USER MANUAL UMAX141000
Version 1.0.3

DATA LOGGER

CAN/RS-232/Ethernet

with Linux

USER MANUAL

P/N: AX141000

Axiomatic Technologies Oy
Höytämöntie 6
33880 LEMPÄÄLÄ, Finland
Tel. +358 103 375 750
salesfinland@axiomatic.com
www.axiomatic.fi

Axiomatic Technologies Corporation
1445 Courtneypark Dr. E.
Mississauga, ON Canada L5T 2E3
Tel. 1 905 602 9270
sales@axiomatic.com
www.axiomatic.com

VERSION HISTORY

Version	Date	Author	Modifications
1.0.0	Mar. 21, 2024	Antti Keränen	Initial draft
1.0.1	Mar. 28, 2024	M Ejaz	Marketing review Updated technical specifications
1.0.2	Oct. 22, 2025	M Ejaz	Updated dimensional drawing Added drawings of mating cables
1.0.3	Oct. 27, 2025	M Ejaz	Updated technical specifications Updated the drawing of AX070532 cable

ACRONYMS

BATT +/-	Battery positive (a.k.a. Vps) or Battery Negative (a.k.a. GND)
DIN	Digital Input used to measure active high or low signals
SYNC	Digital Input used to synchronize datalogging start and stop
GND	Ground reference (a.k.a. BATT-)
Vps	Voltage Power Supply (a.k.a. BATT+)

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

REFERENCES

TDAX141000 Technical Datasheet, Data Logger, CAN/RS-232/Ethernet, Axiomatic Technologies

1. OVERVIEW OF THE DATA LOGGER

The AX141000 data logger runs a custom OpenSTLinux distribution. The device has 128GB eMMC for storing both the operating system and the logged data. A supercap provides backup power for the CPU for finalizing the data log save before the CPU shuts down in case of a power failure and an active low SYNC pin is used for data logging start and stop synchronization.

The available communications interfaces include 1Gb Ethernet, two CANs and one RS-232. The default data logging application supports CAN data logging, the RS-232 port runs Linux console by default.

The device runs SSH and SFTP servers. Linux can-utils are included.

1.1. Data logging operation

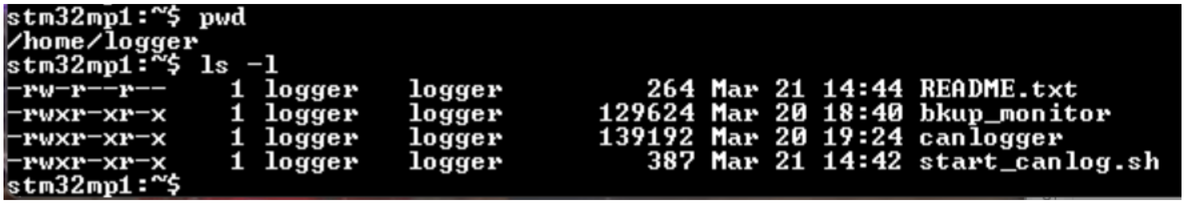
The operating system has two user accounts by default: logger and root. The RS-232 on the M12 male connector A (see also section 2.1) supports Linux console (115200, 8N1). The AX141000 runs SSH and SFTP servers by default, the IP address is set using DHCP.

Default user accounts and passwords:
logger, password: **kukkuu**
root, password: **ax022003**

The default installation contains software for both data logging and backup power management. These utilities can be started from command line using the following commands:

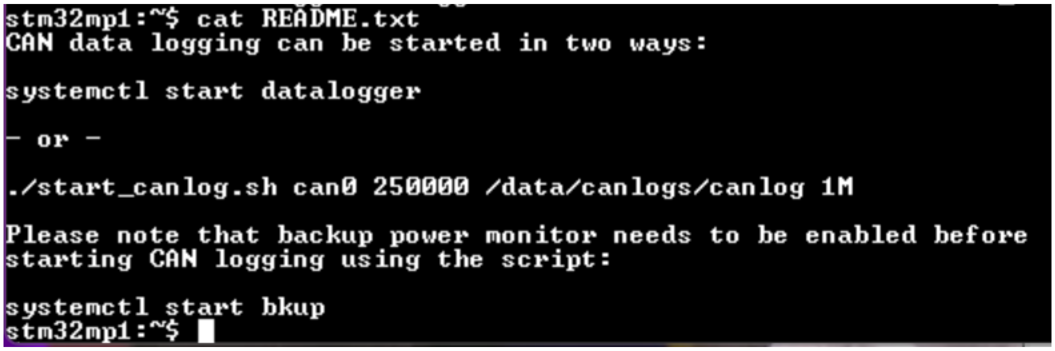
```
systemctl start bkup

systemctl start datalogger
```

