PowerHome

PowerHome Users Manual

Version 1.03.4.5

Revision 1

Web: http://www.powerhomeautomation.com

Email: support@myx10.com

Special Notice

This manual has not been updated for version 1.03.4.5. While most of the program remains the same with relatively minor changes that should be able to be adjusted for in actual use, there are a couple of significant changes which will require a more thorough explanation below.

The SENDKEYS table has been renamed to FORMULAS. The Send Keys screen has also been renamed to Formulas. Whereas previously, the send keys formula field was evaluated and the output always sent to the send keys generator, the new formula field is simply evaluated. There is a new column called OUTPUT_SK which you can check to have the result of the formula sent to the send keys generator. This table also had significant changes to its button definition fields detailed below.

The Control Center has been completely restructured in preparation for additional controls. In previous versions you could define Control Center buttons in the Infrared, X-10, Macros, and Send Keys sections. Starting with version 1.03.4.1, all button definitions have been relocated to a single table and screen. This new table is the CCBUTTONS table. Additionally, the old CUSTOMTABS table has been renamed to CCTABS. These changes are reflected in the PowerHome Explorer with a new section labeled “Control Center”. Your first step in creating buttons is to still create the appropriate Infrared, X-10, Macros, and Formulas that you would like to have as buttons. You’ll also create the appropriate Control Center tabs to contain these buttons. In the PowerHome Explorer, each Control Center tab screen has a child screen that will allow you to create the appropriate button types and their corresponding actions. There is also a new menu option in each of the four IR, X-10, Macro, and Formula screens to have the buttons automatically generated for you.

Because of the Control Center restructuring, several other changes will become apparent. In addition to the CCBUTTONS table, a new CCBUTTONSTEMP table has been created as well. Everytime PowerHome is launched or the client side Control Center is opened, the CCBUTTONS table will be copied to the CCBUTTONSTEMP table. Both the client Control Center as well as the Web based Control Center will be constructed from the CCBUTTONSTEMP table. Any changes made to the Control Center via the ph_setcc functions will result in the changes being made to the CCBUTTONSTEMP table as well as reflecting upon the client Control Center if it is open. This means that both the client and web versions of the Control Center can be synchronized and on the fly changed made to the Control Center appearance. In the Setup section of the PowerHome Explorer, you’ll also find that the Control Center now has it’s own section. There are a number of additional parameters that will allow you to tailor the client Control Center to your liking. You’ll also find that you can define a macro that will be executed everytime the client Control Center is opened so you can apply any changes you would like.

Due to these changes, nearly every ph_getcc and ph_setcc function was modified. While I attempt to make changes in such a way that the enduser will not have to make code changes, it was unavoidable this time. Additionally, the ph_lookupcctab function has been removed. Also, the ph_lookupccbtn function has been changed. To assist the enduser in locating code containing these functions so that they can be changed, there is a new report titled “Database Where Used” which will allow you to search the entire database for code using wildcard searches.

The rest of the changes are detailed in the “WHATS NEW” section of the README.RTF file and should be relatively easy to adapt to.

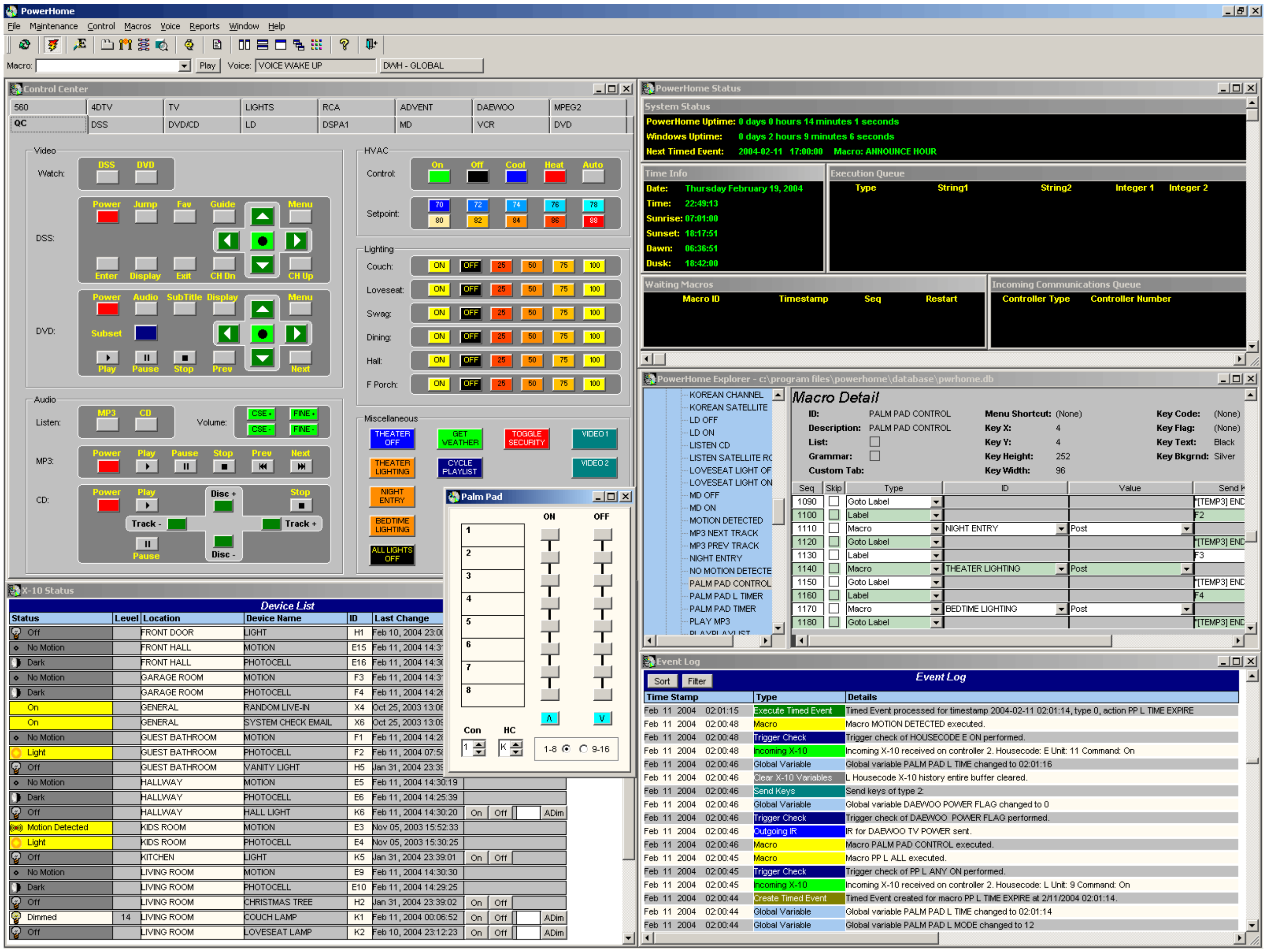
Introduction

PowerHome is an extremely powerful and flexible Windows based software package that will give you full control over your home's lighting, appliances, and infrared devices when used in conjunction with the appropriate control hardware.

With PowerHome's programmable interface, this control can be achieved via keyboard, mouse, web, Email, X-10, IR, Voice recognition, Socket communications, Windows Messaging, and even your internet enabled cell phone.

This user's manual is for version 1.03.4 of PowerHome. If this is your first time using PowerHome, we highly recommend that you first review the QuickStart Guide.

Upgraders be sure and check out Appendix D for a list of important changes from version 1.00.1



PowerHome
File Maintenance Control Macros Voice Reports Window Help
Macro: Play Voice: VOICE WAKE UP DWH - GLOBAL
Control Center
560 4DTV TV LIGHTS RCA ADVENT DAEWOO MPEG2
QC DSS DVD/CD LD DSPA1 MD VCR DVD
Video Watch: DSS DVD
DSS: Power Jump Fav Guide Menu Enter Display Exit CH Dn CH Up
DVD: Power Audio SubTitle Display Menu Subset Play Pause Stop Prev Next
HVAC Control: On Off Cool Heat Auto
Setpoint: 70 72 74 76 78 80 82 84 86 88
Lighting Couch: Loveseat: Swag: Dining: Hall: F Porch: ON OFF 25 50 75 100
Audio Listen: MP3 CD Volume: CSE + FINE + CSE - FINE -
MP3: Power Play Pause Stop Prev Next
CD: Power Play Disc + Stop Track - Track + Pause Disc -
Miscellaneous THEATER OFF GET WEATHER TOGGLE SECURITY VIDEO 1 THEATER LIGHTING CYCLE PLAYLIST VIDEO 2 NIGHT ENTRY BEDTIME LIGHTING ALL LIGHTS OFF
Palm Pad ON OFF 1 2 3 4 5 6 7 8 Con HC 1 K 1-8 9-16

X-10 Status

Device List

Status	Level	Location	Device Name	ID	Last Change				
Off		FRONT DOOR	LIGHT	H1	Feb 10, 2004 23:0				
No Motion		FRONT HALL	MOTION	E15	Feb 11, 2004 14:3				
Dark		FRONT HALL	PHOTOCELL	E16	Feb 11, 2004 14:3				
No Motion		GARAGE ROOM	MOTION	F3	Feb 11, 2004 14:3				
Dark		GARAGE ROOM	PHOTOCELL	F4	Feb 11, 2004 14:2				
On		GENERAL	RANDOM LIVE-IN	X4	Oct 25, 2003 13:0				
On		GENERAL	SYSTEM CHECK EMAIL	X6	Oct 25, 2003 13:0				
No Motion		GUEST BATHROOM	MOTION	F1	Feb 11, 2004 14:2				
Light		GUEST BATHROOM	PHOTOCELL	F2	Feb 11, 2004 07:5				
Off		GUEST BATHROOM	VANITY LIGHT	H5	Jan 31, 2004 23:3				
No Motion		HALLWAY	MOTION	E5	Feb 11, 2004 14:30:19				
Dark		HALLWAY	PHOTOCELL	E6	Feb 11, 2004 14:25:39				
Off		HALLWAY	HALL LIGHT	K6	Feb 11, 2004 14:30:20	On	Off		ADim
Motion Detected		KIDS ROOM	MOTION	E3	Nov 05, 2003 15:52:33				
Light		KIDS ROOM	PHOTOCELL	E4	Nov 05, 2003 15:30:25				
Off		KITCHEN	LIGHT	K5	Jan 31, 2004 23:39:01	On	Off		
No Motion		LIVING ROOM	MOTION	E9	Feb 11, 2004 14:30:30				
Dark		LIVING ROOM	PHOTOCELL	E10	Feb 11, 2004 14:29:25				
Off		LIVING ROOM	CHRISTMAS TREE	H2	Jan 31, 2004 23:39:02	On	Off		
Dimmed	14	LIVING ROOM	COUCH LAMP	K1	Feb 11, 2004 00:06:52	On	Off		ADim
Off		LIVING ROOM	LOVESEAT LAMP	K2	Feb 10, 2004 23:12:23	On	Off		ADim

PowerHome Status

System Status

PowerHome Uptime: 0 days 0 hours 14 minutes 1 seconds
Windows Uptime: 0 days 2 hours 9 minutes 6 seconds
Next Timed Event: 2004-02-11 17:00:00 Macro: ANNOUNCE HOUR

Time Info

Date: Thursday February 19, 2004
Time: 22:49:13
Sunrise: 07:01:00
Sunset: 18:17:51
Dawn: 06:36:51
Dusk: 18:42:00

Execution Queue

Type	String1	String2	Integer 1	Integer 2

Waiting Macros

Macro ID	Timestamp	Seq	Restart

Incoming Communications Queue

Controller Type	Controller Number

PowerHome Explorer - c:\program files\powerhome\database\pwrhome.db

KOREAN CHANNEL
KOREAN SATELLITE
LD OFF
LD ON
LISTEN CD
LISTEN SATELLITE RC
LOVESEAT LIGHT OF
LOVESEAT LIGHT ON
MD OFF
MD ON
MOTION DETECTED
MP3 NEXT TRACK
MP3 PREV TRACK
NIGHT ENTRY
NO MOTION DETECTE
PALM PAD CONTROL
PALM PAD L TIMER
PALM PAD TIMER
PLAY MP3

Macro Detail

ID:	PALM PAD CONTROL	Menu Shortcut:	(None)	Key Code:	(None)
Description:	PALM PAD CONTROL	Key X:	4	Key Flag:	(None)
List:	☐	Key Y:	4	Key Text:	Black
Grammar:	☐	Key Height:	252	Key Bkgrnd:	Silver
Custom Tab:		Key Width:	96		

Seq	Skip	Type	ID	Value	Send K
1090	☐	Goto Label			"[TEMP3] END
1100	☐	Label			F2
1110	☐	Macro	NIGHT ENTRY	Post	
1120	☐	Goto Label			"[TEMP3] END
1130	☐	Label			F3
1140	☐	Macro	THEATER LIGHTING	Post	
1150	☐	Goto Label			"[TEMP3] END
1160	☐	Label			F4
1170	☐	Macro	BEDTIME LIGHTING	Post	
1180	☐	Goto Label			"[TEMP3] END

Event Log

Sort Filter

Time Stamp	Type	Details
Feb 11 2004 02:01:15	Execute Timed Event	Timed Event processed for timestamp 2004-02-11 02:01:14, type 0, action PP L TIME EXPIRE
Feb 11 2004 02:00:48	Macro	Macro MOTION DETECTED executed.
Feb 11 2004 02:00:48	Trigger Check	Trigger check of HOUSECODE E ON performed.
Feb 11 2004 02:00:48	Incoming X-10	Incoming X-10 received on controller 2. Housecode: E Unit: 11 Command: On
Feb 11 2004 02:00:46	Global Variable	Global variable PALM PAD L TIME changed to 02:01:16
Feb 11 2004 02:00:46	Clear X-10 Variables	L Housecode X-10 history entire buffer cleared.
Feb 11 2004 02:00:46	Send Keys	Send keys of type 2:
Feb 11 2004 02:00:46	Global Variable	Global variable DAEWOO POWER FLAG changed to 0
Feb 11 2004 02:00:46	Trigger Check	Trigger check of DAEWOO POWER FLAG performed.
Feb 11 2004 02:00:46	Outgoing IR	IR for DAEWOO TV POWER sent.
Feb 11 2004 02:00:46	Macro	Macro PALM PAD CONTROL executed.
Feb 11 2004 02:00:45	Macro	Macro PP L ALL executed.
Feb 11 2004 02:00:45	Trigger Check	Trigger check of PP L ANY ON performed.
Feb 11 2004 02:00:45	Incoming X-10	Incoming X-10 received on controller 2. Housecode: L Unit: 9 Command: On
Feb 11 2004 02:00:44	Create Timed Event	Timed Event created for macro PP L TIME EXPIRE at 2/11/2004 02:01:14.
Feb 11 2004 02:00:44	Global Variable	Global variable PALM PAD L TIME changed to 02:01:14
Feb 11 2004 02:00:44	Global Variable	Global variable PALM PAD L MODE changed to 12

1.1 Features

- Fully programmable interface via your choice of languages. You can use the internal Macro and formula scripting or any language supported by Windows Script such as VBScript and JScript.
- Internal Web server for remote control and monitoring. Supports user defined dynamic content via PSP (PowerHome Server Pages)
- Voice Recognition
- Text to Speech
- User definable remote layouts allow for virtually any keystroke combination

- Touchscreen friendly interface
- Customizable display allows you to show floorplan layouts and place controls over actual photos
- X-10 control via CM11A, CM17A firecracker, MR26A Mouse Remote, CPU-XA/Ocelot, W800, W800RF32, PowerLinc RS-232
- Infrared control via CIR, Multi-CIR, RedRat2, RedRat3, CPU-XA/Ocelot, Slink-e
- Digital Input, Digital Output, and Analog Input control with the CPU-XA/Ocelot and additional Adicon modules
- Up to 5 infrared controllers, 5 X-10 controllers, and 5 Other controllers can be defined. Supports multi-zone IR
- CD jukebox player control with playlists and autoprogramming
- CD database with freedb support
- Video database
- Macros
- Triggers
- Timed Events
- Full event and Web logging
- Multi-X control allows a sequence of X-10 commands for control (access 256 macros from your palm pad by pressing 2 buttons, 64 macros from your keychain remote by pressing 3 buttons.)
- Sunrise, Sunset, and Civil Twilight calculations
- Socket Server for remote control of PowerHome from other applications and machines
- Windows Messaging Interface for additional remote control capabilities
- Windows Script Host Interface
- Built-in WAP server for remote control via an internet enabled cell phone

1.2 Technical Overview

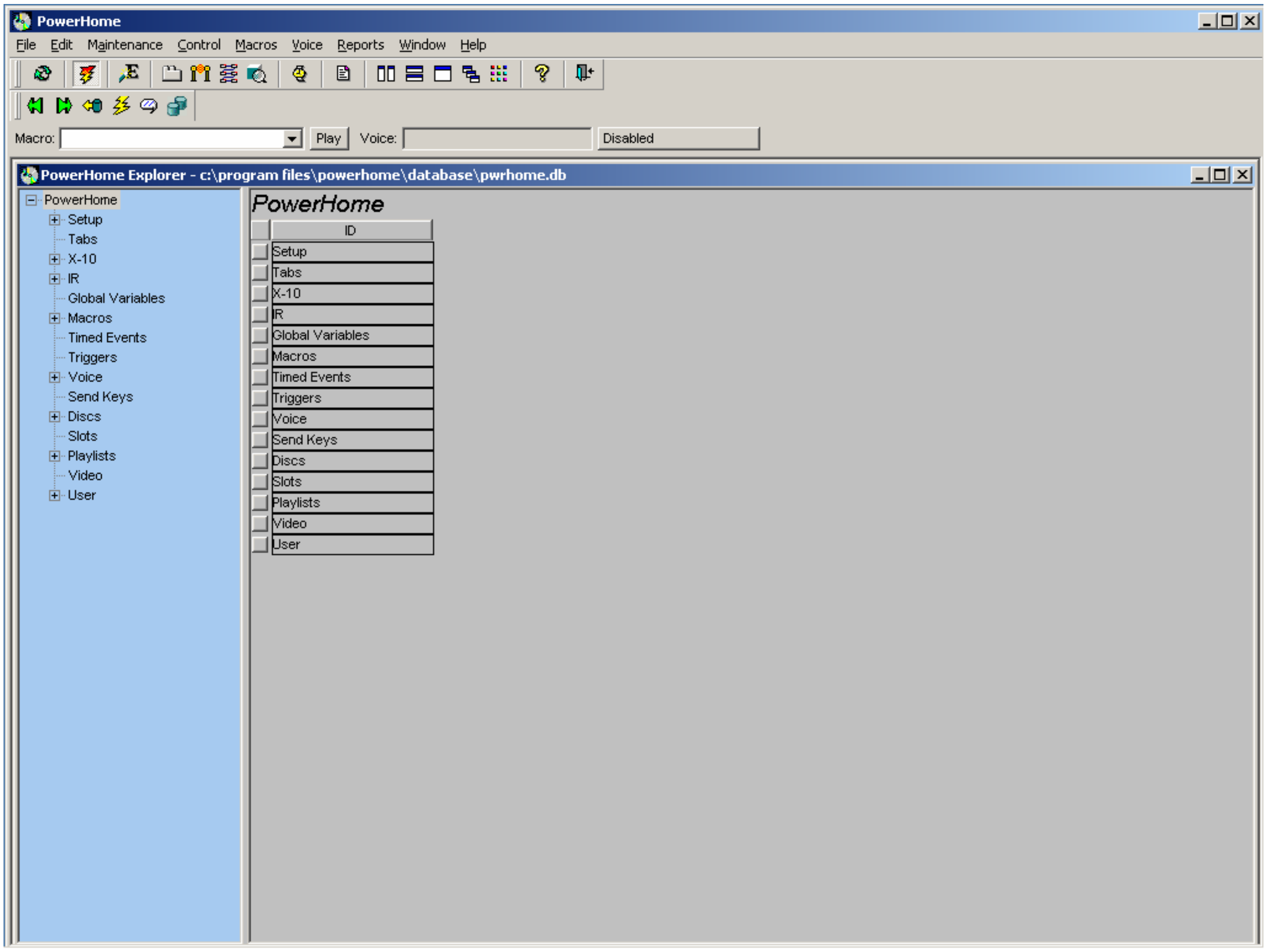
- Written in Sybase PowerBuilder version 9.0.1
- All code and data maintained in Sybase Adaptive Server Anywhere Database system.
- Custom DLL's written in Microsoft Visual C and assembly.
- Voice Recognition and Text to Speech components licensed from Microsoft.
- Remote Access Server Dialup control licensed from SocketWrench.

1.3 System Requirements

- Windows 95/98, Windows NT/2000/XP (CIR device requires Windows 95/98)
- 486 with 16 megs of RAM (Pentium with 32 megs of RAM recommended minimum for voice recognition)
- Unused Serial port/s (CM11A interface, CM17A firecracker, MR26A Mouse remote, RedRat2, CPU-XA/Ocelot, Slink-e, W800, W800RF32, PowerLinc RS-232)
- Unused Parallel port/s (CIR interface, Multi-CIR)
- Unused USB port (RedRat3)
- Sound card (Text to Speech and Voice Recognition)
- Internet Connection (for remote Web Access and freedb support)

2.1 Starting PowerHome the First Time

Once you have successfully installed PowerHome, the first thing you should do is configure the system for your interface devices and personal preferences. The first time you launch PowerHome you will be asked to enter a registration name. This name must be between 5 and 50 characters and will be used to generate your registration number. This registration number can be found under the menu Help->About. After the registration name has been entered, PowerHome will now be running. All configuration and maintenance is done through a single screen called the PowerHome Explorer. You can access this screen from the menu Maintenance->PowerHome Explorer.



ID
Setup
Tabs
X-10
IR
Global Variables
Macros
Timed Events
Triggers
Voice
Send Keys
Discs
Slots
Playlists
Video
User

This screen functions similar to the Windows Explorer. The left pane will show your configurable options in a treeview and allows for easy navigation. The right pane displays the currently selected option and is where changes will be entered. Single-clicking items in the left pane will expand any available sub-items as well as bringing up the appropriate maintenance screen in the right pane. Right-clicking in the left pane will shrink the treeview and maximize your maintenance area in the right pane. Once minimized, you can right click again and restore the treeview to its standard size. Double-clicking in the left pane will have no effect. Single-clicking in the right pane will allow you to change focus between individual fields. Right-clicking will bring up a pop-up menu if available for the current screen. If enabled, double-clicking in the right pane performs two actions. First, any changes made will be automatically saved to the PowerHome database. Second, if the cursor is on a valid item with sub-items, the sub-item screen will be made the current screen. You can also change to a sub-item screen by clicking on the button to the left of each row. When the PowerHome Explorer window is closed, any changes will be automatically saved.

Any changes made to the Setup area of PowerHome Explorer will not take effect until PowerHome is restarted or reinitialized. You must restart or reinitialize PowerHome if you make changes to the Setup area in order for these changes to be recognized.

2.2 Setup

The first area that should be configured is the Setup area. This is where you will define your external interface devices as well as your personal preferences. Navigate to this area by single-clicking Setup in the left pane, double-clicking Setup in the right pane (if double-clicking is enabled), or by clicking on the button to the left of Setup. You will then see the available areas to be configured under Setup.

2.2.1 Preferences

a. ***Background Windows***
PowerHome has three main windows, the Control Center, the X-10 Status Screen, and the Web Center. Each of these windows can be configured to act like normal windows which can be resized and closed. They can also be configured to act as backgrounds for the PowerHome program. When configured as a background, the window cannot be resized, minimized or closed. Other windows will open on top of the background, but the background can always be brought to the top with a hotkey. ***NOTE: The background window preferences for the Web Center do not yet appear on this screen. They can however be edited in the pwrhome.ini file.***

b. ***Drop Down Lines***
The Drop Down Line parameter is default set to 0 and this is fine as the functionality for this has not been fully implemented yet. The only place that it currently has an effect is on the drop down for the Macro Player on the main PowerHome frame.

c. ***Explorer Detail Double Click***
This parameter controls whether you want double clicking the right pane of the PowerHome Explorer enabled. Double clicking in the right hand pane of the explorer will cause any changes to be saved and the data re-queried as well as moving to a child maintenance screen if the double click was on a row that has a child.

d. ***Export Append***
Checking this parameter will cause PowerHome to append any exports made from the PowerHome Explorer to an already existing file rather than overwriting the file if it already exists.

e. ***Resume Waiting Macros***
When a macro is made to wait with the “Wait” command, its state is written to the database. If PowerHome were to be shutdown and restarted (manual or power failure), checking this box would cause any macros which were waiting prior to the shutdown to resume and execute.

f. ***Confirm Shutdown***
Checking this parameter will cause PowerHome to display a messagebox before it shuts down confirming that you indeed would like to shutdown.

g. ***Explorer Color Scheme***
This section allows you to customize the color scheme used in the PowerHome Explorer. You can set the foreground and background colors for the left hand pane (the tree) and the foreground and background colors for both even and odd rows in the right hand pane maintenance screens.

h. ***Event Logging***
The next parameter controls what events are logged in the Event Log. You can select to log only the events that you are interested in tracking. I recommend that you initially log every event until you are comfortable with PowerHome as the log is extremely useful in troubleshooting any problems you may be having.

i. ***Proxy Server***
If you only have access to the internet via a proxy server, this section allows you to define the parameters so that PowerHome can access the internet for freedb access on the formula function ph_geturl.

2.2.2 X-10

a. ***X-10 Controllers***
PowerHome can simultaneously use up to 5 X-10 controllers for both sending and receiving X-10 signals. Select which devices you have and their appropriate ports.

b. ***X-10 Controller received commands***

The checkboxes for each device control what X-10 commands the device should respond to for receiving. Any configured X-10 device can be used to transmit any device supported X-10 signal. However, you wouldn't want each device responding to the same received command. If you had both a CM11A and an Ocelot receiving on, off, and dim commands, then any triggers on these commands would be executed twice, once for the Ocelot, and once for the CM11A. And, in the case of dim commands, the X-10 level status would not be accurate as the dim value would be applied twice, once for each controller. What would be entirely appropriate however, would be to have the Ocelot receive on, off, and dim commands and the CM11A receive extended commands. There would be no conflict and this would give you the best of both worlds. Not selecting a particular command for a particular device doesn't mean that it is totally ignored. The controller does not realize that the command should be ignored so signals PowerHome that a command has been received. PowerHome will respond properly with the controller and do all that is necessary for the controller to continue operating. PowerHome will not process any triggers against the command, nor will it use the command to update any internal variables for X-10 status.

c. ***Previous X-10 Incoming Timeout (sec)***
This parameter is used primarily for multi-x operations and controls what is the maximum time between incoming X-10 signals before it is not considered part of a multi-x operation. Last is whether or not PowerHome should clear its internal X-10 status variables each time it starts. If checked, PowerHome will clear any last known X-10 status and set it to unknown upon startup. The status will only be updated when commands that are sent or received change the status. If unchecked, PowerHome will not clear its variables upon startup and will display the last known status for X-10 devices.

d. ***Clear X-10 Status on Startup***
Sets the X-10 status variables to 'Unknown' whenever PowerHome is started. If cleared, PowerHome will use the last known status.

e. ***Set X-10 Status to Unknown on Status Request***
Enabling this parameter will cause the internal X-10 status variables and X-10 status screen to display 'Unknown' for status whenever a status request command is issued to a device that supports status request.

2.2.3 IR

a. ***IR Controllers***
PowerHome can support up to 5 separate infrared devices for both sending and receiving. Select any infrared devices you have and their appropriate ports.

b. ***Button Delay***
Controls the amount of delay that is automatically inserted before sending infrared commands. The default value of 200 milliseconds should be adequate and keep any infrared commands from becoming garbled.

2.2.4 Other

a. ***Other Controllers***
PowerHome can support up to 5 other controllers for both sending and receiving data other than X-10 and IR such as digital inputs/outputs and analog inputs. Select any other controllers you have and their appropriate ports.

2.2.5 Script

a. ***Global Variable Max Substitutions***
This field determines the maximum global variable substitutions that can occur before an error occurs. This is mainly to prevent an infinite loop situation from occurring. When a global variable is used within a formula or other situations, its actual value is looked up and substituted for the global variable. Since a global variable can contain other global variables that can also contain global variables, this value places a limit on the maximum number of substitutions. An infinite loop situation would occur if global variable {ABC} contained an actual value of {XYZ} which is another global variable. If global variable {XYZ} contains {ABC}, then the substitution algorithm would continue forever and the PowerHome program would be hung. The default value for this field is 25.

b. ***Maximum Number of Recursive Calls***
This field is similar to the Maximum Global Variable Substitutions field in that its purpose is to prevent infinite loop situations. This value comes into effect when a macro calls another macro that calls another macro, ad infinitum. If a situation where macro A calls macro B which calls macro C which calls macro A were to occur,

this value would prevent the PowerHome program from hanging within an infinite loop. The default value for this field is 25.

c. ***Macro Jump Limit***
   This field will allow you to limit the number of times a macro may transfer control to another line within the same macro. Useful in preventing infinite loop situations but could also limit a legitimate macro from looping properly if set to low. Set this value to 0 to disable this feature.
d. ***Use Proxy Server in ph_geturl function***
   Specifies whether the ph_geturl function should use the proxy server values defined under preferences.

2.2.6 Timed

a. ***Activate Timed Events at Startup***
   You can select whether or not PowerHome should automatically launch its Timed Events timer upon startup.
b. ***Execute Past Due Events on Startup***
   If you do have PowerHome automatically start Timed Events, you can also select whether or not you want any past due Timed Events executed at startup.

If you don't execute Timed Events at Startup, then PowerHome will automatically calculate the appropriate time for the Timed Event to next be run. When these parameters are checked, PowerHome will still give you the option at startup to manually cancel either selection. If it has been a long time since PowerHome was last run with a lot of past due Timed Events, it could take an extremely long time for PowerHome to catch up. In this case, it would be best to let Timed Events automatically start, but not execute past due events.

2.2.7 Sun

This Setup area allows you to define your latitude and longitude which is used in the sunrise, sunset, and civil twilight calculations. You can select your state and city, or manually enter your latitude and longitude. You must also manually select whether or not you observe daylight savings time in your area.

2.2.8 CD

The Setup CD section is primarily for defining parameters for a CD jukebox and for the programming of playlists. Most CD jukeboxes allow a sequence of tracks to be programmed into it that it will later playback.

a. ***Maximum Songs in a Playlist***
   This parameter should be set to the maximum number of tracks that your jukebox will support being programmed at a time. This is typically around 30.
b. ***Maximum Number of Jukebox Slots***
   This is the number of slots that your jukebox has. In the case of Pioneer, many of their players have an extra slot for quick playing of a single CD. You would not include this extra slot in the total.
c. ***Number of Digits when Programming Discs***
   This refers to the minimum number of digits to use when programming a playlist into your jukebox. When programming a playlist into your jukebox, some manufacturers only require the necessary digits for a disc. In this case, set this parameter to 0. Other manufacturers may require a minimum of 2 digits. An example would be programming disc number 8. Some jukeboxes may require the disc to be programmed as 0 then 8. Set the parameter to 2 in this case.
d. ***Number of Digits when Programming Tracks***
   This works the same way as Number of Digits when Programming Discs but sets the requirements for tracks instead of discs.
e. ***PlaySlotTrack Macro***
   This field allows you to override the default macro ID of PLAYSLOTTRACK and allows you to designate an alternate macro ID for this functionality. This feature is useful if you require the additional functionality and commands that are available in a standard macro.
f. ***PlayPlayList Macro***
   This field allows you to override the default macro ID of PLAYPLAYLIST and allows you to designate an alternate macro ID. As detailed in the PlaySlotTrack Macro parameter above, this feature is useful for gaining extra functionality if required.

2.2.9 freedb

This section is for setting up freedb. freedb is an online database of CD's and allows you to place CD's into your computers CD player, generate a unique serial number, look that number up online in the freedb database, and save the disc and track information into the Discs section of your PowerHome database. You can select the freedb server you wish to connect to but the default server freedb.freedb.com should be fine. You will need to enter your email address for proper freedb queries to be built. You should also check the Use Proxy Server box if you must use a proxy server to access the internet. The proxy server values are defined in the Preferences section.

2.2.10 Socket Server

PowerHome includes a Socket Server Interface that is useful for remote interfacing with PowerHome. You can also fire triggers in response to socket communications. The Socket Server protocol is loosely based upon the HTTP 1.0 specification.

a. ***Enable Socket Server***
   Check this box if you wish to enable the Socket Server.
b. ***Socket Server Port***
   Set the port that the Socket Server should listen to. The default is 8500.
c. ***UserID and Password***
   Set the userid and password to appropriate values. The socket server uses standard Base64 encoding.
d. ***Restricted IP's***
   If you want to restrict access to the Socket Server to only certain IP's, you must enter them in the Restricted IP's field. If you declare any restricted IP masks, then the socket server may only be accessed if the remote IP matches one of the declared IP masks. If you want the socket server to be accessible by any IP, leave the Restricted IP's field blank. Whether you use restricted IP's or not, the socket server always requires a valid userid and password. The masks may contain exact IP's or end with an asterisk (wildcard character) to match multiple IP's. When entering multiple restricted IP's, you must separate them with a ; (semicolon).

2.2.11 Web

PowerHome includes an internal web server allowing for remote web based control and monitoring of your system. You can use the system generated web pages which provide most of the support you would ever need and/or you can create your own dynamic content using PSP (PowerHome Server Pages). With this formula language, you can totally customize the web interface that you are presented with for PowerHome control.

a. ***Enable Web Server***
   Check this box if you want to make the internal web server available.
b. ***Allow Guest on Internal Pages***
   If the web server is enabled, you may also choose whether to allow the guest login to access the internal web pages. The guest account is userid guest with a password of guest. The guest account cannot make changes to your system with the internal web pages and is limited in what it can see. You may still have sensitive information contained within certain areas of your setup that you don't want the guest account to see. Unless you have a good reason, you should leave the guest account disabled.
c. ***Web Server Directory***
   The web server directory should be set to where PowerHome installed its web files. This directory should be default located in C:\Program Files\PowerHome\Web. If you are creating custom HTML or PSP pages to be served up by the internal web server, you should place them in this directory or a subdirectory of it.
d. ***Web Server Port***
   Set the web server port to the port that you would like to use. If this is your only web server, you should set this to 80 as that is the default port used by web servers. You can set this to a different value, but you will have to enter the port number in the address line of your browser to access PowerHome.
e. ***Master UserID and Password***
   Set the userid and password to appropriate values for you to access the system. The password is hidden on this screen.
f. ***Access***

Within this section you can set the authorization level for each type of external web page that can be served by the PowerHome web server. If you set the PSP pages as 'Any', you can use internal PSP functions for a finer level of control over user access.

g. ***Trusted IP's***
You may any number of trusted IP masks within this section. A trusted IP is not required to enter a userid / password and will have the same authorization as the master user. You may declare the masks as an exact IP or include an asterisk character as a wildcard. If you were running an internal network at home and wanted all of your internal IP's to be trusted, you could enter an IP mask of 192.168.0.*. When entering multiple trusted IP's, separate them with a ; (semicolon).

h. ***Custom Pages***
PowerHome allows you to declare up to 5 custom pages that will be displayed on the menu bar in the PowerHome main webpage. Declare both the title to be displayed and the web based relative path to the HTML or PSP file.

2.2.12 Email

This section controls whether or not PowerHome will process Email with embedded Send Keys, Formulas, or Macros. PowerHome has the ability to retrieve Email at intervals you specify and process any that are intended to control the system.

a. ***Enable Control via Email***
Check this box to enable this feature. You will still have to program PowerHome to periodically check the email either through the macro command Process Email or the formula function ph_processemail.

b. ***Restrict Email Control to Address***
If you've enabled the Control via Email option, you can restrict processing to EMail from up to five Email addresses for added security. If you choose this option, enter the EMail addresses you will process EMail from.

c. ***Restricted Email Control Addresses***
Enter as many email addresses as you like that you will accept Control Email from. Separate multiple email addresses with a , (comma).

d. ***Delete Processed Control Email***
Once you've processed a Control Email, you can have the system delete it.

e. ***MAPI Client Login***
If your MAPI client requires a logon then use this field and the password field to enter your credentials.

f. ***MAPI Client Password***
Enter the password for your default MAPI client if one is required.

