Microcontroller interface to configure HO transducer

Contents

1. Introduction

This document aims to describe the programming function of HO series current transducer.

2. ASIC Description

2.1 Overview

HO series current transducer has ASIC (Application Specific Integrated Circuit) with EEPROM, and operates according to the setting data loaded from the EEPROM at power-ON.
You can select sensitivity (gain), reference voltage (V_{ref}), output filter setting, and so on by changing the data in EEPROM or RAM using serial communication.

2.2 ASIC Configuration EEPROM

The EEPROM loaded in ASIC consists of 32 words (16bits x 32).

Address (EEPROM)	Data (16 bits)															
	[15] (lock bit)	[14:10] (setting data : 5bits)					[9:5] (setting data : 5bits)					[4:0] (setting data : 5bits)				
00	X	D4	D3	D2	D1	D0	D4	D3	D2	D1	D0	D4	D3	D2	D1	D0
. . .	. . .															
26	X	D134	D133	D132	D131	D130	D134	D133	D132	D131	D130	D134	D133	D132	D131	D130
27	X	CRC-10 [4:0]					CRC-10 [4:0]					CRC-10 [4:0]				
28	X	CRC-10 [9:5]					CRC-10 [9:5]					CRC-10 [9:5]				
29 30 31	Read Only															

Table 2.1 EEPROM Address Map

- Address 00 ~ 26 : ASIC setting data
- Address 27 ~ 28 : CRC-10 (Check data)
- Address 29 ~ 31 : ASIC ID (Read Only)

ASIC setting data

To reduce a risk of incorrect operation by loss of memory data due to exogenous noise and so on, this ASIC operates by the majority vote data of 3 setting data which are stored in the EEPROM.
3 setting data are originally the same and they are divided in every 5 bits. The ASIC copies the majority vote data to the ROM when it is powered ON.
For example, the value of 'D0' is stored in address '00' at [0]bit (LSB), [5]bit, and [10]bit. ASIC reads the 3 data, and if more

than two data are “1”, ASIC will copy “1” to the ROM as the value of ‘D0’. Therefore, it is required to write same value into each 5 bits which are [4:0]bits, [9:5]bits, and [14:10]bits in the 16bits when you write a data to EEPROM.
Change of the EEPROM data will be applied at the next power-ON. If you need to change the current transducer operation immediately, you should change the ROM data.

CRC-10

CRC-10 is the check data called Cyclic Redundancy Check and it is defined by the standard ITU-T (International Telecommunication Union Telecommunication Standardization Sector) I.610.
You need to update the CRC value by defined computation method and write into the memory whenever you change the memory data. You should write same value into each 5 bits as previously described when you change the EEPROM data (See “ComputeCRC10” and “UpdateCRC10” function in the sample program).

The polynomial of CRC-10 : $x^{10} + x^9 + x^5 + x^4 + x + 1$ (11000110011)

The ASIC re-computes the value of CRC-10 from all of the setting data in the EEPROM and compares the computed data with the CRC-10 data stored in the address 27 and 28 at power-ON. If there is a difference between the two CRC-10 data, ASIC determines there is a defect at EEPROM data, and the output of HO series current transducer (V_{out}) is forced to 0V (at ARF = 1 setting).

ASIC ID
ASIC ID is a unique production number of the ASIC and it cannot be modified.

RAM
The RAM loaded in ASIC consists of 5bits x 28.

Table 2.2:
RAM address Map

Address (RAM)	Data (5bits) [4:0] (setting data : 5bits)				
	D4	D3	D2	D1	D0
00					
.					
.					
.					
26	D134	D133	D132	D131	D130
27	CRC-10 [4:0]				
28	CRC-10 [9:5]				

The ASIC behaves on the setting data in the ROM while the ASIC is power-ON. And you can change the operation of current transducer by changing the data in the ROM.
The data stored in ROM will disappear when the ASIC is powered OFF, and setting data will be copied from EEPROM at every powered ON. Therefore, you need to change the EEPROM data also if you need to keep the setting after you power off the current transducer. On the other hand, you can restore the default setting at every power-ON if you only change the ROM setting data.

ASIC setting parametes
The following table and description shows the seting parameters (EEPROM memory map) and each behaviour.

