

AND9594/D

LC717A10 Application Note

Overview

This application note describes the registers of LC717A10 and explains its various functions.

And it shows examples of application configuration and control method.



ON Semiconductor®

www.onsemi.com

APPLICATION NOTE

Index

Chapter 1 REGISTERS

1.1 Register Map	3
1.2 Register State at Reset.....	5
1.3 Details of Registers	7
1.4 Recommended Setting.....	27

Chapter 2 FUNCTIONS

2.1 Description of Functions	29
2.1.1 Interval Mode and Sleep Mode	29
2.1.2 Short Interval Mode and Long Interval Mode	29
2.1.3 Calibration	30
2.1.4 Noise Suppression	31
2.2 Measurement Functions	31
2.2.1 Relation Between Measurement Timing and Long Interval / Short Interval Setting	31
2.2.2 Description of Dynamic Offset Calibration	39
2.2.3 Approximate Calculation Formula for Measurement Interval.....	41
2.2.4 Description of Dynamic Threshold Control When Moving from ON State to OFF State.....	42

Chapter 3 OPERATION SEQUENCE

3.1 An Example of Operation Sequence After “External Reset”	45
3.2 An Example of Operation Sequence in the “Interval Mode”	47
3.3 An Example of Operation Sequence in the “Sleep Mode”.....	50

Chapter 4 CONTROL EXAMPLES FOR A MICROCONTROLLER

4.1 Control Flow Examples	53
4.1.1 The Microcontroller Reads the Measurement Result With the INTOUT Signal in the “Interval Mode”.....	53
4.1.2 The microcontroller Reads the Measurement Result Without the INTOUT Signal in the “Interval Mode”.....	56
4.1.3 The microcontroller Reads the Measurement Result With the INTOUT Signal in the “Sleep Mode”.....	59
4.2 Pseudo Code Examples for a Microcontroller to Control an LC717A10	62
4.2.1 Pseudo Code Example When a Microcontroller Reads the Measurement Result With the INTOUT Signal in the “Interval Mode”	65
4.2.2 Pseudo Code Example When a Microcontroller Reads the Measurement Result Without the INTOUT Signal in the “Interval Mode”.....	68
4.2.3 Pseudo Code Example When a Microcontroller Reads the Measurement Result With the INTOUT Signal in the “Sleep Mode”	71

Chapter 1 REGISTERS

1.1 Register Map

Register Address	R/W	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0x00	RW	Use Channel 1 Register	Cin7EN	Cin6EN	Cin5EN	Cin4EN	Cin3EN	Cin2EN	Cin1EN	Cin0EN
0x01	RW	Use Channel 2 Register	Cin15EN	Cin14EN	Cin13EN	Cin12EN	Cin11EN	Cin10EN	Cin9EN	Cin8EN
0x02	RW	Cin1/0 2nd Gain Register	Gain1_S3	Gain1_S2	Gain1_S1	Gain1_S0	Gain0_S3	Gain0_S2	Gain0_S1	Gain0_S0
0x03	RW	Cin3/2 2nd Gain Register	Gain3_S3	Gain3_S2	Gain3_S1	Gain3_S0	Gain2_S3	Gain2_S2	Gain2_S1	Gain2_S0
0x04	RW	Cin5/4 2nd Gain Register	Gain5_S3	Gain5_S2	Gain5_S1	Gain5_S0	Gain4_S3	Gain4_S2	Gain4_S1	Gain4_S0
0x05	RW	Cin7/6 2nd Gain Register	Gain7_S3	Gain7_S2	Gain7_S1	Gain7_S0	Gain6_S3	Gain6_S2	Gain6_S1	Gain6_S0
0x06	RW	Cin9/8 2nd Gain Register	Gain9_S3	Gain9_S2	Gain9_S1	Gain9_S0	Gain8_S3	Gain8_S2	Gain8_S1	Gain8_S0
0x07	RW	Cin11/10 2nd Gain Register	Gain11_S3	Gain11_S2	Gain11_S1	Gain11_S0	Gain10_S3	Gain10_S2	Gain10_S1	Gain10_S0
0x08	RW	Cin13/12 2nd Gain Register	Gain13_S3	Gain13_S2	Gain13_S1	Gain13_S0	Gain12_S3	Gain12_S2	Gain12_S1	Gain12_S0
0x09	RW	Cin15/14 2nd Gain Register	Gain15_S3	Gain15_S2	Gain15_S1	Gain15_S0	Gain14_S3	Gain14_S2	Gain14_S1	Gain14_S0
0x0A	RW	Cin0 Threshold Register	Cin0TH7	Cin0TH6	Cin0TH5	Cin0TH4	Cin0TH3	Cin0TH2	Cin0TH1	Cin0TH0
0x0B	RW	Cin1 Threshold Register	Cin1TH7	Cin1TH6	Cin1TH5	Cin1TH4	Cin1TH3	Cin1TH2	Cin1TH1	Cin1TH0
0x0C	RW	Cin2 Threshold Register	Cin2TH7	Cin2TH6	Cin2TH5	Cin2TH4	Cin2TH3	Cin2TH2	Cin2TH1	Cin2TH0
0x0D	RW	Cin3 Threshold Register	Cin3TH7	Cin3TH6	Cin3TH5	Cin3TH4	Cin3TH3	Cin3TH2	Cin3TH1	Cin3TH0
0x0E	RW	Cin4 Threshold Register	Cin4TH7	Cin4TH6	Cin4TH5	Cin4TH4	Cin4TH3	Cin4TH2	Cin4TH1	Cin4TH0
0x0F	RW	Cin5 Threshold Register	Cin5TH7	Cin5TH6	Cin5TH5	Cin5TH4	Cin5TH3	Cin5TH2	Cin5TH1	Cin5TH0
0x10	RW	Cin6 Threshold Register	Cin6TH7	Cin6TH6	Cin6TH5	Cin6TH4	Cin6TH3	Cin6TH2	Cin6TH1	Cin6TH0
0x11	RW	Cin7 Threshold Register	Cin7TH7	Cin7TH6	Cin7TH5	Cin7TH4	Cin7TH3	Cin7TH2	Cin7TH1	Cin7TH0
0x12	RW	Cin8 Threshold Register	Cin8TH7	Cin8TH6	Cin8TH5	Cin8TH4	Cin8TH3	Cin8TH2	Cin8TH1	Cin8TH0
0x13	RW	Cin9 Threshold Register	Cin9TH7	Cin9TH6	Cin9TH5	Cin9TH4	Cin9TH3	Cin9TH2	Cin9TH1	Cin9TH0
0x14	RW	Cin10 Threshold Register	Cin10TH7	Cin10TH6	Cin10TH5	Cin10TH4	Cin10TH3	Cin10TH2	Cin10TH1	Cin10TH0
0x15	RW	Cin11 Threshold Register	Cin11TH7	Cin11TH6	Cin11TH5	Cin11TH4	Cin11TH3	Cin11TH2	Cin11TH1	Cin11TH0
0x16	RW	Cin12 Threshold Register	Cin12TH7	Cin12TH6	Cin12TH5	Cin12TH4	Cin12TH3	Cin12TH2	Cin12TH1	Cin12TH0
0x17	RW	Cin13 Threshold Register	Cin13TH7	Cin13TH6	Cin13TH5	Cin13TH4	Cin13TH3	Cin13TH2	Cin13TH1	Cin13TH0
0x18	RW	Cin14 Threshold Register	Cin14TH7	Cin14TH6	Cin14TH5	Cin14TH4	Cin14TH3	Cin14TH2	Cin14TH1	Cin14TH0
0x19	RW	Cin15 Threshold Register	Cin15TH7	Cin15TH6	Cin15TH5	Cin15TH4	Cin15TH3	Cin15TH2	Cin15TH1	Cin15TH0
0x1A	R	Cin0 Data Register	DATA07	DATA06	DATA05	DATA04	DATA03	DATA02	DATA01	DATA00
0x1B	R	Cin1 Data Register	DATA17	DATA16	DATA15	DATA14	DATA13	DATA12	DATA11	DATA10
0x1C	R	Cin2 Data Register	DATA27	DATA26	DATA25	DATA24	DATA23	DATA22	DATA21	DATA20
0x1D	R	Cin3 Data Register	DATA37	DATA36	DATA35	DATA34	DATA33	DATA32	DATA31	DATA30
0x1E	R	Cin4 Data Register	DATA47	DATA46	DATA45	DATA44	DATA43	DATA42	DATA41	DATA40
0x1F	R	Cin5 Data Register	DATA57	DATA56	DATA55	DATA54	DATA53	DATA52	DATA51	DATA50
0x20	R	Cin6 Data Register	DATA67	DATA66	DATA65	DATA64	DATA63	DATA62	DATA61	DATA60
0x21	R	Cin7 Data Register	DATA77	DATA76	DATA75	DATA74	DATA73	DATA72	DATA71	DATA70
0x22	R	Cin8 Data Register	DATA87	DATA86	DATA85	DATA84	DATA83	DATA82	DATA81	DATA80
0x23	R	Cin9 Data Register	DATA97	DATA96	DATA95	DATA94	DATA93	DATA92	DATA91	DATA90
0x24	R	Cin10 Data Register	DATA107	DATA106	DATA105	DATA104	DATA103	DATA102	DATA101	DATA100
0x25	R	Cin11 Data Register	DATA117	DATA116	DATA115	DATA114	DATA113	DATA112	DATA111	DATA110
0x26	R	Cin12 Data Register	DATA127	DATA126	DATA125	DATA124	DATA123	DATA122	DATA121	DATA120
0x27	R	Cin13 Data Register	DATA137	DATA136	DATA135	DATA134	DATA133	DATA132	DATA131	DATA130
0x28	R	Cin14 Data Register	DATA147	DATA146	DATA145	DATA144	DATA143	DATA142	DATA141	DATA140
0x29	R	Cin15 Data Register	DATA157	DATA156	DATA155	DATA154	DATA153	DATA152	DATA151	DATA150

