BOYLE FIRMWARE DOCUMENTATION

Author: Herbert Schaunig
Email: herbert.schaunig@infineon.com
Phone: + 43 5 1777 6038
Revision: 1.1
Date: 13.1.2021

1. What does the Firmware do?

2. Command sequence for initialization of the PCB hardware and ASIC

3. Commands

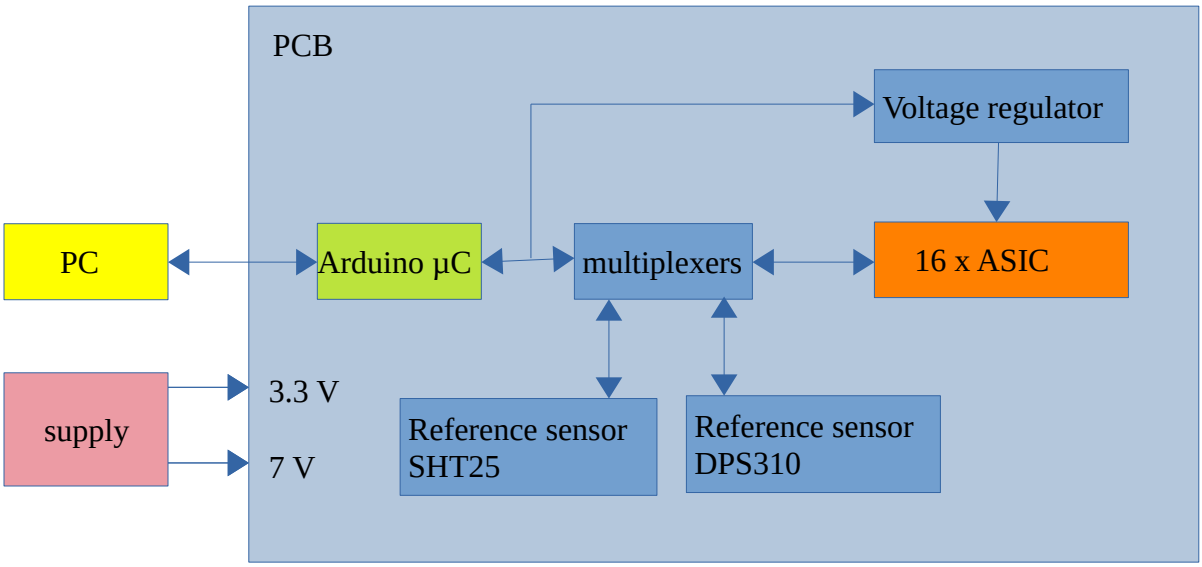
3.1	*IDN?	Ask hardware for identifier
3.2	STOP*	Terminate selftimed mode
3.3	autoscale	Do autoscaling of all internal reference resistors
3.4	measure_pid_t_offset	Calibration of heater PID controller
3.5	read_rref_settings	Read settings of internal reference resistors
3.6	SPS	Delay time between measurements in selftimed mode
3.7	SW	Open/Close hardware switches on the PCB
3.8	GBR	Block read off all ASIC registers
3.9	adc_st_16x	Start ASIC selftimed mode
3.10	r_reg	Read ASIC register
3.11	w_reg_data	Write ASIC registers
3.12	all_heaters_on	Enable all heaters
3.13	all_heaters_off	Disable all heaters
3.14	heater_t_on	On time of heater
3.15	heater_t_off	Off time of heater
3.16	heater_T_high	Heater high temperature
3.17	heater_T_low	Heater low temperature
3.18	heater_toggle_mode	Mode of heater operation
3.19	clean_on	On time of cleaning pulse
3.20	clean_off	Off time of cleaning pulse
3.21	clean_temp	Clean pulse temperature
3.22	clean_shift	Delay of first cleaning pulse to first heater pulse
3.23	dps310_pressure?	Read pressure from reference sensor DPS310
3.24	dps310_temperature?	Read temperature from reference sensor DPS310
3.25	sht25_humidity?	Read humidity from reference sensor SHT25
3.26	sht25_temperature?	Read temperature from reference sensor SHT25
3.27	E6	Set VDD of ASIC
3.28	version?	Read out firmware version
3.29	reference_sensors_enabled	enable reference sensor read out during selftime mode
3.30	reference_sensors_disabled	disable reference sensor read out durin selftime mode
3.31	ref_sens_interval	interval of reference sensor readout
3.32	heater timing recangular	one step between heater T_high and T_low
3.33	heater timing ramp	ramp between heater T_high and T_low
3.34	rising_steps	number of rising steps in heater ramp mode

3.35	falling_steps	number of falling steps in heater ramp mode
3.36	time_per_rising_step	duration of one rising step in heater ramp mode
3.37	time_per_falling_step	duration of one falling step in heater ramp mode

1. What does the Firmware do?

The main purpose of the firmware is to communicate with the multigas ASICs and provide an interface via a virtual COM port to the host (PC, raspberry, …).

