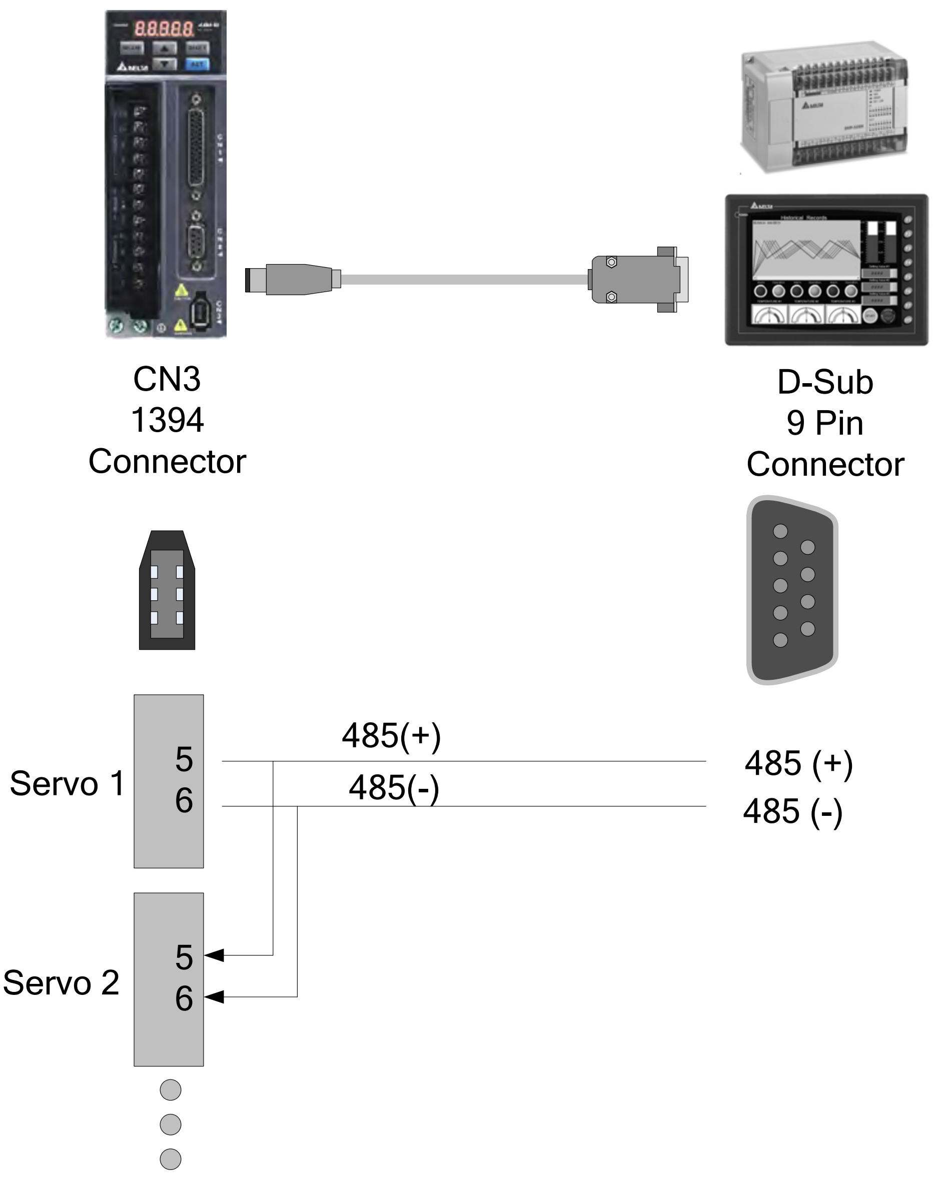
RS-485

■ **Connection**



NOTE

1) The maximum cable length is 100m (39.37inches) when the servo drive is installed in a location where there are only a few interferences. Please note, RFI / EME noise should be kept to a minimum, communication cable should kept apart from high voltage wires. If a transmission speed of 38400bps or greater is required, the maximum length of the communication cable is 15m (50ft.) which will ensure the correct and desired baud rate.
2) The number shown in the pervious figure indicates the terminal number of each connector.
3) The power supply should provide a +12V and higher DC voltage.
4) Please use a REPEATER if more than 32 synchronous axes are required.
5) For the terminal identification of CN3, please refer to Section 3.5.

