LumbergI/ONetwork

CANopen

Technical Manual

LioN-Link BusHead CANopen-Slave

0940 CSL 601

lumbergautomation

A BELDEN BRAND

LioN-Link
BusHead CANopen-Slave

0940 CSL 601

Author: Carsten Cyfka
Datum: 29.08.06
Version: 00V90

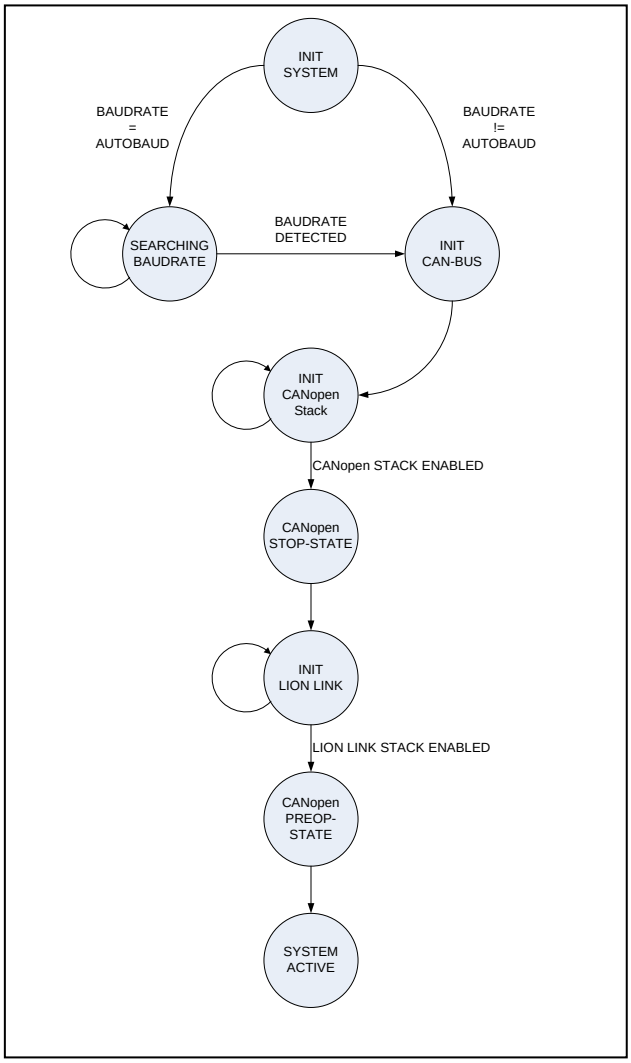
History:

Name	Änderung	Datum / Rev.

Contents

1.0 System Boot-Up

The following state diagram describes the CANopen LioN-Link BusHead boot up behaviour.



Picture 1: Boot- up State diagram

2.0 Process data

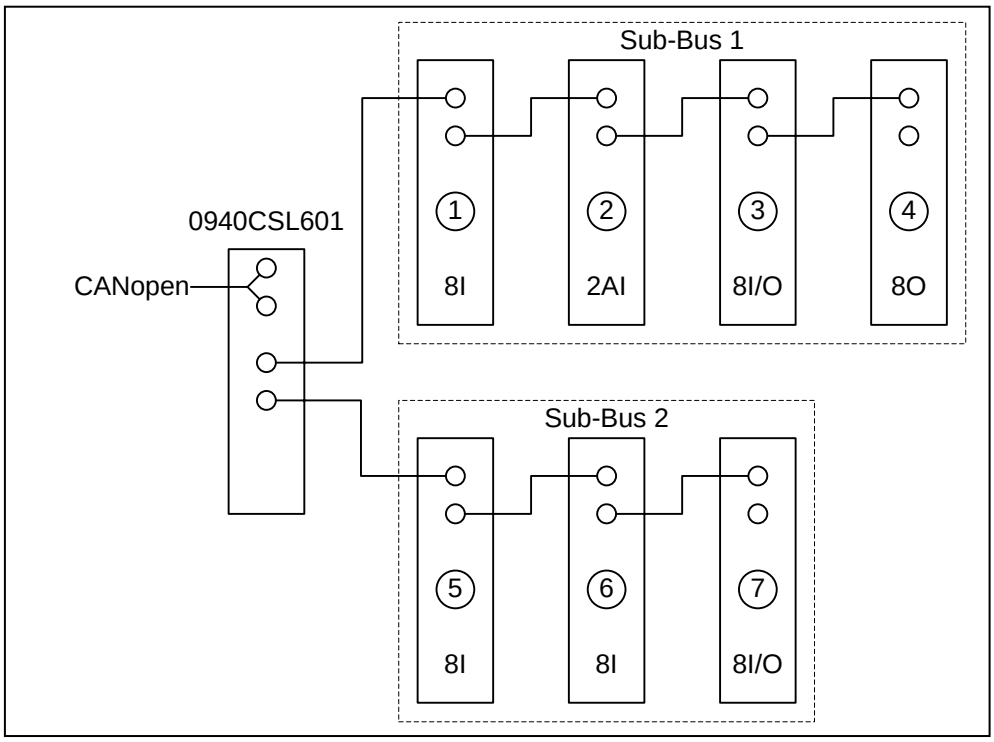
The 0940 CSL 601 module has no process data. The whole process data come from the LioN- Link Slaves which are connected to the sub bus lines 1 and 2. The connection of 20 slave modules is allowed. The slave module data is accessible via the CANopen object dictionary. The object dictionary entries are set up in the state “INIT LION LINK”, after a successfully initialisation.

