SoftwareSerial Library

The SoftwareSerial library allows serial communication on other digital pins of an Arduino board.

AUTHOR: **Arduino**

LAST REVISION: **09/14/2022, 03:09 PM**

The SoftwareSerial library allows serial communication on other digital pins of an Arduino board, using software to replicate the functionality (hence the name "SoftwareSerial"). It is possible to have multiple software serial ports with speeds up to 115200 bps. A parameter enables inverted signaling for devices which require that protocol.

The version of SoftwareSerial included in 1.0 and later is based on the NewSoftSerial library by 'Mikal Hart'.

To use this library:

COPY

```
#include <SoftwareSerial.h>
```

Limitations of This Library

SoftwareSerial library has the following known limitations:

- It cannot transmit and receive data at the same time.
- If using multiple software serial ports, only one can receive data at a time.
- Not all pins on the Mega and Mega 2560 boards support change interrupts, so only the following can be used for RX: 10, 11, 12, 13, 14, 15, 50, 51, 52, 53, A8 (62), A9 (63), A10 (64), A11 (65), A12 (66), A13 (67), A14 (68), A15 (69). Not all pins on the Leonardo and Micro boards support change interrupts, so only the following can be used for RX: 8, 9, 10, 11, 14 (MISO), 15 (SCK), 16 (MOSI).
- On Arduino or Genuino 101 boards the current maximum RX speed is 57600bps.
- On Arduino or Genuino 101 boards RX doesn't work on digital pin 13.

If your project requires simultaneous data flows, see Paul Stoffregen's AltSoftSerial library.

Examples

- SoftwareSerial example: sometimes one serial port just isn't enough!
- Two port receive: Work with multiple software serial ports.

Methods

`SoftwareSerial()`

Syntax

COPY

```
SoftwareSerial(rxPin, txPin, inverse_logic)
```

Parameters

rxPin: the pin on which to receive serial data.

txPin: the pin on which to transmit serial data.

inverse_logic: used to invert the sense of incoming bits (the default is normal logic). If set, SoftwareSerial treats a LOW (0v on the pin, normally) on the RX pin as a 1-bit (the idle state) and a HIGH (5V on the pin, normally) as a 0-bit. It also affects the way that it writes to the TX pin. Default value is false.

Returns

None.

Example

COPY

```
#include <SoftwareSerial.h>

const byte rxPin = 2;
const byte txPin = 3;

// Set up a new SoftwareSerial object
SoftwareSerial mySerial (rxPin, txPin);
```

See also

available()

begin()

isListening()

overflow()

peek()

read()

print()

println()

listen()

write()

available()

Get the number of bytes (characters) available for reading from a software serial port. This is data that has already arrived and stored in the serial receive buffer.

Syntax

COPY

```
mySerial.available()
```

Returns

The number of bytes available to read.

Example

COPY

```
#include <SoftwareSerial.h>

#define rxPin 10
#define txPin 11

// Set up a new SoftwareSerial object
SoftwareSerial mySerial =  SoftwareSerial(rxPin, txPin);

void setup()  {
    // Define pin modes for TX and RX
    pinMode(rxPin, INPUT);
    pinMode(txPin, OUTPUT);

    // Set the baud rate for the SoftwareSerial object
    mySerial.begin(9600);
}

void loop() {
    if (mySerial.available() > 0) {
        mySerial.read();
    }
}
```

See also

SoftwareSerial()

begin()

isListening()

overflow()

peek()

read()

print()

println()

listen()

write()

begin()

Sets the speed (baud rate) for the serial communication. Supported baud rates are: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 31250, 38400, 57600, and 115200 bauds.

Syntax

COPY

```
mySerial.begin(speed)
```

Parameters

speed: the desired baud rate (long). Supported baud rates are: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 31250, 38400, 57600, and 115200 bauds.

None.

