



Ultrasonic-Flow-Converter

Data Sheet

TDC-GP30

Firmware, Memory and ROM Overview

June 9th, 2016

Document-No: DB_GP30Y_Vol4_en V0.5

acam-messelectronic gmbh is now a member of ams group



Copyrights & Disclaimer

Copyright acam-messelectronic gmbh, Friedrich-List-Str. 4, 76297 Stutensee, Germany-Europe. Trademarks Registered. All rights reserved. The material herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner.

Devices sold by acam-messelectronic gmbh are covered by the warranty and patent indemnification provisions appearing in its General Terms of Trade. acam-messelectronic gmbh makes no warranty, express, statutory, implied, or by description regarding the information set forth herein. acam-messelectronic gmbh reserves the right to change specifications and prices at any time and without notice. Therefore, prior to designing this product into a system, it is necessary to check with acam-messelectronic gmbh for current information. This product is intended for use in commercial applications. Applications requiring extended temperature range, unusual environmental requirements, or high reliability applications, such as military, medical life-support or life-sustaining equipment are specifically not recommended without additional processing by acam-messelectronic gmbh for each application. This product is provided by acam-messelectronic gmbh "AS IS" and any express or implied warranties, including, but not limited to the implied warranties of merchantability and fitness for a particular purpose are disclaimed.

acam-messelectronic gmbh shall not be liable to recipient or any third party for any damages, including but not limited to personal injury, property damage, loss of profits, loss of use, interruption of business or indirect, special, incidental or consequential damages, of any kind, in connection with or arising out of the furnishing, performance or use of the technical data herein. No obligation or liability to recipient or any third party shall arise or flow out of acam-messelectronic gmbh rendering of technical or other services.

"Preliminary" product information describes a product which is not in full production so that full information about the product is not yet available. Therefore, acam-messelectronic gmbh ("acam") reserves the right to modify this product without notice.

Support / Contact

For direct sales, distributor and sales representative contacts, visit the acam web site at:
www.acam.de www.ams.com

For technical support you can contact the acam support team:
support.stutensee@ams.com
or by phone +49-7244-74190.

Notational Conventions

Throughout the GP30 documentation, the following style formats are used to support efficient reading and understanding of the documents:

- Hexadecimal numbers are denoted by a leading 0x, e.g. 0xAF = 175 as decimal number. Decimal numbers are given as usual.
- Binary numbers are denoted by a leading 0b, e.g. 0b1101 = 13. The length of a binary number can be given in bit (b) or Byte (B), and the four bytes of a 32b word are denoted B0, B1, B2 and B3 where B0 is the lowest and B3 the highest byte.
- Abbreviations and expressions which have a special or uncommon meaning within the context of GP30 application are listed and shortly explained in the list of abbreviations, see following page. They are written in plain text. Whenever the meaning of an abbreviation or expression is unclear, please refer to the glossary at the end of this document.
- **Variable names** for hard coded registers and flags are in bold. Meaning and location of these variables is explained in the datasheet (see registers CR, SRR and SHR).
- ***Variable names*** which represent memory or code addresses are in bold italics. Many of these addresses have a fixed value inside the ROM code, others may be freely defined by software. Their meaning is explained in the firmware and ROM code description, and their physical addresses can be found in the header files. These variable names are defined by the header files and thus known to the assembler as soon as the header files are included in the assembler source code. Note that different variable names may have the same address, especially temporary variables.
- *Physical variables* are in italics (real times, lengths, flows or temperatures).

If the contents of a memory cell represents a number, the following notation is used: The hexadecimal numbers x_{hex} are either int (integer) and can be directly converted decimal numbers, or fdN, for example fd16. The latter means N, for example 16, floating digits, and the corresponding decimal number x_d is calculated as

$$x_d = HEX_TO_DEC(x_{hex})/2^N \quad [1]$$

For physical variables, it is also necessary to know the units of the number. They can be given as usual, or in chip-internal units like high speed clock periods (typically 250 ns; all raw TDC results are in this unit) or voltage steps (typically 0.88 mV). An example would be a first hit voltage level of 0x55 int, meaning 74.8 mV (= 85 * 0.88 mV) or a raw TOF result of 0x010862D1 fd16, meaning ~66.1 μ s (= 17326801 * 0.25 μ s / 2¹⁶). Remark: The high speed clock period would in addition need to be corrected by HSC calibration for full accuracy.

Abbreviations

AM	Amplitude measurement
CD	Configuration Data
CPU	Central Processing Unit
CR	Configuration Register
CRC	Cyclic Redundancy Check
DIFTOF,	Difference of up and down ->TOF
DIFTOF_ALL	
DR	Debug Register
FEP	Frontend Processing
FDB	Frontend data buffer
FHL	First hit level (physical value V_{FHL})
FW	Firmware, software stored on the chip
FWC	Firmware Code
FWD	Firmware Data
FWD-RAM	Firmware Data memory
GPIO	General purpose input/output
Hit	Stands for a detected wave period
HSO	High speed oscillator
INIT	Initialization process of ->CPU or -> FEP
IO	Input/output
I2C	Inter-Integrated Circuit bus
LSO	Low speed oscillator
MRG	Measurement Rate Generator
NVRAM, NVM	Programmable Non-Volatile Memory
PI	Pulse interface
PP	Post Processing
PWR	Pulse width ratio
R	RAM address pointer of the CPU, can also stand for the addressed register
RAA	Random Access Area
RAM	Random Access Memory
RI	Remote Interface
ROM	Read Only Memory
ROM code	Hard coded routines in ROM
SHR	System Handling Register
SPI	Serial Peripheral Interface
SRAM	Static RAM
SRR	Status & Result Register
SUMTOF	Sum of up and down TOF
Task	Process, job
TDC	Time-to-digital-converter
TOF, TOF_ALL	Time of Flight
TS	Task Sequencer
TM	Temperature measurement
UART	Universal Asynchronous Receiver & Transmitter
USM	Ultrasonic measurement
V_{ref}	Reference voltage
X,Y,Z	Internal registers of the CPU
ZCD	Zero cross detection, physical level V_{ZCD}

For details see the glossary in section 13.

Content

1	Introduction	1-3
1.1	Alternative firmware options overview	1-3
1.2	General remarks on TDC-GP30 operation with firmware	1-4
1.3	General remarks on interface communication and result data storage.....	1-5
2	acam's self-contained firmware: Features and usage overview	2-7
2.1	A first impression.....	2-7
2.2	acam firmware features	2-10
2.3	Setup and customization	2-12
2.4	Creating and downloading a firmware data file	2-13
2.5	Calibration process in development and production	2-16
2.6	Configuring and using the pulse interface	2-18
3	Setting of acam firmware parameters in firmware data (FWD)	3-19
3.1	Parameters for general operation control	3-19
3.2	Calibration parameters	3-22
3.3	Parameters for zero flow and negative flow operation	3-24
3.4	Parameters for error handling	3-25
3.5	Parameters which influence error counters and error interrupt	3-28
3.6	Parameters which control FHL regulation.....	3-29
3.7	Parameters for sensor temperature measurement	3-31
3.8	(Optionally) Unused FWD cells	3-32
4	First hit level setting	4-34
4.1	First hit level selection criteria.....	4-34
4.2	First hit level regulation methods of acam firmware.....	4-38
5	Error handling and operation safety measures	5-44
5.1	Error handling.....	5-44
5.2	Error handling flow chart.....	5-47
5.3	Error counters	5-49
5.4	Error interrupt.....	5-50
5.5	Error signals through pulse interface.....	5-50
6	acam firmware variables in RAM	6-51
6.1	Results in memory cells.....	6-51
6.2	acam firmware variables in RAM-part of FWD.....	6-62
7	acam firmware parameters in firmware data (FWD)	7-63
7.1	<i>FWD_FW_CONFIG</i> (Firmware configuration) 0x16A.....	7-67
8	acam firmware structure	8-69
8.1	List of related files	8-71
8.2	acam firmware version numbers	8-72
9	Firmware with combined customer and acam code	9-73

9.1	Internal firmware structure	9-73
9.2	Programming the chip.....	9-75
10	Full customer firmware code.....	10-76
11	Demo code: Example_DIF_OVER_PI	11-76
12	ROM routines	12-80
12.1	Overview	12-80
12.2	Currently available ROM routines for general usage	12-80
13	Glossary.....	13-83
14	Miscellaneous	14-89
14.1	Bug Report	14-89
14.2	Last Changes	14-89

1 Introduction

TDC-GP30 is the next generation in acam's development for ultrasonic flow converters. Using its integrated CPU and code memory, TDC-GP30 can be operated with a dedicated firmware for result evaluation and operation control. This is typically the case when TDC-GP30 is employed in flow meter or heat meter mode. acam offers firmware support on different levels. This document describes the supported options in detail, as listed below. In addition, a short example code is discussed in chapter 11 and an overview of available ROM routines is given in chapter 12. For details on ROM routines and programming see Manual Volume 2: CPU, Memory and Firmware.

1.1 Alternative firmware options overview

The three alternative options for firmware are in overview:

1) acam's proprietary self-contained firmware (no customer firmware coding)

- Is described in detail in the following chapters
- The customer performs no firmware code programming on GP30 at all.
- The chip is delivered pre-programmed. The customer only modifies firmware data, using acam's control software or simple scaling rules. The firmware data contains configuration and calibration parameters.
- **In development phase of the flowmeter**, the customer has to test and qualify the device with GP30 and its firmware, to decide for a measurement configuration, and to prepare mass calibration by calculating basic calibration parameters.
- **In production phase of the flowmeter**, the basic calibration parameters are modified and adapted to the given spool piece, using a fast 2-point calibration for every device. The individual calibration parameters are stored in firmware data on every chip.
- acam supports the configuration and calibration process in development as well as in production phase by software, standard examples and documented procedures.
- The firmware runs fully autonomously and provides results in memory cells which can be read by a master controller over the interface (SPI or UART). In the simplest case, the standard pulse interface of GP30 is used for water volume readout (see section 2.6).
- The firmware can be configured for cold water meter as well as heat meter application. The cold water meter provides water flow, volume and temperature information and can be used up to a water temperature of 60 °C. The heat meter mode adds precision temperature measurement using e.g. up to two PT 500/1000 sensors in two or four-wire configuration. Bubble- and no-water-detection are included, too.
- Overall current consumption of GP30 in cold water meter mode with 8 Hz measurement frequency can be for example 6 µA. In heat meter mode with 1 Hz measurement frequency and temperature measurement every 30 seconds, it may for example be 3.5 µA

2) Customer firmware code in combination with acam's proprietary firmware.

- This alternative is described in chapter 9.
- The chip is delivered pre-programmed with acam's firmware parts.
- The customer adds his own code parts by overwriting the open firmware code part.
- Customer firmware coding is supported by an assembler tool, which is integrated in the evaluation software. A number of standard ROM routines simplify the firmware development and support useful functions, e.g. for filtering or simplified configuration of the pulse interface. For details see Manual Volume 2: CPU, Memory and Firmware.
- The customer may still simply use memory cells and the interfaces for external communication, but also adapt and further evaluate results. Operation control and any kind of event handling may be modified or expanded in customer firmware part as required.
- Qualification, configuration and calibration have to be done by the customer the same way as in the preceding case.

3) Full customer firmware code (without using any part of acam's proprietary firmware).

- This alternative is very shortly sketched in chapter 10.
- The chip is delivered empty, without acam's firmware.
- Firmware coding is supported by an assembler which is integrated in the evaluation software and by a number of standard ROM routines, for example for filtering or simplified configuration of the pulse interface. For details see Manual Volume 2: CPU, Memory and Firmware.
- The customer has the full flexibility to write the firmware according to his needs and application.

1.2 General remarks on TDC-GP30 operation with firmware

TDC-GP30 is designed for flow meter operation with a dedicated firmware which evaluates results at lowest power consumption. It aims at relaxing demands for the external controller, which may mainly serve as interface to the outside. The more TDC-GP30 takes control over chip operation, the more the firmware has to pay attention to error handling and long term operation stability.

The acam firmware always aims at operating in active measurement state, configured according to the stored firmware data. When it is disturbed or interrupted, it will resume nominal operation by a watchdog reset, and when its configuration is modified, it will restore it latest after one hour. Some configurations can't be changed at all, see sections 3.1.2 and 3.1.4 for details.

The right way to stop the firmware and change configurations, for example for testing, is to switch off post processing and disable the watchdog (else a reset happens within typically 13 s). Then any configuration can be tested without firmware operation. With firmware operating, the desired

configuration should be stored in firmware data, see section 2.4.1. With a few exceptions, a modified configuration can be written to the chip directly while the firmware operates. But note that then the firmware will restore its stored configuration at any full hour of the built-in real time clock

SRR_TS_HOUR and **SRR_TS_MIN_SEC** (0x0E6 and 0x0E7).

The unusual case of permanent sensor temperature measurements (**TM_RATE** = 1) needs special treatment for reliable communication with the chip when operating with acam firmware. Please contact acam when you want to operate the chip with this setting.

Sometimes you may want to connect a chip with unknown stored firmware data and configuration. In particular for a correct measurement display, it is highly recommended to synchronize the configuration of the chip and the PC software. This can be done by connecting the chip, then opening menu tools/registers, and then clicking "Read and Transfer", or activate "Read Config from RAM first" before starting measurement from the main window. This loads the chip's currently stored configuration into the PC software. Please check if the values for TOF rate and First Hit Level are reasonable, since they are most critical for operation. Returning to the measurement sheet, click "Write Config". This writes the configuration that was just read out back into the chip. It should not actually do any change to the chip operation, but it informs the PC software that its current status is actively stored on the chip.

1.3 General remarks on interface communication and result data storage

No matter which firmware option is chosen, the question of stable and efficient communication between chip and microcontroller must always be solved. When operating without firmware, new measurement results must always be read out before the next measurement. This is typically done using an interrupt issued by the chip after each measurement, which of course generates a high amount of data traffic. With firmware, the data traffic can be minimized since the firmware stores and evaluates intermediate measurement results. Thus a firmware can provide processed results after even long time periods without communication. Then the decision on how and when the controller retrieves data from TDC-GP30 depends, among others, on the following criteria:

- Is it required to have the data in external controller always up to date? Which update rate would be acceptable?
- Is it possible to retrieve new data on request (for example when results are requested from outside)? How much delay time is acceptable for such a process?
- How should an error case be handled? How is a permanent and error-free result data storage guaranteed? How much data loss is acceptable in case of serious errors?

These factors have to be balanced against the power consumption costs caused by communication and by waking up the external controller.

In general, permanent storage of result data must be done outside TDC-GP30. It may, however,

be acceptable to update for example flow volume data at a low rate (say every hour) and to accept the mistake caused by some (unexpected and presumably extremely rare) fatal error event which may erase the stored flow volume on TDC-GP30. The last resort of TDC-GP30 in case of a fatal error is a watchdog reset, after which all stored measurement data will be initialized to zero, including flow volume. This will only happen if it is the only way the chip can return to normal operation (serious data corruption after a power drop, or chip operation blocked by remote commands; note that in both cases the stored data is already getting wrong).

With all these considerations in mind, some proposals for stable and efficient communication between chip and controller can be made:

- Reduce communication to a low level for low power consumption.
- But store important results outside and at an acceptable rate.
- Use error-triggered interrupts to inform the controller quickly about errors (see section 3.5)
- Synchronize communication with chip operations; else wrong data may be read out (see also Manual Volume 3: User Manual, section 4.2.2):
 - Use interrupts to synchronize: Either regular interrupts like “End of Task Sequencer” (configurable) or, for arbitrarily reduced communication rates, a communication request over remote command (see Manual Volume 1: General Data and Frontend Description, section 5.4.7). The acam firmware will answer with a synchronous firmware interrupt when ready for communication.
 - Or by keeping track of chip operations over time and accessing the chip only during IDLE (see Manual Volume 3: User Manual, section 4.2.2 and 7.1). To synchronize internal and external clocks, the task sequencer time in **SRR_TS_TIME** (0x0E9) can be used. This register contains the time elapsed since the last task sequencer trigger.
 - Or access data at any time, but control the validity of results, e.g. by writing and reading a pair of RAM cell at beginning and end of communication (writing one value to a RAM cell and directly reading it back is not a valid test, since the buffer of the interface may just return the last written value; write two values sequentially and read them back as validity check).

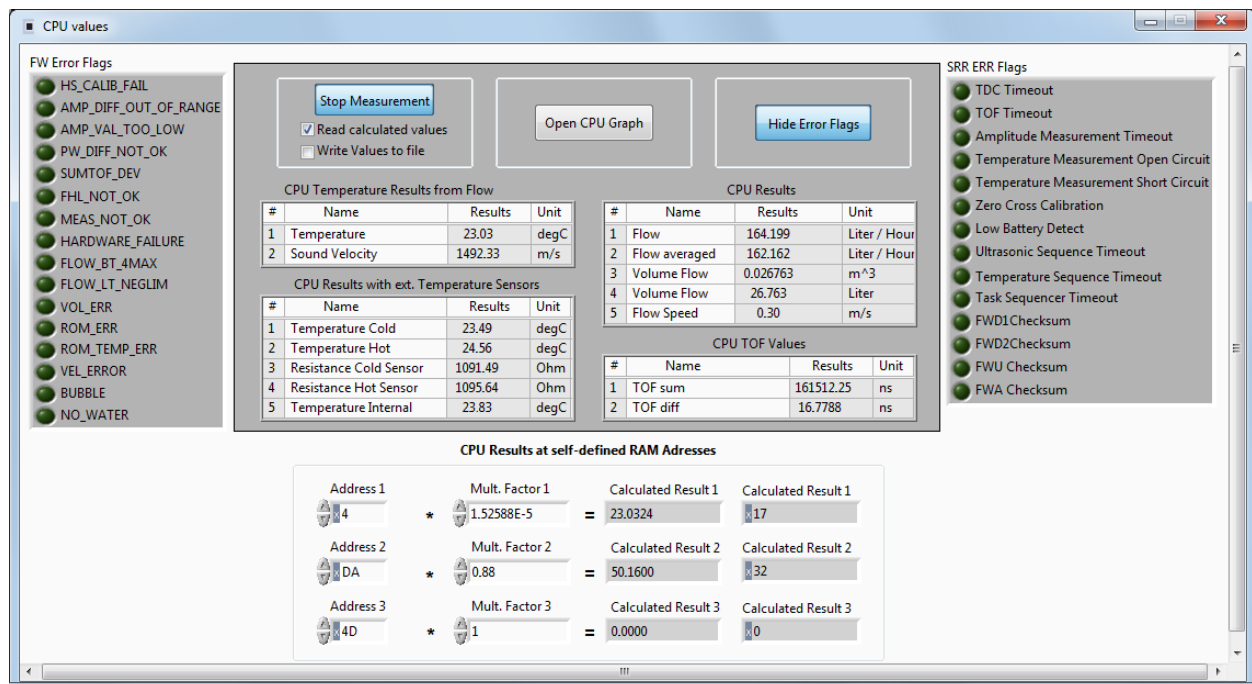
Despite all security remarks on possibly losing volatile data on TDC-GP30, the stored data is secured to a high degree, as long as sufficient operation voltage is available. Typically stored RAM data remains down to operation voltage levels of about 2 V. The acam firmware signals low voltage by a “low” on GPIO6, which can be evaluated from outside even when chip communication is not possible any more. The most important measurement result, the flow volume, can be secured against data corruption by an optional redundant storage (see section 3.1.1).

2 acam's self-contained firmware:

Features and usage overview

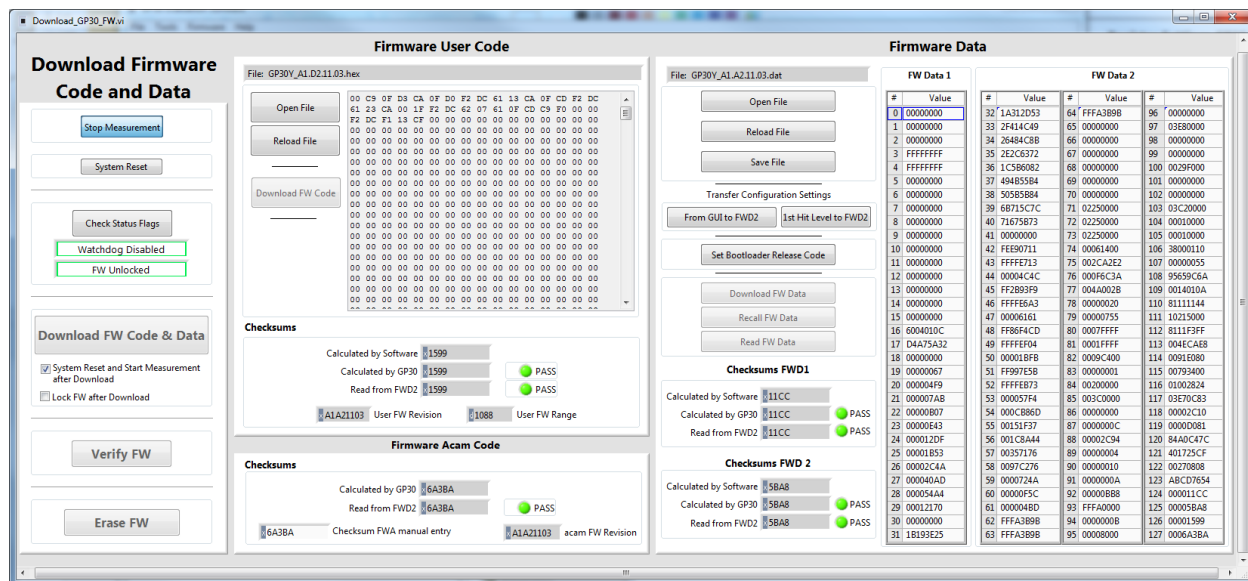
2.1 A first impression

When starting to work with acam firmware on TDC-GP30, it is proposed to get a first impression from our demo kits with operating firmware. In the acam GP30 evaluation software, under the menu point Firmware/CPU values the following window can be found:



This example screenshot gives an overview of the acam firmware capabilities: Calculation of actual and accumulated flow as well as water temperature and sensor temperature measurement on the one hand (middle panel); operation control and error signal and evaluation on the other hand (error flag panels at each side). This window also demonstrates how results are read out: The lower section contains three rows where the contents of any RAM cell can be read and displayed after multiplication by a suitable factor. For example, under address 0x004 in the first row the current water temperature is stored in some hexadecimal format. The numeric value in °C, displayed under “Calculated result 1”, is achieved by multiplication with the suitable factor $1/2^{16}$. Information about location of results, format and suitable factors is given in chapter 6. As another example, the addresses 0x0DA and 0x04D, which were set in the other two rows, contain the current first hit level in up direction (FHL, in mV under “Calculated Result 2”) and the current error count (in “Calculated Result 3”), which is zero here. FHL values have a major influence on operation and are controlled by the firmware, as discussed in detail in chapter 4. Details on error counting can be found in section 5.3.

Another window under the same menu Firmware/Firmware download gives some insight about the code and how parameters are stored (see also zoom-in on next page):

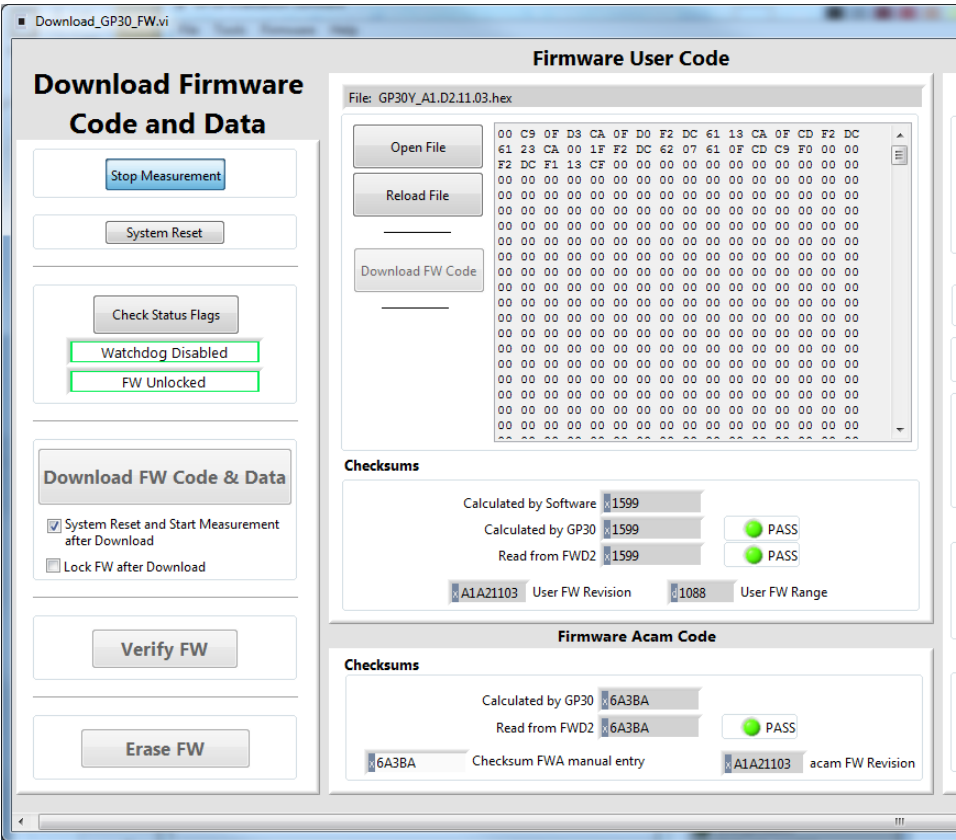


This window is the interface of the PC software for downloading and checking firmware code and firmware data. Firmware code, in the two middle panels, separates into firmware user code (higher panel) and firmware acam code (lower panel). The user code part can be opened and displayed, and of course downloaded to the chip. The “User FW range” denotes how much bytes are free for user code - this number is determined by the acam code part. The acam code part is pre-programmed at delivery and can’t be changed. Both code parts have a version number which is displayed after “Verify FW”. For details on version numbers see section 8.2. The panel on the right side contains the firmware data. This is the memory space where permanent configurations, parameters and calibration coefficients are stored. This data can also be opened, displayed and stored to file, and even be modified in this window. In contrast to firmware code, firmware data can also be read from the chip. For details on how to modify this data see section 2.4. The functional meaning of the parameters is discussed throughout chapters 3 to 5, and the overview of all parameters is given in chapter 7.

For both code and firmware data memory, TDC-GP30 calculates a checksum, and the PC software compares it to the values stored in firmware data cells 124 to 127 as well as to the displayed data. This permits a check on data integrity. Note that checksums may be updated by the PC software.

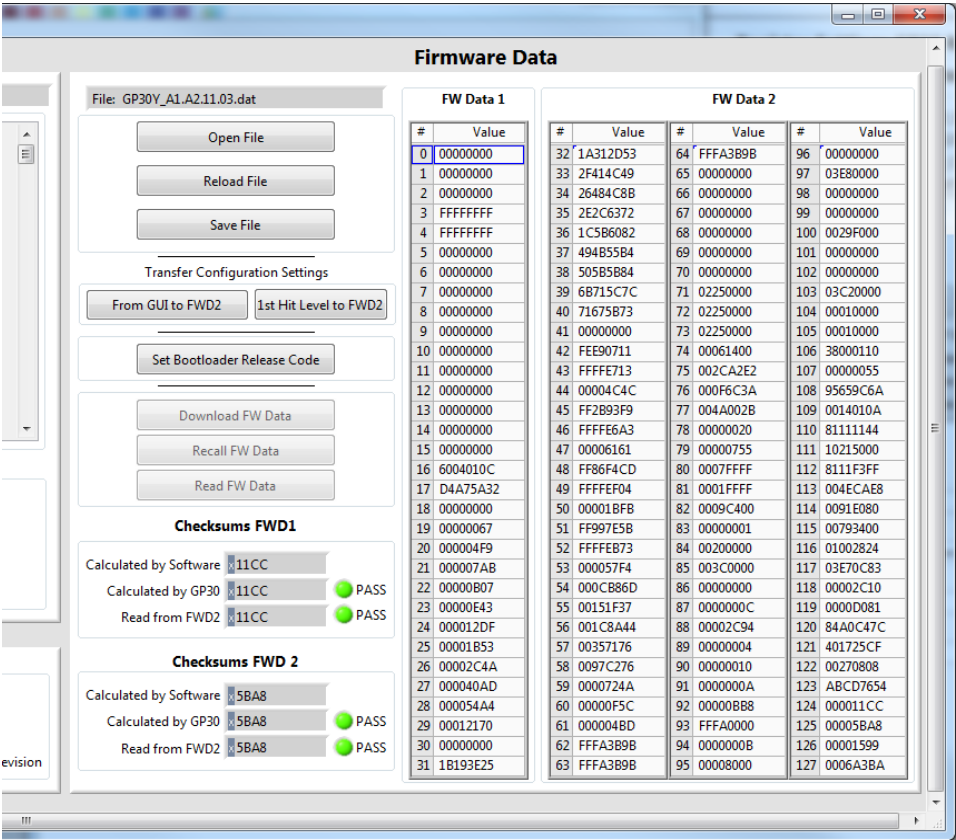
The control buttons on the left side permit start and stop of chip operation, reset and download (only possible when chip is stopped – but in the screen shot the chip is running, indicated by the blue “Stop measurement” button and by greying out inactive buttons). Erasing the stored open data is also possible, except for the acam firmware code which remains permanently in the chip as delivered. After erase, delivery state can always be restored from the source files provided by acam.

