LEGO MINDSTORMS EV3

Hardware Developer Kit

LEGO® MINDSTORMS® EV3 Hardware Developer Kit

Table of Contents

1 Hardware specifications for MINDSTORMS EV3 Programmable brick

LEGO MINDSTORMS EV3 programmable brick is the central processing unit within the new LEGO MINDSTORMS platform. The programmable brick consists of various advanced electronics to enable its wide range of functionalities.
Please reference the appendix list for the hardware schematics on the individual elements.

Below list is a summary of the hardware specifications for the EV3 Programmable brick.

Main processor:	32-bit ARM9 processor, Texas Instrument AM1808 - 300 MHz - OS: LINUX
Memory:	64 MB DDR RAM 16 MB FLASH 256 KB EEPROM
Micro SD-Card interface	SDHC standard, 2 – 32 GB
Bluetooth wireless communication	Bluetooth V2.1 EDR, Panasonic PAN1325 module - Texas Instrument CC2550 chip - BlueZ Bluetooth stack - Primary usage, Serial Port Profile (SPP)
USB 2.0 Communication, Client interface	High speed port (480 MBit/s)
USB 1.1 Communication, Host interface	Full speed port (12 MBit/s)

4 input ports	6 wire interface supporting both digital and analog interface
	- Analog input 0 – 5 volt
	- Support Auto-ID for external devices
	  - UART communication
	    o Up to 460 Kbit/s (Port 1 and 2)
	    o Up to 230 Kbit/s (Port 3 and 4)

4 output ports	6 wire interface supporting input from motor encoders
Display	178x128 pixel black & white dot-matrix display - Viewing area: 29.9 x 41.1 mm
Loudspeaker	Diameter, 23 mm
6 Buttons User interface	Surrounding UI light
Power source	6 AA batteries - Alkaline batteries are recommended - Rechargeable Lithium Ion battery, 2000 mAH
Connector	6-wire industry-standard connector, RJ-12 Right side adjustment

2 EV3 Technical Overview

This section provides a graphical representation of the main functional areas within EV3 Programmable P-brick. The figure only includes high-level units within the EV3. For detailed information on how the individual elements are connected, see the hardware schematic in Appendix 1.

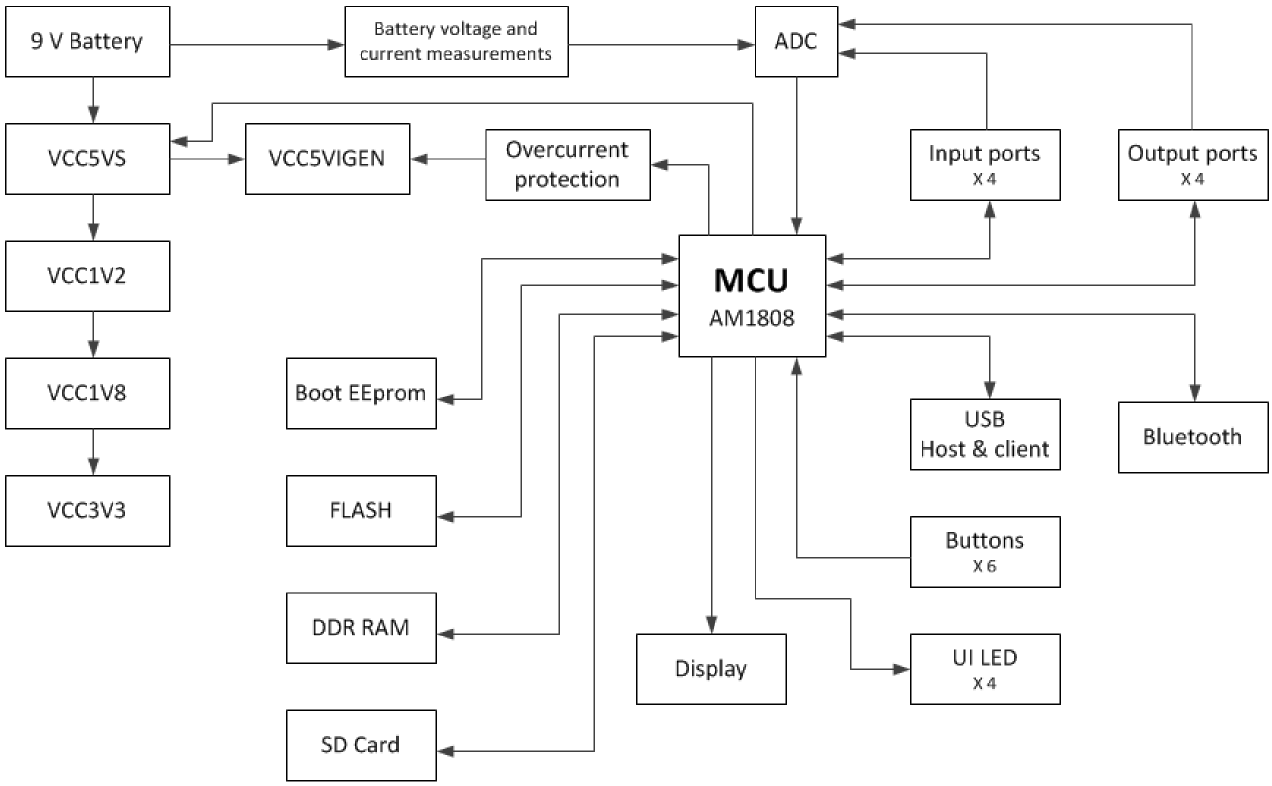


Figure 1: High level block diagram for EV3 Programmable P-brick.

As can be seen in above figure, the heart of the system is the ARM9 controller which handles all main functionality.

3 Output ports

The LEGO MINDSTORMS EV3 has four output ports used for controlling actuators connected to the EV3 programmable brick.

The main functionality for the 4 individual output ports is to enable control of external actuators. A 6 wire interface is implemented to enable external devices to send back data to the EV3 P-Brick without having to use up an input port as well.

Below figure shows the schematic details behind port A of the P-Brick.

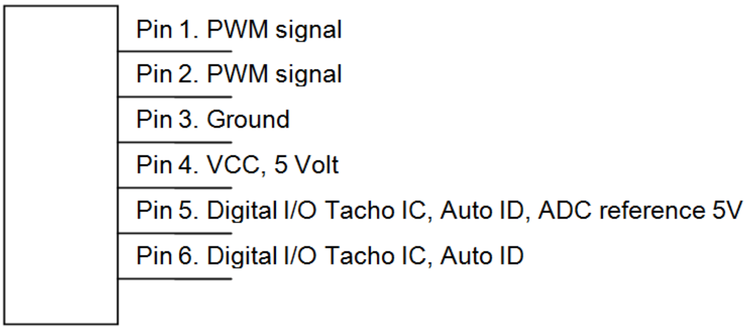


Figure 2: Pin configuration implemented in output ports for MINDSTORMS EV3 P-brick

Pin1 and Pin2 are PWM signals for controlling external actuators. These signals are controlled by an internal motor driver which can supply 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 signal (Enabling breaking or floating the actuator). The motor driver has thermal protection built-in, which means that if too much power is continually drawn from the brick or the output is short circuit, the motor driver will automatically adjust the output current.
Pin 5 and 6 serves as input ports. The pins is used for auto detection, see below section or further details, while also serving at input for the tacho signal from the MINDSTORMS EV3 motors.
Besides functioning as a general I/O pin, pin 5 is also connected to a 10-bit AD converter which enables reading an analog value if needed.

3.1 ID system on output ports

The LEGO MINDSTORMS EV3 platform supports auto detection of external elements. Detecting external unit on the output ports is done in a sequence where pin 6 is been controlled by using the motor driver while a value is being read on pin 5.
Current version of the firmware is able to detect the following elements: MINDSTORMS EV3 Large motor, MINDSTORMS EV3 Medium motor, MINDSTORMS NXT motor, no device connected. All other combination will result in an error:

Auto ID detection approach on output connection	
LEGO MINDSTORMS EV3 Large motor	Pin 6 high and pin 5 read at approximately 2.0 V Pin 6 low and pin 5 read at approximately 1.6 V
LEGO MINDSTORMS EV3 Medium motor	Pin 6 high and pin 5 read at approximately 0.45 V Pin 6 low and pin 5 read at approximately 0.25 V
LEGO MINDSTORMS NXT motor	Pin 6 high and pin 5 read at approximately 0.45 V Pin 6 low and pin 5 read at approximately 0.25 V
Other combination	Current FW will not auto detect any element

4 Input ports

The LEGO MINDSTORMS EV3 has four input ports used for controlling sensors connected to the EV3 programmable brick.

