IS-70048 User Manual

Revision A

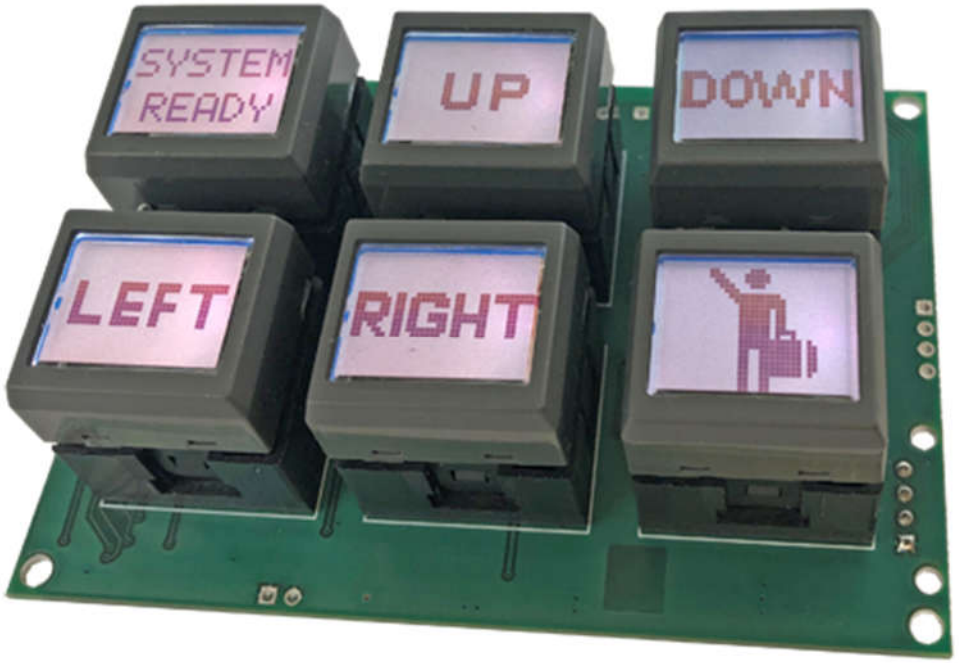


Table of Contents

1. LCD 36x24 SmartDisplay Switch

The LCD 36x24 SmartDisplay switch is a graphic 36x24 LCD display mounted in the key cap of a momentary pushbutton. It has an RGB backlight with discrete control.

Please contact engineering@nkkswitches.com with your requirements for custom solutions.

NKK can supply subsystems with any configuration and number of LCD 36x24 switches with USB and Ethernet communication. SmartDisplay allows designers to dynamically change switch legends and images based on desired application functions. The system is ready to interface with a customer's application through CAN and USB. It can receive commands, send information, and update the SmartDisplay images.

SmartDisplay is ideal for use in applications with multiple, complex functions which would ordinarily require many dedicated switches and complex training. The dynamic nature of the system allows for instantaneous transitions from generalized lists of categories down to function specific actions. This reduces the need for complicated controls and shortens the time for training by only displaying relevant options and commands.

To help with development, NKK Switches provides free software, Engineering Kits Communicator, to save and erase images on the controller. Also, NKK Switches provides all the documentation necessary to get up and running quickly on our website: https://www.nkkswitches.com/SmartDisplay-resources/

2. General Features

The system is a CAN-controlled 8 programmable display-on-pushbutton system in a compact form factor. It comes with the following features:

Features:

- 8 36x24 LCD SmartDisplays with momentary pushbutton functionality.
- USB or CAN controlled.
- Power Specs: 120-240VAC, Max 24 Watts. (with included power cord).
- The unit comes with a 6-foot USB 2.0 A to Mini-B cable (IS-USB1).
- On-board memory for 60,000 images
- 8 levels of brightness.
- Real-time control by host.
    - Save images to memory.
    - Show any saved image on any switch.
    - Reports switch activity to host.
    - Ability to send images directly to switches without saving to memory.

  - Write text on switches in two different font sizes
- Controller board firmware can be customized based on customer requirements.
- Firmware field upgradable via USB.
- Windows based software is available for communication.
  - Accepts bitmap files, extracts the images and download them to the controller.
  - Allows typing of commands and downloading to the controller.
  - Messages to and from the controller are displayed in different colors.
- The communication protocol can be modified to meet customer's requirements.
- Please contact the factory about custom builds and firmware modifications.

3. Electrical Specifications

Power Specs: Max 3 Watts.
USB +5 V
CAN +30VDC max

4. Communication

The systems can communicate over USB and CAN. If using CAN, the system must be configured over USB first. All commands and responses are detailed in the associated Command List. A non-inclusive list of commands is as follows:

- Acknowledge.
- Erase flash memory.
- Get/Set ethernet settings.
- Reset system.
- Query version.
- Save image to flash memory.
- Send image directly to switch.
- Set image from flash memory on specific switch.

The system shows up as a generic USB COM port. This allows quick testing, loading of images, and integration with customer software. For testing, the NKK Engineering Kits Communicator or a standard terminal program such as Putty can be used.

5. Images

Images can be created in any graphics software such as Paint, Photoshop, etc, or even user-created software. All images can be saved onto the system by using the free Engineering Kits Communicator, located on the NKK Website:
https://www.nkkswitches.com/

(Images can also be loaded onto the system with user-created software as long as the rules for the images and communications are followed.)

