

IPC-HERMES-9852

The global standard for "M2M" in SMT assembly

IPC-HERMES-9852

IPC-HERMES-9852

The Global Standard for Machine-to-Machine Communication in SMT Assembly

Version 1.5

Contributing companies:

4IR.UK

6TL Engineering

Achat Engineering GmbH

allSMT

ASM Assembly Systems GmbH

Asscon

ASYS Automatisierungssysteme GmbH

BESI

Bright Machines

BTU

CKD

CTI Systems

CTS

CYBEROPTICS

Digitaltest

ECD

Essemtec

EUNIL

Eunil Co., Ltd.

Exelsius

Famecs

FlexLink

GKG

GÖPEL electronic GmbH

Hanwha

Heller Industries

Innomelt

IPTE

IBL-Löttechnik GmbH

ITW EAE

JAPAN UNIX Co. Ltd.

JOT Automation Kft.

Keysight Technologies

KIC

KOH YOUNG Technology Inc.

kolb Cleaning Technology GmbH

Kulicke & Soffa

Kurtzrsa

Magic Ray Technology

MIRTEC

MYCRONIC

Nordson ASYMTEK & MATRIX

Nutek Europe B.V.

OMRON Corporation

OSAI

PARMI

Pemtron

Rehm Thermal Systems GmbH

Rejoint

RG Elektrotechnologie

SEICA SpA & SEICA Automation

SAKI Corp

Scheid IT

SEHO Systems

SICK AG

SMT-Wertheim

SolderStar

Sonic Technology

SPEA S.p.A.

SYNEO

Test Research, Inc. (TRI)

Takaya

Universal Instruments

VISCOM AG

ViTrox

YJ Link Co., Ltd.

YXLON



Contents:

1	SCOPE	5
2	TECHNICAL CONCEPT	6
2.1	Prerequisites	6
2.2	Board IDs	6
2.3	Machine-to-Machine Communication (Horizontal Channel)	6
2.3.1	Topology	6
2.3.2	Connecting, Handshake and Detection of Connection Loss	7
2.3.3	Normal Operation.....	9
2.3.4	Transport Error Handling	10
	Scenario U1a	11
	Scenario U1b	12
	Scenario U2	13
	Scenario U3	14
	Scenario D1	15
	Scenario D2	16
	Scenario D3	17
2.3.5	Handling of BoardForecast	18
	Scenario 1	18
	Scenario 1 (error handling)	20
	Scenario 2.....	21
2.3.6	Protocol States and Protocol Error Handling	22
2.3.7	Handling of Attribute 'Route'	23
2.3.8	Handling of Attribute 'Action'	23
2.4	Remote Configuration	24
2.4.1	Topology	24
2.4.2	Remote Configuration	24
2.5	Communication With Supervisory System (Vertical Channel)	24
2.5.1	Topology	24
2.5.2	Connecting, Handshake and Detection of Connection Loss	24
2.5.3	Protocol States and Protocol Error Handling	27
3	MESSAGE DEFINITION	28
3.1	Message Format	28
3.2	Root Element.....	28
3.3	CheckAlive	29



3.4	ServiceDescription	29
3.5	Notification.....	31
3.6	BoardAvailable	31
3.7	RevokeBoardAvailable	35
3.8	MachineReady	35
3.9	RevokeMachineReady	36
3.10	StartTransport.....	36
3.11	StopTransport	37
3.12	TransportFinished.....	37
3.16	BoardForecast	37
3.17	QueryBoardInfo	40
3.18	SendBoardInfo.....	41
3.19	SetConfiguration	43
3.20	GetConfiguration	45
3.21	CurrentConfiguration	45
3.22	SupervisoryServiceDescription.....	46
3.23	BoardArrived.....	46
3.24	BoardDeparted	50
3.25	QueryWorkOrderInfo	53
3.26	SendWorkOrderInfo.....	54
3.27	ReplyWorkOrderInfo	57
3.28	Command	58
4	Appendix.....	60
4.1	Special Scenarios.....	60
4.1.1	Board Tracking When Board Is Torn Out From the Line	60
4.1.2	Board Tracking When Board Is Temporarily Removed From the Line	61
4.1.3	Board Tracking When Board Was Transferred Without Data.....	62
4.1.4	Oven Error Loop	64
4.1.5	Request Pause / Confirm Pause and Resume Operation	65
4.1.6	Board Removal at Downstream Conveyor	67
4.1.7	Reversal Transportation to a Flipping Unit Located Downstream a Process Machine ...	68
4.1.8	Reversal Transportation to a Flipping Unit Located Upstream a Process Machine	69
4.1.9	Board Routing Within a Production Line by Predefined Routes	70
4.1.10	Board Routing Within a Production Line Towards Target Locations	70





4.2	Glossary / Abbreviations	71
4.3	References	72
4.4	History	72



1 SCOPE

The aim of this specification is to create a state-of-the-art communication protocol for handling board transfers and associated data at surface-mount technology (SMT) production lines. Therefore, this new communication protocol has to cope with the following:

- Replace the electrical SMEMA interface as specified in IPC-SMEMA-9851
- Extend the interface to communicate:
 - Unique identifiers for the handled printed circuit boards (PCBs)
 - Equipment identifiers of the first machine noticing a PCB
 - Barcodes
 - Conveyor speed and intended board route
 - A lightweight digital twin of the product containing e.g.,
 - Product type identifier
 - Length
 - Width
 - Thickness
 - Board state

With respect to version numbers The Hermes Standard adheres to the rules of Semantic Versioning 2.0.0 [SemVer_2.0.0].

Hints on naming:

- Wherever a feature is described by the word "**shall**" it is mandatory.
- The word "machine" is used for any equipment which can be found in a SMT production line (e.g., printers, placement machines, ovens, AOIs, transport modules, shuttles, stackers).
- The term "PCB" may also refer to carriers transporting PCBs.
- The word "Hermes" is used as abbreviation for "The Hermes Standard".
- "The Hermes Standard" and IPC-HERMES-9852 are synonymys for the standard specified in this document and might be used interchangeably.



2 TECHNICAL CONCEPT

2.1 Prerequisites

This specification is based on the prerequisite, that any application implementing this protocol has to provide connectivity based on Internet Protocol (IP) [IETF_RFC_791] / [IETF_RFC_2460] via Transmission Control Protocol (TCP) [IETF_RFC_793] (ISO / OSI model [ISO_7498-1] layer 3) to the adjacent machines machines and for communication with supervisory systems.

2.2 Board IDs

Board individuals are identified by board IDs. These must be Globally Unique Identifiers (GUIDs) according to [ITU-T_REC_X.667], e.g., 123e4567-e89b-12d3-a456-426655440000. They are generated by the first machine in a consecutive row of machines implementing the Hermes protocol. The board ID is passed from machine to machine. If a machine in a line does not implement the Hermes protocol, the board ID is lost and a new one will be generated by the next machine implementing Hermes.

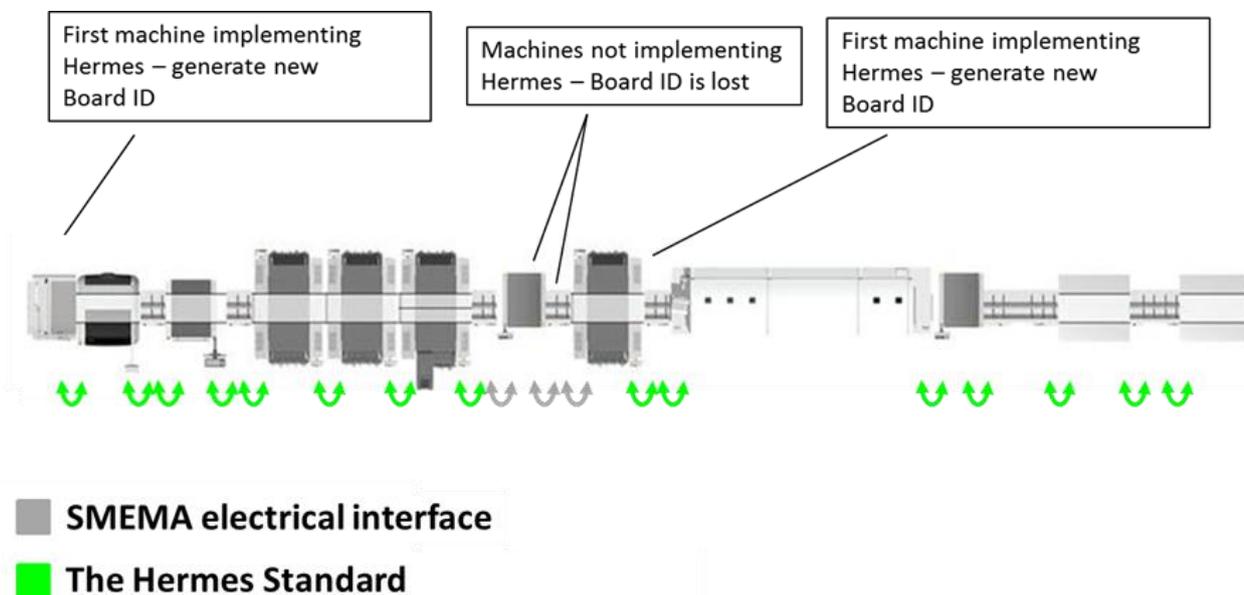


Figure 1 Generation of Board IDs

2.3 Machine-to-Machine Communication (Horizontal Channel)

2.3.1 Topology

Any machine in a line offers one TCP server per lane on its downstream side. Further servers per lane might also be necessary (e.g., if reverse transportation is supported). The TCP port number is not specified but can be configured by the user. The recommended port numbers are 50100 plus lane identifier (ID) with lanes being enumerated looking downstream from right to left beginning with 1 (i.e., for the left lane of a dual lane machine,



the upstream machine server accepts connections on port 50102). For every further server, plus 10 is recommended to be added to the port number.

The downstream machine opens one connection for every lane and every transportation interface on its upstream side to the upstream machine(s). So every PCB handover point corresponds to one TCP connection per exchange direction.

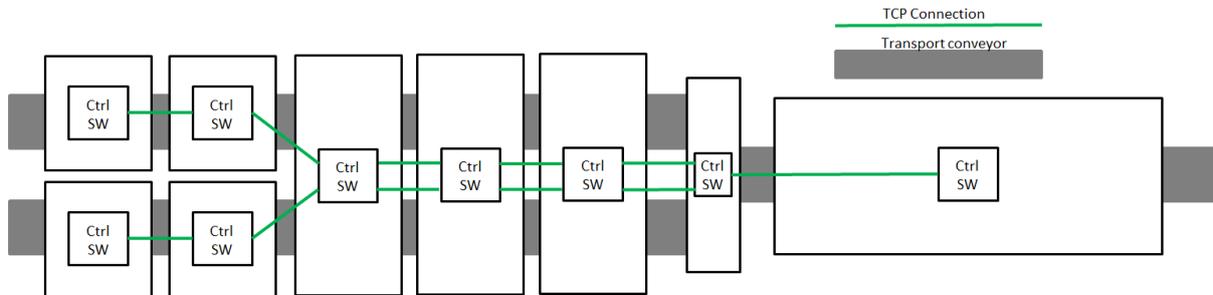


Figure 2 TCP Connections in a Line

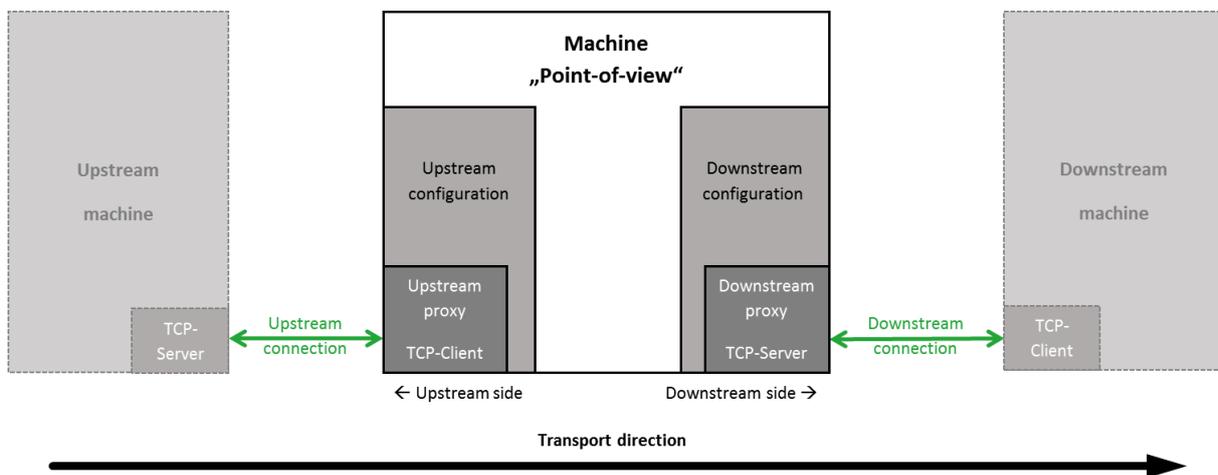


Figure 3 Upstream and Downstream From the Perspective of the Machine

2.3.2 Connecting, Handshake and Detection of Connection Loss

After booting, the downstream machine starts cyclic connection attempts to the configured upstream machines. When a connection is established, the downstream machine starts sending a ServiceDescription message whereupon the upstream machine answers with its own ServiceDescription. This ServiceDescription message contains the lane ID and interface ID (optional) of the sending machine related to this TCP connection. It also contains the implemented version and a list of all optional features and additional features of a higher version which are implemented by the machine. The features of the Hermes specification version 1.0 have to be supported by any implementation.



If a downstream machine is already connected to the lane and the transportation interface, this connection will be retained. A Notification message **shall** be sent to the new connection before it is closed.

After exchanging the handshake messages, both machines may begin to send BoardAvailable / MachineReady messages (see 2.3.3).

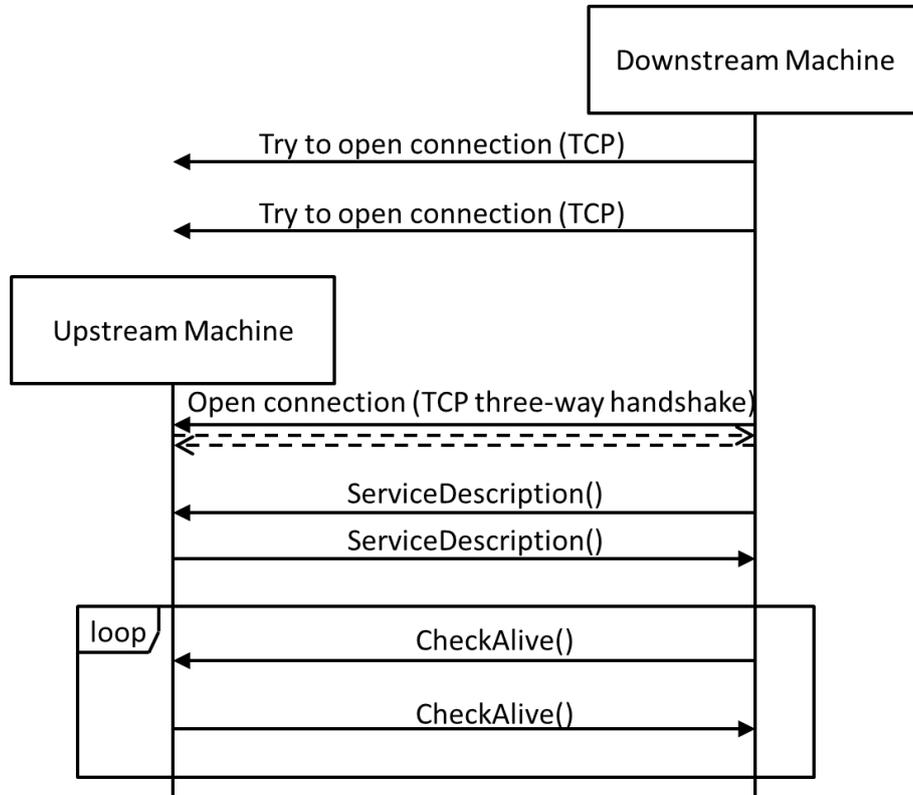


Figure 4 Connection, Handshake and Connection Loss Detection on Horizontal Channel

The connections are kept open all the time. As TCP by itself does not detect connection losses (“half-open connections” caused by e.g., process- / computer crash, unplugged network cables) both sides of a connection have to send cyclic CheckAlive messages. Those messages do not have to be answered by the remote side—the TCP stack will detect a connection loss when trying to send the packet. If the server detects a connection loss, it ends the connection and waits for a new connection by the client. If the client detects a connection loss, it ends the connection and re-starts with cyclic connection attempts.

As not all TCP stacks recognize correctly the loss of connection when sending messages it is possible to extend the implementation of this functionality to an exchange of CheckAlive messages. Machines which have implemented this function do have the tag FeatureCheckAliveResponse in the ServiceDescription. The exchange of CheckAlive messages then works like shown in Figure 5.



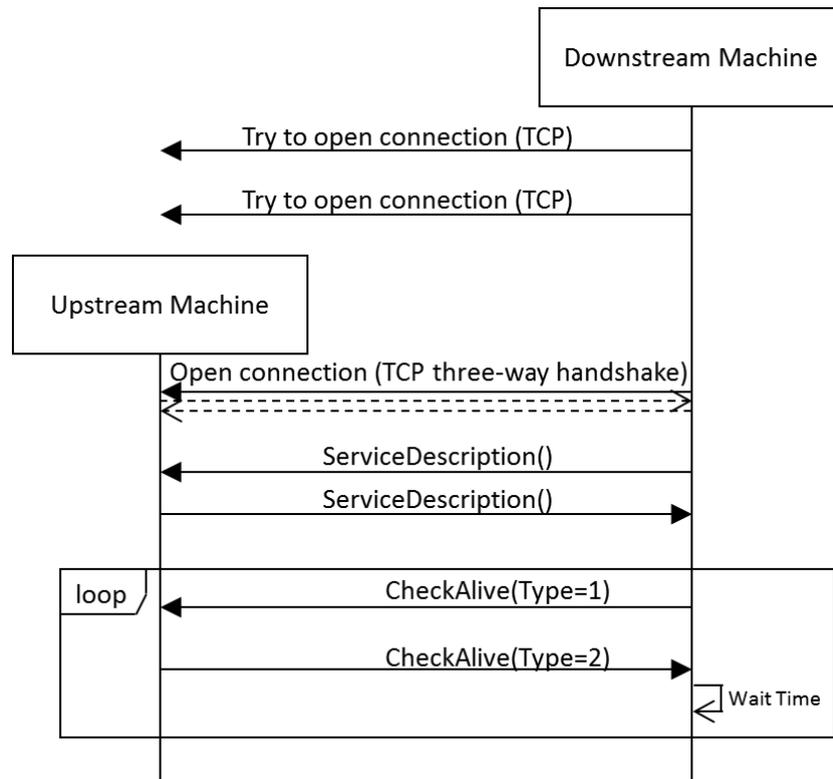


Figure 5 Example for Connection Loss Detection With FeatureCheckAliveResponse on Horizontal Channel

One of the machines (in the figure, the downstream machine but it could be also the upstream machine) sends a (ping) CheckAlive message, that is a CheckAlive message with the attribute Type set to 1. The peer machine then responds immediately with a (pong) CheckAlive message, that is a CheckAlive message with the attribute Type set to 2 and the Id matching the Id of the (ping) CheckAlive message.

