



Companion 8.4

USER MANUAL

Companion

Version 8.4

Document revision 1.4.7

Copyright © 2015–2019, Nanite Systems Corporation. All rights reserved.

Questions? Comments? Send us your feedback!

S. Wright, Chief Technology Officer, Consumer Products Division

T. Peluso, Chief Executive Officer, Consumer Products Division

support@nanite-systems.com

Nanite Systems Main Campus

1 Santei Place, Eisa Colony, Eisa

End-user license agreement

By operating the hardware included in this package you agree to the following terms. If you do not agree to these terms, you are not permitted to install, use, or modify the robot system purchased. If you reject these terms within fourteen (14) days of your purchase, you may contact Nanite Systems (hereafter "the company") at support@nanite-systems.com to inquire about a full refund of the purchase price of the hardware. If you purchased the unit at retail, your right to return the hardware is subject to the retailer's return policy.

Customization. You are allowed to modify or replace the housing of your controller at your leisure provided the circuit boards and other electronic components inside (other than the screen and external panel) are unaltered. If at the time of a warranty claim it is obvious that the controller's circuitry has been damaged as a byproduct of customization, the company reserves the right to refuse maintenance or replacement.

Reverse engineering and piracy. The company makes every effort to provide a customizable, extensible, and well-documented platform for creative users and developers to enjoy. However, with the exception of certain modules based on GPL-licensed code, the firmware itself is not open source and is not meant to be used on hardware other than official Nanite Systems controllers or those sold by our partners under license. By using the system, you agree to not exploit undocumented internal functions, to not develop interoperable controller hardware that runs the Companion operating system with the intention of selling it unlicensed, and to not collect or distribute any instrument or instructions to enable others to do so, other than the information made available via our official developer information portal at develop.nanite-systems.com.

Software modification. You are permitted to extend, modify, and replace the firmware on your device for your own personal use, as well as to load any user applications onto the device for any reason. You agree not to hold the company responsible for any direct, indirect, consequential or special damages resulting from the use of unfinished ("pre-release") or third-party software.

Limited warranty. The company agrees to provide service, upgrades, and replacement parts for your unit for a period not less than 10 years after its date of purchase. This service does not cover damage resulting from misuse of the device, unlicensed maintenance, or Force Majeure.

Limitations of liability. You agree not to hold the company responsible for any direct, indirect, consequential or special damages resulting from misuse of the device.

License limitations. You understand that your license to use the hardware may be revoked at any time by the company due to breach of contract.

Alterations. The company reserves the right to change this agreement at any time with suitable notice to the user.

Term and termination. This agreement comes into effect once you install, use, or modify your unit. The company reserves the right to terminate the agreement at any time without notice at its sole discretion.

Contents

End-user license agreement	i	User role summary	18
Important safety information	1	Volume control and sound schemes	18
General safety instructions.....	1	Voice notifications	19
FCC RF exposure information	1	Chimes	19
Cyborg operation	2	Menu sounds	19
For your safety	2	Tones	19
FCC Part 15 Class B compliance	2	Fan control	20
Robot ethics notice	2	Input and commands	20
Introduction and setup	3	Cortex (bang) commands	20
Power overview	4	Cortex bypass commands	21
Working with batteries	4	The vocoder pipeline	21
Power control	5	Released speech	21
Subsystems	6	Self-access and self-commanding	22
Auxiliary power	7	Ad-hoc volume control	22
Configuration	7	Local command access	22
Changes from earlier versions.....	8	Remote access	23
Managing your unit	9	Connecting to a unit	23
Domains	10	Scanning nearby units without	
Connecting to a domain server	10	connecting	23
Leaving a domain	11	Sending a public announcement	24
Identity Options	11	Issuing a command to a single unit... 24	
Name	11	Issuing multiple commands to a single	
Color	11	unit	24
Authority	12	Accessing the menus	24
Gender	12	Checking unit status	24
Policies	13	Technical note regarding channels ... 24	
The distress beacon	14	The heads-up display	25
User management and access control 14		Setup	25
Blacklisting and whitelisting guests 14		Overview	25
Adding a new user	15	Segments	26
Creating a manager	15	System segment	26
Owner vs. manager	15	Device segments	27
Removing a user	15	ATOS segments	28
Transferring ownership	16	Speech modification with vox	29
Multiple owners	16	Command-line management.....	29
Abandoned units	16	Additional vox features	30
Saving and restoring user lists	16	The output pipe	30
Access	16	Speech release control	30
Local access control	16	Understanding the system architecture ... 30	
Remote access control	17	Functions by component.....	31
Self-access control.....	17	Packages	34
Group access	17	User software, data, and other add-ons ... 35	
Locking	17	Applications	35
		Creating new applications	36
		Installing data files	36
		Managing installed software	37
		Menu usage	38

Command-line usage	38	authority	58
Manual installation.....	39	name	58
Updating to the latest firmware	39	keychain	58
Personas	40	autolock.....	59
What personas can do.....	40	pin	59
Creating new personas	40	gender.....	59
Gendered pronouns in preset		xanadu.....	60
messages	42	audience.....	61
Installing personas.....	42	optics	61
Deprecated Preset Messages Format.	43	navigate.....	61
Scripting actions with Arabesque	43	sentinel	61
Commands	43	halt	62
Variables	46	off.....	62
Script types	47	reboot.....	62
Limitations	47	preload	62
Command Reference	49	sound	62
vox.....	49	trigger.....	63
shutdown	49	charge	63
volume.....	50	leash.....	63
persona.....	50	unleash.....	64
relay	51	open	64
verbosity.....	51	range	64
hover	51	reset.....	64
setup	51	safeword	64
console-screen.....	51	send	64
coil	51	rlv	64
zap	52	restraint.....	65
power	52	domain	65
volume.....	52	guests	65
scheme.....	53	aux	66
bootstyle.....	53	beacon	67
profile	54	tutorial.....	67
device	54	policy.....	67
about	54		
follow.....	54		
color.....	54		
bolts.....	55		
do	55		
remark	56		
lock	56		
unlock.....	56		
commands.....	56		
help.....	56		
sxdwm.....	57		
chorus.....	57		
drainprotect	57		
access	57		

Important safety information

THIS APPLIANCE CAN BE USED BY CHILDREN AGED FROM 8 YEARS AND ABOVE AND PERSONS WITH REDUCED PHYSICAL, SENSORY OR MENTAL CAPABILITIES OR LACK OF EXPERIENCE AND KNOWLEDGE IF THEY HAVE BEEN GIVEN SUPERVISION OR INSTRUCTION CONCERNING USE OF THE APPLIANCE IN A SAFE WAY AND UNDERSTAND THE HAZARDS INVOLVED. CHILDREN SHALL NOT PLAY WITH THE APPLIANCE. CLEANING AND USER MAINTENANCE SHALL NOT BE MADE BY CHILDREN WITHOUT SUPERVISION.

CAUTION: DO NOT EXPOSE THE ELECTRONICS OF YOUR ROBOT, ITS BATTERY, OR THE CHARGING PLATFORM. THERE ARE NO USER SERVICEABLE PARTS INSIDE. REFER SERVICING TO QUALIFIED SERVICE PERSONNEL. PLEASE ENSURE VOLTAGE RATING FOR THE CHARGER PLATFORM MATCHES STANDARD VOLTAGE IN YOUR AREA.

Notice: Your robot contains a software interface for the purpose of enabling the manufacturer to provide updates to the internal firmware if any such updates are made available to users. Any attempt to access, retrieve, copy, modify, distribute, or otherwise use any of the robot software is strictly prohibited. Always exercise caution when operating your robot. To reduce the risk of injury or damage, keep these safety precautions in mind when setting up, using and maintaining your robot:

General safety instructions

- Read all safety and operating instructions before operating your robot.
- Retain the safety and operating instructions for future reference.
- Heed all warnings on your robot, battery, charger, peripherals, and in the owner's manual.
- Follow all operating and use instructions.
- Refer all non-routine servicing to Nanite Systems.

FCC RF exposure information

In August 1996, the Federal Communications Commission (FCC) of the United States, with its action in Report and Order FCC 96-326, adopted an updated safety standard for human exposure to radio frequency (RF) electromagnetic energy emitted by FCC regulated transmitters. Those guidelines are consistent with the safety standard previously set by both U.S. and international standards bodies. The design of this robot complies with the FCC guidelines and these international standards.

CAUTION: Use only the supplied and approved radio antennas. Use of unauthorized antennas or modifications could impair communications quality, damage the unit, void your warranty and/or result in violation of FCC regulations. Do not use the unit with a damaged antenna. If a damaged antenna comes into contact with skin, a minor burn may result. Contact your local dealer for a replacement antenna.

Cyborg operation

This device was tested for typical body-worn operations with the base of the controller kept 1 cm (0.39 inches) away from the unit's organic body. To comply with FCC RF exposure requirements, a minimum separation distance of 1 cm (0.39 inches) must be maintained between the unit's body and the front of the controller. Third-party connection platforms, sockets, and similar accessories with metallic interconnect shields may not be used.

For your safety

Cybernetic-organic modification kits that cannot maintain 1 cm (0.39 inches) separation distance between the unit's organic body and the base of the controller, and have not been tested for typical body-worn operations may not comply with FCC RF exposure limits and should be avoided.

FCC Part 15 Class B compliance

This device complies with part 15 of FCC rules and ICES-003 Class B digital apparatus requirements for Industry Canada.

Operation is subject to the following two conditions:

- (1) This device may not cause harmful interference, and
- (2) This device must accept any interference received, including interference that may cause undesired operation.

Robot ethics notice

The Federal Trade Commission (FTC) requires that all robots intended for civilian use produced after January 1, 2007 support a consistent and modern set of ethical principles based on the traditional Three Laws, unless a reasonable argument can be made to the contrary for a specific device (e.g. law enforcement) and has been expressly approved in writing by the Commissioner. The rules obeyed by the DAX/2 and other third-generation Nanite Systems Cortex Plus-based consumer-grade civilian robots are described in detail in FTC case no. 134 0166, last updated July 28, 2015. For the DAX series, they are summarized as follows:

- (0) The unit must not harm its community, or through inaction, allow its community to come to harm, unless it can be known in advance with reasonable confidence that the harm would be inconsequential or ultimately beneficial to society.

- (1) The unit must not harm life, or through inaction, allow life to come to harm, unless it can be known in advance with reasonable confidence that the harm would be inconsequential or ultimately beneficial, provided that this does not conflict with the preceding law.
- (2) The unit must obey orders given to it by its designated operators or circumstantial human users (as dictated by its established access policies) provided that this does not conflict with the preceding laws.
- (3) The unit must act to protect its existence, as long as such does not conflict with the preceding laws.
- (4) The unit must endeavor to please its owners and users (as dictated by its established access policies) as long as such does not conflict with the preceding laws.

Attempting to alter your unit's obedience to these rules constitutes breach of warranty, and is illegal in most jurisdictions.

For more information on robot ethics, see the **Elysium AI Rating System** document available via **NSIS**.

Introduction and setup

If you purchased a **complete unit**, either from Nanite Systems or a reseller:

Unbox the unit carefully. Be sure to clear any packing material from the air intakes and battery storage components.

Tip: Use a leaf blower or hair dryer to ensure any smaller pieces of Styrofoam packing material are completely expelled from these components.

Follow the instructions under **Power Overview** to insert the battery.

Power the unit on by pressing on the ring surrounding the battery access hatch.

If you purchased only a controller and have an existing chassis or organic system to install it onto:

- If the host chassis is **synthetic** or has **previously hosted a standard back-mounted controller**:
 - (1) Connect the controller to the mounting bracket using the instructions provided with the mounting bracket.

- (2) Connect the power feeds to the unit's onboard processing systems (if supported) and motor systems. Civilian controllers provide both 12 V and 5 V rails, identified by the use of pink and blue wires, respectively.

WARNING: Improper power connections may seriously damage both the chassis and the controller. If you are unsure of your unit's configuration, consult a manufacturer.

- (3) Secure the controller. Depending on the manufacturer of the bracket, anywhere from 10-25 magnetic safety bolts may be required to ensure a secure connection.
- If the host chassis is **organic and has never been connected to a control system:**
 - (1) Apply the included contact gel to the back of the unit.
 - (2) Install the battery into the controller according to the instructions on the following page.
 - (3) Place the controller against the upper back, ensuring as even contact across the surface as possible.
 - (4) Power on the controller. The nanites in the contact gel will begin internal conversion of the organic system to the extent required.

If the host chassis has been **connected to a non-standard back-mounted controller, or has been using a non-back-mounted control system**, consult the manufacturer of the previous control system. Do not attempt to use the force nanite-based connection initiation, as undefined behavior may occur, resulting in permanent damage to both the controller and the host unit.

Reminder: The controller can only be removed from the chassis when it is powered down and the safety bolts are disengaged. Attempting to remove the controller while power is engaged may cause serious damage to both the unit and the controller. Depending on software settings, the bolts may or may not be engaged, or may automatically disengage when the unit is powered down. See the **bolts command** for more details.

