Programming Guide
GT-521F Series

Optical Fingerprint Recognition ***EMBEDDED*** *Module*

Version 1.2

Jun 10th, 2019

www.adh-tech.com.tw
sales@adh-tech.com.tw

Table of Contents

Revision History

Version	Data	Description
V1.0	Jun 19, 2017	Created
V1.1	Sep 14,2017	Add anti-proof function
V1.2	Jun 5th,2019	Correct unused commands

1. General Description

1.1. About this document

The document is created to facilitate the firmware porting for the embedded system.

1.2. Related documents

You can find relative datasheets, include:

GT-521F datasheet

GT-521F programming guide

GT-521F SDK

2. Protocol: Packet Structure

(Multi-byte item is represented as Little Endian.)

Command Packet (Command)

OFFSET	ITEM	TYPE	DESCRIPTION
0	0x55	BYTE	Command start code1
1	0xAA	BYTE	Command start code2
2	*Device ID*	WORD	Device ID: default is 0x0001, always fixed
4	*Parameter*	DWORD	Input parameter
8	*Command*	WORD	Command code
10	*Check Sum*	WORD	Check Sum (byte addition) OFFSET[0]+…+OFFSET[9]=*Check Sum*

Response Packet (Acknowledge)

OFFSET	ITEM	TYPE	DESCRIPTION
0	0x55	BYTE	Response start code1
1	0xAA	BYTE	Response start code2
2	*Device ID*	WORD	Device ID: default is 0x0001, always fixed
4	*Parameter*	DWORD	**Response == 0x30:** (ACK) Output Parameter **Response == 0x31:** (NACK) Error code
8	*Response*	WORD	**0x30:** Acknowledge (ACK). **0x31:** Non-acknowledge (NACK).
10	*Check Sum*	WORD	Check Sum (byte addition) OFFSET[0]+…+OFFSET[9]=*Check Sum*

Data Packet (Data)

OFFSET	ITEM	TYPE	DESCRIPTION
0	0x5A	BYTE	Data start code1
1	0xA5	BYTE	Data start code2
2	*Device ID*	WORD	Device ID: default is 0x0001, always fixed
4	*Data*	N BYTES	N bytes Data The size is pre-defined per protocol stage
4+N	*Check Sum*	WORD	Check Sum (byte addition) OFFSET[0]+…+OFFSET[4+N-1]=*Check Sum*

3. Protocol: Commands Summary

In a command packet *Command* can be one of below.

Number (HEX)	Alias	Description
01	*Open*	Initialization
02	*Close*	Termination
04	*ChangeBaudrate*	Change UART baud rate
12	*CmosLed*	Control CMOS LED
20	*GetEnrollCount*	Get enrolled fingerprint count
21	*CheckEnrolled*	Check whether the specified ID is already enrolled
22	*EnrollStart*	Start an enrollment
23	*Enroll1*	Make 1st template for an enrollment
24	*Enroll2*	Make 2nd template for an enrollment
25	*Enroll3*	Make 3rd template for an enrollment, merge three templates into one template, save merged template to the database
26	*IsPressFinger*	Check if a finger is placed on the sensor
40	*DeleteID*	Delete the fingerprint with the specified ID
41	*DeleteAll*	Delete all fingerprints from the database
50	*Verify*	1:1 Verification of the capture fingerprint image with the specified ID
51	*Identify*	1:N Identification of the capture fingerprint image with the database
60	*CaptureFinger*	Capture a fingerprint image(256x256) from the sensor
62	*GetImage*	Download the captured fingerprint image(Width x High =202x258)
63	*GetRawImage*	Capture & Download raw fingerprint image(Width x High=160x120)
70	*GetTemplate*	Download the template of the specified ID
71	*SetTemplate*	Upload the template of the specified ID
F0	*SetSecurityLevel*	Set Security Level
F1	*GetSecurityLevel*	Get Security Level
F9	*EnterStandbyMode*	Enter Standby Mode (Low power mode)
30	*Ack*	Acknowledge.
31	*Nack*	Non-acknowledge.

4. Protocol: Error Codes

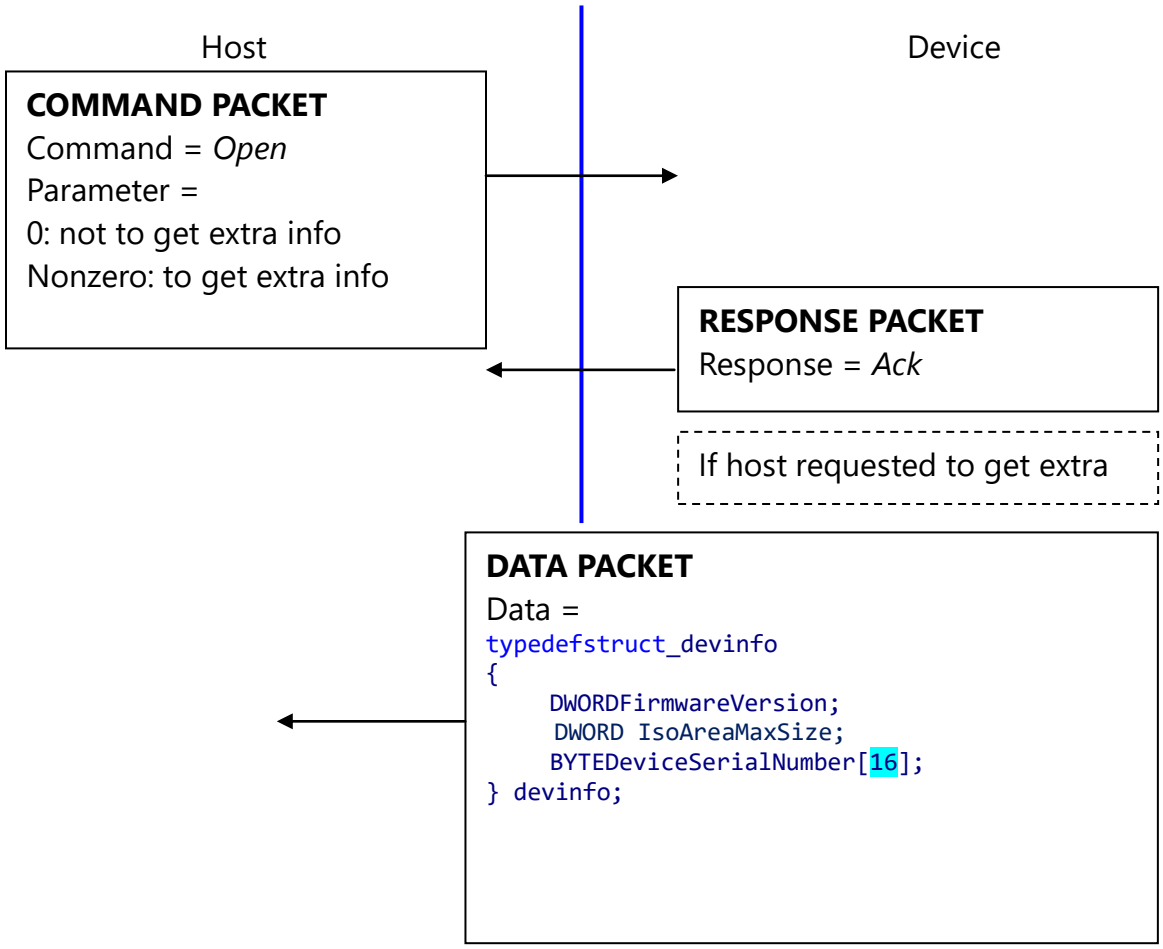
When response packet is Non-acknowledge, *Parameter* represents an error code as below.