Continued to the next page.

AND9594/D

Continued from the previous page.

Register Address	R/W	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0x2A	R	Result Data 1 Register	Cin7ACT	Cin6ACT	Cin5ACT	Cin4ACT	Cin3ACT	Cin2ACT	Cin1ACT	Cin0ACT
0x2B	R	Result Data 2 Register	Cin15ACT	Cin14ACT	Cin13ACT	Cin12ACT	Cin11ACT	Cin10ACT	Cin9ACT	Cin8ACT
0x2C	R	Error Status Register	YSERR	-	-	-	-	-	-	CALERR
0x2D	R	Error Channel Status 1 Register	Cin7ERR	Cin6ERR	Cin5ERR	Cin4ERR	Cin3ERR	Cin2ERR	Cin1ERR	Cin0ERR
0x2E	R	Error Channel Status 2 Register	Cin15ERR	Cin14ERR	Cin13ERR	Cin12ERR	Cin11ERR	Cin10ERR	Cin9ERR	Cin8ERR
0x2F	RW	Control 1 Register	WriteReq	Rsvd6	Rsvd5	Rsvd4	IntMode	ParaCh	StaCal	Measure
0x30	RW	Average Count Register	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
0x31	RW	Filter Parameter Register	FP2_3	FP2_2	FP2_1	FP2_0	FP1_3	FP1_2	FP1_1	FP1_0
0x32	RW	Debounce Count 1 Register	DCT1_7	DCT1_6	DCT1_5	DCT1_4	DCT1_3	DCT1_2	DCT1_1	DCT1_0
0x33	RW	Debounce Count 2 Register	DCT2_7	DCT2_6	DCT2_5	DCT2_4	DCT2_3	DCT2_2	DCT2_1	DCT2_0
0x34	RW	Short Interval Time Register	SIVAL7	SIVAL6	SIVAL5	SIVAL4	SIVAL3	SIVAL2	SIVAL1	SIVAL0
0x35	RW	Long Interval Time Register	LIVAL7	LIVAL6	LIVAL5	LIVAL4	LIVAL3	LIVAL2	LIVAL1	LIVAL0
0x36	RW	Short Interval Dynamic OffCal Cycle Register	DCYC7	DCYC6	DCYC5	DCYC4	DCYC3	DCYC2	DCYC1	DCYC0
0x37	RW	Dynamic OffCal Count Plus Register	DCALP7	DCALP6	DCALP5	DCALP4	DCALP3	DCALP2	DCALP1	DCALP0
0x38	RW	Dynamic OffCal Count Minus Register	DCALM7	DCALM6	DCALM5	DCALM4	DCALM3	DCALM2	DCALM1	DCALM0
0x39	RW	Static OffCal CDAC Base Register	DACB7	DACB6	DACB5	DACB4	DACB3	DACB2	DACB1	DACB0
0x3A	RW	Measurement Mode 1 Register	INTMD1	INTMD2	PDCLP	TOFFTH	LIVALB	Rsvd2	Rsvd1	Rsvd0
0x3B	RW	Measurement Mode 2 Register	Rsvd7	Rsvd6	Rsvd5	CIN8CINP2	CIN0CINP2	CIN8CINP	CIN0CINP	Rsvd0
0x3C	RW	Long Interval Mode Start Count Register	LIMSC_7	LIMSC_6	LIMSC_5	LIMSC_4	LIMSC_3	LIMSC_2	LIMSC_1	LIMSC_0
0x3D	RW	Cin 1st Gain Adjust Register	Gain8_F3	Gain8_F2	Gain8_F1	Gain8_F0	Gain0_F3	Gain0_F2	Gain0_F1	Gain0_F0
0x3E	-	Reserved area (Write prohibited)	-	-	-	-	-	-	-	-
0x3F	-	Reserved area (Write prohibited)	-	-	-	-	-	-	-	-
0x40	RW	Control 2 Register	SoftRst	Rsvd6	Hold	Rsvd4	Rsvd3	Rsvd2	IntOut	WakeUp
0x41 to 0x6B	-	Reserved area (Write prohibited)	-	-	-	-	-	-	-	-
0x6C	R	CDAC Offset Plus Register	CdacP7	CdacP6	CdacP5	CdacP4	CdacP3	CdacP2	CdacP1	CdacP0
0x6D	R	CDAC Offset Minus Register	CdacM7	CdacM6	CdacM5	CdacM4	CdacM3	CdacM2	CdacM1	CdacM0
0x6E to 0x7E	-	Reserved area (Write prohibited)	-	-	-	-	-	-	-	-
0x7F	R	SLAVE Address Register	Rsvd7	Slave6	Slave5	Slave4	Slave3	Slave2	Slave1	SA0
0x80 to 0xFF	-	Reserved area (Write prohibited)	-	-	-	-	-	-	-	-

