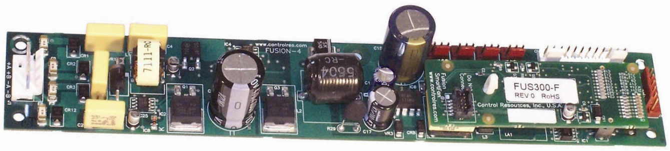
The driving force
of electronics cooling.

Advantages of I^2C Communication for Fan Speed Control in Telecom Cooling Applications

It is generally clear why the I^2C bus has been adapted as the standard format for management communications within the telecom chassis. With the wide selection of temperature sensors and EEPROM devices that can all be interconnected using only two lines, there is a strong advantage to use this interface. A logical progression is to expand this bus to interface with the fan tray to reduce connector pin count, saving cost on the interconnects, while adding flexibility to the system. This type of system can allow the shelf management to monitor thermal data from each card in the chassis and then determine what speed to run the fans at. The fan controller can report its revision and inventory information to allow decisions to be made with the same shelf manager using different variations of fan controls. In addition, nearly any number of alarm indicators can be sent with just two wires, along with data that was not available before, such as temperature readings, fan speed data, voltage output, etc.

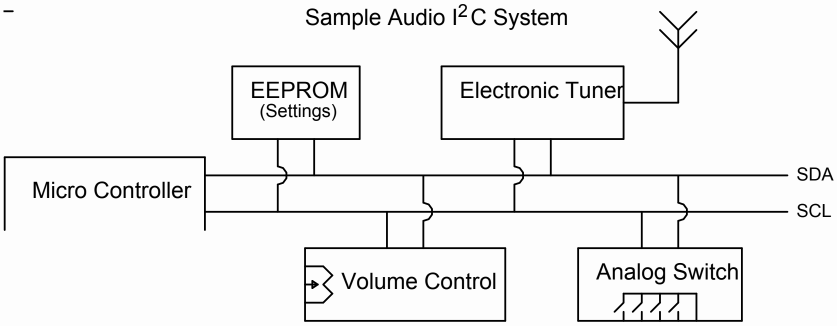


SmartFan Fusion-4 Fan Control and Tach Alarm with dual I2C Interface

Another large advantage with I^2C is that the protocol is defined and is generally already used to interface to the shelf manager's EEPROM and local temperature sensor. While a two wire interface can provide many advantages described above, it is helpful to understand the origins of this chipset to avoid decisions that can cause system problems throughout product life. Fortunately, many of these problems can be eliminated with virtually no added cost by understanding how to implement the I^2C bus correctly for off-board communications. This brief article provides an overview of historical development of the two wire bus, how this initial intent brings certain limitations, and how to overcome the inherent limitations of the two wire open collector format. This article does expect some knowledge of the I^2C protocol and does not attempt to instruct on the general use of the protocol. For a description of the I^2C protocol, I refer readers to "The I^2C-bus and how to use it" from Philips Semiconductor.

What is I^2C anyway?

I^2C or IIC stands for Inter-Integrated Circuit Bus. The original design intent was for use in audio equipment where a microcontroller based system needed to control volume/balance controls, a tuner circuit, an LCD panel, and EEPROM at a minimum and often several other devices as well. The I/O requirements to interface each device in a small circuit board that fit into an automotive head unit became in impossible task given space and cost constraints. In response Philips created the I^2C protocol, allowing all these devices to be attached to a single two wire bus, each having a unique address. To simplify the drive requirements and allow bi-directional communication, these two lines were low current open collector. When this bus is contained on a very small circuit board with a single ground, this design of I/O works properly.



