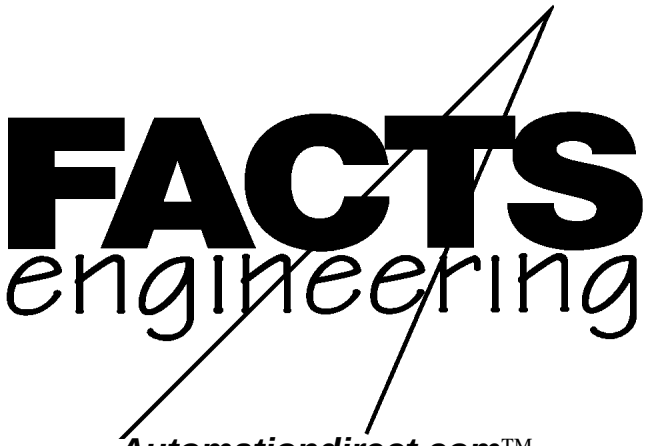


Automationdirect.com™

Direct Logic 205

Triple Port BASIC CoProcessor

F2-CP128

Order Number: F2-CP-M

TRADEMARKS

COPYRIGHT

Last Issued Date: May 1999
Current Issued Date: October 2007

WARNING

Thank you for purchasing automation equipment from FACTS Engineering. We want your new FACTS Engineering automation equipment to operate safely. Anyone who installs or uses this equipment should read this publication (and any other relevant publications) before installing or operating the equipment.

To minimize the risk of potential safety problems, you should follow all applicable local and national codes that regulate the installation and operation of your equipment. These codes vary from area to area and usually change with time. It is your responsibility to determine which codes should be followed, and to verify that the equipment, installation, and operation is in compliance with the latest revision of these codes.

At a minimum, you should follow all applicable sections of the National Fire Code, National Electrical Code, and the codes of the National Electrical Manufacturers Association (NEMA). There may be local regulatory or government offices that can help determine which codes and standards are necessary for safe installation and operation.

Equipment damage or serious injury to personnel can result from the failure to follow all applicable codes and standards. We do not guarantee the products described in this publication are suitable for your particular application, nor do we assume any responsibility for your product design, installation, or operation.

If you have any questions concerning the installation or operation of this equipment, or if you need additional information, please call us at 1-800-783-3225.

This document is based on information available at the time of its publication. While efforts have been made to be accurate, the information contained herein does not purport to cover all details or variations in hardware and software, nor to provide for every possible contingency in connection with installation, operation, and maintenance. Features may be described herein which are not present in all hardware and software systems. FACTS Engineering assumes no obligation of notice to holders of this document with respect to changes subsequently made. FACTS Engineering retains the right to make changes to hardware and software at any time, without notice. FACTS Engineering makes no representation or warranty, expressed, implied, or statutory with respect to, and assumes no responsibility for the accuracy, completeness, sufficiency, or usefulness of the information contained herein. No warranties of merchantability of fitness for purpose shall apply.

MANUAL HISTORY

Refer to to this history in all correspondence and/or discussion of this manual.

Title: 205 CoProcessors User's Manual
Part Number F2-CP-M

Issue / Date	Effective Pages	Description of Changes
First 5/99		First Edition
5/03	Ch1 Ch2	Added 250(-1) and 260 references Added 260 BMOVE Operand Table Added R portion description and examples to SHAREd Added 250/260 IEEE FP Description to SHARED Added 260 S205_ Operand Table Added R operand description and examples to S205_ Added 250/260 IEEE FP Description to S205_
11/05		Various

TABLE OF CONTENTS

CHAPTER 1 : INTRODUCTION

This manual describes details specific to the 205 BASIC CoProcessor. This document should be used to supplement the FACTS Extended BASIC User's Reference (FA-BASIC-M) when programming the FACTS Engineering 205 CoProcessor modules.

205 CoProcessor modules are installed in Slot 1 to 7 of a D2-240, D2-250, D2-250-1 or D2-260 CPU base. Slot 0 (1st I/O slot beside CPU) cannot be used. The D2-230 CPU is not supported.

The CoProcessor module communicates to the DL205 PLC CPU using the S205_, BMOVE, and SHARED instructions. A high speed dual port RAM interface, across the parallel bus of the DL205 backplane, is used for CoProcessor to PLC and PLC to CoProcessor communications. Up to 128 bytes can be transferred by the CoProcessor in one PLC scan using the BMOVE instruction. No PLC ladder logic is required for CoProcessor to PLC or PLC to CoProcessor communications. The CoProcessor does not take any X's or Y's from the DL205 PLC CPU's memory map.

The DL205 PLC ladder logic can generate an interrupt in the CoProcessor with the RX or WX ladder instructions and the ONPLC CoProcessor statement. In addition to the 128 bytes that can be transferred using the BMOVE instruction, 128 bytes can be transferred using an RX or WX triggered ONPLC interrupt.

The CoProcessor module communicates to external devices using the built in serial port(s)

DL205 CPU SYNCHRONIZATION

Upon application of power the 205 CoProcessor resets and establishes communication with the DL205 PLC CPU. Next the operating mode saved by the last AUTOSTART command is executed. Please see AUTOSTART in the FACTS Extended BASIC User's Reference for additional information.

Unlike the 305 I/O BASIC Modules, the CoProcessor does not reset when the DL205 PLC CPU is reset. If desired, the current state of the DL205 PLC CPU may be determined by examining Special Purpose relays SP11-20. See Chapter 2 (205 CoProcessor Statements) for a description of the S205_ statement. See the DL205 User's Manual for a description of DL205 PLC CPU special relays.

Example
```
10  IF S205_SP(11) THEN PRINT "Forced running state"
20  IF S205_SP(12) THEN PRINT "TERM RUN state"
30  IF S205_SP(13) THEN PRINT "TEST RUN state"
40  IF S205_SP(15) THEN PRINT "TEST PGM state"
50  IF S205_SP(16) THEN PRINT "TERM PGM state"
60  IF S205_SP(17) THEN PRINT "Forced STOP state"
70  IF S205_SP(20) THEN PRINT "PGM Mode"
```

Often a DL205 CPU control relay or stage status is used as a permissive in the BASIC program. Control relays and stage status bits are used to communicate program status information to the CoProcessor. For example, a control relay may be used to signal the start of a shift report or to simply indicate that the DL205 CPU is running.

Example
```
10  IF S205_C(0) THEN PRINT "CR 0 Energized"
20  IF S205_SG(10) THEN PRINT "Stage 10 is active"
```

CHAPTER 2 : 205 COPROCESSOR STATEMENTS

BMOVE

Function — Directly access a block of DL205 CPU memory

Syntax — BMOVE *direction*, *starting operand*(*number*), *ending operand*(*number*)
BMOVE *direction*, *starting operand*(*number*), K (*number of bytes*)

See Also — SHARED, ONPLC, and S205_

Usage — Up to 128 bytes of DL205 memory may be read or written in one scan using BMOVE. Memory in the DL205 CPU is referenced using any one of 11 different operands specified with an octal address *number*.

Block move reads begin in the ABM at shared memory location SHARED(0) and in the DL205 CPU at *starting operand*(number). Block move writes begin in the ABM at shared memory location SHARED(128) and in the DL205 CPU at *starting operand*(number). The block move continues through consecutive memory addresses up to and including *ending operand*(number). Alternately, the number of bytes to transfer may be specified as an expression in parenthesis following "K". If *number* of bytes is 0 then 128 bytes will be copied.

Use either a "R" or "W" for *direction* to specify a DL205 memory Read or Write. "R" will read DL205 CPU memory and copy to SHARED memory. "W" will read SHARED memory and copy to DL205 CPU memory.

If *starting operand* or *ending operand* is a BIT data type, the entire V-Memory address containing the operand is used.

