DISTECH CONTROLS™

Data Recovery Service Guide

User Guide

Connecting People with
Intelligent Building Solutions

Data Recovery Service Guide

Distech Controls, Inc.
Brossard, Quebec,
Canada

Legal Notice

Contents

About this Guide

This topic contains important information about the purpose, content, context, and intended audience for this document.

Product Documentation

This document is part of the EC-Net™ technical documentation library. Released versions of EC-Net software include a complete collection of technical information that is provided in both online help and PDF format. The information in this document is written primarily for Systems Integrators. In order to make the most of the information in this book, readers should have some training or previous experience with EC-Net™ 4 or EC-Net$^{AX TM}$ software.

Document Content

This document describes the Data Recovery Service. Topics in this guide include tasks, concepts, and reference information related to using this service.

Document change log

Updates to this document are listed below.

March 17, 2022

Improved and corrected illustration showing station output from the EC-BOS DataRecoveryService.

October 1, 2020

Updated, “Recording (normal operation)” topic to add information about Non-Critical flag to the History Config ID slot.

October 21, 2019

Revised the procedure, “Reformatting the SRAM in a controller”, to clarify that the **Alternate Boot** menu options differ depending on the controller and which EC-Net release is installed.

April 14, 2016

Updates include additional information to reflect the EC-BOS-8-AX Station Pack feature in EC-Net^{AX}-3.8U1.

November 13, 2015

Many minor changes throughout the document.

November 3, 2015

Updates include additional information to reflect EC-Net 4 v4.1 and EC-BOS-8 product release.

Initial EC-Net 4 release document: August 18, 2015

Related documents

Additional information about is available in the following documents.

- *EC-Net 4 Platform Guide*
- *Getting Started with EC-Net 4*

Chapter 1 EC-BOS Data Recovery Service

Topics covered in this chapter

♦ SRAM and/or battery options
♦ Battery-less EC-BOS scenarios
♦ SRAM plus battery scenarios
♦ Unsuitable station for SRAM support

EC-Net 4 includes support for “battery-less” EC-BOS operation, where EC-BOS controllers use a capacitor-charged SRAM (static random access memory), to preserve RAM-resident data when a power outage occurs. This includes station data not yet committed to non-volatile flash memory.

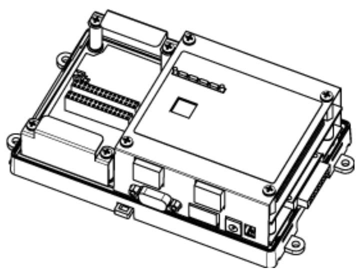
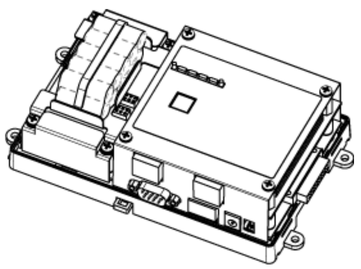
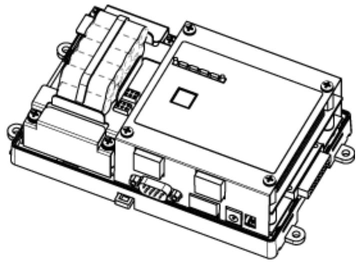
- Initially, this applied only to a QNX-based controller with an installed SRAM option card. Previously, such controllers were always considered “battery-less”, where any installed backup battery was removed when the option card was installed. However, with later releases, including EC-Net 4, the EC-BOS controller can use both SRAM and a backup battery.
- The QNX-based EC-BOS models have onboard SRAM as standard—no option card required. This includes the EC-BOS-8 series controller, plus the EC-BOS-3^{AX}, and NPM6E processor-based series (EC-BOS-6^{AX} and “retrofit board” EC-BOS-603^{AX} and EC-BOS-645^{AX} controllers). The EC-BOS-8 controllers ship “battery-less”. The EC-BOS-6^{AX} and EC-BOS-3^{AX} controllers ship “battery-less” as well however, you can optionally install an NiMH backup battery (identical to the one in EC-BOS-6^{AX} series controllers).
- EC-Net 4 v4.1 and later support the EC-BOS-8 controller model, as does update build EC-Net^{AX}-3.8U1 but with some feature limitations. EC-Net^{AX}-3.7 and EC-Net^{AX}-3.8 support all of the controller models (except the EC-BOS-8) and also all controllers with an installed SRAM option card. For EC-BOS-3^{AX} support, build EC-Net^{AX}-3.7.105 or later is required.

  NOTE: The Security EC-BOS platform does not include SRAM, nor does it support the SRAM option card/ DataRecoveryService in any build of EC-Net^{AX}.

For hardware mounting details, refer the installation document that ships with each SRAM option card or SRAM-equipped controller. This document summarizes usage scenarios and software operation details of the SRAM memory feature.

SRAM and/or battery options

With any SRAM-equipped controller, including the “onboard SRAM” EC-BOS controllers like the EC-BOS-8, EC-BOS-6^{AX}, and EC-BOS-3^{AX}, there are two different ways to utilize SRAM. There is also a scenario in which you may elect *not* to make use of SRAM, even if the controller includes it standard (onboard).

Battery-less EC-BOS (with SRAM)	EC-BOS with SRAM plus backup battery	EC-BOS using only backup battery
EC-BOS-8/EC-BOS-6^{AX}/EC-BOS-3^{AX} (SRAM card unnecessary) 	EC-BOS-6^{AX}/EC-BOS-3^{AX} (SRAM card unnecessary) 	EC-BOS-6^{AX}/EC-BOS-3^{AX} or EC-BOS-6^{AX} series 
This is the “battery-less” controller configuration for the EC-Net implementation of SRAM. No backup battery (NiMH or otherwise) is attached to the EC-BOS. This configuration also applies to the EC-BOS-8 controllers (not shown), which do not support battery backup.	Since EC-Net^{AX} 3.6.44 or later, an SRAM-equipped EC-BOS can utilize both SRAM and an installed backup battery, like the NiMH battery pack shown above.	Sometimes a station is a poor candidate for SRAM support. Disabling SRAM operation is best in this case, even if “onboard” SRAM.

Battery-less EC-BOS scenarios

A standard EC-BOS-8, EC-BOS-6^{AX} or EC-BOS-3^{AX} controller, or any SRAM-equipped EC-BOS (via SRAM option card), when installed *without* an integral, rechargeable battery pack (or external 12V battery), provides a “battery-less” installation. This can offers advantages in certain situations. Some example scenarios are:

- Installation of any battery-equipped device is *forbidden*, due to job site or local regulations.
- Installation without a battery may allow a *higher temperature environment rating*, such as that allowed for the EC-BOS-6^{AX} or EC-BOS-3^{AX} (60 °C maximum, vs. 50 °C maximum if with a rechargeable NiMH battery pack).
- Over time, the periodic replacement of the NiMH battery in the EC-BOS presents too many costly obstacles. For example, your EC-BOS may be installed in a difficult-to-reach location. And as the NiMH battery’s condition declines, “battery bad” alarm notifications are received.

You can meet such issues by installing a standard EC-BOS-8, EC-BOS-6^{AX} or EC-BOS-3^{AX} controller, or if another earlier model (e.g. EC-BOS-6,-7), by removing its NiMH battery pack and installing the SRAM option card. This makes the unit “battery-less”. As a consequence of SRAM-only backup support, EC-BOS battery monitoring no longer occurs—no more “battery bad” alarm notifications.

NOTE: A station running in a battery-less EC-BOS has no seamless immunity to “power bumps”. Although all station data, including components, histories, and alarms, are automatically restored to “pre-event” values as part of station startup (following power restoration), the briefest power outage results in a controller reboot. For more details, see “Battery-less versus battery trade-offs”.

