余锋实验室

HDL组织的实验室参观，算是仪器方向的大组。

【科研学术】求问生仪学院马振国老师 https://www.cc98.org/topic/6119103

无人机阵列抗干扰，干扰卫星

高性能嵌入式系统，数模混合

异构计算

rtos 实时传感数据同步

微处理器系统

微处理器系统，囊括了各种类型的计算机，微控制器/单片机。世界上的微处理器系统的总数比人类总数还多得多。它的基本工作原理是用程序控制系统的行为。

微处理器系统的基本操作过程是中央处理器（Central Processing Unit, CPU）不断地从存储器取指并执行，实现对系统的全面管理。

一、CPU结构和功能CPU的结构（下图为CPU的结构）

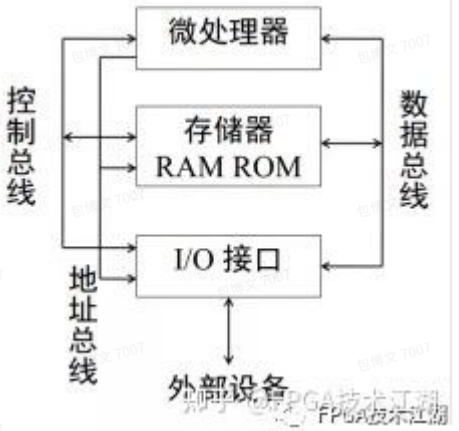
1. 控制器：完成指令的读入、寄存、译码和执行。
2. 寄存器：暂存用于寻址和计算过程的产生的地址和数据。
3. I/O控制逻辑：负责CPU中与输入/输出操作有关的逻辑。
4. 算数逻辑运算单元（Arithmetic & Logic Unit, ALU）：运算器核心，负责进行算术运算、逻辑运算和移位操作，用来进行数值计算和产生存储器访问地址。

CPU的功能：

1. 与存储器之间交换信息。

2. 和I/O设备之间交换信息。
3. 为了使系统正常工作而接收和输出必要的信号，如复位信号、电源、输入时钟脉冲等。

二、微处理器系统的结构（下图为微处理器系统的结构）



1. CPU的外部特征就是数量有限的输入输出引脚。
2. 数据总线：用于CPU和存储器或I/O接口之间传送数据，双向通信；数据总线的条数决定了CPU和存储器或I/O设备一次最多能交换数据的位数，是微处理器的位数的判据，例如：Intel 386DX、ARM Cortex-M3是32位微处理器；Intel采用了IA-64架构的处理器、PowerPC 970是64位处理器；类似地，还有更加古老的8位、16位处理器等。
3. 地址总线：CPU通过地址总线输出地址码用以选择某一存储单元或某一成为I/O端口的寄存器，单向通信；地址总线的条数决定了地址码的位数，进而决定了存储空间的大小，例如：地址总线宽度（条数）为8，则可以标记2^8 = 256个存储单元，若每个存储单元的字长为8 bit，则最大可以接入系统的存储空间为256kB。
4. 控制总线：用来传送自CPU发出的控制信息或外设送到CPU的状态信息，双向通信； 微处理器系统的程序设计语言：程序设计语言（Programming Language），又称为编程语言，是用来定义计算机程序的，通过代码向处理机发出指令。编程语言让开发者能够准确地提供计算机所使用的数据，并精确地控制在不同情况下所应当采取的行动。最早的编程语言是在计算机发明之后产生的，当时是用来控制提花织布机及自动演奏钢琴的动作。在电脑领域已发明了上千不同的编程语言，而且每年仍有新的编程语言诞生。很多编程语言需要用指令方式说明计算的程序，而有些编程语言则属于声明式编程，说明需要的结果，而不说明如何计算。
   机器语言：机器语言的每条语句即是处理器可以直接执行的一条指令，这些指令是以二进制0、1序列的形式表示，对应数字集成电路的高低电平。不同的处理器指令的机器代码各不相同，完成的具体功能也将不相同，按一种计算机的机器指令编写的程序，不能在另一种计算机上执行。
   示例：（仅作为示例，不代表真实硬件的机器代码）