Block diagram of setup:



2. Command sequence for initialization of the PCB hardware

Send command	Answer from PCB
SW 1 D	SW 1 13*
E6 00 00	E6*
Wait 100 ms (not a command!)	
E6 0D 00	E6*
con FFFF	connect_to 255 255*

Now send following default values to the ASIC via this command:

```
for (i=0;i<128;i++)
        w_reg_data 35 reg value
end
```

i.e.: w_reg_data 35 19 80 → write 0x80 to ASIC register 0x19

reg	value	reg	value	reg	value	reg	value
0	0x00	20	0x00	40	0x03	60	0x00
1	0x00	21	0x60	41	0x00	61	0x00
2	0x00	22	0x80	42	0x00	62	0x00
3	0x00	23	0x00	43	0x00	63	0x00
4	0x00	24	0x25	44	0x00	64	0x00
5	0x00	25	0x05	45	0x00	65	0x00
6	0x00	26	0x00	46	0x00	66	0x00
7	0x00	27	0x00	47	0x00	67	0x00
8	0x00	28	0x64	48	0x00	68	0x00
9	0x00	29	0x00	49	0x00	69	0x80
A	0x00	2A	0xb8	4A	0x00	6A	0x00
B	0x00	2B	0x0b	4B	0x00	6B	0x00
C	0x00	2C	0x00	4C	0x00	6C	0x00
D	0x00	2D	0x00	4D	0x00	6D	0x00
E	0x00	2E	0xd4	4E	0x00	6E	0x00
F	0x00	2F	0xb0	4F	0x00	6F	0x00
10	0x00	30	0xa5	50	0x00	70	0x00
11	0x00	31	0x01	51	0x00	71	0x00
12	0x00	32	0x80	52	0x00	72	0x00
13	0x00	33	0x00	53	0x00	73	0x00
14	0x00	34	0x00	54	0x00	74	0x00
15	0x00	35	0x00	55	0x00	75	0x00
16	0x00	36	0x74	56	0x00	76	0x00
17	0x00	37	0x00	57	0x00	77	0x00
18	0x00	38	0x00	58	0x00	78	0x00
19	0x80	39	0x00	59	0x00	79	0x00
1A	0x02	3A	0x00	5A	0x00	7A	0x00
1B	0x00	3B	0x88	5B	0x00	7B	0x00
1C	0x00	3C	0x7a	5C	0x00	7C	0x00
1D	0x23	3D	0x00	5D	0x00	7D	0x00
1E	0x23	3E	0x00	5E	0x00	7E	0x80
1F	0x09	3F	0x03	5F	0x00	7F	0x00

Description of SW 1:

It's a I2c controlled switch. U$8 on the 16x hardware

command	bit	function
SW 1	0	connect multiplexed INT signal to levelshifter for Arduino
SW 1	1	connect multiplexed INT signal to onboard ADC
SW 1	2	not used
SW 1	3	not used
SW 1	4	not used
SW 1	5	not used
SW 1	6	connect multiplexed INT signal to 300k (used for sinking bias current)
SW 1	7	not used

3. Commands

3.1 command: ***IDN?**
answer: BOYLE*

Ask hardware for identifier. This command can be used to scan all COM ports for the connected hardware.

3.2 command: **STOP***
answer: no answer

The stop command is used to terminate the self time mode (adc_st_16x)

3.3 command: **autoscale 2000** (2000 = value of external reference resistor)
answer: see description below

Do autoscaling of all internal reference resistors

This command will take some seconds to execute! The best internal reference resistors for all channels and all ASICs are chosen automaticly. Also the connected external reference resisor is used to calibrate all the internal reference resistors.
The µC will answer with a string containing all the calibration values for all ASCIs and all internal Rref settings.

