C-Bus Application Messages & Behaviour

Chapter 2 – Lighting, Switching and Load Control

Document Number: CBUS-APP/02
Issue: 1.5
Date: 13 February 2007

Comments on this document should be addressed to:

Engineering Manager
Clipsal Integrated Systems
PO Box 103 Hindmarsh
South Australia 5007

TABLE OF CONTENTS

2 LIGHTING, SWITCHING AND LOAD CONTROL APPLICATION

2.1 Default Application ID

$38 (The range $30 to $5F can be used. $38 is the preferred default.)

2.2 Description

This application has been historically called "Lighting", but has in fact had a long history of use mainly for lighting, but also more generally for switching loads (including motors, fans, curtain & shutter control).

The Lighting, Switching and Load Control Application is used to control and monitor lighting or other electrical loads. This is usually comprised of incandescent or fluorescent lighting and associated control devices such as switches, dimmers and relays.

More generally, devices in this application are used to switch electrical loads on and off, or to regulate the amount of electrical energy delivered to an electrical load.

2.2.1 Overview

The C-Bus Lighting, Switching and Load Control Application is used to make a ***logical*** association between one or more electrical terminals (to which an electrical load is attached), and one or more control points.

The control point might be, for example:

- a switch being opened or closed by a person;
- a button being pressed by a person; or
- a computer requesting a change to the power delivered to the load at a scheduled date and time.

The logical link between the control point and the electrical load terminal is called a ***group***.

2.2.2 Terminology for Lighting, Switching and Load Control Devices

Lighting, Switching and Load Control devices are generally categorised as:

a. ***Input units***, which people interact with and ultimately have a controlling function; and

b. ***Output units***, which are typically connected to the lights and provide the power control required to switch a mains voltage on and off (a relay) or process the mains voltage in some way to control brightness (a dimmer).

2.2.3 Other Devices

Other devices on a network can also issue Lighting, Switching and Load Control Application commands. If they do so, they must adopt the behaviour of an Input Unit.

2.2.4 Example of Groups with Input and Output Units

In the example shown below, a series of switches (SW1 .. SW 4) are on the same C-Bus as a relay (R1) and a dimmer (D1).

The relay controls loads L1 and L2. The dimmer controls loads L3 and L4.

The association between the button on a switch, and load, is made using a group.

Example

To make:

a. a button on SW1 (don't care which button) control lamp L1;

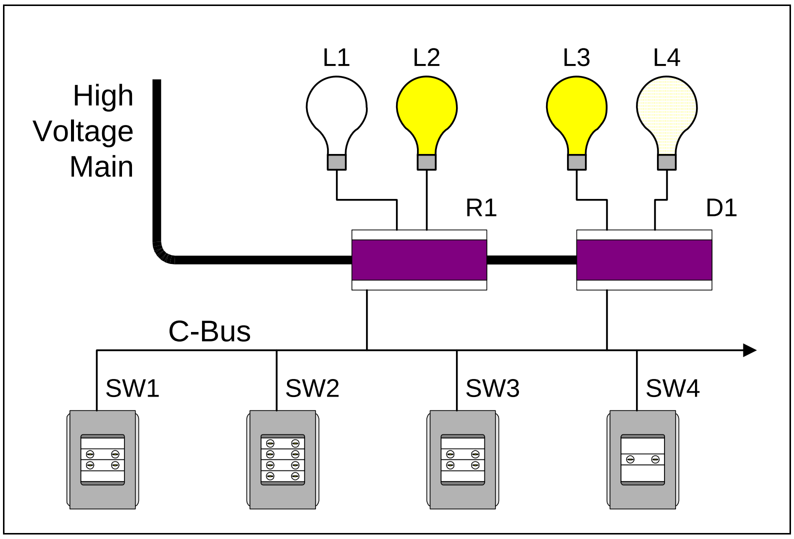
b. a button on SW3 (don't care which button) control lamp L4; and

c. a button on SW2 and another button on SW4 (don't care which buttons) control lamps L2 and L3.