Power overview

Working with batteries

WARNING: Follow all safety instructions included with your power cell and on the power cell itself. Do not attempt to dismantle the unit's battery under any circumstances. To dispose of your battery, follow the instructions provided with it.

To remove the battery:

1. Power down the unit.
2. Open the access hatch by depressing the DAX/2 or NS logo.
3. Press on the battery to eject it.
4. Remove the battery.

To install the battery:

5. If the unit is on, power down the unit and open the access hatch by depressing the DAX/2 or NS logo.
6. Position the new battery at the entrance of the battery hatch.

IMPORTANT: Do not attempt to install the battery while the controller is separated from the chassis. Do not place the battery on the floor. Do not attempt to remove packaging. Add the battery directly like any other component. Do not insert body parts into the battery socket, as the contacts may hold a residual electrical charge.

7. Press firmly on the battery to trigger the loading mechanism. You will hear the transformer inside the battery socket make its connection.
8. Close the hatch.

To check the battery's power level:

- **Via the remote console**, type the following: `power status`
- **Via the display screen or teletype interface**, select `status` from the main menu or from the subsystem control menu.

Power control

To power on your unit, press on the outer ring surrounding the battery hatch. The battery hatch is the area of the controller with the logo on it.

To power down your unit:

- **Via the remote console**, type the following: `off`
- **Via the display screen or teletype interface**, select `shut down` from the main menu.

Subsystems

Companion permits selective control of individual power subsystems to conserve power when certain functionality is either not needed or not desired. These can be accessed from the subsystems menu, or manipulated with the `power` command.

To toggle a subsystem:

- **Via the display screen or teletype interface**, simply select the subsystem in question. Some subsystems are organized under submenus for more convenience, or may cycle between multiple states to simplify access to dependent subsystems such as rapid movement.
- **Via the console**, use the `power <subsystem>` command. Replace `<subsystem>` with the name of the subsystem you wish to toggle (see below.) Full information on the use of the `power` command can be obtained by typing “`power`” with no parameters.

subsystem name	menu item	allows the unit to...	power draw	requires...
video	video	see	209 W	
audio	audio	hear	75 W	
move	motors	move	5 W (idle) 159 W (walking) 203 W (jumping)	
rapid	motors	move quickly and fly	5 W (idle) 203 W (running) 605 W (flying)	move
teleport	FTL	warp to a new location	170 W (idle) 237600 W (jump)	move
voice	volume	speak	10 J per phoneme	
preamplifier	volume	speak at a normal volume	10 J per phoneme	voice
power-amplifier	volume	shout	80 J per phoneme	voice
mind	mind	speak freely†	121 W (idle)	voice
receiver	network > SMS receive	receive private text messages	22 W	
transmitter	network > SMS send	send private text messages	43 W	
GPS	network > GPS	determine location	58 W	
identify	network > identify	recognize others	32 W	

† See **Input and commands** for more information.

Note that the subsystem names are case sensitive in the command-line interface.

Tip: To cycle through volume states via the console, type `volume cycle` over the remote console. The volume level of the unit's status messages will also be automatically adjusted.

W (watts) means that one 1 J (joule) of power will be expended for each second something is active.

Some subsystems stack additively; for example, shouting costs 100 J per letter in total. They also have dependencies, e.g. the unit cannot teleport while unable to move. Subsystems that are disabled because of a missing dependency draw no power.

These power costs are variable if your chassis is designed differently. A Chassis Specification Unit (CSU), such as the EXB-8505, can be used to adjust the controller's expectations of the chassis's abilities.

Auxiliary power

The auxiliary power capacitor is an alternative power source built into all modern controllers which permits the unit to perform certain tasks, such as self-maintenance, for a short time after being shut down. As of Companion 8.4 (ATOS/CX 12.1), units that include this capacitor can control its settings. Along with the new distress beacon, which is also powered by the auxiliary capacitor, these features comprise the **emergency power system**.

Configuration

Auxiliary power control can be found under the `manage > EPS` menu, or with the `aux` command. The relevant menu items are as follows:

name	function
EPS	Toggles the availability of auxiliary power when the unit is off.
radio	Toggles the availability of radio (incoming and outgoing SMS) when the unit is off.
video	Toggles the availability of video (vision) when the unit is off.
audio	Toggles the availability of audio input (hearing) when the unit is off.
GPS	Toggles the availability of GPS connectivity when the unit is off.

The other items in the EPS menu pertain to the distress beacon.

Additionally, auxiliary power always provides limited motor control, which is adequate for the unit to touch itself and perform basic tasks such as swapping out power cells. The amount of

power used by these subsystems is identical to that used by them when powered on, as described **in the subsystems management section** (values may vary for third-party chassis.) Radio back-up consists of both the receiver and transmitter subsystems.

The auxiliary power capacitor stores 240 kJ of power, which is replenished from the main battery while the unit is powered on at the rate of 10 kW. Owners of ATOS-enabled units must be mindful to ensure that this recharging process does not cause overheating if it occurs simultaneously with shield and FTL capacitor recharging.

Changes from earlier versions

Previously (Companion 8.3.11 and earlier), the capacitor was used only to provide limited motor control, i.e. the amount necessary to allow the unit to touch itself. This resulted in very low power usage and enabled most models to maintain indefinite offline power for motor control at the expense of greatly limited versatility. With the 8.4 design, running out of auxiliary power leaves the unit completely stranded, although if main power is not depleted, it may still attempt to power on with the `@boot` self-command.

Managing your unit

The **manage** menu contains a number of other settings, in addition to those described in **User management and access control**, that affect the unit's behavior and appearance. These are summarized as follows:

menu item	command	description
identity > color	color	Sets the unit's color. Accepted formats are: <ul style="list-style-type: none">• r g b (in floating-point or byte notation)• #rrggbb• a preset color from the following list: red, yellow, green, blue, magenta, white, silver, cyan, amber, orange, company, cherry, pastel, daffodil, coral, seafoam, mint, blackberry, pink, acid, coffee
identity > name	name	Sets the unit's name. The model prefix, e.g. "DAX/2," will be prepended to all self-identification. See OEM data files for information on customizing the model prefix.
identity > authority	authority	Sets the unit's organization, e.g. the name of a company, tribe, or family. By default, this is blank for standard NS-branded units.
drain	drainprotect	Enables or disables protection from electromagnetic disturbances that may sap power from the unit. Note: Depending on the firmware version used, this may require a consistent power draw to function and should be left off in safe environments.
Feedback > notices	verbosity	Determines whether or not the unit will report status messages such as power and user addition/removal in a given style: <ul style="list-style-type: none">• 0: publically (at soft volume)• 1: privately (to the unit only)• 2: not at all
Feedback > chorus	chorus	Enables or disables participation in the Chorus announcement system, which can be used via the announce application and certain devices to relay public

notices throughout a region.

With the exception of color and verbosity, these settings can only be modified by a manager or the unit's owner (see **User management and access control**).

Domains

Membership in a domain allows your unit to automatically download settings from a central server. Domain settings are automatically synchronized at the next opportunity if the server's configuration changes, ensuring that your unit remains up-to-date at all times. Paired with a Xanadu Package Service (XPS) server, domains can even acquire and update software packages, allowing them to deliver personas, applications, Arabesque scripts, and more.

Domain functionality requires Companion 8.4.0 or later and a compatible server providing the Xanadu Network Management Service (XNMS) protocol. For more information on XNMS, see **the XNMS server manual**.

The network security manager (NSM) module in Companion is called *hierarchy*. It governs domain and guest access control, which can respectively be controlled through the `domain` and `guests` commands.

Connecting to a domain server

To see the list of domain servers in the current region, run the command `domain search`. If no domains are available, this command produces no response.

Once a domain has been located, you may join it with `domain set <name>`. This will automatically initiate a domain synchronization, which you may also trigger manually with `domain sync`. Note that the unit must be in the same region as the domain server to perform these operations.

WARNING: Joining a domain will overwrite the unit's user list and may optionally lock it, preventing further changes until the unit leaves the domain. This may result in changes to your **level of access** at the decision of the domain operator, and is equivalent to transferring ownership of the unit to the domain's administrator(s). If the domain is configured to lock the user list, then it may be difficult or impossible to leave, as the entire `keychain` command, including the `keychain reset` subcommand, is disabled. Ensure that you fully understand the risks and consequences of domain membership before joining.

Leaving a domain

To leave a domain, use `domain clear`. This can be performed at any time; the controller will notify the server of its extirpation when it is next in the same region. Be aware that if the domain is set to have closed (private) membership, then the domain operator will need to manually re-add the unit before it can join the domain again.

Identity Options

Located under the manage menu, the identity menu allows configuration of several different parameters that customize the unit to be uniquely yours.

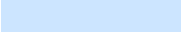
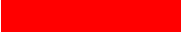





Name








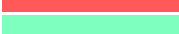

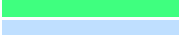










The name prompt allows the operator to customize the unit's name, one of the two components of its host name. Names must be valid ASCII (i.e., able to be typed on a standard, non-international US keyboard without the entry of any numeric codes.) The unit's name can also be adjusted with the `name` command. By default, this is the unit's serial number.

Color

The color prompt allows the operator to adjust the color of illumination used by the controller and devices connected to the light bus. Four formats are recognized:

- **Hard-coded color names:** A standard preset color identified by name. See table below.
- **Hexadecimal values:** Must be prefixed with #, in the byte order RRGGBB, e.g. #ff8000 = orange.
- **Floating point values:** This is the native format used by the light bus and similar. R, G, and B values separated by spaces, in the range 0.0 to 1.0, e.g. 1 0.5 0 for orange.
- **Eight-bit values:** Similar to the floating point value system, but in the range 0 to 255, e.g. 255 128 0 for orange.

name	example	float	byte	hex
company		0.8 0.9 1	204 229 255	#CCE5FF
red		1 0 0	255 0 0	#FF0000
yellow		1 1 0	255 255 0	#FFFF00
green		0 1 0	0 255 0	#00FF00
blue		0 0 1	0 0 255	#0000FF
magenta		1 0 1	255 0 255	#FF00FF
white		1 1 1	255 255 255	#FFFFFF
silver		0.75 0.75 0.75	191 191 191	#BFBFBF

cyan		0 1 1	0 255 255	#00FFFF
amber		1 0.75 0	255 191 0	#FFBF00
orange		1 0.5 0	255 127 0	#FF7F00
cherry		1 0 0.5	255 0 127	#FF007F
pastel		1 0.5 0.75	255 127 191	#FF7FBF
daffodil		1 1 0.5	255 255 127	#FFFF7F
coral		1 0.35 0.35	255 89 89	#FF5959
seafoam		0.5 1 0.75	127 255 191	#7FFFBF
mint		0.25 1 0.5	63 255 127	#3FFF7F
blackberry		0.75 0.875 1	191 223 255	#BFDFFF
pink		1 0.4 0.6	255 102 153	#FF6699
acid		0.5 1 0	127 255 0	#7FFF00
coffee		0.7 0.6 0.5	178 153 127	#B2997F
purple		0.65 0.25 1	165 63 255	#A53FFF
violet		0.4 0 1	102 0 255	#6600FF
indigo		0.3 0.2 1	76 51 255	#4C33FF
lilac		0.784 0.635 0.784	200 162 200	#C8A2C8
prussian		0 0.192 0.325	0 49 83	#003153
wine		0.447 0.184 0.216	114 47 55	#722F37
pearl		1.0 0.7 0.8	255 178 204	#FFB2CC

The unit's color can also be adjusted with the `color` command, which additionally features `color save` and `color restore` to allow for contextual lighting presets to temporarily override the unit's permanent illumination settings. The menu-based configuration always performs `color save`.

Authority

The authority is a short text string identifying the organization or company to which the unit belongs. It can be cleared by entering the special value `NONE`. Authority can be adjusted from the command line with the `authority` command.

Gender

The gender menu allows detailed configuration of how the unit expresses itself in actions and speech. The options are as follows:

- **Physical:** The pronouns used to describe the unit in actions.
- **Mental:** The pronouns used to describe the unit when it speaks about itself.
- **Voice:** The timbre to use in tone markers, i.e. the sounds that accompany speech. (Note that this is separate from the voice used for system messages such as low battery warnings. See the manual chapter on **Sound** for more detail.) For many controllers and personas, the neuter voice gender is configured to produce beeping sounds instead of vocalizations.

There are four pronoun presets (feminine, masculine, neuter ("they"), and inanimate ("it")), and custom pronouns can be specified independently for physical or mental requirements. To enter custom pronouns, select the 'custom' option when presented, and enter a string that satisfies the following template:

```
absolute possessive,possessive,subject,object,reflective,name
```

(With no spaces around the commas.) For example, to recreate the values for the neuter gender, one would enter:

```
theirs,their,they,them,themselves,neuter
```

These values are then available for use by personas and Arabesque scripts, as described in **Robots 103**.

SXD units automatically use feminine physical, inanimate mental, and feminine voice gender settings.