By default, the datalogger utility (*/home/logger/canlogger*) expects that the backup power manager is running, so the bkup.service needs to be started first. Starting the datalogger.service calls the script “start_canlog.sh”, the datalogger utility’s settings, including backup power status monitoring is defined in that script.



```
stm32mp1:~$ pwd
/home/logger
stm32mp1:~$ ls -l
-rw-r--r--    1 logger   logger        264 Mar 21 14:44 README.txt
-rwxr-xr-x    1 logger   logger     129624 Mar 20 18:40 bkup_monitor
-rwxr-xr-x    1 logger   logger     139192 Mar 20 19:24 canlogger
-rwxr-xr-x    1 logger   logger        387 Mar 21 14:42 start_canlog.sh
stm32mp1:~$
```

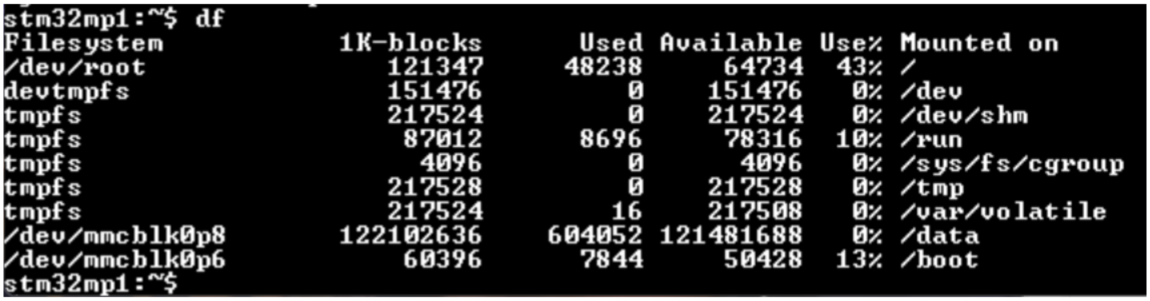
Figure 1 – /home/logger directory contents



```
stm32mp1:~$ cat README.txt
CAN data logging can be started in two ways:

systemctl start datalogger

- or -

./start_canlog.sh can0 250000 /data/canlogs/canlog 1M

Please note that backup power monitor needs to be enabled before
starting CAN logging using the script:

systemctl start bkup
stm32mp1:~$
```

Figure 2 – README.txt

The CAN data will be stored to */data/canlogs* directory which is on the separate data partition on the eMMC.



```
stm32mp1:~$ df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/root               121347     48238     64734  43% /
devtmpfs                151476         0    151476   0% /dev
tmpfs                   217524         0    217524   0% /dev/shm
tmpfs                    87012      8696     78316  10% /run
tmpfs                     4096         0      4096   0% /sys/fs/cgroup
tmpfs                   217528         0    217528   0% /tmp
tmpfs                   217524        16    217508   0% /var/volatile
/dev/mmcblk0p8       122102636    604052 121481688   0% /data
/dev/mmcblk0p6           60396      7844     50428  13% /boot
stm32mp1:~$
```

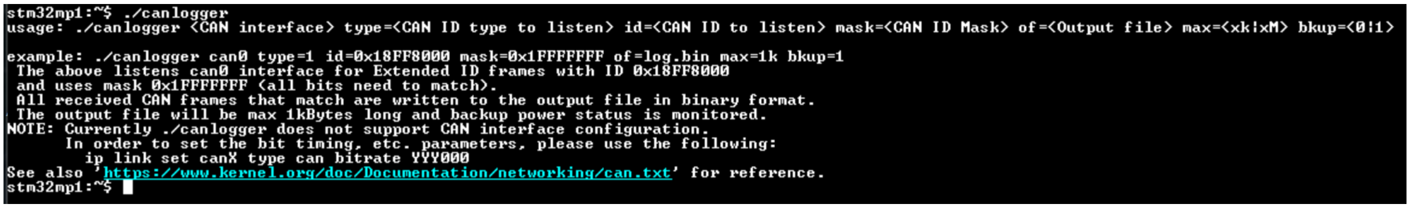
Figure 3 – eMMC partitions

In case the datalogging needs to be started automatically at power on, the following commands can be used for enabling that functionality.

```
systemctl enable bkup
```

```
systemctl enable datalogger
```

