Processing of subpackets in a SimpleMotion V2 slave device
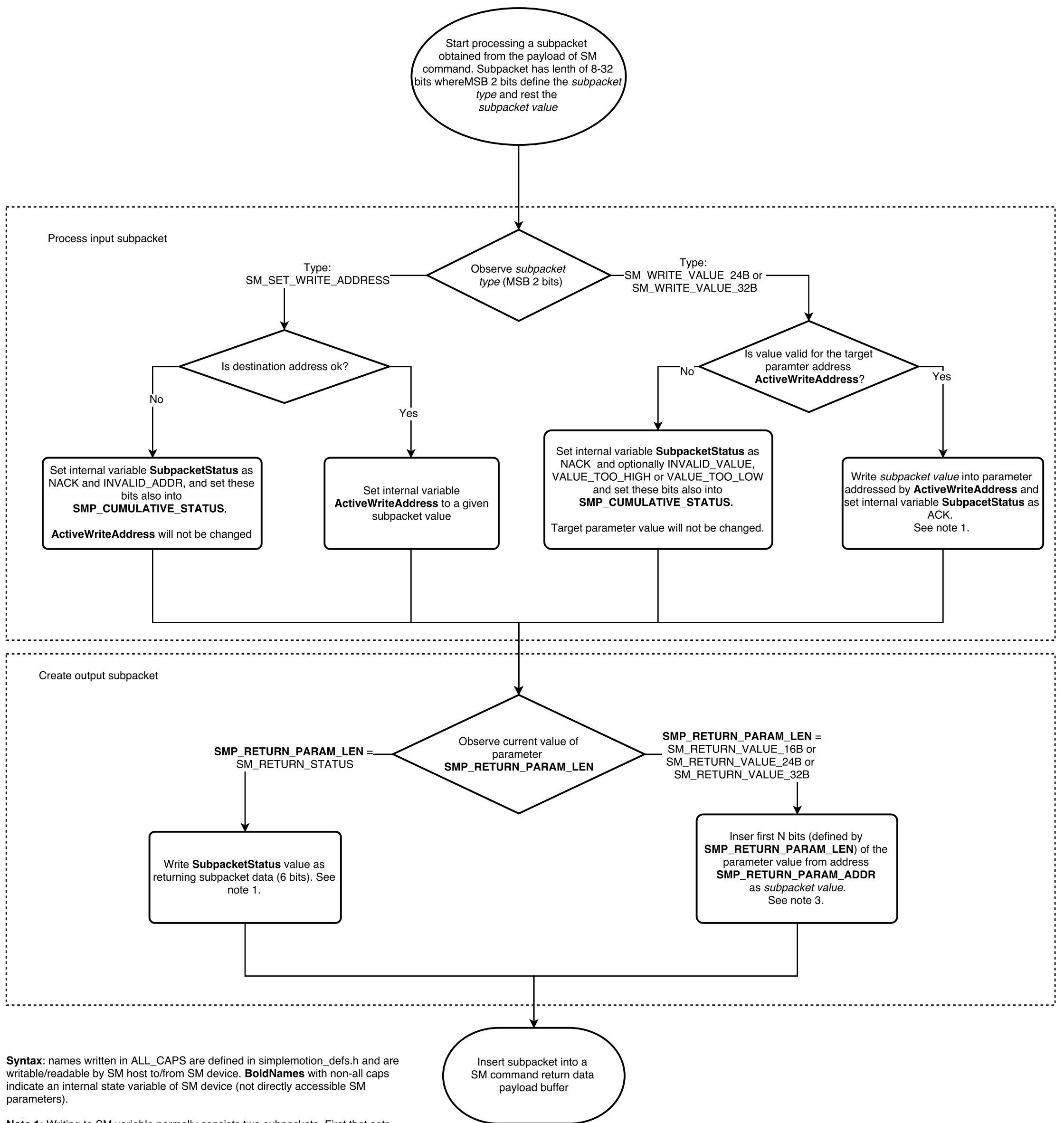
For introduction to SimpleMotion protocol and subpackets, see https://github.com/GraniteDevices/SimpleMotionV2/blob/master/doc/SMV2Protocol.png.

Start processing a subpacket obtained from the payload of SM command. Subpacket has lenth of 8-32 bits whereMSB 2 bits define the *subpacket type* and rest the *subpacket value*

Process input subpacket

Observe *subpacket type* (MSB 2 bits)

Type: SM_SET_WRITE_ADDRESS

Type: SM_WRITE_VALUE_24B or SM_WRITE_VALUE_32B

Is destination address ok?

Is value valid for the target paramter address **ActiveWriteAddress**?

No → Set internal variable **SubpacketStatus** as NACK and INVALID_ADDR, and set these bits also into **SMP_CUMULATIVE_STATUS**.

ActiveWriteAddress will not be changed

Yes → Set internal variable **ActiveWriteAddress** to a given subpacket value

No → Set internal variable **SubpacketStatus** as NACK and optionally INVALID_VALUE, VALUE_TOO_HIGH or VALUE_TOO_LOW and set these bits also into **SMP_CUMULATIVE_STATUS**.

Target parameter value will not be changed.

Yes → Write *subpacket value* into parameter addressed by **ActiveWriteAddress** and set internal variable **SubpacketStatus** as ACK. See note 1.

Create output subpacket

Observe current value of parameter **SMP_RETURN_PARAM_LEN**

SMP_RETURN_PARAM_LEN = SM_RETURN_STATUS

SMP_RETURN_PARAM_LEN = SM_RETURN_VALUE_16B or SM_RETURN_VALUE_24B or SM_RETURN_VALUE_32B

Write **SubpacketStatus** value as returning subpacket data (6 bits). See note 1.

Inser first N bits (defined by **SMP_RETURN_PARAM_LEN**) of the parameter value from address **SMP_RETURN_PARAM_ADDR** as *subpacket value*. See note 3.

Insert subpacket into a SM command return data payload buffer

Syntax: names written in ALL_CAPS are defined in simplemotion_defs.h and are writable/readable by SM host to/from SM device. **BoldNames** with non-all caps indicate an internal state variable of SM device (not directly accessible SM parameters).

Note 1: Writing to SM variable normally consists two subpackets. First that sets **ActiveWriteAddress** and second that writes the value to that address.

Optimization trick: any number of consequent writes to the same address can be done after setting **ActiveWriteAddress**. I.e. set **ActiveWriteAddress**=553 (SMP_ABSOLUTE_SETPOINT) and after that just write setpoints as needed.

Note 2: Unless SMP_RETURN_PARAM_LEN is set to SM_RETURN_STATUS, then the immediate value of SubpacketStatus is lost. However it's active bits will be always written into exposed internal vaiable **SMP_CUMULATIVE_STATUS** that may be read at any moment to check errors after several subpacket operations**.**

Note 3: in order to read any variable from SM device, host must first write an address value into SMP_RETURN_PARAM _ADDR and SMP_RETURN_PARAM_LEN.

Diagram version 1.0.0, 27.2.2018