APPLICATION NOTE

Product Family: DirectLogic PLCs

Number: AN-MISC-029

Subject: How to write DirectNET protocol

Date Issued: 09/04/08

Revision: Original

This document explains the particulars of the DirectNET protocol such as message content and message flow. It assumes that the user has working knowledge and experience in writing communication protocols.

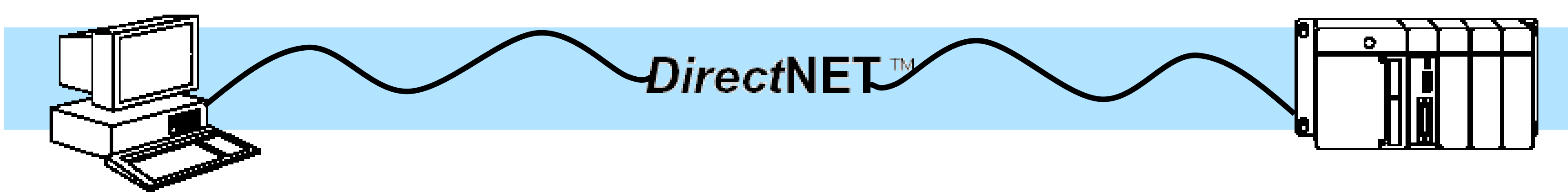
This document does not address the particulars of any code language or platform. It is the user's responsibility to generate the necessary code for their system when developing an application using this protocol.

DirectNET Protocol

```
4e 21 05          01 30 31 30 31 30 30 30 31 30
      4e 21 06
30 30 32 30 30 17 03                06     04
              06 02 34 12 03 26      04
4e 21 05             01 30 31 30 31 30 30 30 31
      4e 21 06 0d
30 30 30 34 30 30 17 30 35
                    06 0d 02 33 34 31 32
        06     04 4e 21 05           01 30
03 30 34 0d   04 0d       4e 21 06
31 38 31 30 30 30 31 30 30 30 32 30 30 17 0b
                                           06
02 34 12 03 26    04 4e 21 05            01 30
          06            4e 21 06 0d
31 38 31 30 30 30 31 30 30 30 34 30 30 17 30 44

   02 33 34 31 32 03 30 34      04 4e 21 05
06 0d                  06 0d
      01 30 31 38 31 30 30 30 31 30 30 38 30
4e 21 06
30 30 17 01   02 34 12 00 00 00 00 00 00 00 00
        06
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
00 00 00 00 00 00 03 26     04 4e 21 05
                    06            4e 21 06
   01 30 31 38 31 30 30 30 31 30 31 30 30 30 30
0d
17 30 38      02 33 34 31 32 30 30 30 30 30 30
         06 0d
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 03 30 34           04 4e 21
                           06 0d 04 0d
05            01 30 31 30 31 30 30 30 31 30 31
   4e 21 06 0d
30 30 30 30 17 30 30
                   06 0d 02 33 34 31 32 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
```

```
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30

30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
                                             06
30 30 30 30 30 30 30 30 30 30 03 30 34 0d    04
   04 4e 21 05         01 30 31 30 31 30 30 30
0d            4e 21 06
31 30 30 34 30 30 30 17 05
                           06 02 34 12 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
                                          06    04
00 00 00 00 00 00 00 00 00 00 00 03 26    04
4e 21 05        01 30 31 30 31 30 30 30 31 30
   4e 21 06
30 38 30 30 30 17 09
                    06 02 34 12 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

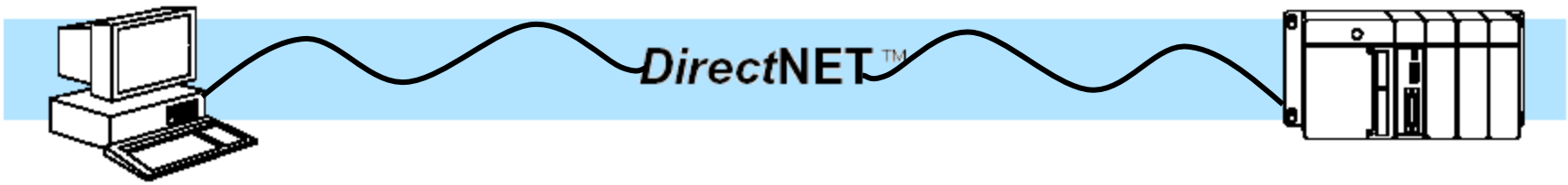


Control Characters used in DirectNET

- ENQ (0x05) Enquiry to start communications
- ACK (0x06) Acknowledge (data received and no errors
- NAK (0x15) Negative Acknowledge (data received but there were errors)
- SOH (0x01) Start of Header
- ETB (0x17) End of Transmission Block (intermediate block)
- STX (0x02) Start of Text (beginning of data block)
- ETX (0x03) End of Text (end of last data block)
- EOT (0x04) End of Transmission (transaction complete)

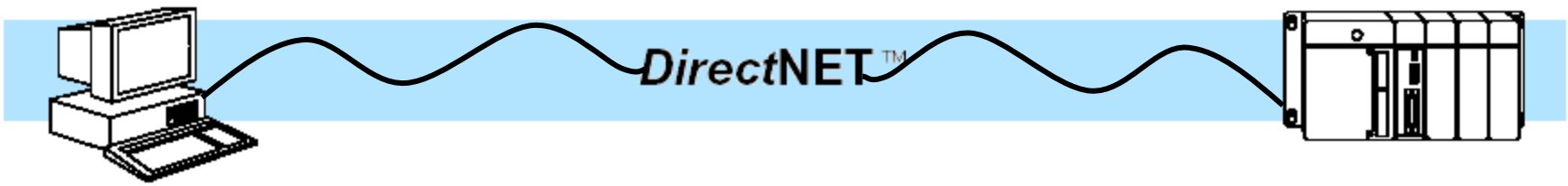
These are standard ASCII Control characters and are not unique to the DirectNET protocol.

NOTE: All control characters are in Hex format



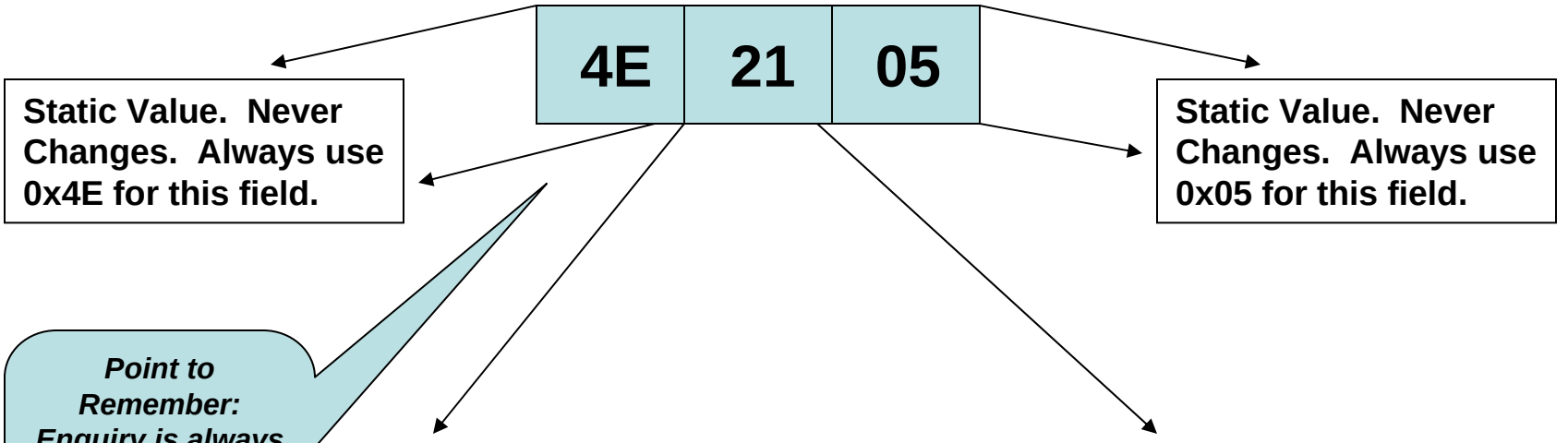
Components of DirectNET

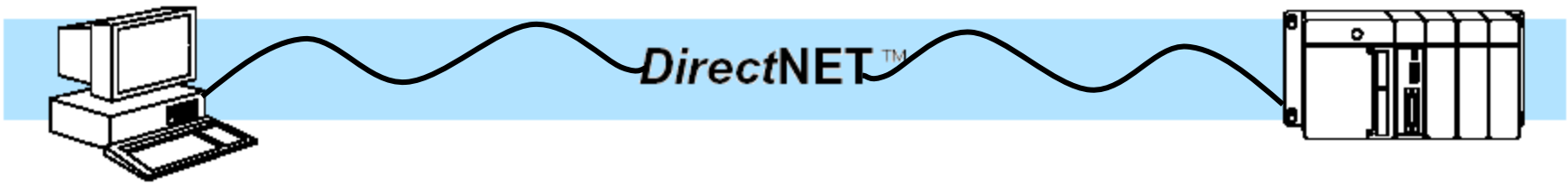
- Enquiry
- Header
- Data Packet
- Ack (Acknowledge)
- EOT (End of Transmission)



Components of DirectNET

Enquiry:



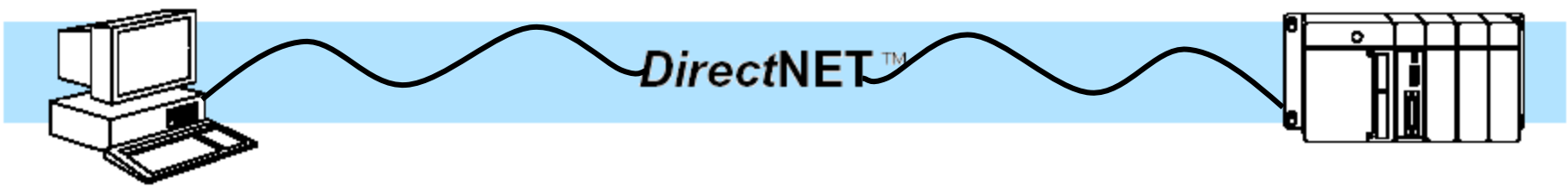


Components of DirectNET

Header:

Byte:	1	2+3	4	5	6+7	8+9	10+11	12+13	14+15	16	17 (+18)
	01	**3034**	**30**	**31**	**3431**	**3031**	**3031**	**3930**	**3031**	**17**	**08** **30 38**

SOH (Start of Header): Never Changes. Always use 0x01 for this field.

