Communication Specification

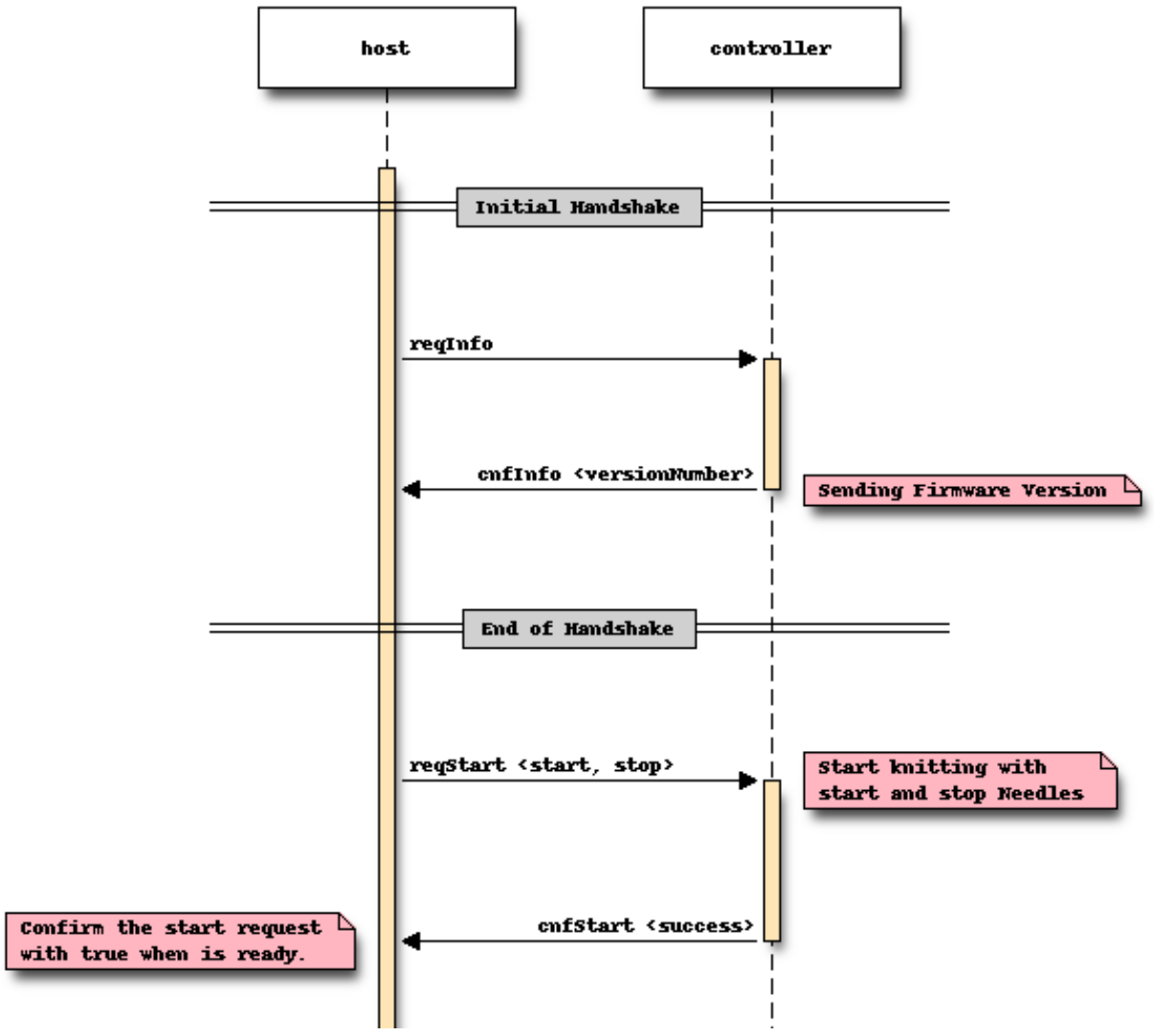
This document specifies the communication between the host and a controller with the AYAB firmware.

