FPC-AM3 Area Sensor Module

Product Specification

FINGERPRINTS

Features

- Embedded stand-alone fingerprint identification system
- Extremely easy to integrate minimizing time-to-market
- Small size
- Thick and hard protective sensor coating, scratch and ESD resistant
- One-To-One verification mode
- One-To-Many identification mode (recommended limit: 500 templates)
- Onboard template storage (991 templates)
- Straightforward serial command interface
- Download/upload template functionality

Application examples

- Access control systems
- Time & Attendance
- Locks, safes
- POS terminals

General description

The FPC-AM3 module acts as a biometric sub-system with onboard template storage.

Integrating the FPC-AM3 module into a product drastically reduces time-to-market with its easy-to-integrate serial command interface and proven robust fingerprint sensor solution.

FPC-AM3 features the robust fingerprint sensor FPC1011F, with its new hard protective coating. The coating protects the sensor against ESD well above 15 kV, as well as scratches, impact and everyday wear-and-tear. The sensor FPC1011F with its 3D pixel sensing technology can read virtually any finger; dry or wet.

FPC-AM3 can easily be integrated into virtually any application and be controlled by a host sending basic commands for enrolment and verification via the serial interface. FPC-AM3 comes preloaded with software and is ready to use at delivery. Fingerprint templates are automatically created and stored in the internal flash memory. Templates used for verification can also be imported from an external storage, e.g. central database, smart card or portable flash memory.

FPC-AM3 can be connected to a host via a board-to-board connector or by using a standard ribbon cable.

Quick reference data

PARAMETER		VALUE
Dimension	(L x W x T)	40 x 23 x 6 mm (Processor board)
Number of templates		991*
Verification time	(1:1)	0.2 s (typical)
Identification time	(1:150)	1.0 s (typical)
Enrolment time		7.0 s (typical)
False-Rejection-Rate	(FRR)	Adjustable, Data dependent
False-Acceptance-Rate	(FAR)	Adjustable from 1/1,000 to 1/100,000
Interface		Serial UART or SPI
Supply voltage		2,5 - 3.3 VDC
Average Supply current active 3,3 V		70 mA @ full speed 25 mA@ half speed
Supply current idle mode 3,3 V		10 mA
Supply current sleep mode 3,3 V		35 µA
Active sensing area		10.64 x 14.00 mm
Pixel resolution		256 gray scale values (8 bit)
ESD protection		> 15 kV
Wear-and-tear		> 1 million wear cycles

* *Recommended number of templates during Identification: <500*

This page is intentionally left blank.

Table of contents

Functional description

1.1 System overview

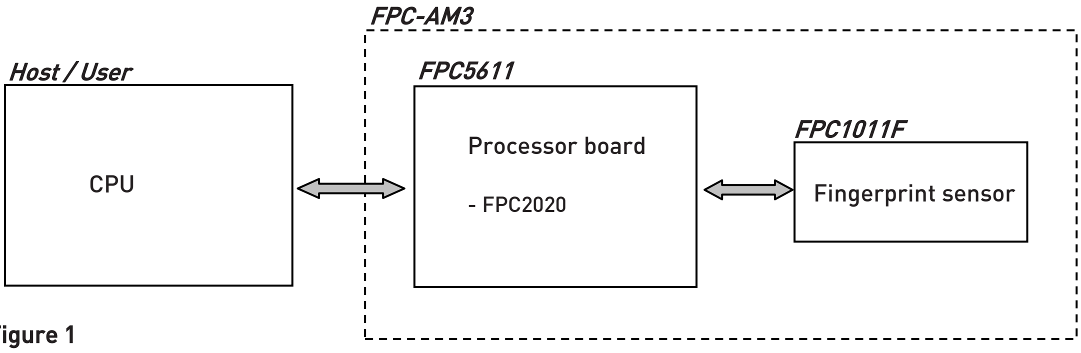


Figure 1
System overview

1.2 General description

The FPC-AM3 area sensor module is a versatile stand-alone fingerprint verification system.

The FPC-AM3 consists of two parts; the area sensor FPC1011F and the processor board FPC5611.

The host CPU, selected and provided by the customer, is executing the main application, interfacing the FPC-AM3 module. Requirements on the host processor associated with the module communication are extremely low. Hence the host processor can be selected entirely to suit the main application at hand.

All CPU capabilities on the FPC5611 processor board are used to perform biometric related functions. Software (firmware) is pre-loaded by Fingerprint Cards and there is no space available for customer applications.

The FPC5611 processor board acquires the fingerprint image from the fingerprint sensor. Thus there is no direct interaction between the host processor and the fingerprint sensor.

The interface between the host processor and the FPC5611 processor board is based on a simple-to-use serial UART or SPI command interface.

1.3 Algorithm performance

The pre-loaded firmware is based on the high performing DAD algorithm developed by Fingerprint Cards.

1.4 FPC1011F Area sensor

The FPC1011F is a small and easy-to-integrate area sensor. The sensor chip is based on capacitive technology and utilizes a reflective measurement method. This method requires a galvanic contact point outside the sensor chip, seen as a conductive frame in the sensor package.

Details on the FPC1011F sensor can be found:

- FPC1011F Product specification

In order to obtain good quality images it is important that the sensor is correctly mounted in the enclosure. A number of general recommendations and useful hints can be found in the ergonomic guideline for area sensor based systems.

- TNT2005 Area sensor guidelines

Details on appropriate enclosure thickness and other important measures are given in the *Mechanical integration* section of this document.

1.5 FPC5611 Processor board

The FPC5611 make use of the FPC2020 fingerprint ASIC for all biometric operations and a flash for program and template storage.

2 Performance characteristics

SYMBOL	PARAMETER	CONDITION	1 USER	10 USERS	100 USERS	UNIT
FRR	False-rejection-rate	High convenience	< 1	< 1	< 4	%
		Default	< 1	< 1	< 5	%
		High security	< 2	< 2	< 7	%
FAR	False-acceptance-rate	High convenience	< 0.1	< 1	1	%
		Default	< 0.01	< 0.1	0.1	%
		High security	< 0.001	< 0.01	0.01	%

Table 1
Algorithm performance characteristics, from a dataset of habituated users.

PROCESS	Full speed			Half speed			UNIT
	MIN	TYP	MAX	MIN	TYP	MAX	
Enrolment	5	7	10	14	20	29	s
Verification 1:1	0.15	0.20	0.42	0.50	0.70	1.7	s
Identification 1:36	0.50	0.70	1.2	1.2	1.5	3.1	s
Identification 1:500	1.7	2.0	2.9	5.5	7.5	10	s

Table 2
Algorithm processing speed.

3 Electrical characteristics

3.1 Normal operation

Operating temperature: -20°C to +60°C

SYMBOL	PARAMETER	CONDITION	MIN	TYP	MAX	UNIT
V_{DD}	Supply voltage		2.5		3.45	V
I_{DD}	Supply current 3,3 V	Idle		10		mA
		Active full speed		70*		mA
		Active half speed		25*		mA
		Sleep		30		µA
Digital inputs						
V_{IL}	Logic '0' voltage		0		0.8	V
V_{IH}	Logic '1' voltage		$0.7V_{DD}$		V_{DD}+0.3	V
I_{IL}	Logic '0' current (V_I = GND)				±10	µA
I_{IH}	Logic '1' current ($V_I = V_{DD}$)				±10	µA
Digital outputs						
V_{OL}	Logic '0' output voltage				0.45	V
V_{OH}	Logic '1' output voltage		$0.85V_{DD}$			V

Table 3
Electrical characteristics

*The average supply current for active mode during fingerprint verification.

3.2 Absolute maximum ratings

Operating temperature	-20°C to +85°C	*Note:* Stress beyond values listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated as normal operation this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.
Storage temperature	-40°C to +85°C	
Supply voltage	-0.5 V to +4.6 V	
Input voltage	-0.5 V to V_{DD} + 0.5 V	

Table 3
Absolute maximum ratings

4 Software command interface

To communicate with the area sensor module a serial command interface is used between the host processor and processor board. This interface is designed to be easy to use and performs the basic biometric functions needed in a fingerprint authentication system.

4.1 Serial interface settings

The software settings for the serial protocol is (using UART):

- Communication speed: factory default baud rate set to 9600 baud (range from 9600 to 115200 baud)
- Format: 8 data bits, odd parity, one stop bit.
- Bit order: least significant bit first

, or (using SPI):

- Communication speed: Guaranteed maximum speed: 1MBit/s. Certain commands can require a slower speed. For details, see the separate section on SPI timing requirements.
- SPI Mode: Mode 3, Clock Polarity High and Clock Phase Rising
- Chip Select: Active Low
- Bit order: most significant bit first

4.2 Command send structure

Command, sent from host:

0	1	2	3	4	5
STX	IDX-LSB	IDX-MSB	COMMAND	PAYLOAD-LSB	PAYLOAD-MSB

STX: Start byte, should be 0x02
IDX-LSB: Index value, least significant byte
IDX-MSB: Index value, most significant byte
COMMAND: Command byte
PAYLOAD-LSB: If any additional data is sent, the payload is a counter of how many bytes that will be sent (not including the CRC-code), otherwise zero.
PAYLOAD-MSB: Payload most significant byte, if no data, set to zero

If PAYLOAD != 0, then additional data should follow in the stream according to the following:

6	...	n	n+1	n+2	n+3	n+4
DATA-1	DATA-...	DATA-n	CRC-LSB	CRC-BYTE2	CRC-BYTE3	CRC-MSB

Note:

1) The CRC size (4 bytes) is not included in the payload counter. Its value is calculated from all the data bytes, and is used for checking if an error occurred during the transmission.

