Guide to the Kobuki.h Arduino library

Authors: Michael J. Conte, Pratheek Manjunath

1. Introduction:

This 'User Guide' explains how to control the Kobuki robot base by sending serial data from a micro-controller. In order to send serial data, one needs a microcontroller to generate the commands and a Transmit/Receive port on the robot which is capable of receiving the serial byte streams.

The <Kobuki.h> file is a convenient resource developed for the Arduino IDE which can be employed to create the serial byte streams required in order to command the robot and to also seek feedback from the robot sensors. This library requires the use of an Arduino Mega since it uses two Serial ports - one port for communicating with the robot (Serial1) and the other for sending data over USB to the Serial Monitor.

2. Hardware/Wiring:

In order to connect the Kobuki to the Arduino, wire pin 1 (TX pin) of the Arduino to the Rx pin (pin 1) of the Kobuki's DB25 expansion port. Next, wire pin 0 (RX pin) of the Arduino to the Tx pin (pin 2) of the DB25. Lastly wire the 5V and GND ports of the Arduino to the EX5 and GND pins respectively on the DB25 serial port. See the diagram below for reference.

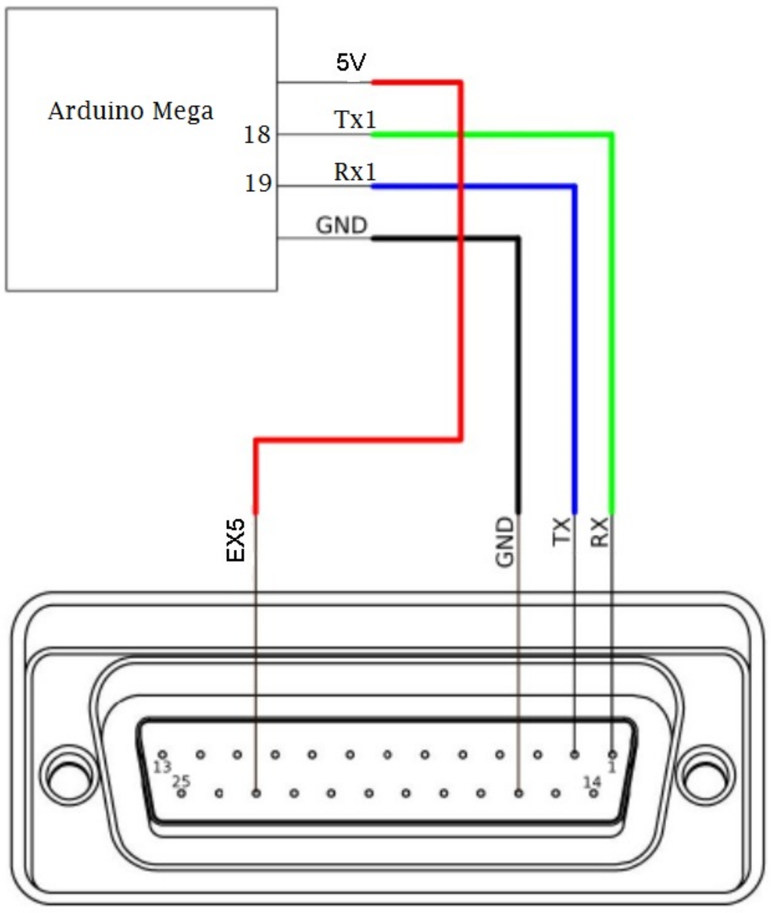

Figure 1: Kobuki - Arduino wiring diagram

3. The Kobuki Header File:

This header file gives you access to the “Kobuki” class which contains three functions: *command*, *feedback* and *refresh_sensors*.

3.1 Command:

This function gives you control over Kobuki’s actuators responsible for its mobility, audio playback and visual indication. These are its drive motors, speaker and LEDs respectively.

3.1.1 Syntax:

```
kobuki.command (opcode, payload1, payload2);
```

Opcode- An integer which corresponds to a specific function of the Kobuki
Payload1- The first argument to be provide which is specific to an opcode. This could be of datatype ‘int’ or ‘byte’ or ‘char’.
Payload2- The second argument corresponding to an opcode. Some opcodes require only one payload byte.

3.1.2 Examples:

- Move forward at 270 mm/s: `Kobuki.command(1, 270, 0x0000);`
- Move backwards at 35 mm/s: `Kobuki.command(1, -35, 0x0000);`
- Rotate clockwise in place at 45 degrees/s:
`Kobuki.command(1, 100, 0xffff)` *OR* `Kobuki.command(1, -100, 0x0001);`
- Rotate counter-clockwise in place at 90 degrees/s:
`Kobuki.command(1, 200, 0x0001)` *OR* `Kobuki.command(1, -200, 0xffff);`
- Move at 270 mm/s in an arc with radius of curvature 100mm:
`Kobuki.command(1, 270, 0x0064);`

3.1.3 Summary of Opcodes and their Respective Payload Fields

Opcode	Description	Payload1	Range1	Payload2	Range2
1	Control wheel motors	Speed in mm/s,	-700 to +700 (positive velocities to move forward, negative velocities to move in reverse)	Radius of curvature in mm (Use hexadecimal format)	Left turn: 0x0000 - 0x7FFF Right turn: 0xFFFF - 0x8000 (negative radii will cause robot to turn right and positive values will make it turn left)

3	Play custom sounds	Note Frequency in Hz	70 to 180	Duration in ms	0 to 255
4	Play predefined sounds	Predefined 2-byte hex Values, see command layout section	N/A	N/A	N/A
12	Control general purpose outputs	Predefined 2-byte hex Values, see command layout section	N/A	N/A	N/A

Table 1: Opcode Reference Chart

3.1.4 Special Payloads:

Certain opcodes have pre-defined payloads which correspond to a certain functionality. Following is a summary for these opcodes.