8.2 Communication Parameter Settings

The following describes the communication addresses for the communication parameters.

Parameters P3-00, P3-01, P3-02 and P3-05 are required to be set for any communication between the servo drives and motors. The other optional parameters such as P3-03, P3-04, P3-06, P3-07 and P3-08 are used depending on different customer demands and applications.

For optional communication parameters, please refer to the Chapter 7.

P3-00●	**ADR**	**Communication Address Setting**		**Address: 0300H 0301H**
	Operation Interface:	Keypad / Software	Communication	Related Section: Section 8.2
	Default:	0x7F		
	Control Mode:	ALL		
	Unit:	N/A		
	Range:	0x01 ~ 0x7F		
	Data Size:	16-bit		
	Display Format:	Hexadecimal		

Settings: This parameter is used to set the communication slave address in hexadecimal format.

Display	0	0	Y	X
Range	-	-	0 ~ 7	0 ~ F

X: Axis number which indicates the value must be within the range from 0 through F.

Y: Group number which indicates the value must be within the range from 0 to through 7

When using RS-232/485 communication, this parameter is used set the communication address in hexadecimal format. If the AC servo drive is controlled by RS-232/485 communication, each drive (or device) must be uniquely identified. One servo drive only can set one address. If the address is duplicated, there will be a communication fault. This address is an absolute address which represents the servo drive on a RS-232/485 network.

Please note:

1. When the address of host (external) controller is set to 0xFF, it is with auto-respond function. Then, the servo drive will receive from and respond to host (external) controller both no matter the address is matching or not. However, the parameter P3-00 cannot be set to 0xFF.

P3-01	BRT	Transmission Speed		Address: 0302H 0303H
	Operation Interface:	Keypad / Software	Communication	Related Section: Section 8.2
	Default:	0x0033		
	Control Mode:	ALL		
	Unit:	bps		
	Range:	0x0000 ~ 0x0055		
	Data Size:	16-bit		
	Display Format:	Hexadecimal		

Settings: This parameter is used to set the baud rate and data transmission speed of the communications.

	0	Z	Y	X
COM Port	-	-	RS-485	RS-232
Range	0	0	0 ~ 5	0 ~ 5

Settings:
0: Baud rate 4800 (data transmission speed: bits / second)
1: Baud rate 9600 (data transmission speed: bits / second)
2: Baud rate 19200 (data transmission speed: bits / second)
3: Baud rate 38400 (data transmission speed: bits / second)
4: Baud rate 57600 (data transmission speed: bits / second)
5: Baud rate 115200 (data transmission speed: bits / second)

P3-02	PTL	Communication Protocol		Address: 0304H 0305H
	Operation Interface:	Keypad / Software	Communication	Related Section: Section 8.2
	Default:	0x0066		
	Control Mode:	ALL		
	Unit:	N/A		
	Range:	0x0000 ~ 0x0088		
	Data Size:	16-bit		
	Display Format:	Hexadecimal		

Settings: This parameter is used to set the communication protocol. The alphanumeric characters represent the following: 7 or 8 is the number of data bits; N, E or O refers to the parity bit, Non, Even or Odd; the 1 or 2 is the numbers of stop bits.

	0	Z	Y	X
COM Port	-	-	RS-485	RS-232
Range	0	0	0 ~ 8	0 ~ 8

0: Modbus ASCII mode, <7,N,2>
1: Modbus ASCII mode, <7,E,1>
2: Modbus ASCII mode, <7,O,1>
3: Modbus ASCII mode, <8,N,2>
4: Modbus ASCII mode, <8,E,1>
5: Modbus ASCII mode, <8,O,1>
6: Modbus RTU mode, <8,N,2>
7: Modbus RTU mode, <8,E,1>
8: Modbus RTU mode, <8,O,1>

