

## Flowcontrol

Mit Hilfe der Flowcontrolfunktion kann bei der CBOX300-310 ein Datensatz kontrolliert über Profibus-DP übertragen werden. (Vergleichbar bei der seriellen Schnittstelle mit einem Handshake; es muss zuerst ein Datensatz quittiert werden, bevor ein weiterer gesendet werden kann)

Die Flowcontrolfunktion unterstützt folgende Funktionen:

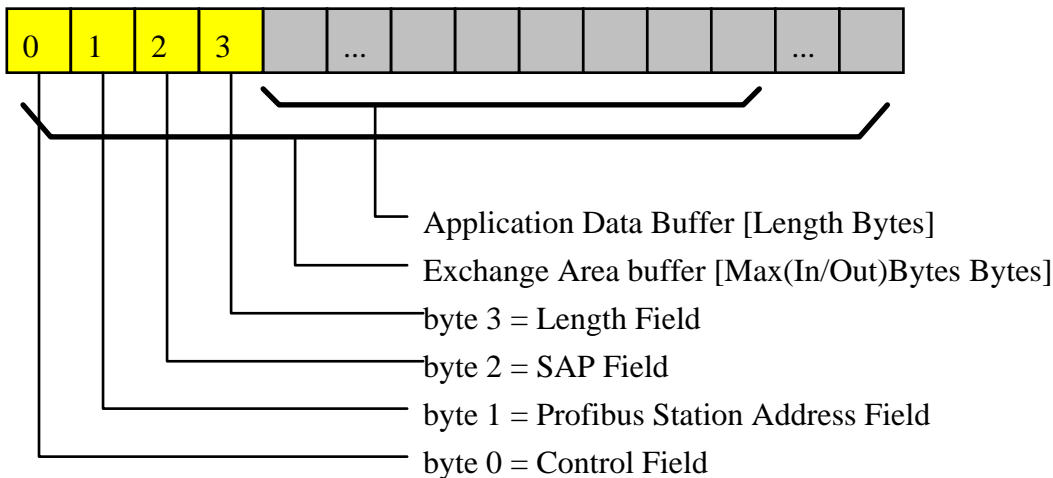
- Synchronisieren des Flowcontroltreibers
- Quittieren von Datensätzen im Eingangsbereich als auch im Ausgangsbereich.
- Fragmentierung von Datensätzen die länger als der gewählte Datenaustauschbereich sind.
- Löschen von internen Buffern und Queues

## Datenformat DPD-Treiber

Achtung! Es müssen generell alle Datensätze quittiert werden, um neue Daten Profibusseitig zu empfangen.

Ist dieser Treiber ausgewählt, so werden die Daten mit Hilfe des Flowcontrol-Mechanismus übertragen. Mittels des Treibers besteht die Möglichkeit eine komplette Kontrolle über den Zustand des Gerätes zu bekommen. Zusätzlich zu den eigentlichen Daten werden 4 Headerbyte vorangestellt. (Statusbyte, Profibusadresse, SAP-Feld, Längenbyte)

Die genaue Bedeutung und Funktionsweise entnehmen Sie bitte der folgenden Beschreibung.



Byte0 dient für Flow control, Fragmentierung, und Synchronisation

Byte1 enthält die Profibus Stations Adresse

Byte2 SAP (Service Access Point) Feld, steht für spätere Erweiterung des Datalogic Profibus Treibers zur Verfügung (nicht zu verwechseln mit Profibus SAP des internationalen Standards) Aktuell ist SAP 0 (normale Daten) und SAP 255 (FF)(rücksetzen der internen Queue) definiert

Byte3 Länge der tatsächlichen Nutzdaten



## Aufbau INPUT Byte 0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
------	------	------	------	------	------	------	------

