期末复习

徐志伟 zxu@ict.ac.cn

提纲

- 期末复习
  1. 计算机学科的研究对象
  2. 计算机学科的基本研究方法
     - 计算思维基本解题思路：PEPS（活力方法论）
     - 对计算思维的十个理解：**Acu-Exams-CP**
  3. 系统思维的“以一耦万”抽象栈思想
  4. 信息隐藏程序回顾：自顶向下设计
  5. 用时序电路设计4位串行加法器、减法器
  6. 网络拓扑：第二代搜索引擎示例
  7. 协议栈：互联网通信示例

- 期中复习的内容不再赘述

课件中可能包含素材引用，特此致谢！

1. 计算机科学的研究对象

比照《计算机科学基础报告》

- “计算机科学是研究计算机以及它们能干什么的一门学科。它研究**抽象计算机**的能力与局限，**真实计算机**的构造与特征，以及用于求解问题的无数**计算机应用**。”
  - 计算机科学涉及**符号**及其操作；
  - 关注多种**抽象**概念的创造和操作；
  - 创造并研究**算法**；
  - 创造各种**人工结构**，尤其是不受物理定律限制的结构；
  - 利用并应对**指数增长**；
  - 探索计算能力的**基本极限**；
  - 关注与人类**智能**相关的复杂的、分析的、理性的活动。

知道概念，能够举例说明其特点

笔记本电脑

有限状态机、图灵机 、
冯诺依曼模型

- “计算机科学是研究计算机以及它们能干什么的一门学科。它研究**抽象计算机**的能力与局限，**真实计算机**的构造与特征，以及用于求解问题的无数**计算机应用**。”
  - 计算机科学涉及**符号**及其操作；
  - 关注多种**抽象**概念的创造和操作；
  - 创造并研究**算法**；
  - 造各种**人工结构**；
  - 用并应对**指数增长**；
  - 探索计算能力的**基本极限**；
  - 关注与人类**智能**相关的复杂的、分析的、理性的活动。

快速排序程序

各种编程抽象和系统抽象

个人作品

逻辑推理
命题逻辑
谓词逻辑

斐波那契递归程序
P与NP

图灵机不可计算问题
哥德尔不完备定理

计算思维概念

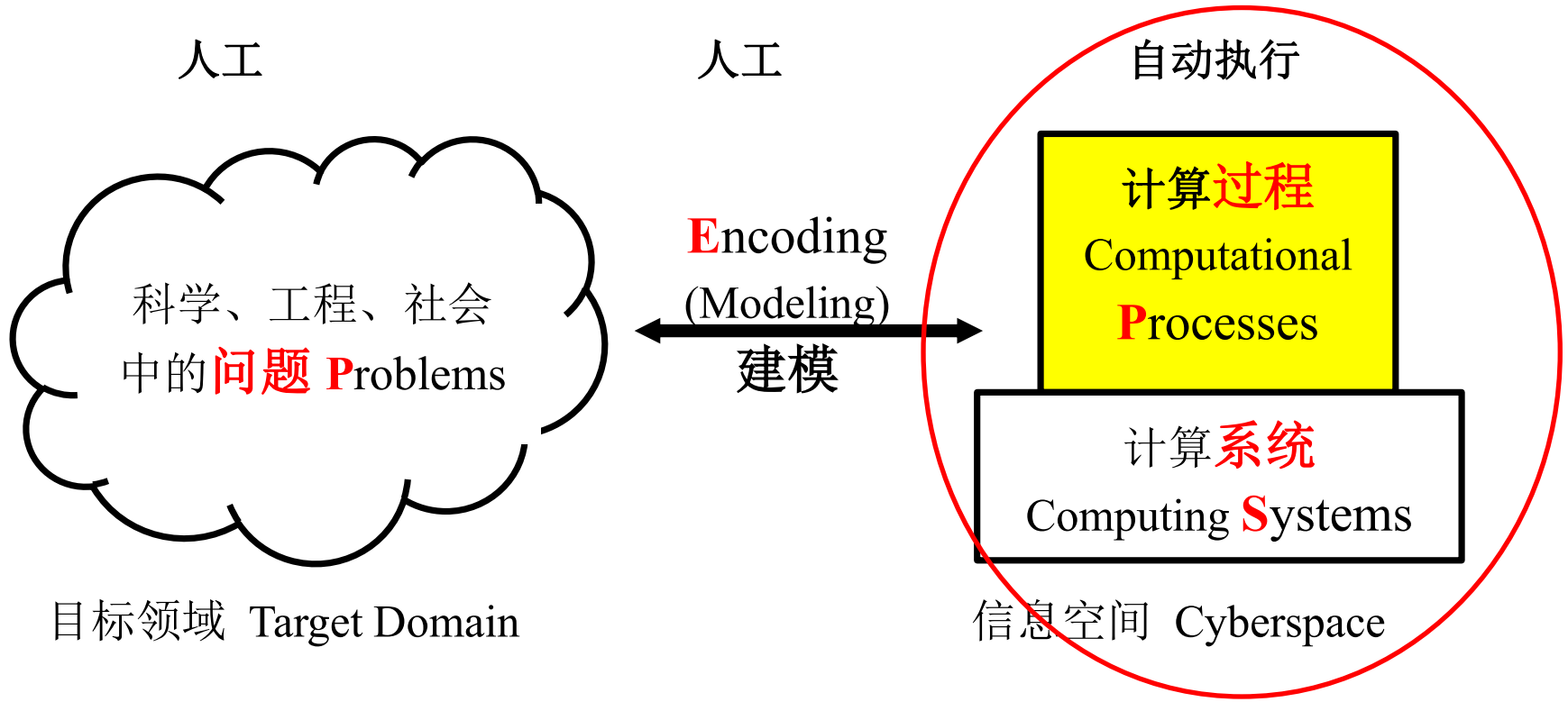
对计算思维的十个理解： Acu-Exams-CP

计算机科学是研究计算过程的科学

(A): Automatically execution	**自动执行**。计算机能够自动执行由离散步骤组成的计算过程。
(C): Correctness	**正确性**。计算机求解问题的正确性往往可以精确地定义并分析。
(U): Universality	**通用性**。计算机能够求解任意可计算问题。
(E): Effectiveness	**构造性**。人们能够构造出聪明的方法让计算机有效地解决问题。
(X): Complexity	**复杂度**。这些聪明的方法（算法）具备时间/空间复杂度。
(A): Abstraction	**抽象化**。少数精心构造的计算抽象可产生万千应用系统。
(M): Modularity	**模块化**。多个模块有规律地组合成为计算系统。
(S): Seamless Transition	**无缝衔接**。计算过程在计算系统中流畅地执行。
(C): Connectivity	**连接性**。很多问题涉及用户/数据/算法的连接体，而非单体。
(P): Protocol Stack	**协议栈**。连接体的节点之间通过协议栈交互。

2. 计算思维基本方法：活力解题法（PEPS）

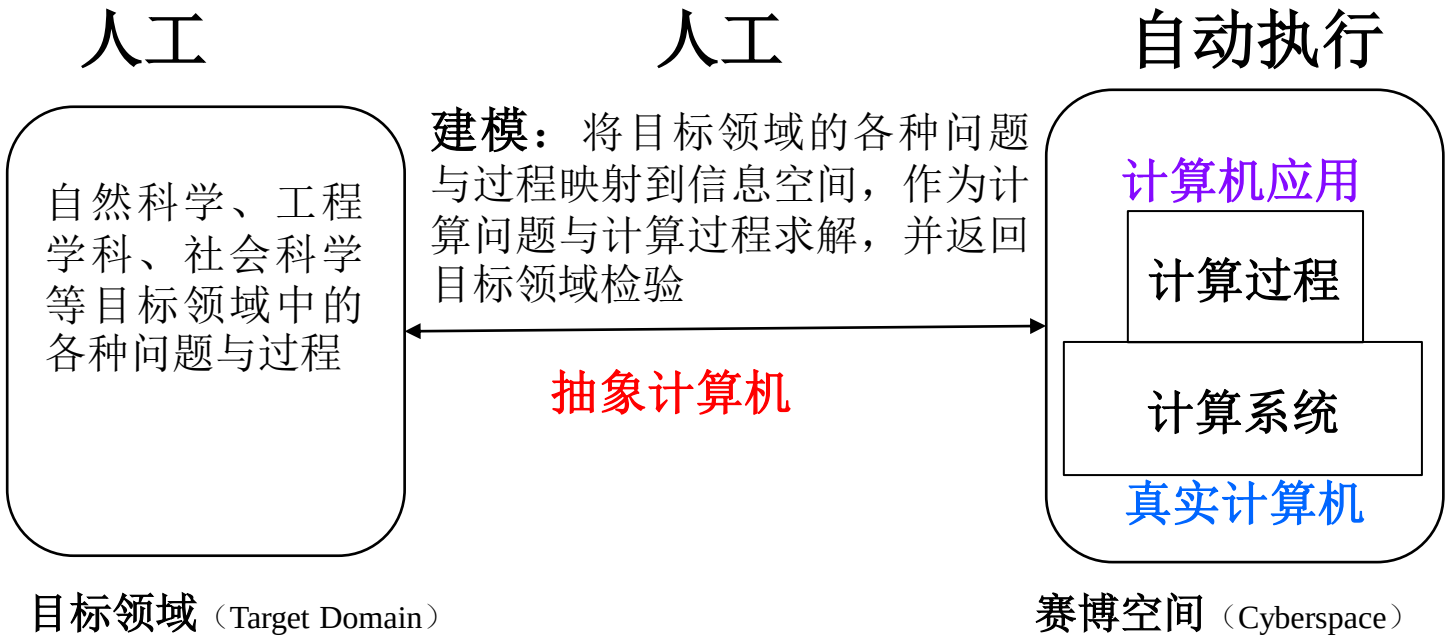
- 定义领域**问题**（**P**roblem in target domain）
- **建模**（**E**ncoding）到信息空间（赛博空间，cyberspace）的计算问题
- 自动执行计算**系统**（**S**ystem）上的计算**过程**（**P**rocess），解决问题
- 映射回到目标领域检验



计算思维基本方法

- 一种认识世界、定义问题、解决问题的科学方法
  - 计算系统 = 计算机
  - 使用算法、程序、状态转移表等刻画计算过程
- 建模是双向映射
- 目标领域也可以是赛博空间

计算机科学研究抽象计算机，真实计算机，计算机应用。

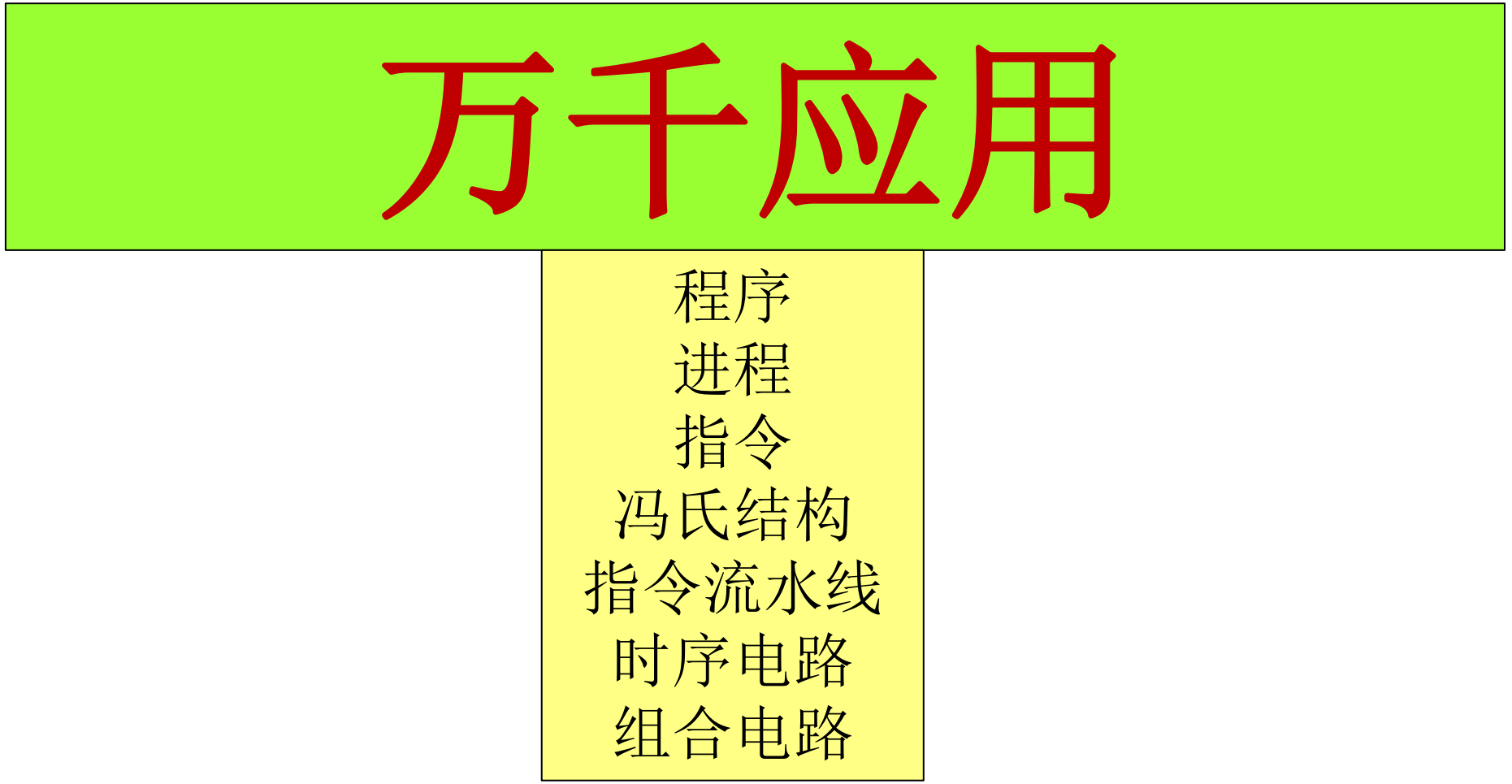


3. 计算系统思维

- 通过**抽象**，将**模块**组合成为系统，**无缝**执行计算过程
  - **抽象化**：一个通用抽象代表多个具体需求
  - **模块化**：系统由多个模块组合而成
    - 计算机 = 硬件 + 系统软件 + 应用软件
    - 信息隐藏原理，接口概念
  - **无缝衔接**
    - 避免缝隙：
      - 扬雄周期原理、波斯特尔鲁棒性原理、冯诺依曼穷举原理
    - 重视瓶颈：阿姆达尔定律
      - 系统性能改进受限于系统瓶颈（针对某任务）
        - 加速比 = $1 / ((1-f)/p + f) \rightarrow 1/f$ 当 $p \rightarrow \infty$

