

# **Block Interface Guide**

**For**

## **DeltaV Connect™ Solution for Bailey® Systems**

**Command Series / NETWORK 90® / INFI 90®**

NETWORK 90 and INFI 90 are registered trademarks of ABB (Formerly Bailey Controls Company).

DeltaV is a mark of Emerson Process Automation (Formerly Fisher-Rosemount Systems, Inc.)

All other marks are the property of their respective owners.

© 1999-2008 Emerson Process Automation (Formerly Fisher-Rosemount Systems, Inc.). All rights reserved.

|        |  |    |
|--------|--|----|
| 1      | Introduction.....  | 4  |
| 2      | Functional Description.....                                  | 4  |
| 3      | Installation .....   | 6  |
| 3.1    | Uninstalling Bailey Connect software .....                   | 6  |
| 4      | Bailey Communication Overview .....                          | 7  |
| 5      | Translation of Engineering Units from Bailey to DeltaV ..... | 9  |
| 6      | Redundancy .....   | 12 |
| 7      | DeltaV Connect Bailey – Blocks.....                          | 14 |
| 7.1    | Analog Input Loop (AIL) .....                                | 14 |
| 7.2    | Analog Output Loop (AOL) .....                               | 16 |
| 7.3    | Block Data (BLK).....  | 18 |
| 7.3.1  | Reading A Bailey Block .....                                 | 20 |
| 7.3.2  | Tuning A Bailey Block.....                                   | 20 |
| 7.3.3  | Reading The Entire Bailey Configuration .....                | 20 |
| 7.3.4  | Changing Bailey Controller Module Modes.....                 | 21 |
| 7.3.5  | Writing Bailey Controller Blocks.....                        | 22 |
| 7.3.6  | Modifying Bailey Controller Blocks .....                     | 22 |
| 7.3.7  | Deleting Bailey Controller Blocks.....                       | 22 |
| 7.3.8  | Configuration Command Processing.....                        | 23 |
| 7.4    | Block Data Values (BLKVAL).....                              | 28 |
| 7.5    | Data Acquisition Digital (DADIG) .....                       | 29 |
| 7.6    | Data Acquisition Analog (DANG).....                          | 32 |
| 7.7    | Device Driver (DD).....                                      | 36 |
| 7.8    | Digital Input Loop (DIL) .....                               | 38 |
| 7.9    | Digital Output Loop (DOL) .....                              | 39 |
| 7.10   | Interface Definition Block (IDB) .....                       | 40 |
| 7.10.1 | SCSI Communication .....                                     | 45 |
| 7.11   | Multi-State Device Driver (MSDD).....                        | 48 |
| 7.12   | Multiplex CIU (MUXCIU) .....                                 | 51 |
| 7.13   | Poll Any Block (POUT) .....                                  | 53 |
| 7.14   | Remote Control Memory (RCM).....                             | 55 |
| 7.15   | Remote Motor Control (RMC) .....                             | 57 |
| 7.16   | Remote Manual Set Constant (RMSC) .....                      | 60 |
| 7.17   | Module Status (STAT).....                                    | 61 |
| 7.18   | Station PID Control (STN) .....                              | 63 |
| 7.19   | Text Selector (TXT) .....                                    | 66 |
| 8      | Simulation of the Bailey System.....                         | 67 |
| 9      | Troubleshooting Hints .....                                  | 72 |
| 9.1    | Validating State Of Individual DVCBailey Blocks.....         | 73 |

|     |  |    |
|-----|--|----|
| 9.2 | No Communication.....  | 73 |
| 9.3 | Appears To Be Communicating But No Data is Being Received .....  | 74 |
| 9.4 | Not All Blocks Are Receiving Data.....                           | 74 |
| 9.5 | Cannot Export or Control Data Within Bailey.....                 | 75 |
| 9.6 | BLK Block Takes A Long Time To Complete A Read / Tune .....      | 75 |
| 9.7 | DVCBailey Will Not Time Sync Bailey .....                        | 75 |
| 9.8 | Both App Stations Become Active Upon Failure Of DeltaV ACN ..... | 76 |
| 9.9 | Redundant App Stations SCSI CIU Addresses Don't Match.....       | 76 |

## 1 Introduction

The RoviSys Company working with Emerson Process Management have developed a tightly integrated interface between the DeltaV and Bailey DCS that promotes future plant upgrades using the DeltaV system. This interface is called the DeltaV Connect™ Solution for Bailey® Systems (DVCBailey). DVCBailey allows easy, user-intuitive replacement of Bailey operator consoles with NT based DeltaV consoles. It also provides an open migration path from the Bailey control platform to the DeltaV system as plant time, resources and schedule permits.

DVCBailey enables seamless DeltaV DCS access to the Bailey Controls Command Series, NETWORK 90 and INFI 90 Distributed Control Systems. The DVCBailey can communicate at the network level via the appropriate Computer Interface Unit (NCIU0X), Plant Loop to Computer Interface (INPCIOX), Infi-net to Computer Interface (INICI01) or, within a single Process Control Unit, via a Computer Interface Command (CIC01), Serial Port Module (NSPM01, IMSPM01) and Computer Port Module (CPM02/03). System integrity, functionality, and data throughput is maintained by utilizing standard exception reporting techniques. Bandwidth improvements are realized over existing Bailey use of these same hardware interfaces by implementing dual channel capability and redundant interfaces.

This document is intended for individuals who are familiar with the Bailey configuration principals in terms of the types and how to access exception report blocks. An understanding of how to use DeltaV Process Explorer, Control Studio and Graphics Studio is also required.

Throughout this document the DeltaV Connect™ Solution for Bailey® Systems will be referenced as DVCBailey or DVC. The Bailey system is referenced as *BLY* which is defined to include Command Series, NETWORK 90 and INFI 90 systems.

## 2 Functional Description

DVCBailey runs within a standard DeltaV application station. It is implemented as a collection of DeltaV function blocks that are specific to each type of exception report block found within a Bailey system. A special block called the Interface Definition Block (IDB) is configured to setup the communication port(s) used to access the Bailey interface, define required update rates and report various statuses of the communication channels. All other DVCBailey blocks are linked to an IDB which provide the necessary information it needs to manage communication with the Bailey system. The various other DVCBailey blocks linked to the IDB contain the address within Bailey where the exception report block is located. The IDB uses this address and block type to determine how the point is established and managed within the Bailey interface. As data is received from the Bailey interface for each of the established

points, it is parsed and copied to the appropriate attributes within the corresponding DeltaV DVCBailey block. This implementation has several advantages:

- It is extremely intuitive to Bailey users.
- Point capacity is not limited by type of points utilized by the plant but only by the type of Bailey interface available (typically up to 10,000 tags). Plant graphics and control faceplate generation uses existing DeltaV applications and maintains plant tag names that the Bailey user is very familiar with.
- Alarm levels are set in real time by data received from the Bailey system (this eliminates the need to maintain alarm level settings in two systems).
- It includes ability to tune Bailey control loops from DeltaV consoles.
- It supports redundant Bailey interfaces or redundant communication channels to a single Bailey interface.

### **3 Installation**

For installation instructions, please consult the document entitled “DVC Bailey Installation” included with the DVCBailey installation CD. Electronic versions of that document are also available on the CD.

#### **3.1 Uninstalling Bailey Connect software**

Use the Windows “Add or Remove Programs” feature available from Control Panel to uninstall the Bailey Connect software from the Professional Plus and Application Station. Note that the interface primitive blocks and templates can only be removed by creating a new database.

## 4 Bailey Communication Overview

Bailey utilizes a communication technique for data exchange called exception reporting. Exception reporting means the data is sent when it has changed significantly or a maximum time has expired since the last time it was sent. All exception reported data is made available to the rest of the system via a set of exception report related function blocks that are matched by an equivalent DVCBailey block. Bailey produces several RS232 serialbased devices called Computer Interface Units (CIUs) that receive exception report data. The DVCBailey interface provides a single driver that can communicate to the following Bailey CIUs:

| Bailey Interface | Max. Number of Blocks | Exception Reporting | Control Operation |
|------------------|-----------------------|---------------------|-------------------|
| NSPM01*          | 500                   | No                  | No                |
| IMSPM01*         | 500                   | No                  | No                |
| IMCPM02*         | 500                   | No                  | No                |
| IMCPM03          | 3000                  | Yes                 | Yes               |
| NCIC01           | 500                   | Yes                 | Yes               |
| NCIU01           | 500                   | Yes                 | Yes               |
| NCIU02           | 2,500                 | Yes                 | Yes               |
| NCIU03           | 5,000                 | Yes                 | Yes               |
| NCIU04           | 10,000                | Yes                 | Yes               |
| INPCI01          | 500                   | Yes                 | Yes               |
| INPCI02          | 5,000                 | Yes                 | Yes               |
| INICI01          | 10,000                | Yes                 | Yes               |
| INICI12**        | 10,000                | Yes                 | Yes               |
| INICI03**        | 10,000                | Yes                 | Yes               |

\* These interfaces do not support exception reporting. Data collection is limited to the DVCBailey block, POUT, which polls for block output values.

\*\* When using these interfaces, the ABB Bailey semAPI software environment is not required. The INICI12 and INICI03 only support one serial port. When using the INICI03 interface, the SCSI or serial port connection are supported. See the Interface Definition Block for details on utilizing SCSI communication.

The following DVCBailey blocks and associated Bailey exception report block types are supported:

| <b>DVCBailey Block Name</b> | <b>Bailey Block Name</b>                          | <b>F.C. Number</b> |
|-----------------------------|---|--------------------|
| AOL                         | Analog Input / Loop – AIL                         | 26, 121            |
| AIL                         | Analog Output / Loop – AOL                        | 30, 70, 158        |
| BLK                         | Configuration and tuning any block specifications | any function code  |
| BLKVAL                      | Configuration and tuning any block specifications | any function code  |
| DADIG                       | Data Acquisition Digital – DADIG                  | 211                |
| DANG                        | Data Acquisition Analog – DAANG                   | 177                |
| DD                          | Device Driver – DD                                | 123                |
| DOL                         | Digital Input / Loop – DIL                        | 42, 122            |
| DIL                         | Digital Output / Loop – DOL                       | 45                 |
| MSDD                        | Multi-State Device Driver – MSDD                  | 129                |
| POUT                        | Poll Value  | any block          |
| RCM                         | Remote Control Memory – RCM                       | 62                 |
| RMC                         | Remote Motor Control – RMC                        | 136                |
| RMSC                        | Remote Manual Set Constant – RMSC                 | 68                 |
| STAT                        | Module Status                                     | any module         |
| STN                         | Control Station – STN                             | 21, 22, 23, 80     |
| TXT                         | Text Selector – TEXT                              | 151                |

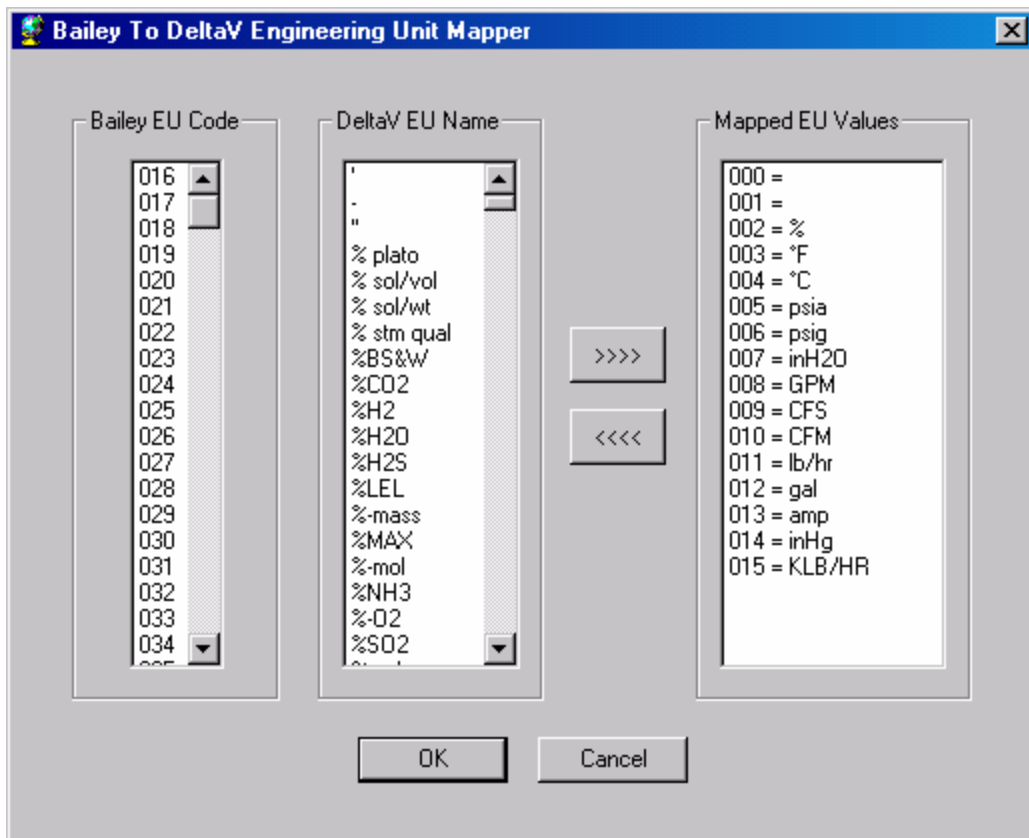


## 5 Translation of Engineering Units from Bailey to DeltaV

The DVCBailey interface can be configured to automatically translate Bailey engineering unit (EU) codes to corresponding DeltaV engineering descriptors. The Bailey EU codes are received as part of the normal exception reporting mechanisms for those Bailey blocks that communicate analog values. DVCBailey can automatically translate these codes to the appropriate DeltaV engineering descriptors associated with the OUT\_SCALE attribute.

Two steps are required to configure automatic translation of Bailey EU codes. The first is to check the “Enable EU Mapping” option found in the IDB block OPTION attribute.

The second step is to configure the EU mapping using the EUMAP program installed in the “DeltaV\bin” directory. This program allows the Bailey EU codes to be mapped to equivalent DeltaV EU descriptors. The EUMAP program displays the following configuration screen:



The first list box contains Bailey EU codes that have not been mapped. Bailey has predefined definitions for the first 16 EU codes (0-15). The EU map for those codes has been pre-configured and loaded in the Pro+ database library and App station DVCBailey interface. The remaining Bailey EU codes are intended for customization on

a plant by plant basis. If used by a plant site, these codes must be mapped for each specific plant.

The second list box contains DeltaV EU names that are available to be mapped to a Bailey EU code. The EUMAP program automatically fills this list box from EU values defined within the following files:

```
Delta\Fhx\engUnits.fhx
Delta\Fhx\engUnitsBaileyPredefined.fhx
Delta\Fhx\engUnitsBaileyCustom.fhx
```

The “engUnits.fhx” file contains the definition of all standard DeltaV EU descriptors. The “engUnitsBaileyPredefined.fhx” file contains additional EU descriptors that are required for the predefined Bailey EU codes not defined within the standard DeltaV “engUnits.fhx” file. The “engUnitsBaileyCustom.fhx” file is intended to allow plant site specific EU descriptions to be defined that can be mapped to the custom Bailey EU codes in the range 16 and greater that are not already defined with the standard DeltaV “engUnits.fhx” file. If a DeltaV EU descriptor has not been defined for a custom Bailey EU code it should be defined within the “engUnitsBaileyCustom.fhx” file. One of two methods can be used to customize this file.



The first method is to use DeltaV Explorer to define custom engineering units. Double click on Setup->Engineering Units. After adding the custom engineering units, export engineering units to Delta\Fhx\engUnitsBaileyCustom.fhx. Copy this file to the App Station Delta\Fhx directory.

The second method, using notepad you can open the “engUnitsBaileyPredefined.fhx” file and copy the last line to the clipboard. Open the “engUnitsBaileyCustom.fhx” file and paste the contents of the clipboard at the end of the file. The following line will appear:

```
ENGINEERING_UNIT NAME="%NH3" INDEX=45044 { }
```

Modify the EU descriptor defined between the quotes and increase the index definition by one. After defining as many new EU descriptors required for the plant site, save the file and import it using the DeltaV Process Explorer Select File -> Import -> Standard DeltaV Format -> Delta\Fhx\engUnitsBaileyCustom.fhx IMPORT button. You must also perform a setup data download to the Pro+ and copy the modified engUnitsBaileyCustom.fhx file to the App Station Delta\Fhx\ directory. Note that if you have defined and imported other custom EU descriptors in other files make sure the index numbers are unique with respect to those added to engUnitsBaileyCustom.fhx. EUMAP can include the other custom EU descriptors by including name (and path) as program arguments when running EUMAP. See the DeltaV books online for additional information regarding customizing engineering units.

The third EUMAP list box shows the currently mapped Bailey EU codes with their equivalent DeltaV EU descriptor.

To map a Bailey EU code click on it in the first list box, click on its associated DeltaV EU descriptor in the second list box and then click on the  button. To un-map a Bailey EU code click on it in the third list box and click on the  button. To quit EU mapping without saving any changes click on the cancel button. Otherwise, when you click on the ok button EUMAP exits, saving the mapped values in a file called "Dvc\_Bly\_EuMap.ini". This file is maintained in the Delta\DVDData\ABB.

It is **EXTREMELY IMPORTANT** to note a few file locations when configuring the automatic DVCBailey EU mapping feature. For your convenience the EUMAP program is installed in the Delta\Bin directory on both the Pro+ and Application station. Likewise, the engUnits.fhx, engUnitsBaileyPredefined.fhx and engUnitsBaileyCustom.fhx files are also installed on both stations in the Delta\Fhx directory. This will allow configuration of the DVCBailey EU map using either DeltaV PC station.

*If the definition of additional DeltaV EU descriptors is not required, it is recommended to utilize the EUMAP program on the App station.*

*If additional EU descriptors are required they must be defined in the engUnitsBaileyCustom.fhx file on the Pro+. Remember that you also must import them using the DeltaV Process Explorer import feature and you should copy the updated engUnitsBaileyCustom.fhx file to the App station. After running EUMAP on the Pro+ make sure you always copy the "Dvc\_Bly\_EuMap.ini" file to the App Station. This file is maintained by EUMAP in the Delta\DVDData\ABB directory.*

*Remember that DVCBailey automatic translation of Bailey EU codes is enabled by checking the "Enable EU Mapping" option found in the IDB block OPTION attribute.*

## 6 Redundancy

Two levels of redundancy are supported. The IDB block SCEHME, PORT\_A and PORT\_B attributes can be used to setup redundant communication channels to a single Bailey CIU (assuming the particular CIU model supports two communication channels). These same IDB block attributes can be used to setup communication with redundant Bailey CIUs. Both levels provide “warm” failover. Warm failover means the Bailey CIU database does not have to be re-established if a failover happens and only a temporary interruption of data flow from the Bailey system occurs.

When dual CIU redundancy is used, it is important to pay attention to the Bailey addresses assigned to the redundant CIUs. When the DVCBailey database does not utilize DVCBailey AOL and DOL blocks, it is best to assign unique Bailey addresses to the redundant CIUs. When DVCBailey AOL and DOL blocks are utilized, the Bailey addresses assigned to the redundant CIUs should be identical. This setup makes it easy for Bailey controllers and consoles to read the DVCBailey AOL and DOL blocks since they can be addressed to a single location. On startup, the IDB block determines the CIU addresses and if identical will establish the database in both, but only command one of the CIUs online. If that CIU fails, the IDB block will command the other CIU online.

The second level of redundancy is Redundant Application Stations. Redundant App Stations function with the two levels of IDB CIU redundancy just mentioned. When a CIU fails or both of its communication channels fail, the Standby App Station will take on the Active role assuming its CIU communication is healthy. The previously Active App Station will transition to the Standby role. The time it takes for the old Active App Station to come back as the Standby is variable and based on the size of the DVCBailey database. The minimum time is approximately 2 minutes. The IDB block MESSAGE attribute (viewable from the IDB faceplate) and STANDBY\_STATUS attribute will indicate when the Standby is not available. It will also indicate if an available Standby App Station has problems with communication on either of its PORT\_A or PORT\_B channels. Alarms are associated with the status of the standby App Station and its CIU communication ports. These are new alarms defined in the Universal version of the DVCBailey interface and must be enabled when redundant App Stations are being utilized. For users upgrading from the non-universal version of the DVCBailey interface, they must redefine the IDB block using the updated IDB module template to gain access to these alarms.

