C1028 Thermal Camera Module With Uart Interface

User Manual

Release Note:
Jan 05, 2023 – official released v1.0

COMedia Ltd.
康大科技有限公司

Rm 802, Nan Fung Ctr, Castle Peak Rd, Tsuen Wan NT, Hong Kong
Tel: (852) 2498 6248 Fax (852) 2414 3050
Email: sales@comedia.com.hk
https://www.comedia.com.hk

General Descriptions

C1028 is thermal camera module that can be attached to a PC or IoT host. Users can send out a capture command from the host in order to capture and get an array of temperatures and transfer to the host thru serial port.

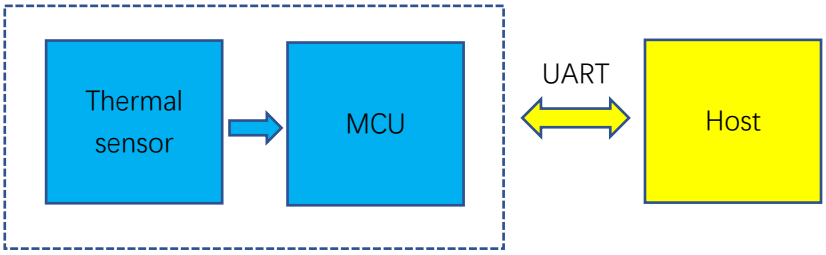


Figure 1 – System block diagram

Features

- Small in size, 25mm×40mm
- 80*62 resolution
- 5V operation
- Low power consumption
- User friendly commands to control the module
- UART interface of up to 921.6Kbps

Board Layout

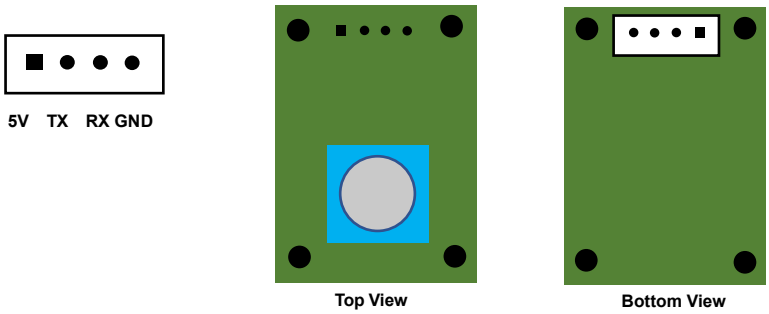


Figure 2 – C1028 board layout and serial interface pin assignment

Pin	VCC	TX	RX	GND
Description	Power 5V DC	Data transmit (3.3V)	Data receive (3.3V)	Power Ground

Serial Interface

1.baud rate

C1028 supports total 8 types of baud rate: **921600** bps, **750000** bps,**512000** bps, **460800** bps,**256000** bps, **115200** bps, **57600** bps, and **19200** bps. Default baud rate is **115200** bps. In other words, host needs to sync with module using **115200** bps when power up. After connection, host can change baud rate to other value.

2.Single Byte Timing Diagram

A single byte RS-232 transmission consists of the start bit, 8-bit contents and the stop bit. A start bit is always 0, while a stop bit is always 1. LSB is sent out first and is right after the start bit.

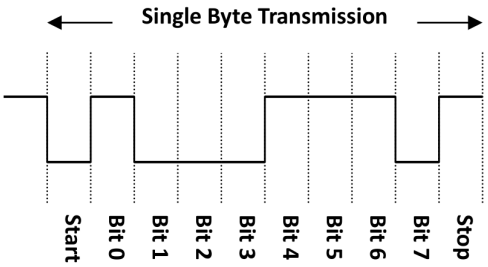


Figure 3 – RS-232 single byte timing diagram

3. Command Timing Diagram

A single command consists of 5 continuous single byte RS-232 transmissions. The following is an example of SYNC (AA 0D 00 00 00) command.

