

Serial Protocol Specification for the Communication between Programmable Logical Controller PLC and MDB Vending Peripherals via SPS2MDB Interface.

This protocol specification was initially defined by BonusData AG on February 15th, 2006 and published as SPS2MDB document, Version 1.00

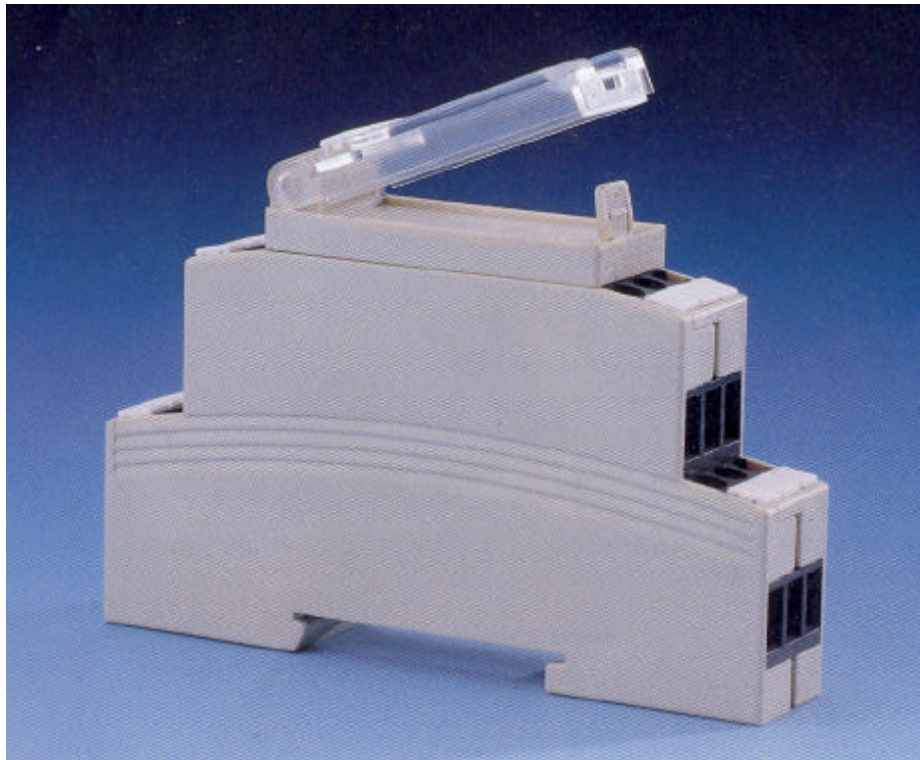
Copyright © 2006-2008 by BonusData AG, Tiergartenweg 1, CH-4710 Balsthal.

<http://www.bonusdata.net>

Mailto: team@bonusdata.net

Phone: +41 62 3919600

Fax. +41 62 3919601



Contents:

CHAPTER 1: BASICS	4
<i>Introduction</i>	4
<i>Basic conditions</i>	4
<i>Electrical connection</i>	4
<i>Operating states</i>	5
<i>Start Condition</i>	5
<i>Stop Condition</i>	5
<i>Transmitting one byte</i>	5
<i>Transmitting more than one byte</i>	6
<i>Synchronization</i>	6
<i>Timeout</i>	6
CHAPTER 2: VEND SEQUENCE	8
<i>Introduction</i>	8
<i>Simplified vend procedure</i>	8
CHAPTER 3: COMMANDS	9
<i>Command: Status Request (0xA0)</i>	9
<i>Command: Enable (0xA1)</i>	9
<i>Command: Disable (0xA2)</i>	10
<i>Command: Begin Session (0xA3)</i>	10
<i>Command: Vend Request (0xA4)</i>	10
<i>Command: Vend Approved (0xA5)</i>	10
<i>Command: Vend Denied (0xA6)</i>	11
<i>Command: Vend Success (0xA7)</i>	11
<i>Command: Vend Failure (0xA8)</i>	11
<i>Command: End Session (0xA9)</i>	11
<i>Command: Cancel Session (0xAA)</i>	11
<i>Command: Device Reset (0xAB)</i>	12
CHAPTER 4: APPLICATION SPECIFIC FUNCTIONALITY	13
<i>Introduction</i>	13
<i>Setup Of Prices</i>	13
<i>Multi Vend</i>	13
CHAPTER 5: HARDWARE DESCRIPTION	14
<i>SPS2MDB Technical Data</i>	14
<i>SPS2MDB to PLC Interconnection</i>	14
CHAPTER 6: SOFTWARE DOWNLOAD	15
<i>Connecting Hyper Terminal</i>	15
<i>Monitor</i>	16
<i>Download of a new software release</i>	16
CHAPTER 7: SETUP OF A TEST APPLICATION	17
FIGURE: 4	17
TEST SEQUENCES FOR SPS/PLC COMMUNICATION COMMANDS	17
<i>Status</i>	17
<i>Enable / Disable</i>	17
<i>Enter session</i>	17
<i>Enter vend / Vend denied</i>	18
<i>Enter vend / Vend failed</i>	18
<i>Enter vend / Vend success</i>	18
<i>Enter vend / Vend approved (No further coins accepted)</i>	18
<i>Escrow (Acceptor only)</i>	18
<i>Remove card (Reader only)</i>	18
ALLOWED COMMANDS WITHIN CERTAIN STATES	18
<i>Command / State</i>	18
CHAPTER 8: APPENDICES	20
A: Figures	20

B: Document History.....20
C: Glossary.....20

Chapter 1: Basics

Introduction

The project name for ‚Connect MDB Vending Peripherals to a Programmable Logical Controller (PLC)‘ is SPS2MDB. Where SPS is the shortcut for ‚Speicher Programmierbare Steuerungen‘ like PLC stands for ‚Programmable Logical Controller‘.

