红外遥控器的原理

一. 关于遥控器

遥控器其核心元器件就是编码芯片，将需要实现的操作指令例如选台、快进等事先编码，设备接收后解码再控制有关部件执行相应的动作。显然，接收电路及 CPU 也是与遥控器的编码一起配套设计的。编码是通过载波输出的，即所有的脉冲信号均调制在载波上，载波频率通常为 38K。载波是电信号去驱动红外发光二极管，将电信号变成光信号发射出去，这就是红外光，波长范围在 840nm 到 960nm 之间。在接收端，需要反过来通过光电二极管将红外线光信号转成电信号，经放大、整形、解调等步骤，最后还原成原来的脉冲编码信号，完成遥控指令的传递，这是一个十分复杂的过程。

红外线发射管通常的发射角度为 30-45 度之间，角度大距离就短，反之亦然。遥控器在光轴上的遥控距离可以大于 8.5 米，与光轴成 30 度（水平方向）或 15 度（垂直方向）上大于 6.5 米，在一些具体的应用中会充分考虑应用目标，在距离角度之间需要找到某种平衡。

对于遥控器涉及到如下几个主要问题：

1. 遥控器发出的编码信号驱动红外线发射管，必须发出波长范围在 940nm 左右的的红外光线，因为红外线接收器的接收二极管主要对这部分红外光信号敏感，如果波长范围不在此列，显然无法达到控制之目的。不过，几乎所有的红外家电遥控器都遵循这一标准。正因为有这一物理基础，多合一遥控器才有可能做成。

2. 遥控器发出一串编码信号只需要持续数十 ms 的时间，大多数是十多 ms 或一百多 ms 重复一次，一串编码也就包括十位左右到数十位二进制编码，换言之，每一位二进制编码的持续时间或者说位长不过 2ms 左右，频率只有 500kz 这个量级，要发射更远的距离必需通过载波，将这些信号调制到数十 khz，用得最多的是 38khz，大多数普通遥控器的载波频率是所用的陶瓷振荡器的振荡频率的 1/12，最常用的陶瓷振荡器是 455khz 规格，故最常用的载波也就是 455khz/12=37.9khz，简称 38k 载波。此外还有 480khz（40k）、440khz（37k）、432khz（36k）等规格，也有 200k 左右的载波，用于高速编码。红外线接收器是一体化的组件，为了更有针对性地接收所需要的编码，就设计成以载波为中心频率的带通滤波器，只容许指定载波的信号通过。显然这是多合一遥控器应该满足的第二个物理条件。不过，家用电器多用 38k，很多红外线接收器也能很好地接收频率相近的 40k 或 36k 的遥控编码。

3. 一个设备受控，除了满足上面提到的两个基本物理条件外，最重要的变化多种多样的当然应该是遥控器发出一串二进制编码信号了，这也是不同的遥控器不能相互通用的主要原因。由于市场上出现成百上千的编码方式并存，并没有一个统一的国际标准，只有各芯片厂商事实上的标准，这也是模拟并替换各种原厂遥控器最大的难点。随着技术的不断发展，很多公司开发家电设备的遥控子系统时还不采用通用的编码芯片，而是用通用的单片机随心所欲地自编一些编码，这就使通用遥控的问题更加复杂化了。

4. 采用同样的编码芯片，也不意味着可以通用，因为还有客户码。客户码设计的最初本意就是为了不同的设备可以相互区分互不干扰。最初芯片厂商会从全局考虑给不同的家电厂商安排不同的客户码以规范市场，例如录像机和电视机就用不同的设备码，给甲厂分配的设备码和乙厂分配的设备码就区分在不同的范围内。

5. 采用同样的编码芯片、同样的客户码下，也不能意味着一定可以通用，因为对命令码的分配与使用上，仍然是没有固定的模式可以遵循，遥控器编码芯片简单的支持数十种命令码，多的上千种，但遥控器往往只有数十个键，甚至只有几个键，如何从中选取这数十个键，这些键如何分配使用，不同的系统设计师都自搞一套，这样一来事情就更复杂化了。