For details on firmware code and data hardware organization refer to Manual Volume 1: General Data and Frontend Description, Chapter 11.



Download window
zoom-in, left
side...

... and Download
window zoom-in,
right side



2.2 acam firmware features

acam offers a complete firmware with no need for additional customer code. Measurement results like flow and temperature are communicated to an external controller over SPI or UART interface just by reading out some memory cells on demand (according to the memory allocation map, see chapter 6). The pulse interface is also supported by the firmware (see section 2.6). The measurement is running permanently without any need of external control. Typical properties of the firmware are:

- Firmware NVRAM usage (of 4kB available): 2.9 kB
- RAM usage (of 176 x 32b-Words):
 - ~ 96 -88 words permanently used
 - ~ 57 words temporarily used
 - ~ 23 - 31 words free / unused
- Firmware data usage (of 128 x 32b-Words):
 - 20 words configuration (always)
 - ~66 words used
 - ~42 words free
- Expected total current consumption (example):
 - ~8 uA @ 8 Hz flow meas. rate
 - ~4 uA @ 2 Hz flow meas. rate
- Flow measurement can be freely configured
- Full calibration within legal-for-trade specifications (depending on spool piece), either using a standard piece-wise linear correction or acam's proprietary method.
- Temperature measurement for cold water meter from sound speed. Works up to 60 °C.
- Optional: 2-wire or 4-wire PT 500/1000 temperature measurement for heat meter application.
- Optional internal chip temperature measurement.
- Zero flow detection down to 0.5 l/h for Q3=4000 l/h.
- Full communication (input and output) over SPI or UART; flow volume and error output also over standard two-wire pulse interface.
- Prepared for two point calibration (zero flow and high flow @ room temperature) in series production.
- Bubble detection and error detection included to recognize wrong measurements.

The subsequent sections describe in detail how TDC-GP30 is configured and operated. Every single step is supported by software and template files, but all these steps are necessary to customize TDC-GP30 and its firmware. In overview, the necessary **development steps** are:

1. Define the suitable chip configuration for your particular flow meter system.
For details on chip configuration see Manual Volume 1: General Data and Frontend Description and Volume 3: User Manual.
2. Decide for the desired firmware configuration according to your needs.
For details see chapters 3 and 7.

3. Characterize your system and do a master calibration for the particular type of spool piece. See an overview on section 2.5 and for details Manual Volume 5: Description Calibration Engine.
4. Scale the master calibration according to calibration measurements of each spool piece. See an overview on section 2.5 and for details Manual Volume 5: Description Calibration Engine.
5. Write both chip and firmware configuration as well as the individual calibration into a firmware data file and store this file into the particular chip which operates this spool piece. For details see section 2.4.

Of course, these steps will be repeated recursively during development. **In production**, only one individual calibration step is needed to define and store an individual calibration into each spool piece. Then TDC-GP30 will be controlled typically by the microcontroller of the particular flow meter system, and all chip communication as well as the calibration scaling procedure will be done over that system.

With the firmware running autonomously, the customer has the following choices for data readout:

- All results as well as all messages (operation mode messages or error flags) are stored in specific RAM cells. Details on the organization of results data are given in section 6.1.
- Typically, the desired results from the firmware are read out over SPI or UART interface. Access to these RAM cells is possible after any measurement and is done independent of the chip's firmware – it requires no programming work on the chip. Communication over those interfaces can be organized in various ways, some proposals are discussed in section 1.3. Please refer to the Manual Volume 1: General Data and Frontend Description and Volume 3: User Manual for details on remote communication.
- Alternatively, a GP30-based flow meter can directly replace a mechanical device when using the built-in standard pulse interface for flow volume counting. External data display, storage and further evaluation is done in the same way as with mechanical devices, in the simplest way by a counter.
- Finally, particular error messages can be configured to issue an interrupt to initiate communication (see section 3.5). While many error events are handled by the firmware, some (unexpected ones) should be monitored by the external controller, for example checksum errors which indicate chip memory failure. The external controller should also monitor the chip's real time clock to identify preceding reset events, for example watchdog resets (though also unexpected). See chapter 5 for more details.
The pulse interface can signal errors, too (see section 3.1.4).

The fastest way to proceed for an experienced user is surely to use the provided template files and modify them according to the summary lists in chapters 6 and 7. The following sections and chapters 3 to 5 explain details on that process.

2.3 Setup and customization

When a flow meter system is designed around TDC-GP30, it is usual to operate in the beginning remotely in time conversion mode. This is done to define the basic setup and configuration for the measurement operation, like TOF rate, number of hits, first hit level and so on. For details see Manual Volume 1: General Data and Frontend Description and Volume 3: User Manual.

This preparation phase leads to a customized chip and measurement configuration, which is adapted to the particular flow meter setup and application. With such a customized configuration at hand, operation of the firmware can be started easily.

When the firmware is purchased by the customer, TDC-GP30 chips will be delivered pre-programmed with some standard-configuration. The measurement starts running automatically when the chip is powered on. The chip can be operated with the customized configuration in two ways:

- 1) Customize configuration temporarily (for quick tests in development): In principle, the customized configuration can be loaded by PC-software or microcontroller as usual. But the running firmware will restore its stored configuration automatically – this will happen at latest after one hour of operation, in some cases even immediately. Thus, to test (temporarily) with a customized configuration, it is safer to switch off firmware operation by disabling post processing. The watchdog must also be disabled to prevent watchdog resets after typically 13 s, which would again restore the stored configuration.
- 2) Store customized configuration permanently to the chip: If the chip should keep the customized configuration permanently, this configuration has to be stored in the firmware data of the chip. The following section describes how this is done in detail. With the desired configuration stored, TDC-GP30 will load this configuration automatically at power-on and after system reset, and the acam firmware will restore it latest every hour.

After first tests, it is in any case necessary to create a firmware data file which contains chip configuration, firmware configuration and individual calibration data for the particular spool piece. The next section presents in detail how customized firmware data files can be created.

Note that some configuration settings will be modified or enforced by the acam firmware. See sections 3.1.2 and 3.1.4 for details.

2.4 Creating and downloading a firmware data file

The evaluation package for the GP30 firmware contains a template firmware data file named “GP30Y_A1.A2.11.03.dat” (or higher version numbers in later releases). It is initialized with calibration data and configuration for a typical DN20 spool piece with an ultrasonic flow measurement length of 0.06 m. This file should be used as template for a customized firmware data file. The meaning of the file entries is explained throughout this and the next chapter.

Three steps are necessary to customize this file:

1. Store the custom chip configuration in the file (see section 2.4.1).
2. Customize the firmware configuration parameters (see section 2.4.2).
3. Also store the individual calibration data for the particular flow meter in the firmware data file.
In production, each flow meter will have its own firmware data file, containing the individual calibration data of this flow meter (see section 2.4.3 and Manual Volume 5: Description Calibration Engine).

Finally, with all custom parameters written to the firmware data file, this file is downloaded to the particular TDC-GP30 chip of the flow meter. After this last step, configuration and calibration data is permanently stored on the chip and automatically loaded or restored.

The subsequent sections describe the single steps in more detail.

2.4.1 Storing the chip configuration in the file

With the description of the parameters in the file at hand (see chapter 7), this step could be done manually, by modifying the memory cells which correspond to configuration registers (cells 108 to 123, cell addresses 0x16C to 0x17B). The TDC-GP30 PC software makes this step easier:

- Set the PC software to the desired configuration. You can easily test the performance by downloading this configuration directly to the chip (switch off post processing and the watchdog to keep the downloaded configuration, else an operating firmware may overwrite it).
- Go to Firmware/Firmware Download. Stop the measurement. You may open a template firmware data file or read the FW data stored on the chip as template. Now you can click “From GUI to FWD2”. This transfers the current configuration settings of the software into cells 108 to 122. A few things have to be considered:
 - Post processing must be switched on to run the firmware. If you have switched it off for testing the configuration, switch it on again before clicking “From GUI to FWD2”.
 - TOF rate is not stored in the configuration. If the chosen TOF rate is 1, nothing needs to be done. Else, write your chosen TOF rate as hex number into B3 of **FWD_USM_PRC** (cell 116). Example: TOF rate is 8=0x8, or TOF rate is 20=0x14; **FWD_USM_PRC** is 0x00002824. Change it to 0x08002824, or to 0x14002824.
 - Change cell 108 to a number not equal zero (usually 0x95659C6A – this is an arbitrary test value). This enables the hourly configuration refresh.

- Click “Set bootloader release code”. This sets cell 123 to the value 0xABDC7654. This is the bootloader release code. With that value stored in cell 123, the chip will transfer the configuration stored in cells 108 to 122 to the actual configuration registers of TDC-GP30 at power-on and after resets.
- The first hit levels are also not stored in the configuration registers. Define the first hit levels through the firmware configuration. If you click “1st Hit Level to FWD2”, your current value will be transferred to cells 93 and 107, but this will not be the final definition. See chapter 4 for details on first hit level definitions.

A firmware data file with the modifications described above can be downloaded to the chip. Then TDC-GP30 will automatically move the stored configuration into its configuration registers at startup or at reset, and the firmware will refresh the configuration settings every hour. Download to the chip can be done via remote commands (see Manual Volume 3: User manual, section 6.3.3) or using the download window of the PC software (see PC software documentation).

Note that some configuration settings will be modified or enforced by the acam firmware. See sections 3.1.2 and 3.1.4 for details.

2.4.2 Customizing firmware configuration parameters

This step is not supported by PC software yet and has to be done manually by changing entries in the firmware file. The meaning of the individual parameters are explained in detail in the subsequent sections – see chapter 7 for a complete overview of parameters. Firmware operation configuration is mainly done in register **FWD_FW_CONFIG** (cell 106, address 0x16A). The functions controlled by this register are described in detail in chapter 3 and summarized in section 7.1.

A convenient way to create a customized firmware data file is to use the template firmware data file “GP30Y_A1.A2.11.03.dat” (or higher version later) and to modify the firmware parameters. It is recommended to check all parameters described in chapter 7; usually many settings can remain unchanged, but others like e.g. the low amplitude limit (**FWD_R_AM_MIN**, cell 85, address 0x155) should be adapted to the particular spool piece.

With parameters and configuration register **FWD_FW_CONFIG** checked and, where desired, adapted, the customized firmware data file should be stored and can be downloaded to the chip. In combination with an updated chip configuration (see preceding section), this file will set the chip into the desired measurement and firmware operation. The last missing adaption for a fully customized firmware data file is then a suitable calibration, as discussed in the next section.

2.4.3 *Creating and storing individual calibration data (Overview)*

Calibration data for operation with acam firmware can only be generated through the calibration engine, as described in detail in Manual Volume 5: Description Calibration Engine. The calibration engine uses an initial firmware data file, ideally a customized file as described in the two preceding sections. For the next actions, development phase and production phase must be distinguished:

- In development phase, a sufficiently high number of measurements is needed to characterize the flow meter design and to generate a master calibration. This should be done with the identical configuration as defined in section 2.4.1.
- The calibration engine stores this master calibration into the initial firmware data file. This is the first (nearly) complete firmware data file to totally configure the system as a calibrated flow meter.

The following points are the only actions that need to be done in production phase:

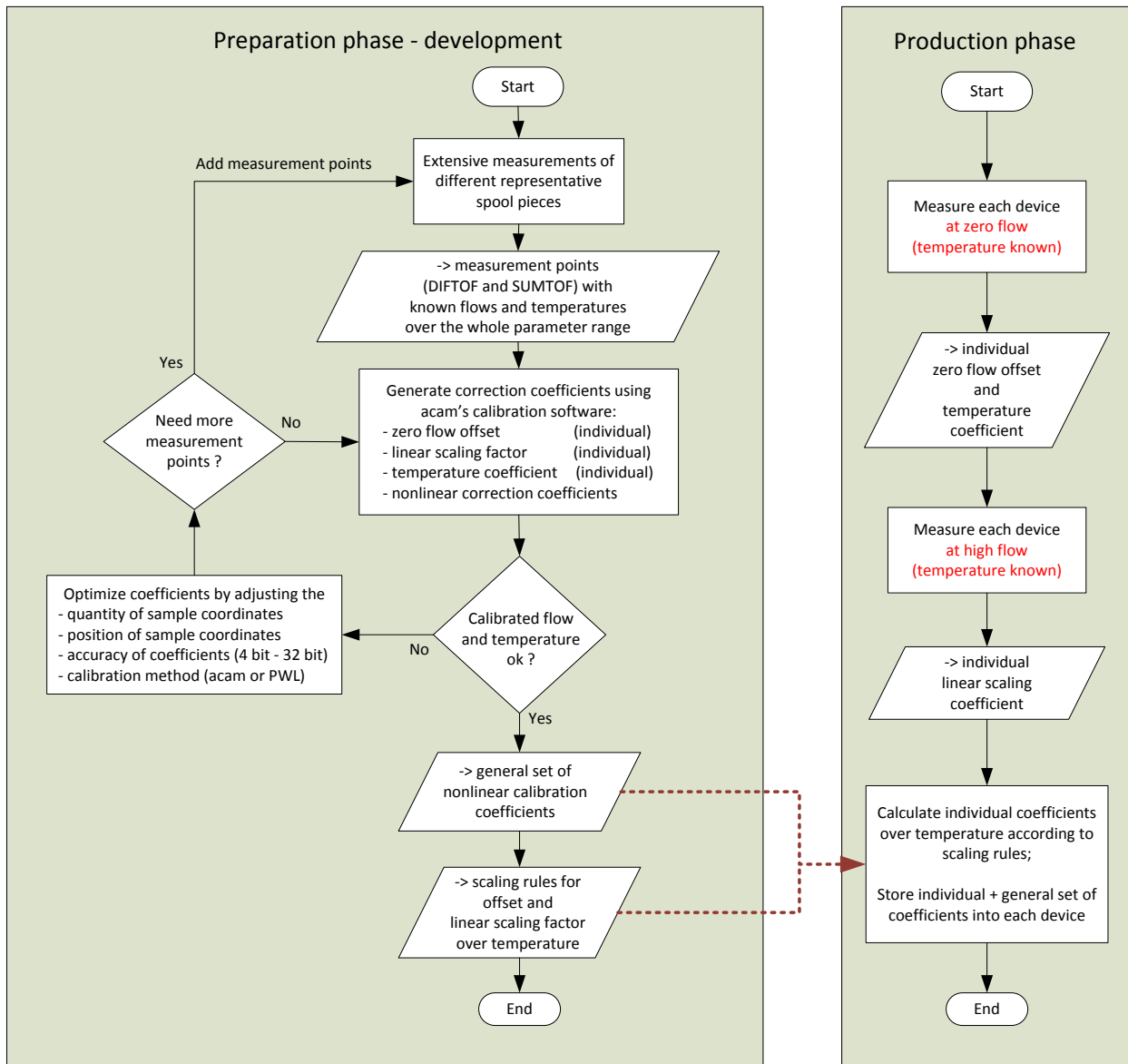
- Do calibration measurements for each particular flow meter. This will be typically one zero flow measurement and one high flow measurement.
- Scale the master calibration from production phase, using these measurements, according to the scaling rules in Manual Volume 5: Description Calibration Engine. This requires as input the customized firmware data file, with the master calibration for this flow meter system stored.
- Store the modified calibration values in a copy of the firmware data file. This file is then the individual calibration file for this particular flow meter. So this process creates one file for each flow meter which is being calibrated.
- Finally, the checksums should be calculated and stored to the files (cells 124 to 127, addresses 0x17C to 0x17F, see section 3.4). Storing the right checksums permits regular integrity checks of the stored data.
- The individual firmware data files created in this process should then be downloaded to the corresponding flow meter chips. This is the final step to get a calibrated flow meter with TDC-GP30 in operation.

The five actions in production phase listed above may of course be done in one process by an external controller, without even storing a separate individual firmware data file.

2.5 Calibration process in development and production

The customer has to do the calibration and store calibration and configuration data in the firmware data memory. acam supports the procedure of generating this data through its calibration engine.

The procedure is sketched in the following flow charts:



In practice, this means that the customer has to do a thorough calibration, actually a characterization, **in development phase** on some representative devices. These initial calibrations require a number of measurements at flows and temperatures, sufficiently large to characterize the devices and covering the complete range of application. The calibration coefficients gained by this preparation (the master calibration) are adapted **in mass production phase** using dedicated scaling rules and based on only two measurements at zero flow and high flow and at an arbitrary (but known) temperature.

The accuracy of the applied calibration depends on the particular spool piece and its production stability. To achieve good results with this (or any other) calibration, the following should be considered:

- The production of the spool piece must be sufficiently repeatable:
 - no measurement deviations due to device tolerances beyond accuracy limits,
 - linear scaling over temperature is sufficiently comparable among devices,
 - non-linear behavior is sufficiently comparable among devices.
- Tolerance and parameter changes due to aging effects must be limited as well.
- Influence of uncontrolled parameters must be insignificant (e.g. housing and environment temperature).

If above criteria are not fulfilled, more individual calibration in series production and/or additional measurements of uncontrolled parameters are needed. **Note that such enhanced methods are currently not implemented in acam's firmware.** Enhanced calibration methods thus require a dedicated firmware, typically developed by the customer himself (see chapter 10).

Please also consider possible influences of the water quality on the measurement results.

acam's firmware supports a complete linear calibration and offset correction, and two different calibration schemes for nonlinear calibration. One is based on the well-known piecewise linear (PWL), the other incorporates a proprietary method which needs less coefficients and is inherently smooth. It depends on the actual application which one is preferable, customers who order acam's firmware can select any of them.

The calibration engine supports the linear calibration as well as both nonlinear methods. It provides automatic calibration coefficients generation and optimization, and it is able to store the results directly into firmware data. An overview description of linear and nonlinear calibration parameters, of their determination and optimization, and of their usage in mass production, can be found in section 3.2.

2.6 Configuring and using the pulse interface

The built-in pulse interface of TDC-GP30 can be used to read out flow volume values in the same way as in a mechanical spool piece. With acam firmware, only the number of pulses per liter and the maximum flow needs to be defined (FWD cells 91 and 92). The number of pulses should be chosen such that it does not exceed 100 pulses per second at maximum flow. The parameters are described in section 3.1.4. The configuration register **CR_PI_E2P** also contains pulse interface parameters, but they are overwritten by the firmware. With acam's firmware, pulse width is automatically chosen as wide as possible, and pulses are configured to appear in regular cycles. Only the definition of output ports need to be done in configuration register **CR_GP_CTRL** (see Manual Volume 1: General Data and Frontend Description). A typical definition would be GPIO0 as pulse output and GPIO1 as direction output.

With a configured pulse interface, the measured flow volume just has to be added up by counting pulses externally. This is fully compatible to the pulse interface of mechanical water meters.

acam's firmware can be configured to signal error conditions over the pulse interface, too (see section 5.5). In case of error, the pulse output goes permanently high, and the direction output toggles with the TOF measurement frequency. This creates no additional pulse count and can be easily identified by some external readout device or master controller. With bit 22 of the firmware configuration register **FWD_FW_CONFIG** it can in addition be chosen if a no-water situation should also be signaled as error or not.

The load resistance applied to the pulse interface can have a significant influence on current consumption, since in case of high flow, pulses would permanently drive current through this load. With a load impedance of 1 MOhm, the maximum additional current consumption is in the range of 1.5 μ A.

3 Setting of acam firmware parameters in firmware data (FWD)

Most of the values in various cells of the firmware data should be customized or at least checked. Chapter 7 gives a complete overview of all firmware data parameters. The quickest start with acam firmware may be checking the parameter table there and use it to customize the template firmware data file provided by acam. If necessary, more detail information can be found in the following sections, which discuss the overview table of chapter 7 in different segments, ordered for their particular function. Some sections just refer to later chapters or different Manual Volumes where major functionalities are explained in detail.

The major register for operation control is **FWD_FW_CONFIG** (cell number 106). Its control bits are summarized in section 7.1. In the following sections, only the currently relevant bits of this register will be noted.

3.1 Parameters for general operation control

3.1.1 Major firmware operation control register and volume storage protection

The most important register which controls most firmware operation options is **FWD_FW_CONFIG**, which is described throughout the next sections and in summary in section 7.1.

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
106	0x16A	FWD_FW_CONFIG	acam firmware configuration register; see section 7.1	bits

One bit of this register is for general operation control purposes: Bit 29 set to “1” lets the firmware apply a protection for stored flow volume. Then the calculated flow volume is stored in three pairs of RAM cells, such that a correction can be applied when one of them contains corrupted data (for any reason). When all three flow memories contain different data, an error flag is raised.

With this flow volume storage protection, it is not easily possible to change the stored volume. Switch this option off when the stored volume should be changed intentionally.

Bit	Description of function in FWD_FW_CONFIG register	Format	Default
29	BNR_FWCONF_VOL 0: Don't apply flow volume storage protection 1: Apply flow volume storage protection	BIT	b0

3.1.2 Parameters for TDC-GP30 hardware configuration

Hardware configuration is most conveniently defined and tested using the GP30 PC software. It is described in detail in Manual Volume 1: General Data and Frontend Description, section 7.3. Values for configuration data are stored in FWD cells 108 to 123. A few remarks are necessary:

- TOF rate is not stored in TDC-GP30 configuration registers. The firmware uses B3 of cell 116 (**FWD_USM_PRC**) to define TOF rates above 1. TOF rate 0 is not permitted with unmodified acam firmware.
- FHL values are not stored in TDC-GP30 configuration registers. For FHL setting and regulation see section 4.2.
- When operating with acam firmware, the watchdog is always enabled. Cell 108 (**FWD_R_CD**), which would without firmware be reserved for the watchdog disable code (0x48DBA399), has a different function: Watchdog is always enabled, but with 0x00000000 in cell 108 the firmware does not refresh the configuration and other permanent settings hourly (see below; actually the firmware sets the RAM part of this cell to zero and recognizes a recall by values unequal zero). Setting cell 108 to zero is helpful during development, when the configuration under test is frequently varied. But in production a number unequal zero should be used (proposed: 0x95659C6A, a standard test value).
- One more configuration setting is enforced by the firmware, no matter what is written in **CR_MRG_TS** or **FWD_MRG_TS**: The checksum timer is always set to “hourly”, such that every full hour a data recall is done. This restores the RAM part of the NVRAM memory. With cell 108 unequal 0x00000000, the firmware also restores the configurations and other permanent settings after recall. This is a safety measure in case RAM data was corrupted for any reason.
- When setting **TM_RATE** = 1 (sensor temperature measurement always), communication problems may appear due to a steady data recall issued by the firmware. For proper communication handling in this case, please contact support.

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
108 - 123	0x16C - 0x17B	<i>Initial values for configuration registers</i>	If configured by FWD_FW_RLS (cell #123), the values of these cells are copied to configuration registers by the bootloader at startup and after reset See manual Vol.1.	
108	0x16C	FWD_R_CD	Watchdog disable code Without acam firmware, 0x48DBA399 disables the watchdog while any other code enables it. With acam firmware, the watchdog can't be disabled, but 0x00000000 disables configuration restore after recall. Proposed stored value 0x95659C6A	bits
109	0x16D	FWD_PI_E2P	Configuration data for CR_PI_E2P	bits
110	0x16E	FWD_GP_CTRL	Configuration data for CR_GP_CTRL	bits

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
111	0x16F	FWD_UART	Configuration data for CR_UART	bits
112	0x170	FWD_IH	Configuration data for CR_IH	bits
113	0x171	FWD_CPM	Configuration data for CR_CPM	bits
114	0x172	FWD_MRG_TS	Configuration data for CR_MRG_TS	bits
115	0x173	FWD_TM	Configuration data for CR_TM	bits
116	0x174	FWD_USM_PRC	Configuration data for CR_USM_PRC ; acam firmware interprets B3 as TOF_RATE; see chapter 3	bits
117	0x175	FWD_USM_FRC	Configuration data for CR_USM_FRC	bits
118	0x176	FWD_USM_TOF	Configuration data for CR_USM_TOF	bits
119	0x177	FWD_USM_AM	Configuration data for CR_USM_AM	bits
120	0x178	FWD_TRIM1	Configuration data for CR_TRIM1 ; set to 0x84A0C47C	bits
121	0x179	FWD_TRIM2	Configuration data for CR_TRIM2 ; set to 0x401725CF	bits
122	0x17A	FWD_TRIM3	Configuration data for CR_TRIM3 ; set to 0x00270808	bits
123	0x17B	FWD_FW_RLS	Bootloader release code 0xABCD7654 activates the bootloading process after startup and system reset	bits

3.1.3 Parameters which define operation with pure water

Four parameters are used to describe the velocity of sound in pure water. They also define acceptable velocity limits for error handling, see section 3.4. These parameters should not be changed when operating with water. For other media, please contact support.

FWD cell #	FWD cell Address	Variable name	Description and (if applicable) default value	Format
74	0x14A	FWD_SOUND_VEL_MAX	Maximum of speed of sound in m/s Default value for water 0x00061400	fd8
75	0x14B	FWD_1_BY_A	Medium constant Default value for water 0x002CA2E2	fd16
76	0x14C	FWD_CONST_C	Medium constant Default value for water 0x000F6C3A	fd24
77	0x14D	FWD_THETA_MAX	B3/B2/B1: Temperature at maximum speed of sound in °C / B0: minimal speed of sound Default value for water 0x004A002B	fd16 / fd-5

3.1.4 Parameters which define pulse interface operation

The outputs used for pulse interface are defined by hardware configuration in configuration register **CR_GP_CTRL** (see Manual Volume 1: General Data and Frontend Description). It is enabled and configured in **CR_PI_E2P** (also see Volume 1). However, most configuration settings in this register are being overwritten by the acam firmware to simplify pulse interface configuration. FWD cells 91

and 92 define the number of pulses per liter and the permissible maximum flow, and the pulse interface settings are chosen such that a maximum pulse width is generated while not producing double pulses. The number of pulses per liter can be chosen such that a minimum pulse period of 10 ms should not be undercut. The maximum flow in cell 92 is also used to limit flow values in error case (measured flow never higher than double maximum flow, see section 3.4).

FWD cell #	FWD cell Address	Variable name	Description and (if applicable) default value	Format
91	0x15B	FWD_R_PULSE_PER_LITER	Pulse interface: Number of pulses per liter	fd0
92	0x15C	FWD_R_PULSE_MAX_FLOW	Pulse interface / maxflow error limit: Maximum permissible flow in l/h; see section 2.6	fd0
109	0x16D	FWD_PI_E2P	Configuration data for CR_PI_E2P	bits
110	0x16E	FWD_GP_CTRL	Configuration data for CR_GP_CTRL	bits

The firmware configuration register **FWD_FW_CONFIG** contains two bits for pulse interface control: Bit 21 determines if error should be signaled over the pulse interface at all, and bit 22 configures if no-water should be signaled as an error over the pulse interface. Error is signaled over the pulse interface by setting the pulse port to “permanently high” while the direction port toggles.

Bit	Description of function in FWD_FW_CONFIG register	Format	Default
22	BNR_FWCONF_PI_ERROR 0: Don't signal no-water as error on pulse interface 1: Also signal no-water as error on pulse interface	BIT	b0
21	BNR_FWCONF_NO_PI_ERR 0: Signal error on pulse interface as configured in bit 22 1: Never signal error on pulse interface	BIT	b0

3.2 Calibration parameters

All calibration parameters are generated by the calibration engine, see Manual Volume 5: Description Calibration Engine. Cells number 58, 59 and 62 – 73 have to be adapted to individual spool pieces by 2-point calibration. Note that the address range for PWL coefficients can be selected through B0 of **FWD_FW_CONFIG** (see below) to optimize memory space needs. Usually only one calibration method is used, and the coefficients of the other one are not stored. But to some extent it is possible (if there is still enough space for all PWL coefficients) to have both types of calibrations stored, such that a simple comparison can be done just by switching between acam and PWL method.