NACK Parameter	Value	Description
NACK_TIMEOUT	0x1001	**Obsolete**, capture timeout
NACK_INVALID_BAUDRATE	0x1002	**Obsolete**, Invalid serial baud rate
NACK_INVALID_POS	0x1003	The specified ID is not between 0~Max
NACK_IS_NOT_USED	0x1004	The specified ID is not used
NACK_IS_ALREADY_USED	0x1005	The specified ID is already used
NACK_COMM_ERR	0x1006	Communication Error
NACK_VERIFY_FAILED	0x1007	1:1 Verification Failure
NACK_IDENTIFY_FAILED	0x1008	1:N Identification Failure
NACK_DB_IS_FULL	0x1009	The database is full
NACK_DB_IS_EMPTY	0x100A	The database is empty
NACK_TURN_ERR	0x100B	**Obsolete**, Invalid order of the enrollment (The order was not as: **EnrollStart -> Enroll1 -> Enroll2 -> Enroll3**)
NACK_ENROLL_FAILED	0x100D	Enrollment Failure
NACK_IS_NOT_SUPPORTED	0x100E	The specified command is not supported
NACK_DEV_ERR	0x100F	Device Error, especially if Crypto-Chip is trouble
NACK_CAPTURE_CANCELED	0x1010	**Obsolete**, The capturing is canceled
NACK_INVALID_PARAM	0x1011	Invalid parameter
NACK_FINGER_IS_NOT_PRESSED	0x1012	Finger is not pressed
NACK_RAM_ERROR	0x1013	Memory setting fail
NACK_TEMPLATE_CAPACITY_FULL	0x1014	Template capacity is full

NACK_COMMAND_NO_SUPPORT	0x1015	Function no support
Duplicated ID	0 – Max	There is duplicated fingerprint (while enrollment or setting template), This error describes just duplicated ID

5. Protocol: Command Details

5.1. Initialization(*Open*)

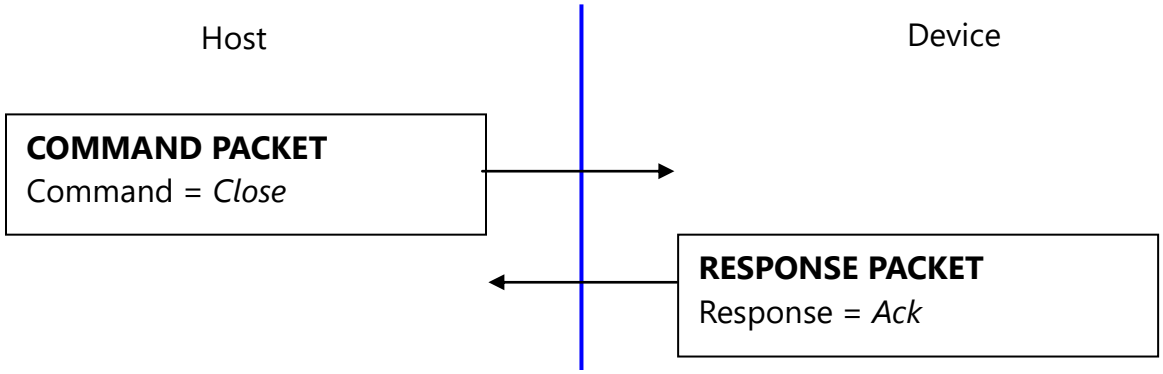


Open command is used to initialize the device; especially it gets device's static info.

NOTE:

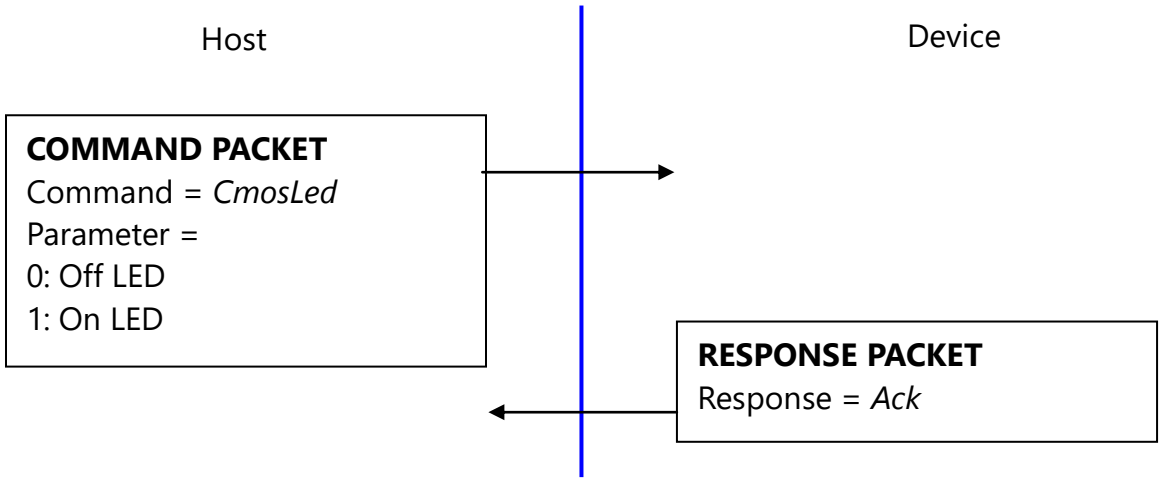
After the finger module is powered on, Must be wait 100ms and then send the *OPEN* command.

5.2. Termination(*Close*)



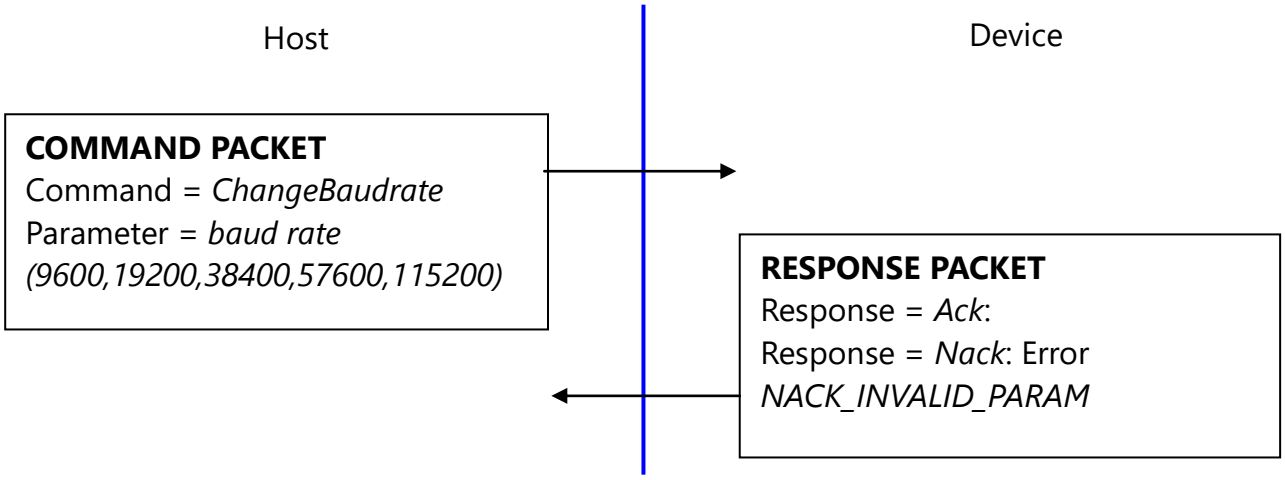
Close command does nothing.

