Version: 1.00

LEGO® MINDSTORMS® NXT
Hardware Developer Kit

TABLE OF CONTENTS

HARDWARE SPECIFICATION FOR THE NXT BRICK

The LEGO® MINDSTORMS® NXT brick uses various advanced electronics to yield its broad functionality. To view the hardware schematics of the LEGO® MINDSTORMS® NXT, see Appendix 1 and 2; for hardware schematics of the LEGO MINDSTORMS® NXT® sensors, see Appendices 3-6.

Here is a summary list of hardware specifications for the NXT brick:

Main processor:	Atmel® 32-bit ARM® processor, AT91SAM7S256 - 256 KB FLASH - 64 KB RAM - 48 MHz
Co-processor:	Atmel® 8-bit AVR processor, ATmega48 - 4 KB FLASH - 512 Byte RAM - 8 MHz
Bluetooth wireless communication	CSR BlueCore™ 4 v2.0 +EDR System - Supporting the Serial Port Profile (SPP) - Internal 47 KByte RAM - External 8 MBit FLASH - 26 MHz
USB 2.0 communication	Full speed port (12 Mbit/s)
4 input ports	6-wire interface supporting both digital and analog interface - 1 high speed port, IEC 61158 Type 4/EN 50170 compliant
3 output ports	6-wire interface supporting input from encoders
Display	100 x 64 pixel LCD black & white graphical display - View area: 26 X 40.6 mm
Loudspeaker	Sound output channel with 8-bit resolution - Supporting a sample rate of 2-16 KHz
4 button user-interface	Rubber buttons
Power source	6 AA batteries - Alkaline batteries are recommended - Rechargeable Lithium-Ion battery 1400 mAH is available
Connector	6-wire industry-standard connector, RJ12 Right side adjustment

NXT TECHNICAL OVERVIEW

This section provides a graphical overview of how different functions are connected and controlled within the intelligent brick. The figure only includes higher-level units within the NXT. For detailed information on how individual elements are connected, see the hardware schematic in Appendix 1 and 2.

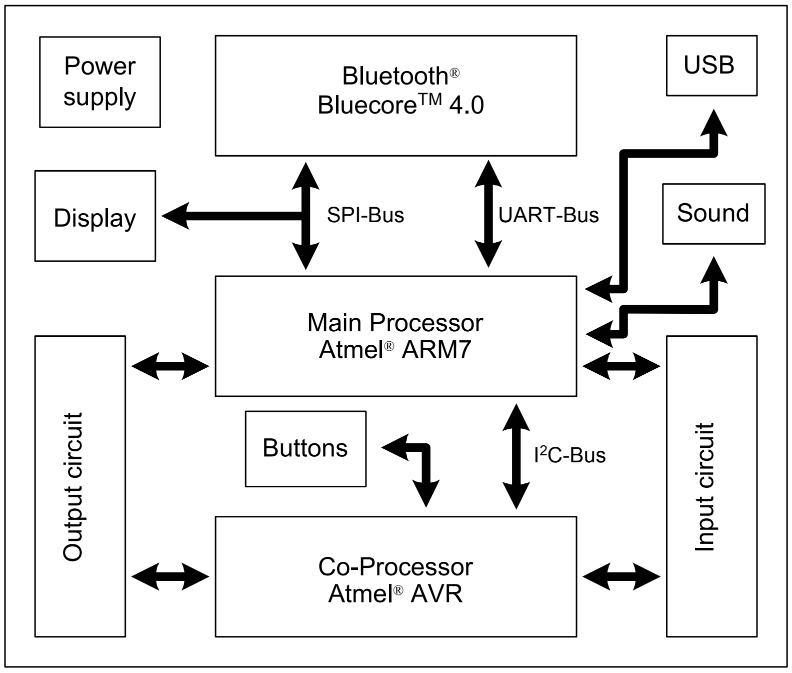


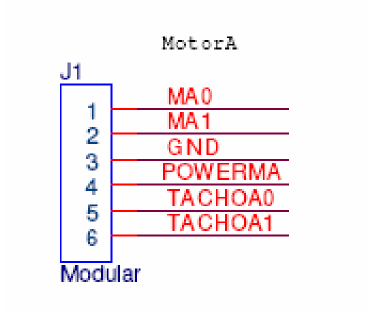
Figure 1: Hardware block diagram of the NXT brick

OUTPUT PORTS

The LEGO® MINDSTORMS® NXT has three output ports used for controlling actuators connected to the NXT brick.

A 6-wire digital user interface on the output ports was implemented so that output devices could send information back to the NXT brick without having to use up an input port as well.

The figure below shows the schematic details behind port A of the brick. The schematics for ports B and C are identical.



Pin 1, MA0	PWM output signal for the actuators
Pin 2, MA1	PWM output signal for the actuators
Pin 3, GND	Ground signal related to the output supply
Pin 4, POWERMA	4.3 Volt output supply
Pin 5, TACHOA0	Input value that includes Schmitt trigger functionality
Pin 6, TACHOA1	Input value that includes Schmitt trigger functionality

MA0 and MA1 are output signals for controlling actuators. These signals are controlled by an internal motor driver which can supply a continuous 700 mA to each output port and a peak current of approximately 1 A. The output signal is a PWM signal, which can be controlled to either break or float between the signals. The motor driver has thermal protection built-in, which means that if too much power is continually drawn from the brick, the motor driver will automatically adjust the output current.

The output power (POWERMA) is connected internally to all of the power outputs in the output and input ports. The maximum output current that can be drawn from this supply is approximately 180 mA. This means that each port has approximately 20 mA. If more power is drawn, the total output current will be decreased automatically without further warning. If the power signal is short circuited to ground, the NXT brick will reset.

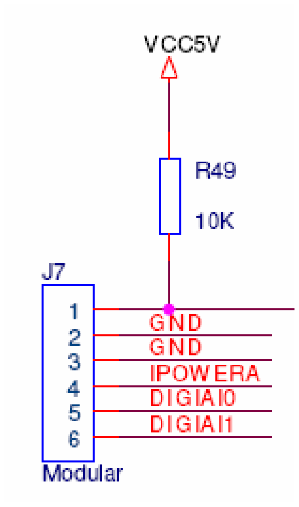
The TACHOA0 and TACHOA1 are input ports that have a Schmitt trigger mounted between the ports and the input pins on the ARM7 processor. These two signals allow the possibility of having a quadrature detector within the system. Within the standard firmware these two signals are used to count the numbers of tacho pulses from the motors and detect whether the motor is running clockwise or counterclockwise.