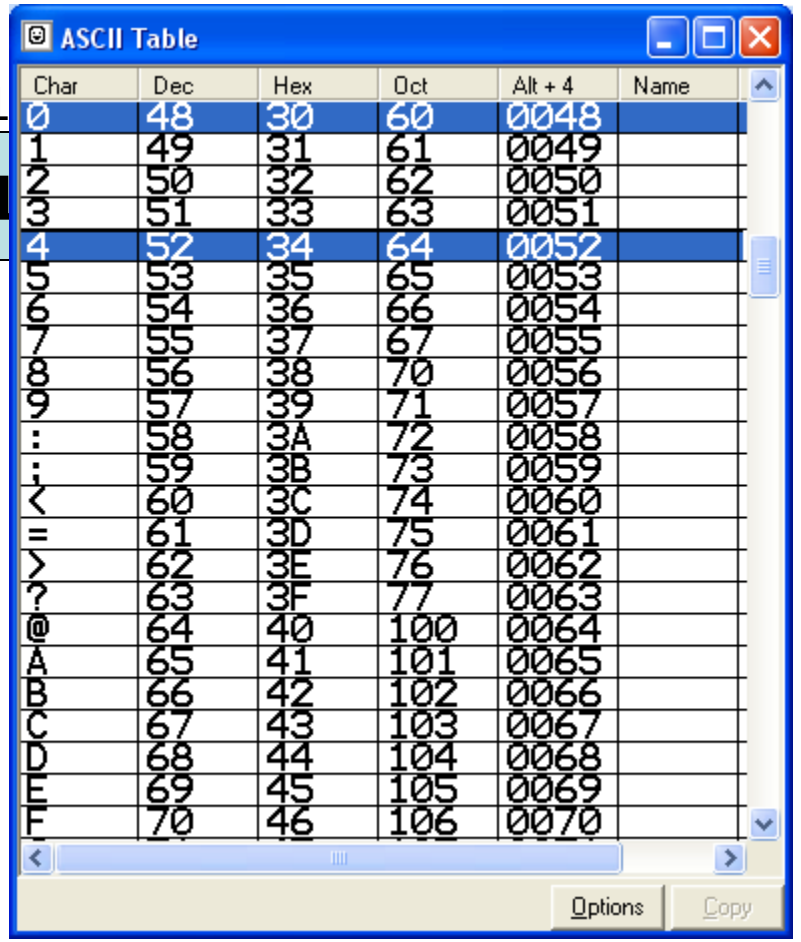


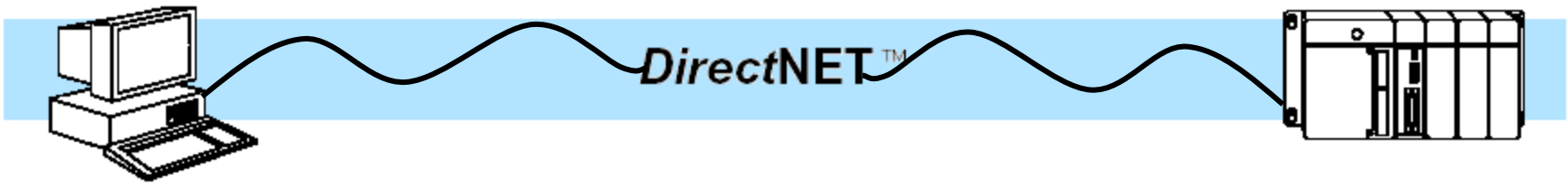
Components of DirectNET

Header:

Byte:	1	2+3	4	5	6+7	8+9	10+11
	01	**3034**	**30**	**31**	**3431**	**3031**	**3031**

Address Field *without* offset. Simple Conversion: Take the decimal slave value. Convert to Hex. (NOTE: DCM address already in hex format) Example: Decimal 04= Hex 0x04. Look at ASCII Table: 0=30 4=34. Enter 3034 into the slave address field. Another example: Decimal 60= Hex 0x3C. 3=33 C=43. Enter 3343 into slave address field.



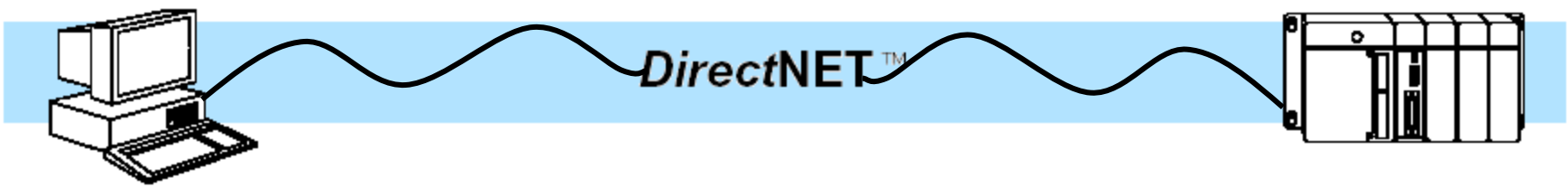


Components of DirectNET

Header:

Byte:	1	2+3	4	5	6+7	8+9	10+11	12+13	14+15	16	17 (+18)
	01	**3034**	**30**	**31**	**3431**	**3031**	**3031**	**3930**	**3031**	**17**	**08** **30 38**

Read or Write Request Field: Enter 30 if Read or 38 if write.

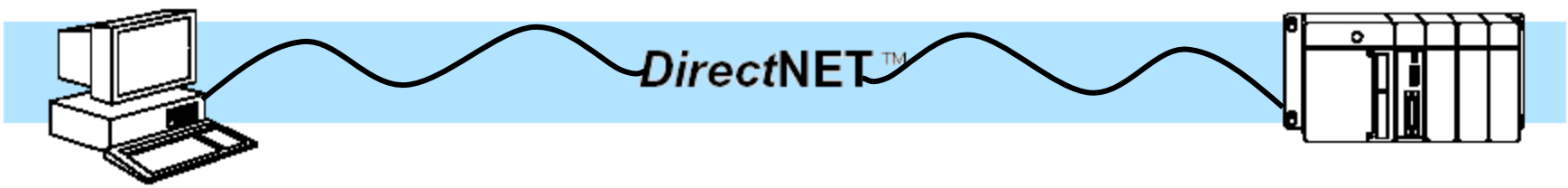


Components of DirectNET

Header:

Byte:	1	2+3	4	5	6+7	8+9	10+11	12+13	14+15	16	17 (+18)
	01	**3034**	**30**	**31**	**3431**	**3031**	**3031**	**3930**	**3031**	**17**	**08** **30 38**

Data Type Field: Refer to Appendixes D-F for the appropriate PLC mapping. Example: DL205 V-memory is 31.