P3-05	**CMM**	**Communication Selection**		**Address: 030AH 030BH**
	Operation Interface:	Keypad / Software	Communication	Related Section: Section 8.2
	Default:	1		
	Control Mode:	ALL		
	Unit:	N/A		
	Range:	0x00 ~ 0x01		
	Data Size:	16-bit		
	Display Format:	Hexadecimal		

Settings: RS-232 Communication interface selection
0: RS-232 via Modbus communication
1: RS-232 upon ASDA-Soft software

8.3 MODBUS Communication Protocol

When using RS-232/485 serial communication interface, each ASDA-B2 series AC servo drive has a pre-assigned communication address specified by parameter "P3-00". The computer then controls each AC servo drive according to its communication address. ASDA-B2 series AC servo drive can be set up to communicate on a MODBUS networks using on of the following modes: ASCII (American Standard Code for Information Interchange) or RTU (Remote Terminal Unit). Users can select the desired mode along with the serial port communication protocol in parameter "P3-02".

■ **Code Description:**

ASCII Mode:

Each 8-bit data is the combination of two ASCII characters. For example, a 1-byte data: 64 Hex, shown as '64' in ASCII, consists of '6' (36Hex) and '4' (34Hex).

The following table shows the available hexadecimal characters and their corresponding ASCII codes.

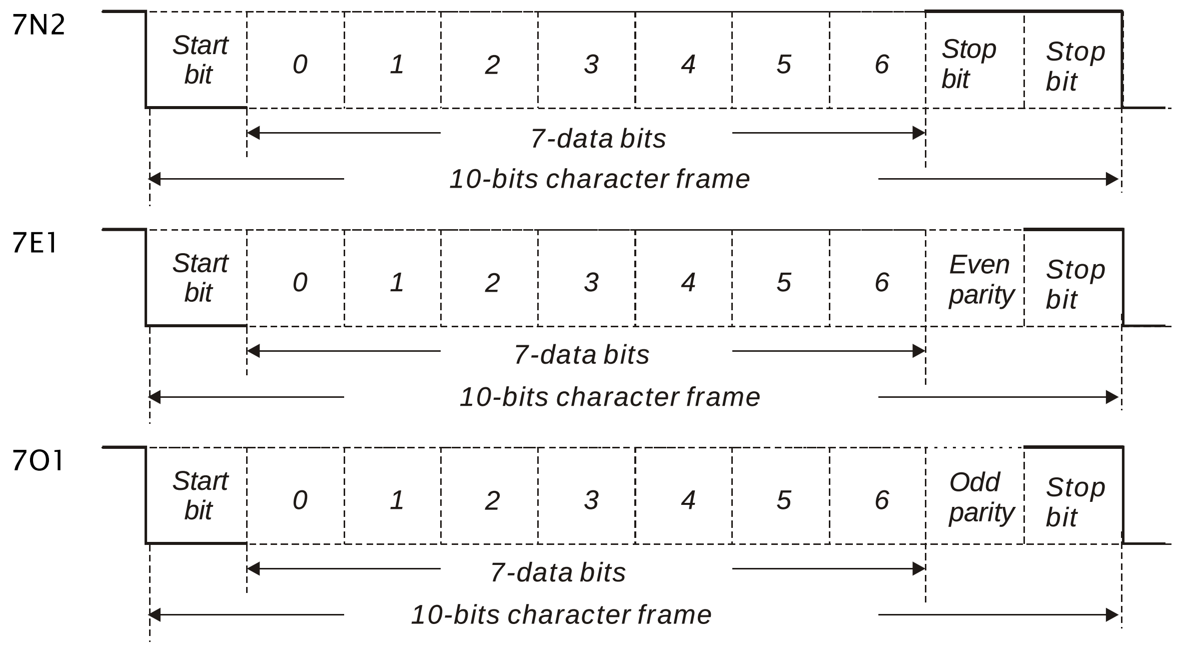
Character	'0'	'1'	'2'	'3'	'4'	'5'	'6'	'7'
ASCII code	30H	31H	32H	33H	34H	35H	36H	37H
Character	'8'	'9'	'A'	'B'	'C'	'D'	'E'	'F'
ASCII code	38H	39H	41H	42H	43H	44H	45H	46H

RTU Mode:

Each 8-bit data is the combination of two 4-bit hexadecimal characters. For example, a 1-byte data: 64 Hex.