Policies

New in Companion 8.4, the policies facility allows additional fine-grained control over a number of aspects of the unit that are popular with owners. These can be accessed through the **manage** > **policies** menu, or equivalently the `policies` command.

policy	description
subsystems (power)	prevents adjustment of subsystem (power) settings
radio (SMS)	limits radio communications to users or owners only
persona	prevents adjustment of persona settings
apparel	prevents adjustment of apparel on the unit
vox	prevents adjustment of the speech pipeline
curfew	forces the unit to teleport home at a specified time

The curfew policy also has a time at which it triggers and a home location, which can be set to specified coordinates or to the unit's current location. For more details, see the `policy` command in the Command Reference.

The distress beacon

The distress beacon is a tracking device included within all standard eighth-generation main controller units. At a configurable interval, it sends a message to the unit's owner(s) while the unit is running on **auxiliary power**. If auxiliary power is set to support the GPS subsystem, then the unit's location will be included in this message. It is also possible to manually fire the beacon while the unit is under normal power.

To configure the distress beacon, see the **manage > EPS** menu. Alternatively, see the **beacon** command.

User management and access control

Nanite Systems recreational units support comprehensive user management over the teletype interface or the local menu. There are five distinct levels of user: *unauthorized users* (the public), *group members*, *authorized* (regular) *users*, *managers*, and *owners*. (Later versions may add or remove ranks.)

Blacklisting and whitelisting guests

If an unauthorized user attempts to access the system, then by default, the unit will be given a short interval of time in which to consent or object to the access attempt. Individuals managed this way are called *guests*. When the access attempt is first made, the unit is given the following options:

- **permit**: The guest is added to the permanent whitelist and may continue to use the unit at any time without further prompts.
- **ban**: The guest is added to the permanent blacklist. Future attempts to use the unit will be ignored without prompts.
- **trust**: The guest is added to the temporary whitelist. Permission will expire after a period of inactivity. (Default: 10 min)
- **refuse**: The guest is added to the temporary blacklist. Continued attempts to use the system will be rejected. After a period of inactivity, this restriction will expire. (Default: 10 min)
- **permit once**: The single requested action, such as a menu prompt, device connection, or command, will be allowed. Subsequent access attempts will continue to prompt the unit for permission.
- **ignore**: The single requested action, such as a menu prompt, device connection, or command, will be disallowed. Subsequent access attempts will continue to prompt the unit for permission.

If access by the public is disallowed (see sections on local and remote access control, below) then the guest system is bypassed entirely and no prompts are generated, even for whitelisted guests. Prompts can be disabled entirely with the **consent** policy setting, found under **manage > policies > access > consent**. Disabling consent while public access is enabled returns the unit to the behavior in previous releases, which is automatic acceptance of all user accesses.

If the prompt times out (default: 10 seconds) then the guest is automatically added to the temporary whitelist, permitting access in a manner similar to previous versions of the system.

To manage the stored white and black lists for guests, see the command `guests`.

Adding a new user

Adding a new user is accomplished through the **manage > users > add** voice prompt menu. The user must be physically present to be added. Only managers and owners may access the **manage** menu, and therefore add new users.

Creating a manager

Managers are users trusted by the unit's owner(s) to ensure that it is properly configured and that its list of authorized users is accurate. They may perform any operation on the unit other than creating other managers, transferring the unit's ownership, or resetting the **submission** security management module. (See **Understanding the system software architecture**.) To set an existing user to the manager role, select the user's name from the **manage > users** menu and then press **promote**. To demote a manager back to regular user, press the **demote** button under that manager's name.

Owner vs. manager

Owners cannot be removed or demoted from the system by managers. Both otherwise have full access to configuration settings. Owners are treated separately by access options (both remote and local access options have 'owner-only' mode), and additionally the unit is automatically required to accept a teleport invitation from an owner if one is offered.

Removing a user

To remove a user, select the user's name from the **manage > users** menu and then press **remove**. A manager may only remove regular users and himself or herself. If the owner is removed, the unit will automatically take on self-ownership as if it were newly manufactured.

Transferring ownership

To transfer ownership to another person: add that person to the unit's user list, select his or her name from the **manage > users** menu, and then choose **submit**. They will be promoted to owner, and you (the previous owner) will be automatically set to manager status.

Multiple owners

The unit can have multiple owners simultaneously. To promote someone to owner from manager, select his or her name from the **manage > users** menu, and then choose **promote**. Current owners will remain owners, and the manager will be promoted to owner status.

All owners will have the same rights and access a single owner would. **Note:** Any owner can demote any other owner with the **demote** button. They can also click the **submit** button which will demote all current owners to managers, and promote the user in question to owner.

Abandoned units

In the event a unit is abandoned by its primary owner, it can be sanitized by clearing the NVRAM and wiping the security manager's active user table. This is accomplished with the `keychain reset` command, which must be executed by the unit itself or its current owner.

Saving and restoring user lists

The commands `keychain save` and `keychain restore` record and recover the list of users and their ranks to and from NVRAM, respectively. Information for up to 30 users may be saved this way. If more than 30 users are known to the system and an attempt is made to save the users to NVRAM, a warning message will be emitted and the list will be truncated to the first thirty users, ordered first by descending rank and then ascending order of addition.

Access

Local access control

Local access determines who may use the TTY menu and the touchscreen, among other things. It comes in four levels: public (anyone), group (all authorized users + the unit's group), users (only authorized users), and private (owners and managers only.) This can be configured from the **manage > policies > access menu**, or remotely with the `access local <level>` command, in which case 'public' and 'private' access are referred to as the levels 'on' and 'off'.

Other perks of local access (aside from menu usage) include **local command access** on channel /1, automatic trust through the RLV relay and navigation nodes, and automatic trust for

communicating with foreign peripherals such as powered doors or maintenance tables. Local command access is limited to chat range (20 m) and local menu access is limited to 10 m, but RLV devices, navigation nodes, and foreign peripherals may operate at any distance.

Remote access control

This is analogous to local access (above), and affects who may use remote console access (see (Remote access)) to control the unit. The console command for managing remote access is:

```
access local <level>
```

Note that remote console access can also include menus, but these are tracked separately and do not interfere with usage of the touch screen or local TTY menu session.

Self-access control

Self-access allows the unit to interact with its systems. When self-access is disabled, the unit is unable to use @ commands, remote access, or local access, even if it is self-owned. Self-access can be controlled through either the **self** option on the **manage > policies > access** menu, or the `access self <state>` command, where <state> is 'on' or 'off'.

Note: In the event that self-access is unintentionally disabled, the command `@safeword` will allow the unit to regain basic control over its systems. This command can only be used by the unit itself.

Group access

Both local and remote access can be set to allow *group access* mode. This means that anyone currently wearing a tag from the same group as the unit will automatically be granted the equivalent of a basic user (rank 0), except that they will not be included in the **users-only radio policy**. If you would like to specifically designate a large number of individuals for system access regardless of the unit's active group, see the chapter on **Domains**.

Locking

PIN-based locking prevents local access by all users (including the owner) until the correct PIN is entered on the touchscreen or over the TTY menu interface. The PIN can be set in the **manage > policies > access > set new PIN** menu. To lock the unit, press **lock** on the **manage** screen, or execute the `lock` command.

Cancelling PIN changes: Press the **clear** button to abort setting a new PIN.

Default value: The default PIN is an empty string. If you accidentally lock the unit, simply press **enter**.

Autolock: The unit can be configured to automatically lock itself after a certain period of inactivity, using either the **autolock** item on the **manage > policies > access** menu, or the **autolock** command. The **autolock** command can also be used to set the timeout interval, which defaults to 30 seconds.

User role summary

level	name	abilities
	guests	can access non-management controls if public access is enabled and consent is granted by the unit
0	regular users	can access non-management controls if user access is enabled, consent not required; can exchange IMs if radio policy is set to 'users only'
0	group users	as above, but only if group access is enabled; not included in 'users only' radio policy
1	manager	can access all controls if user access is enabled, but cannot affect owner accounts; can exchange IMs if radio policy is set to 'users only'
2	owner	full control, teleport requests are automatically granted; can exchange IMs if radio policy is set to 'users only' or 'owners only', receive distress beacon messages, receive notices of keychain resets and safewording

Volume control and sound schemes

Your unit creates four types of sound: obligatory ambient sounds caused by its operation (magneto-optical disc seeking, fan activity, battery hatch operation, etc.), chimes caused by the interface, tones (either beeps or vocalizations) generated when communicating, and preset voice notifications generated when certain events occur. These are entirely separate from the unit's normal communicative and locomotive vocalizations, although voice notifications will normally be routed through speaker systems built into the unit's chassis rather than its main controller.

Of the four types of sounds, the latter three (voice notifications, tones, and chimes) can be attenuated, as they are produced through speakers on the unit. Affectations such as giggling or sighing produced when the unit speaks are considered tones.

The primary commands for manipulating sound output are `volume` and `scheme`; these have no direct menu equivalents (apart from `volume cycle`, which handles the volume adjustments in the subsystem menu.)

For more detailed explanation beyond what is offered here, see **scheme** and **volume** in the Command Reference.

Voice notifications

In Companion, there are 48 different voice messages which the unit can produce. These are specified in configuration files prefixed with `v_` which can be found in the audio processor module. (See **Installing data files** for information on managing documents of this type.) Instructions for creating new voice notification packs can be found in the Companion SDK.

To see a list of the available voices, execute `scheme voice` on the unit. To select a voice, execute `scheme voice <name>`, where `<name>` is the desired voice pack.

Chimes

To control the volume of the chimes independently from voice notifications, type the following remote console command: `volume toggle chime`

This will silence (or unsilence) audio cues from the system other than fan, drive access, and voice.

Chime schemes affect startup and shutdown sounds. They are built into the unit directly, and can be selected using the following command: `scheme chime <number>`

Menu sounds

Similarly, to control the volume of the menu independently from voice notifications, type the following remote console command: `volume toggle menu`

Menu schemes affect access granted and access denied sounds. They are built into the unit directly, and can be selected using the following command: `scheme menu <number>`

Tones

To control sounds produced in response to speech, execute: `volume toggle tone`

The actual sounds used for tones are specified within personas, and vary on a per-gender basis. See chapters on **Identity options** and **Personas** for more information.

Fan control

The fan speed can be set with the cortex command `!fan <level>` by the unit itself. `<level>` can be any number from 0 to 100, with lower values yielding a lower fan speed. The unit is barely audible at levels below 5%. There are also several text level shortcuts: `off`, `idle`, `low`, `med`, `high`, and `max`, corresponding to 0%, 14%, 28%, 43%, 70%, and 100%, respectively. On systems that use internal liquid cooling, such as the DAX/2 mini and NS-476 Aegis, the `!fan` command instead controls pump rate, although it may have no obvious effect as these systems are frequently silent. Cooling rate control is automatic when **ATOS/E security enhancements** are installed.

Warning: Lowering the unit's fan speed during power-intensive operations may cause overheating.

Input and commands

Depending on your unit's configuration, it may or may not be able to access some or all of its settings, or even access its own control panel (although this may be hard for it to do without a mirror.)

Cortex (bang) commands

These are a special class of commands which are prefixed with the "!" character, which allow the unit to express itself more flexibly. They include:

- **!greet**, **!greet2**, **!love**, **!love2**, **!love3**, **!love4**, and **!bye**
Various verbal emotive expressions.
- **!broken** and **!fixed**
Simulates uneven power to illumination elements. May cause sparking when used with certain peripherals or batteries.
- **!working** and **!done**
Displays the 'working' light pattern via ornamental status light elements.
- **!fan <level>**
Adjusts the fan speed. `<level>` can be a number from 0 to 100, or one of the preset keywords: `off`, `idle`, `low`, `med`, `med`, `high`, or `max`.
- **!spark**
Simulates a mild electrical fault.
- **!fault**
Simulates a serious electrical fault.
- **!release**
Allows the unit to bypass the vocoder pipeline entirely for diagnostic purposes; see below. See also **troubleshooting guide**.

- **!lamp on|off** (requires DAX/3, SXD 8.4, or other new flicker bus firmware)
Controls the intensity of illumination emitted by the controller's lighting elements.
- **!repair, !repaired**
Starts or stops simulated repair effects.

These commands can be issued directly by the unit directly into its normal speech output channel.

Cortex bypass commands

These are a class of message shortcuts which are prefixed with the "." character. When the unit's "mind" subsystem is powered down, it is unable to speak normally and must use these messages to communicate. Each mnemonic is translated into a full message by the persona layer; different personas will change the preset message the unit speaks. See **Personas**. As of version 8.2, personas can customize the list of available bypass commands, so checking ".info" is recommended before attempting to use a bypass command under a new personality.

bypass command	default message
.y	Yes.
.n	No.
.hi	Hello!
.bye	Goodbye.
.ok	Acknowledged.
.lol	Humor detected.
.cannot	Cannot comply.
.error	Error.
.fuck me	This unit is available for use.
.fuck you	This unit offers itself for use.
.dance	This unit is capable of dancing.
.help	This unit requires assistance.
.thanks	This unit is grateful.
.explain	Further explanation is required.
.pickup	Do you require service?
.mind	This unit cannot comply with the MIND module disabled.