Because PLC devices often do not have a RS-232 interface, or if there is one, it is quite difficult or even impossible to implement the MDB protocol programmatically. Even then a electrical interface from RS-232 to MDB (Multi Drop Protocol) of the vending peripherals. So it was obviously to design a direct interface from standard digital inputs and outputs to MDB. This way the interface can be used with any PLC with a few spare inputs and outputs. Thus allows to transfer small amounts of data at low baud rate in both direction.

MDB is a ‚Multi Drop Protocol‘ mainly used in vending machines for the internal communication among the vending machine controller VMC and the peripherals like cash less payment system, coin acceptors, bill validator, credit card reader etc.

It is not part of this document to define all the needed error handling but to specify layer one and two of the OSI model.

Basic conditions

- The PLC needs to have two available inputs and two outputs to connect a SPS2MDB device.
- The standard voltage is 24VDC.
- Passive is defined as near 0V equal a logical ‚0‘.
- Active is define as near 24VDC equal I logical ‚1‘.
- In each direction there is a data line and a clock line.
- The device starting the communication with a start condition is master for this data transfer session and must end the communication by a stop condition.
- The devices synchronize each other with the clock and acknowledge.
- Numbers in parentheses (X) are states in diagram #1.

Electrical connection

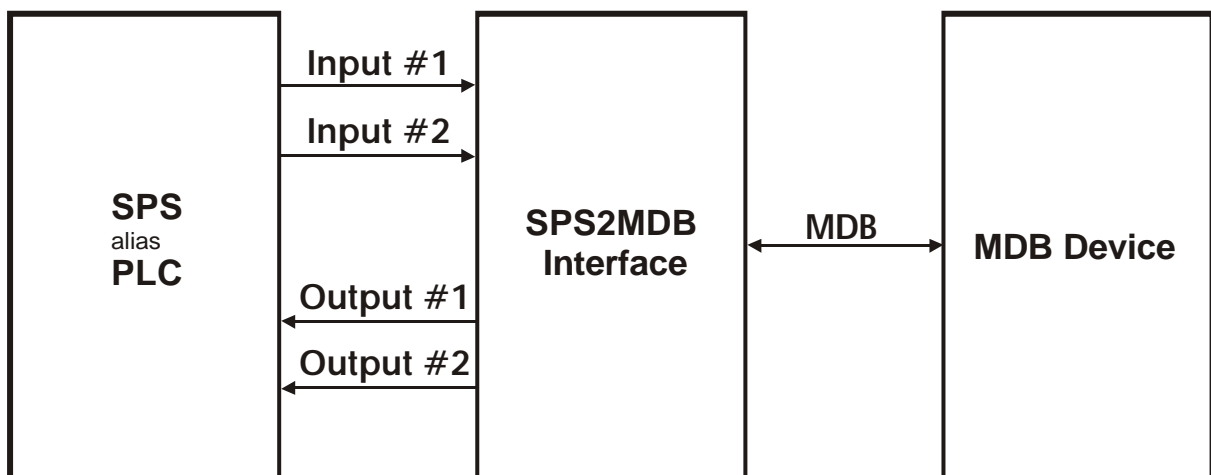


Figure 1

Operating states

- a) No device connected (1):
 - Inputs 'Passive' (not connected).
 - Outputs 'Active'.
- b) Device connected and idle (2):
 - Inputs 'Active' by connected device.
 - Outputs 'Active'.
- c) Device connected and communication:
 - Clock 'Passive' -> Invalid data.
 - Clock 'Active' -> Valid data must be present.

Start Condition

See diagram #1:

The start condition consists of following sequence:

- The device set its data line 'Passive' (3).
- The opposite device set its data line 'Passive' -> Acknowledge to master (4).
- The opposite device set its clock line 'Passive' (4).
- The device set its clock line 'Passive' (5).
- The device is now master and the opposite device is now slave (5).
- The opposite device sets clock line 'Active' (6).

Stop Condition

See diagram #1:

The stop condition consists of following sequence:

- The master set its clock line 'Active' (12).
- The slave set its clock line 'Passive' -> Acknowledge to master (13).
- The master set its data line 'Active' (14).
- The slave set its data line 'Active' -> Acknowledge to master (15).
- The slave set its clock line 'Active' -> The devices are now idle (16).
- Both devices are now idle and ready for a new start condition.

Transmitting one byte

See diagram #1:

1. The master starting the communication with a start condition:
 - The device set its data line 'Passive' (3).
 - The opposite device set its data line 'Passive' -> Acknowledge (4).
 - The opposite device set its clock line 'Passive' (4).
 - The device set its clock line 'Passive' (5).
 - The device is now master and the opposite device is now slave (5).
 - The opposite device sets clock line 'Active' (6).
2. The master transmits 8 bits with following sequence in a loop:
 - According the value of the most significant data bit, the master set its data line 'Active | Passive' (6).
 - The master set its clock line 'Active' (7).
 - The slave reads the value of the data line 'Active | Passive'.
 - The slave set its clock line 'Passive' -> Acknowledge to master (8).
 - The master set its clock line 'Passive' -> Acknowledge to slave ((9).
 - The slave set its clock line 'Active' -> Acknowledge to master (10).
3. The above sequence is repeated for the following data bits.
4. The master ends the communication with a stop condition:
 - The master set its clock line 'Active' (12).
 - The slave set its clock line 'Passive' -> Acknowledge to master (13).

