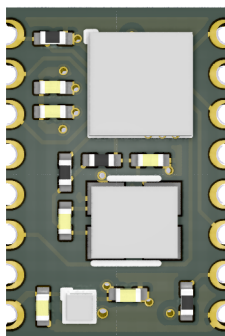


CDCTL-BX Data Manual



Contents

1	Overview	3
2	CDBUS Protocol	3
3	CDBUS IP Core	4
3.1	Block Diagram	4
4	CDCTL-B1	5
4.1	External Reference Circuit	5
4.2	Pin Definition	5
4.3	Mechanical Specifications	6
4.4	Absolute Maximum Ratings	6
4.5	Recommended Operating Conditions	6
4.6	DC Electrical Characteristics	6
4.7	Timing Specifications	7
5	Register Reference	7
5.1	SETTING:	7
5.2	FILTER:	7
5.3	DIV_xx_x:	8
5.4	INT_FLAG:	8
5.5	RX_CTRL:	8
5.6	TX_CTRL:	8
5.7	RX_PAGE_FLAG:	8

6	Peripheral Interface	9
6.1	SPI	9
6.2	I ² C	9
7	Operate Demonstration	10
7.1	Init	10
7.2	TX	10
7.3	RX	11
8	Copyright Statement	12
9	Contact Information	12
10	Change History	12

1 Overview

- CDCTL-BX is a CDBUS IP core based controller that provides I²C and SPI peripheral interfaces.
- The CDBUS IP Core is an open source implementation of the CDBUS Protocol.
- The CDBUS Protocol is a protocol for Asynchronous Serial Communication.

2 CDBUS Protocol

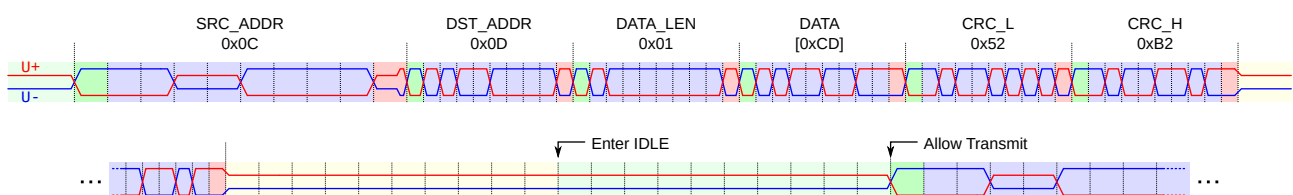
CDBUS is a protocol for Asynchronous Serial Communication, it has a 3-byte header: [src_addr, dst_addr, data_len], then user data, and finally 2 bytes of checksum.

It's suitable for one-to-one communication, e.g. UART or RS232. In this case, the address for each side are usually carefully selected and fixed, e.g: [0x55, 0xaa, data_len, ...], and the backward is: [0xaa, 0x55, data_len, ...].

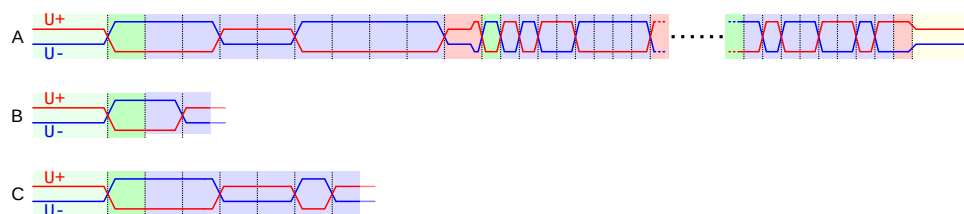
The CDBUS protocol is more valuable for bus communication, e.g. RS485 or Single Line UART. In this case:

- It introduces an arbitration mechanism that automatically avoids conflicts like the CAN bus.
- Support dual baud rate, provide high speed communication, maximum rate ≥ 10 Mbps.
- Support broadcast. (set dst_addr to 255)
- Max payload data size is 253 byte.
- Hardware packing, unpacking, verification and filtering, save your time and CPU usage.
- Backward compatible with traditional RS485 hardware. (still retains arbitration function)

The protocol example timing, include only one byte user data:
(How long to enter idle and how long to allow sending can be set.)