PowerHome uses MAPI for its Email processing and you must have a MAPI compliant Email client properly setup. Any Email that is not intended for PowerHome will be retrieved but left unaltered in the inbox of your MAPI compliant client.

2.2.13 VR

This area is for the Voice Recognition abilities of PowerHome. PowerHome uses the Microsoft SAPI 4 modules for its voice recognition and text to speech.

a. ***Enable Voice Recognition***
Choose whether to enable Voice Recognition or not.

b. ***Voice Recognition Autostart***
If VR is enabled, you can choose whether the engine should automatically start when PowerHome is launched.

c. ***Voice Recognition Autouser and Grammar***
If VR is automatically launched at startup, set the user and grammar that it will automatically start with. The Microsoft SAPI allows you to individually train the VR engine on a per user basis.

The Automatic Gain, Complete Timeout, and Incomplete Timeout parameters directly affect the Microsoft SAPI engine. You should not change the default settings unless you are familiar with the engine and the parameters. See the VR Setup section in the Voice chapter.

2.2.14 TTS

This allows you to select whether or not you would like to have Text to Speech enabled and the engine you would like to use. If you would like TTS enabled, you should try to use the Direct SS engine as it is more efficient. Some systems may have problems with this engine so if it doesn't work, you can try the TTS Class engine instead. If you have SAPI 5 installed, you may also use the SAPI 5 selection.

Once you have made all the necessary changes to the SETUP area of PowerHome, reinitialize or shut the PowerHome program down and restart in order for the changes to take effect.

PowerHome Basics

PowerHome is structured around a relational database management system. The database engine that is used is Sybase Adaptive Server Anywhere. This is a SQL based database that is accessible via ODBC from a number of programs including Microsoft Access. Typically, only advanced users with knowledge of databases and SQL would make use of this feature as all database maintenance is fully supported from within PowerHome.

PowerHome is made up of multiple core sections. Understanding these core sections and how they interact with one another is essential to achieving maximum functionality from the PowerHome program.

3.1 Global Variables

Global variables are user defined storage places. Users are able to declare a virtually unlimited number of global variables. Users define the global variable name (up to 25 characters) and the contents of the global variable (up to 1024 characters). Global variables can contain other global variables and can be "stacked" to achieve lengths greater than 1024. All information is stored in a character format. This includes numeric data. If a global variable contains numeric data and is used in a numeric operation, its contents will be converted into the appropriate numeric value. The user does not need to concern themselves with this for the most part. Global variables are declared in the global variable section of PowerHome Explorer. An initial value can also be assigned at this time. Global variables can have their contents changed from within macros and formulas. Global variable contents can be referenced from within formulas and macros where global variable substitution is performed. Global variables maintain their state even when PowerHome is shutdown as they are stored within the database.

3.2 System Variables

System variables are system defined storage areas. They are different from global variables because the user has little control over most of them. They are also only stored within RAM and when PowerHome is shutdown they lose their values. System variables can be accessed just the same as Global Variables. The System Variables that are modifiable by the user have virtually no limit to the amount of data that may be stored within them (a little more than 4 gigabytes). Example system variables include X-10 status, Sunrise, Sunset, local variables, temporary variables, and the Previous Incoming X-10 buffer.

3.3 Formulas

Formulas are where most of your automation power is derived. Formulas are used within macros on several functions, within triggers, within PSP (PowerHome Server Pages) and all Send Keys are a formula whose result is sent to the Send Keys interpreter. Formulas can return text, numeric, and date/time values. Before a formula is evaluated, variable substitution is performed upon it. Once variable substitution is complete, the formula is passed to the formula evaluator. The results of the formula are then passed back. Formulas themselves can contain a number of functions. The Formula Functions are completely documented within the online help.

3.4 Timed Events

Timed Events are simply actions that are solely based on the system clock. Timed Events are declared in the Timed Events section of the PowerHome Explorer. Timed Events can also be created from within macros and formulas. When a timed event is executed, its action is either a macro, formula, or a Send Keys formula. Timed Events are based upon a timestamp (specific time and date) and can have either a recurring frequency (executes and automatically creates a copy of itself a specified interval in the future) or be a one shot (executes one time and then is destroyed).

3.5 Triggers

Triggers are based upon external actions. Triggers can be declared for Outgoing Infrared, Incoming Infrared, Outgoing X-10, Incoming X-10 powerline signals, and Incoming X-10 RF signals. Triggers also include a formula which is checked whenever a trigger fires. If the formula is true then the trigger's action is performed. Actions can be either a macro, formula, or a Send Keys formula. Triggers are declared from within the Triggers section of the PowerHome Explorer.

3.6 Send Keys

Send Keys are PowerHome formulas that after they are evaluated, are basically sequences of keystrokes. These keystrokes can be sent to either the PowerHome application or another running application. Send Keys also has some control elements sprinkled in. Virtually any windows supported keystroke can be emulated through a Send Keys command.

3.7 Macros

Macros are the heart of the PowerHome program. It is here that most actions are defined and if-then logic is performed. Having well defined and thought out macros will enable you to maximize your use of the PowerHome Program. From within macros you can: send pre-defined infrared commands, send raw X-10 commands, call other macros, set global variables, set certain system variables, launch applications, switch to other applications, perform a Send Keys, initiate a dial-up connection, disconnect a dial-up connection, send Email, perform Text to Speech, create timed events, play a CD jukebox playlist, perform delays, trim the event log, get input, post a messagebox, perform direct SQL on the database, retrieve data from the database, clear the X-10 history variables, and nearly any other function for automation control. The Formula language used within macros allows for sophisticated if-then-else logic and looping. Macros are defined within the Macro section of the PowerHome Explorer.

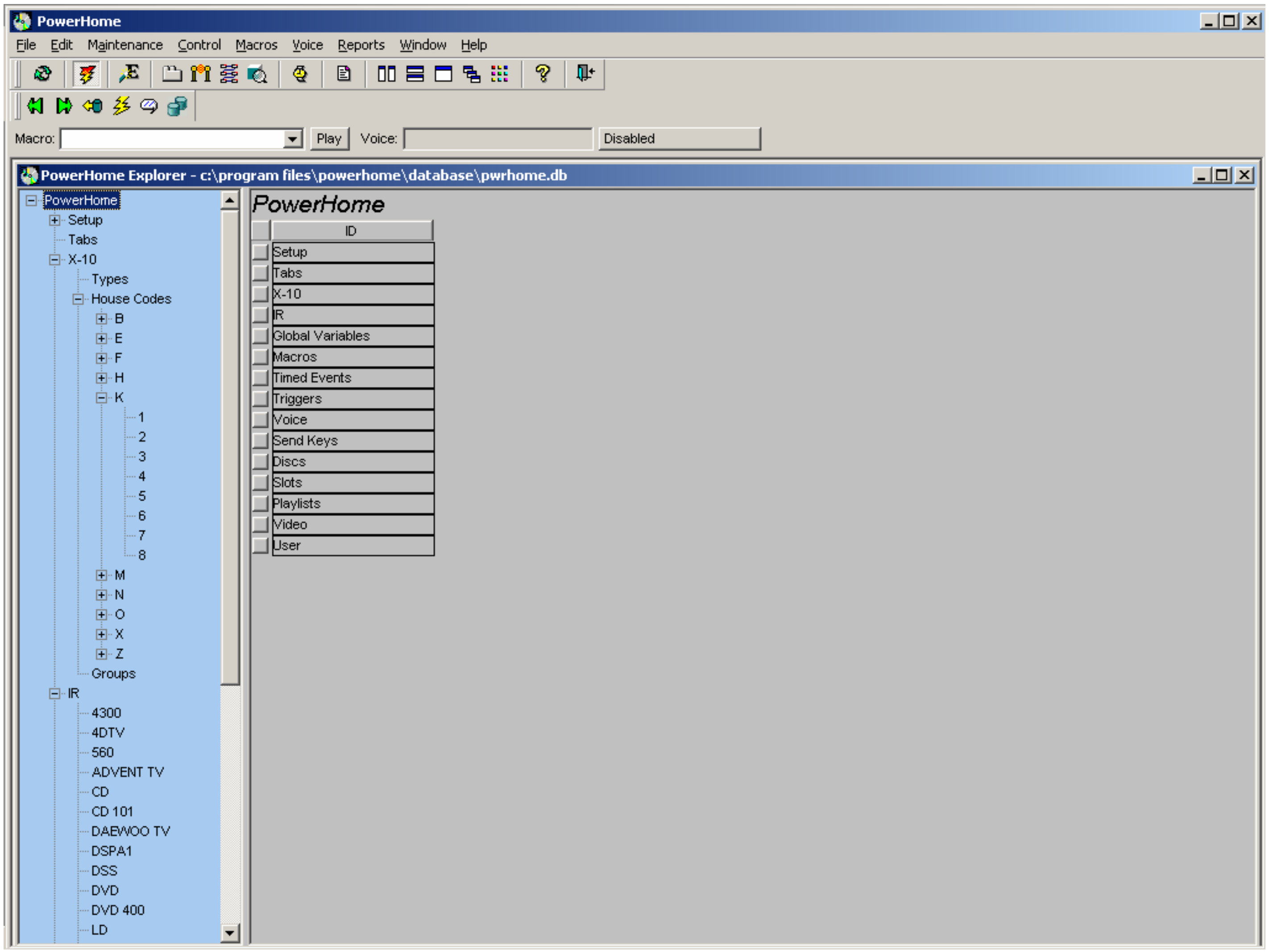
3.8 WSH (Windows Script Host)

WSH is a comprehensive scripting infrastructure for the Windows platform. Script files are saved as ordinary text files on your system and can be in any language that you have installed an engine for. You can execute any WSH file from within a PowerHome formula. PowerHome extends the functionality of WSH with a library of functions that allows the script to directly control and retrieve information from PowerHome. With WSH, you achieve nearly limitless programmability and control of the PowerHome program.

WSH is not part of the PowerHome installation and may have to be installed on your system separately. You can download the WSH files directly from the Microsoft website:
http://msdn.microsoft.com/downloads/default.asp?url=/downloads/sample.asp?url=/msdn-files/027/001/733/msdncompositedoc.xml

PowerHome Explorer

The PowerHome Explorer is where all maintenance to the PowerHome system is performed. This single window allows you to jump section to section with ease and provides a standard interface similar to the Windows Explorer. The PowerHome Explorer can be launched from the toolbar or from the menu under Maintenance / PowerHome Explorer. The Explorer consists of two panes, a left pane which is a hierarchical treeview control that allows for easy navigation to all the PowerHome elements and a right pane where actual changes and maintenance is performed.



You can set the color scheme for the Explorer in the Explorers Setup -> Preferences section. The left hand pane supports single left-clicking and single right-clicking. By left-clicking on the text of a section, that section's maintenance screen will appear in the right hand pane and any children under the section will be expanded and visible. By left-clicking on a plus to the left of the section text, the section will be expanded showing its children without changing the maintenance screen in the right pane. Right-clicking anywhere within the left pane will minimize the left pane or restore it to its default size if already minimized. The Explorer also has a toolbar associated with its functions. Hovering the mouse over any of the toolbar buttons will reveal the tool tip associated with the button as well as the hotkey to directly fire the action. The far left button is the 'Back' button and can be activated by pressing the 'F2' key. The Explorer window maintains a history of each maintenance screen referenced. As you change from screen to screen, this history is built up. Pressing the 'Back' button will step you back through the Explorer's history to the previous screens. The next button is the 'Forward' button and can be activated by pressing the 'F3' key. You can only move forward after you have built some history and moved back with the 'Back' button. Toggling between the 'Back' and 'Forward' buttons will allow you to quickly move between maintenance screens. The next button is the 'Up' button and can also be referenced by pressing the 'F4' key. The 'Up' button may appear similar to the 'Back' button but instead of stepping through history, it will move you from a child to its parent. The next button is the 'Refresh' button and can also be referenced by pressing the 'F5' key. This key will save any changes to the maintenance screen immediately, refresh the left hand pane with any updated information, and re-retrieve all rows within the right hand pane. The button to the right of the ‘Refresh’ button is the ‘Popup’ button. When this sticky button is depressed, it will display long formula fields in a popup windows as you mouse over them. You can also trigger/untrigger this functionality by pressing the ‘F6’ key. The last button is the ‘Database’ button and can be referenced by pressing the ‘F7’ key. This button will allow you to open a different database other than the currently connected one such a sample database.

As you navigate from screen to screen, any changes you make are automatically saved. If you exit the Explorer, or PowerHome, any changes will also be saved.

The right hand pane supports single left-clicking, double left-clicking (if enabled within the preferences section), and single right-clicking. This is where changes to the database are actually performed. Some windows in the right hand pane do not support changes and can only be used for navigation. These windows will typically be all gray. By double-clicking in the right hand pane, a couple of actions are performed. If you double-click in an area that is not a maintenance row, then any changes that have been made are automatically saved and the data is re-retrieved. This is the same as pressing either the 'F5' key or the 'Refresh' tool button. If you double-click on a row, then in addition to the changes being saved, the maintenance screen will change to the screen of the child of the row that you clicked. Left-clicking in the right hand pane will allow you to change rows and fields if you click within an editable field. Right-clicking will bring up a context sensitive pop-up menu with actions that can be performed. For example, if you were in the Macro Header maintenance screen and right-clicked the third row down, a pop-up menu would appear giving you the option to 'Insert', 'Delete', 'Play', 'Copy', or ‘Export’. By clicking 'Insert' you will insert a new blank third row and the original third row will now be the fourth row. If you had clicked 'Delete', the third row would have been deleted. By clicking 'Play', you could play that macro. Clicking 'Copy' would have created a new row at the bottom of all the rows with a copy of the third row macro including all of its children rows (the macro detail). You could then click the ID column of this newly created row and type over the highlighted text, renaming the macro to whatever you wanted. Clicking ‘Export’ will allow you to export the macro header record along with its detail children to a file in SQL format. This file could then be modified and/or emailed to another user and imported into the database using the Direct SQL window.

To add rows into the maintenance screen, you would typically right-click and choose 'Insert' from the pop-up menu. This is a requirement to get at least the first row into a maintenance screen. Where you click determines where the row will be inserted. If you right-click on the fifth row, then the row will be inserted as the fifth row moving the original fifth row and all others below it down one. If you right-click below all rows, then the new row will be the last row. But there is an easier way to add rows if you are entering one row after another. After getting your initial row via the pop-up menu, as you fill out the row and navigate between fields using the 'tab' key, when you fill out the last field and then tab, a new row is created automatically for you. This only happens when the row you are on is the last row (or the only row) and you press the 'tab' key when you are in the last field. This allows for rapid data entry especially when writing macro details.

Another feature of the PowerHome Explorer window are the resizeable panes. If you hover your mouse in the gray area between the left and right panes, you will see that your cursor will change from an arrow to a double left and right arrow. When the cursor looks like this, you can left click and drag the left pane larger or smaller. When you release the mouse the left pane will adjust to the size you selected and the right pane will fill any gaps. Both panes include horizontal and vertical scroll bars when necessary. The right hand pane also includes a split-scroll window. At the botton of the right hand pane, just to the left of the horizontal scrollbar's left arrow button is a black area. When the mouse is hovered over this area, the cursor will change to a double left and right arrow with two vertical lines between them. You can click and drag this to the right and you will be effectively splitting the right hand pane into two pieces, each with its own independent horizontal scrollbar. This is useful when working on long maintenance rows where you might be scrolled far to the right and can no longer see the ID of the rows you are working on.

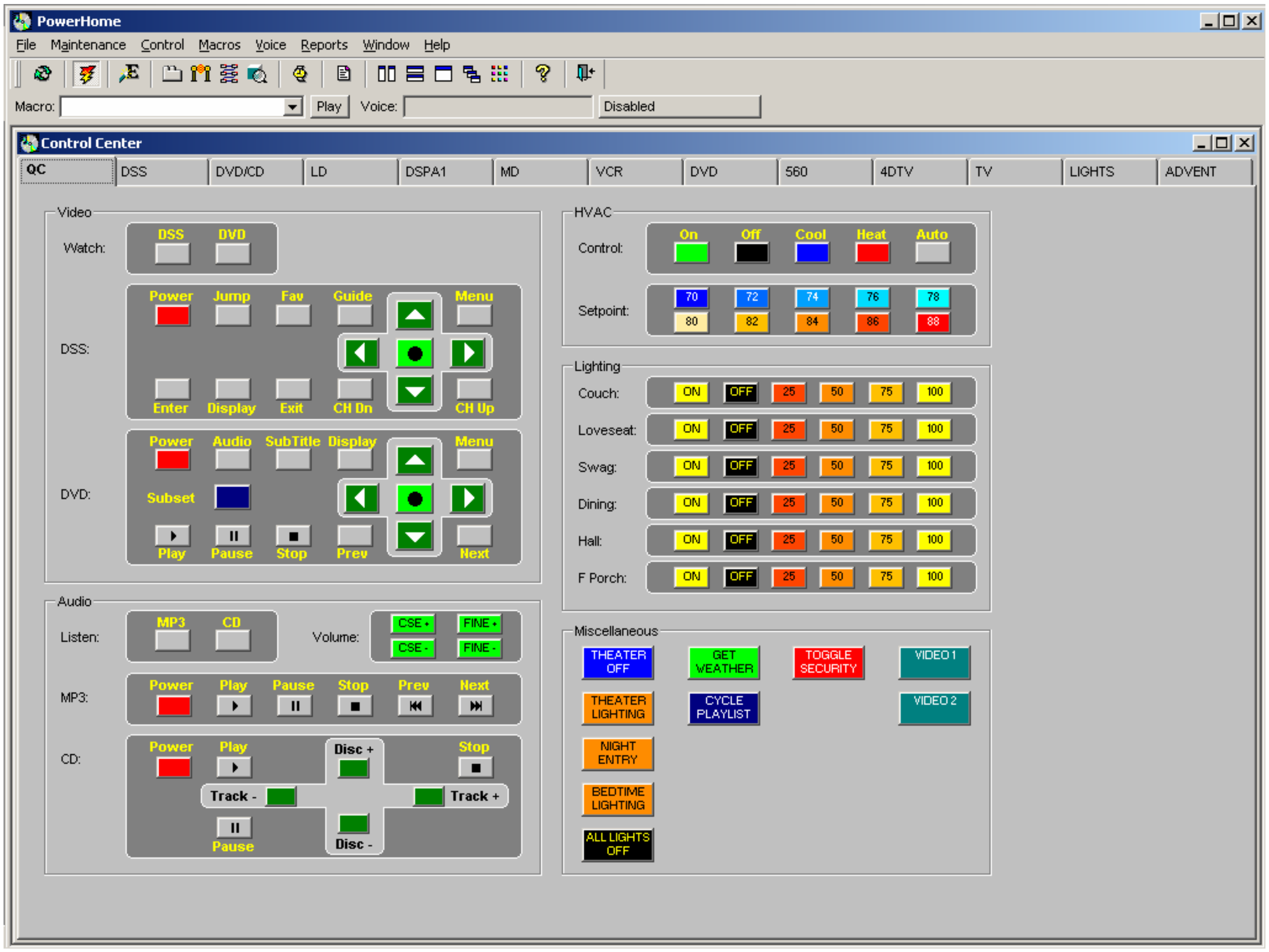
4.1 ID Names

Most of the maintenance screens within PowerHome require you to identify an item by an ID name. These ID names must be 25 or less characters, numbers, and symbols. Certain characters are NOT allowed within an ID name. These unallowed characters are curly braces { }, brackets [ ] , single quote ‘ , double quote “ , and hash mark `. In addition to these restricted characters, the plus + symbol is not allowed in ID names for infrared equipment and playlists. This is a holdover from earlier versions and was necessary to ensure that the Send Keys function did not get confused with the inline functions ir and playlist. This restriction is planned for removal in a future version of PowerHome.

Control Center

The Control Center is the main PowerHome window. This window can be set from within Preferences to behave as a normal window with minimize and maximize buttons or can be set to appear as a background for other windows. The Control Center is fully user definable with the number of tabs that it contains, what appears on each tab, as well as the actions that any buttons on the tab may perform. You can place background bitmaps on any tab which could contain a floormap or an actual picture of an Infrared remote control in which you place hotspots which can be clicked to perform the desired actions. The buttons that you place on the Control Center can be X-10 commands, IR commands, Macros, or Send Keys. You can assign virtually any keystroke combination to these buttons as well as using the mouse to activate them or using a touch screen.

There can only be one Control Center open. It can be opened from the toolbar, from the menu Control / Control Center, or by pressing the hotkey 'Alt-F1'. This hotkey combination works anywhere from within PowerHome and will instantly bring up the Control Center. This is useful if you are trying to use an infrared or RF keyboard without a monitor handy as a super wireless remote. Once the control center is active, you can select the current tab by either clicking on it or pressing a hotkey combination 'F1' thru 'F12' and 'Shift-F1' thru 'Shift-F12'. The far left tab will always be 'F1'. The tab to the right of this will be 'F2', etc. If you have more than 12 tabs, then the 13th tab will be 'Shift-F1', etc. You cannot hotkey to tabs above 24 but you can still mouse to them. You create tabs for the Control Center from the Tabs section of the PowerHome Explorer. You can also set the tabs order from within this maintenance area also.


PowerHome
File Maintenance Control Macros Voice Reports Window Help
Macro: Play Voice: Disabled
Control Center
QC DSS DVD/CD LD DSPA1 MD VCR DVD 560 4DTV TV LIGHTS ADVENT
Video
Watch: DSS DVD
DSS: Power Jump Fav Guide Menu Enter Display Exit CH Dn CH Up
DVD: Power Audio SubTitle Display Menu Subset Play Pause Stop Prev Next
Audio
Listen: MP3 CD
Volume: CSE + FINE + CSE - FINE -
MP3: Power Play Pause Stop Prev Next
CD: Power Play Disc + Stop Track - Track + Pause Disc -
HVAC
Control: On Off Cool Heat Auto
Setpoint: 70 72 74 76 78 80 82 84 86 88
Lighting
Couch: ON OFF 25 50 75 100
Loveseat: ON OFF 25 50 75 100
Swag: ON OFF 25 50 75 100
Dining: ON OFF 25 50 75 100
Hall: ON OFF 25 50 75 100
F Porch: ON OFF 25 50 75 100
Miscellaneous
THEATER OFF
GET WEATHER
TOGGLE SECURITY
VIDEO 1
THEATER LIGHTING
CYCLE PLAYLIST
VIDEO 2
NIGHT ENTRY
BEDTIME LIGHTING
ALL LIGHTS OFF

Different sections of the PowerHome database allow for the definition of buttons on various tabs in the Control Center. When you define a piece of Infrared equipment, you can set a tab that the equipment buttons will appear on. You can also do this when you define a piece of X-10 equipment. When you declare a macro, you can declare what tab a macro button will appear on. You can also declare Send Keys buttons and what tab they will appear on. Typically, you will have a tab per piece of Infrared equipment. You cannot put some buttons from a piece of infrared equipment on one tab and the rest on another. They must all appear on the same tab, but you can declare a Send Keys for the infrared command and mix and match these Send Keys buttons on whatever tab you like.

5.1 Design View

When you declare buttons for the control center, you can set their location, height, width, text color, background color, and a hotkey for them. You can do this from within the PowerHome Explorer, but it's far easier from within the Control Center Design View. Anytime you declare a button or make a change to a button from the PowerHome Explorer, those changes will not be visible until you close and reopen the Control Center (if windowed. If the Control Center is set in preferences as a background you must either restart PowerHome or Reinitialize from the file menu). When you declare buttons in the PowerHome Explorer, they will all default to the same location and same size. Its difficult to design your Control Center layouts from here, so just let everything go to default. Once you are done defining buttons, close and reopen the Control Center so that the newly defined buttons will appear. Next go into Design View by right-clicking on the Control Center whose tab you wish to edit and select Open Design View from the pop-up menu. You will only get the popup menu if buttons have been defined to appear on the tab. An empty tab cannot be designed upon. A new window will open with a copy of the Control Center tab displayed within it. The Design View window will show you approximate screen resolution bars. This design view window also has a toolbar associated with it. There are a number of functions on this toolbar allowing you to design your button layouts graphically.

5.1.1 Design View Toolbar

a. ***Reselect F1***
   Reselects the last group of buttons which were selected.
b. ***Deselect F2***
   Deselects any buttons which are currently selected.
c. ***Select Shift-F8***
   Selects all buttons.
d. ***Set XY F3***
   Allows you to set the X and Y coordinates to a specific value or adjust them by an offset for all currently selected buttons.
e. ***Set HW F4***
   Allows you to set the Height and Width to a specific value or adjust them by an offset for all currently selected buttons.
f. ***Align L F5***
   Align all selected buttons so that their left hand edges are in line. The first selected button is the edge all other buttons will align to.
g. ***Align VC F6***
   Align all selected buttons so that their vertical centers are in line. The first selected button is the vertical center all other buttons will align to.
h. ***Align R F7***
   Align all selected buttons so that their right hand edges are in line. The first selected button is the edge all other buttons will align to.
i. ***Align T F8***
   Align all selected buttons so that their top edges are in line. The first selected button is the edge all other buttons will align to.
j. ***Align HC F9***
   Align all selected buttons so that their horizontal centers are in line. The first selected button is the horizontal center all other buttons will align to.
k. ***Align B F10***
   Align all selected buttons so that their bottom edges are in line. The first selected button is the edge all other buttons will align to.
l. ***Size V Shift-F1***
   Size all selected buttons so that they are the same height. The first selected button is the height all other buttons will size to.
m. ***Size H Shift-F2***

Size all selected buttons so that they are the same width. The first selected button is the width all other buttons will size to.

n. ***Size B Shift-F3***
Size all selected buttons so that both their height and width are the same. The first selected button is the height and width all other buttons will size to.

o. ***Spce V Shift-F4***
Space all selected buttons equidistant vertically. The distance between the first selected button and the second selected button is the distance that is used for all buttons.

p. ***Spce H Shift-F5***
Space all selected buttons equidistant horizontally. The distance between the first selected button and second selected button is the distance that is used for all buttons.

q. ***T Color F11***
Allows you to select the text color for all selected buttons.

r. ***B Color F12***
Allows you to select the background color for all selected buttons.

s. ***Default Shift-F11***
Takes all buttons and sizes them to a default size and then spaces them out in a grid format so that they can be easily manipulated. Useful for when you've just added a number of buttons to a blank tab and they are all default stacked one on top of another. You can only choose this option if no buttons are selected.

t. ***D / D Shift-F10***
Enables Drag and Drop mode. Any selected buttons will be deselected and you can size or move a single button at a time by clicking and dragging.

u. ***Save Shift-F12***
Save any design changes made to the database.

When working within Design View, you can select a button by clicking upon it. The button will display as selected by appearing depressed. Multiple buttons may be selected so that toolbar operations can be performed on all selected buttons. Some toolbar items are only available when two or more buttons are selected (the size functions) and some toolbar items are only available when three or more buttons are selected (the space functions). Whenever a toolbar operation is performed, all selected buttons will be deselected. You can quickly re-select the buttons by pressing the Reselect toolbar item or pressing the F1 key. When buttons are selected, they can also be manipulated by pressing the arrow keys to nudge all selected buttons in a specific direction. If you press the Shift key along with an arrow key, you will adjust the size of all selected buttons. You can also right-click on an area in the Design View and relocate all selected buttons to where you right-clicked. The buttons will be moved in the same layout that they currently are with the first selected button's upper left hand corner moving to the spot that was right clicked and all other buttons moving relative to it.

After saving any changes and closing the Design View window, close and reopen the Control Center (or reinitialize if running in background mode) so that the new design changes will be visible.

Variable Substitution

Within the PowerHome environment, you have access to two types of variables. The first are global variables which can be unlimited in number and are defined by the user and the second are system variables which are finite, some of which can be indirectly created by the user. Global variables are created from within the global variable maintenance section of the PowerHome Explorer. Once created, their values can be changed over and over again from within macros and formulas. A global variable will retain its state or value until changed. Even if PowerHome is shut down, restarted, or the computer should physically lose power, a global variable will retain its last value. Global variables are referenced from within formulas by their ID name (UPPER CASE A MUST) surrounded by curly braces or through an appropriate formula function. A global variable defined within the global variable maintenance section as TEST can be referenced via variable substitution as {TEST}. A system variable is similar to a global variable in that it stores information, but system variables are not defined by the user. System variables also are reinitialized everytime PowerHome is started. They DO NOT retain their state. Some system variables cannot have their values directly changed either. System variables are referenced by their name surrounded by square brackets (UPPER CASE A MUST) or through an appropriate formula function. An example of a system variable via variable substitution is the sunset system variable [SUNSET]. Variable substitution occurs when you have text with references to system and / or global variables. The way this substitution process takes place is with a repetitive looping process. When a formula is processed for variable substitution, global and system variables are searched for from the start of the formula. If a global variable is found first, its value is looked up in the database and substituted for the global variable reference. If a system variable is found first, its value is referenced from memory and substituted for the system variable reference. This process is repeated indefinitely until no system or global variables exist to be replaced or until the max variable substitution limit is reached. This limit is set in the PowerHome Setup section under Scripts. Note that it is possible for global and system variables to contain references to other global and system variables. When a global or system variable is substituted, if the value is a number, you should be careful to place a space before and after the global or system variable if any math is to be performed. This is to solve the problems one will encounter when trying to perform math on global or system variables and not placing a space between the operators and the global or system variables.

NOTE: When referencing global variables and system variables with variable substitution from within formulas, remember to CAPTITALIZE the ID name. If this is not done, it will not be recognized as a variable and no substitution will be performed. Certain Send Keys sequences will appear as a global variable and will be looked up in the global variable table. If not found, they will then be treated as a Send Keys text and not be substituted. For this reason you should not name any global variables F1 through F12 because {F1} through {F12} is a valid Send Keys string. A special case of curly braces is included in the global variable substitution function. An open and close curly brace with nothing between them, ie. {}, will be removed during the substitution and not left in the result as would be the case with any other use of curly braces that is not a global variable. This special situation is useful when dealing with certain issues of ODBC databases which prevent one from storing certain combinations of characters in the first position. An example of this is trying to store {DSS POWER FLAG} in a formula. The {D is assumed to be an ODBC control character. You can eliminate this problem by placing the open and close curly braces in front of the global variable like this: {}{DSS POWER FLAG}. Since the open and close curly braces will be removed during substitution, the net effect is the same as if they never existed.

NOTE: PowerHome has a number of new formula functions which allow you to access the contents of global and system variables without using variable substitution. These functions will return the variables data as the specific data type you request within the formula. Using these functions is slightly more efficient than variable substitution as the functions access the variables contents at evaluation time instead of being searched and replaced prior to formula evaluation. You can find details on these functions in the online help under ph_getglobal, ph_getsystem, and ph_getvar.

Formulas

The formula language within PowerHome is a very powerful language that will allow you to do most PowerHome tasks by calling a series of formula functions. The actions for Timed Events, Triggers, Voice Recognition, and Send Keys buttons can all be declared as a Send Keys (a formula) or a Formula action. A formula will first go through variable substitution. After all variables are substituted, the result is passed to the formula evaluator. If the formula is a Send Keys action then the resulting output from the evaluator will then be passed the Send Keys interpreter. When working with formulas, you can right click on the formula field and open the formula builder window. This window lists all available functions, global variables and system variables. Once your formula is constructed, you can verify that it has no errors and evaluate the formula in a test situation. If a formula returns an exclamation point (!), then the formula has an error. This doesn't necessarily mean that the formula is wrong since the formula may be based upon data that would only be available during runtime.

The way formulas are evaluated and strings are handled will probably be the most difficult concept to grasp. Basically, literal text within a formula must be WITHIN quotes. If you wanted to place a date and time within the formula field of the Create Timed Event macro function and have the date and time 30 minutes in the future with the proper format, you would have to use the following formula: string(today(), "yyyy-mm-dd") + " " + string (relativetime (now(), 1800)). In this case, you must use the string function in order for the formula to evaluate properly. If you have a global variable with text data contained within it and you want to use this string within a formula with variable substitution, you must put the global variable within single or double quotes.

When working with strings, single or double quotes may be used. If you want single or double quotes to actually be a part of the string, you must nest them properly. If you want double quotes within a string, surround the whole string with single quotes and vice versa. For example if you want a string such as: Tom said "Thats neat!", then you would enter the string as: 'Tom said "Thats neat!"'. Notice that the entire string is surrounded with single quotes. The resulting string will actually contain the double quotes emphasizing what Tom said.

When working with formulas, it is recommended that you always check the formula. It is very easy to make a mistake and if a formula has a syntax error, PowerHome will not work the way you intended. In certain situations, doing a formula check will return an error (an error is represented by an exclamation point !) even when there is none and the formula will work just fine when actually executed. This most often occurs when your formula uses certain system variables that are undefined. The [LOCAL1] thru [LOCAL10] system variables only contain values when PowerHome is executing. If you do a check formula and receive an error and are certain there is none, you can temporarily check the formula by assigning it to a global variable and then doing a message box on the global variable and executing the macro. If the message box shows the answer you were expecting, then you can remove this temporary debugging code and assume the formula is valid. If you have an error in your formula and you just can't figure it out, be sure and double check that you have a space before and after each operator. The dash character is allowed within definitions and if you don't have a space before and after this character, it will NOT be interpreted as a minus operator.

Execution Queue

The execution queue is what makes things happen in PowerHome. When you press a button on the control window, or a menu hotkey to launch a macro, or when a trigger or timed event causes an action to be taken, the request is sent to the execution queue. PowerHome was specifically designed to take this myriad of tasks and force them all to be executed one at a time, in the order they were received. This process ensures that nothing happens out of order and keeps the system synchronized. Exceptions have been programmed in however to allow you to override the execution queue and execute an action immediately. This can be seen in the triggers screen where you have the option to either post the trigger action or execute it immediately. You can also see this in the macros section where you can either post or immediately launch another macro or Send Keys sequence. Another exception to processing commands in the order they are received is when an external controller receives a command such as incoming IR or incoming X-10 signals. In this case, the incoming commands are processed as soon as possible before any waiting commands in the execution queue. This is necessary especially in the case of X-10 commands because if the controller is sending it to PowerHome, then that means that any X-10 devices on the powerline has already received and processed the commands. If you had X-10 send commands in the execution queue and processed these before you processed the receive commands, you would lose your X-10 status synchronization.

NOTE: In the Triggers section of PowerHome, a formula field named boolean exists. The result of this formula determines whether or not the trigger is executed. This formula is evaluated at the time a trigger is checked and bypasses the execution queue. It is important that you DO NOT use any formula functions that would normally be sent to the Execution queue. Examples of these functions are ph_macro and ph_sendkeys. Any function that communicates with an external controller should also not be executed within this formula. Examples of these functions are ph_ir and ph_x10. You should ALSO NOT use any of these functions within a trigger action that is set as immediate. A trigger action of immediate also bypasses the execution queue. It is only safe to use the ph_macro function within these formulas if the macro does not in turn attempt to communicate with an external controller or perform an action that would normally be queued.