The following entries are supported and dynamically allocated by the BusHead:

OD Index	Type	Size
0x6000	Digital Inputs	8Bit
0x6200	Digital Outputs	8Bit
0x6404	Analogue Inputs	8Bit
0x6414	Analogue Outputs	8Bit
0x4000	Module Parameter	8Bit

The sub index entries are corresponding with the physical placement of the LioN Link IO-Slave module.

Example:
Picture 2 shows the actual Lion-Link-Sub-Bus configuration.



Picture 2: Lion Link example configuration

The slave modules deliver the following process information:

- Module 1: 8 digital inputs
- Module 2: 2 analogue inputs 16Bit/each
- Module 3: 8 digital inputs / 8 digital outputs
- Module 4: 8 digital outputs
- Module 5: 8 digital inputs
- Module 6: 8 digital inputs
- Module 7: 8 digital inputs / 8 digital outputs

After the initialisation is done, the layout of the object dictionary is like shown in picture 3.

DIGITAL INPUTS 0x6000		
INDEX	SUBINDEX	DESCRIPTION
0x6000	0	COUNT OF DIGITAL INPUT BYTES
0x6000	1	MODULE 1 - 8 INPUTS
0x6000	2	MODULE 3 - 8 INPUTS
0x6000	3	MODULE 5 - 8 INPUTS
0x6000	4	MODULE 6 - 8 INPUTS
0x6000	5	MODULE 7 - 8 INPUTS
DIGITAL OUTPUTS 0x6200		
INDEX	SUBINDEX	DESCRIPTION
0x6200	0	COUNT OF DIGITAL OUTPUT BYTES
0x6200	1	MODULE 3 - 8 OUTPUTS
0x6200	2	MODULE 4 - 8 OUTPUTS
0x6200	3	MODULE 7 - 8 OUTPUTS
0x6200	4	MODULE 7- 8 OUTPUTS
ANALOGUE INPUTS 0x6404		
INDEX	SUBINDEX	DESCRIPTION
0x6404	0	COUNT OF ANLOGUE BYTES
0x6404	1	MODULE 2 ANALOGUE IN 1 LOW BYTE
0x6404	2	MODULE 2 ANALOGUE IN 1 HIGH BYTE
0x6404	3	MODULE 2 ANALOGUE IN 2 LOW
0x6404	4	MODULE 2 ANALOGUE IN 2 HIGH

Picture 3: Object dictionary example layout

Process data exchange via PDOs is only possible in the CANopen operational state.

3.0 Process Data Objects (PDOs)

The CANopen module 0940CSL601 supports 4 transmit PDOs and 4 receive PDOs. Each PDO can transmit 8 bytes of process data information.

3.1 Receive PDOs (RPDOs)

Receive PDO data contents the output information. The following object dictionary entries can be mapped to a RPDO:

- Object 0x6200 / Digital outputs
- Object 0x6414 / Analogue outputs

3.1.1 Receive PDO Communication Settings

The communication settings for the RPDOs 1..4 can be configured via the object dictionary entries 0x1400 (RPDO1) .. 0x1403 (RPDO4). The default transmission type is asynchronous event triggered mode.