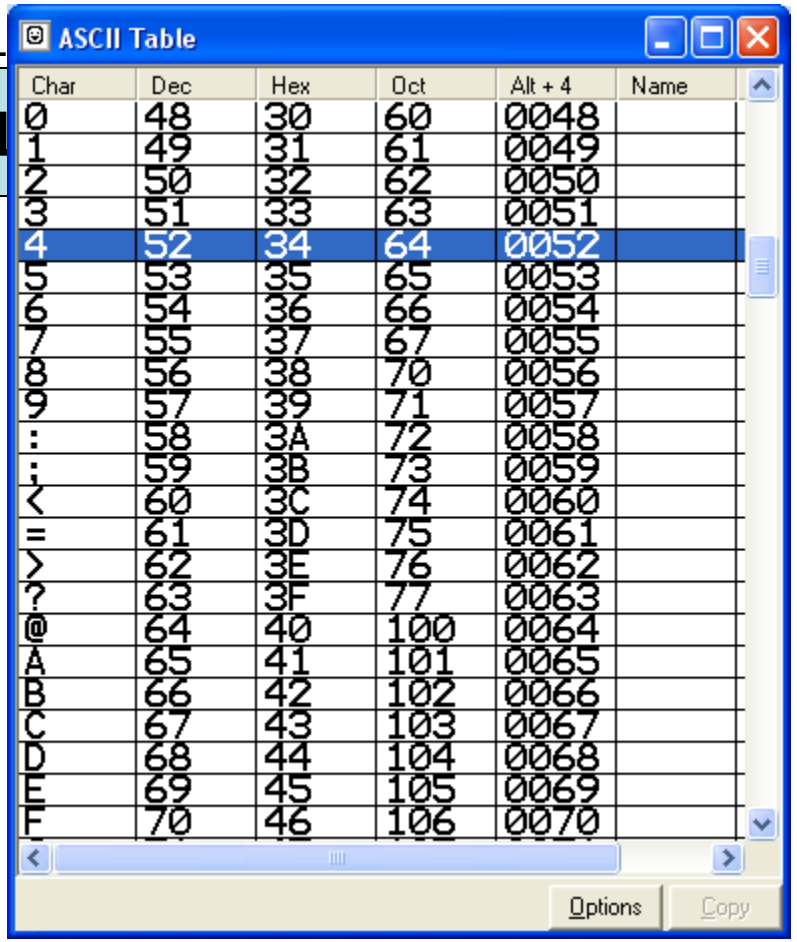


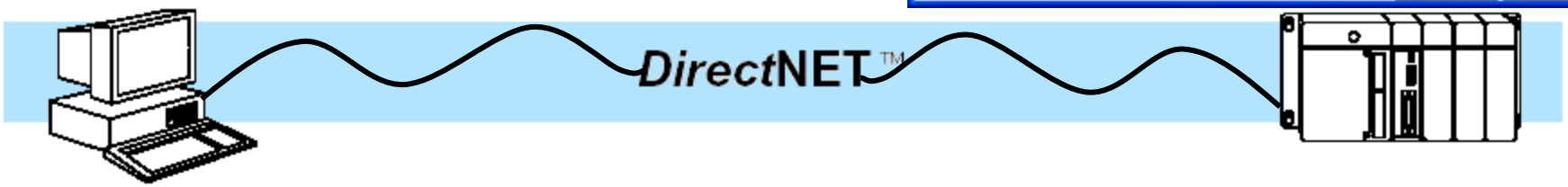
Components of DirectNET

Header:

Byte:	1	2+3	4	5	6+7	8+9	10+11
	01	**3034**	**30**	**31**	**3431**	**3031**	**3031**

Starting Address Fields: Refer to Appendixes D-F for the appropriate 4 digit PLC *Reference* address to start read or writing at. The value that is entered into these fields is a octal to hex conversion plus an offset of 1. For example: V40400=octal 40400 -> 0x4100 + 1 = 4101 Reference address. You then convert this value with the ASCII table: 4=34 1=31 0=30 1=31 to get 3431 3031.



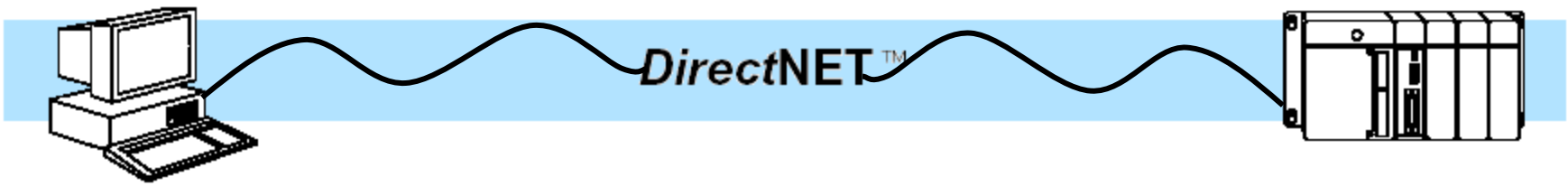


Components of DirectNET

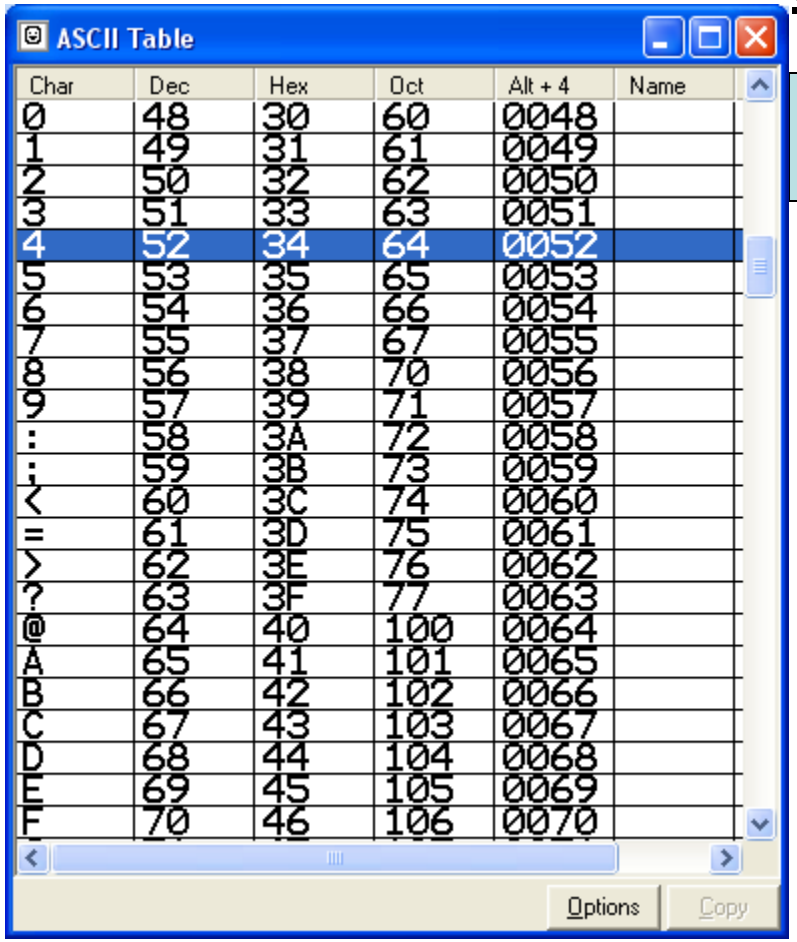
Header:

Byte:	1	2+3	4	5	6+7	8+9	10+11	12+13	14+15	16	17 (+18)
	01	**3034**	**30**	**31**	**3431**	**3031**	**3031**	**3930**	**3031**	**17**	**08** **30 38**

Number of Complete Data Blocks: Anytime you need more than 256 bytes of data, you would use this field. If you place a value of 1 into this field, you will get 256 bytes of data. If you place a value of 2 into this field, you will get 512 bytes of data (2 Data Blocks). Everytime you increment this value, you get 256 more bytes of data. Once you determine how many complete data blocks you want, you convert the number to hex and then use the ASCII Table to convert to the value to enter into the field.



Components of DirectNET

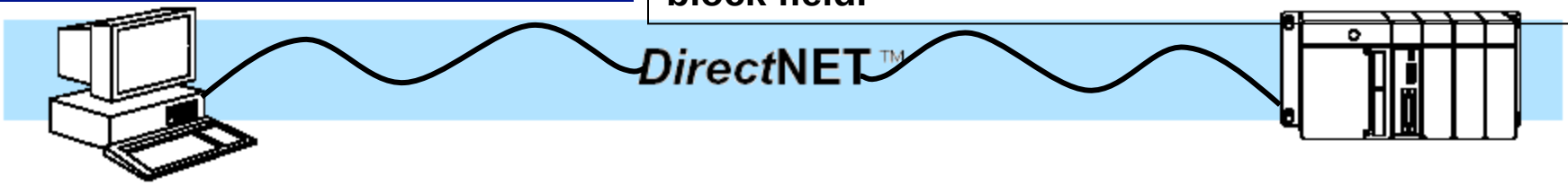


Header:

8+9	10+11	12+13	14+15	16	17 (+18)
3031	3031	3930	3031	17	08 30 38

Partial Data Blocks: You use this field anytime you want less than a complete block of data (<256 bytes). You do the same conversion as before. For example, if you want 72 V memory locations (144 bytes), you convert decimal 144= Hex 0x90. Then do the ASCII table look up. 9=39 0=30. You enter 3930 into the partial data block field.

Remember in ASCII mode, you have to request 4 bytes per desired V memory location. So to get 50 V memory locations (200 bytes), you convert 200= Hex 0xC8. Then do the ASCII table look up. C=43 8=38. You enter 4348 into the partial data block field.

