RAA2P4520

Introduction

This manual provides information for software developers on how to communicate, configure, and program the RAA2P4520 for various applications.

Contents

Figures

Tables

1. Overview

The RAA2P4520 inductive position sensor is used in high- and low-speed automotive, industrial, medical, and consumer applications. It has an internal analog frontend and ADC for data acquisition of the sine and cosine input signals. These signals are fed into a Cordic to compute the absolute angle information. A 16-point linearization function allows to increase the accuracy.

Supported RAA2P4520 output interfaces are:

- SENT (up to 14bit)
- UART (14bit)

Supported programming interfaces are:

- 1-Wire UART (Default)
- 2-Wire UART (After interface is changed in device configuration)

1.1 Related Information

For the hardware specification, electrical characteristics, product details, such as features, pin descriptions, functionality, and circuit descriptions of the RAA2P4520, see the RAA2P4520 Datasheet.

Additional information about implementation in safety-related systems as per ISO26262 can be found in the *RAA2P4520 Functional Safety Manual*.

1.2 Quick-Start / Default Configuration

The RAA2P4520 comes with a default configuration for the transmitter bias current, automatic gain control, and output mode. Basic diagnostic alarms are enabled. The sensor outputs the position relative to the target position after power-up over the default output interface.

Default configuration:

Supply Mode:	5V
Programming:	1-Wire Uart at startup (Baud rate auto adaption, 9600 default)
Output Interface:	SENT, SPC
AGC Mode:	ON
UART Address:	0hex (default)
TX bias current:	496µA

1.3 Programming Procedure

The programming parameters depend on the application requirements. Most parameters do not change for production parts. Some parameters can be programmed individually to compensate production tolerances. A programming procedure usually includes following steps:

1. General Configuration (Select the correct supply voltage, Output mode, etc. mainly in system configuration registers, Adr 0x001C, 0x01E)
2. Transmitter Configuration (TX Oscillator Current Setting, Adr 0x0016)
3. Receiver Configuration (can be programmed individually if needed)
   a. Static Gain Adjustment in AGC configuration register if AGC is switched off (angle1_agc, Adr 0x0020; angle2_agc, Adr 0x0026)
   b. Coil Offset Compensation (angle1_coil_offs, Adr 0x0024; angle2_coil_offs, Adr 0x002A)

4. Sensor Calibration / Linearization
   a. Zero Position Register to configure the position where the output is zero (zero_pos_angle1, Adr 0x003C; zero_pos_angle2, Adr 0x003E)
   b. Linearization mode configuration, Angle hysteresis configuration (data_path_ctrl0, Adr 0x002E)
   c. Linearization coefficients (Adr 0x0040…0x006E)
5. Diagnostics Configuration
   a. Diagnostic Mask Enable to configure which diagnostic is signalled. All diagnostic functions are active but only enabled functions will signal error state at the output. (irq_en, Adr 0x000C…0x0014).
   b. Alarm Levels need to be configured as according to the application requirements (for example, magnitude static check or magnitude ripple check (Adr 0x0032, 0x002C)).

Notes:

- To avoid possible error signaling during a calibration procedure, disable the diagnostic signaling (irq_en, Adr 0x000C…0x0014) during calibration and enable it at the end of the programming procedure.
- Disable the AGC and configure a fixed gain during configuration of offset compensation to avoid distortion of measurements during a compensation procedure.

1.4 RAA2P4520 Block Diagram

The following block diagrams show the main components of the RAA2P4520 for the available output interfaces. Note that pin assignment depends on the IC configuration. For an overview, see the “Input and Output Pin Configuration summary” in the RAA2P4520 Datasheet.

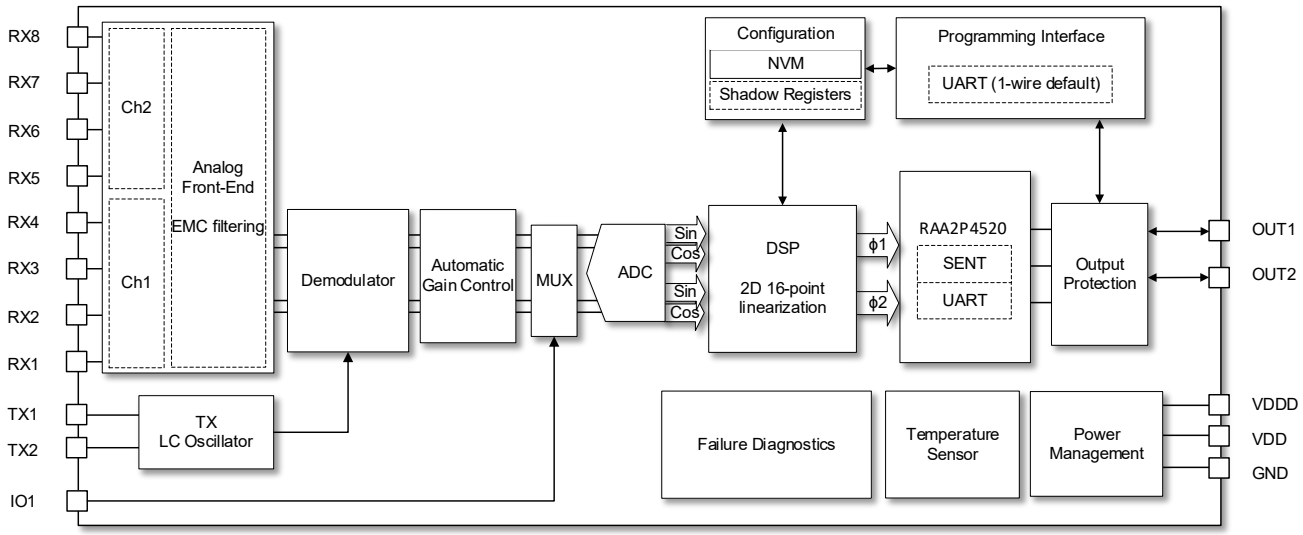

Figure 1. RAA2P4520 Block Diagram

1.5 Programming Mode

To access the NVM registers and enable write access to shadow registers, the device must be set into programming mode.

There are two different programming interfaces available:

- Programming over OUT1 pin using 1-wire UART: To enter the programming mode, a programming entry command must be sent during power-up over the programming interface (1-Wire UART).
- Programming over 2-wire UART is possible after the device configuration was changed to 2-wire UART. Programming mode is enabled by sending the programming entry command after startup.

1.5.1 Programming over OUT1 pin: 1-Wire UART

The 1-wire programming interface is used to program the device for remote applications using SENT interface.

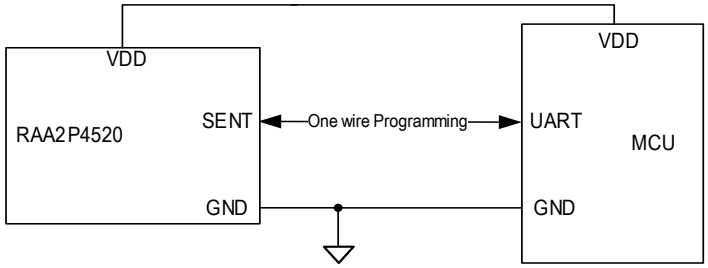

Figure 2. Output Interfaces

1.5.1.1 Programming Entry Enabled (Default)

Per default, the programming entry window is enabled and allows to enter programming mode over the programming interface (1-Wire UART). If no programming entry command is received, the device enters normal operation after power-up time.