- 指令的机器代码：

> 0000’0000 STORE
>
> 0000’0001 LOAD

- 地址的机器代码：

> 0000’0000 寄存器R0
>
> 0000’0001 寄存器R1

优点：功能和代码一一对应，CPU可以直接执行，效率最高。

缺点：只有二进制0、1序列，枯燥，难以辨识。

汇编语言：用简洁的英文字母、符号串来替代一个特定的机器语言指令——二进制0、1序列：用助记符（Memoni）代替操作码，用地址符号（Symbol）或标号（Label）代替地址码。汇编语言与机器语言一一对应，所以和机器语言一样对计算机硬件的依赖性很大。

示例：加法运算（分号表示接注释）

```
MOV  R1,? R3;
将寄存器R3的值赋予R1
LDR  R2, [R4]
;将寄存器R4的值代表的地址对应的存储空间的值赋予R2
ADCS R0,  R1, R2
;将寄存器R1、R2与之前的进位值相加且进位，存储到寄存器R0
```


优点：汇编语句和机器语言一一对应，助记符与标号往往与实际意义相关，相比于机器语言，更加直观，容易理解，执行效率上类似。

缺点：不同的处理器指令集不同，移植性不好；即使完成简单的数据处理（如累加，简单排序等）所需的代码体积很大，处理实际问题所需的工作量夸张，成本高。

高级语言：使用接近于数学语言或人类语言的表达描述程序。

特点：相比于面向机器开发的机器语言和汇编语言，高级语言拥有较高的可读性，并且代码量大大减少；高级语言通常远离对硬件的直接操作，安全性较高，也有部分高级语言可以使用调用汇编语言的

接口操控硬件；高级语言有很多成熟、易于使用、可移植的数据结构与算法，使开发流程大大简化，节省开发成本，易于维护；发展迅速，社区完备，可以很方便地求助，解决遇到的各种问题；已经有很多各具特色、用以解决不同领域问题且发展相当完备的高级语言供开发者选用，如：适合初学者了解编程思想的Basic；效率颇高，接近于硬件操控，适合系统、硬件驱动编程与嵌入式开发的C/C++；跨平台、可移植特性优良的Java；搭配Visual Studio可以快速开发项目的C#.NET；适合于数据分析、人工智能，越来越被青睐的Python；Microsoft公司为未来的量子计算而开发的Q#，等等。诸如MATLAB、HTML、JavaScript这样的用以在不同领域大显身手的语言亦可以称之为高级语言。

示例：加法运算

```
int a = 1, b = 2, c;

c = a + b;
```


优点：不依赖于硬件，移植性好；不用场合选用适合的语言，开发效率高。

缺点：不直接使用硬件，需要编译-链接执行或解释执行，没有利用到具体硬件的特点，效率相比于机器语言和汇编语言不高；先天的特点决定了高级语言在底层的设计中无法完全取代机器语言和汇编语言。

可以看出，微处理器系统的核心部件是CPU，使用微处理器系统控制外部的设备工作的实质就是使用编写软件程序的手段来控制外部设备。由于CPU已经是一个完整的、封装好的部件，系统的设计人员只能通过编写软件，再经由编译器或解释器翻译为机器能够理解的代码来执行，CPU并没有专门的硬件电路来实现完全地控制外部设备的运行，这种实现方式是软件实现，是一种通用的实现，控制信号从软件到硬件要经过若干次转化，但有的时候，工程和设计领域往往需要高速高性能的芯片来实现控制与计算，这时候就需要更加强大的CPU或将几个CPU用一些技术并行起来协同工作，成本就会增加。这时候，可以不妨试试设计专门的硬件来满足工作的需求。

三、专用集成电路

专用的集成电路（Application Specific Integrated Circuit, ASIC）是一种为专门目的而设计的集成电路。是指应特定用户要求和特定电子系统的需要而设计、制造的集成电路。ASIC的特点是面向特定用户的需求，ASIC在批量生产时与通用集成电路相比具有体积更小、功耗更低、可靠性提高、性能提高、保密性增强、成本降低等优点。

ASIC分为全定制和半定制。全定制设计需要设计者完成所有电路的设计，包括芯片设计的所有流程，因此需要大量人力物力，灵活性好但开发效率低下。如果设计较为理想，全定制能够比半定制的ASIC芯片运行速度更快。半定制ASIC使用准逻辑单元（Standard Cell），设计时可以从标准逻辑单元库中选择SSI（小规模集成电路，如门电路）、MSI（中规模集成电路，如加法器、比较器等）、数据通路（如ALU、存储器、总线等）、存储器甚至系统级模块（如乘法器、微控制器等）和IP核，这些逻辑单元已经布局完毕，而且设计得较为可靠，设计者可以较方便地完成系统设计。

