Clipsal by Schneider Electric

C-Gate Manual

Design for C-Gate version 2.10

Document version 1.1.0

Document creation date October 2013

Table of Contents

Section Page

1 Introduction 17

2 About C-Gate 18

2.1 What is C-Gate? 18
2.2 Why use C-Gate? 18
2.3 How does C-Gate work with C-Bus? 18
2.4 When to use C-Gate 19
2.5 When not to use C-Gate 19
2.6 Limitations 20

3 Quick Start Guide 21

3.1 Start Here 21
3.2 Concepts 21
3.3 C-Bus Network 22
3.4 System Requirements 22
3.5 Installing C-Gate 23
3.6 Running C-Gate (Windows Service) 24
3.7 Running C-Gate (Console Application) 24
3.8 Connect To C-Gate 25
3.9 Enter Commands 27
3.10 Working with Projects 27
3.11 Starting a Project or Network 28
3.12 Explore the Network 29
3.13 Operation 30
3.14 Events 32
3.15 Getting Help for Commands 32
3.16 Online Resources 32

4 Reference 34

4.1 C-Gate Launcher (Windows) 34
4.1.1 C-Gate Launcher Configuration 34
4.1.2 New vs Old Launcher 35
4.1.3 C-Gate Launcher Diagnostics 35
4.1.4 C-Gate As A Service 36
4.2 C-Gate Jar 36
4.2.1 Zip Distribution 36
4.2.2 Java Runtime Environment 37
4.2.3 Manual Launch 37
4.3 Command and Monitoring Interfaces 38
4.3.1 Command Interface 39
4.3.1.1 Console Connection Details 39
4.3.1.2 TCP/IP Connection Details 40

4.3.1.3 Commands and Replies 40
4.3.1.4 Commands 41
Terminology 41
Objects and Addresses 41
Array Filtering 44
Unique Command IDs 46
Adding IDs to commands 46
Background processing 46
Verbose output 47
Responses 47
Example of Command ID 47
4.3.1.5 C-Gate Response Codes 47
General Response Codes 49
Pre-Release Debugging Information 50
Successful Completion Messages 50
Security Messages 50
Object Information Messages 51
Client Side Command Errors 52
Macro Errors 52
Access Control Errors 53
Method Failures 53
Internal Errors 54
Network Errors 54
Continuation Confirmation 54
4.3.2 Event Interface 54
4.3.2.1 Event Buffer Overflow 55
4.3.2.2 Setting the event level 55
4.3.2.3 Event Table 56
Information Event Codes 56
Energy Related 56
Command Events 57
Network Information 57
PCI Synchronisation 58
Scene and License 59
Unit Configuration Warnings 59
Debugging Information 59
Command Related 59
Security Related 60
Non Critical Network Errors 60
Critical Alarms 62
Connected Network Alarms 62
Interface Errors 62
File errors 63
Licensing Errors 63
Unit Specification Errors 63
Scene Errors 64
Access Control 64
4.3.3 Status Change Port 65
4.3.4 Config Change Port 65
4.3.5 Session ID and Command ID 65
4.3.6 SSL Connection 65
4.4 Logging 66

4.4.1 C-Gate logs 66
4.4.2 New vs old C-Gate log 67
4.4.3 Installer logs 68
4.4.4 Launcher logs 68
4.5 Command Descriptions 68
4.5.1 # 68
4.5.2 // 69
4.5.3 AIRCON 69
4.5.4 AIRCON REFRESH 70
4.5.5 AIRCON SET_HUMIDITY_SETBACK_LIMIT 71
4.5.6 AIRCON SET_HUMIDITY_LOWER_GUARD_LIMIT 72
4.5.7 AIRCON SET_HUMIDITY_UPPER_GUARD_LIMIT 73
4.5.8 AIRCON SET_HVAC_LOWER_GUARD_LIMIT 74
4.5.9 AIRCON SET_HVAC_SETBACK_LIMIT 75
4.5.10 AIRCON SET_HVAC_UPPER_GUARD_LIMIT 76
4.5.11 AIRCON SET_WARD_OFF 77
4.5.12 AIRCON SET_WARD_ON 77
4.5.13 AIRCON SET_ZONE_HUMIDITY_MODE 78
4.5.14 AIRCON SET_ZONE_HVAC_MODE 79
4.5.15 APIVER 81
4.5.16 AUDIO 82
4.5.17 AUDIO CURRENT_FEED 83
4.5.18 AUDIO DYNAMIC_1 84
4.5.19 AUDIO DYNAMIC_2 85
4.5.20 AUDIO HIGH_PRIORITY 86
4.5.21 AUDIO MUTE 87
4.5.22 AUDIO NEXT_FEED 87
4.5.23 AUDIO NEXT_LANGUAGE 88
4.5.24 AUDIO OFF 89
4.5.25 AUDIO ON 90
4.5.26 AUDIO OUTPUT_COMMON_CONTROL 90
4.5.27 AUDIO OUTPUT_DEVICE_STATUS_REQUEST 91
4.5.28 AUDIO OUTPUT_ERROR_CODE 92
4.5.29 AUDIO PREVIOUS_FEED 93
4.5.30 AUDIO RAMP 93
4.5.31 AUDIO REQUEST_CURRENT_FEED 95
4.5.32 AUDIO SET_FEED 95
4.5.33 AUDIO TERMINATERAMP 96
4.5.34 AUDIO ZONE_DESCRIPTOR_REQUEST 97
4.5.35 AUDIO ZONE_FEED_LABEL_REQUEST 98
4.5.36 BROADCAST_EVENT 98
4.5.37 CGL 99
4.5.38 CGL EXPORT 100
4.5.39 CGL IMPORT 101
4.5.40 CLOCK 102
4.5.41 CLOCK DATE 102

4.5.42 CLOCK REQUEST_REFRESH 103
4.5.43 CLOCK TIME 104
4.5.44 CONFIG 105
4.5.45 CONFIG GET 105
4.5.46 CONFIG INFO 106
4.5.47 CONFIG LOAD 107
4.5.48 CONFIG SAVE 108
4.5.49 CONFIG SET 109
4.5.50 CONFIG OBGET 110
4.5.51 CONFIG OBSET 110
4.5.52 CONFIG OBRESET 111
4.5.53 CONFIRM 112
4.5.54 DBADD 113
4.5.55 DBADDSAFE 113
4.5.56 DBCOPY 115
4.5.57 DBCOPYSAFE 116
4.5.58 DBCREATE 118
4.5.59 DBCREATENET 118
4.5.60 DBDELETE 119
4.5.61 DBGET 120
4.5.62 DBGETXML 121
4.5.63 DBLOAD 122
4.5.64 DBNETWORKPATH 123
4.5.65 DBNEW 124
4.5.66 DBRENAMENET 125
4.5.67 DBRENAMENETSAFE 125
4.5.68 DBSAVE 127
4.5.69 DBSET 127
4.5.70 DBSETSAFE 128
4.5.71 DBSETXML 129
4.5.72 DBTAGLIST 130
4.5.73 DBUPDATE 131
4.5.74 DBVALIDATE 132
4.5.75 DBVERIFY 132
4.5.76 DO 133
4.5.77 ENABLE 134
4.5.78 ENABLE LABEL 134
4.5.79 ENABLE REMOVE 136
4.5.80 ENABLE SET 137
4.5.81 EREPORT 137
4.5.82 EREPORT MESSAGE 138
4.5.83 EVENT 139
4.5.84 GET 140
4.5.85 GETSTATE 141
4.5.86 HELP 142
4.5.87 LIGHTING 143

4.5.88 LIGHTING LABEL 143
4.5.89 LIGHTING UNICODELABEL 145
4.5.90 LIGHTING OFF 147
4.5.91 LIGHTING ON 147
4.5.92 LIGHTING RAMP 147
4.5.93 LIGHTING TERMINATERAMP 147
4.5.94 LOCK 147
4.5.95 LOGIN 148
4.5.96 LOGOUT 148
4.5.97 MEASUREMENT 149
4.5.98 MEASUREMENT DATA 149
4.5.99 MEDIATRANSPORT 150
4.5.100 MEDIATRANSPORT CATEGORY_NAME 151
4.5.101 MEDIATRANSPORT ENUMERATE 152
4.5.102 MEDIATRANSPORT ENUMERATION_SIZE 153
4.5.103 MEDIATRANSPORT FORWARD 153
4.5.104 MEDIATRANSPORT NEXT_CATEGORY 154
4.5.105 MEDIATRANSPORT NEXT_SELECTION 155
4.5.106 MEDIATRANSPORT NEXT_TRACK 156
4.5.107 MEDIATRANSPORT PAUSE 156
4.5.108 MEDIATRANSPORT PLAY 157
4.5.109 MEDIATRANSPORT REPEAT 158
4.5.110 MEDIATRANSPORT REWIND 158
4.5.111 MEDIATRANSPORT SELECTION_NAME 159
4.5.112 MEDIATRANSPORT SET_CATEGORY 160
4.5.113 MEDIATRANSPORT SET_SELECTION 161
4.5.114 MEDIATRANSPORT SET_TRACK 161
4.5.115 MEDIATRANSPORT SHUFFLE 162
4.5.116 MEDIATRANSPORT SOURCE_POWER 163
4.5.117 MEDIATRANSPORT STATUS_REQUEST 163
4.5.118 MEDIATRANSPORT STOP 164
4.5.119 MEDIATRANSPORT TOTAL_TRACKS 165
4.5.120 MEDIATRANSPORT TRACK_NAME 165
4.5.121 NET 166
4.5.122 NET CHECKUNIT 167
4.5.123 NET CLOCKS 168
4.5.124 NET CLOSE 170
4.5.125 NET CREATE 171
4.5.126 NET DELETE 172
4.5.127 NET FLUSH 173
4.5.128 NET LEARN 173
4.5.129 NET LIST 174
4.5.130 NET LIST_ALL 175
4.5.131 NET LOAD 176
4.5.132 NET OPEN 177
4.5.133 NET PINGU 177

4.5.134 NET PROJECT_IDENTIFY 178
4.5.135 NET RENAME 179
4.5.136 NET SAVE 180
4.5.137 NET SET_PROJECT_IDENTIFY 181
4.5.138 NET SYNC 181
4.5.139 NET SYNCNEW 183
4.5.140 NET UNRAVELUNIT 184
4.5.141 NET UNRAVEL 185
4.5.142 NETWORK 187
4.5.143 NETWORK LOCATE 188
4.5.144 NEW 189
4.5.145 NOOP 190
4.5.146 OFF 190
4.5.147 OID 191
4.5.148 ON 192
4.5.149 PORT 192
4.5.150 PORT CNISCAN 193
4.5.151 PORT CNISCAN2 194
4.5.152 PORT IFLIST 195
4.5.153 PORT LIST 196
4.5.154 PORT PROBE 197
4.5.155 PORT REFRESH 198
4.5.156 PROJECT 199
4.5.157 PROJECT CLOSE 199
4.5.158 PROJECT COPY 200
4.5.159 PROJECT DELETE 201
4.5.160 PROJECT DIR 202
4.5.161 PROJECT LIST 202
4.5.162 PROJECT LOAD 203
4.5.163 PROJECT NEW 204
4.5.164 PROJECT RENAME 205
4.5.165 PROJECT REPAIR 206
4.5.166 PROJECT RESTORE 206
4.5.167 PROJECT SAVE 207
4.5.168 PROJECT START 208
4.5.169 PROJECT STOP 209
4.5.170 PROJECT USE 209
4.5.171 QUIT 210
4.5.172 RAMP 211
4.5.173 REPORT 212
4.5.174 RUN 212
4.5.175 SCENE 213
4.5.176 SECURITY 214
4.5.177 SECURITY ARM 214
4.5.178 SECURITY DISPLAY_MESSAGE 215
4.5.179 SECURITY EMULATE_KEYPAD 216

4.5.180 SECURITY RAISE_ ALARM ... 216
4.5.181 SECURITY REQUEST_ZONE_NAME ... 217
4.5.182 SECURITY STATUS_REQUEST ... 218
4.5.183 SECURITY TAMPER ... 218
4.5.184 SESSION_ID ... 219
4.5.185 SESSION_ID ALL ... 220
4.5.186 SESSION_ID TAG ... 221
4.5.187 SET ... 221
4.5.188 SHOW ... 222
4.5.189 SHUTDOWN ... 222
4.5.190 SHORTMESSAGE ... 223
4.5.191 SHORTMESSAGE REFRESH ... 224
4.5.192 SHORTMESSAGE SEND ... 225
4.5.193 STOP ... 226
4.5.194 TELEPHONY ... 227
4.5.195 TELEPHONY CLEAR_DIVERSION ... 227
4.5.196 TELEPHONY DIVERT ... 228
4.5.197 TELEPHONY ISOLATE_SECONDARY_OUTLET ... 228
4.5.198 TELEPHONY RECALL_LAST_NUMBER_REQUEST ... 229
4.5.199 TELEPHONY REJECT_INCOMING_CALL ... 230
4.5.200 TEMPERATURE ... 231
4.5.201 TEMPERATURE BROADCAST ... 231
4.5.202 TERMINATERAMP ... 232
4.5.203 TEST_SPAM ... 233
4.5.204 TEST_SPAM EREPORT ... 233
4.5.205 TEST_SPAM LIGHTING ... 234
4.5.206 TEST_SPAM LIST ... 235
4.5.207 TEST_SPAM STOP ... 236
4.5.208 TOPOLOGY ... 237
4.5.209 TOPOLOGY EXPLORE ... 237
4.5.210 TREE ... 238
4.5.211 TREEXML ... 239
4.5.212 TREEXMLDETAIL ... 240
4.5.213 TRIGGER ... 242
4.5.214 TRIGGER EVENT ... 243
4.5.215 TRIGGER LABEL ... 243
4.5.216 TRIGGER UNICODELABEL ... 245
4.5.217 UNLOCK ... 246
4.6 Configuration ... 247
4.6.1 Configuration File ... 248
4.6.2 Scope and Scope Objects ... 248
4.6.3 Configuration Commands ... 249
4.6.4 Configuration Parameters ... 249
4.6.4.1 accept-connections-from ... 249
4.6.4.2 access-control-file ... 250
4.6.4.3 allow-fast-start ... 250

4.6.4.4 allow-v3-pci 251
4.6.4.5 application.catalog.filename 251
4.6.4.6 application.catalog.directory 252
4.6.4.7 auto-reopen 252
4.6.4.8 cbus-application 253
4.6.4.9 cbus-tx-delay 253
4.6.4.10 cbus.tx-cache 254
4.6.4.11 ccp.display-oids 254
4.6.4.12 ccp.display-state 255
4.6.4.13 cgate-name 255
4.6.4.14 clock.master 256
4.6.4.15 clock.mastermode 256
4.6.4.16 clock.update-interval 257
4.6.4.17 cgroups-file 257
4.6.4.18 command-local-address 257
4.6.4.19 command-port 258
4.6.4.20 command.encoding 258
4.6.4.21 comms-debug 259
4.6.4.22 config-change-port 259
4.6.4.23 config-path 260
4.6.4.24 console.enable-commands 260
4.6.4.25 default-tag-db 261
4.6.4.26 enable.save-state 261
4.6.4.27 event-filename 262
4.6.4.28 event-file.event-level 262
4.6.4.29 event-file.keep-days 263
4.6.4.30 event-file.split 263
4.6.4.31 event-file.split-count 263
4.6.4.32 event-file.split-size 264
4.6.4.33 event-host 264
4.6.4.34 event-millis 265
4.6.4.35 event-mode 265
4.6.4.36 event-port 266
4.6.4.37 event-printer 266
4.6.4.38 event.display-oids 267
4.6.4.39 file.base 267
4.6.4.40 global-event-level 268
4.6.4.41 heartbeat-time 268
4.6.4.42 hide-project-names 269
4.6.4.43 instance.lock-file 269
4.6.4.44 instance.lock-timeout 269
4.6.4.45 lighting.learn-update 270
4.6.4.46 load-change-port 270
4.6.4.47 local-flow-control 271
4.6.4.48 macro-path 271
4.6.4.49 memory-report 272

4.6.4.50 network.application-connect 272
4.6.4.51 network.error.commands-failed 273
4.6.4.52 network.error.units-failed 273
4.6.4.53 network.error.units-failed-hysteresis 274
4.6.4.54 network.pci.poll-interval 274
4.6.4.55 network.retries 275
4.6.4.56 network.retries.pci-check 275
4.6.4.57 network.source 275
4.6.4.58 network.state-interval 276
4.6.4.59 networks-file 276
4.6.4.60 patch.archive-file 277
4.6.4.61 pci-flow-control 277
4.6.4.62 pci.local-sal 278
4.6.4.63 pp.spec-base-directory 278
4.6.4.64 project.default 279
4.6.4.65 project.default.archive-dir 279
4.6.4.66 project.default.dir 279
4.6.4.67 project.start 280
4.6.4.68 reopen-delay 280
4.6.4.69 report-new-objects 281
4.6.4.70 response-delay 281
4.6.4.71 scene-base 282
4.6.4.72 secure.bind-address 282
4.6.4.73 secure.port-base 283
4.6.4.74 serial.fixbaud 283
4.6.4.75 speed-write 284
4.6.4.76 startup-delay 284
4.6.4.77 sweep-timeout 285
4.6.4.78 sync-fast-pci 285
4.6.4.79 sync-time 286
4.6.4.80 sync.free-periods 286
4.6.4.81 sync.global-pool-size 287
4.6.4.82 sync.padding.enabled 288
4.6.4.83 sync.padding.sync-time-factor 288
4.6.4.84 sync.gateway-pool-size 289
4.6.4.85 tag-autosave 289
4.6.4.86 tag-name-output 290
4.6.4.87 tag-use-zip 290
4.6.4.88 tag-validate-db 291
4.6.4.89 unit-auto-update-db 291
4.6.4.90 unit-auto-delete 291
4.6.4.91 unitcatalog.directory 292
4.6.4.92 unitcatalog.filename 292
4.6.4.93 use-1.0-addressing 293
4.6.4.94 use-cgroups 293
4.6.4.95 use-config-change-port 294

4.6.4.96 use-event-file 294
4.6.4.97 use-load-change-port 295
4.6.4.98 use-queue-sweeper 295
4.6.4.99 use-scenes 295
4.6.4.100 use-tags 296
4.7 C-Bus Applications 296
4.7.1 Specifying Applications 296
4.7.1.1 applications.xml 297
4.7.1.2 applications.spe override 300
4.7.2 Airconditioning Application 300
4.7.2.1 Airconditioning Application Overview 301
4.7.2.2 Objects 301
4.7.2.3 Events 302
Status Events 302
Humidity Schedule Entry 302
HVAC Schedule Entry 303
Zone Humidity Plant Status 304
Zone HVAC Plant Status 305
Zone Humidity 307
Zone Temperature 307
Command Events 308
Refresh 308
Set Ward On 309
Set Ward Off 309
Set Zone HVAC Mode 309
Set HVAC Upper Guard Limit 310
Set HVAC Lower Guard Limit 311
Set HVAC Setback Limit 311
Set Zone Humidity Mode 312
Set Humidity Upper Guard Limit 312
Set Humidity Lower Guard Limit 313
Set Humidity Setback Limit 313
4.7.2.4 Commands 314
4.7.2.5 Specification 314
4.7.3 Audio Application 315
4.7.3.1 Audio Application Overview 315
4.7.3.2 Objects 316
4.7.3.3 Events 316
Current Feed 316
Dynamic 1 317
Dynamic 2 317
High Priority 318
Mute 318
Next Feed 319
Next Language 319
Off 320
On 320
Output Common Control 321
Output Device Status Request 321
Output Error Code 322
Previous Feed 323

Ramp 323
Request Current Feed 324
Set Feed 324
Terminate Ramp 325
Zone Descriptor Request 326
Zone Feed Label Request 326
4.7.3.4 Commands 327
4.7.3.5 Specification 327
4.7.4 Clock and Timekeeping Application 327
4.7.4.1 Application Overview 327
4.7.4.2 Objects supported 328
4.7.4.3 Methods 328
4.7.4.4 Commands 328
4.7.4.5 Events issued 329
4.7.4.6 Status Change Port 329
4.7.4.7 Network time 329
4.7.4.8 Configuration properties 329
4.7.4.9 Usage Notes 329
4.7.4.10 Specification 329
4.7.5 Enable Control Application 329
4.7.5.1 Enable Introduction 329
4.7.5.2 Enable Application Overview 330
4.7.5.3 Persistent storage 330
4.7.5.4 Objects supported 330
Enable Control Application Object 330
Enable Network Variables 330
4.7.5.5 Commands supported 331
4.7.5.6 Events issued 331
4.7.5.7 Status Change Port Reporting 331
4.7.5.8 Project database 331
4.7.5.9 Usage Notes 332
4.7.5.10 Enable Specification 332
4.7.6 Error Reporting Application 332
4.7.6.1 Application Overview 332
4.7.6.2 Objects Supported 332
4.7.6.3 Send Event 333
4.7.7 Measurement Application 333
4.7.7.1 Specification 333
4.7.7.2 Application Overview 333
4.7.7.3 Objects Supported 334
Measurement Application Object 334
Commands Supported 334
MEASUREMENT ? 335
Events Supported 335
Object Parameters 335
Measurement Device Object 336
Commands Supported 336
Events Supported 336

Object Parameters 336
Channel Object 336
Commands Supported 337
MEASUREMENT DATA 337
Events Supported 337
Measurement Data 337
Object Parameters 338
4.7.7.4 Events Issued 339
Data 339
4.7.7.5 Unit Code Table 339
4.7.8 Media Transport Control Application 340
4.7.8.1 Application Overview 341
4.7.8.2 Objects Supported 342
4.7.8.3 Events Issued 342
Category Name 342
Enumerate 343
Enumeration Size 344
Forward 344
Next Selection 345
Next Category 345
Next Track 346
Pause 347
Play 347
Repeat 348
Rewind 348
Selection Name 349
Set Selection 350
Set Category 350
Set Track 351
Shuffle 351
Source Power Control 352
Status Request 352
Stop 353
Track Name 353
4.7.9 Security Application 354
4.7.9.1 Security Application Overview 354
4.7.9.2 Objects Supported 355
Security Application Object 355
Commands Supported 356
Events Supported 356
System Armed/Disarmed 356
System Disarmed 356
Exit Delay Started 357
Entry Delay Started 357
Alarm On 358
Alarm Off 358
Tamper Off 358
Tamper On 359
Current Alarm Type 359
Panic Cleared 360
Panic Activated 360
Low Battery Corrected 361
Low Battery Detected 361

Battery Charging ... 362
Status Report 1 ... 362
Status Report 2 ... 363
Password Entry Status ... 364
Mains Failure ... 365
Mains Restored ... 365
Arm Ready ... 365
Line Cut Alarm Raised ... 366
Line Cut Alarm Cleared ... 366
Fire Alarm Raised ... 367
Fire Alarm Cleared ... 367
Arm Failed Raised ... 367
Arm Failed Cleared ... 368
Gas Alarm Raised ... 368
Gas Alarm Cleared ... 369
Other Alarm Raised ... 369
Other Alarm Cleared ... 370
Object Parameters ... 370
Security Zone Object ... 372
Commands Supported ... 372
Events Supported ... 372
Zone Unsealed ... 372
Zone Sealed ... 373
Zone Open ... 373
Zone Short ... 373
Zone Isolated ... 374
Zone Name ... 374
Arm Not Ready ... 375
Object Parameters ... 375
4.7.9.3 Specification ... 376
4.7.10 Short Message Application ... 376
4.7.10.1 Short Message Overview ... 376
4.7.10.2 Events Issued ... 377
Short Message Refresh ... 377
Short Message Send ... 378
4.7.11 Telephony Application ... 379
4.7.11.1 Telephony Application Overview ... 379
4.7.11.2 Objects supported ... 379
Telephony Application Object ... 379
4.7.11.3 Commands supported ... 380
4.7.11.4 Events supported ... 380
Line On Hook ... 380
Line Off Hook ... 380
Dial Out Failure ... 381
Dial In Failure ... 381
Ringing ... 381
Recall Last Number Response ... 382
Internet Connection Request Made ... 382
Isolate Secondary Outlet ... 382
Recall Last Number Request ... 383
Reject Incoming Call ... 383
Divert ... 383
Clear Diversion ... 384

Configuration properties ... 384
4.7.11.5 Usage Notes ... 384
4.7.11.6 Specification ... 384
4.7.12 Temperature Broadcast Application ... 384
4.7.12.1 Application Overview ... 384
4.7.12.2 Objects Supported ... 385
Temperature Broadcast Application Object ... 385
Commands Supported ... 386
Events Supported ... 386
Object Parameters ... 386
Temperature Group Object ... 386
Commands Supported ... 387
Events Supported ... 387
Temperature Broadcast ... 387
Object Parameters ... 387
4.7.12.3 Specification ... 388
4.7.13 Trigger Control Application ... 388
4.7.13.1 Trigger Overview ... 388
4.7.13.2 Trigger Objects Supported ... 388
Trigger Application Object ... 388
Trigger Group Object ... 389
4.7.13.3 Trigger Events ... 389
Trigger Commands ... 389
4.7.13.4 Status Change Port ... 389
4.7.13.5 Configuration ... 389
4.7.13.6 Project Database Issues ... 389
4.7.13.7 Usage Notes ... 390
4.7.13.8 Trigger Specification ... 390
4.8 Projects ... 390
4.8.1 Concept of Project ... 390
4.8.2 Projects And Tags ... 390
4.8.3 Project Names ... 390
4.8.4 Projects And Networks ... 390
4.8.5 Addressing ... 390
4.8.6 Default Project ... 391
4.8.7 Current Project ... 391
4.8.8 Project Properties ... 391
4.8.9 Access Control Issues ... 391
4.9 Networks ... 391
4.9.1 Opening a Network ... 392
4.9.1.1 Network Interface State Diagram ... 393
4.9.2 Network State ... 393
4.9.2.1 Network State Diagram ... 395
4.9.3 Networks In Error ... 396
4.9.4 Network Syncing ... 396
4.9.4.1 Calculation of NextSyncTime ... 398
4.9.4.2 Sync Padding ... 398
4.9.4.3 Sync-Free Periods ... 399

4.9.5 Unit Syncing ... 400
4.9.5.1 Unit State Diagram ... 401
4.9.6 Command Retries ... 401
4.9.7 C-Bus Clock Recovery ... 402
4.9.8 PCI Connection Check ... 402
4.9.9 PCI Polling ... 403
4.9.10 Unit Online Status ... 404
4.9.10.1 Unit Online Status Diagram ... 406
4.10 Access Control ... 407
4.10.1 Access Control Basics ... 407
4.11 CGL Format ... 408
4.11.1 CGL Specification v1.0 ... 408
4.11.1.1 CGL Specification v1.0 Example ... 408
4.11.1.2 CGL Specification v1.0 Constraints ... 409
4.12 Object Overview ... 412
4.12.1 Working with Objects ... 412
4.12.2 All Objects ... 414
4.12.3 CGate Object ... 415
4.12.4 Projects ... 416
4.12.5 CBusNetworkManager ... 416
4.12.6 CBusNetwork ... 416
4.12.7 CBusUnit ... 422
4.12.8 CBus Output Units ... 424
4.12.9 CBus Output Terminal ... 425
4.12.10 DIN and Pro Dimmers and Relays ... 425
4.12.11 CBus Din/Pro Terminal ... 426
4.12.12 CBus Temperature Sensor ... 426
4.12.13 CBus PE Cell / Light Level Sensor ... 427
4.12.14 CBusApplication ... 428
4.12.15 CBusGroup ... 428
4.13 Building Clients for C-Gate ... 430
4.13.1 Communicating with C-Gate ... 430
4.13.2 Controlling Communications with C-Gate ... 430
4.13.3 Multiple Clients ... 431
4.13.4 Logging with C-Gate ... 431
5 Version History ... 432
5.1 v2.11.6 ... 432
5.2 v2.11.5 ... 432
5.3 v2.11.4 ... 432
5.4 v2.11.3 ... 432
5.5 v2.11.2 ... 432
5.6 v2.11.1 ... 432
5.7 v2.11.0 ... 432
5.8 v2.10.6_3169 ... 433
5.9 v2.10.6 ... 433

5.10 v2.10.5 433
5.11 v2.10.2 433
5.12 v2.10.0 433
5.13 v2.9.8 434
5.14 v2.9.7 434
5.15 v2.9.5 434
5.16 v2.9.4 434
5.17 v2.9.2 435
5.18 v2.9.0 435
5.19 Upgrade from 1.5 to 2.5 435
5.19.1 Major changes from 1.5 to 2.x 435
5.19.2 Setting 1.5 compatibility 436
6 Not Used 438
6.1 Set Plant HVAC Level 438
6.2 Set Plant Humidity Level 438
Index 440

1 Introduction

The C-Gate Manual gives basic information about installing and operating the C-Gate server. This is the complete reference documentation for working with C-Gate.

Documentation History

This manual was newly released with version 2.5 of the C-Gate server.
It combines the previously released C-Gate Server Reference and C-Gate User Guide, plus newer material like the Quick Start Guide into a new C-Gate Manual.

For deep detail, the Reference section is the place to begin.

Or, for an overview, keep reading into About C-Gate.

If you are building client software to integrate with C-Gate, please see the Building Clients for C-Gate section.

Media

This manual is available as a Adobe® Acrobat® PDF or Microsoft® HTML help. Always get the latest version from the Clipsal Integrated Systems website http://www2.clipsal.com/cis/technical/downloads

Version

This document is designed for C-Gate version 2.10.
This is version C-Gate Manual, created on October 2013.

Copyright

Trademarks

Disclaimer

Clipsal Integrated Systems reserves the right to change specifications or designs described in this manual without notice and without obligation.

2 About C-Gate

2.1 What is C-Gate?

C-Gate is a software package that monitors and controls the components of the Clipsal C-Bus wiring system.

C-Gate runs on a PC or server platform to provide high-speed monitoring and control of multiple C-Bus networks.

The C-Gate software is either run on a separate server or in the background on a PC, to allow other C-Bus front end software or building management systems to have high-speed, high level control and monitoring of C-Bus.

C-Gate uses industry-standard TCP/IP interfaces to support:

- **Multiple C-Bus networks** - connected to a TCP/IP backbone network using TCP/IP terminal servers
- **Multiple connections** - from one or more front-end or building management systems using TCP/IP sockets
- **Simple connection** - to web servers for Internet-based C-Bus control and monitoring.

A lot of flexibility comes from using the TCP/IP standard, opening up the ability to control, connect to, monitor and Internet-enable C-Bus.

Internally, C-Gate builds an object model of the C-Bus Networks that it is controlling and monitoring.

It uses the smart monitoring features of C-Bus to fully simulate the operation of C-Bus input and output devices.

So, C-Gate can rapidly give up-to-date information about the status of C-Bus without having to slowly poll individual devices.

2.2 Why use C-Gate?

C-Gate allows you to connect one or more networks with C-Bus devices to other software or systems for control or monitoring.

C-Gate's high level Command and Event Interfaces mean that you don't need to go into the details of the C-Bus protocol in order to interface to C-Bus.

C-Gate allows you to connect to C-Bus networks across a TCP/IP backbone network as well as through a local RS232 interface or via a modem or other mechanisms.

A number of C-Bus networks can be managed in parallel at high speed, meaning that control is rapid and monitoring is accurate - even where a number of networks are involved.

2.3 How does C-Gate work with C-Bus?

The C-Gate server software opens a connection to each C-Bus network it is going to manage.

To connect to a C-Bus network, C-Gate tries to open a connection to a C-Bus Network through one of the available connection methods, which include:

- C-Bus PC Interface (model 5100PC)
- C-Bus Network Interface (model 5500CN)
- via a modem to a C-Bus Telephone Interface
- via a TCP/IP socket connection to a terminal server

Once connected, C-Gate scans each C-Bus network and builds an object model of the network and the network's C-Bus units. This model is the base that C-Gate uses for control and monitoring.

Once this initial model is established, C-Gate listens to each network, receiving monitoring events from the network that will update the model with the latest state of the network.

An event is received by C-Gate every time a button is pushed or every time a C-Bus Group Level changes as a result of a button push or activity from a sensor or other C-Bus device.

To ensure the C-Gate model of the network is up to date, the network is re-scanned on a continuous basis. This also ensures that the network is still connected and communicating with the C-Gate server.

Applications or building management systems connect to C-Gate's Command Interface and can issue commands (such as ON, OFF and RAMP and commands associated with other C-Bus Applications) and query the status of any object (using the GET and SET commands) in C-Gate's model of each network.

C-Gate also provides several event,status and configuration interfaces to allow applications or building management systems to receive a continuous real-time list of events that have occurred on the C-Bus networks that C-Gate is managing.

2.4 When to use C-Gate

Use the C-Gate server when:

- You need to accurately monitor and control one or more C-Bus networks
- You need high performance control of one or more C-Bus networks
- You need to control the C-Bus network from another Building management or control system
- You want to use a custom front-end or user interface to C-Bus, such as C-Lution or Citect
- You want to provide an Internet or TCP/IP interface to control and monitor C-Bus networks
- Control based on a C-Bus network connected by bridges is too slow
- You want to monitor status of the C-Bus devices on a C-Bus network.

2.5 When not to use C-Gate

Don't use the C-Gate server when:

- You are implementing a small, stand-alone C-Bus network that has **no need** for any external control or monitoring or any interface to a computer or other devices.

2.6 Limitations

The C-Gate server has the following limitations that should be considered when using C-Gate to control and monitor C-Bus networks:

- There is no absolute maximum number of networks that C-Gate can interface to. A reasonable operating maximum is 64 C-Bus networks.

- Above 64 networks, server configuration will have to be carefully arranged to ensure C-Gate will perform reasonably. Please contact Clipsal Integrated Systems Pty. Ltd. for this configuration information

3 Quick Start Guide

Start Here...

To quickly get C-Gate up and running for you, start reading here. We'll guide you through connecting to and controlling C-Bus from C-Gate in just a few minutes.

Proceed to Start Here

3.1 Start Here

Here are your basic steps to start working with C-Bus and C-Gate:

1. Understand some basic Concepts about C-Bus and C-Gate
2. Find a C-Bus Network to work with
3. Install C-Gate, if not already installed
4. Run C-Gate, if not installed as a Windows Service
5. Connect to C-Gate
6. Enter Commands
7. Working with Projects
8. Starting a Project or Network
9. Explore the Network
10. Operation
11. Events
12. Getting Help for Commands
13. Online Resources

Then, dive into the Reference section for the details of C-Gate's commands, events, operation, C-Bus applications and many more things.

3.2 Concepts

There are a few concepts we need to explain to get started.

When we talk about **C-Bus Devices** and **C-Bus Units**, we mean hardware modules that are designed to connect to other **C-Bus Devices** using C-Bus cable.

When a group of C-Bus Devices (like switches, dimmers and other devices) are wired together with C-Bus cable, we call this a **C-Bus Network**.

Two C-Bus Networks can be connected together using a C-Bus Bridge. A bridge has connections to two separate C-Bus Networks.

A Project is a group of C-Bus Networks grouped together, and operated together by C-Gate or C-Bus Toolkit. Think of a Project as the set of C-Bus Networks used in, for example, a house. The C-Bus Networks in the house work together to automate the house, so they are grouped together as a project. Find out more about Projects in the Projects section of the Reference.

Read on to find out more about C-Bus Networks.

3.3 C-Bus Network

In order to start working with C-Gate, you will need a C-Bus Network to control.

You can connect C-Gate to the C-Bus network in two common ways:

1. **Using a PC Interface (PCI) connected to your computer's serial port**

   Connect your C-Bus network's PCI to a serial port on your computer, or via a USB to RS232 Serial converter. You should have a cable It is best to know the name of the serial port you are connecting to. The name will be something like COM*x* where x is a number. Typically it is COM1, COM2, COM3, or even something like COM7 or COM12.

2. **Using a C-Bus Network Interface (CNI) connected to an Ethernet network that your computer is connected to**

   You'll need to know the IP address of the CNI you want to connect to. Refer to the CNI documentation and software to find out or set the IP address of your CNI. You can also try and discover your CNI's IP address using C-Gate commands, which will be explained in Running C-Gate.

3.4 System Requirements

To install and operate C-Gate, your system should preferably have the following as a minimum:

Hardware

- Pentium 4 processor, 2 GHz OR Core 2 Duo processor E6300 or better.
- 512 MB of RAM.
- 5 GB of free disk space.
- CD/ROM drive (for software installation from CD only).
- Mouse.
- Colour display capable of 800 x 600 resolution.
- USB port for use with a USB/serial converter OR a DB9 standard serial port.
- Network adaptor 10 Mbps or better (optional).

Software

- Microsoft Windows XP, Service Pack 2 or later.
- Adobe Acrobat Reader v4.0 (or higher) required for viewing and printing the Online Documentation.

- Windows Media Player v7 or higher, required for viewing the Video Presentation included with the Online Documentation.

- Anti-virus software.

System RAM, CPU and network resource requirements are increased as more C-Bus networks are opened. The requirements quoted here are for small-scale C-Gate systems accessing 8 or less networks.

3.5 Installing C-Gate

Installation

The easiest way to get C-Gate installed on Windows is to install the C-Bus Toolkit software. C-Bus Toolkit requires C-Gate to communicate with C-Bus networks, and includes C-Gate in the installer. If you already have C-Bus Toolkit installed, you'll have C-Gate installed already.

C-Gate requires Sun's Java Runtime Environment 8 or later for operation. A recent stable version of Java is also included in the installer (of either C-Bus Toolkit or C-Gate).

Have a look

Look for the `\Clipsal\C-Gate2` directory on your default hard disk. If the directory is there, you already have C-Gate installed. Go ahead to Running C-Gate.

Installing C-Bus Toolkit and C-Gate

1. Download C-Bus Toolkit from the CIS website at http://www2.clipsal.com/cis/technical/downloads/software
2. Run the installer, and install C-Bus Toolkit using the standard defaults. This will also install C-Gate.

Installing C-Gate alone

If installing C-Gate independently of C-Bus Toolkit:

1. Download C-Gate from the CIS website at http://www2.clipsal.com/cis/technical/downloads/software
2. Run the installer, and install C-Gate using the standard defaults.

C-Gate as a Windows Service

Note that during C-Gate installation the Components page now has an option called "CGate as a Windows Service".

C-Gate as a Windows Service

If you choose to install C-Gate as a Windows Service, it will automatically start running after

installation. See Running C-Gate (Windows Service) for more information on starting and stopping the C-Gate Service.

C-Gate as a Console Application

If you choose NOT to install C-Gate as a Windows Service, it will install in the traditional manner, as a console application. After installation is complete, you will need to start C-Gate yourself. Proceed to Running C-Gate to learn more.

Note that the best way to switch between service and console mode is to re-run the C-Gate installer.

3.6 Running C-Gate (Windows Service)

Windows platforms:

When C-Gate is installed as a Windows service, it will be started automatically after installation, and also after the computer boots into Windows.

There are several ways to stop and start C-Gate:

1. You can stop and start C-Gate from the Service Properties dialog as above.

2. You can also control the C-Gate service from the Start Menu:

Start / All Programs / Clipsal / C-Gate / Start the C-Gate service

Start / All Programs / Clipsal / C-Gate / Stop the C-Gate service

3. You can send the SHUTDOWN command to instruct C-Gate to shutdown. When C-Gate is shut down, the service will automatically stop.

Note that when C-Gate is running as a service, you will not see the C-Gate console window. See Connect To C-Gate for how to issue commands to CGate.

3.7 Running C-Gate (Console Application)

(For Windows platforms)

If C-Gate is not installed as a Windows Service, or the service is stopped or uninstalled, then C-Gate needs to be run as a console application.

Some applications such as C-Bus Toolkit will automatically start C-Gate in console mode if it is not already running. If you are using these applications you should allow them to start C-Gate where possible.

In some situations you may want to start and stop C-Gate yourself.

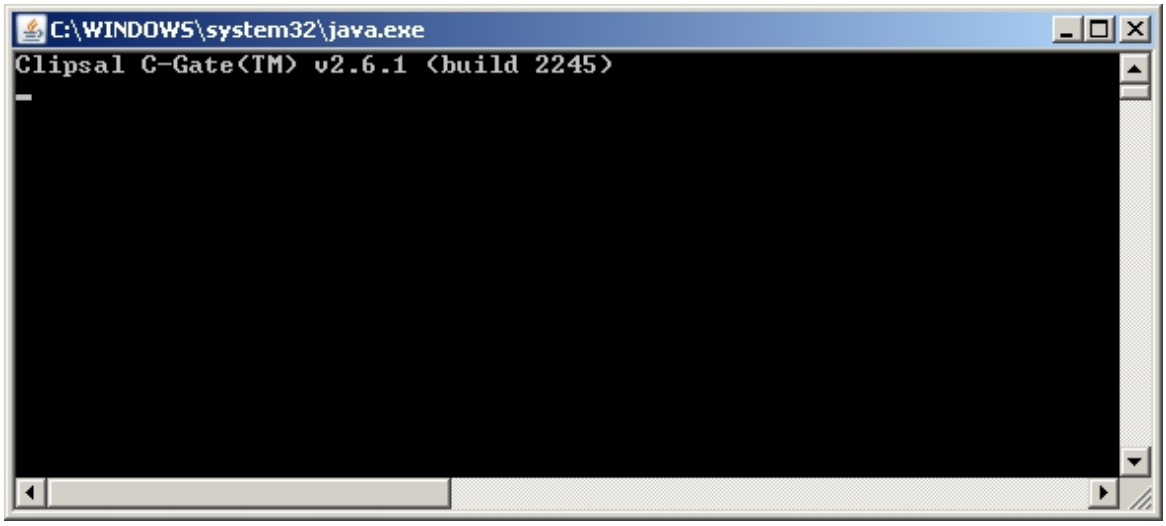
To run C-Gate, select the following from the Start Menu:

Start / All Programs / Clipsal / C-Gate / C-Gate

Alternatively, run the cgate.exe executable located within the C-Gate directory:

C:\Clipsal\C-Gate2\cgate.exe

This will open a Windows console window, running C-Gate inside of it. You can view the stream of events indicating what C-Gate is up to.
Here is an example of what you should see when you start the console version of C-Gate:



Note: we do not recommend running the cgate.jar Java binary directly. This is because cgate.exe takes care of things such as allocating the correct amount of memory for C-Gate.

To find out more about customising the launch configuration of cgate.exe, see Launch Configuration.

To stop C-Gate:

1. Use the SHUTDOWN command in the console window, or
2. Close the console window C-Gate is running in, or
3. Type Ctrl-C in the console window.

To use C-Gate:

You can enter commands directly into the console window, but it is easier to open a separate window for commands to avoid confusion.

So, let's Connect To C-Gate and then try out some commands.

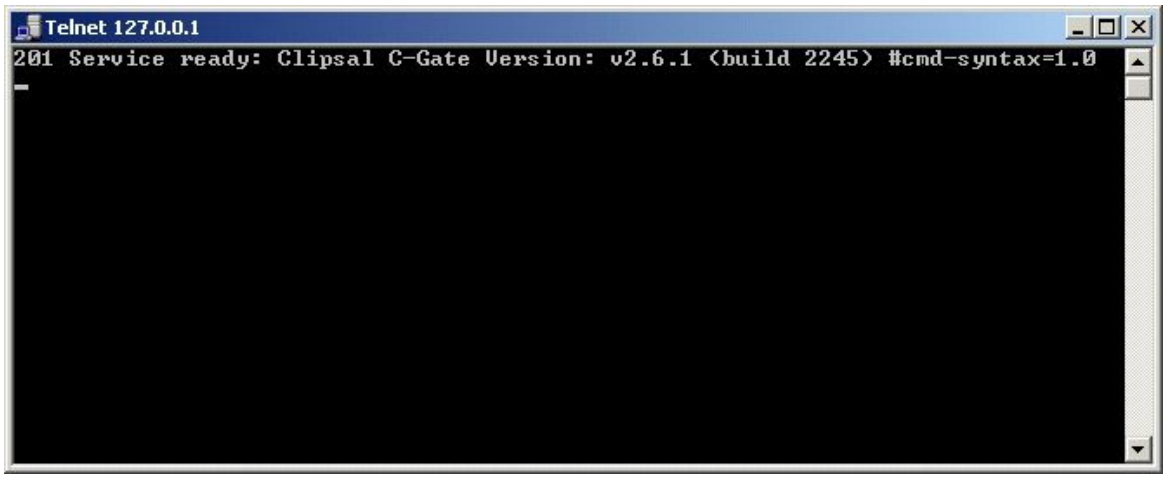
3.8 Connect To C-Gate

Windows platforms:

To connect to C-Gate, you are going to open a command-line session using the Telnet program that comes with Windows. The C-Gate installation makes it easy to make a connection from the start menu:

Start / All Programs / Clipsal / C-Gate / Connect to C-Gate

A new window will open similar to this one:



If the C-Gate server is not running, then the window will close almost immediately. Otherwise you are ready to start

The line:

```
201 Service ready: Clipsal C-Gate Version: v2.6.1 (build 2245)
```

shows C-Gate is ready to accept commands.

See Enter Commands to take the next step.

Telnet Client on Windows Vista, Windows 7 and Windows 8:

Since Windows Vista, the Windows Telnet client is turned off by default. You need to enable the telnet client feature in Windows before using the 'Connect to C-Gate' start menu item. To enable telnet client on Windows Vista, Windows 7 and Windows 8:

- Open Control Panel from the Windows Start menu.
- Select Programs.
- Under Programs and Features, click Turn Windows features on or off.
- From the list that appears, select the Telnet Client checkbox to enable telnet.

Other platforms:

You'll need to find and run the telnet client for your platform. In most cases, issuing a command like this will open a command session in the current window.

```
telnet localhost 20023
```

If this succeeds, you should see a welcome message like "`201 Service ready: Clipsal C-`

`Gate Version: v2.6.1 (build 2245)"` which indicates C-Gate is ready to receive commands.

See Enter Commands to take the next step.

3.9 Enter Commands

C-Gate is controlled by a series of commands typed on the C-Gate command line. Each command starts at the beginning of the line and is ended by pressing the Enter or Return key. C-Gate will send a response back for each command, indicating whether the command succeeded or failed and any other information requested in the command.

The rhythm of this is enter a command, wait for a response, enter a command, wait for a response, enter a command, wait for a response.

For example, the simplest command is noop, which just returns an 'OK' response:

type the word:

noop

then press Return or Enter

C-Gate will respond with:

```
200 OK.
```

In this quick start guide, commands we type are in **`bold`** while C-Gate's responses `are not`.

Try it yourself.

noop
`200 OK.`

Each C-Gate response starts with a three-digit number, in this case 200, followed by some text that gives you an idea what happened. The three digit number is there to make it easy for client programs connected to C-Gate to understand C-Gate responses. The text is there to make it easy for you to understand what is going on.

The noop command does nothing else but return an OK response. It is a good way of making sure that you are talking to C-Gate's command interface.

3.10 Working with Projects

When you configure or program a network with the C-Bus Toolkit software, it relies on C-Gate to create the project and perform the programming. Once the project has been created, you can load and use this project to control and monitor your C-Bus networks.

Available projects

Projects that are in C-Gate's memory are ready to be used immediately. To get a list of these projects, use the PROJECT LIST command.


```
project list
123 project=HOME state=stopped
```

As you can see, there is one loaded project, and it is stopped. We can use this project by entering a PROJECT USE command. This makes sure that this command session is working with the project we want.

```
project use HOME
200 OK.
```

Projects that are stored on disk can be loaded into C-Gate and then used.

Now, you can move on to Starting a Project or Network

Loading a project from disk into C-Gate

If your project doesn't show up in the results of the PROJECT LIST command, you can load it from disk.

To get a list of the projects on disk, use PROJECT DIR:

```
project dir
123-project=CIS_PROJ
123-project=CIS_TEST
123 project=HOME
```

The project we want, HOME , can be loaded with the PROJECT LOAD command:

```
project load HOME
20060206-144513 761 cmd1 - Command: project load HOME
```

and then set up this command session to use it:

```
project use HOME
200 OK.
```

Now, you can move on to Starting a Project or Network

3.11 Starting a Project or Network

To start C-Gate working with a C-Bus network, the network has to be opened. Opening a network means that C-Gate opens communication with the network and starts to build a model of the devices and operation of the network so you can use commands to manipulate the devices on the network. For example, to type in commands to turn on or off some relays controlling lights, the C-Bus network needs to be opened.

To open just one network use the NET OPEN command:

```
net open 254
```



```
120-initializing
120-opening port
120-starting network threads
120-pci reset
120-open complete
200 OK: //HOME/254
```

To open all the networks in your project, use the PROJECT START command.

```
project start
200 OK.
```

For more information and advanced usage see: Opening a Network.

Once opened the network will begin to sync which may take a minute or two. For more information on this see: Network Syncing.

Once the network is synced you can Explore the Network.

3.12 Explore the Network

Tree

Once a network has been opened, you can start to explore C-Gate's model of the network using some commands. The TREE command lists a lot of information about a network and the C-Bus Devices on the network.

```
tree 254
320- Network name:254 type=cni address=localhost:14000 state=ok
320-  Unit count=12
320- Units:
320-//HOME/254/p/0 ($0) type=PC_INT_1 app=255($ff),255($ff) state=ok
groups=
320-//HOME/254/p/1 ($1) type=DIMDN8 app=56($38),255($ff) state=ok
groups=13,14,15
320-//HOME/254/p/2 ($2) type=KEYM2 app=56($38),255($ff) state=ok
groups=13,14,35,36,37,38,39,40,5
320-//HOME/254/p/3 ($3) type=KEYM4 app=56($38),255($ff) state=ok
groups=13,14,15,38,39,40
320-//HOME/254/p/4 ($4) type=KEYM8 app=56($38),255($ff) state=ok
groups=13,2,3,4,5,6,7,8
320-//HOME/254/p/5 ($5) type=SCNCTL5 app=56($38),202($ca) state=ok
groups=
320-//HOME/254/p/6 ($6) type=RELDN12 app=56($38),255($ff) state=ok
groups=0,2,3,4,5,6,7,8,9,10,11,12,5,6
320-//HOME/254/p/9 ($9) type=PC_INT_1 app=255($ff),255($ff) state=ok
groups=
320-//HOME/254/p/14 ($e) type=KEYBL5 app=56($38),136($88) state=ok
groups=1,44,10,2
320-//HOME/254/p/15 ($f) type=PC_CTA app=255($ff),255($ff) state=ok
groups=
320-//HOME/254/p/19 ($13) type=PCLOCAL4 app=255($ff),255($ff) state=ok
groups=
320-//HOME/254/p/249 ($f9) type=GATEWLSN app=56($38),49($31) state=ok
```



```
groups=
320-  //HOME/254/56/10 ($a) level=0 state=ok units=6,14
320-  //HOME/254/56/11 ($b) level=0 state=ok units=6
320-  //HOME/254/56/12 ($c) level=0 state=ok units=6
320-  //HOME/254/56/13 ($d) level=255 state=ok units=1,2,3,4
320-  //HOME/254/56/14 ($e) level=255 state=ok units=1,2,3
320-  //HOME/254/56/15 ($f) level=0 state=ok units=1,3
320-  //HOME/254/56/35 ($23) level=255 state=ok units=2
320-  //HOME/254/56/36 ($24) level=255 state=ok units=2
320-  //HOME/254/56/37 ($25) level=255 state=ok units=2
320-  //HOME/254/56/38 ($26) level=255 state=ok units=2,3
320-  //HOME/254/56/39 ($27) level=255 state=ok units=2,3
320-  //HOME/254/56/40 ($28) level=255 state=ok units=2,3
320-  //HOME/254/56/44 ($2c) level=255 state=ok units=14
320- Application 202 ($ca) [trigger]
320- Net Vars:
320-  //HOME/254/202/255 ($ff) level=0 state=ok
320 -end-
```

The tree command is pretty much designed for human reading. Client applications that want this kind of information can use TREEXML to get a structured output that is much easier to parse.

So, what does the TREE output contain:

- The name, state and interface address of the network
- The number of C-Bus Units on the network
- A list of all the units on the network, with their Unit Type string, the applications they support, the unit state, and any Groups that are used.
- A list of applications in use on the network, with a listing of Groups or Network Variables, depending on the applications.

Applications, Groups and Network variables are used to control the C-Bus Devices. Find out how to use them in the next section on Operation.

3.13 Operation

Once a network is up and running, we can use commands to control some of the C-Bus devices on the network. For now, let's talk about controlling lighting devices as these are the most common you'll encounter on C-Bus.

Lighting is one of the Applications that C-Bus supports. Applications group a set of commands together for a broad purpose. Lighting Application commands are commands like ON, OFF and RAMP. The Lighting Application has an application number, which is 56.

In the lighting application, turning on and off and dimming lights occurs by changing the values of some network variables in the Lighting Application. Just in the lighting application, these network varibles are called Groups. There are 255 Groups, with addresses starting at 0 and going up to 254. And each group has a value between 0 and 255.

C-Bus Units are programmed using the C-Bus Toolkit software to associate Groups with keypresses on Switches, and control of Dimmers and Relays to turn light on or of as well as dim them. Normally a group is set to 0 to turn it off, and 255 to turn it on, with the values in between allowing dimming.

There are three main commands for manipluating group values, ON, OFF and RAMP.

ON sets a group value to 255
OFF sets a group value to 0
RAMP can set a value to any level between 0 and 255, with optional rate of change control.

So let's find some groups on our network. Remember the application number is 56. Let's have a look at some of the output from the TREE command again.

```
...
320-Applications:
320- Application 56 ($38) [lighting]
320- Groups:
320-  //HOME/254/56/0 ($0) level=0 state=ok units=6
320-  //HOME/254/56/1 ($1) level=255 state=ok units=14
320-  //HOME/254/56/2 ($2) level=0 state=ok units=4,6,14
320-  //HOME/254/56/3 ($3) level=0 state=ok units=4,6
320-  //HOME/254/56/4 ($4) level=0 state=ok units=4,6
320-  //HOME/254/56/5 ($5) level=0 state=ok units=2,4,6
320-  //HOME/254/56/6 ($6) level=0 state=ok units=4,6
320-  //HOME/254/56/7 ($7) level=0 state=ok units=4,6
320-  //HOME/254/56/8 ($8) level=228 state=ok units=4,6
320-  //HOME/254/56/9 ($9) level=0 state=ok units=6
...
```

You can see that groups 0 through 9 are defined for lighhting application 56 on our network 254 in our project HOME. Group 0 is at level 0, and group 1 is at level 255. So 0 is off, and 1 is on.

The `//HOME/254/56/0` gives the fully qualified address of group 0. We can refer to the group in commands as:
`//HOME/254/56/0`
or given we have previously run the PROJECT USE HOME command, we can also use:
`254/56/0`
because the project part is assumed from the PROJECT USE command.

So, let's turn on group 0:

```
on 254/56/0
200 OK: //HOME/254/56/0
```

Now, let's check the level has changed using the GET command.

```
get 254/56/0 level
300 //HOME/254/56/0: level=255
```

And let's turn that off again.

```
off 254/56/0
200 OK: //HOME/254/56/0
```

And now, let's set it to a mid level like, say, 100, but do it slowly:

```
ramp 254/56/0 100 2m
200 OK: //HOME/254/56/0
```


This will take approximately two minutes to reach level 100.

Whereas:

```
ramp 254/56/0 100
200 OK: //HOME/254/56/0
```

will cause the level change to take place immediately.

As you can see, this series of commands allow control of lighting devices from the command line. However, in addition to this control, C-Gate produces a lot of event information that allows client programs to subscribe to a range of events that occur on the C-Bus networks. Read on to find out about Events.

3.14 Events

C-Gate can provide a wide range of events to client programs to allow them to keep up-to-the-second information as to the state of the network.

Events are all delivered as line-by-line text, where the lastest information is presented as a single line indicating the event.

To gain an understanding of events, please read about Command and Monitoring Interfaces in the Reference section.

To learn about events which are logged on disk, refer to the Logging section.

3.15 Getting Help for Commands

C-Gate has a HELP command, and you can use that to get some help for commands that you want to work with.

Type help and the command name, and C-Gate will give you some help with the command if there is some available.

To find documentation for available commands, look in the Commands section of the C-Gate Reference.

3.16 Online Resources

Here are a few selected online resources to help with C-Bus and C-Gate.

C-Bus Forums
http://www.cbusforums.com

Several thousand people using C-Bus swap tips and ideas at the C-Bus Forums. Go here for general advice or esoteric tips and tricks, plus information about the latest software releases.

Clipsal Integrated Systems Website
http://www.clipsal.com/cis

Consult the official Clipsal Integrated Systems website for technical support and software

downloads.

4 Reference

This section is the reference guide to C-Gate.

4.1 C-Gate Launcher (Windows)

The Windows installer includes everything needed to get started out of the box. This includes:

- a launcher executable (cgate.exe), and
- a private Windows 32-bit Java Runtime Environment (/jre8 subdirectory).

The launcher executable serves multiple functions:

1. It can register C-Gate as a Windows Service that runs automatically as long as the computer is turned on.
2. It can run C-Gate in console mode.
3. It can find the included JRE and pass parameters to it as needed.

These operations are controlled with specific configurations.

4.1.1 C-Gate Launcher Configuration

The C-Gate Launcher uses a default configuration which works for most users. For certain scenarios a custom configuration may be needed.

The C-Gate Launcher takes its configuration from the following methods in order of priority:

1) Launcher parameters

cgate.exe accepts the following command-line parameters:

`-install`	Install as a Windows service
`-uninstall`	Uninstall the Windows service
`-start`	Start the Windows service if CGateService is not disabled
`-stop`	Stop the Windows service
`-console`	Start C-Gate as console application
`-restart`	Restart C-Gate, or start it if it was not previously running
`-silent`	Suppresses pop-up error dialogs (most of them). Can be used with any other parameter.

If no parameter is supplied the executable first attempts to start the Windows service and failing that launches C-Gate as a console application.

2) INI config file options

cgate.exe can use an optional cgate.ini file in the same directory which supports the following options:

`JAVA_DIR=<value>`	Use the Java binary at `<value>/bin/java.exe` to launch C-Gate. Useful when there are multiple versions of Java on the machine. If not specified, defaults to: `<C-Gate root directory>/jre`

3) Registry options

cgate.exe will look in the following registry location for an optional key that determines the full string

of parameters to be passed to the Java VM.

HKEY_LOCAL_MACHINE\SOFTWARE\Clipsal Integrated Systems\C-Gate\CurrentVersion\
String value: "JavaParameters"

Example value: "-Xms64M -Xmx256M"

4) Registry options set by C-Bus Toolkit

The following options are able to be changed using C-Bus Toolkit's Preferences dialog.

cgate.exe will look in the following registry location for an optional key that determines the Maximum memory amount to be allocated to the Java Virtual Machine.

HKEY_CURRENT_USER\SOFTWARE\Clipsal Integrated Systems\C-Gate\CurrentVersion\
String value: "JavaHeapMax"

Note: This option does not have any effect if the JavaParameters registry key is defined.

5) YAJSW Configuration

As the launcher utilises the open-source YAJSW wrapper it is possible for experienced users to customise the configuration files in the /yajsw directory. These configurations are made at the user's own risk and are not supported by Clipsal technical support.

4.1.2 New vs Old Launcher

C-Gate 2.11.0 introduced a new cgate.exe launcher to replace the old one.

It offers improved stability and logging, while retaining the same behaviours and command-line parameters as the old launcher.

There are some small differences:

- Memory options are now respected in Service mode. Whereas previously Service mode would always use 512MB, the maximum memory amount specified in the Toolkit preferences will now apply to the Service mode as well as to the Console mode.
- The -silent parameter no longer applies as the new launcher does not produce any popup dialogs.

The new launcher produces far more detailed logging, which it writes to two new log files on disk, see Launcher logs.

It's no longer necessary to run C-Gate in console mode to see the text of unhandled exceptions, and messages are no longer sent to the Windows Event Viewer.

4.1.3 C-Gate Launcher Diagnostics

As well as cgate.jar's own logging, the cgate.exe launcher writes to two log files:

- logs/launcher.txt – this captures logging specific to cgate.exe that was previously sent to Windows Event Viewer.

- logs/wrapper.txt – this captures the stdout/stderr output that was previously only visible in Console mode.

4.1.4 C-Gate As A Service

When C-Gate is installed as a Windows service, it will be started automatically after installation, and also after the computer boots into Windows.

You can confirm that it is working via the Services dialog in Windows. The easiest way to access this is as follows:

- From the Windows Start Menu, select Start / Run..
- Type 'services.msc' and press ENTER.
- The Services dialog will open.
- Find and double-click on the service called 'C-Gate Service' to open the Service Properties dialog.
- Confirm that the Startup Type is 'automatic'.
- Confirm that the service status is 'Started'.

4.2 C-Gate Jar

C-Gate at its core is a Java application (cgate.jar) and it can run on a range of platforms as long as there is an appropriate Java Runtime Environment.

For users on Windows platforms it is recommended to use the C-Gate Launcher as it provides a greatly simplified C-Gate experience.

For users on other platforms it is recommended to use the C-Gate zip distribution and you will need to supply the appropriate JRE and a launch mechanism.

4.2.1 Zip Distribution

The zip distribution of C-Gate is recommended for non-Windows users.

It is functionally identical to the installation by the C-Bus Toolkit installer, with the following differences:

1. A private JRE is not included. It is up to you to provide a Java Runtime Environment on the system.
2. A simple cgate.sh launcher is included to launch from a Linux shell.
3. C-Gate is not installed as a service or daemon. It is up to you to configure C-Gate to launch as and when desired.
4. Windows USB drivers are not included. The first-time you install on a Windows system you should use the C-Bus Toolkit installer.
5. Built-in support for serial ports on Java 32-bit for Windows, Linux and Mac OSX. Note that you can install a 32-bit Java on a 64-bit operating system.
6. Built-in support for serial ports on Java 64-bit for Linux.
7. Configurable support for serial ports on Java for Windows 64-bit and Java for MacOS X 64-bit by moving and renaming files (see \lib\serial-readme.txt).

Installation is straight-forward: unzip to the directory of your choice.

WARNING: Please don't contact Technical Support for problems with the zip distribution. You should have moderate familiarity with your target OS, with platform scripting and with executing Java applications.

4.2.2 Java Runtime Environment

C-Gate is compiled with the following version of Java:

Oracle Java 8 JDK for Windows

A copy of the equivalent JRE is included in the Windows installer and the C-Gate Launcher will automatically use it. The C-Gate zip distribution does not include a JRE.

It is possible to use your own JRE. For best results use one that is the same major version and which is also compiled for 32-bit architectures.

Bear in mind that some libraries included with C-Gate may not work if the JRE:

- is a higher major version (e.g. JRE 9),
- uses a different architecture (eg. 64-bit, ARM),
- or is for a different platform (e.g. Linux, Mac).

So far the known limitations include:

1. The SerialIO library needs manual copying of files to work with 64-bit JREs.
2. The SerialIO library does not work on Linux ARM.

There may be other limitations not yet documented.

4.2.3 Manual Launch

JRE Location

First you should confirm the availability and location of your JRE. For example you can check to see if it is on your path:

```
$ java -version
java version "1.8.0_91"
Java(TM) SE Runtime Environment (build 1.8.0_91-b15)
Java HotSpot(TM) 64-Bit Server VM (build 25.91-b15, mixed mode)
```

JRE Parameters

In order to reproduce the configuration used by C-Bus Toolkit and the C-Gate Launcher you should pass the following parameters to your JRE:

```
-Xms64M                 # don't need to change this.
-Xmx256M                # increase to 512 etc as needed.
```

Not doing this may cause C-Gate to run out of memory when working with large project files.