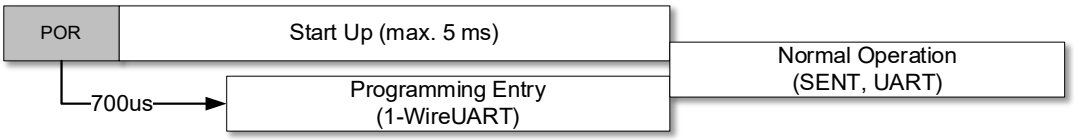

Figure 3. Programming Window Enabled (Default)

1.5.1.2 UART Programming Request Frame

During the programming entry window, a correct UART programming request frame (the address byte of the programming entry UART frame) opens a programming window of 100ms. If in these 100 ms programming window the programming mode entry command is received correctly, the device will enter programming mode.

Note: In all UART modes, the bit order is MSB to LSB.

Op.	Byte	Start	D7	D6	D5	D4	D3	D2	D1	D0	Stop
write	#1	0	1	Acc(0x0)				DevAddr[1:0]		0	1
write	#2	0	0x0				CRC_A[3:0] = 0101				1
write	#3	0	RegisterAddress[15:8] = 0x06								1
write	#4	0	RegisterAddress[7:0] =0x06								1
write	#5	0	0x7			CRC[4:0] = 11101					1
write	#6	0	WriteData[15:8] = 0x00								1
write	#7	0	WriteData[7:0] = 0xCA								1
write	#8	0	0x7			CRC[4:0] = 01111					1
wait	#9	GAP Time									

UART Request frame: Byte #1 must start before 3.5 ms

Figure 4. UART Programming Request Frame

1.5.1.3 Programming Entry Successful

Figure 5 shows a successful programming entry sequence. After the programming entry sequence is successfully completed reading and writing on NVM and system registers is enabled. In case the programming entry is not successful, the device enters normal operation after the programming window.

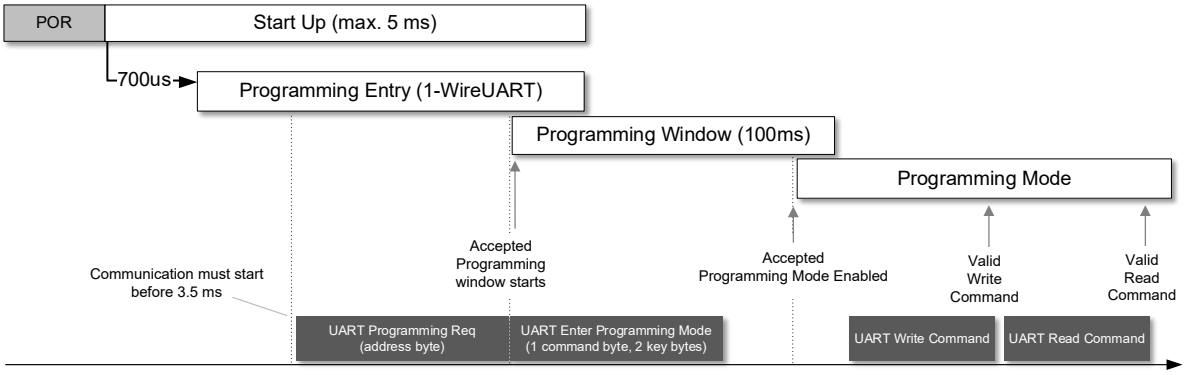

Figure 5. Programming Entry Successful

Note: In case the programming entry should not be successful, the device enters normal operation after the programming window.

1.5.1.4 Enter Programming Mode Sequence

To enter programming mode, write 0xCA into register 0x0606. The programming mode is needed to enabled read and write access to NVM and system registers. Then read from register 0x0606, expected data is 0xCA. The sequence is shown in the following figure.

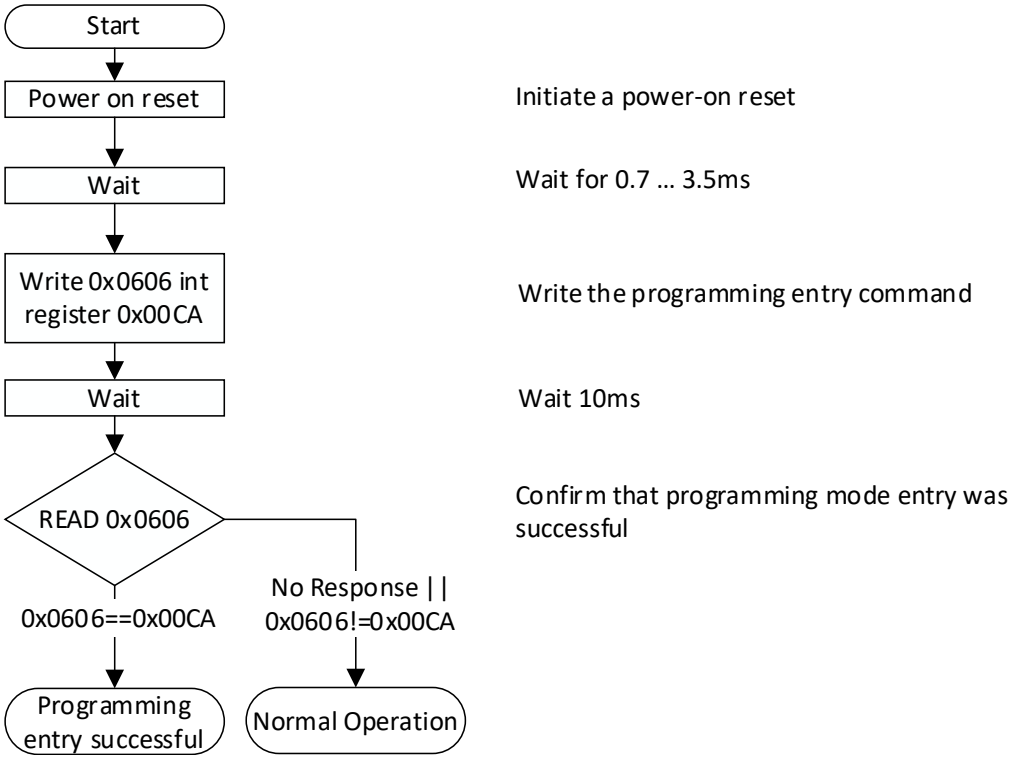

Figure 6. Enter Programming Mode Sequence

A plot of a programming mode entry sequence is shown in the following figure.

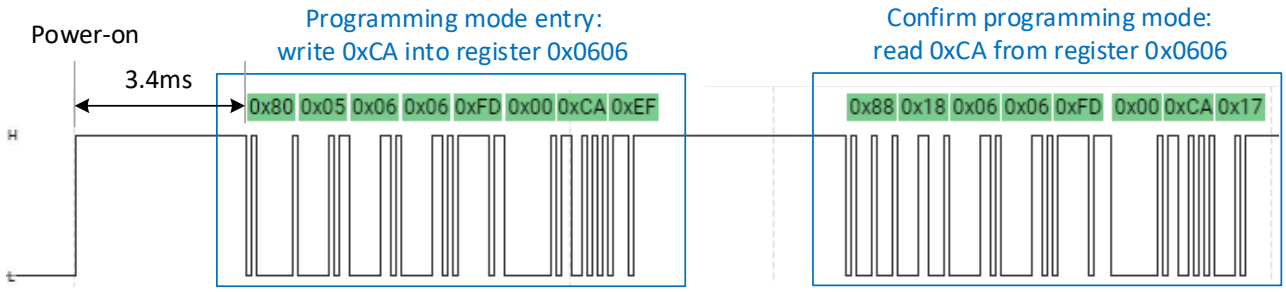

Figure 7. Plot of a 1-Wire UART Programming Entry Sequence

Note: The IC exits programming mode via power-on reset.

1.5.1.5 Programming Entry Disabled

After successful programming of the device, the programming entry window can be disabled to reduce the startup time to 3ms.