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
7 - 53	0x107 - 0x135	<i>Unused or used for nonlinear calibration coefficients</i>	Typical usage see below (configurable)	
16 - 41	0x110 - 0x129	(PWL coefficient table)	(optional) Typical position of PWL calibration coefficients table, generated by cal. engine	-

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
42 - 53	0x12A - 0x135	(acam coefficient table)	(optional) Fixed position for acam calibration coefficients table, generated by cal. engine	-
54 - 73	0x12A - 0x135	Linear coefficients table	Linear coefficients table, all generated by cal. engine; values in cells # 58, 59 and 62 – 73 have to be adapted to individual spool pieces by 2-point calibration	-
54	0x136	FWD_R_TEMP_TC1	1 st temperature for linear calibration in °C	fd16
55	0x137	FWD_R_TEMP_TC2	2 nd temperature for linear calibration in °C	fd16
56	0x138	FWD_R_TEMP_TC3	3 rd temperature for linear calibration in °C	fd16
57	0x139	FWD_R_TEMP_TC4	4 th temperature for linear calibration in °C	fd16
58	0x13A	FWD_R_TOF_OFFSET	Offset time for SUMTOF in raw TDC units	fd0
59	0x13B	FWD_TOF_DIFF_CAL	DIFTOF at high flow calibration point in raw TDC units	fd0
60	0x13C	FWD_DIST_WITH_FLOW	Ultrasonic sound path length along flow in m	fd16
61	0x13D	FWD_DIST_NO_FLOW	Ultrasonic sound path length w/o flow in m	fd16
62	0x13E	FWD_R_ZERO_OFFSET_TC2	Zero flow DIFTOF at TC2	fd16
63	0x13F	FWD_R_ZERO_OFFSET_TC3	Zero flow DIFTOF at TC3	fd16
64	0x140	FWD_R_ZERO_OFFSET_TC4	Zero flow DIFTOF at TC4	fd16
65	0x141	FWD_R_O_SLOPE_TC12	Zero flow slope between TC1 and TC2	fd16
66	0x142	FWD_R_O_SLOPE_TC23	Zero flow slope between TC2 and TC3	fd16
67	0x143	FWD_R_O_SLOPE_TC34	Zero flow slope between TC3 and TC4	fd16
68	0x144	FWD_R_F_SLOPE_TC12	Proport. factor slope between TC1 and TC2	fd16
69	0x145	FWD_R_F_SLOPE_TC23	Proport. factor slope between TC2 and TC3	fd16
70	0x146	FWD_R_F_SLOPE_TC34	Proport. factor slope between TC3 and TC4	fd16
71	0x147	FWD_R_F_OFFSET_TC2	Proportionality factor F at TC2	fd16
72	0x148	FWD_R_F_OFFSET_TC3	Proportionality factor F at TC3	fd16
73	0x149	FWD_R_F_OFFSET_TC4	Proportionality factor F at TC4	fd16

The firmware configuration register FWD_FW_CONFIG switches between PWL and acam calibration method with bit 31. The two lower bytes of this register define PWL coefficients: B0 (bits [7:0]) sets the starting address of the PWL block for memory optimization, while B1 (bits [15:8]) defines an exponent which scales the PWL coefficients for maximal numerical accuracy. Bit 30 can be used when a water temperature sensor is available. Then the sensor measured water temperature may be used for calibration tables instead of a temperature calculated from flow.

Bit	Description of function in FWD_FW_CONFIG register	Format	Default
31	BNR_FWCONF_PWL 0: Apply acam calibration method 1: Apply PWL calibration method	BIT	b0

Bit	Description of function in <i>FWD_FW_CONFIG</i> register	Format	Default
30	BNR_FWCONF_TSENS 0: Use temperature value from flow meas. for calibration coefficients 1: Use temperature value from hot sensor meas. for calibration coeffs.	BIT	B0
15:8	PWL_EXP (optional) Exponent of scaling factor for PWL coefficients: Scale up each value by $2^{(PWL_EXP)}$	UINT [7:0]	0x01
7:0	PWL_ADDR (optional) Start address of PWL coefficients table in FWD (without leading address bit 8, which is always 1 in FWD addresses)	UINT [7:0]	0x10

3.3 Parameters for zero flow and negative flow operation

In zero flow case, the acam firmware can switch to an operation mode with reduced measurement rate and thus reduced current consumption. The following four parameters control this operation: The raw DIFTOF value in cell 94 is a limit for skipping calculation, with an absolute DIFTOF below that value no calculation is done and the flow is considered zero. In contrast, the limit in cell 95 is given as a flow, for example 0.5 l/h. If the actual averaged flow is below this value, the chip switches to zero flow operation: The TOF rate is increased by the factor defined on cell 89. In other words, the measurement rate is reduced by this factor. As soon as the average flow exceeds the value in cell 95, the chip switches back to full operation. Or, at the first single measurement (not averaged) which exceeds the value in cell 95 by a factor of 8, the chip also immediately returns to full operation. The flow averaging factor in cell 90 defines the range of averaging (actual averaging filter length is 16 times the value in cell 90, which must be a power of 2). This way the level of remaining noise after averaging can be controlled. A value of $0x00000010 = 16$ is proposed, which means average flow is calculated as the arithmetic mean over the last $16 \cdot 16 = 256$ measurements.

If a negative flow is defined in cell 93, any reverse flow exceeding this value is not added to flow volume, but stored in cells 0x04F to 0x051 (see section 6.1). If no reverse flow should be counted, this value is typically set to a small negative flow. Don't set it too small, else negative values from noise at zero flow will not average out to zero, but will leave some positive offset in flow volume.

If a negative flow limit is used, cells 0 and 1 are internally utilized for intermediate values and can't be used for other purposes.

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
0	0x100	FWD_R_FLOW_VOLUME_INT	(optional): Integer part of negative flow volume in cubic meters - internally used if a negative flow limit is defined in cell # 93.	fd0
1	0x101	FWD_R_FLOW_VOLUME_FRACTION	(optional): Fractional part of negative flow volume in cubic meters - internally used if a negative flow limit is defined in cell # 93.	fd32
89	0x159	FWD_TOF_RATE_FACTOR	Factor for TOF rate scaling in zero flow case	fd0

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
90	0x15A	FWD_FLOW_AVG_FACTOR	2^N number of flow values for averaging; this factor *16 determines the total number of samples for long term averaged flow, as used for the zero flow decision.	fd0
93	0x15D	FWD_NEG_FLOW_LIMIT	Cutoff limit for negative flow in l/h; positive cell values switch off negative flow detection.	fd16
94	0x15E	FWD_R_TOF_DIFF_LIMIT	Minimum limit for DIFTOF values in raw TDC units. At lower DIFTOF , temporary zero flow is assumed and no calculation is done.	fd0
95	0x15F	FWD_ZERO_FLOW_LIMIT	Zero flow limit in l/h: When the absolute of the long term average flow is smaller, long term zero flow is assumed and the TOF rate is scaled by FWD_TOF_RATE_FACTOR	fd16

3.4 Parameters for error handling

Proper error handling is an important feature since situations where no normal measurement is possible will appear regularly, may it be no-water or bubbles. In addition, suitable error handling increases operation reliability in long term operation, which is typical for flow meters. Due to its importance, this topic is discussed in chapter 5 in detail. The current section just lists the parameters which have influence on error handling. Note that all error handling activities can be switched off either by setting the limit to zero or, when this is impossible (like in cells 74, 77 and 92), through a bit setting in **FWD_FW_CONFIG**. See end of this section for details.

The parameters in cells 74 and 77 (fixed for water) define upper and lower limits for the acceptable speed of sound. Calculated results beyond these limits indicate wrong measurements and are thus ignored, the chip keeps the last valid result instead.

Cell 78 sets the number of low amplitude measurements before a special mode for hardware check is activated. Low amplitude, as defined in cell 85, indicates either no-water or a hardware failure. To distinguish these two cases, the acam firmware enters a special measurement mode. Since in this case the measurements will be unusable, a number of wrong measurements should be counted before this mode is entered, to be sure it is not a short term distortion; proposed is $0x00000020 = 32$.

Cell 81 defines a variation limit for the current SUMTOF, compared to the average of the last 8 measurements. SUMTOF changes slowly with water temperature, such that a sudden change in value can be considered wrong. A typical reason for jumping SUMTOF values is detecting a wrong first hit (see chapter 4). It is therefore a good idea to set cell 81 to, for example, half a period of the measurement frequency, to detect such kinds of errors (at 1 MHz measurement frequency for example $0x0001FFFF = 0.5 \mu s$). Setting the cell to zero switches off this error detection.

Cell 82 defines an absolute variation limit for the high speed clock (HSC) calibration factor. It can be used as a quality check for the HSC oscillator ceramic, or to control if the chip is properly configured, e.g. when using an 8 MHz HSC. The limit is in raw TDC units, given as deviation from a

measurement of 4 low speed clock periods (which is $4 \times 30.52 \mu\text{s}$, and, as raw value at 4 MHz HSC, 0x01E84800); 0x0009C400 = 2% permissible deviation at 4MHz HSC is a reasonable choice. Setting the cell to zero switches off this error detection.

Cells 84 and 87 define two variation limits between current results from TOF_UP and TOF_DOWN measurements, the permissible difference in measured amplitudes and in measured pulse width ratios, respectively. In combination with the SUMTOF deviation limit of cell 81, these cells are used for bubble detection. Their values depend on measurement noise, so they should be chosen based on representative measurements with the actual flow meter system. The reliability of bubble detection depends strongly on a reasonable setting of these parameters. Values from the template firmware data file are, for example, 0x00200000 for cell 84, meaning 32 mV amplitude variation, and 0x0000000C for cell 87, meaning an absolute difference of pulse width ratios of 0.094. Setting one of the cell values to zero switches off the corresponding error detection.

Cell 85 defines the low amplitude limit (also see cell 78 above). The value should be chosen such that it is always exceeded by every acceptable measurement. When a lower amplitude appears while there is still water, the acam firmware will signal a hardware error after the number of measurements defined in cell 78. Setting the cell to zero switches off this error detection.

Cell 92 defines the maximum flow for the pulse interface, given as integer in l/h. It is also used to set a limit of 2x this maximum flow for calculated flow, to keep the possible influence of undetected measurement mistakes on flow volume limited. This parameter should not be set to zero, the flow limitation can be switched off, if desired, through bit 16 of **FWD_FW_CONFIG**.

Finally, hardware error flags as provided by register **SRR_ERR_FLAG** (see Manual Vol. 1) are also in use. Configuration register **CR_IEH** defines which of these error flags are activated. Firmware data cell 112 is copied to **CR_IEH** through the bootloader, so this cell must contain the desired configuration setting. For a proper function of the checksum error flags, four checksum values for comparison must be provided in cells 124 – 126. The simplest way to get the right values is to load the final firmware data file into the GP30 PC software's download window. Checksums for firmware data will be immediately calculated there (cells 124 and 125) as well as the checksum for the firmware code user part, if it is also opened (cell 126). The checksum for the firmware code acam part should be taken from acam's template firmware data file (get it into the download window by "FWA manual entry").

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
74	0x14A	FWD_SOUND_VEL_MAX	Maximum of speed of sound in m/s Default value for water 0x00061400	fd8
77	0x14D	FWD_THETA_MAX	B3/B2/B1: Temperature at maximum speed of sound in °C / B0: minimal speed of sound Default value for water 0x004A002B	fd16 / fd-5
78	0x14E	FWD_LONG_TERM_ERROR	Number of low AM measurements before hardware failure / no-water checks are done Proposed value 0x00000020	fd0

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
81	0x151	FWD_TOFSUM_VAR_LIM	Error limit for deviation of SUMTOF from former average (raw TDC units); see section 4.2	fd16
82	0x152	FWD_HSC_DEV	Error limit for HSC calibration in raw TDC units (deviation time from reference measurement of 4 LSC periods)	fd16
84	0x154	FWD_AM_DIFF_LIM	Error limit for deviation between currently measured amplitude UP and DOWN in mV see section 5.1	fd16
85	0x155	FWD_R_AM_MIN	Error limit for minimal amplitude in mV see section 5.1	fd16
87	0x157	FWD_PW_DEV	Error limit for deviation between currently measured UP and DOWN pulse width see section 5.1	fd7
92	0x15C	FWD_R_PULSE_MAX_FLOW	Pulse interface / maxflow error limit: maximum permissible flow in l/h see sections 2.6 and 5.1	fd0
112	0x170	FWD_IEH	Configuration data for CR_IEH	bits
124 - 127	0x17C - 0x17F	<i>Checksums for non-volatile memories</i>	<i>These checksums are stored for comparison to values calculated by the chip. Deviations in comparison cause checksum errors. Can be calculated by the GP30 PC software.</i>	
124	0x17C	FWD_R_FWD1_CS	Checksum Firmware Data 1	fd0
125	0x17D	FWD_R_FWD2_CS	Checksum Firmware Data 2	fd0
126	0x17E	FWD_R_FWU_CS	Checksum Firmware Code User	fd0
127	0x17F	FWD_R_FWA_CS	Checksum Firmware Code acam	fd0

The following bits in **FWD_FW_CONFIG** define error handling: Bit 27 enables the control of calculated speed of sound, and bit 16 limits to double the maximum flow. Bits 21 and 22 define if the pulse interface signals errors, and if no-water should be signaled as error, too.

Bit	Description of function in FWD_FW_CONFIG register	Format	Default
27	BNR_FWCONF_VLIM 0: Disable control of speed of sound limits 1: Enable control of speed of sound limits	BIT	b0
22	BNR_FWCONF_PI_ERROR 0: Don't signal no-water as error on pulse interface 1: Also signal no-water as error on pulse interface	BIT	b0
21	BNR_FWCONF_NO_PI_ERR 0: Signal error on pulse interface as configured in bit 22 1: Never signal error on pulse interface	BIT	b0
16	BNR_FWCONF_2MAX_NOZERO 0: Set flow to zero when exceeding 2x maximum flow and signal error 1: Flow remains even when exceeding 2x maximum flow	BIT	b0

3.5 Parameters which influence error counters and error interrupt

Details on error counters are given in section 5.3. If average error counters 1 and 2 are enabled by **FWD_FW_CONFIG** (see further below), they are configured by firmware data cells 3 (counter 1) and 4 (counter 2). There is no basic difference between these two counters. To define which error flags should be counted, set the corresponding bits in cell 3 or 4 according to the bits defined in **RAM_R_FW_ERR_FLAGS** (see section 6.1.2). The results for peak average error count/hour will appear in **RAM_ERROR_COUNT_21** in the upper or the lower two Byte, respectively. With average error counters enabled, another RAM cell **RAM_ERROR_COUNT_43** will also contain counts for any error (upper two Byte) and hardware errors (lower two Byte). Then, FWD cells 5 and 6 must be set to 0x00000000. These cells count the errors during one hour between the regular hourly recalls, which reset them to the stored 0x00000000. The two result RAM cells store the peak value reached in each of these hourly counters.

Example: Counter 1 should count bubble events and TOF timeouts, counter 2 should count no-water events. Set cell 3 to 0x00024000 and cell 4 to 0x00008000, and activate error counters (compare bits in section 6.1.2). During the first hour after activation, the numbers in **RAM_ERROR_COUNT_21** and **RAM_ERROR_COUNT_43** will rise according to the actual events - each measurement cycle with the configured error(s) is one event. Then they will remain at their maximum values until, in another hour, more events appeared. In long term, these cells will contain the long term peak value of an hourly average count. An example result may be 0x00EA0132 in cell **RAM_ERROR_COUNT_21** (234 times “no-water” and 306 bubble events and/or TOF timeouts) and 0x014C0121 in cell **RAM_ERROR_COUNT_43** (totally 332 events, of which 289 where hardware errors – probably TOF timeouts, so there were probably 17 bubble events in comparison).

Cell 83 defines the errors which may issue a synchronous firmware interrupt in a similar way as the error counter configurations above: Set the desired bits according to error flag positions in **RAM_R_FW_ERR_FLAGS** (see section 6.1.2) and switch on the synchronous firmware interrupt in **CR_IEH** and cell 112, respectively. This should be used to issue an irregular interrupt at a user-defined error condition. A typical application of such an interrupt would be if TDC-GP30 communicates its results only rarely to the system’s microcontroller, but special events should be recognized immediately. For example, writing 0x00008000 in cell 83 will issue an irregular interrupt on the interface as soon as no-water is detected.

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
3	0x103	FWD_ERROR_COUNT_CONF1	(optional) Define error flag positions to be counted in error counter 1 (see section 5.3)	bits
4	0x104	FWD_ERROR_COUNT_CONF2	(optional) Define error flag positions to be counted in error counter 2 (see section 5.3)	bits
5	0x105	FWD_ERROR_COUNT_21	(optional) Temporary storage of error counts 2 (B3,B2) and 1 (B1,B0) (see section 5.3); Set to 0x00000000 if error counters are used	int

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
6	0x106	FWD_ERROR_COUNT_43	(optional) Temporary storage of error counts 4 - each error (B3,B2) and 3 - hardware errors (B1,B0) (see section 5.3); Set to 0x00000000 if error counters are used	int
83	0x153	FWD_ERR_INTERRUPT	Define error flag positions that issue an interrupt; see section 5.4 and 6.1.2 (RAM_R_FW_ERR_FLAGS)	bits
112	0x170	FWD_IEH	Configuration data for CR_IEH	bits

Bit 28 in **FWD_FW_CONFIG** switches average error counters on or off:

Bit	Description of function in FWD_FW_CONFIG register	Format	Default
28	BNR_FWCONF_ERR 0: Disable average error counters 1: Enable average error counters	BIT	b0

3.6 Parameters which control FHL regulation

First hit level (FHL) regulation is a major function of the acam firmware and is described in section 4.2. The different FHL regulation methods are set with bits 24 and 35 of **FWD_FW_CONFIG**, see below. The following parameters are used with the different FHL regulation methods:

Cell 79 defines the “trusted” FHL for methods 1 to 4. The trusted FHL is a level where a well-defined first hit is determined at any operation condition. According to the chosen setting in bit 26 of **FWD_FW_CONFIG**, this value is either interpreted as absolute voltage (only Byte 0 is read in this case), or the two lower Bytes are interpreted as a ratio of the measured amplitude. The latter can be used to compensate production tolerances and aging, but it may cause errors when the amplitude measurement is wrong (note: Measurements which have recognized errors are never used to calculate an amplitude or to regulate FHL). Example: A value of 0x00001755 in cell 79 may be interpreted as 9.1 % of the amplitude (bit 26 = 1), say 36.5 mV of 400 mV receive amplitude, or as 74.8 mV (bit 26 = 0; $0x55 \cdot 0.88\text{mV}$).

Cell 80 defines in method 3 the nominal SUMTOF difference between normal operation FHL and the trusted FHL value. It is typically a multiple of measurement frequency periods. Method 3 is used when the well-defined first hit, determined through operation with the trusted FHL, is not suitable for stable long term operation. The trusted FHL is then only used to check consistency of the currently operating FHL with a nominal time distance, as given in cell 80, between the current first hit and the one defined by the trusted FHL.

The SUMTOF variation limit in cell 81 was explained in section 3.4 above. It has an important function in FHL regulation methods 2 and 3: Consistency of the operating FHL with the trusted FHL is checked by regularly switching to the trusted FHL. As long as the measured SUMTOF does not

deviate more than the value in cell 81 from the expected value, the operating FHL is considered consistent, and operation is proceeded without changing the current FHL.

The nominal pulse width ratio (PWR) value in cell 86 switches on PWR regulation. It is good practice to define an optimal PWR value which provides optimal clearance between those FHL values where the detected first wave changes (this happens at peak amplitudes of the neighboring waves). PWR regulation takes place every 32 measurements, but with 16 measurements offset to FHL regulation, to minimize mutual influences. Setting cell 86 to zero switches off PWR regulation.

Cell 88 contains the configuration which is temporarily copied to **CR_USM_TOF** when using FHL regulation method 4.

Cell 107 is start and fallback value of the FHL, always given in absolute mV. TDC-GP30 with acam firmware always starts with this FHL value when switched on, before starting to regulate. In case of long term error, when no FHL regulation is done, the firmware regularly switches to this FHL value to check if a valid measurement may be possible.

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
79	0x14F	FWD_FHL_USER	B1/B0: trusted FHL ratio (option B) / or B0: absolute trusted FHL, LSB=0.88mV; see section 4.2	fd16 / fd0
80	0x150	FWD_TOF_SUM_DELTA	FHL method 3: Nominal time difference (raw TDC units) in SUMTOF between operating and trusted FHL; see section 4.2.3	fd16
81	0x151	FWD_TOFSUM_VAR_LIM	Error limit for deviation of SUMTOF from former average (in raw TDC units); see section 4.2	fd16
86	0x156	FWD_PW_NOM	Nominal Pulse width for FHL option A; see section 4.2	fd7
88	0x158	FWD_TEST_CONFIG	Temporary value of CR_USM_TOF for FHL method 4; see section 4.2.4	fd16
107	0x16B	FWD_R1_FHL_VALUE	Start / fallback value of FHL, LSB=0.88mV; see chapter 4	fd0

Most of all, bits 24 and 25 of **FWD_FW_CONFIG** configure the FHL regulation method, as described in the table. Bit 26 determines if the FHL value given in FWD cell 79 is interpreted as a full amplitude ratio or an absolute voltage. With bit 23 it can be chosen if FHL regulation should happen all 32 measurements or only in error case. Regulating only in error case may increase operation stability, but it has the disadvantage that first an actual error must happen before FHL is being corrected. Finally, bit 17 configures if zero flow state is switched off while regulating FHL. This significantly speeds up retuning to the desired operation state after detecting an inconsistent FHL, but it comes at the cost of increased power consumption (which should be negligible, since in stable operation FHL should always be consistent with the desired operation).

Bit	Description of function in <i>FWD_FW_CONFIG</i> register	Format	Default
26	BNR_FWCONF_FHL_RATIO: Configuration for FHL regulation option B 0: Interpret FHL-values as fixed voltage 1: Interpret FHL-values as ratio to measured amplitude	BIT	b0
25:24	BNR_FWCONF_FHL: Configuration of FHL regulation methods 00: Method 1, fixed FHL 01: Method 2, consistency check against trusted FHL 10: Method 3, consistency check against trusted FHL with offset time 11: Method 4, consistency check against special configuration	BIT2	b00
23	BNR_FWCONF_TESTMODE: Configuration for FHL regulation option C 0: Enter FHL test mode regularly (each 32 measurements) 1: Enter FHL test mode only in error case	BIT	b0
17	BNR_FWCONF_FHL_ZEROFLOW 0: With FHL regulation active, disable zero flow state (always assume full flow as long as FHL is considered not ok) 1: Apply zero flow state independently of FHL regulation	BIT	b0

3.7 Parameters for sensor temperature measurement

Sensor temperature measurement with acam firmware is either with one or two PT1000 or PT500 sensors, or the measurement using the internal sensor. Other types of external sensors require different reference curves and may be used with modified firmware (see chapter 9). For external PT sensors, only three parameters are needed: cell 97 contains the reference resistor, for example 1 kOhm = 0x03E80000 for PT1000, and cells 104 and 105 contain the 0°C resistance ratios to reference for cold and hot sensor, respectively. These values are normally 1 = 0x00010000.

The internal sensor is not very accurate and may be operated with the nominal values given in the table below. For higher accuracy, the values for nominal resistance and resistance temperature slope in cells 97 and 100, respectively, should be calibrated (simplest calibration would be changing cell 97 to get the known temperature, changing cell 100 and even the offset value cell 96 would require the knowledge of measurements at one more temperature).

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
96	0x160	FWD_CAL_PTR_OFFSET	Reference branch offset resistance in internal reference in Ohms; typical value 0x00000000 (calibrate if desired)	fd16
97	0x161	FWD_EXT_REF_VAL	Value of external reference resistor in Ohms; typical value for PT1000: 1kOhm = 0x03E80000	fd16
100	0x164	FWD_PT_INT_SLOPE	Internal sensor resistance slope in Ohms/K; typical value 0x0029F000 (calibrate if desired)	fd16
103	0x167	FWD_PT_INT_NOM	Internal sensor nominal resistance in Ohms; typical value 0x03C20000 (calibrate if desired)	fd16
104	0x168	FWD_PTC_RATIO_INV	Nominal ratio of reference resistor to PT cold sensor resistance at 0°C; typical value 1 = 0x00010000	fd16

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
105	0x169	FWD_PTH_RATIO_INV	Nominal ratio of reference resistor to PT hot sensor resistance at 0°C; typical value 1 = 0x00010000	fd16

Bit 30 in **FWD_FW_CONFIG** determines if the sensor temperature measurement should be used to select calibration coefficients:

Bit	Description of function in FWD_FW_CONFIG register	Format	Default
30	BNR_FWCONF_TSENS 0: Use temperature value from flow meas. for calibration coefficients 1: Use temperature value from hot sensor meas. for calibration coeffs.	BIT	B0

3.8 (Optionally) Unused FWD cells

The listed FWD cells are unused or optionally unused (if some function is disabled). They may be used for any custom purpose. Parameters used in sensor temperature measurement (see preceding section) are generally available for other purposes if this function is not activated.

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
0	0x100	FWD_R_FLOW_VOLUME_INT	(optional): Integer part of negative flow volume in cubic meters - internally used if a negative flow limit is defined in cell # 93.	fd0
1	0x101	FWD_R_FLOW_VOLUME_FRACTION	(optional): Fractional part of negative flow volume in cubic meters - internally used if a negative flow limit is defined in cell # 93.	fd32
2	0x102	<i>Not used</i>	<i>not used</i>	
3	0x103	FWD_ERROR_COUNT_CONF1	(optional) Define error flag positions to be counted in error counter 1 (see section 5.3)	bits
4	0x104	FWD_ERROR_COUNT_CONF2	(optional) Define error flag positions to be counted in error counter 2 (see section 5.3)	bits
5	0x105	FWD_ERROR_COUNT_21	(optional) Temporary storage of error counts 2 (B3,B2) and 1 (B1,B0) (see section 5.3); Set to 0x00000000 if error counters are used	int
6	0x106	FWD_ERROR_COUNT_43	(optional) Temporary storage of error counts 4 - each error (B3,B2) and 3 - hardware errors (B1,B0) (see section 5.3); Set to 0x00000000 if error counters are used	int
7 - 53	0x107 - 0x135	<i>Unused or used for nonlinear calibration coefficients</i>	Typical usage see below (configurable)	
16 - 41	0x110 - 0x129	(PWL coefficient table)	(optional) Typical position of PWL calibration coefficients table, generated by cal. engine	-
42 - 53	0x12A - 0x135	(acam coefficient table)	(optional) Fixed position for acam calibration coefficients table, generated by cal. engine	-
96	0x160	FWD_CAL_PTR_OFFSETR	Reference branch offset resistance in internal reference in Ohms; typical value 0x00000000 (calibrate if desired)	fd16

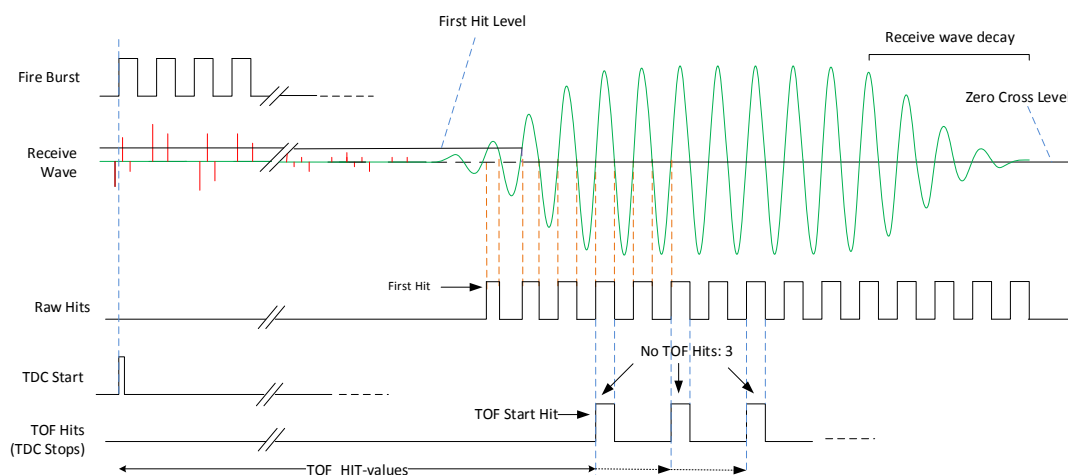
FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
97	0x161	FWD_EXT_REF_VAL	Value of external reference resistor in Ohms; typical value for PT1000: 1kOhm = 0x03E80000	fd16
98	0x162	<i>Unused</i>	<i>Unused</i>	
99	0x163	<i>Unused</i>	<i>Unused</i>	
100	0x164	FWD_PT_INT_SLOPE	Internal sensor resistance slope in Ohms/K; typical value 0x0029F000 (calibrate if desired)	fd16
101	0x165	<i>Unused</i>	<i>Unused</i>	
102	0x166	<i>Unused</i>	<i>Unused</i>	
103	0x167	FWD_PT_INT_NOM	Internal sensor nominal resistance in Ohms; typical value 0x03C20000 (calibrate if desired)	fd16
104	0x168	FWD_PTC_RATIO_INV	Nominal ratio of reference resistor to PT cold sensor resistance at 0°C; typical value 1 = 0x00010000	fd16
105	0x169	FWD_PTH_RATIO_INV	Nominal ratio of reference resistor to PT hot sensor resistance at 0°C; typical value 1 = 0x00010000	fd16

4 First hit level setting

4.1 First hit level selection criteria

The receive signal of an ultrasonic flow meter is typically a more or less slowly rising sine wave burst. Time-of-flight (TOF) measurements usually refer to the distance between the start of the fire pulse signal and a number of selected zero crossing points of the receive wave, which are interpreted as hits by the TDC. To achieve comparable TOF values, a decision has to be taken which of the various receive hits are evaluated. This decision deserves careful investigations, since it has a strong influence on measurement quality and operation stability. Relevant criteria are:

- Which noise level is related to a hit:
Zero crossings of the early waves with low amplitude create hits with higher noise.
- How strong is the influence of device tolerances on hits:
After the last fire pulse is received, the receive signal decays with a frequency which is determined not by the fire frequency, but by the resonance frequency of the particular receiving transducer. These waves and their corresponding hits are not suitable for device-independent measurements (decaying waves in picture below).
- How stably can a particular hit be identified:
For operation stability, and in particular for a correct calculation of sound velocity and temperature, a reliable identification of always the same receive hit as reference for the measurement is crucial. This so-called first hit must remain the same over the whole operation time. Flow is calculated from differences of TOF_UP and TOF_DOWN (DIFTOF), which - in well-designed systems - does not depend on the first hit, as long as the first hits for both TOFs are the same. Otherwise, the measurement is wrong and may be widely misleading. So a stable and unambiguous first hit detection is also important for the flow measurement itself.