Note that a “battery-less” configuration for an SRAM-equipped controller is now one of two possible configurations where SRAM is used. For more details, see “SRAM plus battery scenarios”.

Battery-less versus battery trade-offs

Consider the following information when thinking about the advantages and disadvantages of battery-less backup.

Any EC-BOS controller with a charged backup battery holds a key advantage over a battery-less SRAM-equipped EC-BOS. The advantage is that station operation continues (uninterrupted) across very short power outages, that is “power bumps” lasting only a few seconds—without initiating an orderly shutdown. A UPS could be used to mitigate this, but this would re-introduce battery maintenance for the UPS.

The NiMH battery provides enough power for this, immediately recharging when power is restored. If a power outage lasts longer than its defined “shutdown delay” time, the NiMH battery allows sufficient time for the EC-BOS to perform an “orderly shutdown”, including saving the station’s database (config.bog) and all recorded alarm and history records.

However, the EC-BOS without SRAM but with a weak NiMH battery is exposed to a different power outage issue—where the potential exists for data loss (since the last station save), due to insufficient battery power to complete an orderly shutdown. Potentially, this could also occur if enough power outages occur in rapid succession—draining the controller’s battery to a low level.

A battery-less EC-BOS solves that problem, as all station-generated data (changed from that stored in its non-volatile flash memory at the time of power loss) is always preserved in SRAM. Upon power restoration, this data is “played back” in the station during startup, then saved in its non-volatile flash memory.

SRAM does not preserve data or files external to a station

Please note that if the EC-BOS power event occurs when station users have unsaved file changes, say in a Px file or Nav file being edited, those unsaved changes are lost. This behavior may seem different from a battery-equipped EC-BOS entering an ordered shutdown—but it is not.

The practical difference is that a battery-equipped EC-BOS may keep running over a short “power bump”. Station users may be aware of such an event, and react by saving changes (click **Save** button in the active view). Providing that communications are still established, the file edited may be saved. Or, power may be lost only momentarily, and then remain stable until the user does a normal save.

NOTE: A battery-less SRAM-equipped EC-BOS does not provide a similar save opportunity after a power bump—it is already busy rebooting. Therefore, as a best practice, you should advise system users of battery-less SRAM-equipped EC-BOSs to save often when editing items like Px graphics and Nav files.

SRAM plus battery scenarios

An SRAM-equipped controller can utilize both its SRAM and an installed backup battery, such as the EC-BOS-6^{AX} or EC-BOS-3^{AX} controller with an optional internal NiMH battery pack, or a controller (with SRAM option card) and its standard NiMH battery pack. This also applies to a QNX-based EC-BOS controller with an external 12V sealed lead-acid battery, either as its main backup battery (EC-BOS-603^{AX} , EC-BOS-645^{AX}) or as a battery in “parallel” with its NiMH battery (EC-BOS-7^{AX}).

NOTE: This topic covers only those controller models that support battery backup.

The hybrid “SRAM plus battery” configuration offers:

- Immunity to power bumps, thanks to the backup battery—similar to a unit with backup battery only.
- A longer allowable “Shutdown Delay” time, up to 10 minutes for a unit with only an NiMH backup battery. This is beyond the 30 seconds (recently 1 minute) maximum period. After continuous operation this long on battery power, at shutdown the EC-BOS saves its database and powers off. So, a unit with a reasonably healthy NiMH backup battery could continue running over a power outage of up to 10 minutes. Note that the station’s “DataRecoveryService” must be enabled to specify this longer shutdown delay in station’s “PowerMonitorService”. If the unit is equipped with a sealed lead-acid (SLA) battery, this shutdown delay can be specified up to 15 minutes.
- Data independence from backup battery condition, again a consequence of the ongoing SRAM operation. If the charge on the backup battery weakened such that the controller was unable to complete a database save on a controlled shutdown, or even if immunity to power bumps was lost, data integrity still remains. Upon reboot from a power restoration, the station’s DataRecoveryService restores (replays) the previously recorded station runtime data from SRAM.

  NOTE: A discharged battery could happen if multiple, consecutive power outages occurred, draining the charge on the rechargeable backup battery. A longer “Shutdown Delay” time could contribute to this. However, unlike with a controller not using SRAM, station runtime data is safe in this case.

Gain the features above by installing an optional NiMH battery pack in a standard EC-BOS-6^{AX} or EC-BOS-3^{AX}, or if another earlier model, by retaining its NiMH battery pack when installing the SRAM option card. By default, this hybrid configuration is standard with a “retrofit board” EC-BOS-603^{AX} or EC-BOS-645^{AX}, assuming the controller’s 12V SLA battery (in the controller’s enclosure) is retained.

Note in this hybrid configuration, EC-BOS monitoring of backup battery(ies) continues—so “battery bad” alarm notifications are still possible, and the regular replacement of backup batteries is still needed.

Again, the SRAM-plus-battery configuration for an SRAM-equipped controller is one of two possible configurations where an SRAM is used: the other configuration is for a controller without a battery.

NOTE: Although this hybrid configuration is typically the most desirable, note that some stations may be poor candidates for SRAM operation, with SRAM support even counter-productive. In this case, a controller platform that includes “built-in” SRAM (such as the EC-BOS-6^{AX} or EC-BOS-3^{AX}) can be configured to disable its DataRecoveryService, and use only its (optional) installed backup battery. Or, an SRAM option card can be removed from another EC-BOS controller, such that it also uses only its installed backup battery.

Unsuitable station for SRAM support

The DataRecoveryService writes current values as they occur to a block of SRAM. When a block is full, the service copies it from SRAM to the controller’s flash memory. A station that creates rapid COV (change of value) histories may fill the SRAM data blocks too frequently, triggering a database save possibly every couple of minutes. Ideally, such database saves to flash memory should occur no more than once an hour.

Saving the database too frequently results in inefficient use of controller CPU time and potential flash problems. Flash memory is designed to be written to a certain number of times. A number of variables contribute to how often the database needs to be saved, including:

- Rate of changes that need to be persisited
- Size of the changes (histories, alarms, and setpoint changes differ in size)
- Amount of free flash memory space

Two examples use only the number of changes that occur per hour to demonstrate the need to consider the COV rate when selecting station backup options:

- The first EC-BOS-6^{AX} station with 176 COV histories. Each writes every 5 seconds. In addition, the station has other NRIO and BACnet components that add to the histories. 176 COV histories changing every 5 seconds yields 126,720 changes per hour (176 * 12/min * 60 min/hr). Automatic database saves occur every 6 minutes. This number of changes is well beyond the recommendation for a controller using only SRAM and flash memory to backup station data.
- A second EC-BOS-6^{AX} station has 1200 interval histories, each with a collection interval of 10 to 15 minutes. If all histories change every 10 minutes (worst case scenario) the result comes to 7,200 changes per hour (1200 * 6 changes/hr). Now compare this number of changes to those in the first example: 126,720/7,200. In the first example, changes occur nearly 18 times more often than in the second. Applying this ratio to the second example, an automatic database save occurs approximately every 1.5 hours (6 minutes * 17.6 = 105 minutes), well within the acceptable range.

Applications that require frequent writes to flash memory should use a controller with a backup battery and disable the use of internal SRAM/flash memory.

Unsuitable station example

An example test station running in an NPM2–type controller was found to fit this unsuitable category. This station had 176 COV histories, each writing every 5 seconds, along with other NRIO components and BACnet components (making little additional history usage). SRAM support in this station resulted in an ongoing automatic save approximately *every 6 minutes*—about a magnitude away from ideal, which is no more than once per hour.