[DAWN]	This variable returns the start of civil twilight as the number of seconds past midnight.
[DUSK]	This variable returns the end of civil twilight as number of seconds past midnight.
[SUNRISE]	This variable returns the time of sunrise as the number of seconds past midnight.
[SUNSET]	This variable returns the time of sunset as the number of seconds past midnight.
[DIALUPIP]	This variable contains the IP address associated with the current dialup connection.
[INPUTRET]	This variable holds the default value to placed in an input box as well as the value returned from an input box.
[MBRET]	This variables holds the number of the key that was pressed in the most recent message box.
[TIMEOUT]	This variable sets the number of seconds that a macro message box or input box will automatically timeout in. Set this variable to zero if you do not want your boxes to automatically timeout.
[EMAILNAME]	The EMail address to which the Send Email command will send to.
[EMAILSUBJECT]	The EMail subject which the Send Email command will use.
[DIALUPUSER]	The user that the Dialup Connect command will use.
[DIALUPPASSWORD]	The password that the Dialup Connect command will use.
[SQLROWS1]	The number of rows currently retrieved into data retrieval area 1 from an SQL Select statement.
[SQLROWS2]	The number of rows currently in data retrieval area 2.
[SQLROWS3]	The number of rows currently in data retrieval area 3.
[SQLROWS4]	The number of rows currently in data retrieval area 4.
[SQLROWS5]	The number of rows currently in data retrieval area 5.
[SQLROWS6]	The number of rows currently in data retrieval area 6.
[SQLROWS7]	The number of rows currently in data retrieval area 7.
[SQLROWS8]	The number of rows currently in data retrieval area 8.
[SQLROWS9]	The number of rows currently in data retrieval area 9.
[SQLROWS10]	The number of rows currently in data retrieval area 10.
[X10P0]	The most recent incoming X-10 signal that was not excluded in the global buffer.
[X10P1]	
[X10P2]	
[X10P3]	
[X10P4]	
[X10P5]	
[X10P6]	
[X10P7]	
[X10P8]	
[X10P9]	
[X10P10]	
[X10P11]	
[X10P12]	
[X10P13]	
[X10P14]	
[X10P15]	
[X10P16]	
[X10P17]	
[X10P18]	
[X10P19]	The oldest incoming X-10 signal that was not excluded in the global buffer.
[X10P0TIME]	The time of the most recent incoming X-10 signal that was not excluded in the global buffer in the format of hh:mm:ss
[X10P1TIME]	
[X10P2TIME]	
[X10P3TIME]	
[X10P4TIME]	
[X10P5TIME]	
[X10P6TIME]	
[X10P7TIME]	

[X10P8TIME]	
[X10P9TIME]	
[X10P10TIME]	
[X10P11TIME]	
[X10P12TIME]	
[X10P13TIME]	
[X10P14TIME]	
[X10P15TIME]	
[X10P16TIME]	
[X10P17TIME]	
[X10P18TIME]	
[X10P19TIME]	The time of the oldest incoming X-10 signal that was not excluded in the global buffer in the format of hh:mm:ss
[X10STAT??] and [X10LEVEL??]	For each X-10 device declared within the X-10 maintenance screen, two system variables are generated. If an X-10 device identified by housecode A and unit code 2 is declared, then you will have available corresponding system variables of [X10STATA2] and [X10LEVELA2]. The [X10STAT??] variable will return a value of 0 – 2 with 0 being unknown, 1 being off, and 2 being on. The [X10LEVEL??] variable will return a value between 0 and 100 inclusive representing the current dim/bright value.

There are a total of 35 special system variables. 10 of these only have scope within the macro, PSP, or formula from which they were called. These system variables are [LOCAL1] through [LOCAL10]. These system variables can be used as temporary variables to store characters or numbers. This will save you from using global variables as temporary storage and incurring the database read and write. Another 10 special system variables are [TEMP1] through [TEMP10]. These variables serve two purposes. The first is that they can be referenced and set from within macros, PSP, or formulas. When a macro or formula calls another macro or formula, any values within these system variables are passed continuously down through subsequent calls. These are not global in the sense that the values are the same for all macros, PSP, and formulas. For example, if a timed event launches a macro called TEST1, this macro may set the [TEMP1] through [TEMP10] variables to certain values. If this TEST1 macro calls another macro called TEST2, then the TEST2 macro will be able to reference the values that the TEST1 macro placed in the [TEMP1] through [TEMP10] system variables. TEST2 can in turn call a Send Keys formula that can also reference the values in the variables. When the original macro TEST1 completes, the values in the [TEMP1] through [TEMP10] variables are lost. The second use of these variables is to pass parameters from triggers and voice commands. When certain triggers and / or voice commands are processed, some of the [TEMP1] through [TEMP10] system variables will be pre-populated with relevant information that the resulting macro or Send Keys formula can reference. These pre-populated values will be detailed in their relevant sections. The last 20 special variables are [GLOBAL1] through [GLOBAL20]. These variables are truly global and can be seen from within any formula or macro. They are not stored in the database however and will be lost when PowerHome is shut down.

Control Center Tabs

The tab screen in the PowerHome Explorer allows you to create tabs within the Control Center window. The Control Center window is the primary interface for the PowerHome program. This interface consists of a number of tabs. You may create tabs for each piece of infrared equipment, each defined X-10 housecode, tabs for macros, and tabs for Send Keys. Using the Send Keys button maintenance screen, you can place buttons which will evaluate a Send Keys formula. This formula can evaluate to a result which actually sends keystrokes to an application or the formula itself may launch macros, play IR commands, and send X-10 commands.

The tab order controls where the tab will appear in the control center. If you do not want a tab to appear, set this value to 0. Otherwise, set this value to a non negative integer and the tab will appear in the proper order along with all other tabs.

The background color allows you set the background color for the tab in the control center.

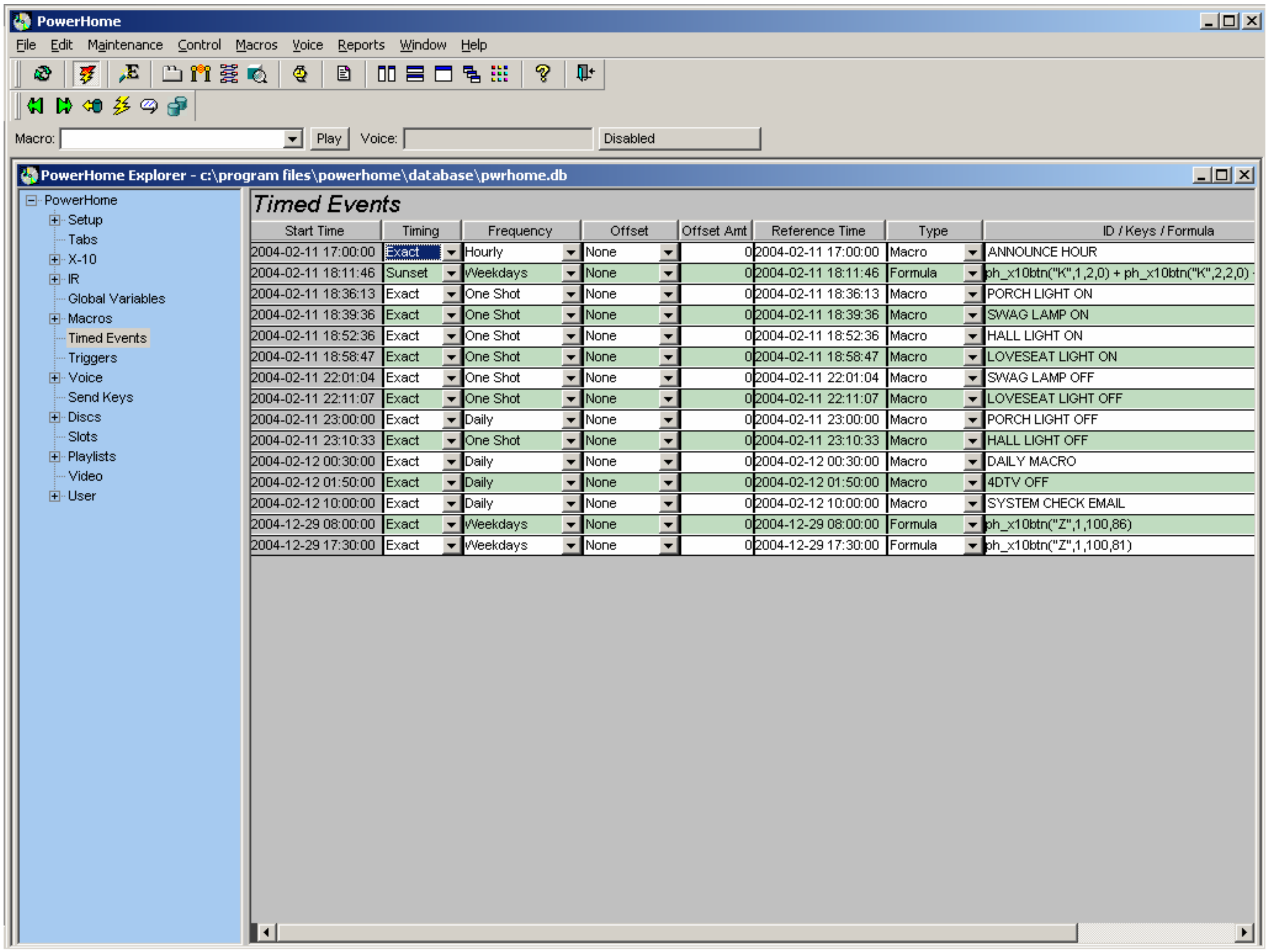
The background graphic allows you to specify a graphic image to be used as the background for the tab in the control center. Leave this selection blank if you do not want a background graphic.

The background graphic resize determines whether or not to resize the background graphic if one is specified. If this box is checked, the background graphic is automatically sized to fill the entire tab control. If this box is not checked, the specified background graphic is placed on the tab at its default size. If the graphic is not large enough to fill the tab, then the background color will fill the rest of the tab.

Timed Events

Timed Events are actions that are performed strictly based upon date and time. When you enter the timed event maintenance window from the PowerHome Explorer, you will see your currently created timed events in the order they are to be executed. To create a new timed event, right click on the window and press insert. You will have a new blank timed event row inserted where you clicked. Once you enter the time and the rest of the timed event parameters and save your changes, your timed event will be sorted according to when it will be executed. In your blank timed event, enter the date and time (24 hour format) that you wish to have your action performed. Next enter the frequency you wish the timed event to be performed. The standard choices are One Shot, Hourly, Daily, and Weekly, but you can enter a number in this field that corresponds to minutes. For example, if you enter a 5 in this field, the timed event will execute every 5 minutes. Enter a 0 if you want your timed event to execute only 1 time and then be deleted. This is equivalent to the One Shot option. Similarly, the Hourly option equates to 60, the Daily equates to 1440 and the Weekly equates to 10080. After setting the frequency, select the type of action to be performed. You can choose either Macro or Send Keys. Next select the Macro, if you chose macro, or enter your Send Keys formula if you chose Send Keys.

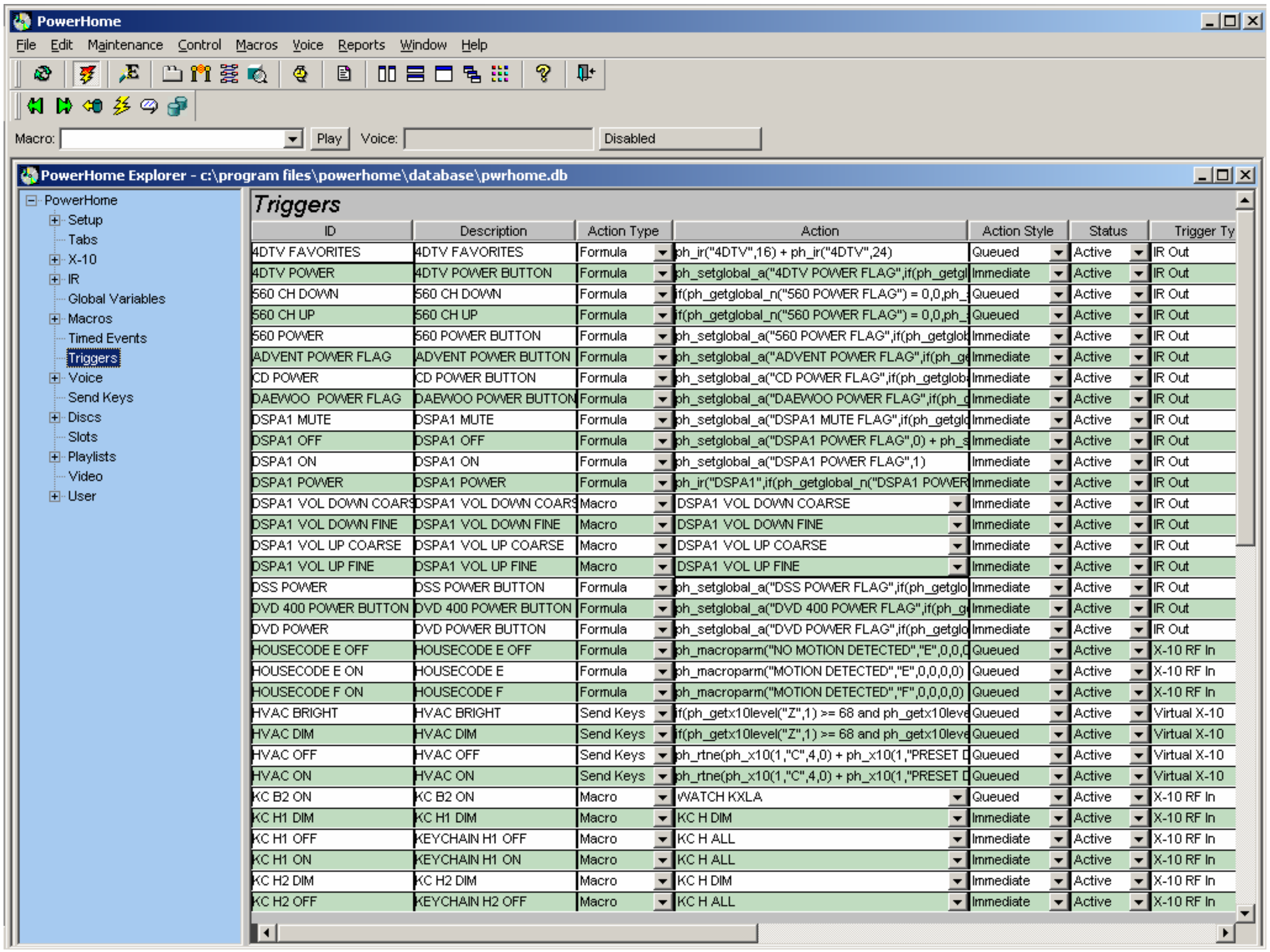
In situations where you want an action performed at a certain time but based on other conditions being satisfied, you would still use a timed event with an action most likely of macro. You could then perform the other condition checking from within the macro, and if the those conditions were not met, merely exit the macro or you could have the macro create another timed event (one shot only) in the future (say 2 minutes) that calls the same macro as the original timed event. This will give you the ability to have an action performed at a specific time, but only if certain conditions were met and if the conditions were not met, check again every two minutes until the action is performed.



Start Time	Timing	Frequency	Offset	Offset Amt	Reference Time	Type	ID / Keys / Formula
2004-02-11 17:00:00	Exact	Hourly	None	0	2004-02-11 17:00:00	Macro	ANNOUNCE HOUR
2004-02-11 18:11:46	Sunset	Weekdays	None	0	2004-02-11 18:11:46	Formula	ph_x10btn("K",1,2,0) + ph_x10btn("K",2,2,0)
2004-02-11 18:36:13	Exact	One Shot	None	0	2004-02-11 18:36:13	Macro	PORCH LIGHT ON
2004-02-11 18:39:36	Exact	One Shot	None	0	2004-02-11 18:39:36	Macro	SWAG LAMP ON
2004-02-11 18:52:36	Exact	One Shot	None	0	2004-02-11 18:52:36	Macro	HALL LIGHT ON
2004-02-11 18:58:47	Exact	One Shot	None	0	2004-02-11 18:58:47	Macro	LOVESEAT LIGHT ON
2004-02-11 22:01:04	Exact	One Shot	None	0	2004-02-11 22:01:04	Macro	SWAG LAMP OFF
2004-02-11 22:11:07	Exact	One Shot	None	0	2004-02-11 22:11:07	Macro	LOVESEAT LIGHT OFF
2004-02-11 23:00:00	Exact	Daily	None	0	2004-02-11 23:00:00	Macro	PORCH LIGHT OFF
2004-02-11 23:10:33	Exact	One Shot	None	0	2004-02-11 23:10:33	Macro	HALL LIGHT OFF
2004-02-12 00:30:00	Exact	Daily	None	0	2004-02-12 00:30:00	Macro	DAILY MACRO
2004-02-12 01:50:00	Exact	Daily	None	0	2004-02-12 01:50:00	Macro	4DTV OFF
2004-02-12 10:00:00	Exact	Daily	None	0	2004-02-12 10:00:00	Macro	SYSTEM CHECK EMAIL
2004-12-29 08:00:00	Exact	Weekdays	None	0	2004-12-29 08:00:00	Formula	ph_x10btn("Z",1,100,86)
2004-12-29 17:30:00	Exact	Weekdays	None	0	2004-12-29 17:30:00	Formula	ph_x10btn("Z",1,100,81)

Triggers

Triggers are where you will define actions that will take place not based upon time of day or date but when a certain external condition triggers it. This currently can be incoming or outgoing infrared, incoming or outgoing X-10 signals, virtual X10, socket server communications, and Windows messaging. In the trigger ID field, type in the name you want associated with this trigger (25 chars or less). Next enter the description and then select the action type. This can be either a Macro or a Send Keys. Depending upon the action type selected, choose the macro or type in an appropriate Send Keys formula. Next, select the action style. This can be either immediate or queued. If you choose immediate, this will be before any actions currently waiting in the execution queue, but after ALL triggers that satisfy the triggering condition are checked. What this means is that you can have multiple triggers that all respond to the same condition, such as an incoming X-10 command on housecode G, unit code 8, function ON. In this situation, all the triggers that respond to this action will be checked in alphabetical order based upon their ID. If the boolean expression evaluates to a nonzero or “true” value, then the triggers action will be performed. If the triggers action style is queued, the action will be posted to the execution queue. If the action is immediate, the action is NOT executed immediately, but instead posted to a temporary execution queue. The next trigger will then have its boolean expression checked and if it is satisfied, its action will be either queued or posted to the temporary execution queue. This will continue for every trigger that satisfies the triggering condition. Once every trigger has been checked, the temporary execution queue will have its actions executed in the order they were inserted (this order is based upon the alphabetical order of the trigger ID). Once the trigger execution function has executed all actions in the temporary execution queue, the trigger terminates and any actions that were sent to the execution queue will now be executed.



ID	Description	Action Type	Action	Action Style	Status	Trigger Ty
4DTV FAVORITES	4DTV FAVORITES	Formula	ph_ir("4DTV",16) + ph_ir("4DTV",24)	Queued	Active	IR Out
4DTV POWER	4DTV POWER BUTTON	Formula	ph_setglobal_a("4DTV POWER FLAG",if(ph_getgl	Immediate	Active	IR Out
560 CH DOWN	560 CH DOWN	Formula	if(ph_getglobal_n("560 POWER FLAG") = 0,0,ph_	Queued	Active	IR Out
560 CH UP	560 CH UP	Formula	if(ph_getglobal_n("560 POWER FLAG") = 0,0,ph_	Queued	Active	IR Out
560 POWER	560 POWER BUTTON	Formula	ph_setglobal_a("560 POWER FLAG",if(ph_getglob	Immediate	Active	IR Out
ADVENT POWER FLAG	ADVENT POWER BUTTON	Formula	ph_setglobal_a("ADVENT POWER FLAG",if(ph_g	Immediate	Active	IR Out
CD POWER	CD POWER BUTTON	Formula	ph_setglobal_a("CD POWER FLAG",if(ph_getglob	Immediate	Active	IR Out
DAEWOO POWER FLAG	DAEWOO POWER BUTTON	Formula	ph_setglobal_a("DAEWOO POWER FLAG",if(ph_g	Immediate	Active	IR Out
DSPA1 MUTE	DSPA1 MUTE	Formula	ph_setglobal_a("DSPA1 MUTE FLAG",if(ph_getgl	Immediate	Active	IR Out
DSPA1 OFF	DSPA1 OFF	Formula	ph_setglobal_a("DSPA1 POWER FLAG",0) + ph_s	Immediate	Active	IR Out
DSPA1 ON	DSPA1 ON	Formula	ph_setglobal_a("DSPA1 POWER FLAG",1)	Immediate	Active	IR Out
DSPA1 POWER	DSPA1 POWER	Formula	ph_ir("DSPA1",if(ph_getglobal_n("DSPA1 POWER	Immediate	Active	IR Out
DSPA1 VOL DOWN COARS	DSPA1 VOL DOWN COARS	Macro	DSPA1 VOL DOWN COARSE	Immediate	Active	IR Out
DSPA1 VOL DOWN FINE	DSPA1 VOL DOWN FINE	Macro	DSPA1 VOL DOWN FINE	Immediate	Active	IR Out
DSPA1 VOL UP COARSE	DSPA1 VOL UP COARSE	Macro	DSPA1 VOL UP COARSE	Immediate	Active	IR Out
DSPA1 VOL UP FINE	DSPA1 VOL UP FINE	Macro	DSPA1 VOL UP FINE	Immediate	Active	IR Out
DSS POWER	DSS POWER BUTTON	Formula	ph_setglobal_a("DSS POWER FLAG",if(ph_getglo	Immediate	Active	IR Out
DVD 400 POWER BUTTON	DVD 400 POWER BUTTON	Formula	ph_setglobal_a("DVD 400 POWER FLAG",if(ph_g	Immediate	Active	IR Out
DVD POWER	DVD POWER BUTTON	Formula	ph_setglobal_a("DVD POWER FLAG",if(ph_getglo	Immediate	Active	IR Out
HOUSECODE E OFF	HOUSECODE E OFF	Formula	ph_macroparm("NO MOTION DETECTED","E",0,0,0	Queued	Active	X-10 RF In
HOUSECODE E ON	HOUSECODE E	Formula	ph_macroparm("MOTION DETECTED","E",0,0,0,0)	Queued	Active	X-10 RF In
HOUSECODE F ON	HOUSECODE F	Formula	ph_macroparm("MOTION DETECTED","F",0,0,0,0)	Queued	Active	X-10 RF In
HVAC BRIGHT	HVAC BRIGHT	Send Keys	if(ph_getx10level("Z",1) >= 68 and ph_getx10leve	Queued	Active	Virtual X-10
HVAC DIM	HVAC DIM	Send Keys	if(ph_getx10level("Z",1) >= 68 and ph_getx10leve	Queued	Active	Virtual X-10
HVAC OFF	HVAC OFF	Send Keys	ph_rtne(ph_x10(1,"C",4,0) + ph_x10(1,"PRESET D	Queued	Active	Virtual X-10
HVAC ON	HVAC ON	Send Keys	ph_rtne(ph_x10(1,"C",4,0) + ph_x10(1,"PRESET D	Queued	Active	Virtual X-10
KC B2 ON	KC B2 ON	Macro	WATCH KXLA	Queued	Active	X-10 RF In
KC H1 DIM	KC H1 DIM	Macro	KC H DIM	Immediate	Active	X-10 RF In
KC H1 OFF	KEYCHAIN H1 OFF	Macro	KC H ALL	Immediate	Active	X-10 RF In
KC H1 ON	KEYCHAIN H1 ON	Macro	KC H ALL	Immediate	Active	X-10 RF In
KC H2 DIM	KC H2 DIM	Macro	KC H DIM	Immediate	Active	X-10 RF In
KC H2 OFF	KEYCHAIN H2 OFF	Macro	KC H ALL	Immediate	Active	X-10 RF In

The [TEMP1] system variable will always be populated with the ID of the trigger and this info will be accessible to the macro or Send Keys formula. When working with incoming and outgoing X-10, Socket, WM_COPYDATA, and

WM_USER trigger types, the [TEMP2] thru [TEMP5] system variables are populated with additional relevant information that can be utilized by the macro or Send Keys formula. This information is summarized in the table below:

	[TEMP1]	[TEMP2]	[TEMP3]	[TEMP4]	[TEMP5]
X-10 Specific Address	ID	-	-	-	-
X-10 Any Address	ID	-	Unitcode	-	-
X-10 Any All Units Off, All Lights Off, All Lights On	ID	-	-	-	-
X-10 Specific Dim, Bright	ID	Dim	-	-	-
X-10 Any Dim, Bright	ID	Dim	-	-	-
X-10 Specific Extended	ID	-	Command	Data	-
X-10 Any Extended	ID	Unitcode	Command	Data	-
X-10 Specific Preset Dim	ID	-	Dim	-	-
X-10 Any Preset Dim	ID	-	Dim	Unitcode	-
Socket	ID	-	-	-	Data
WM_COPYDATA	ID	-	Wparam	Dwdata	Data
WM_USER	ID	-	Wparam	Lparam	-

NOTE: Care should be taken when defining the formulas for the Send Keys Action if the Action Style is Immediate and for the Boolean expression. You should not execute formula functions that issue commands to external controllers or other commands that would normally go through the execution queue. More details on this subject can be found the Execution Queue section.

Send Keys

Send Keys start out as a PowerHome formula. This formula may be as simple as a sequence of keystrokes to be sent to the current window surrounded by single or double quotes so that the formula will evaluate properly. After the formula goes through variable substitution and evaluation, the result is passed to the Send Keys interpreter. The interpreter will then parse the result and simulate keystrokes or perhaps execute one of the Send Keys functions. To simulate someone pressing the 'A' key, the Send Keys formula would be the letter 'a' in lower case surrounded by single or double quotes. This string would be passed to the variable substitution routine where it would return unchanged. Next the formula evaluator will evaluate the formula and return the lower case 'a' without the surrounding quotes. The Send Keys interpreter would then create a Windows message telling the currently active application that the 'a' key was pressed. If you wanted to simulate someone pressing the shift key and the 'A' key, the command would be the letter 'A' in upper case surrounded by single or double quotes. This is the technique used for sending the standard typeable letters, numbers and special characters. If a key sequence does not have a standard sequence such as the tab key, you would use an alternative syntax such as '{tab}'. The curly braces denote a special typing sequence. To simulate an alt + tab key combination, the syntax is '{alt+tab}'. Notice there are no spaces in this sequence. Having a space in a Send Keys sequence will simulate the space key being pressed. Also note that all the letters are lower case. This is important. Upper case letters in a special typing sequence are only used in certain situations. The + symbol is used to separate items in a special typing sequence. To simulate a function key such as F1, the syntax is '{F1}'. Note that the F is capitalized. This is the only time capitalization should be used in a special typing sequence. What follows is the syntax for the keys in the special typing sequence:

{shift}
{ctrl}
{alt}
{backspace}
{tab}
{enter}
{pause}
{capslock}
{esc}
{pageup}
{pagedown}
{end}
{home}
{leftarrow}
{rightarrow}
{uparrow}
{downarrow}
{printscreen}
{insert}
{delete}
{multiply}
{plus}
{minus}
{point}
{divide}
{numlock}
{scrolllock}

To simulate the numbers on the numpad keys such as the 1 key, use the syntax '{numpad+1}'. In the above example where we wanted to simulate the shift and 'A' key being pressed, we merely had to enter a capital 'A'. We could also use the syntax '{shift+a}' to accomplish the same. If you want to simulate a key sequence such as the control key, the shift key and the 'A' key, your syntax would be either '{shift+ctrl+a}' or '{ctrl+shift+a}'. Remember, when building Send Keys formulas, do not separate special typing sequences or groups of characters by spaces unless you want the space key to be pressed. To simulate a sequence of keys pressed one after the other, string the appropriate Send Keys syntax together one after another. Control A followed by the letter b followed by the enter key followed by the F1 key followed by an alt + tab sequence would be entered into a Send Keys formula as: '{ctrl+a}b{enter}{F1}{alt+tab}'.

Curly braces are also used to delimit inline functions within Send Keys sequences. The inline functions do not simulate keystrokes, but instead performs a specific action. The inline function list follows:

{comment}	Include a comment within the Send Keys.
{exit}	Exit the PowerHome program immediately.
{#} or {formula}	Evaluate a formula.
{}	Null command.

The {comment} function requires the actual text of the comment to be included within the function separated by the + symbol. The actual text of the comment after the + symbol may include spaces and you can use whatever capitalization you like. An example of this is: '{comment+This send keys will send the letter A to the current window}'.

The '{exit}' function will cause PowerHome to exit immediately.

The {#} or {formula} inline function allows you to evaluate a formula from within a Send Keys. This formula is evaluated after the Send Keys formula is evaluated and the result sent to the Send Keys interpreter. The syntax for having a formula evaluated requires you to place the formula inside a DOUBLE set of open and close braces immediately after the formula command {#}. Care should be taken that a double set of close braces does not appear within the actual formula. The formula within the double open and close braces will be evaluated and the result discarded. An example of this is: '{comment+Set the LOCAL1 variable to 5}{#}{{ph_setvar_a(1,1,5)}}'.

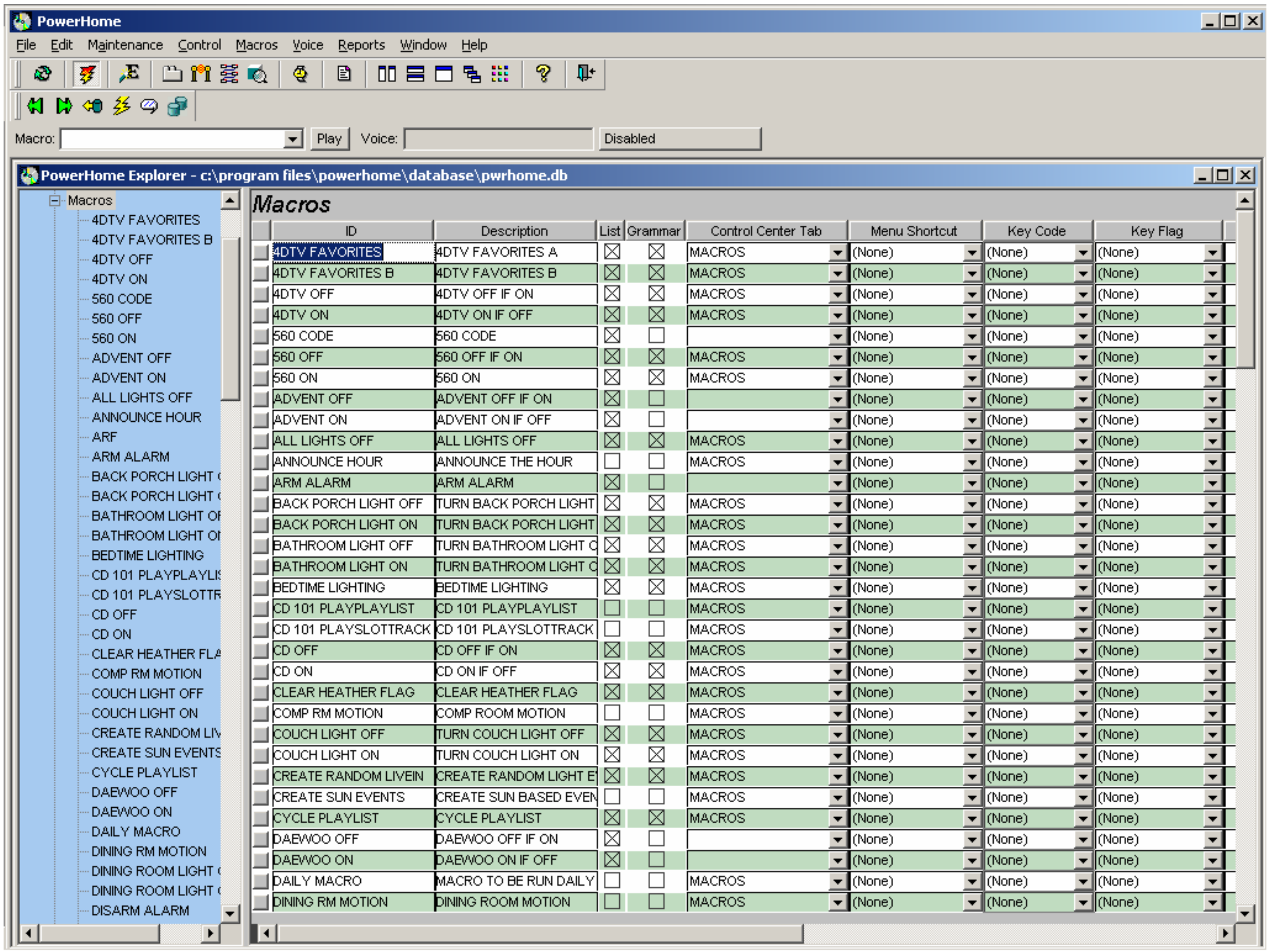
The '{}' sequence (curly braces with nothing in between) has no effect and is ignored.

Remember: All Send Keys are first a formula. You can achieve dynamic Send Keys by having this formula evaluate to a sequence of keystrokes that will then be passed to the Send Keys interpreter. Sometimes you may not even want to simulate keystrokes and the actual functionality desired is contained within the formula. In this situation, be sure that the formula evaluates to an empty string or to the null Send Keys sequence "{}". Otherwise the result of the formula will be simulated as keystrokes. The ph_rtne is useful for making most formulas evaluate to an empty string.

NOTE: Case is extremely important when working with Send Keys commands. If you are not careful concerning case, your Send Keys will not be interpreted properly. When evaluating a formula within your Send Keys with the {#} function, make sure that the formula contained within your double open and close braces does not appear as a global variable. If your formula exactly matches a global variable (including case), then the formula will be substituted with the data contained within the global variable.

Macros

You can declare macros in the macro maintenance section of the PowerHome Explorer. Macros are defined in two parts, a macro header and a macro detail. In the header section you declare the ID, a description, whether or not you want a button on the Control Center, etc. The macro detail section allows you to define the specific individual actions that a macro will perform. The first thing to consider when declaring a macro is the ID. This macro id can be up to 25 characters but its better to keep them short and descriptive. If they are short, you can easily launch a macro by typing its name in the macro launcher and pressing enter. Macros can also be launched if they are assigned a menu hot key. You can also launch a macro from a button on the Control Center by pressing the appropriate keystroke combination or clicking with the mouse. When giving the macro a description, you'll want to keep it short and to the point because the description field is what will appear on any Control Center buttons. The List checkbox determines whether the macro will appear in the Macro Player dropdown list in both the PowerHome frame and the Web server main page. The grammar checkbox will determine whether the macro will appear in the voice recognition grammar for macros. Next, pick the custom tab you wish this button to appear on if any. If you wish the macro button to appear on a custom tab, you must first create that custom tab and then assign the macro button to it. If you would like the macro to be instantly accessible from anywhere within the PowerHome application, you can assign it a menu shortcut. However, you can only assign a maximum of 48 macros to the menu shortcut keys. Use this feature for your most used macros. Assign an appropriate key code / key flag combination if you wish. Since the number of macros can become quite large, it is unlikely that you will assign a key code / key flag to every macro and probably will only assign a few since its easier to simply type the macro name in the macro player. The X, Y, Width, Height, Text color, and Button color fields can be filled out here but it easier to do it from within the Control Center Design View window.


