清華大學
Tsinghua University

第一单元　第五讲

运算器部件组成及设计

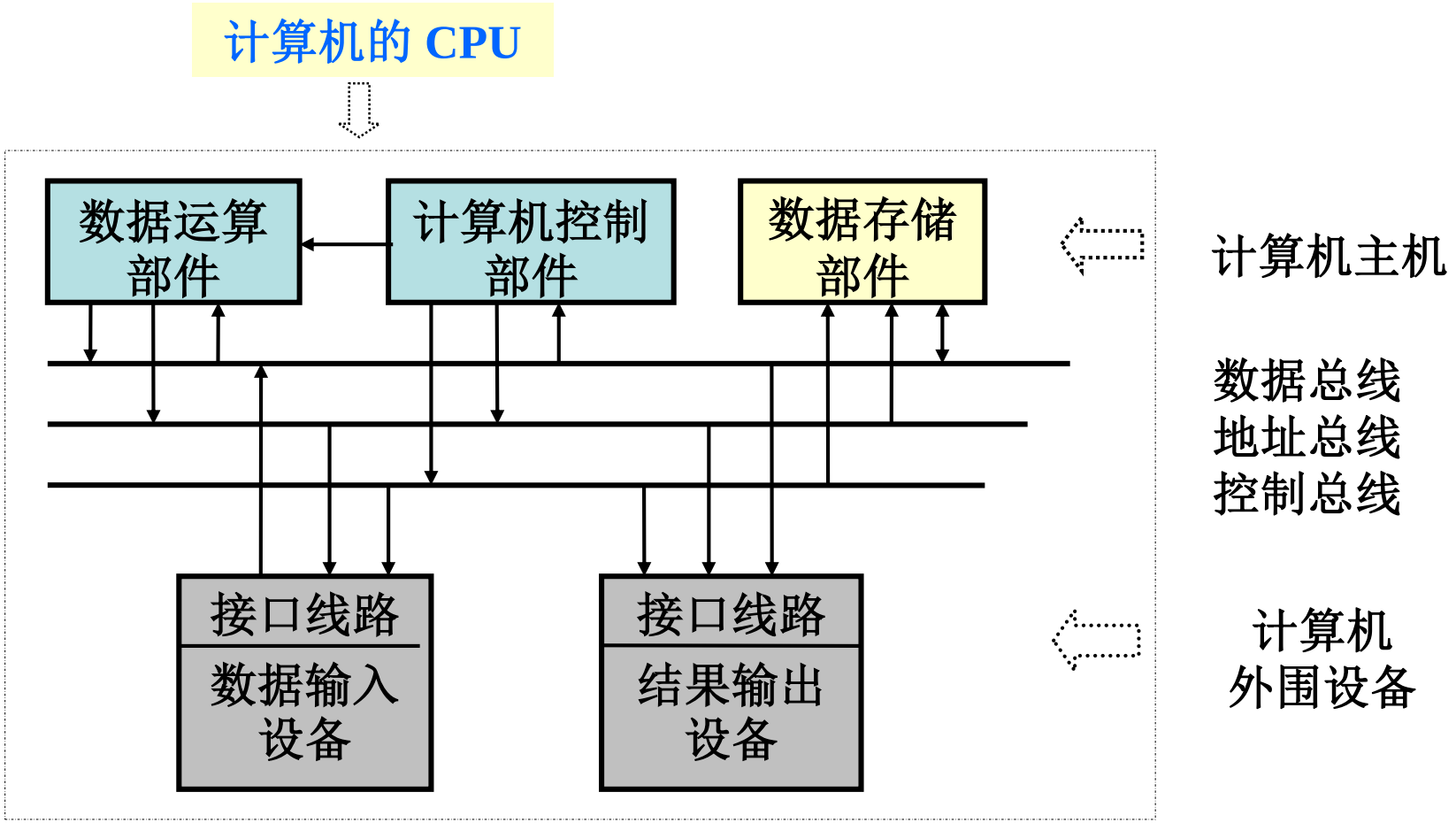
刘卫东

计算机科学与技术系

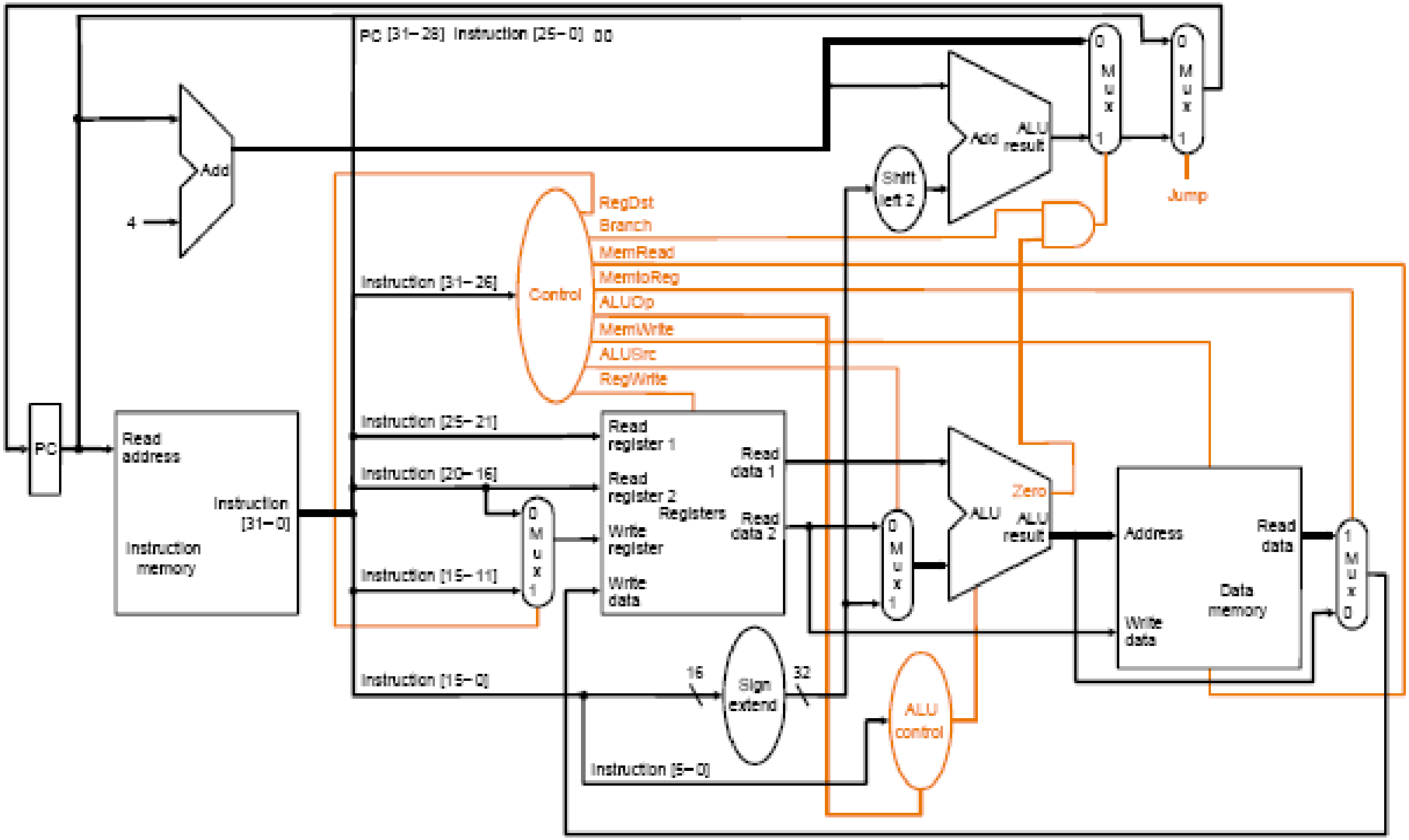
本讲概要

- 运算器功能与组成概述
- 定点运算器实例Am2901
- 用VHDL描述Am2901

硬件系统的功能部件


计算机的 CPU
数据运算部件
计算机控制部件
数据存储部件
计算机主机
数据总线
地址总线
控制总线
接口线路
数据输入设备
接口线路
结果输出设备
计算机外围设备

CPU示例


PC [31–28] Instruction [25–0] 00
Add
4
PC
Read address
Instruction [31–0]
Instruction memory
Instruction [31–26]
Control
RegDst
Branch
MemRead
MemtoReg
ALUOp
MemWrite
ALUSrc
RegWrite
Instruction [25–21]
Instruction [20–16]
Instruction [15–11]
Instruction [15–0]
Instruction [5–0]
Read register 1
Read register 2
Registers
Write register
Write data
Read data 1
Read data 2
16
Sign extend
32
Shift left 2
Add
ALU result
M u x
0
1
Jump
ALU
Zero
ALU result
ALU control
Address
Read data
Data memory
Write data