176 COV histories changing every 5 seconds yields 126,720 changes/hour (176 * 12/min * 60 min/hr).

Second example station - suitable for SRAM support

Now consider the suitability of a station with 1200 interval histories, each with a collection interval of 10 to 15 minutes. Planning for the “biggest load” configuration, that is all histories at 10 minute intervals:

1200 histories changing every 10 minutes yields 7,200 changes/hour (1200 * 6/hr).

Note this example station results in a rate of history changes nearly 18 times less than the first (Unsuitable station example), where the ratio of these two “total changes per hour” is 17.6 (126,720/7,200).

Let’s assume everything else is equal between this station and the first (unsuitable) example, including the same amount of free flash space. Then in this case, ongoing automatic database saves would occur approximately every 1.5 hours (6 minutes * 17.6, or 105 minutes)—well inside the “ideal range”.

NOTE: A number of different variables factor into the actual operation of SRAM support, including:

- Rate of changes that need to be persisted.
- Types of changes (histories, alarms, and setpoint changes all have different sizes).
- Amount of free flash disk space on the controller.

Therefore, the examples above do not reflect a full range of different scenarios.

Also see the following related sections:

- The next section, “Disabling SRAM support.”
- For detailed background information on the operation of SRAM support, see “How SRAM support in a EC-BOS works”.
- For tools to monitor operation, see Diagnostic tools for the DataRecoveryService”.

Disabling SRAM support

NOTE: This topic covers only those controller models that support battery backup.

In situations where a station is not compatible with SRAM support, you can equip its host EC-BOS controller with a suitable backup battery and disable SRAM operation. Depending on the controller series, do this in one of several ways:

- For the EC-BOS-6^{AX} or EC-BOS-3^{AX}, install the optional NiMH battery pack. In its station’s **DataRecoveryService**, set the “Service Enabled” property to `false`. In the station’s **PlatformServices** container, make sure the “Battery Present” property is true.
- For the EC-BOS platform using an SRAM option card, remove that option card and install its NiMH battery pack. Then using the platform **Software Manager**, uninstall the `platDataRecovery` module to remove the **DataRecoveryService** from the station. Again, in the station’s **PlatformServices** container, make sure the “Battery Present” property is true.

Chapter 2 Requirements for EC-BOS controller SRAM support

Topics covered in this chapter

♦ Licensing
♦ About the DataRecoveryService

The following is required for SRAM support in the EC-BOS controller, using either “onboard” SRAM or the SRAM option card, along with the platform DataRecoveryService:

- EC-BOS-8, EC-BOS-6^{AX}, or EC-BOS-3^{AX} controller with onboard SRAM, or

  EC-BOS-603^{AX} or EC-BOS-645^{AX} with onboard SRAM, licensed to run EC-Net^{AX} stations, *or*

  EC-BOS-2^{AX}, EC-BOS-6^{AX}, and EC-BOS-7^{AX} controller running EC-Net^{AX}-3.6 or later, with an available option card slot, *and*

  – An SRAM option card, installed in that option card slot.

  – The “`platDataRecovery`” module must be installed in the EC-BOS (this module is automatically installed in platforms with "onboard" SRAM).

- The EC-BOS must also be licensed for this feature—see “Licensing”.

Licensing

The EC-BOS needs a license entry as shown here, to support the SRAM option card and DataRecoveryService:

```
<feature name="dataRecovery" expiration="2030-12-31"/>
```

EC-BOS controllers with onboard SRAM should have the `dataRecovery` license feature by default. For EC-BOSs with SRAM option cards, ensure the feature is present in the license. If the license feature is missing, SRAM will not work - either the DataRecoveryService in its station will be missing or will otherwise remain in a fault condition.

About the DataRecoveryService

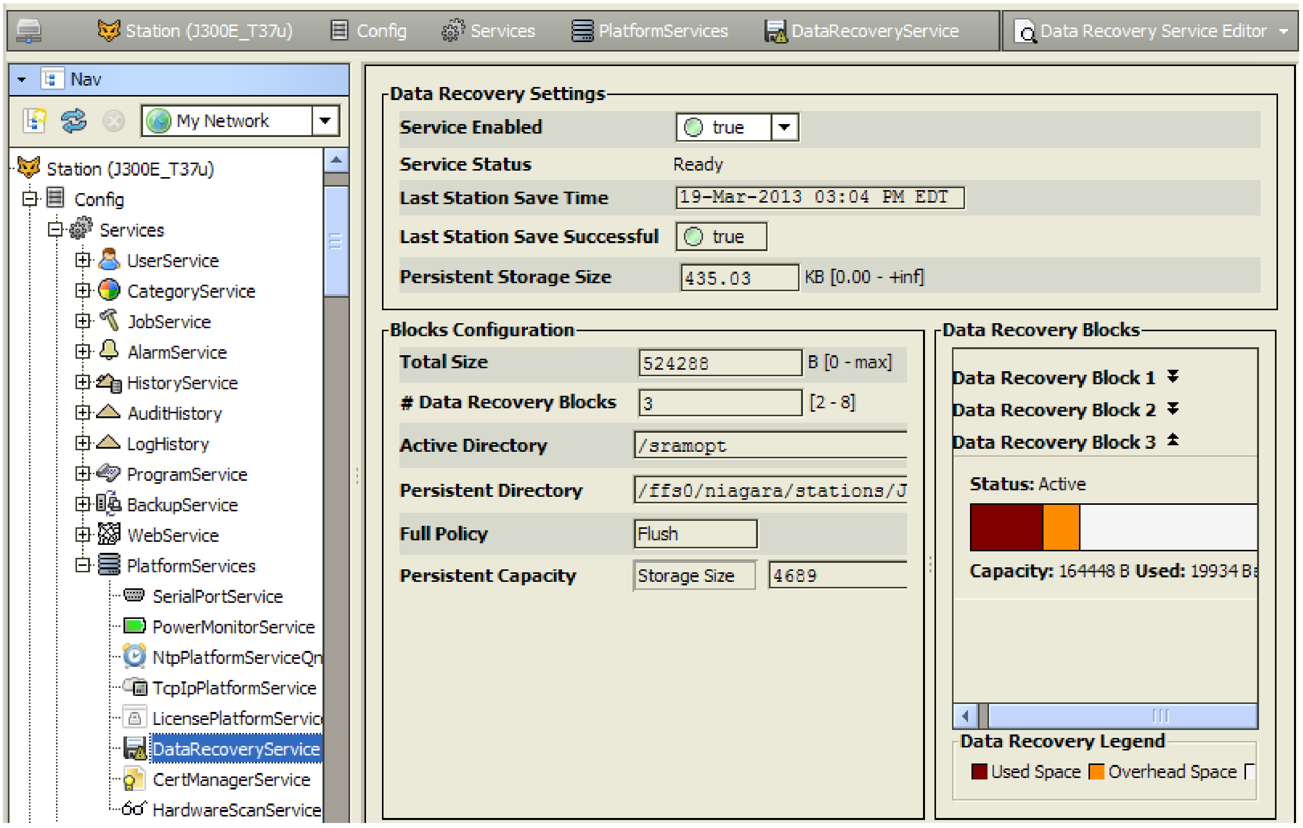
The DataRecoveryService is the station platform service that provides SRAM support for any EC-BOS equipped with SRAM. Providing the `platDataRecovery` module is installed, this service automatically appears under PlatformServices.

NOTE:

- If the EC-BOS is not properly licensed, this service remains in fault.
- This service includes a “Service Enabled” configuration property. The DataRecoveryService does not replace the PowerMonitorService.

  NOTE: Starting in EC-Net 4 v4.0, the **Data Recovery Service Editor** view includes additional configurable fields. For example, **Station Save Limit**, **Station Save Limit Period**, **Generate Alert on Replay** and **Platform Alarm Support**.