- Bit 0 TX Buffer full, toggelt wenn neue Daten anliegen (handshake)
- Bit 1 RX Buffer leer, toggelt wenn rx block von Output gelesen ist (handshake)
- Bit 2 Resync Ack, 1 für ACK als resync request (Master kann mittels dieses Bits erkennen ob Slave Online ist)
- Bit 3 More Bit, ist auf 1 wenn mehrere Blöcke (Fragmente) aus einem Datensatz folgen 0, wenn letzter Block übertragen ist (wird erzeugt, wenn Daten+Header größer ist, als der gewählte Eingangsbereich).
- \* Bit 7 ist gesetzt auf 1 bei DAD –Treiber (Datalogic Anybus Driver)

## Aufbau OUTPUT Byte 0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
------	------	------	------	------	------	------	------

- Bit 0 TX Buffer leer, zu bedienen (toggle) wenn übertragener Block vom Master gelesen wurde (handshake) bzw. Quittierung
- Bit 1 RX Buffer voll, zu bedienen (toggle) wenn neuer Block vom Master zur Verfügung steht (handshake) und dieser gesendet wird (Ereignis ist im Eingangsbyte 0,bit1 zu sehen)
- Bit 2 Resync Request, auf 1 gesetzt für 1 sec. zur resynchronisation des Slaves. Alle 4 handshake bits sind danach 0
- Bit 3 More Bit, muß auf 1 gesetzt werden, wenn mehrere Blöcke (Fragmente) aus einem Datensatz folgen, muß auf 0 gesetzt werden, wenn letzter Block übertragen wird
- \* Bit 7 muss auf 1 gesetzt werden bei DAD -Treiber

**Achtung:** Toggle bedeutet, dass mit jedem Wechsel 0→ 1 und 1→ 0 neue Daten da sind, bzw. Daten gesendet werden können.

## Beispiel: Nur empfangen von Daten mit flow control:

Bsp: Scanner (z.B. DS2100-1000 default configuration) an CBOX300/310. Scanner wird mittels Hardwarekontakt getriggert  
(Achtung: Daten können an SPS S7 nur mit Standard Bausteinen SFC14/SFC15 dargestellt werden)

Spezifische Einstellungen (Projektierung SPS)

Projektiert: 32in / 8out bytes  
Profibusadresse CBOX300/310: 5

Achtung: Serielle Schnittstellenparameter müssen an CBOX als auch am Scanner gleich eingestellt sein!  
(Ist bei default beider Geräte auf jeden fall gegeben)

Beispiel:

Scanner liest 00112233 , als Header + Terminator sind STX und CR LF programmiert (am Scanner und CBOX)  
Das abschneiden der Terminatorzeichen ist disabled d.h. die Zeichen sind auch auf der Profibusseite vorhanden.

Bereiche in der SPS:

Output in (hex)		Input in (hex)	
Byte 0	00	Byte 0	01
Byte 1	00	Byte 1	05
Byte 2	00	Byte 2	00
Byte 3	00	Byte 3	0B
Byte 4	00	Byte 4	02
Byte 5	00	Byte 5	30
Byte 6	00	Byte 6	30
Byte 7	00	Byte 7	31
		Byte 8	31
		Byte 9	32
		Byte 10	32
		Byte 11	33
		Byte 12	33
		Byte 13	0D
		Byte 14	0A
		.....	00
		Byte31	

Byte 0,bit 0 signalisiert dass neue Daten da sind, durch spiegeln des bits in den Ausgang wird dieser Datensatz quittiert, das heißt, wenn nun neue Daten gelesen werden oder schon da sind, werden nun diese eingetragen. Neue Daten werden dann durch einen erneuten Wechsel des bit 0 signalisiert. Usw,.....