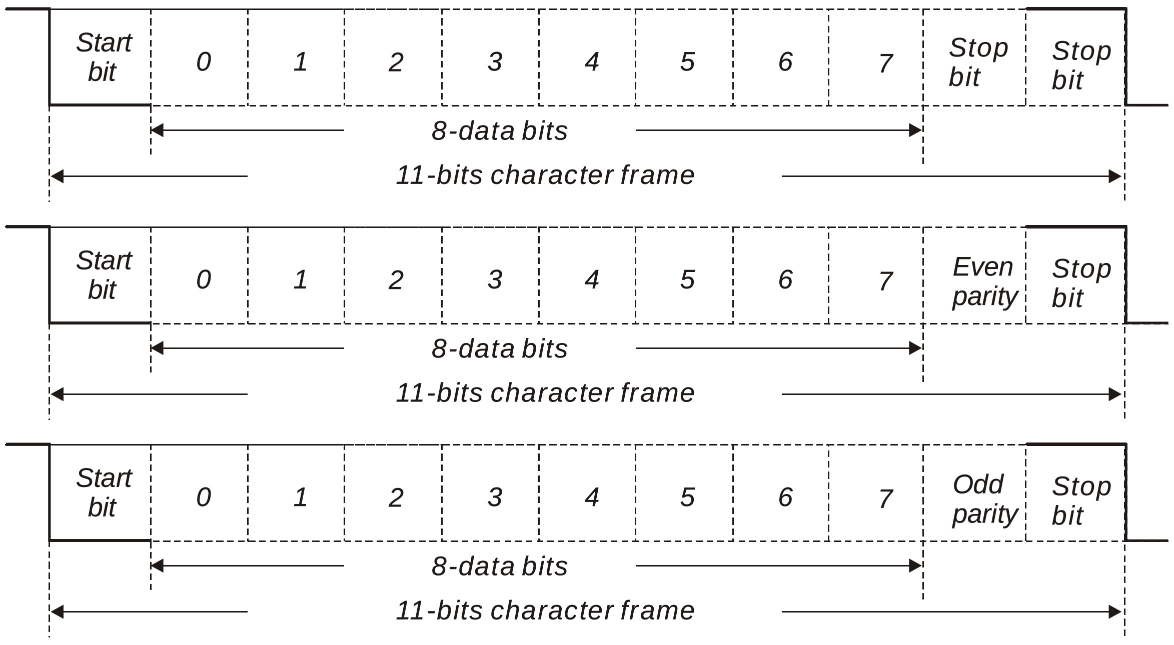
■ **Data Format:**

10-bit character frame (For 7-bit character)



11-bit character frame (For 8-bit character)

8N2



8E1

8O1

■ Communication Protocol:

ASCII Mode:

Start	Start character': ' (3AH)
Slave address	Communication address: 1-byte consists of 2 ASCII codes
Function	Function code: 1-byte consists of 2 ASCII codes
DATA(n-1)	Contents of data: n word = n x 2-byte consists of n x 4 ASCII codes, n≤10
…….	
DATA(0)	
LRC	LRC check sum: 1-byte consists of 2 ASCII codes
End 1	End code 1: (0DH)(CR)
End 0	End code 0: (0AH)(LF)

ASCII Mode: ':' character

ADR (Communication Address) consists of 2 ASCII codes and it ends in CR (Carriage Return) and LF (Line Feed)

CR (Carriage Return) is represented by ASCII number 13, and LF (Line Feed) is represented by ASCII number 10.

There are communication address, function code, contents of data, LRC (Longitudinal Redundancy Check) between start and end characters.

RTU Mode:

Start	A silent interval of more than 10ms
Slave address	Communication address: 1-byte
Function	Function code: 1-byte
DATA(n-1)	Contents of data: n word = n x 2-byte, n≤10
…….	
DATA(0)	
CRC	CRC check sum: 1-byte
End 1	A silent interval of more than 10ms

RTU Mode: A silent interval of more than 10ms

RTU (Remote Terminal Unit) starts from a silent signal and ends at another silent signal.