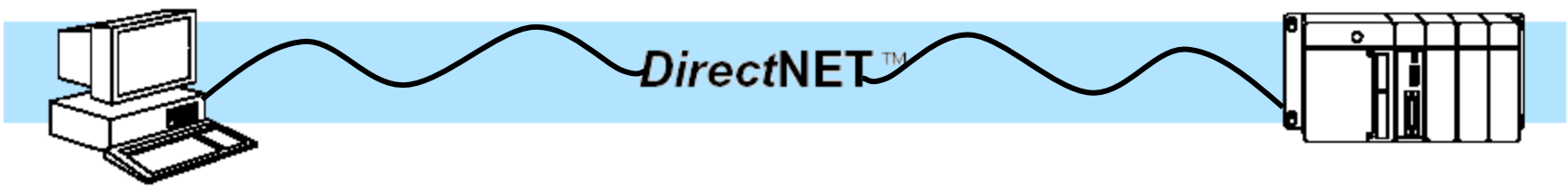


Components of DirectNET

Header:

Byte:	1	2+3	4	5	6+7	8+9	10+11	12+13	14+15	16	17 (+18)
	01	**3034**	**30**	**31**	**3431**	**3031**	**3031**	**3930**	**3031**	**17**	**08** **30 38**

Master ID: This field holds the value of the Master ID number. This will almost always be either 3030 for 0 or 3031 for 1. If you wanted a different value, you do the same conversions as before.

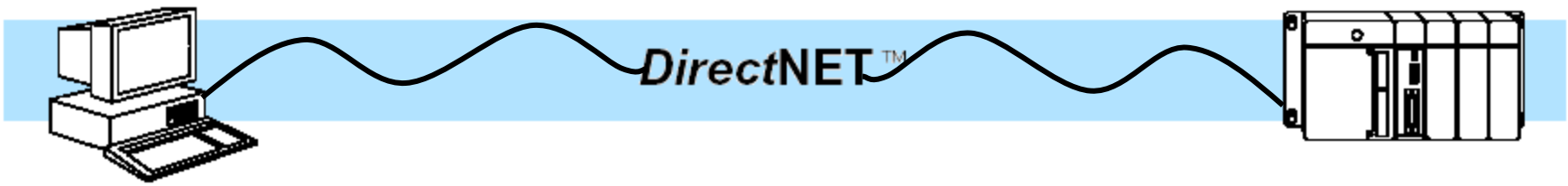


Components of DirectNET

Header:

Byte:	1	2+3	4	5	6+7	8+9	10+11	12+13	14+15	16	17 (+18)
	01	**3034**	**30**	**31**	**3431**	**3031**	**3031**	**3930**	**3031**	**17**	**08** **30 38**

End of Transmission: This field always holds the value of 0x17 since there is only one header for any given transaction in DirectNET.



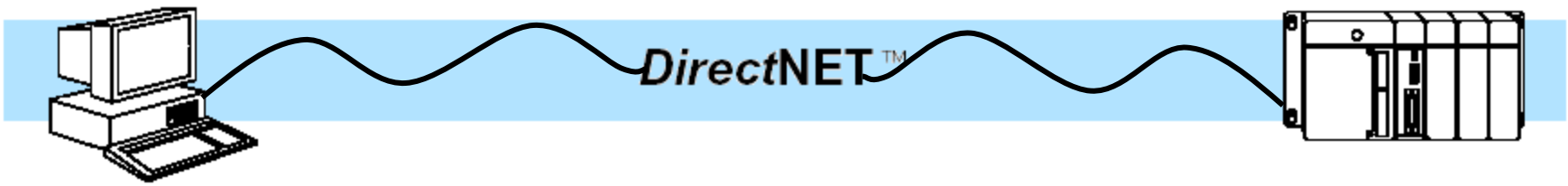
Components of DirectNET

Header:

Byte:	1	2+3	4	5	6+7	8+9	10+11	12+13	14+15	16	17 (+18)
	01	3034	30	31	3431	3031	3031	3930	3031	17	08 30 38

Point to remember: The header checksum is the first place that the format actually changes between Hex and ASCII mode

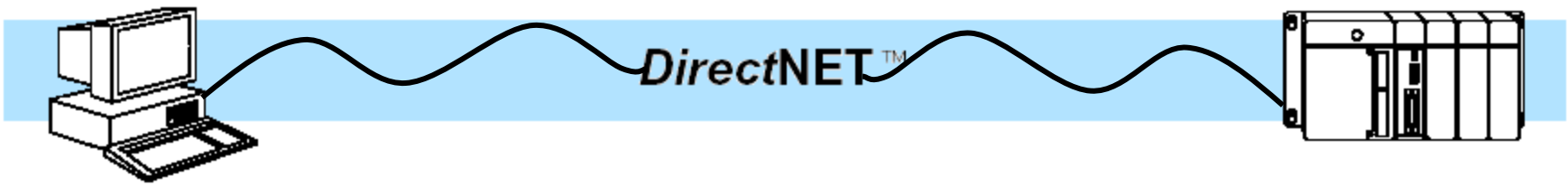
LRC Checksum: This field holds the checksum. The value in the upper field is the DirectNET Hex representation and the value in the lower field is the DirectNET ASCII representation. Remember that only bytes 2 – 15 are calculated in the LRC (represented in red here). Refer to the next slide for a simple chart method of calculating the LRC.



LRC Calculation:

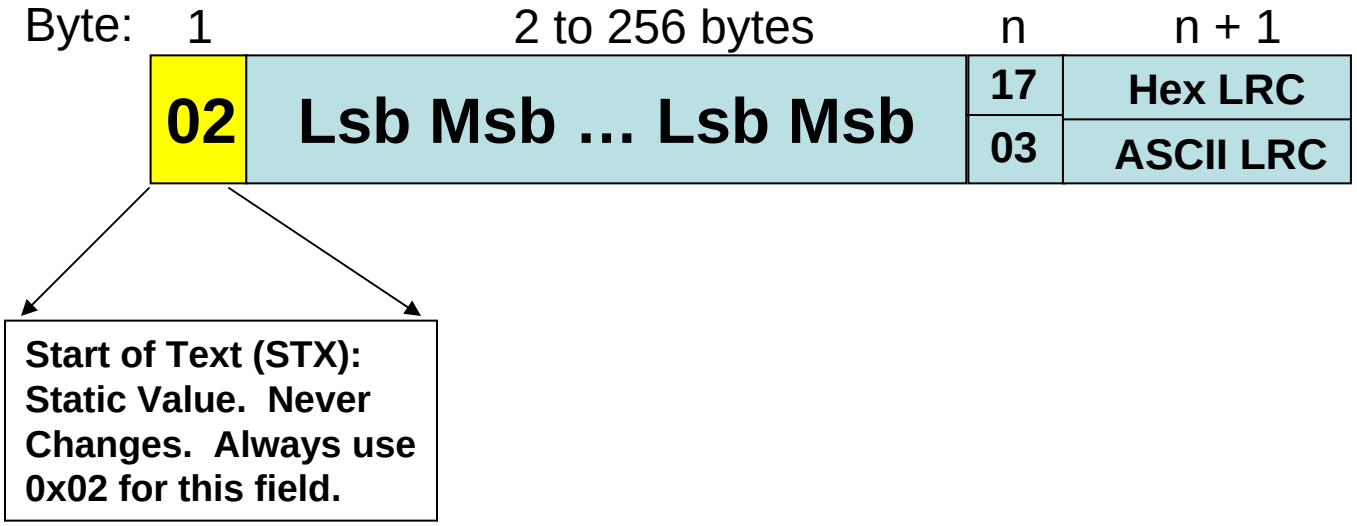
Using the values from the last slide, you can see that it is a very simple method of XORing (exclusive ORing) through bytes 2 – 15 of the Header. This chart shows a simple method of counting all the 1's and if they are even, the result is a 0, if they are odd, the result is a 1.