The datalogger utility can be configured using command line arguments. The default startup script */home/logger/start_canlog.sh* starts the utility for logging all received CAN frames from interface **can0** (CAN1 pins on the M12 female connector C, described in section 2.1) to log files that are maximum 1MB in size.



```
stm32mp1:~$ ./canlogger
usage: ./canlogger <CAN interface> type=<CAN ID type to listen> id=<CAN ID to listen> mask=<CAN ID Mask> of=<Output file> max=<xk|xM> bkup=<0|1>

example: ./canlogger can0 type=1 id=0x18FF8000 mask=0x1FFFFFFF of=log.bin max=1k bkup=1
 The above listens can0 interface for Extended ID frames with ID 0x18FF8000
 and uses mask 0x1FFFFFFF (all bits need to match).
 All received CAN frames that match are written to the output file in binary format.
 The output file will be max 1kBytes long and backup power status is monitored.
NOTE: Currently ./canlogger does not support CAN interface configuration.
      In order to set the bit timing, etc. parameters, please use the following:
         ip link set canX type can bitrate YYY000
See also 'https://www.kernel.org/doc/Documentation/networking/can.txt' for reference.
stm32mp1:~$ 
```

Figure 4 – 'datalogger' command line arguments

Note that to enable data logging using the *canlogger* uility, **the SYNC pin needs to be connected to GND**. Disconnecting the SYNC pin will stop data logging and close the data log file. A new file will be opened on the consequent SYNC pin connection to GND.

1.2. Backup power management

The backup power manager reports the power status using a TCP socket at port 3000 on local loopback interface (127.0.0.1). The status is an 8-bit data with following flags:

0x01 – Backup operation ready

0x02 – Power good

1.3. Converting CAN data log files

The CAN frames are stored to the CAN data log binary file with a timestamp. The binary file uses a proprietary file format, that can be converted to Linux can-utils candump format using a software tool available from Axiomatic (source code also available in Appendix B, compiles for example with MinGW)

```
---------------------------------------------------------------------------------------------------------
Datalog (binary) file contents:
---------------------------------------------------------------------------------------------------------
(32 byte header timestamp)
(24 byte CAN data entry #1: 8 byte timestamp, 16 byte CANframe)
(24 byte CAN data entry #2: 8 byte timestamp, 16 byte CANframe)
...
```

