芯岭技术

# XI5300 整机标定指引

2021/11/24

# 目录

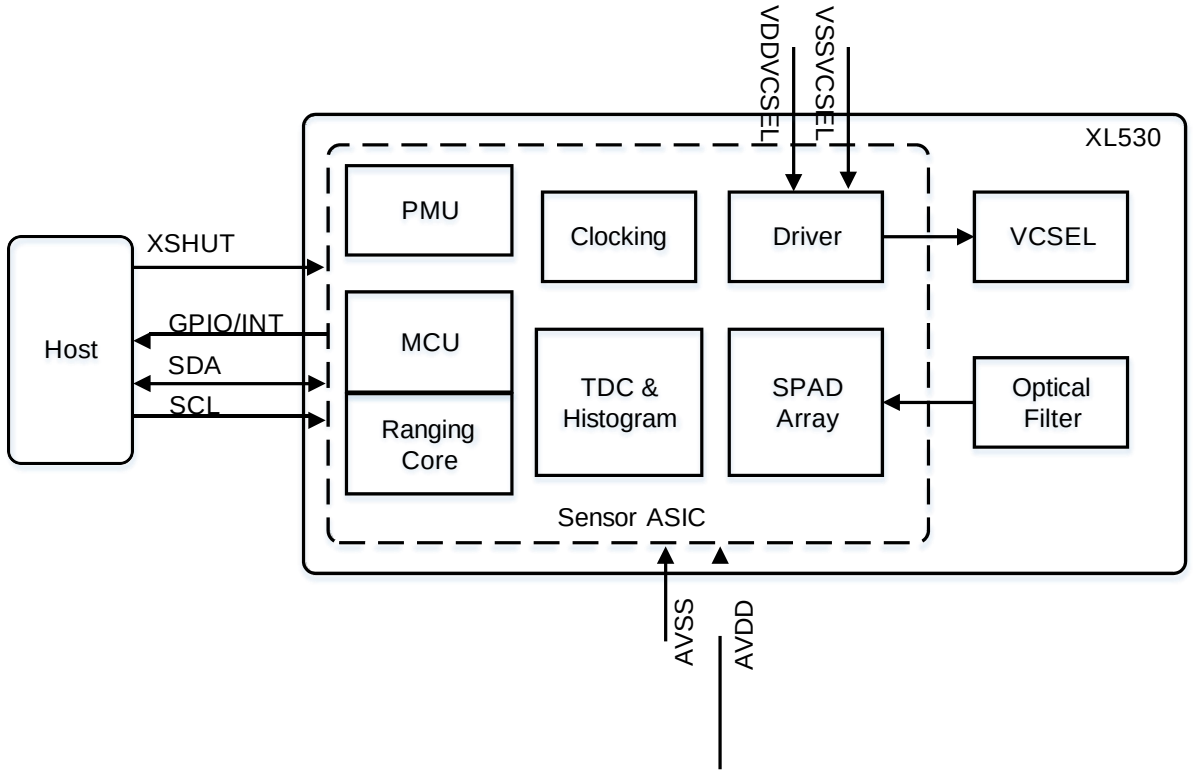
# 1D dTOF-XL5300简介

XL5300

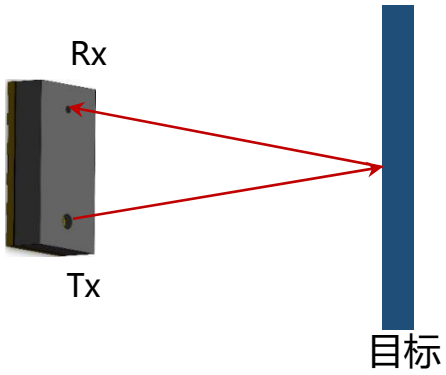
4m max.

