Application Note

Document Number

Auto Gain & Optimization & Correction

AS7343 Evaluation Kit

v0-04 • 2021-Dec-09

Content Guide

1 Introduction 3

2 Auto-Algorithm To Optimize Parameters.................................... 5

2.1 Steps in algorithm – Search for Auto-Gain... 5
2.2 Steps in algorithm – Calculation With Auto-Gain 8
2.3 Possible Extension: Auto-TINT 10
2.4 Correction Gain 11

3 Appendix 15

4 Revision Information 21

5 Additional Documents 22

6 Legal Information......................... 23

1 Introduction

The aim of Auto Gain and Optimization is to automatically find the best (maximum possible) parameter options for the gain, to measure the maximum optimized raw value in a defined range, with which the best possible sensor results can be achieved. This can be achieved with parameter levels that are adapted to the lighting situation. The algorithm calculates the best parameters based on a sample measurement and thus always ensures a maximum adapted resolution. When using different parameter stages, e.g. gain, minimal nonlinearity occurs, i.e. due to manufacturing tolerances in semiconductor production, the stages differ slightly from each other after normalization. These deviations can be corrected individually or loosely - if the deviations lead to errors that violate the specification.

Suggestions

- ✔ Avoid measuring noise and saturation.
- ✔ Use the full dynamic sensor to increase the sensor sensitivity.
- ✔ Use the optimal parameters in the setup to get the highest accuracy.
- ✔ Adapt the integration time to the available measurement time.
- ✔ Use the Gain to adapt and optimize the counts.
- ✔ Consider the dependence of counts from the parameter setup – normalize the counts before use.
- ✔ Correct production-related nonlinearities in case of high accuracy, if needed.

Read all the technical notes regarding Gain (Gain) and Integration Time (TINT) settings, saturation, dynamic and working range, accuracy and linearities in the datasheet [1], software manual [2], and application notes [3]. Consider any limitations in using gains and integration times in automatic processes. Check beforehand whether possible nonlinearities of different amplifications by gain and integration times meet the accuracy requirements of the application or have to be corrected. Information about this is in the datasheet or must be found via tests. The software GUI in EVK demonstrates all settings of the ADC parameters and allows both automatic use and individual setting. Saturation is displayed, the retention of optimal ADC results - the counts - is not displayed. However, the higher the count, the better the resolution and accuracy, assuming no saturation. It should also be noted that saturation occurs as a function of the set integration time for Full-Scale Range (FSR = max. Counts in Figure 1).

Figure 1:
Relationship Between Setting TINT - FSR/Saturation And Max. Counts

TINT (ms)	f (kHz)	Resolution (bit)	Counts
0,0056	180,00000	1	2
0,0111	90,00000	2	4
0,0222	45,00000	3	8
0,0444	22,50000	4	16
0,0889	11,25000	5	32
0,1778	5,62500	6	64
0,3556	2,81250	7	128
0,7111	1,40625	8	256
1,4222	0,70313	9	512
2,8444	0,35156	10	1 024
5,6889	0,17578	11	2 048
11,3778	0,08789	12	4 096
22,7556	0,04395	13	8 192
45,5111	0,02197	14	16 384
91,0222	0,01099	15	32 768
182,0444	0,00549	16	65 536
364,0889	0,00275	16	65 536

In the main GUI, by default, Auto Gain is ON. Optimization of the gain is enabled by checking “Max AGAIN” and denoting the maximum gain value to be considered in the list box besides the checkbox. Similarly, optimization of Integration time is enabled by checking “Max TINT” and mentioning the maximum value to consider in the drop-down list. The Gain and time should be checked separately for optimization. Either can also be used automatically or fixed individually and separately. Disable the “Optimized Gain Detection” after taking one measurement, as it keeps changing the parameter values for measurement - to achieve an optimized raw value in each measurement if they are enabled.

The following equation describes the algorithm of the Auto-Procedure. The results of this procedure are always the adjusted parameters Gain and TINT, which should lead to optimal Raw_Counts if the situation allows it. Therefore, Raw_Counts always refer to adjusted parameters and must therefore be normalized into a form independent of parameters (Basic_Counts - see [2] or [3] chapter 2.1).

Equation 1:

$$\text{Basic\_Counts} = \frac{\text{Raw\_Counts}}{(\text{Gain} \times \text{Integration Time})}$$

The GUI always displays in the results and tables Raw_Counts, Basic_Counts and Corrected_Counts.

Equation 2:

$$Corrected\_Counts = Correction\_Values * (Basic\_Counts - Offset)$$

Corrected_Counts are calculated and corrected Basic_Counts, considering static and linear shifts from the application or sensor. Corrections are briefly described in this document to correct any nonlinearities that may occur. All the steps described here can be performed in the GUI of the EVK and are described in detail in [2].

2 Auto-Algorithm To Optimize Parameters

The steps in Auto Gain optimization are divided into sections. One part contains the Auto Gain, and the second part optimizes the derived Auto Gain. In the Auto Gain section, a gain between the maximum and minimum range is automatically calculated by the results of the test measurement. Therefore, the sensor's raw value is placed as close to the maximum as possible, without saturation.

Here only the optimization of the gain is shown as an automatic algorithm. The adjustment of the integration time is similar and is explained and shown as a simple formula.

Note that all gain and integration time settings in sum always affect all sensor channels. That is, the channel with the highest sensitivity determines the tuning of the parameters for all channels.

