TM

Version 1.20

November 2004

This specification is owned by UP³I Limited. Find more information at http://www.up3i.org

Members of the UP³I Core Group are:

Duplo Corporation
Hunkeler AG
IBM Corporation
Strålfors AB
Océ Printing Systems GmbH
Xerox Corporation

Please send any comments or requests on this specification to one of the following addresses:

Contact@up3i.org

Peter Dyson	Duplo Corporation	peter@duplointernational.com
Toni Egli	Hunkeler AG	t.egli@hunkeler.ch
Ron Parrish	IBM Corporation	ronp@us.ibm.com
Jan Persson	Strålfors AB	jan.persson@stralfors.se
Christian Sack	Océ Printing Systems GmbH	christian.sack@ops.de
Jerry Sturnick	Xerox Corporation	jerry.sturnick@usa.xerox.com

Trademarks

UP^{3}I and the UP^{3}I-logo are trademarks of UP³I Limited.

Any other trademarks mentioned in the UP³I Specification are the property of their respective owners.

History / Changes

Revision 1.20

- The PAC method was established in UP³I. See chapter 2.2.9, PAC Protocol Concept and chapter 2.4.2, PAC Frames.

Table of Contents

Table of Figures

1 Introduction

1.1 Purpose

This document describes the intelligent printer pre- and post-processing interface „UP^{3}I“.

The purpose of UP^{3}I is the connection of

- production printers (continuous form and cut sheet),
- in-line paper pre- and post-processing devices,
- workflow manager and remote operating stations (via a UP^{3}I Manager).

This document is a proposal for an open standard. The intended audience is printer, pre-processing and post-processing designers, system integrators and related application programmers.

1.2 Objective / Goal

The starting point for the development of the 𝕌niversal ℙrinter-, ℙre- and ℙost-processing 𝕀nterface (UP^{3}I) was the desire to have an “Intelligent Interface.“ The following topics describe the goal of such an intelligent interface:

- Shortening of Set-up / Preparation time
- Standard “Look and feel“ for User / Customer -> Operating Panels with common GUI elements
- Standard for future devices (Pre- and Post-processing and Printer) – NOT vendor specific
- Operating from one place / remote operation and monitoring
- Possibility for (easy) expansion of new functionalities and devices
- Possibility for implementation with job tickets
- Backward Compatibility to Type I, Type II, DFA,... Interfaces
- Support of continuous-form and cut sheet printing systems
- No need for a defined “Power On Sequence“ of the UP^{3}I devices
- Enhanced error recovery in case of paper jams
- Less Paper Waste – optimized paper synchronizing procedure
- NOT part of the UP³I proposal are: integrity control, production management, auditing, job ticket definition (CIP4, JDF, ...), information security, print data verifications, front and back correlation, etc.

UP³I is a compatible and complementary means to an end which bonds existing facilities (like JDF, CIP4...) together. ⇨ 'The intelligent link'.

1.3 UP³I Topology

The UP^{3}I topology allows a choice of one of four major levels of functionality to be implemented.

1. **Mixing "old" (Type I, II) and new (UP^{3}I) devices**	No additional functionality compared to Type I, II... interfaces.
2. **Pure UP3I production line**	Broadcasting of changed set-up values, like paper width (the operator needs to change the value only once).
3. **Adding a simple UP3I Manager**	Additional functionality like Remote Operating, Remote Set-up, ... are available.
4. **UP3I Manager**	Full UP3I functionality.

1.3.1 Compatibility to Type I, Type II, DFA, ... Interfaces

- A UP^{3}I converter converts „Type x" (Type I, Type II, ...) Interfaces to UP^{3}I and vice versa UP^{3}I to „Type x" Interfaces.
- A UP^{3}I Converter can be built e.g. with a PC including IEEE1394 and Type I or Type II, DFA, ... Cards or special PCB's or even with emulation software running on any UP3I device (⇨see "2.2.8 UP3I Emulation device").
- The main problem for converters may by the real time topic (e.g. 1/6 inch pulse...) – This problem can be solved e.g. with specialized PCB's or high end µProcessors...

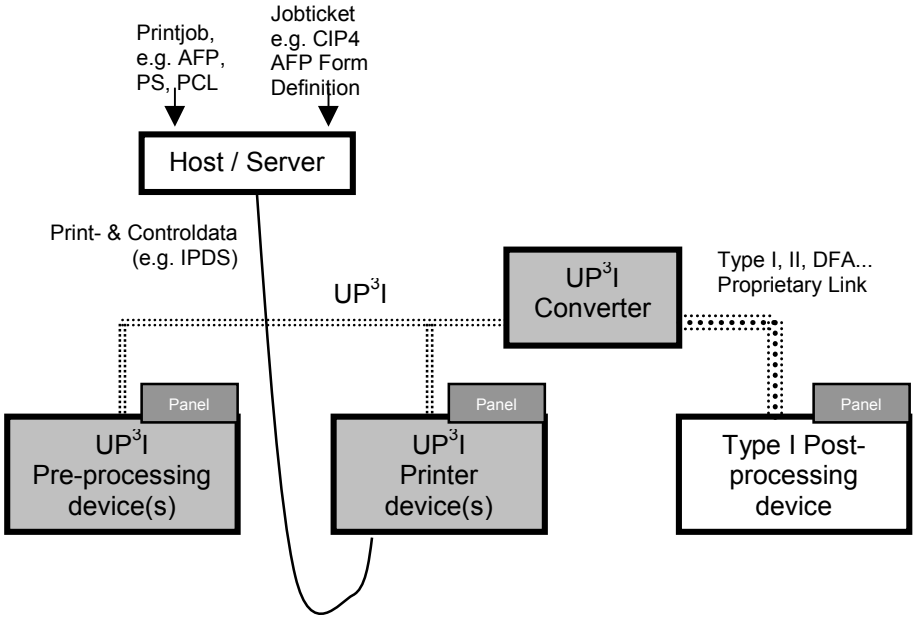


Figure 1-1: UP^{3}I Compatibility to Type I, Type II, ... devices

Compared to Type I, II interfaces, there may (depending upon the "intelligence" of the converter and the proprietary link) an enhanced error recovery and eventually an automatic setup be possible. Find more information at Chapter 2.2.8 "UP3I Emulation device".

1.3.2 Substitution of Type I, Type II, DFA, ... Interfaces

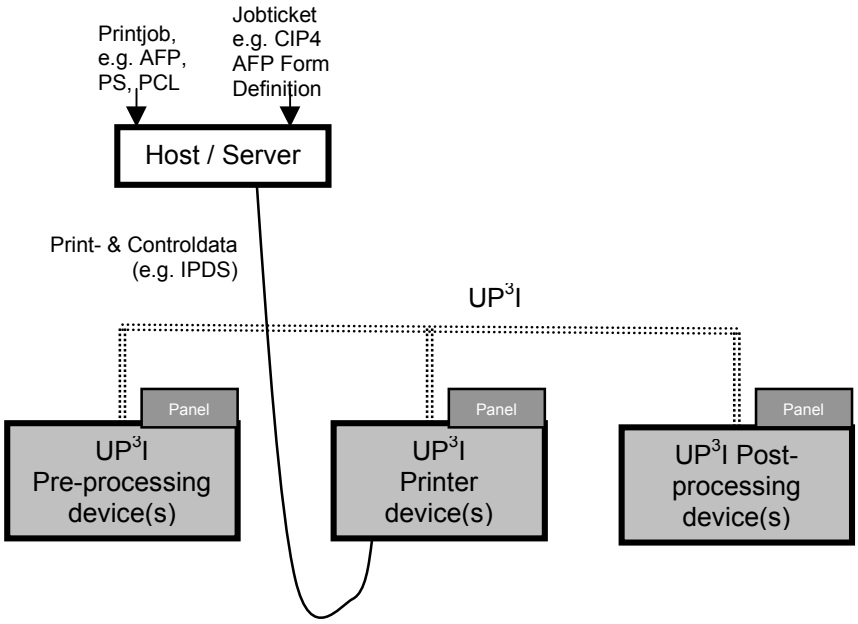


Figure 1-2: Substitution of Type I, Type II, ... devices

Up to this step no UP³I Manager is available. So UP³I devices still offer only a reduced functionality enhancement compared to Type I, Type II, DFA... interfaces. Find some examples listed below:

- All post-processing information that is included (by the server/host) in the print data is forwarded by the UP³I printer. With IPDS this is e.g.: DGB (Define Group Boundary), SGO (Specify Group Operation), AOS (Alternate Offset Stacker)...
- Changed values on the operating panel are broadcast to the other UP³I devices. The operator needs to change the value only once.
- With an “Extended Jam Recovery Point“ the printer also requests pages, which have been jammed at the post-processing device.

1.3.3 UP³I environment with Remote Operating capability

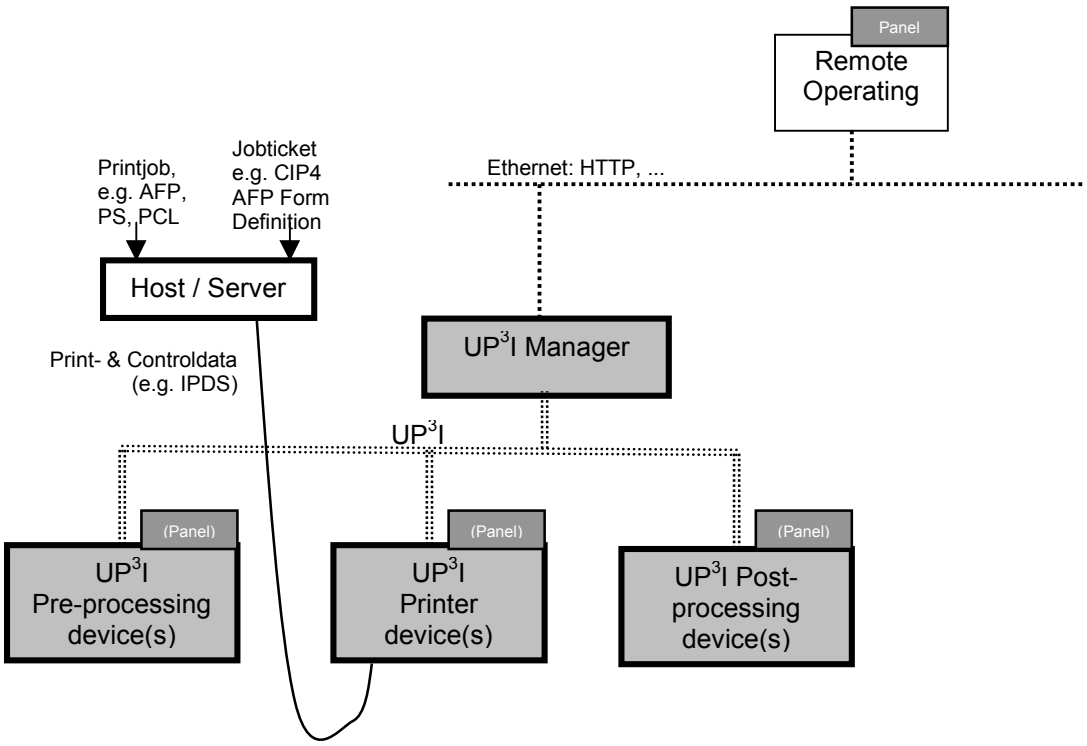


Figure 1-3: UP³I with Remote Operating capability

For additional UP³I functionalities a "UP³I Manager" is necessary.

For remote operation the remote operating station has to connect only to the UP³I Manager. At the remote operating station it is possible to operate the panel remotely and to store / reload complete set-ups (e.g. job corresponding).

Only the UP³I Manager communicates with the tight coupled UP³I devices.

The Host or Remote Operating station have no direct access to the Printer and the Pre- and Post-processing devices.

The UP³I Manager delivers no (IPDS...) Print Data.

The UP³I Manager works as a bridge between host / server, remote operating and printer, pre- and post processing devices. The UP³I Manager may physically be included in a UP³I device (e.g. as part of the printer controller).

1.3.4 UP³I environment with Management capability

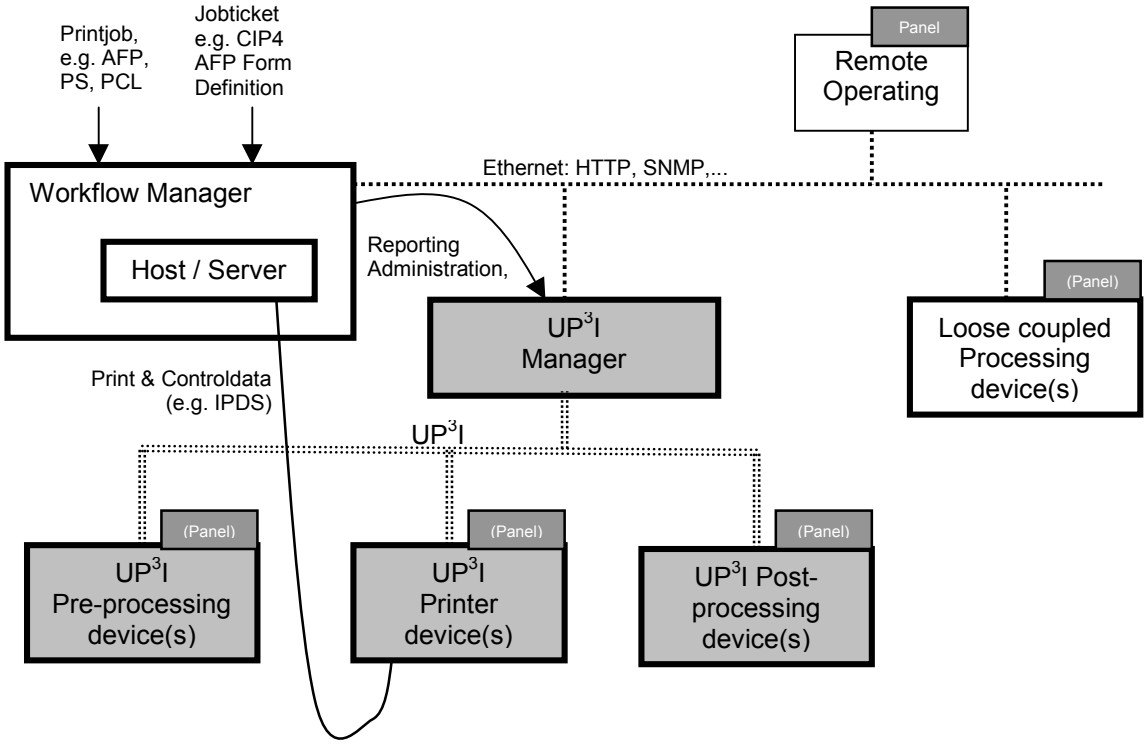


Figure 1-4: UP³I with Management capability

For full management control the workflow manager communicates with the UP³I Manager. Reporting and administrative data is exchanged between them. The UP³I Manager is communicating on the UP³I (real time) side with the UP³I devices using a UP³I specific proprietary protocol.

1.3.5 Near line Processing with UP³I

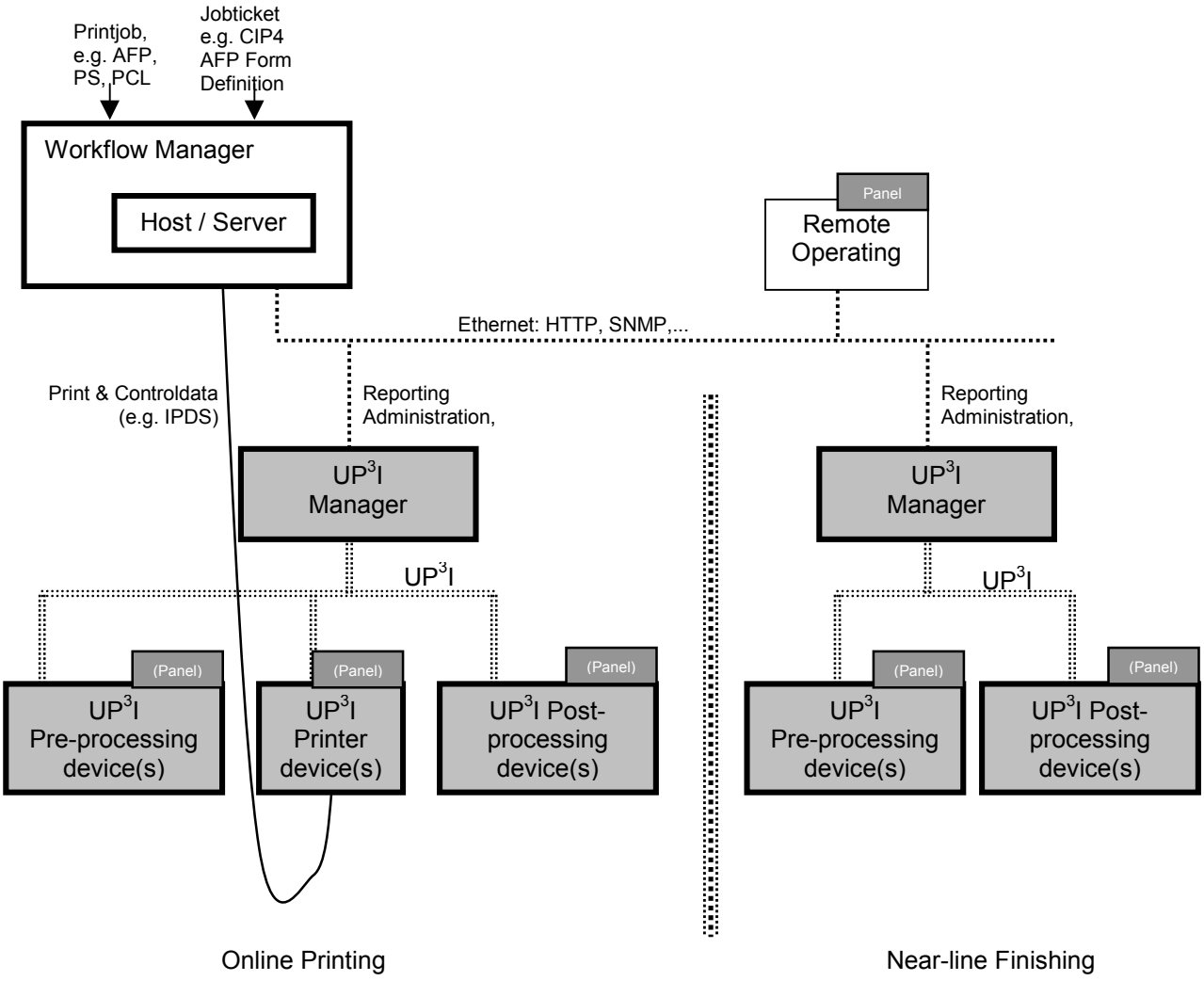


Figure 1-5: Near-line processing with UP³I

For Near line processing two independent steps are to be differentiated:

1.) On-line Printing: Printing (and post-processing) with data from host.
The UP³I Manager does not trace or print additional information!

The printed paper becomes now the input for the “Near-line Finishing“ process.

2.) Near-line Finishing: The processing devices communicate using the UP³I protocol.
The Set-up of the devices can be done via the UP³I Manager by the workflow control.
A dedicated UP³I device has to take the part of the printer (concerning the UP³I frames / communication)

1.3.6 Host Intercommunication

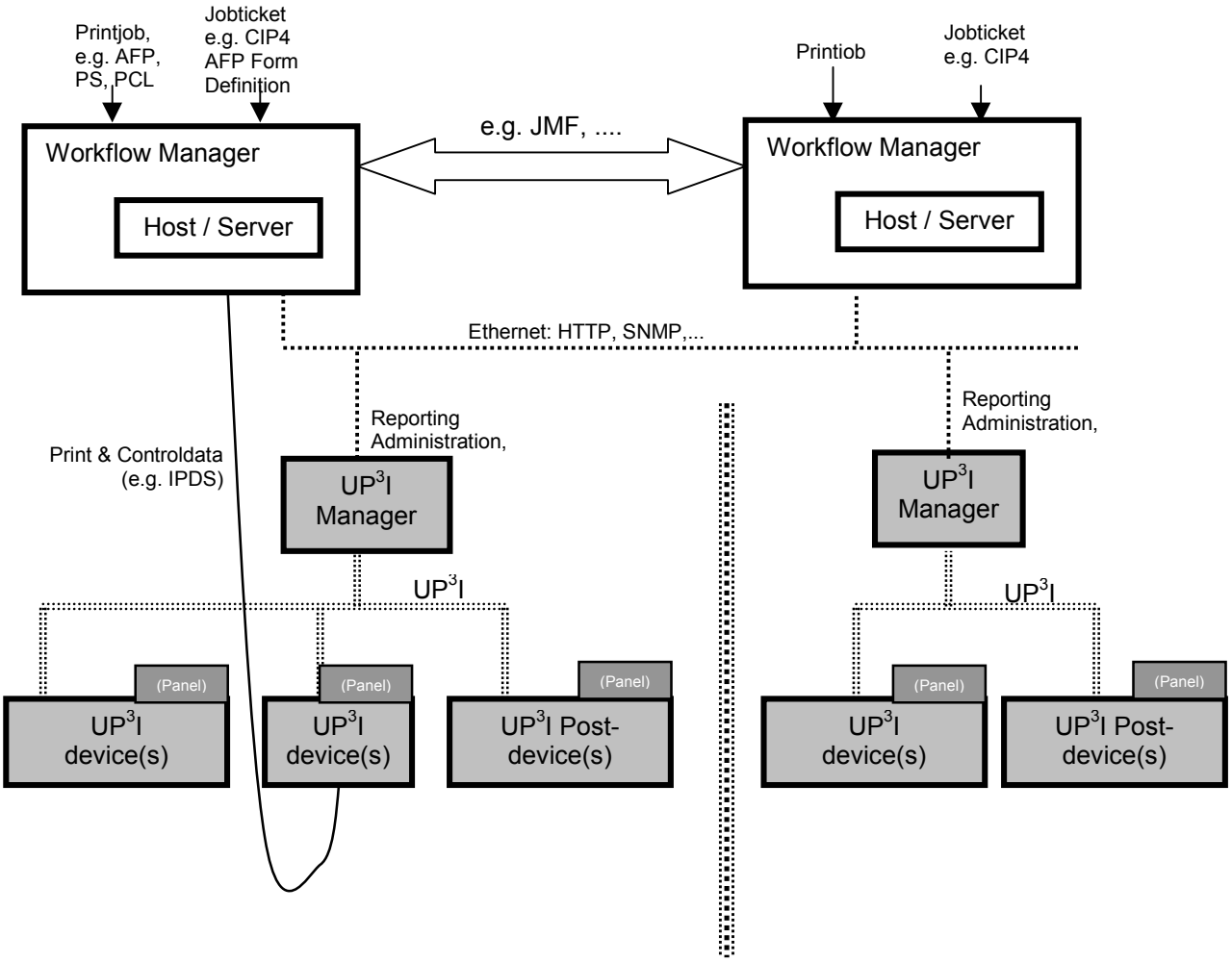


Figure 1-6: Host Intercommunication

Any host intercommunication is independent from UP³I. Therefore the communication between them is open to choice, but from a UP³I point of view JDF (CIP4) is recommended.

2 Real Time Interface

2.1 Tasks of UP³I Real Time Interface

The Real Time Interface...

- connects all **tight coupled UP³I devices** - loose coupled (near-line) devices are not connected with the real time interface.
- is designed for all page relevant information (inclusive of device state Ready, Not Ready...). No (IPDS, PCL...) print data is transferred.
- may also be used to transfer (synchronization) information between single UP³I devices (e.g. synchronization between printer1 and printer2).
- can (easy) be monitored with a standard IEEE 1394 analyzer.

To substitute printed marks (or at least for controlling the production line) the needed page relevant information is transmitted via the real time interface.

A proprietary access to UP³I devices is not allowed. For debugging / service purposes, predefined IEEE1394-Frames are available.

2.2 Concept

2.2.1 UP³I Paper Sequence ID

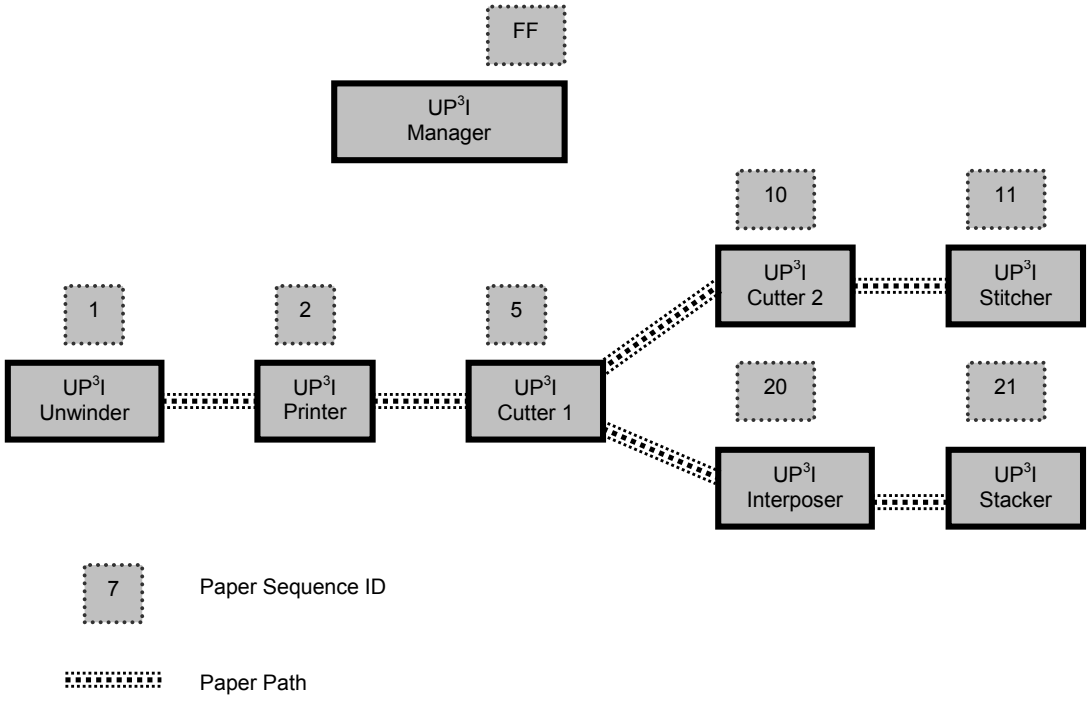
As there is no information about the current paper run sequence each UP³I device needs a manual set-up of its "UP³I Paper Sequence ID" during installation of the production line. The ID has to be unique (within the production line). The range of the UP³I paper sequence ID's runs from 0x01 to 0xFE (0xFF is reserved for the UP³I Manager device, 0x00 is used for broadcasts).

After physically installing the devices of the production line all possible paper path combinations in the process line are described with UP³I tupels. The chain of UP3I Paper Sequence ID's in the tupel will be preferably of an increasing value and must correlate to the true paper path. If there is no branch or pass-through device only one paper run sequence may be available. The description of the paper run sequence may be done using a GUI at the UP³I Manager side - or (e.g. if no UP³I Manager is available) one UP³I device offers the input media (e.g. the operating panel). The tupels have to be stored at this UP³I device.

The paper run sequences are broadcast during bootstrapping – and with correlation to the SDF (self defining field) information (broadcast by all UP³I devices), the current paper run sequence is determined by each UP³I.

If there is a change in the paper run sequence (e.g. a device goes to pass through mode) the changed information is immediately broadcast.

The UP³I device that stores the paper run sequence information may check, if the broadcast form acknowledge frames are in their correct sequence. In case of an error this UP³I device stops the production line.



Possible tupels: 01. 02. 05. 10. 11.
01. 02. 05. 20. 21.

Figure 2-1: Example of UP³I Paper Sequence ID's with a paper branch

2.2.2 UP³I Definition of UP³I "Form", "Page", "Medium Origin", "Front" and "Reverse"

Definition of Form and Page

For a UP³I device a physical sheet of paper is named „form". On each form there may be several (logical) pages printed.

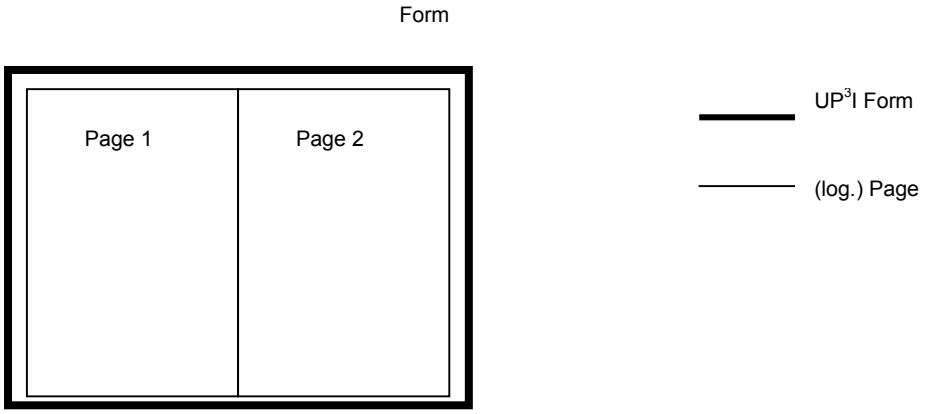


Figure 2-2: Two-Up Example of UP³I Form and Page

Definition of Front Side and Reverse Side of a Form

The front side of a form is the one, that is viewable from the ceiling, the reverse side is visible from the floor. More definitions are necessary, if any devices pass forms in a vertical manner.

Definition of Medium Origin

The following definitions are valid only for Form Exit frames transferred on the UP³I bus between UP³I devices. Form Exit frames in IPDS, PCL or any other data stream use the coordinate system of the respective environment. The transformation from one coordinate system to the other is performed by the printer. The origin for each form and page is the ***top left edge*** in the transport direction of the paper. This is valid for the front side and the reverse side of each form. So the medium origin of the front side is physically different from the medium origin of the reverse side.
After rotation of a medium, the origin persists relating to paper movement direction. Refer to chapter 2.4.3, Form Exit Frame, rotation examples. The Medium Origin of a Form is always the top left corner of a form. It does not depend on the paper length and paper width of the form.
The dimensions for 'Paperlength' and 'Paperwidth' depends also on the paper movement direction. If a sheet is rotated 90 degrees, then the dimension for length and width will become exchanged.

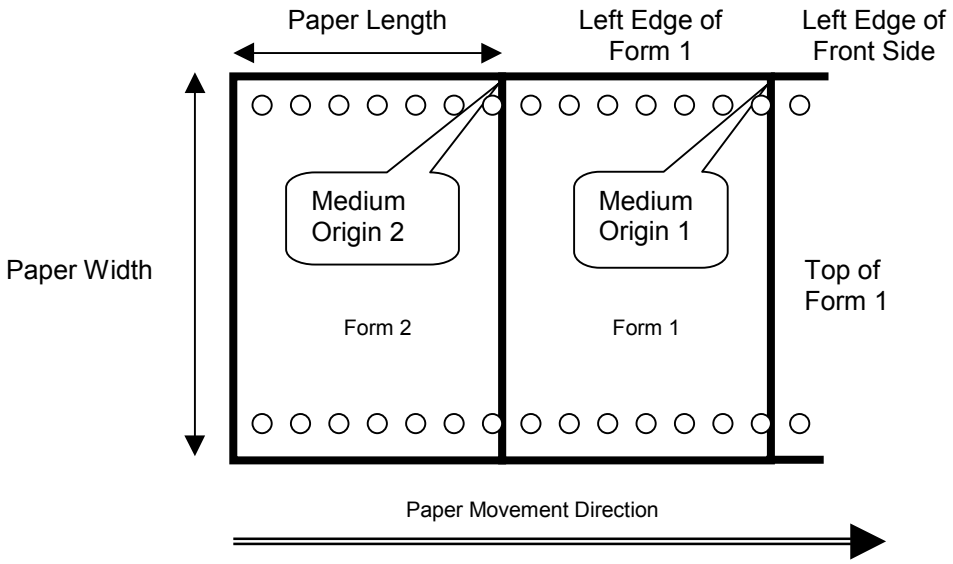


Figure 2-3: Medium Origin Definition, Front Side, View from above

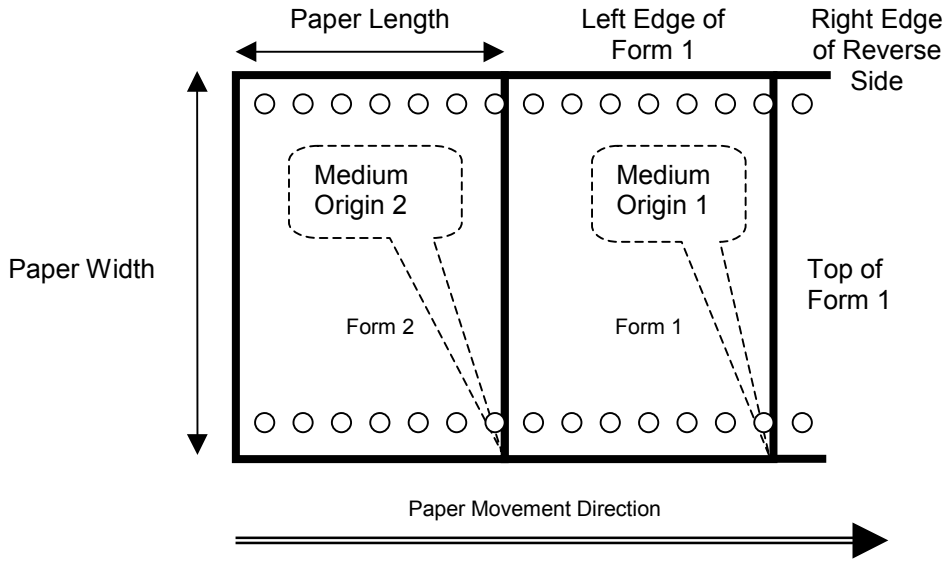


Figure 2-4: Medium Origin Definition, Reverse Side, View from above

2.2.3 UP³I Differentiation of Form, Set and Job

Definition of Form

For a UP³I device a physical sheet of paper is named „form“. On each form there may be several (logical) pages printed (see chapter 2.2.2).

Definition of Set

For a UP³I device a “Set” consists of one or more (logical) pages. All pages within a Set apply to a common finishing operation.
E.g.: a set is used for stitching or stacking with separation offset.

Definition of Job

For a UP³I device a “Job” consists of one or more forms and / or sets. Job Begin and Job End means nothing special for the finishing operations.

Simple Example for Form, Set and Job Handling

This example demonstrates a print job consisting of 5 sheets. All sheets have to be cut from continuous pin less paper. The sheets one to three are stitched together and the sheets four and five are additionally loosely stacked. Finally the complete pile is trimmed. The trailer page (sheet number six) is dedicated to waste.

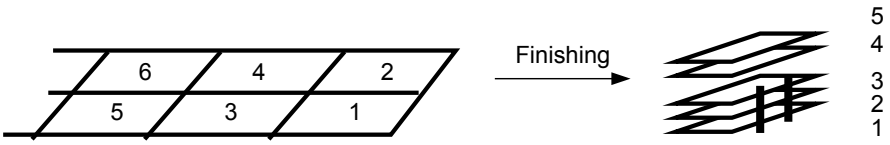
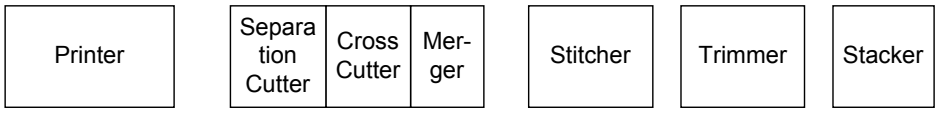


Figure 2-5: Simple Finishing Example

Following production line needs to fulfill this task:



Example 1: (10 --> 20, 30, 40)

The printer will describe this job with following Form exit frames:

1st Form Exit Frame:	Form Size Triplet	Paper Length and Width (= Page 1 & 2)
	Form Finishing Triplet	Dest=20, Finishing = Cut, Separation Cut
	Form Finishing Triplet	Dest=30; Finishing = Cut, Cross Cut
	Form Finishing Triplet	Dest=40; Finishing = Merge, Right first
	Page Triplet Triplet	
	UP³I Page ID Triplet	PAGE_ID = 01
	Page Size Triplet	physical dimensions of Page 1
	Page Sequence Triplet	Tupel = 10 - 20 - 30 - 40 - 60 - 70 - 80
	Start of Job Triplet	Job ID = 01
	Start of Set Triplet	Set ID = 01; Dest=70; Page Finishing = Trim
	Start of Set Triplet	Set ID = 02; Dest=60; Page Finishing = Stitch
	Page Triplet Triplet	
	UP³I Page ID Triplet	PAGE_ID = 02
	Page Size Triplet	physical dimensions of Page 2
	Page Sequence Triplet	Tupel = 10 - 20 - 30 - 40 - 60 - 70 - 80
2nd Form Exit Frame:	Form Size Triplet	Paper Length and Width (= Page 3 & 4)
	Form Finishing Triplet	Dest=20, Finishing = Cut, Separation Cut
	Form Finishing Triplet	Dest=30; Finishing = Cut, Cross Cut
	Form Finishing Triplet	Dest=40; Finishing = Merge, Right first
	Page Triplet Triplet	
	UP³I Page ID Triplet	PAGE_ID = 03
	Page Size Triplet	physical dimensions of Page 3
	Page Sequence Triplet	Tupel = 10 - 20 - 30 - 40 - 60 - 70 - 80
	End of Set Triplet	Set ID = 02
	Page Triplet Triplet	
	UP³I Page ID Triplet	PAGE_ID = 04
	Page Size Triplet	physical dimensions of Page 4
	Page Sequence Triplet	Tupel = 10 - 20 - 30 - 40 - 60 - 70 - 80
3rd Form Exit Frame:	Form Size Triplet	Paper Length and Width (= Page 5 & 6)
	Form Finishing Triplet	Dest=20, Finishing = Cut, Separation Cut
	Form Finishing Triplet	Dest=30; Finishing = Cut, Cross Cut
	Form Finishing Triplet	Dest=40; Finishing = Merge, Right first
	Page Triplet Triplet	
	UP³I Page ID Triplet	PAGE_ID = 05
	Page Size Triplet	physical dimensions of Page 5
	Page Sequence Triplet	Tupel = 10 - 20 - 30 - 40 - 60 - 70 - 80
	End of Set Triplet	Set ID = 01
	End of job Triplet	Job ID = 01
	Page Triplet Triplet	
	UP³I Page ID Triplet	PAGE_ID = 06
	Page Size Triplet	physical dimensions of Page 6
	Page Sequence Triplet	Waste = TRUE

Example 2 (20;30;40 --> 60):

Cutter/Merger to Stitcher

1st Form Exit Frame: Form Size Triplet Paper Length and Width (= Page 1)
Page Triplet Triplet
UP³I Page ID Triplet PAGE_ID = 01
Page Size Triplet physical dimensions of Page 1
Page Sequence Triplet Tupel = 10 - 20 - 30 - 40 - 60 - 70 - 80
Start of Job Triplet Job ID = 01
Start of Set Triplet Set ID = 01; Dest=70; Page Finishing = Trim
Start of Set Triplet Set ID = 02; Dest=60; Page Finishing = Stitch

2nd Form Exit Frame: Form Size Triplet Paper Length and Width (= Page 2)
Page Triplet Triplet
UP³I Page ID Triplet PAGE_ID = 02
Page Size Triplet physical dimensions of Page 2
Page Sequence Triplet Tupel = 10 - 20 - 30 - 40 - 60 - 70 – 80

3rd Form Exit Frame: Form Size Triplet Paper Length and Width (= Page 3)
Page Triplet Triplet
UP³I Page ID Triplet PAGE_ID = 03
Page Size Triplet physical dimensions of Page 3
Page Sequence Triplet Tupel = 10 - 20 - 30 - 40 - 60 - 70 - 80
End of Set Triplet Set ID = 02

4th Form Exit Frame: Form Size Triplet Paper Length and Width (= Page 4)
Page Triplet Triplet
UP³I Page ID Triplet PAGE_ID = 04
Page Size Triplet physical dimensions of Page 4
Page Sequence Triplet Tupel = 10 - 20 - 30 - 40 - 60 - 70 – 80

5th Form Exit Frame: Form Size Triplet Paper Length and Width (= Page 5)
Page Triplet Triplet
UP³I Page ID Triplet PAGE_ID = 05
Page Size Triplet physical dimensions of Page 5
Page Sequence Triplet Tupel = 10 - 20 - 30 - 40 - 60 - 70 - 80
End of Set Triplet Set ID = 01
End of Job Triplet Job ID = 01

6th Form Exit Frame: Form Size Triplet Paper Length and Width (= Page 6)
Page Triplet Triplet
UP³I Page ID Triplet PAGE_ID = 06
Page Size Triplet physical dimensions of Page 6
Page Sequence Triplet Waste = TRUE

Example 3 (60 --> 70):

Stitcher to Trimmer

1st Form Exit Frame: Form Size Triplet Paper Length and Width (= Page 1, 2, 3)
Page Triplet Triplet
UP³I Set ID Triplet
Page Size Triplet physical dimensions of Page 1
Page Sequence Triplet Tupel = 10 - 20 - 30 - 40 - 60 - 70 - 80
Start of Job Triplet Job ID = 01
Start of Set Triplet Set ID = 01; Dest=70; Page Finishing = Trim

2nd Form Exit Frame: Form Size Triplet Paper Length and Width (= Page 4)
Page Triplet Triplet
UP³I Page ID Triplet PAGE_ID = 04
Page Size Triplet physical dimensions of Page 4
Page Sequence Triplet Tupel = 10 - 20 - 30 - 40 - 60 - 70 – 80

3rd Form Exit Frame: Form Size Triplet Paper Length and Width (= Page 5)
Page Triplet Triplet
UP³I Page ID Triplet PAGE_ID = 05
Page Size Triplet physical dimensions of Page 5
Page Sequence Triplet Tupel = 10 - 20 - 30 - 40 - 60 - 70 - 80
End of Set Triplet Set ID = 01
End of job Triplet Job ID = 01

4th Form Exit Frame: Form Size Triplet Paper Length and Width (= Page 6)
Page Triplet Triplet
UP³I Page ID Triplet PAGE_ID = 06
Page Size Triplet physical dimensions of Page 6
Page Sequence Triplet Waste = TRUE

2.2.4 UP³I Page Information Flow

The page finishing information between the coupled UP³I devices is distributed with UP³I Form Exit frames.
The information in the paper path direction is sent from one device to the next (no broadcasting). Any change (like cutting...) of the form / page is observed in the „Form Exit“ Frame.
After form / page processing the new Form Exit Frame(s) is/are sent to the next device (if there is one) **and** an acknowledgement is broadcast to every UP³I device (so the printer knows the place of each paper).
If a malfunction (e.g. paper jam...) occurs the post-processing device does not acknowledge the specific page – but the jam is reported with a state frame and the corresponding UP³I_PAGE_ID.

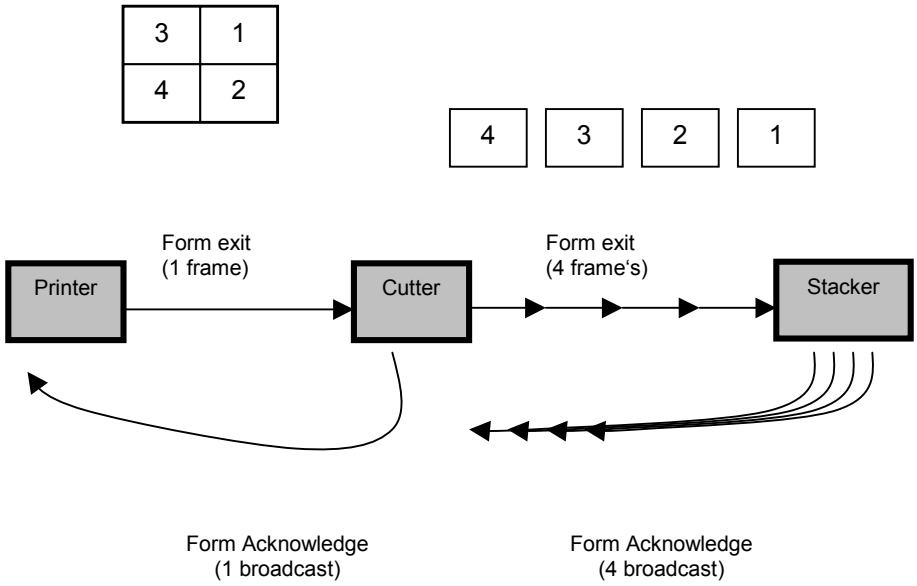


Figure 2-6: Example for UP³I Page Information Flow

Description of the Example:

- In the above example the printer sends one “Form exit“ frame. This frame describes the 4-up form and the four page information. The frame is sent to the next UP³I device (Cutter) when printing the form.
- The cutter device stacks the information.
- As soon as the announced form has been (successfully) processed by the cutter device, a “Paper Acknowledge“ frame is broadcast.
- The cutter device splits the “Form exit“ frame into four frames with the corresponding page information.
- The four frames are immediately sent to the next device in paper sequence direction (Stacker device).
- As soon as a page is stacked the stacker device broadcasts the corresponding “Form Acknowledge“ frame.

The printer reports the successful handling to the host.

2.2.5 UP³I Registration Mark

The UP³I Registration Mark provides the means to position cut marks on the sheet. After printing, these marks can be used by the finishing devices to adapt the theoretical positions (e.g. as specified in the Form Exit Frame) to the real position of the corresponding block on the printed sheet.

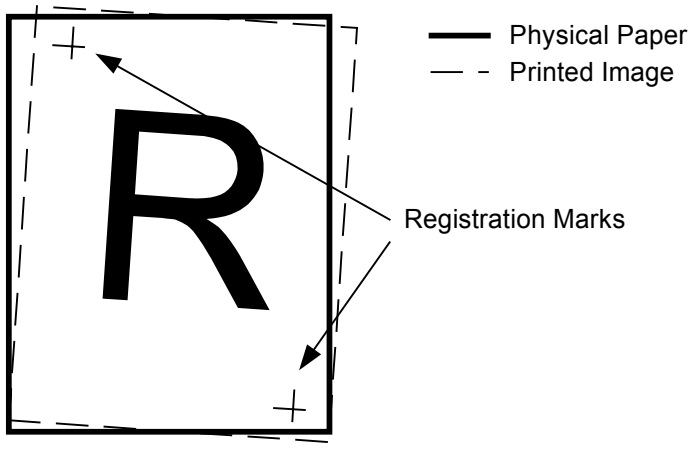


Figure 2-7: UP³I Registration Mark

More than one registration mark may be printed onto a form.

The x/y coordinates of the UP³I frames refer to the medium origin.

The finishing machines may calculate the actual x/y coordinates of the printed image with optical recognition of the registration mark

The registration Mark...