当今ASIC的设计方向已经越来越多地使用可编程逻辑器件来构造，开发门槛和难度不断降低，流程不断简化，成本不断下降，业务也开始变得丰富且多元化。目前ASIC已经走向了深度学习、人工智能、第五代移动通信技术（5G）等高新技术领域，在可编程逻辑器件两大巨头Xilinx和Altera的推动下，可以预见未来的ASIC设计将是可编程逻辑器件（尤其是现场可编程门阵列，FPGA）的天下。

四、可编程逻辑器件

可编程逻辑器件（Programmable Logic Device, PLD）是一种通用集成电路，它是ASIC的一个子集，逻辑功能可以按照用户对器件编程来确定。一般的PLD的集成度很高，足以满足设计一般的数字系统的需要。这样就可以由设计人员自行编程而把一个数字系统“集成”在一片PLD上，而不必去请芯片制造厂商设计和制作ASIC芯片了，因为如果芯片需求量不大，设计制造ASIC的单片成本是很高的。

PLD与一般数字芯片不同的是：PLD内部的数字电路可以在出厂后才规划决定，甚至可以无限制改变，而一般数字芯片在出厂前就已经决定其内部电路，无法在出厂后再次改变，事实上一般的模拟芯片、通信芯片、微控制器也都一样，出厂后就无法再对其内部电路进行更改。最近闹得沸沸扬扬的Intel公司的芯片漏洞事件，就是因为CPU的内部电路已经无法更改，所以只能设计新的CPU芯片来解决，或是损失一些性能用软件修补的方法来弥补。

五、可编程逻辑器件的发展历程

最早的可编程逻辑器件（PLD）是1970年制成的可编程只读存储器（PROM），它由固定的与阵列和可编程的或阵列组成。PROM采用熔丝技术，只能写一次，不能擦除和重写。随着技术的发展，此后又出现了紫外线可擦除只读存储器（UVEPROM）和电可擦除只读存储器（EEPROM）。由于其价格便宜、速度低、易于编程，适合于存储函数和数据表格。

可编程逻辑阵列（PLA）于20世纪70年代中期出现，它是由可编程的与阵列和可编程的或阵列组成，但由于器件的价格比较贵、编程复杂、资源利用率低，因而没有得到广泛应用。

可编程阵列逻辑（PAL）是1977年美国MMI公司率先推出的，它采用熔丝编程方式，由可编程的与阵列和固定的或阵列组成，采用双极性工艺制造，器件的工作速度很高。由于它的设计很灵活，输出结构种类很多，因而成为第一个得到普遍应用的可编程逻辑器件。

通用阵列逻辑（GAL）是1985年Lattice公司最先发明的可电擦写、可重复编程、可设置加密位的PLD。GAL在PAL的基础上，采用了输出逻辑宏单元形式（EECMOS）工艺结构。在实际应用中，GAL对PAL仿真具有百分之百的兼容性，所以GAL几乎完全代替了PAL，并可以取代大部分标准SSI、MSI集成芯片，因而获得广泛应用。

可擦除可编程逻辑器件（EPLD）是20世纪80年代中期Altera公司推出的基于UVEPROM和CMOS技术的PLD，后来发展到采用EECMOS工艺制作的PLD，EPLD的基本逻辑单元是宏单元，宏单元是由可编程的与阵列、可编程寄存器和可编程I/O三部分组成的。从某种意义上讲，EPLD是改进的GAL，它在GAL基础上大量增加输出宏单元的数目，提供更大的与阵列，集成密度大幅提高，内部连线相对固定，延时小，有利于器件在高频下工作，但内部互连能力较弱。

复杂可编程逻辑器件（CPLD）是20世纪80年代末Lattice公司提出了在线可编程技术（SP）以后于20世纪90年代初推出的。CPLD至少包含三种结构：可编程逻辑宏单元可编程I/O单元和可编程内部连

线，它是在EPLD的基础上发展起来的，采用EECMOS工艺制作，与EPLD相比，增加了内部连线，对逻辑宏单元和I/O单元也有很大改进。

现场可编程门阵列（FPGA）器件是Xilinx公司1985年首家推出的，它是一种新型的高密度PLD，采用CMOS-SRAM工艺制作。FPGA的结构与门阵列PLD不同，其内部由许多独立的可编程逻辑模块（CLB）组成，逻辑块之间可以灵活地相互连接，CLB的功能很强，不仅能够实现逻辑函数，还可以配置成RAM等复杂的形式。配置数据存放在芯片内的SRAM中，设计人员可现场修改器件的逻辑功能，即所谓的现场可编程。FPGA出现后受到电子设计工程师的普遍欢迎，发展十分迅速。