2) Do not send a new command before a response has been received from the previous command. One exception is when using the CANCEL-command, see “Command description / API_CANCEL”.

3) The default choice for IDX-LSB and IDX-MSB is 0x00, if nothing else is stated.

4.3 Response structure

Response from device:

0	1	2	3
STX	RESULT	PAYLOAD-LSB	PAYLOAD-MSB

STX: Start byte, should be 0x02
RESULT: Result byte
PAYLOAD-LSB: If any additional data is sent, the payload is a counter of how many bytes that will be sent (not including the CRC-code), otherwise zero.
PAYLOAD-MSB: Payload most significant byte, if no data, set to zero

If PAYLOAD != 0, then additional data should follow in the stream according to the following:

4	...	n	n+1	n+2	n+3	n+4
DATA-1	DATA-...	DATA-n	CRC-LSB	CRC-BYTE2	CRC-BYTE3	CRC-MSB

Note:
The CRC size (4 bytes) is not included in the payload counter. Its value is calculated from all the data bytes, and is used for checking if an error occurred during the transmission.

4.4 SPI Timing Requirements

The SPI interface of the FPC-AM3 is a slave interface, implying that the host (which is the master) determines when each data is sent to and from the FPC-AM3. Since the host cannot know when the FPC-AM3 has completed processing a given command, a polling process must be implemented by the host when trying to read the response for a given command request. (This is different from the UART interface, where the host knows that the received response is the correct response.)

The required implementation of the Request/Response process is the following:

1. Let the host send the 6 command bytes (+ a possible payload and crc).
2. Wait 5 ms.
3. Let the host send a byte with value 0x52 to the slave. Here, the host asks the FPC-AM3 if it is ready to send the response back to the host.
4. Check if the received byte is 0x02. If not, the slave is not ready, and need more time to complete processing the command. Otherwise, the FPC-AM3 returns a 0x52, as the 'BUSY' signal.
   Repeat steps 3-4 until a 0x02 byte is received as response.
5. This 0x02 value is the first byte in the regular response consisting of 4 bytes (+ a possible payload and crc). Read it as usual. Important: Do not read more bytes than the complete response, since that could set the command interpretator in an incorrect state.

The SPI data transfer speed can be at least 2.5MBits/s during a single byte transmission, but certain delays are required between bytes, for all commands to work properly. This delay varies with the different commands. For example, uploading an image with CRC generation turned on requires a larger byte delay that if CRC generation is skipped.
The suggested default byte delay is 18µs, low speed byte delay is 30µs, and high speed byte delay is 4µs. The Low speed delay must be used during image upload with CRC, template upload with CRC, and the Firmware Upgrade command. The high speed byte delay can be used with image up/download without CRC. Effectively, this means that an image can be transferred with over 1.2MBit/s.

4.5 Table of commands

BIOMETRIC COMMANDS	HEX	DESCRIPTION
API_CAPTURE_IMAGE	0x80	Capture image from sensor (before enrol).
API_CAPTURE_AND_ENROL_RAM	0x81	Enrol into RAM (includes Capture Image)
API_CAPTURE_AND_VERIFY_RAM	0x82	Verify against RAM (includes Capture Image)
API_CAPTURE_AND_VERIFY_FLASH	0x83	Verify against single FLASH slot (includes Capture Image) Set slot number in IDX
API_CAPTURE_AND_IDENTIFY_FLASH	0x84	Identify against all FLASH slots (includes Capture Image)
API_ENROL_RAM	0x85	Enrol into RAM
API_VERIFY_RAM	0x86	Verify against RAM
API_VERIFY_FLASH	0x87	Verify against single FLASH slot Set slot number in IDX
API_IDENTIFY_FLASH	0x88	Identify against all FLASH slots
API_CAPTURE_IMAGE_FINGERPRESENT	0x89	Capture Image from sensor (once a finger is present)
API_ENROL_FLASH	0x92	Enrol into FLASH memory
API_CAPTURE_AND_ENROL_FLASH	0x93	Enrol into FLASH memory (includes Capture Image)

IMAGE TRANSFER	HEX	DESCRIPTION
API_UPLOAD_IMAGE	0x90	Upload image from RAM
API_DOWNLOAD_IMAGE	0x91	Download image to RAM

TEMPLATE HANDLING	HEX	DESCRIPTION
API_UPLOAD_TEMPLATE	0xA0	Upload template from RAM
API_DOWNLOAD_TEMPLATE	0xA1	Download template to RAM
API_COPY_TEMPLATE_RAM_TO_FLASH	0xA2	Copy template from RAM to permanent FLASH storage Set slot number (0 to 990) in IDX
API_UPLOAD_TEMPLATE_FROM_FLASH	0xA3	Upload template from single FLASH slot Set slot number (0 to 990) in IDX
API_DELETE_TEMPLATE_RAM	0xA4	Erase template from RAM
API_DELETE_SLOT_IN_FLASH	0xA5	Delete single slot in FLASH Set slot number (0 to 990) in IDX
API_DELETE_ALL_IN_FLASH	0xA6	Delete all FLASH slots
API_DOWNLOAD_TEMPLATE_TO_FLASH	0xA7	Download a template to FLASH

Table 5
Table of commands.

ALGORITHM SETTINGS	HEX	DESCRIPTION
API_SECURITY_LEVEL_RAM	0xB0	Set security level, setting saved in RAM IDX-LSB: 0x04 = high convenience 0x05 = standard 0x06 = high security
API_SECURITY_LEVEL_STATIC	0xB1	Set security level, setting saved in non-volatile (static) memory
API_GET_SECURITY_LEVEL	0xB2	Get current security level, value sent as payload data

FIRMWARE COMMANDS	HEX	DESCRIPTION
API_FIRMWARE_VERSION	0xC0	Upload the version string for this device
API_FIRMWARE_UPDATE	0xC1	Start download of new firmware

COMMUNICATION COMMANDS	HEX	DESCRIPTION
API_SET_BAUD_RATE_RAM	0xD0	Set baud rate, setting saved in RAM IDX-LSB: 0x10 = 9600 baud 0x20 = 14400 baud 0x30 = 19200 baud 0x40 = 28800 baud 0x50 = 38400 baud 0x60 = 57600 baud 0x70 = 76800 baud 0x80 = 115200 baud
API_SET_BAUD_RATE_STATIC	0xD1	Set baud rate, setting saved in non-volatile (static) memory IDX-LSB: 0x00 = current speed (recommended usage) 0x10 = 9600 baud 0x20 = 14400 baud 0x30 = 19200 baud 0x40 = 28800 baud 0x50 = 38400 baud 0x60 = 57600 baud 0x70 = 76800 baud 0x80 = 115200 baud
API_TEST_HARDWARE	0xD2	Test hardware components

OTHER COMMANDS	HEX	DESCRIPTION
API_CANCEL	0xE0	Cancel ongoing command, only valid for: API_CAPTURE_AND_ENROL_RAM, API_CAPTURE_AND_VERIFY_RAM, API_CAPTURE_AND_VERIFY_AGAINST_FLASH, API_CAPTURE_AND_IDENTIFY_AGAINST_FLASH
API_ENTER_SLEEP_MODE	0xE1	Enter sleep mode (wake up by activating proper pin)
API_POWER_SAVE_MODE_RAM	0xE2	Set power save mode, setting saved in RAM IDX-LSB: 0x00 = enable 0x01 = disable
API_POWER_SAVE_MODE_STATIC	0xE3	Set power save mode, setting saved in non-volatile (static) memory IDX-LSB: 0x00 = enable 0x01 = disable
API_GET_POWER_SAVE_MODE	0xE5	Get current power save mode, value sent as payload data: 0x00 = enabled 0x01 = disabled
API_ADVANCED_SETTINGS	0xE8	Managing advanced settings, e.g. supply voltage control.