2.1 Steps in algorithm – Search for Auto-Gain

2. Middle Gain value is calculated from the maximum possible gain for optimization (`maxOptGain`) and taken as the currentGain.

```
currentGain = (byte)(maxOptGain / 2.0 + 0.5);
```

3. If currentGain is greater than the given maxGain (user given Maximum gain for optimization `selMaxOptGain`). Then, currentGain will be equal to the given maxGain.

```
maxGain = selMaxOptGain;

if (currentGain > maxGain)

{

    currentGain = maxGain;

}
```

4. Inside the while loop, the gain of the device is set to currentGain. Then, reads out the raw measurement and checks the saturation or noise state of rawValue measurements. If any of the conditions are true, it will enter the corresponding loop.

If the raw value is above the maximum range of the raw values (RawValueStates.Saturation) gain correction is made by reducing the gain by half of the currentGain using the algorithm below.

```
while (true)

{

    _sensor.setGain(currentGain);

    rawValueState = CheckRawValues(ref checkState, ref rawVal, ref basicVal,
                                   ref corrVal);

    measureCount++;

    if (rawValueState == RawValueStates.Saturation)

    {

        if (currentGain == 0)

        {

             break;

        }

        // Set the saturation gain flag

        if (currentGain < saturationGain)

        {

            saturationGain = currentGain

        }

        // set new gain value by reducing to half of current gain
        // by right shift method

        currentGain >>= 1;

    }
```

5. If the raw value is below the minimum range of raw values (RawValueStates. Noise), the gain correction is made by increasing the gain when it is in noise state is shown in the algorithm below.

```
else if (rawValueState == RawValueStates.Noise)

{

    // in case of low gain value use the middle between
    // max and current gain

    if (currentGain == maxGain)

    {

        break;

    }

    newGain = (byte)((maxGain + currentGain) / 2.0 + 0.5);

    if (newGain == currentGain)

    {

        newGain++;

    }

    // check if new gain value greater than saturation gain

    if (newGain >= saturationGain)

    {

         break;

    }

    // currentGain takes the newGain

    currentGain = newGain;

}
```

6. If the raw value is not above the maximum range of raw values (RawValueStates.Saturation) and below the minimum range of raw values (RawValueStates. Noise), no gain correction is made.

```
    else
    {
        break;
    }
}
```

7. If the rawValueState is still in saturation, optimization is not possible. The error is handled in the method below.

```
if (rawValueState == RawValueStates.Saturation)
{
    errorText = "Optimization not possible due to saturation";
    return errorcode;
}
```

2.2 Steps in algorithm – Calculation With Auto-Gain

As discussed in chapter 2.1, now we have a gain, which reads raw values between the Saturation and Noise. The next step is to calculate an optimized gain in such a way that the raw values are closer to the maxRawVal limit.

1. The maximum value of the current raw value is taken from the measured raw values. The range of the maximum raw value and minimum raw value is calculated based on the parameter below.

```
currentRawVal = rawVal.Max();
maxRawVal = _sensor.MaxCounts * _maximumAdcRange;
minRawVal = _sensor.MaxCounts * _minimumAdcRange;
```

Where,

`_sensor.MaxCounts` : the maximum number of counts for the current ATIME and ASTEP value.

`_maximumAdcRange: 0.90`

`_minimumAdcRange: 0.50`

2. The logarithmic value of maxRawVal divided by currentRawvalue is calculated with a base of two. This value is then rounded to the largest integer, less than or equal to that value.

```
diffGain = (int)Math.Floor(Math.Log(maxRawVal / currentRawVal, 2));
```

3. Considering an example of this calculation, the maximumRaw value is 50000, and the currentRawvalue is 30000. The idea behind the above step is to scale 30000 to reach closer to 50000. Since the gain in AS7341 is a multiple of 2, the base of calculation is 2.

30000 x 2^x = 50000

$$X = \log_2(maxRawvalue/CurrentRawValue)$$

$$X = \log_2(50000/30000)$$

4. When the currentGain added with the diffGain is greater than the maxGain (user given maximum gain for optimization), the diffGain is calculated as the difference between maxGain and currentGain.

```
if (currentGain + diffGain > maxGain)

{

    diffGain = maxGain - currentGain;

}
```

5. When the currentGain added with the diffGain is less than zero, the diffGain is calculated as the difference between zero and currentGain.

```
if (currentGain + diffGain < 0)
```

```
{
    diffGain = 0 - currentGain;
}
```

6. If the diffGain is greater than or equal to zero, the currentRawValue is incremented by a left shift of diffGain times. On the other hand, if the difference in gain is negative, the currentRawValue is decrement by a right shift of (-)diffGain times.

```
currentRawVal = (UInt16)(  diffGain >= 0
                         ? currentRawVal << diffGain
                         : currentRawVal >> -diffGain);
```

7. Current gain is added with the calculated diffGain.

```
currentGain = (byte)((int)currentGain + diffGain);
```

8. Optimized Gain is set as the Gain. Finally the raw value measurement with optimized gain is made.

```
_sensor.setGain(currentGain);

// measurement with optimized values

getMeasurementValues(ref checkState, ref rawVal, ref basicVal, ref corrVal);
```

2.3 Possible Extension: Auto-TINT

Gains can only be increased by a factor of 2. This means that the gain adaptation is very limited - to optimally fit the counts into the dynamic range up to 16 bits (Full-Scale Range = 2^16 Counts => 65535 Counts). If the control requirement with the gain is exhausted, the integration time can be used for further optimization. The background is such that the integration time can be increased in {(ATIME

+ 1) x (ASTEP + 1) x 2.78µs}[1] steps, which also lead to higher counts, higher resolution, and, in the case of an increased signal/noise ratio, to higher accuracy.

