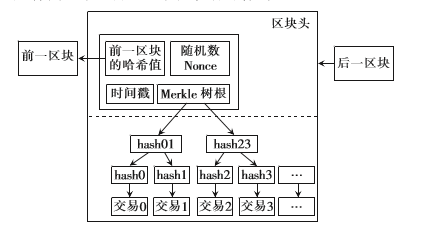
**哈希函数**(Hash Fuction)能够把任意长度的输入（又叫做函数前像）映射成固定长度的输出，即散列值。哈希函数是现代密码学的基本构件之一，最初被用于数字签名。为了实现对长文件的数字签名，先对消息进行哈希函数运算，然后对消息的哈希值进行签名而不是对原始消息进行签名。

哈希函数具有单向性、抗碰撞性等特点，还可用于承诺方案设计、流媒体认证、公平交易、电子拍卖等领域。在区块链中，哈希函数主要用于数据完整性、数据加密、共识计算的工作量证明、区块之间链接等。

区块链采用了双SHA256、RIPEMD160等哈希函数，SHA256主要用于加密交易形成区块，RIPEMD160则用于生成比特币的地址。

我们着重分析SHA256算法。区块链系统中的双SHA256函数是将不同长度的消息经过两次 SHA256计算处理，输出256位二进制字符串统一存储。因为哈希函数具有单向性，即对于给定的消息摘要h，找到消息M使得h=H(M)在计算上是不可行的。同时，哈希函数还具有抗碰撞性的特点，即找到两个不同的消息M和M’使得H(M)＝H(M’)在计算上是不可行的。这种特性被用来确保区块链中信息的完整性，在存储、传输的过程中，使得信息不会被篡改，以及篡改后能及时发现。

另外，**哈希函数在区块链的另外一个应用是作为区块间相连的哈希指针**，如图２所示。



**SHA256**是SHA-2下细分出的一种算法SHA-2，名称来自于安全散列算法2（英语：Secure Hash Algorithm 2）的缩写，一种密码散列函数算法标准，由美国国家安全局研发，属于SHA算法之一，是SHA-1的后继者。SHA-2下又可再分为六个不同的算法标准包括了：SHA-224、SHA-256、SHA-384、SHA-512、SHA-512/224、SHA-512/256。这些变体除了生成摘要的长度 、循环运行的次数等一些微小差异外，算法的基本结构是一致的。

**常量初始化**

8个质数（2,3,5,7,11,13,17,19）的平方根的小数部分取前32bit

前64个质数(2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97…)的立方根的小数部分取前32bit而来

**信息预处理**

**附加填充比特**

在报文末尾进行填充，使报文长度在对512取模以后的余数是448

填充是这样进行的：先补第一个比特为1，然后都补0，直到长度满足对512取模后余数是448。

例：以信息“abc”为例显示补位的过程。

a,b,c对应的ASCII码分别是97,98,99

于是原始信息的二进制编码为：01100001 01100010 01100011

补位第一步，首先补一个“1” ： 0110000101100010 01100011 1

补位第二步,补423个“0”：01100001 01100010 01100011 10000000 00000000 … 00000000

**为什么是448?**

因为在第一步的预处理后，第二步会再附加上一个64bit的数据，用来表示原始报文的长度信息。而448+64=512，正好拼成了一个完整的结构。

**附加长度值**

附加长度值就是将原始数据（第一步填充前的消息）的长度信息补到已经进行了填充操作的消息后面。

SHA256用一个64位的数据来表示原始消息的长度。

**逻辑运算**

按位“与”

按位“补”

按位“异或”

循环右移n个bit

右移n个bit

**Map(H{i-1}) = H{i}**

图中，ABCDEFGH这8个字（word）在按照一定的规则进行更新，其中深蓝色方块是事先定义好的非线性逻辑函数，上文已经做过铺垫红色田字方块代表 mod $ 2^{32} $ addition，即将两个数字加在一起，如果结果大于$ 2^{32} ，你必须除以，你必须除以，你必须除以 2^{32} $并找到余数。

ABCDEFGH一开始的初始值分别为$ H\_{i-1}(0),H\_{i-1}(1),…,H\_{i-1}(7) ，$Kt是第t个密钥，对应我们上文提到的64个常量，Wt是本区块产生第t个word。原消息被切成固定长度512-bit的区块，对每一个区块，产生64个word，通过重复运行循环n次对ABCDEFGH这八个字循环加密。