Table 6
Table of commands (continued)

4.6 Table of response bytes

COMMAND	HEX
API_FAILURE	0x00
API_SUCCESS	0x01
API_NO_FINGER_PRESENT	0x02
API_FINGER_PRESENT	0x03
API_VERIFICATION_OK	0x04
API_VERIFICATION_FAIL	0x05
API_ENROL_OK	0x06
API_ENROL_FAIL	0x07
API_HW_TEST_OK	0x08
API_HW_TEST_FAIL	0x09
API_CRC_FAIL	0x0A
API_PAYLOAD_TOO_LONG	0x0B
API_PAYLOAD_TOO_SHORT	0x0C
API_UNKNOWN_COMMAND	0x0D
API_NO_TEMPLATE_PRESENT	0x0E
API_IDENTIFY_OK	0x0F
API_IDENTIFY_FAIL	0x10
API_INVALID_SLOT_NR	0x11
API_CANCEL_SUCCESS	0x12
API_APPL_CRC_FAIL	0x14
API_SYS_CRC_FAILED	0x16
API_LOW_VOLTAGE	0x17

Table 7
Table of response bytes

5 Command description

This section describes the individual commands of the main application, along with their parameters, and responses.

5.1 Capture image API_CAPTURE_IMAGE

An image is captured from the fingerprint sensor. The fingerprint image is placed in RAM and can be uploaded by the command API_UPLOAD_IMAGE. Calculation is done on the image to determine if a finger is present or not present on the sensor. No payload is sent with this command.

Response command:
- API_NO_FINGER_PRESENT = No finger present on sensor
- API_FINGER_PRESENT = Finger present on sensor

No payload is received with the response from this command.

5.2 Capture and enrol (RAM) API_CAPTURE_AND_ENROL_RAM

An image is captured from the fingerprint sensor and enrolment of this image is done. The command waits for "finger present" before it starts the enrolment. This means that images are captured in a loop from the sensor until a finger is present. The command returns with response when the enrolment is complete or if the enrolment fails for any reason. After enrolment the template is stored in RAM and can be uploaded or moved to FLASH storage. No payload is sent with this command.

Response command:
- API_ENROL_OK = Enrolment successful
- API_ENROL_FAIL = Enrolment failed

No payload is received with the response from this command.

Note:
It is possible to cancel the current enrol operation by sending the command API_CANCEL. This cancels the enrolment and the device returns to its normal command loop.

5.3 Capture and verify (RAM) API_CAPTURE_AND_VERIFY_RAM

A template must be present in RAM before starting the verification, either by using the Download Template command (API_DOWNLOAD_TEMPLATE) OR the command API_CAPTURE_ENROL_RAM. Thereafter the verification can be started. This command also captures an image from the fingerprint sensor. The command waits for "finger present" before it starts the verification. This means that images are captured in a loop from the sensor until a finger is present. The command returns with response when the verification is complete or if the verification fails for any reason. No payload is sent with this command.

Response command:
- API_VERIFICATION_OK = Verification successful
- API_VERIFICATION_FAIL = Verification failed
- API_NO_TEMPLATE_PRESENT = No template present

No payload is received with the response from this command.

Note:
It is possible to cancel the current verification operation by sending the command API_CANCEL. This cancels the verification and the device returns to its normal command loop.

5.4 Capture and verify (FLASH) API_CAPTURE_AND_VERIFY_FLASH

The FLASH slot number must be given in the IDX bytes. This command first captures an image from the fingerprint sensor. The command waits for "finger present" before it starts the verification. This means that images are captured in a loop from the sensor until a finger is present. The command returns with response when the verification is complete or if the verification fails for any reason. No payload is sent with this command.

Response command:
- API_VERIFICATION_OK = Verification successful
- API_VERIFICATION_FAIL = Verification failed
- API_NO_TEMPLATE_PRESENT = No template in given FLASH slot
- API_INVALID_SLOT_NR = Wrong slot number

No payload is received with the response from this command.

Note: It is possible to cancel the current verification operation by sending the command API_CANCEL. This cancels the verification and the device returns to its normal command loop.

5.5 Capture and identify (FLASH) API_CAPTURE_ AND _IDENTIFY_FLASH

Identification is made against all FLASH slots. This command first captures an image from the fingerprint sensor. The command waits for "finger present" before it starts the identification. This means that images are captured in a loop from the sensor until a finger is present. The command returns with response when the identification is complete or if the identification fails for any reason. No payload is sent with this command.

Response command:
- API_ IDENTIFY _OK = Identification successful
- API_ IDENTIFY _FAIL = Identification fails

In a successful identification, the slot index is received as payload in two bytes (LSB first) plus the 4 CRC bytes.

Note:
It is possible to cancel the current identification operation by sending the command API_CANCEL. This cancels the identification and the device returns to its normal command loop.
For good biometric performance, it is recommended to use no more than 500 templates during identification.

5.6 Enrol (RAM) API_ENROL_RAM

A fingerprint image must be present in RAM before starting the enrolment, either by capturing an image from the fingerprint sensor using the command API_CAPTURE_IMAGE OR by using the Download Image Command (API_DOWNLOAD_IMAGE). The command returns with response when the enrolment is complete or if the enrolment fails for any reason. After enrolment the template is stored in RAM and can be uploaded or moved to FLASH storage. No payload is sent with this command.

Response command:
- API_ENROL_OK = Enrolment successful
- API_ENROL_FAIL = Enrolment failed

No payload is received with the response from this command.

5.7 Verify (RAM) API_VERIFY_RAM

A template AND a fingerprint image must be present in RAM before starting the verification. To handle the template use the command Download Template API_DOWNLOAD_TEMPLATE) OR the command API_CAPTURE_ENROL_RAM. To handle the image use the command Download Image API_DOWNLOAD_IMAGE) OR the command API_CAPTURE_IMAGE. Thereafter the verification can be started. The command returns with response when the verification is complete or if the verification fails for any reason. No payload is sent with this command.

Response command:
- API_VERIFICATION_OK = Verification successful
- API_VERIFICATION_FAIL = Verification failed
- API_NO_TEMPLATE_PRESENT = No template present

No payload is received with the response from this command.

5.8 Verify (FLASH) API_VERIFY_FLASH

A fingerprint image must be present in RAM before starting the verification, use the command Download Image API_DOWNLOAD_IMAGE) OR the command API_CAPTURE_IMAGE. The FLASH slot number must be given in the IDX bytes. The command returns with response when the verification is complete or if the verification fails for any reason. No payload is sent with this command.

Response command:
- API_VERIFICATION_OK = Verification successful
- API_VERIFICATION_FAIL = Verification failed
- API_NO_TEMPLATE_PRESENT = No template in given FLASH slot
- API_INVALID_SLOT_NR = Wrong slot number

No payload is received with the response from this command.

5.9 Identify (FLASH) API_IDENTIFY_FLASH

A fingerprint image must be present in RAM before starting the verification, use the command Download Image API_DOWNLOAD_IMAGE) OR the command API_CAPTURE_IMAGE. Identification is made against all FLASH slots. The command returns with response when the identification is complete or if the identification fails for any reason. No payload is sent with this command.

Response command:
- API_ IDENTIFY _OK = Identification successful
- API_ IDENTIFY _FAIL = Identification failed
- API_NO_TEMPLATE_PRESENT = All FLASH slots are empty

In a successful identification, the slot index is received as payload in two bytes (LSB first), plus the 4 CRC bytes.

Note: For good biometric performance, it is recommended to use no more than 500 templates during identification.

5.10 Capture image (Fingerpresent) API_CAPTURE_IMAGE_FINGERPRESENT

An image is captured from the fingerprint sensor once the system detects a finger on the sensor. The fingerprint image is placed in RAM and can be uploaded by the command API_UPLOAD_IMAGE. No payload is sent with this command. The system waits until a finger is detected, and can only be terminated with the API_CANCEL command.

Response command:
- API_FINGER_PRESENT = Finger present on sensor

No payload is received with the response from this command.

5.11 Enrol (FLASH) API_ENROL_FLASH

A fingerprint image must be present in RAM before starting the enrolment, either by capturing an image from the fingerprint sensor using the command API_CAPTURE_IMAGE OR by using the Download Image Command (API_DOWNLOAD_IMAGE). The command returns with response when the enrolment is complete or if the enrolment fails for any reason. After enrolment the template is stored in FLASH and can be uploaded or moved to FLASH storage. The desired FLASH slot number must be given in the IDX bytes. No payload is sent with this command.

Response command:
- API_ENROL_OK = Enrolment successful
- API_ENROL_FAIL = Enrolment failed
- API_INVALID_SLOT_NR = Incorrect FLASH slot number

No payload is received with the response from this command.

5.12 Capture Enrol (FLASH) API_CAPTURE_AND_ENROL_FLASH

This command first captures an image from the fingerprint sensor. The command waits for "finger present" before it starts the verification. This means that images are captured in a loop from the sensor until a finger is present. The command then returns with a response when the enrolment is complete or if the enrolment fails for any reason. After enrolment the template is stored in FLASH and can be uploaded or moved to FLASH storage. The desired FLASH slot number must be given in the IDX bytes. No payload is sent with this command.

Response command:
- API_ENROL_OK = Enrolment successful
- API_ENROL_FAIL = Enrolment failed
- API_INVALID_SLOT_NR = Incorrect FLASH slot number

No payload is received with the response from this command.