For Opcode 4

Sequence Number	Description
0	ON sound
1	OFF sound
2	RECHARGE sound
3	BUTTON sound
4	ERROR sound
5	CLEANING START sound
6	CLEANING END sound

Table 2: Sounds Reference chart

For Opcode 12

Output Flag	Description
0x0001	Sets digital output pin ch. 0 on expansion port high
0x0002	Sets digital output pin ch. 1 on expansion port high
0x0004	Sets digital output pin ch. 2 on expansion port high
0x0008	Sets digital output pin ch. 3 on expansion port high
0x0010	turn on 3.3V ch. external power
0x0020	turn on 5V ch. external powers
0x0040	turn on 12V/5A ch. external powers
0x0080	turn on 12V/1.5A ch. external powers
0x0100	turn LED1 red
0x0200	turn LED1 green
0x0400	turn LED2 red
0x0800	turn LED2 green

Table 3: General Purpose Output Reference

3.2 Refresh_Sensors:

This function updates and stores the most recent values outputted by the sensors. Calling the refresh_sensors( ) ensures that the feedback function fetches the most current values.

3.2.1 Syntax:

kobuki.refresh_sensors( );

3.3 Feedback:

This function provides you with data published by the kobuki's sensors. The function expects a 'sensor_id' and returns the output of the corresponding sensor.

3.3.1 Syntax:

kobuki.feedback(sensor_id);

Sensor_id: An integer corresponding to the sensor to be queried. Refer to Table 4

3.3.2 Examples:

- Get left encoder value: `Kobuki.feedback(8);`
  A value of 17082 corresponds to a distance of 1460mm.
  (11.7 ticks / mm travelled)
  (For wheel odometry specs, refer to page 14 of the Kobuki User Guide)

- Get bump sensor status: `Kobuki.feedback(5);`

If the value returned is 0x02 (i.e. 0b0010), it means that the center bumper has been pressed. Similarly, a value of 0x06 (i.e. 0b0110) means that the left and center bumpers have been pressed.

3.3.3 Summary of Sensors and their Respective IDs

Sensor ID	Sensor Name	Description	Byte(s) Returned
5	Bumper	Bits will be set corresponding to the three bump sensors.	0x01 : right bumper 0x02 : central bumper 0x04 : left bumper
6	Wheel Drop Sensor	Bits will be set corresponding to the three wheel-drop sensors	0x01 : right wheel 0x02 : left wheel 0x03 : both wheels
7	Cliff Sensors	Bits will be set corresponding to the three cliff sensors	0x01 : right cliff sensor 0x02 : central cliff sensor 0x04 : left cliff sensor
8	Left Encoder	Accumulated encoder data of left and right wheels in ‘ticks’.	Circulates from 0 to 65535 (Increasing increments of this value means the kobuki is moving forward, decreasing means it's moving backward)
10	Right Encoder		

12	Left PWM	Prints the PWM value that was applied to left and right motors in order to make it move. The variable storing this value should be a signed data type to accurately represent direction.	-128 to 127 (Negative sign indicates backward motion)
13	Right PWM		
14	Buttons	Bits will be set corresponding to the three buttons	0x01 : Button 0 0x02 : Button 1 0x04 : Button 2
15	Charger	Indicates various charging states of the Kobuki.	0 : DISCHARGING 2 : DOCKING_CHARGED 6 : DOCKING_CHARGING 18 : ADAPTER_CHARGED 22 : ADAPTER_CHARGING
16	Battery	Indicates battery voltage in 0.1 volt increments Typically 16.7 V when fully charged	0d000 to 0d167 OR 0x00 to 0xA7
17	Overcurrent Flags	Bits will be set based on the two current limiters	0x01 : left wheel 0x02 : right wheel
20	IR Right Signal	Bits will be set corresponding to the three infrared sensors	0x01 : NEAR_LEFT 0x02 : NEAR_CENTER 0x04 : NEAR_RIGHT 0x08 : FAR_CENTER 0x10 : FAR_LEFT 0x20 : FAR_RIGHT
21	IR Center Signal		
22	IR Left Signal		
25	Gyro Angle	Angle of deviation measured along the z-axis	0 to 360 (degrees)

27	Gyro Angle Rate	Angular Velocity about Z-axis. Also known as yaw rate. Unit: degrees/s	-250 °/s to 250 °/s
34	Right Cliff	ADC output of each photo detector	Data range: 0 to 4095 (0 to 3.3V) Distance range: 2 to 15 cm
36	Center Cliff		
38	Left Cliff		
42	Left Motor Current	Current Supplied to Motors in multiples of 10mA	0 to 75 (0 mA to 750mA)
43	Right Motor Current		

Table 4: Sensors Reference