Example:

A measurement with Gain = 128 and TINT = 182ms (= FSR with max. 65535 counts) gives a result (the channel with the highest count number) of 5000 counts. Via gain optimization, it can be calculated that an increase to gain = 1024 would then bring a count number of about 40000. A gain with one step higher would lead to saturation (=80000 > 65535). However, the result of the gain optimization uses only 2/3 of the potential dynamic range (FSR). Increasing the integration time could further increase the counts, e.g. to FSR * 0.8 (80% of the sensor dynamics at defined integration time). The integration time to be extended, which must be added to the previously defined integration time, can be calculated using the following formula:

Equation 3:

$$\mathrm{TINT}_{\mathrm{increase}} = (2\text{^}\mathrm{MIN}(\mathrm{LOG}(\mathrm{TINT}/(2/720);2);16) * 0{,}8 - \mathrm{Counts}) * 2/720$$

$Counts_{expected}$ is here 40000 - the expected counts after Gain optimization.

FSR is Full Scal Range 65535 by TINT = 182ms (FSR is always depending on TINT).

2/720 is the numerical value in µs which is needed to integrate 1 Bit

0.8 is defined here as 80% of FSR and represents a defined limitation as maximum for counts.

In the example, the increase leads to adding 34.52ms to the previously set integration time of 182ms.

This leads to an optimized setup of the parameters of TINT = 217ms at Gain = 1024, which would purely mathematically result in 52428 counts, i.e. 80% of FSR = 65535. Thus the sensor dynamics are fully implemented with the set limits.

If this formula is included in the software after the gain optimization, then maximum counts are always achieved. However, there are always changed integration times, which may be included several times in the measurement time due to the SMUX configuration.

2.4 Correction Gain

An increase of the integration time is linear and does not bring any linearity errors when changing. The gains are different.

Different gain stages use different semiconductor structures on the chip and are therefore affected by production-related deviations in the series. A change in Gain as a setup of the ADC by a factor of 2, results in a factor 2 changed number of counts of the ADC. This deviation is very small, individually

[1] Both settings –ATIME and ASTEP – must not be set to “0”.

different, and is called gain error or nonlinearity. All channels and stages are affected differently. This leads to changes of the channels to each other, which is wrongly interpreted as a spectral shift in further calculations. As long as a gain is used in the measurement series or cycles, the deviations do not have any effect on the measurement, unless calibration and measurement use different gain stages. If automatic and optimized gain levels are used, then gain correction is recommended.

Depending on the test results and accuracy requirements, gain correction can be performed individually or batch-wise for the sensors, for all channels individually, or on average per gain stage. Gain tests can be made with the EVK. A stable light source is necessary for this (better to use more than one and calculate the average of multiple light sources). The tests can be done with the GUI using a script (shown in Figure 2) which makes the initialization, controls the protocol file, sets alternative gains, and starts the measurement for each gain.

The result of this process is a protocol (log, see Figure 3) file for the tests including the Setup, Raw_Counts, Basic_Counts, temperature, etc. These results can be normalized at one gain for all channels to see the variations (Figure 4), and to get one correction factor for each channel and gain. Use this matrix as a Lookup Table (LUT as matrix) to correct the Basic_Counts according to the used gain and channel. An alternative can be to use the averaged correction factors for all channels of a gain (LUT as a vector).

Figure 2:
Tracer File For Gain Test

```
/// Trace file for Gain Test

Autogain off

AutoTint on

max_Tint = 2048

startLog

gain 0,5

readsamples 1

gain 1

readsamples 1

…

gain 1024

readsamples 1

gain 2048

readsamples 1

stopLog

saveLog

clearlog
```

Figure 3:
Protocol File From Gain Test With Basic_Counts

Gain [x]	Basic F1 (400nm)	Basic F2 (424nm)	Basic FZ (450nm)	Basic F3 (473nm)	Basic F4 (514nm)	Basic FY (555nm)	Basic F5 (547nm)	Basic FXL (595nm)	Basic F6 (635nm)	Basic F7 (685nm)	Basic F8 (745nm)	Basic VIS	Basic NIR
0.5	0.00352	0.01367	0.06172	0.04219	0.06602	0.09766	0.02813	0.08086	0.07813	0.03633	0.00430	0.11602	0.00703
1	0.00352	0.01367	0.06172	0.04238	0.06602	0.09785	0.02813	0.08086	0.07813	0.03633	0.00430	0.11582	0.00703
2	0.00352	0.01367	0.06172	0.04258	0.06621	0.09795	0.02822	0.08105	0.07822	0.03633	0.00449	0.11602	0.00732
4	0.00352	0.01382	0.06201	0.04272	0.06646	0.09829	0.02832	0.08135	0.07852	0.03657	0.00454	0.11650	0.00762
8	0.00344	0.01335	0.05991	0.04138	0.06426	0.09514	0.02732	0.07856	0.07571	0.03545	0.00444	0.11250	0.00757
16	0.00344	0.01337	0.05991	0.04142	0.06429	0.09522	0.02733	0.07861	0.07576	0.03550	0.00450	0.11256	0.00778
32	0.00346	0.01340	0.06003	0.04154	0.06448	0.09548	0.02742	0.07884	0.07601	0.03567	0.00457	0.11287	0.00800
64	0.00349	0.01348	0.06032	0.04185	0.06491	0.09613	0.02762	0.07936	0.07657	0.03597	0.00466	0.11357	0.00832
128	0.00348	0.01346	0.06017	0.04176	0.06498	0.09595	0.02760	0.07945	0.07652	0.03594	0.00472	0.11307	0.00859
256	0.00355	0.01370	0.06122	0.04268	0.06628	0.09803	0.02815	0.08111	0.07806	0.03673	0.00485	0.11495	0.00898
512	0.00367	0.01419	0.06333	0.04453	0.06899	0.10208	0.02925	0.08447	0.08111	0.03827	0.00506	0.11836	0.00947
1024	0.00350	0.01364	0.06096	0.04120	0.06650	0.09455	0.02795	0.08160	0.07758	0.03528	0.00488	0.11519	0.00913
2048	0.00349	0.01394	0.06229	0.04151	0.06831	0.09428	0.02819	0.08500	0.07838	0.03451	0.00500	0.11571	0.00937