The selection of hits for evaluation depends on all three factors and should be optimized for a given spool piece, current consumption specification and accuracy demand. With a given first hit, GP30 permits the selection of a sum of hits with configurable start hit and length. Details about available configuration options can be found in Manual Volume 1: General Data and Frontend Description, chapter 3. It should be noted that in general the first hit itself and the second hit are not suitable for evaluation since they are involved in the first hit detection process and are not exactly corresponding to a well-defined zero crossing point of the receive signal.

The remaining part of this section concentrates on the first hit detection process and options and on the regulation algorithms offered by the acam firmware.

TDC-GP30 offers a number of auxiliary measurements and settings to support stable first hit detection:

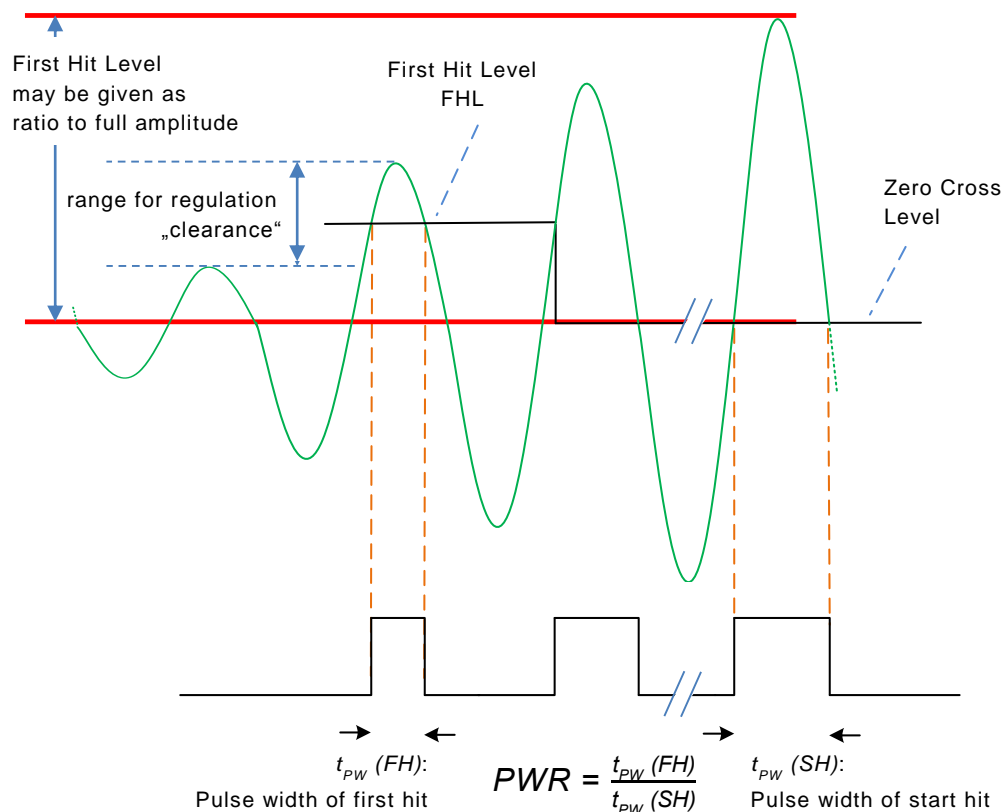
- Peak amplitude measurement of selected waves of the receive signal
- First hit pulse width ratio (PWR) measurement to optimize the quality of the first hit detection
- Automated zero crossing calibration
- First hit level setting to detect the chosen wave at some particular amplitude
- Alternatively, a start hit delay to determine the start hit not by detecting some particular first hit, but after a user-defined delay time

The last point, using a start hit delay, is a well-known method, not depending on any amplitude measurements. It usually requires accurate regulation of delay times since the TOF values, which relate to the delays, change strongly with flow and temperature. The acam firmware does not support this method since it is highly device-dependent. Customers which apply the start hit delay need to implement their own regulation schemes in their user firmware or their external controller.

By setting a fixed first hit level (FHL), a particular first hit can be detected as reference for the measurement. This detection is stable as long as the chosen FHL is sufficiently different from the peak wave amplitudes around the first hit. When it is lower than the peak amplitude of the preceding wave, one hit before the desired one will be detected, and when it is higher than the next peak wave amplitude, one hit after the desired one is detected. Thus the FHL value has a usable voltage range between these two wave amplitudes. For maximum clearance, which means optimal EMI immunity, FHL should be chosen in the middle between these two amplitudes. The pulse width ratio (PWR) measurement is an indicator for the right selection of FHL. It approaches zero for too high FHL (measured pulse width gets small close to the wave peak), and for too low FHL some maximal PWR value is reached just before the first hit detection jumps to the preceding wave. The maximum PWR is determined by the rise time of the receive signal, which means by the quality factor of the employed transducers. The optimal FHL is found where a PWR between this maximum pulse width and zero is measured. In summary, a first selection of FHL level can be done the following way:

1. Measure the first few receive signal wave amplitudes using GP30 (You can do that using an oscilloscope, but the probe head's impedance will influence the measured peak values, often too strongly).
2. For maximum clearance, select the wave at the highest amplitude difference as position for the chosen first hit (this selection may be revised for more sophisticated reasons, see point 4. below).
3. Set FHL to the middle value between these two wave amplitudes and measure the pulse width ratio. This would be the nominal pulse width ratio for the optimal FHL.

This value gives some first information about the suitability of the selection, and of the transducers as well. Values of 0.6 – 0.7 would be quite good for this nominal pulse width ratio. If the result is significantly smaller, the quality factors of the employed transducers may be too high. This causes long rise and decay times, with small differences between subsequent wave amplitudes, as well as a high sensitivity against production tolerances. Both is undesired and should be avoided.



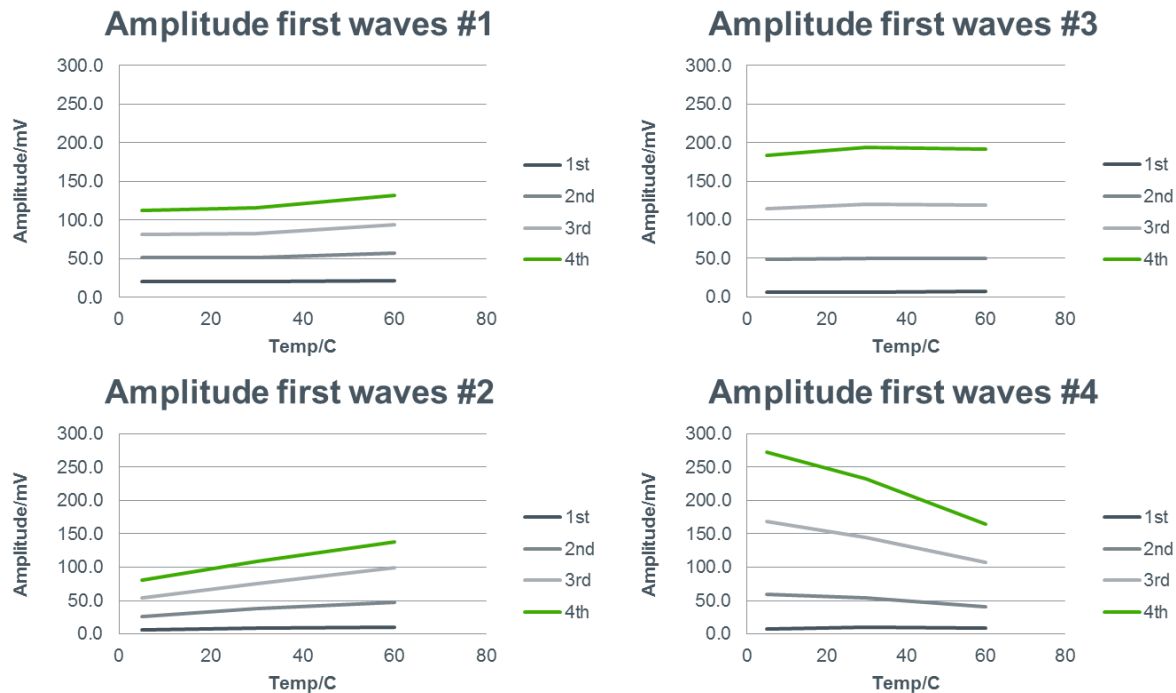
This is the first important step of the FHL selection. Until here, it is not related to acam firmware and can be used in remote control operation as well. But now it should be noted that the result of the selection process described above can change over temperature, production tolerances and aging; for some devices and transducers it may even depend on flow and pressure. The problem is to

obtain reliable and stable first hit detection under the influence of all these different factors. So a fourth step should follow:

4. Check how stable the chosen selection is against temperature, tolerances, aging, flow and pressure. It can happen that, with regard to temperature changes, the selection of a different FHL with less clearance, but better temperature stability may be preferable (see picture below).

If strong dependences of optimal FHL on actual flow or pressure are observed, the performance of the transducers should be checked critically. Usually, transducers which respond to pressure changes with strong signal changes are not optimal for flow meters.

Regarding the other influence factors tolerances and aging, only an active reaction on parameter changes can improve the situation. This may be done on customer side, for example by defining an individual FHL in relation to the receive signal amplitude, or even actively by a user firmware code or an external controller. The acam firmware has a number of regulation methods implemented which gives the user some choice for an optimized operation stability. The next section describes these methods.



These example plots demonstrate how different receive wave amplitudes - and with them the corresponding FHL levels – can vary over temperature for different spool pieces #1 - #4. Accordingly, different spool pieces require different FHL regulation methods for stable first hit detection.

4.2 First hit level regulation methods of acam firmware

As discussed above, there may be reasons to adapt the first hit level (FHL) according to environment parameters. The acam firmware offers four different regulation methods and three configuration options, to permit an optimal choice for operation stability of an individual device.

In general, all methods aim at controlling and, if necessary, adapting FHL such that the device operates stably at the chosen working point. This is particularly important after situations where no valid measurement is possible – typically after a no-water situation, but also after critical error situations like a strong appearance of bubbles.

Most of the time, the operation of an optimized device is error-free. All FHL regulation methods react slowly. This increases the time needed to return to normal operation in failure case, but adds a huge amount of operation stability in normal operation. The actual implementation is that FHL regulation is done every 32nd measurement, or even less (see option C below). This way the regulation is not confused by transients, and does not itself confuse measurement operation by sudden changes.

It should be pointed out that any regulation also adds some amount of failure probability. So the general recommendation is to examine carefully which level of regulation is needed, maybe even redesign the system to add stability, and then decide for a minimum of regulation mechanisms for highest operation reliability.

The available **regulation methods** in overview:

1. Keep FHL constant: This is of course the most stable method, but is suitable only if the spool piece behaves stably enough over temperature. Whatever additional option may be chosen, this method enforces a given FHL every 32 measurements.
2. Return to a trusted FHL in case of inconsistency: It may be advantageous to let FHL freely regulate, for example for optimized pulse width (see option A below), but then return to a less-optimal, but stably operating FHL value when problems appear.
3. Offset trusted FHL: The method also assumes that one trusted FHL level exists which ensures a desired working point over temperature and other parameter changes. But it may happen that this point has low clearance between the wave amplitudes and thus is not the best choice. Instead a different FHL is used for operation, at a given time offset, meaning a given number of hits away from the trusted level. So the method temporarily switches every 32 measurements to the trusted level to check the currently operating FHL for consistency with the user-given time offset condition. In case of inconsistency it starts regulating.
4. acam fallback method: This is the only method that requires no assumptions about any trusted FHL. For an FHL consistency check, a suitable alternative configuration setting is used, see the detail description below. The method switches every 32 measurements to the

alternative configuration to check for consistency and starts regulating when the currently active FHL appears inconsistent.

Note that this measurement causes error flags at each test measurement. Since the acam firmware neglects singular error events, this does not cause an actual measurement error.

Any of these methods can be combined with the following **options**:

- A. Regulate PWR: FHL is adapted every 32 measurements (16 measurements offset from the FHL regulation methods above) to achieve a user-defined PWR. This option is useful to optimize EMI immunity. Such a regulation must be combined with a method for consistency check (like methods 2 - 4 above), else it may get stuck at undesired FHL levels after, for example, no-water situations. Since this regulation can contradict an active FHL regulation, it is always temporarily switched off when FHL regulation gets active.
Option A is activated by setting **FWD_PW_NOM** to the desired nominal PWR value. Setting the variable to 0x0 switches off the option (see section 3.6).
- B. Define FHL as ratio to receive amplitude: Instead of defining FHL levels as fixed voltages, this option interprets FHL values as ratios to the measured amplitude. Option B is useful to automatically compensate production tolerances and aging changes. Of course this adds some amounts of uncertainty since the currently measured receive amplitude may be corrupted, for example by bubbles. The acam firmware incorporates precautions to identify corrupted measurements, it will never modify any FHL level based on a measurement which is considered questionable.
Option B is activated by setting bit **BNR_FWCONF_FHL_RATIO** in **FWD_FW_CONFIG** (see sections 3.6 or 7.1).
- C. Activate FHL regulation modes only in failure case: This option disables the regulations of method 1 - 4 as long as operation is considered correct and free of errors. It does not disable PWR regulation. Only in case of recognized measurement errors, it activates FHL regulation. The advantage of this option is that it avoids frequent intentional measurement interruptions caused by the regulation methods (most of all method 4, but also 2 and 3). The disadvantage is that FHL is not checked unless an actual error appeared. Switching on this option also prevents FHL regulation from correcting errors which are only recognized by FHL consistency checks. Thus it may happen that the device operates for long times in an undesired working point without being noticed. Decide to activate this option only after careful examination.
Option C is activated by setting bit **BNR_FWCONF_TESTMODE** in **FWD_FW_CONFIG** (see sections 3.6 or 7.1).

Formally, all options can be combined with any regulation method. It should be understood that a few combinations are not reasonable. For example activating PWR-regulation (option A) combined with fixed FHL (method 1, and without option C) will always cancel the steps of PWR regulation.

In zero flow case, the measurement rate is usually slowed down. In standard configuration (bit 17 of **FWD_FW_CONFIG** = 0), this reduction of measurement frequency is disabled whenever FHL regulation is active, to avoid extremely slow regulation.

Finally, a different FHL value is defined in FWD cell 107: **FWD_R1_FHL_VALUE** (always in mV, never interpreted as amplitude ratio) as fallback option. This is the start value when the chip is switched on. In case of TOF timeout (typically no-water, but may also be some error case), the firmware switches temporarily (every 32 measurements) to this user-defined FHL just to check if at this FHL value normal operation may resume. This is just a safety measure against unexpected cases of corrupted operation.

The parameter settings for configuration of the different FHL regulation methods and options are described in section 3.6 in overview, and in more detail in the following functional descriptions.

4.2.1 FHL regulation method 1: Keep FHL constant

This method is simple and straightforward: It enforces the user-defined FHL value in **FWD_FHL_USER** every 32 measurements (or, if option C is activated, in error case only). It offers a maximum of operation stability on firmware side, but of course it is suitable only if the spool piece behaves stably enough over temperature and other parameter changes.

Option A (Regulate PWR) can be activated, but its regulations get cancelled after another 16 measurements unless option C (Activate FHL regulation modes only in failure case) is active, too. Option B (Define FHL as ratio to receive amplitude) can be activated in addition to compensate production tolerances and aging; see comments above for general pros and cons of this option. FHL regulation method 1 is activated by setting **BNR_FWCONF_FHL** to 0x00. It only uses **FWD_FHL_USER** in addition. See sections 3.6 or 7.1 for details.

4.2.2 FHL regulation method 2: Return to a trusted FHL in case of inconsistency

This method is similar to method 1 above, but it does not generally enforce the user-given FHL value. It rather regularly checks the currently active FHL against a trusted FHL value given in **FWD_FHL_USER**. It does that by switching to a test mode every 32 measurements or, with option C active, only in error case. The criterion for consistency is the deviation between the SUMTOF measured in test mode and the average of the last eight SUMTOF measurements (with error values and outliers excluded). If the deviation is larger than **FWD_TOFSUM_VAR_LIM**, the currently active FHL is considered inconsistent, and gets replaced by the trusted FHL value.

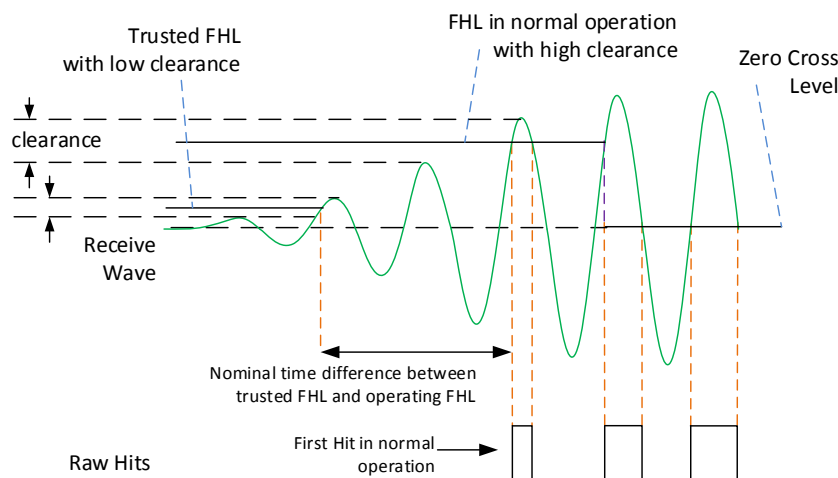
Option A (regulate PWR) can be activated and has enduring influence on the active FHL, in contrast to method 1. Thus method 2 is suitable for optimized EMI immunity, but of course only when the spool piece features at least one sufficiently temperature-stable trusted FHL with acceptable EMI clearance (see picture above).

Option B (define FHL as ratio to receive amplitude) can be activated in addition to compensate production tolerances and aging; see comments above for general pros and cons of this option.

FHL regulation method 2 is activated by setting **BNR_FWCONF_FHL** to 0x01. It uses in addition **FWD_FHL_USER** and **FWD_TOFSUM_VAR_LIM**. See sections 3.6 or 7.1 for details.

4.2.3 FHL regulation method 3: Offset trusted FHL

This method adds one more feature to method 2: Instead of checking directly against a trusted FHL, it checks for a user-defined SUMTOF-offset in **FWD_TOF_SUM_DELTA** (two times the nominal time difference in the plot below). As always, it does that by switching to a test mode every 32 measurements or, with option C active, only in error case. The check for consistency is now that the deviation of the SUMTOF measured in test mode and the average of the last eight SUMTOF measurements (with error values and outliers excluded) is closer than **FWD_TOFSUM_VAR_LIM** to **FWD_TOF_SUM_DELTA**. If the deviation to **FWD_TOF_SUM_DELTA** is larger than **FWD_TOFSUM_VAR_LIM**, the currently active FHL is considered inconsistent, and FHL regulation gets active: Depending on the sign of the deviation, the active FHL is increased or reduced until the consistency check is successful again.



The essential difference of this method compared to methods 1 and 2 is that the active FHL can be arbitrarily different from the trusted FHL. For example, the trusted FHL may be suitable for detecting the second wave as first hit, over all temperatures and tolerances. But it may feature low EMI clearance (small differences between neighboring wave amplitudes), such that a later wave with higher amplitude and higher clearance may be preferable, for example the fourth wave. Now the

corresponding FHL for the later wave may not be stable over temperature and not suitable as trusted FHL. Then this FHL regulation method 3 uses the “small” trusted FHL as reference only and regulates the active FHL such that its SUMTOF deviates by a user-defined value

FWD_TOF_SUM_DELTA from the trusted FHL, for example by 4 μ s. If, for example, the measurement frequency is 1 MHz, a SUMTOF difference of 4 μ s corresponds to a first hit which appears two periods ($2 \times 1 \mu$ s) later than the SUMTOF measured with the trusted FHL. This way, a first hit with high EMI immunity can be chosen even though it may not be suitable as trusted FHL. Still at least another trusted FHL must exist to use this method.

It should be noted that this method typically causes temporary SUMTOF and other errors when regulation is active. This happens every 32 measurements, such that there is no actual influence on measurement results.

Option A and Option B can be activated with the same implications as under method 2, see above. FHL regulation method 3 is activated by setting **BNR_FWCONF_FHL** to 0x10. It uses in addition **FWD_FHL_USER**, **FWD_TOFSUM_VAR_LIM** and **FWD_TOF_SUM_DELTA**. See sections 3.6 or 7.1 for details.

4.2.4 FHL regulation method 4: acam fallback method

This is the only FHL regulation method which does not need any assumption about trusted FHL levels. It should be tested for spool pieces which feature too strong changes of wave amplitudes over temperature. The details of the method are confidential, so this section only describes the steps necessary to employ it, without explanations of the actual meaning.

As all other methods, FHL regulation method 4 switches to a test mode every 32 measurements or, with option C active, only in error case. It temporarily applies a test configuration

FWD_TEST_CONFIG to **CR_USM_TOF** to check for the following condition:

The measurement result for the individual TOF hits **TOF_7_UP** and **TOF_7_DOWN** must be 4096 μ s. If this is not the case, FHL regulation gets active and changes FHL until the condition is met.

To define the configuration value in **FWD_TEST_CONFIG**, do the following steps (most easily with the acam PC software; the descriptions in brackets refer to the steps to be done in PC software):

- Optimize the parameter settings, FHL and receive hit configuration, of GP30 with your spool piece as usual. Let the chip operate in your desired configuration, then stop operation. Store your configuration and note down the current FHL.
- Now switch off post processing (“General control”/switch off “Enable post processing”) and zero cross calibration (“Ultrasonic measurement control”/ set “Zero cross calibration rate” to “disabled”), and set both FHLs to 0 (“Ultrasonic measurement control”/ set “first hit level up” and “first hit level down to “0”). Also disable the watchdog. Then write this configuration to the chip.

- Read the current **SHR_ZCD_LVL** value in cell 0x0D9 and add to it your chosen FHL level. Write then result back to the chip's cell 0x0D9.
- Start measurement at this point. It should be more or less the same as your measurement before.
- Now start with changing the number of ignored hits ("Ultrasonic measurement control"/ "No. of ignored hits") until a zero appears for TOF8_UP/_DOWN (configuration bit **TOF_HITS_TO_FDB** must be set to 1 to store the first 8 hit values in Frontend data buffer, see Manual Vol. 1 section 7.3.11). Then change the start hit until a value of 4096 μ s appears in TOF7_UP/_DOWN (change "Ultrasonic measurement control"/ "Selected start hit..."). This is the desired measurement result in the test mode of this regulation method.
- Now check the clearance range of FHL values where this measurement result remains. This is easily done by varying 0x0D9 – the wider the clearance range, the better. You should do this check over the whole temperature range. It does not matter if the absolute FHL level changes over temperature, but the clearance range should remain at an acceptable level (say, larger than 10mV at minimum).
- You may revise your choice of FHL for a better clearance.
- When the result is satisfactory, store the current value of **CR_USM_TOF** (read chip cell 0x0CA) in **FWD_TEST_CONFIG**. When you now activate FHL regulation method 4, the consistency check in test mode regulates the current FHL such that your chosen first hit is detected as desired.

Note that this regulation method causes active error flags at each test measurement. Since the acam firmware neglects singular error events, this does not cause an actual measurement error.

Option A and Option B can be activated with the same implications as under method 2, see above. It is highly recommended to combine method 4 always with option A (PWR regulation), else there will be no fine-adjustment of FHL.

FHL regulation method 4 is activated by setting **BNR_FWCONF_FHL** to 0x11. It uses in addition **FWD_TEST_CONFIG** as test configuration. See sections 3.6 or 7.1 for details.

5 Error handling and operation safety measures

Flow meter systems have to face two typical problems: They have to handle one special operation case when there is no medium in the spool piece (**no-water** case), and they have to operate stably and reliably over long times without any maintenance. In addition, ultrasonic spool pieces suffer from bubbles which may also appear from time to time. Thus a well-designed ultrasonic flow meter device should basically handle three different types of events:

- Usual long-term interruptions of normal flow measurement, like no-water: Such events are expected to happen. They should not modify the measured results, and normal operation should be resumed quickly and reliably after the event.
- Usual short-term interruptions, like bubbles: Such events will influence the measurement results, since at some level of distortion the flow measurement will be wrong or impossible. But their influence should be kept under control: The quantity of events should be known, and they should not interrupt normal operation. As far as possible, corrections should be done.
- Unusual, even very rare failure and distortion events: Considering even very rare and improbable events is appropriate when reliable long-term operation of huge amounts of devices should be achieved without the need or possibility of regular maintenance.

This chapter discusses the tools and processes the acam firmware offers to handle the majority of such special or failure events. Regarding rare failure events, like a possible corruption of stored data or configuration, acam firmware development was done under the assumption that the chip should resume normal operation even in case any arbitrary volatile memory cell of the chip may have lost its content. It should be noted that this is not an expected event. Actually, data corruption of volatile cells was only observed at permanent supply voltage drops below 2 V. However, the firmware is designed to handle even extremely improbable cases, for example caused by some temporary strong electromagnetic interference, to achieve highest reliability for a mass production device.

In addition, error handling is strongly related to first hit regulation, since a wrong first hit selection will cause errors on the one hand; on the other hand, corrupted measurement data should never be used for first hit regulation.

5.1 Error handling

The following list of events and operation modes ranges from normal operation over typical special conditions like bubbles or no-water to unusual and unexpected severe error events. It discusses the tools and processes the acam firmware offers to handle the majority of these events, as well as proposals for additional precautions on side of the external user. The setting of corresponding parameters is explained in section 3.4.

Description	Behavior of TDC-GP30 firmware	Occurrence	External user tasks
Normal operation	Flow and temperature are calculated according to calibration, Flow volume is stored in volatile memory.	Normal operation	Further evaluation, storage and communication of results
Singular error events, caused e.g. by bubbles or EMI	Singular events are always ignored by the implemented outlier-filter (no matter if they are identified as error or not). They do not influence any measurement result. They may still be temporarily signaled by an error flag.	Frequently, depending on overall device design and application situation. With recommended circuitry on test stand, EMI events are typically never observed.	-
Short sequences of errors e.g. by bubbles	Sequences of not more than four errors are ignored not by filtering, but by replacing them with former valid measurement values	Frequently, depending on overall device design and application situation. Typically caused by bubbles.	Sequences of errors must be identified by their measurement results. User should define error limits, see section 3.4.
No-water	Recognize and indicate no-water	Normal operation situation	No-water is mainly indicated by a low amplitude. User has to set the low-amplitude limit suitably.
Long sequences of errors, e.g. by impurities or large amounts of air in the water	Recognize and indicate error situation; stop adding volume	Should be avoided by construction – no reliable measurement possible	Sequences of errors must be identified by their measurement results. User should define error limits, see section 3.4.
FHL regulation errors	Should be resolved after some regulation time, according to the chosen regulation mode.	Typically after an error situation or after no-water	User has to carefully evaluate the behavior of his spool piece and configure the suitable FHL regulation method.
Corruption of volatile data in TDC-GP30	Several possibilities: Can range from non-recognized small changes to corrupted operation, ending in watchdog reset; acam firmware issues a data recall every hour, such that corrupted volatile data is overwritten and thus revised regularly. The stored flow volume has an optional additional security storage.	Never observed even in extensive tests. Still the firmware has procedures implemented to resolve even such very improbable situations. Note that corrupted data can of course be generated by intentional or unintentional wrong input from outside.	User should in any case control the real time clock to identify resets – GP30 will not remember any volatile data after a reset, so it is important to recognize such events. It is proposed in addition to set a maximum volume/h limit on controller side to limit the influence of unrecognized errors of that type.
Corruption of permanently stored data in GP30	GP30 can't correct that, but the acam firmware regularly controls the checksums of stored data to recognize and signal such errors.	Should not be observed and has to be considered a hardware defect. May also be caused by wrong storage processes over the interface.	The checksums initially stored in firmware data must be correct to enable regular checks. User must evaluate the error flags and may try refreshing the stored data in error case.
Low voltage	GP30 indicates low voltage, if configured, by an error flag and by a low signal on GPIO6. It takes no further action and keeps operating as long as possible.	Always at the end of battery lifetime	User has to evaluate the low voltage flag or, preferably, the signal on GPIO6, and take external precautions.