If an Active App Station experiences any other failure besides IDB detection of bad CIU communication, it will stop the DeltaV services and not come back as the Standby App Station. The new Active App Station will report the Standby is not available in the IDB MESSAGE attribute (viewable from the IDB faceplate). More detailed information is available from DeltaV Diagnostics. The reason for the failure must be determined from the DeltaV logs and Windows Event viewer. The failure must be corrected before restarting the PC so it will come back as the Standby App Station.

There are many events that can lead to a "switchover". How long it takes is really dependent on what type of event occurs. After the event occurs, the period of time it takes to get data flowing again is directly related to whether or not the Bailey CIUs attached to the Active and Standby App Stations are configured for unique Bailey addresses or the same address. For the case in which they are configured for unique addresses, data begins to flow immediately after the switch over occurs. For the case in which they are at the same address, the data cannot begin to flow immediately because the newly Active App must wait to command its CIU online because it knows that the former Active App CIU can take a few seconds before it automatically takes itself off the Bailey loop. The ability to allow redundant Apps to setup their CIUs at the same address is a great feature when the DVCBailey AOL and DOL blocks are being utilized. Note that this was not supported by the former RoviSys DVRapsMan solution. It allows the receivers of the data (Bailey controllers) provided by the DVCBailey AOL and DOL blocks to only look to one address (CIU ring, node and module two address). Without this feature, special logic must be configured in the Bailey system to receive the data from two separate addresses and make a decision as to which one to use based on the quality of each.

Now lets talk about the longest case scenario that leads up to a switchover. Assume the Active App CIU dies. The Bailey interface communication logic does 3 retries on failed messages with 7 second time out for each retry. The timeout length is based on the worse case theoretical maximum time it could take for the Bailey CIU to response to a poll class message which is 6 seconds. Without getting into too much Bailey CIU communication theory, let's just say that the DVCBailey interface uses a mechanism called keyed messages for the poll class messages that can take up to 6 seconds to reply. This mechanism allows those types of messages to be keyed which means the CIU is working on getting their reply while it is free to process other messages such as the exception report reads. So if the CIU dies, it will take 21 seconds (three retries at 7 seconds each) for the interface to declare the CIU dead and force a switchover. At that point the switchover happens really fast (less then 1 second). When uniquely addressed CIUs are being used, the data from the former standby App begins to flow immediately after the switchover occurs. When same address CIUs are being used, there will be a small delay (less than 5 seconds) after switchover before the data begins to flow again.

Probably the shortest case scenario that leads to switchover is the App Station PC dies. Under this condition, the switchover happens really fast (less then a second) and the new Active begins providing data immediately (when using unique CIU addresses).

DeltaV App Station redundancy is not compatible with the RoviSys DVRapsMan application and solution. When utilizing DeltaV App Station redundancy, the DVRapsMan software must be uninstalled. The associated network switches originally installed as part of the DVRapsMan solution must also be decommissioned.

## 7 DeltaV Connect Bailey – Blocks

The following sub-sections define the DVCBailey blocks available for interfacing with Bailey systems. For ease of access, the blocks are presented in alphabetical order. Note that an Interface Definition Block (IDB) *must* be configured for the DVCBailey to successfully communicate with the Bailey system. This block declares an instance of the DeltaV Bailey driver and its operational characteristics. The remaining DVCBailey blocks provide access to the various Bailey exception report blocks, polling outputs of non-exception reported blocks, system status, and configuration reading and tuning of any Bailey block in the system. All of these blocks have attributes to configure the Bailey addressing information. The other attributes are used to customize operation of each block and provide data place holders for the data received from Bailey. These place holder attributes, in turn, provide a linkage into DeltaV console graphics and faceplates. Note also, most DVCBailey blocks include pre-built faceplates. Please reference the DVC-Bailey help file for details on the data provided by these faceplates.

### 7.1 Analog Input Loop (AIL)

The AIL DeltaV block is used to retrieve the exception reported output from a Bailey Analog Output / Loop block (function code 30), Analog Point Definition (function code 70) and Enhanced Analog Point Definition (function code 158). Italicized attribute names indicate the values received from Bailey. This block is used to receive analog values from the Bailey system to the DeltaV system.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02).

| ATTRIBUTE  | TYPE                    | DESCRIPTION  |
|------------|-------------------------|--|
| IDB_ID     | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE    | String                  | Message describing state of the block operation.   |
| RING       | 8 Bit Unsigned Integer  | Bailey ring address.   |
| NODE       | 8 Bit Unsigned Integer  | Bailey node address.   |
| MODULE     | 8 Bit Unsigned Integer  | Bailey module address.   |
| BLOCK      | 16 Bit Unsigned Integer | Bailey block address.  |
| DISCONNECT | 8 Bit Unsigned Integer  | When true requests that the data for this block be established in the Bailey interface but not connected. This stops data flow unless it goes in or out of an alarm condition. The functionality of this attribute is a possible future feature that is not currently implemented. |
| ALARM_HYS  | Floating Point          | The amount the alarm value must return within the alarm limit before the associated active alarm condition clears.   |

|            |                            |   |
|------------|----------------------------|---|
| HI_HI_LIM  | Floating Point             | The setting for the alarm limit used to detect the high high alarm condition.   |
| HI_LIM     | Floating Point             | The setting for the alarm limit used to detect the high alarm condition. Writing a new value to this attribute automatically tunes the alarm limit of the associated Bailey block.                  |
| LO_LIM     | Floating Point             | The setting for the alarm limit used to detect the low alarm condition. Writing a new value to this attribute automatically tunes the alarm limit of the associated Bailey block.                   |
| LO_LO_LIM  | Floating Point             | The setting for the alarm limit used to detect the low low alarm condition.   |
| EU_CODE    | 8 Bit Unsigned Integer     | Bailey engineering units code which is value of; FC 30 S2, FC 70 S3 or FC 158 S3.   |
| QUALITY    | 8 Bit Unsigned Integer     | Quality of the data (0 = good, 1 = bad)   |
| OUT        | Floating Point With Status | Current output value and status received from Bailey (note 1).  |
| HI_HI_ACT  | Alarm                      | High high alarm active indicator.   |
| HI_ACT     | Alarm                      | High alarm active indicator.  |
| LO_ACT     | Alarm                      | Low alarm active indicator.   |
| LO_LO_ACT  | Alarm                      | Low low alarm active indicator.   |
| OUT_SCALE  | Scaling                    | High and low OUT limits (received from Bailey) along with DeltaV engineering units. Also includes declaration of digits to the right of the decimal point associated with OUT for display purposes. |
| OUT_HI_LIM | Floating Point             | OUT high limit (duplicated in OUT_SCALE).   |
| OUT_LO_LIM | Floating Point             | OUT low limit (duplicated in OUT_SCALE).  |
| SIM_TYPE   | Named Set                  | Defines type of simulated value. Simulation is active when the IDB SCHEME attribute is set to the "Simulated Interface" selection.  |
| SIM_TIME   | 32 Bit Unsigned Integer    | Defines simulation time in milliseconds. Consult "Simulation of Bailey System" for additional details.  |

## 7.2 Analog Output Loop (AOL)

The AOL DeltaV block is used to send an exception reported output generated by value changes to the block's input. The exception reports can be received by a Bailey console tag, a Bailey Analog Input / Loop block (function code 26) and Analog Input / Inifinet blocks (function code 121). The block will automatically generate the appropriate quality, and alarming status based on the value presented at the block input. This block is used to send analog values from the DeltaV system to the Bailey system.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM, CPM02 & CPM03) and computer interface command series (CIC).

| ATTRIBUTE  | TYPE                       | DESCRIPTION  |
|------------|----------------------------|--|
| IDB_ID     | 32 Bit Unsigned Integer    | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE    | String                     | Message describing state of the block operation.   |
| BLOCK      | 16 Bit Unsigned Integer    | Block number to establish this point at within the Bailey interface (see note 1).  |
| IN         | Floating Point With Status | Input value to be exception reported to Bailey.  |
| OUT        | Floating Point With Status | Last input value reported to Bailey.   |
| SIG_CHANGE | Floating Point             | Amount that IN must change, cycle to cycle before it will be reported to Bailey. Expressed as a percentage of the OUT_SCALE span (high limit minus low limit). |
| MAX_TIME   | 16 Bit Unsigned Integer    | If IN never changes significantly, it will be reported at the maximum time interval (seconds) defined by this attribute.                                       |
| OUT_SCALE  | Scaling                    | High and low OUT limits along with engineering units code. Also includes number of digits to the right of the decimal point associated with OUT.               |
| EU_CODE    | 8 Bit Unsigned Integer     | Bailey engineering units code to use when establishing this point.   |
| ALARM_HYS  | Floating Point             | The amount the alarm value must return within the alarm limit before the associated active alarm condition clears.   |
| HI_ACT     | Alarm                      | High alarm active indicator.   |
| HI_LIM     | Floating Point             | The setting for the alarm limit used to derive the high alarm condition.   |
| LO_LIM     | Floating Point             | The setting for the alarm limit used to derive the low alarm condition.  |
| LO_ACT     | Alarm                      | Low alarm active indicator.  |

Notes:

- 1.) Make sure the BLOCK number attribute is unique with respect to the other AOL, DOL, ORCM, ORMSC and OSTN DeltaV blocks associated with the same Bailey Interface Definition block. The BLOCK number attribute must also be defined within the range of 1 to maximum number of allowed



outputs set up within the associated DeltaV Interface Definition block (see IDB MAX\_OUTPUTS attribute). The Bailey system will receive data from this block at the ring and node address of the Bailey CIU interface, module address two and block number defined by the BLOCK attribute.

### 7.3 Block Data (BLK)

The BLK DeltaV block provides Bailey function block configuration capabilities. It supports reading Bailey configuration, tuning specifications, changing module modes, writing new blocks, modifying existing blocks, deleting blocks and saving and restoring module configurations. The BLK block has a faceplate associated with it called "TUNE\_FP" that supports all of these features. The BLK block should be configured using its predefined module template that includes a reference to its faceplate (TUNE\_FP). A dynamo is also available to provide a graphical click point to the faceplate. By default the BLK block loads and saves Bailey configuration files within the DVDData\ABB directory. When DeltaV Application Station redundancy is enabled, the interface will automatically copy the files in the DVDData\ABB directory to the equivalent directory on the Standby App Station. This file synchronization occurs every 60 seconds. If the user decides to maintain Bailey configuration files in any directory other than DVDData\ABB, it is the responsibility of the user to copy those files to the Standby App Station.

Restrictions: None, this DeltaV block can be utilized with all Bailey interface types and blocks.

| ATTRIBUTE | TYPE                        | DESCRIPTION   |
|-----------|-----------------------------|---|
| IDB_ID    | 32 Bit Unsigned Integer     | Associated Interface Definition Block ID (See IDB block).   |
| MESSAGE   | String                      | Message describing state of the block operation.  |
| BLY_OUT   | 32 Bit Unsigned With Status | Output used as input to an associated BLKVAL block which is used to change tunable Bailey block specifications.   |
| RING      | 8 Bit Unsigned Integer      | Bailey ring address.  |
| NODE      | 8 Bit Unsigned Integer      | Bailey node address.  |
| MODULE    | 8 Bit Unsigned Integer      | Bailey module address.  |
| BLOCK     | 16 Bit Unsigned Integer     | Bailey block address.   |
| READ      | 8 Bit Unsigned Integer      | Set this attribute to request the specifications for the addressed block to be read. The driver will reset this attribute when the read request is completed.   |
| NEXT      | 8 Bit Unsigned Integer      | Set this attribute to request a read of the specifications for the next block from the current one. The driver will reset this attribute when the next block read request is completed and update the BLOCK attribute to reflect the next block read. |
| TUNE      | 8 Bit Unsigned Integer      | Set this attribute to request a tune operation for any specifications that have been changed after reading them. The driver will reset this attribute when the tune read request is completed.  |
| DEFAULT   | 8 Bit Unsigned Integer      | Set to request a read of the default specification settings for the function code indicted by the FC attribute. The driver will reset this attribute when the read default request is completed.  |
| PASSWORD  | String                      | Password to unlock the OPERATION attribute  |

|             |                        |   |
|-------------|------------------------|---|
| FILE_TYPE   | 8 Bit Unsigned Integer | and some commands supported by the COMMAND attribute. Will indicate "???????" when locked and "-----" when unlocked. See note 3 for additional details concerning this attribute.<br>When reset the configuration file type used by the save and load command is the DVC-Bailey "C90" file type format. When set the ABB Bailey "CFG" file type format is used. |
| CMD         | String                 | Supports a variety of Bailey controller configuration activities using simple text based commands. See user configuration command processing for usage details.   |
| CMD_ACT     | 8 Bit Unsigned Integer | Set when a text based configuration command is active, otherwise reset when command is not active or completed.   |
| OPERATION   | 8 Bit Unsigned Integer | Commands Bailey controllers to different operating modes per the following values:<br>1 = request controller software reset<br>2 = request configure mode<br>3 = request execute mode<br>4 = request configuration initialization<br>The driver will reset this attribute to zero when the module operation request has been completed.                         |
| MODE_T      | String                 | Returns the current operating mode (Execute, Configure, Error) of the module addressed by the MODULE attribute.   |
| WRITE       | 8 Bit Unsigned Integer | Set to request a block to be written or modified. The driver will reset this attribute when the block write or modification request is completed.   |
| DELETE      | 8 Bit Unsigned Integer | Set to request a block to be deleted. The driver will reset this attribute when the delete block request is completed.  |
| RESULT      | String                 | Message indicating result of the last configuration related request.  |
| ACKNAK      | 8 Bit Unsigned Integer | Result of the last block read or tune activity. A non-zero code indicates an error has occurred.  |
| FC          | 8 Bit Unsigned Integer | Function code number returned for the last block read.  |
| FCNAME      | String                 | Function code name returned for the last block read.  |
| SX_COUNT    | 8 Bit Unsigned Integer | Number of valid specifications for last block read.   |
| SPEC_FORMAT | Named Set              | Determines how specification data is to be formatted (see note 1).  |
| S01 to S63  | String                 | Data for specifications 1 through 63  |

Notes:

- 1.) The format of how the specification data is returned can be selected utilizing the SPEC\_FORMAT attribute. The following choices are available:

|               |   |
|---------------|---|
| Sx Name Value | Format includes spec number, its name and current value |
| Name Value    | Format includes spec name and its current value         |
| Sx Name       | Format includes spec number, and its name               |
| Name          | Format includes just the name of the spec               |

- 2.) The PASSWORD attribute provides protection against performing operations that can be detrimental to plant operations by users of this block who are not familiar with Bailey configuration principals. The BLK block is enabled to perform these operations when the correct password is entered from Control Studio online or from the BLK faceplate. The password defaults to “password” when the block is initially configured. A new password can only be configured using DeltaV Explorer or Control Studio offline. The new password takes effect after the block is downloaded.

### 7.3.1 Reading A Bailey Block

To retrieve existing block specification information, the block address must first be written to the RING, NODE, MODULE and BLOCK attributes of this block. Once the address is written, the READ tag can be set TRUE which retrieves the block specifications. The specification values are copied to the corresponding BLKVAL block S01\_VALUE through S63\_VALUE attributes. The S01 through S63 attributes are also updated with a description of each returned specification. The FC attribute will return the block function code number and the FCNAME attribute will return the name of the function code. Completion of the read request is signaled when the READ attribute is reset to FALSE. The RESULT tag will also return a text message indicating completion of the read operation or the reason for failure if the read cannot be completed. When the ACKNAK tag is zero the request was successful, non-zero indicates an error occurred.

Since Bailey function blocks can have up to 63 specifications, attributes exist for the maximum number of specifications. The actual number of valid specifications for any given block read can be determined by examining SX\_COUNT attribute or the descriptions assigned to each specification (S01 through S63 attributes). Null descriptors indicate the specification number is not valid for the block read.

### 7.3.2 Tuning A Bailey Block

Some Bailey specifications can be modified while the block is executing. These are called tunable specifications and are indicated as such by the specification description having the 'T' character in front of it. The corresponding BLKVAL block S01\_VALUE through S63\_VALUE attributes can be used to change the value of tunable specifications. When the TUNE attribute is set TRUE, any tunable S01\_VALUE through S63\_VALUE attribute that was changed will be included in the Bailey block tune operation. Completion of the tune operation is flagged when the TUNE attribute is reset to FALSE. The RESULT attribute will also return a text message indicating completion of the tune operation or the reason for failure if the tune cannot be completed. When the ACKNAK attribute is zero the request was successful, non-zero indicates an error occurred.

### 7.3.3 Reading The Entire Bailey Configuration

The entire Bailey configuration can be read using the READ and NEXT attributes (also see save command processing). First start by using the RING, NODE, MODULE, BLOCK and READ attributes to read block zero which is the start of any given module configuration. Each time the NEXT attribute is set TRUE, the next block from the last one just read will be retrieved. The returned specification values are copied to the corresponding BLKVAL block S01\_VALUE through S63\_VALUE attributes. The S01 through S63 attributes

are also updated with a description of each returned specification. The FC attribute will return the block function code number and the FCNAME attribute will return the name of the function code. Completion of the next read operation is flagged when the NEXT attribute is reset to FALSE. The RESULT attribute will also return a text message indicating completion of the next read operation or the reason for failure if the next read cannot be completed. When the ACKNAK attribute is zero the request was successful, non-zero indicates an error occurred.

### 7.3.4 Changing Bailey Controller Module Modes

Bailey controller modules are always in one of three operating modes. These modes are execute, configure and error. When in the execute mode the controller runs its block configuration. In this mode of operation, existing blocks can be tuned (see section 7.2) but new blocks cannot be added and existing blocks cannot be modified or deleted. When in the configure mode, the controller is not running its block configuration. In this mode of operation, existing blocks can be modified or deleted and new ones can be added. A controller module enters the error mode of operation when it detects a configuration error on transition to the execute mode. A controller can also enter the error mode when a trip block (Bailey Function Code 32) receives a trip signal.

The OPERATION attribute can be used to change the current operating mode of any Bailey controller. The MODE\_T attribute can be used to monitor the module's current mode of operation. It will return a text message indicating the current operating mode. **WARNING**, changing the operating mode of a controller from execute to configure can cause a plant trip or major outage to occur. For this reason, writing to the OPERATION attribute has been placed under password control. When the BLK is configured, a case sensitive password is assigned to it. Before writes to the OPERATION attribute are accepted, the correct password must first be written to the PASSWORD attribute. The PASSWORD attribute returns the text "?????????" when the password has never been received or the received password is incorrect. The PASSWORD attribute returns the text "-----" when the correct password has been received. After password validation, writes to OPERATION will cause the currently addressed module (set using the RING, NODE and MODULE attributes) to transfer to the requested operating mode. The following OPERATION attribute writes are supported:

- 1 = Request module to perform a software reset
- 2 = Request module to enter the configure mode of operation
- 3 = Request module to enter the execute mode of operation
- 4 = Request module to initialize (clear) its current block configuration

Completion of the requested module operation is flagged when the OPERATION attribute is reset to a value of zero (0). The RESULT attribute will also return a text message indicating completion of the module operation or the reason for failure if the requested operating mode cannot be completed. When the ACKNAK attribute is zero the request was successful, non-zero indicates an error occurred. If for example, a request to enter the execute mode actually results in error mode, the RESULT attribute will display a message indicating why this occurred and the Bailey block that caused the error.

For security, it is highly recommended that once a user has completed use of the OPERATION attribute, it should lock it by sending an invalid password to the PASSWORD attribute. Note that it will automatically become invalid after 60 minutes transpires with no user write activity to the BLK block.