Figure 4:
Normalized Basic_Counts Form Gain Test

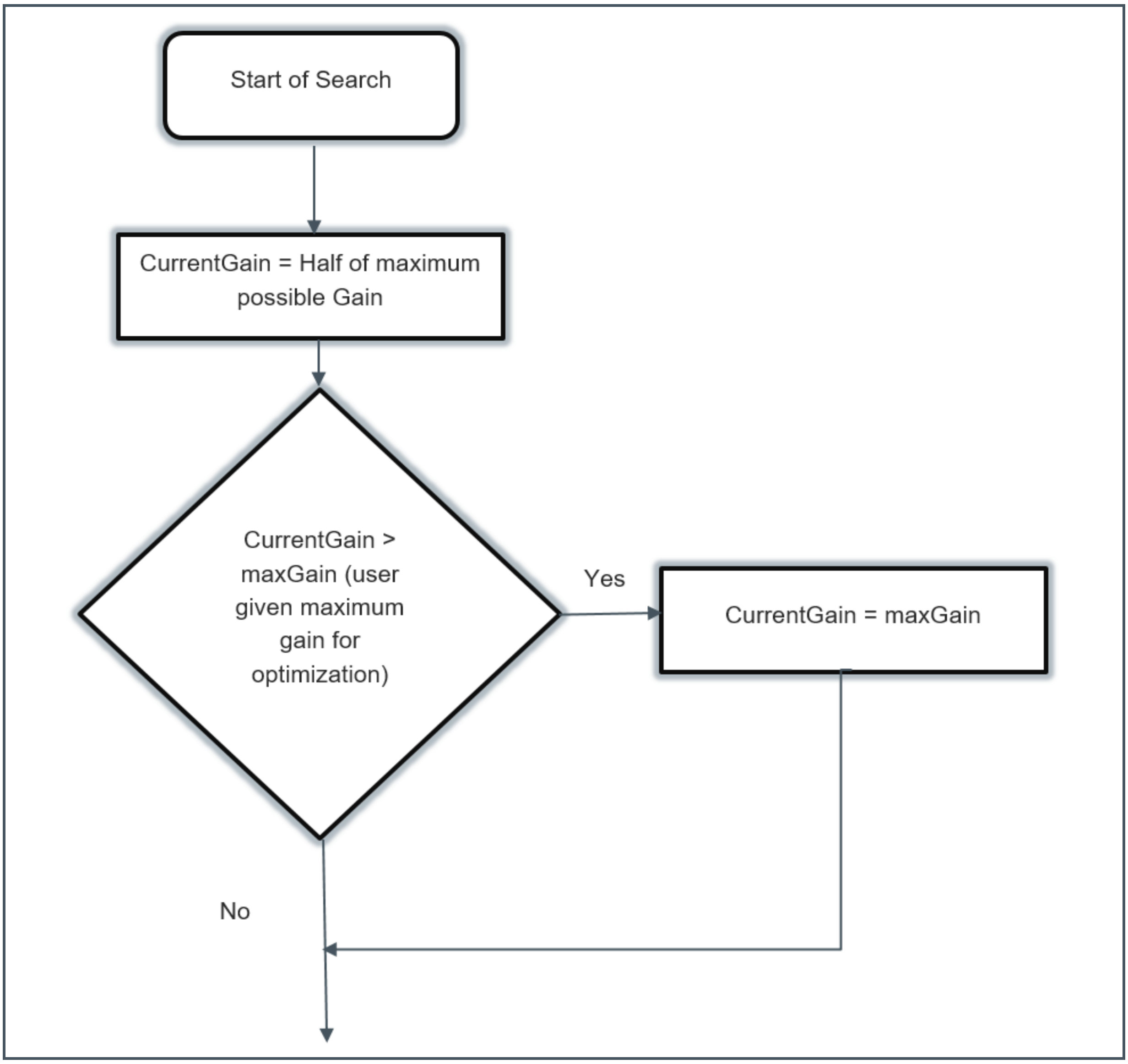
Gain [x]	Basic F1 (400nm)	Basic F2 (424nm)	Basic FZ (450nm)	Basic F3 (473nm)	Basic F4 (514nm)	Basic FY (555nm)	Basic F5 (547nm)	Basic FXL (595nm)	Basic F6 (635nm)	Basic F7 (685nm)	Basic F8 (745nm)	Basic VIS	Basic NIR
0.5	1.00890	1.01402	1.02318	1.00815	1.01696	1.01591	1.01847	1.01884	1.02029	1.01009	0.92210	1.02157	0.84548
1	1.00890	1.01402	1.02318	1.01283	1.01696	1.01793	1.01847	1.01884	1.02029	1.01009	0.92210	1.01986	0.84548
2	1.00890	1.01402	1.02318	1.01749	1.01998	1.01895	1.02202	1.02129	1.02156	1.01009	0.96395	1.02157	0.88071
4	1.00890	1.02485	1.02803	1.02098	1.02374	1.02251	1.02553	1.02499	1.02539	1.01688	0.97446	1.02588	0.91595
8	0.98766	0.99043	0.99322	0.98891	0.98989	0.98974	0.98928	0.98992	0.98873	0.98565	0.95343	0.99062	0.91005
16	0.98766	0.99140	0.99322	0.98977	0.99045	0.99051	0.98972	0.99054	0.98937	0.98702	0.96652	0.99116	0.93506
32	0.99311	0.99407	0.99514	0.99269	0.99336	0.99324	0.99283	0.99338	0.99263	0.99177	0.98112	0.99384	0.96224
64	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
128	0.99971	0.99800	0.99755	0.99802	1.00097	0.99818	0.99957	1.00113	0.99939	0.99942	1.01330	0.99561	1.03295
256	1.01750	1.01587	1.01495	1.01993	1.02104	1.01982	1.01941	1.02196	1.01938	1.02135	1.04077	1.01220	1.08033
512	1.05280	1.05221	1.04992	1.06404	1.06281	1.06193	1.05932	1.06435	1.05924	1.06403	1.08648	1.04220	1.13865
1024	1.00545	1.01157	1.01051	0.98454	1.02445	0.98354	1.01206	1.02820	1.01314	0.98081	1.04678	1.01434	1.09764
2048	1.00115	1.03419	1.03271	0.99195	1.05236	0.98082	1.02068	1.07095	1.02365	0.95954	1.07232	1.01890	1.12626