Figure 1 Data Recovery Service Editor in PlatformServices of SRAM-equipped EC-BOS



Data Recovery Settings

Service Enabled	true
Service Status	Ready
Last Station Save Time	19-Mar-2013 03:04 PM EDT
Last Station Save Successful	true
Persistent Storage Size	435.03 KB [0.00 - +inf]

Blocks Configuration

Total Size	524288 B [0 - max]
# Data Recovery Blocks	3 [2 - 8]
Active Directory	/sramopt
Persistent Directory	/ffs0/niagara/stations/J
Full Policy	Flush
Persistent Capacity	Storage Size 4689

Data Recovery Blocks

Data Recovery Block 1

Data Recovery Block 2

Data Recovery Block 3

Status: Active

Capacity: 164448 B Used: 19934 B

Data Recovery Legend

🟥 Used Space 🟧 Overhead Space

The figure above shows the default view (for an EC-Net^{AX} release) for the service: the **Data Recovery Service Editor**.

Note the example above reflects a scenario where a station save has occurred since the service was created. Some SRAM “data recovery blocks” have already been flushed to flash (“Persistent Storage Size” is not 0.00 KB).

Data Recovery Service Editor

This Data Recovery Service Editor is the default view of the Data Recovery Service.

The Data Recovery Service Editor view has the following three main areas:

- Data Recovery Settings
- Blocks Configuration
- Data Recovery Blocks

Data Recovery settings

Include the following:

- Service Enabled

  Defaults to `true`, to enable SRAM support via this service. For the EC-BOS with onboard SRAM (e.g. EC-BOS-6^{AX} or EC-BOS-3^{AX}), you can set this to `false` to *disable* SRAM support—in which case it relies on its installed backup battery and its **PowerMonitorService** to preserve station data upon loss of power. (If an SRAM option card-equipped EC-BOS, you can simply *remove* the SRAM option card and *uninstall* the `platDataRecovery` software module).

- Service Status

  The current status of the DataRecoveryService, which is typically “Ready”. Other states include “Starting”, “Configuring”, “Replaying”, “Saving”, “Stopping”, “Stopped”, “Fault” and “Unknown”.

- Last Station Save Time

  Reflects the last time a station save occurred (config.bog written to flash memory). This save may (or may not) have occurred as a result of the DataRecoveryService.

- Last Station Save Successful

  Boolean that reflects if last station save attempt was successful, as either “`true`” or “`false`”. This save may (or may not) have occurred as a result of the DataRecoveryService.

  Note in the case of a newly-created DataRecoveryService, this is “`false`” until the next save occurs.

- Station Save Limit

  Configurable in EC-Net 4 v4.0 and later. Number of station saves that are allowed to occur during the Station Save Limit Period before it is determined that the station is spending too much time saving. Exceeding the limit throws the Data Recovery Service into fault since too much data is being generated.

- Station Save Limit Period

  Configurable in EC-Net 4 v4.0 and later. The period of time for Station Save Limit. If enough saves occur during Station Save Limit Period to exceed the Station Save Limit then the service goes into fault. For example, more than 5 station saves in 3 minutes period triggers a fault.

- Persistent Storage Size

  Reflects the total size of all the “flushed to flash” data block files (“.drdb” files) that exist in the station’s `/dataRecovery` folder, in KB. Initially, this will be 0, until the first SRAM block flushes to flash. It will then increment by that KB amount for each subsequent SRAM block flushed.

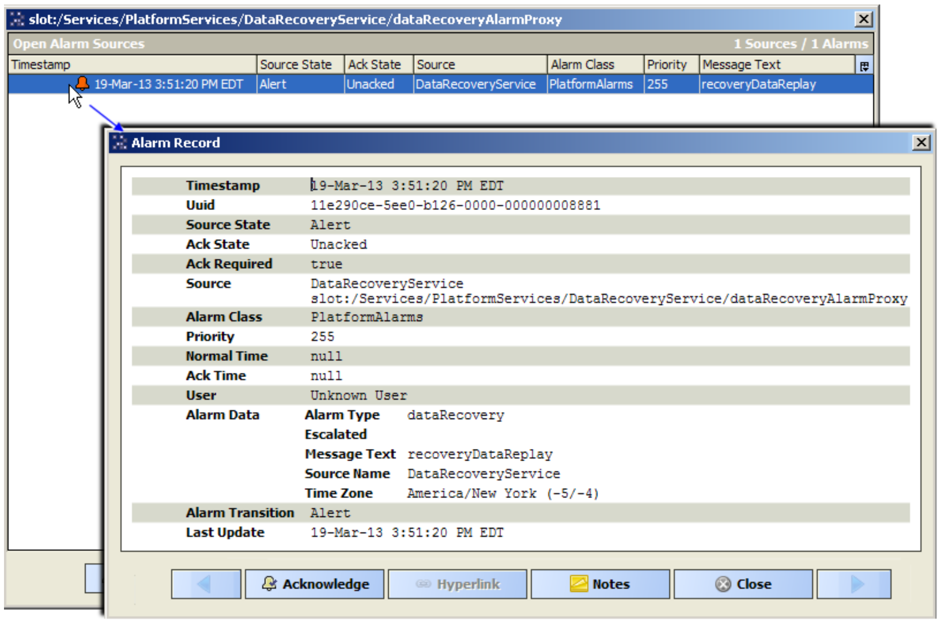
  Note this value is continually compared to the “Persistent Capacity” property in the Blocks Configuration property section.

- Generate Alert On Replay

  Configurable in EC-Net 4 v4.0 and later. Boolean (true/false) value, generates an alert (low priority alarm type) that will indicate that a Data Recovery Replay (power was lost) occurred. This is a persistent artifact that will show up in the alarm console, since it can be useful to know when power loss occurred.

  Default is `false`. If set to `true`, upon any controller boot sequence in which SRAM recorded data is discovered and played back, a corresponding *alert* is routed to the Alarm Class named in the **Data Recovery Alarm Support** container. The following figure shows details for such an example alert.

Figure 2 Example Alarm Record details for an alert generated by DataRecoveryService



Timestamp	Source State	Ack State	Source	Alarm Class	Priority	Message Text
19-Mar-13 3:51:20 PM EDT	Alert	Unacked	DataRecoveryService	PlatformAlarms	255	recoveryDataReplay

Timestamp		19-Mar-13 3:51:20 PM EDT
Uuid		11e290ce-5ee0-b126-0000-000000008881
Source State		Alert
Ack State		Unacked
Ack Required		true
Source		DataRecoveryService slot:/Services/PlatformServices/DataRecoveryService/dataRecoveryAlarmProxy
Alarm Class		PlatformAlarms
Priority		255
Normal Time		null
Ack Time		null
User		Unknown User
Alarm Data	**Alarm Type**	dataRecovery
	Escalated	
	Message Text	recoveryDataReplay
	Source Name	DataRecoveryService
	Time Zone	America/New York (-5/-4)
Alarm Transition		Alert
Last Update		19-Mar-13 3:51:20 PM EDT

- Data Recovery Alarm Support

  Configurable in EC-Net 4 v4.0 and later. This is the standard container slot for routing platform service-generated alarms or alerts, in this case an *alert* from the **DataRecoveryService** upon any controller boot sequence in which SRAM recorded data is discovered and played back. These properties work in the same fashion as those in an alarm extension for any control point.

Blocks Configuration

These status properties include the following:

- Total Size

  Reflects, in bytes, the total amount of SRAM buffer memory available to the service. For example, this is "`524288`" for the 512 KB SRAM option card.

  NOTE: For EC-BOS-8 controllers this is "262144" bytes for the 256 KB SRAM memory.

- Number of Data Recovery Blocks

  Reflects the number of data block partitions of SRAM used, for example, 3.

- Active Directory

  Reflects the directory used in SRAM for the active data block.