RPDO 1			
INDEX	SUBINDEX	DESCRIPTION	VALUE
0x1400	0	COUNT OF ENTRIES	0x02
0x1400	1	COB-ID	0x00000200+Node-Id
0x1400	2	TRANSMISSION TYPE	0xFF
RPDO 2			
INDEX	SUBINDEX	DESCRIPTION	VALUE
0x1401	0	COUNT OF ENTRIES	0x02
0x1401	1	COB-ID	0x00000300+Node-Id
0x1401	2	TRANSMISSION TYPE	0xFF
RPDO 3			
INDEX	SUBINDEX	DESCRIPTION	VALUE
0x1402	0	COUNT OF ENTRIES	0x02
0x1402	1	COB-ID	0x00000400+Node-Id
0x1402	2	TRANSMISSION TYPE	0xFF
RPDO 4			
INDEX	SUBINDEX	DESCRIPTION	VALUE
0x1403	0	COUNT OF ENTRIES	0x02
0x1403	1	COB-ID	0x00000500+Node-Id
0x1403	2	TRANSMISSION TYPE	0xFF

Picture 4: Receive PDO Communication Settings

3.2 Transmit PDOs (TPDOs)

Transmit PDO data contents the input information. The following object dictionary entries can be mapped to a TPDO:

- Object 0x6000 / Digital inputs
- Object 0x6404 / Analogue inputs

3.2.1 Transmit PDO Communication Settings

The communication settings for the TPDOs 1..4 can be configured via the object dictionary entries 0x1800 (TPDO1) .. 0x1803 (TPDO4). The default transmission type is asynchronous event triggered mode.

TPDO 1			
INDEX	SUBINDEX	DESCRIPTION	VALUE
0x1800	0	COUNT OF ENTRIES	0x02
0x1800	1	COB-ID	0x00000180+Node-Id
0x1800	2	TRANSMISSION TYPE	0xFF
TPDO 2			
INDEX	SUBINDEX	DESCRIPTION	VALUE
0x1801	0	COUNT OF ENTRIES	0x02
0x1801	1	COB-ID	0x00000280+Node-Id
0x1801	2	TRANSMISSION TYPE	0xFF
TPDO 3			
INDEX	SUBINDEX	DESCRIPTION	VALUE
0x1802	0	COUNT OF ENTRIES	0x02
0x1802	1	COB-ID	0x00000380+Node-Id
0x1802	2	TRANSMISSION TYPE	0xFF
TPDO 4			
INDEX	SUBINDEX	DESCRIPTION	VALUE
0x1803	0	COUNT OF ENTRIES	0x02
0x1803	1	COB-ID	0x00000480+Node-Id
0x1803	2	TRANSMISSION TYPE	0xFF

Picture 5: Transmit PDO Communication Settings

3.3 Mapping of process data

There are two ways of mapping the process data into the available PDOs.

3.3.1 User Mapping

A CANopen master (e.g. PLC) can manual configure the mapping table by writing the mapping parameter objects (0x1600..0x1603 for RPDOs and 0x1A00..0x1A03 for the TPDOs). The mapping procedure is described in the CiA specification 301.

The user mapping can be activated by clearing the AUTOMAP bit in the Lion-Link configuration byte (Object 0x3000) or simply overwrite the auto map entries.

CAUTION: Be careful to map only process data that is available. Mapping of not present object dictionary entries will be ignored and answered with the corresponding abort code (refer to CiA Spec. 301).

3.3.2 Auto mapping

Auto mapping means, that the module writes the mapping table by itself after the detection of the connected slave modules.

Auto mapping is the default setting. Auto mapping can be activated by setting the AUTOMAP bit in the Lion-Link configuration byte.

Auto mapping works according to the following rules:

First the digital input information is mapped to the TPDOs, starting with TPDO1. After all digital input bytes are mapped, the analogue input bytes will be added to the remaining TPDOs. If more process data is available than the 4 TPDOs can handle, the system only configures the 4 TPDOs.

The digital output information is mapped in the same kind to the RPDOs.

For the example configuration of picture 2, the mapping is shown in picture 6.

DIGITAL INPUTS 0x6000			
INDEX	SUBINDEX	DESCRIPTION	MAPPING
0x6000	0	COUNT OF DIGITAL INPUT BYTES	
0x6000	1	MODULE 1 - 8 INPUTS	TPDO_1 BYTE 0
0x6000	2	MODULE 3 - 8 INPUTS	TPDO_1 BYTE 1
0x6000	3	MODULE 5 - 8 INPUTS	TPDO_1 BYTE 2
0x6000	4	MODULE 6 - 8 INPUTS	TPDO_1 BYTE 3
0x6000	5	MODULE 7 - 8 INPUTS	TPDO_1 BYTE 4
DIGITAL OUTPUTS 0x6200			
INDEX	SUBINDEX	DESCRIPTION	MAPPING
0x6200	0	COUNT OF DIGITAL OUTPUT BYTES	
0x6200	1	MODULE 3 - 8 OUTPUTS	RPDO_1 BYTE 0
0x6200	2	MODULE 4 - 8 OUTPUTS	RPDO_1 BYTE 1
0x6200	3	MODULE 7 - 8 OUTPUTS	RPDO_1 BYTE 2
0x6200	4	MODULE 7- 8 OUTPUTS	RPDO_1 BYTE 3
ANALOGUE INPUTS 0x6404			
INDEX	SUBINDEX	DESCRIPTION	MAPPING
0x6404	0	COUNT OF ANLOGUE BYTES	
0x6404	1	MODULE 2 ANALOGUE IN 1 LOW BYTE	TPDO_1 BYTE 5
0x6404	2	MODULE 2 ANALOGUE IN 1 HIGH BYTE	TPDO_1 BYTE 6
0x6404	3	MODULE 2 ANALOGUE IN 2 LOW	TPDO_1 BYTE 7
0x6404	4	MODULE 2 ANALOGUE IN 2 HIGH	TPDO_2 BYTE 0