Manual Launch

Putting all the above together, to launch C-Gate in a standard shell:

```
$ cd ~/cgate2
$ java -jar cgate.jar -Xms64M -Xmx256M
```

This will start C-Gate in the current shell.

C-Gate will display a message like this:

```
Clipsal C-Gate(TM) v2.6.1 (build 2245)
```

You can enter commands directly into the console session but it is easier to open a separate window for commands to avoid confusion.

For more information see the Command and Monitoring Interfaces.

Automatic Launch

It is up to the user to configure daemons or scheduled tasks for their operating system to automatically launch C-Gate.

4.3 Command and Monitoring Interfaces

Once C-Gate is running, it makes available a number of interfaces to allow command to be issued to the C-Gate server and to allow monitoring and event information to be received from C-Gate. These interfaces are available as telnet ports, with optional SSL encryption with client authorisation. Access control also applies to these connections. In addition, the console can be used for the delivery of event and issuing commands.

For all of these interfaces, multiple clients can connect at once, allowing multiple operations and control clinets to talk to C-Gate and control and monitor connected C-Bus Networks.

All of these interfaces are telnet-style TCP/IP sockets, which can be accessed directly using a telnet client program to allow for manual use. The SSL interfaces require a SSL telnet client or similar.

The interfaces are:

Command Interface

The command interface is used to issue commands and get responses back from C-Gate. This is the prime method used by client applications that wish to interface to C-Gate. There is a rich command set available for this interface allowing all full configuration and operation of C-Gate from a remote location.

The command interface is normally found at TCP/IP port 20023, or port 20123 for a secured SSL connection.

Note: using the EVENT command it is possible to flexibly enable the delivery of all sorts of events on the Command Interface.

Event Interface

The event interface is used to provide a continuous stream of events as they occur on the server. Clients can listen to the event interface and receive the stream of events as they occur. The events have date, time and object information attached to them, and the list of events is well defined.

The event interface is normally found at TCP/IP port 20024, or port 20124 for a secured SSL connection.

Status Change Port (SCP)

The Status Change Port allows listening clients to recieve an ongoing series of events that result from status changes in C-Bus Applications. Status changes include lighting application on, off and ramp messages along with messages for other C-Bus applications.

The output of this port is rendered as a series of command interface commands and comments, which allows their recording for client macro facilities.

The Status Change Port is normally found at TCP/IP port 20025, or port 20125 for a secured SSL connection.

Config Change Port (CCP)

The Config Change Port allows listening clients to see any configuration change in the C-Bus networks connected to C-Gate. These configuration changes include addition or deletion of C-Bus Units and unit synchronisation states.

The Config Change Port is normally found at TCP/IP port 20026, or port 20126 for a secured SSL connection.

4.3.1 Command Interface

Commands can be sent to C-Gate and responses and events received via the console (startup window) or via a TCP/IP interface, with or without SSL encryption.

The command interface supports a wide range of commands used to configure, control and monitor C-Bus networks and the C-Bus server itself.

4.3.1.1 Console Connection Details

When the C-Gate server starts. a console window is normally opened. This console window can be used to view events and submit commands to the C-Gate server.

The ability to type commands can be controlled by the console.enable-commands configuration option. It is enabled by default.

The console acts as a special command session. It is started without a `201 Service Ready`

message and is started with event output already turned on. The console starts up in event mode e +s0c0. The event mode can be changed by the EVENT command.

The console starts up with access control level `Program`. This can not be changed.

4.3.1.2 TCP/IP Connection Details

The command interface for the C-Gate server is accessed as a TCP/IP socket interface using a series of text commands and responses to allow both human and automated use.

To access the C-Gate server to type commands and view status:

1. Start the telnet program.
2. Connect to the hostname or IP address of the C-Gate server, using port 20023, or the port number set in the cgate system parameter command-port.
3. The C-Gate server will respond with the 201 Service Ready response if the connection is established successfully.

Automatic systems wishing to access the C-Gate server should open a TCP/IP socket to the IP address of the C-Gate server; port 20023. The 201 Service Ready message should be received in response to the connection.

8 bit bytes are supported over the socket interface. All command and response data is 8 bit USASCII characters. The high-order bit is ignored in commands.

Note: if the C-Gate server's cgate parameter accept-connections-from is set to a list of IP addresses, then connections will only be accepted from hosts with those IP addresses. See section 0 for further details of this parameter.

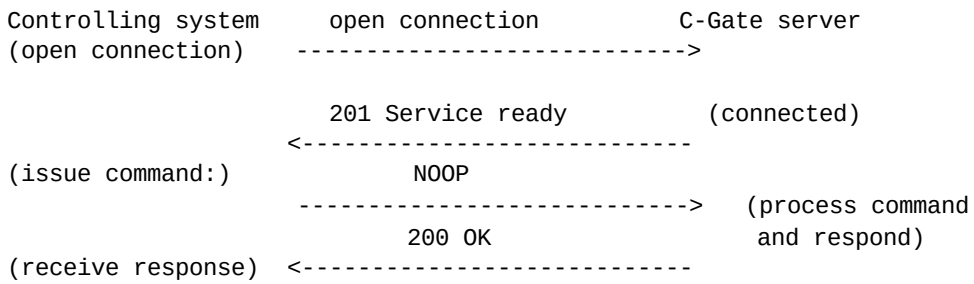
4.3.1.3 Commands and Replies

The command interface is based on the issuing of a command and a response following the execution of the command by the C-Gate server. All commands are one line only, terminated by a CRLF pair. Each response is one line long.

The figure shows the opening of a connection to the C-Gate server and the ongoing conversation. Once the controlling system opens the connection, the C-Gate server responds with a 201 Service Ready message. This response is to the opening of the connection and shows the controlling system that the IG is ready to receive commands.

When the controlling system issues the NOOP command, CGate responds with a 200 OK message, a single response message for this command.

When the controlling system sends a GET command, a reply is sent.




```
(issue command:)         GET 2/56/1 level
                     ---------------------------->   (process command
                        300 2/56/1: level=102         and respond)
(receive response)  <----------------------------

(issue next command:)       …
                     ---------------------------->
```

A command is a line of ASCII text terminated by a CRLF pair. Each command line begins with an action verb and contains one or more parameters following the action verb. White space (ASCII space and tab characters) are used to separate commands and parameters.

Command syntax is described here, and details of individual commands are described in the Commands section.

```
command = action-verb [ parameters ] CRLF

action-verb = name ; See the list of commands below.
```

4.3.1.4 Commands

This section describes the commands supported by the C-Gate Command Interface.

4.3.1.4.1 Terminology

Augmented BNF

The descriptions of C-Gate commands uses an augmented Backus-Naur Form (ABNF) to describe the grammar of commands, responses and events. Reference should be made to the IETF's RFC2234 (http://www.ietf.org/rfc/rfc2234.txt) which fully defines augmented ABNF as used in this specification.

This additional definition is used widely in the ABNF used in this document:

```
NAME = ALPHA [*<CHAR except CTRL, whitespace, "$", "/", "~", "-", ",", ":",  or
"."">]
```

4.3.1.4.2 Objects and Addresses

Object Approach in C-Gate

The C-Gate server adopts an object-oriented approach to controlled devices. All controlled devices and many internal components of the C-Gate server are exposed to the command and event interfaces as objects that can be accessed with commands.
In general, every object has:

- An object identifier that allows the object to be referred to. Object identifiers may be network addresses, symbolic names, or reserved system names.
- Parameters that may bet set or viewed, using the SET and GET commands
- Methods which may be executed with the DO command

Objects are either created by the system on startup, created as a C-Bus network is scanned, or are created with the NEW command.

System objects provide parameters and methods to allow configuration and operation of the C-Gate server itself. Key system objects include the `cgate` object and defined C-Group objects. Parameters for these objects are described in

Addressing Controlled Components and Objects

Both commands and events reference the components and objects in networks attached to the C-Gate and contained within it. This section defines the format of addresses for network devices used in the gateway and for object identifiers, that can reference addressed components on networks, but can also reference local system objects.

Addresses

Addresses are constructed from one or more address parts separated by the "/" (slash) character. Address parts get more specific to the right hand side of an address. Address parts except for network names, are all expressed as positive decimal integers, in the range from 0 to 32767 or as hexadecimal integers proceeded by the '$' character.

Object Identifiers

An object identifier can contain:

- An address, as described above, or
- an object name that is known to the C-Gate and is part of a flat namespace typically used for system objects and CGroups.

C-Bus address components

The table shows the components of a C-Bus address that would be used to control and monitor C-Bus networks connected to the C-Gate.

Address type	Description
Project Name	The name of a project. Generally project names shall not start with a number, and will be 8 or less uppercase characters. E.g. `HOME` or `CIS`.
C-Bus Network name	This is a name or number describing a C-Bus network controlled by the C-Gate server. Important notes: Network names are case-sensitive. Network A42 and network a42 are different networks. When using C-Gate with Clipsal C-Lution, you must use numbers for the C-Bus network name, and the numbers must be in the range 1 through 255. Network name 0 is reserved for system use.
C-Bus Unit number	Identifies a single physical C-Bus unit on a C-Bus network. Unit numbers are unique on a C-Bus network. C-Bus unit numbers range from 0 to 255. Unit 255 is the initialisation address for new C-Bus units that have not been allocated a device address, but otherwise operates as a normal unit address.
C-Bus terminal number	This is the number of a unique relay terminal, dimmer circuit, or key switch pushbutton on a C-Bus unit. C-Bus relays and dimmers typically have from

	1 to 12 terminals or circuits and this part of an address allows them to be individually addressed. There is no terminal 0 (zero) on any C-Bus unit.
C-Bus Application Number	This is the number of a C-Bus application. There are 255 possible applications, ranging in number from 0 through 254. Application number 255 indicates the unassigned application in a C-Bus unit.
C-Bus Group or Area number	C-Bus Group and area numbers are used in sending commands and monitoring commands sent to the C-Bus network. In the programming of a C-Bus network, C-Bus Units are programmed to send or act upon commands that specify a Group or an Area for a specific application. C-Bus Group and Area numbers range from 0 to 255 in any C-Bus network. Groups share the numbering space with Area numbers.

Representing Addresses

The augmented BNF forms below show how addresses are represented. There are two main types of addressing:

- physical addressing by network components or physical devices, and
- group and application addressing which addresses the network by C-Bus applications and C-Bus groups (used in C-Bus for command and status information).
-

Physical addresses are indicated by the addition of the "p/" path element to the beginning of the address.
A wildcard (match all objects) address can be indicated by using a "*" symbol in the last part of an address.

```
numeric-address = 1*DIGIT | "$" 1*HEX_DIGIT | "*"

terminal-number = numeric-address
unit-number = numeric-address
network-name = 1*DIGIT  | NAME
application-number = numeric-address | "~"
group-number = numeric-address
project-name = 8*CHAR             ;; should be uppercase
project-prefix = "//" + project-name + "/"
physical-address = [project-prefix | "/"] "p" "/" network-name "/" unit-number
[ "/" terminal-number ]

application-address = [project-prefix | "/"] network-name (("/" application-number
) | "//")

group-address = application-address "/" group-number

network-address = [project-prefix | "/"] network-name

project-address = "//" + project-name

system-address = NAME
```

Representing Object Identifiers

The augmented BNF below shows the form of an object identifier. Note that an object identifier full encompasses the representation of an address.

```
object-identifier = NAME | network-address | physical-address | application-
address | group-address | project-address
```

Addressing Examples

Some example addresses are shown below:

To address network 254 in project HOME, the address would be:

```
254
```

or

```
//HOME/254
```

To address physical C-Bus unit 15 on network 57 the address would be constructed as:

```
p/57/15
```

or

```
//HOME/p/57/15
```

To address all the units on network 57 the address would be constructed as:

```
p/57/*
```

```
To address terminal 2 on the same unit:
```

```
p/57/15/2
```

To address C-Bus group 12 on C-Bus network 2 on application 56, the address would be:

```
2/56/12
```

To address group 5 in the default application of network 11

```
11//5
```

or

```
11/~/5
```

4.3.1.4.3 Array Filtering

Array Filtering is a syntax permitted in certain C-Gate commands that allows you to reference an object in an Array not by its unique index but by filtering on a field of the object.

Where you would normally use the array index in brackets e.g. **[3]** you would use a **[name=value]** syntax.

This is particularly useful with the database commands such as DBGET.

Examples

This is the conventional method to reference an array object. The index [3] has no meaning to the user other than as an unique index, so it is not easy to find the object you want.

```
dbget 254/Unit[3]
342-254/Unit[3]/OID=b31a30f0-0736-1033-a235-9e41e9666ddf
342-254/Unit[3]/TagName=NEWUNIT
342-254/Unit[3]/Address=207
342-254/Unit[3]/Description=
```


...

Instead we can use an array filter to locate a Unit object by its C-Bus address:

```
dbget 254/Unit[Address=207]
342-254/Unit[3]/OID=b31a30f0-0736-1033-a235-9e41e9666ddf
342-254/Unit[3]/TagName=NEWUNIT
342-254/Unit[3]/Address=207
342-254/Unit[3]/Description=
...
```

We can add a second array filter to retrieve a known field from the PP array:

```
dbget 254/Unit[Address=207]/PP[Name=ClockGenEnable]
342-254/Unit[3]/PP[9]/Name=ClockGenEnable
342 254/Unit[3]/PP[9]/Value=1
```

Depending on the unit type the PP fields are found at different indexes or may not even exist. We can query all the units at once:

```
dbget 254/Unit/PP[Name=ClockGenEnable]
342-254/Unit[1]/PP[14]/Name=ClockGenEnable
342-254/Unit[1]/PP[14]/Value=1
342-254/Unit[2]/PP[9]/Name=ClockGenEnable
342-254/Unit[2]/PP[9]/Value=0
342-254/Unit[3]/PP[9]/Name=ClockGenEnable
342 254/Unit[3]/PP[9]/Value=1
```

A further refinement to just retrieve the Values:

```
dbget 254/Unit/PP[Name=ClockGenEnable]/Value
342-254/Unit[1]/PP[14]/Value=1
342-254/Unit[2]/PP[9]/Value=0
342 254/Unit[3]/PP[9]/Value=1
```

Filtering by blank fields:

```
dbget 254/Unit/Description
342-254/Unit[1]/Description=woh
342-254/Unit[2]/Description=
342 254/Unit[3]/Description=

dbget 254/Unit[Description=]/Description
342-254/Unit[2]/Description=
342 254/Unit[3]/Description=
```

Known Limitations

1) You can't use a space in your filter value:

```
# 8 characters long inc space at end
dbget 254/Unit/PP[Value=NEWUNIT ]
400 Syntax Error: Too many parameters
```



```
# array of integers
dbget 254/Unit/PP[Value=0x0 0x0]
400 Syntax Error: Too many parameters
```

2) Even if you are filtering on what you believe to be a unique identifer there is no guarantee that you'll only get one object in response. You should validate the contents of the response including the index or indexes.

History

- C-Gate v2.11.0 implemented the Array Filtering feature.

4.3.1.4.4 Unique Command IDs

Each command to the C-Gate server can have a unique ID attached to it. When an id is added to a command, all the responses for that command are prefixed with the id given for the command. This makes it easier for client programs to connect commands and responses.

Commands can also be set to run in the background, allowing the command to run in a separate thread and complete while other commands are executed.

Verbose output can be enabled as well, resulting in additional responses from some commands.

4.3.1.4.4.1 Adding IDs to commands

An ID prefix can be added to any command, and then all responses to the command are prefixed with the same prefix.

The ID prefix has the following syntax:

```
['*'] [ ['&' [ priority-digit ] ] '[' id-string ']' ] command
```

where:

`command` is any existing C-Gate command, including any required parameters.

`priority-digit` is a digit in the range 0 through 9 giving the thread priority of this command, and 4 is the regular (normal) priority.

`id-string` is a string provided by the client program connecting to C-Gate. This ID string should conform to the following rules:

1. can not contain whitespace characters, control characters, or NUL
2. can not contain left or right square brackets ('[' or ']')
3. can not contain left and right angle brackets ('<' or '>')

In addition, it is useful to make the *id-string unique for each command sent*. If this is done, responses can be related to command easily. Otherwise, there is little point in adding IDs to commands.

4.3.1.4.4.2 Background processing

If it is desired that the command be processed in the background so that another command can be entered immediately, adding a '&' to the front of the command line, and an optional single digit priority will run the given command in a separate thread allowing another command to be entered immediately.

The standard priority for the new thread is `4`. A priority of `0` is the lowest possible, while `9` is the highest possible.

USE PRIORITIES WITH CARE, and note that they may behave very differently on different platforms.

DO NOT habitually use really high or really low priorities.

4.3.1.4.4.3 Verbose output

Some commands may provide additional verbose output. This may be useful for debugging purposes. Verbose output can be turned on by making the first character of the command line a '*' (star).

4.3.1.4.4.4 Responses

When an id string is added to a C-Gate server command, then all responses will have the following syntax:

```
'[' id-string ']' response
```

where `id-string` is the id-string given in the command that this response is from, and `response` is a standard C-Gate response, including a response code and a description.
Note: even when XML snippets are being delivered as responses (without leading response code values) there will still be an [id] added to the commencement of each line of XML.

4.3.1.4.4.5 Example of Command ID

The following shows an example session using these ID numbers:

```
[1] noop
[1] 200 OK.
&1[bgsync] do 1 sync
&[2] get cgate networks
[ab1134] tree 1
[2] 300 cgate: version=v1.6.0 pre-alpha (build 1604)
[ab1134] 320- Network name:1 type=Cni address=10.1.1.25:10001 state=new
[bgsync] 202 Done.
[ab1134] 320-  Unit count=0
[ab1134] 320- Units:
[ab1134] 320-Applications:
[ab1134] 320 -end-

# example of stopping a command
[33] net sync myNet
stop 33
[33] 207 Stopped.
```

4.3.1.5 C-Gate Response Codes

The C-Gate Response Codes may be broken into several separate categories. These are described in the following sections.

Every command results in a response. Responses can be one or more lines of text. Each response line is terminated with a CRLF pair. If event output is enabled using the EVENT command, then responses can include events that are not in response to commands.

The BNF form of all valid responses responses are shown below. Note that responses to commands can come from backgrounded commands with id strings, and events can be included with or without

prefix codes.

```
 response = ( command-response|event-response|buffer-overflow-indicator ) CRLF

 command-response = [command-id] command-data
 command-id = '[' id-string ']'
 command-data = response-code ( continuation-mark | " ") text CRLF
 continuation-mark = "-" ; this indicates another line follows
 response-code = 3*DIGIT;

 event-response = [event-id] event-data
 event-id = '#' e|s|c '#' SP
 event-data = event-time event-code object-identifier event-info
 ;; event-id is not given when the event mode e+ is used.

 buffer-overflow-indicator = "###!!!Event buffer overflow. Events have been
missed.!!!###"
```

The digits in the response code are coded to make dealing with the response easy. The table explains the general use of the three digit response codes. Note that the 7xx, 8xx and 9xx responses are used to define events in the event interface.

First DIGIT of response-code	General meaning	Use
0xx	Not used	Not used.
1yz Informational	Informational response only, to indicate useful information	Generally for human users, help and suggestions. Generally ignored by automatic systems.
2yz Successful	Operation successful and complete	Usual in response to successful SET commands.
3yz Object Status	Returns status of objects or other requested information. May be one or more response lines of object information which need to be decoded.	Generally 3xx codes are issued in response to a SHOW command
4yz Client Error	Incorrect or malformed command, address, object id or other client-side problem has caused the problem to fail	Example: 400 Syntax Error
5yz Server Error	Failure of the command due to some failure in the connected controlled devices or in the IG itself.	
6yz Continuation	Issued when a following command is required to complete the specified	For example, a restart command requires another confirming command. 6xx is issued to ask for

	operation	confirmation before performing a restart.
7yz	Reserved for events	
8yz	Reserved for events	
9yz	Reserved for events	

The second and third digits give increased granularity to the error codes. Refer to the full response code table for details of these codes.

4.3.1.5.1 General Response Codes

Code	Meaning	Syntax of Response
10x	Help Information	
101	Help information	(typically multiple lines such as) 101-help-text 101-help-text 101-help-text
110	Starting macro file	"Starting macro file:" filename
111	Ending macro file	"Ending macro file:" filename
112	Indicating a macro command from an executing macro file	"Macro command:" command-line
120	Additional information from executing method or command	method-name":" information
121	Names of parameter programming session	"121 name=" name "address=" address
122	Indicates that no parameter programming session exists	"no open sessions"
123	List of projects (from PROJECT DIR command)	"123 project=" project-name
124	Indicates that no projects exist in directory	"124 no projects found"
125	Port info from PORT LIST command	"125 port=" port-name "status=" status
126	No ports found from PORT LIST command	"126 no ports found"
127	List of interfaces from PORT IFLIST command	"127 address=" ip-address "interface=" interface-name
128	No interfaces found from PORT IFLIST command	"128 no interfaces found"
129	List of CNIs from PORT CNISCAN command	"129 found CNI ip-address=" ip-address "status=" status "port=" port-number
130	No CNIs found from PORT CNISCAN command	"130 no CNIs found"
131	List of defined networks from NET LIST command	"131 network=" network-address "state=" status "InterfaceState=" interface-state
132	No network found from NET LIST or NET LIST_ALL command	"132 no networks found"
133	List of catalog numbers and	"133 catalogNumber=" "description=" description

	descriptions	
134	Calculator results	"134-result:" "FAILED" | "OK" "134-current_supply (mA)=" value "134-current_consumption (mA)=" value "134-impedance (ohms)=" value "134-units_calculated=" unit-count "134-units_not_calculated=" unit-count
135	Results from NET LIST_ALL command	"135 project=" project-name "address=" net-address "OID=" net-oid "interfaceType=" interface-type "interfaceAddress=" interface-address "state=" object-state "interfaceState=" interface-state
136	Results from DBNETWORKPATH command, compact form	"136" hex-bytes
137	Results from DBNETWORKPATH command, OID form	"137" oids
138	api version results from APIVER command	"138" api=version

4.3.1.5.2 Pre-Release Debugging Information

Code	Meaning	Syntax of response
19x	Pre-release debugging information	
191	Pre-release debugging information	Unstructured information (not for automated use)

4.3.1.5.3 Successful Completion Messages

Code	Meaning	Syntax of response
20x	Successful completion	
200	Completed successfully	"OK"
201	Service ready	"C-Gate-name=" NAME "Service ready on "C-Gate-name"
202	Method sucessfully executed	"Done:" object-id
203	Macro complete	"Macro:" filename "complete."
204	Closing connection	"Closing connection"
205	Restart confirmed	"Restart confirmed."
206	Shutdown confirmed	"Shutdown confirmed."
207	Command Stopped by STOP command	"Stopped"
208	Queued for sending via a backgrounded command	"Queued for sending"

4.3.1.5.4 Security Messages

Code	Meaning	Syntax of response
2xy	Security	
210	Indicating the current access level	"Access level" access-level-name
211	Indicating the new access level	"Access level set to" access-level-name
225	Indicating a successful lock operation	object-identifier ":" "Locked."

226	Indicating a successful unlock operation	object identifier ":" "Unlocked."
230	Network detected from PORT PROBE command	"Probe succeeded: C-Bus network detected"
231	Indication of network time	"231 Time set to:" hh:mm:ss ds hh = 00..23 hours mm= 00..59 minutes ss= 00..59 seconds ds= 1 | 0 for daylight savings time on or off respectively
232	Indication of network date	"Date set to:" yyyy-mm-dd dow yyyy= Year as four digits (eg 2003) mm=01..12 months dd=01..31 days dow= 0..6 day of week (eg 0=Monday, 3= Thursday etc)

4.3.1.5.5 Object Information Messages

Code	Meaning	Syntax of response
30x	Object information	
300	Object information	parameter-name = token parameter-value = token | *DIGIT | quoted-string one or more lines of: object-identifier ":" *(parameter-name "=" parameter-value) For example: 300 1/56/1: level=200
301	Object ID information (results from a DBADD command where an OID is created)	"OID=" oid-value for example: `301 OID=df7c9feb-5919-11d7-a52d-004854d15455`
302	Unit list from network (from NET PINGU command)	"302" "Units=" unit-list for example: `302 Units= 1,2,3,4,255`
303	Type Version Serial info from NET SYNCNEW command	"303" "Unit found: address=" unit-address "stype=" type "version=" version "serial=" serial-number
304	Directory information from FILE DIR command	"304" "directory=" dir-name "files=" file-count
305	File information from FILE DIR command	"305" "name=" filename "size=" size "modified=" modified-dates
306	Event mode information from EVENT command	"306" event-mode event-mode = "e" (DIGIT | '+') "s" DIGIT "c" DIGIT
315	Parameter programming , parameter information returned from a PP command.	parameter-name = token parameter-value = token | *DIGIT | quoted-string parameter-name "=" parameter-value
316	Return of raw data	"RawData=" hex-byte-pairs (note that bytes with no valid values are replaced with ??
320	Tree information. Returned from the tree command.	The tree command returns a complex set of lines designed for human consumption.
321	Network serial number from topology explore	"321" "Network Serial"net-name serial-number
322	Alternative path found in	"322" "Alternative Path Found" net-name port-path

	Topology	
323	Network found in topology scan	"323" "Network Found" net-name port-path
324	Bridge found in topology scan	"324" "Bridges Found " name number-of-bridges
325	Tunnelling Found in topology explore	"Tunnelling Found" Network to Network "ALL" or "app, app"
326	Application connect found in topology explore	"Application Connect Found" net type "ALL" or "app, app"
327	Patch information line	"327" "Version=" patch-version "Name=" patch-file-name "Special=" ("true" | "false") "Description=" quote description quote quote = ""
343	Indicates the start of an XML snippet	"Begin XML Snippet"
344	Indicates the end of an XML snippet	"End XML Snippet"
345	Indicates start of a file download	"Start file download for file: " filename
346	Indicates end of a file download	"End file download"
347	An XML data line	XML data

4.3.1.5.6 Client Side Command Errors

Code	Meaning	Syntax of response
40x	Client side command error	
400	Syntax error	"Syntax error."
401	Bad object or device ID	bad-object-id "=" object-identifier or "Bad object address" or"Field not found" or various others
402	This object can't do the requested operation (see the C-Gate Command Interface Guide to see what is supported by each object	"Operation not supported by:" object-identifier
403	Unsupported ramp level	"Unsupported Ramp Level:" ramp-level
404	Unsupported ramp time	"Unsupported Ramp Time:" ramp-time
405	Parameter out of range	bad-parameter "=" parameter-name "Parameter out of range: bad-parameter
406	Error creating object	"NEW command error: details
407	Error processing scene. A scene could not be played or recorded due to an error.	"Error processing scene."
408	Indicates that a SET, GET or other method failed for a given object	"Operation failed:" details or "OID field can not be changed"

4.3.1.5.7 Macro Errors

Code	Meaning	Syntax of response
41x	Macro errors	
410	Macro command error	"Macro command error at:" filename line-number line-number = 1*DIGIT
411	Enable command not allowed in macro	"Enable not allowed in macro at:" filename line-number
412	Macro loop detected	"Macro loop detected at:" filename line-number

413	Macro file not found	"Macro file not found:" filename

4.3.1.5.8 Access Control Errors

Code	Meaning	Syntax of response
42x	Access control errors	
420	Access denied	"Access denied."
421	A command connection was refused.	"Connection refused."
422	Bad username and password in a login command	"Username and Password do not match."
425	Can't perform a lock command on something that is already locked	"Already locked by:" session-name
426	Can't perform an unlock command	"Unlock failed."
427	Can't perform a cancel lock command	"Cancel lock failed."

4.3.1.5.9 Method Failures

Code	Meaning	Syntax of response
4xy	Method failures	
430	Parameter sync failed	"Parameter sync failed."
431	PORT PROBE failed because port is in use	"431 Probe failed: Port in use."
432	PORT PROBE failed because there is no network	"432 Probe failed: No -Bus network detected."
442	Antimatter injectors failed to shut down	"442 Core breach imminent"
446	Unable to set a DB entry	"Unable to set: " error-message
444	XML snippet could not be produced	"XML creation failed:" reason
460	Programming parameter name not found	"Parameter not found."
461	Array index out of bounds	"Array index out of bounds"
462	Parameter value is out of range	"Parameter value out of range"
463	A bad parameter value was supplied	"Bad parameter value"
464	Project already loaded	"Project already loaded."
465	The project could not be found	"Project not found."
466	Can't save the project	"Can not save the project."
467	Tried to close a project without shutting it down first.	"Can not close running project."
468	A network can't be deleted while it is still open	"Can not delete open network"
469	Can't create or load a network	"Can not create network:" reason
470	Bad interface specification	"Bad interface specification for " net-name " interface-spec
471	Cannot create network	"Can not create network" net-name interface-spec

472	Cannot open network	"Can not open network" net-name

4.3.1.5.10 Internal Errors

Code	Meaning	Syntax of response
50x	Internal errors	
500	Internal error of some kind	"Internal Error." debug-information debug-information = text describing error

4.3.1.5.11 Network Errors

Code	Meaning	Syntax of response
52x	Network errors	
520	Network unavailable	"Network" object-identifier "unavailable"
521	Unit unavailable	"Unit" object-identifier "unavailable"
522	Unit error	"Unit error:" unit-error-information unit-error-information = text describing what the error is
523	Interface error	"Interface error on:" interface-name interface-name = NAME
524	Error sending command to network	"Command send error"

4.3.1.5.12 Continuation Confirmation

Code	Meaning	Syntax of response
60x	Continuation confirmation	
600	Confirm that you want to proceed by typing the word "confirm"	"Critical operation. Type CONFIRM to continue."

4.3.2 Event Interface

Network Connection

The event interface operates on a similar basis to the way responses are provided on the command interface. However, the event interface uses a separate TCP/IP connection to provide event information.

On startup of the C-Gate server, the C-Gate server will attempt to open a TCP/IP connection to the host IP address and port number specified in the sys-event object parameters. Once this connection is established, the C-Gate server will send events on a line-by-line basis, one event per line. The level and type of events that are delivered is controlled by the parameters of the sys-event object. Parameters to this can be shown and set via the command interface and stored in the system configuration file for startup.

Structure of Individual Events

The structure of an event line is shown in the augmented BNF form below:

```
event-response = event-time event-code object-identifier event-info
```



```
event-time = YYYYMMDD-HHMMSS ["."mmm]
; YYYY = 4 digit year code
; MM = 2 digit month code (01-12)
; DD = 2 digit day code (01-31)
; HH = 2 digit hour (00-23)
; MM = 2 digit minute code (00-59)
; SS = 2 digit second code (00-59)
; mmm = 3 digit millisecond code (optional if enabled with event-millis)

event-code = ( "7" | "8" | "9" ) 2*DIGIT ; three digit event code
; 7 series for status reports
; 8 series for medium priority
; 9 series for alarms (high priority)

event-info = <see event table for details of event information>
```

Millisecond timing can be added to events by setting the config parameter **event-millis** to **yes**. This adds a dot and a three digit millisecond count to the event time. It is diabled by default.

The event-time is based on the internal clock used by the C-Gate server. It is set to the system time at startup.

4.3.2.1 Event Buffer Overflow

Client programs that connect to command or event streams must frequently read any events sent to them. Failure to read often enough can result in events being lost as the C-Gate server can only buffer approximately 200 events before buffers overflow.

Any event stream that displays the message:

```
###!!!Event buffer overflow. Events have been missed.!!!###
```

has lost some events. If this occurs, the client software should be redesigned to read events more often.

4.3.2.2 Setting the event level

As a guide to setting the correct event level, the following table gives an overview of the levels and the types of events that occur at that level. As an example, setting the event level to 5 means that all events with a **lower** or **equal** event level will be sent to the event outputs.

Event level	Events at this level
1	Critical errors, such as syntax errors in **c-gateconfig, cgroups, networks** files. Internal errors in cgate Operational failure messages
2	Network errors Interface errors Discovery of unknown devices
3	Startup and shutdown advice messages Network sync failures Object information (from 701, GETSTATE command) Power usage (720)
4	Recoverable C-Bus communication errors: checksum failures, PCI busy errors

5	Heartbeart events (700) Network open/close advice Command interface open/close advice, Enable on/off advice
6	Network syncronisation messages
7	Reserved
8	Network device information (73x) Device configuration Reports from sync operations
9	Individual send/receive commands for each interface

In typical operation, event level 5 provides a good level of events. At a minimum, event level 3 allows the receipt of object information and critical errors. Event level 9 is principally used for diagnostic purposes.
The **global-event-level** parameter of the **cgate** system object sets the event-level used by all new objects when they are created. Setting this parameter to the right level at startup, via the **cgate** config file, will ensure that all objects report at the appropriate level. In addition, every object has a **event-level** parameter of its own, and this can be set from the command interface or in one of the other startup files to allow more detailed monitoring of a particular aspect of system operation.

4.3.2.3 Event Table

This section shows the event table information for C-Gate.

4.3.2.3.1 Information Event Codes

70x	Event Level	Information Code Message	Message
700	5	Heartbeat to indicate C-Gate server is operating.	"Heartbeat"
701	3	Object parameter information. This is an event that does nothing but report the state of a pre-defined object. These events can be triggered by the GETSTATE command.	parameter-name "=" value
702	8	Application information event - the text of this event varies with the particular C-Bus application that is using it. (This is also used by network variables in applications for similar reasons)	'[' application-name ']' application-information
703	3	Send an event cause by a broadcast_event command.	'broadcast_event' event-class [event-text]
704	3	Network state information	"interfaceType=" interface-type "interfaceAddress=" interface-address "state=" object-state "interfaceState=" interface-state

4.3.2.3.2 Energy Related

72x	Event Level	Energy Use Message	Message
720	3	Power usage report	An event indicating that the calculated power usage for the loads connected to the object (and sub-objects) has changed. This calculation requires that individual sub-unit definitions for wattage of loads are set. These are set on the load-power parameter for each unit and sub-unit.

			power-usage = *DIGIT
			"power=" power-usage "W"

4.3.2.3.3 Command Events

73x	Event Level	Command events	Message
730	7	New level advice for group	"New level=" *DIGIT "sourceunit=" *DIGIT "ramptime=" *DIGIT ["sessionID=" string "commandID=" strings
731	8	Unit advice	"unit type:" type "version:" version
732	8	Group advice	"group at" *DIGIT
733	9	Unit type advice	"address:" unit-address "unit type:" TOKEN "version:" version
734	9	Response line:	"response:" STRING
735	9	Command send advice	"send cmd" ["(fastpci)"]:" command-string
736	8	Application advice	"applications:" application-number "," application-number
737	8	Area advice	"area: " *DIGIT
738	7	New level advice from sync	"SyncUpdate: new level=" *DIGIT

4.3.2.3.4 Network Information

74x and 75x	Event Level	Network Information	Message
740	5	Opened C-Bus network	"Opened cbus network:" network-name
741	5	Closed C-Bus network	"Closed cbus network:" network-name
742	5	Network created	"Network created type=" type "address=" address
743	5	Network deleted	"Network deleted"
744	7	Unit online status has changed. This is used to indicate to clients that some units have changed their online statuses, e.g. online, offline, error, etc.	"Unit online status changed: " online_status_name=online_status_unit_list
751	5	Tag information has changed. This is used to indicate to clients that information has changed at this level or deeper, in the tag database.	"Tag information changed at tag address:" tag-address "oldtag:" old-TagName "newtag:" new-TagName
752	5	Indicates that as a part of a DBUPDATE operation, a unit element in the tag database has not been matched and will be moved aside (have its address set to null) and replaced with a new unit.	"Unmatched Unit element moved aside, OID=" oid

753	7	Provides detailed network sync information from background or interactive sync operations	"Net Sync:" details
754	7	Indicates attempted network state changes. Only appears if the state actually changed.	net-addr oid "State="state "Previous="state "Reason=" text
755	7	Indicates network interface state change	"State=" state
756	7	Indicates network synchronisation state change.	"SyncState=" sync-state sync-state = "idle" | "init" | "mmi" | "pci" | "bridge" | "units" | "app"
757	9	Indicates attempted network state changes. Only appears if the state didn't change value.	net-addr oid "State="state "Previous=(unchanged) Reason=" text
758	9	Provides detailed network parameters	"Network parameters: " + parameter list
759	9	Gives detailed network information about failures, errors and retries	• "command-fail-count=" + cmdFailCount + " unit=" + unitAddress + " reason=" + reason • "Checking PCI before retrying command (" + original command + ")" • "command timed out: " + original command + " after " + response delay + "ms, beginning retry " + retries + " of " + max retries • "command timed out: " + original command + " after " + response delay + "ms, retry limit of " + max retries + " reached" • "confirm-retry-count=" + retries + " (in packetConfirm)"

4.3.2.3.5 PCI Synchronisation

76x	Event - Level	PCI Synchronisation Event	Message
761	9	Command detail	"Command: " + command-line
762	6	Network synchronisation has succeeded	"Network sync ok"
763	6	Version 3 or later PCI detected (allowing fast communications)	"C-Bus PCI V3 detected at unit:" unit-address
765	8	Indication of confirm packet for sent command (with V3 PCI)	"got packet confirm:" CHAR CHAR

4.3.2.3.6 Scene and License

77x	Event Level	Scene or License Event	Message
770	6	Scene playing notification	"Playing scene:" scene-name
771	6	Scene recording notification	"Recording scene:" scene-name
772	6	Reloading scene notification	"Reloading scene:" scene-name
775	8	Licence check	"Checking license"

4.3.2.3.7 Unit Configuration Warnings

78x	Event Level	Unit configuration warnings (non-critical)	Message
781	8	The expected count of terminals for this output unit was not found in the unit	"Terminal count does not match for unit:" unit-number
782	8	The expected count of minimum values for this output unit was not found	"Min level terminal count does not match for unit:" unit-number
783	8	The expected count of group values was not found for this unit	"Group Value count does not match for unit:" unit-number
784	8	The expected set of current sensors does not match for this unit	"Current sense does not match for unit:" unit-number
785	8	The extendo-dignostic summary for this unit is missing or bad.	"Bad extendo-diagnostic summary for unit:" unit-number
786	8	Automatic database update failed	"Database update for unit" unit-address "failed (" message ")"
787	7	Warning of a unsigned unit specification in use.	"Unsigned unit specification: " filename " is being used"

4.3.2.3.8 Debugging Information

79x	Event Level	Information	Message
79x	9	No currently defined codes for this series	No messages yet

4.3.2.3.9 Command Related

80x	Event Level	Command Message Meaning	Message
800	5	Startup	"C-Gate started"
801	3	Shutdown	"C-Gate shutdown"
802	3	Restart	"C-Gate will restart"
803	5	Command interface or monitoring	• "Host:" hostname "/" ip-address

		interface opened	"opened command interface from port: " port-numbers • "Host:" hostname "/" ip-address "opened event interface from port: " port-numbers • "Host:" hostname "/" ip-address "opened load change interface from port: " port-numbers • "Host:" hostname "/" ip-address "opened config change interface from port: " port-numbers
804	5	Command interface or monitoring interface closed	• "Host:" hostname "/" ip-address "closed command interface from port: " port-number • "Host:" hostname "/" ip-address "closed event interface from port: " port-number • "Host:" hostname "/" ip-address "closed load change interface from port: " port-number • "Host:" hostname "/" ip-address "closed config change interface from port: " port-number
805	5	Command connection refused	"Host:" ip-address "command connection refused."
806	5	Connection refused by access control	"Access control refused" [connection-type] "connection from" STRING
807	?	Reserved for future audit commands	
808	?	Reserved for future audit commands	

4.3.2.3.10 Security Related

81x	Event Level	Security Message	Message
810	5	Enable succeeded	"Command interface:" interface-no "user:" username enabled
811	5	Enable failed	"Command interface:" interface-no "user:" username failed enable
812	5	End enable	"Command interface:" interface-no "user:" unenabled
813	5	Lock cancelled	"Lock cancelled name=" lock-name "network=" network-address "command_session=" command-session-id

4.3.2.3.11 Non Critical Network Errors

82x	Event Level	Network Error (Non Critical)	Message
820	2	Network unavailable	"Network" object-identifier "unavailable"
821	2	Unit unavailable	"Unit unavailable:" additional-details
822	2	Unit error	"Unit error:" unit-error information
823	3	PCI busy	"PCI busy indication"

824	2	Unknown cbus response	"unknown cbus response:" STRING
825	2	Response for unknown group	"response for unknown group:" STRING
826	4	C-Bus receive error	"cbus receive exception:" additional-information
827	2	C-Bus send error	"cbus send error" ["(fast pci)"] ":" additional-information
828	2	Checksum error	"receive checksum error"
829	3	New C-Bus unit not defined for this network	"new cbus unit: address=" unit-address "type=" unit-type "version=" version "not defined for this network."
830	3	New C-Bus group found on this application	"new group: address=" group-address "not defined for this application."
831	3	Error creating a unit as a C-Gate object	"error creating cbus unit: address="unit-address "type=" unit-type "version=" version
832	4	Response for unknown application	"Response for unknown application address=" application-address
833	3	New C-Bus area not defined for this application	"new cbus area: address=" unit-address "not defined for this application."
834	2	The transmitter thread has been restarted after being paused and not restarted due to communication issues	"Restarting stuck transmitter"
835	2	Failed to send command to C-Bus – timeout after retries	"C-Bus Send Error: command timed out, failed"
838	2	Unit is not responding and has been deleted	"Unit not responding, will be deleted"
839	5	Unit has new serial number and therefore appears to have been replaced	"Unit replaced: new serial=" serial-number
840	2	The PCI appears not to be responding, either because it isn't physically connected or it doesn't have power	"PCI not responding"
841	2	A network port is not available	"Network port not available"
842	2	The terminal adapter, providing RS232 connections to C-Bus PCIs, is not available	"Terminal adapter not available"
843	3	Synchronisation of groups on this application has failed.	"Group sync failed"
844	3	Synchronisation of units on this network has failed.	"Unit sync failed"
845	2	Network synchronization has failed	"Network sync failed"
850	5	Can't execute scene action	"Can't execute scene action for scene:" scene-name "address:" address
851	5	Application warning event generated by a C-Bus application	'[' application-name ']' application-warning-info
82x	**Event Level**	**Network Error (Non Critical)**	**Message**
852	3	Network powerup detected from PCI PUN message	"C-Bus Network Powerup Detected"
853	3	PCI parameter change notification	"PCI parameter change detected"

		received	

4.3.2.3.12 Critical Alarms

90x	Event Level	Critical Error Condition	Message
900	1	Internal error message	"Internal error: " debug-information
902	1	The unit specification file could not be found, so unit types can not be established	"Missing unit specification file" file-name
903	1	An exception was noted and information has been saved to a file.	"Critical Server Exception: refer exception.info for details"
904	1	The unit specification file (cbusunit.spe) is missing, so unit types can't be established.	"Missing unit specification file" filename

4.3.2.3.13 Connected Network Alarms

91x	Event Level	Connected network and interface alarms	Message
910	2	Can't open C-Bus network interface	"Can't open event interface at:" ip-address "port:" port-number
911	2	The expected count of minimum values for this output unit was not found	"Can't open:" debugging-information
912	2	Can't open command interface	"Can't open command interface on port:" *DIGIT
913	2	Can't start event server socket	"Can't start event server socket on port:" *DIGIT
914	2	Unable to open event file	"Unable to open event file:" file-name
915	2	Can't start config change port	"Can't start config change port socket on port:" *DIGIT
916	1	Unable to open event printer	"Unable to open event printer:" file-name
917	2	Can't start load change port	"Can't start load change port socket on port:" *DIGIT

4.3.2.3.14 Interface Errors

92x	Event Level	Interface errors	Message
920	1	A checksum error was noted in a received message	"Receive checksum error:"
921	1	Bad or unknown interface type	"Unknown/unimplemented CBus interface type" interface-type
922	1	Unable to open C-Bus Network	debugging-information

923	1	Error closing C-Bus Network	error-information

4.3.2.3.15 File errors

93x	Event Level	File Error	Message
931	1	Can't open networks file	"Can't open networks file:" file-name
932	1	Bad token in networks file	"bad token in networks file: " additional-detail
933	1	Can't open Cgroup file	"Can't open Cgroups file:" file-name
934	1	Bad token in networks file line	"Bad token in networks file line:" line-name
935	1	Syntax error in configuration parameter	"Syntax error:" additonal-details
936	1	C-Gate could not start the given class as a network interface	"Unable to start network class:" class "(" additional-details ")"
937	1	Unknown object in cgroup	"Unknown object " object "
938	8	Deprecated option in C-GateConfig.txt is not using the default value or the option is obsoleted	• "config warning: deprecated option has non-default value: " name=value • "config warning: obsolete option will be ignored: " name

4.3.2.3.16 Licensing Errors

94x	Event Level	Licensing error	Message
940	1	Hardware key not found	"Protection key not found" error-details
941	1	Network count exceeded	"Network license exceeded: (" *DIGIT "in use," *DIGIT "licensed.)"
942	1	C-Gate has shut down because there are more networks connected than are allowed for by the license	"Cgate Shutdown – licence exceeded"
945	1	Error event issued by a C-Bus application	[' application-name ']' application-error

4.3.2.3.17 Unit Specification Errors

95x	Event Level	Unit Specification Errors	Message
950	8	The expected count of terminals for this output unit was not found in the unit	"Bad class in unit specification type: " type "Class:" class-name "not found" "(" additional-information ")"
951	8	Unable to load the class indicated in the class specification file due to a	"Error in constructor for Type: "type "Class:" class-name "(" additional-

		error in the constructor	information ")"
952	8	Error reading unit specification	"Error reading unit specification:" file-name
953	8	Syntax error in unit specification file	"Syntax error in unit specification file:" name "after type:" details
954	8	Unable to start class to represent unit	"Unable to start unit class" "(" additional-detail ")"
955	8	No specification available for unit	"No specification for unit address:" unit-number "type:" type "version:" version "-using base CBusUnit"

4.3.2.3.18 Scene Errors

96x	Event Level	Scene Error	Message
961	1	"Unknown notify from scene control group. Scene" scene-name "group" group-number)	"Unknown notify from scene control group. Scene" scene-name "group" group-number)
962	1	Missing scene file	"Missing scene file" file-name
963	1	Error reading scene file	"Error reading scene file" file-name
964	1	Syntax error in scene file	"Syntax error in scene file:" file-name "last token:" TOKEN
965	1	Bad address in scene	"Bad address" group-address "in scene:" scene-name
966	1	Error writing scene	"Error writing recorded scene:" scene-name "error:" additional-information
967	1	Scene directory not found	"Scene directory not found:" directory-name

4.3.2.3.19 Access Control

97x	Event Level	Access Control	Message
970	1	Missing access control file	"Missing access control file" file-name
971	1	Error reading access control file	"Error reading access control file:" filename
972	1	Syntax error in access control file	"Syntax error in access control file:" file-name "after word:" TOKEN
974	1	Can't resolve address in access control file	"Can not resolve address: " address "on line" sp TOKEN sp "in access control file:" file-name
975	1	Bad access level in access control file	"Bad AccessLevel in access control file, setting to None"

4.3.3 Status Change Port

The Status Change Port or SCP as it is also known, allows listening clients to recieve an ongoing series of events that result from status changes in **C-Bus Applications**. Status changes include lighting application on, off and ramp messages along with messages for other C-Bus Applications.

The output of this port is rendered as a series of command interface commands and comments, which allows their recording for client macro facilities.

The Status Change Port is normally found at TCP/IP port 20025, or port 20125 for a secured SSL connection.

4.3.4 Config Change Port

The Config Change Port allows listening clients to see any configuration change in the C-Bus networks connected to C-Gate. These configuration changes include addition or deletion of C-Bus Units and unit synchronisation states.

The Config Change Port is normally found at TCP/IP port 20026, or port 20126 for a secured SSL connection.

4.3.5 Session ID and Command ID

All application events visible in the Status Change Port and the Event port can include additional sessionId and commandId information if the command resulting in the event was initiated from a C-Gate command session (and not from a unit on the network). This is in addition to the specified events as shown in the individual application's documentation.

Format

The additional information is a text string as shown below appended as the last item to the event:

```
"sessionID=" session-id "commandID=" command-id
```

where

`session-id` is a string that defines the individual instance of the command interface used to send the command. Typically this is something like `cmd`*n* where *n* is the count of command sessions that have been opened.

`command-id` is a a unique id given to the command as explained in the Unique Command ID section of this reference, or is given as `{none}` if no session was used to issue the command.

Note this is only added when a command is issued from a command interface.

The session id for a command interface can be determined by the SESSION_ID command.

4.3.6 SSL Connection

C-Gate supports SSL connections for the Command, Event, Config Change and Status Change interfaces when enabled..

The SSL interfaces are always enabled.

4.4 Logging

A summary of all logs in the system follows:

- C-Gate logs in \logs\event*.txt
- Installer logs in \logs\install\Setup Log*.txt
- Launcher logs in \logs\launcher.txt and \logs\wrapper.txt

History

- C-Gate v2.11.0 added a new launcher with new logs. See New vs Old Launcher.
- C-Gate v2.10.0 introduced a new logging system. See New vs old C-Gate log.

4.4.1 C-Gate logs

C-Gate implements logging by directing event messages to a file.

Log files are stored in the /logs subdirectory.

Log files are named as follows: `event-yyyymmdd-x.txt`

For example the first two logs for Christmas Day 2012 would be:

```
event-20121225-0.txt
event-20121225-1.txt
```

Default Logging Configuration

By default C-Gate logs at event level 9 and the log files roll over daily or when reaching a size of 5 megabytes. The file-name includes the time of creation. Logs are kept for seven days with older logs being discarded.

This default configuration is defined by the following configuration variables:

use-event-file is set to yes.
event-file.event-level is set to 9. See Setting the event level for more information.
event-file.keep-days is set to 7.
event-file.split-size is set to 5000000 (4.768 MB)

Increase the number of logs kept (recommended)

If storage space permits, please increase the amount of logs kept to retain more historical information. For example:

event-file.keep-days can be set to `28` to keep logs for four weeks instead of 7 days.

Restart C-Gate to make sure the changes take effect.

Adjust file sizes

The maximum size of individual files can be adjusted but this won't affect the total size of logs kept. For example:

event-file.split-size can be set to `20000000` to keep 20 megabyte files instead of 5.

Restart C-Gate to make sure the changes take effect.

Please note that log files greater than 5 MB are problematic to send via e-mail unless they are zipped beforehand.

Decrease the detail of logging

At the default event level of 9 the event file captures everything, including but not limited to commands, responses, status and config change events, C-Bus messages, and debugging information.

event-file.event-level can be set to `5` to reduce the level of logging but this may be insufficient for diagnostics.

4.4.2 New vs old C-Gate log

New logging system

The new logging system introduced in C-Gate v2.10.0 is different to the old system in that it:

- saves log files in a dedicated sub-directory.
- defaults to the maximum (level 9) amount of logging.
- the maximum size threshold is now defined in terms of days not megabytes. Care should be taken to ensure there is sufficient storage space for the logs.
- is asynchronous and should not block other threads.

The following configuration variable is still relevant in C-Gate but it no longer affects the logging system:

global-event-level. This should remain at level 5. See Setting the event level for more information.

These configuration variables are not relevant to the new logging system and will eventually be deprecated:

event-filename
event-file.split
event-file.split-count

Revert to the old Logging System

It is possible to revert to the old logging system used in C-Gate v2.9 and earlier with the following configuration:

use-event-file is set to `old`.
global-event-level is set to 5. See Setting the event level for details on how to do this.
event-file.split-size is set to 5000000.
event-filename is set to `event.log`.
event-file.split is set to `yes`.
event-file.split-count is set to `50`.

Restart C-Gate to make sure the changes take effect.

4.4.3 Installer logs

The C-Gate installer writes a log to the /logs/install subdirectory. The filename is in the format:

```
Setup Log YYYY-MM-DD #NNN.txt
```

The log is written here once the installation is successfully completed. If the installation is failing without being completed, you can find the installer log in this directory instead:

```
C:\Users\YOUR_ACCOUNT\AppData\Local\Temp
```

4.4.4 Launcher logs

The cgate.exe launcher logs to two files in the /logs subdirectory.

`launcher.txt`

Captures all output from the YAJSW library. This includes all the shell commands being executed and the versions of Java being used.

`wrapper.txt`

Captures the stdout and stderr output from cgate.jar. This includes any unhandled exceptions.

4.5 Command Descriptions

4.5.1

Command

#

Access Level

None

Syntax

```
# [any-text]
```

Use

Inserts a comment into the command stream. Comments are ignored by the command processor.

'//' can also be used as a comment marker.

Example

```
# turn on these lights
```

Successful Response

No response is given to the command

Failure response

No response is given to the command

See Also

//, NOOP

4.5.2 //

Command

```
//
```

Access Level

Any

Syntax

```
// [any-text]
```

Use

Inserts a comment into the command stream. Comments are ignored by the command processor.

Because // is also used in specifying a fully-qualified address including a project, it is only a comment when it is the first token on the line.

'#' can also be used as a comment marker.

Example

```
// try out the lights
```

Successful Response

No response is given to the command

Failure response

No response is given to the command

See Also

#, NOOP

4.5.3 AIRCON

Command

AIRCON

Access Level

Operate

Syntax

```
AIRCON [?]
```

Use

Lists the AIRCON sub-commands:

```
AIRCON REFRESH - Sends a refresh request to an air-conditioning ward.
AIRCON SET_HUMIDITY_SETBACK_LIMIT - Sets the error allowed in the set humidity for
zones.
AIRCON SET_HUMIDITY_LOWER_GUARD_LIMIT - Sets the absolute minimum humidity allowed
in zones.
AIRCON SET_HUMIDITY_UPPER_GUARD_LIMIT - Sets the absolute maximum humidity allowed
in zones.
AIRCON SET_HVAC_LOWER_GUARD_LIMIT - Sets the absolute minimum temperature allowed
in zones.
AIRCON SET_HVAC_SETBACK_LIMIT - Sets the error allowed in the set temperature for
zones.
AIRCON SET_HVAC_UPPER_GUARD_LIMIT - Sets the absolute maximum temperature allowed
in zones.
AIRCON SET_WARD_OFF - Switches off all plant in all the zones in the specified
ward.
AIRCON SET_WARD_ON - Returns an air-conditioning ward to its previous operational
state.
AIRCON SET_ZONE_HUMIDITY_MODE - Broadcasts Humidity mode and level required for
zones.
AIRCON SET_ZONE_HVAC_MODE - Broadcasts HVAC mode and level required for zones.
```

Success Response

A series of 101 Help lines giving a list of the AIRCON commands.

Failure Responses

```
400 Syntax Error
420 Access denied
```

4.5.4 AIRCON REFRESH Command

AIRCON REFRESH

Access Level

Operate

Syntax

```
AIRCON REFRESH app ward
```



```
app = air-conditioning application address (net/$AC)
ward = air-conditioning ward number (0..255)
```

Use

Sends a refresh request to an air-conditioning ward.

Example

```
aircon refresh 254/172 1
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.5 AIRCON SET_HUMIDITY_SETBACK_LIMIT

Command

AIRCON SET_HUMIDITY_SETBACK_LIMIT

Access Level

Operate

Syntax

```
AIRCON SET_HUMIDITY_SETBACK_LIMIT app ward zone-list level mode raw-flag

app = air-conditioning application address (net/$AC)
ward = air-conditioning ward number (0..255)
zone-list = comma-separated list of zone numbers (0..6) (0 is the 'unswitched'
zone)
level = required setback limit
mode = required Humidity mode
raw-flag = 0 if level is a humidity percentage, 1 if level is a raw value
```

- mode:
  0 = off
  1 = humidify only
  2 = dehumidify only
  3 = humidity control

- humidity level is 0% to 100%

- raw humidity level can be a fraction of plant capacity eg 50% or -10%

Use

Sets the error allowed in the set humidity for zones.

Example

```
aircon set_humidity_setback_limit 254/172 1 3,4 15 1 0
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.6 AIRCON SET_HUMIDITY_LOWER_GUARD_LIMIT

Command

AIRCON SET_HUMIDITY_LOWER_GUARD_LIMIT

Access Level

Operate

Syntax

```
AIRCON SET_HUMIDITY_LOWER_GUARD_LIMIT app ward zone-list level mode raw-flag

app = air-conditioning application address (net/$AC)
ward = air-conditioning ward number (0..255)
zone-list = comma-separated list of zone addresses (0..6) (0 is the 'unswitched'
zone)
level = required lower humidity limit
mode = required Humidity mode
raw-flag = 0 if level is a humidity percentage, 1 if level is a raw value
```

- mode:
  0 = off
  1 = humidify only
  2 = dehumidify only
  3 = humidity control

- humidity level is 0% to 100%

- raw humidity level can be a fraction of plant capacity eg 50% or -10%

Use

Sets the absolute minimum humidity allowed in zones.

Example

```
aircon set_humidity_lower_guard_limit 254/172 1 0,1,2 20 3 0
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.7 AIRCON SET_HUMIDITY_UPPER_GUARD_LIMIT

Command

AIRCON SET_HUMIDITY_UPPER_GUARD_LIMIT

Access Level

Operate

Syntax

```
AIRCON SET_HUMIDITY_UPPER_GUARD_LIMIT app ward zone-list level mode raw-flag

app = air-conditioning application address (net/$AC)
ward = air-conditioning ward number (0..255)
zone-list = comma-separated list of zone addresses (0..6) (0 is the 'unswitched'
zone)
level = required lower humidity limit
mode = required Humidity mode
raw-flag = 0 if level is a humidity percentage, 1 if level is a raw value
```

- mode:
  0 = off
  1 = humidify only
  2 = dehumidify only
  3 = humidity control

- humidity level is 0% to 100%

- raw humidity level can be a fraction of plant capacity eg 50% or -10%

Use

Sets the absolute maximum humidity allowed in zones.

Example


```
aircon set_humidity_lower_guard_limit 254/172 1 1,2 70 3 0
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.8 AIRCON SET_HVAC_LOWER_GUARD_LIMIT

Command

AIRCON SET_HVAC_LOWER_GUARD_LIMIT

Access Level

Operate

Syntax

```
AIRCON SET_HVAC_LOWER_GUARD_LIMIT app ward zone-list level mode raw-flag

app = air-conditioning application address (net/$AC)
ward = air-conditioning ward number (0..255)
zone-list = comma-separated list of zone addresses (0..6) (0 is the 'unswitched'
zone)
level = required lower temperature limit
mode = required HVAC mode
raw-flag = 0 if level is a temperature value, 1 if level is a raw value
```

- mode:
  0 = off
  1 = heat only
  2 = cool only
  3 = heat & cool
  4 = vent/fan only

- temperature level is in degrees centigrade

- raw temperature level can be a fraction of plant capacity, eg 50% or -10%

Use

Sets the absolute minimum temperature allowed in zones.

Example

```
aircon set_hvac_lower_guard_limit 254/172 1 0,1,2 18 3 0
```


Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.9 AIRCON SET_HVAC_SETBACK_LIMIT

Command

AIRCON SET_HVAC_SETBACK_LIMIT

Access Level

Operate

Syntax

```
AIRCON SET_HVAC_SETBACK_LIMIT app ward zone-list limit mode raw-flag

app = air-conditioning application address (net/$AC)
ward = air-conditioning ward number (0..255)
zone-list = comma-separated list of zone addresses (0..6) (0 is the 'unswitched'
zone)
limit = required setback limit
mode = required HVAC mode
raw-flag = 0 if level is a temperature value, 1 if level is a raw value
```

- mode:
  0 = off
  1 = heat only
  2 = cool only
  3 = heat & cool
  4 = vent/fan only

- temperature level is in degrees centigrade

- raw temperature level can be a fraction of plant capacity, eg 50% or -10%

Use

Sets the error allowed in the set temperature for zones.

Example

```
aircon set_hvac_setback_limit 254/172 1 0,1,2 2 3 0
```

Success Response


```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.10 AIRCON SET_HVAC_UPPER_GUARD_LIMIT Command

AIRCON SET_HVAC_UPPER_GUARD_LIMIT

Access Level

Operate

Syntax

```
AIRCON SET_HVAC_UPPER_GUARD_LIMIT app ward zone-list level mode raw-flag

app = air-conditioning application address (net/$AC)
ward = air-conditioning ward number (0..255)
zone-list = comma-separated list of zone addresses (0..6) (0 is the 'unswitched'
zone)
level = required lower temperature limit
mode = required HVAC mode
raw-flag = 0 if level is a temperature value, 1 if level is a raw value
```

- mode:
  0 = off
  1 = heat only
  2 = cool only
  3 = heat & cool
  4 = vent/fan only

- temperature level is in degrees centigrade

- raw temperature level can be a fraction of plant capacity, eg 50% or -10%

Use

Sets the absolute maximum temperature allowed in zones..

Example

```
aircon set_hvac_lower_guard_limit 254/172 1 1,2 27 3 0
```

Success Response

```
200 OK.
```


Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.11 AIRCON SET_WARD_OFF

Command

AIRCON SET_WARD_OFF

Access Level

Operate

Syntax

```
AIRCON SET_WARD_OFF app ward

app = air-conditioning application address (net/$AC)
ward = air-conditioning ward number (0..255)
```

Use

Switches off all plant in all of the zones in the specified ward.

Example

```
aircon set_ward_off 254/172 1
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.12 AIRCON SET_WARD_ON

Command

AIRCON SET_WARD_ON

Access Level

Operate

Syntax


```
AIRCON SET_WARD_ON app ward

app = air-conditioning application address (net/$AC)
ward = air-conditioning ward number (0-255)
```

Use

Returns an air-conditioning ward to its previous operational state.

Example

```
aircon set_ward_on 254/172 1
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.13 AIRCON SET_ZONE_HUMIDITY_MODE Command

AIRCON SET_ZONE_HUMIDITY_MODE

Access Level

Operate

Syntax

```
AIRCON SET_ZONE_HUMIDITY_MODE app ward zone-list mode raw-flag setback-enabled
guard-enabled use-aux-level type level aux-level

app = air-conditioning application address (net/$AC)
ward = air-conditioning ward number (0..255)
zone-list = comma-separated list of zone numbers (0..6) (0 is the 'unswitched'
zone)
mode = required Humidity mode
raw-flag = 0 if level is a humidity percentage, 1 if level is a raw value
setback-enabled = 1 for setback enabled
guard-enabled = 1 for guard enabled
use-aux-level = 1 if aux-level is used, else 0 for automatic operation
type = type of the Humidity plant
level = humidity % value or raw level
aux-level = auxiliary level value
```

- `mode:`
  `0 = off`


```
   1 = humidify only
   2 = dehumidify only
   3 = humidity control
```

- type:
  ```
   0 = none
   1 = evaporative
   2 = refrigerative
   3 = both
  ```

- humidity level is 0% to 100%

- raw humidity level can be a fraction of plant capacity eg 50% or -10%

- aux-level is flags & fan mode combined into one byte: RMmmmmmm where...
  ```
    R = reserved, always 0
    M = 0 for automatic fan, 1 for continuous fan
    fan speed mmmmmm: 0 = default speed, 1..63 = speed setting (plant dependent)
  For example, binary 01000011 = decimal 67, which is: fan on, speed = 3.
  binary 00000010 = decimal 2, which is: fan automatic, speed = 2.
  ```

Use

Broadcasts humidity mode and level required for zones.

Example

```
aircon set_zone_humidity_mode 254/172 1 0,1,2 3 0 0 0 1 2 40 64
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.14 AIRCON SET_ZONE_HVAC_MODE Command

AIRCON SET_ZONE_HVAC_MODE

Access Level

Operate

Syntax

```
AIRCON SET_ZONE_HVAC_MODE app ward zone-list mode raw-flag setback-enabled guard-
enabled use-aux-level type level aux-level
```



```
app = air-conditioning application address (net/$AC)
ward = air-conditioning ward number (0..255)
zone-list = comma-separated list of zone numbers (0..6) (0 is the 'unswitched'
zone)
mode = required HVAC mode
raw-flag = 0 if level is a temperature value, 1 if level is a raw value
setback-enabled = 1 for setback enabled
guard-enabled = 1 for guard enabled
use-aux-level = 1 if aux-level is used, else 0 for automatic operation
type = type of the HVAC plant
level = temperature or raw level
aux-level = auxiliary level value (eg fan speed and mode)
```

- mode:
  0 = off
  1 = heat only
  2 = cool only
  3 = heat & cool
  4 = vent/fan only

- type:
  0 = none
  1 = furnace
  2 = evaporative
  3 = reverse-cycle
  4 = heat-pump-heating
  5 = heat-pump-cooling
  6 = furnace/evap
  7 = furnace/heat-pump-cooling
  8 = hydronic
  9 = hydronic/heat-pump-cooling
 10 = hydronic/evap
 11-254 = reserved (don't use)
255 = any (type 255 allows service to select a plant type from those available)

- temperature is in degrees centigrade

- raw temperature level can be a fraction of plant capacity, eg 50% or -10%

- aux-level is flags & fan mode combined into one byte: RMmmmmmm where...
  R = reserved, always 0
  M = 0 for automatic fan, 1 for continuous fan
  fan speed mmmmmm: 0 = default speed, 1..63 = speed setting (plant dependent)
  For example, binary 01000011 = decimal 67, which is: fan on, speed = 3.
  binary 00000010 = decimal 2, which is: fan automatic, speed = 2.

