HTPA SPI[HP] Module Transferprotocol

Rev.1 2012.04.17 Fg

SPI Interface:

SCK-Frequencies:

HTPA8x8: 5 MHz
HTPA32x31: 10 MHz

Protocol Specifications:

Data format:	8 data bits
Frame Sync:	None
Module-Selection:	$\overline{SS}$-Pin
Clock Edge Select:	Serial output data changes on transition from idle to active clock state
SPI Data Input Sample Phase:	Data sampled on transition from active to idle clock state
Clock Polarity:	Idle State is low level, active is high level.

Electrical Specifications:

VDD:	Supply (+3.3V DC)
SPI Transmit/Receive:	TTL
VSS	GND
Power Supply:	**3.3** VDC +/- 5%, 300mA
IDD (Idle mode)	75 mA
IDD (Operating mode)	135 mA

Pinout:

No.	Function	Type
1	VDD	Power
2	VDD	Power
3	VSS	Power
4	VSS	Power
5	SS#	Input
6	UTX/SDO **	Output
7	URX/SDI **	Input
8	SCK	Input
9	CTX*	Output
10	CRX*	Input

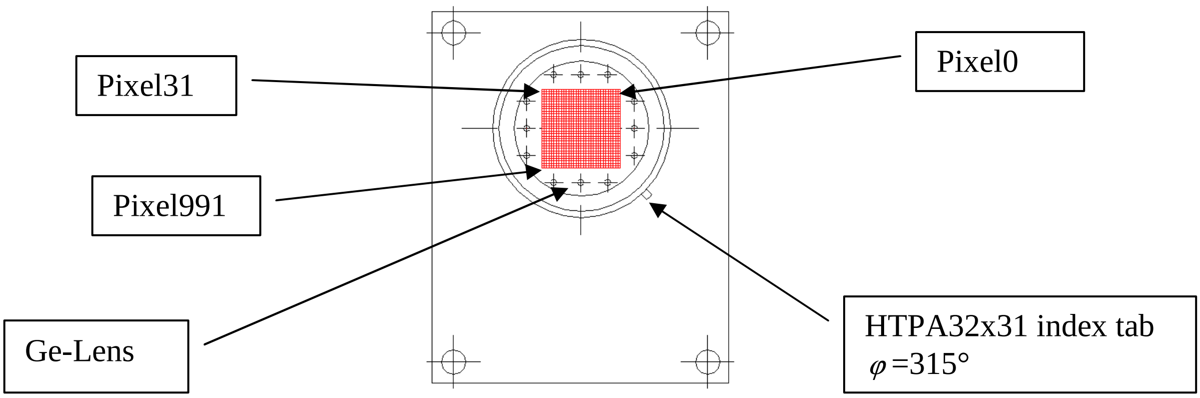
* Pins planned for CAN Bus.
** URX/UTX is used for the UART Module

SDI/SDO/SS#/SCK pins support the Module with SPI Interface.

Connector layout: see drawings.

HTPA32x31 SPI[HP] Module Optical Orientation of Pixels:

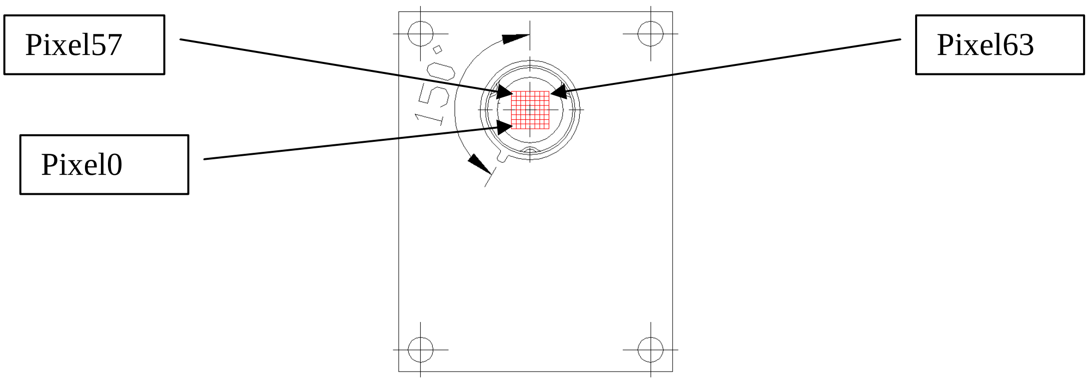
(Physical, lens system may mirror the image)



Top View

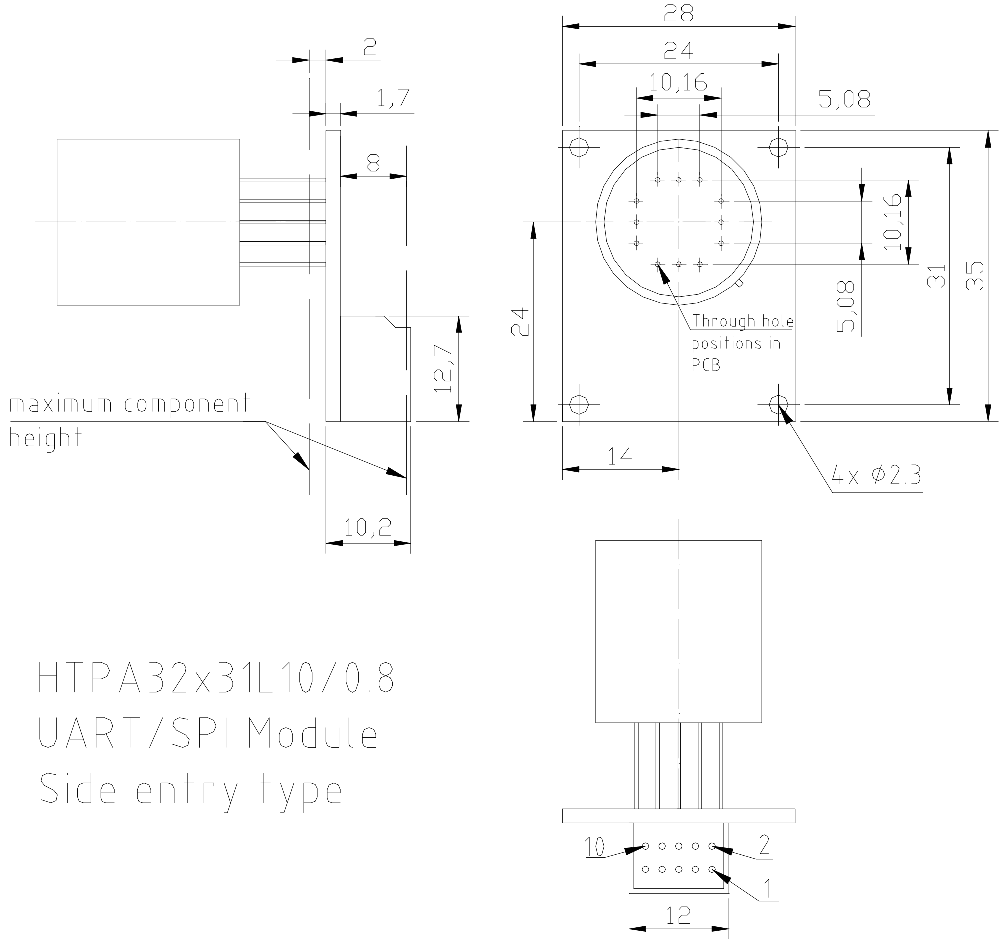
HTPA8x8 SPI[HP] Module Optical Orientation of Pixels:

(Physical, lens system may mirror the image)



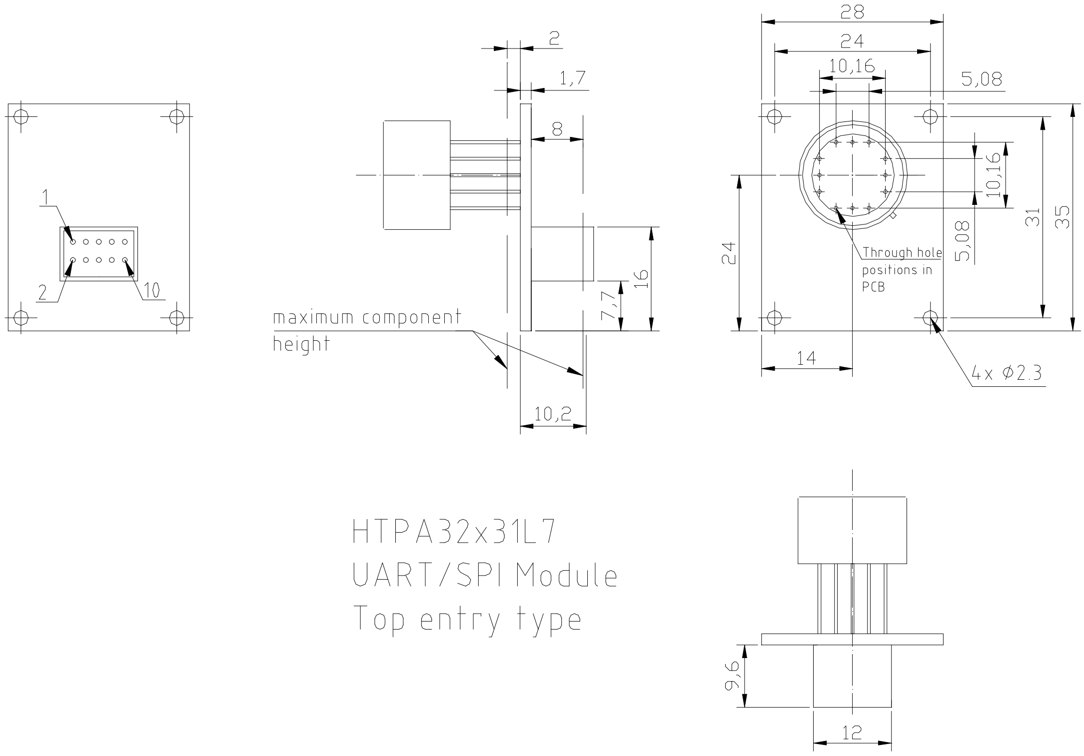
Top View

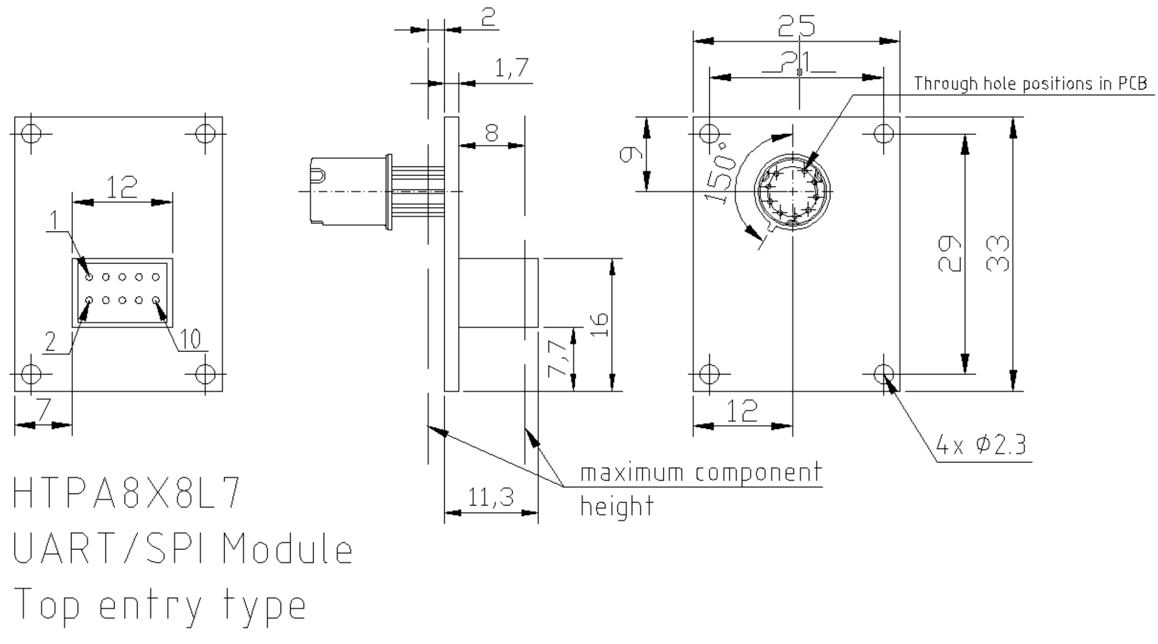
Module dimensions:



In these drawings always only one optic option is shown for each HTPA type. If another optic is built in, refer to the datasheet of the HTPA sensor. PCB size, pin length of the HTPA and positions on the PCB remain the same, only the cap of the HTPA changes.

Module dimensions (continued):

