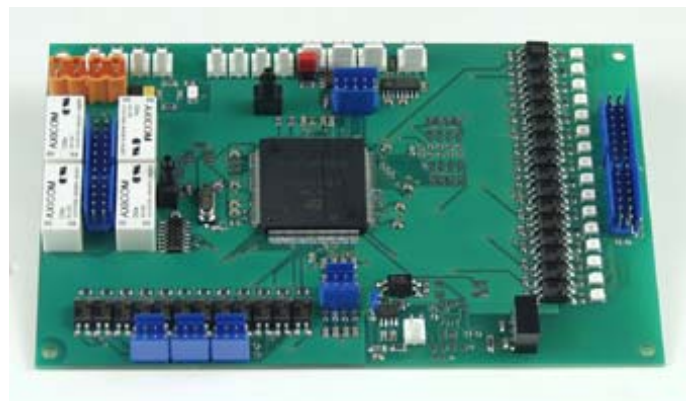


Neobotix IOBoard 16/16

Technical Description



author: Dr.-Ing. Oliver Barth

date: 11.2004

version: 1.02

Contents

1	Introduction	3
1.1	Technical Data IOBoard 16/16	3
2	Software	3
3	CAN communication protocol	4
3.1	CAN Message description	5
4	RS-232 communication protocol	7
5	IOBoard States	7
6	Picture of IOBoard16/16	8
7	Connector assignment	9
8	Appendix A: Parts	10

1 Introduction

The IOBoard 16/16 is designed for multipurpose digital input/ output and analog input. It provides optoisolated digital input and output, relay output and analog input. Special features include optoisolated analog input and incremental encoder interfaces.

1.1 Technical Data IOBoard 16/16

- 16 optoisolated digital inputs
- 12 optoisolated digital outputs
- 4 relays, 2 amperes max. current
- 2 optoisolated analog inputs, input voltage range 0 .. 48 V
- 3 incremental encoder interfaces
- 8 analog input, voltage range 0 .. 5 V
- SSI interface
- Single supply voltage +5V, 400 mA
- Communication interface CAN or RS-232

2 Software

The software is running on a 16 bit/ 40 MHz CPU. It provides access to all board functionality.

The board may be used by RS-232 interface or CAN bus or both at the same time.

The baud rate values are fixed:

CAN	RS 232
500.000 Baud	19.200 Baud

If other baud rates are requested, please contact Neobotix.

The CAN standard receive identifier is 0x101, the standard send identifier is 0x100. If other identifiers are requested, please contact Neobotix.

3 CAN communication protocol

The communication is based on 8 byte CAN messages, so the data length code of send and receive messages is 8 byte always.

Command values in CAN messages are taken from an enumeration named CmdIOBoard:

```
enum CMD_IOBOARD
{
    CMD_IOBOARD_CONNECT,
    CMD_IOBOARD_DISCONNECT,
    CMD_IOBOARD_GETDIGIN,
    CMD_IOBOARD_SETDIGOUT,
    CMD_IOBOARD_GETANALOGIN,
    CMD_IOBOARD_GETGYROVAL,
    CMD_IOBOARD_ZEROGYRO,
    CMD_IOBOARD_GETJOYVAL,
    CMD_IOBOARD_GETVBATT,
    CMD_IOBOARD_GETSTATUS,
    CMD_IOBOARD_SETCTRLPARA,
    CMD_IOBOARD_GETCTRLPARA,
    CMD_IOBOARD_UNKNOWN,
    CMD_IOBOARD_WRTIETOLCD
    CMD_IOBOARD_INITLCD
};
```

To evaluate received messages the command is returned in byte 8 of the message left shift by two. The first two bits of byte 8 are used for returning if the command was executed successfully or not. The enumeration *MsgState* defines return codes as follows:

```
enum MsgState
{
    MSG_OK,
    MSG_ERROR,
    MSG_NOT_ACCEPT,
    MSG_NO_ACTION
};
```

3.1 CAN Message description

Command	CMD_GYROBOARD_CONNECT
Purpose	Connect to the microcontroller, switches the internal state machine to connected
Message format	(CMD_GYROBOARD_CONNECT, 0, 0, 0, 0, 0, 0)
Answer	(0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, MSG_OK (CMD_GYROBOARD_CONNECT << 2) - numbers in return message are for testing purposes only

Command	CMD_GYROBOARD_DISCONNECT
Purpose	Disconnect from the microcontroller, switches the internal state machine to disconnected
Message format	(CMD_GYROBOARD_DISCONNECT, 0, 0, 0, 0, 0, 0)
Answer	No answer