Picture 6: Object dictionary example layout with mapping information.

4.0 LED Indicators

For state and diagnostic information there are different LED at each module. Zur Status und Diagnose sind auf Modulebene LED verfügbar

4.1 LED Module Status (MS)

Operating mode	LED Green	LED Red
IDLE	ON	OFF
ERROR	OFF	1Hz
CANopen PreOp – State	1Hz	OFF
CANopen Stop - State	2Hz	Off
CANopen Reset - State	1Hz Alternating	1Hz Alternating

4.2 LED Network Status (NS)

Operating mode	LED Green	LED Red
IDLE	ON	OFF
CAN Error Passive	1Hz Alternating	1Hz Alternating
CAN Bus Off	OFF	ON
AUTOBAUD	2Hz	OFF

4.3 LED I/O Status Sub-Bus (I/O S_N)

Operating mode	LED Green	LED Red
Sub-Bus- Line in data exchange mode	ON	OFF
Sub-Bus-Line in error mode	OFF	ON
Sub-Bus-Line in configuration or offline mode	OFF	OFF

4.4 LED Sub-Bus-Supply-Voltage (U_{SN})

Operating mode	LED Green
Sub-Bus-Line Voltage in range	ON
Sub-Bus-Line Voltage out of range	OFF

5.0 Error Codes (EMCY Messages)

If there is a diagnostic message the CANopen module will send an EMCY (Emergency) telegram.

This message can be read at the device specifically diagnostic address. For further information also take note of the CiA specification 301.

EMCY Code Table:

Error	EMCY Code	Error Reg.	Additional Information				
			BYTE 1	BYTE 2	BYTE 3	BYTE 4	BYTE 5
Internal Software Error	0x6100	0x01	0x00	0x00	0x00	0x00	0x00
CAN Message Overrun	0x8110	0x11	0x00	0x00	0x00	0x00	0x00
CAN Error Passive	0x8120	0x11	0x00	0x00	0x00	0x00	0x00
CAN Error Control	0x8130	0x11	0x00	0x00	0x00	0x00	0x00
PDO Length to small	0x8210	0x11	0x00	0x00	0x00	0x00	0x00
PDO Length to high	0x8230	0x11	0x00	0x00	0x00	0x00	0x00
Configuration Mismatch if configuration Lock is set	0x7000	0x01	0x00	0x00	0x00	0x00	0x00
Placeholder Configuration Error	0x7001	0x01	A	0x00	0x00	0x00	0x00
Sub Bus Module communication error	0x7002	0x01	B	C	D	E	0x00
Sub Bus Voltage low	0x7003	0x01	F	0x00	0x00	0x00	0x00
Sub Bus Module Supply Error	0x7004	0x01	G	H	I	0x00	0x00

A: Sub Bus Line with placeholder error (1/2)

To be completed

B: Module Status Sub Bus Line 1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X	Slave 15	Slave 14	Slave 13	Slave 12	Slave 11	Slave 10	Slave 9

C: Module Status Sub Bus Line 1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Slave 8	Slave 7	Slave 6	Slave 5	Slave 4	Slave 3	Slave 2	Slave 1

D: Module Status Sub Bus Line 2

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X	Slave 15	Slave 14	Slave 13	Slave 12	Slave 11	Slave 10	Slave 9

E: Module Status Sub Bus Line 2

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Slave 8	Slave 7	Slave 6	Slave 5	Slave 4	Slave 3	Slave 2	Slave 1

If the bit is set, the corresponding slave cause is responsible for the error message.