Octal numbering and data types for BMOVE operands

240 CPU BMOVE Operands

Description	Operand	Qty	Octal numbering	Data Type	V-Memory Octal Word
Timer Current	T	128	0-177	BCD	0-177
Count Current	CT	128	0-177	BCD	1000-1177
V-Memory	VH			HEX or BCD	
Volatile		1024	2000-3777		2000-3777
Non-volatile		256	4000-4377		4000-4377
System Parameters		106	7620-7737		7620-7737
			7746-7777		7746-7777
Inputs	X	320	0-477	Bit	40400-40423
Outputs	Y	320	0-477	Bit	40500-40523
Internal Relays	C	256	0-377	Bit	40600-40617
Stage Status	SG	512	0-777	Bit	41000-41037
Timer Status	TS	128	0-177	Bit	41100-41107
Counter Status	CS	128	0-177	Bit	41140-41147
Special Relays	SP	144	0-137	Bit	41200-41205
			540-617	Bit	41226-41230

250(-1) CPU BMOVE Operands

Description	Operand	Qty	Octal numbering	Data Type	V-Memory Octal Word
Timer Current	T	256	0-377	BCD	0-377
Count Current	CT	128	0-177	BCD	1000-1177
V-Memory	VH			HEX, BCD, or	
Volatile		3072	1400-7377	Float (Double	1400-7377
		4096	10000-17777	Word)	10000-17777
System Parameters		256	7400-7777		7400-7777
		512	37000-37777		37000-37777
Inputs	X	512	0-777	Bit	40400-40437
Outputs	Y	512	0-777	Bit	40500-40537
Internal Relays	C	1024	0-1777	Bit	40600-40677
Stage Status	SG	1024	0-1777	Bit	41000-41077
Timer Status	TS	256	0-377	Bit	41100-41117
Counter Status	CS	128	0-177	Bit	41140-41147
Special Relays	SP	512	0-777	Bit	41200-41237

260 CPU BMOVE Operands

Description	Operand	Qty	Octal numbering	Data Type	V-Memory Octal Word
Timer Current	T	256	0-377	BCD	0-377
Count Current	CT	256	0-377	BCD	1000-1377
V-Memory	VH			HEX, BCD, or	
Volatile		256	400-777	Float (Double	400-777
		3072	1400-7377	Word)	1400-7377
		11264	10000-35777		10000-35777
		256	7400-7777		7400-7777
System Parameters		512	37000-37777		37000-37777
Inputs	X	1024	0-1777	Bit	40400-40477
Outputs	Y	1024	0-1777	Bit	40500-40577
Internal Relays	C	2048	0-3777	Bit	40600-40777
Stage Status	SG	1024	0-1777	Bit	41000-41077
Timer Status	TS	256	0-377	Bit	41100-41117
Counter Status	CS	256	0-377	Bit	41140-41157
Special Relays	SP	512	0-777	Bit	41200-41237

Example Load a table of 6 constants into user V-Memory starting at V2000

```
10  REM  Load the table into shared memory
20  SHARED(128)=10H
30  SHARED(130)=20H
40  SHARED(132)=25H
50  SHARED(134)=30H
60  SHARED(136)=100H
70  SHARED(138)=9798H
80  REM  Copy the table to DL205 CPU V-Memory
90  BMOVE W, VH(2000), K(12)
```

Example Multiply a range of user V-Memory by a constant value

```
10  BMOVE R, VH(2000), K(32) : REM  Get the values
20  REM  Multiply by 2.5
30  FOR ADDR = 0 TO 31 STEP 2
40  SHARED(ADDR+128)=SHARED(ADDR)*2.5
50  NEXT ADDR
60  BMOVE W, VH(2000), K(32) : REM Put the values back
```

Example Get the DL240 X (Input) image table

```
10  BMOVE R, X(0), X(477)
```

Advanced

If no operand is specified then address number is the hexadecimal representation of the Octal V-Memory address plus one (80H = Octal V-Memory 177). BMOVE R, VH(2000), K(10) is the same as BMOVE R, (401H), K(10).

This feature simplifies FOR-NEXT loops and other types of "calculated" PLC memory accesses.

Example

Find all user V-Memory locations which match a constant

```
10 K = 1234 : REM   Match value
15 REM   Search all of user V-Memory
20 FOR INDEX=401H TO 1000H STEP 127 : REM 2 BYTES/V-MEM
30 BMOVE R, (INDEX), K(127)
40 FOR ADDR = 0 TO 125 STEP 2
50 IF SHARED(ADDR)<>K THEN 70
60 PRINT1 "Matched at V-Memory hex address = ",
62 PRINT1 HEX$(INDEX+ADDR-1)
70 NEXT ADDR
80 NEXT INDEX
```

SHARED

Function — Read or write memory shared with the DL205

Syntax — SHARED (*address*, *portion*) = expression
variable = SHARED (*address*, *portion*)

Usage — SHARED (shared memory) is used in conjunction with ONPLC interrupt and BMOVE (block move) statements to access the DL205.

The SHARED operator retrieves the value at the shared memory address and assigns it to the variable.

The SHARED statement stores the value of expression at the shared memory address.

address is an expression from 0 to 387, which selects two bytes of shared memory. SHARED retrieves or assigns an integer value (0 to 65535) at address.

portion is optional and is used to specify a bit position, a nibble (group of 4 bits), a byte (group of 8 bits), a BCD word (2 bytes), or an IEEE Floating Point value (4 bytes).

Use "B(n)" to specify one of 16 bit positions, where n = 0-15.
Use "N(n)" to specify one of four nibbles, where n = 0-3.
Use "H" to specify the high byte or use "L" to specify the low byte.
Use "B" to specify a word hexadecimal to BCD conversion.
Use “R” to specify a BASIC Floating Point to 250/260 IEEE Floating Point conversion.
NOTE: Firmware version 1.06/HS or above required to use the R portion.

The first 128 bytes of shared memory, SHARED(0) to SHARED(127), are used by the BMOVE statement when reading data from the PLC. The second 128 bytes of shared memory, SHARED(128) to SHARED(255), are used by the BMOVE statement when writing data to the PLC.

The next 128 bytes of shared memory, SHARED(256) to SHARED(383), are used in conjunction with the ONPLC statement. This block of memory is accessed by the DL205 using the WX and RX instructions. The last 4 bytes of shared memory, SHARED(384) to SHARED(387), are control bytes for WX and RX (see ONPLC for a complete description).

250/260 IEEE Floating Point

Numeric Variables in the 205 CoProcessor module are stored internally as a floating point value in the range of ±1E-127 to ±.99999999E+127. The D2-250/260 CPU can store numbers as a BCD, BINary, or as an IEEE floating point value in the range of ±3.402823E+38. If you are using IEEE floating point values in the 250/260 and you want to operate on those values in the 205 CoProcessor module use BMOVE and SHARED with the R portion or S205_VR.

Example Retrieve a 4 digit BCD (0-9999) value from shared memory

```
10  REM   Put a BCD number at V-Memory 2000
20  S205_VB(2000)=1234
30  REM   Get it back with a block move
40  BMOVE R, VH(2000), K(2)
50  PRINT1 "BCD value at V-Memory 2000 =",
52  PRINT1 HEX$(SHARED(0))
```

NOTE: Use DirectSoft DataView and BCD/HEX display format to view BCD data in the PLC.

Example Store 8 digit BCD (0-99999999) values in V-Memory 2000 and 2001 using BMOVE

```
10  SHARED(128) = 1234H : REM  Constant for V-Memory 2000
20  A = 5678 : REM A Must be a BCD value from 0 - 9999
30  SHARED(130) = VAL(HEX$(A)) : REM Same as SHARED(130,B)=A
40  BMOVE W, VH(2000), VH(2001)
```

NOTE: Use DirectSoft DataView and BCD/HEX display format to view BCD data in the PLC.

Example Retrieve a Hex/Integer (0-FFFFH/0-65535d) value from shared memory

```
10  REM   Put a Hex/Decimal number at V-Memory 2000
20  S205_VH(2000)=1234
30  REM   Get it back with a block move
40  BMOVE R, VH(2000), K(2)
50  PRINT1 "Integer value at V-Memory 2000 =",
52  PRINT1 HEX$(SHARED(0))
```

NOTE: Use DirectSoft DataView and Decimal display format to view Integer data in the PLC.

Example Store a 250 Floating Point value then retrieve a value

```
10 REM Write a Float Value to V1400/1401 and Read a Float from V1410/1411
20 SHARED(128,R)= +3.402823E+38
30 BMOVE W,VH(1400),K(4) : REM Floats use 2 words/4 bytes in the 250
40 BMOVE R,VH(1410),K(4) : REM Floats use 2 words/4 bytes in the 250
50 X=SHARED(0,R)
```

NOTE: Use DirectSoft DataView and Real or Exponential display format to view IEEE Floating Point data in the PLC.

Example Using SHARED with PICK statement type modifiers

```
1000   V=1120H
1010   SHARED(128)=V :  PRINT1 "Retrieving values from SHARED"
1020   PH1. "SHARED(128) = ",V," in hexadecimal"
1030   PRINT1 "1st nibble = ",SHARED(128,N(0)), SPC (5),
1040   PRINT1 "3rd nibble = ",SHARED(128,N(2))
1050   PRINT1 "SHARED(128) in binary = "; :  FOR BT=15 TO 0 STEP -1
1060   IF SHARED(128,B(BT)) THEN  PRINT1 "1"; ELSE  PRINT1 "0";
1070   NEXT BT :  PRINT1
1080   PH1. SHARED(128),
1090   PRINT1 " or ",V," treated as BCD = ",SHARED(128,B)," decimal"
1100   HB=SHARED(128,H) :  REM   Swap the bytes
1110   SHARED(128,H)=SHARED(128,L) : SHARED(128,L)=HB
1120   PH1. "Value with bytes swapped = ",SHARED(128)
1130   PRINT1  :  PRINT1 "Assigning bits and nibbles in SHARED"
1140   SHARED(128)=0
1150   FOR BT=0 TO 15
1160   SHARED(128,B(BT))=1
1170   IF BT=8 THEN  PRINT1
1180   PH1. SHARED(128), SPC (3),
1190   NEXT  :  PRINT1
1200   SHARED(0)=0
1210   FOR N=0 TO 3
1220   SHARED(0,N(N))=0FH
1230   PH1. SHARED(128), SPC (3),
1240   NEXT  :  PRINT1
1250   PRINT1 "BCD ASSIGNMENT"
1260   SHARED(128,B)=1120
1270   PH1. SHARED(128)," = 1120"
READY
>run
Retrieving values from SHARED
SHARED(128) = 1120H in hexadecimal
1st nibble = 0      3rd nibble = 1
SHARED(128) in binary = 0001000100100000
1120H or 4384 treated as BCD = 1120 decimal
Value with bytes swapped = 2011H

Assigning bits and nibbles in SHARED
0001H   0003H   0007H   000FH   001FH   003FH   007FH   00FFH   01FFH   03FFH   07FFH   0FFFH
1FFFH   3FFFH   7FFFH   FFFFH   000FH   00FFH   0FFFH   FFFFH BCD ASSIGNMENT
1120H = 1120
```

ONPLC

Function — Ladder logic based interrupt of normal BASIC program flow

Syntax — ONPLC line number

See Also — BMOVE, SHARED, and S205_

Usage — ONPLC enables interruption of normal BASIC program flow in response to requests made by the DL205 CPU.