5.3. CMOS LED control(CmosLed)



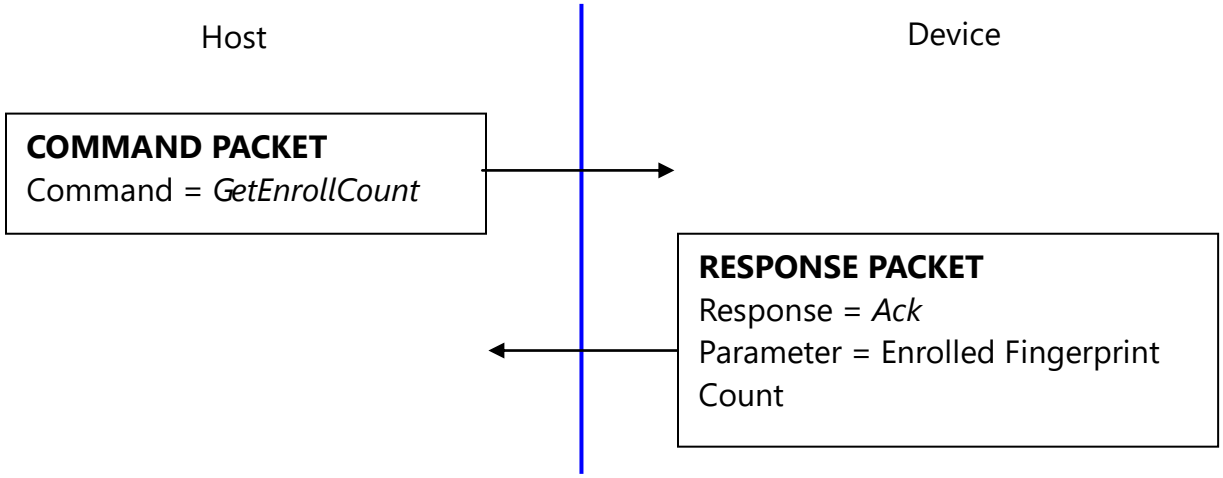
When user want to enroll or identify, must send the CmosLed command first.

5.4. Changing UART baud rate (*ChangeBaudrate*)

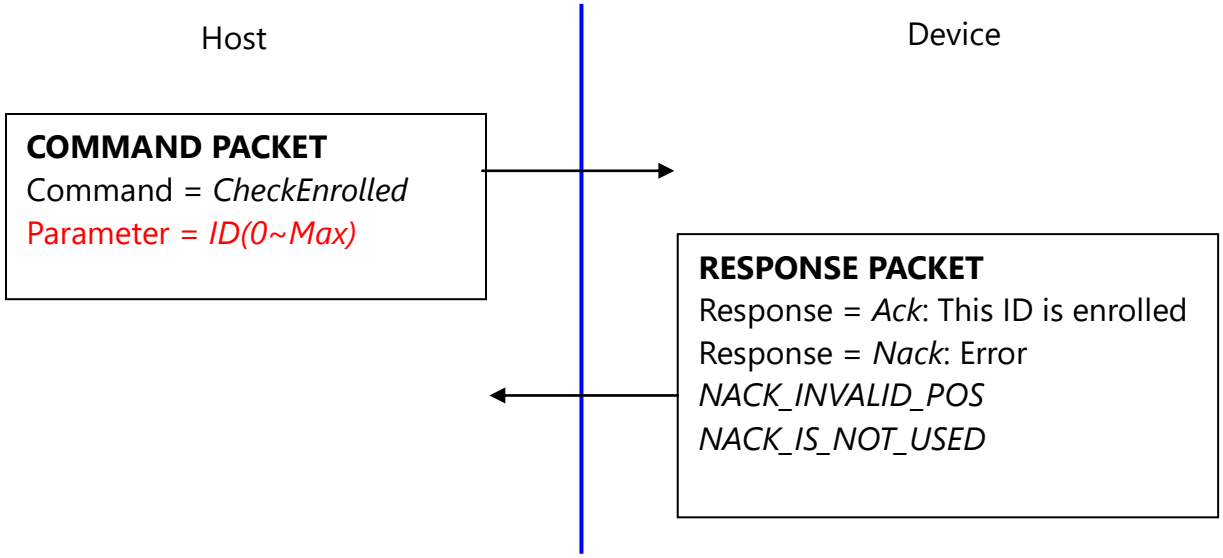


This command changes the UART baud rate at the run-time.
The device initializes its UART baud rate to 9600 bps after power on.

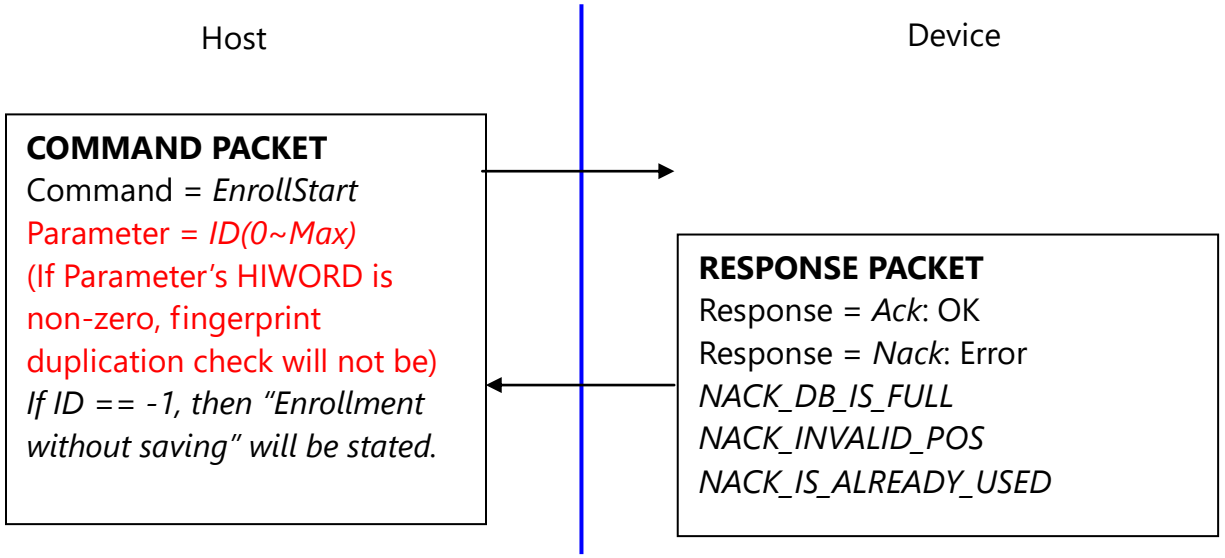
5.5. Get enrolled fingerprint count(*GetEnrollCount*)



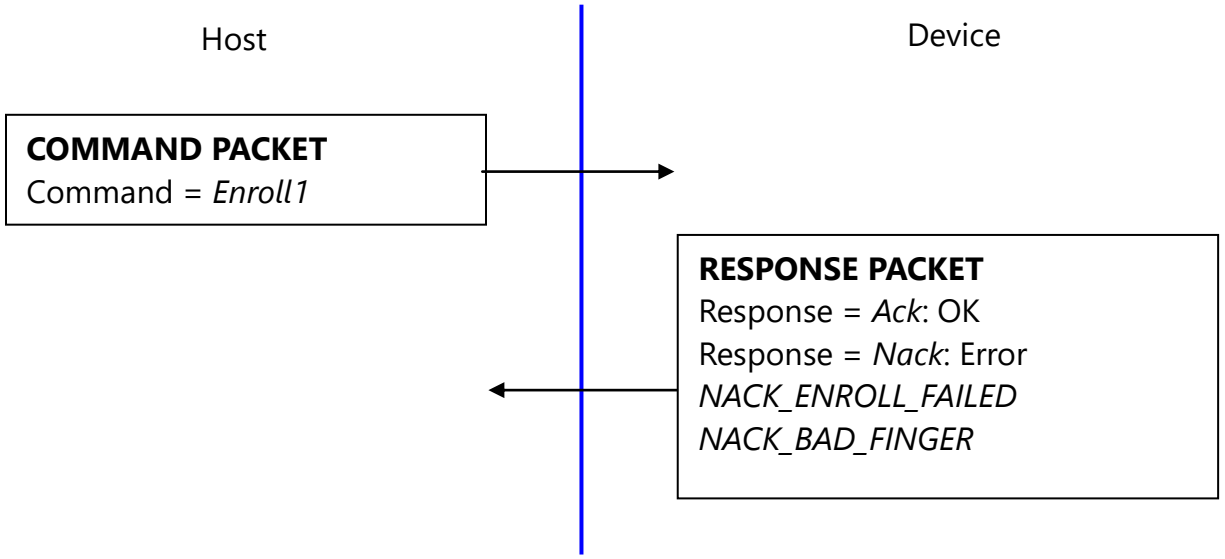
5.6. Check enrollment status(*CheckEnrolled*)