The main functionality for the 4 individual input ports is to enable the system to react to its environment through feedback from sensors. A 6 wire interface is implemented to enable external devices to send back data to the EV3 P-Brick in various ways. The system support data feedback using one of the following approaches: Analog values, I2C communication, UART communication.

Below figure shows the schematic details behind port 1 of the P-Brick.

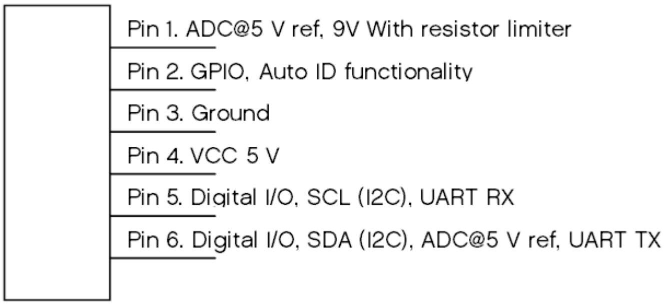


Figure 3: Pin configuration implemented in input ports for MINDSTORMS EV3 P-brick

As can be seen in above figure the individual pins are used for various functionalities depending on the requested functionality/context.
Pin 1 support reading analog values or supporting sensors that require a higher voltage output level, as maximum of 16 mA can be sourced through pin 1. Pin 2 is used during the auto ID functionality. When the system automatically has detected the sensor type being attached the system configures the pin 5 and 6 to the appropriate functionality.

4.1 I2C communication

The EV3 platform supports the same level of I2C communication as MINDSTORMS NXT. This means that system support a maximum of 9600 bit/s and maximum 32 bytes communication buffers. All I2C communication is running within software drivers. External devices are required to include pull-up resistors on both pin 5 and 6 on 82K ohm. If multiple sensors are used on the same bus the total impedance needs to be adjusted accordingly.

4.2 UART communication

To enable bi-directional and faster communication to external devices the EV3 platform supports UART communication on its input ports. The UART communication functions as asynchronous communication, supporting from 2400 bit/s to 460 Kbit/s on port 1 & 2, while ports 3 & 4 supports up to 230 Kbit/s. The UART communication uses 1 stop bit, 8 data bits and 1 stop bit.
To establish UART communication between the EV3 Programmable brick and an external device a specific communication sequence should be followed.

4.3 Auto ID system on Input ports

The LEGO MINDSTORMS EV3 platform supports auto detection of external elements. Detecting external unit on the input ports is done in a sequence where pin 1, 2, 5 and 6 is being used in a special sequence to identify which external element are connected.

Initial an input port is identified as "Open port" with the following configuration:

- All I/O pins within a sensor ports is set as input ports
    - Value at pin 1 is greater than 4800 mV, (AD measurement)
    - Level at pin 2 is high, (Digital I/O read)
    - Level at pin 5 is high, (Digital I/O read)
    - Level at pin 6 is low, (Digital I/O read)
    - Value at pin 6 is lower than 150 mV, (AD measurement)

To enable stability in recognizing the attached external elements the system requires the connection to be stable in 350 mS.
Current version of the firmware is able to detect the following elements: MINDSTORMS EV3 Touch sensor, MINDSTORMS EV3 Temperature sensor, MINDSTORMS EV3 digital sensors which currently consists of MINDSTORMS EV3 Color sensor, MINDSTORMS EV3 Gyro sensor, MINDSTORMS EV3 Ultrasonic sensor, MINDSTORMS EV3 IR sensor and no device connected.
The system is also able to detect the following MINDSTORMS NXT Sensors: MINDSTORMS NXT Touch sensor, MINDSTORMS NXT Light sensor, MINDSTORMS NXT Color sensor, MINDSTORMS NXT Temperature sensor and MINDSTORMS NXT Sound sensor. All other combination will result in an error:

The detection sequence will use the following sequence:

Auto ID detection approach on input connection	
LEGO MINDSTORMS I2C device	Pin 2 is low Pin 5 is high Pin 6 is high Further validation requires communication
LEGO MINDSTORMS NXT Light sensor	Pin 2 is low Pin 5 is low
LEGO MINDSTORMS NXT color sensor	Pin 2 is low Pin 1 is lower than 100 mV
LEGO MINDSTORMS NXT touch sensor	Pin 2 is low Pin 1 is higher than 4800 mV
LEGO MINDSTORMS NXT touch sensor	Pin 2 is low Pin 1 is between 850 mV – 950 mV
LEGO MINDSTORMS NXT touch sensor	Pin 2 is low None of above scenarios are active
LEGO MINDSTORMS EV3 Digital sensor	Pin 2 is high Pin 1 is lower than 100 mV
LEGO MINDSTORMS EV3 simple sensor option	Pin 2 is high Pin 1 is between 100 mV – 3100 mV
LEGO MINDSTORMS NXT I2C temperature sensor	Pin 2 is high Pin 1 is higher than 4800 mV Pin 6 is high

An input port will not be active if the system cannot detect it as a valid device according to above list.

4.4 Digital sensor communication

The input supports bi-directional UART communication to enable fast and reliable communication between the EV3 Programmable brick and external devices. The EV3 P-Brick support UART communications from 2400 bits/s – 460 Kbit/s on port 1 & 2, while port 3 & 4 supports a maximum communication speed of 230 Kbit/s. The UART communication is using 1 START bit, 8 data bit, No parity and 1 STOP bit. Standard deviation of 3 percent on the communication is accepted. Larger deviation can result in communication problems.

The communication architecture implemented for external devices requires the devices to follow specific guidelines. Following this guideline enables many more possibilities for the external devices when interfacing to the EV3 Programmable P-brick.
The new architecture requires the external devices to automatically send back data to the EV3 Programmable P-brick. The EV3 Programmable P-brick is able to handle new sensor data from the external device at maximum speed of 1 mS.
In the following the initialization sequence for the EV3 Color sensor will be documented to help exemplify who the communication sequence flows:

The external devices start by transmitting back all it configuration values to enable the EV3 Programmable brick to update the system accordingly.
Configuration data includes:

- Start by keeping UART Tx-pin low for minimum 500 mS. (This will indicate a break condition).
- Device type
- Number of mode the device supports
- Supported UART communication speed
- It then starts communicating the individual modes supported. The default mode for the device needs to be communicated within mode 0 which also needs to be the last transmitted mode. Start with transmitting the highest mode and end with transmitting mode 0. Good practice is to add 10mS delay between each mode (IE. Transmit all relevant data for the mode and then wait 10 mS before starting in the next mode).
    - Mode name
    - Raw value range
    - SI value range
    - Symbol
    - Data format (This data ends the current mode information)
- After having sent all relevant mode data the external device should send an ACK and start waiting on an ACK form the EV3 programmable brick. If no ACK is received within 80 mS the external device should reset as not receiving an ACK from the EV3 programmable brick means that the EV3 programmable brick has encounter an error during the communication. By resetting it should automatically start sending its relevant data again.
- When an ACK is received the EV3 programmable brick has acknowledge the data and will switch to the requested communication speed and are ready to start receiving data from mode zero.
- Please reference the protocol documentation towards how data needs to be packages when sending them back to the EV3 programmable brick. If the data is not packages correctly the EV3 programmable brick will not be able to handle the data correctly.
- The EV3 programmable brick automatically sends a NACK command to the external devices every 300 mS. The MINDSTORS EV3 sensors uses this command to reset the internal watchdog timer within the sensor. This enables the EV3 programmable brick and the sensors to continuously be in sync. The MINDSTORMS EV3 sensors re-transmits the last valid data

value upon this request. If no feedback values are received within 5 NACK request the NACK command is not send from EV3 programmable brick, this will result in the watchdog not being reset. The EV3 programmable brick will expect a watchdog timeout within the external device which will result in a reset of the external element.