设计需考虑的问题是如何“同化”不同遥控器发射信号之间的差异。遥控编码方式涉及很多方面，首先是数字 0 和 1 的表示（调宽还是调相，脉宽和占空比）；其次是帧结构（引导码和结束码，客户码和命令码长度及发送方式）；再次是帧间结构（仅发一次还是反复多次，多帧交替发送，帧间间隔变化）；最后是载波频率，以 38Khz 居多，也有 40Khz 甚至 200khz 等特殊载波。

设计相应电路和软件时对上述诸多因素加以分析、归纳，将编码特点用一串二进制位表示出来形成设备码，对应于一个具体的遥控器。同一个设备码下也就是同一个遥控器不同的按键则用命令码来表示。代码型遥控器用软件的方式对这些统一的编码进行解释，驱动一个个命令码按指定设备码格式加以“封装”，形成所需要的遥控信号，达到控制家电的目的。

二. 红外遥控器原理

一般的红外遥控系统是由红外遥控信号发射器、红外遥控信号接收器和微控制器及其外围电路等三部分构成的。遥控信号发射器用来产生遥控编码脉冲，驱动红外发射管输出红外遥控信号，遥控接收头完成对遥控信号的放大、检波、整形、解调出遥控编码脉冲。遥控编码脉冲是一组组串行二进制码，对于一般的红外遥控系统，此串行码输入到微控制器，由其内部 CPU 完成对遥控指令解码，并执行相应的遥控功能。

在红外遥控系统中，解码的核心是 CPU。它接收解调出的串行二进制码，在内部根据本系统的遥控信号编码格式将串行码对应成遥控器上的按键。显然，这种在 CPU 内部解码出的遥控指令是不便我们利用的，而且我们也不需要获取它。我们只需利用一般红外遥控系统中的遥控发射器、遥控接收头，自行设计解码电路直接对遥控接收头解调出的遥控编码脉冲进行解码，就可以得到原始的按键信息。

1. 红外遥控编码

目前应用中的各种红外遥控系统的原理都大同小异，区别只是在于各系统的信号编码格式不同。

红外遥控发射器组成了键扫描、编码、发射电路。当按下遥控器上任一按键时，TC9012 即产生一串脉冲编码。

遥控编码脉冲对 40kHz 载波进行脉冲幅度调制(PAM)后便形成遥控信号，经驱动电路由红外发射管发射出去。红外遥控接收头接收到调制后的遥控信号，经前置放大、限幅放大、带通滤波、峰值检波和波形整形，从而解调出与输入遥控信号反相的遥控脉冲。