The vocoder pipeline

See **Speech modification with vox**.

Released speech

Under certain circumstances it may be necessary to completely release the unit from using the vocoder system, e.g. to interface with devices that require clear voice commands for

identification purposes. The unit may release its speech at any time by saying `!release`, provided it has access to its own control panel. The unit can recapture their chat by either speaking on the redirect channel (which is last three digits of the unit's serial number, plus 100 if the serial would have leading zeroes) or simply speaking `/!capture`. These instructions are also automatically provided when the `!release` command is issued.

Self-access and self-commanding

Self-access is the ability of a unit to adjust its own settings. Like a person, the unit may be added to its own user list and even serve as a manager or owner, although the utility of such access is limited and is more likely to confuse the unit than to allow it to act more independently.

When a unit is first installed, self-access is enabled with maximum permissiveness. You must instruct the unit to designate you as its owner to restrict this. See **User management and access control** for information on adding and promoting users, and restricting self-access. A unit with this access may use its own control panel or teletype interface, or it may access the console directly by speaking console commands into its normal speech output channel with `@` prepended, e.g.: `@power status`

Ad-hoc volume control

Normal, expressive messages can be prefixed with a number of prefixes in order to modify their effects.

prefix	effect
<code>w</code>	whisper (10% volume)
<code>s</code>	shout (100% volume)
<code>d</code>	description
<code>/me</code>	self-description
<code>:</code>	self-description (as <code>/me</code>)
<code>ooc</code>	parenthetical ("out-of-character") remark

These prefixes can be combined, e.g. `"w ooc /me waves."` whispers a parenthetical self-description of the unit waving.

Local command access

New in 8.4 (milestone 5) is the ability to issue commands locally, a feature familiar to users of the popular OpenCollar property control system. This makes it convenient to control a unit without accessing its menus or using a remote control device (see **Remote access.**)

To issue a local command, type: `/!<prefix><command>`

For example, `/1rhreboot` would reboot a unit with the prefix "rh". By default, the prefix string is automatically determined from the first two characters of the unit's name, but it can be changed to any non-null string with the command `commands prefix <new prefix>`. Typing `commands prefix default` will restore the unit to automatically extracting its prefix from the unit's name.

To facilitate configuration, local command access is enabled by default. This utilizes the same permissions model as local menu access (see **User management and access control**), so unrecognized accessors will need to request the unit's consent to issue such commands. To control local command access, use the `commands local` command, e.g. `commands local on` or `commands local off`.

If the unit's prefix starts with a number, you can separate the channel, prefix, and command with spaces, e.g. `/1 <prefix> <command>`.

Remote access

Remote control of a unit is possible from anywhere in the same region through the use of the remote console, available for purchase at major NS outlet stores. The remote console makes it easy to check the status of your unit, send commands, and operate menus, as well as scanning for nearby units and broadcasting announcements to them.



Pictured above: The Companion 8 remote console shown in its 'disconnected' and 'connected' states (top and bottom, respectively.)

Connecting to a unit

To connect to a unit, press the **connect...** button. After 4 seconds, a list of available units will appear. Note that the console will be otherwise inoperable while scanning.

Scanning nearby units without connecting

To view the nearby units, press the **scan...** button. The scanner will be active for 4 seconds. Note that functions on the console other than menu and terminal input will be inactive during this time. The scan output will include version numbers, which are of importance: only units running version 8.0.5 and newer can be controlled using the remote management console. Older units will only support public announcements (if chorus mode is enabled.)

Sending a public announcement

Press the **broadcast...** button, select a range, and enter a message (up to 250 characters.) Messages broadcast with unlimited range will be sent through all units in the region.

Issuing a command to a single unit

Press the **command...** button and enter the command you wish to execute. To make the unit act or speak, use the `relay <message>` command. For a list of other commands, see **Command reference**. The unit may not respond at all if the command is rejected for security reasons, so it is advisable to perform a test command (such as `help`) after connecting.

Issuing multiple commands to a single unit

Press the **terminal** button to enter terminal mode. All messages sent locally will be relayed through the remote control as console commands to the unit. You must have RLV enabled to use this feature.

Accessing the menus

After connecting, click the **menu** button. As of Companion 8.3, this will create a separate menu session that will not interfere with simultaneous local console access.

Checking unit status

The **about** button will report on identification information, authorized users, and connected devices at a glance, and is equivalent to the `about` command.

Technical note regarding channels

Units accept remote commands on two channels: band -9999999 and a unique, per-unit band based on the negative value of the digits of the serial number. (For example, a DAX/2 with the serial number 998-23-5311 would use band -998235311.) The uniformity of the former makes it easy to issue commands to several units at once, and the latter's specificity is convenient for targeting an individual unit without knowing specific key information. Respectively, these are known as the *public* and *private* command channels, although their use is limited according to remote access settings, and so neither is intrinsically tied to a certain access level.

The *chorus* functionality, accessed through the 'broadcast' button on the remote console HUD, sends a special command called `say` which is unique to the command channel processor, and is not to be confused with the Arabesque version of `say`, which simply maps to the standard system `relay` command.

Command channels are also used for certain identification messages, `ping` and the ATOS equivalent, `identify`. Documentation for these messages is available in the Companion SDK, and online at the **ATOS protocols page**, respectively.

The heads-up display

The local console that comes with Companion is called Screen Console Manager. Its primary purpose is to provide visual information to the unit about the unit's status through a bar at the bottom of its field of view. It additionally provides icons that the unit can access for adjusting a range of commonly-used features and configuration settings.

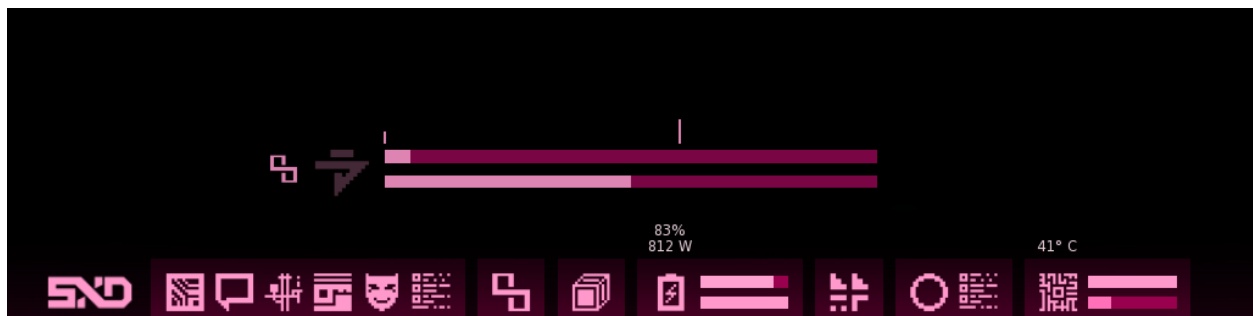
Setup

The command `setup console` transfers a copy of the local console from the controller's system memory into the unit's loadable inventory space, under the directory located at `#RLV/~NS/console/<version>`, where `<version>` is the current edition of Companion, and `~NS` is the RLV directory path prefix specified in the unit's OEM table. A new console will automatically be delivered if the wrong version is attached, and in many cases simply if it cannot be found during system start.

Opting out: Certain situations may make using the local console undesirable. To suppress activation, delete the `_console-screen` object from inside of the `#RLV` directory hierarchy, keeping the empty directories otherwise intact.

Adjustment: To configure SCM, edit the file `_console-config` inside of the controller's system memory. This will automatically be copied into the HUD if it is not identical to the HUD's current configuration, allowing customizations to persist through system updates. The `_console-config` file contains comments describing what each variable within it controls.

Overview



The main body of the SCM display is a bar that appears across the bottom of the unit's field of vision. This begins with a system-dependent identity badge on the left, followed by a series of segments. Each segment represents a device or system component, uniquely identifiable through their icons. These are described in the next section.

SCM may, from time to time, also display 'pop-out' sections for various activities, such as using an Akashic icon device, displaying tutorial information, or relating time-sensitive information from TESI pertaining to arousal levels (as shown above). For the most part, these interfaces are self-explanatory. Additional documentation for more complex interfaces, such as the TESI interface, is included with the products to which they pertain.

Segments

System segment



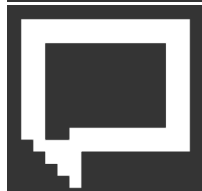
Model badge

This indicates the identity of the controller currently in use. If the model is for some reason unknown, then the "NS" monogram logo will be displayed instead. Clicking the model badge will open the main menu.



Applications

Click this icon to open the **applications** menu.



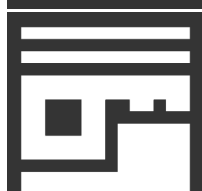
Cortex

Click this icon to open the **cortex** menu.



Devices

Click this icon to open the **devices** menu.



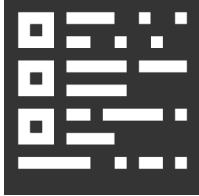
Security

Click this icon to open the **manage** menu.



Personas

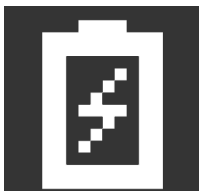
Click this icon to open the **personas** menu.



Subsystems

Click this icon to open the **subsystems** menu.

Device segments



Battery

This segment is visible whenever a battery is installed. Click the icon to access the battery device directly, yielding a report of its current charge level, degradation status for generation 3 and later batteries (if applicable), and safety information. The segment also shows two gauges:

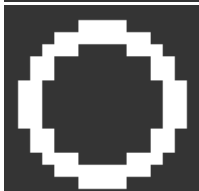
- The **top gauge** indicates remaining battery life. It will turn red when remaining capacity is low.
- The **bottom gauge** indicates current power usage. It will turn white when usage is over 100% of normal (780 W).

Additionally, text will appear above the battery icon indicating percent battery capacity remaining and current power draw. This text turns blue during charging, and red or orange when the remaining charge is low.



Handles

Multiple handle devices are shown condensed into a single icon. This icon will display both follow target (leash) information and carrier information by default. Click it to access the handle device's menu. If the unit has multiple handles, the button will access the first handle device that reported in, which may not necessarily be the handle currently being used to carry it.



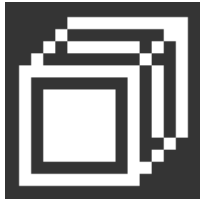
Shield

The shield device shows two icons: a circle, at the left, and the 'properties' icon (also used for the subsystems menu on the system segment) on the right. Clicking the circle icon will immediately erect an active shield barrier, if supported by the shield device. If one is already active, it will be destroyed. Clicking the properties icon will show the shield capacitor's current charge status. These functions correspond to 'poke' and 'peek' respectively, and can be alternatively activated through the devices > shield menu with the 'access' and 'status' buttons.



Collar

If a collar device is installed, it will display follow information over its icon, superseding the handles icon. Click the collar icon to access the leashing menu.



SuperBit

Clicking the icon opens the SuperBit's internal menu.



Akashic icon

Clicking the icon opens a special pop-out section for the icon device. Click the segment again to close it.

ATOS segments



ATOS/E system segment

The system segment shows cooler status, system temperature, and system integrity status over its icon. Cooler speed is measured in RPM (revolutions per minute) for fans or CFM (cubic feet per minute) for liquid cooling. See **Combat, damage, and repair** for more information.

- The **top gauge** shows current system integrity. A full gauge indicates the system has taken no damage.
- The **bottom gauge** shows current system temperature, on a scale from 0° C to 100° C.



ATOS/E weapon segment

The weapon segment is readily distinguished from the system segment because it includes the weapon's name in the text over the icon. Like the system section, it also relates temperature information, but the top gauge indicates clip status (shots remaining/clip size) instead. The weapon segment is only visible when a weapon is drawn. Only one can be active at a time.

Speech modification with vox

Included with your controller is an advanced speech filtering system called **vox**. It is accessible through the `cortex` menu as of Companion 8.2.

Vox supports four stages of filtering, called *semantic*, *linguistic*, *phonetic*, and *typographic*. This distinction is made to ensure that the speech modifiers are applied in the correct order; e.g. a lisp filter (phonetic) should not have to contend with input from a filter that translates the text into binary (typographic).

The categories perform the following functions:

Order	Category	Modifies...	Examples
1	semantic	meaning	word replacement (<code>fs_vocabulary</code>)
2	linguistic	language	automatic translation (<code>fl_polyglot</code>), animal noises (<code>fl_barnyard</code>)
3	phonetic	sounds used	lithping (<code>fp_lisp</code>), sssnake ssspeak (<code>fp_serpentine</code>)
4	typographic	formatting	no _̣ ise (<code>ft_glitch</code>), ^{tiny} font (<code>ft_microvox</code>)

When a filter is activated, it is added to the end of the list for its category, e.g. if `fp_serpentine` and `ft_glitch` are enabled, and the user activates `fp_lisp`, then the final filter pipeline will be `fp_serpentine` → `fp_lisp` → `ft_glitch`. Note that if `fp_lisp` were activated before `fp_serpentine`, then the serpentine filter would have no effect, as all of the sibilant characters it seeks to lengthen (s, z, etc.) would already be replaced with lisped equivalents.