A1_0.5k A1_1k … A1_1024k A2_0.5k … A2_1024k … A16_1024k *\r\n
A1_0.5k → ADC value for Rext_ref from ASCI1 with an internal Rref of 0.5k
A1_1k → ADC value for Rext_ref from ASCI1 with an internal Rref of 1k
A16_1024k→ ADC value for Rext_ref from ASCI16 with an internal Rref of 1024k

i.e.: a typical answer will look like this:

32767 32590 16270 8135 4059 2021 1017 507 239 121 56 11 … *\r\n

Now the real value of all internal reference resistors can be calculated. With the values, which you get from the „autoscale“ command it is possible to calculate the exact resistance of the internal Rrefs (example with Rref_external = 1k and internal Rref = 2k):

int_rref_abs_value_ohm = 32768 / A1_2k * value_of_external_rref;

value_of_external_rref = 2000;

A1_2k is one of the values, which the „autoscale“ command is answering.

Absolute value of sensor = ADC_LSBs / 32768 * **rref_abs_value_ohm**

3.4 command: **measure_pid_t_offset**
answer: 16x value of PID_T_OFFSET + ASIC temperature

i.e.: 45056 45056 57344 45056 57344 47104 57344 57344 59392 45056
55296 57344 45056 45056 57344 45056 22.55 *

This command needs to be executed once to calibrate the heater PID controller. The answer is just for information/debugging. No need for further processing of these values. All necessary stuff is done in the firmware.

3.5 command: **read_rref_settings**
answer: 16 x 5 bytes

Read settings of internal reference resistors.

```
// 5bytes per ASIC are transmitted
// 0x3B = RF2<7:4), RF1<3:0>
// 0x3C = RF4<7:4), RF3<3:0>
// 0x3D = RF6<7:4), RF5<3:0>
// 0x3E = RF8<7:4), RF7<3:0>
```

```
      // 0x3F = RT2<7:4), RT1<3:0>

int k = 0;
for(int i=0;i<16;i++)
{
  for(int j=0;j<5;j++)
  {
      selected_rref_for_all_ascics[i][j] = data[k++];
  }
  rref1_index[i] = selected_rref_for_all_ascics[i][0] & 0x0F;
  rref2_index[i] = (selected_rref_for_all_ascics[i][0] & 0xF0) >> 4;
  rref3_index[i] = selected_rref_for_all_ascics[i][1] & 0x0F;
  rref4_index[i] = (selected_rref_for_all_ascics[i][1] & 0xF0) >> 4;
  rref5_index[i] = selected_rref_for_all_ascics[i][2] & 0x0F;
  rref6_index[i] = (selected_rref_for_all_ascics[i][2] & 0xF0) >> 4;
  rref7_index[i] = selected_rref_for_all_ascics[i][3] & 0x0F;
  rref8_index[i] = (selected_rref_for_all_ascics[i][3] & 0xF0) >> 4;
  rrefrt1_index[i] = selected_rref_for_all_ascics[i][4] & 0x0F;
  rrefrt2_index[i] = (selected_rref_for_all_ascics[i][4] & 0xF0) >> 4;

}
```

3.6 command: **SPS xx**
answer: SPS yy *

xx … delay time in ms (in hex)
yy … delay time in ms (in dec)

Sets the delay time between measurements in selftimed mode.

i.e.: SPS 10 → delay 16ms between measurements
answer: SPS16* → answer in decimal

3.7 command: **SW x y**
answer: SW u v *

x y … switch number and value in hex
u v … answer in decimal

Open/Close hardware switches on the PCB

i.e.: SW 1 D
answer: SW 1 13 *

3.8 command: **GBR** 35 00
answer: 128 bytes (binary)

35 = I2C address of ASIC

00 = start address

Block read off all ASIC registers.