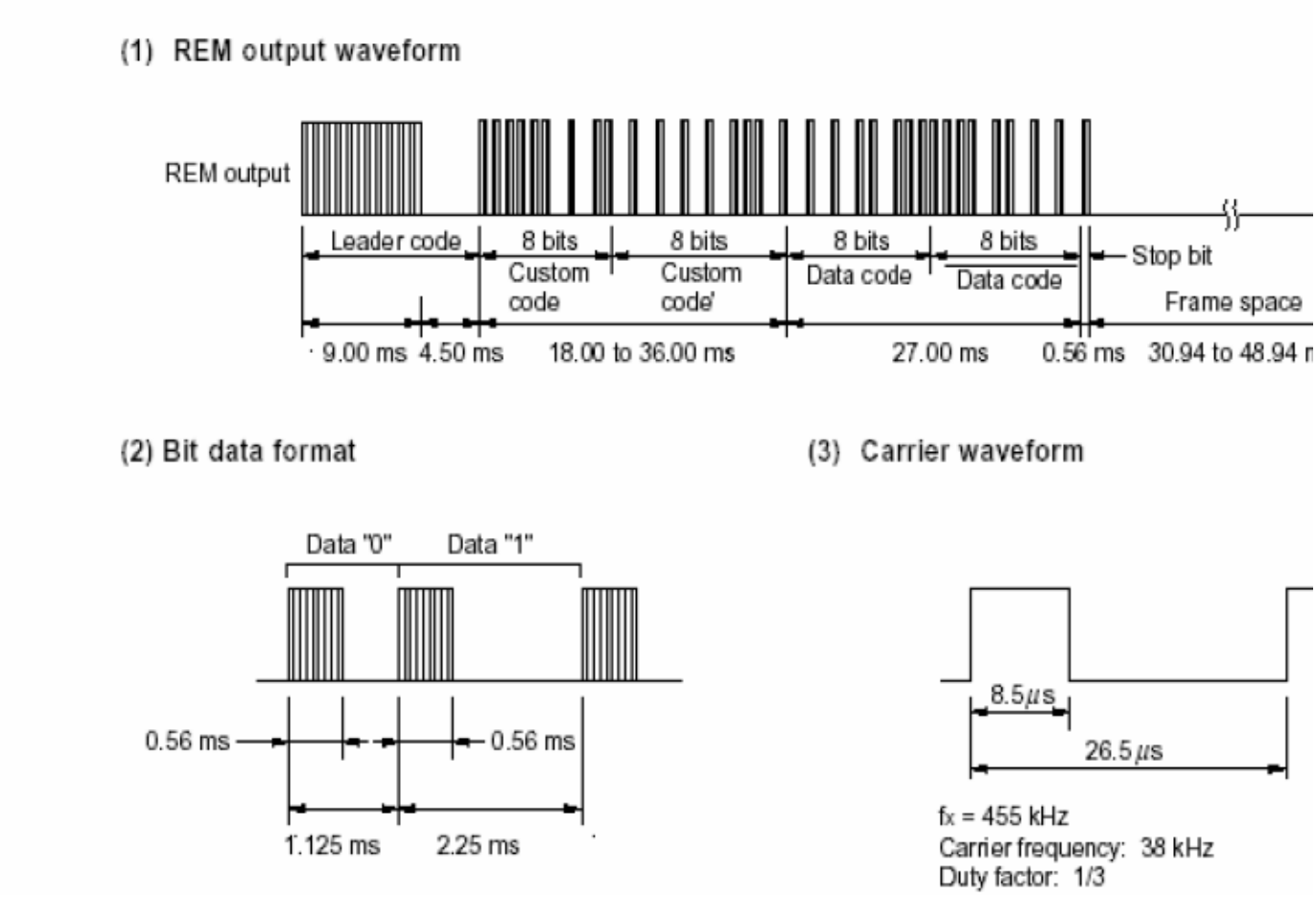
一次按键动作的遥控编码信息为 32 位串行二进制码。对于二进制信号“0”，一个脉冲占 1.2ms；对于二进制信号“1”，一个脉冲占 2.4ms，而每一脉冲内低电平均为 0.6ms。从起始标志到 32 位编码脉冲发完大约需 80ms，此后遥控信号维持高电平。若按键未释放，则从起始标志起每隔 108ms 发出 3 个脉冲的重复标志。

在 32 位的编码脉冲中，前 16 位码不随按键的不同而变化，我们称之为用户码。它是为了表示特定用户而设置的一个辨识标志，以区别不同机种和不同用户发射的遥控信号，防止

误操作。后 16 位码随着按键的不同而改变，我们就是要读取这 16 位按键编码，经解码得到按键键号，转而执行相应控制动作。

那么，不同的按键编码脉冲是怎样和遥控器上不同的按键一一对应的呢？我们借助于逻辑分析仪记录下来遥控器上每一个按键的编码脉冲序列，破译出了各按键的编码。截取 16 位键码的 8 位（比如后 8 位）就可达到识别按键的目的。当然，要加强遥控系统的抗干扰能力，还需接收全 16 位键码甚至 16 位用户码加以识别。

