

MLX90640 Thermal Camera

用户手册

产品特点

本模块一款红外热像仪模块，32×24 像素，I2C 接口通信，兼容 3.3V/5V 电平，支持 Raspberry Pi、Arduino 等主控。我采用 MLX90640 远红外热传感器阵列，可精确检测特定区域和温度范围内的目标物体，尺寸小巧，可方便集成到各种工业或智能控制应用中。

- 采用 MLX90640 远红外热传感器阵列，32×24 像素
- 支持 I2C 接口通信，可设置为快速模式(速率可达 1MHz)
- 噪声等效温差(NETD)仅为 0.1K RMS@1Hz 刷新率，噪声性能好
- 板载电平转换电路，可兼容 3.3V/5V 的工作电平

[照片]

参数:

工作电压:3.3V/5V

工作电流:<23mA

通信接口:I2C (地址为 0x33)

视场角(水平视角×垂直视角):

- MLX90640-D55 Thermal Camera: 55° × 35° (角度小, 适合远距离测量)
- MLX90640-D110 Thermal Camera: 110° × 75° (角度大, 适合近距离测量)

工作温度:-40℃~85℃

目标温度:-40℃~300℃

检测精度:±1℃

刷新速率:0.5Hz~64Hz (可编程设置)

产品尺寸:28mm×16 mm

固定孔尺寸:2.0mm

主要用途:

- 高精度非接触性物体温度检测
- 红外热像仪、红外测温仪
- 智能家居、智能楼宇、智能照明
- 工业温度控制、安防、入侵/移动检测

接口说明(以接入 MCU 为例):

VCC: 接 3.3V

GND: 接 GND

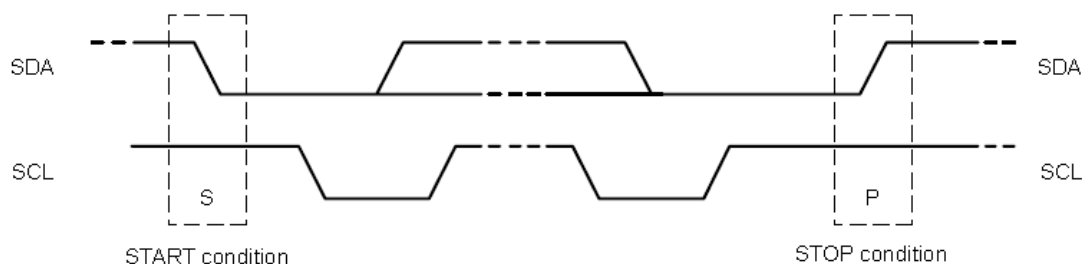
SDA: 接 MCU I2C 数据线

SCL: 接 MCU I2C 时钟线

通讯方式:

本模块的通讯方式为 I2C，支持 I2C 高速模式(最高可达 1MHz)，只能作为 I2C 总线上的从设备，SDA 和 SCL 端口可以承受 5V 电压，可以直接接入到 5V I2C 总线中，模块的设备地址是可以编程的，最多可以有 127 个地址，出场默认值为 0x33。

与一般 I2C 总线一样，在传送数据过程中共有三种类型信号：开始信号、结束信号和应答信号。

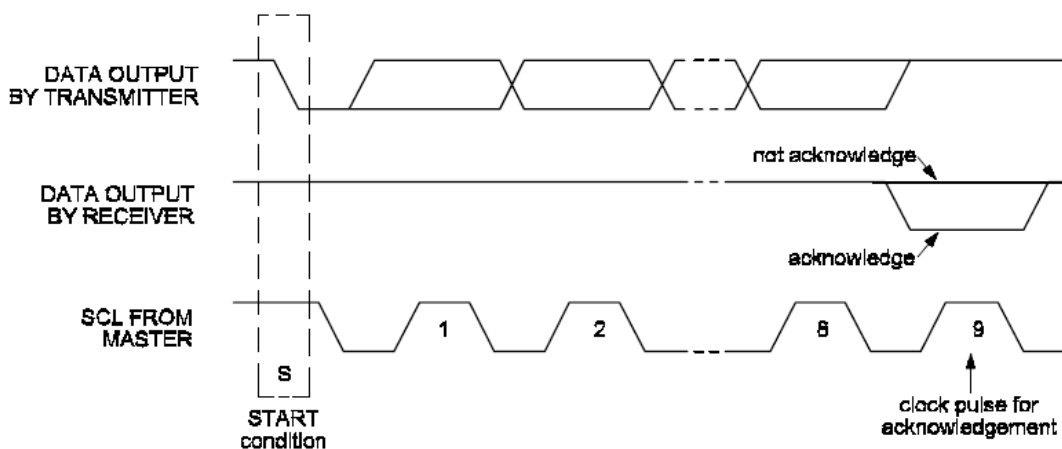


起始和停止条件

开始信号：SCL 为高电平，SDA 由高电平转换为低电平。

结束信号：SCL 为高电平，SDA 由低电平转换为高电平。

可以看出开始信号和结束信号都是在 SCL 总线为高电平时刻完成的。

I²C 总线的响应

应答信号：在每个字节传输之后的第 9 个时钟期间内，发送数据端设备释放 SDA 总线，接受数据端设备拉低 SDA 总线表示收到字节 (ACK)，或者是 SDA 总线为高电平不应答 (NoACK)。

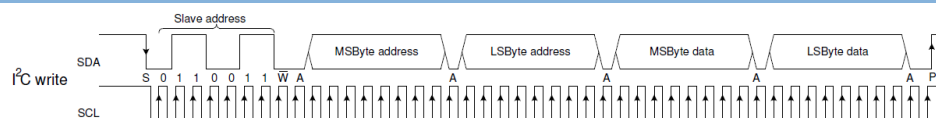
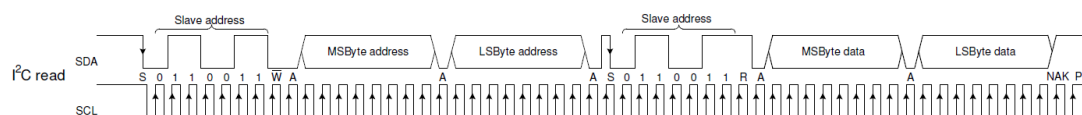
设备地址：