To use this software, images must be saved in the proper format:

LCD 36x24	Monochrome bitmap (.bmp) 36x24 pixels

Please note that the **flash memory must be erased before new images are loaded**, or images will not display properly. Erasing can take up to 2 minutes depending on the size of the flash memory.

The Engineering Kits Communicator will auto-convert the monochrome .bmp file to the switch format and send the data. If writing custom software, be aware bitmap format specifies the bottom-left corner as the “top”. Therefore, to send images properly to the switches the data needs to be sent last row first, followed by next to last, etc.

The system expects LCD image pixels to be monochromatic. Pixels are on or off. Only the backlighting has color. Each bit corresponds to a pixel in the image.

When saving images to flash, the data needs to be converted to ASCII hex for 240 bytes of data. If streaming live images, data should be sent as-is for 120 bytes of data.

Monochrome bitmap (.bmp) 36x24 pixels	1 byte per 8 pixels	120 bytes per image

*the last 4 bits of every byte are dummy bits and not used

6. Operational Overview

Upon power-up the system configures and turns on the switches. Images 1-6 from memory are loaded on switches 1-6 respectively. The system then waits for a command from the host. The only action the system performs automatically is reporting switch presses. All other actions must be commanded from the host.

Images are loaded using the engineering kits communicator over USB.

The flash memory stores both CAN settings and images. They are stored in separate sections of memory, so erasing the images will not remove the ethernet settings and vice versa. Each image is assigned a sequential address when saved into flash memory; A list of which image is stored at which address must be kept on the host software to know which image to display where.

When a switch is pressed, the system reports that back to the host software. The system only reports switch state changes (a press or release). More than one switch can be in the pressed position at the same time. Bits are set when the switch is pressed and cleared when the switch is released. Examples of a switch response is:

CANOpen ID	Message	Notes
0x195	00 00 00 00 00 00 00 00	No switch pressed
0x195	01 00 00 00 00 00 00 00	Switch 0 pressed
0x195	04 00 00 00 00 00 00 00	Switch 2 pressed
0x195	80 00 00 00 00 00 00 00	Switch 7 pressed

PDOs **MUST** be enabled for switch scan to report, or the “active on startup” flag must be set >0.

7. Saving Images Using Engineering Kits Communicator

The Engineering Kits Communicator loads the images in alphanumeric order according to the image files names. It auto-assigns a sequential address to each image. Be sure to keep this in mind when naming images so that video images or animations are listed in the desired order. Avoid using symbols in the names as some symbols interfere with alphanumeric ordering. All images to be loaded should be saved in a single folder. The default starting address is 0001. This can be changed if needed.

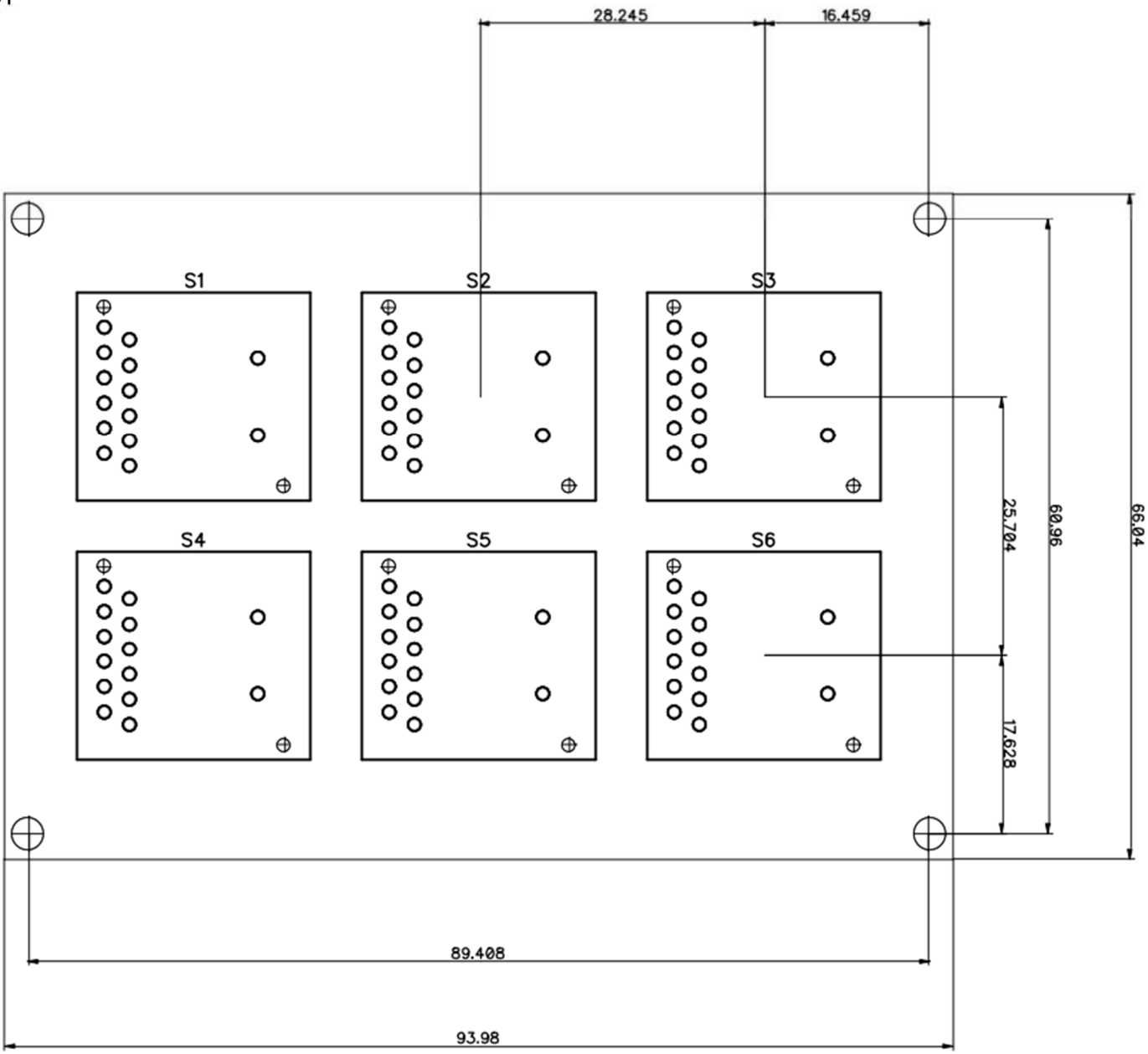
To save images to the system:

1. Open the Engineering Kits Communicator.
2. From the drop-down menu at the top, select the COM port of the system (usually the last one).
3. Click the 'Open Port' button.
4. Press the call button and verify the system responds with '61' in blue text in the left text box.
5. Select the image type from the drop-down in the 'Loading Images' section.
6. Click the 'Import Images' button.
7. Navigate to the directory with all the images and select one and click 'Open'.
8. Note that the images are loaded alphanumerically and automatically assigned addresses.
    a. If some/all images do not show up in the image list after selecting the directory, it is because the image is not in the proper resolution or file type (.bmp). Double-check the image size is correct *before* downloading. If an image was skipped the images will load one address off and will have to be erased before reloading them.
9. If images were previously saved, click the 'Erase Flash' button.
    a. Note that this operation can take up to **2 minutes**.
10. Click the 'All selected images' button at the bottom.
11. Wait for the 'Success' message. If the process fails, click the 'All selected images' button again.

If writing custom software to save images, all data after the command must be sent in ASCII hex (See Sections Images and ASCII Hex).

8. Board Dimensions

Typical dimensions.



9. ASCII Hex

All USB data is sent as ASCII hex as a safety measure to avoid being interpreted as a command. ASCII hex is a normal data byte split into two halves and converted to their ASCII equivalent (see www.asciitable.com). Therefore, all data received will be in the 0x30-0x46 number range, and anything received outside that is invalid. For example:

Data to be sent is 0x3D.
Each nibble is assigned its own byte: 0x3 and 0xD.
Each of those bytes is converted to ASCII equivalent: 0x33 0x44 (0x33 is the ASCII number '3'. 0x44 is the ASCII letter 'D').

Receiving is the same process reversed:

Data received is 0x41 0x37.
Each byte converted from the ASCII equivalent is 0xA 0x7 (0x41 is the ASCII letter 'A', 0x37 is the ASCII number '7').
Combine the two bytes to get the data byte 0xA7.

10. Key Terms & Definitions

Host	Any computer, terminal, or other device that can communicate over the USB line.
Byte	An eight-bit hex value ranging from 00H to FFH (Decimal 0 to 255). The bit format of a byte is: (B7 B6 B5 B4 B3 B2 B1 B0) where B7 is most significant and bit B0 is least significant bit.
Nibble/Hex Digit	A four-bit value ranging from 0H to FH. A byte consists of two nibbles.
ASCII	A byte value representing a symbol.
Communication Format	There are two formats to transmit a byte: 1. **Hex format** - A hex byte is transmitted without any change to it. [**xxH**] will be used to denote this. All commands and some data are sent by using this format. 2. **ASCII HEX format** - Each nibble of the byte is converted to ASCII code and sent as a byte. [**xxAH**] will be used to denote this. For example, the hex byte 5AH is transmitted in two bytes, **35H** and **41H**. The ASCII value for **5** is **35H** and the ASCII value for **A** is **41H**. All addresses and most data are sent using this format.
Address	A two-byte value ranging from 0001H to 03E8H representing the 1000 memory locations for images on the flash memory.

11. Commands to the Controller

See the Command List on the next page.