就 NEC 消費性遙控器格式來說:



2. 红外遥控解码

红外遥控接收头解调出的编码是串行二进制码，包含着遥控器按键信息。但它还不便于 CPU 读取识别，因此需要先对这些串行二进制码进行解码。下面所讲的红外遥控信号解码电路，它主要包括遥控编码脉冲串并转换电路与 PLD 解码电路。

遥控编码脉冲的串并转换如下所述:

红外遥控接收头解调出的遥控编码脉冲经一非门反相后引入计数器 4020 的复位端(RST)，4020 的脚 10(CP)端引入 1MHz 计数脉冲。遥控信号（已反相）中每一正脉冲到来时其高电平对 4020 复位，经过 0.6ms 遥控信号变为低电平，4020 复位结束，开始以 1MHz 的频率计数，直到下一个正脉冲到来时为止。二进制码“0”每一脉冲周期低电平时间为 0.6ms，二进制码“1”每一脉冲周期低电平时间为 1.8ms，4020 的 Q11 端即可以区分二进制码“0”或“1”。每一遥控编码正脉冲上升沿到来时，若 Q11 端为“1”，说明前一位遥控码为“1”；若 Q11 端为“0”，说明前一位遥控码为“0”。

将 4020 的 Q11 端作为 74HCS9S 的串行移位输入端(SER)，便可在每一个遥控编码脉冲上升沿到来时并在 4020 复位之前，将 74HC595 中的数据沿 Q0 到 Q7 方向依次移一位，且 4020 的 Q11 端数据移入 74HC595 的 Q0 端。对于一组遥控编码脉冲，共有 33 次上升沿（包括起标志），而 74HC595 仅为 8 位移位寄存器，所以移位的最终结果，只有遥控编码脉冲的最后 8 位保留在 74HC595 中。

当一组遥控编码脉冲（反相后）来到时，其起始标志的上跳沿触发了双单稳 74HC123 的 1B，在 1Q 上产生了一个宽度为 120ms 的正脉冲。1Q 同时又触发了 74HC123 的 2B，在产生一个宽度为 80ms 的负脉冲，1Q 和相与后作为锁存信号送至 74HC595 的 RCLK 端，即一组遥控编码脉冲到来 80ms 后，产生一个锁存信号。此时 74HC595 已经移过了一组遥控码，芯片中保留的是最后 8 位遥控码，锁存信号将这最后 8 位遥控码锁存。

3. 红外遥控信号编码、发射原理

所有红外遥控器的输出都是用编码后串行数据对 38～40kHz 的方波进行脉冲幅度调制而产生的。如果直接对已调波进行测量，由于单片机的指令周期是微秒（μs）级，而已调波的脉宽只有 20 多μs，会产生很大的误差。因此先要对已调波进行解调，对解调后的波形进行测量。

用遥控脉冲信号调制 38kHz 方波，然后将已调波放大，驱动红外发光二极管，就可以得到遥发射信号。调制可用一个或门实现，38kHz 方波可用 8751 的定时器 T1 产生。有些遥控器的载频可能是 40kHz，只须稍微加大发射功率仍然可用 38kHz 载频使其接收电路动作。