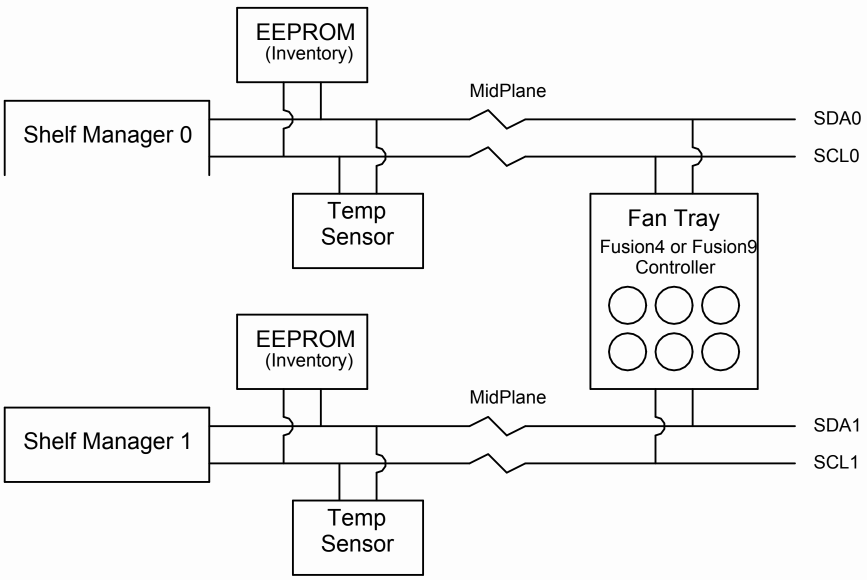
Because the interface was so successful in audio products several IC's were produced supporting this interface. This wide selection of chips - from memory, to sensors, to A/D and D/A converters - fueled the use of I^2C in numerous other applications including telecom chassis management.

Advantages of I^2C Communication for Fan Speed Control in Telecom Cooling Applications

I^2C Adapted to Telecom Use

Without going over every variation and detail of the I^2C specification, there are a few key points that will help ensure a stable and error free system. Keeping in mind that a telecom chassis is not the same environment as a local temperature sensor and recognizing that a more error resistant methodology is required, are the most important steps to reliable system design. Three of the most common issues are: proper care for open collector drive, correct observance of clock stretching requirements, and sufficient error detection and handling in the master driver.

Sample Telcom I^2C System

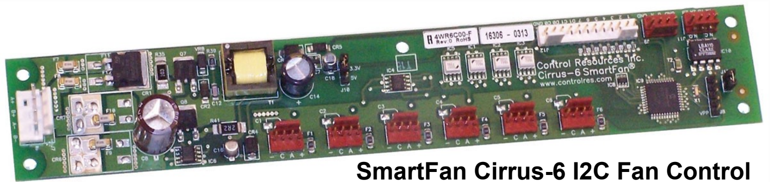


Open Collector Considerations:

The clock (SCL) and data (SDA) lines are both open collector with 3mA minimum sink capability and *both* are bi-directional. Only 3mA of sink capability is required to be compliant, so the pull-ups must be kept weak. The maximum bus capacitance is specified at 400pf and this must be observed. If ignored, random errors can occur and capacitive filtering *cannot* be added. These requirements limit what is possible to enhance noise immunity. Long lines to the fan control board will add capacitance and must be taken into account when calculating the total bus capacitance. If you find that the bus capacitance is greater than 400pf, then the I^2C bus routing must be modified to reduce it. Many devices may work sporadically with greater than 400pf capacitance, but proper operation cannot be guaranteed. Another important consideration is the unshielded harness in the fan tray itself. Often to simplify wiring, the communications and power are bundled in a single wrap. While this practice can work in some cases, splitting the power and I^2C signals will *always* yield better performance. Ideally the power is brought in on a separate connector. If this is not possible, separating the signals and power will greatly improve the noise margin. Another method to reduce noise is to terminate both ends of the transmission line with pull-up resistors. The total source current of the bus must be observed when several cards have their own pull-ups. The total source current cannot exceed 3mA to be compliant with the protocol. This specification can be ignored in a specific case; if you know the sink current of all the devices on the bus and they all have greater limits than the 3mA minimum specified, a lower pull-up resistor can be used to improve noise performance.

