Hadou-CAN User Guide

Table of Contents

Revision History

Date	Revision	Comments
20190419	-	Initial private release
20191024	A	Limited release
20200416	B	Public release
20200512	C	V08 release
20201013	D	V09 release

Introduction

The Suburban Embedded Hadou-CAN is a CAN to USB adapter capable of sustained 1 Mbps CAN 2.0A and 2.0B traffic as well as CAN FD with a fast data phase of up to 12 Mbps. USB ground is fully isolated from CAN ground, allowing isolation between CAN networks. A time synchronization feature can provide a common time base for recieved CAN traffic between multiple networks or external systems.

Theory of Operation

A USB PHY connects a microcontroller to a host PC. The host PC sends commands to the device which are interpreted as configuration data, commands, or CAN bus packets. The MCU loads outgoing packets into its CAN controller's priority queue which then drives the isolated CAN line driver when it is able to become a bus master.

Internal to the MCU, a 16-bit timer is captured automatically by the CAN controller when a packet is received. This timer can drive an outgoing pulse on rollover, or alternatively be asynchronously reset by an external timesync pulse per second (PPS) signal. The frequency and duty of the timesync line can be adjusted.

Electrical Specifications

Param	Min	Typical	Max	Unit
Input Voltage	4.4	5.0	5.25	V
Current		125	500	mA
Timesync high	2.0	3.3	3.6	V
Timesync low	0.0		0.8	V
Timesync impedance		50		ohm
Timesync current			+/-24	mA
CAN Termination		120		ohm

Mechanical Specifications

Param	Min	Typical	Max	Unit
Length		98		mm
Width		58		mm
Height		28		mm
Mass		98		g
Operating Temperature	0	20	50	°C
Storage Temperature	-20		60	°C

Connectors

Hadou-CAN Connectors

The following part numbers are currently used on the Hadou-CAN and provided as an aid to integration. Parts may be substituted in future builds without notice.

Connector	MF	MPN
USB	Molex Cheng Feng electronics	670688000 usb bf90
SMA	TE Connectivity AMP Connectors	619540-1
CAN	Assmann WSW Components	A-DS 09 A/KG-T2S

Hadou-CAN Pinout

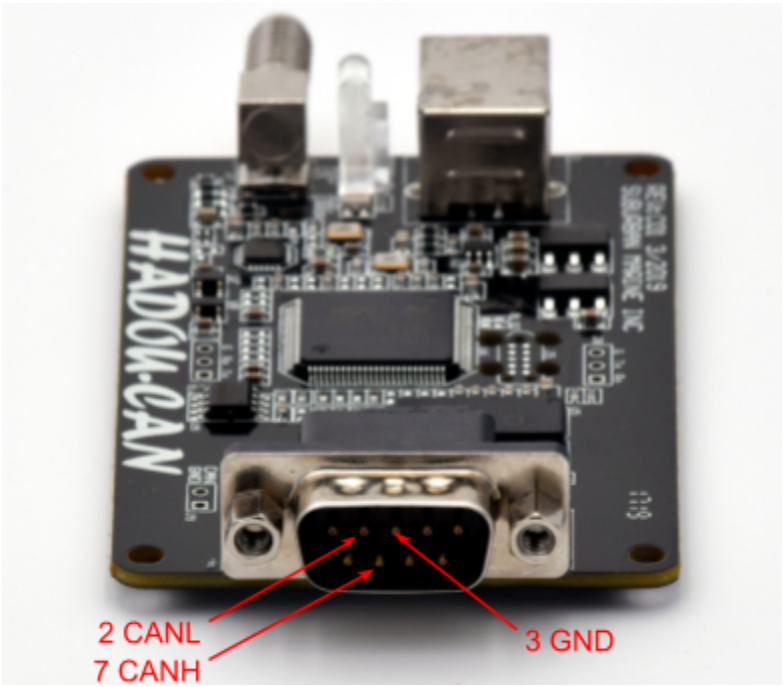
USB

Pin	Function
Shield	USB shield, connected to device ground through an inductor
1	USB VBus
2	USB D-
3	USB D+
4	USB Ground, connected to device ground through an inductor

SMA

Pin	Function
Shield	Ground, hard connected to device ground
Pin	Timesync, 3.3V 50 ohm

CAN Pinout



Pin	Function
2	CAN Low
3	CAN Ground, not connected to USB/SMA ground. May be omitted on short CAN runs if common mode voltage requirements are met.
7	CAN High

All other pins on the DE-9 connector are not connected internally and should be left open.