Byte #	description
0	sample counter MSB
1	sample counter LSB
2	RTEMP LSB
3	RTEMP MSB
4	RSENS_1 LSB
5	RSENS_1 MSB
6	RSENS_2 LSB
7	RSENS_2 MSB
8	RSENS_3 LSB
9	RSENS_3 MSB
10	RSENS_4 LSB
11	RSENS_4 MSB
12	RTEMP LSB
13	RTEMP MSB
14	RREF_EXT LSB
15	RREF_EXT MSB
16	ASIC TEMP 0 LSB
17	ASIC TEMP 0 CSB
18	ASIC TEMP 0 MSB
19	ASIC TEMP 1 LSB
20	ASIC TEMP 1 CSB
21	ASIC TEMP 1 MSB
22	humidity MSB
23	humidity LSB
24	pressure LSB
25	pressure CSB
26	pressure MSB
27	temperature MSB
28	temperature LSB
407	RTEMP LSB
408	RTEMP MSB
409	RSENS_1 LSB
410	RSENS_1 MSB
411	RSENS_2 LSB
412	RSENS_2 MSB
413	RSENS_3 LSB
414	RSENS_3 MSB
415	RSENS_4 LSB
416	RSENS_4 MSB
417	RTEMP LSB
418	RTEMP MSB
419	RREF_EXT LSB
420	RREF_EXT MSB
421	ASIC TEMP 0 LSB
422	ASIC TEMP 0 CSB
423	ASIC TEMP 0 MSB
424	ASIC TEMP 1 LSB
425	ASIC TEMP 1 CSB
426	ASIC TEMP 1 MSB
427	humidity MSB
428	humidity LSB
429	pressure LSB
430	pressure CSB
431	pressure MSB
432	temperature MSB
433	temperature LSB
434	end of data *\r\n\r\n

3.9

command: **adc_st_16x**

answer: continues real time data stream of all ASIC result registers (binary)

Start ASIC selftimed mode. The mode can be stopped with the “STOP*” command

In total 435 bytes are transmitted for every measurement cycle.

The table below is showing the structure and order of the bytes.

3.10 command: **r_reg 35 xx**

answer: yy *

xx … register address (in hex)
35 … I2C address of ASIC

Read ASIC register content.

i.e.: r_reg 35 18 → read register 0x18
answer: 10 *

3.11 command: **w_reg_data 35 xx yy**
answer: w_reg_data 35 uu vv *

xx … register address (in hex)
yy … register value (in hex)
uu … register address (in dec)
vv … register value (in dec)
35 … I2C address of ASIC

Write ASIC registers.

i.e.: w_reg_data 35 18 0A → write 0x0A to register 0x18
answer: w_reg_data 53 24 10 *

3.12 command: **all_heaters_on**
answer: all_heaters_on *

Enable all heaters in self timed mode.

3.13 command: **all_heaters_off**
answer: all_heaters_off *

Disable all heaters during self timed mode.

3.14 command: **heater_t_on xx**
answer: heater_t_on yy *

xx … heater on time in units of 100ms (in hex)
yy … heater on time in units of 100ms (in dec)
On time of heater. See 3.31 heater timing diagram.

3.15 command: **heater_t_off xx**
answer: heater_t_off yy *

xx … heater off time in units of 100ms (in hex)
yy … heater off time in units of 100ms (in dec)

Off time of heater. See 3.31 heater timing diagram.

3.16 command: **heater_T_high xx**
answer: heater_T_high yy *

xx … heater high temperature in °C (in hex)
yy … `(xx - rtemp_t_offset_calib_temp)/2` (in dec)

Heater high temperature. To achieve absolute temperature accuracy, the heater PID loop needs to be calibrated with “measure_pid_t_offset” command before.

`rtemp_t_offset_calib_temp:` `This value is stored by the` measure_pid_t_offset routine. The divided by 2 in the return value is needed, because the ASIC register for the target temperature only takes values in units of 2°. The return value is the real value in the ASIC register. See 3.31 heater timing diagram.

3.17 command: **heater_T_low xx**
answer: heater_T_low yy *

xx … heater low temperature in °C (in hex)
yy … `(xx - rtemp_t_offset_calib_temp)/2` (in dec)

Heater low temperature. To achieve absolute temperature accuracy, the heater PID loop needs to be calibrated with “measure_pid_t_offset” command before.

`rtemp_t_offset_calib_temp:` `This value is stored by the` measure_pid_t_offset routine. The divided by 2 in the return value is needed, because the ASIC register for the target temperature only takes values in units of 2°. The return value is the real value in the ASIC register. See 3.31 heater timing diagram.

3.18 command: **heater_toggle_mode xx**
answer: heater_toggle_mode yy *

xx … heater toggle mode

yy … heater toggle mode

Mode of heater operation.

heater_toggle_mode	description
0	no automatic toggling in self timed mode
1	heater is toggled between T_high and heater off
2	heater is toggled between T_high and T_low

3.19 command: **clean_on xx**
answer: clean_on yy *

xx … on time of the cleaning pulse in units of 100ms(hex)
yy … on time of the cleaning pulse in units of 100ms(dec)

On time of cleaning pulse. See 3.31 heater timing diagram.
If clean_on time is set to zero, the cleaning pulse feature is disbled.