F: BusHead Sub Bus Voltage diagnostic

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X	X	X	X	TH2	TH1	OL2	OL1

OL1: Overload Supply Sub Bus line 1
OL2: Overload Supply Sub Bus line 2
TH1: System Threshold Sub Bus line 1
TH2: System Threshold Sub Bus line 2

G: Sub Bus Line ½
H: Module in Sub Bus Line (Including place holder offset)
I: Diagnostic Byte

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X	X	X	X	AOL	SOL	SUV	AUV

AUV: Actuator low voltage if set
SUV: Sensor low voltage if set
SOL: Sensor Overload (possible shorted)
AOL: Actuator Overload (possible shorted)

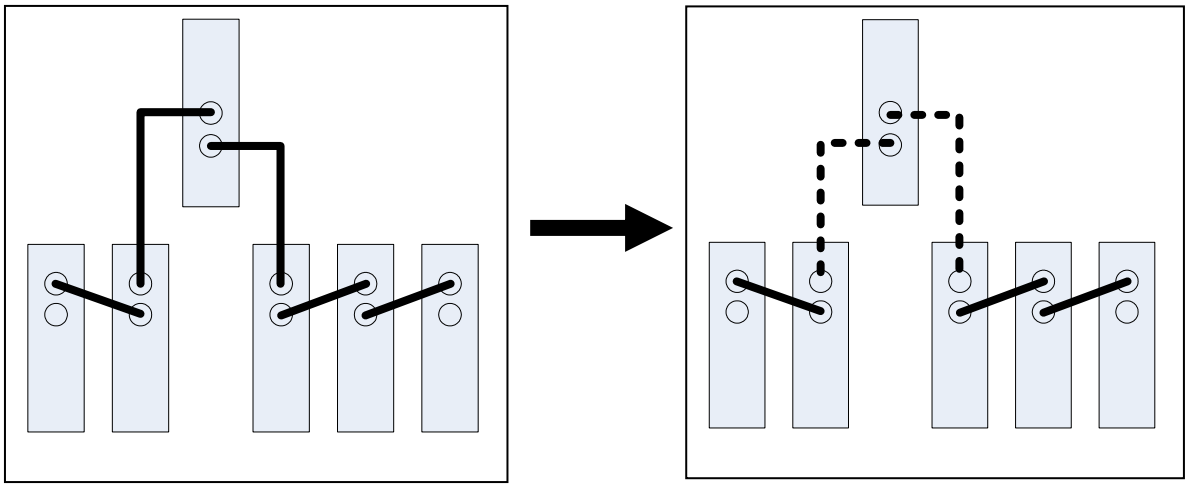
Lion-Link configuration diagnostic

The Lion-Link system has the ability to lock the active configuration. A locked configuration will be checked on each try to start the Lion-Link system. If a configuration failure occurs, the corresponding sub bus LED on the BusHead signals the failure state by a red light, furthermore the sub bus slave modules, which caused the faulty configuration, signals the failure by a static red "I/O" LED as well.
In the case of an error the BusHead try's to restart the system after 5 seconds, until the error source is removed.

The actual configuration is locked by setting bit 0 (SBCL) in the "Lion-Link-Configuration-Register" (refer to Chapter (TODO)). The setting of the SBCL bit is non volatile (stored in EEPROM) until the CANopen master clears the bit.

The following cases can be handled:

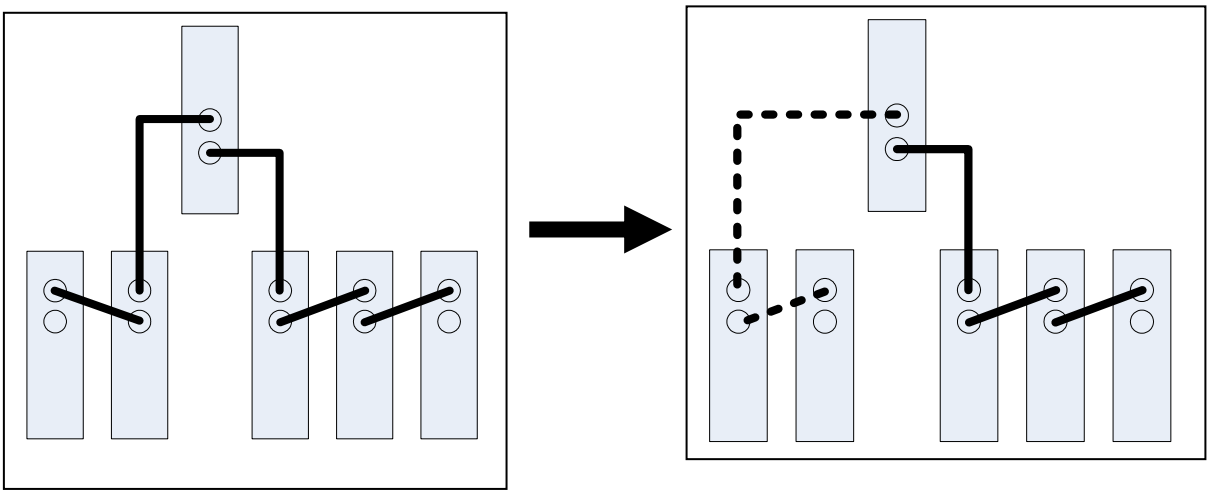
CASE A: Interchanged bus lines on the BusHead



