一、简答题（48 分）

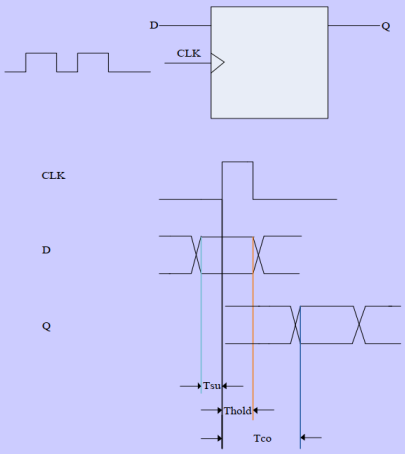
*3 请解释 Tsu、Thold 和 Tco 并画图说明

Tsu、Thold、Tco综合时序图

Tsu:时钟上升沿到来时，输入端数据（D端）必须提前准备好（setup）的时间（相对于时钟上升沿）。

Thold:时钟上升沿结束后，为保证数据正确传输到输出端（Q端），输入端（D端）数据必须保持（hold）的一段时间（相对于时钟上升沿）。

Tco:时钟上升沿与输出端（Q端）数据稳定输出的时间差。

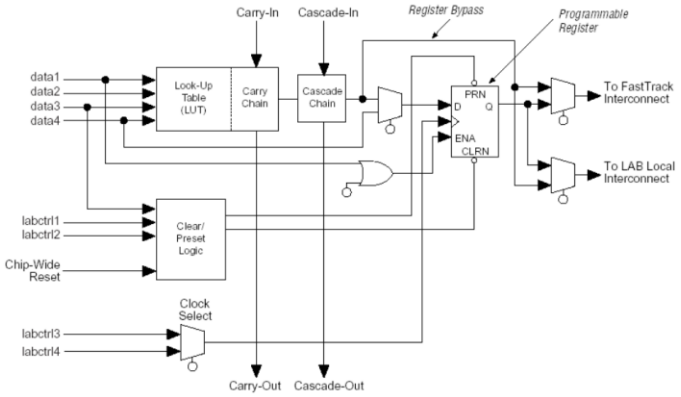


*2 画出 LE 和宏单元 Macrocell 的主要结构图，并说明分别由哪些组成
LE : FPGA 的逻辑单元，也就是最小的完整逻辑结构。LE = Logic Element

逻辑单元LE

- 逻辑单元LE是**FLEX10K结构中的最小逻辑单位。**
- 每个LE包含一个**4输入LUT**、一个带有同步使能的**可编程触发器、一个进位链、一个级连链。**
- LUT是一个函数发生器，它可以快速计算4变量的任意函数。
- **每个LE驱动局部互连线以及FastTrack互连线。**

Figure 6. FLEX 10K Logic Element



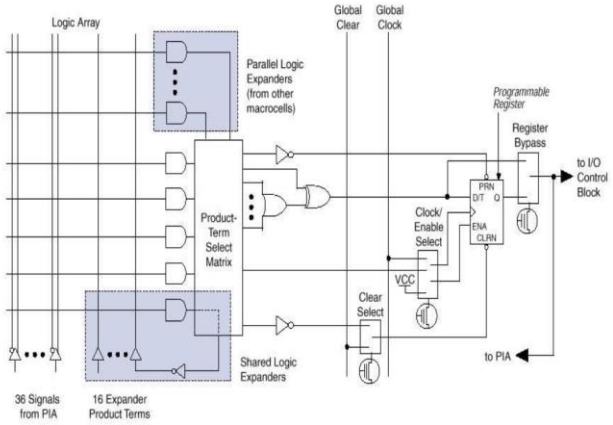
宏单元 Macrocell 是 CPLD 的最小逻辑单元

2．宏单元 Macrocells

MAX7000的宏单元可以被单个配置成**时序**逻辑或**组合**逻辑。
Macrocells由三个功能块组成：

- **逻辑阵列（logic array）**
  可以实现组合逻辑，每个宏单元可以提供5个乘积项。
- **乘积项选择矩阵（product-term select matrix）**
  用来分配这些乘积项，它们或者作为基本逻辑输入实现组合逻辑功能，或者作为宏单元的寄存器的clear、preset、clock、clock enable控制功能。
  有两种扩展乘积项：
  a. 共享扩展项，反馈进逻辑阵列
  b. 并行扩展项，从近邻宏单元借来的扩展乘积项

Figure 3. EPM7032, EPM7064 & EPM7096 Device Macrocell



*3 SignalTap II 有什么特点？和传统逻辑分析仪相比有什么优点？简要说明 SignalTapII 的使用步骤

8-1 SignalProbe（信号探针）

- 在逻辑调试过程中，为了观察内部的信号，很多人通常将感兴趣的信号引到某个输出管脚，以利测试，测试完成后删除。
- 将信号引到管脚，实际上是增加了一个管脚和布线，一般会影响原来真实逻辑的适配。我们观察到的信号往往不是真实的信号时序。

SignalProbe 使用步骤（QuartusII8.0）

Step1：在设计代码或原理图中，增加输出管脚，**完整编译**。

Step2：Assignment >Assignment Editor> assign pins：

- 选择Location；选择IO Standard；
- 选择Reserve pin : as signalprobe output；
- **完整编译一遍**。

Step3：Tools>SignalProbe pins… >add source,

用node finder，选择可以作为signalprobe 的驱动源信号；

Step4：Signalprobe编译：

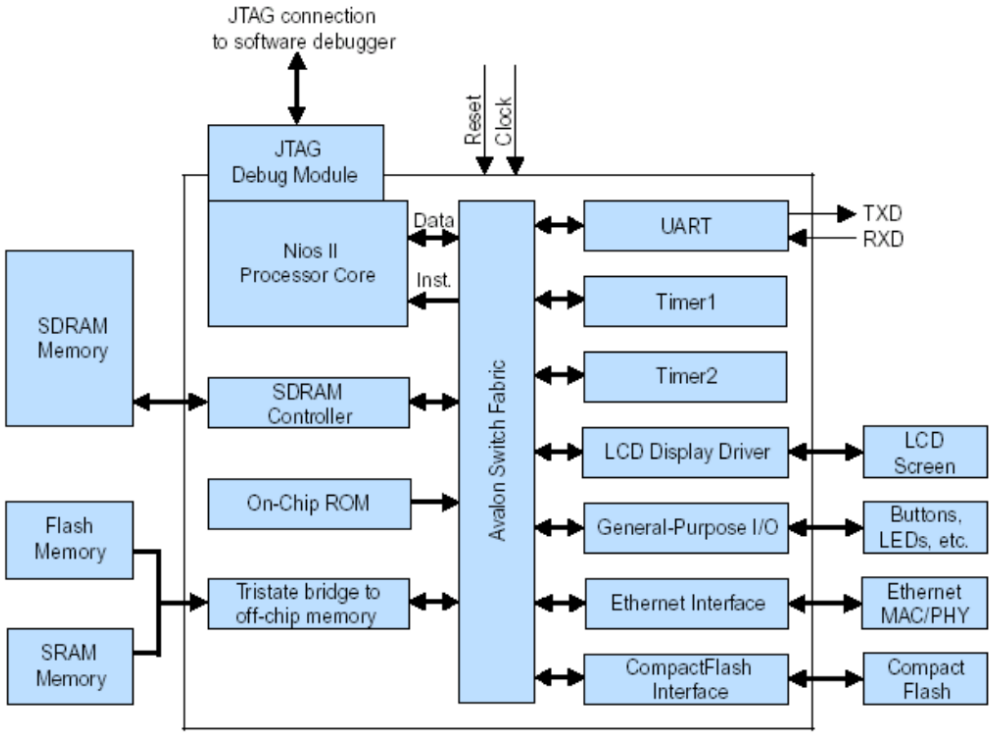
- Processing >start>start signalprobe compilation；
- **注意：切不可完全编译，否则观察不到仿真信号。**

Step5：时序仿真，观察信号时序.