FPGA和CPLD都具有体系结构和逻辑单元灵活、集成度高以及适用范围宽的特点。这两种器件兼容了简单PLD和通用门阵列的优点，可实现较大规模的电路，编程也很灵活，与ASIC相比，具有设计开发周期短、设计制造成本低，开发工具先进、标准产品无须测试、质量稳定等优点，用户可以反复地编程、擦除、使用，或者在外围电路不动的情况下用不同软件就可实现不同的功能以及可实时在线检验。

CPLD是一种比PLD复杂的逻辑元件。CPLD是一种用户可根据各自需要而自行构造逻辑功能的数字集成电路。与FPGA相比，CPLD提供的逻辑资源相对较少,但是经典CPLD构架提供了非常好的组合逻辑实现能力和片内信号延时可预测性，因此对于关键的控制应用比较理想。

FPGA是在PAL、GAL、EPLD等可编程器件的基础上进一步发展的产物。它是作为ASIC领域中的一种半定制电路而出现的，提供了丰富的可编程逻辑资源、易用的存储、运算功能模块和良好的性能，既解决了定制电路的不足，又克服了原有可编程器件门电路数有限的缺点。

FPGA和CPLD因为结构上的区别，各具自身特色。因为FPGA的内部构造触发器比例和数量多，所以它在时序逻辑设计方面更有优势:而CPLD因具有与或门阵列资源丰富、程序掉电不易失等特点，适用于组合逻辑为主的简单电路。总体来说，由于FPGA资源丰富功能强大，在产品研发方面的应用突出，当前新推出的可编程逻辑器件芯片主要以FPGA类为主，随着半导体工艺的进步，其功率损耗越来越小，集成度越来越高。
在微处理器系统上，软件设计师用程序设计语言控制整个系统的正常运转，而在可编程器件领域，操作的对象不再是一组组数据类型，而是一些硬件器件，如存储器，计数器等，甚至是一些更加底层的触发器、逻辑门，有的甚至要精确到集成晶体管开关级的控制。并且很多器件不再是顺序的阻塞式工作，而是并行的触发工作，经典的程序流程控制思想在可编程器件领域不适用。设计人员需要使用一种能够构造硬件电路的语言，即硬件描述语言。

六、硬件描述语言

硬件描述语言（Hardware Description Language, HDL）是一种用形式化方法描述逻辑电路和系统的语言。利用这种语言，逻辑电路系统的设计可以从上层到下层（从抽象到具体）逐层描述自己的设计思想，用一系列分层次的模块来表示极其复杂的逻辑系统。然后，利用电子设计自动化（EDA）工具，逐层进行仿真验证，再把其中需要变为实际电路的模块组合，经过自动综合工具转换到门级电路网表。接下来，再用专用集成电路（ASIC）或现场可编程门阵列（FPGA）自动布局布线工具，把网表转换为要实现的具体电路布线结构。据统计，目前在美国的硅谷约有90%以上的ASIC和PLD采用硬件描述语言进行设计。

硬件描述语言HDL的发展至今已有30多年的历史，其成功地应用于设计的各个阶段：建模、仿真、验证和综合等。到20世纪80年代，已出现了上百种硬件描述语言，对设计自动化曾起到了极大的促进和推动作用。但是，这些语言一般各自面向特定的设计领域和层次，而且众多的语言使用户无所适从。因此，需要一种面向设计的多领域、多层次并得到普遍认同的标准硬件描述语言。20世纪80年代后期至90年代，VHDL和Verilog HDL语言适应了这种趋势的要求，先后成为电气和电子工程师协会（Institute of Electrical & Electronics Engineers, IEEE）标准。

现在，随着超大规模FPGA以及包含SoC内核FPGA芯片的出现，软硬件协调设计和系统设计变得越来越重要。传统意义上的硬件设计越来越倾向于与系统设计和软件设计结合。硬件描述语言为适应新的情况，迅速发展，出现了很多新的硬件描述语言，像System Verilog，SystemC、Cynlib C++等；另一方面，PLD设计工具在原先仅支持硬件描述语言设计输入的基础上，日益增加对传统高级设计语言（如C/C++）的设计支持。