主机通过在 START 条件后发送 7 位从机地址来寻址从机。前七个位是该地址专用，第 8 个是读/写 (R / W) 位。该位指示传输方向：

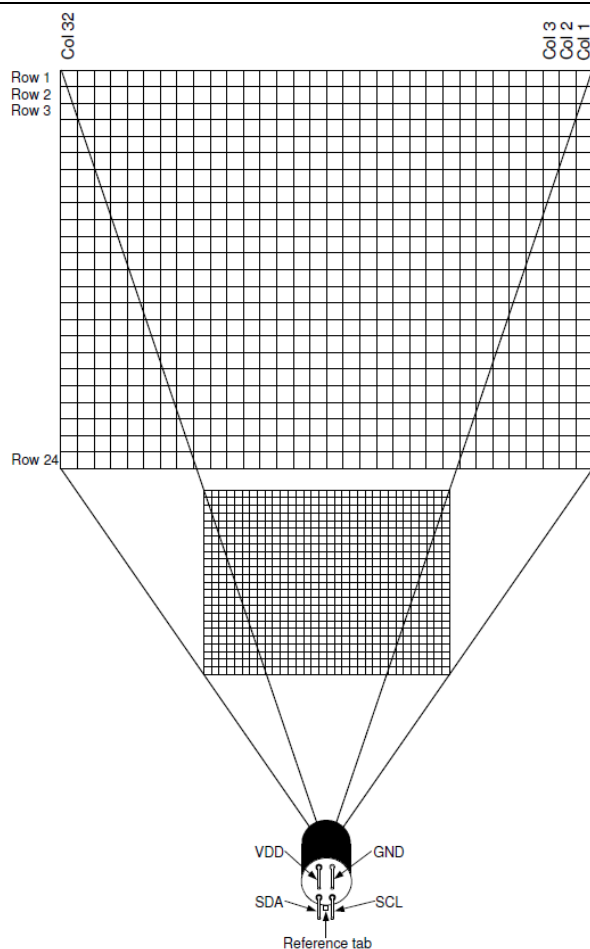
Read (1) 表示主机将从从机读取数据

Write (0) 表示主机将向从机发送数据

通信格式：

Figure 4 I²C write command format (default SA=0x33 is used)Figure 5 I²C read command format (default SA=0x33 is used)

本模块共具有 768 个 IR 传感器 (也称为像素) 组成。每个像素的行和列位置标识为 Pix (i, j)，其中 i 是其行号 (从 1 到 24)，j 是其列号 (从 1 到 32)，像素具体到某一平面可以参照下图。



*需要说明的是传感器原厂在传感器出厂时允许有 4 个以内的坏点，每个坏点都在 EEPROM 表中有标识，所以模块可能会有一定几率存在坏点，也就是说这不能作为退换货的依据，对此原厂的建议是使用相邻像素的平均值代替。

内存地址分布：

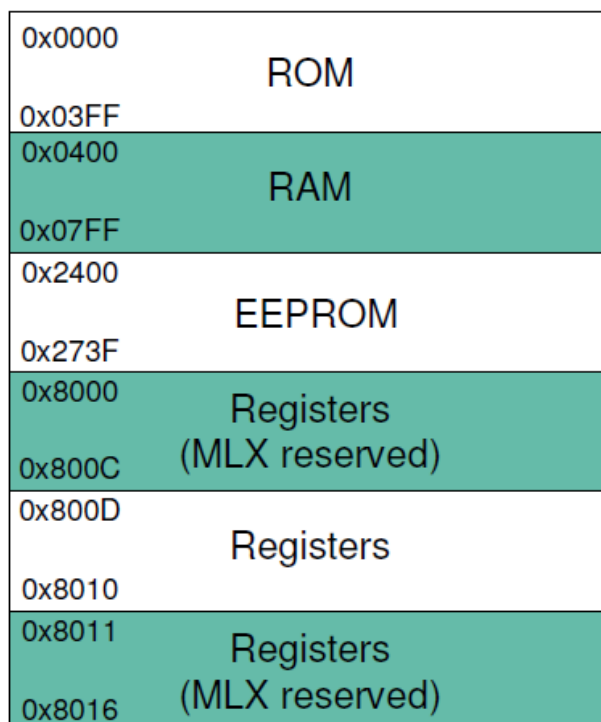


Figure 10 MXL90640 memory map

两种 RAM 数据模式:

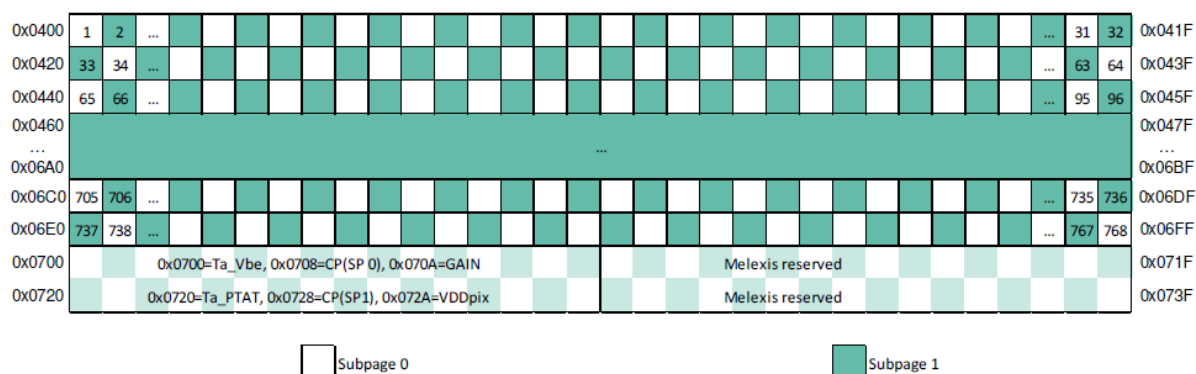


Figure 14 RAM memory map (Chess pattern mode) – factory default mode

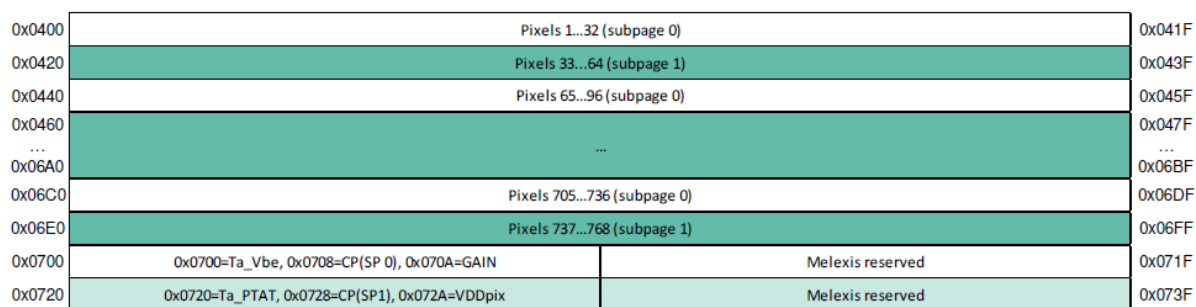


Figure 15 RAM memory map (Interleaved mode)