INPUT PORTS

The LEGO® MINDSTORMS® NXT has four input ports that allow the NXT to measure different parameters in the physical world (depending on the connected sensor).

The 6-wire digital user interface on the input ports enables having both an analog and digital interface within each connector. This allows the possibility of developing both analog and digital sensors for the NXT brick.

The figure below shows the schematic details behind port 1 on the brick. Ports 2, 3, and 4 have identical schematics. Behind port 4, the digital pins (DIGIxI0 and DIGIxI1) are connected to a RS485 controller that handles high-speed communication.



• Pin 1, ANA	Analog input and possible current output signal
• Pin 2, GND	Ground signal
• Pin 3, GND	Ground signal
• Pin 4, IPOWERA	4.3 Volt output supply
• Pin 5, DIGIAI0	Digital I/O pin connected to the ARM7 processor
• Pin 6, DIGIAI1	Digital I/O pin connected to the ARM7 processor

The input pin (ANA) is the analog input pin that is connected to a 10-bit A/D converter within the AVR processor. This is also connected to the current generator which is used for generating power for the active LEGO® MINDSTORMS® Robotic Invention System sensors. (See the section about active sensors in this chapter.) The A/D input signals are sampled with the same sampling rate for all analog sensors. As described in the active sensor documentation, analog sensors need 3 mS of supply power output before any measurements can occur. The sampling rate used for all analog sensors is 333 Hz.

Output power (IPOWERA) is connected internally to all of the power outputs in the output and input ports. The maximum output current that can be drawn from this supply is approximately 180 mA. This means that each port has approximately 20 mA available. If more power is drawn, the total output current will be decreased automatically without further warning. If the power signal is short circuited to ground, the NXT brick will reset.

The digital I/O pins (DIGIAI0 & DIGIAI1) are used for digital communication implemented as I^2C compliant communication running at 9600 bit/s. The NXT can only function as a master in relation to I^2C communication and requires that external devices have pull-up resistors included on their communication pins. See the I^2C Communication chapter for further details.

In addition, the I/O pin on the ARM7 that is connected to DIGIXI1 can be set up to function as an analog input pin (although this is not supported directly within the standard firmware). This allows the possibility of implementing an analog input pin with a higher sampling rate.

ACTIVE SENSORS

To ensure backwards compatibility with LEGO® MINDSTORMS® Robotic Invention System sensors (developed for the RCX intelligent brick), a current generator has been added to the NXT brick to yield correct power and measurement intervals for these older sensors. Together with the standard LEGO firmware the current generator yields the same functionality as is available within the RCX intelligent brick. The current generator provides approximately 18 mA of output current.

The generator controls the power delivery to active sensors. It supplies the sensor with power for 3 mS and then measures the analog value during the following 0.1 mS.

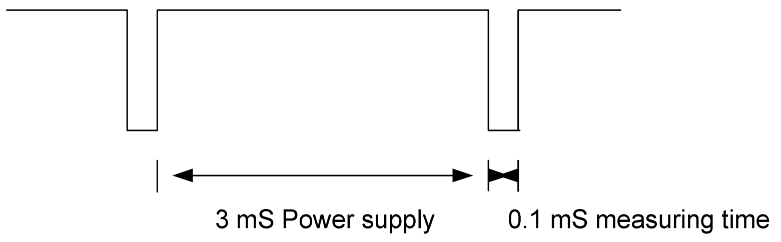


Figure 2: Timing diagram for the A/D input signal pin when using active sensors

The following sensors from the LEGO® MINDSTORMS® Robotics Invention system are active sensors:

- Light sensor
- Rotation sensor

PASSIVE SENSORS

All sensors that do not need the special power/measurement timing mentioned above are called passive sensors. These sensors are also sampled every 3 mS because sampling using the A/D converter is simultaneous and therefore, must uphold the timing required by the active sensors.

The following sensors are passive sensors:

- Touch sensor (both the RCX and NXT versions)
- Light sensor supplied in the LEGO MINDSTORMS® NXT sets
- Sound sensor
- Temperature sensor

DIGITAL SENSORS

All sensors that use I²C communication are named digital sensors because they include an external micro-controller that handles the sampling of the physical environment.

The following sensors are digital sensors:

- Ultrasonic sensor

HIGH-SPEED COMMUNICATION PORT

Port 4 on the NXT intelligent brick can function as a high-speed communication port. A RS485 communication chip is implemented behind the normal input circuit. This allows the implementation of high-speed bi-directional data communication on a multipoint data line over wider distances. Currently LEGO® has not developed any devices that need this communication functionality. However, if future devices are developed that require higher communication speeds, the NXT brick is prepared. For such future devices, LEGO may use the P-Net communication protocol (www.P-net.org), which enables multipoint data communication.

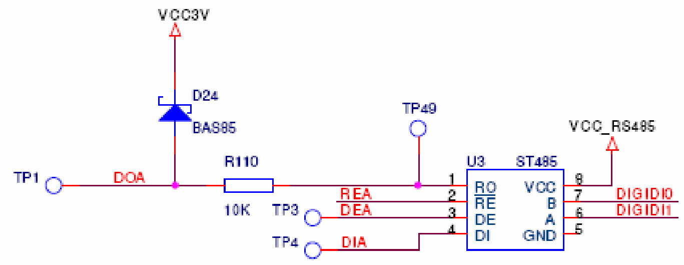


Figure 3: Hardware schematic for the RS485 chip behind port 4 on the NXT brick

Note: The RS485 chip is using 5 volts as its supply voltage and the ARM7 processor is using 3.3 volts. For this reason a level shifter has been applied between the RS485 chip and the ARM7 processor. See the schematics for further details.

The following communication parameters are set up for high-speed communication within the standard firmware:

• Communication speed:	921.6 Kbit/s
• Data bits:	8 bit
• Stop bit:	1 bit
• Parity:	0 bit

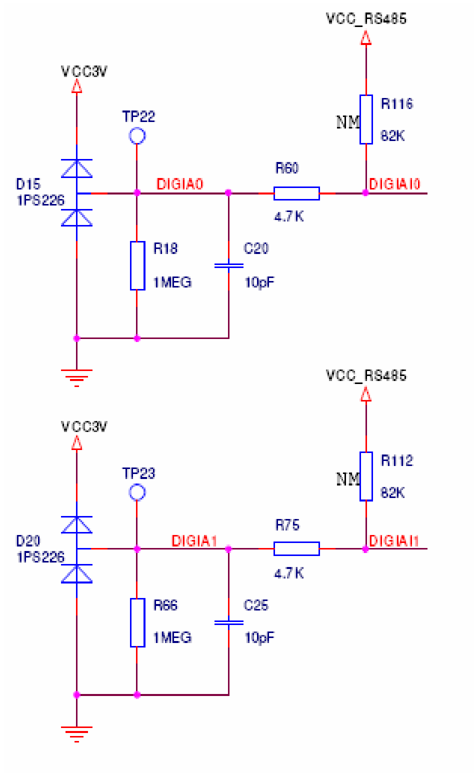
I²C COMMUNICATION

Within the NXT brick, a digital interface has been implemented using the I²C protocol. I²C is an industrial communication standard that was developed by Philips Semiconductors in the early 1980s. It has since been used in many different industrial components where simple digital communication is required.