以一耦万

向编程者提供抽象，比特精准地将抽象映射到硬件



本课程学到的一些基本抽象

- 是什么、通过实例解释、能够举一反三

Data Type **数据类型**	bit (1 bit), hexadecimal number (4 bits), byte (8 bits), uint8 (8-bit unsigned integer), integer (64 bits); array (n elements of the same type), slice (a descriptor pointing to an array); text file, BMP image file; hypertext and hyperlink 比特、字节、整数、数组、切片、文本文件、图像文件、超链接	
Software **软件**	Algorithm 算法	Smart method of information transformation, such as quicksort, hiding text in a BMP file, etc. 快排算法
	Program 程序	Code realizing algorithms in computer language, such as hide.go in the Text Hider project 信息隐藏程序 **hide.go**
	Process 进程	Program in execution, such as the "hide" process running in a Linux environment **> ./WebServer &** 系统显示 PID=79
	Instruction 指令	The smallest unit of software, directly executable by computer hardware 班级快排指令集，斐波那契计算机指令集
von Neumann Architecture: a computer model bridging software and hardware 冯诺依曼计算机模型		
Hardware **硬件**	Instruction Pipeline 指令流水线	The basic hardware mechanism to automatically execute any instruction "取指-译码-执行"三级流水线
	Sequential Circuit 时序电路	More precisely, only consider Synchronous Sequential Circuit comprised of combinational circuits and state circuits and driven by a clock signal; equivalent to the automata concept 串行加法器
	Combinational Circuit 组合电路	Also known as Boolean circuit, realizing a Boolean function 使用全加器的波纹进位加法器

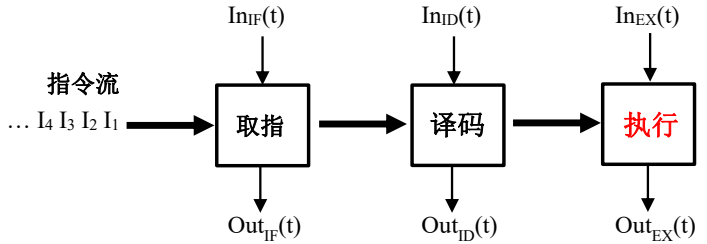
比特精准地将抽象映射到最底层硬件

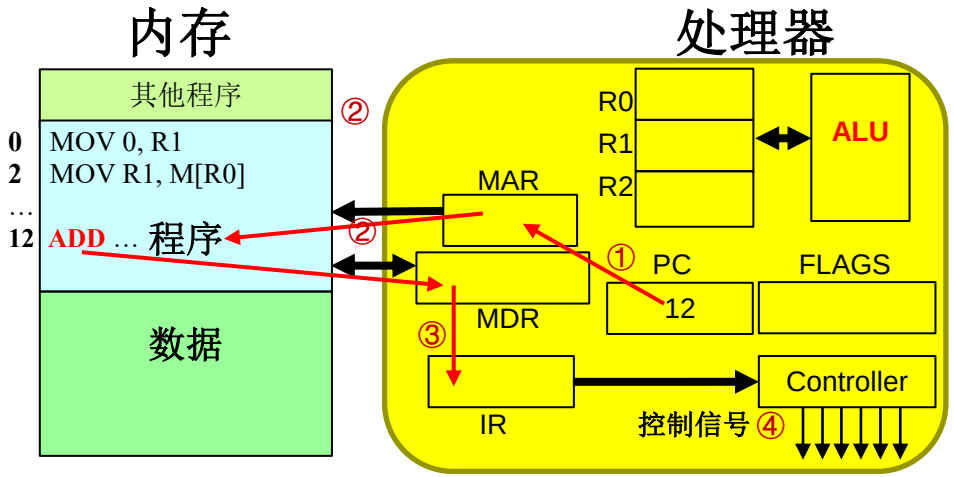
万千应用

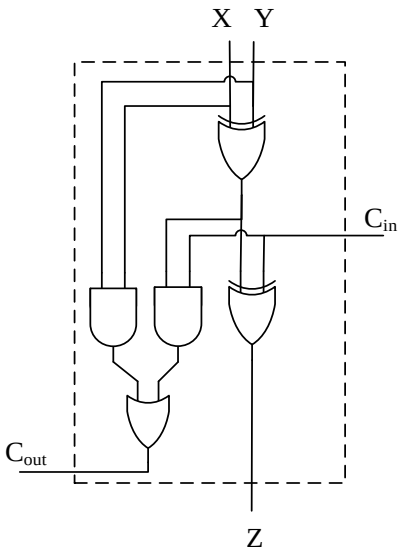
程序
进程
指令
冯氏结构
指令流水线
时序电路
组合电路

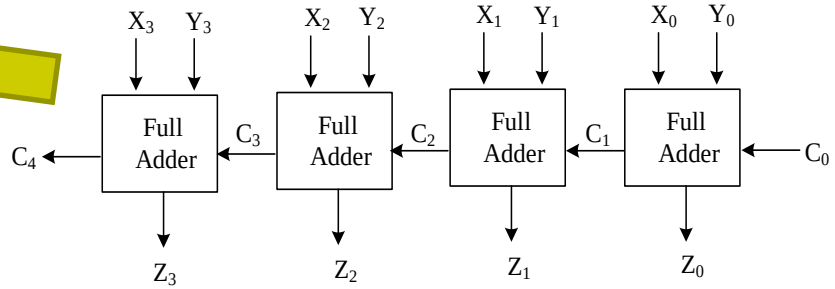
```
MOV 0, R1
MOV R1, M[R0]
MOV 1, R1
MOV R1, M[R0+8]
MOV 2, R2    // i:=2
MOV 0, R1    // label Loop
ADD M[R0+R2*8-16], R1
ADD M[R0+R2*8-8], R1
MOV R1, M[R0+R2*8-0]
INC R2       // i++
CMP 51, R2   // i < 51?
JL Loop   // if '<' goto Loop
```

```
fib[0] = 0
fib[1] = 1
for i := 2; i < 51; i++ {
   fib[i] = fib[i-1] + fib[i-2]
}
```









4. 信息隐藏程序回顾

- Write a program hide-0.go
  - to hide the text of Shakespeare's *Hamlet*
    - in a picture file Autumn.bmp
    - such that the doctored file shows no visible difference from the original picture
- and another program show-0.go to recover the text

HAMLET

DRAMATIS PERSONAE

CLAUDIUS king of Denmark. (KING CLAUDIUS:)

HAMLET son to the late, and nephew to the present king.

…..

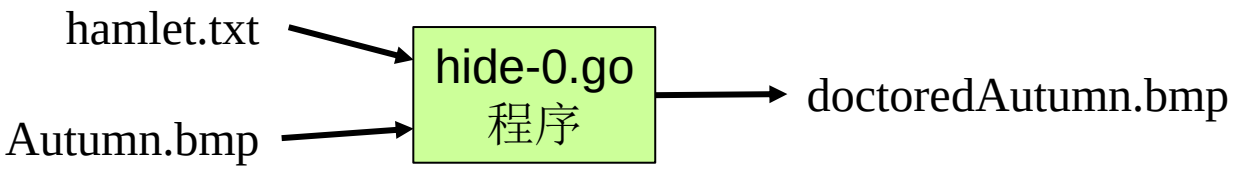
ACT ISCENE I Elsinore. A platform before the castle.
…..
PRINCE FORTINBRAS Let four captains
Bear Hamlet, like a soldier, to the stage;
For he was likely, had he been put on,
To have proved most royally: and, for his passage,
The soldiers' music and the rites of war
Speak loudly for him.
Take up the bodies: such a sight as this
Becomes the field, but here shows much amiss.
Go, bid the soldiers shoot.
[A dead march. Exeunt, bearing off the dead
bodies; after which a peal of ordnance is shot off]
…..
] 换行键0x0A

Original picture

After careless hiding

After careful hiding

程序hide-0.go的设计思路：自顶向下



信息隐藏算法

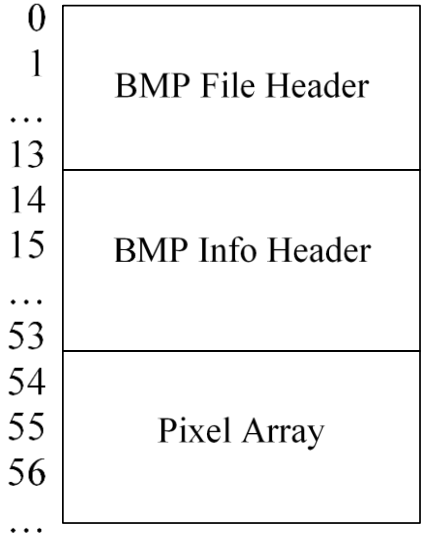
- **输入：**文本文件hamlet.txt 与图像文件 Autumn.bmp。
- **输出：**新图像文件doctoredAutumn.bmp，它隐藏了hamlet.txt的全部内容，且与Autumn.bmp无可见差别。
- **计算过程的步骤：**

1.	将文本文件hamlet.txt读进变量t	// t 代表文本（text）	读文件
2.	将图像文件Autumn.bmp读进变量p	// p 代表图像（picture）	读文件
3.	将hamlet.txt的文件长度隐藏到变量p中Pixel Array的头32个字节		隐藏文件长度
4.	将hamlet.txt的文件内容隐藏到变量p中Pixel Array的剩余字节		隐藏文件内容
5.	将p写到新文件doctoredAutumn.bmp		写文件

要点1：将文件以字□ 切片的方式□ □ 内存，以便定位和操作

要点2：使用“基址+索引+偏移量”寻址模式，便于循环

Autumn.bmp



在图像中隐藏文本

- 代码
  - p, _ := ioutil.ReadFile("./Autumn.bmp")
    to read the image file into byte slice p
  - t, _ := ioutil.ReadFile("./hamlet.txt")
    to read the text file into byte slice t
  - 使用循环语句隐藏t[i], i = 0~len(t)

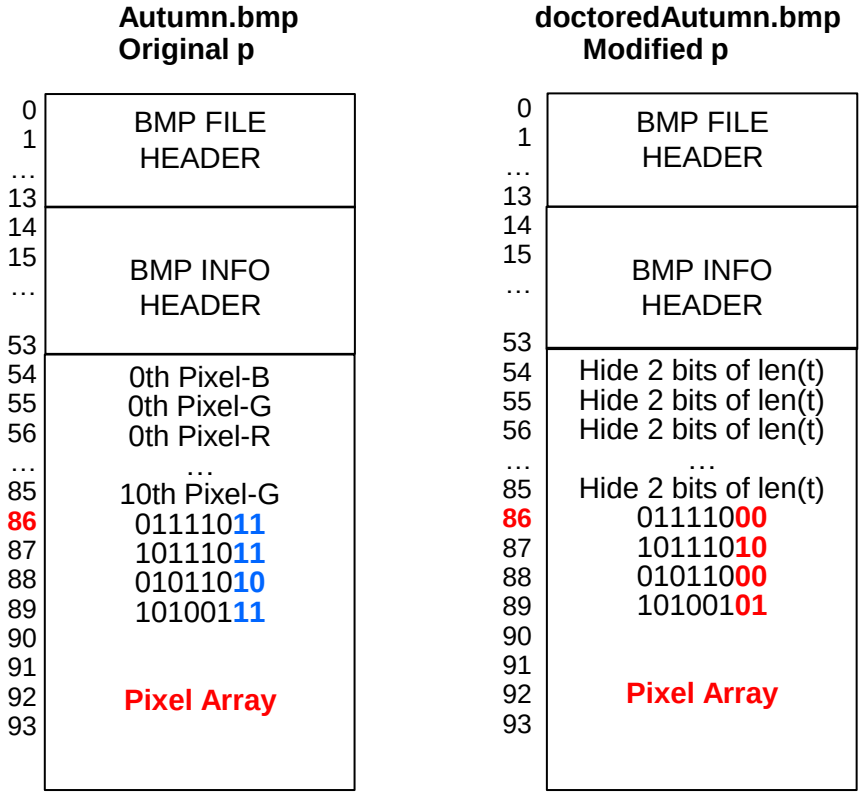
```
for i:=0; i<len(t); i++{
   offset := S+T+(i*4)
   modify(int(t[i]), p[offset:offset+C], C)
}
```

