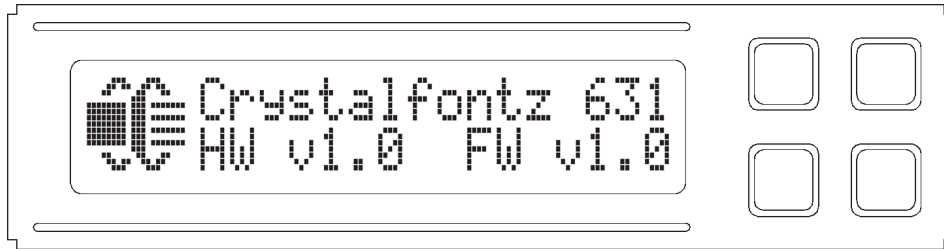


CrystalFontz America, Incorporated

USB LCD MODULE SPECIFICATIONS



CrystalFontz Model Number	CFA631-TMF-KU
Hardware Version	v2.0 August 2005
Firmware Version	v2.0 August 2005
Data Sheet Version	v2.0a December 2005
Product Pages	www.crystalfontz.com/products/631

Customer Name	
Customer Part Number	

CrystalFontz America, Incorporated

12412 East Saltese Avenue
Spokane Valley, WA 99216-0357

Phone: (888) 206-9720

Fax: (509) 892-1203

Email: techinfo@crystalfontz.com

URL: www.crystalfontz.com



REVISION HISTORY

HARDWARE	
2005/08/01	Start Public Version Tracking. Current hardware version: v2.0

FIRMWARE	
2005/08/01	Start Public Version Tracking. Current firmware version: v2.0 Command 1: Get Hardware & Firmware Version (Pg. 15) returns: "CFA631:h2.0,v2.0"

DATA SHEET	
2005/08/01	Start Public Version Tracking. Data Sheet version: v2.0 Changes since last released version (v1.9): Added Revision History (this page). Added GPIO Current Limits (Pg. 8) . Added APPENDIX C: CALCULATING THE CRC (Pg. 47) . Added note on operating system delays (Pg. 13) . Added note on length of command 30 reply (Pg. 30) . Added documentation for commands requiring the CrystalFontz SCAB accessory. Corrected length returned by command 30 (Pg. 30) .
2005/12/20	Current Data Sheet version: v2.0a Changes since last released version (v2.0): Corrected " Character Size " and added " Character Pitch " (Pg. 7). Corrected specification for supply voltage maximum (Pg. 8) . Corrected return "type" for command 26: Set Fan Tachometer Glitch Filter (SCAB required) (Pg. 26) . Corrected return "type" for command 27: Query Fan Power & Fail-Safe Mask (SCAB required) (Pg. 27) . Corrected "type" for command 33: Set Baud Rate (Pg. 32) . Corrected length returned by reply for command 35: Read GPIO Pin Levels and Configuration State (Pg. 34) . Formatting, content organization, and minor rewording to improve readability. For added convenience, a separate data sheet is available for each CFA-631 module variant.



CONTENTS, CONTINUED

26: Set Fan Tachometer Glitch Filter (SCAB required)-	26
27: Query Fan Power & Fail-Safe Mask (SCAB required) -	27
28: Set ATX Power Switch Functionality (SCAB required) -	28
29: Enable/Disable and Reset the Watchdog (SCAB required)-	29
30: Read Reporting & Status-	30
31: Send Data to LCD -	31
32: Key Legends -	31
33: Set Baud Rate -	32
34: Set or Set and Configure GPIO Pin -	32
35: Read GPIO Pin Levels and Configuration State-	34
CHARACTER GENERATOR ROM (CGROM) -	36
CFA-631 MODULE OUTLINE DRAWING -	37
JUMPER LOCATIONS AND FUNCTIONS -	38
CARE AND HANDLING PRECAUTIONS -	39
APPENDIX A:	
CONNECTING A DS2450 1-WIRE QUAD A/D CONVERTER (SCAB REQUIRED)-	41
APPENDIX B:	
CONNECTING A DS1963S SHA IButton (SCAB REQUIRED) -	43
APPENDIX C:	
CALCULATING THE CRC -	47
Algorithm 1: "C" Table Implementation -	47
Algorithm 2: "C" Bit Shift Implementation -	48
Algorithm 3: "PIC Assembly" Bit Shift Implementation -	49
Algorithm 4: "Visual Basic" Table Implementation -	51
Algorithm 5: "Java" Table Implementation -	52
Algorithm 6: "Perl" Table Implementation -	53
APPENDIX D:	
QUALITY ASSURANCE STANDARDS-	55