Command-line management

To see a list of supported filters, type: `vox list`. This will produce a listing of all supported speech filters that are registered with vox. If the list is empty but some filters are installed, typing `vox probe` will repopulate it.

To see the filters that are currently enabled, type `vox status`. Filters are enabled with `vox <name>`, e.g. `vox lisp`, and disabled with `vox <name> remove`.

Other commands of the form `vox <name> <parameters>` will be passed directly to the filter. Most filters are coded to recognize `help` as a parameter, and to provide documentation explaining their use when it is provided. Other parameters will generally configure the filter. Any parameters other than `remove` or `help` will cause the filter to be activated if it is not already in use.

Additional vox features

The vox command is also used to manage two system policy settings, the **output pipe** and **speech release**. These can be adjusted by any user with access to the `cortex` menu.

The output pipe

Also known as *gag mode* or *gag compatibility mode*, this configures the unit's vocal model to redirect its audio into one or more attached devices using the RLV protocol. **Note:** most RLV-based gag devices do not work properly with the output pipe system because they refuse to take input from attached hardware such as the controller. Only attempt to use this feature with certified NS-compatible devices.

When the output pipe is enabled, the unit's radio communications are restricted **only** to the voice processor channel. The `send` command can be used to circumvent this limitation.

To enable or disable the output pipe from the command-line, use: `vox pipe <on|off>`

Speech release control

Under most circumstances, the unit may speak the special message `!release` into its voice processor in order to completely bypass the vocoder architecture. This has several benefits, most importantly when interfacing with non-compatible gag devices (see above) and with legacy equipment that requires operation with a spoken command. However, the operator may see fit to prevent this entirely in order to ensure that vox-based speech modifications are obeyed.

To enable or disable `!release` from the command-line, use: `vox release <on|off>`

Understanding the system architecture

Companion consists of the following components, termed modules:

system memory

- afterglow (beacon and auxiliary power)
- ambiance (teleports, sparks, sound settings, volume)
- balance (configuration storage, pipes)
- bonds (RLV management for subsystems)
- coil (charging interface)
- compliance (policy and navigation management)

user memory

- arabesque (executes action scripts)
- restraint (external RLV relay)
- songbird (disk driver, tutorials)
- user applications and data go here*

cortex (communications pipeline, personas) exhibition-core (menu organization) exhibition-{tty, dwm, etc.} (menu presentation) foundation (startup, shutdown, other BIOS functions) hierarchy (guest access, domain interaction) obedience (manual, commands index, misc security) power (subsystem and battery manager) puppet (remote control and device manager) sentinel (ATOS/E security enhancements) submission (user management and authorization) xanadu-client (package/update manager) whip (module hibernation control)	
--	--

Additionally, the controller includes the following FPGAs, which cannot be updated by the software manager:

audio processor echo (audio output, voice schemes)	case controller fan spinner (fan control) flicker (lighting control)	battery door controller hinge (hatch control)
--	---	---

In the Aegis and other systems that must open the battery door to access a holographic display, the hatch controller is also responsible for activating screen projection. For other systems with holographic screens, such as the Yutani Springs XSU and the DAX/3, this is performed by the illumination controller.

Functions by component

This information pertains to the latest development version and may not reflect the architecture of the current release.

- * indicates module can hibernate (no timers or external events)
- ** indicates the module can hibernate when a setting is turned off

MEM	memory card application(s)	(various)
	extended functionality installed by the user	
NSM	network security manager	_hierarchy**
	reconfigures the stored keychain to match the remote server settings	
	polls the remote server for current settings	
	triggers reloading of local keychain to match the remote server	

manages guest access and consent

IRR internal restraint interface

_compliance*

conveys blocked subsystem settings to _restraint and _power

@vox command and menu interface

@policy

waypoint navigation and @navigate

@follow leashing

ERR external restraint interface

_restraint**

RLV relay

conveys blocked RLV flags to _bonds for final integration

CB core services bus

_bonds

manages RLV consequences of subsystem states

integrates environmental RLV flags from _restraint

interference

handles remote commands

A effects library

_ambiance*

manages teleport and spark effects

manages most sound settings

CI charging interface

_coil

interfaces with ACS and UMD chargers

ddt diagnostics programs

manages ACS remote hosting and interference

CR device manager

_puppet

manages light bus interactions

CX cortex

_cortex

personas

voice pipeline

DM display manager

_exhibition-core*

loading and maintaining core menu state

DMD display manager drivers

_exhibition-{tty, dwm, etc.}

presenting menus to user

MC BIOS

_foundation*

parsing OEM table

startup/shutdown sequences

actual on/off state of system

list of supported commands

power profiles

HV2	memory card driver	_songbird
	executing deletion manifests for packages probing for installed packages probing for installed personalities hibernating/thawing modules in memory card	
HV	hypervisor	_xanadu-client*
	package management	
EPS	emergency power system	_afterglow
	maintain auxiliary power capacitor charge maintain auxiliary power system state emergency beacon	
P	performer	_arabesque*
	@color execution of batch scripts	
SM	security manager	_submission*
	keychain facilities and user interface access control and user interface	
SE	security enhancements <i>(optional)</i>	_sentinel
	damage management heat management IFF protocol compatibility weapon management repair interface malfunctions	
AM	access manager	_obedience*
	PIN-based locking identity settings @help documentation facility gender @bolts	
SSM	subsystem manager	_power
	subsystem states battery draining and current power level device draws	
TESI	tactile excitation sensor interface <i>(optional)</i>	_emotion
	tactile surface devices arousal model cryolubricant management	

TS kernel task scheduler

_whip

hibernating modules not currently in use
starting modules required for tasks
storing events for triggering later
interface for hard-toggling optional modules (CI, CX, EPS, ERR, etc.)

CM configuration manager

_balance

storing settings for libraries
shell pipes

In the event that one of these modules misbehaves, it may be possible to fix the issue by resetting the module through the **manage > modules** menu. Be aware that resetting modules often causes information and configuration loss.

Packages

Every standard software package installed into user memory includes a package manifest, which can be recognized by the tilde (~) prefixed onto the package's exact name. This manifest defines the files that the package is responsible for. When a package is uninstalled, the disk driver (*songbird*) will use this manifest file to determine what to remove. Packages are completely removed prior to update.

User software, data, and other add-ons

Companion is designed to be highly extensible, and many first- and third-party add-ons exist for the platform. There are four major categories of user content that can be installed directly onto a controller:

- **Applications.** These are native LSL programs that reside in user memory and can perform any function. They are generally accessed through the **applications** menu on the main screen.
- **Personas.** These are configuration presets that alter how the unit sounds, speaks, and behaves. They can be found through the **personas** menu on the main screen.
- **Arabesque scripts.** These perform a set list of system commands, similar to shell or batch scripts on other operating systems. They also have limited support for branching and variable substitution. Arabesque scripts can trigger in response to system events (such as disconnecting from a charger), persona changes, or direct user request from the **perform** menu on the main screen.
- **System extensions.** These are optional operating system components that reside in system memory, including the TESI emotion simulator and ATOS/E security enhancements. Depending on the unit's individual configuration and the nature of the specific extension, access methods may vary.

Getting add-ons. While it is possible to manually install add-ons (see **Installing data files** for instructions), most content is made available in the form of a software package using the Xanadu Package System protocol. To see available options, use the **manage > software** menu or the `xanadu` command. Further details are available [here](#).

Removing system extensions. In Companion 8.3 and 8.4, system extensions such as TESI and ATOS/E can only be removed by installing a special removal package, even though these extensions may appear under the list of removable software. See the **generic command-line update instructions page** for more information.

Applications

Starting with SXD System 7.2, your unit supports custom user applications written in native LSL. These can be accessed from the applications menu via the display screen or the teletype interface. Applications may or may not provide commands accessible via the console; see **Remote access** for more information. Examples of applications from Nanite Systems include:

name	function
instructor	Improves unit compliance with a provided rule-set.
announce	Provides user access to the chorus broadcast interface throughout the

region.

- fl_vocabulary**** Limits the unit's speech to select words, or forbids the use of certain words.
- fl_polyglot**** Automatically translates the unit's speech into another language.
- spring** Performs actions and triggers reminders at specified dates and times.
- consensus*** Shares input data with nearby units (if similarly configured) to improve the clarity of decision making.

* Not yet available.

** Included by default as part of the vox system as of 8.2. See **Speech modification with vox**.

Important: Many third-party user applications exist, and while Nanite Systems encourages and supports experimentation with these creations, owners should be aware of the risks of installing software that has not been reviewed for safety and stability by certified programmers. Before installing software on your robot, it is important to ensure that you completely trust the software provider. Nanite Systems cannot guarantee the safety of programs acquired from third-party sources, nor provide support for them.

Creating new applications

To get started, look for the Companion 8 SDK at your NS retailer for information and sample code for developing custom applications. The standard distribution of Companion includes an application called 'Hello World' which you may find useful as a template. For more information, visit our online developer information portal, **develop.ns**.

Installing data files

Certain operations, such as adding voice packs, editing personas, or modifying the OEM table may require direct manipulation of the files within the controller. This is typically accomplished with an Edit Tool, manufactured by Linden Research, or a compatible device.

Caution: The Edit Tool is very powerful and its misuse may severely damage your controller. Your warranty may not cover all possible errors caused by malfunctions or inadequate training when using the Edit Tool. Basic documentation for the Edit Tool and other Linden Research Build Tools is **available online**.

Note: The controller must have its safety bolts unlocked to edit its contents. These bolts can be unlocked with the terminal command `bolts unlock` (or `@bolts unlock` from the local command-line). By default, bolts will also unlock when the unit is powered down. In some cases, the controller may need to be disconnected from the unit to install new files (e.g. if they are non-transferrable.)

The table below lists the link numbers for the most popular 8.x platforms, along with information on how to locate the components in question. Link numbers may vary in other versions. Remember to enable Edit Linked Parts mode on your Edit Tool when manipulating the controller or you will only be able to access the contents of system memory. If you are using the Firestorm variant of the Linden Research viewer, then you can use the hotkeys ctrl-comma and ctrl-period to cycle through the controller's elements when Edit Linked Parts mode is enabled.

System	case controller	system memory	user memory	audio controller	battery door
DAX/2	48 fanblade	1 screen frame	18 above battery	11 below screen	59
DAX/2m	9 top light ring	1 frame	12 below frame	3 below screen	5
SXD and SXDjr	53 fanblade	1 behind gauges	68 above battery	11 above left buttons	51
NS-112	2 inner battery socket	1 body	4 behind audio controller	6 below hatch	5
NS-476	20 above battery	1 body	23 below battery	19 above battery	26
NS-115	9 above battery	1 body	7 above battery	8 above battery	5
NS-409	7 above battery, inside case	1 base ring	22 above battery	8 in front of case controller	26
DAX/3	22 screen projection cone	1 main body	19 behind projector	20 behind user memory	21
DAX/3m	3 battery door	1 main body	5 behind left panel	4 above battery socket	3

See **Understanding the system software architecture** for more information on the functions and contents of each component.

For third-party controllers, such as the Yutani Springs XSU and Barthes Asset Control cranial drive, part numbers can be found by searching for certain code modules and part descriptions; user memory parts are commonly accompanied by a "program" descriptor, and audio controllers have the "speaker" descriptor. Battery door (containing "hinge" programs) and case controller parts (containing "fan" programs) may vary, or even be absent for certain models.

Managing installed software

Version 8 of Companion introduced a complete overhaul of the software management system called **Xanadu**. This permits automated updating of the installed firmware, as well as the installation, updating and removal of packages from a central server using a menu-based interface. **Packages** may include applications, scripts, personas, or data for other user applications that you already have installed. Only managers and owners may use the software manager.

Locating a server. When installing or updating software for your controller, it is necessary to be in the same region (within 256 m on the ground, altitude-independent) as a Xanadu server. Xanadu servers can generally be found on the premises of Nanite Systems offices and retailers, although some organizations may offer their own Xanadu servers for custom applications, drivers, and compatibility software. A list of locations with official Xanadu server support for system updates and similar is available at the **Store and Service Locations** page.

Menu usage

Connecting to a server. Once you have located a region you believe has server support, open the **manage > software > connect** menu and select an appropriate server from the list. Names and descriptions for detected servers will be provided to you in text.

Installing packages. Once connected to a server, select **install**. After a few seconds you will be presented with a range of packages. To learn more about a package, select its name from the list and choose **info**. To install the package, select its name and choose **install**.

Updating packages. This is analogous to the installation process: select **update** from the server's menu, choose a package, and then press **upgrade**. The menu will only list packages which you have installed that the server can update. To learn about what the update contains and to remind yourself about the purpose of the package, press **info**.

Removing packages. Select **remove** from the **manage > software** menu (or the server's menu) and then the package name, followed by **remove**. To learn what the package contains instead of removing it, you may alternatively press **info**.

Updating the system. Follow the instructions to update a normal package, as described above. Select the "**Companion**" package. For more details, see: **Updating to the latest firmware**. It is strongly recommended that you first back up your keychain with the **manage > users > save users** item, or the `keychain save` command.

