memjet

DURAFLEX™

Software Release Notes

Rev #: 1.01
SW Version: R5.2.2

Date: 29 March 2022

Memjet Confidential

This document is provided to the recipient subject to and in accordance with the terms of

- *Non-Disclosure agreement*
- *Provision of Technical Documentation and Samples Standard Terms and Conditions; and/or*
- *Component OEM Agreement*

signed by recipient and Memjet.

To the extent permitted by law, this document is PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING BUT NOT LIMITED TO, IMPLIED WARRANTIES OF ACCURACY, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OF ANY THIRD-PARTY INTELLECTUAL PROPERTY RIGHTS.

Memjet will not be liable for any misstatements or omissions, including but not limited to misstatements or omissions in relation to instructions for the operation, use or maintenance of any equipment, samples, components, or software. Recipient acknowledges and agrees that whilst care has been taken in compiling this information, it may contain estimates and draft information, and may not be current, accurate or complete. The information contained herein is subject to change without notice.

In the event of any conflict between the terms of this document and any executed agreement between recipient and Memjet, the terms in such executed agreement shall control and prevail. For the sake of clarity, only warranties expressly made in executed agreements will be binding on recipient and Memjet. Nothing herein should be construed as constituting an additional warranty.

Revision History

Doc. Version	SW Release	Date	Details
V1.00	R5.2.2	04 Mar 2022	First release
V1.01	R5.2.2	29 Mar 2022	Updated Section 6.1 and 6.2 with reminder to save a copy of the PrinterKeyStore file, and restore it after the upgrade is complete.

Contents

Figures

1 Introduction

DuraFlex® software version R5.2.2 is a minor release that includes one correction.

These release notes cover upgrading from R5.2.1 to R5.2.2

1.1 Typographic Conventions

Throughout this document, the following typographic conventions are used:

`Code Character`	`Courier` font is used to identify HTTP GET and POST commands with associated arguments, as well as references to source code, job states, registry settings, directory/file names, XCI commands, and XML settings.
Bold	Text that appears on-screen in the user interface is shown in **bold** font. This includes UI buttons, engine states, warning codes, and fault codes.
Yellow Highlighting	Yellow highlighting indicates sections that are new or updates in this version of the document, compared to the previous version.

1.2 Additional Documentation or Access

For additional product-related technical documents, go to your Memjet Partner Site.

If you need access, enter a case in Service Desk (*https://OEMsupport.memjet.com*), send an email to Memjet Customer Support (*customer.support@memjet.com*), or contact your Technical Account Manager.

2 New Features and Improvements

This section describes enhancements made to the software in this release.

2.1 Medium Service Algorithm Updated

In some scenarios the Medium Service was not effective at recovering a printhead, while a Heavy Service, took too much time and used too much ink. With this release, the Medium Service algorithm has been adjusted to use a Heavy Wipe instead of the Medium Wipe, no other aspects have changed. This will allow OEMs more flexibility between Light Services and Heavy Services.

New MediumService:

```
MediumService(flush_volume=20, flush_speed=70, circulate_speed=26,
wipe_type=HeavyWipe, num_wipes=1)
```

2.2 System Now Checks for IR Tank Overflow Before Draining or Depriming Printhead

Previously there was no way for the PES software to know if an IR tank was full, and the use of some operations could cause the IR tank to be overfilled. The system will now check if the IR tank is already full before starting any process that may cause it to become overfull, i.e. draining or depriming the printhead.

2.3 New In-job Maintenance Settings in PES Software

There are 4 distinct types: declog/pre-page spitbars/inter-page spitbars/KWS

These, together with 'sacrificial' pages and inter-page gaps, can be combined in various ways to achieve a wide variety of maintenance regimes. For more detail see Section *5 Intra Job Maintenance*

2.4 Improved Handling of Indeterminate Priming of Printhead

Previously it was possible for the PES software to get into a state where it was unsure whether the printhead was primed or not. For example:

- If the Prime or Deprime algorithm failed.
- If the PES software thinks it is primed but the printhead "primed" flag is not set due to unexpected activity (i.e., e-stop or other operator intervention).

If this occurred, the PES would set the state to "deprimed", as printing is not possible or may cause damage. However, the printhead may be partially inked, making removal difficult.

With this update the PES software now explicitly tracks and reports the indeterminate primed state. It will not permit printing, but it will allow selected algorithms to run, and will update its state if a Prime or Deprime are performed successfully.

2.5 Deprime Can Now be Run During a PES Fault

The deprime function must only be called during `EngineState.DEPRIMED_IDLE`, `EngineState.PRIMED_IDLE`, or `EngineState.FAULT` states. If the deprime function is called while the engine is in `EngineState.FAULT`, the engine will return to `EngineState.FAULT` when complete. To determine the success of the operation, examine the `PrintheadStatus.primedState`. On success it will be `PrimedState.DEPRIMED`. On error it will be `PrimedState.UNKNOWN`.

2.6 Wiper Indexing Now Occurs Before and During a Heavy Service

To improve printer reliability after a period of non-use, the wiper is now indexed prior to the first wipe. This way any dehydrated ink left on the wiper during storage will not contaminate the printhead nozzles should they happen to already be clean.

2.7 PesClientExample.py Updated for R5.1

The PesClientExample.py application has been updated for R5.1

2.8 Printer Behavior After Resuming a PAUSED Print Job is Now Consistent with Starting a Print Job

Previously when resuming a paused print job, not all maintenance steps were performed before printing resumed. With this update, after a paused print job is restarted, the printer will follow the same behavior as when starting a print job and all maintenance procedures will be identical.

2.9 Printing a Subset of Pages from a PDF

With this release, OEMs can now print a subset of pages from a PDF, for example:

- Page N to the end
- Page N only
- Pages N to M

This will allow OEMs to print from page N to the end of the PDF to support restarting of jobs that have been cancelled due to a paper jam or similar.

Note: This feature only applies for OEMs using the embedded RIP. The kenmarecat utility has been modified with a '`--pagerange`' setting.

2.10 Implement Spit In-Flight

Spit-in-flight is an option available to avoid the need to run a declog spit when the printhead reaches the print zone. Please contact Memjet support for more details if you are interested in this feature.

2.11 PES Software Will Now Reject a Job if the Selected Dither or ICC Profile is Not Available

Currently if the specified dither and/or ICC profile are not matched when the job is sent to the printer then the system will do one of two things:

- do a best attempt with a default dither/profile, or
- reject the job and inform the User of the error.

