Fuse board SW interface specification v1.1

2024-09-13

1. Introduction

This document specified the human and machines interfaces used to communicate with DefSecIntel Fuse Board firmware.

2. Overview

Fuse boards are used in a system that consists of multiple Fuse boards where one is configured as a master and others are configured as slave devices. Master Fuse board interfaces with the system (e.g. central controller) over Ethernet and with slave Fuse boards over the RS-485 interface.

From a system control point of view the master and slaves appear as a single device where master is the entry point. It means that when a command is sent to master that concerns slaves, the master communicates with slaves in the background. There are a few exceptions where the system can address slave devices separately, such as when reading device identifiers of slaves.

The communication protocol for all functions except firmware upgrade, shall be Modbus. Master device acts as a Modbus TCP server on Ethernet and Modbus master on RS-485. Slave devices act as Modbus slaves on RS-485 and are silent on Ethernet.

Modbus registers shall be designed in a way that they appear to be in master Fuse board, but they are actually synchronized with the registers in slaves.

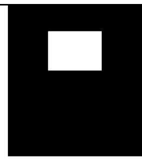
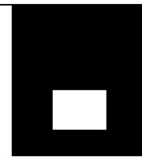
Firmware shall be the same for master and slave Fuse boards.

Todo: Add overview drawing

3. Functions

3.1. Device configuration

Primary operational mode configuration is defined with DIP switches on the PCB.
DIP switches are counted from left to right. They can be up or down (default). *Todo: add photo.*

Switch up:		Switch down:	

DIP switches functions:

Switch	**Function**		**Up**	**Down**
	Master	**Slave**		
0	Device mode		Master	Slave
1	IP mode	*None*	Static	Dynamic (DHCP)
2	Slave count	Device address	See table below	
3				
4				
5	*Reserved*			
6	*Reserved*			
7	*Reserved*			

Table explaining DIP switch 2-4 positions:

Master slave count	0	1	2	3	4	5	6	7
Slave device address	*Invalid*	1	2	3	4	5	6	7
DIP 2	Down	Up	Down	Up	Down	Up	Down	Up
DIP 3	Down	Down	Up	Up	Down	Down	Up	Up
DIP 4	Down	Down	Down	Down	Up	Up	Up	Up

Firmware reads DIP switch position once during start-up (and power-up). Therefore restart (re-powering) is required after changing DIP switches.

3.2. Modbus addressing

Modbus provides 16-bit addressing. All the addresses in the document are represented in hexadecimal format, starting with a 0x prefix from C language. Two types of addressing representations are used: device and port based.
Regardless of representation, all addresses are unique. Addresses are unique also across Modbus functions. However, only specific functions are available with addresses and that is documented under each chapter.

3.2.1. Device based address

Address is represented in format: 0xFN##.
F is hexadecimal digit, N is a device number, # is device internal address digit.

Address	Device number (N)	Device name
0xF0##	0	Master
0xF1##	1	Slave 1
0xF2##	2	Slave 2
0xF3##	3	Slave 3
0xF4##	4	Slave 4
0xF5##	5	Slave 5
0xF6##	6	Slave 6
0xF7##	7	Slave 7
0xF8##	Reserved	
…		
0xFF##		

3.2.2. Port based address

Address is represented in format: 0xPP##

PP is a port number. Valid values are 0x00 (0) to 0x3F (decimal 63). Values 0x40 (64) to 0xEF (239) are reserved.

Values 0xF0 (240) to 0xFF (255) shall not be used as they collide with device-based addresses. # is a port internal address digit.

Port numbers are directly related to device number and device port as illustrated in the table and math below.

Address	Port number (PP)	Device number (N) and name	Device port
0x00##	0	0 (master)	0
0x01##	1	0 (master)	1
	…		
0x07##	7	0 (master)	7
0x08##	8	1 (slave 1)	0
0x09##	9	1 (slave 1)	1
	…		
0x3E##	62	7 (slave 7)	6
0x3F##	63	7 (slave 7)	7
0x40##	Reserved		
…			
0xEF##			

Math is following:

Port = Device number * 8 + device port

Device port = Port % 8 *(modulus)*

Device number = Port / 8 *(integer division)*

3.3. Device identification

The following read-only identifiers apply for all Fuse board devices – master and slaves. The identifiers shall be readable as Modbus input (internal) registers with function 0x04. From a system point of view each device shall be separately addressed because each device has its own identifiers, hence device-based addressing shall be used. The first two are special and apply for master only.