目前，硬件描述语言可谓是百花齐放，有VHDL、Verilog HDL、Superlog、SystemC、System Verilog、Cynlib C++、C Level等。整体而言，在PLD开发领域应用最广的还是VHDL和Verilog HDL。随着逻辑系统开发规模的不断增大，SystemC和System Verilog等系统级硬件描述语言也得到越来越多的应用。

VHDL

早在1980年，因为美国军事工业需要描述电子系统的方法，美国国防部开始进行VHDL的开发。1987年，IEEE将VHDL制定为标准。参考手册为IEEE VHDL语言参考手册标准草案1076/B版，于1987年批准，称为IEEE 1076-1987。然而，起初VHDL只是作为系统规范的一个标准，而不是为设计而制定的。第二个版本是在1993年制定的，称为VHDL-93，增加了一些新的命令和属性。

虽然有“VHDL是一个4亿美元的错误”这样的说法，但VHDL毕竟是1995年以前唯一制定为标准的硬件描述语言，这是它不争的事实和优势；但同时它的使用确实比较麻烦，而且其综合库至今也没有标准化，不具有晶体管开关级模拟设计的描述能力。目前来说，对于特大型的系统级逻辑电路设计，VHDL是较为合适的。

实质上，在底层的VHDL设计环境是由Verilog HDL描述的器件库支持的，因此，它们之间的互操作性十分重要。目前，Verilog和VHDL的两个国际组织OVI（Open Verilog International）、VI正在筹划这一工作，准备成立专门的工作组来协调VHDL和Verilog HDL语言的互操作性。OVI也支持不需要翻译，由VHDL到Verilog的自由表达。

Verilog HDL

Verilog HDL是在1983年，由GDA（Gateway Design AUTOMATION）公司的Phil Moorby首创的。Phil Moorby后来成为Verilog-XL的主要设计者和Cadence公司的第一合伙人。在1984-1985年，Phil Moorby设计出了第一个名为Verilog-XL的仿真器；1986年，他对Verilog HDL的发展又作出了另一个巨大的贡献：提出了用于快速门级仿真的XL算法。

随着Verilog-XL算法的成功，Verilog HDL语言得到迅速发展。1989年，Cadence公司收购了GDA公司，Verilog HDL语言成为Cadence公司的私有财产。1990年，Cadence公司决定公开Verilog HDL语言，于是成立了OVI组织，负责促进Verilog HDL语言的发展。基于Verilog HDL的优越性，IEEE于1995

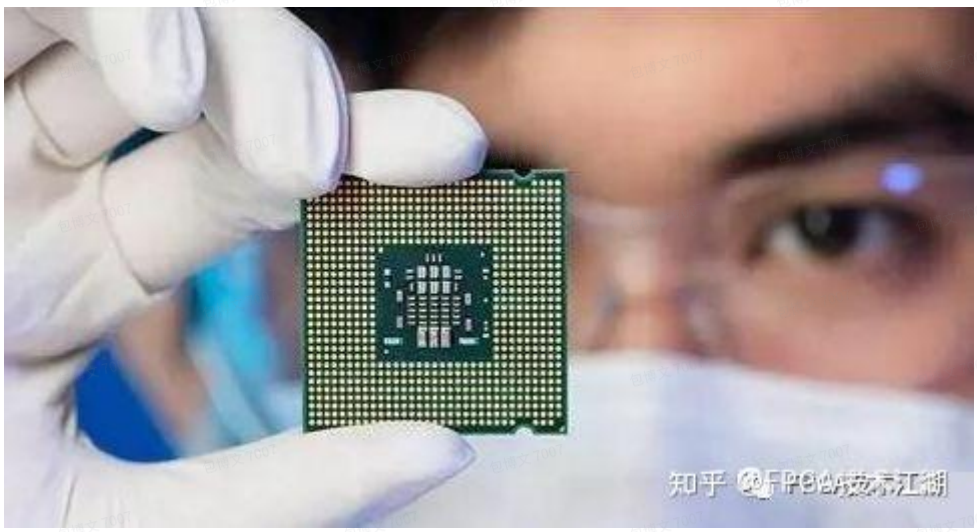
年制定了Verilog HDL的IEEE标准，即Verilog HDL 1364-1995；2001年发布了Verilog HDL 1364—2001标准，在这个标准中，加入了Verilog HDL - A标准，使Verilog HDL有了模拟设计描述的能力。