LIST OF FIGURES

Figure 1. CFA631-TMF-KU System Block Diagram -	6
Figure 2. CFA631-TMF-KU USB Host Connection-	9
Figure 3. CFA631-TMF-KU has five GPIO connections on header "H1" -	10
Figure 4. CFA-631 connected to a CrystalFontz SCAB using the WREXTY19 cable -	11
Figure 5. Character Generator ROM (CGROM) -	36
Figure 6. CFA-631 Module Outline Drawing (v2.0 identical to v1.0)-	37
Figure 7. CFA-631 Hardware v2.0 Jumper Locations and Functions -	38
Appendix A Figure 1. CFA-631 Test Circuit Schematic -	41
Appendix B Figure 1. Connect CFA-631 to Maxim/Dallas DS19632 SHA iButton using DS9094 iButton Clip 44	



FEATURES

- 20x2 LCD has compact size: 3½ inch floppy drive form factor and easily fits in 1U cases.
- USB interface (115200 baud equivalent throughput).
- Integrated LED backlit 4-button translucent silicon keypad.
- White edge LED backlit with STN-blue negative mode LCD (displays light characters on blue background).
- Blue LED backlit keypad.
- Key legends allow assignment of keys to be shown easily on the LCD.
- LCD characters are contiguous in both X and Y directions to allow the host software to display “gapless” bar graphs in horizontal or vertical directions.
- Fully decoded keypad: any key combination is valid and unique.
- Robust packet-based communications protocol with 16-bit CRC.
- Built-in reprogrammable microcontroller (factory operation).
- Nonvolatile memory capability (EEPROM):
 - Customize the “power-on” display settings.
 - 16-byte “scratch” register for storing IP address, netmask, system serial number . . .
- Expandable firmware and configurable hardware can be customized to add specific features for your system needs (tooling fee and minimum order may apply).
 - Other additional analog or digital I/O devices.
 - Provide “dongle” functionality for software copy protection.
 - Autonomous hardware monitoring.
- Firmware support for the CrystalFontz accessory System Cooling Accessory Board (SCAB) For more information on the SCAB accessory see [CFA-SCAB](#) Data Sheet. The combination of the CFA631-TMF-KU with the SCAB (written as “CFA631-TMF-KU+SCAB” in this data sheet) allows:
 - ATX power supply control functionality allows the buttons on the CFA631-TMF-KU to replace the “power” and “reset” switches on your system, simplifying front panel design.
 - Four fan connectors with RPM monitoring and variable PWM fan power control.
 - Fail-safe fan power settings allows safe host fan control based on temperature.
 - Temperature monitoring: up to 32 channels at up to 0.5 degrees C absolute accuracy (using optional CrystalFontz [WRDOWY17](#) cable with Dallas 1-Wire sensor).
 - Hardware watchdog can reset host on host software failure.
 - “Live Display” shows up to four temperature or fan readings without host intervention, allowing fans and temperatures to be shown immediately at boot, even before the host operating system is loaded.
 - RS-232 to Dallas Semiconductor 1-Wire bridge functionality allows control of other 1-Wire compatible devices (ADC, voltage monitoring, current monitoring, RTC, GPIO, counters, identification/encryption). (Additional hardware required).



ORDERING INFORMATION

PART NUMBER	FLUID	LCD GLASS COLOR	IMAGE	POLARIZER FILM	BACKLIGHTS
CFA631-TMF-KU	STN	blue	negative	transmissive	LCD: white edge LEDs Keypad: blue LEDs

SYSTEM BLOCK DIAGRAM

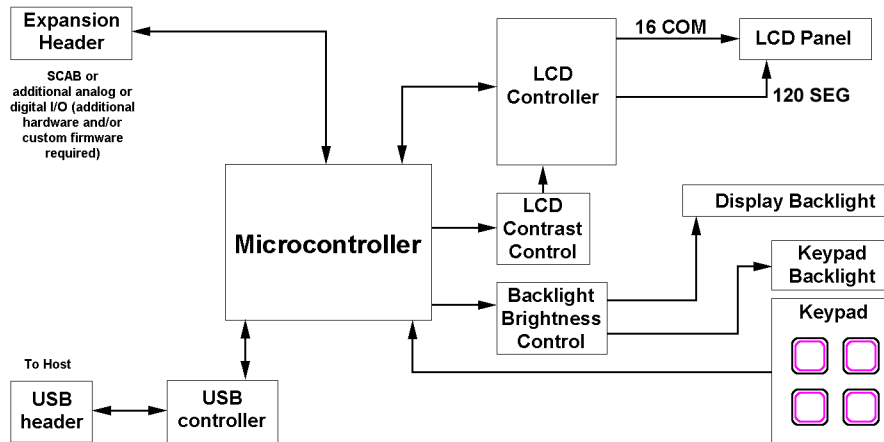


Figure 1. CFA631-TMF-KU System Block Diagram



PHYSICAL CHARACTERISTICS

ITEM	SIZE (mm)
Module Dimensions	101.6 (W) x 25.4 (H)
Viewing Area	66.0 (W) x 13.8 (H)
Active Area	63.55 (W) x 10.35 (H)
Character Size	2.6 (W) x 4.5 (H)
Character Pitch	3.18 (W) x 5.2 (H)
Dot Size	0.48 (W) x 0.60 (H)
Dot Pitch	0.53 (W) x 0.65 (H)
Depth:	
Without Keypad or Overlay	90.0
Without Keypad, with Overlay	90.6
With Keypad	93.1
Keystroke Travel (approximate)	2
Weight	80 grams (typical)