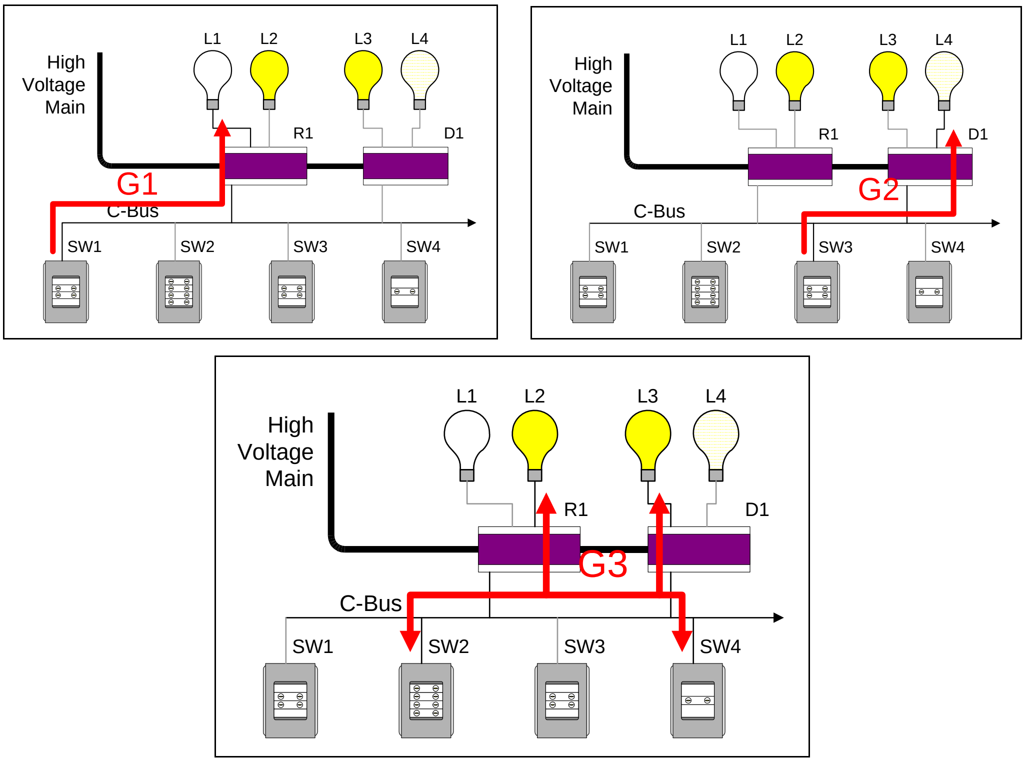
NOTICE that two buttons on two inputs units (SW2 and SW4) are going to control 2 loads, these loads are on different output units, and furthermore, one of those output units is a dimmer and another is a relay.

The process to follow is:

- Allocate 3 new unique group numbers (all that matters is that they have not been used before). Call these G1, G2 and G3.
- Program the relay R1 so that the terminals for load L1 are controlled by group G1, and so that the terminals for load L2 are controlled by group G3.
- Program the dimmer D1 so that the terminals for load L4 are controlled by group G2, and so that the terminals for load L3 are controlled by group G3.
- Program SW1 so that the button of interest controls group G1.
- Program SW3 so that the button of interest controls group G2.
- Program SW2 and SW4 so that the buttons of interest control group G3.



The programming has created a set of logical linkages between the buttons on the input units, and the terminals on the output units, thus:



The most sophisticated case with group G3 shows that for multiple input units, the bus messages travel between input units ***as well as*** from input units to output units.

This ensures that multiple input units controlling a single group will always maintain a synchronised state.

In the case where a group is used to control different load types (in this case, relay and dimmer), the interpretation of the meaning of a level in a group is performed in the output unit.

Clipsal C-Bus Relay units typically interpret any non-0 level as meaning ON, although this can be modified by an installer.

2.3 Document Convention

Numbers are shown in decimal (base ten) with no other special prefixes or indications.

Binary numbers (base 2) are shown with the prefix %.

Hexadecimal numbers (base 16) are shown with the prefix $.

Example: 157 = %10011101 = $9D

2.4 Data Conventions

2.4.1 Group Address

The C-Bus Lighting, Switching and Load Control Application treats each Group as a shared Network Variable, each of which can store a number (the value of the network variable)

The following convention is used for Group Addresses:

Group Address:

Size: 8-bit byte

Range: $00 .. $FF

Special Cases: $FF is reserved and indicates that the group address is unused. This value can be set as a valid Area Address, but this practice is discouraged.

2.4.2 Level

The value stored by each of the Group Address network variables is a single byte and is interpreted as meaning a level.

The level is a neutral format used to represent the amount of electrical energy to deliver to the load. Levels transported over the network can be scaled, or modified by a profile in output units.

The neutral format for levels is:

- Value 0 means that the Group is OFF;
- Value 255 means that the Group is fully ON; and
- Value 1 .. 254 mean the Group is set to the fraction (GAV / 255).

2.4.3 Languages

A language is specified when a label is applied, and when the preferred language is set.