- is optional printed
- is not the PTL mark
- is usually printed (by the printer of the production line) outside of the user printable area (but also may be inside)
- coordinates and the form are sent within the Form Exit frame
- coordinates are related to the logical center of the registration mark

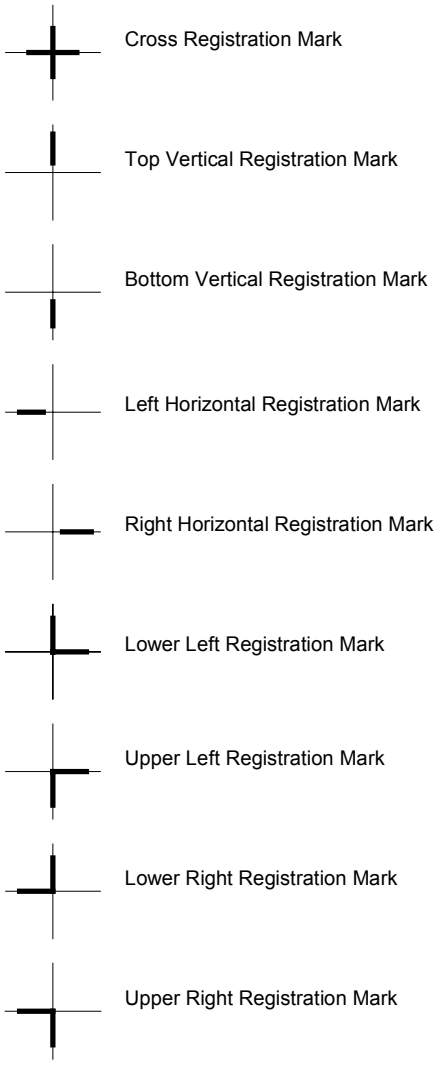


Figure 2-8: Registration Mark Types

2.2.6 Production Line Synchronization / UP³I Mark

To synchronize the information transmitted on the real time interface to the physical paper to which it relates, usually at least one form (page) with an UP³I mark has to be printed. This synchronization procedure has to be performed whenever at least one UP³I device is in asynchronous state (e.g. after power on, after each paper jam, after a new roll). The printed form has to be recognized and decoded by each UP³I post-processing device. The optical recognition may be done manually (by an operator) or – preferred - with an optical reader.

Size and shift of this synchronization mark are sent with the corresponding “Form Exit“ Frame. The post-processing devices have to stack the information until the printed page arrives there.

The position of the synchronization mark is delivered with the corresponding “Form Exit” Frame.

The print job may start being printed at the first available form, if
- it is accepted (by the customer) to have the UP³I mark on the paper, or
- the mark is cut away in the further post process.

Otherwise the printer starts printing the job at the second form and the leading synchronization form is waste.

The UP³I synchronization mark consists of two parts:
- a mandatory synchronization start mark and
- an optional coded sequence mark.

Both UP³I synchronization mark parts are described in the corresponding form exit frame.

If necessary / wanted another mark which is not part of this UP³I specification may be printed (e.g. workflow control systems like IBM Infoprint Workflow, Océ Prisma audit...). This mark may be useful to be read at the subsequent post processing devices (e.g. with rotated paper).

2.2.6.1 Mandatory Synchronization Start Mark

The mandatory UP³I synchronization start mark is at least ½ inch long and at maximum ½ of the page length. The current length is announced in the “Form Exit“ frame. There are slight differences how the mandatory synchronization start mark looks like:

a) Pinless Continuous Paper Printer
Print a “longer“ PTL mark which is in general three times the PTL length.
The width and leading edge position are the same as the normal PTL mark.

b) Tractor (pin feed) Continuous Printer:
Print a mark with the length announced in the “Form Exit“ frame.
At the position announced with the “Form Exit“ frame relative to the top left edge (top view) in paper movement direction.

c) Cut Sheet Printer:
Print a mark which is in general at the length announced with the “Form Exit“ frame
At the position announced with the “Form Exit“ frame relative to the top left edge (top view) in paper movement direction.

2.2.6.2 Optional Coded Sequence

The information, whether the optional coded sequence is printed, is announced in the corresponding form exit frame. This frame contains all needed information.

If printed, the optional coded sequence transports the last four hex digits of the UP^{3}I_PAGE_ID on the physical paper. E.g. if the UP³I_PAGE_ID is 1000 == 0x3E8, the optional coded sequence displays 03E8.The coded sequence is coded with barcode „Code 39“. Find the specification of this well known barcode e.g. at “http://www.barcode-1.net“.

For transparency (e.g. needed with manual operator intervention) the four UP^{3}I_PAGE_ID digits also may be printed.

If the coded sequence is announced, the post-processing devices have to decode (by optical recognition) the physical paper form and verify the synchron state of the production line. Each passing to synchron and asynchron state has to be broadcast with a „Device State“ Frame.

The coded sequence is added for the possibility to identify/verify a synchronization page (e.g. if there is more than one). It is NOT intended for security reasons.

Added information security, print data verifications, front and back correlation, etc. are NOT part of the UP³I proposal.

Synchronization Mark and UP³I Coded Sequence Mark

If a synchronization mark is printed, the preferred location for the UP³I Coded Sequence mark is just “behind“ the synchronization mark:

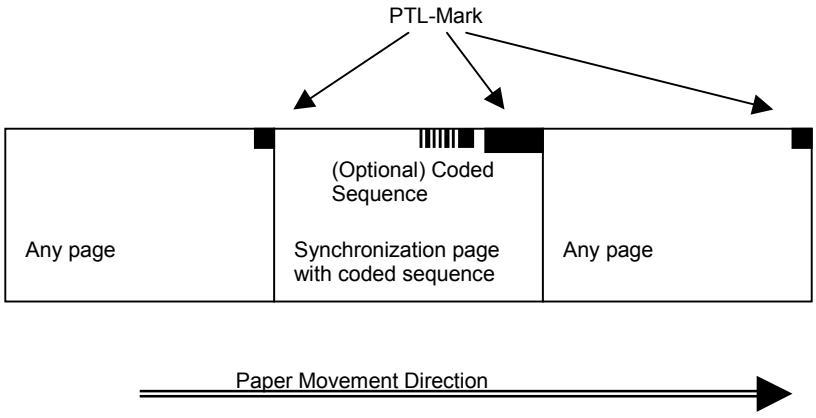


Figure 2-9: Synchronization Mark and UP^{3}I Mark

Definition of the UP³I Coded Sequence Mark

The origin point of each form and page is the top left edge (top view) in paper transport direction.

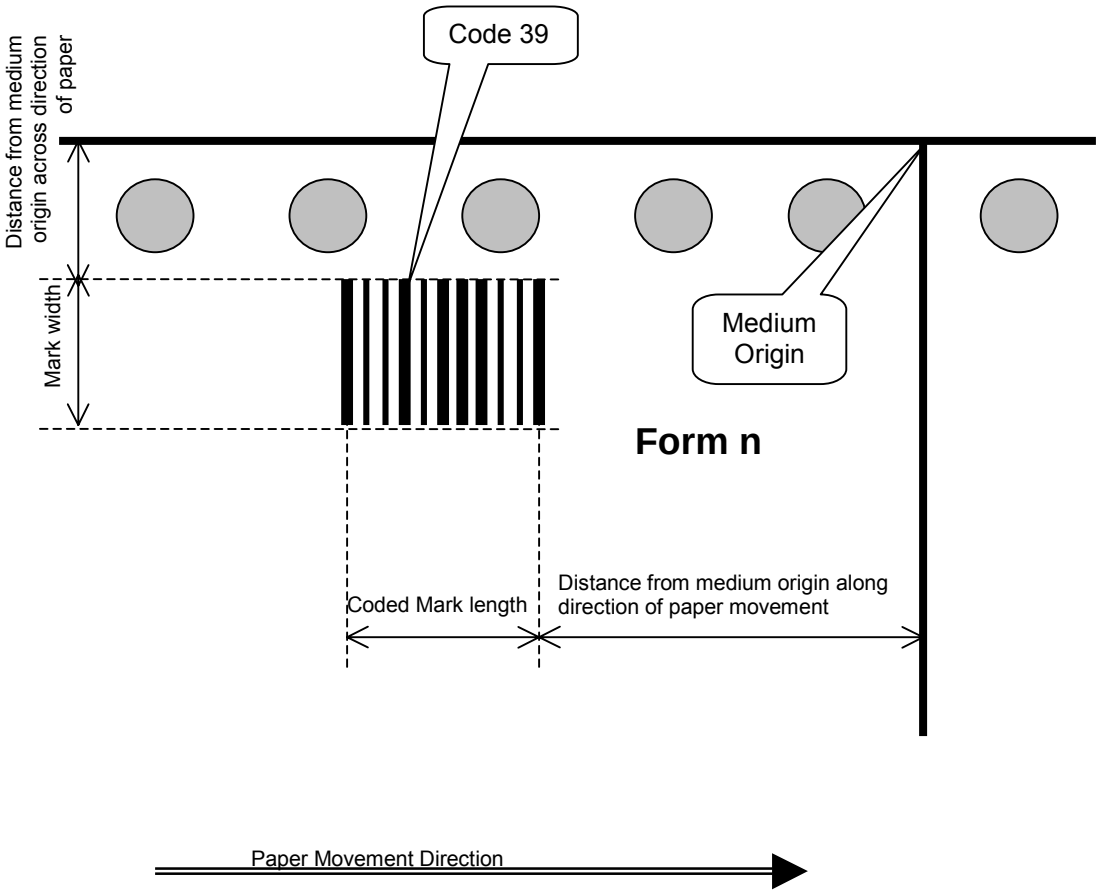


Figure 2-10: UP³I Mark

2.2.6.3 Synchronization Methods

The UP³I synchronization marks may be printed on a form that directly belongs to the print job or on an extra “waste“ form:

a) The UP³I synchronization mark on the information page may be
   - cut away after printing
   - left on the page

b) The UP³I synchronization mark on an extra waste form needs to be thrown away. If no automatic waste bin is available, manual handling needs to be done. The printer sets for this synchronization form the waste flag in the “Form Exit“ Frame.

Remark:

The Synchronization acknowledgement method may be done
- “electrical“ (with readers on post-processing devices) or
- manually acknowledged by the operator.

This is dependent on the specific UP³I devices. It is not relevant for the UP3I device to know how the synchronization is done. The UP³I devices just need to put the synchron information to the real-time interface (Device State Frame).

For enhanced security the UP^{3}I coded sequence may be printed several times on a form (e.g. twice for 2-up printing) because devices behind a cutter also may read the corresponding coded sequence (UP^{3}I_PAGE_ID).

For checking the synchronization regularly, synchronization forms may be printed.

2.2.6.4 Parameters to Set-up the Synchronization Method

The user should be able to decide whether the synchronization mark is printed on every page or not. Therefore set-up parameters (e.g. at the UP³I printer) have to be available.

Below are the necessary parameters shown that need to be known at the printer side. The information is sent with corresponding form exit frames to the post processing devices.

Set-up Parameter	Possible choices	Default
Print Synchronization mark...	• on a waste form • on the job form	⇦
Print "Waste forms" in front of the Synchronization form	0 .. 100 forms	0
Length of the mandatory start synchronization mark	Depends on tractor / pinless web. See above	
Print Optional Coded Sequence	• Yes • No	⇦
Optional Coded Sequence: Width Length Position	Input (millipoint, mm, inch...)	
Tractor Web: Sync Start Mark Length	Input (millipoint, mm, inch...)	
Regular Synchronization Check on every ___ page	0....Not regular 1....100000	⇦

2.2.7 UP³I Error Handling

Each UP³I device reports an error situation by its own with a broadcast message. The UP³I devices “behind“ the jam may (if possible) process the remaining pages. The printer waits until all coupled devices are in “stop“ state and then sends the jam with the valid page counters to the system / host.
After clearing the paper path the system restarts printing. In general it is possible that pages are printed twice (e.g. stacking pages in output bins with different physical distances).

2.2.8 UP³I Emulation device

For some simple paper processing devices it is possible to emulate the UP³I interface task. The emulation need not run directly on this simple machine. The UP³I interface task for the simple device can be emulated e.g. by the UP³I Manager (or any other coupled UP³I device).
This “generic UP³I device driver“ simulates a device with no delay for form (page) transporting.
But at least there must be a way to detect the error conditions from the simple paper processing device.

This emulation method may also be used for processing devices which support only the old Type I, Type II, .. Interface. With this procedure even a simple upgrade to UP³I is possible.

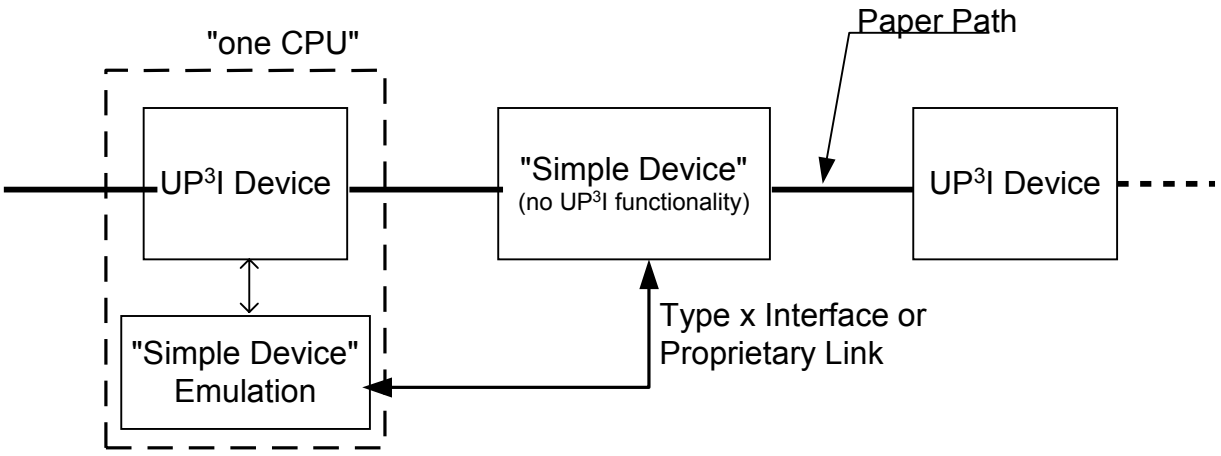


Figure 2-11: Example for UP³I Emulation device

2.2.9 PAC Protocol Concept

The Propose/Accept/Confirm protocol (PAC) is an extension of the standard form exit protocol. The PAC protocol allows the system to confirm that it is possible to send a sheet through the system prior to actually sending it. The PAC protocol is particularly useful in cut sheet systems where various mechanical constraints may require additional delays between sheets in various conditions. For example, a finisher may require extra time after receiving the last sheet of a set while it processes the previous set.

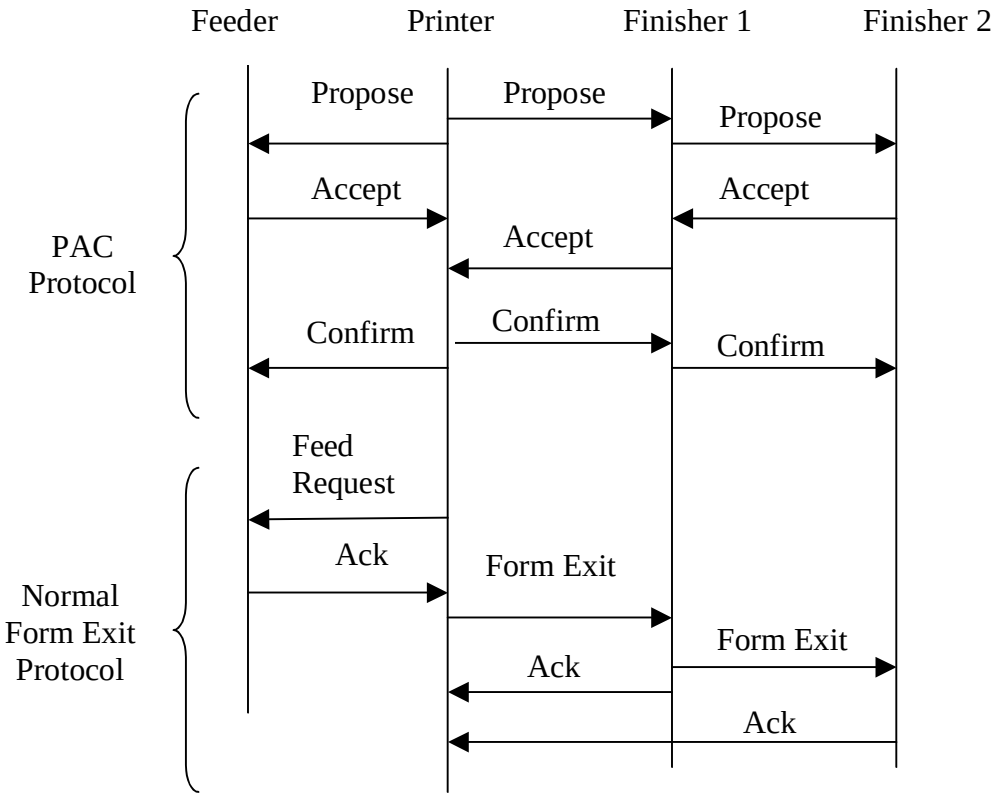
The PAC protocol does not replace the form exit protocol. Rather, it supplements the form exit protocol. For example, regardless of whether PACs are used, devices still send form exit frames downstream when passing sheets, still send ack frames when the sheets are delivered, and still send device state frames when a jam occurs. Also note that the Form Exit information is actually being sent twice – once in the Propose, and then again in the Form Exit. This keeps the Form Exit protocol identical with or without PAC, and allows for a finishing device that can always accept sheets to not have to remember accepted proposals. This can be important for finishing devices with very limited memory capabilities.

tThe PAC protocol is required for all cut-sheet (and set) devices. The PAC protocol is typically initiated by the printer device. The PAC protocol is not used for continuous feed devices. In mixed continous/cut sheet system, the web line cutter device receives only Form Exits from upstream (no PAC involved), and initiates the PAC protocol downstream.

The PAC protocol relies on the IEEE 1394 clock time to define a common time across all devices.

2.2.9.1 Proposing a Capability to Execute

The PAC protocol is a two-phase commit protocol. A simple example is illustrated below. Note that proposals can be transmitted upstream as well (e.g. to propose the feeding of a sheet).



The Proposer *proposes* that the device execute a particular capability at a specific (future) time. This time is called the *reference time*. If the proposal is sent upstream, then the reference time is when the sheet will leave the device (specifically, when the lead edge of the sheet crosses the docking plane). If the proposal is sent downstream, then the reference time is when the lead edge of the sheet will enter the device (specifically, when the lead edge of the sheet crosses the docking plane). Note that if the sheet is long the upstream device may still be processes the sheet at this time.

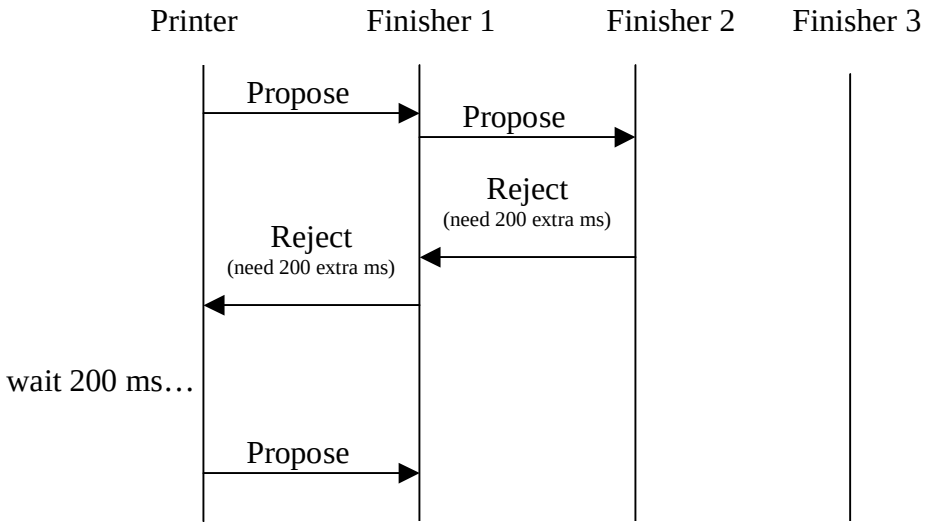
The propose also includes information about commitment, such as the sheet size, and the finishing operation required. This content is the same as in the form exit frame used in the continuous feed protocol.

If the device is able to schedule the execution of that capability when requested, it sends an *accept* to the Proposer. The Proposer then *confirms* that the device should, in fact, execute that capability. The purpose of the second phase of the protocol (i.e., the final *confirm*) from the Proposer is to allow the Proposer to synchronize other devices that must perform some coordinated actions (e.g., feeding a sheet of paper at an appropriate time to synchronize with finishing of the sheet). Finally, when the capability is executed, the device sends an acknowledge frame to the Proposer, same as in the continue feed protocol.

The agreement between the device and the Proposer is referred to as a *commitment*. All commitments created from a single propose are referred to as a *commitment group*. The commitment lifecycle is summarized at the end of this section.

2.2.9.2 Rejecting Commitments

If the device is unable to schedule execution of the requested capability at the specified time, it *rejects* the proposal. In this case, the device specifies information on when the capability could be available. The Proposer will then make a later proposal consistent with the supplied information.



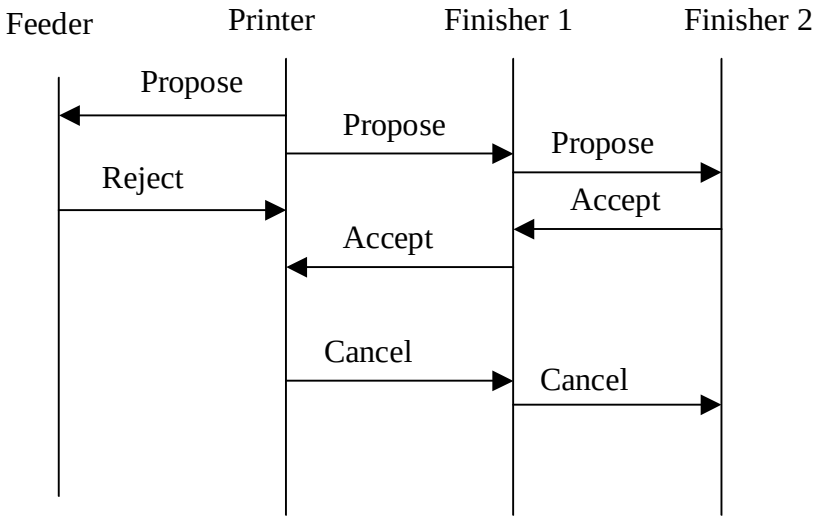
Protocol continues as per normal PAC…

Note that if a device rejects, it does not forward the propose any further downstream. In the example above, since Finisher 2 rejected, it did not forward the propose to Finisher 3.

Another reason that a device may reject a proposal is “commitment overflow”. Devices often can only keep track of a set number of outstanding commitments. If a propose would exceed this number, the device rejects. As the device completes its outstanding commitments it will become able to accept new proposals.

2.2.9.3 Canceling Commitments

If a device has accepted a commitment but another device rejects the commitment, the Proposer sends a *cancel* to all devices who accepted. For example, a marker proposes a feed from an upstream feeder and some finishing from some downstream finishers. If the finishers accept but the feeder rejects, the marker must cancel the proposal to the finishers.



Note: Cancel is sent to devices which already accepted a prior propose.

In the case of a *cancel*, the device proceeds as if the associated *propose* had never occurred. It should not feed the sheet, set up jam checking, etc.
The Proposer may send a *cancel* to a device without having to wait for the device to send *accept*.

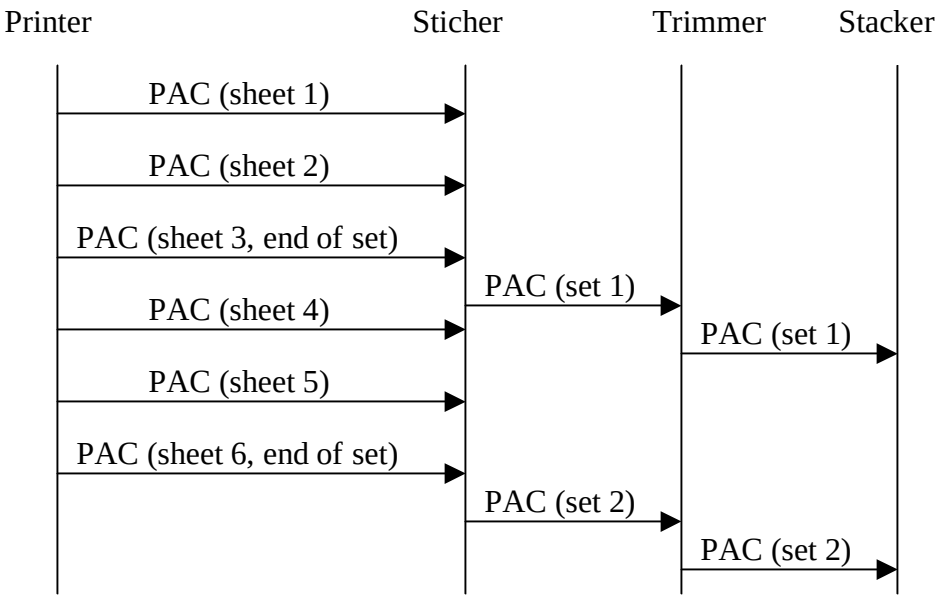
2.2.9.4 Breaking Commitments

If a device has accepted a proposal and later cannot execute the capability, the device must break the commitment. A common use of breaking commitment occurs during a paper jam. A sheet jams in a device, and therefore breaks the sheet's commitment. The device does this by broadcasting the Device State frame, indicating that it is not ready, and passing the sheet ID. This is the same mechanism used in the continuous feed protocol.

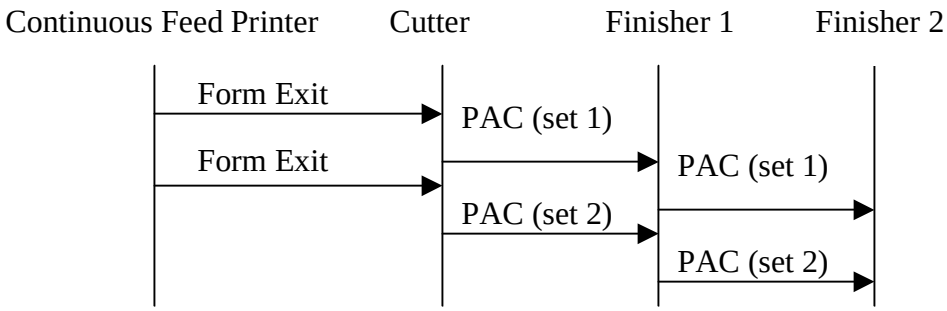
2.2.9.5 Example Systems

In general, PACs are proposed by the device that receives the job information (e.g. a printer) or the device that introduces the sheet or set into the paper path (e.g. a sheet cutter). Below are some examples. Note that in these examples, the entire PAC protocol is abbreviated with a single arrow labled "PAC".

- Compiling Sets. Suppose we have a system with a printer, sticher, trimmer, and stacker. The printer PACs to the feeder and the stitcher. After the stitcher has compiled and stitched a set, the stitcher PACs to the trimmer and stacker



- Mixed continous feed and cut sheet system. In a system with a roll feeder (unwinder), continuous feed printer, cutter, and additional cut sheet finishers, the printer uses the standard form exit protocol to coordinate production across the unwinder and cutter. The cutter then PACs to the downstream finishers.

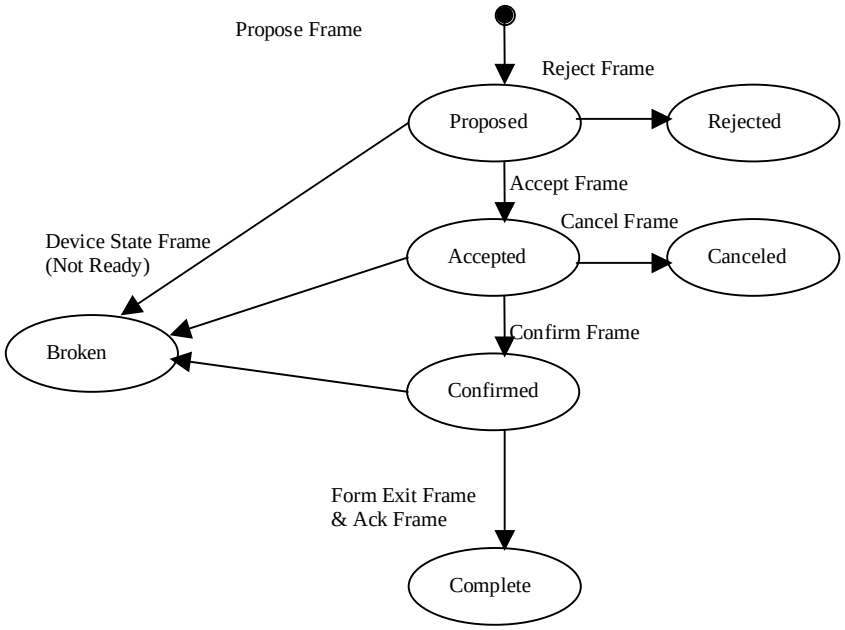


-
-
-
-

Note that the time in the propose is the time at which the leading edge will cross the docking plane into Finisher 1. The FormExit happens after finishing successful procedure

2.2.9.6 Summary of a Commitment Lifecycle

Below is a summary of a commitment lifecycle. Commitments that have not been confirmed are sometimes called *proposals*. Commitment that have been confirmed but not been completed or broken are often called *outstanding commitments*.



Note well: All commitments end their "lifetime" in one of the terminal states – rejected, canceled, broken, or complete. In particular, once confirmed, a device must send either a 'complete' or 'broken' for the commitment depending on whether the commitment execution was successful or not.

The "Device State Not Ready Frame" breaks all (even confirmed) sheet ID's from the mentioned "not ready" devices upstream.

Once a commitment reaches a broken, complete, rejected, or canceled state a device ignores all further calls for that commitment (page ID). The reason is that e.g. by coincidence two independent devices may have a paper jam at the same time.

If a device or Proposer receives a call that violates the state diagram it is an error. Implementations log such occurrences in a debug log for analysis and debug at a later date.

2.2.10 Physical Interface

The tight coupled devices (Printer and the Pre- and Post-processing devices) and the UP³I Manager are connected via the IEEE 1394 Interface.

The IEEE 1394 specification is a document maintained and distributed by the Institute of Electrical and Electronic Engineers (IEEE). Copies of the specification are available from the IEEE. To order a copy, e.g. see http://www.ieee.org or email customer.service@ieee.org.

The UP³I interfaces needs at least IEEE 1394a functionality.

For more information see also this documents Appendix.

2.2.10.1 Interface Cabling

The IEEE 1394 cable technology enables both branching and daisy-chaining of nodes. To guarantee full functionality also with a powered off UP^{3}I device the star cabling method with a central IEEE 1394 hub is preferred.

Note: The UP^{3}I Manager may be included in a UP^{3}I device (e.g. integrated to the UP^{3}I printer device). With reliable cables the specified cable length per segment (at 400 Mbit/s) is limited to a length of 4.5 meters. If there is a longer distance between the UP^{3}I device and the hub a “IEEE 1394 Repeater“ device has to be inserted.

Preferred star connecting example:

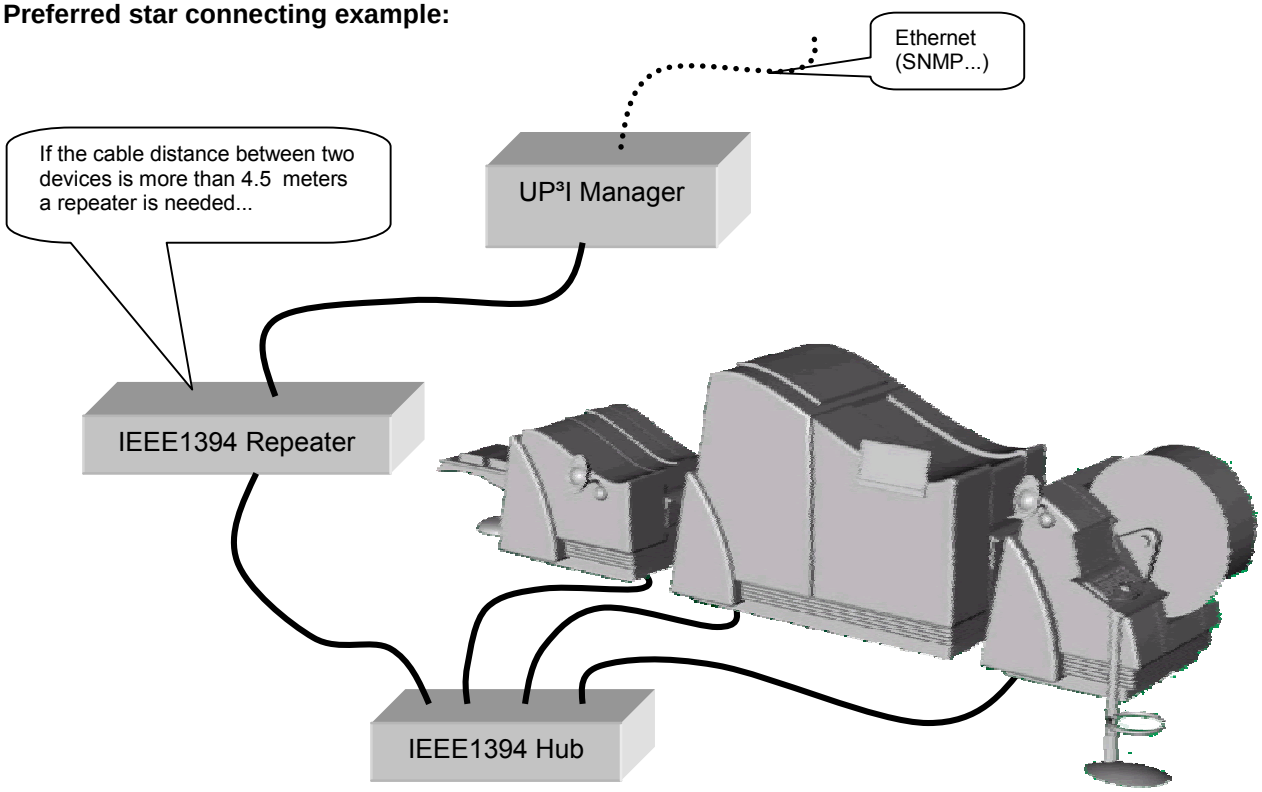


Figure 2-12: UP^{3}I cabling

2.2.10.2 Configuration ROM

All UP³I devices need to support al least the 'Minimal Configuration ROM', as defined in ISO/IEC13213.
Remarks:

(a) The IEEE1394 value “Unit_SW_Version” (key 0x13) is not relevant and should be set to zero. Instead the UP3I SW Version is announced within the “UP³I Version” triplet contained in the “Self Defining Field” frame.
(b) There is no UP³I “OUI” or “Spec ID” available. It is recommended to set these values to zero.
(c) Find a documentation in the IEEE P1212r specification, which extends the ISO/IEC 13213:1994 spec. In this specification OUIs are referred to as "RID" (Registration Authority ID)

2.2.10.3 Cable Power Distribution

UP³I devices must not consume power from the IEEE 1394 bus.

2.2.10.4 Remote Power On

To enable a remote power on functionality (“Wake on IEEE 1394”) it is recommended, that each UP³I device:

(a) accepts IEEE1394 defined Resume Packets
(b) sends IEEE1394 defined Resume Packets during its power up cycle.

All other power changes (Hibernate, PowerOff, Idle) can be modified by sending UP³I’s “Set power state” frame.

2.2.10.5 Isochronous Cycle Clock

It is recommended, that there is no isochronous cycle clock until there is no isochronous data transfer. See section 2.4.7, Paper Movement Frame.

2.2.11 UP³I Frame and Triplet Format

2.2.11.1 IEEE 1394 Asynchronous packet format

The format of an IEEE 1394 asynchronous packet, as specified by clause 6.2.2 of IEEE Std 1394-1995, is illustrated by the following figure.

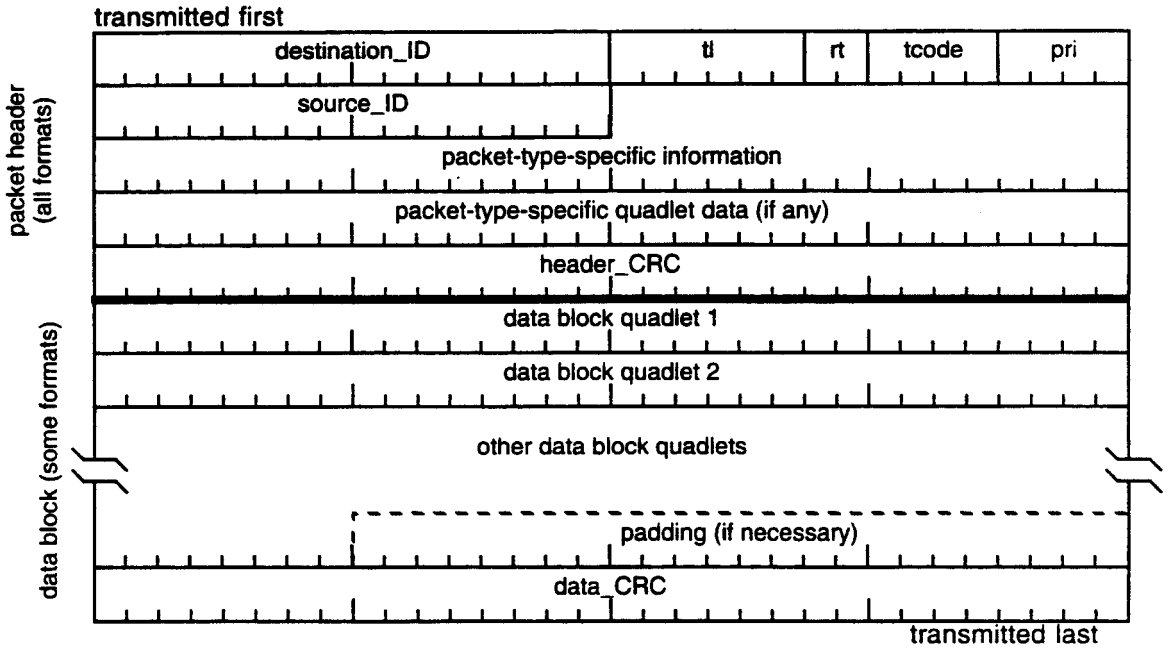


Figure 2-13: IEEE1394 Asynchronous packet format

In the address field (here called packet-type-specific information) the following address is used: FFFF:F0010000. With this start address any packet size is supported.

"Data quadlet" (code: 0x00) and "data block" (code: 0x01) write requests are the possible asynchronous transaction types (tcode field).

The following section has two types of UP3I types (broadcast and dedicated), where the node ID for broadcasts will be 0x3F.Dedicated UP³I frames will use the IEEE 1394 node ID..

2.2.11.2 IEEE 1394 Isochronous packet format

The format of an IEEE 1394 isochronous packet, as specified by clause 6.2.3 of IEEE Std 1394-1995, is illustrated by the following figure.

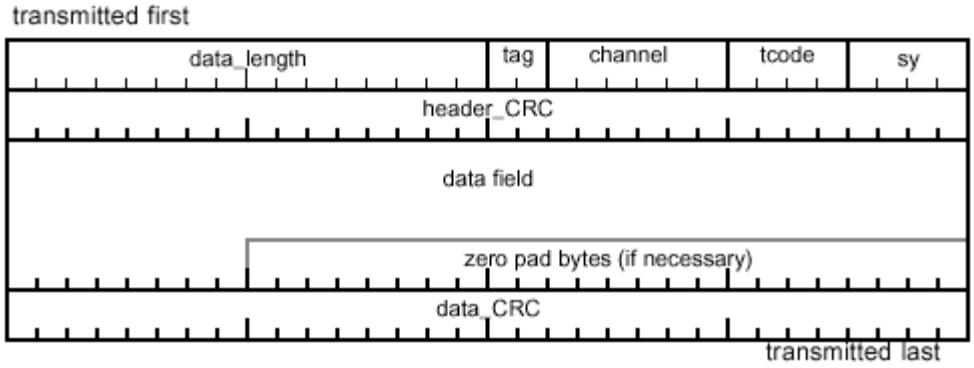


Figure 2-14: IEEE1394 Isochronous data block packet format

2.2.11.3 UP³I Frame Format

The format of a **UP³I Frame** uses the IEEE 1394 defined Frame headers and trailers (CRC).

1. *UP³I Asynchronous Frame:*
   The UP³I specific data is transferred in the IEEE 1394 „data block quadlets“ and illustrated by the following figure

2. *UP³I Isochronous Frame:*
   The UP³I specific data is transferred in the IEEE 1394 „data field“ and illustrated by the following figure:

transmitted first

UP³I Command ID (32 bit)
Optional data (in Triplet format) – find definition below .....

transmitted last

Figure 2-15: UP³I Data Field format

Within a byte, the most significant bit, *ms*b, is that which is transmitted first and the least significant bit, *ls*b, is that which is transmitted last on Serial Bus, as illustrated below.

transmitted first

MSB	interior bits (decreasing significance left to right)	LSB

transmitted last

Figure 2-16: Bit ordering within a byte

Within a quadlet, the most significant byte is that which is transmitted first and the least significant byte is that which is transmitted last on Serial Bus, as shown below.

transmitted first

most significant byte	Second most significant byte	Next to least significant byte	Least significant byte
address + 0	*address* + 1	*address* + 2	*address* + 3

transmitted last

Figure 2-17: Bit ordering within a quadlet

2.2.11.4 UP³I Triplet Format

A triplet is a self-identifying parameter that contains three components.

- the length of the triplet,
- an ID identifying the triplet
- and the associated parameters.

The general format for the triplet data structure is shown below.

Bytes	***Description***	***Value (hex):***
0..1	Length of the Triplet, including this parameter itself	0003 to nnnn
2	Triplet Identifier	00..FF
3..nnnn	Associated parameters (if any)	....

Figure 2-18: UP³I Triplet Format

The triplets appear immediately after the UP^{3}I Command ID. Some frames may contain repeating groups of triplets. An optional triplet may not appear at all.

The mandatory Paper Sequence ID Triplet always has to be the first triplet in every frame. All other triplets need not to be ordered.

2.2.12 Overview of UP³I Frames

Command ID	*Type*	*Meaning*	*Response*	*Page*
000F 0001	Broadcast	Self Defining Field Changed	Self Defining Field Request	51
000F 0002	Broadcast	Self Defining Field Request	Self Defining Field	53
000F 0003	Dedicated	Self Defining Field	(none)	54
000F 0004	Broadcast	Device State Request	Device State	71
000F 0005	Broadcast	Device State	(none)	72
0001 0001	Dedicated	Running Indication	Running Acknowledge	79
0001 0002	Dedicated	Running Acknowledge	(none)	80
0001 0003	Dedicated	Form Exit	Form Acknowledge(s)	87
0001 0004	Dedicated	Print Data	(none)	105
0001 0005	Dedicated	Free Page Information	(none)	108
0001 0006	Isochronous	Paper Movement	(none)	109
0001 0008	Dedicated	Paper Request	(none)	114
0001 0009	Broadcast	Form Acknowledge	(none)	116
0001 000A	Broadcast	Form Invalidation Frame	(none)	103
0001 000C	Dedicated	Estimated Paper Movement Frame	(none)	112
0001 0011	Dedicated	Propose Frame	Accept or Reject	81
0001 0012	Dedicated	Accept Frame	(none)	82
0001 0013	Dedicated	Reject Frame	(none)	83
0001 0014	Broadcast	Confirm Frame	(none)	85
0001 0015	Dedicated	Cancel Frame	(none)	86
0003 0001	Dedicated	EJECT	EJECT Reply	118
0003 0011	Broadcast	EJECT Reply	(none)	119
0003 0002	Dedicated	NPRO	NPRO Reply	120
0003 0012	Broadcast	NPRO Reply	(none)	121
0003 0003	Dedicated	Test Print	Test Print Reply	122
0003 0013	Broadcast	Test Print Reply	(none)	123
0003 0004	Dedicated	Adjust Paper Speed	(none)	124
0005 0001	Dedicated	Get UP³I_PAGE_ID Queue	UP³I_PAGE_ID Queue	125
0005 0011	Dedicated	UP³I_PAGE_ID Queue	(none)	126
0005 0002	Broadcast	Desynchronize (Prod. Line)	Device State	128
0005 0003	Broadcast	Ping	Echo	129
0005 0013	Dedicated	Echo	(none)	130
0005 0004	Dedicated	Set Device State	Device State	131
0005 0005	Dedicated	Get Error Log	Error Log	132
0005 0015	Dedicated	Error Log	(none)	133
0005 0025	Broadcast	Synchronize Error Log	(none)	134
0005 0006	Broadcast	Set Power State	Power State Reply	135

Meaning	*Remark*	*Send*	*Receive*
Self Defining Field Changed	For all devices	M	M
Self Defining Field Request	For all devices	M	M
Self Defining Field	For all devices	M	M
Device State Request	For all processing devices	O	M
Device State	For all processing devices	M	M
Running Indication	For all processing devices	M	M
Running Acknowledge	For all processing devices	M	M
Form Exit	Only Printer and post processing devices	M	M
Print Data	For Printing devices only	O	O
Free Page Information		O	O
Paper Movement	For printer and post processing devices handling web paper	M	O
Estimated Paper Movement Frame	For Web Devices only	O	O
Paper Request	For printer and pre processing devices	M	M
Form Acknowledge	For printer and post processing devices	M	O
Form Invalidation Frame	For printer and post processing devices	O	M
EJECT	For printer and post processing devices handling web paper	O	M
EJECT Reply	For printer and post processing devices handling web paper	M	M
NPRO	For printer and post processing devices handling web paper	O	M
NPRO Reply	For printer and post processing devices handling web paper	M	M
Test Print		O	M
Test Print Reply	For all processing devices	M	O
Adjust Paper Speed		O	O
Get UP^{3}I_PAGE_ID Queue	For all processing devices	O	M
UP^{3}I_PAGE_ID Queue	For all processing devices	M	O
Desynchronize (Prod. Line)	For printer and post processing devices handling web paper	O	M
Ping	For all devices	O	M
Echo	For all devices	M	O
Set Device State	For all processing devices	O	M
Get Error Log	For all devices	O	M
Get Error Log	Manager	M	M
Error Log	For all devices	M	O

Meaning	*Remark*	*Send*	*Receive*
Error Log	Manager	O	M
Synchronize Error Log	For all devices	O	M
Set Power State	For all devices	O	M
Power State Reply	For all devices	M	O
Get Current Power State	For all devices	O	M
Information Management	For all devices with GUI	M	M
Error Detected	For all devices	M	M
File Request		O	O
File	For all devices	O	O
Long Frame Packet	For all devices	M	M

Where a frame is indicated as mandatory the remarks field describes the devices for which it applies. Otherwise it is optional.
Receive: Mandatory: The device must understand that frame and send a relating answer.
Send: Mandatory: The device must be able to send this frame, either as an answer or as a reaction on a relating event.