PowerHome
File Edit Maintenance Control Macros Voice Reports Window Help
Macro: Play Voice: Disabled
PowerHome Explorer - c:\program files\powerhome\database\pwrhome.db

Macros
- 4DTV FAVORITES
- 4DTV FAVORITES B
- 4DTV OFF
- 4DTV ON
- 560 CODE
- 560 OFF
- 560 ON
- ADVENT OFF
- ADVENT ON
- ALL LIGHTS OFF
- ANNOUNCE HOUR
- ARF
- ARM ALARM
- BACK PORCH LIGHT (
- BACK PORCH LIGHT (
- BATHROOM LIGHT OI
- BATHROOM LIGHT OI
- BEDTIME LIGHTING
- CD 101 PLAYPLAYLIS
- CD 101 PLAYSLOTTR
- CD OFF
- CD ON
- CLEAR HEATHER FLA
- COMP RM MOTION
- COUCH LIGHT OFF
- COUCH LIGHT ON
- CREATE RANDOM LIV
- CREATE SUN EVENTS
- CYCLE PLAYLIST
- DAEWOO OFF
- DAEWOO ON
- DAILY MACRO
- DINING RM MOTION
- DINING ROOM LIGHT (
- DINING ROOM LIGHT (
- DISARM ALARM

Macros

ID	Description	List	Grammar	Control Center Tab	Menu Shortcut	Key Code	Key Flag
4DTV FAVORITES	4DTV FAVORITES A	☒	☒	MACROS	(None)	(None)	(None)
4DTV FAVORITES B	4DTV FAVORITES B	☒	☒	MACROS	(None)	(None)	(None)
4DTV OFF	4DTV OFF IF ON	☒	☒	MACROS	(None)	(None)	(None)
4DTV ON	4DTV ON IF OFF	☒	☒	MACROS	(None)	(None)	(None)
560 CODE	560 CODE	☒	☐		(None)	(None)	(None)
560 OFF	560 OFF IF ON	☒	☒	MACROS	(None)	(None)	(None)
560 ON	560 ON	☒	☒	MACROS	(None)	(None)	(None)
ADVENT OFF	ADVENT OFF IF ON	☒	☐		(None)	(None)	(None)
ADVENT ON	ADVENT ON IF OFF	☒	☐		(None)	(None)	(None)
ALL LIGHTS OFF	ALL LIGHTS OFF	☒	☒	MACROS	(None)	(None)	(None)
ANNOUNCE HOUR	ANNOUNCE THE HOUR	☐	☐	MACROS	(None)	(None)	(None)
ARM ALARM	ARM ALARM	☒	☐		(None)	(None)	(None)
BACK PORCH LIGHT OFF	TURN BACK PORCH LIGHT	☒	☒	MACROS	(None)	(None)	(None)
BACK PORCH LIGHT ON	TURN BACK PORCH LIGHT	☒	☒	MACROS	(None)	(None)	(None)
BATHROOM LIGHT OFF	TURN BATHROOM LIGHT O	☒	☒	MACROS	(None)	(None)	(None)
BATHROOM LIGHT ON	TURN BATHROOM LIGHT O	☒	☒	MACROS	(None)	(None)	(None)
BEDTIME LIGHTING	BEDTIME LIGHTING	☒	☒	MACROS	(None)	(None)	(None)
CD 101 PLAYPLAYLIST	CD 101 PLAYPLAYLIST	☐	☐	MACROS	(None)	(None)	(None)
CD 101 PLAYSLOTTRACK	CD 101 PLAYSLOTTRACK	☐	☐	MACROS	(None)	(None)	(None)
CD OFF	CD OFF IF ON	☒	☒	MACROS	(None)	(None)	(None)
CD ON	CD ON IF OFF	☒	☒	MACROS	(None)	(None)	(None)
CLEAR HEATHER FLAG	CLEAR HEATHER FLAG	☒	☒	MACROS	(None)	(None)	(None)
COMP RM MOTION	COMP ROOM MOTION	☐	☐	MACROS	(None)	(None)	(None)
COUCH LIGHT OFF	TURN COUCH LIGHT OFF	☒	☒	MACROS	(None)	(None)	(None)
COUCH LIGHT ON	TURN COUCH LIGHT ON	☒	☒	MACROS	(None)	(None)	(None)
CREATE RANDOM LIVEIN	CREATE RANDOM LIGHT E	☒	☒	MACROS	(None)	(None)	(None)
CREATE SUN EVENTS	CREATE SUN BASED EVEN	☐	☐	MACROS	(None)	(None)	(None)
CYCLE PLAYLIST	CYCLE PLAYLIST	☒	☒	MACROS	(None)	(None)	(None)
DAEWOO OFF	DAEWOO OFF IF ON	☒	☐		(None)	(None)	(None)
DAEWOO ON	DAEWOO ON IF OFF	☒	☐		(None)	(None)	(None)
DAILY MACRO	MACRO TO BE RUN DAILY	☐	☐	MACROS	(None)	(None)	(None)
DINING RM MOTION	DINING ROOM MOTION	☐	☐	MACROS	(None)	(None)	(None)

Once done defining your macro in the header section, go to the detail by double-clicking the macro in the header section. From here, you can use any of the available macro commands to accomplish the desired task. Successful macro writing will take some time and playing around to get the hang of it. Although much easier than writing straight code, it still takes practice. Following is a list of the available macro commands.

14.1 Macro Commands

14.1.1 IR

This function allows you to send an infrared code that is already learned by the system. You will need to select the equipment and the infrared button.

14.1.2 X-10 1 thru X-10 5

These functions allow you to send raw X-10 commands to the any one of five defined X-10 controllers. It will typically take 2 of these functions to perform any one X-10 action. You will select the housecode, and either the unitcode or action to be performed against previous unitcodes. In the case of selecting dim or bright, you will also need to input a formula which will evaluate to a number from 1 to 100 as the relative dim/bright percentage.

14.1.3 Macro

You can launch another macro from within this macro. When you do this, the current macros execution is either suspended and the new macro is executed or the new macro is posted to the execution queue to be executed sometime after the current macro completes. You will need to select the macro ID you would like to execute and whether or not to execute the macro immediately or have it posted.

14.1.4 Set Global

You can set the value of a global variable. You will specify the ID of the global variable and a formula to be evaluated. The result of the formula will be assigned to the global variable. All values in global variables are stored in character format. More information on formulas can be found in the formula section.

14.1.5 Set System

You can set the value of certain system variables. You will specify the system variable and a formula whose result will be assigned to the system variable.

14.1.6 Launch App

You can launch another application on the computer. You will specify the full path and executable of the application to be launched as a formula. If the executable is known a design time and not determined dynamically, merely enclose the full path and executable name within single or double quotes.

14.1.7 Switch to App

You can switch to another currently running application and make it the foreground application. This is useful if you want PowerHome to Send Keys to the application to control it. You will specify the type of application switching by selecting either Class Name or Window Name in the value field. Choosing Class Name means to use the class name of the running application for switching purposes. You would typically use this method on applications that do not have a constant window name such as Winamp. Winamp’s window name will change based upon the current song playing, however the class name of an application will not change. To find the class name of an application you need to use a program such as spy. The class name for Winamp by the way is “Winamp v1.x” without the quotes and paying attention to the capitalization. For other applications whose window name remains the same (such as PowerHome), use Window Name for the value field. When you switch to an application from PowerHome and Send Keys to it, you will want to switch back to PowerHome when you are done. The window name for PowerHome is a single space followed by the word PowerHome followed by two spaces. The class name or window name must be entered as a formula within the Send Keys field.

14.1.8 Send Keys

You can simulate keystrokes with this function as well as perform other processing in the evaluation phase of the formula which makes up the Send Keys. Whatever is the current foreground application will receive the keystrokes. If you want to Send Keystrokes to an application other than PowerHome you must first launch it and switch to it. You also have the option to either have the keys sent immediately or be posted to the execution queue. If the Send Keys is posted to the execution queue, the entire formula is posted. After the Send Keys formula is serviced by the execution queue, the Send Keys formula will then be evaluated and the result sent to the Send Keys interpreter. It is important to note that the formula is NOT evaluated before going to the execution queue if you choose post. You will enter the Send Keys formula in the Send Keys field. An explanation of the valid Send Keys will be explained in the Send Keys section.

14.1.9 Dialup Connect

You can launch an already defined dialup connection. You must select either Continue or Wait in the value field. A value of Continue will launch the dialup connection in the background while the macro continues to run. A value of Wait will launch the dialup connection and force the macro to suspend execution until the connection is established. Enter a formula that evaluates to the exact name of the Dialup connection in the Send Keys field.

14.1.10 Disconnect Dialup

You can disconnect a currently connected dialup session. You must select either Continue or Wait in the value field. A value of Continue will close the dialup connection in the background while the macro continues to run. A value of Wait will suspend execution of the macro until the dialup connection is disconnected.

14.1.11 Send Email

You can send email. You must first set the email system variables. This is accomplished by doing a Set System for [EMAILNAME] and a Set System for [EMAILSUBJECT]. Once these system variables are set, you can then do a Send Email. Remember, that system variables are reset everytime PowerHome starts so you will usually set these system variables every time just before you do the Send Email. With this command, you must enter a formula that evaluates to the text of the email message you would like to send in the Send Keys field. This can be as simple as the text that you would like sent surrounded by single or double quotes. This command uses standard MAPI mail calls. You must have a default MAPI mail client declared for this function to work. This will usually be your default mail client such as Outlook Express. Check the options of your mail client and verify that you have a default MAPI client assigned. You can also use the formula function ph_sendemail to send email. This function does not rely upon the [EMAILNAME] and [EMAILSUBJECT] system variables and allows you to declare the recipient and subject in the function header.

14.1.12 Process Email

This command will use your default MAPI client to receive EMail. If you have enabled EMail control in the Setup area of the PowerHome Explorer, it will also process any EMail intended to control PowerHome. This command takes no arguments.

14.1.13 TTS

Allows you to send Text to Speech. If you have enabled text to speech, you can have the PowerHome computer speak to you. You must enter a formula that evaluates to the message you would like to have spoken in the Send Keys field.

14.1.14 Create Timed Event

You can create timed events. The timed event will always have an action type of macro and will always have a frequency of one shot. Select the macro to be assigned to the timed event in the ID field and enter a formula that evaluates to the time the timed event will fire in the Send Keys field. This formula must evaluate to a valid timestamp in the form of yyyy-mm-dd hh:mm:ss or a number representing the number of minutes in the future from the current time.

14.1.15 Jump

The jump command is a basic command that handles if-then-else type processing. Normally, a macro is executed starting at line 1 and every line is executed in order all the way to the last line. The jump command allows you change this normal execution order and jump to a different statement within the current macro. The jump command expects a formula in the Send Keys field that evaluates to a positive or negative integer. If the formula evaluates to 2, the jump command will transfer execution down 2 lines, in effect skipping the line just below the jump command. A negative integer causes control to transfer above the jump line. If the formula does not return an integer, the jump will be ignored and control will transfer to the next line. This allows the user to create their own error handling by placing any error code on the line immediately after the jump. To exit the current macro immediately with no further processing, have the formula return a positive integer beyond the last line of the macro. The important thing to remember with the jump command is that the line jumped to is relative to the current jump line. You cannot jump to an absolute macro line. See the Label and Goto Label commands for additional flow control processing.

14.1.16 Playlist

This command allows you to have a predefined playlist programmed into your CD jukebox. Select the playlist to be programmed in the ID field and either Normal or Randomize in the Value field. A value of Normal will cause the playlist to be programmed in the order of the playlist. A value of Randomize will first randomize the songs in the playlist and then send the infrared commands to the jukebox.

14.1.17 Clear X-10 History

This command will clear the X-10 history variable buffers. You can select whether to clear the Global Buffer, All Buffers, or a particular housecode buffer. When you clear a buffer, you can clear just the first entry which will place a string of five zeroes into the first multi-x entry or clear the entire buffer which will place zeros in the entire multi-x buffer. It will not blank out the times associated with these variables. Versions of PowerHome prior to 1.00.1 had a single global buffer of 6 entries referenced by system variables [X10P0] through [X10P5] and [X10P0TIME] through [X10P5TIME]. Starting with version 1.00.1, PowerHome has expanded the global buffer to 20 entries and added a 20 entry buffer for each housecode. Version 1.01.1 has expanded the system variables to access all 20 of the global buffers commands and times. To access the housecode buffers, use the new functions ph_multix and ph_multixtime.

14.1.18 Delay

Delays macro execution for the specified number of milliseconds. Enter the number of milliseconds of delay you want as a formula in the Send Keys field. There are 1000 milliseconds in 1 second. For a delay of 2 seconds, enter 2000 in the Send Keys field. Delays should be used very sparingly as nothing else will execute while in a delay. If you want an effect such that a macro is launched when a motion sensor is triggered so that a light is turned on for 60 seconds and then back off, don’t use a delay to achieve this. Instead, launch a macro to turn the light on and then either create a timed event for 60 seconds in the future to turn the light off or use the Wait command. This way other processing can continue while PowerHome is counting down to turn the light off.

14.1.19 Trim Event Log

This command is useful for keeping the event log a manageable size. Enter a formula resolving to the number of days you wish to keep in the log in the Send Keys field. A value of 1 in this field will erase all entries in the event and web log except for the entries dated from midnight forward of the current day. If you set this up to execute on a daily basis at 12:05 am and place a value of 2 in the Send Keys field, then you will only have entries for the previous day and all 5 minutes worth of the current day.

14.1.20 Direct SQL

This command should be used with GREAT CARE and only by those who are very familiar with SQL database programming. This command allows you to directly manipulate almost any table within the PowerHome database. Enter a formula that evaluates to a SQL statement (update, insert, delete) into the Send Keys field. The SQL statement is executed and any changes are commited immediately. There is NO chance of a rollback in case of errors.

14.1.21 Message Box

This command allows you to display information to someone who is monitoring the PowerHome program. You do not have to worry about using message boxes and hanging the system until you respond. A message box can optionally timeout by setting the [TIMEOUT] system variable to the number of seconds you wish the box to appear. If you set this value to zero, then the box will never timeout, however the system will still not be hung as the macro which launched the message box is terminated so other items in the execution queue will continue to be executed. Once the message box is closed, the macro that spawned the box will resume where it left off. Select whether or not you want the text displayed in the message box to also be sent to the Text to Speech engine. Select the buttons to appear on the message box in the value field. Next, enter a formula evaluating to the message you wish displayed (and optionally sent to TTS) in the Send Keys field. When the message box is acknowledged or times out, the system variable [MBRET] will contain the number of the button that was pressed or a zero in the case of a timeout. In the case of a button type of YESNOCANCEL, pressing the Yes button will place a 1 in the [MBRET] variable. Pressing No will return a 2, and pressing Cancel will return a 3 and a timeout will return a zero. The message box will have the window title the same as the macro ID.

14.1.22 Input Box

This command operates the same as the Message Box command in respect to time outs and the execution queue continuing to process events while a box is open. This box will obtain input from the user. You can enter a formula that evaluates to a brief message that will appear in the input box (and optionally sent to TTS) in the Send Keys field. Once the input box is opened, the user may enter up to 256 characters in the input field. Input is accepted and the box is closed once the user clicks OK. The user inputted text will then be available in the [INPUTRET] system variable. If any text is currently already in the [INPUTRET] variable when the input box is called, then this text will be pre-entered into the input field of the input box. Remember that system variables are reinitialized everytime PowerHome is started. If you want to have a default value in your input box, you should set the [INPUTRET] variable using the set system command prior to calling the input box command. The input box will have the window title the same as the macro ID.

14.1.23 Label

The label command works in conjunction with the Goto Label command and allows you to label sections of your macros with any name you want so that you can later transfer control to this label without having to use the jump function and count the number of lines between your jumps. You can enter your label in the Send Keys field. This is not a formula and no substitution of any kind is performed. What you type is what you get for a label. When a label command is encountered in normal macro processing, it is merely skipped.

14.1.24 Goto Label

The goto label command allows you to transfer control to a label within the current macro. This can be easier than keeping track of the number of lines between statements like you have to do when using the jump command. It should be noted however that the jump command is more efficient than the Goto Label command but the difference will be negligible. You enter a formula that evaluates to the label that you want to transfer control to in the Send Keys field. Since this field is a formula, you can use a case statement or similar to transfer control to multiple locations. Keep in mind if you have a line labeled abc, when you jump to it with a Goto Label command, you must use “abc” in quotes so that the formula in Goto Label is evaluated as a string. If the label is a number, then you don’t have to place the number in quotes in the formula field of Goto Label. When a goto label command is encountered, execution continues at the line immediately following the matching label.

14.1.25 SQL Select

This command works in conjunction with the Destroy Select command and allows the user to directly query the PowerHome database on the fly. This command should only be used by individuals familiar with SQL programming and intimate with the structure of the PowerHome database. The PowerHome system includes 10 global database retrieval areas. You must select which of the 10 database retrieval areas your SQL Select command will retrieve to. If the database retrieval area is not initialized, it will automatically be created for you. If it already exists and has data, it will be overwritten with the new data. The database retrieval areas and its data utilize system memory and will persist until destroyed with a Destroy Select statement. There are 10 system variables associated with the database retrieval areas

([SQLROWS1] thru [SQLROWS10]) and contain the number of rows currently retrieved in the retrieval areas. There are also several formula functions (such as ph_getdata and ph_finddata) which allow you full manipulation of your retrieved data. When using this command you must select the data retrieval area number and then enter a formula that evaluates to a valid SQL select statement in the Send Keys field.

14.1.26 Destroy Select

This command is used to return memory to the system when you are done with the data in a data retrieval area that was previously retrieved with a SQL Select command. Enter the data retrieval area that you wish to have destroyed.

14.1.27 Wait

This command will suspend execution of the current macro but is different from the Delay command in that it will NOT stop the execution queue from processing waiting commands. The macro is actually terminated but the System Variables [LOCAL1] thru [MACRO10] and [TEMP1] thru [TEMP10] are stored and a timer started. When the timer is up, the macro is reinserted into the execution queue and will resume processing at the statement following the Wait command. Enter a formula that evaluates to the number of seconds you would like to wait in the Send Keys field. Remember that the Delay command works in milliseconds, the Wait command works in seconds.

14.1.28 User Message

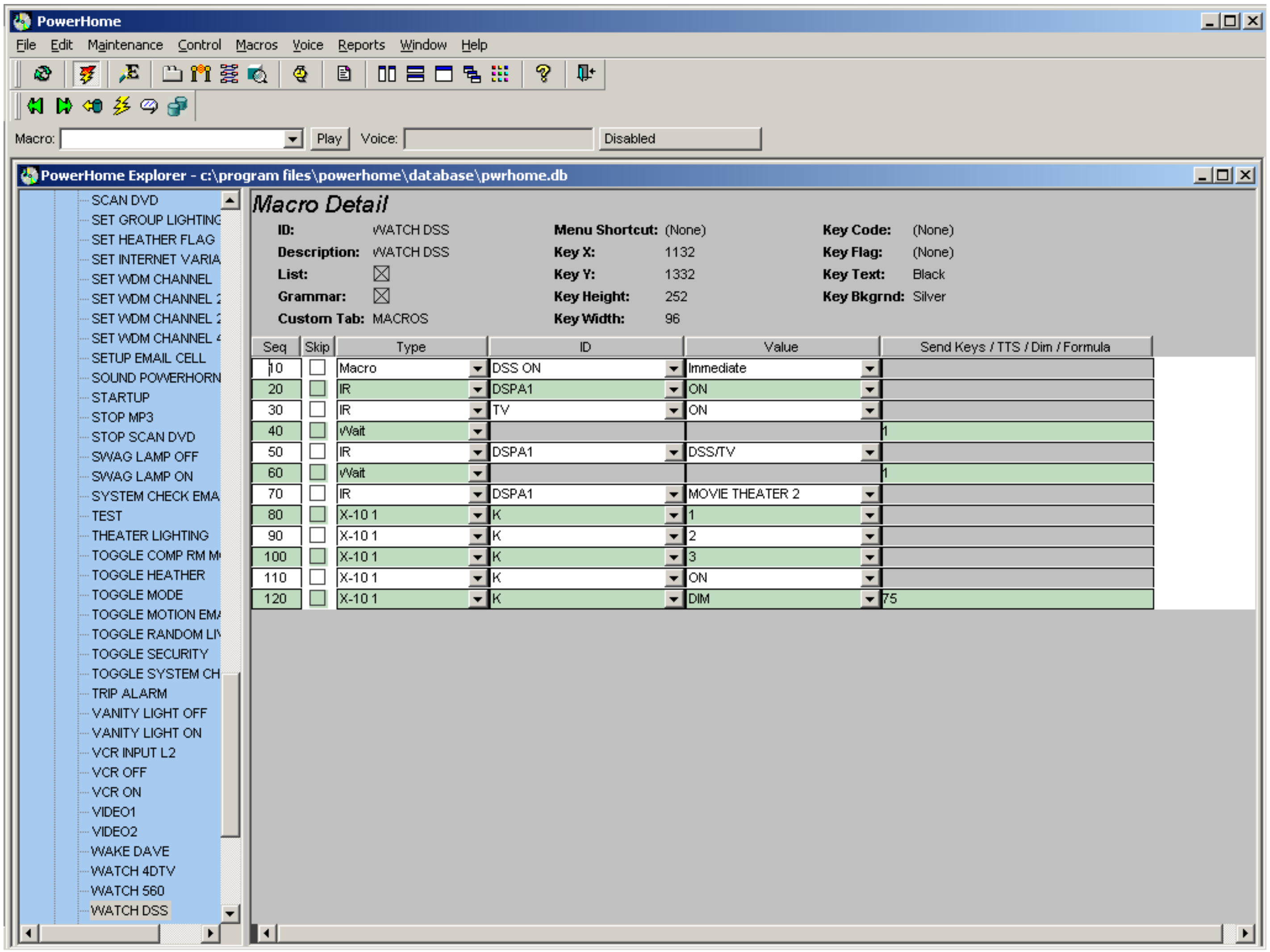
This command will create a user message event within the event log if the logging of user messages is enabled within the PowerHome preferences. Enter a formula that evaluates to the text of the user message within the Send Keys Field.

14.1.29 Comment

This command allows you to add a comment to your macro. You may place the text of your comment within the Send Keys Field. This is not a formula and will not be evaluated. Comment commands are simply skipped during macro execution.

14.1.30 Formula

This command allows you to have a formula entered in the Send Keys field evaluated. The results of the formula are discarded. In this formula you can use any of the defined formula functions.


PowerHome
File Edit Maintenance Control Macros Voice Reports Window Help
Macro: Play Voice: Disabled
PowerHome Explorer - c:\program files\powerhome\database\pwrhome.db

SCAN DVD
SET GROUP LIGHTIN
SET HEATHER FLAG
SET INTERNET VARIA
SET WDM CHANNEL
SET WDM CHANNEL 2
SET WDM CHANNEL 3
SET WDM CHANNEL 4
SETUP EMAIL CELL
SOUND POWERHORN
STARTUP
STOP MP3
STOP SCAN DVD
SWAG LAMP OFF
SWAG LAMP ON
SYSTEM CHECK EMA
TEST
THEATER LIGHTING
TOGGLE COMP RM M
TOGGLE HEATHER
TOGGLE MODE
TOGGLE MOTION EM
TOGGLE RANDOM LI
TOGGLE SECURITY
TOGGLE SYSTEM CH
TRIP ALARM
VANITY LIGHT OFF
VANITY LIGHT ON
VCR INPUT L2
VCR OFF
VCR ON
VIDEO1
VIDEO2
WAKE DAVE
WATCH 4DTV
WATCH 560
WATCH DSS

Macro Detail

ID:	WATCH DSS	**Menu Shortcut:**	(None)	**Key Code:**	(None)
Description:	WATCH DSS	**Key X:**	1132	**Key Flag:**	(None)
List:	☒	**Key Y:**	1332	**Key Text:**	Black
Grammar:	☒	**Key Height:**	252	**Key Bkgrnd:**	Silver
Custom Tab:	MACROS	**Key Width:**	96		

Seq	Skip	Type	ID	Value	Send Keys / TTS / Dim / Formula
10	☐	Macro	DSS ON	Immediate	
20	☐	IR	DSPA1	ON	
30	☐	IR	TV	ON	
40	☐	Wait			1
50	☐	IR	DSPA1	DSS/TV	
60	☐	Wait			1
70	☐	IR	DSPA1	MOVIE THEATER 2	
80	☐	X-10 1	K	1	
90	☐	X-10 1	K	2	
100	☐	X-10 1	K	3	
110	☐	X-10 1	K	ON	
120	☐	X-10 1	K	DIM	75

14.2 Special Macros

There are three special macros that deserve explanation. The first is the startup macro. This macro is no different from any other macro other than its ID. If you create a macro with an ID of STARTUP, this macro will be executed automatically everytime PowerHome is started. If you do not desire this, then simply do not create a macro called STARTUP. The next two special macros are called PLAYSLOTTRACK and PLAYPLAYLIST. These macros have a different set of commands associated with them. These macros are intended for IR control of CD players or CD jukeboxes. If you have a jukebox or CD player that you wish to control via infrared so that you can automatically have songs played for you from a playlist or from selecting off of a CD discs report, you must setup these macros appropriately. We start by describing the PLAYSLOTTRACK macro. There are only 4 commands available for this macro. They are:

14.2.1 IR

- This is exactly like the normal macro IR command and allows you to select an IR key

14.2.2 Delay

- Again, just like the normal macro delay and allows a short delay between commands.

14.2.3 Slot

- This is a dynamic command that is used to send a sequence of numbers to the CD player in reference to a specific slot or disc to play. You must select the ID of the CD player and most importantly, place a formula that evaluates to the correct value in the Send Keys field. If you set up your IR maintenance screen correctly for your CD player (specifically: you setup the numeric keys for your player in order from 1 to 0), then the formula in the Send Keys field should evaluate to the key number for the '1' key for your CD player.

14.2.4 Track

- Similar to the Slot command and used to tell your CD player that were playing a specific numbered track. Must be set up just like the slot command where the formula must evaluate to the key number for the '1' key for your CD player.

Be sure you have entered the appropriate values into the 'Number of digits when programming discs' and 'Number of digits when programming tracks' fields in the PowerHome Setup screen on the CD tab. These two fields are used to control how numbers are sent to the jukebox when programming discs and tracks. Some players may want a set number of digits when programming a disc or track. If your player requires you to enter all tracks as 2 digits such as track 5 as 05, then set the 'Number of digits when programming tracks' field to 2. If your player is indifferent to the number of digits, then set the fields to 0 and only the necessary digits will be sent. My experience is that you will set both fields to 0 with Pioneer jukeboxes and the tracks field to 2 with Sony jukeboxes.

When setting up the PLAYSLOTTRACK macro, you will probably want to make it generic such that any time you select a particular song for playback from the discs report, it will be played whether or not the CD is currently playing a CD. A sample sequence that would be useful for a Pioneer jukebox follows:

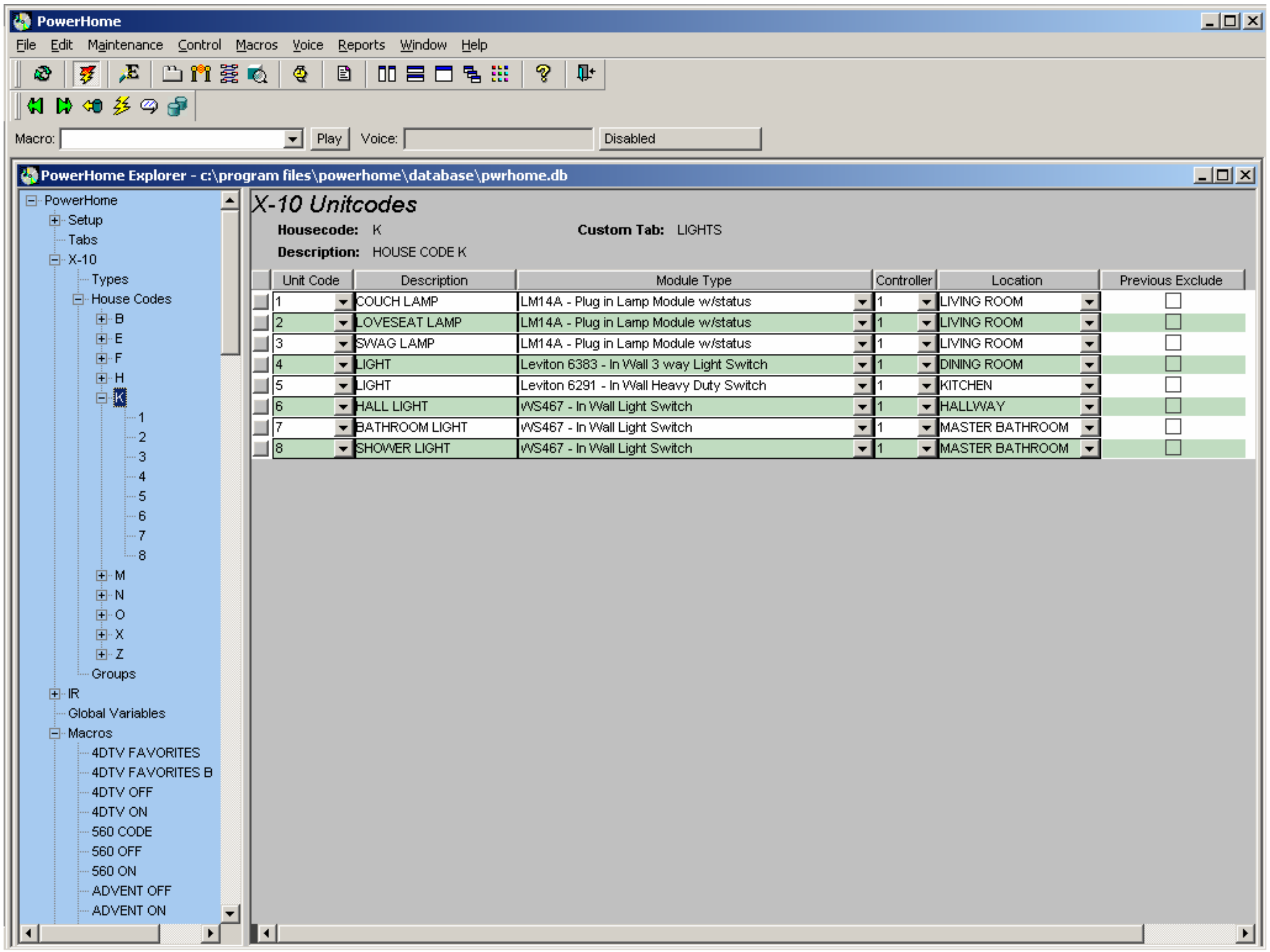
IR	CD	STOP	-	stop any currently playing disc
IR	CD	PGM	-	this and the next line takes the player out of playlist mode.
IR	CD	MODE		
Slot	CD	9	-	sends a slot to be played to the player. The 9 is the key number of the '1' key for the player.
IR	CD	DISC SET		
Track	CD	9	-	sends the track to be played to the player.
IR	CD	TRACK SET		
IR	CD	PLAY	-	play the slot/track we just sent.

You will need to refer to the manual for your particular player to determine how to set up the PLAYSLOTTRACK and PLAYPLAYLIST macros. You might also have to use the remote and experiment with combinations of IR codes to place the CD player in the appropriate mode for using these macros. The PLAYPLAYLIST macro has the same commands that the PLAYSLOTTRACK macro has as well as 2 additional commands to allow for the repetitive nature for programming a playlist. They are 'Start Loop' and 'End Loop' and are used to denote the section of the macro which is looped repetitively while sending down the slot and tracks to be programmed. My Sony jukebox (CDP-CX300) was particularly difficult at getting this to work and I blame a bug in the Sony's ROM. It's been awhile and I can't remember the exact particulars, but had something to do with the number of tracks on a CD. If the first CD in the playlist had less than 10 tracks, then I was unable to program any songs in the playlist with a track greater than 9. I ultimately worked around it by first launching a song on a disc that had more than 10 tracks and programming the playlist while it played. Once the playlist was programmed, I stopped the playback of the song and then started the playback of the playlist. My Pioneer never had a problem like this. Anyhow, refer to the sample database to see how I set up the PLAYPLAYLIST for my Sony and see the macro CD 101 PLAYPLAYLIST for how I set it up with my Pioneer jukebox.

The X-10 screen within the PowerHome Explorer will allow you to define the X-10 devices that you have. The first X-10 screen you should configure is the X-10 Types. The default database is preconfigured with a number of common X-10 devices. Verify that all your device types are defined within this and add any others that are not. This screen will have a profound effect on the X-10 Status Screen and how X-10 status is maintained.

Once you've defined your device types, the next screen to setup would be the housecodes. This screen will allow you to define the housecodes that you are using within your automation configuration.

After you've defined a housecode, you may double-click the record so you can define the unitcodes that you are using within the housecode. Within this screen you can select the Type of the X-10 device along with a description and location. The controller field determines the X-10 controller 1 thru 5 that any buttons defined in the next screen will transmit on. The Previous Exclude determines whether commands received for this housecode/unitcode will be cycled through the previous incoming X-10 buffers (the system variables [X10P0] thru [X10P19] as well as the ph_multix function). If you check this box, the received commands will not go through this buffer. Typically, you will exclude all actual X-10 devices if you are using the previous incoming buffer in a multi-X situation. You would only want incoming X-10 commands intended to fire macros and perform other means of control to go through the buffer.



X-10 Unitcodes

Housecode: K **Custom Tab:** LIGHTS

Description: HOUSE CODE K

Unit Code	Description	Module Type	Controller	Location	Previous Exclude
1	COUCH LAMP	LM14A - Plug in Lamp Module w/status	1	LIVING ROOM	☐
2	LOVESEAT LAMP	LM14A - Plug in Lamp Module w/status	1	LIVING ROOM	☐
3	SWAG LAMP	LM14A - Plug in Lamp Module w/status	1	LIVING ROOM	☐
4	LIGHT	Leviton 6383 - In Wall 3 way Light Switch	1	DINING ROOM	☐
5	LIGHT	Leviton 6291 - In Wall Heavy Duty Switch	1	KITCHEN	☐
6	HALL LIGHT	WS467 - In Wall Light Switch	1	HALLWAY	☐
7	BATHROOM LIGHT	WS467 - In Wall Light Switch	1	MASTER BATHROOM	☐
8	SHOWER LIGHT	WS467 - In Wall Light Switch	1	MASTER BATHROOM	☐

At this point you don't need to go any further unless you want to define some default X-10 buttons for the unitcode. If you would like to define these buttons, double-click the record to enter the X-10 Details screen. You can define a maximum of 4 default buttons per unitcode consisting of 'On', 'Off', 'Dim', and 'Bright'. These buttons will be placed on the custom tab within the Control Center that you assigned to the housecode. This is not the only way to achieve buttons for X-10 control on the Control Center however. You may also define macros or Send Keys buttons that would give much finer control of your X-10 devices.

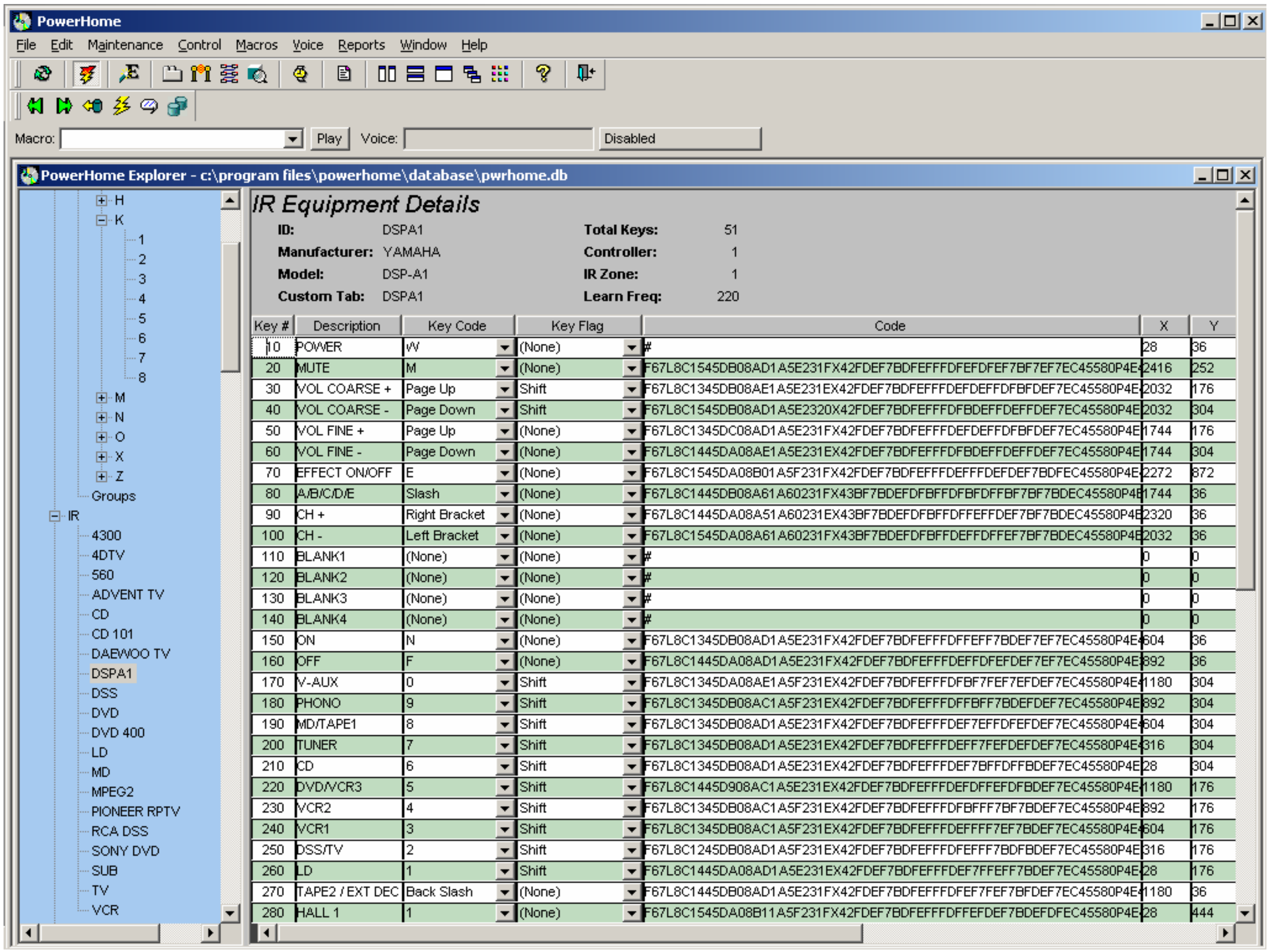
Note: The sending of X-10 commands via macros or formulas are not dependent upon any settings within the X-10 maintenance screens (with the exception of the ph_x10btn function). The receiving of X-10 commands is not dependent upon any of the settings within this area either. Triggers will be fired even if a device is not defined because they are dependent upon the sent and received commands only and not on any actual devices. The settings within this screen are for the X-10 Status Screen and the X10 status system variables. The Types screen will allow you to define what commands an individual device type will respond to.   Defining a housecode / unitcode combination will place an entry into the X-10 Status Screen.