Example

COPY

```
#include <SoftwareSerial.h>

#define rxPin 10
#define txPin 11

// Set up a new SoftwareSerial object
SoftwareSerial mySerial =  SoftwareSerial(rxPin, txPin);

void setup()  {
    // Define pin modes for TX and RX
    pinMode(rxPin, INPUT);
    pinMode(txPin, OUTPUT);

    // Set the baud rate for the SoftwareSerial object
    mySerial.begin(9600);
}

void loop() {
    // ...
}
```

See also

SoftwareSerial()

available()

isListening()

overflow()

peek()

read()

print()

println()

listen()

write()

isListening()

Tests to see if requested software serial object is actively listening.

Syntax

COPY

```
mySerial.isListening()
```

Parameters

None.

Returns

Boolean.

```
// Set up a new SoftwareSerial object with RX in digital pin 10 and TX in dig
SoftwareSerial portOne(10, 11);

void setup() {
    // Set the baud rate for the Serial port
    Serial.begin(9600);

    // Set the baud rate for the SerialSoftware object
    portOne.begin(9600);
}

void loop() {
    if (portOne.isListening()) {
        Serial.println("portOne is listening!");
    }

    // ...
```

See also

SoftwareSerial()

available()

begin()

overflow()

peek()

read()

print()

println()

listen()

write()

overflow()

Tests to see if a SoftwareSerial buffer overflow has occurred. Calling this function clears the overflow flag, meaning that subsequent calls will return false unless another byte of data has been received and discarded in the meantime. The SoftwareSerial buffer can hold up to 64 bytes.

Syntax

COPY

```
mySerial.overflow()
```

Parameters

None.

Returns

Boolean.

Example

```
void setup() {
    // Set the baud rate for the Serial port
    Serial.begin(9600);

    // Set the baud rate for the SerialSoftware object
    portOne.begin(9600);
}

void loop() {
    if (portOne.overflow()) {
        Serial.println("portOne overflow!");
    }

    // ...
```

See also

SoftwareSerial()

available()

begin()

isListening()

peek()

read()

print()

println()

listen()

write()

peek()

Return a character that was received on the RX pin of the software serial port. Unlike read(), however, subsequent calls to this function will return the same character. Note that only one SoftwareSerial object can receive incoming data at a time (select which one with the listen() function).

Syntax

COPY

```
mySerial.peek()
```

Parameters

None.

Returns

The character read or -1 if none is available.

Example

COPY

```
#include <SoftwareSerial.h>

// Set up a new SoftwareSerial object with RX in digital pin 10 and TX in dig
```

```
}

void loop() {
    char c = mySerial.peek();
}
```

See also

SoftwareSerial()

available()

begin()

isListening()

overflow()

read()

print()

println()

listen()

write()

read()

Return a character that was received on the RX pin of the SoftwareSerial objecto. Note that only one SoftwareSerial object can receive incoming data at a time (select which one with the listen() function).

Syntax

COPY

```
mySerial.read()
```

Parameters

None.

Returns

The character read or -1 if none is available.

Example

COPY

```
#include <SoftwareSerial.h>

// Set up a new SoftwareSerial object with RX in digital pin 10 and TX in dig
SoftwareSerial mySerial(10, 11);

void setup() {
    // Set the baud rate for the SerialSoftware object
    mySerial.begin(9600);
}

void loop() {
    char c = mySerial.read();
}
```

print()

Prints data to the transmit pin of the SoftwareSerial object. Works the same as the Serial.print() function.

Syntax

COPY

```
mySerial.print(val)
```

Parameters

val: the value to print.

Returns

The number of bytes written (reading this number is optional).

Example

COPY

```
#include <SoftwareSerial.h>

// Set up a new SoftwareSerial object with RX in digital pin 10 and TX in d
SoftwareSerial mySerial(10, 11);

int analogValue;

void setup() {
    // Set the baud rate for the SerialSoftware object
    mySerial.begin(9600);
}

void loop() {
    // Read the analog value on pin A0
    analogValue = analogRead(A0);

    // Print analogValue in the Serial Monitor in many formats:
    mySerial.print(analogValue);         // Print as an ASCII-encoded decim
    mySerial.print("\t");                // Print a tab character
    mySerial.print(analogValue, DEC);    // Print as an ASCII-encoded decim
    mySerial.print("\t");                // Print a tab character
    mySerial.print(analogValue, HEX);    // Print as an ASCII-encoded hexad
    mySerial.print("\t");                // Print a tab character
    mySerial.print(analogValue, OCT);    // Print as an ASCII-encoded octal
    mySerial.print("\t");                // Print a tab character
    mySerial.print(analogValue, BIN);    // Print as an ASCII-encoded binar
    mySerial.print("\t");                // Print a tab character
```

```
33      mySerial.println();                  // Print a line feed character
```