Command	CMD_IOBOARD_GETDIGIN
Purpose	Get the state of digital optoisolated inputs stored in 16 bit integer value P2, 1 bit per input, 16 inputs
Message format	(CMD_IOBOARD_GETDIGIN, 0, 0, 0, 0, 0, 0)
Answer	(P2 >> 8, P2, 0, 0, 0, 0, (CMD_IOBOARD_GETDIGIN << 2) Msg_OK)

Command	CMD_IOBOARD_GETDIGOUT
Purpose	Set digital optoisolated outputs P7 and P8 of the controller, ports have 8 bit each
Message format	(CMD_IOBOARD_SETDIGOUT, P7, P8, 0, 0, 0, 0) - P7: unsigned char, 1 bit per port - P8: unsigned char, 1 bit per port
Answer	No Answer

Command	CMD_IOBOARD_GETANALOGIN
Purpose	Get a single analog input value from Channel, return value has 10 bit. If Channel = 16, 8 analog values are sent in two consecutive CAN messages
Message format	CMD_IOBOARD_GETANALOGIN, Channel, 0, 0, 0, 0, 0)
Answer	If Channel != 16, return message is (AnalogVal >> 8, AnalogVal, 0, 0, 0, 0, (CMD_IOBOARD_GETANALOGIN << 2) Msg_OK); - AnalogVal: value of requested channel If Channel == 16, return messages are (((iAnIn[3] & 0x300) >> 2) ((iAnIn[2] & 0x300) >> 4) ((iAnIn[1] & 0x300) >> 6) ((iAnIn[0] & 0x300) >> 8), 0, 0, MSG_OK (CMD_IOBOARD_GETANALOGIN << 2)) (((iAnIn[7] & 0x300) >> 2) ((iAnIn[6] & 0x300) >> 4) ((iAnIn[5] & 0x300) >> 6) ((iAnIn[4] & 0x300) >> 8), 1, 0, MSG_OK (CMD_IOBOARD_GETANALOGIN << 2));

Command	CMD_IOBOARD_GETVBATT
Purpose	Returns the battery voltage, measured at optoisolated analog input 1
Message format	(CMD_IOBOARD_GETVBATT, 0, 0, 0, 0, 0, 0, 0)
Answer	(iVBatt >> 8, iVBatt, 0, 0, 0, 0, 0, (CMD_IOBOARD_GETVBATT << 2) Msg_OK) - iVBatt: Battery voltage, units Volt

Command	CMD_IOBOARD_GETSTATUS
Purpose	Returns the status of the IOBoard as four CAN messages which consists of joystick values, gyro value, battery voltage, digital input P2 and analog inputs
Message format	(CMD_IOBOARD_GETSTATUS, 0, 0, 0, 0, 0, 0, 0)
Answer	<p>1. message: (g_iJoyValX, g_iJoyValY, g_iVBatt >> 8, g_iVBatt, P2 >> 8, P2, 0, (CMD_IOBOARD_GETSTATUS << 2) Msg_OK);</p> <p>2. message: (IGyroVal >> 24, IGyroVal >> 16, IGyroVal >> 8, IGyroVal, 0, 0, 0, (CMD_IOBOARD_GETGYROVAL << 2) Msg_OK);</p> <p>3. message: (iAnIn[0], iAnIn[1], iAnIn[2], iAnIn[3], ((iAnIn[3] & 0x300) >> 2) ((iAnIn[2] & 0x300) >> 4) ((iAnIn[1] & 0x300) >> 6) ((iAnIn[0] & 0x300) >> 8), 0, 0, Msg_OK (CMD_IOBOARD_GETANALOGIN << 2));</p> <p>4. message: (iAnIn[4], iAnIn[5], iAnIn[6], iAnIn[7], ((iAnIn[7] & 0x300) >> 2) ((iAnIn[6] & 0x300) >> 4) ((iAnIn[5] & 0x300) >> 6) ((iAnIn[4] & 0x300) >> 8), 1, 0, Msg_OK (CMD_IOBOARD_GETANALOGIN << 2));</p>

Command	CMD_IOBOARD_WRITETOLCD
Purpose	Transfers iData to the LCD controller which is attached to the SSI port (optionally)
Message format	(CMD_IOBOARD_WRITETOLCD, iData (HighByte, LowByte), 0, 0, 0, 0, 0)
Answer	(0, 0, 0, 0, 0, 0, 0, Msg_OK)

Command	CMD_IOBOARD_INITLCD
Purpose	Initializes the LCD controller which is attached to the SSI port (optionally)
Message format	(CMD_IOBOARD_INITLCD, iData (HighByte, LowByte), 0, 0, 0, 0, 0)
Answer	(0, 0, 0, 0, 0, 0, 0, (CMD_IOBOARD_INITLCD << 2) MSG_OK)

4 RS-232 communication protocol

The RS-232 protocol is identical to the CAN protocol. Send 8 byte formatted as written above without any delimiter like LF or CR.