EEPROM 用于存储校准常数和设备的配置参数:

EEPROM address	Access	Meaning
0x2400	Melexis	Melexis reserved
0x2401	Melexis	Melexis reserved
0x2402	Melexis	Melexis reserved
0x2403	Melexis	Configuration register
0x2404	Melexis	Melexis reserved
0x2405	Melexis	Melexis reserved
0x2406	Melexis	Melexis reserved
0x2407	Melexis	Device ID1
0x2408	Melexis	Device ID2
0x2409	Melexis	Device ID3
0x240A	Melexis	Device Options
0x240B	Melexis	Melexis reserved
0x240C	Customer	Control register_1
0x240D	Customer	Control register_2
0x240E	Customer	I2CConfReg
0x240F	Customer	Melexis reserved / I2C_Address

Table 7 Configuration parameters memory

刷新率:

本模块共支持 8 种刷新率，最高可达 64Hz，刷新率由控制寄存器 1-0x800D 控制，如下图:

B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
Melexis reserved			Reading pattern	Resolution control		Refresh rate control			Select subpage			Enable subpages repeat	Enable data hold	Melexis reserved	Enable subpages mode
Control register 1 - 0x800D															
0 No subpages, only one page will be measured															
1 Subpage mode is activated (default)															
0 Keep this bit = "0" (default)															
0 Transfer the data into storage RAM at each measured frame (default)															
1 Transfer the data into storage RAM only if en_overwrite = 1 (check 0x8000)															
0 Toggles between subpage "0" and subpage "1" if Enable subpages mode = "1" (default)															
1 Select subpage determines which subpage to be measured if Enable subpages mode = "1"															
0 0 0 Subpage 0 is selected (default)															
0 0 1 Subpage 1 is selected															
0 1 0 Not Applicable															
0 1 1 Not Applicable															
1 0 0 Not Applicable															
1 0 1 Not Applicable															
1 1 0 Not Applicable															
1 1 1 Not Applicable															
0 0 0 IR refresh rate = 0.5Hz															
0 0 1 IR refresh rate = 1Hz															
0 1 0 IR refresh rate = 2Hz (default)															
0 1 1 IR refresh rate = 4Hz															
1 0 0 IR refresh rate = 8Hz															
1 0 1 IR refresh rate = 16Hz															
1 1 0 IR refresh rate = 32Hz															
1 1 1 IR refresh rate = 64Hz															
0 0 ADC set to 16 bit resolution															
0 1 ADC set to 17 bit resolution															
1 0 ADC set to 18 bit resolution (default)															
1 1 ADC set to 19 bit resolution															
0 Interleaved (TV) mode															
1 Chess pattern (default)															
Melexis reserved															

8 种刷新率的设置是取决于控制寄存器 1-0x800D 的位 7、位 8、位 9。

像素排列模式：

国际象棋模式（出厂默认设置）

Subpage 0 --> 0x8000 = 0xFFFF

0x000	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31
0x020	34	36	38	40	42	44	46	48	50	52	54	56	58	60	62	64
0x040	65	67	69	71	73	75	77	79	81	83	85	87	89	91	93	95
0x060	96	98	100	102	104	106	108	110	112	114	116	118	120	122	124	126
0x080	129	131	133	135	137	139	141	143	145	147	149	151	153	155	157	159
0x0A0	162	164	166	168	170	172	174	176	178	180	182	184	186	188	190	192
0x0C0	193	195	197	199	201	203	205	207	209	211	213	215	217	219	221	223
0x0E0	226	228	230	232	234	236	238	240	242	244	246	248	250	252	254	256
0x100	257	259	261	263	265	267	269	271	273	275	277	279	281	283	285	287
0x120	290	292	294	296	298	300	302	304	306	308	310	312	314	316	318	320
0x140	321	323	325	327	329	331	333	335	337	339	341	343	345	347	349	351
0x160	354	356	358	360	362	364	366	368	370	372	374	376	378	380	382	384
0x180	385	387	389	391	393	395	397	399	401	403	405	407	409	411	413	415
0x1A0	418	420	422	424	426	428	430	432	434	436	438	440	442	444	446	448
0x1C0	449	451	453	455	457	459	461	463	465	467	469	471	473	475	477	479
0x1E0	482	484	486	488	490	492	494	496	498	500	502	504	506	508	510	512
0x200	513	515	517	519	521	523	525	527	529	531	533	535	537	539	541	543
0x220	546	548	550	552	554	556	558	560	562	564	566	568	570	572	574	576
0x240	577	579	581	583	585	587	589	591	593	595	597	599	601	603	605	607
0x260	610	612	614	616	618	620	622	624	626	628	630	632	634	636	638	640
0x280	641	643	645	647	649	651	653	655	657	659	661	663	665	667	669	671
0x2A0	674	676	678	680	682	684	686	688	690	692	694	696	698	700	702	704
0x2C0	705	707	709	711	713	715	717	719	721	723	725	727	729	731	733	735
0x2E0	738	740	742	744	746	748	750	752	754	756	758	760	762	764	766	768

