3.25 Command

Note: The function of **Command** is 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 oven intelligent counter is to control the quantity of board flow between the oven inlet conveyor and the buffer after the oven, ensuring enough empty slots are available in the buffer catered for the boards present in the oven and other machines between oven and the buffer. This prevents the boards from being damaged when the buffer is full and the boards are held inside the oven.
- 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
- Transfer out board count
- 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 Transfer out board count

7…999 reserved for future use

1000… customer defined commands

Recommendation to improve Oven Info Error

- Referring to the Hermes document section 4.1.4 below, the commands including Lock Input Conveyor (command = 1) and Unlock Input Conveyor (command = 2) are used to optimize the oven error loop.
- As useful as a (Un)lock input conveyor feature in command, it still remains uncertain if the board in Oven has already reached the maximum board processing limit (unless capped by setting) above the unoccupied slot of a buffer.
- By the time the buffer reaches its cut-off limit set in the machine setting, there could be more board inside the oven than the unoccupied buffer slot.
- A suggested way to improve the oven error loop is to include a new command, named **Transfer out board count <command command="6" \>** representing board transfer out pulse signal sent from the input conveyor of the oven to the buffer.

Attachment from Section 4.1.4 from Hermes 1.6

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.*

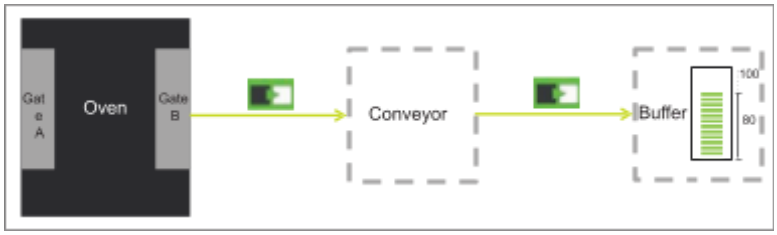


Figure 27 SMT Subline That Is Involved in Oven Error Loop

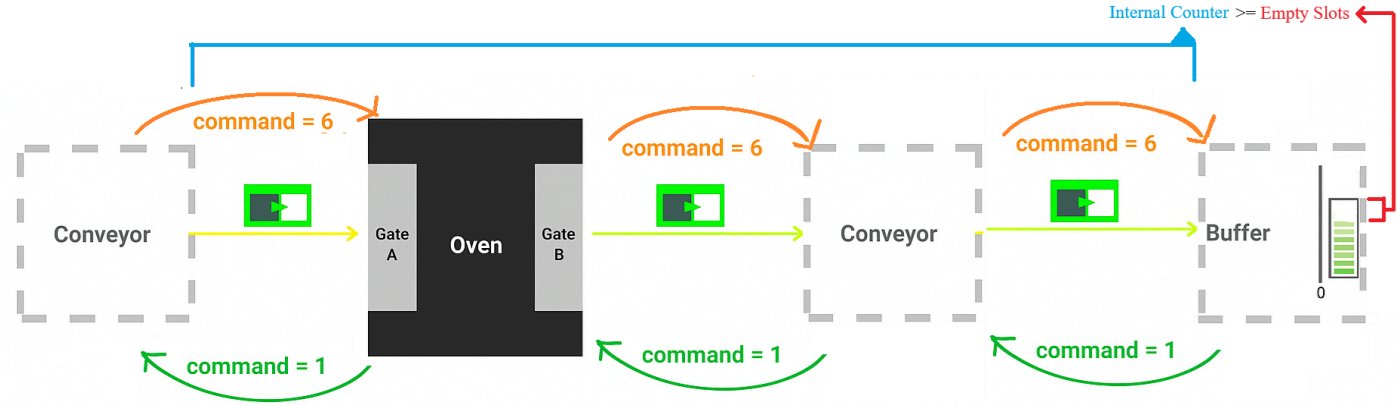
Assumption: Buffer can take 100 boards, oven can take 20 boards. Hence, if the buffer's fill level exceeds 80 boards, then the buffer can no longer take all the boards that are inside the oven. ***This situation must be avoided.***

Example sequence of events:

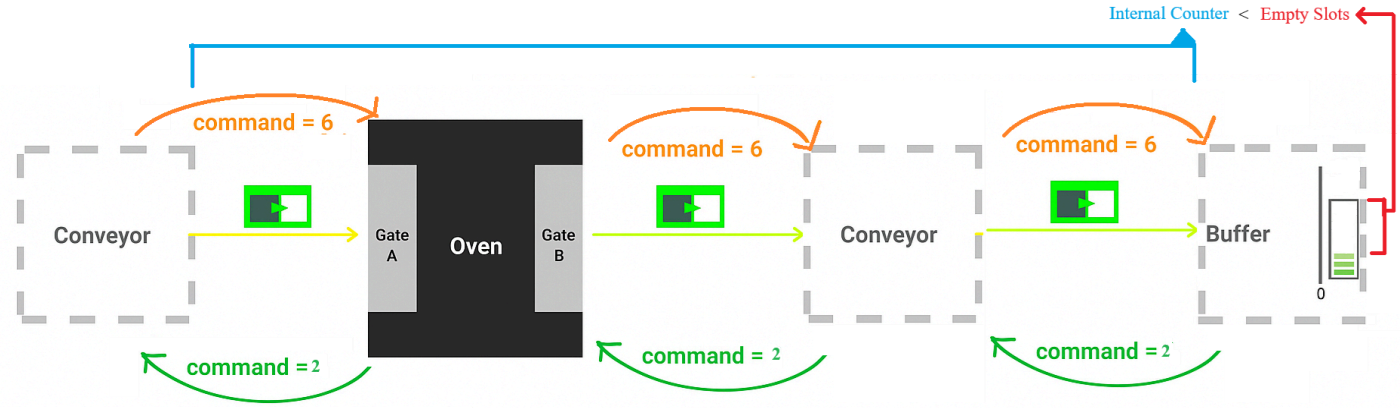
- *Buffer's fill level reaches 80 boards: Buffer sends Command LockInput to conveyor*
- *Conveyor is configured not to execute any Command; thus, conveyor passes Command on to oven*
- *Oven receives Command LockInput and locks its input conveyor*
- *Oven continues normal operation, remaining 20 boards 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*

Improved version of **Intelligent Oven Error Loop**

The commands including Lock Input Conveyor (command = 1) and Unlock Input Conveyor (command = 2) and Transfer Out Board Count (command=6) are used to optimize the oven error loop.



- When a board is inserted into the oven, a command "6" is sent to the buffer, the buffer will process the command and add one unit of board to its existing internal counter, if the counter exceeds (>=) the threshold of the buffer current empty slot. The buffer then sends command "1" to the oven input conveyor to pause the transfer of the incoming boards.
- When a board is transferred into the buffer, the counter will deduct one unit of board to its existing counter. Hence, the internal counter representing the board quantity spanning from the input conveyor till the buffer



- When the buffer slot is recovered to unoccupied state or slot increased due to newly inserted magazine, and the unoccupied slot is more than the internal counter, then command "2" will be sent to the inlet conveyor to resume board transferring
- This guarantees there is always an empty slot reserved for every incoming board from the input conveyor. This allows board counting and dynamically keeps the board quantity in the oven under a buffer accepting limit.
- The internal counter is adjustable interactively in case the board is manually removed in the middle conveyor.