Address	DATA bit [15] (Lock bit)	[14]	[13]	[12]	[11]	[10]	[9]	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
0	1	GA[4] (D4)	GA[3] (D3)	GA[2] (D2)	GA[1] (D1)	GA[0] (D0)										
1	1	GA[9] (D9)	GA[8] (D8)	GA[7] (D7)	GA[6] (D6)	GA[5] (D5)										
2	1	GB[2] (D14)	GB[1] (D13)	GB[0] (D12)	GA[11] (D11)	GA[10] (D10)										
3	1	GB[7] (D19)	GB[6] (D18)	GB[5] (D17)	GB[4] (D16)	GB[3] (D15)										
4	1	GC[0] (D24)	GB[11] (D23)	GB[10] (D22)	GB[9] (D21)	GB[8] (D20)										
5	1	GC[5] (D29)	GC[4] (D28)	GC[3] (D27)	GC[2] (D26)	GC[1] (D25)										
6	1	GC[10] (D34)	GC[9] (D33)	GC[8] (D32)	GC[7] (D31)	GC[6] (D30)										
7	1	RESERVED (D39)	RESERVED (D38)	RESERVED (D37)	RESERVED (D36)	GC[11] (D35)										
8	0	DVR (D44)	RSI[1] (D43)	RS[0] (D42)	SGE[1] (D41)	SGE[0] (D40)										
9	0	STE[2] (D49)	STE[1] (D48)	STE[0] (D47)	SFI[1] (D46)	SF[0] (D45)										
10	0	TCCOLD[3] (D54)	TCCOLD[2] (D53)	TSS (D52)	TDT (D51)	ARF (D50)										
11	0	RTA[1] (D59)	RTA[0] (D58)	RESERVED (D57)	RESERVED (D56)	RESERVED (D55)										
12	0	RTC[0] (D64)	RTB[2] (D63)	RTB[1] (D62)	RTB[0] (D61)	RTA[2] (D60)										
13	0	RTD[2] (D69)	RTD[1] (D68)	RTD[0] (D67)	RTC[2] (D66)	RTC[1] (D65)										
14	0	OTA[4] (D74)	OTA[3] (D73)	OTA[2] (D72)	OTA[1] (D71)	OTA[0] (D70)										
15	0	OTB[4] (D79)	OTB[3] (D78)	OTB[2] (D77)	OTB[1] (D76)	OTB[0] (D75)										
16	0	OTC[4] (D84)	OTC[3] (D83)	OTC[2] (D82)	OTC[1] (D81)	OTC[0] (D80)										
17	1	T4C[4] (D89)	T4C[3] (D88)	T4C[2] (D87)	T4C[1] (D86)	T4C[0] (D85)										
18	0	T1C[4] (D94)	T1C[3] (D93)	T1C[2] (D92)	T1C[1] (D91)	T1C[0] (D90)										
19	0	TCHOT[2] (D99)	TCHOT[1] (D98)	TCHOT[0] (D97)	T1C[6] (D96)	T1C[5] (D95)										
20	0	SPARE bit (D104)	SPARE bit (D103)	TCHOT[3] (D102)	TCCOLD[1] (D101)	TCCOLD[0] (D100)										
21	0	RESERVED (D109)	RESERVED (D108)	RESERVED (D107)	RESERVED (D106)	SPARE bit (D105)										
22	0	RESERVED (D114)	SETTHRESH (D113)	RESERVED (D112)	RESERVED (D111)	RESERVED (D110)										
23	0	RESERVED (D119)	RESERVED (D118)	RESERVED (D117)	RESERVED (D116)	RESERVED (D115)										
24	0	RESERVED (D124)	RESERVED (D123)	RESERVED (D122)	RESERVED (D121)	RESERVED (D120)										
25	0	RESERVED (D129)	RESERVED (D128)	RESERVED (D127)	RESERVED (D126)	RESERVED (D125)										
26	0	SPARE bit (D134)	SPARE bit (D133)	SPARE bit (D132)	RESERVED (D131)	RESERVED (D130)										
27	0	CRC10 [04] (D139)	CRC10 [03] (D138)	CRC10 [02] (D137)	CRC10 [01] (D136)	CRC10 [00] (D135)										
28	0	CRC10 [09] (D144)	CRC10 [08] (D143)	CRC10 [07] (D142)	CRC10 [06] (D141)	CRC10 [05] (D140)										

Table 2.3:
ASIC Setting Parameters (EEPROM Memomry Map)

SGE : Gain Range Select (2bits)

The “SGE” parameter selects the sensitivity of the current transducer. The V_{out} of the transducer will be V_{ref} ±0.8V when its primary current is “± (each range value)”.