Subpage 1 --> 0x8000 = 0xFFFF

0x000	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32
0x020	35	37	39	41	43	45	47	49	51	53	55	57	59	61	63	65
0x040	66	68	70	72	74	76	78	80	82	84	86	88	90	92	94	96
0x060	97	99	101	103	105	107	109	111	113	115	117	119	121	123	125	127
0x080	130	132	134	136	138	140	142	144	146	148	150	152	154	156	158	160
0x0A0	161	163	165	167	169	171	173	175	177	179	181	183	185	187	189	191
0x0C0	194	196	198	200	202	204	206	208	210	212	214	216	218	220	222	224
0x0E0	225	227	229	231	233	235	237	239	241	243	245	247	249	251	253	255
0x100	258	260	262	264	266	268	270	272	274	276	278	280	282	284	286	288
0x120	289	291	293	295	297	299	301	303	305	307	309	311	313	315	317	319
0x140	322	324	326	328	330	332	334	336	338	340	342	344	346	348	350	352
0x160	353	355	357	359	361	363	365	367	369	371	373	375	377	379	381	383
0x180	386	388	390	392	394	396	398	400	402	404	406	408	410	412	414	416
0x1A0	417	419	421	423	425	427	429	431	433	435	437	439	441	443	445	447
0x1C0	450	452	454	456	458	460	462	464	466	468	470	472	474	476	478	480
0x1E0	481	483	485	487	489	491	493	495	497	499	501	503	505	507	509	511
0x200	514	516	518	520	522	524	526	528	530	532	534	536	538	540	542	544
0x220	545	547	549	551	553	555	557	559	561	563	565	567	569	571	573	575
0x240	578	580	582	584	586	588	590	592	594	596	598	600	602	604	606	608
0x260	609	611	613	615	617	619	621	623	625	627	629	631	633	635	637	639
0x280	642	644	646	648	650	652	654	656	658	660	662	664	666	668	670	672
0x2A0	673	675	677	679	681	683	685	687	689	691	693	695	697	699	701	703
0x2C0	706	708	710	712	714	716	718	720	722	724	726	728	730	732	734	736
0x2E0	737	739	741	743	745	747	749	751	753	755	757	759	761	763	765	767

Figure 9 Chess reading pattern (only highlighted cells are updated)

电视交错模式

Subpage 0 --> 0x8000 = 0xFFFF

0x0000	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
0x0400	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
0x0800	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96
0x0C00	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160
0x0400	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224
0x0800	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288
0x0C00	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352
0x0400	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426
0x0800	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480
0x0C00	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	544
0x0400	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607	608
0x0800	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671	672
0x0C00	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735	736

Subpage 1 --> 0x8000 = 0xFFFF

0x0000	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
0x0400	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
0x0800	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128
0x0C00	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192
0x0400	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264
0x0800	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320
0x0C00	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384
0x0400	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448
0x0800	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512
0x0C00	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	576
0x0400	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	640
0x0800	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703	704
0x0C00	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735	736
0x0400	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767	768

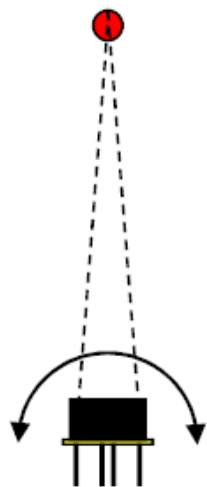
Figure 8 TV mode reading pattern (only highlighted cells are updated)

两种模式在子页面的更新方式上不同，这里需要注意的是传感器仅在国际象棋模式下进行过出厂校准，因此在国际象棋模式下可以获得更好的固定图案噪声行为，因此为了获得最佳效果建议使用国际象棋棋盘模式，两种模式的设定取决于控制寄存器 1-0x800D 的位 12。

测量原理：

对于非接触式红外测温模块，很重要的一个概念是“视场（FOV）”。视场是由温差电堆接收到 50% 的辐射信号来确定的，并且和传感器的主轴线相关。测得的温度是视场内被测物体的温度加权平均值，所以当被测物体完全覆盖 FOV 视场时的准确度是最高的。