Suggested Mating Connectors and Cable Assembly

The following parts have been tested to work, however are for reference only as they may not meet your exact requirements. Generally, any commodity DE-9 or USB cable should work. Suburban Embedded will stock some accessories including parts shown below.

	MF	MPN	Notes
USB cable assembly, A male to B male, 2 meter	CNC Tech	102-1030-BL-00200	
CAN cable assembly, DE-9 F to DE-9 F, 120 ohm internal termination, 2 meter	Grid Connect	GC-CAN-CAB-2MT-GC	Recommended for high speed 500kbps+ operation. Impedance controlled & shielded. Remove internal termination jumpers if used.
CAN terminator, 120 ohm, inline DE-9	Grid Connect	GC-CAN-TERM-GC	Useful to avoid opening the case to attach the internal termination.
DE-9 socket, female, solder cup	Assmann WSW Components	A-DF 09 LL/Z	Uncontrolled hook-up wire may be used for short / low speed runs. Easy to hand solder.
DE-9 backshell	Assmann WSW Components	A-FT 09	Strongly recommended for custom DE-9 cables, provides strain relief.

LED Status

Both yellow	Bootloader is running, initializing & loading program
Both flashing red	Bootloader is running, waiting for command - application image may be corrupt or missing
Top solid green	Power on, application running
Bottom blinking red	CAN TX activity
Bottom blinking green	CAN RX activity

Protocol

The Hadou-CAN is a USB 2.0 High Speed device. It implements the USB CDC class device standard, and should show up as a virtual serial port with no additional driver. The protocol is similar to the LAWICEL AB CAN232 protocol, available at http://www.can232.com/docs/can232_v3.pdf, although certain modifications and extensions are made for CAN FD support and some additional functionality.

Packet Format

The can bus packet is encoded as an ascii text string as follows, followed by a carriage return (`\r`) character:

CAN 2.0A

tiiidXXYYZZ

t	CAN std packet
iii	id
d	DLC
XXYYZZ..	0 - 8 data bytes as indicated by DLC

CAN 2.0B

TiiiiiiiidXXYYZZ

T	CAN ext packet
iiiiiiii	id
d	DLC
XXYYZZ..	0 - 8 data bytes as indicated by DLC

CAN 2.0 RTR
If the RTR bit is set, r and R are used instead of t and T, with the id, dlc, and data fields encoded the same as before.

CAN FD STD

diiidXXYYZZ

d	CAN FD std packet
iii	id
d	Data length code ● 0 - 8: 0 - 8 ● 9: 12 bytes ● A: 16 bytes ● B: 20 bytes ● C: 24 bytes ● D: 32 bytes ● E: 48 bytes ● F: 64 bytes
XXYYZZ..	data bytes, length as indicated by dlc

CAN FD EXT

DiiiiiiiidXXYYZZ

D	CAN FD ext packet
iiiiiiii	id
d	Data length code ● 0 - 8: 0 - 8 ● 9: 12 bytes ● A: 16 bytes ● B: 20 bytes ● C: 24 bytes ● D: 32 bytes ● E: 48 bytes ● F: 64 bytes
XXYYZZ..	data bytes, length as indicated by dlc

CAN FD STD BRS

biiidXXYYZZ

b	CAN FD BRS std packet
iii	id
d	Data length code ● 0 - 8: 0 - 8 ● 9: 12 bytes ● A: 16 bytes ● B: 20 bytes ● C: 24 bytes ● D: 32 bytes ● E: 48 bytes ● F: 64 bytes
XXYYZZ..	data bytes, length as indicated by dlc

CAN FD EXT BRS

BiiiiiiiidXXYYZZ

B	CAN FD BRS ext packet
iiiiiiii	id
d	Data length code ● 0 - 8: 0 - 8 ● 9: 12 bytes ● A: 16 bytes ● B: 20 bytes ● C: 24 bytes ● D: 32 bytes ● E: 48 bytes ● F: 64 bytes
XXYYZZ..	data bytes, length as indicated by dlc

Commands

Items highlighted in red are planned, but not yet implemented.