Quittieren der Daten

Output in (hex)

Input in (hex)

Byte 0	<b>01</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>00</b>
Byte 4	<b>00</b>
Byte 5	<b>00</b>
Byte 6	<b>00</b>
Byte 7	<b>00</b>

Byte 0	<b>01</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>0B</b>
Byte 4	<b>02</b>
Byte 5	<b>30</b>
Byte 6	<b>30</b>
Byte 7	<b>31</b>

Byte 8	<b>31</b>
Byte 9	<b>32</b>
Byte 10	<b>32</b>
Byte 11	<b>33</b>
Byte 12	<b>33</b>
Byte 13	<b>0D</b>
Byte 14	<b>0A</b>
.....	
Byte31	<b>00</b>

Scanner liest nun Barcode ABCDEFGH

Quittieren der Daten

Output in (hex)

Byte 0	<b>01</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>00</b>
Byte 4	<b>00</b>
Byte 5	<b>00</b>
Byte 6	<b>00</b>
Byte 7	<b>00</b>

Input in (hex)

Byte 0	<b>00</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>0B</b>
Byte 4	<b>02</b>
Byte 5	<b>41</b>
Byte 6	<b>42</b>
Byte 7	<b>43</b>
Byte 8	<b>44</b>
Byte 9	<b>45</b>
Byte 10	<b>46</b>
Byte 11	<b>47</b>
Byte 12	<b>48</b>
Byte 13	<b>0D</b>
Byte 14	<b>0A</b>
.....	
Byte31	<b>00</b>

Quittieren wiederum durch spiegeln von Eingangsbyte 0 bit 0 in Ausgangsbyte 0 bit 0.

....

Im weiteren verweisen wir auf das Handbuch zu CBOX 300/310.



## Beispiel Scanner Start Stop über Profibus mit flow control enabled:

Bsp: Scanner (z.B. DS2100-1000 operating mode serial on line, serial start STX serial stop ETX) an CBOX300/310. Scanner wird über Profibus getriggert  
(Achtung: Daten können an SPS S7 nur mit Standard Bausteinen SFC14/SFC15 dargestellt werden)

Spezifische Einstellungen (Projektierung SPS)

Projektiert: 32in /8out bytes  
Profibusadresse CBOX: 5  
Startzeichen: STX (02hex)  
Stopzeichen: ETX (03hex)

Achtung: Serielle Schnittstellenparameter müssen an CBOX als auch am Scanner gleich eingestellt sein!  
(Ist bei default beider Geräte auf jeden fall gegeben)

Beispiel:

Scanner liest 00112233 , als Header + Terminator sind STX und CR LF programmiert (am Scanner und CBOX)  
Das abschneiden der Terminatorzeichen ist disabled d.h. die Zeichen sind auch auf der Profibusseite vorhanden.

Bereiche in der SPS:

Scanner aktivieren.

Durch Toggle von outbyte0 bit1 wird die Anzahl Zeichen die in outbyte3 steht gesendet. Beginnend mit Outbyte 4. → Es wird 02 hex zum Scanner gesendet. (Momentan ist nur serial Start /Stop möglich)

Output in (hex)	Input in (hex)		
Byte 0 <table border="1"><tr><td>02</td></tr></table>	02	Byte 0 <table border="1"><tr><td>02</td></tr></table>	02
02			
02			
Byte 1 <table border="1"><tr><td>05</td></tr></table>	05	Byte 1 <table border="1"><tr><td>00</td></tr></table>	00
05			
00			
Byte 2 <table border="1"><tr><td>00</td></tr></table>	00	Byte 2 <table border="1"><tr><td>00</td></tr></table>	00
00			
00			
Byte 3 <table border="1"><tr><td>01</td></tr></table>	01	Byte 3 <table border="1"><tr><td>00</td></tr></table>	00
01			
00			
Byte 4 <table border="1"><tr><td>02</td></tr></table>	02	Byte 4 <table border="1"><tr><td>00</td></tr></table>	00
02			
00			
Byte 5 <table border="1"><tr><td>00</td></tr></table>	00	Byte 5 <table border="1"><tr><td>00</td></tr></table>	00
00			
00			
Byte 6 <table border="1"><tr><td>00</td></tr></table>	00	Byte 6 <table border="1"><tr><td>00</td></tr></table>	00
00			
00			
Byte 7 <table border="1"><tr><td>00</td></tr></table>	00	Byte 7 <table border="1"><tr><td>00</td></tr></table>	00
00			
00			
	Byte 8 <table border="1"><tr><td>00</td></tr></table>	00	
00			
	Byte 9 <table border="1"><tr><td>00</td></tr></table>	00	
00			
	Byte 10 <table border="1"><tr><td>00</td></tr></table>	00	
00			
	Byte 11 <table border="1"><tr><td>00</td></tr></table>	00	
00			
	Byte 12 <table border="1"><tr><td>00</td></tr></table>	00	
00			
	Byte 13 <table border="1"><tr><td>00</td></tr></table>	00	
00			
	Byte 14 <table border="1"><tr><td>00</td></tr></table>	00	
00			
	..... <table border="1"><tr><td>00</td></tr></table>	00	
00			
	Byte31 <table border="1"><tr><td>00</td></tr></table>	00	
00			