比例因子是4，因为每个字符有8位，需要p的4个字节。这4个字节的偏移量分别是0、1、2、3。第0次迭代执行modify(int(t[0]), p[86:90], 4)，修改P[86]， P[87]， P[88]， P[89]

基址+索引*4+偏移量

回忆支持循环的基址+索引+偏移量寻址模式

- R1+ M[**R0** +**R2** *8 -16] → R1
  基址 索引 比例因子 偏移量



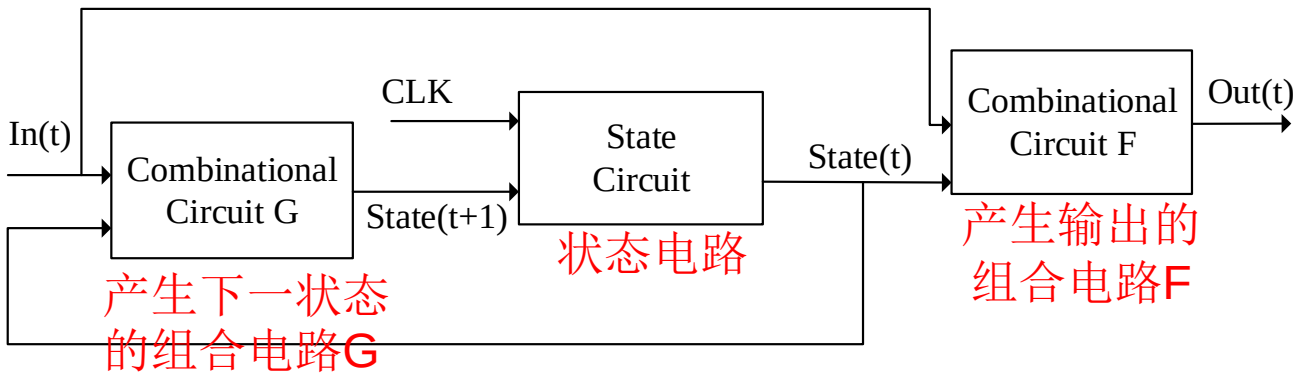
课堂小测验

- 姓名：　　学号：
- 信息隐藏程序hide-0.go读入文本文件hamlet.txt 与图像文件Autumn.bmp，输出另一个图像文件doctoredAutumn.bmp。假设我们想要将一段音频文件myMusic.mp3隐藏在图像文件中，请选择正确做法（）

A. 将doctoredAutumn.bmp替换成myMusic.mp3
B. 将hamlet.txt替换成myMusic.mp3
C. 将Autumn.bmp替换成myMusic.mp3
D. 上述做法都是错的，程序hide-0.go只能将文本文件隐藏在图像文件中，不能隐藏音频文件

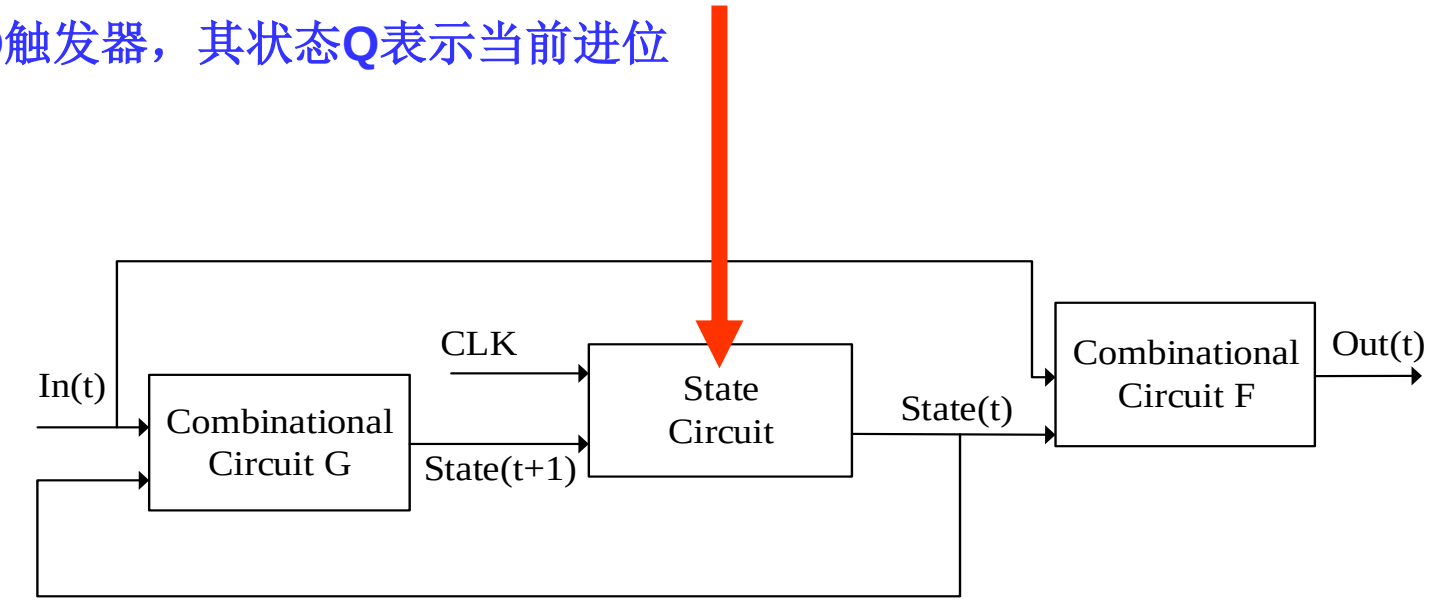
5. 时序电路

- 两个组合电路，一个状态电路；时钟信号驱动
- 状态电路由一个或多个D触发器组成
- 两个组合电路是
  - 输出电路F：Out(t) = F(In(t), State(t))
  - 下一状态电路G：State(t+1) = G(In(t), State(t))
- 时序电路的逻辑线路图（时序电路的一般组成）
  - 也称为时钟同步的时序电路（synchronous sequential circuit）



示例：设计4位串行加法器

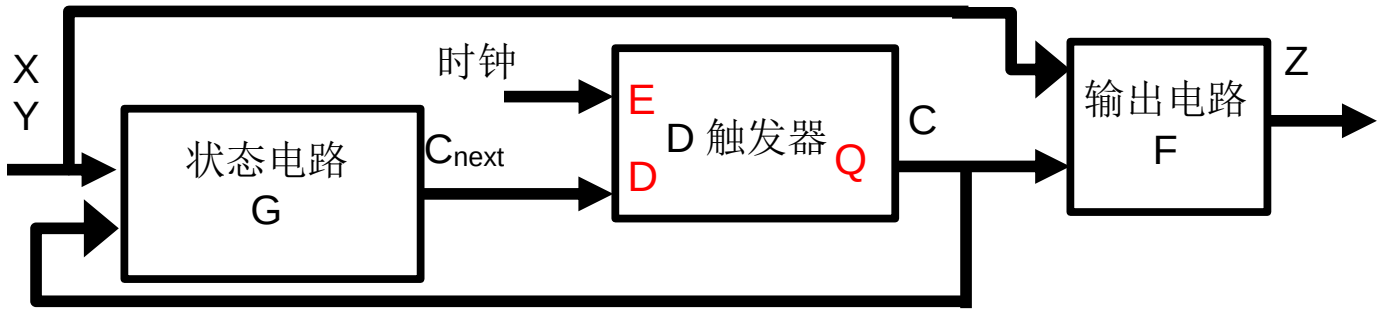
- 在四个时钟周期完成加法 $Z_3Z_2Z_1Z_0 = X_3X_2X_1X_0 + Y_3Y_2Y_1Y_0$
  - 每一周期完成一个全加操作
- 再用一个实例验证
  - $11_{10} + 9_{10} = 1011_2 + 1001_2 = 10100_2 = 20_{10} = 4_{10}$ and overflow
    输入X=1011，Y=1001；则输出Z=0100且产生进位溢出
- 设计过程
  - 第1步：首先确定状态电路，需要多少个D触发器？
  - 只需要一个 D触发器，其状态Q表示当前进位



设计4位串行加法器

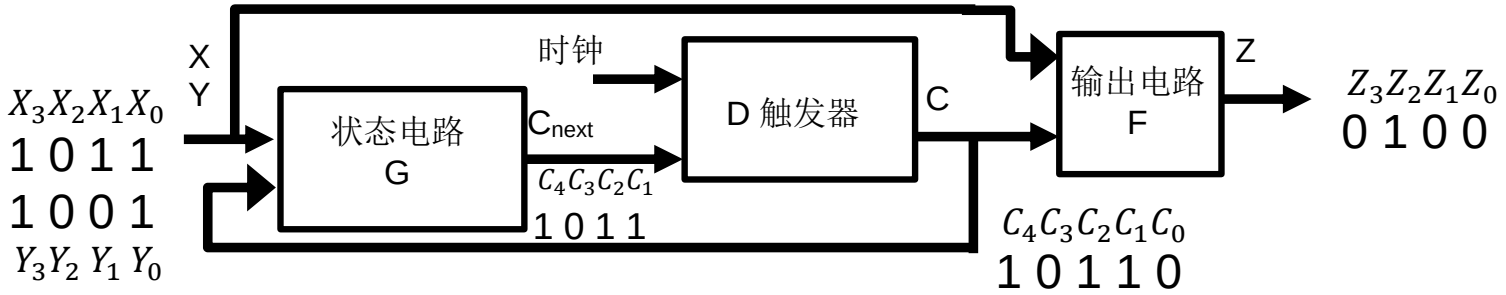
- 设计过程
  - 第1步：首先确定状态电路，需要多少个D触发器？
  - 只需要一个 D触发器，其状态Q表示当前进位C

  - 第2步：套用时序电路一般结构
  - In是输入X，Y；Out是输出Z；Q是进位C；Enable=时钟信号CLK

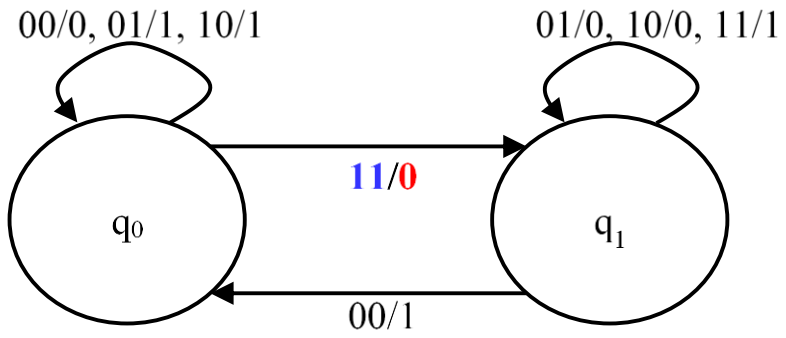


设计过程

- **第3步：求输出电路F与状态电路G的布尔表达式**
  - 画出时序电路的状态转换图



XY/Z



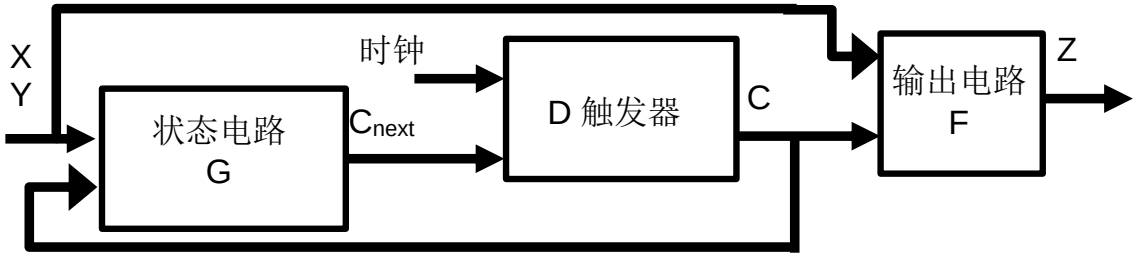
画出时序电路的状态转换图

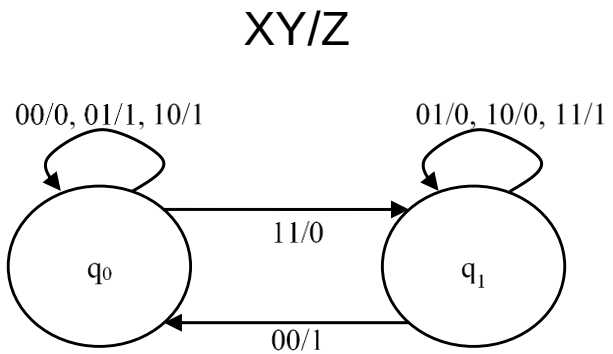
q_0 denotes C=0
q_1 denotes C=1

	1 0 1 **1** = X
+	1 0 0 **1** = Y
	1 0 1 **1** **0** = C
	0 1 0 **0** = Z

设计过程

- **第3步：求输出电路F与状态电路G的布尔表达式**
  - 画出时序电路的状态转换图；**进而导出真值表**





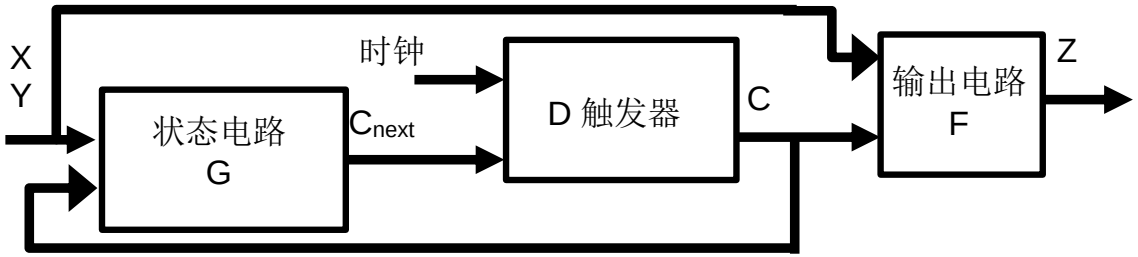
q_0 denotes C=0
q_1 denotes C=1

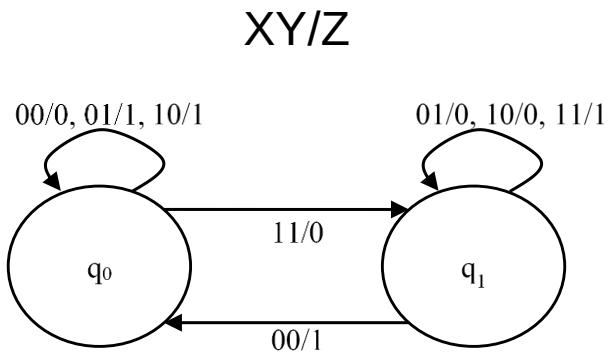
C	X	Y	Z	C_{next}
q_0	0	0	0	q_0
q_0	0	1	1	q_0
q_0	1	0	1	q_0
q_0	1	1	0	q_1
q_1	0	0	1	q_0
q_1	0	1	0	q_1
q_1	1	0	0	q_1
q_1	1	1	1	q_1