With this release, this choice is now configurable on a per-system basis. The default for this choice is to do a best attempt with the current selection as this means that OEM's that rely on current behavior will not be impacted. To change the settings, create a file named `tmp.cfg` in `/opt/memjet/kenmare/config` and give it the following contents with true/false settings as you need:

```
[Configs]
rejectJobWithUnknownDither=true
rejectJobWithUnknownIccProfile=false
```

2.12 Image Justification Can be Applied via the PES API

With this release, the PES API supports setting of image justification. This allows the printed image to be aligned to the left, right, or center of the page. See the `imageJustification` setting in the PES IDL files for more information.

2.13 Avoiding Servicing Mode When Checking for PeriodicIdle No Longer Required

Previously, any checks to determine whether period maintenance is required were performed within the `PeriodicIdle` which could disrupt the ability to print.

With this release, any system checks to determine whether any action is required are performed by inspecting the status store. This will provide improved interaction with the printer controller as PES requests will only be rejected if periodic maintenance is being performed.

2.14 New PES Commands Added for Setup Printhead

A non-functioning, setup printhead will soon been available to assist with commissioning DuraFlex printers. New PES commands have been created to simulate the printhead electronics and to circulate ink through the setup printhead. See the following commands in the PES IDL files for more information:

```
circulateInk
drainInkFromPrinthead
```

2.15 Embedded RIP Failures Now Included in PES Software Thrift Status

Improvements have been made to show errors when the embedded RIP fails to RIP a PDF. The failure is currently only reported in the Kenmare log file.

2.16 PES Interface Now Reports OEM Job Data from the Embedded RIP

It is now possible to retrieve OEM job data from the embedded RIP. The PES command `getJobCustomInfo()` will return OEM job data (if supplied) when using the embedded RIP.

2.17 Embedded RIP Provides Improved Status Via PES API

When using the embedded RIP, more status information is available, such as job state, activity, etc. See the PES IDL files for details.

2.18 Job First Page Length Now Displayed via PES Interface

The `JOB_QUEUED` and/or `JOB_STATUS` events return the length of the first page in a job. This information may be useful in some applications where page size is constant for all pages of a job but not known until print time by the printer controller.

3 Bug Fixes

The section includes corrections implemented in this release.

3.1 End of Life Printheads Can Cause Print Engine to Become Unresponsive

This release resolves an issue whereby a printhead reaching its maximum allowed print life can prevent the print engine from responding to commands.

3.2 Use of Print Engine Shutdown Command Clarified in PES IDL

The print engine shutdown command can only be called while the print engine is in the IDLE state, i.e., either primed or deprimed. This information has been added to the PES IDL.

3.3 PDF Rotation Flag Now Respected by Embedded RIP

Previously when a PDF was rotated using Adobe Acrobat (the rotation implemented by adding a rotate flag to the image) and then ripped with the internal RIP, the image was no longer rotated. This issue has been resolved and PDF images are rotated correctly.

3.4 PES Software Job Chaining Issue Fixed

Previously, there was a race between the PES Software idle loop checking for chainable jobs, and `prepareToPrint()` being called to start a new print session. This could cause job chaining to be attempted which would fail if the first job was short. This issue has been fixed.

3.5 Printer Actuator Timeout Failure Fixed

In some cases, DuraFlex components may timeout if the printer was left on for several weeks. This issue has been resolved.

3.6 Some Actions Will Fault While Ink Tank Refill is in Progress

The Periodic Idle, Prime, Service, Flush actions will fault if they are run while an ink tank refill is in progress. This will prevent actions from being performed with low ink levels.

3.7 Print Engine Checks Wiper is Present at Start/End of Wipe

The print engine will detect if a wiper cartridge is present at the start and the end of a wipe.

3.8 Print Engine Shutdown Wipe Fixed

Performing a print engine shutdown will not trigger a printhead wipe if a printhead is not installed.

3.9 PES Software Updated to Cancel Internal RIP Jobs

The `KPesCommand.clearJobQueue()` command has been updated to call the required `KenmareCommand` job-cancel methods if running in Internal RIP mode.

3.10 Mid Job Service No Longer Triggers Error

Performing a mid-job service no longer triggers the "gymea not ready" error when the printer is close to end of job.

4 Known Issues

The section details known issues with this or past releases.

4.1 Clarify Issue with MaxSpeedIps

The allowed `MaxSpeedIps` returned from the DuraFlex system is an estimate and should not be directly used as the input to the `prepareToPrint()` for the intended print speed. The intended print speed should match the speed of the media specifically such that the proper KWS is used. In this release, the allowed `MaxSpeedIps` returned will cause a fault if used as input to `prepareToPrint()` because it is slightly higher than allowed. This may be a useful warning to set the proper speed.

4.2 Cut Sheet Inter-page Gap Causes Job to Stop Printing

When the new intra-job maintenance features are used with cut sheet printing, printing may stop during a job for no obvious reason. If it occurs the print job will stop printing however the KWS and intra-page spit bars (if enabled) will still be printed on media fed through the print zone.

This issue occurs when the physical page gap between cut sheet pages is smaller than the minimum gap set by the "`secondaryPrePageSpitLength`" + "`prePageSpitGap`". If this situation occurs, consider adjusting the physical page gap or the settings.

5 Intra Job Maintenance

This section describes the PES API (Kareela) support for various types of in-job maintenance, that is, additional ink ejection which can be coordinated with a job's print data in datapath pipeline (Gymea) to improve print quality. There are 4 distinct types of maintenance:

- declog
- pre-page spitbars
- inter-page spitbars
- KWS

These, together with 'sacrificial' pages and inter-page gaps, can be combined in various ways to achieve a wide variety of maintenance regimes.

5.1 Where Ink Gets Ejected

This section describes various settings provided in the PES API (Kareela) and the effect they have upon where ink is ejected.

5.1.1 Vertically

Ejection of ink in the vertical dimension (direction of media movement) is synchronized to OEM-provided input signals i.e., media encoder and optional top of form (TOF) signal. Each set of media timing signals and the printheads that use them is called a media timing group.

The position of media as it moves through the engine is tracked according to the media encoder input signals. This allows pages to be printed and other maintenance ejections to occur at the correct vertical scale, independent of media speed.

The vertical positions on the media where the engine can start printing a page are referred to here as top-of-page points. If the optional TOF input signal is being used then the points on the media where the TOF signal becomes active (adjusted for the distance from the sensor to the print zone, using the `mediaReadyOffset` setting) become candidate top-of-page points. If a TOF signal is not used, then top-of-page points implicitly occur as soon as vertical space is available for printing.