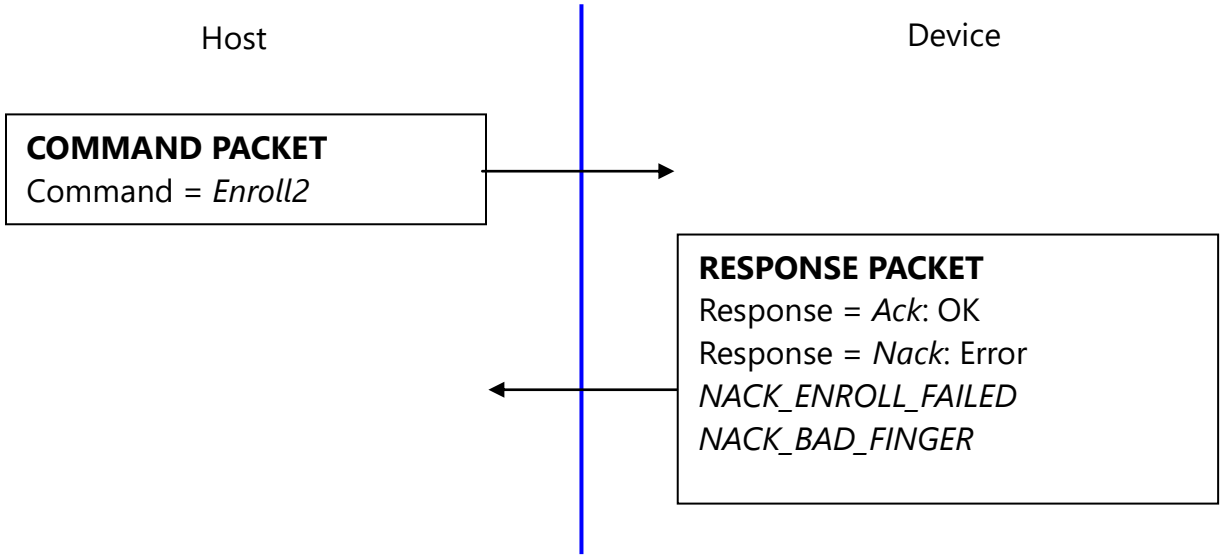
5.7. Start an enrollment(*EnrollStart*)



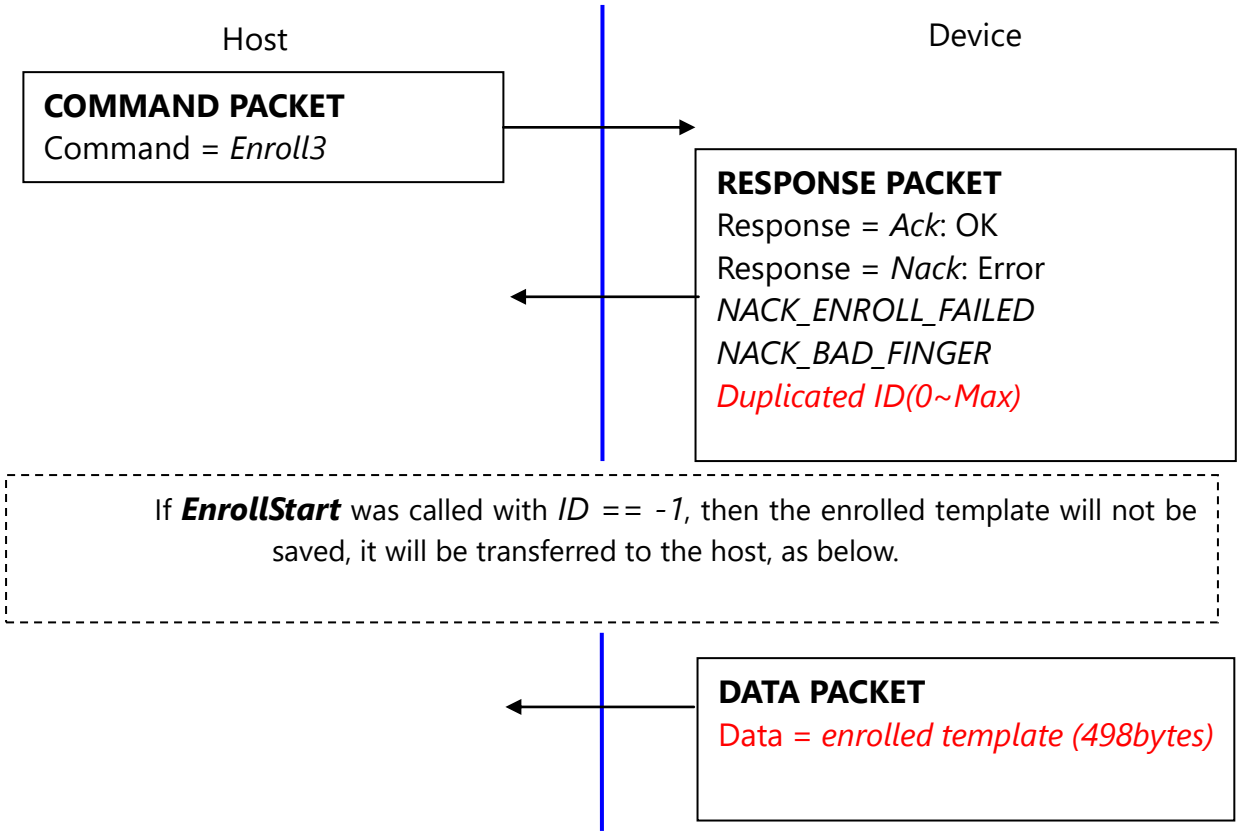
5.8. Make 1st template for an enrollment(*Enroll1*)



5.9. Make 2nd template for an enrollment(*Enroll2*)

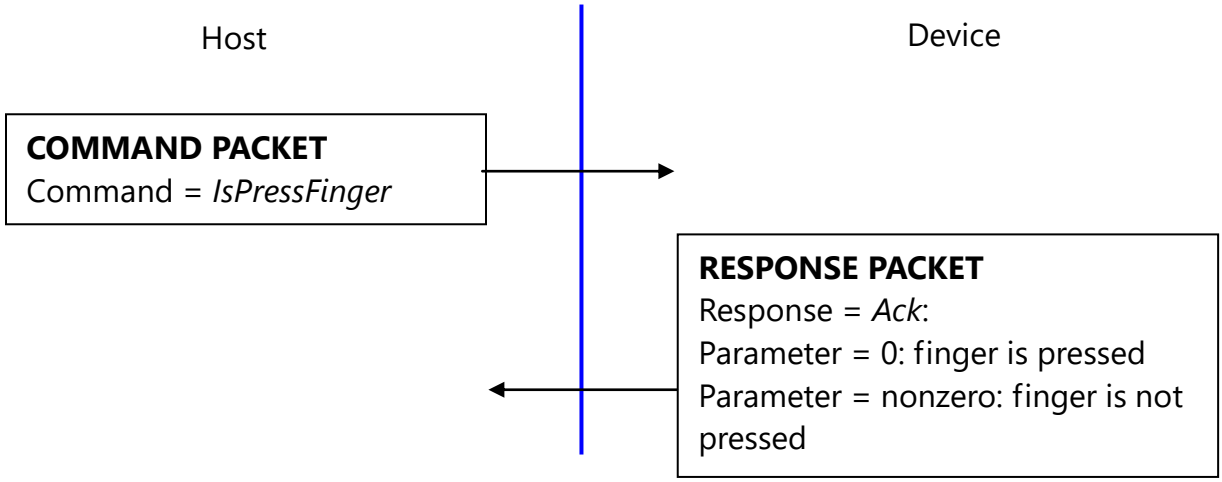


5.10. Make 3rd template for an enrollment, merge three templates(*Enroll3*)



To enroll a fingerprint, the host must issue above 4 commands, later chapter describes how to organize these commands.