Setting data (2bits)		Behaviour
SGE [1] (D41)	SGE[0] (D42)	
0	0	25A range (default)
0	1	15A range
1	0	8A range
1	1	setting disable

RS : Reference Select (2bits)

The “RS” parameter selects the reference output voltage : V_{ref} of the transducer.

Setting data (2bits)		Behaviour
RS[1] (D43)	RS[0] (D42)	
0	0	V_{ref} = 2.5V (default)
0	1	V_{ref} = 1.65 V
1	0	V_{ref} = 1.5 V
1	1	V_{ref} = 0.5 V

DVR Disable Vref output (1bit)

The “DVR” parameter selects enable or disable the output of the V_{ref} . This transducer outputs the reference voltage from V_{ref} terminal. You can disconnect the internal reference of the ASIC and V_{ref} terminal by this parameter.

Setting data (1bits)	Behaviour
DVR [0] (D44)	
0	V_{ref} output : enabled (default)
1	V_{ref} output : disabled

SF Set output filter band width (2bits)

The “SF” parameter selects the cut-off frequency of the output filter of the transducer. You can select the cut-off frequency by changing this parameter.
You can reduce the noise level by selecting lower cut-off frequency but the response time will be longer at the same time.

Setting data (2bits)		Behaviour
SF[1] (D46)	SF[0] (D45)	
0	0	No filter
0	1	fc = 600kHz (default)
1	0	fc = 250kHz
1	1	fc = 100kHz

HO-NPPR Application Note

ASIC Description

STE Threshold level select (3bits)

The “STE” parameter selects the over current detection level. The over current detection : OCD terminal (pin #5) will be forced to ‘Low-level’ (internal open-drain-circuit turns to ON) when the primary current exceeds the value (peak) of ratio which are shown in the following list.

SET THRESH Threshold level range select (1bit)

The “SET THRESH” parameter selects the OCD range. This parameter should be used in combination with the parameter “STE” above (See the list below).

Setting data (3bits)			SET THRESH setting data (1 bit)	
			SET THRESH (D113)	
STE [2] (D49)	STE[1] (D48)	STE[0] (D48)	8A, 15A range:1 25A range:0	25A range:1 * (25A range only)
0	0	0	1.7 x I_{PN}	0.68 x I_{PN}
0	0	1	2.3 x I_{PN}	0.93 x I_{PN}
0	1	0	2.9 x I_{PN}	1.17 x I_{PN}
0	1	1	3.6 x I_{PN}	1.4 x I_{PN}
1	0	0	4.0 x I_{PN}	1.6 x I_{PN}
1	0	1	4.8 x I_{PN}	1.9 x I_{PN}
1	1	0	5.2 x I_{PN}	2.1 x I_{PN}
1	1	1	5.8 x I_{PN}	2.3 x I_{PN}

*The ratios in the right side of the list are available at 25A range only.

TDT Threshold Detect Time (1bits)

The “TDT” parameter selects the response time of the OCD signal. It specifies the number of internal clock of the ASIC from the over current detection to the OCD signal output.

Setting data (1bits)	Behaviour
TDT (D51)	
0	ASIC internal clock x 2 (default)
1	ASIC internal clock x 3

TSS Threshold Signal Select (1bits)

The “TSS” parameter selects the pulse width of the OCD signal.

Setting data (1bits)	Behaviour
TSS (D52)	
0	1 ms (default)
1	10 µs

ARF Allow Ready/Fail action (1bits)
The “ARF” parameter selects the behavior of when memory error occurred at power-ON. If there is a difference between the computed CRC-10 value from all of the setting data and the stored CRC-10 value (address 27 and 28) in the EEPROM, the output of current transducer : Vout will be forced to 0V at ARF = 1. If “ARF” is set as “0”, the V_{out} operates normally and it does not notify the occurrence of the error.