15.1 Previous X-10 Incoming Buffers and Multi-X

The previous X-10 incoming timeout is the amount of seconds between incoming X-10 commands before a command is considered the start of a new multi-X sequence. Multi-X is a term we have coined to describe the ability of PowerHome to easily use incoming sequences of X-10 commands to control functionality. With the typical palm pad controller, you can either control 16 lights / appliances or 32 macros or some combination thereof. With multi-X, you can greatly increase your macro control by pushing combinations of buttons instead of a single button. An example would be to use a palm pad and use unit codes 1 thru 8 for direct control and unit codes 9 thru 16 for multi-X macro control. With the switch on 1-8, you directly control 8 frequently used lights and appliances. With the switch on 9-16, you press combinations of 2 buttons which gives you control of 256 macros. There are several formula functions and parameters within PowerHome that makes the multi-X technology easy to implement. The previous X-10 incoming timeout is one of those. The default value is 3 seconds. Set this value to 0 if you don't want any previous X-10 incoming timeout. A little background on what happens internally will help you understand the importance of this value. X-10 commands are either addresses or functions. When you press a palm pad key, you actually wind up with 2 X-10 transactions. The first transaction is the housecode and unit code. The second transaction is the housecode and the actual action to be carried out (on, off, dim, etc). With a palm pad this is pretty simple. With other controllers (like the CM11A) you can send multiple addresses before sending a function. This allows you to do something like sending addresses for A1, A2, A4 and then send a function like A On so you turn all three lights on simultaneously. An understanding of this X-10 functionality will help with understanding incoming X-10 triggers. When PowerHome receives incoming X-10, if it's an address then the internal X-10 matrix within PowerHome is updated. When a function arrives, any addressed cells within the matrix that match the housecode are processed. The first thing that occurs is the internal X-10 status is updated if applicable. Then the address/function combination is checked to see if it should be excluded from the previous incoming X-10 buffers. The previous incoming X-10 buffers are internal structures that keep track of the last several incoming X-10 address/function combinations and the time they were inserted into the buffer. This is important to keep in mind if an address came in at 10:01:15 and the matching function came in at 10:05:30 then the timestamp for the address/function combination would be 10:05:30. When using a device such as the palm pad, this issue is moot as the palm pad sends the address and functions one after the other. If the address/function combination is not excluded, then all previous incoming X-10 commands in the buffer are shifted by one and the oldest command is lost. The current command moves into the most recent slot. Prior to version 1.00.1, there existed a single multi-X buffer for all housecodes that stored the last 6 incoming commands. These commands were referenced with the system variables [X10P0] for the most recent through [X10P5] for the oldest. The time components for these commands were in system variables [X10P0TIME] through [X10P5TIME]. Starting with version 1.00.1, the multi-x buffers have been expanded such that each housecode has its own buffer storing the last 20 incoming commands as well as a global buffer where the last 20 commands for all housecodes are stored. The original system variables [X10P0] through [X10P5] and [X10P0TIME] through [X10P5TIME] still reference the first six entries of this global buffer. The new formula functions ph_multix and ph_multixtime allows access to the expanded multi-X buffers. With version 1.01.1 the system variables have been expaned so that all 20 entries of the global buffer are accessible via variable substitution. The format the command is stored within the system variables or returned from the multi-X functions are housecode, unitcode, function. An example of K2 off would be "K0203". This value is stored as character data. The first position will always be the housecode. The next two positions will be the unitcode with a prefix of 0 if the unitcode is less than 10. The last two positions will be the function code with a prefix of 0 if the function is less than 10. The most commonly used functions are 02 for on, 03 for off, 04 for dim, and 05 for bright. The time is stored as character data in the format of hh:mm:ss. When a new incoming command is received that is not excluded, its value is moved into the first slot of the appropriate housecode buffer and the first slot of the global buffer along with its appropriate time components. If an incoming X-10 previous timeout is declared (value greater than 0) then the timestamp on the second slot is compared to the timestamp in the newly updated first slot. If the difference in seconds is GREATER than the incoming X-10 previous timeout then the second slots value is set to "00000". The time component is not changed. What this effectively does is blank out the previous command so that multi-X checks will fail.

PowerHome gives you the ability to easily control all of your infrared devices with an appropriate controller. To do this, you'll need to setup your devices in the IR section of the PowerHome Explorer. The first screen is the IR Equipment Screen. You'll need to define an ID for each piece of Equipment. Next define the manufacturer, the model, and the custom tab you want the devices buttons to appear on. Next define the total number of keys that you will define for this device. This may be more than the actual number of keys on your remote. Most IR controllers and PowerHome do not support modifying the IR stream based upon the length of time a button is pressed. This is commonly seen with volume controls and some menu access functions. If you have a button on your remote that activates a menu if briefly pressed and the same button activating a different menu if continuously pressed for a second or two, you'll need to define this button twice. Once being pressed briefly, and once with the button pressed for the amount of time to activate the second menu. The controller is the number of the IR controller 1 thru 5 that you defined in the IR setup. If your controller supports multiple zones, enter the appropriate zone number otherwise leave it at 1. Some controllers require you to set the learn frequency that they learn IR commands. Set this appropriately. See the individual controller information in Appendix A for information regarding the IR Zone and Learn Frequency fields.

After defining your equipment, double-click the record to enter the IR Equipment Details screen. Give a description for each of the keys. You'll probably want to keep this short as this is the text that will appear on the buttons in the Control Center.


PowerHome
File Edit Maintenance Control Macros Voice Reports Window Help
Macro: Play Voice: Disabled
PowerHome Explorer - c:\program files\powerhome\database\pwrhome.db
H
K
1
2
3
4
5
6
7
8
M
N
O
X
Z
Groups
IR
4300
4DTV
560
ADVENT TV
CD
CD 101
DAEWOO TV
DSPA1
DSS
DVD
DVD 400
LD
MD
MPEG2
PIONEER RPTV
RCA DSS
SONY DVD
SUB
TV
VCR

IR Equipment Details

ID:	DSPA1	**Total Keys:**	51
Manufacturer:	YAMAHA	**Controller:**	1
Model:	DSP-A1	**IR Zone:**	1
Custom Tab:	DSPA1	**Learn Freq:**	220

Key #	Description	Key Code	Key Flag	Code	X	Y
10	POWER	W	(None)	#	28	36
20	MUTE	M	(None)	F67L8C1545DB08AD1A5E231FX42FDEF7BDFEFFFDFEFDFEF7BF7EF7EC45580P4E	2416	252
30	VOL COARSE +	Page Up	Shift	F67L8C1345DB08AE1A5E231EX42FDEF7BDFEFFFDEFDEFFDFBFDEF7EC45580P4E	2032	176
40	VOL COARSE -	Page Down	Shift	F67L8C1545DB08AD1A5E2320X42FDEF7BDFEFFFDFBDEFFDEFFDEF7EC45580P4E	2032	304
50	VOL FINE +	Page Up	(None)	F67L8C1345DC08AD1A5E231FX42FDEF7BDFEFFFDEFDEFFDFBFDEF7EC45580P4E	1744	176
60	VOL FINE -	Page Down	(None)	F67L8C1445DA08AE1A5E231EX42FDEF7BDFEFFFDFBDEFFDEFFDEF7EC45580P4E	1744	304
70	EFFECT ON/OFF	E	(None)	F67L8C1545DA08B01A5F231FX42FDEF7BDFEFFFDEFFFDEFDEF7BDFEC45580P4E	2272	872
80	A/B/C/D/E	Slash	(None)	F67L8C1445DB08A61A60231FX43BF7BDEFDFBFFDFBFDFFBF7BF7BDEC45580P4E	1744	36
90	CH +	Right Bracket	(None)	F67L8C1445DA08A51A60231EX43BF7BDEFDFBFFDFFEFFDEF7BF7BDEC45580P4E	2320	36
100	CH -	Left Bracket	(None)	F67L8C1545DA08A61A60231FX43BF7BDEFDFBFFDEFFDFFEF7BF7BDEC45580P4E	2032	36
110	BLANK1	(None)	(None)	#	0	0
120	BLANK2	(None)	(None)	#	0	0
130	BLANK3	(None)	(None)	#	0	0
140	BLANK4	(None)	(None)	#	0	0
150	ON	N	(None)	F67L8C1345DB08AD1A5E231FX42FDEF7BDFEFFFDFFEFF7BDEF7EF7EC45580P4E	604	36
160	OFF	F	(None)	F67L8C1445DA08AD1A5E231FX42FDEF7BDFEFFFDEFFDFEFDEF7EF7EC45580P4E	892	36
170	V-AUX	0	Shift	F67L8C1345DA08AE1A5F231EX42FDEF7BDFEFFFDFBF7FEF7EFDEF7EC45580P4E	1180	304
180	PHONO	9	Shift	F67L8C1345DB08AC1A5F231EX42FDEF7BDFEFFFDFFBFF7BDEFDEF7EC45580P4E	892	304
190	MD/TAPE1	8	Shift	F67L8C1345DB08AD1A5E231FX42FDEF7BDFEFFFDEF7EFFDFEFDEF7EC45580P4E	604	304
200	TUNER	7	Shift	F67L8C1345DB08AD1A5E231EX42FDEF7BDFEFFFDEFF7FEFDEFDEF7EC45580P4E	316	304
210	CD	6	Shift	F67L8C1345DB08AD1A5E231EX42FDEF7BDFEFFFDEF7BFFDFFBDEF7EC45580P4E	28	304
220	DVD/VCR3	5	Shift	F67L8C1445D908AC1A5E231EX42FDEF7BDFEFFFDEFDFFEFDFBDEF7EC45580P4E	1180	176
230	VCR2	4	Shift	F67L8C1345DB08AC1A5F231EX42FDEF7BDFEFFFDFBFFF7BF7BDEF7EC45580P4E	892	176
240	VCR1	3	Shift	F67L8C1345DB08AC1A5F231EX42FDEF7BDFEFFFDEFFFF7EF7BDEF7EC45580P4E	604	176
250	DSS/TV	2	Shift	F67L8C1245DB08AD1A5F231EX42FDEF7BDFEFFFDFEFFF7BDFBDEF7EC45580P4E	316	176
260	LD	1	Shift	F67L8C1445DA08AD1A5E231EX42FDEF7BDFEFFFDEF7FFEFF7BDEF7EC45580P4E	28	176
270	TAPE2 / EXT DEC	Back Slash	(None)	F67L8C1445DB08AD1A5E231FX42FDEF7BDFEFFFDFEF7FEF7BFDEF7EC45580P4E	1180	36
280	HALL 1	1	(None)	F67L8C1545DA08B11A5F231FX42FDEF7BDFEFFFDFFEFDEF7BDEFDFEC45580P4E	28	444

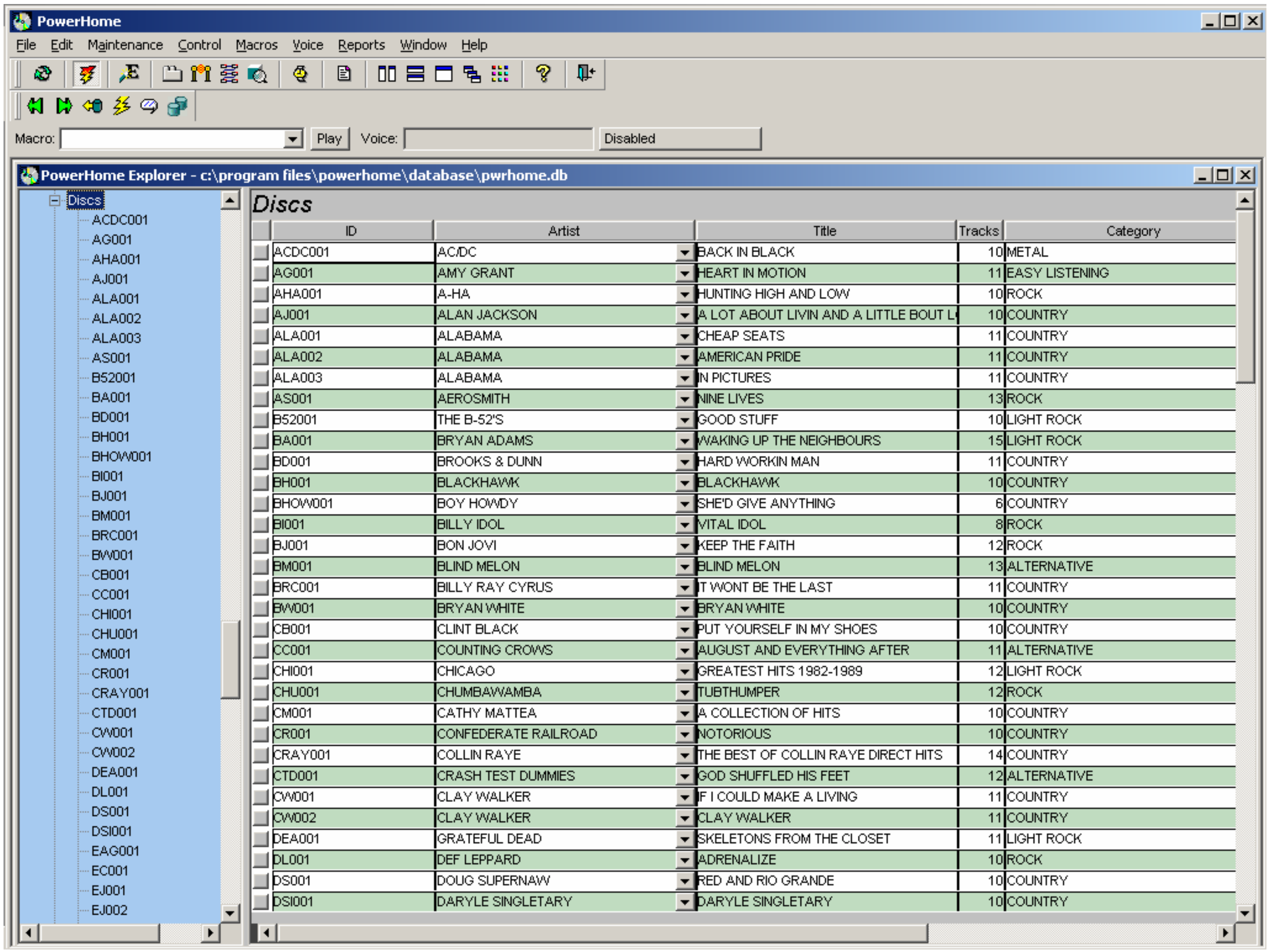
The order of the keys is unimportant except when it comes to the number keys. If the device you are learning is a CD player or jukebox and you want to be able to program playlists or select a disc for play directly from a report, you'll have to define the numeric keys all together in a block. You should do this starting at 1, continuing thru 9, and ending with 0.

Next define the key codes that will trigger the sending of the infrared command. The code field contains the information about the Infrared code. See the controller information in Appendix A for information on this field.

After you’ve defined your individual buttons, you'll want to learn the IR commands. You can do this one at a time by right-clicking on the row you want to learn and selecting "Record" or you can do them all at once by selecting "Auto Learn". Once you’ve learned the IR codes for your equipment, you'll be able to send them from macros and formulas as well as execute them straight from the Control Center.

Discs

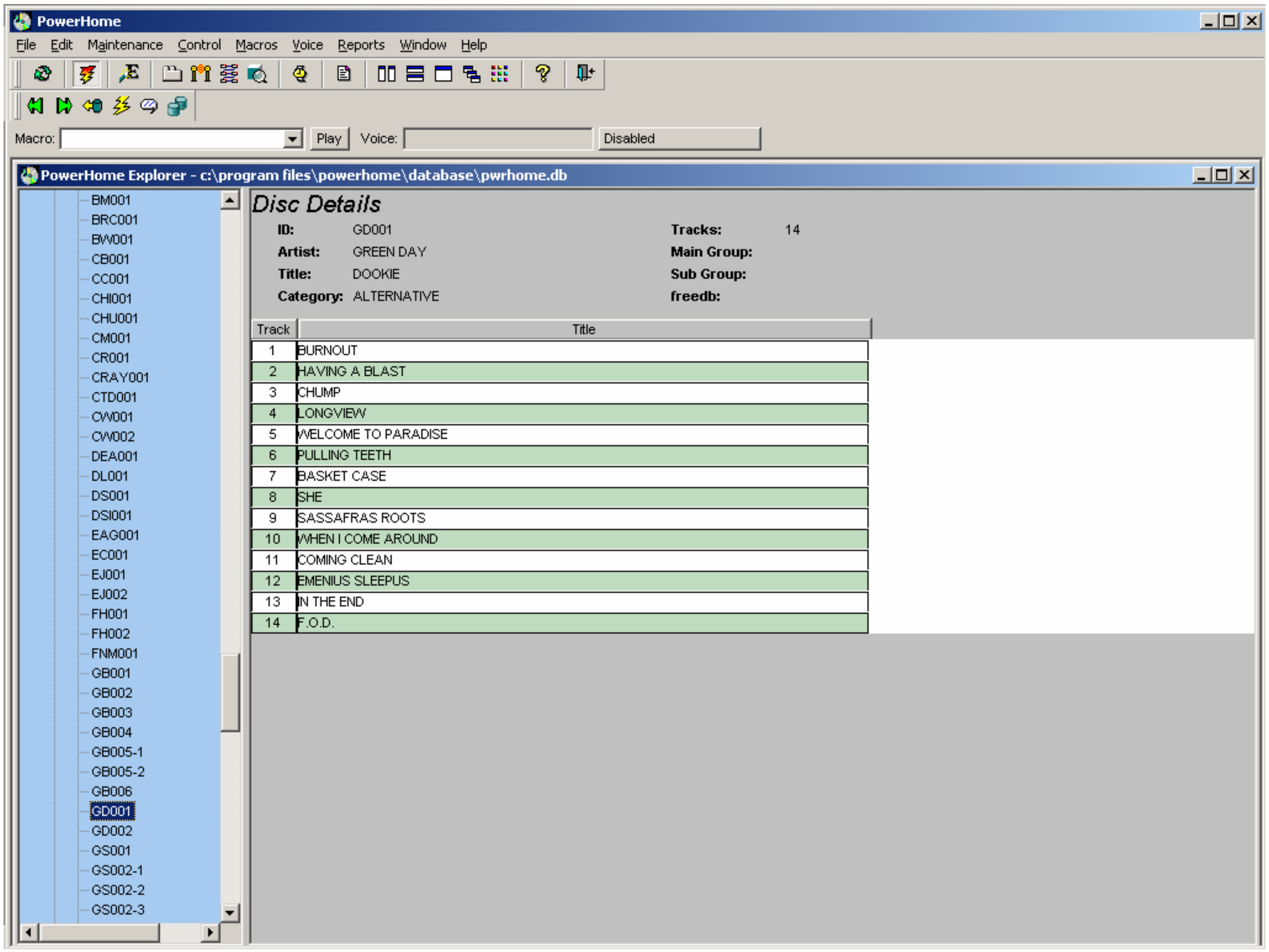
The discs screen in the PowerHome Explorer allows you to track your CD collection and is the basis of populating the slots table and building playlists for CD jukeboxes. To enter a disc, you first need to give it a unique ID. This can be simply a number or some short hand method of identifying the disc.



Discs

ID	Artist	Title	Tracks	Category
ACDC001	AC/DC	BACK IN BLACK	10	METAL
AG001	AMY GRANT	HEART IN MOTION	11	EASY LISTENING
AHA001	A-HA	HUNTING HIGH AND LOW	10	ROCK
AJ001	ALAN JACKSON	A LOT ABOUT LIVIN AND A LITTLE BOUT L	10	COUNTRY
ALA001	ALABAMA	CHEAP SEATS	11	COUNTRY
ALA002	ALABAMA	AMERICAN PRIDE	11	COUNTRY
ALA003	ALABAMA	IN PICTURES	11	COUNTRY
AS001	AEROSMITH	NINE LIVES	13	ROCK
B52001	THE B-52'S	GOOD STUFF	10	LIGHT ROCK
BA001	BRYAN ADAMS	WAKING UP THE NEIGHBOURS	15	LIGHT ROCK
BD001	BROOKS & DUNN	HARD WORKIN MAN	11	COUNTRY
BH001	BLACKHAWK	BLACKHAWK	10	COUNTRY
BHOW001	BOY HOWDY	SHE'D GIVE ANYTHING	6	COUNTRY
BI001	BILLY IDOL	VITAL IDOL	8	ROCK
BJ001	BON JOVI	KEEP THE FAITH	12	ROCK
BM001	BLIND MELON	BLIND MELON	13	ALTERNATIVE
BRC001	BILLY RAY CYRUS	IT WONT BE THE LAST	11	COUNTRY
BW001	BRYAN WHITE	BRYAN WHITE	10	COUNTRY
CB001	CLINT BLACK	PUT YOURSELF IN MY SHOES	10	COUNTRY
CC001	COUNTING CROWS	AUGUST AND EVERYTHING AFTER	11	ALTERNATIVE
CHI001	CHICAGO	GREATEST HITS 1982-1989	12	LIGHT ROCK
CHU001	CHUMBAWAMBA	TUBTHUMPER	12	ROCK
CM001	CATHY MATTEA	A COLLECTION OF HITS	10	COUNTRY
CR001	CONFEDERATE RAILROAD	NOTORIOUS	10	COUNTRY
CRAY001	COLLIN RAYE	THE BEST OF COLLIN RAYE DIRECT HITS	14	COUNTRY
CTD001	CRASH TEST DUMMIES	GOD SHUFFLED HIS FEET	12	ALTERNATIVE
CW001	CLAY WALKER	IF I COULD MAKE A LIVING	11	COUNTRY
CW002	CLAY WALKER	CLAY WALKER	11	COUNTRY
DEA001	GRATEFUL DEAD	SKELETONS FROM THE CLOSET	11	LIGHT ROCK
DL001	DEF LEPPARD	ADRENALIZE	10	ROCK
DS001	DOUG SUPERNAW	RED AND RIO GRANDE	10	COUNTRY
DSI001	DARYLE SINGLETARY	DARYLE SINGLETARY	10	COUNTRY

Once you have entered the ID, you may continue to enter the rest of the fields or if you have an internet connection and a CD-ROM drive in your computer and have filled out the appropriate freedb fields in the PowerHome Setup screen, you can automatically get the rest of the disc and track information entered into the database for you. This will be a huge timesaver. To use this feature place the audio CD you are currently entering into your 1st CD-ROM drive, right click on the disc you are entering and select freedb from the popup menu. The PowerHome program will then access the freedb database and retrieve the disc information for you. In the event that your disc is not in the freedb database or you are not equipped to access this service, you can fill in the appropriate fields manually. The artist field is self explanatory. The title field should have the title of the disc entered. Next, type in the total tracks on the CD. The category field allows you to define your music into categories of your choosing. The main group and sub group fields are numeric fields that allow you to further group or categorize your CD's and are useful when querying your CD's in the Discs report. Once the disc information is entered, double-click the record to enter the track information.



Disc Details

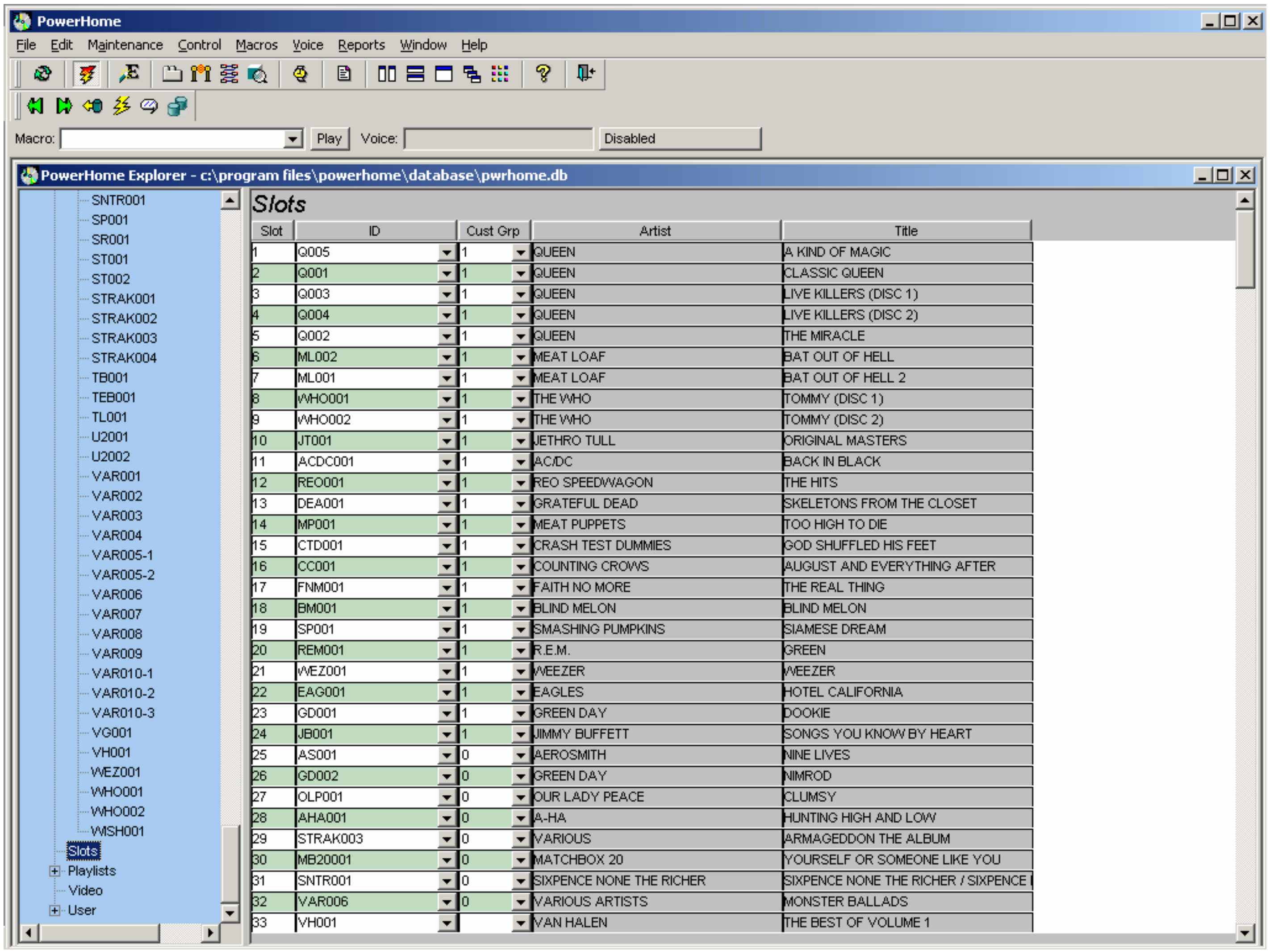
ID:	GD001	**Tracks:**	14
Artist:	GREEN DAY	**Main Group:**	
Title:	DOOKIE	**Sub Group:**	
Category:	ALTERNATIVE	**freedb:**	

Track	Title
1	BURNOUT
2	HAVING A BLAST
3	CHUMP
4	LONGVIEW
5	WELCOME TO PARADISE
6	PULLING TEETH
7	BASKET CASE
8	SHE
9	SASSAFRAS ROOTS
10	WHEN I COME AROUND
11	COMING CLEAN
12	EMENIUS SLEEPUS
13	IN THE END
14	F.O.D.

The appropriate number of track fields will be generated for you based upon the total tracks field. Enter the appropriate track titles in these fields.

Slots

Once you have entered all your discs into the database, you can next set up the slots on your CD jukebox. Be sure you have entered the total number of slots your jukebox supports in the PowerHome Setup screen in the CD section. Select the appropriate disc ID for each slot in your jukebox. The custom group field is not currently utilized so you can leave this field blank. The artist and title fields will not be populated until save and refresh the screen.



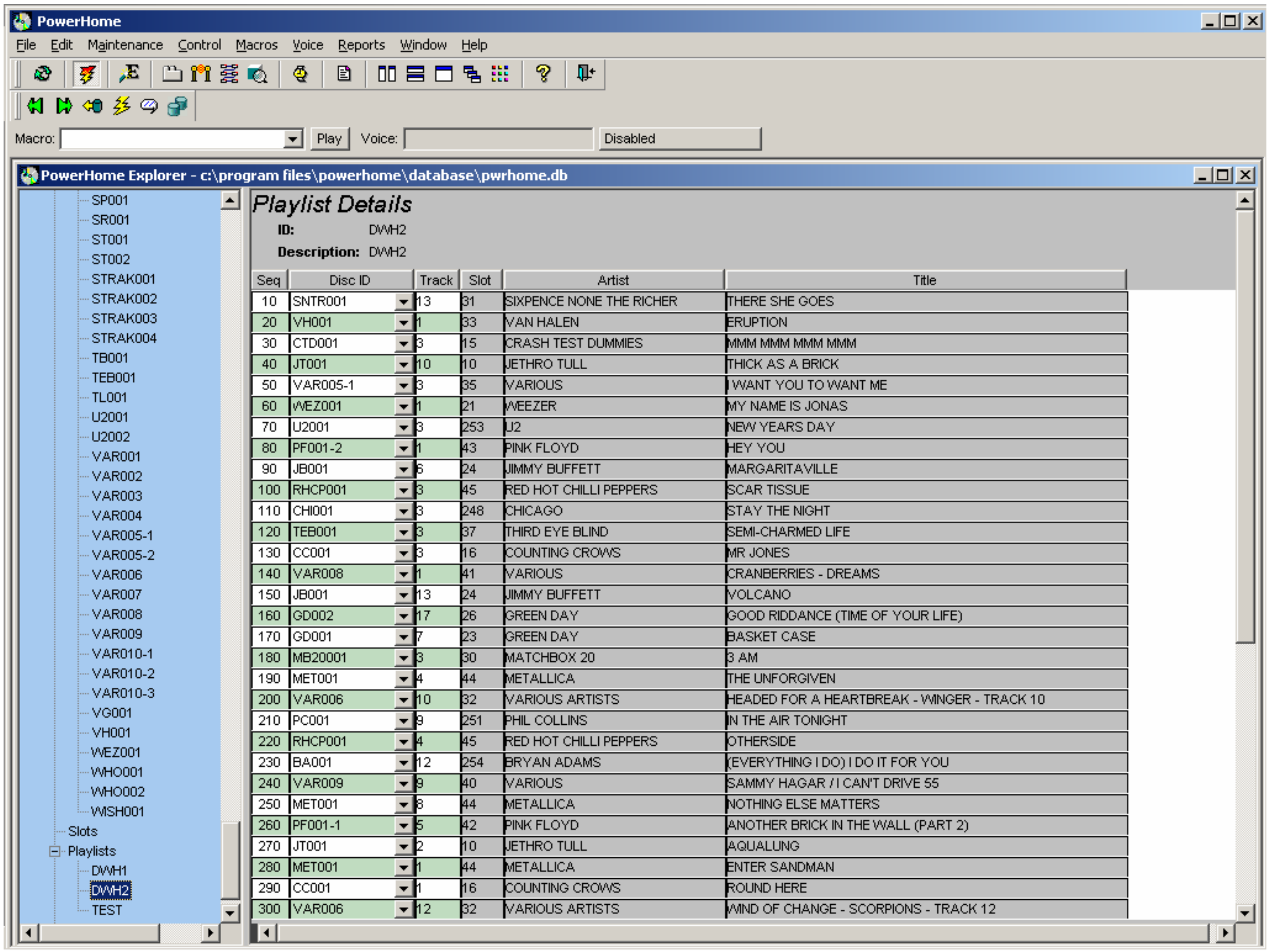
SNTR001
SP001
SR001
ST001
ST002
STRAK001
STRAK002
STRAK003
STRAK004
TB001
TEB001
TL001
U2001
U2002
VAR001
VAR002
VAR003
VAR004
VAR005-1
VAR005-2
VAR006
VAR007
VAR008
VAR009
VAR010-1
VAR010-2
VAR010-3
VG001
VH001
WEZ001
WHO001
WHO002
WSH001
Slots
Playlists
Video
User

Slots

Slot	ID	Cust Grp	Artist	Title
1	Q005	1	QUEEN	A KIND OF MAGIC
2	Q001	1	QUEEN	CLASSIC QUEEN
3	Q003	1	QUEEN	LIVE KILLERS (DISC 1)
4	Q004	1	QUEEN	LIVE KILLERS (DISC 2)
5	Q002	1	QUEEN	THE MIRACLE
6	ML002	1	MEAT LOAF	BAT OUT OF HELL
7	ML001	1	MEAT LOAF	BAT OUT OF HELL 2
8	WHO001	1	THE WHO	TOMMY (DISC 1)
9	WHO002	1	THE WHO	TOMMY (DISC 2)
10	JT001	1	JETHRO TULL	ORIGINAL MASTERS
11	ACDC001	1	AC/DC	BACK IN BLACK
12	REO001	1	REO SPEEDWAGON	THE HITS
13	DEA001	1	GRATEFUL DEAD	SKELETONS FROM THE CLOSET
14	MP001	1	MEAT PUPPETS	TOO HIGH TO DIE
15	CTD001	1	CRASH TEST DUMMIES	GOD SHUFFLED HIS FEET
16	CC001	1	COUNTING CROWS	AUGUST AND EVERYTHING AFTER
17	FNM001	1	FAITH NO MORE	THE REAL THING
18	BM001	1	BLIND MELON	BLIND MELON
19	SP001	1	SMASHING PUMPKINS	SIAMESE DREAM
20	REM001	1	R.E.M.	GREEN
21	WEZ001	1	WEEZER	WEEZER
22	EAG001	1	EAGLES	HOTEL CALIFORNIA
23	GD001	1	GREEN DAY	DOOKIE
24	JB001	1	JIMMY BUFFETT	SONGS YOU KNOW BY HEART
25	AS001	0	AEROSMITH	NINE LIVES
26	GD002	0	GREEN DAY	NIMROD
27	OLP001	0	OUR LADY PEACE	CLUMSY
28	AHA001	0	A-HA	HUNTING HIGH AND LOW
29	STRAK003	0	VARIOUS	ARMAGEDDON THE ALBUM
30	MB20001	0	MATCHBOX 20	YOURSELF OR SOMEONE LIKE YOU
31	SNTR001	0	SIXPENCE NONE THE RICHER	SIXPENCE NONE THE RICHER / SIXPENCE
32	VAR006	0	VARIOUS ARTISTS	MONSTER BALLADS
33	VH001		VAN HALEN	THE BEST OF VOLUME 1

Playlists

Most CD jukeboxes have the ability to be programmed to play a specific sequence of songs. To do this manually is a painstaking process but with the PowerHome playlist capability and IR control, this task becomes a breeze. This screen allows you to organize playlists of your creation for later programming into your CD jukebox. The first thing you must do is create an ID and description for a playlist. You can then double-click the record to get to the Playlist details and select songs for insertion into your playlist. You must enter the disc ID and the track number for each playlist entry. Right-clicking in the playlist detail screen will present you with a menu for inserting, deleting, and organizing your playlist.