从输出管脚，用示波器观察波形.

*2 如何理解 Nios II processor system？简要说明 SOPC 的使用步骤

Figure 1–1. Example of a Nios II Processor System



世界上最通用的处理器

通过将CPU、外设、存储器接口和定制硬件外设进行完美组合，达到每一新设计的特殊要求，Nios®II处理器能够灵活的满足您的需求：

- **最优的CPU系列**——从几种Nios II CPU中进行选择，每一种都针对特定的价格/性能点进行了优化，均由相同的软件工具链提供支持。
- **可定制外设集**——利用**SOPC Builder**和提供的外设知识产权**(IP)**，针对应用需要，建立并配置合适的外设、存储器和**I/O**电路，而这些是商用处理器所不能提供的。
- **目标芯片选择**——**Altera**提供多种**FPGA**和结构化**ASIC**，能够满足较大范围的成本/性能需求。**Nios II**处理器支持所有的**Altera**主流**FPGA**和结构化**ASIC**。
- **可更新的性能**——通过硬件加速器、定制指令，以及多处理器系统来提升软件性能。现在，**Altera**还提供自动加速工具。 嵌入式开发人员面临的主要挑战是选择一款最适用的处理器，既不浪费性能又不会牺牲功能。市场上有数百种处理器可供选择，许多供应商提供各种外设、存储器、接口，它们在性能上各有特点。您很难避免花费过多(确保达到功能和性能的要求)或者没有满足实际需要(为了避免在不需要的功能或者性能上超支)。

Steps for developing a Nios II system:

1. **Analyzing system requirements**
2. **Defining and generating Nios II system hardware in SOPC Builder**
3. **Integrating the SOPC Builder system into a Quartus II project**
4. **Compiling the Quartus II project & verifying timing**
5. **Download the FPGA configuration file (.sof) to the target board.**
6. **Creating a new project in the Nios II IDE**
7. **Compiling the project：Build the project**
8. **Running the software on the ISS and target hardware：**
   Right click the project name and choose Run As > Nios II Hardware.
   The IDE downloads the program to the FPGA on the target board and starts execution.

Nios II 处理器系统是一种由 Altera 公司（现为英特尔旗下公司）设计的可定制的嵌入式处理器架构。这种软核处理器允许开发者在可编程逻辑设备，如现场可编程门阵列（FPGA）上实现自定义的处理器系统。Nios II 处理器的灵活性在于其可配置性，允许开发者根据具体的应用需求选择不同的功能和性能选项。

SOPC（System On a Programmable Chip，可编程芯片上的系统）是一种设计方法，它允许开发者在单一的可编程逻辑芯片上集成完整的系统，包括处理器、存储器、外设和自定义逻辑。Nios II 处理器经常被用于 SOPC 设计中，因为它提供了高度的灵活性和定制能力。

SOPC 的使用步骤通常包括以下几个阶段：

需求分析和规划：确定系统的需求，包括所需的处理能力、存储器、外设接口等。

选择合适的硬件平台：基于需求分析选择合适的 FPGA 或其他可编程逻辑设备。

设计处理器系统：使用像 Quartus 这样的工具，配置 Nios II 处理器的参数，添加所需的外设和自定义逻辑。

开发软件：为处理器系统编写软件，这可能包括操作系统、驱动程序和应用程序。

模拟和测试：在实际部署到硬件之前，使用模拟器对设计进行测试和验证。

系统实现：将设计下载到 FPGA 或其他目标平台，并进行实际测试和调试。

优化和调整：根据测试结果进行必要的调整和优化，以满足性能和功能的要求

Nios II 处理器系统在 FPGA 内部的实现依赖于 FPGA 的可编程逻辑资源，其中包括逻辑数组块（LAB）和嵌入式数组块（EAB）。Nios II 本身是一个软核处理器，意味着它是以硬件描述语言（如 VHDL 或 Verilog）形式实现的，可以在 FPGA 的可编程逻辑中配置和实例化。这种处理器的实现既涉及 LAB 也可能涉及 EAB，具体取决于其配置和所用 FPGA 的架构。

1. **逻辑数组块（LAB）**：这些是构成 FPGA 的基本组成部分，由多个可编程逻辑单元（如查找表、寄存器和逻辑门）组成。Nios II 的大部分功能，包括 ALU（算术逻辑单元）、寄存器以及控制逻辑等，通常是在这些 LAB 中实现的。

2. **嵌入式数组块（EAB）**：这些是 FPGA 中用于存储的专用区域，通常用于实现 RAM、ROM 或其他形式的存储器。Nios II 处理器系统可能会使用 EAB 来实现其指令缓存、数据缓存或其他内存需求。

Nios II 处理器的一个关键特点是其高度的可定制性。开发者可以根据应用需求选择不同的配置选项，如不同的缓存大小、指令集选项、外设接口等。这种灵活性使得 Nios II 能够适应广泛的应用场景，从简单的控制任务到更复杂的数据处理任务。

实现 Nios II 处理器时，开发者通常会使用 Altera（现为英特尔）提供的 Quartus Prime 设计软件，该软件提供了 Nios II 处理器的配置界面，允许用户根据需要选择各种参数和功能。配置完成后，软件会生成相应的硬件描述语言代码，该代码随后被编译并加载到 FPGA 中，实现 Nios II 处理器的具体硬件实例。在这个过程中，Quartus 软件会自动处理逻辑资源的分配，包括 LAB 和 EAB 的使用。

*2 组合逻辑电路是什么？毛刺有什么危害，是如何产生和消除的？组合逻辑电路的毛刺是如何产生和消除的？什么电路不能有毛刺
（因为输入信号到达时间不同步，消除可以加延时同步，加触发器修正，或者直接修改电路）
组合逻辑电路是一种数字逻辑电路，其输出仅取决于当前的输入值，而与之前的输入或状态无关。这与时序逻辑电路不同，后者的输出不仅取决于当前的输入，还取决于历史输入（即电路的历史状态）。组合逻辑电路的例子包括逻辑门（如 AND、OR、NOT）和更复杂的结构，如多路选择器、解码器和算术逻辑单元（ALU）。

毛刺是在数字电路中经常出现的一个问题，特别是在组合逻辑电路中。毛刺是指输出信号中的短暂和非预期的变化，通常是由电路中不同部分的延迟差异引起的。例如，当一个组合逻辑电路的多个输入同时变化时，由于各个输入路径的传播延迟不同，可能会在输出上产生短暂的错误状态，即毛刺。

毛刺的危害包括：

1. **错误操作**：毛刺可能导致电路的后续部分错误地触发，引发不稳定和不可预测的行为。
2. **数据损坏**：在数据传输或存储操作中，毛刺可能导致错误的数据被写入。
3. **增加功耗**：频繁的毛刺可能导致电路功耗增加。

产生毛刺的原因：

1. **传播延迟**：电路中不同路径上的信号传播延迟不一致。
2. **电源和接地噪声**：电源线和接地线上的噪声也可能引起毛刺。
3. **环境因素**：如温度变化、电磁干扰等。