As a top-of-page point passes under an individual printhead it will, if ready, begin printing the next page of the current print job.

As shown in *Figure 1*, preceding each top-of-page point there is an optional pre-page spit target area onto which maintenance ink (declog and pre-page spitbars) can be ejected. There are two types of pre-page spit target area:

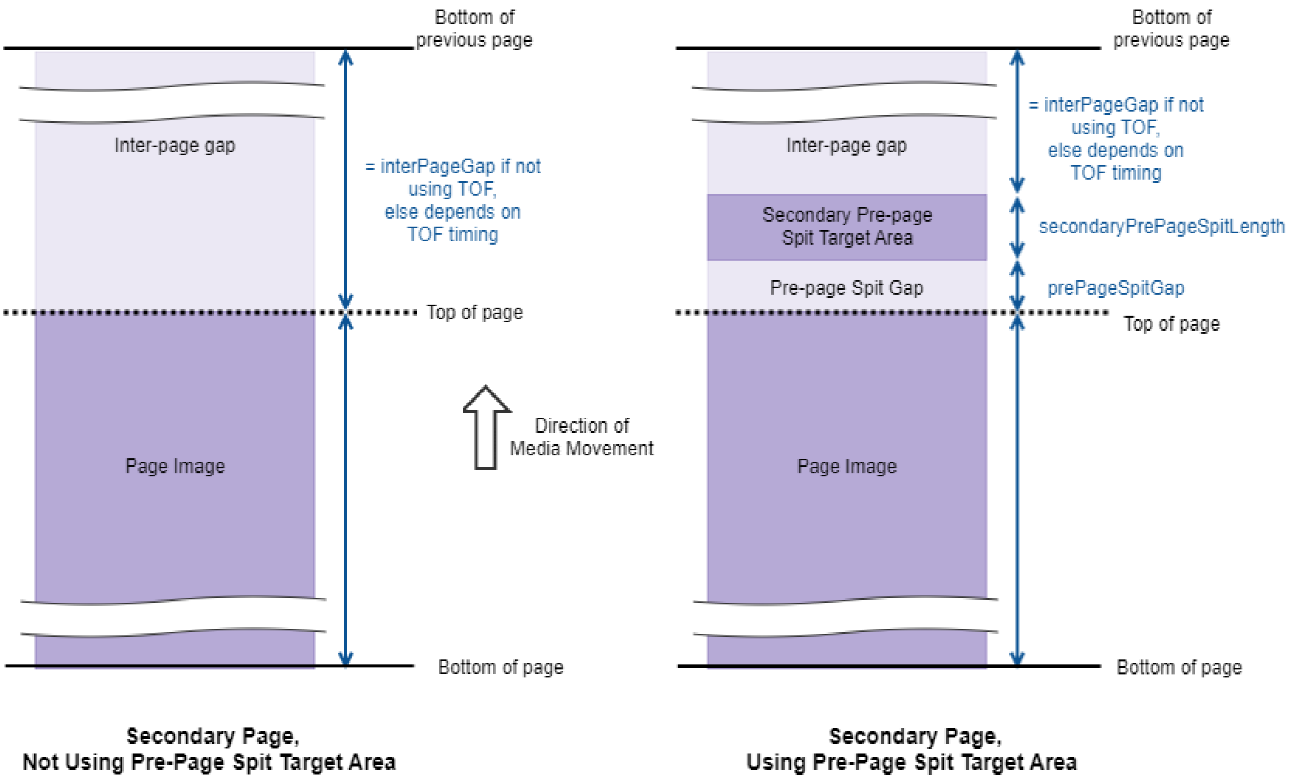
- a first pre-page spit target area, which precedes the first page of a job chain. That area's length is configured by the engine's `firstPrePageSpitLength` setting, and the gap between the bottom of that area and the top-of-page point for the first page is configured by the engine's `prePageSpitGap` setting.
- a secondary pre-page spit target area, which precedes all other pages of a job chain. That area's length is configured by the engine's `secondaryPrePageSpitLength` setting, and the gap between the bottom of that area and the top-of-page point for the page that follows is configured by the engine's `prePageSpitGap` setting.

In this document the term pre-page spit target area is used to refer to both types of target area.

The type of pre-page ejections, and therefore whether the pre-page spit target area is used at all, is configured through the engine's `declogMode` and `prePageSpitIntensityPct` settings.

Any vertical distance remaining between the bottom of a page and the next page's pre-page spit target area (or its top-of-page point if the optional pre-page spit target area is not being used) is known as an inter-page gap. The length of this gap will depend upon the relative time between consecutive TOF input signal transitions. If a TOF signal is not used, then an inter-page gap with length equal to the engine's `interPageGap` setting will be synthesized. If a TOF signal is not used and the `interPageGap` setting is zero, and no pre-page spit target area is configured, then there will be no gap between pages i.e., the top of a page will begin printing immediately after the bottom of the previous page.

Figure 1 – Inter-page Gap and Target Area Definitions


Bottom of previous page
Inter-page gap
= interPageGap if not using TOF, else depends on TOF timing
Top of page
Page Image
Direction of Media Movement
Bottom of page
Secondary Page, Not Using Pre-Page Spit Target Area
Bottom of previous page
Inter-page gap
= interPageGap if not using TOF, else depends on TOF timing
Secondary Pre-page Spit Target Area
secondaryPrePageSpitLength
Pre-page Spit Gap
prePageSpitGap
Top of page
Page Image
Bottom of page
Secondary Page, Using Pre-Page Spit Target Area

5.1.2 Horizontally

Ejection of in-job maintenance ink (KWS, inter-page spitbars, pre-page spitbars and declog) is performed across the entire width of each printhead.

5.2 Types of Pre-page Ejection

There are two types of pre-page ejection:

- **Declog**: an energetic ejection of ink, across the entire printhead, which aims to clear any partial clogging of nozzles due to dehydration.
- **Spitbar**: regular printing type of ejection, across the entire printhead. Each color can be configured to print at a specified print density.

Pre-page spitbars are enabled when the `prePageSpitIntensityPct` settings for one or more colors are set to be greater than zero. If pre-page spitbars are enabled, then they appear before each page in a pre-page spit target area.

In some declog modes declog ejections and spitbars must share the pre-page spit target areas. Declog will start at the top of the pre-page spit target area and continue for the configured number of lines. However, the number of declog lines will be truncated if necessary to ensure that declog does not extend beyond the target area. Pre-page spitbars are placed at the bottom of the pre-page spit target area.

