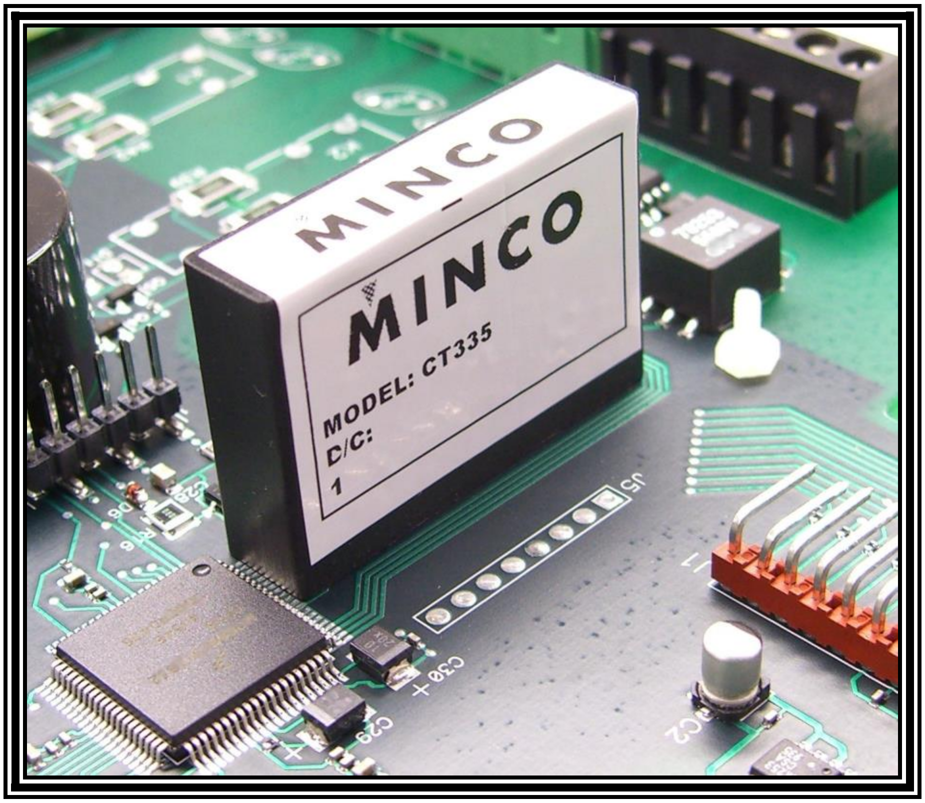
CT335 Serial Peripheral Interface (SPI) Communication Instruction Manual



Introduction

The purpose of this manual is to present the user with the CT335 SPI communication protocol, and how manipulations of the CT335 parameters are done.

CT335 SPI overview

The CT335 is configured to always be in slave mode in SPI master-slave configuration and operates using 5V logic levels for the SPI communication. The SPI for the CT335 consists of four signals:

SCK: Serial Clock (internal pull-up)
MOSI: Master Out Slave In (internal pull-up)
MISO: Master In Slave Out
SS: Slave Select (internal pull-up)