I²C communication functions as the digital interface for external devices which needs to communicate with the NXT. Having a digital interface enables external devices to perform functionality individually and then only send the result back to the NXT or receive new information from the NXT.

The NXT brick has four I²C communication channels, one for each of the four input ports. The I²C digital communication is implemented as "master only" functionality, which means that the NXT is controlling the dataflow in each of the communication channels.

An important aspect to allowing I²C communication between two devices is the hardware set-up within each of the devices. The figure below shows the hardware schematic internally for input port 1 in the NXT. The schematic for input ports 2, 3 and 4 are the same with respect to I²C communication.



Important parameters to notice:

- There is a 4.7 k resistor in serial with the signal line.
- There is no pull-up resistor mounted internally in the NXT. This needs to be mounted in the external device. We recommend using a 82 K resistors as pull-up resistors on both the data and clock lines.
- DIGIx0 (Pin 5 within the connector) is the CLK signal and DIGIx1 (Pin 6 within the connector) is the DATA signal for I²C communication.
- The digital I/O pins on the NXT cannot be set to open drain directly. Therefore the NXT will drive the digital I/O pins either high or low depending on the situation, i.e., the NXT uses Push-Pull. When the NXT should not control the I/O lines, it will be set as input (e.g., when reading data from a device or when reading acknowledgement).
- The I²C communication is running at 9600 bit/s.
- Each channel has a 16 byte input buffer and a 16 byte output buffer. Therefore a maximum of 16 bytes can be sent and received during each data communication cycle.
- If multiple sensors are connected in sequence to the same sensor port, the resulting pull-up resistor needs to be of 82 k. For this reason, some consideration is needed when multiple sensors are connected in sequence to the same port.

Digital devices have some advantages compared to analog devices. Digital devices can include device names and can reference various individual parameters that are device specific (like calibration values, startup times, and so on). To be able to distinguish different digital device from each other, LEGO® has started an addressing scheme for its sensors that will expanded as the company develops new digital devices or approves third-party devices. Currently, the applicable list includes only the Ultrasonic sensor, which has been given address 1 (within a 7 bit context). This address is sent together with the communication direction bit, and as with all other data bytes, the address is transferred with the most significant bit first.

DEVICE MEMORY ARRANGEMENT

When using I^{2}C communication, there are multiple ways to define and implement the functionality for reading and writing data to and from an external device.

We characterize LEGO external devices as external memory areas from which we can read data or to which we can write data. By knowing the specifics behind each of the data memory locations in external devices, it's possible to control the device and read the desired data. The example memory map below shows how memory can be divided into areas that enable easier read and write access:

	Transmitted from NXT			
Command	**Byte 0**	**Byte 1**	**Byte 2**	**Length**
		Addr.		
Constants				
Read version	Device address	0x00	R + 0x03	8
Read product ID	Device address	0x08	R + 0x03	8
Read sensor type	Device address	0x10	R + 0x03	8
Read factory zero (Cal 1)	Device address	0x18	R + 0x03	1
Read factory scale factor (Cal 2)	Device address	0x19	R + 0x03	1
Read factory scale divisor	Device address	0x1A	R + 0x03	1
Read measurement units	Device address	0x1B	R + 0x03	7
Variables				
Read variable 1	Device address	0x40	R + 0x03	1
Read variable 2	Device address	0x41		
…				
Commands				
Command 1	Device address	0x80	0xXX	
Command 2	Device address	0x81		
…				

Figure 4: Example of a memory map for an external digital device

By building up the memory with the trailing commands as shown above, it is much easier to read data from the sensors and potentially set multiple parameters in one read/write cycle. For details on how the memory is divided up for the Ultrasonic sensor, see Appendix 7.

DISPLAY

A dot matrix display has been added to the NXT brick to improve the user interface. This display is a black and white graphical LCD display with a resolution of 100 x 64 pixels. The display has a viewing area of 26 x 40.6 mm. The LCD-controller used to control the display is an UltraChip 1601. See the Links section for detailed data sheets on the display's LCD-controller.

There is an SPI interface from the ARM7 microcontroller to the display's LCD-controller UltraChip 1601. The SPI interface is running at 2 MHz within the standard LEGO® firmware and two full memory maps are set aside within the firmware for updating the display. The display is continuously updated in a line sequence requiring 17 mS for a total display update.

LCD data is allocated within memory as a two dimensional array, Normal[8][100] (Normal[Y / 8][X]). Data is sent to the LCD controller in the following order: the first byte controls the first 8 pixel vertical (starting at (0,0)) and the second byte controls the next 8 vertical pixel horizontal.

The pixels within the display are allocated as follows:

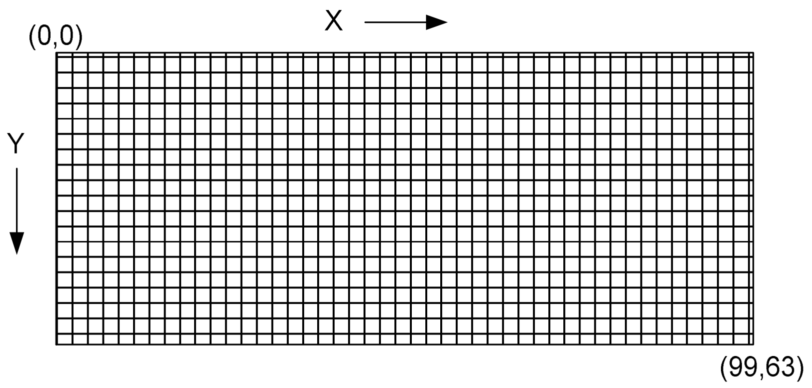


Figure 5: Bitmapping within the display

Technical specifications for the display:

• Format:	100 x 64 dots
• LCD mode:	STN / Positive Reflective Mode / Gray
• Viewing direction:	6 o'clock
• Driving scheme:	1/65 duty cycle, 1/9 bias
• Power supply voltage (V_{DD}):	3.0V
• LCD driving voltage (VLCD):	9.0V (adjustable for best contrast)

BLUETOOTH®

The NXT brick supports wireless communication using Bluetooth® by including a CSR BlueCore™ 4 version 2 chip. The NXT brick can be connected wirelessly to three other devices at the same time but can only communicate with one device at a time. This functionality has been implemented using the Serial Port Profile (SPP), which can be considered a wireless serial port. The NXT brick can communicate with Bluetooth devices that can be programmed to communicate using the LEGO® MINDSTORMS® NXT Communication Protocol and that support the Serial Port Profile (SPP). It's possible to send programs and sound files between NXT bricks and to use wireless communication to send and receive information between bricks during program execution. To reduce the power consumption used by Bluetooth, the technology has been implemented as a Bluetooth® Class II device, which means that it can communicate up to a distance of approximately 10 meters.

BLUETOOTH® FUNCTIONALITY WITHIN THE NXT BRICK

The Bluetooth® functionality within the NXT brick is set up as a master/slave communication channel. This means that one NXT within the network needs to function as the master unit and that other NXT bricks communicate through it if they need to. The figure below shows which NXT devices can communicate directly within a network.

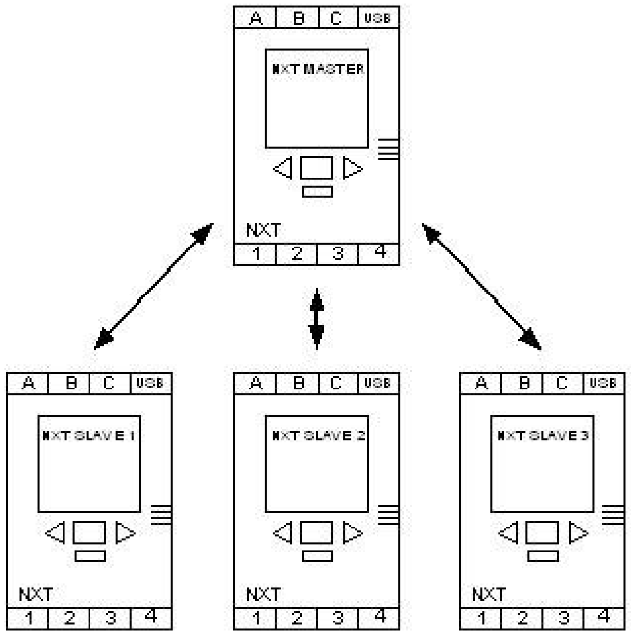


Figure 6: NXTs communicating using Bluetooth®

As shown in the figure above, the master NXT can be connected to three other Bluetooth® devices at the same time. The master NXT can only communicate with one of the slave units during a given moment, meaning that if the master NXT is communicating with NXT Slave 1 and NXT Slave 3 starts sending data to the master NXT, the master NXT will not evaluate the received data until it switches to NXT slave 3.
An NXT is not able to function as both a master and slave device at the same time because this could cause lost data between NXT devices.

Connections to other Bluetooth devices occur through channels. The NXT has four connection channels used for Bluetooth communication. Channel 0 is always used by slave NXT devices in communicating with the master NXT (i.e., *towards* the master NXT) while channels 1, 2, and 3 are used for outgoing communication *from* the master device to the slave devices.

INTERFACING WITH THE BLUECORE™ CHIP

Bluetooth® functionality within the NXT is implemented using a standalone chip, a CSR BlueCore™ 4 with an external 8 Mbit FLASH memory. The Bluetooth chip from CSR contains all the necessary hardware to run a self-contained Bluetooth node. A 16-bit integrated processor runs the Bluetooth stack implemented by CSR, called Bluelab. Bluetooth® is implemented within the NXT using version 3.2 of Bluelab. The firmware within the BlueCore integrates a user programmable VM-task allowing us to control and run small amounts of application code. A command interpreter is integrated within the VM that is able to decode and respond to commands received through the UART interface from the ARM7 processor.

The VM has a full implementation of both the Bluetooth SPP-A and SPP-B profiles. The SPP-A profile is used when the local BlueCore is the connection initiator (MASTER device) while the SPP-B profile is used when another Bluetooth device initiates the connection (SLAVE device). The BlueCore uses what is referred to as “stream-mode” to exchange data at a rate of <= 220 K baud after a connection is established. When BlueCore is not in “stream-mode,” it is in “command-mode” which is used to control the VM application within BlueCore and by extension, the Bluetooth functionality within the NXT. Which communication type the UART includes is controlled by two interface signals (ARM7__CMD & BC4__CMD).

For a detailed description of the communication protocol used between the ARM7 processor and the BlueCore chip, see Appendix 8.

The figure below shows the interface between the ARM7 processor and the BlueCore chip. (Its functionality is explained below the figure.) For a detailed description of the pin layout, see the hardware schematics for the NXT brick.

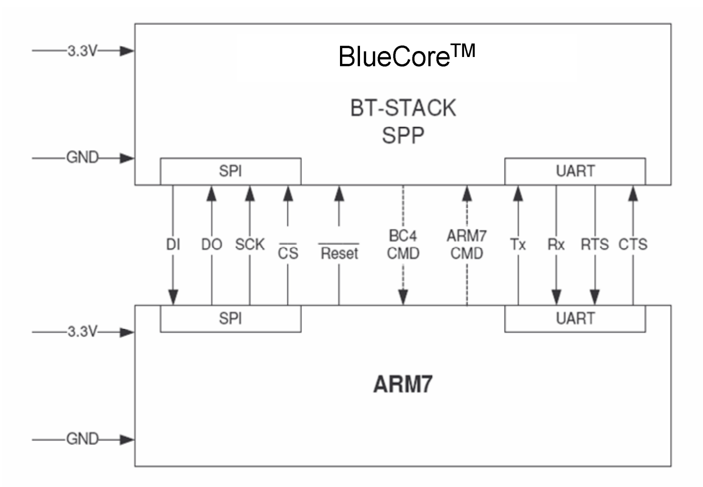


Figure 7: Interface between the ARM7 processor and the BlueCore chip

The SPI interface allows the possibility of updating the BlueCore chip. It is not in use during normal operation of the NXT brick. The SPI interface is shared with the display within the NXT brick.

The Reset pin is used at startup to re-initialize the chip correctly and to disable Bluetooth functionality.