	I/O SB_1	I/O SB_2	I/O S_1	I/O S_2	I/O S_3	I/O S_4	I/O S_5
GREEN							
RED	**X**	**X**	**X**	**X**	**X**	**X**	**X**

I/O SB_1 and I/O SB_2 : Sub-Bus Line LED on the BusHead
I/O $S_{1..5}$: I/O LED on the corresponding slave

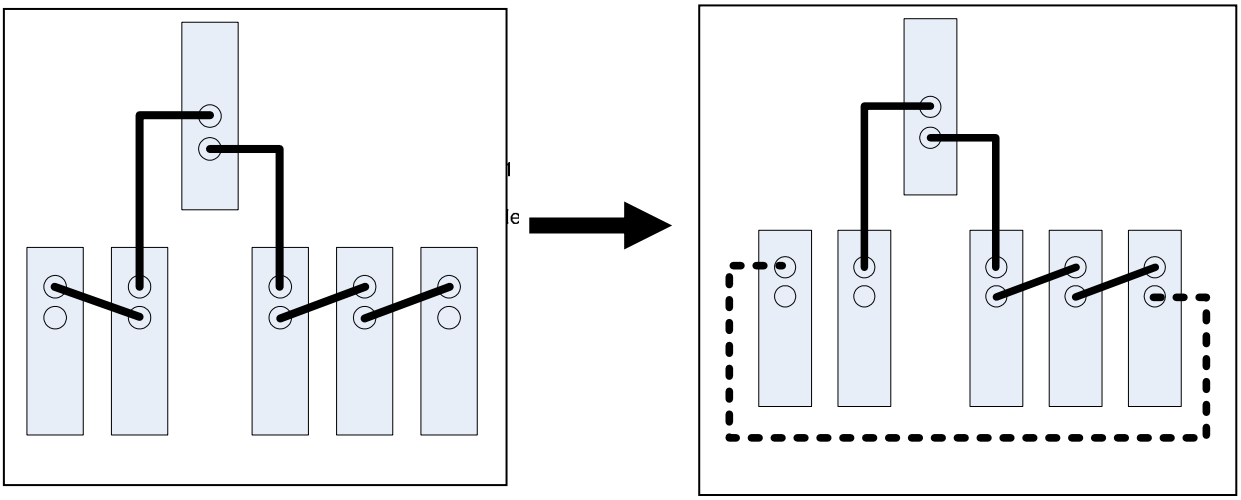
CASE B: Interchanged slaves at on bus line



	I/O SB_1	I/O SB_2	I/O S_1	I/O S_2	I/O S_3	I/O S_4	I/O S_5
GREEN		**X**	Master		**X**	**X**	**X**
RED	**X**		**X**	**X**			

I/O SB_1 and I/O SB_2 : Sub-Bus Line LED on the BusHead
I/O $S_{1..5}$: I/O LED on the corresponding slave

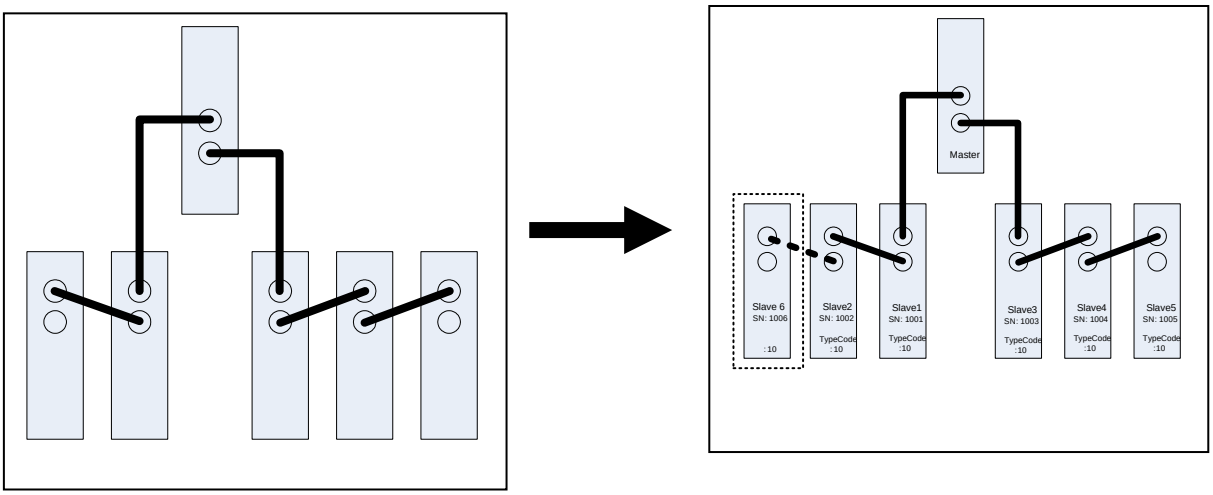
CASE C: Interchanged slaves between both bus line