5.13 Upload image API_UPLOAD_IMAGE

By using this command it is possible to upload the fingerprint image present in RAM. The response is the API_SUCCESS command followed by the image data. The size of image data is 30400 bytes. The first byte is the upper left pixel and then data follows row-wise (X-direction). Each pixel has one byte value (256 gray scales). There is no image header. No payload is sent with this command. Note that if IDX_LSB = 0x01, the CRC generation is skipped. This will increase the maximum SPI transfer speed for this command.

Response command:
- API_ SUCCESS = Upload successful
- API_ FAILURE = Upload failed

The received payload in a successful upload consists of 30400 bytes plus the 4 CRC bytes.

Download image API_DOWNLOAD_IMAGE

By using this command it is possible to download a fingerprint image to RAM. The size of image data must be 30400 bytes. The first byte is the upper left pixel and then data follows row-wise (X-direction). Each pixel has one byte value (256 grey scales). There is no image header. Payload size of this command must be 30400 bytes, plus the 4 CRC-bytes sent after the payload. Note that if IDX_LSB = 0x01, the CRC check is skipped (but it must still be included in the request). This will increase the maximum SPI transfer speed for this command.

Response command:
- API_ SUCCESS = Download successful
- API_ FAILURE = Download failed

No payload is received with the response from this command.

5.14 Upload template API_UPLOAD_TEMPLATE

After a successful enrolment the template is uploaded from RAM using the Upload Template command (API_UPLOAD_TEMPLATE). The response is the API_SUCCESS command followed by the template data. The template consists of 938 bytes binary data with no public information. No payload is sent with this command.

Response command:

- API_ SUCCESS = Upload successful
- API_ FAILURE = Upload failed

The received payload in a successful upload consists of 938 bytes plus the 4 CRC bytes.

5.15 Download template API_DOWNLOAD_TEMPLATE

Before verification the template is downloaded to RAM using the Download Template command (API_DOWNLOAD_TEMPLATE). The size of the template is 938 bytes. Payload size of this command must be 938 bytes, plus the 4 CRC-bytes sent after the payload.

Response command:

- API_ SUCCESS = Download successful
- API_ FAILURE = Download failed

No payload is received with the response from this command.

5.16 Copy template from RAM to FLASH API_COPY_TEMPLATE_FROM_RAM_TO_FLASH

This command copies the template currently in RAM to FLASH. The FLASH slot number must be given in the IDX bytes. No payload is sent with this command.

Response command:

- API_ SUCCESS = Template storage successful
- API_ FAILURE = Template storage failed
- API_INVALID_SLOT_NR = Wrong slot number

No payload is received with the response from this command.

5.17 Upload template from FLASH API_UPLOAD_TEMPLATE_FROM_FLASH

This command uploads the template from FLASH. The FLASH slot number must be given in the IDX bytes. No payload is sent with this command.

Response command:

- API_ SUCCESS = Upload successful
- API_ FAILURE = Upload failed
- API_INVALID_SLOT_NR = Wrong slot number

The received payload in a successful upload consists of 938 bytes plus the 4 CRC bytes.

5.18 Delete template in RAM API_DELETE_TEMPLATE_RAM

This command deletes the template currently stored in RAM. No payload is sent with this command.

Response command:

- API_ SUCCESS = Template removal successful
- API_ FAILURE = Template removal failed

No payload is received with the response from this command.

5.19 Delete single template in FLASH API_DELETE_SLOT_IN_FLASH

By using the command Delete slot in FLASH one can choose which slot to delete (include slot number in index value of command). No payload is sent with this command.

Response command:
- API_ SUCCESS = Template removal successful
- API_ FAILURE = Template removal failed
- API_INVALID_SLOT_NR = Wrong slot number

No payload is received with the response from this command.

5.20 Delete all templates in FLASH API_DELETE_ALL_IN_FLASH

It is possible to delete all templates in FLASH by issuing the command Delete all in FLASH. No payload is sent with this command.

Response command:
- API_ SUCCESS = Template removal successful
- API_ FAILURE = Template removal failed

No payload is received with the response from this command.

5.21 Download template to FLASH API_DOWNLOAD_TEMPLATE_TO_FLASH

Downloads a template from the host directly into the FLASH memory (and into RAM). The FLASH slot number must be given in the IDX bytes. The size of the template is 938 bytes. Payload size of this command must be 938 bytes, plus the 4 CRC-bytes sent after the payload.

Response command:
- API_ SUCCESS = Download successful
- API_ FAILURE = Download failed
- API_INVALID_SLOT_NR = Wrong slot number

No payload is received with the response from this command.

5.22 Security level (RAM) API_SECURITY_LEVEL_RAM

The security level to be used during verification and identification can be set by the command Set security level. The value of the security level should be set in the index value (IDX-LSB) of the command. The factory default security level is set to value 0x04. The value is stored in RAM and the setting is lost after reset. The security level is not stored together with the template. During enrolment there is no effect when changing the security threshold. The created template will support all security settings. No payload is sent with this command.

VALUE (IDX-LSB)	SECURITY LEVEL
0x04	High convenience (factory default)
0x05	Standard
0x06	High security

Response command:
- API_ SUCCESS = New security level set
- API_ FAILURE = Security level out of range

No payload is received with the response from this command.

5.23 Security level (STATIC) API_SECURITY_LEVEL_STATIC

The security level to be used during verification and identification can be set by the command Set security level. The value of the security level should be set in the index value (IDX-LSB) of the command. The factory default security level is set to value 0x04. The value is stored in non-volatile memory and the setting is saved even after reset. This means that the factory default value will be changed. The security level is not stored together with the template. During enrolment there is no effect when changing the security threshold. The created template will support all security settings. No payload is sent with this command.

VALUE (IDX-LSB)	SECURITY LEVEL
0x04	High convenience (factory default)
0x05	Standard
0x06	High security

Response command:

- API_ SUCCESS = New security level set
- API_ FAILURE = Security level out of range

5.24 Get current security level API_GET_SECURITY_LEVEL

This command returns the value of the current security setting that the module uses. The value is sent as payload data. No payload is sent with this command.

VALUE	SECURITY LEVEL
0x04	High convenience (factory default)
0x05	Standard
0x06	High security

Response command:

- API_ SUCCESS = Command OK
- API_ FAILURE = Command fail

No payload is received with the response from this command.

5.25 Firmware version API_FIRMWARE_VERSION

This command returns the firmware version of the main application. The response is the API_SUCCESS command followed by the firmware version string. No payload is sent with this command.

Response command:

- API_ SUCCESS = Request successful, version string follows as payload
- API_ FAILURE = Request failed

A payload + 4 CRC bytes will be received in a successful request. The size of this payload could vary with the version of the firmware.

5.26 Firmware update API_FIRMWARE_UPDATE

It is possible to update the pre-loaded firmware, by downloading the new program to the FLASH memory. This is done in several steps – filling one FLASH memory slot at a time.
The new firmware is divided in N pieces of 256 bytes each, and each such 256-byte code payload is sent to FPC-AM3 using the API_FIRMWARE_UPDATE command, with the IDX bytes starting at '0' for the first command, and ending at 'N-1' for the last command.
Thus, a complete upgrade requires N successful applications of this command. If the upgrade is not completed, or fails in any way, the application will start in 'Boot Mode' after next reset. For more info on this mode, see the section on Boot Mode. Payload size of this command must be 256 bytes, plus the 4 CRC-bytes sent after the payload.

Response command:
- API_ SUCCESS = Uploaded bytes were successfully received
- API_ CRC_FAIL = Uploaded bytes were incorrectly received

No payload is received with the response from this command.

If the firmware is corrupt it is still possible to set baud rate and perform firmware updates.

5.27 Set baud rate (RAM) API_SET_BAUD_RATE_RAM

It is possible to change baud rate for the serial communication between host and FPC-AM3. The table below shows the available baud rates. Factory default baud rate is 9600. The selected value should be set in the IDX-LSB byte of the command. The value is stored in RAM and the setting is lost after reset. No payload is sent with this command.

VALUE (IDX-LSB)	BAUD RATE
0x10	9600 (factory default)
0x20	14400
0x30	19200
0x40	28800
0x50	38400
0x60	57600
0x70	76800
0x80	115200

Response command:
- API_ SUCCESS = Baud rate change accepted
- API_ FAILURE = Baud rate out of range

No payload is received with the response from this command.

Note:
Once a baud rate change has been accepted, next command must be sent with new baud rate.
However the response command above is sent with the old baud rate.

5.28 Set baud rate (STATIC) API_SET_BAUD_RATE_STATIC

It is possible to change baud rate for the serial communication between host and FPC-AM3. The table below shows the available baud rates. Factory default baud rate is 9600.
The selected value should be set in the IDX-LSB byte of the command. The value is stored in non-volatile memory and the setting is saved even after reset. This means that the factory default value will be changed.
Note that if the index 0 is used, the currently active baud rate will be stored. This is the recommended option, since it will ensure that a non-compatible UART speed not is permanently selected. No payload is sent with this command.