## 产品主要特点

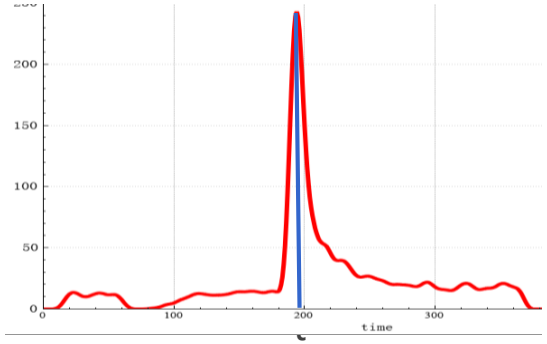
- 封装紧凑，4.4x2.4x1.0mm
- 测距范围：2.5cm- 4m
- 测距精度：+/-4%
- 帧率： 90Hz max.
- 人眼安全等级 Class 1，豁免级
- 整体方案dTOF 模组，应用设计简单



# XL5300 测距基本原理

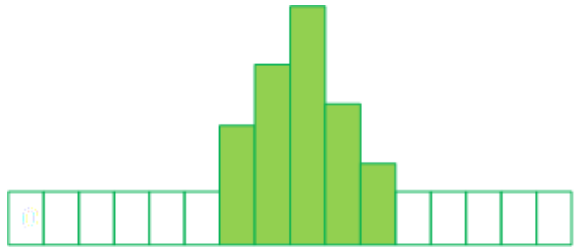



算法处理后的直方图
D=c*t/2




算法处理

多次发射光脉冲累计
结果 形成直方图



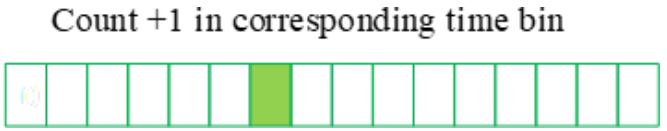
Tx发射激光脉冲

反射光触发
SPAD产生脉冲

数字整形

脉冲上升沿触发
TDC

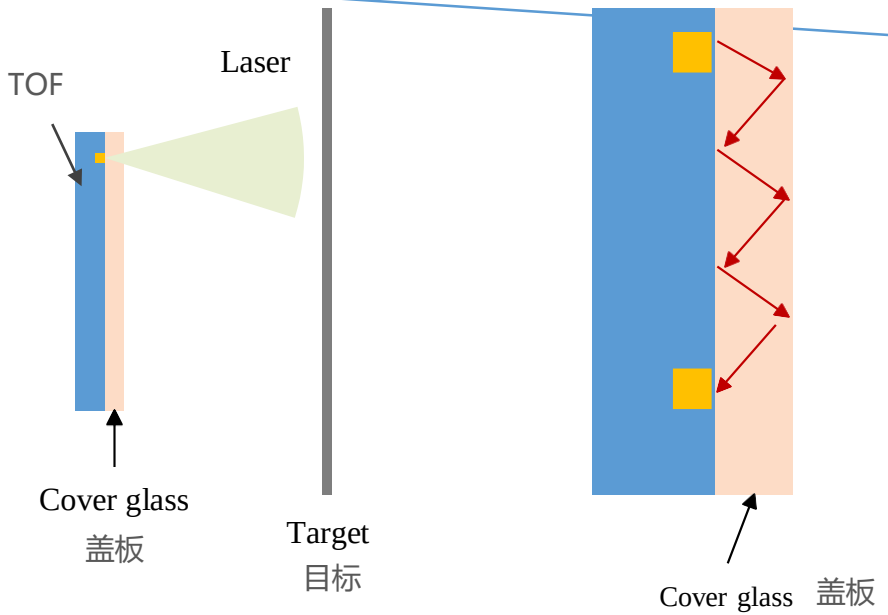
TDC

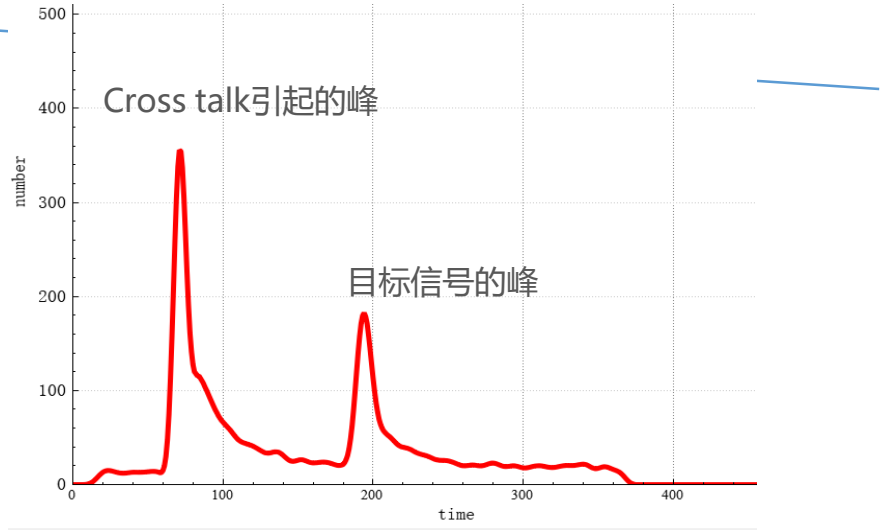



在对应时刻计数
1次

单次光脉冲可能产生的计数

# XTalk 现象以及影响



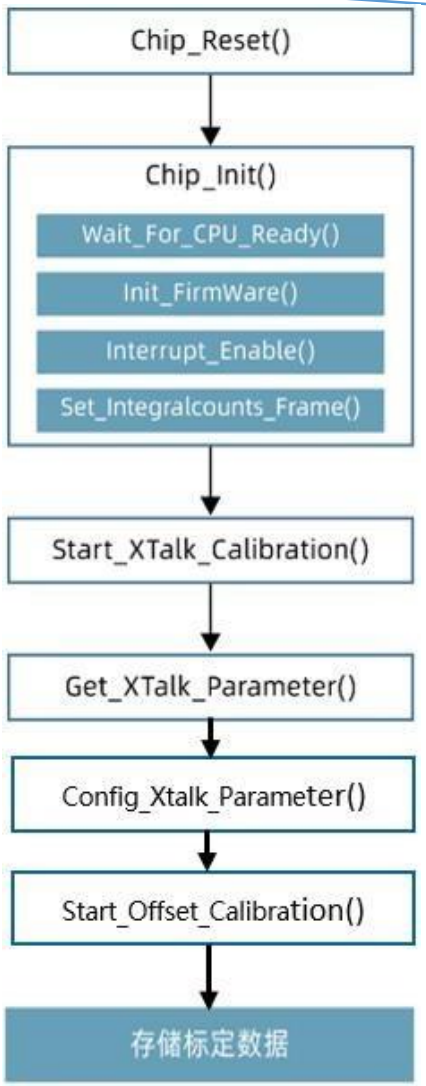




直方图

- 整机在XL5300 前方装上盖板（cover glass）后，部分光在盖板内多次反射后进入Rx 端形成Cross talk（串扰）信号。
- 当Cross talk 太大 超过目标信号的强度时，cross talk信号会被当成目标信号，此时只能测到盖板距离，无法正确测量目标信号。
- 且cross talk 太大，会影响近距离测距精度。所以 需要做cross talk标定并卡控cross talk 的范围。

注：XTalk 为cross talk的 简称。

# XL5300 整机标定




- **MCU平台**：

