数字电路

Digital Circuits and System

李文明

liwenming@ict.ac.cn

中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

中国科学院大学
University of Chinese Academy of Sciences

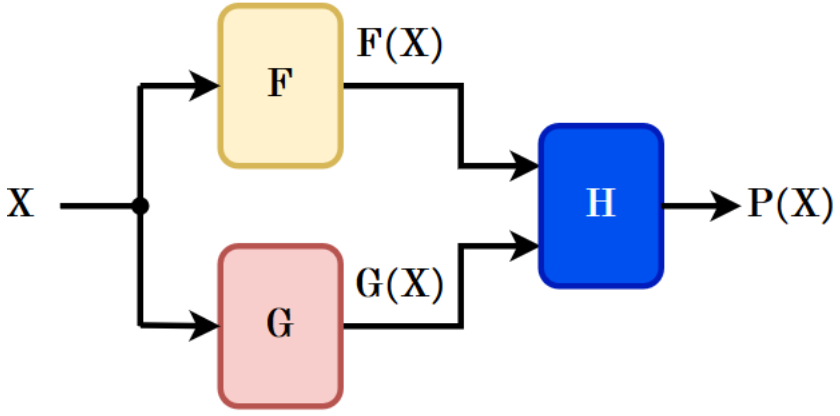
流水线电路

流水线电路

- 流水线的必要性
- 流水线实现方法
- 分层流水线
- 交错电路
- 流水线举例
- 传输控制方法及其分类
- 小结

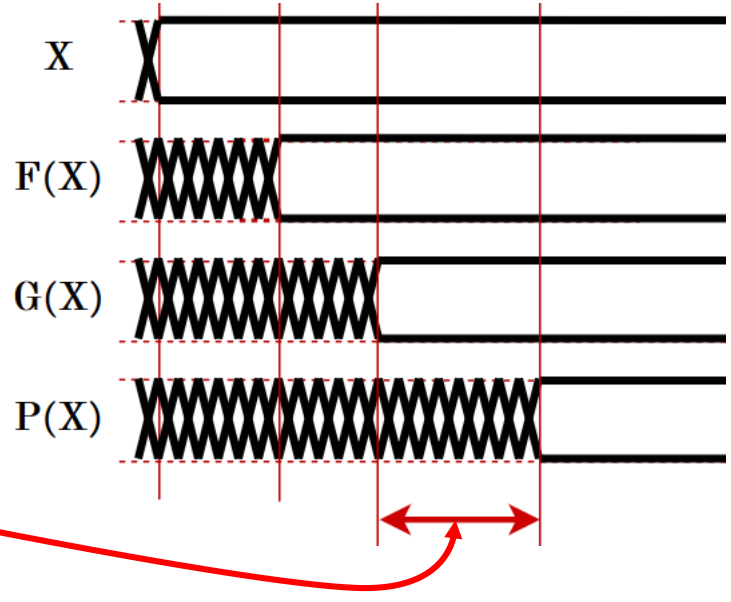
流水线的必要性

- 组合逻辑中
  - 输入到输出的延迟时间：t_{PD}
  - 则其吞吐率 $= {}^{1}/_{t_{PD}}$



- 在 P(X) 处获得结果的速度受限于逻辑模块的延迟时间
- 问题：能否尽可能高效率地使用各个逻辑单元?

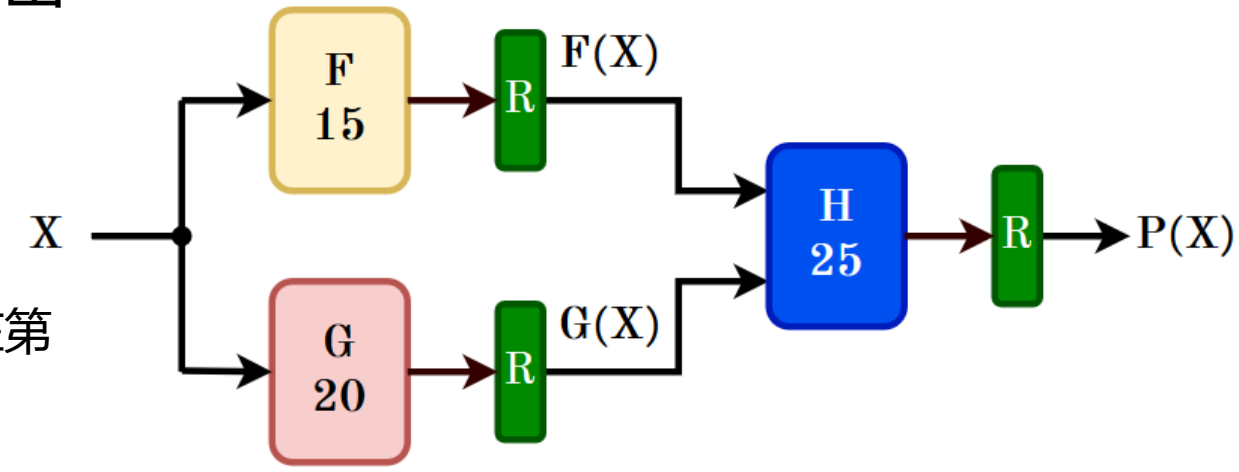
F 和 G 处于空闲，需要保持其输出不变，
以保证H能够计算出 ***P(X)***



流水线电路(Pipeline Circuit)

- 使用寄存器，来保持组合逻辑的输出

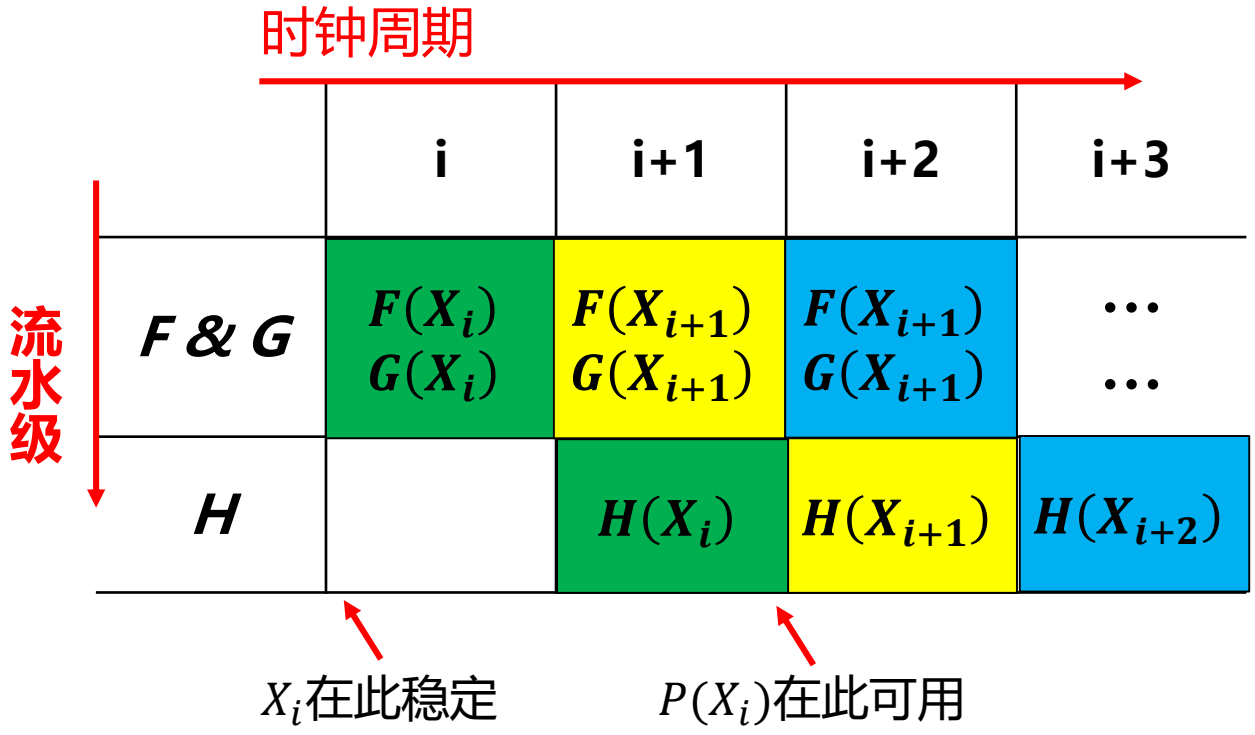
1. 当 ***H*** 计算 X_i 时， ***F 和 G*** 可以计算 X_{i+1}
2. 这是一个包括两个流水级的流水线电路
3. 如果在第 j 个时钟周期，输入值为 X_j ，则在第 j+2 个时钟周期会在输出端获得 $P(X_j)$