USB Command List

ASCII Hex

An ASCII Hex byte is a normal hex byte split in two halves and converted to their ASCII equivalent (www.asciitable.com). This is a safety measure so that all data sent isn't accidentally interpreted as a command. Most data sent after commands and sub-commands are in ASCII Hex. The only exception is when sending live images (images not saved in memory and sent directly to the switches)

Example: To set switch 4 to image 679 (0x02A7):
Switch 4 is 0x03 (0-based), change to 2 bytes: 0x30 (ASCII '0') and 0x33 (ASCII '3')
Image high byte 0x02 to 2 bytes: 0x30 (ASCII '0') and 0x32 (ASCII '2')
Image low byte 0xA7 to 2 bytes: 0x41 (ASCII 'A') and 0x37 (ASCII '7')
Using the table below, this would result in the command:
0x2D 0x56 0x30 0x33 0x30 0x32 0x41 0x37

Conversion **TO** ASCII Hex:
x = ((data & 0xF0) >> 4)
y = ((data & 0x0F) >> 0)
If (0x0 <= x <= 0x9) x+= 0x30
If (0xA <= x <= 0xF) x+= 0x37
If (0x0 <= y <= 0x9) x+= 0x30
If (0xA <= y <= 0xF) x+= 0x37

Conversion **FROM** ASCII Hex:
x = ASCII hex byte 1
y = ASCII hex byte 2 (conversion not shown)
z = converted byte
If (0x30 <= x <= 0x39) x -= 0x30
If (0x41 <= x <= 0x46) x-= 0x37
z = (x << 4) + y

Command	Sub-Command	Description	Command Format	Sending example	Notes
0x01		Queries the controller if it is up and running (Controller responds with 0x61)	**0x01** (1 hex byte)	0x01	
0x21	0x55 0xAA 0x56 0x56	Erases all images on the flash memory	**0x21 0x55 0xAA 0x56 0x56** (5 hex bytes)	0x21 0x55 0xAA 0x56 0x56	Responds with 0x65 until the erase is complete, then sends 0x79
	0x55 0xAA 0x56 0x56 AA	Erases all images and settings on the flash memory	**0x21 0x55 0xAA 0x56 0x56 0xAA** (5 hex bytes)	0x21 0x55 0xAA 0x56 0x56 0xAA	Responds with 0x65 until the erase is complete, then sends 0x79
	0x55 0xAA 0x56 0x56 DD	Erases all images on the flash memory	**0x21 0x55 0xAA 0x56 0x56 0xDD** (5 hex bytes)	0x21 0x55 0xAA 0x56 0x56 0xDD	Responds with 0x65 until the erase is complete, then sends 0x79
	0x55 0xAA 0x56 0x56 EE	Erases all settings on the flash memory	**0x21 0x55 0xAA 0x56 0x56 oxEE** (5 hex bytes)	0x21 0x55 0xAA 0x56 0x56 0xEE	Responds with 0x65 until the erase is complete, then sends 0x79
0x24		Resets the controller	**0x24** (1 hex byte)	0x24	
0x26		Retrieves the firmware version number	**0x26** (1 hex byte)	0x26	Returns the current version number in ASCII hex. It will be the system number in ASCII hex followed by the letter 'v' followed by 4 ASCII hex bytes signifying the major version, followed by '.', followed by 4 ASCII hex bytes signifying the minor version number Example: 0x49 0x53 0x2D 0x37 30 30 34 32 2E 0x30 0x30 0x30 x31 0x2E 0x30 x030 0x30 0x30 in ASCII is IS-70042v0001.0000
0x28	0x56	Saves an image to the flash memory	**0x28 0x56** (2 hex bytes) **0xAA 0xBB 0xCC 0xDD** (4 ASCII hex bytes for the image address) After the controller responds with 0x61, send image data in ASCII hex. The controller responds with 0x79 when the save completes.	0x28 0x56 0x30 0x30 0x30 0x31 (Send an image, save at address 0x0001) (Followed by bytes of image data in ASCII hex after the controller responds with 0x61)	1000 images maximum (0x3E8 is the maximum address). Going beyond the maximum address will result in image corruption.
0x2B	0x48 0x4F 0x53 0x54 0x43 0x4F 0x4D 0x4D 0x41 0x4E 0x44	Commands the firmware to return to the bootloader and remain there	0x2B 0x48 0x4F 0x53 0x54 0x43 0x4F 0x4D 0x4D 0x41 0x4E 0x44 (12 hex bytes)	0x2B 0x48 0x4F 0x53 0x54 0x43 0x4F 0x4D 0x4D 0x41 0x4E 0x44 (Returns the system to the bootloader for firmware update)	The system will respond with ready but not a success message
0x2C		Saves CAN settings to flash memory	**0x2C** (1 hex byte) **0xAA 0xBB** (2 ASCII hex bytes of baud rate) **0xCC 0xDD** (2 ASCII hex bytes of CANOpen ID) **0xEE 0xFF** (2 ASCII hex bytes of boot-up service enable) **0xGG 0xHH** (2 ASCII hex bytes of active on startup enable)	0x2C 0x31 0x31 0x30 0x31 0x30 0x30 (Set baud rate to 500k, boot-up service enabled, active on startup disabled	Baud rate defaults to 250k. Options are: CAN_SPEED_125k = 0x0F, CAN_SPEED_250k = 0x10 CAN_SPEED_500k = 0x11 CAN_SPEED_1M = 0x12 Boot-up service sends a 0x700 + ID message when powered on. Active on startup bypasses the need for the NMT startup command and PDOs start working immediately
0x2D	0x56 0x60	Sets the image on a particular switch from an image saved in flash memory	**0x2D 0x56 0x60** (3 hex bytes) **0xBB 0xCC** (2 ASCII hex bytes switch #, 0 based) **0xDD 0xEE 0xFF 0xGG** (4 ASCII hex bytes for the image address)	0x2D 0x56 0x60 0x30 0x33 0x30 0x30 0x41 0x37 (Set image, switch 4, image 167 (0x00A7))	Switch number is 0 based. 1000 images maximum (0x3E8 is the maximum address). Going beyond the maximum address will result in image corruption.

