HTD2230-12C Digital Temperature and Humidity Sensor

[Highlighting strengths]

■Full range of temperature and humidity digital direct output, providing development software package;

■Solid polymer structure of the patent;

■ Outstanding dustproof performance;

■High humidity and dew prevention heating control;

■Better than ±3%RH and ±0.25℃ full interchange;

■ 0. 03%RH and 0.01℃ resolution;

■Fast temperature and humidity response, high humidity recovery; ■ Excellent long-term stability;

■Low energy consumption, small size (11mm×27mm);

■High performance price ratio;

■Not sensitive to light, no need to avoid light;

■The calibration data of each sensor is stored in the sensor, which is not lost in the case of power failure, and the accuracy is not affected by the transmission distance.

[Typical application]

■ Air conditioner, humidifier, dehumidifier

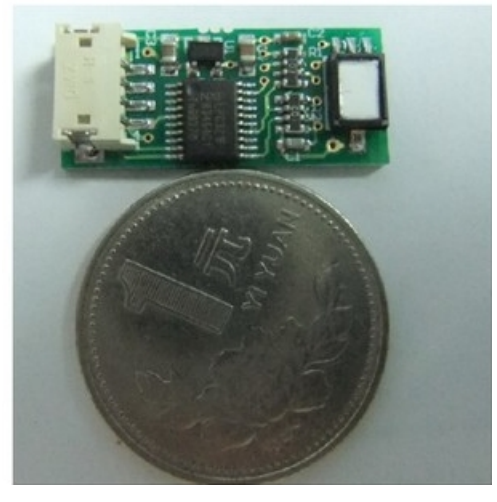
■ Humidity control, humidity transmitter

■ Humidity meter, humidity recorder

■ data logging

■ copycat

■ Clock, weather forecast barometer



[Product Description]

HTD2230-I2C provides OEM customers with accurate and reliable digital measurement of temperature and humidity.

HTD2230-I2C contains an I2C digital interface, which directly outputs the temperature compensated humidity, temperature, dew point and check CRC information according to the I2C instruction protocol.

Users do not need to carry out secondary calculation of digital output, nor do they need to compensate humidity for temperature, so as to obtain accurate temperature and humidity information; the free development kit can be embedded in user code, making it easy for users to master and use the sensor.

The special design of the sensor ensures that the humidity measurement in high humidity, dust, dirt and other harsh environment has obvious advantages over similar products.

In environments with fog, haze, rain, or condensation where humidity approaches saturation, sensors become less sensitive to their surroundings when exposed to moisture. This makes accurate humidity measurement particularly challenging in such conditions. To address this, specialized heating functions are integrated into the sensors to dehumidify and warm the environment. Under normal humidity measurement conditions, the default detection speed is 1 Hz (125ms per cycle), while the sensor remains in low-power standby mode during idle periods. In foggy, hazy, rainy, or condensation environments, the system accelerates detection to full-speed mode (8 cycles per second). During this high-speed operation, the sensor's microprocessor operates at full capacity, increasing current consumption to 4-5mA. This continuous environmental heating effectively reduces surrounding humidity levels, thereby significantly lowering the risk of condensation on sensor probes.

Most humidity sensors require installation in clean environments or use specialized dust covers, which inherently increase their size and cost. The HTD2230-I2C circuit employs innovative hardware and software designs that reduce dust and dirt interference by up to one-tenth compared to similar products. Additionally, the sensor's built-in dust-proof membrane minimizes contamination effects on accurate humidity measurement.

The HTD2230-I2C digital calibration information is stored in the Flash chip. For batch users, the calibration and calibration functions of the sensor are open. Users can perform batch and rapid secondary calibration or calibration on the sensor by using special instructions.

【 interface specification 】

HTD2230-I2C digital temperature and humidity sensor is used as a slave, and the communication between the host (user) and the slave is carried out by I2C bus.

Since humidity is highly sensitive to temperature, unless operating in high-humidity environments prone to condensation, it's crucial to strictly control the average power consumption of sensors. This helps minimize the impact of sensor heat generation on surrounding humidity levels. Therefore, sensors should automatically enter sleep mode when not measuring temperature and humidity to reduce power usage. By adjusting the sensor's temperature/humidity output refresh rate – essentially setting the automatic wake-up interval – we can effectively manage power consumption.

The HTD2230-I2C operates in active request mode, where the sensor automatically refreshes temperature and humidity readings at a user-specified rate. Upon completing data refresh, the system outputs a low signal on the SDA line to indicate readiness. When the host reads data from the slave (sensor), it must first pull the SCL line low to acknowledge the slave. Upon receiving this acknowledgment, the slave releases the SDA line to restore a high signal. This enables standardized I2C protocol communication between the host (user device) and the slave (sensor). It should be noted that the system determines readiness for temperature and humidity data transmission through SDA output.

From now on, the user must complete the reading of the temperature and humidity information frame within 125ms, otherwise the data will be lost.

pin	name	explain
1	VDD	Power supply, typical 5.0VDC or 3.3VDC
2	SCL	Serial clock line, which is also the response signal line for the host to respond to the sensor to read the data
3	SDA	Serial data line, which is also the digital temperature and humidity conversion ready (low level) signal line
4	GND	the earth
		RHT

【 qualification 】

parameter		symbol	minimum	typical case	maximum	unit
humidity	measuring range	RH	0		100	%RH
	certainty of measurement			±3	±5	
	resolution ratio			0.03	0.05	
temperature	measuring range	Tcc	-45		85	℃
	certainty of measurement			±0.25	±0.5	
	resolution ratio			0.01	0.02	
Power Supply Voltage		Vs	3.0	3.3/5.0	5.5	VDC
Recovery time (after 150 hours of storage at 100%RH)		tr		10		S
Humidity lag				±1		%RH
long term stability				±0.5		%RH/ year
Average current consumption	1 time/2 seconds	Is			0.25	mA
	1 time per second				0.5	
	8 times per second				4	
Peak load current		Ipeak			4	mA
Storage environment	temperature	Tstg	-40		85	℃

【 reliability test 】

HTD2230-I2C has passed the following tests: vibration, impact, high temperature, high humidity, ESD, etc.