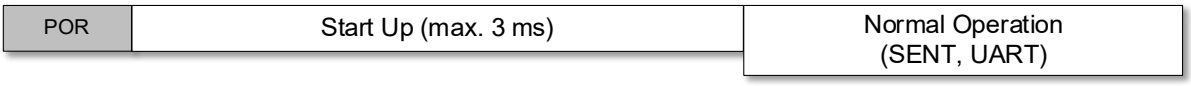

Figure 8. Programming Entry Disabled

1.5.1.6 Uart crc Check

The CRC check for Uart communication is always activated. The device will respond to command frames including correct CRC data only.

The polynomial for the data (CRC) is x^5+x^2+1. The seed is 0x1F. The notation is 0x5.

The polynomial for address (CRC_A) is x^4+x+1. The seed is 0xF. The notation is 0x3.

Note: A generic CRC code example is shown in Appendix A.1.

1.5.1.7 Uart Register Write Access

This mode writes one register per frame. All bytes are write-accessible, and data is sent from master to the device.

Note: In all modes, the bit order is MSB to LSB.

Legend:

- Acc = Access type: must be 0x0 for register write.
- DevAddr = Device address: default = 0x0. If several sensors are connected in parallel, it defines their slave address (0x1..0x3).
- Register Address = The register to be written to
- Write Data = The data to be written to register address
- CRC = Cyclic redundancy check

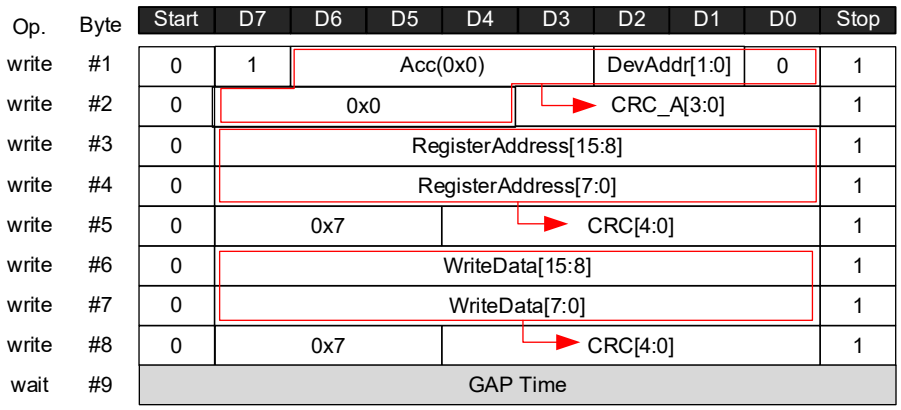

Figure 9. Uart Register Write Access

1.5.1.8 Uart Register Read Access

This mode reads one register per frame. The first 5 bytes are write-accessible, data is sent from master to the device, bytes #6 - #8 are read access, data is read from the chip.

Note: In all modes, the bit order is MSB to LSB.

Legend:

- Acc = Access type: must be 0x1 for register read.
- DevAddr = Device address: default = If several sensors are connected in parallel, it defines their slave address (0x1..0x3).
- Register Address = The register to be read from
- Write Data = The contents of the data read from Register address
- CRC = Cyclic redundancy check.
- Status = 1 bit status information of the sensor (0 = normal operation , 1 = error)
- RC = Rolling counter, a counter that increments with each frame and wraps around 0 when overflowing
- Wait = 1 bit wait time for master before receiving data (in pseudo-differential mode only)

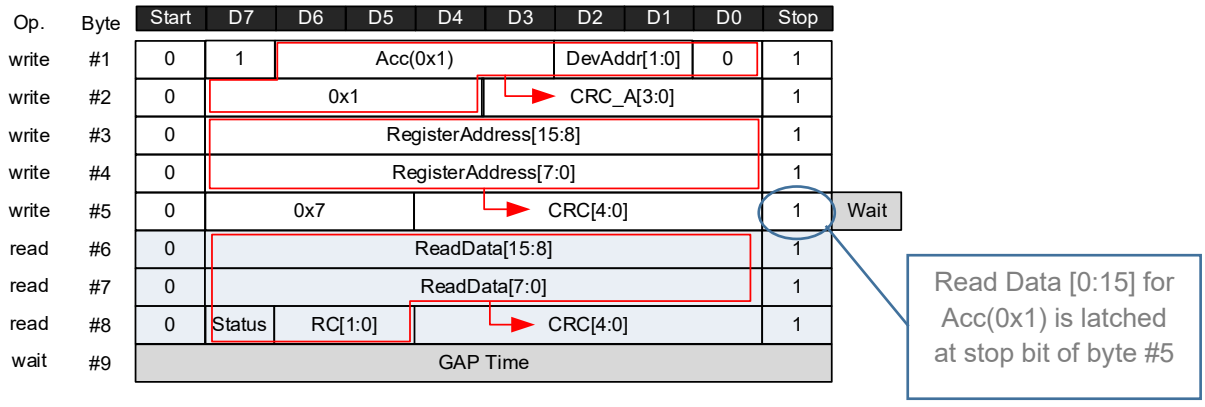

Figure 10. Uart Register Read Access

The rolling counter (RC) is incremented by 1 at every successful start of execution of Frame #5. After reset it starts counting from 0 and wraps up to 0 once it counts to its maximum value.

The status bit (Status) is generated as a summary information for all related to UART Controller status flags.

2. Sensor Configuration

The RAA2P4520 is configured with a default memory content during production test. Depending on the sensor requirements, the configuration must be changed to enable the desired functionality.

The RAA2P Evaluation Software plugin for RICBox allows together with a RAA2P-comboard to configure the sensor IC in laboratory environment. It can be used to reset the device to default configuration, to program the configuration of an existing sensor, or to completely configure the IC according to individual requirements.

Alternatively, python code examples are available to support the sensor configuration.

Refer to the download section on the secure portal:

https://www.renesas.com/en/secure/raa2p3xxx-secure-content#downloads

The sensor configuration is divided into two parts. The application-specific configuration is performed during the sensor design validation to configure the product according to the application requirements for the desired sensor output. The sensor calibration is used to achieve best sensing performance.

For a detailed description of all registers and bitfields, see the Product Register Map:

https://www.renesas.com/en/secure/raa2p3xxx-secure-content#documents

2.1 Application-Specific Configuration

First, the sensor is generally configured for the application to output the expected sensor signal for the application. This is configuration is usually done once for a specific sensor design.

2.1.1 General Device Configuration

The general device configuration like supply voltage mode, output mode, angle hysteresis, etc. is done in following registers:

- **io_cfg0, io_cfg1** – These register are used to configure the UART interface and sensor address.
- **sent_cfg0, sent_cfg1, sent_cfg2** – To configure the SENT or SPC.
- **sys_cfg0** – In this register it is possible to configure the Fault Detection Time Interval (FDTI) configuration, Coil configuration, select the supply voltage, apply a memory lock and select the desired output interface.
- **sys_cfg1** – Here the high resolution mode can be configured and enabled.
- **data_path_ctrl0** – This register allows to apply an angle hysteresis, enable the output calibration (linearization function) and enable and configure the propagation delay compensation.
- **data_path_ctrl1** – Using this register it is possible to configure the automatic input signal amplitude mismatch and offset compensation, change the direction by inverting the slope of the output signal, apply low pass filters for coil1.
- **data_path_ctrl2** – The settings in this register are used to configure the automatic input signal amplitude mismatch and offset compensation, change the direction by inverting the slope of the output signal, apply low pass filters for coil2. Furthermore the magnitude ripple check can be enabled and configured.

2.1.2 Automatic Gain Control

As default, the automatic gain control is active. The AGC uses the magnitude signal of the internal CORDIC block to control the signal gain as defined in the product datasheet.

In case the AGC is disabled, the angle1_init_gain and angle2_init_gain register must be configured for the desired gain in registers angle1_agc and angle2_agc.

It is not recommended to disable the AGC in normal operation. After power-up, the AGC starts to work from the initial gain value.