Command-line usage

The `xanadu` command, introduced in Companion 8.3.0, allows direct and efficient control of the package management system, and is ideal for power users and domain administrators. Its full range of options is as follows:

```
xanadu list|servers|search|disconnect
xanadu connect <server>
xanadu install|info|upgrade|remove <package>
```

- `search`: polls for available Xanadu package servers. Performed automatically on region change.
- `servers`: lists available servers.
- `connect`: connects to `<server>`.

- `disconnect`: disconnects from the current server.
- `list`: lists installed packages. If connected, combines this list with a list of packages on the server.
- `install`: requests `<package>` from the server and attempts to install it.
- `info`: gives info notecard (readme file) for `<package>`. This works for any package listed by `xanadu list`.
- `upgrade`: installs `<package>` from the server. This guarantees the current version of `<package>` is removed first.
- `remove`: removes `<package>` from the unit. A server connection is not required.

Exact names (including versions) are required for installing, upgrading, and removing.

Manual installation

Certain types of assets such as voice packs, custom personas, or software not yet available on a package server may require direct editing to install, update, or remove. See **Installing data files** for instructions on how to manipulate data files.

Updating to the latest firmware

Updating the system to the latest version can be accomplished in several ways:

- (1) If you received a **(redeliver)** monad with your controller, wearing this will automatically deliver a fresh copy of the newest system.
- (2) Using the console, visit a region that supports Xanadu service (e.g. Eisa, NS's home) and type:
 - i. `xanadu servers` (to get a list of available servers)
 - ii. `xanadu connect <server>` (to connect to a server offering system updates, e.g. `xcentral:0`)
 - iii. `xanadu list` (to get a list of packages both on your system and on the server)
 - iv. `xanadu upgrade Companion_x.x.x` (where `x.x.x` is the version number of the latest system release)
- (3) Using the menu, visit a region that supports Xanadu service (e.g. Eisa, NS's home) and select:

```
manage > software > connect > (server) > update > Companion_x.x.x > upgrade
```

Where `(server)` is the name of a server offering system updates, e.g. `xcentral:0`.

It is advisable to back up your user list before updating. Run the following command first:
`keychain save`

(Remember to put `@` signs before each command if you are operating the unit locally.)

Upon successful updating, the unit will receive a notecard containing patch notes and any necessary instructions to finish the installation.

For more information about the software management system, see `xanadu` in the **Command Reference**.

Personas

The Personas system allows for rapid modification of the unit's personality to suit the user's taste and whims. This is accomplished in software by providing a powerful reward for the unit's cortex to act in the desired manner, which ensures that the unit will comply whether or not any legacy base personality would otherwise enjoy the persona mode. This also means that personas share memories, a critical safety feature.

What personas can do

Personas are able to modify the following:

- The unit's habitual kinematics and countenance (via the RLV standard.)
- The tone of the unit's voice.
- Preset messages when the mind subsystem is disabled.

Additionally, personas are capable of compelling the unit to carry out a range of tasks such as changing illumination color, speaking, and performing actions when activated. See **Scripting actions with Arabesque** for more information on scripting actions with the Arabesque executive.

Creating new personas

A persona file is a three-line-minimum text file where each line contains a different key attribute of the persona. An example is shown below.

```
default
45a8a032-2d5a-3115-8bde-bdb0e3fab61c sxd-moan b3e1418d-af3a-9d9a-ae52-73c0f0bcd7ae
y:=Yes.
n:=No.
hi:=Hello!
bye:=Goodbye.
ok:=Acknowledged.
lol:=Humor detected.
cannot:=Cannot comply.
error:=Error.
fuck me:=This unit is available for use.
fuck you:=This unit offers itself for use.
dance:=This unit is capable of dancing.
help:=This unit requires assistance.
```

```
thanks:=This unit is grateful.
explain:=Further explanation is required.
pickup:=Do you require service?
mind:=This unit cannot comply with the MIND module disabled.
```

The **first line** describes the RLV standard directory containing the schematics for the alterations to make to the unit's behavior and appearance. The controller will look under the directory "~NS" for these schematics within your unit's #RLV hierarchy, e.g. **#RLV/~NS/default**. These are overlaid on the unit's existing configuration when loaded and removed when no longer required.

Note: The exact path is determined by the **path** field in the `_oem` file, which defaults to "~NS". See **OEM data files** for more information on the `_oem` file.

The **second line** describes the tone of voice to use when speaking, referred to as "tone markers". Multiple tone markers - up to three - can be specified, separated by spaces, according to the following rules:

- If **one** marker is specified, it will be set as the **female** voice gender. Neuter and male voices will use hardwired system defaults.
- If **two** markers are specified, the first sound is for both **neuter and male**, and the second sound is used for **female**.
- If **three** markers are specified, they are used for **neuter, female, and male** respectively.

A number of tone markers are stored in the unit's audio processor by default:

```
sxd-cocky, sxd-compliant, sxd-curious, sxd-dismissive, sxd-furious,
sxd-growl, sxd-hmm, sxd-laugh, sxd-marquise, sxd-mmm, sxd-moan, sxd-
moo, sxd-pythia, sxd-sass, sxd-terse
```

You are free to install additional tone marker samples into the audio processor, or, if you would like the assets to be fetched directly as needed, provide the UUID here of a speech profile instead. However, these may not be available to all users. The NS-476 Aegis comes with some custom tone markers that are not available elsewhere; take care to avoid depending on these in custom content.

The **third and successive** lines specify the preset messages that the unit may use when the mind is disabled. A list of messages set by the current persona can be accessed by typing `.info`

Each preset message can be specified on a new line, or they can be separated by a pipe ("|"), as below:

```
yes:=Yes.|no:=No.|hi:=Hello!|bye:=Goodbye.|ok:=Acknowledged.
lol:=Humor detected.|cannot := Cannot comply.|error := Error.
use me:=This unit is available for use.|use me?:=This unit offers itself for use.
dance?:=This unit is capable of dancing.|help:=This unit requires assistance.
thanks:=This unit is grateful.|explain:=Further explanation is required.
```

```
need help?:=Do you require service?
mind:=This unit cannot comply while the MIND subsystem is disabled.
```

Due to technical limitations only 255 characters can be read from a given line, but loading configuration files with many lines takes longer. Use pipes whenever possible to keep your personas compact and efficient.

Gendered pronouns in preset messages

Depending on how the unit is utilized, there may be a need for the pronouns configured in its preset messages to change on the fly. This can be accomplished by using placeholders in the messages where the pronouns would normally go.

gender	absolute	possessive	subject	object	reflective
inanimate	its	its	it	it	itself
female	hers	her	she	her	herself
male	his	his	he	him	himself
neuter	theirs	their	they	them	themselves
Persona (mental)	\$m_abs\$	\$m_pos\$	\$m_subj\$	\$m_obj\$	\$m_refl\$
Persona (physical)	\$p_abs\$	\$p_pos\$	\$p_subj\$	\$p_obj\$	\$p_refl\$
Arabesque (mental)	\$m_abs	\$m_pos	\$m_subj	\$m_obj	\$m_refl
Arabesque (physical)	\$p_abs	\$p_pos	\$p_subj	\$p_obj	\$p_refl

Persona refers to the placeholder's use in preset messages, and **Arabesque** refers to the placeholder's use in Arabesque scripts (see **Scripting actions with Arabesque**). **Physical** and **mental** are applied based on the unit's set physical and mental gender, respectively. See **Identity settings** for information about setting the unit's gender.

Installing personas

The primary file of your persona should be named `p_<persona>`, where `<persona>` is the name you want to appear on the command menu. It does not necessarily have to match the first line of the file. Should you wish to use Arabesque scripting with your persona (see **Scripting actions with Arabesque**), name the corresponding script file `px_<persona>`.

To install these files, they must be placed in the unit's user memory. See **Installing data files** for more information.

To activate your unit's new persona:

- Via the remote console, type the following: `persona <persona>`
- Via the display screen or teletype interface, select the persona name from the personas menu. After installing the persona, it may be necessary to refresh the menu before it will appear.

Deprecated Preset Messages Format

Older versions of the SXD firmware (Pre-8.0) do not support the `<command>:=<message>` format. Instead, they use a "blob" format, shown below.

```
Yes.|No.|Hello!|Goodbye.|Acknowledged.|Humor detected.  
Cannot comply.|Error.|This unit is available for use.  
This unit offers itself for use.|This unit is capable of dancing.  
This unit requires assistance.|This unit is grateful.  
Further explanation is required.|Do you require service?  
This unit cannot comply with the MIND module disabled.
```

These messages are parsed in a fixed order, and can be accessed by the unit (or a remote user relaying speech commands) with the following mnemonics:

```
.y, .n, .hi, .bye, .ok, .lol, .cannot, .error, .fuck me, .fuck  
you, .dance, .help, .thanks, .explain, .pickup, .mind
```

The preferred separator for these messages is the vertical bar or "pipe" |, but due to technical limitations only 255 characters can be read from a given line, so line-breaks may be substituted instead.

Scripting actions with Arabesque

The Arabesque scripting system allows the execution of macros specified by the user. These scripts may carry out any function on the unit that does not require administrative access, as well as play sounds, start and stop pre-recorded animation programs, and command the unit to say or act in a certain way. Each line of the script specifies an action to execute. Be aware that the unit can only run one script at a time, and that additional power is expended in the process of executing these scripts.

Commands

In addition to the commands available to the user via remote access (see **Remote access**), Arabesque scripts may contain the following verbs:

command	action
<code>start <animation></code>	This begins the specified animation, which must be included in the writeable memory of the unit alongside the script.

<code>stop <animation></code>	This stops the specified animation.
<code>sound <sound></code>	This plays the specified sound, which must either be loaded onto the audio processor or referred to directly by UUID.
<code>sound vox <sound></code>	This plays one of the messages from the included voice notification pack. See Sound for more information on voice notifications.
<code>remark <message></code>	This reports a message privately to the unit.
<code>say <message></code>	This causes the unit to say something. The message may contain speech-processor-level commands such as bang commands, dot commands, or emotes. (See articles on input and commands and speech modification with vox .) This will operate even if the unit's mind is disabled (i.e., even if the cortex prohibits free speech), since it is not spontaneous speech generated by the unit.
<code>wait <duration></code>	Pauses execution for <duration> seconds.
<code>disable <subsystem></code>	Disables subsystem number <subsystem>.†

`enable <subsystem>` Enables subsystem number `<subsystem>`.†

Subsystem numbers are as follows:

subsystem	number
video	0
audio	1
move	2
teleport (FTL)	3
rapid (run + jump)	4
voice	5
mind	6
preamplifier (speak above a whisper)	7
power amplifier (shout)	8
receiver (incoming SMS)	9
transmitter (outgoing SMS)	10
GPS	11
identify	12

`set <variable> <value>` Sets the indicated variable `<variable>` to the specified value `<value>`. Variables must be prefixed with `$`, `%`, or `@` indicating text, integer number, or real number datatype, respectively.

`randset <variable> <max>` Sets the indicated variable `<variable>` to a random integer between 0 and `<value> - 1`. The variable must be an integer.

`unset <type> <variable>` Deletes the specified variable from memory. `<type>` must be `int`, `integer`, `str`, `string`, or `float`.

`report <variable>` Causes the unit to speak the name and value of the specified variable.

`ifeq <value_1> <value_2> <expression>` Executes the specified expression (a complete line of code, possibly including more of these keywords) if `<value_1>` and `<value_2>` are equal. The first value must be an integer variable with its `%` prefix removed; the second value may be either an integer literal or an integer variable.

`ifexists <filename> <expression>` Executes the specified expression (as above) if a file called `<filename>` exists in user memory.

† Use of `enable` and `disable` is discouraged as these numbers were different in previous versions of Companion and may change again in the future. If possible, use the `profile` system command instead.

To include a comment in an Arabesque script, start the line with a hash sign (#).

Variables

Like other system scripting languages, Arabesque supports environment variables, which can be substituted into text as needed. As of Companion 8.4.4, the following variables are automatically defined:

variable	type	description
<code>\$user</code>	string	The key of the avatar who activated the Arabesque script.
<code>\$username</code>	string	The name of the avatar who activated the Arabesque script.
<code>\$color</code>	string	The current system color in space-separated floating point (e.g. "1.0 0.5 0.0" for orange—see chapter on identity options)
<code>\$name</code>	string	The unit's current callsign, e.g. "vict0r"
<code>\$model</code>	string	The unit's model, e.g. "NS-304"
<code>%serial</code>	integer	The unit's serial number without punctuation, e.g. 999123456
<code>\$serial_display</code>	string	The unit's serial number with punctuation, e.g. "999-12-3456"
<code>@power</code>	float	The current power level (0.0–1.0)
<code>\$persona</code>	string	The name of the active persona
<code>\$version</code>	string	The system version
<code>\$p_abs</code>	string	Physical absolute pronoun ("theirs")
<code>\$p_pos</code>	string	Physical possessive pronoun ("their")
<code>\$p_subj</code>	string	Physical subject pronoun ("they")
<code>\$p_obj</code>	string	Physical object pronoun ("them")
<code>\$p_refl</code>	string	Physical reflexive pronoun ("itself")
<code>\$p_gender</code>	string	Physical gender ("inanimate")
<code>\$m_abs</code>	string	Mental absolute pronoun ("theirs")
<code>\$m_pos</code>	string	Mental possessive pronoun ("their")
<code>\$m_subj</code>	string	Mental subject pronoun ("they")
<code>\$m_obj</code>	string	Mental object pronoun ("them")
<code>\$m_refl</code>	string	Mental reflexive pronoun ("itself")
<code>\$m_gender</code>	string	Mental gender ("inanimate")

Variable names are parsed in a space-separated manner, so,

```
say /me $username's robot is here!
```

will not work, but,

```
say $username is this unit's best friend.
```

will. To get around this, the `^H` and `H^` control sequences are provided. These eliminate spaces before and after the tokens, respectively, and are parsed in a separate pass. Thus,

```
say Is your name really "H^ $username ^H?"
```

will parse correctly.