See also

- SoftwareSerial()
- available()
- begin()
- isListening()
- overflow()
- peek()
- read()
- print()
- println()
- listen()
- write()

println()

Prints data to the transmit pin of the SoftwareSerial object followed by a carriage return and line feed. Works the same as the Serial.println() function.

Syntax

COPY

```
mySerial.println(val)
```

Parameters

val: the value to print.

Returns

The number of bytes written (reading this number is optional).

Example

COPY

```
#include <SoftwareSerial.h>

// Set up a new SoftwareSerial object with RX in digital pin 10 and TX in d
SoftwareSerial mySerial(10, 11);

int analogValue;

void setup() {
    // Set the baud rate for the SerialSoftware object
    mySerial.begin(9600);
}

void loop() {
    // Read the analog value on pin A0
    analogValue = analogRead(A0);

    // Print analogValue in the Serial Monitor in many formats:
    mySerial.print(analogValue);         // Print as an ASCII-encoded decim
    mySerial.print("\t");                // Print a tab character
```

```
25      mySerial.print("\t");                // Print a tab character
26      mySerial.print(analogValue, BIN);    // Print as an ASCII-encoded binar
27      mySerial.print("\t");                // Print a tab character
28      mySerial.print(analogValue/4, BYTE); // Print as a raw byte value (divi
29                                           // value in 4 because analogRead()
30                                           // from 0 to 1023, but a byte can
31
32      mySerial.print("\t");                // Print a tab character
```

See also

SoftwareSerial()

available()

begin()

isListening()

overflow()

peek()

read()

print()

listen()

write()

listen()

Enables the selected SoftwareSerial object to listen. Only one SoftwareSerial object can listen at a time; data that arrives for other ports will be discarded. Any data already received is discarded during the call to listen() function (unless the given instance is already listening).

Syntax

COPY

```
mySerial.listen()
```

Parameters

None.

Returns

Returns true if it replaces another.

Example

COPY

```
#include <SoftwareSerial.h>

// Set up a new SoftwareSerial object with RX in digital pin 10 and TX in d
SoftwareSerial portOne(10, 11);

// Set up a new SoftwareSerial object with RX in digital pin 8 and TX in di
SoftwareSerial portTwo(8, 9);

void setup() {
    // Set the baud rate for the Serial object
    Serial.begin(9600);
```

```
void loop() {
    // Enable SoftwareSerial object to listen
    portOne.listen();

    if (portOne.isListening()) {
        Serial.println("portOne is listening!");
    } else {
        Serial.println("portOne is not listening!");
    }

    if (portTwo.isListening()) {
        Serial.println("portTwo is listening!");
    } else {
        Serial.println("portTwo is not listening!");
    }
```

See also

SoftwareSerial()

available()

begin()

isListening()

overflow()

peek()

read()

print()

println()

write()

write()

Prints data to the transmit pin of the SoftwareSerial object as raw bytes. Works the same as the Serial.write()function.

Syntax

COPY

```
mySerial.write(val)
```

Parameters

val: the binary value to print.

Returns

The number of bytes written (reading this number is optional).

Example

COPY

```
#include <SoftwareSerial.h>

// Set up a new SoftwareSerial object with RX in digital pin 10 and TX in dig
SoftwareSerial mySerial(10, 11);

void setup() {
```

```
12      // Send a byte with the value 45
13      mySerial.write(45);
14
15      //Send the string “hello” and return the length of the string.
16      int bytesSent = mySerial.write(“hello”);
17  }
```

See also

SoftwareSerial()

available()

begin()

isListening()

overflow()

peek()

read()

print()

println()

listen()

Back to top