- Persistent Directory

  Reflects the full flash file directory path used to store flushed ".drdb" files, which equates to: `/dataRecovery`

- Full Policy

  Reflects the current policy when an SRAM data block becomes full (currently "Flush").

- Persistent Capacity

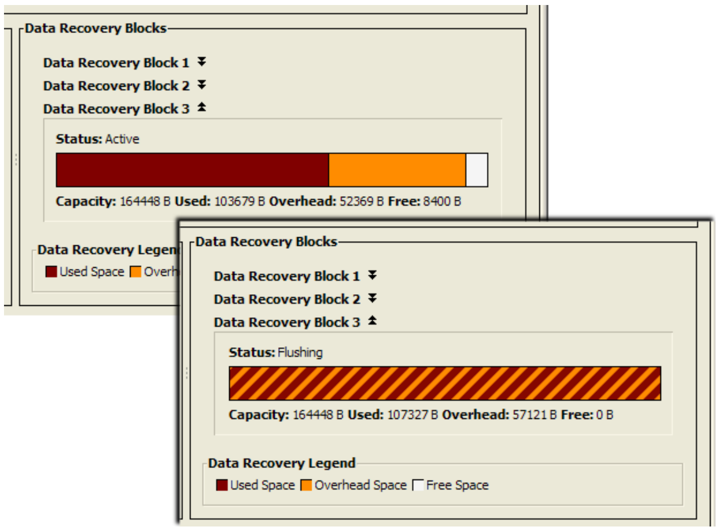
  Reflects the size limit, in KB, for the total of all "flushed to flash" data block files (".drdb" files). If this limit is exceeded (see property "Persistent Storage Size"), the service automatically triggers a *station save* operation. For related details see "Station save effects".

Data Recovery Blocks

This area provides expandable bar graphs for each of the SRAM buffer data blocks, to visually represent the current amount of used space, overhead space, and available free space, along with numerical values.

By default, the currently active SRAM block is expanded, showing a bar graph of current buffer usage.

Figure 3 Example SRAM data block, show both states “Active” and “Flushing”


Data Recovery Blocks

Data Recovery Block 1

Data Recovery Block 2

Data Recovery Block 3

Status: Active

Capacity: 164448 B **Used:** 103679 B **Overhead:** 52369 B **Free:** 8400 B

Data Recovery Blocks

Data Recovery Block 1

Data Recovery Block 2

Data Recovery Block 3

Status: Flushing

Capacity: 164448 B **Used:** 107327 B **Overhead:** 57121 B **Free:** 0 B

Data Recovery Legend

🟥 Used Space 🟧 Overhead Space ☐ Free Space

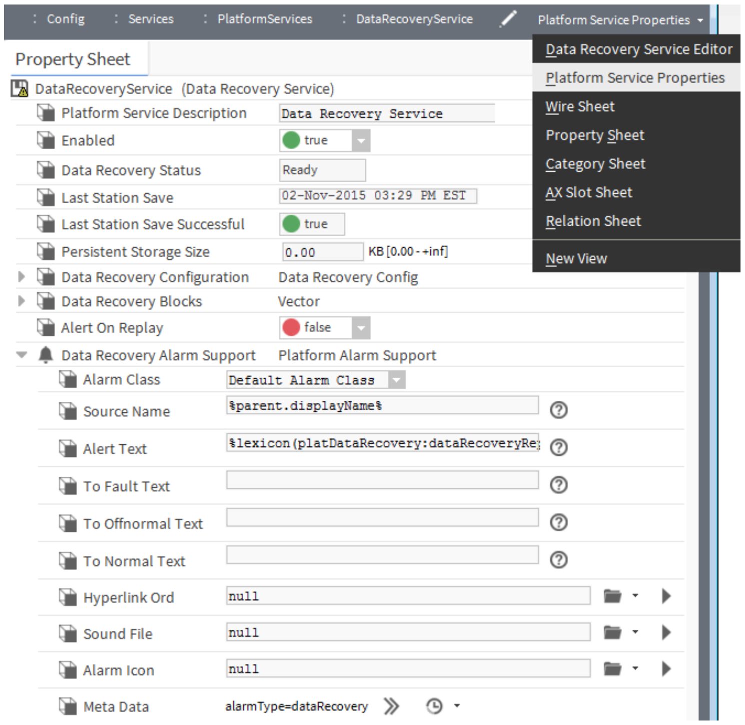
The image above shows an example active SRAM block first near full, then “flushing” momentarily to flash—an operation that lasts only a second or two. Another SRAM block becomes active when this happens, and it is used until it fills and needs to flush to flash.

Above the bar graph of each block, its **Status** is described, typically as either: “Active”, “Idle”, or sometimes “Flushing”, with other states “Purging”, “Awaiting Idle”, “Flush Queued”, “Defragmenting”, “Reserved”, “Fail”, and “Unknown”.

Below the bar graph of each block, numerical amounts display, in bytes, for its total Capacity, currently Used space, calculated Overhead Space, and available Free Space.

Data Recovery Service properties

In addition to the (default) Data Recovery Service Editor view, the Data Recovery Service also has properties on its **Platform Service Properties** view, many of which are shown here.



DataRecoveryService (Data Recovery Service)

Platform Service Description	Data Recovery Service
Enabled	true
Data Recovery Status	Ready
Last Station Save	02-Nov-2015 03:29 PM EST
Last Station Save Successful	true
Persistent Storage Size	0.00 KB [0.00 - +inf]
Data Recovery Configuration	Data Recovery Config
Data Recovery Blocks	Vector
Alert On Replay	false
Data Recovery Alarm Support	Platform Alarm Support
Alarm Class	Default Alarm Class
Source Name	%parent.displayName%
Alert Text	%lexicon(platDataRecovery:dataRecoveryRe
To Fault Text	
To Offnormal Text	
To Normal Text	
Hyperlink Ord	null
Sound File	null
Alarm Icon	null
Meta Data	alarmType=dataRecovery

Figure 4 EC-BOS-8 Platform Service Properties view of DataRecoveryService

Most of these properties are also on the **Data Recovery Service Editor** default view.

This view alsoincludes the following read-only informational fields (not pictured above):

- Too Many Saves — Boolean (true/false). Default is `false`.
- Station Save Limit (editable on the Recovery Service Editor view) -
- Station Save Limit period (editable on the Recovery Service Editor view)

Chapter 3 How SRAM support works in a controller

Topics covered in this chapter

♦ Usage goal
♦ SRAM based solution
♦ Operation overview

This section explains how SRAM support in functions in a controller.

The following sections are included under this topic:

Usage goal

When recovering from a power outage (where a “controlled shutdown” did not occur), the goal is to retain all runtime station data in volatile DRAM (Dynamic RAM) that had changed prior to power lost, that is, since the last station save to the EC-BOS’s NVRAM, also known as “flash” memory. Changes apply to all normally persisted station data, including any changes to components, histories, and alarms.

Simply increasing the flash write frequency (for example upon each value change) is not viable because of eventual damage to flash memory components. Additionally, the latency of flash writes is significant; such writes may not complete if power is lost.

Replacing all flash memory in a controller with SRAM is cost prohibitive; however, a memory caching scheme using SRAM was developed. See the section “SRAM based solution”.

SRAM based solution

The patent-pending SRAM support feature employs both hardware and software, as follows:

- Hardware: SRAM buffer with dedicated capacitor for memory refresh (backup). EC-BOS-8 has “onboard” 256 KB of SRAM, while EC-BOS-3^{AX}, and NPM6E-based controllers (EC-BOS-6^{AX}, EC-BOS-603^{AX} , EC-BOS-645^{AX}) have “onboard” 512 KB of SRAM, backup capacitor plus supporting circuitry. The SRAM option card provides all of this on an installable option card, for EC-BOS-6,-7 series controllers.
- Software: The EC-BOS-8 requires EC-Net 4 v4.1, EC-Net 4 v4.2, or update build EC-Net^{AX}-3.8U1, the `dataRecovery license`, and the `platDataRecovery`module. The other EC-BOSs require EC-Net^{AX}-3.6 or later, the `dataRecovery` license, and the `platDataRecovery` module. The module works with the SRAM buffer to continuously save changes as “deltas”. Essentially, this feature automatically “records” all changes during runtime, for “playback” upon station startup after a power outage.