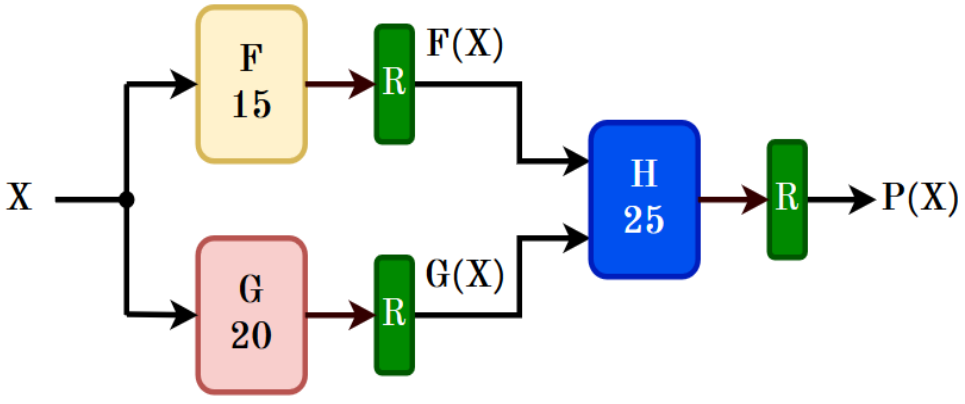


假设 F、G、H 的传输延迟时间分别是：15ns、20ns、25ns，假设寄存器的延迟时间为0

	latency	Throughput
Non-Pipelined	**45ns**	**1/45**
2-Stage Pipeline	**50ns**	**1/25**
	延时增加	吞吐率提高

流水线图(Pipeline Diagram)





对于一组特定的输入，其相应的输出按照流水线图的对角线方向流出得到
每个时钟周期都就可以获得一个输出结果

流水线的惯例(Pipeline Convention)

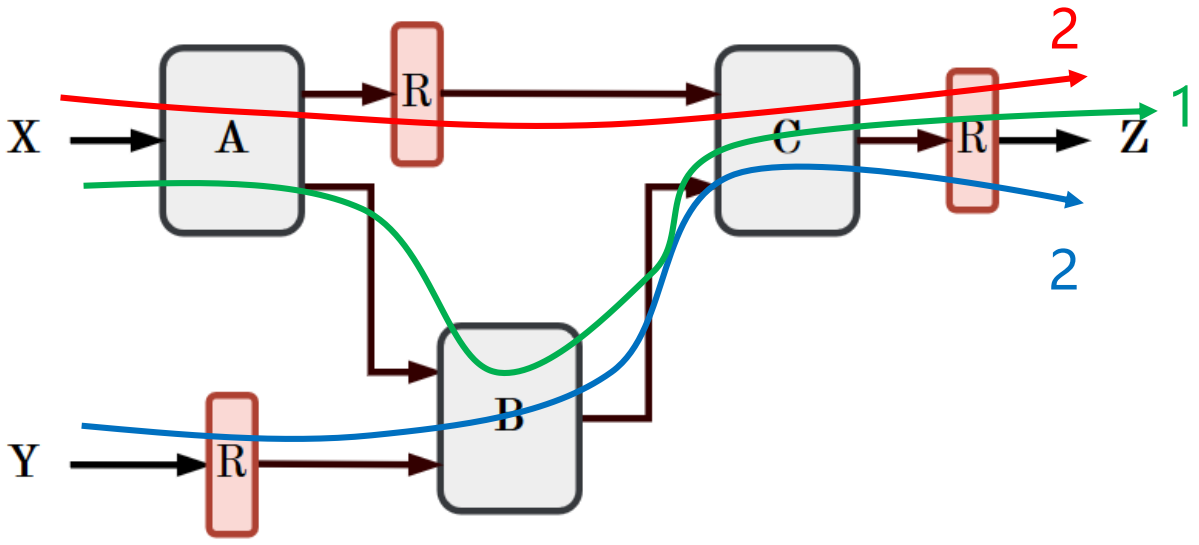
- 定义
  - 一个优化设计的 ***k*** 级流水线(“***k-Pipeline***”)是一个在每个输入到输出的路径上都有***k***个寄存器的非循环连接电路
  - 组合逻辑电路的流水级是 ***0***
- 构成惯例
  - 每个流水级，以及第k个流水级，都在输出端包含有寄存器，输入端不设寄存器
- 总是
  - 时钟信号连接到所有寄存器
  - 时钟信号的周期要大于路径上的组合逻辑传输延迟时间+寄存器的传输延迟时间+寄存器的建立时间

***k*级流水线的延迟时间：** $Latency = k \times T_{clock}$

***k*级流水线的吞吐率：** $Throughput = f_{clock}$

不合理的流水线设计

- 考虑如图流水线结构



该结构是几级流水线电路?