Playlist Details

ID: DWH2

Description: DWH2

Seq	Disc ID	Track	Slot	Artist	Title
10	SNTR001	13	31	SIXPENCE NONE THE RICHER	THERE SHE GOES
20	VH001	1	33	VAN HALEN	ERUPTION
30	CTD001	3	15	CRASH TEST DUMMIES	MMM MMM MMM MMM
40	JT001	10	10	JETHRO TULL	THICK AS A BRICK
50	VAR005-1	3	35	VARIOUS	I WANT YOU TO WANT ME
60	WEZ001	1	21	WEEZER	MY NAME IS JONAS
70	U2001	3	253	U2	NEW YEARS DAY
80	PF001-2	1	43	PINK FLOYD	HEY YOU
90	JB001	6	24	JIMMY BUFFETT	MARGARITAVILLE
100	RHCP001	3	45	RED HOT CHILLI PEPPERS	SCAR TISSUE
110	CHI001	3	248	CHICAGO	STAY THE NIGHT
120	TEB001	3	37	THIRD EYE BLIND	SEMI-CHARMED LIFE
130	CC001	3	16	COUNTING CROWS	MR JONES
140	VAR008	1	41	VARIOUS	CRANBERRIES - DREAMS
150	JB001	13	24	JIMMY BUFFETT	VOLCANO
160	GD002	17	26	GREEN DAY	GOOD RIDDANCE (TIME OF YOUR LIFE)
170	GD001	7	23	GREEN DAY	BASKET CASE
180	MB20001	3	30	MATCHBOX 20	3 AM
190	MET001	4	44	METALLICA	THE UNFORGIVEN
200	VAR006	10	32	VARIOUS ARTISTS	HEADED FOR A HEARTBREAK - WINGER - TRACK 10
210	PC001	9	251	PHIL COLLINS	IN THE AIR TONIGHT
220	RHCP001	4	45	RED HOT CHILLI PEPPERS	OTHERSIDE
230	BA001	12	254	BRYAN ADAMS	(EVERYTHING I DO) I DO IT FOR YOU
240	VAR009	9	40	VARIOUS	SAMMY HAGAR / I CAN'T DRIVE 55
250	MET001	8	44	METALLICA	NOTHING ELSE MATTERS
260	PF001-1	5	42	PINK FLOYD	ANOTHER BRICK IN THE WALL (PART 2)
270	JT001	2	10	JETHRO TULL	AQUALUNG
280	MET001	1	44	METALLICA	ENTER SANDMAN
290	CC001	1	16	COUNTING CROWS	ROUND HERE
300	VAR006	12	32	VARIOUS ARTISTS	WIND OF CHANGE - SCORPIONS - TRACK 12

Playlist entry from this screen is tedious however. A quicker way to make your playlist song selections is to go to the menu and under control, select Set Active Playlist. Type the name of the playlist you want to set (the playlist ID DOES NOT have to exist. It will be created if it does not already exist) and press enter. Next go to the reports menu and select discs. Enter any appropriate query parameters or leave them all blank to select all discs. Press the Go button and a printable report of all your discs will be presented on screen. To simply add a song to the current playlist, right-click on the song and select Add to Playlist from the pop-up menu. Keep doing this for all the songs you want in the current playlist. Once done entering songs into your playlist from the discs report, go back into PowerHome Explorer's playlist maintenance and bring up the playlist you just entered. From here, you can use the right-click menu to organize your selections the way you want. Several fields in the PowerHome Setup screen under the CD section pertain directly to playlists. The maximum number of songs in a playlist field should reflect the maximum number of songs your jukebox supports to be programmed to it. This is typically around 30 for most jukeboxes. If the number of songs in your playlist exceeds this number, when programming a playlist into the jukebox, any songs greater than the max will simply be skipped. If you send a playlist to your jukebox with the randomize option set, you will get a different set of songs each time. In order to send a playlist to your jukebox via IR, you must first have set the macros PLAYSLOTTRACK and PLAYPLAYLIST to appropriate settings. These macros are also dependant upon settings in the PowerHome Setup screen CD section concerning the number of digits when programming and must be set properly for playlist programming to work.

Video

The video maintenance screen in PowerHome Explorer allows you to maintain a database of your video collection such as on DVD, LD, or VCR. No reports have been developed yet for this database and future enhancements to this screen and reporting are planned.

Voice

This maintenance screen in the PowerHome Explorer allows you to maintain the grammars used by the voice recognition abilities of PowerHome. As with most of the maintenance screens, you must enter an ID for the grammar you are working with.

After entering the ID, the first thing you need to determine for your grammar is whether or not you want Persistent Send Keys actions. You will see this in the included SYSTEM grammar. When this box is checked, if a SHIFT, CONTROL, or ALT key is sent in the Send Keys formula, it will persist until the next Send Keys command that is not a SHIFT, CONTROL, or ALT key. This is necessary in the SYSTEM grammar where a single voice command is mapped to each key. To simulate a control-k keypress using this grammar, you would say CONTROL KEY which would be recognized as a single voice command and since the persistent box is checked, the control key stays pressed. You would next say KILO which is the syntax for the k key. This command would be recognized and a control-k keystroke would be simulated on the system. Without the persistent keys box selected, the first command would have pressed and released the control key and the second command would have pressed and released the k key.

After selecting whether or not you want persistent keys, select whether or not you want the GLOBAL grammar included. When this box is checked, this grammar will also include the complete command list of the GLOBAL grammar (the global grammar is simply the grammar with the id GLOBAL).

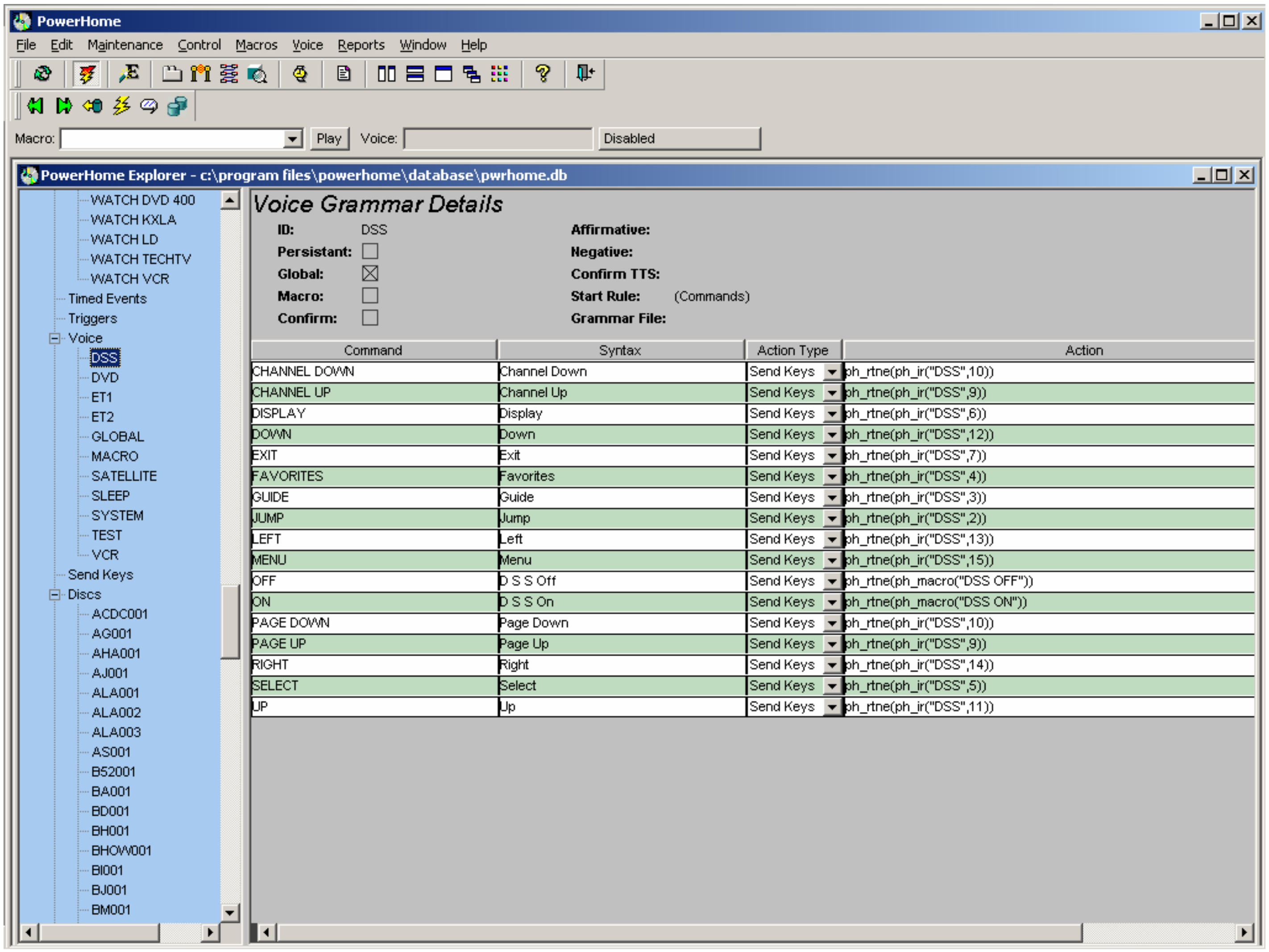
Next, select whether or not this grammar will include macros. If you include macros, any macros where you have selected to include in grammar will also be included in this grammar. When you include macros, the command as well as the syntax will be the macro ID. The action for a macro will be to simply execute the macro.

Next, select whether or not you want a confirmation. If you want a confirmation, then whenever PowerHome recognizes a voice command, it will use TTS and speak the phrase in the confirm TTS box. A simple syntax for this would be "[COMMAND], is this correct?" without the quotes. As you can see, there is a system variable [COMMAND]. This system variable is ONLY valid within this context and will be replaced by the command (not the syntax) that was just recognized. At this point, the voice recognition grammar is changed and is looking for either your affirmative or negative syntax. If you speak the affirmative syntax, the action for the command will be executed. If you speak the negative syntax, the action will not be executed. In either case, once a proper affirmative or negative syntax is recognized, the grammar will be changed back to the previous grammar.

The start rule will default to "(Commands)". Generally you would not want to change this unless you are familiar with Microsoft's syntax for SAPI grammars.

The last field is the grammar file field. You can declare a file to be added on to the computer generated grammar file. This file will be added to the end of the grammar syntax created by PowerHome. Refer to Microsoft's documentation on SAPI grammars before trying to use this feature.

Once the grammar header is entered, double-click the record to edit the grammar detail. From here you can right-click to insert or delete detail records. Each detail record equates to a voice command within the grammar. For each detail record, determine and enter a unique command, then enter the syntax for that command. In most cases this will probably be the command itself, but you can refer to the Microsoft grammar syntax to allow multiple combinations of spoken syntax to activate the command. A very quick example of what can be done follows. If you have a command called LISTEN MUSIC you can have the syntax be "listen music". In this situation the voice recognition will only respond to the exact syntax and won't execute the command unless the phrase "listen music" is spoken exactly. If you wanted to support the phrase "listen to music" as well as the phrase "listen music", you could use the syntax "listen [opt] to music"; the [opt] syntax tells the parser to optionally accept the word immediately following the [opt] which in this case is the word "to". You could also use a syntax "[opt] (JunkBegin) Listen [opt] to music". In this case the accepted syntax could be "listen to music", "listen music", or "please listen to music". The (JunkBegin) syntax equates to common words that are commonly spoken at the beginning of a sentence and the [opt] prior is so that any junk words are optional and not a requirement. The full voice recognition syntax is documented in the Microsoft voice recognition grammar documentation. After entering the syntax, next enter the action. The action MUST be a valid Send Keys formula. If you want the action to be a macro, simply use the formula: ph_rtne(ph_macro("MACROID")). The ph_rtne function will translate the number 0 returned by the ph_macro function into an empty string so that no keystrokes will actually be simulated by the Send Keys interpreter.



Voice Grammar Details

ID: DSS
Persistant: ☐
Global: ☒
Macro: ☐
Confirm: ☐
Affirmative:
Negative:
Confirm TTS:
Start Rule: (Commands)
Grammar File:

Command	Syntax	Action Type	Action
CHANNEL DOWN	Channel Down	Send Keys	ph_rtne(ph_ir("DSS",10))
CHANNEL UP	Channel Up	Send Keys	ph_rtne(ph_ir("DSS",9))
DISPLAY	Display	Send Keys	ph_rtne(ph_ir("DSS",6))
DOWN	Down	Send Keys	ph_rtne(ph_ir("DSS",12))
EXIT	Exit	Send Keys	ph_rtne(ph_ir("DSS",7))
FAVORITES	Favorites	Send Keys	ph_rtne(ph_ir("DSS",4))
GUIDE	Guide	Send Keys	ph_rtne(ph_ir("DSS",3))
JUMP	Jump	Send Keys	ph_rtne(ph_ir("DSS",2))
LEFT	Left	Send Keys	ph_rtne(ph_ir("DSS",13))
MENU	Menu	Send Keys	ph_rtne(ph_ir("DSS",15))
OFF	D S S Off	Send Keys	ph_rtne(ph_macro("DSS OFF"))
ON	D S S On	Send Keys	ph_rtne(ph_macro("DSS ON"))
PAGE DOWN	Page Down	Send Keys	ph_rtne(ph_ir("DSS",10))
PAGE UP	Page Up	Send Keys	ph_rtne(ph_ir("DSS",9))
RIGHT	Right	Send Keys	ph_rtne(ph_ir("DSS",14))
SELECT	Select	Send Keys	ph_rtne(ph_ir("DSS",5))
UP	Up	Send Keys	ph_rtne(ph_ir("DSS",11))

You can also use Microsoft Agent syntax for your grammar commands. Just prefix the syntax with a ~ tilde character. The MSAgent syntax allows you to build powerful, full featured grammars but will usually require you to do a little extra work in your action to process them properly. This syntax supports additional characters for creating a grammar. They are the open and close square brackets [ ], the vertical bar |, left and right parenthesis ( ), ellipsis ..., the asterisk *, and the plus sign +. The square brackets [ ] denote words or groups that are optional. The parenthesis ( ) is used to group words. The vertical bar is the separator for alternative words. The asterisk * denotes 0 or more instances. The plus + denotes 1 or more instances. The ellipsis is a wild card standing for any group of words. It is best to describe this syntax by demonstrating it with some examples.

Syntax: ~[please] change [the] channel
Result: You could speak any of the following phrases: change channel, please change channel, change the channel, and please change the channel. This is because the words "please" and "the" are optional words. The square brackets indicate this. The words "change" and "channel" are required words and must be spoken in order for the phrase to be recognized.

Syntax: ~change the channel to (1|2|3|4|5|6|7|8|9)
Result: Any of the phrases that start with "change the channel to" followed by a number from 1 to 9 will be recognized. The vertical bar separates alternate words. Note that alternate words must be enclosed with parenthesis.

Syntax: ~[...] change channel [to] (0|1|2|3|4|5|6|7|8|9)+
Result: The ellipsis will match any combination of words. Since it is enclosed in square brackets, these words are optional. This means that you could start your phrase with "Please change channel" or "I want to change channel". The word "to" is also enclosed in square brackets so it is optional also. The vertical bars and parenthesis denote a group of alternate words which may be spoken. The plus sign at the end of the group

denotes that you must speak at least one word from the group and can speak as many words as you want. This allows you to say a phrase like "Please change channel to 5 2 5" or "Change channel to 2 3".

When working with MSAgent grammar syntax, remember that you can get a number of phrases recognized within a single detail line. If you have multiple detail lines each using MSAgent syntax, your grammar can be quite large and could be ambiguous if you're not careful with your syntax.

In the last example above, a large number of phrases will be recognized. Each recognized phrase will trigger the same action but you would want to really perform different actions based upon what was recognized. In this case, you would be best to have your action call a macro and parse what was said. When a voice command is recognized and an action called, the [TEMP1] and [TEMP2] system variables will have the command and phrase populated within them respectively. A macro called from within the action will then have access to these variables and would be able to parse the contents of the phrase in the [TEMP2] variable and thus take appropriate action.

When working with grammars, it's much better to have a number of smaller grammars than fewer large grammars. Your recognition accuracy will be much better if the software only has to recognize between a short list rather than a large list. The PowerHome program allows you to quickly and easily switch between grammars using the appropriate formula functions. You also want to stay away from short, one syllable syntax commands as these can be easily confused between one another. You will get much greater recognition accuracy if your syntax commands are two or three words and each of the commands within a grammar are as different from one another as possible. Having commands "move up" and "move down" may not be enough difference for the system to differentiate between these two commands. Even though each command consists of two words, each command contains the word "move" so that the only difference between the commands is a one syllable word. You would get better recognition accuracy by changing the commands to "shift up" and "move down". While this becomes a pain, it is one of the flaws of voice recognition. In any event, the faster your machine and the more RAM you have, the faster your voice recognition will be and the more accurate.

21.1 VR Setup

When initially setting up voice recognition, you should go through the various voice recognition setups on the Voice menu item. When you do voice recognition training, the training will be against the current voice user. The first item you should work with is the Microphone Setup. This will set your system for the proper levels for voice recognition. Next, you'll want to do some training. Just doing two or three training sessions will greatly improve your recognition accuracy. After properly training the system, if you still aren't getting the desired accuracy, you may need to adjust the settings under Recognition Setup.

There are six settings which control the internal workings of voice recognition within the VR Setup screen in the PowerHome Explorer. Generally the default settings are adequate, but if you want to change them, they are explained in detail below.

21.1.1 Automatic Gain

Automatic gain controls the state of the automatic gain for the incoming audio stream. A value of 0 indicates that automatic gain is disabled. A value of 100 indicates that the voice recognition engine always adjusts the gain, so that if the next utterance is spoken at the same level, the gain is set perfectly. A value between 0 and 100 moderates the automatic adjustments on a linear scale. For example, an automatic gain value of 50 adjusts the gain to half the extent that the engine otherwise would. When automatic gain is enabled, the speech recognition engine may adjust the level at the end of an utterance and increase or decrease the gain.

21.1.2 Complete Timeout

Complete timeout is the number of milliseconds that the voice recognition engine waits before regarding a phrase as complete after the user has stopped speaking. For example, if complete timeout is 500, a user speaking "Send mail to Fred" would see results 0.5 seconds after finishing the phrase. Complete timeout should be shorter than Incomplete timeout.

21.1.3 Incomplete Timeout

Incomplete timeout is the number of milliseconds that the voice recognition engine waits before discarding an incomplete phrase because the user has stopped speaking. For example, if incomplete timeout is 2000, a user speaking "Send mail to " could pause for 2 seconds before the engine would assume that the user has stopped speaking the phrase. Incomplete timeout should be longer than complete timeout.

21.1.4 Energy Floor

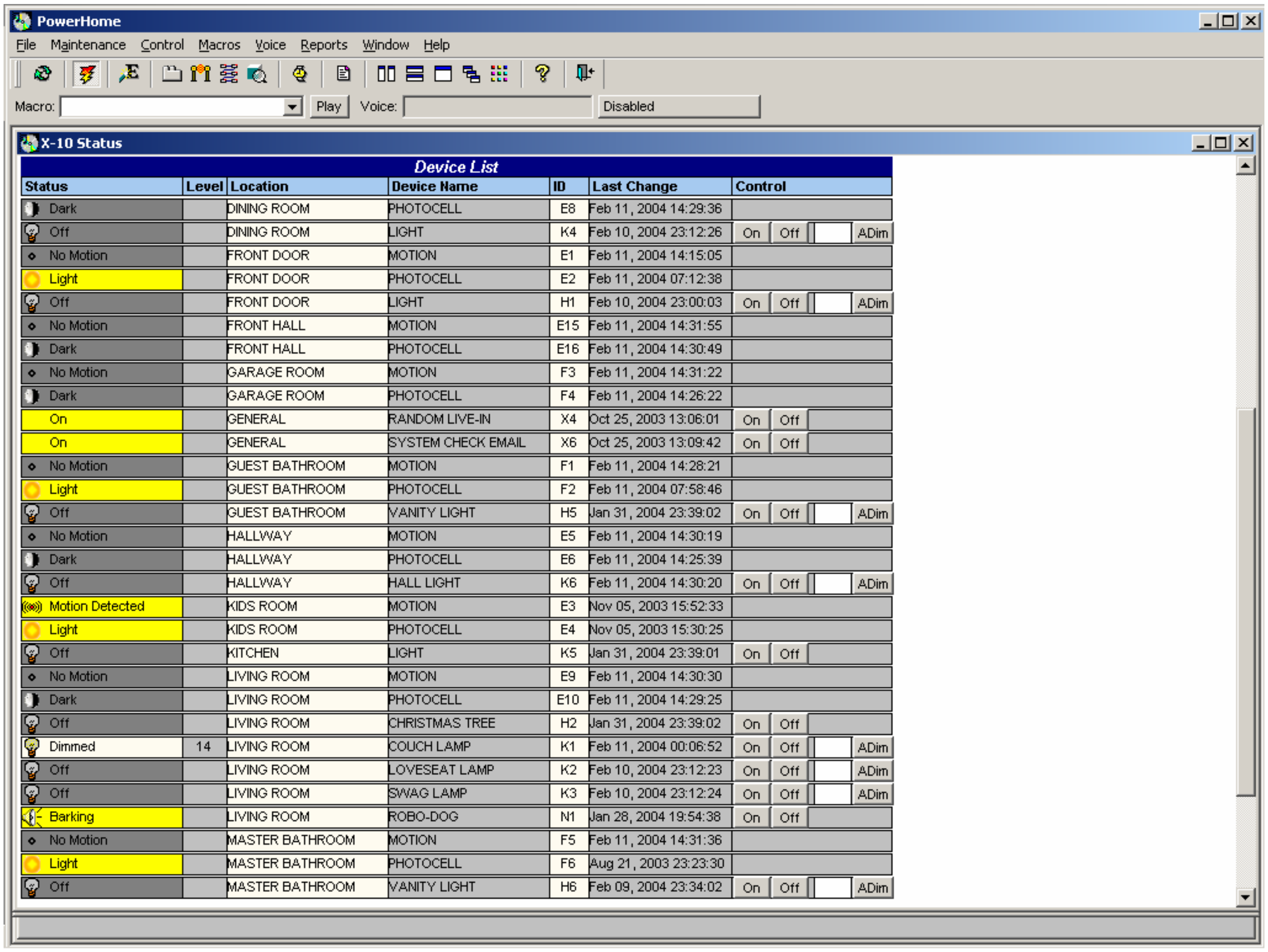
Energy floor is the value of the noise floor in negative decibels. The noise floor is the noise value in the signal-to-noise ratio (SNR) for a particular environment. In general, the higher the noise value in the ratio, the more sensitive the speech recognition engine is to background noise. For example, a quiet office would have a high SNR (low floor), whereas a noisy factory would have a low SNR (high floor).

21.1.5 Real Time

The real time setting is the percentage of processor time that the engine expects to use during constant speech. If real time is 100, the engine takes one full minute of processor time to process one minute of speech. If real time is 50, the engine takes 30 seconds of processor time to process the same minute of speech.

X-10 Status Screen

The X-10 Status Screen is the second of two main screens in PowerHome, the other being the Control Center. You can always bring this screen up and on top by pressing the "Alt+F6" key combination. Within this screen you'll be able to see the current status of all the X-10 devices you've defined within the PowerHome Explorer as well as control the non status-only devices. Even though PowerHome responds to and tracks all X-10 commands, if you don’t define a housecode / unitcode combination within the X-10 maintenance screens, you wont get a corresponding item in the Status Screen. This screen is updated in real time.

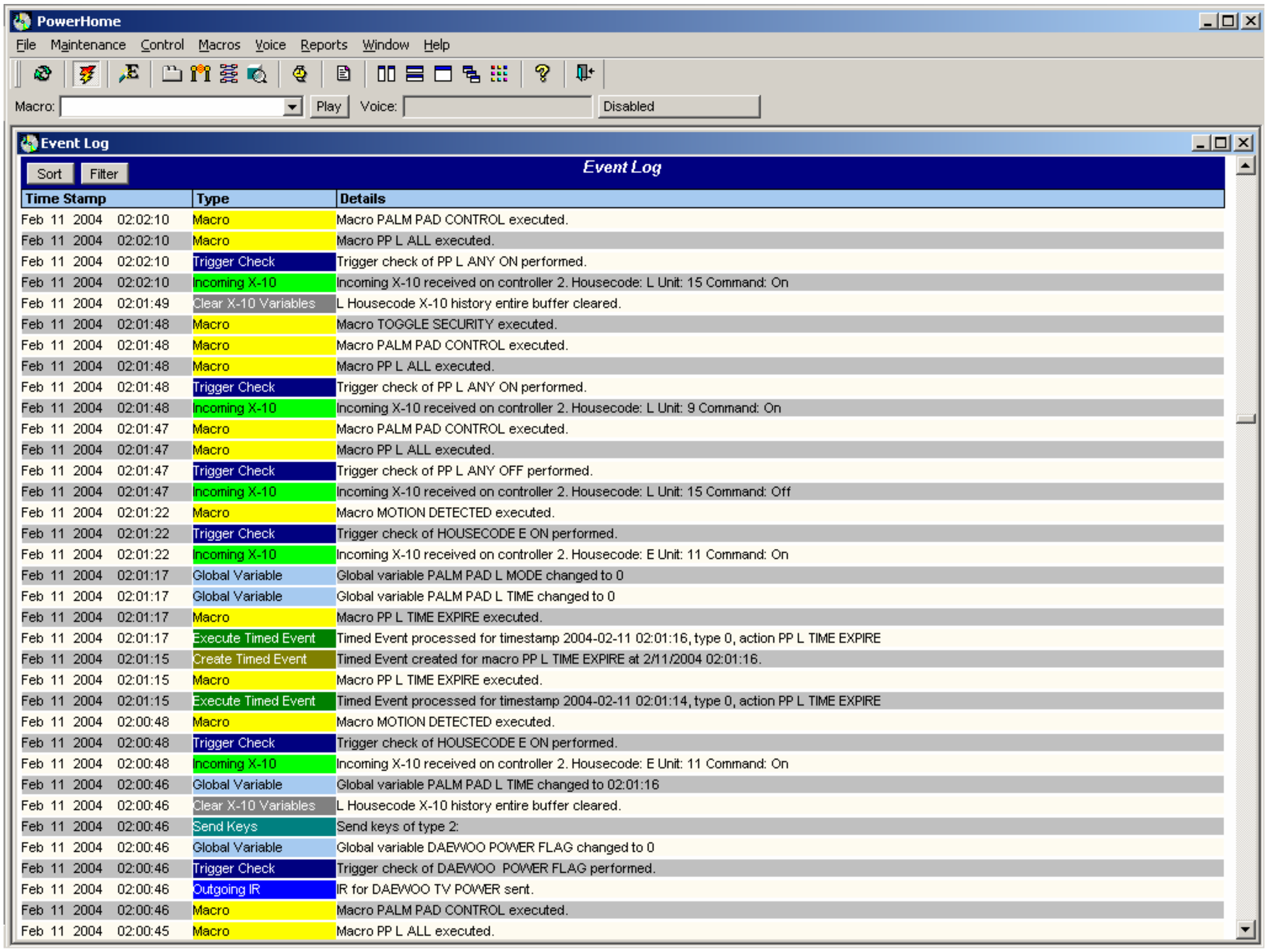

PowerHome
File Maintenance Control Macros Voice Reports Window Help
Macro: Play Voice: Disabled
X-10 Status

Device List

Status	Level	Location	Device Name	ID	Last Change	Control
Dark		DINING ROOM	PHOTOCELL	E8	Feb 11, 2004 14:29:36	
Off		DINING ROOM	LIGHT	K4	Feb 10, 2004 23:12:26	On Off ADim
No Motion		FRONT DOOR	MOTION	E1	Feb 11, 2004 14:15:05	
Light		FRONT DOOR	PHOTOCELL	E2	Feb 11, 2004 07:12:38	
Off		FRONT DOOR	LIGHT	H1	Feb 10, 2004 23:00:03	On Off ADim
No Motion		FRONT HALL	MOTION	E15	Feb 11, 2004 14:31:55	
Dark		FRONT HALL	PHOTOCELL	E16	Feb 11, 2004 14:30:49	
No Motion		GARAGE ROOM	MOTION	F3	Feb 11, 2004 14:31:22	
Dark		GARAGE ROOM	PHOTOCELL	F4	Feb 11, 2004 14:26:22	
On		GENERAL	RANDOM LIVE-IN	X4	Oct 25, 2003 13:06:01	On Off
On		GENERAL	SYSTEM CHECK EMAIL	X6	Oct 25, 2003 13:09:42	On Off
No Motion		GUEST BATHROOM	MOTION	F1	Feb 11, 2004 14:28:21	
Light		GUEST BATHROOM	PHOTOCELL	F2	Feb 11, 2004 07:58:46	
Off		GUEST BATHROOM	VANITY LIGHT	H5	Jan 31, 2004 23:39:02	On Off ADim
No Motion		HALLWAY	MOTION	E5	Feb 11, 2004 14:30:19	
Dark		HALLWAY	PHOTOCELL	E6	Feb 11, 2004 14:25:39	
Off		HALLWAY	HALL LIGHT	K6	Feb 11, 2004 14:30:20	On Off ADim
Motion Detected		KIDS ROOM	MOTION	E3	Nov 05, 2003 15:52:33	
Light		KIDS ROOM	PHOTOCELL	E4	Nov 05, 2003 15:30:25	
Off		KITCHEN	LIGHT	K5	Jan 31, 2004 23:39:01	On Off
No Motion		LIVING ROOM	MOTION	E9	Feb 11, 2004 14:30:30	
Dark		LIVING ROOM	PHOTOCELL	E10	Feb 11, 2004 14:29:25	
Off		LIVING ROOM	CHRISTMAS TREE	H2	Jan 31, 2004 23:39:02	On Off
Dimmed	14	LIVING ROOM	COUCH LAMP	K1	Feb 11, 2004 00:06:52	On Off ADim
Off		LIVING ROOM	LOVESEAT LAMP	K2	Feb 10, 2004 23:12:23	On Off ADim
Off		LIVING ROOM	SWAG LAMP	K3	Feb 10, 2004 23:12:24	On Off ADim
Barking		LIVING ROOM	ROBO-DOG	N1	Jan 28, 2004 19:54:38	On Off
No Motion		MASTER BATHROOM	MOTION	F5	Feb 11, 2004 14:31:36	
Light		MASTER BATHROOM	PHOTOCELL	F6	Aug 21, 2003 23:23:30	
Off		MASTER BATHROOM	VANITY LIGHT	H6	Feb 09, 2004 23:34:02	On Off ADim

Event Log

The Event Log tracks activity within the PowerHome program. You can access this screen from the "Reports" menu item. This screen IS NOT updated in real time so you must reselect to see any updates. Nearly every function that occurs within PowerHome can be seen and tracked in the Event Log. You can disable any log items you don't want to see in the Setup screen in the PowerHome Explorer under the Preferences section. Initially, you'll probably want to enable the logging of all events until you become comfortable with PowerHome. Then you'll probably want to disable some in order to reduce log clutter.

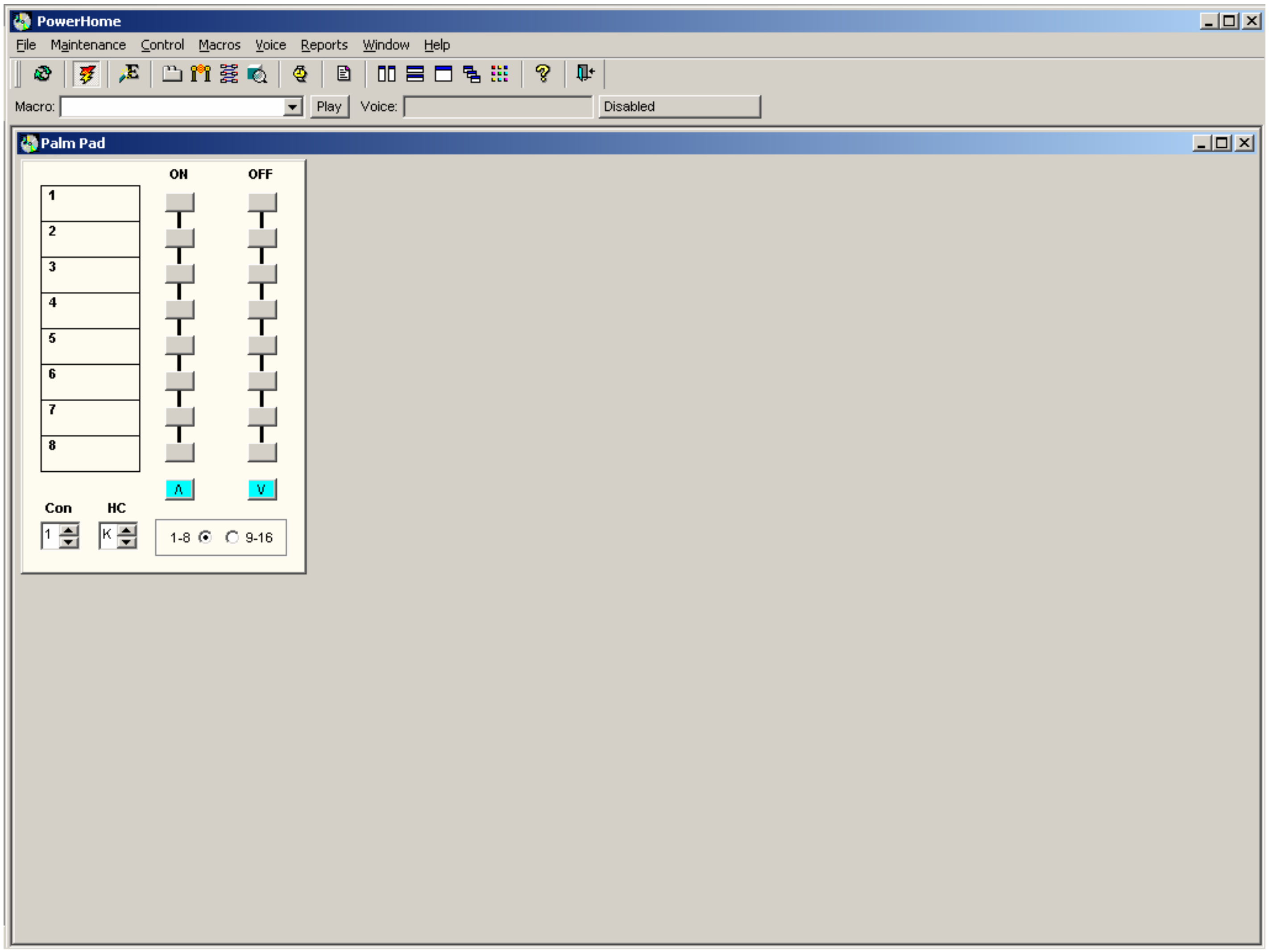


Time Stamp	Type	Details
Feb 11 2004 02:02:10	Macro	Macro PALM PAD CONTROL executed.
Feb 11 2004 02:02:10	Macro	Macro PP L ALL executed.
Feb 11 2004 02:02:10	Trigger Check	Trigger check of PP L ANY ON performed.
Feb 11 2004 02:02:10	Incoming X-10	Incoming X-10 received on controller 2. Housecode: L Unit: 15 Command: On
Feb 11 2004 02:01:49	Clear X-10 Variables	L Housecode X-10 history entire buffer cleared.
Feb 11 2004 02:01:48	Macro	Macro TOGGLE SECURITY executed.
Feb 11 2004 02:01:48	Macro	Macro PALM PAD CONTROL executed.
Feb 11 2004 02:01:48	Macro	Macro PP L ALL executed.
Feb 11 2004 02:01:48	Trigger Check	Trigger check of PP L ANY ON performed.
Feb 11 2004 02:01:48	Incoming X-10	Incoming X-10 received on controller 2. Housecode: L Unit: 9 Command: On
Feb 11 2004 02:01:47	Macro	Macro PALM PAD CONTROL executed.
Feb 11 2004 02:01:47	Macro	Macro PP L ALL executed.
Feb 11 2004 02:01:47	Trigger Check	Trigger check of PP L ANY OFF performed.
Feb 11 2004 02:01:47	Incoming X-10	Incoming X-10 received on controller 2. Housecode: L Unit: 15 Command: Off
Feb 11 2004 02:01:22	Macro	Macro MOTION DETECTED executed.
Feb 11 2004 02:01:22	Trigger Check	Trigger check of HOUSECODE E ON performed.
Feb 11 2004 02:01:22	Incoming X-10	Incoming X-10 received on controller 2. Housecode: E Unit: 11 Command: On
Feb 11 2004 02:01:17	Global Variable	Global variable PALM PAD L MODE changed to 0
Feb 11 2004 02:01:17	Global Variable	Global variable PALM PAD L TIME changed to 0
Feb 11 2004 02:01:17	Macro	Macro PP L TIME EXPIRE executed.
Feb 11 2004 02:01:17	Execute Timed Event	Timed Event processed for timestamp 2004-02-11 02:01:16, type 0, action PP L TIME EXPIRE
Feb 11 2004 02:01:15	Create Timed Event	Timed Event created for macro PP L TIME EXPIRE at 2/11/2004 02:01:16.
Feb 11 2004 02:01:15	Macro	Macro PP L TIME EXPIRE executed.
Feb 11 2004 02:01:15	Execute Timed Event	Timed Event processed for timestamp 2004-02-11 02:01:14, type 0, action PP L TIME EXPIRE
Feb 11 2004 02:00:48	Macro	Macro MOTION DETECTED executed.
Feb 11 2004 02:00:48	Trigger Check	Trigger check of HOUSECODE E ON performed.
Feb 11 2004 02:00:48	Incoming X-10	Incoming X-10 received on controller 2. Housecode: E Unit: 11 Command: On
Feb 11 2004 02:00:46	Global Variable	Global variable PALM PAD L TIME changed to 02:01:16
Feb 11 2004 02:00:46	Clear X-10 Variables	L Housecode X-10 history entire buffer cleared.
Feb 11 2004 02:00:46	Send Keys	Send keys of type 2:
Feb 11 2004 02:00:46	Global Variable	Global variable DAEWOO POWER FLAG changed to 0
Feb 11 2004 02:00:46	Trigger Check	Trigger check of DAEWOO POWER FLAG performed.
Feb 11 2004 02:00:46	Outgoing IR	IR for DAEWOO TV POWER sent.
Feb 11 2004 02:00:46	Macro	Macro PALM PAD CONTROL executed.
Feb 11 2004 02:00:45	Macro	Macro PP L ALL executed.