The following languages are defined for C-Bus Label commands:

Language	Code (hex)
English	0x01
English (Australia)	0x02
English (Belize)	0x03
English (Canada)	0x04
English (Carribean)	0x05
English (Ireland)	0x06
English (Jamaica)	0x07
English (New Zealand)	0x08
English (Philippines)	0x09
English (South Africa)	0x0A
English (Trinidad)	0x0B
English (UK)	0x0C

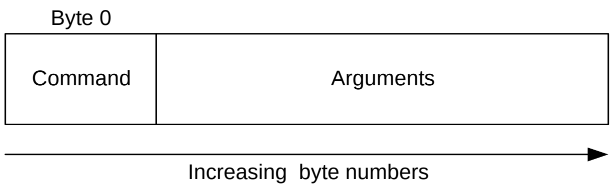
Language	Code (hex)
English (USA)	0x0D
English (Zimbabwe)	0x0E
Afrikaans	0x40
Basque	0x41
Catalan	0x42
Danish	0x43
Dutch (Belgium)	0x44
Dutch (Netherlands)	0x45
Faeroese	0x46
Finnish	0x47
French (Belgium)	0x48
French (Canada)	0x49

Language	Code (hex)
French	0x4A
French (Luxembourg)	0x4B
French (Monaco)	0x4C
French (Switzerland)	0x4D
Galician	0x4E
German (Austria)	0x4F
German	0x50
German (Liechtenstein)	0x51
German (Luxembourg)	0x52
German (Switzerland)	0x53
Icelandic	0x54
Indonesian	0x55
Italian	0x56
Italian (Switzerland)	0x57
Malay (Brunei)	0x58
Malay	0x59
Norwegian	0x5A
Norwegian (Nynorsk)	0x5B
Portuguese (Brazil)	0x5C
Portuguese	0x5D
Spanish (Argentine)	0x5E
Spanish (Bolivia)	0x5F
Spanish (Chile)	0x60

Language	Code (hex)
Spanish (Colombia)	0x61
Spanish (Costa Rica)	0x62
Spanish (Dominican Republic)	0x63
Spanish (Ecuador)	0x64
Spanish (El Salvador)	0x65
Spanish (Guatemala)	0x66
Spanish (Honduras)	0x67
Spanish	0x68
Spanish (Mexico)	0x69
Spanish (Nicaragua)	0x6A
Spanish (Panama)	0x6B
Spanish (Paraguay)	0x6C
Spanish (Peru)	0x6D
Spanish (Perto Rico)	0x6E
Spanish (Traditional)	0x6F
Spanish (Uruguay)	0x70
Spanish (Venezuela)	0x71
Swahili	0x72
Swedish	0x73
Swedish (Finland)	0x74
Chinese_cp936	0xCA

2.5 Message Structure

Lighting, Switching and Load Control Application messages are between 2 and 21[1] bytes long, and have the form:



The format of the arguments portion is variable, and is dependent on the command.

The command byte is broken into bit-fields to support encoding of a command and the number of bytes following as parameters. There are two possible codings, to support a

[1] A 21 byte message only applies to the longest possible Label command.

large number of commands with short arguments, and a small number of commands with long arguments.

The short argument command form is:

7 ←bit→ 0

0	C	C	C	C	L	L	L

The long argument command form is:

7 ←bit→ 0

1	C	C	L	L	L	L	L

Where “C” represents a bit of a command, and “L” represents a bit of the length.

2.5.1 Short Form Commands

The following short form commands are supported:

RAMP to LEVEL:	%0 xxxx 010
OFF:	%0 0000 001
ON:	%0 1111 001
TERMINATE RAMP:	%0 0001 001

The (3 bit) length field reflects the number of arguments.

All other possible short form commands are reserved, and shall not be used.

2.5.2 Long Form Commands

The following long form commands are supported:

LABEL:	%1 01 xxxxx

The (5 bit) length field reflects the number of arguments.

All other possible long form commands are reserved, and shall not be used.

2.6 Defined Commands

Lighting, Switching and Load Control Application commands are typically emitted by an input unit.

To ensure synchronisation of network variables (the values of Group Addresses), messages shall be received and processed by all Units operating in the Lighting, Switching and Load Control Application (including Lighting input and output units) that have a C-Bus Interface level of 2 or more.

Lighting, Switching and Load Control Application messages are C-Bus Specific Application Language (SAL) messages.

All messages listed are mandatory for C-Bus Lighting, Switching and Load Control Application devices, unless explicitly stated otherwise. Deviation from these messages will cause C-Bus devices to be incompatible. Consult Clipsal Integrated Systems before deviating from these messages.

2.6.1 Off

Command:	$01
Arguments:	<Group Address>
Meaning:	Set the Group Address Variable to 0 (ie OFF)
Originator:	Anywhere (usually a Lighting, Switching and Load Control Application input unit)
Notes:	Refer to section 2.4.1 for the group address conventions.