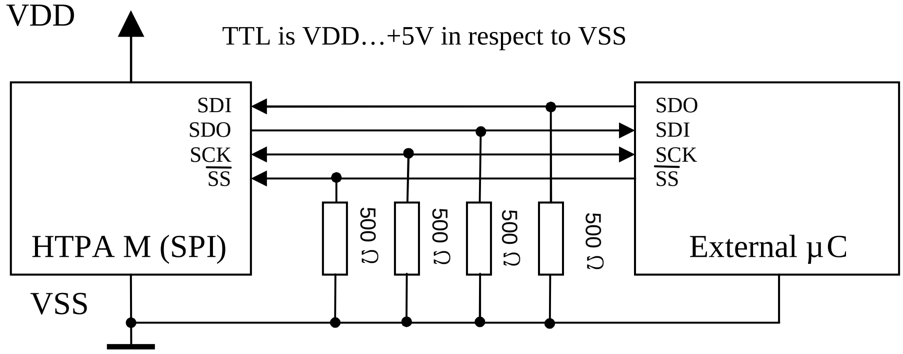




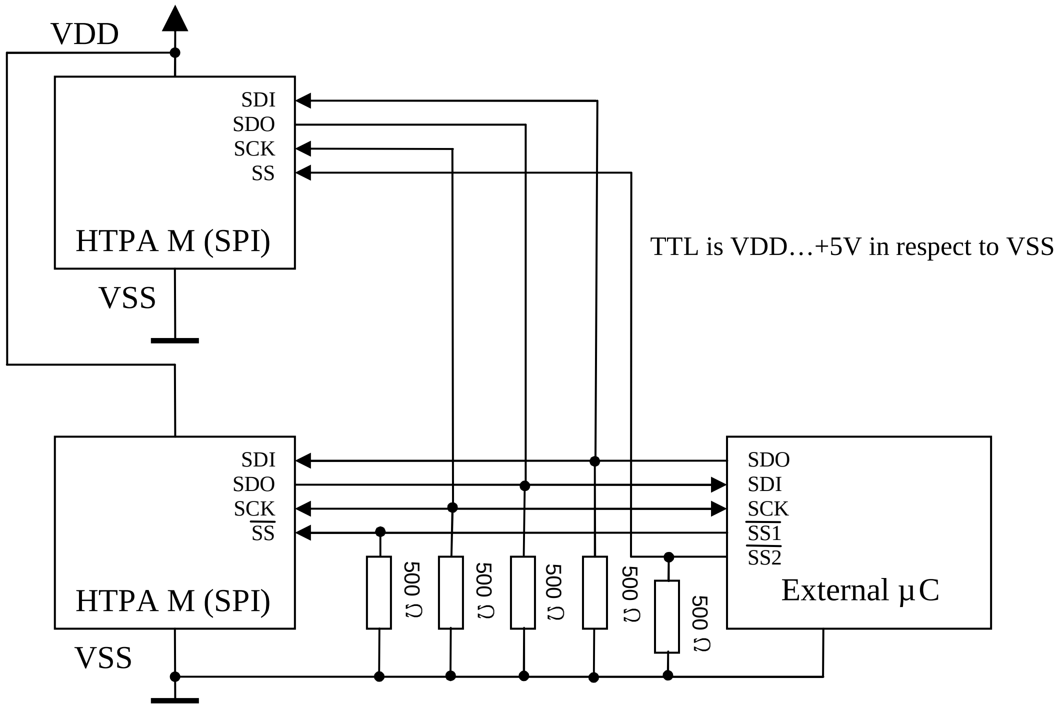
In these drawings always only one optic option is shown for each HTPA type. If another optic is built in, refer to the datasheet of the HTPA sensor. PCB size, pin length of the HTPA and positions on the PCB remain the same, only the cap of the HTPA changes.

Electrical Connections:

Single Module:



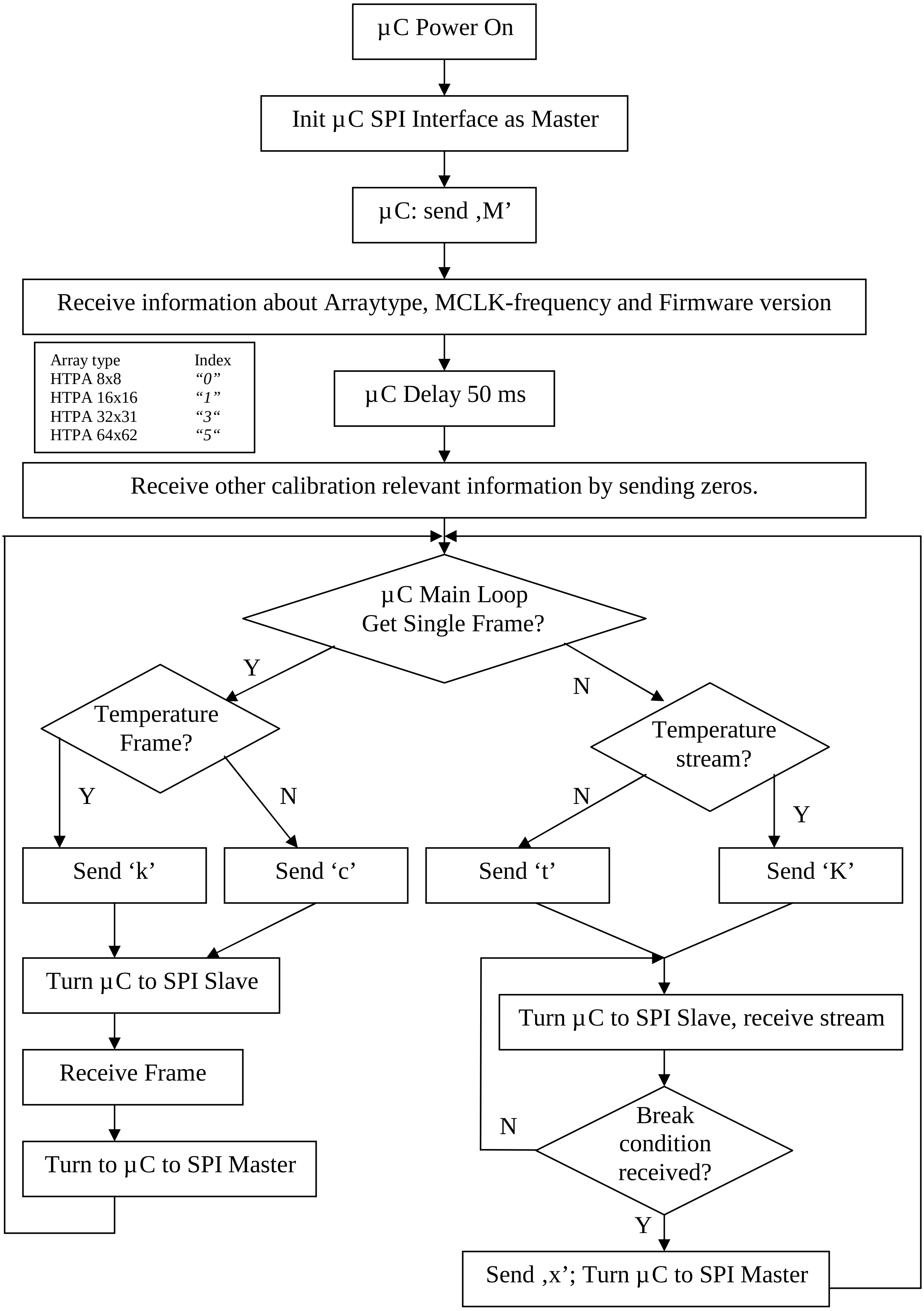
Multiple Modules (preliminary):



SCK is bidirectional because HTPA M (SPI) becomes Master if streaming is in progress. Streaming can be aborted any time by sending any command (preferable ‘x’) or by toggling $\overline{\text{SS}}$ line.
In case of multiple devices only one slave must be selected. Selection of multiple slaves will result in collisions on SPI bus and maybe in damage of the devices.