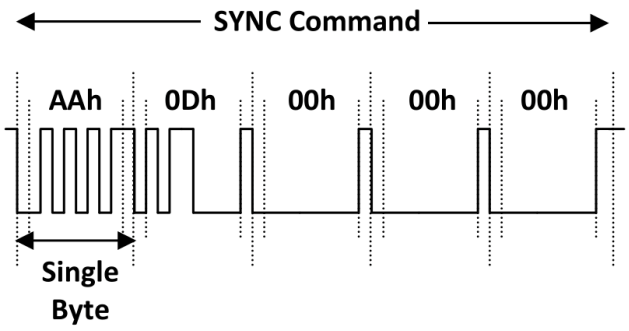


Figure 4 – RS-232 SYNC command timing diagram

Command Summary

Command	Function
Snapshot	Capture current frame and save data in buffer
Get Package	Get data from buffer
Reset Sensor	Reset sensor
Change baud rate	Change uart baud rate
Sleep	Sleep sensor or the whole module Parameter1 = 0x00 only sensor in sleep mode, the current is 30-40mA Parameter1 = 0x01 the whole module in sleep mode, the current is less than 1mA
Set Filter	Enable/disable filter Parameter1 = 0 disable filter Parameter1 = 1 enable filter
SYNC	Check uart if function is available
ACK	Module to host only
NAK	Module to host only

Command Set

Command	ID	Parameter1	Parameter2	Parameter3
Snapshot	0x01	0x00	0x00	0x00
Get Package	0x02	Package ID	0x00	0x00
Reset Sensor	0x04	0x00	0x00	0x00
Change baud rate	0x05	baud rate	0x00	0x00
Sleep	0x06	Sleep mode	0x00	0x00
Set Filter	0x07	Disable/enable	0x00	0x00
SYNC	0x0D	0x00	0x00	0x00
ACK	0x0E	Command ID	0x00	0x00
NAK	0x0F	Command ID	Error Type	0x00

1. Snapshot

Users can snapshot one frame of data by sending this command. After the snapshot, the data is saved in a temp buff, users should get the data by sending "Get Package" command.

2. Get Package

After snapshot, users can get data from module by sending this command.

One frame data is divided into 10 packages. Users can change Parameter1 to get different package, the range is from 0x00 to 0x09

02 00 00 00 00 get first package

02 01 00 00 00 get second package

….

02 09 00 00 00 get last package

One package contains 1024 Bytes. (Byte 0 send first and Byte 1023 send last)

Byte0	Byte1	Byte2-1009	Byte1010-1019	Byte1020-1023
Package ID	~Package ID	DATA	0X1A	CRC32

Byte0: the current ID of package, range from 0x00 to 0x09.

Byte1: ~ (Package ID)

Byte2-1009: the data

Byte1010-1019: fixed to 0X1A

Byte1020-1023: CRC32 of Byte0-By1019

Put all data together, user will get total 5040 words.

Header:80 words

Temperature data:4960 words

Frame Header	Frame counter 1 word	SenXor VDD 1 word	SenXor die temperature 1 word	Time stamp 2 words	Max pixel value 1 word	Min pixel value 1 word	CRC 1 word	Reserved 72 words
Tempe-rature data	80 column * 62 row Pixel Data, i.e. 4960 words							

Every 2 Bytes represents the temperature of a pixel,
as a 16-bit unsigned integer in units of 0.1 K
Example: **0X0C 0X75**

0x0C75 = 3189, so T = 318.9K = (318.9-273.1) ℃ = 45.8 ℃

3. Reset

Users can reset the thermal camera, after resetting it, users should wait at least 2 seconds to re-communicate with the module.

4. Change baud rate

C1028 supports multiple baud rates, it's set to 115200 bps by default. Users can change that value.

Parameter1	Baud rate
0x00	921600
0x01	750000
0x02	512000
0x03	460800
0x04	256000
0x05	**115200 (default)**
0x06	57600
0x07	19200

5. Sleep

There are 2 sleep mode.

Mode 0: only thermal camera enters sleep mode. In this mode, the current is reduced by half.

Mode 1: whole system enters sleep mode. In this mode, the current will be reduced to less than 1mA. It's suitable for battery operation.