In summary, the following precautions are proposed on user side to support optimal reliability and operation stability:

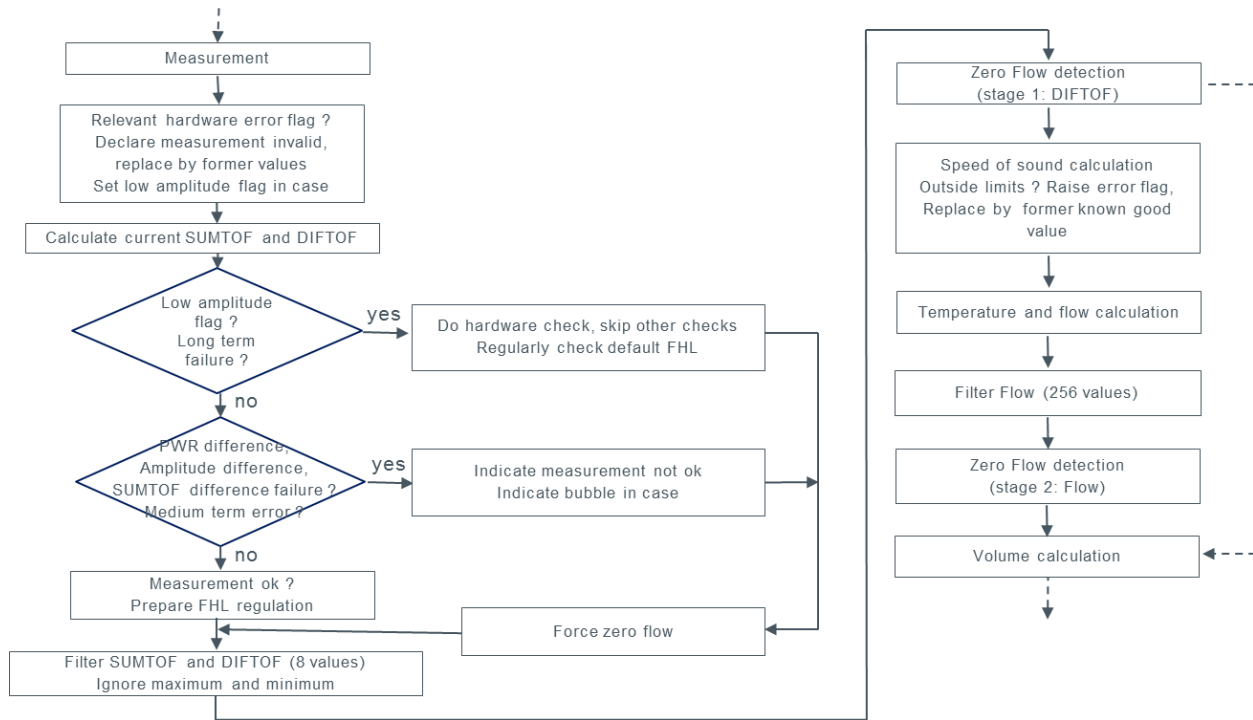
- In development and evaluation:
 - Set FHL regulation parameters and error limits carefully for stable regulation and reliable detection of bubbles and other errors.
 - Make sure to store the right checksums in FWD cells 124 to 127 (see section 3.4)
 - Select which error flags should be observed by the controller and configure error counters and interrupts accordingly. It is the free decision of the user which corrective actions, in addition to the built-in processes, may be taken. Note that acam firmware already takes precautions through its built-in processes to resolve any temporary problem.
- In operation (tasks for the controller of the flow meter system):
 - Watch the relevant error flags which indicate problems during running operation and take appropriate corrective actions (as freely defined by user as result of the evaluation).
 - Monitor the real time clock of GP30 in **SRR_TS_MIN_SEC** and **SRR_TS_HOUR**, to recognize resets (for example caused by watchdog events). After a reset, GP30 has lost all volatile data and restarts with the stored configuration. Resets may also be recognized by storing some value in an unused RAM cell and checking if the cell returned to zero.
 - The following errors can't be resolved by GP30 chip or acam firmware and must be treated (by corrective action or at least recognition) on side of the external controller: Low voltage, checksum errors, uncorrectable volume storage error.
Low voltage check is supported by TDC-GP30 through an adjustable low voltage limit and an error flag. In addition, low operation voltage is signaled through the firmware by a "low" level on GPIO6. If voltage increases again, this signal is revised every full hour.
 - In addition, it is proposed to set an upper limit to total flow volume/h to limit the influence of any unrecognized data corruption. Such events are highly unexpected, and this would just be a precaution which acknowledges the fact that even very improbable events may still happen sometimes.

The following precautions are automatically enforced by the acam firmware:

- Watchdog is active – in case of fatal errors a reset happens to resume operation
- Hourly recall is configured (with **TM_RATE**=1, recall with every measurement)
- TOF rate is read at every measurement from **FWD_USM_PRC/B3**

5.2 Error handling flow chart

The flow chart below sketches the sequence of error handling in the acam firmware.



A number of decisions depends strongly on the frequency of events and the duration of errors. Information on error counts can be retrieved from the error counters described in the next section. The following numbers are important for error handling and flow calculation as well as for first hit level regulation and zero flow detection:

- 1: Single measurements which deviate from the last seven measurements are always ignored by the averaging filter, no matter if an error flag was raised or not. Independent from that, single measurement which come with relevant error messages are in general ignored and, to some extent, replaced by former results. Such measurements are marked not o.k. At the first error-free measurement the “measurement not o.k.” flag is cleared.
- 4: After four measurement errors in a sequence, the measurement process is considered corrupted and an internal flag **BNR_MEAS_FAILURE_ALERT** (bit 10 of **RAM_R_FW_STATUS**) is set. Then flow is forced to zero until the next valid measurement. Bubble detection flag is raised when four or more consecutive measurements appeared which violated the limits given in **FWD_TOFSUM_VAR_LIM**, **FWD_AM_DIFF_LIM** and **FWD_PW_DEV** (see section 3.4)
- 8: Fixed length of the input DIFTOF and SUMTOF filter.

- 32: FHL and PWR regulations take place each 32nd measurement, with an offset of 16 measurements between FHL and PWR. This way, the regulations don't interfere mutually and with the input filter length. In consequence, all regulations are comparably slow, which increases operation stability, but also increases the time before inconsistencies are resolved by regulation.

FHL and PWR regulation is never done based on measurement results which are considered corrupted. As far as applicable, former results are utilized instead.

- **FWD_LONG_TERM_ERROR** (FWD cell 78): After this number of low-amplitude errors, the firmware switches into special configurations to distinguish no-water from hardware defects.
- 16* **FWD_FLOW_AVG_FACTOR** (FWD cell 90): This number defines the length of the flow averaging filter, and thus the noise level of the averaged flow result, in particular of zero flow. Longer filters permit lower zero flow levels (limited by the quality and repeatability of the zero flow offset calibration). Of course, longer filters also mean longer settling times for averaged flow, and consequently longer times before the decision for zero flow is taken. Note that flow volume is calculated from unfiltered flow values. Note also that switching back into full flow mode is done not only when averaged flow exceeds the zero flow limit **FWD_ZERO_FLOW_LIMIT**, but also when the unfiltered flow result exceeds this limit by a factor of 8.
- **FWD_TOF_RATE_FACTOR** (FWD cell 89): This number determines the reduction of measurements in zero flow case. With **BNR_FWCONF_FHL_ZEROFLOW** (bit 17 of **FWD_FW_CONFIG**) set to 0, the measurement rate is reset to normal operation when FHL regulation gets active, to avoid too long regulation times.
- **BNR_FWCONF_VLIM** (bit 27 of **FWD_FW_CONFIG**) set to 1 limits calculated results for speed of sound to physically reasonable values (see section 3.4, FWD cells 74 and 77). Results which violate these limits are ignored and the last valid result is taken instead. In consequence, the calculated water temperature also remains unchanged until a new valid measurement appears.
- **BNR_FWCONF_2MAX_NOZERO** (bit 16 of **FWD_FW_CONFIG**): When this bit is set to 0, the calculated flow gets limited to 2* **FWD_R_PULSE_MAX_FLOW** (FWD cell 92). Higher results are ignored, and current flow is set to zero.
- **FWD_NEG_FLOW_LIMIT** (FWD cell 93) defines a negative flow limit. When exceeded, the current flow is set to zero. For details, see section 3.3.

5.3 Error counters

To keep track of error events, the acam firmware offers functions to count errors and store peak values of errors per hour. There are two types of counters: one adds up the number of consecutive measurement error events, separated into

- Ultrasonic measurement errors (**RAM_R_USM_ERR_CTR**),
- Amplitude measurement errors (**RAM_R_AM_ERR_CTR**),
- Low amplitude events (**RAM_LOW_AM_ERR_CTR**),
- Sensor temperature measurement errors (**RAM_R_TM_ERR_CTR**),
- Task sequencer timeouts (**RAM_TS_ERR_CTR**) and
- Ultrasonic measurement error events which prevent new flow calculation (**RAM_R_FHL_ERR_CTR**)

These counters are all reset to zero at the next correct measurement. They are used for error handling control, and can be helpful for configuration check during development.

The other type of counter is optional and user configurable, and adds up all configured error events during one hour. These counters are reset for each new hour, and their peak value is stored in **RAM_ERROR_COUNT_21** and **RAM_ERROR_COUNT_43**. They can be used for statistics, long term evaluation and operation diagnostics. To switch on these counters, set bit **BNR_FWCONF_ERR** in **FWD_FW_CONFIG** (see section 7.1). Counter 4 or 3 (Bytes B3/B2 or B1/B0 in **RAM_ERROR_COUNT_43**) are configured to count all errors or all hardware defined error flags, respectively. Counters 2 or 1 can be configured by setting the bits corresponding to the error flags of **RAM_R_FW_ERR_FLAGS** (see section 6.1.2) in their configuration registers **FWD_ERROR_COUNT_CONF2** or **FWD_ERROR_COUNT_CONF1**, respectively.

For example, if counter 1 should measure how often the **BNR_BUBBLE** error flag of **RAM_R_FW_ERR_FLAGS** is raised during one hour, set **FWD_ERROR_COUNT_CONF1** to 0x00004000 (only bit 14 = position of **BNR_BUBBLE** set). Then **RAM_ERROR_COUNT_21** will count all bubble error flag events in bytes B1/B0 during the first hour, and will be increased if in a subsequent hour the number of error events is higher. This way, the peak value of average error events can be measured. Counter 1 and 2 can be configured for any desired combination of error flags.

If the average error counters should be reset to zero by the external controller, it is necessary also to reset FWD cells **FWD_ERROR_COUNT_21** and **FWD_ERROR_COUNT_43**, since in the RAM part of these NVRAM cells contains the current error count.

5.4 Error interrupt

A user-defined combination of errors can be configured to issue a synchronous firmware interrupt in a similar way as the error counter configurations above: Set the desired bits according to error flag positions in **RAM_R_FW_ERR_FLAGS** (see section 6.1.2) and switch on the synchronous firmware interrupt in **CR_IH** and FWD cell 112, respectively. This should be used to issue an irregular interrupt at a user-defined error (combination). A typical application of such an interrupt would be if TDC-GP30 communicates its results only rarely to the system's microcontroller, but special events should be recognized immediately. For example, writing 0x00008000 in cell 83

FWD_ERR_INTERRUPT will issue an irregular interrupt on the interface as soon as no-water is detected.

5.5 Error signals through pulse interface

acam's firmware can be configured to signal error conditions over the pulse interface, too (see section 2.6). In case of error, the pulse output goes permanently high, and the direction output toggles with the TOF measurement frequency. This creates no additional pulse count and can be easily identified by some external readout device or master controller. With bit 22 of the firmware configuration register **FWD_FW_CONFIG** it can in addition be chosen if a no-water situation should also be signaled as error or not.

This way of signaling errors is not suitable if flow in both directions should be measured and signaled over two pins for different directions (this is the alternative configuration for the pulse interface). In this case error signaling through the pulse interface should be switched off. In case no reverse flow is counted, error signaling can be used as described even using only the positive pulse pin.

6 acam firmware variables in RAM

The TDC-GP30 has a total of 176 32-bit RAM memory cells (see Manual Volume 1: General Data and Frontend Description, chapter 7). This volatile memory block is used by firmware as well as by the frontend data buffer. In addition, any memory cell of FWD (address 0x100 to 0x17F) can be used as volatile RAM cell, too – keeping in mind that these cells are overwritten at recalls by their corresponding flash memory content. The following section lists the complete RAM usage of the acam firmware as well as the bit definitions in **RAM_R_FW_STATUS** and **RAM_R_FW_ERR_FLAGS**. The subsequent section summarizes the cells of FWD which are used as RAM by the acam firmware.

6.1 Results in memory cells

The following list is a complete overview of RAM memory cell usage by the acam firmware. Cells marked in green contain final results or other functions which may be of interest. All other cells are given mainly as lookup reference. For the format definition, see the notational conventions at the first pages of this document.

All variable names and address definitions can also be found in file GP30Y_A1.D2.11.03.h, contained in the evaluation package.

Address	Variable name	Description	Format
0x000 - 0x008	<i>Variables with final results from flow calculation</i>	These variables contain results from flow calculations which should be read by the external controller	
0x000	RAM_R_FLOW_VOLUME_INT	Integer part of total volume of water flow in cubic meters	fd0
0x001	RAM_R_FLOW_VOLUME_FRACTION	Fractional part of total volume of water flow in cubic meters	fd32
0x002	RAM_R_FLOW_LPH	Presently calculated flow volume (l/h), unfiltered	fd16
0x003	RAM_FILTERED_FLOW_LPH	Filtered flow volume (l/h)	fd16
0x004	RAM_R_THETA	Temperature (°C) calculated from SUMTOF	fd16
0x005	RAM_SOUND_VEL	Velocity of sound (m/s)	fd8
0x006	RAM_FLOW_SPEED	Calculated speed of flow (m/s)	fd16
0x007	RAM_R_TOF_DIFF	Current DIFTOF in raw TDC units	fd16
0x008	RAM_R_TOF_SUM	Current SUMTOF in raw TDC units	fd16
0x009 – 0x01B	<i>Permanent variables and parameters for flow calculations</i>	These variables are used to transfer former results and settings to the current flow calculation. Do not overwrite.	-
0x01C – 0x01E	<i>Variables for amplitude calculations</i>	These variables are used in amplitude calculation and error check	-
0x01C	RAM_R_AM_MIN_RAW	Minimal acceptable signal amplitude, calculated from latest calibration values for	-

Address	Variable name	Description	Format
		direct comparison to raw amplitude measurement value.	
0x01D	RAM_R_AMC_GRADIENT	Latest amplitude calibration gradient value, see Manual Vol. 3: User Manual, section 2.2.	-
0x01E	RAM_R_AMC_OFFSET	Latest amplitude calibration offset value, see Manual Vol. 3: User Manual, section 2.2.	-
0x01F	RAM_R_V1F_COEFF_ADR / RAM_R_V1F_SHIFT	Temporary variable for PWL coefficients table address or rapid shifting	-
0x020 - 0x024	<i>Variables with final results from sensor temperature measurement</i>	These variables contain results from sensor temperature measurement which should be read by the external controller	
0x020	RAM_R_PTC_TEMPERATURE	Cold sensor temperature (°C)	fd16
0x021	RAM_R_PTH_TEMPERATURE	Hot sensor temperature (°C)	fd16
0x022	RAM_PTC_RES	Cold sensor resistance (Ω)	fd16
0x023	RAM_R_PTH_RES	Hot sensor resistance (Ω)	fd16
0x024	RAM_R_PT_INT_TEMPERATURE	Internal sensor temperature (°C)	fd16
0x025 - 0x028	<i>Status information</i>	These four variables contain the internal status information	
0x025	RAM_R_FW_STATUS	Firmware status bits, see 6.1.1	bits
0x026	RAM_FLOW_COUNTER	Internal counter for flow values after last average update	int
0x027	RAM_R_FW_ERR_FLAGS	Firmware and hardware error flags, see 6.1.2 Note that most of these flags are not permanent and erased with each new measurement (flow or sensor temperature).	bits
0x028	RAM_R_FHL_ERR_CTR	Internal error counter, increased with every error which prevents flow calculation and reset to zero at every correct measurement. At values below 4, corrupted measurements are replaced by valid preceding results. At values of 4 and above, measurement failure is signaled, flow is set to zero and bubble detection is checked.	int
0x029 – 0x033	<i>Temporary variables for flow and sensor temperature calculations</i>	These variables are used temporarily. They can be overwritten and reused outside flow or temperature calculations. Note that they will of course be modified by the firmware.	
0x034	RAM_V34_AM_HW_OFFSET	Temporary variable, only permanent in case of low amplitude error (used for hardware failure check).	-
0x035 – 0x04C	<i>Unused</i>	These memory cells are completely unused and are freely available for custom usage. Note that addresses up to 0x03F permit faster access, so they should be used for variables which are often accessed.	

Address	Variable name	Description	Format
0x04D – 0x04E	<i>Optional: Average error counters Unused when counters are disabled</i>	These cells store the maximum count of errors within an hour until the last reset. They thus measure the peak hourly error count.	
0x04D	RAM_ERROR_COUNT_21	(optional) Peak hourly error count of counters 2 (B3/B2) and 1 (B1/B0), as configured in 0x103 and 0x104	int
0x04E	RAM_ERROR_COUNT_43	(optional) Peak hourly error count of counters 4 (B3/B2; all errors) and 3 (B1/B0; hardware defined errors)	int
0x04F – 0x051	<i>Optional: Flow values for negative flow exceeding the limit given in FWD cell 93 Unused if no limit is set</i>	These variables store the results for negative flow which is not considered in flow volume.	
0x04F	RAM_NEG_FLOW	(optional) Presently calculated negative flow volume (l/h), unfiltered	fd16
0x050	RAM_NEG_FLOW_VOLUME_INT	(optional) Integer part of total volume of negative water flow in cubic meters	fd0
0x051	RAM_NEG_FLOW_VOLUME_FRACTION	(optional) Fractional part of total volume of negative water flow in cubic meters	fd32
0x052 – 0x055	<i>Optional: redundant storage for flow volume Unused if no secure flow volume storage configured</i>	These four cells contain, if configured, 2x2 safety copies of RAM_R_FLOW_VOLUME_INT and RAM_R_FLOW_VOLUME_FRACTION . Volume values are considered valid when at least two of these three values are identical, else the error flag BNR_VOL_ERR is raised.	
0x052	RAM_C1_FLOW_VOLUME_INT	(optional) safety copy 1 of flow volume, integer part	fd0
0x053	RAM_C2_FLOW_VOLUME_INT	(optional) safety copy 2 of flow volume, integer part	fd0
0x054	RAM_C1_FLOW_VOLUME_FRACTION	(optional) safety copy 1 of flow volume, fractional part	fd32
0x055	RAM_C2_FLOW_VOLUME_FRACTION	(optional) safety copy 2 of flow volume, fractional part	fd32
0x056	RAM_FEP_STF	Copy of recent SRR_FEP_STF , to keep front end status after clear of SRR_FEP_STF	bits
0x057 – 0x05C	<i>Error- and cycle counters</i>	These error counters count the number of measurements with error, they are cleared at every valid measurement	
0x057	RAM_LOW_AM_ERR_CTR	Counter for consecutive low AM cases; this counter is compared to FWD_LONG_TERM_ERROR to do a no-water or hardware check	int
0x058	RAM_TS_ERR_CTR	Counter for consecutive Task sequencer timeout errors	int

Address	Variable name	Description	Format
0x059	RAM_R_TM_ERR_CTR	Counter for consecutive TM errors	int
0x05A	RAM_R_USM_ERR_CTR	Counter for consecutive USM errors	int
0x05B	RAM_R_AM_ERR_CTR	Counter for consecutive AM errors	int
0x05C	RAM_CYCLE_COUNTER	This counter controls the slow FHL changes Bits 4..0: cycle counter 1-16; bit 5 indicates if PWR or FHL regulation is due at value 16	int
0x05D – 0x05F	<i>Clock related variables</i>	These variables are related to HSC clock calibration	
0x05D	RAM_R_HSC_SCALE_FACT	HS Clock scaling factor , deviation from nominal frequency	fd24
0x05E	RAM_R_TDC_PERIOD	(real HSC period in s)/(distance w. flow in m) = ((250*10 ⁻⁹ s or 125*10 ⁻⁹ s) * RAM_R_HSC_SCALE_FACT) / S _f) * 2 ⁴³⁹	fd39
0x05F	RAM_R_TDC_CLK	real HSC frequency: (4MHz: 0x003D0900 or 8MHz: 0x007A1200) / RAM_R_HSC_SCALE_FACT	fd0
0x06A – 0x07F	<i>Permanent variables for filter functions</i>	These variables are used for filtering over some measurement cycles. Don't modify !	
0x060 - 0x06F	RAM_R_ROLAVG_1 ... RAM_R_ROLAVG_16	Rolling average filter for flow, starting at 0x060: RAM_R_ROLAVG_1	
0x070 - 0x077	RAM_ROLAVG_DIFTOF_1 ... RAM_ROLAVG_DIFTOF_8	Rolling average filter for DIFTOF, starting at 0x070: RAM_ROLAVG_DIFTOF_1	
0x078 - 0x07F	RAM_ROLAVG_SUMTOF_1 ... RAM_ROLAVG_SUMTOF_8	Rolling average filter for SUMTOF, starting at 0x078: RAM_ROLAVG_SUMTOF_1	
0x080 – 0x09B	<i>Frontend data buffer</i>	These variables contain all measurement results of the measurement frontend. The cells may be overwritten and reused after evaluation. acam firmware usually doesn't modify these cells. This table just lists the variable names for results for TOF and TM measurements, for details please see Manual Vol. 1, 3.2 and 3.3	
0x080	FDB_US_TOF_ADD_ALL_U / FDB_TM_PP_M1	TOF Sum Up of all the configured hits / t _{pp} : Offset delay comp. value of Meas. 1	
0x081	FDB_US_PW_U / FDB_TM_PTR_RAB_M1	US Pulse Width Ratio Up / t _{RAB} : Reference Impedance value of Meas. 1	
0x082	FDB_US_AM_U / FDB_TM_PTC_CAB_M1	US Amplitude Value Up / t _{CAB} : PT Cold Impedance value of Meas. 1	
0x083	FDB_US_AMC_VH / FDB_TM_PTH_HAB_M1	US Amplitude Calibrate Value High / t _{HAB} : PT Hot Impedance value of Meas. 1	
0x084	FDB_US_TOF_ADD_ALL_D / FDB_TM_PTR_RA_M1	TOF Sum Down of all the configured hits / t _{RA} : 1 st Offset resistance value of Meas. 1	
0x085	FDB_US_PW_D / FDB_TM_PP_M2	US Pulse Width Ratio Down / t _{pp} : Offset delay comp. value of Meas. 2	

Address	Variable name	Description	Format
0x086	FDB_US_AM_U / FDB_TM_PTR_RAB_M2	US Amplitude Value Down / t_{RAB} : Reference Impedance value of Meas. 2	
0x087	FDB_US_AMC_VL / FDB_TM_PTC_CAB_M2	US Amplitude Calibrate Value Low / t_{CAB} : PT Cold Impedance value of Meas. 2	
0x088	FDB_US_TOF_0_U / FDB_TM_PTH_HAB_M2	Ultrasonic TOF Up Value 0 / t_{HAB} : PT Hot Impedance value of Meas. 2	
0x089	FDB_US_TOF_1_U / FDB_TM_PTR_RA_M2	Ultrasonic TOF Up Value 1 / t_{RA} : 1 st Offset resistance value of Meas. 2	
0x08A	FDB_US_TOF_2_U / FDB_TM_PTR_4W_RB_M1	Ultrasonic TOF Up Value 2 / t_{RB} : Reference 2 nd Offset res. val. of Meas. 1	
0x08B	FDB_US_TOF_3_U / FDB_TM_PTC_4W_CA_M1	Ultrasonic TOF Up Value 3 / t_{CA} : PT Cold 1 st Offset res. value of Meas. 1	
0x08C	FDB_US_TOF_4_U / FDB_TM_PTC_4W_CB_M1	Ultrasonic TOF Up Value 4 / t_{CB} : PT Cold 2 nd Offset res. value of Meas. 1	
0x08D	FDB_US_TOF_5_U / FDB_TM_PTC_4W_AC_M1	Ultrasonic TOF Up Value 5 / t_{AC} : PT Cold 3 rd Offset res. value of Meas. 1	
0x08E	FDB_US_TOF_6_U / FDB_TM_PTC_4W_BC_M1	Ultrasonic TOF Up Value 6 / t_{BC} : PT Cold 4 th Offset res. value of Meas. 1	
0x08F	FDB_US_TOF_7_U / FDB_TM_PTH_4W_HA_M1	Ultrasonic TOF Up Value 7 / t_{HA} : PT Hot 1 st Offset res. value of Meas. 1	
0x090	FDB_US_TOF_0_D / FDB_TM_PTH_4W_HB_M1	Ultrasonic TOF Down Value 0 / t_{HB} : PT Hot 2 nd Offset res. value of Meas. 1	
0x091	FDB_US_TOF_1_D / FDB_TM_PTH_4W_AH_M1	Ultrasonic TOF Down Value 1 / t_{AH} : PT Hot 3 rd Offset res. value of Meas. 1	
0x092	FDB_US_TOF_2_D / FDB_TM_PTH_4W_BH_M1	Ultrasonic TOF Down Value 2 / t_{BH} : PT Hot 4 th Offset res. value of Meas. 1	
0x093	FDB_US_TOF_3_D / FDB_TM_PTR_4W_RB_M2	Ultrasonic TOF Down Value 3 / t_{RB} : Reference 2 nd Offset res. val. of Meas. 2	
0x094	FDB_US_TOF_4_D / FDB_TM_PTC_4W_CA_M2	Ultrasonic TOF Down Value 4 / t_{CA} : PT Cold 1 st Offset res. value of Meas. 2	
0x095	FDB_US_TOF_5_D / FDB_TM_PTC_4W_CB_M2	Ultrasonic TOF Down Value 5 / t_{CB} : PT Cold 2 nd Offset res. value of Meas. 2	
0x096	FDB_US_TOF_6_D / FDB_TM_PTC_4W_AC_M2	Ultrasonic TOF Down Value 6 / t_{AC} : PT Cold 3 rd Offset res. value of Meas. 2	
0x097	FDB_US_TOF_7_D / FDB_TM_PTC_4W_BC_M2	Ultrasonic TOF Down Value 7 / t_{BC} : PT Cold 4 th Offset res. value of Meas. 2	
0x098	FDB_TM_PTH_4W_HA_M2	t_{HA} : PT Hot 1 st Offset res. value of Meas. 2	
0x099	FDB_TM_PTH_4W_HB_M2	t_{HB} : PT Hot 2 nd Offset res. value of Meas. 2	
0x09A	FDB_TM_PTH_4W_AH_M2	t_{AH} : PT Hot 3 rd Offset res. value of Meas. 2	
0x09B	FDB_TM_PTH_4W_BH_M2	t_{BH} : PT Hot 4 th Offset res. value of Meas. 2	
0x09C – 0x0AF	<i>Temporary parameters</i>	These variables are used by firmware or ROM for temporary results. They can be used temporarily for custom codes, too.	