Point heat source



Rotated sensor

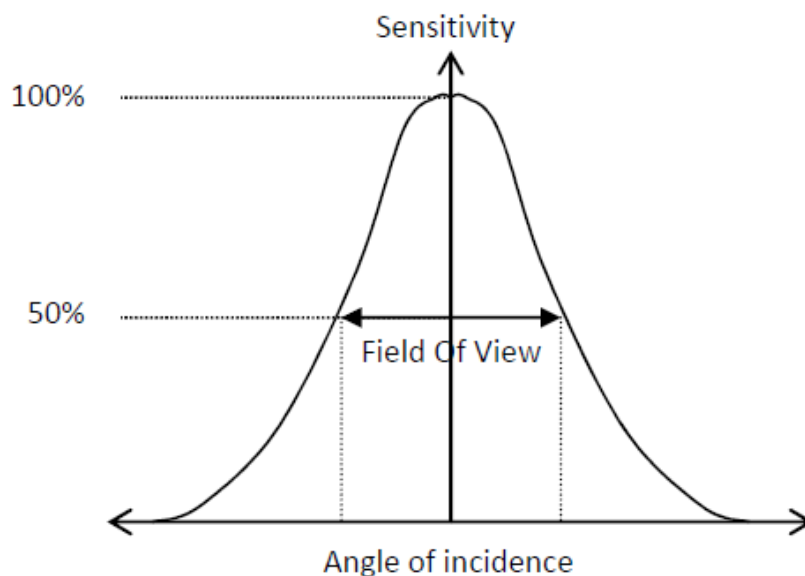


Figure 24: Field Of View measurement

其中被测物体的面积与模块测量距离满足以下关系：

$$s : D = 2 \tan \left(\frac{FOV}{2} \right)$$

测量流程：

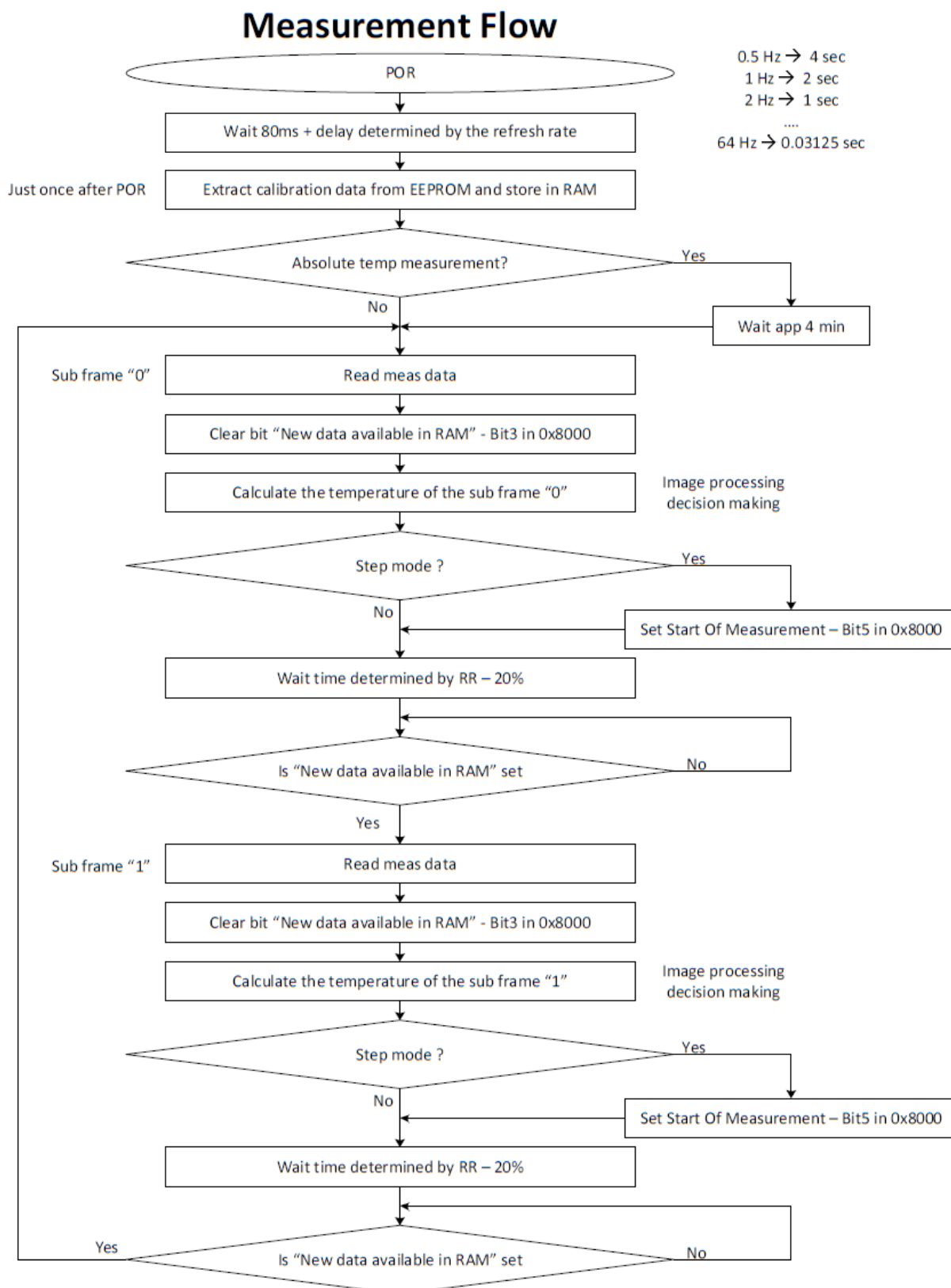


Figure 7 Recommended measurement flow

可以看出整个测量流程还是比较麻烦的，其中还包含了很多数学计算，比如计算供电电压：

11.2.2.2. Supply voltage value calculation (common for all pixels)

$$V_{dd} = \frac{Resolution_{corr} * RAM[0x072A] - V_{dd25}}{K_{Vdd}} + V_{dd0}$$

Where: Constants calculation of the EEPROM stored values (can be done just once after POR)

$$K_{Vdd} = \frac{EE[0x2433] \& 0xFF00}{2^8} = \frac{0x9D68 \& 0xFF00}{2^8} = 0x009D = 157$$

$$\text{If } 157 > 127 \rightarrow K_{Vdd} = 157 - 256 = -99$$

$$K_{Vdd} = K_{Vdd} * 2^5 = -99 * 32 = -3168$$

$$V_{dd25} = EE[0x2433] \& 0x00FF = 0x9D68 \& 0x00FF = 0x0068 = 104$$

$$V_{dd25} = (V_{dd25} - 256) * 2^5 - 2^{13} = -152 * 32 - 8192 = -13056$$

VDD calculations:

$$RAM[0x072A] = 0xCCC5 = 52421$$

$$\text{If } 52883 > 32767 \rightarrow RAM[0x072A] = 52421 - 65536 = -13115 \text{ LSB}$$

$$V_{dd} = \frac{1 * -13115 - (-13056)}{-3168} + 3.3 = \frac{-59}{-3168} + 3.3 \approx 0.0186 + 3.3 \approx 3.319V$$

从上图可以看到公式还是比较繁琐的，如果用代码写出来有可能还会出错，当然这仅仅是一个参数的计算，还有很多参数都是需要我们计算的，不过不用担心，迈来芯的工程师已经为我们写好驱动了，我们只需要移植一下驱动就可以很简单方便的计算出每一个像素点的温度值了，驱动可以从下面链接上获取到最新版本：