2. INSTALLATION INSTRUCTIONS

2.1. Dimensions and Pinout

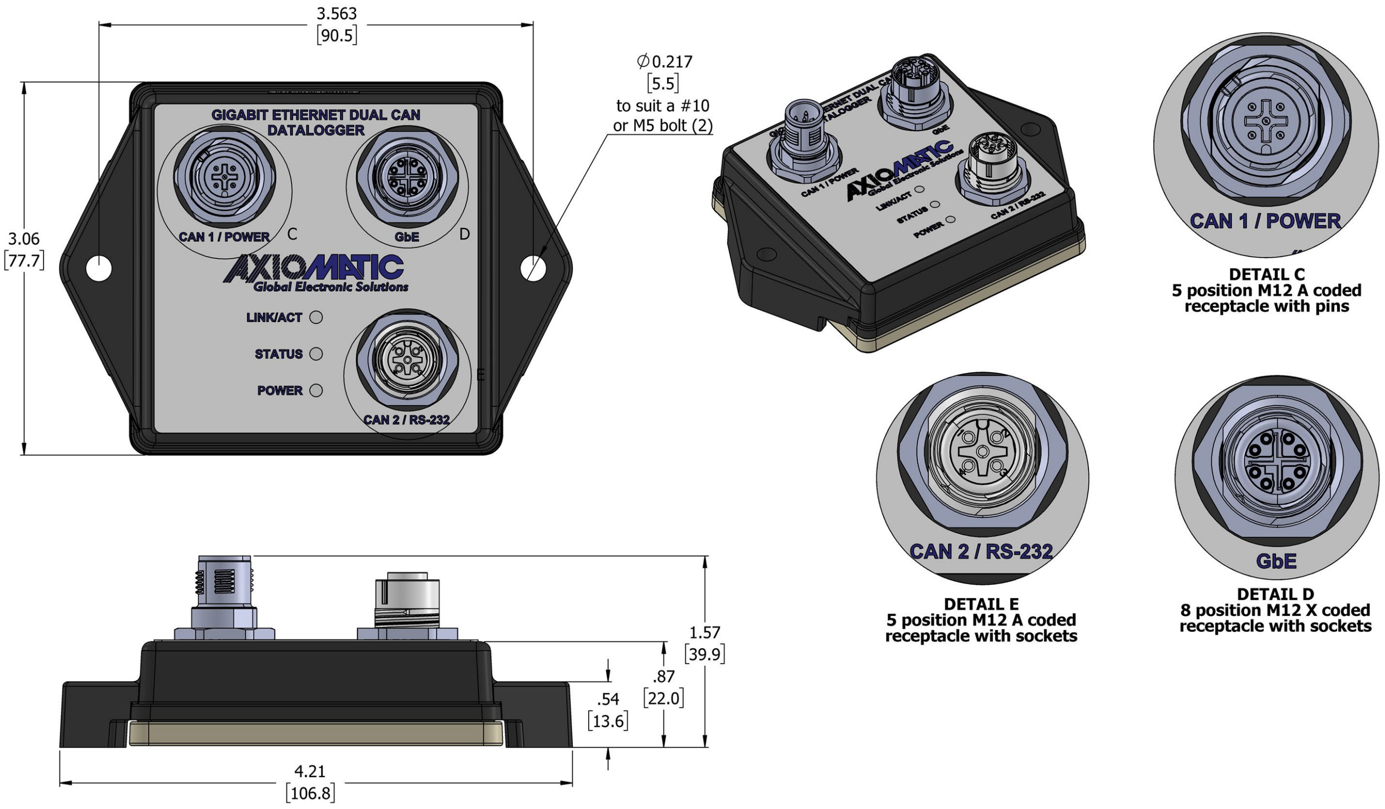
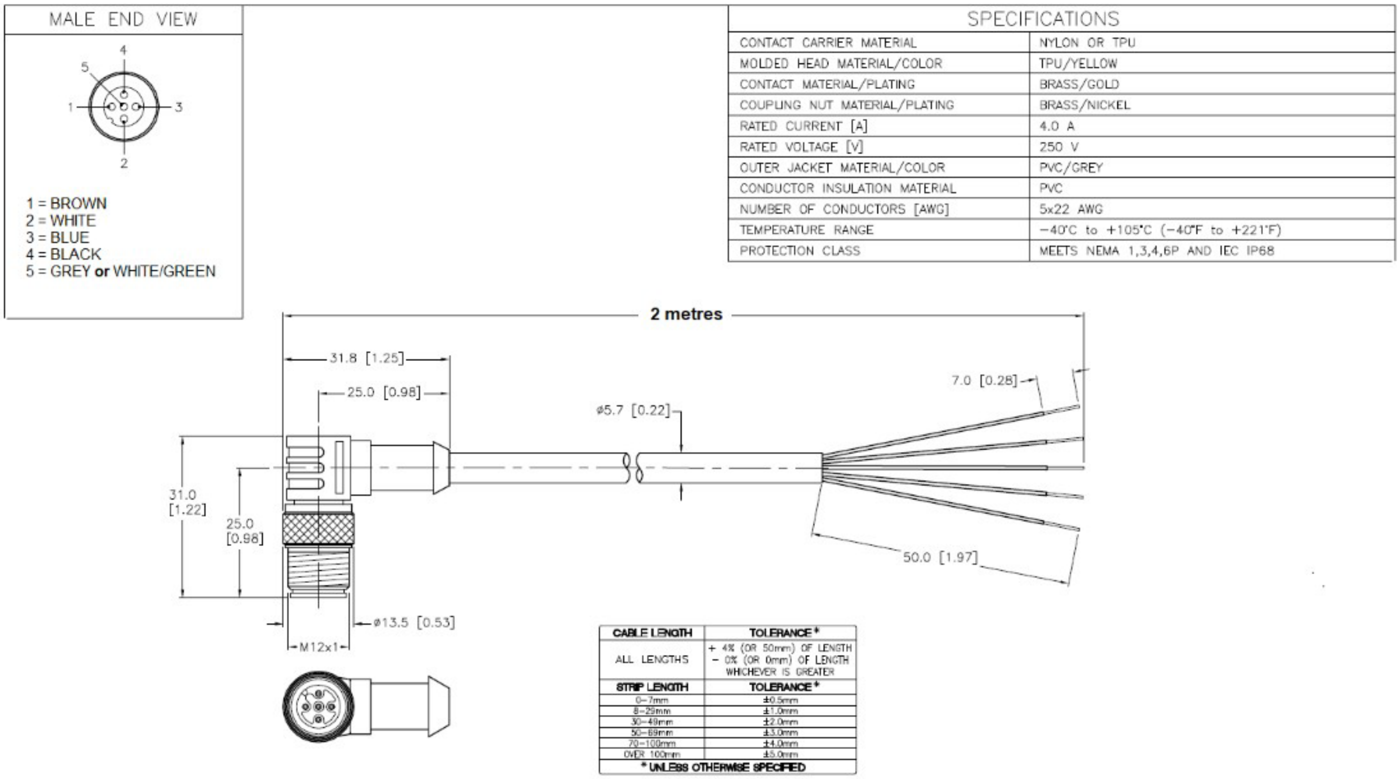