答案：都不是

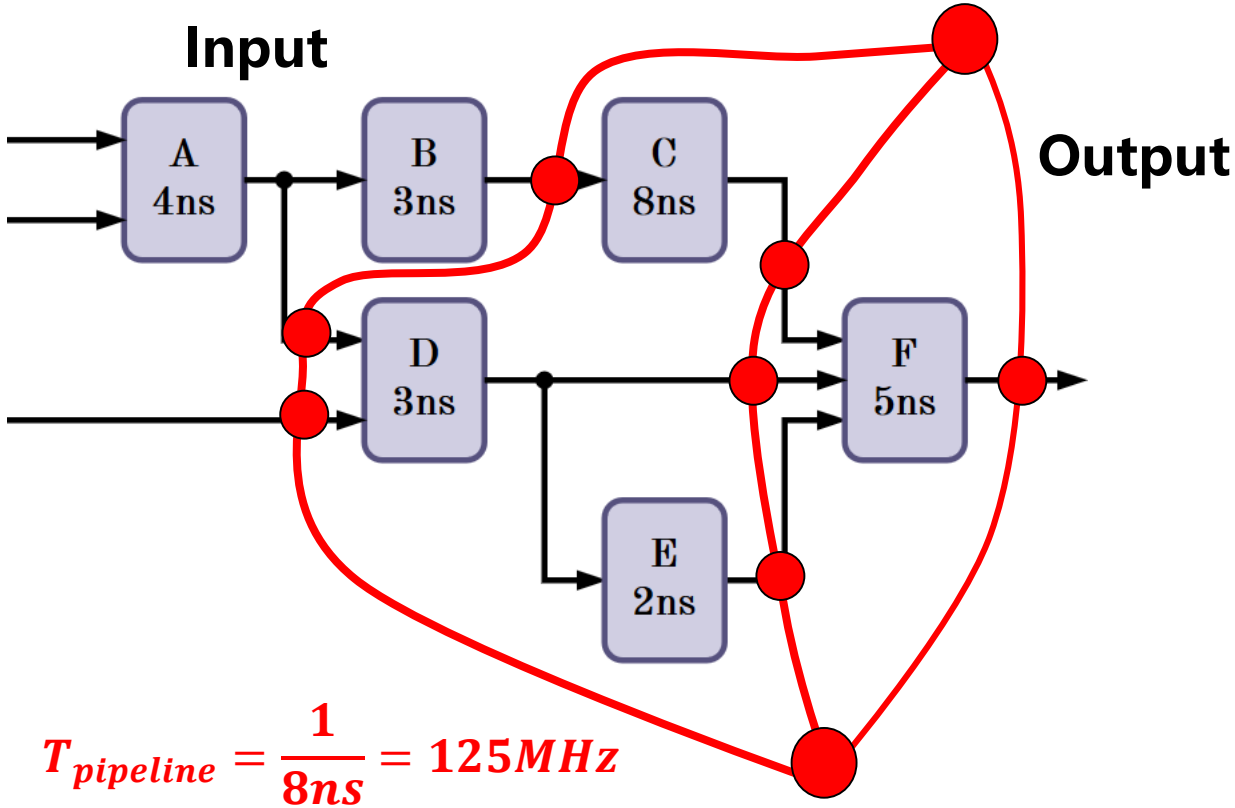
该电路存在的问题

1. 连续两个时钟周期内的数据在组合逻辑中发生了混合，如$B(A(X_{i+1}), Y_i)$
2. 原因在于不同路径上的寄存器数量不一致，有的是“1”，有的是“2”

设计合理的流水线不允许上述情况发生

流水线化方法(Pipelining Methodology)

- Step1
  - 划一条经过电路中的每一个输出端的线，在与输出端相交的地方做标记
- Step2
  - 从右向左重复Step1，遍历模块间的连线，这些轮廓线就把电路的输入、输出、以及中间部分分成了不同区域，从而确定了电路的流水级，
- Step3
  - 在轮廓线与模块间连线的交点处插入寄存器
  - 流水线的时钟周期取决于延迟最大的模块



$$T_{pipeline} = \frac{1}{8ns} = 125MHz$$

$$Latency = 3 \times T_{pipeline} = 24ns$$

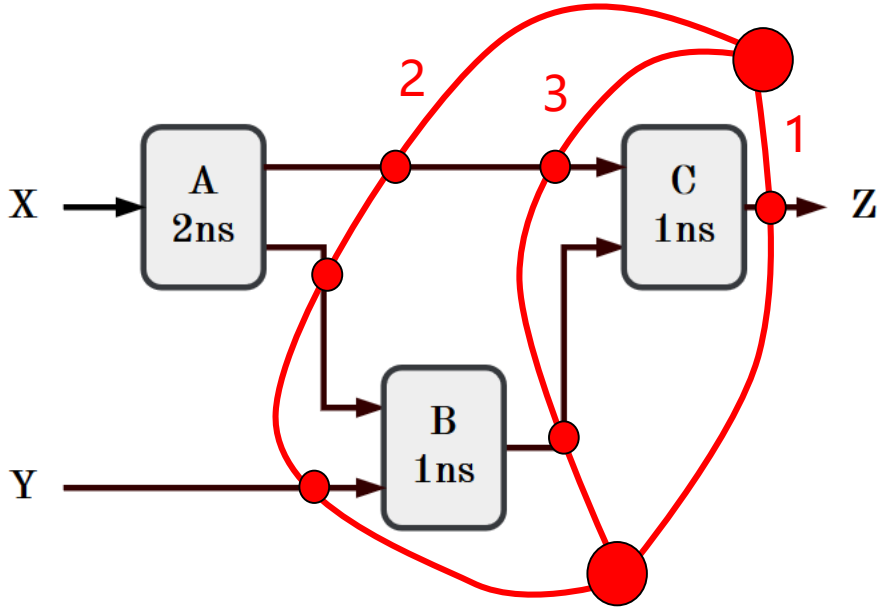
关键点：把寄存器插入在电路中延迟时间最长的关键路径上(Bottleneck)

流水线举例

- 观察结果
  - 1级流水线对延迟时间和吞吐率没影响
  - 2级流水线可以提高吞吐率，不影响延迟时间，原因在于把关键路径上的长延迟组合逻辑分开，从而提高了电路的工作频率
  - 更多的流水级不能改善吞吐率，同时会增加延迟时间
  - 为改善流水线性能，电路中常需要直接级联的寄存器?

解决电路的瓶颈问题（Bottleneck）加入流水线

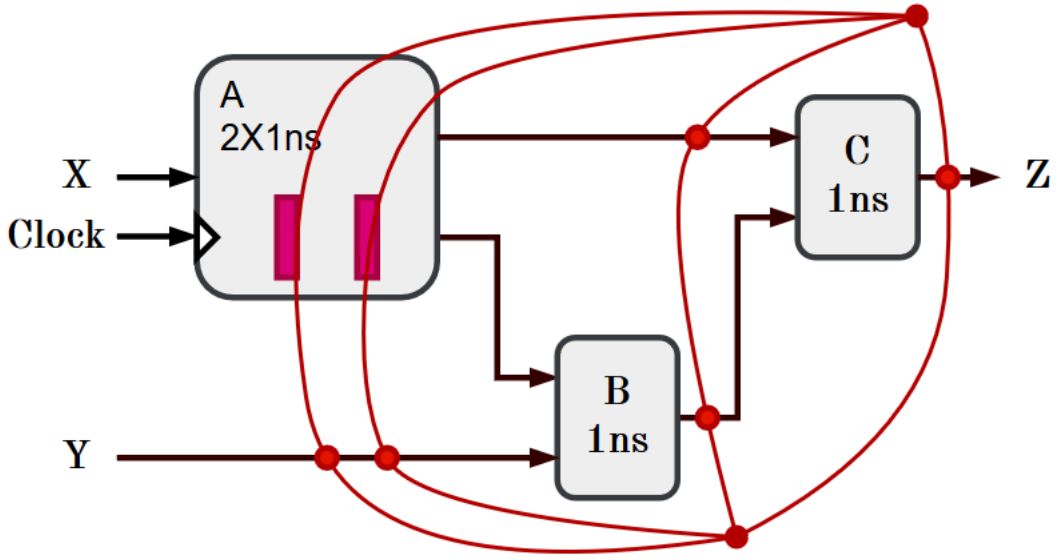
- 优点：提高电路的吞吐率
- 缺点：也会增加电路的延迟时间



	Latency	Throughput
0-pipeline	4	1/4
1-pipeline	4	1/4
2-pipeline	4	1/2
3-pipeline	6	1/2