消除毛刺的方法：

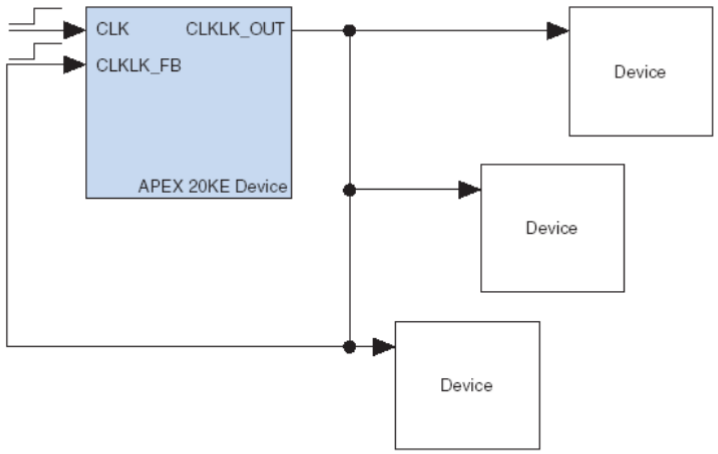
1. **添加滤波器**：使用低通滤波器可以平滑输出信号，消除毛刺。
2. **重新设计逻辑**：通过重新设计电路布局和逻辑，以均衡不同路径的延迟。
3. **使用触发器稳定**：在组合逻辑电路的输出端添加触发器（如 D 触发器），在时钟边沿锁存输出值，可以滤除毛刺。
4. **改善电源和接地**：优化电源和接地设计，减少噪声引入。

对于**不能有毛刺的电路**，最典型的例子是时钟信号和同步电路。时钟信号通常用于驱动时序逻辑电路，如触发器和寄存器。毛刺可能导致时序错误，引起数据损坏或系统不稳定。因此，对于这些电路，通常会采取特别的设计措施，如使用专用的时钟生成和分配电路，以确保时钟信号的稳定性和准确性。

消除毛刺的方法：
② 通过调整布线让端口输出延迟一致，会受到外界条件因素影响；
② 通过添加触发器让端口输出按时钟同步，从而消除毛刺；
③ 通过修改逻辑，采用类似格雷码的设计，减少或消除同一时刻端口同时变化的可能，从根本上杜绝毛刺现象。

*3 使用 APEX 20KE，用输出时钟驱动三个元件，如何使输入时钟和他们同步？画图说明

Figure 21. Reducing Board Delay Using an APEX 20KE Device *Note (1)*

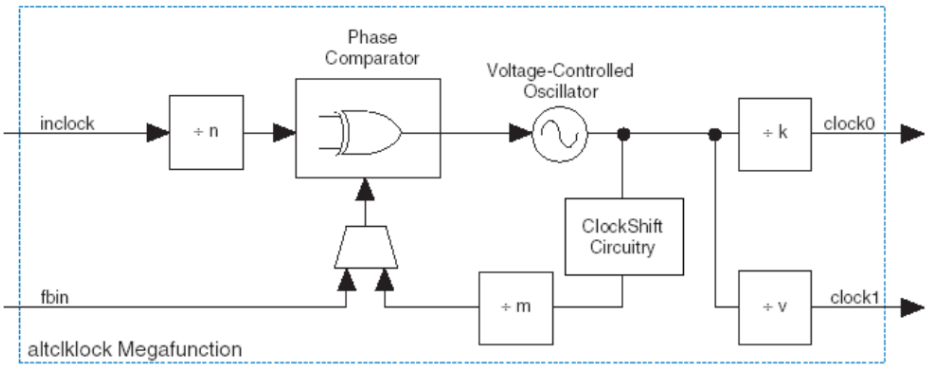


Note:

(1) For board design, the route delay from `CLKLK_OUT1` to each device and the return route delay to `CLKLK_FB1` should be equal.

Minimize the delays between the `CLKLK_OUT` and `CLKLK_FB` signals in APEX 20KE devices. Be sure the board fly time is less than 5 ns or 50% of the input clock period, whichever is less.

Figure 5. ClockLock & ClockBoost Circuitry in APEX 20KE Devices



这个题目的意思是，利用 FPGA 这个芯片，可以做到将输入的时钟源复制 3 份驱动三个不同的器件。然后要求这个三个不同的器件得到的时钟和原始的时钟之间同步。这里使用了神奇的 clk 反馈

4. 具有锁相环的时钟管理电路

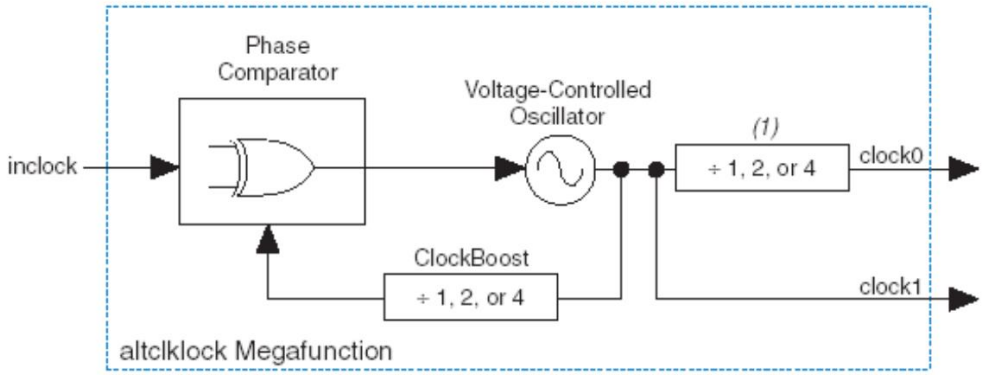
- 内有low-skew时钟分配树
- 多达8个GCLK
- **ClockLock，可以减少时钟延时和偏差**
- **ClockBoost，可以提供时钟倍频和分频**
- **ClockShift可编程时钟相位和延时的移动（shifting）**

- APEX具有ClockLock and ClockBoost特性。
- ClockLock：减小clock delay and clock skew，在维持 0 hold time的同时，减小 clock-to-output and setup times。
- ClockBoost： 20K可以固定2X或4X倍频。20KE可以任意放大或减小输入的时钟频率 。
- **Clock Skew**定义：the difference between clock delays to different registers.
- **Tco**延时中的大部分是来自clock delay和clock skew。ClockLock and ClockBoost可以减小tco。

- ClockLock & clockBoost电路具有自己的电源和地，它提供电源给PLL以及它的专用时钟输出脚。PLL的电源和地必须与器件的其它电源或其它器件的电源隔离，可以减小时钟jitter。
- 对混合系统，已有模拟和数字电源和地，则可以将VCC_CKLK[4..1]，GND_CKLK[4..1]接到模拟电源，将输出VCC_CKOUT、GND_CKOUT接到数字电源和地。
- 对全数字系统，增加层数费用较高，可以产生一些隔离岛。