Arbitration example:

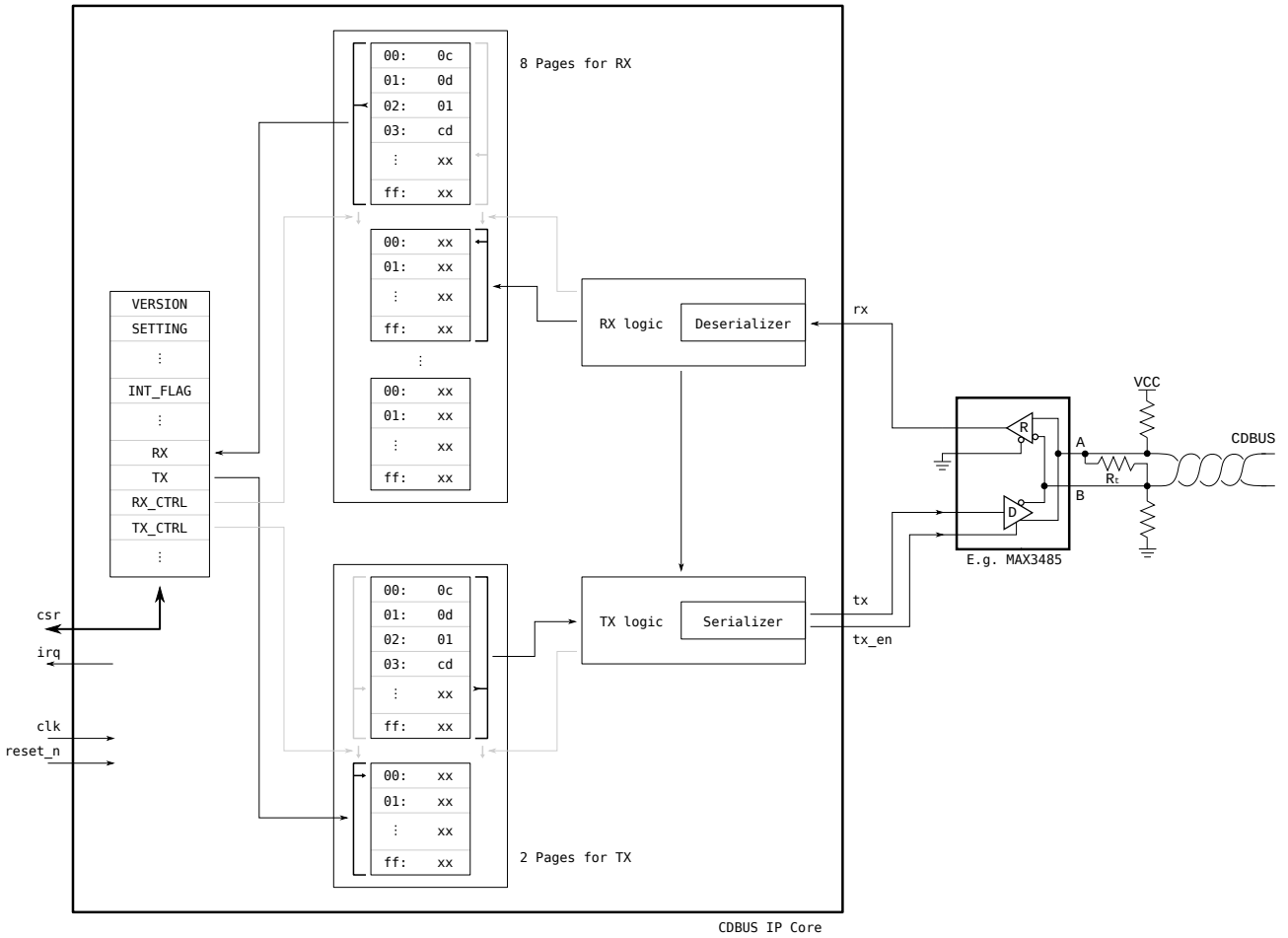


The idea of CDBUS was first designed and implemented by DUKELEC in 2009.