Note: If you read a register in the reserved area, the value is not guaranteed.

AND9594/D

1.2 Register State at Reset

Register Address	Reset Value	Name	Description
0x00	0xFF	Use Channel 1 Register	Channels from Cin0 to Cin7 are enabled
0x01	0xFF	Use Channel 2 Register	Channels from Cin8 to Cin15 are enabled
0x02	0x66	Cin1/0 2nd Gain Register	- Cin0 2nd Gain = 7 times - Cin1 2nd Gain = 7 times
0x03	0x66	Cin3/2 2nd Gain Register	- Cin2 2nd Gain = 7 times - Cin3 2nd Gain = 7 times
0x04	0x66	Cin5/4 2nd Gain Register	- Cin4 2nd Gain = 7 times - Cin5 2nd Gain = 7 times
0x05	0x66	Cin7/6 2nd Gain Register	- Cin6 2nd Gain = 7 times - Cin7 2nd Gain = 7 times
0x06	0x66	Cin9/8 2nd Gain Register	- Cin8 2nd Gain = 7 times - Cin9 2nd Gain = 7 times
0x07	0x66	Cin11/10 2nd Gain Register	- Cin10 2nd Gain = 7 times - Cin11 2nd Gain = 7 times
0x08	0x66	Cin13/12 2nd Gain Register	- Cin12 2nd Gain = 7 times - Cin13 2nd Gain = 7 times
0x09	0x66	Cin15/14 2nd Gain Register	- Cin14 2nd Gain = 7 times - Cin15 2nd Gain = 7 times
0x0A	0x32	Cin0 Threshold Register	Cin0 threshold = 50
0x0B	0x32	Cin1 Threshold Register	Cin1 threshold = 50
0x0C	0x32	Cin2 Threshold Register	Cin2 threshold = 50
0x0D	0x32	Cin3 Threshold Register	Cin3 threshold = 50
0x0E	0x32	Cin4 Threshold Register	Cin4 threshold = 50
0x0F	0x32	Cin5 Threshold Register	Cin5 threshold = 50
0x10	0x32	Cin6 Threshold Register	Cin6 threshold = 50
0x11	0x32	Cin7 Threshold Register	Cin7 threshold = 50
0x12	0x32	Cin8 Threshold Register	Cin8 threshold = 50
0x13	0x32	Cin9 Threshold Register	Cin9 threshold = 50
0x14	0x32	Cin10 Threshold Register	Cin10 threshold = 50
0x15	0x32	Cin11 Threshold Register	Cin11 threshold = 50
0x16	0x32	Cin12 Threshold Register	Cin12 threshold = 50
0x17	0x32	Cin13 Threshold Register	Cin13 threshold = 50
0x18	0x32	Cin14 Threshold Register	Cin14 threshold = 50
0x19	0x32	Cin15 Threshold Register	Cin15 threshold = 50
0x1A	0x00	Cin0 Data Register	Measurement value of Cin0
0x1B	0x00	Cin1 Data Register	Measurement value of Cin1
0x1C	0x00	Cin2 Data Register	Measurement value of Cin2
0x1D	0x00	Cin3 Data Register	Measurement value of Cin3
0x1E	0x00	Cin4 Data Register	Measurement value of Cin4
0x1F	0x00	Cin5 Data Register	Measurement value of Cin5
0x20	0x00	Cin6 Data Register	Measurement value of Cin6
0x21	0x00	Cin7 Data Register	Measurement value of Cin7
0x22	0x00	Cin8 Data Register	Measurement value of Cin8
0x23	0x00	Cin9 Data Register	Measurement value of Cin9
0x24	0x00	Cin10 Data Register	Measurement value of Cin10
0x25	0x00	Cin11 Data Register	Measurement value of Cin11
0x26	0x00	Cin12 Data Register	Measurement value of Cin12
0x27	0x00	Cin13 Data Register	Measurement value of Cin13
0x28	0x00	Cin14 Data Register	Measurement value of Cin14
0x29	0x00	Cin15 Data Register	Measurement value of Cin15

Continued to the next page.

AND9594/D

Continued from the previous page.

Register Address	Reset Value	Name	Description
0x2A	0x00	Result Data 1 Register	"ON"/"OFF" judgment of Cin0 to Cin7
0x2B	0x00	Result Data 2 Register	"ON"/"OFF" judgment of Cin8 to Cin15
0x2C	0x00	Error Status Register	Error status
0x2D	0x00	Error Channel Status 1 Register	Calibration error status for Cin0 to Cin7
0x2E	0x00	Error Channel Status 2 Register	Calibration error status for Cin8 to Cin15
0x2F	0x0B	Control 1 Register	- Not reflect setting (WriteReq = 0) - Operation in interval mode (IntMode = 1) - Not request for changing parameters (ParaCh = 0) - Static offset calibration is requested (StaCal = 1) - Measurement operation (Measure = 1)
0x30	0x40	Average Count Register	Average counts = 64 times
0x31	0x04	Filter Parameter Register	- Filter parameter 1 = 4 - Filter parameter 2 = 0
0x32	0x02	Debounce Count 1 Register	Debounce count (OFF to ON) = 3 counts
0x33	0x02	Debounce Count 2 Register	Debounce count (ON to OFF) = 3 counts
0x34	0x05	Short Interval Time Register	Short interval time = 5 ms (Typ)
0x35	0x01	Long Interval Time Register	Long interval time = 101 ms (Typ)
0x36	0x07	Short Interval Dynamic OffCal Cycle Register	The judgment cycle number for dynamic offset calibration execution in short interval mode = Once every 7 measurements
0x37	0x03	Dynamic OffCal Count Plus Register	The plus side's dynamic offset calibration execution count number = 24 points
0x38	0x03	Dynamic OffCal Count Minus Register	The minus side's dynamic offset calibration execution count number = 3 points
0x39	0x80	Static OffCal Base Register	The reference capacity used at static offset calibration = 4 pF
0x3A	0x00	Measurement Mode 1 Register	- INTOUT is always asserted after every measurement - INTOUT is not negated automatically - The plus side's dynamic offset calibration is performed when all enabled channels are OFF - Touch OFF threshold = 3/4 of peak of the value while touch is detected - Long interval base time = 100 ms
0x3B	0x00	Measurement Mode 2 Register	- Cin8 to Cin15 measurement = Use only Cref - Cin0 to Cin7 measurement = Use only Cref
0x3C	0x00	Long Interval Mode Start Count Register	Cycle before start of long interval = 0 point
0x3D	0x00	Cin 1st Gain Adjust Register	- Cin8 to Cin15 1st Gain = 1600 (Min) - Cin0 to Cin7 1st Gain = 1600 (Min)
0x40	0x00	Control 2 Register	- Normal operation - INTOUT is negated
0x7F	0x16/0x17 0x18/0x19	SLAVE Address Register	I ² C bus slave address