运算器的基本功能

- 完成算术、逻辑运算，产生运算结果

  ALU执行 +、—、×、÷、∧、∨、¬

  并给出运算结果的状态信息

  C、Z、V、S

- 暂存运算所用操作数

  寄存器组、立即数、数据总线

- 暂存运算的中间结果

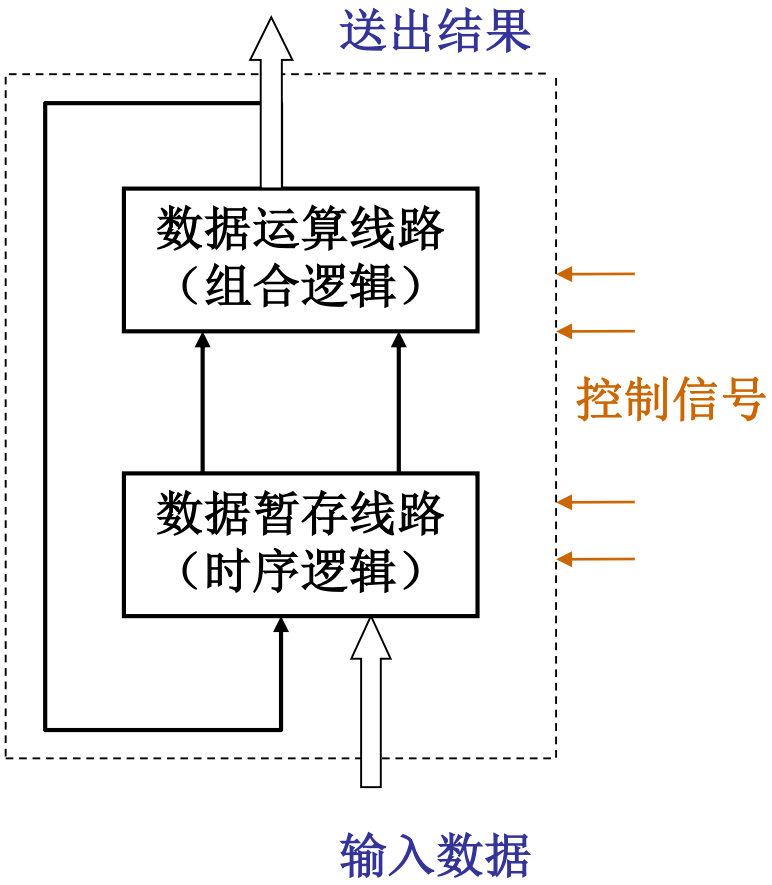
  寄存器组、Q寄存器、移位线路

- 输出运算结果

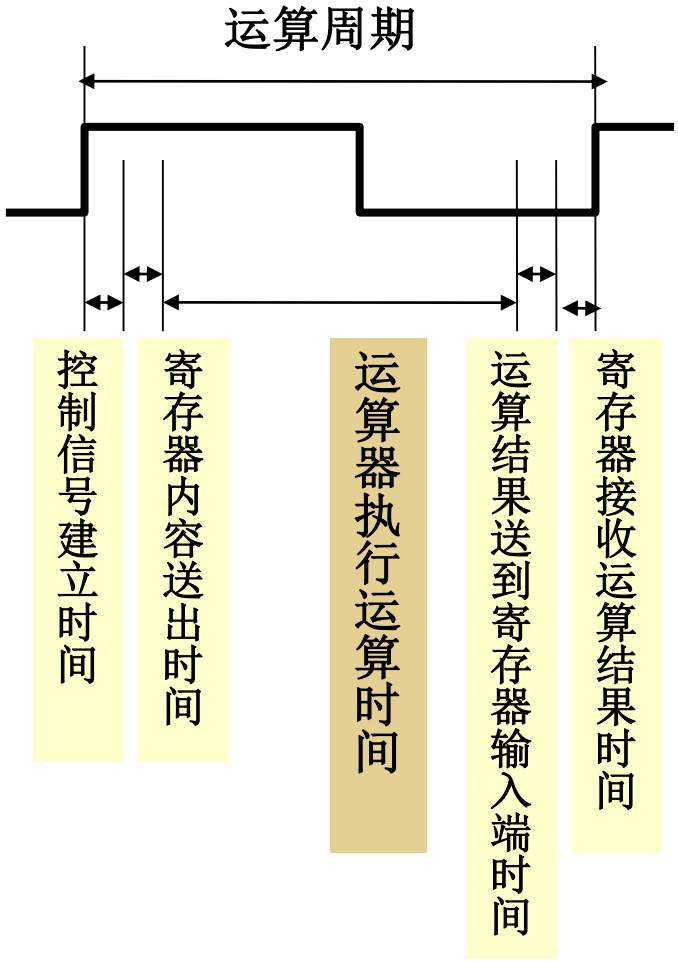
  寄存器组、数据总线

运算器是计算机系统中执行数据运算、处理的功能部件，类似于一个工厂中的生产加工车间，由ALU和寄存器组等组成

执行一次运算的时序关系



运算器组成示意图



一个运算周期中各时间段示意图

两种不同类型的运算器

运算器包括 定点运算器 和
浮点运算器 两种类型

定点运算器:

主要完成对整数类型数据的算术运算、逻辑类型数据的逻辑运算

浮点运算器:

主要完成对浮点类型数据的算术运算，也用于完成长整数、BCD等类型的数据运算

定点运算器功能与组成

- 完成算术与逻辑运算功能
  - 算术逻辑单元（ALU）
- 暂存参加运算的数据和中间结果
  - 通用寄存器组
- 乘除法运算的硬件支持线路
  - 乘商寄存器（Q寄存器）
- 接收数据输入和送出运算结果

通过几组多路选择器电路实现相互连接，以便数据传送

- 作为处理机内部数据通路（Data Path）

ALU的实现方案

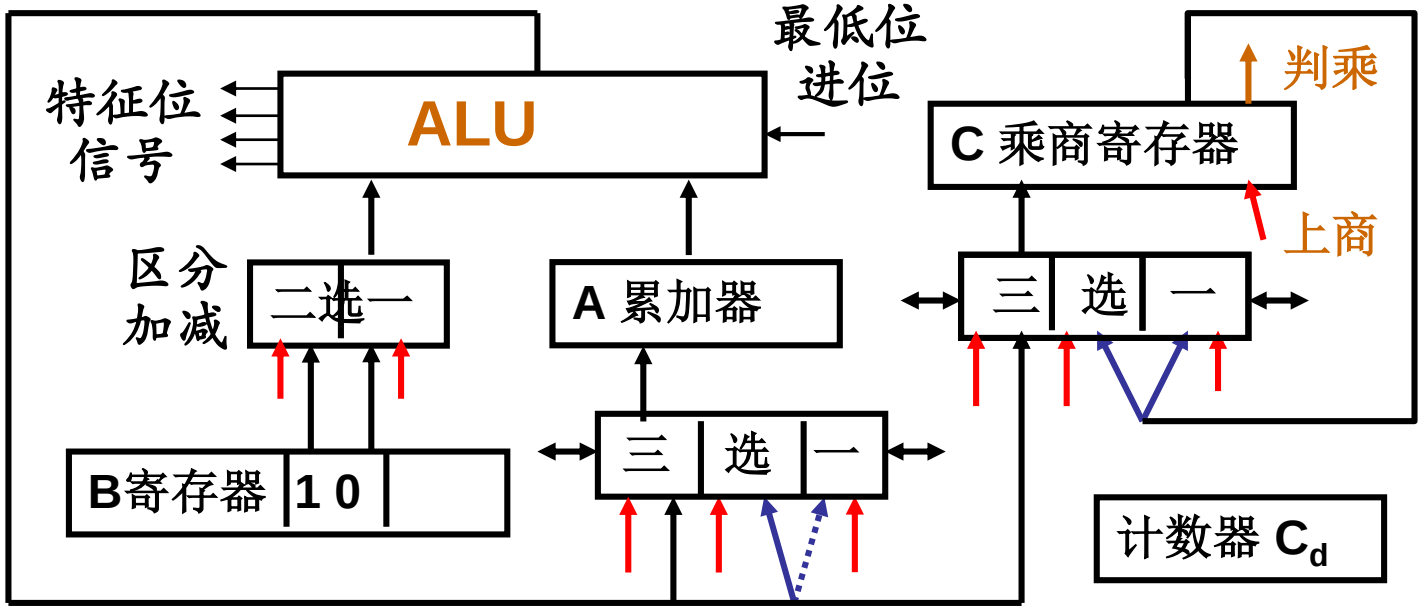
- ALU用于执行2路数据的算术和逻辑运算
  例如：+、—、×、÷、∧、∨ 等
- 在第4讲已经讲过实现这几种运算的算法
  例如：补码加减、逻辑运算、原码一位乘除等

还给出了实现这些运算的原理性线路框图

A存放：
被加数
高位积
被除数
逻辑数1

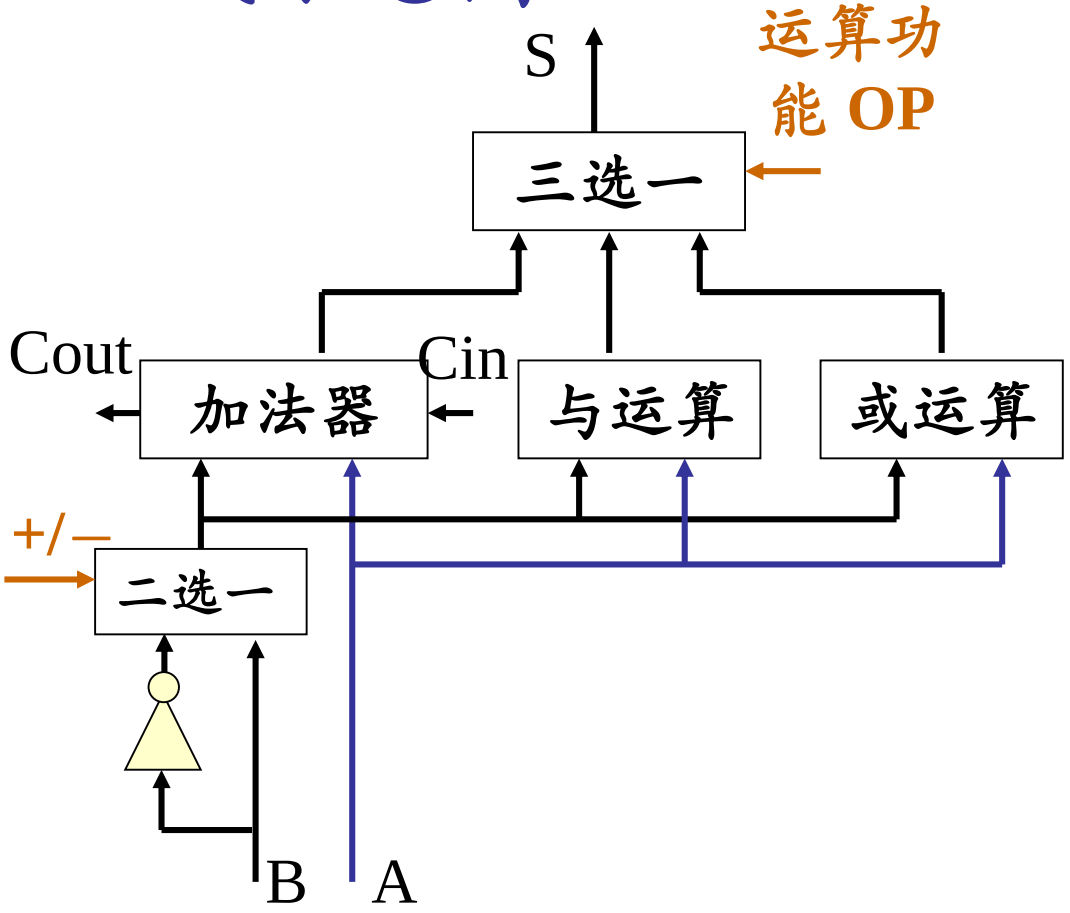
B存放：
加数
被乘数
除数
逻辑数2

乘除运算
用加减和
移位多次
迭代完成



设计实现 ALU 线路

一位的ALU的线路设计：1、首先画出其组成示意图



输入数据 A、B

加减进位输入 Cin

加减进位输出 Cout

3种运算用3部分电路

运算结果S，3选1得到

运算功能选择输出

加/减选择 B 或 /B

设计实现 ALU线路

2、接着写出功能的真值表

OP	A	B	Cin	S	Cout
0 0	0	0	0	0	0
0 0	0	1	0	1	0
0 0	1	0	0	1	0
0 0	1	1	0	0	1
0 0	0	0	1	1	0
0 0	0	1	1	0	1
0 0	1	0	1	0	1
0 0	1	1	1	1	1

(加法)