5 User Interface

The user interface within MINDSTORMS EV3 consists of a display, a button interface plus led's indicating current state of the robot.

5.1 Display

The display within the MINDSTORMS EV3 Programmable brick consist of a graphics black & white dot-matrix display at 178x100 pixels, with a viewing area of 41.1 mm (W) x 29.9 mm (H).
The LCD-controller used for controlling the display is an ST7586S, see the links for detailed datasheets for the display LCD-controller.

LEGO MINDSTORMS EV3 uses a standard SPI interface in LINUX, running at 10 MHz burst. The display driver includes two full memory maps which enables swapping between display content quickly.

The pixels within the display is allocated as follows:

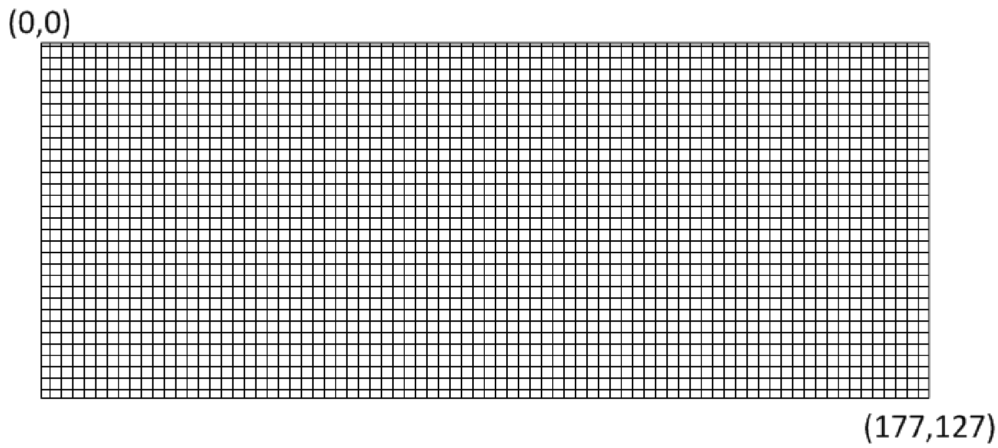


Figure 4 : Illustrating how the display for EV3 P-brick is divided into segments

Technical specification for the display:

Format	: 178x128 Dots
Dot size	: 0.195 mm (W) x 0.195 mm (H)
Dot Pitch	: 0.21 mm (W) x 0.21 mm (H)
LCD mode	: FSTN, Positive Reflective Mode / Grey
Viewing direction	: 6 o'clock
Driving scheme	: 1/128 Duty cycle, 1/12 Bias
Communication interface	: 4-line SPI interface
Power supply voltage (V_{DD})	: 3.0V
LCD driving voltage (VLCD):	13.2V

5.2 6 button user interface

The 6 button is implemented as a simple 4 direction joystick including a center accept button. A back / stop button is placed at the side. All buttons is simple individual switches.

5.3 UI light

The UI light surrounds the 4 directions button interface and are primarily used for indicating the state of the EV3 Programmable brick. It is also possible to control these lights from within a running program. Controlling the light within a program or app. will take precedence, therefore below list is only valid when the UI light is not controlled from within a running program and app.

UI light meaning	
Green	EV3 programmable brick Ready and responsive
Green blinking	Program or App executing
Orange	Warning, EV3 programmable brick busy / unresponsive
Orange blinking	Warning while executing program or App,
Red	EV3 programmable brick booting
Red blinking	Busy

6 Sound

The EV3 P-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 ARM9 controller. The filters introduced before the amplifier will reduce the oversampling noise in the signal.

The sound driver (TPA6211A1) is a mono differential sound amplifier chip from Texas Instrument that can deliver up to 3.1 Watt. For detailed information about the sound amplifier, please reference the data-sheet from Texas Instrument.

The loudspeaker in the EV3 P-brick is a 16 ohm speaker with a diameter of 23 mm. The table below shows the current and power consumption when sounds are played at two frequencies.

Frequency	Current	Power
440 Hz	174 mA	484 mW
4 KHz	163 mA	425 mW

Sound current consumption

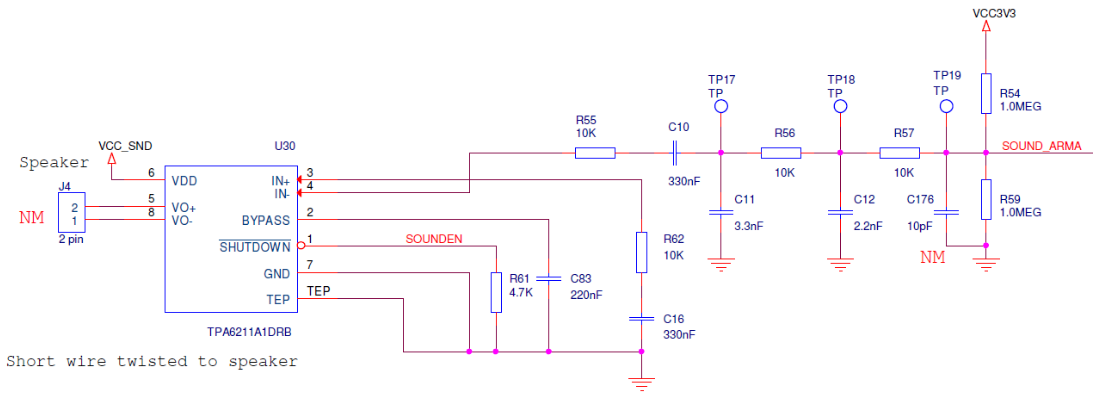


Figure 5: Schematic for sound output within the EV3 P-brick

7 Wireless communication

The EV3 P-Brick support two wireless communication possibilities. Bluetooth communication is supported through built in functionality while WiFi communication requires connecting an external WiFi dongle.

7.1 Bluetooth®

MINDSTORMS EV3 Programmable brick enables wireless communication using Bluetooth communication, using a TI 2640 Bluetooth chip, controlled through BlueZ. The EV3 brick can be connected to 7 other devices at the same time, but can only communicate with on device at a time. The functionality has been implemented using the Serial Port Profile (SPP) which can be seen as wireless serial port.
The EV3 brick can communicate with other Bluetooth devices which support the Serial port Profile, and which can be programmed to communicate using the EV3 Communication protocol. This enables sending or queering data from the EV3 brick wirelessly.

Thought Bluetooth communication it is possible to send project files between EV3 bricks or other Bluetooth devices and it is possible to use the wireless communication to send and receive information between brick during program execution.
To reduce the power consumption in relation to Bluetooth it has been implemented as a Bluetooth Class II device which means that it can communicate up to a distance of approximately 10 meters.

7.2 Bluetooth functionality within the EV3 Brick

Bluetooth functionality within the intelligent EV3 brick is a master/slave communication channel which means that one EV3 in the network needs to function as a master while the others are slave EV3 bricks. The figure below shows how the master EV3 brick communicates with the slave bricks.

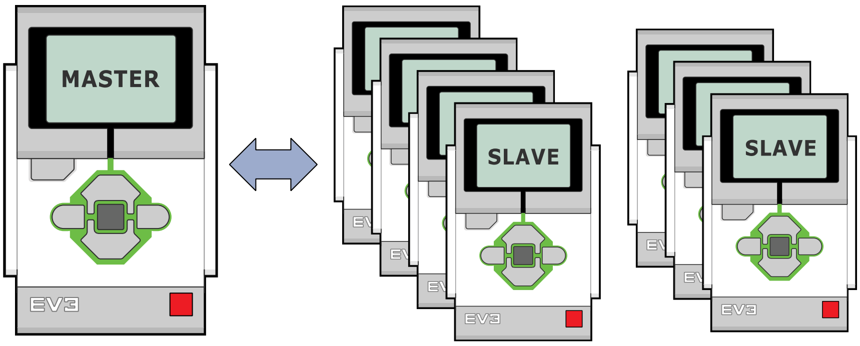


Figure 6 : Illustration of master and slave communication using Bluetooth for EV3 P-brick

As can be seen on the figure above the master EV3 is able to be connected to seven other Bluetooth devices at the same time. But the master EV3 can only communicate with one of the slave unit at a given time, meaning that if it is communicating with “EV3 slave 1” and “EV3 slave 3” starts to send data to the master EV3, the master EV3 will not evaluate the receive data before it automatically switches to evaluate data from “EV3 slave 3”. The EV3 do have a small buffer for data that it has received from “EV3 slave 3” while it was communicating with “EV3 slave 1”, but if the communication gets to intense, data can be lost.