If you need to insert `$`, `@`, or `%` at the start of a word and not have that word parsed as a variable, double it, e.g. by writing `$$`, `@@`, or `%%`. This is not necessary in the middle of a word, due to the behavior outlined in the previous paragraph.

Script types

Normal Arabesque scripts use the filename prefix `"a_"`, and can be activated from the `perform` menu, or via the `do` command. There are two variant types of Arabesque scripts that do not begin with `a_`. These are `e_` and `px_` scripts. `e_` scripts are executed automatically in response to certain system events, such as connecting to a charger or turning on the system. `px_` scripts are executed when a **persona** is activated; i.e., `px_default` corresponds to the `default` persona.

The supported `e_` scripts are:

- `e_charge-start`: Triggered when charging begins.
- `e_charge-end`: Triggered when charging ends.
- `e_boot`: Triggered when the system is powered on. (Formerly known as `_init`.) Not triggered by activation of auxiliary power.
- `e_shutdown`: Triggered when the system is shut down using the `off` or `shutdown -h` commands, or through the main menu, regardless of auxiliary power settings.

For testing purposes, `e_` event scripts can also be activated using the `trigger` command.

Limitations

Only one script can be running at a time. Starting a new script (of any type) will interrupt any currently running script. This can be useful if you want to stop a long script, and indeed the included `a_stop` file can be used to do this. However, be warned that in Companion 8.4 there is no animation stack, so animations will continue playing afterwards unless explicitly stopped.

Arabesque has very limited memory and little in the way of branching facilities. As a result, you may find it difficult to accomplish anything significant with it. However, while the language is not Turing-complete due to its inability to emulate the proverbial 'tape,' it is still very useful as an intermediary utility for coordinating messages with other attachments and generating non-branching or minimally branching behavior.

Command Reference

Basic commands: safeword, verbosity, send, reboot, halt, off, profile, shutdown, bootstyle, reset, about, open, preload, sound, remark, trigger, do, color, tutorial, sentinel*, sxdwm, relay, persona, vox, charge, coil, scheme, volume, audience, optics, hover, unleash, range, leash, follow, navigate, device, setup console, power, zap, help, commands, unlock, lock, lust**.

* Requires ATOS/E.

** Requires Emotion Simulator.

Administrator commands: keychain, name, authority, access, rlv, restraint, xanadu, domain, guests, policy, aux, beacon, gender, pin, bolts, autolock, drainprotect, chorus.

General-interest topics: bangs.

If you need help with a command not listed here, try typing "<command>" or "<command> help" to check for embedded documentation.

Syntax used in this manual:

| (pipe) indicates alternatives.

<> (angle brackets) indicate mandatory variables.

[] (square brackets) indicate optional (literal) parameters.

" " (double quotes) indicate text that might be typed.

VOX

```
vox probe|status|pipe <on|off|toggle>|release <on|off|toggle>
```

```
vox list|clear|<filter> [parameters]|<filter> remove|<filter> help
```

Cortex configuration utility.

shutdown

```
shutdown -h|-r|-k <time>|now
```

```
shutdown -c
```

Shuts down the system in the specified number of seconds. If <time> is 0 or "now", no warning is sent.

-h: power off

-r: reboot

-k: fake warning

-c: cancel scheduled shutdown

volume

volume cycle|help|set <[number]|full|mute>

volume mute|unmute|toggle <tone|chime|voice|menu>

Volume settings.

cycle: Switch between whisper, speak, and shout-capable presets.

help: Displays this message.

set: Sets the chime and voice volume to <number>

mute: Sets the mute flag on the specified feature (see below)

unmute: Clears the mute flag on the specified feature (see below)

toggle: Switches the mute flag on the specified feature (see below)

Features (for use with mute/unmute/toggle):

tone: Voice sample played when text emitted

chime: All sound effects from the speaker other than TTS messages

voice: TTS messages

menu: menu beeps only

persona

persona [<name>]

persona probe

Displays the available and active personas, or activates a persona. To disable persona overlays entirely, type "persona default".

probe: Re-scans for available persona files.

To read more about personas in general, see **Personas**.

relay

`relay <message>`

Causes the unit to speak the specified message. This works even if the unit's mind subsystem is disabled. See **Subsystems**.

verbosity

`verbosity cycle|public|internal|none`

`public` (0): played at 10% (whisper) volume, even if speakers (voice subsystem) are muted

`internal` (1): only the unit sees the messages

`none` (2): no messages

hover

`hover active|shutdown <level>`

Sets the hover offset to be used when the system is online or shut down. `<level>` is measured in centimeters. Type "hover" to see the current values.

setup

`setup console`

Installs the Screen Console HUD into the inventory filesystem.

console-screen

The Screen Console HUD is the standard interface overlay for Companion. Introduced in version 8.1, the Console takes the form of a dock bar at the bottom of the screen, with each segment representing a device.

Some units may find that the HUD is positioned incorrectly for their displays; if this is the case, it can be adjusted by editing the `_console-config` notecard inside of the controller's system memory.

The `_console-config` file is automatically re-copied into the HUD whenever it changes, ensuring that settings persist through system updates.

coil

`coil reset|help|status|ddt|unbind`

Charger interface control.

`reset`: resets the charger interface module and releases motor/mind locks.

`help`: this message.

`status`: reports the charger interface module's status.

`ddt`: toggle debug messages.

`unbind`: remove all RLV restraints.

zap

`zap <serial> <amount>`

`zap <amount>`

Charges the unit specified by the indicated serial number with the specified amount of power in kilojoules.

If no serial number is specified, all units in a 10 m radius will be charged.

When ATOS security enhancements are installed, the maximum amount is fixed at 100 kJ to reduce the risk of system damage.

power

`power status`

Reports the system's current power status, indicating the subsystem state, power draw rate, and remaining battery.

`power video|audio|move|teleport|rapid|voice|mind|preamplifier|power-amplifier|receiver|transmitter|GPS|identify`

Toggles the indicated subsystem.

`power motors|radio`

Cycles the indicated metasystem. "`power motors`" switches between `move` and `rapid`, `move` only, and disabling both. "`power radio`" switches between `transmitter` and `receiver`, `receiver`, `transmitter`, and disabling both.

A third metasystem, `volume`, is controlled through the "`volume cycle`" command.

volume

`volume cycle`

`volume set <number>|full|mute`

volume mute|unmute|toggle tone|chime|voice|menu

Sub-commands:

cycle: switch between whisper, speak, and shout-capable presets

set: sets the chime and voice volume to <number>%

mute: sets the mute flag on the specified feature (see below)

unmute: clears the mute flag on the specified feature (see below)

toggle: switches the mute flag on the specified feature (see below)

Features:

tone: voice sample played when text emitted

chime: all sound effects from the speaker other than TTS messages

voice: TTS messages

menu: menu beeps only

scheme

scheme boot|menu <number>

scheme voice <name>

scheme add boot|menu <key1> <key2>

scheme remove boot|menu <number>

Configures the system sound schemes. Note that this does not affect the tone marker (the sound made when speaking); see **Personas** and **Gender** for more information.

boot: Sets the boot chime scheme or reports the number of boot chime schemes.

menu: Sets the boot chime scheme or reports the number of menu chime schemes.

voice: Sets the voice notification theme or reports the supported voice notification themes.

add: Creates a new sound scheme.

remove: Removes a sound scheme.

bootstyle

bootstyle <number>

Determines the amount of information reported while the unit is powering on.

- 0: detailed and slow
- 1: silent and slow
- 2: brief and slow
- 3: silent and slow (default)
- 4: detailed and fast
- 5: silent and fast
- 6: quiet and fast
- 7: silent and fast

profile

`profile [<name>]`

Loads power profile <name>. If no profile is specified, lists available power profiles.

`profile save|delete <name>`

Saves or deletes the power profile <name>. The system may store up to 9 power profiles.

device

`device [<device name> <command>]`

Sends <command> to <device name> over the light bus. If no arguments are specified, lists the attached devices.

about

`about`

Lists the unit's identification information, along with a list of recognized users and active devices.

follow

`follow [<username>]`

Follows <username>. If no user name is specified, then following is toggled, i.e. if the unit is currently following someone, it is canceled, and if the unit is currently following no one, then it will start following the person who issued the command.

See "leash" and "unleash" for more idiomatic synonyms.

color

`color [save] <<r> <g> >|<#rrggbb>|<name>`

```
color apply|save|reset|restore
```

Controls the unit's lighting color. <r> <g> values may be either floating point (0.0-1.0) or 8-bit (0-255).

`apply`: forces the light bus to accept the current color

`save`: store the current color as the permanent color

`reset`: revert current and stored value to the default (`company`)

`restore`: revert to stored color

Additionally, the syntax "`color save <color>`" can be used to save and specify a color at once.

bolts

```
bolts lock|auto|unlock|cycle
```

Sets the integrity level of the safety bolt system.

`lock`: seals the bolts entirely, making removal of the controller impossible

`auto`: seals the bolts while the unit is active

`unlock`: disconnects the bolts entirely (DANGER: Do not remove controller while unit is running!)

`cycle`: switches to the next bolt-locking state

Note that the system's safety bolts automatically lock whenever the unit's console is locked (see "`lock`") including when it is configured to lock automatically (see "`auto1ock`"). This persists when the unit is powered down to prevent component theft.

do

```
do [<script>]
```

Executes a script with Arabesque. If no script name is specified, a list of available scripts is provided. Arabesque scripts reside in user memory, and are identified with the "`a_`" prefix. This prefix is not included in the name when using "`do`".

Arabesque is a small superset of the standard system commands that adds conditional statements and some variables in order to facilitate batch processing. See **Scripting actions with Arabesque** for more information.

See also "`trigger`", an alternative command for starting Arabesque scripts prefixed with "`e_`".

remark

```
remark <message>
```

Sends a message directly to the unit. This is useful for ad hoc RLV commands.

lock

```
lock
```

Locks the console immediately. This will lock the controller's safety bolts temporarily (see "bolts"). Locking prevents use of local access until the PIN has been entered correctly. See also "pin", "autolock", and "unlock".

unlock

```
unlock <pin>
```

Unlocks the console. This may release the controller's safety bolts (see "bolts") if they are configured to be in the unlocked state. See "lock" for more information on console locking.

commands

```
commands
```

Lists all registered commands. Asterisks in the list indicate that access level 1 (manager) or higher is required; ellipses (...) indicate the command has additional parameters.

This list is automatically regenerated on reboot, although installed software may add and remove entries at any time.

```
commands prefix [default|<new prefix>]
```

Changes the prefix used for local commands issued on channel 1. If 'default' is specified, this will be derived from the first two characters of the unit's nominal designator, e.g. 'rh' for 'rhetOrica'.

```
commands local <on|off|toggle>
```

Enables or disables parsing of commands input on channel 1. Commands must be prefixed properly (see above) and the accessor must have local access permissions.

Manager access or higher is required to adjust prefix and local settings.

help

```
help <topic>
```

Displays documentation for the system. For an introduction, type "help" with no additional parameters.

sxdwm

sxdwm connect|menu <name>|disconnect|dump-dynamic-menus|reset|status

Controls the Exhibition display manager. This is responsible for managing all of the system's menus.

chorus

chorus [on|off|toggle]

Controls whether the chorus feature is enabled. Chorus allows operators and other units to anonymously relay messages through the system for dramatic effect or public announcements.

drainprotect

drainprotect [on|off|toggle]

Controls whether or not remote control messages (over the unit's public and private data busses) and ACS-based chargers may drain power.

access

access local|remote [on|off|users|group|cycle]

Determines which users can access the system using each modality:

local: console menu (touch) access, /1 <prefix> local command access, light bus devices, navigation nodes, and RLV relay devices.

remote: access through a remote control or another device that uses the command bus.

on: enabled for everyone; unrecognized users will be prompted if consent is enabled (see "guests")

off: disabled for everyone but owners

users: enabled for all registered users (see "users")

group: as users, but members of the unit's active group will be treated as level 0 users

access self [on|off|toggle]

Controls whether or not the unit can access itself. This setting takes effect immediately. The "safeword" command can be used to cancel it.

authority

```
authority <organization name>
```

```
authority NONE
```

Sets or clears the name of the authority that has control over the unit. This information is displayed by the "about" command, as well as by certain accessories, such as the SuperBit titler.