流水线化的逻辑模块

- 流水线化的数字电路系统可以分层实现
  - 把长延迟时间的组合逻辑模块，先分成 ***k*** 级流水线结构，这样可以降低流水线的时钟周期
  - 在计算总的数字系统流水级时，要把其中子模块的流水级计算在内



4级流水线，$Throughput = 1, Latency = 4ns$

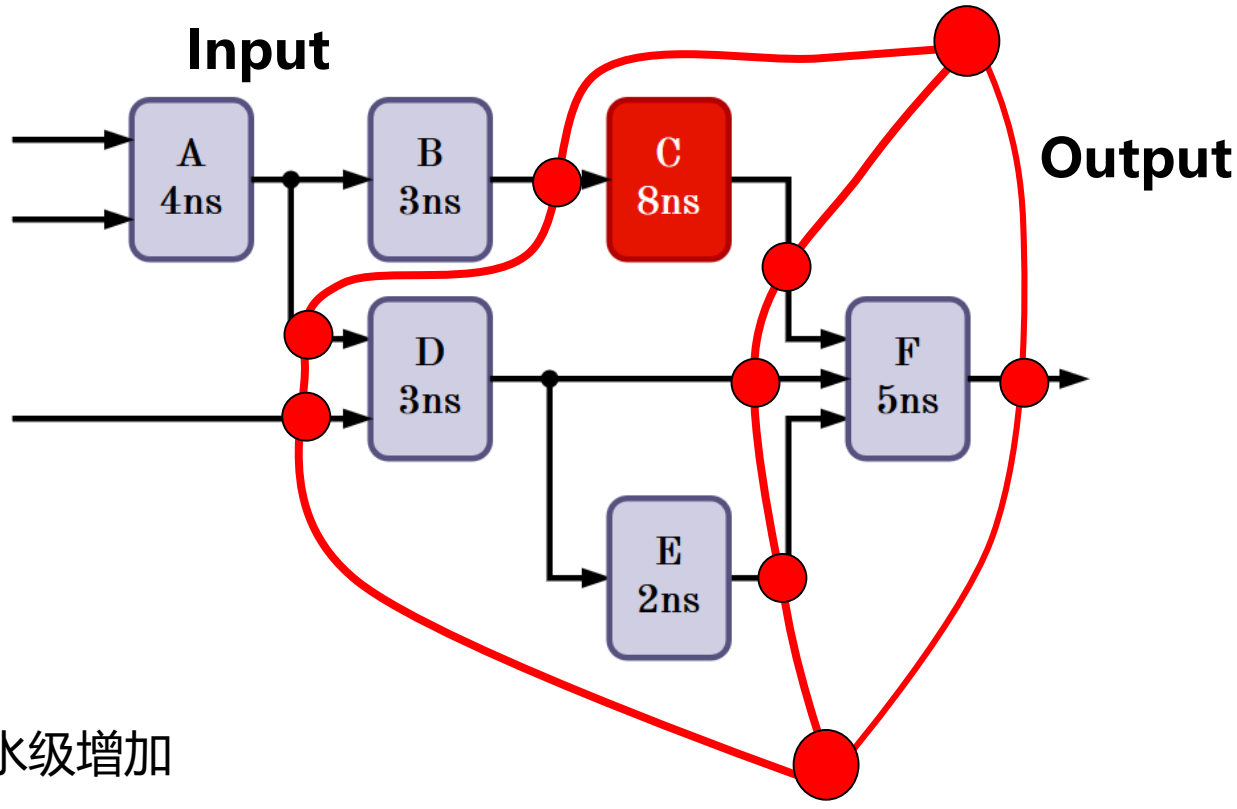
考虑前面的例子

其中子模块C的延迟时间最长：8ns

因此整个系统的吞吐率为：1/8=125MHz

$$T_{pipeline} = \frac{1}{8ns} = 125MHz$$

$$Latency = 3 \times T_{pipeline} = 24ns$$



对此系统如何进一步提高其性能，即提高吞吐率？

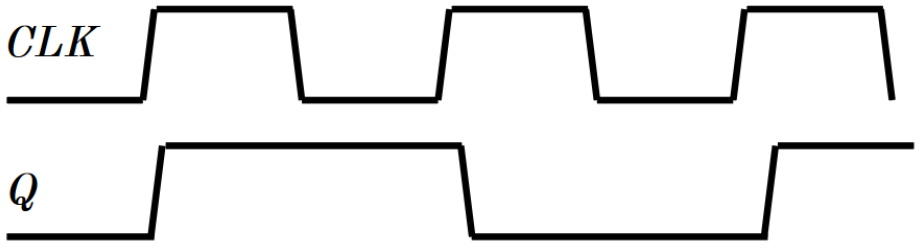
1. 寻找延迟时间更小的子模块C
2. 寻找就有流水线的子模块C，但整个系统的流水级增加
3. 把子模块C的输出复制，让下级模块交替使用（空间换时间）

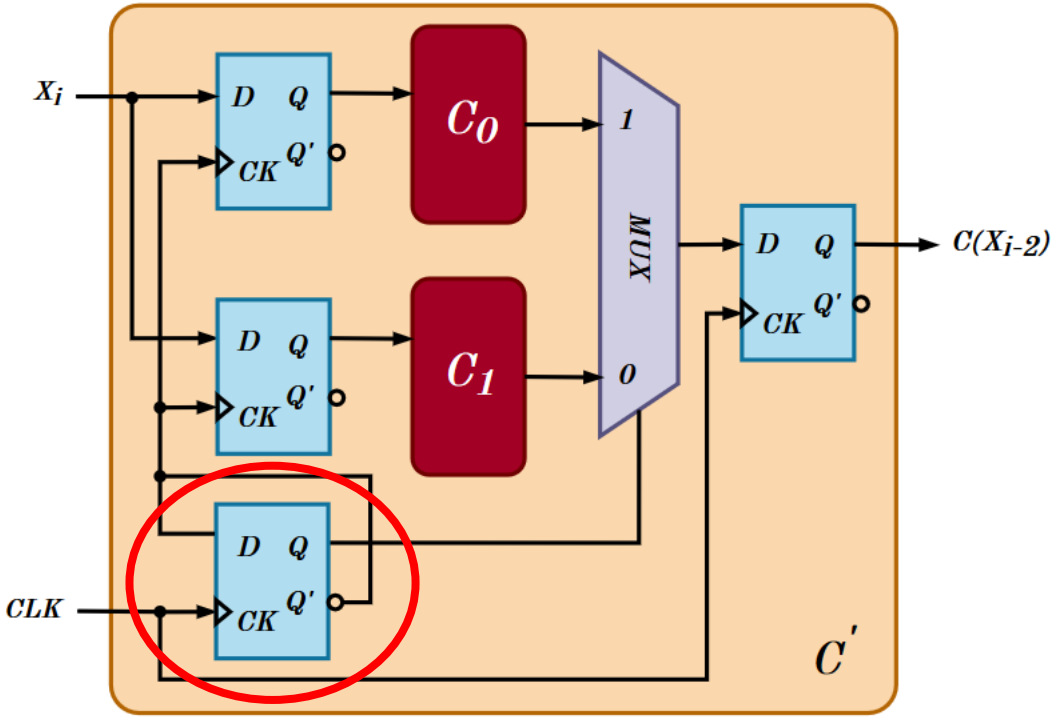
交错电路(Circuit Interleaving)