Use

Broadcasts HVAC mode and level required for zones.

Example

```
aircon set_zone_hvac_mode 254/172 1 0,1,2 3 0 1 0 1 255 23 64
```

Success Response


```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.15 APIVER

Command

APIVER

Access Level

Monitor

Syntax

```
APIVER [details]
```

Use

Lists API versions for C-Gate commands, modules and othe components.

Add the `'details'` option to give the details of the last API change.

Example

```
apiver
138-event=1.0
138-ccp=1.0
138-scp=1.0
138-schema-tag=1.1
138-schema-unitspec=1.0
138-schema-patch=1.0
138-schema-cbusunits=1.0
138-schema-transform=1.0
138-schema-applications=1.0
138-obj-application=1.2
138-obj-unit=1.4
138-obj-network=1.0
138-cmd-syntax=1.0
138-cmd-project=1.1
138-cmd-pp=1.1
138-cmd-net=1.0
138-cmd-port=1.0
138-cmd-file=1.0
138-cmd-general=1.0
138-cmd-cbus=1.0
```



```
138-cmd-db=1.1
138-cmd-transform=1.1
138-topology=1.0
138-calculator=1.0
138-config=1.0
138-access=1.0
138 scene=1.0

apiver details
138-event=1.0 LastChange="initial version"
138-ccp=1.0 LastChange="initial version"
138-scp=1.0 LastChange="initial version"
138-schema-tag=1.1 LastChange="DBVersion updated to 2.1"
138-schema-unitspec=1.0 LastChange="initial version"
138-schema-patch=1.0 LastChange="initial version"
138-schema-cbusunits=1.0 LastChange="initial version"
138-schema-transform=1.0 LastChange="initial version"
138-schema-applications=1.0 LastChange="initial version"
138-obj-application=1.2 LastChange="added ereport application"
138-obj-unit=1.4 LastChange="new class and added params for DALI gatetway"
138-obj-network=1.0 LastChange="initial version"
138-cmd-syntax=1.0 LastChange="initial version"
138-cmd-project=1.1 LastChange="additional responses to indicate wrong project
DBVersion"
138-cmd-pp=1.1 LastChange="fixed bugs in bit field processing (4879)"
138-cmd-net=1.0 LastChange="initial version"
138-cmd-port=1.0 LastChange="initial version"
138-cmd-file=1.0 LastChange="initial version"
138-cmd-general=1.0 LastChange="initial version"
138-cmd-cbus=1.0 LastChange="initial version"
138-cmd-db=1.1 LastChange="fixed bug in hashtable creation stopping dbset working
properly"
138-cmd-transform=1.1 LastChange="minor command response syntax changes"
138-topology=1.0 LastChange="initial version"
138-calculator=1.0 LastChange="initial version"
138-config=1.0 LastChange="initial version"
138-access=1.0 LastChange="initial version"
138 scene=1.0 LastChange="initial version"
```

Success Response

A series of 138 messages. See samples above.

Failure Responses

```
400 Syntax Error
420 Access denied
```

4.5.16 AUDIO

Command

AUDIO

Access Level

Operate

Syntax

```
AUDIO [?]
```

Use

Lists the AUDIO sub-commands:

```
AUDIO CURRENT_FEED - Reports the current feed for the given zone.
AUDIO DYNAMIC_1 - Request a matrix switcher to send a Dynamic 1 operation in the
given zone.
AUDIO DYNAMIC_2 - Request a matrix switcher to send a Dynamic 2 operation in the
given zone.
AUDIO HIGH_PRIORITY - Request an output device to turn on and go to a set output
level and feed.
AUDIO MUTE - Set the mute mode of an amplifier.
AUDIO NEXT_FEED - Set the next feed for the given zone.
AUDIO NEXT_LANGUAGE - Set the next language for the given zone.
AUDIO OFF - Send an off operation.
AUDIO ON - Send an on operation.
AUDIO OUTPUT_COMMON_CONTROL - Request all devices to perform a function.
AUDIO OUTPUT_DEVICE_STATUS_REQUEST - Request status of all devices.
AUDIO OUTPUT_ERROR_CODE - Send an error code notification.
AUDIO PREVIOUS_FEED - Set the previous feed for the given zone.
AUDIO RAMP - Send a ramp operation.
AUDIO REQUEST_CURRENT_FEED - Request the current feed of a zone.
AUDIO SET_FEED - Set the current feed of a zone.
AUDIO TERMINATERAMP - Terminate a ramp in progress.
AUDIO ZONE_DESCRIPTOR_REQUEST - Send feed description to DLT labelling devices.
AUDIO ZONE_FEED_LABEL_REQUEST - Send feed description and dynamic labels to DLT
labelling devices.
```

Success Response

A series of 101 Help lines giving a list of the AUDIO commands.

Failure Responses

```
400 Syntax Error
420 Access denied
```

4.5.17 AUDIO CURRENT_FEED

Command

AUDIO CURRENT_FEED

Access Level

Operate

Syntax


```
AUDIO CURRENT_FEED app multiplexer zone feed gain
AUDIO CURRENT_FEED app Z function gain


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
feed = feed number (0..7)
function = zone function (0..255)
gain = amount of gain (0..4)
```

Use

Reports the current feed for the given zone.

Example

```
audio current_feed 254/205 2 7 7 4
audio current_feed 254/205 Z 192 4
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.18 AUDIO DYNAMIC_1

Command

AUDIO DYNAMIC_1

Access Level

Operate

Syntax

```
AUDIO DYNAMIC_1 app multiplexer zone
AUDIO DYNAMIC_1 app Z function


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```


Use

Instruct the matrix switcher in the given zone to send a feed-specific Dynamic Function 2 command to the internal NIRT / C-Bus.

Example

```
audio dynamic_1 254/205 1 2
audio dynamic_1 254/205 Z 192
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.19 AUDIO DYNAMIC_2

Command

AUDIO DYNAMIC_2

Access Level

Operate

Syntax

```
AUDIO DYNAMIC_2 app multiplexer zone
AUDIO DYNAMIC_2 app Z function


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Use

Instruct the matrix switcher in the given zone to send a feed-specific Dynamic Function 1 command to the internal NIRT / C-Bus.

Example

```
audio dynamic_2 254/205 1 2
audio dynamic_2 254/205 Z 192
```


Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.20 AUDIO HIGH_PRIORITY

Command

AUDIO HIGH_PRIORITY

Access Level

Operate

Syntax

```
AUDIO HIGH_PRIORITY app multiplexer level feed


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
level = level (0..255)
feed = feed number (0..7)
```

Use

Request an output device in a zone to turn on and go to a set output level and feed, or if the level is zero to return to the prior state.

Example

```
audio high_priority 254/205 2 192 7
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```


4.5.21 AUDIO MUTE

Command

AUDIO MUTE

Access Level

Operate

Syntax

```
AUDIO MUTE app multiplexer zone mode
AUDIO MUTE app Z function mode

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
mode = mode (0, 2, 5, 7, 255)
           0 = turn amplifier off
           2 = amplifier on, volume normal, speakers off
           5 = amplifier on, volume preset, speakers off
           7 = amplifier on, volume preset, speakers on
           255 = amplifier on, volume normal, speakers on
```

Use

Set the mute mode of an amplifier.

Example

```
audio MUTE 254/205 2 1 7
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.22 AUDIO NEXT_FEED

Command

AUDIO NEXT_FEED

Access Level

Operate

Syntax

```
AUDIO NEXT_FEED app multiplexer zone
AUDIO NEXT_FEED app Z function


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Use

Sets the next feed for the given zone.

Example

```
audio next_feed 254/205 2 7
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.23 AUDIO NEXT_LANGUAGE Command

AUDIO NEXT_LANGUAGE

Access Level

Operate

Syntax

```
AUDIO NEXT_LANGUAGE app multiplexer zone
AUDIO NEXT_LANGUAGE app Z function


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```


Use

Sets the next language for the given zone.

Example

```
audio next_language 254/205 2 7
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.24 AUDIO OFF

Command

AUDIO OFF

Access Level

Operate

Syntax

```
AUDIO OFF app multiplexer zone code
AUDIO OFF app Z function


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
code = zone code (0..7)
function = zone function (0..255)
```

Use

Set the value of the Function Code referenced by the Zone Function Variable to 0.

Example

```
audio off 254/205 2 4 7
```

Success Response

```
200 OK.
```


Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.25 AUDIO ON

Command

AUDIO ON

Access Level

Operate

Syntax

```
AUDIO ON app multiplexer zone code
AUDIO ON app Z function


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
code = zone code (0..7)
function = zone function (0..255)
```

Use

Set the value of the Function Code referenced by the Zone Function Variable to 255.

Example

```
audio on 254/205 2 4 7
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.26 AUDIO OUTPUT_COMMON_CONTROL

Command

AUDIO OUTPUT_COMMON_CONTROL

Access Level

Operate

Syntax

```
AUDIO OUTPUT_COMMON_CONTROL app control-code


app = audio application address (net/$CD)
control-code = control code (0)
           0 = turn off
```

Use

All output devices are to act upon this command according to the value of <Control Code>.

Example

```
audio output_common_control 254/205 0
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.27 AUDIO OUTPUT_DEVICE_STATUS_REQUEST

Command

AUDIO OUTPUT_DEVICE_STATUS_REQUEST

Access Level

Operate

Syntax

```
AUDIO OUTPUT_DEVICE_STATUS_REQUEST app parameter


app = audio application address (net/$CD)
parameter = parameter (0)
```


Use

Requests that all output devices return their status.

Example

```
audio output_device_status_request 254/205 0
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.28 AUDIO OUTPUT_ERROR_CODE Command

AUDIO OUTPUT_ERROR_CODE

Access Level

Operate

Syntax

```
AUDIO OUTPUT_ERROR_CODE app multiplexer zone code


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
code = error code (0..1)
           0 = overheating
           1 = low voltage
```

Use

A device is reporting an error condition.

Example

```
audio output_error_code 254/205 1 2 0
```

Success Response

```
200 OK.
```


Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.29 AUDIO PREVIOUS_FEED Command

AUDIO PREVIOUS_FEED

Access Level

Operate

Syntax

```
AUDIO PREVIOUS_FEED app multiplexer zone
AUDIO PREVIOUS_FEED app Z function


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Use

Sets the previous feed for the given zone.

Example

```
audio previous_feed 254/205 2 7
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.30 AUDIO RAMP Command

AUDIO RAMP

Access Level

Operate

Syntax

```
AUDIO RAMP app multiplexer zone code level rate
AUDIO RAMP app Z function level rate


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
code = zone code (0..7)
function = zone function (0..255)
level = level to ramp to (0..255)
rate = time taken to perform the ramp (0..15)
          0 = instantaneous
          1 = 4 seconds
          2 = 8 seconds
          3 = 12 seconds
          4 = 20 seconds
          5 = 30 seconds
          6 = 40 seconds
          7 = 1 minute
          8 = 90 seconds
          9 = 2 minutes
          10 = 3 minutes
          11 = 5 minutes
          12 = 7 minutes
          13 = 10 minutes
          14 = 15 minutes
          15 = 17 minutes
```

Use

Ramp the zone function to the specified level at the given rate.

Example

```
audio ramp 254/205 2 4 7 255 1
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```


4.5.31 AUDIO REQUEST_CURRENT_FEED Command

AUDIO REQUEST_CURRENT_FEED

Access Level

Operate

Syntax

```
AUDIO REQUEST_CURRENT_FEED app multiplexer zone
AUDIO REQUEST_CURRENT_FEED app Z function


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Use

Request the current feed for the given zone.

Example

```
audio request_current_feed 254/205 2 7
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.32 AUDIO SET_FEED Command

AUDIO SET_FEED

Access Level

Operate

Syntax

```
AUDIO SET_FEED app multiplexer zone feed option
```



```
AUDIO SET_FEED app Z function option


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
feed = feed number (0..7)
function = zone function (0..255)
option = option (0..1)
           0 = allow labels and annunciation on feed change
           1 = disable labels and annunciation on feed change
```

Use

Sets the current feed for the given device.

Example

```
audio set_feed 254/205 2 7 7 4
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.33 AUDIO TERMINATERAMP

Command

AUDIO TERMINATERAMP

Access Level

Operate

Syntax

```
AUDIO TERMINATERAMP app multiplexer zone code
AUDIO TERMINATERAMP app Z function


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
code = zone code (0..7)
function = zone function (0..255)
```


Use

Terminate a ramp operation in progress on the given zone.

Example

```
audio terminateramp 254/205 2 4 7
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.34 AUDIO ZONE_DESCRIPTOR_REQUEST Command

AUDIO ZONE_DESCRIPTOR_REQUEST

Access Level

Operate

Syntax

```
AUDIO ZONE_DESCRIPTOR_REQUEST app multiplexer zone
AUDIO ZONE_DESCRIPTOR_REQUEST app Z function


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Use

Instruct the matrix switcher in the given zone to send the feed description to DLT labelling devices.

Example

```
audio zone_descriptor_request 254/205 1 2
```

Success Response

```
200 OK.
```


Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.35 AUDIO ZONE_FEED_LABEL_REQUEST Command

AUDIO ZONE_FEED_LABEL_REQUEST

Access Level

Operate

Syntax

```
AUDIO ZONE_FEED_LABEL_REQUEST app multiplexer zone
AUDIO ZONE_FEED_LABEL_REQUEST app Z function


app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Use

Instruct the matrix switcher in the given zone to send the feed description, Dynamic 1 and 2 labels to DLT labelling devices.

Example

```
audio zone_feed_label_request 254/205 1 2
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error:
401 Bad object or device ID:
405 Parameter out of range:
408 Operation failed
```

4.5.36 BROADCAST_EVENT Command

BROADCAST_EVENT

Access Level

Operate

Syntax

```
BROADCAST_EVENT event-class [event-text]

event-class = A string (ending at the first whitespace) that gives the class of
this event.
              There are no restriction on values of the class. This is determined
by the using application.

event-text = (optional) Further descriptive string to the end of the line.
```

Use

Broadcasts an event on the event port and status change port with the given *event-class* and *event-text*.

Example

```
broadcast_event ShutDown C-Gate will be shutdown in 10 minutes
200 OK.
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
420 Access denied
```

4.5.37 CGL Command

CGL

Access Level

Admin

Syntax

```
CGL [?]
```

Use

Lists the CGL sub-commands:

CGL EXPORT - Exports a network to CGL format.
CGL IMPORT - Imports a network in CGL format to the given project.

Success response

A series of lines listing these commands in the form:

```
101-Help: help-information
```

4.5.38 CGL EXPORT

Command

CGL EXPORT

Access Level

Monitor

Syntax

```
CGL EXPORT project-name [network-list [application-list]]
```

Use

Gets a part of the current Tag Database as a snippet in CGL Format.

The db-address given is any object in the XML database, and can be specified as either a path from the base of the tag database (ie /Installation/InstallationDetail/Installer), or a C-Bus network address, of a tag name.

This command returns:
- a 343-Begin XML snippet response
- The XML snippet requested, as one or more lines starting with "347-". This starts with a <?xml ... ?> line. Note that the XML is not pretty-printed.
- a 344 End XML snippet response

If a single element is selected by address, for example an element that simply contains a string, then an XML snippet similar to the following will be returned just giving the raw type of the field:

```
347-<?xml version="1.0"?>
347-<string>2001-11-01T13:05:34.153</string>
```

Examples

```
DBGETXML Installation/Version
DBGETXML p/1/20
```

Successful Response (example)

```
343-Begin XML snippet
347-<?xml version="1.0"?>
347-<Unit xmlns="http://www.clipsal.com/cis/schema/2001/cbus.xsd">
```



```
347-<TagName>n1_u20</TagName><Address>20</Address><UnitType>KEYM8</
UnitType><UnitName>NEWUNIT </UnitName><FirmwareVersion>1.3.06</
FirmwareVersion></Unit>
344 End XML snippet
```

Failure response

```
401 Bad object or device ID
440 There is no tag database to perform this operation on.
444 XML creation failed
```

See also

CGL IMPORT

4.5.39 CGL IMPORT

Command

CGL IMPORT

Access Level

Admin

Syntax

```
CGL IMPORT project-name << end-tag
```

Use

Replaces all or part of the current Tag Database with a snippet in CGL Format given after the command. The project-name must be a project that has already been loaded in C-Gate.

The CGL required for this command is provided as a here document, meaning is it provided in one or more lines following the command, with the command terminated by the `end-tag` given on the command line after the << characters, on a line by itself.

This command returns:

- a series of 380 messages for each item that was added or modified.
- a 200 OK message when concluded.
- one or more errors if other conditions occur.

Examples

```
CGL IMPORT home << EOF
.. one or more lines of CGL data ...
EOF
```

Successful Response

```
301 OID=oid-value
```

This response returns the OID of the created element

Failure response

```
401 Bad object or device ID: No project specified
401 Bad object or device ID: Project not found
400 Syntax Error: No CGL data supplied
408 CGL valiation failed:
408 CGL import failed:
```

See also

CGL EXPORT

4.5.40 CLOCK

Command

CLOCK

Access Level

Operate

Syntax

```
CLOCK [?]
```

Use

Lists the CLOCK sub-commands:
`CLOCK DATE` - Gets or sets the date.
`CLOCK REQUEST REFRESH` - Request a time-master to broadcast the date and time.
`CLOCK TIME` - Gets or sets the time.

Success response

A series of lines giving help for these commands in the form: `101 Help: help-information`

Failure responses

```
400 Syntax Error
420 Access denied
```

4.5.41 CLOCK DATE

Command

CLOCK DATE

Access Level

Operate

Syntax


```
CLOCK DATE app [date | "system"]

app = clock application address (net/223)
date = yyyy-mm-dd

• yyyy = 4 year date (eg. 2008)
• mm = months (01..12)
• dd = days (01-31)

"system" -if this is given instead of yyyy-mm-dd, C-Gate's current date is used
```

Use

Gets or sets the date.

`CLOCK DATE app` with no parameters will return the current date. Adding a date will set the network date by sending a `clock date` command to the network.

Success response

```
232 Date set to: yyyy-mm-dd dow

• yyyy = 4 year date (eg. 2008)
• mm = months (01..12)
• dd = days (01-31)
• dow = day of week (0..6 where 0 = Monday)
```

Failure responses

```
401 Bad object or device ID
402 Operation not supported by: ...
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.42 CLOCK REQUEST_REFRESH

Command

CLOCK REQUEST_REFRESH

Access Level

Operate

Syntax

```
CLOCK REQUEST_REFRESH app

app = clock application address (net/223)
```

Use

Broadcasts a request_refresh message, which will cause a time-master device to broadcast the current date and time.

Example

```
clock request_refresh 254/223
```

Success response

```
200 OK.
```

Failure responses

```
400 Syntax Error
```

4.5.43 CLOCK TIME

Command

CLOCK TIME

Access Level

Operate

Syntax

```
CLOCK TIME app [time | "system" [daylight-flag]]

app = clock application address (net/223)
time = hh:mm:ss
-if "system" is given instead, C-Gate's current time is used
```

- hh = 00..23
- mm = 00..59
- ss = 00..59

```
daylight-flag:
```
- 0 = no daylight saving
- 1 = this time includes 1 hr advance
- 255 = daylight saving offset unknown

- If the daylight-flag is omitted, it defaults to a value of 255 ($FF).

Use

Gets or sets the time.

`CLOCK TIME app` with no parameters will return the current time as understood by this application. Adding a time and possibly a daylight saving flag will set the network time by sending a clock time command to the network.

Success response


```
231 Time set to: hh:mm:ss daylight-flag
```

- hh = 00..23
- mm = 00..59
- ss = 00..59
- daylight-flag = 0 | 1 | 255

Failure responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.44 CONFIG

Command

CONFIG or CONFIG ?

Access Level

Admin

Syntax

```
CONFIG [?]
```

Use

Lists the CONFIG sub-commands:

```
CONFIG GET - Get the value of a configuration parameter
CONFIG INFO - Get information about a configuration parameter
CONFIG LOAD - Load config parameters from either a project db or a file or both
CONFIG SAVE - Save the current config parameters to either a project db or a file
CONFIG SET - Set the value of a configuration parameter
CONFIG OBGET - Get the value of a configuration parameter for a scope object
CONFIG OBSET - Set the value of a configuration parameter for a scope object
```

Success Response

A series of 101 Help lines giving details of the config commands

Failure Responses

```
400 Syntax Error
420 Access denied
```

4.5.45 CONFIG GET

Command

CONFIG GET

Access Level

Admin

Syntax

```
CONFIG GET config-parameter

config-parameter = config parameter name or "*" to get all config parameters
```

Use

Returns the value of a configuration parameter, or values of all known configuration parameters if the parameter is given as `*`.

See Configuration for more information.

Example

```
config get sync-time
303 sync-time=200
```

Success Response

```
303 config-parameter "=" value
```

Failure Responses

```
400 Syntax Error
408 Operation failed: config parameter not found
420 Access denied
```

4.5.46 CONFIG INFO

Command

CONFIG INFO

Access Level

Admin

Syntax

```
CONFIG INFO config-parameter

config-parameter = config parameter name or "*" to get all config parameters
```

Use

Returns full information over several lines about a configuration parameter, or values of all known configuration parameters if the parameter is given as `*`.

Example

```
config info config-path
304-parameter=config-path
304-value=config
304-description=Path to directory where configuration files are held
304-defaultValue=config
304-scope=global
304 effective=restart
```

Success Response

Six lines per parameter in the following form and order:

```
304-parameter= name of the parameter
304-value= current value of the parameter
304-description= description of this parameter
304-defaultValue= the default value of this parameter
304-scope= the scope of this parameter (what part of the server is it active over)
304 effective= when do changes of this parameter come into effect
```

See the Configuration section in this guide to decode the meanings of these lines.

Failure Responses

```
400 Syntax Error
408 Operation failed: config parameter not found
420 Access denied
```

4.5.47 CONFIG LOAD

Command

CONFIG LOAD

Access Level

Admin

Syntax

```
CONFIG LOAD config-type [filename]

config-type = "project" | "global" | "all"
```

Use

Loads configuration parameters into the server from either the current project database (as defined by the PROJECT USE command and the PROJECT commands) or from the global configuration file, or both.

Use this command to load an existing on-disk configuration into the server. Note that there may be some immediate impact from the loading of some parameters.

Example

```
config load global
200 OK.

# or

config load project
200 OK.
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
408 Operation failed: No project in use
408 Operation failed: Global config load from filename failed: further-details
420 Access denied
```

4.5.48 CONFIG SAVE

Command

CONFIG SAVE

Access Level

Admin

Syntax

```
CONFIG SAVE config-type [filename]

config-type = "project" | "global" | "all"
```

Use

Saves configuration parameters from the server to either the current project database `"project"` (as defined by the PROJECT USE command and the PROJECT commands) or to the global configuration file `"global"`, or both `"all"`.

Use this command to save a current configuration for later use. This has no impact on the currently operating parameters in the server.

Example

```
config save global
200 OK.

# or
```



```
config save all
200 OK.
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
408 Operation failed: No project in use
408 Operation failed: Global config save to filename failed: further-details
420 Access denied
```

4.5.49 CONFIG SET

Command

CONFIG SET

Access Level

Admin

Syntax

```
CONFIG SET config-parameter [value]

config-parameter = config parameter name
value = new value for the parameter, can include spaces
```

Use

Sets the value of a configuration parameter to the given value. if the value is not given, the parameter takes on the null value. The value given can have embedded spaces and these will be included in the value.

See Configuration for more information on setting config parameters, their scope and when changes are effective.

Example

```
config set sync-time 100
200 OK.
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
408 Operation failed: unknown parameter parameter-name
420 Access denied
```


4.5.50 CONFIG OBGET

Command

CONFIG OBGET

Access Level

Admin

Syntax

```
CONFIG OBGET object config-parameter

object = the global, project or network scope object from which to get the config
parameter
config-parameter = config parameter name or "*" to get all config parameters
```

Use

Returns the value of a configuration parameter of the object, or values of all known configuration parameters of the object if the parameter is given as *.

See Configuration for more information. See Scope and Scope Objects for more information on scope objects.

Example

```
config obget global global-event-level
303 global-event-level=5

config obget project network.retries
303 network.retries=2

config obget 254 sync-time
303 sync-time=200
```

Success Response

```
303 config-parameter "=" value
```

Failure Responses

```
400 Syntax Error
408 Operation failed: config parameter not found
420 Access denied
```

4.5.51 CONFIG OBSET

Command

CONFIG OBSET

Access Level

Admin

Syntax

```
CONFIG OBSET object config-parameter [value]

object = the global, project or network scope object on which to set the config
parameter
config-parameter = config parameter name
value = new value for the parameter, can include spaces
```

Use

Sets the value of a configuration parameter of an object to the given value. if the value is not given, the parameter takes on the null value. The value given can have embedded spaces and these will be included in the value.

See Configuration for more information. See Scope and Scope Objects for more information on scope objects.

Example

```
config obset sync-time 100
200 OK.
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
408 Operation failed: unknown parameter parameter-name
420 Access denied
```

4.5.52 CONFIG OBRESET Command

CONFIG OBRESET

Access Level

Admin

Syntax

```
CONFIG OBRESET object [config-parameter]

object = the project or network scope object on which to reset the config
parameter to the global value
config-parameter = config parameter name
```


Use

Resets the value of a configuration parameter of an object to the global value. if the parameter is not given, all parameters are reset.

See Configuration for more information. See Scope and Scope Objects for more information on scope objects.

Example

```
config obreset sync-time
200 OK.
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
408 Operation failed: unknown parameter parameter-name
420 Access denied
```

4.5.53 CONFIRM Command

CONFIRM

Access Level

Admin

Syntax

```
CONFIRM
```

Use

Used when a 600 message is received, to confirm the execution of a shutdown or restart before it proceeds. This is meant to stop people shutting down the server by having their fingers typing on automatic.

Examples

```
#in response to 600 message from shutdown or restart

CONFIRM
```

Successful Response

```
206 Shutdown confirmed.
```


Failure response

```
400, 420
```

See Also

SHUTDOWN

4.5.54 DBADD

Command

DBADD

Access Level

Operate

Syntax

```
DBADD db-address element-type

db-address = address of an existing element under which to add the new element
element-type = type of the new object to add to the database
Suitable types include: Project, Network, Unit, Application, Group, Level.
```

Use

Adds an object to a database.

Examples

```
DBADD Installation InstallationDetail
```

Successful Response

`200 OK.`
(where no OID is used for this element), or

`301 OID=`*`oid-value`*
(where an OID is applicable to the element that has been created)

Failure response

```
400 Syntax Error.
401 Bad object or device ID:
440 There is no tag database to perform this operation on.
```

4.5.55 DBADDSAFE

Command

DBADDSAFE

Access Level

Operate

Syntax

```
DBADDSAFE parent-address element-type [element-address] [tag-name]

parent-address = address of an existing element under which to add the new element
element-type = type of the new object to add to the database
element-address = address of the new object, it is mandatory if the object has an
Address field
tag-name = tag name of the new object, it is mandatory if the object has a TagName
field
Allowed types include but not limited to: Project, Network, Unit, Application,
Group, Level, TagDLT.
```

Use

Adds an object to a database and apply business rules check when the object has Address and TagName fields.

The rules for element-address:

- it can not be null or blank
- if element-type is Network, Unit, Application, Group or Level, it must be the string representation of decimal number (i.e. only 0~9 is allowed in the element-address parameter)
- if element-type is Network, Unit, Application, Group or Level, it must be within the range of [0, 255]
- it can not be the same as the address of an existing object under parent-address

The rules for tag-name:

- it can not be null or blank
- it can not be the same as the tag name of an existing object under parent-address

Additional rules:

- Group object can not be added to Application 255
- Level object can not be added to Group 255

Examples

```
DBADDSAFE 254/56 Group 0 Lighting Group 0
```

Successful Response

`200 OK.`
(where no OID is used for this element), or

`301 OID=`*`oid-value`*
(where an OID is applicable to the element that has been created)

Failure response


```
400 Syntax Error.
401 Bad object or device ID: Can't add groups to Application 255
401 Bad object or device ID: Can't add levels to Group 255
401 Bad object or device ID: Element address in use
401 Bad object or device ID: Element address is required for this type
401 Bad object or device ID: Element address must be a number
401 Bad object or device ID: Element address must be in the range of 0 to 255
401 Bad object or device ID: Element tag name is required for this type
401 Bad object or device ID: Element tagname in use
401 Bad object or device ID: Parent address is not unique. Unique parent address
required for dbaddsafe
401 Bad object or device ID: Parent element not available
401 Bad object or device ID: Unable to add element: Field <elementType> not found
in <parentAddress>.
401 Bad object or device ID: Unable to add element: <exception info>
401 Bad object or device ID: <exception info>
440 There is no tag database to perform this operation on.
```

4.5.56 DBCOPY

Command

DBCOPY

Access Level

Operate

Syntax

```
DBCOPY source-address destination-parent-address

source-address = the database address of the part of the database to copy
destination-parent-address = the parent element to hold the copied part of the
database
```

Use

Copies part of a tag database tree to another place. This means a unit or network definiton, for example, can be copied to create a new entry.

You can copy parts of databases between projects by using //*project-name*/ as address prefixes

Notes:

The destination part of the tree has new object identifiers (OIDs) generated for all elements that have OIDs.

If the source and destination databases are the same (copying within one database) then the TagName and Address fields are not copied and are set to null in the destination. These addresses will need to be set to allow the database contents to be addressed by Address and TagName.

However, if he source and destination databases or projects are different, then TagNames and Addresses are copied with the rest of the elements.

Examples

```
# copy the network adressed as j1 in project jim to the current project
called system
DBCOPY //jim/j1 system

# copy the unit with tagname newunit in project jim to network 1 on
project system
DBCOPY //jim/newunit //system/1
```

Successful Response

`200 OK.`
(where no OID is used for this element), or

`301 OID=oid-value`
(where an OID is applicable to the element that has been created)

Failure response

```
400 Syntax Error.
401 Bad object or device ID: reason
408 Operation failed: reason
440 There is no tag database to perform this operation on.
```

4.5.57 DBCOPYSAFE

Command

DBCOPYSAFE

Access Level

Operate

Syntax

```
DBCOPYSAFE source-address destination-parent-address [destination-element-address]
[destination-tag-name]

source-address = the database address of the part of the database to copy
destination-parent-address = the parent element to hold the copied part of the
database
destination-element-address = the address of the copied object; it is mandatory if
the object has an Address field and is being copied to the same parent node
destination-tag-name = the tag name of the copied object; it is mandatory if the
object has a TagName field and is being copied to the same parent node
```

Use

Copies part of a tag database tree to another place. This means a unit or network definiton, for example, can be copied to create a new entry. The source object is not restricted to unit or network, e.g. applicatoin or group object can also be copied.

You can copy parts of databases between projects by using //*project-name*/ as address prefixes

Notes:

The destination part of the tree has new object identifiers (OIDs) generated for all elements that have OIDs.

All fields of the source object are copied to the destination object, including Address and TagName fields.

There are some business rules check before copying the object and the rules are:
- the destination object's parent node must have the same type as the source object's parent node
- if element-type is Network, Unit, Application, Group or Level, destination-element-address must be the string representation of decimal number and its value must be within the range of [0, 255]
- the destination-element-address can not be the same as the address of an existing object under destination-parent-address if it is provided
- the destination-tag-name can not be the same as the tag name of an existing object under destination-parent-address if it is provided

Examples

```
# copy the group adressed as 0 in application 56 under network 254 to
the same application as address 1 with tag name Group-1
DBCOPYSAFE 254/56/0 254/56 1 Group-1

# copy the unit at address 10 in network 254 from project jim to network
1 on project system
DBCOPYSAFE //jim/254/p/10 //system/1
```

Successful Response

```
200 OK.
```
(where no OID is used for this element), or

```
301 OID=oid-value
```
(where an OID is applicable to the element that has been created)

Failure response

```
400 Syntax Error.
401 Bad object or device ID: Bad source address: <exception info>
401 Bad object or device ID: Bad destination address: <exception info>
401 Bad object or device ID: Can't get source or destination parent element
401 Bad object or device ID: Destination Address is not unique.
401 Bad object or device ID: Destination address is required for this object when
copying under the same node
401 Bad object or device ID: Destination tag name is required for this object when
copying under the same node
401 Bad object or device ID: Element address must be a number
401 Bad object or device ID: Element address must be in the range of 0 to 255
401 Bad object or device ID: Element address in use
401 Bad object or device ID: Element tagname in use
401 Bad object or device ID: Source Address is not unique.
408 Operation failed: reason
```



```
440 There is no tag database to perform this operation on.
```

4.5.58 DBCREATE

Command

```
DBCREATE
```

Access Level

Admin

Syntax

```
DBCREATE
```

Use

Generates a tag database that represents the current connected C-Bus networks & C-Gate implementation. This replaces the current tag database.

So, to capture the state of the network and store it, perform a DBCREATE then a DBSAVE command to save the database to disk.

This command creates a new database, which fills in compulsory elements with the string `[default]` where no value can be determined from the network or system.

Examples

```
dbcreate
200 OK.
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error.
443 Error creating tag database
```

See also

DBNEW

4.5.59 DBCREATENET

Command

```
DBCREATENET
```

Access Level

Admin

Syntax

```
DBCREATENET address name type interface-address

address = address of the new network (0..255)
name = name of the new network
type = type of the network interface (Serial | Cni | Bridge)
interface-address = interface details
```

Use

Creates a new network in the database. This command is equivalent to the following sequence of commands:

dbadd Installation/Project Network
dbset <oid>/NetworkNumber "address"
dbset <oid>/Address "address"
dbset <oid>/TagName "name"
dbadd <oid> Interface
dbset <oid2>/InterfaceType "type"
dbset <oid2>/InterfaceAddress "interface-address"

Note that this command on its own does not persist the new object to disk. You need to follow this with a PROJECT SAVE.

Examples

```
dbcreatenet 254 "My Network 1" Serial com1
200 OK.

dbcreatenet 251 "My Network 2" Cni 192.168.0.1:14000
200 OK.

dbcreatenet 253 "My Network 3" Bridge 254/p/253
200 OK.
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error.
440 There is no tag database to perform this operation on.    443 Error creating
tag database
```

4.5.60 DBDELETE Command

```
DBDELETE
```


Access Level

Operate

Syntax

```
DBDELETE element-address
```

Use

Deletes an element from the current tag database.

Examples

```
# remove group 1 from lighting on network 1
DBDELETE 1/56/1
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error.
401 Bad object or device ID:
440 There is no tag database to perform this operation on.
```

4.5.61 DBGET

Command

DBGET

Access Level

Monitor

Syntax

```
DBGET parameter-address

parameter-address = Address of a tag database element or a C-Gate object address
```

Use

Gets the value of an element at an address. If the address points to an element with one or more sub-elements, the sub-elements are displayed.

Examples

```
DBGET Installation/Version

DBGET Installation/Project/Network[1]/TagName
```


Successful Response

`342 parameter-address=value` (there may be multiple responses)

Failure response

```
400 Syntax Error
401 Bad object or device ID:
440 There is no tag database to perform this operation on.
```

4.5.62 DBGETXML

Command

DBGETXML

Access Level

Monitor

Syntax

```
DBGETXML db-address
```

Use

Gets a part of the current Tag Database as a snippet of XML.

The db-address given is any object in the XML database, and can be specified as either a path from the base of the tag database (ie /Installation/InstallationDetail/Installer), or a C-Bus network address, of a tag name.

This command returns:

- a 343-Begin XML snippet response
- The XML snippet requested, as one or more lines starting with "347-". This starts with a <?xml … ?> line. Note that the XML is not pretty-printed.
- a 344 End XML snippet response

If a single element is selected by address, for example an element that simply contains a string, then an XML snippet similar to the following will be returned just giving the raw type of the field:

```
347-<?xml version="1.0"?>
347-<string>2001-11-01T13:05:34.153</string>
```

Examples

```
DBGETXML Installation/Version
DBGETXML p/1/20
```

Successful Response (example)


```
343-Begin XML snippet
347-<?xml version="1.0"?>
347-<Unit xmlns="http://www.clipsal.com/cis/schema/2001/cbus.xsd">
347-<TagName>n1_u20</TagName><Address>20</Address><UnitType>KEYM8</
UnitType><UnitName>NEWUNIT </UnitName><FirmwareVersion>1.3.06</
FirmwareVersion></Unit>
344 End XML snippet
```

Failure response

```
401 Bad object or device ID
440 There is no tag database to perform this operation on.
444 XML creation failed
```

See also

DBSETXML

4.5.63 DBLOAD

Command

DBLOAD

Access Level

Operate

Syntax

```
DBLOAD filename
```

Use

In normal use, the PROJECT LOAD command should be used instead of DBLOAD to open project databases.

Loads a project database into C-Gate in the current project.

This makes the database the active project database for the current project.

If and only if the filename given has an extension of ".zip", C-Gate will attempt to uncompress a ZIP archive containing the XML tag database. If the extension ".gz" is given, C-Gate will attempt to use GZIP to uncompress the file before reading. If any other extension is given, or no extension is given, then the file is assumed and expected to be in an XML form, though C-Gate does *not* assume a .XML extension.

If the filename is an absolute path, then the file will be loaded from that absolute filename. If a relative path is given, then the path will be relative to the path given in the `tag-base-directory` property.

Examples


```
DBLOAD myproj.xml DBLOAD project.zip
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error.
442 Error reading tag database: details
```

See also

DBSAVE

4.5.64 DBNETWORKPATH Command

DBNETWORKPATH

Access Level

Admin

Syntax

```
DBNETWORKPATH start-net-address end-net-address [OID | COMPACT]

start-net-address = the starting network address in the database
end-net-address = the ending network address in the database
```

Use

Finds a network path between two networks representation in the project database, via bridge devices.

It is assumed that the standard bridge and network addresssing convention is used: the unit address of a bridge device is the same as the network address on the far side of the bridge device. A path longer than 6 bridges will not be returned.

With the OID option given, a series of OIDs are given for all the networks in the path, including the ending network, but not the source network in one or more 137 responses.

With the COMPACT option given, a compact bridge path hex string is returned in a single 136 response.

The OID option is the default if no option is given

Examples

```
#OID form
dbnetworkpath 254 252 oid
```



```
137-47951260-07e6-1027-ae52-f53240c219d9
137 4795d5b0-07e6-1027-ae56-f53240c219d9

#compact form
dbnetworkpath 254 252 compact
136 FDFC
```

Successful Response

Either:

```
"136" hex-bytes
```

or one or more lines of:

```
"137" oid
```

Failure response

```
400 Syntax Error.
401 Bad object or device ID: reason
408 Operation failed: reason
440 There is no tag database to perform this operation on.
```

4.5.65 DBNEW

Command

DBNEW

Access Level

Admin

Syntax

```
DBNEW
```

Use

Clears the tag database for the currrent project and creates a new blank database.

This blank database contains nothing more than an Installation element, so additional information has be to added before the database is valid to the tag database specification.

Examples

```
dbnew
200 OK.
```

Successful Response

```
200 OK.
```


Failure Response

```
408 Project not found
```

4.5.66 DBRENAMENET Command

DBRENAMENET

Access Level

Admin

Syntax

```
DBRENAMENET db-net-address new-net-address

db-net-address = the database address of a <Network> stored in the project
database.
new-net-address = the parent element to hold the copied part of the database
```

Use

Re-addresses a <Network> in the project database, and additionally ensures that and other <Networks> that are of the bridge type have any references to the old network corrected to support the new network. Re-addressing means that the network's <Address> element is given a new value.

Warning: This command does not prevent you from re-addressing a network to be the same address as another existing network. This could be confusing if used with reckless abandon.

Examples

```
# re-address network 1 to be network 2
DBRENAMENET 1 2
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error.
401 Bad object or device ID: reason
408 Operation failed: reason
440 There is no tag database to perform this operation on.
```

4.5.67 DBRENAMENETSAFE Command

DBRENAMENETSAFE

Access Level

Admin

Syntax

```
DBRENAMENETSAFE db-net-address new-net-address

db-net-address = the database address of a <Network> stored in the project
database.
new-net-address = the parent element to hold the copied part of the database
```

Use

Re-addresses a <Network> in the project database, and additionally ensures that and other <Networks> that are of the bridge type have any references to the old network corrected to support the new network. Re-addressing means that the network's <Address> element is given a new value.

There are some business rules check before renaming the network and the ruels are:

- db-net-address and new-net-address must be string representations of decimal number (i.e. only 0~9 is allowed)
- new-net-address can not be the same as db-net-address
- new-net-address must be within the range of [0, 255]
- new-net-address can not be the same as the address of an existing network in the same project

Examples

```
# re-address network 1 to be network 2
DBRENAMENETSAFE 1 2
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error.
401 Bad object or device ID: Address is not unique
401 Bad object or device ID: Can't rename network to the same address
401 Bad object or device ID: Element address must be a number
401 Bad object or device ID: Element address must be in the range of 0 to 255
401 Bad object or device ID: Get project database entry failed
401 Bad object or device ID: Invalid network address
401 Bad object or device ID: New network address in use
401 Bad object or device ID: No network given
401 Bad object or device ID: No new address given
401 Bad object or device ID: No project in database
401 Bad object or device ID: Not a network database entry
408 Operation failed: reason
440 There is no tag database to perform this operation on.
```


4.5.68 DBSAVE

Command

DBSAVE

Access Level

Admin

Syntax

```
DBSAVE filename
```

Use

Saves the current Tag database to the file or URL specified. The current tag database will be saved in the given filename.

The filename extension determines how the file will be saved:

- If the extension given is **.zip** the tag database is saved in a zip archive. The tag database is saved in the archive as an entry called 'tagdb.xml' and is the only entry in the archive.
- If the extension given is .gz the tag database will be compressed using GZIP.

Otherwise, the file is saved in XML format. The files will only be saved with an extension of **.xml** if this is given in the filename.

Examples

```
DBSAVE project.xml
DBSAVE compressed-project.zip
```

Successful Response

```
200 OK.
```

Failure response

```
440 There is no tag database to perform this operation on.
441 Error writing tag database
```

See also

DBLOAD, PROJECT SAVE

4.5.69 DBSET

Command

DBSET

Syntax

```
DBSET parameter-address value
```



```
parameter-address = address of a tag database element or a C-Gate object address
value = the value to set the parameter to. This includes all characters to the end
of the line.
```

Use

Sets a tag database field by address.

Examples

```
DBSET Installation/Version 2.01
200 OK.
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error.
401 Bad object or device ID:
440 There is no tag database to perform this operation on.
446 Unable to set:
```

4.5.70 DBSETSAFE

Command

DBSETSAFE

Syntax

```
DBSET parameter-address value

parameter-address = address of a tag database element or a C-Gate object address
value = the value to set the parameter to. This includes all characters to the end
of the line.
```

Use

Sets a tag database field by address and apply business rules check.
The rules are:

- if the field to be set is Address and the object to be set is Network, Unit, Application, Group or Level, the value must be the string representation of decimal number (i.e. only 0~9 is allowed) and the value must be within the range of [0, 255]
- if the field to be set is Address, the value can not be the same as the address of an existing object under the same parent
- if the field to be set is TagName, the value can not be blank
- if the field to be set is TagName, the value can not be the same as the tag name of an existing object under the same parent

Examples


```
DBSETSAFE Installation/Version 2.01
200 OK.
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error.
401 Bad object or device ID: Address is not unique. Unique address required for
dbset operation
401 Bad object or device ID: Bad object address
401 Bad object or device ID: Element address must be a number
401 Bad object or device ID: Element address must be in the range of 0 to 255
401 Bad object or device ID: Element address in use
401 Bad object or device ID: Element tagname in use
401 Bad object or device ID: Field not available
401 Bad object or device ID: Field not found
401 Bad object or device ID: TagName can't be null or blank
401 Bad object or device ID: OID field can not be changed
401 Bad object or device ID: <exception info>
440 There is no tag database to perform this operation on.
446 Unable to set:
```

4.5.71 DBSETXML

Command

DBSETXML

Access Level

Admin

Syntax

```
DBSETXML db-address << end-tag
```

Use

Sets a part of the current Tag Database as a snippet of XML given after the command issued. The db-address given is any object in the XML database, and can be specified as either a path from the base of the tag database (ie /Installation/InstallationDetail/Installer), a C-Bus network address, a tag name or OID.

The XML required for this command is provided as a here document, meaning is it provided in one or more lines following the command, with the command terminated by the `end-tag` given on the command lien after the << characters, on a line by itself.

This command returns:

- a 301 OID= message if the XML setting succeeds. The OID returned is the OID given to the newly-created element that results from the XML setting operation on the database.
- one or more errors if other conditions occur.

Examples

```
DBGETXML Installation/Version << EOF
.. one or more lines of XML data ...
EOF
```

Successful Response

```
301 OID=oid-value
```
This response returns the OID of the created element

Failure response

```
401 Bad object or device ID:
440 There is no tag database to perform this operation on.
```

See also

DBGETXML

4.5.72 DBTAGLIST

Command

DBTAGLIST

Access Level

Admin

Syntax

```
DBTAGLIST [pattern]

pattern = substring to match the tagnames against
If not given, all tag names are listed.
```

Use

Lists tag names and their associated objects.

Examples

```
DBTAGLIST

DBTAGLIST lighting
```

Successful Response

one or more lines of tag names ...
```
eg:
342-254/203/1/TagName=Group 1
342 254/56/TagName=Lighting
```


Failure response

```
401 Bad object or device ID:
420 Access denied.
440 There is no tag database to perform this operation on.
```

4.5.73 DBUPDATE

Command

DBUPDATE

Access Level

Admin

Syntax

```
DBUPDATE network-address [unit-delete]

network-address = a C-Gate or Tag database address that resolves to a
network or a unit
unit-delete = If this parameter is provided and set to the word
"UnitDelete", then any units that are present in the database but are
not present in the phyisical network will be deleted from the database.
```

Use

Updates the current database with the current physical network or unit configuration. This overwrites the current networks.

Examples

```
# update network 1
DBUPDATE 1

# update network n1
DBUPDATE n1

#update unit p/1/22
DBUPDATE p/1/22
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error.
401 Bad object or device ID:
408 Operation failed:
440 There is no tag database to perform this operation on.
```


4.5.74 DBVALIDATE

Command

DBVALIDATE

Access Level

Operate

Syntax

```
DBVALIDATE db-address
```

Use

Validates the section of the current tag database given by the address, checking that it conforms to the tag specification. This validations check that required fields are filled in correctly and that there are no violations of the tag specification.

Examples

```
dbset 1/TagName Jim
200 OK.
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error.
440 There is no tag database to perform this operation on.
```

4.5.75 DBVERIFY

Command

DBVERIFY

Access Level

Admin

Syntax

```
DBVERIFY
```

Use

Verifies the current network against the current database and reports differences. DBVERIFY compares the live networks that C-Gate is connected with and entries in the tag database in order to detect differences or omissions. These are reported as a series of 345 responses.

Examples

```
dbverify
200 OK.
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error.
440 There is no tag database to perform this operation on.
```
Zero of more lines of:
```
345 Difference: difference-details
408 Operation failed: (Verify failed: <count> differences)
```

4.5.76 DO

Command

DO

Access Level

Operate

Syntax

```
DO object-identifier method-name *(method-parameter)

method-parameter = token
```

Use

Calls the methods of an object.

A `202 Done` response is returned.

The method may return one or more lines of 120 response that are information relating to this execution of the method.

Note that the ON, OFF and RAMP commands are shorthand versions of DO commands with the methods on off and ramp.

Examples

```
do 1/56/1 on
202 Done: //HOME/1/56/1
```

Successful Response

```
"202 Done:" object-identifier
```


Also may include one or more lines of 120 responses giving additional method information.

Failure response

```
400, 401, 402, 405, 408, 420
```

See also

GET or SHOW, SET, ON, OFF, RAMP

4.5.77 ENABLE

Command

ENABLE

Access Level

Operate

Syntax

```
ENABLE [?]
```

Use

Lists the ENABLE sub-commands:

```
ENABLE LABEL - Set labels on devices that support dynamic labels
ENABLE REMOVE - Remove any reference to this network variable
ENABLE SET - Set an enable variable to the given value
```

Success response

A series of lines giving help for these commands in the form: 101 Help: help-information

Failure responses

4.5.78 ENABLE LABEL

Command

ENABLE LABEL

Access Level

Operate

Syntax

```
ENABLE LABEL app language group-number action-sel [variant] options

app = application address (normally //proj/net/$CB)
language =  language code (0..255)
```



```
group-number = relevant C-Bus group number (0..255)
action-sel = action selector | "-"  (dash indicates unset)
variant = F0 | F1 | F2 | F3 (sets the variant for this label)

options = ( "text" text-label ) |
          ( "icon" icon-selector ) |
          ( "dynamic" icon-selector icon-width icon-height vertical-offset
dynamic-icon ) |
          ( "set_language") |
          options-byte hex-bytes

text-label = ASCII text to end of line to be used as label
icon-selector = numeric icon selection in the range 0 through 65535
icon-width = width of the icon in pixels
icon-height = height of the icon in pixels

dynamic-icon = block of ASCII-encoded bytes representing the icon, as 16 lines of
up to 62 bits, encoded as up to
16 characters of HEX, with each block separated by a colon character.

If a block is less than 16 characters, the block will be zero-filled on the right.
For example:

AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:
AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:
AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:
AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644

options-byte = integer in the range of 0 through 255, used as the value of the
options 1 byte in the command.

hex-bytes = a string of hex bytes to follow the command

Note: all numeric values except for dynamic-icon can be entered as decimal, 0x or
$ prefixed hex, or 0b prefixed binary.
```

Use

Performs label setting with devices that support dynamic labels.

Example

```
#set a text label
enable label 1/56 1 2 3 text hello
200 OK.

#set an existing icon
enable label 1/56 1 2 - icon $1234
200 OK.

#set a dynamic icon
enable label 1/56 1 2 - dynamic
0102030405060708:0102030405060708:0102030405060708:0102030405060708:01020304050607
08:0102030405060708:0102030405060708:0102030405060708:0102030405060708:01020304050
60708:0102030405060708:0102030405060708:0102030405060708:0102030405060708:01020304
05060708:0102030405060708
```



```
200 OK.

# change language
enable label 1/56 1 2 - set_language
200 OK.

# send a raw command
enable label 1/56 1 2 - 6 3F2B0102
200 OK.
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.79 ENABLE REMOVE

Command

ENABLE REMOVE

Access Level

Operate

Syntax

```
ENABLE REMOVE enable-netvar-address
```

Use

Removes any memory that the server has of this enable network variable. This means that the network variable is removed, and any on-disk storage of the value of the network variable is also removed.

Example

```
ENABLE REMOVE 2/$cb/1
200 OK.
```

Success response

```
200 OK
```

Failure responses


```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.80 ENABLE SET

Command

ENABLE SET

Access Level

Operate

Syntax

```
ENABLE SET enable-netvar-address value ["force"]

value = integer in the range 0..255 | percentage value (0..100) followed by the %
sign | level tag
```

Use

Sends an *enable set* message to the network to set the enable network variable given as *enable-netvar-address,* to the *value* given. This is equivalent to a LIGHTING RAMP immediate command for the lighting application.

Value is either: an integer in the range 0..255; or a percentage value (0..100) followed by the % sign; or a level tag.

Specifying the optional parameter *force*, will allow the command to be sent regardless of network state.

Success response

```
200 OK
```

Failure responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.81 EREPORT

Command

EREPORT

Access Level

OPERATE

Syntax

```
EREPORT [?]
```

Use

Gives an overview of the EREPORT sub-commands:
```
EREPORT MESSAGE  - Send an error reporting event with choice of headers
```

Success response

A series of lines giving help for these commands in the form:

```
101 Help: help-information
```

Failure responses

A functional copy of C-Gate will always respond to this command

4.5.82 EREPORT MESSAGE Command

EREPORT MESSAGE

Access Level

Operate

Syntax

```
EREPORT MESSAGE app type category most-recent acknowledged most-severe severity
unit-id [<data-byte-1 [data-byte-2]]

app = application address (net/$CE)
type = type of message (RECENT | ERROR_REPORT | ACK | CLEAR | 0..255)
category = error category number (0..1023)
most-recent = true if the error is the most recent (y | n | 1 | 0)
acknowledged = true when the error has been acknowledged (y | n | 1 | 0)
most-severe = true if the error is the most severe to date (y | n | 1 | 0)
severity = severity level (0..4)
unit-id = address of the error monitoring device (0..255)
data-byte-1 = optional byte containing category specific meaning (0..255)
data-byte-2 = optional byte containing category specific meaning (0..255)
```

- `severity level: 0 = ALL_OK; 1 = OK; 2 = minor failure; 3 = general failure; 4 = extreme failure`

Use

Sends an error report to the network using the Error Reporting Application.

Note that EREPORT MESSAGE does not yet implement any of the error-reporting constraints enforced by some of the devices - it will allow you to send messages that are syntactically incorrect. This allows it to be more flexible during the transition to a new error reporting protocol.

Messages received with an $05 or $15 header will continue to produce the old status change message, unchanged. However, they will no longer produce a level 8 event message.

Examples

```
ereport message 254/$CE ACK 1023 n n n 7 255 255 255
```

Successful Response

```
200 OK
```

Failure responses

```
400, 401, 402, 405, 408, 420
```

4.5.83 EVENT
Command

EVENT

Access Level

Monitor

Syntax

```
EVENT ON | OFF | event-mode

event-mode = "e" (IDIGIT | '+' ) "s" DIGIT "c" DIGIT
```

Use

Controls the output of events, status change and config change information to the command session that the command is executed in.

EVENT ON will turn on the output of events only with no prefix on each event. This is the same as event-mode `e+s0c0`.

EVENT OFF disables event output. This is equivalent to setting an event-mode of `e0s0c0`.

(Can also use the command EVENTS.)

Event Modes:

Giving an event-mode expression can give find grained control of event output. An even mode is made up of a set of identifiers and an event level for each identifier.

For events, the identifer is: `e` and valid event levels are `+` and `0` through `9`. In this case, the event level indicates the maximum level of events that will be displayed. When these events are delivered, they show a prefix of `#e#`. The '+' level is a special code to indicate that no prefix is to be used and the default event level of C-Gate object are used.

For status change events, the identifier is `s` and valid events levels are `0` and `1` corresponding to disabled and enabled. When these events are delivered, they are prefixed with `#s#`.

For config change events, the identifier is `c` and valid events levels are `0` and `1` corresponding to disabled and enabled.When these events are delivered, they are prefixed with `#c#`.

Console

The console is opened in event mode `e+s0c0`. This can also be changed by command.

Examples

```
# turn on events in this session
EVENT ON

#enable scp, ccp and events at level 5
events e5s1c1

# turn off events
EVENTS OFF

# return current event mode
events
306 e5s1c1
```

Successful Response

```
200 OK.
```
or
```
306 event-mode
```

4.5.84 GET

Command

GET

Access Level

Monitor

Syntax

```
GET object-identifier [parameter | ? | * | ??]
```

Use

Retrieves the values of objects. Object parameters may be viewed by naming them in the

command.

The special parameter '?' returns a list of all parameters of this object. The special parameter * returns a list of all parameter values for an object. The special parameter ?? returns one line for each parameter with a desciption of the parameter.

(Can also use the SHOW command.)

Examples

```
show cbus networks
get //proj/level57/ units
get //proj/level57/$38/1 level
get //proj/level57/2/1 power
get //proj/level57/2 type
get //proj/level57/2 ?
```

Successful Response

```
One or more lines of the following 3xx series responses:

response-code [-] object-identifier ":" parameter-name=parameter-value

For example: 300 level57/56/1: level=200
```

Failure response

```
400, 401, 402, 408, 420
```

See also

SET, DO

4.5.85 GETSTATE

Command

GETSTATE

Access Level

Monitor

Syntax

```
GETSTATE (network-address | cgroup-name)
```

Use

Causes a series of events to be sent to the event stream, that gives the *current* status of a connected network. Events are returned for all devices in the network. Events are returned giving:

- status information for all units and groups
- level information for all groups and terminals

This command is an easy way for an application to get the current state of the network as a starting point for a model or for subsequent event tracking or polling.

Examples

```
GETSTATE 57
200 OK.
```

Successful Response

```
200 OK
```

Failure response

```
400, 401, 420
```

See also

GET, TREE

4.5.86 HELP

Command

HELP

Syntax

```
HELP [command-or-topic]
```

Access Level

Monitor

Use

Gives brief command-line help. Note that the help command will only give help for commands that are accessible by the access level of the current user.

`HELP *` returns a list of all the commands.

? can be used as an abbreviation of `HELP`.

Examples

```
help set
101-Help: syntax: SET <object-id> <param-name> <value>
101-Help: Set the parameter given in <param-name> to the value in
101-Help: <value> for the object <object-id>.
101-Help: <object-id> is a network, group, unit, or system entity
```



```
101-Help: <param-name> is a named parameter.
101 Help: <value> is the value to set the parameter to.
```

Successful Response

One or more lines in the following form:

```
101 [-] help-information
```

Failure response

```
400, 420
```

4.5.87 LIGHTING Command

LIGHTING

Access Level

Operate

Syntax

```
LIGHTING [?]
```

Use

Lists the LIGHTING sub-commands:
`LIGHTING LABEL` - Sets labels on devices that support dynamic labels.
`LIGHTING OFF` - Turns off a specified group.
`LIGHTING ON` - Turns on a specified group.
`LIGHTING RAMP` - Ramps a group to a % level using a specified ramp time.
`LIGHTING TERMINATERAMP` - Stops any ramping operation for a given group.

Success Response

A series of lines giving help for these commands in the form:
`101 Help: help-information`

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.88 LIGHTING LABEL Command

LIGHTING LABEL

Access Level

Operate

Syntax

```
LIGHTING LABEL app language group-number action-sel [variant] options

app = application address (normally //proj/net/$38)

language =  language code (range 0 through 255)

group-number = relevant C-Bus group number (range 0 through 255)

action-sel = action selector | "-"  (dash indicates unset)

variant = F0 | F1 | F2 | F3 (sets the variant for this label)

options = ( "text" text-label ) |
          ( "icon" icon-selector ) |
          ( "dynamic" icon-selector icon-width icon-height vertical-offset
dynamic-icon ) |
          ( "set_language") |
          options-byte hex-bytes

text-label = ASCII text to end of line to be used as label, and its maximum length
is 14 characters

icon-selector = numeric icon selection in the range 0 through 65535

icon-width = width of the icon in pixels

icon-height = height of the icon in pixels

dynamic-icon = block of ASCII-encoded bytes representing the icon, as 16 lines of
up to 62 bits, encoded as up to
16 characters of HEX, with each block separated by a colon character.

If a block is less than 16 characters, the block will be zero-filled on the right.
For example:

AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:
AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:
AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:
AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644

options-byte = ; integer in the range of 0 through 255, used as the value of the
options 1 byte in the command.

hex-bytes = ; a string of hex bytes to follow the command, and its maximum length
is 14 characters (i.e. 28 hex byte characters in the command)

Note: all numeric values except for dynamic icon can be entered as decimal, 0x or
```



```
$ prefixed hex, or 0b prefixed binary.
```

Use

Sets labels on devices that support dynamic labels.

Note: For devices that support unicode dynamic labels, if a group/variant already has a unicode label it will not accept text labels sent with this command. To set or clear unicode labels see the LIGHTING UNICODELABEL command.

Examples

```
#set a text label
lighting label 254/56 1 2 3 text hello
200 OK.

#set an existing icon
lighting label 254/56 1 2 - icon $1234
200 OK.

#set a dynamic icon
lighting label //proj/1/56 1 2 - dynamic
0102030405060708:0102030405060708:0102030405060708:0102030405060708:01020304050607
08:0102030405060708:0102030405060708:0102030405060708:0102030405060708:01020304050
60708:0102030405060708:0102030405060708:0102030405060708:0102030405060708:01020304
05060708:0102030405060708
200 OK.

# change language
lighting label 254/56 1 2 - set_language
200 OK.

# send a raw command
lighting label 254/56 1 2 - 6 3F2B0102
200 OK.
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.89 LIGHTING UNICODELABEL Command

LIGHTING UNICODELABEL

Access Level

Operate

Syntax

```
LIGHTING UNICODELABEL app language group-number action-sel variant option text-
label|hex-bytes

app = application address (normally //proj/net/$38)

language =  language code (range 0 through 255)

group-number = relevant C-Bus group number (range 0 through 255)

action-sel = action selector | "-"  (dash indicates unset) (should always be dash)

variant = F0 | F1 | F2 | F3 (sets the variant for this label)

option = ( "text" text-label ) |
         ( "raw" hex bytes string encoded in UTF-8 )

text-label = ASCII text to end of line to be used as label if option = text

hex-bytes =  a string of hex bytes to follow the command if option = raw
```

Use

Sets unicode labels on devices that support unicode dynamic labels. The label string should be encoded in UTF-8.

To clear a unicode label simply provide no parameter for text-label or hex-bytes.

Note: Please use the raw option to send unicode characters to C-Gate. Using the text option to send unicode may not always work.

Examples

```
#set a text label
lighting unicodelabel 254/56 1 2 - F0 text hello
200 OK.

#set a raw hex bytes string
lighting unicodelabel 254/56 1 2 - F0 raw 414243444546
200 OK.
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
```



```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.90 LIGHTING OFF

See OFF

4.5.91 LIGHTING ON

See ON

4.5.92 LIGHTING RAMP

See RAMP

4.5.93 LIGHTING TERMINATERAMP

See TERMINATERAMP

4.5.94 LOCK

Command

LOCK

Access Level

Operate

Syntax

```
LOCK object-identifier
```

Use

Sets an advisory lock on an object. This lock is typically used by cooperating applications that want exclusive access to a device or object for programming of configuration. A lock survives until unlocked with the UNLOCK command or the command session ends.

Note that the locking is advisory only. Applications must cooperate to make a locking system work.

Examples

```
LOCK p/254/2
225 //HOME/254/p/2: Locked.
```

Successful Response

```
225 <object-identifier>: Locked
```

Failure response

```
400, 401, 420, 425
```

See also

UNLOCK

4.5.95 LOGIN

Command

LOGIN

Access Level

Connect

Syntax

```
LOGIN [username password]

username = user name
password = user password
```

Use

Changes the current access level of this command session, by providing a username and a password.