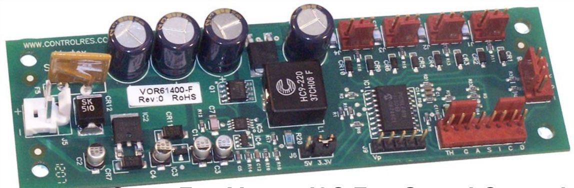
Advantages of I^{2}C Communication for Fan Speed Control in Telecom Cooling Applications

Clock Stretching:
The clock (SCL) is also bidirectional. Most synchronous communications are clocked by the master. I^{2}C defines that the master initiates the clock signals; however, the master must observe when the clock is held externally low and wait for the clock to be released. This allows multi-master bus devices to handle arbitration and allows the slave to delay transmission until data is ready. When communicating with a dedicated memory or sensor chip, this part of the protocol can often be ignored with no notable effects. This is because these parts use dedicated hardware to load the registers and the time delay is less than the clock cycle so clock stretching is not required. With a fan controller, or any general microcontroller device, there can be a delay while firmware determines what data is loaded into the data buffer. I^{2}C allows for this delay by allowing the slave to hold the clock low while retrieving the data. If the master simply toggles the clock without monitoring its level, it may sample data when the clock is still low and the data will give an erroneous high data. Additionally the master will think it produced more clock pulses than were generated on the bus which will cause the slave to be out of synchronization with the master, causing further errors. When this happens on a read there is no way for the slave to inform the master of the error, so the master can continue reading invalid data indefinitely.



SmartFan Cirrus-6 I2C Fan Control and Tach Alarm for 4-Wire Fans

Error Detection:
Another common cause of problems is the misconception that I^{2}C is a complete protocol. I^{2}C is a hardware definition; it is not a software protocol. Once a device receives a read address from the master it will respond with an acknowledge. This indicates to the master that a device is connected with this address and it is ready. From this point on in the communication string, the *master* generates the acknowledge signal to tell the slave when to end the transmission. The slave has *no* way to indicate an error to the master during this entire transmission. A transmission that is prone to noise glitches cannot be implemented with a "read and go on" approach and ensure consistent results. With a short bus contained in a quiet circuit this approach can often work. With long unshielded cables bundled with high power lines, this approach is nearly certain to fail. The question is when and how often. Fortunately, this problem can be easily overcome by using single byte access commands and oversampling, which can be handled in the master's firmware/software and does not generally add cost to the system hardware.



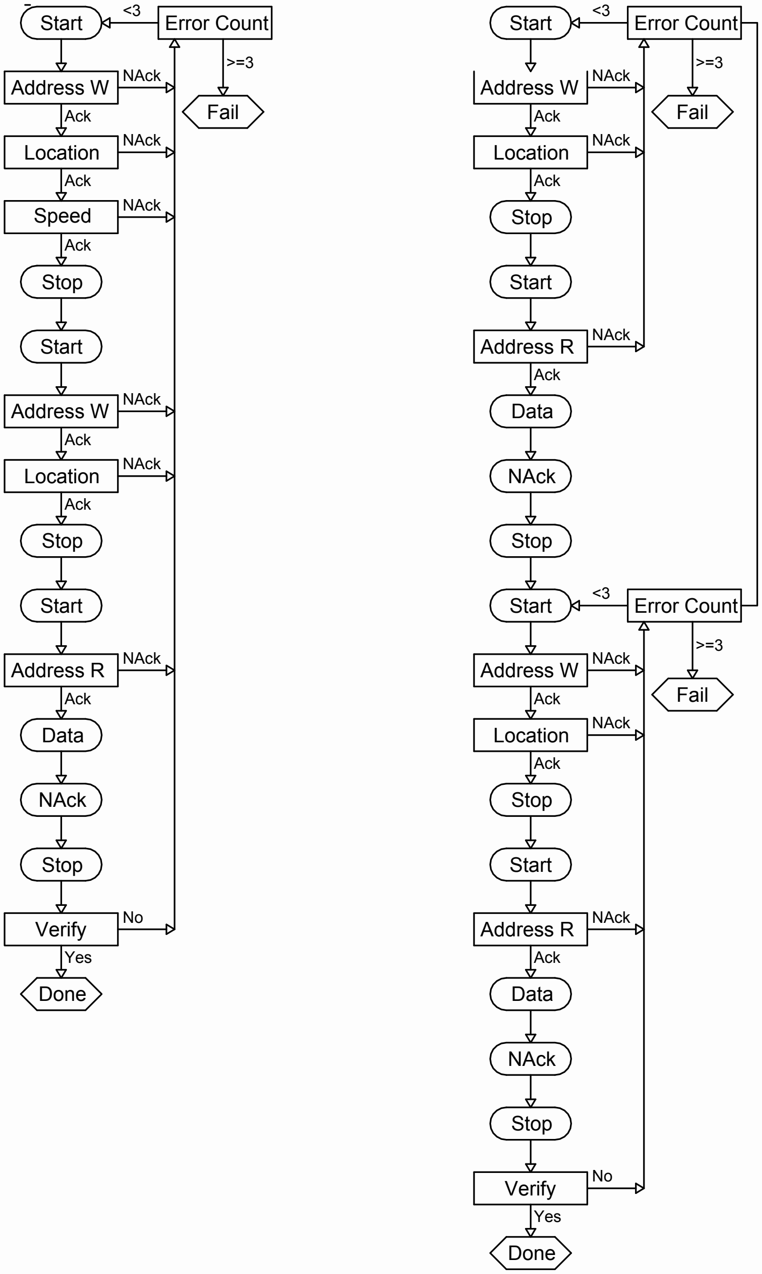
SmartFan Vortex I2C Fan Speed Control and Tach Alarm for 12 VDC Fans