3 CDBUS IP Core

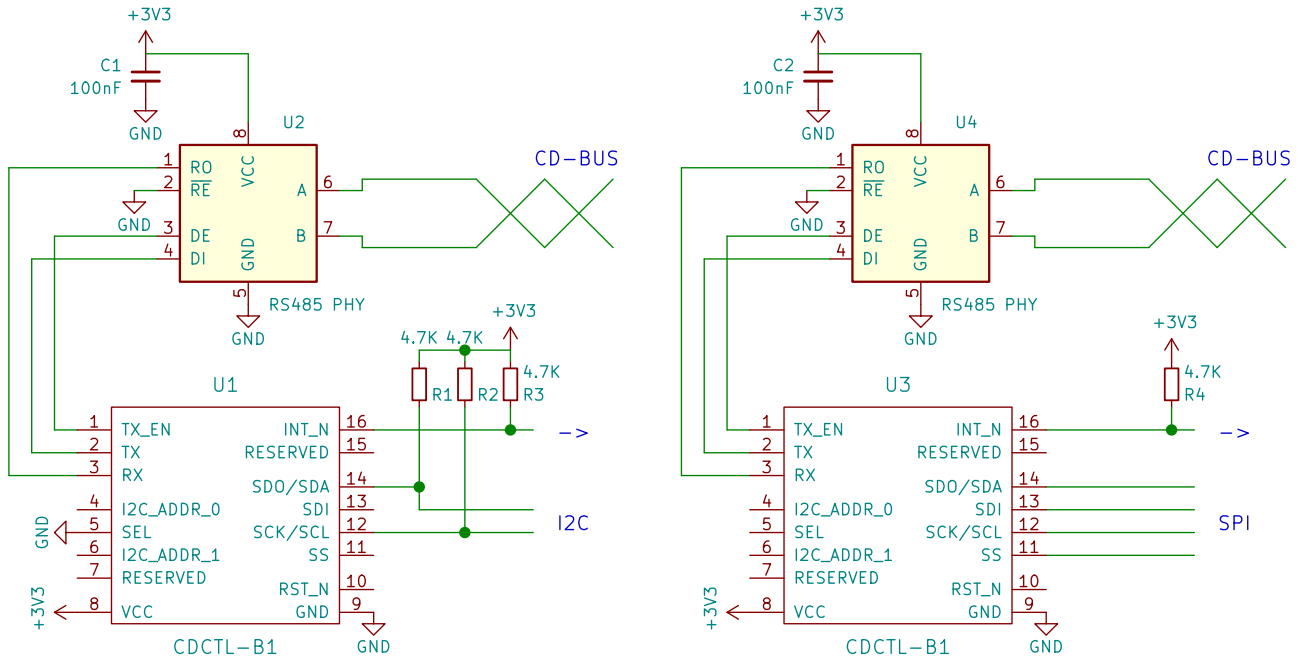
Source code and more details: https://github.com/dukelec/cdbus_ip

