silicon crystal. Typically, the wafer has a diameter of 75–300mm and thickness around 1mm.

Silicon is a fundamentally important material for integrated circuit manufacturing because its electrical properties can be modified by relatively simple chemical process. **Pure (intrinsic) silicon** has relatively high resistance, that is, there are relatively few free electrons able to conduct electric current, therefore it is usually **doped** with impurities which modify its electrical properties:

- **p-type** silicon is obtained by doping it with **acceptors** what creates an excess of the **positive** charges (holes),
- **n-type** silicon is obtained by doping it with **donors** what creates an excess of the **negative** charges (electrons).

Doped silicon has significantly lower resistance.

Integrated circuits are fabricated by a series of four basic chemical processing steps:

Diffusion (Figure 1.14)

During diffusion process dopant atoms are inserted into the silicon crystal lattice.

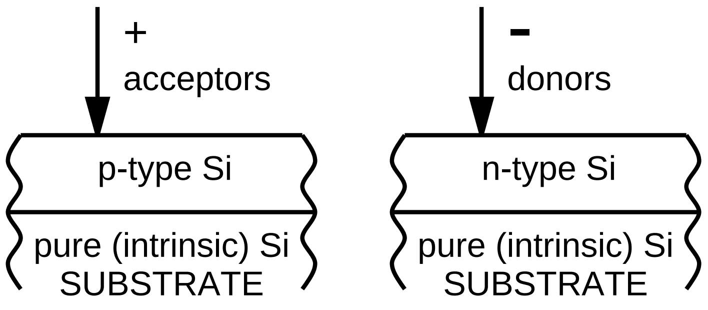

Figure 1.14: Diffusion process

- Acceptors form the p-type silicon by accepting some electrons already present in the silicon hence creating positive charges, called holes.
- Donors supply more electrons thus creating the n-type silicon.

Oxidation (Figure 1.15)

Heating the silicon surface in the presence of oxygen (O_2) creates a layer of Silicon Dioxide SiO_2 (oxide for short) which is a very good insulator.

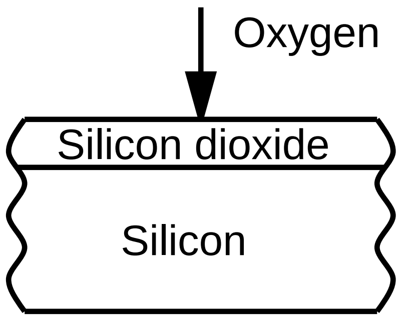

Figure 1.15: Oxidation process

Deposition (Figure 1.16)

Deposition is used to create a layer of a conducting material usually on the top of the silicon dioxide layer. Two conducting materials are used:

- **metal**, usually aluminium, Al, and
- **polysilicon**, which is a polycrystalline silicon modified with some metal impurities to obtain a relatively good conductance.

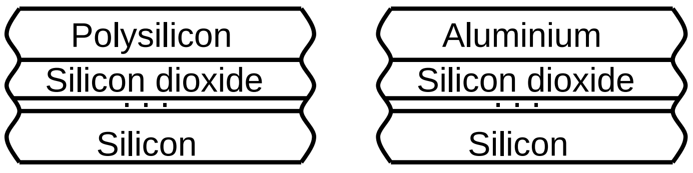

Figure 1.16: Diffusion process

Etching

Current layer of the material is etched to form a required pattern as illustrated in Figure 1.17.

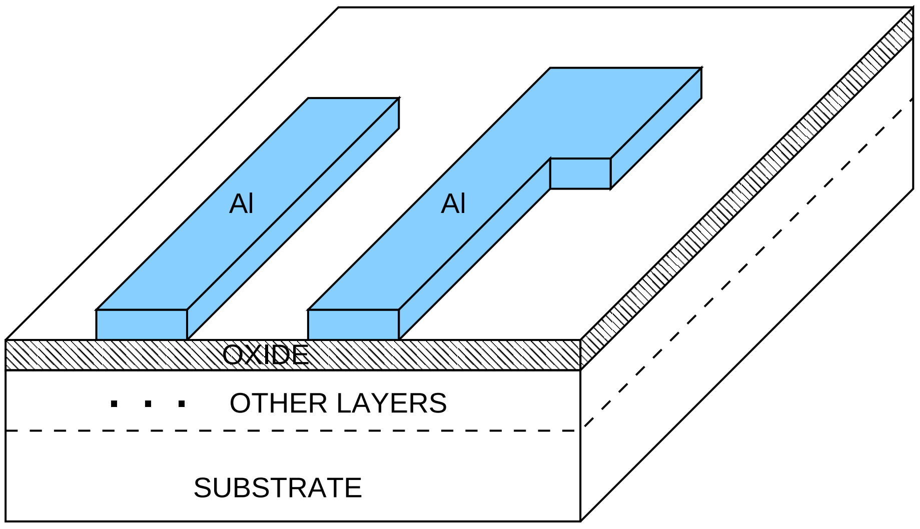

Figure 1.17: Etching

Patterning is performed with the use of the **photo-lithographic masks**. The masks are obtained from the **circuit layout**.

The size of the smallest features on the silicon surface is now as small as 0.13μm.

(1μm = 10^{-6}m = 10^{-3}mm = 10^{3}nm)

1.4 IC technology today

- The ”Moore’s Law” (1965, Gordon Moore, co-founder of Intel):

  > The maximum number of transistors on a chip doubles every 18 months.

- As a result the on-chip transistor count has increased one million times in three decades and has reached 30 million transistors (in microprocessors).
- Semiconductor memories (DRAM) reached the density of 1Gbits per chip.
- The increase of the transistor count is the result of reduction in the size of the smallest manufacturable feature, say, the minimum path/line width.
- The minimum feature size has decreased from around 10μm in the 1970s to less than 0.2μm in 2000.
- In order to reduce power consumption the supply voltage has been reduced from 5V to around 2V.
- It has been predicted that in 2009 we will have:

  200M transistors on a chip with the minimum feature size 100nm and the supply voltage 0.6–0.9V. The DRAM will have 64Gb per chip.
- We will be working with AMI05 and TMS035 technologies with the minimum feature size being 0.5μm and 0.35μm, respectively.
- Silicon (Si) is the most popular material use to build integrated circuits. Other possibilities include germanium (Ge) and gallium arsenide (GaAs).