Figure 5 – AX141000 Dimensional Drawing

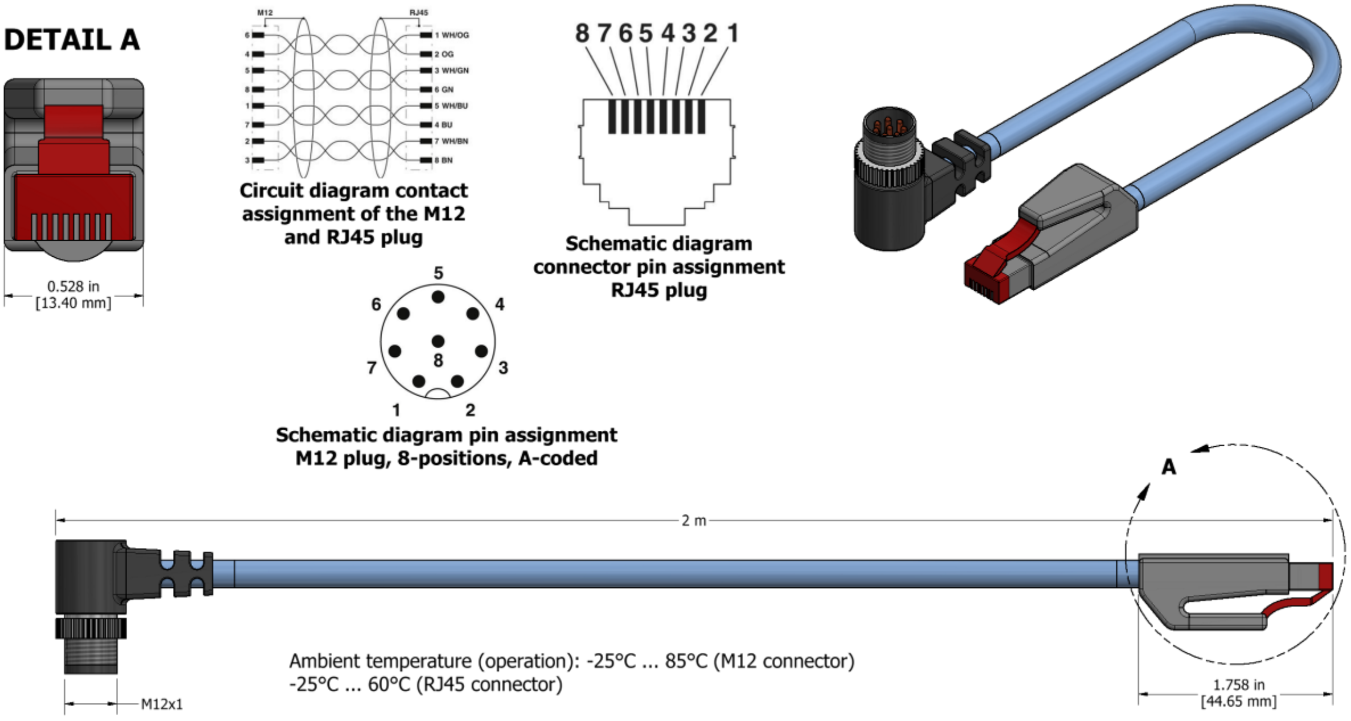


1 = BROWN
2 = WHITE
3 = BLUE
4 = BLACK
5 = GREY or WHITE/GREEN

SPECIFICATIONS	
CONTACT CARRIER MATERIAL	NYLON OR TPU
MOLDED HEAD MATERIAL/COLOR	TPU/YELLOW
CONTACT MATERIAL/PLATING	BRASS/GOLD
COUPLING NUT MATERIAL/PLATING	BRASS/NICKEL
RATED CURRENT [A]	4.0 A
RATED VOLTAGE [V]	250 V
OUTER JACKET MATERIAL/COLOR	PVC/GREY
CONDUCTOR INSULATION MATERIAL	PVC
NUMBER OF CONDUCTORS [AWG]	5x22 AWG
TEMPERATURE RANGE	−40°C to +105°C (−40°F to +221°F)
PROTECTION CLASS	MEETS NEMA 1,3,4,6P AND IEC IP68

CABLE LENGTH	TOLERANCE *
ALL LENGTHS	+ 4% (OR 50mm) OF LENGTH − 0% (OR 0mm) OF LENGTH WHICHEVER IS GREATER
STRIP LENGTH	**TOLERANCE ***
0–7mm	±0.5mm
8–29mm	±1.0mm
30–49mm	±2.0mm
50–69mm	±3.0mm
70–100mm	±4.0mm
OVER 100mm	±5.0mm
* UNLESS OTHERWISE SPECIFIED	

Figure 6 – AX070532 Mating Cable



Circuit diagram contact assignment of the M12 and RJ45 plug

Schematic diagram connector pin assignment RJ45 plug

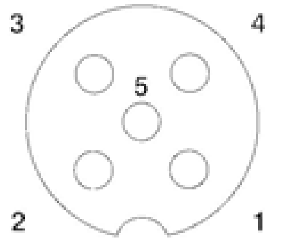
Schematic diagram pin assignment M12 plug, 8-positions, A-coded

