Libpldm API Documentation

Libpldm is a library supporting the encoding, decoding, packing, and unpacking of PLDM Type messages and their respective commands. Since there are multiple PLDM Types supported with similar API's this document will list the files and functions of the SetStateEffecterStates command of the Platform Monitoring and Control Type.

Repository: https://github.com/rios240/libpldm-cerberus

test/libpldm_cerberus_test.cpp

TEST(PlatformMonitoringControl, testSetStateEffecterStates)
Initializes command specific field variables and declares a pldm_msg struct called 'request' that points to a std::array buffer called 'requestMsg'. The test then passes the field variables and the request pointer into **platform.h/encode_set_state_effecter_states_req**. After encoding the 'request' is placed in the buffer and sent to a socket via **socket_connect.h/socket_send_pldm_message**.

base.h

Defines enumerations, structs, macros, and functions for PLDM Messaging Control and Discovery commands. The pldm_msg_header, pldm_msg, and pldm_header_info structs and the pack_pldm_header() function are used by lipldm_cerberus_test.cpp and encode functions to construct the PLDM header and message body.

pack_pldm_header

```
int pack_pldm_header(const struct pldm_header_info *hdr,
                     struct pldm_msg_hdr *msg);
```

Parameters

Name	Type	Description
hdr	pldm_header_info *	Pointer to the PLDM header information
msg	pldm_msg_hdr *	Pointer to PLDM message header

Returns

Value	Description
[0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x20, 0x21]	0 on success, otherwise PLDM error codes.

strcuts, enums, and macros:

```
enum pldm_completion_codes {
        PLDM_SUCCESS = 0x00,
        PLDM_ERROR = 0x01,
        PLDM_ERROR_INVALID_DATA = 0x02,
        PLDM_ERROR_INVALID_LENGTH = 0x03,
        PLDM_ERROR_NOT_READY = 0x04,
        PLDM_ERROR_UNSUPPORTED_PLDM_CMD = 0x05,
        PLDM_ERROR_INVALID_PLDM_TYPE = 0x20,
        PLDM_INVALID_TRANSFER_OPERATION_FLAG = 0x21
}
```

```
struct pldm_msg_hdr {
#if defined(__LITTLE_ENDIAN_BITFIELD)
        uint8_t instance_id : 5;  //!< Instance ID
        uint8_t reserved : 1;     //!< Reserved
        uint8_t datagram : 1;     //!< Datagram bit
        uint8_t request : 1;      //!< Request bit
#elif defined(__BIG_ENDIAN_BITFIELD)
        uint8_t request : 1;      //!< Request bit
        uint8_t datagram : 1;     //!< Datagram bit
        uint8_t reserved : 1;     //!< Reserved
        uint8_t instance_id : 5; //!< Instance ID
#endif

#if defined(__LITTLE_ENDIAN_BITFIELD)
        uint8_t type : 6;         //!< PLDM type
        uint8_t header_ver : 2; //!< Header version
#elif defined(__BIG_ENDIAN_BITFIELD)
        uint8_t header_ver : 2;  //!< Header version
        uint8_t type : 6;         //!< PLDM type
#endif
        uint8_t command;          //!< PLDM command code
} __attribute__((packed));
```

```
struct pldm_msg {
        struct pldm_msg_hdr hdr;   //!< PLDM message header
        uint8_t payload[1];        //!< &payload[0] is the beginning of the payload
} __attribute__((packed));
```

```
struct pldm_header_info {
        MessageType msg_type;     //!< PLDM message type
        uint8_t instance;         //!< PLDM instance id
        uint8_t pldm_type;        //!< PLDM type
        uint8_t command;          //!< PLDM command code
        uint8_t completion_code;  //!< PLDM completion code, applies for response
};
```

platform.h

Defines enumerations, structs, macros, and functions for PLDM Platform Monitoring and Control commands. Contains structs to represent each pldm command fields.

encode_set_state_effecter_states_req

```
int encode_set_state_effecter_states_req(uint8_t instance_id,
                                         uint16_t effecter_id,
                                         uint8_t comp_effecter_count,
                                         set_effecter_state_field *field,
                                         struct pldm_msg *msg);
```