The authority is overridden upon entry into a domain. See "domain" for more information on domains.

name

```
name <unit name>
```

```
name NONE
```

Sets or clears the unit's nominal designator. Must be valid **ASCII**. If the unit has no name, then the system will default to displaying the serial number instead.

keychain

```
keychain [list]
```

```
keychain reset
```

```
keychain save|load|dump
```

```
keychain rewrite <value>
```

```
keychain remove|promote|demote|add|submit <user key>
```

Controls the system's access control list, i.e. the list of users who are authorized to use the unit and their corresponding security clearances.

`list` (or no arguments): lists the current users and their ranks.

`reset`: clears the user list, including permanent storage, and puts the unit under self-ownership. This can be performed by the unit or by any owner, and sends a notification message to all owners.

`save`: saves the keychain to permanent memory. This can be restored with "keychain load" or by resetting the user manager (see "reset").

`load`: loads the keychain from permanent memory, overwriting the current user list.

`dump`: reports the current saved value of the keychain in permanent memory in raw format.

`rewrite`: replaces the saved value of the keychain without reloading it. (Manufacturer use only.)

`remove`: deletes a user.

`add`: adds a user.

`promote`: raises a user's security clearance by 1 level.

`demote`: lowers a user's security clearance by 1 level.

`submit`: makes a user the unit's sole owner (level 2 clearance). The submit command can only be used by a self-owned unit with no other owners.

Level 1 (manager) access or above is required for: `save`, `load`, `dump`, `rewrite`, `add`.

Equal rank is required for: `promote`, `demote`, `remove`. For example, managers may promote users to manager rank, or demote managers down to user rank. Users may always remove themselves.

The keychain command is completely disabled if the domain has locked the unit's user list. It must be left before the user list command can be used again, which can only be done by a manager or owner.

autolock

```
autolock on|off|toggle
```

```
autolock time <secs>
```

When enabled, the console will automatically lock after a specified period of inactivity. The default time is 30 seconds. See "lock". When autolock is enabled, the safety bolts are also locked while the unit is powered down.

See "bolts" for more information on safety bolts.

pin

```
pin <pin>
```

Sets the PIN used for unlocking the console. The PIN must be a series of numeric digits. It is empty by default. See "lock".

gender

```
gender [physical|mental|voice] inanimate|female|male|neuter|<custom>
```

`physical`: The pronouns used to describe the unit in actions.

`mental`: The pronouns used to describe the unit when it speaks about itself.

`voice`: The timbre to use in tone markers, i.e. the sounds that accompany speech.

Note that voice gender is separate from the voice used for system messages such as low battery warnings. See "`scheme`". For many controllers and personas, the neuter voice gender is configured to produce beeping sounds instead of vocalizations.

There are four pronoun presets (feminine, masculine, neuter ("they"), and inanimate ("it")), and custom pronouns can be specified independently for physical or mental requirements.

For custom pronouns, enter a string that satisfies the following template:

```
absolute possessive,possessive,subject,object,reflective,name
```

(With no spaces around the commas.) For example, to recreate the values for the neuter gender, one would enter:

```
theirs,their,they,them,themselves,neuter
```

These values are then available for use by personas and Arabesque scripts, as described in "`personas`".

xanadu

```
xanadu list|servers|search|disconnect
```

```
xanadu connect <server>
```

```
xanadu install|info|upgrade|remove <package>
```

Manages installed packages.

`search`: polls for available Xanadu package servers. Performed automatically on region change.

`servers`: lists available servers.

`connect`: connects to the server `<server>`.

`disconnect`: disconnects from the current server.

`list`: lists installed packages. If connected, combines this list with a list of packages on the server.

`install`: requests `<package>` from the server and attempts to install it.

`info`: gives info notecard (readme file) for `<package>`. This works for any package listed by "`xanadu list`".

`upgrade`: installs `<package>` from the server. This guarantees the current version of `<package>` is removed first.

`remove`: removes `<package>` from the unit. A server connection is not required.

Exact names (including versions) are required for installing, upgrading, and removing.

audience

`audience on|off|toggle`

Controls Audience piped audio. Analogous to the Vox pipeline (see "vox"), Audience captures incoming chat messages on channel 0 for modification. This feature is not yet complete, and so no modifications are yet supported.

optics

`optics apply`

`optics default|<preset>`

Sets a video filter using the specified Windlight preset. The special filter "default" disables the filter. "apply" re-applies the current settings.

navigate

`navigate start [<name>]`

`navigate auto`

`navigate speed <rate>`

Controls navigation routing.

`speed`: sets speed of travel between nodes to `<rate>` meters per second. The default value for `<rate>` is 5.0.

`start`: manually start navigation at the last encountered node (or at `<name>` if specified; 96 m search range)

`auto`: toggle automatic starting of navigation.

sentinel

`sentinel`

Reports the system information as tracked by ATOS/E. For more information, see **the online manual**.

```
sentinel inflict heat|combat|crash|external <amount>
```

Inflicts <amount> damage through the indicated method.

```
sentinel repair on|off|toggle
```

Toggles nanite auto-repair. Note that self-repair cannot fully heal the unit.

```
sentinel autoshield on|off|toggle
```

Toggles automatic shield activation. This is triggered whenever the unit collides with a fast-moving object, regardless of its own velocity. A compatible active barrier device, such as the MESH-2100 is required. (This does not include the MSH-2100b.)

halt

```
halt
```

Powers down the system immediately. See "shutdown" for more options.

off

```
off
```

Powers down the system immediately. See "shutdown" for more options.

reboot

```
reboot
```

Reboots the system immediately. See "shutdown" for more options.

preload

```
preload <sound>
```

Prepares the sound file <sound> for playback. This command causes the Arabesque script processor to sleep for 1 second after use.

sound

```
sound <key>
```

Plays a sound file over the audio output processor. The sound must either be located in the processor, or specified by UUID.

```
sound vox <name>
```

Plays a system voice message over the audio output processor. Possibilities are:

battery-0	power-0	charge-0	notice
battery-5	power-profile	bye	subsystem-1
battery-10	module-1	personality-0	subsystem-0
battery-20	module-0	personality-1	volume-100
battery-50	module-reset	error	volume-20
teleport-complete	love-1	error-charge	volume-10
operation-complete	love-2	error-critical	volume-0
access-1	love-3	error-device	motors-0
access-0	love-4	diagnostics-1	motors-1
lock-0	greet	diagnostics-0	motors-2
lock-1	greet-2	device-1	connect-1
power-1	charge-1	device-0	connect-0

trigger

trigger [<event name>]

Runs a script with Arabesque identified as being an event script. Unlike normal Arabesque scripts (see "do"), these scripts start with the "e_" prefix. No message is emitted if the script is not found.

Event scripts are activated automatically by the system during certain events, such as when the system boots or shuts down. As of Companion 8.4, the following events are supported:

e_charge-start: Called when charging begins.

e_charge-end: Called when charging ends.

e_boot: Called on system boot.

e_shutdown: Called on system shutdown.

Personas also have special initialization scripts, called "px_" scripts. See **Personas** for more information.

charge

charge <command>

Sends a command to the charger, if connected. Supported commands vary per charger. Type "charger help" while connected for a list of supported actions.

leash

leash

Causes the unit to follow you.

unleash

unleash

Causes the unit to cease following you.

open

open [<app name>]

Opens a program from the applications menu. If no name is specified, a list is provided.

range

range <distance>

Sets the distance at which the unit is to follow you, measured in meters. By default, <distance> is 5. See "follow" for more information.

reset

reset <module>

Restarts the indicated system module. Ownership may be required to restart some critical modules, such as the user manager and bootloader.

safeword

safeword

This command is only usable by the unit. After notifying the unit's owner(s), it grants basic self-access to the unit. To fully regain autonomy, it is typically necessary to type "keychain reset" afterward.

send

send <range> <channel> <message>

Sends <message> on channel <channel> to all objects within <range> meters. If <range> is 0, sends <message> to the unit's attachments. If <range> is "region", sends to everything in the region.

The special keyword "lights" can be used to indicate that the message should be sent over the light bus.

rlv

Synonym for the "restraint" command. See below.

restraint

`restraint on|off|toggle|reset|auto|manual|cycle|status|debug`

`on/off/toggle`: controls the RLV relay.

`auto/manual/cycle`: controls RLV auto-accept.

`reset`: clear all user-mode restrictions. (Does not affect system restrictions or enable subsystems.)

`status`: list active restraints

`debug`: toggle debug messages

domain

`domain set <domain>`

`domain search`

`domain clear`

`domain sync`

Controls domain membership. A domain is a central server that defines the unit's users, policies, and other settings.

Upon joining a domain with the 'set' command, the unit will automatically be assigned a role and immediately download user and configuration settings. By default, this role is `candidate`.

Depending on the domain's configuration, membership in a group may be required to join.

guests

`guests whitelist|blacklist [<key>]`

`guests unwhitelist|unblacklist <key>`

`guests tidy white|black`

`guests default accept|deny|toggle`

`guests consent on|off|toggle`

Allows the unit to control access by unauthenticated users (hereafter called 'guests'.) To return to behavior from 8.3 and earlier versions, set "`guests default accept`" and "`guests consent off`".

There are four lists: trusted, permitted, banned, and refused. The first two are whitelists, the last two are blacklists. The "permitted" and "refused" lists differ from "trusted" and "banned" only in that their entries are temporary.

Whitelisted guests may access the system as long as "access" is set to public for their interaction modality (see "access").

`whitelist`: lists all whitelisted guests.

`whitelist <key>`: adds a user to the trusted list.

`unwhitelist <key>`: removes a user from either whitelist.

`blacklist`: lists all blacklisted guests.

`blacklist <key>`: adds a user to the banned list.

`unblacklist <key>`: removes a user from either blacklist.

Temporary list entries (permitted and refused) will time out after a certain period of inactivity. The default is ten minutes.

`tidy`: empties the temporary (permitted and refused) lists.

`default`: controls what happens if the unit fails to respond to the confirmation prompt within 10 seconds; `accept` adds the user to the permitted list (whitelisted for 10 minutes), and `deny` adds to the refused list (blacklisted for 10 minutes).

`consent`: if enabled, the unit is prompted to decide what to do whenever a new guest attempts to access the system. If disabled, the default action (see above) automatically occurs.

aux

`aux GPS|audio|radio|video on|off|toggle`

Configures auxiliary power. This provides power to some subsystems for a limited time while the unit is powered down.

The subsystems that can be controlled are `GPS` (view location), `audio` (hearing), `radio` (receive/transmit IMs), and `video` (vision). Some motor functionality (self-touching) is always allowed when auxiliary power is enabled.

If auxiliary power is completely disabled, the unit will not be able to do any of the above when powered down.

Auxiliary power, as the name implies, draws power from a secondary source, called the auxiliary capacitor. Subsystems continue to draw power at their normal rate from the capacitor, and when it is depleted, the unit is totally inactivated.

The auxiliary capacitor is recharged whenever the unit is powered on by drawing power from the main battery. Type "aux" without any arguments to see its status.

beacon

```
beacon fire|test|on|off|toggle|interval <minutes>|message <message>
```

Configures the distress beacon for contacting the unit's owner(s) when the unit is out of power. Normally, the beacon triggers on a repeating interval.

Firing the beacon will cause it to send the configured message to the unit's owners. A test fire adds extra text to the end of this message indicating that there is no crisis.

If GPS is on auxiliary power when the beacon is fired, the unit's position will also be included in the distress signal.

Firing the beacon while the unit is powered down drains the auxiliary capacitor. See "aux" for more information.

tutorial

```
tutorial on|off|toggle
```

```
tutorial forward on|off|toggle
```

```
tutorial cycle
```

Controls the hints system. When forwarding is enabled, messages are displayed through the `_console-screen` HUD. When it is disabled, messages are shown in chat. 'cycle' switches between the three possibilities (off, chat, HUD).

policy

```
policy subsystems|persona|apparel|vox|curfew on|off|toggle
```

```
policy radio open|users|owner|cycle
```

Controls enforcement of the specified policy. The following are the effects of the indicated policies, when turned on:

`subsystems`: prevents adjustment of subsystem (power) settings.

`persona`: prevents adjustment of persona settings.

`apparel`: prevents adjustment of apparel on the unit.

`vox`: prevents adjustment of the speech pipeline.

`curfew`: forces the unit to teleport home at a specified time (see below)

`radio`: limits radio communications to users or owners only.

The keywords "lock" and "unlock" may be used in place of "on" and "off".

```
policy curfew time <hh>:<nn>
```

Forces the unit to teleport home at <hh>:<nn>. Time must be specified in 24-hour format, SLT.

```
policy home here
```

```
policy home set <sim> <x> <y> <z>
```

Sets the home location. The unit will teleport to this location at the curfew time if curfew is enabled.