A missing response (It is recommended to wait for 3 seconds.) indicates a connection loss.

2.3.3 Normal Operation

When an upstream machine has a PCB available for handover, it sends a BoardAvailable message, while a downstream machine ready to accept a PCB sends a MachineReady message. The naming of these messages is inspired by the electrical SMEMA interface. However, the messages do not represent the state of a machine's interface directly but are events for initiating a PCB handover.



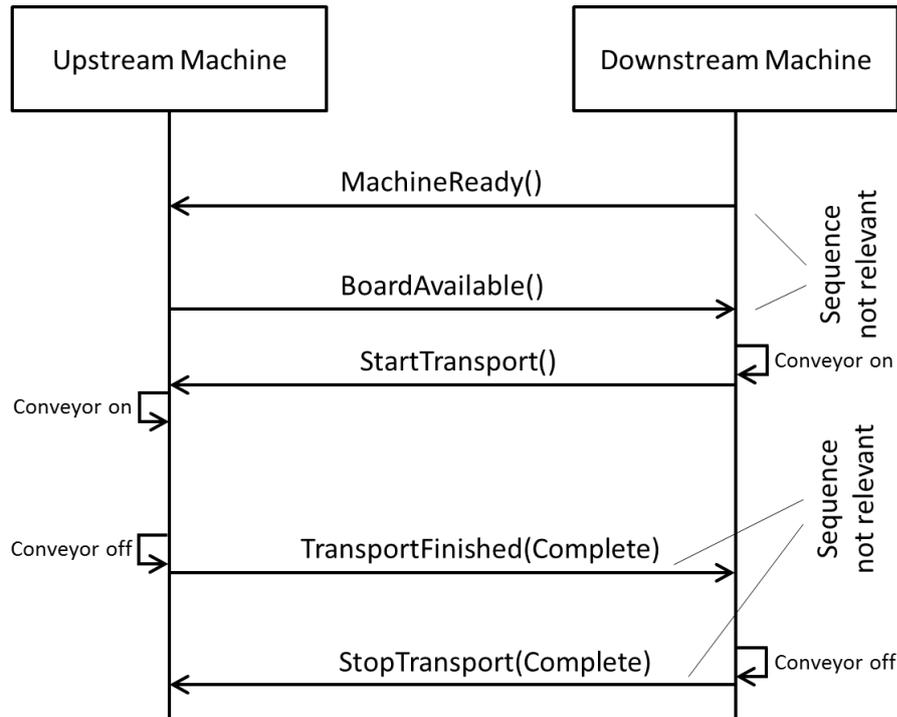


Figure 6 Communication Sequence for Board Transport

When both machines have indicated their readiness to handover the PCB, the downstream machine initiates the transfer by switching on its conveyor and sending the `StartTransport` message. Upon receiving this message, the upstream machine switches on its conveyor and the PCB moves into the downstream machine.

When the upstream machine is able to state that the PCB has fully left the machine, it sends the `TransportFinished` message. When the downstream machine has full control of the board, it sends the `StopTransport` message. The handover of a PCB is finished and is ready to start over.

If the upstream machine receives a `StopTransport` message and has not sent the `TransportFinished` message yet, it has to stop its conveyor and send the `TransportFinished` message.

The `MachineReady` message does not trigger an action on one of the machines directly. However, it still is necessary to realize machines like e.g., shuttles which have to react to the availability of their downstream machines.

2.3.4 Transport Error Handling

To keep this protocol hardware independent, the handling of transport errors is described based on a very simple model of the board handover. The handover process is structured into the three phases:

- a) NotStarted: The board is fully inside the upstream machine.
- b) Incomplete: The board is partly inside both machines.
- c) Complete: The board is fully inside the downstream machine.



Any state or event which prevents one or both machines from handing over a PCB is interpreted as an error. An error may be detected by any of the machines in any of the three handover phases. It is up to the application how to detect the current handover phase, how to detect errors and how to solve them eventually (e.g., sensors, model based prediction, timeouts, user interaction).

The following sequence charts give an overview of the communication within this protocol depending on the machine which detects the error and the phase in which it is detected. The point in the sequence where the error is detected is marked by the following symbol: 

Scenario U1a

- Error detected by the upstream machine
- PCB fully inside the upstream machine
- Error detected before StartTransport has been received

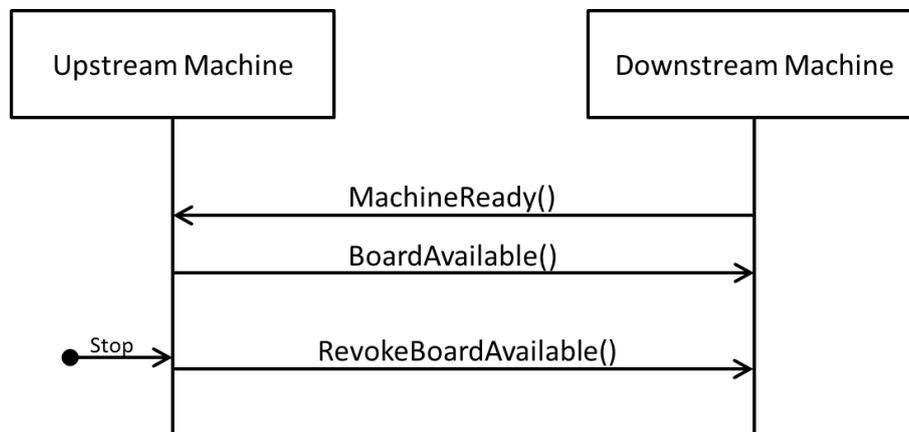


Figure 7 Communication Sequence in Scenario U1a

Error detection: The error is detected before any transport started.

Reaction on upstream machine: The upstream machine sends a RevokeBoardAvailable message.

Reaction on downstream machine: None.

Resolution: After the error is solved, the regular transport sequence can start from the beginning.

Scenario U1b

- Error detected by the upstream machine
- PCB fully inside the upstream machine
- Error detected after StartTransport has been received

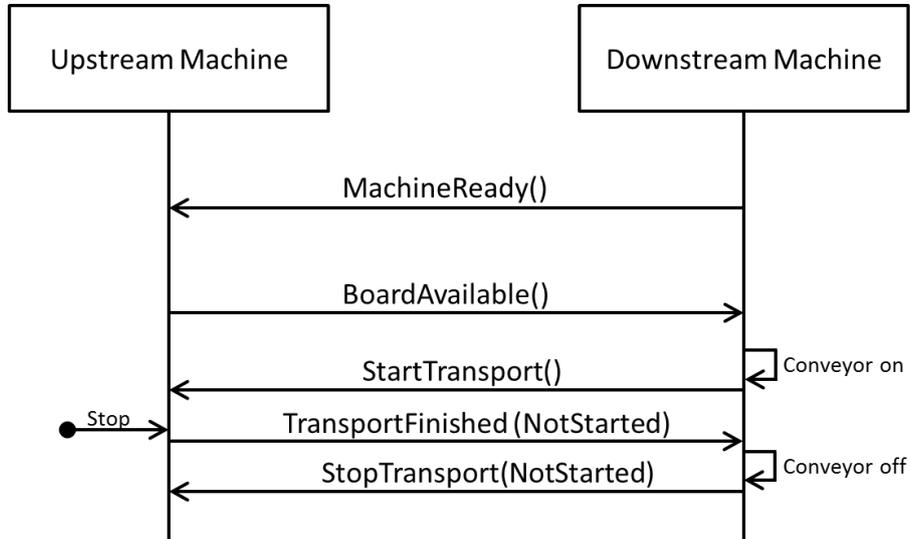


Figure 8 Communication Sequence in Scenario U1b

Error detection: The error is detected after the downstream machine started its conveyor and has sent the StartTransport message.

Reaction on upstream machine: The upstream machine sends a TransportFinished message indicating that it has not started the transport.

Reaction on downstream machine: Upon the TransportFinished message, the downstream machine stops its conveyor and sends a StopTransport message indicating that no transport has started.

Resolution: After the error is solved, the regular transport sequence can start from the beginning.

Scenario U2

- Error detected by the upstream machine
- PCB partly inside both machines

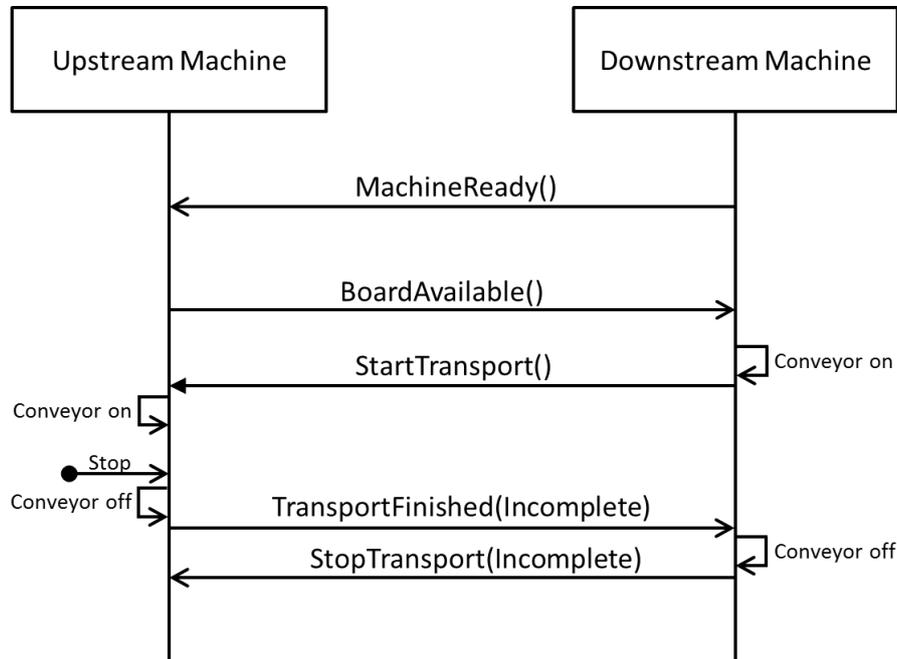


Figure 9 Communication Sequence in Scenario U2

Error detection: The error is detected after both machines started their conveyors. The upstream machine assumes that the PCB may have partly entered the downstream machine.

Reaction on upstream machine: The upstream machine sends a TransportFinished message indicating that the PCB might be located between the machines.

Reaction on downstream machine: Upon the TransportFinished message, the downstream machine stops its conveyor and sends a StopTransport message indicating the state of the PCB handover. Note that in Figure 9 the StopTransport message is represented with parameter "Incomplete". However, in this scenario, the downstream machine could send any of the allowed transport states.

Resolution: After the error is solved, the regular transport sequence can start from the beginning. The regular transport message sequence also applies to a PCB located between the two machines.



Scenario U3

- Error detected by the upstream machine
- PCB fully inside the downstream machine

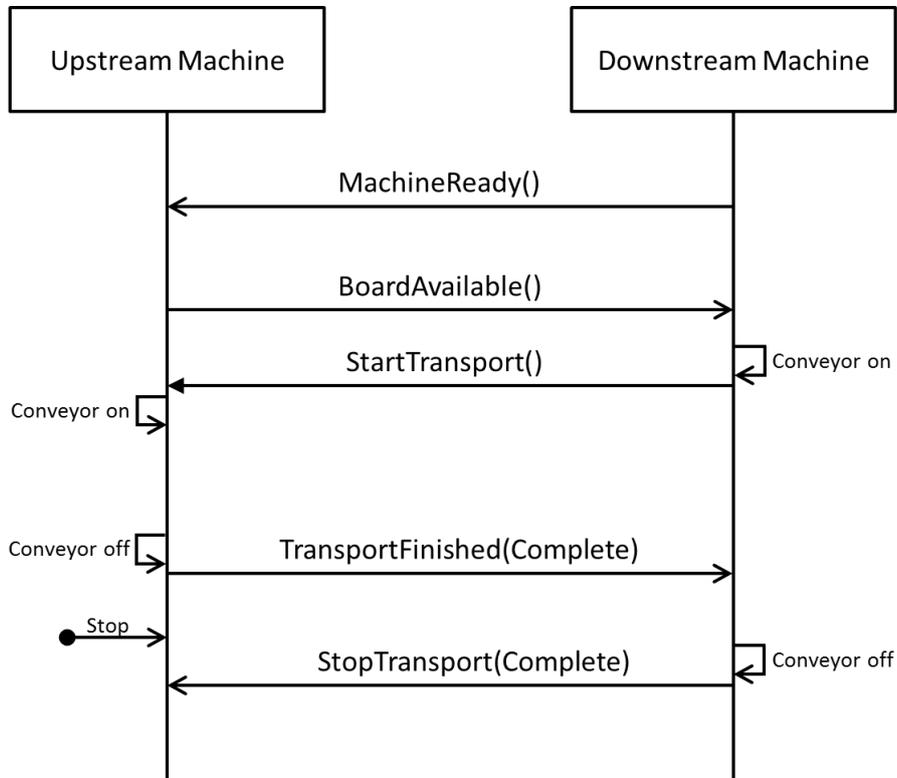


Figure 10 Communication Sequence in Scenario U3

Error detection: The error is detected after the PCB is fully inside the downstream machine.

Reaction on upstream machine: None. Although the machine detected an error, it is irrelevant for the handover process.

Reaction on downstream machine: None. The downstream machine is not aware of any error.

Resolution: This scenario is irrelevant for the Hermes protocol. It is just listed for completeness.



Scenario D1

- Error detected by the downstream machine
- PCB fully inside the upstream machine
- Error detected before StartTransport has been sent

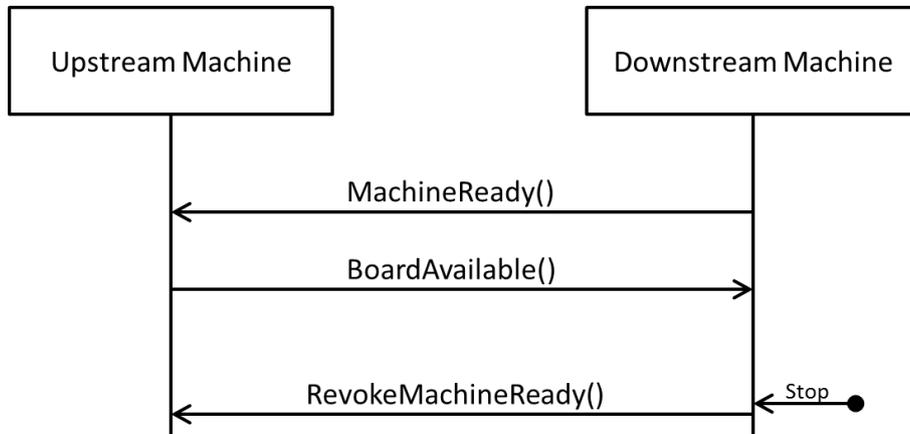


Figure 11 Communication Sequence in Scenario D1

Error detection: The error is detected before any transport started.

Reaction on upstream machine: None.

Reaction on downstream machine: The downstream machine sends a RevokeMachineReady message.

Resolution: After the error is solved, the regular transport sequence can start from the beginning.



Scenario D2

- Error detected by the downstream machine
- PCB partly inside both machines

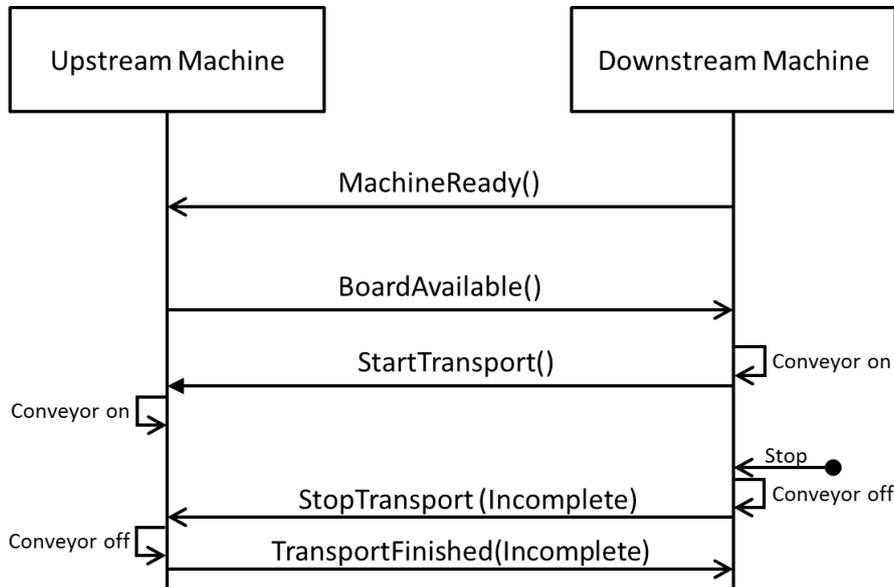


Figure 12 Communication Sequence in Scenario D2

Error detection: The error is detected after both machines started their conveyors. The downstream machine assumes that the PCB may already have entered its conveyor.

Reaction on upstream machine: Upon the StopTransport message from the downstream machine, the upstream machine stops its conveyor and sends a TransportFinished message indicating the state of the PCB handover. Note that in Figure 12 the TransportFinished message is represented with parameter "Incomplete". However, in this scenario, the upstream machine could send any of the allowed transport states.

Reaction on downstream machine: The downstream machine stops its conveyor and notifies the upstream machine of the error by sending a StopTransport message indicating an incomplete PCB handover.

Resolution: After the error is solved, the regular transport sequence can start from the beginning. The regular transport message sequence also applies for a PCB located in between the two machines.