Declog takes precedence over spitbars, so where they would overlap, only declog ejections will appear. Thus if, for example, declog is configured such that it completely fills (or over-fills and so is truncated to fill) the target area then no pre-page spitbars will appear.

If truncation of the declog ejections become likely, the sacrificial modes may be more appealing.

There are some constraints in the current implementation:

- the length of all pre-page spitbars will be limited to the smaller of `firstPrePageSpitLength` and `secondaryPrePageSpitLength`.
- further, if spitbars are enabled but there is **no** declog, then the length of both first and secondary pre-page spit target areas will be the smaller of the two. For example: if `firstPrePageSpitLength` = 20mm and `secondaryPrePageSpitLength` = 10mm, and there is no declog, then a 10mm spitbar will appear before both first and secondary pages. If declog **is** enabled, each pre-page spit target area where declog will appear will be its configured length, but the spitbar component printed in the area will be limited to the smaller of `firstPrePageSpitLength` and `secondaryPrePageSpitLength`. Thus, in the earlier example, when declog will appear in the first pre-page spit target area then it will be 20mm long, with declog positioned starting at the leading edge, and a 10mm spitbar positioned to end at the trailing edge (possibly partly or completely replaced by declog if they overlap). And when declog will appear in the secondary pre-page spit target area then it will be 10mm, with declog at the leading edge and spitbar effectively filling the remaining (if any) portion of the 10mm area.
- the intensity value used for a spitbar will be the `prePageSpitIntensityPct` setting value rounded to the nearest multiple of 10. For example, setting `prePageSpitIntensityPct` to a value of 5 to 14 inclusive will result in a spitbar with intensity of 10% for that color.
- when determining how many declog ejections will fit into a given vertical area, the maximum allowed speed is used, rather than the intended speed of the job. If intended speed is less than the allowed maximum speed, The effect is that less declog ejections are used than would have fit at intended speed.

5.3 Declog Modes

Note that declog is time based. Due to the nature of a declog spit, each spit takes a fixed amount of time, and, once it starts, is synchronized with media movement. Thus, at slow speed it will be vertically more concentrated in an area, as speed increases it will be more spread out.

The detailed configuration to be used for declog ejections for the two types of pre-page spit target area (first and secondary) are set in the `hwparamstore` JSON file. This config includes the number of lines of declog to be ejected. This number is treated as an 'ideal', but the actual number used may be reduced if that is necessary to restrict the declog ejections to keep within the corresponding pre-page spit target area, given the intended media speed for the job being printed. If the number of lines of declog must be reduced in either or both pre-page spit target areas then this will be reported by the engine, by setting `FIRST_PAGE_SPIT_TRUNCATED` and/or `SECONDARY_PAGE_SPIT_TRUNCATED` conditions in the `engineStatus` field of the `AllStatus` structure when the job starts printing. The detailed declog config for the sacrificial page or pre-job declog, if they are enabled, will be the `hwparamstore` JSON file settings for the first pre-page spit target area.

Declog modes when not using a TOF signal for all pages

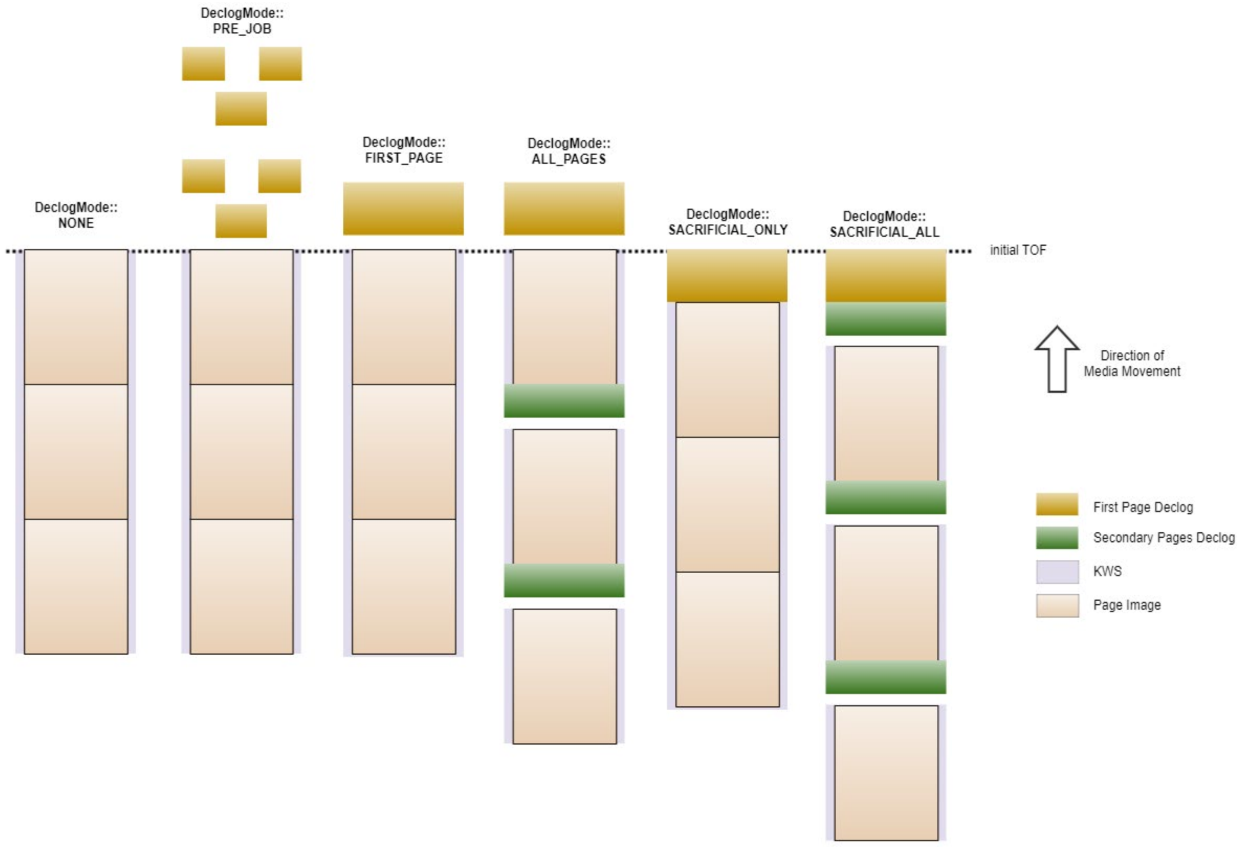
`TofSyncMode::NONE` or `TofSyncMode::FIRST_PAGE` can be used on a roll-fed printer, which doesn't use a TOF sensor input signal to synchronize all pages.