通过复制数字系统中关键路径上延迟时间最长的组合电路子模块，并把输入值复制后交替提供给复制的子模块使用

在输出端交替选择复制子模块的输出

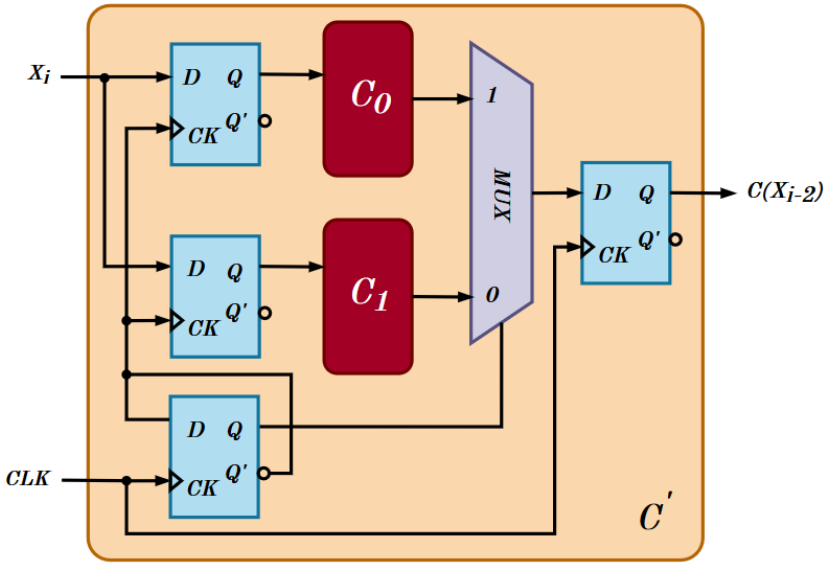
这是一个2级流水线结构

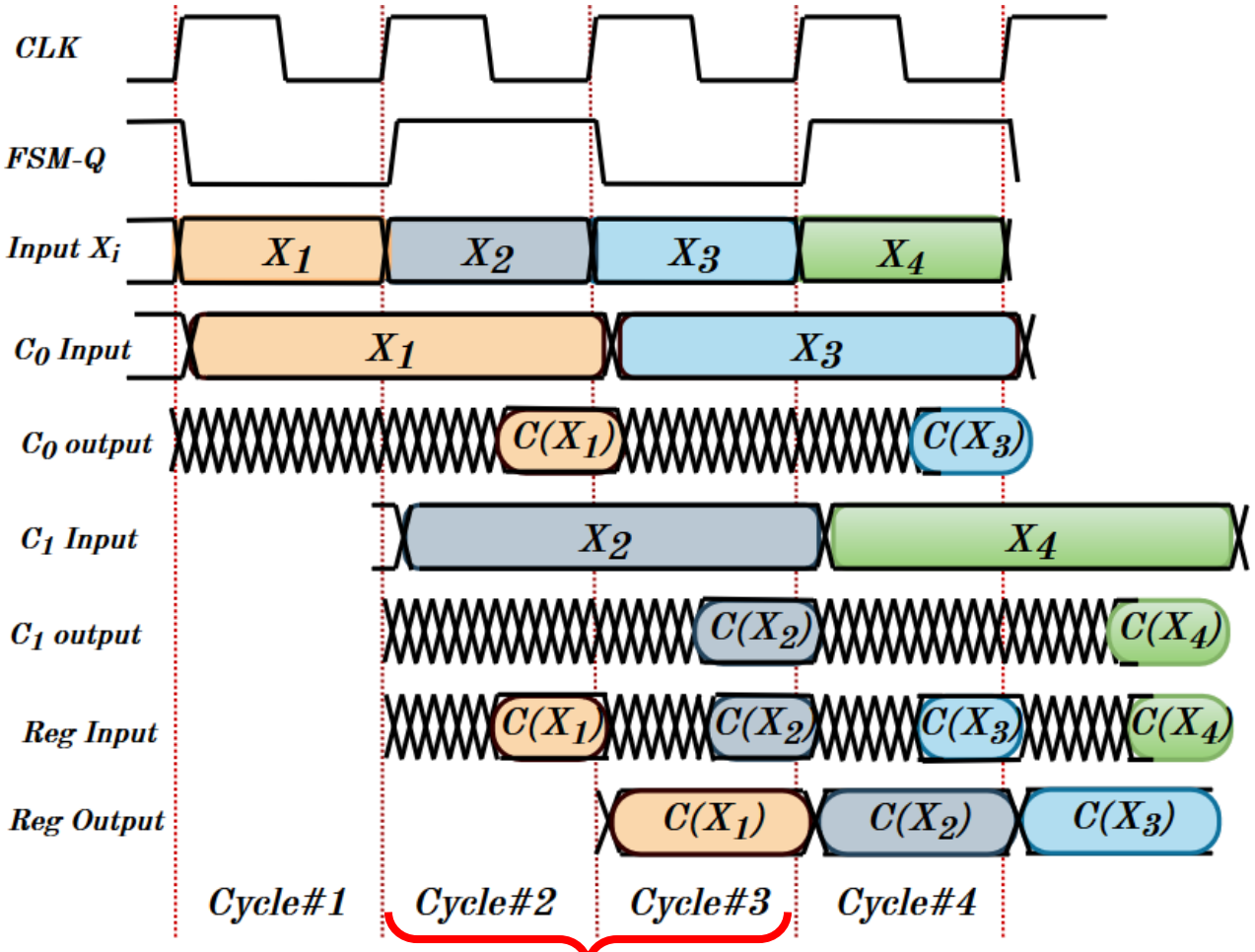




简单的2状态的FSM，实现交替控制

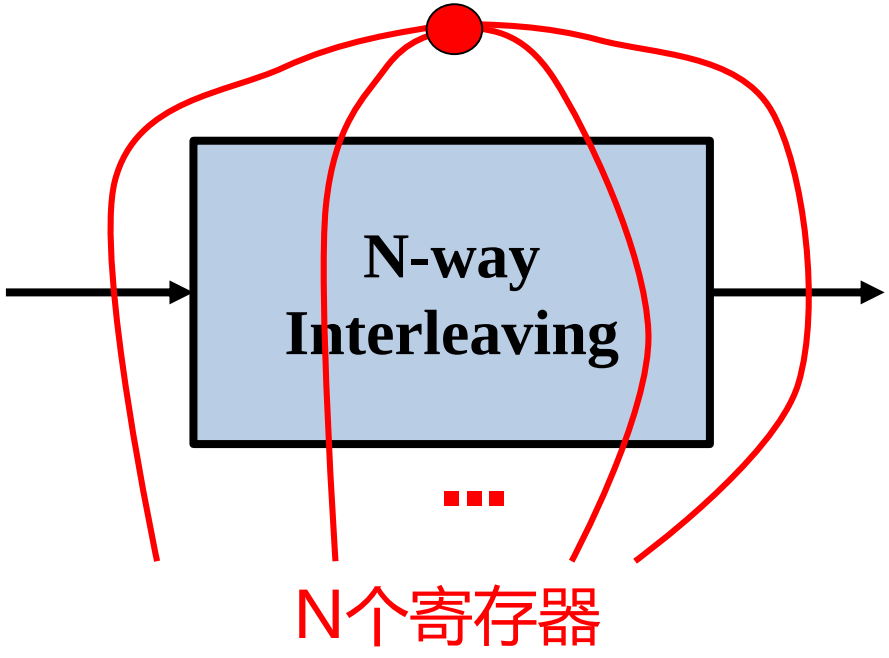
交错电路时序图



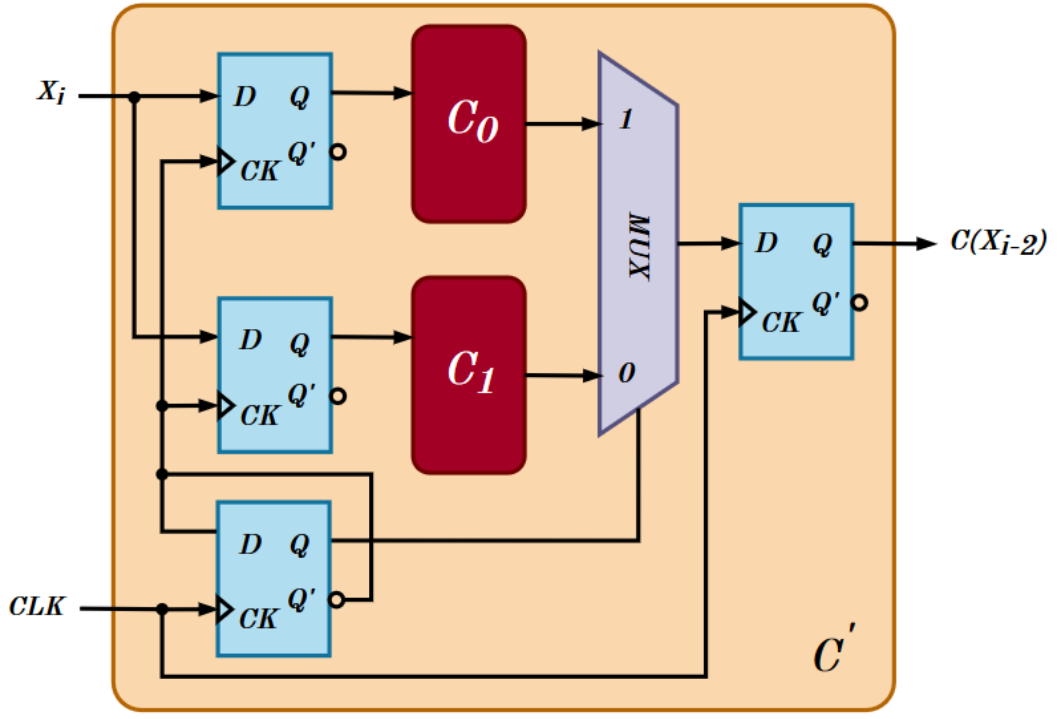


$$2 \cdot t_{clk} \geq (t_{PD.upstreamREG} + t_{PD.Latch} + t_{PD.C} + t_{PD.MUX} + t_{Setup.REG.})$$