AND9594/D

1.3 Details of Registers

●Use Channel 1 Register

Address	Bit	7	6	5	4	3	2	1	0
0x00	Name	Cin7EN	Cin6EN	Cin5EN	Cin4EN	Cin3EN	Cin2EN	Cin1EN	Cin0EN
	Reset	1	1	1	1	1	1	1	1
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to enable/disable each channel of LC717A10. The Cin0EN bit (Bit0) is used to enable or disable Cin0, and the Cin7EN bit (Bit7) is used to enable or disable Cin7.

CinXEN (X=0 to 7)

0: Disable the channel.

1: Enable the channel.

●Use Channel 2 Register

Address	Bit	7	6	5	4	3	2	1	0
0x01	Name	Cin15EN	Cin14EN	Cin13EN	Cin12EN	Cin11EN	Cin10EN	Cin9EN	Cin8EN
	Reset	1	1	1	1	1	1	1	1
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to enable/disable each channel of LC717A10. The Cin8EN bit (Bit0) is used to enable or disable Cin8, and the Cin15EN bit (Bit7) is used to enable or disable Cin15.

CinXEN (X=8 to 15)

0: Disable the channel.

1: Enable the channel.

AND9594/D

●Cin1/0 2nd Gain Register

Address	Bit	7	6	5	4	3	2	1	0
	0x02	Name	Gain1_S3	Gain1_S2	Gain1_S1	Gain1_S0	Gain0_S3	Gain0_S2	Gain0_S1
Reset		0	1	1	0	0	1	1	0
R/W		RW	RW	RW	RW	RW	RW	RW	RW

In this LSI, there is a two-stage amplifier that detects changes in the capacitance and outputs an analog signal corresponding to the changes. The two stage amplifier consists of 1st-amplifier and 2nd-amplifier. This register is used to specify the 2nd-amplifier gain and to adjust sensitivity of Cin0 and Cin1. The initial values are 7 times (0x66).

The gain of 1st-amplifier is specified with the Cin 1st Gain Adjust Register (Address=0x3D).

Gain0_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin0 by the lower 4 bits.

Gain1_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin1 by the upper 4 bits.

Gain0_S3	Gain0_S2	Gain0_S1	Gain0_S0	2nd-amplifier gain ga2 [times]
Gain1_S3	Gain1_S2	Gain1_S1	Gain1_S0	
0	0	0	0	1 (Minimum gain setting)
0	0	0	1	2
0	0	1	0	3
0	0	1	1	4
0	1	0	0	5
0	1	0	1	6
0	1	1	0	7 (Initial value)
0	1	1	1	8
1	0	0	0	9
1	0	0	1	10
1	0	1	0	11
1	0	1	1	12
1	1	0	0	13
1	1	0	1	14
1	1	1	0	15
1	1	1	1	16 (Maximum gain setting)

<Calculation formula>

Output voltage of a two-stage amplifier

Output voltage of the 1st-amplifier: $\Delta V_1 = (\Delta C/C_f) \times V_{CDRV}$ $\Delta V_1 < 0.8 \times V_{DD}$

Output voltage of the 2nd-amplifier: $\Delta V_2 = \Delta V_1 \times ga2$ $\Delta V_2 < 0.8 \times V_{DD}$

ΔC : Change in input capacitance (Capacitance change when touch)

C_f : The 1st-amplifier gain setting (Feedback capacitance in the LSI)

V_{CDRV} : High output voltage of Cdrv (= V_{DD})

$ga2$: The 2nd-amplifier gain setting (Times)

●Cin3/2 2nd Gain Register

Address	Bit	7	6	5	4	3	2	1	0
	0x03	Name	Gain3_S3	Gain3_S2	Gain3_S1	Gain3_S0	Gain2_S3	Gain2_S2	Gain2_S1
Reset		0	1	1	0	0	1	1	0
R/W		RW	RW	RW	RW	RW	RW	RW	RW

This register is the same as Cin1/0 2nd Gain Register.

Gain2_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin2 by the lower 4 bits.

Gain3_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin3 by the upper 4 bits.

AND9594/D

●Cin5/4 2nd Gain Register

Address	Bit	7	6	5	4	3	2	1	0
0x04	Name	Gain5_S3	Gain5_S2	Gain5_S1	Gain5_S0	Gain4_S3	Gain4_S2	Gain4_S1	Gain4_S0
	Reset	0	1	1	0	0	1	1	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is the same as Cin1/0 2nd Gain Register.

Gain4_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin4 by the lower 4 bits.

Gain5_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin5 by the upper 4 bits.

●Cin7/6 2nd Gain Register

Address	Bit	7	6	5	4	3	2	1	0
0x05	Name	Gain7_S3	Gain7_S2	Gain7_S1	Gain7_S0	Gain6_S3	Gain6_S2	Gain6_S1	Gain6_S0
	Reset	0	1	1	0	0	1	1	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is the same as Cin1/0 2nd Gain Register.

Gain6_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin6 by the lower 4 bits.

Gain7_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin7 by the upper 4 bits.

●Cin9/8 2nd Gain Register

Address	Bit	7	6	5	4	3	2	1	0
0x06	Name	Gain9_S3	Gain9_S2	Gain9_S1	Gain9_S0	Gain8_S3	Gain8_S2	Gain8_S1	Gain8_S0
	Reset	0	1	1	0	0	1	1	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is the same as Cin1/0 2nd Gain Register.

Gain8_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin8 by the lower 4 bits.

Gain9_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin9 by the upper 4 bits.