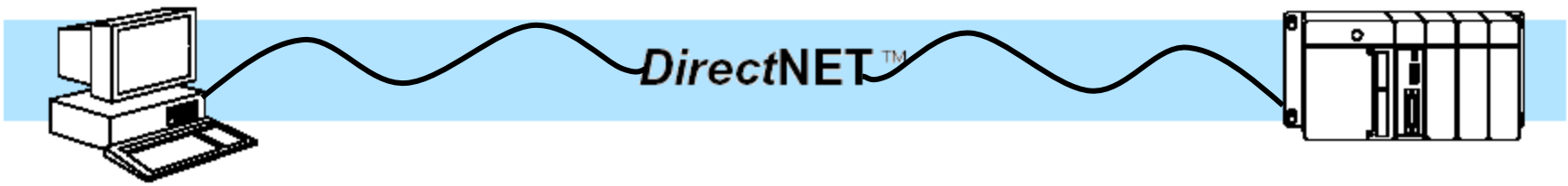
Byte		HEX ASCII Value	Binary Representation							
			B7 128	B6 64	B5 32	B4 16	B3 8	B2 4	B1 2	B0 1
Target slave:	(byte 2)	30	0	0	1	1	0	0	0	0
	(byte 3)	34	0	0	1	1	0	1	0	0
Read or write:	(byte 4)	30	0	0	1	1	0	0	0	0
Data Type:	(byte 5)	31	0	0	1	1	0	0	0	1
Data address MSB:	(byte 6)	34	0	0	1	1	0	1	0	0
	(byte 7)	31	0	0	1	1	0	0	0	1
Data address LSB:	(byte 8)	30	0	0	1	1	0	0	0	0
	(byte 9)	31	0	0	1	1	0	0	0	1
Complete blocks:	(byte 10)	30	0	0	1	1	0	0	0	0
	(byte 11)	31	0	0	1	1	0	0	0	1
Bytes in last block:	(byte 12)	39	0	0	1	1	1	0	0	1
	(byte 13)	30	0	0	1	1	0	0	0	0
Master address:(byte 14)		30	0	0	1	1	0	0	0	0
	(byte 15)	31	0	0	1	1	0	0	0	1
Total Number of "1s"			0	0	14	14	1	2	0	6
Even (E) or Odd (O)			E	E	E	E	O	E	E	E
Exclusive OR Results:			0	0	0	0	1	0	0	0
Hexadecimal Value			0				8			
HEX ASCII Code			30				38			



Components of DirectNET

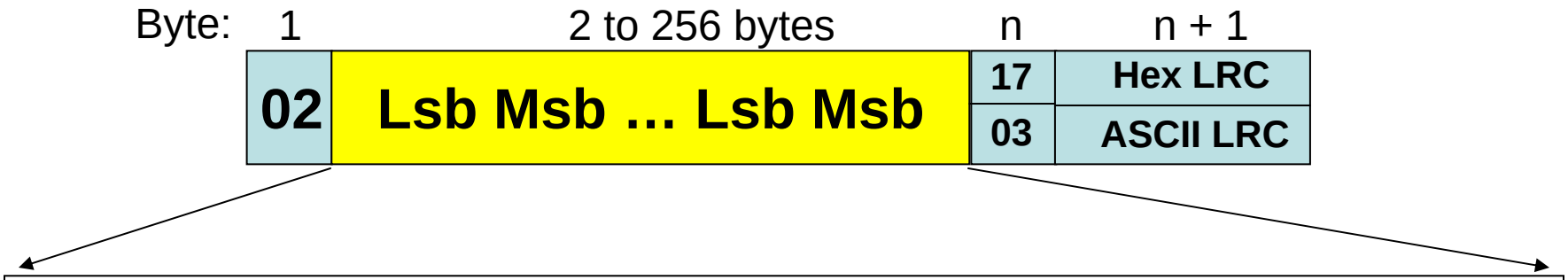
Data Packet:





Components of DirectNET

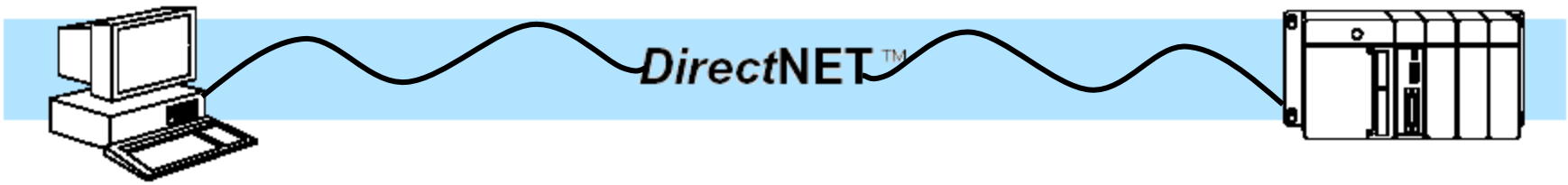
Data Packet:



Data Field: This field holds the actual data that is being written or read. Notice that the byte order is backwards from what how you would view it in data view (hex mode). This is another field that would differ in format between Hex and ASCII mode.

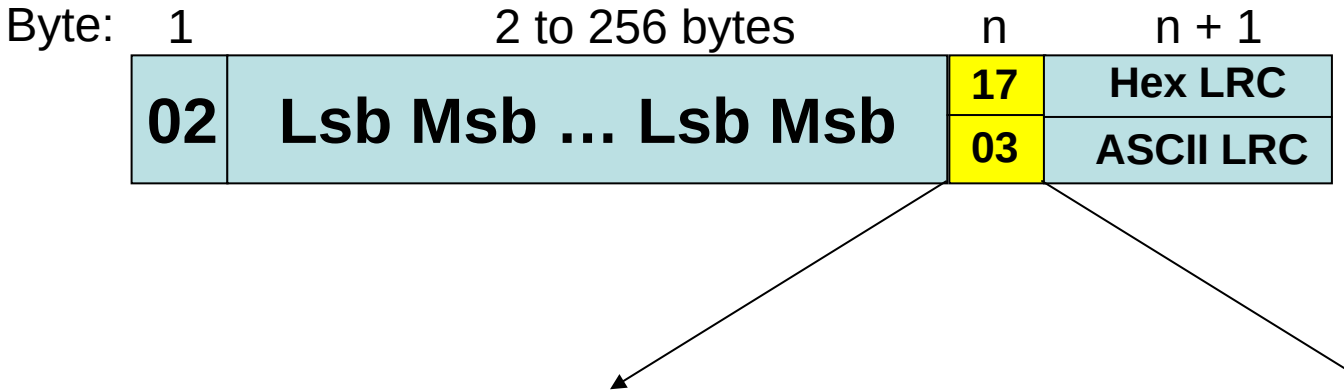
Example (Hex Mode): If you had placed a value of “1234” into a V memory location in Data view using the standard “BCD/Hex” format, you would see 34 12 in the data field.

Example (ASCII Mode): Same value as above entered into Data View in BCD/Hex format would show as 33 34 31 32 in the data field. NOTE: It takes twice as many bytes in ASCII Mode as opposed to Hex Mode. So…Remember that you need to request twice as many bytes for the same amount of data if using ASCII Mode.

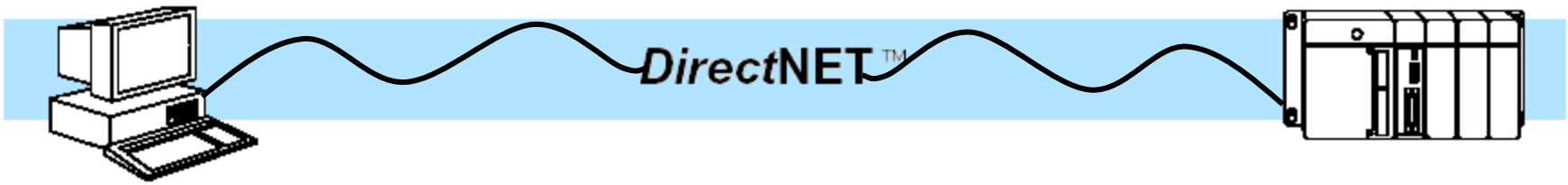


Components of DirectNET

Data Packet:

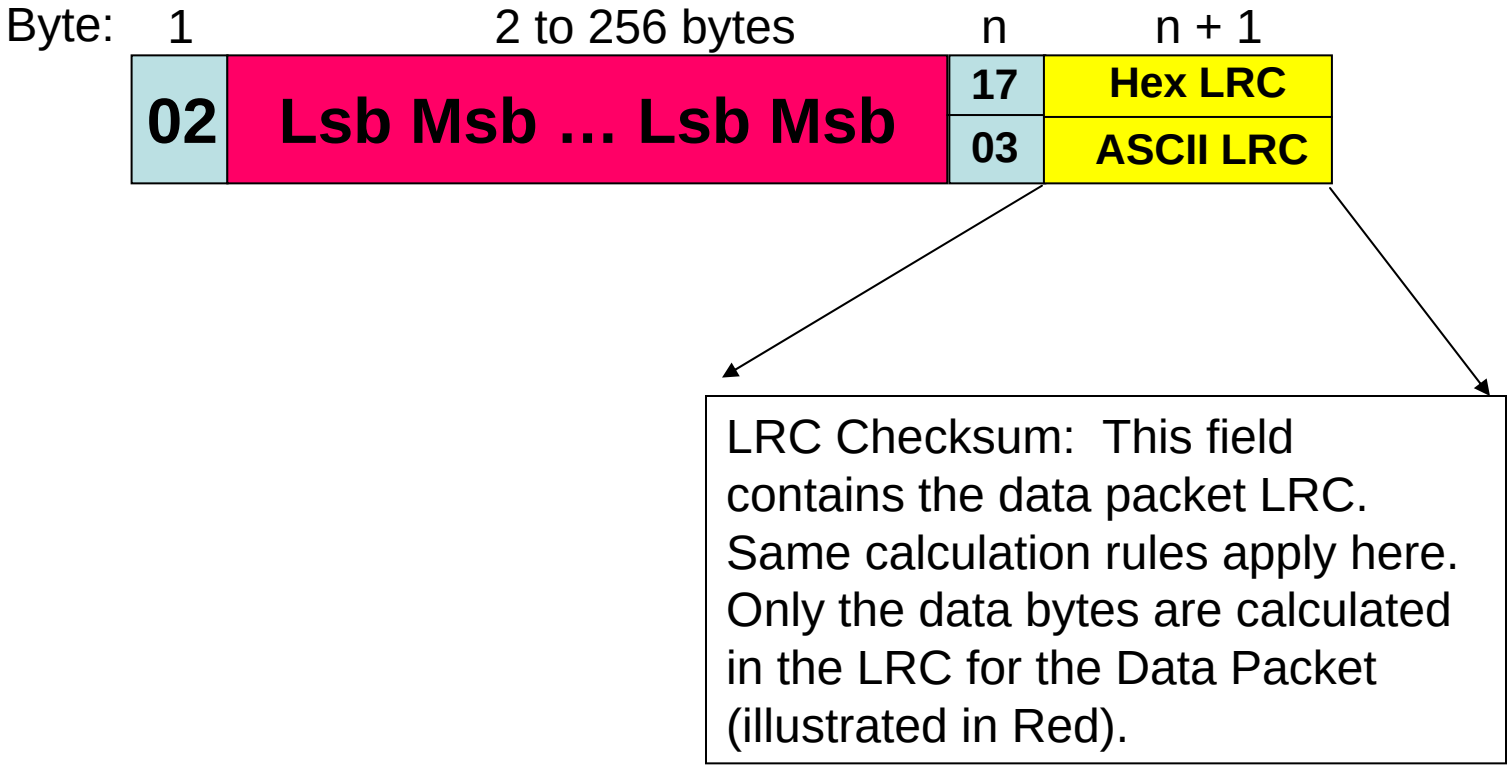


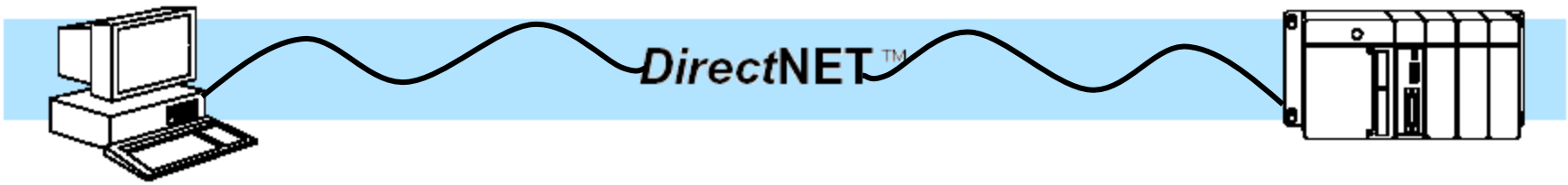
End of Text/End of Block: If you are reading or writing multiple blocks of data, this field should contain a 0x17(ETB) for all data blocks except for the last data block. The last data block should contain a 0x03(ETX).