2.6.2 On

Command:	$79
Arguments:	<Group Address>
Meaning:	Set the Group Address Variable to 255 (ie ON)
Originator:	Anywhere (usually a Lighting, Switching and Load Control Application input unit)
Notes:	Refer to section 2.4.1 for the group address conventions.

2.6.3 Ramp to Level

Command: %0RRRR010

Arguments: <Group Address>, <Level>

Meaning: Ramp the Group Address Variable to the specified level

Originator: Anywhere (usually a Lighting, Switching and Load Control Application input unit)

Notes: Refer to section 2.4.1 for the group address conventions.

<Level> is a single byte output level, with $00 being fully off and $FF being fully on.

The ramp rate is specified by the 4 bits (RRRR) of the command byte, as follows:

%0000 = instantaneous
%0001 = 4 s ramp rate from min to max or max to min
%0010 = 8 s ramp rate from min to max or max to min
%0011 = 12 s ramp rate from min to max or max to min
%0100 = 20 s ramp rate from min to max or max to min
%0101 = 30 s ramp rate from min to max or max to min
%0110 = 40 s ramp rate from min to max or max to min
%0111 = 1 min sec ramp rate from min to max or max to min
%1000 = 1.5 min ramp rate from min to max or max to min
%1001 = 2 min ramp rate from min to max or max to min
%1010 = 3 min ramp rate from min to max or max to min
%1011 = 5 min ramp rate from min to max or max to min
%1100 = 7 min ramp rate from min to max or max to min
%1101 = 10 min ramp rate from min to max or max to min
%1110 = 15 min ramp rate from min to max or max to min
%1111 = 17 min ramp rate from min to max or max to min

2.6.4 Terminate Ramp

Command: $09

Arguments: <Group Address>

Meaning: Terminate the ramp currently active on the given Group Address Variable. Input and Output units are left at the (internal) Group Address Variable Level they had at the instant they received the command.

Originator: Anywhere (usually a Lighting, Switching and Load Control Application input unit)

Notes: Refer to section 2.4.1 for the group address conventions.

2.6.5 Label

Command: %101LLLLL

Arguments: <Group Address> <Options> <Language> <Bytes>

Meaning: Place a Label against a Group, for all devices on the network that support labelling.

Originator: Anywhere (usually a PCI or CNI)

Notes: Labels are only applied to units with label display capabilities. All other units shall receive and discard this message.

Refer to section 2.4.1 for the group address conventions.

The “L” bits must be set appropriate to the count of the number of bytes that follow the command. The shortest possible command has 3 bytes following.

Loading a label with no text bytes will erase any previously loaded label for the selected language and Group.

When setting the Preferred Language, there shall not be any text bytes present.

Multiple label commands are not allowed in the same message.

The label command may be concatenated after other lighting commands, however the label command shall be the last command in the message.

<Options> is a set of bit-field codes, with the following meanings for the Lighting, Switching and Load Control Application:

Bit 0: For Lighting, this bit must be 0

Bits 2,1: 00 = Text label
01 = Predefined icon
10 = Load Dynamic icon
11 = Set Preferred Language

Bit 3: For Lighting, this bit must be 0

Bit 4: Reserved, must be 0

Bit 6, 5: 00 = Flavour 0
01 = Flavour 1
10 = Flavour 2
11 = Flavour 3

For a label to be displayed, the Flavour bits must match the flavour stored in units that receive label commands. This allows up to 4 different labels against the same Group.

Bit 7: Reserved, must be 0. Only set to 1 during certain parts of a Dynamic Icon Load (refer section 2.9.6.4).

<Language> is a Language Code as defined in section 2.4.2.

<Bytes> is either the label text, an index to a predefined icon, or a stream used to define a dynamic icon, as follows:

For a text label, a stream of up to 16 ASCII coded text bytes;

For Predefined icons, exactly 3 bytes must be present:

$01, Icon_H, Icon_L

Where Icon_H and Icon_L are the high and low order bytes (respectively) of the predefined icon number to display.

For Dynamic icons, up to 16 bytes can be present. The method of loading Dynanmic icons is described in section 2.9.6.4.

2.7 Message Priority

C-Bus Lighting, Switching and Load Control Application messages that are initiated by human events (for example, key press) are transmitted at Class 3 (medium low) priority.

Lighting, Switching and Load Control Application messages generated by complex software controlled systems (for example, a scheduling system or a security system) are transmitted at Class 4 (low) priority.

Message priority is part of the C-Bus message header (refer to the C-Bus PC Interface documentation), and is set by the two most significant bits of the C-Bus header field, as follows:

00 = Class 4, lowest priority
01 = Class 3, Medium low priority
10 = Class 2, Medium high priority
11 = Class 1, High priority

Thus, to send a Class 3 message, use a message header of (for example) $45 instead of $05 for a Class 4 message.