There are communication address, function code, contents of data, CRC (Cyclical Redundancy Check) between start and end characters.

DATA (Data Characters)

The format of data characters depends on the function code. The available command codes and examples for AC servo drive are described as follows:

Example 1

Function code: 03H, read N words (multiple words). The maximum value of N is 10.

For example, reading continuous 2 words from starting address 0200H of AC servo drive with address 01H.

ASCII Mode:

Command message:

Start	‘:’
Slave address	‘0’
	‘1’
Function	‘0’
	‘3’
Starting data address	‘0’
	‘2’
	‘0’
	‘0’
Number of data	‘0’
	‘0’
	‘0’
	‘2’
LRC Check	‘F’
	‘8’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

Response message:

Start	‘:’
Slave address	‘0’
	‘1’
Function	‘0’
	‘3’
Number of data (In Byte)	‘0’
	‘4'
Contents of starting data address 0200H	‘0’
	‘0’
	‘B’
	‘1’
Contents of second data address 0201H	‘1’
	‘F’
	‘4’
	‘0’
LRC Check	‘E’
	‘8’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

RTU Mode:

Command message:

Slave address	01H
Function	03H
Starting data address	02H (Upper bytes)
	00H (Lower bytes)
Number of data (In Word)	00H
	02H
CRC Check Low	C5H (Lower bytes)
CRC Check High	B3H (Upper bytes)

Response message:

Slave address	01H
Function	03H
Number of data (In Byte)	04H
Contents of starting data address 0200H	00H (Upper bytes)
	B1H (Lower bytes)
Contents of second data address 0201H	1FH (Upper bytes)
	40H (Lower bytes)
CRC Check Low	A3H (Lower bytes)
CRC Check High	D4H (Upper bytes)

Please note that a silent interval of more than 10ms is required before and after data transmission in RTU mode.

Example 2

Function code: 06H, write 1 word

For example, writing 100 (0064H) to starting data address 0200H of ASDA-B2 series with address 01H.

ASCII Mode:

Command message:

Start	‘:’
Slave address	‘0’
	‘1’
Function	‘0’
	‘6’
Starting data address	‘0’
	‘2’
	‘0’
	‘0’
Content of data	‘0’
	‘0’
	‘6’
	‘4’
LRC Check	‘9’
	‘3’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

Response message:

Start	‘:’
Slave address	‘0’
	‘1’
Function	‘0’
	‘6’
Starting data address	‘0’
	‘2’
	‘0’
	‘0’
Content of data	‘0’
	‘0’
	‘6’
	‘4’
LRC Check	‘9’
	‘3’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

RTU Mode:

Command message:

Slave Address	01H
Function	06H
Starting data address	02H (Upper bytes)
	00H (Lower bytes)
Content of data	00H (Upper bytes)
	64H (Lower bytes)
CRC Check Low	89H (Lower bytes)
CRC Check High	99H (Upper bytes)

Response message:

Slave Address	01H
Function	06H
Starting data address	02H (Upper bytes)
	00H (Lower bytes)
Content of data	00H (Upper bytes)
	64H (Lower bytes)
CRC Check Low	89H (Lower bytes)
CRC Check High	99H (Upper bytes)

Please note that a silent interval of more than 10ms is required before and after data transmission in RTU mode.

Example 3

Function code: 10H, write N words (multiple words). The maximum value of N is 10.

For example, writing continuous 2 words, 0BB8H and 0000H from starting address 0112H and address 0113H.

ASCII Mode:

Command message:

Start	':'
Slave Address	'0'
	'1'
Function	'1'
	'0'
Starting data address	'0'
	'1'
	'1'
	'2'
Number of data (In Word)	'0'
	'0'
	'0'
	'2'
Number of data (In Byte)	'0'
	'4'
Content of 1st data	'0'
	'B'
	'B'
	'8'
Content of 2nd data	'0'
	'0'
	'0'
	'0'

Response message:

Start	':'
Slave Address	'0'
	'1'
Function	'1'
	'0'
Starting data address	'0'
	'1'
	'1'
	'2'
Content of data	'0'
	'0'
	'0'
	'2'
LRC Check	'D'
	'A'
End 1	(0DH)(CR)
End 0	(0AH)(LF)