Ambient temperature (operation): -25°C ... 85°C (M12 connector)
-25°C ... 60°C (RJ45 connector)

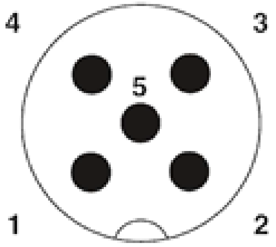
Degree of protection: IP65 (M12 connector), IP67 (M12 connector when plugged), IP20 (RJ45 connector)

Figure 7 – AX070535 Mating Cable

5-pin Phoenix Contact M12, female connector (A-coded), P/N: 1441778

Pin	Description	
1	SYNC	
2	Power +	
3	Power -	
4	CAN 1 High	
5	CAN 1 Low	

5-pin Phoenix Contact M12, male connector (A-coded), P/N: 1441765

Pin	Description	
1	RS-232 RX	
2	RS-232 TX	
3	Ground	
4	CAN 2 High	
5	CAN 2 Low	

8-pin Phoenix Contact M12, female connector (A-coded), P/N: 1441817

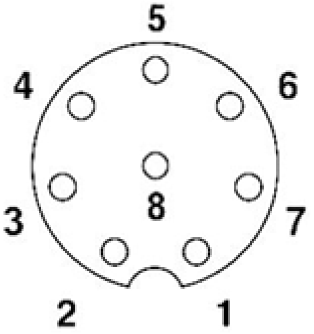
Pin	Description	
1	BI DC P	
2	BI DD P	
3	BI DD N	
4	BI DA N	
5	BI DB P	
6	BI DA P	
7	BI DC N	
8	BI DB N	

Table 1 – AX141000 Connector Pinout

APPENDIX A - TECHNICAL SPECIFICATION

Specifications are indicative and subject to change. Actual performance will vary depending on the application and operating conditions. Users should satisfy themselves that the product is suitable for use in the intended application. All our products carry a limited warranty against defects in material and workmanship. Please refer to our Warranty, Application Limitations & Return Materials Process as described on https://www.axiomatic.com/service/.

Power Input

Power Supply Input	12 or 24 VDC nominal (8 to 36 VDC range)
Quiescent Current	80 mA @12 V, 50 mA @ 24V
Protections	Surge and transient protection Reverse polarity protection up to -50 V. Under-voltage protection. Hardware shuts down at 4 VDC. Over-voltage protection. Hardware shuts down at 38 VDC.
Backup Battery	Supercapacitor backup of 3 seconds for ensuring data integrity

CAN Port

CAN	2 CAN ports (CAN 2.0B) SAE J1939 / CANopen® compliant Baud rate: 250 kbit/s (default) CAN FD supported (Contact Axiomatic for the supported data transfer speeds)
Protection	ESD protection for signal lines
SYNC	Digital input for starting or stopping CAN recording Min. 4 V for Input High Max. 1 V for Input Low Max voltage on input: 0 to Vps

RS-232 Port

RS-232	1 RS-232 port Default baud rate: 115200 kbit/s
Protection	ESD protection for signal lines
User Interface	Any terminal emulator that supports serial communication. Default functionality: Linux root console

Ethernet Port

Port Type	1 port 10/100/1000 Mbit/s Ethernet Auto-negotiation and full-duplex supported Automatic polarity correction TCP/IP
MDIX	Auto-MDI/MDIX (crossover)
PHY	KSZ9131RNXI (10BASE-T, 100BASE-T, 1000BASE-T)
Protocol	Ethernet IEEE 802.3bw or IEEE 802.3 (2015)
Protection	ESD protection for signal lines
User Interface	SSH & SFTP server

General Specifications

<table>
<tr><td>Microcontroller</td><td>STM32MP157A, 32-bit, 512 MB RAM
128 GB flash program memory (eMMC) out of which the default system uses ~200 MB</td></tr>
<tr><td>LED Indicators</td><td>POWER LED
GREEN = System Booted and Ready
RED = System in Backup Mode and powering down

LINK LED
Bicolor LED for Ethernet only
<table>
<tr><th>LED Color</th><th>State</th><th>Description</th></tr>
<tr><td>OFF</td><td></td><td>No Link</td></tr>
<tr><td rowspan="2">GREEN</td><td>On</td><td>1000 Link / No Activity</td></tr>
<tr><td>Blink</td><td>1000 Link / Activity (RX, TX)</td></tr>
<tr><td rowspan="2">RED</td><td>On</td><td>100 Link / No Activity</td></tr>
<tr><td>Blink</td><td>100 Link / Activity (RX, TX)</td></tr>
<tr><td rowspan="2">YELLOW / ORANGE</td><td>On</td><td>10 Link / No Activity</td></tr>
<tr><td>Blink</td><td>10 Link / Activity (RX, TX)</td></tr>
</table>