Since so much information is logged, the Event Log and the database can grow to enormous proportions if the log is not periodically trimmed. You have full control when the Event Log is trimmed and how many days worth of information you want to maintain. The Event Log is trimmed using the macro command "Trim Event Log" or the formula function ph_trimeventlog. This command allows you to empty the log except for the last X days. You can set X to any value you want, but would probably be satisfied with 2. You could then create a macro that runs on a daily basis and execute this command.

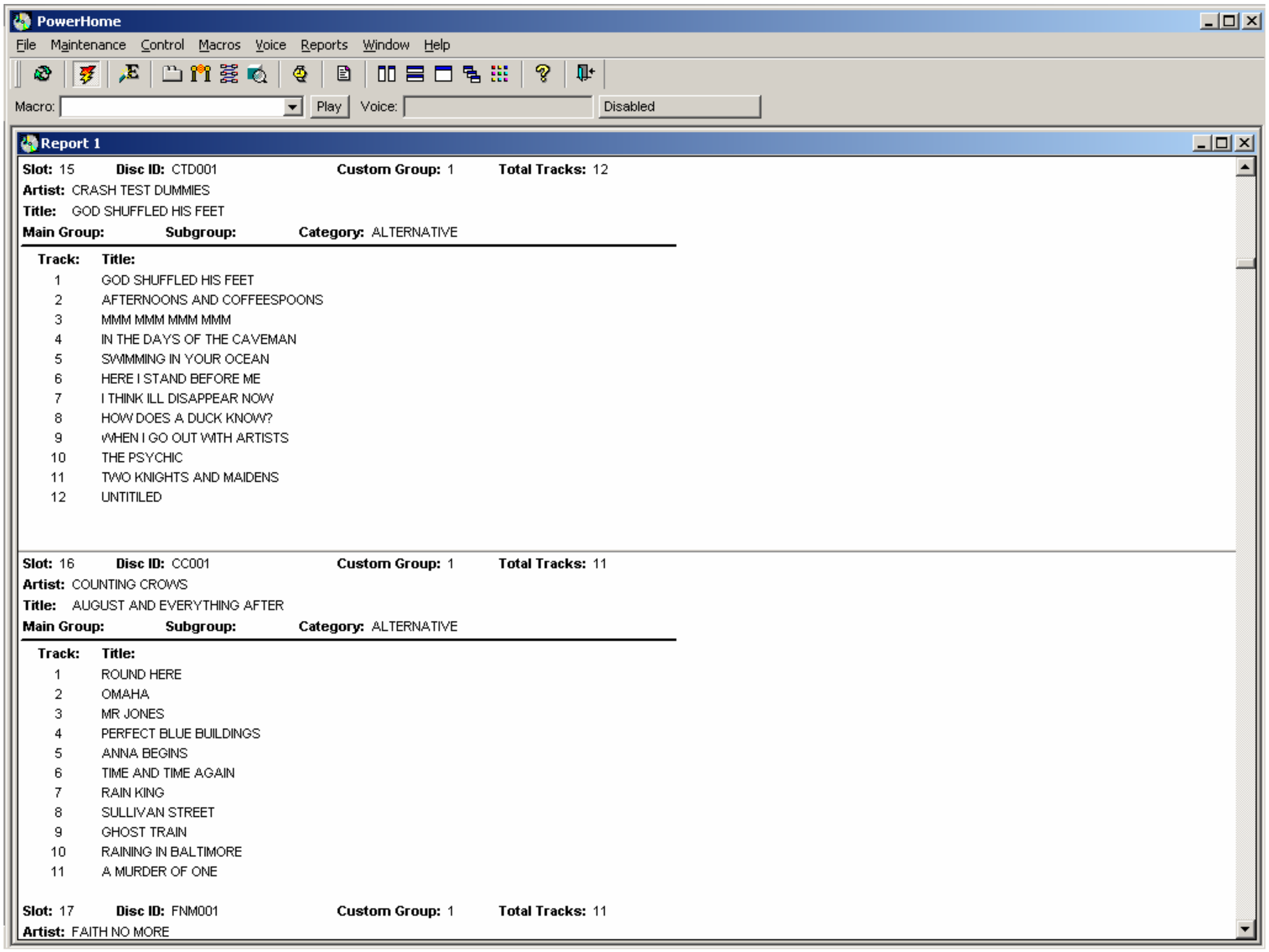
Palm Pad

The palm pad can be accessed by pressing the "Alt+F7" hotkeys. This screen appears similar to a standard X-10 palm pad and is useful for quickly testing your X-10 control with a minimal of setup. You can select the controller from which you want to send as well as the housecode.



Discs Report

This report is useful for querying your CD collection. Any discs that you have entered into the PowerHome database through the Explorer can be queried with this report. Once the report is on screen, it is the easiest way to create playlists for a CD jukebox. Just set the Active Playlist from the Control menu and then right click on the tracks that you want to add to a playlist. Choose Add to Playlist and your selection will be automatically added. You can also directly play a track from this menu by choosing Play. Choosing Play will use the PLAYSLOTTRACK macro to send the appropriate commands to your jukebox to play the selected track.



Slot: 15 **Disc ID:** CTD001 **Custom Group:** 1 **Total Tracks:** 12
Artist: CRASH TEST DUMMIES
Title: GOD SHUFFLED HIS FEET
Main Group: **Subgroup:** **Category:** ALTERNATIVE

Track:	Title:
1	GOD SHUFFLED HIS FEET
2	AFTERNOONS AND COFFEESPOONS
3	MMM MMM MMM MMM
4	IN THE DAYS OF THE CAVEMAN
5	SWIMMING IN YOUR OCEAN
6	HERE I STAND BEFORE ME
7	I THINK ILL DISAPPEAR NOW
8	HOW DOES A DUCK KNOW?
9	WHEN I GO OUT WITH ARTISTS
10	THE PSYCHIC
11	TWO KNIGHTS AND MAIDENS
12	UNTITILED

Slot: 16 **Disc ID:** CC001 **Custom Group:** 1 **Total Tracks:** 11
Artist: COUNTING CROWS
Title: AUGUST AND EVERYTHING AFTER
Main Group: **Subgroup:** **Category:** ALTERNATIVE

Track:	Title:
1	ROUND HERE
2	OMAHA
3	MR JONES
4	PERFECT BLUE BUILDINGS
5	ANNA BEGINS
6	TIME AND TIME AGAIN
7	RAIN KING
8	SULLIVAN STREET
9	GHOST TRAIN
10	RAINING IN BALTIMORE
11	A MURDER OF ONE

Slot: 17 **Disc ID:** FNM001 **Custom Group:** 1 **Total Tracks:** 11
Artist: FAITH NO MORE

PSP (PowerHome Server Pages)

The internal PowerHome web server is capable of providing content from three different sources. The first is the built-in web pages that allow you to monitor and control PowerHome. The second is from standard HTML files and graphics located within the PowerHome web server directory. The third is from PSP pages located within the PowerHome web server directory. With PSP, an advanced user is capable of creating and tailoring dynamic PowerHome content.

A PSP file is a standard HTML file with PowerHome formulas embedded within. These formulas can in turn call a WSH file allowing for a very flexible dynamic web content environment. A PSP page must end with the extension .PSP. When a PSP page is called from the internal web server, it first goes into the queue for evaluation. Standard HTML will just be passed through, but embedded PowerHome formulas will be evaluated. The results of the formula evaluations will then be passed through the web server along with the HTML.

A PSP is constructed similar to an ASP page in that the PowerHome formula must be enclosed within an opening tag of '<%' and a closing tag of '%>'. Whenever the web server encounters the PSP tags, the text between them will be sent to the PowerHome formula evaluator. It will first have variable substitution performed and then be evaluated. The results of the evaluation will replace the PSP tags and the formula between them. All of the PowerHome standard formula functions are available for use in PSP. There are several functions that are intended solely for use within PSP. You will be able to create forms and access the returned values of either a POST or GET command. You can also control the returned content-type and the return value. You can access the base64 encoding of a userid / password as well as the authorization level. Users will be able to completely customize their PowerHome web experience.

The best way to get started with PSP is to look at the included samples located in the PSP directory under the PowerHome web server directory. You will find a complete set of pages duplicating the internal web server pages. You can use these pages as a starting point and modify them to create the interface you like.

Socket Server

The Socket Server Interface provides users with a means of controlling PowerHome from external programs. Using this interface, a user can control PowerHome via a program running on the same machine or from a remote machine on a network. Since the Socket Server utilizes an open protocol, the controlling program does not have to be limited to the Windows platform. In order to use the Socket Server, you must first enable it within the Socket Server section of Setup in the PowerHome Explorer. You can use any valid port but the default is 8500. Be sure that this port is not in use by other applications on the same machine. You must also define an appropriate userid and password for the Socket Server. This userid and password is encoded using the standard Base64 encoding used by the HTTP specification. The protocol the Socket Server uses is loosely based upon the HTTP 1.0 specification. You may also restrict access to the socket server by declaring up to 5 restricted IP masks.

To use this interface, an external program must first establish a socket connection with the PowerHome socket server by using the appropriate IP address and port. After the connection is established, the application must send an appropriate request to PowerHome via the socket. A proper Socket Server request consists of a Header and a Body. The Header is separated from the Body by a double carriage return / line feed pair or double line feed. The Body consists of a valid PowerHome formula that will be evaluated. The Header has three sections which are each separated by a carriage return / line feed pair or just a line feed. The first section is the request type. The next two sections are the authorization and the content-length. Every Socket Server request must follow this format.

A valid request is detailed below:

FORMULA\nAuthorization: Basic eW91cm5hbWU6eW91cnBhc3N3b3Jk\nContent-Length: 45\n\nph_tts("This is the PowerHome Socket Server")

The body of the request is: ph_tts("This is the PowerHome Socket Server")
It is separated from the header by the double line feeds preceeding it. It could also be separated from the header by double carriage return / line feed pairs. This particular request is expecting the body to contain a valid PowerHome formula.

The Header of the request is: FORMULA\nAuthorization: Basic eW91cm5hbWU6eW91cnBhc3N3b3Jk\nContent-Length: 45
It has three separate sections separated by line feed characters. The sections could alternately be separated by carriage return / line feed pairs.

The first section of the Header is: FORMULA
This is the request type. Valid request types are FORMULA, FORMULATRIGGER, TRIGGER1, TRIGGER2, TRIGGER3,TRIGGER4, and TRIGGER5. The FORMULA request type expects the body to be a formula.   It does not fire a trigger. The FORMULATRIGGER request is exactly like the FORMULA request but it will also fire a trigger with the formula (data) being accessible via the [TEMP5] system variable. The TRIGGER1 through TRIGGER5 request types can contain anything within the body. They will all fire the appropriate trigger and the body (data) is accessible via the [TEMP5] system variable.

The second section of the Header is: Authorization: Basic eW91cm5hbWU6eW91cnBhc3N3b3Jk
The text after the word Basic is the userid and password separated by a colon. This string is then encoded using a Base64 encoding algorithm. The clear text in the above sample is yourname:yourpassword. Base64 encoding algorithms can be found on the web.

The third section of the Header is: Content-Length: 45
It contains the length of the Body section in bytes. In the above example this is 45 bytes for the formula within the Body.

After PowerHome has processed the request, it will return the results via the socket and then close the connection. A sample return from the above FORMULA request is detailed below:

PHSSP/1.0 200 OK\nServer: PowerHome Socket Server/1.02.1\nContent-type: text\nContent-Length: 1\n\n0

The return from the request also has a Header and a Body. The Header and Body are separated from each other in the same manner the request was with either double line feeds or double carriage return / line feed pairs. With a request type of FORMULA or FORMULATRIGGER, the body will contain the results of the formula evaluation. With a request type of TRIGGER1 through TRIGGER5, the body will be empty.

In the above example, the body is: 0
This is the result of evaluating the ph_tts function.

The first section of the Header are the results of the request: PHSSP/1.0 200 OK
This is the results of the request and follows the HTTP 1.0 specification. A 200 signifies success. Other possible results are 401 Unauthorized, 400 Bad Request, and 500 Server Error.

The second section of the Header is the server and version: PowerHome Socket Server/1.02.1

The third section of the Header: Content-type: text

The fourth section of the Header is: Content-Length: 1
This is the length of the body in bytes.

Windows Messaging

PowerHome includes a Windows Message Interface as a second means of controlling PowerHome from a remote program. Using Windows Messaging requires that the remote program be running on the same machine as PowerHome. PowerHome includes support for both the WM_COPYDATA message and the WM_USER messages. The WM_COPYDATA message (message number 74) uses a COPYDATASTRUCT and allows the user to evaluate a valid PowerHome formula, evaluate a formula and fire a trigger, or pass data and fire a trigger. The WM_USER message (message number 1024) through WM_USER + 4 (message number 1028) allows the user to pass data in both the lparam and wparam fields of the sendmessage function and fire a trigger.

To use the WM_COPYDATA message, the user must first populate an appropriate COPYDATASTRUCT. This structure is defined as:

Long	dwdata
Long	cbdata
Long	lpdata

The dwdata field should be populated with a value representing the type of WM_COPYDATA message to be executed. Valid values are 0, 1, and 11. A value of 0 will evaluate the formula pointed to by the lpdata parameter. A value of 1 will also evaluate the formula pointed to by the lpdata parameter as well as fire a trigger with the formula available in the system variable [TEMP5]. A value of 11 will just fire a trigger with the data pointed to by lpdata available in the system variable [TEMP5].

The cbdata field should be populated with the length of the data pointed to by the lpdata parameter.

The lpdata field should point to a valid PowerHome formula in the case of a type 0 or type 1 WM_COPYDATA message or to a string representing data in the case of a type 11 message.

After the COPYDATASTRUCT structure is populated, a SendMessage function can be executed. The handle contained in the SendMessage function should be the handle of the PowerHome program. This handle can be obtained by calling a FindWindow function and using the title of the PowerHome frame window. This title is " PowerHome ". It consists of a single space followed by the word 'PowerHome' followed by two spaces. The wparam field of the SendMessage function should contain the handle of the calling program. The lparam field of the SendMessage function should point to the COPYDATASTRUCT structure.

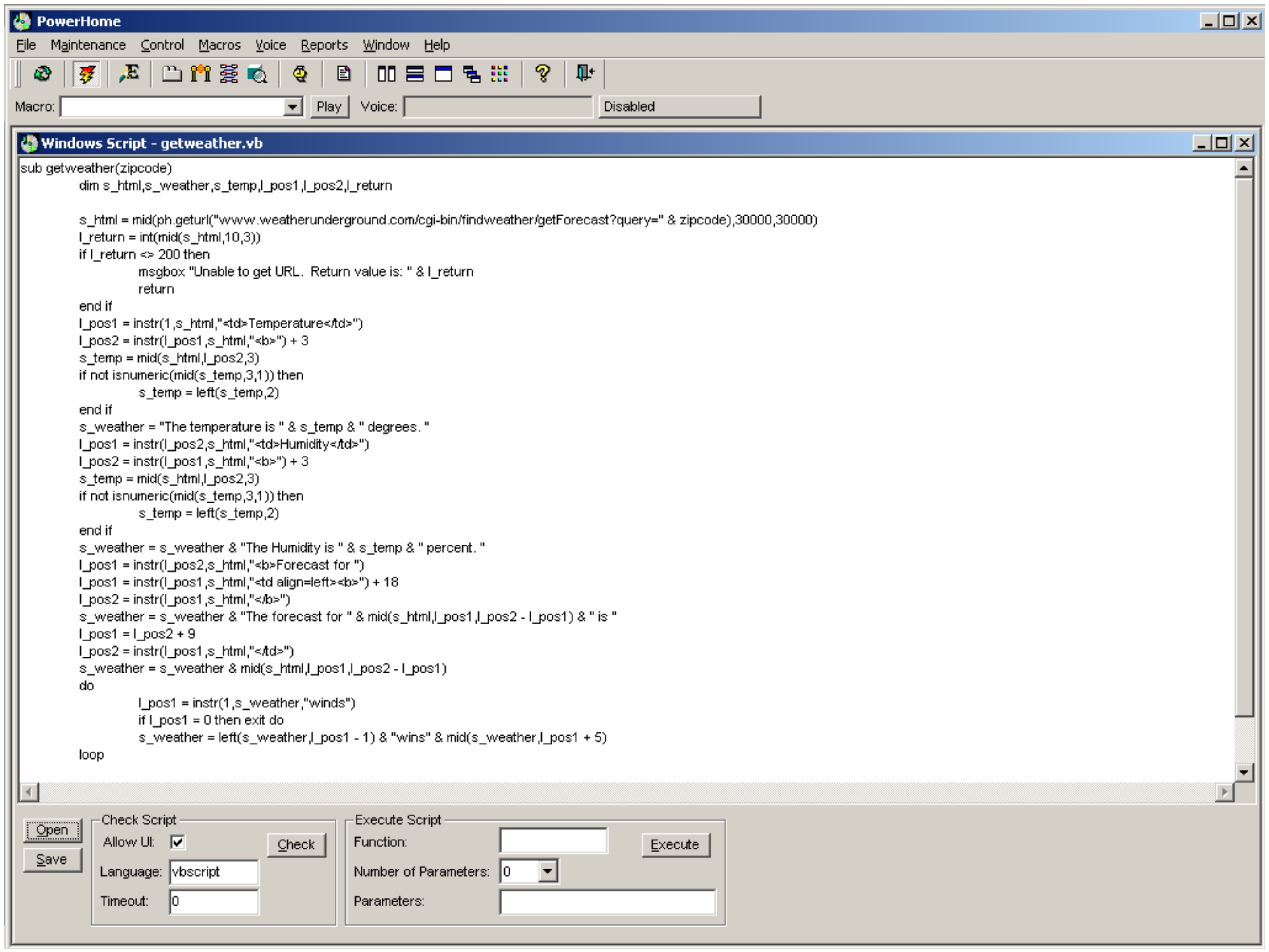
After successfully sending a WM_COPYDATA message, PowerHome will respond with its own WM_COPYDATA message if the type was a 0 or 1 and the wparam field was not 0. In the case of a 0 or 1 message, the result of the evaluated formula will be passed back. It is important that you supply the correct handle to your program in the wparam field if you want the result of the formula returned to you. If you don't care for the result of the formula, place a 0 in the wparam field. A type 11 message will never return anything and is used just to pass data and fire the appropriate trigger.

To use the WM_USER messages, simply execute a SendMessage function with whatever wparam and lparam values you wish to pass. PowerHome will fire the appropriate trigger (Trigger 1 for WM_USER, Trigger 2 for WM_USER + 1, etc). The wparam and lparam values will be available in the system variables [TEMP3] and [TEMP4] respectively. This is useful for quickly executing a predefined trigger within the PowerHome program.

WSH (Windows Script Host)

PowerHome includes the ability to execute Windows Scripts in any language for which you've installed the appropriate engine using the Windows Script Host components. WSH is not part of the PowerHome installation and will have to be downloaded directly from Microsoft and installed separately if your system does not already have them and you desire this capability. WSH is included with Microsoft Windows 2000 but will have to be installed on Windows 95, 98, and NT. WSH will default install the engines for both VBScript and JScript. Additional engines are available on the web. In addition to having the full capabilities of the Script language at your disposal, PowerHome also extends this functionality with an OLE component containing functions that directly control and retrieve information from the PowerHome program. With this facility, you will no longer be confined to the functionality contained within the internal PowerHome macro and formula scripting.

WSH scripts are stored in normal text files on the PowerHome machine and are not a part of the PowerHome database. You can execute a script by calling one of the six formula functions ph_runscript_0 through ph_runscript_5. A script must consist of functions and/or subroutines. You can have multiple functions and/or subroutines in a single script file. Each of these functions or subroutines can call other functions or subroutines in the same script file. When you call one of the six runscript formula functions, you can begin execution at any function or procedure contained within the script file and pass parameters to the formula or procedure in the case of the ph_runscript_1 through ph_runscript_5 functions. See the online help for a detailed description on the runscript functions. A list of the OLE functions available to WSH scripts for directly interfacing with PowerHome are also in the online help. They are almost exact duplicates of the ph_ formula functions.


PowerHome
File Maintenance Control Macros Voice Reports Window Help
Macro: Play Voice: Disabled
Windows Script - getweather.vb

```
sub getweather(zipcode)
        dim s_html,s_weather,s_temp,l_pos1,l_pos2,l_return

        s_html = mid(ph.geturl("www.weatherunderground.com/cgi-bin/findweather/getForecast?query=" & zipcode),30000,30000)
        l_return = int(mid(s_html,10,3))
        if l_return <> 200 then
                msgbox "Unable to get URL.  Return value is: " & l_return
                return
        end if
        l_pos1 = instr(1,s_html,"<td>Temperature</td>")
        l_pos2 = instr(l_pos1,s_html,"<b>") + 3
        s_temp = mid(s_html,l_pos2,3)
        if not isnumeric(mid(s_temp,3,1)) then
                s_temp = left(s_temp,2)
        end if
        s_weather = "The temperature is " & s_temp & " degrees. "
        l_pos1 = instr(l_pos2,s_html,"<td>Humidity</td>")
        l_pos2 = instr(l_pos1,s_html,"<b>") + 3
        s_temp = mid(s_html,l_pos2,3)
        if not isnumeric(mid(s_temp,3,1)) then
                s_temp = left(s_temp,2)
        end if
        s_weather = s_weather & "The Humidity is " & s_temp & " percent. "
        l_pos1 = instr(l_pos2,s_html,"<b>Forecast for ")
        l_pos1 = instr(l_pos1,s_html,"<td align=left><b>") + 18
        l_pos2 = instr(l_pos1,s_html,"</b>")
        s_weather = s_weather & "The forecast for " & mid(s_html,l_pos1,l_pos2 - l_pos1) & " is "
        l_pos1 = l_pos2 + 9
        l_pos2 = instr(l_pos1,s_html,"</td>")
        s_weather = s_weather & mid(s_html,l_pos1,l_pos2 - l_pos1)
        do
                l_pos1 = instr(1,s_weather,"winds")
                if l_pos1 = 0 then exit do
                s_weather = left(s_weather,l_pos1 - 1) & "wins" & mid(s_weather,l_pos1 + 5)
        loop
```

Allow UI: ✓
Language: vbscript
Timeout: 0
Function:
Number of Parameters: 0
Parameters:

PowerHome also includes a limited script editor. You can access this editor from the Maintenance menu. You can open, edit, and save scripts. You can also perform a syntax check and execute scripts from within this screen to check for runtime errors. More powerful editors are available and you can use any editing environment you are comfortable with.

For more information on WSH and scripting in general, please visit the Microsoft website.

Email Control

PowerHome also has the ability to be remotely controlled via Email messages. You must enable this setting in the Setup -> Email section of the PowerHome Explorer if you wish to use this feature. Once enabled, you can send email to PowerHome with embedded Send Keys, Formulas, or Macros. To use this feature, you must have a default MAPI client properly installed and setup. Whatever Email accounts you have enabled in the default MAPI client may have Control Email sent to them. Use the Macro command Process Email or the Formula functions ph_processemail and ph_processemailthread to retrieve and process any Control Email. You can do this periodically with a Timed Event.

The Control Email is recognized from non-control email via the subject. Valid subjects for Control Email must be one of the following: Sendkeys, Formula, or Macro. Case is not important for the subject. If the subject does not match one of the previous cases, then the email is not processed as a control email. The subjects are self-explanatory and the body of the email will be processed appropriately. In the case of a Sendkeys control message, the body is considered a formula and is evaluated. The result of the formula will then be processed as a sendkeys. A Formula control message will have the body of the message evaluated and the result discarded. A macro control message expects the ID of a single macro to be executed.

When processing Control Email, normally the entire body is used. With some mailing programs, this is not desirable because of additional characters that may be generated and tacked onto the end of the message. These extra “garbage” characters would be interpreted as part of the Control message and would cause unknown results. To prevent this from happening, you can insert an open and close curly brace ({}) in the body of your message to signal the end of the Control message body. Any characters after the first set of {} characters will be ignored.

Virtual X-10

PowerHome provides support for “virtual” X-10 devices. Virtual X-10 devices behave exactly like actual X-10 devices as defined by the X-10 type they have been assigned. Housecodes Q through Z with unitcodes 1 through 99 represent virtual X-10 devices within PowerHome. Of course you can’t receive commands on virtual X-10 devices but you can send X-10 commands to them and have triggers fire. They also appear on the X-10 status screen so you can see the values change in real time. For certain data, you can use these virtual X-10 devices instead of Global Variables and get the benefits of triggers and real time data display. You send X-10 commands to a virtual X-10 device just as you would an actual X-10 device. The device will respond depending upon the X-10 device type you have set. All outgoing X-10 triggers available for actual X-10 are available for virtual X-10. You can also directly update virtual X-10 with the ph_x10setstat function. This will result in an update to the X-10 status screen but will not fire any triggers.

Appendix A

Controllers

CM11A:

The CM11A is a bi-directional X-10 only controller made by X-10. It is capable of receiving and sending extended commands. It is capable of very simple stand-alone operation or direct control via its serial port. PowerHome only supports the latter mode of operation. It is only while coupled to a PC that the CM11A is capable of sophisticated if-then-else processing. While it is possible to download macros into the CM11A using the ActiveHome software and use the programmed macros in conjunction with PowerHome, this generally wouldn't be desirable. Note that PowerHome does not currently clear the CM11A memory and that you must use ActiveHome to do this. If you are experiencing unexplained X-10 actions, it may be caused by downloaded macros within the CM11A and you may have to clear its memory.

The CM11A has a single parameter within the "pwrhome.ini" file and is the "Retry". This parameter controls how many times PowerHome will retry sending an X-10 command before it will timeout.

CM17A:

The CM17A is a small transmit only RF X-10 controller manufactured by X-10. It is commonly referred to as the "firecracker". It is only capable of simple commands consisting of on, off, dim, bright, all units off, and all lights on. In order to use this device you must have something capable of receiving RF X-10 commands such as the TW-523 or RR-501 transceivers.

The CM17A is not capable of just addressing a module. It must send an address and command together. PowerHome is designed such that you have full control over X-10 addressing and commands. With a device such as the CM11A, you can send multiple addresses followed by a single command to control a group of devices together. The CM17A will not work in this fashion. When a command is sent to the CM17A, it will only send the full address and command for the last address it received.

The CM17A has five parameters within the "pwrhome.ini" file. Normally the defaults should be fine but they can be fine-tuned for a particularly troublesome unit. The firecracker derives its power from the DTR and RTS lines of the serial port. By toggling one or the other line off, the appropriate signal is generated. One or both of the lines must be active at any time in order for a proper signal to be generated. The first four parameters in the "pwrhome.ini" file represent the number of microseconds. The first parameter is the Standby and is the amount of time for initializing and connecting to the CM17A. The default is 100000 which is equivalent to 100 milliseconds. The second parameter is Signal. This parameter represents the amount of time a '1' or '0' bit is held when communicating with the CM17A. The default is 600 which equates to 600 microseconds. The third parameter is Wait. This is the amount of time between bits. The default is 600. The fourth parameter is Send Delay. This is the amount of time to wait for the CM17A to send its RF X-10 signal once it has received all the bits. The default is 800000 which is equal to 800 milliseconds. The fifth parameter is the DimPercent. When sending dim signals from the CM17A, the amount of actual dim which ultimately appears on the powerline is dependent upon the receiver that you use. This setting only affects the level that is reported to the X-10 Status Screen and has nothing to do with the actual amount of dim that the CM17A sends.

MR26A:

The MR26A is a receive only RF X-10 controller manufactured by X-10. It is sometimes referred to as the "mouse remote". The one nifty feature of this device is that it is capable of receiving all 16 housecodes simultaneously. It receives the following commands: on, off, dim, bright, all units off, all lights on, and all lights off. Coupled with PowerHome and a powerline X10 transmitter, this device can eliminate the multitude of X-10 transceivers you may have plugged into your wall. It is also extremely fast and solves one of the biggest problems with PC control of lights triggered by motion sensors. Without the MR26A, a hawkeye motion sensor sends an RF signal when motion is detected. This RF signal is received by an X-10 transceiver. This X-10 transceiver will output the signal onto the powerline. An X-10 receiver hooked to your automation PC then receives the X-10 command and passes it to the PC for processing. Most CM11A's will have a short delay before sending the command to the PC while it waits to see if another command is coming. Once the PC receives the command and sends the appropriate X-10 command to turn on a light, 2 or 3 seconds may have passed. In real time, this is quite slow. The MR26A bypasses most of the steps and when the hawkeye sends an RF signal in response to motion, the MR26A receives it immediately and transmits it directly to the automation PC via serial port. The PC responds with the X10 code to turn on the light almost immediately.

One drawback to this device is the antenna. The rat-tail antenna supplied has limited range. A number of external antenna designs vastly improving the range have been posted on the internet.

The operation of the MR26A is handled differently from the other X-10 controllers supported by PowerHome. The first difference is that only X-10 RF In triggers are triggered for this device. The X-10 In trigger is not. When an RF command is received, it is not automatically echoed onto the powerline. You must either do this manually from a trigger or by changing the echo and echocontroller settings in the pwrhome.ini file. Also, no X-10 status variables are modified by the MR26A unless the updatestatus setting in the pwrhome.ini file is set to yes. The internal X-10 matrix is also not modified. The incoming previous X-10 multi-x buffers ARE updated however. If you want the command echoed to the powerline, this can be easily accomplished via a macro or Send Keys called by a trigger or the appropriate ini file setting. This action will cause a change in the X-10 status screen. If the command is not necessary to be transmitted on the powerline but you still want an update in the X-10 status screen, use the ph_x10setstat formula from within a trigger to change the status or set the updatestatus parameter in the pwrhome.ini file to automatically update devices whose type is set as status only.

The MR26A has five parameters within the “pwrhome.ini” file. The first parameter is the timeout and is in milliseconds. When a typical X-10 RF transmitter such as a palm pad sends a signal, the signal is repeated a number of times. The MR26A responds to this by sending multiple signals to the PC. Normally you want this to be interpreted as a single signal. The timeout parameter is the amount of time after the receipt of the first repeated signal before the MR26A considers it to be a different command. The default value is 500 which is equivalent to 500 milliseconds. If you get more than one signal registered in your event log from a single button press on a palm pad or activation of a motion sensor, you may have to increase this value. Beginning with PowerHome version 1.01.3, the MR26A controller was upgraded to allow the automatic update of status only devices as well as the capability of automatically echoing received commands using another X-10 controller. The second parameter is updatestatus and should be set to yes if you want X-10 devices that are flagged as status only to automatically have their status updated with received commands. Set to no if you wish to handle this yourself. The third parameter is the echo. You should set this parameter to the housecodes that you want received commands to be echoed for. If you don’t want your commands echoed, then set this parameter blank. The fourth parameter is the X-10 controller number that you wish to use for echoing commands. Set this to a number from 1 to 5. Any housecodes that appear in the echo command will be sent using this controller number. If a housecode is to be echoed, status only devices will NOT be updated by the updatestatus and will instead be updated as a result of the command being echoed. The fifth parameter is useful when combining the MR26A with the CM17A. Set the IgnoreCM17A parameter to ‘yes’ so the MR26A will not respond to commands sent by PowerHome to the CM17A.

CIR:

The CIR or "Computerized Infrared Remote" is a parallel port infrared sending device. It can receive IR commands for learning only. This device is extremely simple to build with a parts cost of $15.00 or less. The plans for this interface can be found at http://www.ziplabel.com/cir . This device can only be used with Windows 95/98. It will not work on NT or 2000. This interface must disable interrupts during operation because of the timing issues associated with the sending and receiving of infrared commands. During this period, your PC will appear frozen. This will hardly be noticeable during the playback of infrared commands, but will be very noticeable during infrared command learning. Once all of your infrared commands have been learned, this should not be a problem.

The CIR device uses a custom DLL designed specifically for PowerHome and is radically different from the freeware DLL's currently available on the web. If you have used this device before, you are aware that it is most troublesome during the tedious learning process. The custom DLL should alleviate most of the issues with teaching the CIR infrared commands.

When setting this device up in the Setup screen of the PowerHome Explorer, you must tell the program what parallel port your CIR device is connected to. This will commonly be LPT1 which is generally 378 hex but can sometimes by 3BC hex. LPT2 is commonly 278 hex.

When defining IR equipment which will use the CIR device, leave the IR zone at 1. The Learn Freq can be left at any value as it is not used with the CIR interface.

The most difficult aspect of using the CIR interface is the learning of infrared commands. The positioning of the remote to be learned in relation to the CIR interface must be nearly exact. While you are learning a command, your computer will appear to be frozen. Your mouse will not move and no keys will respond. Position the equipment's infrared remote approximately 1 inch away from the CIR device. Try to maintain a steady distance. Place the infrared remote on a flat surface if necessary. With your distance set, press and hold the corresponding remote key for 2 full seconds. If you do not receive a message, let up on the remote button briefly and press again for 2 full seconds. Repeat this process up to five times or until you get a message. If after 5 times, you have not received a message, try moving the remote either closer or further away by a quarter inch and repeat the process. If you do get a message saying you are either too far away or too close, adjust your distance accordingly and try to learn again. You will have to right-click and select Record again if you have gotten a message. Once you have gotten a message, your computer will no longer be locked. If after repeated tries you can get no message and your computer is locked, re-verify the proper connection of the CIR device. If you still cannot receive a proper signal, you will have to power down your computer or press the reset key to regain control. Once you have successfully learned the infrared signal, make note of the distance required. This distance should be good for learning every other key associated with that remote. A different remote may require a different distance, but this initial distance should be a good starting point. If you did receive a clean signal, you should see that the code field has been updated with a lengthy hexadecimal code. Save your changes by double-clicking or pressing the 'Refresh' button, then right-click the key number you just learned and select play. Be sure that the CIR device is facing the piece of equipment you just learned. If all went well, the CIR device should be able to control the button you just learned. If not, you may have to relearn the key again. Keep repeating this process until you are able to successfully record and playback the first key, taking careful note of the distance between the CIR and the remote required to achieve good results. Once you have successfully completed this step, right-click in the lower window area and select Auto Learn from the popup menu. Follow the instructions and you will be prompted to learn each key in order. Watch the messages and you should be able to quickly learn the infrared commands for your equipment.