2.3 Bootstrapping / Self Defining Field Frames

Each UP³I device may learn what kind of peripheral devices are connected to the real-time interface in which (paper) sequence. Each UP³I device has to fulfill a minimum requirement of learning the UP³I paper sequence device ID's of the preceding and following devices.

To obtain this information the UP³I device sends a „Self Defining Field Request" broadcast frame. All attached UP³I devices reply to this broadcast message with a „Self Defining Field" frame.

The printer must be ready to present the host with a so called Finishing Operations Self Defining Field (SDF). Additionally, the Printer has to present it's SDF as well. More device specific information may be obtained by SNMP (by using the UP³I Manager).

2.3.1 Self Defining Field Changed Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x000F 0001
Information Triplet[..]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

Each UP³I device sends this broadcast after a successful

- Power On
- IEEE 1394 Bus Reset, or
- change in the Self Defining Field (SDF).

A UP³I device that is interested in the changed configuration then reacts with broadcasting "Self Defining Field Request".

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including this field	0005
2	Paper Sequence Triplet ID	01
3	UP³I Paper Sequence ID (of the broadcasting UP³I device)	..
4	not used for broadcasts	00

SDF Changed cause Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including this field	0004
2	SDF Changed cause Triplet ID	02
3	Cause:	
	Power down	01
	Boot	02
	Value changed	03
	reserved	00, 04 - FF

2.3.2 Self Defining Field Request Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x000F 0002
Information Triplet[...]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

Each UP³I device may send this broadcast after

- Power On
- each IEEE 1394 Bus Reset , or
- Receiving a “Self Defining Field Changed“ broadcast

All UP³I devices then react within 5 seconds by sending „Self Defining Field“ Frames to the requesting device.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including this field	0005
2	Paper Sequence Triplet ID	01
3	UP³I Paper Sequence ID (of the broadcasting UP³I device)	..
4	not used for broadcasts	00

2.3.3 Self Defining Field Frame

Frame Format: „Self Defining Field“

transmitted first

UP³I Command ID (32 bit) = 0x000F 0003
Information Triplets[...]

transmitted last

Asynchronous packet

Frame Information Flow:

Each UP³I device sends this frame as an answer to a “Self Defining Field Request“ broadcast to the broadcasting device within 5 seconds.

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	Paper Sequence Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

UP³I Version Triplet **(mandatory)**

Further UP³I Specifications will have an increased UP³I Version number. The UP³I Version number is the UP³I superior number of the corresponding specification.

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Version Triplet ID	02
3	UP³I Version (according supported UP³I version)	..
4	UP³I Release (according supported UP³I release)	..

Paper Sequence Tupel Triplet **(optional / mandatory)**

This information triplet is sent by the UP³I device which stored the possible paper run sequences (e.g. the UP³I Manager) during production line set-up. The triplet is repeated according to the amount of available paper run sequences. Find more information in this document at chapter "UP³I Paper Sequence ID".

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	0004 – 00FF
2	Paper Sequence Tupel triplet ID	03
3 – n	Paper Sequence Tupel	.. ..

Paper Sequence Extended Tupel Triplet **(optional / mandatory)**

This Triplet replaces the paper Sequence Tupel Triplet. New implementations should use this triplet.
In this triplet, there is an ASCII string added, to give each tupel a name. This name may e.g. used by the host to address the tupels.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	0007 – m+1
2	Paper Sequence Extended Tupel triplet ID	19
3	Number of PaperSequenceIDs in this tupel	
4 – n	Paper Sequence Tupel	.. ..
n+1 - m	ASCII String, that names this tupel	

Device Type Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004
2	Device Type Triplet ID	04
3	Processing Device Type	
	Pre-Processing Device	01
	Printer Device	02
	Post-Processing Device	03
	non processing device (e.g. UP³I Manager)	04
	reserved	00, 05 - FF

Self Defining Field Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004
2	Self Defining Field Triplet ID	05
3	Device Type: Identical UP³I devices (e.g. Twin Printers) are differentiated by the Paper Sequence ID)	
	Printer	
	Front Page Printer	01
	Rear Page Printer	02
	Duplex Printer	03
	~~Postprocessing printer (e.g. MICR, color, ...)~~	~~04~~
	Paper Input Feeder / Cover Feeder	05
	Postprocessing Printer Front side	06
	Postprocessing Printer Reverse side	07

Bytes	*Description*	*Value (hex):*
	Postprocessing Printer Duplex	08
	Winder	
	Unwinder	10
	Rewinder	11
	Folder	
	V-Folder (Centre-fold-in)	20
	Z-Folder	21
	Spiral / Continuous -Folder	22
	Other Folder (more Info in “Additional-Info-triplet”)	23
	Stapler / Stitcher	
	Corner Staple	30
	Saddle Stitch In	31
	Saddle Stitch Out	32
	Edge Stitch	33
	Cutter	
	Separation cut	40
	Perforation cut	41
	Burster	42
	Cross Cutter	44
	Trimmer	
	Front Edge	50
	3 Edge	51
	5 Edge	52
	Banding Device	
	Single Band Wrap	60
	Double Band Wrap	61
	Cross Band Wrap	62
	Shrink Wrapper	
	Shrink Wrapper	68
	Table (e.g. conveyor table, destination)	70
	Bypass (a device that does not support any finishing)	7F
	Offset / Group Separator / Job Separator	80
	Stacker	81
	Rotator	82
	Accumulator Buffer	83
	Puncher	84
	Binder	85
	Merger	86
	Booklet Maker	87
	Inserting System	89
	Interposer	8A
	Other generic UP^{3}I device	90

Bytes	*Description*	*Value (hex):*
	UP^{3}I Manager	FF

Bypass Device:
The original aim of bypassing is that if a device is broken (i.e. can not fulfill its normal function), then it is bypassed to allow the rest of the process line to keep working. This can be done at the GUI by selecting 'bypass' or 'disable function'. The device now becomes a device type 'Bypass' and a new SDF is broadcast. The bypass device continues to support the Ready/not ready signals as a bypass. If a device is completely out of action or powered off, then at the GUI the device is selected as 'disabled' and the tupel is removed from the active list.
A bypassed device:

- Is typically not a totally failed (broken) device, but is still capable of transporting paper to the following device.
- Uses a paper sequence ID which is included in the process path tupel
- Has the finishing operation 'bypass' only.
- Continues sending and receiving frames, such as form acknowledge and form exit.

The state of a bypassed device:

- Can be ready although some unused finishing parts are broken (see: "the aim of bypassing").
- Refers to the ability to bypass

Paper Input Format Triplet **(mandatory)**
Optional for non processing devices (see Device Type triplet, Byte 3).

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004
2	Paper Input Format Triplet ID	06
3	Paper Input Format	
	Continuous form	01
	Cut sheet (single sheets)	02
	Group / Set (gathered sheets)	03
	No Input (e.g. Unwinder, Sheet feeder device)	04
	Reserved	all other

Paper Output Format Triplet **(mandatory)**
Optional for non processing devices (see Device Type triplet, Byte 3).

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004
2	Paper Output Format Triplet ID	07
3	Paper Output Format	
	Continuous form	01
	Cut sheet (single sheets)	02
	Group / Set (gathered sheets)	03
	No Output (e.g. Rewinder, Stacker device)	04
	reserved	all other

UP³I Product ID Triplet (mandatory)

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, incl. this field	0073
2	UP³I Product ID Triplet ID	08
3..18	Vendor ID String (ASCII string)	....
19..50	Vendor Device Type Name String (ASCII string)	....
51..82	Unique Identifier / Serial Number (ASCII string)	....
83..114	Engineering level / Software Release / Version (ASCII string)	....

All following mandatory triplets are optional for non processing devices (see Device Type triplet, Byte 3).

Additional Device Information Triplet (optional)

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	0004 – 00FF
2	Additional device information triplet ID	09
3 - n	*Additional device specific information*	.. ..

Paper Input Media Triplet (optional)

Relevant only with Paper Input Feeder and Interposer. For new implementations use the next Paper Input Media Extended Triplet.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself and all (optional) added Paper Input Media sub triplets	Xxxx
2	Paper Input Media Triplet ID	0A
3	Input Media ID	..
4..xxxx	*Paper Input Media ... Sub Triplets (as defined below)*	.....

Paper Input Media Extended Triplet (optional)

Relevant only with Paper Input Feeder and Interposer

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself and all (optional) added Paper Input Media sub triplets	Xxxx
2	Paper Input Media Triplet ID	18
3..6	Input Media ID	..
7..xxxx	*Paper Input Media ... Sub Triplets (as defined below)*	.....

Paper Input Media Name Sub Triplet (optional)

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	00xx
2	Paper Input Media Name Sub Triplet ID	01
3 - n	Input Media Name / Paper Name (ASCII string)	.....

Paper Input Media Coating Sub Triplet **(optional)**

What pre-process coating has been applied to the surface of the media

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0005
2	Paper Input Media Coating Sub Triplet ID	02
3	Front Coatings None (the default) Glossy HighGloss Matte Satin Semigloss Reserved	 00 01 02 03 04 05 06..FF
4	Back Coatings Identical to Front Coatings, but applied to the back surface of the media.	..

Paper Input Media Brightness Sub Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0005
2	Paper Input Media Brightness Sub Triplet ID	03
3 - 4	reflectance percentage	00..100

Paper Input Media Color Sub Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0004
2	Paper Input Media Color Sub Triplet ID	04
3	A color resource that provides the color of the chosen medium	..

Paper Input Media Imagable Side Sub Triplet **(optional)**

Side of the medium that may be marked

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0005
2	Paper Input Media Imagable SideSub Triplet ID	05
3 - 4	Possible values are: Front Back Both – Default value. Neither	 01 02 03 04

Paper Input Media Color Name Sub Triplet **(optional)**

Definition examples see JDF specification, chapter 'Named Color'

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0006 – 00FF
2	Paper Input Media Color Name Sub Triplet ID	06
3	Color Prefix 1 Default Clear reserved	 01 02 00, 03 – FF
4	Color Prefix 2 Default Light Dark reserved	 01 02 03 00, 04 - FF
5	A name for the color White Black Gray Red Yellow Green Blue Turquoise Violet Orange Brown Gold Silver No Color Pink Buff Ivory Goldenrod Mustard Custom Color	 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 20
6 – 254	Media's Custome Color as an (ASCII string), only used when byte 5 is Custom Color (leading blanks not allowed).	

Paper Input Media Set Count Sub Triplet **(optional)**

When the input media is grouped in sets, this identifies the number of pieces of media in each set. For example, if the UserMediaType is ‘precuttabs’, a Media Set Count of 5 would indicate that each set includes 5 tab sheets.

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0005
2	Paper Input Media Set Count Sub Triplet ID	07
3..4	Number of sheets/pieces in set if there are no sets, the set count is 1	.. ..

Paper Input Media Opacity Sub Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0004
2	Paper Input Media Opacity Name Sub Triplet ID	08
3	The opacity of the media: Opaque – the media is opaque. The default. Transparent – the media is transparent	 01 02

Paper Input Media Pre Printed Sub Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0004
2	Paper Input Media Pre Printed Sub Triplet ID	09
3	Paper is preprinted: False (the default) True.	 00 01

Paper Input Media Recycled Sub Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0004
2	Paper Input Media Recycled Sub Triplet ID	0A
3	Paper is recycled: False (the default) True.	 00 01

Paper Input Media Roll Diameter Sub Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0007
2	Paper Input Media Roll Diameter Sub Triplet ID	0B
3-6	Roll Diameter in Millipoints	....

Paper Input Media Thickness Sub Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0007
2	Paper Input Media Thickness Sub Triplet ID	0C
3-6	The thickness of the chosen medium. Measured in micron [µm].	....

Paper Input Media User Media Type Sub Triplet **(optional)**

A human-readable description of the type of media. The value can be used by an operator to select the correct media to load. The semantics of the values will be site-specific.

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0004 – 00FF
2	Paper Input Media user Media Type Sub Triplet ID	0D

3	Possible values include:.		
	Continuous	Continuously connected sheets of an opaque Material - which edge is connected is not specified.	01
	Continuous Long	Continuously connected sheets of an opaque material connected along the long edge.	02
	Continuous Short	Continuously connected sheets of an opaque material connected along the short edge.	03
	Envelope	Envelopes that can be used for conventional mailing purposes.	04
	Envelope Plain	Envelopes that are not preprinted and have no windows.	05
	Envelope Window	Envelopes that have windows for addressing purposes.	06
	Full Cut Tabs	Media with a tab that runs the full length of the medium so that only one tab is visible extending out beyond the edge of non-tabbed media.	07
	Labels	Label stock [For example, a sheet of peel-off labels].	08
	Letterhead	Separately cut sheets of an opaque material including a letterhead.	09
	Multi Layer	Form medium composed of multiple layers which are pre-attached to one another; e.g., for use with impact printers.	0A
	Multi Part Form	Form medium composed of multiple layers not pre-attached to one another; each sheet may be drawn separately from an input source.	0B
	Photographic	Separately cut sheets of an opaque material to produce photographic quality images.	0C
	Pre Cut Tabs	Media with tabs that are cut so that more than one tab is visible extending out beyond the edge of non-tabbed media.	0D
	Stationery	Separately cut sheets of an opaque material	0E
	Tab Stock	Media with tabs (either pre-cut or full-cut).	0F
	Transparency	Separately cut sheets of a transparent material.	10
	Plain	Plain sheet	11
	Custom Type		FF
	Reserved		12..FE
4 - 254	This field provides the name of the Media's Custome Type (ASCII string) only used when byte 3 contains 0xFF (leading blanks are not allowed)		

Paper Input Media Weight Sub Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0007
2	Paper Input Media Weight Sub Triplet ID	0E
3-6	Weight of the chosen medium. Measured in grams per square meter [g/m²].	....

Paper Input Media Pin Hole Sub Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0004
2	Paper Input Media Pin Hole Sub Triplet ID	0F
3	Possible values include:. Continuous-forms media with pin holes Continuous-forms media without pin holes Reserved	 01 02 All other

Paper Input Media Dimensions Sub Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	000B
2	Paper Input Media Dimensions Sub Triplet ID	10
3..6	Paper Width in millipoints (=1/72000 inch)	xxxx xxxx
7..10	Paper Length in millipoints (=1/72000 inch): Continuous-forms media Cut-sheet media	 0000 0000 xxxx xxxx

Paper Input Media Ordered Set Piece Sub Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0005
2	Paper Input Media Ordered Set Piece Triplet ID	11
3..4	Input Media Ordered Set Piece, between 1 and the value specified in Paper Input Media Set Count Sub Triplet, bytes 3..4.	1..xxxx

Paper Input Media Predrilled Hole Count Sub Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	0005
2	Paper Input Media Ordered Set Piece Triplet ID	12
3..4	number of predrilled holes in input media	0..xxxx

~~Post-processing Handling Time Triplet~~ **~~(optional)~~**

Not longer in use.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	xxxx
2	triplet ID reserved for compatibility purpose	0B

Device Recovery Method Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004
2	Device Recovery Method Triplet ID	0D
3	Device Recovery Method Form recovery Set recovery Job recovery	 01 02 03

Paper Movement Time Triplet **(mandatory)**

This Triplet announces the time interval, in which the device will send Paper Movement Frames. Zero Milliseconds means that no Paper Movement Frames will be sent.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	0007
2	Paper Movement Distance triplet ID	0E
3..6	Paper Movement Distance in Milliseconds 00... no Paper Movement Frames will be sent	....

Shaft Distance Triplet **(optional)**

Relevant only for devices with Cut Sheet Paper Output Format

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	000F
2	Shaft Distance triplet ID	0F
3 – 6	Distance from last driven shaft to paper exit point in millipoints (=1/72000 inch)	....
7 – 10	Distance from last shaft to paper exit point in millipoints	....
11 – 14	Paper exit velocity at paper exit point in millipoint per second	....

Device GUI Init Class Name Triplet **(optional)**

This Triplet announces the class name of the device GUI class, which implements the interface InitDeviceGuiInterface. The class name itself has to be unique and therefore should comply with common Java habits. Like com.vendor. _____" (E.g.: "com.oce.ops.printer.panel.InitDeviceGuiImpl")

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	0005..00FF
2	Device GUI Init Class Name triplet ID	10
3..nnn	Class Name (ASCII string)(max. 252 character)	....

Device GUI Jar File Name Triplet **(optional)**

This Triplet announces the name of the jar file which can include all classes, images, help files etc. of the Device GUI.
If there is more than one jar file for the GUI, multiple "Device GUI Jar File Name Triplets" are used.
The order of these multiple "Device GUI Jar File Name Triplets" defines the classpath.
Jar file name is a URL, if the file is requested via a server.
If the location is "At the UP3I Device", the named jar files are requested with the "File Request" frame.
A path is not necessary, if the filename is clearly indicated.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	000C..00FF
2	Device GUI Jar File Name triplet ID	11
3..4	Device GUI Jar File Date: Year: yyyy (e.g.2004)	….
5	Month: Mm (e.g. 01)	..
6	Day: Dd (e.g. 26)	..
7	Hour: Hh (e.g. 10)	..
8	Minute: Mm (e.g. 58)	..
9	Second: Sec (e.g. 17) If the File Date is unknown, all values are zero. The example File Date is 26th of January, 10:58:17	..
10	Device GUI Jar File Location: At the UP³I device Installed at the UP³I Manager Available on the Internet	 01 02 03
11..nnn	Jar File Name (ASCII string) Corresponding to the Device GUI Jar File Location this ASCII string may be an Internet address or file name (inclusive the full path if the file is at the UP3I device file system). The type of separators is either slash or backslash .e.g. http://xxx or ftp://yyy or c:\dir\...(max. 244 character)	....

Device GUI Init Class Name Extended Triplet (optional)

This Triplet announces the class name of the device GUI class, which implements the interface InitDeviceGuiInterface. The class name itself has to be unique and therefore should comply with common Java habits. Like com.vendor. _____" (E.g.: "com.oce.ops.printer.panel.InitDeviceGuiImpl")

This Triplet is the extended version of the 'Device GUI Init Class Name Triplet', expanded by GUI Type. For new applications use this triplet only.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	0009 .. 00FF
2	Device GUI Init Class Name Extended triplet ID	12
3..6	GUI Type (used bit by bit) Device GUI Service GUI reserved	 01 02 00, 04..FF
7..nn	Class Name (ASCII string) (max. 248 character)	....

Device GUI Jar File Name Extended Triplet (optional)

This Triplet announces the name of the jar file which can include all classes, images, help files etc. of the Device GUI.
If there is more than one jar file for the GUI, multiple "Device GUI Jar File Name Triplets" are used.
The order of these multiple "Device GUI Jar File Name Triplets" defines the classpath.
Jar file name is a URL, if the file is requested via a server.
If the location is "At the UP3I Device", the named jar files are requested with the "File Request" frame.
A path is not necessary, if the filename is clearly indicated.

This Triplet is the extended version of the 'Device GUI Jar File Name Triplet', expanded by GUI Type. For new applications use this triplet only.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	0010..00FF
2	Device GUI Jar File Name Extended triplet ID	13
3..4	Device GUI Jar File Date: Year: yyyy	….
5	Month: Mm	..
6	Day: Dd	..
7	Hour: Hh	..
8	Minute: Mm	..
9	Second: Sec If the File Date is unknown, all values are zero. Format see Device GUI Jar File Name Triplet	..
10	Device GUI Jar File Location: At the UP³I device Installed at the UP³I Manager Available on the Internet	 01 02 03
11..14	GUI Type (used bit by bit) Device GUI Service GUI reserved	 01 02 00, 04 .. FFFFFFFF
15..FF	Jar File Name (ASCII string) Corresponding to the Device GUI Jar File Location this ASCII string may be an Internet address or file name (inclusive the full path if the file is at the UP3I device file system). The type of separators is either slash or backslash .e.g. http://xxx or ftp://yyy or c:\dir\...(max. 240 character)	....

Caution: For details regarding GUI clustering see chapter 4.4.7, Device GUI Clustering.

~~**Paper Arrival Time Triplet (optional)**~~

Not longer in use.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	xxxx
2	~~Paper Arrival Time triplet ID~~ (reserved for compatibility purpose).	14

~~Tupel Changing Time Triplet~~ ~~(optional)~~

Not longer in use

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	xxxx
2	~~Tupel Changing Time triplet~~ ID (reserved for compatibility purpose)	15

Tupel Cluster Triplet **(optional)**

Sometimes it is requested to connect two or more devices. In particular different output bins are often treated as a single one to support an uninterruptible print. In UP³I output bins are treated as tupels, so with this triplet two or more tupels can be clustered.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	m +1
2	Tupel Cluster triplet ID	16
3	Number of 1st Tupel Paper Sequence IDs	..
4 .. n	List of 1st Tupel Paper Sequence IDs	
n+1	Number of 2nd Tupel Paper Sequence IDs	
n+2 .. m	List of 2nd Tupel Paper Sequence IDs	

Postprocessing Printer Print Data Format ID Triplet **(optional)**

This triplet describes, which Print Data Format Ids are supported by this printer. It is mandatory for postprocessing printers.

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	08
2	Postprocessing Printer Print Data Format ID Triplet ID	17
3..6	Print Data Format ID (PDFID) Bar Code (BCOCA) support xx describes the supported bar codes (see list below). Reserved	 BCy000xx All others
7	Mapping Option The upper left corner of the bar code object presentation space is positionedat an offset from the UP³I medium origin. The offset is specified by configuration parameters set in the UP³I device reserved	70 All others

y (hex)	*Mixing Rules for foreground data of post processing printer *)*
1	Opaque
2	Transparent
3	Mixing rule not defined (device specific)

*)Background data always mixes transparent. Foreground and background for BCOCA data are defined by the BCOCA architecture.

Figure 2-19: Details of Barcode PDF Ids, Mixing Rules

xx (hex)	Supported Bar Code
01	Code 39 (3-of-9 Code), AIM USS-39
02	MSI (modified Plessey code)
03	UPC/CGPC Version A
05	UPC/CGPC Version E
06	UPC Two-digit Supplemental
07	UPC Five-digit Supplemental
08	EAN 8 (includes JAN-short)
09	EAN 13 (includes JAN-standard)
0A	Industrial 2-of-5
0B	Matrix 2-of-5
0C	Interleaved 2-of-5, AIM USS-I 2/5
0D	Codabar, 2-of-7, AIM USS-Codabar
11	Code128, AIM USS-128
16	EAN Two-digit Supplemental
17	EAN Five-digit Supplemental
18	POSTNET
1A	RM4SCC
1B	Japan Postal Bar Code
1C	Data Matrix (2D bar code)
1D	MaxiCode (2D bar code)
1E	PDF417 (2D bar code)
1F	Australia Post Bar Code
20	QR Code
21	Code 93

Figure 2-20: Details of Barcode PDF IDs

This Table is described also in IBM's Specification Bar Code Object Content Architecture Reference S544-3766 6th edition or later (p. 30 in 6th edition).

Stack Mode Triplet **(optional)**

With this triplet, a stacker reports the way it stacks the sheets. Some devices stack sheets face down or collated, other face up or unccollated.

Bytes	*Description*		*Value (hex):*
0..1	Length of the triplet, including the parameter itself		04
2	Stack Mode triplet ID		1B
3	Stack Mode		
	Face Up	Sheet Sequence 2,1 4,3 6,5 ..	01
	Face Down	Sheet Sequence 1,2 3,4 5,6 ..	02
	reserved		00, 03 .. FF

Maximum and Minimum Sheet Size Triplet **(optional)**

With this triplet, a device may reports the maximum and minimum size of a sheets it is able to handle.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	13
2	Maximum Sheet Size triplet ID	1C
3..6	Maximum Paper Width in millipoints (=1/72000 inch)	....
7..10	Maximum Paper Length in millipoints (=1/72000 inch)	....
11..14	Minimum Paper Width in millipoints (=1/72000 inch)	....
15..18	Minimum Paper Length in millipoints (=1/72000 inch)	....

Stop Position Triplet **(optional)**

With this triplet, a device may the its stop position of a sheet. This is necessary, if sheets with tabs should be processed.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	13
2	Stop Position triplet ID	1D
3	Stop Position	
	Top of Form	01
	Left Edge of Form	02
	Bottom Edge of Form	03
	Right Edge of Form	04
	reserved	00, 05--FF

2.3.4 Device State Request Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x000F 0004
Information Triplet[...]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

Each UP^{3}I device may learn the device state of peripheral devices connected to the real-time interface.

To obtain this information the UP^{3}I device sends a “Device State Request“ broadcast frame. All attached UP^{3}I devices reply to this broadcast message with a “Device State“ frame.

This Frame is sent:

- after a successful Power On Sequence
- whenever a device needs the current process line information

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the broadcasting UP³I device)	..
4	not used for broadcasts	00

2.3.5 Device State Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x000F 0005
Information Triplets[...]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

This frame is sent:
- With each device state change (e.g. Ready, Not Ready)
- If the device achieves or looses synchronization
- As a reply to the “Device State Request“ Frame
- after a paper jam / system error (as this leads to a “ Device Not Ready” state)
- after receiving a "Set Device State" Frame

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the broadcasting UP³I device)	..
4	not used for broadcasts	00

Device State Triplet **(mandatory)**

Bytes	*Description*		*Value (hex):*
0..1	Length of the Triplet, including the parameter itself		0004
2	Device State Triplet ID		02
3	Device State		
	Device Ready	The device is ready for operation	02
	Device Not Ready	The device is Not Ready for operation. The printer stops (if the device with Not Ready state is in the paper path involved) paper movement and is only allowed to restart if this device has sent a “Device Ready“ Frame again.	03
	Not Ready pending	The device wants to enter Not Ready state, but is still active till current action is over and predecessor device is Not Ready or until the device is no longer part of an actice tupel.	04
	Reserved		00, 01, 05..FF

Device Synchronous Triplet **(mandatory)**

Bytes	*Description*		*Value (hex):*
0..1	Length of the Triplet, including the parameter itself		0004
2	Device Synchronous Triplet ID		03
3	Device Synchronous State:		
		Device Synchronous	01
		Device Asynchronous	02
		Reserved	03..FF

Correlating UP³I_PAGE_ID Triplet **(mandatory / optional)**

This triplet is only attached, if the frame has a direct correlation to a specific UP³I_PAGE_ID.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	0008
2	Correlating UP³I_PAGE_ID Triplet ID	04
3-6	UP³I_PAGE_ID	....
7	UP³I Paper Sequence ID of UP³I source device	..

State Description Triplet **(optional)**

There are several reasons (which may be valid at the same time) for a specific device state. The UP³I device has to describe this reason:

Bytes	*Description*	*Value (hex):*
0..1	Length of this triplet, including this field	0006
2	State Description triplet ID	05
3	State Description: Paper Jam (page has to be reprinted)	00

	Device transaction due to Operating Panel (Key pressed) Bin Access Request	10 11
	Logical Error on page / synchronization lost	22
	Paper end Toner end Stacker / Rewinder full Staple end Waste Box full Collator Overflow, stack to thick to staple Resource end	31 32 33 34 35 36 3F
	Paper unexpected (e.g. invalid Barcode ID) ~~Post-processing Handling Time can't be guaranteed~~ (reserved for compatibility purpose)	40 41
	Preparation / Service Cycle active For additional information the "Preparation triplet" may be added	50
	Repeat Synchronize Form The specific device missed the sync form (e.g. a manual set-up is needed) and wants another chance. This request can usually be triggered by manual intervention (operator).	51
	Device Stopped The pre- / post processing device may send this frame to stop the production line only for a short period of time. This situation is known as „ADEV Stop". This state is also used as an acknowledge, when a device reached this state.	80
	Device Ready Complementary signal to Device Stopped. The printer is allowed to start moving the paper again	81
	Reserved	All other
4 - 5	Device specific Error ID. This ID is used for Error Log and sense bytes.	
	No ID available	0000
	Print data stream errors	
	UP³I finishing operation not supported	0001
	Invalid finishing operation reference corner/edge	0002
	Print Data Position Check (Post Processing Printers)	0003
	Print Data Format ID (PDFID) not supported	0004
	Data error	0005
	reserved	0006..00FF
	Paper Jam	
	paper jam	0100
	infeed error	0101
	outfeed error	0102
	paper transport error	0103

	reserved	0104..01FF
	Reserved	
	reserved	0200..1FFF
	UP³I Protocol error	
	reserved	2000..2FFF
	Out of Supply error	
	Infeed empty	3000
	reserved	3001..3FFF
	Unexpected event	
	reserved	4000
	paper out of sequence(opt. Coded Sequence ID invalid/wrong)	4001
	sheet missing	4002
	sheet to early	4003
	emergency stop	4004
	communication error	4005
	power supply failure	4006
	sensor failure	4007
	actuator failure	4008
	reserved	4009..40FF
	Stop Conditions	
	Waste-box full	4100
	Outfeed full	4101
	Cover / Guard open	4102
	reserved	4103..41FF
	Reserved	
	reserved	4200..4FFF
	Timing error	
	reserved	5000
	~~post processing handling time cannot be guaranteed~~ (reserved for compatibility purpose)	5001
	reserved	5002..50FF
	Position Checks	5100..51FF
	Reserved	5100
	Position Check detected by Postprocessing Printer	5101
	Reserved	5102..51FF
	Reserved	
	reserved	5200..FFFF

String for Operating Panel Triplet **(optional)**

ATTENTION: With this string no language separation is possible. This triplet should only be used for debugging or to fulfill elder operating panel requests. For a UP³I operator panel use the “official” GUI path.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	04 – 00FF
2	ASCII string for Operating Panel triplet ID	06
3+n	ASCII string (max. 252 Characters)	.....

Warning Field Triplet: **(optional)**

0..1	Length of the triplet, including the parameter itself	0004

2	Warning triplet ID (send string with String for Operation Panel Triplet)	07
3	Warning Type: Paper soon empty Toner soon empty Stacker / Rewinder soon full Staple soon empty Waste Box soon full Resource soon empty reserved	 01 02 03 04 05 0F All other

Error Category Triplet: **(optional)**

In case of an device error (e.g. paper jam...) the UP³I device needs to describe the recovery category. The new device state is “Device Not Ready”.

0..1	Length of the triplet, including the parameter itself	0004
2	Error category triplet ID	08
3	Error category: Operator Intervention required: Continue seamless with “received page+1” after operator intervention Internal restart: Paper Path not affected (= no paper jam). Continue seamless with “committed page+1” Paper-Jam: Page recovery: continue with “stacked page+1” Set recovery: continue with “begin of affected set” System Error: Reboot necessary, operator needs to check the output; continue like paper jam category Syntax Check: A Post Processing Printer detected a Syntax Error in the Print Data	 01 02 03 04 05
	Position Check: A Post Processing Printer detected a Position Check in the Print Data	06
	Reserved	All other

Preparation Triplet: **(optional)**

In case of a preparation (service cycle) a UP³I device announces this situation. Other attached UP³I devices which also need to go to service cycle within a short time may use this information to coordinate their upcoming service cycle with this running one.

0..1	Length of the triplet, including the parameter itself	0008
2	Preparation triplet ID	09
3..6	Expected Preparation time [seconds] If the preparation time cannot be predicted the preparation time needs to be set to zero.	....
7	Preparation category: Operator Intervention required: Internal Preparation: Reserved	 01 02 All other

2.4 Paper Motion Information Frames

2.4.1 Running Up/Down, Blower / Cycle Up / Be Prepared Frames

Frame Information Flow:

a) Starting paper movement:
Immediately before starting the paper movement (printing) the “Running Indication“ frame is initially sent by the printer device to the adjacent UP³I devices. The adjacent devices forward the “Running Indication“ frame to their neighbors (if available). As soon as a UP³I device is prepared and ready, it acknowledges the “Running Indication” frame using the “Running Acknowledge” frame. The printer device starts the paper movement immediately when the “Running Acknowledge” frames are received.

E.G.: A stacking UP³I device may acknowledge the Running Indication Frame from the printer although the next UP³I device in the paper path direction (e.g. a book binder) has not acknowledged the running indication. The stacker assumes (with reference to the cycle up time property of the binder), that the binder will send the “Running Acknowledge” frame before the stacked Set is sent to the binding device. If the “Running Acknowledge” frame is not received in time, the stacker may simply stop the printer again (ADEV Stop).

b) Stopping paper movement:
When the printer is stopping paper movement for a period expected to be more than 20 seconds it sends the “Running Indication“ frame is to its neighboring UP³I devices. The neighbor devices forward the “Running Indication“ frame to their neighbors (if available). The “Running Indication” frame is immediately answered using the “Running Acknowledge” frame.

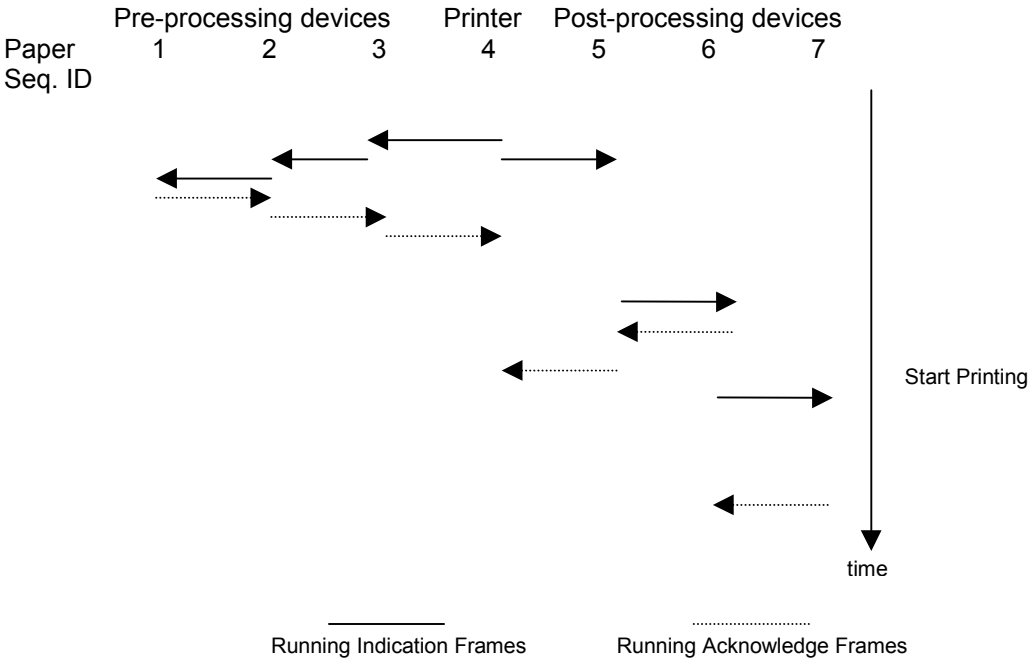


Figure 2-21: Running Frame Information flow

2.4.1.1 Running Indication Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0001 0001
Information Triplets[...]

transmitted last

Asynchronous packet

Cycle Up Frame / Be Prepared Frame

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the sending UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device)	..

Device Running Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004
2	Running State Triplet ID	02
3	Running State	
	Blower On / Run Indication, Running Up: A new „Running Up“ Frame is only sent if in the meantime a „Running Down“ Frame was sent.	01
	Blower Off / Stop Indication, Running Down: This asynchronous Frame is sent if the paper movement stop is expected to be more than 20 sec.	02
	Reserved	03..FF

Note: For additional status information, devices may send the Device State frame (Preparation Triplet).

Selected Tupel Triplet **(optional)**

With this optional triplet, it is possible to send a Running Indication for only one tupel/paper path of a complete UP³I line. The selected tupel is called the ‘active tupel’.
If this triplet is omitted, all tupels are active and all connected devices must prepare for running.
With Selected Tupel Triplet, this frame is a dedicated Running Indication.

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004 .. n
2	Selected Tupel Triplet ID	03
3 - n	Paper Sequence Tupel, list of selected Paper Sequence IDs	.. ..

See Chapter 2.8.3, Starting Production Line with dedicated Running Indication Frame (positive Acknowledge) and Chapter 2.8.4, Starting Production Line with dedicated Running Indication Frame (negative Acknowledge).

2.4.1.2 Running Acknowledge Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0001 0002
Information Triplets[...]

transmitted last

Asynchronous packet

Cycle Up Acknowledge Frame / Be Prepared Acknowledge Frame

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the sending UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device)	..

Device Running Acknowledge Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004
2	Running State Triplet ID	02
3	Running State	
	Blower Now On / Run Indication, Running Up Done:	01
	Blower Now Off / Stop Indication, Running Down Done:	02
	Reserved	03..FF

Selected Tupel Triplet **(optional)**

If the Running Indication Frame contained a Running Tupel Triplet, also the Running Acknowledge Frame needs a Running Tupel Triplet to assign the Running Acknowledge to the according tupel.

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004 .. n
2	Selected Tupel Triplet ID	03
3 - n	Paper Sequence Tupel, list of selected Paper Sequence IDs	.. ..

2.4.2 PAC Frames

2.4.2.1 Propose Frame (former 'Form Exit without Form')

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0001 0011
Information Triplets[...]

transmitted last

Asynchronous dedicated packet

Frame Information Flow:

- This Frame is part of the PAC cycle and valid only for Cut Sheet Devices.
- This Frame is sent as soon as possible to inform the print line about future sheets. It is identical to the relating Form Exit Frame with an additional Expected Time Triplet.
- The frame is daisy chained from the printer to the downstream devices.

The propose frame requests work to be performed by a device such as feeding a sheet, perforating a sheet, or stapling a stack. The device must determine if it can perform the work at the reference time specified.
Note that a proposal does not tell the device to execute the capability. Only if the device subsequently receives a Confirm frame for the same page id should the device plan on executing the capability.

Information Triplets:

The end_of_compilation Boolean specifies the end of a compilation. At this point the device performs the capability specified in the associated propose_compilation.

The response to Propose is Accept or Reject frame. A proposal does not tell the device to execute the capability. Only if the device subsequently receives a Confirm for the same page ID should the device plan on executing the capability.
It is possible to send several information entries with this form exit frame. Every form includes at least one page. If there is more than one logical page included (e.g. n-up) the device describes the logical pages as well. In this case each logical printed page receives a separate UP^{3}I_PAGE_ID (by the printer). Also if both sides of the paper are printed only one form is announced with this frame.

All Triplets of the Form Exit Frame are used. The Reference Time (Sub-) Triplet () as a part of the Page/Set/Job Triplet Triplet is the only additional one.

Reference Time (Sub-) Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the End of Job ID, inclusive this field	000B
2	End of Job Triplet ID	19
3..10	IEEE 1394 timestamp, when will this page performed	....

2.4.2.2 Accept Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0001 0012
Information Triplets[...]

transmitted last

Asynchronous dedicated packet

Frame Information Flow:

- This Frame is part of the PAC cycle and valid only for Cut Sheet Devices.
- This asynchronous Frame is the acknowledge to the Propose Frame.
- It is daisy chained the opposite direction as the Propose Frame.

The device sends this frame to indicate that it accepts the proposer's proposal of the same page id.

Information Triplets:

The page_id identifies the commitment being accepted. If this is an invalid page id the proposer's behavior is undefined.

An acceptance does not mean the device will execute the commitment. Only if the device subsequently receives a Confirm for the same page ID should the device plan on executing the commitment.

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the broadcasting UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

UP³I_PAGE_ID Triplet **(mandatory)**

At least one UP³I_PAGE_ID Triplet is necessary.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0008
2	Page Description Triplet ID	02
3	UP³I Paper Sequence ID of UP³I source device (need not to be the printer device but e.g. an interposer device)	..
4-7	UP³I_PAGE_ID	....

2.4.2.3 Reject Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0001 0013
Information Triplets[...]

transmitted last

Asynchronous dedicated packet

Frame Information Flow:

- This Frame is part of the PAC cycle and valid only for Cut Sheet Devices.
- This asynchronous Frame is the negative acknowledge to the Propose Frame.
- It is daisy chained the opposite direction as the Propose Frame.

.

Information Triplets:

The page_id identifies the commitment being rejected. If this is an invalid page id the proposer's behavior is undefined.

The reject_reason indicates the reason for the rejection. The possible reason are:

If the reject reason is timing, the proposal was rejected because of a timing problem. The retry_reference_time is an absolute time that informs the proposer that the device won't be able to accept a reference time prior to this value.

If the reject reason is capability_state, the proposal was rejected because the capability is not ready. The state is the current state of the proposed capability. Note that the proposer should know the capability state as it is observing all state changes. However, sending this information in the reject eliminates potentially proposer thrashing due to race conditions.

If the reject reason is wrong_ordered_stock_index, the proposal was rejected because the proposer proposed to feed an ordered stock index that does not match the actual index that would be fed on the next proposal. The next_ordered_stock_index indicates which index the feed point can feed next. This reject reason is relevant only to feed points loaded with ordered stock.

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the broadcasting UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device)	..

UP^{3}I_PAGE_ID Reject Triplet **(mandatory)**

If the page described in this triplet cannot be finished without any restrictions, this triplet is used to describe the restriction. These restrictions are not any errors or missing resources. These are reported with the Device State Frame.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0010
2	Page ID Reject Triplet ID	02
3	UP³I Paper Sequence ID of UP³I source device (need not to be the printer device but e.g. an interposer device)	..
4-7	UP³I_PAGE_ID	....
8..11	Reject Reason Need more time (an extra gap) Invalid/Unsupported Finishing Operation Reserved	 0001 0002 All other
12..15	Extra gap in milliseconds (if Reject Reason == 0001)	....

2.4.2.4 Confirm Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0001 0014
Information Triplets[...]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

- This Frame is part of the PAC cycle and valid only for Cut Sheet Devices.
- This is the last step in positive PAC cycle. After propose and accept, this is the acknowledgement for all relating devices, that the page(s) described in the UP³I_PAGE_ID triplet will be processed as proposed and confirmed.
- The printer device sends this frame to the post-processing device(s) to confirm the announced page(s).
- This frame is given from one device to next one until the last device in the addressed tupel got it.
- There is no acknowledge necessary for this frame.

The Confirm frame confirms that a capability should be executed as proposed in the in proposal. The proposer sends this frame if all other device involved in the commitment group accepted the proposal.

Information Triplets:

The page_id identifies the commitment group being confirmed. Upon receiving this message the device is scheduled to perform the work in the referenced commitment. If the device finds it cannot do the work to which it committed after receiving this call, it must call broken.

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (not used for broadcasts)	0

UP³I_PAGE_ID Triplet **(mandatory)**

At least one UP³I_PAGE_ID triplets is mandatory.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0007
2	Page Description Triplet ID	02
3-6	UP³I_PAGE_ID	....

2.4.2.5 Cancel Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0001 0015
Information Triplets[...]

transmitted last

Asynchronous dedicated packet

Frame Information Flow:

- This Frame is part of the PAC cycle and valid only for Cut Sheet Devices.
- This is the last step in negative PAC cycle. After propose and reject, this is the acknowledgement for all relating devices, that the page(s) described in the UP³I_PAGE_ID triplet will not be processed as proposed and confirmed.
- The printer device sends this frame to the post-processing device(s) to cancel the announced page(s).
- This frame is given from one device to next one until the last device in the addressed tupel got it.
- There is no acknowledge necessary for this frame.

The cancel frame indicates that the capability should not be executed as proposed in the proposal of the specified page id. The proposer cancels a proposal if another device rejected the proposal.
.

Information Triplets:

The page_id identifies the commitment group being confirmed. Upon receiving this message the device is scheduled to perform the work in the referenced commitment. If the device finds it cannot do the work to which it committed after receiving this call, the device must break the commitment

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

UP³I_PAGE_ID Triplet **(mandatory)**

At least one UP³I_PAGE_ID triplet is mandatory.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0007
2	Page Description Triplet ID	02
3-6	UP³I_PAGE_ID	....

2.4.3 Form Exit Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0001 0003
Information Triplets[...]

transmitted last

Asynchronous dedicated packet

Frame Information Flow:

- Each device sends this „Form Exit" Frame to the next device in the direction of paper movement sequence (the last paper processing device just acknowledges the pages).
- The frame is sent immediately when a sheet of paper *starts to be processed* by a UP³I device. In the case of a printer this is at the start of printing the page.
- The frame is generated by a printer or interposer device (which is noticed by the page source triplet).
- If a parameter belonging to the form changed after sending the form exit frame another full form exit frame may be sent. For correlation to the specific UP³I Form, Set or Job the new frame needs to include the identical UP³I Page- Set or Job-ID. The current parameter(s) overwrite the previously sent. If the finishing device receives the changed form exit frame too late (e.g. the operation has already started) the device changes to NOT READY state and reports an error.

Information Triplets:

It is possible to send several information entries with this form exit frame. Every form includes at least one page. If there is more than one logical page included (e.g. n-up) the device describes the logical pages as well. In this case each logical printed page receives a separate UP³I_PAGE_ID (by the printer). Also if both sides of the paper are printed only one form is announced with this frame.

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

Form Size Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	000E
2	Form Size Triplet ID	02
3-6	Paper Length in millipoints (=1/72000 inch).	....
7-10	Paper Width in millipoints (=1/72000 inch).	....
11-12	Paper Weight in g/m²	.. ..
13	interposer form this form will be interposed later this form is already in the paper stream reserved	 01 02 00, 03 - FF

Explanation of the 'interposer form' byte:
This byte is necessary to inform any PPDs of forms, that are interposed into the paper stream later on. A “form exit” frame with interposer form == ' this form will be interposed later ', describes a form, that does not exist yet. A non interposer PPD receiving this frame hands this frame on to the next device in the current tupel. Any “Begin of Set” triplets in this frame are valid from this sheet until the relating “End of Set”. Any “End of Set” Triplet in this frame closes the relating “Begin of Set”.
This “Form Exit” frame is acknowledged by a “Form Acknowledge” frame.
The interposer will interpose the required form and set the 'interposer form' byte to 'this form is already in the paper stream' in the relating “form exit” frame.

The Page ID given by the printer for this form or sheet stays valid over the whole lifetime. There is no need for an interposer device to give a page ID to an interposed form.

Form Finishing Operating Triplet **(mandatory)**

Remark: If there are more finishing procedures, the triplet is repeated as often as finishing operations have to be done.
The Finishing Information is sent with each form. If the operation needs a prior collection of forms (e.g. stitching) the finishing operation has to be fulfilled with the “End of Set” Form.
WARNING: if the Default values for the Finishing Operation Parameter is used the result of operation may be device (vendor) dependent! In general use the dedicated finishing operation (not the Default).