2.8 Internetwork Routing

Clipsal key input units never generate messages routed through bridges. (To extend range or node capacity, two adjacent networks can be linked by a bridge in “Application Connect” mode, which makes the bridge appear transparent).

Other devices participating in the Lighting, Switching and Load Control Application can generate messages routed through one or more C-Bus bridges. Clipsal C-Bus input and output units will correctly receive and interpret such messages. The Network routing information will be discarded.

2.9 Application Behaviour

2.9.1 Concatenated Commands

A Lighting, Switching and Load Control Application device may receive a message containing more bytes than a single command. This permits a single C-Bus transmission to contain multiple commands for a single application.

Devices using C-Bus Lighting, Switching and Load Control Application messages must process all received bytes. This is achieved by placing the received bytes in a buffer, and using the following simple algorithm:

```
WHILE the buffer contains bytes LOOP

    The first byte defines the command type and argument
    count (refer section 2.4).

    Process the first (command) byte and its arguments

    Once processed, remove the command and argument bytes
    from the buffer

END LOOP
```

Note that Label commands must always be the last command to be processed as they do not support further concatenation.

2.9.2 Reception of Unknown or Unsupported Commands

Devices receiving a message containing a command header that is not supported shall:

1. Ignore the command, and use the length bits to determine the number of subsequent bytes to skip; and
2. Resume checking for commands after skipping the unknown or unsupported command.

2.9.3 Maintenance of Group Address Variable State

In the Lighting, Switching and Load Control Application , the Group Address is the means of communicating a setting between devices.

All devices programmed to be a member of a given Group Address share a single network variable, known as the Group Address Variable.

A C-Bus network is inherently high reliability. However, in the highly unlikely event of a communication failure of some kind, two combined systems are used to ensure integrity, and to correct discrepancies. These systems are:

a. State monitoring and correction; and
b. Monitor scheduling.

These systems are used by C-Bus Interface Level 4 devices, or higher. The integrity systems are robust enough that devices with C-Bus Interface Level 1 to Level 3 will not disrupt reliable operation of the network.

Lighting, Switching and Load Control Application devices that are capable of transmitting a command are collectively called input units. These include standard

Clipsal Key Input Units, Touchscreens, the Serial Interface and so on (generally everything except output units and system units such as a power supply).

Input units shall be capable of receiving and interpreting all Lighting, Switching and Load Control Application messages appearing on the bus.

2.9.3.1 State Monitoring and Correction

The master holder of the state of a Lighting, Switching and Load Control Application Group Address Variable depends on the nature of the event initiating the last change. The nature of events for Input Units are either:

a. Initiated by human interaction, such as a light switch being pressed; or

b. Initiated without human interaction (but including via remote control), such as a scheduled event, or an operation performed remotely via a telephone link.

In the event a discrepancy is found:

a. For human initiated events, for example a light switch, it is preferable for the input units to change the state of their indicator than to have the room lighting change.

b. For non-human initiated events, for example, a C-Bus scheduler, the converse applies: it is preferable for the input unit to try and impose its state to ensure the output unit terminals, and hence the lighting accurately reflect what was supposed to happen.

The following rules apply to determination of Active and Passive Modes for Lighting, Switching and Load Control Application input units:

a. A unit that transmits (generally due to some event occurring, such as somebody pressing a key) shall change its mode, for that Group Address Variable, to ACTIVE.

b. A unit that receives a message for one of its Group Address Variables shall change its mode, for that Group Address Variable, to PASSIVE.

The following rules shall be used to monitor the state of Group Address Variables and, if required, resolve conflicts:

a. Input units in both ACTIVE and PASSIVE mode shall monitor all MMI responses on the Lighting, Switching and Load Control Application , looking for discrepancies[2] or errors[3] in the Group Address Variables of interest to it;

---

[2] A discrepancy is a difference between the reported state from an output unit and the internal state record of the input unit.

[3] An Error is when two or more output units are transmitting conflicting states (for example, one reporting ON and another reporting OFF will be seen as ERROR)

b. If an ACTIVE mode input unit detects 2 reported Group Address Variable Error indications[4] in a row, it shall issue a Ramp to Level command to force that group into the state consistent with its internal records.

c. If an ACTIVE mode input unit detects 2 identical discrepancies in a row, its behaviour is determined by the type of event which caused it to enter ACTIVE mode:

   1. If it was a human initiated event, it shall update its internal state record to that reported in the MMI; or

   2. If it was a non-human initiated event, it shall retransmit the Lighting, Switching and Load Control Application command to cause the output to be changed. It shall retry the setting of a Group Address Variable after each 2 successive discrepancies. If after 3 tries, discrepancies are still reported, the input unit shall cease trying and shall set its internal state to match that reported.

d. If a PASSIVE mode input unit detects 2 identical discrepancies in a row, it shall update its internal state record to that reported in the MMI

e. If a PASSIVE mode input unit detects 5 reported Group Address Variable Error indications in a row, it shall issue a Ramp to Level command to force that group into the state consistent with its internal records, and thereby become the ACTIVE unit.