LRC Check	‘1’
	‘3’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

RTU Mode:

Command message:

Slave Address	01H
Function	10H
Starting data address	01H (Upper bytes)
	12H (Lower bytes)
Number of data (In Word)	00H (Upper bytes)
	02H (Lower bytes)
Number of data (In Byte)	04H
Content of 1st data	0BH (Upper bytes)
	B8H (Lower bytes)
Content of 2nd data	00H (Upper bytes)
	00H (Lower bytes)
CRC Check Low	FCH (Lower bytes)
CRC Check High	EBH (Upper bytes)

Response message:

Slave Address	01H
Function	10H
Starting data address	01H (Upper bytes)
	12H (Lower bytes)
Content of data (In Word)	00H (Upper bytes)
	02H (Lower bytes)
CRC Check Low	E0H (Lower bytes)
CRC Check High	31H (Upper bytes)

Please note that a silent interval of more than 10ms is required before and after data transmission in RTU mode.

LRC (ASCII Mode):

LRC (Longitudinal Redundancy Check) is calculated by summing up, module 256, the values of the bytes from ADR to last data character then calculating the hexadecimal representation of the 2’s-complement negation of the sum.

Example

STX	‘:’
ADR	‘7’
	‘F’
CMD	‘0’
	‘3’
Starting data address	‘0’
	‘5’
	‘C’
	‘4’
Number of data	‘0’
	‘0’
	‘0’
	‘1’
LRC Check	‘B’
	‘4’

End 1	(0DH)(CR)
End 0	(0AH)(LF)

7FH + 03H + 05H + C4H + 00H + 01H = 14CH, the 2's complement negation of 4CH is B4H. Hence, we can know that LRC CHK is 'B','4'.

CRC (RTU Mode):

CRC (Cyclical Redundancy Check) is calculated by the following steps:

Step 1: Load a 16-bit register (called CRC register) with FFFFH.

Step 2: Exclusive OR the first 8-bit byte of the command message with the low order byte of the 16-bit CRC register, putting the result in the CRC register.

Step 3: Extract and examine the LSB. If the LSB of CRC register is 0, shift the CRC register one bit to the right. If the LSB of CRC register is 1, shift the CRC register one bit to the right, then Exclusive OR the CRC register with the polynomial value A001H.

Step 4: Repeat step 3 until eight shifts have been performed. When this is done, a complete 8-bit byte will have been processed, then perform step 5.

Step 5: Repeat step 2 to step 4 for the next 8-bit byte of the command message.

Continue doing this until all bytes have been processed. The final contents of the CRC register are the CRC value.

NOTE

1) When transmitting the CRC value in the message, the upper and lower bytes of the CRC value must be swapped, i.e. the lower order byte will be transmitted first.

2) For example, reading 2 words from address 0101H of the AC servo drive with address 01H. The final content of the CRC register from ADR to last data character is 3794H, then the command message is shown as follows. What should be noticed is that 94H have to be transmitted before 37H.

Command Message	
ADR	01H
CMD	03H
Starting data address	01H (Upper byte)
	01H (Lower bytes)
Number of data (In Word)	00H (Upper bytes)
	02H (Lower bytes)
CRC Check Low	94H (Lower bytes)
CRC Check High	37H (Upper bytes)

End1, End0 (Communication End)

ASCII Mode:

In ASCII mode, (0DH) stands for character '\r' (carriage return) and (0AH) stands for character '\n' (new line), they indicate communication end.

RTU Mode:

In RTU mode, a silent interval of more than 10ms indicates communication end.