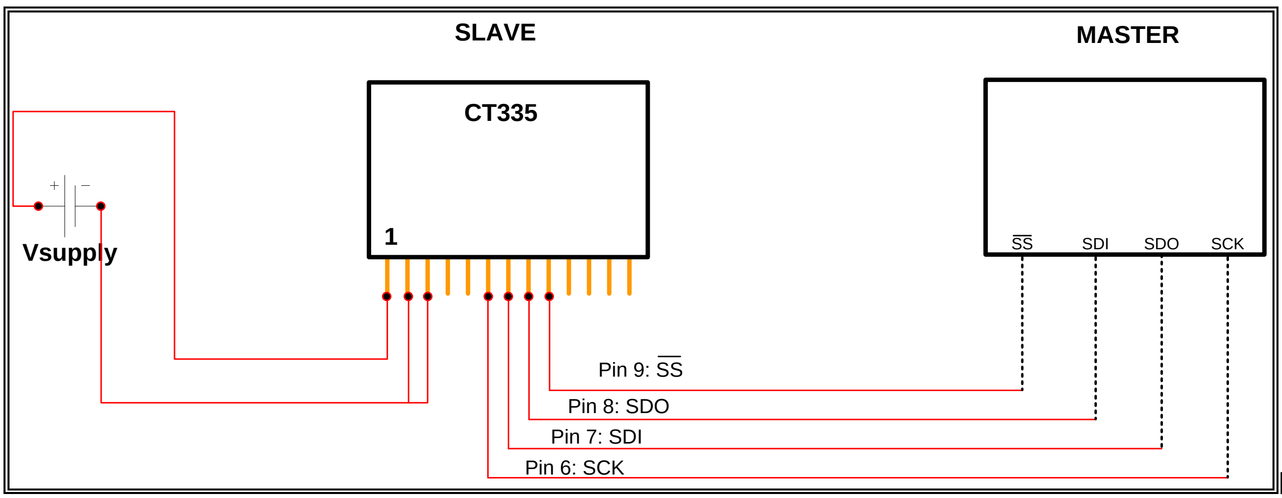


Fig. 1

Master/Slave interface

The CT335 is designed to receive 9 bytes of information from the master device, in order to communicate correctly with the CT335 all relevant bytes sent from the master device must be sent in the following order and as a packet of 9 bytes:

a) 1 Function code byte
b) 1 Variable code byte
c) 1 Data length byte
d) 4 Data bytes
e) 1 Checksum byte
f) 1 Garbage byte (recommend 0x00).

Since SPI communication is full duplex, for every byte that is sent by the master device, it receives one back. In the CT335 case, the master device can expect to get what it sent echoed back except when the sent byte is invalid. When the master device sends an invalid byte, it will receive a *0xBB* error code back. Due to the amount of data being transferred, it is recommended that the baud-rate be 9.6kbps and no more than 11.7kbps. See fig. 2 for CT335 SPI communication waveform.