1.进行整机标定时，直接执行执行XL5300_Start_XTalk_Calibration()进行Xtalk标定，

2.执行XL5300_Get_XTalk_Parameter() 获得Xtalk标定结果

3.调用XL5300_ Config_XTalk_Parameter() 设置Xtalk参数

4.执行XL5300_Start_Offset_Calibration()进行offset标定获得offset标定结

果5.保存Xtalk标定值和offset标定值

- **Linux 平台：**

进行整机标定时，上层下发标定cmd(ioctl cmd/或者文件节点cmd)，

1. 执行 XL5300_Start_XTalk_Calibration() 进行Xtalk标定，

2. 执行XL5300_Get_XTalk_Parameter() 获得Xtalk标定结果，

3. 调用XL5300_ Config_XTalk_Parameter() 设置Xtalk参数

4. 执行XL5300_Start_Offset_Calibration()进行offset标定获得offset标定结果

5. .保存Xtalk标定值和offset标定值

# XL5300 XTalk标定

- **Xtalk标定环境**

  - 在60厘米内无遮挡物
  - 无IR 光干扰

- **Xtalk标定结果**

```
struct XL5300_XTALK_Calib_Data {
        uint8_t xtalk_cal;          //表示cover glass的位置，以LSB为单位
        uint16_t xtalk_peak;        //表示Xtalk的强度，读出即可，需在一定的范围内
        };
```