3.20 command: **clean_off xx**
answer: clean_off yy *

xx … off time of the cleaning pulse in units of 100ms(hex)
yy … off time of the cleaning pulse in units of 100ms(dec)

Off time of cleaning pulse. See 3.31 heater timing diagram.

3.21 command: **clean_temp xx**
answer: clean_temp yy*

xx … cleaning pulse temperature (hex)
yy … `(xx - rtemp_t_offset_calib_temp)/2` (in dec)

Heater cleaning temperature. To achieve absolute temperature accuracy, the heater PID loop needs to be calibrated with “measure_pid_t_offset” command before. See 3.31 heater timing diagram.

`rtemp_t_offset_calib_temp:` `This value is stored by the` measure_pid_t_offset routine. The divided by 2 in the return value is needed, because the ASIC register for the target temperature only takes values in units of 2°. The return value is the real value in the ASIC register.

3.22 command: **clean_shift xx**
answer: clean_shift yy *

xx … shift of the cleaning pulse to the heater pulse (hex)
yy … shift of the cleaning pulse to the heater pulse (dec)

Delay of first cleaning pulse to first heater pulse. See 3.31 heater timing diagram.

3.23 command: **dps310_pressure?**
answer: xx*

xx … pressure in Pascal (dec)

Read pressure from reference sensor DPS310

3.24 command: **dps310_temperature?**
answer: xx*

xx … temperature in °C (dec)

Read temperature from reference sensor DPS310

3.25 command: **sht25_humidity?**
answer: xx*

xx … humidity in RH% (dec)

Read humidity from reference sensor SHT25

3.26 command: **sht25_temperature?**
answer: xx*

xx … temperature in °C (dec)

Read temperature from reference sensor SHT25

3.27 command: **E6**
answer: E6*

xx … ASIC VDD in mV MSB (hex)
yy … ASIC VDD in mV LSB (hex)

Set VDD of ASIC.

i.e.: E6 0D 00 → 0x0D00 = 3328 mV

3.28 command: **version?**

Answer: 1.4.2020 * (i.e.)

Read out firmware version

3.29 command: **reference_sensors_enabled**
answer: reference_sensors_enabled*

Enable reference sensor read out during selftime mode.

Since the readout of the reference sensors during self timed mode takes quite some time, the user can choose, if they should be read out or not. This command is enabling it.

3.30 command: **reference_sensors_disabled**
answer: reference_sensors_disabled*

Disable reference sensor read out durin selftime mode.

Since the readout of the reference sensors during self timed mode takes quite some time, the user can choose, if they should be read out or not. This command is enabling it.

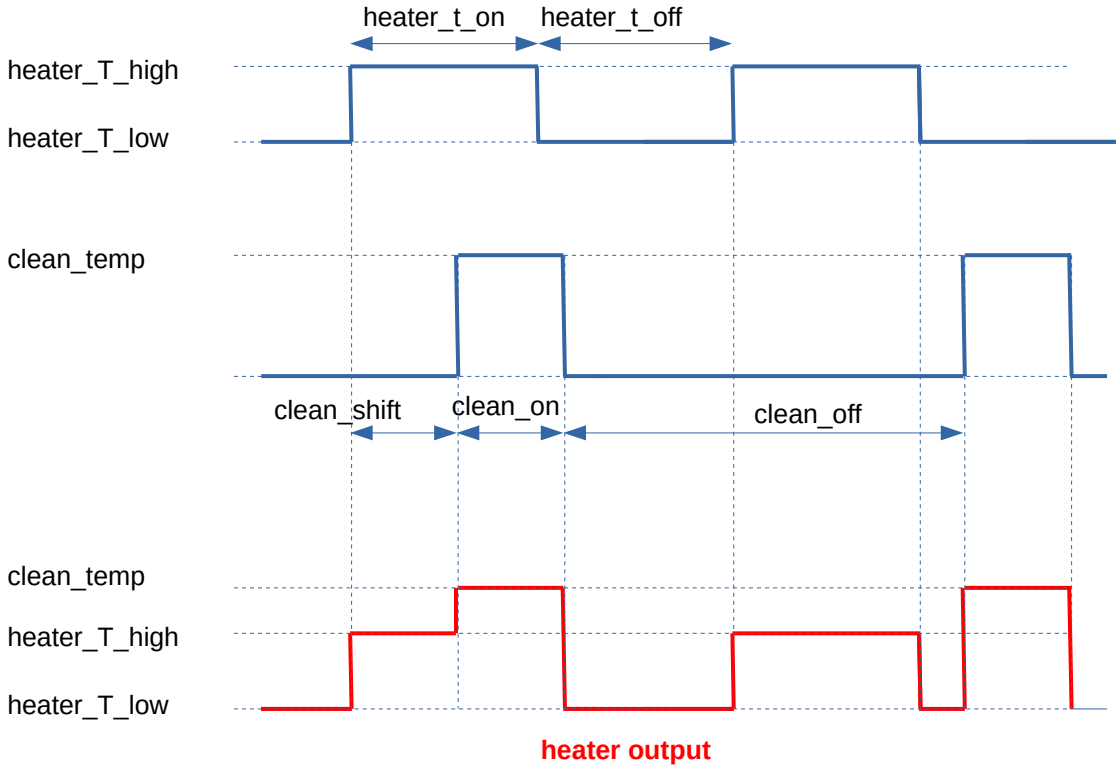
3.31 command: **ref_sens_interval xx**
answer: ref_sens_interval *