- The master set its data line 'Active' (14).
- The slave set its data line 'Active' -> Acknowledge to master (15).
- The slave set its clock line 'Active' -> The devices are now idle (16).
- Both devices are now idle and ready for a new start condition.

Transmitting more than one byte

To transmit more than one byte the following sequence is used:

1. Start condition
2. Transmitting byte #1 (MSB)
3. Transmitting byte #2
4. Transmitting byte #3
5. Transmitting byte #4 (LSB)
6. Stop condition

Important Note:

See diagram #1:

The slave can detect whether the master will transmit a next byte at (14).

- If the master set its data line 'Active' -> It is stop condition.
- If the master set its clock line 'Passive' -> It is the first bit of the following data byte.

Synchronization

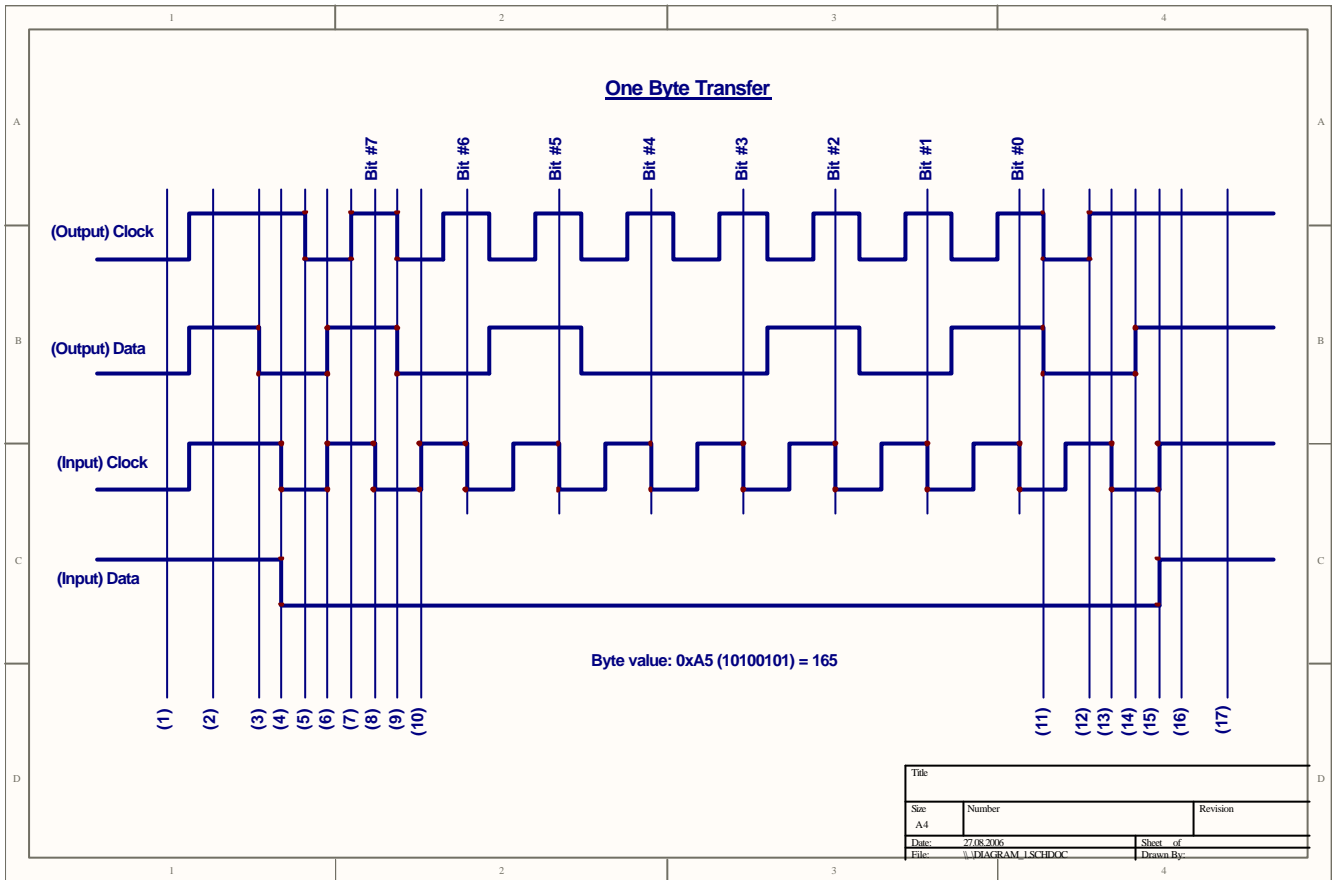
The synchronization is mainly given by the clock lines. Both devices are always waiting for the clock state transition -> Acknowledge of the opposite device. This results in a dynamic adapting of the communication speed. The start and stop conditions are synchronized by the data line transitions.