CT335 SPI MODE:
CKP = 1, CKE = 0
CKP = Clock Polarity, CKE = Clock Edge

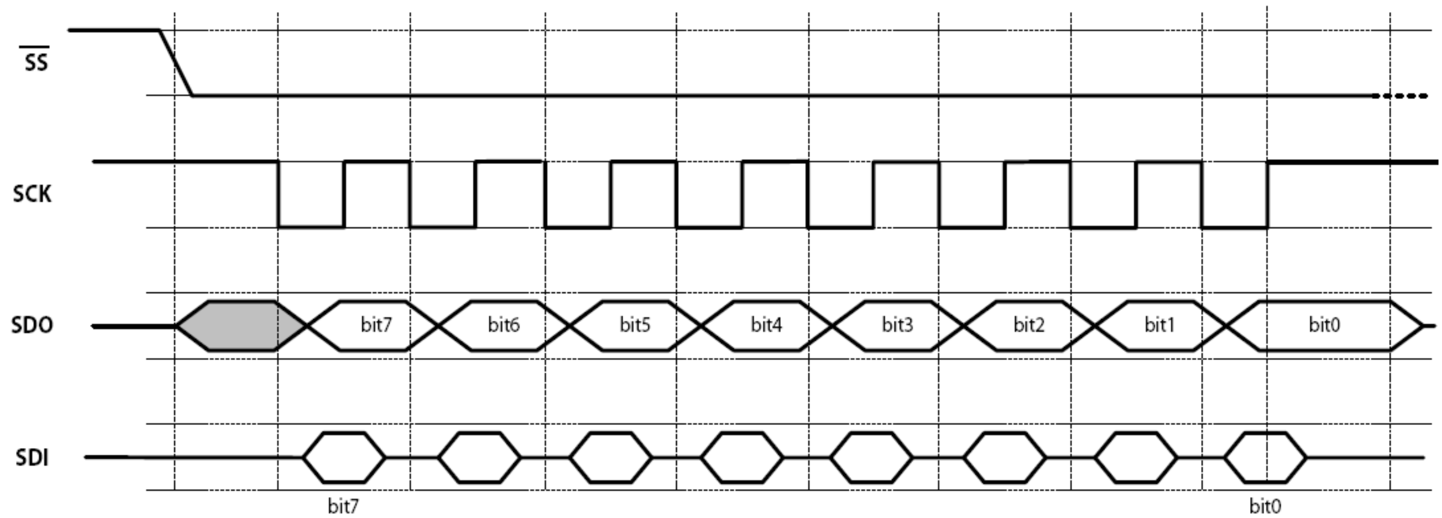


Fig 2.

Note: Slave Select (SS) should be low for the entire transmission of 9 bytes (packets).

Commands

Function code byte:

- 0x01 – Read from CT335
- 0x02 – Write to CT335

Variable code byte:

- 0x11 – Setpoint channel 1 (-40º to 200º C)
- 0x12 – Setpoint channel 2 (-40º to 200º C)
- 0x21 – Proportional band channel 1 (0.1º to 10º C)
- 0x22 – Proportional band channel 2 (0.1º to 10º C)
- 0x51 – Dead band channel 1 (0.1º to 10º C)
- 0x52 – Dead band channel 2 (0.1º to 10º C)
- 0x91 – Control type (1 or 2)
- 0xB1 – Sensor 1 (for monitoring temperature) – *Read only*
- 0xB2 – Sensor 2 (for monitoring temperature) – *Read only*
- 0xC1 – Offset for channel 1 (0.0º to 10º C)
- 0xC2 – Offset for channel 2 (0.0º to 10º C)

Note: Control type is equal to 1 for On/Off control and equal 2 for Proportional control.

Data length byte:

- 0x04 – The data length byte for the CT335 will always be 0x04.

Checksum byte:

- The CT335 uses Longitudinal Redundancy Check (LRC) for its checksum. Checksum is calculated by XOR-ing the function code, variable code, data length, and the four bytes of data for example:

- Function code = 0x01 (read)
- Variable code = 0x11 (setpoint)
- Data length = 0x04
- Data = 0x85 0x48 0x00 0x00 (since it is a read, data can be anything).

Checksum = [0x01]xor[0x11]xor[0x04]xor[0x85]xor[0x48]xor[0x00]xor[0x00]

- Checksum should equal 0xD9.

Garbage byte:

- In theory the garbage byte can be anything, but it is recommended that the garbage byte be 0x00 for a write/read.

4 Data bytes:

- The four data bytes of the CT335 uses Microchip 32-bit floating point formats[1].

	Bias exponent (eb)	*f0*	*f1*	*f2*
Microchip 32-bit	xxxx xxxx	***S***·xxx xxxx	xxxx xxxx	xxxx xxxx

Legend: **s** is the Sign bit, · = radix point

The ***eb*** is an 8-bit exponent with bias equal to 127, ***s*** is the sign bit and is equal to one for negative numbers and zero for positive number , and ***f0, f1, f2*** makes up the fraction with ***f0*** the being most significant byte with implicit MSB = 1.

EXAMPLE 1: CONVERTING POSITIVE DECIMAL TO MICROCHIP FLOAT FORMAT (See example 2 for converting negative decimal number to Microchip float format)

1) Lets convert **25.785** to Microchip float format
2) Find the ***eb***:

   2^z = 25.785
   z = ln(25.785) / ln(2) → z = 4.68846 → e = int(z) = 4
   ***eb* = e + 0x7F → *eb* = 4 + 0x7F → *eb* = 0x83**

3) Find the fraction *f*:

   ***f* = 25.785 / 2^4 → 1.6115625**
   Since we are working with a positive number, s is equal to zero. So do the following to find the fraction:
   → *f* ' = 0.6115625 → 0.6115625*2^{23} = 5130158.08 (where *f'* = *f* – 1)
   → Round 5130158.08 to nearest integer which gives 5130158
   → Convert 5130158 to hex gives 0x4E47AE