### 7.3.5 Writing Bailey Controller Blocks

To write Bailey function blocks to a controller it must be in the configure mode of operation. See the preceding section to understand how to change Bailey controller modes of operation. There are five easy steps a user must follow to write a Bailey function block. First they write the function code number of the intended new block to the FC attribute. Second they write the new block number to the BLOCK attribute. Third they request the default specification settings for the function code by setting the DEFAULT attribute to TRUE. The number of specifications is returned in the SPEC\_COUNT attribute, the default specification values in the corresponding BLKVAL block S01\_VALUE through S63\_VALUE attributes and the specification descriptors in the S01 through S63 attributes. Completion of the read default request is signaled when the DEFAULT attribute is reset to FALSE. The RESULT attribute will also return a text message indicating completion of the read default operation or the reason for failure if it cannot be completed. When the ACKNAK attribute is zero the request was successful, non-zero indicates an error occurred. The fourth step is to change the appropriate specifications from their default values to values that are pertinent to the new block about to be written. This is accomplished by writing to the corresponding BLKVAL block S01\_VALUE through S63\_VALUE attributes. Note that these attributes are always written as floating point values. DVC-Bailey will automatically take care of translating the values to the appropriate types required by the Bailey function block. The fifth and last step is to request the block to be written to the Bailey controller. This is accomplished when the WRITE attribute is set TRUE. The values of the corresponding BLKVAL block S01\_VALUE through S63\_VALUE attributes appropriate for the block function code are translated to the correct data types required for the Bailey function code and the block write occurs. Completion of the write operation is flagged when the WRITE attribute is reset to FALSE. The RESULT attribute will also return a text message indicating completion of the write operation or the reason for failure if the write cannot be completed. When the ACKNAK attribute is zero the request was successful, non-zero indicates an error occurred.

### 7.3.6 Modifying Bailey Controller Blocks

To modify a Bailey function block, the controller must be in the configure mode of operation. See section 6.3.4 to understand how to change Bailey controller modes of operation. There are three easy steps a user must follow to modify a Bailey function block. First it must read the function block to be modified. See section 6.3.1 to understand how to read a block. The second step is to change the appropriate specifications from their current values to the values to be modified. This is accomplished by writing to the corresponding BLKVAL block S01\_VALUE through S63\_VALUE attributes. Note that these attributes are always written as floating point values. DVC-Bailey will automatically take care of translating the values to the appropriate types required by the Bailey function block. The third and last step is to request the block modification by setting the WRITE attribute TRUE. The values of the corresponding BLKVAL block S01\_VALUE through S63\_VALUE attributes appropriate for the block function code are translated to the correct data types required for the Bailey function code and the block modification occurs. Completion of the modification operation is flagged when the WRITE attribute is reset to FALSE. The RESULT attribute will also return a text message indicating completion of the modification operation or the reason for failure if the modification cannot be completed. When the ACKNAK attribute is zero the request was successful, non-zero indicates an error occurred.

### 7.3.7 Deleting Bailey Controller Blocks

To delete a Bailey function block, the controller must be in the configure mode of operation. See section 6.3.4 to understand how to change Bailey controller modes of operation. There are two easy steps a user must follow to delete a Bailey function block. First it must read the function block to be deleted. See

section 6.3.1 to understand how to read a block. The second step is to request the block deletion by setting the DELETE attribute to a value of TRUE. Completion of the delete is flagged when the DELETE attribute is reset to FALSE. The RESULT attribute will also return a text message indicating completion of the delete operation or the reason for failure if the deletion cannot be completed. When the ACKNAK attribute is zero the request was successful, non-zero indicates an error occurred.

### 7.3.8 Configuration Command Processing

The CMD attribute allows the user to perform a variety of Bailey configuration related activities using simple text commands. The user sends the text command to the CMD attribute. It can monitor completion of the requested command using the CMD\_ACT attribute. The RESULT attribute will contain a text message showing the result of the command when it has finished. The following table presents an overview of the supported commands:

| <b>Command</b>   | <b>Arguments</b>  | <b>Description</b>   |
|------------------|---|--|
| <i>OPERATION</i> | RESET or 1 or CONFIGURE or 2 or EXECUTE or 3 or INITIALIZE or 4<br><RING> <NODE> <MODULE> | Changes a Bailey controller mode of operation.                         |
| SAVE             | <FILE_NAME> <RING> <NODE><br><MODULE>   | Saves a Bailey controller configuration to a file.                     |
| LOAD             | <FILE_NAME> <RING> <NODE><br><MODULE>   | Restores a Bailey controller configuration from a file.                |
| CANCEL           | none  | Cancels the currently active command.                                  |
| READ             | <BLOCK> <RING> <NODE><br><MODULE>   | Reads a function block configured in a Bailey controller.              |
| NEXT             | none  | Read next block.   |
| GET              | <FUNCTION_CODE> <BLOCK><br><RING> <NODE> <MODULE>   | Gets the default specifications for a Bailey controller function code. |
| <i>WRITE</i>     | <START_BLOCK><br><NUMBER_BLOCKS><br><BLOCK_INCREMENT> <RING><br><NODE> <MODULE>           | Writes a function block to a Bailey controller.                        |
| <i>MODIFY</i>    | <START_BLOCK><br><NUMBER_BLOCKS><br><BLOCK_INCREMENT> <RING><br><NODE> <MODULE>           | Modifies an existing function block within a Bailey controller.        |
| <i>DELETE</i>    | <START_BLOCK> <END_BLOCK><br><RING> <NODE> <MODULE>                                       | Deletes one or more blocks within a Bailey controller.                 |

The command syntax is simple. The commands and arguments are case insensitive. Each argument must be separated by one or more spaces (not commas). Note that the < > characters are not part of the above arguments. They indicate the argument is optional. If an optional argument is omitted the arguments following it cannot be included in the command.

Note that the italicized commands are under password control. **WARNING**, execution of these commands can cause a plant trip or major outage to occur. They require the correct password to be sent to the PASSWORD attribute before the command can be executed. When the DVC-Bailey BLK is configured, a case sensitive password is assigned to it. Before accepting protected commands being written to the CMD attribute, the user must first write the correct password to the PASSWORD attribute. The PASSWORD attribute returns the text "?????????" when the password has never been received or the received password

is incorrect. The PASSWORD attribute returns the text "-----" when the correct password has been received. For security, it is highly recommended that once a user has completed their use of the CMD attribute, they should lock it by sending an invalid password to the PASSWORD attribute. Note that it will automatically be invalidated after 60 minutes transpires with user write activity to the BLK block.

### Operation Command

Use this command to change the current operating mode of a controller. See section 7.2 entitled "Changing Bailey Controller Module Modes" for a detailed discussion on these modes. The command argument can be the name of the mode or its corresponding mode number. If the address arguments are omitted the current address specified in the RING, NODE, MODULE attributes will be used as the destination address for the operation. Following are some example operation commands:

| Command                    | Action   |
|----------------------------|--|
| OPERATION CONFIGURE        | Changes currently addressed module to configure mode.    |
| OPERATION CONFIGURE 1 3 6  | Changes ring 1 node 3 module 6 to configure mode.        |
| OPERATION 2                | Changes currently addressed module to configure mode.    |
| OPERATION 2 1 3 7          | Changes ring 1 node 3 module 7 to configure mode.        |
| OPERATION EXECUTE 1 3 7    | Changes ring 1 node 3 module 7 to execute mode.          |
| OPERATION INITIALIZE 1 3 6 | Initializes the configuration of ring 1 node 3 module 6. |

### Save Command

Use this command to save a Bailey module function block configuration to a file. Note that only the function blocks are saved. BASIC programs or C programs with any associated data files are not included in the save. Contact Emerson Process Management if your needs include saving programs. Any file name can be specified. If the filename contains spaces it must be enclosed in double quotes. If the file name is excluded, it will automatically be named based on the current address specified in the RING, NODE and MODULE attributes. For example assume these attributes are currently set to the corresponding values of 1, 2 and 3. The file name will be "00100203.C90" or "10203.CFG". This convention makes it easy to identify what Bailey module the file corresponds with. Configurations can be saved in one of two file formats. The FILE\_TYPE attribute selects the file format to utilize. When FILE\_TYPE is reset (0), Bailey function block configuration data will be saved in files that have a "C90" extension. This is the DVC-Bailey file format. When the file name is included in the command, it will automatically be appended with the "C90" extension if it is missing. Including the "C90" extension in the file name will override the current FILE\_TYPE attribute setting. The "C90" files are stored on the Application Station in the "DVDData\ABB" subdirectory unless an alternative directory path is included as part of the file name. The "DVDData\ABB" directory is automatically created when the first save operation is requested. When FILE\_TYPE is set (1), Bailey function block configuration data will be saved in files that have a "CFG" extension. These files must already exist and are of the format generated by the ABB Bailey CADEWS software and its derivatives. They must be copied to the Application Station "DVDData\ABB" directory. Create this directory if it does not already exist. When the file name is included in the command, it will automatically be appended the "CFG" extension if it is missing. Including the "CFG" extension in the file name will override the current FILE\_TYPE attribute setting. The "CFG" files are also stored on the Application Station in the "DVDData\ABB" subdirectory unless an alternative directory path is included as part of the file name. If the save command is cancelled or it fails, the file will not be generated and any original file of that name that had been previously saved will remain unchanged. Following are some example save commands:

| Command | Action |
|---------|--------|
|---------|--------|



|                     |  |
|---------------------|--|
| SAVE                | Saves currently addressed module.                              |
| SAVE MFP01          | Saves currently addressed module to a file called "MFP01.C90". |
| SAVE MFP01.CFG      | Saves currently addressed module to a file called "MFP01.CFG". |
| SAVE DIGESTER 1 3 5 | Saves ring 1 node 3 module 5 to a file called "DIGESTER.C90".  |

### Load Command

Use this command to load a Bailey module function block configuration from a previously saved file. Remember the module must first be in configure mode before it can be loaded with a new configuration. It should also be initialized before loading the new configuration (see OPERATION command). BASIC programs or C programs with any associated data files are not included in the load. Contact Emerson Process Management if your needs include loading of programs. Any file name can be specified. If the file name includes spaces it must be enclosed in double quotes. If the file name is excluded, a file name based on the current address specified in the RING, NODE and MODULE attributes will be loaded. For example assume these attributes are currently set to the corresponding values of 1, 4 and 5. The file name that will be loaded is "00100405.C90" or "10405.CFG". DVC-Bailey can load the configuration from one of two file formats. The FILE\_TYPE attribute selects the file format to utilize. When FILE\_TYPE is reset (0), the Bailey function block configuration data will be loaded from files that have a "C90" extension. This is the DVC-Bailey file format. When the file name is included in the command, it will automatically be appended with the "C90" extension if it is missing. Including the "C90" extension in the file name will override the current FILE\_TYPE attribute setting. The "C90" files are loaded from the Application Station "DVDData\ABB" subdirectory unless an alternative directory path is included as part of the file name. When FILE\_TYPE is set (1), the Bailey function block configuration data will be loaded from files that have the "CFG" extension. These files are of the format generated by the ABB Bailey CADEWS software and its derivatives. They must be initially copied to the Application Station "DVDData\ABB" directory. Create this directory if it does not already exist. When the file name is included in the command, it will automatically be appended with the "CFG" extension if it is missing. Including the "CFG" extension in the file name will override the current FILE\_TYPE attribute setting. The "CFG" files are loaded from the Application Station "DVDData\ABB" subdirectory unless an alternative directory path is included as part of the file name. Following are some example load commands:

| Command           | Action  |
|-------------------|---|
| LOAD              | Load currently addressed module.                              |
| LOAD SLC          | Load currently addressed module from a file called "SLC.C90". |
| LOAD SLC.CFG      | Load currently addressed module from a file called "SLC.CFG". |
| LOAD ROLLER 1 4 6 | Load ring 1 node 4 module 6 from a file called "ROLLER.C90".  |

### Cancel Command

Use this command to cancel a previously started command. Some commands like SAVE or LOAD might take minutes to complete. The cancel command can be sent to cancel the previously started command.

| Command | Action   |
|---------|--|
| CANCEL  | Cancels the previously entered and active command. |

### Read Command

Use this command to read an existing function block configured in a Bailey module. Following are some example read commands:

| Command | Action |
|---------|--------|
|---------|--------|

|                 |   |
|-----------------|---|
| READ            | Read currently addressed function block.                |
| READ 30         | Read function block 30 from currently addressed module. |
| READ 40 1 10 20 | Read function block 40 from ring 1 node 10 module 20.   |

#### Next Command

Use this command to read the next block in the Bailey module from the last block just read. Following is the only use of this command

| Command | Action   |
|---------|--|
| NEXT    | Read next block from currently addressed function block. |

#### Get Command

Use this command to read default function block data for the any function code supported by a given Bailey module. Note that for some older Bailey modules, the get request must include a block number value that is consistent with the function code number. For example with a Bailey COM module specifying function code 21 for block 0 will return an error because configuring function code 21 at block 0 is illegal. More current modules will return the default data even if the function code could not be configured at the indicated block number. Following are some example get commands:

| Command        | Action   |
|----------------|--|
| GET            | Get default specifications for currently addressed function code, block and address. |
| GET 80         | Get default specifications for function code 80 for current address.                 |
| GET 30 1 32 30 | Get default specifications for function code 30 for ring 1 node 32 module 30.        |

#### Write Command

Use this command to write a new function block to a Bailey module. Remember the module must first be in configure mode before a new block can be written. The steps involved when writing a new function block is to first define it and then write it. First define it by using the READ command to read an existing function block of the same type or using the GET command to read the default settings for the desired function block type. Next change the specification settings by writing to the corresponding BLKVAL block S01\_VALUE through S63\_VALUE attributes. After the specifications have been changed to the desired values enter this command. Following are some example write commands:

| Command               | Action   |
|-----------------------|--|
| WRITE                 | Write one function block to the currently addressed module.  |
| WRITE 200             | Write one function block at block 200 to the currently addressed module.   |
| WRITE 100 10          | Starting at block 100 write ten function blocks to the currently addressed module.   |
| WRITE 500 3 6         | Starting at block 500 write three function blocks incrementing the written block number by 6 for each write to the currently addressed module. This end result is writing blocks 500, 506 and 512. |
| WRITE 1000 20 2 1 4 6 | Starting at block 1000 write twenty function blocks incrementing the written block number by 2 for each write to ring 1 node 4 module 6.   |

#### Modify Command

Use this command to modify an existing function block in a Bailey module. Remember the module must first be in configure mode before a block can be modified. First use the READ command to read the block to be modified. Next change the specification settings by writing to the corresponding BLKVAL block S01\_VALUE through S63\_VALUE attributes. After the specifications have been changed to the desired new values enter this command. Following are some example modify commands:

| <b>Command</b>         | <b>Action</b>  |
|------------------------|--|
| MODIFY                 | Modify one function block in the currently addressed module.   |
| MODIFY 300             | Modify one function block at block 300 in the currently addressed module.  |
| MODIFY 50 5            | Starting at block 50 modify five function blocks in the currently addressed module.  |
| MODIFY 750 4 3         | Starting at block 750 modify four function blocks incrementing the modifying block number by 3 for each modification in the currently addressed module. This end result is modifying blocks 750, 753, 756 and 759. |
| MODIFY 800 10 3 1 10 7 | Starting at block 800 modify ten function blocks incrementing the modifying block number by 3 for each modification in ring 1 node 10 module 7.  |

### Delete Command

Use this command to delete one or more existing function blocks configured in a Bailey module. Remember the module must first be in configure mode before blocks can be deleted. Following are some example delete commands:

| <b>Command</b>         | <b>Action</b>   |
|------------------------|---|
| DELETE                 | Delete currently addressed function block.                            |
| DELETE 30              | Delete function block 30 from currently addressed module.             |
| DELETE 30 40           | Delete function blocks 30 through 40 from currently addressed module. |
| DELETE 100 120 1 10 20 | Delete function blocks 100 through 120 from ring 1 node 10 module 20. |

## 7.4 Block Data Values (BLKVAL)

The BLKVAL DeltaV block is used in conjunction with an associated BLK DeltaV block. The association is made by linking the BLY\_IN attribute of the BLKVAL block to the BLY\_OUT output attribute of the BLK block in DeltaV Control Studio. When a block read request is issued to the BLK block, the specification values returned are copied to the BLKVAL block S01\_VALUE through S63\_VALUE attributes. When the Bailey controller module is in execute mode, the tunable specification values (see BLK block specification description) can be changed. When the BLK block TUNE attribute is set, any tunable S01\_VALUE through S63\_VALUE attribute that was changed will be included in the Bailey block tune operation. When the Bailey controller module is in configure mode, all specification values (see BLK block specification description) can be changed. When the BLK block WRITE attribute is set, the S01\_VALUE through S63\_VALUE attributes applicable to the type of function block are included in the Bailey block write operation.

Restrictions: None, this DeltaV block can be utilized with all Bailey interface types.

| ATTRIBUTE              | TYPE                        | DESCRIPTION   |
|------------------------|-----------------------------|---|
| IDB_ID                 | 32 Bit Unsigned Integer     | Associated Interface Definition Block ID (See IDB block). |
| MESSAGE                | String                      | Message describing state of the block operation.          |
| BLY_IN                 | 32 Bit Unsigned With Status | Input used to link this block to an associated BLK block. |
| S01_VALUE to S63_VALUE | Floating Point              | Values for specifications 1 through 63                    |

## 7.5 Data Acquisition Digital (DADIG)

The DADIG DeltaV block is used to retrieve the exception reported output from a Bailey Data Acquisition Digital block (function code 211). Italicized attribute names indicate the values received from Bailey. This block is used to receive discrete values from Bailey Infi 90 system to the DeltaV system.

