CBUS DCC Command Station FLiM programming

The CBUS DCC command station version 4 firmware is designed so that it can be built for different hardware, based on PIC18 series CAN processors. All versions share a common code base and FLiM interface.

The term MERGCMD has been used to describe the common source code that can be built for different hardware. The most commonly used hardware at present is the MERG CANCMD.

At present, MERGCMD can be built for:

- CANCMD - MERG CBUS DCC command station
- CANBC – Original MERG BC1a command station with CBUS daughter board

Definitions have also been created for:

- CANGC3 – Rocrail implementation of cancmd
- CANKCMD – Cancmd ported to 18F25k80 processor
- AMCTRLR – Animation controller incorporating CBUS DCC command station

although these options have not yet been implemented.

The processor condition compilation is separate from the target hardware definition, so any hardware variant can be built for any supported processor.

Not all of these combinations are useful. For example, building cancmd for 18F4580 or canbc for 18F2580 is not useful because the processor built for cannot be fitted in the target hardware.

However, it does mean that, for example, any target hardware can be built for the 18F or 18Fxxk series processors, if the developer so chooses.

Node Variables

MERGCMD version 4 has a total of 144 Node Variables.

These consist of 14 node variables for general options, 2 spare variables that can be added without affecting the overall structure, and a shuttle table with 32 entries, each of 4 bytes.

NV1 – Command Station Number

Default value: 0

This is always set to zero at present, but allows for expansion to support multiple command stations on the CBUS in the future.

NV2 – User Flags

Default value: Binary: 00100110 Hex 0x26 (Steal and share enabled, stop on timeout)

- Bit 0 - Silent — Set to enable silent mode, the beeper will never sound so you have to look at the LEDs to see if you have a short circuit or other error. This is for those whose railway is within hearing distance of the rest of the family late at night when they are asleep!
- Bit 1 - Permit Steal — Set to enable the steal option
- Bit 2 - Permit Share — Set to enable the share option
- Bit 3 – Permit event reset — Set true to allow a taught event to reset the CS
- Bit 4 - Map event — When set, CBUS event numbers are mapped directly to DCC accessory addresses. This avoids the need to teach every event, but every event on the given node number will cause a DCC accessory command. See also NV11 and NV12.
- Bit 5 - Stop on timeout — When set, if a loco session times out, the train is brought to a stop before the session is released. When not set, the train is dispatched whilst moving.
- Bits 6 – Start of day — Issue an event 1 on startup which can be used as a start of day event *
- Bit 7 – Enable Shuttles — Enable the DCC shuttles feature. Version 4 has a hard coded “proof of concept” shuttle which is enabled by this flag.

* The event is fixed as of version 4, but in version 5 it will be possible for it to be changed by teaching a producer event

The “speed must be zero to reverse” flag has been removed from here because this feature is better implemented in the cab.

NV3 – Operations flags

Default Value: Binary 00000001 Hex 0x01 (Output controlled by J7)

These flags affect the operation of the DCC signal generation.

Any software that allows the user to change these flags should issue a warning that changing these values may affect operation of the command station and, in certain cases, could cause hardware damage so users should only change these values if they know what they are doing.

- Bit 0 - Jumper control For command stations that have a jumper to set which outputs are used for the main and programming tracks, such as cancmd J7, set this bit to enable jumper control of the outputs. When this bit is 0, the output selection is controlled by the flag bits in this NV. This bit has no effect on command stations that have fixed output selection, such as canbc.
- Bit 1 - Main output on board When jumper control is disabled, this flag controls the output selection. When set, the main layout is the on board DCC output, when clear, the main layout is the booster output.
- Bit 2 - Analogue detection On some hardware, such as canbc, analogue measurement of the main output current is an option. Set this bit to use analogue main current detection, When clear, digital overload detection will be used. This option has no effect on hardware that has a fixed overload detection mechanism, such as cancmd.
- Bit 3 - ZTC mode Set this flag to use ZTC timing on the programming track.
- Bit 4 – Track off with stop all When set, the track power will be turned off after a stop all command is executed.
- Bits 5-7 Spare for future enhancements

NV4 – Debug flags

Default value: 0 (debug packets disabled)

These flags allow DCC packets to be echoed as ACDAT CBUS packets so that they are visible on a CBUS packet monitor, such as FCU or JMRI. This should only be used for debug or testing, as it can generate a lot of CBUS packets.