交错电路概念扩展



N个寄存器



N路交错等价于N级流水线

2路交错电路，在t_i时刻的输入，需要延迟2个时钟周期，在t_{i+2}时刻在输出端得到结果

$$Throughput = 1/T_{clock}$$
$$Latency = 2 \times T_{clock}$$

多种流水线技术共用

可以结合交错电路和流水线分级技术共用来提高电路的吞吐率

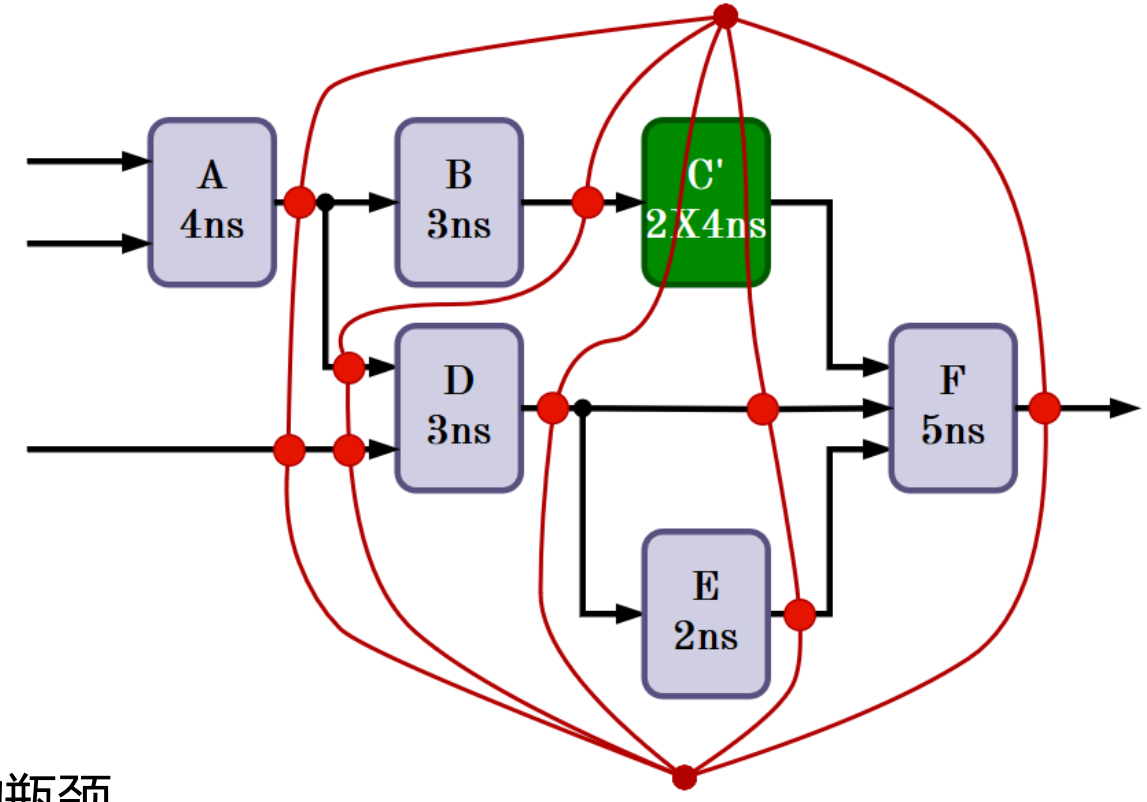
C' 子模块采用交错电路实现，相当于分成了2级流水线

$$Throughput = 1/4ns = 250MHz$$

$$Latency = 2 \times 4ns = 8ns$$

因为C'内包括2个流水级，分割流水级的切分线要经过C'两次

通过采用Interleaving和pipelining技术，电路中的瓶颈(Bottleneck)从模块C转移到了模块F



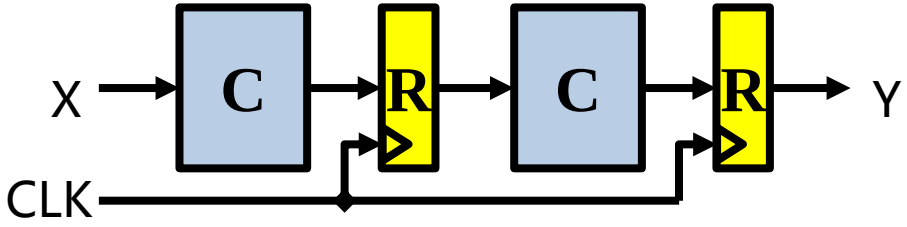
$$Throughput = 1/5ns = 200MHz$$

$$Latency = 5 \times 5ns = 25ns$$

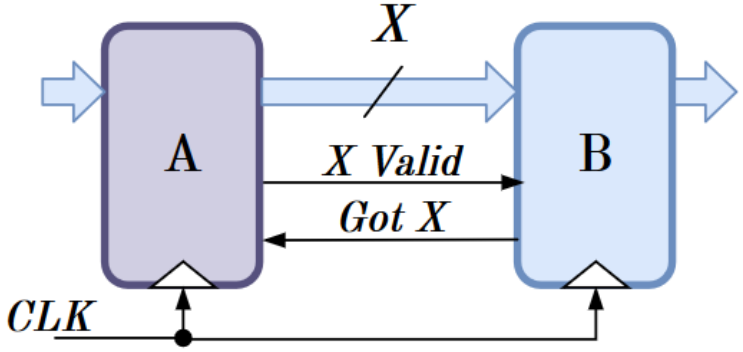
整个电路系统变成了一个5级流水线电路，流水线时钟周期为5ns

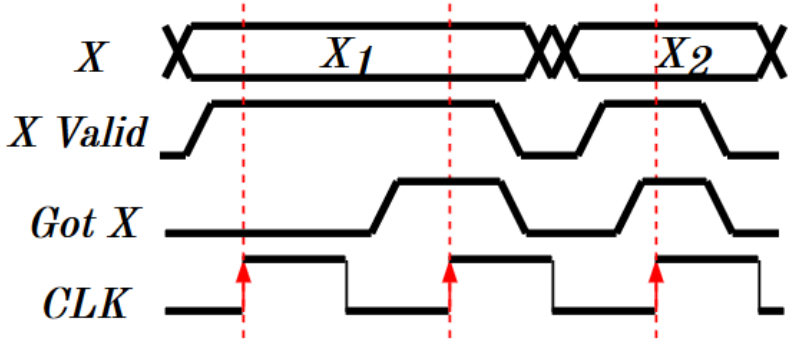
异步传输控制(Control Structure Alternative)