If no username and password are given, the command returns the current access level.

Usernames and passwords are case-sensitive.

Examples

```
LOGIN

LOGIN james aabbc
```

Successful Response

```
210 Access level: level 211
210 Access level set to: level
```

Failure response

```
400, 422
```

4.5.96 LOGOUT

Command

LOGOUT

Access Level

Connect

Syntax


```
LOGOUT
```

Use

Logs out of the current access level of this command session.

If a user hasn't logged in, just reports the current access level.

Examples

```
LOGOUT
```

Successful Response

```
211 Access level set to: level
```

Failure response

```
(Always succeeds.)
```

4.5.97 MEASUREMENT

Command

MEASUREMENT

Access Level

Operate

Syntax

```
MEASUREMENT [?]
```

Use

Lists the MEASUREMENT sub-commands:
`MEASUREMENT DATA - Measurement data for a Channel.`

Success Response

A series of 101 Help lines giving a list of the MEASUREMENT commands.

Failure Responses

4.5.98 MEASUREMENT DATA

Command

MEASUREMENT DATA

Access Level

Operate

Syntax

```
MEASUREMENT DATA channel value multiplier units

channel = channel address (eg //proj/net/$E4/1/1)
value = scaled measurement value (-32768..32767)
multiplier = power of ten unit multiplier (-128..127)
units = measurement units (0..$FF)

For valid units values see the Unit Code Table.
```

Use

Measurement Data Messages are sent by a Measurement Device either:
(a) based on an elapsed time interval,
(b) based on a change in the measured value, or
(c) in response to a specific Control Trigger application request

Examples

```
MEASUREMENT DATA 254/$E4/1/2 37 1 0
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
401
402
405
408
420
```

4.5.99 MEDIATRANSPORT Command

MEDIATRANSPORT

Access Level

Operate

Syntax

```
MEDIATRANSPORT [?]
```

Use

Lists the MEDIATRANSPORT sub-commands:

```
MEDIATRANSPORT CATEGORY_NAME - Sets the category name.
MEDIATRANSPORT ENUMERATE - Request up to 16 names for a category, selection or
track.
MEDIATRANSPORT ENUMERATION_SIZE - Indicates the number of items in an enumeration.
MEDIATRANSPORT FORWARD - Fast-forwards the current track.
MEDIATRANSPORT NEXT_CATEGORY - Changes to the next category.
MEDIATRANSPORT NEXT_SELECTION - Changes to the next selection.
MEDIATRANSPORT NEXT_TRACK - Changes to the next track.
MEDIATRANSPORT PAUSE - Pauses/unpauses playing.
MEDIATRANSPORT PLAY - Starts playing.
MEDIATRANSPORT REPEAT - Sets state to 'repeat'.
MEDIATRANSPORT REWIND - Rewinds the current track.
MEDIATRANSPORT SELECTION_NAME - Sets a selection name.
MEDIATRANSPORT SET_CATEGORY - Chooses a category.
MEDIATRANSPORT SET_SELECTION - Chooses a selection.
MEDIATRANSPORT SET_TRACK - Chooses a track.
MEDIATRANSPORT SHUFFLE - Sets state to 'shuffle'.
MEDIATRANSPORT SOURCE_POWER - Turns on/off the output unit.
MEDIATRANSPORT STATUS_REQUEST - Requests current status of an output unit.
MEDIATRANSPORT STOP - Stops play.
MEDIATRANSPORT TOTAL_TRACKS - Indicates how many tracks are available
MEDIATRANSPORT TRACK_NAME - Sets the track name.
```

Success Response

A series of 101 Help lines giving a list of the MEDIATRANSPORT commands.

Failure Responses

4.5.100 MEDIATRANSPORT CATEGORY_NAME Command

MEDIATRANSPORT CATEGORY_NAME

Access Level

Operate

Syntax

```
MEDIATRANSPORT CATEGORY_NAME app group wni total index text

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
wni = name identification (0..7)
   0 = current category
   1 = next category
   2 = next+1 category
   3,4 = reserved
   5 = previous category
   6 = previous-1 category
```



```
   7 = enumerated category
total = total number of packets - 1 (0..3)
index = index/sequence of this packet (0..3)
text = all/part of the category name in UTF-8 (max 11 bytes per packet)
```

Use

Sets the name of a *category* in the output unit in the given *media link group*.
The name is constructed from one or more packets, sent in sequence, coded as UTF-8, limited to 40 bytes total length.

Examples

```
MEDIATRANSPORT CATEGORY_NAME 254/$C0 2 1 1 0 iPod
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.101 MEDIATRANSPORT ENUMERATE Command

MEDIATRANSPORT ENUMERATE

Access Level

Operate

Syntax

```
MEDIATRANSPORT ENUMERATE app group type start

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
type = enumeration type where 0=category, 1=selection, 2=track
start = enumerate from this index (0..$FF)
```

Use

Requests a dump of the names of the first 16 categories, selections or tracks in the given *media link group*.

Examples

```
MEDIATRANSPORT ENUMERATE 254/$C0 2 2 0
```

Successful Response


```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.102 MEDIATRANSPORT ENUMERATION_SIZE

Command

MEDIATRANSPORT ENUMERATION_SIZE

Access Level

Operate

Syntax

```
MEDIATRANSPORT ENUMERATION_SIZE app group type start size

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
type = enumeration type where 0=category, 1=selection, 2=track
start = enumerate from this index (0..$FF)
size = number of items in the enumeration following (0..$0F)
```

Use

Describes the number of entries the unit is about to send in response to an enumeration request on the given *media link group*.

Examples

```
MEDIATRANSPORT ENUMERATION_SIZE 254/$C0 2 2 0 6
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.103 MEDIATRANSPORT FORWARD

Command

MEDIATRANSPORT FORWARD

Access Level

Operate

Syntax


```
MEDIATRANSPORT FORWARD app group operation

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0..$FF):
   0 = cease fast-forward, play at normal speed
   2 = fast-forward at 2x speed
   4 = fast-forward at 4x speed
   6 = fast-forward at 8x speed
   8 = fast-forward at 16x speed
  10 = fast-forward at 32x speed
  12 = fast-forward at 64x speed
  all other values reserved
```

Use

Requests that the *track* currently being played in the given *media link group* be fast-forwarded. Continues to be in effect until a PLAY, FORWARD 0 or REWIND command is received.
If the output unit is not in the *play* state, the command is ignored.

Examples

```
MEDIATRANSPORT FORWARD 254/$C0 2 1 2
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
401
402
405
408
420
```

4.5.104 MEDIATRANSPORT NEXT_CATEGORY

Command

MEDIATRANSPORT NEXT_CATEGORY

Access Level

Operate

Syntax

```
MEDIATRANSPORT NEXT_CATEGORY app group operation

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0..$FF):
```



```
   0 = select previous category
   non-0 = select next category
```

Use

Requests that the *output unit* in the given *media link group* change to the next or previous *category*.

Examples

```
MEDIATRANSPORT NEXT_CATEGORY 254/$C0 2 1
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.105 MEDIATRANSPORT NEXT_SELECTION

Command

MEDIATRANSPORT NEXT_SELECTION

Access Level

Operate

Syntax

```
MEDIATRANSPORT NEXT_SELECTION app group operation

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0..$FF):
   0 = select previous selection
   non-0 = select next selection
```

Use

Requests that the *output unit* in the given *media link group* change to the next or previous *selection*.

Examples

```
MEDIATRANSPORT NEXT_SELECTION 254/$C0 2 1
```

Successful Response

```
200 OK.
```

Failure response


```
400 Syntax Error
```

4.5.106 MEDIATRANSPORT NEXT_TRACK Command

MEDIATRANSPORT NEXT_TRACK

Access Level

Operate

Syntax

```
MEDIATRANSPORT NEXT_TRACK app group operation

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0..$FF):
   0 = select previous track
   non-0 = select next track
```

Use

Requests that the *output unit* in the given *media link group* change to the next or previous *track*.

Examples

```
MEDIATRANSPORT NEXT_TRACK 254/$C0 2 1
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.107 MEDIATRANSPORT PAUSE Command

MEDIATRANSPORT PAUSE

Access Level

Operate

Syntax

```
MEDIATRANSPORT PAUSE app group operation

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
```



```
operation = one of (0,$FF):
   0 = pause track
   255 = resume playing track
   other values reserved
```

Use

Requests that the *output unit* in the given *media link group* pause or resume the current *track*.
If the unit is in the stopped or paused state, a pause operation is ignored.
If the unit is in the stopped or playing state, a resume operation is ignored.

Examples

```
MEDIATRANSPORT PAUSE 254/$C0 2 1
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.108 MEDIATRANSPORT PLAY Command

MEDIATRANSPORT PLAY

Access Level

Operate

Syntax

```
MEDIATRANSPORT PLAY app group

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
```

Use

Requests that the *output unit* in the given *media link group* play tracks from the current *selection*.
If the *output unit* is in the stopped state, play begins from the first track in the *selection*. (This may be randomised if the SHUFFLE modifier is active.)
If the *output unit* is in the paused state, playing resumes from where it was paused.
If the *output unit* is in the playing state, the command has no effect other than turning off any fast-forward state activated.

Examples

```
MEDIATRANSPORT PLAY 254/$C0 2
```


Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.109 MEDIATRANSPORT REPEAT Command

MEDIATRANSPORT REPEAT

Access Level

Operate

Syntax

```
MEDIATRANSPORT REPEAT app group operation

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0,$FF):
   0 = repeat is off
   1..254 = repeat current track
   255 = repeat all tracks
```

Use

Sets the *repeat* modifier of the given *media link group*.
The repeat modifier can be changed at any time.

Examples

```
MEDIATRANSPORT REPEAT 254/$C0 2 1
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.110 MEDIATRANSPORT REWIND Command

MEDIATRANSPORT REWIND

Access Level

Operate

Syntax

```
MEDIATRANSPORT REWIND app group operation

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0,$FF):
   0 = cease rewind, play at normal speed
   2 = rewind at 2x speed
   4 = rewind at 4x speed
   6 = rewind at 8x speed
   8 = rewind at 16x speed
  10 = rewind at 32x speed
  12 = rewind at 64x speed
  all other values reserved
```

Use

Requests that the *track* currently being played in the given *media link group* be played in reverse. Continues to be in effect until a PLAY, FORWARD 0 or another REWIND command is received. If the *output unit* is not in the play state, the command is ignored.

Examples

```
MEDIATRANSPORT REWIND 254/$C0 2 0
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.111 MEDIATRANSPORT SELECTION_NAME Command

MEDIATRANSPORT SELECTION_NAME

Access Level

Operate

Syntax

```
MEDIATRANSPORT SELECTION_NAME app group wni total index text

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
wni = name identification (0..7)
   0 = current selection
```



```
   1 = next selection
   2 = next+1 selection
   3,4 = reserved
   5 = previous selection
   6 = previous-1 selection
   7 = enumerated selection
total = total number of packets - 1 (0..3)
index = index/sequence of this packet (0..3)
text = all/part of the selection name in UTF-8 (max 11 bytes per packet)
```

Use

Sets the name of a *selection* in the output unit in the given *media link group*.
The name is constructed from one or more packets, sent in sequence, coded as UTF-8, limited to 40 bytes total length.

Examples

```
MEDIATRANSPORT SELECTION_NAME 254/$C0 2 1 1 0 "List 2"
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.112 MEDIATRANSPORT SET_CATEGORY Command

MEDIATRANSPORT SET_CATEGORY

Access Level

Operate

Syntax

```
MEDIATRANSPORT SET_CATEGORY app group category-number

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
category-number = category to select (0..127)
(the meaning of this depends on the output unit)
```

Use

Requests that the given *category-number* be used by the given *media link group's output unit*.
If the given *category-number* doesn't exist, no action is taken.

Examples


```
MEDIATRANSPORT SELECT_SOURCE 254/$C0 2 5
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.113 MEDIATRANSPORT SET_SELECTION Command

MEDIATRANSPORT SET_SELECTION

Access Level

Operate

Syntax

```
MEDIATRANSPORT SET_SELECTION app group selection-number

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
selection-number = selection number to select
```

Use

Requests that the *selection-number* used by the given *media link group* be changed to that specified.
If the given *selection-number* doesn't exist in the *active category*, then the *active selection* isn't changed.

Examples

```
MEDIATRANSPORT SET_SELECTION 254/$C0 2 0 2
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.114 MEDIATRANSPORT SET_TRACK Command

MEDIATRANSPORT SET_TRACK

Access Level

Operate

Syntax

```
MEDIATRANSPORT SET_TRACK app group track-number

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
track-number = track to select (0..$FFFF)
(the meaning of this depends on the output unit)
```

Use

Requests that the given `track-number` be played by the given *media link group's output unit*.
If the given `track-number` doesn't exist in the *active selection*, no action is taken.

Examples

```
MEDIATRANSPORT SET_TRACK 254/$C0 2 5
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.115 MEDIATRANSPORT SHUFFLE Command

MEDIATRANSPORT SHUFFLE

Access Level

Operate

Syntax

```
MEDIATRANSPORT SHUFFLE app group operation

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0,$FF):
   0 = shuffle is off
   255 = shuffle is on
   1..254 = reserved
```

Use

Sets the *shuffle* modifier of the given *media link group*.
The shuffle modifier can be changed at any time.

Examples

```
MEDIATRANSPORT SHUFFLE 254/$C0 2 0
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.116 MEDIATRANSPORT SOURCE_POWER Command

MEDIATRANSPORT SOURCE_POWER

Access Level

Operate

Syntax

```
MEDIATRANSPORT SOURCE_POWER app group state

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
state = one of 0 or 1..255
   0 = power off
   not 0 = power on
```

Use

Turns on or off, the *output unit* in the given *media link group*.
If the unit doesn't have power-control ability, this command is ignored.

Examples

```
MEDIATRANSPORT SOURCE_POWER 254/$C0 2 1
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.117 MEDIATRANSPORT STATUS_REQUEST Command

MEDIATRANSPORT STATUS_REQUEST

Access Level

Operate

Syntax

```
MEDIATRANSPORT STATUS_REQUEST app group

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
```

Use

Requests current status of the *output unit* in the given *media link group*.

Examples

```
MEDIATRANSPORT STATUS_REQUEST 254/$C0 2
```

Successful Response

```
########################################
```

Failure response

```
400 Syntax Error
```

4.5.118 MEDIATRANSPORT STOP Command

MEDIATRANSPORT STOP

Access Level

Operate

Syntax

```
MEDIATRANSPORT STOP app group

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
```

Use

Requests that the *output unit* in the given *media link group* stop playing.
If the unit is already in the stopped state, this command is ignored.

Examples


```
MEDIATRANSPORT STOP 254/$C0 2
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.119 MEDIATRANSPORT TOTAL_TRACKS Command

MEDIATRANSPORT TOTAL_TRACKS

Access Level

Operate

Syntax

```
MEDIATRANSPORT TOTAL_TRACKS app group tracks

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
tracks = number of tracks (0..$FFFF)
```

Use

An output unit on the Media Link Group is describing how many tracks are available in the currently selected *category* and *selection*

Examples

```
MEDIATRANSPORT TOTAL_TRACKS 254/$C0 2 17
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.120 MEDIATRANSPORT TRACK_NAME Command

MEDIATRANSPORT TRACK_NAME

Access Level

Operate

Syntax

```
MEDIATRANSPORT TRACK_NAME app group wni total index text

app = application address (normally //proj/net/$C0)
group = media link group (0..$FF, where $FF indicates unused)
wni = name identification (0..7)
   0 = current track
   1 = next track
   2 = next+1 track
   3,4 = reserved
   5 = previous track
   6 = previous-1 track
   7 = enumerated track
total = total number of packets - 1 (0..3)
index = index/sequence of this packet (0..3)
text = all/part of the track name in UTF-8 (max 11 bytes per packet)
```

Use

Sets the name of a *track* being played by the output unit in the given *media link group*.
The name is constructed from one or more packets, sent in sequence, coded as UTF-8, limited to 40 bytes total length.

Examples

```
MEDIATRANSPORT TRACK_NAME 254/$C0 2 1 2 0 "Adagio for "
```

Successful Response

```
200 OK.
```

Failure response

```
400 Syntax Error
```

4.5.121 NET

Command

NET

Access Level

Admin

Syntax

```
NET [?]
```

Use

Lists the NET sub-commands:
NET CHECK_UNRAVEL - Checks to see if a networks needs to be unravelled.
NET CLOSE - Closes a network.
NET CREATE - Defines a C-Bus network to add to the current project.
NET DELETE - Deletes a network entry by name.
NET FLUSH - Flushes or deletes the in-memory objects attached to a network.
NET LEARN - Sends learn-mode commands to a network.
NET LIST - Returns a list of available networks defined in the current project.
NET LIST_ALL - Returns a list of all networks defined in all projects.
NET LOAD - Loads network definitions for the current project.
NET OPEN - Opens a network.
NET PINGU - Performs a single MMI to get a list of units from the network.
NET PROJECT_IDENTIFY - Returns the project name of the C-Bus network attached to the specified interface.
NET RENAME - Re-addresses a network, giving it a new name and address.
NET SAVE - Saves the current network configuration.
NET SET_PROJECT_IDENTIFY - Stores a project name in a unit on a network.
NET SYNC - Initiates a network synchronisation operation.
NET SYNCNEW - Checks a network or a single unit address for any new units.
NET UNRAVELUNIT - Find all units at the given addresses and move them to unique addresses.
NET UNRAVEL - Find all units on the network and where necessary move them to unique addresses.

Success response

A series of lines giving help for these commands in the form:
`101-Help: help-information`

Failure responses

4.5.122 NET CHECKUNIT

Command

NET CHECKUNIT

Access Level

Admin

Syntax

```
NET CHECKUNIT net-address [unit-addresses]

net-address = address of the network
unit-addresses = comma delimited sequence of unit addresses, or *
```

Use

Interrogates unit addresses to determine whether there are single or multiple units at each address.

This command performs a single MMI followed by parallel queries to all unit addresses in question. The entire process takes about two seconds to run.

If the unit-addresses parameter is not provided or is a * wildcard, all occupied addresses in the MMI will be queried.

Each response line will begin with either "Duplicate units detected", "Single unit detected", "Single unit with error detected", "One or more units detected", "One or more units with error detected" or "No units detected", and then be followed by the text " at address: xxx".

Examples

```
net checkunit 254 0
120-completed MMI 1 of 1.
120 Duplicate units detected at address: 0

net checkunit 254 5
120-completed MMI 1 of 1.
120 Single unit detected at address: 5

net checkunit 254 0,10,15
120-completed MMI 1 of 1.
120-Duplicate units detected at address: 0
120-Single unit detected at address: 10
120 No units detected at address: 15

net checkunit 254 *
120-completed MMI 1 of 1.
120-Duplicate units detected at address: 0
120-Single unit detected at address: 5
120-Single unit detected at address: 10
120-Single unit detected at address: 11
120 Single unit detected at address: 104
```

Success response

```
120 [xxxxx] detected at address: [xxx]
```

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

NET UNRAVEL

4.5.123 NET CLOCKS

Command

NET CLOCKS

Access Level

Admin

Syntax

```
NET CLOCKS net-address [target | R]

net-address = address of the network
target = the target number of enabled clocks to achieve
R = a constant to perform an emergency recovery of clock only
```

Use

This command is used for managing the number of C-Bus clocks enabled on the network. Not to be confused with the CLOCK commands.

The command has three different behaviours depending on the *target* parameter.

If the *target* parameter is:

- not supplied, then the command returns a list of the unit addresses that have C-Bus clock enabled. This also includes additional information showing the clock generator and the software burdens enabled.
- an integer, then C-Gate will if needed enable or disable additional clocks in order for the network to reach the *target* number of clocks enabled. This may take several seconds.
- the letter 'R', the command will perform an emergency recovery by enabling clock on the gateway. This takes less than a second. Note that C-Gate also has an automatic clock recovery feature that achieves the same outcome.

In all cases the command requires the network to be open first.

The standard recommendation is that each C-Bus network should have three (3) clocks enabled.

WARNING: This command does not take into account network topology. The best practice is to enable clocks on units located in the physical centre of the network topology and to achieve that you will need to program units individually. However this command will always attempt to keep existing clocks enabled, so if you have enabled two clocks by hand and you specify a *target* value of '3', it will preserve the original two enabled clocks and enable one more.

Example

```
# List the clocks enabled, which includes the clock generator (clocks_active).
# Note that unravelled addresses may report more than one unit.
net clocks 254
120-address=0 output_units=1 clocks_enabled=1 clocks_active=1 burdens_enabled=0
120-address=1 output_units=1 clocks_enabled=0 clocks_active=0 burdens_enabled=0
120-address=3 output_units=1 clocks_enabled=0 clocks_active=0 burdens_enabled=1
120-address=4 output_units=1 clocks_enabled=0 clocks_active=0 burdens_enabled=0
120-address=10 output_units=1 clocks_enabled=0 clocks_active=0 burdens_enabled=0
120-address=55 output_units=0
200 OK.

# Units 0 and 3 are unplugged, we can't communicate with this network any more.
net clocks 254
408 Operation failed: MMI failed.

# Emergency recovery.
```



```
net clocks 254 r
120-address=4 output_units=1 clocks_enabled=0 clocks_active=0 burdens_enabled=0
120-Gateway clock at address 4 is now enabled.
200 OK.

# Boosting the network to the recommended number of three clocks.
net clocks 254 3
120-Keeping the master clock at address 4 enabled.
120-Clock at address 1 is now enabled.
120-Clock at address 10 is now enabled.
200 OK.

# Showing the network with three clocks.
net clocks 254
120-address=1 output_units=1 clocks_enabled=1 clocks_active=0 burdens_enabled=0
120-address=4 output_units=1 clocks_enabled=1 clocks_active=1 burdens_enabled=0
120-address=10 output_units=1 clocks_enabled=1 clocks_active=0 burdens_enabled=0
120-address=55 output_units=0
200 OK.
```

Success response

```
200 OK.
```

Failure responses

```
401 Bad object or device ID: Network not found
401 Bad object or device ID: addressed object is not a network
408 Operation failed: ... (a range of other options here depending on particular
failures)
420 Access denied
```

See also

NET OPEN
C-Bus Clock Recovery

4.5.124 NET CLOSE

Command

NET CLOSE

Access Level

Admin

Syntax

```
NET CLOSE net-address
```

Use

Closes the network given on the command line as `net-address`.

Success response

A response giving the project and network closed like:

```
200 OK: //project/net-address
eg
200 OK: //HOME/254
```

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

NET OPEN

4.5.125 NET CREATE

Command

NET CREATE

Access Level

Admin

Syntax

```
NET CREATE name type interface-address [options...]

name = the name of the network to create
type =  one of the valid network types (serial, cni, etherlite, socket, bridge,
modem, wiser)
interface-address = address for the PORT of this network. (This address depends on
the actual type)
serial = local serial port name (as would be returned by the PORT LIST command, eg
COM1)
cni = IP address with an optional port number preceeded by a ':'. With no port
number given, the default CNI port is used.
socket = IP address, followed by a ':' and a port number
etherlite = IP address followed by a ':' and a serial port number, where 0 is the
first port.
bridge = C-Gate object address for the near-side of the bridge that the network is
beyond.
for wiser it is the same as type cni
```

Use

Defines a C-Bus network to add to the current project. This creates the network in state new in the current project. This command also will create an entry for this network in the project tag database, if tag databases are being used. An event is sent on the creation of a network by this mechanism.

Note: this command does not change or update a networks.txt file -- this requires use of the NET SAVE FILE command.

Success response

```
200 OK.
```

Failure responses

```
408 Operation failed: ...
420 Access denied
```

See also

NET DELETE

4.5.126 NET DELETE

Command

NET DELETE

Access Level

Admin

Syntax

```
NET DELETE net-address

net-address = address of the network to delete
```

Use

Deletes a network entry by name.

This command removes the network from the C-Gate server memory.

An event is sent on the deletion of a network by this mechanism.

The network must be in InterfaceState closed in order to be deleted.

Success response

```
200 OK.
```

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
468 Can not delete open network
```


See also

NET CREATE, NET LIST

4.5.127 NET FLUSH

Command

NET FLUSH

Access Level

Admin

Syntax

```
NET FLUSH net-address

net-address = address of the network to flush
```

Use

Flushes or deletes the in-memory objects attached to this network. This means that a subsequent NET SYNC operation will start from scratch. Both Unit objects and Application objects are flushed.

Note: this command can only be executed when a sync is not in progress.

Success response

```
200 OK.
```

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

NET SYNC

4.5.128 NET LEARN

Command

NET LEARN

Access Level

Admin

Syntax


```
NET LEARN net-address app-number grade group
```

Use

Sends learn-mode commands to the specified network and application, using the given grade and group.

Grade values are:

init relay:	0x01
init dim:	0x02
cancel:	0x80
exit relay:	0x81
exit dim:	0x82
exit area:	0x83

Success response

```
200 OK.
```

Failure responses

One or more lines of

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

4.5.129 NET LIST

Command

NET LIST

Access Level

Admin

Syntax

```
NET LIST [project]

project = the name of a project
```

Use

Returns a list of networks defined in the given project. If no project name is given, then the current project is used. The current project is determined by the command session that this command is issued in - see the PROJECT USE command for details of setting the current project. All networks in the project are displayed, whether the connections have been opened or not.

Example

```
net list example
131-network=254 State=new InterfaceState=closed
131 network=100 State=new InterfaceState=closed
```

Success response

One or more lines of:

```
131 network=network-address State=status InterfaceState=interface-state
```

If no networks are found, the following is returned as a response:
`132 no networks found`

Failure responses

```
408 Operation failed
420 Access denied
```

4.5.130 NET LIST_ALL

Command

NET LIST_ALL

Access Level

Admin

Syntax

```
NET LIST_ALL
```

Use

Returns a list of networks from all open projects along with status information about the networks.

Success response

One or more lines of:

```
"135 project=" project-name "address=" net-address "OID=" net-oid "interfaceType="
interface-type "interfaceAddress=" interface-address "state=" object-state
"interfaceState=" interface-state
```

If no networks are found, the following is returned as a response:
`132 no networks found`

Failure responses


```
408 Operation failed
420 Access denied
```

4.5.131 NET LOAD

Command

NET LOAD

Access Level

Admin

Syntax

```
NET LOAD DB | FILE [project]

project = the name of a project
```

Use

Loads network definitions from the given project into memory, from either the project tag database, or from the networks file for that project. If no project name is given, then the current project is used. The current project is determined by the command session that this command is issued in - see the PROJECT USE command for details of setting the current project. This will not over-write any existing definitions of networks held in the C-Gate server memory which have the same names as given in the networks file.

Example

```
net load db example
200 OK.

net load db //example
200 OK.

project use home
200 OK.
net load db
200 OK.
```

Success response

```
200 OK.
```

Failure responses

One or more lines of

```
469 Can not create network: reason-information
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```


See also

NET CREATE, PROJECT LOAD, PROJECT SAVE

4.5.132 NET OPEN

Command

NET OPEN

Access Level

Admin

Syntax

```
NET OPEN net-address
```

Use

Opens a network. Opening a network does the following:

- Attempts to open the port address given in the NET CREATE command -- thus initiating a connection to the C-Bus network that is accessed by the port
- Performs an initial synchronisation of the attached network

Once both these steps are complete, the network `State` will be `ok` and the `InterfaceState` will be `running`.

Success response

A series of responses using:

`120-NET OPEN: ...` delivering progress information as the network is opened, followed by:
`200 OK.`

if all succeed, or by one of the failure responses if the open does not succeed.

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

NET CLOSE

4.5.133 NET PINGU

Command

NET PINGU

Access Level

Admin

Syntax

```
NET PINGU net-address

net-address = address of the network to do a unit ping on
```

Use

Performs a single quick MMI operation to get a list of units from the network. A single MMI is issued, and no retries are performed.

Success response

```
302 Units= comma-separated-list-of-units

For example:

302 Units= 1,2,3,45,255
```

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

NET SYNC

4.5.134 NET PROJECT_IDENTIFY

Command

NET PROJECT_IDENTIFY

Access Level

Admin

Syntax

```
NET PROJECT_IDENTIFY interface-address

interface-address = address of the interface to use, in the form type@address
```

Use

Returns the project name of the C-Bus network attached to the specified interface.

Example


```
net project_identify CNI@localhost:14000
305 Project=TEST UnitCount=6
```

Success response

```
305 Project=project-name UnitCount=unit-count
```

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

NET SET_PROJECT_IDENTIFY

4.5.135 NET RENAME

Command

NET RENAME

Access Level

Admin

Syntax

```
NET RENAME net-address new-net-address ["nofixrefs"]

net-address = address of the network to be renamed/readdresses
new-net-address = new address for the network
"nofixrefs" prevents the command from changing bridge definitions in other
networks in this project to support the new network name.
```

Use

Re-addresses a network, giving it a new name and address.

This takes place immediately.

This command also updates any references to the old network address in any bridge networks unless the optional `nofixrefs` is given at the end of the command. The network address of the bridge network can only be updated when the network is not open, and if this command fails to update the network address, there will be a message in the response that the network address is not changed.

Note: This command doesn't affect or rename the database instance.

Example

```
NET RENAME 1 2
```



```
120-Interface not changed for network 2 because it is open
200 OK.
```

Success response

```
200 OK.
```

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

DBRENAMENET

4.5.136 NET SAVE

Command

NET SAVE

Access Level

Admin

Syntax

```
NET SAVE DB | FILE [project]

project = the name of a project
```

Use

Saves the network configuration for the given project to either the project database, or to a networks file appropriate to the project. If no project name is given, then the current project is used. The current project is determined by the command session that this command is issued in - see the PROJECT USE command for details of setting the current project. Note: this command does not save a project database to disk -- this is done with the PROJECT SAVE command.

Example

```
net save db example
200 OK.

net save db //example
200 OK.

project use home
200 OK.
net save db
200 OK.
```


Success response

```
200 OK.
```

Failure responses

```
408 Operation failed: ...
420 Access denied
```

See also

NET CREATE, PROJECT SAVE, NET LOAD

4.5.137 NET SET_PROJECT_IDENTIFY

Command

NET SET_PROJECT_IDENTIFY

Access Level

Admin

Syntax

```
NET SET_PROJECT_IDENTIFY net-address project-name

net-address = address of the network to be identified
project-name = new project name, must be no more that 8 characters and conform to
the sixbit packing standard
```

Use

Stores a project name in the appropriate unit on the given network.

Success response

```
200 OK.
```

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

NET PROJECT_IDENTIFY

4.5.138 NET SYNC

Command

NET SYNC

Access Level

Admin

Syntax

```
NET SYNC net-address ["fast"] [retry-count]

net-address = address of the network to be synchronised
retry-count = number of retries for commands during the sync process
```

Use

Initiates a network synchronisation operation, returning a number of lines of status information as the synchronisation succeeds. Applications can use this progress information to present progress graphics or show a status screen of some kind if required.

If the optional parameter 'fast' is given, the sync only completes an MMI, type, version and serial number scan before returning. This gives a quick view of a network.

If a retry-count is given, then the network's Retries parameter is set to this value for the duration of this command.

Note: if another sync of the same network is in progress, either a background sync or a sync started by 'NET SYNC' command, the sync command will attach itself to the on going sync and respond with the cached sync outputs in the beginning followed by the remaining outputs as a normal sync operation.

Note: the background sync is explained in Starting a Project or Network. The background sync is controlled by network's AutoSync property which should always be set to yes as explained in Starting a Project or Network. It is not necessary to turn off AutoSync for C-Bus Toolkit, Schedule Plus and other C-Gate clients to work properly. Network's NextSyncTime property indicates when the next scheduled automatic background sync will be executed.

Example command output

```
NET SYNC 254
120-Start of cached output.
120-Net Sync: started at Wed Oct 16 13:52:33 CST 2013
120-Net Sync: synchronising PCIs
120-Net Sync: starting network discovery
120-Net Sync: Unit count: 6
120-Net Sync: Unit error count: 0
120-Net Sync: Units at: 3, 4, 5, 8, 11, 18
120-Net Sync: Unit errors at:
120-Net Sync: address=3 type=PCINT4 version=4.6.00 serial=100385.2535
catalog=5500PC
120-Net Sync: address=4 type=RELDN12 version=2.7.00 serial=100800.2302
catalog=L5512RVF
120-Net Sync: address=5 type=PCLOCALU version=4.4.08 serial=100120.2490
catalog=5500PCU
120-End of cached output.
120-Net Sync: address=8 type=PCINTU version=5.4.00 serial=100779.1755
```



```
catalog=5500PCU
120-Net Sync: address=11 type=KEYV3 version=1.8.00 serial=100399.2339
catalog=5093NL
120-Net Sync: address=18 type=DIMDN8 version=2.7.00 serial=100800.4036
catalog=L5508D1A
120-Net Sync: synchronising bridges
120-Net Sync: synchronizing unit 1 of 6 at address 3
120-Net Sync: address 3 type PCINT4 version 4.6.00 catalog 5500PC
120-Net Sync: synchronizing unit 2 of 6 at address 4
120-Net Sync: address 4 type RELDN12 version 2.7.00 catalog L5512RVF
120-Net Sync: synchronizing unit 3 of 6 at address 5
120-Net Sync: address 5 type PCLOCALU version 4.4.08 catalog 5500PCU
120-Net Sync: synchronizing unit 4 of 6 at address 8
120-Net Sync: address 8 type PCINTU version 5.4.00 catalog 5500PCU
120-Net Sync: synchronizing unit 5 of 6 at address 11
120-Net Sync: address 11 type KEYV3 version 1.8.00 catalog 5093NL
120-Net Sync: synchronizing unit 6 of 6 at address 18
120-Net Sync: address 18 type DIMDN8 version 2.7.00 catalog L5508D1A
120-Net Sync: finished at Wed Oct 16 13:52:49 CST 2013
200 OK.
```

Success response

```
200 OK.
```

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

CBusNetwork Starting a Project or Network

4.5.139 NET SYNCNEW

Command

NET SYNCNEW

Access Level

Admin

Syntax

```
NET SYNCNEW net-address [unit-number]

net-address = address of the network to do a syncnew operation on
unit-number = (optional) specific unit address to check
```

Use

Checks a network or a single unit address to see if there are any new units.

If no unit number is given, the command checks for any new units on the network and if they are found, returns the type, version and serial number for the units.

If a unit number is given, the command check for a new unit at that address, also checking to see if any duplicate units are found at that address.

Success response

```
# command with unit address given
net syncnew 1 0
120-completed MMI 1 of 1.
120-unit found
120-duplicate test 1/3
120-duplicate test 2/3
120-duplicate test 3/3
120-no duplicate found
120-identifying unit
303 New Unit Found: address=0 type=BRIDGE2N version=4.1.00 serial=79833.98

#command without unit address given
net syncnew 1
120-completed MMI 1 of 1.
303-New Unit Found: address=2 type=PC_GIM version=4.2.00 serial=1048575.4095
303-New Unit Found: address=3 type=DIMDN8 version=1.2.18 serial=69260.99
303-New Unit Found: address=4 type=PCINT4 version=4.0.0 serial=77432.153
303-New Unit Found: address=11 type=PC_CBTI version=4.2.00 serial=84037.22
303-New Unit Found: address=18 type=GATEWLS version=4.2.00 serial=1048575.4095
303-New Unit Found: address=19 type=PC_LOCAL version=3.12 serial=69.13
303-New Unit Found: address=37 type=GATEWLS version=4.2.00 serial=1048575.4095
303 New Unit Found: address=255 type=RELDN12 version=1.2.18 serial=69923.58
```

Failure responses

```
401 Bad object or device ID: Network not found
401 Bad object or device ID: addressed object is not a network
408 Operation failed: Unit at given address not found
408 Operation failed: Unit already in model
408 Operation failed: No new units found
408 Operation failed: ... (a range of other options here depending on particular
failures)
420 Access denied
```

See also

NET SYNC

4.5.140 NET UNRAVELUNIT

Command

NET UNRAVELUNIT

Access Level

Admin

Syntax

```
NET UNRAVEL net-address unit-addresses [matchdb]

net-address = address of the network
unit-addresses = comma delimited sequence of unit addresses, or *
matchdb = if specified, units will be moved to their database addresses where
possible
```

Use

Performs a series of operations on a network to find all units at the given address or addresses and where necessary move them to unique addresses.

This is similar to the NET UNRAVEL command but only affects the addresses given. Please see the help for that command for more information about the unravelling process.

Example

```
net unravel 254 6
120-completed MMI 1 of 1.
120-Unravel: Unit count: 9
120-Unravel: Units at: 0, 1, 2, 3, 4, 5, 6, 7, 45
120-Unravel: Scanning unit 1 of 9 at address 6 (0x06).
120-Unravel: Unit at address 6 (0x06) with serial number 664712.6 left in place.
120-Unravel: Complete.
200 OK.
```

Success response

```
200 OK.
```

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

NET UNRAVEL, NET SYNC

4.5.141 NET UNRAVEL

Command

NET UNRAVEL

Access Level

Admin

Syntax

```
NET UNRAVEL net-address [matchdb]

net-address = address of the network
matchdb = if specified, units will be moved to their database addresses where
possible
```

Use

Performs a series of operations on a network to find all units and where necessary move them to unique addresses. This is a necessary operation to commission a C-Bus network, as all units invariably come from the factory set to address 255.

The unravel process will:

- Perform an MMI to find occupied addresses.
- Interrogate the gateway unit and ensure no other units are at that address.
- If the gateway unit is a bridge, move it to the correct address matching the near side network.
- Interrogate each remaining unit address. If duplicates are found, they will be moved to unique addresses.
- If 'matchdb' is specified and a matching unit is found in the database, ensure the unit ends up at that address.
- Leave one unit at each originally occupied address. This ensures an already unravelled network will not change if it is unravelled again.
- Move all other units to vacant addresses starting at 2 and increasing.
- Cope with old units without serial numbers (though it will proceed slowly at these addresses).

Typical responses are:

```
120-completed MMI 1 of 1.
120-Unravel: Unit count: <count>
120-Unravel: Units at: <comma delimited addresses>
120-Unravel: Unravelling gateway units...
120-Unravel: Top unit was sent to <address> (<hex-address>).
120-Unravel: Restored top unit to address <address> (<hex-address>).
120-Unravel: Scanning unit <no> of <count> at <address> (<hex-address>).
120-Unravel: Unit at address <address> (<hex-address>) with serial number <serial>
moved to address <address> (<hex-address>).
120-Unravel: Unit at address <address> (<hex-address>) moved to address <address>
(<hex-address>).
120-Unravel: Last unit at address <address> (<hex-address>) with serial number
<serial> left in place.
120-Unravel: Complete.
120-Unravel: Updating network model at address <address> (<hex-address>)
200 OK.
```

Note that there is also a NET UNRAVELUNIT command that allows you to specify a subset of addresses to be unravelled.

Example

```
net unravel 254
120-completed MMI 1 of 1.
```



```
120-Unravel: Unit count: 1
120-Unravel: Units at: 255
120-Unravel: Unravelling gateway units...
120-Unravel: Unit at address 255 (0xFF) moved to address 2 (0x02).
120-Unravel: Top unit was sent to 2 (0x02).
120-Unravel: Unit at address 255 (0xFF) with serial number 100296.2545 moved to
address 3 (0x03).
120-Unravel: Unit at address 255 (0xFF) with serial number 69260.9 moved to
address 4 (0x04).
120-Unravel: Unit at address 255 (0xFF) with serial number 68928.124 moved to
address 5 (0x05).
120-Unravel: Unit at address 255 (0xFF) with serial number 65504.1 moved to
address 6 (0x06).
120-Unravel: Unit at address 255 (0xFF) with serial number 68742.214 moved to
address 7 (0x07).
120-Unravel: Unit at address 255 (0xFF) with serial number 68426.48 moved to
address 8 (0x08).
120-Unravel: Unit at address 255 (0xFF) moved to address 9 (0x09).
120-Unravel: Unit at address 255 (0xFF) moved to address 10 (0x0A).
120-Unravel: Unit at address 255 (0xFF) moved to address 11 (0x0B).
120-Unravel: Unit at address 255 (0xFF) moved to address 12 (0x0C).
120-Unravel: Scanning unit 1 of 1 at address 255 (0xFF).
120-Unravel: Complete.
120-Unravel: Updating network model at address 255 (0xFF)
120-Unravel: Updating network model at address 12 (0x0C)
120-Unravel: Updating network model at address 11 (0x0B)
120-Unravel: Updating network model at address 10 (0x0A)
120-Unravel: Updating network model at address 9 (0x09)
120-Unravel: Updating network model at address 8 (0x08)
120-Unravel: Updating network model at address 7 (0x07)
120-Unravel: Updating network model at address 6 (0x06)
120-Unravel: Updating network model at address 5 (0x05)
120-Unravel: Updating network model at address 4 (0x04)
120-Unravel: Updating network model at address 3 (0x03)
120-Unravel: Updating network model at address 2 (0x02)
200 OK.
```

Success response

```
200 OK.
```

Failure responses

```
401 Bad object or device ID
408 Operation failed: ...
420 Access denied
```

See also

NET UNRAVELUNIT, NET SYNC

4.5.142 NETWORK Command

NETWORK

Access Level

Admin

Syntax

```
NETWORK [?]
```

Use

Lists the NETWORK sub-commands:
NETWORK LOCATE - Sends a locate-yourself message to the matched units.

Success response

A series of lines giving help for these commands in the form:
`101-Help: help-information`

Failure responses

4.5.143 NETWORK LOCATE Command

NETWORK LOCATE

Access Level

Admin

Syntax

```
NETWORK LOCATE app options mode

app = application address for this application, normally net/$d0

options = ( "unit" address ) - locate unit address
          | ( "app" address )  - locate units in application
          | ( "group" app group-address ) - locate units using group
          | ( "serial" manufacturer serial-number - locate using serial &
manufacturer

mode = "on" | "off" | integer in range 0-255
```

Use

Sends a locate-yourself message to the matched units. This puts the unit into a commissioning mode.

Success Response

```
200 OK.
```


Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.144 NEW

Command

NEW

Access Level

Operate

Syntax

```
Defining a group:

NEW object-type object-id param

object-type = "UNIT" | "GROUP" | "PHANTOM"
object-id = a network, group, unit or system entity
param = a parameter appropriate to the object type
```

Use

Creates a new object. The new object is created in state *new*.

Examples

```
NEW unit p/1/$34 DIMMER4 1   (last param is version)
NEW group 42/56/1 area
NEW group 42/56/2 phantom
NEW group 42/56/3
```

Successful Response

```
200 OK.
```

Failure response

```
400, 401, 402, 406, 420
```

See also

GET, SET

4.5.145 NOOP

Command

NOOP

Access Level

Connect

Syntax

```
NOOP
```

Use

Results in a 200 OK reply message with no additional details if the interface is working. Designed to allow simple verification that the interface is operating correctly. A 500 Internal error can only occur when the server has a serious internal error. If no response is received to a NOOP command, then the current command session is not responding. This may be due to the fact that this command interface is still processing an existing command.

Example

```
NOOP
```

Successful Response

```
200 OK
```

Failure response

```
500 Internal error [or no response in the case of a broken connection or other
condition]
```

See Also

or // (comment)

4.5.146 OFF

Command

OFF

Access Level

Operate

Syntax

```
OFF object-identifier ["force"]
```


Use

Turns off the specified group. This command sends a C-Bus group OFF command to the appropriate network.

Specifying the optional parameter `force` allows the command to be sent even if the group is not in an `ok` state.

Examples

```
OFF 254/56/1
```

Successful Response

```
200 OK
```

Failure responses

```
400, 401, 402, 405, 408, 420
```

See also

ON, RAMP, DO

4.5.147 OID

Command

OID

Access Level

Operate

Syntax

```
OID
```

Use

Generates an Object ID. Object IDs are unique, and are based on underlying hardware depending on OS support.

Example

```
oid
301 OID=3da71864-0e90-11d8-ba38-a385e17d8d20
```

Success Response

```
302 OID=oid-value
```

Failure Responses

None

4.5.148 ON

Command

ON

Access Level

Operate

Syntax

```
ON object-identifier ["force"]
```

Use

Turns on the specified group. This command sends a C-Bus group ON command to the appropriate network.

Specifying the optional parameter *force*, allows the command to be sent even if the group is not in an ok state.

Examples

```
ON 254/56/1
```

Successful Response

```
200 OK
```

Failure responses

```
400, 401, 402, 405, 408, 420
```

See also

OFF, RAMP, DO

4.5.149 PORT

Command

PORT

Access Level

Program

Syntax


```
PORT [?]
```

Use

Lists the PORT sub-commands:
PORT CNISCAN - Lists the CNIs found on a network.
PORT IFLIST - Lists the IP network interfaces available on this server.
PORT LIST - Lists available ports that *might* be connected to a C-Bus PCI.
PORT PROBE - Probes a port to locate responding PCIs.
PORT REFRESH - Refreshes the known local serial ports.

Success response

A series of lines giving help for these commands in the form:
101 Help: help-information

Failure responses

4.5.150 PORT CNISCAN

Command

PORT CNISCAN

Access Level

Admin

Syntax

```
PORT CNISCAN [ip-address] ["fast"]

ip-address = (optional) An IP range to scan for CNI devices.  If not given, all
networks connected to the server will be scanned for CNI devices.

If "fast" is given, don't attempt to connect to each CNI.
```

Use

Scans a network looking for CNIs. A list of found CNIs will be returned. For a CNI to response to this command and thus be detected, it must not have network configuration disabled, and must have an IP address assigned to it that matches the network being scanned.

Success response

One or more lines of: `129 found CNI ip-address=`*`ip-address`* `status=`*`status`* port=*`port-number`*
where:
ip-address is the address of the CNI found, and
status is either:

available -- indicating that the port is available for use
unknown -- the port can not be connected to or is in use (it is not possible to differentiate these conditions for a CNI) or, if no CNIs are found: `130 no CNIs found`

port-number is the TCP port number used to connect to the CNI's C-Bus interface.

If no CNIs are detected, the response is:
```
130 no CNIs found
```

Note: this command may detect non-CNI devices that are based on similar components to the CNI. You must use PORT PROBE to actually verify that there is a PCI attached to a device.

Failure responses

```
408 Operation failed: ...
420 Access denied
```

See also

PORT IFLIST, PORT PROBE

4.5.151 PORT CNISCAN2

Command

PORT CNISCAN2

Access Level

Admin

Syntax

```
PORT CNISCAN2 [interface-address] [ip-address] ["fast"]

interface-address = (optional) A network interface on the computer. If not given,
defaults to '0.0.0.0' (wildcard) in which case the Java + OS subsystem will try
and route packets intelligently through all available interfaces.

ip-address = (optional) An IP range to scan for CNI devices.  If not given, all
networks connected to the server will be scanned for CNI devices.

If "fast" is given, don't attempt to connect to each CNI.
```

Examples

```
// invalid adapter
port cniscan2 1.1.1.1
408 Operation failed: scan failed: Can not send to network: Cannot
assign requested address: Cannot bind

// selected adapter
port cniscan2 10.176.60.37
129-ip-address=10.176.148.27 status=unknown port=10001 type=CNI
129-ip-address=10.176.61.60 status=available port=10001 type=CNI2
mac=00:17:DD:01:0E:33 serial=00100700.3526 cbus-unit-address=37
129-ip-address=192.168.0.10 status=available port=0 type=CNIAC
mac=00:17:DD:09:00:17 serial=00000000.0000 cbus-unit-address=0
129 ip-address=10.176.60.34 status=available port=0 type=CNIAC
mac=00:17:DD:09:00:0C serial=00100327.0683 cbus-unit-address=11
```



```
// selected adapter, plus device ip-address filter
port cniscan2 10.176.60.37 10.176.60.34
129 ip-address=10.176.60.34 status=available port=0 type=CNIAC
mac=00:17:DD:09:00:0C serial=00100327.0683 cbus-unit-address=11

// wildcard with device ip-address filter
port cniscan2 0.0.0.0 10.176.60.34
129 ip-address=10.176.60.34 status=available port=0 type=CNIAC
mac=00:17:DD:09:00:0C serial=00100327.0683 cbus-unit-address=11
```

Use

Scans a network looking for CNIs. A list of found CNIs will be returned. For a CNI to response to this command and thus be detected, it must not have network configuration disabled, and must have an IP address assigned to it that matches the network being scanned.

Success response

One or more lines of:

`129 found CNI ip-address=`*`ip-address`* `status=`*`status`* port=*`port-number`* `type=`*`type`* `mac=`*`mac`*
`serial=`*`serial`* `cbus-unit-address=`*`cbus-unit-address`*

where:
ip-address is the address of the CNI found, and
status is either:

available -- indicating that the port is available for use
unknown -- the port can not be connected to or is in use (it is not possible to differentiate these conditions for a CNI) or, if no CNIs are found: `130 no CNIs found`

port-number is the TCP port number used to connect to the CNI's C-Bus interface.
type is the type of the device, e.g. CNI, CNI2, CNIAC.
serial is the C-Bus serial number of the device.
cbus-unit-address is the C-Bus unit address of the device.

If no CNIs are detected, the response is:

```
130 no CNIs found
```

Note: this command may detect non-CNI devices that are based on similar components to the CNI. You must use PORT PROBE to actually verify that there is a PCI attached to a device.

Failure responses

```
408 Operation failed: ...
420 Access denied
```

See also

PORT IFLIST, PORT PROBE

4.5.152 PORT IFLIST Command

PORT IFLIST

Access Level

Admin

Syntax

```
PORT IFLIST
```

Use

Returns a list of IP network interfaces available on this server. Will return a list of all the IP addresses that this host responds to, so this is principally used to work out the IP interfaces on a server to make scanning for CNIs easier.

Note: the loopback interface (127.0.0.1) will not show up in the list of interfaces.

Success response

One or more lines of:

```
127 address=ip-address interface=interface-name
```

Where *interface-name* is the name of the interface and *address* is the IP address of the interface.

The message `128 no interfaces found` is presented if no interfaces are located.

Note: there will be at least **5** seconds of delay between issuing the command and the display of any results. Clients should anticipate this delay.

Failure responses

```
408 Operation failed: Path not found
420 Access denied
```

4.5.153 PORT LIST

Command

PORT LIST

Access Level

Admin

Syntax

```
PORT LIST
```

Use

Returns a list of available ports that *might* be connected to a C-Bus PCI. The list of ports returned is restricted to local serial ports or network ports that use port-redirectors to appear as a local serial port.

Success response

One or more lines of:

```
125 port=port-name status=status
```

Where *port-name* is the name of the port and
status is one of:

inuse -- indicating that the port is already in use and is therefore not available
available -- indicating that the port is available for use
unknown -- for some reason, port state can not be determined
error -- the port is in an error state

If no ports are found, the following is returned as a response:

```
126 no ports found
```

Failure response

```
408 Operation failed
420 Access denied
```

See also

PORT CNISCAN

4.5.154 PORT PROBE

Command

PORT PROBE

Access Level

Admin

Syntax

```
PORT PROBE type address
```

`type` = The type of device to probe. This can be the string: `serial` or `socket`, `cni, wiser` or `etherlite`. (Case is not significant)
`address` = The address to probe.
This address depends on the type above:
for type `serial`, the address is a local serial port name (as would be returned by the PORT LIST command).
for type `cni`, the address is an IP address with an *optional* port number preceeded by a ':'. With no port number given, the default CNI port is used.
for type `wiser`, it is the same as type `cni`
for type `socket`, the address is an IP address, followed by a ':' and a port number
for type `etherlite`, the address is an IP address followed by a ':' and a serial port number, where 0 is the first port.

Note: the probe will fail if a cni device is configured to use a password on connection or is configured to not accept connections from the server.

Use

Probes a port attempting to locate an interface to a C-Bus network (a PCI) that is responding. This command does the following:

1. Attempts to opens a connection of the specified type of port at the given address. If a connection can not be opened, the command fails with an error
2. If possible, sets appropriate HW flow control pins (necessary for C-Touch in PCI mode)
3. Probes the port for a connected and operating PCI.

The tests used will attempt to identify all versions of PCI by tests like: Sending ~~<CR> and then checking for an echo response Sending a CAL command to PCI and check for valid response

Success response

```
230 Probe succeeded: C-Bus network detected (PCIserial=<serial_no>)
```

Note serial number returned if it can be located for the unit in question. If no serial number is present, then the bracketed `PCIserial=` will not be included in the response.

Failure responses

```
431 Probe failed: Port in use
432 Probe failed: No C-Bus network detected
408 Operation failed:
420 Access denied
```

See also

PORT LIST, PORT CNISCAN, PORT IFLIST

4.5.155 PORT REFRESH

Command

PORT REFRESH

Access Level

Admin

Syntax

```
PORT REFRESH
```

Use

Refreshes the known local serial ports.

Success response

```
200 OK.
```

Failure responses

```
408 Error <message>
420 Access denied
```

4.5.156 PROJECT

Command

PROJECT or PROJECT?

Access Level

Admin

Syntax

```
PROJECT [?]
```

Use

Lists the PROJECT sub-commands:
PROJECT ARCHIVE - Archives a project from the existing project directory to a given filename.
PROJECT CLOSE - Closes a project.
PROJECT COPY - Copies a project in the project directory on the server disk.
PROJECT DELETE - Deletes a project from the project directory on the server disk or at another path.
PROJECT DIR - Returns a list of available projects.
PROJECT LIST - Returns a list of all open projects in this C-Gate server.
PROJECT LOAD - Loads a project from disk into the C-Gate server.
PROJECT NEW - Creates a new project.
PROJECT RENAME - Renames a project in the project directory on the server disk.
PROJECT RESTORE - Restores a project from a given archive file.
PROJECT SAVE - Saves a project to disk from the C-Gate server.
PROJECT START - Starts the named or current project.
PROJECT STOP - Stops the named or current project.
PROJECT USE - Sets the context of the current command session to the given project name.

Success response

A series of lines listing these commands in the form:

```
101-Help: help-information
```

4.5.157 PROJECT CLOSE

Command

PROJECT CLOSE

Access Level

Admin

Syntax

```
PROJECT CLOSE [project-name]
```

`project-name` = (optional) Name of the project to close. If not given, then the current project name is used.

Use

Closes a project. Closing a project means that it is no longer held in C-Gate server memory, and is thus can no longer be referenced. This command should be used to close projects that are not actively being used to conserve server resources. You can not close a running project, it must be stopped first.

Success response

```
200 OK.
```

Failure responses

```
401 Bad Object or Device ID
408 Operation failed: ...
420 Access denied
467 Can not close running project
```

See also

PROJECT DIR, PROJECT SAVE, PROJECT LOAD

4.5.158 PROJECT COPY Command

PROJECT COPY

Access Level

Admin

Syntax

```
PROJECT COPY project-name new-project-name
```

`project-name` = Name of the project that will be copied

`new-project-name` = Name that the project will be copied to. *new-project-name* must be a valid project name and use characters allowed in the server's filesystem.

Use

Copies a project in the project directory on the server disk. *Note: this only copies the on-disk project. It has **no** impact on the current loaded projects in memory.*

Success response

```
200 OK.
```

Failure responses

```
401 Bad Object or Device ID
408 Operation failed: ...
420 Access denied
```

See also

PROJECT DIR, PROJECT SAVE, PROJECT LOAD, PROJECT DELETE, PROJECT RENAME

4.5.159 PROJECT DELETE

Command

PROJECT DELETE

Access Level

Admin

Syntax

```
PROJECT DELETE [project-name]
```

Parameters

`project-name` = (optional) Name of the project to delete. If not given, the current project is deleted.

Use

Deletes a project from the project directory on the server disk or at another path. A deleted project will have the filename of the project changed (by the addition of ".deleted" to the end of the filename) so that the project will no longer appear in the list given by the PROJECT DIR command. You can undelete a project by removing the ".deleted" from the project filename. Note that you can delete a currently loaded and running project. This will have no impact on the current operation of the project.

Success response

```
200 OK.
```

Failure responsese


```
401 Bad Object or Device ID
408 Operation failed: ...
420 Access denied
```

See also

PROJECT DIR, PROJECT SAVE, PROJECT LOAD

4.5.160 PROJECT DIR

Command

PROJECT DIR

Access Level

Admin

Syntax

```
PROJECT DIR [path]
```

`path` = (optional) Path to the directory *on the C-Gate server* where the project list is to come from. If a relative path is given, then that path is relative to the C-Gate server's base directory (which, on windows, is normally `\Clipsal\C-Gate`) on the system drive).

Use

Returns a list of available projects, either from the **default project directory**, or from the specified path on the **C-Gate server computer** if an optional path name is given.

Success response

One or more lines of:

`123 project=`*`project-name`* Where *project-name*is the name of the project file, or

`124 no projects found` if there are no projects in the project directory.

Failure responses

```
408 Operation failed: Path not found
420 Access denied
```

4.5.161 PROJECT LIST

Command

PROJECT LIST

Access Level

Admin

Syntax

```
PROJECT LIST
```

Use

Returns a list of all open projects in this C-Gate server. An open project is one that has been loaded into server memory either at startup or via the PROJECT LOAD command.

Success response

One or more lines of:

```
123 project=project-name state=project-state
```

Where *project-name* is the name of the project file and *project-state* is the state of the project (one of *started*, *stopped*, *starting* or *stopping*), or

```
124 no projects found
```

If there are no open projects.

Failure responses

```
408 Operation failed: ..
420 Access denied
```

See also

PROJECT DIR, PROJECT LOAD

4.5.162 PROJECT LOAD

Command

PROJECT LOAD

AccessLevel

Admin

Syntax

```
PROJECT LOAD [project-name [file-name]]
```

`project-name` = (optional) name of the project to load. If not specified, the **current project** is used.
`file-name` = (optional) Name of the file the project can be found in. This can only be specified if the **project-name** is given.

Use

Loads a project from disk into the C-Gate server. This makes the project available to be started, or

accesses with DB and other commands.

Examples

Normal Opening

```
project load cis
200 OK.
```

Loading a project of the wrong DBVersion will fail:

```
project load cis
408 Operation failed: Project load failed: Incompatible project: DBVersion must be
2.1 - perform a TRANSFORM PROJECT command to fix this
```

Success response

```
200 OK.
```

Failure responses

```
401 Bad Object or Device ID
408 Operation failed: ...
420 Access denied
464 Project already loaded
465 Project not found
408 Operation failed: Project load failed: Incompatible project: DBVersion must be
2.1 - perform a TRANSFORM PROJECT command to fix this
```

See also

PROJECT DIR, PROJECT SAVE, PROJECT CLOSE

4.5.163 PROJECT NEW

Command

PROJECT NEW

Access Level

Admin

Syntax

```
PROJECT NEW project-name
```

`project-name` = Name of the project to create.

Use

Creates a new project with the given name.

The project is created under the given name and is initialized in the following way:

- There are no network entries associated with the project
- A blank tag database is created and attached to the project.
- The project is in state `stopped`.

The new project exists in server memory only. PROJECT SAVE must be used to save the project to a project file on the server disk.

Success response

```
200 OK.
```

Failure responses

```
401 Bad Object or Device ID
408 Operation failed: ...
420 Access denied
```

See also

PROJECT DIR, PROJECT SAVE, PROJECT LOAD

4.5.164 PROJECT RENAME

Command

PROJECT RENAME

Access Level

Admin

Syntax

```
PROJECT RENAME project-name new-project-name
```

`project-name` = Name of the project that will change name.
`new-project-name` = New name of this project. The new name must be a valid project name and use characters allowed in the server's filesystem.

Use

Renames a project in the project directory on the server disk. *Note: this only renames the on-disk project. It has **no** impact on the current loaded projects in memory.*

Success response

```
200 OK.
```

Failure responses

```
401 Bad Object or Device ID
408 Operation failed: ...
420 Access denied
```


See also

PROJECT DIR, PROJECT SAVE, PROJECT LOAD, PROJECT DELETE

4.5.165 PROJECT REPAIR

Command

PROJECT REPAIR

Access Level

Admin

Syntax

```
PROJECT REPAIR project-name
```

`project-name` = Name of the project to repair.

Use

Attempts to repair a project that is corrupt or contains invalid data.

This command will:

1. Manually process the file and fix common corruptions in the XML.
2. Execute the transform 'repair.xslt'.
3. Execute the transform 'tidyduplicategroups.xslt'.

A backup copy of the project will be placed in the tag directory.

Success response

```
200 OK.
```

Failure responses

```
401 Bad Object or Device ID
408 Operation failed: ...
420 Access denied
```

See also

PROJECT ARCHIVE, PROJECT DIR, PROJECT SAVE, PROJECT LOAD, PROJECT DELETE

4.5.166 PROJECT RESTORE

Command

PROJECT RESTORE

Access Level

Admin

Syntax

```
PROJECT RESTORE project-name archive-file
```

`project-name` = Name of the project that will change name.

`archive-file` = File to restore from. The filename extension will determine the behaviour of the restore process. If the extension is ".zip", the project will be restored from a ZIP archive (where the tag database is stored as the only entry in the archive, called tagdb.xml). If the extension is ".gz", then file will be ungzipped. If any other extension is given (including .xml), the file will be assumed to be uncompressed XML.

Use

Restores a project from the given archive file to the project name given. The extension of the archive file determines if decompression of the archive file is attempted by this command. See parameters above for details of file extensions and behaviour.

Success response

```
200 OK.
```

Failure responses

```
401 Bad Object or Device ID
408 Operation failed: ...
420 Access denied
```

See also

PROJECT ARCHIVE, PROJECT DIR, PROJECT SAVE, PROJECT LOAD, PROJECT DELETE

4.5.167 PROJECT SAVE

Command

PROJECT SAVE

Access Level

Admin

Syntax

```
PROJECT SAVE [project-name]
```

`project-name` = (optional) Name of the project to save. If not specified, the **current project** is saved.

`file-name` = (optional) Name of the file the project will be saved to. This can only be specified if the `project-name` is given. If not specified, then the filename used for the loading of the project will be used.

Use

Saves a project to disk from the C-Gate server, in a form that can be loaded later.

Success response

```
200 OK.
```

Failure responses

```
401 Bad Object or Device ID
408 Operation failed: ...
420 Access denied
466 Can not save project
```

See also

PROJECT DIR, PROJECT SAVE, PROJECT CLOSE

4.5.168 PROJECT START

Command

PROJECT START

Access Level

Admin

Syntax

```
PROJECT START [project-name]
```

`project-name` = (optional) Name of the project. Project names are as returned by the PROJECT DIR command.

Use

Starts a named project (or the current one). This means that C-Gate will open and commence operating with the networks specified in the definition of this project.

Success response

```
200 OK.
```

Failure responses

```
401 Bad Object or Device ID
408 Operation failed: ...
```



```
420 Access denied
```

See also

PROJECT STOP

4.5.169 PROJECT STOP Command

PROJECT STOP

Access Level

Admin

Syntax

```
PROJECT STOP [project-name]
```

`project-name` = (optional) Name of the project to stop. Project names are as returned by the PROJECT DIR command.

Use

Stops the given or current project.

Success response

```
200 OK.
```

Failure responses

```
401 Bad Object or Device ID
408 Operation failed: ..
420 Access denied
```

See also

PROJECT START

4.5.170 PROJECT USE Command

PROJECT USE

Access Level

Admin

Syntax

```
PROJECT USE [project-name]
```

`project-name` = (optional) Name of the project. Project names are as returned by the PROJECT DIR command.