ONPLC specifies the beginning line number where program execution will continue when the interrupt occurs. The interrupt is delayed until the current BASIC statement is completed (Execution begins immediately if the current statement is IDLE or DELAY).

After a RETI statement is executed, execution resumes with the statement following the last statement executed before the interrupt occurred.

The ONPLC statement will enable only a single BASIC program interrupt to occur. Future ONPLC interrupts are disabled until another ONPLC statement is executed. Normally another ONPLC statement is included in the interrupt subroutine.

An ONPLC statement with a line number of 0 will disable the ONPLC interrupt.

The DL205 CPU passes data to the ABM and causes an ONPLC interrupt to occur using a Write Data To Network (WX) instruction. Up to 128 bytes of data may be transferred with one WX instruction. The data is transferred to the CoProcessor dual port locations SHARED(256) to SHARED(383). The number of bytes written is stored in SHARED(385).

Executing a DL205 RX or WX instruction will turn ON the Special Purpose Data Communication BUSY relay associated with the ABM's slot. A BASIC RETI statement resets the BUSY Relay.

Special Purpose (SP) Data Communications Relays								
Slot	0	1	2	3	4	5	6	7
BUSY	N/A	122	124	126	130	132	134	136

Example Write V-Memory to the ABM using WX

The high byte of the first load (LD) in the following example, holds the ABM's base number (0) and the slot number (0-7). The low byte contains a two digit BCD code from 1 to 90 which gets written to the ABM at SHARED(384). This value may be used as required in the application program and does not effect the execution of the WX instruction. The value loaded will be in the second stack register when the WX instruction executes.

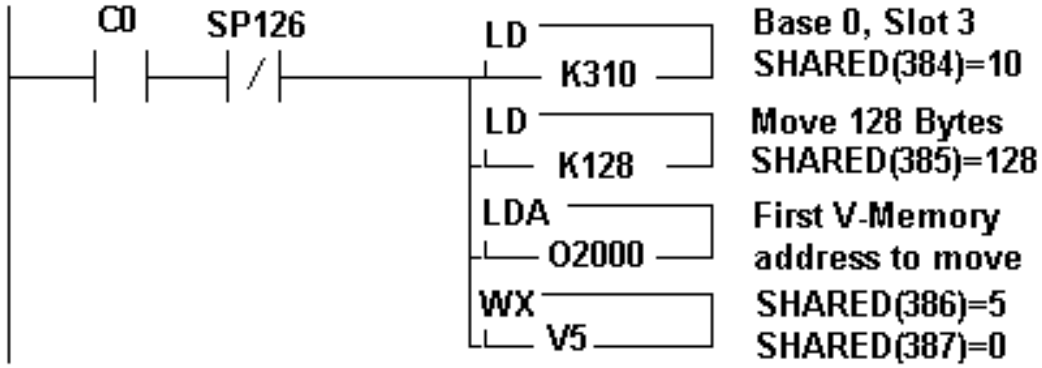
The first stack register holds the BCD number of bytes to write to the ABM. This is specified by the second LD in the following example. The byte count gets stored in the ABM at SHARED(385).

The accumulator holds the octal V-Memory source address of the data in the DL205. This is specified by the LDA instruction in the following example. Up to 128 bytes or 64 consecutive V-Memory locations may be moved to the ABM with one WX instruction. The data is stored in the ABM beginning at SHARED(256).

The address used with the WX instruction is arbitrary. This address is converted from octal to hexadecimal and is stored in the ABM, low byte first, in SHARED(386) and SHARED(387).

In the ladder logic example following, an arbitrary coil, C0 comes ON to initiate an ONPLC interrupt. SP126 is used to prevent another WX from executing while the ABM is busy.

LD K0310 directs the WX to the ABM in CPU base (base 0) of slot 3 and stores the value 10 in SHARED(384). LD K128 specifies that 128 bytes will be written. LDA O2000 specifies the source V-Memory address. Data will be moved from V-Memory 2000-2077 to SHARED(256) - SHARED(383). The WX V5 instruction sets the BUSY relay SP126, writes the data, and stores 5 in SHARED(386).




```
10   ONPLC 100
20   GOTO 10 : REM Do nothing while we wait for the interrupt
100  REM
110  REM  Start of PLC interrupt service routine - PRINT the data
120  REM
130  PRINT "Data identification codes ",
140  PRINT SHARED(384,L),SPC(2),SHARED(386)-1
150  FOR K = 0 TO SHARED(385,L)-1 STEP 2
160  PRINT "Dual Port Word ",K/2+1," = ",SHARED(256+K)
170  NEXT K
180  RETI
```

S205_

Function — Directly access DL205 CPU memory

Syntax — S205_*operand*(*number*) = expression
variable = S205_*operand*(*number*)

Shorthand — S.

See Also — BMOVE, SHARED, and ONPLC

Usage — DL205 memory may be accessed directly each scan using any one of 12 different operands specified with an octal address *number*.

The S205_ statement moves the value of expression into the DL205 memory address specified by operand(*number*). If the memory address is written to by the DL205 CPU program, the S205_ statement will be overridden.

The S205_ operator copies the value from the DL205 memory address specified by operand(*number*) into a numeric variable.

S205_ values will be BCD (VB), HEXadecimal (VH), BIT (X,Y,C) or 250/260 IEEE Floating Point (VR) data types depending on the operand used. Discrete operands such as I/O points and control relays operate on bits and return logical values. Timer and counter accumulated values are in BCD.

NOTE: Firmware version 1.06/HS or above required to use the VR operand.

The table below specifies the octal numbering and data types for each of the S205_ operands (typical VB and VH operand usage is shown).

250/260 IEEE Floating Point

Numeric Variables in the 205 CoProcessor module are stored internally as a floating point value in the range of ±1E-127 to ±.99999999E+127. The D2-250/260 CPU can store numbers as a BCD, BINary, or as an IEEE floating point value in the range of ±3.402823E+38. If you are using IEEE floating point values in the 250/260 and you want to operate on those values in the 205 CoProcessor module use BMOVE and SHARED with the R portion or S205_VR.

Octal numbering and data types for S205_ operands

240 CPU S205_ Operands

Description	Operand	Qty	Octal numbering	Data Type	V-Memory Octal Word
Timer Current	T	128	0-177	BCD	0-177
Count Current	CT	128	0-177	BCD	1000-1177
V-Memory	VH, VB,			HEX or BCD	
Volatile	or VR	1024	2000-3777		2000-3777
Non-volatile		256	4000-4377		4000-4377
System Parameters		106	7620-7737		7620-7737
			7746-7777		7746-7777
Inputs	X	320	0-477	Bit	40400-40423
Outputs	Y	320	0-477	Bit	40500-40523
Internal Relays	C	256	0-377	Bit	40600-40617
Stage Status	SG	512	0-777	Bit	41000-41037
Timer Status	TS	128	0-177	Bit	41100-41107
Counter Status	CS	128	0-177	Bit	41140-41147
Special Relays	SP	144	0-137	Bit	41200-41205
			540-617	Bit	41226-41230

250(-1) CPU S205_ Operands

Description	Operand	Qty	Octal numbering	Data Type	V-Memory Octal Word
Timer Current	T	256	0-377	BCD	0-377
Count Current	CT	128	0-177	BCD	1000-1177
V-Memory	VH, VB,			HEX, BCD, or	
Volatile	or VR	3072	1400-7377	Float (Double	1400-7377
		4096	10000-17777	Word)	10000-17777
System Parameters		256	7400-7777		7400-7777
		512	37000-37777		37000-37777
Inputs	X	512	0-777	Bit	40400-40437
Outputs	Y	512	0-777	Bit	40500-40537
Internal Relays	C	1024	0-1777	Bit	40600-40677
Stage Status	SG	1024	0-1777	Bit	41000-41077
Timer Status	TS	256	0-377	Bit	41100-41117
Counter Status	CS	128	0-177	Bit	41140-41147
Special Relays	SP	512	0-777	Bit	41200-41237