VALUE (IDX-LSB)	BAUD RATE
0x00	Currently active baudrate
0x10	9600 (factory default)
0x20	14400
0x30	19200
0x40	28800
0x50	38400
0x60	57600
0x70	76800
0x80	115200

Response command:
- API_ SUCCESS = Baud rate change accepted
- API_ FAILURE = Baud rate out of range

No payload is received with the response from this command.

Note:
Once a baud rate change has been accepted, next command must be sent with new baud rate. However the response command above is sent with the old baud rate.

5.29 Test hardware API_TEST_HARDWARE

This command tests the FPC-AM3 hardware. It performs a check of the different components on the module. No payload is sent with this command.

Response command:
- API_HW_TEST_OK = Hardware check successful
- API_HW_TEST_FAIL = Hardware check failed, contact technical support

No payload is received with the response from this command.

5.30 Cancel current command API_CANCEL

It is possible to cancel the following ongoing commands:

API_CAPTURE_IMAGE_FINGERPRESENT
API_CAPTURE_AND_ENROL_RAM
API_CAPTURE_AND_ENROL_FLASH
API_CAPTURE_AND_VERIFY_RAM
API_CAPTURE_AND_VERIFY_FLASH
API_CAPTURE_AND_IDENTIFY_ FLASH

The module will respond with API_CANCEL_SUCCESS and the return to normal command loop. No payload is sent with this command.

Response command:
- API_ CANCEL_SUCCESS = Cancel successful
- API_ FAILURE = Cancel failed

No payload is received with the response from this command.

5.31 Enter sleep mode API_ENTER_SLEEP_MODE

SLEEP MODE is entered by issuing the command API_ENTER_SLEEP_MODE. In SLEEP MODE the device runs on low power. To wake up the device, a wakeup interrupt must occur. This is triggered by one of three possible signals: HOST_SI (active low), HOST_SCN (active low) or SENSOR_IRQ (active high). No payload is sent with this command. Before the device enters SLEEP MODE it responds with one of the following:

Response command:
- API_ SUCCESS = Request accepted, entering SLEEP MODE
- API_ FAILURE = Request failed

No payload is received with the response from this command.

5.32 Power save mode (RAM) API_POWER_SAVE_MODE_RAM

In POWER SAVE MODE the module reduces the clock frequency of the processor by half to lower power consumption. To enter POWER SAVE MODE, issue the command Power Save Mode with the value 0 in the IDX-LSB byte. To exit POWER SAVE MODE, issue the command Power Save Mode with the value 1 in the IDX-LSB byte. The setting is stored in RAM and the setting is lost after reset.
No payload is sent with this command. The factory default setting is that the module is NOT in power save mode (value=1).

VALUE (IDX-LSB)	DESCRIPTION
0	Half Speed
1	Full Speed (factory default)

Response command:
- API_ SUCCESS = Request accepted, entering POWER SAVE MODE
- API_ FAILURE = Request failed

No payload is received with the response from this command.

5.33 Power save mode (STATIC) API_POWER_SAVE_MODE_STATIC

In POWER SAVE MODE the module reduces the clock frequency of the processor by half to lower power consumption. To enter POWER SAVE MODE, issue the command Power Save Mode with the value 0 in the IDX-LSB byte. To exit POWER SAVE MODE, issue the command Power Save Mode with the value 1 in the IDX-LSB byte. The setting is stored in non-volatile memory and the setting is saved even after reset. This means that the factory default setting (value=1) will be changed. No payload is sent with this command.

VALUE (IDX-LSB)	DESCRIPTION
0	Half Speed
1	Full Speed (factory default)

Response command:
- API_ SUCCESS = Request accepted, entering POWER SAVE MODE
- API_ FAILURE = Request failed

No payload is received with the response from this command.

5.34 Get current power save mode API_GET_POWER_SAVE_MODE

This command returns the value of the current setting of power save mode.
The value is received as payload data. No payload is sent with this command.

VALUE	DESCRIPTION
0	Half Speed
1	Full Speed (factory default)

Response command:

- API_ SUCCESS = Command OK
- API_ FAILURE = Command fail

The received payload in a successful upload consists of 1 byte plus the 4 CRC bytes.

5.35 Manage Advance Settings API_ADVANCED_SETTINGS

This command is used both to get current, and to set current advanced settings. The advanced settings presently supported are the Supply Voltage Control, and disabling of the UART host interface.
The Supply Voltage Control uses the 'Brown Out Detection' functionality in FPC-AM3 to check that the Supply Voltage is at a proper level. If enabled, the system will respond with the response code API_LOW_VOLTAGE to any command, whenever the supply voltage falls below the specified level. (See the section on Brown Out Detection on details of how to determine the specified level.)
The Supply Voltage Control can be enabled temporarily (in RAM) or statically (in non-volatile memory). In the latter case, the setting will remain after a system reset. The factory setting is a disabled Supply Voltage Control.
If using the FPC-AM3 in an SPI chain with other SPI components, it is necessary to disable the UART host interface of the processor, (since SPI data and UART data connectors are shared, and UART is active when SPI is inactive, and vice versa). This can be done with the API_ADVANCED_SETTINGS command. The setting can be stored in RAM or statically in the Flash memory. Note that the UART interface can only be disabled if the command is sent using an SPI host interface, otherwise the command will fail.
Note that when issuing this command, the host is forced to set the values of both the advanced settings with the same payload byte.

When this command is sent with a one-byte payload, it is used to put the system in the desired mode, by turning on/off individual bits in the payload byte:

COMMAND PAYLOAD VALUE	DESCRIPTION
Bit 0 (LSB) = 0	Disable Supply Voltage Control
Bit 0 (LSB) = 1	Enable Supply Voltage Control
Bit 1 = 0	Do not store bit 0 setting statically
Bit 1 = 1	Store bit 0 setting statically
Bit 2 = 0	Enable both UART and SPI
Bit 2 = 1	Disable UART
Bit 3 = 0	Do not store bit 2 setting statically
Bit 3 = 1	Store bit 2 setting statically

When this command is sent with no payload, it will return a response including a one byte payload, which represents the state that the system is currently in:

RESPONSE PAYLOAD VALUE	DESCRIPTION
Bit 0 (LSB) = 0	Supply Voltage Control disabled
Bit 0 (LSB) = 1	Supply Voltage Control enabled
Bit 1	Not used
Bit 2 = 0	Both UART and SPI active
Bit 2 = 1	Only SPI active

Response command:

- API_ SUCCESS = Command OK
- API_ FAILURE = Command fail

Note that the CRC of 4 bytes are always added to the 1 byte payload, and that the Index bytes of the command structure are NOT used with this command. Note also that the UART disable command is only present from firmware version1D.

6 CRC calculation

The CRC calculation can be implemented as a table of pre-computed effects to ensure efficiency. The CRC value is 32 bits long. The table is indexed by the byte to be encoded and thus the table contains 256 double words (256 * 32 bits).

The CRC algorithm implementation was initially developed by the University of California, Berkeley and its contributors, but has been changed and somewhat simplified to fit the embedded nature of FPC-AM3. The algorithm uses the CCITT-32 CRC Polynomial.

The source code for the CRC implementation is available on the CD, or as a download from the Fingerprint Cards homepage. It should compile with no or small changes on most environments.

7 Power Management

The FPC-AM3 uses an external crystal with a frequency of 7.3728 Mhz, but this clock is internally multiplied to generate a different frequencies, depending on the current state of the system. The 'Power Save Mode' commands (API_POWER_SAVE_MODE_RAM, and API_POWER_SAVE_MODE_STATIC) allows the user to set the system in two different modes: Full Speed (Power Save Mode=disabled), or Half Speed (Power Save Mode=enabled).

In Full Speed Mode, the following frequencies are used in the processor:

- 7 Mhz during Idle mode (waiting for commands)
- 59 MHz during processing of most commands
- 96 MHz during the most computationally complex commands (enrol, verify, and identify).

In Half Speed Mode, measures have been taken to reduce the overall power, but also to limit the current peaks. The frequencies used in this mode are:

- 7 Mhz during Idle mode (waiting for commands)
- 29 MHz during processing of all commands
- 14 MHz during the most power consuming hardware supported computations (during enrol, verify, and identify).

The following images illustrate the current usage for the command API_VERIFY_RAM in Full Speed vs. Half Speed Mode.