STATUS LED (software defined)
GREEN = Recording CAN
RED = Stop
OFF = Recording not started</td></tr>
<tr><td>Operating Temperature</td><td>-40 to 65 °C (-40 to 149 °F)</td></tr>
<tr><td>Storage Temperature</td><td>-40 to 70 °C (-40 to 158 °F)</td></tr>
<tr><td>Compliance</td><td>RoHS Directive</td></tr>
<tr><td>Enclosure and Dimensions</td><td>Injection molded enclosure and cover, Laser welded
PA66, 30% glass fiber reinforced
4.21 in. x 3.06 in. x 1.57 in. (106.8 mm x 77.7 mm x 39.9 mm)
L x W x H including connectors
See dimensional drawing.
Flammability Rating: UL 94 HB</td></tr>
<tr><td>Weight</td><td>0.283 lb. (0.128 kg)</td></tr>
<tr><td>Protection Rating</td><td>IP67
IEC 60529 with mated connectors</td></tr>
<tr><td>Installation</td><td>Suits two M5 or #10 mounting bolts.</td></tr>
<tr><td>Mating Cables</td><td>The following part numbers are available from Axiomatic.

<b>AX070532</b> CAN Cable - 1.5 m (5 ft.), 5-pin M12 A-coded, Unterminated Leads, can be used for experimenting. The cable is rated for -40 to 105 °C.

<b>AX070139</b> CAN Cable - 1.5 m (5 ft.), 5-pin M12 A-coded, Unterminated Leads

<b>AX070535</b> Ethernet Cable 2 m (6.5 ft.), 8-pin M12 A-coded, Ethernet Jack
Note: Cable supplier is Phoenix Contact Network cable NBC-M12MR/2,0-94B/R4AC US – 1406112. The M12 connector on the harness assembly is rated for -20 to 85 °C and the RJ45 ethernet jack is rated as -20 to 60 °C.</td></tr>
</table>

APPENDIX B - DATA LOG FILE CONVERSION TOOL SOURCE CODE

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <time.h>
#include <Windows.h>

#define EXIT_AFTER_ERROR      \
        Sleep(3000);                  \
        return EXIT_FAILURE

int main(int argc, char *argv[]) {

        FILE *file_in, *file_out;
        char fileName_input[128], fileName_output[128], tmpStr[128];
        unsigned char line[64];
        size_t len = 0, dlen;
        ssize_t read;
        float fTS = 0.0, fmsec;
        unsigned int i, uiIDh;
        int firstLines, framesConverted;
        time_t seconds;

        if( argc == 3 ) {
                memset((void *)fileName_input, 0x00, sizeof(fileName_input));
                strncpy(fileName_input, argv[1], sizeof(fileName_input));
                memset((void *)fileName_output, 0x00, sizeof(fileName_output));
                strncpy(fileName_output, argv[2], sizeof(fileName_output));
        }
        else {
                printf("Usage: %s canlog.file canlog.dump\n", argv[0]);
                EXIT_AFTER_ERROR;
        }

        /*
         * Reading in the lines from the specified file
         */
        printf("Converting \"%s\" to \"%s\"...\n", fileName_input, fileName_output);

        if( (file_in = fopen(fileName_input, "rb")) == NULL ) {
                printf("ERROR opening \"%s\" for reading\n", fileName_input);
                EXIT_AFTER_ERROR;
        }
        if( (file_out = fopen(fileName_output, "w")) == NULL ) {
                printf("ERROR opening \"%s\" for writing\n", fileName_output);
                EXIT_AFTER_ERROR;
        }

        // Read lines from original file and write them into the new one
        /*
        ----------------------------------------------------------------
        Datalog (binary) file contents:
        ----------------------------------------------------------------
        (32 byte header timestamp)
        (24 byte CAN data entry #1: 8 byte timestamp, 16 byte CANframe)
        (24 byte CAN data entry #2: 8 byte timestamp, 16 byte CANframe)
        ...
        ----------------------------------------------------------------
        CANDUMP file contents:
        ----------------------------------------------------------------
        (1519238827.147822) can0 03029000#0000000000000000
        (1519238827.152676) can0 06254000#00FFFF00FFFFFF00
        (1519238827.157823) can0 03029100#0000000000000000
        ----------------------------------------------------------------
        */
        // Setting a flag for skipping over first lines with no CAN frames
        firstLines = 1;
        len = 32; // first read is the header timestamp
        framesConverted = 0;

        while( (read = fread(line, len, 1, file_in)) != -1 ) {
                if( read > 0 ) {
                        // Got something, parsing contents
                        if(firstLines) {
                                printf("Converting file with timestamp: \"%s\"\n", line);
                                firstLines = 0;
                                len = 24;
                        }
                        else {
                                seconds = line[0] + (line[1]<<8) + (line[2]<<16) + (line[3]<<24);
                                fmsec = (float)(line[4] + (line[5] << 8));
                                uiIDh = (*(unsigned int *)&line[8]) & 0x1FFFFFFF;
                                dlen = line[12];