Description	ASCII	Description
Open	O	Open channel
Close	C	Close channel
Listen	L	Open channel in listen only mode
Poll All	A	Poll all messages in buffer
Poll one	P	Poll single can message
Auto poll	X0 X1	Turn off auto poll Turn on auto poll
Set nominal bitrate	S[0-8]	Set nominal bitrate S0 - 10kbps S1 - 20kbps S2 - 50kbps S3 - 100kbps S4 - 125kbps S5 - 250kbps S6 - 500kbps S7 - 800kbps S8 - 1Mbps
CAN 2.0A	tiiidAABBCCDDEEFFGGHH	Send a 2.0A message iii - id 000 - 7FF d = length, 0 - 8
CAN 2.0B	TiiiiiiiidAABBCCDDEEFFGGHH	Send a 2.0B message iiiiiiii - id 00000000 - 1FFFFFFF d = length, 0 - 8
CAN 2.0A RTR	riiid	Send a 2.0A RTR message iii - id 000 - 7FF d = length
CAN 2.0B RTR	Riiiiiiiid	Send a 2.0B RTR

		message iiiiiiii - id 00000000 - 1FFFFFFF d = length
CAN FD std id	diiidd<payload in ascii hex>	iii - id 000 - 7FF d - [0-8, 9: 12, A: 16, B: 20, C: 24, D: 32, E: 48, F:64]
CAN FD ext id	Diiiiiiiid<payload in ascii hex>	iiiiiiii - id 00000000 - 1FFFFFFF d - [0-8, 9: 12, A: 16, B: 20, C: 24, D: 32, E: 48, F:64]
CAN FD BRS std id	biiidd<payload in ascii hex>	iii - id 000 - 7FF d - [0-8, 9: 12, A: 16, B: 20, C: 24, D: 32, E: 48, F:64]
CAN FD BRS ext id	Biiiiiiiid<payload in ascii hex>	iiiiiiii - id 00000000 - 1FFFFFFF d - [0-8, 9: 12, A: 16, B: 20, C: 24, D: 32, E: 48, F:64]
Get Flags	F	
Filter Mode	W	SJA1000 Compat Filter Setting 0: Dual 1: Single
Set accept code	Mxxxxxxxx	SJA1000 Compat Filter Setting
Set accept mask	mxxxxxxxx	SJA1000 Compat Filter Mask 1=don’t care
Get terse version	V	Vhhss hh - hardware id ss - software version ex: V0104
Get terse serial number	N	A 2 byte hash of an md5 hash of a 12 byte

		globally unique serial number
Enable timestamp	Z0 Z1	Turn timestamp off Turn timestamp on
Reboot to bootloader	!bootloader	Reboot to bootloader
Update configuration	!config:<doc>	Send new config document
Print configuration	!printconfig	
Restore default configuration	!defconfig	Restore defaults
Print bitrate table	!printtable	Print the bitrate table, showing tseg1, tseg2, sjw, etc for all kernel clocks.
Update bitrate table	!table	Send a new bitrate table
Get extended serial number	!serial	A 24 character hex string. The terse serial number is hashed, so will in general not be a substring of this string. This is the same as the serial number reported to the host over USB.
Get extended version number	!version	Vhhss hh - hardware id ss - software version ex: V0104

Timesync

The Hadou-CAN can act as either a time sync master or slave.

In master mode, an output pulse is sent once per second. The timesync output has a high-q 50 ohm filter for 50 ohm controlled impedance wiring. Coaxial cable and a 50 ohm termination load should be used to avoid ringing. The time sync is capable of driving a several foot cable with <10ns rise time when properly terminated.

When in slave mode, an internal timer is reset on the falling edge of an input 3.3V level pulse. The value of this timer is automatically recorded on incoming CAN packets if configured. See also the Z[01] command to display the timestamp when a packet is received.