Use

Sets the context of the current command session to the project name given. This means that all further addressing of networks, groups etc can be performed without using absolute addressing by using the *`//project-name/`* prefix. Note that the project must have already been loaded before it can be set to be the default project with the PROJECT USE command. If no *project-name* is given the command returns a line: `123 project=`*`project-name`*

Success response

```
200 OK.
```

or, if no project name is given:

```
123 project=project-name
```

Failure responses

```
401 Bad Object or Device ID
408 Operation failed: ...
420 Access denied
```

See also

PROJECT LIST

4.5.171 QUIT

Command

QUIT

Access Level

Connect

Syntax

```
QUIT | EXIT
```

Use

Closes the connection to the C-Gate server command interface after sending a final response. Note that this command does not stop the C-Gate server operations or stop operation of the event interface.

Examples


```
QUIT

EXIT
```

Successful Response

```
204 Closing Connection
```

Failure response

```
400 Syntax Error
```

4.5.172 RAMP

Command

RAMP

Access Level

Operate

Syntax

```
RAMP object-identifier ramp-level [ramp-time ["force"]]

ramp-level = 1*3DIGIT ["%"]; in the range 0-255, or 0-100 if "%" is used
ramp-time = 1*2DIGIT ["s" | "m"]; s for seconds, m for minutes
```

Use

Ramps the specified group or area to the percentage level using the specified ramp time. Ramp time means the time need to perform this particular ramp operation.

Ramp Level Parameter

The ramp level can be given in an absolute (0..255) or percentage (0%-100%) range.

Ramp Time Parameter

Supported ramp times are between 0 seconds and 17 minutes. This command sends the C-Bus Group RAMP command to the specified network.

If there is no symbol following the ramp time, the time is assumed to be in seconds. If followed by an 's', then seconds are used. Following the time by 'm' means the time is assumed to be in minutes.
The ramp time parameter is optional. If not given, then the ramp time is set to 0 seconds and the ramp is immediate.

Specifying the optional parameter *force*, will allow the command to be sent even if the group is not in an ok state.

Examples


```
RAMP 57/$38/15 0 4s

RAMP Lobby/$38/12 253 5m

ramp 57/56/1 255 0

ramp 1/56/1 100%
```

Successful Response

```
200 OK
```

Failure responses

```
400, 401, 402, 405, 408, 420
```

See also

ON, OFF, DO

4.5.173 REPORT

Command

REPORT

Access Level

Monitor

Syntax

```
REPORT network-address

Deprecated.  Use TREE.
```

Use

See TREE.

4.5.174 RUN

Command

RUN

Access Level

Operate

Syntax

```
RUN filename [QUIET]
```


Use

Executes a series of commands in the file specified.
Specifying `QUIET` means no output or response events are shown unless an error occurs.

Examples

```
RUN all-lights-off-slowly
```

Successful Response

```
203 Run complete.
```

Failure response

```
4xx syntax and client errors
410 Macro command error
411 Enable command not allowed in macros
412 Macro loop detected
```

See also

```
cgate parameter macro-path
```

4.5.175 SCENE

Command

SCENE

Access Level

Operate

Syntax

```
SCENE command scene-set scene

command = ("PLAY" | "RECORD")

scene-set = NAME

scene = NAME
```

Use

Performs functions for scenes created with the Scene Module – see the section on the Scene Module in the C-Gate manual.

Use this command to record or play a scene from the command line.

Examples


```
SCENE play conf1 allon
```

Successful Response

```
200 OK.
```

Failure response

```
401, 402, 407, 420
```

4.5.176 SECURITY

Command

SECURITY

Access Level

Operate

Syntax

```
SECURITY [?]
```

Use

Lists the SECURITY sub-commands:
SECURITY ARM - Arms a security device.
SECURITY DISPLAY_MESSAGE - Displays a message on the security panel display.
SECURITY EMULATE_KEYPAD - Sends a key press message to the Security system.
SECURITY RAISE_ALARM - Raises an alarm condition with the Security system.
SECURITY REQUEST_ZONE_NAME - Sends a request zone name message to the security system.
SECURITY STATUS_REQUEST - Sends a Security Status Request message to the Security device.
SECURITY TAMPER - Sends a Tamper message to the Security device.

Success Response

A series of lines giving help for these commands in the form:
`101 Help: help-information`

Failure Responses

4.5.177 SECURITY ARM

Command

SECURITY ARM

Access Level

Operate

Syntax

```
SECURITY ARM app arm-mode

app = application address for this application, normally net/$d0

arm-mode =    "away" - arm to away mode
              | "night" - arm to night (home) mode
              | "day" - arm to day mode
              | "vacation" - arm to vacation mode
              | "highest" - arm to highest level of protection
```

Use

Arms a security device.

Success Response

```
200 OK.
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.178 SECURITY DISPLAY_MESSAGE Command

SECURITY DISPLAY_MESSAGE

Access Level

Operate

Syntax

```
SECURITY DISPLAY_MESSAGE app [message]

app = application address for this application, normally net/$d0
message = ASCII message to display, up to 17 characters long
```

Use

Displays a message on the security panel display. The message is given as the last argument to the command, and includes all characters to the end of the line. If the message is missing, then a display message will be generated with no display information.

This message may not be supported by some security systems.

Success Response

```
200 OK.
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.179 SECURITY EMULATE_KEYPAD Command

SECURITY EMULATE_KEYPAD

Access Level

Operate

Syntax

```
SECURITY EMULATE_KEYPAD app key

app = application address for this application, normally net/$d0
key = value from 0 to 127 representing an ASCII character, or a value above 127
representing a custom keypad value
```

Use

Sends a key press message to the Security system. This is equivalent to pressing the given key on the security system's keypad.

This message may not be supported by some security systems.

Success Response

```
200 OK.
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.180 SECURITY RAISE_ ALARM Command

SECURITY ARM

Access Level

Operate

Syntax

```
SECURITY RAISE_ALARM app

app = application address for this application, normally net/$d0
```

Use

Raises an alarm condition with the Security system.

Success Response

```
200 OK.
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.181 SECURITY REQUEST_ZONE_NAME Command

SECURITY REQUEST_ZONE_NAME

Access Level

Operate

Syntax

```
SECURITY REQUEST_ZONE_NAME zone-address

zone-address = address for the zone, normally: net/$d0 zone-number
zone-number = decimal number, or HEX number (preceded by '$')
```

Use

Sends a *request zone name* message to the security system. The security system will report the zone name using a **Zone Name** event.

This message may not be supported by some security systems.

Success Response


```
200 OK.
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.182 SECURITY STATUS_REQUEST Command

SECURITY STATUS_REQUEST

Access Level

Operate

Syntax

```
SECURITY STATUS_REQUEST app ("1"|"2")

app = application address for this application, normally net/$d0
```

Use

Sends a Security Status Request message to the Security device. There are two possible request messages, Status Request 1 and Status Request 2. Select the appropriate number as the argument following the application address.

This command simply performs the request. Applications connected to the C-Gate server that wish to access the results of the status request must receive and decode the **Status Report 1** and **Status Report 2** event.

Success Response

```
200 OK.
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.183 SECURITY TAMPER Command

SECURITY TAMPER

Access Level

Operate

Syntax

```
SECURITY TAMPER app RAISE | DROP

app = application address for this application, normally net/$d0
```

Use

Sends a Tamper Raise or Tamper Drop message to the Security device.

Success Response

```
200 OK.
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.184 SESSION_ID

Command

SESSION_ID

Access Level

Operate

Syntax

```
SESSION_ID
```

Use

When a client application (like C-Bus Toolkit or Schedule Plus) connects to C-Gate it is assigned a sequential command session id to identify its connection with C-Gate. This command returns the ID of the client application's command session.

Example

```
session_id
300 sessionID=cmd3
```

Success Response


```
"300 sessionID="ID

ID: The command session id.
```

Failure Responses

```
400 Syntax Error
```

4.5.185 SESSION_ID ALL

Command

SESSION_ID ALL

Access Level

Operate

Syntax

```
SESSION_ID ALL
```

Use

Returns the ID with extended information of all open command sessions.

Example

```
session_id all
300-sessionID=cmd1 origin=internal from=20130715-135940 tag=Console
300-sessionID=cmd5 origin=MyLaptop/127.0.0.1:63459 from=20130715-142035 tag=C-Bus
Toolkit v1.12.6 build 985
300 sessionID=cmd7 origin=/127.0.0.1:64588 from=20130715-153547
```

Success Response

One or more lines of:

```
"300 sessionID="ID "origin="Origin "from="From ["tag="Tag]

ID: The command session id.

Origin: The combination of the host name, ip address and port number of the client
connection and the format is 'Host Name'/'IP Address':'Port Number'. In some cases
the Host Name field can be blank. For C-Gate's internal console command session it
is 'internal'.

From: The time when the client application establishes the connection with C-Gate.
The time format is 'YYYYMMDD-HHMMSS'.

Tag: The name and version information about the client application which is
reported by the client application using 'SESSION_ID TAG' command. It will not be
included in the response if the client application has not reported the
```



```
information to C-Gate.
```

Failure Responses

```
400 Syntax Error
500 no session found
```

4.5.186 SESSION_ID TAG

Command

SESSION_ID TAG

Access Level

Operate

Syntax

```
SESSION_ID TAG tag_information
```

Use

Reports the client application's name and version information to C-Gate which is shown in 'SESSION_ID ALL' command's responses. This command can only be used once per connection.

Example

```
session_id tag C-Bus Toolkit v1.12.6 build 985
200 OK
```

Success Response

```
200 OK
```

Failure Responses

```
400 Syntax Error
408 Operation failed
```

4.5.187 SET

Command

SET

Access Level

```
Operate
```

Syntax

```
SET object-identifier parameter parameter-value
```



```
parameter = token parameter-value = token
```

Use

Sets parameters in objects. Objects may be part of connected controlled networks, or may be internal to the C-Gate server. Object parameters are described in detail in the guide for the particular object.

Many parameters can be set with the set command. Major general parameters for most or all objects are described here. Specific detail parameters for objects are dealt with in the specific guides for the relevant objects.

Example

```
set #57.1.1 load-power 125

set level42.1 name lobby
```

Successful Response

```
200 OK
```

Failure response

```
400, 401, 402, 405, 408, 420
```

See also

GET or SHOW

4.5.188 SHOW Command

SHOW

Deprecated. Use GET.

Syntax

See GET.

4.5.189 SHUTDOWN Command

SHUTDOWN

Access Level

Admin

Syntax


```
SHUTDOWN
```

Use

Shuts down the C-Gate server. The server will **not** restart, so this command must be used with care. Once the command has been confirmed with the CONFIRM command, the C-Gate server will shut down, closing the command connection.

Examples

```
SHUTDOWN
```

Success Response

```
600 Critical operation. Type CONFIRM to continue.

CONFIRM
206 Shutdown confirmed.
```

Failure response

```
420 Access denied.
```

See also

CONFIRM

4.5.190 SHORTMESSAGE Command

SHORTMESSAGE

Access Level

Operate

Syntax

```
SHORTMESSAGE [?]
```

Use

Lists the SHORTMESSAGE sub-commands:

```
SHORTMESSAGE REFRESH - Transmit the information message of the given type as soon
as possible.
SHORTMESSAGE SEND - Sends a short message.
```

Success Response

A series of 101 Help lines giving a list of the SHORTMESSAGE commands.

Failure Responses

4.5.191 SHORTMESSAGE REFRESH

Command

SHORTMESSAGE REFRESH

Access Level

Operate

Syntax

```
SHORTMESSAGE REFRESH app info-type

app = application address for this application, normally net/$AD

info-type = the type of the message (0..$3F)
   0 = generic text
   1 = weather forecast
   2 = surf report
   3 = tide times
   4 = unread emails
   5 = voice-mail count
   6 = stock prices ticker
   7 = news headline
   8 = other headlines
   9 = air-quality forecast
  10 = UV radiation forecast
  11 = pollen count
  12 = traffic report
  13 = sports result
  14 = horoscope
  15 = electrical brownout warning
  16 = irrigation notice
```

Use

Requests that an information message of the given type be transmitted as soon as possible.

Examples

```
SHORTMESSAGE REFRESH 254/$AD 4
```

Success Response

```
200 OK.
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```


4.5.192 SHORTMESSAGE SEND

Command

SHORTMESSAGE SEND

Access Level

Operate

Syntax

```
SHORTMESSAGE SEND app total index info-type number symbol text

app = application address for this application, normally net/$AD

total = total number of messages (0..7)

index = the number for this message (0..7)

info-type = the type of the message (0..$3F)
   0 = generic text
   1 = weather forecast
   2 = surf report
   3 = tide times
   4 = unread emails
   5 = voice-mail count
   6 = stock prices ticker
   7 = news headline
   8 = other headlines
   9 = air-quality forecast
  10 = UV radiation forecast
  11 = pollen count
  12 = traffic report
  13 = sports result
  14 = horoscope
  15 = electrical brownout warning
  16 = irrigation notice

number = a value in the range 0..$FFFF or a dash (-) if no number portion

symbol = a value in the range 0..$FF or a dash (-) if no symbol portion

text = a text string of up to 14 bytes coded as UTF-8, which can be quoted or
escaped
```

Use

Sends a short message.

Examples

```
SHORTMESSAGE SEND 254/$AD 7 1 4 - - "test"
```


Success Response

```
200 OK.
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.193 STOP

Command

STOP

Access Level

Operate

Syntax

```
STOP command-id
```

Use

Stops a command that is running in the background of this command session. Comands placed in the background are given a unique ID with special command syntax. See the section on Unique Commands IDs for details.

Stop only works with commands that specially support it, and all stops will be performed in a safe way.

Example

```
# example of stopping a command
[33] net sync myNet
stop 33
[33] 207 Stopped.
```

Success Response

A response is returned if the command is stopped, using the id of the stopped command.

```
[id] 207 Stopped.
```

Failure Responses

```
400 Syntax Error
401
408 Operation failed: Stop failed: message
```



```
420 Access denied
```

4.5.194 TELEPHONY

Command

TELEPHONY

Access Level

OPERATE

Syntax

```
TELEPHONY [?]
```

Use

Lists the TELEPHONY sub-commands:
TELEPHONY CLEAR_DIVERSION - Instructs the telephony device to clear any diversion currently in place.
TELEPHONY DIVERT - Sends a message to the telephony device requesting a diversion be set to the given number.
TELEPHONY ISOLATE_SECONDARY_OUTLET - Sends a message to change the isolation behaviour of the secondary outlet of the Clipsal 5500TAU Telephone Interface.
TELEPHONY RECALL_LAST_NUMBER_REQUEST - Asks a telephony device to return the last number called or received.
TELEPHONY REJECT_INCOMING_CALL - Sends a message to the telephony device to reject the current incoming call.

Success response

A series of lines giving help for these commands in the form:

```
101 Help: help-information
```

Failure responses

A functional copy of C-Gate will always respond to this command

4.5.195 TELEPHONY CLEAR_DIVERSION

Command

TELEPHONY CLEAR_DIVERSION

Access Level

Operate

Syntax

```
TELEPHONY CLEAR_DIVERSION app
```


Use

Instructs the telephony device to clear any diversion currently in place.

Success response

```
200 OK
```

Failure responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.196 TELEPHONY DIVERT

Command

TELEPHONY DIVERT

Access Level

Operate

Syntax

```
TELEPHONY DIVERT app number

app = address of the telephony application
number = up to 16 characters that can be interpreted as a telephone number
```

Use

Sends a message to the telephony device requesting a diversion be set to the given number.

Success response

```
200 OK
```

Failure responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.197 TELEPHONY ISOLATE_SECONDARY_OUTLET

Command

TELEPHONY ISOLATE_SECONDARY_OUTLET

Access Level

Operate

Syntax

```
TELEPHONY ISOLATE_SECONDARY_OUTLET app mode

app = address of the telephony application
mode = "normal" | "isolate"
```

Use

Sends a message to change the isolation behaviour of the secondary outlet of the Clipsal 5500TAU Telephone Interface.

Setting `mode` to `normal` will make the secondary outlet behave as normal.

Setting `mode` to `isolate` isolates, or switches out, the secondary outlet

Success response

```
200 OK
```

Failure responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.198 TELEPHONY RECALL_LAST_NUMBER_REQUEST Command

TELEPHONY RECALL_LAST_NUMBER_REQUEST

Access Level

Operate

Syntax

```
TELEPHONY RECALL_LAST_NUMBER_REQUEST app type

app = address of the telephony application
direction = "out" | "in"
```

Use

Requests that a telephony device return the last number called or received. The information is

returned in a RECALL_LAST_NUMBER_RESPONSE message, which will be delivered to the Status Change Port and the event list if appropriately configured.

Setting `direction` to *`out`* will return the number of the last outgoing call made.

Setting `direction` to *`in`* will return the number of the last incoming call received

Success response

```
200 OK
```

Failure responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.199 TELEPHONY REJECT_INCOMING_CALL

Command

TELEPHONY REJECT_INCOMING_CALL

Access Level

Operate

Syntax

```
TELEPHONY REJECT_INCOMING_CALL app

app = address of the telephony application
```

Use

Sends a message to the telephony device to reject the current incoming call.

Success response

```
200 OK
```

Failure responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```


4.5.200 TEMPERATURE

Command

TEMPERATURE

Access Level

Operate

Syntax

```
TEMPERATURE [?]
```

Use

Lists the TEMPERATURE sub-commands:
TEMPERATURE BROADCAST - Broadcasts a temperature value on the given Temperature Group.

Success response

A series of lines giving help for these commands in the form:
`101-Help: help-information`

Failure responses

4.5.201 TEMPERATURE BROADCAST

Command

TEMPERATURE BROADCAST

Access Level

Operate

Syntax

```
TEMPERATURE BROADCAST temp-group-address celsius-temperature ["force"]

temp-group-address = the address of the temperature group to send the temperature
for as a C-Gate address, for example //proj/net1/$19/1

celsius-temperature = temperature in degrees celsius with a maximum of one decimal
place, with no units given
```

Specifying the optional parameter *force* will allow the command to be sent even if the group is not in an ok state.

Use

Broadcasts a temperature value on the given Temperature Group.

Success Response


```
200 OK.
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.202 TERMINATERAMP

Command

TERMINATERAMP

Access Level

Operate

Syntax

```
TERMINATERAMP object-identifier [force]
```

Use

Sends a TERMINATERAMP message on the specified C-Bus Group in the Lighting application. This stops any ramping operation currently occurring for the given group.

Specifying the optional parameter `force` will allow the command to be sent even if the group is not in an `ok` state.

Example

```
# stop ramping of group 1 on lighting application
terminateramp 1/56/1
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
402
405
408 Operation failed: message
420 Access denied
```

See Also

RAMP

4.5.203 TEST_SPAM

Command

TEST_SPAM

Access Level

PROGRAM

Syntax

```
TEST_SPAM [?]
```

Use

Lists the TEST_SPAM sub-commands:
TEST_SPAM EREPORT - Generate random error reporting messages on a network
TEST_SPAM LIGHTING - Generate random lighting messages on a network
TEST_SPAM LIST - List active generation sessions.
TEST_SPAM STOP - Stop an active generation session.

Success response

A series of lines giving help for these commands in the form:

```
101 Help: help-information
```

Failure responses

A functional copy of C-Gate will always respond to this command

4.5.204 TEST_SPAM EREPORT

Command

TEST_SPAM EREPORT

Access Level

PROGRAM

Syntax

```
TEST_SPAM EREPORT net-address interval [max]

interval = number of milliseconds between messages
max = the maximum number of messages to generate
```

Use

Generate random error reporting application messages on a C-Bus network.

The messages are completely random and not guaranteed to use the network's actual groups, thus

it is expected that some of the messages will have no impact.

Example

```
# Send five messages one second apart
test_spam ereport 254 1000 5
301-id=5
200 OK.
```

Success response

```
200 OK.
```

Failure responses

```
401 Bad object or device ID: Network not found
401 Bad object or device ID: addressed object is not a network
408 Operation failed: ... (a range of other options here depending on particular
failures)
420 Access denied
```

4.5.205 TEST_SPAM LIGHTING

Command

TEST_SPAM LIGHTING

Access Level

PROGRAM

Syntax

```
TEST_SPAM LIGHTING net-address interval [max]

interval = number of milliseconds between messages
max = the maximum number of messages to generate
```

Use

Generate random lighting application messages (on application address 56) on a C-Bus network.

The messages are completely random and not guaranteed to use the network's actual groups, thus it is expected that some of the messages will have no impact.

Example

```
# Send messages 100 milliseconds apart indefinitely until stopped
test_spam lighting 254 100
301-id=4
```



```
200 OK.
```

Success response

```
200 OK.
```

Failure responses

```
401 Bad object or device ID: Network not found
401 Bad object or device ID: addressed object is not a network
408 Operation failed: ... (a range of other options here depending on particular
failures)
420 Access denied
```

See also

4.5.206 TEST_SPAM LIST

Command

TEST_SPAM LIST

Access Level

PROGRAM

Syntax

```
TEST_SPAM LIST
```

Use

Display the currently running spam sessions.

Each row provides the spam session id, the type (lighting or ereport), the network it's running on, the time it was started, the interval in milliseconds, the maximum number of messages (if specified) and the current number of messages already sent.

Example

```
test_spam list
300-id=1 type=lighting network=254 started=20131203-172002 interval=1000 max=-1
current=9
300 id=2 type=ereport network=254 started=20131203-172007 interval=500 max=-1
current=7
```

Success response

```
300
```


Failure responses

```
401 Bad object or device ID: Network not found
401 Bad object or device ID: addressed object is not a network
408 Operation failed: ... (a range of other options here depending on particular
failures)
420 Access denied
```

4.5.207 TEST_SPAM STOP

Command

TEST_SPAM STOP

Access Level

PROGRAM

Syntax

```
TEST_SPAM STOP [id]
```

Use

Stop the spam session with the given id.

If no id is given, all spam sessions are stopped.

Example

```
test_spam stop
207-stopped session 1
207-stopped session 2
200 OK.
```

Success response

```
200 OK.
```

Failure responses

```
401 Bad object or device ID: Network not found
401 Bad object or device ID: addressed object is not a network
408 Operation failed: ... (a range of other options here depending on particular
failures)
420 Access denied
```


4.5.208 TOPOLOGY

Command

TOPOLOGY

Access Level

PROGRAM

Syntax

```
TOPOLOGY [?]
```

Use

Lists the TOPOLOGY sub-commands:
TOPOLOGY EXPLORE - Instructs the telephony device to clear any diversion currently in place.

Success response

A series of lines giving help for these commands in the form:

```
101 Help: help-information
```

Failure responses

A functional copy of C-Gate will always respond to this command

4.5.209 TOPOLOGY EXPLORE

Command

TOPOLOGY EXPLORE

Access Level

PROGRAM

Syntax

```
TOPOLOGY EXPLORE addresses

addresses = list of space-separated addresses in the form:
<type>@address;option=value;option=value"
Some address examples are serial@COM1, cni@192.168.1.40:10001,
modem@COM2;number=85431111;username=222;password=2345

Note: A port serial must not be in use, to run this command on it.
```

Use

Explores the topology of a set of interfaces.

Example

```
TOPOLOGY EXPLORER serial@COM1
```

Success response

A series of lines giving help for these commands in the form:

```
323-Network Found NET0 COM1
321-Network Serial NET0 1048575.4095
324 Bridges Found NET0 0
```

Failure responses

```
472 Can not open network:
```

4.5.210 TREE

Command

TREE

Access Level

Monitor

Syntax

```
TREE network-address
```

Use

Returns a tree representation of a network.

Examples

```
TREE 1
```

Successful Response

Response lines starting with 320 that give human-readable information about the network

Failure response

```
400,
401,
402,
420
```

See also

GETSTATE, GET, TREEXML

4.5.211 TREEXML

Command

TREEXML

Access Level

Monitor

Syntax

```
TREEXML network-address ["withsync"] ["withpsync"] ["withqsync"]
```

Use

Returns an XML-formatted view of an open network and basic information about unit parameters. The *withsync*, *withpsync* and *withqsync* options can be used to initiate the appropriate methods on the unit objects prior to producing the XML output.

Alternatively TREEXMLDETAIL command can be used to get <State> and <OnlineStatus> of units in addition to all the parameters returned for TREEXML command.

Examples

```
TREEXML 1
```

Successful Response

A starting line in the form:

```
343-Begin XML snippet
```
followed by zero or more lines of XML data preceeded by "347-" in the treexml format, followed by:
```
344 End XML snippet
```

Example

```
343-Begin XML Snippet
347-<Network>
347- <Unit>
347-  <Error>DUPLICATE</Error>
347-  <Address>3</Address>
347- </Unit>
347- <Unit>
347-  <Type>PC_LOCAL</Type>
347-  <Version>3.12</Version>
347-  <SerialNo>69.13</SerialNo>
347-  <Address>19</Address>
347-  <PartName>NEWUNIT </PartName>
347-  <Application>255, 255</Application>
347-  <Groups></Groups>
347-  <Voltage>33.6</Voltage>
347-  <Burden>no</Burden>
```



```
347-  <Clock>no</Clock>
347- </Unit>
347- <Unit>
347-  <Type>GATEWLS</Type>
347-  <Version>4.2.00</Version>
347-  <SerialNo>1048575.4095</SerialNo>
347-  <Address>37</Address>
347-  <PartName>W2      </PartName>
347-  <Application>56, 255</Application>
347-  <Groups></Groups>
347-  <Voltage>31.0</Voltage>
347-  <Burden>no</Burden>
347-  <Clock>no</Clock>
347- </Unit>
347- <Unit>
347-  <Type>RELDN12</Type>
347-  <Version>1.2.18</Version>
347-  <SerialNo>69923.58</SerialNo>
347-  <Address>255</Address>
347-  <PartName>NEWUNIT </PartName>
347-  <Application>56, 255</Application>
347-  <Groups>17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32</Groups>
347-  <Voltage>31.5</Voltage>
347-  <Burden>no</Burden>
347-  <Clock>no</Clock>
347- </Unit>
347-</Network>
344 End XML Snippet
```

Failure response

```
401 Bad object or device ID
402 Not supported by this object
408 Operation failed
420 Access denied
```

See also

TREE TREEXMLDETAIL

4.5.212 TREEXMLDETAIL

Command

TREEXMLDETAIL

Access Level

Monitor

Syntax

```
TREEXMLDETAIL network-address ["withsync"] ["withpsync"] ["withqsync"]
```

Use

Returns an XML-formatted view of an open network and basic information about unit parameters. Comparing to the TREEXML command, TREEXMLDETAIL returns two additional parameters for the unit information, e.g. <State> and <OnlineStatus>.
The *withsync*, *withpsync* and *withqsync* options can be used to initiate the appropriate methods on the unit objects prior to producing the XML output.

Possible values for <OnlineStatus> are:
- new: as defined in the network property <DBUnitAddressesNew>
- missing: as defined in the network property <DBUnitAddressesMissing>
- online: as defined in the network property <DBUnitAddressesOnline>
- error: as defined in the network property <DBUnitAddressesError>
- duplicate: as defined in the network property <DBUnitAddressesDuplicate>
- unknown: this status is not defined and one possible reason is that the unit address is not in the project database or the MMI report but it has been seen by C-Gate at least once since the project is opened in C-Gate, e.g. a unit is plugged into the network then unplugged while it is not in the project database

Examples

```
TREEXMLDETAIL 254
```

Successful Response

A starting line in the form:

```
343-Begin XML snippet
```
followed by zero or more lines of XML data preceded by "347-" in the treexml format, followed by:
```
344 End XML snippet
```

Example

```
343-Begin XML Snippet
347-<Network>
347- <Unit>
347-  <Type>PCINT4</Type>
347-  <Version>4.6.00</Version>
347-  <SerialNo>100385.2535</SerialNo>
347-  <Address>1</Address>
347-  <PartName>PCI1    </PartName>
347-  <Application>255, 255</Application>
347-  <Groups></Groups>
347-  <Voltage>30.6</Voltage>
347-  <Burden>yes</Burden>
347-  <Clock>yes</Clock>
347-  <State>ok</State>
347-  <OnlineStatus>online</OnlineStatus>
347- </Unit>
347- <Unit>
347-  <Error>DUPLICATE</Error>
347-  <Address>3</Address>
347- </Unit>
347- <Unit>
347-  <Type>PCLOCALU</Type>
```



```
347-  <Version>4.4.08</Version>
347-  <SerialNo>100120.2490</SerialNo>
347-  <Address>5</Address>
347-  <PartName>PCIU    </PartName>
347-  <Application>255, 255</Application>
347-  <Groups></Groups>
347-  <Voltage>31.7</Voltage>
347-  <Burden>yes</Burden>
347-  <Clock>no</Clock>
347-  <State>ok</State>
347-  <OnlineStatus>online</OnlineStatus>
347- </Unit>
347- <Unit>
347-  <Type>PCINTU</Type>
347-  <Version>5.4.00</Version>
347-  <SerialNo>100779.1755</SerialNo>
347-  <Address>6</Address>
347-  <PartName>TVE212  </PartName>
347-  <Application>255, 255</Application>
347-  <Groups></Groups>
347-  <Voltage>31.8</Voltage>
347-  <Burden>no</Burden>
347-  <Clock>no</Clock>
347-  <State>ok</State>
347-  <OnlineStatus>new</OnlineStatus>
347- </Unit>
347-</Network>
344 End XML Snippet
```

Failure response

```
401 Bad object or device ID
402 Not supported by this object
408 Operation failed
420 Access denied
```

See also

TREE TREEXML

4.5.213 TRIGGER

Command

TRIGGER

Access Level

Operate

Syntax

```
TRIGGER [?]
```

Use

Returns help information for the TRIGGER commands

Success response

A series of lines giving help for these commands in the form: 101 Help: help-information

Failure responses

4.5.214 TRIGGER EVENT

Command

TRIGGER EVENT

Access Level

Operate

Syntax

```
TRIGGER EVENT trigger-group-address action-selector
```

Use

Sends a trigger event message to the network to set the trigger-group given as an address to the value of the given action-selector. This is equivalent to a LIGHTING RAMP immediate command for the lighting application. *`action-selector`* is either: an integer in the range 0 through 255; or a percentage value (0 - 100) followed by the `%` sign.

Success response

```
200 OK
```

Failure responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.215 TRIGGER LABEL

Command

TRIGGER LABEL

Access Level

Operate

Syntax


```
TRIGGER LABEL app language group-number action-sel [variant] options

app = application address (normally //proj/net/$CA)
language =  language code (range 0 through 255)
group-number = relevant C-Bus group number (range 0 through 255)
action-sel = action selector | "-"  (dash indicates unset)
variant = F0 | F1 | F2 | F3 (sets the variant for this label)
options = ( "text" text-label ) |
          ( "icon" icon-selector ) |
          ( "dynamic" icon-selector icon-width icon-height vertical-offset
dynamic-icon ) |
          ( "set_language") |
          options-byte hex-bytes
text-label = ASCII text to end of line to be used as label, and its maximum length
is 14 characters
icon-selector = numeric icon selection in the range 0 through 65535
icon-width = width of the icon in pixels
icon-height = height of the icon in pixels
dynamic-icon = block of ASCII-encoded bytes representing the icon, as 16 lines of
up to 62 bits, encoded as up to
16 characters of HEX, with each block separated by a colon character.

If a block is less than 16 characters, the block will be zero-filled on the right.
For example:

AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:
AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:
AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:
AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644:AABBCCDDEEFF6644
```

- `options-byte = integer in the range 0-255, used as the value of the options 1 byte in the command.`
- `hex-bytes = a string of hex bytes to follow the command, and its maximum length is 14 characters (i.e. 28 hex byte characters in the command)`

```
Note: all numeric values except for dynamic icon can be entered as decimal, 0x or
$ prefixed hex, or 0b prefixed binary.
```

Use

Performs label setting on devices that support dynamic labels.

Note: For devices that support unicode dynamic labels, if a group/variant already has a unicode label it will not accept text labels sent with this command. To set or clear unicode labels see the TRIGGER UNICODELABEL command.

Example

```
#set a text label
trigger label 1/56 1 2 3 text hello
200 OK.

#set an existing icon
trigger label 1/56 1 2 - icon $1234
200 OK.
```



```
#set a dynamic icon
trigger label 1/56 1 2 - dynamic
0102030405060708:0102030405060708:0102030405060708:0102030405060708:01020304050607
08:0102030405060708:0102030405060708:0102030405060708:0102030405060708:01020304050
60708:0102030405060708:0102030405060708:0102030405060708:0102030405060708:01020304
05060708:0102030405060708
200 OK.

# change language
trigger label 1/56 1 2 - set_language
200 OK.

# send a raw command
trigger label 1/56 1 2 - 6 3F2B0102
200 OK.
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.216 TRIGGER UNICODELABEL

Command

TRIGGER UNICODELABEL

Access Level

Operate

Syntax

```
TRIGGER UNICODELABEL app language group-number action-sel variant option text-
label|hex-bytes

app = application address (normally //proj/net/$CA)

language =  language code (range 0 through 255)

group-number = relevant C-Bus group number (range 0 through 255)

action-sel = action selector | "-"  (dash indicates unset) (should always be dash)

variant = F0 | F1 | F2 | F3 (sets the variant for this label)
```



```
option = ( "text" text-label ) |
         ( "raw" hex bytes string encoded in UTF-8 )

text-label = ASCII text to end of line to be used as label if option = text

hex-bytes =  a string of hex bytes to follow the command if option = raw
```

Use

Sets unicode labels on devices that support unicode dynamic labels. The label string should be encoded in UTF-8.

To clear a unicode label simply provide no parameter for text-label or hex-bytes.

Note: Please use the raw option to send unicode characters to C-Gate. Using the text option to send unicode may not always work.

Examples

```
#set a text label
trigger unicodelabel 254/56 1 2 - F0 text hello
200 OK.

#set a raw hex bytes string
trigger unicodelabel 254/56 1 2 - F0 raw 414243444546
200 OK.
```

Success Response

```
200 OK.
```

Failure Responses

```
400 Syntax Error
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.5.217 UNLOCK

Command

UNLOCK

Access Level

Operate

Syntax

```
UNLOCK object-identifier
```


Use

Releases an advisory lock on an object. Such a lock would have been set with the LOCK command.

Note that the locking is advisory only. Applications must cooperate to make a locking system work.

Examples

```
UNLOCK p/1/11
```

Successful Response

```
226 <object-identifier>: Unlocked
```

Failure response

```
400, 401, 420, 425
```

See also

LOCK

4.6 Configuration

The behaviour of C-Gate is controlled by Configuration Parameters that are found in different places in a hierarchy. Parameters defined lower down in the hierarchy will override the same parameters defined higher up.

The configuration hierarchy is as follows:

1. **Internal Configuration**

   C-Gate has an internal default configuration that is hard-coded. All Configuration Parameters and their default values are defined here.

2. **Global Configuration**

   This is retrieved from an external configuration file (e.g. C-GateConfig.txt). These values override the Internal Configuration.

3. **Project Configuration**

   These parameters are stored in the project database and will override the Global Configuration. For more information see Scope and Scope Objects.

4. **Network Configuration**

   These parameters are stored in the project database and will override the Project Configuration. For more information see Scope and Scope Objects.

4.6.1 Configuration File

The Configuration File is the customary method for the user to manipulate Configuration Parameters.

Default File

When C-Gate starts it looks for a Global Configuration from a file called C-GateConfig.txt located in the /config directory.

The first time C-Gate is started it will use its default configuration to generate this file from scratch.

The configuration file can also be re-loaded again with the CONFIG LOAD command.

Alternate File

The configuration filename can be altered by supplying a new configuration file as the first parameter when C-Gate is run from the command line. For example, where C-Gte might have been run using:

```
java -jar cgate.jar
```

the following command:

```
java -jar cgate.jar c:\alternate\alternateconfig.txt
```

would load an alternate configuration file instead of the normal C-GateConfig.txt.

4.6.2 Scope and Scope Objects

Scope

Configuration parameters belong to one of three levels of scope.

Global parameters are defined only in the Configuration File. They are specific to the C-Gate instance and cannot be defined at the Project or Network level.

Project parameters are copied from the configuration file to a newly created project. After this point the parameters no longer depend on or use the configuration file. Any changes to project parameters are saved to the project file.

Network parameters are also copied to a newly created project. Initially they behave like project parameters and are only stored once in the file. However different values for the parameter can also be stored against individual networks.

To find out a parameter's scope you can look at its page in Configuration Parameters or read its generated comments in the Global Configuration file.

Scope Objects

In order to manipulate these three levels of scope C-Gate recognises the following four types of scope objects:

`global`	This is a literal constant that refers to the global scope.
`project`	This is a literal constant that refers to the current project in use.
<project>	This is a reference to the project address. See Objects and Addresses.

<network> This is a reference to the network address. See Objects and Addresses.

C-Gate provides the following commands: CONFIG OBGET and CONFIG OBSET.

In order to discard a property value, resetting it to a value higher in scope, use the CONFIG OBRESET command.

Use of the legacy commands CONFIG GET and CONFIG SET is discouraged.

4.6.3 Configuration Commands

The values of configuration parameters can be viewed and set using the following commands in the CONFIG command set. Refer to the descriptions in the Command Descriptions section for more details of using these commands:

- CONFIG - Gives an overview of the CONFIG commands
- CONFIG GET - Get the value of a configuration parameter
- CONFIG SET - Set the value of a configuration parameter
- CONFIG INFO - Get information, including description, scope, default value about a configuration parameter
- CONFIG LOAD - Load configuration parameters from file or project
- CONFIG SAVE - Save configuration parameters to file or project
- CONFIG OBGET - Get the value of a configuration parameter for a scope object
- CONFIG OBSET - Set the value of a configuration parameter for a scope object
- CONFIG OBRESET - Reset the value of a configuration parameter for a scope object to the global value.

4.6.4 Configuration Parameters

This sections lists the configuration parameters and explains their use.

4.6.4.1 accept-connections-from

Parameter Name

accept-connections-from

Description

This parameter gives a space-separated list of host names or IP addresses that C-Gate will accept connections from. The special value `all` will accept connections from all hostnames and addresses.

This is a separate mechanism to Access Control, and it is recommended that the Access Control features be used for security instead of setting this parameter.

Default Value

`all`

Scope

Global

Effective

Restart

4.6.4.2 access-control-file

Parameter Name

access-control-file

Description

This parameter sets the name of the access control file. The file is relative to the

Default Value

`access.txt`

Scope

Global

Effective

Restart

4.6.4.3 allow-fast-start

Parameter Name

allow-fast-start

Description

Setting this parameter to `yes` will allow C-Bus Networks to fast start. Fast starting means that when the network is opened, an initial set of units, groups and other network entities are recovered from the project database, and the details are then verified by checking the network, replacing units in the model that don't physically exist. This means that starting C-Gate for a known network can be performed much faster.

In general, turn this on for stable systems, and turn it off when the networks a C-Gate server connects to frequently change.

Default Value

no

Scope

Network

Effective

CloseOpen

4.6.4.4 allow-v3-pci

Parameter Name

allow-v3-pci

Description

If this parameter is set to yes, C-Gate will take advantage of advanced features in version 3 and later of the C-Bus PCI and CNI modules for network connections. The new features make network operations more reliable.

Leave this set to yes. Setting it to no will decrease C-Bus network reliability.

Default Value

yes

Scope

Global

Effective

CloseOpen

4.6.4.5 application.catalog.filename

Parameter Name

application.catalog.filename

Description

The filename of the application catalog.

Default Value

applications.xml

Scope

Global

Effective

Restart

4.6.4.6 application.catalog.directory

Parameter Name

application.catalog.directory

Description

The directory name where the application catalog is found.

Default Value

`unitspec`

Scope

Global

Effective

Restart

4.6.4.7 auto-reopen

Parameter Name

auto-reopen

Description

This parameter, if set to `yes`, ensures that C-Gate will attempt to re-open a connection to a C-Bus network that has failed due to network errors or other problems. If set to `no`, the network will close as a result of an error and will not reopen without user intervention.

Note for re-sync of the network after auto-reopen: if the network is closed due to network errors or other problems but not 'NET CLOSE' command, after the network is automatically reconnected, a background is scheduled immediately after the reconnection.

Note for bridge network: when the routing network is closed by 'NET CLOSE' command or due to network errors, the bridge network is closed automatically. In either case, C-Gate will try to re-open the connection to the bridge network if auto-reopen is set to `yes`, thus after the routing network is reconnected by automatic re-open or by 'NET OPEN' command, the bridge network will be reconnected if there is no other issue with the bridge connection.

Default Value

`yes`

Scope

Network

Effective

CloseOpen

4.6.4.8 cbus-application

Parameter Name

cbus-application

Description

This parameter gives the default application number to be used when a C-Bus Network is asked about defaults. It is given in decimal. The default of `56` is the normal Lighting Application.

Default Value

`56`

Scope

Network

Effective

CloseOpen

4.6.4.9 cbus-tx-delay

Parameter Name

cbus-tx-delay

Description

This parameter sets the delay in milliseconds between commands sent to a C-Bus Network if tehre is no other flow control mechanism in use (like flow control or sync pci). The default value of `250` is the optimal setting for a network where there is only one PC Interface or CNI connected to it. This value should be increased if other PCIs are connected and sending packets into the network.

Default Value

`250`

Scope

Network

Effective

CloseOpen

4.6.4.10 cbus.tx-cache

Parameter Name

cbus.tx-cache

Description

If this parameter is set to yes, C-Gate will cache outgoing C-Bus commands if they are identical to a previously sent command that C-Gate is still waiting for a response to. When the original command completes C-Gate will apply the same response to the cached commands. In short the redundant commands are never issued onto C-Bus. This behavior is transparent to C-Gate clients however it can be observed in the C-Gate log.

Note that disabling the cache can result in intermittent but unpredictable behavior due to different threads executing identical commands.

Default Value

yes

Scope

Network

Effective

CloseOpen

4.6.4.11 ccp.display-oids

Parameter Name

ccp.display-oids

Description

If set to yes, then the **Config Change Port** will display Object Identifiers (OIDs) along with config change information.

Default Value

yes

Scope

Global

Effective

Restart

4.6.4.12 ccp.display-state

Parameter Name

ccp.display-state

Description

If set to yes, then the **Config Change Port** will send object state change information out to the port.

Default Value

yes

Scope

Global

Effective

Restart

4.6.4.13 cgate-name

Parameter Name

cgate-name

Description

This parameter allows this particular C-Gate server to be given a name that is reported when connections are made to the server.

Default Value

Clipsal C-Gate

Scope

Global

Effective

Restart

4.6.4.14 clock.master

Parameter Name

clock.master

Description

If set to `yes`, then the **Clock and Timekeeping Application** will operate in master mode.

Default Value

no

Scope

Network

Effective

CloseOpen

4.6.4.15 clock.mastermode

Parameter Name

clock.mastermode

Description

This parameter sets the type of master mode used by the **Clock and Timekeeping Application**.

Valid values are `primary` and `secondary`. In primary master mode, C-Gate is much more assertive about setting the time for the C-Bus Networks it is connected to.

Default Value

`secondary`

Scope

Network

Effective

CloseOpen

4.6.4.16 clock.update-interval

Parameter Name

clock.update-interval

Description

This parameter sets the time in minutes between normal clock and timekeeping updates being sent to the C-Bus Network. *Leave this set at the default value unless you are very sure you want to make it different. Use combinations of clock.master and clock.mastermode to change Clock and Timekeeping Application behaviour.*

Default Value

`30`

Scope

Network

Effective

CloseOpen

4.6.4.17 cgroups-file

Parameter Name

cgroups-file

Description

This parameter gives the filename used to hold C-Groups information, relative to the config directory set with the config-path parameter.

Default Value

`Cgroups.txt`

Scope

Project

Effective

CloseOpen

4.6.4.18 command-local-address

Parameter Name

command-local-address

Description

If set to an IP address or hostname, this is the address that Command Interface listens on for connections.

Default Value

The default value is the empty string

Scope

Global

Effective

Restart

4.6.4.19 command-port

Parameter Name

command-port

Description

This parameter gives the TCP/IP port that the Command Interface listens on waiting for connections.

Default Value

`20023`

Scope

Global

Effective

Restart

4.6.4.20 command.encoding

Parameter Name

command.encoding

Description

This parameter gives the character encoding to be used on the command, even, Status Change and Config Change Port. Encodings can be chosen to suit devices connected to C-Gate, though

operating with utf-8 is likely to give the best integration when working with project databases.

Default Value

`utf-8`

Scope

Global

Effective

Restart

4.6.4.21 comms-debug

Parameter Name

comms-debug

Description

If set to `yes`, this parameter enables logging of all network traffic to disk in a file called `comm-debug-<network-name>` in the C-Gate home directory.

Leave this set at it's default value of `no` unless you are asked to change it by Clipsal Staff.

Do not turn this on for fun, as it will cause the server to work hard and it will fill a disk quickly, as it verbosely logs everything going in and out of all network ports.

Default Value

no

Scope

Network

Effective

CloseOpen

4.6.4.22 config-change-port

Parameter Name

config-change-port

Description

This parameter gives the TCP/IP port number for the Config Change Port.

Default Value

`20026`

Scope

Global

Effective

Restart

4.6.4.23 config-path

Parameter Name

config-path

Description

This parameter sets the path that C-Gate uses to look for configuration files, like `C-GateConfig.txt` and `access.txt`. This is given relative to the C-Gate home directory.

Default Value

`config`

Scope

Global

Effective

Restart

4.6.4.24 console.enable-commands

Parameter Name

console.enable-commands

Description

If set to `yes`, allows commands to be entered on the C-Gate console.

Default Value

`yes`

Scope

Global

Effective

Restart

4.6.4.25 default-tag-db

Parameter Name

default-tag-db

Description

This parameter is deprecated.

Default Value

there is no default value.

Scope

Global

Effective

Restart

4.6.4.26 enable.save-state

Parameter Name

enable.save-state

Description

If set to yes, this causes the Enable Control Application to save the state of network variables persistantly to disk when they change.

Default Value

no

Scope

Global

Effective

Restart

4.6.4.27 event-filename

Parameter Name

event-filename

Description

This parameter gives the filename for the an on-disk copy of the event log. This filename is relative to the C-Gate home directory. The given file will have events added to the end of it it if parameter use-event-file is enabled.

Default Value

`event.log`

Scope

Global

Effective

Restart

4.6.4.28 event-file.event-level

Parameter Name

event-file.event-level

Description

This parameter controls the event level of the event file independently of global-event-level. The default value is 9. If adverse performance is experienced this value may be decreased to 5.

Default Value

9

Scope

Global

Effective

Restart

4.6.4.29 event-file.keep-days

Parameter Name

event-file.keep-days

Description

This parameter determines the number of days worth of event-logs to keep. The default value is seven days. Note that there may be several log files for each day.

Note that zero is not a valid value, ie. it is not possible to keep all logs indefinitely.

Default Value

7

Scope

Global

Effective

Restart

4.6.4.30 event-file.split

Parameter Name

event-file.split

Description

This parameter determines whether the event-file will be split.

Default Value

yes

Scope

Global

Effective

Restart

4.6.4.31 event-file.split-count

Parameter Name

event-file.split-count

Description

This parameter determines the maximum number of split event-files.

Default Value

`50`

Scope

Global

Effective

Restart

4.6.4.32 event-file.split-size

Parameter Name

event-file.split-size

Description

This parameter determines the size in bytes at which to split the event-file. The default value is 5,000,000 bytes (4.768 MB)

Default Value

`5000000`

Scope

Global

Effective

Restart

4.6.4.33 event-host

Parameter Name

event-host

Description

This parameter sets the name of the host to send events to. This is used when the **event-mode** parameter is set to `socket`.

Default Value

`localhost`

Scope

Global

Effective

Restart

4.6.4.34 event-millis

Parameter Name

event-millis

Description

If set to `yes`, this parameter causes millisecond timing to be added to events.

Default Value

no

Scope

Global

Effective

Restart

4.6.4.35 event-mode

Parameter Name

event-mode

Description

This parameter selects the mode for serving or sending events. There are two valid values:

`server`, which will run an event server on `event-port` and wait for connections, and
`socket`, which will send the list of events to the host specified in `event-host` at port `event-port`.

Default Value

Scope

Global

Effective

Restart

4.6.4.36 event-port

Parameter Name

event-port

Description

This parameter gives the TCP/IP port number that events will be provided on or delivered to. (See **event-mode** for more information about this).

Default Value

`20024`

Scope

Global

Effective

Restart

4.6.4.37 event-printer

Parameter Name

event-printer

Description

If given a value, this parameter gives the name of the printer to send events to. As such, the parameter is a filename, which can be set to values like LPT1: to redirect output to the printer. If you mis-name the printer, it is likely you will end up with a local file of the name given in the parameter that will continue to grow as C-Gate runs.

Default Value

The default value of this parameter is the empty string.

Scope

Global

Effective

Restart

4.6.4.38 event.display-oids

Parameter Name

event.display-oids

Description

If set to yes, this parameter causes the OID of the referenced obect to be added to event output.

Default Value

yes

Scope

Global

Effective

Restart

4.6.4.39 file.base

Parameter Name

file.base

Description

This parameter sets the base or root directory for the FILE commands. The FILE commands allow the upload and download of files from C-Gate via a command session. The FILE command allow access to any files below this directory.

There are security implications with the settings of this parameter. Be careful when setting it.

Default Value

The default value is . or the directory from which C-Gate was started.

Scope

Global

Effective

Immediate

4.6.4.40 global-event-level

Parameter Name

global-event-level

Description

This parameter sets the level of events sent to the event port. Valid values are between `0` (no events issued) and `9` (all events, includings lots of debugging).

Default Value

5

Scope

Global

Effective

Restart

4.6.4.41 heartbeat-time

Parameter Name

heartbeat-time

Description

This parameter sets the delay in seconds between heartbeat events being issued. This can be used as a keepalive mechanism or a way of ensuring that C-Gate is still operating. A value of `0` disables heartbeat events.

Default Value

`0`

Scope

Global

Effective

Restart

4.6.4.42 hide-project-names

Parameter Name

hide-project-names

Description

If set to yes, this parameter will hide all project names from event and command output. This is provided to work with legacy environments.

Default Value

no

Scope

Global

Effective

Restart

4.6.4.43 instance.lock-file

Parameter Name

instance.lock-file

Description

Filename, relative to the C-Gate base directory, that give the lock file used to ensure that only a single instance of C-Gate is run. To run multiple instances of C-Gate on one computer, make a different configuration file for each instance and start C-Gate specifiying the configuration file on the command line.

Default Value

`cgate.lock`

Scope

Global

Effective

Restart

4.6.4.44 instance.lock-timeout

Parameter Name

instance.lock-timeout

Description

Time in seconds that an instance lock file (see instance.lock-file) is still valid. This timeout is kept short, so after a reboot, any old locks will be invalid and won't prevent C-Gate starting.

Default Value

20

Scope

Global

Effective

Restart

4.6.4.45 lighting.learn-update

Parameter Name

lighting.learn-update

Description

If set to yes, this parameter will allow the Lighting Application to perform database updates as a result of learn mode on the C-Bus Network.

Default Value

yes

Scope

Network

Effective

CloseOpen

4.6.4.46 load-change-port

Parameter Name

load-change-port

Description

This parameter sets the TCP/IP port number used for the **Status Change Port**.

Default Value

`20025`

Scope

Global

Effective

Restart

4.6.4.47 local-flow-control

Parameter Name

local-flow-control

Description

If set to `yes`, flow control will be performed in C-Gate rather than by the serial port. This applies to serial network connections only.

Default Value

no

Scope

Network

Effective

CloseOpen

4.6.4.48 macro-path

Parameter Name

macro-path

Description

This parameter sets the pathname, relative to the C-Gate home directory, where macros are stored.

Default Value

`macros`

Scope

Global

Effective

Immediate

4.6.4.49 memory-report

Parameter Name

memory-report

Description

If set to `yes`, causes memory usage reports to be issued along with heartbeat events. `heartbeat-time` must be non-zero for memory reports to be issued.

Default Value

no

Scope

Global

Effective

Restart

4.6.4.50 network.application-connect

Parameter Name

network.application-connect

Description

If set to `yes`, models bridge Application Connect features by ensuring that any SAL messages sent to a network are relayed to networks via CBus Bridges with Application Connect or Fowarding enabled.

Default Value

`yes`

Scope

Network

Effective

CloseOpen

4.6.4.51 network.error.commands-failed

Parameter Name

network.error.commands-failed

Description

This parameter defines the number of consecutive commands that need to fail before the network is deemed to be in error.

Default Value

3

Scope

Network

Effective

OpenClose

4.6.4.52 network.error.units-failed

Parameter Name

network.error.units-failed

Description

This parameter defines the threshold beyond which the number of units that fail during a sync will place the network into error.

Thus for a value of 3, the network will go into error if 4 units fail to sync.

Default Value

3

Scope

Network

Effective

OpenClose

4.6.4.53 network.error.units-failed-hysteresis

Parameter Name

network.error.units-failed-hysteresis

Description

This parameter controls the number of units that need to recover in the subsequent sync before the network can be taken out of error.

It applies only to networks that are already in error, and is subtracted from the value of *network.error.units-failed* to determine a new threshold for failed units.

For example, if *network.error.units-failed=3* and *network.error.units-failed-hysteresis=1*, and a network sync failed with four units in error, then the next sync must report two or fewer units in error in order to return to an ok state.

This is useful in cases where an intermittent faulty unit is repeatedly sending a network in and out of error. Using a hysteresis will keep the network in error until the problem is properly fixed.

Default Value

`0`

Scope

Network

Effective

OpenClose

4.6.4.54 network.pci.poll-interval

Parameter Name

network.pci.poll-interval

Description

This parameter defines the interval in seconds to poll gateway to check the network connection status and unit online statuses.

Default Value

`60`

Scope

Network

Effective

OpenClose

4.6.4.55 network.retries

Parameter Name

network.retries

Description

This parameter gives the default number of retries to be performed by each command sent to a network for a response or for processing. This sets the value of the Retries parameter for all networks when they are opened.

Default Value

2

Scope

Network

Effective

OpenClose

4.6.4.56 network.retries.pci-check

Parameter Name

network.retries.pci-check

Description

This parameter indicates when to perform a PCI connection check when retrying a failed command. A value of 1 means that the command will be retried once, then a PCI connection check will take place, then the remainder of the retries will take place up to the number specified by *network.retries*. A value of zero means that no PCI connection check will be performed.

Default Value

1

Scope

Network

Effective

OpenClose

4.6.4.57 network.source

Parameter Name

network.source

Description

This parameter gives the default source for network definitions. Valid values are `db`, for project database derived network definitions, or `file`, for network.txt file network definitions.

Default Value

`db`

Scope

Project

Effective

Restart

4.6.4.58 network.state-interval

Parameter Name

network.state-interval

Description

if non-zero, this parameter gives the interval in seconds between reporting of network state information as an event 704.. A value of 0 disables the reporting.

Default Value

`0`

Scope

Global

Effective

Restart

4.6.4.59 networks-file

Parameter Name

networks-file

Description

This parameter gives the name of the file used to hold network definitions. This is prefixed by the project name to make the network filename.

Default Value

`networks.txt`

Scope

Global

Effective

Restart

4.6.4.60 patch.archive-file

Parameter Name

patchset.file

Description

This parameter sets the filename for the patch set.

Default Value

`patchset.zip`

Scope

Global

Effective

Restart

4.6.4.61 pci-flow-control

Parameter Name

pci-flow-control

Description

If set to `yes`, enables XON/XOFF flow control in version 3 or later PC Interfaces.

Default Value

`yes`

Scope

Network

Effective

CloseOpen

4.6.4.62 pci.local-sal

Parameter Name

pci.local-sal

Description

If set to yes, places version 3 or later PCI or CNi devices into Local_SAL mode, which makes application messaging compatible with bridge Application Connect messaging.

Default Value

yes

Scope

Network

Effective

CloseOpen

4.6.4.63 pp.spec-base-directory

Parameter Name

pp.spec-base-directory

Description

This parameter sets the base directory for Unit Specifications.

Default Value

unitspec/

Scope

Global

Effective

Restart

4.6.4.64 project.default

Parameter Name

project.default

Description

This parameter sets the name of the default project.

Default Value

`system`

Scope

Global

Effective

Restart

4.6.4.65 project.default.archive-dir

Parameter Name

project.default.archive-dir

Description

This parameter sets the default directory used for archiving project using the PROJECT ARCHIVE and PROJECT RESTORE commands.

Default Value

tag/archived

Scope

Global

Effective

Restart

4.6.4.66 project.default.dir

Parameter Name

project.default.dir

Description

This parameter gives the default directory to look for project database files in. This is relative to the C-Gate home directory.

Default Value

tag/

Scope

Global

Effective

Restart

4.6.4.67 project.start

Parameter Name

project.start

Description

This parameter gives a space-separated list of projects to start when C-Gate is started. If there are no projects listed, then no projects will be started.

Default Value

`system`

Scope

Global

Effective

Restart

4.6.4.68 reopen-delay

Parameter Name

reopen-delay

Description

This parameter gives the delay time in milliseconds between attempts to open a network that has

been closed or failed during opening.

Default Value

15000

Scope

Network

Effective

CloseOpen

4.6.4.69 report-new-objects

Parameter Name

report-new-objects

Description

If set to yes, this parameter will cause events to be issued indicating when new units are located on the C-Bus Networks that C-Gate has connected to.

Default Value

no

Scope

Network

Effective

CloseOpen

4.6.4.70 response-delay

Parameter Name

response-delay

Description

This parameter gives the time in milliseconds to wait for a response from a C-Bus Network. If no response is received in this time, then retries will be initiated.

Default Value

3000

Scope

Network

Effective

CloseOpen

4.6.4.71 scene-base

Parameter Name

scene-base

Description

This parameter gives the base directory for holding Scenes. This is relative to the C-Bus home directory.

Default Value

scene

Scope

Global

Effective

Restart

4.6.4.72 secure.bind-address

Parameter Name

secure.bind-address

Description

This parameter gives the local address to bind **SSL Connections** to. This can be given as a hostname or IP address. The default value is the empty string, meaning bind to all local connections.

Default Value

The default value is the empty string.

Scope

Global

Effective

Restart

4.6.4.73 secure.port-base

Parameter Name

secure.port-base

Description

This parameter gives the base TCP/IP port number for the SSL Connection. The SSL Connections are all indexed from this base. So, if `secure.port-base=20123`, the default, then:

- command port is on `base+0` or `20123`
- event port is on `base+1` or `20124`
- status change port (SCP) is on `base+2` or `20125`
- config change port (CCP) is on `base+3` or `20126`

Default Value

`20123`

Scope

Global

Effective

Restart

4.6.4.74 serial.fixbaud

Parameter Name

serial.fixbaud

Description

This parameter, valid for serial network types, allows C-Gate to attempt to perform baud-rate scanning and changing the baud rate of a serial connection if there is no response at the standard 9600bps.

Default Value

`yes`

Scope

Network

Effective

CloseOpen

4.6.4.75 speed-write

Parameter Name

speed-write

Description

If set to yes, this parameter enables fast writing of lighting and other application commands to the connected C-Bus networks, by performing write caching and command concatenation. This results in much higher throughput to the C-Bus Networks, though commands are queued and OK responses are given immediately, meaning that individual errors will go undetected if the client does not watch the output from the event port.

Use speed-write=yes in situations where a client application wants to send a large block of commands to C-Gate and requires quick responses.

Default Value

yes

Scope

Network

Effective

CloseOpen

4.6.4.76 startup-delay

Parameter Name

startup-delay

Description

This parameter is DEPRECATED. Do not use.

Default Value

0

Scope

Network

Effective

CloseOpen

4.6.4.77 sweep-timeout

Parameter Name

sweep-timeout

Description

This parameter gives the delay in milliseconds between executions of the queue sweeper.

Default Value

2000

Scope

Network

Effective

CloseOpen

4.6.4.78 sync-fast-pci

Parameter Name

sync-fast-pci

Description

If set to yes, this parameter causes network connections to be operated synchronously when supported by a PC Inteface or CNI device. In general, this should be left set to yes to ensure connection integrity.

Default Value

yes

Scope

Network

Effective

CloseOpen

4.6.4.79 sync-time

Parameter Name

sync-time

Description

This parameter gives the time in seconds over which a network will be synchronised. As network synchronisation is spaced out to perform continuous sync operations, this is the maximum time that will occur before a unit is re-syncronised.

Note: This option
Note: C-Bus Toolkit provides a "C-Gate Config Options" dialog that can directly modify this value.

Default Value

`3600`

Scope

Network

Effective

CloseOpen

4.6.4.80 sync.free-periods

Parameter Name

sync.free-periods

Description

This parameter defines a set of periods during which background syncs are avoided. This is done by means of a comma-delimited string which represents one or more schedules.

Only weekly schedules are supported at this time. A weekly schedule takes the format:

```
      W[:<days-of-week>]:<period>

      <days-of-week> is in one of two formats:
            <dow-int>   a decimal integer [1..127] representing a
bitmask of days from Sunday to Saturday.
            <dow-bit>   seven binary digits representing days from
Sunday to Saturday, eg. 1010101.

            These formats are equivalent, for example:
                  0000001 (bin) ==   1 (dec) == Saturday
                  0111110 (bin) ==  62 (dec) == Monday to Friday
                  1000001 (bin) ==  65 (dec) == Saturday and Sunday
```



```
                    1111111 (bin) == 127 (dec) == Every day of the week

            If the value is zero, then no days are active ie. the
schedule is 'deactivated'.
            If this field is not provided, then every day is assumed.

      <period> is in one of two formats:
            <start-time>-<end-time>                  where all times
and durations are hours and minutes
            <start-time>+<duration>                  in military time,
eg. 0800, 1030, 0200.

            These formats are equivalent and are both supported for
convenience.
```

Some useful examples are:

```
      W:0111110:0800-1000                No background syncing between
8am and 10am on weekdays.
      W:62:0800+0200                     Equivalent to the above.
      W:0:0800+0200                      Schedule is deactivated.
      W:0800+0200                        No background syncing between
8am and 10am every day.

      W:0111110:0800-1000,W:2:1600+0200  No background syncing between
8am and 10am on weekdays,
                                         or between 4pm-6pm Fridays.
```

For more information on how this parameter is used by C-Gate, see Sync-Free Periods.

Default Value

<blank>

Scope

Network

Effective

CloseOpen

4.6.4.81 sync.global-pool-size

Parameter Name

sync.global-pool-size

Description

This parameter controls the maximum number of background network syncs that may be in progress at any one time. Too much network sync traffic can saturate the network infrastructure between C-

Gate and the C-Bus networks or it can affect the performance of the C-Gate software, its clients, or the machine C-Gate is running on.

Default Value

25

Scope

Global

Effective

Restart

4.6.4.82 sync.padding.enabled

Parameter Name

sync.padding.enabled

Description

This parameter enables the Sync Padding feature.

Default Value

yes

Scope

Network

Effective

CloseOpen

4.6.4.83 sync.padding.sync-time-factor

Parameter Name

sync.padding.sync-time-factor

Description

When sync.padding.enabled is yes, this parameter provides a multiplier that is applied to sync-time to determine the amount of time over which to space syncs.

It should not be less than zero or exceed 1.0.

Default Value

0.75

Scope

Network

Effective

CloseOpen

4.6.4.84 sync.gateway-pool-size

Parameter Name

sync.gateway-pool-size

Description

This parameter controls the maximum number of background network syncs that may be in progress at any one time through a single C-Gate gateway device (PCI, CNI, etc). This only applies when network bridges are in use to connect to more than one network behind a gateway device. Too much network sync traffic can overwhelm the limited processing capability in the gateway device.

Default Value

3

Scope

Global

Effective

Restart

4.6.4.85 tag-autosave

Parameter Name

tag-autosave

Description

If set to yes, this parameter causes a project database modified by a learn update to be automatically saved to disk.