Restrictions: This block cannot be used with NETWORK 90 interfaces. It is valid for all INFI 90 interface types.

| ATTRIBUTE        | TYPE                    | DESCRIPTION   |
|------------------|-------------------------|---|
| IDB_ID           | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).   |
| MESSAGE          | String                  | Message describing state of the block operation.  |
| RING             | 8 Bit Unsigned Integer  | Bailey ring address.  |
| NODE             | 8 Bit Unsigned Integer  | Bailey node address.  |
| MODULE           | 8 Bit Unsigned Integer  | Bailey module address.  |
| BLOCK            | 16 Bit Unsigned Integer | Bailey block address.   |
| DISCONNECT       | 8 Bit Unsigned Integer  | When true requests that the data for this block be established in the Bailey interface but not connected. This stops data flow unless it goes in or out of an alarm condition. The functionality of this attribute is a possible future feature that is not currently implemented.  |
| OUT_D_LSD0       | String                  | Logic state descriptor for output of zero.  |
| OUT_D_LSD1       | String                  | Logic state descriptor for output of one.   |
| COMMAND          | Named Set               | Bailey DADIG block control as follows (note 1):<br>* Reset custom input,<br>* Set custom input,<br>* Select custom input,<br>* Select primary input,<br>* Select alternate input,<br>* Enable alarm suppression,<br>* Disable alarm suppression,<br>* Clear alarm latch,<br>* Disable exception reports from this block,<br>* Enable exception reports from this block,<br>* Force an exception report from this block. |
| <i>DISC_LIM</i>  | 8 Bit Unsigned Integer  | Discrete alarm state.   |
| <i>DISC_ACT</i>  | Alarm                   | Alarm active indicator.   |
| <i>QUALITY</i>   | 8 Bit Unsigned Integer  | Quality of the data (0 = good, 1 = bad)   |
| <i>OUT_D</i>     | Discrete With Status    | The Bailey discrete output value and status.  |
| <i>OUT_T</i>     | String                  | Set to OUT_D_LSD0 or OUT_D_LSD1 based on current value of OUT_D.  |
| <i>FACE_TYPE</i> | 8 Bit Unsigned Integer  | Bailey faceplate type code which is value of FC 211 S17.  |

|                   |                         |  |
|-------------------|-------------------------|--|
| <i>ALM</i>        | 8 Bit Unsigned Integer  | Alarm active indicator.  |
| <i>ALM_TOG</i>    | 8 Bit Unsigned Integer  | Return alarm toggle indicator.   |
| <i>OVR_STATUS</i> | 8 Bit Unsigned Integer  | Status override indicator.   |
| <i>SUSPECT</i>    | 8 Bit Unsigned Integer  | Alarm suspect indicator.   |
| <i>E_STOP</i>     | 8 Bit Unsigned Integer  | Emergency stop indicator.  |
| <i>IN_SELECT</i>  | 8 Bit Unsigned Integer  | Input is selectable indicator (note 2).  |
| <i>IN_ALT</i>     | 8 Bit Unsigned Integer  | Alternate input active indicator / request (note 2).   |
| <i>IN_CUSTOM</i>  | 8 Bit Unsigned Integer  | Custom input active indicator / request (note 2).  |
| <i>IN_PRIMARY</i> | 8 Bit Unsigned Integer  | Primary input active indicator / request (note 2).   |
| <i>LATCH</i>      | 8 Bit Unsigned Integer  | Alarm latch indicator / reset request (note 3).  |
| <i>SUPPRESS</i>   | 8 Bit Unsigned Integer  | Alarm suppress indicator / set / reset (note 4).   |
| <i>XRP_OFF</i>    | 8 Bit Unsigned Integer  | Exception report off indicator / set / reset / request (note 5).   |
| <i>SIM_TYPE</i>   | Named Set               | Defines type of simulated value. Simulation is active when the IDB SCHEME attribute is set to the "Simulated Interface" selection. |
| <i>SIM_TIME</i>   | 32 Bit Unsigned Integer | Defines simulation time in milliseconds. Consult "Simulation of Bailey System" for additional details.                             |

Notes:

- 1.) The COMMAND attribute can be used to send various commands to the Bailey DADIG block. Specifically, it allows selection of the CUSTOM input state, what input should currently be used, control of alarm suppression, resetting the alarm latch and control of exception reporting. Note that writing to the attributes that report the current state of that block feature can also control these features (notes 2 – 5).
- 2.) When the IN\_SELECT attribute is TRUE (1), the DADIG block can be commanded to use its various inputs. The IN\_ALT, IN\_CUSTOM and IN\_PRIMARY attributes indicate the current input being used by the block. They can also be used to command the block to use a different input. For example assume the current input is primary, then IN\_PRIMARY will be one, IN\_ALT and IN\_CUSTOM will be zero. When a one is written to IN\_ALT, the DADIG block will begin using the alternate input. Note that writes will be rejected if the IN\_SELECT attribute is not TRUE (1).
- 3.) The DADIG block alarm latch is indicated by this attribute. By writing FALSE (0) to this attribute, the latch will be cleared. Attempting to set the latch by writing a TRUE (1) is not supported. The latch is set within the Bailey controller logic and once set can be commanded reset.
- 4.) This attribute indicates alarm suppression is in effect. It can also be used to set or reset alarm suppression (only if enabled by the DADIG block configuration) by writing a respective TRUE (1) or FALSE (0) to this attribute. DADIG block alarm suppression is not linked to DeltaV alarm suppression. DeltaV alarm suppression supersedes DADIG block alarm suppression.
- 5.) This attribute indicates whether or not exception reporting for this block has been disabled. If XRP\_OFF is TRUE (1), exception reporting is disabled. This attribute can be used to turn exception reporting off or on. Write TRUE (1) to it to turn exception reporting off. Write FALSE (0) to it to enable exception reporting. When exception reporting is off and TRUE (1) is written to

this attribute, an exception report force command will be sent. This message causes the block to exception report once but still leave exception reporting off.

## 7.6 Data Acquisition Analog (DANG)

The DANG DeltaV block is used to retrieve the exception reported output from a Bailey Data Acquisition Analog block (function code 177). Italicized attribute names indicate the values received from Bailey.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02).

| ATTRIBUTE         | TYPE                    | DESCRIPTION  |
|-------------------|-------------------------|--|
| IDB_ID            | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE           | String                  | Message describing state of the block operation.   |
| RING              | 8 Bit Unsigned Integer  | Bailey ring address.   |
| NODE              | 8 Bit Unsigned Integer  | Bailey node address.   |
| MODULE            | 8 Bit Unsigned Integer  | Bailey module address.   |
| BLOCK             | 16 Bit Unsigned Integer | Bailey block address.  |
| DISCONNECT        | 8 Bit Unsigned Integer  | When true requests that the data for this block be established in the Bailey interface but not connected. This stops data flow unless it goes in or out of an alarm condition. The functionality of this attribute is a possible future feature that is not currently implemented. |
| COMMAND           | Named Set               | Command attribute to request Bailey DAANG block as follows (only valid when mode is Auto):<br><br>Use Auto Input<br>Use Calculated Input<br>Suppress Alarms<br>Un-suppress Alarms<br>Turn Scan Off<br>Turn Scan On   |
| <i>QUALITY</i>    | 8 Bit Unsigned Integer  | Quality of the data (0 = good, 1 = bad)  |
| <i>EU_CODE</i>    | 8 Bit Unsigned Integer  | Bailey engineering units code which is value of FC 177 S6.   |
| <i>RED_TAG</i>    | 8 Bit Unsigned Integer  | Bailey red tag indicator (0 = no tag, 1 = tagged)  |
| <i>OUT_SCALE</i>  | Scaling                 | High and low OUT limits (received from Bailey DAANG Block S1 and S2 settings) along with DeltaV engineering units. Also includes declaration of digits to the right of the decimal point associated with OUT for display purposes.   |
| <i>OUT_HI_LIM</i> | Floating Point          | OUT high limit (duplicated in <i>OUT_SCALE</i> ).  |



|                     |                            |   |
|---------------------|----------------------------|---|
| <i>OUT_LO_LIM</i>   | Floating Point             | OUT low limit (duplicated in OUT_SCALE).  |
| <i>QUALITY</i>      | 8 Bit Unsigned Integer     | Quality of the data (0 = good, 1 = bad)   |
| <i>OUT</i>          | Floating Point With Status | Current output value and status received from Bailey.   |
| <i>MODE</i>         | Mode                       | The mode of the Bailey block. MODE contains the actual, target, permitted, and normal modes.  |
| <i>MODE_OFF</i>     | 8 Bit Unsigned Integer     | Mode control flag (0 = on, 1 = off, see note 1).  |
| <i>ALARM_T</i>      | String                     | Indicates current highest priority alarm (see note 3).  |
| <i>HI_HI_HI_ACT</i> | Alarm                      | High high high alarm active indicator.  |
| <i>HI_HI_ACT</i>    | Alarm                      | High high alarm active indicator.   |
| <i>HI_ACT</i>       | Alarm                      | High alarm active indicator.  |
| <i>HI_RATE_ACT</i>  | Alarm                      | High alarm rate active indicator.   |
| <i>HI_RATE_ACT</i>  | Alarm                      | High rate of change alarm active indicator.   |
| <i>LO_RATE_ACT</i>  | Alarm                      | Low rate of change alarm active indicator.  |
| <i>LO_RATE_ACT</i>  | Alarm                      | Low alarm rate active indicator.  |
| <i>LO_ACT</i>       | Alarm                      | Low alarm active indicator.   |
| <i>LO_LO_ACT</i>    | Alarm                      | Low low alarm active indicator.   |
| <i>LO_LO_LO_ACT</i> | Alarm                      | Low low low alarm active indicator.   |
| <i>DV_HI_LIM</i>    | Floating Point             | Deviation high alarm limit.   |
| <i>DV_HI_ACT</i>    | Alarm                      | Deviation high alarm active indicator.  |
| <i>DV_LO_ACT</i>    | Alarm                      | Deviation low alarm active indicator.   |
| <i>DV_LO_LIM</i>    | Floating Point             | Deviation low alarm limit.  |
| <i>RATE_HI_LIM</i>  | Floating Point             | Rate alarm high limit.  |
| <i>RATE_LO_LIM</i>  | Floating Point             | Rate alarm low limit.   |
| <i>HI_HYS</i>       | Floating Point             | The amount the alarm value must lower below the current high alarm level limit before the associated active current alarm condition clears. |
| <i>HI_HI_HI_LIM</i> | Floating Point             | The setting for the alarm limit used to detect the high high high alarm condition (see note 2).   |
| <i>HI_HI_LIM</i>    | Floating Point             | The setting for the alarm limit used to detect the high high alarm condition (see note 2).  |
| <i>HI_LIM</i>       | Floating Point             | The setting for the alarm limit used to detect the high alarm condition.  |
| <i>LO_LIM</i>       | Floating Point             | The setting for the alarm limit used to detect the low alarm condition.   |
| <i>LO_LO_LIM</i>    | Floating Point             | The setting for the alarm limit used to detect the low low alarm condition (see note 2).  |
| <i>LO_LO_LO_LIM</i> | Floating Point             | The setting for the alarm limit used to detect  |

|                  |                         |   |
|------------------|-------------------------|---|
| <i>LO_HYS</i>    | Floating Point          | the low low low alarm condition (see note 2).<br>The amount the alarm value must raise above the current low alarm level limit before the associated active current alarm condition clears. |
| <i>NEXT_HI</i>   | Floating Point          | Next high alarm limit to be reached.  |
| <i>NEXT_LO</i>   | Floating Point          | Next low alarm limit to be reached.   |
| <i>LIMITED</i>   | 8 Bit Unsigned Integer  | Value is limited indicator.   |
| <i>CALC</i>      | 8 Bit Unsigned Integer  | Using calculated input indicator.   |
| <i>OUT_RANGE</i> | 8 Bit Unsigned Integer  | Out of range detected indicator.  |
| <i>SCAN_OFF</i>  | 8 Bit Unsigned Integer  | Scan disabled indicator.  |
| <i>RATE_ACT</i>  | 8 Bit Unsigned Integer  | Rate alarm indicator.   |
| <i>SUP_ACT</i>   | 8 Bit Unsigned Integer  | Suppress alarm indicator (see note 4).  |
| <i>VAR_ACT</i>   | 8 Bit Unsigned Integer  | Variable alarm indicator.   |
| <i>HI_REF</i>    | Floating Point          | High display reference value (DAANG S1).  |
| <i>MID_REF</i>   | Floating Point          | Middle display reference value (DAANG S2).  |
| <i>LO_REF</i>    | Floating Point          | Low display reference value (DAANG S3).   |
| <i>HI_CONST</i>  | Floating Point          | High constraint limit (DAANG S4).   |
| <i>LO_CONST</i>  | Floating Point          | Low constraint limit (DAANG S5).  |
| <i>SIM_TYPE</i>  | Named Set               | Defines type of simulated value. Simulation is active when the IDB SCHEME attribute is set to the "Simulated Interface" selection.  |
| <i>SIM_TIME</i>  | 32 Bit Unsigned Integer | Defines simulation time in milliseconds. Consult "Simulation of Bailey System" for additional details.  |

Notes:

- 1.) The *MODE\_OFF* attribute is a flag the DAANG faceplate utilizes to determine whether or not the faceplate should display the mode control objects. These objects allow the DAANG block to be command between automatic (block provides values) and manual (user provides values). Other mode control objects are ramping the user provided value, selecting between automatic and calculated input, suppressing and un-suppressing alarms and turning the Bailey DAANG block scan off and on. The DAANG faceplate hides the mode control objects when the *MODE\_OFF* attribute is set and displays them when it is reset.
- 2.) The alarm limit is configured as an absolute value in the DVC-Bailey DANG block. Within the Bailey DAANG block it is configured as a difference from the next alarm limit. The DVC-Bailey interface is designed to automatically translate between difference and absolute when exchanging the alarm limit value between the two systems.
- 3.) The *ALARM\_T* attribute is a text string indicating the current single highest priority alarm condition. Displays the following two character strings listed in order of priority (highest to lowest when more then one alarm condition is true):

|      |   |
|------|---|
| “**” | block has bad quality                     |
| “i”  | bailey block alarm suppression is enabled |
| “3H” | high high high alarm active               |

|      |                             |
|------|-----------------------------|
| “3L” | low low low alarm active    |
| “2H” | high high alarm active      |
| “2L” | low low alarm active        |
| “ H” | high alarm active           |
| “ L” | low alarm active            |
| “HD” | high deviation alarm active |
| “LD” | low deviation alarm active  |
| “HR” | high rate alarm active      |
| “LR” | low rate alarm active       |
| “ ”  | no alarm condition          |

- 4.) The SUP\_ACT attribute indicates whether or not the Bailey DAANG block is suppressing its alarm generation information. This indication does not affect the DeltaV alarm suppression which must be enabled / disabled from the associated detailed faceplate.

## 7.7 Device Driver (DD)

The DD DeltaV block is used to retrieve and control the exception reported output from a Bailey Device Driver block (function code 123). Italicized attribute names indicate the values received from Bailey.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02).

| ATTRIBUTE        | TYPE                    | DESCRIPTION  |
|------------------|-------------------------|--|
| IDB_ID           | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE          | String                  | Message describing state of the block operation.   |
| RING             | 8 Bit Unsigned Integer  | Bailey ring address.   |
| NODE             | 8 Bit Unsigned Integer  | Bailey node address.   |
| MODULE           | 8 Bit Unsigned Integer  | Bailey module address.   |
| BLOCK            | 16 Bit Unsigned Integer | Bailey block address.  |
| DISCONNECT       | 8 Bit Unsigned Integer  | When true requests that the data for this block be established in the Bailey interface but not connected. This stops data flow unless it goes in or out of an alarm condition. The functionality of this attribute is a possible future feature that is not currently implemented. |
| OUT_D_LSD0       | String                  | Logic state descriptor for output of zero.   |
| OUT_D_LSD1       | String                  | Logic state descriptor for output of one.  |
| F1_LSD0          | String                  | Logic state descriptor for feedback 1 of zero.   |
| F1_LSD1          | String                  | Logic state descriptor for feedback 1 of one.  |
| F2_LSD0          | String                  | Logic state descriptor for feedback 2 of zero.   |
| F2_LSD1          | String                  | Logic state descriptor for feedback 2 of one.  |
| DISC_LIM         | 8 Bit Unsigned Integer  | Discrete alarm state (see note 1). DISC_LIM should always equal 1. Making DISC_LIM equal to 0 may cause unpredictable alarm behavior.  |
| <i>DISC_ACT</i>  | Alarm                   | Alarm active indicator. DISC_ACT is a function of feedback errors and not merely a function of DISC_LIM.   |
| <i>FACE_TYPE</i> | 8 Bit Unsigned Integer  | Bailey faceplate type code which is value of FC 123 S10.   |
| <i>QUALITY</i>   | 8 Bit Unsigned Integer  | Quality of the data (0 = good, 1 = bad)  |
| <i>OUT_D</i>     | Discrete With Status    | The Bailey discrete output value and status  |
| <i>OUT_T</i>     | String                  | Set to OUT_D_LSD0 or OUT_D_LSD1 based on current value of OUT_D.   |
| <i>RED_TAG</i>   | 8 Bit Unsigned Integer  | Bailey red tag indicator (0 = no tag, 1 = tagged)  |

|                     |                         |  |
|---------------------|-------------------------|--|
| <i>MODE</i>         | Mode                    | The mode of the Bailey block. MODE contains the actual, target, permitted, and normal modes.   |
| <i>F1</i>           | 8 Bit Unsigned Integer  | State of feedback number 1.  |
| <i>F2</i>           | 8 Bit Unsigned Integer  | State of feedback number 2.  |
| <i>F1_T</i>         | String                  | Set to F1_LSD0 or F1_LSD1 based on current value of F1.  |
| <i>F2_T</i>         | String                  | Set to F2_LSD0 or F2_LSD1 based on current value of F2.  |
| <i>BAD_FEEDBACK</i> | 8 Bit Unsigned Integer  | Feedback bad indicator.  |
| <i>OVR_STATUS</i>   | 8 Bit Unsigned Integer  | Override status indicator.   |
| <i>SIM_TIME</i>     | 32 Bit Unsigned Integer | Defines simulation time in milliseconds. Simulation is active when the IDB SCHEME attribute is set to the "Simulated Interface" selection. Consult "Simulation of Bailey System" for additional details. |

Notes:

- 1.) This alarm limit is not associated with the state of the block output. It applies to the internal alarm indicator generated by the Bailey block. The Bailey block alarm indicator is always 0 equals no alarm and 1 equals alarm.

## 7.8 Digital Input Loop (DIL)

The DIL DeltaV block is used to retrieve the exception reported output from a Bailey Digital Output / Loop block (function code 45). It can also be used to retrieve digital inputs defined within Bailey Logic Master Modules Group I/O definitions. Italicized attribute names indicate the values received from Bailey. This block is used to receive discrete values from the Bailey system to the DeltaV system.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02).

| ATTRIBUTE       | TYPE                    | DESCRIPTION  |
|-----------------|-------------------------|--|
| IDB_ID          | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE         | String                  | Message describing state of the block operation.   |
| RING            | 8 Bit Unsigned Integer  | Bailey ring address.   |
| NODE            | 8 Bit Unsigned Integer  | Bailey node address.   |
| MODULE          | 8 Bit Unsigned Integer  | Bailey module address.   |
| BLOCK           | 16 Bit Unsigned Integer | Bailey block address.  |
| DISCONNECT      | 8 Bit Unsigned Integer  | When true requests that the data for this block be established in the Bailey interface but not connected. This stops data flow unless it goes in or out of an alarm condition. The functionality of this attribute is a possible future feature that is not currently implemented. |
| OUT_D_LSD0      | String                  | Logic state descriptor for output of zero.   |
| OUT_D_LSD1      | String                  | Logic state descriptor for output of one.  |
| <i>DISC_LIM</i> | 8 Bit Unsigned Integer  | Discrete alarm state.  |
| <i>DISC_ACT</i> | Alarm                   | Alarm active indicator.  |
| <i>QUALITY</i>  | 8 Bit Unsigned Integer  | Quality of the data (0 = good, 1 = bad)  |
| <i>OUT_D</i>    | Discrete With Status    | The Bailey discrete output value and status.   |
| <i>OUT_T</i>    | String                  | Set to OUT_D_LSD0 or OUT_D_LSD1 based on current value of OUT_D.   |
| SIM_TYPE        | Named Set               | Defines type of simulated value. Simulation is active when the IDB SCHEME attribute is set to the "Simulated Interface" selection.   |
| SIM_TIME        | 32 Bit Unsigned Integer | Defines simulation time in milliseconds. Consult "Simulation of Bailey System" for additional details.   |

## 7.9 Digital Output Loop (DOL)

The DOL DeltaV block is used to send an exception reported output generated by value changes to the block's input. The exception reports can be received by a Bailey console tag, Bailey Digital Input / Loop block (function code 42) and Digital Input / Inifnet blocks (function code 122). The block will automatically generate the appropriate quality, and alarming status based on the value presented at the block input. This block is used to send discrete values from the DeltaV system to the Bailey system.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02) and computer interface command series (CIC).

| ATTRIBUTE   | TYPE                    | DESCRIPTION  |
|-------------|-------------------------|--|
| IDB_ID      | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE     | String                  | Message describing state of the block operation.   |
| BLOCK       | 16 Bit Unsigned Integer | Block number to establish this point at within the Bailey interface (see note 1).                            |
| IN_D        | Discrete With Status    | Discrete input value to be exception reported to Bailey.   |
| MAX_TIME    | 16 Bit Unsigned Integer | If IN_D never changes, it will be reported at the maximum time interval (seconds) defined by this attribute. |
| ALARM_STATE | 8 Bit Unsigned Integer  | The setting for the Bailey alarm limit used to derive the alarm condition (see note 2).                      |
| DISC_LIM    | 8 Bit Unsigned Integer  | DeltaV discrete alarm state.   |
| DISC_ACT    | Alarm                   | Alarm active indicator.  |
| OUT_D       | Discrete With Status    | Last discrete input value reported to Bailey.  |

Notes:

- 1.) Make sure the BLOCK number attribute is unique with respect to the other AOL, DOL, ORCM, ORMSC and OSTN DeltaV blocks associated with the same Bailey Interface Definition block. The BLOCK number attribute must also be defined within the range of 1 to maximum number of allowed outputs set up within the associated DeltaV Interface Definition block (see IDB MAX\_OUTPUTS attribute). The Bailey system will receive data from this block at the ring and node address of the Bailey CIU interface, module address two and block number defined by the BLOCK attribute.
- 2.) This attribute is used to setup output alarming states. A value of 0 indicates flag alarm when output state is zero. A value of 1 indicates flag alarm when output state is one. A value of 2 indicates never flag an alarm.