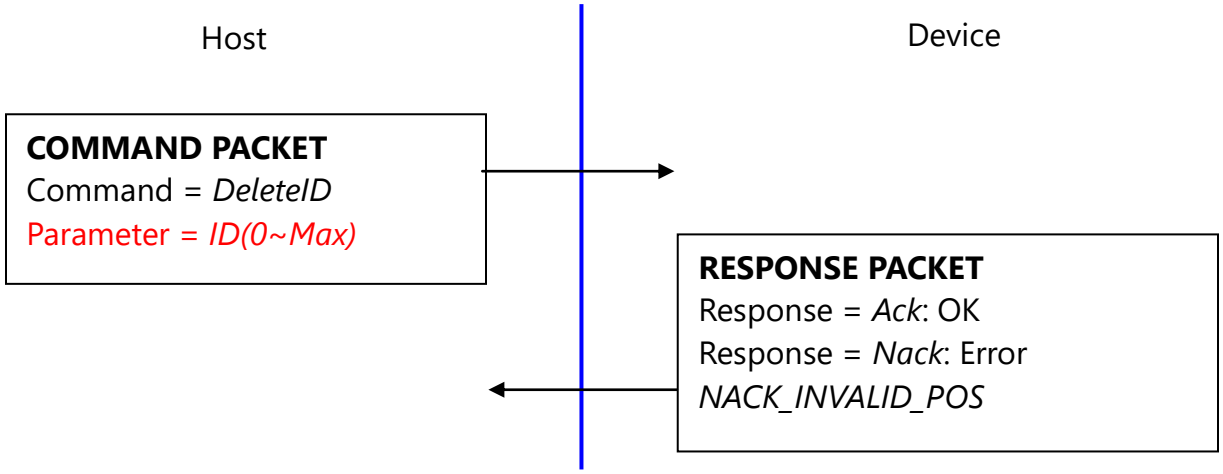
5.11. Check finger pressing status(*IsPressFinger*)



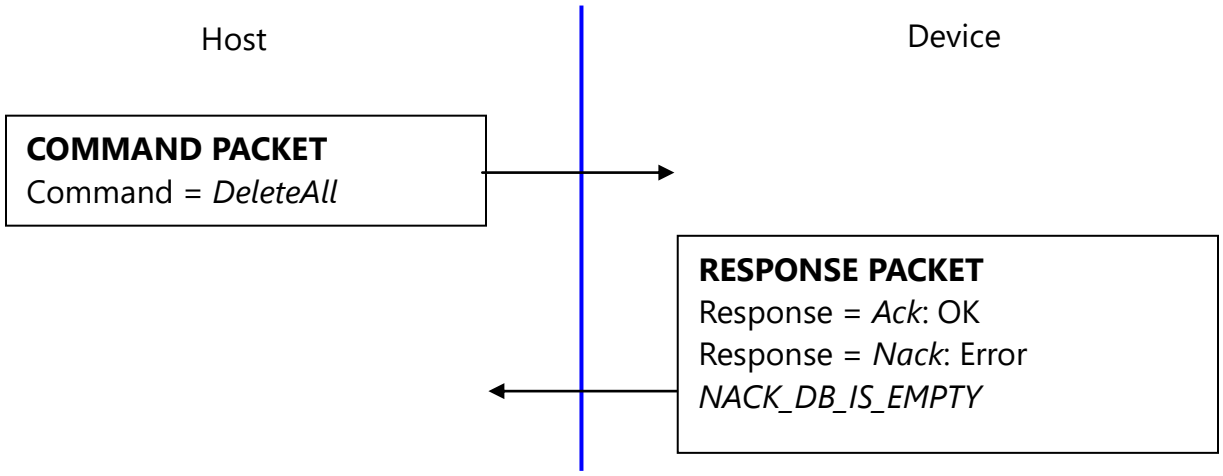
This command is used while enrollment, the host waits to take off the finger per enrollment stage.

NOTE:
Check the take off finger state can use the IsPressFinger command or the touch signal. When the IsPressFinger command return parameter is nonzero or the touch signal is low, the finger is taking off the fingerprint sensor.

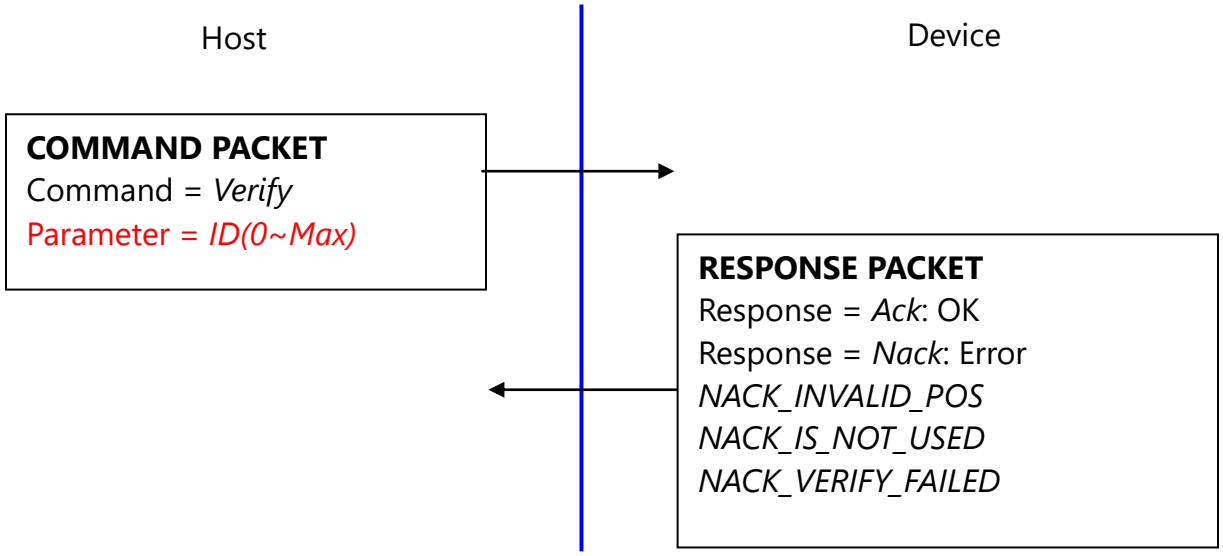
5.12. Delete one fingerprint(*DeleteID*)



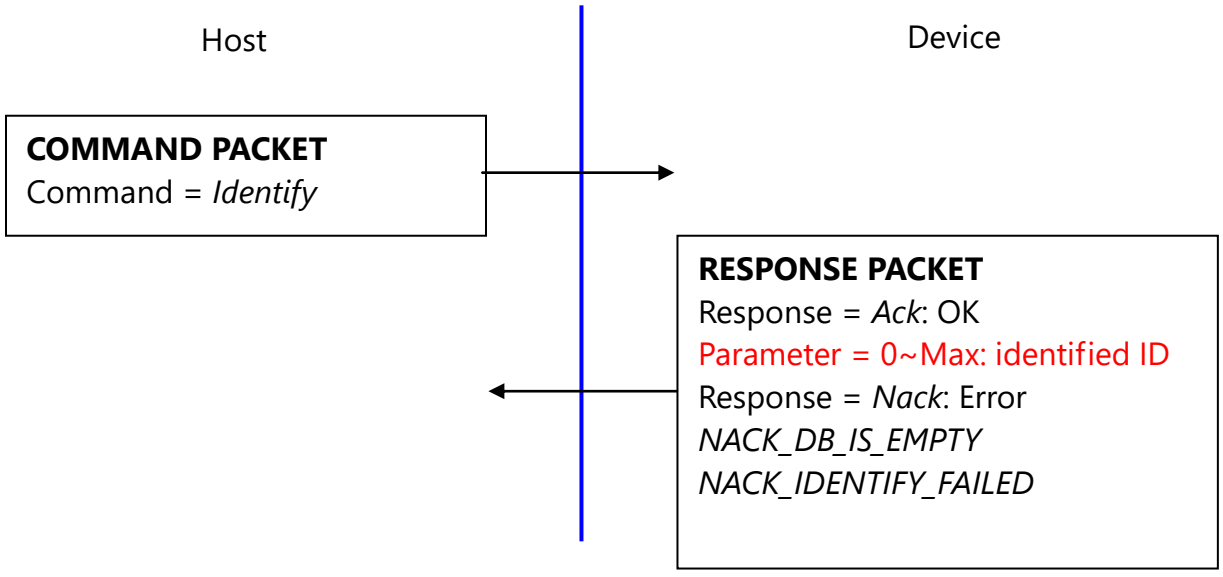
5.13. Delete all fingerprints(*DeleteAll*)