Configuration

An XML document is stored internally to record several operating parameters. A table of time quanta is also configurable, which is critical for operation in CAN FD mode. Updating the configuration is accomplished by sending a new XML document on a single line after the `!config` command, `!config:<doc>`.

Updating Configuration

To update the configuration, send a new config document by sending it on a single line with the config command as so `!config:<doc>\r`.

General Configuration

Following is a terse description of permitted configuration fields. Items highlighted in red are not supported by current firmware versions.

Name	type	Description	Default Value	Allowed Values
config::version	uint		0	0
host_protocol	string	Upstream can protocol	lawicel_can232	lawicel_can232
autopoll	bool		false	true, false
listen_only	bool		false	true, false
auto_startup	bool		false	true, false
timesync_mode	string		slave	master, slave
slope_ctrl	string		auto	slow, fast
timestamp/enable	bool		false	true, false
timestamp/prescaler	uint	The prescaler for the input 100MHz clock	2000	0 - 65535
timestamp/period	uint	The period of the	50000	0 - 65535

		timestamp counter		
tx_delay_comp/enable	bool		false	true, false
tx_delay_comp/offset	uint			
tx_delay_comp/filter_window	uint			
can_clock	uint	The can kernel clock. This is the clock speed of the clock routed to the internal CAN controller. Changning It allows for different bitrates to be achieved with integer multiples of this clock.	24000000	24000000, 60000000, 80000000
bitrate/nominal	uint	Bitrate for 2.0A, 2.0B, and FD non-BRS packets.	500000	See bitrate table
bitrate/data	uint	Fast bitrate for FD BRS packet data segment	2000000	See bitrate table
protocol/ext_id	bool	Allow sending and receiving frames with 29bit id	true	true, false
protocol/fd	bool	Allow sending and receiving CAN FD frames	true	true, false
protocol/fd_iso	bool	Use ISO format CAN FD frames	true	true, false
protocol/brs	bool	Allow BRS frames to be sent and received	true	true, false
filter::type	string			sja1000
filter/enable	bool			
filter/type	string			dual, single
filter/accept_code	string			
filter/accept_mask	string			

debug/log_level				
debug/baud				

Example Configuration Document

```
<?xml version="1.0" encoding="UTF-8"?>
<config version="0">
   <autopoll>false</autopoll>
   <listen_only>false</listen_only>
   <auto_startup>false</auto_startup>
   <timesync_mode>slave</timesync_mode>
   <!--Set slope_ctrl to auto to enable fast slope control if bit
rates will be higher than 1 Mbps-->
   <!--Set slope_ctrl to fast for CAN FD BRS or maybe for ~1MBps-->
   <!--Otherwise set to slow for reduced EMI and better tolerance of
lower quality wiring-->
   <slope_ctrl>auto</slope_ctrl>
   <!--Enable tx_delay_comp to turn on loop delay compensation-->
   <!--Parameters offset and filter_window are in units of mtq, which
is 1/fdcan_tq_ck (configured by the clock parameter)-->
   <!--Propagation delay on the ADM3055E is about 150ns-->
   <tx_delay_comp>
      <enable>false</enable>
      <offset>5</offset>
      <filter_window>0</filter_window>
   </tx_delay_comp>
   <timestamp>
      <enable>false</enable>
      <prescaler>2000</prescaler>
      <period>50000</period>
   </timestamp>
   <!--clock may only be 24000000, 60000000, or 80000000-->
   <clock>24000000</clock>
   <bitrate>
      <nominal>500000</nominal>
      <data>2000000</data>
   </bitrate>
   <protocol>
      <ext_id>true</ext_id>
      <fd>true</fd>
      <brs>false</brs>
      <fd_iso>true</fd_iso>
   </protocol>
   <filter>
      <accept_code>00000000</accept_code>
```

```
      <accept_mask>FFFFFFFF</accept_mask>
   </filter>
   <debug>
      <!--log_level may be TRACE, DEBUG, INFO, WARN, ERROR, FATAL-->
      <log_level>INFO</log_level>
      <uart_baud>921600</uart_baud>
   </debug>
</config>
```

Bitrate Configuration

The following bitrate tables are the default. Other tables can be loaded for compatibility with other systems. Certain bit rates, and especially CAN FD, are sensitive to the sample position, eg, pre, tseg1, tseg2, and sjw must match exactly on all nodes in the system.

