Preliminary

DeviceNet DN2 Extensions

Animatics Part Number Suffix DN2
DeviceNet IC Firmware Version 1.13

Revision 5 03/13/2006 JH

Contents

Application Modifications Necessary For Version DN Users Migrating to DN2

If your DN application has not modified the user program that came with your motor, or does not use variable y in the user program, and you do not use the Forward and Reverse Limits Active DeviceNet Position Controller Supervisor attributes 50 and 51, this section may be skipped.

Motor User Variable y Is Modified by Attribute Access

Motor user variable y is modified by Position Controller Supervisor Class 36 (decimal) attributes 27 and 28, and Position Controller Class 37(decimal) attributes 54, 55, 103, and 104. **Do not use variable y in your user program**.

Change In Meaning of Attributes 50, 51 Limits Active

In prior versions attributes 50 and 51, Forward and Reverse Limits Active, were value 1 if there had been an occurrence of limit becoming active, or of an attempted move into an active limit. Beginning with DeviceNet IC firmware version 1.08 attributes 50 and 51 reflect the real-time state of the limit switch, value 1 if active, and attributes 56 and 57 reflect the latched limit occurrence. **Programs written for prior versions using attributes 50 and 51 that may have been relying on the latched occurrence characteristic of attributes 50 and 51, should replace 50 and 51 with attributes 56 and 57.**

Summary of DN2 Changes

Capabilities have been added to the Animatics DeviceNet SmartMotor™ and DeviceNet ServoStep™ in the form of DeviceNet Position Controller profile and Animatics manufacturer-specific attributes added or modified, and enhanced motor serial port communication, applying to versions DeviceNet firmware versions 1.13 and beyond.

Some users of DeviceNet firmware versions 1.05 and lower may need to modify their applications—see section "**Application Modifications Necessary For Version DN Users Migrating to DN2."**

The new or modified attributes are:

POSITION CONTROLLER SUPERVISOR CLASS 36(decimal) Instance 1

27	Follow Divisor	now signed, GET is value in motor
28	Follow Multiplier	now signed, GET is value in motor

POSITION CONTROLLER CLASS 37(decimal) Instance 1

13	Absolute Position	GET documented
	(24 Reference Direction)	(new DO NOT USE ON SERVOSTEP™)
	(49 Hard Limit Action)	(Use 53 to specify hard and soft limit action)
50	Forward Limit Active	Now is current level, was latched occurrence
51	Reverse Limit Active	Now is current level, was latched occurrence
53	Soft Limit Action	new, also is Hard Limit Action
54	Positive Soft Limit Position	new
55	Negative Soft Limit Position	new
56	Hard Forward Limit Occurrence	new, latched occurrence
57	Hard Reverse Limit Occurrence	new, latched occurrence
101	Reset Motor Error Status	new
102	Reset Motor	new
103	Overheat Setpoint	new
104	Real-time Temperature	new
105	Motor User Variable u	new
106	Motor User Variable v	new
107	Motor User Variable w	new
108	Motor User Variable x	new
109	Motor User Program GOSUBnnn	new
110	Loss of DeviceNet Network Action	new
111	Motor Status Word	new
112	Motor ServoStep™ Calibration-fault Status	new

Motor user variable y is modified by Position Controller Supervisor Class 36 (decimal) attributes 27 and 28, and Position Controller Class 37(decimal) attributes 54, 55, 103, and 104. **Do not use variable y in your user program**.

Additions to Non-volatile EEPROM Configuration Data

Loss of Network Action

Attribute 110, Loss of Network Action, will be set at power up by the gateway to the value contained in the 32-bit long value at EEPROM location 32028.

To set the desired Loss of Network Action using the SMI terminal, send commands to the motor serial port:

```
a=<Loss of Network Action code>
EPTR=32028
VST(a,1)
```

To verify the desired Loss of Network Action code has been stored:

```
b=-1
Rb                  ‘ verify b = -1
EPTR=32028
VLD(b,1)
Rb                  ‘ verify b is Loss of Network Action code
```

Polling Delay: User Program Execution Speed

If you are not running a speed-critical user program in the motor, you may skip this topic.