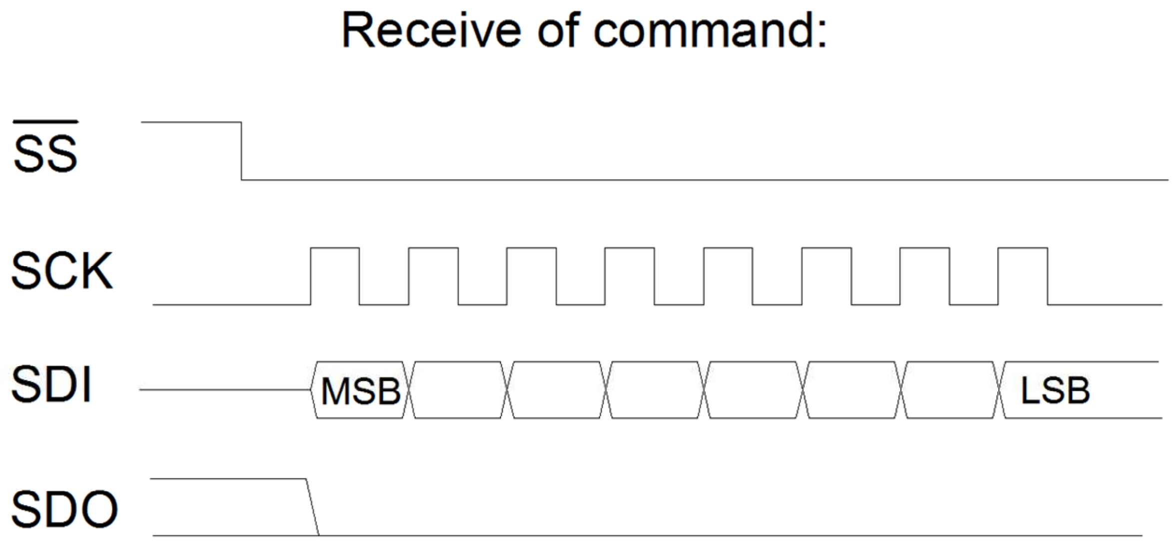
Communication and Timings:

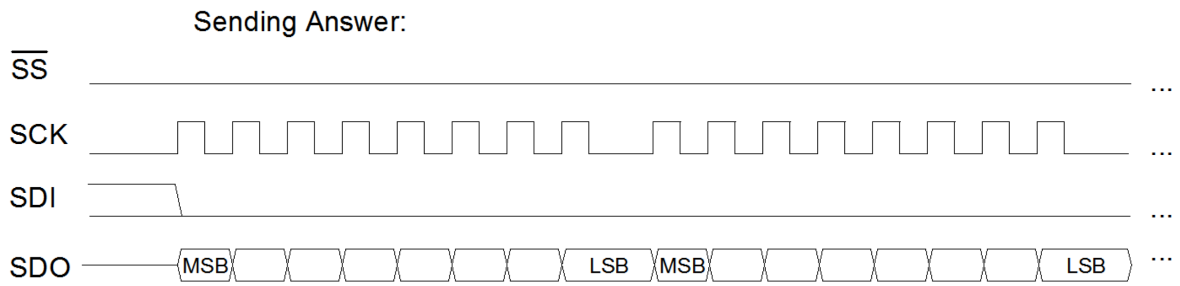
Proposed flow chart of communication. (External controller is refered as µC, HTPA module as HTPA since Master direction toggles)



Communication and Timings (continuation):

SPI data transfer. For SCK frequencies see page 1.



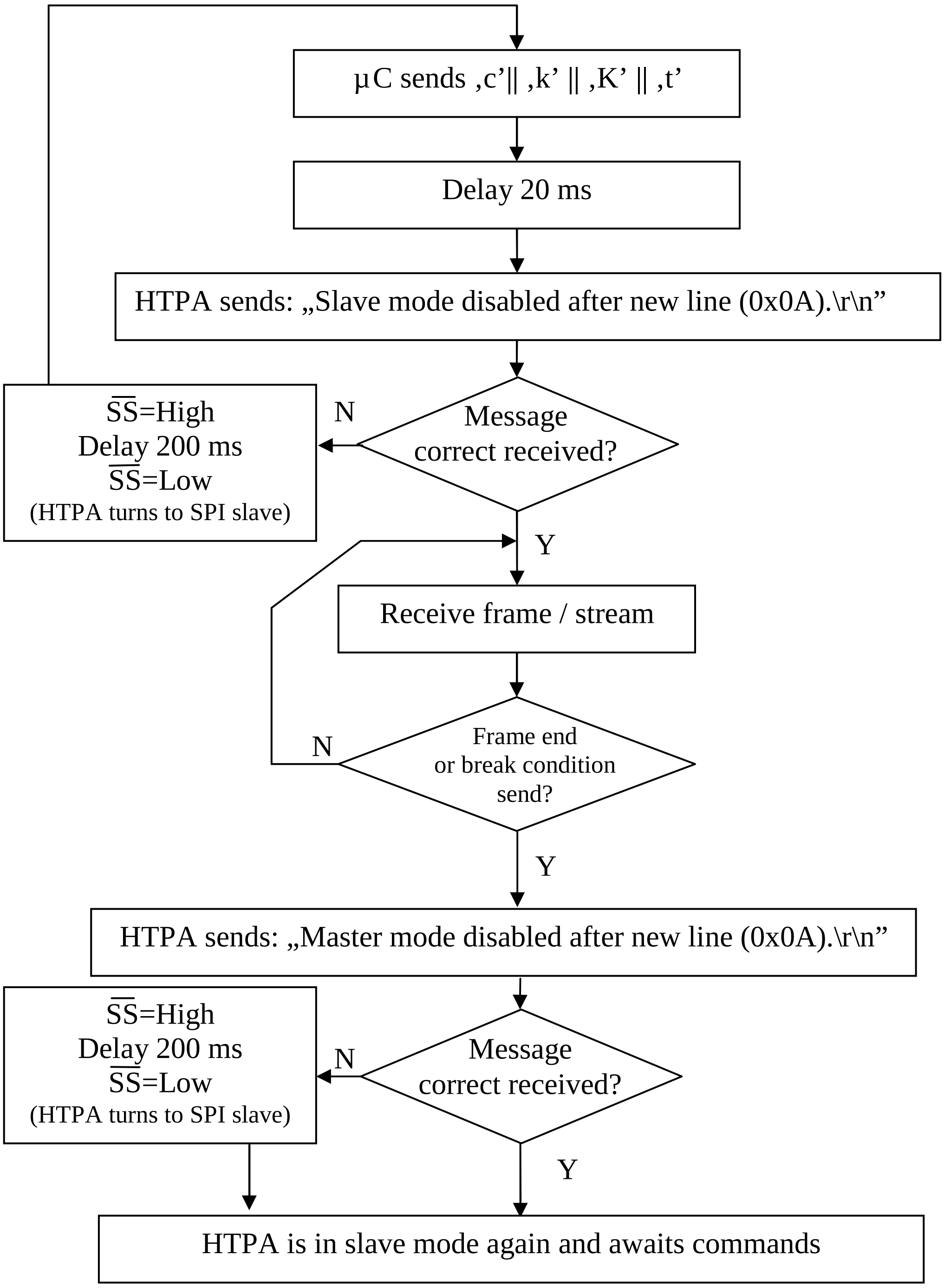


If the module sends an ASCII answer, each line is closed with the sequence of CR and LF (carriage return [0x0D]; line feed [0x0A]).
A single line answer is sent at the receive of following commands:

‘a’/’A’/’b’/’B’/’f’/’F’/’g’/’G’/’J’/’l’/’m’/’o’/’O’/’q’/’Q’/’$’/’y’/’Y’/’z’/’Z’

Communication and Timings (continuation):

Gathering of Frames and streams. Module toggling to SPI Master and back to Slave:
(External controller is refered as µC, HTPA module as HTPA since Master direction toggles)



Communication commands:

Communication via Terminal / UDP				
Sent Char	**HTPA8x8**	**HTPA16x16**	**HTPA32x31**	**Result/Received message**
'a'	X	X	X	Decreases the operating frequency of the array
'A'	X	X	X	Increases the operating frequency of the array
'b'	X	X	X	Measure VDD (referenced to VREF1225)
'C'	X	X		Capture single voltage frame. Use ADC of ASIC. Output via ASCII if sent via UART, binary if sent via UDP.
'c'	X	X	X	Capture single voltage frame. Use ADC of µC. Output via ASCII if sent via UART, binary if sent via UDP.
'd'/'D'	X	X		Toggle POR_N
'f'	X	X	X	Toggle Resetbit
'F'	X	X		Analog operating point is at start of AD-range, only positive signals convertable
'G'	X	X		Analog operating point is in the middle of AD-range, positive and negative signals convertable
'g'	X	X		Analog operating point is at end of AD-range, only negative signals convertable
'h'	X	X	X	pushes binary EEDATA out
'i'			X	Read single voltage frame. Output in ASCII format. Serial order: Pixeldata[K*10], el. Offsets, Ambient Temperature
'I'			X	Read single temperature frame. Output in ASCII format. Serial order: Pixeldata[K*10], el. Offsets, Ambient Temperature
'J'	X	X	X	Toggle Amplification
'k'	X	X	X	Read single temperature frame. Output in binary format.
'K'	X	X	X	send continous binary temperature datastream(µC-ADC)[K*10] Output of a complete cycle in this order:

HTPA 8x8 and HTPA16x16: Pixel0,Pixel1, ….PixelX, el.Offset0, el.Offset1,…, el.OffsetY,PTAT0,PTAT1,…,PTATZ
HTPA32x31: see Table2.
For a detailed Description of the serial order see Table2.

16x16 Array:
X=255; Y=7; Z=7

8x8 Array:
X=63; Y=4; Z=4

One dataset has exactly 2 bytes: first the low-Byte is send, then the high-byte. Each Dataset contains the measured Temperature in Kelvin*10. The first 4 datasets *el.Offset0…el.Offset3* after the last Pixel voltage *PixelX* transmit additional the current VDD in the MSB's:

VDD and TAmb for HTPA8x8 and HTPA16x16:

Dataset	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	…	Bit1	Bit0
elOff0	MSB VDD	…	…	Bit12 VDD	MSB elOff0	…	…	…	LSB elOff0
elOff1	Bit11 VDD	…	…	Bit8 VDD	MSB elOff1	…	…	…	LSB elOff1
elOff2	Bit7 VDD	…	…	Bit4 VDD	MSB elOff2	…	…	…	LSB elOff2
elOff3	Bit3 VDD	…	…	LSB VDD	MSB elOff3	…	…	…	LSB elOff3

The Sensor temperature is available in the datasets after *el.Offset3*:

Dataset	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	…	Bit1	Bit0
elOff3+1	MSB TAmb	…	…	Bit12 TAmb	MSB elOff3+1	…	…	…	LSB elOff3+1
elOff3+2	Bit11 TAmb	…	…	Bit8 TAmb	MSB elOff3+2	…	…	…	LSB elOff3+2
elOff3+3	Bit7 TAmb	…	…	Bit4 TAmb	MSB elOff3+3	…	…	…	LSB elOff3+3
elOff3+4	Bit3 TAmb	…	…	LSB TAmb	MSB elOff3+4	…	…	…	LSB elOff3+4
elOff3+5	0	0	0	0	MSB elOff3+5	…	…	…	LSB elOff3+5

Sent Char	HTPA8x8	HTPA16x16	HTPA32x31	Result/Received message
'l'	X	X	X	Get Ambient Temperature (Calculates the Ambient Temperature from the **last measured** Frame)
'm'	X	X	X	Toggle usage of µC-Buffer for el. Offsets (Stack depth = 64 for HTPA8x8 and HTPA16x16; Stack depth = 32 for HTPA32x31)
'M'	X	X	X	Shows current and calibration settings. Device prints the following stream: **"HTPA series responsed! I am Arraytype X"** Possible values for X: "0"=HTPA8x8, "1"=HTPA16x16, "3"=HTPA32x31 **"Firmware v.X.XX written by B.Forg; Heimann Sensor GmbH; YYYY-MM-DD"** Version information. **"I am running on XXXX.X kHz"** Actual MCLK-setting in kHz **"Amplification is X" Actual set amplification.** Possible strings for X: "low" or "high" **"MAC-ID: X IP: Y DevID: Z\r\n"** (Only Ethernet devices show a MAC-ID, DevID is shown in any case) X= MAC-ID of the device, i.e. "00.97.FF.00.10.08"; Y=current IP of the device, Z=user setable ID, range 00000…65535 **"PIXCvsTA *X* , BFL3 *X* , F8_14 *X* , THvsTA *X* IGNORE_ELOFF *X* ELOFF32 *X* SBY *Y* FC *X* EXP *Z* "**

Table1: Control Characters

Table1, continuation:

Communication via Terminal / UDP				
Sent Char	**HTPA8x8**	**HTPA16x16**	**HTPA32x31**	**Result/Received message**
'o'		X	X	Use external reference voltages
'O'		X	X	Use internal reference voltages
'q'/'Q'	X	X	X	Allow Changes (required for Calibration)
't'	X	X	X	Continuous binary voltage data of the µC-ADC is transmitted. Output of a complete cycle in this order: *HTPA 8x8 and HTPA16x16: Pixel0,Pixel1, ….PixelX, el.Offset0, el.Offset1,…, el.OffsetY,PTAT0,PTAT1,…,PTATZ* *HTPA32x31: see Table2.* ***For a detailed Description of the serial order see Table2.*** 16x16 Array: X=255; Y=7; Z=7 8x8 Array: X=63; Y=4; Z=4 One dataset has exactly 2 bytes: first the low-Byte is send, then the high-byte. Each Dataset contains the ADC-Data in digits and The first 4 datasets *el.Offset0…el.Offset3* after the last Pixel voltage *PixelX* transmit additional the current VDD in the MSB's: **VDD for HTPA8x8 and HTPA16x16:**