*1 APEX20KE 的锁相环是如何实现一个什么功能的，画图说明（那个功能的说明很奇怪，我没见过，我就画了个 CLOCKBOOST 的图上去，不管了）

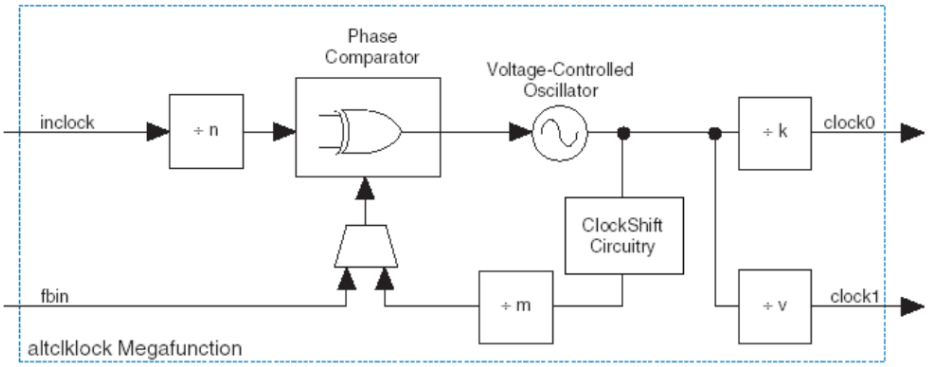
Figure 2. ClockLock & ClockBoost Circuitry in APEX 20K Devices



Note:

(1) This division is used only for the purpose of dividing down a 2×/4× `clock1` to obtain a 1×/2×/4× on `clock0`.

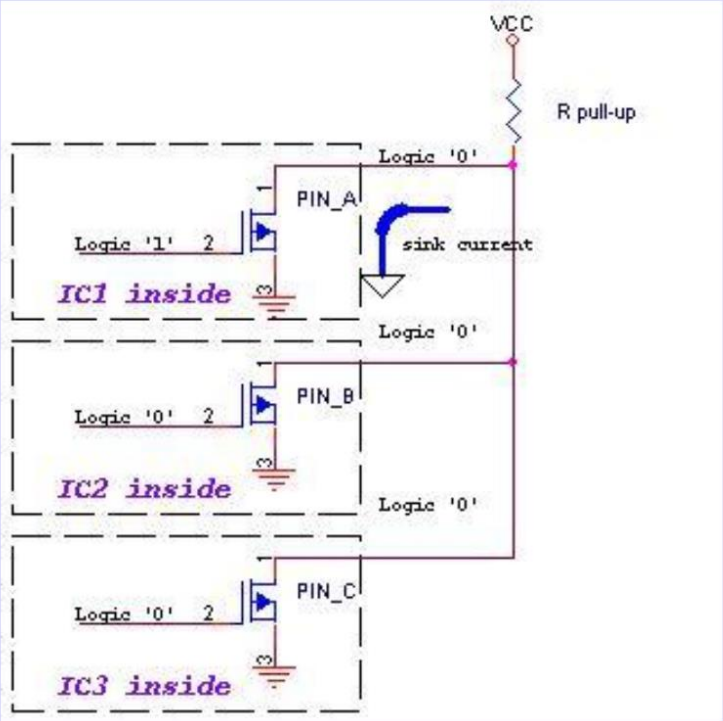
Figure 5. ClockLock & ClockBoost Circuitry in APEX 20KE Devices



*1 请说明 PLD 的 I/O 管脚的漏极开路（OD）的特点

3．漏级开路（Open-Drain）输出选择（仅MAX7000S设备）

- MAX7000S设备提供可选的**漏级开路输出**（功能相当于集电极开路：open-collector）。开漏电路概念中提到的“漏”就是指MOS FET的漏极。
- 这个漏级开路输出，可以实现“**线与**”逻辑，可以被多个设备置低。例如：中断、写使能等。



任何一个OD门输出为低，则总的输出为低。

上拉电阻R pull-up的阻值决定了逻辑电平转换的沿的速度。阻值越大，速度越低功耗越小。

*1 FPGA 从 FLEX 10K 发展到 Stratix II，请简要分析其发展趋势

从 FLEX 10K 到 Stratix II，FPGA（现场可编程门阵列）的发展体现了若干重要的技术趋势和创新。下面简要分析这些趋势：

1. **增加的逻辑密度**：随着制造技术的进步，FPGA 的逻辑密度大幅增加。FLEX 10K 系列作为早期的 FPGA 之一，其逻辑密度相对较低。到了 Stratix II 时代，由于更先进的制程技术，可以在相同或更小的芯片尺寸上实现更多的逻辑单元，允许更复杂的设计和应用。

2. **更高的性能和更低的功耗**：随着工艺节点的缩小和设计的优化，新一代的 FPGA 在性能上有显著提升，同时功耗得到了有效控制。Stratix II 系列相比 FLEX 10K 在这方面有显著的改进，提供更高的操作频率和更低的功耗。

3. **增强的功能集**：新一代 FPGA 通常包含更多的内建功能和资源。例如，Stratix II 相比 FLEX 10K 提供了更多的 DSP（数字信号处理）单元、更高容量的内存块、更多的 I/O 选项和更复杂的时钟管理功能。

4. **改进的编程和配置技术**：随着 FPGA 技术的发展，编程和配置的方法也得到了改进。更高级的设计软件（如 Quartus II）和配置技术的使用使得设计师能够更快、更有效地实现复杂设计。

5. **更广泛的应用领域**：随着性能的提升和成本的降低，FPGA 开始被用于更广泛的应用领域，从早期的简单逻辑替换到复杂的数据处理、通信、图像处理和嵌入式系统。

6. **集成度的提升**：在 Stratix II 等较新的 FPGA 中，集成度显著提高，不仅包括更多的逻辑单元，还包括更高级的系统级功能，如硬核处理器、高速串行接口等。

7. **改进的设计工具和生态系统**：FPGA 的发展伴随着设计工具的改进，包括更好的仿真、验证工具和更广泛的 IP 核资源，这些工具和资源使得 FPGA 的设计和开发更加高效和用户友好。

总的来说，FPGA 从 FLEX 10K 到 Stratix II 的发展反映了行业对更高性能、更低功耗、更高集成度和更广泛应用的不断追求。这些发展趋势不仅推动了 FPGA 技术的进步，也为各种行业的技术创新提供了强大的工具。

*1 CAM 是什么，如何实现 IP 过滤器（CAM 很简单不用说，IP 过滤器 PPT 上有）
CAM（内容可寻址存储器）是一种特殊类型的存储器，它允许通过内容而不是通过地址来快速访问数据。在 CAM 中，数据可以被快速检索，因为它允许同时对存储的所有条目进行搜索操作。当提供给 CAM 的搜索词与存储在其中的任何数据匹配时，CAM 会返回匹配数据的地址。这种能力使 CAM 特别适用于那些需要高速搜索操作的应用，如网络路由器中的 IP 地址查找和数据包过滤。

IP 过滤器是一种网络功能，用于控制数据包的流动，允许或阻止特定的 IP 地址或 IP 地址范围。在实现 IP 过滤器时，CAM 可以用于高效地匹配 IP 地址。以下是使用 CAM 实现 IP 过滤器的基本步骤：

1. **存储 IP 地址规则**：在 CAM 中存储 IP 过滤规则。这些规则可以是允许的 IP 地址、拒绝的 IP 地址或 IP 地址范围。每个规则通常与一个动作相关联，例如“允许”或“阻止”。

2. **接收数据包**：当网络设备接收到一个数据包时，它会提取数据包的源 IP 地址或目标 IP 地址。

3. **搜索 CAM**：网络设备使用提取的 IP 地址作为搜索键，在 CAM 中查找匹配的规则。

4. **动作执行**：如果在 CAM 中找到匹配的规则，执行与该规则关联的动作（例如，允许或阻止该数据包）。如果没有找到匹配项，则可以根据默认策略处理数据包，例如默认允许或默认阻止。