- 同步传输，全局时钟控制

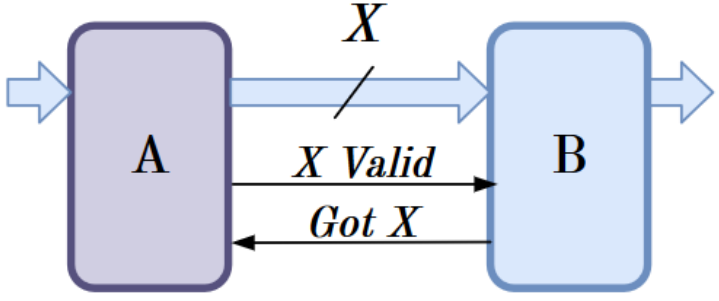


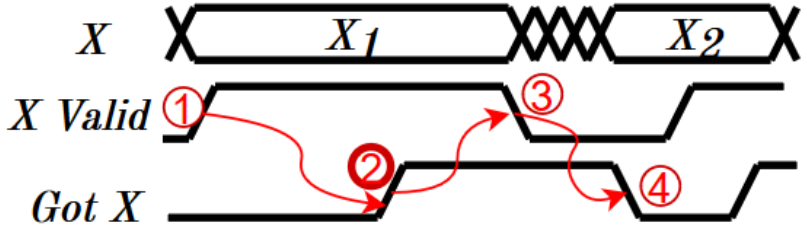
- 同步传输，局部时钟定时，控制收发时序



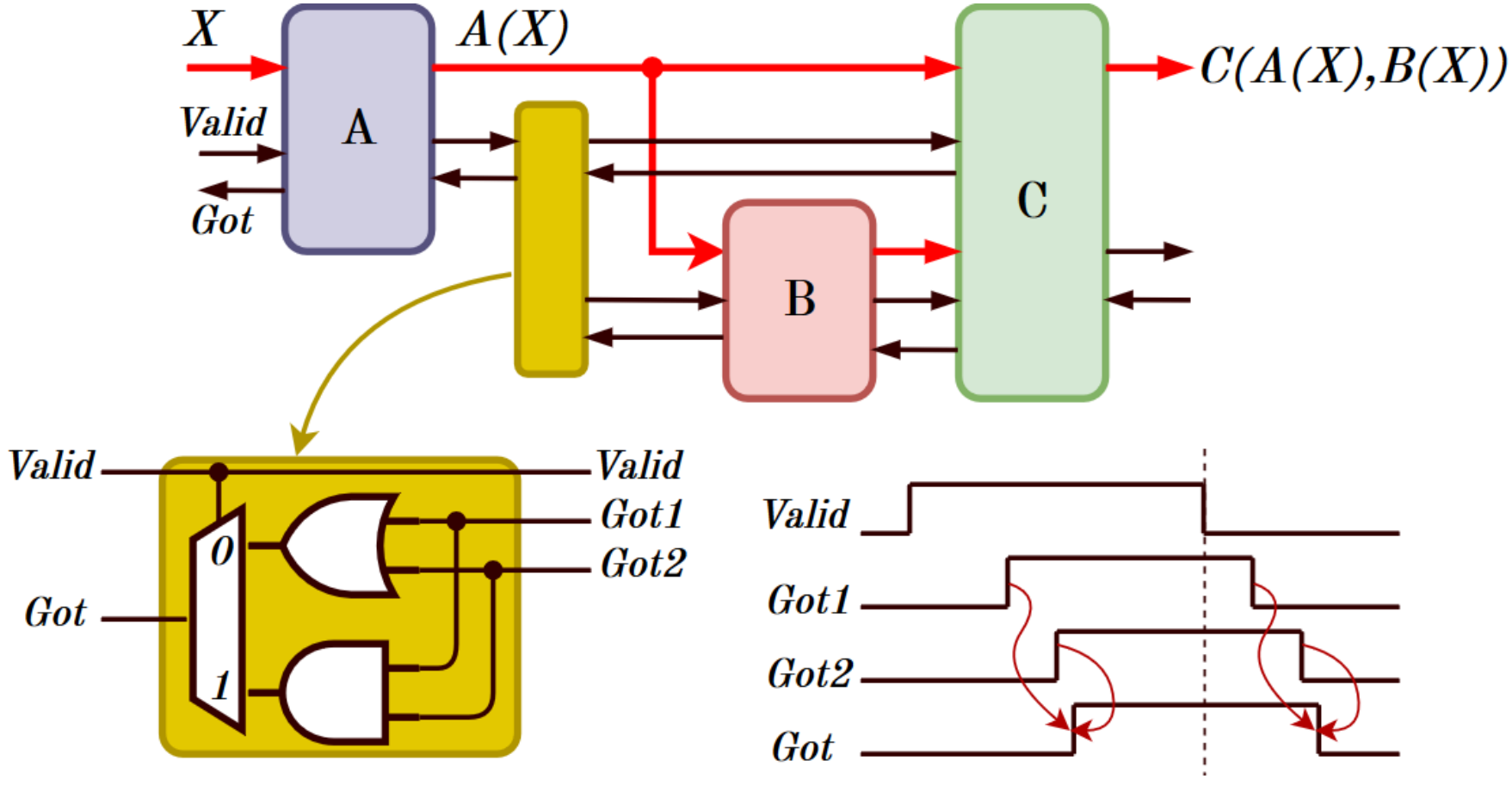


- 异步传输，通过握手信号的电平变化定时，控制收发间的时序



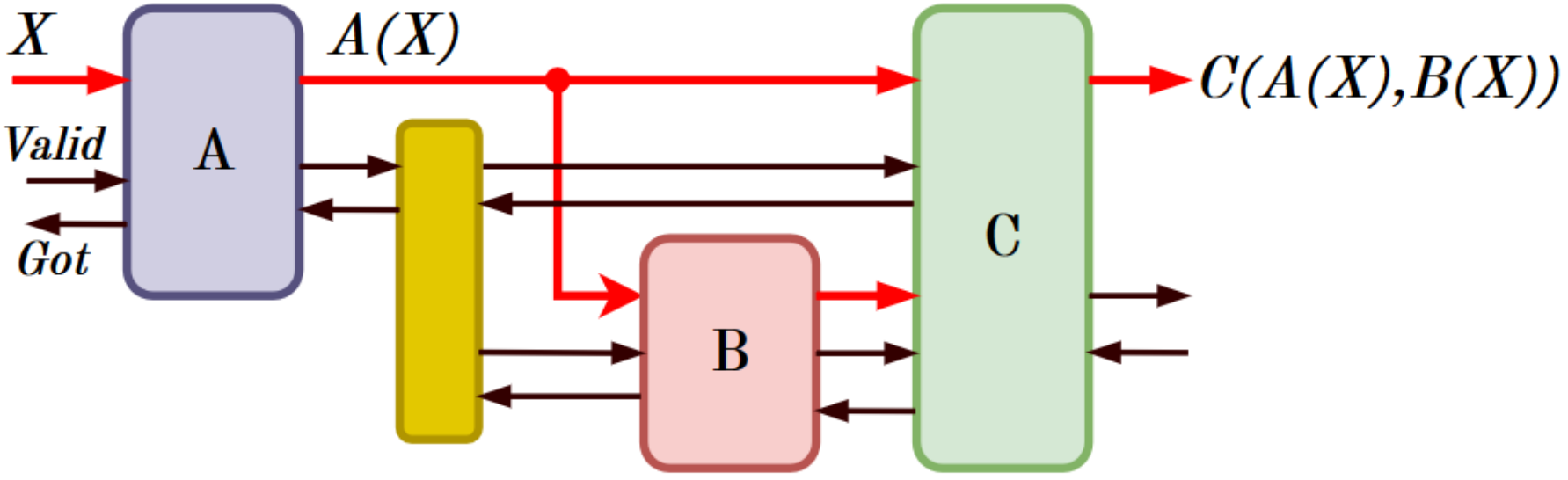


异步传输自定时举例（1）



$Got = 1$, B和C都收到数据；$Got = 0$, B和C都接收数据完成

异步传输自定时举例（2）



好的时间无关电路设计原则

- 对电路中的各个子模块的时序约束相互独立
- 各模块独立处理特例情况（如乘数为“0”）
- 提高各个子模块性能后，整个系统的性能随之改进
- 可以用来实现同步或异步逻辑

传输控制结构分类

易实现，时间间隔固定，会浪费时间
（不依赖于传输的数据有无或长短）

在大规模系统中，系统的定时器生成逻辑变得异常复杂，不可接受

	同步(Synchronous)	异步(Asynchronous)
全局定时 Globally Timed	在全局时钟作用下，中央控制状态机产生所有控制信号	中央控制单元为当前控制任务量身定制所需的时间片