Scenario D3

- Error detected by the downstream machine
- PCB fully inside the downstream machine

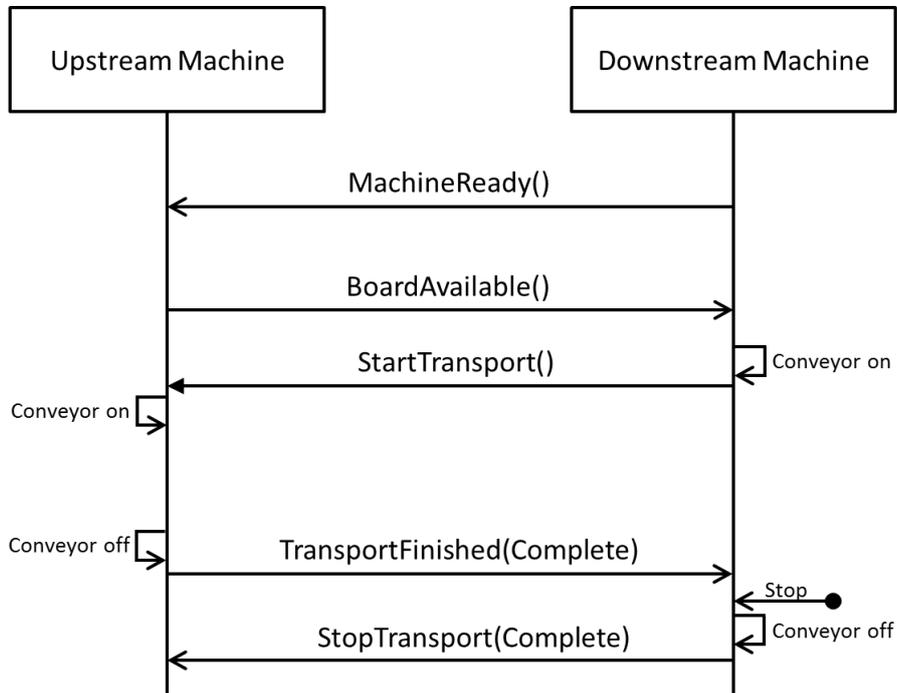


Figure 13 Communication Sequence in Scenario D3

Error detection: The error is detected after the PCB is fully inside the downstream machine.

Reaction on upstream machine: None. The upstream machine is not aware of any error.

Reaction on downstream machine: None (at least in the scope of this protocol).

Resolution: This scenario is irrelevant for the Hermes protocol. As transport sequences are always initiated by the downstream machine sending StartTransport, trouble-shooting (possibly including running the conveyor of the downstream machine) can be executed independently from the upstream machine.



2.3.5 Handling of BoardForecast

Among others the BoardForecast may be used in following scenarios:

- Scenario 1: Anticipating a product change without a board (e.g., because upstream machine does not have stoppers / conveyor that can be stopped).
- Scenario 2: Sending an estimated time to downstream machine until a board will be available (e.g., to allow downstream machine to choose between several upstream machines to get next available board).

Scenario 1

Upstream machine is processing a changeover (new product type) and wants to ensure that the downstream machine is simultaneously also processing a changeover. Upstream machine also needs to check that this actually happens. It sends a BoardForecast with a (forecast-)ID, to which the downstream machine at some point must respond with a MachineReady with the same ID. Upon receiving this MachineReady, the upstream machine can assume that the product change was successful.

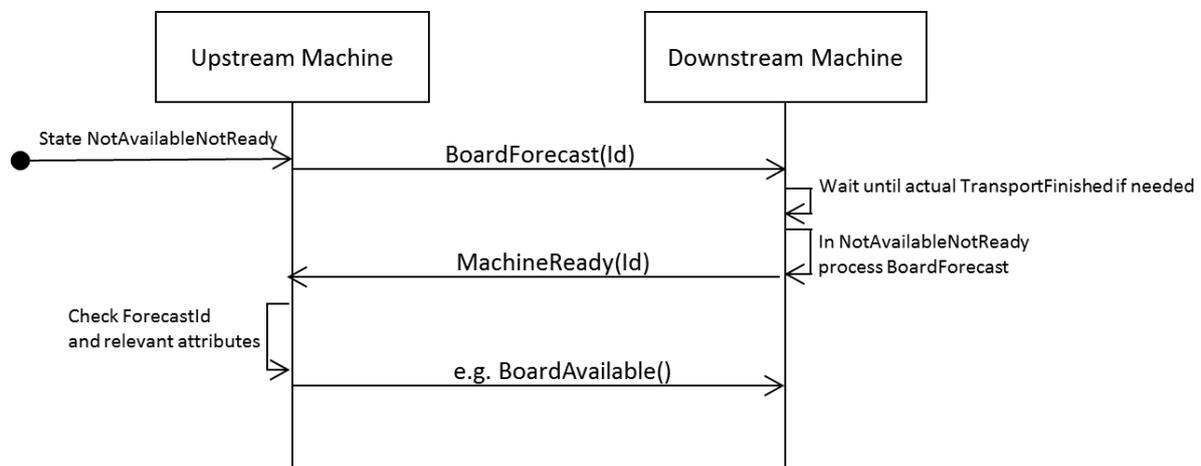


Figure 14 Example of Communication Sequence for BoardForecast

Note: If starting the BoardForecast handling in the state MachineReady, the downstream machine must send a RevokeMachineReady message (see Figure 15).

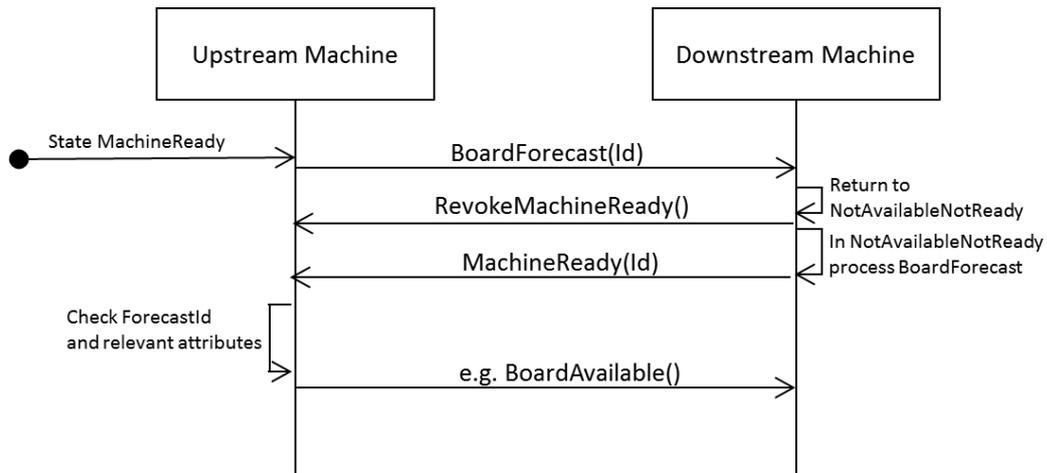


Figure 15 Example of Communication Sequence for BoardForecast With RevokeMachineReady

If several BoardForecast messages (e.g., with different ProductTypeId) are sent in a short delay, the downstream machine may process only the last BoardForecast message:

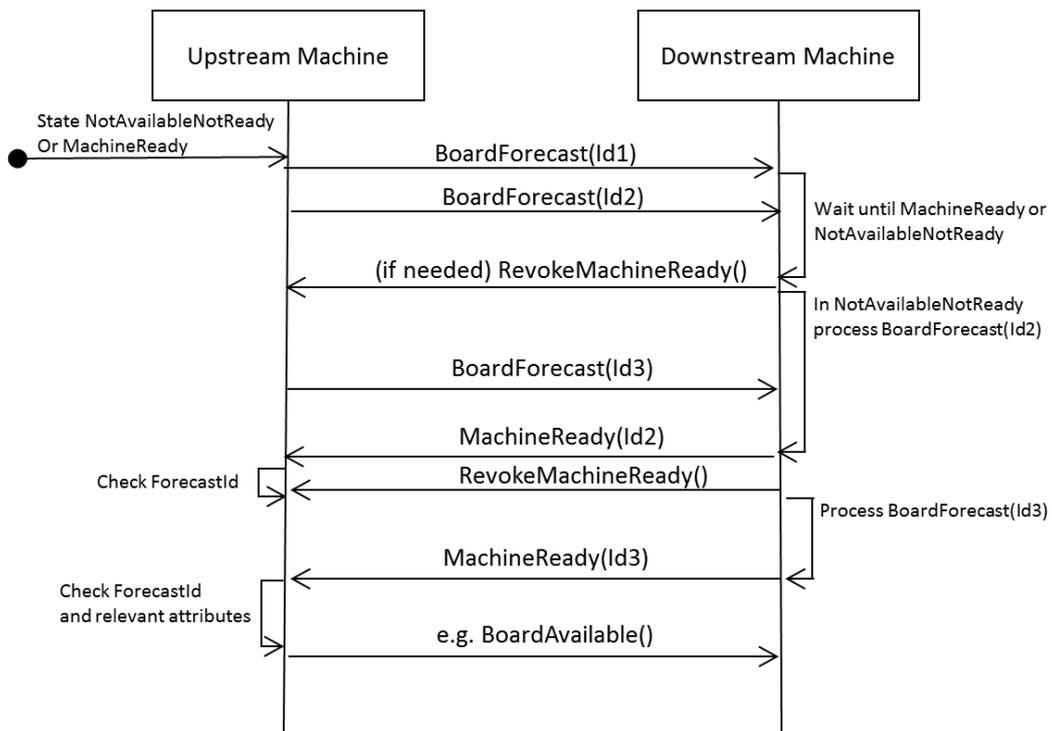


Figure 16 Example of Communication Sequence With Several BoardForecast



Scenario 1 (error handling)

If the downstream machine cannot accept the product exchange (e.g., unknown ProductId or width is physically impossible in machine) it will respond after a `RevokeMachineReady` with a notification of type "BoardForecastError". The upstream machine must then do some error handling (e.g., ask operator if machine should retry the `BoardForecast` or if the operator wants to remove the board).

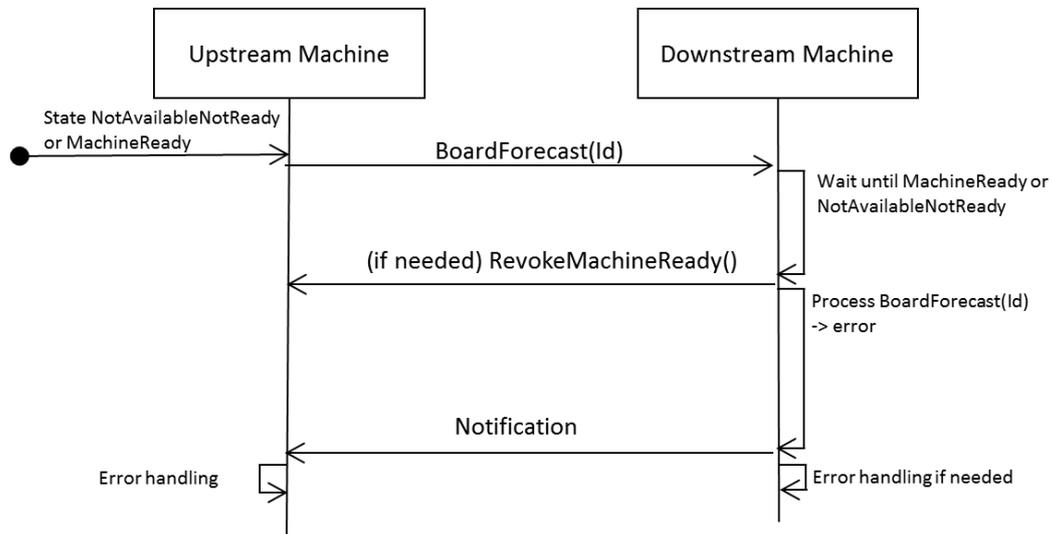


Figure 17 Example of Communication Sequence in Case With Error Handling

Scenario 2

As BoardForecast in that case usually only gives some information to the downstream machine, several BoardForecast may be sent. However, error handling or checking are not needed on the side of the downstream machine. In that scenario ForecastId will not be sent.

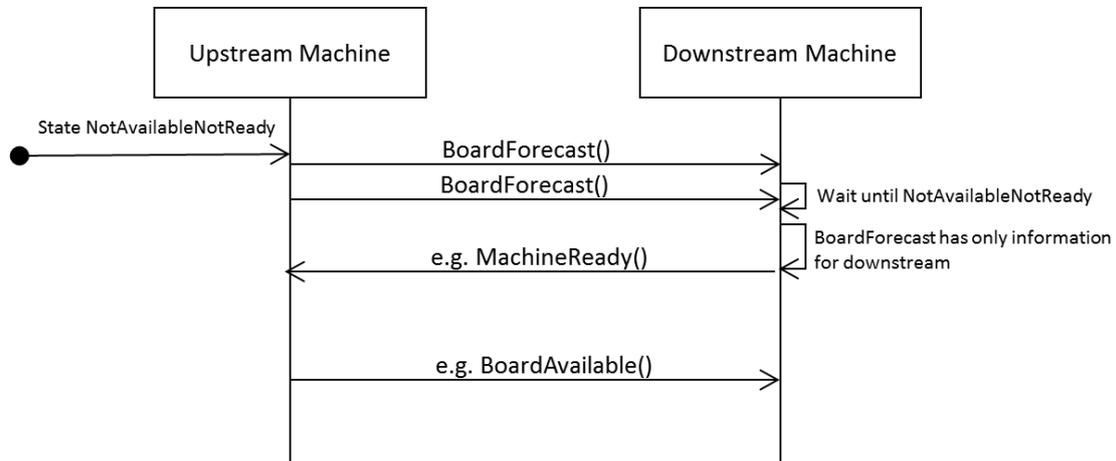


Figure 18 Example of Communication Sequence BoardForecast Without Product Change

Note: The function of BoardForecast is optional. If FeatureBoardForecast is specified in the ServiceDescription, it must be fully supported. Otherwise it can be ignored.



2.3.6 Protocol States and Protocol Error Handling

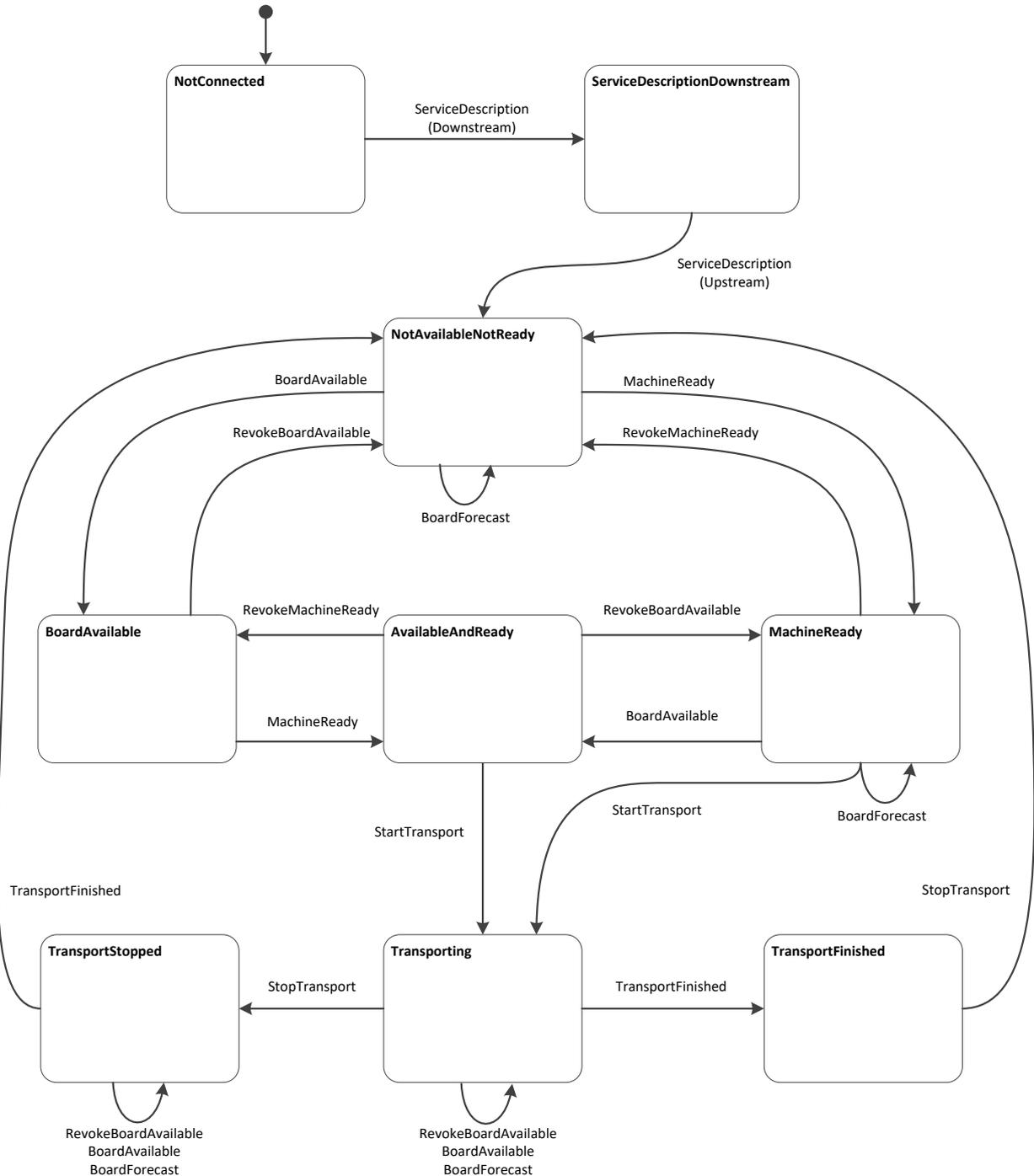


Figure 19 Hermes Interface States on Horizontal Channel

Figure 19 lists all states and transitions of a Hermes interface corresponding to the machine-to-machine (M2M) communication. The state is the comprehensive state of the interface rather than the state of one of the involved machines.



The messages may only be sent if they trigger the corresponding transition shown in the state chart. Any message defined in this standard, except "Notification", "CheckAlive", "QueryBoardInfo", "SendBoardInfo" and "Command", which is received not triggering a transition is interpreted as a protocol error (e.g., a MachineReady message when the interface is in the state Transporting). In case of a protocol error, any running transport **shall** be stopped and the connection is terminated. The interface may start over with a new connection. Any unknown message, which is received, **shall** be ignored and discarded to keep upward compatibility.

Note that due to race conditions, a RevokeBoardAvailable message may overlap with a StartTransport message or even a StopTransport message, so this **shall** not be treated as a protocol error (transition from MachineReady to Transporting and self-transitions on Transporting and TransportStopped).

2.3.7 Handling of Attribute 'Route'

Usually a machine in the production line knows its task by manual configuration, by analyzing the properties of the next coming board, or by an ERP/MES system (vertical communication). However, in some cases it is necessary for process machines to control adjacent transport facilities. The horizontal M2M-communication consequently provides a way to forward such instructions along with the board transfer. This option also makes it possible to keep transport machine simple.