The motor processes one command at a time. The user program supplied with the motor terminates after initializing the communications port, so the motor is devoted to servicing DeviceNet IC polling and motion commands.

For DeviceNet IC firmware versions 1.12 and higher, a polling delay value at EEPROM location 32012, loaded at power-up, has a valid range from 0 to 65535, in 8 millisecond units, with default 0.

If you write a user program that continues to run, and user program execution speed is important, you can marginally increase the user program execution speed by increasing the polling delay. The DeviceNet IC will apply the polling delay between each poll of the motor.

However, when the DeviceNet master requests a value from the DeviceNet IC, the DeviceNet IC will ignore the polling delay and immediately request the value from the

motor. If the DeviceNet master polls less frequently, the user program will execute somewhat faster. If the motor is not connected to the DeviceNet master, setting the polling delay has no effect.

DeviceNet IC Serial Number

The serial number of the DeviceNet IC is stored to EEPROM location 32016 on power up, if it has not already been stored.

DeviceNet Major and Minor Version Numbers

DeviceNet IC firmware versions 1.08 and later store the major and minor version numbers to the EEPROM at locations 32020 and 32024 respectively, on power up, if they have not already been stored. The major number precedes the decimal point, the minor number follows.

Table of EEPROM Addresses

32000	counts per shaft revolution for motor: 2000, 4000
32004	DeviceNet baud rate: 125, 250, 500, factory default 125
32008	DeviceNet MacId: 0 to 63, factory default 63
32012	Polling delay: 0 to 65535, factory default 0
32016	DeviceNet IC serial number
32020	DeviceNet IC major version number (=1 for version 1.13)
32024	DeviceNet IC minor version number (=13 for version 1.13)
32028	Network-lost action: 0 to 9, factory default 0 = servo off

Enhanced Motor Serial Port Communication

The new DN2 version DeviceNet motor allows enhanced use of simultaneous serial port and DeviceNet communication. However, as in the DN version, the motor processor executes one command at a time. Once the motor has begun interpreting a command until it has finished executing the command it cannot begin processing the next command. Thus user program and serial port commands can introduce latency to DeviceNet commands to the motor and the motor's DeviceNet responses to the master.

In particular, consideration should be given to the effect of long-duration commands, for example a PRINT command (especially a long one), the user program LOAD or UPLOAD commands, RCKS, or a character being sent to the serial port that begins interpretation of a command but does not complete the command.

DeviceNet IC Power Up/Reset Protocol—User Variable zzz

If you have written a user program that accesses the motor eeprom or user variables a to n you must not conflict with the DeviceNet IC accesses. If you do not access the motor eeprom using EPTR, VLD(), VST(), or user variables a to n in your user program (recommended), you may skip this topic.

The DeviceNet IC accesses the motor eeprom and modifies user variables a to n during initialization. This may occur both during initial motor power up and after a DeviceNet IC reset that is caused by a fluctuation of DeviceNet cable power. Disconnecting or connecting the DeviceNet cable may result in intermittent pin contact that may cause the DeviceNet IC to reset.

When the DeviceNet IC has completed access to the motor eeprom and a to n, the DeviceNet IC will set motor variable zzz to 1. Therefore, your user program should wait until zzz==1 before setting zzz to 2.

The DeviceNet IC will not again access the motor eeprom or variables a to n unless it resets because of a problem with the DeviceNet cable or DeviceNet power. To prevent the DeviceNet IC from accessing the eeprom and a to n in that event, set zzz=2 in the user program. If zzz==2 (not 0 or 1), instead of accessing the eeprom and a to n, the DeviceNet IC will set zzz=3 to inform the user program it is waiting for permission to complete initialization.

If you have set zzz=2 and wish the DeviceNet node to be able to complete reset and come on line again without power cycling the motor in the event of an interruption in DeviceNet cable power, your user program must periodically check for zzz==3 and, if found, set zzz=0 to allow initialization to proceed.