Setting data (1bits) ARF (D50)	Behaviour
0	V_{out} operates normally even if memory error occured
1	V_{out} is forced to 0V when memory error occured (default)

3. Serial communication

3.1 Overview

The ASIC in the HO series current transducer shifts into a communication mode by connecting the V_{ref} terminal to U_C terminal (power-supply voltage). The data is sent or received through V_{out} terminal.

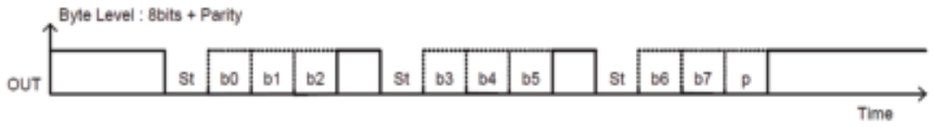
3.2 Timing chart

The ASIC has an UART (Universal Asynchronous Receiver Transmitter) and it operates with the following specification.

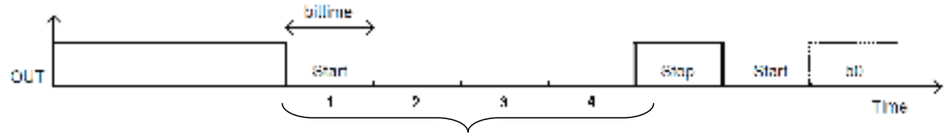
bit-rate 10k bit/sec
data structure ”start” 1bit + “data” 3bits + “stop” 1bit
parity odd

The timing chart is shown below.

The 8 bits data (b0 to b7) are restructured as 9 bits by adding parity bit. They are divided into every 3 bits and are sent in order from LSB (See ‘WriteByte’ and ‘WriteBit’ function in the sample program) .



The communication bit-rate is adjusted by first received initial synchronization pulse automatically. Therefore, it is necessary to create and send the synchronization pulse (start (0)x4bits + stop(1)x1bit) before you send commands of the ASIC when you start the communication (see ‘SynchMode’ function in the sample program).



Synchronization pulse (start (0)x4bits + stop (1)x1bit)

3.3 Command list

The following table shows the command list of the ASIC (The value is given in hexadecimal).

	1Byte (8bits)	1Byte (8bits)	1Byte (8bits)	
NOP				
mosi	00h			
miso				
READ_EE				
mosi	40h + addr			
miso		D[7:0]	D[15:8]	
READ_RAM				
mosi	60h + addr			*Received data is added “11Xb” in its MSB (X is indefinite)
miso		11Xb, D[4:0] @addr + 1	11Xb, D[4:0] @addr + 1	
WRITE/ERASE_EE				
mosi	80h + addr	D[7:0]	D[15:8]	*Wait 5 ms before any other commands.
miso				
WRITE_RAM				
mosi	C0h + addr	000b, D[4:0]		*Send data must be added “00b” in its MSB
miso				
Continuous WRITE_RAM				
mosi	C0h + addr	100b, D[4:0] @ addr	100b, D[4:0] @ addr + 1	000b, D[4:0] @ addr
miso				Send data must be added “100b” in its MSB, and last send data must be added “000b” in its MSB
UNLOCK				
mosi	EEh	36h		
miso				

Table 3.1: ASIC serial communication command list

*addr: The address of the memory

The value of “addr” must be from 0 to 31 for EEPROM and must be from 0 to 28 for RAM.
Wait 5ms before sending any other commands after writing to EEPROM.
The data length in the RAM is 5bits, therefore you need to create the data as 8bits by adding “000”(binary) in its MSB when you send ‘WRITE_RAM’ command. (add “100” when you send ‘Continuous WRITE_RAM’ command.)
The received data by ‘READ_RAM’ command has “11X”(binary, X is indefinite) in its MSB and the 5bits from LSB is actual data.

3.4 Communication Procedure

The ASIC in the HO series current transducer shifts into a communication mode by connecting the V_{ref} terminal to U_C terminal (power-supply voltage) as described above. The data is sent or received through Vout terminal.
The actual waveforms of following procedure are shown below.