Figure 2 shows the ejections for a 3-wide printer when the various `DeclogModes` are combined with these `TofSyncModes`, using configuration:

- pre-page spit target areas defined (`firstPrePageSpitLength` and `secondaryPrePageSpitLength > 0`)
- pre-page spitbars are disabled (`prePageSpitIntensityPct = 0`)

- no inter-page gap (interPageGap = 0)
- a small gap after the pre-page spit target area (prePageSpitGap > 0)
- KWS is enabled
- inter-page KWS is disabled (allowInterPageEjections = false)

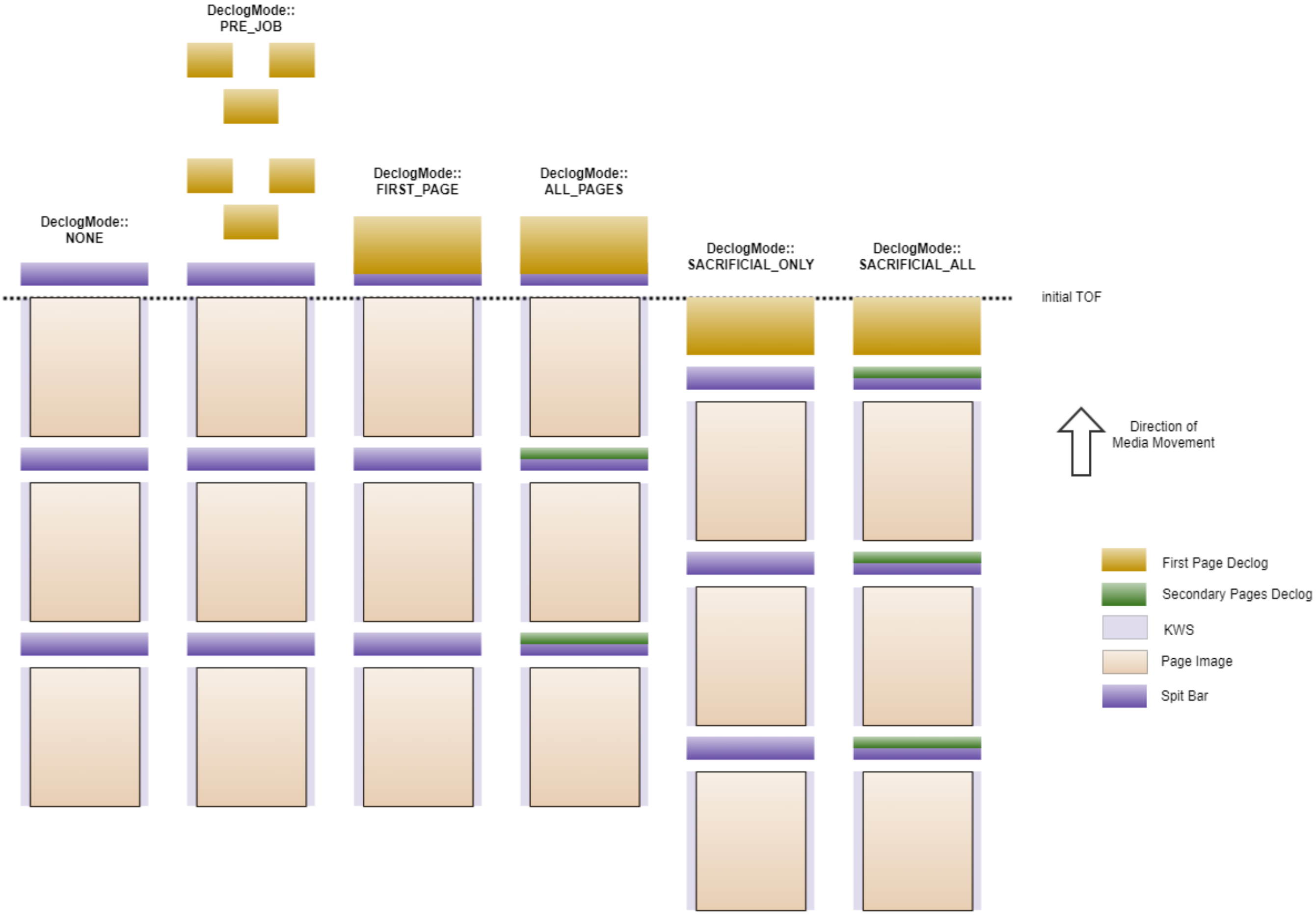
Figure 2 – Various Declog Modes for 3PH-Wide System



The key difference between PRE_JOB and the other declog modes is that PRE_JOB is performed on all printheads simultaneously, so it is spread over a greater distance; where it lands is not related to the first pre-page spit target area, and firstPrePageSpitLength is ignored. For printheads with large Y-offsets, there may be a significant gap between the end of PRE_JOB declog and top of the first page. For the other, non-PRE_JOB modes the declog is synchronized so that each printhead completes declog close together, either in the pre-page spit target area or on the sacrificial page.

Figure 3 is similar to *Figure 2* except with pre-page spitbars enabled and an inter-page gap introduced:

Figure 3 – Various Declog Modes with Spit Bars Enabled



5.4 Declog Modes with TOFs For All Pages

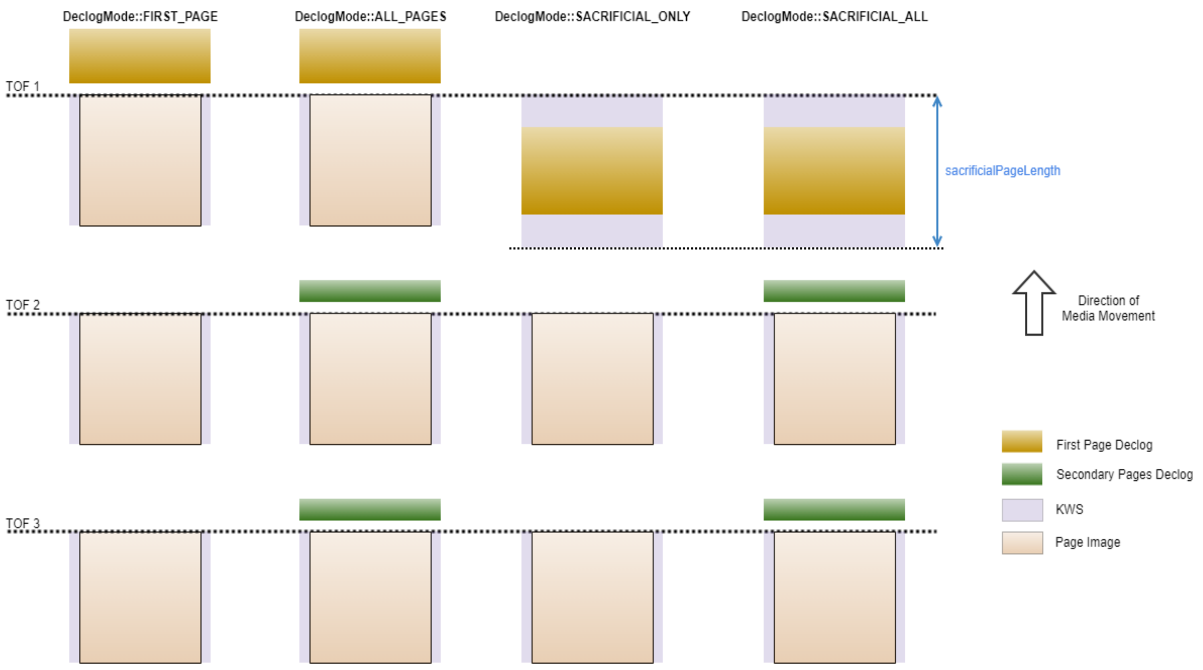
`TofSyncMode::ALL_PAGES` and `TofSyncMode::TRANSACTIONAL` modes are used when a TOF signal is being used to synchronize all pages, typically used for cut-sheet printing.

The following diagram shows the resulting ejections when various `DeclogModes` are combined with these `TofSyncModes`, using configuration:

- pre-page spit target areas defined (`firstPrePageSpitLength` and `secondaryPrePageSpitLength > 0`)
- pre-page spitbars are disabled (`prePageSpitIntensityPct = 0`)
- a small gap after the pre-page spit target area (`prePageSpitGap > 0`)
- KWS is enabled
- inter-page KWS is disabled (`allowInterPageEjections = false`)

The `DeclogMode::NONE` and `DeclogMode::PRE_JOB` modes are not shown.

Figure 4 – Declog Modes with Top of Form Signals

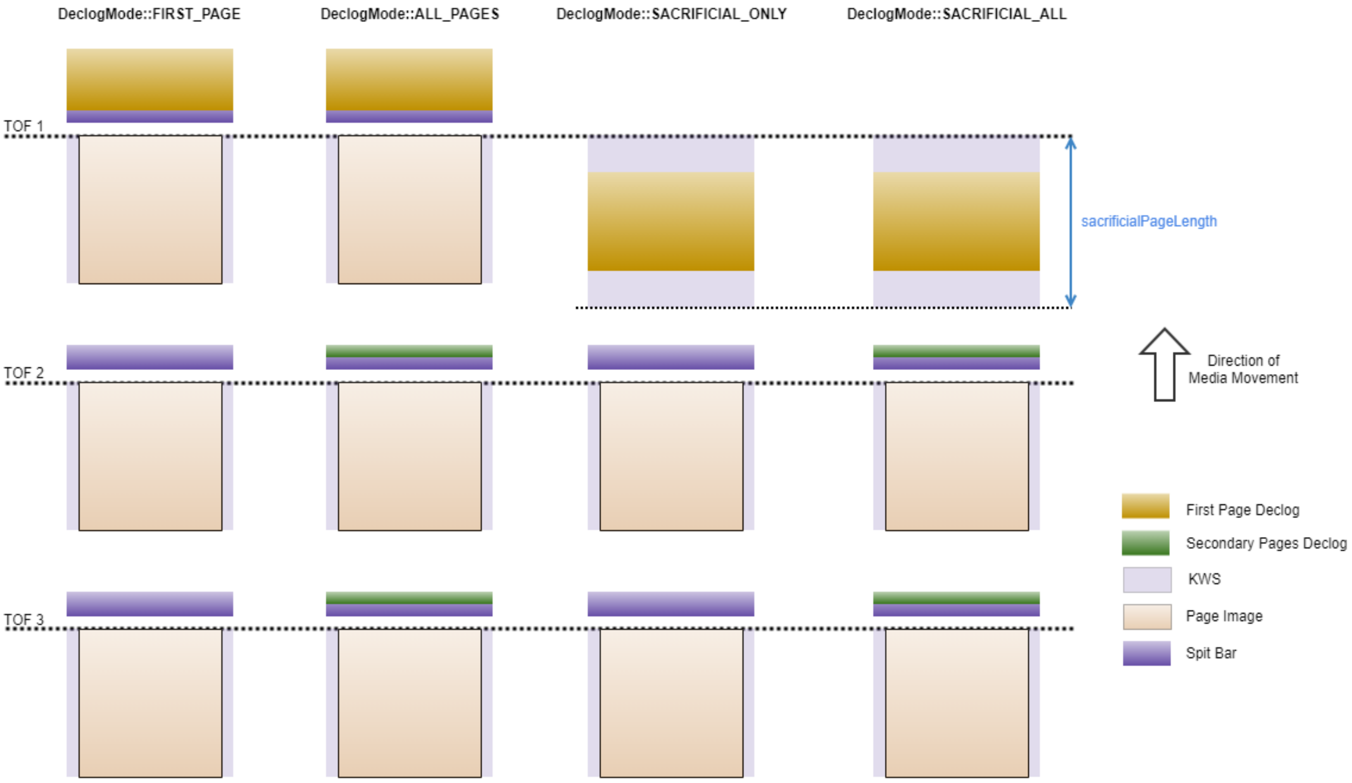


The `FIRST_PAGE` and `ALL_PAGES` modes perform declog only into the pre-page spit target areas (first and/or secondary). Because declog takes a fixed amount of time to complete, for higher media speeds declog requires a greater length of media. Hence to make a desired amount of declog fit in the pre-page target area either a longer target area is required, or the declog will be truncated.

Note:	The length of the target area plus pre-page spit gap must be less than the `mediaReadyOffset` setting, which is the distance from the TOF sensor to the top-of-page point.

If both of those options are undesirable, then one of the sacrificial declog modes can be used. Those modes allow declog to occupy up to the `sacrificialPageLength` setting after the first TOF i.e., the first sheet in a sheet-fed engine is sacrificed for declog and the first page image is instead printed after the second TOF.