5. **更新规则**：根据网络策略的变化，需要更新 CAM 中存储的 IP 过滤规则。

使用 CAM 实现 IP 过滤器的优势在于其高速的搜索能力，这对于处理大量网络流量并实时做出过滤决策非常重要。不过，CAM 的缺点是成本较高和功耗较大，这在某些应用中可能是一个考虑因素。在设计 IP 过滤器时，还需要考虑规则的更新和管理，以及如何处理 CAM 存储空间有限的问题。

1．CAM概述

- APEX20KE系列中，ESB可以实现CAM：Content-Addressable Memory。
  APEX20K系列不支持CAM。
- CAM可以认为是RAM的逆过程。
  读RAM，当给定一个地址，可以得到该地址里的数据
  CAM是对一个给定的数据，可以得到该数据所在存储单元的地址。

二、HDL 设计题（10 分）
1 . 3_8 译码器

2. 设计一个 10bit 减 1 计数器，要求从 200H 到 0H 之间循环计数，并设有低电平复位键和低电平置 1（即输出置 3FFH）

3. 设计 DFF

```
module decoder3to8 (
    input [2:0] in,    // 3-bit input
    output [7:0] out // 8-bit output
);
    assign out = 1 << in;    // Shift left operation for decoding
endmodule

module counter10bit (
    input clk,             // Clock input
    input reset_n,      // Active low reset
    input set_n,         // Active low set to 3FFH
    output reg [9:0] count = 10'h200    // 10-bit counter initialized to 200H
);
    always @(posedge clk or negedge reset_n or negedge set_n) begin
        if (!reset_n)
            count <= 10'h200;    // Reset to 200H
        else if (!set_n)
            count <= 10'h3FF;    // Set to 3FFH
        else if (count == 0)
            count <= 10'h200;    // Reset to 200H when count reaches 0
        else
            count <= count - 1; // Decrement counter
    end
endmodule

module dff (
    input d,              // Data input
    input clk,           // Clock input
    input reset_n,    // Active low reset
    output reg q       // Output
);
    always @(posedge clk or negedge reset_n) begin
        if (!reset_n)
            q <= 1'b0;    // Reset output to 0
        else
            q <= d;          // Store data at clock edge
    end
endmodule
```

三、翻译题(8 分）
取自 Stratix II 的课件
原文取自第十一章 Stratix II 的课件
The basic building block of logic in the Stratix II architecture, the adaptive logic module (ALM), provides advanced features with efficient logic utilization.
Each ALM contains a variety of look-up table (LUT)-based resources that can be divided between two adaptive LUTs (ALUTs). With up to eight inputs to the two ALUTs, one ALM can implement various combinations of two functions. This adaptability allows the ALM to be completely backward-compatible with four-input LUT architectures. One ALM can also implement any function of up to six inputs and certain seven-input functions.
In addition to the adaptive LUT-based resources, each ALM contains two programmable registers, two dedicated full adders, a carry chain, a shared arithmetic chain, and a register chain. Through these dedicated resources, the ALM can efficiently implement various arithmetic functions and shift registers.

文本 1 翻译：
Stratix II 架构中逻辑的基本构建块，自适应逻辑模块（ALM），提供高效的逻辑利用率和先进的特性。每个 ALM 包含多种基于查找表（LUT）的资源，这些资源可以在两个自适应 LUT（ALUT）之间进行分配。两个 ALUT 最多有八个输入，一个 ALM 可以实现两个函数的各种组合。这种适应性使 ALM 能够与四输入 LUT 架构完全向后兼容。一个 ALM 还可以实现最多六个输入的任何功能，以及某些七输入功能。除了基于自适应 LUT 的资源外，每个 ALM 还包含两个可编程寄存器、两个专用全加器、一个进位链、一个共享算术链和一个寄存器链。通过这些专用资源，ALM 可以高效实现各种算术功能和移位寄存器。

值得记忆的科技英语单词：
Adaptive Logic Module (ALM)
Look-Up Table (LUT)
Adaptive LUTs (ALUTs)
Backward-compatible
Programmable registers
Full adders
Carry chain
Arithmetic chain
Register chain

In addition to the general routing outputs, the ALMs in an LAB have register chain outputs.
The register chain routing allows registers in the same LAB to be cascaded together. The register chain interconnect allows an LAB to use LUTs for a single combinational function and the registers to be used for an unrelated shift register implementation. These resources speed up connections between ALMs while saving local interconnect resources3

文本 2 翻译：
除了通用的路由输出外，LAB 中的 ALMs 还有寄存器链输出。寄存器链路由允许同一 LAB 中的寄存器级联在一起。寄存器链互连允许 LAB 使用 LUTs 实现单一组合功能，同时将寄存器用于与之无关的移位寄存器实现。这些资源加速了 ALMs 之间的连接，同时节省了本地互连资源。

值得记忆的科技英语单词：
General routing
Register chain
Cascaded
Combinational function
Shift register
Interconnect

TriMatrix memory consists of three types of RAM blocks: M512, M4K, and M-RAM. Although these memory blocks are different, they can all implement various types of memory with or without parity, including true dual-port, simple dual-port, and single-port RAM, ROM, and FIFO buffers.

文本 3 翻译：
TriMatrix 内存由三种类型的 RAM 块组成：M512、M4K 和 M-RAM。虽然这些内存块各不相同，但它们都可以实现各种类型的内存，包括有或没有奇偶校验的真双端口、简单双端口和单端口 RAM、ROM 和 FIFO 缓冲区。

值得记忆的科技英语单词：
TriMatrix memory
RAM blocks
Dual-port
Single-port
ROM
FIFO buffers

The most commonly used DSP functions are FIR filters, complex FIR filters, IIR filters, fast Fourier transform (FFT) functions, direct cosine transform (DCT) functions, and correlators. All of these use the multiplier as the fundamental building block. Additionally, some applications need specialized operations such as multiply-add and multiply-accumulate operations. Stratix II devices provide DSP blocks to meet the arithmetic requirements of these functions.
Each Stratix II device has from two to four columns of DSP blocks to efficiently implement DSP functions faster than ALM-based implementations. Stratix II devices have up to 24 DSP blocks per column (see Table 2–5). Each DSP block can be configured to support up to:
■ Eight 9 ×9-bit multipliers
■ Four 18 ×18-bit multipliers
■ One 36 ×36-bit multiplier
As indicated, the Stratix II DSP block can support one 36 ×36-bit multiplier in a single DSP block.

最常用的 DSP 功能包括 FIR 滤波器、复 FIR 滤波器、IIR 滤波器、快速傅里叶变换（FFT）功能、直接余弦变换（DCT）功能和相关器。所有这些功能都使用乘法器作为基本构建块。此外，一些应用程序需要特殊操作，如乘加和乘累加操作。Stratix II 设备提供 DSP 块以满足这些功能的算术要求。每个 Stratix II 设备有两到四列 DSP 块，用于比基于 ALM 的实现更高效地实现 DSP 功能。Stratix II 设备每列最多有 24 个 DSP 块（见表 2-5）。每个 DSP 块

可以配置为支持：
■ 最多八个 9 × 9 位乘法器
■ 四个 18 × 18 位乘法器
■ 一个 36 × 36 位乘法器
正如所示，Stratix II DSP 块可以在单个 DSP 块中支持一个 36 × 36 位乘法器。

值得记忆的科技英语单词：
DSP (Digital Signal Processing)
FIR filters
IIR filters
Fast Fourier Transform (FFT)
Direct Cosine Transform (DCT)
Correlators
Multiply-accumulate operations
DSP blocks
Multipliers

Stratix II devices provide a hierarchical(分层的、等级的) clock structure and multiple PLLs with advanced features. The large number of clocking resources in combination with the clock synthesis precision provided by enhanced and fast PLLs provides a complete clock management solution.