Note: It is recommended to disable the AGC and configure a static gain during sensor calibration procedures. After sensor calibration, enable the AGC again.

2.1.3 TX Current Setup

The transmitter current for the LC Tank as shown in Figure 11 must be configured based on the application requirements. Reducing the transmitter bias current decreases the transmitter amplitude and lowers the total current consumption. The behavior is not linear and depends on the configuration of the LC oscillator.

Note: The oscillator requires sufficient bias current to ensure stable oscillation. Refer to the following equation to estimate the bias current setting. The estimated value can be adjusted to meet application requirements.

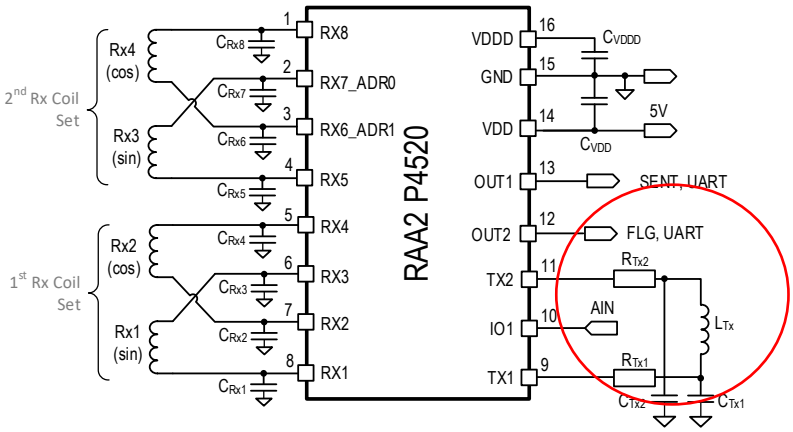

Figure 11. Transmitter Current Configuration for LC Tank

$$I_{BIAS\_SingleTX} = VDD / (35 \times L\_Tx \times Q \times F) \quad \text{or} \quad I_{BIAS\_RedundantTX} = VDD / (50 \times L\_Tx \times Q \times F)$$

Where:

- VDD = Supply voltage in Volt
- L_TX = Inductance of transmitter coil in Henrys
- F = Transmitter oscillator frequency in Hz
- Q = Quality factor of the Transmitter coil, it is calculated according to the following formula:

$$Q = \frac{1}{R_{L\_TX}} \sqrt{\frac{L\_Tx}{C}}$$

Where:

- R_{L_TX} = Resistance of the Transmitter Coil
- C = Capacitance of the Transmitter resonator ($C = C_{Tx1}/2$)

For example:

If L = 6µH, R = 1Ohm, C = 345pF (such that the transmitter oscillates at F = 3.5MHz) and VDD = 5V

According to the formula above:

Q = 132

I_{BIAS} = 51.5µA

The calculated bias current is programmed in register lc_osc_cfg 0x0016.

The bit configuration for the bias current base value and multiplication factor can be selected from Table 1. An example is highlighted by blue, where 26 dec for the base value and 1 dec for the multiplication factor is shown.

Table 1. Ic_current Selection

Ic_curr_trim[7:6]: Multiplication factor		Ic_curr_trim[5:0]: Base value in dec															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0:1x	0	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5	5.5	6	6.5	7	7.5
	1:4x	0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
	2:16x	0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120
	3:64x	0	32	64	96	128	160	192	224	256	288	320	352	384	416	448	480
		16	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
	0:1x	8	8.5	9	9.5	10	10.5	11	11.5	12	12.5	13	13.5	14	14.5	15	15.5
	1:4x	32	34	36	38	40	42	44	46	48	50	52	54	56	58	60	62
	2:16x	128	136	144	152	160	168	176	184	192	200	208	216	224	232	240	248
	3:64x	512	544	576	608	640	672	704	736	768	800	832	864	896	928	960	992
		32	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**
	0:1x	16	16.5	17	17.5	18	18.5	19	19.5	20	20.5	21	21.5	22	22.5	23	23.5
	1:4x	64	66	68	70	72	74	76	78	80	82	84	86	88	90	92	94
	2:16x	256	264	272	280	288	296	304	312	320	328	336	344	352	360	368	376
	3:64x	1024	1056	1088	1120	1152	1184	1216	1248	1280	1312	1344	1376	1408	1440	1472	1504
		48	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**	**60**	**61**	**62**	**63**
	0:1x	24	24.5	25	25.5	26	26.5	27	27.5	28	28.5	29	29.5	30	30.5	31	31.5
	1:4x	96	98	100	102	104	106	108	110	112	114	116	118	120	122	124	126
	2:16x	384	392	400	408	416	424	432	440	448	456	464	472	480	488	496	504
	3:64x	1536	1568	1600	1632	1664	1696	1728	1760	1792	1824	1856	1888	1920	1952	1984	2016

2.1.4 Delay Compensation and Filter Configuration

The device filters must be configured with the recommended configuration shown as follows:

angle_hyst (0x2E[15:13] ; hys_limit) = 4 (HYST_4x16LSB)

delay_comp_dis (0x2E[7]) = 0 (ENABLE_D_COMP)

velocity_filter_depth (0x2E[6:5]) = 2 (M_DEPTH_4)

velocity_filter_dis (0x2E[4]) = 0 (ENABLE_VE_FIL)

lpf_dly_comp_sc_en (0x2E[3]) = 1 (ENABLE_LPF_DLY_COMP_SC)

lpf_dly_comp_en (0x30[14]) = 1 (ENABLE_LPF_DLY_COMP)

angle1_filter_en (0x30[6] lpf_pc_dis) = 0 (ENABLE_LPF_PC)

angle2_filter_en (0x32[6] lpf_sc_dis) = 0 (ENABLE_LPF_SC)

interp_data_rdy_sel (0x32[12])= 1 (TYP_DPATH_DONE)

The LPF and velocity filter settings are selected according to the maximum sensor speed in the application using Table 2, Table 3, and Table 4 shown as follows:

angle1_filter_depth (0x0030[5:3] ; lpf_pc_depth) – depending on speed: Refer to Table 2, Table 3

angle2_filter_depth (0x0032[5:3] ; lpf_sc_depth) – depending on speed: Refer to Table 3

velocity_m_depth (0x2E[2:0]) – depending on LPF, refer to Table 4

Note: Delay compensation as available on angle1 based on the speed of angle1. For angle2, the delay compensation is based on the same speed of angle1.

2.1.4.1 Low Pass Filter Configuration

The low pass filter configuration depth is selected on the maximum speed in the application. Depending on whether the device is configured for single coil operation or dual coil configuration, the LPF depth is selected based on the following tables.

Table 2. Low Pass Filter Configuration for Single Coil Operation

Speed	angle1_filter_depth for 3us DataRate
<70000 rpm electrical	<= depth_7
<150000 rpm electrical	<= depth_6
< 300000 rpm electrical	<= depth_5
< 600000 rpm electrical	<= depth 4

Table 3. Low Pass Filter Configuration for Dual Coil Operation

Speed	angle1_filter_depth for 6us DataRate
< 30000 rpm electrical	<=depth_7
< 70000 rpm electrical	<= depth_6
< 150000 rpm electrical	<= depth_5
< 300000 rpm electrical	<= depth 4
< 600000 rpm electrical	<= depth 3

2.1.4.2 Velocity Filter Configuration

The velocity filter configuration is selected according to LPF depth setting as shown in the following table. The velocity_m_depth must be selected according to following criteria: (2^lpf)/(2^m_depth) <= 1.