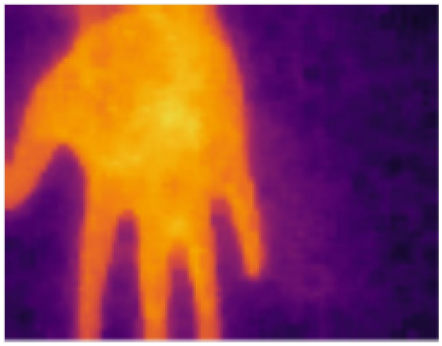
Wake up

To wake up module, users just send sync command.

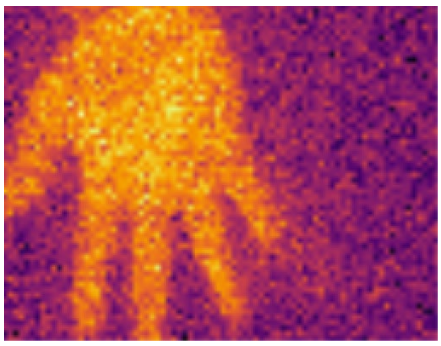
In mode 0, the module will wake up immediately. Less than 500ms from wake-up to data ready.

In mode 1, the whole system will be reset, it will take 1-2s when data is ready.

6. Set Filter



Filter enabled



Filter disabled

When filter is enabled, image will be smoother. By default, it is disabled. After setting this command, C1028 will reinitialize sensor. Please wait 1-2 seconds before doing any other operation.

7. SYNC
The module will response AA 0E 0D 00 00

8. ACK
This command is a handshake command, which means command is received and processed correctly.

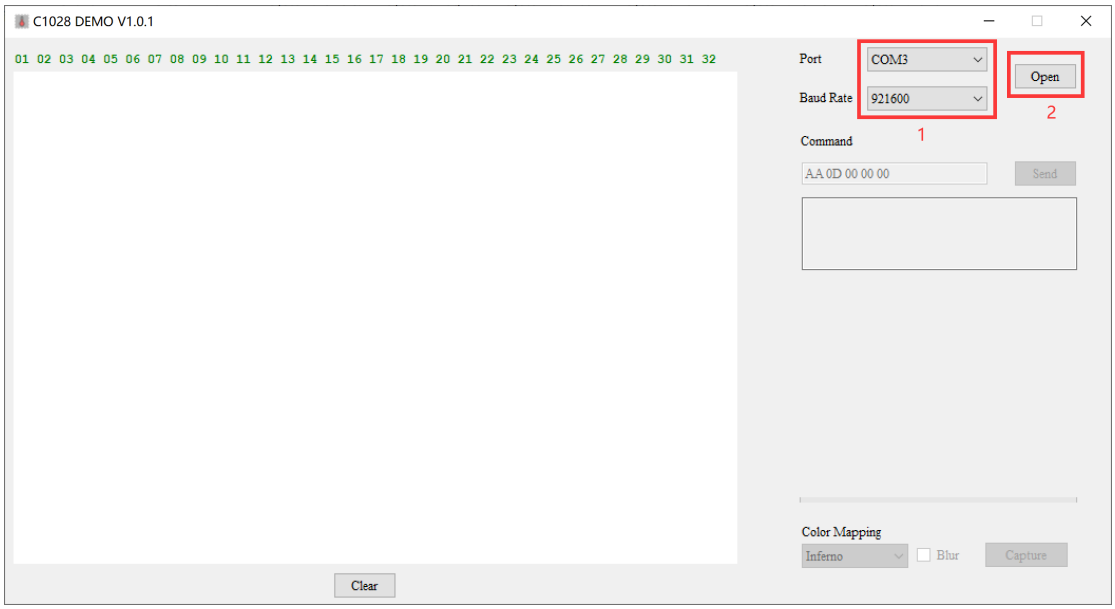
9. NAK
This command is a handshake command, which indicates different type of errors.