通常，红外遥控器是将遥控信号（二进制脉冲码）调制在 38KHz 的载波上，经缓冲放大后送至红外发光二极管，转化为红外信号发射出去的。二进制脉冲码的形式有多种，其中最为常用的是 PWM 码（脉冲宽度调制码）和 PPM 码（脉冲位置调制码）。前者以宽脉冲表示 1，窄脉冲表示 0。后者脉冲宽度一样，但是码位的宽度不一样，码位宽的代表 1，码位窄的代表 0。遥控编码脉冲信号（以 PPM 码为例）通常由引导码、系统码、系统反码、功能码、功能反码等信号组成。引导码也叫起始码，由宽度为 9ms 的高电平和宽度为 4.5ms 的低电平组成（不同的遥控系统在高低电平的宽度上有一定区别），用来标志遥控编码脉冲信号的开始。系统码也叫识别码，它用来指示遥控系统的种类，以区别其它遥控系统，防止各遥控系统的误动作。功能码也叫指令码，它代表了相应的控制功能，接收机中的微控制器可根据功能码的数值去完成各种功能操作。系统反码与功能反码分别是系统码与功能码的反码，反码的加入是为了能在接收端校对传输过程中是否产生差错。为了提高抗干扰性能和降低电源消耗，将上述的遥控编码脉冲对频率为 38KHz（周期为 26.3us）的载波信号进行脉幅调制（PAM），再经缓冲放大后送到红外发光管，将遥控信号发射出去。

根据遥控信号编码和发射过程，遥控信号的识别——即解码过程应是去除 38KHz 载波信号后识别出二进制脉冲码中的 0 和 1。遥控信号识别、存储、还原的硬件电路由 CPU、一体化红外接收头、存储器、还原调制与红外发光管驱动电路组成. 。一体化红外接收头负责红外遥控信号的解调。将调制在 38kHz 上的红外脉冲信号解调并反向后再由单片机进行高电平与低电平宽度的测量。

Crystal	24.576	MHz			
Max Time	13.5	ms	CDIV	40	Maximum is the startpuls (AGC burst) of 9 ms. 1.5 x 9 = 13.5ms

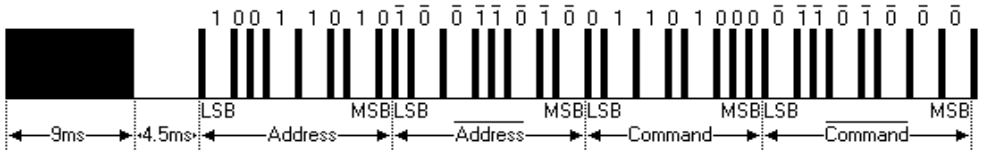
Startbit					
Min Low Time	6.75	ms	ALB	2023	Only low time is start pulse of 9ms. Minimum is 0.75 x 9 = 6.75ms
Max Low Time	11.25	ms	AHB	1349	Only low time is start pulse of 9ms. Maximum is 1.25 x 9 = 11.25ms
Min High Time	1.6875	ms	BLB	506	Minimum high time is space after a repeat pulse (2.25ms). Minimum is 0.75 x 2.25 = 1.6875ms
Max High Time	5.625	ms	BHB	1180	Maximum high time is space after a start pulse (4.5ms). Maximum is 1.25 x 4.5 = 5.625ms

Databit					
Min Low Time	0.42	ms	ALD	126	Only low time is data bit burst of 0.560ms. Minimum is 0.75 x 0.560 = 0.420ms
Max Low Time	0.7	ms	AHD	84	Only low time is data bit burst of 0.560ms. Maximum is 1.25 x 0.560 = 0.7ms
Min High Time	0.42	ms	BLD	126	Minimum high time is space in a logical '0' (1.12ms - 0.560ms = 0.560ms). Minimum is 0.75 x 0.560 = 0.420ms
Max High Time	2.1125	ms	BHD	507	Maximum high time is space in a logical '1' of (2.25ms - 0.560ms). Maximum is 1.25 x 1.69 = 2.1125ms

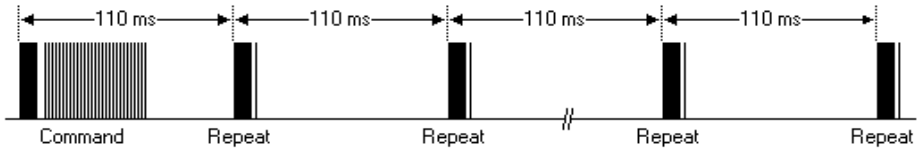
NOTE: Calculated values are based on inverted signal !! Meaning low times refer to high times in picture !