【 output format 】

When the host sends a command to read the temperature and humidity information of the sensor (slave), the sensor outputs a data frame of 6 bytes. **Temperature and humidity output data frame:**

humidity	temperature	continue to have	CRC
2 bytes	2 bytes	2 bytes	1 byte

Humidity (2 bytes) Output format:

high byte

sign bit	bit6	bit5	bit4	bit3	bit2	bit1	bit0
S	2^6	2^5	2^4	2^3	2^2	2^1	2^0

lower byte

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	2^1	2^2	2^3	2^4
2^5	2^6	2	2^8								

Humidity values (% R H)	Binary output (Binary)	Hex output (16-bit)
100.0	0110010000000000	6400
97.3	0110000101001100	614C
75.3	0100101101001100	4B4C
55.0	0011011100000000	3700
32.8	0010000011001100	20CC
11.3	0000101101001100	0B4C
0.0	0000000000000000	0000

Temperature (2 bytes)

Output format: high byte

sign bit	bit6	bit5	bit4	bit3	bit2	bit1	bit0
S	2^6	2^5	2^4	2^3	2^2	2^1	2^0

lower byte

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
2^{-1}	2^{-2}	2^3	2^4	2^5	2^6	2−7	2^8

temperature scale （℃）	Binary output (Binary)	Hex output (16-bit)
+85	0101010100000000	5500
+55.5	0011011110000000	3780
+25.0625	0001100100010000	1910
+10.125	0000101000100000	0A20
+0.5	0000000000001000	0080
0	0000000000000000	0000
-0.5	1111111110000000	FF80
-10.125	1111010111100000	F5E0
-25.0625	1110011011110000	E6F0
-45.0	1101001100000000	D300

CRC check (1 byte):

The last byte of the temperature and humidity data output frame contains an 8-bit CRC checksum for the first six bytes, which verifies the integrity of the received data. When cables between the host and sensor are extended, using this CRC checksum enhances measurement anti-interference capabilities. The CRC algorithm is detailed in the provided C51 software package.

Sensor address (1 byte):

The sensor's I2C address is 56h (hex

Send the command:

There are two commands that users can use for the sensor:

64h (hexadecimal--read temperature and humidity data frames

82h (hexadecimal-Adjusts the refresh rate of temperature and humidity output (0: 8 times per second, 1: 4 times per second **2: 2 times per second, 3: 1 time per second, 4: 1 time per second)** Default value is 1 time per second

Refer to the provided C51 software package for the command generation mode.

HTD2230-12C User Guide

1. Introduction to 12G bus

The interface between HTD2230-I2C and the microcontroller is I2C serial bus line. Here, I will briefly introduce I2C General Agreement Standards. Due to space constraints, the full text of the agreement cannot be included.

For further information on the contents of the ministry, please refer to the website of the Fiji Company(http/www.semiconductorsphilips.com/), Or is it a book by Professor He Lemin entitled I2C Total Line Application System Design". The next section also provides an I2C bus software package written in C51 for reference.

1.1 12C General line overview

Over two decades ago, Philips developed a simple two-wire serial communication bus known as Inter-IC (I2C). Today, the I2C bus has become the industry-standard solution for embedded applications, widely used in various microcontroller-based professional, consumer, and telecommunications products as a control, diagnostic, and power management interface. Multiple I2C-compatible devices can communicate through a single bus without requiring additional address translators. The simplicity of this two-wire serial communication design has been key to its rapid rise as an industry-standard solution.

1.2 12C bus signal lines

The I2C bus requires only two signal lines: the serial data line (SDA) and the serial clock line (SCL). Typically, I2C devices feature both SDA and SCL pins with drain (or collector) open outputs. In practical applications, these signal lines must be connected with pull-up resistors (Rp). Pull-up resistor. The pull-up resistor is typically rated between 3 to 1 0kΩ. The advantages of open-collector configuration are: When the main line is idle, both signal lines maintain high voltage levels, consuming almost no current; it offers excellent AC-DC conversion capability. When connected to a 5V power supply, it can interface with 5V logic components, while connecting to a 3V power supply allows connection with 3V logic components. Due to its open-collector design, SDA/SDA and SCL/SCL between different components can be directly connected without requiring additional conversion circuits.