Address	Variable name	Description	Format
0x0A8 - 0x0AB	<i>Checksums for non-volatile memories</i>	Note that these checksums are calculated by the chip and only available after the calculation process has run (part of the bootloading process). Values here may be overwritten by firmware later	
0x0A8	RAM_R_VA8_FWD1_CS	Checksum Firmware Data 1	fd0
0x0A9	RAM_R_VA9_FWD2_CS	Checksum Firmware Data 2	fd0
0x0AA	RAM_R_VAA_FWU_CS	Checksum Firmware Code User	fd0
0x0AB	RAM_R_VAB_FWA_CS	Checksum Firmware Code acam	fd0

The most important results are directly displayed in the PC software (see below): Select menu “Firmware /CPU values” and click “Read calculated values”; of course measurement and firmware must be running and providing the desired data. The results can also be displayed in a graph over time by clicking “Open CPU Graph”. The “CPU values” sheet also has three configurable cells where the RAM address can be filled in, such that its contents gets multiplied with an arbitrary factor, and the permanently updated value gets displayed. For example, filling in (0x00)4 as address and a multiplication factor of $1/2^{16}=0.000015259$ displays the same temperature as in the dedicated temperature field above. This function can be used to permanently display any result of the firmware.

Of course, in a real flow meter the desired results have to be read out and displayed or evaluated e.g. by an external controller. If the acam firmware runs in stand-alone operation, this is simply done over the SPI or the UART interface by reading the desired RAM cell contents. Please refer to Manual Volume 1: General Data and Frontend Description for details on interface communications.

The screenshot shows the 'CPU values' window with the following sections:

- FW Error Flags:** A list of error flags with green circular icons, including HS_CALIB_FAIL, AMP_DIFF_OUT_OF_RANGE, AMP_VAL_TOO_LOW, PW_DIFF_NOT_OK, SUMTOF_DEV, FHL_NOT_OK, MEAS_NOT_OK, HARDWARE_FAILURE, FLOW_BT_4MAX, FLOW_LT_NEGLIM, VOL_ERR, ROM_ERR, ROM_TEMP_ERR, VEL_ERROR, BUBBLE, and NO_WATER.
- Control Buttons:** 'Stop Measurement', 'Open CPU Graph', and 'Hide Error Flags'. Below 'Stop Measurement' are checkboxes for 'Read calculated values' (checked) and 'Write Values to file'.
- CPU Temperature Results from Flow:**

#	Name	Results	Unit
1	Temperature	23.03	degC
2	Sound Velocity	1492.33	m/s
- CPU Results:**

#	Name	Results	Unit
1	Flow	164.199	Liter / Hour
2	Flow averaged	162.162	Liter / Hour
3	Volume Flow	0.026763	m^3
4	Volume Flow	26.763	Liter
5	Flow Speed	0.30	m/s
- CPU Results with ext. Temperature Sensors:**

#	Name	Results	Unit
1	Temperature Cold	23.49	degC
2	Temperature Hot	24.56	degC
3	Resistance Cold Sensor	1091.49	Ohm
4	Resistance Hot Sensor	1095.64	Ohm
5	Temperature Internal	23.83	degC
- CPU TOF Values:**

#	Name	Results	Unit
1	TOF sum	161512.25	ns
2	TOF diff	16.7788	ns
- CPU Results at self-defined RAM Addresses:**

Address	Mult. Factor	Calculated Result	Calculated Result
Address 1: 4	1.52588E-5	23.0324	17
Address 2: DA	0.88	50.1600	32
Address 3: 4D	1	0.0000	0
- SRR ERR Flags:** A list of error flags with green circular icons, including TDC Timeout, TOF Timeout, Amplitude Measurement Timeout, Temperature Measurement Open Circuit, Temperature Measurement Short Circuit, Zero Cross Calibration, Low Battery Detect, Ultrasonic Sequence Timeout, Temperature Sequence Timeout, Task Sequencer Timeout, FWD1Checksum, FWD2Checksum, FWU Checksum, and FWA Checksum.

6.1.1 RAM_R_FW_STATUS (Firmware status register) 0x025

All bits of this register are controlled by the firmware. They are listed here for information and should normally not be changed from outside.

Bit	Description	Format	Default
31	BNR_TEST_MODE_FHL: Indicate active FHL regulation 0: Normal operation 1: FHL regulation is active to control first hit level or recover from error.	BIT	b0
30	BNR_TOF_DIFF_NEGATIVE 0: Current DIFTOF is positive after zero-flow correction 1: Current DIFTOF is negative after zero-flow correction	BIT	b0
29	BNR_TOF_RATE_REDUCED 0: TOF rate as originally configured in <i>FWD_USM_PRC</i> /B3 1: TOF rate changed for zero flow as given in <i>FWD_TOF_RATE_FACTOR</i>	BIT	b0
28	<i>Unused</i>	BIT	b0
27	<i>Unused, but regularly set to 1 after init or recall</i>	BIT	b0
26	BNR_EN_FHL_MONITORING 0: Disable FHL regulation (for temporary tests) 1: Enable FHL regulation (enforced at init or recall)	BIT	b1
25	BNR_TEST_MODE_EP: indicate no-water test mode 0: Normal operation 1: Test mode to check for no-water or hardware error	BIT	b0
24	BNR_TEST_MODE_ADJ: Indicate active PWR adjustment 0: Normal operation 1: FHL adjustment for nominal PWR is active	BIT	b0
23	BNR_PI_ERR_1ST_DIR: Temporary storage of pulse interface direction when starting error signal	BIT	b0
22	BNR_TEMP_REFRSH 0: Normal operation 1: Refresh of registers after recall needed	BIT	b0
21	BNR_I2C_ABORT (set by ROM routine) 0: No problems with I2C transactions 1: Some I2C transaction was aborted with NOT ACKNOWLEDGE	BIT	b0
20	<i>Unused, but regularly reset at init or recall</i>	BIT	b0
19	BNR_THETA_OUT_OF_RANGE 0: Normal operation 1: Temperature for calibration is outside calibration table	BIT	b0
18	BNR_NON_ZERO_FLOW 0: Zero flow possible 1: No zero flow - instantaneous flow is above 8*zero flow limit, or zero flow state is disabled due to regulations	BIT	b0
17	BNR_PI_UPD_REQ 0: No pulse interface (PI) update 1: PI update is requested – the PI gets new flow volume added	BIT	b0

Bit	Description	Format	Default
16	BNR_HSC_SCALE_EN: Set HSC cal. according to configuration 0: Disable high speed clock calibration in firmware 1: Enable high speed clock calibration in firmware	BIT	b1
15	BNR_TOFSUM_DIV_MODE 0: Division by shifting 1: Normal division	BIT	b1
14	BNR_AVG_TOF_FOR_FLOW 0: Accumulating flow values for long term average 1: Full number of flow values for long term average reached	BIT	b0
13	BNR_UART_IF_ENABLED: Sends flow data over UART if enabled 0: UART interface disabled 1: UART interface enabled – flow data is sent out after calculation	BIT	b0
12	<i>unused</i>	BIT	b0
11	BNR_FILTER_INIT_DONE 0: TOF filters not initialized 1: TOF filters initialized	BIT	b0
10	BNR_MEAS_FAILURE_ALERT 0: No measurement failure signaled 1: Measurement failure: More than 4 measurement errors in sequence	BIT	b0
9	BNR_AM_CLB_REQ: Copy of current state in SRR_FEP_STF 0: No request 1: Amplitude calibration requested	BIT	b0
8	BNR_AM_MON_REQ: Copy of current state in SRR_FEP_STF 0: No request 1: Amplitude calculation requested	BIT	b0
7	BNR_TOF_EDGE: Copy of current state in SRR_FEP_STF 0: Positive TOF edge 1: Negative TOF edge	BIT	b0
6	BNR_FLOW_CALC_REQ: Copy of current state in SRR_FEP_STF 0: No request 1: Flow calculation requested	BIT	b0
5	BNR_US_D_UPD: Copy of current state in SRR_FEP_STF 0: No request 1: TOF_DOWN measurements available	BIT	b0
4	BNR_US_U_UPD: Copy of current state in SRR_FEP_STF 0: No request 1: TOF_UP measurements available	BIT	b0
3	BNR_FLOW_FILT_INIT_DONE 0: Flow filter not initialized 1: Flow filter initialized	BIT	b0
2	BNR_FWI_DONE 0: No firmware initialisation 1: Firmware initialization done	BIT	b1
1	BNR_TM_CALC_REQ: Copy of current state in SRR_FEP_STF 0: No request 1: Sensor temperature calculation requested	BIT	b0

Bit	Description	Format	Default
0	BNR_HSC_CLB_REQ : Copy of current state in SRR_FEP_STF 0: No request 1: High speed clock calibration requested	BIT	b0

6.1.2 **RAM_R_FW_ERR_FLAGS** (Firmware error flag register) 0x027

Most bits of this register are temporary, unless otherwise noted: They indicate an error at the time it appears and clear when it vanishes. Bits 31:16 are copies of the hardware error flags in **SRR_ERR_FLAG** (see section 7.5.2 in Manual Volume 1: General Data and Frontend Description). While the original bits 15:0 in **SRR_ERR_FLAG** are cleared after firmware evaluation, to be able to recognize new errors, their copies in bits 31:16 of **RAM_R_FW_ERR_FLAGS** are kept until the next measurement cycle and can thus be read out.

Bit	Description	Format	Default
31	BNR_CS_FWA_ERR : Copy of bit 15 of SRR_ERR_FLAG before clear Error Flag FWA Checksum: Flag is set when the checksum of FW acam code calculated by GP30 is not the same as stored in FWD_R_FWA_CS .	BIT	b0
30	BNR_CS_FWU_ERR : Copy of bit 14 of SRR_ERR_FLAG before clear Error Flag FWU Checksum: Flag is set when the checksum of FW user code calculated by GP30 is not the same as stored in FWD_R_FWU_CS .	BIT	B0
29	BNR_CS_FWD2_ERR : Copy of bit 13 of SRR_ERR_FLAG before clear Error Flag FWD2 Checksum: Flag is set when the checksum of FWD2 calculated by GP30 is not the same as stored in FWD_R_FWD2_CS .	BIT	b0
28	BNR_CS_FWD1_ERR : Copy of bit 12 of SRR_ERR_FLAG before clear Error Flag FWD1 Checksum: Flag is set when the checksum of FWD1 calculated by GP30 is not the same as stored in FWD_R_FWD1_CS .	BIT	b0
27	Copy of bit 11 of SRR_ERR_FLAG before clear; <i>Not used</i>	BIT	b0
26	BNR_E2C_ACK_ERR : Copy of bit 10 of SRR_ERR_FLAG before clear Error Flag EEPROM Acknowledge: Flag is set when EEPROM communication signaled an acknowledge error, communication failed.	BIT	b0
25	BNR_TSQ_TMO : Copy of bit 9 of SRR_ERR_FLAG before clear Error Flag Task Sequencer Timeout: Flag is set when the task sequencer was not idle when the subsequent measurement cycle started. Some processes may not have finished correctly or need more time for execution.	BIT	b0
24	BNR_TM_SQC_TMO : Copy of bit 8 of SRR_ERR_FLAG before clear Error Flag Temperature Sequence Timeout: Flag is set when the temperature measurement sequence is longer than the configured pause time. Note that without pause the flag is always set. See CR_TM .	BIT	b0
23	BNR_USM_SQC_TMO : Copy of bit 7 of SRR_ERR_FLAG before clear Error Flag Ultrasonic Sequence Timeout: Flag is set when the ultrasonic measurement sequence is longer than the configured pause time. Note that without pause the flag is always set. See CR_USM_PRC .	BIT	b0

Bit	Description	Format	Default
22	BNR_LBD_ERR: Copy of bit 6 of SRR_ERR_FLAG before clear Error Flag Low Battery Detect: Flag is set when the supply voltage V_{cc} drops below the limit (see CR_CPM). acam firmware signals this, in addition, as LOW = 0 V on GPIO6 (normally HIGH = V_{cc}).	BIT	b0
21	BNR_ZCC_ERR: Copy of bit 5 of SRR_ERR_FLAG before clear Error Flag Zero Cross Calibration: Flag is set when zero cross calibration did not converge and may have failed. See GP30 manual vol. 1 sec. 3.4.3	BIT	b0
20	BNR_TM_SC_ERR_FW: Copy of bit 4 of SRR_ERR_FLAG before clear Error Flag Temperature Measurement Short Circuit: Flag is set when the threshold voltage for temperature measurement is reached in less than 1 μ s, which indicates too low resistance / short circuit at one sensor.	BIT	b0
19	BNR_TM_OC_ERR_FW: Copy of bit 3 of SRR_ERR_FLAG before clear Error Flag Temperature measurement open circuit: Flag is set when the threshold voltage for temperature measurement is not reached within the configured discharge time, which indicates too high resistance / open circuit at one sensor.	BIT	b0
18	BNR_AM_TMO_FW: Copy of bit 2 of SRR_ERR_FLAG before clear Error flag amplitude measurement timeout: Flag is set when TDC timeout was reached during amplitude measurement, amplitude value is invalid.	BIT	b0
17	BNR_TOF_TMO_FW: Copy of bit 1 of SRR_ERR_FLAG before clear Error flag TOF timeout: Flag is set when configured TOF timeout was reached while waiting for a signal, see CR_USM_PRC .	BIT	b0
16	BNR_TDC_TMO_FW: Copy of bit 0 of SRR_ERR_FLAG before clear Error flag TDC timeout: Flag is set when TDC timeout was reached while waiting for a signal (after 4.096 ms)	BIT	b0
15	BNR_NO_WATER (see section 5.1 and bit 2 below) 0: No no-water situation detected 1: No-water detected after long-term AMP_VAL_TOO_LOW	BIT	b0
14	BNR_BUBBLE 0: Normal operation 1: Bubble detected: bits 1,3 or 4 are set after at least 4 meas. errors	BIT	b0
13	BNR_VEL_ERROR 0: Calculated velocity of sound is within limits, see section 5.1 1: Calculated velocity of sound is outside limits; former value is used	BIT	b0
12	BNR_ROM_TEMP_ERR Internal indicator for unpowered ROM call supporting BNR_ROM_ERR	BIT	b0
11	BNR_ROM_ERR 0: Normal operation 1: ROM was called unpowered; this bit is not cleared automatically, it needs to be cleared remotely	BIT	b0
10	BNR_VOL_ERR 0: No error on stored flow volume (or no volume control), see 3.1.1 1: Stored flow volume had an unrecoverable error; this bit is not cleared automatically, it needs to be cleared remotely	BIT	b0
9	BNR_FLOW_LT_NEGLIM 0: Flow is not below negative limit, see section 3.3 1: Flow is below negative limit and stored separately, see section 3.3	BIT	b0

Bit	Description	Format	Default
8	BNR_FLOW_BT_2MAX 0: Current flow is not exceeding 2*maximum flow, see section 3.4 1: Current flow exceeds 2*maximum flow and is forced to zero	BIT	b0
7	BNR_HARDWARE_FAILURE (see section 3.4 and bit 2 below) 0: No hardware failure detected 1: Hardware failure detected after long-term AMP_VAL_TOO_LOW	BIT	b0
6	BNR_MEAS_NOT_OK 0: Measurement is o.k. (after short-term correction, if necessary) 1: Measurement is considered wrong (> 4 errors sequentially); Flow is forced to zero until the next error-free measurement occurs	BIT	b0
5	BNR_FHL_NOT_OK 0: Current FHL is consistent with configuration, see section 4.2 1: Current FHL is inconsistent with config., FHL regulation is active	BIT	b0
4	BNR_SUMTOF_DEV : Check deviation of new SUMTOF from average 0: Deviation from former SUMTOF average is below limits, see 3.4 1: Deviation from former SUMTOF average exceeds limits	BIT	b0
3	BNR_PW_DIFF_NOT_OK : Check up and down PWR difference 0: PWR difference of current measurement is within limits, see sect. 3.4 1: PWR difference of current measurement exceeds limits	BIT	b0
2	BNR_AMP_VAL_TOO_LOW 0: Measured amplitude up or down is above minimum 1: Measured amplitude up or down is below minimum, see sect. 3.4	BIT	b0
1	BNR_AMP_DIFF_TOO_HIGH : Check up / down amplitude difference 0: Amplitude difference of current meas. is within limits, see sect. 3.4 1: Amplitude difference of current measurement exceeds limits	BIT	b0
0	BNR_HS_CALIB_FAIL 0: HS Clock deviation within limit, see section 3.4 1: HS Clock Calibration not done because the deviation is too large	BIT	b0

6.2 acam firmware variables in RAM-part of FWD

The following cells of FWD are used as RAM by the acam firmware or modified after recall. Note that FWD is an NVRAM, which consists of a volatile RAM part and a FLASH part for permanent storage. The RAM can be used as usual, but at a recall the data from the FLASH part is transferred to RAM and overwrites its former content. For a complete list of FWD cells, see chapter 7.

FWD cell #	FWD cell address	Variable name	Description and (if applicable) default value	Format
0	0x100	<i>FWD_R_FLOW_VOLUME_INT</i>	(optional): Integer part of negative flow volume in cubic meters - internally used if a negative flow limit is defined in cell # 93.	fd0
1	0x101	<i>FWD_R_FLOW_VOLUME_FRACTION</i>	(optional): Fractional part of negative flow volume in cubic meters - internally used if a negative flow limit is defined in cell # 93.	fd32
5	0x105	<i>FWD_ERROR_COUNT_21</i>	(optional) Temporary storage of error counts 2 (B3,B2) and 1 (B1,B0) (see section 5.3); Set to 0x00000000 if error counters are used	int
6	0x106	<i>FWD_ERROR_COUNT_43</i>	(optional) Temporary storage of error counts 4 - each error (B3,B2) and 3 - hardware errors (B1,B0) (see section 5.3); Set to 0x00000000 if error counters are used	int
107	0x16C	<i>FWD_R_CD</i>	Watchdog disable code Without acam firmware, 0x48DBA399 disables the watchdog while any other code enables it. With acam firmware, the watchdog can't be disabled, but 0x00000000 disables configuration restore after recall. Proposed stored value 0x95659C6A	bits

7 acam firmware parameters in firmware data (FWD)

The TDC-GP30 has a total of 128 32-bit register words in firmware data (FWD, see Manual Volume 1: General Data and Frontend Description, chapter 7).

The following list is a complete overview of FWD memory cell usage by the acam firmware, mainly as lookup reference. For the format definition, see the notational conventions at the first pages of this document. More functional descriptions of the particular cells are given in chapter 3.

All variable names and address definitions can also be found in file GP30Y_A1.D2.11.03.h, contained in the evaluation package.

Cell #	Address	Variable name	Description and (if applicable) default value	Format
0	0x100	FWD_R_FLOW_VOLUME_INT	(optional): Integer part of negative flow volume in cubic meters - internally used if a negative flow limit is defined in cell # 93.	fd0
1	0x101	FWD_R_FLOW_VOLUME_FRACTION	(optional): Fractional part of negative flow volume in cubic meters - internally used if a negative flow limit is defined in cell # 93.	fd32
2	0x102	<i>Not used</i>	<i>Not used</i>	
3	0x103	FWD_ERROR_COUNT_CONF1	(optional) Define error flag positions to be counted in error counter 1 (see section 5.3)	bits
4	0x104	FWD_ERROR_COUNT_CONF2	(optional) Define error flag positions to be counted in error counter 2 (see section 5.3)	bits
5	0x105	FWD_ERROR_COUNT_21	(optional) Temporary storage of error counts 2 (B3,B2) and 1 (B1,B0) (see section 5.3); Set to 0x00000000 if error counters are used	int
6	0x106	FWD_ERROR_COUNT_43	(optional) Temporary storage of error counts 4 - each error (B3,B2) and 3 - hardware errors (B1,B0) (see section 5.3); Set to 0x00000000 if error counters are used	int
7 - 53	0x107 - 0x135	<i>Unused or used for nonlinear calibration coefficients</i>	Typical usage see below (configurable)	
16 - 41	0x110 - 0x129	(PWL coefficient table)	(optional) Typical position of PWL calibration coefficients table, generated by cal. engine	-
42 - 53	0x12A - 0x135	(acam coefficient table)	(optional) Fixed position for acam calibration coefficients table, generated by cal. engine	-
54 - 73	0x12A - 0x135	Linear coefficients table	Linear coefficients table, all generated by cal. engine ; values in cells # 58,59 and 62 – 73 have to be adapted to individual spool pieces by 2-point calibration	-
54	0x136	FWD_R_TEMP_TC1	1 st temperature for linear calibration in °C	fd16
55	0x137	FWD_R_TEMP_TC2	2 nd temperature for linear calibration in °C	fd16
56	0x138	FWD_R_TEMP_TC3	3 rd temperature for linear calibration in °C	fd16
57	0x139	FWD_R_TEMP_TC4	4 th temperature for linear calibration in °C	fd16

Cell #	Address	Variable name	Description and (if applicable) default value	Format
58	0x13A	FWD_R_TOF_OFFSET	Offset time for SUMTOF in raw TDC units	fd0
59	0x13B	FWD_TOF_DIFF_CAL	DIFTOF at high flow calibration point in raw TDC units	fd0
60	0x13C	FWD_DIST_WITH_FLOW	Ultrasonic sound path length along flow in m	fd16
61	0x13D	FWD_DIST_NO_FLOW	Ultrasonic sound path length w/o flow in m	fd16
62	0x13E	FWD_R_ZERO_OFFSET_TC2	Zero flow DIFTOF at TC2	fd16
63	0x13F	FWD_R_ZERO_OFFSET_TC3	Zero flow DIFTOF at TC3	fd16
64	0x140	FWD_R_ZERO_OFFSET_TC4	Zero flow DIFTOF at TC4	fd16
65	0x141	FWD_R_O_SLOPE_TC12	Zero flow slope between TC1 and TC2	fd16
66	0x142	FWD_R_O_SLOPE_TC23	Zero flow slope between TC2 and TC3	fd16
67	0x143	FWD_R_O_SLOPE_TC34	Zero flow slope between TC3 and TC4	fd16
68	0x144	FWD_R_F_SLOPE_TC12	Proport. factor slope between TC1 and TC2	fd16
69	0x145	FWD_R_F_SLOPE_TC23	Proport. factor slope between TC2 and TC3	fd16
70	0x146	FWD_R_F_SLOPE_TC34	Proport. factor slope between TC3 and TC4	fd16
71	0x147	FWD_R_F_OFFSET_TC2	Proportionality factor F at TC2	fd16
72	0x148	FWD_R_F_OFFSET_TC3	Proportionality factor F at TC3	fd16
73	0x149	FWD_R_F_OFFSET_TC4	Proportionality factor F at TC4	fd16
74	0x14A	FWD_SOUND_VEL_MAX	Maximum of speed of sound in m/s Default value for water 0x00061400	fd8
75	0x14B	FWD_1_BY_A	Medium constant Default value for water 0x002CA2E2	fd16
76	0x14C	FWD_CONST_C	Medium constant Default value for water 0x000F6C3A	fd24
77	0x14D	FWD_THETA_MAX	B3/B2/B1: Temperature at maximum speed of sound in °C / B0: minimal speed of sound Default value for water 0x004A002B	fd16 / fd-5
78	0x14E	FWD_LONG_TERM_ERROR	Number of low AM measurements before hardware failure / no-water checks are done Proposed value 0x00000020	fd0
79	0x14F	FWD_FHL_USER	B1/B0: trusted FHL ratio (option B) / or B0: absolute trusted FHL, LSB=0.88mV; see section 4.2	fd16 / fd0
80	0x150	FWD_TOF_SUM_DELTA	FHL method 3: Nominal time difference (raw TDC units) in SUMTOF between operating and trusted FHL; see section 4.2.3	fd16
81	0x151	FWD_TOFSUM_VAR_LIM	Error limit for deviation of SUMTOF from former average (raw TDC units); see section 4.2	fd16
82	0x152	FWD_HSC_DEV	Error limit for HSC calibration in raw TDC units (deviation time from reference measurement of 4 LSC periods); see chapter 3	fd16
83	0x153	FWD_ERR_INTERRUPT	Define error flag positions that issue an interrupt; see section 5.4 and 6.1.2 (RAM_R_FW_ERR_FLAGS)	bits

Cell #	Address	Variable name	Description and (if applicable) default value	Format
84	0x154	FWD_AM_DIFF_LIM	Error limit for deviation between currently measured amplitude UP and DOWN in mV see section 3.4	fd16
85	0x155	FWD_R_AM_MIN	Error limit for minimal amplitude in mV see section 3.4	fd16
86	0x156	FWD_PW_NOM	Nominal Pulse width for FHL option A; see section 4.2	fd7
87	0x157	FWD_PW_DEV	Error limit for deviation between currently measured UP and DOWN pulse width see section 3.4	fd7
88	0x158	FWD_TEST_CONFIG	Configuration value of CR_USM_TOF for FHL method 4; see section 4.2.4	fd16
89	0x159	FWD_TOF_RATE_FACTOR	Factor for TOF rate scaling in zero flow case	fd0
90	0x15A	FWD_FLOW_AVG_FACTOR	2^N number of flow values for averaging; this factor *16 determines the total number of samples for long term averaged flow, as used for the zero flow decision.	fd0
91	0x15B	FWD_R_PULSE_PER_LITER	Pulse interface: Number of pulses per liter	fd0
92	0x15C	FWD_R_PULSE_MAX_FLOW	Pulse interface / maxflow error limit: maximum permissible flow in l/h see section 2.6 and chapter 3	fd0
93	0x15D	FWD_NEG_FLOW_LIMIT	Cutoff limit for negative flow in l/h; positive values are ignored; see chapter 3	fd16
94	0x15E	FWD_R_TOF_DIFF_LIMIT	Minimum limit for DIFTOF values in raw TDC units. At lower DIFTOF , temporary zero flow is assumed and no calculation is done; see chapter 3	fd0
95	0x15F	FWD_ZERO_FLOW_LIMIT	Zero flow limit in l/h: When the absolute of the long term average flow is smaller, long term zero flow is assumed and the TOF rate is scaled by FWD_TOF_RATE_FACTOR; see chapter 3	fd16
96	0x160	FWD_CAL_PTR_OFFSETR	Reference branch offset resistance in internal reference in Ohms; see chapter 3; typical value 0x00000000 (calibrate if desired)	fd16
97	0x161	FWD_EXT_REF_VAL	Value of external reference resistor in Ohms; see chapter 3 typical value for PT1000: 1kOhm = 0x03E80000	fd16
98	0x162	<i>Unused</i>	<i>Unused</i>	
99	0x163	<i>Unused</i>	<i>Unused</i>	
100	0x164	FWD_PT_INT_SLOPE	Internal sensor resistance slope in Ohms/K; see chapter 3; typical value 0x0029F000 (calibrate if desired)	fd16
101	0x165	<i>Unused</i>	<i>Unused</i>	
102	0x166	<i>Unused</i>	<i>Unused</i>	
103	0x167	FWD_PT_INT_NOM	Internal sensor nominal resistance in Ohms;	fd16

Cell #	Address	Variable name	Description and (if applicable) default value	Format
			see section 3.7; typical value 0x03C20000 (calibrate if desired)	
104	0x168	FWD_PTC_RATIO_INV	Nominal ratio of reference resistor to PT cold sensor resistance at 0°C; see section 3.7; typical value 1 = 0x00010000	fd16
105	0x169	FWD_PTH_RATIO_INV	Nominal ratio of reference resistor to PT hot sensor resistance at 0°C; see section 3.7; typical value 1 = 0x00010000	fd16
106	0x16A	FWD_FW_CONFIG	acam firmware configuration register; see section 7.1	bits
107	0x16B	FWD_R1_FHL_VALUE	Start / fallback value of FHL, LSB=0.88mV; see chapter 4	fd0
108 - 123	0x16C - 0x17B	<i>Initial values for configuration registers</i>	If configured by FWD_FW_RLS (cell #123), the values of these cells are copied to configuration registers by the bootloader at startup and after reset See manual Vol.1.	
108	0x16C	FWD_R_CD	Watchdog disable code; see chapter 3 Without acam firmware, 0x48DBA399 disables the watchdog while any other code enables it. With acam firmware, the watchdog can't be disabled, but 0x00000000 disables configuration restore after recall Proposed value 0x95659C6A	bits
109	0x16D	FWD_PI_E2P	Configuration data for CR_PI_E2P	bits
110	0x16E	FWD_GP_CTRL	Configuration data for CR_GP_CTRL	bits
111	0x16F	FWD_UART	Configuration data for CR_UART	bits
112	0x170	FWD_IEH	Configuration data for CR_IEH	bits
113	0x171	FWD_CPM	Configuration data for CR_CPM	bits
114	0x172	FWD_MRG_TS	Configuration data for CR_MRG_TS	bits
115	0x173	FWD_TM	Configuration data for CR_TM	bits
116	0x174	FWD_USM_PRC	Configuration data for CR_USM_PRC ; acam firmware interprets B3 as TOF_RATE; see section 3.1.2	bits
117	0x175	FWD_USM_FRC	Configuration data for CR_USM_FRC	bits
118	0x176	FWD_USM_TOF	Configuration data for CR_USM_TOF	bits
119	0x177	FWD_USM_AM	Configuration data for CR_USM_AM	bits
120	0x178	FWD_TRIM1	Configuration data for CR_TRIM1 ; Set to 0x84A0C47C	bits
121	0x179	FWD_TRIM2	Configuration data for CR_TRIM2 ; Set to 0x401725CF	bits
122	0x17A	FWD_TRIM3	Configuration data for CR_TRIM3 ; Set to 0x00270808	bits
123	0x17B	FWD_FW_RLS	Bootloader release code 0xABCD7654 activates the bootloading process after startup and system reset	bits