260 CPU S205_ Operands

Description	Operand	Qty	Octal numbering	Data Type	V-Memory Octal Word
Timer Current	T	256	0-377	BCD	0-377
Count Current	CT	256	0-377	BCD	1000-1377
V-Memory	VH, VB,			HEX, BCD, or	
Volatile	or VR	256	400-777	Float (Double	400-777
		3072	1400-7377	Word)	1400-7377
		11264	10000-35777		10000-35777
		256	7400-7777		7400-7777
System Parameters		512	37000-37777		37000-37777
Inputs	X	1024	0-1777	Bit	40400-40477
Outputs	Y	1024	0-1777	Bit	40500-40577
Internal Relays	C	2048	0-3777	Bit	40600-40777
Stage Status	SG	1024	0-1777	Bit	41000-41077
Timer Status	TS	256	0-377	Bit	41100-41117
Counter Status	CS	256	0-377	Bit	41140-41157
Special Relays	SP	512	0-777	Bit	41200-41237

Example Using DL205 bit data type operands:

```
10 REM   Display status on Input X4
20 IF S205_X(4) THEN PRINT1 "ON" ELSE PRINT1 "OFF"

10 REM Turn ON DL205 internal Control Relay C400
20 S205_C(400) = 1

10 REM Output Y23=OFF if CT2 is ON and X17 is OFF
20 IF S205_CS(2).AND.NOT(S205_X(17)) THEN S205_Y(23) =0
```

Example Using BCD data type operands:

```
10 REM  Display current count for CNT C10 and TMRF T0
20 PRINT1 "Counter 10 = ",S205_CT(10)
30 PRINT1 "Timer 0 = ",S205_TS(0)/100

10 REM   Divide the current count of CNT C7 by 2
20 S205_CT(7) = S205_CT(7)/2

10 REM Value from Analog Input is in V-Memory 2000
20 REM V-Memory 2001 gets the value for an Analog Out
30 REM Keep the Analog Out proportional to Analog In
40 AOUT = S205_VB(2000) * SCALE - OFFSET
50 REM Limit range of Analog Out value (0-4095)
60 IF AOUT < 0 THEN AOUT = 0
70 IF AOUT > 4095 THEN AOUT = 4095
80 S205_VB(2001) = AOUT
```

Example

Using hexadecimal data types:

```
10  REM  Display the current scan time
20  PRINT1 "Current scan time = ",S205_VH(7775)
```

Advanced

The V-Memory numbering for each operand is shown in the previous table. The VH and VB operands may be used to access any portion of V-Memory.

Display current count for CNT C0

```
>P. S.VB(1000)
```

Display status of first 16 point Input module, X0 - X17

```
>P. S.VH(40400)
```

S205_ with no operand permits hexadecimal V-Memory addressing. The V-Memory hexadecimal address is equal to the octal address + 1. S205_VH(2000) is the same as S205_(401H). This feature is useful for FOR-NEXT loops and other types of "calculated" PLC memory accesses.

CHAPTER 3 : F2-CP128 Triple Port OverDrive CoProcessor

F2-CP128 GENERAL SPECIFICATIONS

Mounting Requirement	- I/O Slot 1 to 7 in the DL205 CPU Local Base (Not Slot 0) - Up to 7 modules per DL205 CPU Local Base
Power Consumption	- 235 mA @ 5 Vdc maximum (supplied by 205 base)
Operating Environment	- 0 to 60 degrees C (32 to 140 degrees F) - 5 to 95% humidity (non-condensing)
Processor	- Dallas 80C320
Clock Speed	- 26 Mhz
User Memory	- 128K Total (64K Data, 64K Program)
Physical Connectors	- 4 Six Conductor RJ12 Plugs - Port 1 and Port 3 RS232 - Port 2 RS232 - Port 1 RS422/485 - Port 2 RS422/485
Indicator LEDs	- TXD1, RXD1, TXD2, RXD2, RTS1/TXD3, CTS1/RXD3, RTS2, CTS2
Port 1	- RS232/422/485 Selectable - 115200 Baud Maximum
Port 2	- RS232/422/485 Selectable - 57600 Baud Maximum
Port 3	- RS232 - 9600 Baud Maximum
Additional Features	- Battery Backed Calendar/Clock - Programmable from Port 1 or Port 2

F2-CP128 DESCRIPTION

This DL205 family compatible CoProcessor Module features 128K of non-volatile memory, three serial ports, real-time battery backed calendar clock, floating point math, and the FACTS Extended BASIC interpreter.

128K bytes of nonvolatile memory allows multiple program storage and execution, DL205 register expansion, and retentive data storage and retrieval.

Port 1 is a high performance 115.2K baud maximum fully configurable RS-232 or RS-422/485 serial interface. Port 2 is a 57.6K baud maximum fully configurable RS-232 or RS-422/485 serial interface. Port 3 is a 9600 baud maximum fully configurable RS-232 serial interface. All three ports have 255 character type-a-head input buffers for simultaneous communication with three or more external devices.

The real-time battery-backed calendar clock maintains time and date when power outages occur. Time based BASIC interrupts can be programmed to .010 of a second.

Floating point math solves complex formulas to 8 significant digits.

The FACTS Extended BASIC interpreter has many features and statements that simplify control oriented programming.

Program from Port 1 or Port 2 (COMMAND@)

Flexible bit manipulation instruction (BITS and PICK)

Serial port and timer interrupts (ONPORT and ONTIME)

Extensive serial port control (SETPORT, SETINPUT, PRINT, INPUT, INPLEN, INLEN)

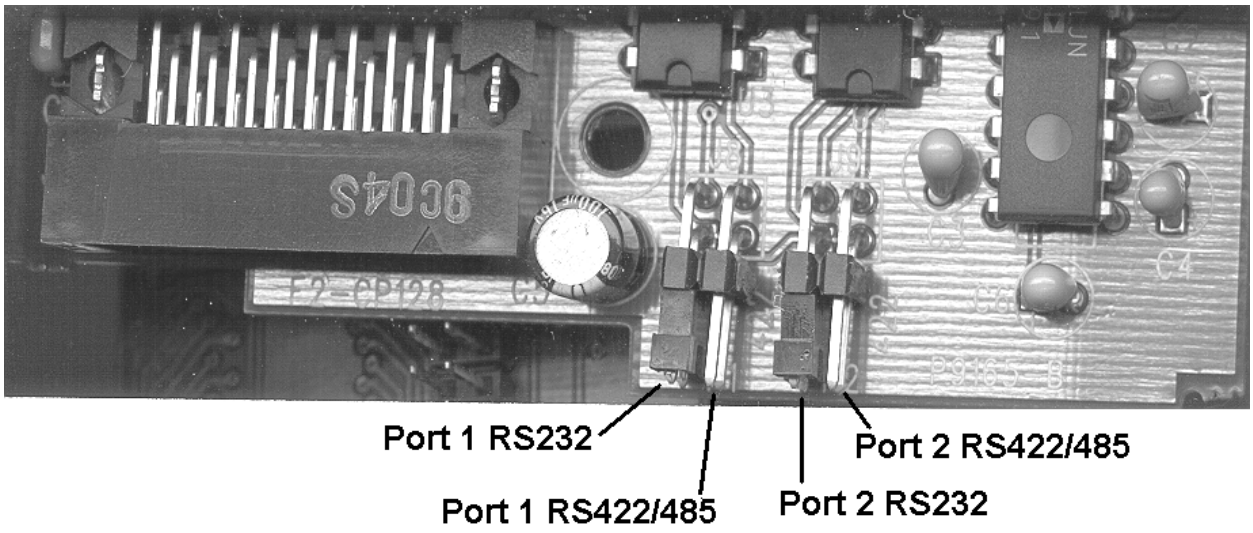
Extensive string manipulation instructions (MID$, LEFT$, RIGHT$, REVERSE$, ASC, CHR$, LCASE$, UCASE$, STR$, VAL, HEX$, OCTHEX$, DATE$, TIME$)