## 7.10 Interface Definition Block (IDB)

The IDB DeltaV block is used to declare an instance of the Bailey driver and define its interfacing data. This block must be defined for each Bailey interface or redundant pair of interfaces which the DeltaV Connect - Bailey is to communicate with. In addition to defining interface specifics, the IDB DeltaV block also declares a unique interface ID (IDB\_ID attribute) referenced by the various other DeltaV DVCBailey blocks. Additionally the IDB block has a number of other attributes used to monitor driver communication health and statistics. Italicized attribute names indicate values updated by the driver in real time.

| ATTRIBUTE   | TYPE                    | DESCRIPTION   |
|-------------|-------------------------|---|
| IDB_ID      | 32 Bit Unsigned Integer | Unique user ID assigned to this interface to be referenced by the other DVCBailey blocks that the driver is to exchange data.   |
| MESSAGE     | String                  | Message describing state of the block operation.  |
| SCHEME      | Named Set               | Desired communication scheme (see note 1):<br><ol style="list-style-type: none"> <li>1.) Single interface,</li> <li>2.) Single interface redundant channels,</li> <li>3.) Dual interfaces single channel,</li> <li>4.) Simulated interface.</li> </ol>  |
| PORT_A      | String                  | Name of the SCSI (see SCSI communication sub-section) or COM port attached to the primary interface channel. When a COM port is specified its communication characteristics are also included. For example<br>"COM1: baud=19200 parity=N data=8 stop=1"<br>selects COM1, set to a baud rate of 19200, no parity, eight data bits and 1 stop bit. An example of specifying a SCSI port would be "S2001:"   |
| PORT_B      | String                  | Name of the SCSI (see SCSI communication sub-section) or COM port attached to the secondary interface channel. When a COM port is specified its communication characteristics are also included. For example<br>"COM2: baud=19200 parity=N data=8 stop=1"<br>selects COM2, set to a baud rate of 19200, no parity, eight data bits and 1 stop bit. Note that this attribute is ignored when the SCHEME attribute is set to single interface. An example of specifying a SCSI port would be "S2002:" |
| MAX_OUTPUTS | 16 Bit Unsigned Integer | Maximum number of output blocks (AOL, DOL, ORCM, ORMSC and OSTN) the driver should reserve point indices (block numbers) for within the Bailey interface device. It is important to note that these indices are specified as part of the AOL, DOL, ORCM, ORMSC and OSTN block definition and must always fall in the  |



|             |                         |  |
|-------------|-------------------------|--|
| WATCHDOG    | 8 Bit Unsigned Integer  | <p>range of one to the maximum outputs defined by this attribute.</p> <p>Desired watchdog timer to be initiated between the driver and Bailey interface device. The timer is expressed in 2.5 second counts. A value of zero disables the watchdog timer. The maximum value is 255 which is equivalent to 637.5 seconds. When the watchdog timer is enabled, the Bailey interface device will remove itself from the communication loop if the elapsed time in which the driver communicates with it exceeds the watchdog timer value. Driver output block values (AOL, DOL, ORCM, ORMSC and OSTN) being received by Bailey controllers will thereafter automatically be marked as bad quality by the Bailey system.</p> |
| XRP_UPDATE  | 32 Bit Unsigned Integer | <p>Frequency in milliseconds at which exception report availability is read from the Bailey interface device. Settings in the range of 500 to 3000 milliseconds are common.</p>  |
| XRP_OUT     | 32 Bit Unsigned Integer | <p>Frequency in milliseconds at which output exception reports are written to the Bailey interface. Settings in the range of 500 to 3000 milliseconds are common.</p>  |
| SPEC_UPDATE | 32 Bit Unsigned Integer | <p>Determines the averaged update rate in milliseconds at which STN/PID and DANG block specifications will be read. This attribute should be set to a value equal to or larger than the total number of STN and DANG blocks in the database times 5000.</p>  |
| LOCK        | 8 Bit Unsigned Integer  | <p>Supervisory control write lock. When True (1), locks out all operator write requests (both supervisory control writes and tuning) from the other DVCBailey blocks that support write capability. The lock only affects those block attributes that exist in the associated Bailey function code. For example High and Low Alarm limits are locked because they exist in the Bailey block but High High and Low Low Alarm limits are not locked because they only exist in the DVC-Bailey block.</p>   |
| OPTIONS     | Option Bitstring        | <p>Supports enabling of various system options (see note 2). The six options supported are:</p> <ol style="list-style-type: none"> <li>1.) Establish points online.</li> <li>2.) Enable Bailey DCS time synchronization.</li> <li>3.) Enable engineering unit mapping.</li> <li>4.) Disable exception report screening.</li> <li>5.) Enable no communication value override.</li> <li>6.) Disable no communication force bad quality.</li> </ol>   |

|                     |                         |   |
|---------------------|-------------------------|---|
| NODE01 to<br>NODE63 | Named Set               | Defines the node map required for Bailey Plantloop based systems when the time syncing option has been enabled in the OPTIONS attribute. For each possible node number, one of the following choices must be selected; Vacant, 1.4K Tag OIU, PCU, CIU01, Other. The node map must be accurately defined for time syncing to function correctly on Plantloop (Network 90) based systems. These attributes are ignored when time syncing is disabled or an InfiNet (Infi 90) based system has been detected.  |
| DEBUG_LOG           | Option Bitstring        | Enables various logging features that may be useful for diagnosing driver operational performance. When any of the logging features are enabled, daily log files are generated in the DeltaV "DVDATA" directory. Leaving all but the error option enabled for extended periods of time can consume a lot of disk space. The four logging options supported are: <ul style="list-style-type: none"> <li>1.) Errors – post miscellaneous errors</li> <li>2.) Events – post communication events</li> <li>3.) Sends – post messages sent to interface.</li> <li>4.) Receives – post messages received from interface.</li> </ul> |
| LOG_DAYS            | 16 Bit Unsigned Integer | Number of days to keep log files before they will be automatically deleted. Setting this value to zero will disable automatic deletion of the log files.  |
| MY_DEVICE           | String                  | Text message indicating the ABB Bailey interface type (i.e. INIC103, CIC01, etc)  |
| MY_RING             | 8 Bit Unsigned Integer  | Ring address of the Bailey interface.   |
| MY_NODE             | 8 Bit Unsigned Integer  | Node address of the Bailey interface.   |
| SYNC_RING           | 8 Bit Unsigned Integer  | Ring address of current time sync master (see note 3).  |
| SYNC_NODE           | 8 Bit Unsigned Integer  | Node address of current time sync master (see note 3).  |
| POINT_TOTAL         | 32 Bit Unsigned Integer | Total blocks for which this IDB block has been requested to exchange data with the Bailey system.   |
| PRI_STATUS          | 8 Bit Unsigned Integer  | Primary communication channel status. A value of zero indicates good and one means bad.   |
| PRI_STATUS_S        | 8 Bit Unsigned Integer  | Standby App Station primary communication channel status. A value of zero indicates good  |

|                       |                         |  |
|-----------------------|-------------------------|--|
| <i>SEC_STATUS</i>     | 8 Bit Unsigned Integer  | and one means bad.<br>Secondary communication channel status. A value of zero indicates good and one means bad.  |
| <i>SEC_STATUS_S</i>   | 8 Bit Unsigned Integer  | Standby App Station secondary communication channel status. A value of zero indicates good and one means bad.  |
| <i>STANDBY_STATUS</i> | 8 Bit Unsigned Integer  | Status of Standby App Station indicated as:<br>0 = Available<br>1 = Communication with Bailey is bad<br>2 = Standby is not available<br>3 = Redundancy is not configured |
| <i>MSG_TOTAL</i>      | 32 Bit Unsigned Integer | Running count of total messages being exchanged with the Bailey interface.   |
| <i>MSG_RATE</i>       | 32 Bit Unsigned Integer | Messages per second being exchanged with the Bailey interface.   |
| <i>XRP_TOTAL</i>      | 32 Bit Unsigned Integer | Running count of total exception reports exchanged with the Bailey interface.  |
| <i>XRP_RATE</i>       | 32 Bit Unsigned Integer | Exception reports per second being exchanged with the Bailey interface.  |
| <i>POLL_TOTAL</i>     | 32 Bit Unsigned Integer | Running count of total values polled from the Bailey interface.  |
| <i>POLL_RATE</i>      | 32 Bit Unsigned Integer | Polled values per second being read from the Bailey interface.   |
| <i>NAK_TOTAL</i>      | 32 Bit Unsigned Integer | Running count of total negative acknowledgments received from the Bailey Interface.  |

Notes:

- 1.) The IDB block is designed to support communication on one or two RS232 ports to a single Bailey interface or two RS232 or SCSI ports to redundant Bailey interfaces. The first redundant scheme is single interface redundant channel. This scheme provides two RS232 communication paths to a single Bailey interface. All Bailey interfaces except (SPM, CPM, CIC, CIU01, INPCI01 and INIC103) have two RS232 ports available for communication. The second port is switch selectable between a "utility" port and computer communication port. Setting it to a computer communication port allows it to be used for the single interface redundant channel communication scheme. With single interface redundant channel, the driver issues exception report read requests to both channels resulting in increased throughput. Each Bailey interface can be assigned the same node address on the Bailey communication highway or given unique addresses. When the Bailey interfaces are configured for the same node address, the driver commands the primary interface online to the Bailey communication loop and commands the secondary interface offline. The database is downloaded to both interfaces so the secondary can be considered to be in a "warm" standby mode, ready to be commanded online if the primary interface fails. When the Bailey interfaces are configured for different node addresses the driver commands both interfaces online receiving exception report data from each. This increases the effective data throughput. The decision of Bailey node address assignment should be based on whether or not any DVCBailey AOL or DOL blocks are to be utilized. When they are used, the Bailey interfaces should be assigned the same node address. Otherwise, when AOLs or DOLs are not used, unique node addresses should be assigned to take advantage of

the extra throughput. The “simulate interface” scheme selection causes the IDB block driver to simulate connection to a Bailey system (see “Simulation of the Bailey System” for details).

- 2.) Option 1 instructs the interface to restart the Bailey interface putting it on-line prior to establishing its block database. Selecting this option in run time has no effect. The IDB block must be downloaded for it to take effect. This option should be selected for Network 90 systems when nodes are observed to be marked offline when the DVC-Bailey system first starts up. It should not be enabled for Infi 90 system since it increases the total time it takes to establish the database in the Bailey interface.

Option 2 enables the interface to time sync the Bailey system time to the application station time. Selecting this option in run time has no effect. The IDB block must be downloaded for it to take effect.

Option 3 enables automatic mapping of Bailey EU codes to DeltaV EU descriptors within the various DVCBailey blocks having the OUT\_SCALE and or PV\_SCALE attributes. See the section entitled “Translation of Engineering Units from Bailey to DeltaV” for more details.

Option 4 disables exception report screening. Selecting this option in run time has no effect. The IDB block must be downloaded for it to take effect. With exception report screening disabled, the Bailey interface will pass every exception report it receives from the loop to the application station. This includes updates with same values as is common with digital point types.

Option 5 enables the no communication value override feature. If communication with the ABB Bailey interface is lost, the values of the AIL, DANG, DIL, STN and TXT blocks are overridden to predefined values. The predefined value for the DIL block is reset (0). The predefined value for the TXT block is message zero (0). The predefined value for the AIL, DANG and STN blocks are the blocks zero value less 25% of its span. So for example a block with a zero value of -100.0 and span of 200.0 would be overridden to a value of -150.0 when communication with the interface is lost. Setting these blocks to the predefined values is useful for visually distinguishing bad values in the historical trends. When communication is restored, the block value reverts back to the actual value.

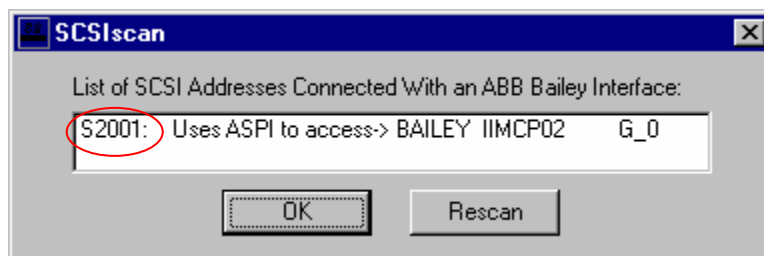
Option 6 disables the no communication force bad quality feature. When this option is not selected and communication with the ABB Bailey interface is lost, the QUALITY attribute of all blocks is forced to one (1) indicating bad quality. DVC-Bailey blocks defined using the standard module templates have a General I/O Failure alarm assigned to the QUALITY attribute. Therefore, when communication is lost all blocks in areas assigned to a stations alarms and events will get logged in the alarm list. When communication is restored, the QUALITY attribute reverts back to its previous state before communication was lost. Setting this option disables the forcing of the QUALITY attribute. Note that the DeltaV status of the block output will still indicate BAD and the MESSAGE attribute will still indicate OFFLINE. Enabling this option stops all blocks from being logged in the alarm list with the General I/O Failure alarm. The loss of communication within the alarm list can be determined using the communication error alarms assigned to the PRI\_STATUS and SEC\_STATUS attributes in the IDB block module template updated in version 7.2b and later.

- 3.) As discussed in the previous note, the IDB block OPTIONS attribute supports selection of the time syncing option. The ability for time syncing to function correction depends on several factors. The most important being that the ABB Bailey interface supports the time syncing feature. Those interfaces that do not support time syncing are NSPM01, IMSPM01, IMCPM02, NCIU01 and INPCI01. Note that time syncing will not work properly on Network 90 systems that have 1400 tag OIUs present on the ABB Bailey Plant Loop. The SYNC\_RING and SYNC\_NODE tags are only updated when the time sync option is enabled.

### 7.10.1 SCSI Communication

The IDB block supports SCSI communication with the INICI03. The PORT\_A and PORT\_B (when dual interface SCHEME is selected) attributes must be set to proper SCSI addresses. Examples of these addresses are S0001:, S2002: etc. The 'S' indicates SCSI followed by the adapter card number, adapter bus number, adapter logical unit number and INICI03 SCSI target ID. Note that the target ID is the value set by switches on the Bailey INICI03 INICI03A module.

A utility program called SCSIscan.exe can be used to determine valid SCSI addresses. It is installed in the DeltaVBin directory. This program scans all SCSI cards installed in the system looking for Bailey SCSI interfaces. Before running SCSIscan, the INICI03 must be powered up, configured to utilize its SCSI communication channel and attached to an Application Station SCSI host adapter card. Note that if the Application Station has been booted before the Bailey SCSI interface was powered up or cabled to the SCSI card, it must be rebooted so the SCSI card driver will detect its presence. Following is an example of running the SCSIscan utility:



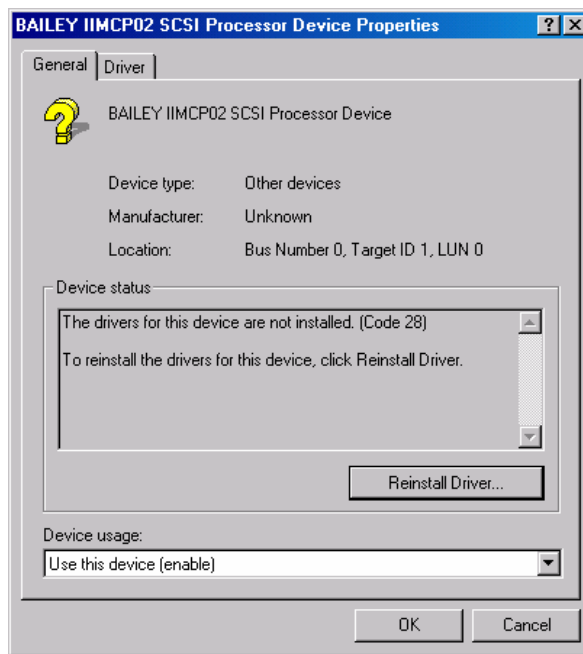
This example shows one SCSI Bailey interface was detected. Its address is S2001: which is the value that should be configured in the IDB PORT\_A attribute.

Most PC SCSI card that supports external connection to the INICI03 IMMPIO1 module's 50 pin SCSI socket should work ok. SCSI cards associated with Raid Controllers should function Ok but are not recommended. When using a SCSI card associated with a Raid Controller if the INICI03 is reset or the SCSI cable removed and re-attached, the App Station must be rebooted for communication to resume. It's best to have a SCSI card exclusively dedicated to the Bailey SCSI interface. RoviSys validated SCSI communication using Adaptec models 29160, 2930 and Domex model 3194U SCSI controller cards. For PCIX based PCs the Adaptec 29160 is supported but not the 29320A. For some SCSI cards, adjustment to the SCSI BIOS might be necessary. The BIOS setup can be configured when the PC is booting. Following are general guidelines for typical settings. Depending on the type of SCSI card, all of these settings or the terminology might be different.

- \* SCSI Parity Checking: Enabled
- \* Host Adapter SCSI Termination: Automatic
- \* Initiate Sync Negotiation: Yes (IDs 0 - 7)
- \* Max Sync Transfer Rate: 20 MB/Sec (IDs 0 - 7)
- \* Enable Disconnect: Yes (IDs 0 - 7)
- \* Send Start Unit Command: No (IDs 0 - 7)
- \* BIOS Multiple LUN Support: No (IDs 0 - 7)
- \* Include in bios scan Yes (IDs 0 - 7)
- \* Plug and Play SCAM support: Disabled
- \* Reset SCSI Bus at IC Init: Disabled

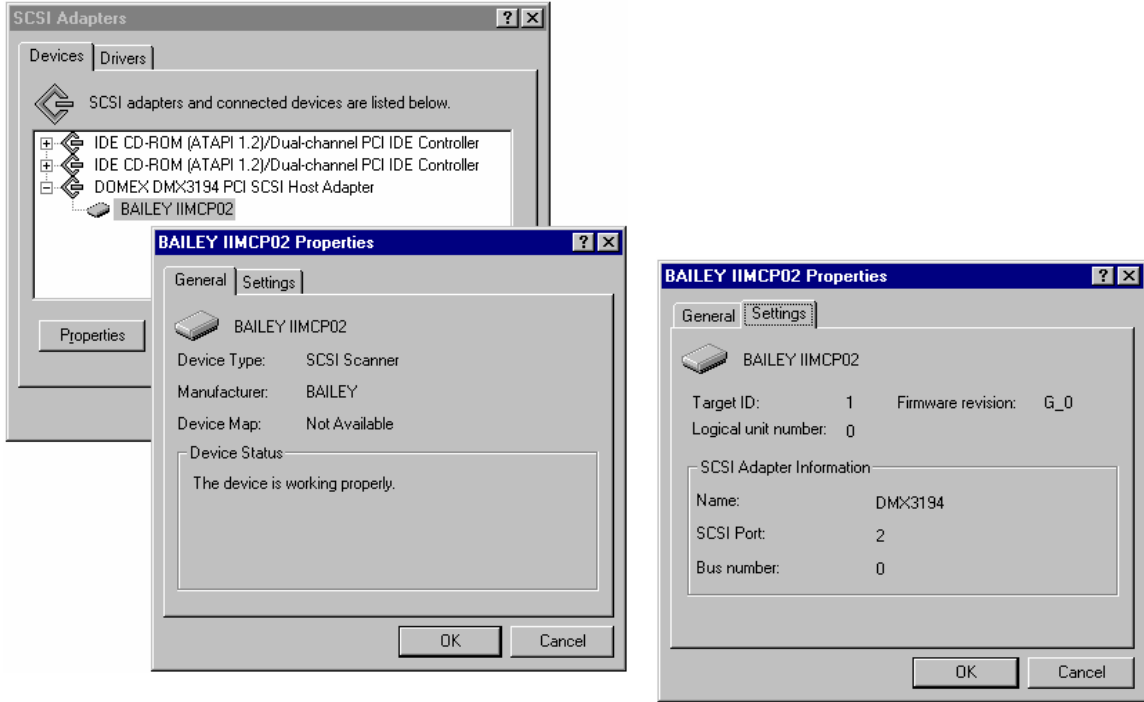
- \* Extended bios Translate for DOS ICs: Enabled
- \* BIOS support for Int 13 extensions: Enabled

The IDB driver utilizes the SCSI Pass Through Interface (SPTI) to communicate with the INICI03 via the host controller driver. The only other driver that needs to be loaded is the standard manufacturer driver provided with the SCSI card or the resident driver recommended by Windows. It is not necessary to load the Bailey semAPI driver. Note that for some plug and play systems, the INICI03 will be listed as "Another Device" when the system is first booted up. This is normal. Accept it as another device and request it to be disabled since a specific driver for that device will not be loaded. The properties for this other device will list the INICI03 as follows:



Notice that the INICI03 identifies itself to the SCSI adapter card as "BAILEY IIMCP02 SCSI Processor Device". Again this is normal. You do not need to reinstall a driver for this device type. Also notice the location of the device is given by the above dialog. For this example this device identity translates to an assigned SCSI address of "S0001:". Please use the SCSIscan utility to determine the proper SCSI addresses.