OP	A	B	S (与)
1 0	0	0	0
1 0	0	1	0
1 0	1	0	0
1 0	1	1	1

OP	A	B	S (或)
1 1	0	0	0
1 1	0	1	1
1 1	1	0	1
1 1	1	1	1

设计实现 ALU线路

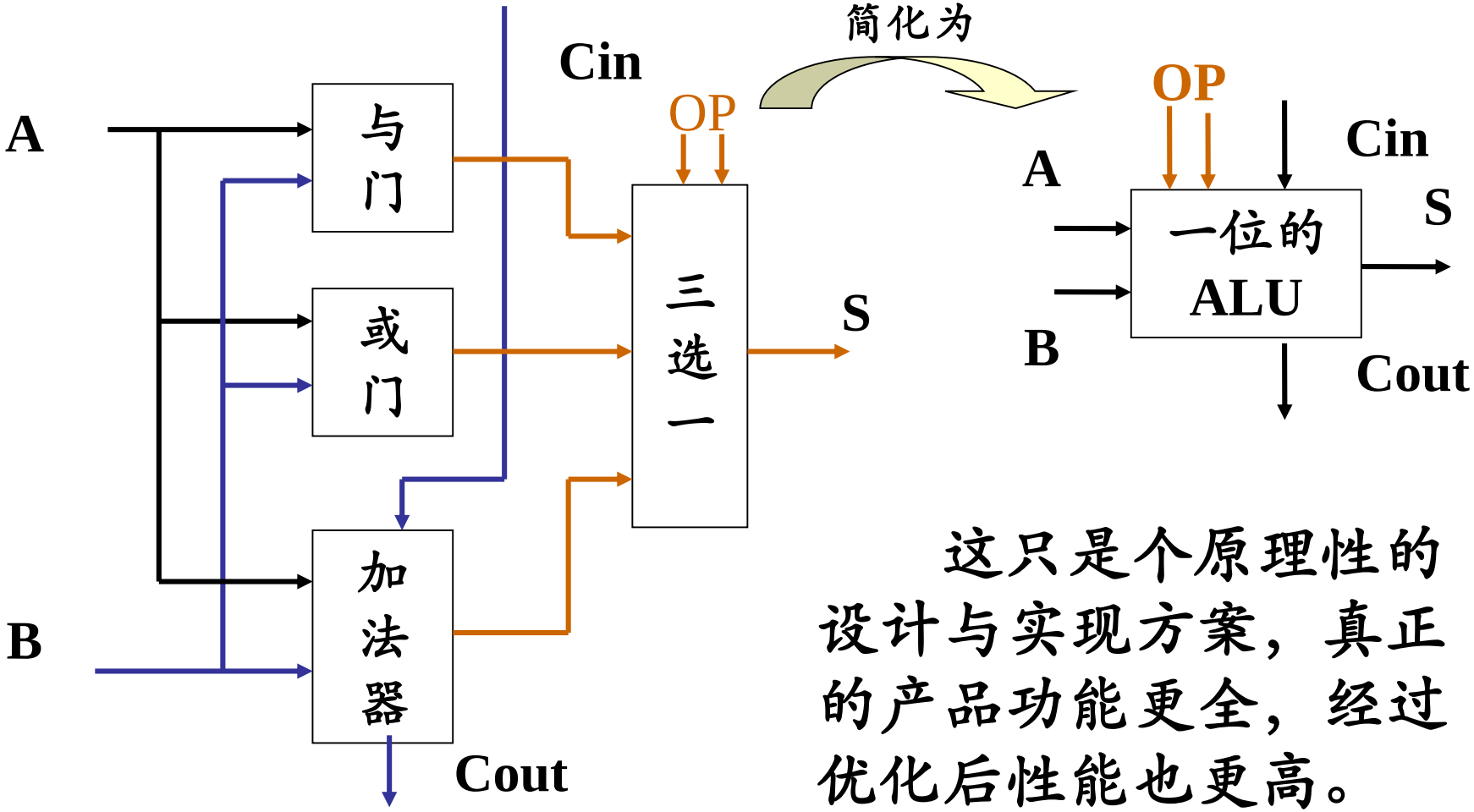
3、按真值表写逻辑表达式

$$
\begin{aligned}
S = {} & (/OP1 \cdot /OP0) \cdot (A \cdot /B \cdot /Cin \quad \text{加法运算} \\
& + /A \cdot B \cdot /Cin \\
& + /A \cdot /B \cdot Cin \\
& + A \cdot B \cdot Cin) \\
& + (OP1 \cdot /OP0) \cdot (A \cdot B) \quad \text{与运算} \\
& + (OP1 \cdot OP0) \cdot (A + B) \quad \text{或运算}
\end{aligned}
$$

$$
\begin{aligned}
Cout = {} & (/OP1 \cdot /OP0) \cdot (A \cdot B \quad \text{加法运算} \\
& + A \cdot Cin \\
& + B \cdot Cin)
\end{aligned}
$$

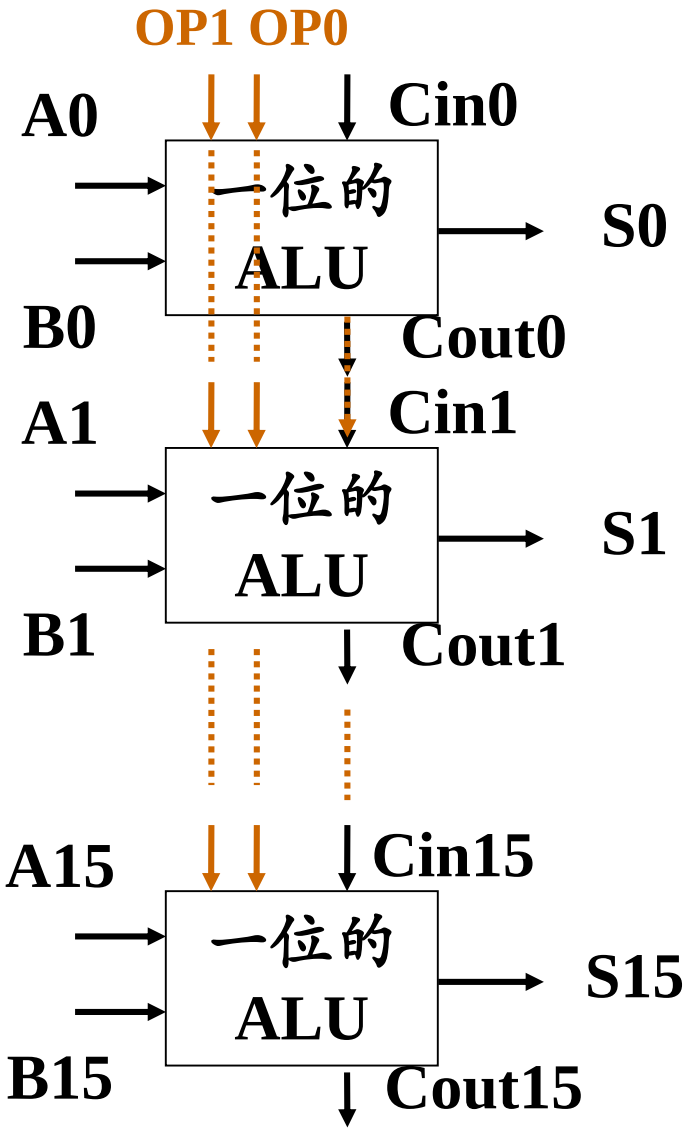
设计实现 ALU线路

4、按逻辑表达式画电路图



这只是个原理性的设计与实现方案，真正的产品功能更全，经过优化后性能也更高。

设计实现 16位ALU



用16个一位的ALU实现一个16位的ALU，使其并行执行算逻运算，使每位ALU处在不同的数位上，信号名加位序号。

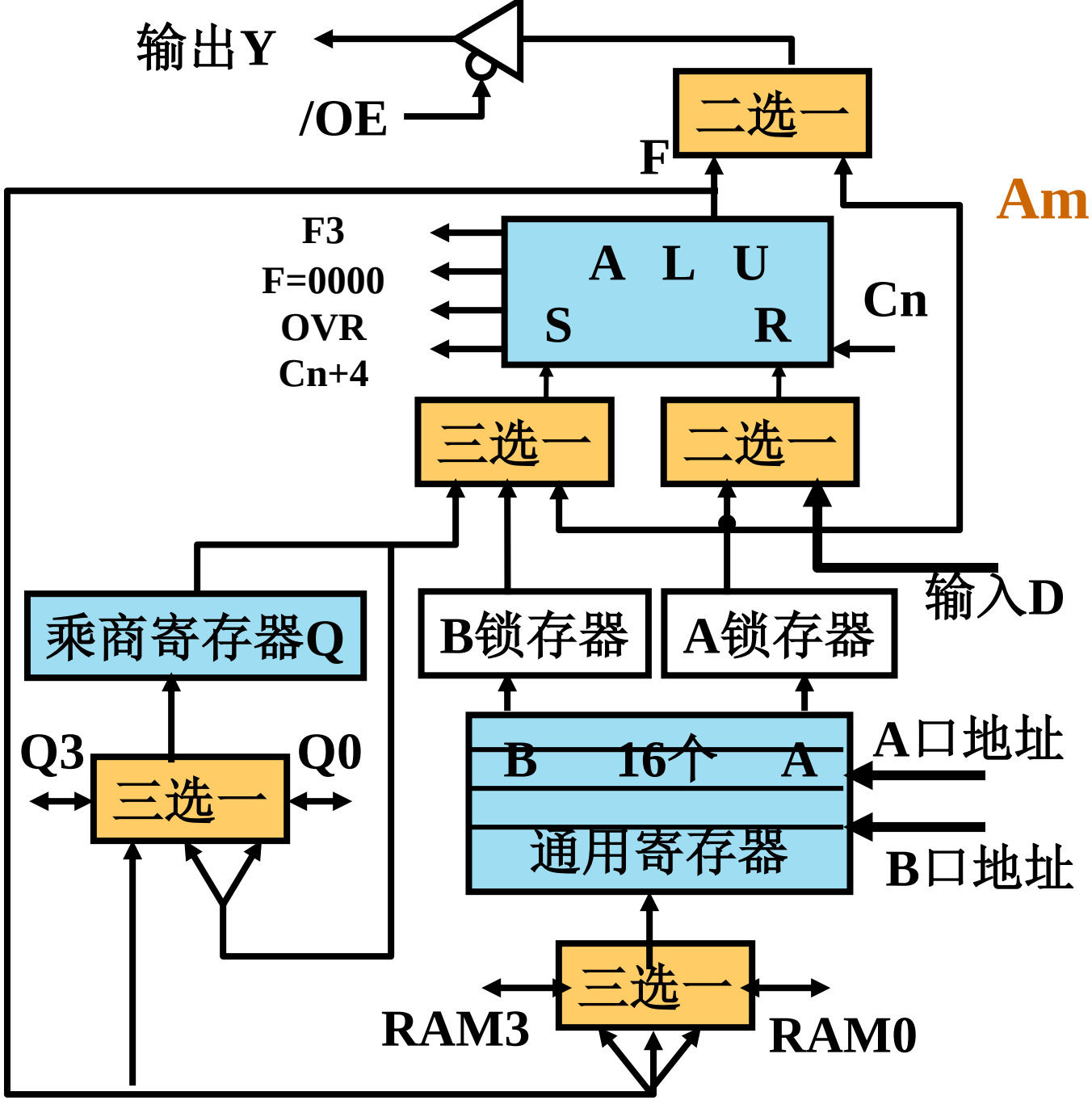
需要把每一位上的ALU的进位输出信号和相邻高位的进位输入信号正确连接；并把全部位上的控制信号连接在一起，使它们同步执行相同的运算功能，从而对两个16位的数据执行正确的运算功能，产生正确的结果。