Generally, the 24MHz should be used as it is easily divided and is commonly used. Other kernel clocks should only be used if specific tseg and unusual bitrates are needed.

24MHz Kernel Clock Table

rate	pre	tseg1	tseg2	sjw
5000	192	16	8	2
10000	120	16	3	2
20000	60	16	3	2
50000	24	16	3	2
100000	12	16	3	2
125000	12	13	2	1
250000	6	13	2	1
500000	3	13	2	1
800000	3	7	2	1
1000000	3	5	2	1
2000000	2	4	1	1
6000000	1	1	2	1

60MHz Kernel Clock Table

rate	pre	tseg1	tseg2	sjw
250000	24	7	2	1
500000	12	7	2	1
1000000	5	8	3	1
2000000	3	7	2	1
4000000	3	3	1	1
6000000	2	3	1	1
10000000	1	3	2	1
12000000	1	2	2	1

80MHz Kernel Clock Table

rate	pre	tseg1	tseg2	sjw
250000	20	12	3	1
500000	10	12	3	1
1000000	10	5	2	1
2000000	4	7	2	1
4000000	2	7	2	1
8000000	4	7	2	1
10000000	1	5	2	1

Bootloader

A bootloader is implemented to allow field update for future features and bug fixes. The factory firmware is encrypted, although the bootloader will also allow un-encrypted firmware to be loaded and run if desired. Contact us for details if you want to load your own firmware. We can share a linker script that can illustrate what address to which the program must be loadable.

Bootloader Mode

Bootloader mode may be entered upon software request (command `!bootloader\n`) or by connecting the logic level CAN TX pin to ground with a jumper during startup. Only connect this pin to ground while the board is off or you may damage the pin, as the MCU will be driving it!

Bootloader Commands

The bootloader commands are currently similar to Android's FastBoot. They will be updated to USB DFU standard in the future. Contact us for details and an example script.

Bootloader Filenames

The following files are searched for in the order below. The first one found is loaded and if it is valid the bootloader will jump to the loaded program's reset vector to begin execution. If no bootable file is found, the device will remain in bootloader mode. The program image is loaded to and run from internal ram. Encrypted firmware is preferred, as AES-GCM encryption is used to ensure firmware integrity during load. Intel hex format will also verify a weak checksum. If a raw binary image is loaded, no verification is performed and there is no protection against errors at boot time.

app.bin.enc / app.bin.enc.xml	Encrypted firmware and IV/tag metadata, AES-GCM-128 mode.
app.hex	Intel hex format image
app.bin	Raw binary image

Operating Notes

CAN FD

CAN FD is extremely sensitive to the time quanta used, especially when bit rate switching is enabled. The bit rate switches mid-bit, and if the time quanta is different on various nodes, they will lose synchronization and will often reject CAN FD BRS packets. You must use the same CAN kernel clock and time quanta settings across the network. The Hadou-CAN allows you to upload a custom time quanta table, if desired.