Default Value

no

Scope

Global

Effective

Immediate

4.6.4.86 tag-name-output

Parameter Name

tag-name-output

Description

If set to yes, this parameter causes tag names to be output in place of addresses in events, and the config and status ports.

Default Value

no

Scope

Global

Effective

Restart

4.6.4.87 tag-use-zip

Parameter Name

tag-use-zip

Description

If set to yes, this parameter causes project databases to be stored in ZIP format.

Default Value

no

Scope

Global

Effective

Restart

4.6.4.88 tag-validate-db

Parameter Name

tag-validate-db

Description

If set to yes, this parameter causes XML validation to always be performed on project databases.

Default Value

yes

Scope

Global

Effective

Restart

4.6.4.89 unit-auto-update-db

Parameter Name

unit-auto-update-db

Description

If set to yes, this parameter causes units to be automatically updated to the tag database as they are synchronised.

Default Value

no

Scope

Network

Effective

Immediate

4.6.4.90 unit-auto-delete

Parameter Name

unit-auto-delete

Description

If set to yes, this parameter causes units to be automatically deleted from the in-memory network model of the unit if no matching unit is found on the corresponding C-Bus Network.

Default Value

yes

Scope

Network

Effective

Immediate

4.6.4.91 unitcatalog.directory

Parameter Name

unitcatalog.directory

Description

This parameter gives the directory where the Unit Catalog is stored. This is relative to the C-Gate home directory)

Default Value

unitspec

Scope

Global

Effective

Restart

4.6.4.92 unitcatalog.filename

Parameter Name

unitcatalog.filename

Description

This parameter gives the filename of the Unit Catalog. This is in the directory given by `unitcatalog.directory`.

Default Value

cbusunits.xml

Scope

Global

Effective

Restart

4.6.4.93 use-1.0-addressing

Parameter Name

use-1.0-addressing

Description

If set to yes, this parameter causes the server to use addresses in the form of C-Gate 1.0 in all output.

Default Value

no

Scope

Global

Effective

Restart

4.6.4.94 use-cgroups

Parameter Name

use-cgroups

Description

If set to yes, this parameter enables C-Groups.

Default Value

no

Scope

Project

Effective

CloseOpen

4.6.4.95 use-config-change-port

Parameter Name

use-config-change-port

Description

If set to yes, this parameter enables the **Config Change Port**.

Default Value

yes

Scope

Global

Effective

Restart

4.6.4.96 use-event-file

Parameter Name

use-event-file

Description

If set to yes, this parameter enables the storage of events as a file in the local filesystem. This filename is determined by `event-filename`.

Default Value

yes

Scope

Global

Effective

Restart

4.6.4.97 use-load-change-port

Parameter Name

use-load-change-port

Description

If set to yes, this parameter enables the Status Change Port.

Default Value

no

Scope

Global

Effective

Restart

4.6.4.98 use-queue-sweeper

Parameter Name

use-queue-sweeper

Description

If set to yes, this parameter enables the queue sweeper, which clears up netoworking issues and packet failures. This shoud be left set to yes, or problems may result.

Default Value

yes

Scope

Network

Effective

CloseOpen

4.6.4.99 use-scenes

Parameter Name

use-scenes

Description

If set to yes, this parameter enables the scene functions in C-Gate.

Default Value

no

Scope

Global

Effective

Restart

4.6.4.100 use-tags

Parameter Name

use-tags

Description

If set to yes, this parameter enables the use of project databases. Leave this set to yes for all normal operation.

Default Value

yes

Scope

Global

Effective

Restart

4.7 C-Bus Applications

4.7.1 Specifying Applications

Application Addresses

All C-Bus applications are identified in C-Bus messages by an application address, an address in the range 0 ($0) through 255 ($FF). The application address is used to associate the particular message with an application on the network.

Application Catalog

The C-Gate server associates Application classes (that is, code to process application messages and comands) with actual applications using the application catalog, normally called applications.xml. This file is normally located in the same place as unit specifications, the unitspec folder.

The application catalog is loaded into the C-Gate server the first time any application is referred to.

For the format and usage of the catalog, see the applications.xml section in this manual.

Extension mechanism (application.spe)

A file called `application.spe` can be placed in the config directory to allow application definitions to be altered at a specific site only. C-Gate does not ship with this file in the `config` directory.

This file is scanned for application definitions and these take precedence over the definitions in the `applications.xml` file.

See the section on application.spe override for details of the file format and extension mechanism.

Best Practice Guide

*Use the **extension mechanism provided by the applications.spe override**, rather than editing the applications catalog itself to make changes to the way C-Gate maps applications to application classes.*

4.7.1.1 applications.xml

The application catalog, `applications.xml`, is the resource that C-Gate uses to resolve network application numbers to decode and encode application messages.

Configuration variables

The name of the application catalog is determined by the value of the `application.catalog.filename` parameter (by default, `applications.xml`) contained in the `application.catalog.directory` directory (default `unitspec`). These values can be changed for special applications if needed. However, general useage and best practice suggest leaving these set to their defaults and using the application.spe override mechanism for specific intergrated applications.

File Format

This is a short but complete applications.xml file. Refer to the actual example in the `config` directory.

```
<Applications>
	<Application>
		<Name>Network</Name>
		<Description>C-Bus Network Management and Control</Description>
		<AddressRangeStart>$FF</AddressRangeStart>
		<AddressRangeEnd>$FF</AddressRangeEnd>
		<DefaultAddress>$FF</DefaultAddress>
		<CGateClass>CBusNetManagementApplication</CGateClass>
		<AllowMMIs>No</AllowMMIs>
```



```
                <AutoCreate>No</AutoCreate>
                <Editable>No</Editable>
                <HasGroups>No</HasGroups>
                <LightingCompatibility>None</LightingCompatibility>
                <NagIfUsed>No</NagIfUsed>
                <RenameAllowed>No</RenameAllowed>
        </Application>
        <Application>
                <Name>lighting</Name>
                <Description>C-Bus Lighting Application</Description>
                <AddressRangeStart>$30</AddressRangeStart>
                <AddressRangeEnd>$5F</AddressRangeEnd>
                <DefaultAddress>$38</DefaultAddress>
                <CGateClass>CBusLightingApplication</CGateClass>
                <AllowMMIs>Yes</AllowMMIs>
                <AutoCreate>Yes</AutoCreate>
                <Editable>No</Editable>
                <HasGroups>No</HasGroups>
                <LightingCompatibility>Complete</LightingCompatibility>
                <NagIfUsed>No</NagIfUsed>
                <RenameAllowed>No</RenameAllowed>
               <KeyMacrofunctions>
                      <KeyMacrofunction>
                         <Name>On/Off</Name>
                         <Microfunction>
                              <Code>SP</Code>
                              <Function>TOGGLE</Function>
                             </Microfunction>
                         <Microfunction>
                              <Code>SR</Code>
                              <Function>TOGGLE</Function>
                             </Microfunction>
                         <Microfunction>
                              <Code>LP</Code>
                              <Function>TOGGLE</Function>
                             </Microfunction>
                         <Microfunction>
                              <Name>LR</Name>
                              <Function>TOGGLE</Function>
                             </Microfunction>
                      </KeyMacrofunction>
               </KeyMacrofunctions>
        </Application>
</Applications>
```

Element Descriptions

An <Application> element contains a definition of an application. It contains the following elements:

<Name> gives the short name used by C-Gate to define the applications. This short name is used to prefix commands and it used to prefix events shown in the event stream and status change stream. This name should be short, have no whitespace, and be in lowercase. It needs to be unique.

<Description> gives a one-sentence description of the application. This is not currently used by C-Gate. This can contain whitespace.

<AddressRangeStart> gives the lowest application address that this application uses. This is in the range 0-255 or $0 - $FF.

<AddressRangeEnd> gives the highest application address that this application uses. This is in the range 0-255 or $0 - $FF.

<DefaultAddress> gives the normal, default address for this application. This is in the range 0-255 or $0 - $FF.

<CGateClass> gives the java classname that C-Gate will load to handle this application. If a fully qualified class path name is given, that is used. Otherwise the class is interpreted relative to com.clipsal.cgate.cbus.app.

The remaining elements are used internally by C-Bus Toolkit and other Clipsal software. These elements are optional.

<AllowMMIs>, if set to yes, allows MMIs to be used on this application.

<AutoCreate>, if set to yes, indicates that the application is automatically created in C-Bus Toolkit projects.

<Editable>No</Editable>, if set to yes, allows the application details to be edited in C-Bus Toolkit.

<HasGroups>No</HasGroups>, if set to yes, indicates that this application has groups.

<LightingCompatibility> indicates the level of lighting application support this application can use. Valid values are: complete, partial, or none.

<NagIfUsed>, if set to yes, ensures that C-Bus Toolkit will verify use of this application during commissioning.

<RenameAllowed>, if set to yes, allows this application to be renamed in projects.

The <KeyMacrofunctions> block gives hints to C-Bus Toolkit for the default definitions used in key input devices. It contains zero or more <Macrofunction> blocks.
Each <Macrofunction> block contains a <Name> and one or more <Microfunction> blocks.

Each Microfunction block has a <Code> and <Function> Element.

Failure modes

C-Gate will fall back to looking for the application.spe file if the applications.spe file is not found.

Historical Note

The application catalog replaces the application.spe file used internally in C-Gate as of version 2.5. The application.spe override is still supported, however.

4.7.1.2 applications.spe override

The application.spe file, contained in the config directory, provides a mechaism to allow individual installations to override the definitions of the the application catalog. This is a plain text file with lines that define the behaviour of C-Gate for particular application addresses.

Warning: by creating overly general overrides in this file, you can stop C-Gate working properly with normal applications.

Lines starting with # are comments, and entries define the following fields, with fields separated by whitespace:

name gives the short name of the application -- this is used as the command prefix and for event prefixes as well.

low and **high** give the range of application numbers that will be recognised as this application. You can see that lighting has a range of application numbers attached to it.

default is the default application number, which will be used when the application is created other than from a network event.

class to load gives the name of the application class (must be a subclass of CBusApplication). If no full package is given on the class, it is assumed to be in the com.clipsal.cgate.cbus.app package. Alternatively if a full package and class is given, that is loaded.

description gives a description of the application.

Some sample `application.spe` entries are given below:

```
#name         low    high default class to load        description
network       $FF    $FF    $FF    CBusNetManagementApplication      C-Bus Network
Management and Control
lighting      $30    $5F    $38    CBusLightingApplication    *C-Bus Lighting
Application
heating       $88    $88    $88    CBusLightingApplication    *C-Bus Heating
Application (Clipsal 5 Star)
rcs           $26    $26    $26    CBusLightingApplication    *Room Control System
(Clipsal 5 Star)
security      $D0    $D0    $D0
com.clipsal.cgate.cbus.app.security.CBusSecurityApplication   Security
```

If no overrides are required, then this file can be empty or missing.

Best practice guide

Only ever use this mechanism if there is no way to renumber C-Bus units to fit into the regular application framework. Renumbering the units will allow other application-aware applications to easily work with the network. Redifining standard applications can cause serious C-Bus malfunctions.

4.7.2 Airconditioning Application

The Airconditioning Application is used to manage airconditioned zones.

A number of events are provided by HVAC devices and a number of commands are available to

control HVAC devices via this application.

The Airconditioning Application has a C-Bus application address of $AC, or 172 decimal.

4.7.2.1 Airconditioning Application Overview

This gives a short overview of the Airconditioning Application implementation in the C-Gate server

Application name

Airconditioning

Application short name

(used as first part of commands or events to refer to this application)

aircon

C-Bus Application range

$AC (172)

Commands

(these can be entered from the C-Gate command interface preceded by the short name of this application. All commands are also recognised as events)

set_humidity_setback_limit
set_humidity_lower_guard_limit
set_humidity_upper_guard_limit
set_hvac_lower_guard_limit
set_hvac_setback_limit
set_hvac_upper_guard_limit
set_ward_off
set_ward_on
set_zone_humidity_mode
set_zone_hvac_mode

Events

(these are detected by the C-Gate server and will be shown in the event list and status port but are not available as commands)

humidity_schedule_entry
hvac_schedule_entry
zone_humidity
zone_humidity_plant_status
zone_hvac_plant_status
zone_temperature

Notes and implementation issues

4.7.2.2 Objects

One new object is provided for the implementation of this application:

Airconditioning Application

This is the application object created in the server when an airconditioning command is detected on the C-Bus network, or if an airconditioning command is sent from the command line. This object provides the following parameters in addition to those supported by the standard Application object.

Parameter Name	Description	Read Access Control	Write Access Control
ShortName	The short name for this application. This is set to **aircon** when the application is loaded	Monitor	Read Only

4.7.2.3 Events

The events in this section are recognised by the Airconditioning application and are made available as events in the C-Gate server event port, or via the Status Change Port (SCP)

The events can be divided into two main groups: Status Events, which emit information about things happening on the application, and Command Events, which indicate that an Airconditioning command has been executed on the application.

4.7.2.3.1 Status Events

Airconditioning Status Events emit information about things happening on the application.

4.7.2.3.1.1 Humidity Schedule Entry

Event

Humidity Schedule Entry

Condition causing event

Setting a humidity schedule entry.

Event text

humidity_schedule_entry app ward zone-list entry format mode raw-flag setback-enabled guard-enabled use-aux-level start-time level

```
app = air-conditioning application address
ward = air-conditioning ward number (0-255)
zone-list = comma-separated list of zone numbers (max 7)
entry = entry number (0..255)
format = format (1..7)
mode = humidity mode (0..3)
raw-flag = 0 if level is a humidity percentage, 1 if level is a raw value
setback-enabled = 1 for setback enabled
guard-enabled = 1 for guard enabled
use-aux-level = 1 if aux-level is used, else 0 for automatic operation
start-time = time this schedule starts
level = humidity % value or raw level
```

- `format:`


```
   1 = 4 periods/day, all days same
   2 = 4 periods/day, week/weekend
   3 = 4 periods/day, each day different
   4 = 6 periods/day, all days same
   5 = 6 periods/day, week/weekend
   6 = 6 periods/day, each day different
   7 = no fixed number of periods
```

- mode:
  0 = off
  1 = humidify only
  2 = dehumidify only
  3 = humidity control

- start-time: number of minutes from Sunday 00:00 (ie Sunday 00:00 = 0)

- humidity level is 0% to 100%

- raw humidity level can be a fraction of plant capacity eg 50% or -10%

Sample Event output

```
20081022-084833 702 sys [aircon] humidity_schedule_entry ward=1 zones=0,1,2
entry=0 format=1 mode=3 rawlevel=0 setbackenabled=0 guardenabled=0 useauxlevel=0
starttime=750 setlevel=13107 sourceUnit=0
```

Sample SCP output

```
# aircon humidity_schedule_entry //BOARD/254/172 1 0,1,2 0 1 3 0 0 0 0 750 13107
#sourceunit=0 OID=...
```

4.7.2.3.1.2 HVAC Schedule Entry

Event

HVAC Schedule Entry

Condition causing event

Setting a HVAC schedule entry.

Event text

hvac_schedule_entry app ward zone-list entry format mode raw-flag setback-enabled guard-enabled use-aux-level start-time level

```
app = air-conditioning application address
ward = air-conditioning ward number (0-255)
zone-list = comma-separated list of zone numbers (max 7)
entry = entry number (0..255)
format = format (1..7)
mode = HVAC mode (0..4)
raw-flag = 0 if level is a temperature, 1 if level is a raw value
```



```
setback-enabled = 1 for setback enabled
guard-enabled = 1 for guard enabled
use-aux-level = 1 if aux-level is used, else 0 for automatic operation
start-time = time this schedule starts
level = temperature or raw level
```

- format:
  1 = 4 periods/day, all days same
  2 = 4 periods/day, week/weekend
  3 = 4 periods/day, each day different
  4 = 6 periods/day, all days same
  5 = 6 periods/day, week/weekend
  6 = 6 periods/day, each day different
  7 = no fixed number of periods

- mode:
  0 = off
  1 = heating only
  2 = cooling only
  3 = heating & cooling
  4 = fan only

- start-time: number of minutes from Sunday 00:00 (ie Sunday 00:00 = 0)

- temperature is in degrees centigrade * 256

- raw temperature level can be a fraction of plant capacity eg 50% or -10%

Sample Event output

```
20081022-093613 702 sys [aircon] hvac_schedule_entry ward=1 zones=0,1,2 entry=0
format=7 mode=3 rawlevel=0 setbackenabled=0 guardenabled=0 useauxlevel=0
starttime=1500 setlevel=5120 sourceUnit=0
```

Sample SCP output

```
# aircon hvac_schedule_entry //BOARD/254/172 1 0,1,2 0 7 3 0 0 0 0 1500 5120
#sourceunit=0 OID=...
```

4.7.2.3.1.3 Zone Humidity Plant Status

Event

Zone Humidity Plant Status

Condition causing event

Reporting the status of a humidity plant.

Event text

zone_humidity_plant_status app ward zone-list type status error


```
app = air-conditioning application address
ward = air-conditioning ward number (0-255)
zone-list = comma-separated list of zone numbers (max 7)
type = type of humidity plant (0..3)
status = humidity plant status (0..255)
error = humidity plant error (0..11)
```

- type:
  0 = none
  1 = evaporative
  2 = refrigerative
  3 = both

- status: a cumulative bitmask of Humidity Plant Status (0..255)
  1 = humidifier active
  2 = dehumidifier active
  4 = fan active
  8 = damper open
 16 = (unused)
 32 = busy
 64 = error
128 = expansion

- error:
  0 = none
  1 = humidifier failure
  2 = dehumidifier failure
  3 = fan failure
  4 = sensor failure
  5 = humidifier problem
  6 = dehumidifier problem
  7 = fan problem
  8 = humidifier service required
  9 = dehumidifier service required
 10 = fan service required
 11 = filter replacement required

Sample Event output

```
20081021-152001 702 sys [aircon] zone_humidity_plant_status ward=1 zones=1,2
type=3 status=1 error=0 sourceUnit=0
```

Sample SCP output

```
# aircon zone_humidity_plant_status //BOARD/254/172 1 1,2 3 1 0 #sourceunit=0
OID=...
```

4.7.2.3.1.4 Zone HVAC Plant Status

Event

Zone HVAC Plant Status

Condition causing event

Reporting the status of a HVAC plant.

Event text

zone_hvac_plant_status app ward zone-list type status error

```
app = air-conditioning application address
ward = air-conditioning ward number (0-255)
zone-list = comma-separated list of zone numbers (max 7)
type = type of HVAC plant (0..10,255)
status = HVAC plant status (0..255)
error = HVAC plant error (0..11)
```

- type:
  0 = none
  1 = furnace
  2 = evaporative
  3 = reverse cycle
  4 = heat-pump heating
  5 = heat-pump cooling
  6 = furnace/evap
  7 = furnace/heat-pump cooling
  8 = hydronic
  9 = hydronic/heat-pump cooling
 10 = hydronic/evap
255 = any

- status: a cumulative bitmask of HVAC plant status (0..255):
  1 = cooler active
  2 = heater active
  4 = fan active
  8 = damper open
 16 = (unused)
 32 = busy
 64 = error
128 = expansion

- error:
  0 = none
  1 = heater failure
  2 = cooler failure
  3 = fan failure
  4 = sensor failure
  5 = heater problem
  6 = cooler problem
  7 = fan problem
  8 = heater service required
  9 = cooler service required
 10 = fan service required
 11 = filter replacement required

Sample Event output

```
20081022-101347 702 sys [aircon] zone_hvac_plant_status ward=1 zones=0,1,2 type=3
status=1 error=0 sourceUnit=0
```

Sample SCP output


```
# aircon zone_hvac_plant_status //BOARD/254/172 1 0,1,2 3 1 0 #sourceunit=0
OID=...
```

4.7.2.3.1.5 Zone Humidity

Event

Zone Humidity

Condition causing event

Reporting the humidity of HVAC zones.

Event text

zone_humidity app ward zone-list level sensor-status

```
app = air-conditioning application address
ward = air-conditioning ward number (0-255)
zone-list = comma-separated list of zone numbers (max 7)
level = humidity in % * 655.35 (ie 100% = 65535)
sensor-status = humidity plant status (0..3)
```

- sensor-status:
  0 = no error
  1 = sensor operating in relaxed accuracy
  2 = sensor out of calibration
  3 = sensor total failure (humidity reading is meaningless)

Sample Event output

```
20081022-124135 702 sys [aircon] zone_humidity ward=1 zones=0,1,2 level=10485
sensorstatus=0 sourceUnit=0
```

Sample SCP output

```
# aircon zone_humidity //BOARD/254/172 1 0,1,2 10485 0 #sourceunit=0 OID=
```

4.7.2.3.1.6 Zone Temperature

Event

Zone Temperature

Condition causing event

Reporting the temperature of HVAC zones.

Event text

zone_temperature app ward zone-list level sensor-status


```
app = air-conditioning application address
ward = air-conditioning ward number (0-255)
zone-list = comma-separated list of zone numbers (max 7)
level = temperature in degrees Centigrade * 256
sensor-status = HVAC plant status (0..3)
```

- temperature is a two-byte signed number, so 0deg = $0000, 0.1deg = $0019, 1deg = $0100, -1deg = $FF00
- sensor-status:
  0 = no error
  1 = sensor operating in relaxed accuracy
  2 = sensor out of calibration
  3 = sensor total failure (temp reading is meaningless)

Sample Event output

```
20081022-113542 702 sys [aircon] zone_temperature ward=1 zones=0,1,2 level=5120
sensorstatus=0 sourceUnit=0
```

Sample SCP output

```
# aircon zone_temperature //BOARD/254/172 1 0,1,2 5120 0 #sourceunit=0 OID=...
```

4.7.2.3.2 Command Events

Airconditioning Command Events indicate that the equivalent Airconditioning command has been executed on the application.

4.7.2.3.2.1 Refresh

Event

Refresh

Condition causing event

The equivalent command has been invoked on the application.

Event text

refresh app ward

For description of parameters see the AIRCON REFRESH command.

Sample Event output

```
20081022-084833 702 sys [aircon] refresh ward=1 sourceUnit=0
```

Sample SCP output

```
# aircon refresh //BOARD/254/172 1 #sourceunit=0 OID=...
```


4.7.2.3.2.2 Set Ward On

Event

Set Ward On

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_ward_on app ward

For description of parameters see the AIRCON SET_WARD_ON command.

Sample Event output

```
20081022-084833 702 sys [aircon] set_ward_on ward=1 sourceUnit=0
```

Sample SCP output

```
# aircon set_ward_on //BOARD/254/172 1 #sourceunit=0 OID=...
```

4.7.2.3.2.3 Set Ward Off

Event

Set Ward Off

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_ward_off app ward

For description of parameters see the AIRCON SET_WARD_OFF command.

Sample Event output

```
20081022-084833 702 sys [aircon] set_ward_off ward=1 sourceUnit=0
```

Sample SCP output

```
# aircon set_ward_off //BOARD/254/172 1 #sourceunit=0 OID=...
```

4.7.2.3.2.4 Set Zone HVAC Mode

Event

Set Zone HVAC Mode

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_zone_hvac_mode app ward zone-list mode raw-flag setback-enabled guard-enabled use-aux-level type level auxlevel

For description of parameters see the AIRCON SET_ZONE_HVAC_MODE command.

Sample Event output

```
20081022-084833 702 sys [aircon] set_zone_hvac_mode ward=255 zone-list=1,2,3,4,5,6
    mode=7 rawlevel=1 setbackenabled=1 guardenabled=1 useauxlevel=1 type=255
    level=65535 auxlevel=255
```

Sample SCP output

```
# aircon set_zone_hvac_mode 254/172 255 1,2,3,4,5,6 7 1 1 1 1 255 65535 255
```

4.7.2.3.2.5 Set HVAC Upper Guard Limit

Event

Set HVAC Upper Guard Limit

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_hvac_upper_guard_limit app ward zone-list limit mode raw-level setback-enabled guard-enabled use-aux-level

For description of parameters see the AIRCON SET_HVAC_UPPER_GUARD_LIMIT command.

Sample Event output

```
20081022-084833 702 sys [aircon] set_hvac_upper_guard_limit ward=255 zone-
    list=1,2,3,4,5,6 limit=65535 mode=7 rawlevel=1 setbackenabled=1 guardenabled=1
    useauxlevel=1
```

Sample SCP output

```
# aircon set_hvac_upper_guard_limit 254/172 255 1,2,3,4,5,6 65535 7 1 1 1 1
#sourceunit=0 OID=...
```


4.7.2.3.2.6 Set HVAC Lower Guard Limit

Event

Set HVAC Lower Guard Limit

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_hvac_lower_guard_limit app ward zone-list limit mode raw-level setback-enabled guard-enabled use-aux-level

For description of parameters see the AIRCON SET_HVAC_LOWER_GUARD_LIMIT command.

Sample Event output

```
20081022-084833 702 sys [aircon] set_hvac_lower_guard_limit ward=255 zone-
    list=1,2,3,4,5,6 limit=65535 mode=7 rawlevel=1 setbackenabled=1 guardenabled=1
    useauxlevel=1
```

Sample SCP output

```
# aircon set_hvac_lower_guard_limit 254/172 255 1,2,3,4,5,6 65535 7 1 1 1 1
#sourceunit=0 OID=...
```

4.7.2.3.2.7 Set HVAC Setback Limit

Event

Set HVAC Setback Limit

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_hvac_setback_limit app ward zone-list limit mode raw-level setback-enabled guard-enabled use-aux-level

For description of parameters see the AIRCON SET_HVAC_SETBACK_LIMIT command.

Sample Event output

```
20081022-084833 702 sys [aircon] set_hvac_setback_limit ward=255 zone-
    list=1,2,3,4,5,6 limit=65535 mode=7 rawlevel=1 setbackenabled=1 guardenabled=1
```



```
    useauxlevel=1
```

Sample SCP output

```
# aircon set_hvac_setback_limit 254/172 255 1,2,3,4,5,6 65535 7 1 1 1 1
#sourceunit=0 OID=...
```

4.7.2.3.2.8 Set Zone Humidity Mode

Event

Set Zone Humidity Mode

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_zone_humidity_mode app ward zone-list mode raw-flag setback-enabled guard-enabled use-aux-level type level auxlevel

For description of parameters see the AIRCON SET_ZONE_HUMIDITY_MODE command.

Sample Event output

```
20081022-084833 702 sys [aircon] set_zone_humidity_mode ward=255 zone-
    list=1,2,3,4,5,6 mode=7 rawlevel=1 setbackenabled=1 guardenabled=1
    useauxlevel=1 type=255 level=65535 auxlevel=255
```

Sample SCP output

```
# aircon set_zone_humidity_mode 254/172 255 1,2,3,4,5,6 7 1 1 1 1 255 65535 255
```

4.7.2.3.2.9 Set Humidity Upper Guard Limit

Event

Set Humidity Upper Guard Limit

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_humidity_upper_guard_limit app ward zone-list limit mode raw-level setback-enabled guard-enabled use-aux-level

For description of parameters see the AIRCON SET_HUMIDITY_UPPER_GUARD_LIMIT command.

Sample Event output

```
20081022-084833 702 sys [aircon] set_humidity_upper_guard_limit ward=255 zone-
    list=1,2,3,4,5,6 limit=65535 mode=7 rawlevel=1 setbackenabled=1 guardenabled=1
    useauxlevel=1
```

Sample SCP output

```
# aircon set_humidity_upper_guard_limit 254/172 255 1,2,3,4,5,6 65535 7 1 1 1 1
#sourceunit=0 OID=...
```

4.7.2.3.2.10 Set Humidity Lower Guard Limit

Event

Set Humidity Lower Guard Limit

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_humidity_lower_guard_limit app ward zone-list limit mode raw-level setback-enabled guard-enabled use-aux-level

For description of parameters see the AIRCON SET_HUMIDITY_LOWER_GUARD_LIMIT command.

Sample Event output

```
20081022-084833 702 sys [aircon] set_humidity_lower_guard_limit ward=255 zone-
    list=1,2,3,4,5,6 limit=65535 mode=7 rawlevel=1 setbackenabled=1 guardenabled=1
    useauxlevel=1
```

Sample SCP output

```
# aircon set_humidity_lower_guard_limit 254/172 255 1,2,3,4,5,6 65535 7 1 1 1 1
#sourceunit=0 OID=...
```

4.7.2.3.2.11 Set Humidity Setback Limit

Event

Set Humidity Setback Limit

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_humidity_setback_limit app ward zone-list limit mode raw-level setback-enabled guard-enabled use-aux-level

For description of parameters see the AIRCON SET_HUMIDITY_SETBACK_LIMIT command.

Sample Event output

```
20081022-084833 702 sys [aircon] set_humidity_setback_limit ward=255 zone-
    list=1,2,3,4,5,6 limit=65535 mode=7 rawlevel=1 setbackenabled=1 guardenabled=1
    useauxlevel=1
```

Sample SCP output

```
# aircon set_humidity_setback_limit 254/172 255 1,2,3,4,5,6 65535 7 1 1 1 1
#sourceunit=0 OID=...
```

4.7.2.4 Commands

The following commands are supported by this application:

- AIRCON REFRESH - Sends a refresh request to an air-conditioning ward.
- AIRCON SET_HUMIDITY_SETBACK_LIMIT - Sets the error allowed in the set humidity for zones.
- AIRCON SET_HUMIDITY_LOWER_GUARD_LIMIT - Sets the absolute minimum humidity allowed in zones.
- AIRCON SET_HUMIDITY_UPPER_GUARD_LIMIT - Sets the absolute maximum humidity allowed in zones.
- AIRCON SET_HVAC_LOWER_GUARD_LIMIT - Sets the absolute minimum temperature allowed in zones.
- AIRCON SET_HVAC_SETBACK_LIMIT - Sets the error allowed in the set temperature for zones.
- AIRCON SET_HVAC_UPPER_GUARD_LIMIT - Sets the absolute maximum temperature allowed in zones.
- AIRCON SET_WARD_OFF - Switches off all plant in all the zones in the specified ward.
- AIRCON SET_WARD_ON - Returns an air-conditioning ward to its previous operational state.
- AIRCON SET_ZONE_HUMIDITY_MODE - Broadcasts Humidity mode and level required for zones.
- AIRCON SET_ZONE_HVAC_MODE - Broadcasts HVAC mode and level required for zones.

4.7.2.5 Specification

For details regarding the function of the Airconditioning Application on the C-Bus network, refer to "C-Bus Application Messages and Behaviour -- Chapter 25: Air Conditioning (CBUS-APP/25), Issue 1.12, 26 August 2008".

4.7.3 Audio Application

The Audio Application is used to manage Multi Room Audio activity.

A number of events are provided by MRA devices and a number of commands are available to control MRA devices via this application.

The Audio Application has a C-Bus application address of $CD, or 205 decimal.

4.7.3.1 Audio Application Overview

This gives a short overview of the Audio Application implementation in the C-Gate server

Application name

Audio

Application short name

(used as first part of commands or events to refer to this application)

audio

C-Bus Application range

$CD (205)

Commands

(these can be entered from the C-Gate command interface preceded by the short name of this application. All commands are also recognised as events)

off
on
ramp
terminateramp
next_feed
previous_feed
mute
dynamic_1
dynamic_2
next_language
high_priority
zone_descriptor_request
zone_feed_label_request
output_device_status_request
output_common_control
output_error_code
request_current_feed
current_feed
set_feed

Events

(these are detected by the C-Gate server and will be shown in the event list and status port but are not available as commands)

label
load_icon

Notes and implementation issues

4.7.3.2 Objects

One new object is provided for the implementation of this application:

Audio Application

This is the application object created in the server when an audio command is detected on the C-Bus network, or if an audio command is sent from the command line. This object provides the following parameters in addition to those supported by the standard Application object.

Parameter Name	Description	Read Access Control	Write Access Control
ShortName	The short name for this application. This is set to **audio** when the application is loaded	Monitor	Read Only

4.7.3.3 Events

The events in this section are recognised by the Audio application and are made available as events in the C-Gate server event port, or via the Status Change Port (SCP).

4.7.3.3.1 Current Feed

Event

Current Feed

Condition causing event

Reporting the current feed for a zone.

Event text

```
current_feed app ( multiplexer zone feed ) | ( Z function ) gain

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
feed = feed number (0..7)
function = zone function (0..255)
gain = amount of gain (0..4)
```

Sample Event output


```
20081022-084833 702 sys [audio] current_feed multiplexer=1 zone=2 feed=4 gain=0
```

Sample SCP output

```
# audio current_feed //BOARD/254/205 1 2 4 0 #sourceunit=0 OID=...
```

4.7.3.3.2 Dynamic 1

Event

Dynamic 1

Condition causing event

Setting the dynamic 1 operation for a zone.

Event text

```
dynamic_1 app ( multiplexer zone ) | ( Z function )

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Sample Event output

```
20081022-084833 702 sys [audio] dynamic_1 multiplexer=1 zone=2
```

Sample SCP output

```
# audio dynamic_1 //BOARD/254/205 1 2 #sourceunit=0 OID=...
```

4.7.3.3.3 Dynamic 2

Event

Dynamic 2

Condition causing event

Setting the dynamic 2 operation for a zone.

Event text

```
dynamic_2 app ( multiplexer zone ) | ( Z function )

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```


Sample Event output

```
20081022-084833 702 sys [audio] dynamic_2 multiplexer=1 zone=2
```

Sample SCP output

```
# audio dynamic_2 //BOARD/254/205 1 2 #sourceunit=0 OID=...
```

4.7.3.3.4 High Priority

Event

High Priority

Condition causing event

Sending a high priority operation for a zone.

Event text

```
high_priority app multiplexer level feed

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
level = level number (0..7)
feed = feed number (0..7)
```

Sample Event output

```
20081022-084833 702 sys [audio] high_priority multiplexer=1 level=2 feed=4
```

Sample SCP output

```
# audio high_priority //BOARD/254/205 1 2 4 #sourceunit=0 OID=...
```

4.7.3.3.5 Mute

Event

Mute

Condition causing event

Setting the mute operation for a zone.

Event text

```
mute app ( multiplexer zone ) | ( Z function ) mode

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
```



```
function = zone function (0..255)
mode = mode (0, 2, 5, 7, 255)
           0 = turn amplifier off
           2 = amplifier on, volume normal, speakers off
           5 = amplifier on, volume preset, speakers off
           7 = amplifier on, volume preset, speakers on
           255 = amplifier on, volume normal, speakers on
```

Sample Event output

```
20081022-084833 702 sys [audio] mute multiplexer=1 zone=2 mode=0
```

Sample SCP output

```
# audio mute //BOARD/254/205 1 2 0 #sourceunit=0 OID=...
```

4.7.3.3.6 Next Feed

Event

Next Feed

Condition causing event

The zone is being set to the next feed.

Event text

```
next_feed app ( multiplexer zone ) | ( Z function )

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Sample Event output

```
20081022-084833 702 sys [audio] next_feed multiplexer=1 zone=2
```

Sample SCP output

```
# audio next_feed //BOARD/254/205 1 2 #sourceunit=0 OID=...
```

4.7.3.3.7 Next Language

Event

Next Language

Condition causing event

The zone is being set to the next language.

Event text

```
next_language app ( multiplexer zone ) | ( Z function )

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Sample Event output

```
20081022-084833 702 sys [audio] next_language multiplexer=1 zone=2
```

Sample SCP output

```
# audio next_language //BOARD/254/205 1 2 #sourceunit=0 OID=...
```

4.7.3.3.8 Off

Event

Off

Condition causing event

An 'off' operation is performed on a zone.

Event text

```
off app ( multiplexer zone code ) | ( Z function )

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
code = function code (0..7)
function = zone function (0..255)
```

Sample Event output

```
20081022-084833 702 sys [audio] off multiplexer=1 zone=2 code=3
```

Sample SCP output

```
# audio off //BOARD/254/205 1 2 3 #sourceunit=0 OID=...
```

4.7.3.3.9 On

Event

On

Condition causing event

An 'on' operation is performed on a zone.

Event text

```
on app ( multiplexer zone code ) | ( Z function )

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
code = function code (0..7)
function = zone function (0..255)
```

Sample Event output

```
20081022-084833 702 sys [audio] on multiplexer=1 zone=2 code=3
```

Sample SCP output

```
# audio on //BOARD/254/205 1 2 3 #sourceunit=0 OID=...
```

4.7.3.3.10 Output Common Control

Event

Output Common Control

Condition causing event

A control command is sent to all output devices.

Event text

```
output_common_control control-code

control-code = control code (0)
          0 = turn off
```

Sample Event output

```
20081022-084833 702 sys [audio] output_common_control control-code=0
```

Sample SCP output

```
# audio output_common_control //BOARD/254/205 0 #sourceunit=0 OID=...
```

4.7.3.3.11 Output Device Status Request

Event

Output Device Status Request

Condition causing event

A request is sent for all output devices to return their status.

Event text

```
output_device_status_request parameter

parameter = reserved value (0)
```

Sample Event output

```
20081022-084833 702 sys [audio] output_device_status_request parameter=0
```

Sample SCP output

```
# audio output_device_status_request //BOARD/254/205 0 #sourceunit=0 OID=...
```

4.7.3.3.12 Output Error Code

Event

Output Error Code

Condition causing event

A device advertises an error condition.

Event text

```
output_error_code app multiplexer zone code

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
code = error code (0..1)
           0 = overheating
           1 = low voltage
```

Sample Event output

```
20081022-084833 702 sys [audio] output_error_code multiplexer=1 level=2 code=1
```

Sample SCP output

```
# audio output_error_code //BOARD/254/205 1 2 1 #sourceunit=0 OID=...
```


4.7.3.3.13 Previous Feed

Event

Previous Feed

Condition causing event

The zone is being set to the previous feed.

Event text

previous_feed app (multiplexer zone) | (Z function)

```
app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Sample Event output

```
20081022-084833 702 sys [audio] previous_feed multiplexer=1 zone=2
```

Sample SCP output

```
# audio previous_feed //BOARD/254/205 1 2 #sourceunit=0 OID=...
```

4.7.3.3.14 Ramp

Event

Current Feed

Condition causing event

A 'ramp' operation is in progress on a zone.

Event text

current_feed app (multiplexer zone code) | (Z function) level rate

```
app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
code = zone code (0..7)
function = zone function (0..255)
level = level to ramp to (0..255)
rate = time taken to perform the ramp (0..15)
          0 = instantaneous
          1 = 4 seconds
          2 = 8 seconds
          3 = 12 seconds
          4 = 20 seconds
```



```
5 = 30 seconds
6 = 40 seconds
7 = 1 minute
8 = 90 seconds
9 = 2 minutes
10 = 3 minutes
11 = 5 minutes
12 = 7 minutes
13 = 10 minutes
14 = 15 minutes
15 = 17 minutes
```

Sample Event output

```
20081022-084833 702 sys [audio] ramp multiplexer=1 zone=2 code=4 level=255 rate=1
```

Sample SCP output

```
# audio ramp //BOARD/254/205 1 2 4 255 1 #sourceunit=0 OID=...
```

4.7.3.3.15 Request Current Feed

Event

Request Current Feed

Condition causing event

The current feed for a zone is requested.

Event text

```
request_current_feed app ( multiplexer zone ) | ( Z function )

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Sample Event output

```
20081022-084833 702 sys [audio] request_current_feed multiplexer=1 zone=2
```

Sample SCP output

```
# audio request_current_feed //BOARD/254/205 1 2 #sourceunit=0 OID=...
```

4.7.3.3.16 Set Feed

Event

Set Feed

Condition causing event

The current feed is being set for a zone.

Event text

```
set_feed app ( multiplexer zone feed ) | ( Z function ) option

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
feed = feed number (0..7)
function = zone function (0..255)
option = value (0..1)
           0 = allow labels and annunciation on feed change
           1 = disable labels and annunciation on feed change
```

Sample Event output

```
20081022-084833 702 sys [audio] set_feed multiplexer=1 zone=2 feed=4 option=0
```

Sample SCP output

```
# audio set_feed //BOARD/254/205 1 2 4 0 #sourceunit=0 OID=...
```

4.7.3.3.17 Terminate Ramp

Event

Terminate Ramp

Condition causing event

A ramp in progres is being terminated on a zone.

Event text

```
terminateramp app ( multiplexer zone code ) | ( Z function )

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
code = function code (0..7)
function = zone function (0..255)
```

Sample Event output

```
20081022-084833 702 sys [audio] terminateramp multiplexer=1 zone=2 code=3
```


Sample SCP output

```
# audio terminateramp //BOARD/254/205 1 2 3 #sourceunit=0 OID=...
```

4.7.3.3.18 Zone Descriptor Request

Event

Zone Descriptor Request

Condition causing event

DLT devices are requested to report their zone descriptor labels.

Event text

```
zone_descriptor_request app ( multiplexer zone ) | ( Z function )

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Sample Event output

```
20081022-084833 702 sys [audio] zone_descriptor_request multiplexer=1 zone=2
```

Sample SCP output

```
# audio zone_descriptor_request //BOARD/254/205 1 2 #sourceunit=0 OID=...
```

4.7.3.3.19 Zone Feed Label Request

Event

Zone Feed Label Request

Condition causing event

DLT devices are requested to report their feed description and dynamic labels.

Event text

```
zone_feed_label_request app ( multiplexer zone ) | ( Z function )

app = audio application address (net/$CD)
multiplexer = multiplexer number (0..2)
zone = zone number (0..7)
function = zone function (0..255)
```

Sample Event output


```
20081022-084833 702 sys [audio] zone_feed_label_request multiplexer=1 zone=2
```

Sample SCP output

```
# audio zone_feed_label_request //BOARD/254/205 1 2 #sourceunit=0 OID=...
```

4.7.3.4 Commands

The following commands are supported by this application:

- AUDIO CURRENT_FEED - Reports the current feed for the given zone.
- AUDIO DYNAMIC_1 - Request a matrix switcher to send a Dynamic 1 operation in the given zone.
- AUDIO DYNAMIC_2 - Request a matrix switcher to send a Dynamic 2 operation in the given zone.
- AUDIO HIGH_PRIORITY - Request an output device to turn on and go to a set output level and feed.
- AUDIO MUTE - Set the mute mode of an amplifier.
- AUDIO NEXT_FEED - Set the next feed for the given zone.
- AUDIO NEXT_LANGUAGE - Set the next language for the given zone.
- AUDIO OFF - Send an off operation.
- AUDIO ON - Send an on operation.
- AUDIO OUTPUT_COMMON_CONTROL - Request all devices to perform a function.
- AUDIO OUTPUT_DEVICE_STATUS_REQUEST - Request status of all devices.
- AUDIO OUTPUT_ERROR_CODE - Send an error code notification.
- AUDIO PREVIOUS_FEED - Set the previous feed for the given zone.
- AUDIO RAMP - Send a ramp operation.
- AUDIO REQUEST_CURRENT_FEED - Request the current feed of a zone.
- AUDIO SET_FEED - Set the current feed of a zone.
- AUDIO TERMINATERAMP - Terminate a ramp in progress.
- AUDIO ZONE_DESCRIPTOR_REQUEST - Send feed description to DLT labelling devices.
- AUDIO ZONE_FEED_LABEL_REQUEST - Send feed description and dynamic labels to DLT labelling devices.

4.7.3.5 Specification

For details regarding the function of the Audio Application on the C-Bus network, refer to "C-Bus Application Messages and Behaviour -- Chapter 14: Air Conditioning (CBUS-APP/14)".

4.7.4 Clock and Timekeeping Application

The Clock and Timekeeping Application allows C-Gate to interact with devices that use clocks on the C-Bus Network, and also to set and get the network time as a slave or primary or secondary master clock.

4.7.4.1 Application Overview

The table gives a short overview of the Clock & Timekeeping Application implementation in the C-Gate server.

Application name

Clock and Timekeeping

Application short name (used as first part of commands or events to refer to this application)

clock

C-Bus Application range

$DF (only one)

Commands (These can be entered from C-Gate command interface preceeded by the short name of this application. All commands are also recognised as events)

date
request_refresh
time

Events (these are detected by C-Gate and will be shown in the event list and status port but are not available as commands)

Notes and implementation issues

4.7.4.2 Objects supported

One new object is provided for the implementation of this application:

Clock and Timekeeping Application

This is the application object created in the server when a clock and timekeeping command is detected on the C-Bus network or if a clock and timekeeping command is sent from the command line. This object provides the following parameters in addition to those supported by the standard Application object.

Parameter Name	Description	Read Access Control	Write Access Control
ShortName	The short name for this application. This is set to **clock** when the application is loaded	Monitor	Read Only

4.7.4.3 Methods

This application support the following methods, which mirror the commands described later:

Method Name	Description	Access Control
ClockDate	See CLOCK DATE	Operate
ClockTime	See CLOCK TIME	Operate
ClockRequestRefresh	See CLOCK REQUEST_REFRESH	Operate

4.7.4.4 Commands

The following commands are supported by this application:

- CLOCK
- CLOCK DATE
- CLOCK TIME

- CLOCK REQUEST_REFRESH

4.7.4.5 Events issued

Events are issued for the clock messages detected on a C-Bus network. They are in the form shown below:

```
20081217-100512 702 sys [clock] time 10:05:12 255 sourceUnit=0 sessionId=cmd1
commandId={none}
20081217-100512 702 sys [clock] date 2008-12-17 2 sourceUnit=0 sessionId=cmd1
commandId={none}
20081217-100512 702 sys [clock] request_refresh, sourceUnit=3 sessionId=cmd3
commandId={none}
```

4.7.4.6 Status Change Port

This application places CLOCK commands into the Status Change Port (SCP) in the same format as if they were issued as CLOCK commands.
Samples are shown below:

```
#s# clock time //BOARD/254/223 10:08:25 0 #sourceunit=3 OID=...
#s# clock date //BOARD/254/223 2008-12-17 2 #sourceunit=3 OID=...
#s# clock request_refresh //BOARD/254/223  #sourceunit=3 OID=...
```

4.7.4.7 Network time

This applications is implemented so that the time indicated on the network is kept separately from the system time maintained by the platform that the C-Gate server is running on.

When a date or time update is received from the network, C-Gate will calculate the offset from the system clock and use this offset to calculate the network time if requested. The offset will be recalculated if a command is sent from the command line, or if a command is received from the network.

#masterIn master mode (configuration property `clock.master`=yes), the time and date will be sent to the network every `clock.update-interval` minutes, subject to the protocol required in the C-Bus Application Messages and Behaviour -- Chapter 23: Clock & Timekeeping (CBUS-APP/23).

4.7.4.8 Configuration properties

The following configuration properties are used by this application:

- clock.master - turn on/off clock master behaviour
- clock.mastermode - select primary or secondary master mode
- clock.update-interval - set the update interval

4.7.4.9 Usage Notes

There are no network variables available for this application.

4.7.4.10 Specification

For detailed knowledge regarding the function of the Clock and Timekeeping Application, refer to "C-Bus Application Messages and Behaviour -- Chapter 23: Clock & Timekeeping (CBUS-APP/23)".

4.7.5 Enable Control Application

This summarizes the Enable Control Application as used by C-Gate

4.7.5.1 Enable Introduction

This document covers the implementation of the C-Bus Enable Control Application within the C-Gate server.

4.7.5.2 Enable Application Overview

The table gives a short overview of the Enable Control Application's implementation in C-Gate.

Application name Enable Control
Application short name (used as first part of commands or events to refer to this application) enable
C-Bus Application range $CB
Commands (these can be entered from C-Gate command interface preceeded by the short name of this application. All commands are also recognised as events) set remove (see notes below)
Events (these are detected by C-Gate and will be shown in the event list and status port but are not available as commands)
Notes and implementation issues C-Gate to keep an on-disk copy of any enable control variables that are in use for a particular project/network/application. This to be turned on/off by the `enable.save-state` configuration parameter. **remove** command is used to clear the C-Gate server's memory of a particular variable. This will erase any value held (delete the network variable?) and remove any disk entry for it.

4.7.5.3 Persistent storage

If the `enable.save-state` configuration parameter is set (set to `yes`, the default value), then this application will store the value of all network variables in this application to disk, and will re-set these values when the server is restarted (eg. after power failure or manual restart).

One file is stored for each network variable value. This file contains a single number represented as a serials of ASCII decimal digits, either terminated by whitespace (including CR or LF, space or other whitespace) or the end of the file.

The files are contained in a new directory structure under the projects directory, in a subdirectory named `app/enable`. Files are names as follows `<project>-<net>-<application>-<netvar>` (the elements of the normal address of the network variable, with the '/' path symbols replaces by the '-' dash character.)

For example, if the projects directory is `C:\Clipsal\C-Gate\tag` and the network variable is `//system/1/203/25` then the filename used to hold the state of the variable would be `C:\Clipsal\C-Gate\tag\app\enable\system-1-203-25` .

4.7.5.4 Objects supported

Two new objects are provided for the implementation of this application:

4.7.5.4.1 Enable Control Application Object

This is the application object created in the server when an enable control application is detected on the C-Bus network. This object provides the following parameters in addition to those supported by the standard Application object.

Parameter Name Description Read Access Control Write Access Control
ShortName The short name for this application. This is set to **`enable`** when the application is loaded Monitor Read Only

4.7.5.4.2 Enable Network Variables

4.7.5.5 Commands supported

The following commands are supported by this application:

- ENABLE
- ENABLE SET
- ENABLE REMOVE

4.7.5.6 Events issued

Events are issued for the enable set messages detected on a C-Bus network. They are in the form show below:

```
20020910-172843.078 702 //system/1/202/144 [enable] set value=2 sourceUnit=131
```

In the example shown, the address given is the enable network variable that was activated, the command given is `event`, the `action=` paramter gives the action selector value, and the `sourceUnit=` gives the source unit id on the network where the command was sent.

The ENABLE REMOVE command does not cause an event to be issued.

4.7.5.7 Status Change Port Reporting

This application places ENABLE SET commands into the Status Change Port (SCP).

A sample are shown below.

```
enable set //system/1/203/144 2 #sourceunit=131
```

The enable REMOVE command does not cause a SCP entry.

4.7.5.8 Project database

This application does **not** support the <Group> element under <Application>. Instead, it supports the <NetVar> element which can contain zero or more <Level> elements containing Level Tag information.

To use level tags:

Add one or more level entries to the <NetVar> element in the tag database. For each of the NetVar elements, add a <Value> element with the appropriate application selector level in the range 0 through 255.

So, an except would look like:

```
<NetVar>
    <TagName>Switch1</TagName>
    <Level>
        <TagName>AllOff</TagName>
        <Value>0</Value>
    </Level>
    <Level>
        <TagName>Level1</TagName>
        <Value>2</Value>
    </Level>
</NetVar>
```

This allows the ENABLE SET command to use the tags AllOff and Level1 as action selector values, for example:


```
enable set Switch1 AllOff
```

4.7.5.9 Usage Notes

Network Variables

Network variables are created automatically when events for them are received, or they are detected in other ways.

4.7.5.10 Enable Specification

For detailed knowledge regarding the function of the Enable Control Application, refer to "C-Bus Application Messages and Behaviour -- Chapter 8: Enable Control (CBUS-APP/08)".

4.7.6 Error Reporting Application

This section of the manual covers the implementation of the Error Reporting Application ($CE) in the C-Gate server.

4.7.6.1 Application Overview

Application name

Error Reporting

Application short name (used as first part of commands or events to refer to this application)

ereport

C-Bus Application range

$CE (206)

Commands/Events (these can be entered from the C-Gate command interface preceded by the short name of this application. All commands are also recognised as events)

message

Events (these are detected by the C-Gate server and will be shown in the event list and status port but are not available as commands)

Notes and implementation issues

See the full documentation of this application to cover operation and use of this application.

4.7.6.2 Objects Supported

One new object is provided for the implementation of this application:

Error Reporting Application

This is the application object created in the server when an ereport command is detected on the C-Bus network or if an ereport command is sent from the command line. This object provides the following parameters in addition to those supported by the standard Application object.

Parameter Name	Description	Read Access Control	Write Access Control
ShortName	The short name for this application. This is set to `ereport` when the application is loaded	Monitor	Read Only

4.7.6.3 Send Event

Event

Send Event

Condition causing event

An Error Report has been received.

This event can be generated by the EREPORT MESSAGE command.

Event Text

```
send app category most-recent acknowledged most-severe severity unit-id [<data-
byte-1 [data-byte-2]]
```

Sample Event output

```
20060111-160154 702 //cis/1/206 37be1ec0-0d69-1028-a7c5-e25002b8b622 [ereport]
ereport message 254/$CE ACK 1023 n n n 7 255 255 255
```

Sample SCP output

```
ereport message 254/$CE ACK 1023 n n n 7 255 255 255
```

4.7.7 Measurement Application

This document covers the implementation of the C-Bus Measurement Application within the C-Gate server.

The Measurement Application is used by C-Bus devices that report analog measurements. Each measurement comes from a Channel, with Channels grouped together in Measurement Devices.

4.7.7.1 Specification

For detailed knowledge regarding the function of the Measurement Application on the C-Bus network, refer to "C-Bus Application Messages and Behavior -- Chapter 28: Measurement (CBUS-APP/28), Issue 1, 7 August 2002".

4.7.7.2 Application Overview

This gives a short overview of the Media Transport Control Application implementation in the C-Gate server

Application name

Measurement

Application short name (used as first part of commands or events to refer to this application)

measurement

C-Bus Application range

$E4 (228)

Commands/Events (these can be entered from the C-Gate command interface preceded by the short name of this application. All commands are also recognised as events)

data

Events (these are detected by the C-Gate server and will be shown in the event list and status port but are not available as commands)

data

Notes and implementation issues

See the full documentation of this application to cover operation and use of this application.

4.7.7.3 Objects Supported

There are three new objects supported by the Measurement Application implementation.

The Measurement Application Object models the application as a whole. The Measurement Application Object is addresssed using the application address for the Measurement Application ($E4 or 228) on the C-Bus network that the measurement devices are connected to.

The Measurement Device Object models a measurement device. The measurement device is addressed as a sub-address of the Measurement Application that it is associated with.

Each measurement device can have one or more Channels, up to a total of 256 Channels per Measurement Device.

4.7.7.3.1 Measurement Application Object

The Measurement Application Object models a C-Bus network's Measurement Application.

The Measurement Application Object supports up to 256 Measurement Device Objects.

Addressing

The Measurement Application Object is addresssed using the standard application address for the Measurement Application (`$E4` or decimal `228`) on the relevant C-Bus network.

For example, for a hypothetical network called `net1`, the Measurement Application's address will be `net1/$E4` or `net1/228`

4.7.7.3.1.1 Commands Supported

The following command line commands are accepted by the Measurement Application Object.

Command
MEASUREMENT ?

Syntax
```
"MEASUREMENT" ["?"]
```

Use
Use this command to get help information about the MEASUREMENT commands. Help information is returned as a number of lines of help information.

Success Response

A series of lines giving help for these commands in the form:
```
101 Help: help-information
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.7.7.3.1.2 Events Supported

There are no events supported by this object.

4.7.7.3.1.3 Object Parameters

The following parameters are supported by the Measurement Application Object.

They can be accessed with the GET command acting on the object address of the relevant Measurement Application object.

Parameter Name	Type & Description	Can set with SET command?	Notes
State	String. The overall state of this object. Valid states are: "ok" = normal operation "new" = the object has been created and has yet to synchronize with the C-Bus network "sync" = the object is synchronising with the C-Bus network "error" = an error has occurred.	No	
Devices	Returns a list of device numbers that are known in this application. The list items are separated by commas	No	

4.7.7.3.2 Measurement Device Object

The Measurement Device object models a device that is capable of making measurements. The meaurement device can have 0 or more Channels, where each channel can provide measurement data.

Addressing Measurement Device Objects

A Measurement Application can support up to 256 separate Measurement Devices.

Measurement Device Objects are addressed as subelements of the Measurement Application Object. Valid Measurement Device addresses range from `0` through `255` (or in hexadecimal, `$00` through `$FF`). No other values are are allowed or valid.

So, on an example network called `net1`, the Measurement Application would be addressed as `net1/$E4` or `net1/228`. Measurement device number 1 would be addressed as `net1/228/1` , and Measurement Device $25 would be addressed as `net1/$E4/$25` or `net1/228/$25`.

4.7.7.3.2.1 Commands Supported

There are no commands supported by the Measurement Device Object.

4.7.7.3.2.2 Events Supported

There are no events supported by this object.

4.7.7.3.2.3 Object Parameters

The following parameters are supported by the Measurement Device Object.

They can be accessed with the GET command acting on the object address of the relevant Measurement Device object.

Parameter Name	Type & Description	Can be set with SET command?	Notes
State	String. The overall state of this object. Valid states are: "ok" = normal operation "new" = the object has been created and has yet to synchronise with the C-Bus network "sync" = the object is attempting to synchronise with the C-Bus network "error" = an error has occurred.	No	
Channels	A comma-separated list of channel numbers that are known on this Measurement Device.	No	

4.7.7.3.3 Channel Object

The Channel Object models a single measurement channel on a Measurement Device.

Addressing Channel Objects

Channel addresses are in the range of 0 through 255.

Channel Objects are addressed as subelements of the Measurement Device Object. Valid Channel addresses range from `0` through `255` (or in hexadecimal, `$00` through `$FF`). No other values are are allowed or valid.

So, on an example network called `net1`, the Measurement Application would be addressed as `net1/$E4` or `net1/228`. Measurement device number 1 would be addressed as `net1/228/1`. Channel 2 on Measurement Device 1 would be addressed as: `net1/$E4/1/2 or net1/228/1/2` .

4.7.7.3.3.1 Commands Supported

The following commands are supported by this object

Command

MEASUREMENT DATA

Syntax

```
"MEASUREMENT" "DATA" channel-address meaurement-value exponent units

channel-address =  ;the address of the measurement channel to send the data for
                       ;as a C-Gate address, for example net1/228/1/1

measurement-value = ; integer part of the measurement (mantissa)

exponent = power of ten to be applied to the measement value

units = an integer giving the units of the measurement value.

See the unit code table for conversion details.
```

Use

Use this command to send measurement data to the C-Bus network.

Success Response

```
200 OK.
```

Failure Responses

```
401 Bad object or device ID
402 Not supported by this object
405 Parameter out of range
408 Operation failed
420 Access denied
```

4.7.7.3.3.2 Events Supported

The following events are supported by the Channel Object.

Event

Measurement Data

Condition causing event

Measurement data has been received. This event can be generated by the MEASUREMENT DATA command.

Event Text

```
"data" measurement-data exponent units

measurement-data = ; integer measurement value

exponent = power of ten to be applied to the measement value

units = an integer giving the units of the measurement value. See the unit code table for conversion details.
```

Sample Event output

```
20020910-172843 702 //system/1/228/1/1 [measurement] data 10234 -2 2
sourceUnit=100
```

Sample SCP output

```
measurement data //system/1/228/1/1 10234 -2 2  #sourceunit=100
```

4.7.7.3.3.3 Object Parameters

he following parameters are supported by the Channel Object.

They can be accessed with the GET command acting on the object address of the relevant Channel object.

Parameter Name	Type & Description	Can be set with SET command?	Notes
State	String. The overall state of this object. Valid states are: "ok" = normal operation "new" = the object has been created and has yet to synchronise with the C-Bus network "sync" = the object is attempting to synchronise with the C-Bus network "error" = an error has occurred.	No	
Data	Four numbers, separated by commas that give the following information from the last measurement received for this channel. The four numbers are: 1. the integer portion of the	No	If no measurement has been received, the this parameter will be set to "0,0,0,-1"

	measurement 2. the exponent (power of 10) to be applied to the integer portion of the measurement 3. the unit code for this measurement 4. the number of milliseconds since this measurement was taken (if set to -1, indicates that no measurement has been received for this channel)		

4.7.7.4 Events Issued

The events in this section are generated by the Measurement application and are made available as events in the C-Gate server event port, or via the Status Change Port (SCP)

4.7.7.4.1 Data

Event

Data

Condition causing event

Receiving a MEASUREMENT DATA command sequence.

Event text

```
[measurement] data device channel units multiplier value sourceUnit

device = measurement device (0..$FF)
channel = measurement channel (0..$FF)
units = measurement units (0..$FF)
multiplier = power of ten unit multiplier (-128..127)
value = scaled measurement value (-32768..32767)
sourceUnit = source of the command

For meanings of units values see the Unit Code Table.
```

Sample Event output

```
20090220-173207 702 //BC/254/228 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[measurement] data device=1 channel=2 units=0 multiplier=1 value=37
sourceUnit=3
```

Sample SCP output

```
#s# measurement data //BC/254/228 1 2 0 1 37 #sourceunit=3 OID=...
```

4.7.7.5 Unit Code Table

The following table gives the relation between unit codes (from the MEASUREMENT DATA command and Measurement Data event), unit names and typical uses.

Unit Code	Units	Typical Use
$00	°C	Temperature
$01	Amps	Current
$02	Angle (degrees)	Angular displacement
$03	Coulomb	(Electric) charge
$04	False = 0 True otherwise	Boolean stuff
$05	Farads	Capacitance
$06	Henrys	Inductance
$07	Hertz	Frequency
$08	Joules	Energy
$09	Katal	Rate of catalytic activity
$0A	Kg / m^3	Density
$0B	Kilograms	Mass
$0C	Litres	Volume
$0D	Litres per hour	Very slow flow rates
$0E	Litres per minute	Slow flow rate
$0F	Litres per second	Flow rate
$10	Lux	Light level
$11	Metres	Distance
$12	Metres per minute	Slow speed
$13	Metres per second	Speed
$14	$Metres/s^2$	Acceleration
$15	Mole	Quantity of substance
$16	Newton metre	Torque
$17	Newtons	Force
$18	Ohms	Resistance
$19	Pascal	Pressure
$1A	Percent	Humidity, generic percentages & linear ratios
$1B	Decibels	Logarithmic ratio
$1C	PPM	Concentrations
$1D	RPM	Angular speed
$1E	Second	Elapsed Time
$1F	Minutes	Elapsed Time
$20	Hours	Elapsed Time
$21	Sieverts	Radiation
$22	Steradian	Units of solid angle
$23	Tesla	Magnetic field strength
$24	Volts	Voltage
$25	Watt hours	Power consumption
$26	Watts	Power
$27	Webers	Magnetic Flux
$FE	No units	Unitless quantities
$FF	Custom	User defined

4.7.8 Media Transport Control Application

The Media Transport Control Application is used to control audio and video equipment attached to, or used with, C-Bus equipment.
This can include recorders, players, tuners, and other audio and video media equipment.

The Media Transport Control Application has a C-Bus application address of $C0, or 192 decimal.

4.7.8.1 Application Overview

This gives a short overview of the Media Transport Control Application implementation in the C-Gate server

Application name

Airconditioning

Application short name

(used as first part of commands or events to refer to this application)

mediatransport

C-Bus Application range

$C0 (192)

Commands

(these can be entered from the C-Gate command interface preceded by the short name of this application. All commands are also recognised as events)

category_name
enumerate
enumeration_size
forward
next_category
next_selection
next_track
pause
play
repeat
rewind
selection_name
set_category
set_selection
set_track
shuffle
source_power
status_request
stop
track_name

Events

(these are detected by the C-Gate server and will be shown in the event list and status port)

category_name
enumerate
enumeration_size
forward

next_category
next_selection
next_track
pause
play
repeat
rewind
selection_name
set_category
set_selection
set_track
shuffle
source_power
status_request
stop
track_name

Notes and implementation issues

4.7.8.2 Objects Supported

An object is provided for the implementation of this application:

Media Transport Control Application

This is the application object created in the server when a Media Transport Control command is detected on the C-Bus network, or if a Media Transport Control command is sent from the command line. This object provides the following parameters in addition to those supported by the standard Application object.