●Cin11/10 2nd Gain Register

Address	Bit	7	6	5	4	3	2	1	0
0x07	Name	Gain11_S3	Gain11_S2	Gain11_S1	Gain11_S0	Gain10_S3	Gain10_S2	Gain10_S1	Gain10_S0
	Reset	0	1	1	0	0	1	1	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is the same as Cin1/0 2nd Gain Register.

Gain10_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin10 by the lower 4 bits.

Gain11_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin11 by the upper 4 bits.

AND9594/D

●Cin13/12 2nd Gain Register

Address	Bit	7	6	5	4	3	2	1	0
0x08	Name	Gain13_S3	Gain13_S2	Gain13_S1	Gain13_S0	Gain12_S3	Gain12_S2	Gain12_S1	Gain12_S0
	Reset	0	1	1	0	0	1	1	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is the same as Cin1/0 2nd Gain Register.

Gain12_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin12 by the lower 4 bits.

Gain13_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin13 by the upper 4 bits.

●Cin15/14 2nd Gain Register

Address	Bit	7	6	5	4	3	2	1	0
0x09	Name	Gain15_S3	Gain15_S2	Gain15_S1	Gain15_S0	Gain14_S3	Gain14_S2	Gain14_S1	Gain14_S0
	Reset	0	1	1	0	0	1	1	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is the same as Cin1/0 2nd Gain Register.

Gain14_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin14 by the lower 4 bits.

Gain15_S0 to 3: This field is used to specify the 2nd-amplifier gain for Cin15 by the upper 4 bits.

●CinX Threshold Register (X=0 to 15)

Address	Bit	7	6	5	4	3	2	1	0
0x0A to 0x19	Name	CinXTH7	CinXTH6	CinXTH5	CinXTH4	CinXTH3	CinXTH2	CinXTH1	CinXTH0
	Reset	0	0	1	1	0	0	1	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify a threshold (0 to 127) to judge “ON”/“OFF” from measurement data of each channel.

Specify the threshold of Cin0 by Cin0 Threshold Register (Address=0x0A), and specify the threshold of Cin15 by Cin15 Threshold Register (Address=0x19).

The threshold value for “ON”/“OFF” judgment has to be changed when changing gain setting.

The threshold value of “Touch OFF” judgment is specified with the TOFFTH bit in the Measurement Mode 1 Register (Address=0x3A). For more information, refer to “2.2.4 Description of Dynamic Threshold Control When Moving from ON State to OFF State” in this document.

Only a plus value can be set (The CinXTH7 bit is fixed to “0”. Setting the CinXTH7 bit to “1” is prohibited.).

The initial values are 50 (0x32).

CinXTH7	CinXTH6	CinXTH5	CinXTH4	CinXTH3	CinXTH2	CinXTH1	CinXTH0	Threshold
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0	1	1	1	1	1	1	0	126
0	1	1	1	1	1	1	1	127
1	0	0	0	0	0	0	0	Prohibited
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	1	1	1	Prohibited

AND9594/D

●CinX Data Register (X=0 to 15)

Address	Bit	7	6	5	4	3	2	1	0
0x1A to 0x29	Name	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0
	Reset	0	0	0	0	0	0	0	0
	R/W	R	R	R	R	R	R	R	R

This register is used to read out the measurement data obtained through each channel (-128 to 0 to 127). The stored data is in 2's complement form (0x80 to 0x00 to 0x7F). The measurement data of Cin0 is stored in the Cin0 Data Register (Address=0x1A) and the measurement data of Cin15 is stored in the Cin15 Data Register (Address=0x29) respectively.

This measurement data may be read by the microcontroller to judge the touch "ON"/"OFF" status on the side of the microcontroller.

DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0	Measurement data	(Hex)
1	0	0	0	0	0	0	0	-128	(0x80)
1	0	0	0	0	0	0	1	-127	(0x81)
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	1	1	0	-2	(0xFE)
1	1	1	1	1	1	1	1	-1	(0xFF)
0	0	0	0	0	0	0	0	0	(0x00)
0	0	0	0	0	0	0	1	1	(0x01)
0	0	0	0	0	0	1	0	2	(0x02)
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0	1	1	1	1	1	1	0	126	(0x7E)
0	1	1	1	1	1	1	1	127	(0x7F)

●Result Data 1 Register

Address	Bit	7	6	5	4	3	2	1	0
0x2A	Name	Cin7ACT	Cin6ACT	Cin5ACT	Cin4ACT	Cin3ACT	Cin2ACT	Cin1ACT	Cin0ACT
	Reset	0	0	0	0	0	0	0	0
	R/W	R	R	R	R	R	R	R	R

This register is used to notify the result of "ON"/"OFF" judgment, which is made from both the measurement data of each channel and the threshold for each channel by this LSI.

The result data of Cin0 corresponds to the Cin0ACT bit (Bit0), the result data of Cin7 corresponds to the Cin7ACT bit (Bit7).

CinXACT (X=0 to 7)

0: Touch OFF judgment (Initial value)

1: Touch ON judgment

●Result Data 2 Register

Address	Bit	7	6	5	4	3	2	1	0
0x2B	Name	Cin15ACT	Cin14ACT	Cin13ACT	Cin12ACT	Cin11ACT	Cin10ACT	Cin9ACT	Cin8ACT
	Reset	0	0	0	0	0	0	0	0
	R/W	R	R	R	R	R	R	R	R

This register is used to notify the result of "ON"/"OFF" judgment, which is made from both the measurement data of each channel and the threshold for each channel by this LSI.

The result data of Cin8 corresponds to the Cin8ACT bit (Bit0), the result data of Cin15 corresponds to the Cin15ACT bit (Bit7).

CinXACT (X=8 to 15)

0: Touch OFF judgment (Initial value)

1: Touch ON judgment

●Error Status Register

Address	Bit	7	6	5	4	3	2	1	0
0x2C	Name	SYSERR	-	-	-	-	-	-	CALERR
	Reset	0	0	0	0	0	0	0	0
	R/W	R	R	R	R	R	R	R	R

This register is used to notify the presence of errors.

SYSERR

0: No system error (Initial value)

1: System error occurred

In the case that the SYSERR bit is set to “1”, perform either software-reset or power-on reset in order to reset the SYSERR bit to “0”. For example, set the SoftRst bit in the Control 2 Register (Address=0x40) to “1”.

CALERR

0: No calibration error (Initial value)

1: Calibration error occurred

In the case that the CALERR bit is set to “1”, perform following processing in order to reset the CALERR bit to “0”. At first, reviews the customer’s noise environment, sensor patterns and register-setting-values. Next, to reflect the new settings to the LSI’s operation, and execute the static offset calibration and the “Measurement” again.

●Error Channel Status 1 Register

Address	Bit	7	6	5	4	3	2	1	0
0x2D	Name	Cin7ERR	Cin6ERR	Cin5ERR	Cin4ERR	Cin3ERR	Cin2ERR	Cin1ERR	Cin0ERR
	Reset	0	0	0	0	0	0	0	0
	R/W	R	R	R	R	R	R	R	R

This register is used to notify the occurrence of either the calibration error from Cin0 to Cin7 or the system error code (0x00).