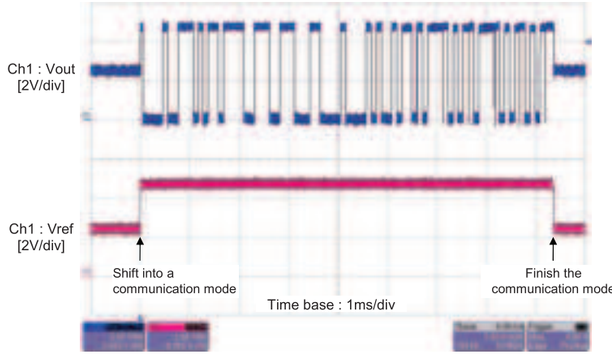


Fig 3.1:
ASIC serial communication waveform

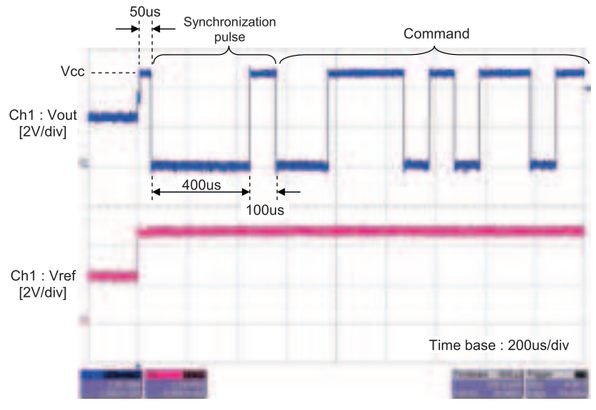


Fig 3.2:
ASIC serial communication waveform (zoom)

(1) Shift into a communication mode by connecting V_{ref} terminal to U_C.
(2) Wait 50 µs.
(3) Send the synchronization pulse (start (0)x4bits + stop(1)x1bit) to V_{out} terminal.
(4) Send and Receive command data
(4.1) Send ‘UNLOCK’ command (EEh,36h)
(4.2) Send ‘READ_EE’ command of EEPROM address:0 (40h + 0 (address))
(4.3) Read data from ASIC (8bit x 2 : D[7:0], D[15:8])
(5) Finish the communication mode by disconnecting V_{ref} terminal from U_C.

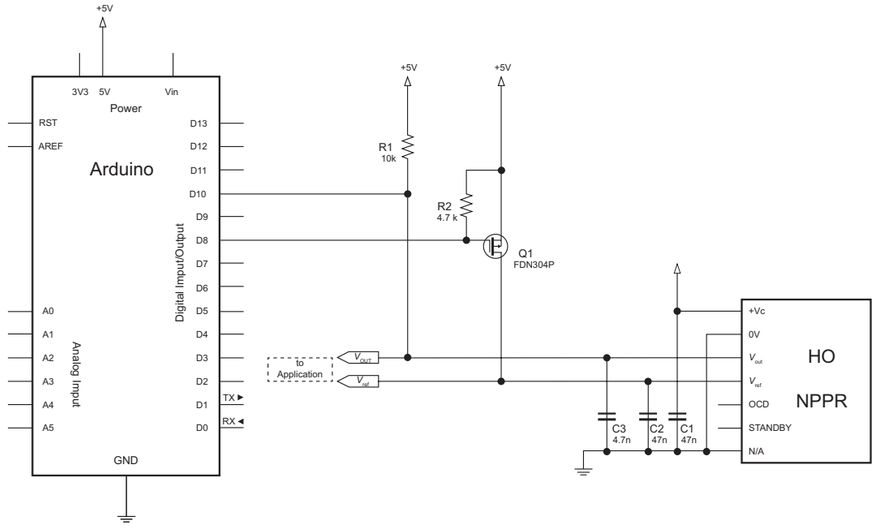
4. Implementation example on Arduino

4.1 Overview

“Arduino” is a general-purpose microcomputer system which contains a PCB with Atmel AVR processor with I/O ports and IDE (integrated development environment).
IDE is freely available on various platforms (Windows, Mac OS X, Linux), and it can be downloaded from “ http://arduino.cc/ “.

This chapter introduces you to the system structure and sample programs which changing the setting of the HO series current transducer.
(Regarding the set-up of IDE of Arduino and so on, please refer the website above or other related documents.)
● Confirmed boards
- Arduino UNO board
- Arduino NANO board

4.2 Schematics



In this example, D8 pin is assigned as a control of V_{ref} terminal (ON/OFF of the communication mode) and D10 pin is assigned as data signal line (send and receive of command data).