BC4-CMD: Indication from the BlueCore to the ARM7 as to which data type the BlueCore expects to send to the ARM7.

ARM7-CMD: Indication from the ARM7 to the BlueCore as to which data type the ARM7 expects to send to the BlueCore.

UART communication is used for both data and command communication between the BlueCore and the ARM7 processor.

UART interface between the ARM7 and the BlueCore chip

The UART within the BlueCore chip is initialized for communication with the ARM7 using the following set-up (both for stream-mode and command-mode):

Communication speed:	460.8 K bit/s
Data bits:	8 bits
Parity:	No parity bits
Stop bit:	One stop bit
Flow control:	Hardware handshake signals (RTS & CTS)

SOUND

The NXT brick includes a sound amplifier chip to improve the sound output level and quality. The sound output is a PWM output signal that is controlled by the ARM7 microcontroller. The filters introduced before the amplifier will reduce the over-sampling noise in the signal.

The sound driver (SPY0030A) is a differential sound amplifier chip from SUNPLUS that can have a maximum gain of 20. For detailed information about the sound amplifier, see the data sheet for the SUNPLUS sound driver.

The loudspeaker within the NXT is a 16 ohm speaker with a diameter of 21 mm. The table below shows the current and power consumption when sounds are played at two different frequencies.

Sound current consumption

Frequency	Current mA	Power mW
440 Hz	102	169
4 KHz	78	97

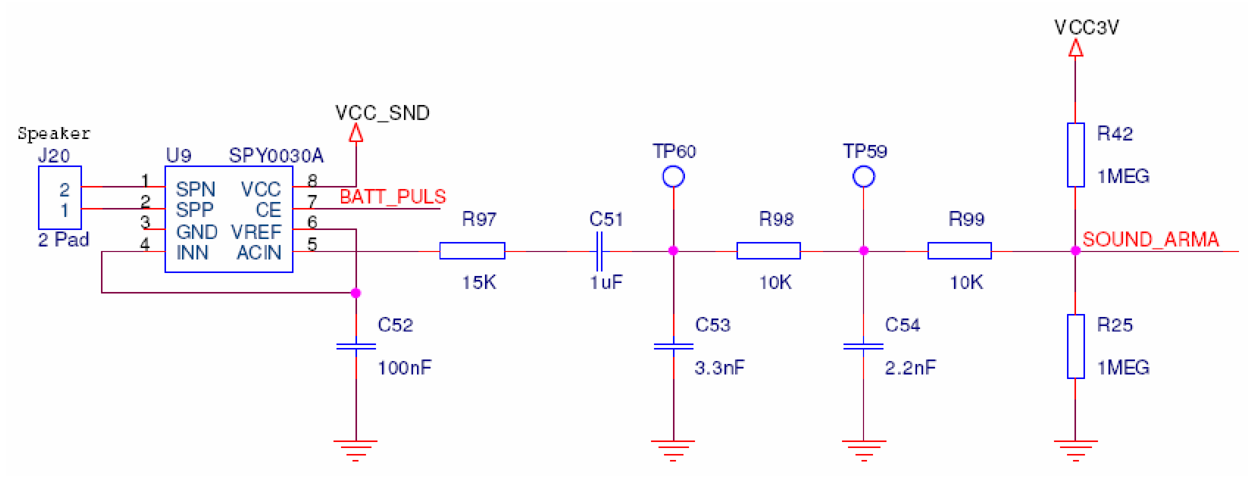


Figure 8: Schematic for sound output within the NXT

DEBUGGING INFORMATION

This section briefly describes how users can interface with the two microcontrollers within the NXT using JTAG interfaces. (Important note: When the NXT is disassembled or when third party firmware is used with the NXT, all warranties are rendered invalid.)

When developing third party firmware for the NXT, developers must be careful when addressing the hardware because incorrect initialization can destroy hardware components. Study the hardware schematic carefully before developing new firmware.

The main processor within the NXT brick is the ARM7 processor that handles all the user-specific functionality. The AVR microcontroller handles lower-level functionality, like controlling the motor PWM, power management, and A/D conversion. To connect a JTAG interface to either the ARM7 or AVR microcontroller, the NXT needs to be disassembled.

INTERFACING WITH THE ARM7 MICROCONTROLLER

The connection points to the ARM7 processor have not been removed from the NXT brick but the necessary hardware to make a connection have not been mounted on the main PCBA to save cost and mounting time during production. Therefore, to interface with the ARM7 processor, a 10-pin connector (single row using 1.27 pitch) must to be mounted to the NXT and a debugging wire connected from the JTAG to the NXT.

The J-tag connector (J17) has the following schematic layout:

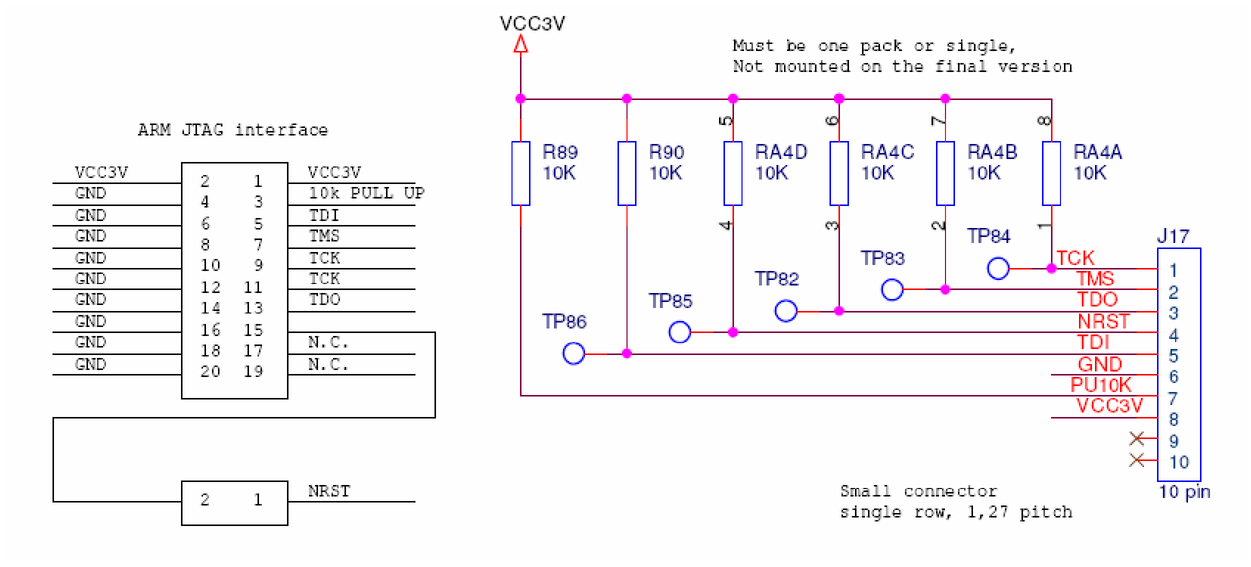


Figure 9: Hardware schematic for interfacing with the ARM7 microcontroller

INTERFACING WITH THE AVR MICROCONTROLLER

The connection points to the AVR processor have not been removed from the brick but the necessary hardware to make a connection has not been mounted on the main PCBA to save cost and mounting time during production. Therefore, to interface with the AVR processor, a 8-pin connector (single row using 1.27 pitch) must be mounted to the NXT and a debugging wire connected from the JTAG to the NXT.