System C

随着半导体技术的迅猛发展，SoC已经成为当今集成电路设计的发展方向，智能手机，平板电脑里的处理器，严格地来说实际上是SoC，因为其上集成了CPU、图形处理单元（Graphic Processing Unit, GPU）、数字信号处理器（Digital Signal Processor）、基带（Baseband）信号处理器等。在系统芯片的各个设计（像系统定义、软硬件划分、设计实现等）中，集成电路设计界一直在考虑如何满足SoC的设计要求，一直在寻找一种能同时实现较高层次的软件和硬件描述的系统级设计语言。

System C正是在这种情况下，由Synopsys公司和Coware公司积极响应目前各方对系统级设计语言的需求而合作开发的。1999年9月27日，40多家世界著名的EDA公司、IP公司、半导体公司和嵌入式软件公司宣布成立“开放式SystemC联盟”。著名公司Cadence也于2001年加入了SystemC联盟。SystemC从1999年9月联盟建立初期的0.9版本开始更新，从1.0版到1.1版，一直到2001年10月推出了最新的2.0版。

七、常见的数据处理芯片



既然已经梳理了两大类（微处理器，专用集成电路）芯片的概念和原理，接下来就了解一下常见的芯片。

MCU

日常生活中最常见得到的微处理器系统就是我们身边的微型计算机，也就是个人电脑（Personal Computer, PC），可以使台式机、笔记本，或是PC界的新秀——各种炫酷的二合一设备。这些看起来复杂无比的电子系统都是由最简单的微处理器系统发展起来的。但是生活中并不需要那么多的电脑，比如想要做一台能够自动控制加热保温的电饭煲，其CPU性能可能只需要电脑这样的大家伙的九牛一毛即可，也不需要复杂的输入输出设备，在设计上大可以大刀阔斧地将用不到的部分砍掉，灵活地将CPU、时钟发生器（Clock）、随机存储器（Random Access Memory, RAM）、只读存储器（Read-Only Memory, ROM）和需要的外部设备集成起来小型化，这种经过大改观的微处理器系统，其所有部分都集成在了一块芯片上，称为微控制器或单片机（Micro Controller Unit, MCU）。

目前MCU是应用最广泛的一种电子控制芯片，其控制程序可以由特殊的烧录工具下载到ROM中，行使系统的功能。这些ROM可以是PROM、UVEPROM、EEPROM等，若MCU上没有集成ROM，也可以外接ROM。按照系统结构，微处理器系统可以分为冯·诺依曼结构（也称普雷斯顿结构）和哈佛结构，其区别是程序与数据的存放方式不同，同样地，MCU芯片也可以分为这两种结构，灵活地满足需要。

MPU

微处理器单元（Micro Processor Unit, MPU），就是把很多CPU集成在一起并行处理数据的芯片。通俗来说，MCU集成了RAM，ROM等设备；MPU则不集成这些设备，是高度集成的通用结构的中央处理器矩阵，也可以认为是去除了集成外设的MCU。

PLD（CPLD/FPGA）

因为目前广泛使用的PLD是CPLD和FPGA，因此把这两种芯片作为例子介绍。前面已经介绍过，CPLD/FPGA的内部结构和CPU完全不同，内部电路可以被多次修改，可以按照用户的编程形成不同的组合逻辑电路、时序逻辑电路结构，是一种“万能”的芯片，CPLD/FPGA看起来像一个CPU，其实不然，因为使用CPLD/FPGA实现控制是纯硬件实现，实质上和使用成千上万基本逻辑门搭建的数字逻辑电路没有区别。

因此可以直接用HDL编程在CPLD/FPGA里搭建出一个“CPU”（有时还有硬盒和软核之分，限于篇幅，不再赘述），再做好相应的I/O、总线，就是一个简单的微处理器系统了。但是这样一来，又变成了软件控制，PLD的硬件控制优势荡然无存。故CPLD/FPGA经常和实际的CPU搭配使用，在CPLD/FPGA上编写一些较复杂算法的运算电路，当CPU处理到这些复杂任务时，就交由CPLD/FPGA进行处理，处理结束以后再将结果返回给CPU，提高控制系统的整体性能。