Timeout

It is strongly recommended to implement two timeouts to avoid dead locks:

1. The synchronization timeout. Whenever the master issues a line state transition it should start the acknowledge timeout of approximately 100ms. If the expected acknowledge can not be detected within this timeout the device must go into idle state and issue an error condition. The same applies for the slave if an acknowledge is expected.
2. The command response timeout. Whenever the master sends a command to the slave with an expected response it should start a response timeout of approximately 1000ms. If the expected response is not received within this timeout the command should be treated as failed and a error condition is returned to the application.

Diagram #1



Chapter 2: Vend sequence

Introduction

This chapter explains the logical communication between PLC and SPS2MDB interface.

Following payment devices are implemented:

- MDB Change Giver or MDB Coin Acceptor
- MDB Card Reader
- MDB Bill Acceptor (Optional)

The SPS2MDB interface offers a unique logical interface against to PLC independent of the connected payment devices. Therefore a simplified transaction procedure has to be defined. The current simplified implementation allows just a single vend. But future releases should also consider to implement a multi vend implementation.

Simplified vend procedure

PLC	Action	SPS2MDB
	(a) Wait for a coin or card insert...	
	Credit available because a coin accepted.	
	<----- 'Begin Session', Credit: 0-65535 units ----->	
	(b) Wait for a product selection...	
	----- 'Vend Request', Price: 0-65535 units ----->	
	(c) Deduct price of the credit...	
	Deduction success, confirm with 'Vend Approved'.	
	<----- 'Vend Approved' ----->	
	(d) or...	
	Deduction failed, confirm with 'Vend Denied'.	
	<----- 'Vend Denied' ----->	
	<----- 'Cancel Session' ----->	
	Terminate transaction, back to (a)...	
	(e) Wait for end of vend...	
	----- 'Vend Success' ----->	
	<----- 'Cancel Session' ----->	
	Terminate transaction, back to (a)...	
	(f) or...	
	----- 'Vend Failure' ----->	
	<----- 'Cancel Session' ----->	
	Terminate transaction, back to (a)...	

Note: Units define the lowest currency value like 'Cents'.

Chapter 3: Commands

Each command can be send either with the SPS interface (2 Inputs / 2 Outputs) or over the serial RS-232 monitor interface. The contents and functions are equal. If a command is send over the serial interface it must be placed inside a frame.

The parameters for this Interface are: 9600 Baud, 8 Data bit, 1 Stop bit, No Parity, No handshake.

The command frame:

STX (0x02)	Length (3/6)	Command [Data]
------------	--------------	----------------

See commands below for the commands and optional data to place inside the frame.

If we look at the status request below for example, the serial command would be:

----- 0x02 - 0x01 - 0xA0 -----> (3 bytes)

The response is then:

<----- 0x02 - 0x01 - 0x00 ----- (3 bytes)

Note: The disadvantage of using the serial interface it that the monitor for debugging and configuration is not available any more.

Command: Status Request (0xA0)

The command 'Status Request' returns the current status of the SPS2MDB interface.

Transmit: 1 Byte

PLC	Action	SPS2MDB
	----- 'Status Request' (0xA0) ----->	

Receive: 1 Byte

PLC	Action	SPS2MDB
	<----- 'Status' (Status) -----	

Status:	Bit 0:	'L' = Disabled 'H' = Enable
	Bit 1:	'L' = No Session 'H' = Active Acceptor Session
	Bit 2:	'L' = No Session 'H' = Active Reader Session
	Bit 3:	Reserved, Always 'L'
	Bit 4:	Reserved, Always 'L'
	Bit 5:	'L' = No Vend 'H' = Vend in process
	Bit 6:	Reserved, Always 'L'
	Bit 7:	'L' = Okay 'H' = Error, command could not be processed.

Command: Enable (0xA1)

The command 'Enable' enables the connected payment systems to accept coins or cards.

Transmit: 1 Byte

PLC	Action	SPS2MDB
	----- 'Enable' (0xA1) ----->	

Receive: 1 Byte

PLC	Action	SPS2MDB
	<----- 'Status' (Status) -----	

See 'Status Request'.

Command: Disable (0xA2)

The command 'Disable' disables the connected payment systems to reject any coins or cards.

Transmit: 1 Byte

PLC	Action	SPS2MDB
	----- 'Enable' (0xA2) ----->	

Receive: 1 Byte

PLC	Action	SPS2MDB
	<----- 'Status' (Status) -----	

See 'Status Request'.

Command: Begin Session (0xA3)

The command 'Begin Session' informs the PLC that there is some credit available and a vend sequence can be accomplished.

Note: If an Acceptor is connected a 'Begin Session' is send as soon as the first coin is accepted. If any further coins are inserted 'Begin Session' is send again with the new total amount available. The command 'Disable' can be used to reject any coins during the vend process.

Transmit: 4 Byte

PLC	Action	SPS2MDB
	<----- 'Begin Session' (0xA3, Price list, Credit) -----	

Price List:	Bit 0..3:	Price list No. 0..15 (Optional: With card reader only)
	Bit 4..7:	Not used, always 'L'
Credit:	Bit 0..15:	0..65535 units (2 Bytes)

Receive: 1 Byte

PLC	Action	SPS2MDB
	<----- 'Status' (Status) -----	

See 'Status Request'.

Command: Vend Request (0xA4)

The consumer has pressed a product selection button and the price is known at this moment. The PLC request to deduct the price of the credit.