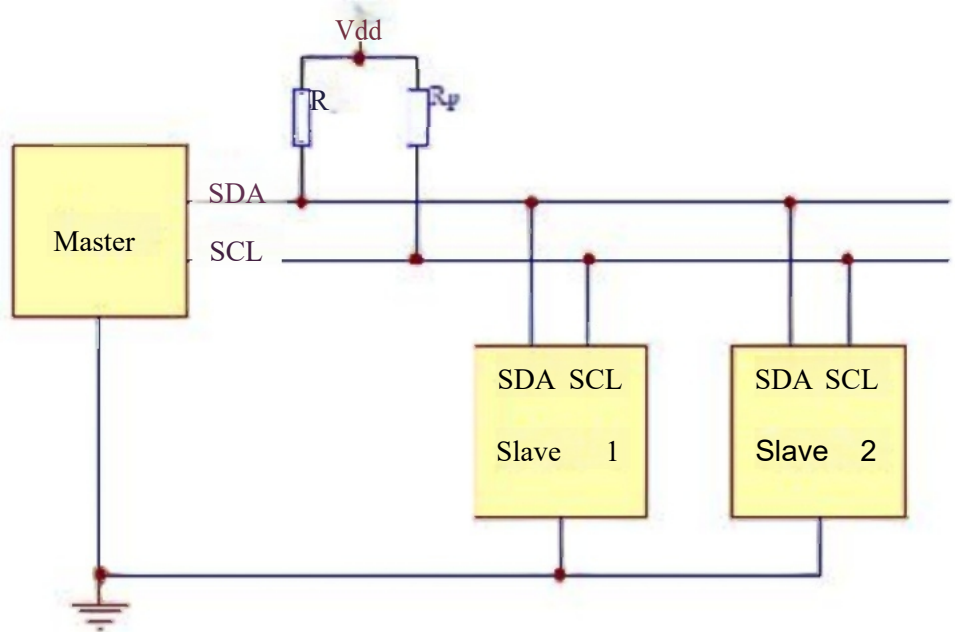


Figure 1.1 I2C Bus Signal Connection Diagram

1.3 Basic concepts of 12C bus

- Transmitter (transmitter): A device that sends data to the bus;
- Receiver: A device that receives data from a bus;
- Master: a device that initializes transmission, generates clock signals, and terminates transmission;
- Slave: A device addressed by the host.

The I2C bus is a bidirectional transmission line, where both the master and slave can act as transmitter and receiver respectively. When transmitting data, the master functions as the transmitter while the slave acts as the receiver. Conversely, when retrieving data from the slave, the master becomes the receiver while the slave serves as the transmitter.

1.4 12C bus data transfer rate

The communication rate of the I2C bus is controlled by the host, which can be fast or slow. However, the maximum rate is limited. The transmission rate of data on the I2C bus can reach 100Kb / s in the standard mode.

1.5 Data Validity on the 12C Bus

The electrical state of the SDA signal must remain stable during the high-level period of the clock line SCL. SDA's state is only permitted to change when SCL is in a low-level state, except at the start and end of I2C bus cycles. While some other serial bus protocols may specify data validity at clock signal edges (rising or falling), I2C bus operates on a level-based validity principle.

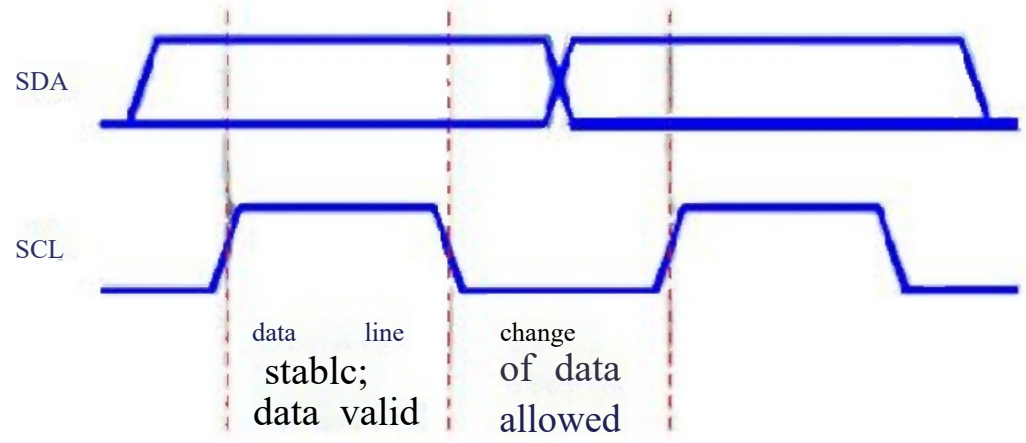


Figure 1.2 Schematic Diagram of Data Validity on I2C Bus

1.6 Start and Stop Conditions (START and STOP Conditions)

Initiating condition: The SDA generates an initiating condition when transitioning from a high to a low level during the SCL's high-level state. The bus enters a busy state after this condition is triggered. This condition is commonly abbreviated as S.

Stop condition: A stop condition is generated when SDA transitions from low to high during a high-level SCL. The bus enters an idle state after the stop condition is generated. The stop condition is abbreviated as P.

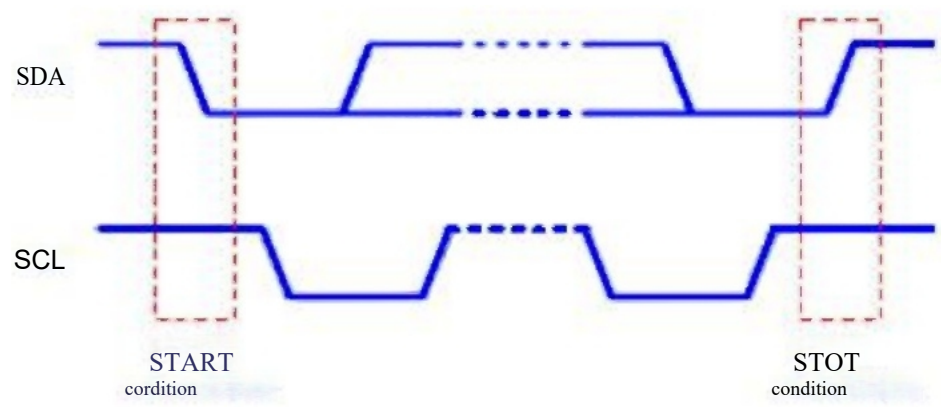


Figure 1.3 Schematic Diagram of I2C Start and Stop Conditions

1.7 Slave Address (Slave Address)

The I2C bus requires neither an additional address decoder nor a chip select signal. Multiple devices with I2C interfaces can be connected to the same bus, distinguished by their device addresses. The host controller (the master device) operates without requiring device addresses, while all other devices function as slaves and must have unique addresses. It is crucial to ensure that all slave addresses on the same I2C bus are uniquely identifiable with no duplicates, as this would prevent proper operation of the I2C bus. Typically, a slave address consists of seven address bits and one read/write flag (R/W). The seven address bits occupy the high 7-bit positions, with the read/write flag positioned at the end. A value of 0 in the read/write flag indicates the host will write data to the slave, while a value of 1 signifies the host will read data from the slave.