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including this field	000D – 00NN
2	Form Finishing Operating Triplet ID	03
3	UP³I Paper Sequence ID of the destination device (that has to fulfill this finishing operation).	..
4	Finishing Operation Type	
	No Operation / Pass through paper	00
	Paper Input / Page interpose (not used for AFP/IPDS)	01
	Fold	03
	Staple / Stitch	04
	Cut	05
	Trim	06
	Offset / Group Separator / Job Separator	07
	Stack	08
	Rotate	09
	Punch	0A
	Bind	0B
	Merge	0C
	Banding	0D
	Shrink Wrap	0E
	Rewind	0F
	Postprocessing Print	10
	Make a Booklet	11
	Table	12
	Special Handling	F0
	Reserved	All other
5..6	Finishing Operation Parameter	
	Interpose	
	Interpose Form from bin xx / Stock Number	0001..00FE

Bytes	Description	Value (hex):
	Default Bin / Stock	FFFF
	Fold	
	See **Figure 2-23: Fold Catalog part 1** and 2 for folding parameters	.. ..
	E.g.: Fold Parameters like “F16-13” ⇨ 0x100D “F12-4” ⇨ 0x0C04	
	No Fold	0000
	Device Default	FFFF
	Staple / Stitch	
	Corner Staple	0001
	Saddle Stitch In	0002
	Saddle Stitch Out	0003
	Edge Stitch	0004
	Default	FFFF
	Cut	
	Separation cut	0001
	Perforation cut	0002
	Cross cut	0003
	Signature cut (used only between Host and Printer). The printer will collect set with this finishing parameter to one sheet and generate one single Form Exit	0004
	Default	FFFF
	Trim	
	Front Edge	0001
	1 Edge	0002
	3 Edge	0003
	5 Edge	0004
	Default	FFFF
	Offset / Group Separator / Job Separator	
	Offset to left	0001
	Offset to right	0002
	Default	FFFF
	Stack	
	Alternate Offset Stack	0001
	Deliver Stack	0002
	Accumulate Paper	0003
	Default	FFFF
	Rotate	
	See **Figure 2-30: Examples for Rotation Operation and Figure 2-31: Examples for Turn Operation**	
	90 degree clockwise	0001
	180 degree clockwise	0002
	270 degree clockwise	0003
	Turn 180 degree x direction	0100
	Turn 180 degree y direction	0200
	Default	FFFF
	Punch	
	Round Hole	0001
	Rectangular hole	0002
	Default	FFFF
	Bind	
	Default	FFFF
	Merge	

Bytes	*Description*	*Value (hex):*
	Handle most left page first	0001
	Handle most right page first	0002
	Default	FFFF
	Banding	
	Single Band Wrap	0001
	Double band wrap	0002
	Crossing band wrap	0003
	Default	FFFF
	Shrink Wrap	
	Shrink Wrap	0001
	Default	FFFF
	Table	
	Default	FFFF
	Winding	
	Rewind	0001
	Default	FFFF
	Print	
	Print, Data see Print Data Frame (if this finishing operation is selected, the following data are ignored. Position and orientation are given in the print data frame)	0001
	Booklet maker	
	Booklet	0001
	Book	0002
	Default	FFFF
	Not listed Operation / Special Handling	
	Specific Parameter (not defined)	0000.FFFE
	Not applicable	FFFF
7	Finishing Operation reference corner / edge This and the following parameters are ignored by postprocessing printers.	
	Bottom-right corner / bottom edge	00
	Top-right corner / right edge	01
	Top-left corner / top edge	02
	Bottom-left corner / left edge	03
	The following 4 entries refer to the UP³I coordinate system (see chapter 2.2.2, UP³I Definition of UP³I "Form", "Page", "Medium Origin"):	
	Bottom-right corner / bottom edge	10
	Top-right corner / right edge	11
	Top-left corner / top edge	12
	Bottom-left corner / left edge	13
	Device default reference corner / edge	FF
	Reserved	04..0F, 14..FE
8	Finishing Operation count	
	Not specified / use device default	00
	Number of operations to apply	01..FF
9..12	Finishing Operation axis offset (relative to reference edge)	
	Device default axis offset	00
	Offset in millipoints (=1/72000 inch).	01..FFFFFFFF

Bytes	*Description*	*Value (hex):*
Zero or more finishing operation positions in the following format (the number of finishing operation positions must match the count value, byte 8):[1]		
13..16	Operation position on finishing operation axis (this parameter specifies a position on the axis when a finishing operation can be positioned along that axis; offset 0 is either at the bottom edge or the left edge of the sheet)	
	Device default axis offset	00
	Offset in millipoints (=1/72000 inch).	01..FFFFFFFF

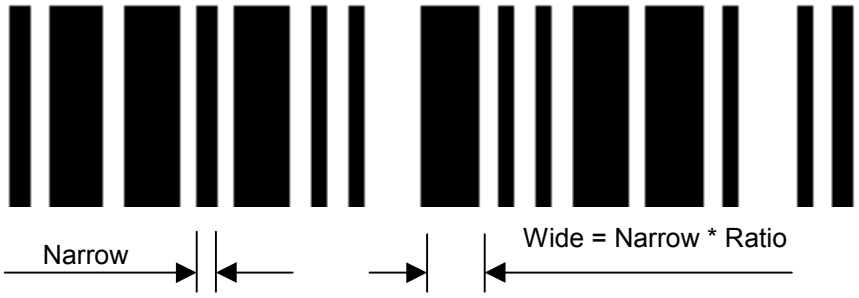
Find some examples of finishing operations at the end of this chapter!

[1] for further information see IBM specification 'IPDS Finishing Operations Including UP3I Enhancements'

UP3I Mark Triplet **(optional)**

The UP³I Mark is described with this triplet if it doesn't belong to a logical page (e.g. it will be cut away during the finishing process). If the UP³I Mark is printed into the physical range of a logical page the description of the mark is done with the UP3I Mark SUB Triplet.

Optional Coded Sequence: The "Narrow" dimension is the width of the narrowest bar or space in the bar code, and all bars and spaces are a multiple of this number. The wide to narrow ratio is "Ratio". The following picture shows a 3:1 wide to narrow ratio.



Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, incl. this field	00xx
2	UP³I Mark Triplet ID	04
3	Mark Position (referring to physical form side) Upper Side Under Side	 01 02
4..7	Mandatory Start Mark Length[2] (in millipoints =1/72000 inch)	....
8..11	Mandatory Start Mark Width (in millipoints)	....
12..15	Mandatory Start Mark Shift in paper motion direction from medium origin (in millipoints)	....
16..19	Mandatory Start Mark Shift across paper motion direction from medium origin (in millipoints)	....
20	Optional Coded Sequence Printed Yes No	 01 02
21-24	Optional Coded Sequence Mark Narrow Bar dimension (in millipoints =1/72000 inch)	....
25	Optional Coded Sequence Mark wide to narrow ratio (X:1)	..
26-29	Optional Coded Sequence Mark Shift in paper motion direction from medium origin (in millipoints)	....
30-33	Optional Coded Sequence Mark Shift across paper motion direction from medium origin (in millipoints)	....

[2] Length in paper motion direction

Registration Mark Triplet **(optional)**

The UP³I Registration Mark is described with this triplet if it doesn't belong to a logical page (e.g. it will be cut away during the finishing process). If the UP³I registration Mark is printed into the physical range of a logical page the description of the mark is done with the UP3I Registration Mark SUB Triplet. This triplet is repeated as often as registration marks are printed on the page. Find more information in chapter 2.2.5.

Bytes	*Description*		*Value (hex):*
0..1	Length of the Triplet, including this field		000C
2	Registration Mark triplet ID		05
3..6	Distance in paper transport direction from Medium Origin to Registration Mark Center (millipoints)		....
7..10	Distance rectangular to paper transport direction from Medium Origin to Registration Mark Center (millipoints)		....
11	Registration Mark Type:	CrossCutMark	01
		TopVerticalCutMark	02
		BottomVerticalCutMark	03
		LeftHorizontalCutMark	04
		RightHorizontalCutMark	05
		LowerLeftCutMark	06
		UpperLeftCutMark	07
		LowerRightCutMark	08
		UpperRightCutMark	09

Paper Speed Triplet **(optional)**

Paper Speed of the regarding Form. Used for Cut Sheet devices only, web devices use the Paper Movement Frame, which is mandatory for web devices.

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including this field	0007
2	Paper Speed Triplet ID	08
3..6	Paper Speed of this Form [millipoints/millisecond]	....

Page/Set/Job Triplet Triplet **(mandatory)**

This triplet (including its sub triplets) is repeated as often as logical pages are printed on the form.

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including this field	0nnn
2	Page triplet ID – With n-Up n page triplets are available	06

It is mandatory to send at least one of the following "optional" marked triplets

- UP³I_PAGE_ID (Sub-) Triplet
- UP³I_SET_ID (Sub-) Triplet
- UP³I_JOB_ID (Sub-) Triplet

For more details see the examples in chapter 2.2.3, UP3I Differentiation of Form, Set and Job.

UP³I_PAGE_ID (Sub-) Triplet (optional)

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0007
2	UP³I_PAGE_ID Triplet ID	01
3..6	UP³I_PAGE_ID The 32 bit width ID is set by the printer or interposer device. If there is no page ID from the host available (e.g. at the interposer device), the UP³I_PAGE_ID is an incrementing number starting with number „0x00000001“. The page origin (printer or interposer) is minuted by the page source triplet.	....

UP³I_SET_ID (Sub-) Triplet (optional)

A complete Set of forms is delivered. The SET_ID is sent with the corresponding mandatory Start- and End of Set Triplet within this frame.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0003
2	UP³I_SET_ID Triplet ID	02

UP³I_JOB_ID (Sub-) Triplet (optional)

A complete Job is delivered. The JOB_ID is sent with the corresponding mandatory Start- and End of Job Triplet within this frame.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0003
2	UP³I_JOB_ID Triplet ID	03

Page Size (Sub-) Triplet (mandatory)

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0013
2	Page Size Triplet ID	11
3..6	Horizontal distance from form origin point (millipoints)	....
7..10	Vertical distance from form origin point (millipoints)	....
11..14	Page Length (millipoints)	....
15..18	Page Width (millipoints)	....

Page Sequence (Sub-) Triplet (mandatory)

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	00xx
2	UP³I Page Sequence ID Triplet ID	12
3	UP³I Paper Sequence ID (of the sending UP³I device-needs not to be the printer but e.g. the interposer device)	..
4	UP³I Paper Sequence ID (of the final destination UP³I device)	..

5	Waste Paper Indication • FALSE (send paper to normal exit) • TRUE (send paper to waste bin)	 00 01
6 - n	Paper Sequence Tupel this entry is optional – but the entry has to be sent if a new Tupel (paper path) is selected.	.. ..

UP³I Mark (Sub-) Triplet **(optional)**

The UP³I Mark is described with this triplet if it belongs to a logical page (e.g. it will not be cut away during the finishing process). If the UP³I Mark is printed into the physical range of a logical page the description of the mark is done with the UP3I Mark SUB Triplet. The coded sequence contains (if printed) the last four digits of the corresponding UP³I_PAGE_ID. This triplet has to be available with the first printed page. If there is no change in the parameters the following pages need not have this triplet.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, incl. this field	00xx
2	UP³I Mark Triplet ID	13
3 –xx	*Definition equal to "UP³I Mark" Triplet (starting with byte 3)*	....

Registration Mark (Sub-) Triplet **(optional)**

The UP³I Registration Mark is described with this triplet if it belongs to a logical page (e.g. it will not be cut away during the finishing process).

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including this field	000C
2	Registration Mark triplet ID	14
3. – xx	*Definition equal to "Registration Mark" Triplet" (starting with byte 3)*	....

Start of Job (Sub-) Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Start of Job ID, inclusive this field	000B
2	Start of Job Triplet ID	15
3..10	Job ID The first page of a job is printed. It is usually used to offset different jobs in a job-queue. The post-processor may control the job-separation with this frame. A Job ID is transferred for safety and / or administration purposes. The post-processor may control the job-separation with this frame. If a Job ID is not available the field is set to zero.	....

Start of Set (Sub-) Triplet **(optional)**

This triplet is sent with the 1st page of a set. The referring finishing operation is valid for all sheets in this set.

Bytes	*Description*	*Value (hex):*
0..1	Length of the Start of Set ID, inclusive this field	00xx
2	Start of Set Triplet ID	16
3..10	Set ID	....
11..12	Height / Thickness of Set in Sheets unknown height number of sheets	 0 1..FFFF
13..xx	Page Finishing Operation (that belongs directly to this Set ID) *Definition equal to „Form Finishing Operation“ Triplet (starting with byte 3)*	.....

End of Set (Sub-) Triplet **(optional)**

This triplet is sent with the last page of a set. The referring finishing operation is achieved with this page.

Bytes	*Description*	*Value (hex):*
0..1	Length of the End of Set ID, inclusive this field	000B
2	End of Set Triplet ID	17
3..10	Set ID - The last page of a set	....

End of Job (Sub-) Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the End of Job ID, inclusive this field	000B
2	End of Job Triplet ID	18
3..10	Job ID - The last page of a job	....

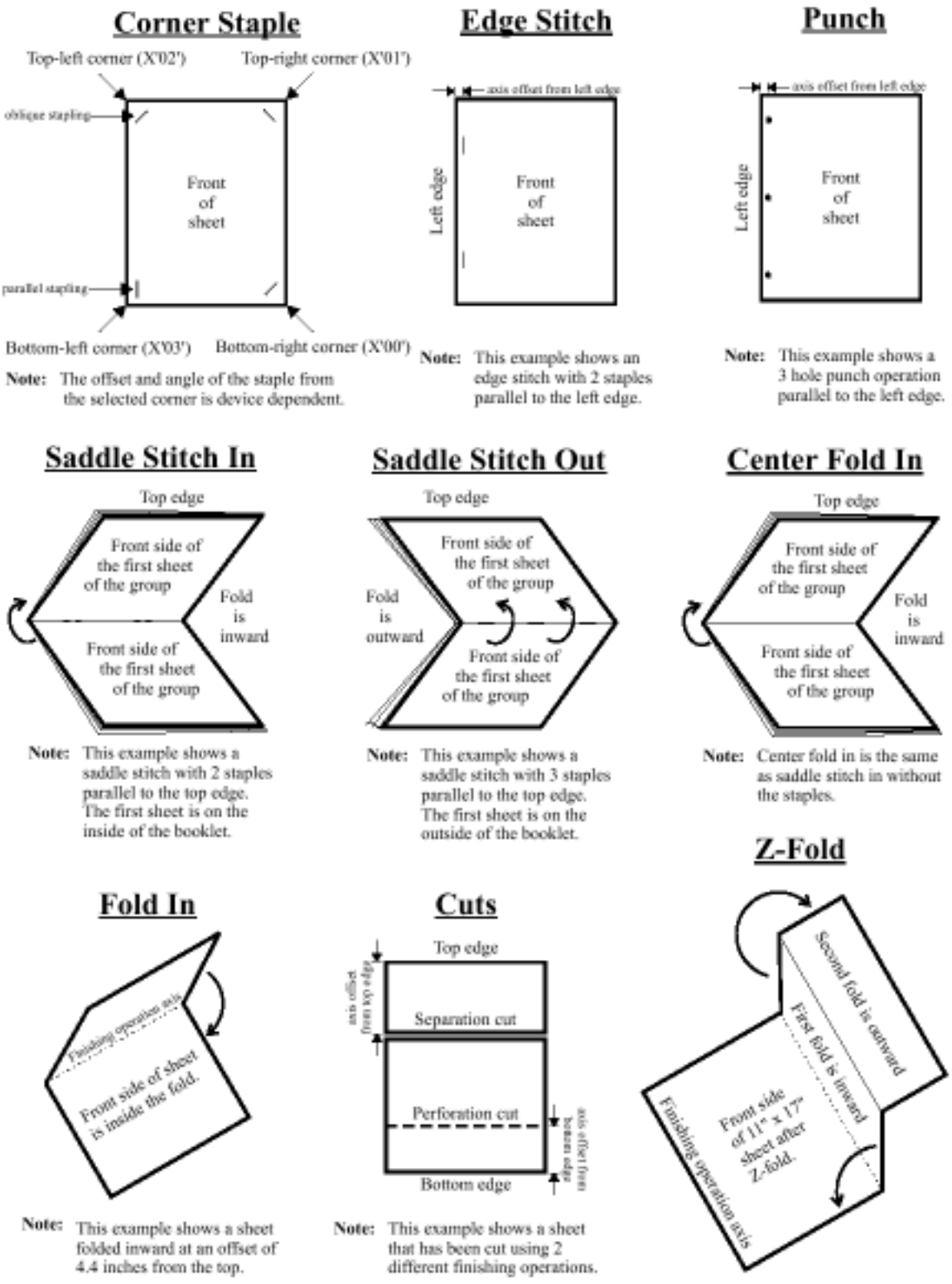


Figure 2-22: Example of finishing operations

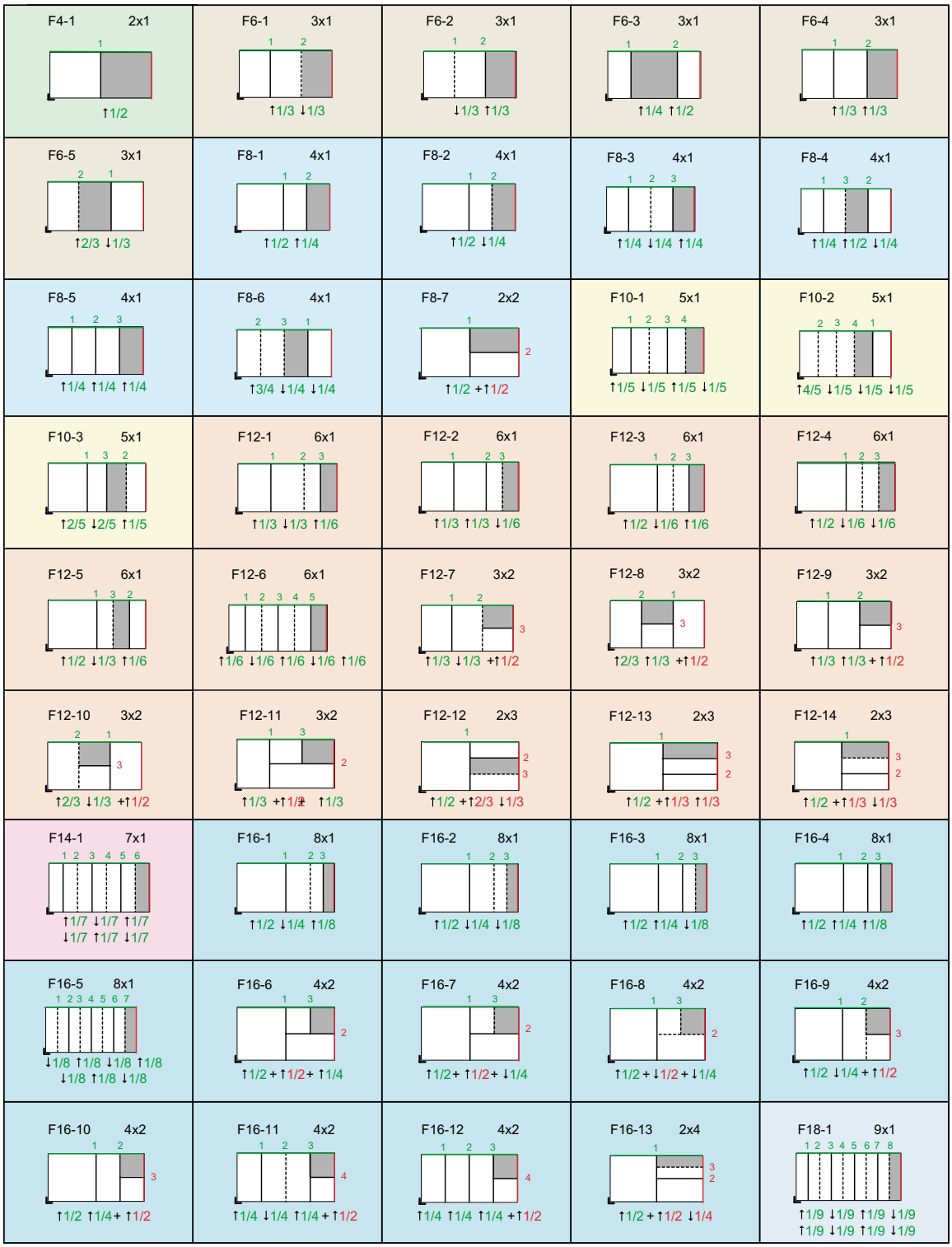


Figure 2-23: Fold Catalog part 1

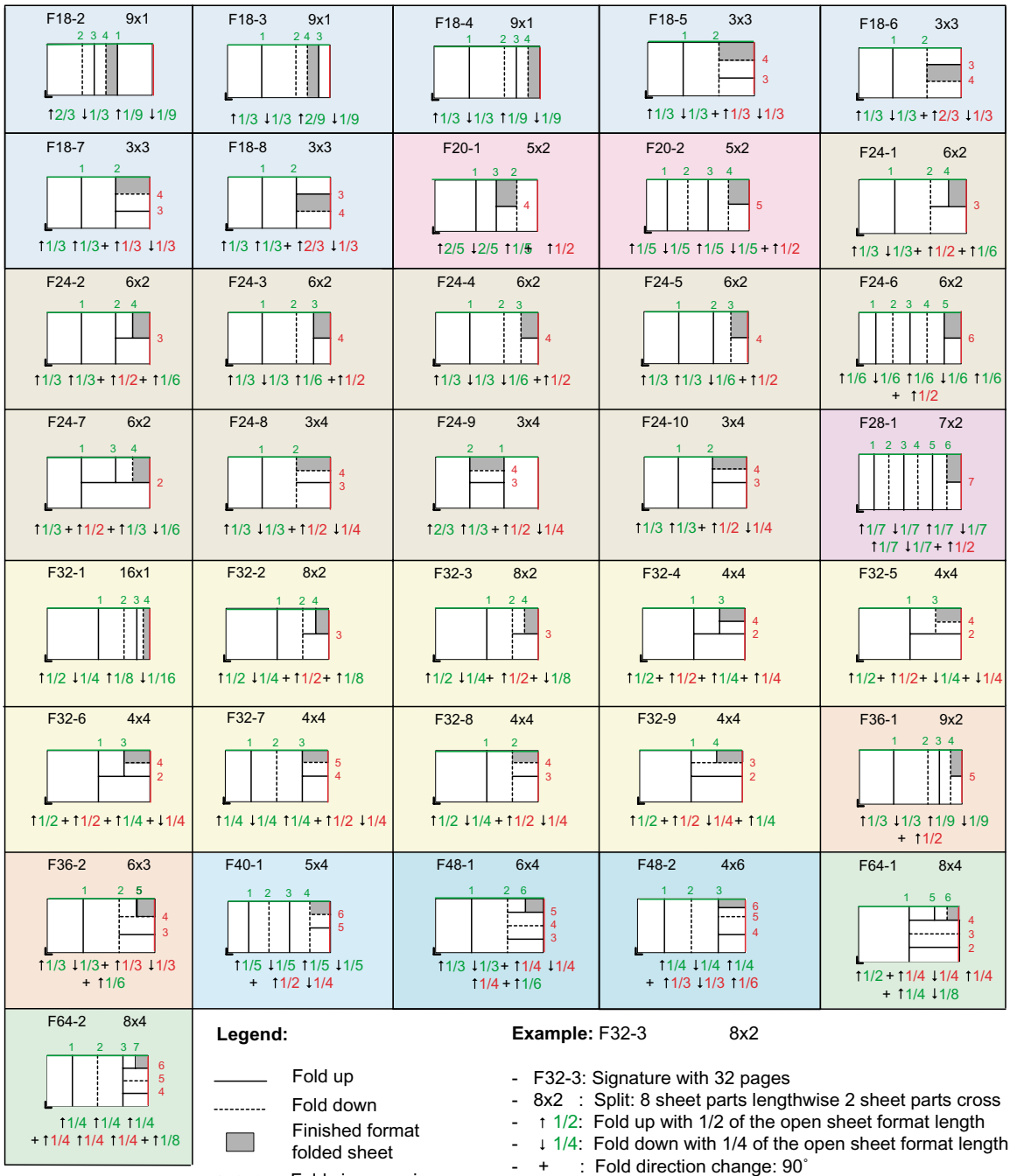


Figure 2-24: Fold Catalog part 2

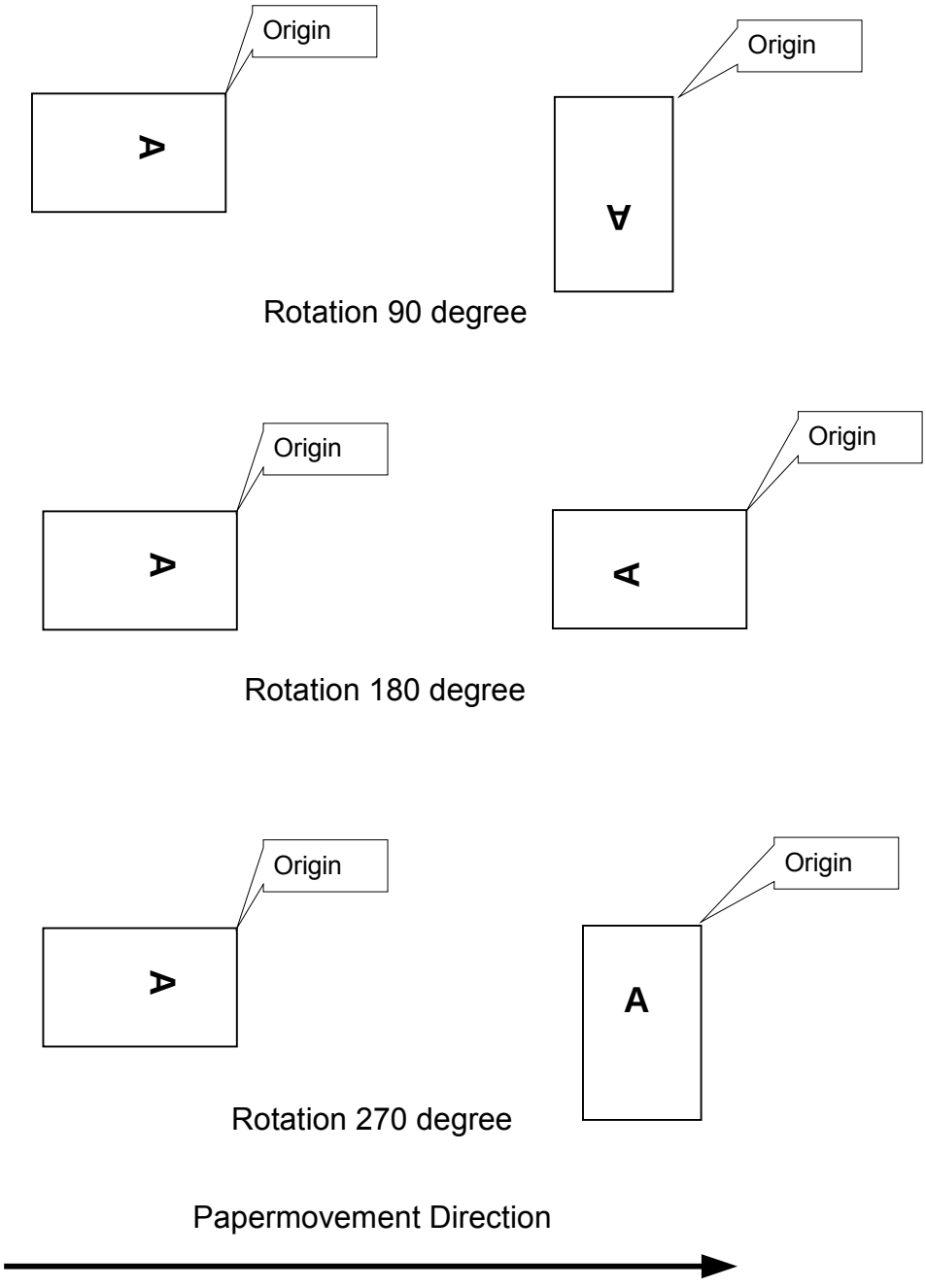
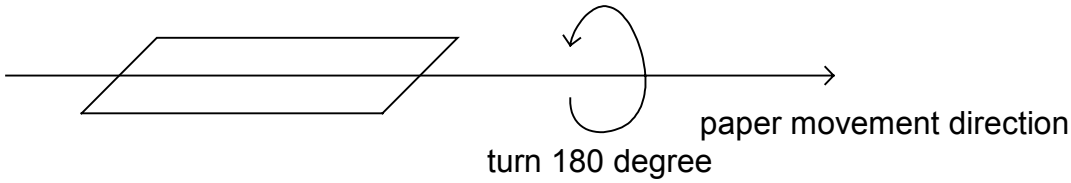
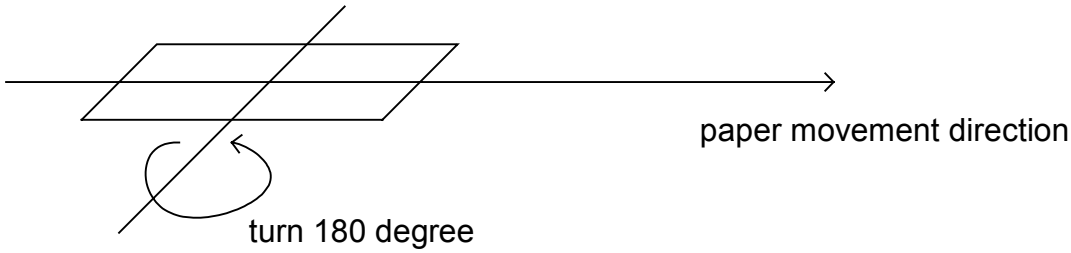


Figure 2-25: Examples for Rotation Operation

The media origin relates to the paper movement direction, regardless of the image or subsequent rotation/inversion processes.



turn 180 y



turn 180 x

Figure 2-26: Examples for Turn Operation

2.4.4 Form Invalidation Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0001 000A
Information Triplets[...]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

- Each device may broadcast this “Form Invalidation“ Frame.
- The frame is sent due to a fault condition to invalidate already handled pages.
- At least one Invalidation triplet has to be provided.

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the broadcasting UP³I device)	..
4	not used for broadcasts	00

Page Source Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0004
2	Page Source Triplet ID	02
3	UP³I Paper Sequence ID of UP³I source device (need not to be the printer device but e.g. an interposer device)	..

It is mandatory to send at least one of the following “optional” marked Invalidations triplets

- Page Invalidation Triplet
- Set Invalidation Triplet
- Job Invalidation Triplet
- Clear Device Triplet

Page Invalidation Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0007
2	Page Invalidation Triplet ID	03
3..6	UP³I_PAGE_ID (announced by form exit frame)	....

Set Invalidation Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	000B
2	Set Invalidation Triplet ID	04
3..10	Set ID (announced by form exit frame)	....

Job Invalidation Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	000B
2	Job Invalidation Triplet ID	05
3..10	Job ID (announced by form exit frame)	....

Clear Device Triplet **(optional)**

Each device may send this “Clear Device“ Triplet.
The frame is sent due to a fault condition to clear the paper path as far as possible without operator intervention. This action results in a “Device Asynchronous State”.

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0003
2	Clear Device Triplet ID	06

2.4.5 Print Data Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0001 0004
Information Triplets[..]

transmitted last

Asynchronous packet

Frame Information Flow:

- The frame is sent immediately when a sheet of paper leaves the UP³I printer..
- The printer device sends this frame to the destination (post-processing) device (the destination device does not forward it further).

Information Triplets:

The optional asynchronous “Print Data“ frame includes print information for post processing devices. This frame is typically used for additional printers (Troy, ACS) which need a small amount of print data.

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

The following three triplets (UP3I_PAGE_ID, Print Location Triplet and Print Data Triplet) belong together and are treated as a unit. That means also, the sequence of these triplets needs to be in this order.
If there is more than one Print Data Triplet necessary, send a new Print Data Frame or a new set of UP3I_PAGE_ID, Print Location Triplet and Print Data Triplet within the same frame.

UP3I_PAGE_ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0007
2	UP³I_PAGE_ID Triplet ID	02
3-6	UP³I_PAGE_ID	....

Print Location Triplet **(optional)**

This optional triplet describes position and orientation of the following print data triplet. If this triplet is omitted, the default values will be used.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0015
2	Print Location Triplet ID	03
3..6	Horizontal distance of the print data window from origin point in millipoints (default: 0)	....
7..10	Vertical distance of the print data window from origin point in millipoints (default: 0)	....
11..14	Horizontal size of the print data area in millipoints (default: page size) Use default:	 FFFFFFFF
15..18	Vertical size of the print data area in millipoints (default: page size) Use default:	 FFFFFFFF
19	Orientation 0 degree (default) 90 degree clockwise 180 degree clockwise 270 degree clockwise Reserved	 01 02 03 04 All others
20	Side Front side (default) Reverse side reserved	 01 02 All others

Print Data Triplet **(mandatory)**

This Triplet contains the description of the data to print. The Print Data Format ID defines what Sub Triplets are included.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	Nnnn
2	Print Data ID Triplet ID	04
3 – 6	Print Data Format ID As defined in the Postprocessing Printer Print Data Format ID Sub Triplet in the SDF	....
7-n	Print Data Sub Triplets, as defined below	....

Print Data BCOCA BSD Sub Triplet (mandatory)

This Triplet is valid and mandatory only for BCOCA PDF-Ids. It transports the data of the BCOCA BSD and must appear exactly one time in the Print Data Triplet.
The BSD data structure is described in IBM's Specification Bar Code Object Content Architecture Reference S544-3766 6th edition or later (p. 26 in 6th edition).

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	nnnn
2	BCOCA BSD Sub Triplet ID	01
3..n	Data of the BSD	

Print Data BCOCA BSA Sub Triplet (mandatory)

This Triplet is valid and mandatory only for BCOCA PDF-Ids. It transports the data of the BCOCA BSA and must appear al least one time in the Print Data Triplet.
The BSA data structure is described in IBM's Specification Bar Code Object Content Architecture Reference S544-3766 6th edition or later (p. 63 in 6th edition).

Bytes	*Description*	*Value (hex):*
0..1	Length of the sub triplet, including the parameter itself	nnnn
2	BCOCA BSA Sub Triplet ID	02
3..n	Data of the BSA	

2.4.6 Free Page Information Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0001 0005
Information Triplets[..]

transmitted last

Asynchronous packet

Frame Information Flow:

- There is no fixed correlation for sending this frame.
- The device send this frame only to the destination device (the destination device does not forward it further).

Information Triplets:

The frame is used for „private“ communication between single UP^{3}I devices.

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the sending UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device)	..

Free Page Information **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Free Page triplet, inclusive this field	00nn
2	Free page Information Triplet ID	02
3..n	Free defined page synchronization / information (e.g. printed marks or bar code etc.). To guarantee more safety it’s possible to print (barcode...) and send page information simultaneously. The post processing device can control both information and stop on error. The need of a previous synchronization of the production line may be done e.g. with a special header page (-> t*o be defined*). Also the „Available Processing Options“ may be used.	....

Private Service Information **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, inclusive this field	00nn
2	Private Service Information Triplet ID	03
3..n	Private (vendor specific) service information If possible: Use the “Error Log” (chapter 2.6.8).	....

2.4.7 Paper Movement Frame

Frame Format:

transmitted first

UP3I Command ID (32 bit) = 0x0001 0006
Information Triplets[..]

transmitted last

Isochronous data block / Asynchronous packet

Frame Information Flow:

- Every announced time interval (-> Self Defining Field, Paper Movement Distance Triplet) the printer sends this "Paper Movement" frame.
- The frame is not sent during manual or automatic paper feeding.
- The frame is used to synchronize the speed of pre- and post processing devices.
- If the announced time interval is more than 100 ms the Isochronous mode is not necessary and the frame may be sent with usual Asynchronous mode.
- In Isochronous mode frames may be lost without notification. Problems resulting of lost Paper Movement Frames can be avoided by accumulating all moved paper of the Paper Distance Triplet in a new variable. This value is set to zero every time the Moved Paper distance since the previous paper movement is zero.

This Frame is also necessary for Type I, Type II, backward compatibility. It is defined and necessary for continuous form devices only.

The need for this (relative) precise timing reference fulfils IEEE1394 with an **Isochronous Transfer Mode**. So every 125 µs there is a possibility to send isochronous data. A "1/6 inch pulse" is transferred by using this mode. This isochronous data is declared as a "broadcast" packet and so the pre- and post processing devices need not acknowledge.

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

Paper Distance Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	000D/0011
2	Paper Distance Triplet ID	02
3..6	Moved Paper distance since the previous paper movement frame in millipoints.	.....
7	Direction of paper movement: Forward paper movement Backward paper movement reserved	 01 02 00, 03 – FF
8..11	Destination Speed [millipoints/milliseconds] Describes the final paper speed after acceleration	.....
12	Direction of Final Speed Forward paper movement Backward paper movement reserved	 01 02 00, 03 – FF
13..16	Accumulated moved paper (used in isochronous packets only)	.....

Paper Movement State Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0004
2	Paper Movement State Triplet ID	03
3	State of Paper Movement Paper started now Paper is moving Paper stopped now reserved	 01 02 03 00, 04 - FF

The time between two Paper Movement Frames depends on the paper speed of the printer.

"With the maximal printer speed the paper distance between two paper movement frames is not larger than 10 cm (<4 inch)."

Speed	Speed	Time Interval [Δt]	Ramping?	Start/Stop
0,5 m/s	30 m/min	200 ms = 10 cm	No problem	Included
1 m/s	60 m/min	100 ms = 10 cm	No problem	Included
1,5 m/s	90 m/min	67 ms = 10 cm	No problem	Included
2 m/s	120 m/min	50 ms = 10 cm	No problem	Included
2,5 m/s	150 m/min	40 ms = 10 cm	No problem	Included

Figure 2-27: Chart for Time Interval between two Paper Movement frames

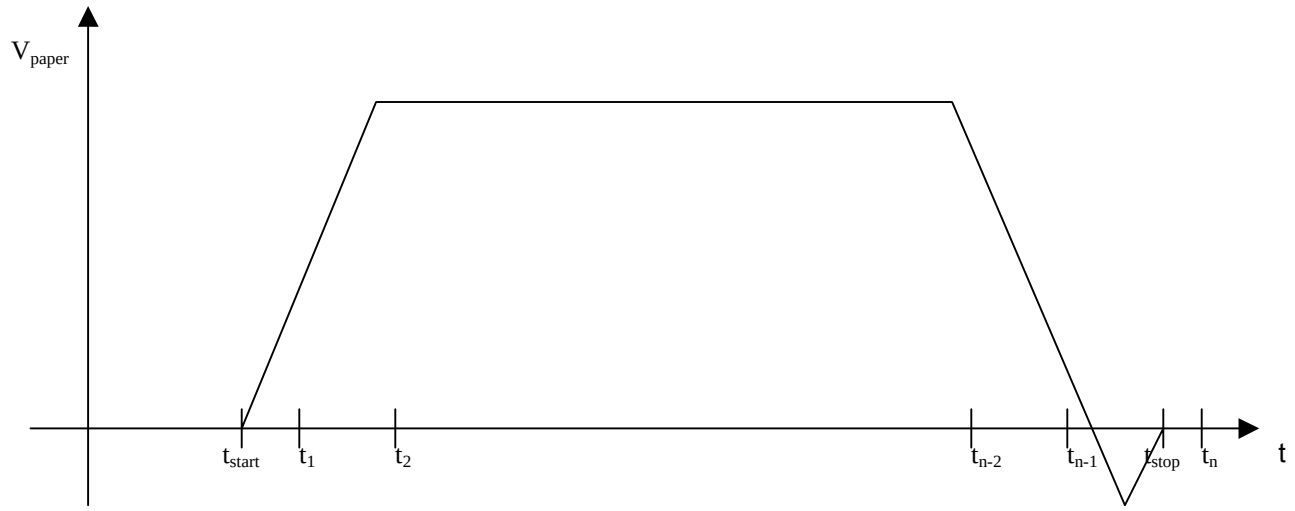


Figure 2-28: Relationship between Paper Movement and Paper Movement Frame

t_{start}:	paper start moving
t_1:	1st Paper Movement frame
t_{stop}:	paper stop moving
$t_1 - t_{start}$	< 4 inch; 4 inch = Δt, see chart, depend on paper speed
$t_2 - t_1$:	Δt, see chart, depend on paper speed
$t_{n-1} - t_{n-2}$:	Δt, see chart, depend on paper speed
$t_n - t_{stop}$:	< 4 inch; 4 inch = Δt

2.4.8 Estimated Paper Movement Frame

Frame Format:

transmitted first

UP3I Command ID (32 bit) = 0x0001 000C
Information Triplets[..]

transmitted last

Asynchronous packet

General:

This optional frame is intended to give a web based device a forecast to near future (timeframe of seconds) paper movement. The given forecast is not obligatorily and can be overwritten and invalidated at any time. The obligatory paper movement indication is only sent by the "Paper Movement Frame". The intention of this "Estimated Paper Movement Frame" is that a downstream device can optimize its finishing task to maintain smoother paper handling instead of start stop operations. For example; a clutching printer will send this frame to its downstream installed web buffer to enable this device to send the paper continuously further downstream (e.g. to a cutter device).

Frame Information Flow:

- An UP³I device sends this Frame to its sequent device as soon as it knows about expected paper movements.
- The frame is not sent during manual or automatic paper feeding.
- The frame is used to support a sequent device to smooth paper movement by anticipating the paper transport speed of adjacent devices.
- Paper jams stop paper movement at once, received Estimated Paper Movement Frames are invalidated by paper jams.
- Up to thirty-two "Estimated Paper Movement" frames may be outstanding (be sent in advance).

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

Estimated Paper Movement Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0021
2	Estimated Paper Movement Triplet ID	02
3	The “Forecast offset time” is relative to ... ... time of sending this frame. ... end time of previously sent “Estimated Paper Movement” frame. Reserved	 01 02 00, 03 - FF
4..7	Forecast offset time: The upcoming paper movement will start in .. milliseconds	.....
16..19	Forecast paper movement time: The upcoming paper movement will last ..... milliseconds	.....
20..23	Forecast paper movement distance: The upcoming paper movement will last ..... millipoints	.....
32	Direction of paper movement Forward paper movement Backward paper movement Reserved	 01 02 00, 03 – FF

Cancel All Estimated Paper Movement Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0003
2	Cancel All Estimated Paper Movement Triplet ID All previously sent “Estimated Paper Movement Frames” are invalidated.	03

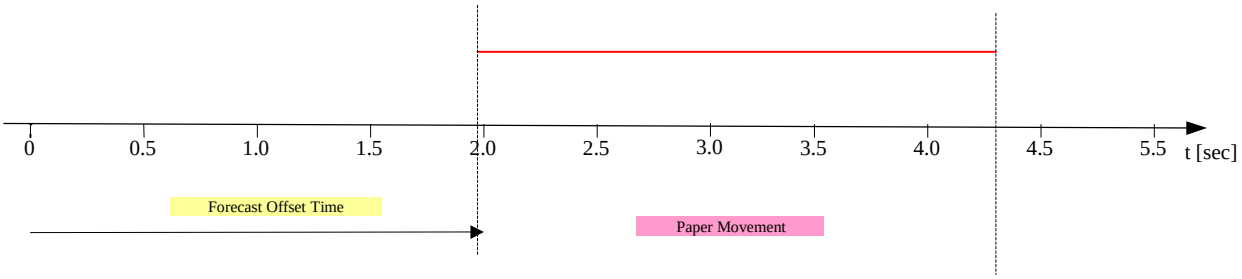


Figure 2-29: Estimated paper movement frame time chart

2.4.9 Paper Request Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0001 0008
Information Triplets[..]

transmitted last

Asynchronous packet

Frame Information Flow:

- The printer device sends this frame to the pre-processing device(s).
- The (pre-processing) destination device(s) neither forwards nor acknowledges this frame.
- This asynchronous Frame is sent when...
  - Cut sheet Printer: ... a sheet of paper is requested by the printer
  - Continuous form Printer: ... a specific length of paper is requested by the printer

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

It is mandatory to send exactly one of the following “optional” marked triplets

- Continuous Form Request Triplet
- Cut sheet Page Request Triplet
- Envelope Media Request Triplet

Continuous Form Request Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0007
2	Continuous Form Request Triplet ID	02
3-6	Paper Length in millipoints (=1/72000 inch).	....

Cut sheet Page Request Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0004
2	Cut sheet Page Request Triplet ID	03
3	Input Media ID	..

Envelope Media Request Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0004
2	Envelope Media Request Triplet ID	04
3	Input Media ID	..

2.4.10 Form Acknowledge Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0001 0009
Information Triplets[..]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

- This asynchronous Frame is broadcast to all attached UP^{3}I devices.
- It is immediately sent when the page was **successfully** handled – In case of an unsuccessful handling the device reports the error / problem with the „Device State“ frame.
- If more pages are handled simultaneously they may be acknowledged within one frame (the triplet has to be repeated accordingly).
- Set, page and job acknowledges may be mixed.

Page Information:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the broadcasting UP³I device)	..
4	not used for broadcasts	00

It is mandatory to send at least one of the following “optional” marked triplets

- UP³I_PAGE_ID Triplet
- UP³I_SET_ID Triplet
- UP³I_JOB_ID Triplet

UP³I_PAGE_ID Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0008
2	Page Description Triplet ID	02
3	UP³I Paper Sequence ID of UP³I source device (need not to be the printer device but e.g. an interposer device)	..
4-7	UP³I_PAGE_ID	....

UP³I_SET_ID Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	000C
2	Set Description Triplet ID	03
3	UP³I Paper Sequence ID of source device	..
4-11	UP³I_Set_ID	....

UP³I_JOB_ID Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	000C
2	Job Description Triplet ID	04
3	UP³I Paper Sequence ID of source device	..
4-11	UP³I_Job ID	....

2.5 Paper Motion Control Frames

2.5.1 EJECT Frame

- The Eject Frame is sent when the “EJECT Button“ is pressed and again when the button is released.
- The EJECT Frame is ignored during “Ready“ state (a specific response is generated).
- The EJECT Frame is ignored if the printer has “Data Ready“ (a specific response is generated).
- During (and after) EJECT the production line is asynchron.
- With EJECT there are no “Form Exit“ Frames sent.
- The typical situation to use the EJECT functionality is during paper insertion.
- Sending this frame causes the device(s) to move the paper through (the printer will not print). The device(s) stop moving paper when the “Button Released“ State is received.