5 IOBoard States

The IOBoard has two states

- *State Idle*
- *State Connected*

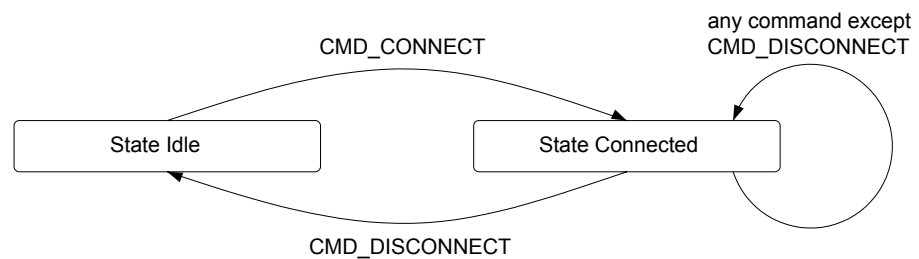


Figure 5-1: States of IOBoard

Only in *State Connected* sensors can be activated/ deactivated or sensor values can be requested.

6 Picture of IOBoard16/16

Figure 6-1 shows a picture of IOBoard16/16. For pin assignment, please see 7. Connector assignment.

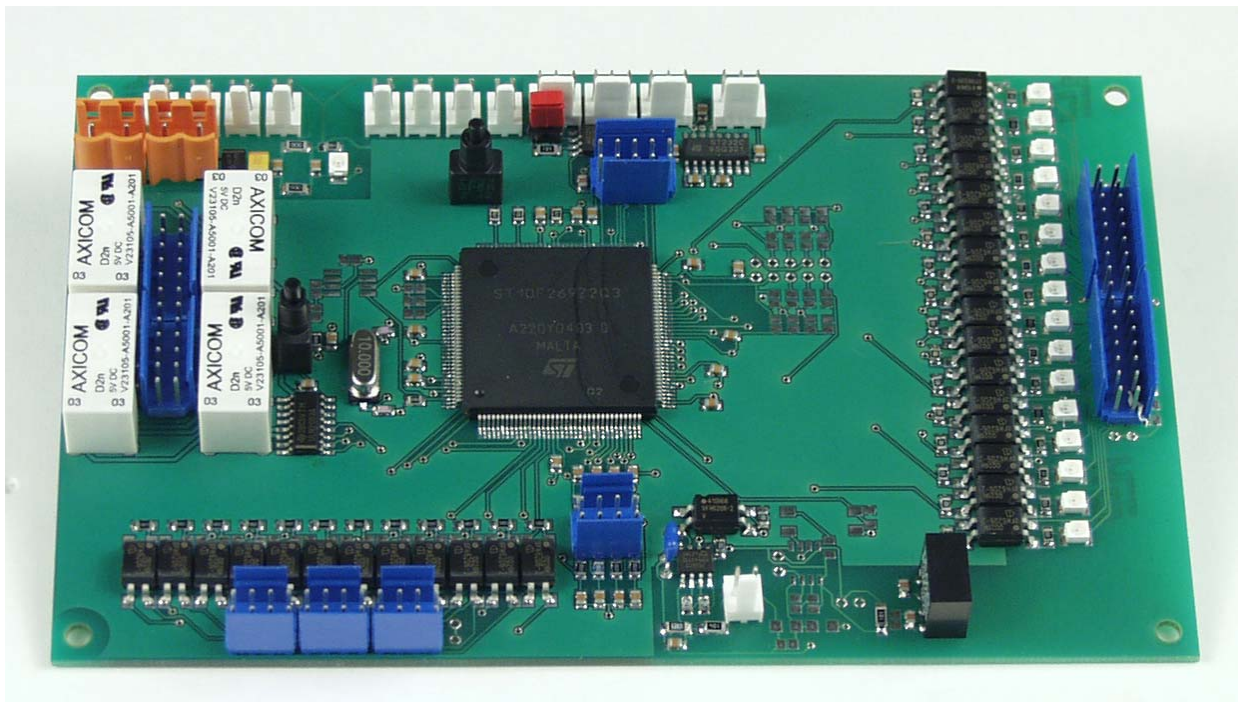


Figure 6-1: picture of IOBoard16/16

7 Connector assignment

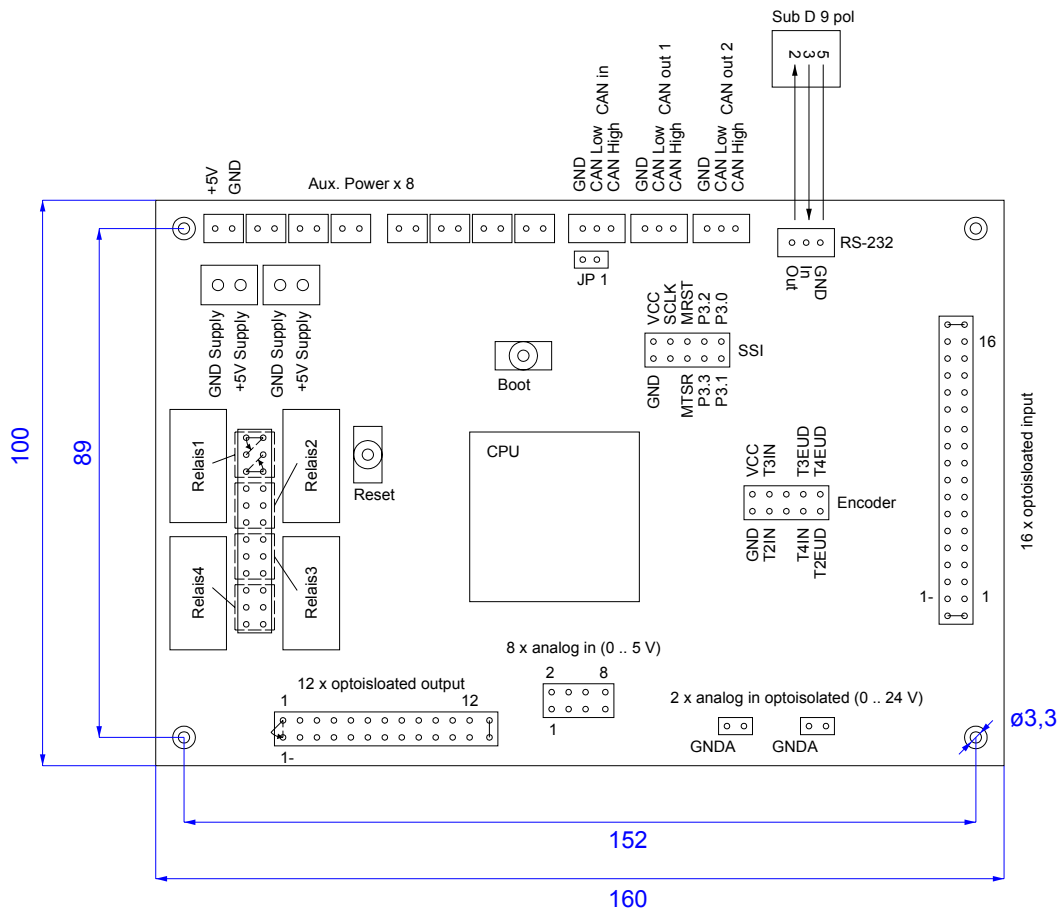


Figure 7-1: Connector assignment of IOBoard16/16

Note: Connect JP1 for activation of CAN terminal resistor (120 Ohm).

8 Appendix A: Parts

For all wire-to-board connectors type and suppliers are listed below. (RS = RS-Components (www.rs-components.de). Alternative supplier is Farnell (www.farnell.de).

Connector for CAN in, CAN out, RS-232

Steckverbinder Molex

Gerade Stiftleisten mit Sperre, 3 Pol (VP=10 Stück)	RS - 453-167	4,75
Buchsengehäuse, 3 Pol (VP=10 Stück)	RS - 467-605	2,58
Crimpkontakte (VP=100 Stück)	RS - 467-598	7,8

Connector for ultrasonic sensor

Steckverbinder Tyco

Stiftleisten mit Kontaktschutz, 3 einreihig (VP=10 Stück)	RS - 531-970	3,15
Kupplung für Crimp-Anschluss, 3 einreihig (VP=10 Stück)	RS - 532-333	2,25
Crimp Kontakte (VP=100 Stück)	RS - 532-456	11,6

Connector for power supply

Steckverbinder Weidmüller

Leiterplattenstecker gerade, 2 Pol (VP=5 Stück)	RS - 403-932	1,45
Klemmleisten für Kabelmontage, 2 Pol (VP = 5 Stück)	RS - 403-875	3,25