Figure 5 – Declog Modes with Top of Form Signals and Spit Bars



5.5 Inter-page Ejections

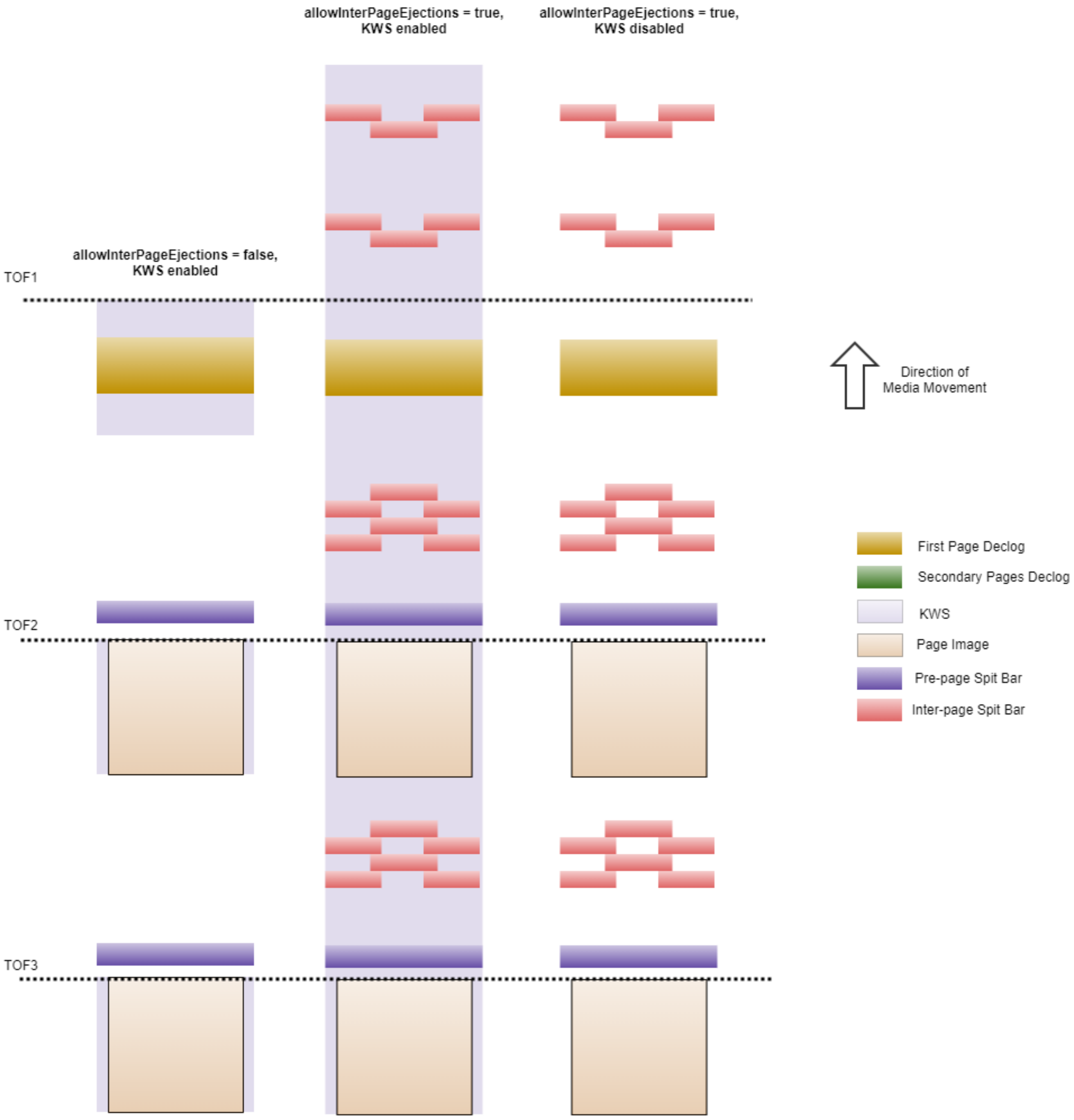
The allowInterPageEjections setting specifies whether the engine is allowed to eject certain types of maintenance ink onto otherwise-unused vertical space before the first page or in the inter-page gaps between secondary pages. As shown in *Figure 6*, these ejections are:

- KWS: If KWS is enabled, it will be ejected, at the configured KWS rate, before the first page (and before the first pre-page spit target area, if one is configured), and in inter-page gaps (and before the secondary pre-page spit target area if one is configured).
- Inter-page spitbars: These will be ejected periodically and appear before the first page and in inter-page gaps if those gaps are large enough. Note that inter-page spitbars only appear before pages that are TOF triggered; for example, when using TofSyncMode::FIRST_PAGE inter-page spitbars will only appear before the first page, and not between subsequent pages. Typically, inter-page spitbars are used for periodic hydration of the printhead in scenarios where the inter-page gap is longer than expected, for example a mis-pick on a sheet-fed machine. Note that inter-page spitbars are in addition to the maintenance spitting in the pre-page spit target areas, which is controlled by the prePageSpitIntensityPct setting. The detailed configuration to be used for inter-page spitbars is set in the hwparamstore JSON file.

On roll-fed printers, users might choose to allow inter-page ejections because that ink would land safely on waste media. Sheet-fed engines where ejection of ink without a sheet under the printhead is not acceptable will typically be configured to disallow it.

The following diagram compares the effect of disallowing versus allowing inter-page ejections, combined with KWS enabled/disabled. In all cases SACRIFICIAL_ONLY declog mode is used with pre-page spitbars enabled.

Figure 6 – Inter-page Ejections and Spit Bar Options



5.6 Keep Wet Spit (KWS)

Keep Wet Spit is also part of in-job maintenance. The KWS rate can be configured in the PES API, via the `userKwsMultiplier` in the `KPesSettings.thrift` IDL.

Horizontally, KWS ink is ejected across the entire width of each printhead. Vertically, KWS ink is ejected onto each printed page, and optionally during inter-page gaps, including the gap before the first page. It is controlled by the `allowInterPageEjections` PES API setting.

6 Software Installation and Upgrade

6.1 Prerequisites

CAUTION: The software upgrade will delete any existing data.

Follow the instructions in this section to install or upgrade the DuraFlex system software.