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0003 0001
Information Triplets[..]

transmitted last

Asynchronous packet

Frame Information Flow:

- A device sends this frame to the pre-processing device(s) when the “EJECT button“ is pressed.
- If the receiving device is able to deliver the form the frame is not forwarded further.
- If the receiving device is not able to deliver the form the frame is forwarded to the next device. At least the printer will be able to deliver a (blank) form...
- The frame is acknowledged with the “Eject Reply“ Frame

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (where the key is pressed)	..
4	UP^{3}I Paper Sequence ID (of the destination device = Printer)	..

Button State Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0004
2	Button state Triplet ID	02
3	EJECT Button State Released Pressed Reserved	 01 02 00, 03..FF

2.5.2 EJECT Reply Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0003 0011
Information Triplets[..]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

- The Eject Reply broadcast Frame is sent by the (last) printer as a reply to the Eject frame.
- If the reply is negative (an Eject will not be performed) the “Eject button released“ frame needs not to be sent

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the broadcasting UP³I device)	..
4	not used for broadcasts	00

Eject Reply Triplet **(mandatory)**

Bytes	*Description*		*Value (hex):*
0..1	Length of the triplet, including this field		0004
2	Eject Reply Triplet ID		02
3	Eject Reply:		
	Eject will be performed	Reply to “Key Pressed”	01
	Eject done	Reply to “Key Released”	02
	Eject will not be performed	Device is in Ready State	10
	Eject will not be performed	Printer has Data Ready	11
	Eject will not be performed	Other reason	12
	Reserved		All other

2.5.3 NPRO Frame

- The NPRO (Non Process Run Out) Frame is sent when the “NPRO Button“ is pressed.
- The situation to use the NPRO functionality is to have the job completed in the last processing device – It’s similar to the Host / System NPRO functionality.
- The printer must not have “Data Ready“
- During (and after) NPRO the production line is in synchron state.
- NPRO forms are generating “Form Exit“ Frames.
- For pinless webs the PTL mark has to be printed.

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0003 0002
Information Triplets[..]

transmitted last

Asynchronous packet

Frame Information Flow:

- A device sends this frame to the pre-processing device(s) when the “NPRO button“ is pressed.
- The receiving device delivers the form and forwards the frame to the next device.
- The printer device will print as many (waste) NPRO-Forms as needed...
- The NPRO “Form Exit“ Frames are marked with WASTE.
- The NPRO “Form Exit“ Frames receive a “normal“ UP^{3}I_PAGE_ID
- A running NPRO can not be cancelled (temporarily stopping is possible).
- The post-processing devices have to process the NPRO forms with a default operation.
- This frame is acknowledged with the NPRO Reply frame

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (where the key is pressed)	..
4	UP^{3}I Paper Sequence ID (of the destination device = Printer)	..

.

2.5.4 NPRO Reply Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0003 0012
Information Triplets[..]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

- The NPRO Reply broadcast Frame is sent by the (last) printer as a reply to the NPRO frame.

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the broadcasting UP³I device)	..
4	not used for broadcasts	00

NPRO Reply Triplet **(mandatory)**

Bytes	*Description*		*Value (hex):*
0..1	Length of the triplet, including this field		0004
2	NPRO Reply Triplet ID		02
3	NPRO Reply:		
	NPRO will be performed		01
	NPRO will not be performed –	Device is in Not Ready State	10
	NPRO will not be performed –	Printer has Data Ready	11
	NPRO will not be performed –	Other reason	1F

2.5.5 Test Print Frame

Caution: The name “Test Print” is somehow misleading. Another name for this function could be “Print a specified number of Pages.”

- The Test Print Frame is sent when the „Test Print Button“ is pressed.
- The situation to use the “Test Print“ functionality is to have a single step form processing.
- The sending device awaits the specified amount of pages. If the preceding device cannot fulfill the request, the request is given to its predecessor (at least the printer can do).
- If there are not enough forms available (from the host) the printer waits...
- With “test print” real job forms are printed (no waste is generated). The forms are not reprinted. The output is valid!
- The “Test Print“ can be cancelled with “Not Ready“ State.
- During (and after) Test Print the production line is in synchron state
- Test Print forms generate “Form Exit“ Frames.
- It is not guaranteed in all cases (in particular at n-up printing), that the exact number of forms can be processed.

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0003 0003
Information Triplets[..]

transmitted last

Asynchronous packet

Frame Information Flow:

- The receiving device may deliver a form and forward the frame to the next device.
- The printer device will print as many forms as specified.
- This “Test Print“ frame is acknowledged with the “Test Print Reply“ Frame.

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the sending UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device)	..

Amount of Pages Triplet: **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0004
2	Amount of Pages Triplet ID	02
3	Amount of pages to be printed	..

2.5.6 Test Print Reply Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0003 0013
Information Triplets[..]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

- The Test Print Reply broadcast Frame is sent by the device that fulfills the test print request and is as a reply to the “Test Print“ frame.

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the broadcasting UP³I device)	..
4	not used for broadcasts	00

Test Print Reply Triplet **(mandatory)**

Bytes	*Description*		*Value (hex):*
0..1	Length of the triplet, including this field		0004
2	Test Print Reply Triplet ID		01
3	Test Print Reply:		
	Test Print will be performed as soon as possible		01
	Test Print will not be performed	Asynchron State	11
	Test Print will not be performed	Other reason	1F

2.5.7 Adjust Paper Speed Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0003 0004
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

A pre- or post processing device indicates to its predecessor, that the speed has to be changed.
This may be used if the pre- or post processing device needs to overcome a temporarily critical situation (e.g. the paper length between two devices runs short...) without stopping and starting the process line. If the device is not able to reduce its speed it just forwards the frame to the previous device.

- Each device may send this frame to the previous device against paper movement direction.
- The frame may be sent without a page correlation.
- The device that originally wanted to reduce the paper speed is responsible to reset the speed to "maximum paper speed" again.
- After booting the default (Paper Speed = Maximal) is assumed. Regularly no frame has to be sent.
- This "Adjust Paper Speed" frame is not acknowledged.

If the reduction of the paper speed is not supported the pre- / post processing device has to stop the printer.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

Speed Information Triplet: **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	0004
2	Speed Information ID	02
3	Paper Speed Maximum paper speed Reduced paper speed Reserved	 01 02 03..FF

2.6 Administrative Frames

2.6.1 Get UP³I_PAGE_ID Queue Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0005 0001
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

Use for:

- Gets information of the current situation (e.g. for UP³I Manager)
- Needed for simplifying debugging / not to be used for "normal" work

The addressed device responds to this request with a "UP³I_PAGE_ID Queue" Frame.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (the device that has to respond)	..

2.6.2 UP³I_PAGE_ID Queue Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0005 0011
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

Use for:

- Sends information of current situation to requesting device
- Needed for simplifying debugging / not to be used for „normal“ work

This frame is sent as a response to „Get UP³I_PAGE_ID Queue“ Frame.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

Page ID Queue Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	00nn
2	Page ID Queue Triplet ID	02
2-nn	UP³I_Page_ID‘s of stacked and still not acknowledged forms / pages	.....

If no pages are stacked the triplet consists of 3 bytes (Length field and Triplet ID).

Set ID Queue Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	00nn
2	Set ID Queue Triplet ID	03
2-nn	UP³I_SET_ID‘s of stacked and still not acknowledged sets	.....

If no pages are stacked the triplet consists of 3 bytes (Length field and Triplet ID).

Job ID Queue Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	00nn
2	Job ID Queue Triplet ID	04
2-nn	UP³I_JOB_ID's of stacked and still not acknowledged sets	.....

If no jobs are stacked the triplet consists of 3 bytes (Length field and Triplet ID).

The sequence of the sent pages, sets and jobs is recommended to be in the order of the reception. So some of the optional queue triplets may be repeated.

Page ID Queue not supported Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0003
2	Page ID Queue not supported	05

2.6.3 Desynchronize Production Line Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0005 0002
Information Triplet[..]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

Use for:

- Force desynchronized situation on the production line
- Reset the stacked UP^{3}I_Page_ID information
- Needed for error handling and simplifying debugging

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0005
2	Source device Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the broadcasting device)	..
4	not used for broadcasts	00

2.6.4 Ping Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0005 0003
Information Triplet[..]

transmitted last

Asynchronous/asynchronous broadcast packet

Frame Information Flow:

Use for:

- Needed for error handling and simplifying debugging

The addressed device responds within five seconds to this request with an „Echo“ Frame.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0005
2	Source device Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the sending device)	..
4	Paper Sequence ID of the destination device zero, if sent as a broadcast	..

2.6.5 Echo Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0005 0013
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

Use for:

- Reply to Ping Frame

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0005
2	Source device Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the sending device)	..
4	Paper Sequence ID of the destination device	..

2.6.6 Set Device State Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0005 0004
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

Use for:

- Set a device to a specific state

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

Set Device State Triplet **(mandatory)**

Bytes	*Description*		*Value (hex):*
0..1	Length of the Triplet, including the parameter itself		0004
2	Set Device State Triplet ID		02
3	Device State		
	Set Device Ready	The device is ready for operation	02
	Set Device Not Ready	The device is Not Ready for operation. The printer stops paper movement (if running) and is only allowed to restart if this device has sent a „Device Ready“ Frame again	03
	Reserved		00, 01, 04..FF

2.6.7 Get Error Log Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0005 0005
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

Use for:

- Error tracing
- Obtaining some device specific statistics
- Obtaining information of current situation (e.g. for UP³I Manager)
- Needed for simplifying debugging / not to be used for „normal“ work

The addressed device responses this request with an „Error Log“ Frame.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the sending UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

2.6.8 Error Log Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0005 0015
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

Use for:

- Sent as a reply to „Get Error Log“ Frame

Recommendation:

For easier reading it is recommended to log parameters in hex format. E.g. ids, enumeration,...
Except to this millipoint values should be logged decimal.
New line has to be done with 0x0D0A.
Each errlog entry uses one line.
It is recommended, that the complete File Frame is not longer than 512 Bytes to avoid a mixture of nested File Frames and Long Frame Packets.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the sending UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device)	..

Error Log Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	nnnn
2	Error Log Triplet ID	02
3 - 6	Triplet Counter, starts with 0, increased by 1	
7	Flags 2^0: 0 to be continued 1 end of error log	
8..nn	Error Log (ASCII string)	.....

2.6.9 Synchronize Error Log Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0005 0025
Information Triplet[..]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

Use for:

- Error Log synchronization of all UP³I devices
- Debugging purpose only

Any UP³I device may send this frame. The reason for sending this frame is a debugging request (V24 input, error occurrence etc.).
All UP³I devices have to write the Paper Sequence ID of the originator and the ASCII string of the Sync Text Triplet into the Error Log.

Proposed trace format: "sync error log: <ASCII string> sender: <paper sequence id>"

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	Trace Synchronization Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the broadcasting UP³I device)	..
4	not used for broadcasts	00

Sync Text Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	00nn
2	Synchronization Triplet ID	02
3..nn - 1	ASCII string	..

2.6.10 Set Power State Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0005 0006
Information Triplet[..]

transmitted last

Asynchronous dedicated or broadcast packet

Frame Information Flow:

Use for:

- Set the production line or a single device to a defined power mode – each addressed device has to respond with the "Power State Reply" frame.
- This frame is sent generally by the UP³I Manager; e.g. if there is no UP³I Manager included also the printer may control the production line.
- The frame can be sent at each device state.
- If a UP³I device doesn't support the requested Power State Level a corresponding answer frame is generated - and the power management request itself is ignored by this device.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0005
2	Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the broadcasting UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device) or not used for broadcasts	..

Set Power State Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004
2	Set Power StateTriplet ID	02
3	Go to power state: Idle (normal operation) Suspend Hibernate Power Off	 01 02 03 04

2.6.11 Get Current Power State Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0005 0026
Information Triplet[..]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

Use for:

- Get the power mode of all devices, each device has to respond with the “Power State Reply” frame.
- This frame is sent generally by the UP³I Manager; e.g. if there is no UP³I Manager included also the printer may control the production line.
- The frame can be sent at each device state.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0005
2	Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the broadcasting UP³I device)	..
4	not used for broadcasts	00

2.6.12 Power State Reply Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0005 0016
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

Use for:

- This frame is sent as a reply to “Set Power State” or “Get Current Power State” frame.
- This frame is sent by each UP³I device:
  a) Set Power State reply: as soon as the power management request is fulfilled
  b) Get Current Power State reply: immediately
  c) With a Power State change

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the sending UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device)	..

One of the next two triplets is mandatory, depending on the reason for sending, Set Power State or Get Current Power State.

Power State Reply Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004
2	Power State Reply Triplet ID	02
3	Power State Reply: State Changed O.K. Power Management Level not supported	 01 02

Current Power State Reply Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004
2	Power State Reply Triplet ID	02
3	Current power state: Idle (normal operation) Suspend Hibernate Power Off	 01 02 03 04
3	Power State Reply: State Changed O.K. Power Management Level not supported	 01 02

2.6.13 Information Management Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0005 0007
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

Use for:

- Communication of information stored in commonly structured databases (Management Information Bases MIBs) between UP³I devices and the UP³I Manager.
- This Frame mirrors the SNMP Protocol to the UP³I communication layer. Using this mechanism each UP³I device is released to implement an IP stack.

The frame is built with an UP³I specific header (Paper Sequence ID triplet) and includes thereafter the well defined SNMP definition (please refer to RFC 1157).

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the sending UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device)	..

It is mandatory to send exactly one of the following “optional” marked SNMP triplets

SNMP Request **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0nnn
2	SNMP Request Triplet ID	02
3..	SNMP packet	..

SNMP Response **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0nnn
2	SNMP Response Triplet ID	03
3..	SNMP packet	..

SNMP Trap **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0nnn
2	SNMP Trap Triplet ID	04
3..	SNMP packet	..

2.6.14 Error Detected Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0005 0008
Information Triplet[..]

transmitted last

Asynchronous broadcast packet

Frame Information Flow:

Each UP3I device sends this frame, only in case of detecting a protocol- or logical error. This frame is not necessary during boot phase to publish a non error situation.

Each UP³I device logs the initiating event in its error log.

Regularly the UP³I Manager will respond with a “Get Error Log” Frame.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the broadcasting UP³I device, which is identical to the UP³I device that detected the error)	..
4	not used for broadcasts	00

UP³I Error Detection Triplet (mandatory)

Bytes	*Description*	*Value (hex):*
0 – 1	Length of the Triplet, including the parameter itself	0005
2	Error Detection Triplet ID	02
3	Error Type: Protocol Error (unknown triplets or packets received) Logical Error (an unexpected sequence of page ids, unknown tupel, unknown paper sequence id etc.) Specification Error UP³I Version does not fit, not supported (different UP³I devices support different UP³I versions) Bus Reset, new device connected, No Error (sent, if the error removed)	 01 02 03 04 05 F0
4	Error Level Warning (display only). Stop, no error recovery needed, but operator needs to be informed. Stop, possible data lost, error recovery needed	 01 02 03

String for Operating Panel Triplet (optional)

ATTENTION: With this string no language separation is possible. This triplet should only be used for debugging or to fulfill elder operating panel requests. For a UP³I operator panel use the “official” GUI path.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	04 – 00NN
2	ASCII string for Operating Panel triplet ID	03
3 - NN - 1	ASCII string (max. 252 Characters)	.....

Error Originator Paper Sequence ID Triplet (optional)

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0004
2	Error Originator Paper Sequence ID Triplet ID	04
3	UP^{3}I Paper Sequence ID of the originator of the error.	..

This triplet is sent only, if the originator of the error is known.

2.6.15 File Request Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0005 0009
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

Use for:

- Needed for file transportation
- Usually the UP³I Manager asks for the UP³I device GUI file with this method

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the requesting UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device)	..

File Name Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0004 .. 00NN
2	File Name Triplet ID	02
3 .. NN – 1	File Name (ASCII string) File name. The type of separators is either slash or backslash.	.....

It is not necessary to name a path, if the filename is clearly indicated for the device.

2.6.16 File Frame

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x0005 0019
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

Use for:

- Needed for file transportation
- Usually the UP³I device answers the UP³I Manager's request for a GUI file with this method
- It is recommended, that the complete File Frame is not longer than 512 Bytes to avoid a mixture of nested File Frames and Long Frame Packets.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the source UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device)	..

Exactly one of the following two triplets is mandatory

File Fragment Number Triplet **(optional)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0007
2	File Fragment Number Triplet ID	02
3..4	Number of this Fragment, first fragment == 0	….
5..6	Number of Fragments in total	….

Large File Fragment Number Triplet **(optional)**

Necessary for files bigger than 30 MB.

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	000B
2	Large File Fragment Number Triplet ID	06
3..6	Number of this Fragment, first fragment == 0	….
7..10	Number of Fragments in total	….

File Data Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	3 - nnnn
2	File Data Triplet ID	03
3..nnnn - 1	File binary data	….

File Type Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0005
2	File Type Triplet ID	04
3	File Type Trace File Log File Binary (Code) File GUI File GUI_LIB File GUI_HELP File GUI RESOURCE File reserved	 01 02 03 04 05 06 07 00, 08 – FF
4	Request Byte this file was requested (e.g. a trace file) this file was not requested (e.g. a code download to an external device) reserved	 01 02 00, 03..FF

File type GUI, GUI_LIB, GUI_HELP, GUI_RESOURCE are stored in the http-area of the UP³I Manager, so they are available from the operator panel.
File type GUI_HELP is necessary, allow the operator panel not to load the complete help files, but to get the necessary help classes from the UP³I Manager.
With GUI_RESOURCE a device may send a not requested file to the UP³I Manager, stored this in the http area of this device.

Received File Name Triplet **(optional)**

Optional, but necessary for File Acknowledge Frame

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	0nnn
2	Received File Name triplet ID	05
3..4	Received File Date: Year: yyyy	….
5	Month: Mm	..
6	Day: Dd	..
7	Hour: Hh	..
8	Minute: Mm	..
9	Second: Sec Format see Device GUI Jar File Name Triplet	..
10	not used	00
11..nnn - 1	File Name (ASCII string, max. 252)	....

2.6.17 File Acknowledge Frame

Frame Format:

transmitted first

UP³I Command ID (32 bit) = 0x0005 0029
Information Triplet[..]

transmitted last

Asynchronous packet

Frame Information Flow:

Used for:

- Acknowledgement after file transfer
- Usually the file receiving UP³I device answers with this frame to the sender of one or more File Frames after
  a) the last received File Frame
  b) any detected error during receiving File Frames

If the sender of File Frame does not receive this File Acknowledge Frame within 5 seconds after the last block, an error may be reported.

Information Triplet:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP³I Paper Sequence ID Triplet ID	01
3	UP³I Paper Sequence ID (of the source UP³I device)	..
4	UP³I Paper Sequence ID (of the destination UP³I device)	..

Received File Name Triplet **(mandatory)**

Describes the name of the of the file to be acknowledged (equal to File Frame).

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including the parameter itself	0nnn
2	Reveived File Name triplet ID	05
3..4	Received File Date: Year: yyyy	….
5	Month: Mm	..
6	Day: Dd	..
7	Hour: Hh	..
8	Minute: Mm	..
9	Second: Sec Format see Device GUI Jar File Name Triplet	..
10	not used	00
11..nnn - 1	File Name (ASCII string, max. 252)	....

File Acknowledge Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	0004
2	File Acknowledge Triplet ID	06
3	Acknowledge Type	
	Acknowledge, File received	01
	Transfer Error, resend File	02
	Transfer Error, do not resend File	03
	Unknown File Type, don't know what to do with it	04
	Unknown File Name, don't know what to do with it	05
	Any other Error	06
	reserved	00, 07 .. FF

2.7 Protocol Support Frames

2.7.1 Long Frame Packet

Frame Format:

transmitted first

UP^{3}I Command ID (32 bit) = 0x000D 0001
Information Triplets[...]

transmitted last

Asynchronous (broadcast) packet

Frame Information Flow:

- With this frame a fragmentation of long (> 512 byte data) frames is possible to avoid the maximum packet size restriction of IEEE 1394.
- The aim is to be independent of the transport medium (no IEEE 1394 registers are used) and to be backward compatible.
- A long (> 512 byte data) frame is sent via several Long Frame Packets (Long Frame Packet cycle).
- The Long Frame Packet is used for asynchronous packets only (broadcasts and dedicated); the maximum length of isochronous frames in UP³I is 16 bytes.

Restriction:

- Each device may start only one Long Frame Packet cycle at a time.
- Each device must not intercept a Long Frame Packet by any other frame.
- A reset on the IEEE 1394 line resets all not complete Long Frame Packet cycles at the receiving device.

Information Triplets:

Paper Sequence ID Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the Triplet, including the parameter itself	0005
2	UP^{3}I Paper Sequence ID Triplet ID	01
3	UP^{3}I Paper Sequence ID (of the sending UP³I device)	..
4	UP^{3}I Paper Sequence ID (of the destination UP³I device), zero for broadcasts.	..

Long Frame Triplet **(mandatory)**

Bytes	*Description*	*Value (hex):*
0..1	Length of the triplet, including this field	9 – max1F7
2	Long Frame Triplet ID	02
3..6	Block Count, starting with 1 (first block), increased by 1.	....
7	Block Descriptor First Block Interior Block Last Block (long frame complete now) Reserved	 01 02 03 00, 04 – FF
8..max 1F7-1	Frame Data of the fragmented Frame.	

Workflow Description for the Sender:
If the sending routine detects a send request for a frame with a data size longer than 512 bytes, it splits the long frame into smaller packets (each one with a maximum of 495 bytes of the original frame), puts these small packets into a Long Frame Packet and sends this to the receiver(s). The order of the Long Frame Packets is given by the original frame.

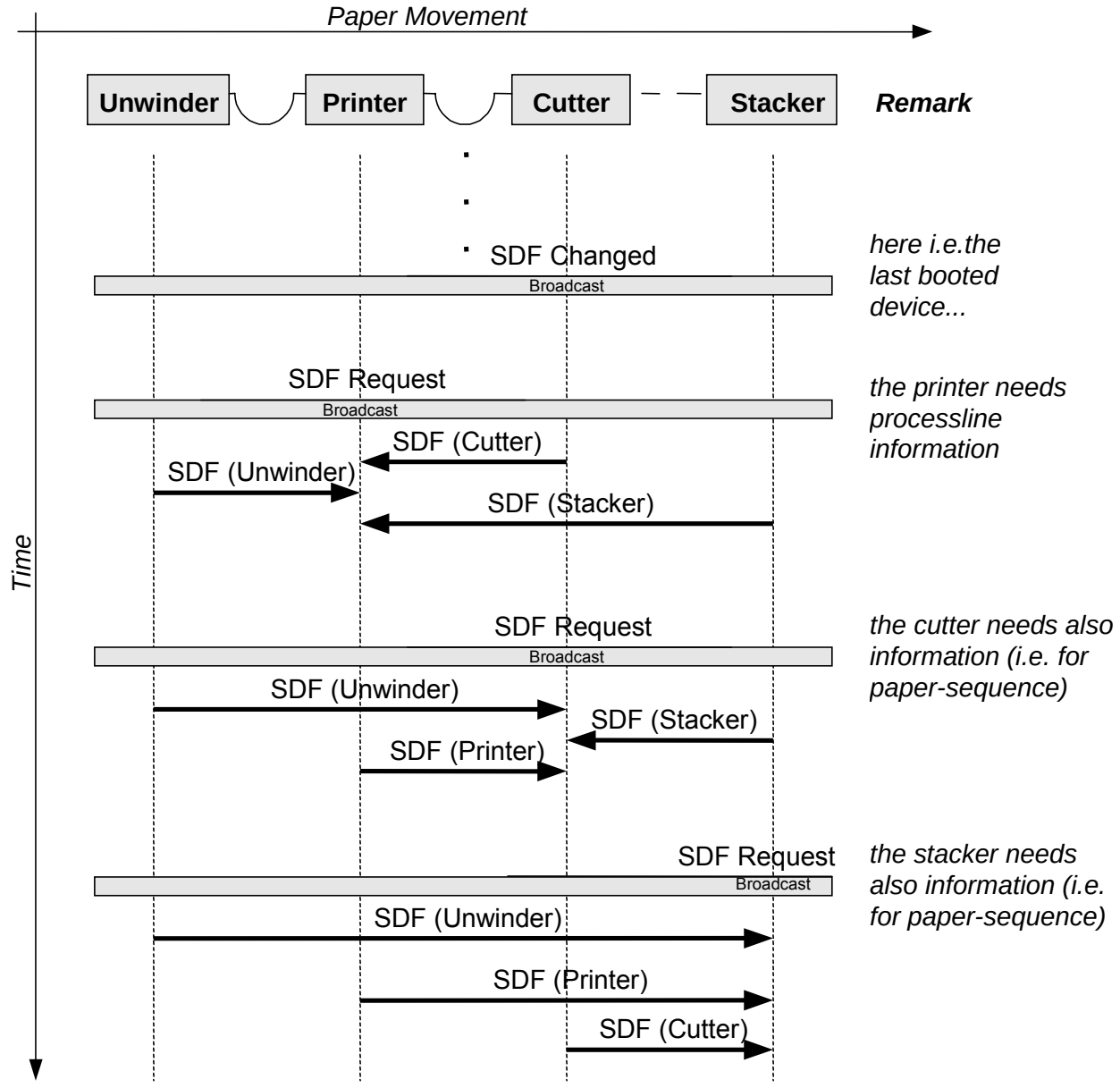
Workflow Description for the Receiver:
If the receiver receives a frame with a command ID unequal to Long Frame Packet, it passes the data to the next higher level. If the command ID is equal to Long Frame Packet, the receiver starts to collect the data. When the frame is complete, the receiver passes the data to the next higher level.

Caution:
It is possible, that a receiver gets Long Frame Packets simultaneously from two or more different senders. It is necessary then to collect the raw frame data in two or more buffers. The maximum number of buffers necessary is equal to the numbers of active devices in the production line.

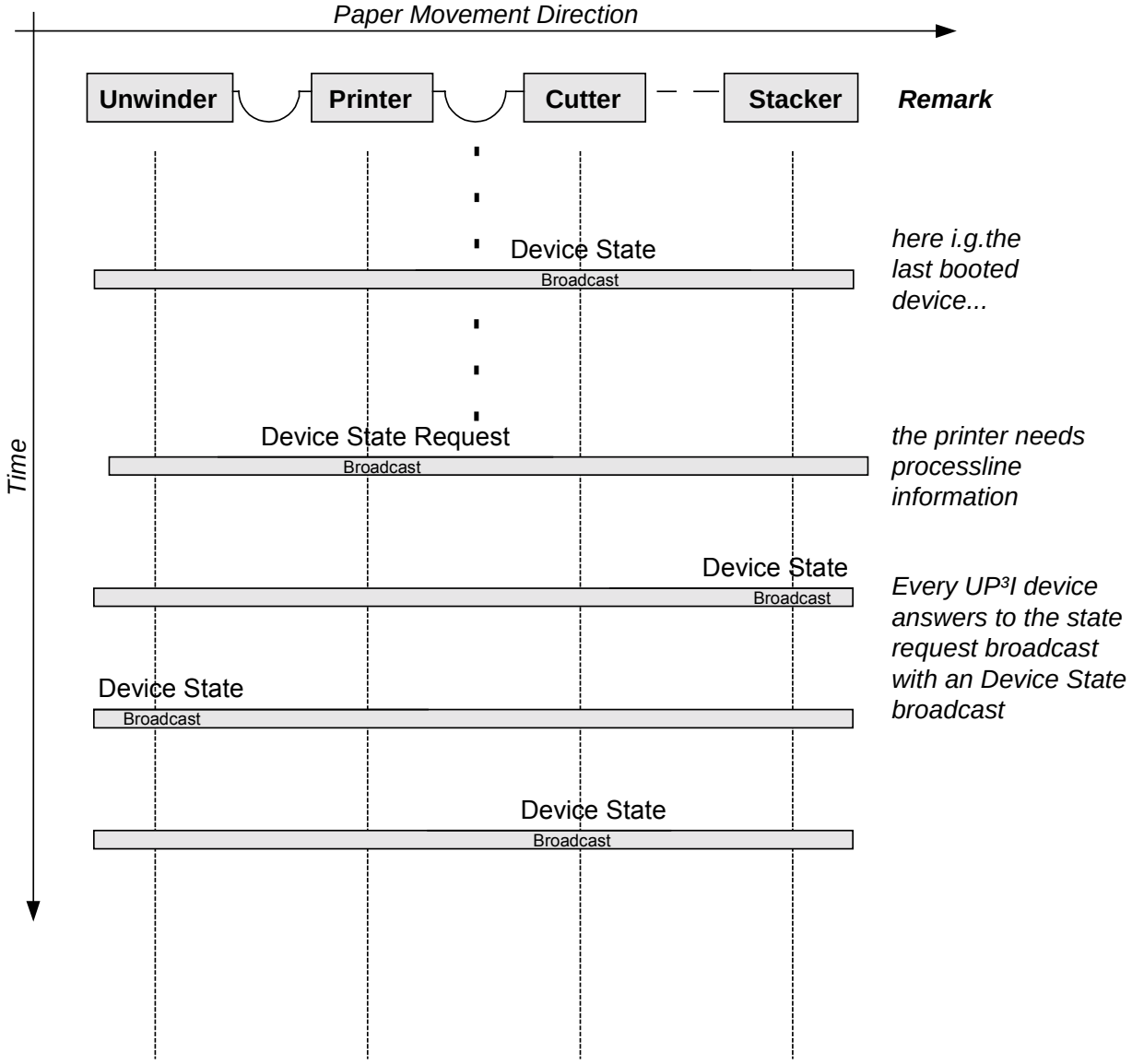
2.8 Frame Communication Examples

Following are some frame communication examples shown.

2.8.1 Initialization from Power On to Off line State



2.8.2 Setting the Production line to On-line and Ready State

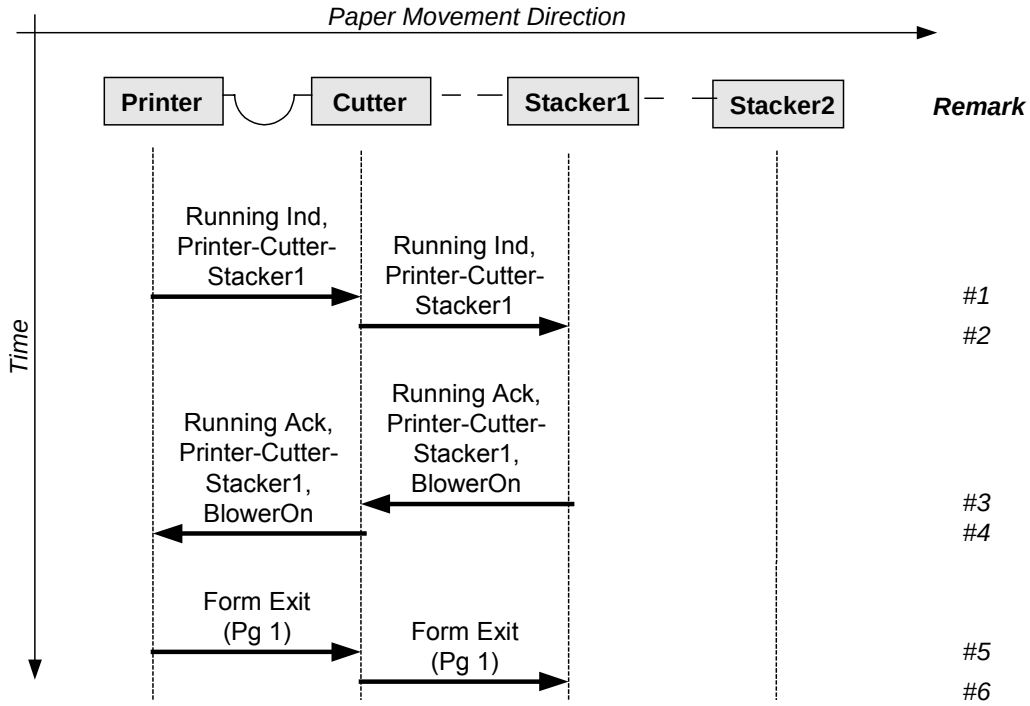


2.8.3 Starting Production Line with dedicated Running Indication Frame (positive Acknowledge)

This example shows, how the dedicated Running Indication Frame works.
Only devices in active tupel need to be ready, devices in inactive tupel may be in any state. Before these devices are needed, the relating tupel is activated by a new dedicated Running Indication Frame. Not used tupel are set to inactive, also with the new Running Indication Frame.

In this example, there are two tupel defined:
Tupel 1: Printer, Cutter, Stacker1 and
Tupel 2: Printer, Cutter, Stacker2.

After the printers boot, there is no active tupel. With the first sheet, the regarding tupel is started with a dedicated Running Indication Frame.
If all devices in this tupel are in Ready State, a Running Acknowledge (Blower On) is given to the printer and the printing is started.



Operating sequence:

1) Printer gets the 1st sheet and sends a dedicated Running Indication Frame(Blower On) for tupel Printer-Cutter-Stacker1 to the cutter.
2) The Cutter sends the dedicated Running Indication Frame to the next device in the addressed tupel.
3) Stacker1 is the last device in the addressed tupel. It starts its motors and answers to the cutter with a dedicated Running Acknowledge Frame (Blower On).
4) The Cutter also starts its motors and passes the Running Acknowledge Frame (Blower On) to the Printer.
5) The Printer now starts to print and sends Form Exits to the Cutter.
6) And so on.

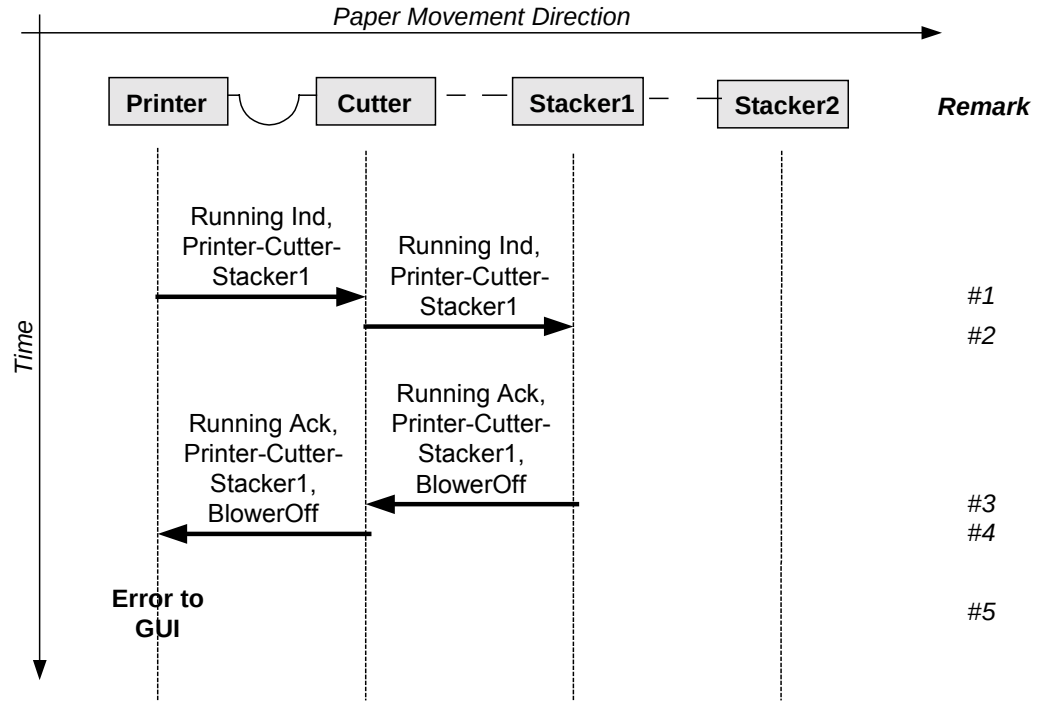
The State of Stacker2 is not relevant for printing this job.

2.8.4 Starting Production Line with dedicated Running Indication Frame (negative Acknowledge)

This example shows, how the dedicated Running Indication Frame works.
Only devices in active tupel need to be ready, devices in inactive tupel may be in any state. Before these devices are needed, the relating tupel is activated by a new dedicated Running Indication Frame. Not used tupel are set to inactive, also with the new Running Indication Frame.

In this example, there are two tupel defined:
Tupel 1: Printer, Cutter, Stacker1 and
Tupel 2: Printer, Cutter, Stacker2.

After the printers boot, there is no active tupel. With the first sheet, the regarding tupel is started with a dedicated Running Indication Frame.
If one of the devices in this tupel is not in ready state, the running indication is answered with a Running Acknowledge Frame (Blower Off, Stop Indication). The printer also interpretes the collected Device State Frames of all devices and posts a error message on its GUI.



Operating sequence:

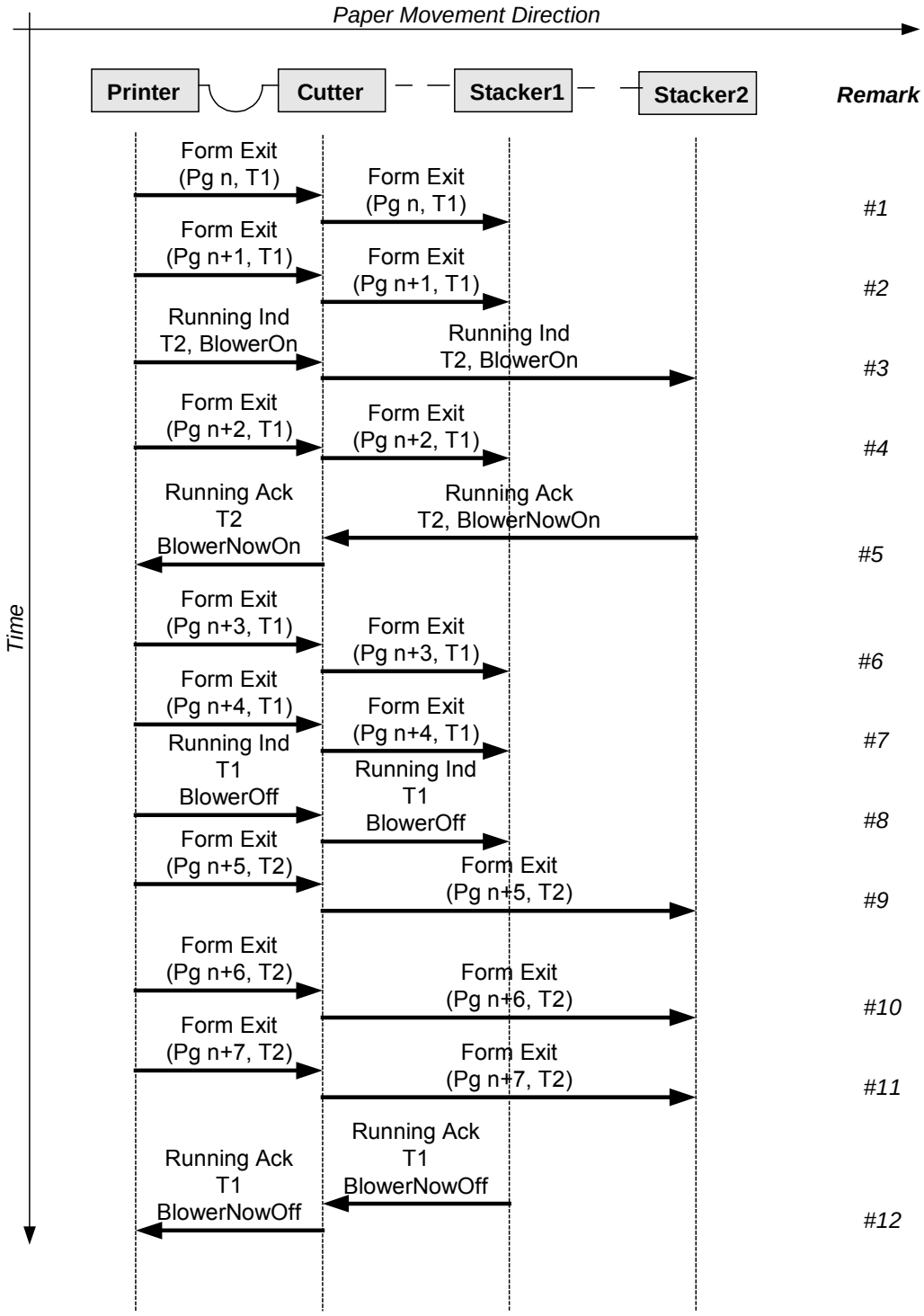
1) Printer gets the 1st sheet and sends a dedicated Running Indication Frame (Blower On) for tupel Printer-Cutter-Stacker1 to the cutter.
2) The Cutter sends the dedicated Running Indication Frame to the next device in the addressed tupel.
3) Stacker1 is the last device in the addressed tupel. For it is not Ready to work, it answers with a Running Acknowledge Frame (Blower Off) to the Cutter.
4) The Cutter passes this Running Acknowledge Frame (Blower Off) to the Printer.
5) The Printer does not start printing and displays an Error on its GUI.

The State of Stacker2 is not relevant for printing this job.

2.8.5 Changing the active Tupel with the dedicated Running Indication Frame

This example shows, how a change between Tupel is done with the dedicated Running Indication Frame.

In this example, there are two tupel defined:
Tupel 1: Printer, Cutter, Stacker1 and
Tupel 2: Printer, Cutter, Stacker2.

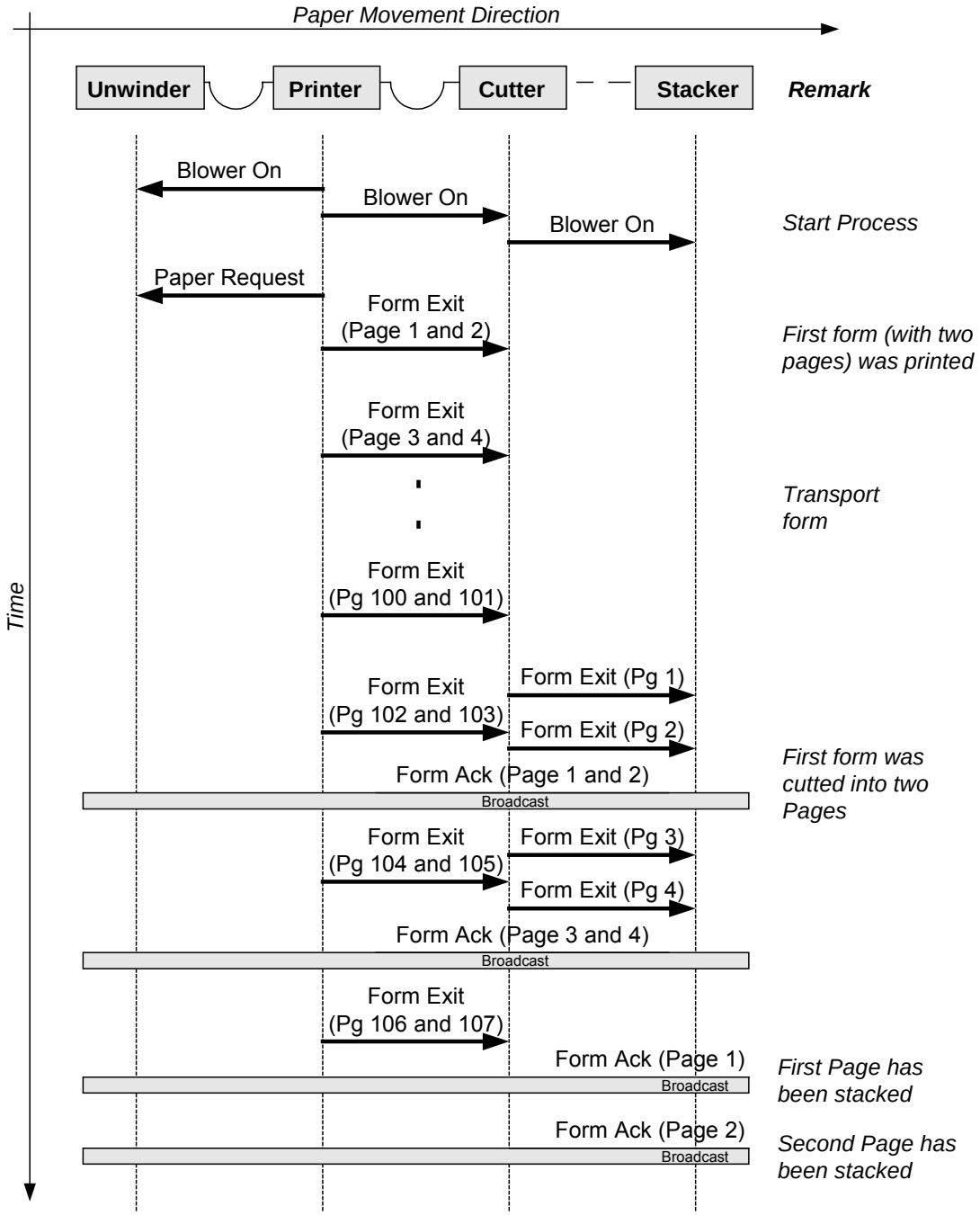


Operating sequence:

1) Page n for Tupel1 is printed and the regarding Form Exit is given to the regarding devices (Cutter, Stacker1).
2) Page n+1 for Tupel1 is printed and the regarding Form Exit is given to the regarding devices.
3) Tupel2 (Cutter, Stacker2) is started from the printer with a dedicated Running Indication (BlowerOn).
4) Page n+2 for Tupel1 is printed and the regarding Form Exit is given to the regarding devices.
5) The motors of the devices in Tupel2 are running and reported with a dedicated Running Acknowledge (BlowerNowOn) to the printer.
6) Page n+3 for Tupel1 is printed and the regarding Form Exit is given to the regarding devices.
7) Page n+4 for Tupel1 is printed and the regarding Form Exit is given to the regarding devices.
8) Tupel1 is no longer necessary and the motors for this tupel are switched off with a dedicated Running Indication (BlowerOff) for Tupel1 (Cutter, Stacker1).
   CAUTION: The Cutter is still running, for it is also part of Tupel2!
9) Page n+5 for Tupel2 is printed and the regarding Form Exit is given to the regarding devices(Cutter, Stacker2).
10) Page n+6 for Tupel2 is printed and the regarding Form Exit is given to the regarding devices.
11) Page n+7 for Tupel2 is printed and the regarding Form Exit is given to the regarding devices.
12) The motors of the devices in Tupel1 are now down and this is reported with a dedicated Running Acknowledge (BlowerNowOff) to the printer.
    CAUTION: The Cutter is still running, for it is also part of Tupel2!

Form Acknowledge Frames are not shown in this diagram.

