

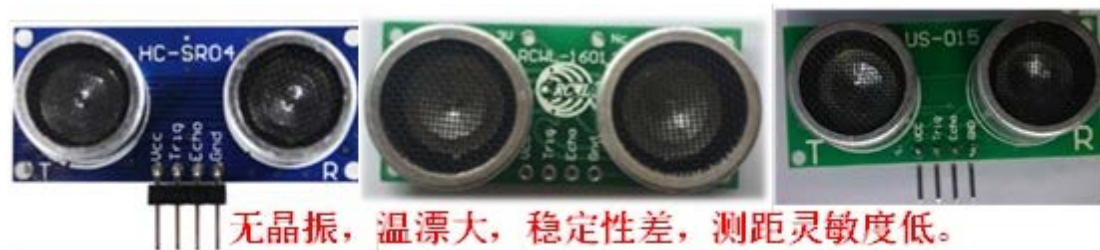
新版 HC-SR04，性能远超老版 HC-SR04，US-015；在测距精度高于老版 HC-SR04 和 US-015 的情况下，测距范围更远，可达 6 米，远超一般超声波测距模块。采用 CS-100A 超声波测距 SOC 芯片，高性能，工业级，宽电压、低价格，成本击穿底价，只有普通超声波测距模块一半的价格，而性能远超普通超声波测距模块。性能与 US-025A 相同，均采用 CS100A 芯片，接口完全兼容。

三个问题：

### 1. 超声波模块为什么要用晶振？

目前不少市场上的产品，不带晶振，可靠性一致性较差，只能适用于低端产品。不用晶振，只能使用片内 RC 振荡器，RC 振荡器一致性很差，并且温度稳定性很差，不同温度及不同个体之间偏移可达 5% 以上，不适用于测试测量场景，不适用于工业场景应用。

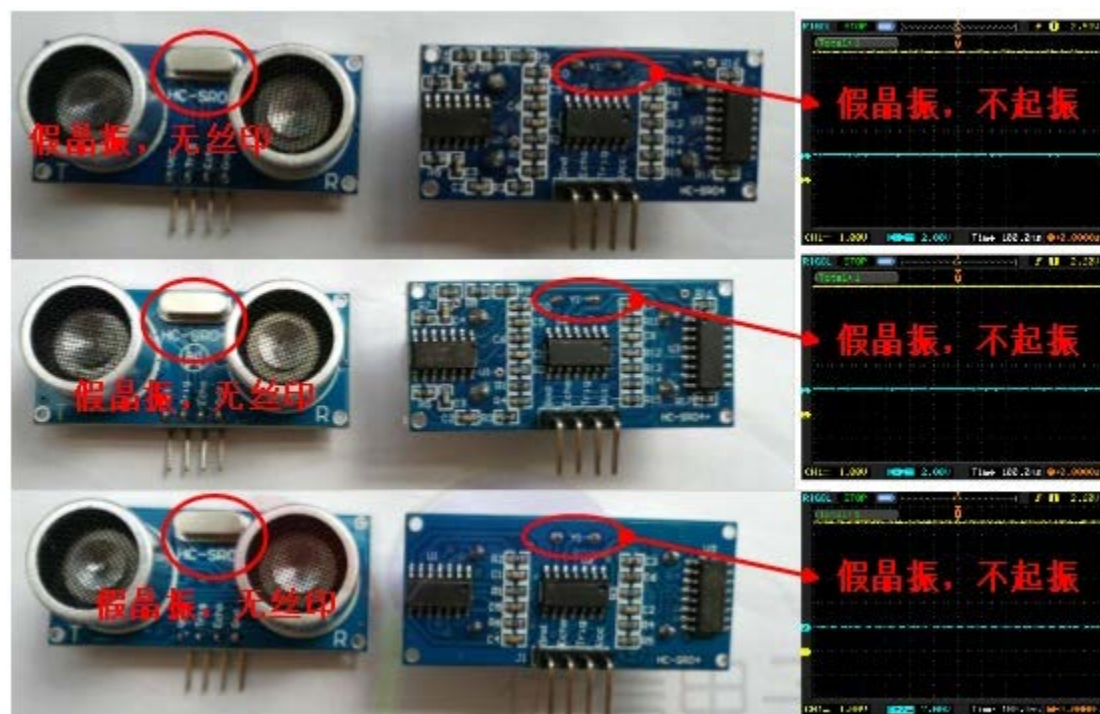
部分不用晶振的产品如下图所示：



超声波探头频率一般误差正负 1%，不采用晶振频率最大会有 5% 的误差，频率对不上导致一致性差，灵敏度降低；不使用外部晶振的模块只适用于低端及可靠性要求极低场景。

### 2. 为什么要用真晶振？

目前淘宝上很多的超声波模块，选用器件只求便宜，不管质量；用的是最差的芯片，甚至是翻新的，假的元器件；最明显的例子是晶振，可以说淘宝上在售的不带 CS100 超声波芯片的 HC-SR04 基本上都是用的假晶振，晶振不能起振；有兴趣的可以用示波器量一下晶振两个管脚，看是否能看到有波形？