**Scanner hat gelesen:**

→ bit 0 wird in inbyte 0 gesetzt

Output in (hex)

Byte 0	<b>02</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>01</b>
Byte 4	<b>02</b>
Byte 5	<b>00</b>
Byte 6	<b>00</b>
Byte 7	<b>00</b>

Input in (hex)

Byte 0	<b>03</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>0B</b>
Byte 4	<b>02</b>
Byte 5	<b>30</b>
Byte 6	<b>30</b>
Byte 7	<b>31</b>
Byte 8	<b>31</b>
Byte 9	<b>32</b>
Byte 10	<b>32</b>
Byte 11	<b>33</b>
Byte 12	<b>33</b>
Byte 13	<b>0D</b>
Byte 14	<b>0A</b>
.....	
Byte31	<b>00</b>

Byte 0,bit 0 signalisiert dass neue Daten da sind, durch spiegeln des bits in den Ausgang wird dieser Datensatz quittiert, das heißt, wenn nun neue Daten gelesen werden oder schon da sind, werden nun diese eingetragen. Neue Daten werden dann durch einen erneuten Wechsel des bit 0 signalisiert. Usw,.....



Quittieren der Daten

Output in (hex)

Byte 0	<b>03</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>01</b>
Byte 4	<b>02</b>
Byte 5	<b>00</b>
Byte 6	<b>00</b>
Byte 7	<b>00</b>

Input in (hex)

Byte 0	<b>03</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>0B</b>
Byte 4	<b>02</b>
Byte 5	<b>30</b>
Byte 6	<b>30</b>
Byte 7	<b>31</b>

Byte 8	<b>31</b>
Byte 9	<b>32</b>
Byte 10	<b>32</b>
Byte 11	<b>33</b>
Byte 12	<b>33</b>
Byte 13	<b>0D</b>
Byte 14	<b>0A</b>
.....	
Byte31	<b>00</b>

**Lesetor beenden:**

Durch Toggle von outbyte0 bit1 wird die Anzahl Zeichen die in outbyte3 steht gesendet. Beginnend mit Outbyte 4. → Es wird 03 hex zum Scanner gesendet.

Output in (hex)

Byte 0	01
Byte 1	05
Byte 2	00
Byte 3	01
Byte 4	03
Byte 5	00
Byte 6	00
Byte 7	00

Input in (hex)

Byte 0	01
Byte 1	05
Byte 2	00
Byte 3	0B
Byte 4	02
Byte 5	30
Byte 6	30
Byte 7	31

Byte 8	31
Byte 9	32
Byte 10	32
Byte 11	33
Byte 12	33
Byte 13	0D
Byte 14	0A
.....	
Byte31	00

Nun ist eine komplette Lesung beendet. Es beginnt nun wieder bei Scanner Start .....

....

Im weiteren verweisen wir auf das Handbuch zu CBOX 300/310.

## Beispiel: Handscanner mit flow control, empfangen von Daten:

Bsp: Handscanner (z.B. DLL6010) an CBOX300/310.