Route is an optional attribute inside the messages 3.6 BoardAvailable, 3.18 SendBoardInfo, 3.23 BoardArrived, 3.24 BoardDeparted and 3.26 SendWorkOrderInfo. A Route number can guide the course of the board within a production line either on a set path or towards a specific destination machine. The concept for board routing must always be defined as part of a production line planning in collaboration with all machines involved. Special cases like reverse transportation or manual removal of the board are also possible. Some common scenarios are described in the appendix 4.1.

If **Route** is received from upstream and insignificant or unknown to the machine it **shall** be forwarded unaltered downstream. But the machine can react to it or set a new route, dependent on its own available function and settings.

2.3.8 Handling of Attribute 'Action'

Action is an optional attribute inside the messages 3.6 BoardAvailable, 3.23 BoardArrived and 3.24 BoardDeparted. It is also intended to control adjacent simple transport facilities. Action refers to a real board transfer and therefore it is not included in messages of data-without-board-transfer. The concept for being initiator or executor of actions must always be defined as part of a production line planning in collaboration with all machines involved. Some common scenarios are described in the appendix 4.1.

If **Action** is received from upstream and insignificant or unknown to the machine it **shall** be forwarded unaltered downstream. When an **Action** is executed by a machine, it **shall** not be forwarded downstream or it **shall** be passed with 0. However, the machine can initiate a new **Action** to be performed downstream if needed.



2.4 Remote Configuration

2.4.1 Topology

Although a machine may offer the possibility to configure the Hermes TCP port(s) and the IP address(es) of its upstream machine(s) locally (e.g., via a graphical user interface of the machine controller), every machine implementing this protocol **shall** offer a possibility to configure these properties remote via TCP. Therefore, the machine **shall** offer a TCP server on port 1248 on at least one network adapter where it accepts configuration messages (see 3.19 to 3.21 for detailed information).

The configuration system opens a connection to each required machine. The connection **shall** only be kept open as long as needed and closed by the configuration system.

2.4.2 Remote Configuration

A SetConfiguration message **shall** contain the full configuration for all Hermes interfaces of a machine. Any existing configuration is overwritten when a SetConfiguration message is received. Whenever a configuration is not applicable (e.g., bad IP address format), the SetConfiguration message is answered with a Notification message (see 3.5). Every time the configuration is changed, affected open Hermes connections will be reset at the next appropriate moment.

It is possible to read the current configuration through the GetConfiguration message answered by a CurrentConfiguration message. The configuration **shall** be persisted until it is changed.

2.5 Communication With Supervisory System (Vertical Channel)

2.5.1 Topology

Any machine in a line **shall** offer one TCP server on the configured supervisory system port on at least one network adapter where it accepts connections from supervisory systems. The used supervisory system port can be retrieved via GetConfiguration. The connection to the supervisory system is e.g., used to allow the configuration of the Hermes connections to the upstream and downstream machine(s) remotely without relying on the capabilities of the machine user interface.

The supervisory system opens a connection to each required machine. The connection **shall** only be kept open as long as needed and closed by the supervisory system.

Note: It is possible to use the same port for the communication with a supervisory system as for the remote configuration.

2.5.2 Connecting, Handshake and Detection of Connection Loss

Upon demand the supervisory system starts cyclic connection attempts to the required machine. When a connection is established, the supervisory system starts sending a SupervisoryServiceDescription message whereupon the machine answers with its own SupervisoryServiceDescription. This



SupervisoryServiceDescription message contains a list of supervisory features which are implemented by the client.

If a new supervisory system tries to connect and no further connections are supported by the machine, the already established connections will be retained. A Notification message **shall** be sent to the new connection before it is closed.

After exchanging the handshake messages, both communication partners may begin to exchange the messages belonging to supervisory features supported by both communication partners.

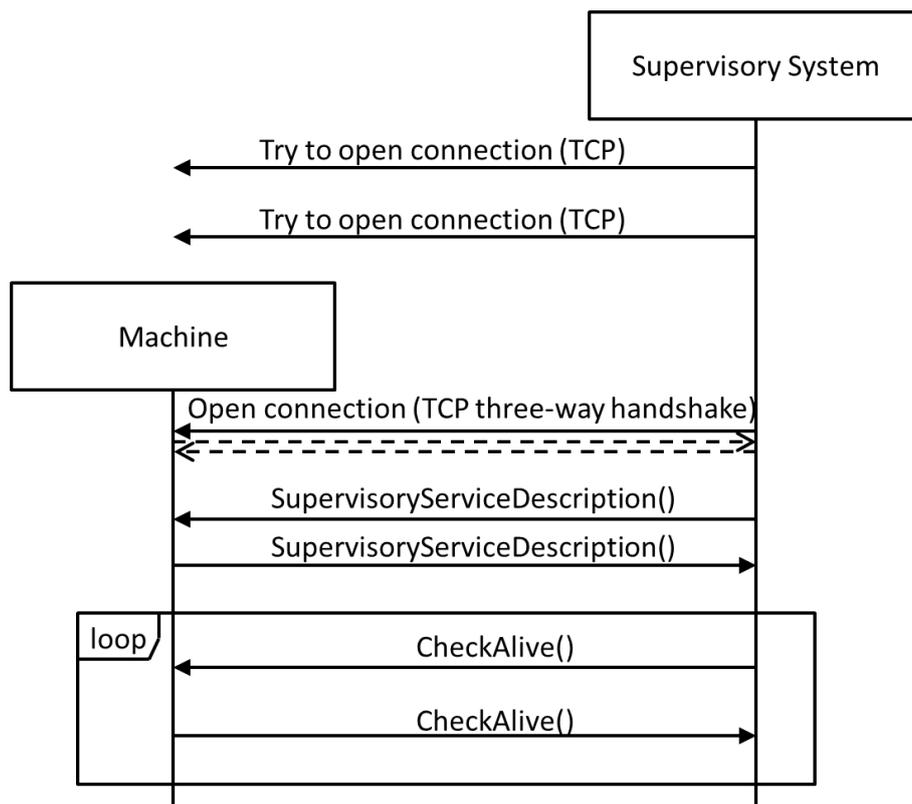


Figure 20 Connection, Handshake and Connection Loss Detection on Vertical Channel

The connections are kept open as long as needed. As TCP by itself does not detect connection losses ("half-open connections" caused by e.g., process- / computer crash, unplugged network cables) both sides of a connection have to send cyclic CheckAlive messages. Those messages do not have to be answered by the remote side – the TCP stack will detect a connection loss when trying to send the packet. If the server detects a connection loss, it ends the connection and waits for a new connection by the client. If the client detects a connection loss, it ends the connection and re-starts with cyclic connection attempts.



As not all TCP stacks recognize correctly the loss of connection when sending messages it is possible to extend the implementation of this functionality to an exchange of CheckAlive messages. Machines which have implemented this function do have the tag FeatureCheckAliveResponse in the SupervisoryServiceDescription.

The exchange of CheckAlive messages then works like shown in Figure 21.

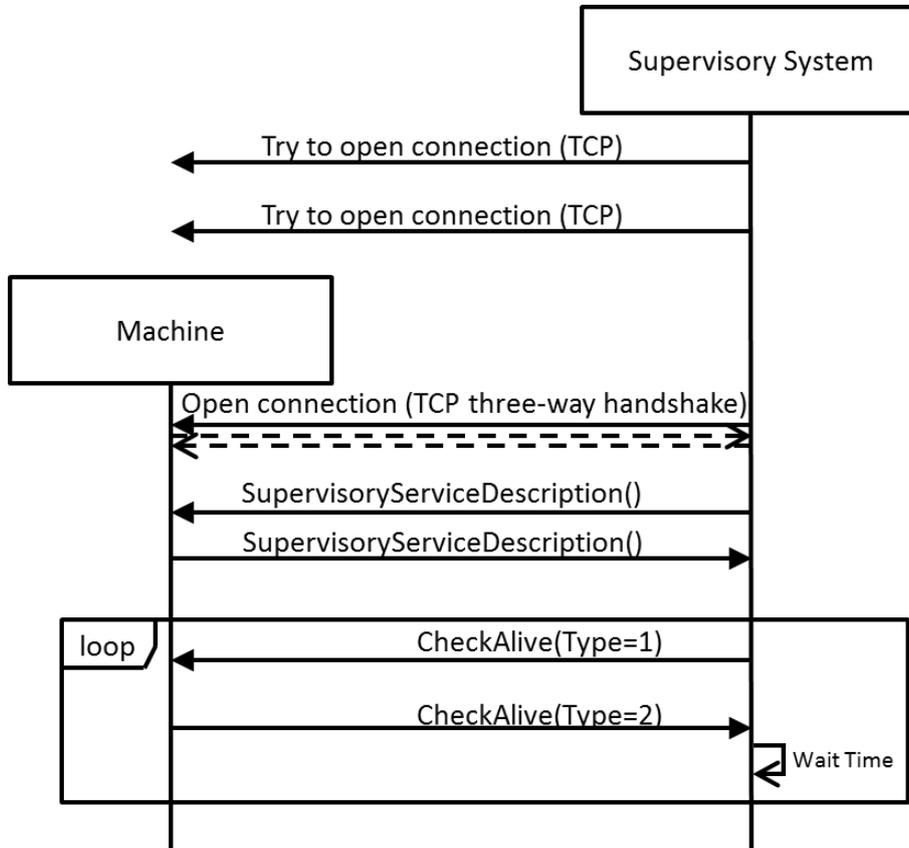


Figure 21 Example for Connection Loss Detection With FeatureCheckAliveResponse on Vertical Channel

One of the communication partners (in the figure the supervisory system but it could be also the machine) sends a (ping) CheckAlive message, that is a CheckAlive message with the attribute Type set to 1. The peer communication partner then responds immediately with a (pong) CheckAlive message, that is a CheckAlive message with the attribute Type set to 2 and the Id matching the Id of the (ping) CheckAlive message.

A missing response (It is recommended to wait for 3 seconds.) indicates a connection loss.



2.5.3 Protocol States and Protocol Error Handling

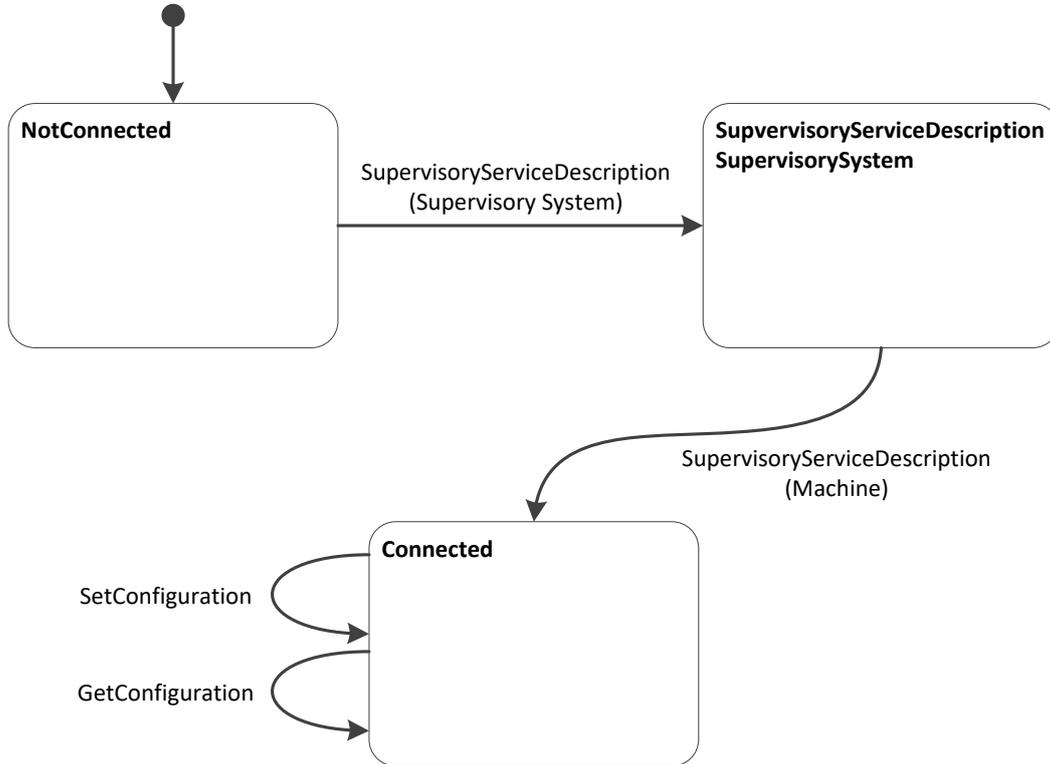


Figure 22 Hermes Interface States on Vertical Channel

Figure 22 lists all states and transitions of a Hermes interface corresponding to the communication with supervisory systems. The state is the comprehensive state of the interface rather than the state of one of the involved communication partners.

The messages may only be sent if they trigger the corresponding transition shown in the state chart. Any message defined in this standard, except “Notification”, “CheckAlive”, “QueryWorkOrderInfo”, “SendWorkOrderInfo”, “Command” and “ReplyWorkOrderInfo”, which is received not triggering a transition is interpreted as a protocol error. In case of a protocol error the connection is terminated. The interface may start over with a new connection. Any unknown message, which is received, shall be ignored and discarded to keep upward compatibility.

3 MESSAGE DEFINITION

3.1 Message Format

Messages use the Extensible Markup Language (XML) format, where at least version 1.1 of XML shall be supported [W3C_XML_1.1].

For character encoding UTF-8 has to be used (No other encoding may be specified in the XML declaration). In the following sections of the document, for a better readable description of the XML data structures, tables are used instead of commonly used schema definitions.

Maximum size for every message is 64 kByte, i.e. 65536 bytes. For every string parameter there is either a fixed or minimum size that must be supported (individual values see tables).

In the tables, XML attributes are marked with the image “

The representation of data types (e.g., floating point numbers, boolean attributes) shall comply with the W3C XML schema recommendation [W3C_XML_Schema].

To keep upward compatibility, any message or attribute unknown by an implementation can be ignored and discarded.

3.2 Root Element

Every message is enveloped by a common root element with tag <Hermes>. The root element optionally includes a timestamp attribute with the following format (based on the W3C note “Date and Time Formats” [W3C_DATE_TIME]):

YYYY-MM-DDThh:mm:ss.s

where:

YYYY = four-digit year
MM = two-digit month (01=January, etc.)
DD = two-digit day of month (01 through 31)
hh = two digits of hour (00 through 23) (am / pm not allowed)
mm = two digits of minute (00 through 59)
ss = two digits of second (00 through 59)
s = one or more digits representing a decimal fraction of a second

The decimal fraction of the second **shall** be given with 3 digit precision.

The timestamp is optional and intended for diagnostic purposes only.

An example for a CheckAlive message would be:

```
<Hermes Timestamp="2017-07-16T19:20:30.452">  
  <CheckAlive />  
</Hermes>
```



A machine is not required to emit a precise timestamp, since this attribute is intended mainly for debugging purposes.

Recommendation: Synchronize all machines in a line to a common time source. For machines that do not have an absolute time source, the year should be set to "0000". At any rate, the timestamp should be monotonic.

3.3 CheckAlive

The CheckAlive message is used to detect connection losses. It therefore does not have to transport data and can be ignored by the receiver. Accordingly there is no response.

However, if a machine supports the FeatureCheckAliveResponse, it must answer CheckAlive messages with Type set to 1 with a CheckAlive message with Type set to 2 and the same Id as the received CheckAlive message.

Note: The function of CheckAliveResponse is optional. If FeatureCheckAliveResponse is specified in the ServiceDescription, it must be fully supported. Otherwise it can be ignored.

CheckAlive	Type	Range / Multiplicity	Optional	Description
◆ Type	int	1..2	yes	Ping / Pong message type.
◆ Id	string	any string (minimum supported length: 80 bytes)	yes	Identifier of the message.

Type may be one of the following values:

- 1 Ping: CheckAlive request.
- 2 Pong: CheckAlive response.

The machine sending CheckAlive message with Type set to 1 chooses a unique for Id (e.g., GUID or time stamp). The machine responding with CheckAlive message with Type set to 2 has to answer using the same Id.

3.4 ServiceDescription

The ServiceDescription message is sent by both machines after a connection is established. The downstream machine sends its ServiceDescription first whereupon the upstream machine answers by sending its own ServiceDescription.



ServiceDescription	Type	Range / Multiplicity	Optional	Description
◆ Machinelid	string	any string (minimum supported length: 80 bytes)	no	ID / name of the sending machine for identifying it in a Hermes enabled production line.
◆ Lanelid	int	1 .. n	no	The sending machine's lane to which this connection is relating to. Lanes are enumerated looking downstream from right to left beginning with 1.
◆ Interfacelid	string	any string (minimum supported length: 80 bytes)	yes	The ID of the sending machine's transportation interface to which this connection is relating to.
◆ Version	string	xxx.yyy (7 bytes)	no	The implemented interface version of the machine.
📁 SupportedFeatures	Feature []	0 .. n	no	List of supported features (empty for version 1.0).

Feature	Type	Range / Multiplicity	Optional	Description
📁 FeatureCheckAliveResponse	FeatureCheckAliveResponse	1	yes	Indication of CheckAliveResponse function implementation.
📁 FeatureBoardForecast	FeatureBoardForecast	1	yes	In the upstream role: Machine sends BoardForecast messages.
📁 FeatureQueryBoardInfo	FeatureQueryBoardInfo	1	yes	Indication of QueryBoardInfo function implementation.
📁 FeatureSendBoardInfo	FeatureSendBoardInfo	1	yes	Indication of SendBoardInfo function implementation.
📁 FeatureCommand	FeatureCommand	1	yes	Indication of Command function implementation.

xxx.yyy must match the regular expression
`[1-9][0-9]{0,2}\.[0-9]{1,3}`

The features specified in version 1.0 of this protocol have to be provided by any implementation and thus are not listed in the SupportedFeatures list of the ServiceDescription explicitly. The same applies for all mandatory



features of the version specified in the Version attribute. All optional features or additional features of an higher version supported by a machine need to be listed in the SupportedFeatures list to indicate there availability.

3.5 Notification

The Notification message is sent by both machines before a connection is terminated, e.g., after protocol errors or before shutdown. It could also be used for general notification purposes.

Notification	Type	Range / Multiplicity	Optional	Description
◆ NotificationCode	int	1 .. n	no	A notification code of the list below. Notification codes above 1000 are not defined by this protocol and may be used by the application.
◆ Severity	int	1 .. 4	no	A value of the list below.
◆ Description	string	any string (minimum supported length: 254 bytes)	no	An English textual description of the notification.

The following NotificationCodes are defined:

- 1 Protocol error (invalid transition in the corresponding state machine)
- 2 Connection refused because of an established connection
- 3 Connection reset because of changed configuration
- 4 Configuration error
- 5 Machine shutdown
- 6 BoardForecast error

Possible values for Severity:

- 1 Fatal error
- 2 Error
- 3 Warning
- 4 Info

3.6 BoardAvailable

The BoardAvailable message is sent to the downstream machine to indicate the readiness of the upstream machine to handover a PCB. When an optional attribute is received from an upstream machine, then it must be passed on (possibly altered) to the next downstream machine.