                                fTS = fmsec/1000.0;

                                //printf("frame @ %fs, ID=0x%X, len=%d\n", fTS, uiIDd, uiDLC);
                                memset((void *)tmpStr, 0x00, sizeof(tmpStr));
                                if(uiIDh <= 0x7FF) {
                                        snprintf(tmpStr, 128, "(%.6f) can0 %03X#",
                                                ((double)seconds+(double)fTS), uiIDh);
                                }
                                else {
                                        snprintf(tmpStr, 128, "(%.6f) can0 %08X#",
                                                ((double)seconds+(double)fTS), uiIDh);
                                }
                                fwrite((const void *)tmpStr, sizeof(char), strlen(tmpStr), file_out);

                                if(dlen>8) dlen=8;
                                else if(dlen<0) dlen=0;

                                memset((void *)tmpStr, 0x00, sizeof(tmpStr));
                                for(i=0;i<dlen;i++) {
                                        sprintf(&tmpStr[(i*2)], "%02X", line[16+i]);
                                }
                                strcat(tmpStr, "\n");
                                fwrite((const void *)tmpStr, sizeof(char), strlen(tmpStr), file_out);
                                framesConverted++;
                                firstLines=0;
                        }
                }
                else if(read == 0) {
                        // end of file.
                        break;
                }
                else {
                        printf("reading failed.\n");
                        break;
                }
        }

        // Closing files
        fclose(file_in);
        fclose(file_out);

        printf("Converted %d frames from bin log to dump.\n", framesConverted);

        return EXIT_SUCCESS;
}
```

OUR PRODUCTS

AC/DC Power Supplies

Actuator Controls/Interfaces

Automotive Ethernet Interfaces

Battery Chargers

CAN Controls, Routers, Repeaters

CAN/WiFi, CAN/Bluetooth, Routers

Current/Voltage/PWM Converters

DC/DC Power Converters

Engine Temperature Scanners

Ethernet/CAN Converters, Gateways, Switches

Fan Drive Controllers

Gateways, CAN/Modbus, RS-232

Gyroscopes, Inclinometers

Hydraulic Valve Controllers

Inclinometers, Triaxial

I/O Controls

LVDT Signal Converters

Machine Controls

Modbus, RS-422, RS-485 Controls

Motor Controls, Inverters

Power Supplies, DC/DC, AC/DC

PWM Signal Converters/Isolators

Resolver Signal Conditioners

Service Tools

Signal Conditioners, Converters

Strain Gauge CAN Controls

Surge Suppressors

OUR COMPANY

Axiomatic provides electronic machine control components to the off-highway, commercial vehicle, electric vehicle, power generator set, material handling, renewable energy and industrial OEM markets. ***We innovate with engineered and off-the-shelf machine controls that add value for our customers.***

QUALITY DESIGN AND MANUFACTURING

We have an ISO9001:2015 registered design/manufacturing facility in Canada.

WARRANTY, APPLICATION APPROVALS/LIMITATIONS

Axiomatic Technologies Corporation reserves the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. Users should satisfy themselves that the product is suitable for use in the intended application. All our products carry a limited warranty against defects in material and workmanship. Please refer to our Warranty, Application Approvals/Limitations and Return Materials Process at https://www.axiomatic.com/service/.

COMPLIANCE

Product compliance details can be found in the product literature and/or on axiomatic.com. Any inquiries should be sent to sales@axiomatic.com.

SAFE USE

All products should be serviced by Axiomatic. Do not open the product and perform the service yourself.

⚠ This product can expose you to chemicals which are known in the State of California, USA to cause cancer and reproductive harm. For more information go to www.P65Warnings.ca.gov.

SERVICE

All products to be returned to Axiomatic require a Return Materials Authorization Number (RMA#) from rma@axiomatic.com. Please provide the following information when requesting an RMA number:

- Serial number, part number
- Runtime hours, description of problem
- Wiring set up diagram, application and other comments as needed

DISPOSAL

Axiomatic products are electronic waste. Please follow your local environmental waste and recycling laws, regulations and policies for safe disposal or recycling of electronic waste.

CONTACTS

Axiomatic Technologies Corporation
1445 Courtneypark Drive E.
Mississauga, ON
CANADA L5T 2E3
TEL: +1 905 602 9270
FAX: +1 905 602 9279
www.axiomatic.com
sales@axiomatic.com

Axiomatic Technologies Oy
Höytämöntie 6
33880 Lempäälä
FINLAND
TEL: +358 103 375 750
www.axiomatic.com
salesfinland@axiomatic.com