Note there is a small possibility of a DeviceNet IC reset occurring at precisely the same time as the user program setting zzz=2. If the DeviceNet IC has queried zzz just prior to the user program setting zzz=2, the DeviceNet IC may be already proceeding with eeprom access. Therefore, it would be most prudent to allow for enough time for the IC eeprom access, had it started, to have completed before accessing the eeprom in the user program, and set zzz=2 again in case a reset had set zzz back to 1. A half-second, or WAIT=4035, is adequate.

A state diagram for zzz is included below.

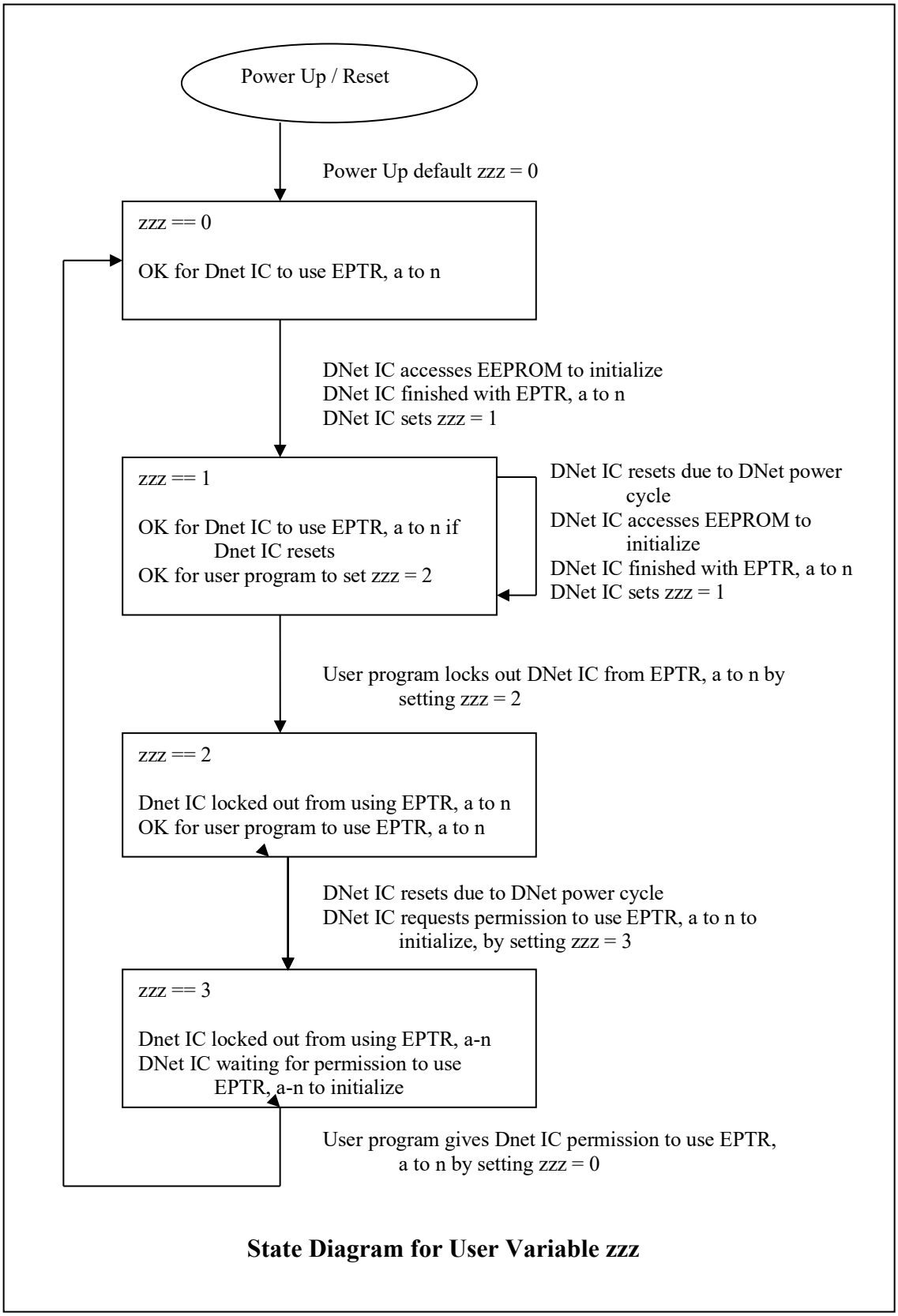
Example, SmartMotor™ user program accesses EEPROM:

```
OCHN(RS4,1,N,19200,1,8,C)
…

WHILE zzz!=1 LOOP        ' wait for DeviceNet IC to set zzz==1
WHILE zzz!=2             ' in case reset set zzz back to 0 or 1
      zzz=2              ' lock out DeviceNet IC if reset occurs
      WAIT=2035          ' just in case a reset just occurred
```

```
LOOP                    ' in case reset set zzz back to 0 or 1
EPTR=0                  ' user program uses EPTR
VLD(p,3)                ' user program uses EPTR
zzz=0                   ' enable DeviceNet IC initialization
                        '     in case reset occurs

…
END
```