Response	Description	Response Format	Response Example	Notes
0x50	Switch Pressed	**0x50 0xBB 0xCC 0xDD 0xEE** (4 ASCII hex bytes of switch number	0x50 0x31 0x34 0x36 0x41 (Switch press/release, switches 13,11,8,4, and 2 are pressed)	The bit position corresponds with the switch number, so bit 0 corresponds to switch 1, bit 1 corresponds to switch 2, etc. . A 1 signifies the switch is pressed, and a 0 signifies the switch is not pressed. Therefore, receiving 0000 0001 1000 1010 (0x01 0x8A) means switches 9, 8, 4, and 2 are pressed. 0x03FF would indicate all 10 switches are pressed.
0x61	Ready/Command received	**0x61**	0x61	
0x65	Busy processing previous command	**0x65**	0x65	If there was data associated with this command, it is ignored. The system is busy with the previous command. If sending for a specific bank, either re-send the command until 0x61 is received, or send data for a bank that is not busy
0x6E	Error, followed by error code	**0x6E**	0x6E 0x02	Error Codes: 01 - Invalid Command 02 - Timeout 03 - Not enough data 04 - Invalid data
0x79	Command successful	**0x79**	0x79	

S0602 CANOpen

Default Settings

Setting	Default	Object
Switch backlight	White	0x2001
Switch image number	Same as switch #	0x2002
Switch brightness	Max (0xF)	0x2003
Baud Rate	250k	0x2010
Boot-up Service	On	0x2011
Active on Startup	Not active (Pre-op)	0x2012
Node ID	0x15	0x2013

NMT

Start Node

Starts the node and allows PDOs to occur

ID	0x00		
Byte 0	1		
Byte 1	XX	The keypad with the CAN ID of XX	
Bytes 2-7	0x00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x000	01 15	Starts node 0x15

Stop Node

Stops the node

ID	0x00		
Byte 0	0x02		
Byte 1	XX	The keypad with the CAN ID of XX	
Bytes 2-7	0x00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x000	02 15	Stops node 0x15

Enter Pre-operational

Has the node enter pre-operational state (SDOs but not PDOs)

ID	0x00		
Byte 0	0x80		
Byte 1	XX	The keypad with the CAN ID of XX	
Bytes 2-7	0x00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x000	80 15	Node 0x15 enters pre-operational

Reset Node

Resets the node

ID	0x00		
Byte 0	0x81		
Byte 1	XX	The keypad with the CAN ID of XX	
Bytes 2-7	00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x000	81 15	Node 0x15 resets

Boot-up Service

Indicates the node has completed initialization and has entered pre-operational

ID	700 + Node ID	Default 715	
Byte 0	00		

Example:

Direction	CANOpen ID	Message	Notes
From Keypad	0x715	00	Node 0x15 has finished initialization and entered pre-operational

PDO

PDO messages have no response. The node MUST have been started (see Start Node command above) or these will not be enabled

Switch State

The state of every switch connected to the system

ID	0x180 + Node ID	Default 0x195
	Switch 0-7 state	
	S7 S6 S5 S4 S3 S2 S1	1 = pressed, 0 = released
Byte 0	S0	Switches have 0-based index
	Keys #8-#15 state	
	S15 S14 S13 S12 S11	
Byte 1	S10 S9 S8	Each bit is a switch state

	Keys #16-#23 state		
	S23 S22 S21 S20 S19		
Byte 2	S18 S17 S16	Each bit is a switch state	
	Keys #24-#31 state		
	S31 S30 S29 S28 S27		
Byte 3	S26 S25 S24	Each bit is a switch state	
Bytes 4-7	00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
From Keypad	0x195	00 00 00 00 00 00 00 00	No switch pressed
From Keypad	0x195	01 00 00 00 00 00 00 00	Switch 0 pressed
From Keypad	0x195	04 00 00 00 00 00 00 00	Switch 2 pressed
From Keypad	0x195	80 00 00 00 00 00 00 00	Switch 7 pressed
From Keypad	0x195	00 02 00 00 00 00 00 00	Switch 9 pressed
From Keypad	0x195	00 00 10 00 00 00 00 00	Switch 17 pressed

Set Backlight

Sets the backlight color on a specific switch

ID	0x200 + Node ID	Default 0x215	
Byte 0	Switch #	Switch number, 0-based index	
	Backlight		
Byte 1	xBGRxxxx	0x70 = white, 0x00 = black	
Bytes 2-7	00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x215	03 40 00 00 00 00 00 00	Switch 3 blue backlight
To Keypad	0x215	05 30 00 00 00 00 00 00	Switch 5 yellow backlight
To Keypad	0x215	00 50 00 00 00 00 00 00	Switch 0 magenta backlight

Set Image

Sets the image on a specific switch

ID	0x300 + Node ID	Default 0x315	
Byte 0	Switch #	Switch number, 0-based index	
Byte 1	Image # low byte		
Byte 2	Image # high byte	high byte << 8 + low byte = image number	
Bytes 3-7	00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x315	02 FF 00 00 00 00 00 00	Switch 2 image 0x00FF (255)
To Keypad	0x315	09 01 16 00 00 00 00 00	Switch 9 image 0x1601 (5633)
To Keypad	0x315	01 E5 03 00 00 00 00 00	Switch 1 image 0x03E5 (997)

Set Brightness

Sets the backlight brightness of all switches

ID	0x400 + Node ID	Default 0x415	
Byte 0	Brightness	Brightness level, from 0x0-0xF	
Bytes 1-7	00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x415	0F 00 00 00 00 00 00 00	Max brightness
To Keypad	0x415	00 00 00 00 00 00 00 00	Min brightness
To Keypad	0x415	08 00 00 00 00 00 00 00	Mid brightness

SDO

SDO messages have a very specific format. They can be used as long as the device isn't stopped.