BoardAvailable	Type	Range / Multiplicity	Optional	Description
◆ BoardId	string	GUID (36 bytes)	no	Indicating the ID of the available board.
◆ BoardIdCreatedBy	string	non-empty string (minimum supported length: 80 bytes)	no	MachinelId of the machine which created the BoardId (the first machine in a consecutive row of machines implementing this protocol). The MachinelId is part of the Hermes configuration.
◆ FailedBoard	int	0 .. 2	no	A value of the list below.
◆ ProductTypeId	string	any string (minimum supported length: 254 bytes)	yes	Identifies a collection of PCBs sharing common properties.
◆ FlippedBoard	int	0 .. 2	no	A value of the list below.
◆ TopBarcode	string	any string (minimum supported length: 254 bytes)	yes	The barcode of the top side of the PCB.
◆ BottomBarcode	string	any string (minimum supported length: 254 bytes)	yes	The barcode of the bottom side of the PCB.
◆ Length	float	positive numbers	yes	The length of the PCB in millimeter.
◆ Width	float	positive numbers	yes	The width of the PCB in millimeter.
◆ Thickness	float	positive numbers	yes	The thickness of the PCB in millimeter.
◆ ConveyorSpeed	float	positive numbers	yes	The conveyor speed preferred by the upstream machine in millimeter per second.
◆ TopClearanceHeight	float	positive numbers	yes	The clearance height for the top side of the PCB in millimeter.
◆ BottomClearanceHeight	float	positive numbers	yes	The clearance height for the bottom side of the PCB in millimeter.
◆ Weight	float	positive numbers	yes	The weight of the PCB in grams.



◆ WorkOrderId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the work order for production of the PCB.
◆ BatchId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the Batch for production of the PCB within a splitted work order
◆ Route	int	0...65535	yes	A routing instruction to downstream machines. Use a value of the list below. See also 2.3.7 Handling of Attribute 'Route
◆ Action	int	0...65535	yes	An action instruction to downstream machine. Use a value of the list below. See also 2.3.8 Handling of Attribute 'Action
📁 SubBoards	SB []	0...n (minimum supported length: 494 bytes)	yes	A list of SubBoards Note: Due to limited retain memory in PLCs this attribute might only be supported for a limited number of subboards.

SB	Type	Range / Multiplicity	Optional	Description
◆ Pos	int	0...65535	no	Position number of subboard according to IPC-2591 CFX Unit numbering rule.
◆ Bc	string	any string	yes	The barcode of the subboard.
◆ St	int	0 .. 4	no	A value of the list below.

GUID must match the regular expression

`[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}`

FailedBoard may be one of the following values:

- 0 Board of unknown quality available
- 1 Good board available
- 2 Failed board available

FlippedBoard may be one of the following values:

- 0 Side up is unknown
- 1 Board top side is up
- 2 Board bottom side is up



If FlippedBoard is 2 (board bottom side is up) then TopBarcode is facing downwards and BottomBarcode is facing upwards. Same applies for TopClearanceHeight and BottomClearanceHeight.

The definition of board bottom and board top side is outside of the scope of The Hermes Standard and left to the customer.

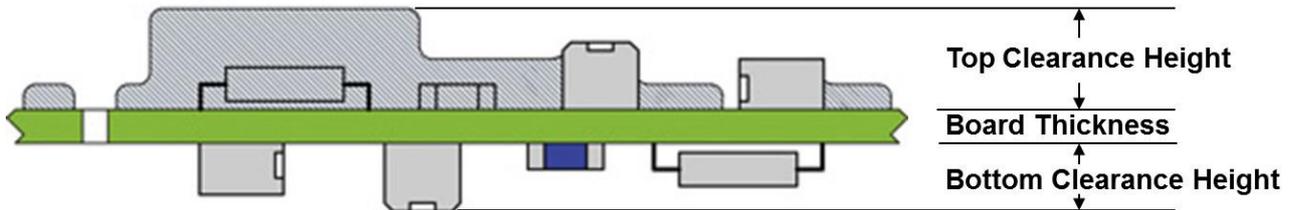


Figure 23 Explanation for Top and Bottom Clearance Height

Route may be one of the following values:

- 0 Route undefined
- 1..99 Transfer the board at route [no.] /defined by line configuration
- 900 Return the board
- (≤ 998 Reserved for future definition)
- 999 Manual removal of the board
- ≥1000 For individual definition within a production line

Action may be one of the following values:

- 0 Action undefined
- 1 Process the board (e.g., Flipping, Marking, ...)
- 2 Pass through the board without processing
- (≤ 999 Reserved for future definition)
- ≥1000 For individual definition within a production line

St (State) may be one of the following values:

- 0 Subboard of unknown quality
- 1 Good Subboard
- 2 Failed Subboard
- 3 Missing Subboard
- 4 Skip Subboard

Note: To keep memory consumption of message as low as possible the XML keywords in the list of Subboards are abbreviated to

- Pos Position
- Bc Barcode
- St State



3.7 RevokeBoardAvailable

With the RevokeBoardAvailable message, the upstream machine signals that it is not ready anymore to handover a PCB.

RevokeBoardAvailable	Type	Range / Multiplicity	Optional	Description
----------------------	------	----------------------	----------	-------------

3.8 MachineReady

The MachineReady message is sent to the upstream machine to indicate the readiness of the downstream machine to accept a PCB.

MachineReady	Type	Range / Multiplicity	Optional	Description
FailedBoard	int	0 .. 2	no	A value of the list below.
ForecastId	string	any string (minimum supported length: 80 bytes)	yes / no	If responding to a BoardForecast message mandatory. It indicates the ID of the original BoardForecast message.
BoardId	string	GUID (36 bytes)	yes	Indicates the ID of the board that will be handed over as next. In case of product change this attribute will not be sent.
ProductTypeId	string	any string (minimum supported length: 254 bytes)	yes	Identifies a collection of PCBs sharing common properties.
FlippedBoard	int	0 .. 2	yes	A value of the list below.
Length	float	positive numbers	yes	The length of the PCB in millimeter.
Width	float	positive numbers	yes	The width of the PCB in millimeter.
Thickness	float	positive numbers	yes	The thickness of the PCB in millimeter.
ConveyorSpeed	float	positive numbers	yes	The conveyor speed used by the upstream machine in millimeter per second.
TopClearanceHeight	float	positive numbers	yes	The clearance height for the top side of the PCB in millimeter.
BottomClearanceHeight	float	positive numbers	yes	The clearance height for the bottom side of the PCB in millimeter.
Weight	float	positive numbers	yes	The weight of the PCB in grams.



◆ WorkOrderId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the work order for production of the PCB.
◆ BatchId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the Batch for production of the PCB within a splitted work order

FlippedBoard may be one of the following values:

- 0 Side up is unknown
- 1 Board top side is up
- 2 Board bottom side is up

FailedBoard may be one of the following values:

- 0 Ready to accept any board
- 1 Ready to accept good boards.
- 2 Ready to accept failed boards

3.9 RevokeMachineReady

With the RevokeMachineReady message, the downstream machine signals that it is not ready anymore to accept a PCB.

RevokeMachineReady	Type	Range / Multiplicity	Optional	Description
--------------------	------	----------------------	----------	-------------

3.10 StartTransport

The StartTransport message is sent to the upstream machine to initiate the PCB handover process. There is no response to this message.

StartTransport	Type	Range / Multiplicity	Optional	Description
◆ BoardId	string	GUID (36 bytes)	no	The ID of the board for which the transport shall be started.
◆ ConveyorSpeed	float	positive numbers	yes	Optional parameter indicating the selected conveyor speed for the handover in millimeter per second.



The downstream machine is responsible for selecting the actual conveyor speed according to the preferred conveyor speed sent in the BoardAvailable message. In general the highest possible speed supported by both machines will be selected.

If a StartTransport message is received for a BoardId which is not the one received with the last BoardAvailable message, the transport **shall** be canceled. This case is not to be treated as a protocol error.

3.11 StopTransport

The StopTransport message is sent by the downstream machine after it has finished the transport.

StopTransport	Type	Range / Multiplicity	Optional	Description
◆ TransferState	int	1 .. 3	no	A value of the list below.
◆ BoardId	string	GUID (36 bytes)	no	The ID of the board to which the message relates to.

Transfer states:

- 1 NotStarted: The PCB never left and hence is fully inside the upstream machine.
- 2 Incomplete: The transfer was cancelled in progress.
- 3 Complete: The transfer ended successfully.

If the BoardId does not match the one from StartTransport, this **shall** be treated as a protocol error. Therefore, the connection would need to be re-established.

3.12 TransportFinished

The TransportFinished message is sent by the upstream machine after it finished the transport.

TransportFinished	Type	Range / Multiplicity	Optional	Description
◆ TransferState	int	1 .. 3	no	A value of the list below.
◆ BoardId	string	GUID (36 bytes)	no	The ID of the board to which the message relates to.

Transfer states:

- 1 NotStarted: The PCB never left and hence is fully inside the upstream machine.
- 2 Incomplete: The transfer was cancelled in progress.
- 3 Complete: The transfer ended successfully.

If the BoardId does not match the one from StartTransport, this **shall** be treated as a protocol error. Therefore, the connection would need to be re-established.

3.16 BoardForecast

The BoardForecast message is sent to the downstream machine to indicate some changes / command execution are needed or to give advanced information about the next board but a PCB is not yet available. If





the ForecastId attribute is set then the downstream machine must at some point respond with a MachineReady carrying the same ForecastId. If needed downstream machine must send a RevokeMachineReady message first. If the forecasted product is not accepted by the downstream machine, then it must respond with a Notification of type "BoardForecastError".

Note: The function of BoardForecast is optional. If FeatureBoardForecast is specified in the ServiceDescription, it must be fully supported. Otherwise it can be ignored.



BoardForecast	Type	Range / Multiplicity	Optional	Description
ForecastId	string	any string (minimum supported length: 80 bytes)	yes	Indicating the ID of forecast message. The ID must be unambiguous and e.g., can be a timestamp or a GUID.
TimeUntilAvailable	float	positive numbers	yes	Number of seconds until a board may be available at downstream machine.
BoardId	string	GUID (36 bytes)	yes	Indicating the ID of the board that will be handed over as next. e.g., in case of product change this attribute will not be sent.
BoardIdCreatedBy	string	any string (minimum supported length: 80 bytes)	yes	MachinedId of the machine which created the BoardId.
FailedBoard	int	0 .. 2	no	A value of the list below.
ProductTypeId	string	any string (minimum supported length: 254 bytes)	yes	Identifies a collection of PCBs sharing common properties.
FlippedBoard	int	0 .. 2	no	A value of the list below.
TopBarcode	string	any string (minimum supported length: 254 bytes)	yes	The barcode of the top side of the next PCB.
BottomBarcode	string	any string (minimum supported length: 254 bytes)	yes	The barcode of the bottom side of the next PCB.
Length	float	positive numbers	yes	The length of the PCB in millimeter.
Width	float	positive numbers	yes	The width of the PCB in millimeter.
Thickness	float	positive numbers	yes	The thickness of the PCB in millimeter.
ConveyorSpeed	float	positive numbers	yes	The conveyor speed preferred by the upstream machine in millimeter per second.



◆ TopClearanceHeight	float	positive numbers	yes	The clearance height for the top side of the PCB in millimeter.
◆ BottomClearanceHeight	float	positive numbers	yes	The clearance height for the bottom side of the PCB in millimeter.
◆ Weight	float	positive numbers	yes	The weight of the PCB in grams.
◆ WorkOrderId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the work order for production of the PCB.
◆ BatchId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the Batch for production of the PCB within a splitted work order

The attributes definition are identical to the BoardAvailable message.

FailedBoard may be one of the following values:

- 0 Ready to accept any board
- 1 Ready to accept good boards
- 2 Ready to accept failed boards

FlippedBoard may be one of the following values:

- 0 Side up is unknown
- 1 Board top side is up
- 2 Board bottom side is up

3.17 QueryBoardInfo

The QueryBoardInfo message is sent to the upstream machine to request information about one of the last boards (see 4.1.3).

Note: The function of QueryBoardInfo is optional. If FeatureQueryBoardInfo is specified in the ServiceDescription, it must be fully supported. Otherwise it can be ignored.

QueryBoardInfo	Type	Range / Multiplicity	Optional	Description
◆ TopBarcode	String	any string (minimum supported length: 254 bytes)	yes / no	The barcode of the top side of the PCB. Either top or bottom barcode must be specified.



◆ BottomBarcode	String	any string (minimum supported length: 254 bytes)	yes / no	The barcode of the bottom side of the PCB. Either top or bottom barcode must be specified.
-----------------	--------	--	----------	--

3.18 SendBoardInfo

The SendBoardInfo message is sent to the downstream machine as response of a received QueryBoardInfo message to transfer stored information about one of the last boards (see 4.1.3). If the upstream machine cannot find any board information it will nevertheless send the SendBoardInfo message without the BoardId and BoardCreatedBy attributes.

Machines supporting the feature FeatureSendBoardInfo **shall** be able to store and supply upon request the info of at least the last 50 handled boards.

Note:

- The function of SendBoardInfo is optional. If FeatureSendBoardInfo is specified in the ServiceDescription, it must be fully supported. Otherwise it can be ignored.
- With regards to limited memory space on PLCs, SendBoardInfo does not contain SubBoard info.

SendBoardInfo	Type	Range / Multiplicity	Optional	Description
◆ BoardId	string	GUID (36 bytes)	yes / no	The ID of the board which data has been requested. This attribute will not be sent if the board information has not been found.
◆ BoardIdCreatedBy	string	non-empty string (minimum supported length: 80 bytes)	yes / no	MachinedId of the machine which created the BoardId. This attribute will not be sent if the board information has not been found.
◆ FailedBoard	Int	0 .. 2	yes / no	A value of the list below. This attribute will not be sent if the board information has not been found.
◆ ProductTypeId	string	any string (minimum supported length: 254 bytes)	yes	Identifies a collection of PCBs sharing common properties.
◆ FlippedBoard	Int	0 .. 2	yes / no	A value of the list below. This attribute will not be sent if the board information has not been found.



◆ TopBarcode	string	any string (minimum supported length: 254 bytes)	yes / no	The barcode of the top side of the next PCB. This attribute is mandatory if it has been in the QueryBoardInfo message.
◆ BottomBarcode	string	any string (minimum supported length: 254 bytes)	yes / no	The barcode of the bottom side of the next PCB. This attribute is mandatory if it has been in the QueryBoardInfo message.
◆ Length	float	positive numbers	yes	The length of the PCB in millimeter.
◆ Width	float	positive numbers	yes	The width of the PCB in millimeter.
◆ Thickness	float	positive numbers	yes	The thickness of the PCB in millimeter.
◆ TopClearanceHeight	float	positive numbers	yes	The clearance height for the top side of the PCB in millimeter.
◆ BottomClearanceHeight	float	positive numbers	yes	The clearance height for the bottom side of the PCB in millimeter.
◆ WorkOrderId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the work order for production of the PCB.
◆ BatchId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the Batch for production of the PCB within a splitted work order

The attributes definition are identical to the BoardAvailable message.

FailedBoard may be one of the following values:

- 0 Board of unknown quality available
- 1 Good board available
- 2 Failed board available

FlippedBoard may be one of the following values:

- 0 Side up is unknown
- 1 Board top side is up
- 2 Board bottom side is up



3.19 SetConfiguration

The SetConfiguration message is sent by an engineering station to configure the Hermes interfaces of a machine. If the sent configuration is not accepted, the machine is expected to send a Notification message (see 3.5).

Note: The function of SetConfiguration is optional on the vertical channel. If FeatureConfiguration is specified in the SupervisoryServiceDescription, it must be fully supported. Otherwise it can be ignored.

SetConfiguration	Type	Range / Multiplicity	Optional	Description
◆ Machined	string	any string (minimum supported length: 80 bytes)	no	ID / name of this machine for identifying it in a Hermes enabled production line.
◆ SupervisorySystemPort	int	0 .. 65535	yes	Port number on which connections from supervisory systems shall be established.
📁 UpstreamConfigurations	Upstream Configuration []	0 .. n	no	Configuration for upstream lanes.
📁 DownstreamConfigurations	Downstream Configuration []	0 .. n	no	Configuration for downstream lanes.



UpstreamConfiguration	Type	Range / Multiplicity	Optional	Description
◆ UpstreamLaneId	int	1 .. n	no	The lane on the upstream side. Lanes are enumerated looking downstream from right to left beginning with 1.
◆ UpstreamInterfaceId	string	any string (minimum supported length: 80 bytes)	yes	The ID of the transportation interface on the upstream side.
◆ HostAddress	string	valid IP address or hostname (minimum supported length: 254 bytes)	no	The IP address or hostname of the upstream machine for this lane and transportation interface.
◆ Port	int	0 .. 65535	no	Port number on which connections shall be established.

DownstreamConfiguration	Type	Range / Multiplicity	Optional	Description
◆ DownstreamLaneId	int	1 .. n	no	The lane on the downstream side. Lanes are enumerated looking downstream from right to left beginning with 1.
◆ DownstreamInterfaceId	string	any string (minimum supported length: 80 bytes)	yes	The ID of the transportation interface on the downstream side.
◆ ClientAddress	string	valid IP address or hostname (minimum supported length: 254 bytes)	yes	The IP address or hostname of the downstream machine for this lane and transportation interface. If not specified, then connections from any IP address are accepted.
◆ Port	int	0 .. 65535	no	Port number on which the server shall accept connections for this lane.

All connections where the machine is acting as board provider are stored in DownstreamConfigurations. All connections where the machine is acting as board receiver are stored in UpstreamConfigurations. These are independent of the board transport direction of the SMT line.



It is up to the user to keep Machinelds unique.

3.20 GetConfiguration

The GetConfiguration message is sent by an engineering station to read out the current configuration of the Hermes interfaces of a machine. The machine is expected to answer with a CurrentConfiguration message.

Note: The function of GetConfiguration is optional on the vertical channel. If FeatureConfiguration is specified in the SupervisoryServiceDescription, it must be fully supported. Otherwise it can be ignored.

GetConfiguration	Type	Range / Multiplicity	Optional	Description
------------------	------	----------------------	----------	-------------

3.21 CurrentConfiguration

The CurrentConfiguration message is sent by a machine in response to the GetConfiguration message.

CurrentConfiguration	Type	Range / Multiplicity	Optional	Description
 Machineld	string	any string (minimum supported length: 80 bytes)	yes	ID / name of this machine for identifying it in a Hermes enabled production line.
 SupervisorySystemPort	int	0 .. 65535	yes	Port number on which connections from supervisory systems shall be established.
 UpstreamConfigurations	Upstream Configuration []	0 .. n	no	Configuration of upstream lanes.
 DownstreamConfigurations	Downstream Configuration []	0 .. n	no	Configuration of downstream lanes.