**Achtung: Die meisten DL-Handscanner haben eine Versorgungsspannung von 5Vdc oder 4,75...14Vdc oder 4,75...20 Vdc .**

**Dies bedeutet, dass Sie ein entsprechendes Anschlusskabel zwischen CBOX und Handscanner verwenden müssen! (Siehe jeweilige Bezeichnung bzw. Schaltbild zu den Kabeln im Anhang)**

(Achtung: Daten können an SPS S7 nur mit Standard Bausteinen OB1→SFC14/SFC15→Variablen-tabelle dargestellt werden, dies erfordert eine Einstellung der Konsistenz über Gesamtstring)

- 2 Arten:
1. Universalmodul auswählen → Konsistenz einstellen
  2. GSD-Datei mit Konsistenz über Gesamtstring anfordern

Spezifische Einstellungen (Projektierung SPS)

Projektiert: 32in / 8out bytes  
 Profibusadresse CBOX300/310: 5

Achtung: Serielle Schnittstellenparameter müssen an CBOX als auch am Scanner gleich eingestellt sein!  
 An der CBOX300/CBOX310 muss der Parameter Scanner Interface → **Auto connect** → **disabled** eingestellt sein  
 An der Lesepistole muss als Schnittstelle RS232 programmiert sein (siehe Handbuch zu dem Handleser)

Beispiel:

Scanner liest 00112233 , als Header + Terminator sind STX und CR LF programmiert (am Scanner und CBOX)  
 Das abschneiden der Terminatorzeichen ist disabled d.h. die Zeichen sind auch auf der Profibusseite vorhanden.

Bereiche in der SPS:

Output in (hex)	Input in (hex)		
Byte 0 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 0 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">01</td></tr></table>	01
00			
01			
Byte 1 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 1 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">05</td></tr></table>	05
00			
05			
Byte 2 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 2 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00
00			
00			
Byte 3 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 3 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">0B</td></tr></table>	0B
00			
0B			
Byte 4 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 4 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">02</td></tr></table>	02
00			
02			
Byte 5 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 5 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">30</td></tr></table>	30
00			
30			
Byte 6 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 6 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">30</td></tr></table>	30
00			
30			
Byte 7 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 7 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">31</td></tr></table>	31
00			
31			
	Byte 8 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">31</td></tr></table>	31	
31			
	Byte 9 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">32</td></tr></table>	32	
32			
	Byte 10 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">32</td></tr></table>	32	
32			
	Byte 11 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">33</td></tr></table>	33	
33			
	Byte 12 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">33</td></tr></table>	33	
33			
	Byte 13 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">0D</td></tr></table>	0D	
0D			
	Byte 14 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">0A</td></tr></table>	0A	
0A			
	.....		
	Byte31 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	
00			

Byte 0, bit 0 signalisiert dass neue Daten da sind, durch spiegeln des bits in den Ausgang wird dieser Datensatz quittiert, das heißt, wenn nun neue Daten gelesen werden oder schon da sind, werden nun diese eingetragen. Neue Daten werden dann durch einen erneuten Wechsel des bit 0 signalisiert. Usw,.....

Quittieren der Daten

Output in (hex)

Input in (hex)

Byte 0	<b>01</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>00</b>
Byte 4	<b>00</b>
Byte 5	<b>00</b>
Byte 6	<b>00</b>
Byte 7	<b>00</b>

Byte 0	<b>01</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>0B</b>
Byte 4	<b>02</b>
Byte 5	<b>30</b>
Byte 6	<b>30</b>
Byte 7	<b>31</b>

Byte 8	<b>31</b>
Byte 9	<b>32</b>
Byte 10	<b>32</b>
Byte 11	<b>33</b>
Byte 12	<b>33</b>
Byte 13	<b>0D</b>
Byte 14	<b>0A</b>
.....	
Byte31	<b>00</b>

Scanner liest nun Barcode ABCDEFGH

Quittieren der Daten

Output in (hex)

Input in (hex)

Byte 0	<b>01</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>00</b>
Byte 4	<b>00</b>
Byte 5	<b>00</b>
Byte 6	<b>00</b>
Byte 7	<b>00</b>