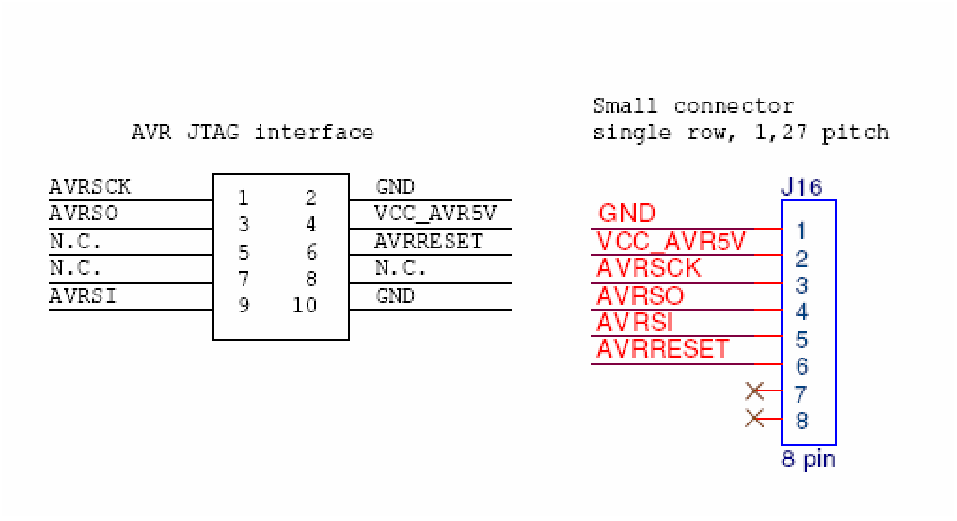


Figure 10: Hardware schematic for interfacing with the AVR microcontroller

FIRMWARE REQUIREMENTS:

The NXT is made up of multiple processors and there are some minor requirements that need to be upheld for the system to function correctly. Some of these requirements are related to the hardware setup; others are implemented in areas that can be changed by the user. This section will describe the requirements that an alternative firmware version must uphold.

For various reasons, overall power distribution within the system is controlled by the AVR microcontroller. A small startup sequence has been implemented to protect the product from misuse. This sequence controls the power to the ARM7 processor and will cause it to be turned off if the ARM7 processor doesn't send back the following message to the AVR microcontroller over I^{2}C communication within 5 minutes of start-up:

"\xCC""Let's samba nxt arm in arm, (c)LEGO System A/S"

This means that the following data should be sent to the AVR:

[0xCC,0x4C,0x65,0x74,0x27,0x73,0x20,0x73,0x61,0x6D,0x62,0x61,0x20,0x6E,0x78,0x74,0x20,0x61,0x72,0x6D,0x20,0x69,0x6E,0x20,0x61,0x72,0x6D,0x2C,0x20,0x28,0x63,0x29,0x4C,0x45,0x47,0x4F,0x20,0x53,0x79,0x73,0x74,0x65,0x6D,0x20,0x41,0x2F,0x53]

AVR TO ARM COMMUNICATION

The interface between the ARM7 microcontroller and the AVR microcontroller is implemented using the hardware I²C communication channel on both the ARM7 and AVR microcontroller. To enable both microcontrollers to run somewhat independently from each other, the communication interface between the two microcontrollers is set up as two memory allocations that are updated on both microcontrollers every 2 mS. The interface is set up to communicate at 380 Kbit/s using the hardware I²C interface within the microcontrollers.

The main tasks for the AVR microcontroller are power management, creating PWM output signals for the three motors, and performing A/D conversion from the input ports. Within the standard NXT firmware, data is sent to the AVR microcontroller through a struct implementation that is continuously updated in the ARM7 microcontroller to match the required functionality of the AVR microcontroller.

Because of limitations within the ARM7 microcontroller chip, the ARM7 can only function as the master within the I²C communication setup. For further details, see the data sheet for the Atmel AT91SAM7S256 microcontroller.

DATA SENT FROM THE ARM7 MICROCONTROLLER

```
typedef   struct
{
  UBYTE   Power;
  UBYTE   PwmFreq;
  SBYTE   PwmValue[NOS_OF_AVR_OUTPUTS];
  UBYTE   OutputMode;
  UBYTE   InputPower;
}IOTOAVR;
```

Power	Command byte that is used during power down and firmware update mode. During normal communication, this byte should be set to zero.
PwmFreq	Holds the PWM frequency used by the PWM signal for the three outputs. It can have the following range: 1 – 32 KHz. Units are in KHz. The standard LEGO firmware uses 8 KHz.
PwmValue	Holds the power level for the individual output. The first element in the array relates to output A. It can range from -100 to +100 where -100 is full power clockwise and +100 is full power counterclockwise.
OutputMode	Holds the output mode that can be either float or break between PWM pulses. 0x00 means break and 0x01 means float.
InputPower	Holds the parameter used to define the 9V sensor supply. Input power is configured as bit fields where bit 0 and 1 is sensor 0, bit 2 and 3 is sensor 1, bit 4 and 5 is sensor 2, and bit 6 and 7 is sensor 3. 0: 9V is off 1: 9V is on unless measuring a sensor (when it is off temporarily each 3 mS) 2: 9V is always on

When controlling the power management and the firmware download mode, other data packages should be sent to the AVR microcontroller but through the struct interface described above.

When powering down the NXT, the Power byte should be set to 0x5A and the PwmFreq should be set to 0x00. This will cause the AVR to turn off the NXT brick and wake up when the “Select” button is pressed.

