Barcode Scanners Communication To AutomationDirect PLCs via RS232

To allow the Code barcode readers to communicate to the AutomationDirect PLCs, you must first set up the scanner. With the scanner powered up, scan the column of code that correlates with the reader's part number.

User will need to select the baud rate, number of data bits, parity, and stop bits and set up RS232 connection through the selected PLC software.

USB Connections (CRA-500-C298, CRA-C508-C298, CRA-C34-C298)



RS232 Connections (CRA-501-C298, CRA-C502-C298, CRA-C503-C298)



DB9 to RJ12 (CRA-519-C298)



CR950/CR1100/CR1500/CR5210 Configuration Codes

Configuration changes will take effect immediately and be saved to memory

Reset and Reboot

Note: When a Factory Reset is performed, a Reboot will be necessary.

Reset to RS232

Factory Defaults

Factory Reset 1A





M20112 01

1B

2B

M20111 01

Baud Rate





M20160 01





M20164 01

- M20161 01
- RS232 Interface 3B 38400 Baud Rate



M20165 01

RS232 Interface 2C 4800 Baud Rate

M20345 01

Reboot Reader



- M20162 01
- RS232 Interface 3C 57600 Baud Rate



M20166 01



Enable RS232



M20251 01



2D



M20163 01





M20167 01

Additional c	configuration codes can be found in the configuration guides as follows:
CR950	https://www.automationdirect.com/pn/doc/manual/CR950-K301-C298
CR1100	https://www.automationdirect.com/pn/doc/manual/CR1100-K201-C298
CR1500	https://www.automationdirect.com/pn/doc/manual/CR1500-K202-C298
CR5210	https://www.automationdirect.com/pn/doc/manual/CR5210-C502-C298







CR950/CR1100/CR1500/CR5210 Configuration Codes

Configuration changes will take effect immediately and be saved to memory



Code Handheld Scanner Supplemental Programming Manual - 3

https://www.automationdirect.com/pn/doc/manual/CR5210-C502-C298

CR5210

CR6022 Series Configuration Codes

Configuration changes will take effect immediately and be saved to memory

Reset and Reboot



Baud Rate





9A RS232 Interface 19200 Baud Rate



0D RS232 Interface 2400 Baud Rate



9B RS232 Interface 38400 Baud Rate



8 RS232 Interface 4800 Baud Rate



RS232 Interface 57600 Baud Rate

M10398 01







M10399 01

Additional configuration codes can be found in the configuration guide as follows:CR6022https://www.automationdirect.com/pn/doc/manual/CR6022-C500-C298

CR6022 Series Configuration Codes (continued)

Configuration changes will take effect immediately and be saved to memory

Data Bits





(RS232 Mode Only)



12C Suffix Line Feed (RS232 Mode Only)





Additional of	configuration codes can be found in the configuration guide as follows:
CR6022	https://www.automationdirect.com/pn/doc/manual/CR6022-C500-C298

Productivity PLC Example

The Productivity PLC ASCII Input Task handles the ASCII input received at the RS-232 port of the CPU. This example assumes the ASCII sending device is terminating each message with a carriage return (0xD)



The Serial Port needs to be set up. This example assumes Baud Rate = 9600, Parity = Odd, Data Bits = 8, and Stop Bits = 1, but any configuration can work.

P1-540			X
P1-540 RUN PWR Productivity	CPU Module Options Ethernet Port Remote Access	Serial Ports	
		RS-232 (RJ-12)	RS-485 (TB Style)
	Port Name	CPU-232	CPU-485
- 212 -	Port Security	Read/Write	Read/Write
	Protocol	ASCII / Custom Protocol	Modbus RTU
	Baud Rate	9600	19.2K
	Node Address		1
ATIO	Parity	Odd	Odd
TOM	Data Bits	8	8
AAU	Stop Bits	1	1
	RTS Mode	Assert During Transmit	
	RTS Off Delay Time (0-5,000 msec)	0 m	sec
	RTS On Delay Time (0-5,000 msec)	0 m	sec
	Timeout between query and response (100-30,000 msec)		5 x100 msec
	Modbus Character Timeout (1-10,000 msec)		
	Response/Request Delay (1-5,000 msec)		
	Comm Heartbeat Value (2-1,000 sec)		
Module Info			OK Cancel Help

CLICK Example

The CLICK programming example is a three-step process that (1) receives the data from the scanner and (2) places that data in holding register before (3) moving it to the final register.