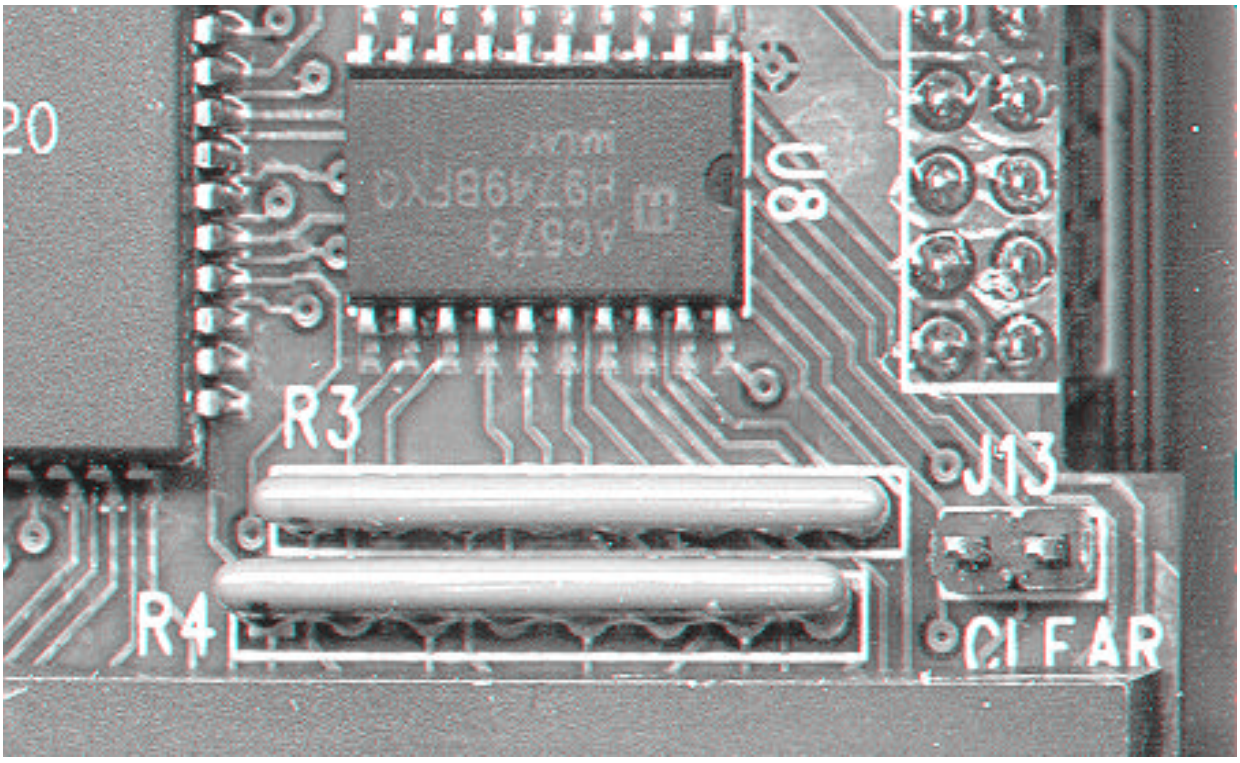
Debugging tools (TRACE, STOP, CONT)

Program chaining (GOPRM)

Statements and control structures common to most BASICs

F2-CP128 JUMPER DESCRIPTIONS AND LOCATIONS





PORT 2

The communication interface type for port 2 is selected by placing a jumper on one of the port 2 options, either RS422/485 or RS232. The RS232 selection is the default factory setting.

PORT 1

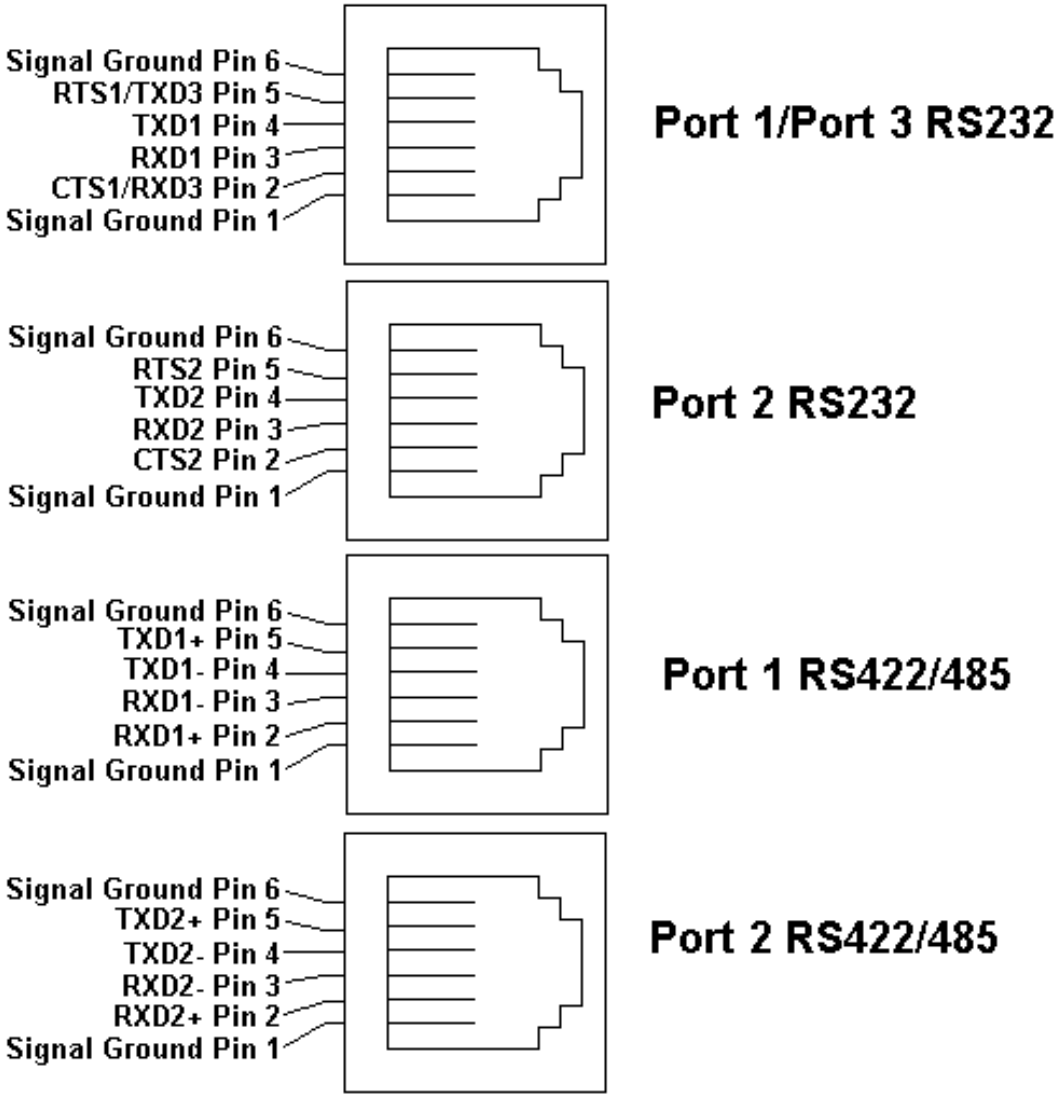
The communication interface type for port 1 is selected by placing a jumper on one of the port 1 options, either RS422/485 or RS232. The RS232 selection is the default factory setting.

CLR ALL

The CLR ALL jumper specifies the AUTOSTART mode that the module will use at reset. Placing the jumper on both posts disables AUTOSTART and waits for a space bar character in port 1. Placing the jumper on one post allows the module to use the last stored AUTOSTART parameters (this is the default factory setting).

CAUTION: Installing the CLR ALL jumper will erase program 0, all stored variables, cancel a COMMAND@2, remove LOCKOUT, and clear stored AUTOSTART information.

F2-CP128 PORT PINOUTS



APPENDIX A : QUICK START

INITIAL MODULE OPERATION USING ABM COMMANDER PLUS

1. Run ABM Commander for Windows.

2. Review the ABM Commander for Windows Help/Instructions.

3. Connect the cable from the computer to the 205 CoProcessor module.. See APPENDIX C for wiring diagrams.

4. Turn ON the power to the PLC.

5. Select 'COMMAND MODE Connect to BASIC Module' from the main window. Select 'SYstem_Stats' from the COMMAND MODE menu. The 'SYstem_Stats' button will send a SPACE BAR character so the BASIC CoProcessor can correctly calculate the baud rate.

6. The module will now respond with the sign on message.
   F A C T S E x t e n d e d B A S I C P l u s
   ...

   READY
   > (">" character indicates BASIC is in COMMAND mode)

   If you do not receive the sign on message, please follow the trouble shooting procedure in APPENDIX B.