CinXERR (X=0 to7)

0: Channel without calibration error (Initial value)

1: Channel with calibration error

●Error Channel Status 2 Register

Address	Bit	7	6	5	4	3	2	1	0
0x2E	Name	Cin15ERR	Cin14ERR	Cin13ERR	Cin12ERR	Cin11ERR	Cin10ERR	Cin9ERR	Cin8ERR
	Reset	0	0	0	0	0	0	0	0
	R/W	R	R	R	R	R	R	R	R

This register is used to notify the occurrence of either the calibration error from Cin8 to Cin15 or the system error code (0x00).

CinXERR (X=8 to15)

0: Channel without calibration error (Initial value)

1: Channel with calibration error

●Control 1 Register

Bit	7	6	5	4	3	2	1	0	
Address	Name	WriteReq	Rsvd6	Rsvd5	Rsvd4	IntMode	ParaCh	StaCal	Measure
0x2F	Reset	0	0	0	0	1	0	1	1
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify operation of the LSI processing.

Note: *When at least one bit of the WriteReq bit, the ParaCh bit and the StaCal bit is not “0”, the microcontroller must not write to the Control 1 Register. (In other word, only when all of the WriteReq bit, the ParaCh bit and the StaCal bit are “0”, the microcontroller is able to write to the Control 1 Register.)*

Exceptions: Only during the period from the time when the LSI asserts the INTOUT to notify the completion of the LSI internal initialization after the release of reset until the time when static offset calibration which is automatically performed based on default parameters in the LSI has completed, the microcontroller can write 0x80 or 0x88 to the Control 1 Register even if the read value of the StaCal bit is “1”. By doing this, the execution of static offset calibration and/or measurement based on LSI’s default parameters can be cancelled. For more information, refer to “Chapter 4. CONTROL EXAMPLES FOR A MICRO-CONTROLLER”.

WriteReq

This flag is used to reflect the setting of each control bit (The IntMode bit, the ParaCh bit, the StaCal bit and the Measure bit in this register) to this LSI operation.

When the WriteReq bit is set to “1”, all the settings of the IntMode bit, the ParaCh bit, the StaCal bit and the Measure bit are reflected to the LSI operation. **When the WriteReq bit is set to “0”, all the settings of the IntMode bit, the ParaCh bit, the StaCal bit and the Measure bit are not reflected to the LSI operation at all.**

After reflecting the settings to the LSI operation, the WriteReq bit returns to “0” automatically.

0: Not reflect all the settings of the IntMode bit, the ParaCh bit, the StaCal bit and the Measure bit (Initial value)

1: Reflect all the settings of the IntMode bit, the ParaCh bit, the StaCal bit and the Measure bit

Rsvd6 to Rsvd4

Set these bits to “0”.

IntMode

Interval mode flag. For the further details of each mode, refer to “2.1 Description of Functions”.

0: Operate in sleep mode

1: Operate in interval mode (Initial value)

ParaCh

This flag is used to request this LSI to change settings (In other words, this flag is used to request this LSI to reflect register settings to the LSI operation).

To reflect the settings of individual registers (Address=0x00 to 0x19 and 0x30 to 0x3D) other than the Control 1 Register to the LSI operation, it is necessary to set the ParaCh bit to “1”.

When completing reflecting the settings to the LSI operation, the contents of the Error Status Register (Address=0x2C), the Error Channel Status 1 Register (Address=0x2D) and the Error Channel Status 2 Register (Address=0x2E) are all cleared to 0x00, and INTOUT is negated. Counters inside this LSI which are used for the processing of dynamic offset calibration, the processing of debounce are all cleared. And then, the ParaCh bit returns to “0” automatically.

0: No request for changing parameters (Initial value)

1: Changing parameters is requested

Continued to the next page.

Continued from the previous page.

StaCal

The flag is used to request this LSI to perform static offset calibration.

After static offset calibration has completed, the StaCal bit returns to “0” automatically.

The static offset calibration performs offset adjustment of the capacitance A/D-converter for parasitism capacitance of each input channel (Cin0 to Cin15) and decides a most suitable plus side’s offset capacitance value and minus side’s offset capacitance value corresponding to each channel.

0: No request for static offset calibration

1: Static offset calibration is requested (Initial value)

Note: When the microcontroller writes the data in which the WriteReq bit is set to “1” and the StaCal bit is set to “0” (For example, 0x88, 0x80) into the Control 1 Register through the I²C/SPI interface during the time that the LC717A10 is performing the static offset calibration, the LC717A10 doesn’t cancel the execution of the static offset calibration immediately. It runs the current static offset calibration processing to the end.

Measure

Measurement flag. It is normally fixed to “1” unless you have specific reasons.

When the setting that LSI does not measure, the contents of the Result Data 1 Register (Address=0x2A) and the Result Data 2 Register (Address=0x2B) are all cleared to 0x00.

0: No measurement

1: Measurement (Initial value)

Note: Processing order (The processing order when all of the WriteReq bit, the ParaCh bit, the StaCal bit and the Measure bit are set to “1” at a time is as follows)

- (1) Control bit reflection processing (WriteReq bit).
After reflecting to this LSI, the WriteReq bit returns to “0”.
- (2) Parameter updating processing (ParaCh bit).
After the processing, the ParaCh bit returns to “0”.
- (3) Static offset calibration processing (StaCal bit).
After the processing, the StaCal bit returns to “0”.
- (4) Measurement processing (Measure bit).
This LSI repeats the measurement processing.

●Average Count Register

Address	Bit	7	6	5	4	3	2	1	0
0x30	Name	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
	Reset	0	1	0	0	0	0	0	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify average counts of measurement data. Don’t set any value but 8, 16, 32, 64 and 128 times. The initial value is 64 times (0x40).

AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Average counts of the measurement data
0	0	0	0	1	0	0	0	8 times
0	0	0	1	0	0	0	0	16 times
0	0	1	0	0	0	0	0	32 times
0	1	0	0	0	0	0	0	64 times (Initial value)
1	0	0	0	0	0	0	0	128 times
Other than those above								Prohibited

●Filter Parameter Register

Address	Bit	7	6	5	4	3	2	1	0
0x31	Name	FP2_3	FP2_2	FP2_1	FP2_0	FP1_3	FP1_2	FP1_1	FP1_0
	Reset	0	0	0	0	0	1	0	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify filter parameter (In other words, noise suppression parameter). The initial value is 0x04.