Transmit: 4 Byte

PLC	Action	SPS2MDB
	----- 'Vend Request' (0xA4, Selection, Price) ----->	

Selection:	Bit 0..7:	Selectet product number: 0..255 (1 Byte)
Price:	Bit 0..15:	Price of the selected product: 0..65535 units (2 Bytes)

Receive: 1 Byte

PLC	Action	SPS2MDB
	<----- 'Status' (Status) -----	

See 'Status Request'.

Command: Vend Approved (0xA5)

The requested price has been deducted of the credit and the PLC can start a vend sequence.

Transmit: 1 Byte

PLC	Action	SPS2MDB
-----	--------	---------

<----- 'Vend Approved' (0xA5) ----->

Command: Vend Denied (0xA6)

The requested price could not be deducted of the credit and a vend is not possible.

Transmit: 1 Byte

PLC	Action	SPS2MDB
-----	--------	---------

<----- 'Vend Denied' (0xA6) ----->

Command: Vend Success (0xA7)

The vend sequence successfully finished.

Note: Use 'Cancel Session' to payout the change or inform the consumer to remove the inserted card. Or send an other 'Vend Request' to start a new vend cycle.**Transmit: 1 Byte**

PLC	Action	SPS2MDB
-----	--------	---------

----- 'Vend Success' (0xA7) ----->

Receive: 1 Byte

PLC	Action	SPS2MDB
-----	--------	---------

<----- 'Status' (Status) ----->

See 'Status Request'.

Command: Vend Failure (0xA8)

The vend sequence failed. Undo the complete transaction as far as possible.

Transmit: 1 Byte

PLC	Action	SPS2MDB
-----	--------	---------

----- 'Vend Failure' (0xA8) ----->

Receive: 1 Byte

PLC	Action	SPS2MDB
-----	--------	---------

<----- 'Status' (Status) ----->

See 'Status Request'.

Command: End Session (0xA9)

The current session has finished.

Transmit: 1 Byte

PLC	Action	SPS2MDB
-----	--------	---------

<----- 'End Session' (0xA9) ----->

Command: Cancel Session (0xAA)

Cancel the current session.

Note: If an Acceptor is connected, this command will payout any remaining amount. If a Reader is connected, its behavior depends on the specific reader implementation.

Transmit: 1 Byte

PLC	Action	SPS2MDB
------------	---------------	----------------

----- 'Cancel Session' (0xAA) ----->

Receive: 1 Byte

PLC	Action	SPS2MDB
------------	---------------	----------------

<----- 'Status' (Status) -----

See 'Status Request'.

Command: Device Reset (0xAB)

Reset the active MDB engines. This is just a software reset and will reinitialize the MDB configuration.

Transmit: 1 Byte

PLC	Action	SPS2MDB
------------	---------------	----------------

----- 'Device Reset' (0xAB) ----->

Receive: 1 Byte

PLC	Action	SPS2MDB
------------	---------------	----------------

<----- 'Status' (Status) -----

See 'Status Request'.

Chapter 4: Application Specific Functionality

Introduction:

Here are some hints for supporting the implementation of the custom specific application.

Setup Of Prices

The prices for the different product selections must be managed by the PLC. If a closed system debit card reader is connected it is possible to implement a set of price lists. The price list control can be accomplished by the price list parameter of 'Begin Session'.

Multi Vend

Multi vend functionality is not implemented in the current version. Multi vend allows several vend sequences within one session, e.g. inserting a big valued coin and make a few selections before ending the session. Here an additional action is required to end the session, like pressing the reject button of the change giver.

Chapter 5: Hardware description

SPS2MDB Technical Data

Power supply:	24VDC regulated +/-20%
Power consumption:	Average: 70 mA, Max. 120 mA
Temperature range:	-20? to +50? celsius
Dimensions:	W x L x H: 18 x 90 x 60 mm
Case material:	PC 30% GF UL 94-V1
Protection:	IP 20
Weight:	65 Grams
Mounting:	On standard 35mm DIN rail
Max. load per output:	200 mA
Input current:	Typical 20 mA