For the definition of UpstreamConfiguration and DownstreamConfiguration see section 3.19.

If no Machineld has been configured yet, the CurrentConfiguration message does not contain the attribute Machineld.



3.22 SupervisoryServiceDescription

The SupervisoryServiceDescription message is sent by both machine and supervisory system after a connection is established. The supervisory system sends its SupervisoryServiceDescription first whereupon the machine answers by sending its own SupervisoryServiceDescription.

SupervisoryServiceDescription	Type	Range / Multiplicity	Optional	Description
SystemId	String	any string (minimum supported length: 80 bytes)	no	ID / name of the sending machine or supervisory system for identifying it in a Hermes enabled production line.
Version	String	xxx.yyy (7 bytes)	no	The implemented interface version of the machine or supervisory system.
SupportedFeatures	SupervisoryFeature []	0 .. n	no	List of supported supervisory features (empty for version 1.0).

SupervisoryFeature	Type	Range / Multiplicity	Optional	Description
FeatureConfiguration	FeatureConfiguration	1	yes	Indication of configuration functions implementation.
FeatureCheckAliveResponse	FeatureCheckAliveResponse	1	yes	Indication of CheckAliveResponse function implementation.
FeatureBoardTracking	FeatureBoardTracking	1	yes	Indication of board tracking functions implementation.
FeatureQueryWorkOrderInfo	FeatureQueryWorkOrderInfo	1	yes	Indication of QueryWorkOrderInfo function implementation.
FeatureSendWorkOrderInfo	FeatureSendWorkOrderInfo	1	yes	Indication of SendWorkOrderInfo function implementation.
FeatureReplyWorkOrderInfo	FeatureReplyWorkOrderInfo	1	yes	Indication of ReplyWorkOrderInfo function implementation.

xxx.yyy must match the regular expression
`[1-9][0-9]{0,2}\.[0-9]{1,3}`

3.23 BoardArrived

The BoardArrived message is sent via Hermes vertical channel to a supervisory system to indicate that a PCB has arrived at this machine. The BoardArrived message **shall** be sent immediately after sending the corresponding StopTransport message.



Note: The function of BoardArrived is optional. If FeatureBoardTracking is specified in the SupervisoryServiceDescription, it must be fully supported. Otherwise it can be ignored.

BoardArrived	Type	Range / Multiplicity	Optional	Description
◆ Machineld	string	any string (minimum supported length: 80 bytes)	no	ID / name of this machine for identifying it in a Hermes enabled production line.
◆ UpstreamLaneld	int	1 .. n	no	The lane on the upstream side. Lanes are enumerated looking downstream from right to left beginning with 1.
◆ UpstreamInterfaceld	string	any string (minimum supported length: 80 bytes)	yes	The ID of the transportation interface on the upstream side.
◆ Magazineld	string	any string (minimum supported length: 80 bytes)	yes	Barcode of a magazine, required to identify the magazine from which the Board was transferred.
◆ SlotId	int	1 .. n	yes	Indicates the slot in the magazine, enumerated from bottom to top, beginning with 1.
◆ BoardTransfer	int	1 .. 3	no	A value of the list below
◆ BoardId	string	GUID (36 bytes)	no	Indicating the ID of the available board
◆ BoardIdCreatedBy	string	non-empty string (minimum supported length: 80 bytes)	no	Machineld of the machine which created the BoardId (the first machine in a consecutive row of machines implementing this protocol). The Machineld is part of the Hermes configuration.
◆ FailedBoard	int	0 .. 2	no	A value of the list below.
◆ ProductTypeld	string	any string (minimum supported length: 254 bytes)	yes	Identifies a collection of PCBs sharing common properties.
◆ FlippedBoard	int	0 .. 2	no	A value of the list below.



◆ TopBarcode	string	any string (minimum supported length: 254 bytes)	yes	The barcode of the top side of the PCB.
◆ BottomBarcode	string	any string (minimum supported length: 254 bytes)	yes	The barcode of the bottom side of the PCB.
◆ Length	float	positive numbers	yes	The length of the PCB in millimeter.
◆ Width	float	positive numbers	yes	The width of the PCB in millimeter.
◆ Thickness	float	positive numbers	yes	The thickness of the PCB in millimeter.
◆ ConveyorSpeed	float	positive numbers	yes	The conveyor speed used for the PCB transfer in millimeter per second.
◆ TopClearanceHeight	float	positive numbers	yes	The clearance height for the top side of the PCB in millimeter.
◆ BottomClearanceHeight	float	positive numbers	yes	The clearance height for the bottom side of the PCB in millimeter.
◆ Weight	float	positive numbers	yes	The weight of the PCB in grams.
◆ WorkOrderId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the work order for production of the PCB.
◆ BatchId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the Batch for production of the PCB within a splitted work order
◆ Route	int	0...65535	yes	A routing instruction to downstream machines. Use a value of the list below. See also 2.3.7 Handling of Attribute 'Route'
◆ Action	int	0...65535	yes	An action instruction to downstream machine. Use a value of the list below. See also 2.3.8 Handling of Attribute 'Action'
📁 SubBoards	SB []	0...n (minimum supported	yes	A list of SubBoards Note: Due to limited retain memory in PLCs this attribute might only be



		length: 494 bytes)		supported for a limited number of subboards.
--	--	--------------------	--	--

SB	Type	Range / Multiplicity	Optional	Description
◆ Pos	int	0...65535	no	Position number of subboard according to IPC-2591 CFX Unit numbering rule.
◆ Bc	string	any string	yes	The barcode of the subboard.
◆ St	int	0 .. 4	no	A value of the list below.

GUID must match the regular expression

`[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}`

FailedBoard may be one of the following values:

- 0 Board of unknown quality available
- 1 Good board available
- 2 Failed board available

FlippedBoard may be one of the following values:

- 0 Side up is unknown
- 1 Board top side is up
- 2 Board bottom side is up

BoardTransfer may be one of the following values:

- 1 Transferred: Board arrived from upstream machine via Hermes or SMEMA.
- 2 Loaded: Board was loaded from a magazine or a stack of Boards.
- 3 Inserted: Board was manually inserted into the machine.

Route may be one of the following values:

- 0 Route undefined
- 1..99 Transfer the board at route [no.] /defined by line configuration
- 900 Return the board
- (≤ 998 Reserved for future definition)
- 999 Manual removal of the board
- ≥1000 For individual definition within a production line

Action may be one of the following values:

- 0 Action undefined
- 1 Process the board (e.g., Flipping, Marking, ...)
- 2 Pass through the board without processing
- (≤ 999 Reserved for future definition)
- ≥1000 For individual definition within a production line

St (State) may be one of the following values:

- 0 Subboard of unknown quality
- 1 Good Subboard



- 2 Failed Subboard
- 3 Missing Subboard
- 4 Skip Subboard

Note: To keep memory consumption of message as low as possible the XML keywords in the list of Subboards are abbreviated to

- Pos Position
- Bc Barcode
- St State

3.24 BoardDeparted

The BoardDeparted message is sent via Hermes vertical channel to a supervisory system to indicate that a PCB has left this machine. The BoardDeparted message **shall** be sent immediately after sending the corresponding TransportFinished message.

Note: The function of BoardDeparted is optional. If FeatureBoardTracking is specified in the SupervisoryServiceDescription, it must be fully supported. Otherwise it can be ignored.

BoardDeparted	Type	Range / Multiplicity	Optional	Description
◆ Machinelid	string	any string (minimum supported length: 80 bytes)	no	ID / name of this machine for identifying it in a Hermes enabled production line.
◆ DownstreamLanelid	int	1 .. n	no	The lane on the downstream side. Lanes are enumerated looking downstream from right to left beginning with 1.
◆ DownstreamInterfacelid	string	any string (minimum supported length: 80 bytes)	yes	The ID of the transportation interface on the downstream side.
◆ Magazinelid	string	any string (minimum supported length: 80 bytes)	yes	Barcode of a magazine, required to identify the magazine to which the Board was transferred.
◆ Slotid	int	1 .. n	yes	Indicates the slot in the magazine, enumerated from bottom to top, beginning with 1.
◆ BoardTransfer	int	1 .. 3	no	A value of the list below.
◆ BoardId	string	GUID (36 bytes)	no	Indicating the ID of the available board.



◆ BoardIdCreatedBy	string	non-empty string (minimum supported length: 80 bytes)	no	MachinelId of the machine which created the BoardId (the first machine in a consecutive row of machines implementing this protocol). The MachinelId is part of the Hermes configuration.
◆ FailedBoard	int	0 .. 2	no	A value of the list below.
◆ ProductTypeId	string	any string (minimum supported length: 254 bytes)	yes	Identifies a collection of PCBs sharing common properties.
◆ FlippedBoard	int	0 .. 2	no	A value of the list below.
◆ TopBarcode	string	any string (minimum supported length: 254 bytes)	yes	The barcode of the top side of the PCB.
◆ BottomBarcode	string	any string (minimum supported length: 254 bytes)	yes	The barcode of the bottom side of the PCB.
◆ Length	float	positive numbers	yes	The length of the PCB in millimeter.
◆ Width	float	positive numbers	yes	The width of the PCB in millimeter.
◆ Thickness	float	positive numbers	yes	The thickness of the PCB in millimeter.
◆ ConveyorSpeed	float	positive numbers	yes	The conveyor speed used for the PCB transfer in millimeter per second.
◆ TopClearanceHeight	float	positive numbers	yes	The clearance height for the top side of the PCB in millimeter.
◆ BottomClearanceHeight	float	positive numbers	yes	The clearance height for the bottom side of the PCB in millimeter.
◆ Weight	float	positive numbers	yes	The weight of the PCB in grams.
◆ WorkOrderId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the work order for production of the PCB.
◆ BatchId	string	any string	yes	Identifies the Batch for production of the PCB within a splitted work order



		(minimum supported length: 80 bytes)		
◆ Route	int	0...65535	yes	A routing instruction to downstream machines. Use a value of the list below. See also 2.3.7 Handling of Attribute 'Route
◆ Action	int	0...65535	yes	An action instruction to downstream machine. Use a value of the list below. See also 2.3.8 Handling of Attribute 'Action
📁 SubBoards	SB []	0...n (minimum supported length: 494 bytes)	yes	A list of SubBoards Note: Due to limited retain memory in PLCs this attribute might only be supported for a limited number of subboards.

SB	Type	Range / Multiplicity	Optional	Description
◆ Pos	int	0...65535	no	Position number of subboard according to IPC-2591 CFX Unit numbering rule.
◆ Bc	string	any string	yes	The barcode of the subboard.
◆ St	int	0 .. 4	no	A value of the list below.

GUID must match the regular expression

```
[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}
```

FailedBoard may be one of the following values:

- 0 Board of unknown quality available
- 1 Good board available
- 2 Failed board available

FlippedBoard may be one of the following values:

- 0 Side up is unknown
- 1 Board top side is up
- 2 Board bottom side is up

BoardTransfer may be one of the following values:

- 1 Transferred: Board moved to downstream machine via Hermes or SMEMA.
- 2 Unloaded: Board was unloaded into a magazine.
- 3 Removed: Board was manually taken out of the machine.

Route may be one of the following values:

- 0 Route undefined
- 1..99 Transfer the board at route [no.] /defined by line configuration



- 900 Return the board
- (≤ 998 Reserved for future definition)
- 999 Manual removal of the board
- ≥1000 For individual definition within a production line

Action may be one of the following values:

- 0 Action undefined
- 1 Process the board (e.g., Flipping, Marking, ...)
- 2 Pass through the board without processing
- (≤ 999 Reserved for future definition)
- ≥1000 For individual definition within a production line

St (State) may be one of the following values:

- 0 Subboard of unknown quality
- 1 Good Subboard
- 2 Failed Subboard
- 3 Missing Subboard
- 4 Skip Subboard

Note: To keep memory consumption of message as low as possible the XML keywords in the list of Subboards are abbreviated to

- Pos Position
- Bc Barcode
- St State

3.25 QueryWorkOrderInfo

The QueryWorkOrderInfo message is sent via Hermes vertical channel from a machine to a supervisory system to query the work order and initial board data for a PCB or a set of PCBs. Four scenarios are covered:

- a) PCBs arrive within a magazine
- b) A stack of PCBs arrives
- c) A PCB is inserted and its barcode is known
- d) Workorder id is available

Note: The function of QueryWorkOrderInfo is optional. If FeatureQueryWorkOrderInfo is specified in the SupervisoryServiceDescription, it must be fully supported. Otherwise it can be ignored.

QueryWorkOrderInfo	Type	Range / Multiplicity	Optional	Description
◆ QueryId	string	any string (minimum supported length: 80 bytes)	yes	Indicates the ID of QueryWorkOrder message. The ID must be unambiguous and e.g., can be a timestamp or a GUID.
◆ MachineId	string	any string	no	ID / name of this machine for identifying it in a Hermes enabled production line.



		(minimum supported length: 80 bytes)		
◆ Magazineld	string	any string (minimum supported length: 80 bytes)	yes	Barcode of a magazine, required to identify the magazine.
◆ SlotId	int	1 .. n	yes	Indicates the slot in the magazine, enumerated from bottom to top, beginning with 1.
◆ Barcode	string	any string (minimum supported length: 254 bytes)	yes	The barcode of the PCB.
◆ WorkOrderId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the work order for production of the PCB.
◆ BatchId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the Batch for production of the PCB within a splitted work order.

GUID must match the regular expression

```
[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}
```

3.26 SendWorkOrderInfo

The SendWorkOrderInfo message is sent via Hermes vertical channel from a supervisory system to a machine to provide the work order and the initial board data for a PCB or a set of PCBs. If the supervisory system cannot find any work order information it will nevertheless send the SendWorkOrderInfo message without any attributes except QueryId, if provided upon request.

Note: The function of SendWorkOrderInfo is optional. If FeatureSendWorkOrderInfo is specified in the SupervisoryServiceDescription, it must be fully supported. Otherwise it can be ignored.

SendWorkOrderInfo	Type	Range / Multiplicity	Optional	Description
◆ QueryId	string	any string	yes / no	ID of QueryWorkOrderInfo this message refers to. This attribute is mandatory if it



		(minimum supported length: 80 bytes)		has been in the QueryWorkOrderInfo message.
◆ WorkOrderId	string	non-empty string (minimum supported length: 80 bytes)	yes	Identifies the work order for production of the PCB.
◆ BatchId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the Batch for production of the PCB within a splitted work order
◆ BoardId	string	GUID (36 bytes)	yes	Indicating the ID of the available board.
◆ BoardIdCreatedBy	string	non-empty string (minimum supported length: 80 bytes)	yes	Machineld of the machine which created the BoardId (the first machine in a consecutive row of machines implementing this protocol). The Machineld is part of the Hermes configuration.
◆ FailedBoard	int	0 .. 2	yes / no	A value of the list below. This attribute will not be sent if the board information has not been found.
◆ ProductTypeId	string	any string (minimum supported length: 254 bytes)	yes	Identifies a collection of PCBs sharing common properties.
◆ FlippedBoard	int	0 .. 2	yes / no	A value of the list below. This attribute will not be sent if the board information has not been found.
◆ TopBarcode	string	any string (minimum supported length: 254 bytes)	yes / no	The barcode of the top side of the PCB. This attribute is mandatory if it has been the barcode in the QueryWorkOrderInfo message.
◆ BottomBarcode	string	any string (minimum supported length: 254 bytes)	yes / no	The barcode of the bottom side of the PCB. This attribute is mandatory if it has been the barcode in the QueryWorkOrderInfo message.



◆ Length	float	positive numbers	yes	The length of the PCB in millimeter.
◆ Width	float	positive numbers	yes	The width of the PCB in millimeter.
◆ Thickness	float	positive numbers	yes	The thickness of the PCB in millimeter.
◆ ConveyorSpeed	float	positive numbers	yes	The conveyor speed used for the PCB transfer in millimeter per second.
◆ TopClearanceHeight	float	positive numbers	yes	The clearance height for the top side of the PCB in millimeter.
◆ BottomClearanceHeight	float	positive numbers	yes	The clearance height for the bottom side of the PCB in millimeter.
◆ Weight	float	positive numbers	yes	The weight of the PCB in grams.
◆ Route	int	0...65535	yes	A routing instruction to downstream machines. Use a value of the list below. See also 2.3.7 Handling of Attribute 'Route'
📁 SubBoards	SB []	0...n (minimum supported length: 494 bytes)	yes	A list of SubBoards Note: Due to limited retain memory in PLCs this attribute might only be supported for a limited number of subboards.

SB	Type	Range / Multiplicity	Optional	Description
◆ Pos	int	0...65535	no	Position number of subboard according to IPC-2591 CFX Unit numbering rule.
◆ Bc	string	any string	yes	The barcode of the subboard.
◆ St	int	0 .. 4	no	A value of the list below.

GUID must match the regular expression

`[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}`

FailedBoard may be one of the following values:

- 0 Board of unknown quality available
- 1 Good board available
- 2 Failed board available

FlippedBoard may be one of the following values:

- 0 Side up is unknown
- 1 Board top side is up
- 2 Board bottom side is up

Route may be one of the following values:



- 0 Route undefined
- 1..99 Transfer the board at route [no.] /defined by line configuration
- 900 Return the board
- (≤ 998 Reserved for future definition)
- 999 Manual removal of the board
- ≥1000 For individual definition within a production line

St (State) may be one of the following values:

- 0 Subboard of unknown quality
- 1 Good Subboard
- 2 Failed Subboard
- 3 Missing Subboard
- 4 Skip Subboard

Note: To keep memory consumption of message as low as possible the XML keywords in the list of Subboards are abbreviated to

- Pos Position
- Bc Barcode
- St State

3.27 ReplyWorkOrderInfo

Note: The function of ReplyWorkOrderInfo is optional. If FeatureReplyWorkOrderInfo is specified in the SupervisoryServiceDescription, it must be fully supported. Otherwise it can be ignored.

The ReplyWorkOrderInfo message may be optionally sent via Hermes vertical channel from a Hermes machine to a supervisory system to reply to a previous SendWorkOrderInfo message. Several consecutive ReplyWorkOrderInfo messages may be sent to inform about status changes over time.

ReplyWorkOrderInfo	Type	Range / Multiplicity	Optional	Description
◆ WorkOrderId	string	non-empty string (minimum supported length: 80 bytes)	no	Identifies the work order for production of the PCB.
◆ BatchId	string	any string (minimum supported length: 80 bytes)	yes	Identifies the Batch for production of the PCB within a splitted work order
◆ Status	int	0 .. 2	no	Result of Work Order execution, a value of the list below

Status may be one of the following values:



- 0 Work Order rejected
- 1 Work Order accepted and ready for execution
- 2 Work Order accepted and queued for execution

3.28 Command

Note: The function of Command optional. If FeatureCommand is specified in the ServiceDescription, it must be handled like described below. Otherwise it can be ignored.