FP1_[3:0]
Filter parameter 1

FP2_[3:0]
Filter parameter 2

●Debounce Count 1 Register

Address	Bit	7	6	5	4	3	2	1	0
0x32	Name	DCT1_7	DCT1_6	DCT1_5	DCT1_4	DCT1_3	DCT1_2	DCT1_1	DCT1_0
	Reset	0	0	0	0	0	0	1	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify the number of debounce count, which is the parameter value used for “ON”/“OFF” judgment by this LSI when the touch decision at each channel changes from “OFF” state to “ON” state. By setting properly the debounce count, you can prevent chattering of a switch. If you would like to set the debounce count to N1, you have to set this register to (N1-1). The initial value is 3 counts (0x02).

The condition for the result of “ON”/“OFF” judgment to change from “OFF” state to “ON” state at each channel:
When the number of consecutive measurement points that measurement data is greater than or equal to the threshold for “ON”/“OFF” judgment becomes equal to the number of debounce count specified by the Debounce Count 1 Register, the result of “ON”/“OFF” judgment changes to be “ON”.

DCT1_7	DCT1_6	DCT1_5	DCT1_4	DCT1_3	DCT1_2	DCT1_1	DCT1_0	Debounce count (Change from “OFF” state to “ON” state)
0	0	0	0	0	0	0	0	1 count
0	0	0	0	0	0	0	1	2 counts
0	0	0	0	0	0	1	0	3 counts (Initial value)
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	1	1	0	255 counts
1	1	1	1	1	1	1	1	Prohibited

AND9594/D

●Debounce Count 2 Register

Address 0x33	Bit	7	6	5	4	3	2	1	0
	Name	DCT2_7	DCT2_6	DCT2_5	DCT2_4	DCT2_3	DCT2_2	DCT2_1	DCT2_0
Reset	0	0	0	0	0	0	0	1	0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify the number of debounce count, which is the parameter value used for “ON”/“OFF” judgment by this LSI when the touch decision at each channel changes from “ON” state to “OFF” state. By setting properly the debounce count, you can prevent chattering of a switch. If you would like to set the debounce count to N2, you have to set this register to (N2-1). The initial value is 3 counts. (0x02).

Specify a value greater than or equal to the value specified by Debounce Count 1 Register (Address=0x32).

The condition for the result of “ON”/“OFF” judgment to change from “ON” state to “OFF” state at each channel:

When the number of consecutive measurement points that measurement data is less than the threshold for “ON”/“OFF” judgment becomes equal to the number of debounce count specified by the Debounce Count 2 Register, the result of “ON”/“OFF” judgment changes to be “OFF”.

This setting value is also used by this LSI when moving from short interval mode to long interval mode. For more information, refer to “2.1 Description of Functions”.

DCT2_7	DCT2_6	DCT2_5	DCT2_4	DCT2_3	DCT2_2	DCT2_1	DCT2_0	Debounce count (Change from “ON” state to “OFF” state)
0	0	0	0	0	0	0	0	1 count
0	0	0	0	0	0	0	1	2 counts
0	0	0	0	0	0	1	0	3 counts (Initial value)
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	1	1	0	255 counts
1	1	1	1	1	1	1	1	Prohibited

●Short Interval Time Register

Address 0x34	Bit	7	6	5	4	3	2	1	0
	Name	SIVAL7	SIVAL6	SIVAL5	SIVAL4	SIVAL3	SIVAL2	SIVAL1	SIVAL0
Reset	0	0	0	0	0	0	1	0	1
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify short interval time during interval mode. “Short interval time” is the interval between the end time of one measurement and the start time of the next measurement when the touch may be detected. The value of short interval time can be specified from 0 ms to 255 ms in units of 1 ms (Typ).

When sleep mode is selected, it operates with the short interval time of 0 ms regardless of this setting.

The initial value is 5 ms (Typ) (0x05).

For more information about the Short Interval Mode, refer to “2.1 Description of Functions”.

SIVAL7	SIVAL6	SIVAL5	SIVAL4	SIVAL3	SIVAL2	SIVAL1	SIVAL0	Short interval time
0	0	0	0	0	0	0	0	0 ms
0	0	0	0	0	0	0	1	1 ms
0	0	0	0	0	0	1	0	2 ms
0	0	0	0	0	0	1	1	3 ms
0	0	0	0	0	1	0	0	4 ms
0	0	0	0	0	1	0	1	5 ms (Initial value)
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	1	1	1	255 ms

●Long Interval Time Register

Address	Bit	7	6	5	4	3	2	1	0
0x35	Name	LIVAL7	LIVAL6	LIVAL5	LIVAL4	LIVAL3	LIVAL2	LIVAL1	LIVAL0
	Reset	0	0	0	0	0	0	0	1
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify long interval time during interval mode. “Long interval time” is the interval between the end time of one measurement and the start time of the next measurement when the touch is not detected at all. The value of long interval time can be specified from 0 ms to 355 ms in units of 1 ms (Typ).

The base time for long interval time can be selected from either 100 ms (Initial value) or 0 ms by specifying the LIVALB bit in the Measurement Mode 1 Register (Address=0x3A).

When sleep mode is selected, it operates with the long interval time of 0 ms regardless of this setting.

The initial value is 100 ms + 1 ms (0x01) = 101 ms (Typ).

If this register is set to 0x00, both long interval time and short interval time are set to 0 ms.

For more information about the Long Interval Mode, refer to “2.1 Description of Functions”.

LIVAL7	LIVAL6	LIVAL5	LIVAL4	LIVAL3	LIVAL2	LIVAL1	LIVAL0	Long interval time	
								LIVALB=0	LIVALB=1
0	0	0	0	0	0	0	0	No interval (0 ms)	No interval (0 ms)
0	0	0	0	0	0	0	1	101 ms (Initial value)	1 ms
0	0	0	0	0	0	1	0	102 ms	2 ms
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	1	1	1	355 ms	255 ms

●Short Interval Dynamic OffCal Cycle Register

Address	Bit	7	6	5	4	3	2	1	0
0x36	Name	DCYC7	DCYC6	DCYC5	DCYC4	DCYC3	DCYC2	DCYC1	DCYC0
	Reset	0	0	0	0	0	1	1	1
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

When this LSI operates with short interval, it makes the judgment whether or not dynamic offset calibration is performed once every *N* measurement points. This register is used to specify the number *N*. The initial value is once every 7 measurements (0x07).

A value of “Short interval time” usually differs from that of “long interval time”. So the measurement-to-measurement interval in short interval mode usually differs from that in long interval mode. Therefore, it is necessary for a microcontroller to write an appropriate value to this register so that a value of interval to perform the processing of dynamic offset calibration in short interval mode may be as equal as possible to that in long interval mode.

When this LSI operates with the long interval time, regardless of this register setting, the judgment is made at every measurement.

When sleep mode is selected, be sure to set this register to 0x01.

When this register is set to 0x00, dynamic offset calibration is not performed at all regardless of other register settings.

For more information, refer to “2.2.2 Description of Dynamic Offset Calibration”.