7. The BASIC CoProcessor is now ready for programming and program upload/download.

EDITING A PROGRAM

User Action	Display Window
Select 'Auto' from the menu bar. Select Mode 0, Program 0, and Click 'OK'.	AUTOSTART 0,0 Mode = 0, Edit Program = 0 Port 1 Baud = 9600 Programming (Port 2 = 9600) (Port 3 = 9600) >
Enter the following on the 'Command Line' field 10 p. <ENTER> 65535 p. <ENTER>	>10 p. >65535 p. >
Select 'ReseT' from the menu bar. Cycling the power to the PLC will also reset the BASIC CoProcessor.	RESET F A C T S E x t e n d e d B A S I C P l u s Series 205 OverDrive CoProcessor Version 1.00/HS (c)Copyright FACTS Engineering, Inc. 1988 - 1999 AUTOSTART Mode, Program, Baud Mode = 0, Edit Program = 0 Port 1 Baud = 9600 Programming (Port 2 = 9600) (Port 3 = 9600) 0 stored programs, 65528 program storage bytes free PRM 0 READY >
Select 'List' from the menu bar. Note that mode zero uses the stored baud rate. The program in the edit buffer, PROGRAM 0, is retained during loss of power in mode zero.	list 10 PRINT1 65535 PRINT1 PRM 0 READY >

SAVING A PROGRAM

User Action	Display Window
Select 'NeW' from the menu bar.	NEW >
Enter the following on the 'Command Line' field: 10 P."MY FIRST PROGRAM" <ENTER>	>10 p. "MY FIRST PROGRAM" >
Select 'SaVe' NOTE: The F2-CP128 is shipped with a diagnostic program in PRM1 so the first SAVEd program will go into PRM2.	SAVE Saving program 2 2 stored programs, 64310 program storage bytes free PRM 0 READY >
Enter the following on the 'Command Line' field: 10 P."MY SECOND PROGRAM" <ENTER>	>10 p. "MY SECOND PROGRAM" >
Select 'SaVe'	SAVE Saving program 3 3 stored programs, 64284 program storage bytes free PRM 0 READY >

AUTO RUN MODE

User Action	Display Window
Select 'Auto' from the menu bar. Select Mode 1, Program 2, and Click 'OK'. This specifies that the BASIC CoProcessor will run program 2 after a reset.	AUTOSTART 1,2 Mode = 1, RUN (CLEAR) Program = 2 Port 1 Baud = 9600 Programming (Port 2 = 9600) (Port 3 = 9600) >
Select 'ReseT' from the menu bar. Cycling the power to the PLC will also reset the BASIC CoProcessor.	RESET MY FIRST PROGRAM PRM 2 READY >
Select 'Sel' from the menu bar. Click the 'Program 0' radio button then 'OK'.	>
Select 'List' from the menu bar. Confirm that the program in the edit buffer (PRM0) is still present.	list 10 PRINT1 "MY SECOND PROGRAM" PRM 0 READY >

DELETING A PROGRAM

User Action	Display Window
Select 'Del' from the menu bar. Enter '2' then click 'OK'. Click 'Yes' on the confirmation dialog.	DELPRM2 2 stored programs, 64309 program storage bytes free >
Select 'ReseT' from the menu bar. Cycling the power to the PLC will also reset the BASIC CoProcessor.	RESET MY SECOND PROGRAM PRM 2 READY >

CANCEL AUTO RUN MODE

User Action	Display Window
Select 'Auto' from the menu bar. Select Mode 0, Program 0, and Click 'OK'. This specifies that the BASIC CoProcessor will start up in edit mode after a reset.	AUTOSTART 0,0 Mode = 0, Edit Program = 0 Port 1 Baud = 9600 Programming (Port 2 = 9600) (Port 3 = 9600) >

CHANGING THE PROGRAMMING PORT

When interfacing to a RS-422 or RS-485 device or when communicating with two or three external devices, you can change the RS-232 programming port from Port 1 to Port 2. This is done as shown below.

User Action	Display Window
In the 'Port Select' field (Bottom Left of the Command Window) select the 'Port 2' radio button.	No Change
In the 'Port Select' field click on the 'Command Port (ABM)' button. Click 'Yes' on the confirmation dialog.	No Change
Move cable from Port 1 to Port 2 then click 'OK' on the dialog prompting the cable change.	No Change
Select 'SYstem_Stats' from the menu bar.	>

APPENDIX B : TROUBLE SHOOTING

UNABLE TO ESTABLISH COMMUNICATION WITH BASIC COPROCESSOR

1. If the Port 1 RXD LED flashes when data is entered on the terminal then go to step 2. If the LED does not flash then use a RS-232 break-out box to determine if the problem is in the cable or the computer.

2. Power off the base, remove the module, and place the "CLR ALL" jumper on both posts.

   CAUTION: Installing the CLR ALL jumper will erase program 0, all stored data, cancel a COMMAND@2, remove LOCKOUT, and clear stored AUTOSTART information.

3. Run ABM Commander for Windows.

4. Review the ABM Commander for Windows Help/Instructions.

5. Connect the cable from the computer to the 205 CoProcessor module. See APPENDIX C for wiring diagrams.

6. Turn ON the power to the PLC.

7 Select 'COMMAND MODE Connect to BASIC Module' from the main window. Select 'SYstem_Stats' from the COMMAND MODE menu. The 'SYstem_Stats' button will send a SPACE BAR character so the BASIC CoProcessor can correctly calculate the baud rate.

8 The module will now respond with the sign on message.
F A C T S E x t e n d e d B A S I C P l u s
...

READY
> (">" character indicates BASIC is in COMMAND mode)

9. Type the following command and press return.

   >AUTOSTART 0,0

10. Power off the base and remove the module. Place the "CLR ALL" jumper on a single post.

11. Install the module and power up the base. The module will now respond with the sign on message.

F A C T S E x t e n d e d B A S I C P l u s
...

READY
> (">" prompt character indicates BASIC is in COMMAND mode)

APPENDIX C : RS232 AND 422/485 WIRING DIAGRAMS

RS-232 STANDARD

RS-232-C (RS-232) is an interface standard from the Electronic Industries Association (EIA). The standard names and defines 20 communication signals, assigned to separate pins in a 25-pin connector. The five unassigned pins may carry nonstandard signals required by any individual system.

Each signal is transmitted as a positive or negative electric current between 3 and 15 volts (usually 12 volts). The signal assigned to each pin flows in one direction only. Signals output, for example, from a computer must input to a terminal, and vice versa.

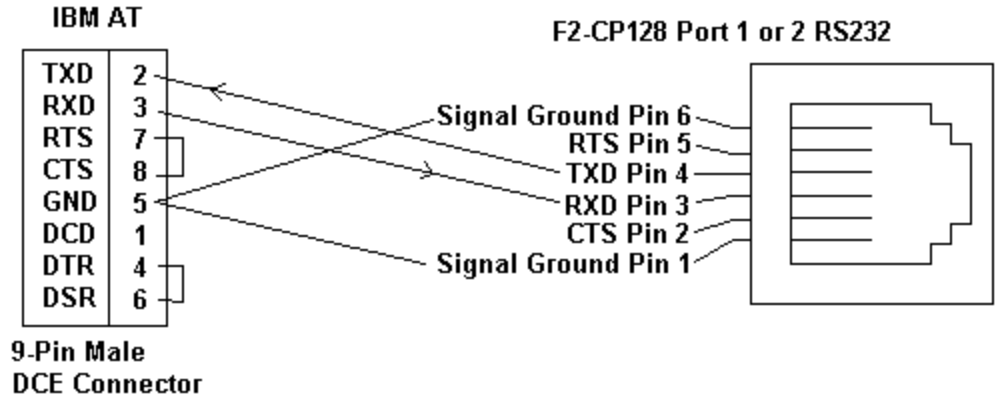
RS-232 signals travel over a serial interface cable that may have up to 25 wires. Since most signals are not required for simple communication, cables have as few as 2 or 3 wires. As shown in the following cabling diagrams, jumpers often are installed at one or both of the connectors to ensure that flow control signals are satisfied.

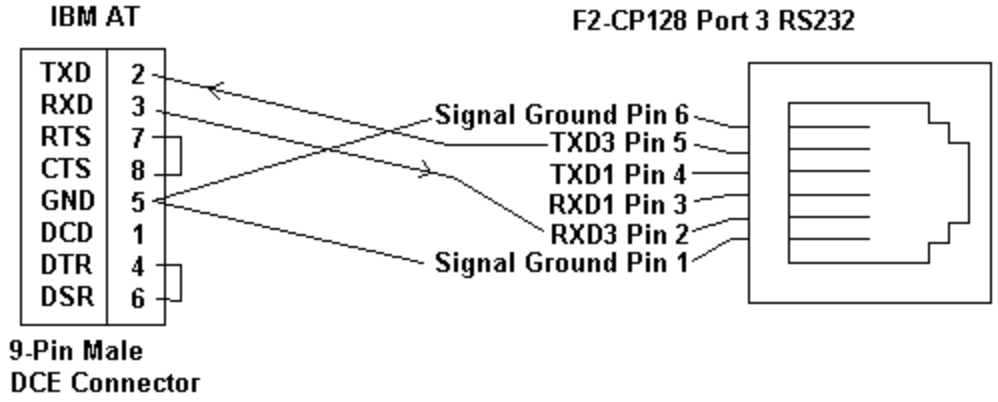
The signals flow between two types of interface ports, data communication equipment (DCE) and data terminal equipment (DTE). The pin names are the same for both DCE and DTE equipment, however, the direction of signal flow is reversed.