4.3 Sample program

You can operate the program which changes the setting of HO series transducer using this sample program with the example schematics in 4.2.
The sample program contains dedicated functions for Arduino microcontroller for setting of serial communication procedure, memory address and so on which are explained in chapter 2 and 3. It allows for changing the HO series transducer configuration by only executing 'update memory function' with setting parameters and memory definition (RAM or EEPROM).

4.3.1 List of sample program

Structure of the sample program provided for Arduino boards is shown below.

"sensorPTC.ico" is a main source file for Arduino board and it operates with included each

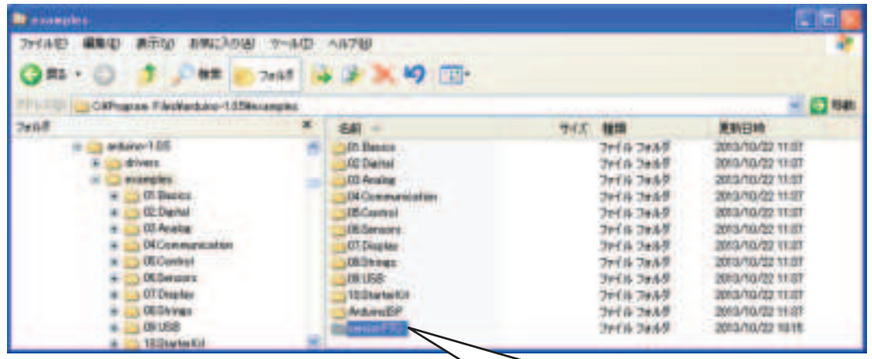
Folder		File name	Description
examples	sensorPTC	sensorPTC.ico	Sketch file for Arduino
libraries	ConfigDemoBoard	ConfigDemoBoard.h	Header file for initialisation
		typeDef.h	Header file for variable definition
	PTC	PTC.h	header file for communication setting
		PTC.cpp	Source file of communication setting functions
	HG2	HG2.h	Header file for parametes setting
		HG2.cpp	Source file of parameters setting functions

header files and source files of functions.
When you operate the sample program, please copy the sketch file to the "examples" folder, and copy the header/source file with subfolders to the "libraries" folder.

HO-NPPR Application Note

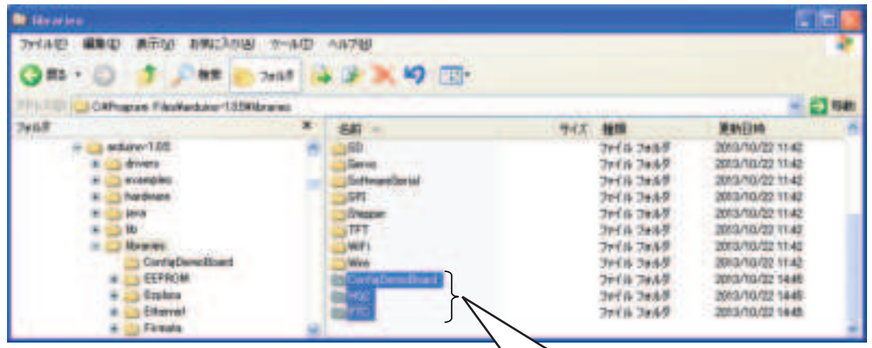
Implementation
Example on Arduino

- **SensorPTC folder**



Copy "sensorPTC" folder with sketch file into the "examples" folder under the Arduino folder.

- **ConfigDemoBoard HG2 PTC folder**



Copy "Config Demo Borad" folder, "HG2" folder, and "PTC" folder into the libraries" folder under the Arduino folder.

4.3.2 List of parameters and functions

Parameters and functions defined in this sample program are shown below.
You can change the setting of HO series transducer by updating setting data in the EEPROM or RAM with these parameters and functions.