C	X	Y	Z	C_{next}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

设计过程

- **第3步：求输出电路F与状态电路G的布尔表达式**
  - 画出时序电路的状态转换图；进而导出真值表；**进而求出F和G的布尔表达式**





q_0 denotes C=0
q_1 denotes C=1

C	X	Y	Z	C_{next}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

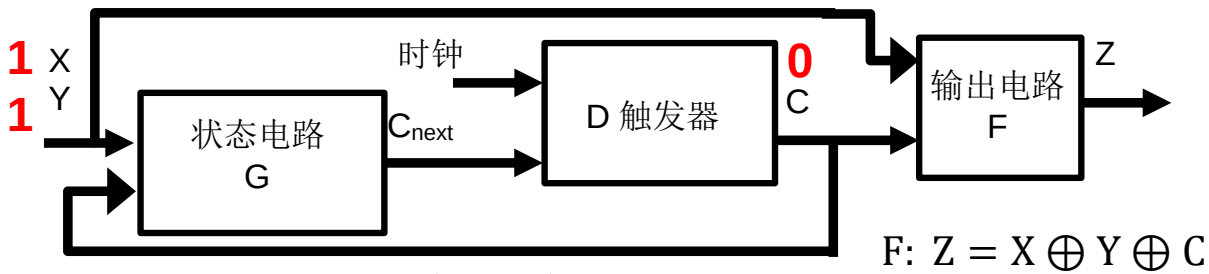
$$Z = F(X, Y, C) = X \oplus Y \oplus C$$

$$C_{next} = G(X, Y, C) = (X \cdot Y) + (X \oplus Y) \cdot C$$

设计过程第4步：验算

- 给定
  - （1）设计好的下图时序电路，
  - （2）输入$X_3X_2X_1X_0 = 1011$, $Y_3Y_2Y_1Y_0 = 1001$,与初始进位$C_0 = 0$
  - 正确结果应为$Z_3Z_2Z_1Z_0 = 0100$，且有$C_4 = 1$表示溢出
- 验算步骤
  - 时钟周期0：$Z_0 = X_0 \oplus Y_0 \oplus C_0 = \mathbf{1} \oplus \mathbf{1} \oplus \mathbf{0}$;
  - $C_1 = (X_0 \cdot Y_0) + (X_0 \oplus Y_0) \cdot C_0 = (\mathbf{1} \cdot \mathbf{1}) + (\mathbf{1} \oplus \mathbf{1}) \cdot \mathbf{0}$

$X_3X_2X_1X_0$
1 0 1 **1**
1 0 0 **1**
$Y_3Y_2\ Y_1\ Y_0$



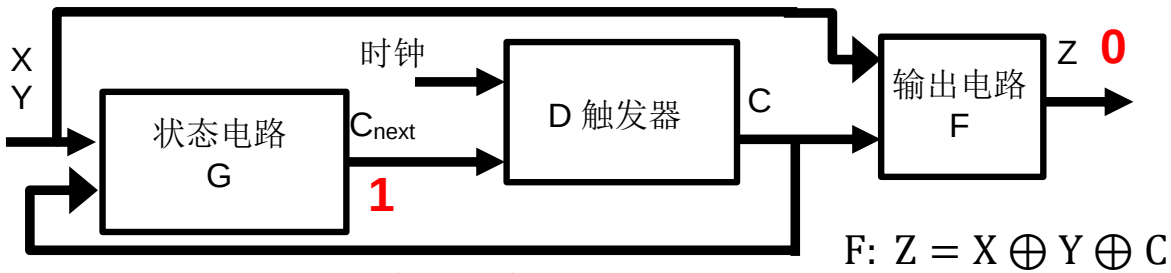
$Z_3Z_2Z_1Z_0$

$C_4C_3C_2C_1C_0$
0

设计过程第4步：验算

- 给定
  - （1）设计好的下图时序电路，
  - （2）输入$X_3X_2X_1X_0 = 1011$, $Y_3Y_2Y_1Y_0 = 1001$,与初始进位$C_0 = 0$
  - 正确结果应为$Z_3Z_2Z_1Z_0 = 0100$，且有$C_4 = 1$表示溢出
- 验算步骤
  - 时钟周期0：$Z_0 = X_0 \oplus Y_0 \oplus C_0 = 1 \oplus 1 \oplus 0 = \mathbf{0}$;
  - $C_1 = (X_0 \cdot Y_0) + (X_0 \oplus Y_0) \cdot C_0 = (1 \cdot 1) + (1 \oplus 1) \cdot 0 = \mathbf{1}$

$X_3X_2X_1X_0$
1 0 1 **1**
1 0 0 **1**
$Y_3Y_2\ Y_1\ Y_0$



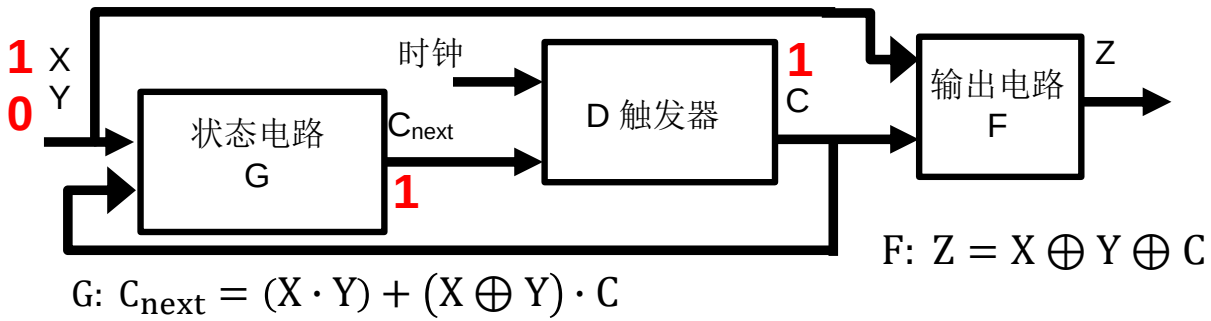
$Z_3Z_2Z_1Z_0$
0

$C_4C_3C_2C_1C_0$
1 0

设计过程第4步：验算

- 给定
  - （1）设计好的下图时序电路，
  - （2）输入$X_3X_2X_1X_0 = 1011$, $Y_3Y_2Y_1Y_0 = 1001$,与初始进位$C_0 = 0$
  - 正确结果应为$Z_3Z_2Z_1Z_0 = 0100$，且有$C_4 = 1$表示溢出
- 验算步骤
  - 时钟周期0：$Z_0 = X_0 \oplus Y_0 \oplus C_0 = 1 \oplus 1 \oplus 0 = 0$；$C_1 = (X_0 \cdot Y_0) + (X_0 \oplus Y_0) \cdot C_0 = (1 \cdot 1) + (1 \oplus 1) \cdot 0 = 1$
  - **时钟周期1**：$Z_1 = X_1 \oplus Y_1 \oplus C_1 = \mathbf{1} \oplus \mathbf{0} \oplus \mathbf{1}$；$C_2 = (X_1 \cdot Y_1) + (X_1 \oplus Y_1) \cdot C_1 = (\mathbf{1} \cdot \mathbf{0}) + (\mathbf{1} \oplus \mathbf{0}) \cdot \mathbf{1}$

$X_3X_2X_1X_0$
1 0 **1** 1
1 0 **0** 1
$Y_3Y_2\ Y_1\ Y_0$



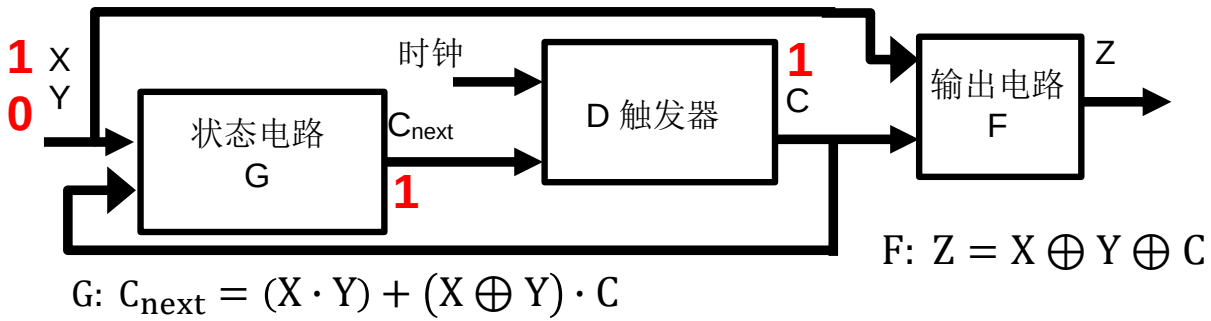
$Z_3Z_2Z_1Z_0$
0

$C_4C_3C_2C_1C_0$
1 0

设计过程第4步：验算

- 给定
  - （1）设计好的下图时序电路，
  - （2）输入$X_3X_2X_1X_0 = 1011$, $Y_3Y_2Y_1Y_0 = 1001$,与初始进位$C_0 = 0$
  - 正确结果应为$Z_3Z_2Z_1Z_0 = 0100$，且有$C_4 = 1$表示溢出
- 验算步骤
  - 时钟周期0：$Z_0 = X_0 \oplus Y_0 \oplus C_0 = 1 \oplus 1 \oplus 0 = 0$；$C_1 = (X_0 \cdot Y_0) + (X_0 \oplus Y_0) \cdot C_0 = (1 \cdot 1) + (1 \oplus 1) \cdot 0 = 1$
  - **时钟周期1**：$Z_1 = X_1 \oplus Y_1 \oplus C_1 = 1 \oplus 0 \oplus 1 = \mathbf{0}$；$C_2 = (X_1 \cdot Y_1) + (X_1 \oplus Y_1) \cdot C_1 = (1 \cdot 0) + (1 \oplus 0) \cdot 1 = \mathbf{1}$

$X_3X_2X_1X_0$
1 0 **1** 1
1 0 **0** 1
$Y_3Y_2\ Y_1\ Y_0$



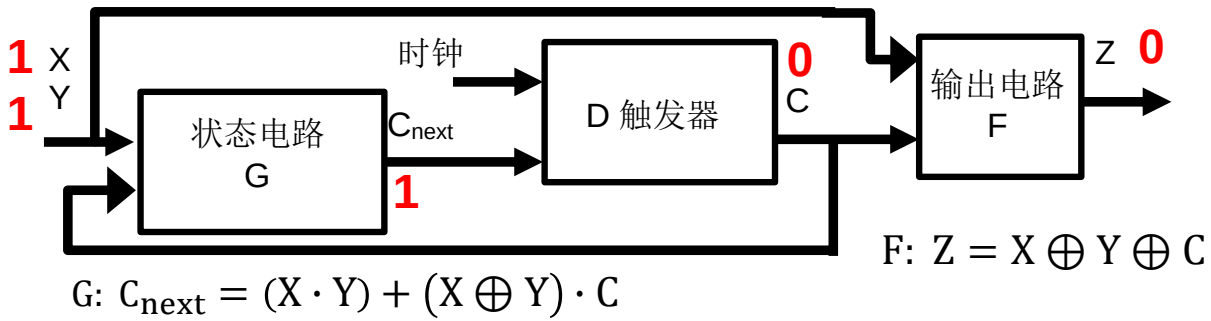
$Z_3Z_2Z_1Z_0$
0 0

$C_4C_3C_2C_1C_0$
1 1 0

设计过程第4步：验算

- 给定
  - （1）设计好的下图时序电路，
  - （2）输入$X_3X_2X_1X_0 = 1011$, $Y_3Y_2Y_1Y_0 = 1001$,与初始进位$C_0 = 0$
  - 正确结果应为$Z_3Z_2Z_1Z_0 = 0100$，且有$C_4 = 1$表示溢出
- 验算步骤
  - 时钟周期0：$Z_0 = X_0 \oplus Y_0 \oplus C_0 = 1 \oplus 1 \oplus 0 = 0$；$C_1 = (X_0 \cdot Y_0) + (X_0 \oplus Y_0) \cdot C_0 = (1 \cdot 1) + (1 \oplus 1) \cdot 0 = 1$
  - 时钟周期1：$Z_1 = X_1 \oplus Y_1 \oplus C_1 = 1 \oplus 0 \oplus 1 = 0$；$C_2 = (X_1 \cdot Y_1) + (X_1 \oplus Y_1) \cdot C_1 = (1 \cdot 0) + (1 \oplus 0) \cdot 1 = 1$
  - 时钟周期2：$Z_2 = X_2 \oplus Y_2 \oplus C_2 = 0 \oplus 0 \oplus 1 = 1$；$C_3 = (X_2 \cdot Y_2) + (X_2 \oplus Y_2) \cdot C_2 = (0 \cdot 0) + (0 \oplus 0) \cdot 1 = 0$
  - 时钟周期3：$Z_3 = X_3 \oplus Y_3 \oplus C_3 = 1 \oplus 1 \oplus 0 = 0$；$C_4 = (X_3 \cdot Y_3) + (X_3 \oplus Y_3) \cdot C_3 = (1 \cdot 1) + (1 \oplus 1) \cdot 0 = 1$