4) Putting it all together, **25.785** in Microchip float format is **0x834E47AE**

1. Reference: Microchip AN575 IEEE 754 Compliant Floating Point Routines for more info.
http://ww1.microchip.com/downloads/en/AppNotes/00575.pdf

EXAMPLE 2: CONVERTING NEGATIVE DECIMAL TO MICROCHIP FLOAT FORMAT

1) Lets convert **-37.863** to Microchip float format
2) Find the ***eb:***

   2^z = -37.863
   z = ln(37.863) / ln(2) → z = 5.2427168 → e = int(z) → 5
   ***eb* = e + 0x7F → *eb* = 5 + 0x7F = 0x84**

3) Find the fraction *f*:

   ***f* = 37.863 / 2^5 → 1.1832188**
   Here the sign bit is equal to one.
   → *f'* = 0.1832188 * 2^{23} + 2^{23} = 9925558.272 (where *f'* = *f* - 1)
   → Round 9925558.272 to nearest integer which gives 9925558
   → Convert 9925558 to hex gives 0x9773B6

4) Therefore, **-37.863** in Microchip float format is **0x849773B6**

EXAMPLE 3: CONVERT MICROCHIP FLOAT FORMAT TO DECIMAL

1) Lets convert **0x85A5224E** to decimal.
2) Find the exponent:

   e = *eb* – bias = 0x85 – 0x7F = 6

   1 here indicates a negative number.

3) Find the fraction:

   0xA5 22 4E →binary representation is: 1.010 0101 00100010 01001110
   Convert 0xA5 22 4E to decimal is: 10822222.0
   ***f'* = (10822222.0 - 2^{23}) / 2^{23} → 0.2901094 → *f'* = *f* – 1 →**
   → *f* = 1 + 0.290109 → 1.2901094

4) Therefore:

   2^ef* = 2^6*1.2901094 = 82.5670016 → remember the sign bit gives: - 82.5670016**
   So 0x85A5224E is equal to - 82.5670016.

WRITING TO THE CT335

Writing to the CT335 requires 9 bytes, lets now change the setpoint of setpoint 1 to 100.0:

- **Function code: 0x02 (Write)**
- **Variable code: 0x11 (setpoint 1)**
- **Data length: 0x04 (always 0x04)**
- **Data: 0x 85 48 00 00 (in Microchip float format)**
- **Checksum: 0xDA (see above for calculation method)**
- **Garbage byte: 0x00**

By sending: **0x02 11 04 85 48 00 00 DA 00** to the CT335, the master device can expect to get back: **0x62 02 11 04 <u>85 48 00 00</u> DA**. Where "85 48 00 00" is the data, and "DA" is the checksum.

0x62 is the garbage byte

Note: Writing any invalid or out of range data to the CT335 will be ignored.

READING FROM THE CT335

Reading from the CT335 also require 9 bytes, let's read setpoint 1 from the CT335:

- **Function code: 0x01 (read)**
- **Variable code: 0x11 (setpoint 1)**
- **Data length: 0x04 (always 0x04)**
- **Data: 0x00 00 00 00**
- **Checksum: 0x14**
- **Garbage byte: 0x00**

If the data of setpoint 1 is 100.00, then the master device can expect to get back:
0x 62 01 11 04 85 48 00 00 D9

Garbage byte

NOTES:

Worldwide Headquarters
7300 Commerce Lane
Minneapolis, MN 55432 USA
Tel: 1.763.571.3121
Fax: 1.763.571.0927
sales@minco.com
www.minco.com

European Headquarters
Usine et Service
Commercial, Z.I.
09310 Aston, France
Tel: (33) 5 61 03 24 01
Fax: (33) 5 61 03 24 09

Asia Pacific Headquarters
20 Science Park Road
#02-31 Teletech Park
Singapore Science Park II
Singapore 117674
Tel: (65) 6511 3388
Fax: (65) 6511 3399

ISO 9001:2000 / AS9100B

(B)
7.8.2016