Main Program

		AF
	*** THE INFORMATION PROVIDED IN AUTOMATIONDIRECT TECHNOTES IS PROVIDED "AS IS" WITHOUT A GUARANTEE OF ANY KIND. These documents are provided by our technical support department to assist others. We do not guarantee that the data is suitable for your particular application, nor do we assume any responsibility for them in your application. ***	
4		(105.)
	First Scan initializes (clears) registers and SETS the "EnableReceive" bit that enables the RECEIVE instruction.	
2		
	ASCII Input: Step 3	
	After the "ASCI_RxSuccess" subroutine, the "EnableReceive" bit is RESET. The "CycleReceiveEnable" subroutine SETS the bit once again, allowing the OFF to ON transition the RECEIVE instruction requires to execute again.	
3		Call CvcleReceiveEnable
	ASCII Input: Step 2	
	Upon the success of the RECEIVE instruction below, execute the "ASCII_RxSuccess" subroutine.	
4		Call ASCIL RYSUCCESS
	ASCII Input: Step 1	
	The EnableReceive bit is triggered automatically with the First Scan and during the "CycleReceiveEnable" subroutine. Therefore, after each successful read of the incoming ASCII string, the EnableReceive bit will be reset and set again, allowing the RECEIVE instruction to be re-enabled for the next incoming string.	
	In order to keep shorter strings from inheriting residual characters from previous strings of a longer length, TXT130 through TXT257 (128 bytes) are temporary holding registers for the incoming ASCII string and will be cleared once the data has been moved from TXT130 (128 bytes) to TXT1 (128 bytes.)	
	EnableReceive	Receive (Port2) ASCII Receiving Type Variable Inter Timeout None First Timeout None Burg Surgen None
0		Terminate Code CR
		Destination LIXI130 FirstChar LIXI130 ElistChar
		FirstErr Charloterval
		U Overflow
	Error Handling:	
	If any error is encountered during the RECEIVE instruction execution, call the "ErrorHandling" subroutine.	
6	I FirstChar II C12	Call FrontHandling
	Charleteval	

The EnableReceive bit is triggered automatically with the First Scan and during the "CycleReceiveEnable" subroutine. Therefore, after each successful read of the incoming ASCII string, the EnableReceive bit will be reset and set again, allowing the RECEIVE instruction to be re-enabled for the next incoming string.

In order to keep shorter strings from inheriting residual characters from previous strings of a longer length, TXT130 through TXT257 (128 bytes) are temporary holding registers for the incoming ASCII string and will be cleared once the data has been moved from TXT130 (128 bytes) to TXT1 (128 bytes.)

Subroutine Programs

"FirstScan" – This subroutine is called upon first scan (SC2). It initializes (clears) registers and SETS the "EnableReceive" bit that enables the RECEIVE instruction. The RECEIVE instruction is located in the main program.

	A	В	С	D	E	F	G	н		J	К		М	N	0	Р	Q	R		AF
1	A TXT129 should hold Clear TXT registers Clear DXT registers Clear DD1 which ho _Always_ON 	B a null. 1 through 12 130 through Ids the numl	C 8 with null. 257 with nu ber of chara	UII. acters recei	E ived from A	F SCII Input.	G	н	1		<u>к</u>		<u> </u>	<u>N</u>	0	P	Q	R	Copy Src Des	AF Fill 00 TXT129 00 TXT1 00 TXT128
																			 Copy Src	Fill
																			Des	00 TXT130 00 TXT257
																			Copy Src	Single DD0
																			Des	CharReceived
	SET the "EnableRed	ceive" bit. Ti	his is the b	it used to t	rigger the F	RECEIVE in	struction. 1	Therefore, tl	nis instructi	on will be a	ctive upon t	he first sca	1 .							
	RESET the "Cycle" RESET Status bits	bit. associated v	vith the RE	CEIVE inst	truction.															
2	_Always_ON																			EnableReceive
-																				Cycle [12] C20 (RST)
																				Receiving Overflow
3																				Return

"ASCII_RxSuccess" – This subroutine is called after success of the RECEIVE instruction below. Move ASCII data from the 128 bytes of temporary registers (TXT130-257) to TXT1-128. Fill the 128 bytes of temporary registers (TXT130-257) with the null from TXT129.