Table 4. Velocity Filter Configuration

angle1_filter_depth			
code		value	velocity_m_depth
0	DEPTH_1_LPF_PC	2	>=2
1	DEPTH_2_LPF_PC	4	>=4
2	DEPTH_3_LPF_PC	8	>=8
3	DEPTH_4_LPF_PC	16	>=16
4	DEPTH_5_LPF_PC	32	32
5	DEPTH_6_LPF_PC	64	32
6	DEPTH_7_LPF_PC	128	32

2.1.5 Diagnostic Configuration

The diagnostic configuration depends on the application requirements.

All diagnostic functions are permanently active and the state of each diagnostic function can be read from registers irq_sts0, irq_sts1, irq_sts2, irq_sts3, irq_sts4.

The configuration in the interrupt enable registers irq_en0, irq_en1, irq_en2, irq_en3, irq_en4 defines which diagnostic function can trigger diagnostic state at the output interface.

The recommended interrupt enable configuration for the specific application can be generated using the Diagnostic Configuration Tool.

2.2 Sensor Calibration

The following steps are used to calibrate the sensor for better performance. Depending on accuracy requirements they may be repeated for each sensor during sensor production or after sensor assembly in the system.

The sensor IC supports static offset and amplitude correction for both coils and dynamic offset an amplitude correction for the second coil.

The secondary coil is used as reference in high-resolution mode only and does not support automatic compensation. The accuracy of the high-resolution output is determined by the first coil.

The automatic compensation can be enabled in data_path_ctrl1 register 0x0030[11:10].

To use a static amplitude correction, it must be enabled in data_path_ctrl1 register 0x0030[11:10] for the first coil and data_path_ctrl2 register 0x0032[8] for the second coil.

2.2.1 Coil Offset Compensation

The coil offset compensation is used to reduce coil offset coming from not ideal sensing coils due to pcb production variation. Coil offset compensation is applied in the analog signal path and is mandatory to reduce possible signal offset before the A/D conversion. The offset compensation is performed by using the angle1_coil_offs 0x0024 and angle2_coil_offs 0x002A registers.

Note: The coil offset compensation is fundamental for good sensor performance. In case the coil offset changes for example after sensor assembly into a metallic housing, the new offset must be compensated again.

2.2.2 Zero Position

The zero position can be configured in registers zero_pos_angle1 and zero_pos_angle2. They are used to set the sensor output to zero after sensor assembly. To reduce the effect from assembly tolerances to the sensor output, calibrate the zero position for each sensor after assembly.

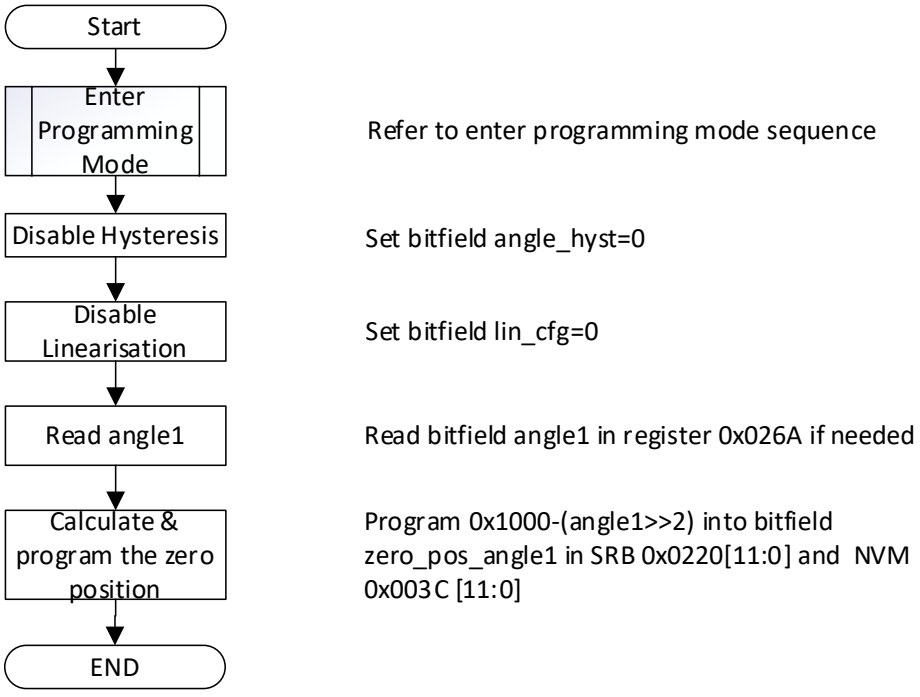

Figure 12. Set Zero Position Procedure

2.2.3 Linearization

The zero position of the angle output can be programmed in bitfield zero_pos_angle1 and zero_pos_angle2.

The output slopes (direction) are configured in registers sys_cfg0 and sys_cfg1.

The RAA2 P45xx offers a very flexible linearization feature to further improve the accuracy of the sensor. The linearization is applied in the digital domain, following the angle calculation, see Figure 1.

The linearization is performed with 16-bits resolution over 360°(el.).

Up to 16 freely programmable linearization points can be positioned within a grid of 0.088° in X (position) and Y (expected output). If two sensor coils are used, the available 16 points are used at both angles. (for example, 8 points each).

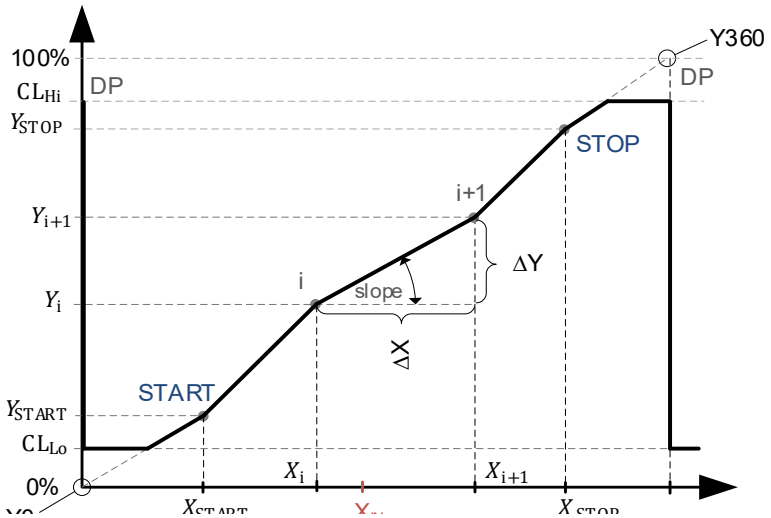

Figure 13. Linearization Transfer Function Parameters

Table 5. Linearization Parameter Settings

Parameter	Description	Programming Options	Resolution
DP	Discontinuity point, Zero position transition from 0°/360°	0° to <360° el.	12 bits 0.088°el. / LSB
X_{Start}	Mechanical start position, first linearization point		
Y_{Start}	Expected output at XStart, first linearization point		
X_i	Mechanical position of linearization point (I = 1 to 16, including start and stop)		
Y_i	Expected output at linearization point (I = 1 to 16 including start and stop)		
X_{Stop}	Mechanical end position, last linearization point		
Y_{Stop}	Expected output at XStop, last linearization point		
CL_{Hi}	Output Clamping level, high	0% to 100% VDD	12 bits (VDD / 4096) / LSB
CL_{Lo}	Output Clamping level, low		
Y0	Position at DP, start value at X=0°	0° / 360° el.	1 bit
Y360	Position at DP, stop value at X=360°	0° / 360° el.	1 bit

Linearization coefficients can be calculated using the Linearization Wizard in the Evaluation Software (RAA2P3xxx-4xxx-4.0.0.0-x64-Combined_Setup.zip) or by using the Linearization Demo Python Script (linearization_demo.zip), and are available for download in the in the RAA2P4XXX Secure Portal.

https://www.renesas.com/en/secure/raa2p4xxx-secure-content#downloads

The procedure for linearization is shown below in Figure 14.