State Diagram for User Variable *zzz*

Attribute Table

Position Controller, Class 37 (decimal) Instance 1

Attribute ID	Access Rule	Name	DeviceNet Data Type	Description of Attribute	Get Action	Set Action
1	Get	Number of Attributes	USINT	Returns the total number of attributes supported by this object in this device	fixed	N/A
2	Get	Attribute List	Array of USINT	Returns an array with a list of the attributes supported by this object in this device	fixed	N/A
3	Set	Mode	USINT	Operating Mode	RMODE, P/R=0, V=1, T=2	**0:** MP **1:** MV **2:** MT
6	Set	Target Position	DINT	Position value to set		**absolute:** P=nnn **incremental:** D=nnn
7	Set	Target Velocity	DINT	Velocity value to set	RV	V=nnn
8	Set	Acceleration	DINT	Acceleration rate	RA	A=nnn
10	Set	Incremental Position Flag	BOOL	0=absolute, 1=incremental		
11	Set	Load Data/ Start Profile/ Profile in Progress	BOOL	On set, loads data and starts the current profile. On get, reports Profile in Progress	RW status bit 0	**1:** G **0:** N/A
13	Get Set	Actual Position	DINT	Actual absolute position. Set to redefine actual position.	RP (or RPW)	O=nnn
14	Get	Actual Velocity	DINT	Reports actual velocity	RV (only valid in Torque Mode)	N/A
15	Get	Commanded Position	DINT	The instantaneous calculated position	RP (or RPW) + RPE	N/A
16	Get	Commanded Velocity	DINT	The instantaneous calculated velocity	RV	N/A
17	Set	Enable	BOOL	0=disable 1=enable		**1:** G or MP D=0 G, allow G or MT **0:** OFF
20	Set	Smooth Stop	BOOL	Smooth Stop motor	0	**1:** X
21	Set	Hard Stop	BOOL	Hard Stop motor	0	**1:** S
23	Set	Direction	BOOL	Instantaneous Direction 0=reverse, 1=forward	Position Mode (direction of move) Velocity Mode (sign of Velocity). Torque Mode, sign of Torque	V=+/-nnn G T=+/-nnn