Cell #	Address	Variable name	Description and (if applicable) default value	Format
124 - 127	0x17C - 0x17F	<i>Checksums for non-volatile memories</i>	<i>These checksums are stored for comparison to values calculated by the chip. Deviations in comparison cause checksum errors. Can be calculated by the GP30 PC software.</i>	
124	0x17C	FWD_R_FWD1_CS	Checksum Firmware Data 1	fd0
125	0x17D	FWD_R_FWD2_CS	Checksum Firmware Data 2	fd0
126	0x17E	FWD_R_FWU_CS	Checksum Firmware Code User	fd0
127	0x17F	FWD_R_FWA_CS	Checksum Firmware Code acam	fd0

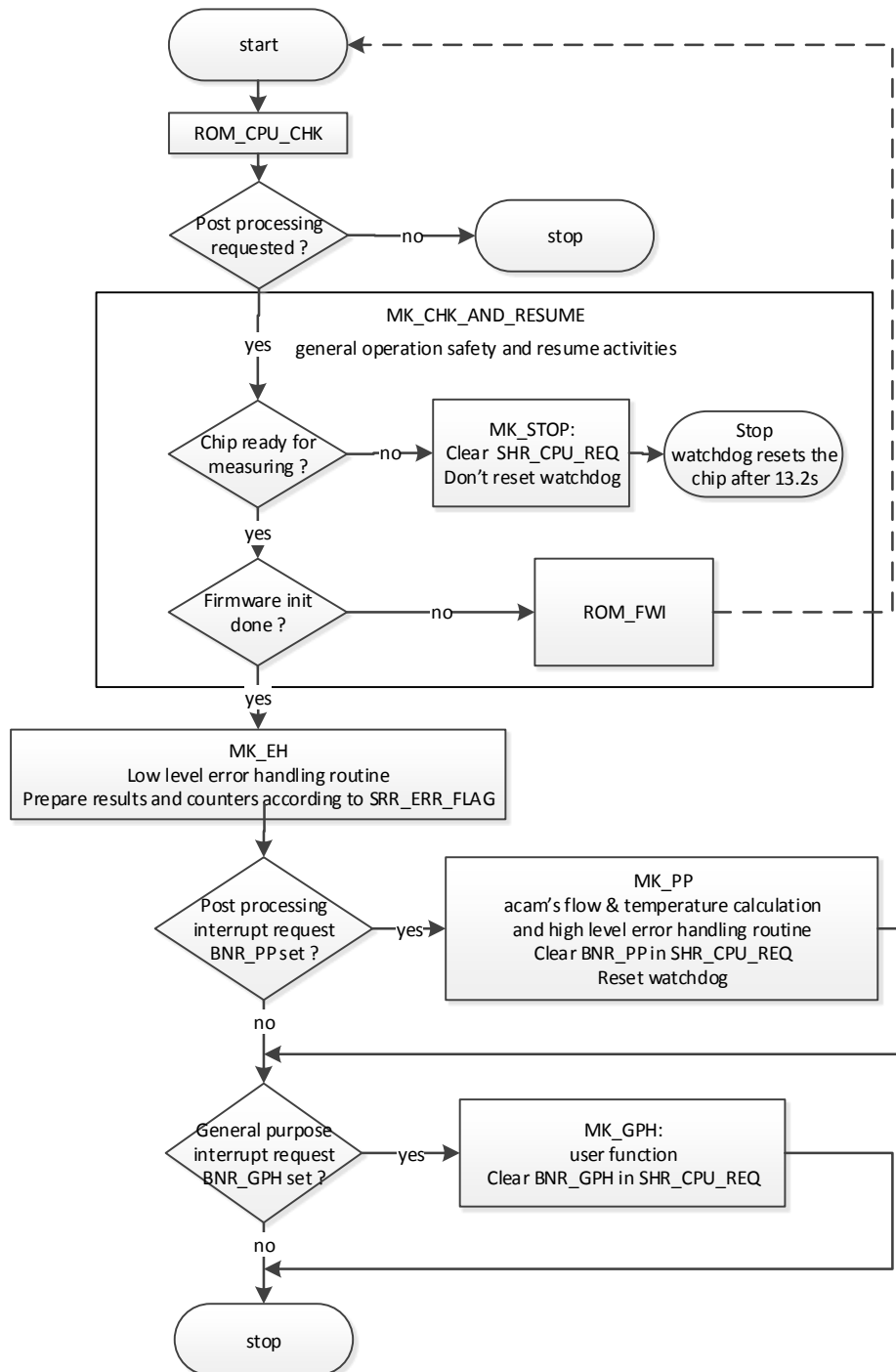
7.1 **FWD_FW_CONFIG** (Firmware configuration) 0x16A

Bit	Description	Format	Default
31	BNR_FWCONF_PWL 0: Apply acam calibration method 1: Apply PWL calibration method	BIT	b0
30	BNR_FWCONF_TSENS 0: Use temperature value from flow meas. for calibration coefficients 1: Use temperature value from hot sensor meas. for calibration coeffs.	BIT	B0
29	BNR_FWCONF_VOL 0: Don't apply flow volume storage protection 1: Apply flow volume storage protection	BIT	b0
28	BNR_FWCONF_ERR 0: Disable average error counters 1: Enable average error counters	BIT	b0
27	BNR_FWCONF_VLIM 0: Disable control of speed of sound limits 1: Enable control of speed of sound limits	BIT	b0
26	BNR_FWCONF_FHL_RATIO: Configuration for FHL regulation option B 0: Interpret FHL-values as fixed voltage 1: Interpret FHL-values as ratio to measured amplitude	BIT	b0
25:24	BNR_FWCONF_FHL: Configuration of FHL regulation methods 00: Method 1, fixed FHL 01: Method 2, consistency check against trusted FHL 10: Method 3, consistency check against trusted FHL with offset time 11: Method 4, consistency check against special configuration	BIT2	b00
23	BNR_FWCONF_TESTMODE: Configuration for FHL regulation option C 0: Enter FHL test mode regularly (each 32 measurements) 1: Enter FHL test mode only in error case	BIT	b0
22	BNR_FWCONF_PI_ERROR 0: Don't signal no-water as error on pulse interface 1: Also signal no-water as error on pulse interface	BIT	b0

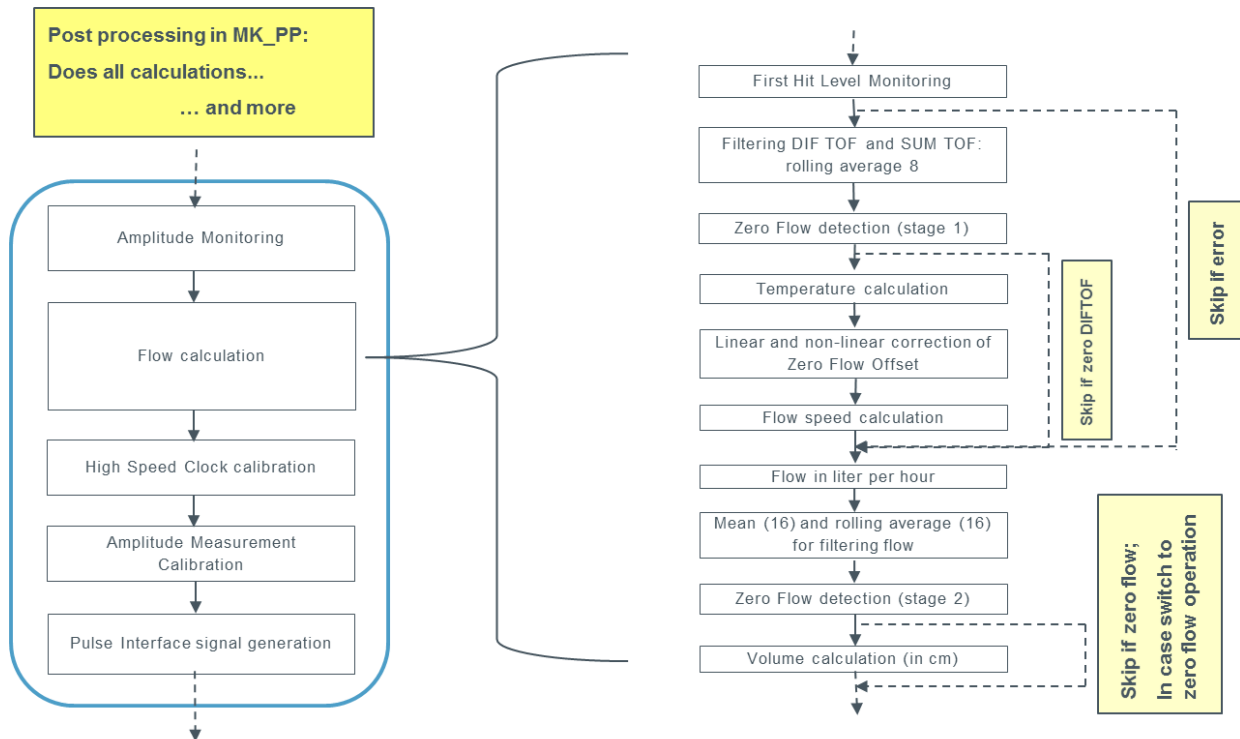
Bit	Description	Format	Default
21	BNR_FWCONF_NO_PI_ERR 0: Signal error on pulse interface as configured in bit 22 1: Never signal error on pulse interface	BIT	b0
20:18	<i>Not used</i>		
17	BNR_FWCONF_FHL_ZEROFLOW 0: With FHL regulation active, disable zero flow state (always assume full flow as long as FHL is considered not ok) 1: Apply zero flow state independently of FHL regulation	BIT	b0
16	BNR_FWCONF_2MAX_NOZERO 0: Set flow to zero when exceeding 2x maximum flow and signal error 1: Flow remains even when exceeding 2x maximum flow	BIT	b0
15:8	PWL_EXP (optional) Exponent of scaling factor for PWL coefficients: scale up each value by $2^{(PWL_EXP)}$	UINT [7:0]	0x01
7:0	PWL_ADDR (optional) Start address of PWL coefficients table in FWD (without leading address bit 8, which is always 1 in FWD addresses)	UINT [7:0]	0x10

8 acam firmware structure

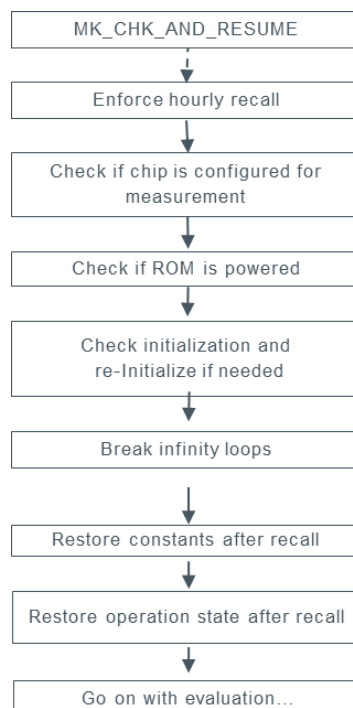
The general structure of the acam firmware is shown in the following flow chart. More detailed information on flow for error handling, bubble detection and FHL regulation is given in section 5.2, together with a dedicated flow chart for error handling. See also detail charts further below.



The routine **MK_PP** is the major calculation and high level error handling routine. Its internal structure is sketched below.



The routine **MK_CHK_AND_RESUME** is the major operation safety routine:



8.1 List of related files

When working with the unmodified acam firmware, only one type of file is needed, the firmware data file which contains chip configurations and calibrations. This file must be adapted to the particular flow meter system of the customer, see section 2.4 for details. The following list contains all available files related to acam firmware:

File name (for firmware version A1.A2.11.03)	Name	Description
GP30Y_config_default_A1.A2.11.03.cfg	Configuration file	This file contains a template configuration. It can be opened by the GP30 PC software. In contrast to the firmware configuration, it has post processing switched off and watchdog disabled, to permit operation without firmware.
GP30Y_A1.A2.11.03.dat	Firmware data file	This file is the major template for firmware data, containing configurations and calibrations. In production, each individual flow meter has its own firmware data.
GP30Y_A1.D2.11.03.asm	Open firmware assembler code	Open part of the acam firmware assembler code. This file is needed for restoring the original delivery state after modifications. It should be used as template for customizations.
GP30Y_A1.D2.11.03.h	Firmware header file	This general header file contains all variable definitions. It can also be used as quick reference.
GP30Y_A1.D2.11.03.hex	Downloadable firmware hex file	This file is generated by the GP30 compiler. It contains downloadable HEX code and corresponding assembler commands as comment. It can be opened by the GP30 PC software.
GP30Y_A1.D2.11.03_sim.hex	Compact firmware hex file	This file is generated by the GP30 compiler. It contains only HEX code and may be more suitable for downloading generated code over a microcontroller.
GP30Y_A1.D2.11.03.obj	Firmware object file	Firmware label and address list
GP30Y_A1.D2.11.03.dbg	Firmware debug file	HEX to assembler file line reference
GP30Y_REG_A1.2.h	GP30 registers header file	General variables definition for GP30 hardware registers
GP30Y_ROM_A1.common.h	GP30 common ROM routines file	Label definition for commonly usable ROM routines
GP30Y_UPD_A1.E2.11.01.h	GP30 routine update file	List of updated routines

8.2 acam firmware version numbers

TDC-GP30 stores two 4-Byte firmware version numbers, one for the (changeable) customer firmware code and one for the (fixed at delivery) acam firmware code. The two numbers are available after the chip's bootloader has stored them into registers **SRR_FWU_REV** (0x0ED) and **SRR_FWA_REV** (0x0EE), typically when the chip has bootloader release code set after power-on. They can also be read in the download window of the PC software after verify.

At delivery, the acam firmware consists of an open "customer firmware" part and a fixed "acam firmware" part. The source code of the open part is available to customers and may be modified (see chapter 0). When the open firmware part is modified, its version number should also be changed to indicate the modification. The following firmware version numbers should be distinguished ("X" and "Y" may be any cypher):

Description	Complete version number	Byte B3: ROM version	Byte B2: FW type and Version number	Byte 1: major and minor release number	Byte 0: build
General description	Unique version number for each firmware release	A1 is currently the only ROM version	The FW type is described by one letter; the version number is increased at essential functional changes	Major or minor release numbers are increased at major or functional changes	Build numbers are typically increased at bug fixes and for internal needs
acam beta firmware versions, acam code and initial user code	A1.A1.XX.YY	A1	A1: Type "A" denotes acam FW, Version "1" denotes beta phase	XX (several)	YY (several)
acam beta firmware versions, refreshed user code	A1.D1.XX.YY	A1	D1: Type "D" denotes a FW part which is not functional standalone	same as acam code of Type "A" with the same version number	Always the highest available; Contains latest bug fixes
acam production firmware, acam code and initial user code	A1.A2.11.0X	A1	A2: Type "A" denotes acam FW, Version "2" denotes production phase	11: First major and minor release	0X: 01...03 current build is 03
acam production firmware, refreshed user code	A1.D2.11.YY	A1	D2: Type "D" denotes a FW part which is not functional standalone	Same as acam code of Type "A" with the same version number	Always the highest available; Contains latest bug fixes
acam empty code	A1.E2.11.02	A1	E2: Type "E" denotes empty FW (contains e.g. trim parameters)	11: First major and minor release	02: Current build
acam example code	A1.F1.12.03	A1	A2: Type "F" denotes special FW	12: 1 st major and 2 nd minor release	03: Current build
Customer code	A1.CY.XX.YY	A1	CY: Type "C" denotes customer FW; use "C" to distinguish from standard FW products	XX (any number)	YY (any number)

9 Firmware with combined customer and acam code

As described in the second case of the overview, the customer may add his own code to the one provided by acam. Actually the TDC-GP30 chips will be delivered in the same state as in case of stand-alone usage of the firmware in section 1.1: The chips are pre-programmed with a part of the code memory read- and write-protected, and with a small main routine as interface. The structure of this main routine was described in chapter 8. As additional support on programming and code structure, a simplified free demo firmware is also delivered within the evaluation package. This demo code is described in chapter 11.

The customer has the free choice to add any code parts, for example enhanced calibration or error handling, data storage or different communication setups. The interfaces to acam's firmware are given on the one hand by the results in RAM cells, on the other hand by freely available subroutines. The customized user code can be placed before or after the main calculation routines of the acam code, just by modifying the open source code of the main routine. The following chip resources are available for user code:

- Firmware NVRAM usage (of 4kB available): ~ 1.1 kB
- RAM usage (of 176 x 32b-Words): ~ 23 - 31 words free / unused
~ 57 words available for temporary storage
- Firmware data usage (of 128 x 32b-Words): 20 words configuration (always)
> 42 words free

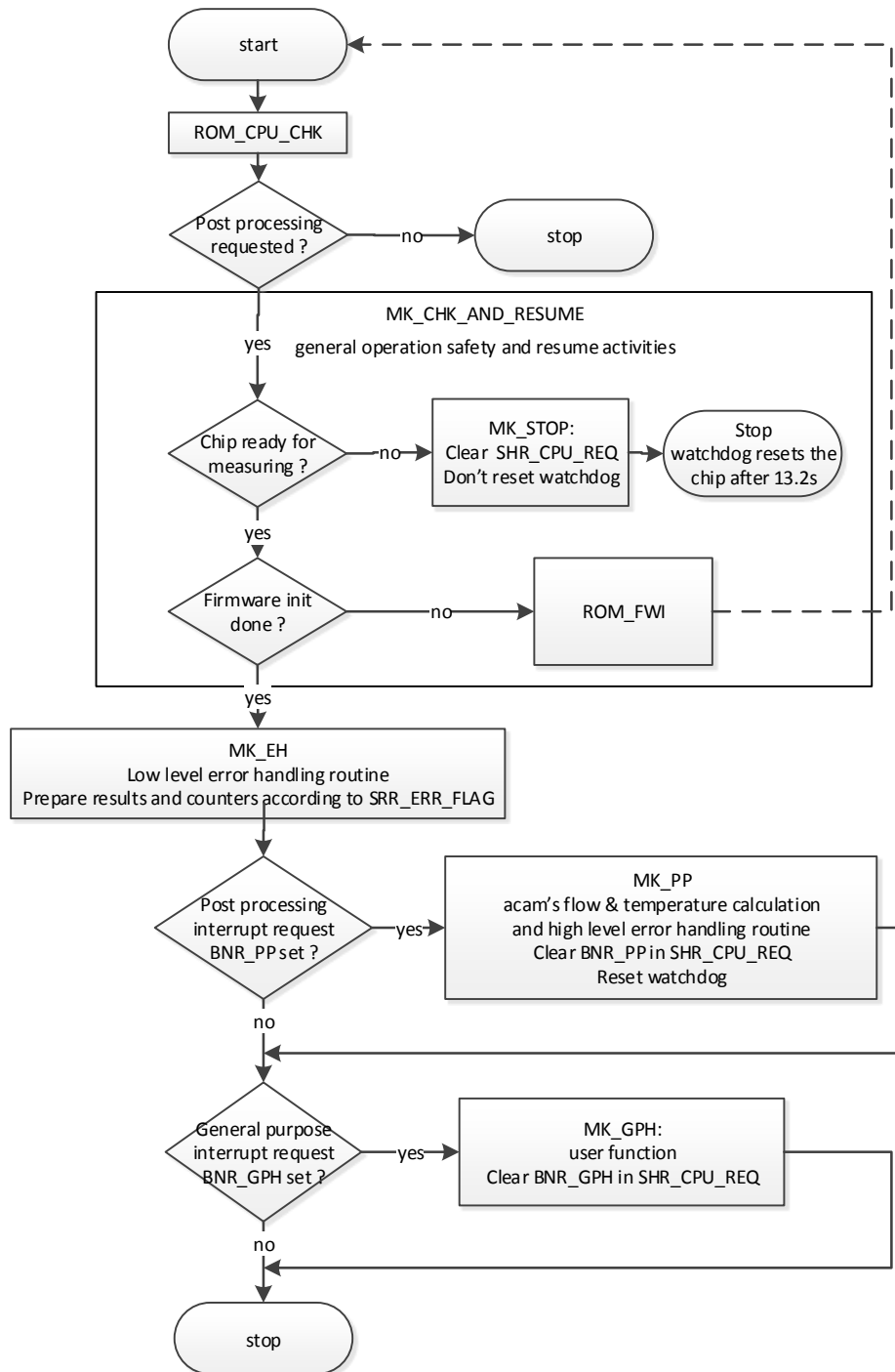
Together with assembler and other programming support, acam delivers a commented definition of memory allocation in file GP30Y_A1.D2.11.03.h. A complete overview of the RAM cell usage is also given in chapter 6. In addition, a number of ROM routines described in chapter 12 are freely available for customer programming. They are called just by their name, when the delivered files GP30Y_ROM_A1.common.h and GP30Y_UPD_A1.E2.11.01.h remain included in the source code. Please have a look into the header files listed in section 8.1 to see the actual memory allocation and check which RAM cells are usable.

The remarks on calibration in section 2.5 remain valid as far as the acam firmware part is used for calibration.

9.1 Internal firmware structure

acam's firmware is controlled by a main loop which does a first check of chip state, then calls an initialization routine and then an error handling routine. Then it jumps to post processing and to an empty template routine for general purpose handling. The whole process is controlled by interrupt requests issued by the chip's measurement rate generator (MRG) and the task sequencer (TS),

according to configuration. The flow diagram in the following figure gives an overview of the firmware main loop structure.



The user can do changes in any part of this loop structure.

There are a few more constructs contained in the main loop code template, namely the include instructions for the four header files and some constants definitions for firmware revision placement: ***FW_VERSION_NUM, FW_VERSION_MAJ, FW_VERSION_MIN, FW_VERSION_BLD***. Together with ***FW_ROMVERSION_REV*** (=A1; leave unchanged) they make up the version number. It is good practice to keep track of firmware versions by changing the version numbers accordingly. If the values of these constants are changed, and the version statement at the end of the code remains as it is, the version number will also appear in the register variable ***SRR_FWU_REV***, and will be displayed in the download window of the PC software. For version number definition see section 8.2.

9.2 Programming the chip

To program GP30, a firmware code is written to the non-volatile FWC (firmware code) memory. This is in general possible over either SPI or UART interface. At the beginning, usually acam's PC software is used as interface. For details on interface communication and PC software usage, please refer to Manual Volumes 1, 2 and 3. In the following, only the major steps are sketched.

The open part of acam's firmware is stored on the chip on delivery, but it is also supplied as source code file GP30Y_A1.D2.11.03.asm and as already compiled hex file GP30Y_A1.D2.11.03.hex with the evaluation package. The package also contains four header files:

- GP30Y_A1.D2.11.03.h: Memory and variable name definition
- GP30Y_ROM_A1.common.h: ROM routine start address definition
- GP30Y_UPD_A1.E2.11.01.h: Addresses of updated ROM routines
- GP30Y_REG_A1.2.h: Addresses and flag bits of hardware registers

Each of these files defines variable names that can be used in programming. Please have a look into the files and refer to Manual Volumes 1, 2 and 3 if the internal comments are not sufficient.

To recover a modified chip to delivery status, simply open the original GP30Y_A1.D2.11.03.hex file in the download window of the PC software, left side under "Firmware User Code". Stop the measurement and click "Download FW Code". After a system reset, you may now start the measurement again.

For actual programming work, the first step is to store the delivered assembler file GP30Y_A1.D2.11.03.asm and the four header files in one directory, and to open the assembler file in the assembler window (called from menu "Firmware"). The acam assembler window is an editor with syntax highlighting according to the assembler instructions. Details on instructions and code structure can be found in Manual Volume 2: CPU, Memory and Firmware. After some programming work is finalized, the code can be compiled from menu "Assembler" – compiler messages appear in the bottom frame. Selecting "Download" from the "Assembler" menu opens the already known download window. The code is then stored on the chip the same way as in case of recovery above. Of course, any of the delivered source code and header files can be modified. However, the

recommendation is to do customizations only in copies of GP30Y_A1.D2.11.03.asm and GP30Y_A1.D2.11.03.h – don't forget to change the file names in INCLUDE instructions of the assembler code accordingly !

It should be noted that of course again a definition of firmware data and a calibration are needed, as described in section 2.4, and that error handling has to be considered. When a user customizes acam's firmware as described above, it is of course his decision which results should be modified and which additional actions should be taken. As far as the structures and actions of the acam code remain, anything discussed in the preceding chapters remains valid.

Finally, it should be noted that the user can read protect his firmware user code as well as the firmware data. Refer to Manual Volume 2: CPU, Memory and Firmware.

10 Full customer firmware code

As described in the last chapter, firmware coding is supported by an assembler which is integrated in the evaluation software (for details see the Manual Volume 2: CPU, Memory and Firmware). In addition, a number of ROM routines described in chapter 12 are freely available for customer programming.

In contrast to the cases in preceding chapters where the acam firmware is used, either unmodified or modified, the case of a full customer firmware is essentially different. The TDC-GP30 chips come unprogrammed, except for some minor software updates placed at the end of the code memory – this small, write-protected code part is always present and used for updates and bug fixes by acam.

The whole process of assembler coding and chip programming remains as described in section 9.2, but now the customer has to do the complete programming. Included in the evaluation package, acam delivers a small demo code that can be helpful for understanding code structures. The next chapter gives a short description of this code.

Again, it should be noted that the user can read protect his firmware user code as well as the firmware data. Refer to Manual Volume 2: CPU, Memory and Firmware.

11 Demo code: Example_DIF_OVER_PI

The demo code provided by acam actually turns a spool piece equipped with TDC-GP30 into a simple flow meter with only proportional linear calibration. Since its memory allocation is fully compatible with acam's firmware, results can be displayed by the PC software the same way as with the firmware (see section 2.1). In addition, the pulse interface is configured and can be used to count water volume. Of course, the code also reveals the shortcomings of such a simple approach: Zero offsets will give a signal without flow (small in TDC-GP30, but still not negligible), and small flows will

exhibit undesired relative errors. So the demo code is good for quickly setting up and running a complete program, but it is also good to understand why more complicated calibrations are really needed.