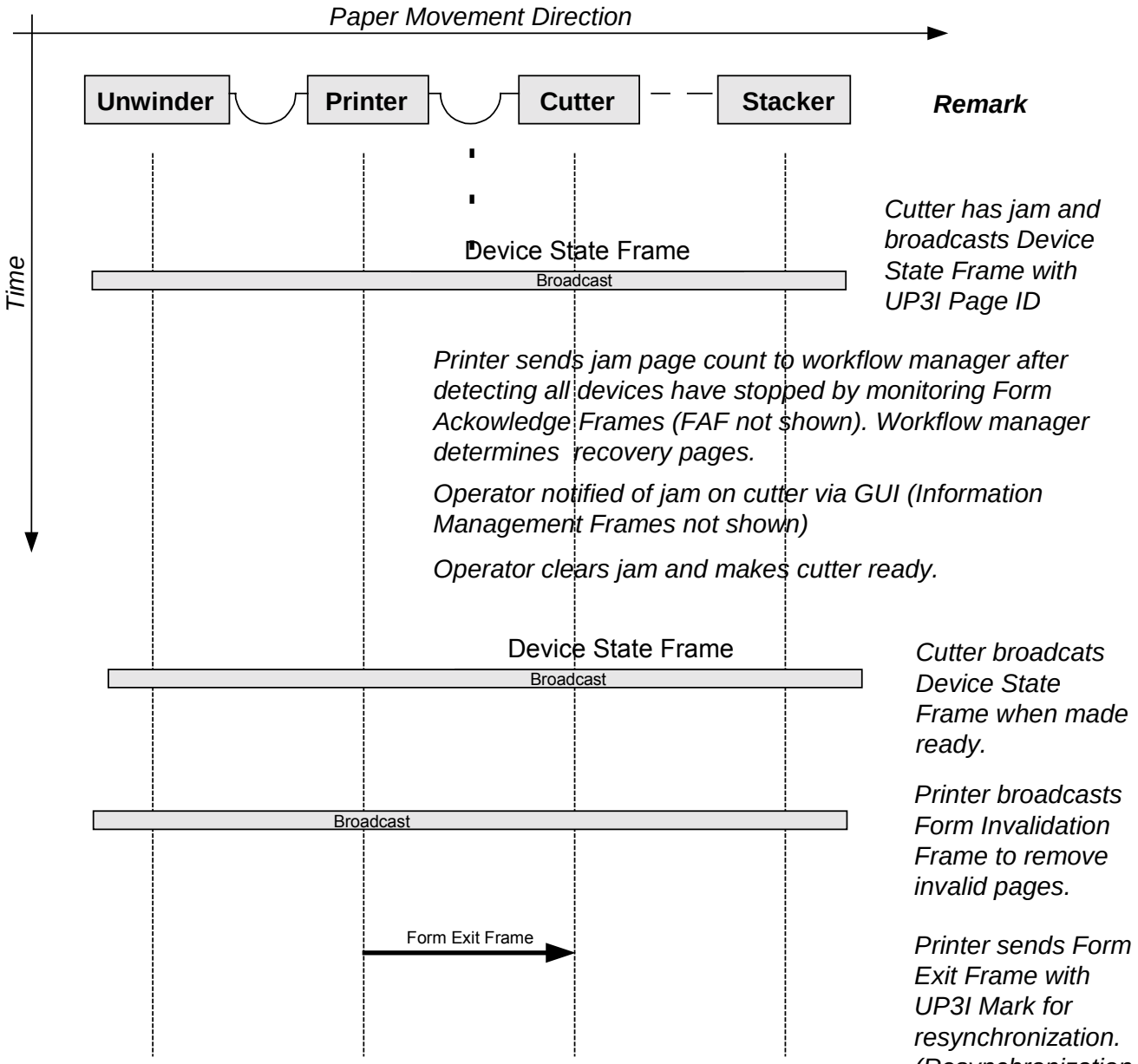
2.8.6 Running Production Line



The following pictures assume that a successful initialization was done and the production line is „synchron“ and the devices are „Device Ready“ State.

In this example the cutter cuts the continuous form paper into two pages. The isochronous „Paper Movement“ Frames are not shown.

2.8.7 Jam of the Production line causes Not Ready State

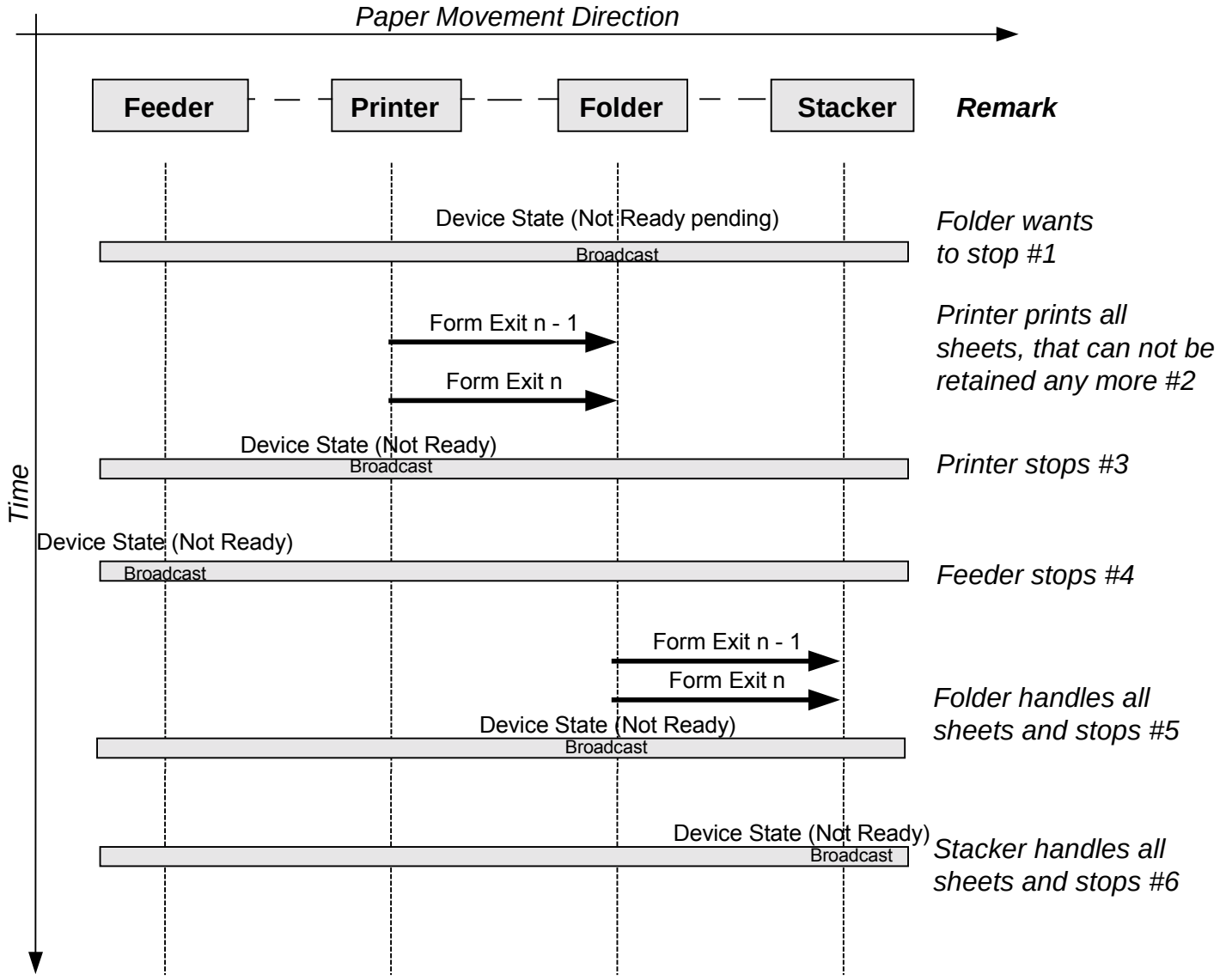


2.8.8 Stop Request from a downstream device in an active tupel in a cut sheet line

In print lines, in particular cut sheet print lines, a stop request from a downstream device needs to be synchronized. It is necessary for the requesting device to know when it received the last sheet, as there are possibly sheets still moving in the line, even when the printer has stopped.

This can be obtained with the following workflow:

Assumption: All devices are in ready state and the print line is active (printing).



Operating sequence:

1) Folders stop button is pressed.
2) Device State broadcast (Not Ready pending, Device transaction due to Operator Panel, Key pressed) is given to all devices. #1
3) Cut sheet printer finalizes all sheets, that cannot be kept back any more. #2
4) Cut sheet printer stops, broadcasts a stop device state. #3
5) Feeder stops after the printer, broadcasts a stop device state. #4
6) After the last sheet left the stacker, it changes into stop state and broadcasts a stop device state. #5
7) After the last sheet is stacked by the stacker, it changes into stop state and broadcasts a stop device state. #6

Form Acknowledge frames are not shown in the diagram.

Every device broadcasts a state frame, when it stopped.
All devices after the printer are allowed to change into stop state, when their predecessor is in stop state.
All devices before the printer are allowed to change into stop state, when the printer is in stop state.

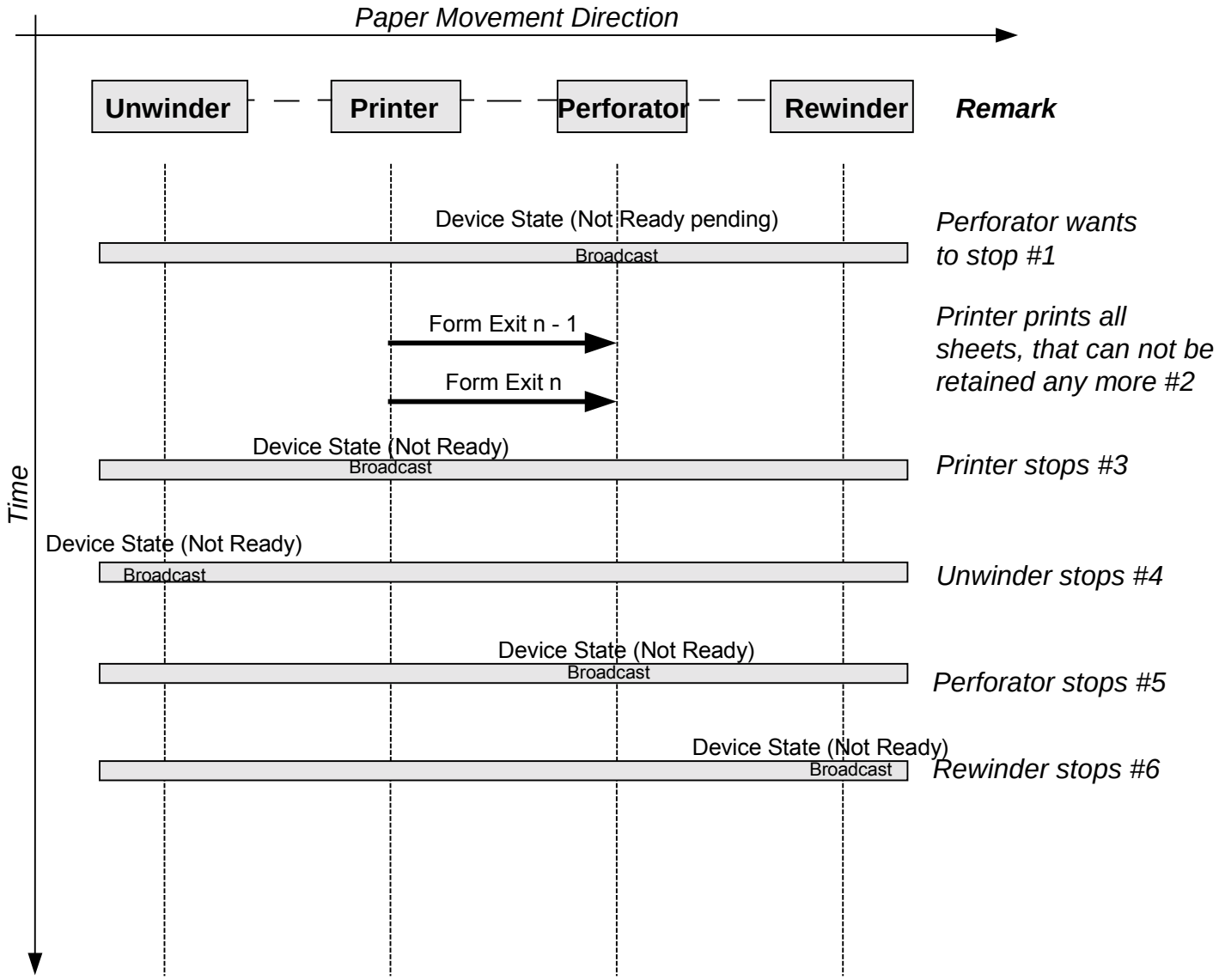
Caution:

This behavior is valid only, if the requesting device is member of an active tupel (see 2.4.1.1,Running Indication Frame). Devices in inactive tupels may change between ready and not ready state without any restrictions.

2.8.9 Not Ready Request from a downstream device in a web line.

A stop request from a downstream device in a web line is easier to handle, than in a cut sheet line. If the printer stops, all downstream devices have to stop as soon as possible.

Assumption: All devices are in ready state and the print line is active (printing).



Operating sequence:

1. Perforators stop button is pressed.
2. Device State broadcast (Not Ready pending, Device transaction due to Operator Panel, Key pressed) is given to all devices. #1
3. Web-Printer finalizes all sheets, that cannot be kept back any more. #2
4. Web-Printer stops, broadcasts a stop device state. #3
5. Unwinder stops after the printer, broadcasts a stop device state. #4
6. Perforator stops, it changes into stop state and broadcasts a stop device state. #5
7. After the last sheet is received by the rewinder, it changes into stop state and broadcasts a stop device state. #6

Form Acknowledge frames are not shown in the diagram.

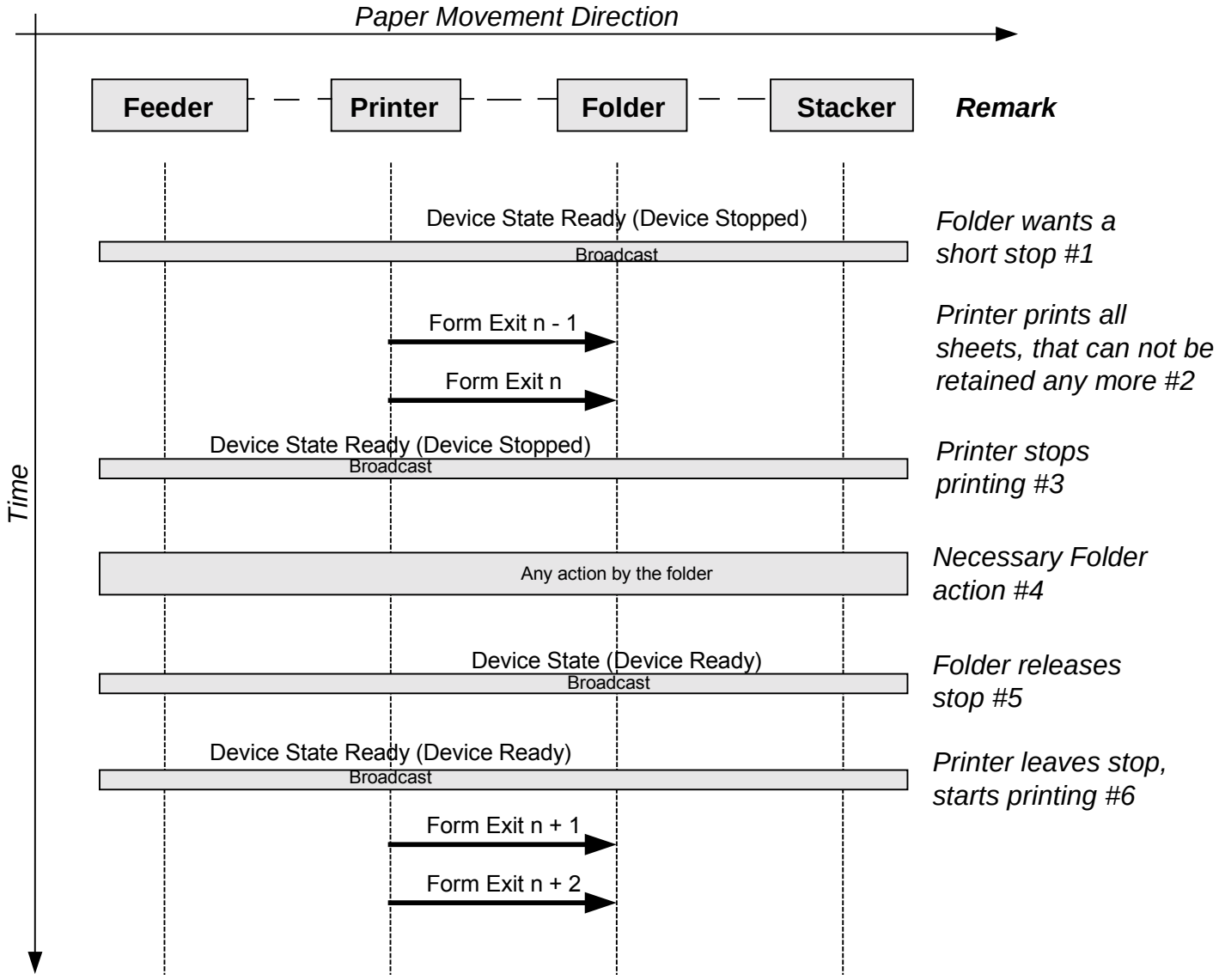
Every device broadcasts a state frame, when it stopped.
All devices after the printer are allowed to change into stop state, when their predecessor is in stop state.
All devices before the printer are allowed to change into stop state, when the printer is in stop state.
The difference between web- and cut sheet – lines is: In a pure web line, all devices (except the unwinder) may change into stop state at once. There is only a weak interconnection between the single devices.

But if all devices react in the same manner as they do in a cut sheet line, everything works fine.

2.8.10 Device Stop request from a downstream device in a cut sheet line (ADEV Stop)

If a cut sheet device wants to stop the line, it cannot be certain that all paper has stopped moving after the stop was requested . This is no problem in a web line, for there is some slack in between two web devices. So the cut sheet device requesting a stop needs to know if there are still sheets arriving or not. This can be achieved by the following sequence.

Assumption: All devices are in ready state and the print line is active (printing).



Operating sequence:

1. Folder wants to stop the print line for a short while. It sends a Device State with 'Ready' and 'Device Stopped' (ADEV Stop) #1
2. The printer prints all those sheets, that cannot be retained any more. #2
3. After that the printer sends a Device State with 'Ready' and 'Device Stopped' (ADEV Stop) #3.
   This Device State can be seen as an acknowledgement for the ADEV stop request.
4. The folder now knows, there are only two more sheets on the way. If these sheets are handled by the folder, its necessary action can be done. #4
5. The folder releases the ADEV stop situation by sending a Device State with 'Ready' and 'Device Ready'. #5
6. The printer leaves ADEV stop by sending a Device State with 'Ready' and 'Device Ready'. #6

The feeder and the stacker devices are not concerned by the ADEV stop, only those devices, that are between printer and the requesting device.

Form Acknowledge frames are not shown in the diagram.

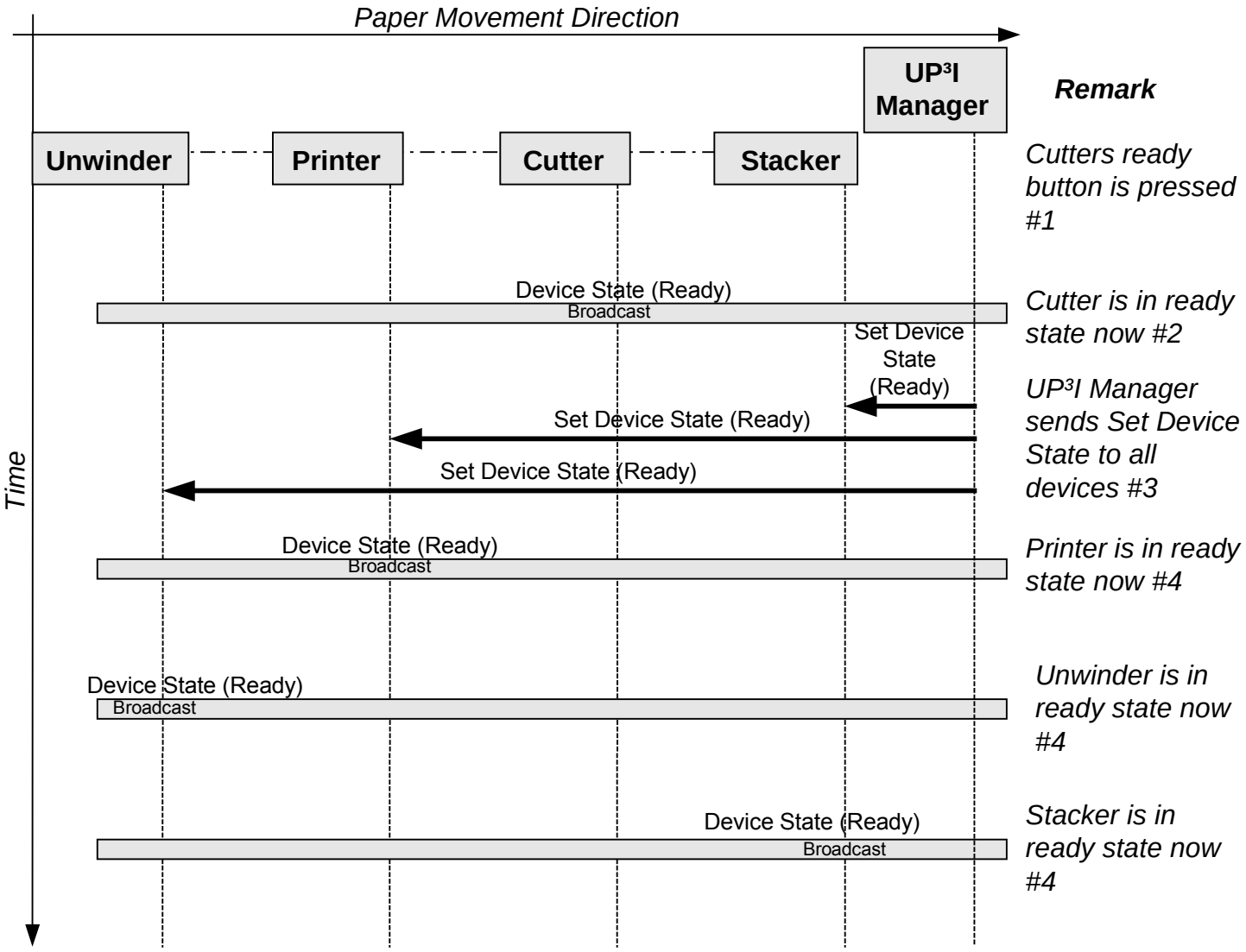
2.8.11 Ready Request from any device

2.8.11.1 With UP³I Manager

It is recommended to be able to set the complete UP³I print line to ready state from every device in the line, as it is possible to do this via the UP³I Manager.

This can be obtained by the following workflow:

Assumption: All device are in not ready state.



Operating sequence:

1. Ready button on any device (in this example the cutter) is pressed. #1
2. This device changed into ready state and broadcasts this state change with 'Device transaction due to Operator Panel (Key pressed)'. #2
3. A Set Device State (Ready) is sent to every device (from the UP³I Manager), but not to the cutter. #3
4. Each device changes into ready state and broadcasts this state change. #4

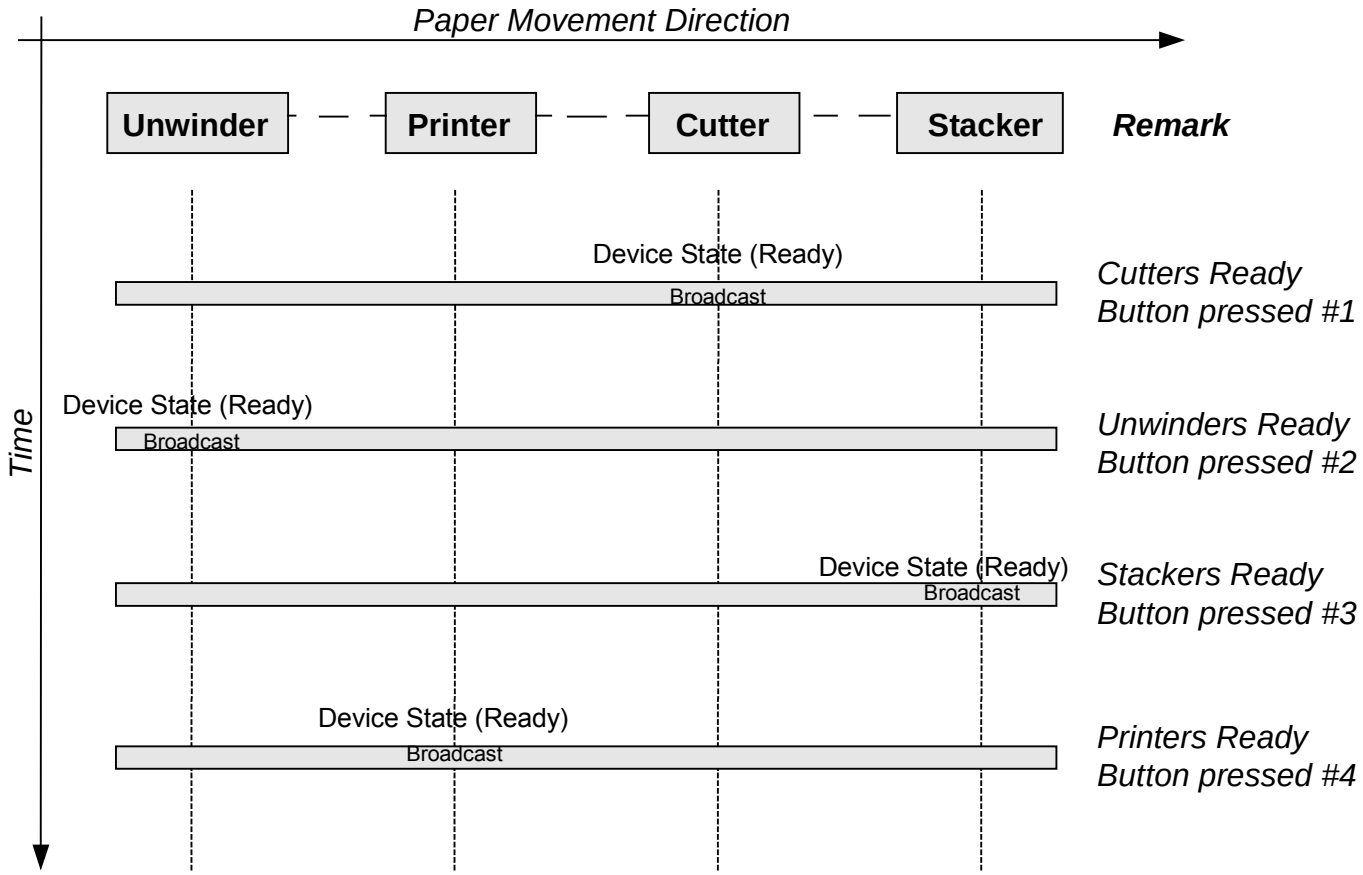
If any device cannot change into ready state because of an error, it broadcasts this with a Device State Frame and the normal error handling takes place.

2.8.11.2 Without UP³I Manager

Without UP³I Manager each device is set to ready state separately via the ready button.

This can be obtained by the following workflow:

Assumption: All device are in not ready state.



Operating sequence:

1. Cutters ready button pressed, cutter sends Device State Broadcast with 'Device transaction due to Operator Panel (Key pressed)'. All other Devices stay not ready. #1
2. Unwinders ready button pressed, unwinder sends Device State Broadcast with 'Device transaction due to Operator Panel (Key pressed)'. Cutter and unwinder are in ready state now. #2
3. Stackers ready button pressed, stacker sends Device State Broadcast with 'Device transaction due to Operator Panel (Key pressed)'. Cutter, unwinder and Stacker are in ready state now. #3
4. Printers ready button pressed, printer sends Device State Broadcast with 'Device transaction due to Operator Panel (Key pressed)'. All Devices are in ready state now. #4

If any device cannot change into ready state because of an error, it broadcasts this with a Device State Frame and the normal error handling takes place.

3 Extensions for Print Languages

3.1 Extension for the Intelligent Printer Data Stream (IPDS)

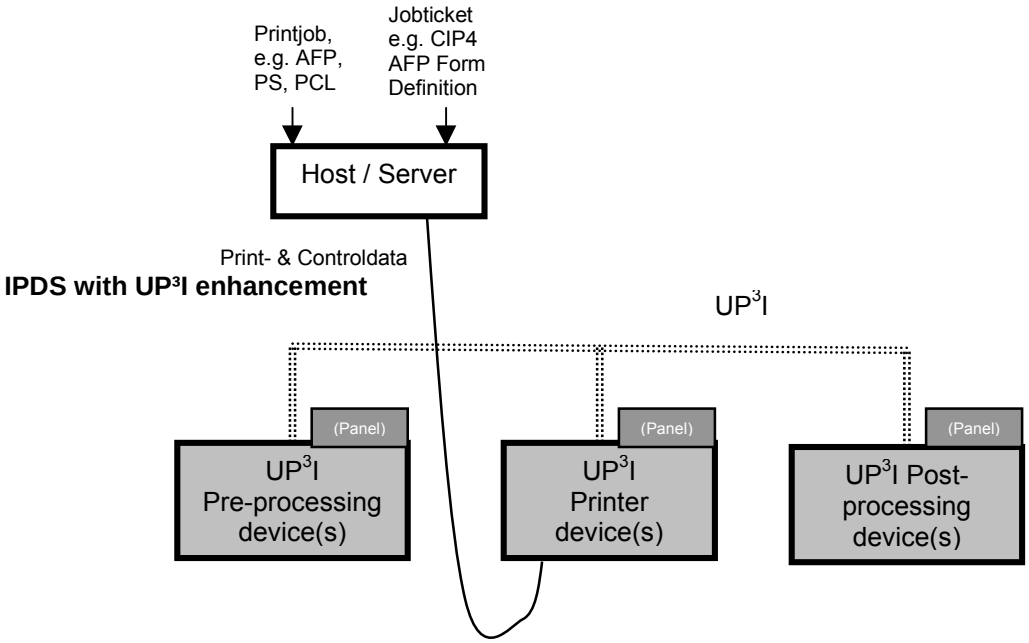


Figure 3-1: IPDS UP³I Workflow

The workflow above shows, how the UP³I control commands reaches the UP³I production line via an IPDS conduit for UP³I. There are IPDS extensions, released by the IBM corporation, that allows any IPDS print server to use UP³I commands for finishing control.

This appendix describes this IPDS conduit for UP³I. This conduit is defined by IBM. It consists in particular of the 0x8E triplet for the AFO and the XOH-DGB commands and of the UP³I Tupel Self Defining Field and the UP³I Paper Input Media Self Defining Field for the XOH-OPC.

During distribution of this UP³I version 1.0, these IPDS commands are not yet released. The final description can be found at IBM's specification "Data Stream and Object Architecture, Intelligent Printer Data Stream Reference, S544-3417-06".

The following commands and triplets are defined by UP³I and are referenced by IBM's specification:
UP³I Finishing Device Entry in chapter 3.1.2.2, which is part of the UP³I Tupel SDF in XOH-OPC and carries the information of the available UP³I devices.
The content of the UP³I Paper Input Media SDF, chapter 3.1.2.3, that describes details of the available paper.
The content of the UP³I Finishing Operation Triplet, chapter 3.1.3, Extensions for XOH-DGB and AFO (Apply Finishing Operation). This triplet is the conduit for the UP³I Form Finishing Operation Triplet, that controls the finishing operation.
The sense bytes 8 – 18 in sense byte format 8, see chapter 3.1.6.6, Sense Byte Format 8

The data format to and from the IPDS host is "big endian."

3.1.1 UP³I property pair for Sense Type and Model (STM)

Property Pair	Meaning
X'F101'	UP³I finishing supported. With this property pair an existing UP³I interface is reported

A printer that reports this property pair supports the whole set of UP³I related commands, as there are:

XOH-OPC UP³I Tupel SDF with the UP³I Finishing Device Entry
XOH-OPC UP³I Paper Input Media SDF
UP³I Finishing Operation Triplet in the XOH-DGB and the AFO commands
XOA–Discard Unstacked Pages
UP³I Exceptions including Sense Format 8 and Set Recovery.

Indicating UP³I support does not necessarily mean that a UP³I device is available in the print system. This is indicated in OPC.

Property Pair	Meaning
X'80F5'	XOA-Discard Unstacked Pages command is supported.

A printer that reports this property pair supports the XOA- Discard Unstacked Pages command.

3.1.2 Extensions for the XOH-OPC:

3.1.2.1 UP³I Tupel SDF

This SDF reports the physical order and properties of the UP³I devices connected to the print system.

Description of the UP³I Tupel SDF

Offset	**Type**	**Name**	**Range (Hex)**	**Meaning**
0 – 1	UBIN	SDF length	0009 – nnnn	Length of this SDF, including the field itself.
2 – 3	CODE	UP³I TUP SDF ID	0019	UP³I Tupel SDF ID
4 – 5	UBIN	UP³I Tupel ID	0001 – FFFF 0000	UP³I Tupel ID Reserved
6 – end	Finishing device information. One or more UP³I finishing device entries.			

One UP³I tupel SDF reports the UP³I devices being in one tupel. There are as many UP³I tupel SDFs in an OPC as there are tupels in an UP³I processing line.
The paper flow within one tupel is described by the paper Sequence IDs, the devices are ordered ascending. The paper moves from the device with the lowest paper sequence ID to the device with the next higher one.

3.1.2.2 UP³I Finishing Device Entry

Offset	**Type**	**Name**	**Range (Hex)**	**Meaning**
0		Entry length	03 – FF	Length of the entry including the field itself.
1		UP³I FD-entry ID	01	UP³I finishing device entry ID
2	CODE	Paper Seq ID	00 - FF	paper sequence ID of finishing device
3 – end	One or more UP³I SDF triplets related to device with the paper sequence ID above.			

One UP³I finishing device entry reports the characteristics of one UP³I device. Several UP³I SDF triplets may be included in one UP³I Tupel SDF. All these entries describe one UP³I device.
The following UP³I Self Defining Field Triplets, specified in chapter 2.3.3, Self Defining Field Frame, are supported by the UP³I Finishing device entry:

- UP³I Version Triplet
- Device Type Triplet
- Self Defining Field Triplet
- Paper Input Format Triplet
- Paper Output Format Triplet
- UP³I Product ID Triplet
- Postprocessing Printer Print Data Format ID Triplet

3.1.2.3 UP³I Paper Input Media SDF

This SDF reports the media attributes of all media that exist in the UP³I line.

Description of the UP³I Paper Input Media SDF

Offset	Type	Name	Range (Hex)	Meaning
0 – 1	UBIN	Length	0005 – FFFF	Length of this SDF, including the field itself.
2 – 3	CODE	UP³I PIM SDF ID	001A	UP³I Paper Input Media SDF ID
4	CODE	Media Source ID	00 – FF	Media Source ID as defined in OPC Printable Area SDF
5 – end				UP³I Paper Input Media Triplet or Paper Input Media Extended Triplet with its Sub Triplets, as defined in the UP³I specification; extra bytes beyond the UP³I-defined bytes are ignored.

The relation between the Printable-Area SDF and this UP³I Media Attribute SDF is done by the
- Media Source ID (Byte 4) of the Printable-Area SDF and by the
- Input Media ID (Byte 3) of the UP³I Paper Input Media Triplet. For every media that is available in the UP³I line there may exist a Paper Input Media Triplet with its Sub Triplets.

The UP³I paper input media (extended) triplet and its sub triplets are described in the Self Defining Field Frame, chapter 2.3.3. For the UP³I paper input media (extended) triplet is optional in the UP³I specification, the length of this SDF can be only five bytes, that means, there are no data available.

Paper Input Media (Extended) Triplet supports the following sub triplets:
- Paper Input Media Name Sub Triplet
- Paper Input Media Coating Sub Triplet
- Paper Input Media Brightness Sub Triplet
- Paper Input Media Color Sub Triplet
- Paper Input Media Imagable Side Sub Triplet
- Paper Input Media Color Name Sub Triplet
- Paper Input Media Set Count Sub Triplet
- Paper Input Media Opacity Sub Triplet
- Paper Input Media Pre Printed Sub Triplet
- Paper Input Media Recycled Sub Triplet
- Paper Input Media Roll Diameter Sub Triplet
- Paper Input Media Thickness Sub Triplet
- Paper Input Media User Media Type Sub Triplet
- Paper Input Media Weight Sub Triplet
- Paper Input Media Pin Hole Sub Triplet
- Paper Input Media Dimensions Sub Triplet
- Paper Input Media Ordered Set Piece Sub Triplet
- Paper Input Media Predrilled Hole Count Sub Triplet

Note: The Media Source ID allows the host program to associate the Media Attributes contained in the UP³I Input Media (extended) Triplet with the Media Sources (Input Bins) specified in the OPC-Printable Area-SDF.

Caution: The direction of the width and length parameters in the paper input media triplet refer (different than it is in IPDS) to the paper movement direction.
The units for distance parameters used in the UP³I specification are different to those in the IPDS specification. UP³I uses millipoints, one millipoint is 1/72000 inch.

However: These values are not relevant for the host software, because they are also defined in the OPC-Printable Area-SDF (i.e. Length/Width of Media Presentation space, Offset/Extend of Printable Area, some Media characteristics). The host program uses the OPC-Printable Area-SDF to control its presentation process. The IPDS controller together with the UP³I Manager guarantee that the information contained in both SDFs is consistent. This also applies to other necessarily double defined information(i.e. media names, media types, media Imagable sides...)

3.1.3 Extensions for XOH-DGB and AFO (Apply Finishing Operation)

UP³I Finishing Operation Triplet

Description of the UP³I Finishing Operation Triplet

Offset	Type	Name	Range (Hex)	Meaning
0	UBIN	Length	0D – FE	Length of this triplet, including this field itself.
1	CODE	TID	8E	Identifies the UP³I Finishing Operation triplet
2 – 3			0000	reserved
4 – end	Data			UP³I Form Finishing Operation Triplet as defined in the UP³I specification; this field contains bytes 4 – end of the UP³I Form Finishing Operation triplet; extra bytes beyond the UP³I-defined bytes are ignored.

The UP³I Form Finishing Operation Triplet is specified in chapter 2.4.3, Form Exit Frame.

Restriction: Finishing operation types "Paper Input / Page interpose" (see 2.4.3, Form Exit Frame, Form Finishing Operation Triplet, Byte 4) are rejected in the IPDS triplet 8E. The existing IPDS interpose functionality has to be used instead.

This triplet can be carried on the Define Group Boundary (DGB) command that starts a group and is valid for all sheets in the group.
This triplet can be carried on an Apply Finishing Operations (AFO) command for a single sheet operation.

3.1.4 XOA – Discard Unstacked Pages

The XOA Discard Unstacked Pages (DUP) command deletes all buffered data from the printer storage (just like DBD), discards all printed but unstacked pages, and returns the printer to home state.

Description of the XOA – Discard Unstacked Pages

Offset	Type	Name	Range (Hex)	Meaning
0 – 1	CODE	Order code	F500	Discard Unstacked Pages (DUP) order code

3.1.5 Finishing Fidelity

When one of the following error conditions occur, the printer normally generates exception ID X'027E..00' (invalid or unsupported parameter specification for a UP3i-controlled device) with the specified action code:

- A print job uses UP3i finishing, but the specific finishing operation cannot be performed (action code X'01' or X'06')
- The printer detects a syntax error in a UP3i Finishing Operation (X'8E') triplet (action code X'01' or X'06')
- A pre- or post-processing device detects a syntax error in a UP3i Finishing Operation (X'8E') triplet (action code X'0A')
- The printer detects a syntax or position-check error in a UP3i Print Data object (action code X'01' or X'06')
- A pre- or post-processing device detects a syntax or position-check error in a UP3i Print Data object (action code X'0A')

The host can use a Finishing Fidelity (X'88') triplet to tell the printer to either report the X'027E..00' NACK or to suppress the NACK. When the triplet from the host says to continue and suppress the NACK, the printer (or UP3i device) changes the finishing operation to the default operation (set on the printer console or UP3i Manager console) and continues.

3.1.6 UP³I-specific exception IDs

3.1.6.1 Intervention Required

X'407E..00' Intervention required on a UP³I-controlled device

Action Code: X'08', X'0A', X'1A', or X'22'

Explanation: A pre-processing or post-processing device attached to the printer has reported an intervention required condition. The specific error is identified in the sense bytes 8-9.
This exception ID uses sense-byte format 8.

Alternate Exception Action: None

Page Continuation Action: None

Support: Mandatory, if the printer supports the UP³I interface.

3.1.6.2 Specification Check

X'027E..00' Invalid or unsupported parameter specification for a UP³I-controlled device

Action Code: X'01', X'06', X'09' or X'0A'

Explanation: A specification error was detected for a UP³I pre-processing or post-processing device. The specific error is identified in the sense bytes 8-9.
The UP³I Finishing Operation (X'8E) triplet is used to specify finishing operations for UP³I pre-processing or post-processing devices attached to the printer.
This exception ID uses sense-byte format 8.

Alternate Exception Action: None

Page Continuation Action: None

Support: Mandatory, if the printer supports the UP³I interface

3.1.6.3 Conditions Requiring Host Notification

X'0109..00' Supported finishing operations changed

Use existing NACK

X'017E..00' Condition requiring host notification on a UP³I-controlled device

Action Code: X'09', 0x0A, X'1A', or X'1D'

> **Explanation:** A pre-processing or post-processing device attached to the printer has reported a condition requiring host notification. The specific error is identified in the sense bytes 8-9.
> This exception ID uses sense-byte format 8.

Alternate Exception Action: None

Page Continuation Action: None

Support: Mandatory, if the printer supports the UP³I interface

3.1.6.4 Equipment Check with Intervention Required

X'507E..00' Intervention required because of an equipment check on a UP³I-controlled device

Action Code: X'08', X'09', X'16', or X'22'

Explanation: A pre-processing or post-processing device attached to the printer has reported an equipment check error that is also an intervention required condition. The specific error is identified in the sense bytes 8-9.
This exception ID uses sense-byte format 8.

Alternate Exception Action: None

Page Continuation Action: None

Support: Mandatory, if the printer supports the UP³I interface

3.1.6.5 Equipment Check

X'107E..00' Equipment check on a UP³I-controlled device

Action Code: X'09', X'22' or X'23'

Explanation: A pre-processing or post-processing device attached to the printer has reported an equipment check error that can not be corrected by an operator. The specific error is identified in the sense bytes 8-9.
This exception ID uses sense-byte format 8.

Alternate Exception Action: None

Page Continuation Action: None

Support: Mandatory, if the printer supports the UP³I interface

3.1.6.6 Sense Byte Format 8

Bytes 8 – 18 are defined by UP³I, all other bytes are part of IBM's exception reporting.

Offset	Range (Hex)		
0 – 1		Exception Class & Exception ID Bytes 1 & 2 of the Error Code	
2		Action Code	Counter adjustments
	01	Data stream exception	Normal IPDS handling
	06	Function no longer achievable	No change
	08	Physical media jam	RPC ⇦ JPC CPC ⇦ JPC OPC ⇦ JPC JPC no change SPC ⇦ JPC
	09 16 1A 1D 23	Data related print exceptions Hardware-Related Print Exception Re-drive buffered pages Printer characteristics changed Temporary HW exception	RPC ⇦ CPC CPC no change OPC no change JPC no change SPC no change
	0A	Pre/Post processor exception	RPC ⇦ JPC CPC ⇦ JPC OPC ⇦ JPC JPC no change SPC ⇦ no change
	22	Printer inoperative	Defined by the printer
3		Printer Dynamic Conditions	
4	DE	Device Error	
5	08	Format Identifier – UP3I-specific sense data format	
6 – 7	0000	IPDS Command in process	
8 – 9	xxxx	UP3I specific error code. This error code is device specific as defined in chapter 2.3.5, Device State Frame in the State Description Triplet.	
10	xx	Paper Sequence ID of the P/P-device which caused the exception	
11 – 12	0000	reserved (for set error recovery)	
13 – 14	0000	reserved (for set error recovery)	
15 – 16	xxxx	ID of the active UP³I tupel	
17 – 18	0000	reserved	
19	00	Byte 3 of the error code	
20 – 23		Page Identifier	

3.1.6.7 Action Codes

All action codes are defined in the IPDS specification. For action code X'06' there is the following UP³I regarding addition:
Exceptions with action code X'06' are reported after processing an End Page command for a sheet to be finished; the End Page command is processed and the received page counter is incremented. The finishing operation is not applied to the sheet and the finishing operation triplet is discarded.

3.1.7 Example: UP³I Finishing Conduit in MO:DCA / IPDS data streams

Purpose: To provide an example showing the kind of processing the host presentation software must do in order to support the new UP³I Finishing Triplet in the MO:DCA and IPDS data streams.

The example should help implementers of MODCA data generators to understand how to control the desired UP³I functionality on a MO:DCA / IPDS platform.

Description: The sample shows a Mixplex (Simplex and Duplex) application printed on a cut-sheet system. The application is printed one-up.

Notes: The parameters of the MFC structured field may affect the generation of sheet ejects even when N-up processing is active. For a description of how sheet and partition ejects are handled when N-up processing is active and an MFC is specified in the Medium Map, see the Media Eject Control triplet 0x45.