TEMPERATURE RANGE

CRITERIA	SPECIFICATION
Operating Temperature Range	0°C minimum to +50°C maximum
Storage Temperature Range	-10°C minimum to +60°C maximum

OPTICAL CHARACTERISTICS

Viewing Direction	12 o'clock
-------------------	------------



ELECTRICAL SPECIFICATIONS

DRIVING METHOD	SPECIFICATION
Duty	1/32
Bias	6.7

SUPPLY VOLTAGE	MINIMUM	NOMINAL	MAXIMUM
Supply voltage for driving the LCD module (normally supplied through USB connection)	+4.75v	+5.0v	+5.25v

TYPICAL CURRENT CONSUMPTION ($V_{DD} = +5v$)	BACKLIGHT ONLY	INCLUDING LOGIC
Logic + USB controller, backlight off	30 mA	
Logic + USB controller, backlight at 100%	60 mA	90 mA

GPIO CURRENT LIMITS	SPECIFICATION
Sink	25 mA
Source	10 mA

ADDITIONAL CRITERIA	SPECIFICATION
Backlight PWM Frequency	320 Hz nominal
Fan Tachometer Speed Range* (assuming 2 PPR) [†]	600 RPM to 3,000,000 RPM
Fan Power Control PWM Frequency*	18 Hz nominal

* When used with SCAB.

[†] PPR is "pulses per revolution", also written as "p/r".



ESD (Electro-Static Discharge) Specifications:

D+ and D- pins of USB connector only: Electrostatic Discharge Voltage ($I < 1 \mu A$): +/- 2000 V

The remainder of the circuitry is industry standard CMOS logic and susceptible to ESD damage. Please use industry standard antistatic precautions as you would for any other PCB such as expansion cards or motherboards. For more information, read [CARE AND HANDLING PRECAUTIONS \(Pg. 39\)](#).

RELIABILITY

ITEM	SPECIFICATION	
LCD portion (excluding Keypad and Backlights)	50,000 to 100,000 hours (typical)	
Keypad	1,000,000 keystrokes	
White* LED Display and Blue LED Keypad Backlights <i>* We recommend that the backlight of the white LED backlit modules be dimmed or turned off during periods of inactivity to conserve the white LED backlight lifetime.</i>	<i>Power-On Hours</i>	<i>% of Initial Brightness</i>
	<10,000	>90%
	<50,000	>50%

USB HOST CONNECTION

The CFA631-TMF-KU is a USB peripheral, requiring only one connection to the host for both data communications and power supply.

The CFA631-TMF-KU uses a low profile 2 mm latching polarized connector for USB connection. CrystalFontz offers two cables that make the connection between the CFA631-TMF-KU and the host. The [WRUSBY03](#) has the mating 2mm connector on one end and a standard "USB A" on the other end. The [WRUSBY11](#) has the mating 2 mm connector on one end and standard single pin connectors on the opposite end. These single pin connectors are suitable to plug directly onto the USB headers typically found on motherboards.

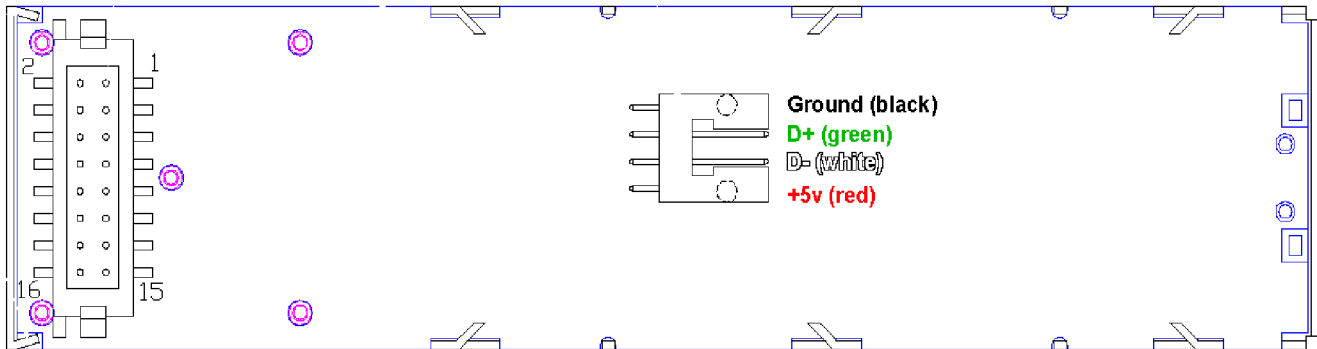


Figure 2. CFA631-TMF-KU USB Host Connection



If you would like to make your own cable, the connector on the CFA631-TMF-KU is:
 FCI/Berg 95000-004: SMT 2 mm connector, 4-position, polarized