Dataset	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	…	Bit1	Bit0
elOff0	MSB VDD	…	…	Bit12 VDD	MSB elOff0	…	…	…	LSB elOff0
elOff1	Bit11 VDD	…	…	Bit8 VDD	MSB elOff1	…	…	…	LSB elOff1
elOff2	Bit7 VDD	…	…	Bit4 VDD	MSB elOff2	…	…	…	LSB elOff2
elOff3	Bit3 VDD	…	…	LSB VDD	MSB elOff3	…	…	…	LSB elOff3

Sent Char	HTPA8x8	HTPA16x16	HTPA32x31	Result/Received message
'T'	X	X		Continuous binary data of the ASIC-ADC is transmitted. Output order is equal to ‘t’.
'u'	X	X		Continuous binary data of the ASIC-ADC is transmitted. PTAT-Voltages are sampled with the uC-ADC. Output order is equal to ‘t’.
'U'	X	X		Capture single frame. Use ADC of ASIC. Output via ASCII. PTAT-Voltages are sampled with the uC-ADC.
'v'	X	X	X	Announce IP (Only Ethernet devices)
'V'	X	X	X	Device awaits control message (only non-Ethernet devices)
'w'	X	X	X	shows Calibration-constants
'W'	X	X	X	Calibration. ATTENTION! Old Dataset cannot be restored!
'x'	X	X	X	Stops Stream without prompt.
'X'	X	X	X	Stops Stream by sending “STOP!\r\n”
'y'	X	X	X	switch off ASIC-Supply (5V)
'Y'	X	X	X	switch on ASIC-Supply (5V)

Table1 (continuation): Control Characters

Serial order in Frame:

HTPA8x8 Temperature Mode	
Dataset	Value
0	Temperature of Pixel0 in K*10
1	Temperature of Pixel1 in K*10
2	Temperature of Pixel2 in K*10
3	Temperature of Pixel3 in K*10
…	…
63	Temperature of Pixel63 in K*10
64	4 bits of VDD and elOff0 in digits (refer to Table1)
65	4 bits of VDD and elOff1 in digits (refer to Table1)
66	4 bits of VDD and elOff2 in digits (refer to Table1)
67	4 bits of VDD and elOff4 in digits (refer to Table1)
68	4 bits of TAmb and PTAT0 in digits (refer to Table1)
68	4 bits of TAmb and PTAT1 in digits (refer to Table1)
68	4 bits of TAmb and PTAT2 in digits (refer to Table1)
68	4 bits of TAmb and PTAT3 in digits (refer to Table1)

HTPA8x8 Voltage Mode	
Dataset	Value
0	absolute Voltage of Pixel0 in in digits
1	absolute Voltage of Pixel1 in in digits
2	absolute Voltage of Pixel2 in in digits
3	absolute Voltage of Pixel3 in in digits
…	…
63	absolute Voltage of Pixel63 in in digits
64	4 bits of VDD and elOff0 in digits (refer to Table1)
65	4 bits of VDD and elOff1 in digits (refer to Table1)
66	4 bits of VDD and elOff2 in digits (refer to Table1)
67	4 bits of VDD and elOff4 in digits (refer to Table1)
68	4 bits of TAmb and PTAT0 in digits (refer to Table1)
68	4 bits of TAmb and PTAT1 in digits (refer to Table1)
68	4 bits of TAmb and PTAT2 in digits (refer to Table1)
68	4 bits of TAmb and PTAT3 in digits (refer to Table1)

HTPA16x16 Temperature Mode	
Dataset	Value
0	Temperature of Pixel0 in K*10
1	Temperature of Pixel1 in K*10
2	Temperature of Pixel2 in K*10
3	Temperature of Pixel3 in K*10
…	…
255	Temperature of Pixel255 in K*10
256	4 bits of VDD and elOff0 in digits (refer to Table1)
257	4 bits of VDD and elOff1 in digits (refer to Table1)
258	4 bits of VDD and elOff2 in digits (refer to Table1)
259	4 bits of VDD and elOff3 in digits (refer to Table1)
260	elOff4 in digits
261	elOff5 in digits
262	elOff6 in digits
263	elOff7 in digits
264	4 bits of TAmb and PTAT0 in digits (refer to Table1)
265	4 bits of TAmb and PTAT1 in digits (refer to Table1)
266	4 bits of TAmb and PTAT2 in digits (refer to Table1)
267	4 bits of TAmb and PTAT3 in digits (refer to Table1)
268	PTAT4 in digits
…	…
271	PTAT7 in digits

HTPA16x16 Voltage Mode	
Dataset	Value
0	absolute Voltage of Pixel0 in in digits
1	absolute Voltage of Pixel1 in in digits
2	absolute Voltage of Pixel2 in in digits
3	absolute Voltage of Pixel3 in in digits
…	…
255	absolute Voltage of Pixel255 in in digits
256	4 bits of VDD and elOff0 in digits (refer to Table1)
257	4 bits of VDD and elOff1 in digits (refer to Table1)
258	4 bits of VDD and elOff2 in digits (refer to Table1)
259	4 bits of VDD and elOff3 in digits (refer to Table1)
260	elOff4 in digits
261	elOff5 in digits
262	elOff6 in digits
263	elOff7 in digits
264	4 bits of TAmb and PTAT0 in digits (refer to Table1)
265	4 bits of TAmb and PTAT1 in digits (refer to Table1)
266	4 bits of TAmb and PTAT2 in digits (refer to Table1)
267	4 bits of TAmb and PTAT3 in digits (refer to Table1)
268	PTAT4 in digits
…	…
271	PTAT7 in digits

HTPA32x31 Temperature Mode	
Dataset	Value
0	Temperature of Pixel0 in K*10
1	Temperature of Pixel16 in K*10
2	Temperature of Pixel1 in K*10
3	Temperature of Pixel17 in K*10
…	…
30	Temperature of Pixel15 in K*10
31	Temperature of Pixel31 in K*10
32	Temperature of Pixel32 in K*10
33	Temperature of Pixel48 in K*10
…	…
991	Temperature of Pixel991 in K*10
992	elOff0 in digits
993	elOff16 in digits
994	elOff1 in digits
995	elOff17 in digits
…	…
1022	elOff15 in digits
1023	elOff31 in digits
1024	least significant 12 bits of VDD
1025	most significant 4 bits of VDD
1026	least significant 12 bits of TAmb
1027	most significant 4 bits of TAmb
1028	no value, ignore
1029	no value, ignore
…	…
1039	no value, ignore
1040	PTAT0 in digits
1041	no value, ignore
1042	PTAT1 in digits
…	…
1053	no value, ignore
1054	PTAT7 in digits
1055	no value, ignore

HTPA32x31 Voltage Mode	
Dataset	Value
0	absolute Voltage of Pixel0 in in digits
1	absolute Voltage of Pixel16 in in digits
2	absolute Voltage of Pixel1 in in digits
3	absolute Voltage of Pixel17 in in digits
…	…
30	absolute Voltage of Pixel15 in in digits
31	absolute Voltage of Pixel31 in in digits
32	absolute Voltage of Pixel32 in in digits
33	absolute Voltage of Pixel48 in in digits
…	…
991	absolute Voltage of Pixel991 in in digits
992	elOff0 in digits
993	elOff16 in digits
994	elOff1 in digits
995	elOff17 in digits
…	…
1022	elOff15 in digits
1023	elOff31 in digits
1024	least significant 12 bits of VDD
1025	most significant 4 bits of VDD
1026	least significant 12 bits of TAmb
1027	most significant 4 bits of TAmb
1028	no value, ignore
1029	no value, ignore
…	…
1039	no value, ignore
1040	PTAT0 in digits
1041	no value, ignore
1042	PTAT1 in digits
…	…
1053	no value, ignore
1054	PTAT7 in digits
1055	no value, ignore

Table2: Serial order

Each dataset consists of a 16 bit value. If a frame consists out of more than one packet, packets are appended. For UART and SPI devices the 16 bit values are transmitted with LSB first.