- Bit 0 - Priority Packets When there is a change of speed or function setting, a DCC packet is sent immediately, rather than waiting for the next refresh cycle. Set this option to echo these priority packets to CBUS.
- Bit 1 - Refresh Speed Packets The speed of each active session is regularly refreshed. Set this option to echo these speed refresh packets to CBUS.
- Bit 2 - Refresh Function packets The function settings of each active session are regularly refreshed. Set this option to echo these function refresh packets to CBUS.
- Bit 3 - Service Mode Packets Set this flag to echo all programming track service mode packets to CBUS.
- Bit 4 - Accessory packets Set this flag to echo all accessory control packets to CBUS
- Bits 5-7 Spare for future enhancements

Note that the ACDAT packet always has 7 bytes of data. Only the first n bytes are valid, where n is the length of the DCC packet being echoed.

NV5 – Walkabout Timeout

Default value: 60 (seconds)

The session timeout in seconds. Values can be set from 1 to 255 seconds.

Set a value of zero to disable timeouts completely. This is not recommended because there is no way to recover sessions if the cab (or PC based throttle software) is removed or crashes.

Do not set the timeout value less than the "keep alive" interval of the cabs you are using, otherwise sessions will continually time out.

NV6 – Main current limit

Default values: 96 (cancmd), 125 (canbc)

This is the raw A->D conversion value for the current limit on the main track. This allows users to reduce the current limit below the maximum for the hardware if they wish.

Software which allows this value to be changed should warn the user that setting a value too low may cause false alarm overload detections, setting it too high may cause erratic behaviour or even hardware damage, so should only be altered by those who know what they are doing.

The default will depend on the rating of the output stage of the hardware. It is not applicable when the main track is using a separate booster.

NV7 – Service current limit

Default values: 96 (cancmd and canbc)

This is the raw A->D conversion value for the current limit on the programming track. This allows users to reduce the current limit below the maximum for the hardware if they wish.

Software which allows this value to be changed should warn the user that setting a value too low may cause false alarm overload detections, setting it too high may cause erratic behaviour or even hardware damage, so should only be altered by those who know what they are doing.

The default will depend on the rating of the output stage of the hardware.

NV8 – Current Multiplier

Default values: 10 (cancmd), 30 (canbc)

This is the value that the raw A->D conversion value should be multiplied to give milliAmps.

This factor is used when the command station reports current consumption using CBUS event+data packets.

Users can tune this value using actual current measurements to get the most accurate reporting

NV9 – Increase for ACK pulses.

Default values: 3 (cancmd), 5 (canbc)

This is the increase in current (raw A->D value) that must be detected to receive an ACK pulse during service track programming.

Software which allows this value to be changed should warn the user that setting a value too low may cause erratic CV reading and programming, setting a value too high may stop CV reading and programming from working at all on the programming track., so this should only be altered by those who know what they are doing.

NV10 – Shoot through delay

Default value: 8 (canbc) – not applicable for cancmd

For command stations that use discrete FETs in their driver stage, such as canbc, this is a delay count between positive and negative pulses on the DCC track to avoid shoot through. The effect of the delay count will depend on the clock rate of the hardware.

On a PIC18 running at 32MHz (8MHz resonator), shoot through delay will be 500nS times the value in this NV. The default value is 8, giving a shoot through delay of 4uS.

On hardware that does not use discrete FETs, this value will have no effect.

Software which allows this value to be changed should warn the user that setting a value too low may cause overheating and hardware damage. Setting a value too high may cause locos to not respond to the DCC commands.

This value should only be altered by those who know what they are doing, and really needs to be done with an oscilloscope monitoring the waveform.

It has been made tunable so that it can be changed if different FETs are used, or substituted.

NV11 & NV12 – Mapped node

Default value: 0

Node number to recognise for mapped accessory commands – set to zero to use short events and map all device numbers to a DCC accessory and output address

NV13 – Send current interval

Default value: 0

If this value is non-zero, the command station will send a CBUS ACON2 event 1, with the command station node id, at regular intervals. The 2 data bytes of the ACON2 event contain the number of milliAmps being drawn from the main track output.

The value of this NV is the number of seconds between these reports.

Note that if the main track output is a booster, then these events will not contain meaningful data because the commands station has no way of knowing how much current is being drawn from any booster(s) in the system. There is an idea in gestation to allow for an input on the cancmd which routes to a spare A->D pin on the PIC which can be connected to a booster current monitor output.

The ACON2 event 1 can be changed by teaching a producer event, long or short. In the latter case, an ASON2 event is sent instead.

NV14 – Sod Delay

Default value: 0

If the start of day flag, in the user flags NV2, is set, then the command station will send a CBUS ACON event 0 after the initial startup delay of 2 seconds, plus the delay in this NV14. The delay in NV14 is half second increments, allowing an additional delay of 0.5 seconds to a little over 2 minutes.

The event to be sent as start of day can be changed by teaching a producer event.