The RECEIVE instruction needs to see a transition from OFF to ON in order to be re-enabled. So RESET the "Success" bit and the "EnableReceive" bit, and SET the "Cycle" bit that will trigger a subroutine to SET the "EnableReceive" bit once again.

Move the "Port2 Data Length" from the system data register, SD50, to user register "CharReceived."

	A		в	С		D	E		F (3	н	1	J	К	L	М	N	0	P	Q	R		AF	
1	Move ASCII data fro Fill the 128 bytes of The RECEIVE instr RESET the "Succes SET the "Cycle" bit Move the "Pont Data _Always_ON 	from the struction of ten struction that the struction of ten struction of ten struction of the struction of	n needs bit and th will trigg	es of t egisters to see : Enat er a su m the s	(TXT13) (TXT13) leRece oroutine	y regis J-257) i ve" bit. to SET	ters (T) with the n OFF f	T130-25 null fron to ON in nableRe SD50, to	7) to TXT1-1 n TXT129. order to be ceive" bit or user registe	28. re-enable ice again rr "CharR	eceived.									<u> </u>		Copy Src Des Copy Src Des	00 TXT130 00 TXT1 00 TXT1 00 TXT130	Biock D TXT257 D TXT128 Fill XT129 D TXT257 Success B C11
																							Ena	_RST) bleReceive IBIC1
																							(Cycle BIC20 SET
																						Сору	_Port2_Rec	Single eived_Data_Le n
																						Src Des	Charf 12	SD50 Received DD1
2														 									F	teturn

"CycleReceiveEnable" – After the "ASCII_RxSuccess" subroutine, the "EnableReceive" bit is RESET. The "CycleReceiveEnable" subroutine SETS the bit once again, allowing the OFF to ON transition the RECEIVE instruction requires to execute again.

	A	B	C	D	E	F	F	G	н	1	J	K	L	M	N	0	P	Q	R	S	Т	AF
1	SET the "EnableRe <u>RESET the "Cycle"</u> <u>Always_ON</u> <u>B</u> SC1	bit that ena	nce again, t	o alllow the	OFF to	ON trans	sition the	RECEIVE	instructio	n requires	to executi	e again.										EnableReceive III C1 Cycle III C20 (RST) Return

"ErrorHandling" – This subroutine is placed here for the user to write code to handle any errors encountered from the RECEIVE instruction (First Character Timeout, Character Interval Timeout, and Overflow.)

	A B C D E F G H I J K L M N O P Q R S T U I	AF
	This subroutine is placed here for the user to write code to handle any errors encountered from the RECEIVE instruction (First Character Timeout, Character Interval Timeout, and Overflow.)	
		()(0.7.)
_		(NOP)
		Datum
2		Return

Below are the bits that are used in the code shown above:

Address	₿↓	Data Type	Nickname	Address	ĝ₽	Data Type	Nickname
C1	RW	BIT	EnableReceive	TXT1	RW	T TEXT	
C10	RW	BBIT	Receiving	TXT2	RW	T TEXT	
C11	RW	BBIT	Success	TXT3	RW	T TEXT	
C12	RW	BBIT	FirstChar	TXT4	RW	T TEXT	
C13	RW	BBIT	CharInterval	TXT5	RW	T TEXT	
C14	RW	BBIT	Overflow	TXT6	RW	T TEXT	
C20	RW	BBIT	Cycle		1.00		
SC1	R	BBIT	_Always_ON	TXT251	RW	T TEXT	
SC2	R	BBIT	_1st_SCAN	TXT252	RW	T TEXT	
DD1	RW	12 INT2	CharReceived	TXT253	RW	T TEXT	
SD50	RW	INT INT	_Port2_Received	TXT254	RW	T TEXT	
SD60	RW	INT INT	_Port3_Received	TXT255	RW	T TEXT	
				TXT256	RW	T TEXT	
				TXT257	RW	T TEXT	

The Com Port needs to be set up. This example assumes Baud Rate = 9600, Parity = Odd, Data Bits = 8, and Stop Bits = 1, but any configuration can work.