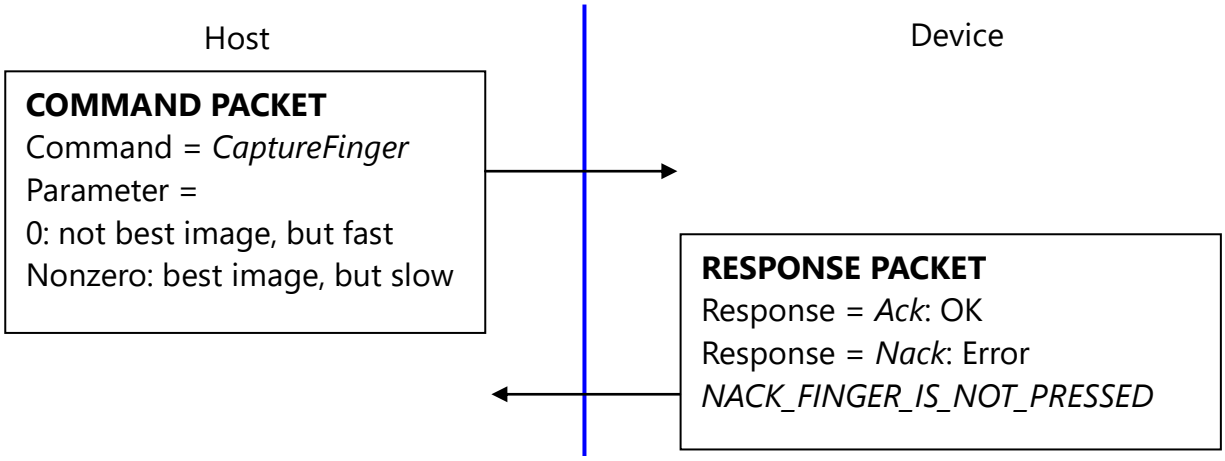
5.14. 1:1 Verification(*Verify*)



5.15. 1:N Identification(*Identify*)



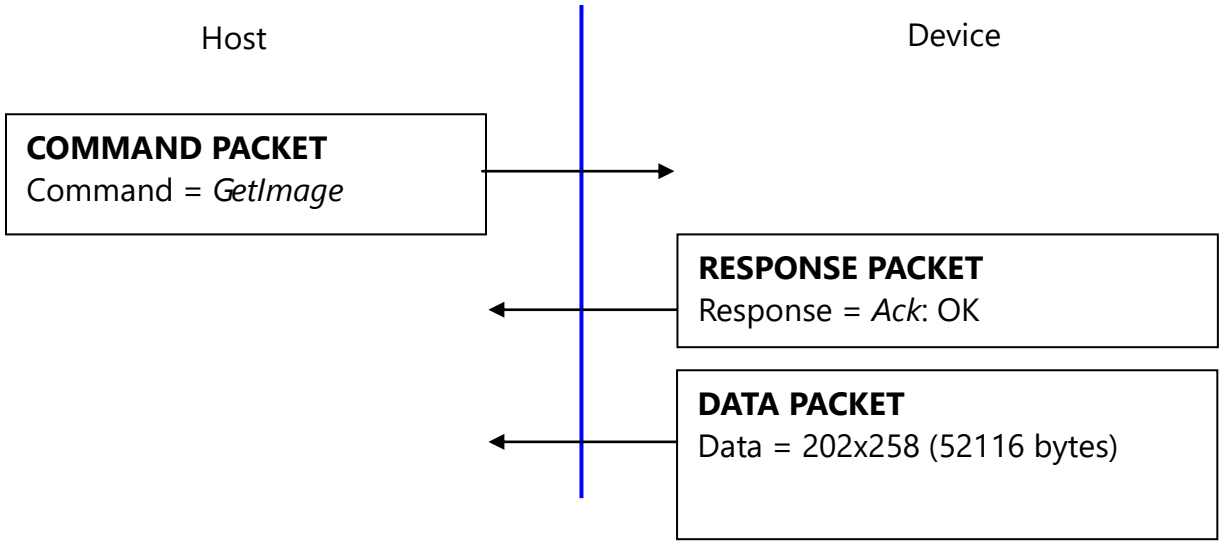
5.16. Capture fingerprint(*CaptureFinger*)

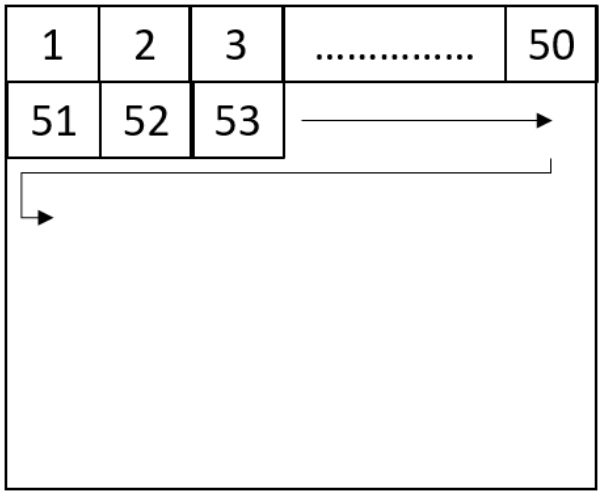


The fingerprint algorithm uses 450dpi 256x256 image for its input.
This command captures raw image from the sensor and converts it to 256x256 image for the fingerprint algorithm. If the finger is not pressed, this command returns with non-acknowledge.
Please use best image for enrollment to get best enrollment data.
Please use not best image for identification (verification) to get fast user sensibility.

NOTE:
Check the finger press on fingerprint sensor must use the CaptureFinger command and Touch signal. When the CaptureFinger command return ACK and the touch signal is high, the finger is pressing on fingerprint sensor.

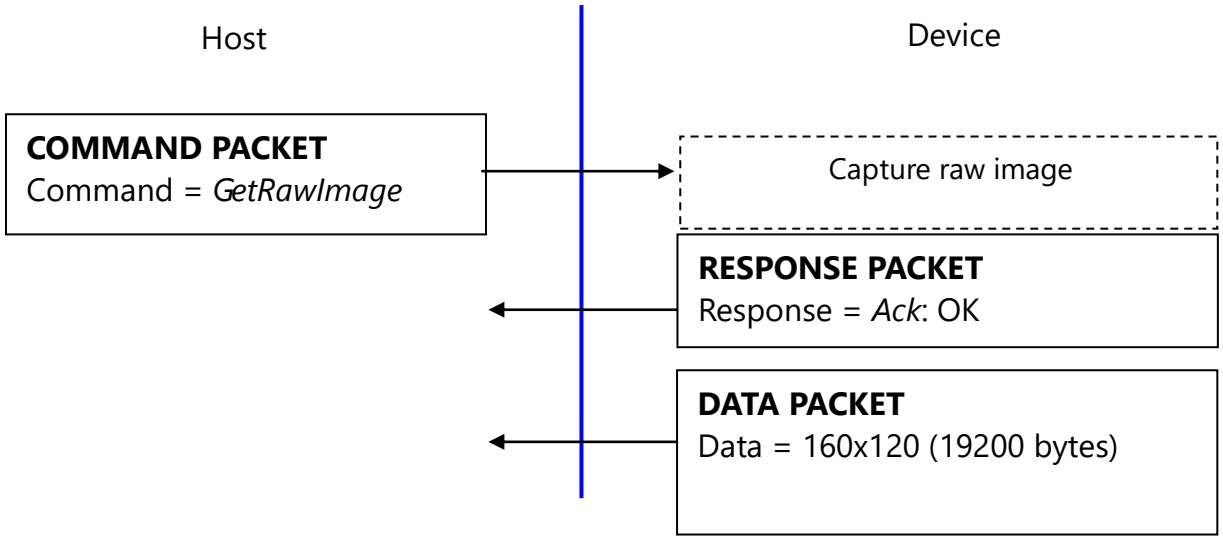
5.17. Get fingerprint image(*GetImage*)