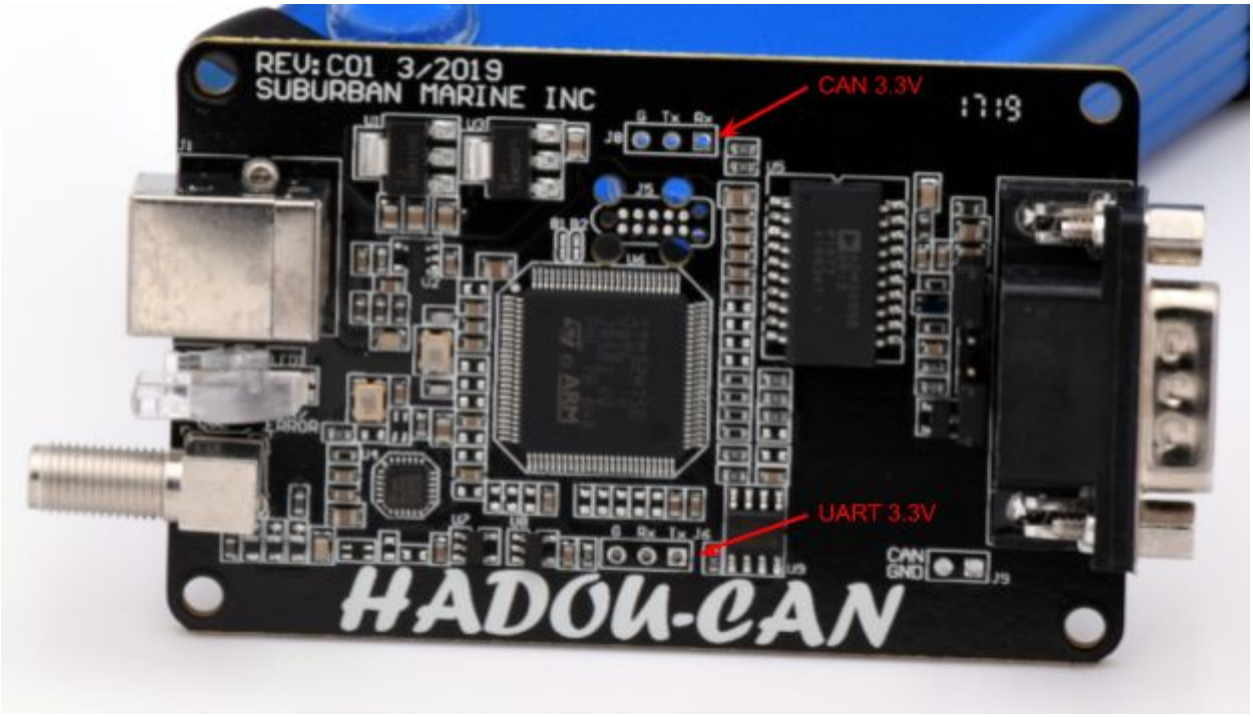
USB ACM

The host operating system may send some bytes to the Hadou-CAN during enumeration. This can interfere with decoding of the first command. Send a carriage return to synchronize the application and reject these bytes,

Timesync Output Drive

The timesync pin is buffered and filtered for 50 ohm drive. The filter will not limit current or drop appreciable voltage.

Debugging

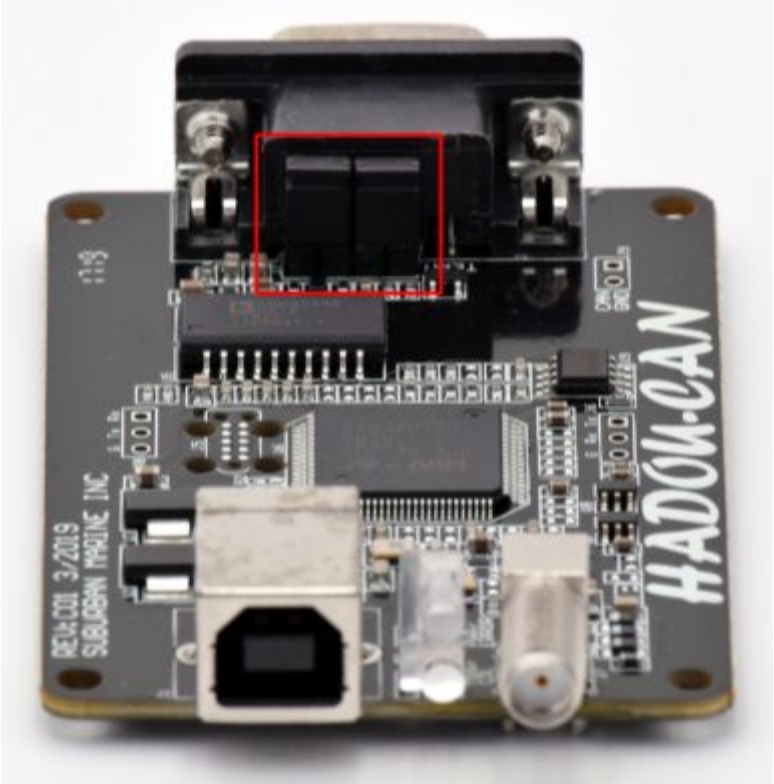


A 3.3V logic level UART is present on J6 that will output boot and diagnostic messages. A 3.3V logic level CAN header, J8, is also present, which can be used as a logic analyzer or oscilloscope connection point. The CAN header is useful for diagnosing bit rate issues and verifying loop delay time.