NV15 – Shuttle Honk Interval

The counting interval between whistles or honks during shuttle operations (ie: it will whistle or honk every n iterations of the shuttle)

NV16 – Maximum Speed

Maximum speed setting. This acts as a speed limiter, and overrides any cab speed above this value. It is a DCC speed value, 1 to 127. Any number above that will have no effect and the default is 130.

This feature is useful ideal when the kids come to play, but also useful to make things happen slowly when debugging automation, such as debugging JMRI dispatcher operations so trains cannot run away at full speed when you get it wrong.

NV17 to NV144 – Shuttle Table

Version 4 contains a hard coded “proof of concept” shuttle facility, with fixed hard coded event values, which makes some use of the shuttle table.

If you would like to try out this feature, please contact Pete Brownlow for details.

Each of the 32 shuttle table entries contains 4 bytes.

Software that allows the user to set up the shuttle entries can hide the complexity of which NVs are which shuttle entry.

Bytes 1 and 2 – Default Loco Address – Setting this value is one way of setting up which loco is to operate a particular shuttle. There is also a mechanism to override this at run time by releasing a loco into a shuttle. A future possible enhancement will be to set a loco into a shuttle by an RFID detection.

Byte 3 - Default speed - If the speed is not modified by other shuttle events, this is the speed which will be set for a loco in this shuttle.

Byte 4 – Shuttle Flags

- Bit 0 - Valid — This bit is set to 1 when the shuttle entry is used
- Bit 1 - Started — Set to 1 in memory when the shuttle is active – start a shuttle by setting this bit (can also be started by a shuttle event)
- Bit 2 - Autostart — Set this bit to automatically start this shuttle whenever the command station starts.
- Bits 3-7 — Spare for future enhancements

EVENTS

The teachable events for CANCMD shuttles will be implemented in version 5.

MERGCMD version 5 will have a table of up to 128 taught events. This may be increased if it is found to be insufficient.

Each event has 8 event variables. The first EV contains flags. The meaning of the other event variables depends on the type of event, set in the flags.

There are two main categories of events the command station can respond to:

a) Shuttle events - a CBUS event, such as a track detection, can cause an action to a shuttle loco
b) Accessory events – a CBUS event can cause a DCC accessory command to be sent on to the track or secondary output.

Note that the same event can be taught multiple times with different actions and, on receipt of that event, the command station will carry out each action in the order taught (the event entries are chained in the command station memory). Teaching the same event again does NOT erase the previously stored action for that event. On receipt of an unlearn command, all actions associated with the event are forgotten.

EV1 – Event flags

This EV defines which kind of event this is and, for accessory events, the remaining bit flags of this EV define the behaviour of the accessory event.

- Bits 0,1 2 bit code for event type as follows:
  - 0 - Control event - General command station control
  - 1 - Accessory event - This event will cause a DCC accessory command to be sent out.
  - 2 - Shuttle event - Control for shuttle operations
  - 3 – Producer event – Use this event for the cancmd event number in EV2
- Bit 2 - On events Respond to on events
- Bit 3 - Off events Respond to off events
- Bit 4 - Polarity Set to reverse polarity, so that an on event is an off command

The meaning of the remaining 3 bits depends on the type of event.

For accessory events:

- Bit 5 - Toggle Toggle accessory value on receipt of event.
- Bit 6 – Use main When set, the accessory command is sent on the main track output. When clear, the command is sent on the “programming track” output. This allows this second output to be used for an dedicated accessory DCC bus.
- Bit 7 – Extended Use extended accessory packet format

For control events, the 3 bits are a code for the event action:

- 0 Stop all
- 1 Track power on/off
- 2 Swap main/program track output (on event, main is booster output)
- 7 Reset command station (if permit reset bit set in user flags)

EV2 – Delay

This EV defines a delay before the action occurs, in half second intervals, up to a maximum of 127.5 seconds. If zero, the action occurs immediately.

At present, the delay EV is not used for producer events. Its value has no effect.

The use of the remaining EVs depends on the type of event. There are no further EVs for control events.

Producer Events

EV3 – Producer event number to be overidden

EV3 contains the cancmd event number that will send this event instead of its default value.

The event number contained in EV3 is the default event number that would be sent by CANCMD is this producer event had not been taught. The taught event can be long or short, the appropriate opcode will be used when sending the taught short event. Appendix A lists the events that CANCMD can send.

Accessory Events

EV3 and EV4 – DCC accessory address

The format of EV3 and EV4 specifies the DCC accessory command to be sent when this CBUS event is received. It is one of two formats depending on whether the extended bit is set in EV1.

For standard accessory commands: 16 bit value in NMRA format containing a 9 bit value (0-511) of the DCC accessory address and 3 bit output select.