$X_3X_2X_1X_0$
1 0 1 1
1 0 0 1
$Y_3Y_2\ Y_1\ Y_0$



$Z_3Z_2Z_1Z_0$
0 1 0 0

$C_4C_3C_2C_1C_0$
1 0 1 1 0

设计过程第4步：验算

- 给定
  - （1）设计好的下图时序电路，
  - （2）输入$X_3X_2X_1X_0 = 1011$, $Y_3Y_2Y_1Y_0 = 1001$,与初始进位$C_0 = 0$
  - 正确结果应为$Z_3Z_2Z_1Z_0 = 0100$，且有$C_4 = 1$表示溢出
- 验算步骤
  - 时钟周期0：$Z_0 = X_0 \oplus Y_0 \oplus C_0 = 1 \oplus 1 \oplus 0 = \mathbf{0}$；$C_1 = (X_0 \cdot Y_0) + (X_0 \oplus Y_0) \cdot C_0 = (1 \cdot 1) + (1 \oplus 1) \cdot 0 = 1$
  - 时钟周期1：$Z_1 = X_1 \oplus Y_1 \oplus C_1 = 1 \oplus 0 \oplus 1 = \mathbf{0}$；$C_2 = (X_1 \cdot Y_1) + (X_1 \oplus Y_1) \cdot C_1 = (1 \cdot 0) + (1 \oplus 0) \cdot 1 = 1$
  - 时钟周期2：$Z_2 = X_2 \oplus Y_2 \oplus C_2 = 0 \oplus 0 \oplus 1 = \mathbf{1}$；$C_3 = (X_2 \cdot Y_2) + (X_2 \oplus Y_2) \cdot C_2 = (0 \cdot 0) + (0 \oplus 0) \cdot 1 = 0$
  - 时钟周期3：$Z_3 = X_3 \oplus Y_3 \oplus C_3 = 1 \oplus 1 \oplus 0 = \mathbf{0}$；$C_4 = (X_3 \cdot Y_3) + (X_3 \oplus Y_3) \cdot C_3 = (1 \cdot 1) + (1 \oplus 1) \cdot 0 = \mathbf{1}$

举一反三：用时序电路实现4位串行减法器

- 在四个时钟周期完成减法 $Z_3Z_2Z_1Z_0 = X_3X_2X_1X_0 - Y_3Y_2Y_1Y_0$
  - 假设数据表示是二进制补码
  - 四个时钟周期后得到正确结果
- 再用一个实例验证
  - (-5) – 1 = -6

- 课上已经讨论过“加减器”

加减器

代表了设计方法4：使用多路复用器和控制信号（本例中的选择输入S）

- 采用多路复用器（multiplexer，MUX）的加减器
  - 控制信号S选择做加法(S=0)还是做减法(S=1)
  - 采用二进制补码，减法等价于加负数　　5-5 = 5+(-5)；　减5 等价于 加-5
    - X的负数（-X）刚好是X的二进制补码
- 假设X = Y = 5。即，$X_3X_2X_1X_0 = Y_3Y_2Y_1Y_0 = 0101$

X – Y　= 5 + (-5) = 5 + (5的补码)

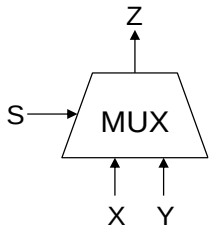
→ $0101 + \left(\overline{0}\,\overline{1}\,\overline{0}\,\overline{1} + 0001\right)$

= 0101 + 1011

= 10000 = $C_4\,Z_3\,Z_2\,Z_1\,Z_0$

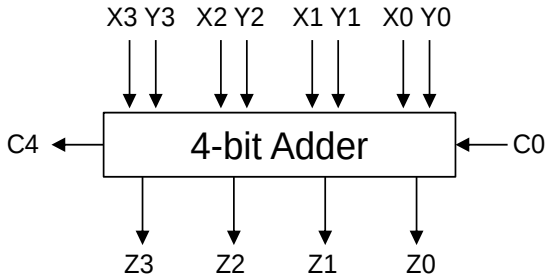
$$Z = \begin{cases} X & \text{if } S = 0 \\ Y & \text{if } S = 1 \end{cases}$$

S	X	Y	Z
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

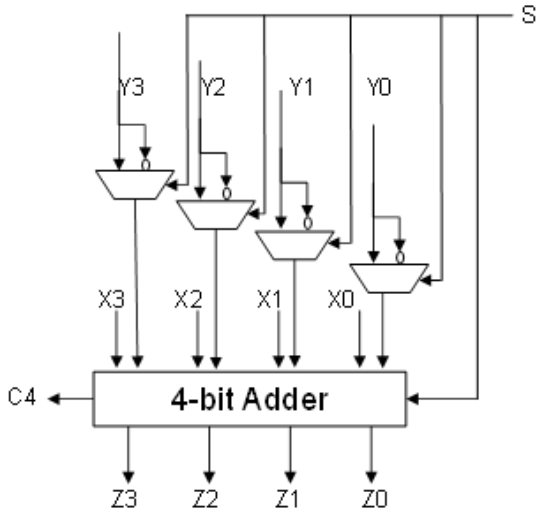


S	Z
0	X
1	Y

Multiplexer 多路复用器



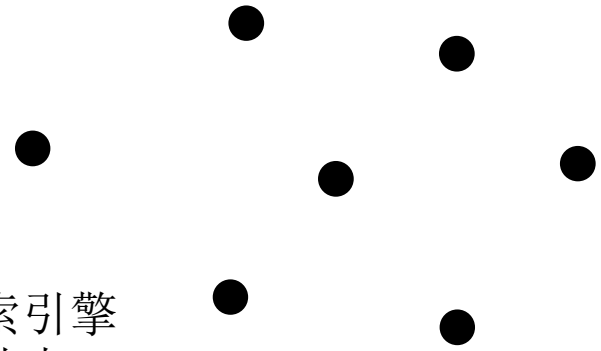
Two's complement 4-bit adder



Adder-subtractor

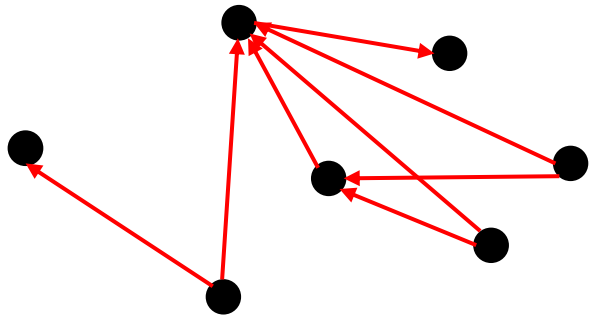
6. 网络拓扑：第二代搜索引擎示例

- 网络拓扑思想：一个网络是一个图，包含两个集合：节点集合与边集合
- 第一代搜索引擎只利用了网页网络的节点集合（网页）
  - Computed search results by matching the keywords in search queries to the contents of webpages (nodes)
- 第二代搜索引擎利用了网络拓扑，即网页网络的节点集合（网页）与边集合（超链接）
  - Web links (edges) also significantly influence the relevance of search results



第一代搜索引擎
只利用了节点

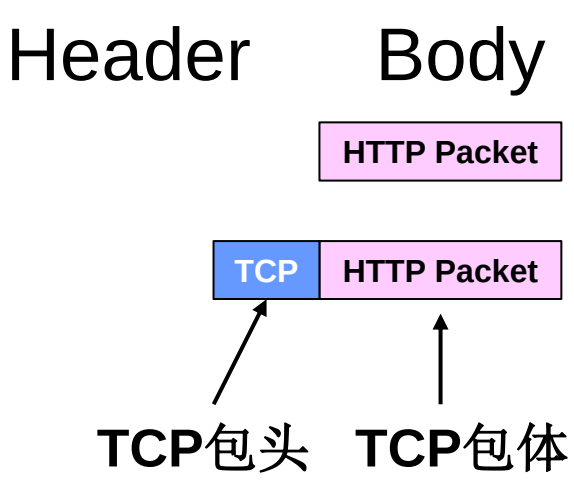
第二代搜索引擎
利用了节点和边

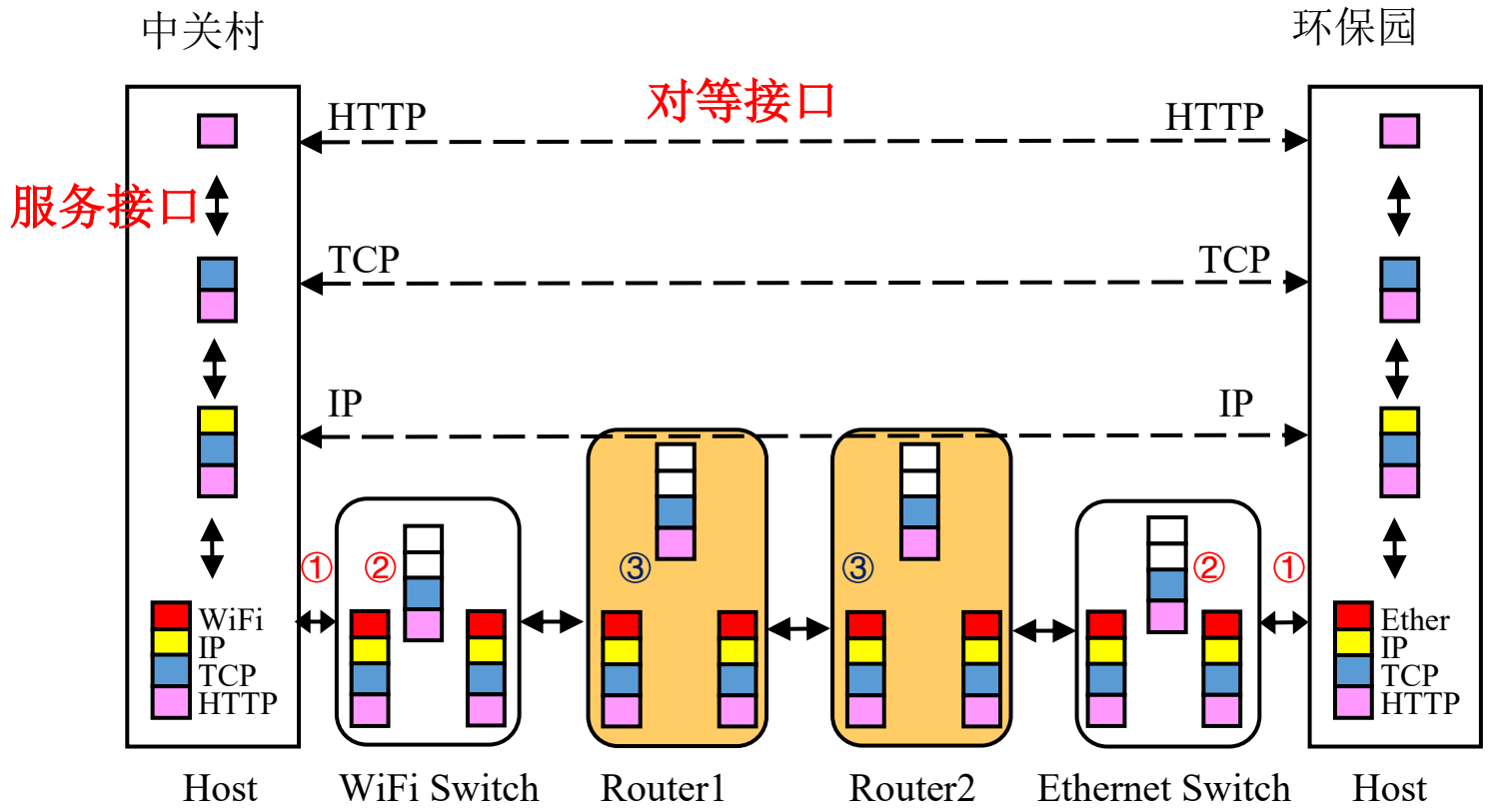


More fully utilizes network thinking and created Google and Baidu, serving billions of users and generating annual revenue over $100 billion