3.1 Block Diagram



4 CDCTL-B1

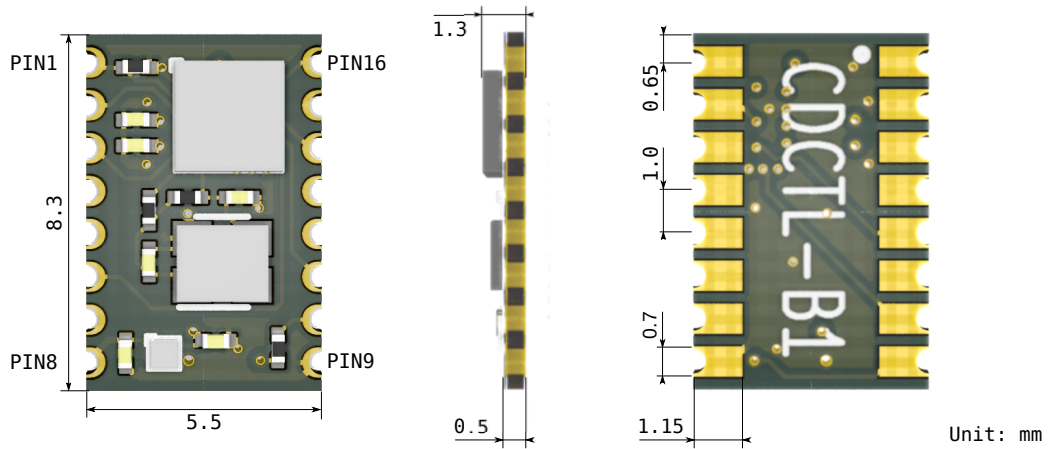
4.1 External Reference Circuit



4.2 Pin Definition

No	Name	I/O	Internal Pull	Description
1	TX_EN	O	Down (10 kOhm)	Transmit enable pin to RS485 PHY
2	TX	O	-	Transmit pin to RS485 PHY
3	RX	I	-	Receive pin from RS485 PHY
4, 6	I2C_ADDR	I	Up (40 kOhm)	Set I ² C address
5	SEL	I	Up (40 kOhm)	High select SPI mode, low select I ² C mode
7, 15	RESERVED			Left empty
8	VCC			Supply voltage
9	GND			Ground
10	RST_N	I	Up (10 kOhm)	Reset pin, ≥ 200 ns low pulse width required to reset (optional, but reset is required when the power supply ramp rate is not guaranteed)
11	SS	I	Up (40 kOhm)	SPI chip select
12	SCK/SCL	I	-	SPI/I ² C clock
13	SDI	I	-	SPI MOSI
14	SDO/SDA	I/O	-	SPI MOSI / I ² C SDA
16	INT_N	O	-	Interrupt pin, open-drain output

4.3 Mechanical Specifications



4.4 Absolute Maximum Ratings

Parameter	Min.	Max.
Supply Voltage VCC	-0.5 V	3.60 V
Storage Temperature (Ambient)	-55 °C	150 °C
Junction Temperature (T _j)	-	125 °C

4.5 Recommended Operating Conditions

Parameter	Min.	Max.
Supply Voltage VCC	3.14 V	3.46 V
Junction Temperature Operation	-40 °C	85 °C
Power supply ramp rate	0.6 V/ms	10 V/ms

4.6 DC Electrical Characteristics

Parameter	Min.	Typ.	Max.
V _{IL}	-0.3 V	-	0.8 V
V _{IH}	2.0 V	-	VCC + 0.2 V
V _{OL}	0.2 V	-	0.4 V
V _{OH}	VCC - 0.4 V	-	VCC - 0.2 V
I _{OL}	-	-	8 mA
I _{OH}	-	-	-8 mA
Input or I/O Leakage	-	-	+/-10 uA
I/O Capacitance (25°C, 1.0 MHz)	-	6 pF	-
Power Consumption	-	-	15 mW
V _{PORUP} (Power-On-Reset threshold)	0.7 V	-	1.6 V
V _{PORDN}	-	-	1.6 V

4.7 Timing Specifications

Symbol	Parameter	Freq.
F _{SYS}	System clock frequency	40 MHz

Symbol	Parameter	Min.	Max.
F _{SCK}	SPI clock frequency	-	20 MHz
F _{SCL}	I ² C clock frequency	-	3 MHz
-	Baud rate	610 bps	10 Mbps

5 Register Reference

Register Name	Addr[7:0]	Access	Default	Description
VERSION	0x00	RD	0x05	Hardware version
SETTING	0x01	RD/WR	0x10	Configs
IDLE_WAIT_LEN	0x02	RD/WR	0x0a (10 bit)	How long to enter idle
TX_WAIT_LEN	0x03	RD/WR	0x14 (20 bit)	How long to allow sending
FILTER	0x04	RD/WR	0xff	Receive filter
DIV_LS_L	0x05	RD/WR	0x5a	Low-speed rate setting
DIV_LS_H	0x06	RD/WR	0x01	
DIV_HS_L	0x07	RD/WR	0x5a	High-speed rate setting
DIV_HS_H	0x08	RD/WR	0x01	
INT_FLAG	0x09	RD	n/a	Status
INT_MASK	0x0a	RD/WR	0x00	Interrupt mask
RX	0x0b	RD	n/a	Read RX page
TX	0x0c	WR	n/a	Write TX page
RX_CTRL	0x0d	WR	n/a	RX control
TX_CTRL	0x0e	WR	n/a	TX control
RX_ADDR	0x0f	RD/WR	0x00	RX page read pointer
RX_PAGE_FLAG	0x10	RD	n/a	RX page flag