Providing that the EC-BOS is properly licensed, no other configuration is required for this combination to provide SRAM support. This feature automatically works for the EC-BOS-8, EC-BOS-6^{AX}, or EC-BOS-3^{AX}, as well as for other platforms with the SRAM option card (providing the platDataRecovery software module is installed).

An interface “window” is automatically provided in the station running on the EC-BOS, via the dynamically created “DataRecoveryService” under its PlatformServices container. A special view on this platform service allows verification of operation, and if ever necessary, a means of troubleshooting.

NOTE: The DataRecoveryService includes the “Service Enabled” configuration property, which is true by default. The use case is for the EC-BOS-3^{AX} or NPM6E-based controller (e.g. EC-BOS-603^{AX}) in which you want to disable SRAM support, using only its installed backup battery instead.

Operation overview

Providing the EC-BOS controller is equipped with SRAM and is properly licensed, the platDataRecovery module dynamically creates a “DataRecoveryService” in the running station’s PlatformServices container. This service presides over the SRAM option card, which in turn acts as a buffer for flash memory.

The new service partitions the SRAM into multiple buffers, or data “blocks”. In the initial usage there are 3 blocks. Only one these blocks is ever “active” at any time.

Operation can be described in the following modes:

- Recording (normal operation)
- Station save effects
- Playback scenario (power lost or reboot)

Recording (normal operation)

Immediately after a station is started, any subsequent change in any of its three object spaces (component, alarm, history) is immediately recorded by the DataRecoveryService, writing to the currently active SRAM block. In bytes, each block has a total capacity, a calculated amount of needed overhead, and the current available free area. Exceptions to this recording are noted below.

NOTE:

- Unsaved edits to files, e.g. Nav or Px files (working in the Px Editor) are not recorded, and thus are lost upon any power event if running battery-less. See SRAM does not preserve data or files external to station.
- Slots of objects that have the Non-Critical config flag set are also not recorded. Note by default, for most slots, this new slot config flag (EC-Net 4 v4.10 and later) is cleared. See Non-Critical config flag.

When the active SRAM block becomes full, i.e. its free area is not large enough for a record write, its data is written out (flushed) to a file in the EC-BOS’s flash memory, then that SRAM block is cleared. Concurrently, while this transfer/flush occurs, another SRAM block is activated and used. This multi-threaded buffer method enables recording station changes that would otherwise be lost or blocked.

Over time, many SRAM blocks may fill, where each results in another file in flash on the EC-BOS. These files are automatically named in a numeric sequence with a “.drdb” extension (data recovery database), stored in the station’s file space under the `\dataRecovery` directory. For example, this folder may contain files “1.drdb”, “2.drdb”, “3.drdb”, and so on, with “1.drdb” being the oldest. This sequence becomes important later, when the DataRecoveryService restores (plays back) data.

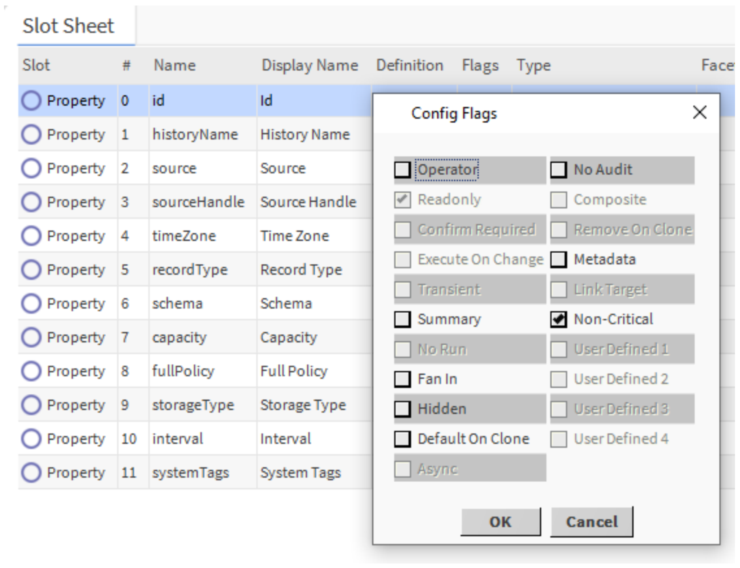
At some point, the collective size of the station’s “.drdb” files may exceed the “persistent capacity” limit used by the DataRecoveryService. This is a “persistent space full” condition. If this occurs, an automatic station save occurs. Effects on the DataRecoveryService are the same as if a station save is manually issued, or if an automatic save from the platform “auto-save” frequency occurs. See “Station save effects”.

Non-Critical config flag

There may be cases where new slots or the data values of those slots is not considered critical, and does not need to be saved by the DataRecoveryService. To accommodate for this, a new “Non-Critical” config flag can be set on any slot (EC-Net 4 v4.10 and later). By default, for most slots, this flag is cleared meaning that it is *not* set.

From EC-Net 4 v4.10 and later, “Non-Critical” config flag can be added to component and history IDs to flag items not necessary to be monitored by data recovery. To flag the history IDs as non critical, navigate to history inside the history space, right-click**Views→ AX Slot Sheet** and set the History ID as **Non Critical**. The setting exists on the history, in history space, this flag cannot be set until the station has been started in a host and the history is created. The configuration is not included in backup. However, the Backup Service configuration can be modified to include histories.

Figure 5 SlotSheet with Config Flag Properties



Slot Sheet

Slot	#	Name	Display Name	Definition	Flags	Type	Face
Property	0	id	Id				
Property	1	historyName	History Name				
Property	2	source	Source				
Property	3	sourceHandle	Source Handle				
Property	4	timeZone	Time Zone				
Property	5	recordType	Record Type				
Property	6	schema	Schema				
Property	7	capacity	Capacity				
Property	8	fullPolicy	Full Policy				
Property	9	storageType	Storage Type				
Property	10	interval	Interval				
Property	11	systemTags	System Tags				

Config Flags

☐ Operator	☐ No Audit
☑ Readonly	☐ Composite
☐ Confirm Required	☐ Remove On Clone
☐ Execute On Change	☐ Metadata
☐ Transient	☐ Link Target
☐ Summary	☑ Non-Critical
☐ No Run	☐ User Defined 1
☐ Fan In	☐ User Defined 2
☐ Hidden	☐ User Defined 3
☐ Default On Clone	☐ User Defined 4
☐ Async	

NOTE: When this flag is set on newly-created slots, the value of such new slots is not saved by the DataRecoveryService. If running "battery-less" and power is lost before the station is saved, these values will be permanently lost.

However, if power is not lost, then the slots are still saved as part of a routine station save. But when the values of the slots change, those changes are not recorded by the DataRecoveryService; they are only saved when a "station save" occurs. Not saving this non-critical data through the DataRecoveryService may offer some performance advantages.

Station save effects

When saving a station running the DataRecoveryService, the following things occur. Note this applies regardless of how the save was issued—for example, a manually invoked command (**Save Station**), or an automatically issued save, e.g. a "persistent space full" condition by the DataRecoveryService, or a station save that occurs as part of a station copy operation, reboot command, or from a "controlled shutdown" while running on backup battery power:

1. The normal station save method is used to capture all changes to all the station's object spaces (components, alarm, history) to flash memory, saved as the files config.bog, alarm.zip and history.zip.
2. Upon a *successful save*, all ".drdb" files holding buffered data are *erased from flash*, as these are no longer necessary.
3. The active SRAM data block is also cleared; however, one block operates in a "reserved state". This is needed to capture any changes that may occur while the station is in the "saving" state. Otherwise, data loss could potentially occur if power was lost during or immediately after a save.