<https://github.com/melexis/mlx90640-library>

代码分析：

下面我们通过 STM32 移植一下驱动代码去实际分析一下工作流程，代码以 STM32F405RGT6 最小系统板为例：

实验现象

1) 用杜邦线将传感器模块按照以下接线方式连接起来：

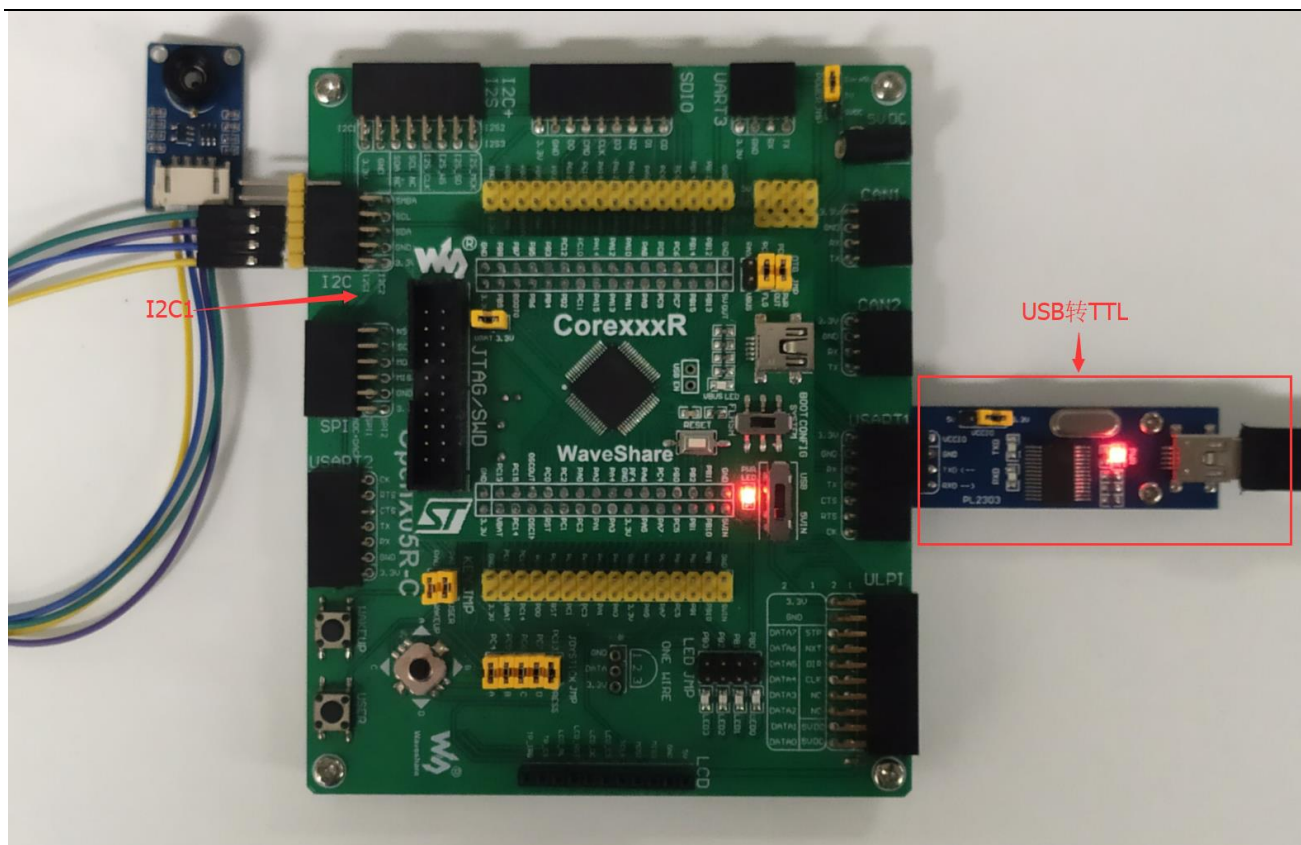
STM32F405RGT6 最小系统板 <-----> MLX90640 Thermal Camera

PB10	SCL
PB11	SDA
3V3	VCC
GND	GND

STM32F405RGT6 最小系统板 <-----> USB 转 TTL 串口助手

PA9	RX
GND	GND

如图：



- 2) 用 keil 软件打开程序：\MDK-ARM\F405_HAL_HWI2C_MLX90640.uvprojx，编译、下载。
- 3) 打开串口监视软件，选择正确的串口号，并设置如下：波特率：115200；数据位：8；停止位：1；校验位：None；控制流：None。