0xAA, 0x0F, 0x01, 0x00, 0x00	INVALID_COMMAND_LENGTH
0xAA, 0x0F, 0x02, 0x00, 0x00	INVALID_COMMAND_FORMAT
0xAA, 0x0F, 0x03, 0x00, 0x00	INVALID_COMMAND_ID
0xAA, 0x0F, 0x04, 0x00, 0x00	INVALID_COMMAND_PARAMETER
0xAA, 0x0F, 0x05, 0x00, 0x00	DATA_NOT_READY
0xAA, 0x0F, 0x06, 0x00, 0x00	GET_SPI_DATA_FAIL
0xAA, 0x0F, 0x07, 0x00, 0x00	I2C_WRITE_FAIL
0xAA, 0x0F, 0x08, 0x00, 0x00	I2C_READ_FAIL
0xAA, 0x0F, 0x09, 0x00, 0x00	SET_COMMAND_FAIL
0xAA, 0x0F, 0x0a, 0x00, 0x00	IN_SLEEP_MODE

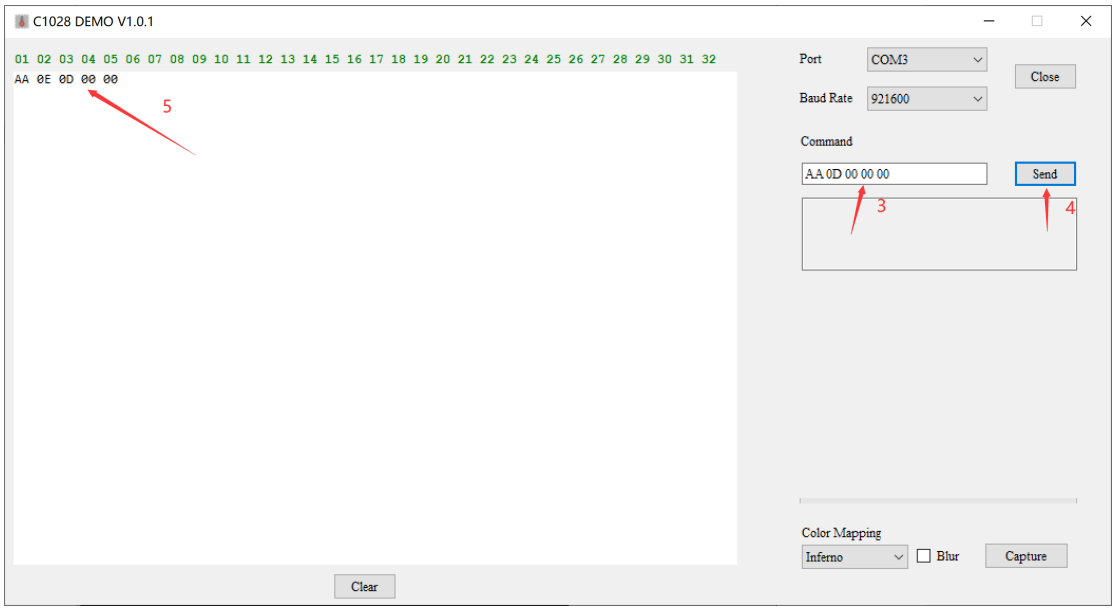
PC Demonstration Tool

We provide a computer tool to help users getting started quickly.