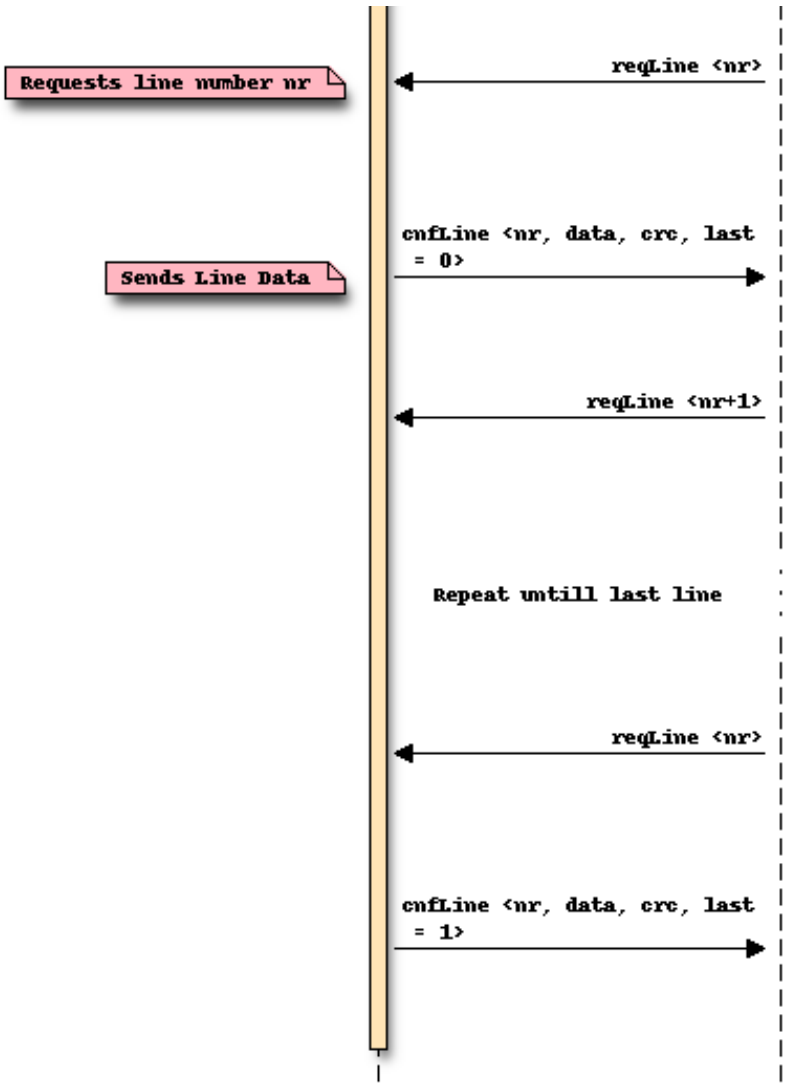
Serial Communication

115200 baud

Line Ending: `\n\r` (10 13) Each message ends with a Line Ending.

Sequence Chart





The host waits for a **indState(true)** message before requesting to start the knitting. On startup, the Arduino continuously checks for the initialization of the machine (carriage passed left hall sensor). When this happens, it sends an **indState(true)** to tell the host that the machine is ready to knit. After receiving this message, the host sends a **reqStart** message, which is immediately confirmed with a **cnfStart** message. When **reqStart** was successful, the Arduino begins to poll the host for line data with **reqLine**, the host answers with **cnfLine**. This reqLine/cnfLine happens each time the carriage moves passed the borders given by the Start/StopNeedle parameters in **reqStart**. When the host does not have any more lines to send, it marks the last line with the *lastLine* flag in its last **cnfLine** message.

To see an example implementation, see the `states of the communication module`.

Message Identifier Format

Messages start with a byte that identifies their type. This byte is called “id” or “message id” in the following document. This table lists all the bits of this byte and assigns their purpose:

Bit	Value	Name	Description and Values
7	128	message source	• 0 = the message is from the host • 1 = the message is from the controller
6	64	message type	• 0 = the message is a request • 1 = the message is a confirmation of a request
5	32	reserved	must be zero
4	16		
3	8	message identifier	These are the values that identify the messag See also Message definitions (API v4)
2	4		
1	2		
0	1		

Message definitions (API v4)

The length is the total length with id and parameters. Note that the two characters `\r\n` following the message are not included in the length.

source	name	id	length	parameters
host	reqStart	0x01	3	`0xaa 0xbb` • `aa` = left end needle (Range: 0..198) • `bb` = right end needle (Range: 1..199) Start and
hardware	cnfStart	0xC1	2	`0x0a` • `a` = success (0 = false, 1 = true)

hardware	reqLine	0x82	2	`0xaa` • `aa` = line number (Range: 0..255)
host	cnfLine	0x42	29	`0xaa 0xbb[24, 23, 22, ... 1, 0] 0xc` • `aa` = line number (Range: 0..255) • `bb[24 to 0]` = binary pixel data • `cc` = flags (bit 0: lastLine) • `dd` = CRC8 Checksum
host	reqInfo	0x03	1	
hardware	cnfInfo	0xC3	4	`0xaa 0xbb 0xcc` • `aa` = API Version Identifier • `bb` = Firmware Major Version • `cc` = Firmware Minor Version