As per NMRA RP9.2.1, the nine bit address bits 0 to 2 are stored in bits 4-6 of EV4 and bits 3-8 are stored in bits 0-5 of EV3. The 3 bit output selection code is stored in bits 0-2 of EV4. The command station sets the other bits to the appropriate values as specified by NMRA when it expands the packet and transmits it onto the track, It does not matter what value the other bits are set to in the EVs.

For extended accessory commands, a 16 bit value containing 11 bit DCC accessory address and 5 bit aspect code.

For extended accessory commands, the 11 bit address is stored in bits 5-7 if EV4 and bits 0-7 of EV3. The aspect code is stored in bits 0-4 of EV4. This is a compression of the NMRA extended packet format. The command station expands this and fills in the other bits when the packet is transmitted onto the track.

EV4 – DCC accessory number

The number of the accessory within the decoder. This is 3 bits, typically bit 0 is used by decoders to specify which of a pair of outputs is to be turned on or off and the other 2 bits specify one of 4 devices on the decoder.

Shuttle Events

EV3 – Shuttle Number

The index into the shuttle table for the shuttle that this event will apply to.

EV4 – Event source and action code

This EV is divided into two 4 bit fields.

Event Source

Bits 0-3 are the event source. If all bits are set to 1 (Hex F) then the source is the CBUS event specified in this event entry. Other numbers indicate that the event is triggered by a transition on a direct input pin of the command station. These spare inputs are designated numbers from 0 upwards. How many (if any) such inputs are available will depend on the command station hardware.

This allows inputs, such as track detectors, to be connected directly to command station inputs, if available. Thus he command station can act as a limited shuttle controller standalone, without the need for any other CBUS nodes once set up.

Action code

Bits 4-7 are the action code. This is the action that will be applied to the specified shuttle on receipt of this event. Currently defined action codes are as follows:

0 – Do nothing
1 – Enable - Allocate a session to the loco on the stack
2 – End shuttle - halt and release loco
3 - Start - Start train moving
4 - Stop - Stop train
5 - Pause, - Stop and restart after a delay eg: station stop
6 - Reverse - Reverse direction
7 - Forwards - Set direction forwards
8 - Backwards - Set direction backwards
9 - Function - Send function (use EV func or override if event + data)
10 - Speed - Set new speed and direction
11 - Set loco - Set loco for shuttle (event + 1 data byte)
12 – Set counter - Set counter value for sequencing (uses first data byte if event + data)

EV5 – Speed

For operation codes that start or change direction, if this value is not hex FF, then the new speed and direction are taken from this EV. MS Bit defines direction. If it is set to hex FF, then the current speed is used. For the set counter action code, EV5 contains the new value to set.

This EV is overridden if the corresponding event + data is received, when the value is taken from the first data byte.

EV6 – Dwell

For the pause and reverse operations, this dwell time is inserted between the stop and restart. The dwell times can be set from 0 to 255 seconds. Note that this is the time between the stop command and the start command. Any deceleration time due to inertia settings programmed into the decoder are part of the dwell time.

EV7 – Random

The event will be actioned sometimes but not always. This allows for some randomisation, for example a train may sometimes stop at a station but not always.

A value of 0 means never and value of 255 means always, any probability can be set between those limits.

It is recommended that if the event is a train detector approaching the buffer stops that you set the stop event to always....

EV8 – Function

A function number to send to the loco when this event is actioned.

Bits 0 to 4 specify the function number.

Bit 5 is set to 1 for a function on and 0 for a function off
Bit 6 is set to 1 to send the function before the rest of the action, set to 0 for after the rest of the action
Bit 7 is set to 1 for momentary – on before the action and off afterwards

For the send function action code, if a corresponding event+data packet is received, the first data byte overrides the value in EV8.

EV9 – Counter

If EV9 is non-zero, only carry out the action if EV9 matches the current counter value. The counter value is dynamic value held in memory for each shuttle and initialised to zero when the shuttle is started. Its value can be changed on receipt of an event with a set counter action code.

This allows sequencing so that different things can happen on subsequent occurrences of the same event. For example, when shunting past the same track detector, points can be set for different sidings.

Appendix A

Events that cancmd can send.

CANCMD will send certain events, when configured to do so,

The default is a long event, using the node number assigned to CANCMD, with the event number listed below. Some events are event+data.

The default event to be sent can be overridden by teaching CANCMD a producer event. If a short producer event is taught, then the opcode sent will be changed accordingly. When teaching a producer event, set EV3 to the default event number in column 1.

Default Event Number	**Default Opcode**	**Meaning**
1	ACON2	Data bytes contain on board output instantaneous current measurement
2	ACON	Start of day (SOD)
3	ACON/ACOF	Track power is on /off *
4	ACON	Emergency stop actioned *

* These are sent in addition to the DCC opcodes for these actions, which enables CBUS consumers to be taught these events for additional control panel indications if desired.