In the gain test, probably not all gains can be served with one integration time. It makes sense to adapt the integration time to the gain. Before using Auto-Tint, a test should verify that the use of variable integration times does not result in nonlinearity, since these would otherwise be included in the gain correction.

3 Appendix

Search for Gain between Saturation and Noise

Figure 5:
Search for Gain between Saturation and Noise



While loop for searching Auto Gain

Figure 6:
While loop for searching Auto Gain – Part I

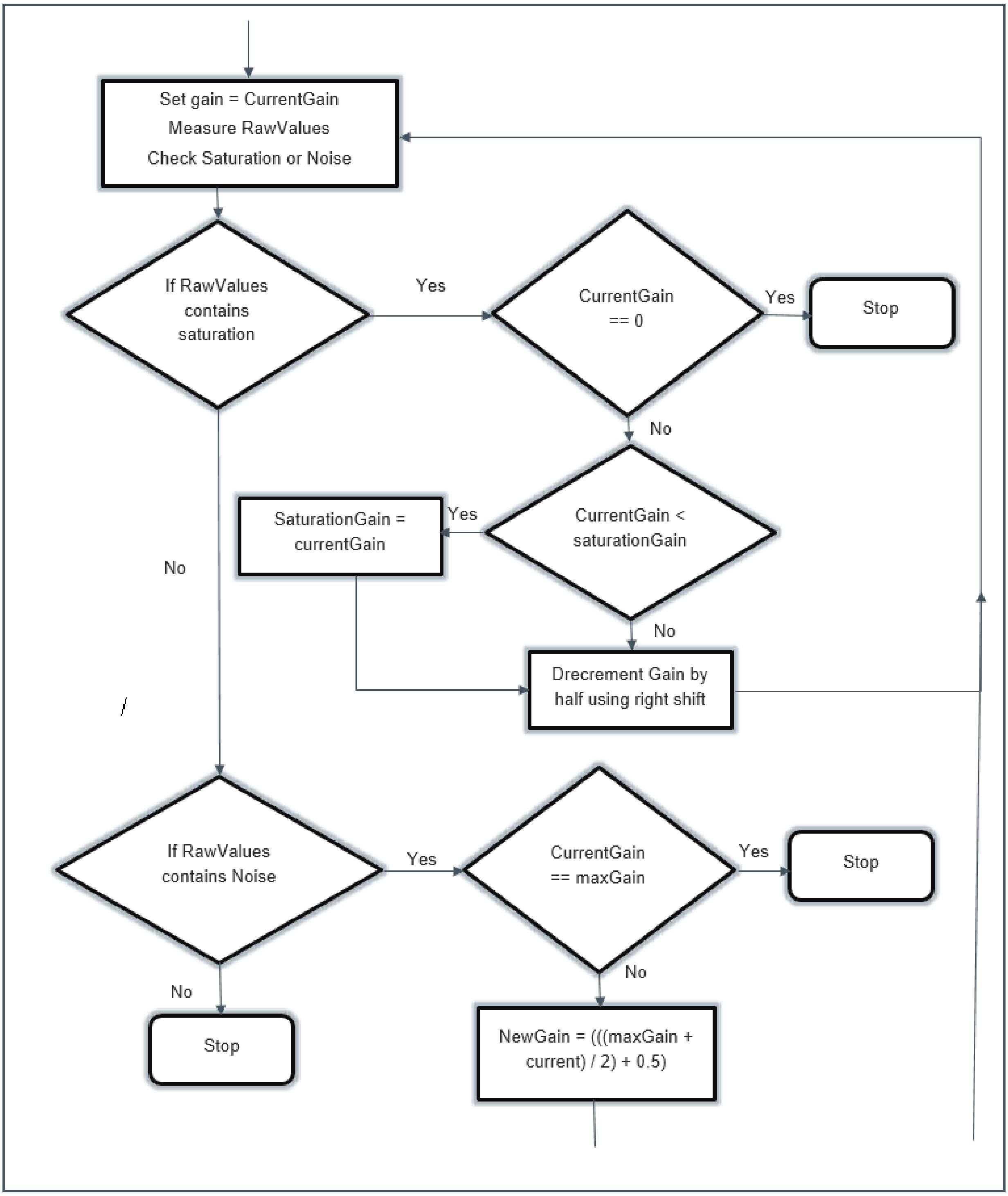
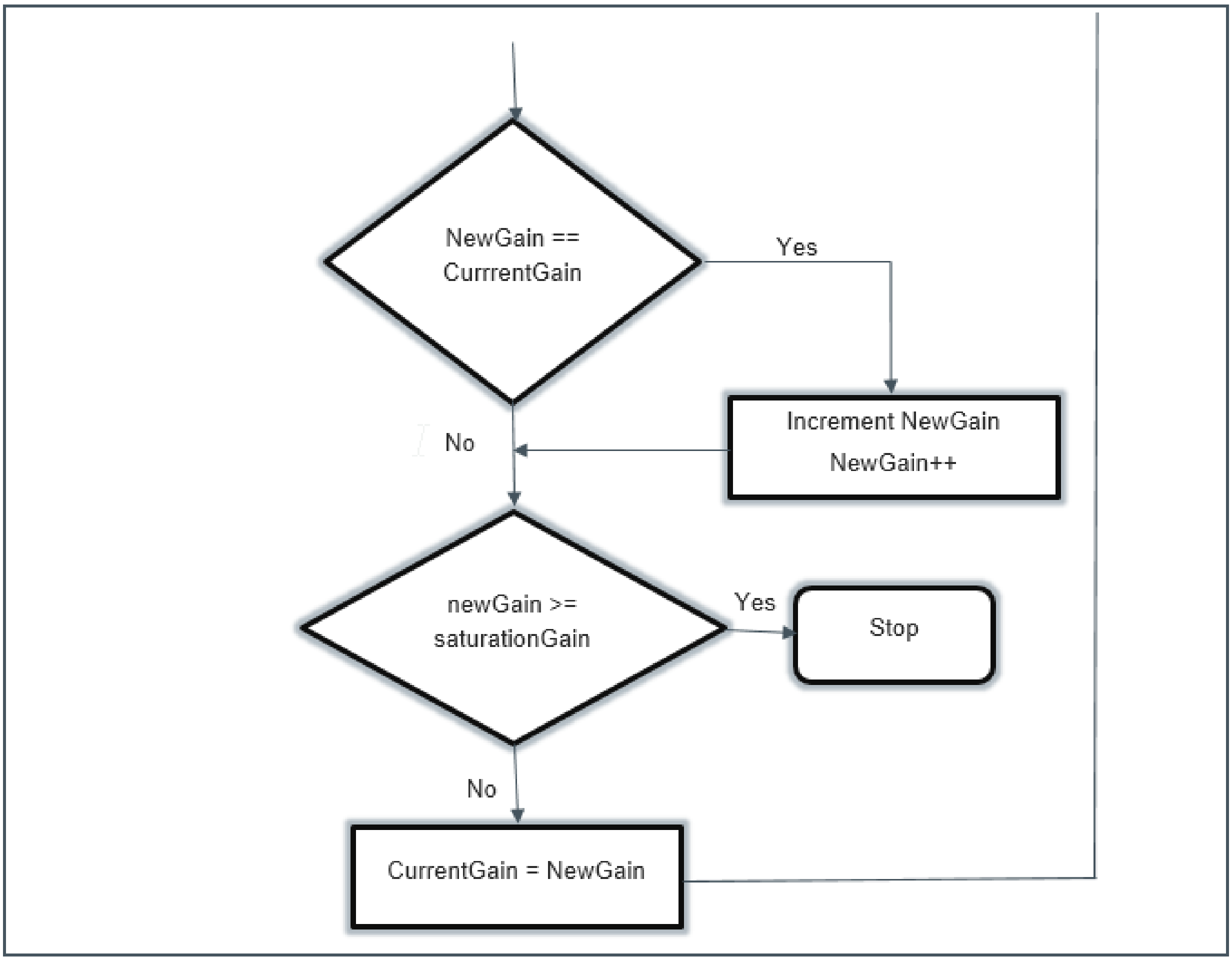
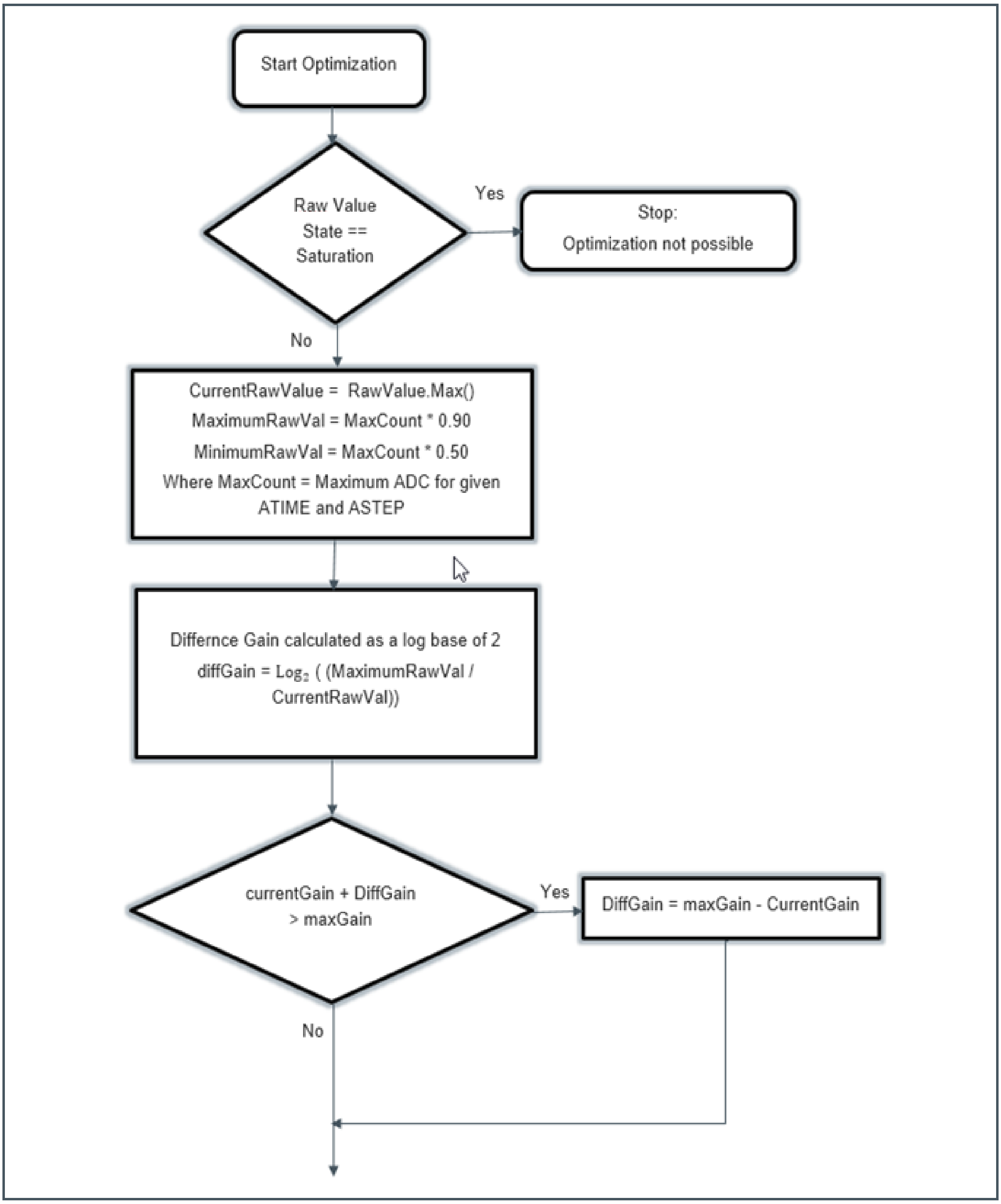