7. 协议栈：互联网通信示例

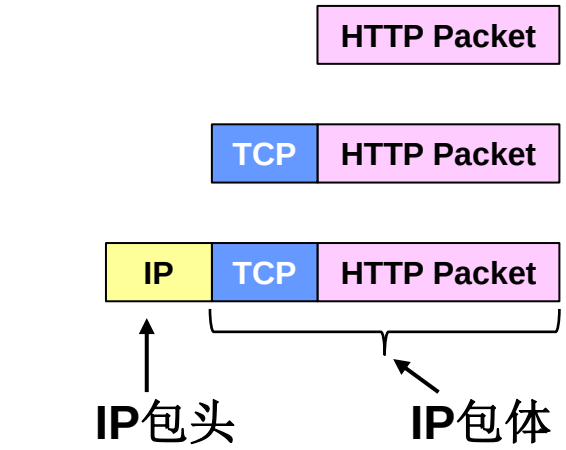
- 从中关村计算所的笔记本电脑通过WiFi访问位于环保园服务器的“猫猫指挥家”图片
- 笔记本电脑IP为192.168.1.101，服务器IP为10.208.104.3，图片URL为：
  http://seafile.solid.things.ac.cn:7019/kitty_band/*0555.JPG

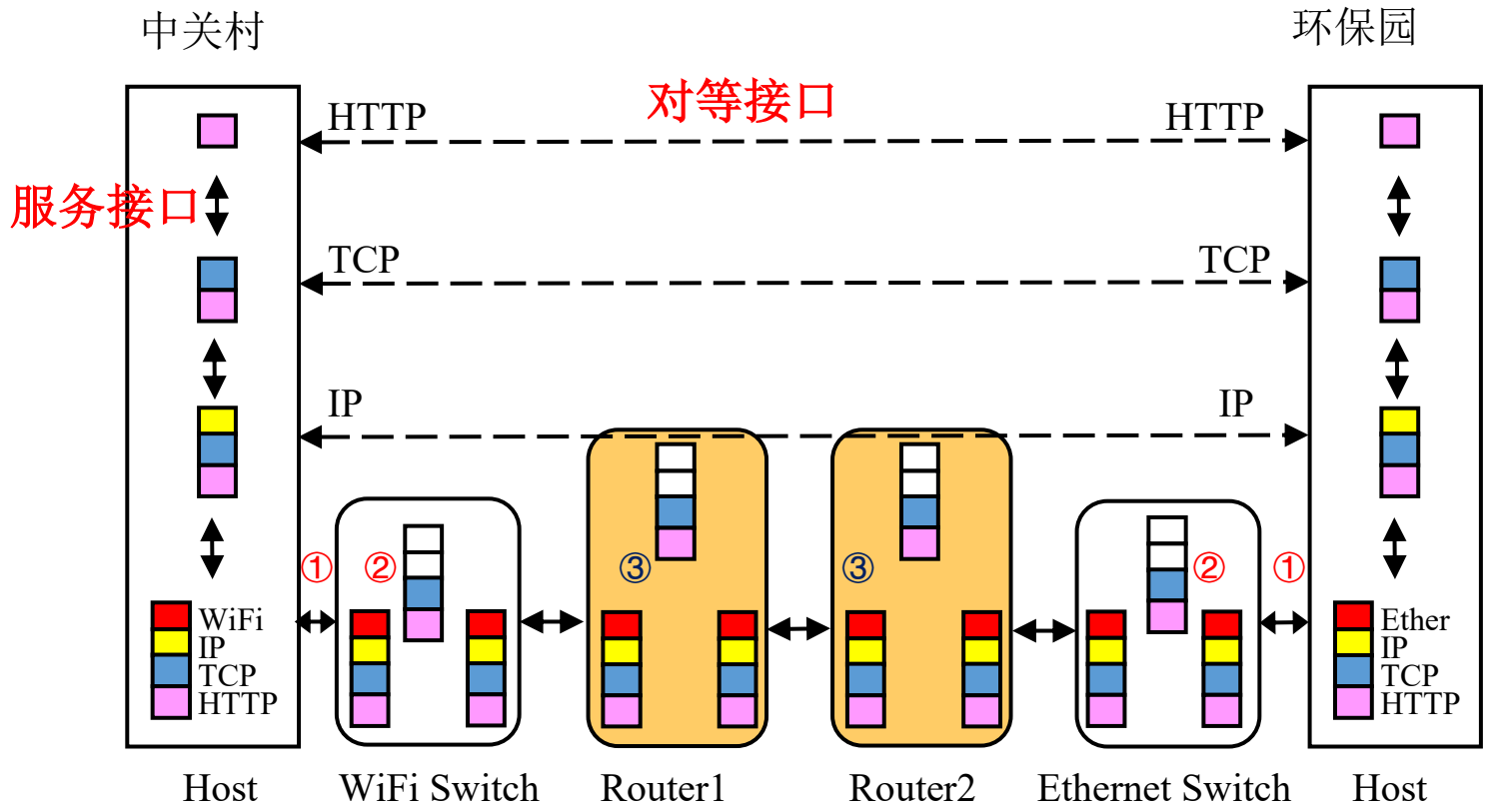




协议栈：互联网通信示例

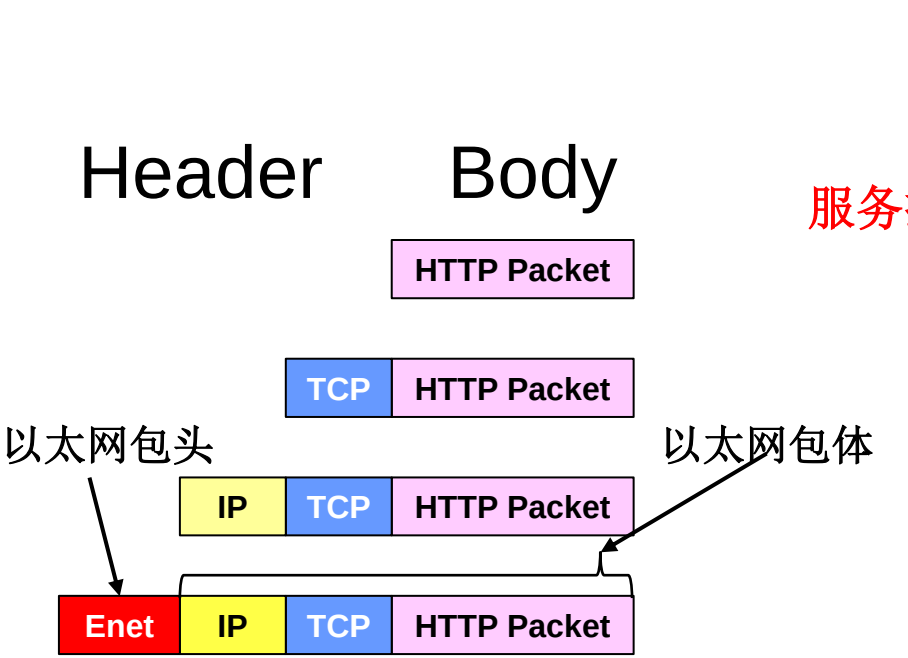
- 从中关村计算所的笔记本电脑通过WiFi访问位于环保园服务器的“猫猫指挥家”图片
- 笔记本电脑IP为192.168.1.101，服务器IP为10.208.104.3，图片URL为：
  http://seafile.solid.things.ac.cn:7019/kitty_band/*0555.JPG

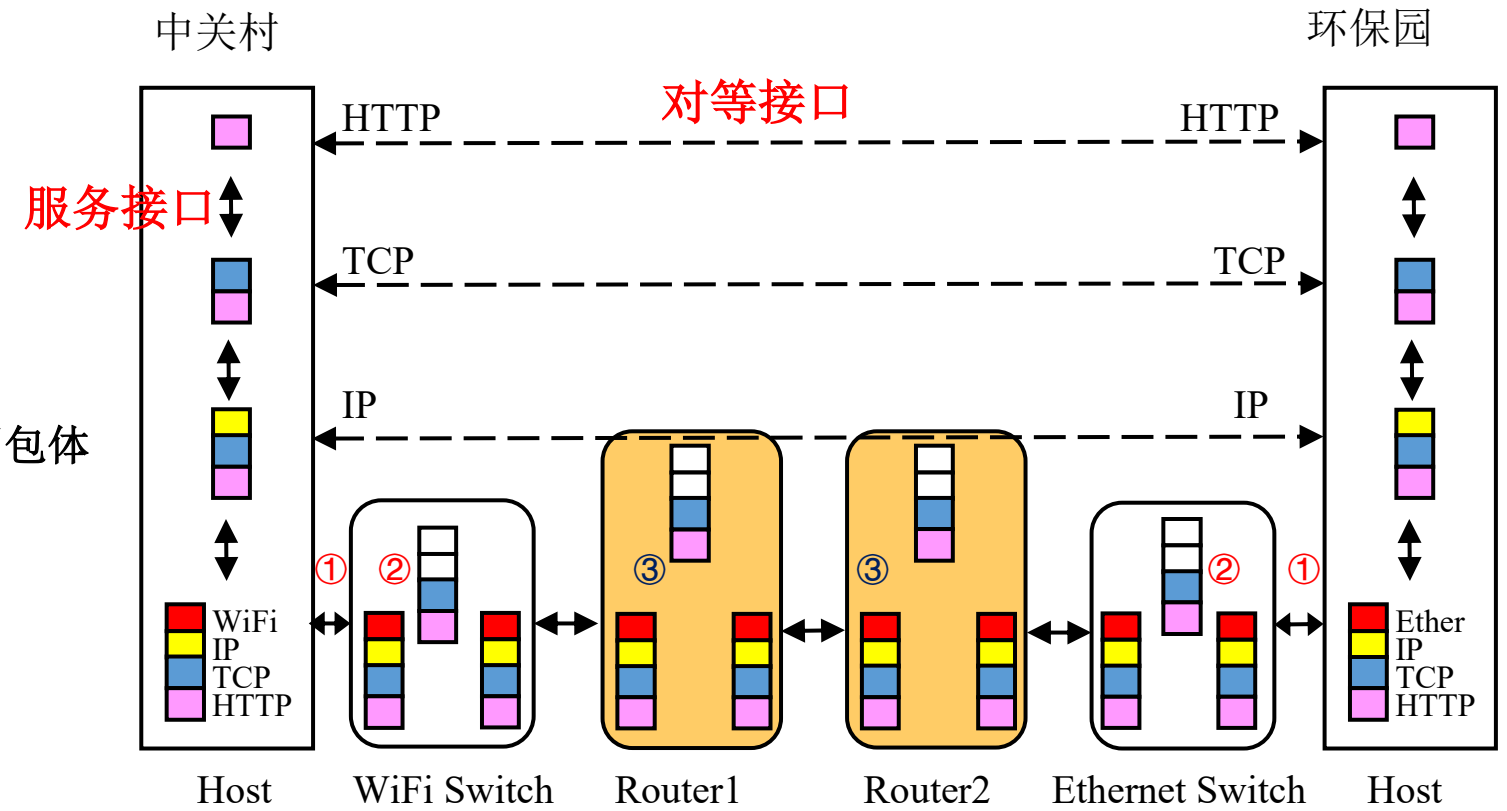




协议栈：互联网通信示例

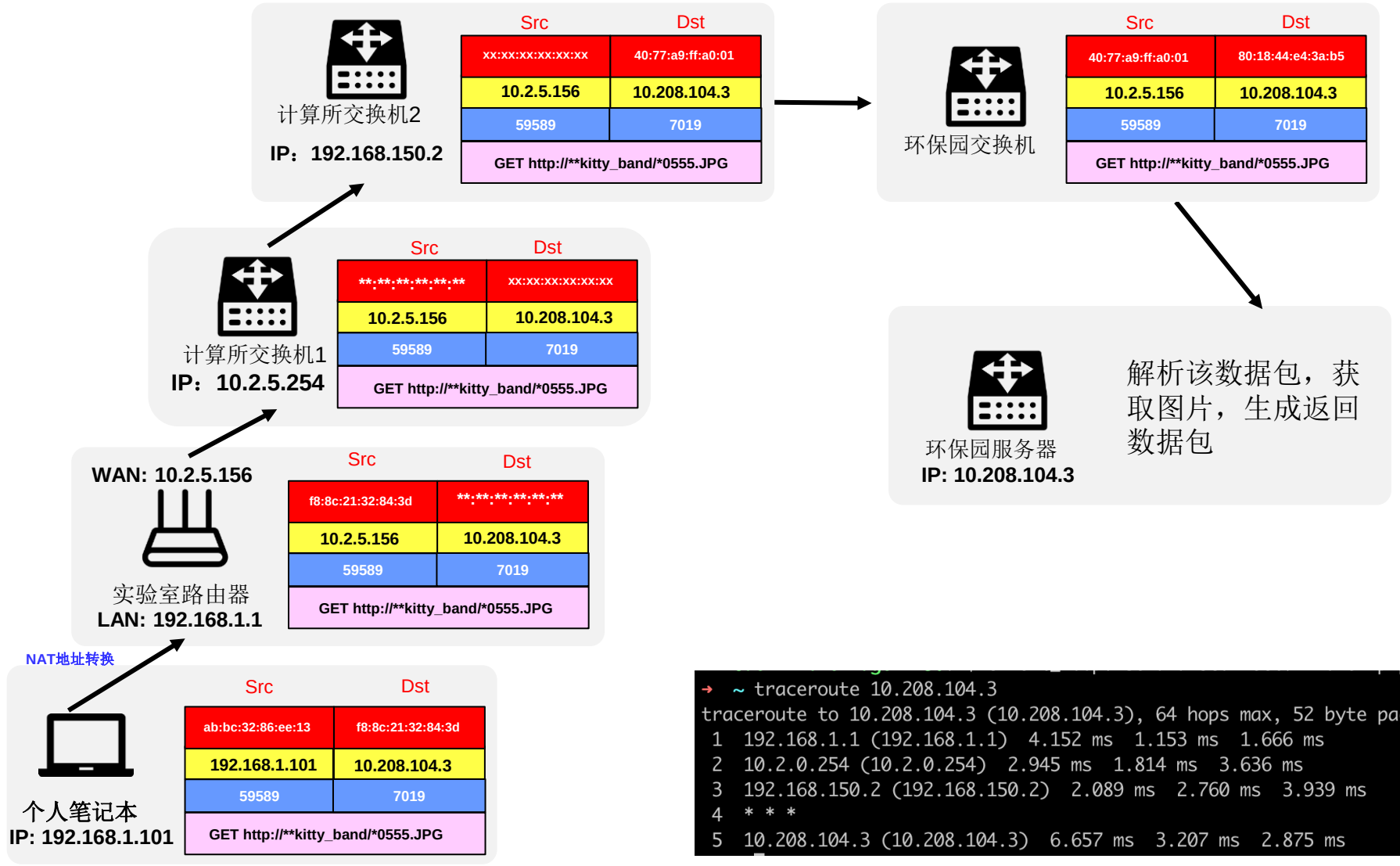
- 从中关村计算所的笔记本电脑通过WiFi访问位于环保园服务器的“猫猫指挥家”图片
- 笔记本电脑IP为192.168.1.101，服务器IP为10.208.104.3，图片URL为：
  http://seafile.solid.things.ac.cn:7019/kitty_band/*0555.JPG