2.9.3.2 MMI (Monitor) Scheduling

The rules used by ACTIVE and PASSIVE mode input units to resolve discrepancies rely on the regular generation of MMI requests. (The MMI is a fast, efficient status reporting technique used in C-Bus).

One input unit in the Lighting, Switching and Load Control Application is designated an MMI MASTER. It is responsible for generating an MMI request, nominally every 3 seconds[5].

The MMI MASTER is just another input unit – it could therefore be the current ACTIVE Mode input unit, but need not be.

A set of rules are used to:

a. Allocate an MMI MASTER when a network is powered up;

b. Pass the role of MMI MASTER between Nodes; and

c. Handle failure or removal of the current MMI MASTER.

To implement the rules, each unit needs three timers:

a. The DELAY_TIMER with increment of 16 milliseconds, used to set a fine granularity between transmissions;

---

[4] Refer to the Serial Interface Specification for details of the reporting format for an MMI. In summary, each group can present a status of NOT_PRESENT, ON, OFF or ERROR.

[5] The default interval is 3 seconds. The interval can normally be configured to other values during network commissioning. Intervals of less than 3 seconds are not permitted.

b. The RETRANSMIT_TIMER, with increment of 1 second, used to set a coarse granularity between transmissions; and

c. The TRANSMIT_TIMER, with increment of 0.25 seconds, used to time out a transmission.

Each unit also needs to store a programmable value, known as the MMI_REPEAT_TIME, which is the nominal interval between MMI requests if that unit is the MMI MASTER, in seconds.

Allocation of MMI MASTER on Network Power Up

The following rules shall be used by any input unit when powered up:

a. The unit assumes it is an MMI MASTER.

b. Set the DELAY_TIMER equal to the unit address of each unit (so the delay period will be 16 ms * unit address)

c. Wait for either a received MMI or expiry of the DELAY_TIMER.

d. If the DELAY_TIMER expires, and no MMI request has been received in the meantime, the unit attempts to transmit an MMI request, and continues the assumption that it is the MMI MASTER.

e. If the DELAY_TIMER has not expired, and an MMI request is received, the unit assumes it is no longer the MMI MASTER.

f. Contention (multiple units attempting to transmit their MMI request simultaneously) is described below.

MMI MASTER during normal operation

The following rules are used after power up and establishment of an MMI MASTER. These rules are used to pass the role of MMI MASTER between units, and to ensure that a network always contains an MMI MASTER, even if the current MMI MASTER fails or is disconnected from the network.

For a unit that is the MMI MASTER:

a. An MMI is scheduled to occur at intervals of MMI_REPEAT_TIME. This is accomplished by loading the RETRANSMIT_TIMER with MMI_REPEAT_TIME and waiting for expiry. When the timer expires, an MMI is transmitted, the timer is reloaded, and the cycle repeats.

b. If the unit receives an MMI that it did not initiate, it shall immediately drop the role of MMI MASTER. The rules for units that are not MMI MASTER are immediately followed.

For a unit that is not the MMI MASTER:

a. If the unit receives an MMI, it loads its RETRANSMIT_TIMER with 1+ MMI_REPEAT_TIME.

b. If RETRANSMIT_TIMER expires, then the rules for power up are followed. (The unit attempts to become the MMI MASTER, after waiting an additional period based on its unit address in the DELAY_TIMER.)

Bus Contention and MMI Transmission Rules

It is possible, though unlikely, for multiple units to attempt simultaneous transmission of an MMI. The C-Bus protocol rules for collision detection and resolution will ensure that only one of the units will succeed. In that case, other units could have a pending MMI for transmission. To ensure that there is not a storm of MMIs due to pending requests, the following rules shall be used:

a. When an MMI is to be transmitted, the TRANSMIT_TIMER is loaded with 0.25 seconds and started.

b. If the TRANSMIT_TIMER expires, the transmission request is aborted.

c. If the transmission request was aborted, a 2 second period must elapse before another MMI transmission attempt is made. If an MMI is received during this 2 second period, the normal operation rules above are followed.

d. MMI retransmission is only attempted if an MMI has not been received from another unit during the 2 second period.