Multi-CIR:

The multi-CIR interface is merely a standard CIR device with multiple stages of IR senders. If you look at the schematic of the CIR, you can see that it really is two separate circuits. One for learning signals and one for transmitting signals. If you duplicate the IR transmit circuit 1 or more times and use the driving signal from the parallel port starting at pin 4 thru pin 8, you can have PowerHome use this interface for multi-zone operation. If this is the case, set the IR Zone field to the zone you want to control with zone 1 controlled by pin 2, zone 2 controlled by pin 4, zone 3 controlled by pin 5, zone 4 controlled by pin 6, etc.

RedRat2:

The RedRat2 is a serial controlled IR sending device. It can receive infrared commands for learning only. This device will work with Windows 95/98,NT, and 2000. If you have any experience working with the CIR, you will be amazed at how easy this device is to work with. This excellent device may be ordered online at http://www.dodgies.demon.co.uk . Commands are easily and quickly learned with distance not being a factor as commands can be successfully learned from 2 inches to 3 feet or more.

When setting up IR Equipment to use this interface, leave the IR Zone at 1. The Learn Freq can be set at any value as the RedRat2 does not use it and instead samples the frequency dynamically while learning a signal.

CPU-XA/Ocelot:

This device supports X-10 transmit and receive, Infrared transmit and receive, as well as digital inputs / outputs and analog inputs with add-on modules. It is capable of stand-alone operation with its included C-MAX software. PowerHome uses this device as a controller and does not support the downloading of information for stand-alone operation. Unlike the CM11A, this controller is capable of sophisticated if-then-else logic and has enough memory to store robust programs. If you intend to use this controller with PowerHome, you must leave it connected to your automation PC with the PowerHome program running. PowerHome supports the X-10 and IR capabilities of the CPU-XA/Ocelot as well as the digital inputs/outputs and analog inputs available with add-on modules.

The CPU-XA/Ocelot uses the TW-523 interface for the sending and receiving of X-10 commands. This means that the CPU-XA/Ocelot cannot receive extended X-10 commands. Limited support is available for the sending of some extended commands, but PowerHome does not support this at this time. You still have available the rest of the X-10 command set including Status Request and Preset Dim. Since the X-10 interface is the TW-523, you will not receive accurate incoming dim signals. The TW-523 only reports that a dim signal was received, unlike the CM11A which reports how much of a dim signal was received.

The CPU-XA/Ocelot has two parameters in the "pwrhome.ini" file. The first is the DimPercent to account for the above mentioned problem associated with the receiving of dim commands. This is default set to 6.25 percent but can be adjusted for what is typical for your situation. The second parameter is the Retry parameter. This is the number of times the CPU-XA/Ocelot shall retry sending an X-10 command before it times out.

When using this interface for IR commands, it is setup a little differently from other IR controllers that PowerHome supports. In most IR controllers, the actual raw IR data is stored within the PowerHome database in the Code field in the IR Equipment Details screen. The CPU-XA/Ocelot instead stores all learned IR commands within its own memory. Early models could only store 512 IR commands and later models stored 1024 commands. These commands are referenced via number from 0 to 511 and 0 to 1023 respectively. **Before learning IR commands for this interface, you must place the number of the command that you want to learn in the Code field manually. It is recommended that you don't use 0 and instead start your commands with 1. Also, if you are going to be using the IR receive feature of the CPU-XA/Ocelot, it is recommended that any commands that you want to recognize on receive be given the lower IR numbers. This is because the CPU-XA/Ocelot will start comparing commands that it receives starting with 1 and working its way up. Also, the CPU-XA/Ocelot will only scan the first X number of commands with X defaulting to 80 from the factory. See the CPU-XA/Ocelot manual for more information.**

When setting up IR Equipment to use this interface, leave the IR Zone at 1 if you are using the onboard IR transmitter. If you have an add-on SECU-16IR (16 zone IR output) module, you can set the IR Zone so that you can send the signal to another zone or all zones. The number you place in the IR Zone is dependant upon the following formula: mnum X 256 + znum. Mnum is the module number of the SECU-16IR that you want to send IR from and will be from 1 to 128. Znum is the zone number on module you are addressing that you want to send from and will be from 0 to 15. You can use 255 for znum to send the IR to all zones on the addressed module. If you have a SECU-16IR setup as module number 12 and want to send an IR command to that modules zone 8, use the following formula: 12 X 256 + 8 = 3080. In this case you would place the number 3080 in the IR Zone field. The Learn Freq field should be set to the IR frequency in Khz. This number can typically range from 36 to 45. A good starting point would be 38. If you are unable to get your IR code to be successfully learned, try changing this value in small increments.

Slink-e:

The Nirvis Slink-e is a serial controller supporting both IR and S-link control of your equipment. This device has not been tested with PowerHome and only limited support has been enabled at this time. If you have this device, feedback on your results with PowerHome would be appreciated.

PowerHome uses the Slink-x Active-X control supplied by Nirvis in their CDJ jukebox software package. You must download and install this software in order for PowerHome to control the Slink-e. Once installed, the first thing you will need to do is use the E-Z Learn software supplied in the CDJ package to learn the IR commands for your individual pieces of IR equipment. This will create files for each of your IR devices with the raw IR codes in the format that the Slink-e requires.

Once you've created the E-Z Learn files, you can setup your IR equipment in the PowerHome Explorer. When defining an ID for your equipment, you must not use spaces as the Slink-e does not support them. Use an underscore character to achieve the desired effect. You must also define 1 extra key than what you have actually learned in the total keys field. Once this is complete, double-click the record to edit the IR Equipment Details. In the last row, with the greatest key #, set the Description to "SLINKE" without the quotes. In the Code field for this record, type the full path and filename of the E-Z Learn file associated with the IR device you're working with. For the rest of the rows, proceed as you normally would using the Description you would like to appear in the Control Center. For the Code, type in the Command Name that you used for the key when you learned the command using E-Z Learn. In the future, this step will be performed automatically by importing the information directly from the E-Z Learn file when you use the Auto-Learn feature.

Appendix B

Database Structure

all_id_data:

id	char(25)	NOT NULL
description	char(25)	NOT NULL
type	smallint	NOT NULL
sort	smallint	NOT NULL

cddbservers:

entrynum	smallint	NOT NULL
site	char(128)	NOT NULL
port	smallint	NOT NULL
address	char(128)	
latitude	char(7)	
longitude	char(7)	
description	char(128)	

PRIMARY KEY (entrynum)

cities:

abbr	char(2)	NOT NULL
airport	char(3)	NOT NULL
city	char(20)	NOT NULL
latitude	numeric(5,2)	NOT NULL
longitude	numeric(5,2)	NOT NULL
timezone	numeric(2,0)	NOT NULL

PRIMARY KEY (abbr, airport)

control:

id	char(25)	NOT NULL
id_value	char(256)	NOT NULL

PRIMARY KEY (id)

customcolor:

id	smallint	NOT NULL
color	char(15)	
value	integer	

PRIMARY KEY (id)

customtab:

id	char(25)	NOT NULL
tab_order	smallint	
bkgrnd_color	integer	
bkgrnd_bmp	char(256)	
bmp_resize	char(1)	
h_scrollbar	char(1)	
v_scrollbar	char(1)	

PRIMARY KEY (id)

discs:

id	char(25)	NOT NULL
artist	char(255)	NOT NULL
title	char(255)	NOT NULL
tracks	smallint	NOT NULL
category	char(255)	NOT NULL
maingroup	smallint	
subgroup	smallint	
cddb	char(8)	

PRIMARY KEY (id)

dropdowns

ddtype	smallint	NOT NULL
dddisplay	char(25)	NOT NULL
ddcharval	char(25)	
ddnumval	smallint	
ddsort	smallint	NOT NULL

PRIMARY KEY (ddtype, dddisplay, ddsort)

equip:

id	char(25)	NOT NULL
manuf	char(25)	NOT NULL
model	char(25)	NOT NULL
total_keys	smallint	NOT NULL
learn_freq	smallint	NOT NULL
controller	smallint	NOT NULL
ir_zone	smallint	NOT NULL
customtab	char(25)	

PRIMARY KEY (id)

eventlog:

time_stamp	timestamp	NOT NULL
type	smallint	NOT NULL
event	char(1024)	NOT NULL
sequence	smallint	NOT NULL

PRIMARY KEY (time_stamp, sequence)

global_variables:

id	char(25)	NOT NULL
value	char(1024)	

PRIMARY KEY (id)

ir:

id	char(25)	NOT NULL
key_number	smallint	NOT NULL
key_desc	char(25)	NOT NULL
key_x	smallint	NOT NULL
key_y	smallint	NOT NULL
key_width	smallint	NOT NULL
key_height	smallint	NOT NULL
key_code	smallint	NOT NULL
key_flag	smallint	NOT NULL

key_text	integer	NOT NULL
key_bkgrnd	integer	NOT NULL
code	char(2048)	
key_border	smallint	NOT NULL

PRIMARY KEY (id, key_number)

macrodetail:

id	char(25)	NOT NULL
sequence	smallint	NOT NULL
type	smallint	NOT NULL
type_id	char(25)	
type_number	smallint	
send_keys	char(1024)	
skip_line	smallint	NOT NULL

PRIMARY KEY (id, sequence)

macroheader:

id	char(25)	NOT NULL
key_desc	char(25)	NOT NULL
key_x	smallint	NOT NULL
key_y	smallint	NOT NULL
key_width	smallint	NOT NULL
key_height	smallint	NOT NULL
key_code	smallint	NOT NULL
key_flag	smallint	NOT NULL
key_text	integer	NOT NULL
key_bkgrnd	integer	NOT NULL
menu_shortcut	smallint	NOT NULL
grammar_include	smallint	NOT NULL
list_include	smallint	NOT NULL
customtab	char(25)	
key_border	smallint	NOT NULL

PRIMARY KEY (id)

macrowait:

id	char(25)	NOT NULL
mod_ts	timestamp	NOT NULL
mod_seq	smallint	NOT NULL
wait_ts	timestamp	NOT NULL
tempvardata	long binary	

PRIMARY KEY (id, mod_ts, mod_seq)

playlistdetail:

id	char(25)	NOT NULL
sequence	smallint	NOT NULL
disc_id	char(25)	NOT NULL
track	smallint	NOT NULL

PRIMARY KEY (id, sequence)

playlistheader:

id	char(25)	NOT NULL
description	char(255)	NOT NULL

PRIMARY KEY (id)

sendkeys:

id	char(25)	NOT NULL
key_desc	char(25)	NOT NULL
key_x	smallint	NOT NULL
key_y	smallint	NOT NULL
key_width	smallint	NOT NULL
key_height	smallint	NOT NULL
key_code	smallint	NOT NULL
key_flag	smallint	NOT NULL
key_text	integer	NOT NULL
key_bkgrnd	integer	NOT NULL
customtab	char(25)	
send_keys	char(1024)	NOT NULL
key_border	smallint	NOT NULL

PRIMARY KEY (id)

slots:

slot	smallint	NOT NULL
id	char(25)	
custom	smallint	

PRIMARY KEY (slot)

states:

abbr	char(2)	NOT NULL
state	char(20)	NOT NULL

PRIMARY KEY (abbr)

timedevents:

starttime	timestamp	NOT NULL
frequency	integer	NOT NULL
type	smallint	NOT NULL
action	char(1024)	NOT NULL
timing	smallint	NOT NULL
reftime	timestamp	NOT NULL
offsettype	smallint	NOT NULL
offsetamount	smallint	NOT NULL

PRIMARY KEY (starttime, frequency, type, action, timing, reftime, offsettype, offsetamount)

tracks:

id	char(25)	NOT NULL
track	smallint	NOT NULL
title	char(255)	NOT NULL

PRIMARY KEY (id, track)

triggers:

id	char(25)	NOT NULL
description	char(25)	NOT NULL
action	char(1024)	NOT NULL
trigger_type	smallint	NOT NULL

trigger_id	char(25)	NOT NULL
trigger_number	smallint	NOT NULL
trigger_value	smallint	NOT NULL
boolean	char(1024)	NOT NULL
status	smallint	NOT NULL
action_style	smallint	NOT NULL
action_type	smallint	NOT NULL

PRIMARY KEY (id)

userdata1:

type	integer
idstring	char(25)
idnum	integer
iddate	timestamp
valstring	char(1024)
valnum	double
valdate	timestamp

video:

id	char(25)	NOT NULL
title	char(255)	NOT NULL
genre	char(255)	NOT NULL
rating	char(10)	NOT NULL
year	smallint	NOT NULL
minutes	smallint	NOT NULL
media	char(10)	NOT NULL
aspect	char(25)	NOT NULL
audio	char(25)	NOT NULL
summary	char(1024)	

PRIMARY KEY (id)

voicedetail:

id	char(25)	NOT NULL
command	char(256)	NOT NULL
syntax	char(256)	NOT NULL
action	char(1024)	NOT NULL
actiontype	smallint	NOT NULL

PRIMARY KEY (id, command)

voiceheader:

id	char(25)	NOT NULL
persistant	smallint	NOT NULL
confirm	smallint	NOT NULL
affirmative	char(256)	
negative	char(256)	
confirmtts	char(256)	
global	smallint	NOT NULL
macro	smallint	NOT NULL
startrule	char(256)	
grammarfile	char(256)	

PRIMARY KEY (id)

weblog:

time_stamp	timestamp	NOT NULL
type	smallint	NOT NULL
info	char(2048)	NOT NULL
sequence	smallint	NOT NULL

PRIMARY KEY (time_stamp, sequence)

x10detail:

id	char(1)	NOT NULL
code	smallint	NOT NULL
type	smallint	NOT NULL
key_x	smallint	NOT NULL
key_y	smallint	NOT NULL
key_width	smallint	NOT NULL
key_height	smallint	NOT NULL
key_code	smallint	NOT NULL
key_flag	smallint	NOT NULL
key_text	integer	NOT NULL
key_bkgrnd	integer	NOT NULL
key_border	smallint	NOT NULL

PRIMARY KEY (id, code, type)

x10housecode:

id	char(1)	NOT NULL
description	char(25)	NOT NULL
customtab	char(25)	

PRIMARY KEY (id)

x10types:

typeid	smallint	NOT NULL
description	char(50)	NOT NULL
all_units_off	char(1)	NOT NULL
all_lights_on	char(1)	NOT NULL
all_lights_off	char(1)	NOT NULL
standard_dim	char(1)	NOT NULL
preset_dim	char(1)	NOT NULL
status_request	char(1)	NOT NULL
extended_commands	char(1)	NOT NULL
memory	char(1)	NOT NULL
status_only	char(1)	NOT NULL
on_message	char(25)	NOT NULL
off_message	char(25)	NOT NULL
on_textcolor	integer	NOT NULL
on_backcolor	integer	NOT NULL
off_textcolor	integer	NOT NULL
off_backcolor	integer	NOT NULL
dim_textcolor	integer	NOT NULL
dim_backcolor	integer	NOT NULL
unk_textcolor	integer	NOT NULL
unk_backcolor	integer	NOT NULL
dim_message	char(25)	NOT NULL
unk_message	char(25)	NOT NULL
on_graphic	char(256)	
off_graphic	char(256)	
dim_graphic	char(256)	

unk_graphic	char(256)	

PRIMARY KEY (typeid)

x10unitcode:

id	char(1)	NOT NULL
code	smallint	NOT NULL
key_desc	char(25)	NOT NULL
typeid	smallint	NOT NULL
prev_exclude	smallint	NOT NULL
status	smallint	NOT NULL
level	double	NOT NULL
lastchange	timestamp	NOT NULL
location	char(25)	
groupoffset	smallint	
group1lev	double	
group2lev	double	
group3lev	double	
group4lev	double	
controller	smallint	NOT NULL

PRIMARY KEY (id, code)

x10validcodes:

id	char(4)	NOT NULL
key_number	smallint	NOT NULL
key_desc	char(25)	NOT NULL

PRIMARY KEY (id, key_number)

Appendix C

Last Minute Updates

PowerHome now allows the user to create their own custom colors that can be used anywhere PowerHome requires a color. You can access the custom color maintenance screen from the Maintenance menu. The user will be able to define up to 14 custom colors of their choosing.

The user is now able to define buttons on the Control Center with a foreground color of transparent. Since transparent does not really exist as a foreground color, PowerHome will interpret this and just not display any text on the button. The background color still allows for transparent so truly transparent buttons on top of a background graphic can be obtained.

Placing an * (asterisk) as the first character in a key description field will cause PowerHome NOT to display the key description text on the Control Center button. The text describing the button hotkey WILL display however. If you want no text to display, set the foreground color to transparent.

Appendix D

Changes from 1.00.1

All fields that were not formulas but supported variable substitution are now formulas. This is a major change to a lot of macro commands. A number of commands that supported variable substitution only such as the TTS command required the user to first set a system or global variable to a formula and then place that system or global variable in the variable substitution field. These fields are all formulas now. The database upgrade utility will have already made this conversion for you by placing quotes around your variable substitution fields to convert them to formulas. This should handle the majority of fields but depending upon how the user set the field up, this may require some tweaking.

All Send Keys fields are now formulas. Previously, the only formula Send Keys field was in the macro command Send Keys with a type of formula. Previously, this was the only way to achieve dynamic Send Keys other than using the Send Keys command {global} or {system}. This change extends the power of the Send Keys field and allows truly dynamic Send Keys processing wherever a Send Keys action occurred. The database upgrade utility will have already made the conversion for you by placing quotes around your Send Keys to convert them to formulas. Some tweaking may be required on these fields as well.

PowerHome Server Pages (PSP). The internal web server now includes support for PSP. A PowerHome Server Page is an HTML file with PowerHome formulas embedded within. These formulas are evaluated before the page is served and replaced with the result of the evaluation. This allows the advanced user to create their own dynamic content tailored specifically to what they want to see and how they want to see it. The complete internal set of PowerHome web pages have been written as PSP pages and included in the PSP directory below the default web server directory as samples.

The variable substitution system variables ([X10P0] through [X10P5]) for the Multi-X global buffer have been expanded to access all 20 entries of the multi-x buffer ([X10P0 through X10P19]) and multi-x time buffer ([X10P0TIME] through [X10P19TIME]).

The local system variable for macros has been renamed from MACRO to LOCAL. MACRO still works but will be removed in a later version.

Send Keys and formulas now have access to their own LOCAL system variables.

The LOCAL and TEMP system variables have been doubled. The GLOBAL system variables have been quadrupled. The 5 MACRO variables (now LOCAL variables) have been expanded from 5 to 10. The 5 TEMP variables have been doubled for a total of 10. The 5 GLOBAL variables are now 20. The 5 data retrieval areas have been expanded to 10. All variable substitutions for these expanded variables have been expanded as well.

The internal mechanism of variable passing and storing has been completely rewritten. This should result in a slight speed increase.

A slew of new formula functions. Too many to list here but they all start with ph_ and can be found in the online help.

A new Control Center Design window with more options including drag and drop capability.

Support for the CM11A with fast macros loaded. Previously, PowerHome would not always interpret an incoming X-10 command properly if that X-10 command fired a fast macro stored within the CM11A. This should be fixed.

Virtual X-10 Codes. Housecodes Q through Z can now be declared each with unitcodes 1 through 99. These virtual X-10 codes behave just like real X-10 devices. You can use these instead of Global Variables and see them update in real time on the X-10 status screen.

Proxy server support for freedb and the geturl formula function.

Custom Colors. No longer are you limited to the same generic colors. Define up to 14 colors in whatever shade you like.

The PowerHome Explorer has a small facelift with the ability to set the color scheme. Spice up those dull maintenance screens.

Add horizontal and vertical scrollbars to Control Center tabs if you like.

Transparent foreground colors on Control Center buttons. Create truly transparent buttons so your background graphics show through.

Send Keys buttons have a separate Key Description from the ID now.

NOTE: The following Send Keys commands will be removed in version 1.03.1 as they have been replaced by equivalent formula functions.

{tts}
{delay}
{macro}
{global}
{system}
{setglobal}
{setsystem}
{directsql}
{launch}
{file}
{switchto}
{chggrammar}
{chgspeaker}
{ir}
{x10}
{x10-1}, {x10-2}, {x10-3}, {x10-4}, {x10-5}
{playlist}
{vrdisable}
{x10setstat}
{disconnect}
{connect}
{usermessage}

Please update your Send Keys code if you are currently using any of these functions. All of these inline commands have formula function equivalents which can either be processed during the formula evaluation of Send Keys or processed during the Send Keys interpretation by using the inline command {#}.

NOTE: The following formula functions have been renamed. The old versions still exist but will be removed in version 1.03.2. Please update any formula that use these functions.

f_findata	**renamed to ph_finddata**
f_getdata	**renamed to ph_getdata**
f_geturl	**renamed to ph_geturl**
f_multix	**renamed to ph_multix**
f_multixtime	**renamed to ph_multixtime**

Appendix E

Changes from1.01.1

What happened to version 1.01.2? Version 1.01.2 was a quick bug fix which was made available to a few people who were having problems using the CPU-XA/Ocelot controller. These bugs are now properly fixed in version 1.01.3.

A bug was introduced in version 1.01.1 which prevented the macro commands messagebox and inputbox from returning values properly. This has now been corrected.

A bug with the internal webserver which allowed remote users to possibly access files outside of the webserver directory structure has been fixed.

The communication routines for all controllers has been rewritten. This was done to solve some problems with communication and to handle the special requirements of the CPU-XA/Ocelot controller. Also, speed improvements should be seen on all controllers. Most noticeable will be the speed and reliability of the CM17A "firecracker" module.

Support for the additional Adicon modules associated with the CPU-XA/Ocelot controller has been added. You can now use the SECU-16, SECU-16I, and RLY8-XA modules. Support is enabled for reading digital inputs/outputs and analog inputs, setting digital outputs, and reading/writing the internal CPU-XA/Ocelot variables.

Additional support for the MR26A "mouse remote" has been added. Three new parameters in the pwrhome.ini file for this device are now available. You can now have the X-10 status screen automatically updated for devices with the status only flag set. You can also select which housecodes and X-10 controller you would like to have received commands automatically echoed to.

Seven new formula functions have been added.

ph_postformula allows you to post a formula to the execution queue for later processing.
ph_x10refreshstat will query the database and refresh the X-10 status screen. This is useful for when you use Direct SQL to update the X-10 statuses.
ph_ocelotgetanalog will retrieve an analog input value from the CPU-XA/Ocelot.
ph_ocelotgetdigital will retrieve the status of a digital input/output from the CPU-XA/Ocelot.
ph_ocelotgetvar will retrieve the value of an internal CPU-XA/Ocelot variable.
ph_ocelotsetdigitalout will turn a remote relay associated with a CPU-XA/Ocelot controller either on or off.
ph_ocelotsetvar will set the value of an internal CPU-XA/Ocelot variable.

No changes were made to the database so a database upgrade is not necessary.

Appendix F

Changes from 1.01.3

Fixed an obscure bug where PowerHome would lockup if a toolbar was dragged and dropped onto the menu border.

Added a new Socket Server Interface. This allows users the ability to remotely control PowerHome from other programs either on the same machine or over a network (including the web). These programs do not have to be limited to Windows programs either.

Added a Windows Messaging Interface. This also allows users the ability to remotely control PowerHome from other programs on the same machine. Don't be limited to controlling PowerHome with just one method. Now take your pick of either the Socket Server or Messaging Interface.

Enhanced the Process Email Macro command so that formulas and macros can be executed in addition to sendkeys.

New triggers have been added to handle triggers for the Windows Messaging Interface and the Socket Server.

Added support for Windows Script Host. Choose your language of choice now. You can now control PowerHome with VBScript, JScript, or any other language supported by Windows Script Host.

Added 8 new formula functions: ph_processemail, ph_handle, ph_runscript_0, ph_runscript_1, ph_runscript_2, ph_runscript_3, ph_runscript_4, and ph_runscript_5.

Appendix G

Changes from 1.02.1

Bug Fixes:

Fixed problem with WSH function ph.getsuntime.
Fixed problem with WSH functions ph.getvar_dt and ph.getglobal_dt.
Fixed date problems with WSH functions ph.setglobal_a and ph.setvar_a. Date, time, or datetime will now be formatted as datetime.

Fixed bug in Timed Events maintenance screen where you cannot directly enter a frequency.

Fixed bug in RedRat2 controller where IR codes could not be learned.

Enhancements:

The internal webserver has been enhanced so that you can declare up to 5 trusted IP masks. Requests from these IP's will be treated as if they logged in with the Master password.
Security can now be defined based upon the type of web page.
Enhanced PSP functionality allowing for greater control over the webserver. PSP pages can now set the content-type and the return code. PSP pages also have access to the remote IP, the user authorization level, and the userid and password passed to the webserver.

The Socket Server has been enhanced so that you can restrict access by up to 5 IP masks. Adds an additional level of security.

Added new MR26A INI parameter to ignore CM17A commands.

Added option to have the X-10 Status set to unknown if a status request is sent to an X-10 device that supports status request.

A little more CPU-XA/Ocelot functionality is available via the ph_ocelotgettimer function.

Limited support for SAPI 5.1 TTS.

New formula functions:

ph_sendmsg, ph_postmsg, ph_sendcopydata, ph_postcopydata, ph_ocelotgettimer, ph_base64encode, ph_setwebcontenttype, ph_setwebreturn, ph_getwebremoteip, ph_getwebbase64auth, ph_getwebauth

New WSH functions:

ph.sendmsg, ph.postmsg, ph.sendcopydata, ph.postcopydata, ph.ocelotgettimer, ph.base64encode, ph.setwebcontenttype, ph.setwebreturn, ph.getwebremoteip, ph.getwebbase64auth, ph.getwebauth, ph.setglobal_n, ph.setglobal_d, ph.setglobal_t, ph.setglobal_dt, ph.setvar_n, ph.setvar_d, ph.setvar_t, and ph.setvar_dt

REMINDER: The Send Keys commands and formula functions listed above in 1.02.1 will be deleted in version 1.03.2. Please update your databases before the next release.

Appendix H

Changes from 1.02.2

Fixed bug in MR26A automatic update status routine where all updates were treated as a bright command.

Fixed problem in Control Center and Design mode where buttons were not accessible after scrolling.

Added automatic "SHUTDOWN" macro.

Improved Timed Events!! Can now specify absolute or random offsets as well as sunrise, sunset, dawn, dusk and weekdays, weekends, Mondays, Tuesdays, Wednesdays, Thursdays, Fridays, Saturdays, and Sundays.

Fixed problem in nexturl of execsendkeys where location replacement would not operate properly with proxy servers.

Changed the database connection to DSN-less. Added new ini parameter DBF to handle the change.

Now can read and set the system volume programmatically

The passthru feature of the CM17A is now supported.

Added support for setting timers in the CPU-XA/Ocelot.

PowerHome now minimizes to the system tray.

Added support for the PowerLinc module

Moved most entries in the control table to the pwrhome.ini file

Fixed bug where eventlog would suddenly jump from the first row to the very end while scrolling up with a wheel mouse

Modified internal webserver so that unauthorized guest account access will now load the appropriate page rather than always going to the main page.

Fixed webserver timed event and global variable screens so that values containing single and/or double quotes would not cause errors.

Added ability for user to define up to 5 custom web links to be added to the PowerHome Main internal web page.

Added new timed event frequencies: Daily Sunrise, Daily Sunset, Daily Dawn, Daily Dusk, Weekdays, Weekends, Sundays, Mondays, Tuesdays, Wednesdays, Thursdays, Fridays, Saturdays

Added new functions ph_windowstate, ph_getmode, ph_setmode, ph_getcctab, ph_setcctab, ph_macroparm, ph_macroparmret, ph_openframe, ph_saveframe, ph_sendsmtpemail, ph_getsysvolmin, ph_getsysvolmax, ph_getsysvol, ph_setsysvol, ph_isdaylightsavings, ph_htmlescape, ph_htmlunescape, ph_writefile, ph_saveurl, ph_saveurlviaproxy, ph_deletefile, ph_movefile, ph_copyfile, ph_filelength, ph_fileexists, ph_ismacrowaiting, ph_secondsafter, ph_secondsdiff, ph_extendmacrowait, ph_sendemailfile, ph_sendsmtpemailfile, ph_sqlselectinto, ph_gettickcount, ph_reinitialize, ph_powerhomeuptime, ph_relativetime, ph_backupdb, ph_setfileattrib, ph_rtnes, ph_getotheranalog, ph_getotherdigital, ph_getothertimer, ph_getothervar, ph_setotherdigitalout, ph_setothertimer, ph_setothervar, ph_getx10dt, ph_getx10seconds, ph_killmacrowait, ph_killallmacrowait, ph_minutesafter, ph_relativedatetime, ph_getwebmethod, ph_getwebheader, ph_getwebheaderparm, ph_getweburl, ph_createglobal, ph_deleteglobal, ph_ccdescribe, ph_ccmodify, ph_base64decode, ph_combuffercount, ph_combufferptr,ph_combufferreset, ph_comclose, ph_comdtr, ph_comopen, ph_comrecvchar, ph_comrecvstring, ph_comrts, ph_comsendchar, ph_comsendstring, ph_getccbtnbcolor, ph_getccbtnborder, ph_getccbtnfcolor, ph_getccbtnheight, ph_getccbtntext, ph_getccbtnwidth, ph_getccbtnx, ph_getccbtny, ph_getcclastbtn, ph_getcclasttab, ph_hextonum, ph_lookupccbtn, ph_lookupcctab, ph_numtohex, ph_prontotorr2, ph_rr2topronto, ph_setccbtn, ph_setccbtnbcolor, ph_setccbtnborder, ph_setccbtnfcolor, ph_setccbtnheight, ph_setccbtntext, ph_setccbtnwidth, ph_setccbtnx, ph_setccbtny

Changed function ph_gethtmlcolor to ph_htmlcolor.

Changed function ph_x10btn so that dim value is an integer instead of a string and supports devices that use extended dim.

Fixed problem with base64 encode routine where problems occurred when trying to encode ascii characters greater than 127.

Changed the formula function ph_sendemail and the Macro command Send Email so that multiple recipients can be specified by separating them with a comma.

The Control Center is now available on the web.

Added new ph-cgi functions: ph-cgi/blank, ph-cgi/controlcenter, ph-cgi/ccbutton, and ph-cgi/evalformula.

Removed unactivated links "Macros" and "Triggers" from main PowerHome web page.

Full support for WAP pages with WML have been added to the internal webserver. Access and control PowerHome from your web enabled cell phone.

Added the ability to set the text for dimmed and unknown X-10 status. Also added the ability to choose a graphic which will display on the main web page and X-10 status screen for each of the 4 X-10 status types.

Internal web pages redesigned and include support for CSS.

Added a "Macro Jump Limit" in the Setup->Scripts maintenance screen. This parameter will limit the number of times a macro can jump or Goto a label if the value is greater than 0. Useful for preventing infinite loop situations.

Reinitializing will now fully reset PowerHome and enable any changes made via the PowerHome Explorer. It is not necessary to restart PowerHome in order to see changes now.

Added a new column to macro detail called skip_line. This allows you to flag lines to be skipped for debugging purposes.

The ph_createtimedevent function can be used to created timed events of type formula as well as macros and send keys.

A bug has surfaced within the PowerBuilder environment. The relativetime function returns a time with microsecond precision. If you try to insert this time into a datetime field within the database, it will most likely crash as the database only wants millisecond precision. The new ph_relativetime function fixes this bug by returning the time to milliseconds precision.

Registration routine has been rewritten. You no longer have to contact the author for a new unlock key when you reinstall.

Updated all dropdowns to retrieve data from a new dropdowns table rather than hard-coded internally.

Added a confirm shutdown preference.

The Key number in the IR maintenance screen can now be changed and resequenced.

Added support for the WGLDesigns W800RF32 and W800 modules.

Added a new trigger: X-10 RF In Ext. This adds support for the W800RF32 extended RF commands.

Added support for the RedRat3 IR interface.

Added a new triggernumber for X-10 In and X-10 RF In: Any (After Buffer). This fires after the previous X-10 incoming buffers are updated. The 'Any' triggernumber was also relabeled to Any (Before Buffer).

Changes made to the ph_connect and ph_disconnect functions to handle "Other" controllers.

The macro command "Formula" has been changed so that you can choose either post or immediate processing.

The "last" unit is passed in X-10 Any (on,off,dim,bright) After Buffer triggers and is available in [TEMP3].

The X-10 status window is updated differently so that it does not scroll or refresh when an X-10 status is updated.

The explorer can now connect to PowerHome databases other than the default.

New macro command "End Macro" added.

Enhancements made to the PLAYPLAYLIST and PLAYSLOTTRACK macros.

Fixed the macrowaiting functions to handle resumed macros in the execution queue.

Fixed the [TEMP1] variable for actions in a trigger. It now will show the ID of the trigger.

Additional info in [TEMP6] thru [TEMP9] system variables for actions from a trigger.

Added the ability to select the border for Control Center buttons.

Added ability to email critical error messages to support@myx10.com

Passwords in the pwrhome.ini file are now encrypted.

Appendix I

Changes from 1.03.2

Fixed a security hole for users who allow GUEST access to their PowerHome webserver. If you enable GUEST access on your webserver, then a user with a cellphone or other WAP device could control your X-10 devices using the GUEST account. This is now fixed.

Added an additional webserver parameter ENABLESMARTNEXTURL. With version 1.03.2, web functions that use the NEXTURL will employ a smart retrieve of the NEXTURL HTML rather than returning javascript to perform a “location.replace” function. This parameter allows the user to choose whether to use the new smart nexturl method or the older “location.replace” method.

Changed the algorithm for detecting mouse clicks on the Control Center. The Control Center buttons are built in a specific order starting with IR, then X-10, then Macros, and finally Send Keys. The checking of mouse clicks was performed in the same order. The new algorithm instead checks for mouse clicks in the reverse order of how the buttons were built. This enables one to use “buttons” without code on them as a means to build backgrounds for other buttons.

Appendix J

Changes from 1.03.3

Added the Web Center window.

Added new formula functions to support the Web Center: ph_wcclose, ph_wcgethome, ph_wcgethtml, ph_wcgoback, ph_wcgoforward, ph_wcgohome, ph_wcgourl, ph_wcopen, ph_wcrefresh, ph_wcsethome, ph_wcsethtml, ph_wcshow, ph_wcstop, ph_processemailthread

Changed the ph_setmode function to allow setting the main window in kiosk mode.

Added new Macro command "SubMacro"

Changed the macro commands Message Box and Input Box so that the displayed text is in a multi-line edit control.

Removed toolbar references from the Preferences screen. Toolbar style and locations are now automatically saved and loaded for both the frame and all sheets that have independent toolbars.

Changed functionality so that when in Kiosk mode, if PowerHome is minimized (ph_windowstate), the tray icon is disabled.

Changed the functionality of the ph_windowstate function. If in Kiosk mode, ph_windowstate(0) will return a 1 and NOT set the window to normal. If PowerHome is minimized to the system tray, ph_windowstate will now bring it back with values of 0 and 2.

Added the ability to bypass the timed event startup messages by using the QUIETSTART parameter.

Added new /ph-cgi/relay command. Allows the user the ability to serve web content from multiple servers all via the PowerHome webserver.

Added new /ph-cgi/directhtml command.

Improved the functionality of the ph_geturl and ph_geturlviaproxy functions so they would be more compatible with various websites.