Figure 7:
While loop for searching Auto Gain – Part II



Calculation of optimized gain closer to maximum limit

Figure 8:
Calculation of optimized gain closer to maximum limit – Part I



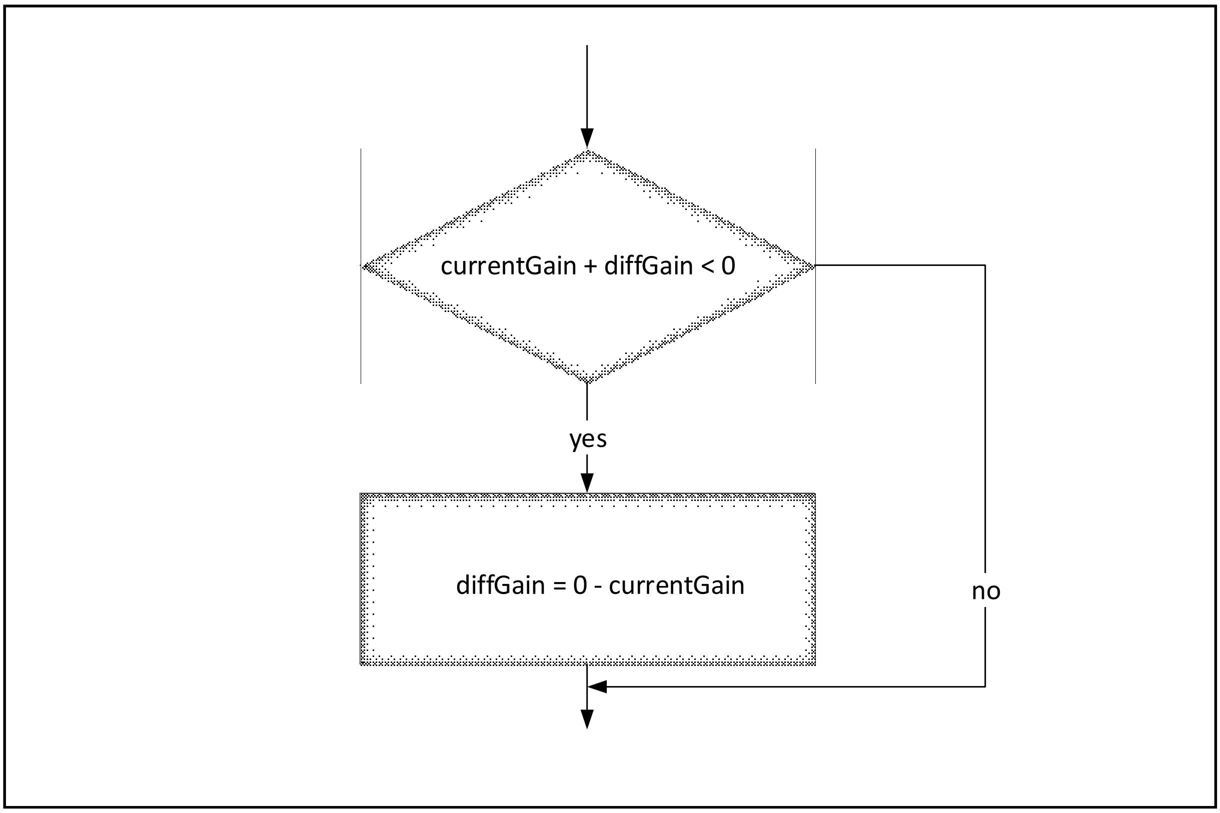
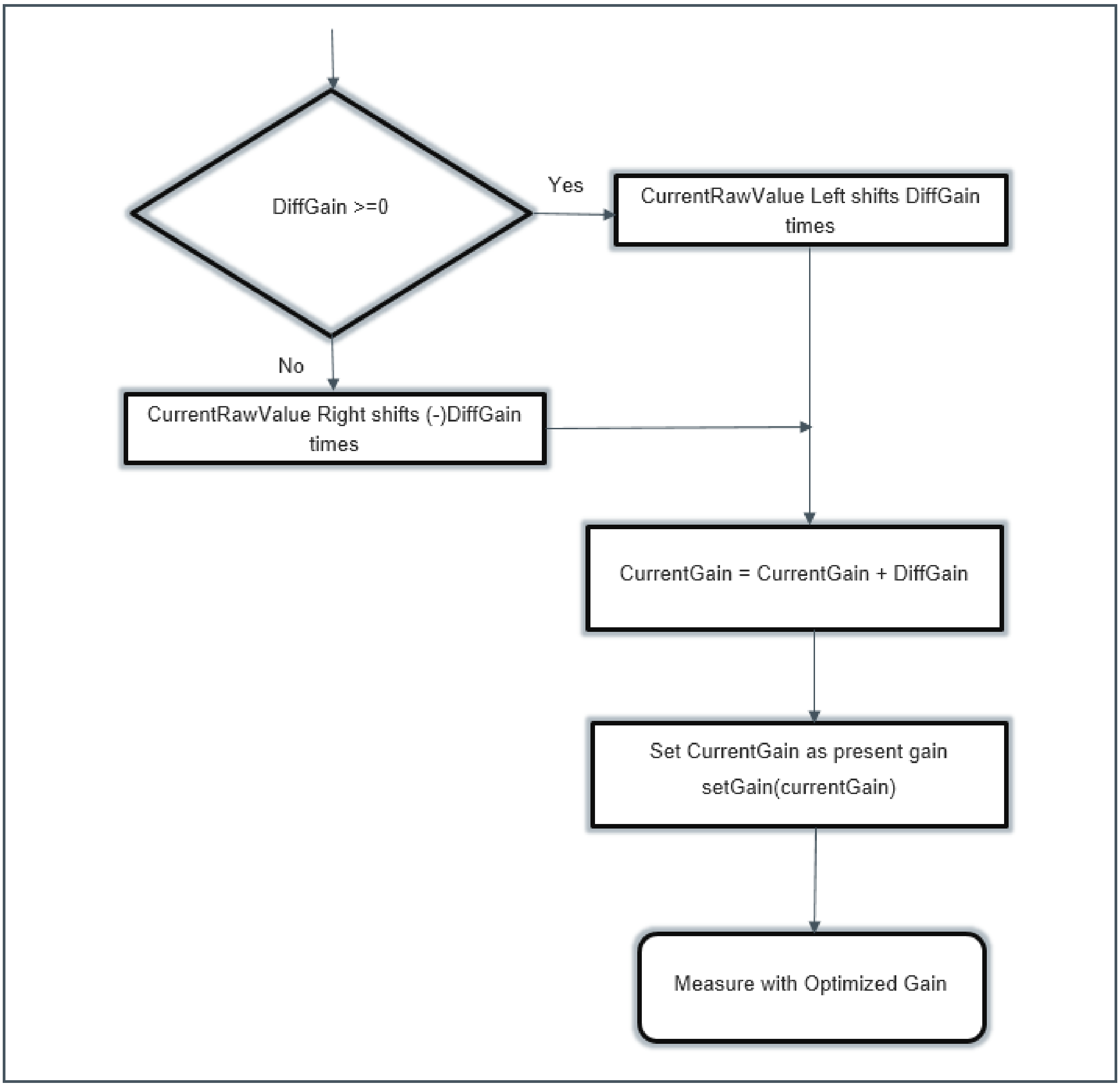