2.9.4 Lighting Output Units

Lighting, Switching and Load Control Application Output units are typically relays or dimmers. Each unit contains one or more channels. A channel is an individually switchable or controllable output, with terminals used for connection to wiring and loads.

2.9.4.1 Number of Group Addresses Variables

Output units shall support at least one Group Address Variable per channel.

2.9.4.2 Assignment of Group Address Variables

Within an output unit, each channel can be assigned a Group Address. If desired, multiple channels in a unit (or even spanning multiple output units) can be allocated the same Group Address. This would cause multiple loads to respond simultaneously to a single command.

It is not permitted for a single channel to have multiple associated Group Addresses (because this allows conflicts to develop very easily).

It is possible, however, to assigned an Area Address to a unit. Area Addresses are a special case, and are described in section 2.9.5.

Some Clipsal Lighting output units include special group addresses that are allocated to logic functions instead of channels. These permit complex conditions to be set up. However, the use of logic functions in output units is discouraged.

2.9.4.3 Response to MMI requests

A Lighting, Switching and Load Control Application output unit responds to an MMI request by inserting into the MMI response the state of any Group Addresses assigned to its channels.

It does not insert a status for any allocated Area Addresses.

2.9.5 Area Addressing

An Area Address can be assigned to a unit. If assigned, it applies to the entire unit (all groups). The same Area Address can be used across multiple units.

Area Addresses behave the same as Group Addresses in all respects (in other words, they are another Network Variable), with the exception that Area Addresses are never reported by output units in an MMI response.

Area addresses are deprecated and should not be implemented in new products.

2.9.6 Labels

2.9.6.1 Device Behaviour for Label Commands

Devices that cannot be labelled shall accept and discard label commands.

Devices that can be labelled shall always accept label commands. If the supplied group matches a group in the unit, then the label is applied to that group. When labels are applied to a group, all buttons with that associated group are considered for labelling. The label is applied to buttons where the button flavour matches the flavour supplied in the label command. This allows up to 4 different labels to be applied to buttons on the same group.

2.9.6.2 Deleting a Label

A label is deleted by applying label text with length 0. This causes the unit to revert the label for any matching groups to a unit-specific default label.

2.9.6.3 Languages

Units capable of displaying labels always show the label, for each group, in the current preferred language.

If a group does not have a label in the current preferred language, it displays the English (Code 1) label for that group instead.

2.9.6.4 Loading Dynamic Icons as Labels

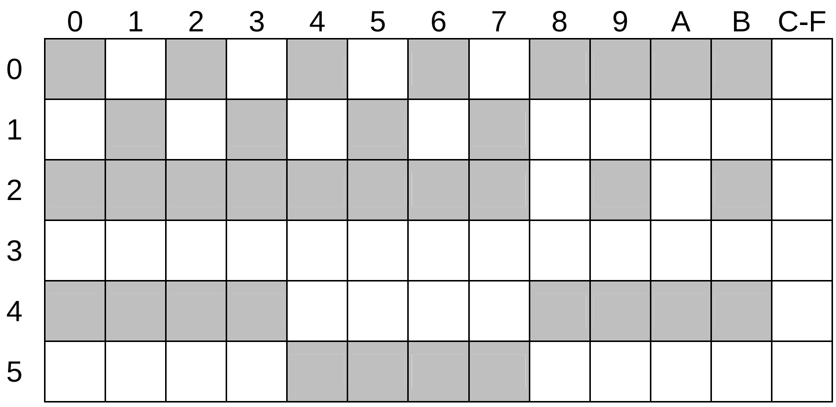
All Dynamic Icons are loaded as uncompressed monochrome bitmaps.

2.9.6.4.1 Icon Structure

Bitmap data is represented as a byte stream, with black pixels represented by "1" bits and white pixels by "0" bits.

Bytes are filled by scanning across bitmap rows from top left to bottom right and filling bytes from Most Sig. bit down. The last byte of each line is padded out to a full byte with 0 bits.

Example, for a 12x6 pixel bitmap:



The Data Stream generated from this bitmap would be:

AAF0
5500
FF50
0000
F0F0
0F00

2.9.6.4.2 Dynamic Icon Size

DLT Dynamic Icon size is unrestricted by firmware. For use by toolkit and interaction with other display elements, the size has been defined as:

Width	62 pixels
Height	16 pixels
Vertical Offset	10 pixels

2.9.6.4.3 Attributes

Each Dynamic Icon has the following attributes:

Icon Number:	16 bits – a reference number used to display the icon
Language:	8 bits
Icon Width:	8 bits – the width of the icon in pixels

Icon Height: 8 bits – the height of the icon in pixels
Vertical Offset: 8 bits – an offset to apply when displaying the icon

After being loaded, the icon is displayed against a button by activating a label with matching Icon Number and language.