新款 HC-SR04，采用优良晶振，输出标准正弦波时钟信号，如下图所示：



2. 为什么要选用正品行货？

目前，已接到多起客户反馈，之前从市场上拿的不带晶振及假晶振的超声波模块，使用中会莫名出现各种各样的问题。低温下漂移太大，长时间工作不准，间歇性工作不正常等种种问题；发现无晶振、假晶振、使用次品芯片和翻新芯片为主要原因。

电气参数	新款HC-SR04
超声波测距芯片	CS100A
工作电压	DC 3V-5.5V
工作电流	5.3mA
工作温度	-40℃-85℃
输出方式	GPIO
感应角度	小于15度
探测距离	2cm-600cm
探测精度	0.1cm+1%



## 测距工作原理

模块测距的时序如图 5 所示：

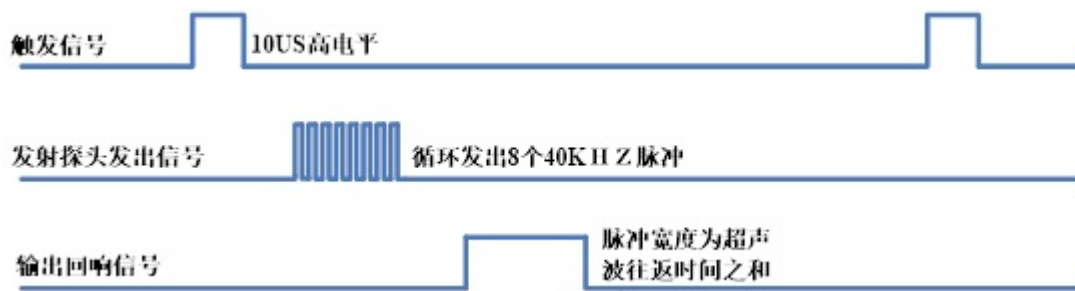


图 5：新款 HC-SR04 测距时序图

图 5 表明：只需要在 Trig 管脚输入一个 10US 以上的高电平，系统便可发出 8 个 40KHZ 的超声波脉冲，然后检测回波信号。当检测到回波信号后，通过 Echo 管脚输出。

根据 Echo 管脚输出高电平的持续时间可以计算距离值。即距离值为： $(\text{高电平时间} \times 340\text{m/s}) / 2$ 。

当测量距离超过 HC-SR04 的测量范围时，仍会通过 Echo 管脚输出高电平的信号，高电平的宽度约为 66ms。如图 6 所示：

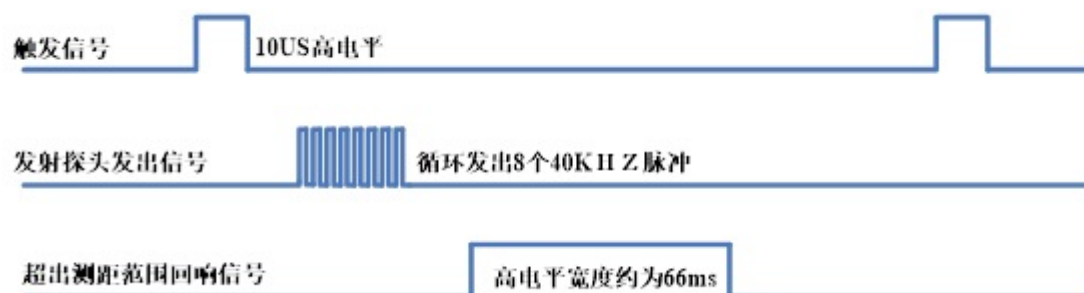


图 6：超出测量范围返回值

测量周期：当接收到 HC-SR04 通过 Echo 管脚输出的高电平脉冲后，便可进行下一次测量，所以测量周期取决于测量距离，当距离被测物体很近时，Echo 返回的脉冲宽度较窄，测量周期就很短；当距离被测物体比较远时，Echo 返回的脉冲宽度较宽，测量周期也就相应的变长。

最坏情况下，被测物体超出超声波模块的测量范围，此时返回的脉冲宽度最长，约为 66ms，所以最坏情况下的测量周期稍大于 66ms 即可（取 70ms 足够）。

附录：新款 HC-SR04 高精度测距例程，（Arduino 例程）

```
unsigned int EchoPin = 2;
unsigned int TrigPin = 3;
unsigned long Time_Echo_us = 0;
//Len_mm_X100 = length*100
unsigned long Len_mm_X100 = 0;
unsigned long Len_Integer = 0; //
unsigned int Len_Fraction = 0;
void setup()
{
  Serial.begin(9600);
  pinMode(EchoPin, INPUT);
```

```
pinMode(TrigPin, OUTPUT);
}
void loop()
{
digitalWrite(TrigPin, HIGH);
delayMicroseconds(50);
digitalWrite(TrigPin, LOW);
Time_Echo_us = pulseIn(EchoPin, HIGH);
if((Time_Echo_us < 60000) && (Time_Echo_us > 1))
{
Len_mm_X100 = (Time_Echo_us*34)/2;
Len_Integer = Len_mm_X100/100;
Len_Fraction = Len_mm_X100%100;
Serial.print("Present Length is: ");
Serial.print(Len_Integer, DEC);
Serial.print(".");
if(Len_Fraction < 10)
Serial.print("0");
Serial.print(Len_Fraction, DEC);
Serial.println("mm");
}
delay(1000);
}
```