Parameters

Name	Type	Description
instance_id	unit8_t	Message's instance id
effecter_id	unit16_t_	used to identify and access the effecter
comp_effecter_count	uint8_t	number of individual sets of effecter information. Up to eight sets of state effecter info can be accessed for a given effecter.
field	set_effecter_state_field *	Each unit is an instance of the stateField structure that is used to set the requested state for a particular effecter within the state effecter. This field holds the starting address of the stateField values. The user is responsible for allocating the memory prior to calling this command. The user has to allocate the field parameter as sizeof(set_effecter_state_field) comp_effecter_count.
msg	struct pldm_msg *	Message will be written to this

Returns

Value	Description
[0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x20, 0x21]	pldm_completion_codes

strcuts, enums, and macros:

```
struct pldm_set_state_effecter_states_req {
        uint16_t effecter_id;
        uint8_t comp_effecter_count;
        set_effecter_state_field field[8];
} __attribute__((packed));
```

socket_connect.h

Defines an interface to connect with a python socket. The following functions are used by **libpldm_cerberus_test.cpp**. The test initializes a connection the socket and sends the PLDM message via a buffer.

initialize_socket_connection

int initialize_socket_connection();

Returns

Value	Description
[0, 1]	0 on success, 1 on failure.

socket_send_pldm_message

int socket_send_pldm_message(const uint8_t* data, size_t data_length);

Parameters

Name	Type	Description
data	const unit8_t *	A pointer to a buffer containing a PLDM message.
data_length	size_t	The size in bytes of the buffer.

Returns

Value	Description
[0, 1]	0 on success, 1 on failure.

close_socket_connection

int close_socket_connection();

Returns

Value	Description
[0, 1]	0 on success, 1 on failure.

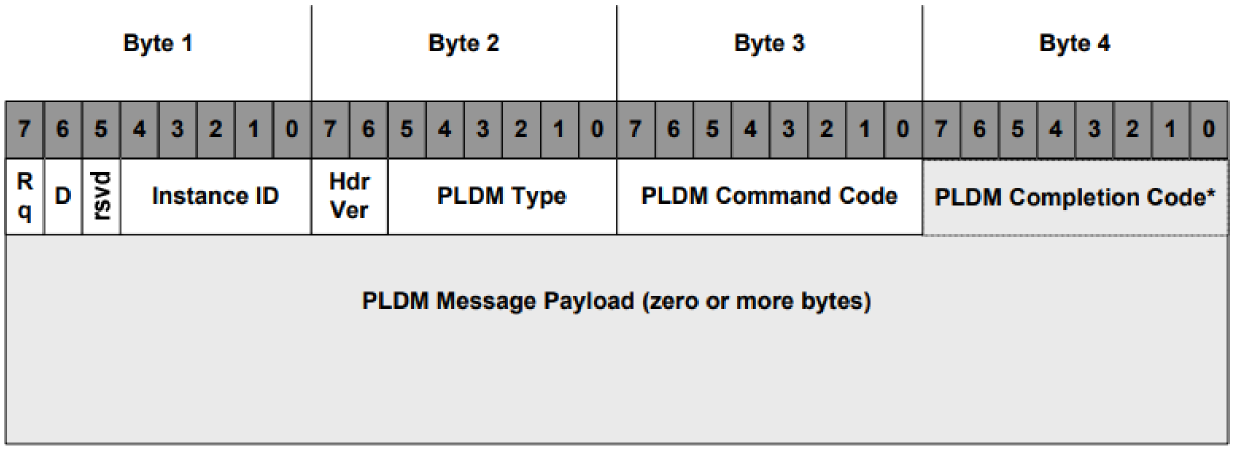
Platform Monitoring and Control SetStateEffecterStates Command Example

100000000000001000111001000010100000000000000010000000010000001000000001000000
110000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000

Above is the byte array of the SetStateEffecterStates command for Platform Monitoring and Control. The message is 22 bytes long.

1	0	0	0	0	0	0	0		0	0	0	0	0	0	1	0		0	0	1	1	1	0	0	1		0	0	0	0	0	0	0	0
00001010 00000000 00000010 00000001 00000010 00000001 00000011 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000																																		

Here is the same message but in a more readable format so that individual bit fields can be verified against the following schema:


Byte 1
Byte 2
Byte 3
Byte 4
7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0
Rq
D
rsvd
Instance ID
Hdr Ver
PLDM Type
PLDM Command Code
PLDM Completion Code*
PLDM Message Payload (zero or more bytes)