还需要解决进位信号传送速度，可采用超前进位的方案。

位片结构运算器Am2901

清華大學 Tsinghua University

- AMD公司1975年推出Am2900系列
- 4位位片结构运算器
- 可实现3种算术运算和5种逻辑运算
- 应用广泛
  - Nova、DEC、TEC
- 作为例子来了解运算器的基本实现
  - 了解运算器（Datapath）和控制器的相互关系
  - 为设计CPU打好基础



清華大學 Tsinghua University

Am2901内部组成

组成

算逻运算部件

通用寄存器组

乘商寄存器 Q

功能

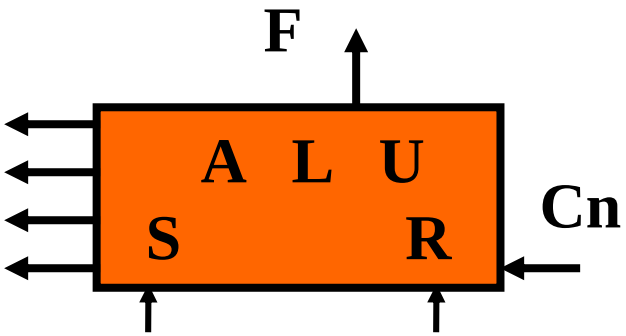
8种运算功能

8种数据组合

8种结果处理

符号位
结果为零
结果溢出
进位输出

F3
F=0000
OVR
Cn+4



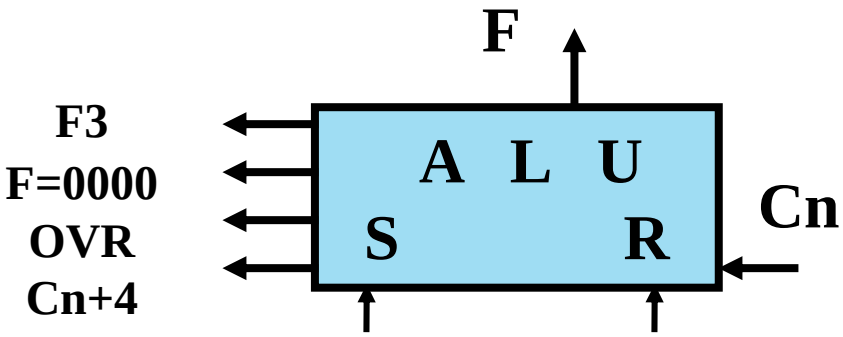
3 位功能选择码　　8 种运算功能

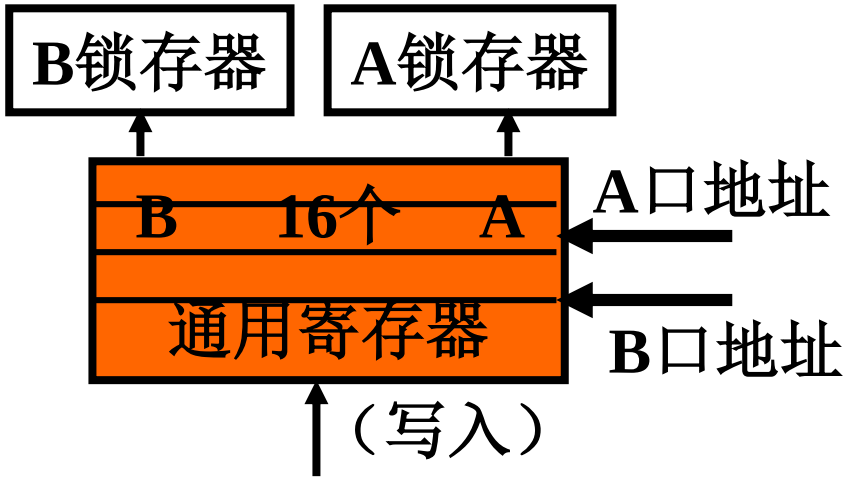
000	R+S
001	S–R
010	R–S
011	R∨ S
100	R∧S
101	$\overline{R}$∧S
110	R⊕S
111	$\overline{R⊕S}$