Power

The Hadou-CAN has an inrush limiting circuit that limits the capacitive load presented to the USB bus and limits the fault / short circuit current to safe levels.

Internal CAN Bus Termination



Internal termination resistors can be selected by inserting 2x 0.1in jumpers onto pins 1&2 and 3&4 of J2, located adjacent to the DE9 connector. This will connect a 120 ohm split termination right after the D Sub connector. The device ships from the factory with the termination enabled.

Storage

Program image and configuration are stored on external NOR flash with minimum 20 year retention and minimum 100k program / erase cycles. It is possible, although unlikely, to exceed the program / erase cycle life of the device through repeated configuration or bootloading. Wear leveling is implemented internally.

Program Image

The program image is loaded from external flash into internal ram for execution. It is encrypted with Galois/Counter mode AES to provide integrity verification.

Firmware Version History

SW version as indicated by V command	Notes
0A (future release)	202010XX Fix RTR packet string Add SJA1000 compat CAN id filters
09	20201007 Fix listen_only mode Fix lights and bell in FD mode
08	20200512 Fix notification endpoint Move driver to interrupt mode Add b and B commands for per-packet BRS configuration Enable FD BRS RX by default Reduce logging
07	20200409 Support configuring ISO and non-ISO CAN FD mode Logging updates
06	Limited release
05	Internal only
04	20191024 Change USB CDC protocol to V250 for in-box windows driver support Add L command Add s command Fix power cycle needed for bit rate to take effect Fixed outgoing packet format to include DLC so slcand/wireshark can work as monitors Fixed deadlock in O command
03	20190828 Public beta Adds windows support using ST driver Fixes 250kbps bitrate table entry Adds Z command

	Adds Q command
02	20190715 Public beta Fix issues with config transfer over USB & time sync
01	Private alpha, initial release

Hardware Version History

HW version as indicated by V command	Notes
01	Initial release. Copper revision C, configuration 01.

Errata

USB 2.0 Compliance

VBUS

The SM-1301 is not compliant with USB ECN “USB 2.0 VBUS Max Limit”. The maximum permissible applied voltage applied to VBUS is 5.25V.

ADP

The 1uF minimum capacitance is behind a ferrite, so ADP may fail.

VBUS Capacitance

The device may discharge more than 10uF into the host when the host is turned off.

Suspend Mode

Device suspend is not supported, and the device may continue to draw nominal operating current or malfunction until a power cycle if the host initiates a suspend.

Firmware Errata

- USB suspend is not implemented
- Most USB CDC class requests are not implemented and may cause undefined operation
- You may need to read from the serial port before you can send commands. Processing of some incoming commands may stall if the USB tx queue is full.

Source Code

Hadou-CAN as an open source project. The encryption keys for factory firmware are currently not published, you will need to reflash your own bootloader with new keys to run modified encrypted software, or load unencrypted .hex or .bin files. Contact us for details.

https://github.com/suburbanembedded/hadoucan-fw
https://github.com/suburbanembedded/hadoucan-bootloader

About Us

Suburban Marine, Inc. is a nimble engineering consulting firm in southern California. We strive to bring advanced technology to the marine environment. Under our Suburban Embedded trade name, we work on advanced aerospace flight software, cutting edge safety critical electronics, and provide turnkey prototype development services.

Contact Us

Please send questions, comments and order inquiries to sales@suburbanembedded.com.

References

User Guide
https://suburbanmarine.io/public/hadoucan/doc/Hadou-CAN_User_Guide.pdf

Datasheet
https://suburbanmarine.io/public/hadoucan/doc/Hadou-CAN_Datasheet.pdf

Quickstart Guide
https://suburbanmarine.io/public/hadoucan/doc/Hadou-CAN_Quickstart_Guide.pdf