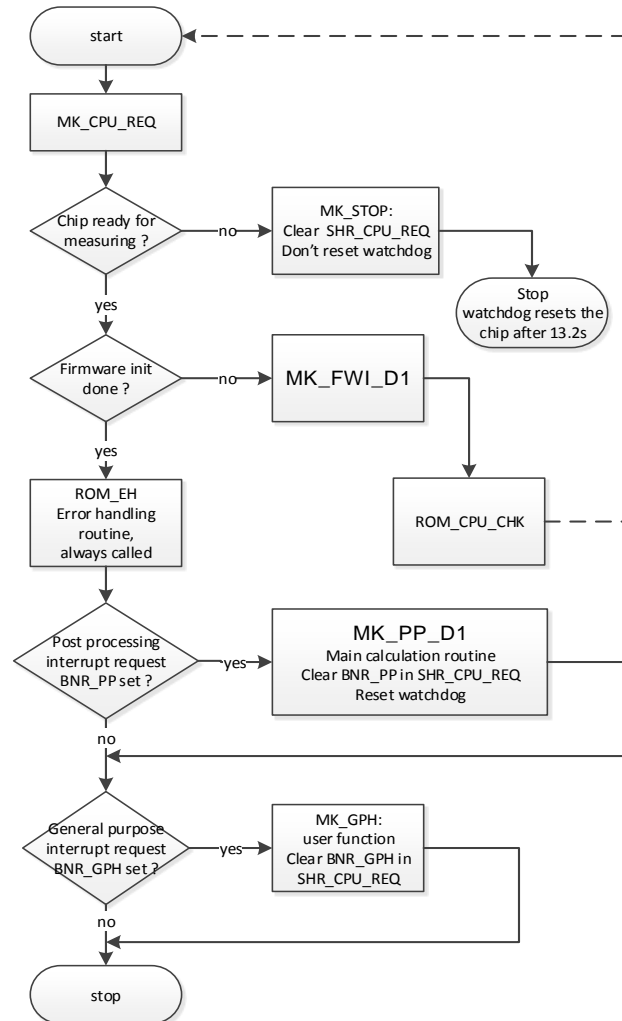
The evaluation package contains the complete source code GP30Y_A1.F1.12.01.asm and its compiled downloadable hex code GP30Y_A1.F1.12.01.h, as well as the required header files

- GP30Y_A1.F1.12.01.h: Memory and variable name definition
- GP30Y_ROM_A1.common.h: ROM routine start address definition
- GP30Y_UPD_A1.E2.11.01.h: Addresses of updated ROM routines
- GP30Y_REG_A1.2.h: Addresses and flag bits of hardware registers

As with the firmware codes, a firmware data template is included, too: GP30Y_A1.F1.12.01.dat. This file has far less content than the firmware data file for the complete firmware. It still needs to be adapted to the user's device by updating the configuration as described in section 2.4.2. This applies mainly to the configuration registers. Apart from that, only four variables have to be adjusted. This can be seen in the following table which lists the relevant cells of the demo firmware data file:

Cell #	Address	Variable name	Description and (if applicable) default value	Format
0	0x100	FWD_SIMPLE_SCALE	Scaling factor between DIFTOF in ns and flow in l/h 0x000A0000 = 10 (l/s)/ns (example)	fd16
91	0x15B	FWD_R_PULSE_PER_LITER	Number of pulses per liter for the pulse interface 0x0000000A = 10 pulse /l (example)	int
92	0x15C	FWD_R_PULSE_MAX_FLOW	Maximum flow for the configuration of the pulse interface in l/h 0x00000BB8 = 3000 l/h (example)	fd0
93	0x15D	FWD_R_FHL_VALUE	First wave level (in 0.88 mV). This value stands here for compatibility reasons, it will become obsolete in later versions (compare Nr 107). This number must be adapted to the measurement device. 0x00000055 = 85*0.88 mV (example)	fd0
107	0x16B	FWD_R1_FHL_VALUE	First wave level (in 0.88 mV). This value stands here for compatibility reasons, it will become obsolete in later versions (compare Nr 107). This number must be adapted to the measurement device. 0x00000055 = 85*0.88 mV (example)	fd0
108 - 123	0x16C - 0x17B	<i>Initial values for configuration registers</i>	If configured by FWD_FW_RLS (cell #123), the values of these cells are copied to configuration registers by the bootloader at startup and after reset See manual Vol.1.	

The demo example has basically the following structure:



The essential parts are the routines

- MK_FWI_D1: Firmware initialization. The demo code contains an open and fully readable simplified version of MK_FWI, such that basic techniques for initialization can be read and tested there.
- MK_PP_D1: The main calculation routine now just calculates DIFTOF and scales it with **FWD_SIMPLE_SCALE** as described in cell number 0 of the preceding table.

Apart from that, the simple ROM routine ROM_CALC_TOF_DIFF is copied into the source code, just for information and as example. Please have a look into the source code file GP30Y_A1.F1.12.01.asm to see the details of the implementation.

The user should feel free to operate and modify this example. Programming, downloading and operating this code or any other customer code remains as described before.

12 ROM routines

12.1 Overview

TDC-GP30 contains 4kB ROM code memory which is currently filled up to 95% with over 50 routines. Some are usable by each customer, while some others represent special functions for acam's firmware. An overview list of subroutines for general usage is given in section 12.2. For details see Manual Volume 2: CPU, Memory and Firmware.

Remark: Depending on customer demands, most of these subroutines can be replaced by custom defined routines. It is possible to use up to about 3.5kB of the ROM code for customer specific purposes. If this option is used, a new ROM mask is required (note: this causes extra cost and lead time).

12.2 Currently available ROM routines for general usage

A part of the implemented ROM routines are not suitable for general purpose or customer usage. They serve as firmware memory extension for acam's firmware or as hardware support, for example for initialization. The second group of ROM routines is made to provide functions of general interest. These routines are listed in the following table.

Name	Description	Remarks
Filtering		
ROM_INIT_FILTER	Routine to initialize the RAM cells for any filter (rolling average or median) with a given value	
ROM_ROLL_AVG	Routine to filter the FILTER_IN values using a rolling average filter	Filter length can be configured
ROM_ROLLAVG_OUTLIER	Routine to filter the FILTER_IN values using a rolling average filter. One value which deviates most is always ignored.	Filter length can be configured
ROM_MEDIAN	Routine to filter the FILTER_IN values using a median filter	Filter length can be configured
ROM_FILTER_FLOW	Routine to filter flow values using the standard rolling average filter, including initialization.	Filter length can be configured
Error detection and handling		
ROM_EH	This routine checks all error flags and suppresses processing of wrong results.	many RAM cells fixed
ROM_PP_AM_MON	Monitor the amplitude values and check limits to identify bad measurements	alternative calls exist
ROM_PP_AM_CALIB	This routine gets the Amplitude Calibration values (H & L) and evaluates the gradient and offset that can be used for calculating the actual amplitude.	alternative calls exist

Pulse interface and flow volume		
ROM_CFG_PULSE_IF	This routine configures the pulse interface with the parameters calculated from the given configuration.	
ROM_PI_UPD	Pulse Interface Update Routine	
ROM_PP_PI_UPD	Pulse Interface Update Routine with input from RAM	
ROM_RECFG_PULSEIF_FOR_ERROR1	Reconfiguring the pulse interface outputs GPIO0 and GPIO1 as normal GPIOs to signal an error	
ROM_SIGNAL_ERROR_ON_PULSEIF1	Signaling error on the pulse interface GPIO0 and GPIO1	
ROM_RECFG_PULSEIF_FOR_PULSE1	Configuring GPIO0 and GPIO1 as pulse interface outputs	
ROM_SAVE_FLOW_VOLUME	This routine is used to store the converted flow (in LPH), cumulatively to flow volume in cubic meter.	alternative versions exist
Sensor temperature measurement		
ROM_TEMP_POLYNOM	Calculates the temperature of a PT sensor using a polynomial approximation	
ROM_TEMP_LINEAR_FN	This routine is used to calculate the temperature of any sensor as a linear function of sensor resistance using the nominal resistance and sensor slope.	
ROM_TM_SUM_RESULT	Sums up the results of double temperature measurements. The double measurements are performed to eliminate the 50/60 Hz disturbance.	
Interface communication		
ROM_I2C_ST	I2C Start Byte Transfer	
ROM_I2C_BT	I2C Byte Transfer	
ROM_I2C_LT	I2C Last Byte Transfer	
ROM_I2C_DWORD_WR	Write 4 bytes of data to a specified address through the I2C interface	
ROM_I2C_BYTE_WR	Write a single byte of data to a specified address through the I2C interface	
ROM_I2C_DWORD_RD	Sequentially read 4 data bytes from the I2C interface	
ROM_I2C_BYTE_RD	Sequentially read a single data byte from the I2C interface	
ROM_COPY_UART_PRB_DATA	This routine is used to copy the relevant data for the UART Master into the Probe data area. The data is then sent over UART interface to the master.	alternative calls for optimized memory
Housekeeping		
ROM_CPU_CHK	Check kind of CPU request: This routine is called by hardware design after any Post Processing (PP) request, it is the starting point of any CPU activity, including the firmware call at MK_CPU_REQ.	automatically started
ROM_USER_RAM_INIT	Initialize the entire user RAM with 0	

High speed oscillator		
ROM_HSO_WAIT_SETTL_TIME	This routine is used to switch on the High speed oscillator clock and wait out its settling time (122 us)	
ROM_HSC_CALIB	This routine evaluates the high speed clock scaling factor for the 4 MHz / 8 MHz clock	
ROM_SCALE_WITH_HSC	Routine to scale the input parameter with the HS Clock Calibration factor	

Configuration		
ROM_RESTORE_TOF_RATE	Routine to reconfigure TOF_RATE generator from a lower rate after ZERO FLOW ends	alternative version for free configuration
ROM_RECFG_TOF_RATE	Routine to reconfigure TOF_RATE generator to a lower rate, depending on the parameter N	
Mathematics		
ROM_FORMAT_64_TO_32BIT	Routine to format a 64-bit value (in Y and X) into a 32 bit result with 16 integer + 16 fractional bits. Useful for formatting 64 bit multiplication results with 32 integer + 32 fractional bits	alternative version: faster, but needs temporary RAM
ROM_DIV_BY_SHIFT	Perform the division of a value Y by X, where $X=2^N$ is an integer power of two	
ROM_SQRT	Evaluate the square root accurately for values in the range $(196 \leq X \leq 5476)$	
ROM_LINEAR_CORRECTION	Linear interpolation of a coefficient between two sampling points	alternative version is fixed to interpolation over THETA
ROM_FIND_SLOPE	Used to find the slope between two points, given the coefficient values and parameter values at the two points.	

13 Glossary

Terms	Meaning	GP30 interpretation
AM	Amplitude measurement	This is a peak measurement of the received signal amplitude. It can be configured to be executed in different time frames, which allows to pick the overall signal maximum (to control the signal level), or to measure only the peak of a selected number of ->wave periods. The latter allows for a more detailed receive signal analysis.
Backup	Permanent storage of a data copy	GP30 is prepared for an external data backup, foreseen over the built-in I2C-bus, which permits write and read with an external EEPROM. In principle, a user may also utilize the ->GPIOs for his own interface implementation for external backup.
Bootloader	System routine that initializes CPU operation	Typically after a system reset, first time when the ->TS calls the ->CPU, the bootloader routine is called. If the ->Firmware is released, the bootloader loads the chip configuration from FWD into CR and does other hardware initialisations like reading firmware revision numbers and calculation of checksums.
Burst	Analog signal containing a number of ->wave periods	For a flow measurement, a ->fire burst, that means a fixed number of ->wave periods of the measurement frequency, is send over a ->transducer into the flow medium. After some travel time (see ->TOF), a receive burst appears at the opposed transducer, which is detected as a number of ->hits. Note that the peak amplitude of the receive burst must not exceed -> V_{ref} to avoid negative voltages.
Calibration	Parameter adjustment to compensate variations	In GP30, different calibration processes are implemented and needed for high quality measurements: ->Firmware calibrations: Flow and temperature calibration, but also the ->FHL adjustment are under full control of the firmware. Half-automated calibrations: ->AM calibration and ->HSO calibration are based on dedicated measurements, initiated by the ->TS on demand. The actual calibrations need further evaluation by the firmware. Fully hard-coded calibrations: these calibrations need no interaction from firmware. One example is ->ZCD level calibration, which only needs to be initiated by the ->TS frequently. Another example is ->TDC calibration which happens automatically before each measurement.
CD	Configuration Data	16 x (up to) 32b words of ->flash memory for configuration of the chip, address range 0x16C - 0x17A (->NVRAM). Is copied to ->CR for actual usage.
Comparator	Device that compares two input signals	See ->ZCD-comparator
CPU	Central Processing Unit	32b processor (Harvard architecture type) for general data processing. The CPU has a fixed instruction set and acts directly on its three input- and result-registers ->X, Y and Z as well as on addressed RAM. The fourth register of the CPU is the ->RAM address pointer R. Instructions for the CPU are read as ->FWC or ->ROM code at an address given by the ->program counter.
CR	Configuration Register	The chip actually uses for its hardware configuration a copy of the ->CD into the CR address range 0x0C0 - 0x0CF (see ->direct mapped registers).
CRC	Cyclic Redundancy Check	Method for checksum calculation to control data integrity, employed in GP30 for ->UART communication.
C0G		Material of a ceramic capacitor with a very low temperature drift of capacity
DIFTOF, DIFTOF_ALL	Difference of up and down ->TOF	The difference between up and down ->TOF is the actual measure for flow speed. (see also ->SUMTOF). DIFTOF_ALL is the DIFTOF using ->TOF_ALL results, averaged over all TOF ->hits.

Terms	Meaning	GP30 interpretation
Direct mapped registers	Registers with direct hardware access	These register cells are not part of some fixed memory block, they rather have individual data access. This makes them suitable for hardware control. See ->SHR, ->SRR, ->CR and ->DR. Labels have the according prefix.
DR	Debug Register	Internal registers of the ->CPU, mapped to the RAA address range 0x0F8 – 0x0FB in debug mode.
FEP	Frontend Processing	Task of the ->TS where frontend measurements are performed
FDB	Frontend data buffer	Part of the -> RAM where the -> frontend temporarily stores its latest measurement results (-> RAA address range from 0x80 up to maximally 0x9B)
FHL, V_{FHL}	First hit level	Voltage level similar to the ->ZCD level, but shifted away from Zero level, for save detection of a first ->hit. The FHL determines, which of the ->wave periods of the receive -> burst is detected as first hit. It thus has a strong influence on ->TOF and must be well controlled, in order to achieve comparable TOF measurements.
Fire, fire burst, fire buffer	Send signal ->burst	The measurement signal on sending side is called fire burst, its output amplifier correspondingly fire buffer.
Firmware	Program code (in a file) for chip operation	The program code can be provided by acam or by the customer, or a combination of both. The complete program code becomes the ->FWC (firmware code) when stored in the ->NVRAM. The term firmware is in general used for all firmware programs, no matter if they make up the complete FWC or not.
Flow meter mode	Operation mode of GP30 as full flow meter system	In flow meter mode, the TDC-GP30 also performs further evaluation of ->TOF results, to calculate physical results like flow and temperature. To do this, it uses a ->firmware running on its internal CPU. See for comparison -> time conversion mode
Frontend	Main measurement circuit block	This part of the GP30 chip is the main measurement device, containing the analog measurement interface (including the -> TDC). The frontend provides measurement results which are stored in the ->FDB.
FWC	Firmware Code	Firmware code denotes the complete content of the ->NVRAM's 4kB section (address range 0x0000 to 0x 0FFF). The difference to the term ->firmware is on the one hand that firmware means the program in the file. On the other hand, a particular firmware may provide just a part of the complete FWC. FWC is addressed by the CPU's program counter, it is not available for direct read processes like RAM.
FWD	Firmware Data	The firmware configuration and calibration data, to be stored in the ->FWD-RAM
FWD-RAM	Firmware Data memory	128 x 32b words of ->NVRAM (built as volatile ->SRAM and non-volatile flash memory). The FWD-RAM is organized in two address ranges, FWD1 (-> RAM addresses 0x100 - 0x11F) and FWD2 (RAM addresses 0x120 – 0x17F). Main purpose is calibration and configuration Due to its structure, FWD-RAM can be used like usual ->RAM by the firmware. But note that with every data recall from flash memory the contents of the SRAM cells get overwritten.
GPIO	General purpose input/output	GP30 has up to 7 GPIO pins which can be configured by the user. Some of them can be configured as ->PI or ->I2C-interface.

Terms	Meaning	GP30 interpretation
Hit	Stands for a detected wave period	<p>The receive ->burst is typically a signal which starts with ->wave periods of the measurement frequency at increasing signal levels. While the first of these wave periods are too close to noise for a reliable detection, later signal wave periods with high level can be detected safely by the ->ZCD-comparator. The comparator converts the analog input signal into a digital signal, which is a sequence of hits. To detect the first hit at an increased signal level, away from noise, the input signal is compared to the ->FHL. After the first hit, the level for comparison is immediately reduced to the ->ZCD level, such that all later hits are detected at zero crossing (note that the ZCD level is defined to zero with respect to the receive signal, it is actually close to -> V_{ref} or another user-defined level).</p> <p>Different hits are denoted according to their usage:</p> <ul style="list-style-type: none"> ▪ Hit (in general) stands for any detected ->wave period. ▪ First hit is actually the first hit in a ->TOF measurement (not the first wave period!) ▪ TOF hits means all hits which are evaluated for ->TOF measurements. Note that typically the first hit is not a TOF hit. ▪ Start hit is the first TOF hit. This is typically not the first hit, but (according to configuration) some well-defined later hit. Minimum the 3rd hit has to set as Start hit. ▪ Stop hit is the last TOF hit. It is also defined by configuration and should not be too close to the end of the receive ->burst. ▪ Ignored hits are all hits which are not evaluated for the TOF measurement: All hits between first hit and start hit, as well any hit between TOF hits or after the stop hit.
HSO	High speed oscillator	The 4 or 8 MHz oscillator of the GP30. In usual operation only switched on when needed, to reduce energy consumption. This is the time base for ->TDC measurements. The HSO is typically less accurate than the ->LSO. It should be frequently ->calibrated against the LSO to obtain the desired absolute accuracy of the ->TDC.
INIT	Initialization process of ->CPU or -> FEP	In GP30 terminology, INIT processes don't reset registers or digital IOs, while -> reset does at least one of it. Several different INIT processes are implemented, see chapter "Reset hierarchy" for details.
IO	Input/output	Connections to the outside world for input or output
I2C	Inter-Integrated Circuit bus	Standard serial bus for communication with external chips. Implemented in GP30 only in part for EEPROM data exchange.
LSO	Low speed oscillator	The 32768 Hz crystal oscillator of the GP30. This oscillator controls the main timing functions (->MRG and ->TS, real time clock).
MRG	Measurement Rate Generator	The measurement rate generator controls the cyclic ->tasks of GP30 by setting task requests in a rate defined by configuration (->CR). When the MRG is activated, it periodically triggers the ->TS for initiating the actual ->tasks.
NVRAM, NVM	Programmable Non-Volatile Memory	GP30 contains two sections of programmable non-volatile memory: One section of 4kB ->FWC memory, and another of ->FWD-RAM (FWD1:-> RAM addresses 0x100 - 0x11F and FWD2: RAM addresses 0x120 - 0x17F), in total 128 x 32b words. It is organized as a volatile SRAM part which is directly accessed from outside, and a non-volatile flash memory part.
PI	Pulse interface	Standard 2-wire interface for flow output of a water meter. Typically outputs one pulse per some fixed water volume (e.g. one pulse per 0.1 l), while the other wire signals the flow direction. Permits stand-alone operation and is fully compatible to mechanical water meters.
PP	Post Processing	Processing activities of the -> CPU, typically after frontend processing (e.g. a measurement), initiated by ->TS

Terms	Meaning	GP30 interpretation
Program counter	Pointer to the current code address of the ->CPU	The program counter addresses the currently evaluated ->FWC or ->ROM-code cell during ->CPU operation. The program counter always starts at 0xF000, when any CPU action is requested. If any kind of firmware code execution is requested, the program counter is continued at 0x0000 (for FW initialization, post processing or general purpose handling).
PWR	Pulse width ratio	Width of the pulse following the first ->hit, related to the pulse width at the start hit. This width indicates the position of the ->FHL relative to the level of the detected ->wave period and thus gives some information on detection safety (small value means FHL is close to the peak amplitude and the desired wave period may be missed due to noise; large value indicates the danger that an earlier wave period may reach FHL level and trigger the first hit before the desired wave period).
R	RAM address pointer of the CPU	The ->CPU acts on the data of the ->X-, Y- and Z-register and on one single RAM cell. The pointer R defines the address of the current RAM cell.
RAA	Random Access Area	Address range from 0x000 to 0x1FF covering the ->RAM addresses. Memory cells within this address range can all be read, most of them can also be written (except ->SRR and ->DR). The RAA covers memory cells of different technology: ->RAM (including ->FDB), ->FWD-RAM (including ->CD), ->direct mapped registers (->SHR, ->SRR, ->CR and ->DR).
RAM	Random Access Memory	176 x 32b words of volatile memory, used by ->FDB and ->Firmware. Address range 0x000 to 0x0AF
RAM address	Address of a cell in the RAA range	A RAM address is used by the firmware or over ->RI to point to a memory cell for data storage or retrieval. Note that RAM addresses cover not only actual RAM, but all cells in the RAA range. Address range from 0x000 to 0x1FF
Register	Memory cell for dedicated data storage	Memory cells are typically called register when they contain flags or configuration bits, or when they have a single dedicated purpose (see ->CPU, ->CR, ->SHR and ->SRR).
Reset	Reset of the chip	GP30 has different processes and commands that can call resets and initializations at different levels. Some of them refresh ->CR or GPIO state, others just (re-) initialize CPU or frontend. The latter are rather denoted ->INIT. See chapter "Reset hierarchy" for details.
RI	Remote Interface	Interface for communication with a remote controller (see ->SPI and ->UART)
ROM	Read Only Memory	4kB of fixed memory, contains hard coded routines for general purpose and parts of acam's ->firmware (ROM code). Address range 0xF000 – 0xFFFF. The ROM code is addressed by the CPU's program counter, it is not available for direct read processes like RAM.
ROM code	Hard coded routines in ROM	See -> ROM.
SCL	Serial Clock	Serial clock of EEPROM interface
SDA	Serial Data	Serial data of EEPROM interface
SHR	System Handling Register	Registers that directly control chip operation. The data & flags of system handling registers have a dynamic character. They are typically updated by post processing, but have to be initially configured before measurement starts.
SPI	Serial Peripheral Interface	Standard interface for communication of the GP30 with an external master controller (alternative to ->UART).
SRAM	Static RAM	GP30 does not use any dynamic RAM, in fact all RAM in GP30 is static RAM. However, the term "SRAM" is in particular used for the RAM-part of the ->NVRAM.
SRR	Status & Result Register	The SRR-registers describe the current state of the chip. They are set by the chip hardware and contain error and other condition flags, timing information and so on.

Terms	Meaning	GP30 interpretation
SUMTOF, SUMTOF_ALL	Sum of up and down TOF	The sum of up and down ->TOF is a measure for the speed of sound in the medium, which can be used for temperature calculation. SUMTOF_ALL is the SUMTOF using ->TOF_ALL results, averaged over all TOF ->hits.
Supervisor	Functional block of GP30 that controls voltage and timing	The supervisor of TDC-GP30 controls chip operation and timing through the measurement rate generator (->MRG) and the task sequencer (->TS). It also covers voltage control and adjustment functions as well as the main oscillators -> LSO and ->HSO
Task	Process, job	The term task is used for a process which aims at fulfilling some fixed purpose, separate from other tasks with different goals. Typical tasks in GP30 are ->TOF measurement, temperature measurement (-> TM), post processing (-> PP), remote communication and voltage measurement.
Time conversion mode	Remotely controlled operation of GP30	In time conversion mode, the TDC-GP30 mainly acts as a ->TOF measurement system. It may operate self-controlled or remotely controlled, but it does no further result evaluation. This operation mode is similar to the typical usage of the acam chips GP21 and GP22. For comparison see ->Flow meter mode
TDC	Time-to-digital-converter	The core measurement device of GP30. Measures times between a start- and a stop-signal at high accuracy and high resolution. The internal fast time base of the TDC is automatically ->calibrated against the ->HSO before each measurement.
TOF, TOF_ALL	Time of Flight	Basic measurement result for an ultrasonic flow meter: The time between send and receive ->burst (with some offset, depending on ->hit detection). Measurements of TOF are done in flow direction (down TOF) and in the opposite direction (up TOF). GP30 also provides the sum of all TOF ->hits in the values TOF_ALL.
TS	Task Sequencer	The task sequencer arranges and initiates the ->tasks which are requested by the ->MRG in one measurement cycle or which are initiated remotely.
TM	Temperature measurement	This task means a temperature measurement using sensors, in contrast to temperatures which are calculated results from a TOF measurement (see -> SUMTOF)
Transducer	Electromechanical conversion device	Transducers for flow measurements are piezoelectric devices that convert an electrical signal into ultrasound and reverse. They are usually matched to the flow medium (e.g. water). GP30 can connect directly to the send and receive transducer.
UART	Universal Asynchronous Receiver & Transmitter	Standard interface for communication of the GP30 with an external master controller (alternative to ->SPI).
USM	Ultrasonic measurement	The principle of an ultrasonic flow meter is to measure ->TOFs of ultrasound in flow direction and against it, and to calculate the flow from the result. See also ->transducer.
V_{ref}	Reference voltage	The analog interface of GP30 refers to V_{ref} , a nominal voltage for -> V_{ZCD} of typically 0.7V. This makes it possible to receive a DC-free AC-signal with a single supply voltage. Up to the level of V_{ref} , negative swings of the receive signal are avoided.
V_{ZCD}	Zero cross detection level	This voltage level represents the virtual zero line for the receive ->burst. It is normally close to -> V_{ref} , just differing by the offset of the ->ZCD-comparator. Needs frequent ->calibration to compensate the slowly changing offset. Optionally, this voltage can be configured differently in SHR_ZCD... through the firmware.
Watchdog, watchdog clear	Reset timer for chip re-initialization	The watchdog of GP30 ->resets the chip (including ->CR refresh) if no watchdog clear (->firmware command clrwdt) within 13.2s (typically) is executed. This is a safety function to interrupt hang-up situations. It can be disabled for remote control, when no firmware clears the watchdog automatically.

Terms	Meaning	GP30 interpretation
Wave period	One period of the signal wave	A period of typically 1us length for a 1 MHz measurement frequency. This may be a digital pulse, for example when sending, or a more sinusoidal wave when receiving. Fire or receive ->bursts are sequences of wave periods.
X-, Y- and Z-register	Input- and result registers of the CPU	The ->CPU acts on these ->registers for data input and result output.
ZCD	Zero cross detection	All ->hits following the first hit are detected when the received signal crosses a voltage level V_{ZCD} , defined as zero with respect to the receive ->burst. In contrast, the first hit is detected when the received signal crosses the different voltage level V_{FHL} (->FHL).
ZCD-Comparator	->comparator for ->hit detection	The ZCD-comparator in GP30 detects ->hits in the received -> burst signal by comparing the received signal level to a given reference voltage (see also -> FHL, ->ZCD and ->hit).

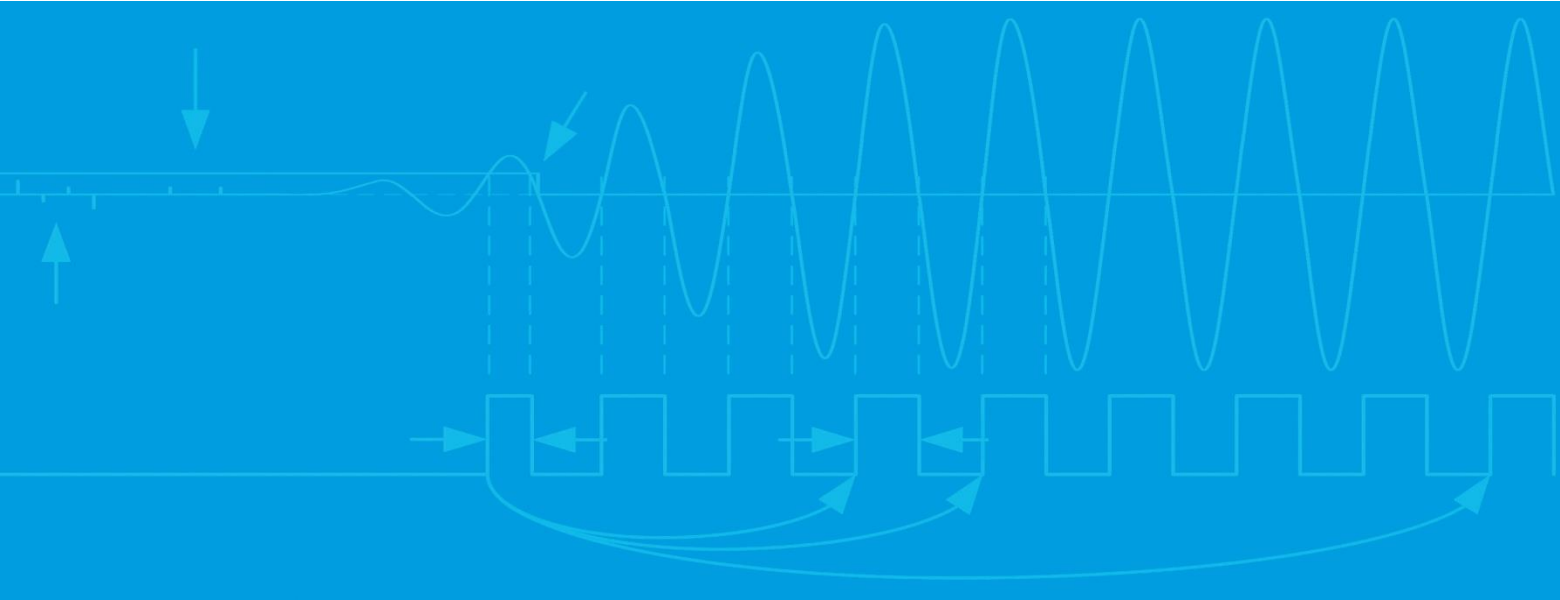
14 Miscellaneous

14.1 Bug Report

-

14.2 Last Changes

9.05.2016	First release
9.06.2016	Expanded description of PWL_ADDR on page 3-24 and 7-68;



acam-messelectronic gmbh
Friedrich-List-Straße 4
76297 Stutensee
Germany
Phone +49 7244 7419 – 0
Fax +49 7244 7419 – 29
E-Mail support.stutensee@ams.com
www.acam.de www.ams.com