hardware	indState	0x84	8	`0x0a 0xBB 0xbb 0xCC 0xcc 0xdd 0xee`
				• `a` = ready (0 = false, 1 = true)
				• `BBbb` = `int` left hall sensor value
				• `CCcc` = `int` right hall sensor value
				• `dd` = the carriage
				  ◦ `0` = no carriage detected
				  ◦ `1` = knit carriage “Strickschlitten”
				  ◦ `2` = hole carriage “Lochmustersch
				• `ee` = the needle number currently in p

hardware	debug	0x23	var	A debug string. The id is the character `#`.
host	reqTest	0x04	1	put the controller into test mode
host	cnfTest	0xC4	2	`0x0a` • `a` = success (0 = false, 1 = true)

The `reqStart` Message

The host starts the knitting process.

- Python: `StartRequest`
- Arduino: h_reqStart
- table: reqStart
- requests answer: The cnfStart Message
- direction: host → controller

The `cnfStart` Message

The controller indicates the success of The reqStart Message.

- Python: `StartConfirmation`
- Arduino: h_reqStart
- table: reqStart
- answers: The reqStart Message
- direction: controller → host

The `reqLine` Message

The controller requests a new line from the host.

More than 256 lines are supported. There are three possibilities for the next line based on the last line:

1. the new line is greater than the last line
2. the new line is lower than the last line
3. the new line is the last line

We choose the line closest to the last line. This is trivial for (3). In case two lines are equally distant from the last line, we choose the smaller line.

This is computed by the function `AYABInterface.utils.next_line()` which is tested and can be seen as a reference implementation for other languages.

- Python: `LineRequest`
- Arduino: Knitter::reqLine

- table: reqLine
- requests answer: The cnfLine Message
- direction: controller → host

The `cnfLine` Message

The host answers The reqLine Message with a line configuration.

This table shows the message content without the first byte that identifies the message:

Byte	Name	Description
0	line number	These are the lowest 8 bit of the line. They must match the line
1 2 ... 24 25	needle positions	Each bit of the bytes represents a needle position. • 0 = “B” • 1 = “D” For the exact mapping of bits to needles see the table b
26	flags	Bits: `0000000L` • `L` - “LastLine” (0 = false, 1 = true)
27	crc8 checksum	This checksum is computed from bytes 0 to 26, including byte

In the following table, you can see the mapping of bytes to needles.

Note

- The **Needles** are counted from the leftmost needle on the machine.
- The **Needle** count starts with `0`.
- The **Byte** numbering is taken from the table above.
- The **Bit** numbering is consistent with Message Identifier Format. The highest bit has the number 7 and the lowest bit has number 0.

Byte	1								2					

Bit	0	1	2	3	4	5	6	7	0	1	2	3	4
Needle	0	1	2	3	4	5	6	7	8	9	...		

- Python: `LineConfirmation`
- Arduino: h_cnfLine
- table: cnfLine
- answers: The reqLine Message
- direction: host → controller

The `reqInfo` Message

The host initializes the handshake.

- Python: `InformationRequest`
- Arduino: h_reqInfo
- table: reqInfo
- requests answer: The reqInfo Message
- direction: host → controller

The `cnfInfo` Message

The controller answers The reqInfo Message with the API version.

- Python: `InformationConfirmation`
- Arduino: h_reqInfo
- table: cnfInfo
- answers: The reqInfo Message
- direction: controller → host

The `indState` Message

This is sent when the controller indicates its state. When `ready` it is

- `1`, then this is the first state indication. The machine is now ready to knit
- `0`, the controller is in test mode. This message is sent periodically. The reqTest Message switches this on.
- Python: `StateIndication`
- Arduino: Knitter::indState
- table: indState

- direction: controller → host

The `debug` Message

This message ends with a `\r\n` like evey message. It contains debug information from the controller.

- Python: `Debug`
- Arduino: DEBUG_PRINT
- table: debug
- direction: controller → host

The `reqTest` Message

This message puts the controller in a test mode instead of a knitting mode.

- Python: `TestRequest`
- Arduino: h_reqTest
- table: reqTest
- requests answer: The cnfTest Message
- direction: host → controller

The `cnfTest` Message

This messsage confirms whether the controller is in the test mode. If success is indicated, the controller sends The indState Message messages periodically, containing the sensor and position values.

- Python: `TestConfirmation`
- Arduino: h_reqTest
- table: cnfTest
- answers: The reqTest Message
- direction: controller → host

References

See also

- the original specification
- the `hardware messages module` for messages sent by the hardware

- the `host messages module` for messages sent by the host
- a discussion about the specification