Parameter	Description	Master/Slave	Address	Size (B)	Format
Number of slaves	Number of slaves (0-7)	M	0xNN00	1	U8
Number of outputs	Number of outputs	M	0xNN01	1	U8
	Reserved	M, S	0xNN02	2	
UID	STM32 unique ID (96-bits)	M, S	0xNN04	12	Binary
MAC	MAC address of Ethernet interface.	M, S	0xNN10	6	Binary
Serial number[1]	Device unique serial number.	M, S	0xNN20	16	ASCII, zero delimited
Hardware revision[1]	To identify the hardware on which the software is running at.	M, S	0xNN30	1	U8
Bootloader version	Currently loaded bootloader version.	M, S	0xNN40	16	ASCII, zero delimited
Firmware version	Currently loaded application firmware version.	M, S	0xNN50	16	ASCII, zero delimited
	Reserved	M, S	0xNN60	16	

Reminder: N is a device number. 0 is master, 1-7 are slaves.

[1] STM32 internal one-time-programmable (OTP) memory shall be used to store serial number and hardware revision. OTP memory is not erasable or re-programmable.

Explanation:

The system shall first ask the master how many slaves there are. Then it can use determined addresses to ask identification parameters of all the slaves.

3.4. Firmware upgrade

A/B boot shall be used. New firmware image shall be downloaded while it continues to operate on the existing one. MCU that is in use (STM32H5) has dual-bank Flash which allows to do it. A reset is required after downloading to make the new firmware operational. When master firmware has validated and upgraded, the master will automatically upgrade slaves with the same firmware.

Firmware download protocol is TBD. Modbus has 16-bits address limit and it's pointless to send same image over same RS-485 from master to all slaves N times. Some type of broadcasting protocol should be used on RS-485 instead. It is very likely a custom protocol.

3.5. Configuration functions

The following chapters explain the software controllable configuration options. Configuration options shall be read and written like Modbus holding registers. Function 0x03 is to read single holding register, function 0x06 is to write single holding register.

3.5.1. Master device configuration options

The following configuration options apply for master. Using device-based addressing.

Parameter	Description	Address	Size (B)	Format
IP address (manual)	IP v4 address. Effective only when DIP switch is configured to use manual IP.	0xNN80	4	Four bytes that create IP: First.second.third.fourth

3.5.2. Output based configuration options

The following configuration options apply for master and slaves, but they all go through the master. Port-based shall be used.

Parameter	Description	Address	Size (B)	Format	Unit
Normal state	Relay normal state: NO (normally open) or NC (normally closed). This is determined by hardware.	0xPP80	1	0 – NO 1 – NC	
Start-up state	Output off / On. Applied with a delay after firmware start-up. *Note: electrically outputs are off (NO) when Fuse board is unpowered, or firmware has not yet started.*	0xPP81	1	0 - Off 1 - On	
Start-up delay	Time in milliseconds after start-up when to change output state to its start-up state.	0xPP82	2	U16 big-endian	ms
Forward current limit	0 to +15A limit for each output.	0xPP84	2	U16 big-endian	mA
Reverse current limit	0 to -15A limit for each output. Limit is expressed in positive numbers.	0xPP86	2	U16 big-endian	mA

Activation filter time	Current limit ignoring time (milliseconds) when output is turned on. To filter out spikes caused by in-rush current.	0xPP88	2	U16 big-endian	ms
Operation filter time	Current limit ignoring time (milliseconds) when output is already on. To filter out spikes that occur during operation.	0xPP90	2	U16 big-endian	ms
Recovery mode	Behavior after the current limit is exceeded and output is turned off for device protection. • No recovery, stay off. Off-on sequence or reset command required to turn on again. • Try to automatically turn it on up to 10 times. Apply delay (recovery off time) between retries. After retries, stay off. Controlled off-on or reset required. *Note: must have a limit on retries to increase relay life.*	0xPP92	1	0 – No recovery 1 – Retry once 2 – Retry twice … 10 – Retry 10 times	
	Reserved	0xPP93	1		
Recovery off time	Time to wait in milliseconds before trying to turn the output on again.	0xPP94	2	U16 big-endian	ms
Reset off time	Time to wait in milliseconds in off state when doing output automatic reset.	0xPP96	2	U16 big-endian	ms

3.6. Operation functions

Operation shall be performed by reading and writing coil registers. Modbus read coil function is 0x01 and write coil function is 0x05.