SPS2MDB to PLC Interconnection

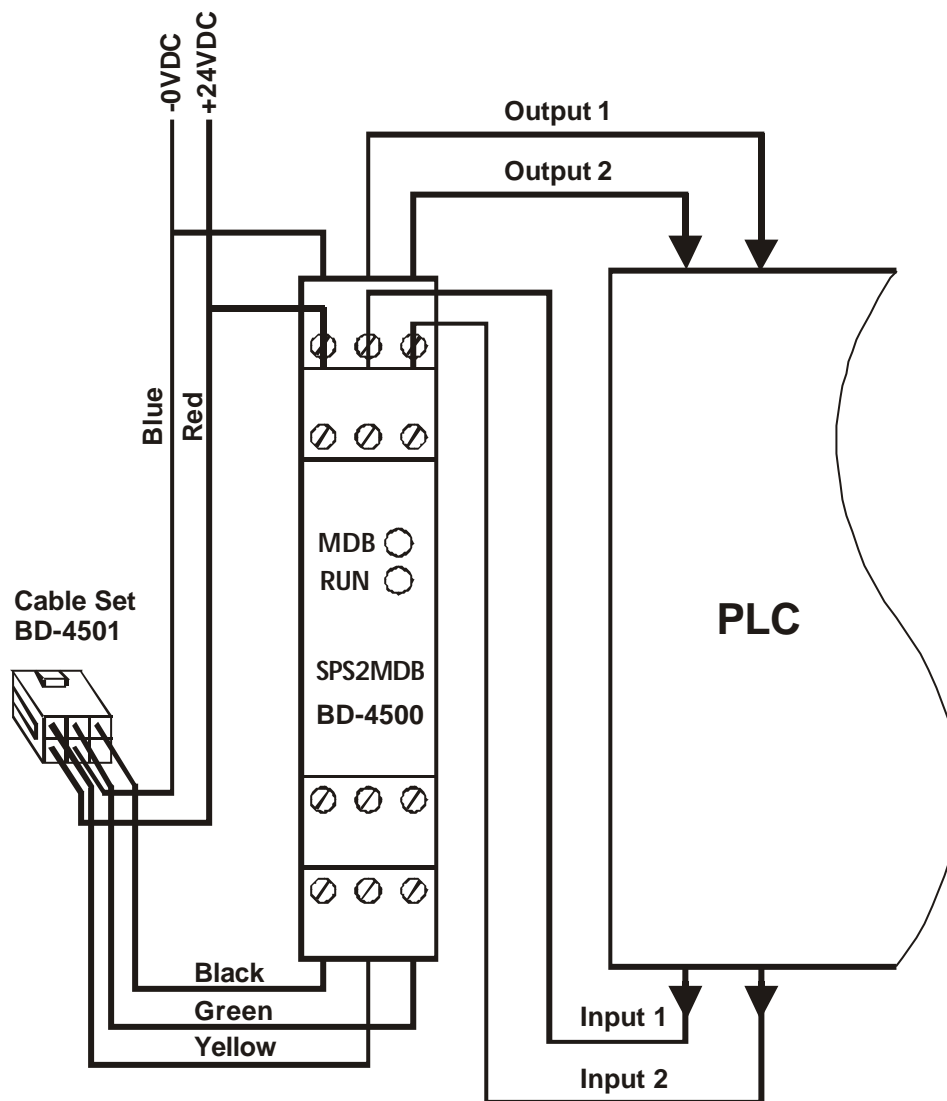


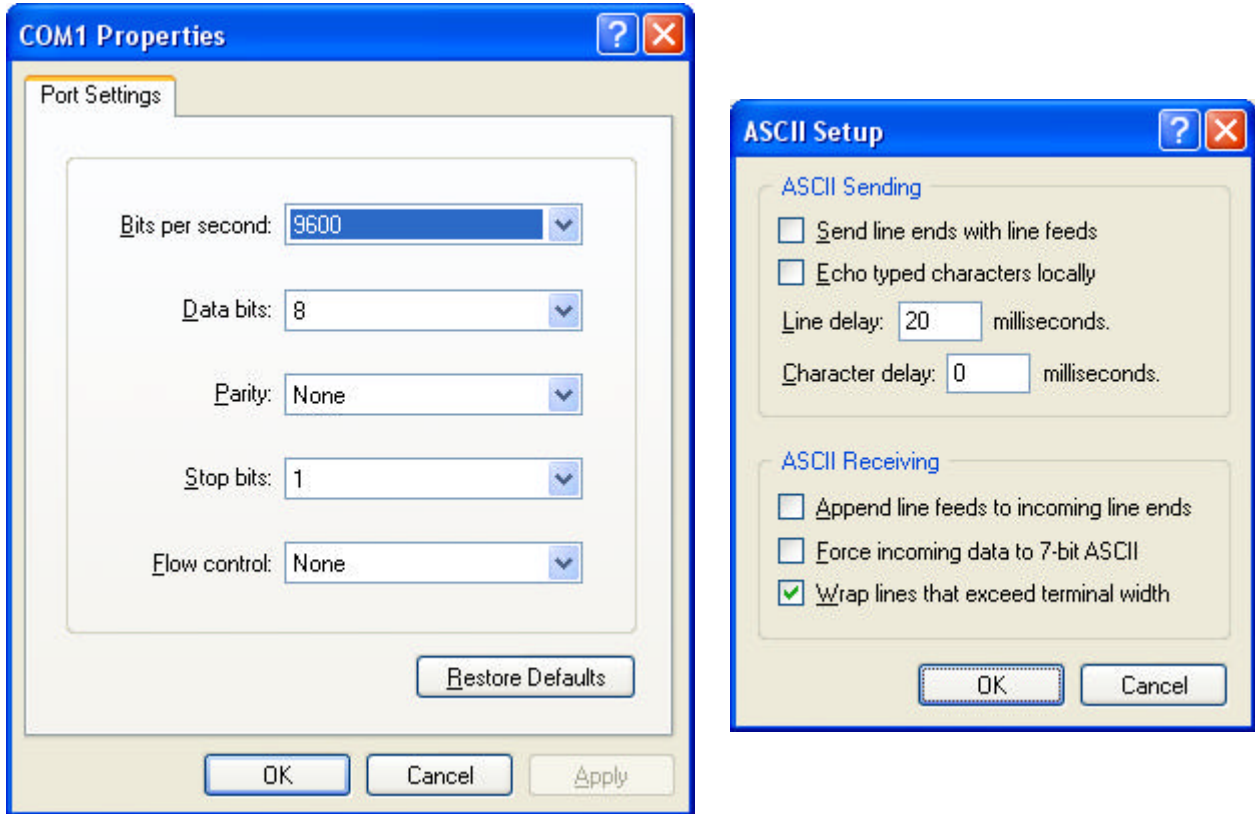
Figure 2

Chapter 6: Software Download

Connecting Hyper Terminal

The SPS2MDB includes a RS-232 interface to connect Microsoft HyperTerminal. The parameters for this Interface are: **9600 Baud, 8 Data bit, 1 Stop bit, No Parity, No handshake.**