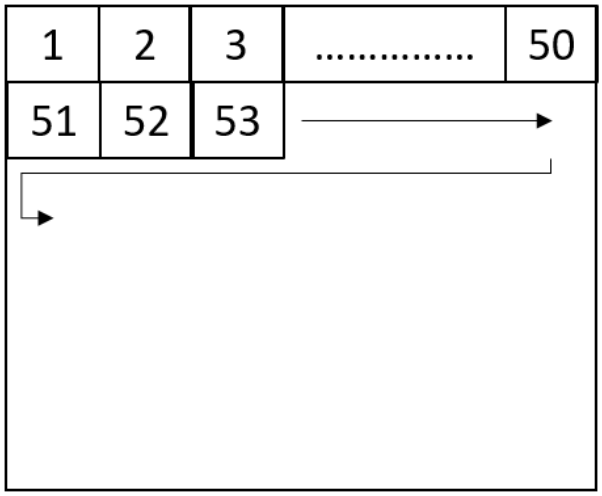




Pixel sequence as above

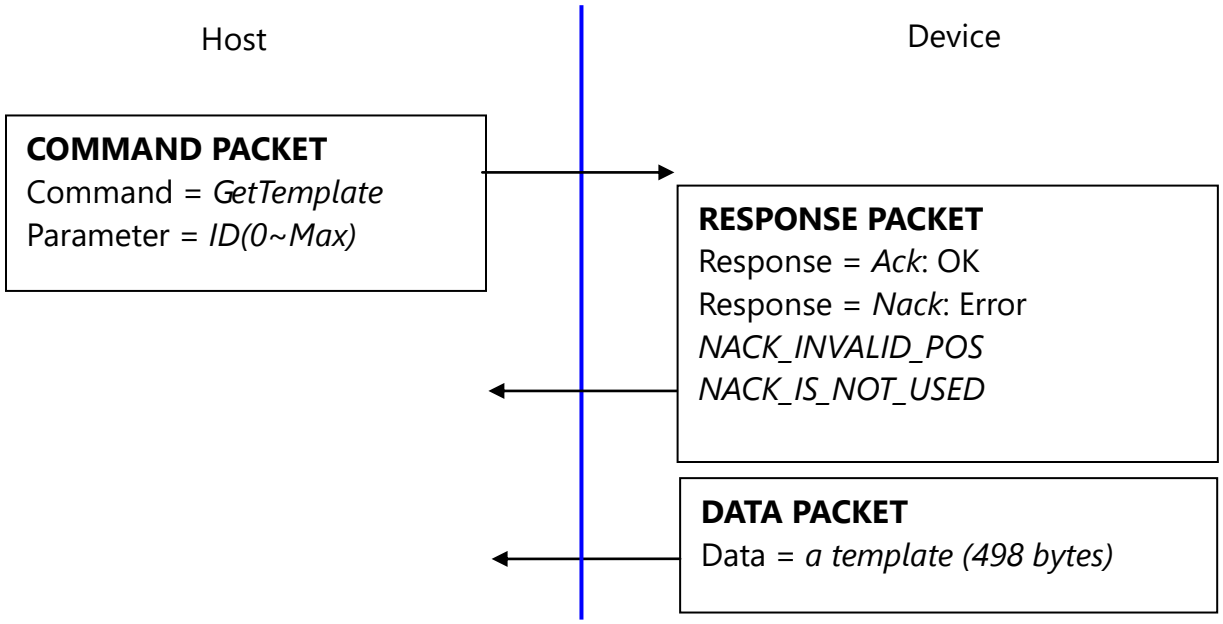
5.18. Get raw image(*GetRawImage*)