For systems that don't support plug and play (default NT installations), the INICI03 device will appear under control panel, SCSI Adapters as follows:



Notice that the INICI03 identifies itself to the SCSI adapter card as “BAILEY IIMCP02 SCSI Scanner” This is normal. The location of the device is given in the settings property dialog. For this example this device identity translates to an assigned SCSI address of “S2001:”. Please use the SCSIscan utility to determine the proper SCSI addresses.

## 7.11 Multi-State Device Driver (MSDD)

The MSDD DeltaV block is used to retrieve and control the exception reported output from a Bailey Multi-State Device Driver block (function code 129). Italicized attribute names indicate the values received from Bailey.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02).

| ATTRIBUTE       | TYPE                    | DESCRIPTION  |
|-----------------|-------------------------|--|
| IDB_ID          | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE         | String                  | Message describing state of the block operation.   |
| RING            | 8 Bit Unsigned Integer  | Bailey ring address.   |
| NODE            | 8 Bit Unsigned Integer  | Bailey node address.   |
| MODULE          | 8 Bit Unsigned Integer  | Bailey module address.   |
| BLOCK           | 16 Bit Unsigned Integer | Bailey block address.  |
| DISCONNECT      | 8 Bit Unsigned Integer  | When true requests that the data for this block be established in the Bailey interface but not connected. This stops data flow unless it goes in or out of an alarm condition. The functionality of this attribute is a possible future feature that is not currently implemented. |
| GS_LSD0         | String                  | Logic state descriptor for good state of zero.   |
| GS_LSD1         | String                  | Logic state descriptor for good state of one.  |
| GS_LSD2         | String                  | Logic state descriptor for good state of two.  |
| GS_LSD3         | String                  | Logic state descriptor for good state of three.  |
| F1_LSD0         | String                  | Logic state descriptor for feedback 1 of zero.   |
| F1_LSD1         | String                  | Logic state descriptor for feedback 1 of one.  |
| F2_LSD0         | String                  | Logic state descriptor for feedback 2 of zero.   |
| F2_LSD1         | String                  | Logic state descriptor for feedback 2 of one.  |
| F3_LSD0         | String                  | Logic state descriptor for feedback 3 of zero.   |
| F3_LSD1         | String                  | Logic state descriptor for feedback 3 of one.  |
| F4_LSD0         | String                  | Logic state descriptor for feedback 4 of zero.   |
| F4_LSD1         | String                  | Logic state descriptor for feedback 4 of one.  |
| DISC_LIM        | 8 Bit Unsigned Integer  | Discrete alarm state (see note 1). DISC_LIM should always equal 1. Making DISC_LIM equal to 0 may cause unpredictable alarm behavior.  |
| <i>DISC_ACT</i> | Alarm                   | Alarm active indicator. DISC_ACT is a function of feedback errors and not merely a function of DISC_LIM.   |



|                    |                         |  |
|--------------------|-------------------------|--|
| <i>FACE_TYPE</i>   | 8 Bit Unsigned Integer  | Bailey faceplate type code which is the value of FC 129 S18.   |
| <i>QUALITY</i>     | 8 Bit Unsigned Integer  | Quality of the data (0 = good, 1 = bad)  |
| <i>OUT_D</i>       | Discrete With Status    | Discrete output value and status of Bailey MSDD first control output signal.   |
| <i>RED_TAG</i>     | 8 Bit Unsigned Integer  | Bailey red tag indicator (0 = no tag, 1 = tagged)  |
| <i>MODE</i>        | Mode                    | The mode of the Bailey block. MODE contains the actual, target, permitted, and normal modes.   |
| <i>F1</i>          | 8 Bit Unsigned Integer  | State of feedback number 1.  |
| <i>F2</i>          | 8 Bit Unsigned Integer  | State of feedback number 2.  |
| <i>F3</i>          | 8 Bit Unsigned Integer  | State of feedback number 3.  |
| <i>F4</i>          | 8 Bit Unsigned Integer  | State of feedback number 4.  |
| <i>F1_T</i>        | String                  | Set to F1_LSD0 or F1_LSD1 based on current value of F1.  |
| <i>F2_T</i>        | String                  | Set to F2_LSD0 or F2_LSD1 based on current value of F2.  |
| <i>F3_T</i>        | String                  | Set to F3_LSD0 or F3_LSD1 based on current value of F3.  |
| <i>F4_T</i>        | String                  | Set to F4_LSD0 or F4_LSD1 based on current value of F4.  |
| <i>GOOD_STATE</i>  | 8 Bit Unsigned Integer  | Good state (see note 2).   |
| <i>REQ_STATE</i>   | 8 Bit Unsigned Integer  | Requested state (see note 2).  |
| <i>GS_T</i>        | String                  | Set to GS_LSD0, GS_LSD1, GS_LSD2 or GS_LSD3 based on current value of GOOD_STATE.  |
| <i>RS_T</i>        | String                  | Set to GS_LSD0, GS_LSD1, GS_LSD2 or GS_LSD3 based on current value of REQ_STATE.   |
| <i>TRAVEL</i>      | 8 Bit Unsigned Integer  | Travel indicator (see note 3).   |
| <i>OVR_CONTROL</i> | 8 Bit Unsigned Integer  | Override control indicator.  |
| <i>OVR_STATUS</i>  | 8 Bit Unsigned Integer  | Override status indicator.   |
| <i>SIM_TIME</i>    | 32 Bit Unsigned Integer | Defines simulation time in milliseconds. Simulation is active when the IDB SCHEME attribute is set to the "Simulated Interface" selection. Consult "Simulation of Bailey System" for additional details. |

Notes:

- 1.) This alarm limit is not associated with the state of the block output. It applies to the internal alarm indicator generated by the Bailey block. The Bailey block alarm indicator is always 0 equals no alarm and 1 equals alarm.
- 2.) The GOOD\_STATE and REQ\_STATE (requested state) attributes are used to view current operating state of the block and command its requested state. Valid values are: (zero = default, one = state 1,

two = state 2, and three = state 3). Writing any of these values to REQ\_STATE commands the MSDD to that associated state.

- 3.) Travel indicator is not actually received from Bailey but composed by the driver for display indication purposes. It is set for the interval of time between commanding a new requested state and waiting for the good state to arrive at the requested state while no alarm has been flagged by the Bailey MSDD block.

## 7.12 Multiplex CIU (MUXCIU)

The MUXCIU DeltaV block allows a configuration utility program, requiring communication access to the Bailey system, to share the CIU device associated with the IDB block. For example, this block will allow the Bailey CADEWS software to communicate to the Bailey system via the DeltaV Connect™ Solution for Bailey® Systems IDB driver. To accomplish this task, connect a RS232 serial cable between the COM port specified by the MUXCIU PORT\_A attribute and the COM port to which the configuration utility program expects the Bailey CIU interface to be attached. Note that this connection must be made with a null modem cable. The MUXCIU block receives CIU commands generated by the configuration utility program via the assigned serial connection. It channels those commands to the Bailey system through the DeltaV Connect Bailey IDB driver. The CIU reply is then returned to the configuration utility program via the assigned serial connection. Commands that would be detrimental to the operation of the DeltaV Connect Bailey interface are handled directly by the MUXCIU block. For example, a common command that a configuration utility program sends upon initially starting up, is the CIU RESTART command. The MUXCIU block intercepts this command and does not actually send it to the DeltaV Connect Bailey IDB driver. Instead it simply returns the response originally received when the DeltaV Connect Bailey IDB driver started up its Bailey interface and had sent the CIU RESTART command. The MUXCIU block has attributes to monitor communication health and statistics. Italicized attribute names indicate values updated by the MUXCIU driver in real time.

A RS232 switch box must be installed to use the MUXCIU block with redundant App Stations. This switch box will be used to connect the PC running the Bailey configuration utility software to the current Active App Station. The cable that normally attaches the Bailey configuration software PC COM port to the Bailey interface should be attached to the switch box common port. The switch box port A should be attached to the Active redundant App Station PC COM port associated with the MUXCIU block. This cable must be a standard RS232 null modem cable. A second null modem cable must be attached between the switch box port B and the Standby App Station PC COM port. Before using the Bailey configuration software, use the switch box to connect the configuration PC to the currently active App Station.

| ATTRIBUTE | TYPE                    | DESCRIPTION   |
|-----------|-------------------------|---|
| IDB_ID    | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).   |
| MESSAGE   | String                  | Message describing state of the block operation.  |
| PORT_A    | String                  | Name and baud rate of the COM port attached to the configuration utility program requiring access to the Bailey system. For example<br>"COM3: baud=19200 parity=N data=8 stop=1"<br>selects COM3, set to a baud rate of 19200, no parity, eight data bits and 1 stop bit (see note 1).  |
| DEBUG_LOG | Option Bitstring        | Enables various logging features that may be useful for diagnosing MUXCIU block operational performance. When any of the logging features are enabled, daily log files are generated in the DeltaV "DVDATA" directory. Leaving all but the error option enabled for extended periods of time can consume a lot of disk space. The four logging options supported are: |

|                   |                         |  |
|-------------------|-------------------------|--|
|                   |                         | <ol style="list-style-type: none"> <li>1.) Errors – post miscellaneous errors</li> <li>2.) Events – post communication events</li> <li>3.) Sends – post commands being sourced by the configuration utility program.</li> <li>4.) Receives – post replies sent back to the configuration utility program.</li> </ol> |
| <i>PRI_STATUS</i> | 8 Bit Unsigned Integer  | Communication status. A value of zero indicates good and one means bad.  |
| <i>MSG_TOTAL</i>  | 32 Bit Unsigned Integer | Running count of total messages being exchanged with the configuration utility program.  |
| <i>MSG_RATE</i>   | 32 Bit Unsigned Integer | Messages per second being exchanged with the configuration utility program.  |
| <i>NAK_TOTAL</i>  | 32 Bit Unsigned Integer | Running count of total negative acknowledgments sent to the configuration utility program.   |

Notes:

- 1.) Supported baud rates are 1200, 2400, 4800, 9600, 19200, 38400, 57600 and 115200. The baud rate setting should be set to the maximum value supported by the configuration utility program. The typical setting is 19200 baud.

### 7.13 Poll Any Block (POUT)

The POUT DeltaV block is used to retrieve polled output values from any Bailey function block output. This includes blocks that have floating point outputs and discrete outputs. Its primary usage is intended for node level data acquisition when the Bailey interface type is a serial port module, CPM02 or CIC. The POUT block can be used to duplicate the Bailey console ad hoc block output queries. Usage of this block with Bailey interfaces other than serial port modules, CPM or CIC should be limited to those few Bailey block output values not currently being exception reported. Data is obtained by polling. Since Bailey is optimized for exception reporting, polling for Bailey outputs is inefficient and should not be used for a large number of values. Italicized attribute names indicate the values received from Bailey.

| ATTRIBUTE         | TYPE                       | DESCRIPTION  |
|-------------------|----------------------------|--|
| IDB_ID            | 32 Bit Unsigned Integer    | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE           | String                     | Message describing state of the block operation.   |
| RING              | 8 Bit Unsigned Integer     | Bailey ring address.   |
| NODE              | 8 Bit Unsigned Integer     | Bailey node address.   |
| MODULE            | 8 Bit Unsigned Integer     | Bailey module address.   |
| BLOCK             | 16 Bit Unsigned Integer    | Bailey block address.  |
| INTERVAL          | 32 Bit Unsigned Integer    | Desired polling interval expressed as milliseconds.  |
| OUT_SCALE         | Scaling                    | High and low OUT limits along with DeltaV engineering units. Also includes declaration of digits to the right of the decimal point associated with OUT for display purposes. |
| OUT_HI_LIM        | Floating Point             | OUT high limit (duplicated in OUT_SCALE).  |
| OUT_LO_LIM        | Floating Point             | OUT low limit (duplicated in OUT_SCALE).   |
| QUALITY           | 8 Bit Unsigned Integer     | Quality of the data (0 = good, 1 = bad)  |
| <i>OUT</i>        | Floating Point With Status | The polled Bailey block output and status (see note 1).  |
| <i>HI_ACT</i>     | 8 Bit Unsigned Integer     | High alarm active indicator (see note 2).  |
| <i>LO_ACT</i>     | 8 Bit Unsigned Integer     | Low alarm active indicator (see note 2).   |
| <i>HI_DEV_ACT</i> | 8 Bit Unsigned Integer     | High deviation alarm active indicator (see note 3).  |
| <i>LO_DEV_ACT</i> | 8 Bit Unsigned Integer     | Low deviation alarm active indicator (see note 3).   |
| SIM_TYPE          | Named Set                  | Defines type of simulated value. Simulation is active when the IDB SCHEME attribute is set to the "Simulated Interface" selection.   |
| SIM_TIME          | 32 Bit Unsigned Integer    | Defines simulation time in milliseconds. Consult "Simulation of Bailey System" for additional details.   |

Notes:

- 1.) Polling discrete Bailey block outputs will be converted to an equivalent floating point value.
- 2.) Some polled Bailey floating point block outputs also contain high and low alarm active bits. The alarm level of any such polled block will be reflected in the HI\_ACT and LO\_ACT attributes. Some polled Bailey discrete block outputs also contain an alarm active bit. The alarm level of any such polled block will be reflected in the HI\_ACT attribute.
- 3.) Some polled Bailey floating point block outputs also contain high and low deviation alarm active bits. The alarm level of any such polled block will be reflected in the HI\_DEV\_ACT and LO\_DEV\_ACT attributes.

## 7.14 Remote Control Memory (RCM)

The RCM DeltaV block is used to retrieve and control the exception reported output from a Bailey Remote Control Memory block (function code 62). Italicized attribute names indicate the values received from Bailey.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02).

| ATTRIBUTE        | TYPE                    | DESCRIPTION  |
|------------------|-------------------------|--|
| IDB_ID           | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE          | String                  | Message describing state of the block operation.   |
| RING             | 8 Bit Unsigned Integer  | Bailey ring address.   |
| NODE             | 8 Bit Unsigned Integer  | Bailey node address.   |
| MODULE           | 8 Bit Unsigned Integer  | Bailey module address.   |
| BLOCK            | 16 Bit Unsigned Integer | Bailey block address.  |
| DISCONNECT       | 8 Bit Unsigned Integer  | When true requests that the data for this block be established in the Bailey interface but not connected. This stops data flow unless it goes in or out of an alarm condition. The functionality of this attribute is a possible future feature that is not currently implemented. |
| F1_LSD0          | String                  | Logic state descriptor for feedback 1 of zero.   |
| F1_LSD1          | String                  | Logic state descriptor for feedback 1 of one.  |
| OUT_D_LSD0       | String                  | Logic state descriptor for output of zero.   |
| OUT_D_LSD1       | String                  | Logic state descriptor for output of one.  |
| DISC_LIM         | 8 Bit Unsigned Integer  | Discrete alarm state (see note 1). DISC_LIM should always equal 1. Making DISC_LIM equal to 0 may cause unpredictable alarm behavior.  |
| <i>DISC_ACT</i>  | Alarm                   | Alarm active indicator.  |
| <i>FACE_TYPE</i> | 8 Bit Unsigned Integer  | Bailey faceplate type code which is value of FC 62 S8 (see note 2).  |
| <i>QUALITY</i>   | 8 Bit Unsigned Integer  | Quality of the data (0 = good, 1 = bad)  |
| <i>OUT_D</i>     | Discrete With Status    | The Bailey RCM discrete output value and status.   |
| <i>OUT_T</i>     | String                  | Set to OUT_D_LSD0 or OUT_D_LSD1 based on current value of OUT_D.   |
| <i>RED_TAG</i>   | 8 Bit Unsigned Integer  | Bailey red tag indicator (0 = no tag, 1 = tagged)  |
| <i>F1</i>        | 8 Bit Unsigned Integer  | State of feedback.   |
| <i>F1_T</i>      | String                  | Set to F1_LSD0 or F1_LSD1 based on current value of F1.  |
| <i>SET_PERM</i>  | 8 Bit Unsigned Integer  | Set permissive.  |

|                   |                         |  |
|-------------------|-------------------------|--|
| <i>OVR_STATUS</i> | 8 Bit Unsigned Integer  | Override status indicator  |
| <i>SIM_TIME</i>   | 32 Bit Unsigned Integer | Defines simulation time in milliseconds. Simulation is active when the IDB SCHEME attribute is set to the "Simulated Interface" selection. Consult "Simulation of Bailey System" for additional details. |

Notes:

- 1.) This alarm limit is not associated with the state of the block output. It applies to the internal alarm indicator generated by the Bailey block. The Bailey block alarm indicator is always 0 equals no alarm and 1 equals alarm.
- 2.) This attribute is used by the standard DVC-Bailey RCM faceplate to determine what combination of output and feedback indicators should be displayed. The combinations are according to the Bailey function code manual which are:
  - 0 = output indicator only
  - 1 = no indicators (output or feedback)
  - 2 = output and feedback indicators
  - 3 = feedback indicator only



## 7.15 Remote Motor Control (RMC)

The RMC DeltaV block is used to retrieve and control the exception reported output from a Bailey Remote Motor Control block (function code 136). Italicized attribute names indicate the values received from Bailey.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02).