	I/O SB_1	I/O SB_2	I/O S_1	I/O S_2	I/O S_3	I/O S_4	I/O S_5
GREEN	**X**		**X**		**X**	**X**	**X**
RED		**X**		**X**			

I/O SB_1 and I/O SB_2 : Sub-Bus Line LED on the BusHead
I/O $S_{1..5}$: I/O LED on the corresponding slave

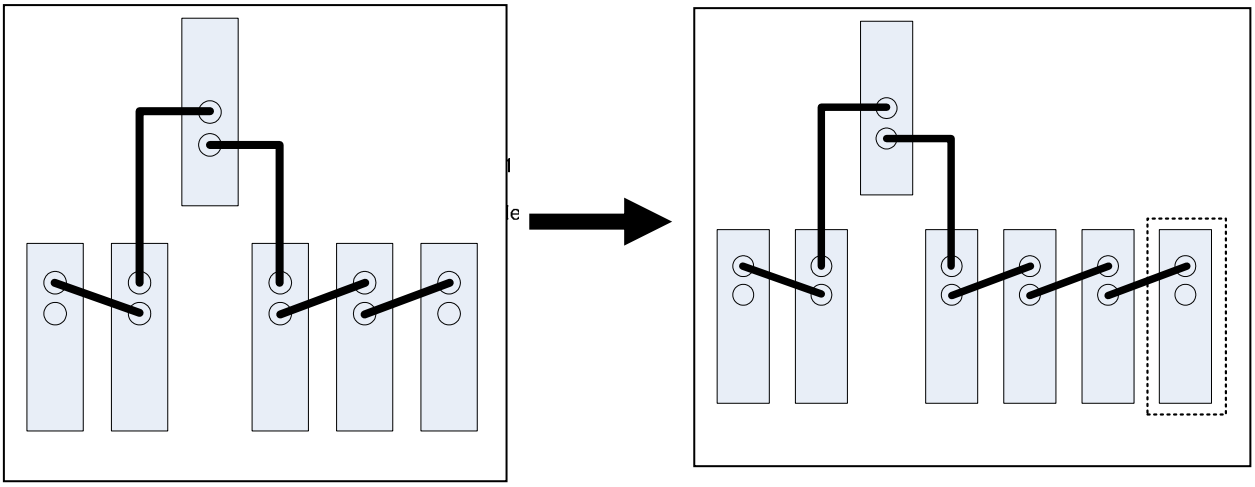
CASE D: Insertion of a new slave at the end of sub bus line 1



	I/O SB_1	I/O SB_2	I/O S_1	I/O S_2	I/O S_3	I/O S_4	I/O S_5	I/O S_6
GREEN		**X**	**X**	**X**	**X**	**X**	**X**	
RED	**X**			Master				**X**

I/O SB_1 and I/O SB_2 : Sub-Bus Line LED on the BusHead
I/O $S_{1..5}$: I/O LED on the corresponding slave