Am2901芯片是一个 4位的位片结构的运算器器件,其内部组成如下：

第一个组成部分是算逻运算部件ALU,完成 3 种算术运算和　5 种逻辑运算，共计 8 种功能。

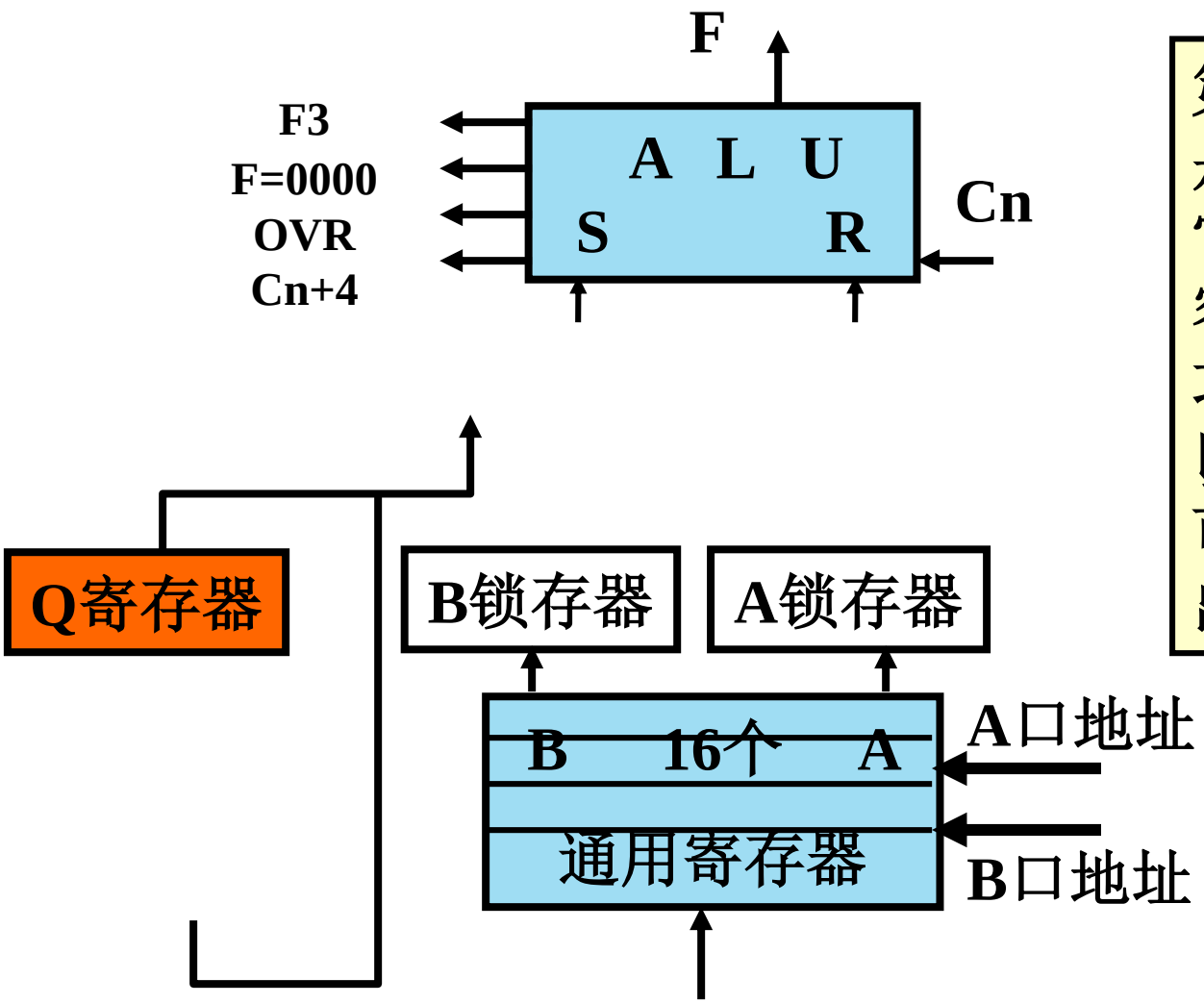
其输出为 F，两路输入为 S、R，最低位进位Cn，4个状态输出信号如图所示



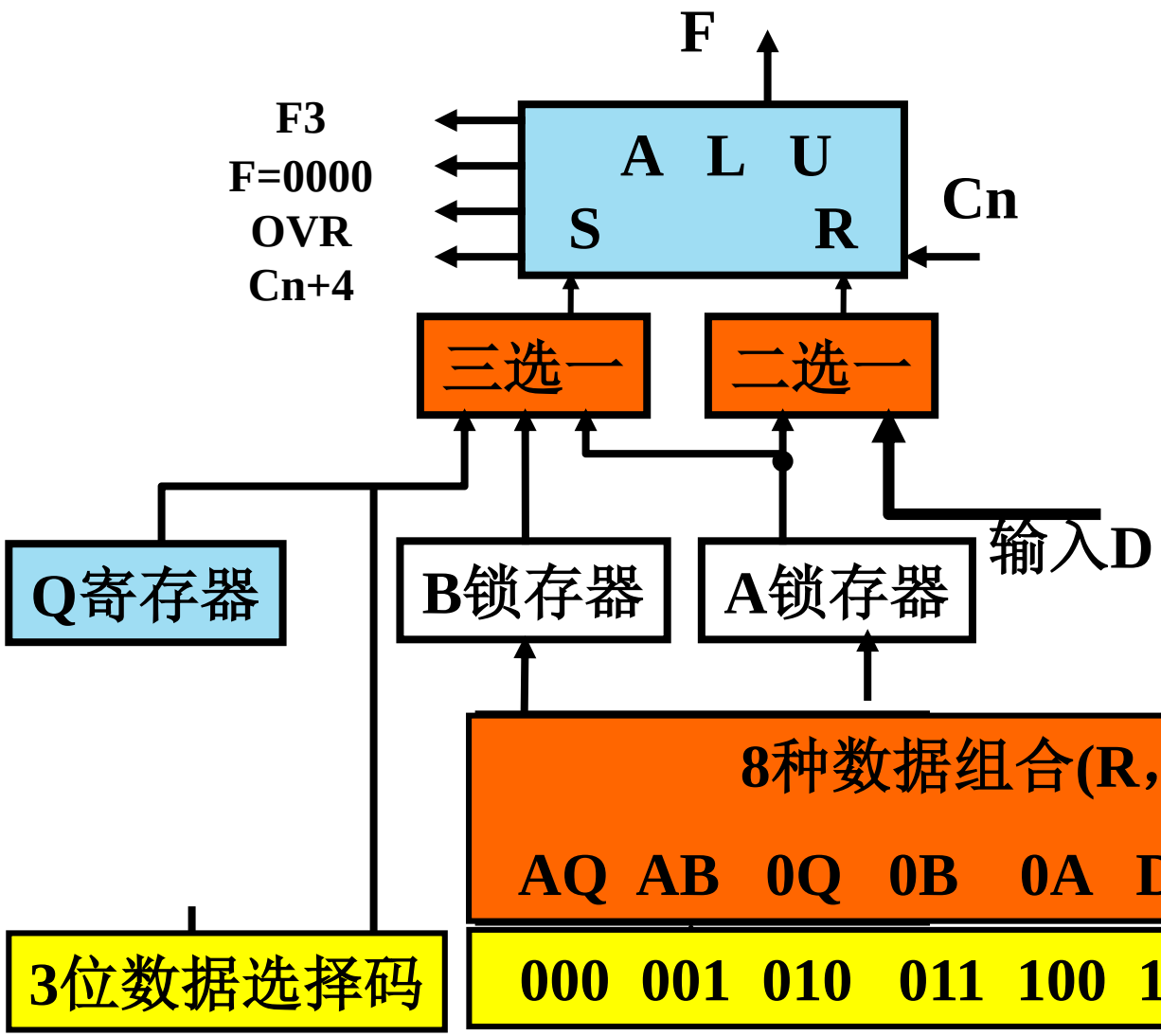


第二个组成部分是通用寄存器组由16个寄存器构成，并通过B口与A口地址选择被读的寄存器，B口地址还用于指定写入寄存器

通过B口地址、A口地址读出的数据将送到B、A锁存器，要写入寄存器的数据由一个多路选择器送来。

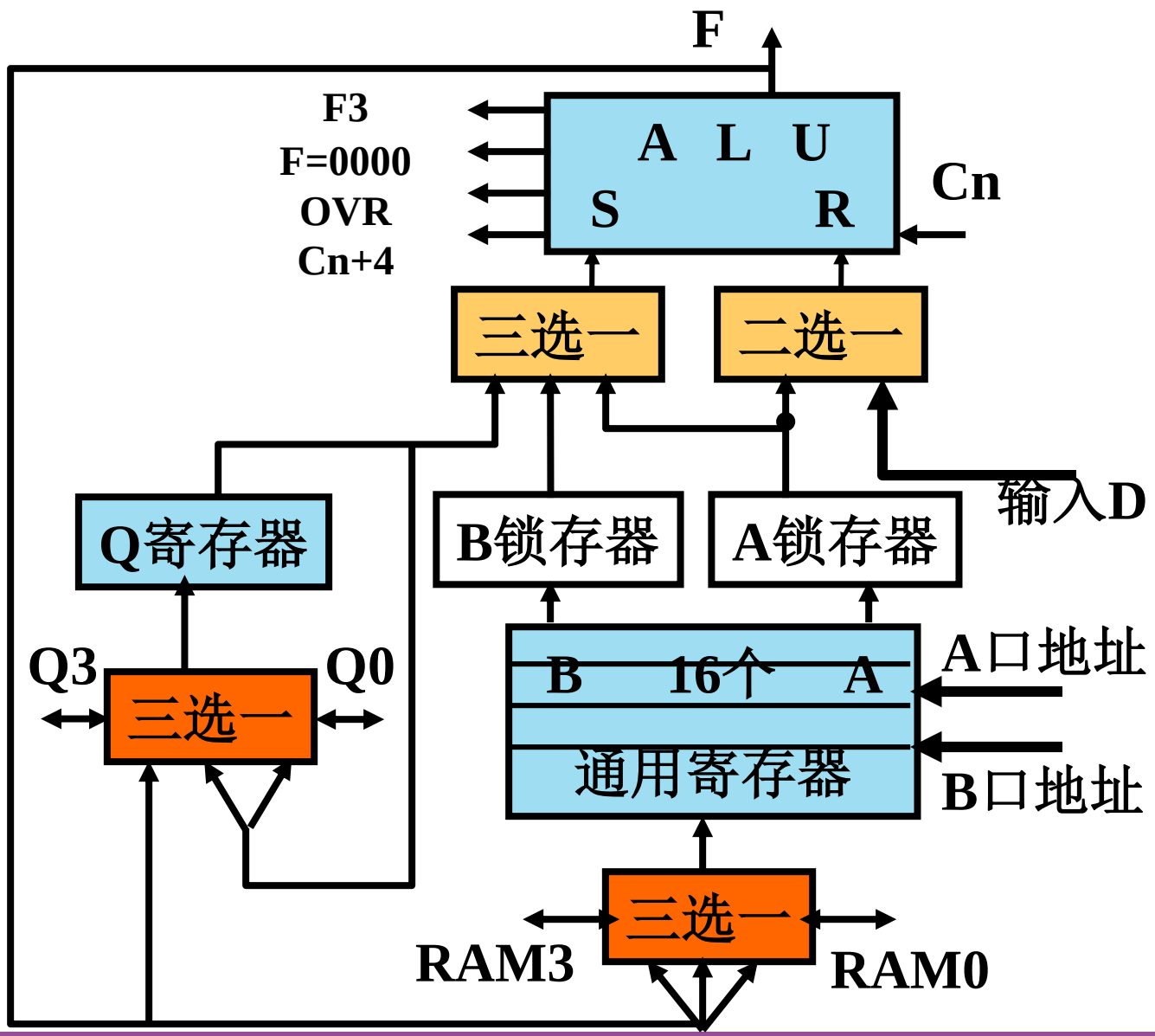


第三个组成部分是乘商寄存器Q它能对自己的内容完成左右移位功能，其输出可以送往ALU，并可接收ALU的输出结果。



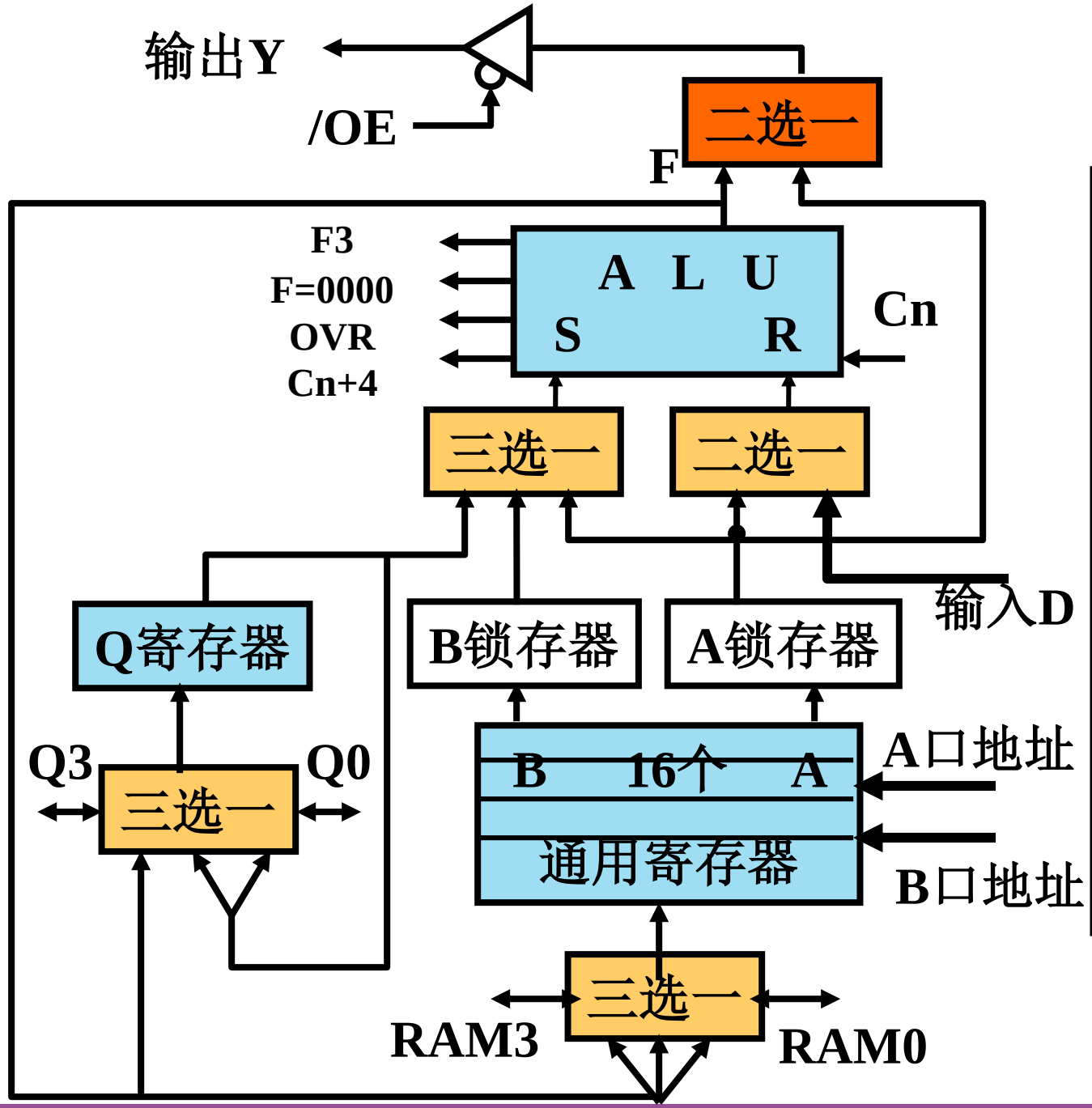
一组三选一门和另一组二选一门用来选择送向ALU的 R、S输入端的数据来源，包括Q寄存器、A口、 B口、外部输入D数据的8 种不同组合。