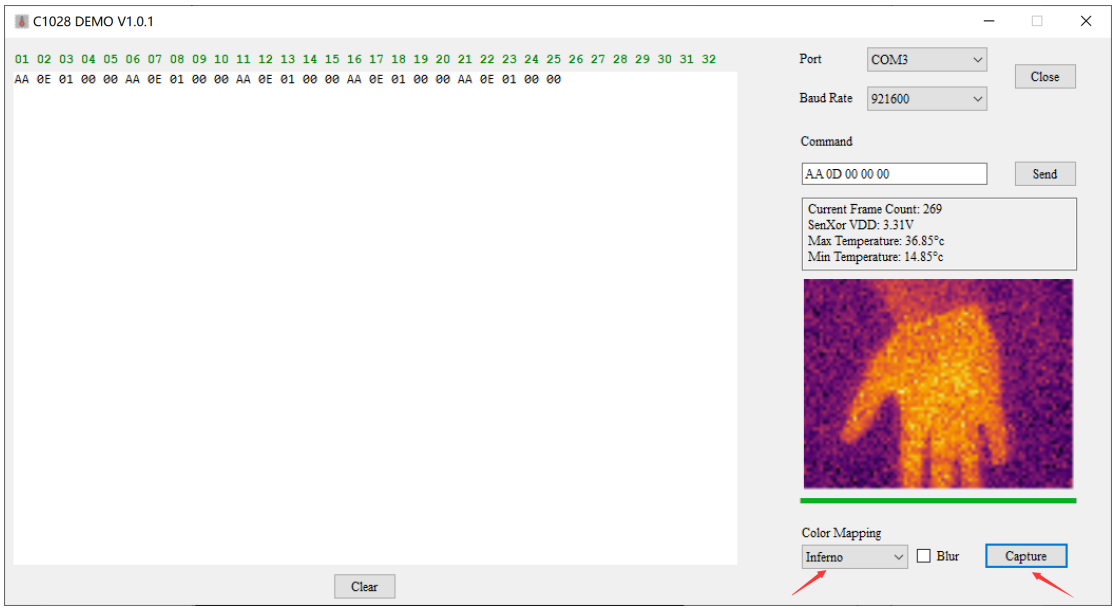
1.Open tool, select the current port and baud rate, then click **Open** button.



2. Try different commands, press Send button, the left window will display the message returned from the serial port



3.Capture one frame, select a color mapping then press **Capture** button. The tool will capture and fetch 10 packets from the module, parse the data, and then display the image on top according to the colors in the color mapping.