Setting up one EV3 brick to functioning both as master and a slave, called scatter net is not supported as this could influence the communication performance unintentional and could result in loosing data. Therefore this functionality has been disables within the standard firmware at this point of time.

Within the standard firmware connection to other Bluetooth device is utilized through device names. This means that if one EV3 brick is connected to a second EV3 Brick named LEGO, then all communication targeting the second EV3 Brick from the master EV3 brick should reference brick called LEGO from within the software application.

7.3 WiFi communication

As mentioned the EV3 brick supports WiFi communication with an external WiFi dongle attached. Currently the EV3 brick support the following WiFi dongle:

- NETGEAR N150 Wireless USB Adapter, WNA1100

The WiFi connection enables having a wireless communication between the MINDSTORMS EV3 Software and the EV3 brick. The system supports WiFi communicating using either no security settings or using WPA2 security settings.

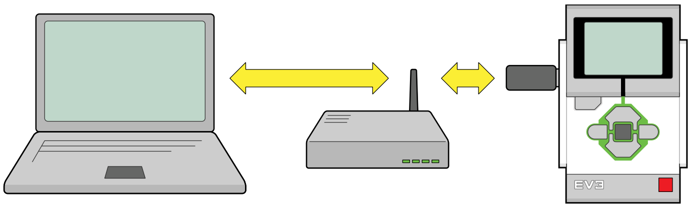


Figure 7 : Illustration of WiFi communication between a PC, Wireless router and EV3 P-brick

Setting up WiFi communication is possible through either the brick user interface or using the LEGO MINDSTORMS EV3 software. Please reference the LEGO MINDSTORMS EV3 manual.

Enabling WiFi can be done within the UI of the EV3 brick, which loads and starts the needed drivers. The EV3 Brick automatically starts sending out a beacon signal including its detailed technical information. Below table includes the technical details within the beacon identification signal:

WiFi Beacon signal transmitted every 5 second The Beacon signal is transmitted as UDP packages on port 3015	
Serial Number:	"*Unique Bluetooth address of the EV3 Brick*"
Port:	5555 (*The used and supported port when using TCP communication*)
Name:	"*EV3 brick name*"
Protocol:	EV3

When a host device wants to establish a WiFi connection with a specific EV3 Brick, both devices needs to be connected to the same network. Upon receiving above UDP packages the host device must first send back an acknowledge packages (A single byte) to the source port of the UDP packages. This will enable the EV3 Brick to receive an unlock string through TCP. Following the UDP packages, a TCP unlock string needs to be send to port 5555, including the string "ET /target?sn=".
When the EV3 Brick accepts the connection it will return with a TCP packages including the string "Accept:EV340". Now a WiFi connection is established between the host device and the EV3 brick.

With a valid WiFi connection it is possible to send both system and direct commands to the EV3 Brick, please reference the communication protocol for further details on the protocols.

8 Daisy chain functionality

The EV3 brick supports daisy chaining up to 4 EV3 bricks using USB. This enables the master EV3 brick to control 16 output ports and 16 input ports within one program. The EV3 brick should be connected as shown below where the master EV3 brick is the one with the USB client port free for connecting to the host computer.

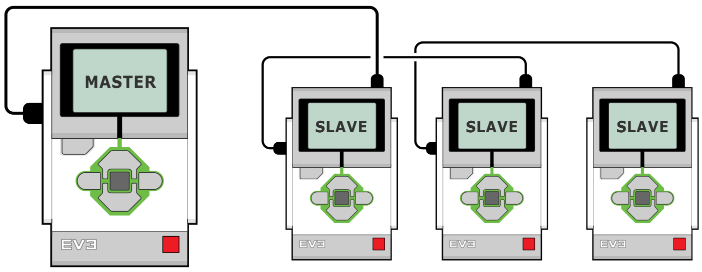


Figure 8 : Illustration of 4 EV3 P-brick's daisy chained using USB communication

The three "Slave" devices will not be executing any individual program but instead be executing individual designated commands. The systems handles automatically transmitting data up and down the 12 Mbit USB chain. The duration between updating all motor and sensor values within the master device depends on the configuration of the daisy chain. Above figure shows the maximum chain configuration and in such a scenario all 16 motors and 16 sensor values will be updated approximately every 100 mSec for the master device.

Daisy commands are routed as commands enveloped in the normal EV3 protocol for direct commands. This encoding is used down through the Daisy Chain except for the last layer (i.e. between layer 3 and 4 if the chain is build by 4 EV3 P-bricks). The communication between the two last brick is relayed as a "normal" direct command. This allows the receiving brick to receive the Daisy Chained commands without any special handling. If only 2 bricks are connected via Daisy Chain the command is always send to the second brick as a "*DIR_CMD_REPLY_WITH_BUSY*" or "*DIR_CMD_NO_REPLY_WITH_BUSY*".

Common DAISY commands:

```
#define      DAISY_COMMAND_REPLY          0x0A    // Daisy command, reply required
#define      DAISY_COMMAND_NO_REPLY       0x8A    // Daisy command, reply not required
```

DAISY commands with motor busy:

```
#define      DIR_CMD_REPLY_WITH_BUSY      0x0F    // Direct command, reply required
#define      DIR_CMD_NO_REPLY_WITH_BUSY   0x8F    // Direct command, reply not required
```

Daisy Chain Protocol Bytes:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8

```
Byte 0 – 1:         Command size, Little Endian

Byte 2 – 3:         Message counter, Little Endian

Byte 4:             Command type. See defines above

Byte 5:             Daisy sub Command. See defines below:
```

Sub-commands:

```
#define    DAISY_CHAIN_DOWNSTREAM                  0xA0 // Write down into the branch of DaisyChained Bricks
#define    DAISY_CHAIN_UPSTREAM                    0xA1 // Data packet sent upstream from Port-Expanding
                                                        // Bricks
#define    DAISY_CHAIN_INFO                        0xA2 // Sensor information packet (max. 54 bytes)
#define    DAISY_UNLOCK_SLAVE                      0xA3 // Ask the slave to start pushing data upstream
#define    DAISY_SET_TYPE                          0xA4 // Set mode/type downstream (no reply needed –
                                                        // status via array)
#define    DAISY_CHAIN_DOWNSTREAM_WITH_BUSY        0XA5 // Payload also includes Magic Cookies (4 pcs)
```

```
Byte 6:             Destination Layer Counter (Counts down)

Byte 7 - 8:         Var alloc (encoded as normal EV3)

Byte 9 - ?:         The normal payload. E.g. bytecode(s) with parameters
```

Motor Command examples:

Below is a command *opOUTPUT_STEP_SPEED* recorded between layer 1 and 2. The originally command sent from the Master-brick is *opOUTPUT_STEP_SPEED*, Layer 2, Motor B. The command is sent as a *DIR_CMD_NO_REPLY_WITH_BUSY* 0x8F because layer 2 is the final and receiving layer.

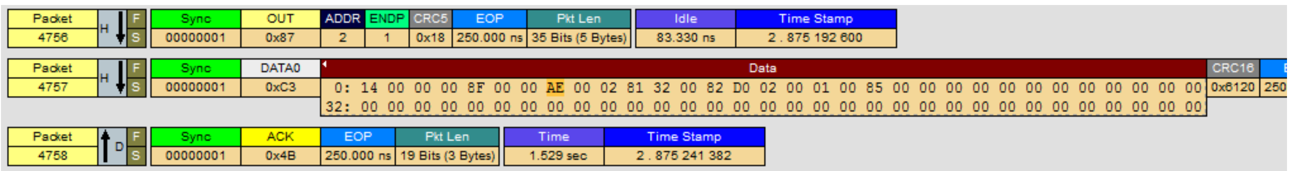


Figure 9 : Low-level USB communication used within Daisy chain functionality

Below is a command *opOUTPUT_STEP_SPEED* recorded between layer 1 and 2. The originally command sent from the Master-brick is *opOUTPUT_STEP_SPEED*, Layer 3, Motor B. The command is sent as a *DAISY_COMMAND_NO_REPLY* 0x8A, sub-command *DAISY_CHAIN_DOWNSTREAM_WITH_BUSY* 0XA5 as the destination layer is one brick further down (See the 1 in byte 6).