Special Note: Since the HTD2230-I2C digital temperature and humidity sensor is an active output type, the SDA and SCL lines are both ready and response functions. Therefore, no other I2C devices can be connected to the SCL and SDA buses.

1.8 The basic format of data transmission

The I2C bus transmits data in bytes. Each byte sent over the SDA line must be 8 bits long, with no fixed number of bytes per transmission. Data is transferred in reverse order – starting with the most significant bit (MSB, 7th position) and ending with the least significant bit (LSB, 0th position). Additionally, each byte is followed by a response bit called the " acknowledgment".

1.9 Should answer (Acknowledge)

During I2C bus data transmission, each byte is accompanied by an acknowledgment status bit. The receiver's acknowledgment status informs the sender of data reception. While the host generates clock pulses for these bits, their data states follow the "who receives, who generates" principle – meaning the receiver always produces acknowledgments. When transmitting data to a slave device, the slave generates these bits; conversely, when receiving from a slave, the host generates them. I2C bus specifications define: A 0 in the acknowledgment bit indicates a receiver acknowledgment (ACK), abbreviated as A; A 1 signifies a non-acknowledgment (NACK), also abbreviated as A. After transmitting the least significant bit (LSB), the sender must release the SDA line by pulling it high.

When the receiver completes receiving all bytes or can no longer receive more data, it must generate a non-response to notify the sender. If the sender detects this non-response status, transmission should be terminated immediately.

1.10 A diagram of the basic data transfer format

Figure 1.4 Basic Format for Sending Data from Host to Slave

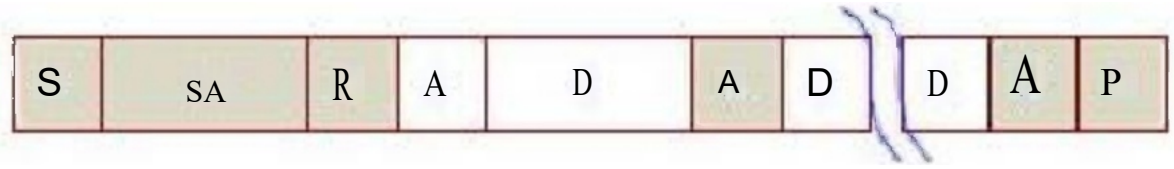


Figure 1.5 Basic Format for Host Receiving Data from Slave

In FIGS. 1.4 and 1.5, the meaning of each symbol is:

- S: Start position (START);
- SA: Slave address (7-bit slave address);
- W: Write the flag, 1 bit write the flag;
- R: Read the flag (Read), 1 bit read flag;
- A: Acknowledge, 1 bit acknowledgment;
- A: Non-acknowledgment (Not Acknowledge), 1 bit non-response;
- D: Data, each data must be 8 bits;
- P: Stop (STOP);
- Shadow: Signal generated by the host;
- No shadow: Signal generated by the slave.

It should be noted that unlike the scenario in Figure 1.5, when transmitting the final byte of data from the host to the slave in Figure 1.4, the slave may either respond or not. However, regardless of the response, the host can still generate a stop condition. If the host detects a non-response from the slave during data transmission (including the slave's address), it should immediately cease the transfer.

1.11 A timing diagram for transmitting a byte of data

To better understand the fundamental data transfer process on I2C buses, the following timing diagram illustrates a basic 1-byte transmission. In Figures 1.6 and 1.7, two SDA signal lines are depicted: one generated by the master device and another by the slave device. In reality, the SDA signal lines of both master and slave devices are always connected as a single SDA line. This dual-line representation helps clarify the distinct operational behaviors between master and slave devices on I2C buses.