局部定时 Locally Timed	在全局时钟作用下，每个主要的子系统生成与时钟同步的开始和结束控制信号	各个子系统异步地接收开始信号，生成异步完成信号 （可能在本地时钟的控制下）

构造大规模系统的最好方法，各个字系统可以使用独立定时器来控制数据传输，效率高，全局可控

几十年来的“大创新想法”，在一些特殊场景下采用，还有很多工作需要研究

小结

- 延迟时间(Latency，L)：从给定的输入有效，到输出有效所经历的时间间隔
- 吞吐率(Throughput，T)：输出端更新数据的速率
- 对组合逻辑来说：$L = t_{PD}, T = 1/L$
- 对 ***k (k>0)*** 级流水线：
  - 输出总是包含寄存器(Register)
  - 从输入到输出的每条路径上，都包含 ***k*** 个寄存器
  - 在 $Clock_i$后的短时间，输入有效；在 $Clock_{i+k}$后的短时间，输出有效
  - $T_{CLK} = t_{PD.REG} + t_{PD.}$最慢流水级延时 $+ t_{SETUP}$
  - 吞吐率 $T = 1/T_{CLK}$
    - 更高的吞吐率 → 把慢的流水级分成更细的流水级
    - 如果不能分成更多流水级，可以采用复制逻辑模块，交错电路进一步降低延迟时间
  - 延迟时间 $L = k \times t_{CLK} = k/T$
    - $L_{piprline} > L_{Combinational\ Logic}$

问题和建议?