第四个是 5 组门,包括



一组三选一门完成把ALU的输出、或右移一位、或左移一位的值送往通用寄存器组，最高、最低位移位信号有双向入/出问题

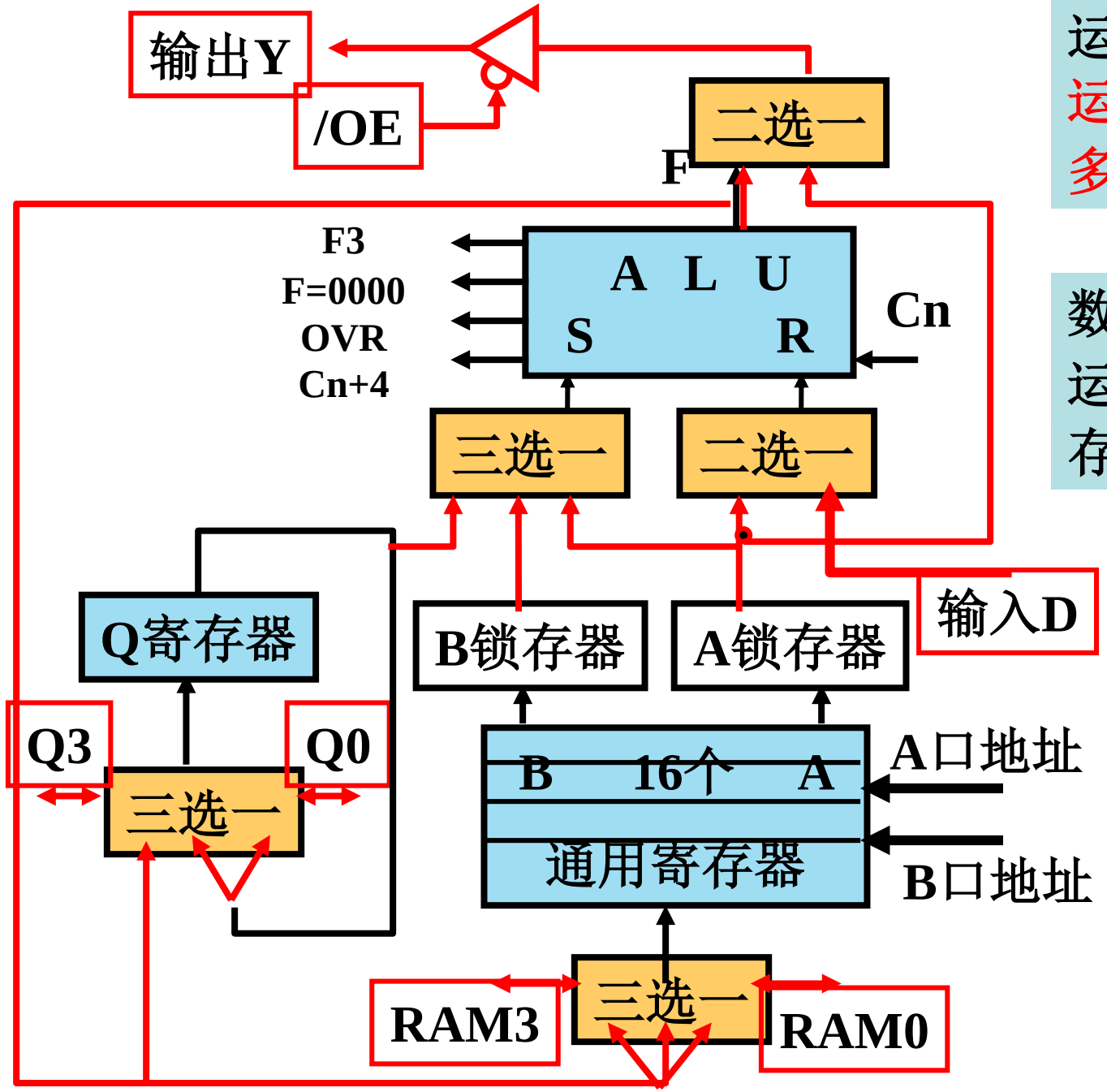
一组三选一门完成Q寄存器的右移一位、或左移一位、或接收ALU输出值的功能，最高、最低位移位信号有双向入/出问题。



一组二选一门，选择把A口数据或ALU结果送出芯片，以给出输出Y的数据，Y输出的有无还受输出使能 /OE信号的控制，仅当/OE为低是才有Y输出，/OE为高，Y输出为高阻态。

8 种结果处理

3位控制码	通用寄存器	Q寄存器	Y输出
000		Q←F	F
001			F
010	B←F		A
011	B←F		F
100	B←F/2	Q←Q/2	F
101	B←F/2		F
110	B←2F	Q←2Q	F
111	B←2F		F



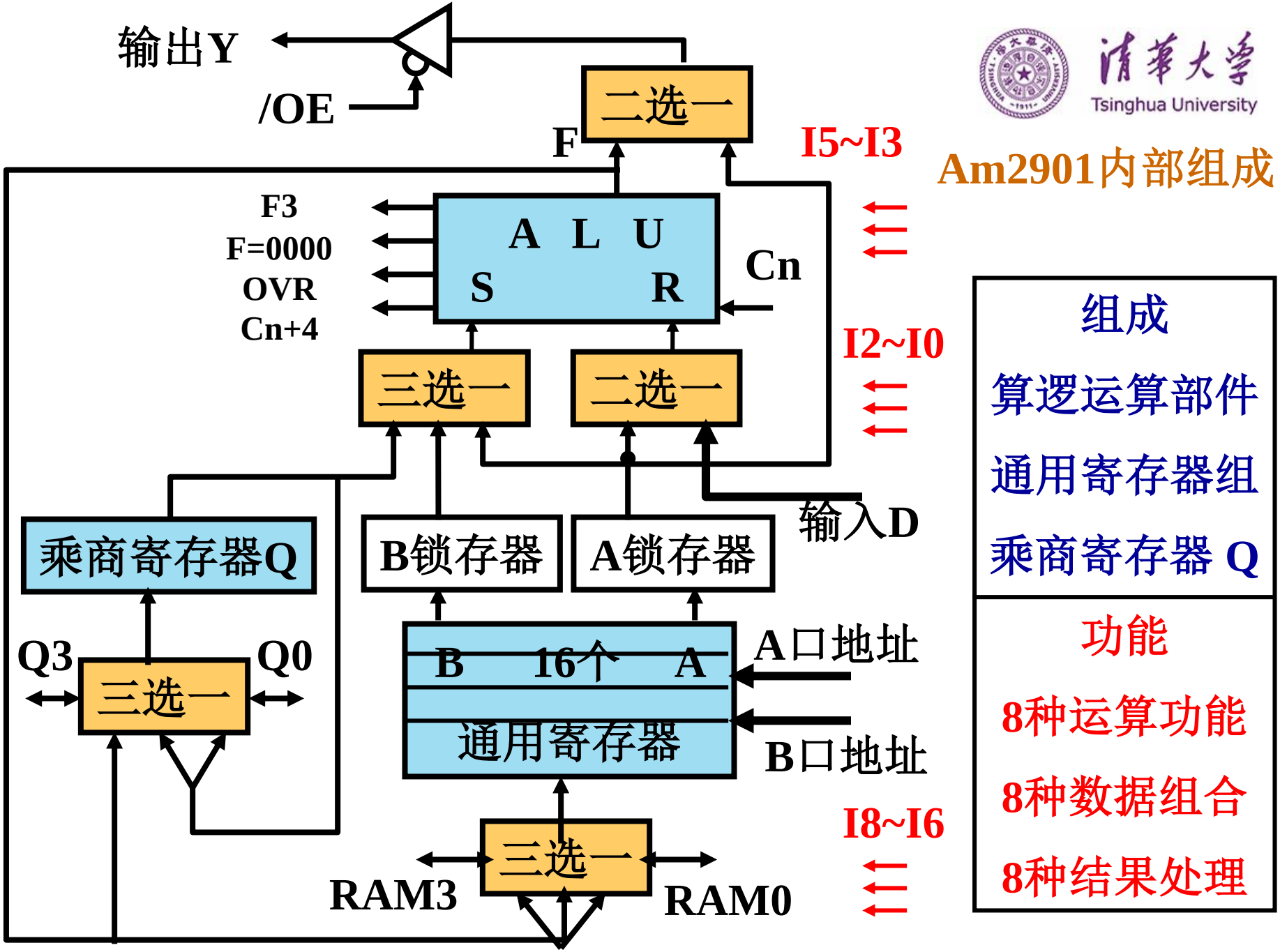
运算器，三大件
运算 暂存 乘除快
多路选通连起来

数据组合有内外
运算功能指明白
存移输出巧安排

运算功能选择
← I5 I4 I3

数据组合选择
← I2 I1 I0

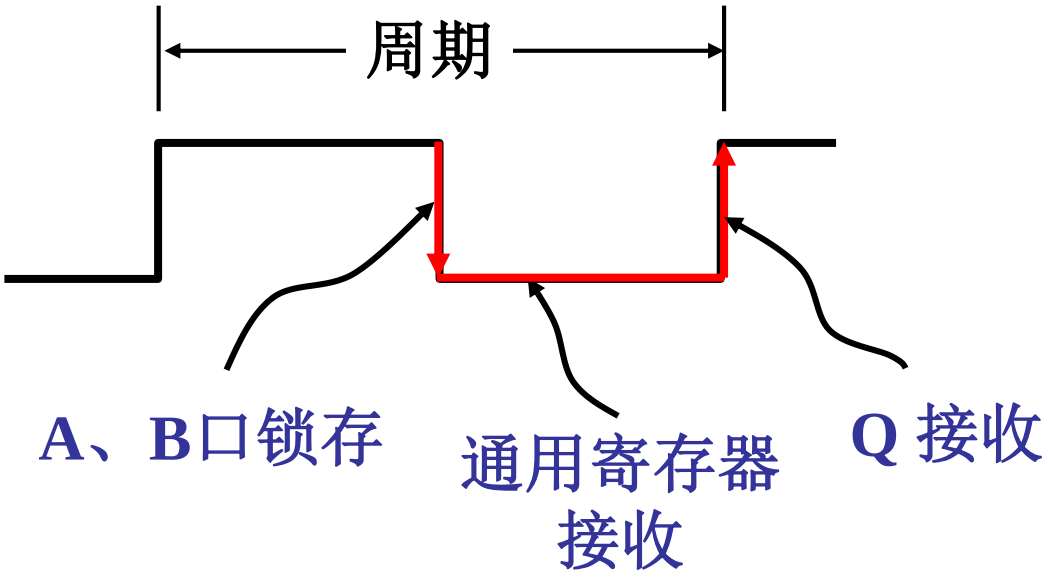
运算结果处理
← I8 I7 I6


输出Y
/OE
二选一
F
I5~I3
Am2901内部组成
F3
F=0000
OVR
Cn+4
A L U
S R
Cn
三选一
二选一
I2~I0
输入D
乘商寄存器Q
B锁存器
A锁存器
Q3
Q0
三选一
B 16个 A
通用寄存器
A口地址
B口地址
I8~I6
三选一
RAM3
RAM0
组成
算逻运算部件
通用寄存器组
乘商寄存器 Q
功能
8种运算功能
8种数据组合
8种结果处理

运算器的时钟信号 CP

时钟脉冲信号 CP 用于控制寄存器和锁存器的动作时间，注意两个跳变沿和低电平的作用。

锁存器，在时钟脉冲的高电平期间，其输入信号直接“穿”过线路送到其输出端，用脉冲下降沿把输入信号存储起来用于后续的输出。



通用寄存器是 R_S触发器，低电平期间接收输入。

Q寄存器是 D 型 触发器，用脉冲上升沿接收输入。

Am2901控制信号汇总表

清華大學 Tsinghua University

编码	I8 I7 I6			I5 I4 I3	I2 I1 I0	
	B	Q	Y		R	S
000		Q← F	F	R + S	A	Q
001			F	S− R	A	B
010	B← F		A	R− S	0	Q
011	B← F		F	R∨ S	0	B
100	B← F/2	Q← Q/2	F	R∧S	0	A
101	B← F/2		F	$\overline{R}$∧S	D	A
110	B← 2F	Q← 2Q	F	R⊕S	D	Q
111	B← 2F		F	$\overline{R⊕S}$	D	0