Parameter Name	Description	Read Access Control	Write Access Control
Name	The name for this application.	Monitor	Read/Write
ShortName	The short name for this application. This is set to **`mediatransport`** when the application is loaded	Monitor	Read Only

4.7.8.3 Events Issued

The events in this section are generated by the Media Transport Control application and are made available as events in the C-Gate server event port, or via the Status Change Port (SCP)

4.7.8.3.1 Category Name

Event

Category Name

Condition causing event

Receiving a MEDIATRANSPORT CATEGORY_NAME command sequence.

Event text

```
[mediatransport] category_name group wni total index text sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
wni = name identification (0..7)
   0 = current category
   1 = next category
   2 = next+1 category
   3,4 = reserved
   5 = previous category
   6 = previous-1 category
   2 = enumerated category
total = total number of packets - 1 (0..3)
index = index of this packet in the sequence (0..3)
text = all/part of the category name in UTF-8 (max 11 bytes per packet)
sourceUnit = source of the command
```

Sample Event output

```
20090220-173207 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] category_name group=2 wni=1 total=1 sequence=0 text=list
\x204 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport category_name //BC/254/192 2 1 1 0 list\x204
#sourceunit=3 OID=...
```

4.7.8.3.2 Enumerate

Event

Enumerate

Condition causing event

Receiving a MEDIATRANSPORT ENUMERATE command sequence.

Event text

```
[mediatransport] enumerate group type start sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
type = enumeration type where 0=category, 1=selection, 2=track
start = enumerate from this index (0..$FF)
sourceUnit = source of the command
```

Sample Event output


```
20090220-164033 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] enumerate group=2 type=1 start=0 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport enumerate //BC/254/192 2 1 0 #sourceunit=3 OID=...
```

4.7.8.3.3 Enumeration Size

Event

Enumeration Size

Condition causing event

Receiving a MEDIATRANSPORT ENUMERATION_SIZE command sequence.

Event text

```
[mediatransport] enumeration_size group type start size sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
type = enumeration type where 0=category, 1=selection, 2=track
start = enumerate from this index (0..$FF)
size = number of items in the enumeration following (0..$0F)
sourceUnit = source of the command
```

Sample Event output

```
20090220-164033 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] enumeration_size group=2 type=1 start=0 size=4
sourceUnit=3
```

Sample SCP output

```
#s# mediatransport enumeration_size //BC/254/192 2 1 0 4 #sourceunit=3
OID=...
```

4.7.8.3.4 Forward

Event

Forward

Condition causing event

Receiving a MEDIATRANSPORT FORWARD command sequence.

Event text

```
[mediatransport] forward group operation sourceUnit
```



```
group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0..$FF):
   0 = cease fast-forward, play at normal speed
   2 = fast-forward at 2x speed
   4 = fast-forward at 4x speed
   6 = fast-forward at 8x speed
   8 = fast-forward at 16x speed
  10 = fast-forward at 32x speed
  12 = fast-forward at 64x speed
  all other values reserved
sourceUnit = source of the command
```

Sample Event output

```
20090220-100958 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] forward group=0 operation=0 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport forward //BC/254/192 0 0 #sourceunit=3 OID=...
```

4.7.8.3.5 Next Selection

Event

Next Selection

Condition causing event

Receiving a MEDIATRANSPORT NEXT_SELECTION command sequence.

Event text

```
[mediatransport] next_selection group operation sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0..$FF):
   0 = select previous selection
   non-0 = select next selection
sourceUnit = source of the command
```

Sample Event output

```
20090220-164033 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] next_selection group=2 operation=1 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport next_selection //BC/254/192 2 1 #sourceunit=3 OID=...
```

4.7.8.3.6 Next Category

Event

Next Category

Condition causing event

Receiving a MEDIATRANSPORT NEXT_CATEGORY command sequence.

Event text

```
[mediatransport] next_category group operation sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0..$FF):
   0 = select previous category
   non-0 = select next category
sourceUnit = source of the command
```

Sample Event output

```
20090223-085323 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] next_category group=2 operation=1 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport next_category //BC/254/192 2 1 #sourceunit=3 OID=...
```

4.7.8.3.7 Next Track

Event

Next Track

Condition causing event

Receiving a MEDIATRANSPORT NEXT_TRACK command sequence.

Event text

```
[mediatransport] next_track group operation sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0..$FF):
   0 = select previous track
   non-0 = select next track
sourceUnit = source of the command
```

Sample Event output

```
20090220-171413 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] next_track group=2 operation=1 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport next_track //BC/254/192 2 1 #sourceunit=3 OID=...
```


4.7.8.3.8 Pause

Event

Pause

Condition causing event

Receiving a MEDIATRANSPORT PAUSE command sequence.

Event text

```
[mediatransport] pause group operation sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0..$FF):
   0 = pause track
   255 = resume playing track
sourceUnit = source of the command
```

Sample Event output

```
20090220-171841 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] pause group=2 operation=0 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport pause //BC/254/192 2 0 #sourceunit=3 OID=...
```

4.7.8.3.9 Play

Event

Play

Condition causing event

Receiving a MEDIATRANSPORT PLAY command sequence.

Event text

```
[mediatransport] play group sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
sourceUnit = source of the command
```

Sample Event output

```
20090220-172356 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] play group=2 sourceUnit=3
```

Sample SCP output


```
#s# mediatransport play //BC/254/192 2 #sourceunit=3 OID=...
```

4.7.8.3.10 Repeat

Event

Repeat

Condition causing event

Receiving a MEDIATRANSPORT REPEAT command sequence.

Event text

```
[mediatransport] repeat group operation sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0,$FF):
   0 = repeat is off
   1..254 = repeat current track
   255 = repeat all tracks
sourceUnit = source of the command
```

Sample Event output

```
20090220-174549 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] repeat group=2 operation=1 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport repeat //BC/254/192 2 1 #sourceunit=3 OID=...
```

4.7.8.3.11 Rewind

Event

Rewind

Condition causing event

Receiving a MEDIATRANSPORT REWIND command sequence.

Event text

```
[mediatransport] rewind group operation sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0,$FF):
   0 = cease rewind, play at normal speed
   2 = rewind at 2x speed
   4 = rewind at 4x speed
   6 = rewind at 8x speed
   8 = rewind at 16x speed
```



```
  10 = rewind at 32x speed
  12 = rewind at 64x speed
sourceUnit = source of the command
```

Sample Event output

```
20090220-175000 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] rewind group=2 operation=0 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport rewind //BC/254/192 2 0 #sourceunit=3 OID=...
```

4.7.8.3.12 Selection Name

Event

Selection Name

Condition causing event

Receiving a MEDIATRANSPORT SELECTION_NAME command sequence.

Event text

```
[mediatransport] selection_name group wni total index text sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
wni = name identification (0..7)
   0 = current selection
   1 = next selection
   2 = next+1 selection
   3,4 = reserved
   5 = previous selection
   6 = previous-1 selection
   2 = enumerated selection
total = total number of packets - 1 (0..3)
index = index of this packet in the sequence (0..3)
text = all/part of the selection name in UTF-8 (max 11 bytes per packet)
sourceUnit = source of the command
```

Sample Event output

```
20090220-172714 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] selection_name group=2 wni=1 total=1 sequence=0
text=list\x204 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport selection_name //BC/254/192 2 1 1 0 list\x204
#sourceunit=3 OID=...
```


4.7.8.3.13 Set Selection

Event

Set Selection

Condition causing event

Receiving a MEDIATRANSPORT SET_SELECTION command sequence.

Event text

```
[mediatransport] set_selection group selection sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
selection = number of selection selected
sourceUnit = source of the command
```

Sample Event output

```
20090220-175233 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] select_selection group=2 selection=1 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport select_selection //BC/254/192 2 1 #sourceunit=3
OID=...
```

4.7.8.3.14 Set Category

Event

Set Category

Condition causing event

Receiving a MEDIATRANSPORT SET_CATEGORY command sequence.

Event text

```
[mediatransport] set_category group category sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
category = number of category selected
sourceUnit = source of the command
```

Sample Event output

```
20090220-175448 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] select_category group=2 category=5 sourceUnit=3
```

Sample SCP output


```
#s# mediatransport select_category //BC/254/192 2 5 #sourceunit=3 OID=...
```

4.7.8.3.15 Set Track

Event

Set Track

Condition causing event

Receiving a MEDIATRANSPORT SET_TRACK command sequence.

Event text

```
[mediatransport] set_track group track sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
track = number of track selected
sourceUnit = source of the command
```

Sample Event output

```
20090220-175822 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] set_track group=2 track=7 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport set_track //BC/254/192 2 7 #sourceunit=3 OID=...
```

4.7.8.3.16 Shuffle

Event

Shuffle

Condition causing event

Receiving a MEDIATRANSPORT SHUFFLE command sequence.

Event text

```
[mediatransport] shuffle group operation sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
operation = one of (0,$FF):
   0 = shuffle is off
   255 = shuffle is on
sourceUnit = source of the command
```

Sample Event output

```
20090220-175956 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] shuffle group=2 operation=255 sourceUnit=3
```


Sample SCP output

```
#s# mediatransport shuffle //BC/254/192 2 255 #sourceunit=3 OID=...
```

4.7.8.3.17 Source Power Control

Event

Source Power Control

Condition causing event

Receiving a MEDIATRANSPORT SOURCE_POWER command sequence.

Event text

```
[mediatransport] source_power group operation sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
operation = one of 0 or 1..255
   0 = power off
   not 0 = power on
sourceUnit = source of the command
```

Sample Event output

```
20090220-173847 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] source_power group=2 operation=255 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport source_power //BC/254/192 2 255 #sourceunit=3 OID=...
```

4.7.8.3.18 Status Request

Event

Status Request

Condition causing event

Receiving a MEDIATRANSPORT STATUS_REQUEST command sequence.

Event text

```
[mediatransport] status_request group sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
sourceUnit = source of the command
```

Sample Event output


```
20090220-174237 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] status_request group=2 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport status_request //BC/254/192 2 #sourceunit=3 OID=...
```

4.7.8.3.19 Stop

Event

Stop

Condition causing event

Receiving a MEDIATRANSPORT STOP command sequence.

Event text

```
[mediatransport] stop group sourceUnit

group = media link group (0..$FF, where $FF indicates unused)
sourceUnit = source of the command
```

Sample Event output

```
20090220-174149 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] stop group=2 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport stop //BC/254/192 2 #sourceunit=3 OID=...
```

4.7.8.3.20 Track Name

Event

Track Name

Condition causing event

Receiving a MEDIATRANSPORT TRACK_NAME command sequence.

Event text

```
[mediatransport] track_name group which total index text

group = media link group (0..$FF, where $FF indicates unused)
wni = name identification (0..7)
   0 = current track
   1 = next track
```



```
   2 = next+1 track
   3,4 = reserved
   5 = previous track
   6 = previous-1 track
   2 = enumerated track
total = total number of packets - 1 (0..3)
index = index of this packet in the sequence (0..3)
text = all/part of the track name in UTF-8 (max 11 bytes per packet)
sourceUnit = source of the command
```

Sample Event output

```
20090220-144729 702 //BC/254/192 fb95a8a0-e046-102b-a64e-ce2810e2f0a5
[mediatransport] track_name group=2 wni=0 total=0 sequence=0 text=track
\x203 sourceUnit=3
```

Sample SCP output

```
#s# mediatransport track_name //BC/254/192 2 0 0 0 track\x203
#sourceunit=3 OID=...
```

4.7.9 Security Application

The Security Application is used by security devices, such as security panels, that are connected to a C-Bus network.

A number of events are provided by security devices and a number of commands are available to control security devices via the Security Application.

The Security Application has a C-Bus application address of $DO, or 208 decimal.

4.7.9.1 Security Application Overview

The table gives a short overview of the Security Application implementation in the C-Gate server

Application name	Security
Application short name (used as first part of commands or events to refer to this application)	security
C-Bus Application range	$D0
Commands (these can be entered from C-Gate command interface preceded by the short name of this application. All commands are also recognised as events)	status_request arm tamper [raise | drop] raise_alarm emulate_keypad display_message request_zone_name

Events (these are detected by C-Gate and will be shown in the event list and status port but are not available as commands)	system_armed system_disarmed exit_delay_started alarm_on alarm_off tamper_on tamper_off panic_cleared panic_activated zone_unsealed zone_sealed zone_open zone_short zone_isolated low_battery_detected low_battery_corrected battery_charging zone_name status_report_1 status_report_2 password_entry_status mains_failure mains_restored arm_ready arm_not_ready
Notes and implementation issues	C-Gate is not capable of behaving as a security panel.

4.7.9.2 Objects Supported

There are two new object supported by the Security Application implementation.

The Security Application Object models security and a security panel as a whole. The security application object is addresssed using the application address for the Security Application ($D0 or 208) on the C-Bus network that the security system is connected to.

The Security Zone Object models one security zone. There can, and generally will be more than one security zone. The security zone is addressed as a sub-address of the Security application that it is associated with.

4.7.9.2.1 Security Application Object

The Security Application Object models a C-Bus network's Security Application as a whole. Generally, one or more security devices or panels are connecte to a C-Bus network and they communuicate using the Security Application messaging.

The Security Applicatin Object receives and models Security Application messages (**events**) and provides the ability to send Security Application messages from the C-Gate server using a number of **commands**.

The Security Application Object also supports a number of **parameters** that can be accessed with the GET and SET commands, quoting the Security Application Object's address.

Addressing

The Security Application Bbject is addresssed using the standard application address for the Security Application (`$D0` or `208`) on the C-Bus network that the security system is connected to.

For example, for a hypothetical network called `net1`, the Security Application's address will be `net1/$D0` or `net1/208`.

4.7.9.2.1.1 Commands Supported

The following commands are supported by the Security Application object:

- SECURITY
- SECURITY ARM
- SECURITY DISPLAY_MESSAGE
- SECURITY EMULATE_KEYPAD
- SECURITY RAISE_ALARM
- SECURITY STATUS_REQUEST
- SECURITY TAMPER

4.7.9.2.1.2 Events Supported

Event
System Armed/Disarmed

Condition causing event

The security system has just become armed. The type of arming is indicated in the `type` parameter.

Event Text

```
"system_arm" type

type = 1*3DIGIT      ; in the range of 1 - 127.  values 128-255 are reserved.
                     ; 0 = disarmed, 1=fully armed, 2=partially armed
                     ; 3 - 127 are manufacturer dependent
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] system_armed 1 sourceUnit=100
```

Sample SCP output

```
#security system_armed //system/1/224 1 #sourceunit=100
```

Event
System Disarmed

Condition causing event

The security system has just become disarmed. This is equivalent to a **System Armed/ Disarmed** condition with an argument of 0.

Event Text

```
"system_disarmed"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] system_disarmed sourceUnit=100
```

Sample SCP output

```
#security system_disarmed //system/1/224 #sourceunit=100
```

Event
Exit Delay Started

Condition causing event

The security system has commenced its exit delay processing.

Event Text

```
"exit_delay_started"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] exit_delay_started sourceUnit=100
```

Sample SCP output

```
#security exit_delay_started //system/1/224 #sourceunit=100
```

Event
Entry Delay Started

Condition causing event

The security system has commenced its entry delay processing.

Entry delay processing will normally commence in a security system when it detects a zone becoming unsealed in some defined entry path. The Entry Delay allows time for disarming the system. If the system is not disarmed during the Entry Delay period, the system will normally raise an alarm condition.

Event Text

```
"entry_delay_started"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] entry_delay_started sourceUnit=100
```


Sample SCP output

```
#security entry_delay_started //system/1/224 #sourceunit=100
```

Event
Alarm On

Condition causing event

The security system has commenced some alarm or notification activity.This message may optionally be followed by an Alarm Type message, to convey additional information about the alarm condition.

Event Text

```
"alarm_on"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] alarm_on sourceUnit=100
```

Sample SCP output

```
#security alarm_on //system/1/224 #sourceunit=100
```

Event
Alarm Off

Condition causing event

The security system alarm has been switched off. An Alarm Off message implies that any alarm condition, tamper condition or panic condition has been cleared

Event Text

```
"alarm_off"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] alarm_off sourceUnit=100
```

Sample SCP output

```
#security alarm_off //system/1/224 #sourceunit=100
```

Event
Tamper Off

Condition causing event

The security system has detected clearing of the tampering.A security system is expected to continue an alarm condition if tampering is removed. Some security systems only allow a tamper condition to be cleared by Disarming.

This event may not be implemented by some security systems.

Event Text

```
"tamper_off"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] tamper_off sourceUnit=100
```

Sample SCP output

```
#security tamper_off //system/1/224 #sourceunit=100
```

Event
Tamper On

Condition causing event

The security system alarm has detected tampering becoming active.The security system may, at the discretion of the manufacturer or installer, cause an alarm condition if tampering is detected. In this case, an Alarm On would be used.

Event Text

```
"tamper_on"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] tamper_on sourceUnit=100
```

Sample SCP output

```
#security tamper_on //system/1/224 #sourceunit=100
```

Event
Current Alarm Type

Condition causing event

The security system uses this event to describe the type of the active alarm. This alarm will have been presented in the previous message.

Event Text

```
"current_alarm_type" type
type = 1*3DIGIT
       ; 0 = reserved
       ; 1 = intruder
       ; 2 = line cut
       ; 3 = arm failed
       ; 4 = fire
       ; 5 = gas
       ; 6 - 127 are reserved
       ; 128 - 254 used at alarm manufacturer's discretion
       ; 255 - reserved
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] current_alarm_type 1 sourceUnit=100
```

Sample SCP output

```
#security alarm_off //system/1/224 1 #sourceunit=100
```

Event
Panic Cleared

Condition causing event

The security system has detected cancellation of the panic condition. A security system may cancel an alarm condition if panic is cancelled. (Note: Some security systems only allow a panic condition to be cleared by Disarming)

Event Text

```
"panic_cleared"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] panic_cleared sourceUnit=100
```

Sample SCP output

```
#security panic_cleared //system/1/224 #sourceunit=100
```

Event
Panic Activated

Condition causing event

The security system has detected operation of the panic button.

Event Text

```
"panic_activated"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] panic_activated sourceUnit=100
```

Sample SCP output

```
#security panic_activated //system/1/224 #sourceunit=100
```

Event
Low Battery Corrected

Condition causing event

The security system has detected that its backup battery **was** running low, and is now acceptable

This event may not be supported by some security systems.

If supported, the security system will emit this message when the battery was previously running low (with less than 1 hour capacity left), but due to some corrective action the battery is now acceptable again.

Event Text

```
"low_battery_corrected"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] low_battery_corrected sourceUnit=100
```

Sample SCP output

```
#security low_battery_corrected //system/1/224 #sourceunit=100
```

Event
Low Battery Detected

Condition causing event

The security system has detected that its backup battery is running low.

This event may not be supported by some security systems.

If supported, the security system will emit this message when the battery has less that one hour of capacity left.

Event Text

```
"low_battery_detected"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] low_battery_detected sourceUnit=100
```

Sample SCP output

```
#security low_battery_detected //system/1/224 #sourceunit=100
```

Event
Battery Charging

Condition causing event

The security system has started or stopped charging its battery. The argument indicates if the battery charging has `stopped` or `started`.

This event may not be supported by some security systems.

Event Text

```
"battery_charging" charge-mode

charge-mode = "stopped" | "started"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] battery_charging started
sourceUnit=100
```

Sample SCP output

```
#security battery_charging //system/1/224 started #sourceunit=100
```

Event
Status Report 1

Condition causing event

The security system reports its current state using this and the **Status Report 2** event. This may result from the security system receiving a **SECURITY STATUS_REQUEST** command for Status Report 1.

Event Text


```
"status_report_1" armed-state tamper-state panic-state zone-report

armed-state = 1*3DIGIT     ; in the range of 1 - 127.  values 1, 128-255 are
reserved.
                           ; 0 = disarmed, 1=fully armed, 2=partially armed
                           ; 3 - 127 are manufacturer dependent


tamper-state = 1*3DIGIT    ; 0 = no tamper active
                           ; 255 = tamper currently active
                           ; 1-254 = reserved

panic-state = 1*3DIGIT     ; 0 = no panic active
                           ; 255 = panic currently active
                           ; 1-254 = reserved

zone-report = 32*(zone-state)
                           ; 32 zone-state digits for zones 1 through 32,
separated by spaces
                           ; zones not present in the system will report as zone
sealed (0)

zone-state = 1*DIGIT ; the state of the zone
                           ; 0 = zone sealed
                           ; 1 = zone unsealed
                           ; 2 = zone open
                           ; 3 = zone short
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] status_report_1 1 0 0 0 0 0 0 0 0 0
0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Sample SCP output

```
#security status_report_1  //system/1/224 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 #sourceunit=100
```

Event

Status Report 2

Condition causing event

The security system reports its current state using this and the **Status Report 1** event. This may result from the security system receiving a **SECURITY STATUS_REQUEST** command for Status Report 2.

Event Text


```
"status_report_2" extended-zone-report

extended-zone-report = 48*(zone-state SP)
                            ; 48 zone-state digits for zones 33 through 80,
separated by spaces
                            ; zones not present in the system will report as zone
sealed (0)

zone-state = 1*DIGIT ; the state of the zone
                            ; 0 = zone sealed
                            ; 1 = zone unsealed
                            ; 2 = zone open
                            ; 3 = zone short
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] status_report_2 0 0 0 0 0 0 0 0 0 0
1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Sample SCP output

```
#security status_report_2  //system/1/224 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  #sourceunit=100
```

Event

Password Entry Status

Condition causing event

The security system reports the state of password entry, including successful attempts, failed attempts, and when entry is barred for a time after too many failed attempts.

Event Text

```
"password_entry_status" password-entry-status


password-entry-status = 1*3DIGIT  ; the state of the zone
                            ; 1 = password entry succeeded
                            ; 2 = password entry failed
                            ; 3 = password entry disabled
                            ; 4 = password entry re-enabled (after previous
disable)
                            ; 0, 5-255 reserved
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] password_entry_status 1
```

Sample SCP output


```
#security password_entry_status  //system/1/224 1 #sourceunit=100
```

Event
Mains Failure

Condition causing event

The security system has detected a failure of its mains power.

Event Text

```
"mains_failure"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] mains_failure
```

Sample SCP output

```
#security mains_failure  //system/1/224 #sourceunit=100
```

Event
Mains Restored

Condition causing event

The security system has detected restoration of its mains power.

Event Text

```
"mains_restored"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] mains_restored
```

Sample SCP output

```
#security mains_restored  //system/1/224 #sourceunit=100
```

Event
Arm Ready

Condition causing event

The security system has armed correctly (all zones are ready)

Event Text

```
"arm_ready"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] arm_ready
```

Sample SCP output

```
#security arm_ready  //system/1/224 #sourceunit=100
```

Event
Line Cut Alarm Raised

Condition causing event

The security system has detected the attached phone line being cut.

Event Text

```
"line_cut_alarm line_cut_alarm_raised"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] line_cut_alarm line_cut_alarm_raised
sourceUnit=100
```

Sample SCP output

```
#security line_cut_alarm line_cut_alarm_raised //system/1/224 #sourceunit=100
```

Event
Line Cut Alarm Cleared

Condition causing event

The security system has detected the attached phone line being reconnected.

Event Text

```
"line_cut_alarm line_cut_alarm_cleared"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] line_cut_alarm
```



```
line_cut_alarm_cleared sourceUnit=100
```

Sample SCP output

```
#security line_cut_alarm line_cut_alarm_cleared //system/1/224 #sourceunit=100
```

Event
Fire Alarm Raised

Condition causing event

The security system has detected fire.

Event Text

```
"fire_alarm fire_alarm_raised"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] fire_alarm fire_alarm_raised
sourceUnit=100
```

Sample SCP output

```
#security fire_alarm fire_alarm_raised //system/1/224 #sourceunit=100
```

Event
Fire Alarm Cleared

Condition causing event

The security system has detected that a fire condition has ceased.

Event Text

```
"fire_alarm fire_alarm_cleared"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] fire_alarm fire_alarm_cleared
sourceUnit=100
```

Sample SCP output

```
#security fire_alarm fire_alarm_cleared //system/1/224 #sourceunit=100
```

Event
Arm Failed Raised

Condition causing event

The security system has failed to arm.

Event Text

```
"arm_failed arm_failed_raised"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] arm_failed arm_failed_raised
sourceUnit=100
```

Sample SCP output

```
#security arm_failed arm_failed_raised //system/1/224 #sourceunit=100
```

Event
Arm Failed Cleared

Condition causing event

The security system has been able to arm after having previously failed.

Event Text

```
"arm_failed arm_failed_cleared"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] arm_failed arm_failed_cleared
sourceUnit=100
```

Sample SCP output

```
#security arm_failed arm_failed_cleared //system/1/224 #sourceunit=100
```

Event
Gas Alarm Raised

Condition causing event

The security system has detected the presence of gas.

Event Text

```
"gas_alarm gas_alarm_raised"
```


Sample Event output

```
20020910-172843 702 //system/1/224 [security] gas_alarm gas_alarm_raised
sourceUnit=100
```

Sample SCP output

```
#security gas_alarm gas_alarm_raised //system/1/224 #sourceunit=100
```

Event
Gas Alarm Cleared

Condition causing event

The security system has detected that the presence of gas has cleared.

Event Text

```
"gas_alarm gas_alarm_cleared"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] gas_alarm gas_alarm_cleared
sourceUnit=100
```

Sample SCP output

```
#security gas_alarm gas_alarm_cleared //system/1/224 #sourceunit=100
```

Event
Other Alarm Raised

Condition causing event

The security system has detected special alarm condition.

Event Text

```
"other_alarm other_alarm_raised"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] other_alarm other_alarm_raised
sourceUnit=100
```

Sample SCP output

```
#security other_alarm other_alarm_raised //system/1/224 #sourceunit=100
```


Event
Other Alarm Cleared

Condition causing event

The security system has detected the removal of a special alarm condition.

Event Text

```
"other_alarm other_alarm_cleared"
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] other_alarm other_alarm_cleared
sourceUnit=100
```

Sample SCP output

```
#security other_alarm other_alarm_cleared //system/1/224 #sourceunit=100
```

4.7.9.2.1.3 Object Parameters

The following parameters are supported by the Security Application Object.

They can be accessed with the GET command acting on the object address of the relevant Security Application object.

Parameter Name	Type & Description	Can set with SET command?	Notes
State	String. The overall state of this object. Valid states are: "ok" = normal operation "new" = the object has been created and has yet to synchronize with the C-Bus network and security system "sync" = the object is synchronising with the C-Bus network and security system "error" = an error has occurred.		In any state except for "ok", all parameters will return "unknown" or -1 results until synchronisation has been achieved. Synchronisation may be initiated by using the **SECURITY STATUS_REQUEST** command which will attempt to refresh the variables below from the results. Note that some variables can not be refreshed in this way. These will be refreshed when the appropriate events are received.
ArmState	Integer. Returns the armed state of the security system. Valid values are:	no	

	0 = disarmed 1 = fully armed 2 = partially armed 3 - 127 = manufacturer dependent -1 = arm state unknown		
AlarmState	String. Returns the state of alarm of the security system. Valid values are: "on" = alarm is on/in progress "off" = no alarm in progress "unknown" = alarm state is unknown	no	
TamperState	String. Returns the state of tamper of the security system. Valid values are: "on" = tampering detected "off" = no tampering detected "unknown" = tamper state is unknown	no	
PanicState	String. Returns the state of panic of the security system. Valid values are: "on" = panic detected "off" = no panic detected "unknown" = panic state is unknown	no	
LowBatteryState	String. Returns the low battery state of the security system: "detected" = low battery charge detected "corrected" = low battery charge corrected "unknown" = battery state is unknown	no	
ChargingState	String. Returns the charging state of the security system: "on" = battery is being charged "off" = battery is not being charged "unknown" = charging state is unknown	no	
PassEntryState	Integer. Returns the current password entry state of the security system. Valid values are: 1 = password entry succeeded 2 = password entry failed 3 = password entry disabled 4 = password entry re-enabled (after previous disable) -1 = unknown	no	
MainsState	String: Return the mains state of the security system. Valid values are:	no	

	"on" = Mains supply on "off" = Mains supply off "unknown" = Mains supply state unknown		

4.7.9.2.2 Security Zone Object

The Security Zone object models a security zone which is controlled by a security panel or other security device that interfaces to a C-Bus network.

A Security Application may have 0 or more security zones associated with them.

Addressing Security Zone Objects

Security Zone Objects are addressed as subelements of the Security Application object. Valid security zone addresses range from `1` through `127` (or in hexadecimal, `$01` through `$7f`). No other values for security zones are allowed or valid.

So, on an example network called `net1`, the security application would be addressed as `net1/$D0` or `net1/208`. Security Zone number 1 would be addressed as `net1/208/1` , and zone $25 would be addressed as `net1/$D0/$25` .

4.7.9.2.2.1 Commands Supported

The following commands are supported by the Security Zone object.

- SECURITY REQUEST_ZONE_NAME

4.7.9.2.2.2 Events Supported

The following events are supported by the Security Zone object.

Event

Zone Unsealed

Condition causing event

The security system alarm has detected a zone becoming unsealed that was previously sealed.

Event Text

```
"zone_unsealed"
```

Sample Event output

```
20020910-172843 702 //system/1/224/1 [security] zone_unsealed sourceUnit=100
```

Sample SCP output

```
#security zone_unsealed //system/1/224/1 #sourceunit=100
```


Event
Zone Sealed

Condition causing event

The security system alarm has detected a zone becoming sealed that was previously unsealed.

Event Text

```
"zone_sealed" zone-number
```

Sample Event output

```
20020910-172843 702 //system/1/224/9 [security] zone_sealed sourceUnit=100
```

Sample SCP output

```
#security zone_sealed //system/1/224/9 #sourceunit=100
```

Event
Zone Open

Condition causing event

The security system alarm has detected a protected loop zone becoming open circuit.

Event Text

```
"zone_open"
```

Sample Event output

```
20020910-172843 702 //system/1/224/5 [security] zone_open sourceUnit=100
```

Sample SCP output

```
#security zone_open //system/1/224/5 #sourceunit=100
```

Event
Zone Short

Condition causing event

The security system alarm has detected a protected loop zone becoming short circuit.

Event Text

```
"zone_short"
```


Sample Event output

```
20020910-172843 702 //system/1/224/2 [security] zone_short sourceUnit=100
```

Sample SCP output

```
#security zone_short //system/1/224/2 #sourceunit=100
```

Event
Zone Isolated

Condition causing event

The security system alarm has isolated, bypassed or shunted a zone.

Event Text

```
"zone_isolated"
```

Sample Event output

```
20020910-172843 702 //system/1/224/3 [security] zone_isolated sourceUnit=100
```

Sample SCP output

```
#security zone_isolated //system/1/224/3 #sourceunit=100
```

Event
Zone Name

Condition causing event

The security system has emitted a text name for a zone. This may result from the security system receiving a **SECURITY REQUEST_ZONE_NAME** command.

Event Text

```
"zone_name" zone-name
```

Sample Event output

```
20020910-172843 702 //system/1/224 [security] zone_name KITCHEN1 sourceUnit=100
```

Sample SCP output

```
#security zone_name //system/1/224 KITCHEN1 #sourceunit=100
```


Event
Arm Not Ready

Condition causing event

This event is sent during Arming if any security zones do no seal correctly. One message will be sent for each zone that does not correctly seal.

Event Text

```
"arm_not_ready"
```

Sample Event output

```
20020910-172843 702 //system/1/224/9 [security] arm_not_ready sourceUnit=100
```

Sample SCP output

```
#security arm_notReady //system/1/224/9 #sourceunit=100
```

4.7.9.2.2.3 Object Parameters

The following parameters are supported by the Security Zone Object.

They can be accessed with the GET command acting on the object address of the relevant Security Zone object.

Parameter Name	Type & Description	Can be set with SET command?	Notes
State	String. The overall state of this object. Valid states are: "ok" = normal operation "new" = the object has been created and has yet to synchronise with the C-Bus network and security system "sync" = the object is synchronising with the C-Bus network and security system "error" = an error has occurred.	No	In any state except for "ok", all parameters will return "unknown" or -1 results until synchronisation has been achieved. Synchronisation may be initiated by using the **SECURITY STATUS_REQUEST** command which will attempt to refresh the variables below from the results. Note that some variables can not be refreshed in this way. These

			will be refreshed when the appropriate events are received.
ZoneState	The state of this zone. Valid values are: 0 = sealed 1 = unsealed 2 = open 3 = short -1 = unknown	No	
ZoneName	String. The name of this zone, if supplied by the security system, or "unknown" otherwise.	No	

4.7.9.3 Specification

For detailed knowledge regarding the function of the Security Application on the C-Bus network, refer to "C-Bus Application Messages and Behavior -- Chapter 5: Security (CBUS-APP/5), Issue 3, 13 December 2002".

4.7.10 Short Message Application

The Short Message Application is used to transfer small messages around a C-Bus network. The messages comprise text, and optionally a single number and/or reference to a predefined graphical icon.

The Short Message Application has a C-Bus application address of $AD, or 173 decimal.

4.7.10.1 Short Message Overview

This gives a short overview of the Media Transport Control Application implementation in the C-Gate server

Application name

Short Message

Application short name

(used as first part of commands or events to refer to this application)

shortmessage

C-Bus Application range

$AD (173)

Commands

(these can be entered from the C-Gate command interface preceded by the short name of this application. All commands are also recognised as events)

refresh
send

Events

(these are detected by the C-Gate server and will be shown in the event list and status port)

refresh
send

Notes and implementation issues

4.7.10.2 Events Issued

The events in this section are generated by the Short Message application and are made available as events in the C-Gate server event port, or via the Status Change Port (SCP)

4.7.10.2.1 Short Message Refresh

Event

Short Message Refresh

Condition causing event

Receiving a SHORTMESSAGE REFRESH command sequence.

Event text

```
[shortmessage] refresh info-type sourceUnit

info-type:
   0 = generic text
   1 = weather forecast
   2 = surf report
   3 = tide times
   4 = unread emails in inbox
   5 = voice mail messages
   6 = stock prices ticker
   7 = news headline
   8 = other headline
   9 = air quality forecast
  10 = UV radiation forecast
  11 = pollen count
  12 = traffic report
  13 = sports result
  14 = horoscope
  15 = joke of the day
  16 = trivia of the day
  17 = todays appointments
  18 = things to do today
  19 = birthdays today
  20 = management notice
  21 = electrical power brownout warning
  22 = irrigation

sourceUnit = source of the command
```

Sample Event output


```
20090310-171721 702 //PROJ/254/173 fdf4fde0-e392-102b-a572-829dbedeadab
[shortmessage] refresh info-type=1 sourceUnit=3
```

Sample SCP output

```
#s# shortmessage refresh //PROJ/254/173 1 #sourceunit=3 OID=fdf4fde0-
e392-102b-a572-829dbedeadab
```

4.7.10.2.2 Short Message Send

Event

Short Message Send

Condition causing event

Receiving a SHORTMESSAGE SEND command sequence.

Event text

```
[shortmessage] send info-type sourceUnit

info-type:
   0 = generic text
   1 = weather forecast
   2 = surf report
   3 = tide times
   4 = unread emails in inbox
   5 = voice mail messages
   6 = stock prices ticker
   7 = news headline
   8 = other headline
   9 = air quality forecast
  10 = UV radiation forecast
  11 = pollen count
  12 = traffic report
  13 = sports result
  14 = horoscope
  15 = joke of the day
  16 = trivia of the day
  17 = todays appointments
  18 = things to do today
  19 = birthdays today
  20 = management notice
  21 = electrical power brownout warning
  22 = irrigation

sourceUnit = source of the command
```

Sample Event output

```
20090310-171721 702 //PROJ/254/173 fdf4fde0-e392-102b-a572-829dbedeadab
[shortmessage] refresh info-type=1 sourceUnit=3
```

Sample SCP output


```
#s# shortmessage refresh //PROJ/254/173 1 #sourceunit=3 OID=fdf4fde0-
e392-102b-a572-829dbedeadab
```

4.7.11 Telephony Application

The Telephony Application gives C-Gate the ability to interact with C-Bus capable telephone devices such as the C-Bus Telephone Interface.

4.7.11.1 Telephony Application Overview

Application name

Telephony

Application short name (used as first part of commands or events to refer to this application)

telephony

C-Bus Application range

$E0

Commands

(these can be entered from C-Gate command interface preceded by the short name of this application. All commands are also recognised as events)

isolate_secondary_outlet
recall_last_number_request
reject_incoming_call
divert
clear_diversion

Events

(these are detected by C-Gate and will be shown in the event list and status port but are not available as commands)

line_on_hook
line_off_hook
dial_out_failure
dial_in_failure
ringing
recall_last_number_response
internet_connection_request_made

4.7.11.2 Objects supported

One new object is provided for the implementation of this application, the Telephony Application Object which is described in the sections which follow.

4.7.11.2.1 Telephony Application Object

This is the application object created in the server when a telephony command is detected on the C-Bus network or if a telephony command is sent from the command line. This object provides the

following parameters in addition to those supported by the standard Application object.

Parameter Name	Description	Read Access Control	Write Access Control
ShortName	The short name for this application. This is set to **telephony** when the application is loaded	Monitor	Read Only

4.7.11.3 Commands supported

The following commands are available in the Telephony application:

- **TELEPHONY**
- **TELEPHONY CLEAR_DIVERSION**
- **TELEPHONY DIVERT**
- **TELEPHONY ISOLATE_SECONDARY_OUTLET**
- **TELEPHONY RECALL_LAST_NUMBER_REQUEST**
- **TELEPHONY REJECT_INCOMING_CALL**

4.7.11.4 Events supported

The events in this section are recognised by the Telephony Application and are made available as events in the C-Gate server event port, or via the Status Change Port (SCP)

4.7.11.4.1 Line On Hook

Event Line On Hook

Condition causing event The telephony device has detected that the line has gone on hook.

Event text `line_on_hook`

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] line_on_hook sourceUnit=131`

Sample SCP output `# telephony line_on_hook //system/1/224 #sourceunit=131`

4.7.11.4.2 Line Off Hook

Event Line Off Hook

Condition causing event The telephony device has detected that the line has gone off hook.

Event text `line_off_hook <direction> <type> [<number>]`

`<direction>` = `in` for incoming calls, `out` for outgoing calls `<reason>` = `voice` when the telephony device is handling a voice call, `data` when the telephony device is handling

a data call, or `other` when another device is handling the call.
`<number>` = (optional) the rightmost 16 bytes of the calling number identification string (numbers or characters)

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] line_off_hook in other 5551212 sourceUnit=131`
Sample SCP output `# telephony line_off_hook //system/1/224 in other 5551212 #sourceunit=131`

4.7.11.4.3 Dial Out Failure

Event Dial Out Failure

Condition causing event The telephony device has indicated that the previous dial-out attempt has failed.

Event text `dial_out_failure <reason>`

`<reason>` = `no_dialtone` or `no_answer` or `no_ack_prompt` or `unobtainable` or `busy`

`no_dialtone` indicates that no dialtone was detected
`no_answer` indicates that the call was not answered
`no_ack_prompt` indicates that there was no valid acknowledgement of prompts
`unobtainable` indicates that the number was unobtainable or does not exist
`busy` indicates that the number was busy

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] dial_out_failure busy sourceUnit=131`
Sample SCP output `# telephony dial_out_failure //system/1/224 busy #sourceunit=131`

4.7.11.4.4 Dial In Failure

Event Dial In Failure

Condition causing event The telephony device has indicated that the previous dial-in attempt has failed.

Event text `dial_in_failure <reason>`

`<reason>` = `no_answer` `no_answer` indicates that the call was not answered

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] dial_in_failure no_answer sourceUnit=131`

Sample SCP output `# telephony dial_in_failure //system/1/224 no_answer #sourceunit=131`

4.7.11.4.5 Ringing

Event Ringing

Condition causing event The telephony device has detected that the telephone line is ringing

Event text `ringing [<number>]`

`<number>` = (optional) the rightmost 16 bytes of the calling number identification string (numbers or characters)

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] ringing 5551212 sourceUnit=131`

Sample SCP output `# telephony ringing //system/1/224 5551212 #sourceunit=131`

4.7.11.4.6 Recall Last Number Response

Event Recall Last Number Response

Condition causing event The telephony device's response to a Recall Last Number Request.

Event text `last_number <direction> <number>`

`<direction>` = `in` for incoming calls, `out` for outgoing calls
`<number>` = (optional) the rightmost 16 bytes of the calling number identification string (numbers or characters)

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] last_number in 5551212 sourceUnit=131`

Sample SCP output `# telephony last_number //system/1/224 in 5551212 #sourceunit=131`

4.7.11.4.7 Internet Connection Request Made

Event Internet Connection Request Made

Condition causing event The telephony device indicates that it has received a request to initiate an Internet connection.

Event text `internet_connection_request_made`

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] internet_connection_request_made sourceUnit=131`

Sample SCP output `# telephony internet_connection_request_made //system/1/224 #sourceunit=131`

4.7.11.4.8 Isolate Secondary Outlet

Event Isolate Secondary Outlet

Condition causing event A network device has requested that the telephony device set the isolation for the secondary outlet

Event text `isolate_secondary_outlet <mode>`

`<mode>` = set to `normal` to request normal operation of the outlet, or set to `isolate` to request isolation of the outlet.

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] isolate_secondary_outlet normal sourceUnit=131`

Sample SCP output `telephony isolate_secondary_outlet //system/1/224 normal #sourceunit=131`

4.7.11.4.9 Recall Last Number Request

Event Recall Last Number Request

Condition causing event A network device has requested that the telephony device recall the last number.

Event text `recall_last_number_request <direction>`

`<direction>` = `in` to get the last incoming call number, `out` for the last outgoing call number

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] recall_last_number_request in sourceUnit=131`

Sample SCP output `telephony recall_last_number_request //system/1/224 in #sourceunit=131`

4.7.11.4.10 Reject Incoming Call

Event Reject Incoming Call

Condition causing event Anetwork device has requested that the telephony device reject the incoming call.

Event text `reject_incoming_call`

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] reject_incoming_call sourceUnit=131`

Sample SCP output `telephony reject_incoming_call //system/1/224 #sourceunit=131`

4.7.11.4.11 Divert

Event Divert

Condition causing event A network device has requested that the telephony device divert calls.

Event text `divert [<number>]`

`<number>` = (optional) the number to perform the divert to

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] divert 5551212 sourceUnit=131`

Sample SCP output `telephony divert //system/1/224 5551212 #sourceunit=131`

4.7.11.4.12 Clear Diversion

Event Clear Diversion

Condition causing event A network device has requested that the telephony device clear any current diversion

Event text `clear_diversion`

Sample Event output `20020910-172843 702 //system/1/224/144 [telephony] clear_diversion sourceUnit=131`

Sample SCP output `telephony clear_diversion //system/1/224 #sourceunit=131`

4.7.11.4.13 Configuration properties

There are no configuration properties for this application.

4.7.11.5 Usage Notes

There are no network variables available for this application.

4.7.11.6 Specification

For detailed knowledge regarding the function of the Telephony Application on the C-Bus network, refer to "C-Bus Application Messages and Behaviour -- Chapter 24: Telephony Status & Control (CBUS-APP/24), Issue 1, 7 August 2002".

4.7.12 Temperature Broadcast Application

The Temperature Broadcast Application is used by C-Bus devices that are performing temperature measurements to broadcast temperature measurements to the C-Bus network.

The C-Gate server can receive temperature broadcast events and send temperature broadcasts from the command line.

4.7.12.1 Application Overview

The table gives a short overview of the Temperature Broadcast Application implementation in the C-Gate server

Application name	Temperature Broadcast
Application short name (used as first part of	temperature

commands or events to refer to this application)	
C-Bus Application range	$19 (decimal 25)
Commands (these can be entered from C-Gate command interface preceded by the short name of this application. All commands are also recognised as events)	broadcast
Events (these are detected by C-Gate and will be shown in the event list and status port but are not available as commands)	
Notes and implementation issues	

4.7.12.2 Objects Supported

There are two new objects supported by the Temperature Broadcast Application implementation.

The Temperature Broadcast Application Object models the application as a whole. The Temperature Broadcast Application Object is addresssed using the application address for the Temperature Broadcast Application ($19 or 25) on the C-Bus network that the temperature sensing devices are connected to.

The Temperature Group Object models one group of temperature readings. The temperature group is addressed as a sub-address of the Temperature Broadcast Application that it is associated with.

4.7.12.2.1 Temperature Broadcast Application Object

The Temperature Broadcast Application Object models a C-Bus network's Temperature Broadcast Application.

The Temperature Broadcast Application Object supports up to 256 **Temperature Group Objects**.

Addressing

The Temperature Broadcast Application Object is addresssed using the standard application address for the Temperature Broadcase Application (`$19` or decimal `25`) on the relevant C-Bus network.

For example, for a hypothetical network called `net1`, the Temperature Broadcast Application's address will be `net1/$1E` or `net1/25`.

4.7.12.2.1.1 Commands Supported

The following command line commands are accepted by the Temperature Broadcast Applcation Object:

- TEMPERATURE BROADCAST

4.7.12.2.1.2 Events Supported

There are no events supported by this object.

4.7.12.2.1.3 Object Parameters

The following parameters are supported by the Temperature Broadcast Application Object.

They can be accessed with the GET command acting on the object address of the relevant Temperature Broadcast Application object.

Parameter Name	Type & Description	Can set with SET command?	Notes
State	String. The overall state of this object. Valid states are: "ok" = normal operation "new" = the object has been created and has yet to synchronize with the C-Bus network "sync" = the object is synchronising with the C-Bus network "error" = an error has occurred.	No	

4.7.12.2.2 Temperature Group Object

The Temperature Group object models a group of one or more temperature sensing devices. The Temperature group used by a C-Bus sensor will normally be set when the temperature sensor device is programmed.

Addressing Temperature Group Objects

A Temperature Broadcast Application can typically have 256 temperature groups associated with the application.

Temperature Group Objects are addressed as subelements of the Temperature Broadcast Application Object. Valid Temperature Group addresses range from `0` through `255` (or in hexadecimal, `$00` through `$FF`). No other values are are allowed or valid.

So, on an example network called `net1`, the temperature broadcast application would be addressed as `net1/$1E` or `net1/208`. Temperature group number 1 would be addressed as `net1/208/1` , and temperature group $25 would be addressed as `net1/$D0/$25` .

4.7.12.2.2.1 Commands Supported

The following commands are supported by the Temperature Group Object.

- TEMPERATURE BROADCAST

4.7.12.2.2.2 Events Supported

The following events are supported by the Temperature Broadcast object.

Event

Temperature Broadcast

Condition causing event

A temperature broadcast has been received.

This event can be generated by the **TEMPERATURE BROADCAST** command.

Event Text

```
"broadcast" celsius-temperature
```

Sample Event output

```
20020910-172843 702 //system/1/224/1 [temperature] broadcast 25.5 sourceUnit=100
```

Sample SCP output

```
temperature broadcast //system/1/224/1 #sourceunit=100
```

4.7.12.2.2.3 Object Parameters

The following parameters are supported by the Temperature Group Object.

They can be accessed with the GET command acting on the object address of the relevant Temperature Group object.

Parameter Name	Type & Description	Can be set with SET command?	Notes
State	String. The overall state of this object. Valid states are: "ok" = normal operation "new" = the object has been created and has yet to synchronise with the C-Bus network "sync" = the object is attempting to synchronise with the C-Bus network	No	

	"error" = an error has occurred.		
Temperature	The last temperature update received for this temperature group.	No	

4.7.12.3 Specification

For detailed knowledge regarding the function of the Temperature Broadcast Application on the C-Bus network, refer to "C-Bus Application Messages and Behavior -- Chapter 9: Temperature Broadcast (CBUS-APP/09), Issue 1, 7 August 2002".

4.7.13 Trigger Control Application

This document covers the implementation of the C-Bus Trigger Control Application within C-Gate.

4.7.13.1 Trigger Overview

This is a short overview of the Trigger Control Application's implementation in C-Gate

Application name

Trigger Control

Application short name (used as first part of commands or events to refer to this application)

trigger

C-Bus Application range

$CA (202)

Commands/Events (these can be entered from the C-Gate command interface preceded by the short name of this application. All commands are also recognised as events)

event

Events (these are detected by the C-Gate server and will be shown in the event list and status port but are not available as commands)

Notes and implementation issues

MMIs supported Units not directly supported Trigger groups are automatically created if named on the command line or if event is received from the network.

4.7.13.2 Trigger Objects Supported

Two new objects are provided for the implementation of this application:

4.7.13.2.1 Trigger Application Object

This is the application object created in the server when a trigger control application is detected on the C-Bus network. This object provides the following parameters in addition to those supported by the standard Application object.

Parameter Name Description Read Access Control Write Access Control
ShortName The short name for this application. This is set to **`trigger`** when the application is loaded Monitor Read Only

4.7.13.2.2 Trigger Group Object

The groups of this application support the following methods, which mirror the command described later:

Method Name	Description	Access Control
TriggerEvent	See TRIGGER EVENT command below	Operate

4.7.13.3 Trigger Events

Events are issued for the Event messages detected on a C-Bus network. They are in the form show below:

```
20020910-172843.078 702 //system/1/202/144 [trigger] event action=2  sourceUnit=131
```

In the example shown, the address given is the trigger group that was activated, the command given is **event**, the **action=** paramter gives the action selector value, and the **sourceUnit=** gives the source unit id on the network where the command was sent.

4.7.13.3.1 Trigger Commands

The following commands are supported by this application:

TRIGGER and
TRIGGER EVENT

4.7.13.4 Status Change Port

This application places TRIGGER EVENT commands into the Status Change Port (SCP). Samples are shown below.

```
trigger event //system/1/202/144 2 #sourceunit=131
```

4.7.13.5 Configuration

There are no applicable parameters.

4.7.13.6 Project Database Issues

This application does **not** support the <Group> element under <Application>. Instead, it supports the <NetVar> element which can contain zero or more <Level> elements containing Level Tag information.

To use level tags:
Add one or more level entries to the <NetVar> element in the tag database. For each of the NetVar elements, add a <Value> element with the appropriate application selector level in the range 0 through 255.

So, an except would look like:

```
<NetVar>
  <TagName>Scene1</TagName>
       <Level Value="0">
            <TagName>One</TagName>
       </Level>
       <Level Value="76">
            <TagName>Two</TagName>
       </Level>
</NetVar>
```

This allows the Trigger Event command to use the tags AllOff and Level1 as action selector values, for example:

```
trigger event Switch1 AllOff
```


4.7.13.7 Usage Notes

Groups are created automatically when events for them are received, or they are detected in other ways.

4.7.13.8 Trigger Specification

For detailed knowledge regarding the function of the Trigger Control Application, refer to "C-Bus Application Messages and Behaviour -- Chapter 7: Trigger Control (CBUS-APP/07)".

4.8 Projects

The project command set supersedes the tag management commands and is used for management of entire projects which consist of multiple C-Bus networks.

4.8.1 Concept of Project

A **project** is a set of C-Bus networks that are operated together or work together. A project is normally programmed as one entity by the C-Bus Tooklit commissioning software.

Information about a project is stored in a Project Database, which is also known as a Tag Database. A Project Database holds the specification of a project in a standard XML form.

4.8.2 Projects And Tags

A project is contained in a single project or tag database. The tag database has a <Project> element that contains the specification for the networks that are to be used in the project. The name of the project is the same as the project database file, excepting extensions (such as .xml or .zip).

To remain compatible with C-Bus Toolkit, all project names should be formatted as described in Project Names below.

4.8.3 Project Names

Project names should be constructed using the following rules to work smoothly across the full set of C-Bus software:

- All uppercase letters only
- First character must not be a number
- Maximum of 8 charcaters in length
- **Not** be one of the reserved words:
  - P, CBUS, VM, CMDINT, CGATE, TAG, CMD*n (where n is a number)*

4.8.4 Projects And Networks

There are one or more C-Bus networks in a Project.

4.8.5 Addressing

The introduction of projects makes some fundamental changes to the absolute addressing used in addressing objects in C-Gate, although all addressing is backward compatible apart from the addressing required to refer to any project that is not the default project, or the current project referred to by a command session.

An absolute project address starts with the characters `//` followed by the project name. Following this, the address has a `/` character, and then either a `p` character or a network name.
An address without the `//project/` is a relative address — it is relative to the current project for the context of the command. This means the setting of the project use command.
Here are some example addresses using absolute and relative addressing:

Address	Meaning
`//BUILDING/1/56/22`	Group 22, application 56, network 1, project `BUILDING`
`1/56/22`	Group 22, application 56, network 1 in the current project

`//PROJ/p/6/20`	Unit 20, network 6 in the project called `PROJ`.
`//MYLIGHTS`	This is the address of the project called `MYLIGHTS`

Note: There is no way to combine pre-1.5 addressing (dot addressing) with project addresses. To use C-Gate 2.0 and later, do not use dot addressing.

4.8.6 Default Project

C-Gate has one default project. This default project is defined at startup by the `project.default` configuration parameter and can be changed by changing this parameter. Note that this is a global parameter and changing it may impact all sessions connected to C-Gate.

4.8.7 Current Project

Each command session has a current project that it uses when relative addresses are given.

The current project is a property of that command session only.

When a command session starts, it sets its command session to the value of C-Gate's default project. This project value can be viewed and changed using the PROJECT USE command.

4.8.8 Project Properties

The following properties are used with the PROJECT commands:

project.default.dir - The default directory where projects tag databases are stored
project.default - The default project for this C-Gate server.
project.start - The list of projects to be started on C-Gate start-up. The list is a list of project names separated by spaces. The projects will be started in the specificed order.

4.8.9 Access Control Issues

All of the commands required an access level of **admin** to perform the commands.

The response `420 Access denied` will be returned if **admin** or higher access level is not held by the command session that is performing the commands.

4.9 Networks

A **network** is a model of a C-Bus network that is normally programmed by the C-Bus Toolkit commissioning software.

C-Gate maintains two different models of a network, the database model and the live model.

Database model of a network

This is a static copy that is maintained in the Tag Database. It is accessed and manipulated using commands such as DBGET and DBSET.

Live model of a network

This is a live model that is maintained in C-Gate only while it is running. It is accessed and

manipulated using commands such as GET and SET.

C-Gate keeps this model synchronised with the C-Bus network through Network Syncing and by listening for C-Bus messages.

There are also commands to copy the live model to the database.

4.9.1 Opening a Network

In order to communicate with a network, C-Gate first needs to *open* it.

When a network is opened, C-Gate will:

1. Set the network object's InterfaceState to *opening.*
2. Identify the gateway unit.
3. Establish an active connection to the gateway unit.
4. If successful:
   a. Set the network object's InterfaceState to *running.*
   b. Place the network into the background sync queue. In most cases it will start syncing almost immediately.
5. If unsuccessful C-Gate will return the network object's InterfaceState back to *closed.*

For more information on InterfaceState see the following section: Network Interface State Diagram.

Opening a network

If you have one network in your project, or if you just want to start one network, then use the NET OPEN command:

```
net open 254
120-initializing
120-opening port
120-starting network threads
120-pci reset
120-open complete
200 OK: //HOME/254
```

See that the net open command returns a number of lines of information as the network is being opened.

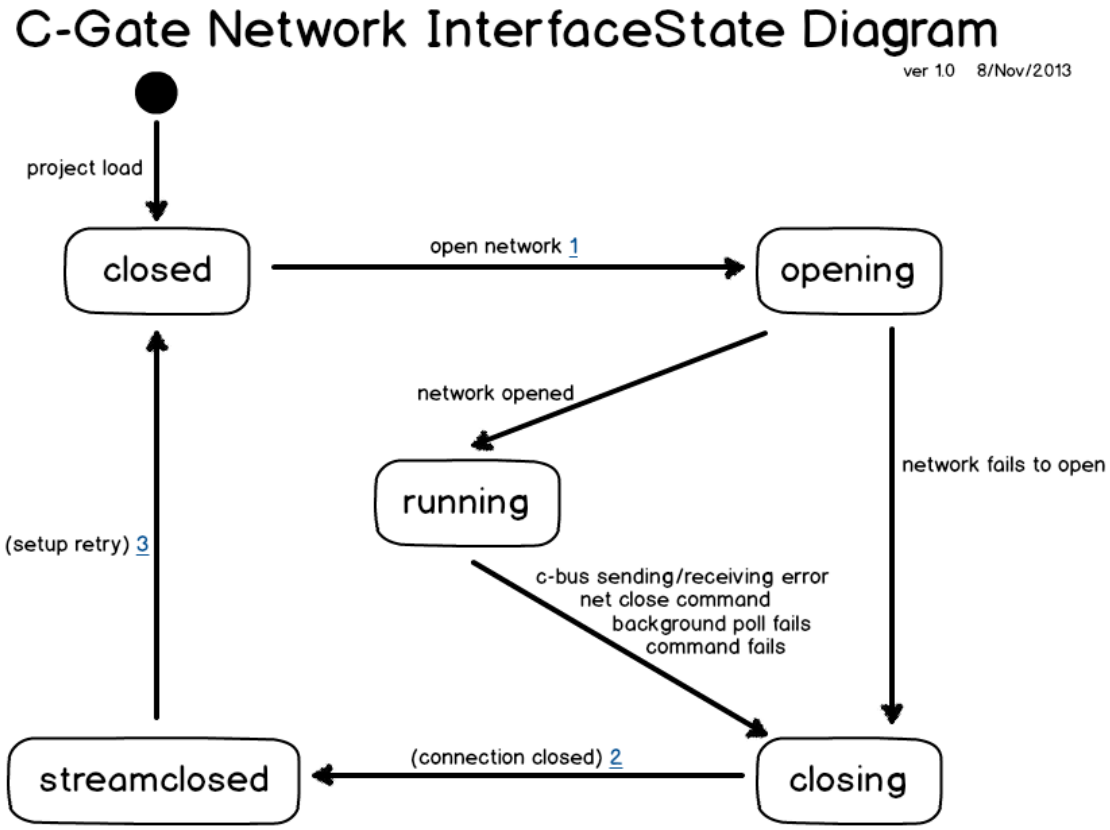
Starting a project

To open all the networks in your project, use the PROJECT START command.

```
project start
200 OK.
```

This will open all networks in the project.

4.9.1.1 Network Interface State Diagram



open network 1

- open network can be triggered by
  - net open command
  - retry after network is closed because of polling failure
  - retry after network is closed because of c-bus sending or receiving errors
  - retry after bridge network is closed because routing network is closed
  - retry after network fails to open
  - (note: retry only happens when auto-reopen is set to yes which is the default value)

(connection closed) 2

- transition from closing to streamclosed is handled internally

(setup retry) 3

- transition from streamclosed closed is handled internally
- during the transition TargetInterfaceState is set to running if a retry is needed, then after a retry delay the network is reattempted to be opened

4.9.2 Network State

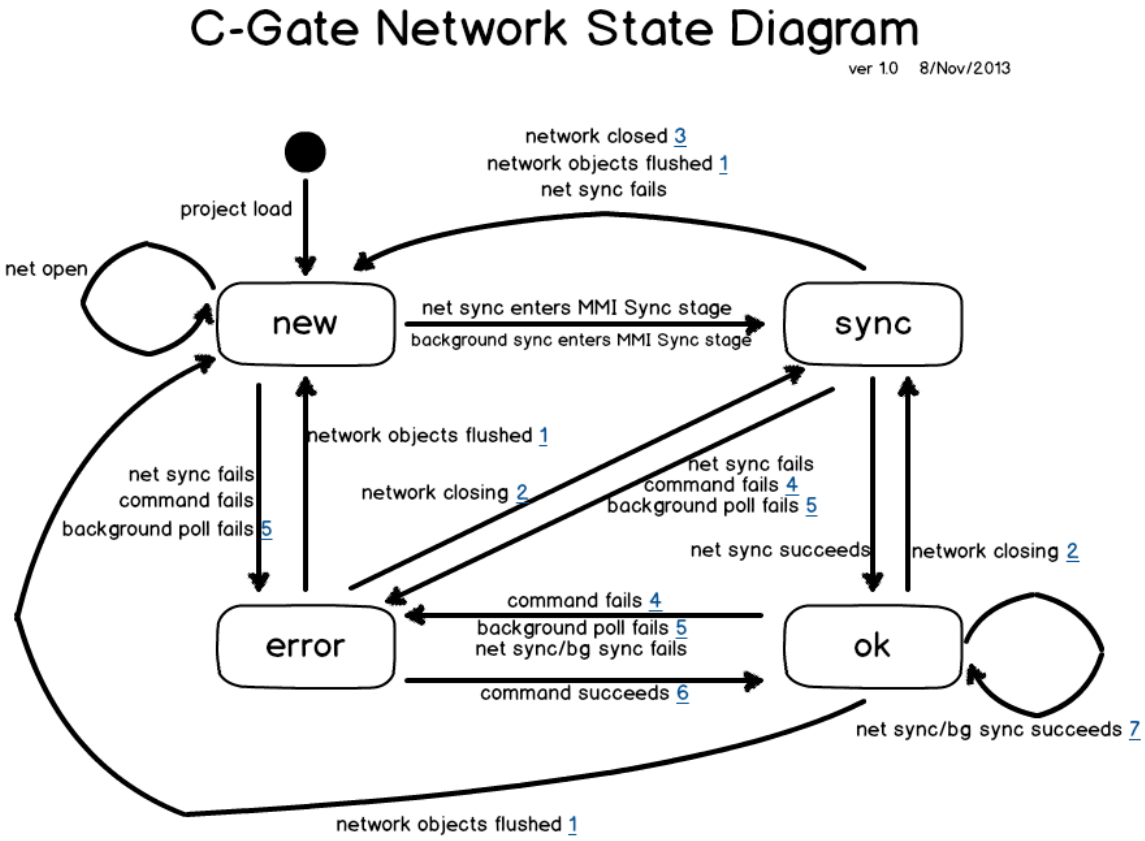
Each network has a state which indicates whether the network has been synchronised or not. The states are:

`new`: no sync has been performed
`sync`: network synchronisation is in progress
`error`: an error occurred while attempting to sync
`ok`: the network is synchronised and operating normally.

The goal of the sync process is for the network to end up in the `ok` state. Once the `ok` state has been reached, then C-Gate has a full model of the network and it is easy to work out how to control and monitor the network.

See the following Network State Diagram for a diagram of these states.

4.9.2.1 Network State Diagram



network objects flushed 1
(note: a background sync starts shortly after objects are flushed)
- net flush command(not allowed during sync)
- net load command after changing network interface in db
- change Interface/InterfaceAddress using set command
(only allowed when interface state is closed)
- change bridge network address using net rename command
(only allowed when interface state is closed)

network closing 2 network closed 3
- network close can be triggered by net close command, polling failure, command retry failure, routing network close, etc
- when network is being closed, the interface state transitions from closing to streamclosed then closed
- when interface state becomes closing, network state is set to sync
- when interface state becomes closed, network state is set to new

command fails 4
- network state is set to error if the number of failed commands exceeds threshold (3 by default)

background poll fails 5
- if background poll fails, the network state will be set to error then the network will be closed shortly after

command succeeds 6
- network state is set to ok if a command is sent and responded

net sync/bg sync succeeds 7
- when network is in the ok state, 'net sync' command or background sync does not change the network state to sync and the network state will remain ok if the sync succeeds

other notes: CG-1589

4.9.3 Networks In Error

This section describes how C-Gate responds to error conditions in the network's state or interface state.

State

If C-Gate is having difficulties communicating effectively with a network, it will place the network into an *error* state (see Network State Diagram).

The following will cause C-Gate to place the network in an *error* state:

- A crucial C-Bus command fails during a network sync, for example:
  - the gateway unit fails to initialise,
  - an Installation MMI fails, or
  - an Application MMI fails.
- Too many units fail to sync during a network sync (see network.error.units-failed).
- Too many consecutive failed C-Bus commands to the network, during a network sync or otherwise (see network.error.commands-failed).