Object 1000 Device Type

Reads the device type, S0602

ID	0x600 + Node ID	Default 0x615	
Byte 0	40		Read command
Byte 1	00	CAN object 1000	
Byte 2	10		
Bytes 3-7	00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 00 10 00 00 00 00 00	Read object 1000
From Keypad	0x595	43 00 10 00 00 02 06 53	Device type S0602

Object 1008 Device Name

Reads the device name (NKK_S0602)

ID	0x600 + Node ID	Default 0x615	
Byte 0	40		Read command
Byte 1	08	CAN object 1008	
Byte 2	10		
Bytes 3-7	00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 08 10 00 00 00 00 00	Read object 1008
From Keypad	0x595	41 08 10 00 90 00 00 00	Acknowledge, 9 byes to read

To Keypad	0x615	60 00 00 00 00 00 00 00	Start segment read
From Keypad	0x595	00 4E 4B 4B 5F 53 30 36	NKK_S06
To Keypad	0x615	70 00 00 00 00 00 00 00	Next segment read
From Keypad	0x595	1B 30 32 00 00 00 00 00	Final data, 02

Object 1009 Hardware Revision

Reads the hardware revision

ID	0x600 + Node ID	Default 0x615	
Byte 0	40		Read command
Byte 1	09	CAN object 1009	
Byte 2	10		
Bytes 3-7	00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 09 10 00 00 00 00 00	Read object 1009
From Keypad	0x595	4F 09 10 00 41 00 00 00	Rev A (starts at byte 4)

Object 100A Firmware Revision

Reads the firmware revision

ID	0x600 + Node ID	Default 0x615	
Byte 0	40		Read command
Byte 1	0A	CAN object 100A	
Byte 2	10		
Bytes 3-7	00	Not used	

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 0A 10 00 00 00 00 00	Read object 100A
From Keypad	0x595	43 0A 10 00 31 2E 30 30	V1.00 (starts at byte 4)

Object 1400 Receive PDO Communication Parameter 0

Describes the PDO for the set backlight command

ID	0x600 + Node ID	Default 0x615
Byte 0	40	Read command
Byte 1	00	CAN object 1400
Byte 2	14	
Byte 3	0	Number of mapped objects
	1	Mapped ID
	2	Transmission type
Bytes 4-7	0	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 00 14 00 00 00 00 00	Read object 1400 subindex 0
From Keypad	0x595	4F 00 14 00 02 00 00 00	2 mapped objects
To Keypad	0x615	40 00 14 01 01 00 00 00	Read object 1400 subindex 1
From Keypad	0x595	43 00 14 01 15 02 00 00	ID is 0x215
To Keypad	0x615	40 00 14 02 00 00 00 00	Read object 1400 subindex 2
From Keypad	0x595	4F 00 14 02 FF 00 00 00	Transmission type 0xFF

Object 1401 Receive PDO Communication Parameter 1

Describes the PDO for the set image command

ID	0x600 + Node ID	Default 0x615
Byte 0	40	Read command
Byte 1	01	CAN object 1401
Byte 2	14	
Byte 3	00	Number of mapped objects
	01	Mapped ID
	02	Transmission type
Bytes 4-7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 01 14 00 00 00 00 00	Read object 1401 subindex 0
From Keypad	0x595	4F 01 14 00 02 00 00 00	2 mapped objects
To Keypad	0x615	40 01 14 01 01 00 00 00	Read object 1401 subindex 1
From Keypad	0x595	43 01 14 01 15 03 00 00	ID is 0x315
To Keypad	0x615	40 01 14 02 00 00 00 00	Read object 1401 subindex 2
From Keypad	0x595	4F 01 14 02 FF 00 00 00	Transmission type 0xFF

Object 1402 Receive PDO Communication Parameter 2

Describes the PDO for the set brightness command

ID	0x600 + Node ID	Default 0x615
Byte 0	40	Read command
Byte 1		02 CAN object 1402
Byte 2	14	
Byte 3	00	Number of mapped objects
	01	Mapped ID
	02	Transmission type
Bytes 4-7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 02 14 00 00 00 00 00	Read object 1402 subindex 0
From Keypad	0x595	4F 02 14 00 02 00 00 00	2 mapped objects
To Keypad	0x615	40 02 14 01 01 00 00 00	Read object 1402 subindex 1
From Keypad	0x595	43 02 14 01 15 04 00 00	ID is 0x415

To Keypad	0x615	40 02 14 02 00 00 00 00	Read object 1402 subindex 2
From Keypad	0x595	4F 02 14 02 FF 00 00 00	Transmission type 0xFF