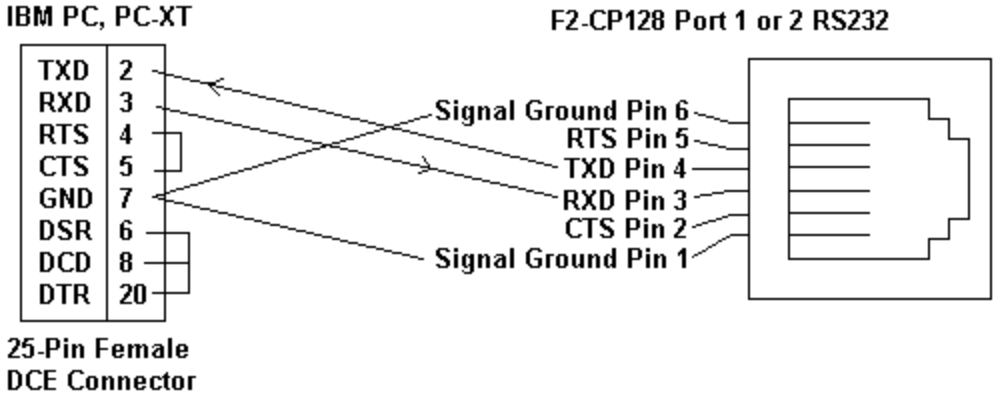
RS-232 DTE and DCE Pin Names and Signal Flow

Pin	Abrev.	Name	Signal Direction		Description
			DCE	DTE	
1	FG	Frame Ground	None	None	
2	TXD	Transmit Data	Input	Output	DTE Output Data Path
3	RXD	Receive Data	Output	Input	DCE Output Data Path
4	RTS	Request to Send	Input	Output	DTE has data to XMIT
5	CTS	Clear to Send	Output	Input	DTE may XMIT data
6	DSR	Data Set Ready	Output	Input	DCE has data to XMIT
7	SG	Signal Ground	Input	Output	
8	DCD	Data Carrier Detect	Output	Input	Modem has carrier
20	DTR	Data Terminal Ready	Input	Output	DCE may XMIT data
22	RI	Ring Indicator	Output	Input	

IBM COMPUTER CABLES



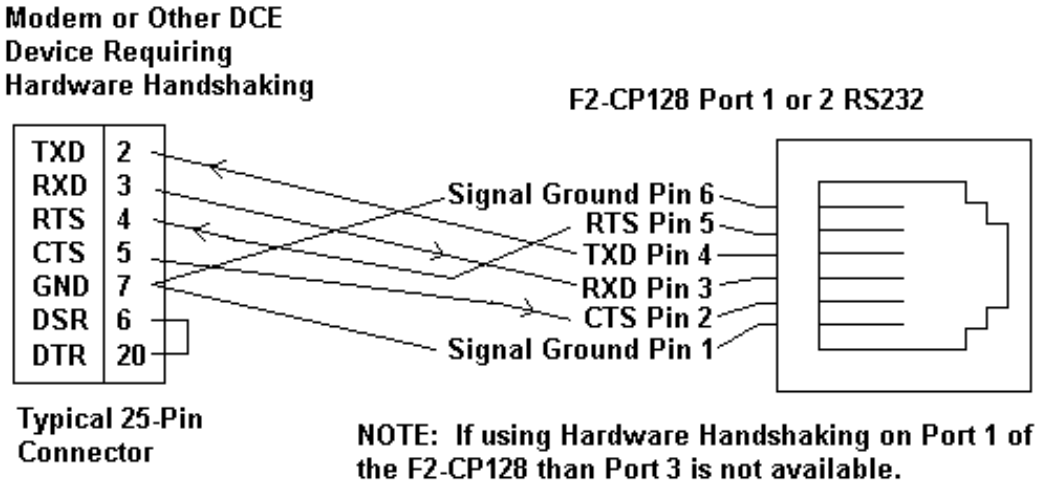


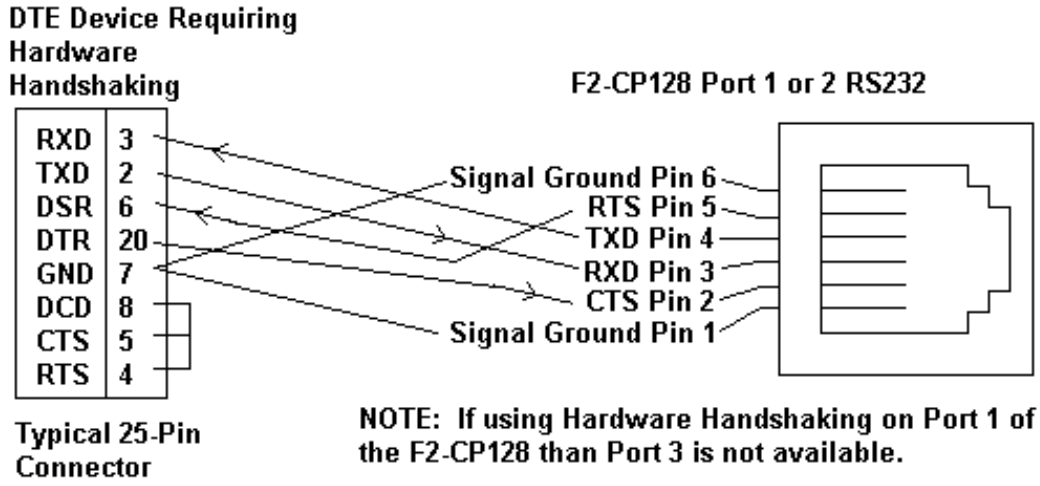


IDENTIFYING A COMMUNICATION PORT AS DCE OR DTE

With an unknown RS-232 port powered, measure the dc voltage between pin-2 and ground (pin-7) and pin-3 and ground. If the most negative pin is pin-2 then the port is DTE. If the most negative pin is pin-3 then the port is DCE. Improper connection of pins 2 and 3 will not damage the interface.

RS-232 WITH HARDWARE HANDSHAKE





RS-422/485 STANDARD

The RS-485 transceivers on CoProcessor's so equipped are compatible with both RS-422 and RS-485 signals.

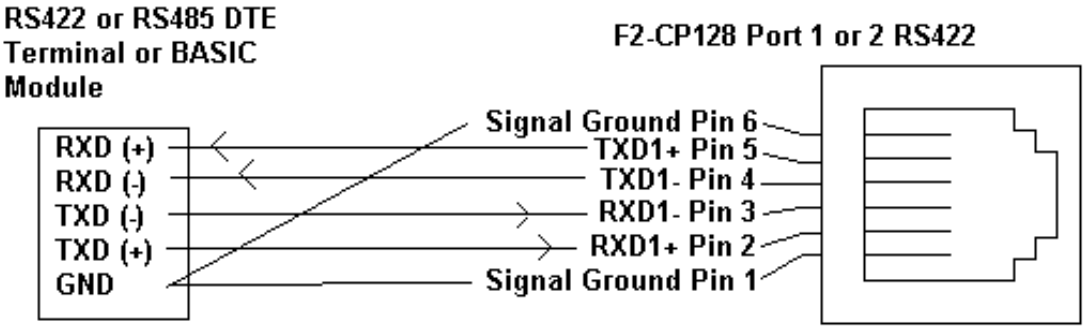
RS-422 uses high current differential outputs and is specified to 4000 feet at 10 Megabaud. Lower speed communications, such as 19.2K baud, may use substantially longer cables.

RS-485 is an upgraded version of EIA RS-422-A and offers higher current tri-state drivers which are internally protected from bus contentions caused by multiple drivers on the same line. RS-485 drivers will also withstand higher voltages on their outputs when disabled (high impedance state). RS-485 is specified for multiple transmitter and multiple receiver systems as well as single and multi-drop RS-422 applications. The RS-422 specification permits only one driver and 10 receivers on a line. The RS-485 standard allows up to 32 drivers and receivers on the same transmission line.

RS-422/485 COMMUNICATION

Most CoProcessors have one RS-422/485 communication interface some have two. To select a port for RS232 or RS422/485 data reception mode, please refer to "JUMPER DESCRIPTIONS AND LOCATIONS" in the chapter for the CoProcessor module that you are using. Transmissions from a selectable port are always available at RS-232 and RS-422/485 signal levels simultaneously.

RS-422/485 POINT-TO-POINT CABLING



RS-422/485 MULTI-DROP MADE EASY

Four wire RS-422 multiple transmitter multi-drop networks and all 2 wire RS-485 connections require that the transmitters float when not in use.