Byte 0	<b>00</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>0B</b>
Byte 4	<b>02</b>
Byte 5	<b>41</b>
Byte 6	<b>42</b>
Byte 7	<b>43</b>
Byte 8	<b>44</b>
Byte 9	<b>45</b>
Byte 10	<b>46</b>
Byte 11	<b>47</b>
Byte 12	<b>48</b>
Byte 13	<b>0D</b>
Byte 14	<b>0A</b>
.....	
Byte31	<b>00</b>

Quittieren wiederum durch spiegeln von Eingangsbyte 0 bit 0 in Ausgangsbyte 0 bit 0.

....

Im weiteren verweisen wir auf das Handbuch zu CBOX 300/310.

## Beispiel: DS6300, Matrix mit flow control, empfangen von Daten:

Bsp: Handscanner, oder stationärer Scanner der nicht mittels Winhost parametriert wird (z.B. DS6300) an CBOX300/310.

(Achtung: Daten können an SPS S7 nur mit Standard Bausteinen SFC14/SFC15 dargestellt werden)

Spezifische Einstellungen (Projektierung SPS)

Projektiert: 32in / 8out bytes  
 Profibusadresse CBOX300/310: 5

Achtung: Serielle Schnittstellenparameter müssen an CBOX als auch am Scanner gleich eingestellt sein!

**An der CBOX300/CBOX310 muss der Parameter Scanner Interface → Auto connect → disabled eingestellt sein**

Beispiel:

Scanner liest 00112233 , als Header + Terminator sind STX und CR LF programmiert (am Scanner und CBOX)  
 Das abschneiden der Terminatorzeichen ist disabled d.h. die Zeichen sind auch auf der Profibusseite vorhanden.  
 (Sollte die Triggerung über Profibus erfolgen, siehe Beispiel Start, Stop über Profibus)

Bereiche in der SPS:

Output in (hex)	Input in (hex)		
Byte 0 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 0 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">01</td></tr></table>	01
00			
01			
Byte 1 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 1 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">05</td></tr></table>	05
00			
05			
Byte 2 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 2 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00
00			
00			
Byte 3 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 3 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">0B</td></tr></table>	0B
00			
0B			
Byte 4 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 4 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">02</td></tr></table>	02
00			
02			
Byte 5 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 5 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">30</td></tr></table>	30
00			
30			
Byte 6 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 6 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">30</td></tr></table>	30
00			
30			
Byte 7 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	Byte 7 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">31</td></tr></table>	31
00			
31			
	Byte 8 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">31</td></tr></table>	31	
31			
	Byte 9 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">32</td></tr></table>	32	
32			
	Byte 10 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">32</td></tr></table>	32	
32			
	Byte 11 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">33</td></tr></table>	33	
33			
	Byte 12 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">33</td></tr></table>	33	
33			
	Byte 13 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">0D</td></tr></table>	0D	
0D			
	Byte 14 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">0A</td></tr></table>	0A	
0A			
	.....		
	Byte31 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="text-align: center;">00</td></tr></table>	00	
00			

Byte 0, bit 0 signalisiert dass neue Daten da sind, durch spiegeln des bits in den Ausgang wird dieser Datensatz quittiert, das heißt, wenn nun neue Daten gelesen werden oder schon da sind, werden nun diese eingetragen. Neue Daten werden dann durch einen erneuten Wechsel des bit 0 signalisiert. Usw,.....