Object 1600 Receive PDO Mapping Parameter 0

Describes the PDO for the set backlight command

ID	0x600 + Node ID	Default 0x615
Byte 0	40	Read command
Byte 1	00	CAN object 1600
Byte 2	16	
Byte 3	00	Number of mapped objects
	01	Mapped Entry
Bytes 4-7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 00 16 00 00 00 00 00	Read object 1600 subindex 0
From Keypad	0x595	40 00 16 00 01 00 00 00	1 mapped objects
To Keypad	0x615	40 00 16 01 00 00 00 00	Read object 1600 subindex 1
From Keypad	0x595	40 00 16 01 10 01 01 20	Object 2001, subindex 1, 0x10 bits

Object 1601 Receive PDO Mapping Parameter 1

Describes the PDO for the set image command

ID	0x600 + Node ID	Default 0x615
Byte 0	40	Read command
Byte 1	01	CAN object 1601
Byte 2	16	
Byte 3	00	Number of mapped objects
	01	Mapped Entry
Bytes 4-7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 01 16 00 00 00 00 00	Read object 1601 subindex 0
From Keypad	0x595	40 01 16 00 01 00 00 00	1 mapped objects
To Keypad	0x615	40 01 16 01 00 00 00 00	Read object 1601 subindex 1
From Keypad	0x595	40 01 16 01 18 01 02 20	Object 2002, subindex 1, 0x18 bits

Object 1602 Receive PDO Mapping Parameter 2

Describes the PDO for the set brightness command

ID	0x600 + Node ID	Default 0x615
Byte 0	40	Read command
Byte 1	02	CAN object 1602
Byte 2	16	
Byte 3	00	Number of mapped objects
	01	Mapped Entry
Bytes 4-7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 02 16 00 00 00 00 00	Read object 1602 subindex 0
From Keypad	0x595	40 02 16 00 01 00 00 00	1 mapped objects
To Keypad	0x615	40 02 16 01 00 00 00 00	Read object 1602 subindex 1
From Keypad	0x595	40 02 16 01 08 01 03 20	Object 2003, subindex 1, 0x08 bits

Object 1800 Transmit PDO Communication Parameter 0

Describes the PDO for the switch state response

ID	0x600 + Node ID	Default 0x615
Byte 0	40	Read command
Byte 1	00	CAN object 1800
Byte 2	18	
Byte 3	00	Number of mapped objects
	01	Mapped ID
	02	Transmission type
Bytes 4-7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 00 18 00 00 00 00 00	Read object 1800 subindex 0
From Keypad	0x595	4F 00 18 00 02 00 00 00	2 mapped objects
To Keypad	0x615	40 00 18 01 00 00 00 00	Read object 1800 subindex 1
From Keypad	0x595	43 00 18 01 95 01 00 00	ID is 0x195
To Keypad	0x615	40 00 18 02 00 00 00 00	Read object 1800 subindex 2
From Keypad	0x595	4F 00 18 02 FF 00 00 00	Transmission type 0xFF

Object 1A00 Transmit PDO Mapping Parameter 0

Describes the PDO for the switch state response

ID	0x600 + Node ID	Default 0x615
Byte 0	40	Read command
Byte 1	00	CAN object 1A00
Byte 2	1A	
Byte 3	00	Number of mapped objects
	01	Mapped Entry
Bytes 4-7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 00 1A 00 00 00 00 00	Read object 1A00 subindex 0
From Keypad	0x595	40 00 1A 00 01 00 00 00	1 mapped objects

To Keypad	0x615	40 00 1A 01 00 00 00 00	Read object 1A00 subindex 1
From Keypad	0x595	40 00 1A 01 20 01 00 20	Object 2000, subindex 1, 0x20 bits

Object 2000 Switch State

The current state of every switch. Every bit is the corresponding switch number. 1 is pressed, 0 is released.

ID	0x600 + Node ID	Default 0x615
Byte 0	40	Read command
Byte 1	00	CAN object 2000
Byte 2	20	
Byte 3	01	Sub-index
Bytes 4-7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	40 00 20 01 00 00 00 00	Read object 2000 subindex 1
From Keypad	0x595	43 00 20 01 02 00 00 00	Switch 2 pressed (0-indexed)
To Keypad	0x615	40 00 20 01 00 00 00 00	Read object 2000 subindex 1
From Keypad	0x595	43 00 20 01 00 04 00 00	Switch 10 pressed (0-indexed)
To Keypad	0x615	40 00 20 01 00 00 00 00	Read object 2000 subindex 1
From Keypad	0x595	43 00 20 01 80 00 00 00	Switch 7 pressed (0-indexed)

Object 2001 Set Backlight

Sets the backlight color on a specific switch

ID	0x600 + Node ID	Default 0x615
Byte 0	2B	Write command
Byte 1	01	CAN object 2001
Byte 2	20	
Byte 3	01	Sub-index
Byte 4	Switch #	0-indexed
	Backlight color	
Byte 5	xBGRxxxx	0x70 = white, 0x00 = black
Bytes 6-7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	2B 01 20 01 03 40 00 00	Switch 3 blue backlight
From Keypad	0x595	60 01 20 01 00 00 00 00	
To Keypad	0x615	2B 01 20 01 05 30 00 00	Switch 5 yellow backlight
From Keypad	0x595	60 01 20 01 00 00 00 00	
To Keypad	0x615	2B 01 20 01 00 50 00 00	Switch 0 magenta backlight
From Keypad	0x595	60 01 20 01 00 00 00 00	