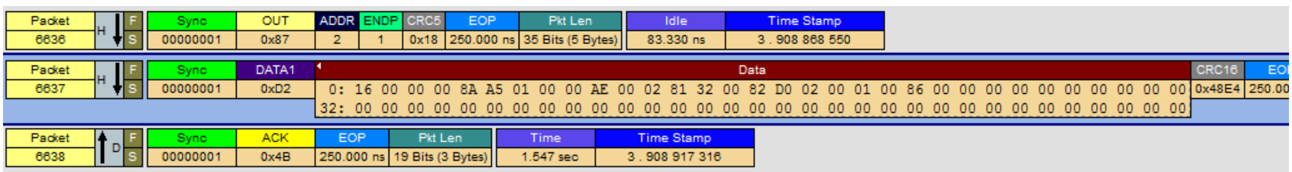


Figure 10 : Low-level USB communication used within Daisy chain functionality

Below is a command *opOUTPUT_STEP_SPEED* recorded between layer 2 and 3. The originally command sent from the Master-brick is *opOUTPUT_STEP_SPEED*, Layer 3, Motor B. As you can see, the command is now sent as a Direct Command - the Layer 3 is the receiving layer.

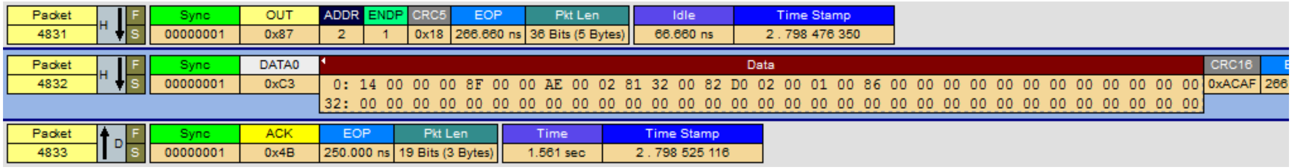


Figure 11 : Low-level USB communication used within Daisy chain functionality

Daisy Chain Protocol Response Bytes:

```
#define     DAISY_REPLY                           0x08 // Daisy command reply
#define     DAISY_REPLY_ERROR                     0x09 // Daisy command reply error
```

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8

```
Byte 0 – 1:        Reply size, Little Endian

Byte 2 – 3:        Message counter, Little Endian

Byte 4:            Reply type. See defines above

Byte 5:            Daisy sub Command. See defines above

Byte 6:            Layer relative to the receiving brick

Byte 7:            Sensor address

Byte 8:            Sensor status

Byte 9:            Sensor type

Byte 10:           Sensor mode

Byte 11 - 43:      Sensor data

Byte 44 - 59:      Busyflags. Layer 1 motor A, Layer 1 motor B………. Layer 4 motor D
```

The busyflags are composed of the following bits:

Bit 7: The busy bit. 1 = BUSY, 0 = READY (done)
Bit 6: Msb of the Magic Cookie. Incremented by 1 at new command.
Bit 5: -
Bit 4: -
Bit 3: -
Bit 2: Lsb of the Magic Cookie
Bit 1: Msb of the Layer address (0-3)
Bit 0: Lsb of the Layer address

The Magic Cookie is used for detecting the lifecycle of the BUSY-flag. I.e. the BUSY flag upstream can only be cleared by a Magic Cookie at same ages or newer (higher number). This ensures that a downstream command with BUSY signal not accidental will be cleared by an older READY signal. The

BUSY-flags are store in the same array as the device data. This means: All upstream bricks not only knows their own sensorstatus but also the status of their downstream partners. Remember each brick has 8 sensor. The motortachos are separate sensors with address set to 0x10 = A, 0x11 = B etc.

Response/Sensor-update packet with the BUSY-flag set for motor B, layer 3. The Magic Cookie is composed by the BUSY-flag (bit 7) = 0x01, the cookie-no. (bit 2-6) = 0x01 and the Layer (bit 0,1) = 0x10 => layer 3. Remark the 0x05 an older BUSY-flag with reset busy, cookie-no. set to 1 and layer 2. This update packet is for the layer 2 (0x01), sensor 4 (0x03).

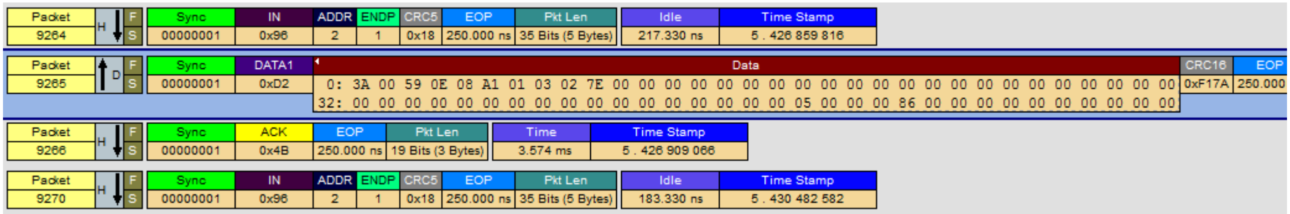


Figure 12 : Low-level USB communication used within Daisy chain functionality

Response/Sensor-update packet with the BUSY-flag reset again for motor B, layer 3. The Magic Cookie is composed by the reset BUSY-flag (bit 7) = 0x00, the cookie-no. (bit 2-6) = 0x01 and the Layer (bit 0,1) = 0x10 => layer 3. This update packet is for the layer 2 (0x01), sensor 16 (= 5) – i.e. Tacho A (offset 0x10).

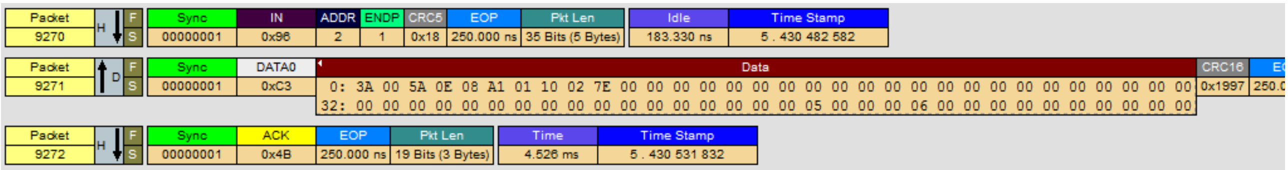


Figure 13 : Low-level USB communication used within Daisy chain functionality

The input modules in the Daisy Chained bricks will also “push” their sensor information upwards (upstream) when a new sensor is detected or from boot. These information packets are encoded as DAISY_REPLY 0x08 and DAISY_CHAIN_INFO 0xA2. The information is used by the Master (position 1) brick to get capabilities of a given sensor and to decode/format the received data. See an example of the information packets for the 5 modes of a Gyro Sensor enumeration (plugin):