Am2901操作使用表

操作功能	控制信号					
	B口	A口	I8 I7 I6	I5 I4 I3	I2 I1 I0	Cn
R0←R0+R1 Y← F	0000	0001	011	000	001	0
R2←R2−R0 Y← A口	0010	0000	010	001	001	1
右移 R0←R0+R1 Y← F	0000	0001	101	000	001	0
Q←R0 Y← F	/	0000	000	000	100	0
R0←R0∧R1 Y← A口	0000	0001	010	100	001	0
R0←R0∧R1 Y← F	0000	0001	011	100	001	0

VHDL语言简介

- 硬件描述语言HDL
  - Hardware Description Language
  - IEEE 国际标准：VHDL， Verilog
- HDL打破软、硬件的界限
  - 传统的数字系统设计分为：
    - 硬件设计（硬件设计人员）
    - 软件设计（软件设计人员）
  - HDL是硬件设计者和 EDA工具之间的界面

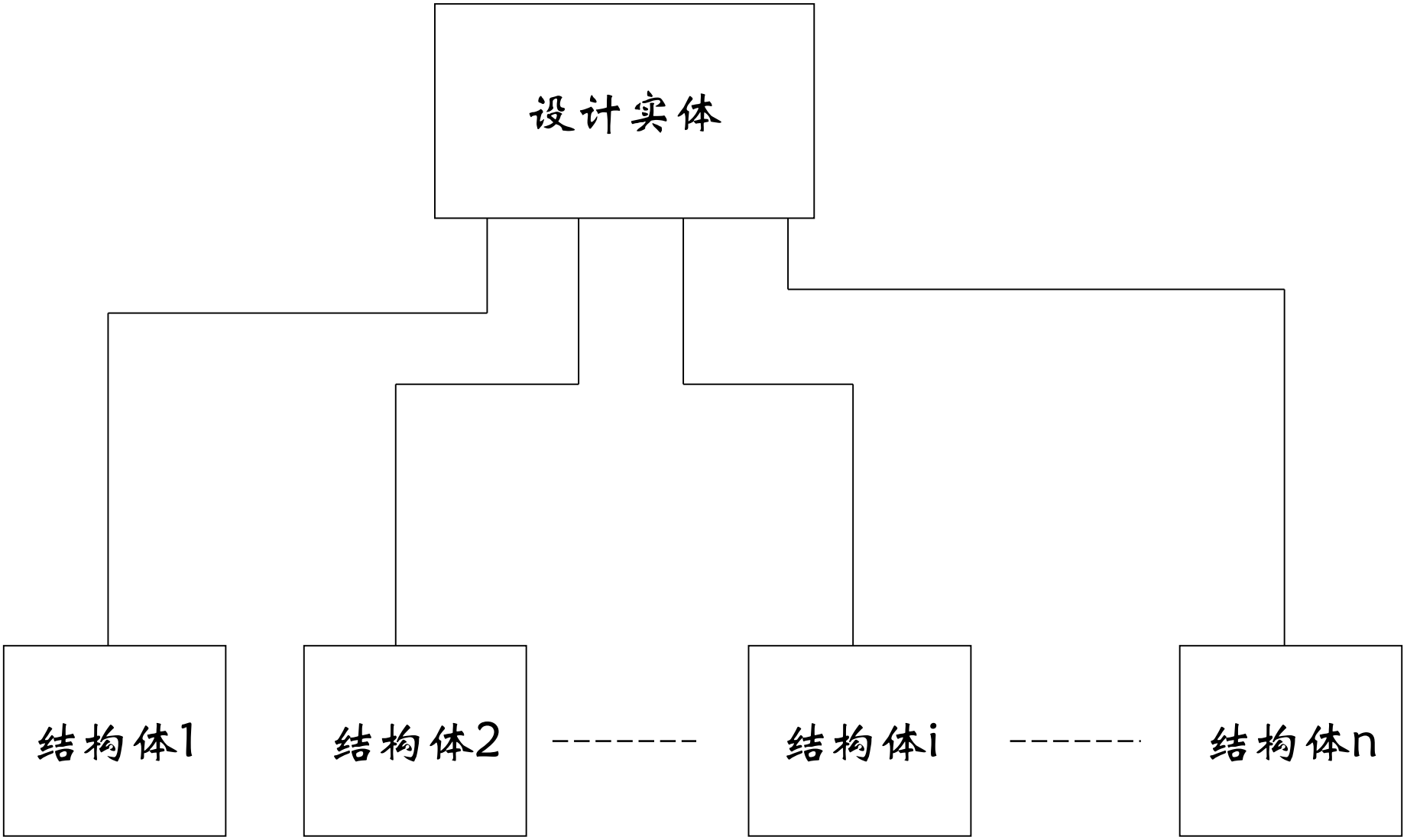
VHDL特点

- VHDL具有强大的语言结构
- 系统硬件描述能力强、设计效率高
- 具有较高的抽象描述能力
- 可读性强，易于修改和发现错误
- 移植性好

基本结构

- 模块化 ——以实体为描述对象
- 层次化 ——元件例化，复合语句
- 行为描述——进程为最小单位
- 灵活组合——编译单元，配置

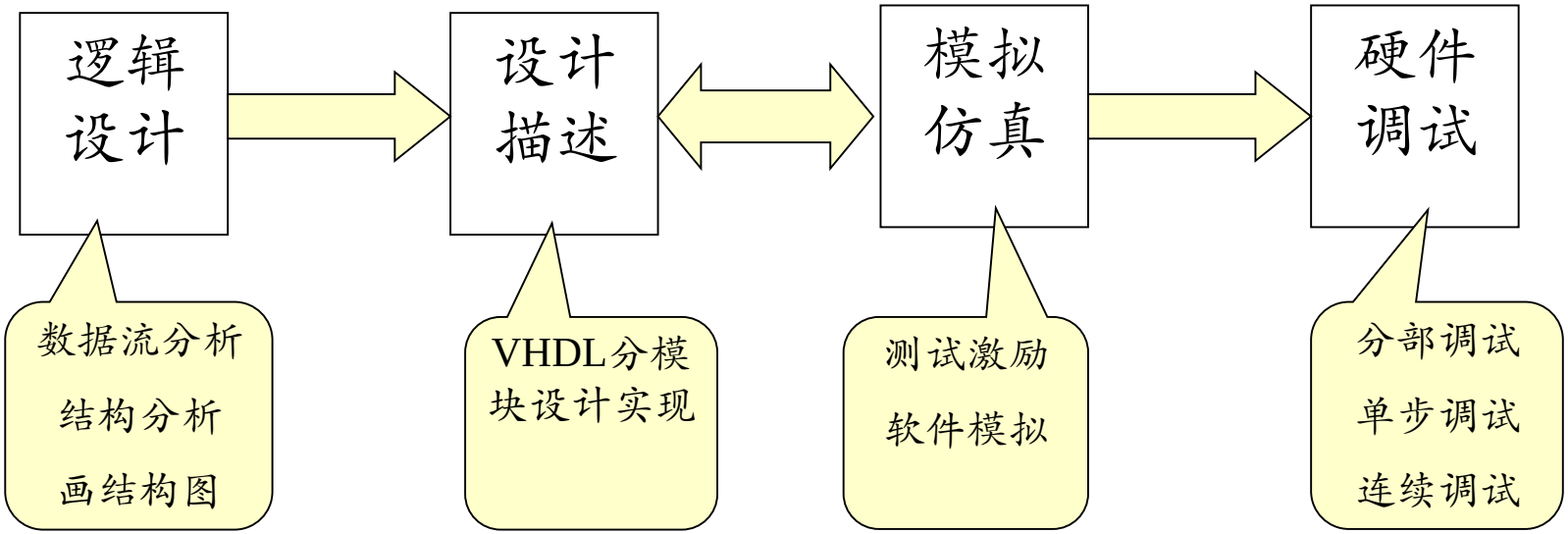
基本设计单元

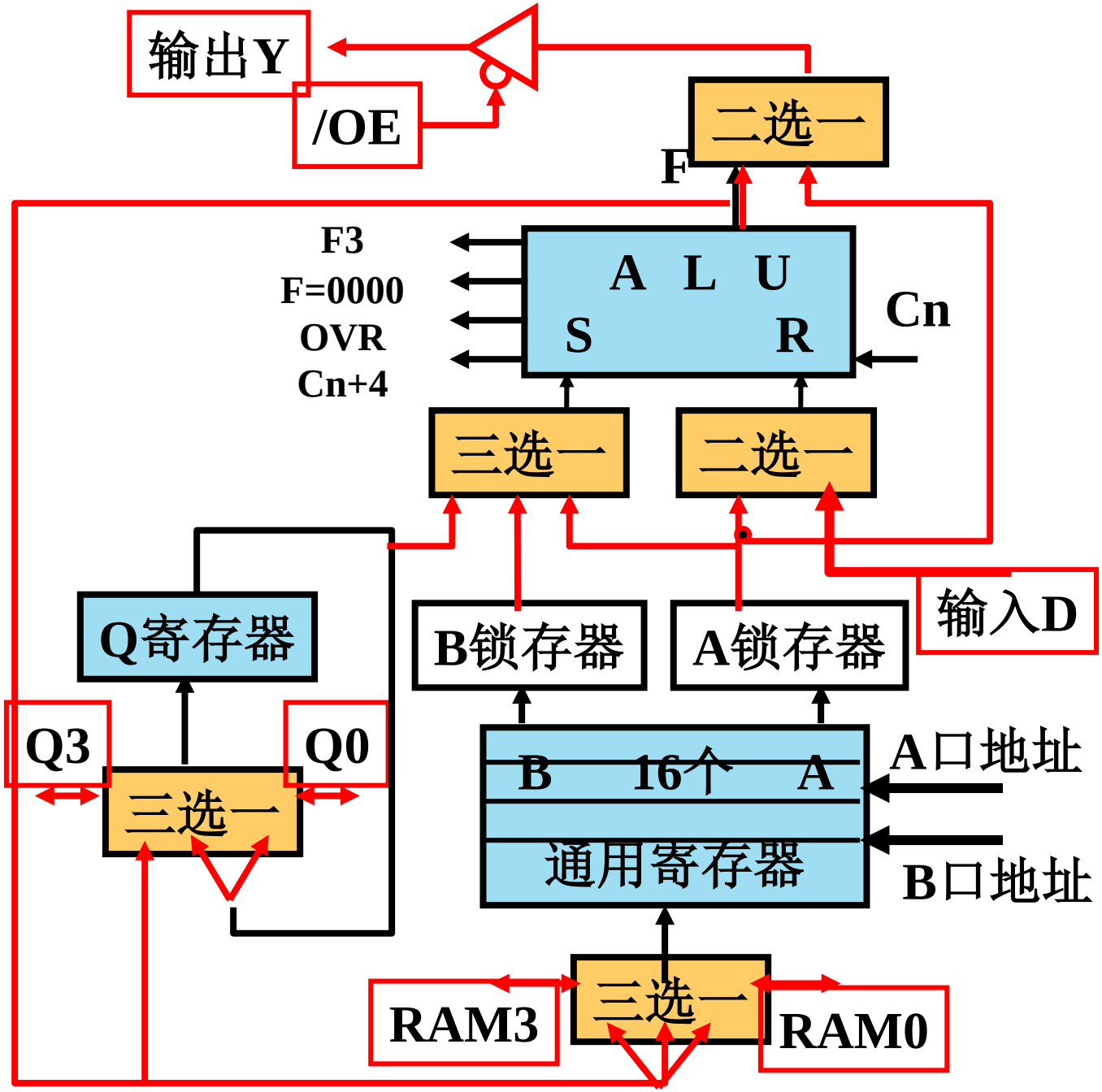


用VHDL设计Am2901

- 逻辑设计
  - 功能分析
- 设计描述
  - 自顶向下设计方法
  - 划分模块
    - ALU、Register Files（读/写）、A和B锁存器
- 模拟仿真
- 硬件调试
  - 分部调试
  - 单步调试
  - 连续调试