Control Messages:

In the set of control messages, expressions in angled braces have to be substituted by following strings:

[DEVID] insert 5 digit device ID in ASCII format, i.e. *“00197”* Range: 00000… 65535

Set of control messages:

Message5:	"Set EEPROM data"
Conditions:	**ATTENTION!** Calibration data is overwritten!!!
Result:	Writes the next received packets into EEPROM. Device writes to EEPROM, until EEPROM is completely filled. EEPROM size depends on Device type: HTPA8x8 16384 byte; HTPA32x31 and HTPA64x62: 65536 byte.
Answer:	"Write was successful.\n\r"

Message6:	"Set DeviceID to **[DEVID]**"
Result:	The given Device ID **[DEVID]** is written to EEPROM. This ID is shown on receive of ‘M’. The Device ID can be used for customer specific purposes.
Answer:	"DeviceID changed to **[DEVID]**\r\n"

Temperature calculation:

To get the calibration settings for your device, request a single temperature frame, by sending ‘k’. Now the device automatically loaded the settings which were used during calibration. After the receive of the temperature frame request the actual configuration of the device by sending ‘M’. Store the information in the line:
"PIXCvsTA *X*, BFL3 *X*, F8_14 *X*, THvsTA *X* IGNORE_ELOFF *X* ELOFF32 *X* SBY *Y* FC *X* EXP *Z*". All cursive letters are device dependent constants. Possible strings: *X*="true" or "false", *Y*="1" or "0", *Z* is the string of a 2 digit decimal value, i. e. "3.47". Store those constants, the calculation algorithm is depending on those.
Now load the calibration constants by sending ‘w’. The module will send the calibration information, according to Table 3. Store the coloured constants. All stored constants will be labelled according to the value in “Name of constant”. If there is no coloured mark in the example string and no insertion in “Name of constant”, it is not necessary to store it.

Header of calibration data		
Example String	**Name of constant**	**Unit(s)**
"ASIC-Register @ Calib was 0x402\r\n"		
"Written back 0x402 to ASIC.\r\n"		
"PTAT-gradient 0.569000 dK/dig PTAT-Offset@0V 1973.000000\r\n"	$PTAT_G$, $PTAT_O$	[dK/dig], [dig]
"Ambient 1: 285.2, Ambient 2: 295.2 Ambient 3: 310.2 Ambient 4: 324.3 at Calibration\r\n"	T_{cT1}, T_{cT2}, T_{cT3}, T_{cT4}	[K]
"TObj1: 373.2, TObj2: 373.2 TObj3: 373.2 TObj4: 373.2 at Calibration\r\n"		
"TObjcal1: 285.9., TObjcal2: 297.2 TObjcal3: 311.2 TObjcal4: 323.3 at Calibration\r\n"	T_{cO1}, T_{cO2}, T_{cO3}, T_{cO4}	[K]
"Arraytype is 3\r\n"		
"Nr. Thermaloff(1) PixC(1) Thermaloff(2) PixC(2) Thermaloff(3) PixC(3) Thermaloff(4) PixC(4) ThGrad ThOff\r\n"		

Table3: Header of calibration data

Constant:	**Meaning:**
$PTAT_G$	Gradient of the PTAT circuit in tenths of Kelvin per digits.
$PTAT_O$	Offset of the PTAT circuit in digits.
T_{cT1}	Absolute ambient temperature during calibration of the thermal offsets in calibration point 1.
T_{cT2}	Absolute ambient temperature during calibration of the thermal offsets in calibration point 2.
T_{cT3}	Absolute ambient temperature during calibration of the thermal offsets in calibration point 3.
T_{cT4}	Absolute ambient temperature during calibration of the thermal offsets in calibration point 4.
T_{cO1}	Absolute ambient temperature during calibration of the pixel constants in calibration point 1.
T_{cO2}	Absolute ambient temperature during calibration of the pixel constants in calibration point 2.
T_{cO3}	Absolute ambient temperature during calibration of the pixel constants in calibration point 3.
T_{cO4}	Absolute ambient temperature during calibration of the pixel constants in calibration point 4.

Table4: Meaning of constants (1)

After the header the pixel dependent constants are immediately transmitted. The number of packets (UDP) or lines (UART) depends on the device type and is equal to the number of sensitive pixels. Each packet / line consists of:

Pixelnumber, Thermal1(X), Pixelconstant1(X), Thermal2(X), Pixelconstant2(X), Thermal3(X), Pixelconstant3(X), Thermal4(X), Pixelconstant4(X).

X represents in the above line the corresponding pixel number.

Store all the pixel dependent constants.

Pixel dependent calibration data			
No.	**Example String**	**Name of constant**	**Comments**
0	"0 127 1316732 112 1182251 98 1126390 72 849857"	$Th_1(0)$, $P_1(0)$, $Th_2(0)$, $P_2(0)$, $Th_3(0)$, $P_3(0)$, $Th_4(0)$, $P_4(0)$,	Number of dataset grey marked
1	"1 137 1396731 132 1482251 128 1516391 127 1549867"	$Th_1(1)$, $P_1(1)$, $Th_2(1)$, $P_2(1)$, $Th_3(1)$, $P_3(1)$, $Th_4(1)$, $P_4(1)$,	Number of dataset grey marked
…	…	…	…
N	"N 127 1516732 112 1182251 98 1126390 72 849857"	$Th_1(N)$, $P_1(N)$, $Th_2(N)$, $P_2(N)$, $Th_3(N)$, $P_3(N)$, $Th_4(N)$, $P_4(N)$,	N equals the number of pixels-1. HTPA8x8→N=63 HTPA16x16→N=255 HTPA32x31→N=991