Attribute ID	Access Rule	Name	DeviceNet Data Type	Description of Attribute	Get Action	Set Action
DO NOT USE FOR SERVO STEP™ 24	Get Set	Reference Direction	BOOL	Shaft Direction forward facing shaft 0=CW, 1=CCW	RF, bit1 **0:** F bit1==0 (F=0) **1:** F bit1==2 (F=2)	**0:** clear F bit1 (F=0) **1:** set F bit1 (F=2)
25	Set	Torque	DINT	Output Torque	RT	T=nnn or T=-nnn
29	Get	Wrap Around	BOOL	Position Wrap Around Indicator Flag	RW status bit 4	N/A
30	Set	Kp	INT	Proportional Gain	RKP	KP=nnn F
31	Set	Ki	INT	Integral Gain	RKI	KI=nnn F
32	Set	Kd	INT	Derivative Gain	RKD	KD=nnn F
33	Set	MaxKi	INT	Integration Limit	RKL	KL=nnn F
35	Set	Velocity Feed Forward	INT	Velocity feed forward gain value	RKV	KV=nnn F
37	Get	Sample Rate	INT	Update sample rate in micro-seconds	RSP	N/A
40	Get	Feedback Resolution	DINT	Number of actual position feedback counts per revolution	EPTR=32000 VLD(a,1) Ra	N/A
41	Get	Motor Resolution	DINT	Number of motor steps per revolution	EPTR=32000 VLD(a,1) Ra	N/A
45	Set	Max Dynamic Following Error	DINT	Maximum allowable following error when the motor is in motion	RE	E=nnn
47	Get	Following Error Fault	BOOL	Following error occurrence flag	RW status bit 5	
48	Get	Actual Following Error	DINT	Actual Following error	RPE	N/A
49 not imple-mented use 53		Hard Limit Action	USINT	Hard Limit Action code (applies to Soft Limit Action also) 0=Servo off 2=Smooth Stop		
50	Get	Forward Limit	BOOL	Forward Limit stop active	RW status bit 9	N/A
51	Get	Reverse Limit	BOOL	Reverse Limit stop active	RW status bit 10	N/A
53	Get Set	Soft Limit Action	USINT	Soft Limit Action Code (applies to Hard Limit Action Also) 0=Servo off 2=Smooth Stop	RF, bit 0 **0:** F bit0==0 (F=0) **2:** F bit0==1 (F=1)	**0:** clear F, bit0 (F=0) **2:** set F, bit0 (F=1)

Attribute ID	Access Rule	Name	DeviceNet Data Type	Description of Attribute	Get Action	Set Action
54	Get Set	Positive Soft Limit Position	DINT	Soft limit positive boundary in counts, enable BOTH soft limits	y=SLP	SLD SLP=nnn SLE
55	Get Set	Negative Soft Limit Position	DINT	Soft limit negative boundary in counts, enable BOTH soft limits	y=SLN	SLD SLN=nnn SLE
56	Get	Positive Limit Triggered	BOOL	Hard or Soft limit forward limit occurrence flag	RW, bit 1	
57	Get	Negative Limit Triggered	BOOL	Hard or Soft limit forward limit occurrence flag	RW, bit 2	
58	Get	Load Data Complete	BOOL	valid data for a valid I/O command message type has been loaded into the position controller		N/A
100	Set	Current Limit	DINT	Current limit of motor 0 to 1023	RAMPS	AMPS=nnn
101	Set	Reset Motor Faults	BOOL	0 = no action 1 = reset faults		ZS
102	Set	Reset Motor	BOOL	0=no action 1=reset motor		Z
103	Get Set	Overheat Setpoint	USINT	Overheat setpoint 0-70 or 0-85 degrees C	y=TH Ry	TH=nn
104	Get	Temperature	INT	Real time temperature	y=TEMP Ry	
105	Get Set	Variable u	DINT	Motor user variable u	Ru	u=nnn
106	Get Set	Variable v	DINT	Motor user variable v	Rv	v=nnn
107	Get Set	Variable w	DINT	Motor user variable w	Rw	w=nnn
108	Get Set	Variable x	DINT	Motor user variable x	Rx	x=nnn
109	Set	GOSUBnnn	UINT	Motor user program command GOSUBnnn		GOSUBnnn
110	Get	Loss of Network Action	USINT	Action if DeviceNet network heartbeat lost 0=Servo off 1=Smooth stop 2=Hard stop 3=Motor reset 4=no action 5 to 9=GOSUBn		
111	Get	Status Word	UINT	Motor status word	RW	
112	Get	Calibration Fault	BOOL	ServoStep™ calibration fault, 1=fault	RBn	