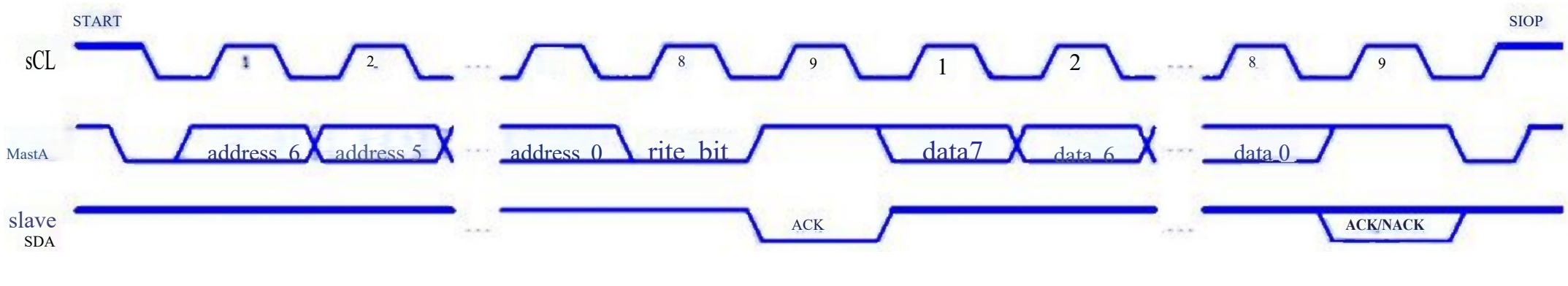


Figure 1.6 Schematic Diagram of a Byte of Data Sent from the Host to the Slave

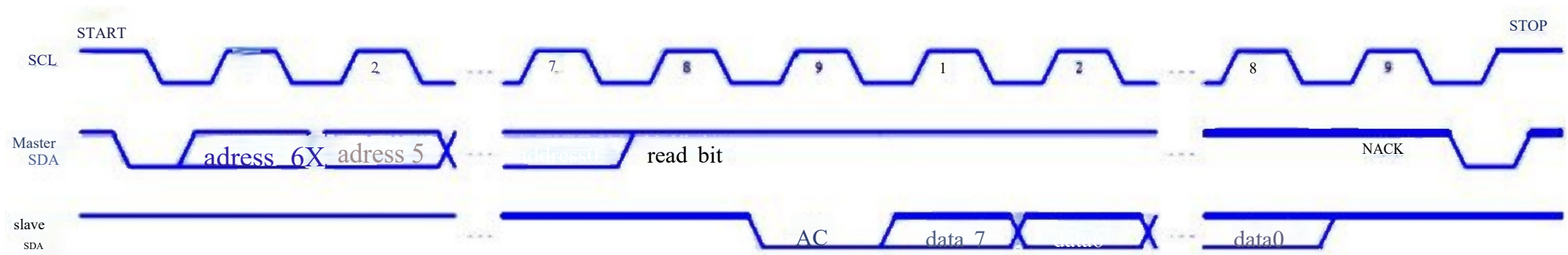


Figure 1.7 Schematic Diagram of a Host Receiving One Byte of Data from a Slave

1.12 A Timing Diagram for Transmitting Multiple Byte Data

It is also easy to understand the situation where the host continuously sends or receives multiple bytes of data from the slave. The following is a timing diagram.

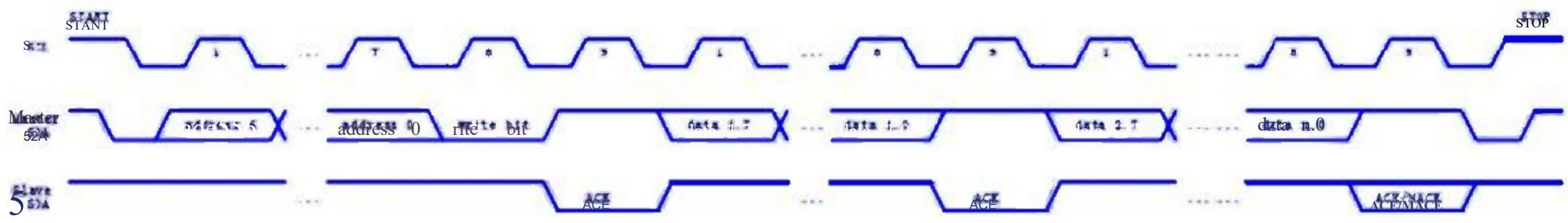


Figure 1.8 Schematic Diagram of Continuous Transmission of Multiple Bytes from Host to Slave

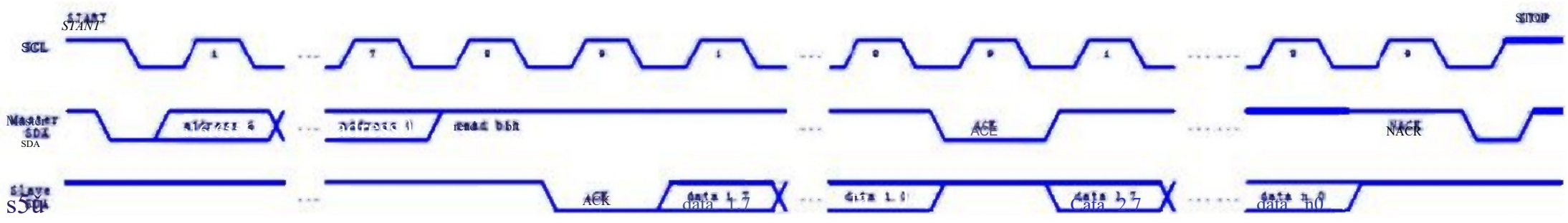


Figure 1.9 Schematic Diagram of Continuous Reception of Multiple Byte Data by Host from Slave

1.13 Repeat the Start Condition (Repeated START Condition)

When communicating between a host and a slave device, it is sometimes necessary to switch the data transmission direction. For example, when accessing an EEPROM with an I2C bus interface, the host first sends address information to the memory cell (transmitting data) before reading the stored content (receiving data). When switching the data transfer direction, there's no need to first generate a stop condition before initiating the next transmission; instead, the start condition can be generated again directly. The act of generating a start condition again when the I2C bus is already busy is referred to as repeated start condition, commonly abbreviated as Sr. While normal start conditions and repeated start conditions show no difference in physical waveform, their distinction lies solely in logic. During multi-byte data transfers, repeated start conditions must be used whenever the data transmission direction changes.

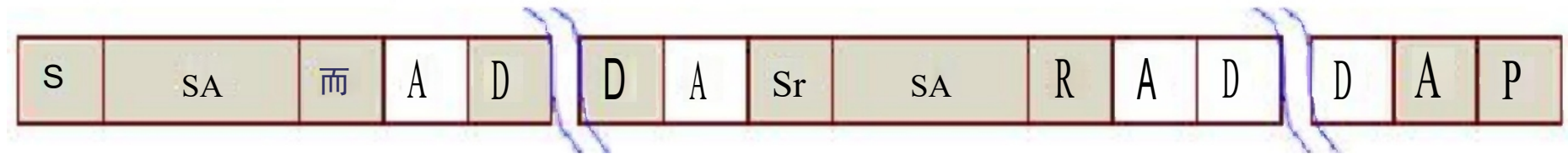


Figure 1.10 Schematic Diagram of Multi-Byte Data Transfer Format With Repeated Start Conditions

Figure 1.10 illustrates a format for multi-byte data transmission with repeated start conditions, where the symbols' meanings correspond to those described in Section 1.10. Particular attention should be paid to the application of the repeated start condition Sr. Readers may wish to construct their own corresponding timing diagrams if they find this approach helpful.