Note that for all of the above C-Gate's command retry mechanism is in effect so it would take more than a single transmission failure to place a network into error.

C-Gate attempts to recover a network in *error* by setting the next sync time for the network to be 5 minutes later. Once the network successfully syncs and returns to an *ok* state the sync intervals will also return to normal (as defined by sync-time).

InterfaceState

If C-Gate cannot communicate with a network at all, it will close the network (see Network Interface State Diagram).

Things that will cause C-Gate to close the network:

- A failure while opening the network (eg. port does not exist or is in use).
- A failed PCI Polling operation.
- A failed PCI Connection Check operation.
- A C-Bus Send Error.

If auto-reopen is set to true C-Gate will attempt to re-open the closed network again in 15 seconds. The next sync time is also set to be in 15 seconds which means that when the network successfully re-opens it will also sync again.

4.9.4 Network Syncing

A network sync reads data from the C-Bus Devices on the C-Bus Network in order to build a model of the network in C-Gate's memory.

Once the first network sync is complete, C-Gate clients are able to understand and manipulate the network through C-Gate.

There are two types of syncing: background syncing and demand syncing.

Background Syncing

C-Gate performs a background network sync when a network is first opened and again at regular intervals as determined by the sync-time option.

Because syncing generates a lot of traffic, C-Bus manages the background syncs with a priority queue as follows:

1. Only 25 networks can be synced at any one time (see sync.global-pool-size option).
2. Only 3 networks can be synced at any one time through a single gateway (see sync.gateway-pool-size option).
3. Networks that have not yet been synced are synced first.
4. Networks that have recently failed to sync or that have gone into error are re-attempted after 5 minutes.
5. Finally, networks that are due for another background sync.

As a result large sites may see some networks not being synced as soon as they are opened, however they will sync eventually.

The CBusNetwork property *NextSyncTime* will indicate when the next background sync is estimated to take place.

WARNING: Reducing *sync-time* or increasing the *pool-size* options may cause performance issues.

Demand Syncing

Demand syncs are initiated by a C-Gate client using commands such as NET SYNC. For example: in C-Bus Toolkit when the user clicks "Scan".

In order to provide responsiveness to clients C-Gate allows demand syncs to bypass the traffic management criteria used for background syncs.

If a demand sync is initiated when there is a background sync already underway for a network, C-Gate will return a cached result of the sync so far and then return the remainder of the sync in real-time.

WARNING: Avoid using demand syncs where possible as they may cause performance issues.

Sync Duration

The duration of a network sync is primarily determined by the number of units on the network.

As a guideline: approximately 25 units are synced per minute. Thus a network of 75 units will take about 3 minutes to sync.

Bear in mind that other factors influence the time taken to sync a network:

- the number of bridges to the network (longer round-trip times)
- the number of older units pre-2006 (commands are not supported and C-Gate falls back to legacy commands)
- unusual gateway units (e.g. B&W C-Touch requires a special initialisation)
- units without mains power (e.g. some commands to MRA units will time out)

- other concurrent syncs (C-Bus traffic bottlenecks)
- high operational traffic (C-Bus traffic bottlenecks, C-Gate PC bottlenecks)
- reverting to the original serial library in use before Toolkit 1.20.0 (new library is faster)

Sync Stages

A sync proceeds through several stages. These are indicated in the log as `SyncSubState` values.

As a guideline:

• `pci`	Gateway initialisation	(1 second)
• `mmi`	Unit presence; and	(1 second)
•	Unit duplicate check	(15 seconds per 25 units)
• `tvs`	Unit Identification	(15 seconds per 25 units)
• `bridge`	Bridge sync	(1 second)
• `units`	Unit sync	(30 seconds per 25 units)
• `app`	Application sync	(1 second per application)

History

- C-Gate v2.9.0 (along with C-Bus Toolkit 1.12.0) introduced permanent background syncing, ie. the *autosync* property can and should be left enabled. This makes it possible to run C-Bus Toolkit, Schedule Plus and other clients side by side. Before this release these clients would fight over this property.

4.9.4.1 Calculation of NextSyncTime

When C-Gate completes a network sync it calculates a new value for the network's NextSyncTime property.

- If the sync FAILED, the NextSyncTime is set to a time five minutes in the future.
- If the sync was a SUCCESS, the NextSyncTime is set to a time in the future as determined by:
  - the sync-time config option,
  - the Sync Padding feature, and
  - any Sync-Free Periods defined for the network.

4.9.4.2 Sync Padding

The Sync Padding feature aims to prevent two or more networks from beginning their sync at the same time by spacing them out at intervals.

It is used only when C-Gate calculates a new NextSyncTime value upon successful completion of a sync.

Sync Padding does not affect:

- the first sync of any network,
- any sync requested by a C-Gate client (eg NET SYNC),
- the value of NextSyncTime manually set by a C-Gate client, or
- the NextSyncTime value that C-Gate calculates following a failed sync.

C-Gate calculates the padding interval between syncs, as follows:

Padding Interval = <project sync-time> * <factor>
<number of networks>

Minimum value : 5 seconds
Maximum value: 300 seconds (5 minutes)

When C-Gate calculates a new NextSyncTime value it will check to see if any other networks already have a NextSyncTime set within the padding interval. If so then the new NextSyncTime will be increased until the desired padding interval is achieved.

Configuration

Next Sync Time Padding is enabled by this option, which has a default value of yes.

```
sync.padding.enabled=yes
```

The following option provides a factor which is applied to the *sync-time* to determine a period of time within which to distribute syncs.

```
sync.padding.sync-time-factor=0.75
```

Any factor of less than 1.0 has the effect of providing a "free" buffer zone at the end of the *sync-time* which provides some allowance for newly created networks to enter the sync rotation easily.

Note that if networks have been configured with individual *sync-time* values, those values are ignored for this calculation. Only the project's sync-time value is used. But the networks will continue to sync according to their configured *sync-time* value.

Example

If:

- the project's sync-time is the default value of 3600 seconds (1 hour),
- the factor is the default value of 0.75 (75%), and
- there are 10 networks in the project

Then:

- the padding interval will be 270 seconds (4.5 minutes), and
- the networks will be distributed evenly across a 45 minute period leaving 15 minutes of allowance.

History

- C-Gate v2.11.0 implemented the Next Sync Time Padding feature.

4.9.4.3 Sync-Free Periods

The Sync-Free Periods feature enables C-Gate to postpone background syncs during times known for high levels of ambient C-Bus network traffic.

It is used only when C-Gate calculates a new NextSyncTime value upon successful completion of a sync.

Sync-Free Periods are not obeyed for:

- the first sync of any network,
- any sync requested by a C-Gate client (eg NET SYNC),
- the value of NextSyncTime manually set by a C-Gate client, or
- the NextSyncTime value that C-Gate calculates following a failed sync.

When C-Gate calculates a new NextSyncTime value it will check to see if it falls into a configured Sync-Free Period. If so then the new NextSyncTime will be increased until it falls outside of the Sync-Free Period. If Sync Padding is enabled it will also be applied so that multiple network syncs don't cluster at the end of the Sync-Free Period.

Configuration

The Sync-Free Periods are controlled by this option:

`sync-free.periods`=

By default, this option is blank meaning that there are no sync-free periods defined.

See the linked page for the config option for information on how to define the periods.

Example

To prevent background syncs in an office when people are starting work and triggering lots of groups and scenes:

```
sync-free.periods=W:0111110:0800-0930
```

History

- C-Gate v2.11.0 implemented the Sync-Free Periods feature.

4.9.5 Unit Syncing

About Unit Syncing

Unit syncing ensures that the C-Gate model of units reflects the units on the network. It is:

- done automatically for every unit as part of a network sync.
- initiated by clients using DO <unit> SYNC | PSYNC | QSYNC commands.

Unit sync states

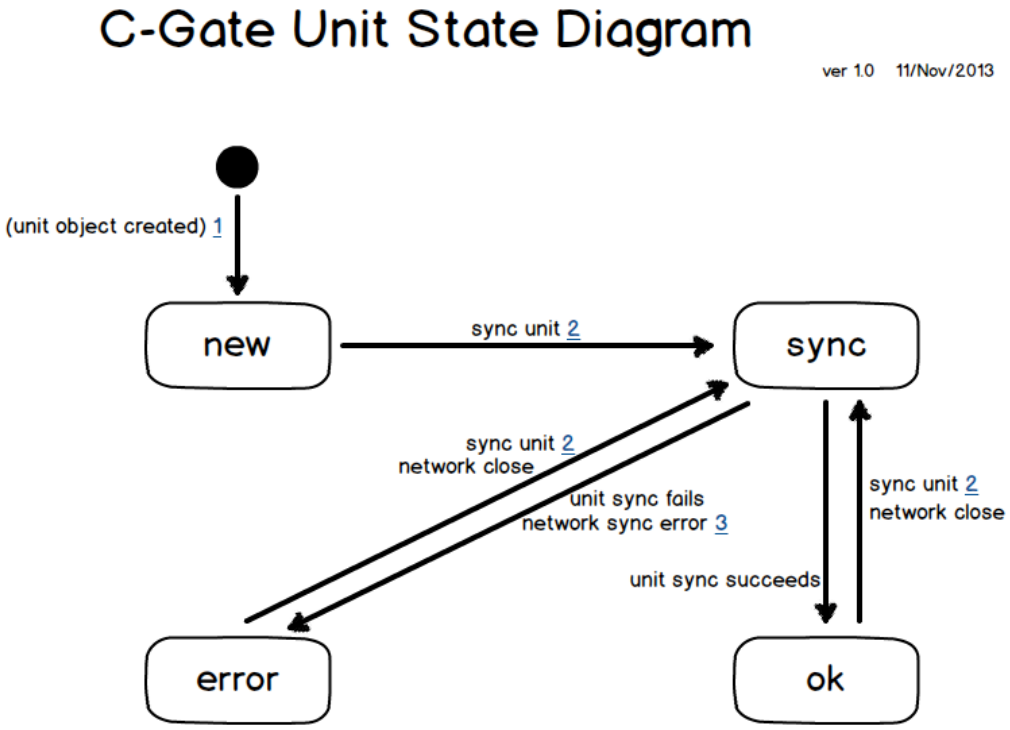
Each unit has a state which indicates whether the unit has been synchronised or not. The states are:

`new`: no sync has been performed
`sync`: unit synchronisation is in progress
`error`: an error occurred while attempting to sync
`ok`: the unit is synchronised and operating normally.

The goal of the sync process is for the unit to end up in the `ok` state.

See the following Unit State Diagram for a diagram of these states.

4.9.5.1 Unit State Diagram



(unit object created) 1
- unit object is created automatically by C-Gate when the unit is first seen during network sync
- unit state never goes back to new

sync unit 2
unit is synced
- for 'do unit_address psync' command (only allowed after the unit object is created during a network sync)
- after the local PCI unit's parameter is changed the PCI unit is re-synced
- as a part of the network sync (note: unit is not re-synced during a network sync if its last sync time is less than PSyncTime ago. PSyncTime can be changed for each unit using set command and it is 5 minutes by default)

network sync error 3
- network sync error means that the network sync completes and there are errors during the sync but the error count does not exceed the threshold to fail the entire network sync

4.9.6 Command Retries

This section describes how C-Gate retries C-Bus commands.

C-Gate Retries

The majority of C-Bus commands sent by C-Gate to a C-Bus network are subject to a time-out period defined by the response-delay option. If a command fails to receive a response in this time it will be retried.

The maximum number of retries is defined by the network.retries option. After all retries are exhausted, the command is deemed to be failed.

While retrying commands, C-Gate may execute a PCI Connection Check to check that the gateway unit is working.

PCI Retries

The C-Bus gateway unit has an internal retry mechanism where it will retry commands that have failed at some point on the C-Bus network between the gateway and the destination unit. Typically each command is attempted up to three times by the gateway. This retry mechanism is not visible to C-Gate nor its clients but may be seen when viewing raw C-Bus messages. It is only mentioned in this document in order to clearly distinguish those retries from C-Gate's retries.

4.9.7 C-Bus Clock Recovery

The C-Bus protocol requires one unit to act as the clock generator (not to be confused with the Clock and Timekeeping Application).

When C-Gate sends a C-Bus command to the network and receives a packet confirmation code terminator (%) that indicates a missing C-Bus clock, it will:

1. Attempt to enable the C-Bus clock on the gateway unit,
2. Wait one second for the network to perform an automatic clock contention process, and
3. Retry the C-Bus command it originally sent.

Upon completion of this recovery process the gateway should now be the clock generator.

To achieve redundancy consider enabling clock on multiple units. The NET CLOCKS command provides a convenient method to do this.

Configuration

The C-Bus Clock Recovery feature is always enabled.

History

- C-Gate v2.11.0 implemented the C-Bus Clock Recovery feature.

4.9.8 PCI Connection Check

When C-Gate is retrying a failed command, it may first perform a PCI Connection Check.

A PCI Connection check is a single, fast command that is sent directly to the local gateway device to confirm that it is still present and working. This provides C-Gate with a diagnostic point of difference compared to a command which may be failing due to long or slow network connections, excessive traffic, multiple C-Bus network bridges, or a destination unit that is missing.

If a PCI connection check fails, C-Gate places the network immediately into error and abandons the command retry process.

Configuration

The PCI Connection Check is controlled by this option:

```
network.retries.pci-check=1
```

With the default value of 1, C-Gate will perform a PCI connection check before every retry.

With a value of 2, C-Gate will perform the first retry, then perform a PCI connection check before the second and subsquent retries.

Setting this value to zero will turn off the PCI connection check entirely. Note that doing this will reinstate the legacy behaviour where C-Gate would spend a long time retrying commands before realising that a network has gone offline.

History

- C-Gate 2.8.0 build 2309 implemented the PCI Connection Check feature.

4.9.9 PCI Polling

PCI Polling is a feature where C-Gate sends a command to the gateway device at regular intervals. This achieves two things:

1. It confirms that the gateway device is still present, and by assumption so is the C-Bus network.

   If the command sent to the PCI receives no response then C-Gate assumes that the network has gone offline and closes the network, placing it in a state where C-Gate will regularly try to re-open it.

   This solves the historical problem that if a C-Bus network goes offline and there are no outgoing commands C-Gate will not notice the network's absence until the next scheduled sync which could be up to an hour later.

2. It updates the Unit Online Status record.

   This is a bonus effect which indicates to clients whether units have been added or removed without having to wait for the next sync to take place.

Configuration

PCI Polling is controlled by these options:

```
network.pci.poll-interval=60    # poll every 60 seconds
reopen-delay=15000              # attempt to reopen a closed network after 15
seconds
```

With this default configuration it will take up to 1 minute for C-Gate to close a missing network, and up to 1 minute to re-open it once it comes back.

Setting the poll-interval to zero will turn off PCI polling entirely. Note that this will reinstate the legacy behaviour where offline networks may not be detected for up to an hour.

Troubleshooting

1. If using a CNI2 through a firewall C-Gate may take up to 15 minutes to recognise that it is back online. This is because the CNI2 is still trying to "keep-alive" the original C-Gate connection and is preventing C-Gate from re-connecting. Some firewalls are known to block the XON/XOFF messages that the CNI2 relies on to detect the dropped connection. This problem doesn't exist with the original CNI because it uses a simpler networking stack.

2. If PCI Polling commands fail regularly it may be due to excessive traffic (especially if you have lots of networks). Try increasing the poll interval to 120 or 300, or set it to zero to turn it off entirely.

History

- C-Gate v2.9.5 introduces the PCI Polling feature (using Identity 0x01 during times of C-Bus inactivity).
- C-Gate v2.10.0 changes the polling command to use an Installation MMI, executed every 60 seconds regardless of C-Bus activity.
- C-Gate v2.10.4 suspends the PCI polling during syncs and prevents it from falling back onto a PCI connection check.
- C-Gate v2.11.0 will fall back to an Identity command if the Installation MMI fails (which sometimes happens on bridged networks).
- C-Gate v2.11.5 suspends the PCI polling during any live traffic and further improves stability.

4.9.10 Unit Online Status

Unit Online Status is a feature where C-Gate tries to provide timely information about which unit addresses are occupied, and where discrepancies exist between the network and the database.

It does this by providing several properties on the network object:

```
DBUnitAddressesNew=31
DBUnitAddressesOnline=1,2,3,6,8
DBUnitAddressesDuplicate=20
DBUnitAddressesError=5
DBUnitAddressesMissing=55
```

These properties are described in detail in the CBusNetwork section.

These properties are updated by:

- a network sync operation.
- a PCI Polling response. If polling changes one of these properties C-Gate will also emit an event message describing the change.

Calculation Method

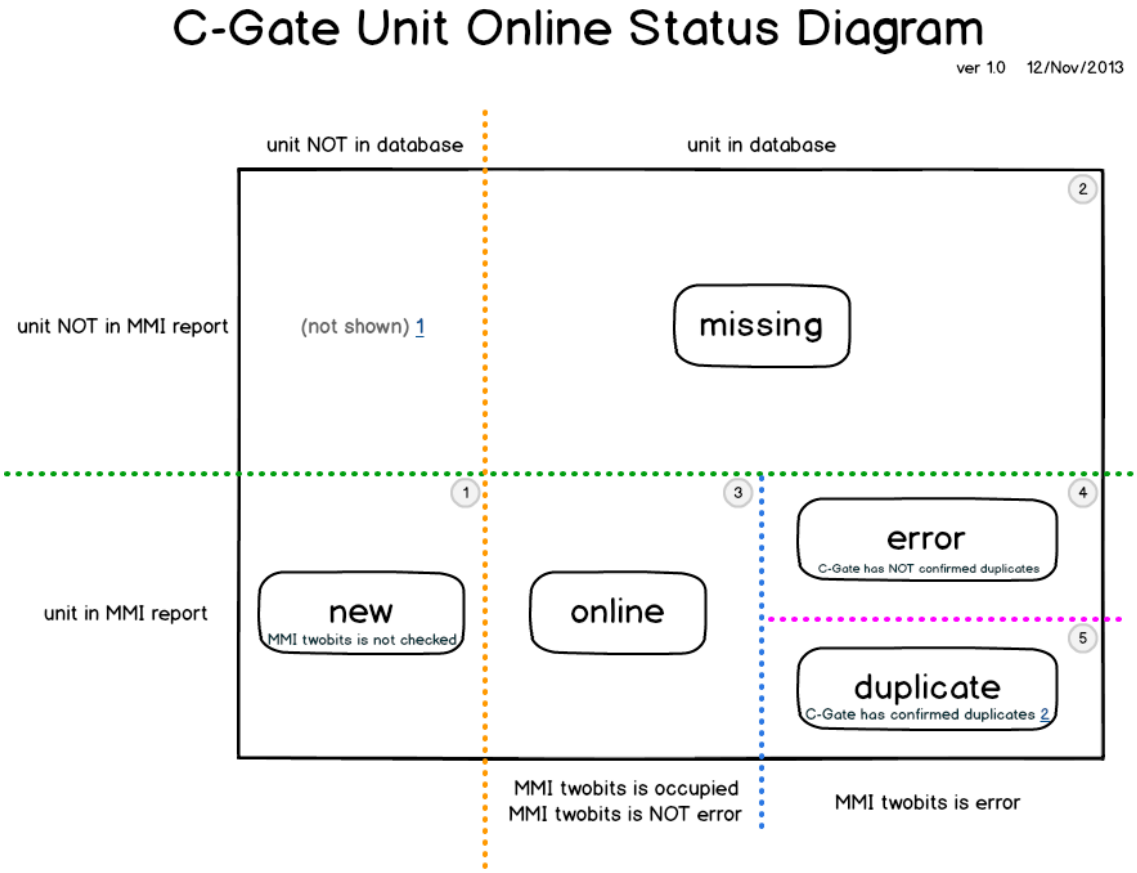
These properties are determined using a complex algorithm. See the following section Unit Online Status Diagram for a graphical representation.

In technical terms C-Gate uses any responses that it receives to Installation MMIs. C-Gate keeps track of the last five MMI responses in order to increase confidence in its duplicate detection - as a result some properties may not change until several PCI pollings have taken place.

History

- C-Gate v2.10.0 introduces the Unit Online Status properties.

4.9.10.1 Unit Online Status Diagram



IMPORTANT NOTE

- unit online status is only valid when the network interface state is running
- unit online status can be retrieved using TREEXMLDETAIL command or from the below network properties
  - DBUnitAddressessNew
  - DBUnitAddressessOnline
  - DBUnitAddressessMissing
  - DBUnitAddressessError
  - DBUnitAddressessDuplicate

(not shown) 1

- if a unit is not in database or MMI report it won't be shown in the TREEXMLDETAIL command response or the DBUnitAddresses* network properties

C-Gate has confirmed duplicates 2

- C-Gate confirms duplicates during network sync or unravel by checking the number of replies from the same address

Reference:

C-Gate Manual >> Reference >> Object Overview >> CBusNetwork

Hints for reading diagram: read from 1 to 5

4.10 Access Control

The C-Gate server provides access control which applies to C-Gate's Command and Monitoring Interfaces. Access control is defined by an in-server-memory access control table that is loaded from the access control file in the config directory.

4.10.1 Access Control Basics

The access control file is normally called `access.txt` , but can be named differently if the 'access-control-file' value in the C-Gate configuration file is altered.
If the file doesn't exist when it is first needed (eg on a login command), a default version is automatically written to disk.

The access control list is a list of lines based on three keywords:

1. `interface` defines an access level for a TCP/IP interface on the C-Gate server
2. `remote` defines an access level for a remote connecting IP address
3. `user` defines a user name & password and access level which can be set with the LOGIN command

When read in, any faulty lines in the file are ignored.

The access control list in memory is checked every time a connection is made to C-Gate command, event, load or status port, or a login command is received.
IP addresses are resolved on the fly at that time, not when the file is read.

At these times, the list of rules in the access control list is reviewed, and the access level granted is the result of the most **favourable** rule. However, using the login command, it is possible to set your access level to a lower level than would normally be configured by the interface and remote rules.

Access levels are as follows, with each later level incorporating the functions of the previous level.

`none` - no accesss at all. Use this to refuse connections.

`connect` - allow a connection to be established (to the command interface only) and execute the LOGIN command or the license challenge & response commands.

`monitor` - allow monitoring and query of the status of objects and C-Bus, but do not allow any changes. This is the minimum level to allow connection to the event, load or configuration change ports.

`operate` - allow set, on, off, ramp operations – allow changes to be made to the system

`admin` - allow C-Gate shutdown and administration functions

`program` - allow C-Bus networks to be programmed

`debug` - allow debugging functions to be performed

Possible entries in the access control list are as follows:

```
interface <IP address> <access level>

remote  <IP address> <access level>
```



```
user <userid> <password> <access level>
```

Where the fields are:

`<IP address>` a full IP address, either for the interface that the connection comes in on, or a remote address. This can be a hostname, or it can be a dotted quad (ie. `10.234.2.23`). If a dotted quad is used, then any part of the address that is set to `255` will match any address, allowing whole networks to be allowed access.

`<userid>` a username that is used in the login command along with the password.

`<password>` the password that goes with the username.

Sample access control file

A sample access control file would be:

```
#sample access control file
interface 127.0.0.1 admin
interface 10.50.1.2 monitor
remote 10.50.1.255 operate
remote microsoft.com none
user debugger xxAA1 debug
```

Blank lines and those begininning with # are ignored.

This file would set access control in the following way:
1. All sessions from the local computer on the *localhost* interface get *admin* level access.
2. All sessions on the ethernet interface 10.50.1.2 (ie all accesses from the external network) get *monitor* level
3. All sessions on the 10.50.1.* subnetwork get *operate* access
4. All connect attempts from the URL *microsoft.com* are denied.
5. A user that types the command:
   login debugger xxAA1
is granted *debug* access level.

4.11 CGL Format

CGL is a data exchange format, nominally a JSON string, that allows a subset of data, chiefly tagnames, to be exported from a C-Gate project, and imported back into a C-Gate project.

The current specification is v1.0.

This format is used by the CGL command set.

4.11.1 CGL Specification v1.0

4.11.1.1 CGL Specification v1.0 Example

```
{
  "cglVersion": "1.0",
  "createdBy": "C-Gate v2.11.0 (build 9999)",
  "createdTime": "2016-07-22T00:18:17.408+0000",
```



```
"networks": [
  {
    "address": 254,
    "name": "n254",
    "routes": [
      {
        "address": 253,
        "path": [ 253 ]
      },
      {
        "address": 252,
        "path": [ 253, 252 ]
      }
    ],
    "applications": [
      {
        "address": 56,
        "type": 56,
        "name": "C-Bus Lighting Application",
        "groups": [
          {
            "address": 1,
            "name": "Kitchen",
            "levels": [
              {
                "address": 255,
                "name": "On"
              }
            ],
          }
        ]
      }
    ]
  }
]
}
```

4.11.1.2 CGL Specification v1.0 Constraints

Property	Sending System Constraints	Receiving System Validations
<all>	Names are in Lower Camel Case (ie. lowerCamelCase).	Properties can be accepted in any order, e.g. “address” may come before or after “name”. If a property is missing it is assumed to have a value of *null*.
cglVersion	A string in “x.y” format where x and y are positive integers. Must be in a known subset, currently { 1.0 }. {0.y} can be used freely for development.	ERROR if major value (x) is unsupported. WARNING or INFO if minor value (y) is unsupported.
createdBy	An arbitrary free-text string describing the system or	None. For informational purposes only.

	software that generated the CGL. Include useful information such as version.	
createdTime	A timestamp in ISO-8601 basic format (no punctuation) to a resolution of seconds and including the timezone offset of the sending system. The inclusion of milliseconds is optional. Example: *20160301T163344 +1030*.	ERROR if invalid timestamp. WARNING if the timestamp is in the future.
Network instances	Up to 255 instances.	
Network “address”	Integer in range 0..254. Must be unique within the “networks” collection.	ERROR if invalid syntax. WARNING if non-unique. Duplicate values are ignored.
Network “name”	A free-text string. Must be unique within the “networks” collection.	
Route instances	Up to 255 instances.	
Route “address”	Integer in range 0..255. Must be unique within the “routes” collection. This is the destination network address.	ERROR if invalid syntax. WARNING if non-unique, and duplicate entries are ignored.
Route “path”	Array of integers, each in the range 0..255. Each integer is an interim network address on the route towards the desination network. The order is important. The first integer is the first and nearest network, and the last integer must always be the destination network itself. No part of the route is allowed to form a loop.	ERROR if the route contains an invalid address. ERROR if the route contains a loop.

Application instances	Up to 256 instances.	
Application “address”	Integer in range 0..255. Must be unique within the “applications” collection.	ERROR if invalid syntax. WARNING if non-unique, and duplicate entries are ignored.
Application “type”	Integer in range 0..255. This defines the known “behaviour” of the Application and is effectively the default address from the standard list of C-Bus Applications. For example, a second Lighting application with address 57 would have a type of 56. However in most cases it would be the same value as “address”.	
Application “name”	A free-text string. Must be unique within the “applications” collection.	
Group instances	Up to 256 instances.	
Group “address”	Integer in range 0..255. Must be unique within the “groups” collection.	ERROR if invalid syntax. WARNING if non-unique, and duplicate entries are ignored.
Group “name”	A free-text string. Must be unique within the “groups” collection.	
Level instances	Up to 256 instances.	
Level “address”	Integer in range 0..255. Must be unique within the “levels” collection.	ERROR if invalid syntax. WARNING if non-unique, and duplicate entries are ignored.
Level “name”	A free-text string. Must be unique within the “levels” collection.	

4.12 Object Overview

This section gives an overview of the parameters and methods of C-Gate objects, including C-Bus objects that represent networks and devices in connected and controlled C-Bus networks.

In general, all object parameters can be retrieved from the command interface using the GET or SHOW command. Some parameters can be set with the SET command.

Methods given here can be invoked from the command interface using the DO command.

4.12.1 Working with Objects

Any addressable object in C-Gate contains one or more parameters and zero or more methods.

Parameters

Parameters are values that can be viewed and possible set from the command interface. In many cases, where the objects in C-Gate are modelling or mirroring the phyiscal or logical devices on a C-Bus network, the parameters represent the latest information retrieved from the actual physical or logical devices.

The GET command makes getting parameters easy.

The SET command allows some parameters to be set to new values from the command line. In the following tables, if the SET access level is shown as *None*, then the set operation will fail.

Methods

Methods allow the command line to be used to cause objects to do something.

The DO command is provided to run object methods.

Exploring Objects

You can explore the parameters provided in any object by using a special form of the GET command.

```
get <obj-add> *
```

which returns a list of parameter names

For example, for a C-Bus Network:

```
get 254 ?
300 //
HOME/254:Parameters=Interface,AutoSync,Retries,AutoUpdate,Options,Type,
State,InterfaceState,AutoUnravel,DefaultApplication,SyncState,QuickDetec
t,LSP,RxQ,ShortSync,SyncTime,TxQ,FastResponse,ResponseDelay,TxEnable,Eve
ntLevel,Units,FreeUnit,FreeApplication,InterfaceAddress,Groups,TargetInt
erfaceState,Stats,Applications,XState,Name
```

To get a little more information, try:

```
get <obj-add> ??
```


returns a list of parameter names with short explanations

For example, again for a C-Bus Network:

```
get 254 ??
102-//HOME/254: Interface - Address of the interface for this network
102-//HOME/254: AutoSync - If set to yes, will perform automatic network
synchronisations.
102-//HOME/254: Retries - The number of retries made before a command
fails
102-//HOME/254: AutoUpdate - If set to yes, automatically update the tag
database for a new unit.
102-//HOME/254: Options - The interface options for this interface
102-//HOME/254: Type - Type of network interface
102-//HOME/254: State - The state of connection of this object
102-//HOME/254: InterfaceState - The current state of this interface
102-//HOME/254: AutoUnravel - If set to yes, will perform automatic
unravels when errors are detected in sync operations.
102-//HOME/254: DefaultApplication - The default application for this
network
102-//HOME/254: SyncState - Get the sync state for this network
102-//HOME/254: QuickDetect - If set to yes, perform continuous install
MMIs looking for new units.
102-//HOME/254: LSP - The count of Lost Sync Packets
102-//HOME/254: RxQ - List of the commands waiting for responses
102-//HOME/254: ShortSync - If set to yes, perform only sync to type-
version-serial.
102-//HOME/254: SyncTime - The time  between full network sync
operations
102-//HOME/254: TxQ - List of the commands in the transmit queue
102-//HOME/254: FastResponse - If set to yes, retries and timeouts are
shortened to give quick responses.
102-//HOME/254: ResponseDelay - Time in ms to delay before a command is
retried.
102-//HOME/254: TxEnable - True if the transmitter is enabled
102-//HOME/254: EventLevel - The level of displayed events for this
object
102-//HOME/254: Units - The list of units that are in this network
102-//HOME/254: FreeUnit - The next free unit address on this network
102-//HOME/254: FreeApplication - The next free application address on
this network
102-//HOME/254: InterfaceAddress - Address of the interface for this
network
102-//HOME/254: Groups - The list of groups in the DEFAULT application
102-//HOME/254: TargetInterfaceState - The current state of this
interface
102-//HOME/254: Stats - List of statistics for this interface
102-//HOME/254: Applications - List of applications on this network
102-//HOME/254: XState - Get the extended state for this network
102 //HOME/254: Name - Name of this network
```

To quickly get a list of all parameters and their values, try:


```
get <obj-add> *
```

For example:

```
get 254 *
300-//HOME/254: Applications=56,228
300-//HOME/254: AutoSync=yes
300-//HOME/254: AutoUnravel=no
300-//HOME/254: AutoUpdate=no
300-//HOME/254: DefaultApplication=56
300-//HOME/254: EventLevel=9
300-//HOME/254: FastResponse=no
300-//HOME/254: FreeApplication=0
300-//HOME/254: FreeUnit=0
300-//HOME/254: Groups=1,2,10,44
300-//HOME/254: Interface=localhost:14000
300-//HOME/254: InterfaceAddress=localhost:14000
300-//HOME/254: InterfaceState=running
300-//HOME/254: LSP=0
300-//HOME/254: Name=1
300-//HOME/254: Options=
300-//HOME/254: QuickDetect=no
300-//HOME/254: ResponseDelay=3000
300-//HOME/254: Retries=2
300-//HOME/254: RxQ=now=1138600828749
300-//HOME/254: ShortSync=no
300-//HOME/254: State=ok
300-//HOME/254: Stats=:SendCommand=114
300-//HOME/254: SyncState=units
300-//HOME/254: SyncTime=300
300-//HOME/254: TargetInterfaceState=running
300-//HOME/254: TxEnable=yes
300-//HOME/254: TxQ=now=1138600828749
300-//HOME/254: Type=cni
300-//HOME/254: Units=2,10,14,15,19,249
300 //HOME/254: XState=ok+units
```

4.12.2 All Objects

All C-Gate objects support the following object parameters. These can be retrieved with the GET and if allowed, can be set with the SET command.

All-Objects Parameters:

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
EventLevel	Values range from 0 (no events) to 9 (all events). Events that are at a level lower to or equal to the value of **event-level** will be sent.	The value of the cgate global event-level parameter at the time this	monitor	operate

		object is created. This is the parameter that is provided in the global configuration.		
State	Represents the state of an object in C-Gate. The following states are possible: **unknown**: The state of this object is not known. **new**: This is a new object that is not synchronised with a real device. This state occurs when an object has been created with the NEW command but is yet to synchronise with a real device. **sync**: This object is in the process of synchronising with a real device. It may not contain valid data in this state. **ok**: Object in state of normal operation, synchronised to external controlled objects. **error**: This object is in an error state and its data may not reflect actual status. **deleted**: This object has been deleted.	A new object is created in state **new**.		

All-Objects Methods:

Method name	Access Level	Parameters	Description
Gc	operate	None	Performs a manual global garbage collection for C-Gate. This operation is provided for debugging purposes only.

4.12.3 CGate Object

The C-Gate system object **cgate**, hold the version of this C-Gate server. The **`get cgate Version`** command is often used to check version compatibility, though the apiver command gives greater detail.

***cgate* object parameters**

Parameter	Values and impact	Default value	GET Access Level	SET Access Level

Version	The current version of this C-Gate	None	monitor	None

4.12.4 Projects

The C-Gate system object **projects**, hold some properties common to all open projects.

The **`get projects *`** command will return the following parameters in addition to those available for All Objects.

***projects* object parameters**

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
NetStateInterval	The interval in seconds between state events for all networks. A value of zero disables state events.	0	monitor	operate

4.12.5 CBusNetworkManager

The C-Bus network manager, system object **cbus**, controls the definition, opening and operation of attached C-Bus networks.

***cbus* object parameters:**

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
Networks	A list of the current know C-Bus networks, comma separated.	empty if no networks are defined	monitor	None

4.12.6 CBusNetwork

Inherits all the parameters and methods from: All Objects

Network object parameters:

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
Applications	Returns a comma-separated list of the C-Bus application numbers in use on this network	empty if no applications are defined	monitor	None

Type	The type of C-Bus network interface this is. Possible values are: **serial**: local serial port interface to C-Bus PCI **socket**: TCP/IP socket interface to C-Bus PCI **cni**: Clipsal C-Bus Network Interface (provides direct TCP/IP connection to C-Bus). **wiser**: same as type **cni**. **etherlite**: Interface to C-Bus PCI via Etherlite module.	No default, defined at interface startup time in networks file.	monitor	None
Interface	A string that defines the interface address and port name. For **serial** type, the string is a COM port identifier, for **socket** and **etherlite or cni** types, the string is in the form: ip-address:port-number, such as: 10.1.1.66:2001	No default	monitor	None
InterfaceState	A string that defines the current state of this interface to the C-Bus network. Possible values and their meanings are: **closed**: The interface has not been opened or has closed. **opening:** The interface has been opened (but is not yet fully operating). **running:** The interface is operating normally. **closing:** The interface is in the process of closing. **streamsclosed:** The interface is in the process of closing.	closed	monitor	None

Network object parameters (continued):

Parameter	**Values and impact**	**Default**	**GET**	**SET**

		value	Access Level	Access Level
Name	The name of this interface. This is equivalent to the network name specified when the C-Bus network is defined.	No default	monitor	None
Units	A list of the C-Bus unit numbers that are known on this network.	No default	montior	None
Groups	A comma-separated list of the C-Bus group numbers **for the default application** of this network.	No default	monitor	None
TxQ	[For debugging only] A list of commands that are queued in the transmit queue waiting to be sent	No default	debug	None
RxQ	[For debugging only] A list of the commands that have been sent and are waiting for a response or confirmation.	No default	debug	None
Stats	Basic statistics for this C-Bus interface	No default	debug	None
FreeUnit	Returns the unit address of the next free unit address.	No default	monitor	None
FreeApplication	Returns the application address of the next free application address	No default	monitor	None
SyncTime	Time (in seconds) between synchronising the local objects and the actual C-Bus networks. (NB, the check for sync-time is only performed at the interval of **scan-time**.	300	monitor	operate

Network object parameters (continued):

Parameter	Values and impact	Default	GET	SET

		value	Access Level	Access Level
ScanTime	Time (in seconds) between checks of psync and sync times. Every **scan-time** checks are made if the **sync-time** has been exceeded, if so, a sync is performed for this network. Also, the **psync-time** is checked for each unit in this network and psyncs are performed on those objects if required.	30	monitor	operate
LSP	Returns the count of lost sync packets. That is, the number of responses that are returned that are not matched to a sent command waiting for a response.	No default	monitor	None
DefaultApplication	The default application number for this network. The default application can be addresses with shorthand and version 1.0 addressing.	56, or $38 (The generic lighting application)	monitor	operate
TxEnable	Returns 'yes' if the transmitter is currently enabled for this network. This is generally used for testing purposes only.	no	monitor	None
SyncState	Returns the state of synchronisation of the network. Valid values are: `idle` - no sync in progress; `init` - sync started; `mmi` - inital network poll done; `tvs` - type, version and serial numbers gathered; `pci` - pci scan complete; `bridge` - bridge scan complete; `units` - full unit	idle	monitor	None

	sync complete; `app` - application scan complete.			

Network object parameters (continued):

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
XState	Returns a combination of network and sync states separated by a `+` character.		monitor	None
Retries	Set the number of command retries before giving up the command as failed.	2	monitor	operate
ResponseDelay	Time in milliseconds to wait for a response to the command from the network	3000 (3 seconds)	monitor	operate
FastResponse	On setting to "yes", sets Retries to 0. On setting to "no" sets Retries to 0.	no	monitor	operate
AutoSync	If set to yes, will perform automatic network synchronisations.	yes	program	program
NextSyncTime	The next scheduled sync time for this network	No default	monitor	program

Network object parameters (continued):

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
DBUnitAddressesNew	Only valid when **InterfaceState** is running; List of unit addresses which are not in project database but are occupied in MMI report, e.g. `DBUnitAddressesNew=31`	No default	monitor	readonly
DBUnitAddressesOnline	Only valid when **InterfaceState** is running; List of unit addresses which	No default	monitor	readonly

	are in project database and are occupied in MMI report and the MMI statuses are not error, e.g. `DBUnitAddressesOnline=1,2,3,4,5`			
DBUnitAddressesMissing	Only valid when **InterfaceState** is running; List of unit addresses which are in project database but are not occupied in MMI report, e.g. `DBUnitAddressesMissing=10,21`	No default	monitor	readonly
DBUnitAddressesError	Only valid when **InterfaceState** is running; List of unit addresses which are in project database and are occupied in MMI report and the MMI statuses are error and C-Gate has not confirmed there are duplicates at the addresses, e.g. `DBUnitAddressesError=9`	No default	monitor	readonly
DBUnitAddressesDuplicate	Only valid when **InterfaceState** is running; List of unit addresses which are in project database and are occupied in MMI report and the MMI statuses are error and C-Gate has confirmed there are duplicates at the addresses, e.g. `DBUnitAddressesDuplicate=18,19`	No default	monitor	readonly

Note for DBUnitAddress* parameters:

- C-Gate periodically polls the network gateway using MMI when its InterfaceState is running to check the network connection and unit online statuses.
  The poll interval is defined by network.pci.poll-interval. During each poll C-Gate updates the DBUnitAddress* parameters and if any of these parameters change, it will also send out an 744 event as described in Network Information Event Table.

- Possible transitions during MMI Polling
  - the unit online status can become Error from New, Online, Missing (note: since the MMI two bits is a random number from the unit, possibly you need to wait for several polls to see the status change)
  - the unit online status can become New, Online, Missing from Error/Duplicate (note: there is

a confidence check before the error status is reset and the threshold is 5, that means after you remove a duplicate unit physically from the network and do not perform a network scan, you need to wait 5 polls before the error status disappear)

- Possible transitions during Network Scan/Unravelling
  - o the unit online status can transition among all 5 statuses

Network object methods

Method name	Access Level	Parameters	Description
Sync	admin	None	Perform a network synchronisation immediately. This pre-empts the usual background synchronisation operation that performs sync operations every **sync-time**.
Open	admin	None	Opens the network interface if it is not already open.
Close	admin	None	Close this network interface.
AllOn	operate	None	Sets the state of all unprotected groups in the **default application** to **on.**
AllOff	operate	None	Sets the state of all unprotected groups in the **default application** to **off.**
AllRamp	operate	ramp-level ramp-time (see the RAMP command for the definition of ramp-level and ramp-time)	Ramp all of the unprotected groups in the **default application** to the specified level in the specified ramp time. See the RAMP command for additional details.
Unravel	program	[unit-address] If unit address is given (optional) then the unravel is performed at that unit address only.	Scan, detect and re-address any C-Bus units that have duplicate unit addresses. Progress is reported as a series of 120 responses.

4.12.7 CBusUnit

The C-Bus Unit is a model of a physical C-Bus device connected to a C-Bus network.

Inherits all the parameters and methods from: All Objects

Unit object parameters:

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
Name	The name of this object. (This is not the same as the address and may be set independently of the address)	"unknown"	monitor	operate
Address	The numeric address of this unit. Setting this value will change the C-BusUnit's address on the network. Use this with care.	No default	monitor	program
Application	The numeric value of the primary application for this C-Bus unit.	255 (The value indicating the application is unassigned)	monitor	None
Application2	The numeric value of the secondary application for this C-Bus unit.	255 (The value indicating the application is unassigned)	monitor	None
Type	The C-Bus unit device type string	No default	monitor	None
Version	The version string for this C-Bus unit.	No default	monitor	None
ClassName	The name of the java class that is used by C-Gate to model this unit.	`com.clipsal.cgate.cbus.CBusUnit`	debug	None
PsyncTime	The time between parameter sync operations for this unit. This is measured in seconds. (Note, parameter syncs can not occur more often than the `scan-time` global parameter unless invoked by the `psync` method (see below)	The default value is set by the cgate global parameter *default-psync-time* when the unit is created.	monitor	operate
PatchVersion	The version of patch usd in this unit	$FF means not patch used in unit. -1 means no snycronised yet.	program	None

SlotGroups	The contents of the unit's group slots	$FF is the uninitialized value	monitor	None

Unit object methods

Method name	Access Level	Parameters	Description
Sync	admin	None	Perform a network synchronisation immediately. This pre-empts the usual background synchronisation operation that performs sync operations every **sync-time**.
Psync	admin	None	Perform a parameter synchronisation on this unit. Psync usually fetches operational paramters that need to be sampled more often than syncs are performed.
Qsync	admin	None	Perform a quick synchronisation of selected unit parameters to support commissioning software.
ResetErrorFlags	admin	None	Immediately reset the physical C-Bus Unit's error flag.

4.12.8 CBus Output Units

Inherits all the parameters and methods from: CBusUnit

Output Unit object parameters:

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
Area	The C-Bus Area address that is set for this unit.	255 (this is the unassigned value)	monitor	None
Groups	A comma-separated list of C-Bus groups used by this unit.	None	monitor	None
Terminals	The comma-separated list of terminals supported by this unit	None	monitor	None
TerminalCount	The number of terminals supported by this unit	0	monitor	None

4.12.9 CBus Output Terminal

This object represents an individual output terminal on a C-Bus output unit. Individual terminals represent an individual dimmer or relay contact set.

Inherits all the parameters and methods from: All Objects

Output Terminal object parameters:

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
Level	The current level of this terminal – in the range 0 through 255.	0	monitor	None
Power	The current output power calculated for this terminal based on the value of the **Level** and the **Load** parameters. Note: this is not an accurate power measurement nor is is based upon actual current flowing.	0	monitor	None
Load	The load power of this terminal. This value defaults to 0 and must be set in order for the **Power** parameter to give any value other than 0.	0	monitor	operate
Groups	A comma-separated list of C-Bus groups used by this terminal	None	monitor	None
Logic	The type of logic used on the groups supported by this terminal. Values are: **greater**: The greater of the C-Bus groups will be used to set the output level. **lesser:** The lesser of the C-Bus groups will be used to set the output level.	None	monitor	None
Name	The name of this terminal	None	monitor	operate

4.12.10 DIN and Pro Dimmers and Relays

C-Bus DIN Rail and Professional Series relays have enhanced features over the standard Relay and Dimmer output devices.

Inherits parameters and methods from: CBusOutputUnit

Dimmer object parameters:

Parameter	Values and impact	Default value	GET Access Level	SET Access Level

NetVoltage	The C-Bus network voltage at this unit. This parameter is read when parameter sync (psync) is performed. Execute the psync method on this object to update this value.	None	monitor	None
Serial	The unique serial number of this device	None	monitor	None

4.12.11 CBus Din/Pro Terminal

Inherits from: CBus Terminal

Terminal object parameters:

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
Current	The measured current for this terminal. -1 is returned if no current sense information is available, otherwise, a value is returned from 0 through 255, representing current as a fraction of the terminal's maximum current rating. This parameter is read from the device using the parameter sync (PSync) operation. Execute the **PSync** method on the unit this terminal is attached to in order to update current ratings.	None	monitor	None

4.12.12 CBus Temperature Sensor

Inherits parameters and methods from: C-BusUnit

Sensor object parameters:

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
Temperature	The temperature recorded by this sensor when the last parameter sync (PSync) operation was performed. The temperature is given in degrees Celsius to 2 decimal places. A typical value might be 20.44.	None	monitor	None
Target	The target temperature value for this unit. This value is given in **whole degrees celsius**. Setting this parameter will change the target	None	monitor	operate

	value set in the unit.			
Margin	The margin value for this unit in whole degrees celsius. Setting this parameter will change the margin value stored in the unit.	None	monitor	operate
Offset	The offset value for this unit in whole degrees. Setting this parameter will change the offset value stored in the unit.	None	monitor	operate
Mode	The heating or cooling mode for this unit. Valid values are **heating** or **cooling**. Setting this value will change the value stored in the unit.	None	monitor	operate
High	The high-side offset for this sensor	None	monitor	None
Low	The low-side offset for this sensor	None	monitor	None

4.12.13 CBus PE Cell / Light Level Sensor

Inherits from: CBusUnit

Sensor object parameters:

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
LightLevel	The light level recorded by this sensor when the last parameter sync (psync) was performed. The light level is given in lux and has an approximate range of between 0 and 1600.	None	monitor	None
Target	The target value for this unit. This value is given in lux. Setting this parameter will change the target value set in the unit. Range: 0 – 1600	None	monitor	operate
Margin	The margin value for this unit. Setting this parameter will change the margin value stored in the unit. Range 0 – 1600	None	monitor	operate
RawTarget	The raw, unscaled value of the **Target** parameter	None	montior	None
RawMargin	The raw, unscaled value of the **Margin** parameter.	None	monitor	None

4.12.14 CBusApplication

Inherits parameters and methods from: All Objects

Application object parameters:

Parameter	Values and impact	Default value	GET Access Level	SET Access Level
Name	The name of this object. (This is not the same as the address and may be set independently of the address)	"unknown"	monitor	operate
Address	The numeric address of this application.	None	monitor	None
Groups	A comma-separated list of the C-Bus group numbers for this application.	None	monitor	None
FreeGroup	The address of the first unassigned group address in this application.	None	monitor	None

Application object methods:

Method name	Access Level	Parameters	Description
AllOn	operate	None	Sets the state of all unprotected groups in this application to **on**.
AllOff	operate	None	Sets the state of all unprotected groups in this application to **off.**
AllRamp	operate	ramp-level ramp-time (see the RAMP command for the definition of ramp-level and ramp-time)	Ramp all of the unprotected groups in this application to the specified level in the specified ramp time. See the RAMP command for additional details.

4.12.15 CBusGroup

Inherits from: All Objects

Group object parameters:

Parameter	Values and impact	Default value	GET Access	SET Access

			Level	**Level**
Name	The name of this object. (This is not the same as the address and may be set independently of the address)	"unknown"	monitor	operate
Address	The numeric address of this group.	None	monitor	None
Units	A comma-separated list of units that this group can control *on the local network.*	None	monitor	None
Protected	If set to yes, this group will not be included in AllOn, AllOff and AllRamp methods executed on this group's Application.	None	monitor	None
Level	The current level of this C-Bus group. Level can be in the range of 0 through 255. 0 is considered off. Level 255 in C-Bus is considered on. Setting the level causes a ramp command to be sent with a ramp time of 0 to the level to be set.	None	monitor	operate
Type	The type of C-Bus group. There are two possible values: **group** and **area**.	"group"	monitor	operate
RampTime	The stored ramp time to be used by the CI command when setting the group level. If a group is set with the CI commands, then the ramp time in seconds stored here is used as the ramp time for the level change.	0	monitor	operate

Group object methods:

Method name	**Access Level**	**Parameters**	**Description**
Off	operate	None	Sets the level of this group to 0 by executing a C-Bus **off** command.
On	operate	None	Sets the level of this group to 255 by executing a C-Bus **on** command.
Ramp	operate	ramp-level ramp-time (see the RAMP command (section 3.3.5) for the	Sets the level of this group to ramp-level using a ramp time of ramp-time.

		definition of ramp-level and ramp-time)	
Protect	operate	None	Set the protected parameter for this group.
UnProtect	operate	None	Clear the protected parameter for this group.

4.13 Building Clients for C-Gate

This section has helper tips and documentation references useful when building client software to work with C-Gate.

4.13.1 Communicating with C-Gate

Client software can communicate with C-Gate a number of ways. This is described in detail in Command and Monitoring Interfaces.

To summarise here:

Standard Input, Output, Error on local Host

For a client running on the same host as the server, the client can start C-Gate as a subprocess, and use the standard input, output and standard error to communicate with C-Gate.

TCP/IP sockets

TCP/IP sockets can be used to access the server. Separate connections are available for Commands, Event, Configuration Changes, and Status changes. The standard ports for these are 20023 - 20026. You must make sure that access control will allow remote connections from your client's location.

TCP/IP sockets with SSL

C-Gate also supports connections over SSL when it is deirable that the information communicated between C-Gate and the client is encrypted. Client authentication via SSL client certificates can also be enabled if necessary. The port range used here is 20123-20126.

See Command and Monitoring Interfaces for more information about these communication options.

Multiple client connections using any or all of the above interfaces is supported by C-Gate

4.13.2 Controlling Communications with C-Gate

C-Gate gives clients a pretty rich set of ways to control line-by-line commands with C-Gate.

Simple commands can be issued and a response waited for. See Commands and Replies for basic details of using communications.

In more complex situations, commands can be issued with a unique identifier, and all responses to that command will then be returned with the unique identifier. See Unique Command Ids for details of this.

And in addtion to this, command issued with a unique identifier can be placed into the background so a new command can be issued. This allows multiple commands to be issued and performed at once if the client can use the unique identifiers to match responses to the commands.

4.13.3 Multiple Clients

Because multiple clients can work with C-Gate, it may be useful for clients to communicate with each other.

C-Gate provides a simple mechanism to perform this, the BROADCAST_EVENT command. With this command, you can share and communicate between client connected to the same C-Gate.

4.13.4 Logging with C-Gate

C-Gate logs event information to disk. This can be a very useful debugging technique.

For more detail, refer to the Logging section in the the reference.

5 Version History

This section covers the significant changes in each new version.

5.1 v2.11.6

C-Gate v2.11.6 was released in April 2021 with C-Bus Toolkit v1.15.8.

Changes:

- Cybersecurity patches.
- Fixed issue causing eDLT units to ignore certain bus messages.

5.2 v2.11.5

C-Gate v2.11.5 was released in June 2019 with C-Bus Toolkit v1.15.6.

Changes:

- Support for 40 Series Modules.
- Improvements to the PCI Polling feature.

5.3 v2.11.4

C-Gate v2.11.4 was released in July 2018 with C-Bus Toolkit v1.15.5.

Changes:

- Upgrade Silabs USB drivers.

5.4 v2.11.3

C-Gate v2.11.3 was released in June 2018 with C-Bus Toolkit v1.15.4.

Changes:

- Fix for bridge routing when NAC/SHAC is used as a gateway.

5.5 v2.11.2

C-Gate v2.11.2 was released in February 2018 with C-Bus Toolkit v1.15.3.

Changes:

- New default values for light level target and margin in sensor units.
- Now produces an event for "[trigger] indicatorkill" message.
- New properties for Clock Application.
- Now correctly sets date on bus if C-Gate is configured as the clock master.
- Better handling of corrupted messages on the bus.

5.6 v2.11.1

C-Gate v2.11.1 was released in June 2017 with C-Bus Toolkit v1.15.1.

Changes:

- Fixed issue with eDLT dialogs after Windows 10 Creator's Update.

5.7 v2.11.0

C-Gate v2.11.0 build 3232 was released on 15th May 2017 with C-Bus Toolkit v1.15.0.

Changes:

- Java runtime upgraded from v7 to v8.

- Silabs USB Drivers upgraded from v6.6.1 to v6.7.3 (improved Windows 10 support).
- Support for the Automation Controller (5500NAC, LSS5500NAC, 5500SHAC, LSS5500SHAC).
- New C-Gate Launcher.
- Sync Padding feature.
- Sync-Free Periods feature.
- Automatic Clock Recovery capability.
- NET CLOCKS command.
- PORT CNISCAN2 command.
- Array Filtering syntax.
- PCI Polling improved for bridged networks.
- Command caching fixes.

5.8 v2.10.6_3169

C-Gate v2.10.6 build 3169 was released on 19th April 2015 with C-Bus Toolkit v1.14.5. Also included with C-Bus Toolkit v1.14.6 and v1.14.7.

Changes:

- Added support for EN_UNIV
- Fixed installation issue on Windows 10

5.9 v2.10.6

C-Gate v2.10.6 build 3145 was released on 7th April 2015 with C-Bus Toolkit v1.14.2. Also included with C-Bus Toolkit v1.14.3 and v1.14.4.

Changes:

- Updated unit catalog and unit specifications.

5.10 v2.10.5

C-Gate v2.10.5 build 3140 was released on 9th December 2014 with C-Bus Toolkit v1.14.0. Also included with C-Bus Toolkit v1.14.1.

Changes:

- Java runtime upgraded from v6 to v7.
- Transmit Cache feature.
- Config options to reduce sync traffic (see separate release notes).
- CONFIG OBRESET command.
- PCI Polling fixes.
- Support for Saturn Zen range, including unit types KEYH1, KEYH2, KEYH3, and KEYH4.

5.11 v2.10.2

C-Gate v2.10.2 build 3106 was released on 28th May 2014 with C-Bus Toolkit v1.13.2. Also included with C-Bus Toolkit v1.13.3.

Changes:

- Fixed phantom burdens reported by eDLT and SENTEMP4.
- Fixed Logging performance issues.

5.12 v2.10.0

C-Gate v2.10.0 build 3087 was released on 9th December 2013 with C-Bus Toolkit v1.13.0.

Changes:

- New Logging subsystem.
- PCI Polling now uses an Installation MMI and executes every minute regardless of traffic.
- TEST SPAM commands.
- DBADDSAFE, DBSETSAFE, DBCOPYSAFE and DBRENAMENETSAFE commands.
- TREEXMLDETAIL command.
- sync.global-pool-size and sync.gateway-pool.size config options.
- USB driver updated to v6.6.
- Support for Windows 8 & 8.1.
- Sync/unravel now handles the edge case where the PCI has moved unexpectedly to another address.
- Modifying the network name or interface now flushes the network model. This solves several problems where clients were retrieving outdated data from the model.
- Fixed a bug where messages from a C-Touch or PAC on a local network with bridges were not replicated by C-Gate to the far-side network model.

This and future releases also include a standalone zip distribution of C-Gate which contains some cross-platform support for Mac and Linux operating systems.

5.13 v2.9.8

C-Gate v2.9.8 build 3077 was released on 18th November 2013 with C-Bus Toolkit v1.12.8.

Changes:

- Support for Occupancy Sensors.
- Updated support for eDLT units.

5.14 v2.9.7

C-Gate v2.9.7 build 2569 was released on 17th September 2013 with C-Bus Toolkit v1.12.7.

Changes:

- PCI Polling feature.
- PROJECT REPAIR command can now remove duplicate groups.
- Support for eDLT units.
- Upgraded the USB driver to v6.4.0.2.
- Closing the network on the near side of a bridge now also closes the network on the far side.
- Reconnecting a disconnected network on the near side of a bridge now allows the network on the far side to reopen automatically.
-

5.15 v2.9.5

C-Gate v2.9.5 build 2460 was released on 27th November 2012 with C-Bus Toolkit v1.12.5.

Changes:

- Fixed a bug introduced in v2.9.2 that would cause large scenes executed in Schedule Plus to fail.

5.16 v2.9.4

C-Gate v2.9.4 build 2458 was released on 13th November 2012 with C-Bus Toolkit v1.12.4.

Changes:

- Fixed a bug introduced in v2.9.2 that caused FILE UPLOAD to fail.

The Partial installer (without a JRE) will no longer be released. For this and future releases only the Full installer will be distributed.

5.17 v2.9.2

C-Gate v2.9.2 build 2455 was released on 25th October 2012 with C-Bus Toolkit v1.12.2.

Changes:

- Installer now installs a private Java run-time for C-Gate instead of installing a system JRE.
- cgate.exe now supports the -restart parameter (new entry for this added to Start Menu).
- Sync now detects and copes with units that have worn-out EEPROMs.
- Fixed exception caused by an invalid C-Bus message produced by a DALI Gateway.
- Fixed a bug in FILE DOWNLOAD that causes a subsequent PROJECT SAVE to fail.
- Fixed a bug introduced in 2.9.0 where DBGET no longer returns a single property.
- Fixed performance issues with CNI v2.

C-Gate v2.9.2 build 2456 was released on 30th October 2012 with C-Bus Toolkit v1.12.3.

Changes:

- Fixed bug causing C-Gate service to shut down after 2 minutes.

5.18 v2.9.0

C-Gate v2.9.0 build 2449 was released on 16th June 2012 with C-Bus Toolkit v1.12.1.

Changes:

- Installer will now default to installing C-Gate as a Windows Service.
- Permanent background syncing.
- Sync now detects and reports duplicate units.
- When a network sync is cancelled by the user the network will now not go into error.
- Unravelling is completely overhauled.
- Client reference counting for projects. This prevents one client from closing a project that another client is using.
- Config options are now saved in the project file.
- DBGET now supports wildcards and responses include the array index.
- Support for Windows 7.
- Fixed a bug when upgrading project files that contain unicode characters.
- … and more.

5.19 Upgrade from 1.5 to 2.5

This section describes major changes from version 1.5 of C-Gate and how to make C-Gate 2.5 operate in a similar way to version 1.5 to allow existing client applications to still function.

5.19.1 Major changes from 1.5 to 2.x

The following are major changes from C-Gate 1.5 to 2.x:

Projects

C-Gate 2 introduced the concept of projects into C-Gate. Projects group several networks together to

operate as a group. C-Gate can run sereral projects simultaneously, each with a separate set of networks.

By default, C-Gate does not open networks immediately on opening. This can be configured, however.

Projects also introduced an aditional addressing element, the project name (eg. //HOME) that appears at the front of fully qualified C-Gate addresses. See Objects and Addresses for details of addressing changes.

The PROJECT command set is available for manipulating projects.

In addition, some project and network specific configuration variables can be stored along with a project file. These are detailed below.

See the Projects section of the reference for details.

Project Database

As of C-Gate 2.x, projects and project information are now held in a Project Database. The contents of the project database are managed by the DB commands. See command starting with DB in the manual.

Network file changes

In C-Gate 1.5, networks were defined in the networks.txt file, and these were the set of networks that operated with C-Gate. To change this, C-Gate had to be restarted after the file contents were changed. As of 2.x, networks are defined in the Project Database, or in a Network file named for the project.

The NET and PORT command sets were introduced to work with networks. Most network operations can now be performed without restarting C-Gate, allowing on-the-fly definition and use of networks and projects.

Configuration changes

Configuration variables are now most easily managed from the command line with the CONFIG command set. See the Configuration section of the guide for details.

Access Control Commands

Access control can now be managed with the ACCESS control commands.

5.19.2 Setting 1.5 compatibility

It is possible to set C-Gate 2.x into a mode where it is basically compatible with 1.5. To do that, you'll need to do the following:

Set the following configuration variables to the values given here. Do this in the C-GateConfig.txt file.

```
hide-project-names=yes
event.display-oids=no
```



```
ccp.display-oids=no
tag-name-output=no
use-1.0-addressing=no
network.source=file
```

Then proceed as for C-Gate 1.5. There are some additional things to be aware of:

1. The output of addresses from C-Gate has changed slightly so that a leading slash '/' is now added to all addresses output. This will required changes to address parsing.

2. On output of addresses, C-Gate now uses the form `/<network>/p/<unit>` instead of the 1.5 style: `p/<network>/<unit>`

3. Many more events are now issued, especially at higher global event levels.

4. Version 1.0 addressing is no longer fully supported.

6 Not Used

Enter topic text here.

6.1 Set Plant HVAC Level

[Not implemented yet]

Event

Set Plant HVAC Level

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_ward_on app ward zone-list mode raw-flag setback-enabled guard-enabled use-aux-level type level auxlevel

For description of parameters see the AIRCON SET_PLANT_HVAC_LEVEL command.

Sample Event output

```
20081022-084833 702 sys [aircon] set_plant_hvac_level ward=255 zone-
    list=1,2,3,4,5,6 mode=7 rawlevel=1 setbackenabled=1 guardenabled=1
    useauxlevel=1 type=255 level=65535 auxlevel=255
```

Sample SCP output

```
# aircon set_ward_off //BOARD/254/172 1 #sourceunit=0 OID=...
```

6.2 Set Plant Humidity Level

[Not implemented yet]

Event

Set Plant Humidity Level

Condition causing event

The equivalent command has been invoked on the application.

Event text

set_ward_on app ward

```
app = air-conditioning application address
ward = air-conditioning ward number (0-255)
```


Sample Event output

```
20081022-084833 702 sys [aircon] set_ward_off ward=1 sourceUnit=0
```

Sample SCP output

```
# aircon set_ward_off //BOARD/254/172 1 #sourceunit=0 OID=...
```

Index

- A -

Access 64
Alarm 62

- B -

Backus-Naur Form 41
Bridge 21

- C -

C-Bus Bridge 21
C-Bus Network 21, 22
C-Bus Unit 21
CNI 22
COM Port 22
Command 57, 59
Command Message 59
Connect 62
Connected 62
Critical 62

- D -

Debug 59

- E -

Energy 56

- F -

File 63

- I -

Information 56
Interface 58, 62

- L -

License 59, 63

- N -

Network 21, 22, 57, 60, 62
Network Error 60
Network Status 57

- P -

PC Interface 58
PCI 58
Power 56
Project 21

- R -

RS-232 22

- S -

Scene 59, 64
Scene Error 64
Security 60
Serial 22
Spec 63
Specification 63

- U -

Unit 59, 63
Unit Spec 63