When downloading new firmware to the NXT brick, a special firmware mode is required within the brick. To enable this functionality, the Power byte should be set to 0xA5 and the PwmFreq should be set to 5A. This will cause the NXT to go into firmware update mode. Firmware update mode is a low-level mode that the ARM7 processor is sent into by the AVR microcontroller. Read more about firmware download mode in the data sheet for the ATMEL AT91SAM7S256 processor, called SAMBA mode.

DATA RECEIVED FROM THE AVR MICROCONTROLLER

```
typedef   struct
{
  UWORD   AdValue[NOS_OF_AVR_INPUTS];
  UWORD   Buttons;
  UWORD   Battery;
}IOFROMAVR;
```

AdValue	Holds the raw value from the 10 bit A/D converter. The first element in the array relates to sensor input 1 on the NXT brick.
Buttons	Holds the status of the buttons. Button 1, 2 and 3 are returned as a 10 bit AD value. Button 0 adds 0x7FF to this. AD values for button thresholds can be calculated from the resistor values on the button PCB schematics.
Battery	Holds information about the measured battery level, whether an Accu pack has been inserted, and the AVR firmware version. 16 bits as follows:

Bit 15	0 = AA batteries 1 = Accu pack inserted
Bit 13 - 14	0..3 = major version
Bit 10 - 12	0..7 = minor version
Bit 0 - 9	0..1023 (multiply with 13.848 to calculate actual mV)

COMMUNICATION SCHEME

As the ARM can only function as a master, it will be the initiator in both receiving and transmitting data to and from the AVR. In normal operation, transmission and reception are interleaved each second millisecond. The only restriction to this communication is that it complies with the transfer data structures mentioned above as these are the only ones supported by the AVR.

The data structures can be transferred at any desired time, meaning that there are no critical dependencies in the firmware. However, sensor and motor timing will be affected by this, i.e., the rotation sensor will not work correctly if data from the AVR is not transferred at a minimum rate of 3 millisecond intervals.

POWER MANAGEMENT

Power within the NXT brick comes from 6 AA batteries or a rechargeable lithium ion battery (that also includes a AC power plug to the LEGO Transformer). Power management within the NXT brick consists of a switch mode power supply, which generates a 5-volt supply from the batteries and from the 5-volt supply, another 3.3-volt supply is generated for the ARM7 processor and the BlueCore chip.

To protect the power supply within the NXT, a poly switch is connected at the beginning of the power circuit. The poly switch has a hold current of 1.85 A and will be triggered at approximately 3.3 A.

Current consumption measurement:

The effects are based on a battery voltage of 9 volts.

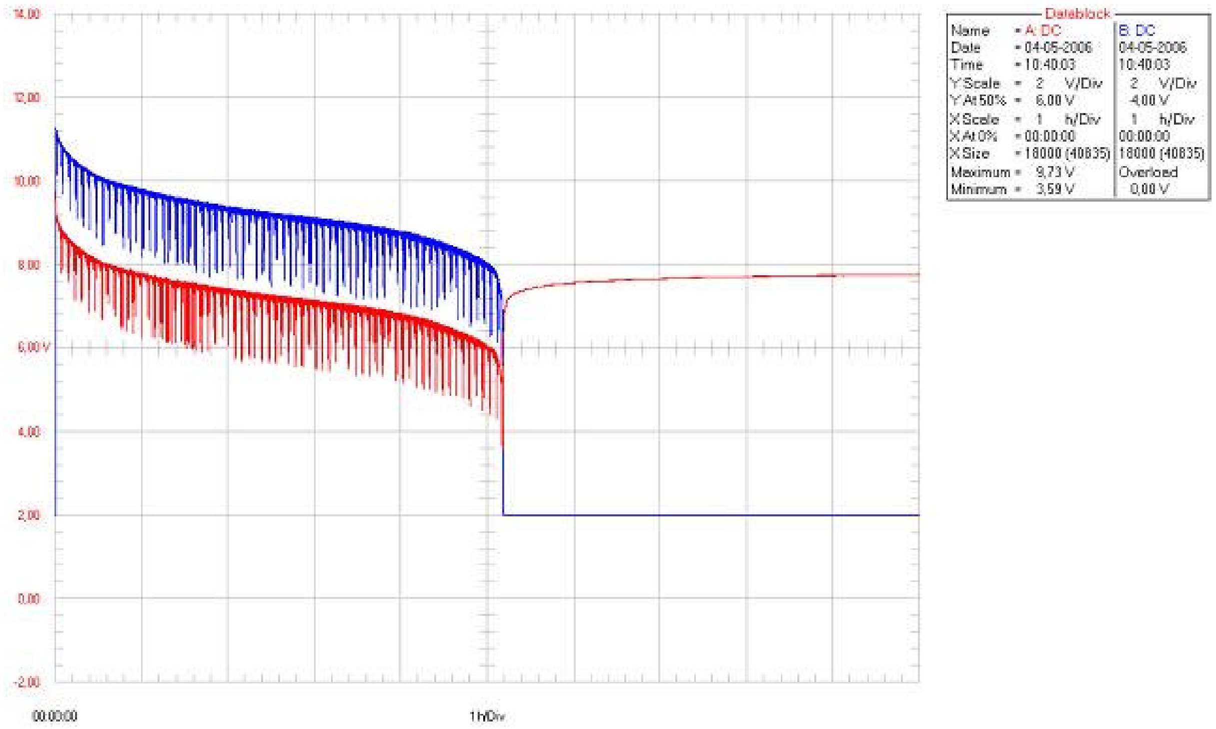
Supply voltage	Current		Effect (Battery = 9 Volts)	
	Max [mA]	Normal [mA]	Max [mW]	Normal [mW]
No load on motors				
9 Volt	339	114	5184	1422
5 Volt	271	112	1744	448
3.3 Volt	72	38	410	216
Load on motors				
9 Volt	2901	848	26109	7632
5 Volt	271	112	1142	307
3.3 Volt	72	38	410	137
Standby	46 uA assumed standby current due to brown out detection			