- **Memory definition**

RAM Select “RAM” of the ASIC in HO series as an operating object
EEPROM Select “EEPROM” of the ASIC in HO series as an operating object

- Definition (PTC.h)

```
typedef enum {
        RAM,
        EEPROM,
        EXT_RAM         ( <- Not used )
} Memory_e;
```

- **Setting parameter’s name**

SGE_REG Select **SGE** setting parameter (Gain Range Select ; 2bits)
RS_REG Select **RS** setting parameter (Reference Select ; 2bits)
DVR_REG Select **DVR** setting parameter (Disable Vref output ; 1bit)
SF_REG Select **SF** setting parameter (Set output filter band width ; 2bits)
STE_REG Select **STE** setting parameter (Threshold level select ; 3bits)
SET_THRESH_REG Select **SET THRESH** setting parameter (Threshold level range select ; 1bit)
TDT_REG Select **TDT** setting parameter (Threshold Detect Time ; 1bits)
TSS_REG Select **TSS** setting parameter (Threshold Signal Select ; 1bits)
ARF_REG Select **ARF** setting parameter (Allow Ready/Fail action ; 1bits)

- Definition (HG2.h)

```
typedef enum {
        GA_REG,             // Gain set A - 12 bits             ( <- Not used )
        GB_REG,             // Gain set B - 12 bits             ( <- Not used )
        GC_REG,             // Gain set C - 12 bits             ( <- Not used )
        SGE_REG,            // Gain range Select - 2 bits
        RS_REG,             // Reference Select - 2 bits
        DVR_REG,            // Disable Vref output - 1 bit, 0=output enabled
        SF_REG,             // Set output Filter bandwidth - 2 bits
        STE_REG,            // Threshold level Select - 3 bits - only applicable for SOIC-8 package
        SET_THRESH_REG,     // Set Threshold High or Low Range - 1 bit
        TDT_REG,            // Threshold detect time - 1 bit
        TSS_REG,            // Threshold Signal Select - 1 bit
        ARF_REG,            // Allow ready/fail action - 1 bit, 0=not active
        TC4_REG,            // Offset drift, 1rst order - 5 bits    ( <- Not used )
        TC1_REG             // 1rst order gain TC - 7 bits          ( <- Not used )
    } ParamHG2_e;
```


HO-NPPR Application Note

Implementation
Example on Arduino

Function list

- **RetrieveMemory**

A function of activating the memory of ASIC in HO series. It is necessary to execute before you operate each memories.

Input : Type of memory (RAM or EEPROM)
Output : Return the status == 0 (OK), 1 (Reading parity error), 2 (Failed)

```
Name : void RetrieveMemory (Memory_e e_type)
-------------------------------------------------------------------------------
    Description :
            - Get current Memory configuration
              This function must be used at the first of each new setting
              (ie: use of SetParam function)
    Input :
            - Memory_e e_type:              Memory type
    Output :
            - Return the status:    0 (OK),1 (Reading Parity Error), 2 (Failed)
    Called by :
    Called Modules :
    References :
            - none
```

- **GetParam**

A function of reading current setting parameter’s value of HO series.

Input : Name of setting parameter (SGE_REG, RS_REG, and so on)
: Type of memory (RAM or EEPROM)
Output : Value of setting parameter (unsigned integer)

```
Name : U16 GetParam (ParamHG2_e eParam, Memory_e e_type)
-------------------------------------------------------------------------------
    Description :
            - Get Param from Memory (RAM | EEPROM | EXTENDED RAM)
    Input :
            - ParamHG2_e eParam:    Configurable register
            - Memory_e e_type:      Memory type
    Output :
            - U16:                  Register value
    Called by :
    Called Modules :
    References :
            - none
```

- **SetParam**

A function of sending each setting parameter's value to memory in the ASIC.

Input : Name of setting parameter (SGE_REG, RS_REG, and so on)
: The value of the setting parameter (unsigned integer)
: Type of memory (RAM or EEPROM)
Output : None

```
Name : U16 SetParam (ParamHG2_e eParam, U16 u16Value, Memory_e e_type)
-------------------------------------------------------------------------------
    Description :
            - Set Param in Memory (RAM | EEPROM | EXTENDED RAM)
    Input :
            - ParamHG2_e eParam:    Configrable register
            - U16 u16Value:         Register value to be set
            - Memory_e e_type:      Memory type
    Output :
            - none
    Called by :
    Called Modules :
    References :
            - none
```