The mating housing and terminals for the cable are:
 FCI/Berg 90312-004: Housing, 2 mm connector, 4-position, polarized
 FCI/Berg 77138-001: Terminal (4 pieces required)

GPIO/GPO CONNECTIONS

CFA631-TMF-KU has 5 GPIOs available on header "H1". These GPIOs can be accessed directly through "H1" or through the CrystalFontz SCAB (System Cooling Accessory Board) connected to "H1". The SCAB can be easily connected to the CFA631-TMF-KU by using either the CrystalFontz [WREXTY15](#) or [WREXTY19](#) cables.

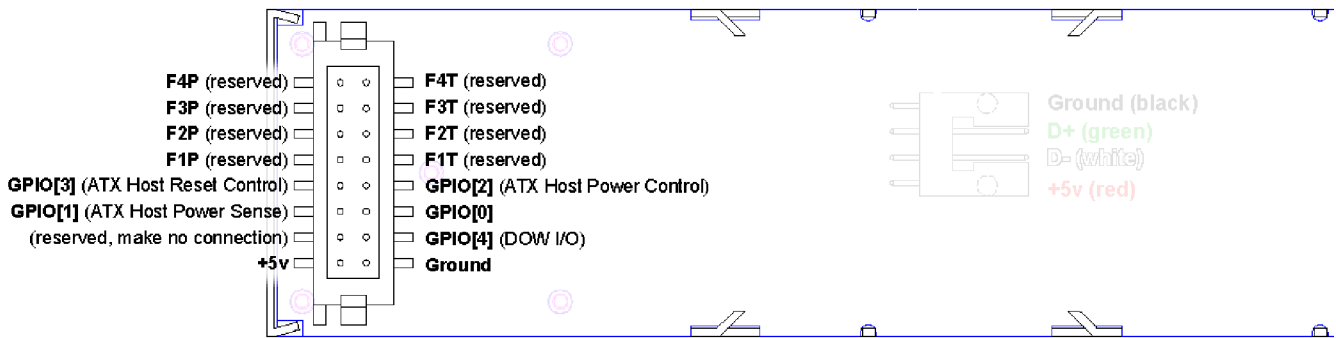


Figure 3. CFA631-TMF-KU has five GPIO connections on header "H1"

Please see the commands [34: Set or Set and Configure GPIO Pin \(Pg. 32\)](#), and [35: Read GPIO Pin Levels and Configuration State \(Pg. 34\)](#) below for details on how to control the GPIOs.

The following parts may be used to make a mating cable for H1:

- 16-position housing: Hirose DF11-16DS-2C / [Digi-Key H2025-ND](#)
- Terminal (tape & reel): Hirose DF11-2428SCF / [Digi-Key H1504TR-ND](#)
- Terminal (loose): Hirose DF11-2428SC / [Digi-Key H1504-ND](#)
- Pre-terminated interconnect wire: Hirose / [Digi-Key H3BBT-10112-B4-ND](#) is typical



SCAB CONNECTION

The Crystalfontz SCAB is designed to connect to a CFA-631's "H1" header. The SCAB will receive the correct signals to operate from the CFA-631.

Here is a photo showing the CFA-631 connected to a SCAB using the [WREXTY19](#) cable:

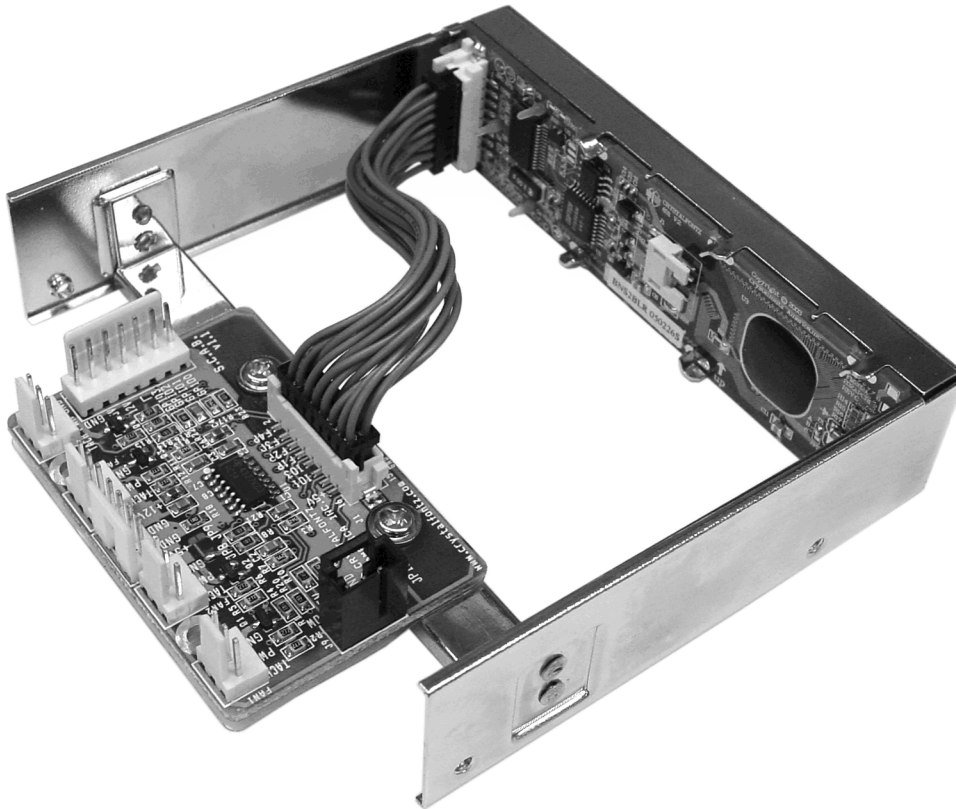


Figure 4. CFA-631 connected to a Crystalfontz SCAB using the WREXTY19 cable

There are two cables available from Crystalfontz to make the connection between the SCAB and the CFA-631:

- [WREXTY15](#): 18" SCAB connection cable
- [WREXTY19](#): 3.5" SCAB connection cable

The [WREXTY15](#) allows the SCAB to be mounted some distance away from the CFA-631. For instance, the SCAB could be mounted in a central location within the PC's case. The WREXTY15 would connect from this central location to the LCD module that is mounted in a drive bay. Then the connections to the fans and temperature sensors would only need to be run to the SCAB, not all the way to the front panel where the LCD module is mounted.

The [WREXTY19](#) is intended to be used when the SCAB is mounted in close proximity to the CFA-631 as is the case when the SCAB is fastened directly to the LCD module's mounting bracket.



automatically re-synchronize to the next valid packet in the event of any communications errors. Please follow the algorithm in the sample code closely in order to realize the benefits of using the packet communications.

ABOUT HANDSHAKING

The nature of CFA-631's packets makes it unnecessary to implement traditional hardware or software handshaking.

The host should wait for a corresponding acknowledge packet from the CFA-631 before sending the next command packet. The CFA-631 will respond to all packets within 250 mS. The host software should stop waiting and retry the packet if the CFA-631 fails to respond within 250 mS. The host software should report an error if a packet is not acknowledged after several retries. This situation indicates a hardware problem—for example, a disconnected cable. Please note that some operating systems may introduce delays between when the data arrives at the physical port from the CFA-631 until it is available to the user program. In this case, the host program may have to increase its timeout window to account for the additional overhead of the operating system.

The CFA-631 can be configured to send several types of report packets along with regular acknowledge packets. The host should be able to buffer several incoming packets and must guarantee that it can process and remove packets from its input buffer faster than the packets can arrive given the 115200 equivalent baud rate of the VCP and the reporting configuration of the CFA-631. For any modern PC using reasonably efficient software, this requirement will not pose a challenge.

The report packets are sent asynchronously with respect to the command packets received from the host. The host should not assume that the first packet received after it sends a command is the acknowledge packet for that command. The host should inspect the `type` field of incoming packets and process them accordingly.

REPORT CODES

The CFA-631 can be configured to report three items. The CFA-631 sends reports automatically when the data becomes available. Reports are not sent in response to a particular packet received from the host. The three report types are:

0x80: Key Activity

If a key is pressed or released, the CFA-631 sends a Key Activity report packet to the host. Key event reporting may be individually enabled or disabled by command [23: Configure Key Reporting \(Pg. 25\)](#).

```
type = 0x80
data_length = 1
data[0] is the type of keyboard activity:
KEY_UL_PRESS    13
KEY_UR_PRESS    14
KEY_LL_PRESS    15
KEY_LR_PRESS    16
KEY_UL_RELEASE  17
KEY_UR_RELEASE  18
KEY_LL_RELEASE  19
KEY_LR_RELEASE  20
```

Please note that the [CFA-633](#) and [CFA-635](#) will return codes 1 through 12. Please see those data sheets on our website for more details.