3.6.1. Device (master, slave) operation functions

Function	R/W	Description	Address	Size	Format
Restart	W	Restart the Fuse board. If firmware has been upgraded meanwhile, it will become effective.	0xNN70	1	0x55 – Restart addressed device. 0x66 – Restart all devices (master only).
Config change	W	To control all configuration parameters (holding registers) at once.	0xNN71	1	0xAA – Reset configuration (all zeroes). 0xBB – Set current configuration as factory default. 0xCC – Restore factory default configuration.
Detect relay normal state	W	Automatically detect relay normal state. No load shall be connected to outputs and fuses must be operational. After detection normal state configuration parameters are automatically updated.	0xNN72	1	0x77 – Run automatic detection.
		Reserved	0xNN73	1	
Uptime	R	Get uptime of firmware. Time in seconds since last start-up.	0xNN74	4	U32 big-endian. Unit: seconds.
		Reserved	0xNN78	8	

3.6.2. Output based operations

These operations are performed through the master. When an output is addressed that is on the slave, the master forwards the operation to the slave.

Function	R/W	Description	Address	Size	Format
Output control	RW	Output control request. Allows to set static state or make an automatic off-on sequence (useful when system asks for self-reset).	0xPP00	1	0 – Turn off 1 – Turn on 2 – Off-on sequence
Output state	R	Returns output actual state. Return state 3 (output on, no voltage) could mean that fuse is burned. Return state 2 indicates some wiring problem.	0xPP01	1	0 – Output is off 1 – Output is on 2 – Output is off, excessive voltage detected. 3 – Output is on, no voltage detected.
Output state reason	R	Returns last reason for output being in the state it is.	0xPP02	1	0 - Starting-up 1 - Start-up default 2 - Control request 3 - Button press 4 - Reset request 5 - Protection (e.g. current limiter) 6 - Recovery (after protection)
		Reserved	0xPP03	1	
Output fault flags	RW	Returns faults that have occurred since device power-up or last fault flags clearing. Fault(s) can be cleared by writing the specific bitmask(s). Fault flags cannot be set.	0xPP04	1	Bitmasks: 1 – Forward current exceeded 2 – Reverse current exceeded 4 – Excessive voltage detected 8 – Absence of voltage detected

Read output voltage	R	Returns output voltage (V).	0xPP10	2	U16 big-endian Factor: x1, Unit: mV
Read output current	R	Return output current (A).	0xPP12	2	S16 big-endian Factor: x10, Unit: mA
Read output power	R	Returns output power (W).	0xPP14	2	S16 big-endian. Factor: x100, Unit: mW
Read output energy	R	Return output energy (J) since turning it on.	0xPP16	4	U32 big-endian Factor: x1, Unit: J
Read switch	R	Read push-button switch state.	0xPP20	1	0 – Not available 1 – Not pressed 2 – Pressed
Output LED control	W	Temporary LED control. Automatically restores LED normal operation after given time.	0xPP21	3	Byte 0-1: RGB565 color. Byte 2: Time in x0.1s.

3.7. Troubleshooting functions

Ideas:

- Have a telnet port to get devices human readable log output.
    - Master needs to combine its own and slave's logs.
- Store errors and critical faults in device non-volatile memory and have a (Modbus) interface to read them out.

3.8. State indication

Each device has two main RGB LEDs and 8 port specific RGB LEDs.

Power indicator (left LED):

State	LEDs
Device off	Stable black
Device on, power good	Stable green
Device on, low input voltage	Stable yellow

Master connection indicator (right LED):

State	LEDs
Device off	Stable black
Ethernet disconnected	Stable red
Ethernet link up, acquiring IP address	Blinking yellow
Ethernet link up, got IP	Stable blue
Modbus TCP/IP connection established	Stable green

Slave connection indicator (right LED):

State	LEDs
Device off	Stable black
Invalid slave address (0)	Stable red
No communication	Blinking yellow
Modbus RS-485 communication active	Stable green

Device ports indicators (LED 1 to 8):

State	LEDs
Output off	Stable black

Output on, has voltage	Stable green
Output on, no voltage (fuse burned)	Blinking red
Output on, overcurrent	Stable yellow