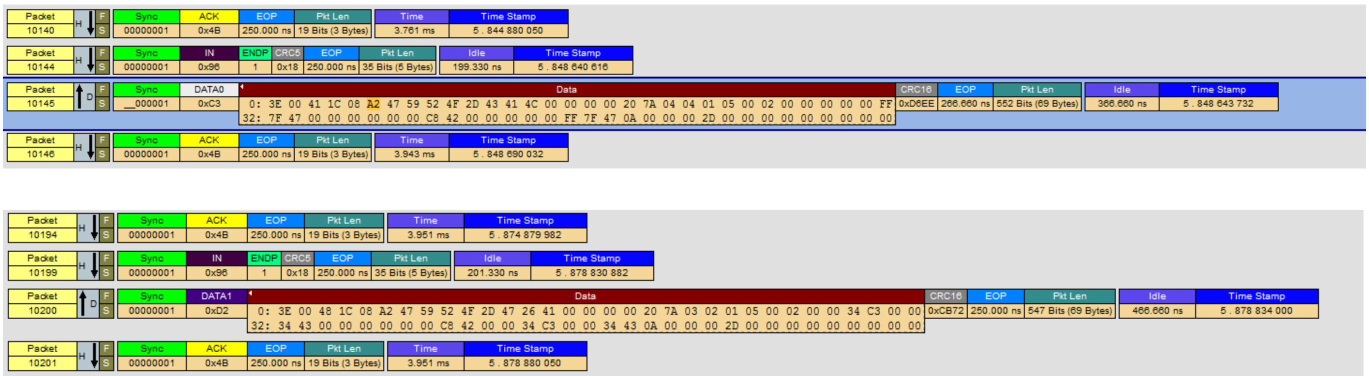


Figure 14 : Low-level USB communication used within Daisy chain functionality

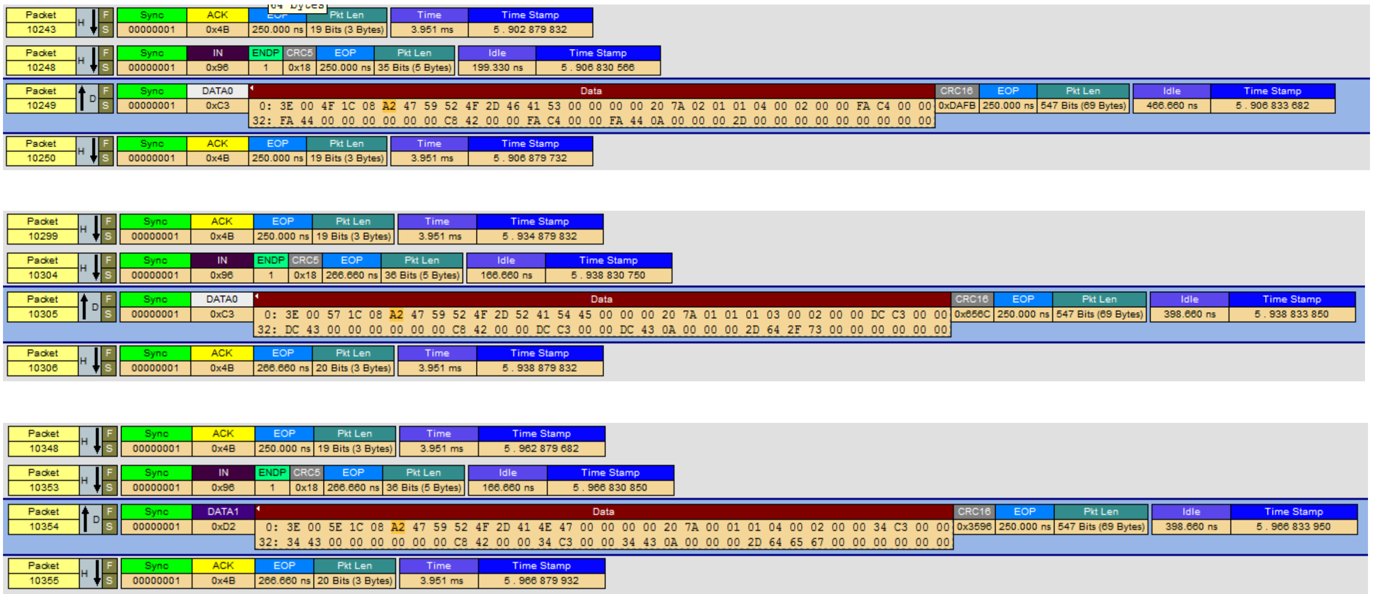


Figure 15: Low-level USB communication used within Daisy chain functionality

9 SD-card interface

The EV3 Brick includes a micro SD-card interface which supports cards with capacities up to 32 GB, SDHC® standard.
The SD memory card enables increasing the available memory for user data plus it enables booting from an alternative destination. Enable being able to boot from the SD-Card requires building and placing the images correctly on the SD-Card.

Creating a SD-Card which can be used for booting can be done by following below steps:

1. Using a PC running Linux console with SD card reader, create a plain text file "*fdisk.cmd*" containing:

```
p
d
1
d
2
p
n
p
1
1
+50M
t
b
n
p
2

+200M
p
w
```

2. Open Linux console and type:
   a. *cat /proc/partitions*
   b. Insert SD card in reader and type again:
   c. *cat /proc/partitions*
3. Identify the difference in above two commands. The difference equals the name of the SD card - assumed "sdx"
   a. CARE MUST BE TAKEN TO USE THE RIGHT NAME !
4. **Un-mount:** Un-mount all partitions by typing (substitute "sdx" with the right name):
   a. *sudo umount /dev/sdx1*
   b. *sudo umount /dev/sdx2*
5. **Partitioning:** Type following (substitute "sdx" with the right name):
   a. *sudo fdisk /dev/sdx < fdisk.cmd*
6. **Formatting:** Type (substitute "sdx" with the right name):
   a. *sudo mkfs.msdos -n LMS2012 /dev/sdx1*
   b. *sudo mkfs.ext3 -L LMS2012_EXT /dev/sdx2*
7. ***Remove***: Use the "Safely Remove Drive" and remove SD card or type:
   a. *sync*

Following above sequence creates an SD-card which afterwards can be used for booting the EV3 Brick when a correct firmware version has been installed correctly to the SD-Card.

10 Power management

Power within the EV3 P-brick comes from 6 AA batteries or a rechargeable Lithium ion battery. The EV3 Rechargeable lithium ion battery is charged through the LEGO® transformer.
Power management within the EV3 P-brick consists of multiple switching regulations which are tightly controlled and interlinked in order to boot the electronic circuit correctly.

To protect the EV3 P-brick from short circuit, 3 poly switches are included, one for each of the two motor drivers and one for the rest of the circuit. Each poly switch has a hold current at approximately 1.1 A and will be triggered at approximately 2.2 A.

<u>Current consumption measurement:</u>

The effects are based on a battery of 9 Volt.

Supply voltage	Current		Effect (Battery = 9 Volt)	
	Max [mA]	Normal [mA]	Max [mW]	Normal [mW]
3 Motors without load 4 sensor No communication				
9 Volt	N.A.	155	N.A.	1395
5 Volt	N.A.	308	N.A.	1540
3.3 Volt	N.A.	404	N.A.	1333
3 Motors without load 4 sensor With BT & WiFi active				
9 Volt	N.A.	242	N.A.	2178
5 Volt	N.A.	395	N.A.	1975
3.3 Volt	N.A.	477	N.A.	1574
Load on motors				
9 Volt	1760	640	15840	5760
5 Volt	2190	592	10950	2960
3.3 Volt	“Low bat”	“Low bat”	“Low bat”	“Low bat”
Standby				
9 Volt	N.A.	0.024	N.A.	0.216
5 Volt	N.A.	0.012	N.A.	0.060
3.3 Volt	N.A.	0.004	N.A.	0.013

10.1 Battery testing with LEGO MINDSTORMS EV3

The EV3 P-brick performance depends on the batteries used and the load applied to the brick. The two figures shown below illustrates the performance of the EV3 P-brick while using 6 standard alkaline batteries and while using the LEGO® MINDSTORMS EV3 lithium ion rechargeable battery. The load test was performed with three LEGO® MINDSTORMS EV3 motors attached, two large motors and one medium motor, to the EV3 P-Brick running at full power with load applied.

The test indicates that the EV3 P-brick performs well both when running on standard alkaline batteries and when running on the LEGO® EV3 rechargeable lithium ion batteries pack. The LEGO lithium ion battery is the only solution that allows the EV3 P-brick to be power continuously by connecting the battery to the LEGO® Transformer. If the load is more than approximately 500 mA, the power will eventually be used even though the transformer is connected.

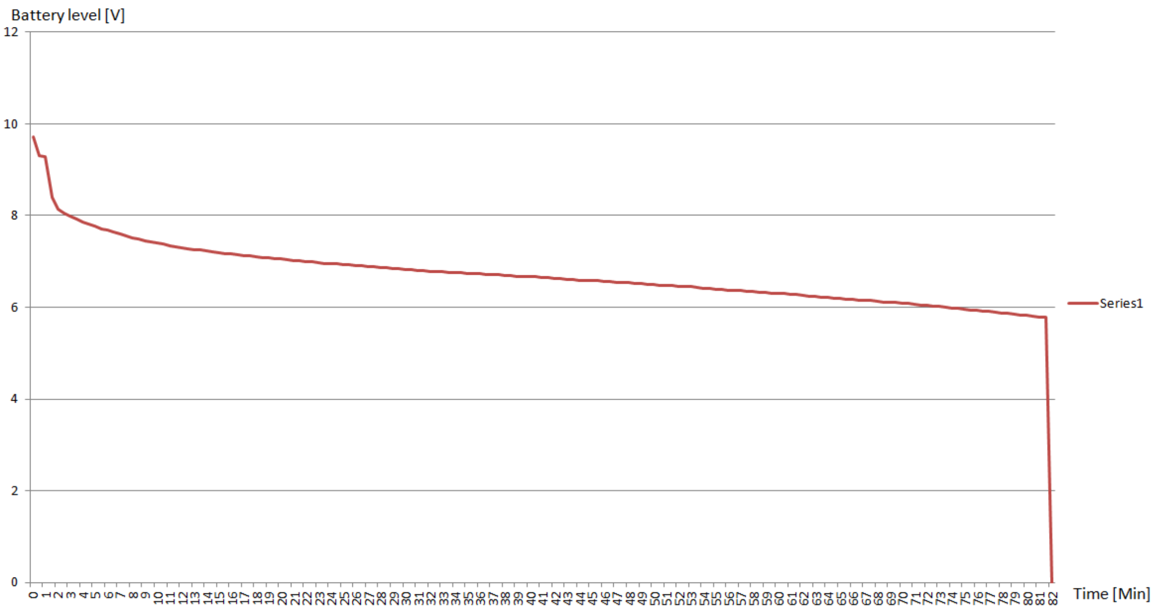


Figure 16 : Battery level over time when running on 6 standard alkaline batteries

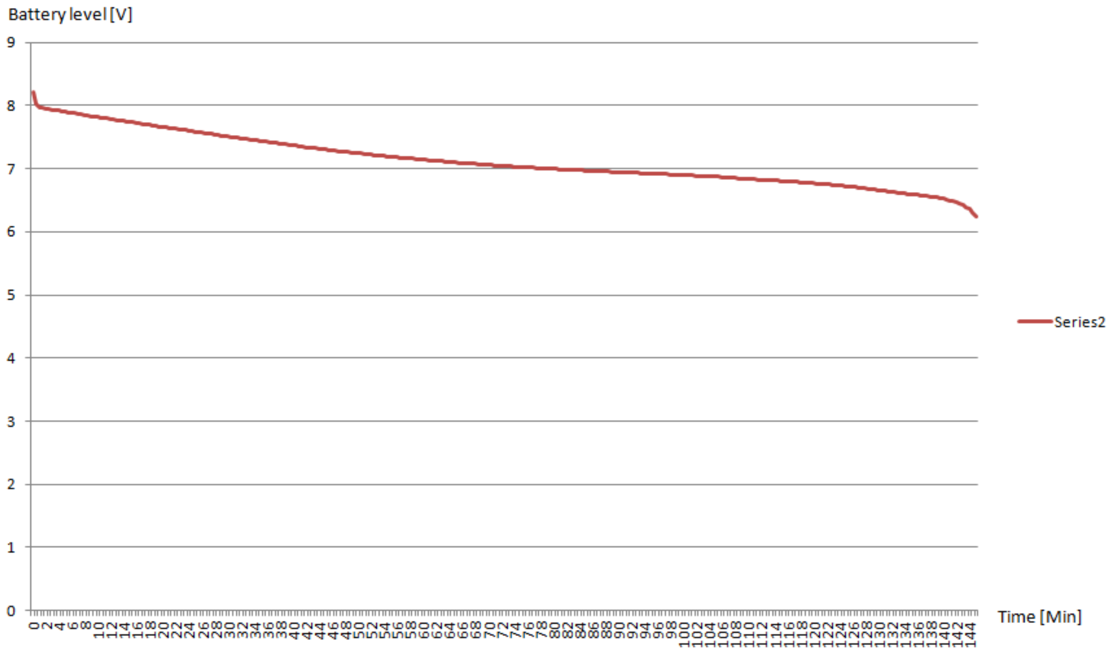


Figure 17 : Battery level over time when running on MINDSTORMS EV3 Rechargeable battery

11 Touch sensor

The touch sensor is a very simple sensor which basically enables building a robot which is able to feel. The sensor consists of a basic mechanical switch which can either be pressed or released. Given the simplicity of the sensor, this sensor is also implemented as a pure analog sensor.

The mechanical switch is connected between pin 4, GND with a 2.2 K Ohm resistor in serial and pin 6 within the 6 pole connector. The EV3 P-brick continuously reads the AD-value on pin 6 and through evaluating this value it is able to determined if the touch sensor is activated or not.

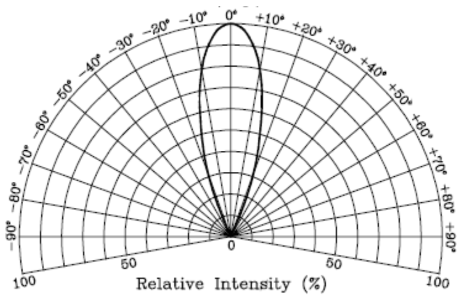
Pin 1 includes a 910 ohm resistor connected to GND, which is used as the ID for the touch sensor.

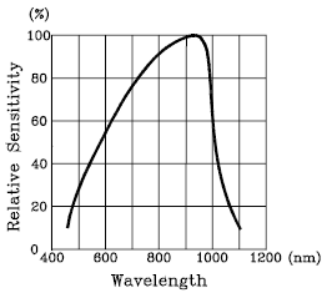
12 Color sensor

The color sensor enables three different functionalities. It can function as a light sensor measuring reflected light, measuring ambient light and measuring colors.
This sensor is implemented as a digital sensor, meaning the sensor includes a small 8-bit microcontroller which communicates with the EV3 P-brick using UART communication on pin 5 and 6.

The microcontroller handles all sensor measurements and data analysis which is then communicated back to the EV3 P-brick. The color sensor is able to return new sensor values 1000 times a second if read surface changes fast enough. The sensor is programmed to automatically return a new sensor value whenever the value changes or if the EV3 P-brick requests a latest sensor value.
When the color sensor are connected to the EV3 P-brick it automatically starts by identifying itself to the EV3 P-brick as a color sensor and it communicates the modes and value ranges it supports before it switches to the default mode which is light sensor mode.

The diagram to the right shows the color sensor angular displacement and relative sensitivity.
The relative wave length sensitive is shown in the figure below. As can be seen the sensor is most sensitive around 900 nm.





Light sensor mode:
Within this mode the sensor turns on the red led and measures the reflected light intensity. The optimal reading distance is between 0.5 and 1.5 LEGO module distance.
The sensor automatically cancels out any backlight. This is done by very quickly turning the red led off and measure the backlight which is then subtracted from the actual reflected reading. This is continuously done 1000 times/sec in order to enable 1000 sample/sec.
The output from the sensor within light sensor mode is a value between 0 – 100 where 0 equal no reflected light measured and 100 equals full light reflection.

Ambient sensor mode:
Within this mode the sensor is able to read ambient light within its surroundings. The sensor turns on the blue led in this mode to visually indicate which mode it is in.
Given that the sensor uses an algorithmic conversion table the sensor is able to measure from very dark to very bright sun light.
The output from the sensor within ambient sensor mode is a value between 0 – 100 where 0 equal darkness while 100 equals directly pointing towards sun-light.

Color sensor mode:

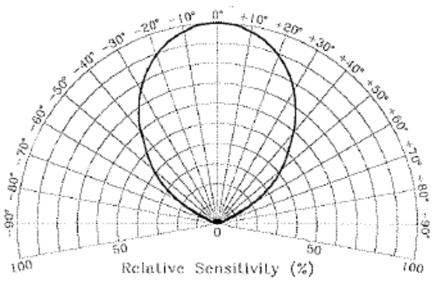
Within this mode the sensor is able to determine the color on the object in front of it. Again the optimal reading distance is between 0.5 – 1.5 LEGO module distance.
Within this mode the sensor automatically quickly reads the individual reflected light intensity for red, green, blue and the backlight. Using these values the sensor is able to determine between 7 different colors and whether no object is observable. The 7 different colors are: black, blue, green, yellow, red, white and brown.

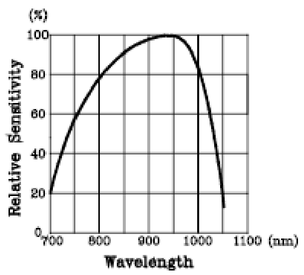
13 Infrared seeker sensor

The infrared seeker sensor enables three different functionalities. It can function as an IR proximity sensor, as an IR seeker sensor and as an IR remote control receiver.
This sensor is implemented as a digital sensor, meaning the sensor includes a small 8-bit micro-controller which communicates with the EV3 P-brick using UART communication on pin 5 and 6.

The microcontroller handles all sensor measurements and data analysis which is then communicated back to the EV3 P-brick. The infrared seeker sensor's ability to send back new values is depended which mode it operates within. In the proximity mode a new measurement are conducted every 50 mS. Within the IR seeker mode and IR remote control mode the duration between new samples are given by the IR beacon and which IR channel is uses. Using the beacon signal, the beacon remote transmit a new value every 105 – 220 mS (depending on which IR channel is used). Using remote commands, the beacon remote transmit new value every 105 - 220 mS (depending on which IR channel and command is used). The sensor is programmed to automatically return a new sensor value whenever the value changes or if the EV3 P-brick requests a latest sensor value.
When the IR seeker sensor are connected to the EV3 P-brick it automatically starts by identifying itself to the EV3 P-brick as a IR seeker sensor and it communicates the modes and value ranges it supports before it switches to the default mode which is proximity sensor mode.

The diagram to the right shows the angular displacement and relative sensitivity for the two individual IR components place each with a 30 degree outwards turn angle. These two IR components therefore overlap each other right in front of the IR sensor while also being relative more sensitive with their individual angle.
The relative wave length sensitive is shown in the figure below. As can be seen the sensor is most sensitive around 900 nm.





Proximity mode:
Within this mode the sensor is able to determine the relative distance to an object placed in front of the sensor. The sensor sends out an IR burst at different levels. Following the burst the sensor evaluates the reflected level of IR and translates this into a relative distance.

IR seeker mode:
Within this mode the sensor is able to determine the relative distance and angle towards the IR beacon, when transmitting in beacon mode. The IR sensor determines the relative distance by evaluating the amplitude of the received IR and the angle by evaluating the difference in amplitude between the two turned IR receivers in the front of the sensor.

IR remote mode:

Within this mode the sensor is able receive the LEGO Power Function IR protocol. The beacon transmit the LEGO Power Function IR protocol and depending on which button combination is pressed the sensor returns a value.

IR Beacon buttons combination	Value
No button pressed	0
Red upper	1
Red lower	2
Blue upper	3
Blue lower	4
Red upper and blue upper	5
Red upper and blue lower	6
Red lower and blue upper	7
Red lower and blue lower	8
Beacon	9
Red upper and red lower	10
Blue upper and blue lower	11

14 Ultrasonic sensor

The ultrasonic sensor enables three different modes. It enables making distance measurement up to 250 cm (100 Inches) with an accuracy of 1 cm (0.393 Inches) in either centimeters or inches. It also enables a listen mode where the ultrasonic sensor is measuring whether other ultrasonic sensor are active within its surroundings.
This sensor is implemented as a digital sensor, meaning the sensor includes a small 8-bit micro-controller which communicates with the EV3 P-brick using UART communication on pin 5 and 6.

The microcontroller handles all sensor measurements and data analysis which is then communicated back to the EV3 P-brick. The ultrasonic sensor sends back new distance measurement whenever a new distance is detected. If the distance is 250 cm (100 Inches) the duration between new sensor values will approximately be every 15 mS. With a smaller distance the ultrasonic sensor will be able to return measurement faster.

The ultrasonic sensor measure the distance to an object by sending out an ultrasonic sound signal, 12 sound burst at 40 KHz. The receive transducer is most sensitive at 40 KHz and has a beam angle of approximately 90 degree.

<u>Distance measurement, centimeter mode:</u>
Within this mode the sensor continuously measures the distance to the nearest object and returns the value with a 0.1 resolution and with an accuracy of 1 cm. Red light is continuously turn on around the transducers in the sensor.

<u>Distance measurement, inches mode:</u>
Within this mode the sensor continuously measures the distance to the nearest object and returns the value with a 0.1 resolution and with an accuracy of 0.393 inches. Red light is continuously turn on around the transducers in the sensor.

<u>Listen mode:</u>
Within this mode the sensor doesn't send out any ultrasonic sound burst itself. It only measures if any other ultrasonic sound signal can be detected. If another ultrasonic sound signal is detected the sensor will return 1 indicating that another signal have been detected. The red light around the transducers will be blinking within this mode.

15 Gyro sensor

The gyro sensor enables three different modes. It enables a standard gyro functionality mode, a relative angle measurement mode and a mode which includes both the gyro and the relative angle measurement at the same time. The gyro sensor support up to 440 degrees/sec with a accuracy at plus/minus 3 degrees within 90 degrees. All measurement are related to the Z-plan.

This sensor is implemented as a digital sensor, meaning the sensor includes a small 8-bit microcontroller which communicates with the EV3 P-brick using UART communication on pin 5 and 6.

The microcontroller handles all sensor measurements and data analysis which is then communicated back to the EV3 P-brick. The gyro sensor is able to return new sensor values 1000 times a second if the sensor reading changes fast enough. The sensor is programmed to automatically return a new sensor value whenever the value changes or if the EV3 P-brick requests a latest sensor value.

When the gyro sensor are connected to the EV3 P-brick it automatically starts by identifying itself to the EV3 P-brick as a gyro sensor and it communicates the modes and value ranges it supports before it switches to the default mode which is angle sensor mode.

Gyro angle mode:

Within this mode the sensor continuously measures and accumulates the angle speed and translates this into a relative angle. The relative angle is relating to the rotation since the last reset of the accumulated angle.

To slow a rotation or faster rotation than 440 degree/sec will result unreliable angle readings.

Gyro mode:

Within this mode the sensor continuously measures the actual rotation/second. When the sensor is not rotating this value will be zero. The fast rotation speed the sensor is able to read is 440 degree/sec.

Gyro and angle mode:

Within this mode the sensor continuously measures both the actual rotation/second and accumulates the relative angle. This mode includes the same limitation as above two modes, but enables getting both values at the same time.

16 Large and medium motor

The LEGO MINDSTORMS EV3 platform includes two different motors, a large motor and a medium motor. Both motors include tacho functionality with a resolution of one count/degree. This enables controlling the speed of the motor more accurately. This functionality is among other things enables the synchronization functionality used making the robot drive straight.

16.1 Large motor

MINDSTORMS EV3 large motor has the following characteristic:

LEGO MINDSTORMS EV3 Large motor	
9 Volt, no load	175 RPM
	60 mA
Stalled	45 Ncm
	1.8 A

16.2 Medium motor

MINDSTORMS EV3 medium motor has the following characteristic:

LEGO MINDSTORMS EV3 medium motor	
9 Volt, no load	260 RPM
	80 mA
Stalled	15 Ncm
	800 mA

17 Appendix list

1. LEGO® MINDSTORMS® EV3 programmable brick main hardware schematics
2. LEGO® MINDSTORMS® EV3 programmable brick display hardware schematics
3. LEGO® MINDSTORMS® EV3 touch sensor hardware schematics
4. LEGO® MINDSTORMS® EV3 color sensor hardware schematics
5. LEGO® MINDSTORMS® EV3 infrared seeker sensor hardware schematics
6. LEGO® MINDSTORMS® EV3 beacon hardware schematics
7. LEGO® MINDSTORMS® EV3 ultrasonic sensor hardware schematics
8. LEGO® MINDSTORMS® EV3 gyro sensor hardware schematics
9. LEGO® MINDSTORMS® EV3 rechargeable battery hardware schematics
10. LEGO® MINDSTORMS® EV3 large motor hardware schematics
11. LEGO® MINDSTORMS® EV3 medium motor hardware schematics