| ATTRIBUTE       | TYPE                    | DESCRIPTION  |
|-----------------|-------------------------|--|
| IDB_ID          | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE         | String                  | Message describing state of the block operation.   |
| RING            | 8 Bit Unsigned Integer  | Bailey ring address.   |
| NODE            | 8 Bit Unsigned Integer  | Bailey node address.   |
| MODULE          | 8 Bit Unsigned Integer  | Bailey module address.   |
| BLOCK           | 16 Bit Unsigned Integer | Bailey block address.  |
| DISCONNECT      | 8 Bit Unsigned Integer  | When true requests that the data for this block be established in the Bailey interface but not connected. This stops data flow unless it goes in or out of an alarm condition. The functionality of this attribute is a possible future feature that is not currently implemented. |
| F1_LSD0         | String                  | Logic state descriptor for feedback 1 when zero.   |
| F1_LSD1         | String                  | Logic state descriptor for feedback 1 when one.  |
| F2_LSD0         | String                  | Logic state descriptor for feedback 2 when zero.   |
| F2_LSD1         | String                  | Logic state descriptor for feedback 2 when one.  |
| PERM1_LSD0      | String                  | Logic state descriptor for 1 <sup>st</sup> permissive when zero.   |
| PERM1_LSD1      | String                  | Logic state descriptor for 1 <sup>st</sup> permissive when one.  |
| PERM2_LSD0      | String                  | Logic state descriptor for 2 <sup>nd</sup> permissive when zero.   |
| PERM2_LSD1      | String                  | Logic state descriptor for 2 <sup>nd</sup> permissive when one.  |
| OUT_D_LSD0      | String                  | Logic state descriptor for output of zero.   |
| OUT_D_LSD1      | String                  | Logic state descriptor for output of one.  |
| DISC_LIM        | 8 Bit Unsigned Integer  | Discrete alarm state (see note 1). DISC_LIM should always equal 1. Making DISC_LIM equal to 0 may cause unpredictable alarm behavior.  |
| <i>DISC_ACT</i> | Alarm                   | Alarm active indicator. DISC_ACT is a function of feedback errors and not merely a function of DISC_LIM.   |

|                    |                         |  |
|--------------------|-------------------------|--|
| <i>FACE_TYPE</i>   | 8 Bit Unsigned Integer  | Bailey faceplate type code which is value of FC 136 S14.   |
| <i>QUALITY</i>     | 8 Bit Unsigned Integer  | Quality of the data (0 = good, 1 = bad)  |
| <i>OUT_D</i>       | Discrete With Status    | The Bailey discrete output value and status.   |
| <i>OUT_T</i>       | String                  | Set to <i>OUT_D_LSD0</i> or <i>OUT_D_LSD1</i> based on current value of <i>OUT_D</i> .   |
| <i>RED_TAG</i>     | 8 Bit Unsigned Integer  | Bailey red tag indicator (0 = no tag, 1 = tagged)  |
| <i>F1</i>          | 8 Bit Unsigned Integer  | State of feedback number 1.  |
| <i>F2</i>          | 8 Bit Unsigned Integer  | State of feedback number 2.  |
| <i>F1_T</i>        | String                  | Set to <i>F1_LSD0</i> or <i>F1_LSD1</i> based on current value of <i>F1</i> .  |
| <i>F2_T</i>        | String                  | Set to <i>F2_LSD0</i> or <i>F2_LSD1</i> based on current value of <i>F2</i> .  |
| <i>FAULT</i>       | 8 Bit Unsigned Integer  | Fault has occurred indicator.  |
| <i>ERR_CODE</i>    | 8 Bit Unsigned Integer  | Fault error code (see note 2).   |
| <i>FAULT_ACK</i>   | 8 Bit Unsigned Integer  | Fault acknowledgment (see note 3).   |
| <i>PERM1</i>       | 8 Bit Unsigned Integer  | Permissive #1 indicator.   |
| <i>PERM2</i>       | 8 Bit Unsigned Integer  | Permissive #2 indicator.   |
| <i>PERM1_T</i>     | String                  | Set to <i>PERM1_LSD0</i> or <i>PERM1_LSD1</i> based on current value of <i>PERM1</i> .   |
| <i>PERM2_T</i>     | String                  | Set to <i>PERM2_LSD0</i> or <i>PERM2_LSD1</i> based on current value of <i>PERM2</i> .   |
| <i>BAD_START</i>   | 8 Bit Unsigned Integer  | Bad start indicator.   |
| <i>HOLD_STATUS</i> | 8 Bit Unsigned Integer  | Status on hold indicator.  |
| <i>SIM_TIME</i>    | 32 Bit Unsigned Integer | Defines simulation time in milliseconds. Simulation is active when the IDB SCHEME attribute is set to the "Simulated Interface" selection. Consult "Simulation of Bailey System" for additional details. |

**Notes:**

- 1.) This alarm limit is not associated with the state of the block output. It applies to the internal alarm indicator generated by the Bailey block. The Bailey block alarm indicator is always 0 equals no alarm and 1 equals alarm.
- 2.) The *ERR\_CODE* attribute indicates error codes when a bad start or fault condition arises. The following error codes can be returned:
  - 0 - no error,
  - 1 - stop input,
  - 2 - interlock #1 input
  - 3 - interlock #2 input
  - 4 - interlock #3 input
  - 5 - interlock #4 input
  - 6 - feedback #1 input is 0
  - 7 - feedback #2 input is 0

- 8 - feedback #1 input is 1
  - 9 - feedback #1 input is 1
- 3.) The FAULT\_ACK attribute should be set true (1) to acknowledge a fault or bad start condition and the Bailey RMC block will reset it to false (0) after acknowledgment is accepted. Acceptance of the fault acknowledgment does not necessarily mean the fault indicator (FAULT) will reset. The fault indicator will clear only after the condition that caused the fault is corrected and the next motor start is initiated.

## 7.16 Remote Manual Set Constant (RMSC)

The RMSC DeltaV block is used to retrieve and control the exception reported output from a Bailey Remote Manual Set Constant block (function code 68). Italicized attribute names indicate the values received from Bailey.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02).

| <b>ATTRIBUTE</b>   | <b>TYPE</b>                | <b>DESCRIPTION</b>   |
|--------------------|----------------------------|--|
| IDB_ID             | 32 Bit Unsigned Integer    | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE            | String                     | Message describing state of the block operation.   |
| RING               | 8 Bit Unsigned Integer     | Bailey ring address.   |
| NODE               | 8 Bit Unsigned Integer     | Bailey node address.   |
| MODULE             | 8 Bit Unsigned Integer     | Bailey module address.   |
| BLOCK              | 16 Bit Unsigned Integer    | Bailey block address.  |
| DISCONNECT         | 8 Bit Unsigned Integer     | When true requests that the data for this block be established in the Bailey interface but not connected. This stops data flow unless it goes in or out of an alarm condition. The functionality of this attribute is a possible future feature that is not currently implemented. |
| <i>EU_CODE</i>     | 8 Bit Unsigned Integer     | Bailey engineering units code which is value of FC 68 S1.  |
| <i>QUALITY</i>     | 8 Bit Unsigned Integer     | Quality of the data (0 = good, 1 = bad)  |
| <i>OUT</i>         | Floating Point With Status | Current output value and status received from Bailey.  |
| <i>OUT_SCALE</i>   | Scaling                    | High and low OUT limits (received from Bailey) along with DeltaV engineering units. Also includes declaration of digits to the right of the decimal point associated with OUT for display purposes.  |
| <i>OUT_HI_LIM</i>  | Floating Point             | OUT high limit (duplicated in OUT_SCALE).  |
| <i>OUT_LO_LIM</i>  | Floating Point             | OUT low limit (duplicated in OUT_SCALE).   |
| <i>SP_TRACKING</i> | 8 Bit Unsigned Integer     | Set point is tracking indicator.   |

## 7.17 Module Status (STAT)

The STAT DeltaV block is used to retrieve module status summary information and detailed module problem reports. This block is designed to work with all Bailey node and module types. Italicized attribute names indicate the values received from Bailey.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02). The Bailey CIU01 does not support reading problem reports from itself and therefore only module status of the CIU01 can be provided by this block.

| ATTRIBUTE                   | TYPE                    | DESCRIPTION   |
|-----------------------------|-------------------------|---|
| IDB_ID                      | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).   |
| MESSAGE                     | String                  | Message describing state of the block operation.  |
| RING                        | 8 Bit Unsigned Integer  | Bailey ring address (note 1).   |
| NODE                        | 8 Bit Unsigned Integer  | Bailey node address (note 1).   |
| MODULE                      | 8 Bit Unsigned Integer  | Bailey module address (note 1).   |
| COLUMNS                     | 8 Bit Unsigned Integer  | Number of columns (characters) per line to use when generating problem report text. Values from 50 to 130 are allowed.  |
| READ                        | 8 Bit Unsigned Integer  | Set this attribute to request module problem reports to be read for the Bailey module status defined for this block. The problem reports are written to the LINE01 to LINE50 attributes. The driver will reset this attribute when the read request is completed. |
| NEXT                        | 8 Bit Unsigned Integer  | Set this attribute to request a read of the next group of problem reports. The problem reports are written to the LINE01 to LINE50 attributes. The driver will reset this attribute when the next problem report read request is completed.                       |
| LINES                       | 8 Bit Unsigned Integer  | Number of lines to utilize when generating problem report text. Values from 2 to 50 are allowed.  |
| QUALITY                     | 8 Bit Unsigned Integer  | Quality of the data (0 = good, 1 = bad)   |
| OUT_D                       | Discrete With Status    | Indicates whether or not the Bailey module is currently experiencing any errors. A value of zero indicates no errors. A value of one indicates one or more errors. They can be determined by examining the STATUS_TEXT attribute and requesting problem reports.  |
| <i>LINE01 to<br/>LINE50</i> | String                  | Requested problem report text strings are written to these attributes.  |
| <i>STATUS_TEXT</i>          | String                  | Overview status of the addressed Bailey module (see notes 2).   |

|                                       |                        |  |
|---------------------------------------|------------------------|--|
| <i>STAT_BYTE01 to<br/>STAT_BYTE16</i> | 8 Bit Unsigned Integer | Bailey module status bytes. Only the first 5 are valid for Command Series and Network 90 systems. All sixteen are valid for Infi 90 systems. |
|---------------------------------------|------------------------|--|

**Notes:**

- 1.) When retrieving the module status of a CIU interface, special attention must be given to the settings of ring, node and module. For all CIU interfaces except the CIU01, the ring must match the ring address of the CIU, node must match the node address of the CIU and module must be set to a value of two. For the CIU01 interface, ring and node must be set to a value of zero and module to a value of two. Failure to follow these rules will result in improper or no module status returned for the CIU interface.
- 2.) The status of the Bailey module is returned as a string in the following format "<Type>, <State>, <Errors>" where type indicates the module type (AMM, MFC, COM, etc), state indicates its current state of operation (Execute, Configure, Error) and errors are a summary of any current errors it is experiencing. The text for module state and type are defined in the DVC\_BLY.INI file stored in the windows base directory. It may be edited to customize message generation. The [MODULE STATE] section defines the text for module state. The [MODULE TYPE] section defines the text for module type. You may edit the names or add new type codes and associated names that correspond to new Bailey modules introduced after any given revision to the DeltaV Connect - Bailey.

## 7.18 Station PID Control (STN)

The STN DeltaV block is used to retrieve and control the exception reported outputs from Bailey Control Station blocks (function codes 21, 22, 23 & 80). Italicized attribute names indicate the values received from Bailey.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02).

| ATTRIBUTE       | TYPE                    | DESCRIPTION  |
|-----------------|-------------------------|--|
| IDB_ID          | 32 Bit Unsigned Integer | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE         | String                  | Message describing state of the block operation.   |
| RING            | 8 Bit Unsigned Integer  | Bailey ring address.   |
| NODE            | 8 Bit Unsigned Integer  | Bailey node address.   |
| MODULE          | 8 Bit Unsigned Integer  | Bailey module address.   |
| BLOCK           | 16 Bit Unsigned Integer | Bailey block address.  |
| DISCONNECT      | 8 Bit Unsigned Integer  | When true requests that the data for this block be established in the Bailey interface but not connected. This stops data flow unless it goes in or out of an alarm condition. The functionality of this attribute is a possible future feature that is not currently implemented. |
| PID_BLOCK       | 16 Bit Signed Integer   | Bailey PID block address associated with this control station (see note 1).  |
| FC              | 8 Bit Unsigned Integer  | Bailey PID function code (18, 19, 156).  |
| ALARM_HYS       | Floating Point          | The amount the alarm value must return within the alarm limit before the associated active alarm condition clears. (see note 2).   |
| HI_HI_LIM       | Floating Point          | The setting for the alarm limit used to detect the PV high high alarm condition (see note 2).  |
| LO_LO_LIM       | Floating Point          | The setting for the alarm limit used to detect the PV low low alarm condition (see note 2).  |
| QUALITY         | 8 Bit Unsigned Integer  | Quality of the data (0 = good, 1 = bad).   |
| <i>EU_CODE</i>  | 8 Bit Unsigned Integer  | Bailey engineering units code which is value of FC 21-23 S13 or FC 80 S12.   |
| <i>STN_TYPE</i> | 8 Bit Unsigned Integer  | Bailey station type which is value of FC 80 S23: (1 - basic with SP, 2 - ratio, 4 - cascade) or implied by FC 21, 22, 23.  |
| <i>RED_TAG</i>  | 8 Bit Unsigned Integer  | Bailey red tag indicator (0 = no tag, 1 = tagged).   |
| <i>MODE</i>     | Mode                    | The mode of the Bailey block. MODE contains the actual, target, permitted, and normal  |

|                     |                            |   |
|---------------------|----------------------------|---|
|                     |                            | modes.  |
| <i>MODE_LOCK</i>    | 8 Bit Unsigned Integer     | Indicates locked into current mode (when set).  |
| <i>SP_TRACKING</i>  | 8 Bit Unsigned Integer     | Indicates set point tracking (when set).  |
| <i>SP</i>           | Floating Point With Status | Current control station set point.  |
| <i>SP_HI_LIM</i>    | Floating Point             | SP high limit always set to the value received for the PV high limit.   |
| <i>SP_LO_LIM</i>    | Floating Point             | SP low limit always set to the value received for the PV low limit.   |
| <i>PV</i>           | Floating Point With Status | Current control station process value.  |
| <i>RI</i>           | Floating Point With Status | Current control station ratio index value.  |
| <i>OUT_TRACKING</i> | 8 Bit Unsigned Integer     | Indicates control output tracking (when set).   |
| <i>OUT</i>          | Floating Point With Status | Current control output value and status received from Bailey.   |
| <i>OUT_SCALE</i>    | Scaling                    | High and low OUT limits along with engineering units code. Also includes number of digits to the right of the decimal point associated with OUT (see note 3). |
| <i>OUT_HI_LIM</i>   | Floating Point             | OUT high limit (duplicated in OUT_SCALE see note 3).  |
| <i>OUT_LO_LIM</i>   | Floating Point             | OUT low limit (duplicated in OUT_SCALE see note 3).   |
| <i>HI_HI_ACT</i>    | Alarm                      | PV High high alarm active indicator (see note 2).   |
| <i>HI_ACT</i>       | Alarm                      | PV High alarm active indicator.   |
| <i>LO_ACT</i>       | Alarm                      | PV Low alarm active indicator.  |
| <i>LO_LO_ACT</i>    | Alarm                      | PV Low low alarm active indicator (see note 2).   |
| <i>PV_SCALE</i>     | Scaling                    | High and low PV limits along with engineering units code. Also includes number of digits to the right of the decimal point associated with PV.                |
| <i>PV_HI_LIM</i>    | Floating Point             | PV high limit (duplicated in PV_SCALE).   |
| <i>PV_LO_LIM</i>    | Floating Point             | PV low limit (duplicated in PV_SCALE).  |
| <i>HI_LIM</i>       | Floating Point             | The setting for the alarm limit used to detect the PV high alarm condition.   |
| <i>LO_LIM</i>       | Floating Point             | The setting for the alarm limit used to detect the PV low alarm condition.  |
| <i>DV_HI_LIM</i>    | Floating Point             | PV Deviation high limit.  |
| <i>DV_LO_LIM</i>    | Floating Point             | PV Deviation low limit.   |
| <i>DV_HI_ACT</i>    | Alarm                      | High deviation alarm active indicator (see note 4).   |
| <i>DV_LO_ACT</i>    | Alarm                      | Low deviation alarm active indicator (see note  |



|                  |                        |  |
|------------------|------------------------|--|
| <i>GAIN</i>      | Floating Point         | 4).<br>Overall PID block gain term (see note 5).   |
| <i>P_TERM</i>    | Floating Point         | PID block proportional gain value (see note 5).  |
| <i>I_TERM</i>    | Floating Point         | PID block integral action time constant (see note 5).  |
| <i>D_TERM</i>    | Floating Point         | PID block derivative action time constant (see note 5).  |
| <i>DLAG_TERM</i> | Floating Point         | Advance PID (only) block derivative lag action time constant (see note 5).                         |
| <i>DIR</i>       | 8 Bit Unsigned Integer | PID block direction (see note 5):<br>(0 = reverse error, SP – PV),<br>(1 = direct error, PV – SP). |
| <i>IONLY</i>     | 8 Bit Unsigned Integer | PID block set point modifier (see note 5):<br>(0 = normal, 1 = integral only on SP change).        |
| <i>AO_BYPASS</i> | 8 Bit Unsigned Integer | Indicates analog output bypass (when set).   |
| <i>DS_BAD</i>    | 8 Bit Unsigned Integer | Indicates digital station (hard DCS station) is bad (when set).                                    |

Notes:

- 1.) The default value of zero instructs the driver to automatically determine the PID block number associated with the Bailey control station block. It checks the block number configured for the Bailey Control Station function block auto input. If the block is not a PID the PID\_BLOCK attribute is set to a value of –1 indicating auto determination failed and a user entered block number is required. Until an actual PID block is determined the OUT\_HI\_LIM, OUT\_LO\_LIM, GAIN, P\_TERM, I\_TERM, D\_TERM and DLAG\_TERM remain unknown and left at their last configured state.
- 2.) Data received from this set of Bailey function blocks does not support two level PV alarm capabilities. The HI\_LIM and LO\_LIM attributes are received from Bailey but the ALARM\_HYS, HI\_HI\_LIM and LO\_LO\_LIM must be configured by the DeltaV system which is processed by the DeltaV STN block.
- 3.) OUT\_HI\_LIM and OUT\_LO\_LIM are the high and low limit specifications read from the Bailey PID block associated with the Bailey Control Station block. Writing these values from the DeltaV system causes the driver to automatically tune them in the appropriate Bailey PID block specification.
- 4.) For compatibility with DeltaV handling of deviation these values are reported as separate attributes. Bailey does not support different settings for low and high deviation alarm limits so although reported as two separate attributes to DeltaV there values are always forced to be the same for compatibility with Bailey.
- 5.) GAIN, P\_TERM, I\_TERM, D\_TERM, DLAG\_TERM, IONLY and DIR are specifications read from the Bailey PID block associated with the Bailey Control Station block. Writing these values from the DeltaV system causes the driver to automatically tune them in the appropriate Bailey PID block specification.

## 7.19 Text Selector (TXT)

The TXT DeltaV block is used to retrieve the exception reported output from Bailey Text Selector blocks (function code 151). Italicized attribute names indicate the values received from Bailey.

Restrictions: This DeltaV block can be utilized with all Bailey interface types except serial port module (SPM & CPM02).

| ATTRIBUTE        | TYPE                                | DESCRIPTION  |
|------------------|-------------------------------------|--|
| IDB_ID           | 32 Bit Unsigned Integer             | Associated Interface Definition Block ID (See IDB block).  |
| MESSAGE          | String                              | Message describing state of the block operation.   |
| RING             | 8 Bit Unsigned Integer              | Bailey ring address.   |
| NODE             | 8 Bit Unsigned Integer              | Bailey node address.   |
| MODULE           | 8 Bit Unsigned Integer              | Bailey module address.   |
| BLOCK            | 16 Bit Unsigned Integer             | Bailey block address.  |
| DISCONNECT       | 8 Bit Unsigned Integer              | When true requests that the data for this block be established in the Bailey interface but not connected. This stops data flow unless it goes in or out of an alarm condition. The functionality of this attribute is a possible future feature that is not currently implemented. |
| <i>QUALITY</i>   | 8 Bit Unsigned Integer              | Quality of the data (0 = good, 1 = bad)  |
| <i>MSG_OUT</i>   | 32 Bit Unsigned Integer With Status | Message number to be displayed.  |
| <i>MSG_OUT_T</i> | String                              | Message text associated with the current value of <i>MSG_OUT</i> (see note 1).   |
| <i>COLOR</i>     | 8 Bit Unsigned Integer              | Color selection for the message.   |
| <i>BLINK</i>     | 8 Bit Unsigned Integer              | Blink the message flag.  |
| <i>SIM_TYPE</i>  | Named Set                           | Defines type of simulated value. Simulation is active when the IDB SCHEME attribute is set to the "Simulated Interface" selection.   |
| <i>SIM_TIME</i>  | 32 Bit Unsigned Integer             | Defines simulation time in milliseconds. Consult "Simulation of Bailey System" for additional details.   |

Notes:

- 1.) The "Dvc\_Bly\_Text.ini" file contains text strings to be associated with any given value of the *MSG\_OUT* attribute. This file can be found in the DeltaV\DVData\ABB directory. Using notepad, edit "Dvc\_Bly\_Text.ini" to define the required associations between *MSG\_OUT* and *MSG\_OUT\_T* values. Each text message can be up to 132 characters in length. Message numbers are always positive whole numbers starting with zero. The *MSG\_OUT\_T* attribute is updated whenever a different value for *MSG\_OUT* is received.