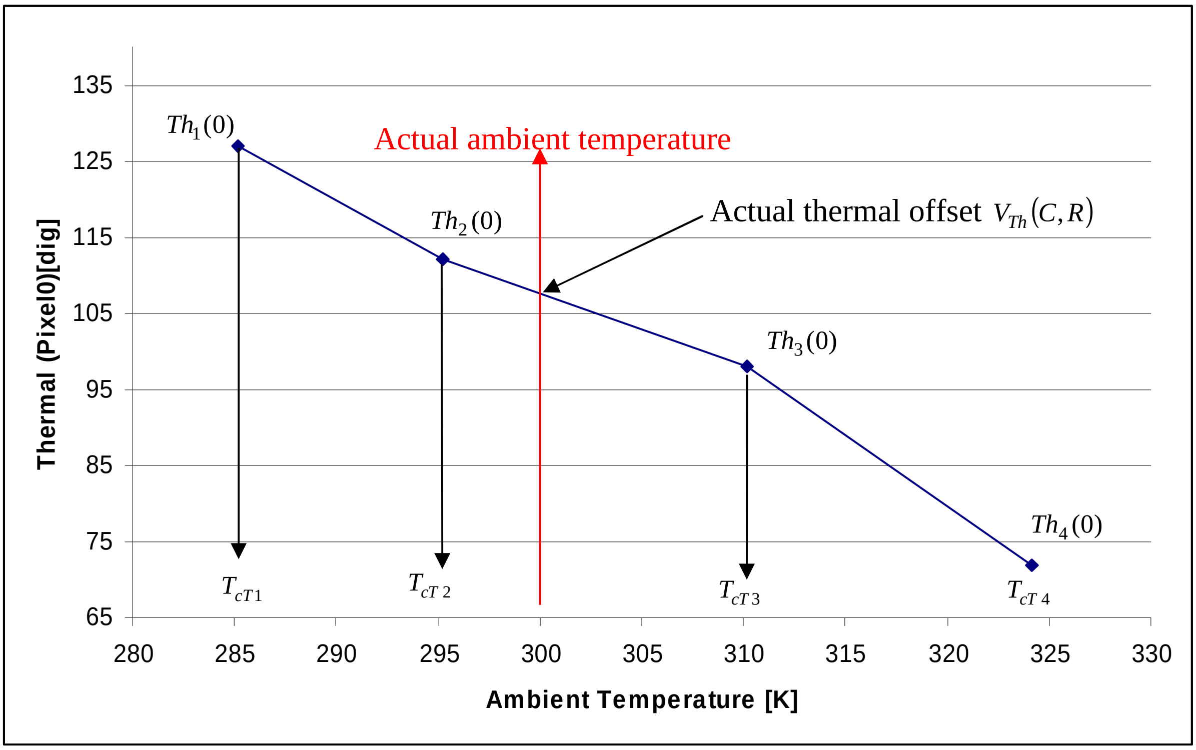
Table5: Pixel dependent calibration data

Constant:	**Meaning:**
$Th_1(X)$	Thermal offset of Pixel X in digits at the thermal calibration point 1 → T_{cT1}
$Th_2(X)$	Thermal offset of Pixel X in digits at the thermal calibration point 2 → T_{cT2}.
$Th_3(X)$	Thermal offset of Pixel X in digits at the thermal calibration point 3 → T_{cT3}
$Th_4(X)$	Thermal offset of Pixel X in digits at the thermal calibration point 4 → T_{cT4}
$P_1(X)$	Pixel constant of Pixel X at the object calibration point 1 → T_{cO1}
$P_2(X)$	Pixel constant of Pixel X at the object calibration point 2 → T_{cO2}
$P_3(X)$	Pixel constant of Pixel X at the object calibration point 3 → T_{cO3}
$P_4(X)$	Pixel constant of Pixel X at the object calibration point 4 → T_{cO4}

Table6: Meaning of constants (2)

The calibration is done with four different ambient temperature calibration points for Thermals and pixel constants, each.
Therefore, it is useful to calculate a gradient and an offset (each for thermal offsets and pixel constants) between two calibration points and to do a linear interpolation between the two calibration points, which are the closest to the actual ambient temperature. This will provide the highest accuracy.
For the following example we took the data of Pixel0, provided in Table 3 and Table 5. The linear interpolation is shown at the example of the thermal offset. Assume an actual ambient temperature of 300 K. Refer to chapter "Ambient temperature" for ambient temperature calculation.

Calculation example for actual thermal offset of Pixel 0 at 300 Kelvin:



To calculate the actual thermal offset $V_{Th}(C,R)$ a linear interpolation is sufficient. *(C,R)* represents the column and row dependency of the value. The same proceeding for the pixel constants is recommended.

Ambient Temperature:

Calculate the ambient temperature by following equation:

$$T_{Amb} = \frac{\sum_{i=0}^{N} PTAT_i}{N+1} \cdot PTAT_G + PTAT_O \text{ [dK]}$$

T_{Amb} is the actual ambient temperature in tenths of Kelvin.
$PTAT_i$ represents the *N* current PTAT-values, refer to Table2. *N* is device dependent.

Object Temperature:

In the followings dependency of the column is displayed as *variable-name (C)*.
Dependency of row and column (pixel dependent) is displayed as *variable-name(C,R)*.

The transmitted absolute pixel voltage consist of the electrical offset of the amplifier $V_{elOff}(C)$, the thermal offset of the pixel $V_{Th}(C,R)$ and the amplified pixel voltage $V_{Pix}(C,R)$:

$$V_{Abs}(C,R) = V_{Pix}(C,R) + V_{Th}(C,R) + V_{elOff}(C)$$

For the HTPA8x8 and the HTPA16x16 the corresponding electrical offset to a pixel voltage can be determined by the modulo-n-check. Divide the pixel number by the number of amplifiers (refer to the datasheet of the sensor), the residue is the corresponding electrical offset. For the HTPA32x31 the column address of the pixel is equal to the number of the electrical offset value.

If your device has a calibration setting of "IGNORE_ELOFF false" it is necessary to subtract the corresponding electrical offset of the amplifier $V_{elOff}(C)$ from the transmitted absolute Voltage of the pixel, else the electrical offset can be set to zero.

IGNORE_ELOFF false→ $V_{Abs}(C,R) - V_{elOff}(C) = V_{Pix}(C,R) + V_{Th}(C,R)$
IGNORE_ELOFF true→ $V_{Abs}(C,R) = V_{Pix}(C,R) + V_{Th}(C,R)$

The next step is to subtract the corresponding thermal offset to get the amplified pixel voltage:
IGNORE_ELOFF false→ $V_{Pix}(C,R) = V_{Abs}(C,R) - V_{elOff}(C) - V_{Th}(C,R)$
IGNORE_ELOFF true→ $V_{Pix}(C,R) = V_{Abs}(C,R) - V_{Th}(C,R)$

As described before, the value of $V_{Th}(C,R)$ should be determined by a linear interpolation between the two thermal offset values out of $Th_1(C,R), Th_2(C,R), Th_3(C,R), Th_4(C,R)$, which are as closest to the actual ambient temperature.

Next step is the linear interpolation to get the actual pixel constant $PixC(C,R)$, in dependency of the current ambient temperature T_{Amb}.

Now the object temperature T_O in tenths of Kelvin can be calculated by:

$$T_O = \sqrt[X]{\frac{V_{Pix}(C,R) \cdot PixC(C,R) \cdot VDM}{\varepsilon} + T_{Amb}^X} \qquad \text{[dK]}$$

ε is the emission coefficient of the surface the pixel is receiving the radiation from.
VDM is a multiplier, which is equal to the VDM constant.
X is an exponent, which is equal to the EXP constant. VDM and EXP can be both obtained with the control character 'M'.