文本 5 翻译：
Stratix II 设备提供了分层的时钟结构和具有高级特性的多个 PLL（相位锁定环）。大量的时钟资源结合增强型和快速 PLL 提供的时钟合成精度，为完整的时钟管理解决方案提供了支持。

Stratix II devices provide 16 dedicated global clock networks and 32 regional clock networks (eight per device quadrant). These clocks are organized into a hierarchical（分层、等级） clock structure that allows for up to 24 clocks per device region with low skew and delay. This hierarchical clocking scheme provides up to 48 unique clock domains in Stratix II devices.
文本 6 翻译：
Stratix II 设备提供 16 个专用的全局时钟网络和 32 个区域时钟网络（每个设备象限 8 个）。这些时钟被组织成分层的时钟结构，允许每个设备区域最多使用 24 个时钟，且具有低偏斜和低延迟。这种分层时钟方案为 Stratix II 设备提供了多达 48 个独特的时钟域。

Using one of eight phase-shifted clocks generated by the fast PLL, the dynamic phase aligner samples the incoming data and aligns the data by choosing the clock phase that is closest to the center of the incoming data. This alignment is continuous and can compensate for dynamic changes in the real-time timing variations between the clock and data signals.

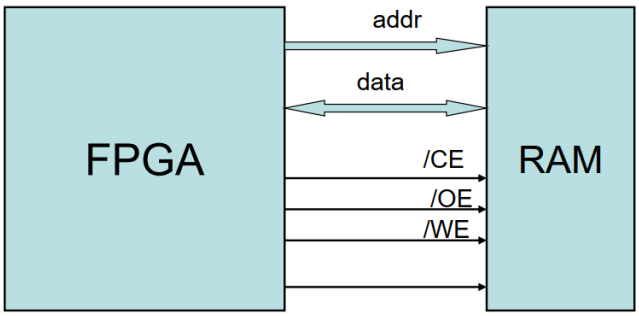
The DPA circuitry supports multiple SERDES factors including
the 3x to 10x modes. Each channel has its own DPA circuit that
provides independent data alignment for each channel; therefore,
DPA can eliminate channel-to-channel skew as well as clockto-channel skew.

文本 7 翻译：

通过使用快速 PLL 生成的八个相移时钟中的一个，动态相位对齐器采样输入数据，并通过选择最接近输入数据中心的时钟相位来对齐数据。这种对齐是连续的，并且可以补偿时钟与数据信号之间实时定时变化的动态变化。DPA（动态相位对齐）电路支持多种 SERDES 因素，包括 3x 到 10x 模式。每个通道都有自己的 DPA 电路，为每个通道提供独立的数据对齐；因此，DPA 可以消除通道间偏斜以及时钟到通道的偏斜。

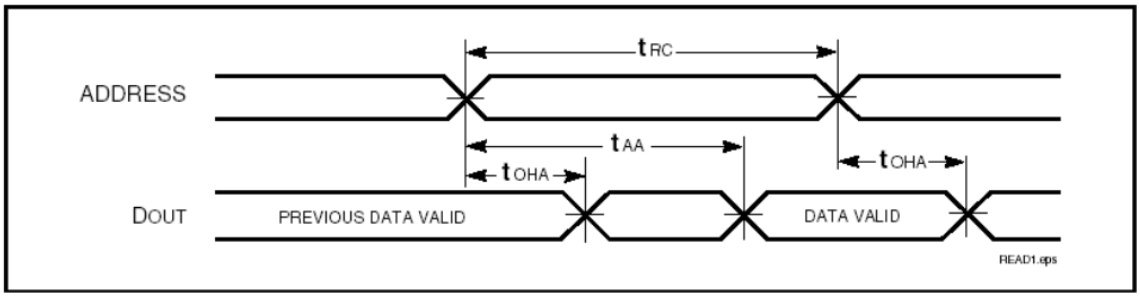
四、时序分析题（34 分）

FPGA读写RAM连接图



AC WAVEFORMS

READ CYCLE NO. 1[1,2] (Address Controlled) ($\overline{CE}$ = $\overline{OE}$ = V_{IL}, $\overline{UB}$ or $\overline{LB}$ = V_{IL})



tRC： min 7ns。 tAA： max 7ns。 tOHA： min 3ns。

1.异步 SRAM 的读时序图分析 异步 SRAM 的写模式的时序分析 经典异步 SRAM 的那个时序图（20 分）

①解释 tRC、tAA、tOHA（8 分）（懂得都懂）

②tRC 最小受什么决定的（6 分）（这尼玛谁知道，反正这玩意肯定比 tAA 大 ）

③如果 tRC 分别为 7ns、10ns，求稳定输出的时间（6 分）

（从图上看，这个稳定时间等于 tRC+tOHA-tAA，但具体值我觉得没办法求，只能求范围，最小的话是 tRC+tOHAmin-tAAmax，最大的话应该就是 tRC，因为 tOHA 应该小于等于 tAA）

(1）解释三个参数，，。read cycle time, address access time，output hold time
（2）TRC 改变，稳定输出信号时间。
（3）最大频率由什么决定，并分析

对于异步 RAM（Asynchronous RAM），以下是对您提出的问题的详细解释：

(1) 参数释义

1. **读周期时间（Read Cycle Time, TRC）**：这是完成一次读操作所需的最小时间。它从提供有效地址开始，直到数据可靠地读取为止。TRC 包括地址访问时间和数据稳定时间。

2. **地址访问时间（Address Access Time, TAA）**：这是从 RAM 的地址输入被提供到相应数据可用于读取的时间。它是读操作过程中的一个关键部分。

3. **输出保持时间（Output Hold Time, TOH）**：在地址或控制信号变化后，数据输出保持不变的时间。这个参数对于确保数据完整性是重要的，特别是在数据和控制信号变化的边缘情况下。

(2) TRC 的变化对稳定输出信号的影响

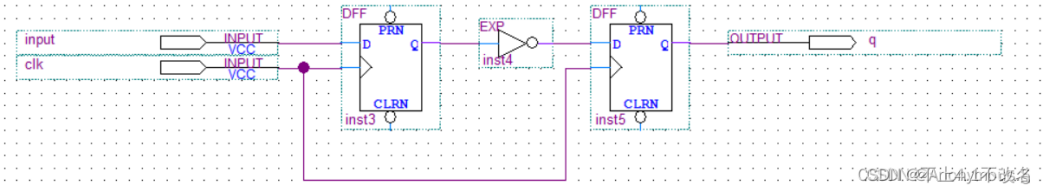
- 当 TRC（读周期时间）变化时，它直接影响了从提供地址到数据稳定可读的总时间。
- 如果 TRC 减小，意味着数据需要更快地变得稳定，这可能会对系统的整体性能造成影响，特别是如果其他组件无法以同样的速率响应时。
- 如果 TRC 增加，虽然提供了更长的时间来稳定数据，但这会降低整体的数据访问速度，从而影响系统性能。