currentGain + diffGain < 0
yes
diffGain = 0 - currentGain
no

Figure 9:
Calculation of optimized gain closer to maximum limit – Part II



4 Revision Information

Changes from previous version to current revision v0-04	Page
Equation 3: Calculation TINT increasing	11

- Page and figure numbers for the previous version may differ from page and figure numbers in the current revision.
- Correction of typographical errors is not explicitly mentioned.

5 Additional Documents

The following list include a selection of available documents with more technical details for the AS7343 sensor and its Evaluation Kit. This list is not fixed and it is constantly changing. Ask us for new details.

For further information, please refer to the following documents:

1. ams AG, AS7343 14-Channel Spectral Sensor, datasheet.
2. ams AG, *AS7343 EVK (AS7343_UG001009_1-01),* user guide.
3. ams AG, AS7343 Spectral Sensor Calibration Methods, application note

6 Legal Information

RoHS Compliant & ams Green Statement

RoHS Compliant: The term RoHS compliant means that ams AG products fully comply with current RoHS directives. Our semiconductor products do not contain any chemicals for all 6 substance categories plus additional 4 substance categories (per amendment EU 2015/863), including the requirement that lead not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, RoHS compliant products are suitable for use in specified lead-free processes.

ams Green (RoHS compliant and no Sb/Br/Cl): ams Green defines that in addition to RoHS compliance, our products are free of Bromine (Br) and Antimony (Sb) based flame retardants (Br or Sb do not exceed 0.1% by weight in homogeneous material) and do not contain Chlorine (Cl not exceed 0.1% by weight in homogeneous material).

Important Information: The information provided in this statement represents ams AG knowledge and belief as of the date that it is provided. ams AG bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. ams AG has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. ams AG and ams AG suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

Headquarters

ams AG

Tobelbader Strasse 30

8141 Premstaetten

Austria, Europe

Tel: +43 (0) 3136 500 0

Please visit our website at www.ams.com

Buy our products or get free samples online at www.ams.com/Products

Technical Support is available at www.ams.com/Technical-Support

Provide feedback about this document at www.ams.com/Document-Feedback

For sales offices, distributors and representatives go to www.ams.com/Contact

For further information and requests, e-mail us at ams_sales@ams.com