If the icon is created with a size smaller than the allocated height of 16 pixels, the vertical offset is used to vertically centre the icon in the 16 pixel high region available to it.

2.9.6.4.4 Command Sequence

When loading Dynamic Icons, a sequence must be followed, in order, to complete the definition.

Start Dynamic Icon

The first command tells a label device (eg DLT) to start a new dynamic icon. This command finds the most appropriate of the 16 available slots in which to put the icon.

The List of icons is searched for each test, and if a match is found, that slot is used:

Priority	Test
1	Language & Icon Number Match
2	Language = 0xFF ie. Icon Unused
3	Icon language not currently in the List of Languages ie. Language not used
4	Language of Icon not current language
5	Fail. No available icon slots

The command to Start Dynamic Icon is:

`\053800A4gg`**`080020`**

Icon Header

Once a Dynamic Icon has been started, the first data loaded must be the icon's properties. These are contained in a header, where the bytes (offset from the start of the label command body) are:

Offset	Data
+0	Icon Language
+1	Icon Number High
+2	Icon Number Low
+3	Icon Width
+4	Icon Height
+5	Icon Vertical Offset
+n	Icon Body (Optional)

The command to load the icon header for a dynamic icon is:

```
\053800Axgg04 <Language> <Icon_Number_Hi> <Icon_Number_Lo>
<Icon_Width> <Icon_Height> <Icon_Vertical_Offset>
<optional_bitmap_data>
```

where x is the number of bytes following.

Appending bitmap data bytes is optional (just the header can be included to keep things simple), but it will reduce the number of commands necessary to transmit the icon. There is no limit to the number of data bytes other than the restrictions imposed by a CBus command, hence the amount of data that can be added could vary depending on the number of C-Bus networks to be crossed.

Icon Bitmap

After the icon header, the icon bitmap data is loaded, as a stream of bytes.

The command to Append Dynamic Icon Data is:

```
\053800A4gg080021
```

The bitmap load uses the process:

```
send \053800A4gg080021
While there is more bitmap data to be transmitted
    {
        send \053800Axgg04<data>
    }
```

Again, the number of data bytes that can be included is limited by the length of a CBus command and will depend on the number of networks crossed.

Complete Dynamic Icon

Once all dynamic icon data has been transmitted, Use Dynamic Icon must be sent. This command registers the icon in the List of Icons and it can then be used as a label.

This command is:

```
\053800A4gg080022
```

Display Icon

Once the Dynamic Icon has been loaded, it is displayed against a button bye issuing a label command with the appropriate language and pre-defined icon number.

Example

This exampe uses the icon data shown at the data of this section, loads and displays it in a Clipsal DLT switch with Group 1 on one of its blocks as follows.

The icon data from above was:

AAF0 5500 FF50 0000 F0F0 0F00

The icon will be loaded as Chinese (Lang = $CA), with Icon number $0013.

Command to send	Comments / Description
\053800A401080020	Start Dynamic Icon
\053800A80104CA00130C0600	Icon Header: [CA = Chinese] [Icon = $0013] [Width = $0C pixels] [Height = $06 pixels] [Vertical Offset = 0 pixels]
\053800A401080021	Append Dynamic Icon
\053800A80104AAF05500FF50	Write icon bitmap data
\053800A801040000F0F00F00	Write icon bitmap data
\053800A401080022	Use Dynamic Icon
\053800A60102CA010013	Display Icon on button assigned to Group 1
Notes: *1. This icon's data could have been loaded using 1 long write command* *2. The vertical offset byte could be set to 5 to cause this icon to be vertically centred in the 16 pixel high region in which it is displayed.*	

2.10 Limitations

The number of Group Addresses support by output units varies with the number of channels. Some later model Clipsal output units support up to 16 group addresses even though the number of channels might be less than 16. In that case, there could be a block of Group Address Variables reported and stored in a unit, but which are not used.

2.11 Examples

These examples assume that a device interfaces to C-Bus using the C-Bus Serial Interface described in more detail in CBUS-SIUG, and assume the Serial Interface SRCHK option is set so that data transfers in both directions use a checksum.

To switch on all electrical loads associated with Group $93, a device could issue:

To PCI: \0538007993B7

To perform the same operation on a remote network (through a single bridge with unit address $56 on the side of the sending device, and unit address $37 on the side of the trigger control device), a device could issue:

To PCI: \03**5609**3879935A

[6] The Installation MMI is issued on Application number $FF. Refer to Chapter 1.

The internetwork routing bytes ($5609) are be modified by the bridge as the message passes through, to construct the reverse route. The receiving device(s) can ignore this, as the reverse route is not important.

2.12 Notes

None.