CRC Program Example

The following is an example of CRC generation using C language. The function takes two arguments:

unsigned char* data;

unsigned char length

The function returns the CRC value as a type of unsigned integer.

```
unsigned int crc_chk(unsigned char* data, unsigned char length) {
        int j;
        unsigned int reg_crc=0xFFFF;

        while( length-- ) {
                reg_crc^= *data++;
                for (j=0; j<8; j++ ) {
                        if( reg_crc & 0x01 ) { /*LSB(bit 0 ) = 1 */
                                reg_crc = (reg_crc >> 1)^0xA001;
                        } else {
                                reg_crc = (reg_crc>>1);
                        }
                }
        }
        return reg_crc;
}
```

PC communication program example:

```
#include<stdio.h>
#include<dos.h>
#include<conio.h>
#include<process.h>
#define PORT 0x03F8      /*  the address of COM 1  */
#define THR 0x0000
#define RDR 0x0000
```

```
#define BRDL 0x0000
#define IER 0x0001
#define BRDH 0x0001
#define LCR 0x0003
#define MCR 0x0004
#define LSR 0x0005
#define MSR 0x0006
unsigned char rdat[60];
/* read 2 data from address 0200H of ASD with address 1 */
unsigned char tdat[60]={':','0','1','0','3','0','2','0','0','0','0','0','2','F','8','\r','\n'};
void main() {
int I;
outportb(PORT+MCR,0x08);              /*  interrupt enable  */
outportb(PORT+IER,0x01);                      /*  interrupt as data in  */
outportb(PORT+LCR,( inportb(PORT+LCR) | 0x80 ) );
/*  the BRDL/BRDH can be access as LCR.b7 == 1  */
outportb(PORT+BRDL,12);
outportb(PORT+BRDH,0x00);
outportb(PORT+LCR,0x06);              /* set prorocol
                                              <7,E,1> = 1AH,              <7,O,1> = 0AH
                                              <8,N,2> = 07H               <8,E,1> = 1BH
                                              <8,O,1> = 0BH
      */

for( I = 0; I<=16; I++ ) {
      while( !(inportb(PORT+LSR) & 0x20) );       /*  wait until THR empty  */
      outportb(PORT+THR,tdat[I]);                 /*  send data to THR  */
}
I = 0;
while( !kbhit() ) {
      if( inportb(PORT+LSR)&0x01 ) {  /*  b0==1, read data ready  */
            rdat[I++] = inportb(PORT+RDR); /*   read data from RDR  */
      }
}
}
```

8.4 Communication Parameter Write-in and Read-out

There are following five groups for parameters:

Group 0: Monitor parameter	(example: P0-xx)
Group 1: Basic parameter	(example: P1-xx)
Group 2: Extension parameter	(example: P2-xx)
Group 3: Communication parameter	(example: P3-xx)
Group 4: Diagnosis parameter	(example: P4-xx)

For a complete listing and description of all parameters, refer to Chapter 7.

Communication write-in parameters for ASDA-B2 series are including:

Group 0: All parameters except P0-00 ~ P0-01, P0-08 ~ P0-13 and P0-46

Group 1: P1-00 ~ P1-76

Group 2: P2-00 ~ P2-67

Group 3: P3-00 ~ P3-11

Group 4: All parameters except P4-00 ~ P4-04 and P4-08 ~ P4-09

NOTE

1) P3-01 After the new transmission speed is set, the next data will be written in new transmission speed.

2) P3-02 After the new communication protocol is set, the next data will be written in new communication protocol.

3) P4-05 JOG control of servo motor. For the description, refer to Chapter 7.

4) P4-06 Force output contact control. This parameter is for the users to test if DO (Digit output) is normal. User can set 1, 2, 4, 8, 16, 32 to test DO1, DO2, DO3, DO4, DO5, DO6 respectively. After the test has been completed, please set this parameter to 0 to inform the drive that the test has been completed.

5) P4-10 Adjustment function selection. If the user desires to change the settings of this parameter, the user has to set the value of the parameter P2-08 to 20 (hexadecimal: 14H) first and then restart. After restarting, the settings of parameter P4-10 can become modified.

6) P4-11 ~ P4-21 These parameters are for offset adjustment. Do not change the factory default setting if not necessary. If the user desires to change the settings of these parameters, the user has to set the value of the parameter P2-08 to 22 (hexadecimal: 16H) first and then restart. After restarting, the settings of parameters P4-11 to P4-21 can become modified.

Communication read-out parameters for ASDA-B2 series are including:

Group 0: P0-00 ~ P0-46

Group 1: P1-00 ~ P1-76

Group 2: P2-00 ~ P2-67

Group 3: P3-00 ~ P3-11

Group 4: P4-00 ~ P4-24