Quittieren der Daten

Output in (hex)

Input in (hex)

Byte 0	<b>01</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>00</b>
Byte 4	<b>00</b>
Byte 5	<b>00</b>
Byte 6	<b>00</b>
Byte 7	<b>00</b>

Byte 0	<b>01</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>0B</b>
Byte 4	<b>02</b>
Byte 5	<b>30</b>
Byte 6	<b>30</b>
Byte 7	<b>31</b>

Byte 8	<b>31</b>
Byte 9	<b>32</b>
Byte 10	<b>32</b>
Byte 11	<b>33</b>
Byte 12	<b>33</b>
Byte 13	<b>0D</b>
Byte 14	<b>0A</b>
.....	
Byte31	<b>00</b>

Scanner liest nun Barcode ABCDEFGH

Quittieren der Daten

Output in (hex)

Input in (hex)

Byte 0	<b>01</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>00</b>
Byte 4	<b>00</b>
Byte 5	<b>00</b>
Byte 6	<b>00</b>
Byte 7	<b>00</b>

Byte 0	<b>00</b>
Byte 1	<b>05</b>
Byte 2	<b>00</b>
Byte 3	<b>0B</b>
Byte 4	<b>02</b>
Byte 5	<b>41</b>
Byte 6	<b>42</b>
Byte 7	<b>43</b>
Byte 8	<b>44</b>
Byte 9	<b>45</b>
Byte 10	<b>46</b>
Byte 11	<b>47</b>
Byte 12	<b>48</b>
Byte 13	<b>0D</b>
Byte 14	<b>0A</b>
.....	
Byte31	<b>00</b>

Quittieren wiederum durch spiegeln von Eingangsbyte 0 bit 0 in Ausgangsbyte 0 bit 0.

....

Im weiteren verweisen wir auf das Handbuch zu CBOX 300/310.

Beispiel:

DS6300, Matrix mit flow control, empfangen von Daten, Morebit gesetzt, d.h. Datenaustauschbereich kleiner als darzustellender Barcode:

Bsp: Handscanner, oder stationärer Scanner der nicht mittels Winhost parametriert wird (z.B. DS6300 oder Matrix) an CBOX300/310.

(Achtung: Daten können an SPS S7 nur mit Standard Bausteinen SFC14/SFC15 dargestellt werden)

Spezifische Einstellungen (Projektierung SPS)

**Projektiert:** 8 in / 8 out bytes  
 Profibusadresse CBOX300/310: 5

Achtung: Serielle Schnittstellenparameter müssen an CBOX als auch am Scanner gleich eingestellt sein!

**An der CBOX300/CBOX310 muss der Parameter Scanner Interface → Auto connect → disabled eingestellt sein**

Beispiel:

Scanner liest 00112233 , als Header + Terminator sind STX und CR LF programmiert (am Scanner und CBOX)  
 Das abschneiden der Terminatorzeichen ist disabled d.h. die Zeichen sind auch auf der Profibusseite vorhanden.  
 (Sollte die Triggerrung über Profibus erfolgen, siehe Beispiel Start, Stop über Profibus)

Bereiche in der SPS:

Output in (hex)		Input in (hex)	
Byte 0	00	Byte 0	09
Byte 1	00	Byte 1	05
Byte 2	00	Byte 2	00
Byte 3	00	Byte 3	04
Byte 4	00	Byte 4	02
Byte 5	00	Byte 5	30
Byte 6	00	Byte 6	30
Byte 7	00	Byte 7	31

Byte 0, bit 0 signalisiert dass neue Daten da sind, durch spiegeln des bits in den Ausgang wird dieser Datensatz quittiert, das heißt, wenn nun neue Daten gelesen werden oder schon da sind, werden nun diese eingetragen. Neue Daten werden dann durch einen erneuten Wechsel des bit 0 signalisiert. Usw,.....

Quittieren der Daten

Output in (hex)		Input in (hex)	
Byte 0	01	Byte 0	08
Byte 1	05	Byte 1	05
Byte 2	00	Byte 2	00
Byte 3	00	Byte 3	04
Byte 4	00	Byte 4	31
Byte 5	00	Byte 5	32
Byte 6	00	Byte 6	32
Byte 7	00	Byte 7	33

Quittieren, freigeben für nächsten Block

Output in (hex)		Input in (hex)	
Byte 0	00	Byte 0	09
Byte 1	05	Byte 1	05
Byte 2	00	Byte 2	00
Byte 3	00	Byte 3	04
Byte 4	00	Byte 4	33
Byte 5	00	Byte 5	34
Byte 6	00	Byte 6	34
Byte 7	00	Byte 7	0D