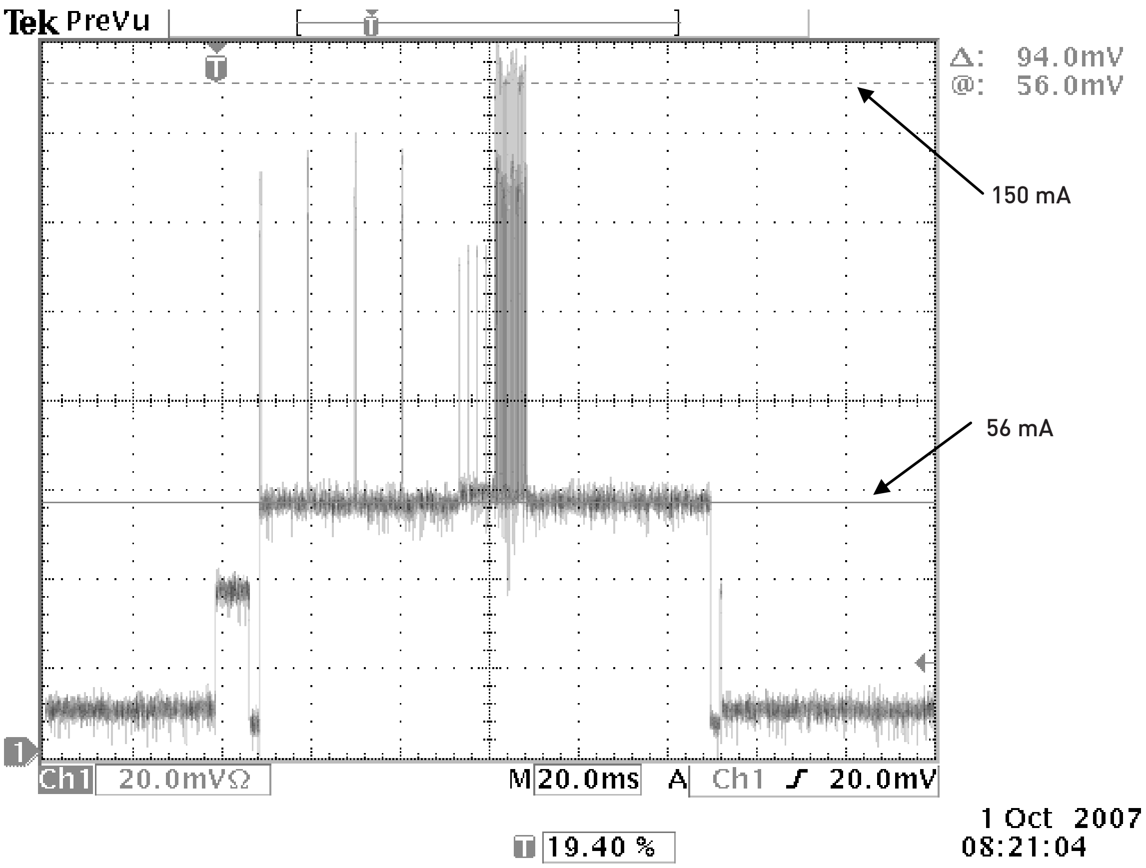


Figure 2.
Supply current during a verification in active full speed

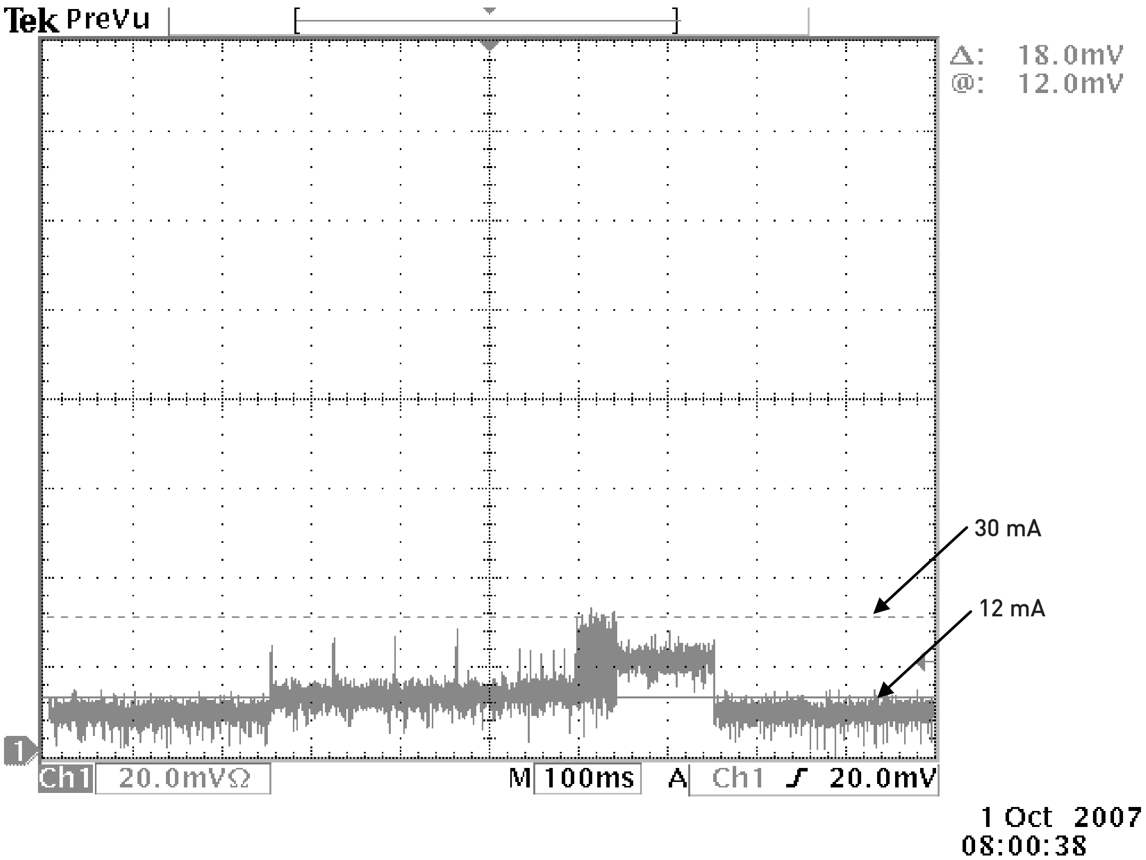


Figure 3
Supply current during a verification in active half speed

8 Boot Mode

At start-up of FPC-AM3, a boot sequence (located in ROM) is executed, which downloads the main application code located in the FLASH memory. If no errors are encountered during this download process, the boot sequence terminates and leaves control to the main application. This is the default behaviour, which typically always should occur in the standard set-up.

If the main application code is corrupt in some way (i.e. FLASH memory is not present or damaged, or a previous upgrade command was terminated in an uncontrolled way), the boot sequence instead waits for input over the UART/SPI host interface. (Default host UART speed is 9600 baud, and maximum SPI speed is 1 MBit/s.)
In this mode, FPC-AM3 only handles the following commands (the communication protocol is identical to what is described in the earlier sections):

- API_RUN_SUPPLIED_PRG (Command code: 0xE6)
- API_INIT_SFR_REGS (Command code: 0xE7)

All other commands receive the response API_APPL_CRC_FAILED or API_SYS_CRC_FAILED, depending on which part of the normal system startup that failed (Application or System).

With the API_RUN_SUPPLIED_PRG-command, the user can send the main application code or the FLASH Installer application as a payload. Both applications are provided by Fingerprint Cards. The Flash Installer application is used to program a system using Flash memory with the main program before its first usage.
Return values are:

- API_ SUCCESS = Download of program successful – New app is launched.
- API_ CRC_FAILED = Download of program failed – Hardware reset required.

With the API_INIT_SFR_REGS-command, the user can modify some internal registers of FPC-AM3, to change the UART speed. Note that this command is optional, and it has no affect on the SPI interface. Sending a payload of size two bytes, according to the following table, changes the bootmode baudrate. Note that 4 CRC bytes are necessary, but their actual values are ignored.

PAYLOAD BYTES	BAUD RATE
0xAA 0xE8	9600 (factory default)
0xAA 0xF0	14400
0xAA 0xF4	19200
0xAA 0xF8	28800
0xAA 0xFA	38400
0xAA 0xFC	57600
0xAA 0xFD	76800
0xAA 0xFE	115200

Return values are:

- API_ SUCCESS = Internal registers adjusted.
- API_ FAILURE = Process failed.

Note: The response is sent *before* the actual baud rate is changed.