Again, note that buffered data in flash and SRAM is not erased until the save (config.bog) is successfully written to flash. Thus, data recovery records are not lost if a power loss occurs during a save.

Playback scenario (power lost or reboot)

- If power is lost, a battery-less EC-BOS is immediately off, with the last active SRAM block preserved by capacitor charge, plus typically one or more flash-preserved ".drdb" files. This could also occur if battery-equipped controller was unable to successfully complete an orderly shutdown. The EC-BOS remains in this dormant state until power is restored, when the EC-BOS boot process begins normally.
- If a reboot command is given, the last active SRAM block is present, but typically no flash-preserved ".drdb" files—because a reboot typically first issues an "orderly shutdown", starting with at station save. See "Station save effects" for details. The EC-BOS boot process begins normally.

NOTE: A reboot issued from system shell is one exception—no orderly shutdown occurs before such a reboot. There may be .drdb files saved when this reboot happens. If so, these are “replayed” the same as if power is lost.

During station startup, after loading into DRAM the station’s last saved database (config.bog), the DataRecoveryService starts up and detects that recovered data exists in the preserved SRAM block, and also possibly in flash-preserved “.drdb” files.

The DataRecoveryService then performs the following actions:

1. It replays (writes to DRAM) all the flash files in sequence, beginning with the oldest.
2. Once flash records have been replayed, the service replays the records found in the SRAM block, as they are the most recent.

Note that in some cases this playback to DRAM can take a few minutes—this depends on how much data there is to replay. Station startup then completes, and an *immediate save* of the DRAM-restored station to flash (config.bog) occurs. All previous “.drdb” files are erased in flash, as well as SRAM data blocks, and DataRecoveryService operation begins anew. See “Recording (normal operation)” for details.

Chapter 4 Diagnostic tools for the DataRecoveryService

Topics covered in this chapter

The DataRecoveryService provides the following diagnostic tools that may be useful if troubleshooting.

Also included here, a procedure for reformatting the SRAM option card (if necessary).

Station output from the DataRecoveryService

When left at the default “`message`” log level, the DataRecoveryService produces minimal station output related to operation. Primarily, related messages are seen at station startup, especially if following a power loss event or reboot command.

```
MESSAGE [11:18:15 13-Aug-10 EDT][sys.registry] Up-to-date [250ms]
MESSAGE [11:18:20 13-Aug-10 EDT][sys.registry] Loaded [2935ms]
MESSAGE [11:18:29 13-Aug-10 EDT][sys] Baja runtime booted ("/ffs0/niagara")
MESSAGE [11:18:29 13-Aug-10 EDT][sys] Loading "/ffs0/niagara/stations/J6_TestW/
config.bog"...
MESSAGE [11:19:34 13-Aug-10 EDT][sys] Loaded (64809ms)
MESSAGE [11:19:56 13-Aug-10 EDT][alarm.database] Loading...
MESSAGE [11:19:58 13-Aug-10 EDT][alarm.database] Loaded [2196ms, 32 alarms, 104 pages]
WARNING [11:19:58 13-Aug-10 EDT][platDataRecovery.service] data recovery detected,
replaying...
MESSAGE [11:20:11 13-Aug-10 EDT][sys] DataRecoveryService restoration
check complete (18368ms)
MESSAGE [11:20:12 13-Aug-10 EDT][sys] Services Initialized (1010ms)
MESSAGE [11:20:12 13-Aug-10 EDT][sys.mixin] Updated [112ms]
MESSAGE [11:20:14 13-Aug-10 EDT][history.db] Starting async warmup of history
config index...
MESSAGE [11:20:14 13-Aug-10 EDT][history.db] Async history config index warmup
completed in 342 ms.
MESSAGE [11:20:37 13-Aug-10 EDT][web.server] HTTP Server started on port 80
MESSAGE [11:20:38 13-Aug-10 EDT][fox] Service started on port 1911
MESSAGE [11:20:39 13-Aug-10 EDT][sys] Niagara Runtime Environment: 3.6.20
MESSAGE [11:20:39 13-Aug-10 EDT][sys] *** Station Started (27273ms) [153291ms total]
*** niagara>
MESSAGE [11:20:41 13-Aug-10 EDT][sys] Saving station...
MESSAGE [11:20:55 13-Aug-10 EDT][history.db] Saved history archive (4197ms)
MESSAGE [11:21:02 13-Aug-10 EDT][sys] Saved /ffs0/niagara/stations/J202_TestW/
config.bog (19654ms)
```

The example above shows related messages mixed in with other station startup entries.

Using “spy”, you can increase the station’s log level, by setting it to “`trace`” on items:

- `platDataRecovery.manager`
- `platDataRecovery.service`

This produces much more debug information in station output. An example small snippet below reflects trace level output when an SRAM data block “flushes” to a flash file.

CAUTION: Turning trace logging on either of the logs above may produce extremely large levels of output, and should not be left in this log level state. This applies particularly to large stations.

```
TRACE [10:31:12 13-Aug-10 EDT][platDataRecovery.service] External append(history:,
encoded key bytes: appended /J202_TestW/AuditHistory...
TRACE [10:31:12 13-Aug-10 EDT][platDataRecovery.manager] Size of used block exceeds
free space, forcing flush of block
TRACE [10:31:12 13-Aug-10 EDT][platDataRecovery.service] Narcissistic append
(encoded key bytes: AA==, data bytes: 0000b10b000e...
TRACE [10:31:12 13-Aug-10 EDT][platDataRecovery.service] Attempt to append data with
key encoded as bytes: AA== successful
TRACE [10:31:12 13-Aug-10 EDT][platDataRecovery.service] Flush operation successful.
TRACE [10:31:12 13-Aug-10 EDT][platDataRecovery.service] Attempt to append data with
key encoded as bytes: appended /J202_TestW/AuditHistory...
```

You can also increase the log level of other items related to (or used by) the DataRecoveryService, including the following:

- `alarm.dataRecovery` — For logs about the AlarmService’s use of the DataRecoveryService
- `history.critical` — For logs about the HistoryService’s use of the DataRecoveryService

  As well as these additional DataRecoveryService related items:

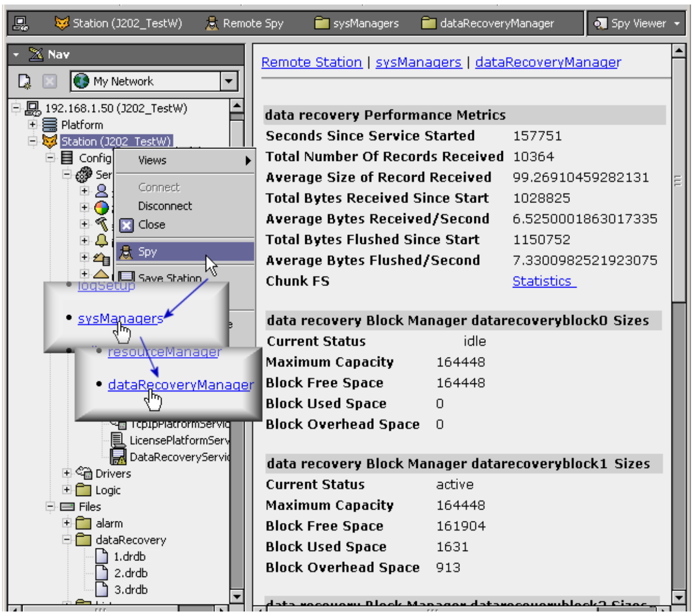
- `sys.critical`
- `sys.critical.load.item` (`item` = `add`, `change`, `facetsChange`, `flagsChange`, `recategorize`, `remove`, `rename`, `reorder`)
- `sys.critical.progObj`
- `sys.critical.save.item` (`item` = `add`, `change`, `facetsChange`, `flagsChange`, `recategorize`, `remove`, `rename`, `reorder`)

NOTE: The same caution applies as before about trace level on these items—some may produce very large levels of station output.