Bibliography: See chapter 5.1, Literature

MO:DCA data stream with UP³I triplets

1			**Begin of Job**						
2	**DEG**	**MFC: MFCScpe=0x02, MedColl=0x00, Triplet_0x8E=shrink wrap**		**S**					
3	BDT								
4	BNG CUSTOMER_1								
5	**IMM LI_BEG**	**MMC: MediaSource=standard, DuplexControl=Simplex, NupControl=OneUp MFC: MFCScpe=0x05, MedColl=0x01, Triplet_0x8E=offset to left MFC: MFCScpe=0x04, MedColl=0x00, Triplet_0x8E=perforation cut**			**O L**	**P C**			
6		BP cover letter							
7		EP							
8	**IMM LI_BEGI**	**MMC: MediaSource=inserterBin, DuplexControl=Simplex, NupControl=OneUp MFC: MFCScpe=0x05, MedColl=0x02, Triplet_0x8E=offset to left**							
9		BP (place holder for Interposer sheet)							
10		EP							
11		BNG SUMMARY							
12	**IMM LSRF_BEG**	**MMC: MediaSource=wideForm, DuplexControl=Simplex, NupControl=OneUp MFC: MFCScpe=0x05, MedColl=0x02, Triplet_0x8E=offset to left MFC: MFCScpe=0x05, MedColl=0x01, Triplet_0x8E=edge stitch MFC: MFCScpe=0x04, MedColl=0x00, Triplet_0x8E=rotate, Triplet_0x8E=fold**				**E S**	**R**	**F**	
13			BP page 1						
14			EP						
15	**IMM LS_CON**	**MMC: MediaSource=standard, DuplexControl=Simplex, NupControl=OneUp MFC: MFCScpe=0x05, MedColl=0x02, Triplet_0x8E=offset to left, Triplet_0x8E=edge stitch**							
16			BP page 2						
17			EP						
18		ENG SUMMARY							
19		BNG ACCOUNT_1							
20	**IMM LC_BEG**	**MMC: MediaSource=standard, DuplexControl=Normal, NupControl=OneUp MFC: MFCScpe=0x05, MedColl=0x02, Triplet_0x8E=offset to left MFC: MFCScpe=0x05, MedColl=0x01, Triplet_0x8E=corner staple**				**C S**			
21			BP page 1						
22			EP						
23			...						
24			BP page 9						
25			EP						
26		ENG ACCOUNT_1							
27		BNG ACCOUNT_2							
28	**IMM LC_BEG**	**MMC: MediaSource=standard, DuplexControl=Normal, NupControl=OneUp MFC: MFCScpe=0x05, MedColl=0x02, Triplet_0x8E=offset to left MFC: MFCScpe=0x05, MedColl=0x01, Triplet_0x8E=corner staple**				**C S**			
29			BP page 1						
30			EP						
31			...						
32			BP page 6						
33			EP						
34		ENG ACCOUNT_2							
35		ENG CUSTOMER_1							
36		BNG CUSTOMER_2							
37	**IMM RI_BEG**	**MMC: MediaSource=standard, DuplexControl=Simplex, NupControl=OneUp MFC: MFCScpe=0x05, MedColl=0x01, Triplet_0x8E=offset to right MFC: MFCScpe=0x04, MedColl=0x00, Triplet_0x8E=perforation cut**			**O R**	**P C**			
38		BP cover letter							
39		EP							
40	**IMM RI_BEGI**	**MMC: MediaSource=inserterBin, DuplexControl=Simplex, NupControl=OneUp MFC: MFCScpe=0x05, MedColl=0x02, Triplet_0x8E=offset to right**							
41		BP (place holder for Interposer sheet)							
42		EP							
43		BNG SUMMARY							
44	**IMM RSRF_BEG**	**MMC: MediaSource=wideForm, DuplexControl=Simplex, NupControl=OneUp MFC: MFCScpe=0x05, MedColl=0x02, Triplet_0x8E=offset to right MFC: MFCScpe=0x05, MedColl=0x01, Triplet_0x8E=edge stitch MFC: MFCScpe=0x04, MedColl=0x00, Triplet_0x8E=rotate MFC: MFCScpe=0x04, MedColl=0x00, Triplet_0x8E=fold**				**E S**	**R**	**F**	
45			BP page 1						
46			EP						
47	**IMM RS_CON**	**MMC: MediaSource=standard, DuplexControl=Simplex, NupControl=OneUp MFC: MFCScpe=0x05, MedColl=0x02, Triplet_0x8E=offset to right MFC: MFCScpe=0x05, MedColl=0x02, Triplet_0x8E=edge stitch**							
48			BP page 2						
49			EP						

MO:DCA data stream with UP³I triplets

1	2	3	4	5

50		BP page 3	
51		EP	
52		ENG SUMMARY	
53		BNG ACCOUNT_1	
54	**IMM RC_BEG**	**MMC: MediaSource=standard, DuplexControl=Normal, NupControl=OneUp** **MFC: MFCScpe=0x05, MedColl=0x02, Triplet_0x8E=offset to right** **MFC: MFCScpe=0x05, MedColl=0x01, Triplet_0x8E=corner staple**	C S
55		BP page 1	
56		EP	
57		...	
58		BP page 3	
59		EP	
60		ENG ACCOUNT_1	
61	ENG CUSTOMER_2		
62	EDT		
63		**End of Job**	

IPDS data stream with UP³I triplets

							SGO Level=0xA0, Operation=0x01 (Keep together as a print unit) DGB Initiate, Level=0xA0, Group_ID_Triplet=MVS and VSE data format	1
S							SGO Level=0x90, Operation=0x04 (Finishing) DGB Initiate, Level=0x90, Triplet_0x8E=shrink wrap	2
								3
								4
	OL	PC					SGO Level=0x80, Operation=0x04 (Finishing) DGB Initiate, Level=0x80, Triplet_0x8E=offset to left	5
							SGO Level=0x70, Operation=0x04 (Finishing) DGB Initiate, Level=0x70, Triplet_0x8E=perforation cut	
							LCC MediaSource=standard, Simplex, OneUp BP cover letter	6
							EP	7
							DGB Terminate, Level=0x70, Triplet_0x8E=perforation cut	
								8
							LCC MediaSource=inserterBin, Simplex, OneUp BP (place holder for Interposer sheet)	9
							EP	10
								11
		ES	R	F			SGO Level=0x70, Operation=0x04 (Finishing) DGB Initiate, Level=0x70, Triplet_0x8E=edge stitch	12
							SGO Level=0x60, Operation=0x04 (Finishing) DGB Initiate, Level=0x60, Triplet_0x8E=rotate, Triplet_0x8E= fold	
							LCC MediaSource=wideForm, Simplex, OneUp BP page 1	13
							EP	14
							DGB Terminate, Level=0x60, Triplet_0x8E= fold, Triplet_0x8E=rotate	
								15
							LCC MediaSource=standard, Simplex, OneUp BP page 2	16
							EP	17
								18
								19
		CS					DGB Terminate, Level=0x70 Triplet_0x8E=edge stitch SGO Level=0x70, Operation=0x04 (Finishing) DGB Initiate, Level=0x70, Triplet_0x8E=corner staple	20
							LCC MediaSource=standard, Duplex, OneUp BP page 1	21
							EP	22
							...	23
							BP page 9	24
							EP	25
								26
								27
		CS					DGB Terminate, Level=0x70, Triplet_0x8E=corner staple SGO Level=0x70, Operation=0x04 (Finishing) DGB Initiate, Level=0x70, Triplet_0x8E=corner staple	28
							LCC MediaSource=standard, Duplex, OneUp BP page 1	29
							EP	30
							...	31
							BP page 6	32
							EP	33
								34
								35
								36
	OR	PC					DGB Terminate, Level=0x70, Triplet_0x8E=corner staple DGB Terminate, Level=0x80, Triplet_0x8E=offset to left SGO Level=0x80, Operation=0x04 (Finishing) DGB Initiate, Level=0x80, Triplet_0x8E=offset to right	37
							SGO Level=0x70, Operation=0x04 (Finishing) DGB Initiate, Level=0x70, Triplet_0x8E=Perforation cut	
							LCC MediaSource=standard, Simplex, OneUp BP cover letter	38
							EP	39
							DGB Terminate, Level=0x70, Triplet_0x8E=Perforation cut	
								40
							LCC MediaSource=inserterBin, Simplex, OneUp BP (place holder for Interposer sheet)	41

1 2 3 4 5

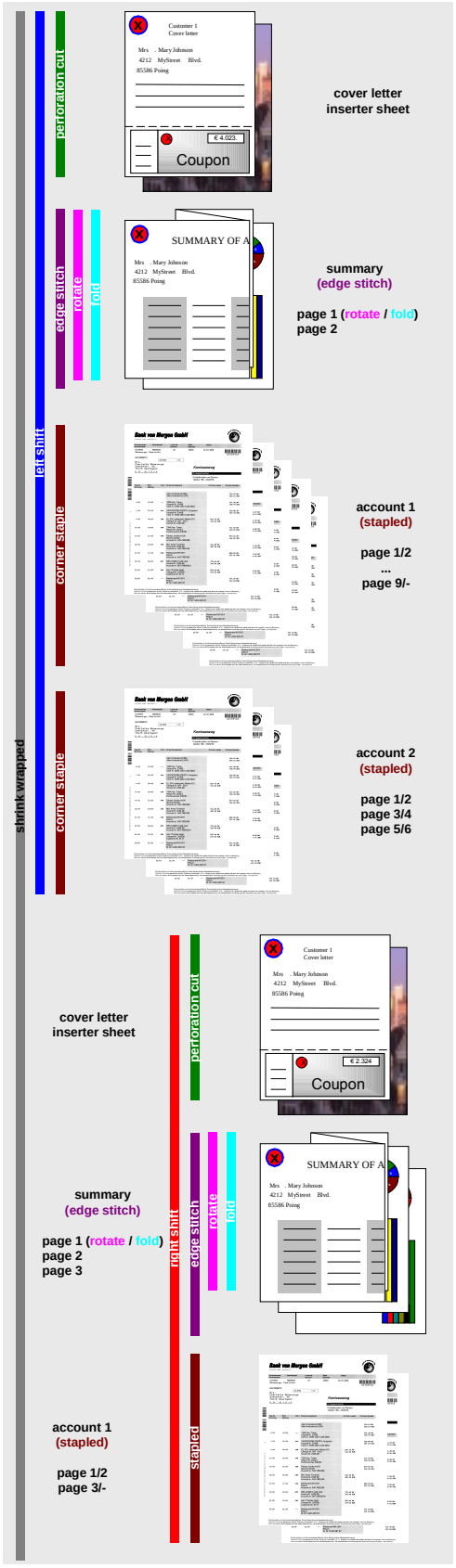
IPDS data stream with UP³I triplets

					EP	42
						43
		E S	R	F	**SGO Level=0x70, Operation=0x04 (Finishing)** **DGB Initiate, Level=0x70, Triplet_0x8E=edge stitch** **SGO Level=0x60, Operation=0x04 (Finishing)** **DGB Initiate, Level=0x60, Triplet_0x8E=rotate** **SGO Level=0x50, Operation=0x04 (Finishing)** **DGB Initiate, Level=0x50, Triplet_0x8E=fold**	**44**
					LCC MediaSource=wideForm, Simplex, OneUp BP page 1	**45**
					EP **DGB Terminate, Level=0x50, Triplet_0x8E=fold** **DGB Terminate, Level=0x60, Triplet_0x8E=rotate**	**46**
						47
					LCC MediaSource=standard, Simplex, OneUp BP page 2	**48**
					EP	**49**
					BP page 3	**50**
					EP	**51**
						52
						53
		C S			**DGB Terminate, Level=0x70, Triplet_0x8E=edge stitch** **SGO Level=0x70, Operation=0x04 (Finishing)** **DGB Initiate, Level=0x70, Triplet_0x8E=corner staple**	**54**
					LCC MediaSource=standardBin, Duplex, OneUp BP page 1	**55**
					EP	**56**
					...	**57**
					BP page 3	**58**
					EP	**59**
						60
						61
					DGB Terminate, Level=0x70, Triplet_0x8E=corner staple **DGB Terminate, Level=0x80, Triplet_0x8E=offset to right** **DGB Terminate, Level=0x90, Triplet_0x8E=shrink wrap**	**62**
					DGB Terminate, Level=0xA0, Group_ID_Triplet=MVS and VSE data format	**63**

Data Stream processing

- Line 1: the presentation software sends the complete print file as a single unit (DGB level 0xA0).
- Line 2: the DEG in the Formdef contains a *document-level* MFC (shrink wrap) for all documents in the file. As a result, the presentation software initiates DGB level 0x90 on the IPDS side.
- Line 3: the print file starts with a BDT. There is no need to start a new DGB level because the current one (level 0x90) is still empty.
- Line 4: page group SF's do not necessarily influence the DGB level on the IPDS side.
- Line 5: IMM LI_BEG is processed. One *medium-map-level-collection-of-sheets* and one *medium-map-level-each-sheet* MFCs are found with one triplet 0x8E each. As a result, two new DGB levels are defined on the IPDS side (0x80 and 0x70). AFO could be used instead of level 0x70.
- Line 6: an LCC is used to indicate that the following sheet(s) will use a Form from the standard media source (input bin). The Form will be printed simplex, 1-UP.
- Line 7: DGB level 0x70 (corresponding to *medium-map-level-each-sheet* operation) is terminated at the end of the sheet. Triplets 0x8E are not required but also not forbidden in a DGB terminate command. They are included here for documentation purposes.
- Line 8: IMM LI_BEGI is processed. One *medium-map-level-collection-of-sheets* MFC is found. As a result, level 0x80 (offset to left) is continued. One MMC is also found.
- Lines 9 and 10: an LCC is used to indicate that the following sheet(s) will use a Form from the Inserter bin. The UP³I "Form Finishing Operation triplet, Interpose Form from bin XX" triplet is **not used** in MO:DCA/IPDS environments. A "place holder" page (which is also part of the input data stream) is "printed".
- Line 11: page group SF's do not necessarily influence the DGB level on the IPDS side.
- Line 12: IMM LSRF_BEG is processed. Two *medium-map-level-collection-of-sheets* (with one triplet 0x8E each)) and one *medium-map-level-each-sheet* (with two triplets 0x8E) MFCs are found. As a result, level 0x80 (offset to left) is continued and two new DGB levels are defined on the IPDS side: level 0x70 with one 0x8E triplet (edge stitch) and level 0x60 with two (rotate/fold).
- Line 13: an LCC is used to indicate that the following sheet(s) will use a Form from the **wideForm** media source (input bin). The Form will be printed simplex, 1-UP.
- Line 14: DGB level 0x60 (corresponding to *medium-map-level-each-sheet* operation) is terminated at the end of the sheet. Level 0x70 (edge stitching) is left open.

Results



- Line 15: IMM LS_CON is processed. One *medium-map-level-collection-of-sheets* MFC (with two triplets 0x8E) is found. The two currently active *medium-map-level-collection-of-sheets* finishing operations are continued. No changes are required on the IPDS side.
- Line 16: an LCC is used to indicate that the following sheet(s) will use a Form from the standard media source (input bin). The Form will be printed simplex, 1-UP.
- Line 18 and 19: page group SF's do not necessarily influence the DGB level on the IPDS side.
- Line 20: IMM LC_BEG is processed. Two *medium-map-level-collection-of-sheets* MFCs (with one triplet 0x8E each)) are found. As a result, level 0x70 (edge stitch) is terminated and a new level 0x70 (corner staple) is initiated. All higher levels (0x80 offset to left and 0x90 shrink wrap) are left unchanged (continued).
- Lines 21 to 25: an LCC is used to indicate that the following sheet(s) will use a Form from the standard media source (input bin). The Form will be printed **duplex**, 1-UP. 9 pages (5 sheets) are printed.
- Lines 26 and 27: page group SF's do not necessarily influence the DGB level on the IPDS side.
- Line 28: IMM LC_BEG is processed. Two *medium-map-level-collection-of-sheets* MFCs (with one triplet 0x8E each)) are found (same as for line 17). As a result, level 0x70 (corner staple) is terminated and a new level 0x70 (corner staple) is initiated. All higher levels (0x80 offset to left and 0x90 shrink wrap) are left unchanged (continued).
- Line 34, 35 and 36: page group SF's do not necessarily influence the DGB level on the IPDS side.
- Line 37: IMM RI_BEG is processed. One *medium-map-level-collection-of-sheets* and one *medium-map-level-each-sheet* MFCs are found with one triplet 0x8E each. As a result, level 0x70 (corner staple) and level 0x80 (offset to left) are terminated. Two new levels (0x80 offset to right and 0x70 perforation cut) are initiated. Level 0x90 (shrink wrap) is left unchanged (continued).
- Line 38: an LCC is used to indicate that the following sheet(s) will use a Form from the standard media source (input bin). The Form will be printed simplex, 1-UP.
- Line 39: DGB level 0x70 (corresponding to *medium-map-level-each-sheet* operation) is terminated at the end of the sheet.
- Line 40: IMM RI_BEGI is processed. One *medium-map-level-collection-of-sheets* MFC is found. As a result, level 0x80 (offset to right) is continued. One MMC is also found.
- Lines 41 and 42: an LCC is used to indicate that the following sheet(s) will use a Form from the Inserter bin. The UP³I "Form Finishing Operation triplet, Interpose Form from bin XX" triplet is **not used** in MO:DCA/IPDS environments. A "place holder" page (which is also part of the input data stream) is "printed".
- Line 43: page group SF's do not necessarily influence the DGB level on the IPDS side.
- Line 44: IMM RSRF_BEG is processed. Two *medium-map-level-collection-of-sheets* and two *medium-map-level-each-sheet* (with one triplet 0x8E each) MFCs are found. As a result, level 0x80 (offset to right) is continued and three new DGB levels are defined on the IPDS side: level 0x70 (edge stitch), level 0x60 (rotate) and level 0x50 (fold). Level 0x90 (shrink wrap) is left unchanged (continued).
- Line 45: an LCC is used to indicate that the following sheet(s) will use a Form from the **wideForm** media source (input bin). The Form will be printed simplex, 1-UP.
- Line 46: DGB levels 0x50 and 0x60 (corresponding to *medium-map-level-each-sheet* operations) are terminated at the end of the sheet. All higher levels (0x70 edge stitch, 0x80 offset to right and 0x90 shrink wrap) are left unchanged (continued).

- Line 47: IMM RS_CON is processed. The two currently active *medium-map-level-collection-of-sheets* finishing operations are continued. No changes required on the IPDS side.
- Line 52 and 53: page group SF's do not necessarily influence the DGB level on the IPDS side.
- Line 54: IMM RC_BEG is processed. Two *medium-map-level-collection-of-sheets* MFCs (with one triplet 0x8E each)) are found. As a result, level 0x70 (edge stitch) is terminated and a new level 0x70 (corner staple) is initiated. All higher levels (0x80 offset to right and 0x90 shrink wrap) are left unchanged (continued).
- Lines 55 to 59: an LCC is used to indicate that the following sheet(s) will use a Form from the standard media source (input bin). The Form will be printed **duplex**, 1-UP. 3 pages (2 sheets) are printed.
- Line 60 and 61: page group SF's do not necessarily influence the DGB level on the IPDS side.
- Line 62: EDT is processed. The currently active *document-level* finishing is terminated. All nested operations are also terminated. As a result, levels 0x70, 0x80 and 0x90 are terminated on the IPDS side. Any active Finishing operation is always terminated at the end of the document.
- Line 63: the end-of-file is reached. As a result, the presentation software terminates DGB level 0xA0.

3.2 Extensions for PCL5/PJL (Printer Job Language)

There are extensions for PCL5 as well as for PJL.

3.2.1 Extensions in PJL

With the PJL extensions, it is possible for the host to request and get information about the connected UP³I devices (IFNO, USTATUS commands) and to control the finishing of complete jobs.

3.2.1.1 INFO Commands

3.2.1.1.1 Command UP3IDEVICE

Host requests information about all installed UP3I devices.

Request:
@PJL INFO UP3IDEVICE<LF>

Answer by the printer (example):

@PJL INFO UP3IDEVICE [3 TABLE]<CR><LF>
<HT>PAPER_SEQUENCE_ID<HT><HT>DEVICE_TYP<CR><LF>
<HT>1< HT >< HT >3<CR><LF>
<HT>32< HT >< HT >130<CR><LF>
<HT>33< HT >< HT >51<CR><LF>
<FF>

3.2.1.1.2 Command UP3ITUPEL

Host requests information about the existing tupel.

Request:
@PJL INFO UP3ITUPEL<LF>

Answer by the printer: (example):

@PJL INFO UP3ITUPEL<CR><LF>
TUPEL4<CR><LF>
ID=101<CR><LF>
<HT>1<CR><LF>
<HT>32<CR><LF>
<HT>48<CR><LF>
...
TUPEL5<CR><LF>
ID=102<CR><LF>
<HT>1<CR><LF>
<HT>20<CR><LF>
<HT>21<CR><LF>
<FF>

All number values are decimal.

3.2.1.1.3 Command UP3IPAPERINPUTMEDIA

Host requests information about the available paper input media.

Request:
@PJL INFO UP3IPAPERINPUTMEDIA<LF>

Answer by the printer (example):

@PJL INFO UP3IPAPERINPUTMEDIA<CR><LF>
INTRAY1<CR><LF>
<HT>NAME=String<CR><LF>
<HT>COLOR=Wert<CR><LF>
<HT>SETCOUNT=Wert<CR><LF>
<HT>MEDIARECYCLED=Wert<CR><LF>
<HT>MEDIATYPE=Wert<CR><LF>
<HT>WEIGHT=Wert<CR><LF>
<HT>PINHOLE=Wert<CR><LF>
<HT>WIDTH=Wert<CR><LF>
<HT>LENGTH=Wert<CR><LF>
INTRAY2<CR><LF>
<HT>NAME=String<CR><LF>

...

INTRAYn
<HT>NAME=String<CR><LF>
<HT>COLOR=Wert<CR><LF>
<HT>SETCOUNT=Wert<CR><LF>
<HT>MEDIARECYCLED=Wert<CR><LF>
<HT>MEDIATYPE=Wert<CR><LF>
<HT>WEIGHT=Wert<CR><LF>
<HT>PINHOLE=Wert<CR><LF>
<HT>WIDTH=Wert<CR><LF>
<HT>LENGTH=Wert<CR><LF>
<FF>

3.2.1.2 USTATUS Commands

With this commands, the host controls the use of the USTATUS commands by the printer.

3.2.1.2.1 Command USTATUS UP3I

Syntax:
@PJL USTATUS UP3I = ON | OFF<LF>
If the value is set to ON, the folllowing message is given to the host, if there are any changes in the UP³I enviroment:

@PJL USTATUS UP3I<CR><LF>
UP3IPAPERINPUTMEDIA | UP3IDEVICE | UP3ITUPEL<CR><LF>
<FF>

The host can get the new values with the relating INFO command.

3.2.1.3 Job related Control Commands

With this PJL commands, finishing for a complete job can be controlled without changing the PCL print data.

Syntax:
@PJL SET Variable = Value

The following values for Variable and Value are defined:

variable: *FINISH*
possible values: *OFF* | *STAPLE*
default: *OFF*

variable: *STAPLEOPTION*
possible values: *TOPLEFT* | *BOTTOMLEFT* | *EDGELEFT*
default: *TOPLEFT*

variable: *STAPLEPOSITION*
possible values: decimal value: 0 .. 500.0 in steps by 0.5. This is the distance in mm between the leading edge and the middle of the staple.
default: 0
STAPLEPOSITION is used only with the option *EDGELEFT*.

3.2.2 Extensions in PCL5

Set and Sheet based UP³I Commands

3.2.2.1 Syntax:

*<Esc> * u # W [# Bytes Data]*

is the number of data bytes.

Description:

Byte	Description	
0..1	Type of Command: 0x0001 – Begin of Set 0x0002 – End of Set 0x0003 – AFO (Apply Finishing Operation)	
2 ... n	Form Finishing Operating Triplet (see Form Exit Frame)	
	Byte	**Beschreibung**
	0 .. 1	Length of Triplet
	2	0x03
	3	0x00
	4	Finishing Operation type
	5 .. 6	Finishing Operation parameter
	7	Finishing Operation reference corner/edge
	8	Finishing Operation count
	9 .. 12	Finishing Operation axis offset (relative to reference edge)
	The following bytes are zero or more 4 Byte values (number of bytes described in byte 8):	
	13 .. 16	Operation position on finishing axis

Byte 2 ... n are not necessary and not allowed if Type of Command is 0x0002 – End of Set.

3.2.2.2 Handling

This Escape Sequence is allowed at any place within the PCL print data.
The following rules are applied:

- Each command ***<Esc> * u # W*** closes the current page and performs am eject to front facing sheet. To avoid unpredictable ractions, it is recommended to place these commands outside of a page, e.g. BeginOfSet before the first sheet of the set, EndOfSet after the last page of a set.
- Nesting of Sets is allowed.
- Invalid commands (content of Byte 0..1 is unknown, invalid finishing operation type, EndOfSet without BeginOfSet) are ignored.
- Invalid Datalength may lead to unpredictable results.

3.2.2.3 Example

Stitch 3 sheets (6 pages duplex)

<Esc> E <Esc>*u19W [00 01 00 11 03 00 04 00 04 00 01 00 00 00 00 00 01 0F 48] ...
6 Pages PCL duplex ... <Esc>*u2W [00 02] <Esc> E

Begin of Set
00 01 : Byte 0 .. 1 : Begin of Set
00 11 : Byte 0 .. 1 : Length of the triplet
03 00 : Byte 2 .. 3 : Stitch
04 : Byte 4 : Finishing Operation Type (Staple/Stich)
00 04 : Byte 5 .. 6 : Finishing Operation Parameter (Edge Stich)
00 : Byte 7 : Operation reference corner/edge
01 : Byte 8 : Finishing Operation Count (Number of 4 Byte Values after byte 12)
00 .. 00 : Byte 9 .. 12 : Operation axis offset (default)
00 01 0F 48: Byte 13 .. 16: Operation position on finishing axis

End of Set
00 01 : Byte 0 .. 1 : End of Set

3.2.2.4 Interaction with PJL

A BeginOfSet, started in PJL is not closed by an EndOfSet in PCL. But a BeginOfSet in PCL is closed by an EndOfSet in PJL (PJL is the outer bracket).

If a PCL data stream is embedded into a PJL job, it is possible with the PJL Job Command to enable or disable the use of the UP3I escape sequences within PCL.

Syntax:
@PJL JOB NAME="NoFinishingForThisJob" OFFSET=FALSE ***UP3I=FALSE****<CR><LF>*

Option *UP3I=FALSE* – ignore UP³I Escape Sequences in PCL.
Option *UP3I=TRUE* – use UP³I Escape Sequences in PCL.

The default value is *TRUE*.

For more details about PCL5 and PJL see:

PCL5e for SRA Controller, U24721-J-Z247-2-7600 and
Océ PJL on SRA Controller, U24398-J-Z247-2-7600

Abbreviations:
<HT> Horizontal Tab
<Esc>
<CR> Carriage Return
<LF> Line Feed
<FF> Form Feed

4 Operating Interface

The Operating (Web / Control / Application / Logical) Interface enables:

- Remote Operation -> Operating from one place
- Automatic set-up –> Reduced Set-up times
- Standard „Look and feel“ for User / Customer -> Operating Panels with common GUI elements
- Workflow control (for tight and loose coupled devices)

4.1 Concept

The UP^{3}I printer, pre- and post-processing devices are connected to a “UP^{3}I Manager“ with the IEEE 1394 Interface using the **asynchronous transfer**.

The “UP^{3}I Manager“ is also connected to Server, Host or Remote Operating station. This connection may be vendor specific but a standard Ethernet LAN is recommended. Server, Remote Operation stations etc. have no direct access to Printer, pre- and post-processing devices.

A loose coupled post-processing device is directly attached to Server, Host or Remote Operating station (via Ethernet / LAN...). There is no protocol connection between UP³I Manager and loose coupled devices. Find more information for near-line processing at chapter 1.3.5 “Near line Processing with UP³I” (Page 17).

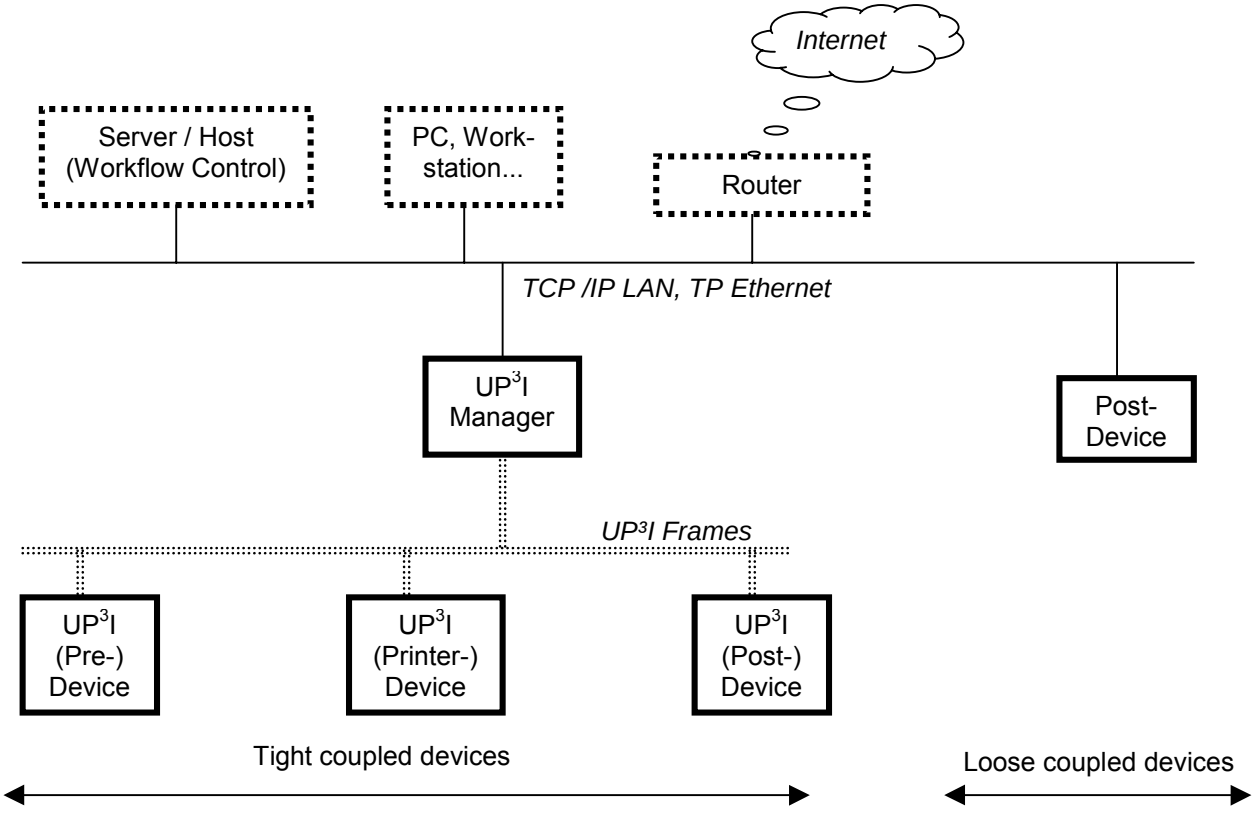


Figure 4-1: Topology with standalone UP^{3}I Manager.

4.1.1 Remote Operating

For remote operation the production line is reached by connecting to the LAN interface of the UP³I Manager. This is possible with a UP³I Manager frame programmed as a Java Application or a Java Applet (which e.g. is running within a standard Web Browser like Netscape Communicator, Microsoft Explorer, ...).

The operating panel of each device can be remotely controlled if a UP³I Manager is available. The UP³I device itself needs not to have an operating panel.

Keep in mind to fulfill international safety standards (like e.g. UL, CSA, CE...) requirements.

Each UP³I device presents its operating panel by writing a Java Class Device GUI Component. This Device GUI may be installed on the UP³I Manager by an appropriate method (e.g. Floppy / CD-ROM...) or be automatically transferred from the UP³I device to the UP³I Manager using the UP³I frames.

The UP³I Manager's web server combines the UP³I devices of one production line.

4.1.2 Automatic Set-up

Save & Load complete set-up of all devices in the process line. If no „Automatic Set-up" functionality is supplied the specific benefits will be missed, but the UP³I is still operable.

The UP³I Manager offers an extra menu page called „Set-up". Here it is possible to save the current values of all tight coupled UP³I devices to a file (with a free selectable name) on the UP³I Manager environment. Vice versa it is possible to load this parameter file again and send the changed parameters to the specific UP³I device.

The set-up parameters are requested and set with Device GUI SetupInterface.java (method setSetupData and getSetupData).

If one UP³I device is not able to set-up all parameters electrically (e.g. some knobs have to be moved mechanically) it changes to STOP state and gives on the operator panel some information about what to do (help screen).

4.1.3 Workflow control (PRISMAaudit, Infoprint Workflow...)

For workflow control systems the UP³I Manager is the logical communication partner. The UP³I Manager is responsible to fetch and deliver the requested information. Due to different workflow control implementations (Infoprint Workflow, PRISMA audit...) there may be different UP³I Manager implementations necessary.

The server communicates with the UP³I Manager using SNMP.

4.1.4 Communication Method

The operating interface communication method (for communication between UP³I Manager and printer, pre- and post-processing devices) is mirrored from the Simple Network Management Protocol (SNMP). This method is specified by the „Network Working Group" in RFC 1157.

The RFC documents are available e.g. at (www.rfc-editor.org).

The relevant SNMP Management Information Bases (MIBs) are defined by IETF „The Internet Engineering Task Force“ (http://www.ietf.org). The UP³I devices additional may have private MIBs. These private MIBs are only relevant for Device GUI operating and are not necessary be accessed from external workstations.

Printer MIB (RFC1759): R. Turner, 01/26/1999. (334475 bytes):
This document provides definitions of models and manageable objects for printing environments. The objects included in this MIB apply to physical, as well as logical entities within a printing device. This MIB definition makes explicit references to the Host Resources MIB (RFC 1514), as well as the Interfaces Group of MIB-II (RFC 1213).

Printer Finishing MIB: H. Lewis, R. Bergman, 02/19/1999. (98802 bytes):
This document defines a printer industry standard SNMP MIB for the management of printer finishing device subunits. The finishing device subunits applicable to this MIB are an integral part of the Printer System. This MIB does not apply to a Finisher Device that is not connected to a Printer System. The Finisher MIB is defined as an extension of the Printer MIB [PrtMIB] and it is expected that the information defined in this document will be incorporated into a future update of the Printer MIB

Only the UP³I Manager is capable to get and set the SNMP parameters. For remote operating the remote operating station has to connect only to the UP³I Manager.

There is no direct SNMP communication between UP³I processing devices. The traffic is managed by the UP³I Manager.

The Host or Remote Operating station have no direct access to the Printer and the Pre- and Post-processing devices.

At the UP³I Manager Ethernet side, remote operation of all UP³I devices is possible. In conjunction with the Server (Host) the UP³I Manager is responsible for sending the correct set-up data to each device within the production line. In case of an error at the production line a workflow control system is able to initiate a reprint (which may be a page, a set or a job...).

The UP³I Manager delivers no (IPDS...) Print Data.

Data Security

The UP³I Manager prevents any external access to the Printer and the Pre- and Post-processing devices. The internet and intranet access to the UP³I Manager has to be restricted and organized by the customer (-> using routers, firewalls...). No additional security (SSL, ...) for the UP³I communication is necessary or available.

To prevent an inconsistent setup, either
- only one operating panel is used simultaneously., or
- the UP³I Manager regulates the access with an appropriate way.

4.2 UP³I Manager

The UP³I Manager works as a bridge between host / server, remote operating and printer, pre- and post-processing devices.
Exactly one UP³I Manager is existing in a production line and may be

- a Standalone PC / Workstation (e.g. Remote Operation Station),
- integrated in one device of the production line or
- integrated to the server / host

The UP³I Manager offers remote operation of printer, pre- and post processing devices and includes therefore a web (HTTP) server. For server / host access an SNMP Agent is available which supports Host Resource MIB, Printer MIB and Finishing MIB. External SNMP accesses to the Printer MIB and Finishing MIB are forwarded to the corresponding UP³I device whereas the Host Resource MIB is handled only within the UP³I Manager. Devices on the UP³I side are represented as devices in the Host Resource MIB, whereby instead of hrDeviceIndex the UP³I Paper Sequence ID is used. For process line management, the UP³I Manager communicates with UP³I frames derived from SNMP to each UP³I device.

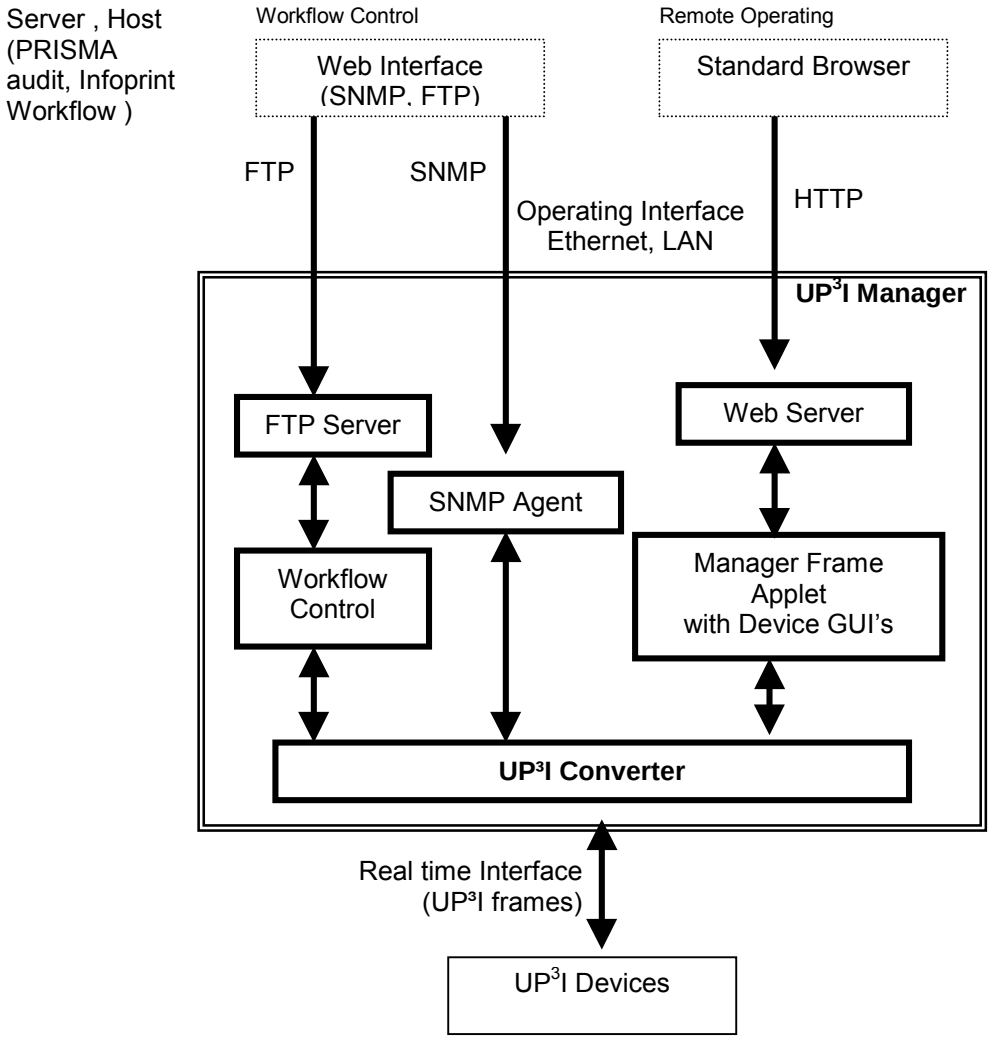


Figure 4-2: Functional Model UP³I Manager

4.3 UP³I Device

The tight coupled UP³I devices communicate with the UP³I Manager using only UP³I frames defined within this specification.

Only the UP³I Manager is capable to get and set the “original” SNMP parameters. For remote operation the remote operating station has to connect only to the UP³I Manager. The UP³I Devices need to support the Printer MIB respectively Finishing MIB in the sense of SNMP.

There is no direct SNMP communication between UP³I processing devices. The real time communication between UP³I devices is not affected.

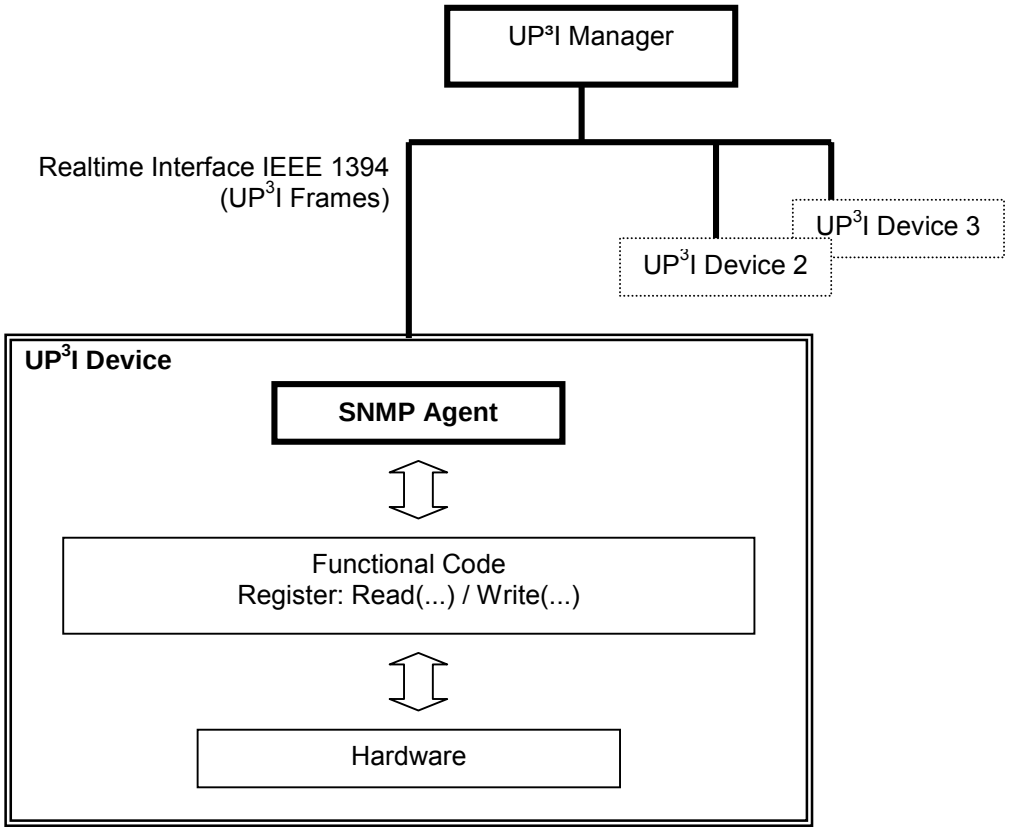


Figure 4-3: Functional Model UP³I Processing Device

4.4 Functionality

4.4.1 Fundamentals

- The UP³I User Interface is implemented in the JAVA programming language (version JDK 1.4 or greater) and maybe operated (at least) via keyboard and mouse – touch screen operating is possible.
- The UP³I Manager provides a UP³I Manager Frame.
- UP³I Device specific GUI's will be delivered by their respective manufacturers.
- The UP³I Device GUI's are started and displayed by the UP³I Manager Frame and notified for which devices they are responsible.
- The UP³I Manager Frame present the UP³I Device GUI's in its frame or in extra frames. The menu navigation may be done e.g. using selection trees, tabbed panes etc.
- The UP³I Manager-Frame is responsible for basic tasks such as menu selection, user administration, etc...
- The UP³I Manager-Frame provides methods to enable the device GUI's to communicate with their respective devices. The specification basis for this communication is SNMP.

4.4.2 GUI ⇔UP³I Manager Communication

- The GUI maybe executed either as an applet or as a Java application depending on the UP³I Manager.
- A GUI is identified to the UP³I Manager by registering the JAR file as associated with a particular manufacturer and device type.
- The UP³I Manager starts the required GUI's in a JAVA environment. This environment, the UP³I Manager, provides facilities for navigating the various device menus and for user administration.
- The UP³I Paper Sequence ID of a device is provided as a parameter to each GUI. Multiple devices of the same type are permitted in a single chain.
- Any number of operator consoles may be used in parallel.
- Mutual exclusion of consoles requires further clarification.
- Remote updates between multiple operator consoles is supported. All GUIs are notified of changes at a single station.
- Hierarchical Error display. In the case of a device error, the error menu for that device should be activated. This allows the error to be more precisely detailed. ***-> needs to be defined***

4.4.3 GUI Properties

- Each GUI belongs to a particular physical device, but may serve several different logical devices (Clustering).
- Each GUI has at least one menu page, any number may be used and a hierarchical structure for the menu pages is possible.
- A fixed Panel Header may be used (a menu header that is used for all menus of a single device).
- Each menu page is constructed as a JPanel.
- A menu page specifically for error messages should be provided.
- It should be possible using the user administration to set all options to read only (grayed)
- JAVA Help is the target help system.
- All text (including help text) should be Unicode text.

- Internationalization should employ JAVA Resource Bundles.
- Each manufacturer should deliver text in at least German French, English and Spanish.
- Additionally a manufacturer may deliver a resource file with language information which maybe universally translated.
- The GUI „look and feel“ should be capable of being controlled externally (from UP³I Manager) so that it fits with the specific implementation of the UP³I Manager (Customization). This should include: design of the SWING elements (buttons, checkboxes,...), fore and background colors (possibly defined in the resource file). This information should not be hard coded in the GUI.
- The use of a layout manager is implied as the sizes of the basic elements may change. A pre-defined fixed layout can thus not work.
- The Device GUI runs in the Manager Frame with an own security manager within a java sandbox. Therefore access to local resources like a hard disk is impossible.

4.4.4 GUI ⇔ UP³I device Communication

- The communication between GUI and device is defined using SNMP.
- The UP³I Manager has a „logical“ SNMP binding to its Manager-Frame (As well as the usual UDP, SNMP frames might also be passed using wrapper protocols such as Java RMI or CORBA).
- SNMP Telegrams are sent as frames from the UP³I Manager and are answered by UP³I-devices.
- Physical devices may be represented at both the UP³I and SNMP levels as multiple devices.
- It is possible to define custom MIBs e.g. for holding the menu parameter information. These MIBs are realized by a SNMP agent. This agent is only accessible from the device GUI.
- The GUI’s communicate via set-/get-/getnext-calls with the SNMP agents, associated with their respective devices. These Methods are provided by the UP³I Manager. A set() call in the device GUI results in a Information Management-Frame being sent to the associated device.
- Devices may notify all GUIs of changes using the SNMP Trap mechanism. A TrapListener Interface is provided by the UP³I Manager Frame.
- Polling for values via a GUI is to be avoided.

4.4.5 General

- Remote Firmware Upgrade is possible using the UP³I file transfer frame.
- It should be possible for UP³I devices with sufficient memory to upload to the UP³I Manager the necessary GUI classes and resources using UP³I frames.