最后一次循环所产生的八个字合起来即是第i个块对应到的散列字符串$ H\_{i} $

**Merkle树**

在区块链中，Merkl树也被广泛应用。Merkle树类似于数据结构中的树形结构，主要为二叉树与 多叉树，在节点中保存信息的哈希值。在区块链的数据区块中数据结构主要为二叉 Merkle树，每个交易记录对应于一个哈希值，并对应于Merkle树的叶子节点，两个叶子节点再次两两配对哈希计算，通过递归的方式直到最后一个哈希值作为Merkle树根存入区块体。如图２所示，叶子节点hash0、hash1分别存储交易0、交易1的哈希值，非叶子节点hash01存储其子节点的哈希值，根节点通过计算其子节点的值得到。在区块链中，用户通过区块头得Merkle树根和别的节点所提供的中间哈希值来验证某个交易是否存在于区块中。另外，Merkle树支持简单支付验证协议SPV。中本聪在比特币的白皮书中指出，节点无需下载完整交易区块也可以完成确认。例如，在比特币系统中，节点只需要保存区块头信息，即可进行交易验证，即沿Merkle树的叶子节点上溯至Merkle根哈希处，然后根据根部哈希与计算交易记录所得的哈希值来快速确认交易的存在性和完整性。这将极大地降低比特币运行所需的带宽和验证时间，提高了共识的效率。

公钥密码学，又叫非对称密码学，是使用一对公钥和私钥的密码学，与只用一个密钥的对称密码学相对应。公钥密码学包括公钥加密算法和数字签名算法。

公钥基础设施PKI是提供公钥加密和数字签名服务的系统或平台，目的是为了管理密钥和证书。一个机构通过采用 PKI框架管理密钥和证书可以建立一个安全的网络环境。除了PKI框架下的公钥密码学，还有身份基公钥密码学，即用户身份作为其公钥，其私钥由PKG产生；无证书公钥密码学，是在传统的基于PKI的公钥密码体制和基于身份的公钥密码体制的基础上提出的一种新的公钥密码体制。无证书公钥密码体制不但解决了传统的基于PKI的公钥密码体制中庞杂的证书管理问题，而且解决了基于身份的公钥密码体制中的密钥托管问题；

基于证书公钥密码体制，该体制是Centry于2003年在欧密会提出的，克服了基于身份公钥密码体制和传统基于PKI公钥密码体制的缺陷。

区块链所用的公钥密码技术不同于上述４种，由于区块链完全无可信中心，不存在PKI、PKG等可信第三方。在区块链上，私钥由用户自己选取，并产生相应的公钥，该公钥对应记录在区块的地址上。一般而言，不同记录对应不同的公钥/私钥对。用户使用公钥对消息进行加密，只有对应的私钥才能解密。同时，私钥可用于对自己的交易信息进行数字签名，而别的用户可利用对应公钥对消息的签名进行验证。

数字签名:签名者对消息进行处理，生成别人无法伪造的一段数字串，这段数字串同时也是对消息的签名者发送消息真实性的一个有效证明。利用数字签名技术，能够确保消息传输的完整性、发送者的身份认证，防止交易中的抵赖发生。在比特币系统中，主要使用了 secp256k1椭圆曲线密码算法与Base58密码算法。比特币钱包的私钥用于证明比特币的拥有者。拥有者必须使用私钥给交易消息签名，以证明消息的发布者是相应比特币地址的所有 者。如果没有私钥，用户就不能给消息签名。比特币公钥用于生成用户的比特币地址，其生成过程如下：比特币的公钥是由上一步生成的私钥通过secp256k1椭圆曲线算法生成65字节的随机数，再经过SHA265、RIPEMD160算法生成20字节长度的消息摘要作为比特币地址的主体信息，再在前面加上版本号信息，在末尾添加4个字节的地址校验码，地址校验码通过对摘要结果进行两次SHA265运算得到，取哈希值的前四位产生，最后把版本信息、主体信息和地址校验码放在一块通过Base56转换为易识别的字符串作为地址。私钥在区块链系统中最为重要，一旦丢失，即代表这对应公钥地址的财产也一并丢失。通过利用数字签名，区块链系统交易保证了消息的完整性与不可抵赖性。

图的解释：

**对文件本身加密可能是个耗时过程，比如这封Email足够大，那么私钥加密整个文件以及拿到文件后的解密无疑是巨大的开销。  
数字签名可以解决这个问题：  
1.A先对这封Email执行哈希运算得到hash值简称“摘要”，取名h1  
2.然后用自己私钥对摘要加密，生成的东西叫“数字签名”  
3.把数字签名加在Email正文后面，一起发送给B  
（当然，为了防止邮件被窃听你可以用继续公钥加密，这个不属于数字签名范畴）  
4.B收到邮件后用A的公钥对数字签名解密，成功则代表Email确实来自A，失败说明有人冒充  
5.B对邮件正文执行哈希运算得到hash值，取名h2  
6.B 会对比第4步数字签名的hash值h1和自己运算得到的h2，一致则说明邮件未被篡改。**