5.1 SETTING:

FIELD	DESCRIPTION
[0]	Enable push-pull output for tx and tx_en pin
[1]	Invert tx output
[2]	Disable hardware CRC
[3]	Save broken frame
[5:4]	tx_en delay before tx output in traditional mode
[6]	Disable arbitration for traditional mode

5.2 FILTER:

Match from top to bottom:

SRC_ADDR	DST_ADDR	FILTER	Receive or drop	Remarks
----------	----------	--------	-----------------	---------

not care	not care	255	Receive	Promiscuous mode
= FILTER	not care	!= 255	Drop	Avoid loopback
!= FILTER	255	not care	Receive	Broadcast
!= FILTER	!= 255	= DST_ADDR	Receive	
not care	!= 255	!= DST_ADDR	Drop	

5.3 DIV_xx_x:

Baud rate divider value:

$\text{DIV_xx}[15:0] = \text{sys_freq} \div \text{baud_rate} - 1$

5.4 INT_FLAG:

FIELD	DESCRIPTION
[0]	1: Bus in IDLE mode
[1]	1: RX page ready for read
[2]	1: RX lost: no empty page for RX
[3]	1: RX error: frame broken
[4]	1: TX page released by hardware
[5]	1: TX collision detected
[6]	1: TX error: conflict continued for 16 times

5.5 RX_CTRL:

FIELD	DESCRIPTION
[0]	Reset RX page read pointer
[1]	Switch RX page
[2]	Clear RX lost flag
[3]	Clear RX error flag
[4]	Reset RX block

5.6 TX_CTRL:

FIELD	DESCRIPTION
[0]	Reset TX page write pointer
[1]	Switch TX page
[2]	Clear TX collision flag
[3]	Clear TX error flag
[4]	Abort TX

5.7 RX_PAGE_FLAG:

Value zero indicate the frame in current RX page is correct;

Non-zero indicate the pointer of last received byte of the disturbed frame, include CRC.

6 Peripheral Interface

Burst read and write are useful for accessing REG_RX and REG_TX.

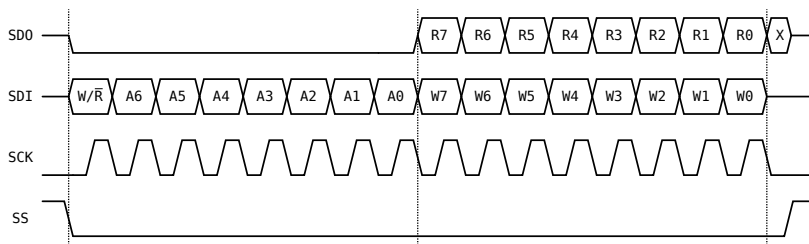
6.1 SPI

Read or write depend by bit W/\bar{R} :

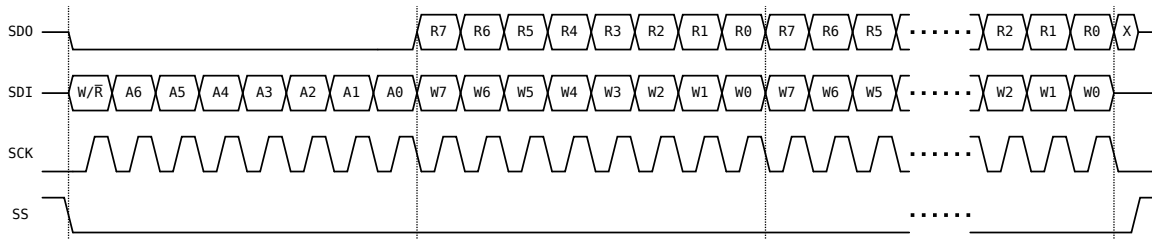
- 0: Read
- 1: Write

FIELD	DESCRIPTION
Ax	Register address
Wx	Write data, don't care in read mode
Rx	Read data, don't care in write mode
X	Don't care