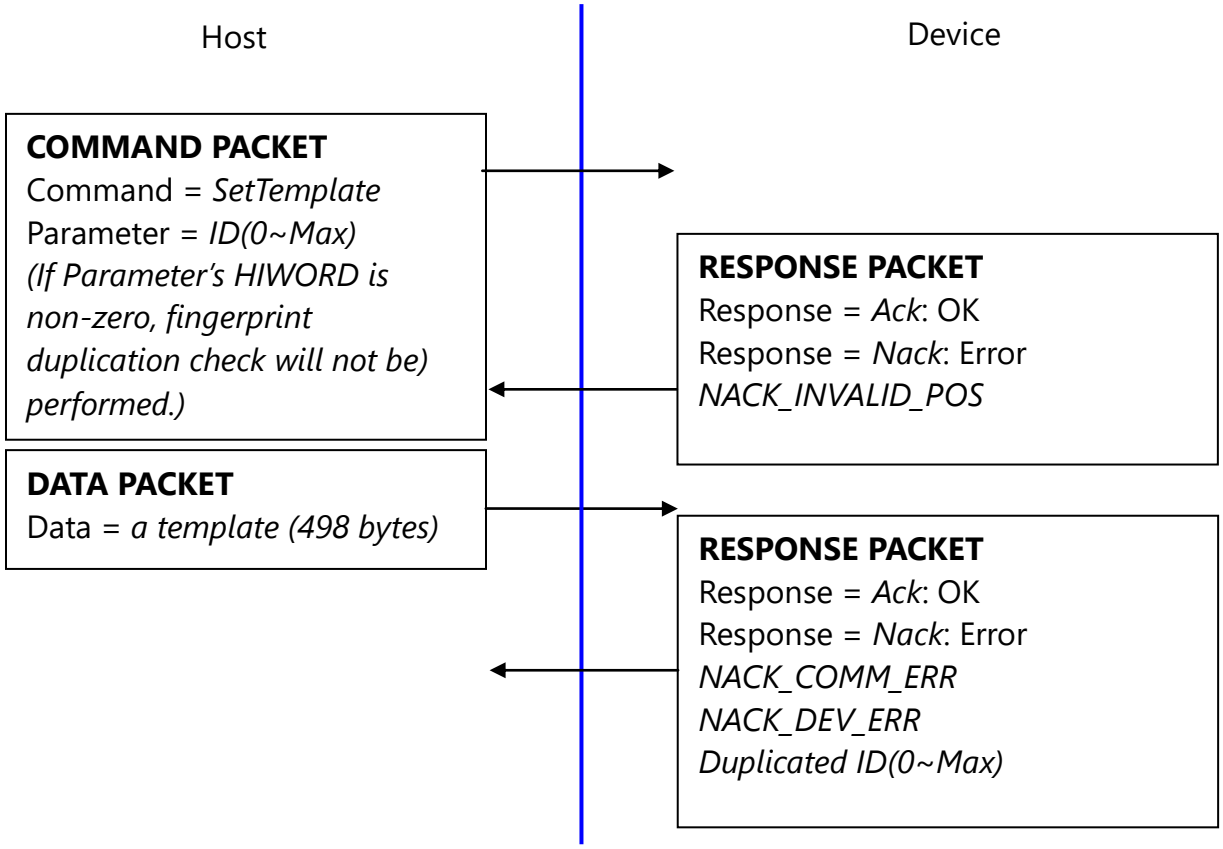


Pixel sequence as above

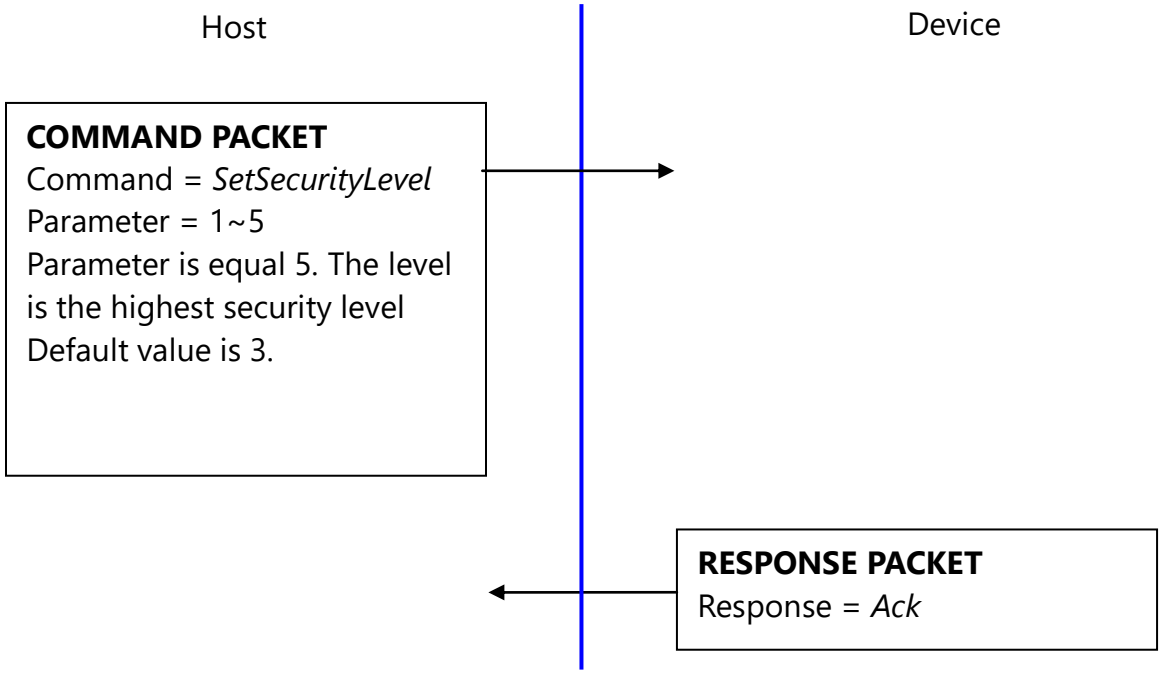
5.19. Get template(*GetTemplate*)



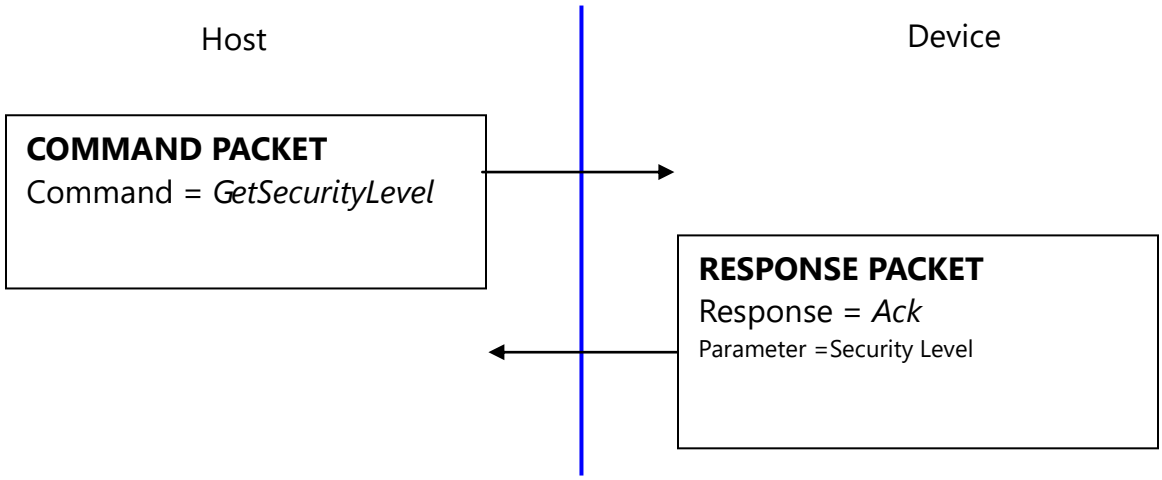
5.20. Set template(*SetTemplate*)



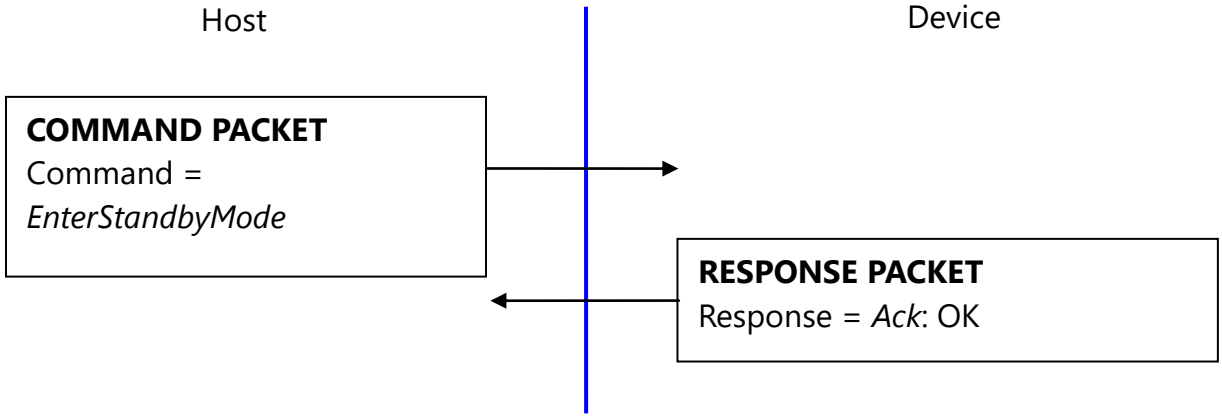
5.21. Set Security Level(*SetSecurityLevel*)



5.22. Get Security Level(*GetSecurityLevel*)



5.23. Standby mode(*EnterStandbyMode*)



Enter Standby Mode (Low power mode). If user want the fingerprint module to leave standby mode, please send the 0x00 first and wait 20ms to weak up and then send the standard command.

6. Protocol: Flowchart, description

6.1. Capture of the fingerprint image

IsPressFinger checks whether a finger placed on the sensor.
This function is used especially while enrollment.

CaptureFinger captures a fingerprint image (256x256), if a finger isn't placed on the sensor, it returns with error.
If this function returns with success, the device's internal RAM keeps valid fingerprint image for the subsequent commands. If the host issues other command, the fingerprint image will be used and destroyed.

GetRawImage captures a raw live image (320x240), it doesn't check whether a finger placed on the sensor, this function is used for debug or calibration.

" *IsPressFinger* & *CaptureFinger* " command must correspond with the touch signal (ICPCK) together.

6.2. Identifying and Verifying

Identify and *IdentifyTemplate* perform 1: N matching operation.
Verify and *VerifyTemplate* perform 1: 1 matching operation.

Just before calling of image-related matching functions (*Identify*, *Verify*), the host must call *CaptureFinger*.

6.3. Enrollment

An enrollment flowchart is as below.

1. *EnrollStart* with a (not used) ID
2. *CaptureFinger & Check ICPCK is “high”.*
3. *Enroll1*
4. Wait to take off the finger using *IsPressFinger*
5. *CaptureFinger & Check ICPCK is “high”.*
6. *Enroll2*
7. Wait to take off the finger using *IsPressFinger*
8. *CaptureFinger & Check ICPCK is “high”.*
9. *Enroll3*