设计流程


逻辑设计
设计描述
模拟仿真
硬件调试
数据流分析
结构分析
画结构图
VHDL分模块设计实现
测试激励
软件模拟
分部调试
单步调试
连续调试


输出Y
/OE
二选一
F
F3
F=0000
OVR
Cn+4
A L U
S R
Cn
三选一
二选一
输入D
Q寄存器
B锁存器
A锁存器
Q3
Q0
三选一
B 16个 A
通用寄存器
A口地址
B口地址
三选一
RAM3
RAM0

Am2901实体设计

```
ENTITY  Am2901  is              --在实体部分说明这个部件的输入输出信号的属性和数据类
PORT (CP                : IN  STD_LOGIC;                --时钟信号
      D                 : IN  STD_LOGIC_VECTOR(3 DOWNTO 0);  --4位的输入数据信号
      Cin               : IN  STD_LOGIC;                --最低位进位输入信号
   ADDR_A,ADDR_B  : IN  STD_LOGIC_VECTOR(3  DOWNTO 0); --A、B口的寄存器编号
      I8_6,I5_3,I2_0 : IN  STD_LOGIC_VECTOR(2  DOWNTO 0); --3组3位的控制信号

      RAM0_,RAM3_,Q0_,Q3_ : INOUT STD_LOGIC;          --双向的移位入出信号
      Cy,Over,Zero,Sign   : OUT STD_LOGGIC;           --输出的各标志位信号
       Y : OUT STD_LOGIC_VECTOR(3 DOWNTO 0)    --4位的输出数据信号
      );
END;
```

定义了Am2901的外部接口

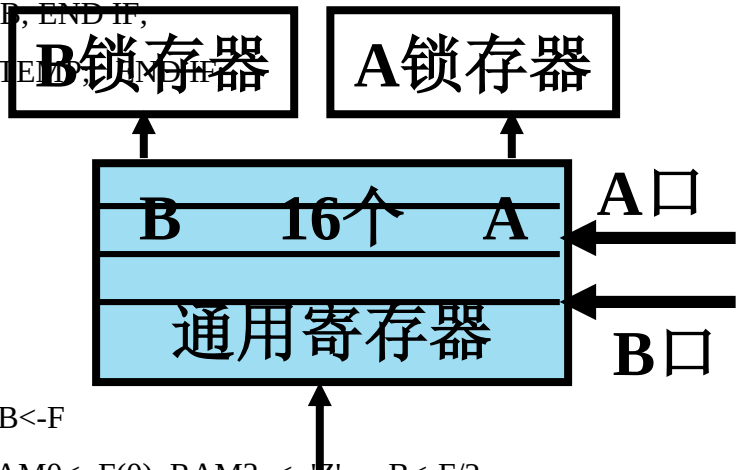
Am2901结构体定义

```
ARCHITECTURE behave OF Am2901 is      --说明本部件的用到的变量与信号
  VARIABLE  A_ADDR,B_ADDR :INTEGER RANGE 0 TO 15;       --寄存器编号的取值范围
                          --说明运算器内部使用的电路及其涉及到的连接关系
  SIGNAL MUL_R_SEL,MUL_S_SEL,MUL_Q_SEL,MUL_REG_SEL :
                                          STD_LOGIC_VECTOR(1 DOWNTO 0);
  SIGNAL R,S,F,A,B,Q,REG_A,REG_B,Q_IN,REG_IN,A_TEMP,B_TEMP :
                                          STD_LOGIC_VECTOR(3 DOWNTO 0);
  SIGNAL Q3_OUT,Q0_OUT,RAM0_OUT,RAM3_OUT : STD_LOGIC;
   SIGNAL  REG_W,Q_W      : IN  STD_LOGIC;                --REG 和Q的写命令
  subtype register_type is std_logic_vector(3 downto 0);
  type    register_heap is array (0 to 15) of register_type;
  signal  regs : register_heap;      --说明由16个寄存器组成的寄存器组
```

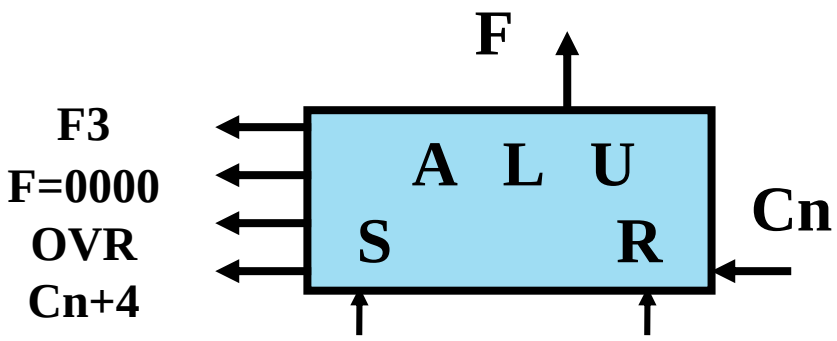
定义了Am2901的内部信号

```
A_ADDR := CONV_INTEGER(UNSIGNED(ADDR_A)); --变换寄存器编号为整数类型
 B_ADDR := CONV_INTEGER(UNSIGNED(ADDR_B));
 REG_A <= REG_ARRAY(ADDR_A);          --用给出的 2 个寄存器编号,
 REG_B <= REG_ARRAY(ADDR_B);          --取出 2 个寄存器的内容
PROCESS(CP)          --锁存器A、B的锁存控制与输出
 BEGIN
   IF CP'EVENT AND CP='0' THEN A_TEMP<=REG_A;  B_TEMP<=REG_B; END IF;
   IF CP='1' THEN  A<=REG_A; B<=REG_B;  ELSE  A<=A_TEMP; B<=B_TEMP; END IF;
 END PROCESS;

 PROCESS(CP)          --选择写入寄存器组的数据，完成写入操作
 BEGIN
    CASE  I8_6 IS       --选择写入的数据
        WHEN "010" OR "011" => REG_IN<=F;                              --B<-F
        WHEN "100" OR "101" => REG_IN<=RAM3_ & F(3 DOWNTO 1); RAM0<=F(0); RAM3_<='Z';  --B<-F/2
        WHEN "110" OR "111" => REG_IN<=F(2 DOWNTO 0) & RAM0_; RAM3<=F(3); RAM0_<='Z';  --B<-2F
        WHEN  OTHERS=>NULL;
    END CASE;
    IF CP='0' THEN       --如果 REGs 允许写，则在时钟的低电平期间完成写入
      IF REG_W='1' THEN REG_ARRAY(B_ADDR)<=REG_IN;  END IF;
    END IF;
 END PROCESS;
```



```
WITH I5_3 SELECT F    --选择 ALU 的运算功能并完成运算
                R+S+CIN       WHEN "000",
                S-R-CIN       WHEN "001",
                R-S-CIN       WHEN "010",
                R OR S        WHEN "011",
                R AND S       WHEN "100",
                NOT(R) AND S   WHEN "101",
                R XOR S       WHEN "110",
                NOT(R XOR S)   WHEN "111",
                (OTHERS =>'Z') WHEN OTHERS;
 PROCESS Flag_bit(CP)    --产生 ALU 运算结果的标志位信息
  CASE I5_3 IS      --只有加减运算影响进位输出 Cy 和溢出 Over 信号
              WHEN "000"=>Cy  <=(S(3) and R(3)) or(R(3) and not F(3))or (not F(3) and S(3));   - -加法运算
                   Over<=(S(3) and R(3) and not F(3)) or (not S(3) and not R(3) and F(3);
              WHEN "001"=>Cy  <= (S(3)and not R(3)) or (not R(3)and not F(3))or (not F(3)and S(3));   --减法运算
                   Over<= (S(3)and not R(3)and not F(3)) or (not S(3)and R(3)and F(3));
              WHEN "010"=>Cy  <= (R(3)and not S(3)) or (not S(3)and not F(3))or (not F(3)and R(3));     --减法运算
                   Over<= (R(3)and not S(3)and not F(3))          or (not R(3)and S(3)and F(3));
               WHEN OTHERS=> null;;
   END CASE;
   Sign<=F(3);                          --结果符号位的值 Sign
   IF F="0000" THEN Zero<='1'; ELSE Zero<='0'; END IF;   --结果为零标志 Zero
 END PROCESS;
```



```
PROCESS (CP,I2_0,I8_6)  --选择ALU的输入数据来源和计算结果处理方案
 BEGIN
   CASE I2_0 IS        --确定送到ALU的 2 路输入数据的来源
     WHEN "000" => MUL_R_SEL<="01"; MUL_S_SEL<="11";    -- A->R,  Q->S
     WHEN "001" => MUL_R_SEL<="01"; MUL_S_SEL<="10";    -- A->R,  B->S
       …….
     WHEN OTHERS => NULL;
   END CASE;
   CASE I8_6 IS    --确定对运算结果是否保存、保存到哪个寄存器、运算器输出的内容
              WHEN "000" => REG_W<='0';  Q_W<='1'; Y<=F;   --F->Q
              WHEN “001” => REG_W<=‘0’;  Q_W<=‘0’; Y<=F;
               ……
              WHEN OTHERS=>NULL;
   END CASE;
 END PROCESS;
  WITH MUL_R_SEL SELECT   R     --选择 ALU 的 R 输入端的数据来源
     A      WHEN “01”,
     D      WHEN “10”,
   “0000”  WHEN OTHERS;
 WITH MUL_S_SEL SELECT  S          --选择 ALU 的 S 输入端的数据来源
     A      WHEN “01”,
     B      WHEN “10”,
     Q      WHEN “11”,       "0000"  WHEN OTHERS;
```

二选一

三选一

二选一

三选一

三选一

小结

- 运算器基本组成
- 运算器的基本设计过程
- 运算器Am2901
  - 4位运算器
  - 8种运算功能
  - 8种数据来源组合
  - 8种数据输出方式
  - VHDL方式实现举例

阅读和思考

- 实验分组
  - 实验分组名单已经发布
  - 暂未分组同学请及时组队
- 阅读
  - Am2901的有关材料
  - 《计算机硬件实验教程》VHDL 一章
- 思考
  - 用VHDL语言实现完整的Am2901.

国庆节快乐