Hyper Terminal setup:



RS-232 interface cable for Microsoft Hyper Terminal:

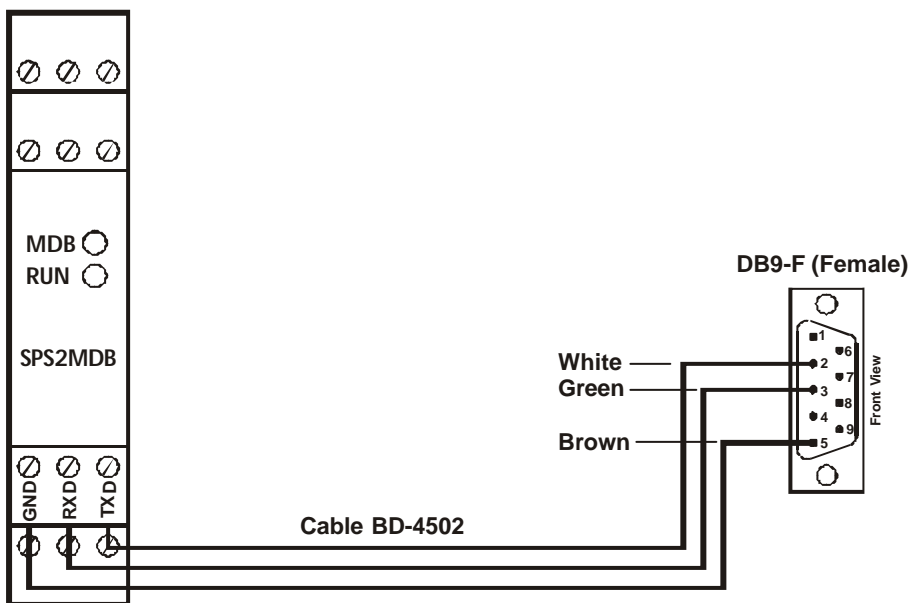


Figure 3

Monitor

The purpose of this monitor interface is:

- Gathering of information like version, settings etc.
- Monitoring errors.
- Debugging of the SPS2MDB application.
- Downloading new releases of the SPS2MDB application.

The monitor presents a screen as below:

```
MDB - Hyper Terminal
File Edit View Call Transfer Help
Press any key to enter Bootloader.
Downloader Version: 01.00, Build: 15.08.06
Copyright (C)2005-2006 by BonusData AG
-----
? = This help menu.
C = Application check sum.
D = Dump flash memory. [Address].[Length]
E = Dump EEPROM memory. [Address].[Length]
G = Start application.
M = Modify EEPROM contents. [New value]
Z = Erase entire flash memory.
: = Download application.
-
Connected 0:00:17  ANSI  38400 8-N-1  SCROLL  CAPS  NUM  Capture  Print echo
```

To enter the monitor, press any key in the first 3 seconds after the SPS2MDB interface is powered. After the 3 seconds has expired it will start the SPS2MDB application. To see a list of monitor commands enter a ?.

Enter monitor:

1. Connect a serial port of your PC to the SPS2MDB interface.
2. Start Microsoft Hyper Terminal. Make sure for a correct setup of the parameters.
3. Make sure Hyper Terminal is connected. (The clock in the left bottom corner is running.)
4. Power on the SPS2MDB interface.
5. Press any button within 3 seconds.

The Monitor should show up as in the print screen above.

Download of a new software release

To download a new software release the monitor must be active.

Select menu option 'Transfer' -> 'Send Text File...', select the file for download and Open. This will automatically start the download. The contents of the file must be a Intel Hexadecimal Object file with extension *.HEX.

Chapter 7: Setup of a Test Application

To setup a test application, two SPS2MDB interfaces can be used. One will run the SPS2MDB application to communicate with the MDB devices on one side and the SPS/PLC on the other side. The other will run the SPS simulator application and communicate through the 4 digital I/O lines. The resulting software and communication model is shown below.