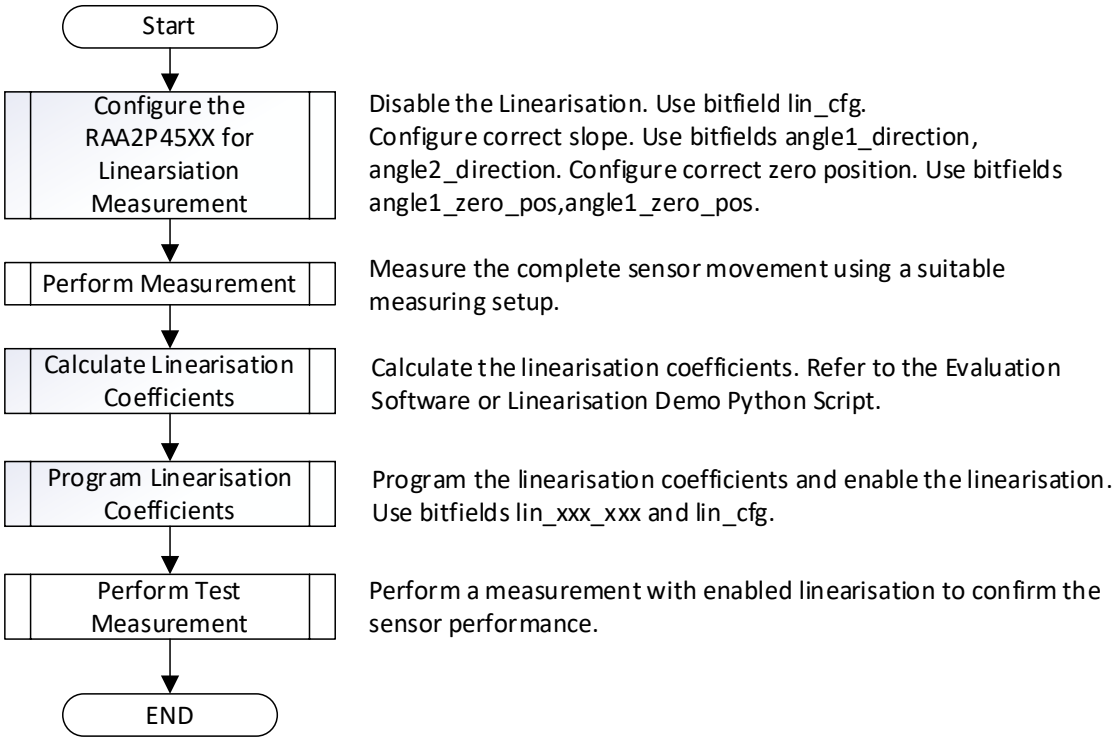

Figure 14. Linearization Procedure

3. Memory Architecture

The internal memory is split into Non-volatile Memory, Shadow registers (SR), and Data registers (DR) as described in Table 6.

Table 6. Non-volatile Memory Architecture

Peripheral	Address range	CRC	Normal Mode	Programming Mode
NVM_CUSTOMER	0x0000 … 0x008E	usrCrc (NVM)	No	RW[1]
CORE_SYS_CUST Shadow Register	0x0200 … 0x0268	usrCrc (SR)	RO	RW[1]
CORE_SYS_CUST Data Register	0x026A … 0x02A8	No	RO	RO
AFE_SYS_CUST Shadow Register	0x0400 … 0x0404	usrCrc (SR)	RO [1]	RW [1]
AFE_SYS_CUST Data Register	0x0406	No	RO [1]	RO [1]
IF_SYS Shadow Register	0x0600 … 0x0604	usrCrc (SR)	RO [1]	RW [1]
IF_SYS Data Register	0x0606	No	RW [1]	RW [1]
SENT Shadow Register	0x0800 … 0x0804	usrCrc (SR)	RO [1]	RW [1]
ERR_HNDLR Shadow Register	0x0C00 … 0x0C08	usrCrc (SR)	RO [1] [1]	RW [1]
ERR_HNDLR Data Register	0x0C0A … 0x0C12	No	RO [1]	RO [1]
ERR_HNDLR Clear Registers	0x0C14 … 0x0C1C	No	WO (Write 1 to Clear)	WO (Write 1 to Clear)
CFG_CRC CRC Shadow Register	0x0E00	No	RO [1]	RW [1]
CFG_CRC Calculated Checksum	0x0E02	No	RO [1]	RO

1. Depends on cyber lock

3.1 Configuration Protection (NVM and Shadow Registers) with CRC Checksum

Two separate CRC checksums are implemented over customer and factory programmed sub-address spaces to guarantee that up to 2-bits error can be detected (CRC polynomial needs to cover hamming distance of 3 bits). When CRC is calculated it is matched against etalon value programmed by customer (for customer sub-space) or ATE (for internal sub-space).

Note: After writing to an NVM register, the CRC for the user NVM register space must be calculated and programmed in register 0x008E: usrCrc. If the value in usrCrc is not correct, the IC enters a diagnostic state during power-up. Communication interfaces available only in normal operation mode after power-up cannot be used.

3.1.1 Initial NVM CRC Calculation

As a part of the device's initialization process, NVM contents are read out and written to the shadow register of all peripherals. In parallel, CRC calculation is performed and resulting checksums are matched against etalon values.

3.1.2 Periodic SR CRC Calculation

In order to detect errors in the shadow registers with hamming distance of 3 bits, periodically a read of all SR is triggered, CRCs are calculated, and matched against etalon values.

3.1.3 Polynomial

The following 12-bits polynomial is used for the error detection with hamming distance of 3 bits. The crc must be calculated over the customer NVM space.