- **UpdateMemory**

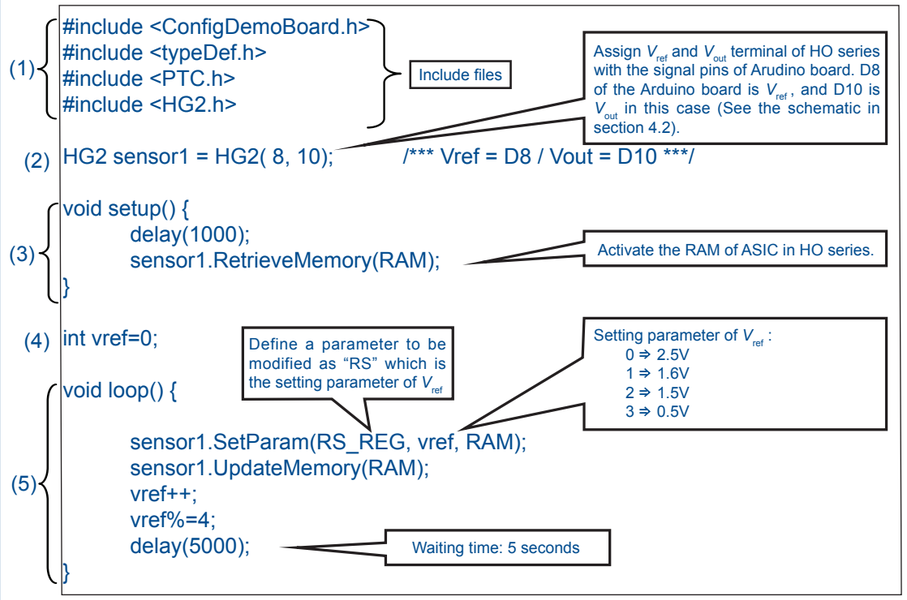
A function of updating the memory.
The defined memory (RAM or EEPROM) will be updated with calculated CRC-10 value after you send a setting parameter's value by SetParam function.
Please note that the memory data will not be updated unless you execute this UpdateMemory function.

Input : Type of memory (RAM or EEPROM)
Output : None

```
Name : void UpdateMemory (Memory_e e_type)
-------------------------------------------------------------------------------
    Description :
            - Update Memory with the current configuration
              This function must be used at the end of each new setting
             (ie: use of SetParam function)
    Input :
            - Memory_e e_type       : Memory type
    Output :
            - none
    Called by :
    Called Modules :
    References :
            - none
```


4.3.3 Explanation of sample program : sensorPTC

Explanation of the sample program "sensorPTC" is shown below.
This program switches the Vref setting of HO series as '2.5V => 1.6V => 1.5V => 0.5V' in every 5 seconds using the example schematics in section 4.2.



(1) Include each header files.
(2) Assign V_{ref} terminal of HO with D8 of Arduino board, and V_{out} with D10 based on the schematic in section 4.2 .[1)]
(3) Wait 1 second (1000ms) , and activate the RAM of ASIC in HO by executing "RetrieveMemory" function.
(4) Declare a variable "vref" and initialize it in "0"
(5) Loop function (Repetitive function)
(5.1) Define "RS_REG" parameter with 'SetParam' function and define its value as "vref".
(5.2) Update the value in the RAM with 'UpdateMemory' function. => The V_{ref} setting of the HO series switches.
(5.3) Add "1" to the variable "vref", and substitute the remainder of dividing it by 4 to the "vref".
(5.4) Wait 5 seconds (5000ms) and go back to the (5.1) .

1) The class named "HG2" is declared in the header file "HG2.h".
This line declares the object named "sensor1" of "HG2" class and assigns its initial values as its argument which define pin assignment information.

5 Appendix

5.1 HO 25-NPPR Initial value of setting parameters

Parameter Name	Value	Behaviour
SGE	0	Primary nomial current: I_{PN} = 25A range
RS	0	Internal reference voltage: V_{ref} = 2.5V
DVR	0	Internal reference voltage output: V_{ref} out is enabled
SF	1	
STE	2	OCD detection value = 2.9x I_{PN}
SET_THRESH	0	
TDT	0	OCD detection time: ASIC internal clock x 2 (1.3 µs)
TSS	0	OCD output pulse width : 1 ms
ARF	1	V_{out} is forced to 0V when memory error occured