Components of DirectNET

Data Packet:



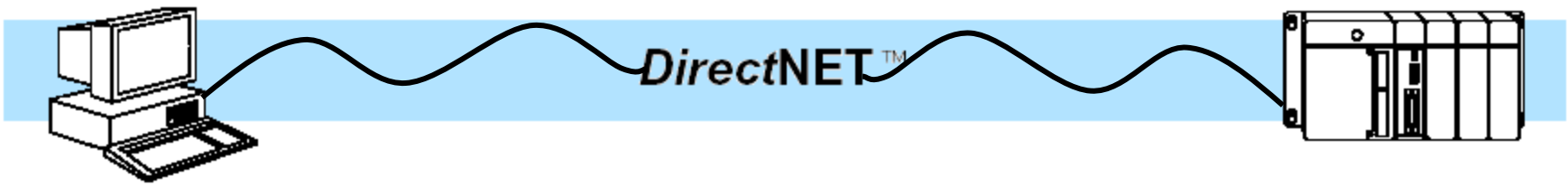


Components of DirectNET

Acknowledge:

06

Acknowledge: This is simply a hex 0x06. This is used to acknowledge different components of the transaction. This will be explained later in the sequence of events description.



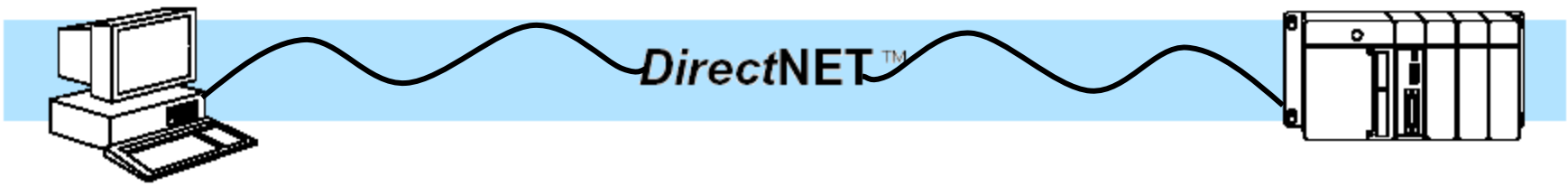
Components of DirectNET

End of Transmission:

04

Point to remember: An EOT given from either the master or slave will terminate the present communications transaction and the master will have to begin anew with the Enquiry.

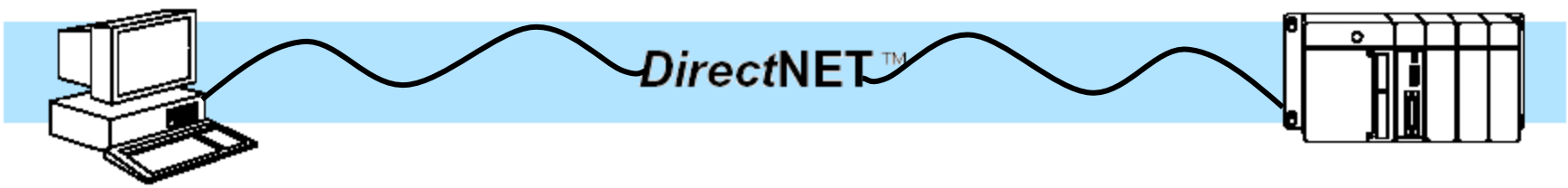
End of Transmission: This is simply an hex 0x04. This signifies the end of transmission on that transaction from the sending device.



Sequence of Events:

Read:

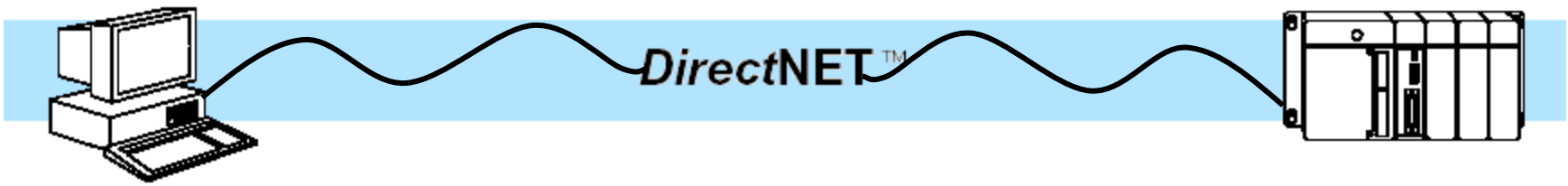
- **Master sends Enquiry**
- **Targeted Slave sends ACK**
- **Master sends Header**
- **Targeted Slave sends requested Data packet**
- **Master sends ACK**
- **If Multiple Data packets were requested, slave sends next data packet. If not, Slave sends EOT.**