ADC、DAC

自然界的物理量分为模拟（Analog）量和数字（Digital）量两种。模拟量在一定范围内的取值是连续的，个数是无穷的；数字量在一定范围内的取值是离散的，个数是有限的。计算机只能处理离散的数字量，所以模拟信号必须经过变换才能交由计算机处理。将自然界的物理量转化为连续变化的电流或电压（故称“模拟”），在满足奈奎斯特采样定理（Nyquist Sampling Theory，也称香农采样定理，Shannon Sampling Theory）的条件下采样，得到时域离散信号，再经量化器（可以是线性量化和非线性量化）量化后数字信号，最后经过一道编码得到二进制的0、1数字信息，才能交由计算机处理。以上的这一道变换称为模数转换（A/D），可以将这部分电路集成到一块芯片上，这就是模数转换电路（Analog Digital Circuit, ADC）,相应的也有数模转换（D/A）和数模转换电路（Digital Analog Circuit, DAC）芯片，进行D/A的时候同样要在数学和信息论上满足相关定理。

DSP

数字信号处理器（Digital Signal Processor, DSP）是用来高速处理数字信号的专用芯片。

经过ADC转化好的数字信号，数据量往往很庞大，直接交由CPU处理的效率是不高的，并且CPU还要进行更多的通用计算的任务。因此，常常采用专用的电路来处理数字信号，如数字滤波、快速傅里叶变换、时频分析、语音信号和图像信号的处理加工等。这些运算往往很复杂，很多涉及复数的累加、累乘运算，举个例子：离散傅里叶变换的计算就十分复杂，但是运用时域抽取或频域抽取的快速傅里

叶变换算法后就可以大大减少运算量，但是电路较为复杂。将能完成这些复杂运算的电路集成在一块芯片上，能在一个时钟周期完成一次乘加运算，使其能完成如基2-FFT蝶形运算、音频滤波、图像处理等复杂运算，这样的芯片叫做DSP。

DSP也是一种特殊的CPU，特别适合信号的处理，如3G中的Node B就大量使用了DSP进行信号处理。DSP对于流媒体的处理能力远远的优于CPU，现在手机上的语音信号都是由DSP处理的。现阶段DSP的概念正在变得模糊，如ARM9的架构就不像是一颗CPU，更像是一颗DSP。现在有很多芯片，其上都集成了DSP，GPU，基带处理器等，越来越多的传统上分立的芯片被集成到一起，协同工作以提高效率，降低能耗，这也是未来的一个趋势。

SoC

随着半导体技术、移动互联网和智能终端的迅猛发展，传统的微处理器系统的发展已经跟不上时代的潮流，现代信息技术迫切地需要一种功能多，性能强，功耗低的芯片来满足越来越多的智能设备的需求。SoC便应运而生。

SoC的全称是System on a Chip，顾名思义，就是在一块芯片上集成一整个信息处理系统，称为片上系统或系统级芯片。这个定义现在也不尽明确，因为不同用途的SoC上集成的部件是不一样的，一般说来，SoC是一个完整的整体，已经拥有了整个数字系统的完整功能它也是一种ASIC，其中包含完整的控制系统并有嵌入式的软件。

SoC也代表着一种技术，是一种以确定系统功能为目标，各个模块的软硬件协同开发，最后把开发成果集成为一块芯片的技术。由于功能丰富，又要求有不俗的性能发挥，SoC已然是功能最为丰富的硬件，其上集成了CPU、GPU、RAM、ADC/DAC、Modem、高速DSP等各种芯片，有的SoC上还必须集成电源管理模块，各种外部设备的控制模块，充分考虑各总线的分布利用……现如今，智能手机里的SoC上就集成了以上的部件和基带处理器等很多相关的通信模块。

SoC的电路相比于传统的微处理器系统更加复杂，其对设计和制造工艺的要求自然更上一层楼，对软硬件协同开发的依赖性相当高。迄今为止，在半导体行业首屈一指的企业才有自主设计制造SoC的能力，目前在性能和功耗敏感的终端芯片领域，SoC已占据主导地位，人们每天使用的手机里面，就有一颗颗性能强劲，永远在线的SoC在为我们服务。就连传统的软件大厂微软也推出了基于高通公司的骁龙835平台的Windows操作系统；而且SoC的应用正在扩展到更广的领域，SoC在无人机技术、自动驾驶，深度学习等行业也有越来越多的应用，用一块单芯片就能实现完整的电子系统，是半导体行业、IC产业未来的发展方向。