Perform the following tasks before a new software installation or upgrade:

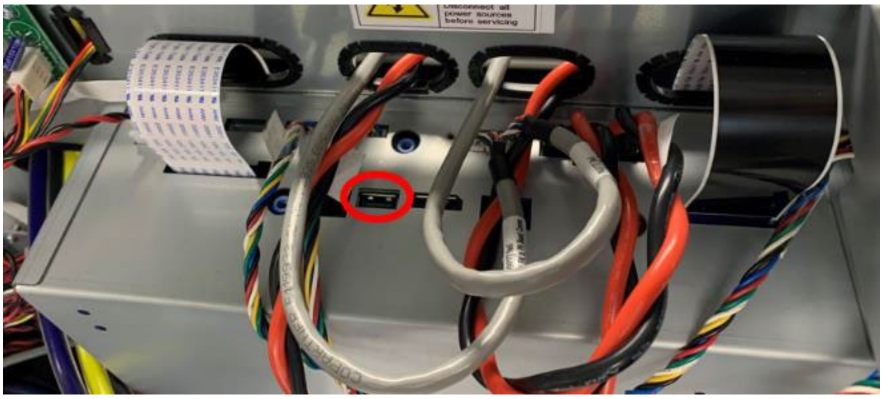
- Create a DPCA LiveUSB drive according to the instructions in the *DuraFlex Installation and Commissioning Guide*.
- Record the Datapath PCA serial number from the label on the electrical enclosure.
- Save any configuration file or logs, e.g. `hwparamstore.json`, etc.
- Save a copy of the `PrinterKeyStore` file. For details on this process, please see Section 5.7.1 of the *DuraFlex Installation and Commissioning Guide*.
- Save any custom ICC profiles or dither profiles.

6.2 Installation/Upgrade Procedure

To install or upgrade software:

1. Set up the system network.
2. Power off DuraFlex.
3. Insert the DPCA LiveUSB drive into the USB port on the printing system.

Figure 7 – Connect LiveUSB Drive via USB Port



4. Power on the DuraFlex system.
5. Log in to DuraFlex using PuTTY with username `duraflex` from the Client PC.

Note: Alternatively, you can also use Windows 10 SSH if that is available.

No password is required. When the login is successful, the PuTTY terminal should respond with a shell prompt: `[duraflex@servername ~]$`

6. In the PuTTY terminal, enter the command below to install the new software on DuraFlex:

```
dtpDpcaSwInstaller
```

7. Wait for the PuTTY terminal to display the following response indicating the upgrade is complete:

```
Installation Complete
Press Return to quit
```

8. While the DPCA LiveUSB drive is still inserted, press **Enter**.

9. Wait until the print unit boots from the DPCA LiveUSB drive again and obtains an IP address.
10. Ping the IP address of the print unit. Verify that the print unit successfully responds.
11. Power off the print unit.
12. Remove the DPCA LiveUSB drive.
13. Power on the print unit and wait until the print unit boots up.
14. Edit the `hwparamstore.json` file to configure the DuraFlex print unit:

    a. From the Client PC, use PuTTY to log in to DuraFlex.
    b. In the PuTTY terminal, use the text editor to open the file:

    ```
    sudo vi /opt/memjet/kareela/data/hwparamstore.json
    ```

    c. Refer to the *DuraFlex Installation and Commissioning Guide*, specifically Section 6.1 Configure the Printing System, to make necessary changes.
    d. Save and close the file.

15. Copy the `PrinterKeyStore` file back onto the printer. See Section 5.7.1 of the *DuraFlex Installation and Commissioning Guide*.

6.3 Set RIP Mode

Starting from software release R4.2.x, the print unit will initially boot in Technictl mode. Therefore, it is required to set the RIP mode to internal or external.

1. Log in to DuraFlex using PuTTY with the credentials (`duraflex` for both username and password).

   When the login is successful, the PuTTY terminal should respond with a shell prompt: `[duraflex@servername ~]$`

Note:	Alternatively, use Windows 10 SSH if that is available. For example, `ssh duraflex@192.168.100.200`

2. Change directory to the `hwparamstore.json` file location:

   ```
   cd /opt/memjet/kareela/data
   ```

3. Open and edit the `hwparamstore.json` file:

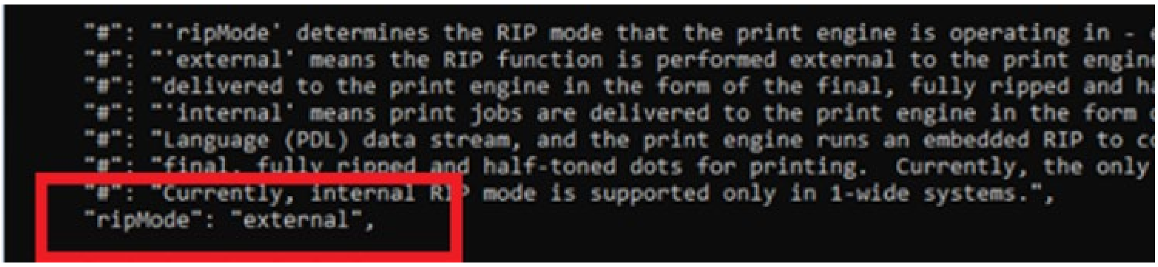
   a. Run the command to open the text editor:

   ```
   sudo vi hwparamstore.json
   ```

   b. Change the value of the `ripMode` variable to match the desired RIP mode.

   For example, if changing the RIP mode to the external RIP mode, set the `ripMode` variable to “`external`”; as shown in *Figure 8*.

Figure 8 – Set RIP Mode in the JSON File



```
    "#": "'ripMode' determines the RIP mode that the print engine is operating in -
    "#": "'external' means the RIP function is performed external to the print engin
    "#": "delivered to the print engine in the form of the final, fully ripped and h
    "#": "'internal' means print jobs are delivered to the print engine in the form
    "#": "Language (PDL) data stream, and the print engine runs an embedded RIP to c
    "#": "final, fully ripped and half-toned dots for printing.  Currently, the only
    "#": "Currently, internal RIP mode is supported only in 1-wide systems.",
    "ripMode": "external",
```

Similarly, if the internal RIP mode is intended, set the `ripMode` variable to “`internal`”.

c. Save the `hwparamstore.json` and exit from the `vi` text editor.

4. Use the PuTTY terminal to enable the same RIP mode that you have set in `hwparamstore.json`.

   a. Disable the current RIP mode:

   ```
   dtpStop
   ```

   b. Power cycle DuraFlex.

   c. Enable the internal or external RIP mode:

   ```
   dtpUseInternalRip or dtpUseExternalRip
   ```