Com Port Setup Details		×
Port: Port2 Protocol: A	SCII 👻	
Basic Configuration		Wiring Details
Node Address (1-247):	1	Port2 RS-232C (Non isolation)
Baud Rate (bps):	9600 👻	
Parity:	Odd 👻	6 pin female modular. (RJ12 phone jack)
Stop Bit:	1 -	
Communication Data (bit):	8 🔻	/ ^{0V} /±5V
Advanced Configuration		
Time-out Setting;	500 ms 🔻	<u>`ov</u>
Character Time-out (2-1000ms);	2	
RTS ON Delay (0-5000ms):	0	
RTS OFF Delay (0-5000ms):	0	
Response Delay Time (0-5000ms);	0	
	ОК	Cancel Help

BRX Example

Main Program

The ASCII Input program (Pgm_ASCII_Input) handles the internal serial input. On first scan RUN this program.



Pgm_ASCII_Input

Upon initially running the program, CLEAR the internal serial port. This will delete any data in the Input Queue data storage, as well as reset the .InQueue value.



IF the character count at the internal Serial buffer (IntSerial.InQueue) is greater than zero, THEN execute the STREAMIN Instruction and populate SL0 (sInputString) element

SG – F Sta	Stage EAD_ASCII_Input pe S1	
sı 4	IF the character count at the Internal Serial buffer (IntSerial.InQueue) is greater than → Execute the STREAMIN instruction and populate SL0 (sInputString) element. IntSerial.InQueue 0 IntSerial.InQueue 0 IntSer	STREAMIN Stream in Data from Device Device GintSerial Complete when received the delimiter character 0x0D OR after a timeout of 100 ms String Structure sinputString On Success, JMP to stage ParselnputString On Error, JMP to stage ErrorHandling

This stage serves two purposes:

- 1) It allows the user to write any code required for parsing the input string and placing the parsed content to other memory elements
- 2) Once the stage is complete, a jump is executed back to stage 1, allowing for a new evaluation of the IntSerial.InQueue value.



Continued...

BRX Example (continued)

This stage is placed here for the user to write any required code for handling any errors encountered during the execution of this program.



The Serial Port needs to be set up. This example assumes Baud Rate = 9600, Parity = Odd, Data Bits = 8, and Stop Bits = 1, but any configuration can work.

ſ	Edit Serial Port Settings
	Device Name: @IntSerial
	Port Settings
	Baud Rate: 9600 💌
	Data Bits: 8
	Stop Bits: 1
	Parity: Odd 🗨
	Transmit Control: Unconditional
	RTS Control: Follows Transmitter
	OK Cancel

DirectLogic Example

The ASCII input (AIN) instruction is used to receive the ASCII data through port 2 of the DL06 PLC. This example is using the termination code 0xD. Ensure that the barcode scanner is set up to send the termination code.

1 AIN CPU/DCM Slot: CPU K0 Port Number: K2 Data2 X E4 Text 16 E4 Text 16 1 V2000 .A12345678A. 2 .A12345678A. C0 2 .A12345678A. C1 1 V2000 .A12345678A. 2	<u> </u>	The ASCII Input (AIN) Instruction is used to receive the ASCII data through port 2 of the DL06 PLC. This example is using the termination code 0xD. Ensure the barcode scanner is setup to send the termination code.	4	Þ ×
Element Status 1 V2000	1	Data2	AIN CPU/DCM Slot: CPU/DCM Slot: C	
		Element Status 1 V2000 .A12345678A. 2	Overflow Error : C10 Busy : C0 Complete : C1 Interchar. Timeout Error : n/a First Char. Timeout Error : n/a	