# XL5300 XTalk标定

- **MCU平台**：
  - 在执行XL5300_Start_XTalk_Calibration()API之后，芯片内部标定时长1s，然后读取标定数据，将结构体内的xtalk_cal（一个字节）存储在flash。
  - 在测距时，下载完firmware之后，从flash里读取xtalk_cal的值，将xtalk_cal配置到芯片firmware内，调用XL5300_Config_XTalk_Parameter() API来完成。
- **Linux 平台：**
  - 在执行XL5300_Start_XTalk_Calibration()API之后，芯片内部标定时长1s，然后读取标定数据，将结构体内的xtalk_cal（一个字节）存储在文件或者NVRAM。
  - 在测距时，下载完firmware之后，从文件里读取xtalk_cal的值，然后传到底层驱动将xtalk_cal配置到芯片firmware内，调用XL5300_Config_XTalk_Parameter() API来完成。

芯片不断电(AVDD不断电同时Xshut 不拉低不断电)，此时测距就不需要重新下发xtalk_cal。AVDD和Xshut其中之一断电，芯片内部 firmware 就会丢失，再次上电工作需要重新下载firmware，同时也需要配置xtalk_cal。总之不重新下载firmware就不需要重新配置xtalk_cal。

# XL5300 offset 标定

- **offset标定目的**
  - 提高整机不同机子之间测距性能一致性

- **offset标定环境**
  - 建议距离10厘米处白卡或灰卡

- **offset标定结果**

```
struct XL5300_OFFSET_Calib_Data
{
        int16_t offset_cal;
        int16_t ref_tof;
};
```

# XL5300 offset 标定

- **MCU平台**：

  - 在执行offset标定时，执行XL5300_Start_Offset_Calibration () API，取30帧数据求取平均值，然后减去真实距离值即为offset标定值，然后将offset标定值存储在flash。

- **Linux 平台：**

  - 在执行offset标定时，执行XL5300_Start_Offset_Calibration () API，取30帧数据求取平均值，然后减去真实距离值，即为offset标定值，然后将为offset标定值存储在文件或者NVRAM。

  - 当使用XL5300测距时，将offset从存储区读取出来，将TOF值减去offset才是最终TOF值，不需要写入芯片内。当MCU一直保持上电时，此offset值不会丢失，XL5300断电不会影响offset值。

注：在进行offset标定前，一定要将xtalk的标定参数配置进芯片，否则芯片测距不准确。

# XL5300 reftof 标定

- **Reftof 标定原因**

  - 记录在出厂环境下当前温度的reftof，在后续测距过程中温度升高，导致BVD校准不准确，必须重新 校准，利用出厂温度下的reftof和测距时的reftof比较，经过一系列计算调整BVD档位和光功率。

- **Reftof标定环境**

  - 无要求

- **Reftof标定结果**

  - struct XL5300_OFFSET_Calib_Data
    {int16_t offset_cal;
    int16_t ref_tof;
    };

- **MCU平台**：

  - 在执行offset标定时，在得到TOF值得同时可以得到reftof， reftof取30帧数据求取平均值即为reftof标定值， 标定完成之后reftof标定值存储在flash。

- **Linux平台**

  - 在执行offset标定时，在得到TOF值得同时可以得到reftof， reftof取30帧数据求取平均值即为reftof标定值， 标定完成之后reftof标定值。当使用XL5300测距时，在下载完firmware之后，测距之前将reftof标定结果从存储区读取出来配置到XL5300的芯片里。XL5300不断电，reftof的值一直保存在芯片内部，不需要重新配置，如果掉电，则需要重新写入。

注：通常reftof标定和offset标定同时完成，同时保存。