In certain use cases it is necessary that a machine can request its upstream or downstream connected neighbour to perform a certain activity. Examples are

- the oven error loop to prevent boards from moving into the oven when the buffer after the oven is almost full,
- the request of a shuttle unit to its neighbours to pause / resume operation so that the door of this shuttle unit can be opened safely,
- etc.

Command will be passed on to further upstream / downstream machines until it reaches the first machine that is able to execute it.

If a machine receives a **Command** from downstream and is the first one in line or if its upstream neighbour does not support **Command**, and if this machine cannot execute the Command, then this Command will be discarded.

If a machine receives a **Command** from upstream and is the last one in line or if its downstream neighbour does not support **Command**, and if this machine cannot execute the Command, then this Command will be discarded.

The **Command** message is sent to the upstream / downstream machine to indicate to this machine to perform one of the following activities:

- No command
- Lock input conveyor
- Unlock input conveyor
- Request pause
- Confirm pause
- Resume operation

The **Command** message

- can be sent at any time, i.e. it is not related to any of the states in the Hermes state diagram,
- will be passed on upstream / downstream by machine that does not handle such a command,
- will be processed in the same sequence as they are received,
- **shall** never be sent on the same interface from where it was received.

When receiving a Command a machine **shall** handle it as soon as possible, independent of the state(s) of its Hermes interface(s). In case of a Command Lock input conveyor an ongoing board transfer **shall** be completed before locking the input conveyor. A machine may execute Command or, if it cannot execute it,



- send the **Command** to the upstream neighbour if it was received from downstream, or
- send the **Command** to the downstream neighbour if it was received from upstream.

If a machine has more than one upstream / downstream port then this machine needs to be configured how to handle this **Command** in one of the following ways:

- Discard it,
- Execute it,
- Pass it on to the configured upstream / downstream connections or, if nothing is configured (by default), to all upstream / downstream connections.

Note: In case of a misconfiguration it is possible that a **Command** might end up in an endless loop and will never be executed, but puts load on the Hermes communication.

Command	Type	Range	Optional	Description
◆ Command	int	0 .. 65535	no	A value of the list below

Command may be one of the following values:

- 0 Command undefined → discard
- 1 Lock input conveyor
- 2 Unlock input conveyor
- 3 Request pause
- 4 Confirm pause
- 5 Resume operation

6...999 reserved for future use

1000... customer defined commands



4 Appendix

4.1 Special Scenarios

The following sections are not part of the Hermes protocol specification. In fact they **shall** show the application of this protocol in some special scenarios.

4.1.1 Board Tracking When Board Is Torn Out From the Line

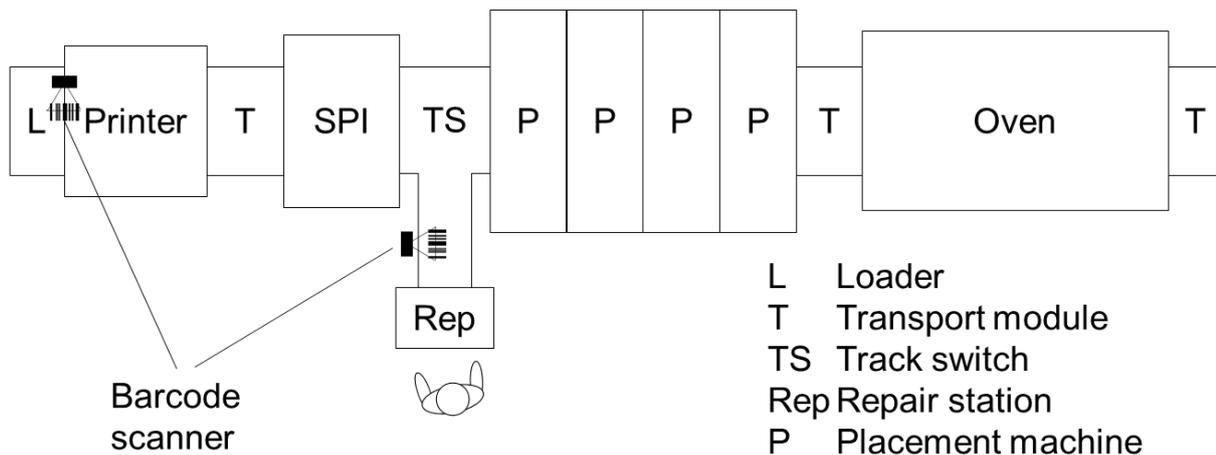


Figure 24 Line Setup With Barcode Readers and Repair Station

In this scenario, a repair station is placed behind the SPI. PCBs failing the solder paste inspection are torn out by the track switch and are presented to an operator at the repair station. The operator may take out the PCB for rework and re-insert it later independent of the PCB sequence.

By removing the PCB from the line, the link between the PCB and the barcode respectively the BoardId is lost. So when the PCB is re-inserted, different approaches are possible to re-establish the tracking of the PCB:

- Create a new Hermes BoardId, read the barcode and report the from now on used tracking information. The tracking information can be merged later by an external system (e.g., MES) using the barcodes.
- Read the barcode first and request the corresponding Hermes BoardId from the external system (e.g., MES). The tracking can be continued using the primarily assigned Hermes BoardId.
- The machine blocks the production of the re-inserted PCB until the operator scans the barcode using a mobile barcode scanner or enters it manually and specifies which board side is currently up. Then the original Hermes BoardId and all the needed information is requested from the upstream machine via the QueryBoardInfo message. The downstream machine sends the QueryBoardInfo with the top or bottom barcode and gets back a SendBoardInfo message from the upstream machine including BoardId. If information for that barcode was not available then the attribute BoardId will be omitted.
- Simplest but most unsecure approach: The repair station prompts the operator to confirm that the inserted PCB is the same which was last removed from the station.



Option a and b are realized with an MES system. Option c and d enables the re-insertion of boards directly at the machine without having an MES system for that line (relying only on functions of The Hermes Standard).

4.1.2 Board Tracking When Board Is Temporarily Removed From the Line

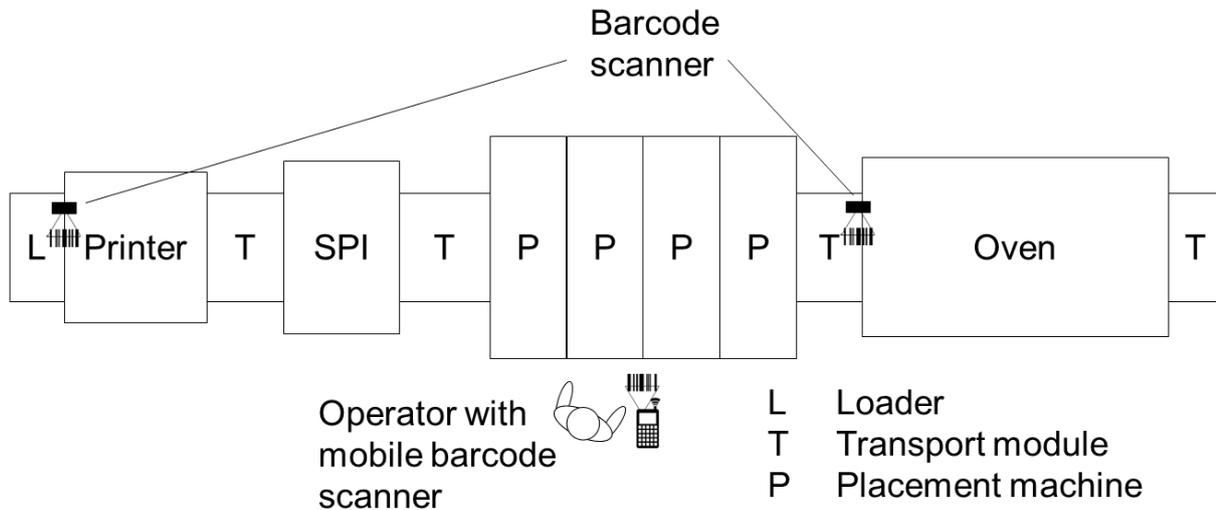


Figure 25 Line Setup With Fixed and Mobile Barcode Readers

In this scenario, the operator removes a PCB for inspection from one of the placement machines. The line continues producing PCBs. At some later point in time, the PCB is re-inserted to complete its production.

By removing the PCB from the line, the link between the PCB and the barcode respectively the BoardId is lost. As in the scenario above, different approaches are possible to re-establish the tracking of the PCB:

- a) The machine blocks the production of the re-inserted PCB until the operator scans the barcode using a mobile barcode scanner or enters it manually. Then either the original Hermes BoardId is requested from an external system (e.g., MES) using the barcode or a new Hermes BoardId is created and the tracking information is merged by the external system.
- b) The machine blocks the production of the re-inserted PCB until the operator scans the barcode using a mobile barcode scanner or enters it manually and specifies which board side is currently up. Then the original Hermes BoardId and all the needed information is requested from the upstream machine via the QueryBoardInfo message. The downstream machine sends the QueryBoardInfo with the top or bottom barcode and gets back a SendBoardInfo message from the upstream machine including BoardId. If information for that barcode was not available then the attribute BoardId will be omitted.
- c) A new Hermes BoardId is created and production is continued without barcode. At the next barcode reader in the line, the barcode information is complemented to the Hermes BoardId. An external system can later merge all the collected tracking information.



Option a and c are realized with an MES system. Option b enables the re-insertion of boards directly at the machine without having an MES system for that line (relying only on functions of The Hermes Standard).

4.1.3 Board Tracking When Board Was Transferred Without Data

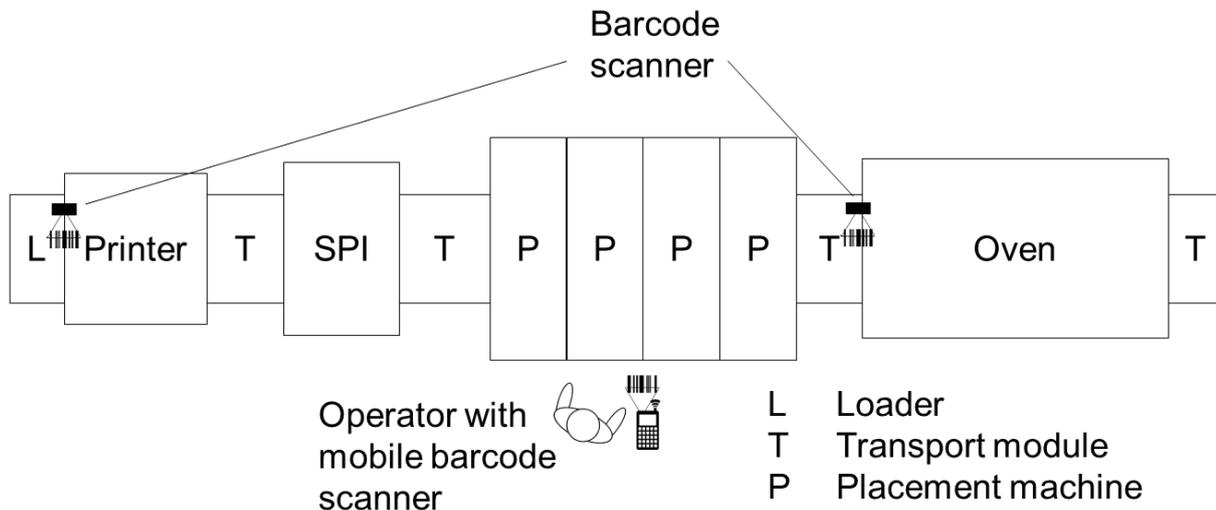


Figure 26 Line Setup With Fixed and Mobile Barcode Readers

In this scenario, one of the machines (e.g., a soldering reflow machine) cannot physically stop the transport of the PCB at the end of the machine. So boards may pile up if the next machine is not able to process the boards.

In that case the operator will temporarily remove the boards from the line and try to reinsert those at the same place a bit later on.

By removing a PCB from the line, the link between the PCB, the BoardId and other information (e.g., width, length) is lost. As in the scenario above, different approaches are possible to re-establish the tracking of the PCB:

- a) The machine blocks the production of the re-inserted PCB until the operator scans the barcode using a mobile barcode scanner or enters it manually. Then either the original Hermes BoardId is requested from an external system (e.g., MES) using the barcode or a new Hermes BoardId is created and the tracking information is merged by the external system.
- b) The machine blocks the production of the re-inserted PCB until the operator scans the barcode using a mobile barcode scanner or enters it manually and specifies which board side is currently up. Then the original Hermes BoardId and all the needed information is requested from the upstream machine, that could not stop the PCB, via the QueryBoardInfo message. The downstream machine sends the QueryBoardInfo with the top or bottom barcode and gets back a SendBoardInfo message from the



upstream machine including BoardId. If information for that barcode was not available then the attribute BoardId will be omitted.

- c) A new Hermes BoardId is created and production is continued without barcode. Information will not be available for the next machine. At the next barcode reader in the line, the barcode information is added to the Hermes data. An external system can later merge all the collected tracking information (if needed).

Option a and c are realized with an MES system. Option b enables the re-insertion of boards directly at the next machine without having an MES system for that line (relying only on functions of The Hermes Standard).



4.1.4 Oven Error Loop

An oven can take a certain number of boards and heats them up with a defined temperature profile. When the soldering process is completed, boards must leave the oven to prevent them from being burned. Usually, there is a conveyor after the oven to cool down the boards. After this conveyor the boards are gathered in a buffer. If the buffer is filling up and cannot take anymore boards, then boards in the oven are blocked from leaving the oven—they will be burned.

The so called “oven error loop” prevents this situation: When the buffer’s fill level exceeds a certain limit, the buffer notifies the oven to lock its input conveyor, so that boards can no longer enter the oven and, later on, get stuck in the oven due to a full buffer.

The **Command** message allows a buffer to instruct the oven

- to lock its input conveyor when a certain fill level at the buffer is exceeded, and
- to unlock its input conveyor when the buffer has sufficient space available.

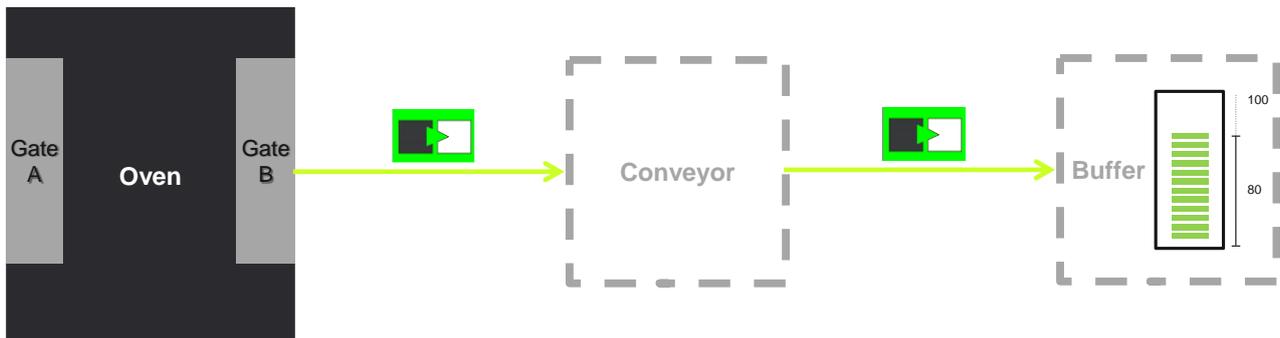


Figure 27 SMT Subline That Is Involved in Oven Error Loop

Assumption: Buffer can take 100 boards, oven can take 20 boards. Hence, if the buffer’s fill level exceeds 80 boards, then the buffer can no longer take all the boards that are inside the oven. **This situation must be avoided.**

Example sequence of events:

- Buffer’s fill level reaches 80 boards: Buffer sends Command LockInput to conveyor
- Conveyor is configured not to execute any Command; thus, conveyor passes Command on to oven
- Oven receives Command LockInput and locks its input conveyor
- Oven continues normal operation, remaining 20 boards in oven are soldered and travel to buffer
- Buffer takes the remaining 20 boards from oven and will be filled completely
- Operator comes, picks up filled magazine from buffer and puts an empty magazine into buffer
- Buffer’s fill level is now 0, and buffer can take boards again: buffer sends Command UnlockInput to conveyor
- Conveyor is configured not to execute any Command; thus, conveyor passes Command on to Oven
- Oven receives Command UnlockInput and unlocks its input conveyor

4.1.5 Request Pause / Confirm Pause and Resume Operation

This section describes a use case for pausing upstream and/or downstream machines. The principle can be used either only upstream, only downstream or for both directions.

In this use case there are 2 neighbour machines (Test handler, magazine handler) that have dangerous movements inside of the machine that needs to be protected during operation. The inner area of those neighbor machines are reachable via the covered belt conveyor when its door is opened.

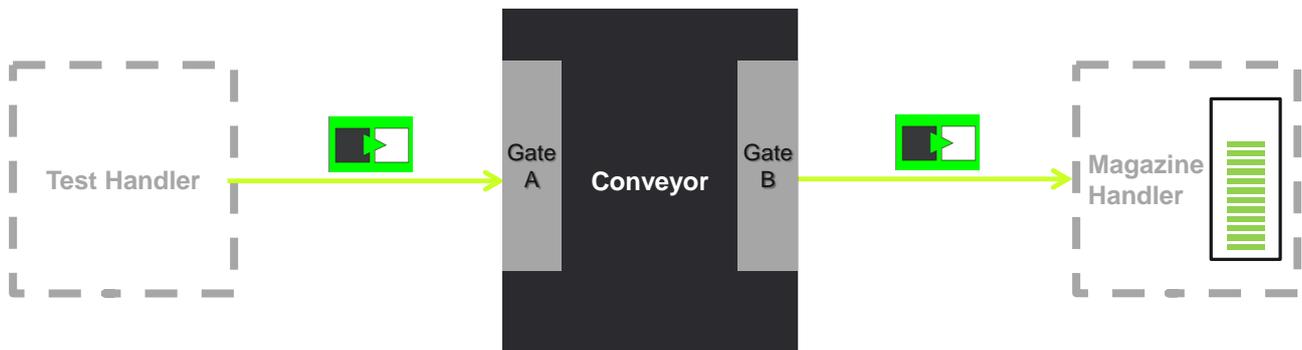


Figure 28 Example Subline Showing Use Case Request Pause / Confirm Pause and Resume Operation

An operator can require several possible actions to do on a machine :

- Request for inserting a board
- Request for removing the next board
- Request to open a door, to be able to check the system inside and do actions (e.g., solving a stucked board, rescan a board with the barcode reader)

To allow an operator to do this it is required to have the possibility to open the door in a controlled and safe way. By default the door of the conveyor in this case is locked.