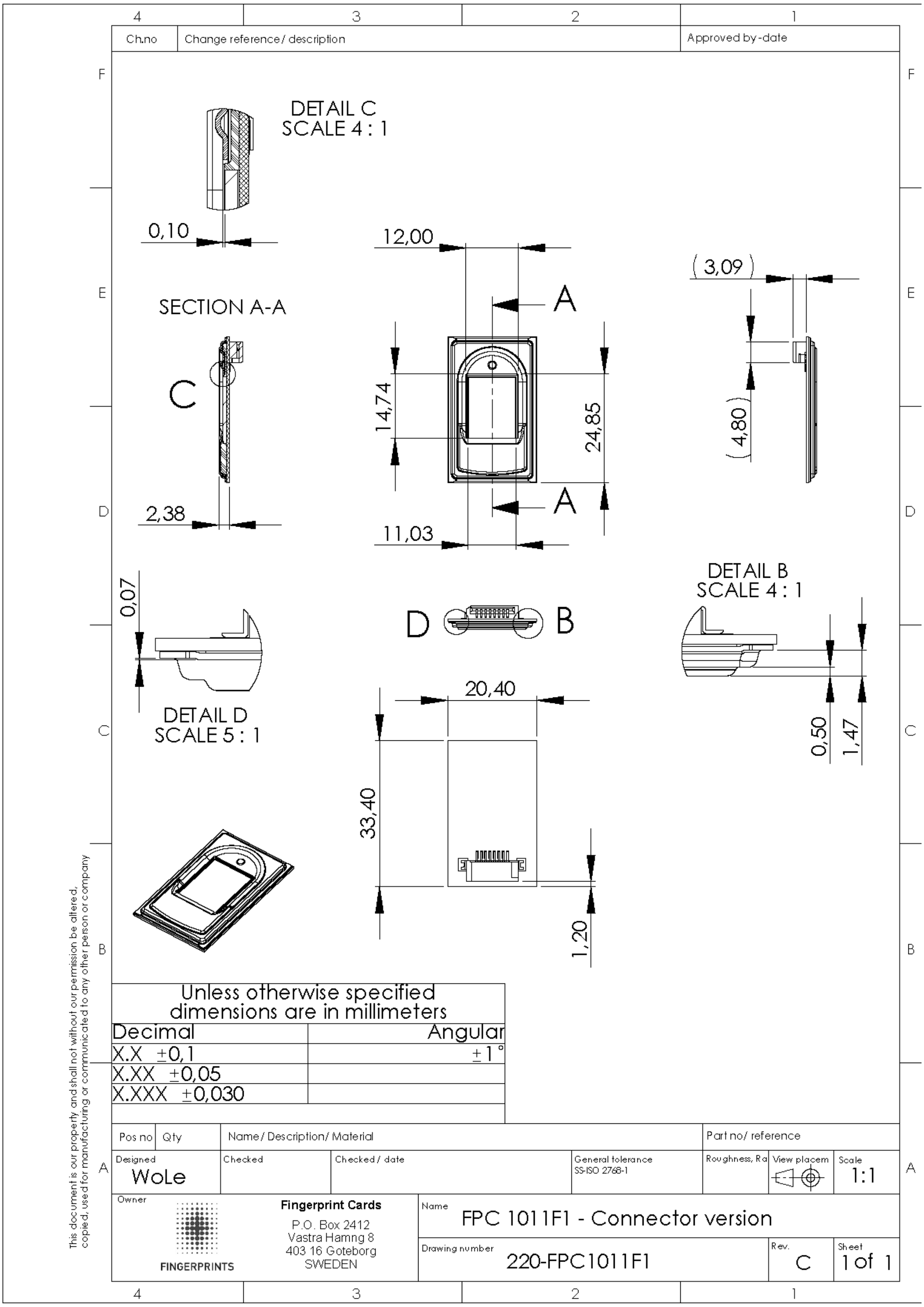


Figure 4

Mechanical drawing FPC1011F Area sensor. Drawing is not to scale.

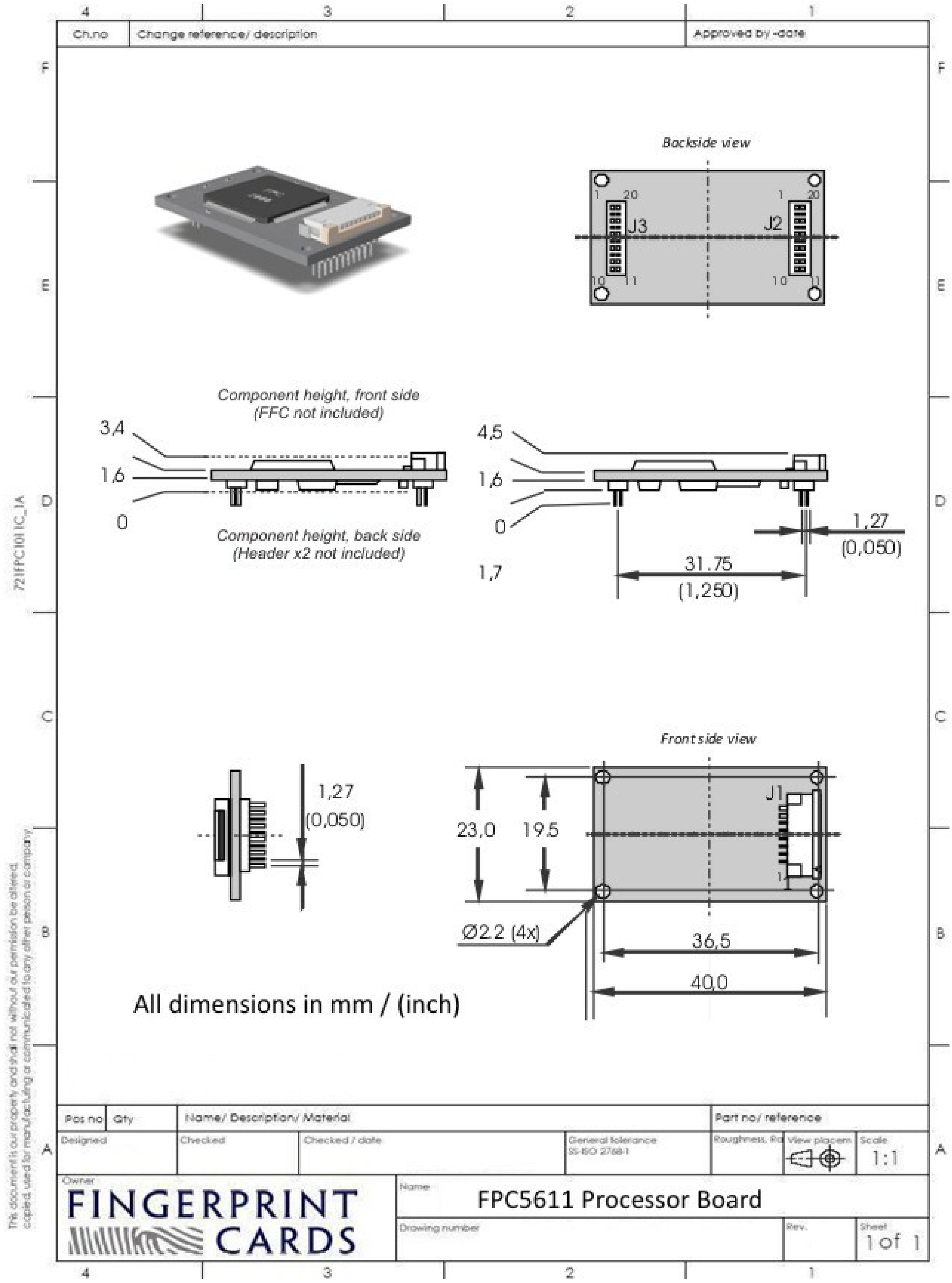


Figure 5
Mechanical drawing FPC5611 Processor board. Drawing is not to scale.

9 Hardware Interface

9.1 Package details

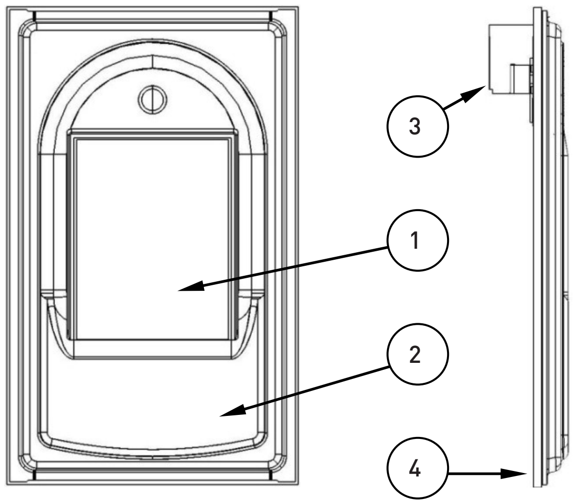


Figure 6
FPC1011F Device overview

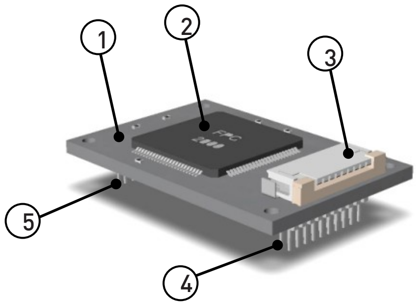


Figure 7
FPC5611 Device overview

ITEM	DESCRIPTION
1	FPC1011 fingerprint area sensor chip
2	Drive electrode, bezel
3	Flex film connector: 8 pin, 1 mm pitch Molex / 0528520870 / low insertion force
4	BT substrate

Table 8
FPC1011F Package details

ITEM	DESCRIPTION
1	FR4 substrate, black, 1.6 mm
2	FPC2020 processor ASIC
3	ZIF connector, 8 pin, 1.0 mm pitch: J1 e.g. Molex 52207-0890
4	Dual header, 20 pin, 1.27 mm (0,05”) pitch: J2-1
5	Dual header, 20 pin, 1.27 mm (0,05”) pitch: J3-1

Mating SMD socket: e.g. Harwin M50-3151042
Mating ribbon cable: e.g. Samtec FFSD-10-D-02.00-01-N