DECVAL = max.low + ((max.high - max.low) / 2) = 86 + ((520 - 86) / 2) = 303

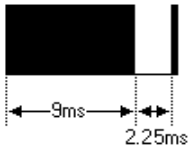
Startpuls (9ms + 4.5ms + 560us)



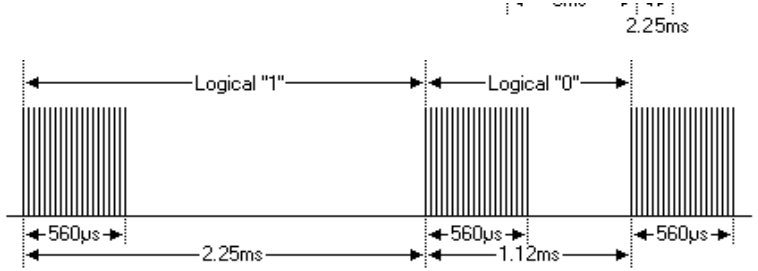
NEC Sequence



Command is transmitted once, repeat code is sent every 110ms.



Databits



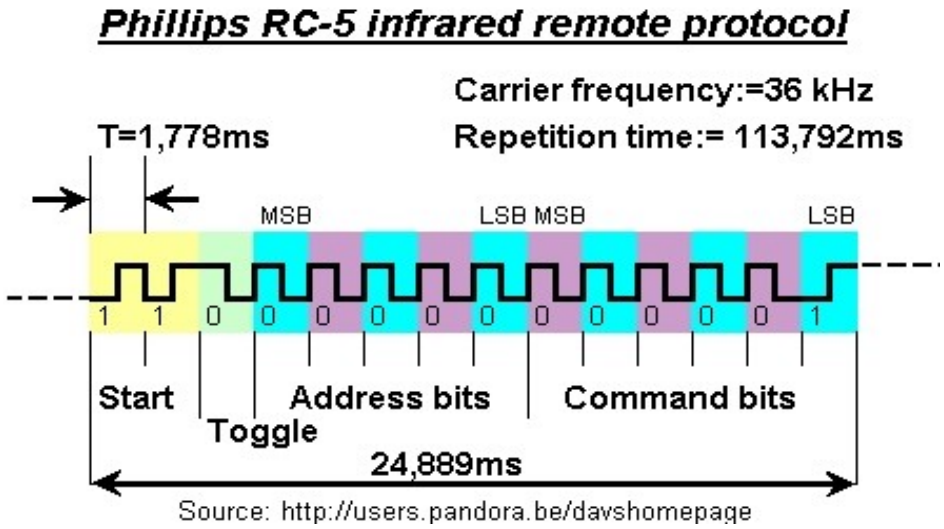
Crystal	24.576	MHz			
Max Time	2.667	ms	CDIV	7	Maximum is burst in 10 or space in 01 transition (2 x 899us). Maximum is 1.5 x 1.778 = 2.667ms

Startbit					
Min Low Time	0	ms	ALB	0	No startbit
Max Low Time	0	ms	AHB	0	
Min High Time	0	ms	BLB	0	
Max High Time	0	ms	BHB	0	

Databit					
Min Low Time	0.67425	ms	ALD	1036	Minimum is a space in 00 or 11 transition of 899us. Minimum is 0.75 x 0.899 = 0.67425ms
Max Low Time	2.2225	ms	AHD	2378	Maximum is a space in a 01 transition (2 x 899us). Maximum is 1.25 x 1.778 = 2.2225ms
Min High Time	0.67425	ms	BLD	1036	Same as ALD
Max High Time	2.2225	ms	BHD	2378	Same as AHD

RC5 uses Manchester coding (14 bits, 2 Startbits, 1 Togglebit, 5 Systembits, 6 Commandbits) --> 13 Bits are used in GTV

DECVAL n.a.



Source: http://users.pandora.be/davshomepage