4.4.6 Diagram of UP³I Device GUI Interface

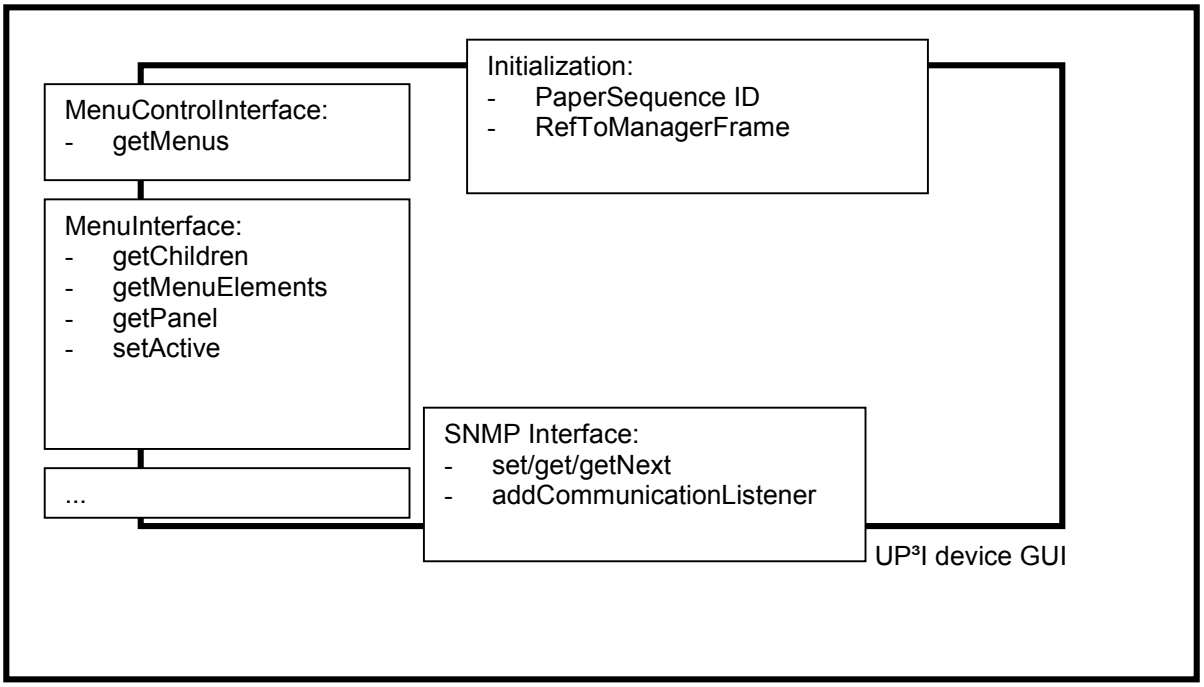


Figure 4-4: Diagram of the UP³I Device GUI Interface

4.4.7 Device GUI Clustering

A Device GUI Cluster is the presentation of several UP³I devices on one GUI. For each UP³I device, there is initDeviceGUI() call by the UP³I Manager frame. The following requirements need to be fulfilled:
Vendor ID, Vendor Device Type Name, the unique identifier (serial number) and the Device GUI Init Class Name must be the same.
It is recommended, that only one of the clustered devices reports device GUI jar files.
If different GUIs should be used, at least one of the following items must be unique: Vendor ID, Vendor Device Type Name, unique identifier (serial number).

4.4.8 Dealing with User Variables

Via the ProfileValuesInterface every device GUI can set and read user defined data. The UP³I Manager associates these data with one or more of the following device characteristics, described in the Self Defining Field frame:

- Vendor ID String (UP³I Product ID Triplet)
- Vendor Device Type Name String (UP³I Product ID Triplet)
- Unique Identifier / Serial Number (UP³I Product ID Triplet)
- Class Name (Device BUI Init Class Name Extended Triplet)

Changes of these device characteristics (e.g. by a new software release) may lead to a new data set for that device, the old data set may not be longer accessible.

4.4.9 The Manager Frame and Device GUI programming interface

- There are several interfaces necessary.
- The interfaces are hierarchically ordered.
- Find the corresponding listings in the appendix of this document and the file itself on the UP³I web site (www.up3i.org).

4.4.9.1 Interfaces provided by the UP³I Manager Frame

The top level interface is specified within the file:

- ManagerFrameInterface.java — Collection of all manager frame interfaces.

The subordinate interfaces are specified within the file:

- HelpInterface.java — used to integrate the help of a Device GUI into the help of the whole system.
- MenuControllerInterface.java — allows a device to communicate with the menu controller (navigator).
- ProfileValuesInterface.java — to set/get user values in the user management of the manager frame.
- CommunicationInterface.java — with this interface the device GUI's can communicate with their corresponding devices in an SNMP based form. With the "same" device GUI's in other instances of manager frames.

  Additional there are some helper classes defined.

  - AsnValueInterface.java
  - OidInterface.java
  - VarBindInterface.java
  - CommunicationListener.java
  - CommunicationEventInterface.java
  - CommunicationSetEventInterface.java
  - CommunicationTrapEventInterface.java
  - ConnectionException.java
  - SnmpException.java
  -

Trace will be done with the Java 1.4 logging, see package java.util.logging.

4.4.9.2 Interfaces implemented by the UP³I Device GUI

The top level interfaces are specified within the file:

- InitDeviceGUIInterface.java used to initialize a device GUI
- DeviceGUIInterface.java collection of all device GUI interfaces

The subordinate interfaces are specified within the file:

- DevicePropertiesInterface.java get menus and other information from the regarding UP³I device
- MenuElementInterface.java to allow the user management to disable the modification of some menu elements or menu groups
- MenuInterface.java interface of a GUI menu
- MenuTreeNodeInterface.java describes the menu tree of all menus of a device GUI
- PropertiesChangedInterface.java over this interface the manager frame informs the device GUI about global changes
- SetupInterface.java storing and loading device setups
- VersionInterface.java get the version of the device GUI
- DeviceGUIAccessInterface.java the UP³I Manager frame gives access rights to the particular GUI

4.4.10 GUI Style guide

The GUI style guide is open to each vendor and doesn't belong to this specification!

5 Appendix

5.1 Literature

Following books / publications where used as helpful sources:

- Interface Specification „Proposal for an Electrical Interface between Pre- and Post-Processing modules and Printing Systems“ Version 0.3, June 17th 1999, Robert Glur, Hunkeler
- Specification of the CIP3 Print Production Format, Version 3.0, Fraunhofer Institute for Computer Graphics
- Océ Specifications Type 1, Type 1+, Type 2
- Document feeding and Finishing, Generic Description, Xerox
- IBM 3835 MICR Printing Interface, RPQ 8B4400, GA32-0149
- IBM Data Stream and Object Architectures, MO:DCA, SC31-6802-06 or later
- IBM Data Stream and Object Architectures, IPDS, S544-3417-06 or later
- IBM Bar Code Object Content Architecture Reference, BCOCA, S544-3766 05 or later.
- Standard for a High Performance Serial Bus Peer-to-Peer Data Transport Protocol P1394.3
- A Simple Network Management Protocol (SNMP), Network Working Group, RFC 1157
- Printer MIB, Network Working Group, RFC 1759
- Printer Finishing MIB, Internet Draft
- JDF Specification, Release 1.0, CIP4 Organization
- IPDS Finishing Operations Including UP³I Enhancements, IBM Corp. David Stone, 18.Oct.2001
- UP³I Finishing Conduit (ACR#108 to MO:DCA Reference SC31-6802-05), IBM Corp Reinhard Hohensee, 24.Oct.2001
- PCL5e for SRA Controller, U24721-J-Z247-2-7600
- Océ PJL on SRA Controller, U24398-J-Z247-2-7600

You can get the IBM manuals at http://www.printers.IBM.com/R5PSC.NSF/Web/Archm

5.2 Definition of terms

Some terms used within this document are predefined here:

Abbrev	Long Name	Meaning
	ASCII String	String of 8 bit wide characters terminated with an Zero (‘\0’). ISO-646 IRV:1991 (US ASCII) is used.
	Asynchronous packet	Frame on IEEE1394; defined by IEEE1394 protocol.
CIP4	Cooperation for the Integration of Processes in Prepress, Press and Post Press	find relevant information e.g. at http://www.cip4.org
	CODE39	A bar code symbology characterized by a variable-length, bi-directional, discrete, self-checking, alphanumeric code. Three of the nine elements are wide and six are narrow. It is the standard for LOGMARS (the Department of Defense) and the AIAG.
	Cut Sheet	Unconnected sheets. Contrast with continuous-form media.
	Continuous Form	Connected sheets. An example of connected sheets is sheets of paper connected by a perforated tear strip. Contrast with cut-sheet media.
	Device State	A UP^{3}I device may be in one of the three following UP³I device States: • NOT READY (device stopped). The printer must not start paper movement, if one connected device in the current paper path is in Not Ready state. • READY (device is ready to work)
DFA	Document Finishing Architecture	defined by Xerox
DFD	Document Finishing Device	Post-processing Device
	Eject	Moving paper through devices without printing. Typically during paper insert cycle. During 'Ready State' or 'Data Ready' Eject is not possible.
EFD	External Feeding Device	Pre-processing Device
	Fan fold	other word for ⇨ continuous form
	Form	For a UP^{3}I device a physical sheet of paper is named „form“. On each form there may be several (logical) pages printed (see chapter 2.2.2).
	IEEE	Institute of Electrical and Electronics Engineers.
	Infoprint Workflow	IBM workflow manager

	In-line	In-line devices are connected to the printer physically and connected by communication signals. In-line is synonymous with tight coupled in this document.
IPDS	Intelligent Printer data Stream	Page description language; defined by IBM
	Isochronous data packet	Real-time data packet on IEEE1394; defined by IEEE1394 protocol.
JDF	Job Definition Format	Find definition at ⇒ CIP4
JMF	Job Messaging Format	Find definition at ⇒ CIP4
	Job	Within this document a job consists of ⇨sets and pages (forms) which are belonging together.
MIB	Management Information Base	the map of the hierarchical order of all managed objects and how they are accessed (via SNMP).
	Millipoint	Physical amount of length (1 Millipoint = 1/72000 inch).
NPRO	Non-Process Run Out	An operation that moves sheets of physical media through the printer without printing on them. This operation is used to stack the last printed sheet.
	N-up	The partitioning of a side of a sheet into a fixed number of equal size partitions. For example, 4-up divides each side of a sheet into four equal partitions.
OPC	Obtain Printer Characteristics	The OPC command causes a set of self-identifying fields, that describe characteristics of the printer and connected UP³I devices to be placed in the special data area of the acknowledge reply to the printers host (IPDS only).
PAC	Propose Accept Confirm	An UP³I enhancement for cut sheet lines to improve performance and reliability.
PDF ID	Print Data Format ID	With the Print Data Format Id a post processing printer reports, what kind of print data it supports.
PDU	Protocol Data Unit	A ⇨SNMP message consists of a version identifier, an SNMP community name, and a protocol data unit (PDU).
	PRISMA audit	Océ Workflow manager
PTL	Pin and Tractor Less	Pinless continuous form post-processing devices need a printed PTL mark at the top of each form for drift detection and compensation.
	Quadlets	Within IEEE1394 4 Bytes are named a “quadlet”
RFC	Request for Comment	Request for Comments are internet standard documents. The documents are available e.g. at (www.rfc-editor.org).

	Set	For a UP³I device a “Set” consists of one ore more (logical) pages. All pages within a Set apply to a common finishing operation. A group of sets build a ⇨job.
SNMP	Simple Network Management Protocol	SNMP is one of many protocols that are collectively known as the Internet Protocol (IP).
	Triplet	A three-part self-defining variable-length parameter consisting of a length byte, an identifier byte, and one or more parameter-value bytes.
	Tupel	Each possible paper path combination in the process line is described by a UP³I Tupel (which is the ordered chain of paper sequence ID’s). Sometimes it is useful to differ in active and inactive tupels. An active tupel is a paper path, that is able to receive and to process sheets. An inactive tupel is a paper path, that is available and addressable, but cannot receive and process sheets without changing all contained devices into ready state.
	Type I	simple finishing interface, signal based
	Type II	simple finishing interface, bus based

5.3 Comparison of signals terminology of former post processing interfaces

5.3.1 Command signals (Printer to DFD, or Printer to EFD)

Xerox DFA	**Océ Type 1**	**Function**
C0 – Sheet Exit	PAPEXIT	Sheet is exiting printer into DFD
C1 – End of Set	No equivalent	Sheet is the last of a set
	JOBCONTROL	Sheet is the last or first of a set (note: in both cases, the signal is generated from the Alternate Offset Stacker command)
C2 – Cycle Up	BLOWERON	The printer instructs the DFD to start
C3 – End of Job	No equivalent	Sheet is last sheet of last set of job
C4 – EFD Feed Sheet	No equivalent	Printer requires sheet from EFD
C5 – EFD Cycle Up	No equivalent	Printer instructs the EFD to start
C6 – Invoke DFD Function 1	No equivalent	DFD should commence designated function 1
C7 – Invoke DFD Function 2	No equivalent	DFD should commence designated function 2
No equivalent	1/6ADIT	One pulse per 1/6” of paper movement from printer
No equivalent	ADJUST PAPER SPEED	
No equivalent	SELF DEFINING FIELD	
No equivalent	PRINT DATA	
No equivalent	FREE PAGE INFORMATION	
No equivalent	EJECT	

5.3.2 Status signals (DFD to printer, or EFD to printer)

Xerox DFA	**Océ Type 1**	**Function**
S0 – Offline	ADEVCONN	The DFD is not available
S1 – Faulted		The DFD is not ready for use (causes hard stop)
	ADEVRDY-N	The DFD is ready for use (loss causes hard stop)
S2 – Full	ADEVSTOP	The DFD has, or will, shortly reach a capacity (causes the printer to stop feeding sheets / soft stop)
S3 - Sheet Delivered	PAPACKN-H	The DFD acknowledges the sheet has reached its destination (paper acknowledge)
S4 - Set Delivered	No equivalent	The DFD acknowledges the set has reached its destination
S5 - EFD bulk input medium low		The input medium in the EFD is getting low
S6 - EFD Not Ready		EFD is not available for use
S7 - EFD Sheet Fed		EFD is delivering a sheet to the printer
No equivalent	SERSEL	Instruction to printer that parallel or serial interface is selected
	NPRO	(Non Process Run Out) Printer will generate unprinted pages

5.4 Discussion on IEEE1394

IEEE1394...

- is a high-speed serial bus - also known as „FireWire“ (Apple) and „i.Link“ (Sony).
- is an interface standard being developed by an IEEE committee which describes a high speed serial bus that is designed for low cost.
- is broadly adopted by the consumer electronics industry (standard on Microsoft, Intel and Apple world) and growing importance in industrial environment.
- has the ability to mix real time (-> isochronous) data and asynchronous data on a single connection (which is necessary for the Type I, II, DFA backward compatibility).
- High Speed 100, 200, 400 (800...) Mbit/s data transfer rates
- Differential, serial data transmission
- Without repeaters up to 10m cable length possible. Up to 144m distance (with repeaters / extenders). A repeater function can be implemented by simply using the physical layer powered through the 1394 cable.
- Peer to Peer Bus, no central master

Safety:
It is recommended to Power off the corresponding UP³I devices during installation and cabling. Refer to the recommendations and the good working practices of the installation guides.

IEEE1394 Evolution:

Version	IEEE1394	**IEEE1394a**	IEEE1394b
	Compatible to 1394a	**Present / actual**	Future
Ratification	IEEE1394 /1995	**IEEE1394-2000**	IEEE1394-????
Transfer rate	Up to 400 Mbit/sec	**Up to 400 Mbit/sec**	3200 Mbit/sec
Physical Layer	Copper	**Copper**	Copper or Fiber
Distances	144 meter with repeater(s)	**144 meter with repeater(s)**	100 meter without repeater...
Power Management		**Advanced**	wake on IEEE1394 is supported

Available Suppliers:

This lists are included just to help you with starting your own market search. They are not considered to be complete!

Components:

- Agilent
- Fujitsu
- Lucent Technologies
- LSI Logic
- National Semiconductor
- NEC
- Oxford
- Philips
- SGS Thomson
- Sony
- Sun
- Symbios
- Texas Instruments
- VIA Technologies
- ~~Intel~~

Boards:

- Adaptec PCI Card (up to 200 Mbit/s , connection up to 144m...)
- Cyclone PMC Module (up to 200 Mbit/s)
- EKF CPCI Card (up to 400 Mbit/s)
- Densan Syst several CPCI Cards...
- SBS technology several PCI Cards...
- Sederta several (C)PCI Cards...
- Orange PCI Boards, Hubs, repeater, cables

Developer Tools:

Software Support

- Thesycon: universal 1394 driver for Windows 98 and 2000 (for user mode programming)
- Intoto: FireStack for embedded systems
- Microsoft native support for Windows98, 2000, (NT...)
- Sun native support
- Apple native support

Hardware Tools

- Data Transit 1394 Bus Analyzer (up to 400 Mbit/s)
- CATC FireInspector, Bus 6 protocol Analyzer (up to 400 Mbit/s)

5.5 Discussion on SNMP

SNMP

- is a standardized protocol to configure devices. It offers useful key / value pairs with common value types.
- belongs to the group of User Datagram Protocols (UDP) – It is defined with the „well-known“ IP port 161.
- has been developed for the transportation of control data. It is mainly used for the monitoring and analysis of networks.
- is **not real time** applicable.
- is available for many operating systems (-> every OS with Socket Support):
  - Windows 9x, NT
  - VxWorks
  - Unix, Solaris
  - Linux
  - .....

Each (Printer and Pre- and Post processing) device has a Management Information Base (MIB) e.g. see RFC 1156. A MIB is a specification of Data Objects. The realization of a MIB is called SNMP agent.

„The Printer MIB provides a method for network users to utilize SNMP and existing SNMP standards to manage networked printers. The MIB objects provide the ability to monitor and control these printers, providing fault, configuration and performance management.“ (http://www.pwg.org/mib/index.htm)

The SNMP „manager“ task is done by managing hosts (like PRISMA, Infoprint Workflow...).

A subset of essential parameters is available to each Pre- and Post-processing device and also to the printer.

Essential MIB Parameters:
- Machine information (Name, Type, Serial Number...)
- Device Status (On-line, Off-line, Ready / Not Ready)
- Counters
- Maintenance information

What is the Printer MIB?

The Printer MIB is the IETF proposed standard RFC1759. The protocol provided therein, provides a standard way for SNMP applications to identify the attributes and status of a printer.

What is the Finishing MIB?

The Finishing MIB is an extension of the Printer MIB, relating to finishing devices for details see also chapter 4.1.4., Communication Method

5.6 The Manager Frame and Device GUI programming interface listings

5.6.1 Manager Frame Interfaces

5.6.1.1 ManagerFrameInterface.java

```
package org.up3i.gui.managerframe;
import org.up3i.gui.managerframe.snmpcommunication.CommunicationInterface;

/**
 *  Collection of all manager frame interfaces.
 */
public interface ManagerFrameInterface
{
  /**
   *  Returns the menu controller interface.
   *
   * @return    the menu controller interface
   */
  public MenuControllerInterface getMenuControllerInterface();


  /**
   *  Returns the profile values interface.
   *
   * @return    the profile values interface
   */
  public ProfileValuesInterface getProfileValuesInterface();


  /**
   *  Returns the help interface.
   *
   * @return    the help interface
   */
  public HelpInterface getHelpInterface();


  /**
   *  Returns the communication interface.
   *
   * @return    the communication interface
   */
  public CommunicationInterface getCommunicationInterface();
}
```

Figure 5-1: ManagerFrameInterface.java

5.6.1.2 HelpInterface.java

```
package org.up3i.gui.managerframe;
import javax.help.HelpSet;

/**
 * Interface to integrate the help of a Device GUI into the help of the whole
 * system (print line).
 * Because Help Sets are normally Locale dependend, in case of Locale changes
 * the Device GUI has to remove the help set in the old Locale and to insert
 * a new Help Set in the new Locale.
 */
public interface HelpInterface
{
  /**
   *  The helpset 'hs' is integrated into the online help and shown there. If
   *  'hs' already exist 'addHelp is ignored.
   *
   * @param  hs  helpset which should be integrated into online help
   * @see        #removeHelpSet(javax.help.HelpSet hs)
   */
  public void addHelpSet(HelpSet hs);


  /**
   *  A helpset 'hs' is removed from the online help system.
   *
   * @param  hs  Helpset which should be removed.
   * @see        #addHelpSet(javax.help.HelpSet hs)
   */
  public void removeHelpSet(HelpSet hs);
}
```

Figure 5-2: HelpInterface.java

5.6.1.3 MenuControllerInterface.java

```
package org.up3i.gui.managerframe;

/**
 *  This interface allows a device to communicate with the menu controller
 *  (navigator).
 */
public interface MenuControllerInterface
{
  /**
   *  Informs the manager frame that a special menu should be shown in
   *  foreground. This can be used for shortcuts e.g. for buttons in the fixed
   *  panel header.
   *
   * @param  key  key for the menu which should come to foreground
   * @see         org.up3i.gui.devicegui.MenuInterface#getKey
   */
  public void showMenu(String key);


  /**
   *  The menu tree of the device has changed (some menus have been added or
   *  removed).
   */
  public void menuTreeChanged();
}
```

Figure 5-3: MenuControllerInterface.java

5.6.1.4 ProfileValuesInterface.java

```
package org.up3i.gui.managerframe;

/**
 *  Interface to get/set user values in the user management of the manager
 *  frame. Different users can have different values. This is thought for being
 *  used in cases like selected units for a length value or things like that.
 */
public interface ProfileValuesInterface
{
  /**
   *  Returns the string that was previously stored for that key (non volatile
   *  storage). If nothing was stored for the key the given default is returned.
   *
   * @param  key           unique key for that value (unique for one device
   *       gui).
   * @param  defaultValue  default String.
   * @return               the stored string for 'key' or 'defaultValue' if
   *       nothing was stored for that key.
   */
  public String getUserValue(String key, String defaultValue);


  /**
   *  Stores the given string in nonvolatile storage in the user management.
   *
   * @param  key    the key under this value should be stored (unique for one
   *       device gui).
   * @param  value  the value to store.
   */
  public void setUserValue(String key, String value);
}
```

Figure 5-4: ProfileValuesInterface.java

5.6.1.5 SNMP Communication Interfaces

5.6.1.5.1 CommunicationInterface.java

```
package org.up3i.gui.managerframe.snmpcommunication;

/**
 *  With this interface the devices guis can communicate with their
 *  corresponding devices.
 */
public interface CommunicationInterface
{
  /**
   *  The get method allows to retrieve values of known mib variables.
   *
   * @param  names                    object identifiers of the mib variables.
   * @return                          AsnValues of the mib variables.
   * @exception  SnmpException        SNMP error generated by the SNMP agent.
   * @exception  ConnectionException  Exception connecting the corresponding
   *       device.
   */
  public AsnValueInterface[] get(OidInterface names[]) throws ConnectionException,
SnmpException;


  /**
   *  The getNext method allows to retrieve the next mib object in the mib
   *  hierarchy with its value.
   *
   * @param  names                    object identifiers of the mib variables.
   * @return                          variable bindings ({@link OidInterface} -
   *       {@link AsnValueInterface} pairs) of mib objects being the "next" in
   *       the mib hierarchy.
   * @exception  SnmpException        SNMP error generated by the SNMP agent.
   * @exception  ConnectionException  Exception connecting the corresponding
   *       device.
   */
  public VarBindInterface[] getNext(OidInterface names[]) throws ConnectionException,
SnmpException;


  /**
   *  The set method allows to set values of known mib variables.
   *
   * @param  names                    object identifiers of the mib variables.
   * @param  values                   AsnValues for the mib variables
   * @exception  SnmpException        SNMP error generated by the SNMP agent.
   * @exception  ConnectionException  Exception connecting the corresponding
   *       device.
   */
  public void set(OidInterface names[], AsnValueInterface values[]) throws
ConnectionException, SnmpException;


  /**
   *  Gets the communication timeout value of the CommunicationInterface. If the
   *  functions get, getNext, set have not returned after this time, they will
   *  return with an ConnectionException.
   *
   * @return    The communication timeout value in milliseconds.
   */
  public int getCommunicationTimeout();


  /**
   *  Adds a CommunicationListener.
   *
   * @param  l  CommunicationListener.
```

```
    */
   public void addCommunicationListener(CommunicationListener l);


   /**
    *  Removes a CommunicationListener.
    *
    * @param  l  CommunicationListener.
    */
   public void removeCommunicationListener(CommunicationListener l);
}
```

Figure 5-5: CommunicationInterface.java

5.6.1.5.2 AsnValueInterface.java

```
package org.up3i.gui.managerframe.snmpcommunication;
import java.net.InetAddress;

/**
 *  This interface represents the ASN1 notation of SNMP variables with ASN1 type
 *  and value.<p>
 *
 *  Note: If ASN-Type and value class do not correspond an
 *  {@link IllegalArgumentException} should be thrown in constructor.<p>
 *
 *  Note: Java ints are signed 32-bit values, but counter, gauge, and timeticks
 *  are unsigned and may use the full 32-bits. To avoid signed/unsigned
 *  conversions, a Long is used. However if values greater that can be
 *  represented by a 32-bit unsigned number are supplied an
 *  {@link IllegalArgumentException} should be thrown in constructor.
 */
public interface AsnValueInterface
{
  /**
   *  ASN1-Type INTEGER, represents an integer value.<p>
   *
   *  ASN_INTEGER --> {@link Integer}
   */
  public final static byte ASN_INTEGER = 2;

  /**
   *  ASN1-Type OCTET STRING, represents a byte string. A Octet is a sequence of
   *  eight Bits.<p>
   *
   *  ASN_OCTSTR --> byte[]
   */
  public final static byte ASN_OCTSTR = 4;

  /**
   *  ASN1-Type NULL.<p>
   *
   *  ASN_NULL --> null
   */
  public final static byte ASN_NULL = 5;

  /**
   *  ASN1-Type OBJECT IDENTIFIER, represents a mib variable. Because the nodes
   *  of a mib tree are defined with numbers an object identifier is something
   *  like a integer array.<p>
   *
   *  ASN_OID --> {@link OidInterface}
   */
  public final static byte ASN_OID = 6;

  /**
   *  ASN1-Type IP Address, an OCTET STRING with four bytes.<p>
   *
   *  ASN_IPADDRESS --> {@link java.net.InetAddress}
   */
  public final static byte ASN_IPADDRESS = 64;

  /**
   *  ASN1-Type Counter, is a unsigned 32-Bit Integer.<p>
   *
   *  ASN_COUNTER --> {@link Long}, see note in interface description
   */
  public final static byte ASN_COUNTER = 65;

  /**
   *  ASN1-Type Gauge, is a unsigned 32-Bit Integer.<p>
   *
```

```
    *  ASN_GAUGE --> {@link Long}, see note in interface description
    */
   public final static byte ASN_GAUGE = 66;

   /**
    *  ASN1-Type TimeTicks, is a unsigned 32-Bit Integer.<p>
    *
    *  ASN_TIMETICKS --> {@link Long}, see note in interface description
    */
   public final static byte ASN_TIMETICKS = 67;

   /**
    *  ASN1-Type Opaque, a byte string.<p>
    *
    *  ASN_OPAQUE --> byte[]
    */
   public final static byte ASN_OPAQUE = 68;


   /**
    *  Returns the ASN1-Type.
    *
    * @return    ASN1-Type, one of the declared ASN_... constants above.
    */
   public byte getType();


   /**
    *  Returns the ASN1-Value.
    *
    * @return    Value. The class of the returned Object depends of the
    *       ASN1-Type. See description of the corresponding ASN1-Type above.
    */
   public Object getValue();
}
```

Figure 5-6: AsnValueInterface.java

5.6.1.5.3 OidInterface.java

```
package org.up3i.gui.managerframe.snmpcommunication;

/**
 *  The interface OID (object identifier) represents a mib variable.
 */
public interface OidInterface
{
  /**
   *  Returns the OID as an integer array.
   *
   * @return    OID as an integer array.
   */
  public int[] get();


  /**
   *  Returns the OID as a string.
   *
   * @return    OID as a string in the format ("1.2.3.4").
   */
  public String getOidString();
}
```

Figure 5-7: OidInterface.java

5.6.1.5.4 VarBindInterface.java

```
package org.up3i.gui.managerframe.snmpcommunication;

/**
 *  VarBindInterface combines AsnValueInterface and OidInterface.
 */
public interface VarBindInterface
{
  /**
   *  Returns the OID.
   *
   * @return    Object identifier.
   */
  public OidInterface getOid();


  /**
   *  Returns the AsnValue.
   *
   * @return    AsnValue.
   */
  public AsnValueInterface getValue();
}
```

Figure 5-8: VarBindInterface.java

5.6.1.5.5 CommunicationListener.java

```
package org.up3i.gui.managerframe.snmpcommunication;

/**
 *  Listener for receiving communication events.
 */
public interface CommunicationListener
{
  /**
   *  This method is called for each communication event.
   *
   * @param  ev  The communication event.
   */
  public void process(CommunicationEventInterface ev);
}
```

Figure 5-9: CommunicationListener.java

5.6.1.5.6 CommunicationEventInterface.java

```
package org.up3i.gui.managerframe.snmpcommunication;

/**
 *  Base class for communication events.
 */
public interface CommunicationEventInterface
{
  /**
   *  Returns the variable bindings ({@link OidInterface} - {@link
   *  AsnValueInterface} pairs) sended with this event.
   *
   * @return    VarBind list or null for none.
   */
  public VarBindInterface[] getVarBinds();
}
```

Figure 5-10: CommunicationEventInterface.java

5.6.1.5.7 CommunicationSetEventInterface.java

```
package org.up3i.gui.managerframe.snmpcommunication;

/**
 *  Communication event for each {@link CommunicationInterface#set(OidInterface[],
 *  AsnValueInterface[])} request from a Device GUI in a Manager frame instance
 *  which is sent to the same Device GUI in all remaining instances of Manager
 *  frames. {@link CommunicationEventInterface#getVarBinds()} must return the
 *  Varbinds from the set request which could be set.
 *  The SetEvent is not given to the initiating Device GUI.
 */
public interface CommunicationSetEventInterface extends CommunicationEventInterface
{
}
```

Figure 5-11: CommunicationSetEventInterface.java

5.6.1.5.8 CommunicationTrapEventInterface.java

```
package org.up3i.gui.managerframe.snmpcommunication;
import java.net.InetAddress;

/**
 *  Communication event which models an SNMP Trap.
 */
public interface CommunicationTrapEventInterface extends CommunicationEventInterface
{
  /**
   *  Cold start trap signal the network management station that the
   *  corresponding SNMP agent does a cold start.
   */
  public final static int SNMP_GENERICTRAP_COLDSTART = 0;

  /**
   *  Warm start trap signal the network management station that the
   *  corresponding SNMP agent does a reinitialization (warm start).
   */
  public final static int SNMP_GENERICTRAP_WARMSTART = 1;

  /**
   *  A interface connected to the SNMP agent has gone to "down" state and is
   *  now not reachable by the SNMP agent.
   */
  public final static int SNMP_GENERICTRAP_LINKDOWN = 2;

  /**
   *  A interface connected to the SNMP agent has gone to "up" state and is now
   *  reachable by the SNMP agent.
   */
  public final static int SNMP_GENERICTRAP_LINKUP = 3;

  /**
   *  An authentification failure trap indicates normally a wrong Community name
   *  in the SNMP telegramm.
   */
  public final static int SNMP_GENERICTRAP_AUTHFAILURE = 4;

  /**
   *  A EGP (Exterior Gateway Protocol) Neighbor Loss trap signal the newtork
   *  management station, that a EGP neighbor, who has been used for exchange of
   *  routing information, is now unrechable.
   */
  public final static int SNMP_GENERICTRAP_EGPNEIGHLOSS = 5;

  /**
   *  Enterprise Specific Trap. Beside the predifined Traps SNMP agnets can use
   *  own Traps which are the enterprise specific.
   */
  public final static int SNMP_GENERICTRAP_ENTERSPECIFIC = 6;


  /**
   *  Returns the IP address of the SNMP agent who has generated this trap.
   *
   * @return    IP address of the SNMP agent.
   */
  public InetAddress getAgentAddress();


  /**
   *  Returns the Enterprise OID, which describes the trap.
   *
   * @return    Enterprise MIB-OID.
   */
  public OidInterface getEnterpriseOid();
```

```
  /**
   *  Returns the generic trap type.
   *
   * @return    Generic-Trap-Type
   */
  public int getGenericTrapType();


  /**
   *  Returns the specific trap type.
   *
   * @return    Specific trap type.
   */
  public int getSpecificTrapType();


  /**
   *  Returns the time stamp generated by the SNMP agent sending this trap.
   *
   * @return    Time-Stamp
   */
  public long getTimeStamp();
}
```

Figure 5-12: CommunicationTrapEventInterface.java

5.6.1.5.9 ConnectionException.java

```
package org.up3i.gui.managerframe.snmpcommunication;
import java.io.IOException;

/**
 *  Connection exception.
 */
public class ConnectionException extends IOException
{
  private final static long serialVersionUID = -4910716123533965 82L;
  private final int reason;


  /**
   *  The data transfer has timed out.
   */
  public final static int CONNECTION_TIMEOUT = 1;


  /**
   *  Creates a ConnectionException object.
   *
   * @param  reason  The reason for the ConnectionException
   */
  public ConnectionException(int reason)
  {
    initCause(null);
    this.reason = reason;
  }


  /**
   *  Creates a ConnectionException object with an error string.
   *
   * @param  reason   The reason for the ConnectionException
   * @param  message  error string.
   */
  public ConnectionException(int reason, String message)
  {
    super(message);
    initCause(null);
    this.reason = reason;
  }


  /**
   *  Creates a ConnectionException object with an error string and the
   *  specified cause.
   *
   * @param  reason   The reason for the ConnectionException
   * @param  message  Description of the Parameter
   * @param  cause    the cause.
   */
  public ConnectionException(int reason, String message, Throwable cause)
  {
    super(message);
    initCause(cause);
    this.reason = reason;
  }


  /**
   *  Creates a ConnectionException object with the specified cause.
   *
   * @param  reason  The reason for the ConnectionException
   * @param  cause   the cause.
   */
```

```
  public ConnectionException(int reason, Throwable cause)
  {
    initCause(cause);
    this.reason = reason;
  }


  /**
   *  Gets the reason for the ConnectionException
   *
   * @return    The reason
   */
  public int getReason()
  {
    return reason;
  }
}
```

Figure 5-13: ConnectionException.java

5.6.1.5.10 SnmpException.java

```
package org.up3i.gui.managerframe.snmpcommunication;
import java.io.IOException;

/**
 *  Snmp exception.
 */
public class SnmpException extends IOException
{
  /**
   *  The PDU (Protocol Data Unit) is to big. The SNMP agent cannot include all
   *  data in one SNMP transfer.
   */
  public final static int SNMP_ERR_TOOBIG = 1;

  /**
   *  The requested mib variable is missing. The SNMP agent has found a unknown
   *  mib variable.
   */
  public final static int SNMP_ERR_NOSUCHNAME = 2;

  /**
   *  Bad value. During decoding the SNMP agent has found a value that is not
   *  allowed for the corresponding mib variable.
   */
  public final static int SNMP_ERR_BADVALUE = 3;

  /**
   *  Request to write to a read only mib variable.
   */
  public final static int SNMP_ERR_READONLY = 4;

  /**
   *  Generic error. All errors which do not fit in the sceme above are reported
   *  as generic errors.
   */
  public final static int SNMP_ERR_GEN = 5;

  private final static long serialVersionUID = 3230149239338799841L;
  private final int errorStatus;
  private final int errorIndex;


  /**
   *  Creates a SnmpException object with error status and error index.
   *
   * @param  errorStatus  error status.
   * @param  errorIndex   error index.
   */
  public SnmpException(int errorStatus, int errorIndex)
  {
    super();
    this.errorStatus = errorStatus;
    this.errorIndex = errorIndex;
  }


  /**
   *  Creates a SnmpException object with error message, error status and error
   *  index.
   *
   * @param  errorStatus  error status.
   * @param  errorIndex   error index.
   * @param  msg          error message.
   */
  public SnmpException(int errorStatus, int errorIndex, String msg)
  {
```

```
    super(msg);
    this.errorStatus = errorStatus;
    this.errorIndex = errorIndex;
  }


  /**
   *  Error status of an SNMP request, filled out by the SNMP agent.
   *
   * @return    error status.
   */
  public int getErrorStatus()
  {
    return errorStatus;
  }


  /**
   *  Error index of an SNMP request, filled out by the SNMP agent.
   *
   * @return    error index.
   */
  public int getErrorIndex()
  {
    return errorIndex;
  }
}
```

Figure 5-14: SnmpException.java

5.6.2 Device GUI Interfaces

5.6.2.1 InitDeviceGUIInterface.java

```
package org.up3i.gui.devicegui;
import org.up3i.gui.managerframe.ManagerFrameInterface;

/**
 * Initialize a device GUI<p>
 *
 * It is possible to have one device gui per paper sequence id or one device gui
 * with multiple paper sequence ids (Clustering).<p>
 *
 * <b>Note:</b> If clustering is needed than the following rules have to be
 * considered:
 * <ul>
 *   <li> The manager frame creates only one instance of the
 *   InitDeviceGUIInterface for the cluster device.</li>
 *   <li> The manager frame calls the method initDeviceGUI of the same
 *   InitDeviceGUIInterface instance for each paper sequence id. The method must
 *   return the same instance of DeviceGUIInterface.</li>
 *   <li> "Vendor ID String", "Vendor Device Type Name String", "Unique
 *   Identifier / Serial Number" must be equal in the "UP3I Product ID Triplet".
 *   </li>
 *   <li> The class path is created from the Triplets "Device GUI Jar File Name
 *   Triplet". The class path must be equal. </li>
 *   <li> The class names of the implementations of interface
 *   InitDeviceGUIInterface must be equal. This is given by the "Device GUI Init
 *   Class Name Triplet"</li>
 * </ul>
 */
public interface InitDeviceGUIInterface
{
  /**
   * Initializes a device.<p>
   *
   * If the manager frame calls this method more than once for the same instance
   * of this InitDeviceGUIInterface (see class comment for information about
   * that handling) with different paper sequence ids and this method returns
   * the same instance of this device gui than the manager frame handles the
   * same instances of the returned device guis as one device gui with multiple
   * paper sequence ids. This means that a device gui with multiple paper
   * sequence ids has among other things one menu tree and one fixed panel
   * header. For this reason the manager frame displays the device gui only one
   * time in its gui. Note that the device gui has nevertheless more instances
   * of ManagerFrameInterface corresponding to the paper sequence id with which
   * it can for example communicate with all its up3i devices.
   *
   * @param managerFrame     collection of all managerframe interfaces for this
   *      paper sequence id
   * @param paperSequenceId  the paper sequence id of the device
   * @return                 collection of all device interfaces
   */
  public DeviceGUIInterface initDeviceGUI(ManagerFrameInterface managerFrame,
      int paperSequenceId);


  /**
   * Tells that this paper sequence id should be removed from the device gui,
   * what means that the corresponding ManagerFrameInterface is invalid after
   * this call. The device gui must free any occupied resources for this paper
   * sequence id. If it has no paper sequence ids remaining it must free all
   * occupied resources.<p>
   *
   * After the manager frame has called this method, none method of the
   * interface ManagerFrameInterface or its subinterfaces may be called which
   * corresponds to the paper sequence id.
   *
```

```
    * @param paperSequenceId  the paper sequence id of the device
    */
   public void removePaperSequenceId(int paperSequenceId);
}
```

Figure 5-15: InitDeviceGUIInterface.java

5.6.2.2 DeviceGUIInterface.java

```
package org.up3i.gui.devicegui;

/**
 * Collection of all device gui interfaces.
 */
public interface DeviceGUIInterface
{
  /**
   * Returns the property interface.
   *
   * @return   the properties interface
   */
  public DevicePropertiesInterface getDevicePropertiesInterface();


  /**
   * Returns the property changed interface.
   *
   * @return   the properties changed interface
   */
  public PropertiesChangedInterface getPropertiesChangedInterface();


  /**
   * Returns the setup interface.
   *
   * @return   the setup interface
   */
  public SetupInterface getSetupInterface();


  /**
   * Returns the version interface.
   *
   * @return   the version interface
   */
  public VersionInterface getVersionInterface();


  /**
   * Returns the DeviceGUIAccessInterface
   *
   * @return   the DeviceGUIAccessInterface
   */
  public DeviceGUIAccessInterface getDeviceGUIAccessInterface();
}
```

Figure 5-16: DeviceGUIInterface.java

5.6.2.3 DevicePropertiesInterface.java

```
package org.up3i.gui.devicegui;
import java.util.Locale;

/**
 * Device properties
 */
public interface DevicePropertiesInterface
{
  /**
   * Warning menu
   */
  public final static int MENU_TYPE_WARNING = 1;
  /**
   * Error menu
   */
  public final static int MENU_TYPE_ERROR = 2;


  /**
   * Returns the menu tree of a device gui.
   *
   * @return   the menu tree
   */
  public MenuTreeNodeInterface getDeviceMenuTree();


  /**
   * Returns the fixed panel header (toolbar) of a device gui.
   *
   * @return   the fixed panel header or null for none
   */
  public MenuInterface getFixedPanelHeader();


  /**
   * Gets the key of the specified menu type. This is used for switching to the
   * given menu in some cases like errors of the corresponding device.
   *
   * @param menuType  One of the MENU_TYPE_... constants
   * @return          The menu key or null for none
   * @see             MenuInterface#getKey
   */
  public String getMenuKey(int menuType);


  /**
   * Returns all supported locales of this device gui.
   *
   * @return   all supported locales
   */
  public Locale[] getSupportedLocales();
}
```

Figure 5-17: DevicePropertiesInterface.java

5.6.2.4 MenuElementInterface.java

```
package org.up3i.gui.devicegui;

/**
 *  Interface to allow the user management to disable the modification of some
 *  menu elements or menu groups.
 */
public interface MenuElementInterface
{
  /**
   *  Returns the unique key for this menu element (unique for one device gui).
   *
   * @return    unique key for this menu element
   */
  public String getKey();


  /**
   *  Name of the menu element translated to the current language.
   *
   * @return    name of the menu element translated to the current language
   */
  public String toString();


  /**
   *  Allows the menu element to be writable (modifiable) or not. Default state
   *  is writable.
   *
   * @param  isWritable  true if the element could be modified from this user
   */
  public void allowWritable(boolean isWritable);
}
```

Figure 5-18: MenuElementInterface.java

5.6.2.5 MenuInterface.java

```
package org.up3i.gui.devicegui;
import javax.swing.JPanel;

/**
 *  Interface of a menu.
 */
public interface MenuInterface
{
  /**
   *  Returns a JPanel of the menu.
   *
   * @return    JPanel object of the menu or null for none
   */
  public JPanel getPanel();


  /**
   *  Returns a unique key of the menu (unique for one device gui).
   *
   * @return    unique key of the menu
   */
  public String getKey();


  /**
   *  Name of the menu translated to the current language.
   *
   * @return    name of the menu translated to the current language
   */
  public String toString();


  /**
   *  With this method the menu can decide if OK/Cancel-Buttons should be
   *  displayed or not.
   *
   * @return    true, if menu buttons should be displayed.
   */
  public boolean showButtons();


  /**
   *  Returns the modified state of the menu. Normally this is asked if the user
   *  selects another menu.
   *
   * @return    true if the menu is modified (one of the menus controls is
   *      modified).
   */
  public boolean isModified();


  /**
   *  Informs the menu about being active on top or being deactivated. This
   *  could be used for tasks which should work immediatly before or direct
   *  after a menu is visible.
   *
   * @param  activate  true if the menu will get visible.
   */
  public void setActive(boolean activate);


  /**
   *  The menu should store modified data into the correspondig device. After
   *  this method is called successfully {@link #isModified()} must return
   *  false.
   *
   * @return    true if the store operation was successful and the menu can be
```

```
    *      made invisible, false if there were problems and the menu should
    *      remain visible.
    */
   public boolean apply();


   /**
    *  The menu should reset all temporary changes, they are then lost. After
    *  this method is called {@link #isModified()} must return false.
    */
   public void reset();


   /**
    *  Returns all menu elements or menu groups of this menu, which writability
    *  should be controlled via user management.
    *
    * @return    menu elements which should be controlled from user management.
    *      if null is returned no elements could be controlled from the user
    *      management.
    */
   public MenuElementInterface[] getMenuElements();
}
```

Figure 5-19: MenuInterface.java

5.6.2.6 MenuTreeNodeInterface.java

```
package org.up3i.gui.devicegui;

/**
 *  Interface which describes the menu tree of all menus of a device gui.
 */
public interface MenuTreeNodeInterface
{
  /**
   *  Return the menu corresponding to this node.
   *
   * @return    the menu corresponding to this node.
   */
  public MenuInterface getMenu();


  /**
   *  Return the child nodes of this node.
   *
   * @return    the child nodes of this node or null if this node has no childs.
   */
  public MenuTreeNodeInterface[] getChildNodes();
}
```

Figure 5-20: MenuTreeNodeInterface.java

5.6.2.7 PropertiesChangedInterface.java

```
package org.up3i.gui.devicegui;
import java.util.Locale;

/**
 *  Over this interface the manager frame informs the device gui about global
 *  changes.
 */
public interface PropertiesChangedInterface
{
  /**
   *  Sets the locale for the device. After this all language dependend strings
   *  should be changed from the device and the active help set should be
   *  replaced by a new one in the current locale.
   *
   * @param  l  the new locale.
   */
  public void setLocale(Locale l);


  /**
   *  The user of the system has changed. That means that all user variables
   *  could be changed. After this all user variables should be reloaded.
   */
  public void userValuesChanged();
}
```

Figure 5-21: PropertiesChangedInterface.java

5.6.2.8 SetupInterface.java

```
package org.up3i.gui.devicegui;
import java.io.Serializable;

/**
 *  Interface for storing and loading setups.
 */
public interface SetupInterface
{
  /**
   *  Method to retrieve setup data from the device gui to the manager frame.
   *  There are two alternatives how this can work:<p>
   *
   *  Method 1: The devices collects its whole setup data and returns it as
   *  Serializable. The setup data is than stored on the manager frame.<p>
   *
   *  Method 2: The device stores its setup data in its own nonvolatile storage
   *  and builds a local setup name from the given setupName. This local setup
   *  name is returned as Serializable.
   *
   * @param  setupName  under this name the setup data is stored on the manager
   *       frame.
   * @return            an Serializable which holds all setup data or
   *       alternatively a Serializable which hold a local setup name.
   */
  public Serializable getSetupData(String setupName);


  /**
   *  Method to activate setupData in a device. The Serializable returned in
   *  getSetupData is given back to the device. If there are problems to
   *  activate the setup the device has to go to STOP state and display the
   *  error.
   *
   * @param  setupData  setup data which was returned in an previous call to
   *       getSetupData.
   */
  public void setSetupData(Serializable setupData);
}
```

Figure 5-22: SetupInterface.java

5.6.2.9 VersionInterface.java

```
package org.up3i.gui.devicegui;

/**
 *  Interface for versions
 */
public interface VersionInterface
{
  /**
   *  Gets the version of the device GUI
   *
   * @return    The device version
   */
  public String getVersion();
}
```

Figure 5-23: VersionInterface.java

5.6.2.10 DeviceGUIAccessInterface.java

```
package org.up3i.gui.devicegui;

/**
 *  Over this interface the manager give access rights to the particular GUI.
 */
public interface DeviceGUIAccessInterface
{
  /**
   *  Sets the GUI access rights to the GUI.
   *
   * @param  isOperatable  true means this GUI may change values in the relating
   *       device.
   */
  public void setDeviceGUIAccess(boolean isOperatable);
}
```

Figure 5-24: DeviceGUIAccessInterface.java

*** end of document ***