From Client to Server

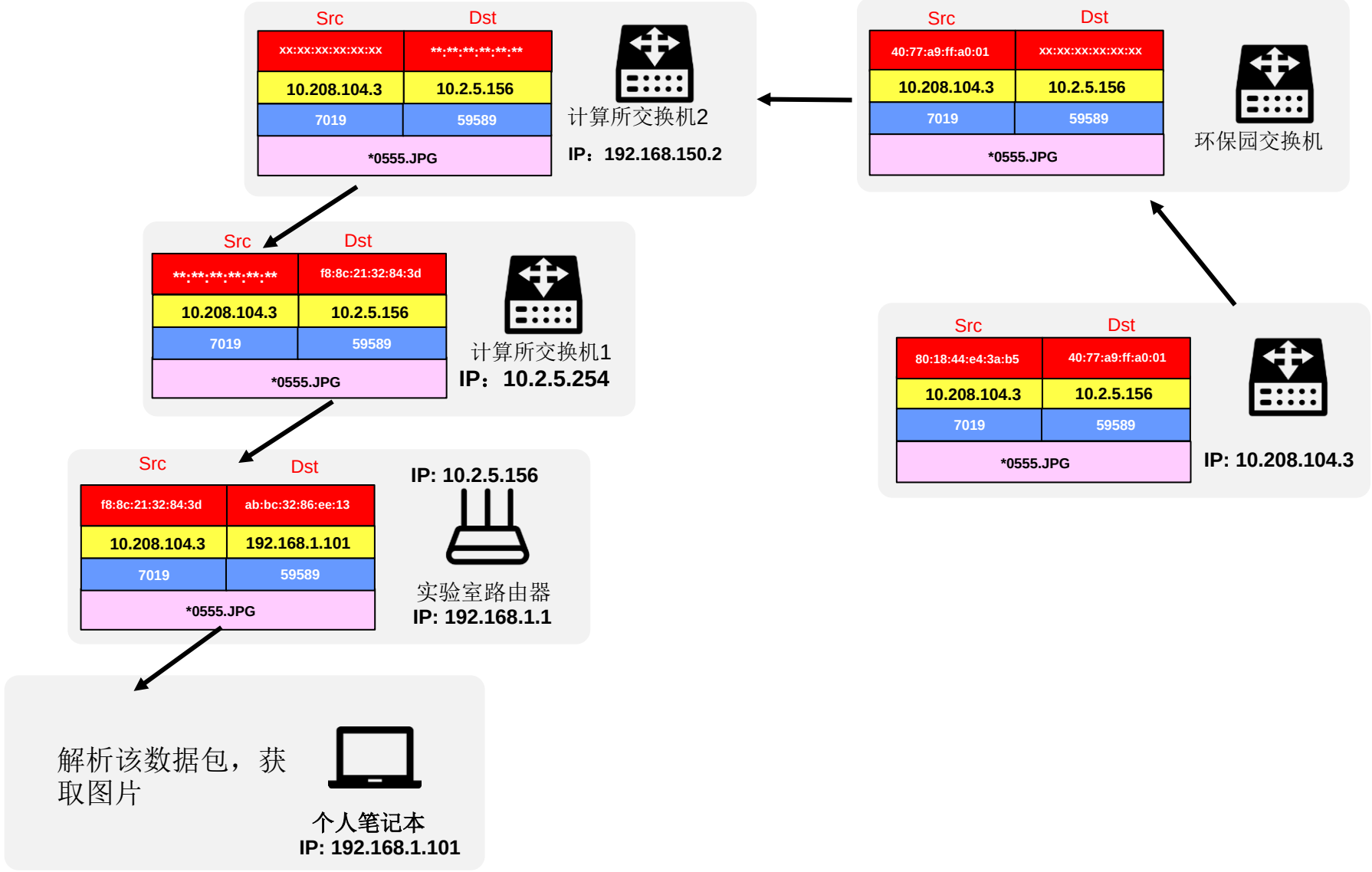
- 个人的笔记本电脑（**源IP地址：192.168.1.101**）通过WiFi发出请求，经过实验室路由器、计算所交换机和环保园交换机到达环保园服务器（**目标IP地址: 10.208.104.3**）
  - 使用traceroute软件追踪从计算所到环保园全路径




```
➜  ~ traceroute 10.208.104.3
traceroute to 10.208.104.3 (10.208.104.3), 64 hops max, 52 byte packets
 1  192.168.1.1 (192.168.1.1)  4.152 ms  1.153 ms  1.666 ms
 2  10.2.0.254 (10.2.0.254)  2.945 ms  1.814 ms  3.636 ms
 3  192.168.150.2 (192.168.150.2)  2.089 ms  2.760 ms  3.939 ms
 4  * * *
 5  10.208.104.3 (10.208.104.3)  6.657 ms  3.207 ms  2.875 ms
```

From Server to Client

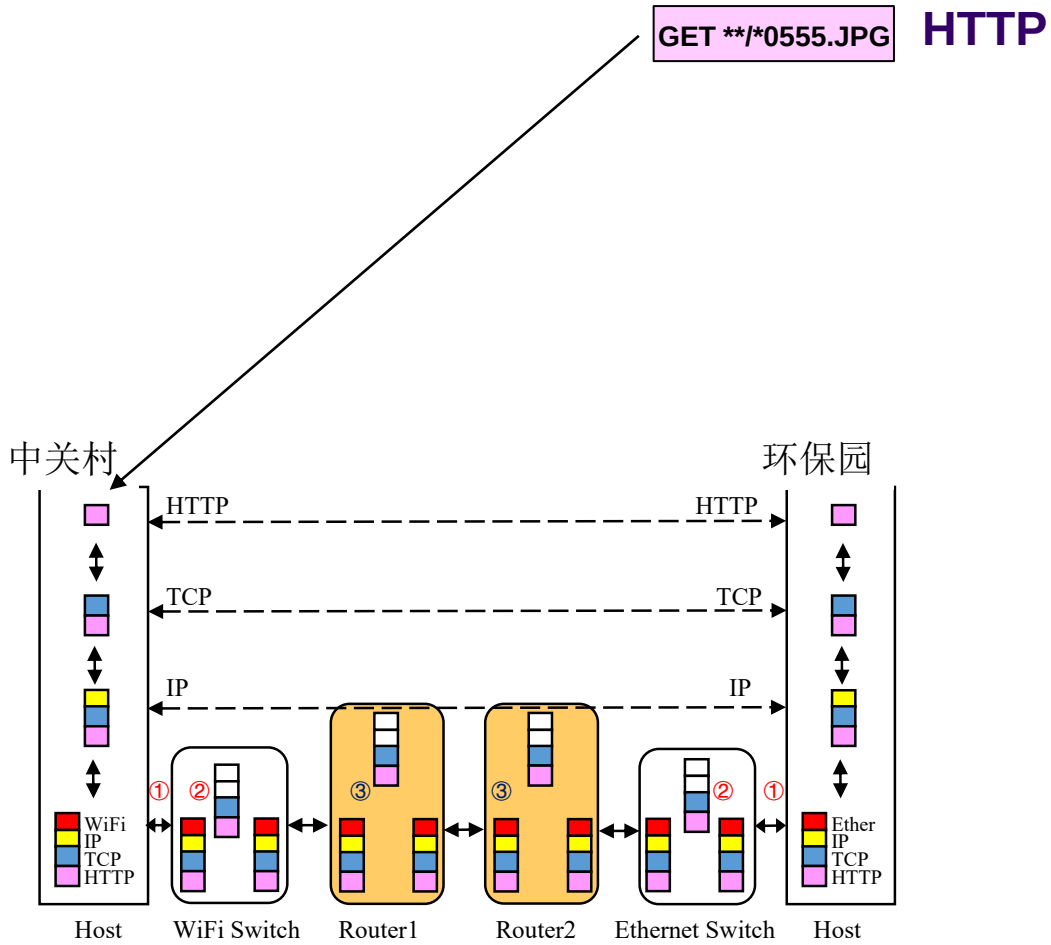
- 环保园的服务器收到请求后，获取图片，生成响应消息的数据包，并返回给个人笔记本
- 使用traceroute软件追踪从计算所到环保园全路径



请求消息

- 从中关村计算所的笔记本电脑通过WiFi访问位于环保园服务器的“猫猫指挥家”图片
- 笔记本电脑IP为192.168.1.101，服务器IP为10.208.104.3，图片URL为：
  http://seafile.solid.things.ac.cn:7019/kitty_band/*0555.JPG

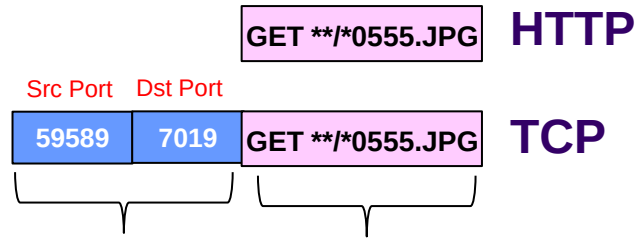
请求消息

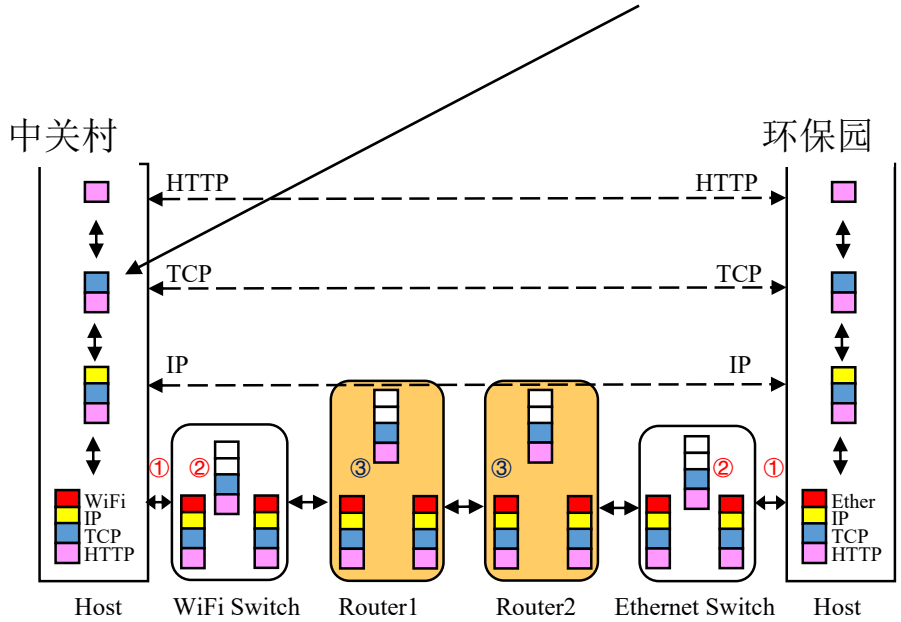


请求消息

- 从中关村计算所的笔记本电脑通过WiFi访问位于环保园服务器的“猫猫指挥家”图片；
- 笔记本电脑IP为192.168.1.101，服务器IP为10.208.104.3，图片地址为：
  http://seafile.solid.things.ac.cn:7019/kitty_band/*0555.JPG