预期结果：

将手掌正对着传感器镜头，距离尽量近一点，可以看到串口打印的温度值大多数都接近人的体温：

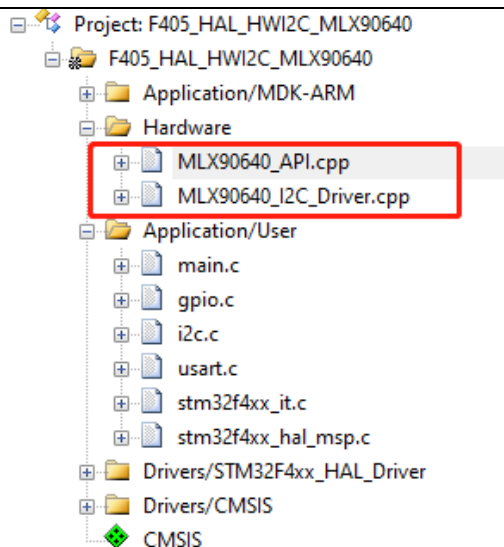
```

WaveShare
30.90 31.54 32.40 32.36 32.75 32.28 32.52 32.64 33.13 32.58 33.26 32.76 33.03 33.10 33.24 32.88 33.39 32.62 32.99 33.37 32.95 32.84 33.11 33.00 33.08 32.87 32.78 32.69 32.66 32.54 32.44 32.49
30.71 31.64 32.77 32.60 32.76 32.80 32.89 33.03 32.70 33.06 33.20 33.10 32.85 33.19 33.04 33.19 32.99 33.13 33.14 33.21 33.11 33.28 32.91 33.23 32.74 32.97 32.97 32.86 32.73 32.91 33.05 32.41
31.84 32.08 32.96 32.66 32.93 32.68 33.25 32.82 33.05 32.99 32.93 33.01 33.00 33.52 33.08 32.97 33.03 33.04 33.20 33.12 33.21 32.96 33.19 33.36 32.97 32.91 33.10 33.06 33.20 32.67 32.43 32.48
32.05 32.30 32.58 32.86 32.59 32.84 32.92 33.06 32.99 32.95 33.21 33.19 33.15 33.07 33.34 33.52 32.99 33.28 33.05 33.18 33.16 33.33 33.36 33.11 33.17 33.12 32.90 33.19 32.96 32.82 32.84 32.71
32.41 31.90 32.79 32.70 33.27 32.68 33.08 33.04 33.04 33.01 33.05 32.81 33.27 33.28 33.17 33.08 33.12 33.40 33.08 33.21 33.41 33.37 33.10 32.96 33.22 33.09 32.94 32.99 33.06 33.05 33.18
32.16 32.31 32.74 32.58 32.72 33.21 33.04 33.03 33.02 33.22 33.03 33.19 33.06 33.14 33.33 33.05 33.28 33.36 33.33 33.36 33.19 33.14 33.37 33.05 33.18 33.08 32.91 32.98 32.94 32.96 32.37 32.87
32.31 32.59 32.64 32.84 33.00 33.02 33.11 32.84 33.29 33.15 33.26 33.12 33.30 33.13 33.69 33.35 33.38 33.31 33.10 33.32 33.18 33.20 32.93 33.31 33.19 32.95 33.04 33.12 32.98 33.07 32.76 32.88
32.11 32.39 32.77 33.28 32.86 33.03 33.11 33.28 33.19 33.16 33.35 33.19 33.20 33.27 33.16 33.49 33.21 33.27 33.28 33.30 33.07 33.34 33.20 33.39 33.07 33.26 32.90 33.22 32.96 32.98 32.74 32.90
32.73 32.54 32.69 33.13 33.39 32.93 33.02 32.97 33.31 32.70 33.14 33.22 33.27 33.18 33.22 33.38 33.43 33.12 33.24 33.22 33.15 33.18 33.30 33.39 33.35 33.25 33.14 33.02 33.19 33.12 32.85 32.65
32.24 32.65 33.08 33.18 32.91 33.08 33.04 33.19 33.04 33.25 33.16 33.22 33.28 33.24 33.38 33.36 33.19 33.20 33.41 33.45 33.33 33.33 33.00 33.32 33.48 33.24 33.26 33.03 33.24 33.17 33.03 33.08 32.87
32.65 32.48 33.18 32.85 32.97 32.80 33.22 32.96 33.30 33.09 33.27 33.16 33.38 33.20 33.41 33.17 33.22 33.25 33.35 33.53 33.17 33.21 33.34 33.29 33.33 33.16 33.30 33.37 33.09 32.71 33.07 32.87
32.50 32.61 32.69 32.90 32.89 32.96 32.93 32.87 33.14 33.29 33.38 33.33 33.00 33.35 33.28 33.32 33.40 33.17 33.36 33.36 33.28 33.24 33.16 33.48 33.04 33.20 32.87 33.16 32.99 33.11 32.92 33.06
32.95 32.73 33.16 32.95 33.20 33.08 33.16 33.32 33.32 33.09 33.25 33.31 33.19 33.17 33.64 33.10 33.34 33.38 33.41 33.45 33.38 33.28 33.30 33.40 33.22 33.28 33.28 33.06 33.10 33.16 33.06 33.16
32.54 32.89 32.82 33.03 32.58 32.94 33.20 33.18 32.93 33.03 33.30 33.41 33.32 33.26 33.37 33.35 33.38 33.31 33.05 33.53 33.19 33.24 33.18 33.23 33.17 32.80 33.18 33.21 33.12 32.97 32.68 33.11
32.86 32.81 32.69 32.69 33.34 33.15 33.10 33.48 33.10 33.22 33.28 33.10 33.24 33.14 33.28 33.43 33.39 33.24 33.44 33.42 33.10 33.33 33.10 33.35 33.26 32.91 33.13 32.95 33.27 32.84 32.98 33.04
32.45 32.53 32.61 33.06 32.95 33.13 32.96 33.04 33.07 33.21 33.27 33.32 33.07 33.26 33.35 33.37 33.22 33.36 33.27 33.62 33.15 33.22 33.06 33.13 32.83 33.08 33.29 33.30 33.18 33.04 32.98 33.39
32.64 32.61 32.79 33.07 33.22 32.86 33.20 33.28 33.17 33.29 33.24 33.39 33.24 33.58 33.38 33.46 33.33 33.34 33.55 33.20 33.46 33.29 33.28 33.18 33.06 33.22 33.10 33.07 33.11 32.90 32.78 32.57
32.52 32.78 32.84 32.95 32.83 32.64 33.07 33.15 32.96 33.01 33.22 33.54 33.34 33.40 33.11 33.20 33.33 33.48 33.28 33.32 33.36 33.26 33.35 33.25 33.11 33.19 33.26 33.27 32.79 33.11 32.79 33.30
32.34 32.57 32.51 33.33 32.98 32.98 33.35 33.19 33.02 33.07 33.27 33.14 33.61 33.12 33.53 33.46 33.20 33.22 33.30 33.32 33.47 33.20 33.16 33.41 32.88 33.04 33.37 32.95 32.99 32.75 33.22 32.96
31.94 32.48 32.56 33.10 32.56 32.78 32.97 33.10 32.77 33.07 33.17 33.13 33.38 33.18 33.25 33.20 33.20 33.27 33.48 33.19 32.91 33.34 33.40 33.35 33.13 33.44 33.44 33.02 32.75 33.56 33.10
32.51 31.96 32.53 32.93 33.06 33.43 32.96 32.88 33.09 33.00 33.27 33.22 33.53 33.17 33.17 33.27 33.22 33.22 33.40 33.42 33.38 33.49 33.44 33.39 33.41 33.22 33.07 32.72 32.76 32.76 32.43 32.97
31.72 32.21 32.62 32.71 32.89 32.71 32.81 32.77 32.88 32.94 32.95 33.22 33.16 33.10 33.18 33.17 33.14 33.35 33.01 33.34 33.08 33.31 32.99 33.30 32.81 33.12 32.90 33.13 32.78 32.77 32.83 32.85
32.05 32.02 32.00 32.23 32.58 32.99 33.08 33.09 33.19 33.13 33.21 33.05 33.26 32.91 33.35 33.21 33.30 32.98 33.16 33.26 33.52 33.31 33.38 33.16 33.17 32.95 33.31 32.90 33.00 32.65 32.72 32.91
29.55 31.52 31.85 32.45 32.41 33.29 32.70 32.83 32.50 33.12 32.90 33.12 32.96 33.31 33.07 33.18 32.71 33.37 33.09 33.35 32.90 33.04 33.31 33.22 32.79 32.91 33.02 33.12 32.61 32.75 32.52 31.81
WaveShare
  
```

接下来我们分析一下迈来芯给的驱动是如何移植到我们的工程中的，以及大体工作流程是怎样的。首先我们先看一下驱动程序都有哪些文件，这里只看一下 C 文件，头文件可以自己去看一下，如下图：

名称	修改日期	类型	大小
MLX90640_API.cpp	2019/9/27 21:43	C++ 文件	40 KB
MLX90640_I2C_Driver.cpp	2019/9/27 22:51	C++ 文件	2 KB
MLX90640_SWI2C_Driver.cpp	2019/9/19 20:52	C++ 文件	8 KB

其中 MLX90640_API.cpp 主要就是前面我们讲的主要负责计算的部分，这一部分也是我们唯一不需要改动的地方，我们需要改动的地方只有其余的两个文件，这里我们使用的是 STM32 的硬件 I2C，所以第三个文件我们不需要，所以可以不用加载到工程中去，最后我们的工程树如下图就可以：



添加好以后我们还需要适配 I2C 驱动，这样才能使得 MLX90640_API.cpp 文件可以正确的通过 MLX90640_I2C_Driver.cpp 文件提供的方法读取传感器的值，如下图，将 MLX90640_I2CRead() 方法和 MLX90640_I2CWrite() 关联 STM32 的 I2C 读写函数即可：