(3) 最大频率的决定因素及分析

- 异步 RAM 的最大操作频率主要由其最慢的操作时间决定，这通常是读周期时间（TRC）。
- 如果 TRC 长，那么 RAM 在完成一次读操作前需要更多时间，从而限制了它在给定时间内可以执行的操作数量，降低了最大频率。
- 对于异步 RAM，其他因素如地址访问时间（TAA）和数据输出的稳定时间也很重要，因为它们直接影响了数据读取的响应速度。
- 在分析最大频率时，还需要考虑系统中其他组件的响应时间和相互作用，以及 RAM 的温度和电源稳定性等环境因素。

总结来说，异步 RAM 的性能和最大操作频率受到其内部时序参数的限制，其中读周期时间是关键因素。在设计和选择 RAM 时，需要综合考虑这些参数及其对整个系统性能的影响。

2.同一时钟驱动的串联 D 触发器，其正常工作的时钟频率最值的计算，大致图如下图所示



2.多时钟分析的那个，和毛刺在同一章不同 PPT 上的，求 CLOCK 频率最大值（14 分）（我觉得应该是 fmax=1/[Tco+Td+Tsu]，原理 PPT 上有）

锐评：

综合这两套题我们可以看出来，简答题出的不是很固定，

但 Tsu、Thold、Tco 画图解释、

CPLD 的宏单元和 FPGA 的 LE、

SignalTapII 的特点和使用步骤、

组合逻辑电路的毛刺、

APEX20KE 的时钟功能是重中之重，

设计题必然是 DFF、38 译码器和计数器三选一，分析题必然是异步 SRAM 的解释分析和多时钟求最大频率

翻译：英语烂的人想哭，好像自从来了科大以来就没怎么学过英语，五年了完全废掉了，建议大家把 Stratix II 的那个课件好好看看

简答题：以往两位学长的试卷中，我天真的以为必考毛刺、LE 和宏单元，就在考前重点背了，结果一个都没考……反而考了一点没看的 OD 门和 NIOS II……还是建议大家都看看都背背，当然 SignalTap II 和三个 T 的分析几乎是年年必考没啥悬念

时序分析：必考题 SRAM 不多说，串联 D 触发器难度也不大，逻辑设计技巧那一章吃透了就行，多时钟都玩明白了的话这种题更是不在话下了

2022 年中科大可编程逻辑器件原理及应用复习考点资料
1) 什么是可编程逻辑器件
2) PLD 发展趋势
3) FPGA 与 CPLD 的区别
4) PLD 编程元件
5) PLD 厂商
6) Tsu、Thold、Tco、Tpd
7) Max7000 结构
8) Microsell 宏单元
9) Max7000 电源
10) Max7000 输出配置：漏极开路（Open-Drain）
11) 在系统编程 ISP（In-System Programmability）
12) Quartus 设计步骤
13) 功能仿真与时序仿真
14) DFF 仿真（行为、时序）
15) FLEX10K 组成
16) FLEX10K 逻辑单元 LE
17) FLEX10K 时钟锁定与时钟自举
18) RAM & FIFO
19) 计数器
20) 时序分析器
21) 二分频逻辑电路
22) Verilog || VHDL
25) APEX20K MultiCore
26) APEX20K MegaLAB
27) APEX20K IOE->LVDS
29) APEX20K 电源
30) APEX20K ClockLock and ClockBoost
31) 外部时钟输出模式
32) APEX20K ClockShift
33) CAM（内容可寻址寄存器）和应用
35) SignalProbe & SignalTapII
37) 逻辑设计技巧
38) SOPC Nios II cpu 软核-
39) Stratix II

1) 什么是可编程逻辑器件
可由用户逻辑编程或配置实现所需逻辑功能的数字逻辑电路；为用户提供各种逻辑功能、速度和电压特性，可在任何时刻对其逻辑进行修改；用软件工具开发、仿真和测试，可快速编程到 PLD 中并在实际运行电路中测试。

2) PLD 发展趋势
另：结构化 ASIC 相比 FPGA：性能提高，功耗降低，成本下降。
①降低互连延迟，提高速度；②设计技术向高层设计转移；③向数模混合编程技术发展；④各种逻辑软核的开发应用；⑤产品日益丰富，性能渐趋完善。

3) FPGA 与 CPLD 的区别
结构上：CPLD 结构基于乘积项技术、EEPROM 或 Flash 工艺，适合做复杂组合逻辑；FPGA 结构基于查找表技术、SRAM 工艺，适合做复杂时序逻辑。
逻辑存储上：CPLD 逻辑固化在芯片内部；FPGA 由于 SRAM 工艺不可直接固化在芯片内部，需要保存在片外 EEPROM 上，上电烧录。
加密上：CPLD 可以加密；FPGA 自己无法加密。

4) PLD 编程元件
①熔丝或反熔丝开关元件：非易失性元件，能保持逻辑数据，只能编译一次。
②浮栅编程元件：非易失性元件，能保持逻辑数据，擦写寿命十几万次。
③SRAM 配置存储器元件：易失性元件，掉电后数据丢失，强抗干扰能力。

5) PLD 厂商
Xilinx、Altera、Lattice。

6) Tsu、Thold、Tco、Tpd

Tpd：I/O 管脚输入到非寄存器输出延时。（通过一个宏单元内组合逻辑）
同步、异步（时钟）

7) Max7000 结构
逻辑阵列块 LAB，宏单元 Macrocells，扩展乘积项 Expander product terms(sharable and parallel)，可编程连线阵列 PIA，I/O 控制块。

8) Microsell 宏单元
逻辑阵列、乘积项选择矩阵（共享扩展项反馈进逻辑阵列、并行扩展项从近邻宏单元借来的扩展乘积项）、可编程寄存器

9) Max7000 电源
VCCINT：固定，供内部电路和输入缓冲器 buffers；
VCCIO：可配置，供给 I/O 输出缓冲器。
低电压摆率可以减少系统噪声，但会增加 4~5ns 延时。

10) Max7000 输出配置：漏极开路（Open-Drain）
实现“线与“逻辑，可被多个设备置低，如中断、写使能。
漏极电路由开漏器件和开漏上拉电阻组成。任何一个 OD 门输出低，总输出为低。上拉电阻阻值决定逻辑电平转换的沿的速度。阻值越大，速度低功耗小。

11) 在系统编程 ISP（In-System Programmability）
JTAG 接口

12) Quartus 设计步骤
新建工程，新建设计文件（VHDL、verilog 或原理图）、编写设计文件、编译工程、仿真、下载执行。

13) 功能仿真与时序仿真
功能仿真是对代码或原理图描述的逻辑功能进行仿真，了解实现的功能是否满足设计要求；时序仿真是在确定器件并完成布局布线后进行包括延时的仿真，检查设计功能是否能工作在设定的速度上。

14) DFF 仿真（行为、时序）

15) FLEX10K 组成
嵌入式阵列块（EAB）：存储功能（RAM、ROM、双口 RAM、FIFO）和复杂逻辑功能（DSP、微控制器、状态机、乘法器）；
逻辑阵列块（LAB）：8 个逻辑单元 LE、进位链、级连链、LAB 控制信号和局部互连线；
快速互连通道 FastTrack；

I/O 控制块。

16) FLEX10K 逻辑单元 LE

17) FLEX10K 时钟锁定与时钟自举
时钟锁定：用同步 PLL 减少时钟延迟和偏差；时钟自举：时钟倍频。（锁相环 PLL 增加速度，减少资源利用）

18) RAM & FIFO

19) 计数器

20) 时序分析器
21) 二分频逻辑电路

22) Verilog || VHDL
38 译码器、计数器、DFF

25) APEX20K MultiCore
APEX20K 第一个实现 SOPC 集成度。
MultiCore 结构：LUT、Product-term、embeded memory
APEX20K 组成：MegaLAB、FastTrack Interconnect、IOE