Quittieren, freigeben für nächsten Block

Output in (hex)		Input in (hex)	
Byte 0	01	Byte 0	00
Byte 1	05	Byte 1	05
Byte 2	00	Byte 2	00
Byte 3	00	Byte 3	01
Byte 4	00	Byte 4	0A
Byte 5	00	Byte 5	32
Byte 6	00	Byte 6	32
Byte 7	00	Byte 7	33

Quittieren, des letzten Blocks, im Eingang sieht man, dass das Morebit nicht mehr gesetzt ist, d.h. dies ist der letzte Block

Output in (hex)

Input in (hex)

Byte 0	<b>00</b>	Byte 0	<b>00</b>
Byte 1	<b>05</b>	Byte 1	<b>05</b>
Byte 2	<b>00</b>	Byte 2	<b>00</b>
Byte 3	<b>00</b>	Byte 3	<b>01</b>
Byte 4	<b>00</b>	Byte 4	<b>0A</b>
Byte 5	<b>00</b>	Byte 5	<b>32</b>
Byte 6	<b>00</b>	Byte 6	<b>32</b>
Byte 7	<b>00</b>	Byte 7	<b>33</b>

Nach dem Quittieren ändert sich der Eingang nicht mehr, d.h. es stehen keine neuen Daten mehr an. Nun beginnt mit der nächsten Lesung der ganze Zyklus von neuem.  
Bei dem Quittieren genügt bit 0 zu spiegeln.



## Software-Beschreibung

flow\_cbox300.DOC  
16.02.2001/ SD 942

### Übersicht Flowcontrolfunktion CBOX300-310 DPD-Treiber

Seite 19 von 19

## Übersicht Anschlusskabel HHR-Scanner an C-BOX (Profibus)

Grundsätzlich ist der Anschluss der HHR-Scanner (Typen DLL/DLC, Gryphon, Dragon, Lynx) mit externer Spannungsversorgung an die C-BOX über das Anschlusskabel Typ CABG-145 möglich (externes Netzteil für Scanner wird benötigt).

Je nach Scannermodell und Versorgungsspannung der C-BOX ist auch alternativ eine direkte Spannungsversorgung einzelner Scanner-Typen möglich (s. Tabelle unten).

*Vorsicht:* Bei direkter Versorgung des Scanners über die C-BOX muss unbedingt auf die Spannungsversorgung der C-BOX geachtet werden um eine Beschädigung des Scanners durch Überspannung zu vermeiden!

HHR-Scanner-Typ	Spannungsversorgung Scanner (Bereich)	Anschlusskabel	Spannungsversorgung	Hinweis Spannungsversorgung C-BOX
DLL6010M / DragonD	4-20VDC	CABG-146	direkt über C-Box	<b>10-20VDC (!)</b>
DLL6010M / DragonD	4-20VDC	CABG-145	extern über Netzteil Typ PG5	10-30VDC
DLL/DLC/GryphonD (5V-Geräte)	5VDC	CABG-137	direkt über C-Box	<b>24 VDC (!)</b>
DLL/DLC/GryphonD (5V-Geräte)	5VDC	CABG-145	extern über Netzteil Typ PG5	10-30VDC
OM6010-R*, OM-Dragon*, OM-Gryphon*	10-28VDC	CABG-145	extern über Netzteil Typ PG220	10-30VDC
Lynx	10-30VDC	CABG-146	direkt über C-Box	10-30VDC

**(\*) Spannungsversorgung ausschließlich extern möglich.**

Anschlusskabel	Art.-Nr.	Ausführung	Kabellänge (ca.)
CABG-145	K61000482	glattes Kabel	2m
CABG-146	K61000483	glattes Kabel	2m
CABG-137	K61000428	Spiralkabel	2m

**Technische Änderungen vorbehalten.**