0x81: Fan Speed Report (SCAB required)

If any of up to four fans connected to CFA-631+SCAB is configured to report its speed information to the host, the CFA-631 will send Fan Speed Reports for each selected fan every 1/2 second. See command [16: Set Up Fan Reporting \(SCAB required\) \(Pg. 20\)](#) below.



```
type = 0x81
data_length = 4
data[0] is the index of the fan being reported:
    0 = FAN 1
    1 = FAN 2
    2 = FAN 3
    3 = FAN 4
data[1] is number of fan tach cycles
data[2] is the MSB of Fan_Timer_Ticks
data[3] is the LSB of Fan_Timer_Ticks
```

The following C function will decode the fan speed from a Fan Speed Report packet into RPM:

```
int OnReceivedFanReport(COMMAND_PACKET *packet, char * output)
{
    int
    return_value;
    return_value=0;

    int
    number_of_fan_tach_cycles;
    number_of_fan_tach_cycles=packet->data[1];

    if(number_of_fan_tach_cycles<3)
        sprintf(output, " STOP");
    else if(number_of_fan_tach_cycles<4)
        sprintf(output, " SLOW");
    else if(0xFF==number_of_fan_tach_cycles)
        sprintf(output, " ----");
    else
    {
        //Specific to each fan, most commonly 2
        int
        pulses_per_revolution;
        pulses_per_revolution=2;

        int
        Fan_Timer_Ticks;
        Fan_Timer_Ticks=(*(unsigned short *)(&(packet->data[2])));

        return_value=((27692308L/pulses_per_revolution)*
            (unsigned long)(number_of_fan_tach_cycles-3))/
            (Fan_Timer_Ticks);
        sprintf(output, "%5d", return_value);
    }
    return(return_value);
}
```

0x82: Temperature Sensor Report (SCAB required)

If any of the up to 32 temperature sensors is configured to report to the host, the CFA-631+SCAB will send Temperature Sensor Reports for each selected sensor every second. See the command [19: Set Up Temperature Reporting \(SCAB required\) \(Pg. 22\)](#) below.

```
type = 0x82
data_length = 4
data[0] is the index of the temperature sensor being reported:
    0 = temperature sensor 1
    1 = temperature sensor 2
    .
    .
    31 = temperature sensor 32
data[1] is the LSB of Temperature_Sensor_Counts
data[2] is the MSB of Temperature_Sensor_Counts
data[3] is DOW_crc_status
```



The following C function will decode the Temperature Sensor Report packet into °C and °F:

```
void OnReceivedTempReport(COMMAND_PACKET *packet, char *output)
{
    //First check the DOW CRC return code from the CFA-631
    if(packet->data[3]==0)
        strcpy(output, "BAD CRC");
    else
    {
        double
            degc;
        degc=(*(short *)&(packet->data[1]))/16.0;

        double
            degf;
        degf=(degc*9.0)/5.0+32.0;

        sprintf(output, "%9.4f°C =%9.4f°F",
                degc,
                degf);
    }
}
```

COMMAND CODES

Below is a list of valid commands for the CFA-631. Each command packet is answered by either a response packet or an error packet. The low 6 bits of the type field of the response or error packet is the same as the low 6 bits of the type field of the command packet being acknowledged.

0: Ping Command

The CFA-631 will return the Ping Command to the host.

```
type = 0
valid data_length is 0 to 16
data[0-(data_length-1)] can be filled with any arbitrary data
```

The return packet is identical to the packet sent, except the type will be 0x40 (normal response, Ping Command):

```
type = 0x40 | 0
data_length = (identical to received packet)
data[0-(data_length-1)] = (identical to received packet)
```

1: Get Hardware & Firmware Version

The CFA-631 will return the hardware and firmware version information to the host.

```
type = 1
valid data_length is 0
```

The return packet will be:

```
type = 0x40 | 1
data_length = 16
data[] = "CFA-631:hX.X,yY.Y"
```

X.X is the hardware revision, "2.0" for example
yY.Y is the firmware version, "v2.0" for example

2: Write User Flash Area

The CFA-631 reserves 16 bytes of nonvolatile memory for arbitrary use by the host. This memory can be used to store a serial number, IP address, gateway address, netmask, or any other data required. All 16 bytes must be supplied.



```
type = 2
valid data_length is 16
data[] = 16 bytes of arbitrary user data to be stored in
        the CFA-631's non-volatile memory
```

The return packet will be:

```
type = 0x40 | 2
data_length = 0
```

3: Read User Flash Area

This command will read the User Flash Area and return the data to the host.

```
type = 3
valid data_length is 0
```

The return packet will be:

```
type = 0x40 | 3
data_length = 16
data[] = 16 bytes user data recalled from the CFA-631's
        non-volatile memory
```

4: Store Current State As Boot State

The CFA-631 loads its power-up configuration from nonvolatile memory when power is applied. The CFA-631 is configured at the factory to display a “welcome screen” when power is applied. This command can be used to customize the welcome screen, as well as the following items:

- Characters shown on LCD, which are affected by:
 - command [6: Clear LCD Screen \(Pg. 17\)](#)
 - command [7: Set LCD Contents, Line 1 \(CFA-633 compatibility\) \(Pg. 18\)](#)
 - command [8: Set LCD Contents, Line 2 \(CFA-633 compatibility\) \(Pg. 18\)](#)
 - command [31: Send Data to LCD \(Pg. 31\)](#)
- Special character font definitions (command [9: Set LCD Special Character Data \(Pg. 18\)](#))
- Cursor position (command [11: Set LCD Cursor Position \(Pg. 19\)](#))
- Cursor style (command [12: Set LCD Cursor Style \(Pg. 19\)](#))
- Contrast setting (command [13: Set LCD Contrast \(Pg. 20\)](#))
- Backlight setting (command [14: Set LCD & Keypad Backlight \(Pg. 20\)](#))
- Fan power settings (command [17: Set Fan Power \(SCAB required\) \(Pg. 21\)](#))
- Settings of any “live” displays (command [21: Set Up Live Fan or Temperature Display \(SCAB required\) \(Pg. 24\)](#))
- Key press and release masks (command [23: Configure Key Reporting \(Pg. 25\)](#))
- Fan glitch delay settings (command [26: Set Fan Tachometer Glitch Filter \(SCAB required\) \(Pg. 26\)](#))
- ATX function enable and pulse length settings (command [28: Set ATX Power Switch Functionality \(SCAB required\) \(Pg. 28\)](#))
- Key legends (command [32: Key Legends \(Pg. 31\)](#))
- Baud rate (command [33: Set Baud Rate \(Pg. 32\)](#))
- GPIO settings (command [34: Set or Set and Configure GPIO Pin \(Pg. 32\)](#))

You cannot store the fan or temperature reporting (although the live display of fans or temperatures can be saved). You cannot store the fan fail-safe or host watchdog. The host software should enable these items once the system is initialized and it is ready to receive the data.

```
type = 4
valid data_length is 0
```



The return packet will be:

```
type = 0x40 | 4  
data_length = 0
```

5: Reboot CFA-631, Reset Host (SCAB required), or Power Off Host (SCAB required)

This command instructs the CFA-631+SCAB to simulate a power-on restart of itself, reset the host, or turn the host's power off. The ability to reset the host may be useful to allow certain host operating system configuration changes to complete. The ability to turn the host's power off under software control may be useful in systems that do not have ACPI compatible BIOS.

NOTE

The GPIO pins used for ATX control must not be configured as user GPIO, and must be configured to their default drive mode in order for the ATX functions to work correctly. These settings are factory default, but may be changed by the user. Please see command [34: Set or Set and Configure GPIO Pin \(Pg. 32\)](#).

Rebooting the CFA-631 may be useful when testing the boot configuration. It may also be useful to re-enumerate the devices on the 1-Wire bus (SCAB required). To reboot the CFA-631, send the following packet:

```
type = 5  
valid data_length is 3  
data[0] = 8  
data[1] = 18  
data[2] = 99
```

To reset the host (SCAB required), assuming the host's reset line is connected to GPIO[3] as described in command [28: Set ATX Power Switch Functionality \(SCAB required\) \(Pg. 28\)](#), send the following packet:

```
type = 5  
valid data_length is 3  
data[0] = 12  
data[1] = 28  
data[2] = 97
```

To turn the host's power off (SCAB required), assuming the host's power control line is connected to GPIO[2] as described in command [28: Set ATX Power Switch Functionality \(SCAB required\) \(Pg. 28\)](#), send the following packet:

```
type = 5  
valid data_length is 3  
data[0] = 3  
data[1] = 11  
data[2] = 95
```

In any of the above cases, the return packet will be:

```
type = 0x40 | 5  
data_length = 0
```

6: Clear LCD Screen

Sets the contents of the LCD screen DDRAM to ' ' = 0x20 = 32 and moves the cursor to the left-most column of the top line.

```
type = 6  
valid data_length is 0
```

The return packet will be:

```
type = 0x40 | 6  
data_length = 0
```



Clear LCD Screen is one of the items stored by the command [4: Store Current State As Boot State \(Pg. 16\)](#).

7: Set LCD Contents, Line 1 (CFA-633 compatibility)

Sets the center 16 characters displayed for the top line of LCD screen. The first two and last two characters are blanked.

This command is provided to allow legacy software that displays data on the CFA-633 to work unchanged on the CFA-631. For new applications, please use the more flexible command [31: Send Data to LCD \(Pg. 31\)](#).

```
type = 7
valid data_length is 16
data[] = top line's display content (must supply 16 bytes)
```

The return packet will be:

```
type = 0x40 | 7
data_length = 0
```

Set LCD Contents, Line 1 is one of the items stored by the command [4: Store Current State As Boot State \(Pg. 16\)](#).

8: Set LCD Contents, Line 2 (CFA-633 compatibility)

Sets the center 16 characters displayed for the bottom line of LCD screen. The first two and last two characters are blanked.