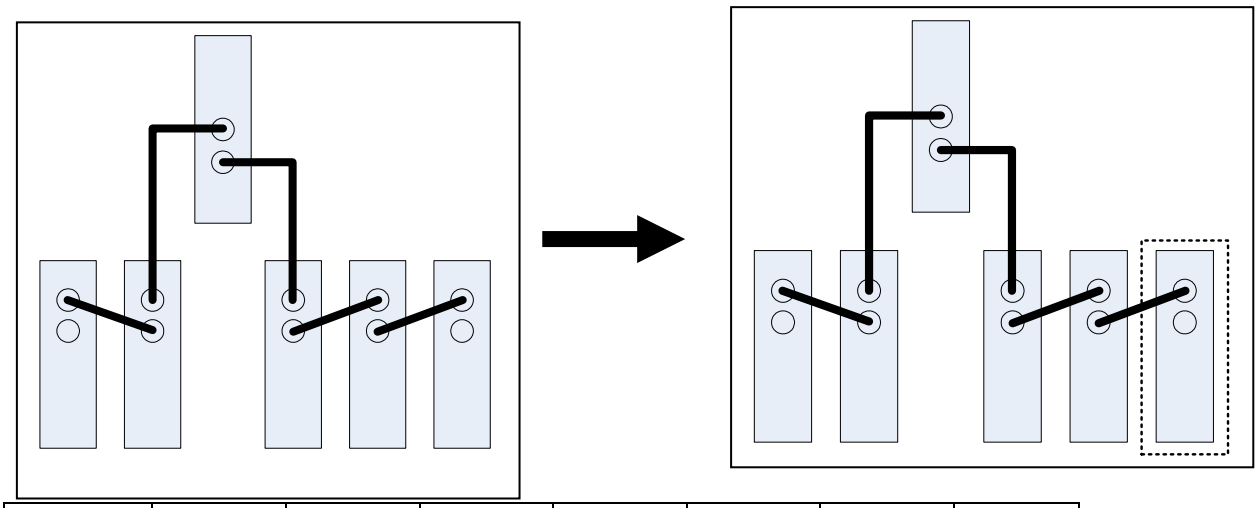
CASE E: Insertion of a new slave at the end of sub bus line 2



	I/O SB_1	I/O SB_2	I/O S_1	I/O S_2	I/O S_3	I/O S_4	I/O S_5	I/O S_6
GREEN	**X**		**X**	**X**	**X**	**X**	**X**	
RED		**X**						**X**

I/O SB_1 and I/O SB_2 : Sub-Bus Line LED on the BusHead
I/O $S_{1..5}$: I/O LED on the corresponding slave

CASE F: Exchange of a module with a different type code

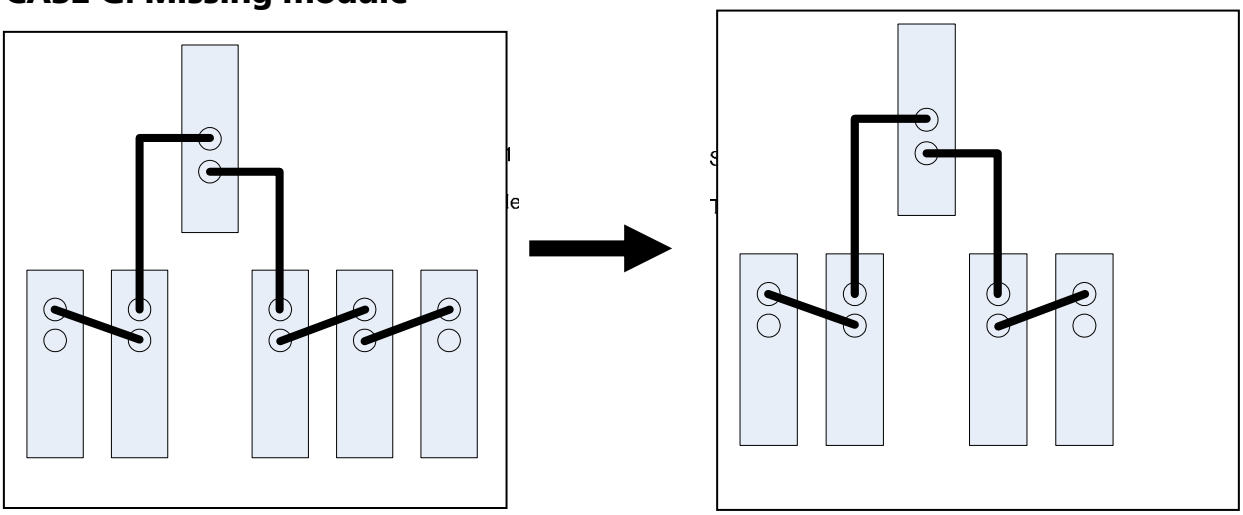


	I/O SB_1	I/O SB_2	I/O S_1	I/O S_2	I/O S_3	I/O S_4	I/O S_5
GREEN	**X**	**X**	**X**	**X**	**X**	**X**	**X**
RED							

I/O SB_1 and I/O SB_2 : Sub-Bus Line LED on the BusHead
I/O $S_{1..5}$: I/O LED on the corresponding slave

Remark: All LED are green but the sub bus will restart after 5s while it is not possible to reconfigure the slave module.

CASE G: Missing module



	I/O SB_1	I/O SB_2	I/O S_1	I/O S_2	I/O S_3	I/O S_4	I/O S_5
GREEN	**X**		**X**	**X**	**X**	**X**	**X**
RED		**X**					

I/O SB_1 and I/O SB_2 : Sub-Bus Line LED on the BusHead
I/O $S_{1..5}$: I/O LED on the corresponding slave