Appendix A: CRC32 implement of C

```
const uint32_t crc32_tab[] = { 0x00000000, 0x77073096, 0xee0e612c, 0x990951ba,
        0x076dc419, 0x706af48f, 0xe963a535, 0x9e6495a3, 0x0edb8832, 0x79dcb8a4,
        0xe0d5e91e, 0x97d2d988, 0x09b64c2b, 0x7eb17cbd, 0xe7b82d07, 0x90bf1d91,
        0x1db71064, 0x6ab020f2, 0xf3b97148, 0x84be41de, 0x1adad47d, 0x6ddde4eb,
        0xf4d4b551, 0x83d385c7, 0x136c9856, 0x646ba8c0, 0xfd62f97a, 0x8a65c9ec,
        0x14015c4f, 0x63066cd9, 0xfa0f3d63, 0x8d080df5, 0x3b6e20c8, 0x4c69105e,
        0xd56041e4, 0xa2677172, 0x3c03e4d1, 0x4b04d447, 0xd20d85fd, 0xa50ab56b,
        0x35b5a8fa, 0x42b2986c, 0xdbbbc9d6, 0xacbcf940, 0x32d86ce3, 0x45df5c75,
        0xdcd60dcf, 0xabd13d59, 0x26d930ac, 0x51de003a, 0xc8d75180, 0xbfd06116,
        0x21b4f4b5, 0x56b3c423, 0xcfba9599, 0xb8bda50f, 0x2802b89e, 0x5f058808,
        0xc60cd9b2, 0xb10be924, 0x2f6f7c87, 0x58684c11, 0xc1611dab, 0xb6662d3d,
        0x76dc4190, 0x01db7106, 0x98d220bc, 0xefd5102a, 0x71b18589, 0x06b6b51f,
        0x9fbfe4a5, 0xe8b8d433, 0x7807c9a2, 0x0f00f934, 0x9609a88e, 0xe10e9818,
        0x7f6a0dbb, 0x086d3d2d, 0x91646c97, 0xe6635c01, 0x6b6b51f4, 0x1c6c6162,
        0x856530d8, 0xf262004e, 0x6c0695ed, 0x1b01a57b, 0x8208f4c1, 0xf50fc457,
        0x65b0d9c6, 0x12b7e950, 0x8bbeb8ea, 0xfcb9887c, 0x62dd1ddf, 0x15da2d49,
        0x8cd37cf3, 0xfbd44c65, 0x4db26158, 0x3ab551ce, 0xa3bc0074, 0xd4bb30e2,
        0x4adfa541, 0x3dd895d7, 0xa4d1c46d, 0xd3d6f4fb, 0x4369e96a, 0x346ed9fc,
        0xad678846, 0xda60b8d0, 0x44042d73, 0x33031de5, 0xaa0a4c5f, 0xdd0d7cc9,
        0x5005713c, 0x270241aa, 0xbe0b1010, 0xc90c2086, 0x5768b525, 0x206f85b3,
        0xb966d409, 0xce61e49f, 0x5edef90e, 0x29d9c998, 0xb0d09822, 0xc7d7a8b4,
        0x59b33d17, 0x2eb40d81, 0xb7bd5c3b, 0xc0ba6cad, 0xedb88320, 0x9abfb3b6,
        0x03b6e20c, 0x74b1d29a, 0xead54739, 0x9dd277af, 0x04db2615, 0x73dc1683,
        0xe3630b12, 0x94643b84, 0x0d6d6a3e, 0x7a6a5aa8, 0xe40ecf0b, 0x9309ff9d,
        0x0a00ae27, 0x7d079eb1, 0xf00f9344, 0x8708a3d2, 0x1e01f268, 0x6906c2fe,
        0xf762575d, 0x806567cb, 0x196c3671, 0x6e6b06e7, 0xfed41b76, 0x89d32be0,
        0x10da7a5a, 0x67dd4acc, 0xf9b9df6f, 0x8ebeeff9, 0x17b7be43, 0x60b08ed5,
        0xd6d6a3e8, 0xa1d1937e, 0x38d8c2c4, 0x4fdff252, 0xd1bb67f1, 0xa6bc5767,
        0x3fb506dd, 0x48b2364b, 0xd80d2bda, 0xaf0a1b4c, 0x36034af6, 0x41047a60,
        0xdf60efc3, 0xa867df55, 0x316e8eef, 0x4669be79, 0xcb61b38c, 0xbc66831a,
        0x256fd2a0, 0x5268e236, 0xcc0c7795, 0xbb0b4703, 0x220216b9, 0x5505262f,
        0xc5ba3bbe, 0xb2bd0b28, 0x2bb45a92, 0x5cb36a04, 0xc2d7ffa7, 0xb5d0cf31,
        0x2cd99e8b, 0x5bdeae1d, 0x9b64c2b0, 0xec63f226, 0x756aa39c, 0x026d930a,
        0x9c0906a9, 0xeb0e363f, 0x72076785, 0x05005713, 0x95bf4a82, 0xe2b87a14,
        0x7bb12bae, 0x0cb61b38, 0x92d28e9b, 0xe5d5be0d, 0x7cdcefb7, 0x0bdbdf21,
        0x86d3d2d4, 0xf1d4e242, 0x68ddb3f8, 0x1fda836e, 0x81be16cd, 0xf6b9265b,
        0x6fb077e1, 0x18b74777, 0x88085ae6, 0xff0f6a70, 0x66063bca, 0x11010b5c,
        0x8f659eff, 0xf862ae69, 0x616bffd3, 0x166ccf45, 0xa00ae278, 0xd70dd2ee,
        0x4e048354, 0x3903b3c2, 0xa7672661, 0xd06016f7, 0x4969474d, 0x3e6e77db,
        0xaed16a4a, 0xd9d65adc, 0x40df0b66, 0x37d83bf0, 0xa9bcae53, 0xdebb9ec5,
        0x47b2cf7f, 0x30b5ffe9, 0xbdbdf21c, 0xcabac28a, 0x53b39330, 0x24b4a3a6,
        0xbad03605, 0xcdd70693, 0x54de5729, 0x23d967bf, 0xb3667a2e, 0xc4614ab8,
        0x5d681b02, 0x2a6f2b94, 0xb40bbe37, 0xc30c8ea1, 0x5a05df1b, 0x2d02ef8d };


uint32_t crc32(const void *buf, size_t size) {
    const uint8_t *p = buf;
    uint32_t crc;
    crc = ~0U;
    while (size--)
        crc = crc32_tab[(crc ^ *p++) & 0xFF] ^ (crc >> 8);
    return crc ^ ~0U;
}
```