Read or write single byte:



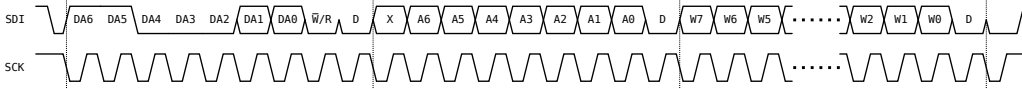
Burst read or write:



6.2 I²C

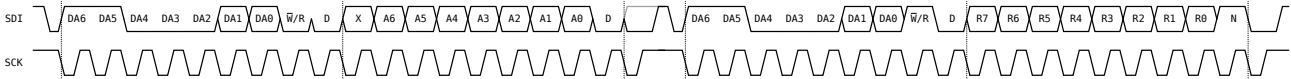
FIELD	DESCRIPTION
DAx	I ² C device address, DA0 & 1 set by I2C_ADDR_x pins
Ax	Register address
Wx	Write data
Rx	Read data
X	Don't care
D	ACK by device
H	ACK by host
N	Host don't ACK after read last byte

Write:

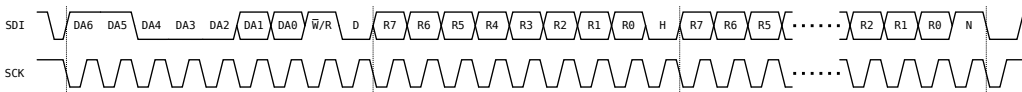


Read:

Write register address first, then read back:



Burst read back:



7 Operate Demonstration

7.1 Init

```
cd_write(REG_SETTING, BIT_SETTING_TX_PUSH_PULL); // enable OUTPUT

cd_write(REG_FILTER, 0x0c); // set FILTER

// set baudrates
cd_write(REG_DIV_LS_L, 39); // 1 Mbps
cd_write(REG_DIV_LS_H, 0);
cd_write(REG_DIV_HS_L, 3); // 10 Mbps
cd_write(REG_DIV_HS_H, 0);

cd_write(REG_RX_CTRL, BIT_RX_RST); // clean RX buffer

// enable interrupts (optional)
// cd_write(REG_INT_MASK, BIT_FLAG_TX_ERROR | BIT_FLAG_RX_ERROR \
| BIT_FLAG_RX_ERROR | BIT_FLAG_RX_PENDING);
```

7.2 TX

```
header_buf[0] = 0x0c; // SRC_ADDR
header_buf[1] = 0x0d; // DST_ADDR
header_buf[2] = 12; // DATA_LEN

cd_write_chunk(REG_TX, header_buf, 3); // Write HEADER
cd_write_chunk(REG_TX, data_buf, header_buf[2]); // Write DATA, do not need to write CRC

// Make sure we can successfully switch to the next page
```

```
while (!(cd_read(REG_INT_FLAG) & BIT_FLAG_TX_BUF_CLEAN));  
  
cd_write(REG_TX_CTRL, BIT_TX_START);           // Trigger send by switching TX page
```

7.3 RX

```
while (!(cd_read(REG_INT_FLAG) & BIT_FLAG_RX_PENDING));  
  
cd_read_chunk(REG_RX, header_buf, 3);         // Read HEADER  
cd_read_chunk(REG_RX, data_buf, header_buf[2]); // Read DATA  
  
cd_write(REG_RX_CTRL, BIT_RX_CLR_PENDING);    // Finish read by switching RX page
```

8 Copyright Statement

The CDBUS protocol is royalty-free for everyone except chip manufacturers.
Copyright (c) 2017 DUKELEC, All rights reserved.

9 Contact Information

- Sales and customer support: sales@dukelec.com
- Technical support: support@dukelec.com
- Business corporation: info@dukelec.com
- Website: <http://dukelec.com>

10 Change History

- 20180603 (v5): Failed to send continuous conflict increased from 4 times to 16 times.
- 20180406: Add description for the RST_N pin: reset is required when the power supply ramp rate is not guaranteed.
- 20180314 (v4): Add tx_abort, and now it's safe to dynamically modify idle_wait_len and tx_wait_len.