These three guidelines will not correct every issue that is possible to encounter in an I^{2}C system, although they will avoid most of the common pitfalls implementing an I^{2}C interface in a telecom fan tray.

About the author: *Nathan Lavoie has a BSEE from the University of Vermont and is Vice President for Engineering at Control Resources, Inc., where he has developed more than 50 I^{2}C based custom fan controllers.*

Advantages of I^{2}C Communication for Fan Speed Control in Telecom Cooling Applications

Sample I^{2}C Protocol Flow Diagram



Speed Command

Alarm Poll

Advantages of I^2C Communication for Fan Speed Control in Telecom Cooling Applications

Standard I^2C Protocol for SmartFan Fusion-4

I2C Bus Protocol

The I2C bus is configured as a slave device that can transmit and receive data. When the Fusion-4 is configured for I2C based speed control, the user can write speed commands to the Fusion-4 and read all alarm status bits. When configured for temperature based control the user can read all alarm status bits from the Fusion-4. Writing an I2C speed command changes the control mode from temperature based to I2C based.

Specifications

The bus supports seven-bit addressing and only acts as a slave device.
The address for the fan controller is 0001A2A1A0.
General call support is not provided.
The fan controller will stretch the clock further if needed. The I2C master must observe clock stretching.
The hardware register will support Standard Mode I2C with speeds up to 100 KHz, however, lower clock speeds are recommended for increased noise immunity.
If temperature control mode is disabled and I2C speed mode is used, then I2C communication must occur within ~2 minutes of the last communication, or there will be an I2C communication timeout error. When an I2C timeout occurs, full supply voltage will be applied to the fans, and fan speed must be reset with a new speed command. Performing any valid I2C operation on either bus will clear the timeout condition and restart the watchdog timer.
Fan failure based on 2000 minimum PPM of fan in slow (default) mode or 4000 minimum PPM in fast mode. Fan failure is not updated when the output is set to 0 VDC.
It is possible for noise to cause the I2C hardware to miss an address or data. In this case an Ack will not be generated. The production tester allows 3 errors to occur before marking the board as failing. Also during the EEPROM write cycle the I2C will not generate ACK signals as no more data can be accepted until the write cycle is complete. Typical write cycle is 8ms.

Table 3.0 defines the bits used in the serial bus protocol.

Table 3.0: I^2C Bus Definitions

ACK	Acknowledge
NAck	Not Acknowledge (End of Transmission)
A6 – A0	Address for Fan Controller, bits 6 – 0
D7 – D0	Data TO/FROM Fan Controller, bits 7 – 0
L7 – L0	Data location being accessed, bits 7-0
P	Stop Bit
R	Read Bit (1)
S	Start Bit
W	Write Bit (0)

Write sequence
S A6 A5 A4 A3 A2 A1 A0 W Ack L7 L6 L5 L4 L3 L2 L1 L0 Ack D7 D6 D5 D4 D3 D2 D1 D0 Ack P

Read sequence
S A6 A5 A4 A3 A2 A1 A0 W Ack L7 L6 L5 L4 L3 L2 L1 L0 Ack P S A6 A5 A4 A3 A2 A1 A0 R Ack D7 D6 D5 D4 D3 D2 D1 D0 NAck P

Advantages of I^2C Communication for Fan Speed Control in Telecom Cooling Applications