1.14 Non-Subaddress Device and Subaddress Device

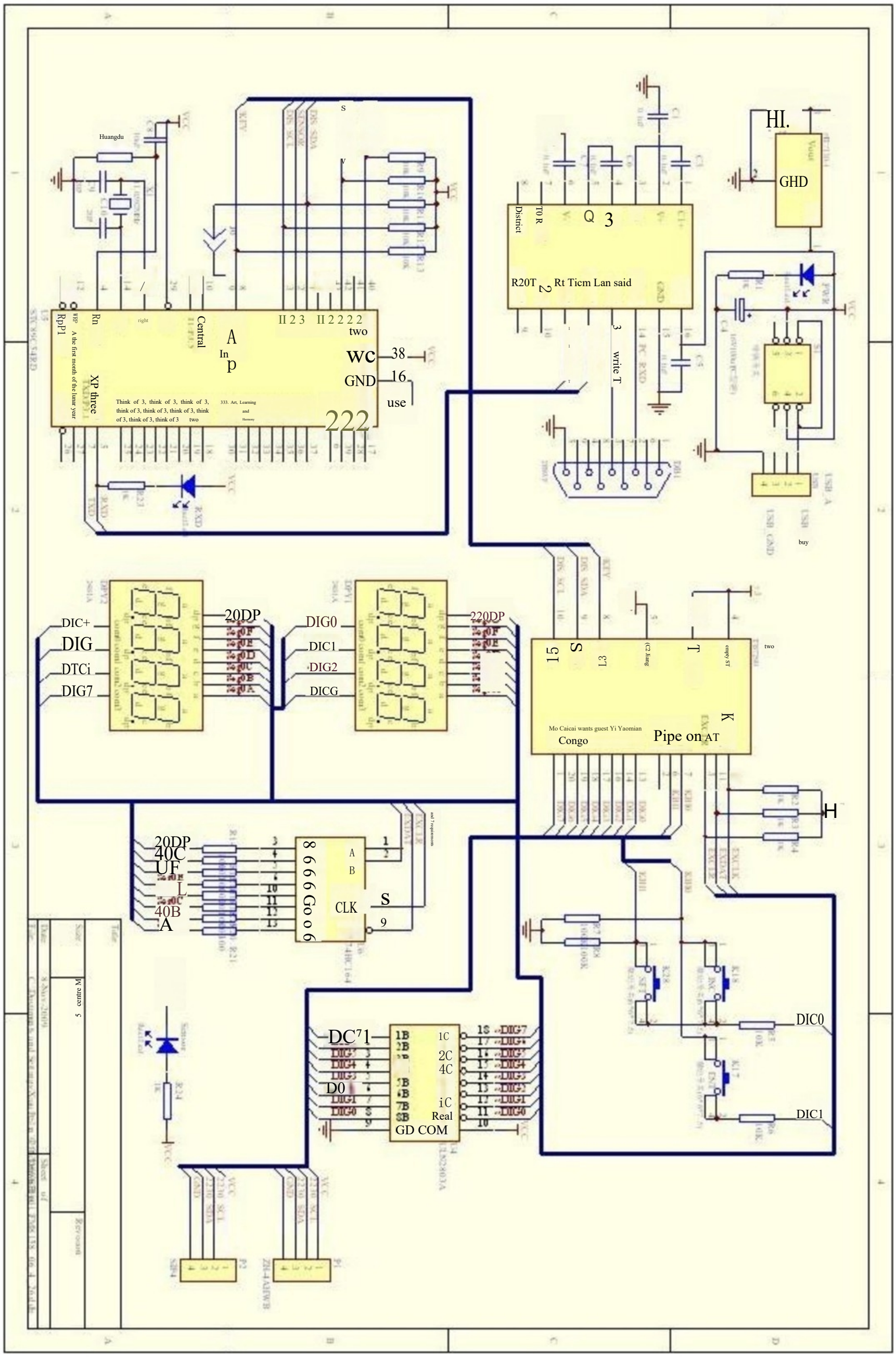
Devices with I2C buses may have both slave addresses (Slave Address) and sub-addresses. The slave address is the address assigned by the host to the device on the I2C bus, while the sub-address refers to the internal addressing of different components or memory units within the device. For example, E2PROMs with I2C bus interfaces are typical examples of devices with sub-addresses. However, some devices (rare exceptions) have simpler internal architectures and may lack sub-addresses, retaining only the necessary slave address. Similar to slave addresses, sub-addresses are transmitted as standard data with identical transmission formats. The distinction between addressing and data transmission relies on specific logical agreements between transceivers. Sub-addresses must consist of an integer number of bytes, ranging from single-byte (8-bit sub-address) to double-byte (16-bit sub-address), and may extend beyond three bytes depending on device specifications.