Figure 11: Current measurement on the MINDSTORMS NXT brick

Battery testing within the LEGO MINDSTORMS NXT

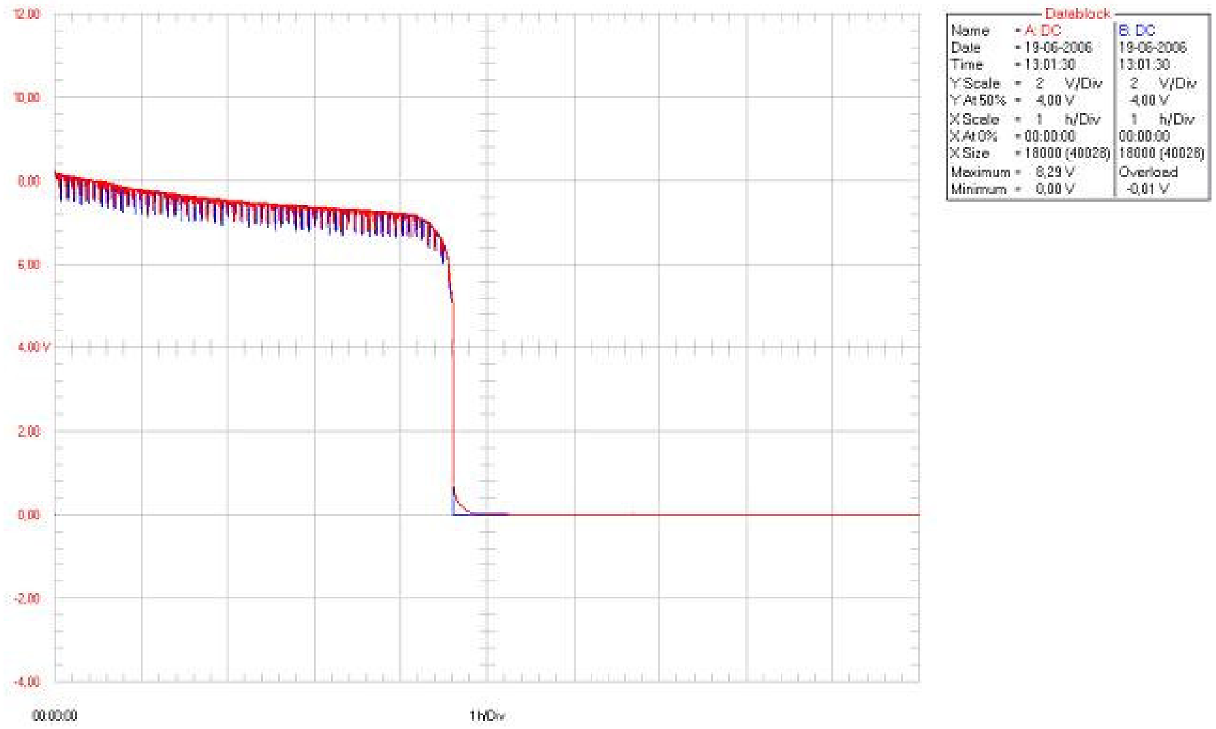
The NXT brick's performance depends on the batteries used and the load applied to the brick. The two figures shown below illustrate the performance of the NXT brick while using 6 standard alkaline batteries and while using the LEGO® MINDSTORMS® Lithium Ion rechargeable battery. The test was performed with two LEGO® MINDSTORMS® NXT motors attached to the NXT with the motors reversing direction every 5 sec while running at full speed.

From this test, it's clear that the NXT brick performs well when running on alkaline batteries, rechargeable lithium ion, and rechargeable Ni-MH batteries. The LEGO Lithium Ion battery is the only solution that allows the NXT to be powered continuously while the battery is connected to A/C power through a transformer. If the load is more than approximately 500mA, the battery will not be recharged.



	A: DC	B: DC
Name	A: DC	B: DC
Date	04-05-2006	04-05-2006
Time	10:40:03	10:40:03
Y Scale	2 V/Div	2 V/Div
Y At 50%	6,00 V	4,00 V
X Scale	1 h/Div	1 h/Div
X At 0%	00:00:00	00:00:00
X Size	18000 (40835)	18000 (40835)
Maximum	9,73 V	Overload
Minimum	3,59 V	0,00 V

Figure 12: Load on the NXT when using standard alkaline batteries



	A: DC	B: DC
Name	A: DC	B: DC
Date	19-06-2006	19-06-2006
Time	13:01:30	13:01:30
Y Scale	2 V/Div	2 V/Div
Y At 50%	4,00 V	4,00 V
X Scale	1 h/Div	1 h/Div
X At 0%	00:00:00	00:00:00
X Size	18000 (40028)	18000 (40028)
Maximum	8,29 V	Overload
Minimum	0,00 V	-0,01 V

Figure 13: Load on the NXT when using the LEGO MINDSTORMS Lithium Ion rechargeable battery

BACKWARDS COMPATIBILITY

The NXT brick is backwards compatible in the sense that it is possible to use LEGO® MINDSTORMS® Robotic Invention System motors and sensors if they are connected to the NXT brick with a converter cable. The converter cable does not transform any of the signals but it does need to be connected to the correct pins of the input and output ports.

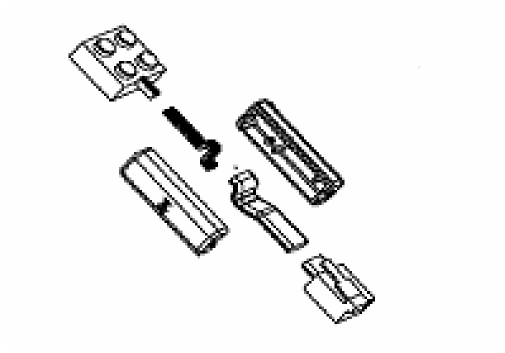


Figure 14: LEGO MINDSTORMS NXT converter cable

On the NXT's display (within the view menu), older sensors and motors have icons proceeded by an "*" icon.

Within the converter cable, pins 1 and 2 in the RJ12 connector are connected to the two connection points of the LEGO® 2x2 el-plate connector. When pins 1 and 2 are used, the converter cable can be used for both motors and sensors.

LINKS

Visit these websites for more detailed documentation on the respective components and protocols:

- www.at91.com
- www.atmel.com
- http://www.standardics.philips.com/literature/books/i2c/pdf/i2c.bus.specification.pdf
- www.P-NET.org
- www.ultrachip.com

APPENDIX

1. LEGO® MINDSTORMS® NXT hardware schematic
2. LEGO® MINDSTORMS® NXT hardware schematic
3. LEGO® MINDSTORMS® NXT Ultrasonic Sensor hardware schematic
4. LEGO® MINDSTORMS® NXT Light Sensor hardware schematic
5. LEGO® MINDSTORMS® NXT Sound Sensor hardware schematic
6. LEGO® MINDSTORMS® NXT Touch Sensor hardware schematic
7. LEGO® MINDSTORMS® NXT Ultrasonic Sensor I^2C communication protocol
8. LEGO MINDSTORMS NXT ARM7 Bluetooth® interface specification