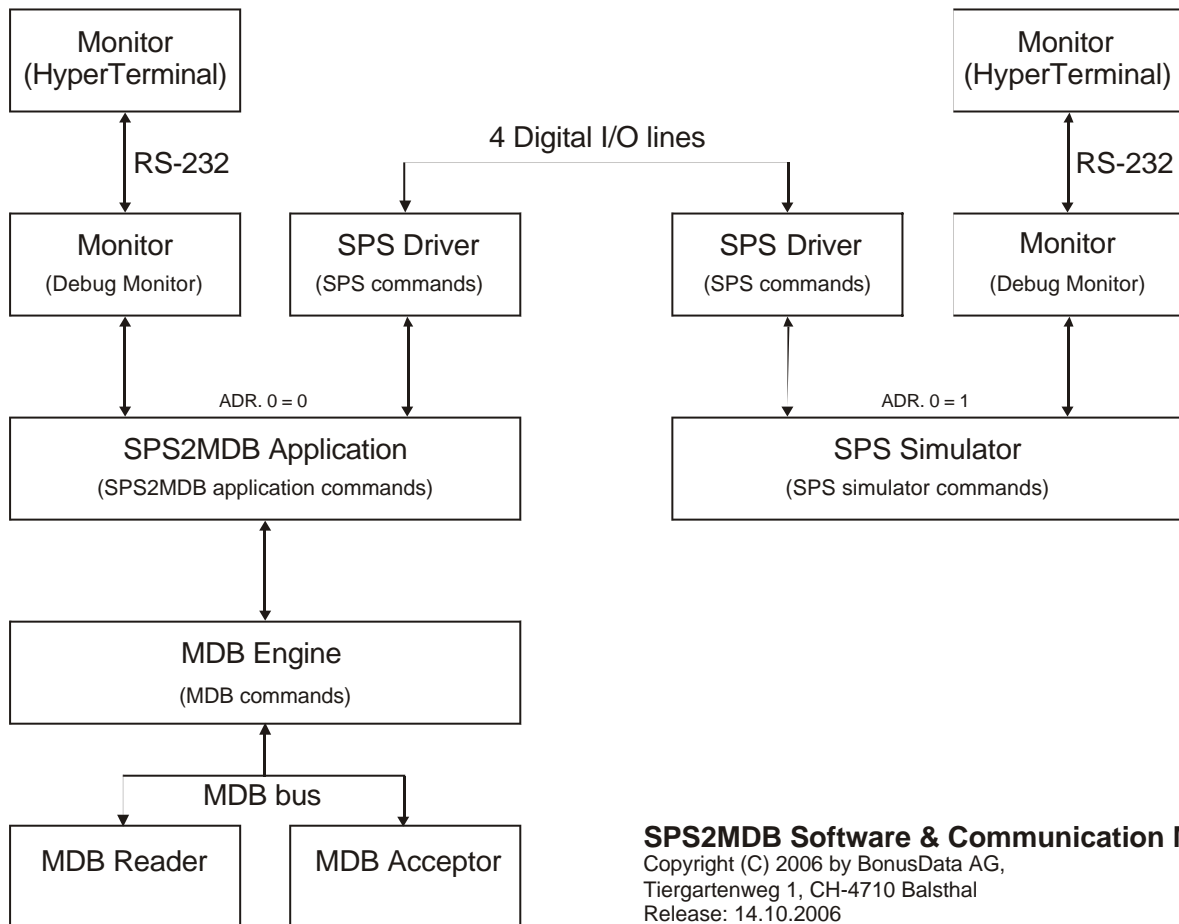


Figure: 4

The test application allows it to simulate with real equipment like an MDB change giver or a MDB card reader the complete application. It helps to get the knowledge for implementing the required software on a real SPS/PLC.

The following test sequences can be used to test the communication model above and also the final implementation with a real SPS/PLC.

Test sequences for SPS/PLC communication commands

Status

- Status request.

Enable / Disable

- Device enable, coins / cards are accepted.
- Device disable, coins / cards are rejected.

Enter session

- Device enable.

- Insert a coin / card -> enter session.
- Cancel session -> payout of coin / remove card.
- Device disable.

Enter vend / Vend denied

- Device enable.
- Insert a coin / card, enter session.
- Issue vend request.
- Receive vend denied.
- Cancel session -> payout of coin / remove card.
- Device disable.

Enter vend / Vend failed

- Device enable.
- Insert a coin / card, enter session.
- Issue vend request.
- Receive vend approved.
- Vend failed.
- Cancel session -> payout of coin / remove card.
- Device disable.

Enter vend / Vend success

- Device enable.
- Insert a coin / card, enter session.
- Issue vend request.
- Receive vend approved.
- Vend success.
- Cancel session -> payout of change / remove card.
- Device disable.

Enter vend / Vend approved (No further coins accepted)

- Device enable.
 - Insert a coin / card, enter session.
 - Device disable.
 - Issue vend request.
 - Receive vend approved.
 - Vend success.
 - Cancel session -> payout of change / remove card.
 - Device enable.
- (This sequence may not be suitable for card reader.)





Escrow (Acceptor only)



















- Device enable.
- Insert a coin -> enter session.
- Escrow -> payout of coin.
- Device disable.

Remove card (Reader only)

- Device enable.
- Insert a card -> enter session.
- Remove card -> exit session.
- Device disable.

Allowed commands within certain states

Command / State	Reset	Idle	Enabled	Session	Vend
A0 – Status	Error				

A1 – Enable	Error				Error
A2 – Disable	Error	Error			Error
A4 – Vend Request	Error	Error	Error		Error
A7 – Vend Success	Error	Error	Error	Error	
A8 – Vend Failure	Error	Error	Error	Error	
AA – Cancel Session	Error	Error	Error		Error
AB – Reset	Error				Error
Escrow	Error	Error	Error		Error
Remove card					

Chapter 8: Appendixes

A: Figures

Diagram #1: Diagram of a One Byte Transfer

Figure 1: Electrical Connection.

Figure 2: SPS2MDB to PLC Interconnection.

Figure 3: RS-232 interface cable for Microsoft Hyper Terminal.

Figure 4: SPS2MDB Software & Communication Model.

B: Document History

- 20.02.06 1.00 First implementation of SPS communication.
- 28.08.06 1.01 Final modifications of SPS communication.
- 14.10.06 1.02 - Implementation of coin acceptor.
- Implementation of debit card reader.
- 02.04.08 1.10 Added serial commands via RS-232 monitor interface.

C: Glossary

- LSB Last Significant Byte.
- MDB Multi Drop Protocol.
- MSB Most Significant Byte.
- OSI Open Systems Interconnection model.
- PLC Programmable Logical Controller.
- SPS Speicher Programmierbare Steuerung.
- VMC Vending Machine Controller.