- **Master sends EOT**



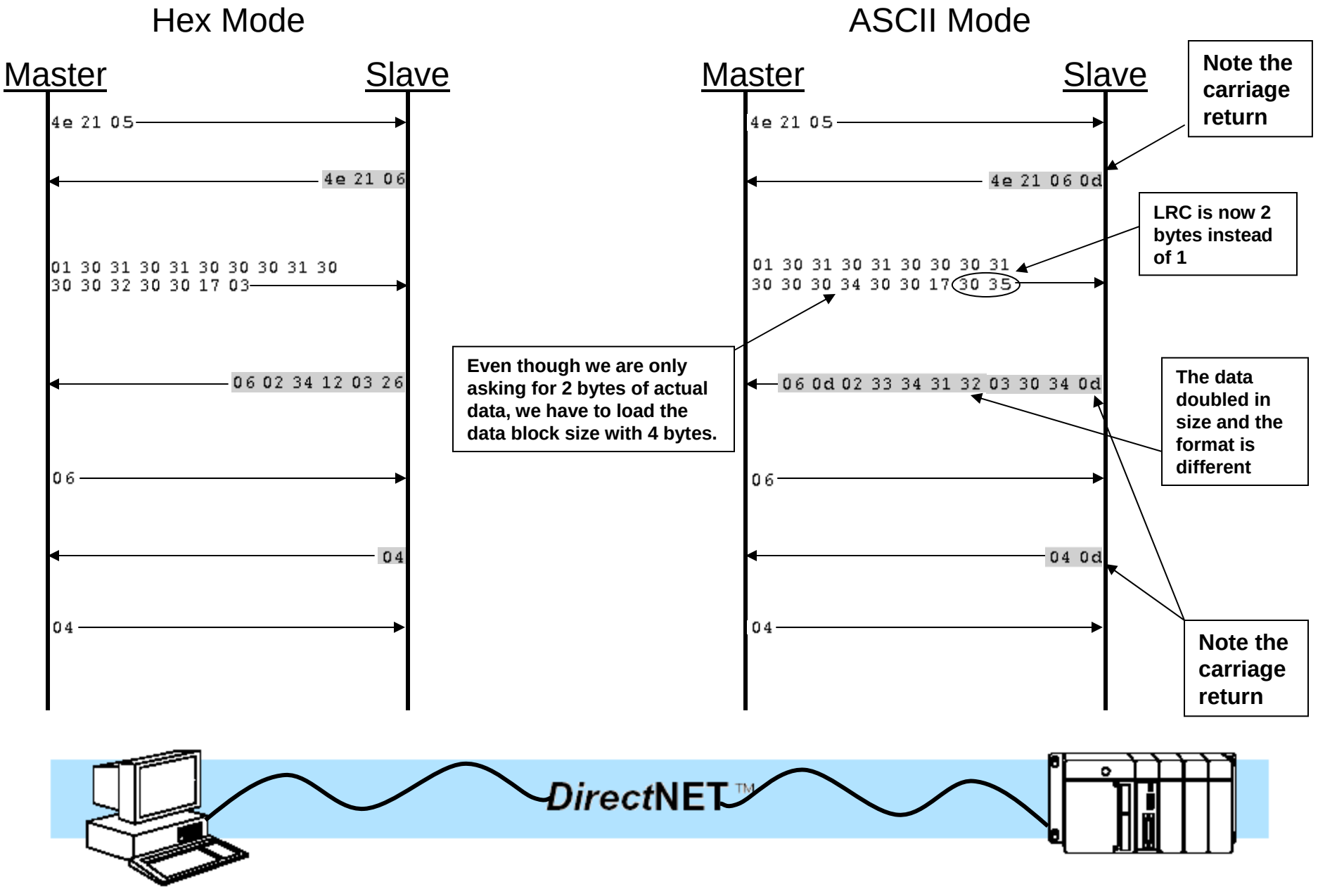
Sequence of Events:

Write:

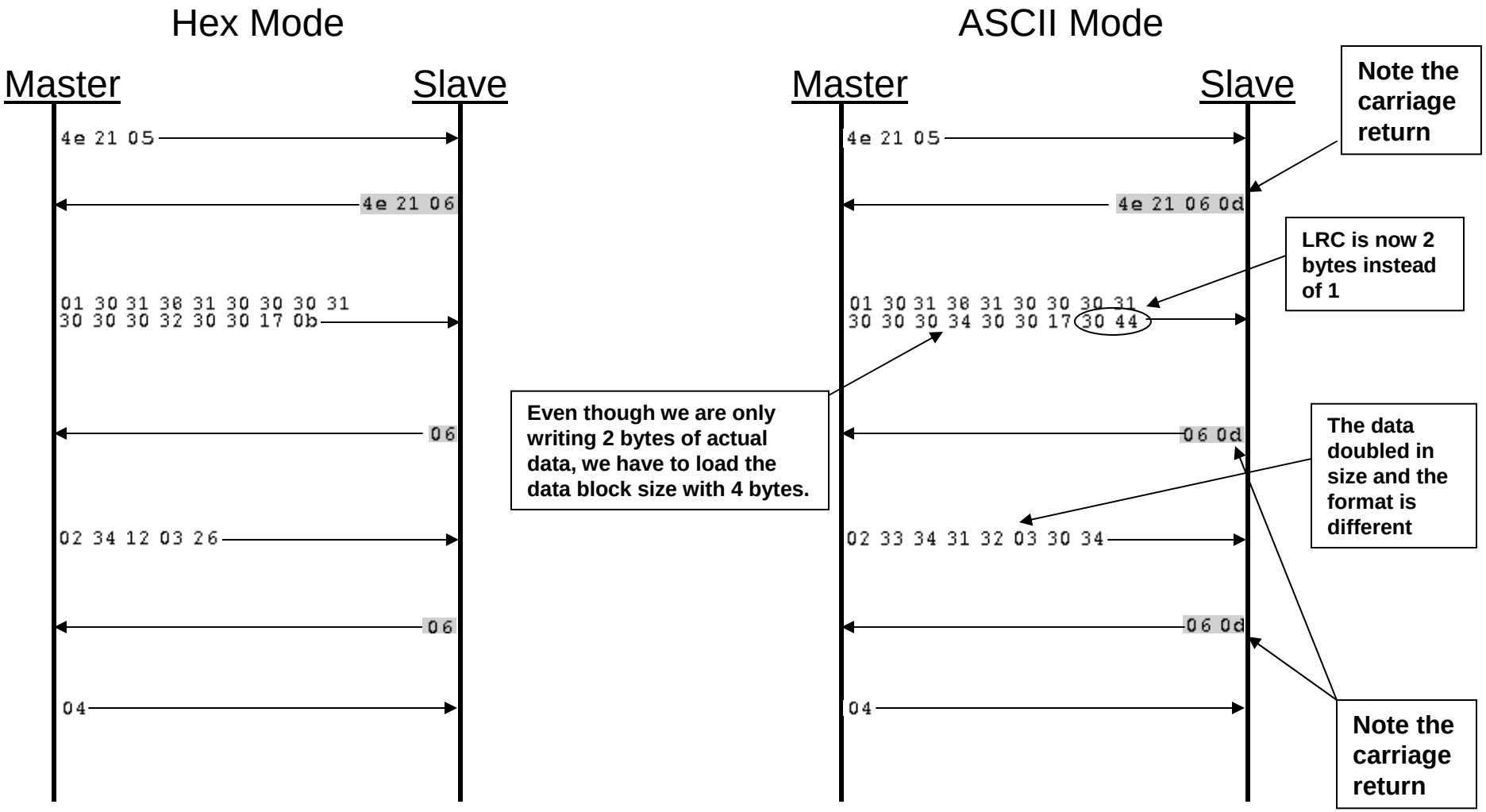
- **Master sends Enquiry**
- **Targeted Slave sends ACK**
- **Master sends Header**
- **Targeted Slave sends ACK**
- **Master sends Data Packet**
- **Slave Responds with ACK**
- **Once the Master has sent all Data Packets, it sends an EOT**

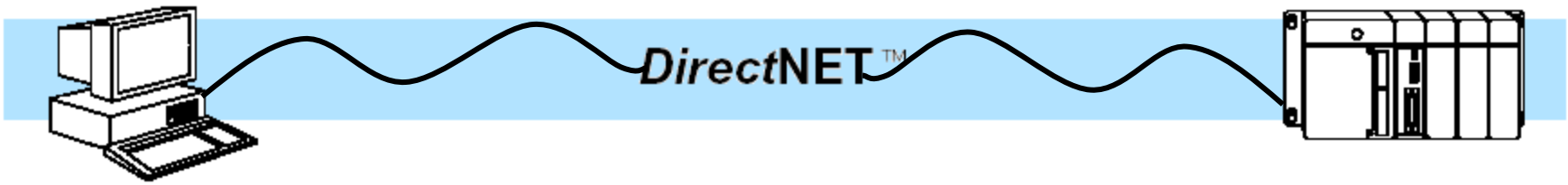


Comparison of a 2 byte read in both Hex and ASCII mode



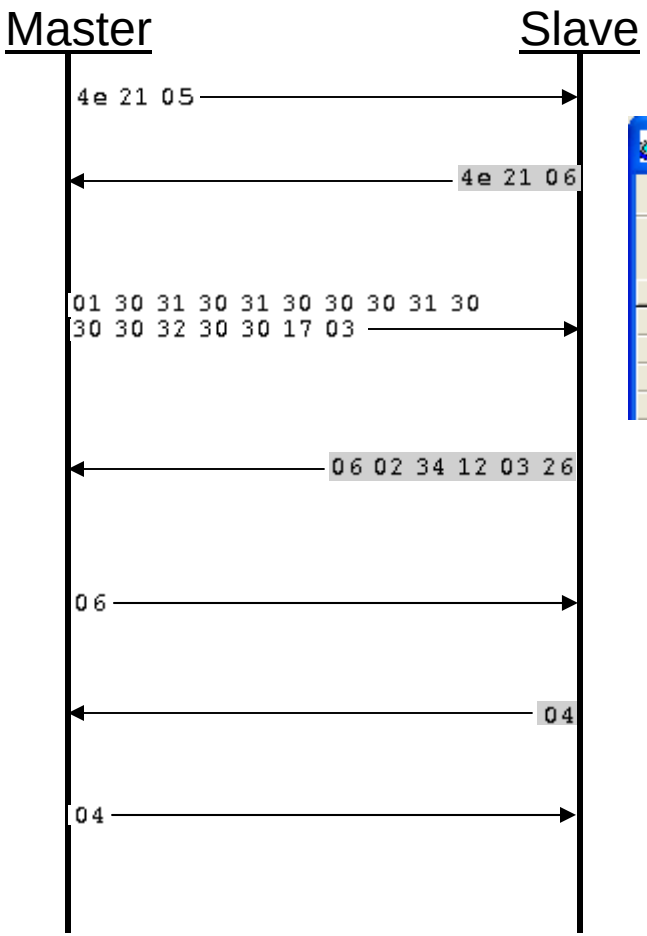
Comparison of a 2 byte write in both Hex and ASCII mode

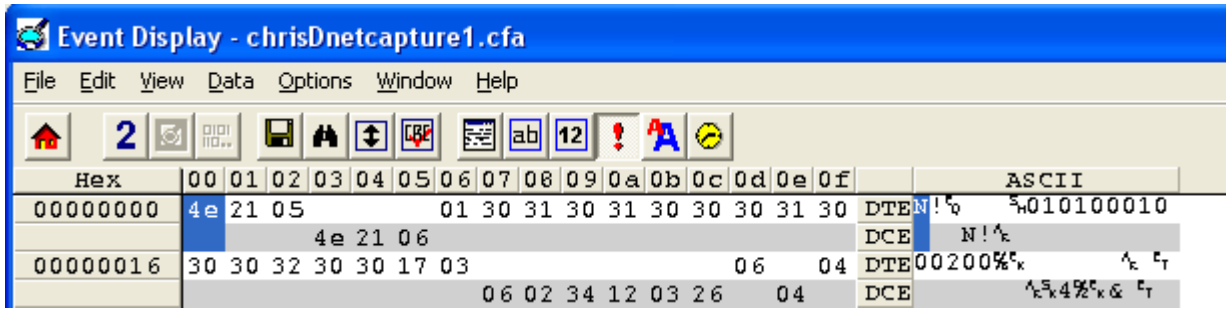


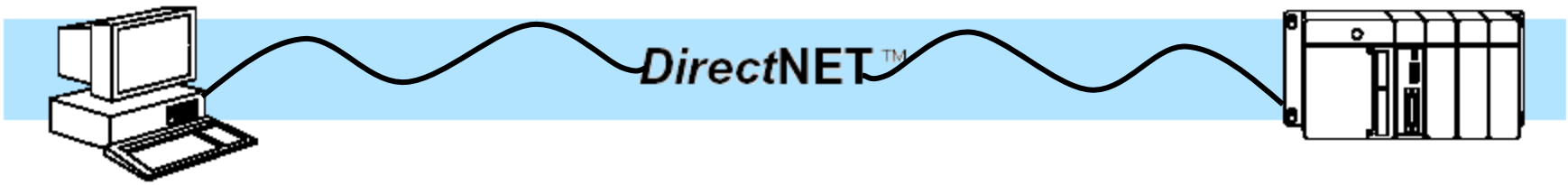


Example of how this would look on a comm analyzer

Hex Mode

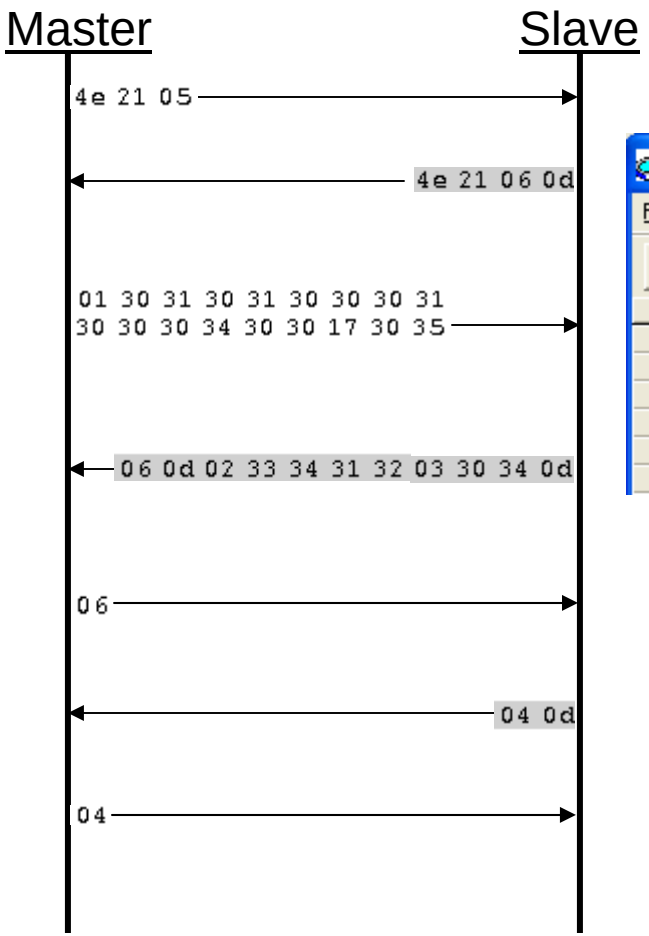


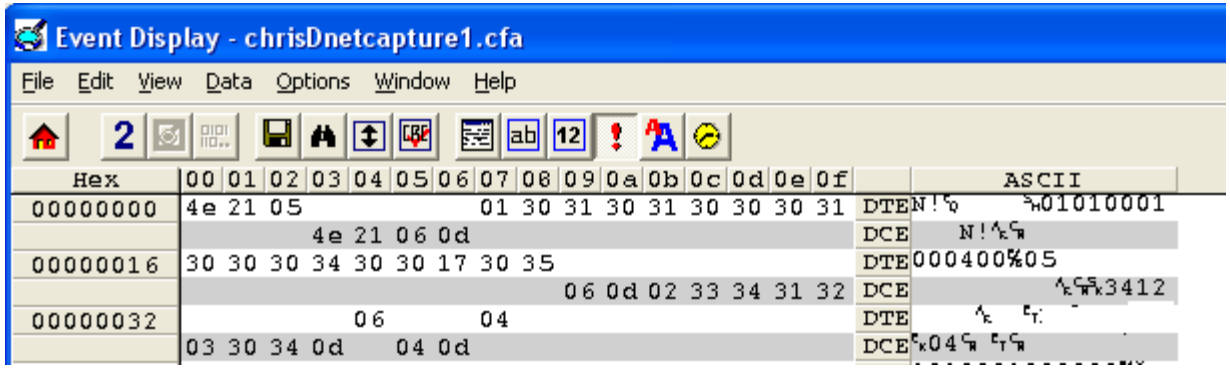


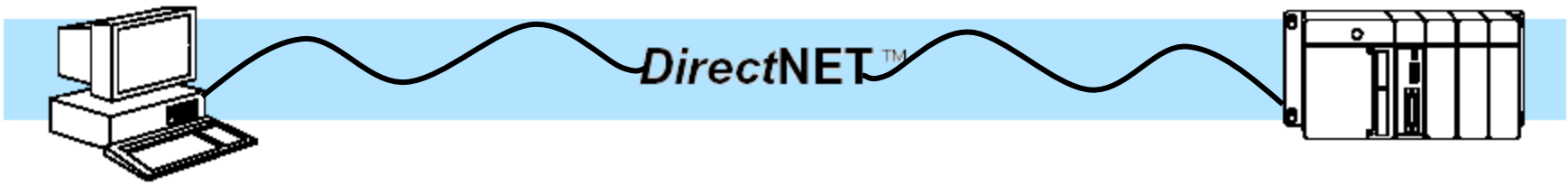


Example of how this would look on a comm analyzer

ASCII Mode

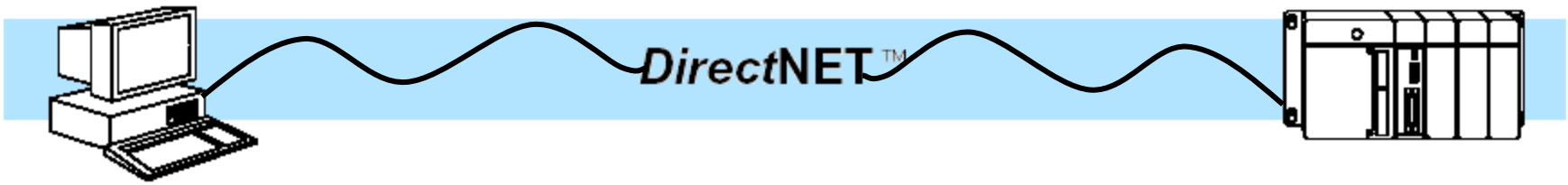






General Questions:

- **Why use ASCII Mode?**
  - No reason to use anymore. When DirectNET was originally created, the editors that were used to write code did not have the support and commands that are used today to detect the beginnings and ends of data packets as well as the conversion commands.
- **Why use Hex Mode?**
  - If reading or writing a small amount of data, there is no real advantage to using Hex Mode. If you are reading or writing a large amount of data, Hex Mode transfers half as many bytes for the same amount of data so Hex Mode will have better throughput.
- **Why is there an offset of 20 in the Enquiry?**
  - So that the address does not get interpreted as a control character. An offset of 20 brings the data above the control character range.



General Questions:

- **Why am I getting an 04(EOT) from the slave before my communications are complete?**
  - This is an indication of a Timeout. The Slave did not receive the expected data in the time allowed for this transaction so it sent an EOT to terminate this communications transaction.
- **Why am I getting an 15(NAK) from the slave?**
  - The data command sent to the PLC was formatted incorrectly or the LRC was incorrect.