xx … interval of reference sensors readout (hex)
yy … interval of reference sensors readout (dec)

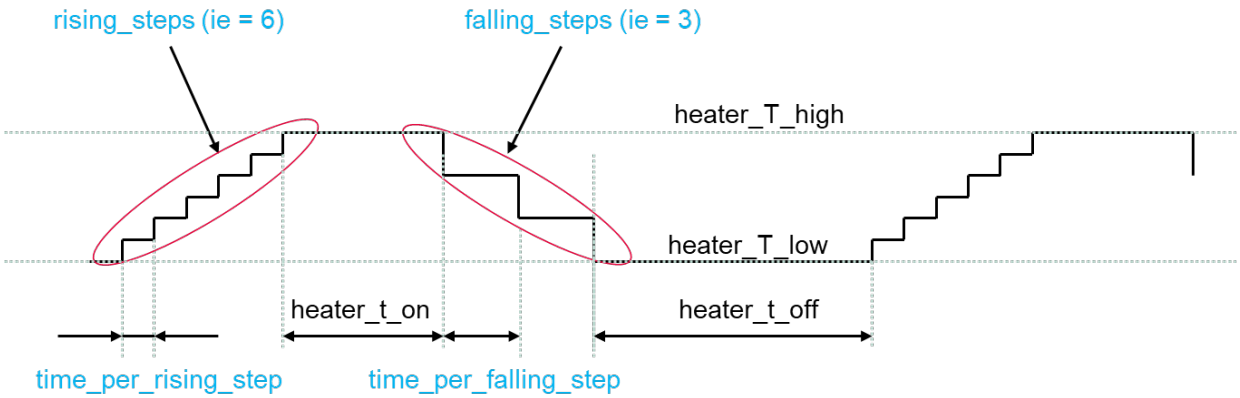
The reference sensors can be read out during selftimed mode if the command **reference_sensors_enabled** was sent. The period of the readout can be set by **ref_sens_interval**. Unit are samples.

i.e.: ref_sens_interval 64 (hex) → measure reference sensor after every 100th ASIC sample.

3.32 Heater timing diagram (no ramp)



3.33 Heater timing diagram (with additional ramp parameters)



Cleaning pulse can also be added additionaly like in standard rectangular operation (see 3.32)

3.34 command: **rising_steps xx**
answer: rising_steps yy *

xx … number of rising steps in heater ramp mode (in hex)
yy … number of rising steps in heater ramp mode (in dec)

Number of rising steps in heater ramp mode. If set to zero, ramping for rising edge is disabled.

3.34 command: **falling_steps xx**
answer: falling_steps yy *

xx … number of falling steps in heater ramp mode (in hex)
yy … number of falling steps in heater ramp mode (in dec)

Number of falling steps in heater ramp mode. If set to zero, ramping for falling edge is disabled.

3.35 command: **time_per_rising_step xx**
answer: time_per_rising_step yy *

xx … time per rising step in units of 100ms (in hex)
yy … time per rising step in units of 100ms (in dec)

Time per rising step in heater ramp mode.

3.36 command: **time_per_falling_step xx**
answer: time_per_falling_step yy *

xx … time per falling step in units of 100ms (in hex)
yy … time per falling step in units of 100ms (in dec)

Time per falling step in heater ramp mode.