## 8 Simulation of the Bailey System

Setting the IDB block scheme attribute to “simulate interface” causes the IDB block and all other blocks associated with it to automatically enter a simulation mode of operation. The IDB block simulates a redundant channel connection to an INFI 90 Bailey system. This simulation closely matches the data update rates that can be expected when actually attached to a Bailey system. Simulation is a useful tool for demonstrating the operational characteristics of the DVC-Bailey interface. It can also be utilized for system configuration checks prior to connection with the Bailey interface. For example simulation will detect illegal or duplicate Bailey addresses entered for any given DVC-Bailey block just as is done when actually connected to the Bailey system. Simulation includes all blocks supported by the DVC-Bailey interface.

Some of the blocks include simulation attributes (SIM\_TYPE and / or SIM\_TIME) that allow configuration of simulation characteristics for the specific block. For blocks that return analog values (AIL, DANG, POUT and TXT) the SIM\_TYPE attribute allows selection of the following types of simulation.

- "Sine Wave No Alarm"
- "Sine Wave With Alarm"
- "Saw Tooth No Alarm"
- "Saw Tooth With Alarm"
- "Drifting Value No Alarm"
- "Drifting Value With Alarm"
- "Random Value No Alarm"
- "Random Value With Alarm"
- "Range Alternate"
- "Constant at EU0"
- "Constant at Half EU Span"
- "Constant at EU100"

Notice from these selections that it is possible to select simulated values that stay within the alarm limits (“No Alarm”), or move into and out of the alarm limits (“With Alarm”). These blocks also include the SIM\_TIME attribute which can be configured with a range of 100 to 4,294,967,295 milliseconds. The function of this attribute is based on the current SIM\_TYPE selection. For wave form simulation (Sine and Saw), the SIM\_TIME attribute defines the period of the wave. For all other simulation types, it defines how often the simulated value is updated.

For blocks that return digital values (DIL and DADIG) the SIM\_TYPE attribute allows selection of the following types of simulation.

- "Toggle state"
- "Random state"

- "Always zero"
- "Always one"

These blocks also include the SIM\_TIME attribute which defines how often the simulated value is changed.

Some blocks (DD, MSDD, RCM and RMC) only require the SIM\_TIME attribute. The purpose of the time it defines is block specific and explained in the simulation section pertaining to each of these block types.

The remaining blocks (AOL, DOL, BLK, BLKVAL, RMSC, STAT and STN) do not require the SIM\_TYPE and SIM\_TIME attributes to accomplish simulation.

#### *AIL Simulation*

Simulated values for this block are configurable using the SIM\_TYPE and SIM\_TIME attributes. For simulation that includes alarmed values, the simulated values will reside within the range configured by the OUT\_SCALE attribute. When the simulation type excludes alarmed values, the simulated values will stay between the LO\_LIM and HI\_LIM settings.

#### *AOL Simulation*

This block operates in identical fashion whether or not simulation is in effect. The output value and status simply follow the input value and status.

#### *BLK and BLKVAL Simulation*

Simulated values for these blocks are based on the Bailey module address requested to be read or tuned. Attempting to read blocks from modules other than 2 through 31 return an "invalid block number error". Otherwise, simulated Bailey block configuration information is returned for modules 2 through 31. Simulated blocks read are those blocks supported by the DVC-Bailey interface along with the typical Bailey executive blocks, segment control block and last block indicator.

#### *DANG Simulation*

Simulated values for this block are configurable using the SIM\_TYPE and SIM\_TIME attributes. For simulation that includes alarmed values, the simulated values will reside within the range configured by the OUT\_SCALE attribute. When the simulation type excludes alarmed values, the simulated values will stay between the LO\_LIM and HI\_LIM settings. Note that alarms can still occur if the LO\_LO\_LIM, LO\_LO\_LO\_LIM

are not set to values below LO\_LIM and HI\_HI\_LIM, HI\_HI\_HI\_LIM are not set to values above HI\_LIM.

### *DD Simulation*

Simulation involves having the two feedback signals follow the output state. When the output state is changed, a random device transition time is generated. This transition time is anywhere from zero to the value specified by SIM\_TIME. The first feedback signal will follow the output state change after ½ the device transition time elapses. The second feedback signal will follow the output state change after the total device transition time transpires. Upon startup, the block mode is randomly selected. When the device mode is auto, the output is cycled between set and reset at a rate equal to the randomly selected device transition time. Also upon startup, the block is randomly selected to be in alarm mode indicating the feedback signals do not match the output state. Once this condition is cleared by changing the device output state it is never generated a second time.

### *DIL Simulation*

Simulated state changes for this block are configurable using the SIM\_TYPE and SIM\_TIME attributes. State changes can be toggling or randomly selected values. The time in between state changes is specified by the SIM\_TIME attribute. Simulation selection also includes a steady state of zero (off) or one (on).

### *DOL Simulation*

This block operates in identical fashion whether or not simulation is in effect. The output value and status simply follow the input value and status.

### *IDB Simulation*

Simulated values for this block are the communication status, message counters and rate outputs. Simulation of these values closely matches those that can be achieved when actually communicating with the Bailey system. The calculated simulated values are based on the number of DVC-Bailey blocks, their types and configured IDB block exception report update and polling rates. Since these values so closely resemble actual interfacing conditions, the IDB block MESSAGE attribute indicates the current operating mode as “Simulated interface”. This information can be seen using the IDB block faceplate.

### *MSDD Simulation*

Simulation involves having the four feedback signals change state to a pre-defined combination based on the new requested state. For the default state all feedback signals are reset. State one has the first two on and the last two off. State two has the first two off and the last two on. State three has all feedback signals on. When a new requested state is initiated, a random device transition time is generated. This transition time is anywhere from zero to the value specified by SIM\_TIME. The four feedback signals change to their new state based on the device transition time. Feedback one arrives in  $\frac{1}{4}$  of the time, feedback two in  $\frac{1}{2}$ , feedback three in  $\frac{3}{4}$  and feedback four after the entire device transition time transpires. Upon startup, the block mode is randomly selected. When the device mode is auto, the requested state is cycled default, state 1, state 2 and state 3 at a rate equal to the randomly selected device transition time. Also upon startup, the block is randomly selected to be in alarm mode indicating the feedback signals do not match the requested state. Once this condition is cleared by changing a new requested state it is never generated a second time.

### *POUT Simulation*

Simulated values for this block are configurable using the SIM\_TYPE and SIM\_TIME attributes. For simulation that includes alarmed values, the simulated values will reside within the range configured by the OUT\_SCALE attribute. Alarmed values are predefined to be any value within 10% of the limits defined by OUT\_SCALE. When the simulation type excludes alarmed values, the simulated values will stay under 10% of the limits defined by OUT\_SCALE.

### *RCM Simulation*

Simulation involves having the feedback signal follow the output state. When the output state is changed, a random device transition time is generated. This transition time is anywhere from zero to the value specified by SIM\_TIME. The feedback signal will follow the output state change after the total device transition time transpires. Randomly one out of nine attempts to set the output will result in a failed feedback signal with the set permissive being lost. This condition is recovered by resetting the output.

### *RMC Simulation*

Simulation involves having the two feedback signals follow the output state. When the output state is changed, a random device transition time is generated. This transition time is anywhere from zero to the value specified by SIM\_TIME. The first feedback signal will follow the output state change after  $\frac{1}{2}$  the device transition time elapses. The second feedback signal will follow the output state change after the total device

transition time transpires. Randomly one out of nine attempts to set the output will result in failed start condition to occur. A randomly selected failure condition will be generated and posted to the block ERROR attribute which is enumerated by the block faceplate.

### *RMSC Simulation*

This block operates in nearly identical fashion to when simulation is not in effect. The only difference being the output value is randomly selected upon startup instead of being the last value entered into the Bailey controller, but thereafter it follows values written by the operator.

### *STAT Simulation*

Simulated values for this block are based on configuration of the Bailey module address attribute. Values of zero and one simulate the module status of a Bailey Infinet PCU experiencing remote I/O problems. Values of two to 31 simulate the modules status of a Bailey MFC also experiencing remote I/O problems. Simulated problem reports for both of these simulated module statuses are also generated when requested from the STAT module faceplate.

### *STN Simulation*

Simulated values for this block will reside within the range configured by the OUT\_SCALE and PV\_SCALE attributes. Upon startup a random mode and values for the process variable and set point are generated. Regardless of the random mode selected, the control output starts at a value of 50%. For auto and cascade modes this value is adjusted as the process variable randomly drifts and set point adjustments are made by the operator (auto mode) and control loop (cascade mode). The control loop acts like an extremely well tuned loop. Set point changes cause the process variable to slowly ramp towards the new set point value. Simulated PID tuning parameters can be read and written via the included detailed faceplate but have no effect on changing the performance of the “well tuned loop”.

### *TEXT Simulation*

Simulated values for this block are configurable using the SIM\_TYPE and SIM\_TIME attributes. The message number range is predefined from zero to 100. Since this block does not include alarms, selection of simulation that includes alarmed values will function identical to the selection that does not include alarm values. The blink and color number are randomly selected every SIM\_TIME interval. The color number range is predefined to be zero to 127.

## 9 Troubleshooting Hints

This section is provided to help the user identify and correct problems that may arise as a result of incorrectly setting up the DVCBailey interface. It is provided as a general guide to allow the user to decipher normal and abnormal operation. If this does not help, DVCBailey can be enabled to post additional error messages to DVCBailey log files using the IDB block DEBUG\_LOG attribute. Use this feature to track down tough problems and discover hard to find configuration errors. Afterwards remember to disable all of its options (except Errors), since leaving them enabled can consume large amounts of disk file space.

After downloading the DVCBailey for the first time and thereafter when it is booted up, the DVCBailey IDB block driver will automatically begin communicating with the Bailey interface. The startup pattern will vary based on the type of Bailey interface being utilized. The first thing you should notice is the Bailey interface serial processing card LEDs begin to sequence. Shortly thereafter you may hear the loop interface termination unit relays click on and off several times as the driver is identifying the Bailey interface type. Just prior to downloading the DVCBailey block database to the Bailey interface, it will be restarted, and the loop interface termination unit relays will click off isolating it from the communication loop. Next you should observe the Bailey interface serial processing card LEDs sequence at a steady rate as the DVCBailey block database is being downloaded. Upon completing the database download, the Bailey interface will be commanded on-line, at which time the loop interface termination unit relays will click on and the loop interface card LEDs begin to count loop messages. Thereafter, the Bailey interface serial processing card LEDs will sequence steadily based on the exception report poll interval setup by in the IDB block and individual DVCBailey POUT blocks (when utilized).

If you experience problems with establishing communication between the DVCBailey and the Bailey interface, if possible, it is a good idea to verify the setup by trying to communicate using the Bailey TXTEWS software. Generally if this software functions OK, you should not experience problems with DVCBailey.

All Bailey interfaces have a series of four or eight red LEDs on the hardware module that processes serial communication and manages its database. Don't confuse this card with the module that handles the interface with the Bailey communication loop which also has a series of LEDs. An indication that communication with the BAILEY interface is occurring can be determined by looking at the serial processing card LEDs. (Hereafter, these LEDs will be referenced as Bailey interface LEDs.) The Bailey interface LEDs count commands and replies occurring between the DVCBailey Application Station computer and the Bailey interface.



## 9.1 Validating State Of Individual DVCBailey Blocks

All DVCBailey blocks have an attribute called MESSAGE. The purpose of this attribute is to help diagnose the operational state of the block. This attribute can take on the following messages:

**OFFLINE:** The block is being edited by Control Studio offline or the IDB block is unable to communicate with the Bailey interface.

**WAITING ON IDB RESTART OF BAILEY INTERFACE:** The block is waiting for its associated IDB block to complete startup of the Bailey interface and make itself available to the other DVCBailey blocks.

**BAILEY INTERFACE REQUIRES A PHYSICAL RESET:** The Bailey interface has locked itself from being able to communication. It must be physically reset to recover communication. This error can only occur for INICI03 and INICI12 Bailey interfaces.

**ESTABLISHING POINT:** The block has requested its associated IDB to establish the point in the Bailey interface.

**WAITING FOR DATA FROM BAILEY:** The point has been established in the Bailey interface and is waiting to receive its initial data from the Bailey system.

**ONLINE:** The block has received data from the Bailey system.

**ILLEGAL BLOCK NUMBER:** The DVCBailey AOL or DOL block has been configured with a block number that has exceeded the IDB MAX\_OUTPUTS attribute setting.

**BLOCK ADDRESS ALREADY USED BY ANOTHER BLOCK:** Another DVCBailey block has been configured for the Bailey address set within this block.

**EXCEEDED BAILEY INTERFACE INDEX CAPACITY:** More DVCBailey blocks have been configured than can be handled by the Bailey interface.

**EXCEEDED DVCBailey POINT LICENSE:** More DVCBailey blocks have been configured than are allowed by the current registered license.

## 9.2 No Communication

If the Bailey interface LEDs do not sequence, this means the driver is not able to successfully communicate with the N90 interface.

- 1.) Verify the DVCBailey IDB block PORT\_A and PORT\_B attributes are associated with the same COM port to which the Bailey interface has been cabled.
- 2.) Verify the PORT\_A and PORT\_B communication settings match those setup within the Bailey interface.

- 3.) Verify the Bailey interface device termination unit/module serial port jumpers are setup correctly.
- 4.) Verify RS232 cable is connected to the correct Bailey termination unit/module connector. Generally this is labeled as the terminal port for the primary channel and printer port for secondary channel when the IDB block SCHEME attribute is set to dual channel single interface.
- 5.) Verify the RS232 cable is connected to the correct PC COM port.
- 6.) Verify the IDB has been downloaded to the Application Station.
- 7.) Check the IDB MESSAGE attribute in Control Studio online looking for a message that might indicate the problem.
- 8.) Verify the Bailey interface red/green led is not red. Try resetting the Bailey interface.

### **9.3 Appears To Be Communicating But No Data is Being Received**

If the Bailey interface LEDs sequence at a very steady and periodic rate this means the driver is connected to the Bailey interface but the DVCBailey IDB block communication parameters might not be set correctly. These settings are determined by the IDB block PORT\_A and PORT\_B attributes. Note that these attributes can be changed while the driver is on-line but not able to communicate with the Bailey interface. If communicating changes will not be permitted.

- 1.) Verify the baud rate settings match between the Bailey interface and COM port the DVCBailey IDB block driver is using.
- 2.) Verify the DVCBailey modules containing the blocks are downloaded to the Application Station.
- 3.) Verify the parity (usually none) and stop bits (usually 1) match between the Bailey interface and COM port the server is using.
- 4.) Verify the IDB Message attribute in Control Studio online does not indicate the Bailey interface requires a physical reset.

If the Bailey interface LEDs are not sequencing but the IDB communication status outputs indicate good communication status (PRI\_STATUS or SEC\_STATUS is zero).

- 1.) Verify the COM port associated with the DEVICE is not a modem port.

### **9.4 Not All Blocks Are Receiving Data**

- 1.) Verify the Bailey block addressed by the DVCBailey block exists and their types match.
- 2.) Verify "BLOCK ADDRESS ALREADY USED BY ANOTHER BLOCK" is not posted in the MESSAGE attribute in any of the DVCBailey blocks not receiving data. If such a message is

found, you have two or more DVCBailey blocks pointing to the same address within Bailey. Eliminate the duplication.

- 3.) Verify "EXCEEDED DVCBailey POINT LICENSE" is not posted in the MESSAGE attribute in any of the DVCBailey blocks not receiving data. If such a message is found, you have exceeded the registered block capacity. Purchase a larger block license.
- 5.) Make sure none of the Bailey controllers have addresses less than two.
- 6.) Indices or memory capacity of the Bailey interface has been exceeded. It is unlikely that you will encounter this error unless your application has a very large number of DVCBailey blocks and the Bailey interface device is a NCIU01 or INCIC01. Enable the IDB DEBUG\_LOG error reporting option and search the DVC log file messages for such an error.
- 7.) Memory capacity of the Bailey PCU node has been exceeded, check its status using a DVCBailey STAT block. Note that this error is extremely rare.
- 8.) Be sure the blocks that are not receiving data are downloaded to the Application Station.

## **9.5 Cannot Export or Control Data Within Bailey**

- 1.) Verify that the Bailey interface device supports this type of activity.
- 2.) Verify that the faceplate attribute entry has been enabled for writes.
- 3.) Verify the allowable ranges for the attribute being written match those configured within Bailey. For example you cannot write an OUT value that exceeds the OUT\_SCALE limits.
- 4.) Verify the Bailey interface does not have monitor mode enabled (very unlikely). Monitor mode is set using an ASCII terminal attached to the termination unit/module printer port and that port selected for the utility mode (consult Bailey interface manual). Normally, monitor mode is disabled when a Bailey interface is received from the Bailey factory. It is unlikely that monitor mode is enabled unless someone had attached an ASCII terminal at one time and was "experimenting" with the available options within the Bailey interface utility menus.
- 5.) Verify the IDB is not "locked" by looking at the IDB LOCK parameter.

## **9.6 BLK Block Takes A Long Time To Complete A Read / Tune**

The interface bandwidth dedicated to background reading of PID and DANG tuning constants is not set correctly. The IDB Block SPEC\_UPDATE attribute needs to be changed to a value that is at least 5000 times the total number of STN and DANG blocks.

## **9.7 DVCBailey Will Not Time Sync Bailey**

- 1.) Verify that the Bailey interface device is not a NSPM01 or NCIU01. These devices do not allow a computer to send time synchronization commands to Bailey. If the Bailey interface device is one of these types, disable time synchronization.
- 2.) Verify that time synchronization has been enabled. This can be found under the OPTIONS of the IDB.
- 3.) For Plantloop based Bailey systems, verify that the IDB block node map is configured correctly for every node in the system regardless of whether or not data is being exchanged with all nodes.
- 4.) Note that DVCBailey gives precedence to MCS and OIS nodes as becoming the time synchronization master. Also note, in the DeltaV system, the Application Station will always receive it's time from the ProPlus Station.

## 9.8 Both App Stations Become Active Upon Failure Of DeltaV ACN

The DeltaV ACN is utilized to maintain a heart beat signal between the Active and Standby App Station. When a fault occurs on both channels of the ACN, the heart beat signal fails and the Standby App goes to the Active State. The current Active remains active since it can no longer sense the presence of the former Standby. For systems that utilize Bailey interfaces at the same address, this type of failure will result in the CIU associated with the formerly Standby App Station to halt itself. The reason it halts is when the former Standby App Station becomes Active, it commands its CIU to the online state. That CIU senses that another CIU is already present on the Bailey communication highway at its address so it halts itself to maintain the integrity of the Bailey communication system. To recover from this type of fault, correct the ACN failure, reboot the former Standby App Station and reset the halted Bailey CIU.

## 9.9 Redundant App Stations SCSI CIU Addresses Don't Match

The SCSI address of ABB Bailey INICT03A/13A modules must be identical between redundant application stations. The address assigned to SCSI devices has three components which are port number, card number and device on that card. For example the address S2107 indicates port two, card one and device seven. Windows automatically assigns the port number based on when new hardware is added. Therefore, sometimes this port number assignment can end up being different between two PCs based on the order in which new hardware is discovered. If this occurs between redundant application stations use the following procedure to re-align the port address.

Using Regedit go to the key named HKEY\_LOCAL\_MACHINE\HARDWARE\DEVICEMAP\Scsi in the registry. Find the branch that contains the Bailey interface. Click the Scsi Port # at the top of the branch and look for the Driver name.

Go to HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\*drivername* for both the device ahead of you and for the SCSI device you want to be first and look at the TAG value. You need to change the tag value so that the device you want listed first (as S1000) is a lower value than the device that is forcing it to S2000:.

Detailed explanation of what this procedure accomplishes follows as gleaned from the Microsoft web site:

Tag REG\_DWORD Specifies a load order within a given group. The value of Tag specifies a number that is unique within the group of which the service is a member. The related GroupName entry under the Control\GroupOrderList subkey specifies a list of tags, in load order. For example, the following services that are members of the Primary Disk group could have these values: Tag=4 for the Abiosdsk subkey, Tag=2 for Atdisk, Tag=1 for Cpqarray, and Tag=3 for Floppy. The value for Primary Disk under the GroupOrderList subkey will use these Tag values to specify the defined order for loading these services. As another example, each SCSI miniport service has a unique Tag value that is used as an identifier in the SCSI miniport value under the GroupOrderList subkey to define which SCSI adapter to load first.