AIN instruction setup

√ פ			0
AIN			
Length Type	CPU/DCM : -	Byte Swap :	
C Fixed Length	CPU	C None	
Variable Length	O DCM	O All	
		All but null	
Slot Number :	KO	Termination Code Le	ength
Port Number :	K2 •	I Character	
Data Destination :	V2000 •	C 2 Characters	
		TermCode 1: 0D	• hexadecimal
* Data Destination = B * Data Destination + 1	yte count = Start of data	TermCode 2 : 00	hexadecimal
Maximum Variable Length :	К32 •	Overflow Error :	C10 •
Interchar. Timeout :	None 💌	Busy :	C0 •
First Char. Timeout :	None 💌	Complete :	C1 •
		Interchar, T/O Error :	C0
		First Char. T/O Error :	C0

Secondary COM port setup

Setup Communication Ports	×
Port: Port 2	Close
Protocol:	Base Timeout: Norman State Sta
DirectNET	800 ms Help
I Non-Seq(ASCII) I Remote I/O	3 Characters (3.44 ms)
Time-out: Base Timeout	-
RTS on delay time: 2 ms	XON/XOFF flow control DIC flow control
RTS off delay time: 2 ms	I I S now control
Data bits: 8 🗨	
Baud rate: 9600 💌	Echo Suppression
Stop bits: 1	 RS-4227485 (4-wire) RS-232C (2-wire)
Parity: Odd 🗨	C RS-485 (2-wire)
Memory Address: TA0	
]	
Port 2: 15 Pin	

DirectLogic Example (continued)

Components used:

- PLC: D0-06AR (using serial port 2)
- ZL-CMA15L (connected to port 2 of the D0-06AR)
- ZL-RTB-RJ12 (the RJ12 cable from the barcode reader connects here and is then wired to the ZL-CMA15L)



C-more HMI Examples for Code Readers

Compatibility			
Scanner	C-more EA7*	C-more EA9	C-more CM5
CR950-K301-C298	Yes	Yes with powered USB hub	Yes
CR1100-K201-C298	Yes	Yes with powered USB hub	Yes
CR1500-K201-C298	Yes	Yes with powered USB hub	Yes
CR5210-CC500-C298	Yes	Yes with powered USB hub	Yes
CR2701-200-C298	Yes**	Yes**	Yes**
CR2702-200-C298	Yes**	Yes**	Yes**

* Please note that the C-more EA7 is no longer in active production.

** Please note that although the CR270x will work with the HMI, the HMI will not provide enough current to charge the scanner. However, the scanner battery can always be charged using an external charger such as the CRA-A274-P1-C298 quad bay battery charger.

There are two ways to get the Code Reader to communicate with the C-more HMI

1. Enable Alternate Operating System (EA9 and CM5 only)

Alternate Operating System ON.



M20305 01

Alternate Operating system OFF.



M20306_01

2. Declare Enumeration (all C-more HMIs)

Declare Enumeration when addressed.



CC004640_01

In all of the above examples, depending on the application, it may become necessary to add an Enter to the suffix by scanning the following:

Suffix Enter (USB keyboard only)



M20219_02

Erase Suffix Data



M20208_01

Additional configuration codes can be found in the configuration guides as follows:		
CR950	https://www.automationdirect.com/pn/doc/manual/CR950-K301-C298	
CR1100	https://www.automationdirect.com/pn/doc/manual/CR1100-K201-C298	
CR1500	https://www.automationdirect.com/pn/doc/manual/CR1500-K201-C298	
CR5210	https://www.automationdirect.com/pn/doc/manual/CR5210-C500-C298	

C-more HMI Examples for CR6022 Series

Compatibility		
Scanner	C-more EA7*	C-more EA9
CR6022-C500-C298	Yes	No

* Please note that the C-more EA7 is no longer in active production.

There are three ways to get the CR6022 to communicate with the EA7 HMI

1. Alternate Operating System ON.



Alternate Operating System OFF.



2. Declare Enumeration after the Get Report descriptor.



M10123_02

3. Declare Enumeration after Set Configuration.



In all of the above examples, depending on the application, it may become necessary to add an Enter to the suffix by scanning the following:

Suffix Enter (USB keyboard only)



Erase Suffix Data (Default)



Additional configuration codes can be found in the configuration guide as follows:		
CR6022	https://www.automationdirect.com/pn/doc/manual/CR6022-C500-C298	