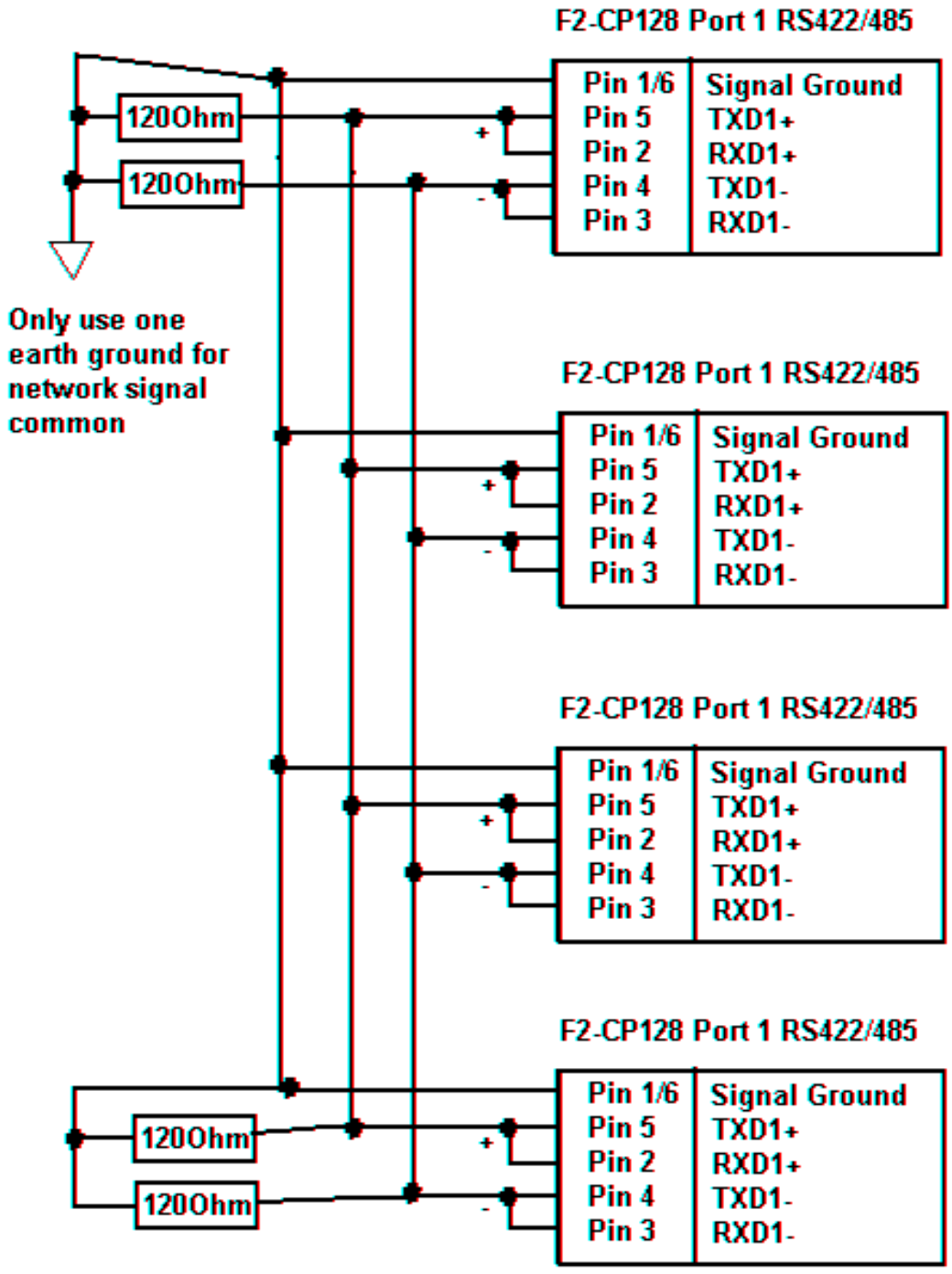
To enable the RS-422/485 transmitters only when PRINTing, use SETPORT to select multi-drop mode "M". Use the multi-drop option when the CoProcessor is a slave in a master/slave configuration or when a peer to peer configuration is required.

To leave the RS-422/485 transmitters ON even when not PRINTing, use SETPORT to select point to point mode "P". Use the point to point option when the CoProcessor is a single master in a master/slave or point to point configuration. This configuration provides the greatest noise immunity because the RS-422/485 drivers remain enabled and prevent noise from being received by the slave devices on the network.

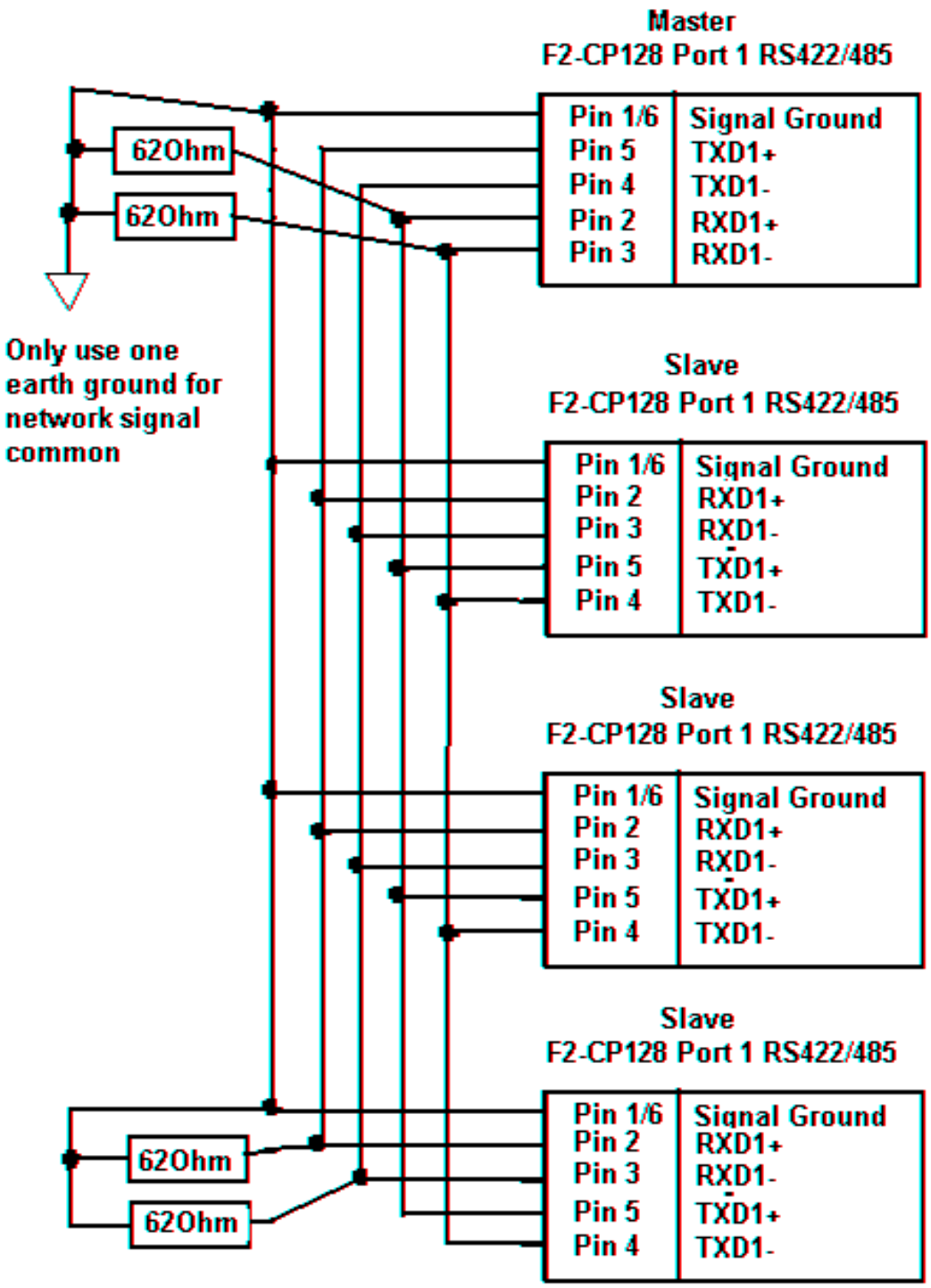
Example: Configure Port 1 for 9600 baud, no parity, 8 bit word, 1 stop bit, software XON/XOFF handshaking, and multi-drop RS-422/485 mode.

```
SETPORT 1, 9600, N, 8, 1, S, M
```

RS-485 TWO WIRE MULTI-DROP



RS-422 FOUR WIRE MULTI-DROP



Cable Shielding

Shielding improves noise immunity (magnetic field protection). It is important to ground the shield at the receiver end only. Grounding the receiver end only provides the least high frequency signal attenuation and the best rejection of unwanted signals. Grounding both ends of the shield will cause magnetic field induced noised currents to flow through ground. Noise may then appear on the data lines due to transformer like coupling with the shield. If the cable shield is used as the system ground conductor then placing a 100 O resistor in series with the shield and the ground connection will reduce noise producing ground currents.

Connecting Cables and Line Termination

A dual twisted pair plus ground connection is recommended for 4-wire RS-422 networks. Proper termination of the balanced transmission line is required to prevent data errors. A typical AWG 22 solid wire with .060 inch plastic cover, twisted 4.5 times per foot has a characteristic impedance of about 120 O. Thus the selection of the two 62 O line-to-ground terminating resistors. Line-to-ground termination is preferred to the often shown line-to-line 120 O termination. In noisy or long line applications the much better line-to-ground common-mode rejection capability is particularly important. In multidrop networks, the line must be terminated at the extreme ends only as shown in the two previous diagrams. Addition of intermediate terminations will adversely load the line. If both the transmit and receive ends of the same twisted pair are terminated, double the value of the termination resistors.

Floating Data Lines Noise Prevention

The RS-422/485 drivers at the host should remain enabled to prevent noise from being received by the slave devices on the network. To prevent noise reception at the host when there is no slave transmitting, add a pair of network biasing resistors to the host as shown in the two previous diagrams. This will pull-up the floating transmit line from the slaves to the RS-422/485 idle state (RXD+ to RXD- > .45 V). The equivalent of this can be done in a CoProcessor using the "P" parameter in the SETPORT statement.