- 包头包含三类信息：地址、查错、控制信息
- 本例忽略了TCP包头中的查错信息与控制信息



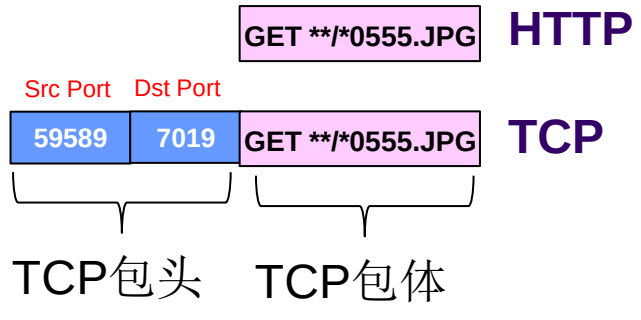


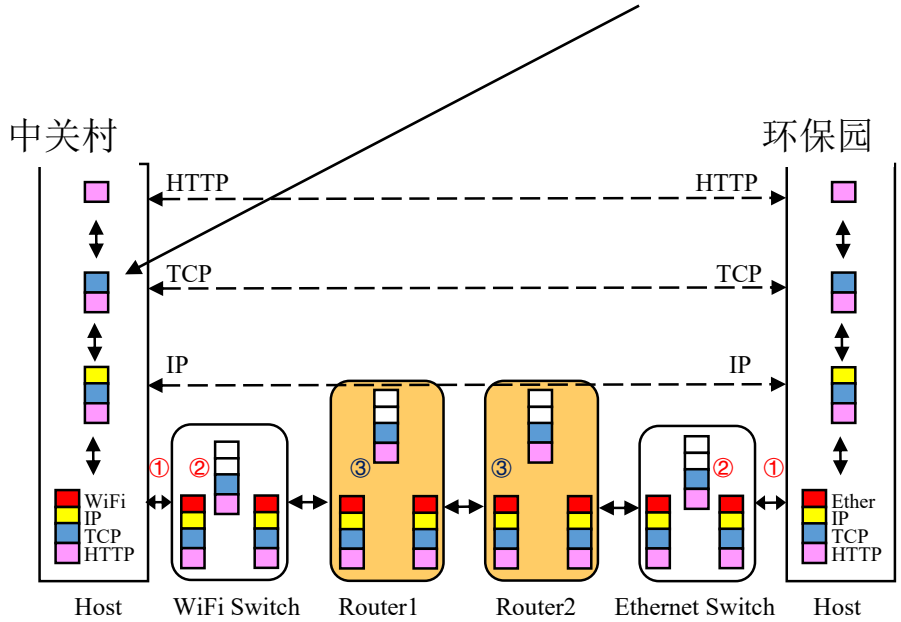
请求消息

- 从中关村计算所的笔记本电脑通过WiFi访问位于环保园服务器的“猫猫指挥家”图片；
- 笔记本电脑IP为192.168.1.101，服务器IP为10.208.104.3，图片地址为：
  http://seafile.solid.things.ac.cn:7019/kitty_band/*0555.JPG

- 包头包含三类信息：地址、查错、控制信息
- 本例忽略了TCP包头中的查错信息与控制信息

- Src: Source 源
- Dst: Destination 目的地、目标



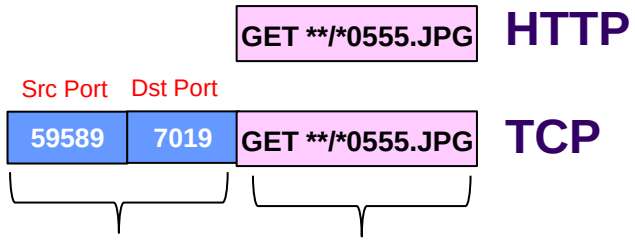


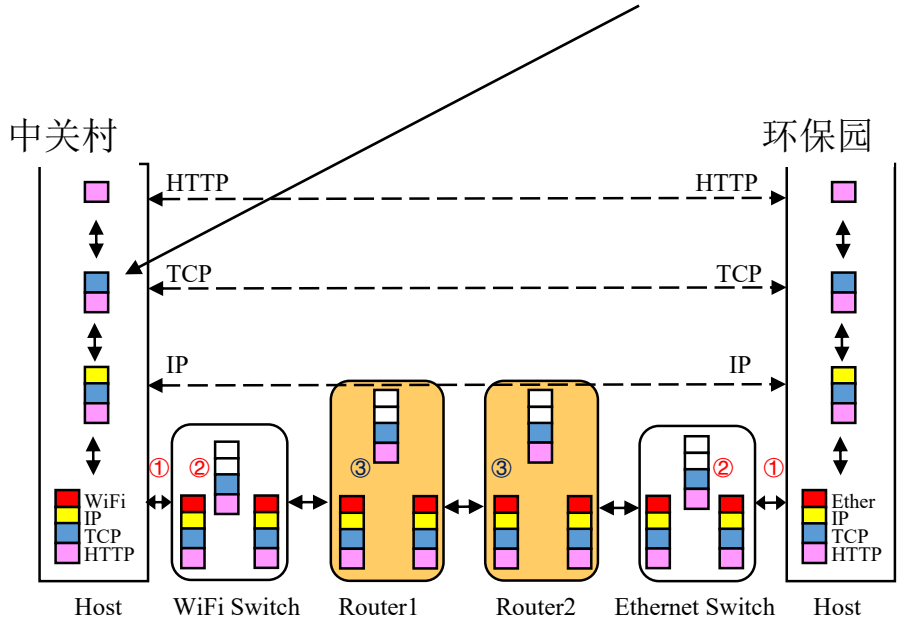
请求消息

- 从中关村计算所的笔记本电脑通过WiFi访问位于环保园服务器的“猫猫指挥家”图片；
- 笔记本电脑IP为192.168.1.101，服务器IP为10.208.104.3，图片地址为：
  http://seafile.solid.things.ac.cn:7019/kitty_band/*0555.JPG

- 包头包含三类信息：地址、查错、控制信息
- 本例忽略了TCP包头中的查错信息与控制信息

- 目的地端口7019指代环保园服务器计算机上的Web服务器进程
- 源端口号59589指代中关村笔记本电脑上的浏览器进程



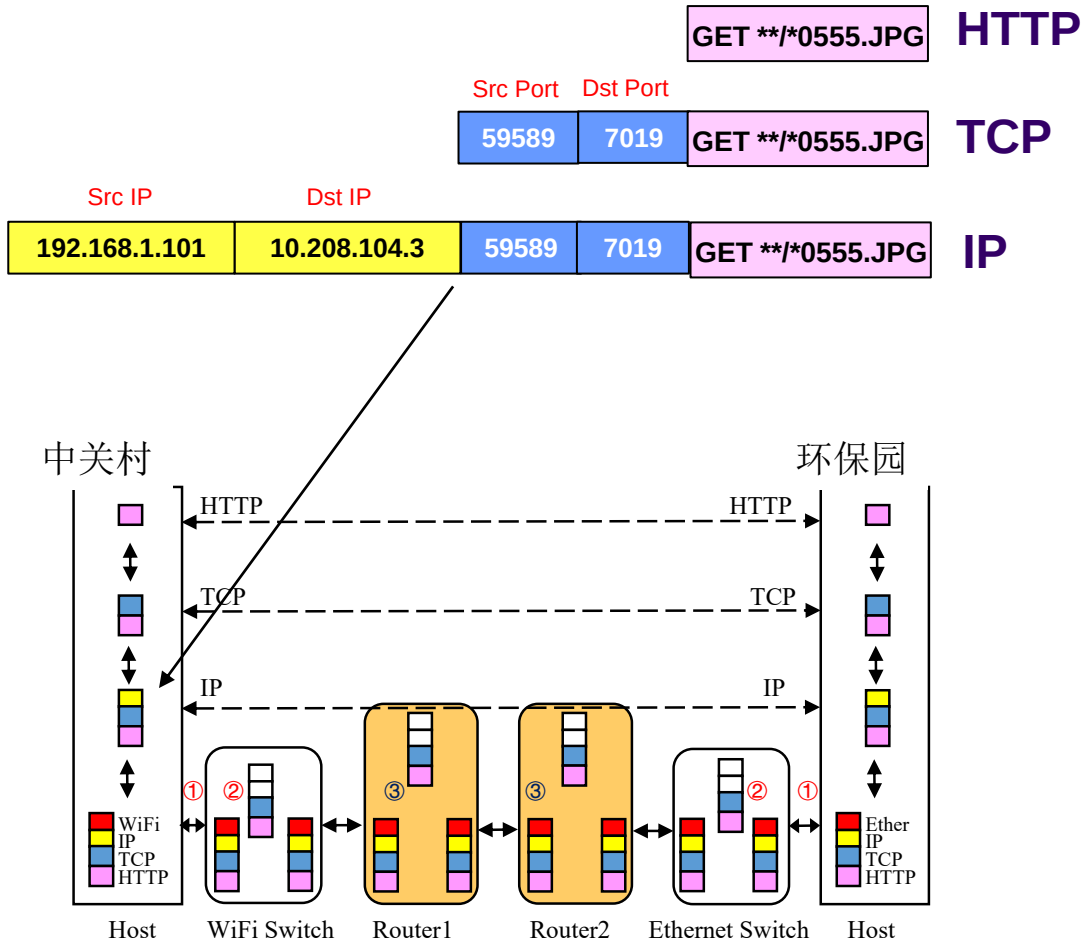


请求消息

- 从中关村计算所的笔记本电脑通过WiFi访问位于环保园服务器的“猫猫指挥家”图片；
- 笔记本电脑IP为192.168.1.101，服务器计算机IP为10.208.104.3，图片地址为：http://seafile.solid.things.ac.cn:7019/kitty_band/*0555.JPG

- 10.208.104.3是内网IP地址（***）

- 哪一段是IP包体？
- 哪一段是IP包头？

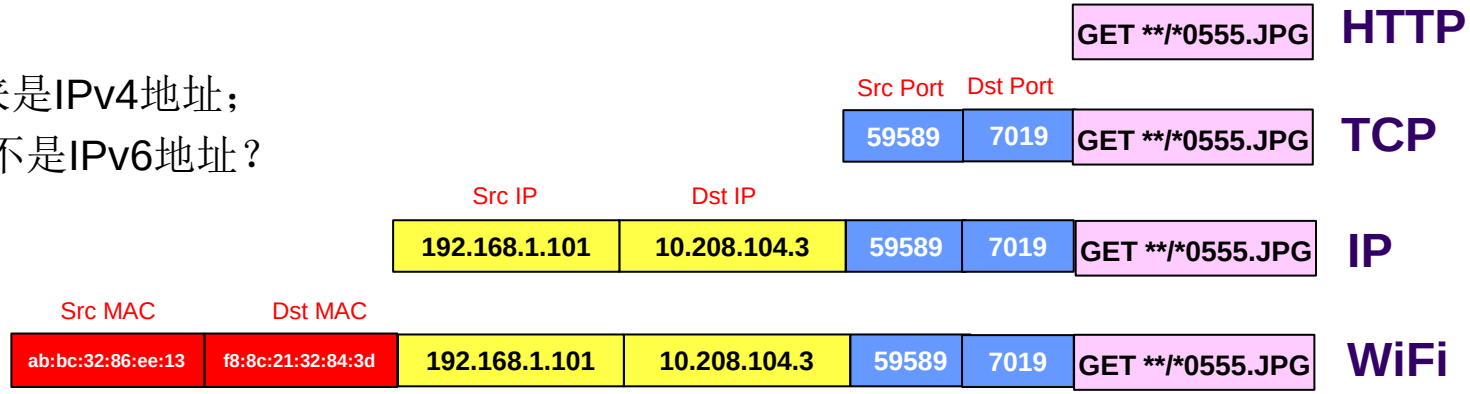


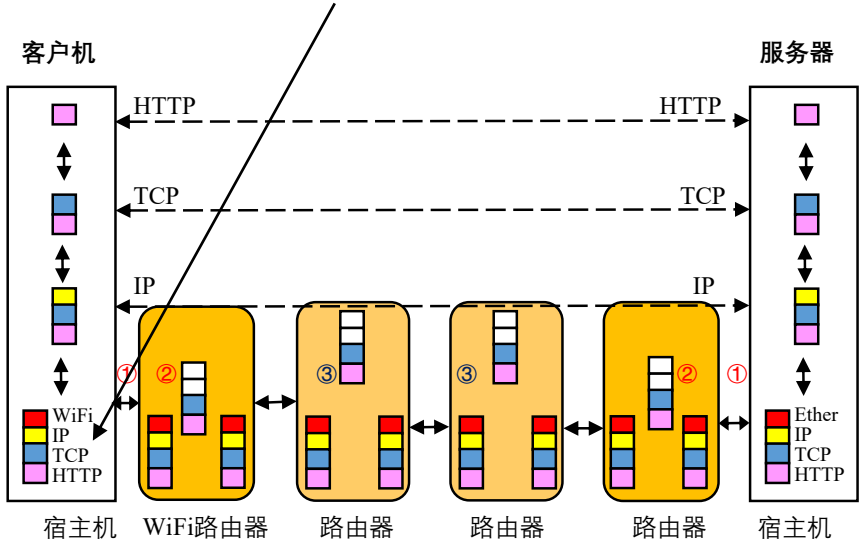
请求消息

- 从中关村计算所的笔记本电脑通过WiFi访问位于环保园服务器的“猫猫指挥家”图片；
- 笔记本电脑IP为192.168.1.101，服务器IP为10.208.104.3，图片地址为：
  http://seafile.solid.things.ac.cn:7019/kitty_band/*0555.JPG

- 数据链路层：那一段是包头，那一段是包体？

192.168.1.101看起来是IPv4地址；
f8:8c:21:32:84:3d是不是IPv6地址？





请求消息：四层的包格式与内容

- 从中关村计算所的笔记本电脑通过WiFi访问位于环保园服务器的“猫猫指挥家”图片；
- 笔记本电脑IP为192.168.1.101，服务器IP为10.208.104.3，图片地址为：
  http://seafile.solid.things.ac.cn:7019/kitty_band/*0555.JPG

该层的消息包
确定了哪些信息？