Spy page for the DataRecoveryService

An extensive “spy” page on a station’s DataRecoveryService is available in EC-Net 4 Pro at the following location: **Spy→sysManagers→DataRecoveryManager**.

Figure 6 Spy page for DataRecoveryService



Remote Station | sysManagers | dataRecoveryManager

data recovery Performance Metrics	
Seconds Since Service Started	157751
Total Number Of Records Received	10364
Average Size of Record Received	99.26910459282131
Total Bytes Received Since Start	1028825
Average Bytes Received/Second	6.5250001863017335
Total Bytes Flushed Since Start	1150752
Average Bytes Flushed/Second	7.3300982521923075
Chunk FS	Statistics

data recovery Block Manager datarecoveryblock0 Sizes	
Current Status	idle
Maximum Capacity	164448
Block Free Space	164448
Block Used Space	0
Block Overhead Space	0

data recovery Block Manager datarecoveryblock1 Sizes	
Current Status	active
Maximum Capacity	164448
Block Free Space	161904
Block Used Space	1631
Block Overhead Space	913

This spy page, partially shown in above, provides numerous statistics about station data being backed up by the service. Details about these statistics are outside the scope of this document.

Reformat the data recovery storage in a controller

When using a "serial shell" connection to a controller with SRAM, a special "alternative boot" option is available. A sub-option allows you to reformat the onboard SRAM installed in the EC-BOS.

You must be near the EC-BOS to make this direct serial shell connection, and also be able to easily power it off and on as needed.

NOTE: In most cases the following reformat procedure will never be necessary. When shipped, SRAM is already properly formatted and does not contain any stored data. This is true as well for models that have the installed SRAM option card. However, in rare cases it may be necessary to reformat SRAM using this procedure.

Reformatting SRAM in a controller

This procedure describes using a terminal emulation program to reformat controller models that have onboard SRAM. Note that the steps for reformatting models with an SRAM option card installed are basically the same as those listed here.

Prerequisites: Be sure to backup the controller to your PC using platform tools prior to reformatting the SRAM.

NOTE: The **Alternate Boot** menu options will differ depending on whether the controller has onboard SRAM (or other data recovery storage option), and depending on which EC-Net release is running on the controller.

Step 1 Power-off the controller, and install the serial shell jumper (if not already installed).

Connect the necessary serial cable between the controller and your PC's COM port.

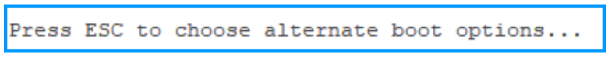
Step 2 On your PC, start a terminal emulation program, for example HyperTerminal, and open a previously saved setup for EC-BOS communications, using that COM port. Settings are listed here.

- Bits per second: 115200
- Data bits: 8
- Parity: None
- Stop bits: 1
- Flow control: Hardware

Step 3 With your HyperTerminal session active, apply power to the controller.

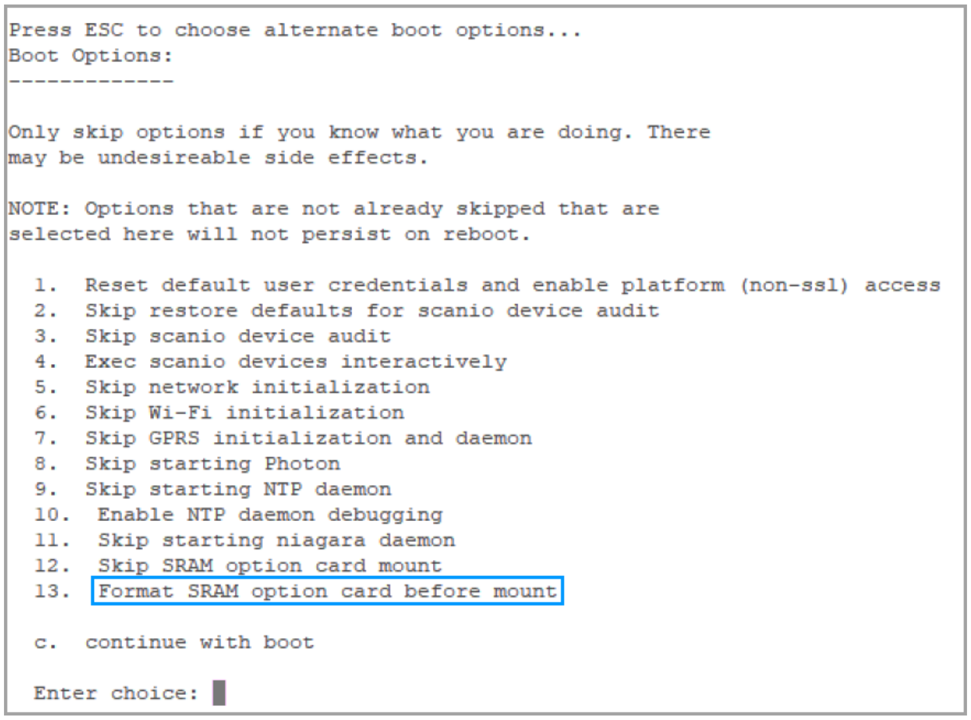
After a short delay, text should appear in the HyperTerminal window similar to the example shown here:

NOTE: You must wait until you see the following prompt, and then press **ESC** .



```
Press ESC to choose alternate boot options...
```

The **Alternate Boot** menu displays numbered options. Shown here is the menu for an EC-BOS-8 running EC-Net 4 v4.7.



```
Press ESC to choose alternate boot options...
Boot Options:
-------------

Only skip options if you know what you are doing. There
may be undesireable side effects.

NOTE: Options that are not already skipped that are
selected here will not persist on reboot.

  1.  Reset default user credentials and enable platform (non-ssl) access
  2.  Skip restore defaults for scanio device audit
  3.  Skip scanio device audit
  4.  Exec scanio devices interactively
  5.  Skip network initialization
  6.  Skip Wi-Fi initialization
  7.  Skip GPRS initialization and daemon
  8.  Skip starting Photon
  9.  Skip starting NTP daemon
  10.  Enable NTP daemon debugging
  11.  Skip starting niagara daemon
  12.  Skip SRAM option card mount
  13.  Format SRAM option card before mount

  c.  continue with boot

  Enter choice: █
```

NOTE: The options visible for your controller may differ from those shown in this example. See the earlier note in this procedure for more details.

Step 4 At the prompt, "`Enter choice`", type the appropriate number for the option to **Format SRAM option card before mount** (for the example menu shown above, you would type "`13`") and press **ENTER**.

The **Alternate Boot** menu reappears, without the selected option.

Step 5 Type `c` (to continue with boot) and press **ENTER**

The text, "`continuing boot...`" appears along with other typical boot messages for TCP/IP initialization.

When the boot completes (at the login prompt), there should be messages indicating that a format operation occurred.

Step 6 When the "login" prompt appears, power off the controller.

If needed (e.g., if reformatting SRAM in an EC-BOS-7AX with SRAM option card), at this point you could remove the SRAM option card to use in another controller.

NOTE: If you remove an SRAM option card from your EC-BOS-7AX controller and do not reinstall it, be sure to uninstall the platDataRecovery module from the controller, using the platform **Software Manager**.

Step 7 Exit from the HyperTerminal program, selecting **Save** if you wish to reuse this setup again.