Standard I^2C Protocol for SmartFan Fusion-4

Register Locations: L7-L0 Label (R/W) [initial condition]

Register 0x00 – 0xEF: Generic EEPROM storage (Read/Write) [undetermined]

Register 0xF0: Firmware Revision (Read only) [00:current rev]

Register 0xF1: Alarm0 Status (Read only) [0x00]
D0: Status of fan on J3 (0=no fault, 1= fault)
D1: Status of fan on J4 (0=no fault, 1= fault)
D2: Status of fan on J5 (0=no fault, 1= fault)
D3: Status of fan on J6 (0=no fault, 1= fault)
D4: Always 0
D5: Always 0
D6: Always 0
D7: Always 0

Register 0xF2: Alarm1 Status (Read only) [0x00]
D0: Always 0
D1-3: Always 0
D4: Status of speed override (0=normal, 1=24VDC)
D5: Status of on-board sensor (0=connected, 1= open)
D6: Status of external sensor (0=connected, 1=open)
D7: Status of controlling sensor (0=valid, 1=alarm)

Register 0xF3: Temperature (Read only) [1111 1111] on-board sensor
D0-7: Temperature from 0 - 70 in degrees C in 0.5 deg steps
0XFF = temperature not read yet or open
(Temperatures above 70 are reported as 254)

Register 0xF4: Temperature (Read only) [1111 1111] external sensor
D0-7: Temperature from 0 - 70 in degrees C in 0.5 deg steps
0XFF = temperature not read yet or open
(Temperatures above 70 are reported as 254)

Register 0xF5: Current Target Speed (Read only) [0000 1100]
D0-7: 0x0C = 24 VDC, 0x00 = 12 VDC
(Commanded speed with alarm status override included)
(In temperature mode this register will read 0x0C)

Register 0xF6: Current Speed (Read only) [0000 1100]
D0-7: 0x0C = 24 VDC, 0x00 = 12VDC
(What the actual current voltage is)
(In temperature mode this register will read 0x0C)

Advantages of I^{2}C Communication for Fan Speed Control in Telecom Cooling Applications

Standard I^{2}C Protocol for SmartFan Fusion-4

Register 0xF7: Commanded Speed (Read/Write) [1111 1111]
D0-3: 0x0C = 24 VDC, 0x00 = 12VDC, 0x0F = off
0x00 – 0C = 12 – 24 VDC, 1VDC per step
If set to 0xFF then temperature mode is enabled and I2C speed control disabled

Register 0xF8: Off Temperature (Read/Write) [1111 1111]
D0-7: Temperature from 0 - 70 in degrees C in 0.5 deg steps
Temperature below which the controller turns off the fan output (0 VDC)
If set to 0xFF then function disabled

Register 0xF9: Control Temperature (Read/Write) [0101 0000]
D0-7: Temperature from 0 - 70 in degrees C in 0.5 deg steps
Full speed temperature, only used if Commanded Speed set to 0xFF.

Register 0xFA: Alarm Temperature (Read/Write) [1111 1111]
D0-7: Temperature from 0 - 70 in degrees C in 0.5 deg steps
Temperature above which the controller alarms
If set to 0xFF then function disabled

Register 0xFB: Config0 Register (Read/Write) [0000 1111]
D0-D3: 0=Mask, 1=Active for an Fan (J3-J6)
D4-D7: Not Used

Register 0xFC: Config1 Register (Read/Write) [0000 0000]
D0: Not Used
D1-D4: Set to 0, may be set to 1 to allow system to indicate a fan controller reset.
D5: 0=5deg slope, 1=10 deg slope for temperature mode only
D6: 0=2000PPM tach, 1=4000PPM tach fan alarm speed
D7: 0=External, 1=On-board sensor selection for temperature mode only

Register 0xFD-0xFF: CRI Test Register (Reserved) [xxxx xxxx]
D0-7: Do not use

Note: If a fan failure occurs the Current Speed will be updated to 24V but the Commanded Speed will remain at its previous value. Once the failure clears the Current Speed will return to the Commanded Speed value. An I2C timeout changes BOTH the Current Speed and Commanded Speed to 24V. The Commanded Speed must be reset after a timeout. During the speed ramp the Current Speed will not match the Commanded Speed until the ramp completes.