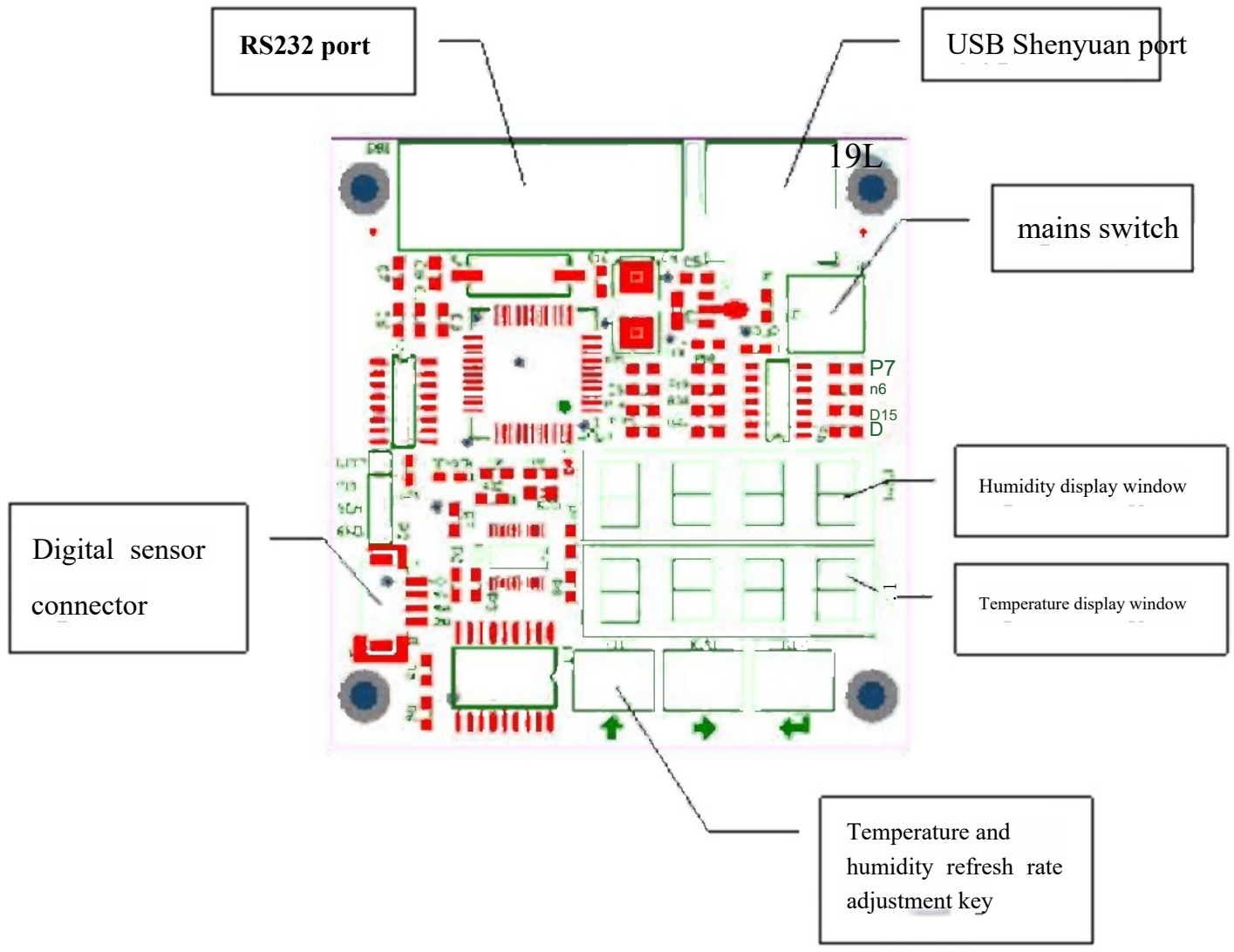
In the I2C bus package in Chapter 3, both address-free and address-within devices have been considered.

2.HTD2230-12G Demo Board

HTD2230-I2C The single chip used in the demo board is STC89C54RD+, Compatible with 8 9C52, no need for a dedicated programming tool can be programmed by ISP, 编

The program package can be downloaded from www.MCU-Memory.com.





3.HTD2230-12C Digital Temperature and Humidity Sensor Keil G51 Development Software Package

3 1 Driver software package

Since both the HTD2230-I2C digital temperature and humidity sensor and the chip for display and keyboard management are I2C devices, the 89 C52-compatible microcontroller C51 driver program's I2C analog software package consists of I2C_HTDDriverH and I2C_HTD_DRIVER.C I2C_DISDriverH, I2C_DIS_DRIVER.C respectively.

Header files

I2C_HTD_DRIVERH、I2C_DIS_DRIVERH Includes the definition of the I/O interface for signal lines SDA and sCL and the declaration of user functions, C language file I2C_HTD_DRIVERC, I2C_DIS_DRIVER.C represents the concrete implementation of these functions. This is a simplified version using C51 to simulate the I2C bus protocol, focusing solely on master-slave mode without considering multi-master configurations or clock synchronization issues. For a clearer understanding of the program's details, please refer to Philips' relevant protocol standards.

3.1.1 HTD2230_I2C driver

- Copy the files "I2C_HTDDriverH" and "I2C_HTDDriverC" together to your project folder;
- According to the actual circuit, modify the definition of two signal lines SCL and SDA on the I2C bus;
- The speed of the I2C bus is adjusted by macro definition I2C_HTDDELAY_VALUE to meet the actual needs;
- Add the file "I2C_HTDDriverC" to the project and include the header file "I2C_DRIVER.H" where needed;
- At the beginning of the main() function, the initialization function I2C_HTD_Init() should be called once;
- I2C_HTD_Puts (and I2C_HTD_Gets () are comprehensive read/write functions for the I2C bus. Please read the comments before using them;
- All global property identifiers start with "I2C_HTD_" to effectively avoid naming conflicts;
- Note: The slave address is expressed as a 7-bit pure address, without read/write bits. That is, the 0th to 6th bits are addresses and the 7th bit is invalid. This may differ from other programs.

3.1.2 TB7290 display and keyboard management driver

- Copy the files "I2C_DIS driver H" and "I2C_DIS driver C" together to your project folder;
- According to the actual circuit, modify the definition of two signal lines SCL and SDA on the I2C bus;
- The I2C DISDELAY_VALUE macro is defined to adjust the speed of the I2C bus to meet the actual needs;
- Add the file "I2C_DIS_DRIVER.C" to the project and include the header file "I2CDriverH" where needed;
- At the beginning of the main) function, the initialization function I2C_DIS _Init() should be called once;
- I2C_DIS_Puts(and I2C_DIS_Gets(are the integrated read/write functions of the I2C bus. Please read the notes before using them;

■ All global property identifiers start with "I2C_DIS_" to effectively avoid naming conflicts;

■ Note: The slave address is expressed as a 7-bit pure address without read/write bits, i.e. bits 0 to 6 are addresses and bit 7 is invalid.