CRC_poly = x12 ^ x11 ^ x3 ^ x2 ^ x ^ 1

Seed = 12'hFFF

3.1.4 Overview of NVM Registers

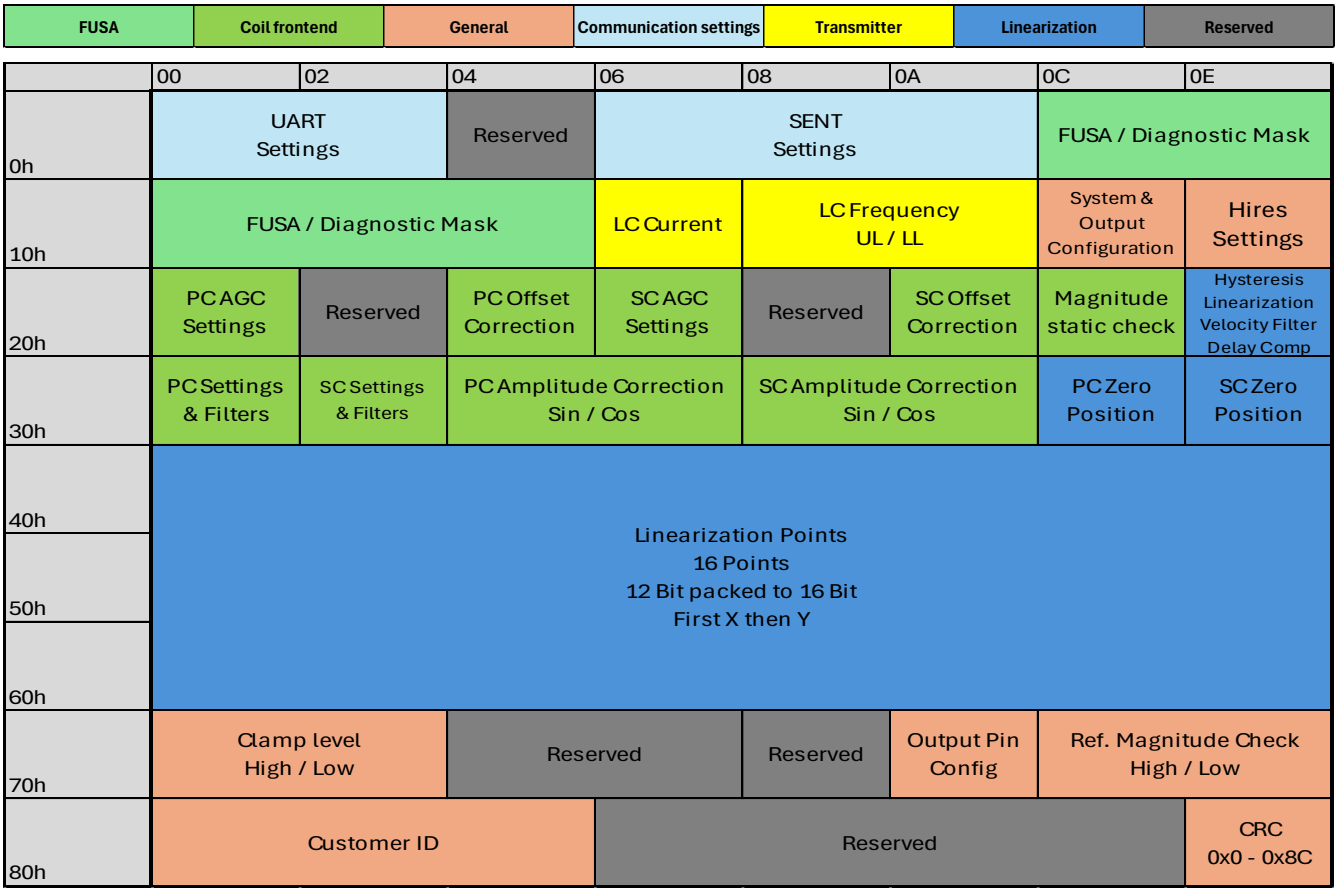

Figure 15. NVM Register Overview

Note: Read rows as the first hexadecimal digits of the address and the columns as the last digit (for example, the TX current register address = 16h).

3.1.5 Summary of Registers

All registers of the RAA2P4520 are summarized in Table 7. For a detailed description of all registers and bitfields, contact Renesas.

Notes:

- Some registers are modified by internal processes.
- After power-on-reset (POR), most of the NVM content is mirrored into the corresponding SRB address range within the start-up time window.
- **Do not change registers, register fields or bits that are not listed or marked as “Reserved”, “Do not use” or “Not supported”.**
- After writing to a NVM register, a programming time of minimum **6ms** is required before a new command can be sent.
- When writing to a register, always change the desired registers, register fields or bits that are explained in the register map only. Always read a register content, modify intended bits only, and write the modified register content to the device. Internal configuration bits are not listed in the register map and must not be changed.
- After writing to a register, read back the content and check for correct register content.

Table 7. Register Summary

Peripheral (Base Address)	Register Name	Register Address	Register Access	Reset Value
NVM_CUSTOMER (0x0000)	io_cfg0	0x0000	read-write	NA
	io_cfg1	0x0002	read-write	NA
	sent_cfg0	0x0006	read-write	NA
	sent_cfg1	0x0008	read-write	NA
	sent_cfg2	0x000A	read-write	NA
	irq_en0	0x000C	read-write	NA
	irq_en1	0x000E	read-write	NA
	irq_en2	0x0010	read-write	NA
	irq_en3	0x0012	read-write	NA
	irq_en4	0x0014	read-write	NA
	lc_osc_cfg	0x0016	read-write	NA
	lc_osc_frq_ll	0x0018	read-write	NA
	lc_osc_frq_ul	0x001A	read-write	NA
	sys_cfg0	0x001C	read-write	NA
	sys_cfg1	0x001E	read-write	NA
	angle1_agc	0x0020	read-write	NA
	angle1_coil_offs	0x0024	read-write	NA
	angle2_agc	0x0026	read-write	NA
	angle2_coil_offs	0x002A	read-write	NA
	agc_msc_cfg	0x002C	read-write	NA
	data_path_ctrl0	0x002E	read-write	NA
	data_path_ctrl1	0x0030	read-write	NA
	data_path_ctrl2	0x0032	read-write	NA
	ampl_corr_sin_angle1	0x0034	read-write	NA
	ampl_corr_cos_angle1	0x0036	read-write	NA
	ampl_corr_sin_angle2	0x0038	read-write	NA
	ampl_corr_cos_angle2	0x003A	read-write	NA
	zero_pos_angle1	0x003C	read-write	NA
	zero_pos_angle2	0x003E	read-write	NA
	lin_x01_x02	0x0040	read-write	NA
	lin_x02_x03	0x0042	read-write	NA
	lin_x03_x04	0x0044	read-write	NA
	lin_x05_x06	0x0046	read-write	NA
	lin_x06_x07	0x0048	read-write	NA
	lin_x07_x08	0x004A	read-write	NA
	lin_x09_x10	0x004C	read-write	NA
	lin_x10_x11	0x004E	read-write	NA
	lin_x11_x12	0x0050	read-write	NA
	lin_x13_x14	0x0052	read-write	NA
	lin_x14_x15	0x0054	read-write	NA
	lin_x15_x16	0x0056	read-write	NA
	lin_y01_y02	0x0058	read-write	NA
	lin_y02_y03	0x005A	read-write	NA
	lin_y03_y04	0x005C	read-write	NA
	lin_y05_y06	0x005E	read-write	NA
	lin_y06_y07	0x0060	read-write	NA
	lin_y07_y08	0x0062	read-write	NA
	lin_y09_y10	0x0064	read-write	NA

Peripheral (Base Address)	Register Name	Register Address	Register Access	Reset Value
	lin_y10_y11	0x0066	read-write	NA
	lin_y11_y12	0x0068	read-write	NA
	lin_y13_y14	0x006A	read-write	NA
	lin_y14_y15	0x006C	read-write	NA
	lin_y15_y16	0x006E	read-write	NA
	clamp_high	0x0070	read-write	NA
	clamp_low	0x0072	read-write	NA
	out_cfg	0x007A	read-write	NA
	afe_ref_magn_ul	0x007C	read-write	NA
	afe_ref_magn_ll	0x007E	read-write	NA
	cust_id0	0x0080	read-write	NA
	cust_id1	0x0082	read-write	NA
	cust_id2	0x0084	read-write	NA
	usrCrc	0x008E	read-write	NA
CORE_SYS_CUST (0x0200)	sys_cfg	0x0200	read-write	0x0000
	sys_cfg1	0x0202	read-write	0x0000
	angle1_agc	0x0204	read-write	0x0000
	angle1_offs	0x0208	read-write	0x0000
	angle2_agc	0x020A	read-write	0x0000
	angle2_offs	0x020E	read-write	0x0000
	agc_msc_cfg	0x0210	read-write	0x0000
	data_path_ctrl0	0x0212	read-write	0x0000
	data_path_ctrl1	0x0214	read-write	0x0000
	data_path_ctrl2	0x0216	read-write	0x0000
	ampl_corr_sin_angle1	0x0218	read-write	0x0000
	ampl_corr_cos_angle1	0x021A	read-write	0x0000
	ampl_corr_sin_angle2	0x021C	read-write	0x0000
	ampl_corr_cos_angle2	0x021E	read-write	0x0000
	clamp_low	0x0258	read-only	0x0000
	out_cfg	0x025E	read-write	0x0000
	afe_ref_magn_ul	0x0260	read-write	0x0000
	afe_ref_magn_ll	0x0262	read-write	0x0000
	cust_id0	0x0264	read-write	0x0000
	cust_id1	0x0266	read-write	0x0000
	cust_id2	0x0268	read-write	0x0000
	angle1	0x026A	read-only	0x0000
	angle2	0x026C	read-only	0x0000
	angle1_dlycmp	0x026E	read-only	0x0000
	angle1_magn	0x0270	read-only	0x0000
	angle2_magn	0x0272	read-only	0x0000
	ref_magn	0x0274	read-only	0x0000
	angle1_sin	0x027A	read-only	0x0000
	angle1_cos	0x027C	read-only	0x0000
	angle2_sin	0x027E	read-only	0x0000
	angle2_cos	0x0280	read-only	0x0000
	angle1_turns	0x0282	read-only	0x0000
	angle2_turns	0x0284	read-only	0x0000
	chip_temp	0x0286	read-only	0x0000
	angle_diff	0x0288	read-only	0x0000