Object 2002 Set Image

Sets the image on a specific switch

ID	0x600 + Node ID	Default 0x615
Byte 0	27	Write command
Byte 1	02	CAN object 2002
Byte 2	20	
Byte 3	01	Sub-index
Byte 4	Switch #	0-indexed
Byte 5	Image # low byte	
Byte 6	Image # high byte	high byte << 8 + low byte = image number
Byte 7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	27 02 20 01 02 FF 00 00	Switch 2 image 0x00FF (255)
From Keypad	0x595	60 02 20 01 00 00 00 00	
To Keypad	0x615	27 02 20 01 09 01 16 00	Switch 9 image 0x1601 (5633)
From Keypad	0x595	60 02 20 01 00 00 00 00	
To Keypad	0x615	27 02 20 01 01 E5 03 00	Switch 1 image 0x03E5 (997)
From Keypad	0x595	60 02 20 01 00 00 00 00	

Object 2003 Set Brightness

Sets the backlight brightness of all switches

ID	0x600 + Node ID	Default 0x615
Byte 0	2F	Write command
Byte 1	03	CAN object 2003
Byte 2	20	
Byte 3	01	Sub-index
Byte 4	Brightness	Brightness level, from 0x0-0xF
Bytes 5-7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	2F 03 20 01 0F 00 00 00	Max Brightness
From Keypad	0x595	60 03 20 01 00 00 00 00	
To Keypad	0x615	2F 03 20 01 00 00 00 00	Min Brightness
From Keypad	0x595	60 03 20 01 00 00 00 00	
To Keypad	0x615	2F 03 20 01 08 00 00 00	Mid Brightness
From Keypad	0x595	60 03 20 01 00 00 00 00	

Object 2010 Set Baud Rate

Sets the baud rate the device communicates at

ID	0x600 + Node ID	Default 0x615
Byte 0	2F	Write command
Byte 1	10	CAN object 2010
Byte 2	20	
Byte 3	00	Sub-index

			0x0F	125k
Byte 4	Baud Rate	If invalid, defaults to 250k	0x10	250k
			0x11	500k
			0x12	1 Meg
Bytes 5-7	00	Not used		

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	2F 10 20 00 0F 00 00 00	125k
From Keypad	0x595	60 10 20 00 00 00 00 00	
To Keypad	0x615	2F 10 20 00 10 00 00 00	250k
From Keypad	0x595	60 10 20 00 00 00 00 00	
To Keypad	0x615	2F 10 20 00 12 00 00 00	1 Meg
From Keypad	0x595	60 10 20 00 00 00 00 00	

Object 2011 Set Boot-up Service

Sets whether the device sends a device initialized message (0x700 + Node ID) after reset

ID	0x600 + Node ID	Default 0x615		
Byte 0	2F	Write command		
Byte 1	11	CAN object 2011		
Byte 2	20			
Byte 3	00	Sub-index		
			0x00	Disabled
Byte 4	Baud Rate	If invalid, defaults to on	0x01	Boot-up service enabled
Bytes 5-7	00	Not used		

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	2F 11 20 00 00 00 00 00	Disable boot-up service
From Keypad	0x595	60 11 20 00 00 00 00 00	
To Keypad	0x615	2F 11 20 00 01 00 00 00	Enable boot-up service
From Keypad	0x595	60 11 20 00 00 00 00 00	

Object 2012 Set Device Active After Reset

Sets whether the device immediately enters the operational state after reset (which enables PDOs)

ID	0x600 + Node ID	Default 0x615		
Byte 0	2F	Write command		
Byte 1	12	CAN object 2012		
Byte 2	20			
Byte 3	00	Sub-index		
			0x00	Disabled
Byte 4	Flag	If invalid, defaults to off	0x01	Active after reset enabled
Bytes 5-7	00	Not used		

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	2F 12 20 00 00 00 00 00	Disable active after reset
From Keypad	0x595	60 12 20 00 00 00 00 00	
To Keypad	0x615	2F 12 20 00 01 00 00 00	Enable active after reset
From Keypad	0x595	60 12 20 00 00 00 00 00	

Object 2013 Set Node ID

Sets the node ID of the device

ID	0x600 + Node ID	Default 0x615
Byte 0	2F	Write command
Byte 1	13	CAN object 2013
Byte 2	20	
Byte 3	00	Sub-index
Byte 4	Node ID	0x01-0x7F; If invalid, defaults to 0x15
Bytes 5-7	00	Not used

Example:

Direction	CANOpen ID	Message	Notes
To Keypad	0x615	2F 13 20 00 0F 00 00 00	Change node ID to 0x0F
From Keypad	0x595	60 13 20 00 00 00 00 00	
To Keypad	0x615	2F 13 20 00 6A 00 00 00	Change node ID to 0x6A
From Keypad	0x595	60 13 20 00 00 00 00 00	
To Keypad	0x615	2F 13 20 00 00 00 00 00	Invalid; Node ID will be 0x15
From Keypad	0x595	60 13 20 00 00 00 00 00	