This may differ from other programs. See the randomly provided documentation notes for specific source code and instructions.

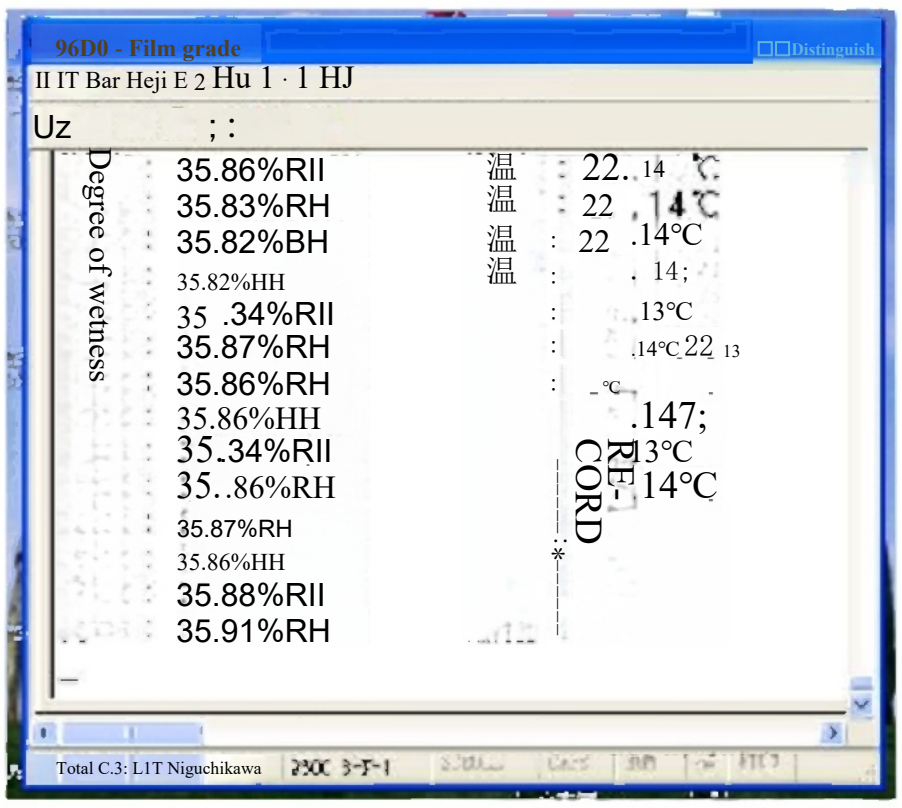
3.2 HTD2230-12C Demo Board Keil C51 Application Software Package

The file HTD2230.H defines the commands, structures, and function declarations for reading digital temperature and humidity data. It HTD2230.C also provides corresponding functions to execute these commands.

See the randomly provided documentation notes for details of the source code and instructions.

Set the PC's Super Terminal communication baud rate to 9600, with data bits (D: 8, parity (P): None, stop bits (S): 1, and data flow control (E): None; The Demo board is configured by the PC. The USB port is powered and connected to the PC RS232, so that it can communicate with the PC, and the temperature and humidity data can be displayed on the screen of the PC.

You can run the super terminal file with predefined Settings directly: 9600



4. Document Information

4.1 reference documentation

[1] I2C bus specification Philips Semiconductor Company· 2000

[2] Ma Zhongmei, Qi Jun, Liu Bin, Ma Yan. "C Language for Microcontrollers: A Guide to Windows Environment Programming". Beijing University of Aeronautics and Astronautics Press. 2003

[3] He Limin. Design of I2C Bus Application System. Beijing University of Aeronautics and Astronautics Press, 1995

The 2009-11-8 version is based on the 2009-9-20 Beta:

1、 The HTD2230-I2C digital temperature and humidity sensor demo board is designed. The demo board uses the single chip microcontroller compatible with 89C52, STC89C52RD+, and can download the HTD2230-I2C demo program through the RS232 port of PC without a special programmer.

- ■ It can display the HTD2230-I2C humidity and temperature values, and adjust the refresh frequency of temperature and humidity through the button.
- ■ Provide the Demo board schematic and detailed description.
- ■ Provides the full source code for demonstrating the HTD2230-I2C digital temperature and humidity sensor, which can be embedded in your target code as needed.
- ■ The TB7290 chip, designed for display and keyboard management, interfaces with the STC89C52RD+ microcontroller via a standard I2C bus. This setup enables control of up to 328-bit common-cathode digital tubes, 16 single-function keys, and 8 multiplexed keys. Through the HTD2230-I2C demo board, users can not only master reliable communication between the 8051 microcontroller and I2C devices but also learn to design display systems for multiple digits (typically over 8 common-cathode tubes) and implement keyboard key processing solutions.

2. Modify the original structure:

```
typedef   struct{
        unsigned        UpdateRate;        // 2 bytes
        unsigned char   CRC;               //CRC of the first two bytes
}UPDATE_RATE_STRUCT;
```

be revised as：

```
typedef   struct{
        unsigned char   UpdateRate;        // 1 byte
        unsigned char   CRC;               CRC of the first byte
}UPDATE_RATE_STRUCT;
```

2009-9-20 Beta version

Address: Room 2007, South Block, Dingcheng International Building, No.7 Zhonghang Road, Futian District, Shenzhen Zip code: 518031
Mobile phone: 13662266995 Ma Shaoliang Tel: 0755-83951311 Fax: 0755-83952401
Official website: cn-sensor.com Email: jackson@jmcomponents. com