Table 9
FPC5611 Package details

9.2 Pin assignment

PIN	SIGNAL NAME	DESCRIPTION
J3:1	GND	Signal ground
J3:2	GND	Signal ground
J3:3	GND	Signal ground
J3:4	-	*Future functionality*
J3:5	-	*Future functionality*
J3:6	-	*Future functionality*
J3:7	-	*Future functionality*
J3:8	-	*Future functionality*
J3:9	-	*Future functionality*
J3:10	-	*Future functionality*
J3:11	RST_N	System reset, active low
J3:12	Flash_Reset	Resest signal to flash
J3:13	Sensor_IRQ	Interrupt, Active high
J3:14	-	*Future functionality*
J3:15	Host_SCLK	SPI clock
J3:16	Host_SCN	Selects between UART and SPI "0" = SPI Mode, "1" = UART
J3:17	Host_TX	Serial data output SPI or UART
J3:18	Host_RX	Serial data input SPI or UART
J3:19	VDD	Power supply 2.5 - 3.3 V
J3:20	VDD	Power supply 2.5 - 3.3 V

PIN	SIGNAL NAME	DESCRIPTION
J2:1	GND	Signal ground
J2:2	GPIO	*Future functionality*
J2:3	GPIO	*Future functionality*
J2:4	GPIO	*Future functionality*
J2:5	GPIO	*Future functionality*
J2:6	GPIO	*Future functionality*
J2:7	GPIO	*Future functionality*
J2:8	GPIO	*Future functionality*
J2:9	GPIO	*Future functionality*
J2:10	GND	Signal ground
J2:11	GND	Signal ground
J2:12	S_GND	Sensor ground
J2:13	S_CS_N	Chip select, active low
J2:14	S_SI	SPI data input
J2:15	S_GND	Sensor ground
J2:16	S_SCK	SPI clock input
J2:17	S_RST_N	Sensor reset, active low
J2:18	S_VDD	Sensor power supply
J2:19	S_SO	SPI data output
J2:20	GND	Signal ground

Table 9
FPC5611 Pin configuration

Note:
1) Several pins (J3:4 - J3:10, J3:14 , J2:2 - J2:9) are reserved for future functionality and should not be connected.
2) An alternative sensor interface is available on pin J2:12 - J2:19. This feature facilitates a remote sensor connection if this is more convenient, e.g. the ZIF connector can be placed on the opposite side of the main board. Unless these pins are used instead of the topside connector (J1), they are not necessary to connect.

9.3 Hardware control pins

- RST_N = Reset signal to FPC-AM3
- Flash_Reset = Reset signal to the flash circuit
- Sensor_IRQ = Interrupt pin is pulled high for system wake up, when module is placed in SLEEP MODE
- Host_SCN = Select signal between UART and SPI could also be used as wake up signal if pulled low
- Host_RX = Serial data input for SPI or UART could also be used as wake up signal if pulled low

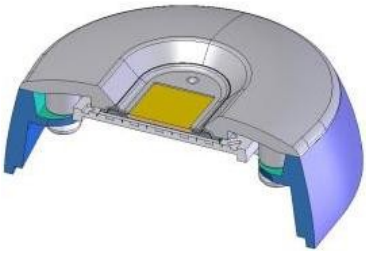
The Wake up signals Sensor_IRQ, Host_SCN and Host_RX should be released when not used. The system is changing the internal frequency in FPC-AM3 to optimize the performance during fingerprint processing and this will not work if any of the three wakes up signals are active.

Pulling both RST_N and Flash_Reset low at the same time will generate a system reset of both FPC2020 and the flash. It is important that the Flash_Reset signal is released before or at least no later than the RST_N signal. If the Flash_Reset signal is released after the RST_signal the normal boot sequence from flash will be stopped and the system will wait for a program boot from one of the serial host interfaces, UART or SPI. Normally this should not be necessary. But if an unforeseen hardware or software error occur it could be vital. It also gives the possibility to start up a temporary program from the host interface while the original program is intact in the flash memory

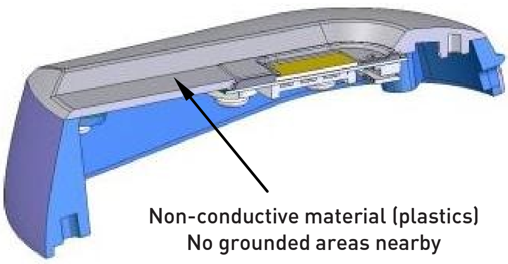
10 Application Information

10.1 Sensor integration

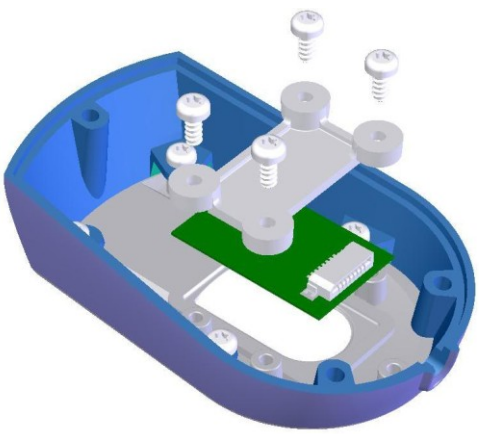
Avoid galvanic contact
Thanks to the conductive frame, containing micro-ergonomics, a smooth transition to exterior mechanics can easily be obtained (example below).



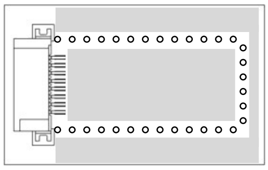
Note that the sensor and its drive electrode (frame) must be mounted in such way that electrical insulation to adjacent conductive surfaces is achieved. It is also recommended to avoid grounded surfaces nearby the drive electrode, since this might interfere with sensor operation.



Proper mechanical support
The best way to ensure a solid sensor mount is to apply a stable, non-conductive, support to the backside of the sensor component.



On the FPC1011F1 package this non-conductive support can preferably be attached to the entire backside area, except for the connector.



10.2 Board assembly

The processor board is furnished with two dual row pin headers. These miniaturized connectors facilitate a number of mounting alternatives. For instance, the processor board can be attached directly to the main board using a board-to-board connector.

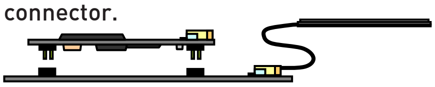


Figure 13
Board-to-board and external flex connector

Above, the sensor is connected directly to the main board and the processor board is treated as an ordinary component.

Keep in mind that the separate flex connector **cannot** be placed too far away from the processor board. Less than **200 mm** is recommended.

By selecting sockets with proper height, the space underneath the processor board can also used for placing components on the main board. Required clearance below the processor board is **2.4 mm**.

Naturally the on-board flex connector (FFC) can be used as well if this is more convienient.

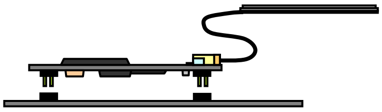


Figure 14
Board-to-board and on-board flex connector

If remote assembly is preferred, a standard high pitch ribbon cable can be used. Only the left header (J3) needs to be connected.

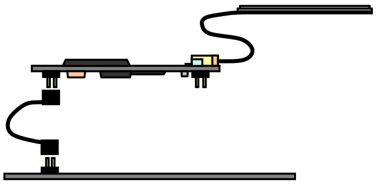


Figure 15
Ribbon cable and on-board flex connector

10.3 Document revision history

REVISION	DATE	CHANGE	Firmware version
A	2008-09-03	First version	Version 1D
B	2009-03-02	General updates	Version 1E

10.4 Product version

The processor board (FPC5611) of the module FPC-AM3 is marked with a label showing the production code.

Example: `0815/0P1B1E`

Description:
```
YYWW/AABBCC
YY = Batch info, production year
WW = Batch info, production week
AA = Production version
BB = Hardware revision
CC = Firmware version
```

Note. Firmware version states the factory default version, i.e. the version of the firmware programmed at manufacturing. The marked firmware version is valid up to delivery from Fingerprint Cards. The firmware can be updated at any time at later stage using the firmware update command.

Minor changes in the firmware may only result in a new firmware release, but no new revision of the product specification. All major software changes will result in an updated product specification. A major change in the firmware will result in stepping the number in the firmware version while a minor change will result in stepping the letter in the firmware version.

10.5 Ordering information

The FPC-AM3 module consists of an area sensor (FPC1011F) and a processor board (FPC5611).

Product name	Description	Order quantity
FPC-AM3	Area sensor module	20 pcs per tray, Minimum order quantity (MOQ) is 60 pcs

10.6 Contact information

Main office
Fingerprint Cards AB
P.O. Box 2412
SE-403 16 Göteborg
Sweden

Visiting address
Västra Hamngatan 8

Telephone
+46 (0)31 60 78 20

Fax
+46 (0)31 13 73 85

Web site
www.fingerprints.com

E-mail
sales@fingerprints.com
tech@fingerprints.com