26) APEX20K MegaLAB
MegaLAB：10~24 个 LAB、1 个 ESB、1 个 MegaLAB interconnect
LAB：10 个 LE、进位和级连链、LAB 控制信号、local interconnect
LE：4 输入 LUT、可编程寄存器、进位链和级连链

ESB1：可配置为宏单元，一个 ESB 具有 16 个宏单元，每个宏单元包括 2 个乘积项和 1 个可编程寄存器。

ESB2：可实现各种内存块：双端口 RAM、ROM、FIFO、CAM。一个 ESB 可提供 2048bit Memory，多个 ESB 可合并成大块 RAM。

27) APEX20K IOE->LVDS
两个 PLL 可以被配置用于 LVDS 发送器和接收器的接口

29) APEX20K 电源
VCCINT：固定，2.5V 或 1.8V(KE)
VCCIO：多电压 IO 支持，可配置：1.8V(KE)、2.5V、3.3V。
每个 I/O 块可以有自己的 VCCIO 以及电平标准

30) APEX20K ClockLock and ClockBoost
ClockLock：减少 clock delay 和 clock skew，在维持 0 hold time 的同时，减少 Tco 和 Tsu
ClockBoost：20K 固定 2x 或 4x 倍频；20KE 任意放大或减小输入时钟频率
clock skew：时钟延迟到不同寄存器的差异。

Fclock0 = (m / (n * k)) * Fin

Fclock1 = (m / (n * v)) * Fin

31) 外部时钟输出模式
零延时缓冲：外部输出时钟与时钟输入脚相位一致，没有延时。相位移动不允许，只能分频不能扩频。
外部反馈：时钟输入脚的时钟与反馈输入脚的时钟相位一致，没有延时。相位移动不允许，只能分频不能扩频。可消除器件间 clock delay 和 clock skew。
正常模式：输出 时钟与输入时钟具有一定的相位延时。

32) APEX20K ClockShift
可编程的时钟延时和相位移动，90°、180°和 270°可精确实现，其他延时精度在 0.5~1ns，其精度与输入频率、用户输入的倍频和分频因子有关
相位移动只能是输出滞后输入。
输入与输出时钟的倍频或分频必须是整数倍关系

33) CAM（内容可寻址寄存器）和应用
由 ESB 实现。
对一个给定的数据，得到该数据所在的存储单元地址。输出 match 信号
CAM 搜索，在全部地址的搜索是并行的，在一个时钟周期内可以并行搜索完全部地址。
写一个数（最大 32bit）需要在一个地址上写两个时钟周期。“don’t care”bit 要第三个周期
CAM READ：单匹配、多匹配、快速匹配；编码模式（单匹配）、非编码模式（多匹配）
CAM 的应用了解：

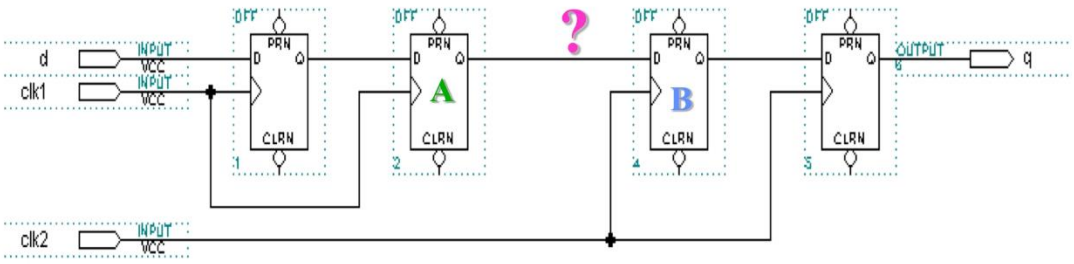
35) SignalProbe & SignalTapII
SignalProbe：在不改变原设计的条件下，利用 FPGA 内部空闲的连线和端口，将用户需要了解的内部信号引出 FPGA 到 PIO。增量布线的特性缩短了硬件验证的过程和 SOPC 的开发时间。
SignalProbe 使用步骤：在设计代码或原理图中增加输出管脚并完整编译；设施输出管脚为 signalprobe output；选择作为 signalprobe 的驱动源信号；SignalProbe 增量编译；时序仿真观察信号时序，或用示波器从输出端口观察波形。

SignalTapII：测量 FPGA 管脚的输入输出数字信号，并在 QuartusII 界面上展示波形。不需要额外的 I/O 端口，设计可以在 FPGA 上全速运行。可以设置触发条件，有更好的精确度，能更好的发现问题。所有抓取到的信号数据都方便的存储在嵌入式 RAM 中，后通过 JTAG 端口将数据传输到计算机进行显示和分析。（传统逻辑分析仪：需要焊接、笨重、价格昂贵）
SignalTapII 使用步骤：对设计代码或原理图完整编译；添加 SignalTapII 文件；设置采样时钟；设置采样深度、缓冲区采集模式以及触发级数等；添加要观察的信号；设置触发条件；完整编译；烧录逻辑；启动 SignalTap 的 run analysis 或 auto analysis，在触发条件满足时可以看到波形；采样得到的波形图保存在.stp 文件中。

37) 逻辑设计技巧
消除毛刺的方法：① 通过调整布线让端口输出延迟一致，会受到外界条件因素影响；② 通过添加触发器让端口输出按时钟同步，从而消除毛刺；③ 通过修改逻辑，采用类似格雷码的设计，减少或消除同一时刻端口同时变化的可能，从根本上杜绝毛刺现象。

建立保持时间：① 通过调整输入信号的相位或延迟，确保数据的建立时间和保持时间满足要求。② 确保输入时钟频率小于等于系统最高工作频率。

异步-同步逻辑设计：异步逻辑设计会在输出信号出现解码毛刺；复位、清零、置位信号建议使用同步逻辑设计。
多时钟系统：



- In order to make sure that B can sample the signal from A, what kind of information needs

  $X \geq T_{co} + T_d + T_{su}$

  - Tco time from register A **(Tco)**
  - trace delay (+ logic delay) between A and B **(Td)**
  - Tsu time of register B **(Tsu)**
  - clock rising edge DIFFERENT BETWEEN clk1 and clk2 **(X)**

38) SOPC Nios II cpu 软核

Nios II 优点：提高系统性能、降低系统成本、功能强大易用的开发工具、使用完全功能的开发包。

Nios II 是通用 RISC 处理核心，具有 32bit 指令集，32bit 数据线和地址线，具备 32 个通用寄存器组和 32 个外部中断资源。

Nios II 处理器系统组成：Nios II 处理器、一系列片上外设、片上存储器、片外存储器接口。

39) Stratix II

基本块：ALM（adaptive logic module）

ALM：包括 ALUTs（adaptive），两个 ALUTs 最多 8 个输入。具有两个程序寄存器、两个专用全加器、一个进位链、一个共享算术链和一个寄存器链

寄存器链线路可以让一个 LAB 中的寄存器级联在一起，形成一个无关联的移位寄存器，在保存本地互联资源时，提高了 LAB 中 ALMs 的连接速度

三种 RAM 块尺寸：可以实现 RAM、ROM、FIFO

DSP Block：乘加、乘累加