Note: The Hermes interfacing is not acting as a safety link of machines, it is just synchronizing machines in this case so that the safety loop between machines are not uncontrolled opened, causing machines to be abruptly stopped in an uncontrolled way. So, Hermes does do the synchronization in this case between the machines, but a separate additional safety connection is still required to fulfil the safety level requirements.

On request of the operator the door can be requested to be opened. On the event of the open door request, the conveyor needs to command **“Request pause”** to the upstream and/or downstream machines to go into a safe mode. The safe mode can be either closing a shutter door, or a stand still mode by not doing any dangerous movements or else. So it could still continue none dangerous operations (ex. Testing the board on the board handler).

On receiving this command **“Request pause”** on the neighbor machines, they can take action to go into standstill mode. From the moment the machine is safe, it needs to report back to the conveyor that it is paused, and that it is safe to open the door on his behalf. This can be done by sending back the command **“Confirm pause”** to the conveyor.

From the moment the conveyor has received all **“Confirm pause”** confirmations from each connection it requested the pause, it can unlock the door and indicate it to the operator to open the door, and let him do the operation he requires to do. After the action the operator has to close the door to continue the operation of the line





From the moment the door is closed, the conveyor **shall** send the command "**Resume operation**" to its upstream and/or downstream connection, to let the neighbor machines continue its operation.

From now on the line can operate further in a normal way.

In case of a very small neighbor machine, it could be required for getting the correct safety distances to send the same request further onwards in the line to the next neighbor machine, and should be handled properly to indicate a safe state to the requestor. This needs to be configured in each of the machines when this would be required, and can be handled with the same commands.



4.1.6 Board Removal at Downstream Conveyor

A process machine instructs the next possible downstream conveyor to stop the board for its manual removal. In this example only conveyor B provides the possibility that the board can be manually removed.

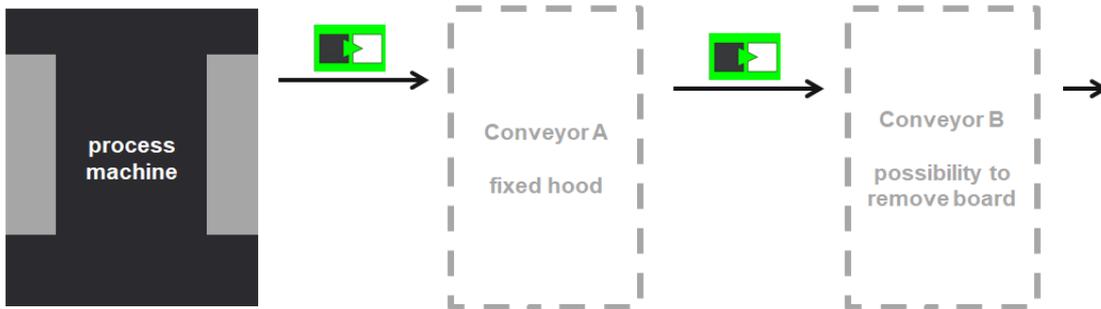


Figure 29 Board Removal at Downstream Conveyor

Example sequence of events

- Conveyor A and B empty: MachineReady
- Process machine: BoardAvailable, Route=999 (Removal)
- Protocol and board handling to conveyor A
- Conveyor A does not offer the option of removing the board manually
- Board passes through to conveyor B: BoardAvailable, Route=999 (Removal)
- Protocol and board handling to conveyor B
- Conveyor B requests the user to remove the board
- (Hermes data is discarded when board is removed)

4.1.7 Reversal Transportation to a Flipping Unit Located Downstream a Process Machine

A board shall be processed both sides and therefore turned over between process steps. The flipping unit is located downstream to the process machine regarding to the direction of the production line.

Precondition: For reversal transportation the machines need to setup one upstream and one downstream connection per gate with reversal transportation (this example: Gate B). The flipping unit processes only when instructed by the process machine.

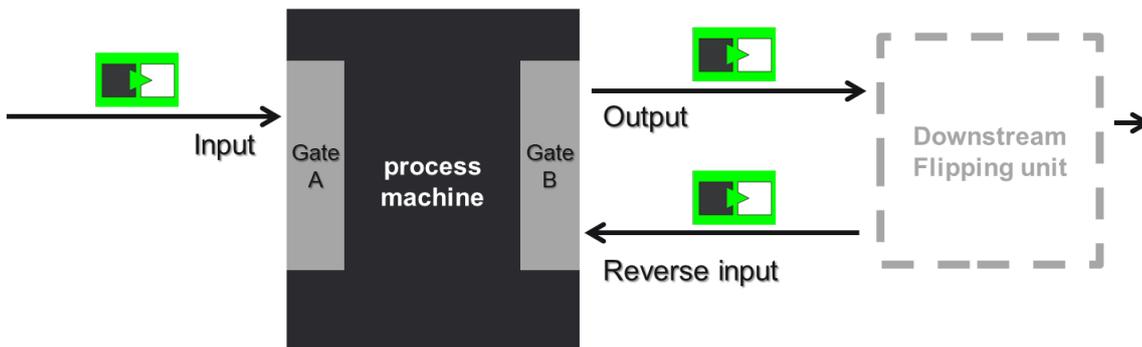


Figure 30 Reversal Transportation, Downstream Flipping Unit

Example sequence of events

- Process machine processes top side of a board
- Output to the flipping unit: **BoardAvailable**, Route=900 (Return the board), Action=1 (Process the board), FlippedBoard=1 (Top side is up)
- Flipping unit sends **BoardForecast** message with *BoardId* to the process machine (at Reverse input channel) to indicate that a board will be transferred back soon
- Protocol and board handling to flipping unit
- Process machine, Reverse input: **MachineReady**
- Process machine must ensure, that board can be returned

Normal case:

- Flipping unit turns over the board as requested
- Flipping unit to Reverse input: **BoardAvailable**, FlippedBoard=2 (Bottom side is up)
- Protocol and board handling to process machine
- Process machine processes bottom side of the board
- Output to flipping unit: **BoardAvailable** Action=2 (Pass through the board)

Note: If the board shall be turned over a second time, then Action=1 (Process the board) can here also be sent

- Protocol and board handling to flipping unit
- Board passes through the flipping unit unturned

Failure case:

- Flipping failed; board was manually removed from the flipping unit
- Flipping unit sends **BoardForecast** message without *BoardId* to the upstream process machine (at Reverse input channel) to indicate that the expected board would not come back.



4.1.8 Reversal Transportation to a Flipping Unit Located Upstream a Process Machine

A board shall be processed both sides and therefore turned over between process steps. The flipping unit is located upstream to the process machine regarding to the direction of the production line.

Precondition: For reversal transportation the machines need to setup one upstream and one downstream connection per gate with reversal transportation (this example: Gate A). The flipping unit processes only when instructed by the process machine.

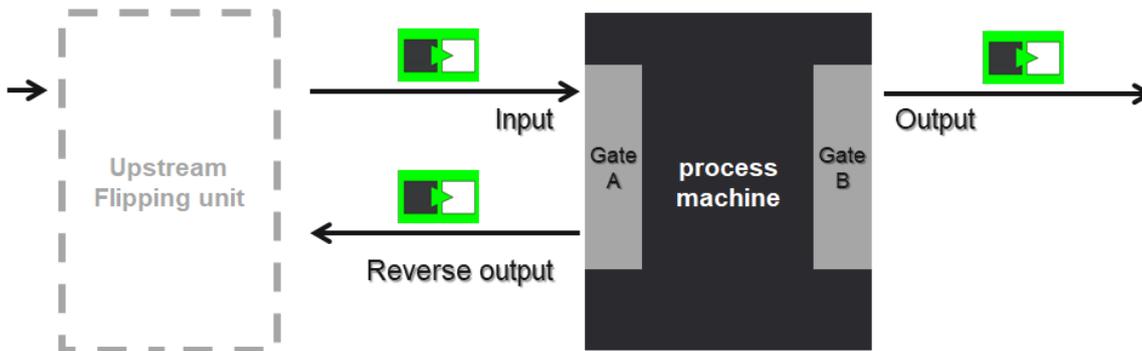


Figure 31 Reversal Transportation, Upstream Flipping Unit

Example sequence of events

- Flipping unit loads a board from upstream
- It passes the board through to the process machine unturned: **BoardAvailable**, FlippedBoard=1 (Top side is up)
- Process machine knows that board has to be processed both sides (e.g., by analyzing board parameters)
- Process machine sends **BoardForecast** message with *BoardId* to the flipping unit (at Reverse output channel) to indicate that a board will be transferred soon and space for the board has to be reserved; no new board loading from upstream.

Note: BoardForecast message has to be sent before StartTransport message

- Protocol and board handling to process machine
- Flipping unit now doesn't await a board from upstream but from the reverse output of the process machine: **MachineReady**
- Process machine processes top side of the board
- Reverse output to flipping unit: **BoardAvailable**, Route=900 (Return the board), Action=1 (Process the board), FlippedBoard=1 (Top side is up)
- Protocol and board handling to flipping unit
- Process machine, Input: **MachineReady**

Normal case:

- Flipping unit turns over the board as requested
- To the process machine, Input: **BoardAvailable**, FlippedBoard=2 (Bottom side is up)

Note: If the board shall be turned over a second time then **BoardForecast** message with *BoardId* can here be sent again by the process machine. Such BoardForecast message has to be sent before StartTransport message

- Protocol and board handling to process machine
- Process machine processes bottom side of the board; flipping unit now can load a new board from upstream if available



- Process machine unloads the board to the downstream Output

Failure case "Flipping failed":

- Flipping failed; board was removed from the flipping unit
- The flipping unit loads a new board from upstream
- To the process machine: **BoardAvailable**
- Process machine recognizes that the board is not the one as expected. It can load the new board and process the top side or handle the situation as an error.

Failure case "Process machine failed and will not return board":

- Process machine failed and will not return board
- Process machine sends **BoardForecast** message without *BoardId* to the flipping unit (at Reverse output channel) to indicate that the board will not be returned
- Flipping unit discards reservation of space for board and is ready to receive a board at its upstream port

4.1.9 Board Routing Within a Production Line by Predefined Routes

This concept is recommended if the production line provides just a small number of routes. It can also be used for a section of a line.

In the production line planning every meaningful route gets defined by a route number. A board can now follow one of these routes from the beginning to the end.

Furthermore it is possible for machines to be optional decision taker and to switch the route depending on some event (e.g., AOI test result). In the example below the decision taker can be either the AOI when unloading the board or the shuttle when loading the board.

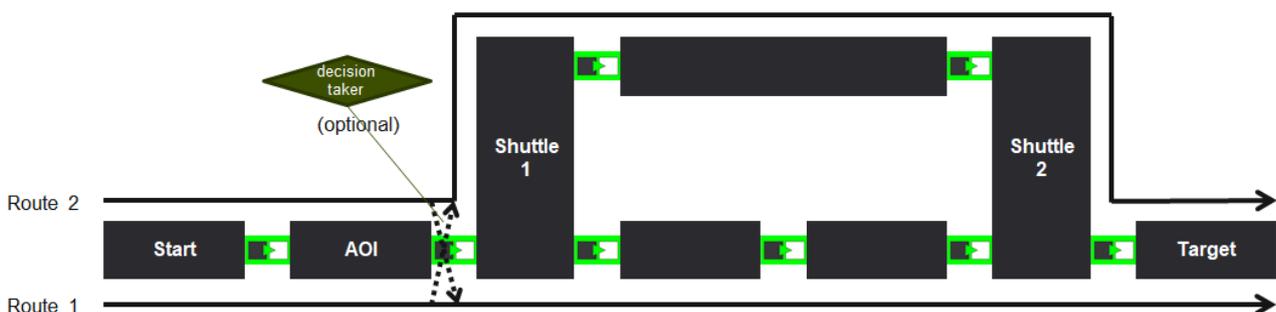


Figure 32 Board Routing, Predefined Routes

Concept features:

- Route attribute has to be processed at machines with multiple outlets (e.g., shuttles) to decide where to unload the board.
- No machine has to change the route attribute (except optional process-related decision makers).
- A more complex line structure can greatly increase the number of routes and also the configuration effort at many machines. → recommendation to use example 4.1.10

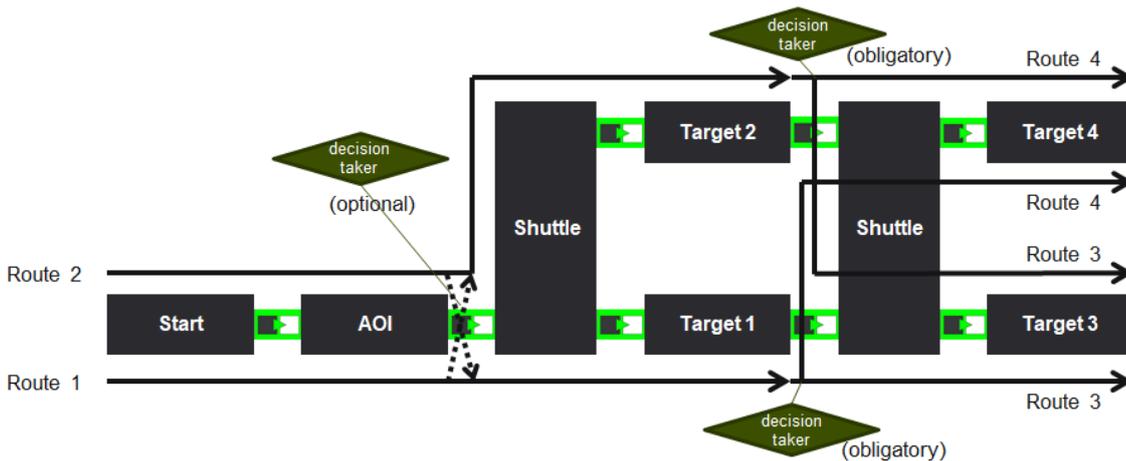
4.1.10 Board Routing Within a Production Line Towards Target Locations

This concept is recommended if the production line provides several target locations or a more complex structure.



In the production line planning every target location gets defined by a route number. Each machine having multiple outlets must be configured which outlet leads to which route. After the board reaches the target the machine has to set the next route number (next target). Therefore, the production line concept must include from where the decision taker gets this information (e.g., board parameters, vertical communication). It should be noted that points in the line has to be defined as additional target locations before line parts are joined together. This can be machines before the shuttle or also the inlet point of the shuttle itself. (example below: machines "Target 1" and "Target 2"). A board with its route attribute will now be guided towards the target location. Is this location not at the end of the line the machine has to be a decision taker and to set the next route number.

Furthermore it is also possible for machines to be optional decision taker and to switch the route depending on some event (e.g., AOI test result).



Concept features:

- Route attribute has to be processed at machines with multiple outputs (e.g., shuttle) to decide where to unload the board.
- Route attribute has to be set before line parts join together. Therefore, additional decision taker at the source of the decision must be defined (e.g., board parameter, vertical communication).

4.2 Glossary / Abbreviations

GUID	Globally Unique Identifier
ID	Identifier
IP	Internet Protocol
ISO / OSI	International Organization for Standardization / Open System Interconnection
M2M	Machine-to-Machine
MES	Manufacturing Execution System
PCB	Printed Circuit Board
SMEMA	Surface Mount Equipment Manufacturers Association



SMT	Surface-Mount Technology
SPI	Solder Paste Inspection
TCP	Transmission Control Protocol
XML	Extensible Markup Language

4.3 References

[IPC_SMEMA_9851] IPC-SMEMA-9851 Mechanical Equipment Interface Standard

[ISO_7498-1] ISO/IEC IS 7498-1: Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model. 1996

[IETF_RFC_791] Internet Engineering Task Force: RFC791: Internet Protocol. September 1981

[IETF_RFC_2460] Internet Engineering Task Force: RFC791: Internet Protocol, Version 6 (IPv6). September 1998

[IETF_RFC_793] Internet Engineering Task Force: RFC793: Transmission Control Protocol. September 1981

[ITU-T_REC_X.667] International Standard "Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 Object Identifier components

[SemVer_2.0.0] Tom Preston-Werner: Semantic Versioning 2.0.0. (Internet: <https://semver.org/spec/v2.0.0.html>, last access: 23. April 2018)

[W3C_XML_1.1] Extensible Markup Language (XML) 1.1 (Second Edition) - W3C Recommendation 16. August 2006, edited in place 29. September 2006

[W3C_DATE_TIME] Date and Time Formats - W3C Recommendation 15. September 1997

[W3C_XML_Schema] XML Schema Part 2: Datatypes Second Edition - W3C Recommendation 28. October 2004

4.4 History

Version	Date	Author	Change
1.0	03/23/17	The Hermes Standard Initiative	Initial Version



1.0, Rev 1	11/13/17	The Hermes Standard Initiative	Incorporation of changes agreed in initiative meeting <ul style="list-style-type: none"> • Add Top and Bottom clearance height attribute to Board Available message • When already connected to a downstream machine, reject new connection attempts • Specify the BoardId to be a true globally unique identifier (GUID / UUID) • Remove BoardIdCreatedBy from Start-Transport, StopTransport, TransportFinished
1.0.2	04/23/18	The Hermes Standard Initiative	Incorporation of changes agreed in initiative meeting <ul style="list-style-type: none"> • Application of Semantic Versioning • Define minimum requirements for strings
1.1	04/23/18	The Hermes Standard Initiative	Incorporation of changes agreed in initiative meeting <ul style="list-style-type: none"> • Adding Interfaceld to the configuration • Add weight attribute to BoardAvailable message • CheckAlive Response • BoardForecast • Reinsert Board
1.2	01/28/19	The Hermes Standard Initiative	Incorporation of changes agreed in initiative meeting <ul style="list-style-type: none"> • Clarification of version number and supported features • Add WorkOrderID to related messages • Foundation of vertical channel • Board tracking to supervisory system • Work order handling
1.3	03/01/21	The Hermes Standard Initiative	Incorporation of changes agreed in initiative meeting <ul style="list-style-type: none"> • Add Batch ID to related Hermes Messages
1.4	09/20/21	The Hermes Standard Initiative	Incorporation of changes agreed in initiative meeting <ul style="list-style-type: none"> • Add new ReplyWorkOrderInfo message • Add new Command message • Add list of SubBoards with Position, Barcode and State to Hermes BoardAvailable, BoardArrived, BoardDeparted and SendWorkOrderInfo messages • Add Action and Route to related Hermes Messages





1.5	06/01/22	The Hermes Standard Initiative	Incorporation of changes agreed in initiative meeting <ul style="list-style-type: none">• Clarification of scope• Add WorkOrderID and BatchID to QueryWorkOrderInfo message
-----	----------	--------------------------------	--