This command is provided to allow legacy software that displays data on the CFA-633 to work unchanged on the CFA-631. For new applications, please use the more flexible command [31: Send Data to LCD \(Pg. 31\)](#).

```
type = 8
valid data_length is 16
data[] = top line's display content (must supply 16 bytes)
```

The return packet will be:

```
type = 0x40 | 8
data_length = 0
```

Set LCD Contents, Line 2 is one of the items stored by the command [4: Store Current State As Boot State \(Pg. 16\)](#).

9: Set LCD Special Character Data

Sets the font definition for one of the special characters (CGRAM).

```
type = 9
valid data_length is 9
data[0] = index of special character that you would like
         to modify, 0-7 are valid
data[1-8] = bitmap of the new font for this character
```

`data[1-8]` are the bitmap information for this character. Any value is valid between 0 and 63, the msb is at the left of the character cell of the row, and the lsb is at the right of the character cell. `data[1]` is at the top of the cell, `data[8]` is at the bottom of the cell.

Additionally, if you set bit 7 of any of the data bytes, the entire line will blink.

The return packet will be:

```
type = 0x40 | 9
data_length = 0
```



Set LCD Special Character Data is one of the items stored by the command [4: Store Current State As Boot State \(Pg. 16\)](#).

10: Read 8 Bytes of LCD Memory

This command will return the contents of the LCD's DDRAM or CGRAM. This command is intended for debugging.

```
type = 10
valid data_length is 1
data[0] = address code of desired data
```

data [0] is the address code native to the LCD controller:

```
0x40 (\064) to 0x7F (\127) for CGRAM
0x80 (\128) to 0x93 (\147) for DDRAM, line 1
0xC0 (\192) to 0xD3 (\211) for DDRAM, line 2
```

The return packet will be:

```
type = 0x40 | 10
data_length = 9
```

data [0] of the return packet will be the address code.

data [1-8] of the return packet will be the data read from the LCD controller's memory.

11: Set LCD Cursor Position

This command allows the cursor to be placed at the desired location on the CFA-631's LCD screen. If you want the cursor to be visible, you may also need to send a command [12: Set LCD Cursor Style \(Pg. 19\)](#).

```
type = 11
valid data_length is 2
data[0] = column (0-19 valid)
data[1] = row (0-1 valid)
```

The return packet will be:

```
type = 0x40 | 11
data_length = 0
```

Set LCD Cursor Position is one of the items stored by the command [4: Store Current State As Boot State \(Pg. 16\)](#).

12: Set LCD Cursor Style

This command allows you to select among four hardware generated cursor options.

```
type = 12
valid data_length is 1
data[0] = cursor style (0-4 valid)
0 = no cursor
1 = blinking block cursor
2 = underscore cursor
3 = blinking block plus underscore
4 = inverting, blinking block
```

The return packet will be:

```
type = 0x40 | 12
data_length = 0
```

Set LCD Cursor Style is one of the items stored by the command [4: Store Current State As Boot State \(Pg. 16\)](#).



13: Set LCD Contrast

This command sets the contrast or vertical viewing angle of the display.

```
type = 13
valid data_length is 1
data[0] = contrast setting (0-255 valid)
    0-39 = very light
    40 = light
    90 = about right
    125 = dark
    126-255 = very dark
```

The return packet will be:

```
type = 0x40 | 13
data_length = 0
```

Set LCD Contrast is one of the items stored by the command [4: Store Current State As Boot State \(Pg. 16\)](#).

14: Set LCD & Keypad Backlight

This command sets the brightness of the LCD and keypad backlights.

```
type = 14
valid data_length is 1
data[0] = backlight power setting (0-100 valid)
    0 = off
    1-99 = variable brightness
    100 = on
```

The return packet will be:

```
type = 0x40 | 14
data_length = 0
```

Set LCD & Keypad Backlight is one of the items stored by the command [4: Store Current State As Boot State \(Pg. 16\)](#).

15: (Deprecated)

16: Set Up Fan Reporting (SCAB required)

This command will configure the CFA-631+SCAB to report the fan speed information to the host every 500 mS.

```
type = 16
valid data_length is 1
data[0] = bitmask indicating which fans are enabled to
report (0-15 valid)
---- 8421 Enable Reporting of this Fan's Tach Input
||||| | | | | -- Fan 1: 1 = enable, 0 = disable
||||| | | | | --- Fan 2: 1 = enable, 0 = disable
||||| | | | | ---- Fan 3: 1 = enable, 0 = disable
||||| | | | | ----- Fan 4: 1 = enable, 0 = disable
```

The return packet will be:

```
type = 0x40 | 16
data_length = 0
```

If data [0] is not 0, then the CFA-631+SCAB will start sending 0x81: Fan Speed Report packets for each enabled fan every 500mS. (See [0x81: Fan Speed Report \(SCAB required\) \(Pg. 13\)](#).) Each of the report packets is staggered by 1/8 of a second.