读函数

```
int MLX90640_I2CRead(uint8_t slaveAddr, uint16_t startAddress, uint16_t nMemAddressRead, uint16_t *data)
{
    uint8_t* bp = (uint8_t*) data;

    int ack = 0;
    int cnt = 0;

    ack = HAL_I2C_Mem_Read(&hi2c2, (slaveAddr<<1), startAddress, I2C_MEMADD_SIZE_16BIT, bp, nMemAddressRead*2, 500);

    if (ack != HAL_OK)
    {
        return -1;
    }

    for(cnt=0; cnt < nMemAddressRead*2; cnt+=2) {
        uint8_t tmpbytelsb = bp[cnt+1];
        bp[cnt+1] = bp[cnt];
        bp[cnt] = tmpbytelsb;
    }

    return 0;
}
```

写函数

```
int MLX90640_I2CWrite(uint8_t slaveAddr, uint16_t writeAddress, uint16_t data)
{
    uint8_t sa;
    int ack = 0;
    uint8_t cmd[2];
    static uint16_t dataCheck;

    sa = (slaveAddr << 1);

    cmd[0] = data >> 8;
    cmd[1] = data & 0x00FF;

    ack = HAL_I2C_Mem_Write(&hi2c2, sa, writeAddress, I2C_MEMADD_SIZE_16BIT, cmd, sizeof(cmd), 500);

    if (ack != HAL_OK)
    {
        return -1;
    }

    MLX90640_I2CRead(slaveAddr, writeAddress, 1, &dataCheck);

    if (dataCheck != data)
    {
        return -2;
    }

    return 0;
}
```

接下来我们就可以通过 API 来进行数据读取了，按照前面讲的测量流程，上电复位后我们首先要读取 EEPROM 中的参数：

```
paramsMLX90640 mlx90640;
status = MLX90640_DumpEE(MLX90640_ADDR, eeMLX90640);
if (status != 0) printf("\r\nload system parameters error with code:%d\r\n", status);
status = MLX90640_ExtractParameters(eeMLX90640, &mlx90640);
if (status != 0) printf("\r\nParameter extraction failed with error code:%d\r\n", status);
```

然后我们可以设置一下刷新速率：

```
MLX90640_SetRefreshRate(MLX90640_ADDR, RefreshRate);
```

接下来我们就可以读取传感器的温度值了，刚开始可能会有一点不准，一般 30S 内数据就会稳定下来：

```
for(int i = 0; i < 2; i++){
    int status = MLX90640_GetFrameData(MLX90640_ADDR, frame);
    if (status < 0)
    {
        printf("GetFrame Error: %d\r\n", status);
    }
    float vdd = MLX90640_GetVdd(frame, &mlx90640);
    float Ta = MLX90640_GetTa(frame, &mlx90640);

    float tr = Ta - TA_SHIFT; //Reflected temperature based on the sensor ambient temperature
    MLX90640_CalculateTo(frame, &mlx90640, emissivity, tr, mlx90640To);
}
```

MLX90640_GetFrameData() 是获取一帧数据，此时还不是我们直观上的温度值，还需要通过计算来获得真正的温度值，计算的时候有几个参数是需要注意，首先是发射率(emissivity)，这是物体的一种属性，根据材料的不同而不同，一般非金属（如塑料、油漆、皮革、纸张等）发射率通常为 0.95，常见物体发射率可以参考下图：

材料名称	规格	发射率	材料名称	规格	发射率
铝	氧化	0.20-0.40	人体皮肤		0.98
	抛光	0.02-0.04	石墨	氧化	0.20-0.60
铜	氧化	0.40-0.80	塑胶	透明度 >0.5mm	0.95
	抛光	0.02-0.05			
黄金		0.01-0.10	橡胶		0.95
铁	氧化	0.60-0.09	塑胶		0.85-0.95
钢	氧化	0.70-0.90	混凝土		0.95
石棉		0.95	水泥		0.96
石膏		0.80-0.90	土壤		0.90-0.98
沥青		0.95	灰泥		0.89-0.91
陶器		0.95	砖		0.93-0.96
木材		0.90-0.95	大理石		0.94
木炭	粉末	0.96	纺织品	各种	0.90
漆器		0.80-0.95	纸	颜色	0.94
漆器	无光泽	0.97			
碳胶		0.90	沙子		0.90
肥皂泡		0.75-0.80	泥土		0.92-0.96
水		0.93	沙砾	餐具	0.95
雪		0.83-0.90	玻璃	规格	0.85-0.92
冰		0.96-0.98	纺织品		0.95

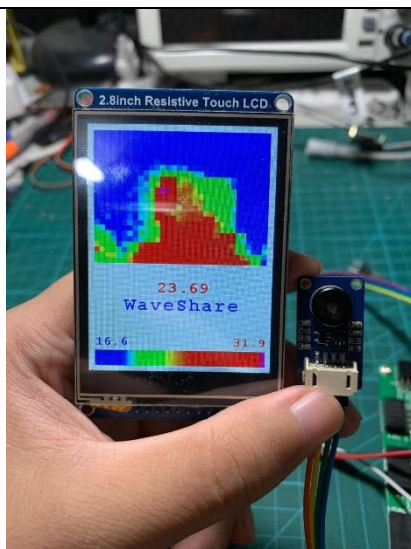
另外一个就是 tr (反射温度基于传感器环境温度) 的值这个参数是通过 $T_a - TA_SHIFT$ 计算出来的, 其中 TA_SHIFT 默认值为 8, `MLX90640_CalculateTo()` 就是将获得的一帧数据与这几个值进行运算从而得到我们所理解的温度值。计算完成后我们就可以通过串口打印查看具体的温度值了, 如下图, 可以将 768 个像素的温度值全部打印出来, 经过计算的温度值储存在 `mlx90640To` 数组中:

```
printf("\r\n=====WaveShare=====\\r\\n");
for(int i = 0; i < 768; i++){
    if(i%32 == 0 && i != 0){
        printf("\\r\\n");
    }
    printf("%2.2f ",mlx90640To[i]);
}
printf("\\r\\n=====WaveShare=====\\r\\n");
```

至此, 我们就成功移植了官方的驱动到我们的工程中, 更多例程请查看示例程序的其他部分。

比如:

① STM32+LCD 图像显示:



② Arduino+ESP32+LCD 图像显示:



③ 树莓派 HDMI 显示屏显示:

