

NXShield Interface Specifications

Power Specs:

NXShield can be powered from external power supply.

Max Power Rating: 10.5 Volts DC

Minimum 6.6 Volts DC needed to run NXT motors or Servos.

Motor Ports Specs:

NXShield has 4 NXT Motor Ports. The ports are rated as follows:

Output Rating: 10.5 V DC, 1 amp, thermally protected.

Sensor Ports Specs:

NXShield has 4 NXT Sensor Ports. Each sensor port has two power pins, For power delivery, the ports are rated as follows:

Power Source 1 (pin 1):

Voltage: 9 V

Max current rating: 50mA continuous current per port.

Power 2: 5 V (VCC pin - pin 4)

Max current rating: 100mA per port

Max current not to exceed: 200mA from combined 4 ports.

I2C Ports Specs:

NXShield has 2 I2C Ports. One has male pin headers and other has female interface.

I2C Voltage: 5 Volts

Max current rating: 100mA per port.

Max current not to exceed: 200mA from combined 2 ports.

Servo Ports Specs:

NXShield-D has 1 set of 6 Servo ports whereas NXShield-M has 2 sets of 6 Servo Ports.

Small servo set (6 servos each on NXShield-D and NXShield-M - next to BBS2):

Servo Voltage: 5 V

Max Current rating: 1.5 amp peak.

Large servo set (6 servos on NXShield-M - next to M1):

Servo Voltage: Same as Supply Voltage

Max Current rating: 3 amp peak.

Analog Ports Specs:

NXShield has 12 Analog Ports.

Voltage: 5 Volts

Max Current rating: 50mA per port.

Max current not to exceed: 50mA from combined 12 ports.

Arduino Software resources Used by NXShield-M&D

Timer 2 from Arduino:

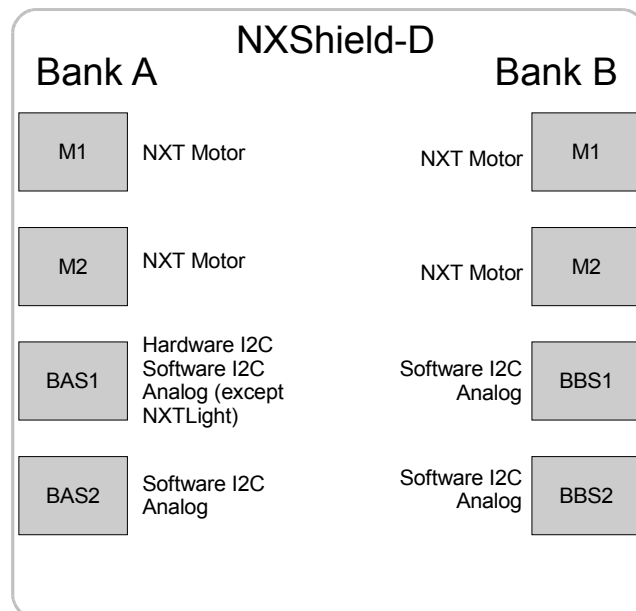
NXShield uses Timer 2 from Arduino resources for its LED and button handling. Note that, Tone Library also uses Timer 2. If you are using Tone generation, you will not be able to use NXShield buttons and LED.

Arduino Hardware resources Used by NXShield-D

Board	PIN	Purpose
D/U	A0	I2C for BAS2 (SDA)
D/U	A1	I2C for BBS1 (SDA)
D/U	A2	I2C for BBS2 (SDA)
D/U	A4	I2C for BAS1 (SDA)
D/U	A5	I2C for BAS1 (SCL)
D/U	Di2	I2C for BAS2 (SCL)
D/U	Di4	I2C for BBS1 (SCL)
D/U	Di7	I2C for BBS2 (SCL)
D/U	TBD	Button and LED
D/U	TBD	Button and LED
D/U	TBD	Button and LED

Port Features of NXShield-D

Picture below illustrates the supported devices on prominent Ports.

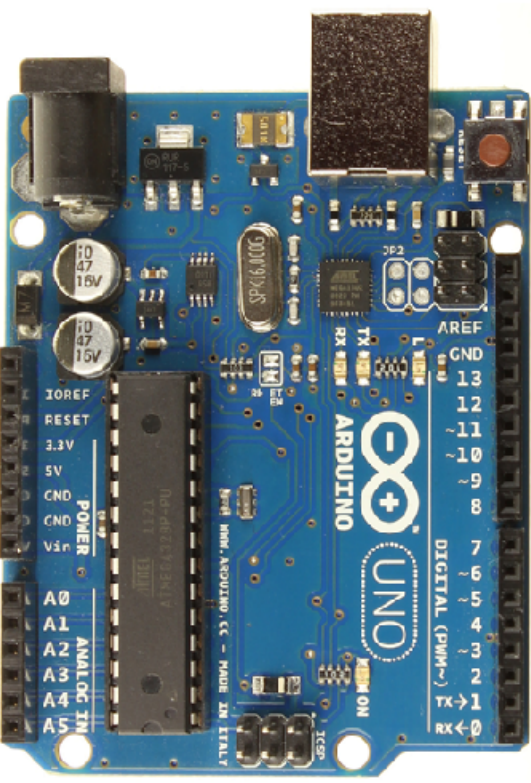


Arduino Uno with Pin mappings (for NXShield-D) –

Green Labels: These pins could be used for other purposes without affecting any of NXShield functionality.

Blue Labels: These pins could be used for other purposes by sacrificing some of the NXShield functionality.

Red Labels: These pins are required for NXShield functionality. These are used for Hardware I2C, and if you need to use for another I2C device, you may share them as long as that device is compliant to I2C specs and has non-conflicting address.



The image shows an Arduino Uno board with various components and pin headers. The board is blue with white text. The ATmega328P microcontroller is visible in the center. The pin headers are labeled with their functions and pin numbers. The board is oriented with the USB Type-B port at the top right and the DC power jack at the top left.

Pin Usage	
Available (shared)	RESET
Available (shared)	3.3V
Available (shared)	5V
Available (shared)	GND
Available (shared)	GND
Available (shared)	Vin
SDA_BAS2	A0
SDA_BBS1	A1
SDA_BBS2	A2
LED_GREEN, BTN_GO	A3
SDA_BAS1	A4
SCL_BAS1	A5

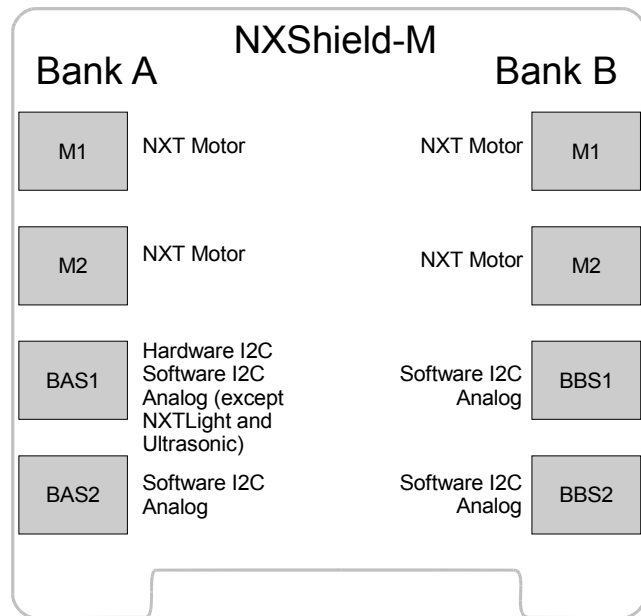
Pin Usage	
AREF	Available(unconnected)
GND	Available (shared)
13	Available(unconnected)
12	BTN_RIGHT, LED_BLUE
11	Routed to Surface
10	Routed to Surface
9	Routed to Surface
8	BTN_LEFT, LED_RED
7	SCL_BBS2
6	Routed to Surface
5	Routed to Surface
4	SCL_BBS1
3	Routed to Surface
2	SCL_BAS2
1	Available(unconnected)
0	Available(unconnected)

Arduino Hardware resources Used by NXShield-M

Board	PIN	Purpose
Mega	A13	I2C for BAS2 (SDA)
Mega	A14	I2C for BBS1 (SDA)
Mega	A15	I2C for BBS2 (SDA)
Mega	Di17	I2C for BBS1 (SCL)
Mega	Di18	I2C for BBS2 (SCL)
Mega	Di19	I2C for BAS2 (SCL)
Mega	Di20	I2C for BAS1 (SDA)
Mega	Di21	I2C for BAS1 (SCL)
Mega	Di13	Button and LED
Mega	Di14	Button and LED
Mega	Di15	Button and LED

Port Features of NXShield-M

Picture below illustrates the supported devices on prominent Ports.

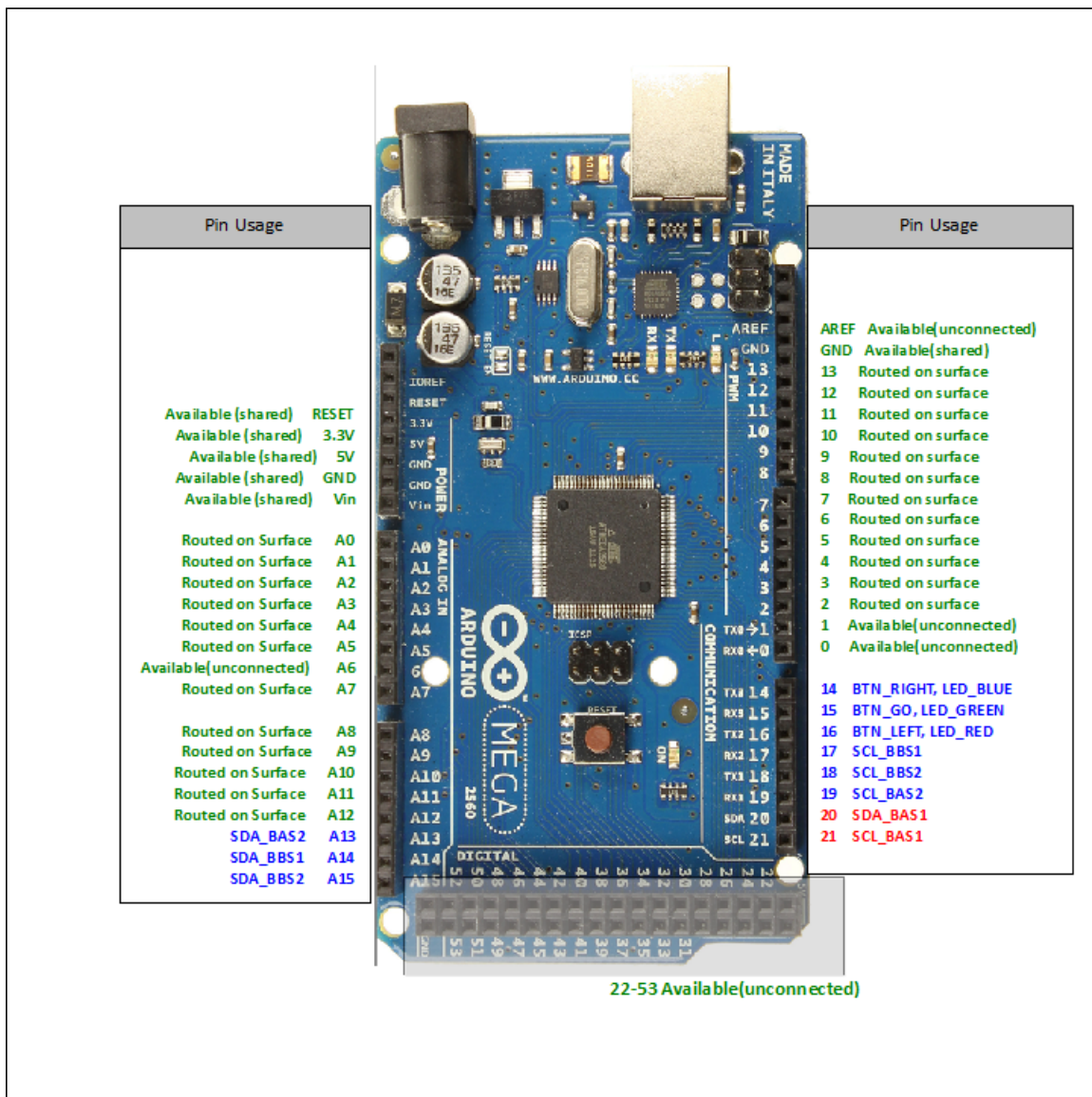


Arduino Mega with Pin mappings (for NXShield-M) –

Green Labels: These pins could be used for other purposes without affecting any of NXShield functionality.

Blue Labels: These pins could be used for other purposes by sacrificing some of the NXShield functionality.

Red Labels: These pins are required for NXShield functionality. These are used for Hardware I2C, and if you need to use for another I2C device, you may share them as long as that device is compliant to I2C specs and has non-conflicting address.



Pros and cons of Software Vs Hardware I2C:

Software I2C: The Software I2C protocol is slower than the Hardware I2C, however it is tailored to work with NXT Digital sensors.

Hardware I2C: This I2C protocol provides high speed communication. But NXT digital sensors (e.g. Ultrasonic) do not work with this protocol.

All mindsensors' I2C devices (and most other I2C devices) can be used with both these protocols (on any of the sensor ports).

You can connect Port Splitter(s) on any of these ports, and connect several more digital/I2C devices.

I2C Bus address

Factory Default Address of Bank-A: 0x06

Factory Default Address of Bank-B: 0x08

Changing the I2C Bus Address:

The I2C bus address of each NXShield bank can be changed separately. To set an address different from default address, send sequence of following commands on the command register of that bank:

0xA0, 0xAA, 0xA5, <new I2C address>

Note: Send these commands with no break/read operation in between. This new address is effective immediately. Please note down your address carefully for future reference.

Changing address from NXT:

You can download NXT executable programs from following location:

http://www.mindsensors.com/index.php?module=documents&JAS_DocumentManager_op=viewDocument&JAS_Document_id=91

To change the address, connect NXT to Port BAS1 of your NXShield.

I2C Registers:

Each bank of NXShield appears as a set of registers as follows:

Register	Read	Write
0x00-0x07	Firmware version - <i>Vxxxx</i>	-
0x08-0x0f	Vendor Id - <i>open/ctrn</i>	-
0x10-0x17	Device ID - NXShld	-
0x41	You can read the NXShield battery voltage at this register. (voltage in milli-volts = register value * 37).	Command
	Motor 1 Write Parameters	
0x42	Encoder Target for Motor 1 (long) 0x42: Least Significant Byte 0x43: Byte 2 0x44: Byte 3 0x45: Most Significant Byte	Encoder Target of Motor 1 (long)
0x46	Speed for Motor 1 (byte)	Speed for Motor 1 (byte)
0x47	Time to run in seconds for Motor 1 (byte)	Time to run in seconds for Motor 1 (byte)
0x48	Command register B for Motor 1	Command register B for Motor 1 (set this value to 0 as this is for future use)
0x49	Command register A for Motor 1 (read the description below for details of this register).	Command register A for Motor 1 (read the description below for details of this register).
	Motor 2 Write Parameters	
0x4A	Encoder target for Motor 2 (long) 0x4A: Least Significant Byte 0x4B: Byte 2 0x4C: Byte 3 0x4D: Most Significant Byte	Encoder Value of Motor 2 (long)
0x4E	Speed for Motor 2 (byte)	Speed for Motor 2 (byte)
0x4F	Time to run in seconds for Motor 2 (byte)	Time to run in seconds for Motor 2 (byte)
0x50	Command register B for Motor 2	Command register B for Motor 2

0x51	Command register A for Motor 2 (read the description below for details of this register).	Command register A for Motor 2 (read the description below for details of this register).
	Motor Read Parameters	
0x62	Encoder position of Motor 1 (long) 0x62: Least Significant Byte 0x63: Byte 2 0x64: Byte 3 0x65: Most Significant byte	-
0x66	Encoder position of Motor 2 (long) 0x66: Least Significant Byte 0x67: Byte 2 0x68: Byte 3 0x69: Most Significant Byte	-
0x72	Status Motor 1 (byte). See section below for details of this register.	
0x73	Status Motor 2 (byte). See section below for details of this register.	
0x76	Tasks Running for Motor 1 (byte)	
0x77	Tasks Running for Motor 2 (byte)	
	Registers for Advanced PID control	Writing these registers has immediate effect on operation. These registers will be reset to factory default values upon power cycle.
0x7A	Kp for Encoder Position Control (int) 0x7A: Least Significant Byte 0x7B: Most Significant Byte	Kp for Encoder Position Control (int)
0x7C	Ki for Encoder Position Control (int) 0x7C: Least Significant Byte 0x7D: Most Significant Byte	Ki for Encoder Position Control (int)
0x7E	Kd for Encoder Position Control (int) 0x7E: Least Significant Byte 0x7F: Most Significant Byte	Kd for Encoder Position Control (int)
0x80	Kp for Speed Control (int) 0x80: Least Significant Byte	Kp for Speed Control (int)

	0x81: Most Significant Byte	
0x82	Ki for Speed Control (int) 0x82: Least Significant Byte 0x83: Most Significant Byte	Ki for Speed Control (int)
0x84	Kd for Speed Control (int) 0x84: Least Significant Byte 0x85: Most Significant Byte	Kd for Speed Control (int)
0x86	Pass Count - The PID controller repeatedly reads internal encoder ticks, this is the number of times the encoder ticks reading should be within tolerance. (default 5)	Pass Count - Higher Pass count gives more time to position internal encoder, thus providing better accuracy in positioning, but will take longer time. Change this only if you need different motor positioning speeds and accuracy. (For normal usage, default values will be OK).
0x87	Tolerance - The Tolerance (in ticks) for encoder positioning. (default 80).	Tolerance - the accuracy you desire while positioning. Low number will position the encoders more accurately, but may take longer time. Change this only if you need different motor positioning speeds and accuracy. (For normal usage, default values will be OK).

I2C Registers for Analog sensors:

	Read	Write
0x8A	Mode for Sensor 1	Mode for Sensor 1
0x8B	Mode for Sensor 2	Mode for Sensor 2
0x8C	Sensor 1 value (LSB)	-
0x8D	Sensor 1 value (MSB)	-
0x8E	Sensor 2 value (LSB)	-
0x8F	Sensor 2 value (MSB)	-

Other I2C Registers:

	Read	Write
0x90	Reason of last Reset 00 - Power On 02 - Reset due to brown-out 03 - Reset due to User Pressed Reset button	

Supported I2C Commands:

CMD	Hex	Description
R	0x52	Reset all Encoder values and motor parameters. (This does not reset the PID parameters).
S	0x53	Issue commands to both motors at the same time, for Synchronized starting of both motors.
		Motor Stopping Commands
a	0x61	Motor 1: Float while stopping.
b	0x62	Motor 2: Float while stopping.
c	0x63	Motor 1 and 2: Float while stopping.
A	0x41	Motor 1: Brake while stopping.
B	0x42	Motor 2: Brake while stopping.
C	0x43	Motor 1 & 2: Brake while stopping.
		Encoder Reset Commands
r	0x72	Motor 1: Reset Encoder to zero
s	0x73	Motor 2: Reset Encoder to zero

These commands are issued on command register (0x41).

Motor Command Register Explained

Each motor has two command registers (Register A and Register B). In current release Register B is reserved for future use and must be set to zero.

Bits in Register A should be set to 1 to avail functionality as described below.

Register Bit	Turn this bit to 1 for following functionality
Least significant bit (bit 0)	Speed control of your motor. The NXShield will honor speed values specified in the speed register for the respective motor.
Bit 1	Ramp the speed up or down. While starting the motor or changing speed, the NXShield will ramp up or ramp down the power to new value. If this bit is zero, the power changes are sudden, e.g. full power is applied to motors as they start.
Bit 2	Relative change based on encoder values. This is useful when Bit 3 is turned on, and in this case, the NXShield will make a relative movement from last seen Encoder position (it will add the new Encoder position to old position and move to that resulting position). Useful when turning by degrees or rotations. If this bit is 0, the Encoder positions are taken as absolute values.

Register Bit	Turn this bit to 1 for following functionality
Bit 3	Encoder control of your motor. The NXShield will honor Encoder values specified in Encoder position register for respective motor. If speed values are also specified, and speed bit is set on, motors will rotate to new encoder position with the specified speed.
Bit 4	Brake or Float at the completion of motor movement. If this bit is 1, motor will Brake at the completion, otherwise it will float.
Bit 5	Encoder active feedback. This bit is used when Encoder control is used. If this bit is set to 1 at the completion of motor movement, the NXShield will continue to hold the Encoder position (i.e. if motor is turned by external force, NXShield will try to restore it to last specified Encoder position). If this bit is zero, the motor may float.
Bit 6	Timed control of your motor. The NXShield will honor the time specified value in 'Time to run' register and run the motor for specified time. (If Time control bit as well as Encoder Control bit is on, the Timed control has precedence over encoder control).
Most significant bit (bit 7)	GO. When this bit is set to 1, all above bit values are brought into effect. This is useful to synchronized starting of both motors. As it takes some time to write each motor's register values. The I2C command 'S' can be used to change these bits to 1 for both motors at once.

Motor Status Register Explained

Each motor has one status register. Each bit in status register indicates various situations with the motor as explained below.

Register Bit	Value 1 indicates the situation is true
Least significant bit (bit 0)	Speed Control is ON. Motor is programmed to move at a fixed speed.
Bit 1	Motor is Ramping (up or down). If the Power ramp is enabled, this bit is 1 while the motor is ramping (while changing its speed).
Bit 2	Motor is powered. (This may not mean motor is moving.)
Bit 3	Positional Control is ON. The motor is either moving towards desired encoder position or holding its

Register Bit	Value 1 indicates the situation is true
	position.
Bit 4	Motor is in Brake mode. (0 value of this bit means motor is floating).
Bit 5	Motor is overloaded. If the external load prevents motor from achieving desired speed, this bit is set to 1.
Bit 6	Motor is in timed mode. This bit is 1 while the motor is programmed to move for given duration.
Most significant bit (bit 7)	Motor is stalled. The external load caused the motor to stop moving.

Running Motors for Unlimited Duration

Not specifying Encoder Control or Timed Control bit will result in unlimited running of motor, (the speed control is honored if specified). To stop motors started with 'Unlimited Duration' use respective Stop command from the command set.



NOTE

When motors are set to run for 'Unlimited Duration', they will continue to run until a Stop command is issued (or power is disconnected). In other words, after starting the motors for 'Unlimited Duration' if your program does something else without stopping the motors, they will continue to run.

How to detect if motor is moving or not moving:

In General:

Anytime when the 'Stalled' bit is 1, the motor is not moving.

Running in encoder mode:

During moving:

Position Control bit (bit 3) is 1

'Motor is Powered' bit (bit 2) is 1

Finished moving normally:

All bits are zero

Stopped due to stall:

Position Control bit (bit 3) is 1

'Motor is Powered' bit (bit 2) is 1

Stalled bit (bit 7) is 1

Running in timed mode:

During moving:

Timed Mode bit (bit 6) is 1

'Motor is Powered' bit (bit 2) is 1

Finished moving normally:

All bits are zero

Stopped due to stall (while time is not over):

Timed Mode bit (bit 6) is 1

'Motor is Powered' bit (bit 2) is 1

Stalled bit (bit 7) is 1

Stopped due to stall (after time is over):

All bits are zero.

Using RCX Motors with NXShield:

You can connect the RCX motors (71427) to NXShield, using NXT Conversion cable (part #1676).