Peripheral (Base Address)	Register Name	Register Address	Register Access	Reset Value
	hires_lsb	0x028A	read-only	0x0000
	hires_msb	0x028C	read-only	0x0000
	angle1_gain	0x028E	read-only	0x0000
	angle1_offs	0x0290	read-only	0x0000
	angle2_gain	0x0292	read-only	0x0000
	angle2_offs	0x0294	read-only	0x0000
	prod_id	0x02A0	read-only	0x0000
AFE_SYS_CUST (0x0400)	lc_osc_cfg	0x0400	read-write	0x0000
	lc_osc_frq_ll	0x0402	read-write	0x0000
	lc_osc_frq_ul	0x0404	read-write	0x0000
	lc_frequency	0x0406	read-only	0x0000
IF_SYS (0x0600)	io_cfg0	0x0600	read-write	0x0000
	io_cfg1	0x0602	read-write	0x0000
	prog_mode	0x0606	read-write	0x0000
SENT (0x0800)	sent_cfg0	0x0800	read-write	0x0000
	sent_cfg1	0x0802	read-write	0x0000
	sent_cfg2	0x0804	read-write	0x0000
ERR_HNDLR (0x0C00)	irq_en0	0x0C00	read-write	0x0000
	irq_en1	0x0C02	read-write	0x0000
	irq_en2	0x0C04	read-write	0x0000
	irq_en3	0x0C06	read-write	0x0000
	irq_en4	0x0C08	read-write	0x0000
	irq_sts0	0x0C0A	read-only	0x0000
	irq_sts1	0x0C0C	read-only	0x0000
	irq_sts2	0x0C0E	read-only	0x0000
	irq_sts3	0x0C10	read-only	0x0000
	irq_sts4	0x0C12	read-only	0x0000
	irq_clr0	0x0C14	write-only	0x0000
	irq_clr1	0x0C16	write-only	0x0000
	irq_clr2	0x0C18	write-only	0x0000
	irq_clr3	0x0C1A	write-only	0x0000
	irq_clr4	0x0C1C	write-only	0x0000
CFG_CRC (0x0E00)	crc_et1	0x0E00	read-write	0x0000
	crc_rslt1	0x0E02	read-only	0x0000

4. Glossary

Term	Description
AGC	Automatic Gain Control
FTP	Few Times Programmable
FSM	Finite State Machine
I^2C	Inter-Integrated Circuit; serial two-wire data bus
LSB	Least Significant Bit
MRC	Magnitude Ripple Check
MSC	Magnitude Static Check
MSB	Most Significant Bit
MSN	Most Significant Nibble
NVM	Nonvolatile Memory
POR	Power-On Reset
RFU	Reserved for Future Use
*_PC	Primary Coil
*_SC	Secondary Coil
SRB	Shadow Register Bank

5. Revision History

Revision	Date	Description
1.01	May 15, 2026	▪ Added “Delay Compensation and Filter Configuration” ▪ Updated link in section 1.1.
1.00	Dec 18, 2025	Initial release

A. Appendix

A.1 CRC Code Example

A generic code example for CRC calculation is shown below.

```
 * @param data The data word to calculate the CRC from
 * @param polynomial The CRC polynonmial.
 * @param seed The initial CRC seed value.
 * @param dataWidth The width of the data word.
 * @param crcWidth The width of the CRC polynomial.
 * @return The CRC value.
 */
inline uint32_t R_CRC(uint32_t data, const uint8_t polynomial, const uint8_t seed,
const uint8_t dataWidth, const uint8_t crcWidth)
{
    const uint8_t crcTbl[2] = { 0x0, polynomial-1 };
    uint8_t lfsr = seed;
    for (int16_t i = dataWidth - 1; i >= 0; i--)
    {
        uint8_t inv = GET_BIT(lfsr, crcWidth-1) ^ GET_BIT(data, i);
        lfsr <<= 1;
        lfsr ^= crcTbl[inv];
        SET_BIT(lfsr, 0, inv);
    }
    return lfsr & ((1 << crcWidth) - 1);
}
```

Figure 16. Generic CRC Code Example

A.2 Python CRC Code Example

A python code example for CRC calculation is shown below.

```
CRC Calculator
# Version 1.1
# Date: 14.01.2025

class CRC:
    def __init__(self, polynomial: int, seed: int, data_width: int, crc_width:
int):
        """CRC Calculation for IPS products

        :param polynomial: int -> polynomial defined by IC
        :param seed: int -> seed defined by IC
        :param data_width: int -> bit with of data used in CRC calculation
        :param crc_width: int -> bit with of crc that will be returned
        """

        self.Polynomial: int = polynomial - 1
        self.Seed: int = seed
        self.Datawidth: int = data_width
        self.CRCwidth: int = crc_width

    def calc_crc(self, data: int) -> int:
        """Calculate CRC

        :param data: int -> data that must be CRC calculated
        :return CRC: int -> Requested CRC
        """
        lsfr = self.Seed
        for i in range(self.Datawidth - 1, -1, -1):
            inv = self.Get_bit(lsfr, self.CRCwidth - 1) ^ self.Get_bit(data, i)
            lsfr = lsfr << 1
            lsfr = lsfr ^ self.check_set_poly(inv)
            lsfr = self.Set_bit(lsfr, 0, inv)
        return lsfr & ((1 << self.CRCwidth) - 1)

    def Get_bit(self, data: int, bit: int) -> int:
        """return the relevant bit

        :param data: int -> input data
        :param bit: int -> bit of the data to be returned
        :return relevant bit: int -> relevant bit
        """
        a = data >> bit & 0x1
        return a

    def Set_bit(self, linear_feedback_reg: int, bit: int, val: int):
        """set CRC bits

        :param linear_feedback_reg: int -> shift register
        :param bit: int -> bits to shift
        :param val: int -> relevant bit of the information
        :return relevant bit: int -> relevant bit
        """
```

```
        linear_feedback_reg = linear_feedback_reg & ~(0x1 << bit)
        return linear_feedback_reg | ((val & 0x1) << bit)

    def check_set_poly(self, insert: bool) -> int:
        """check if polynomial needs to be set

        :param insert: bool -> true if the relevant bit is set
        :return: int -> ether the polynomial or 0"""
        if insert:
            return self.Polynomial
        else:
            return 0x0


# CRC_A
RAA2P_A: CRC = CRC(0x3, 0xF, 11, 0x4)
address_crc: int = RAA2P_A.calc_crc(0x8000)
print(f"RAA2PXXXX Address CRC: 0x{address_crc:04X}")

# CRC
RAA2P: CRC = CRC(0x5, 0x1F, 16, 0x5)
register_crc: int = RAA2P.calc_crc(0x0606)
print(f"RAA2PXXXX CRC: 0x{register_crc:05X}")

data_crc: int = RAA2P.calc_crc(0x00CA)
print(f"RAA2PXXXX CRC: 0x{data_crc:05X}")

# NVM checksum
RAA2P_NVM_CRC: CRC = CRC(0x80F, 0xFFF, 71 * 16, 0xC)
cust_crc: int = RAA2P_NVM_CRC.calc_crc("Register 0x0 - 0x8C with 16b length")
```

Figure 17. Python CRC Code Example

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Disclaimer Rev.5.0-1)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.