DCYC7	DCYC6	DCYC5	DCYC4	DCYC3	DCYC2	DCYC1	DCYC0	The judgment cycle number for dynamic offset calibration execution in short interval mode
0	0	0	0	0	0	0	0	Dynamic offset calibration is not performed.
0	0	0	0	0	0	0	1	Once every 1 measurement
0	0	0	0	0	0	1	0	Once every 2 measurements
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	1	1	1	Once every 255 measurements

●Dynamic OffCal Count Plus Register

Address	Bit	7	6	5	4	3	2	1	0
0x37	Name	DCALP7	DCALP6	DCALP5	DCALP4	DCALP3	DCALP2	DCALP1	DCALP0
	Reset	0	0	0	0	0	0	1	1
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify the plus side's dynamic offset calibration execution count number. The initial value is 0x03. This means that the number of consecutive measurement points is 24 times, which is equal to the register setting value (0x03) times 8.

This LSI performs dynamic offset calibration judgment processing in the following cases by the setting of the PDCLP bit in the Measurement Mode 1 Register (Address=0x3A).

Case1: When the PDCLP bit is set to "0" and all channels are touch OFF states (OFF detection of all channels), this LSI carries out judgment processing.

Case2: When the PDCLP bit is set to "1", this LSI carries out judgment processing for a channel of touch OFF state because dynamic offset calibration is determined and performed at each individual enabled channel.

When the long interval mode, the LSI judges the dynamic offset calibration execution at every measurement. When the measurement data of the judgment processing (Judgment points) under the constant number of times specified by this register (Address=0x37) continues in the dynamic offset calibration operating range (4 to CinX threshold), this LSI executes the dynamic offset calibration.

When the short interval mode, the LSI judges the dynamic offset calibration execution at measurement every execution cycle of the number of times appointed by the Short Interval Dynamic OffCal Cycle Register (Address=0x36). When the measurement data of the judgment processing (Judgment points) under the constant number of times specified by this register (Address=0x37) continues in the plus side's dynamic offset calibration operating range (4 to CinX threshold), this LSI executes the dynamic offset calibration.

When this register is set to 0x00, dynamic offset calibration is not performed at all regardless of other register settings.

For more information, refer to "2.2.2 Description of Dynamic Offset Calibration".

DCALP7	DCALP6	DCALP5	DCALP4	DCALP3	DCALP2	DCALP1	DCALP0	The plus side's dynamic offset calibration execution count number	
								Short interval mode	Long interval mode
0	0	0	0	0	0	0	0	Dynamic offset calibration is not performed.	Dynamic offset calibration is not performed.
0	0	0	0	0	0	0	1	(8 × Execution cycle) points	8 points
0	0	0	0	0	0	1	0	(16 × Execution cycle) points	16 points
0	0	0	0	0	0	1	1	(24 × Execution cycle) points	24 points (Initial value)
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	1	1	1	(2040 × Execution cycle) points	2040 points

AND9594/D

●Dynamic OffCal Count Minus Register

Address	Bit	7	6	5	4	3	2	1	0
	0x38	Name	DCALM7	DCALM6	DCALM5	DCALM4	DCALM3	DCALM2	DCALM1
Reset		0	0	0	0	0	0	1	1
R/W		RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify the minus side's dynamic offset calibration execution count number. The LSI judges the minus side's dynamic offset calibration execution every use channel regardless of a touch ON/OFF state. The initial value is 0x03, which means that the number of consecutive measurement points is 3 times.

When the long interval mode, the LSI judges the dynamic offset calibration execution at every measurement. When the measurement data of the judgment processing (Judgment points) under the constant number of times specified by this register (Address=0x38) continues in the dynamic offset calibration operating range (-128 to -4), this LSI executes the dynamic offset calibration.

When the short interval mode, the LSI judges the dynamic offset calibration execution at measurement every execution cycle of the number of times appointed by the Short Interval Dynamic OffCal Cycle Register (Address=0x36). When the measurement data of the judgment processing (Judgment points) under the constant number of times specified by this register (Address=0x38) continues in the minus side's dynamic offset calibration operating range (-128 to -4), this LSI executes the dynamic offset calibration.

When this register is set to 0x00, dynamic offset calibration is not performed at all regardless of other register settings.

For more information, refer to "2.2.2 Description of Dynamic Offset Calibration".

DCALM7	DCALM6	DCALM5	DCALM4	DCALM3	DCALM2	DCALM1	DCALM0	The minus side's dynamic offset calibration execution count number	
								Short interval mode	Long interval mode
0	0	0	0	0	0	0	0	Dynamic offset calibration is not performed.	Dynamic offset calibration is not performed.
0	0	0	0	0	0	0	1	(1 × Execution cycle) points	1 point
0	0	0	0	0	0	1	0	(2 × Execution cycle) points	2 points
0	0	0	0	0	0	1	1	(3 × Execution cycle) points	3 points (Initial value)
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	1	1	1	(255 × Execution cycle) points	255 points

●Static OffCal CDAC Base Register

Address	Bit	7	6	5	4	3	2	1	0
	0x39	Name	DACB7	DACB6	DACB5	DACB4	DACB3	DACB2	DACB1
Reset		1	0	0	0	0	0	0	0
R/W		RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify the reference capacitance for static offset calibration.

If the reference value (In other words, the measurement data when the touch ON is not detected at all) might not coincide with the vicinity of the center of measurement range (that is, almost 0) after static offset calibration, this problem may be resolved by changing the value of reference capacitance.

Don't set any number but 0x20, 0x40, 0x80. The initial value is 4 pF (0x80).

DACB7	DACB6	DACB5	DACB4	DACB3	DACB2	DACB1	DACB0	The reference capacitance used when static offset calibration is executed.
0	0	1	0	0	0	0	0	1 pF
0	1	0	0	0	0	0	0	2 pF
1	0	0	0	0	0	0	0	4 pF (Initial value)
Other than those above								Prohibited

●Measurement Mode 1 Register

Bit	7	6	5	4	3	2	1	0	
Address	Name	INTMD1	INTMD2	PDCLP	TOFFTH	LIVALB	Rsvd2	Rsvd1	Rsvd0
0x3A	Reset	0	0	0	0	0	0	0	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify measurement mode.

INTMD1

This bit is used to specify the output mode of INTOUT at the end of each measurement. By specifying the INTMD1 bit, a microcontroller can select how to assert the INTOUT signal.

0: INTOUT is always asserted after every measurement (Initial value).

1: INTOUT is asserted after every measurement while touch ON of at least one channel is detected. Otherwise (That is, touch is not detected), INTOUT is not asserted.

INTMD2

This bit is used to specify the output mode of INTOUT either at the end of either interval processing (In the case of interval mode) or at the wake-up of this LSI (In the case of sleep mode). By specifying the INTMD2 bit, a microcontroller can select how to negate the INTOUT signal.

0: INTOUT is not negated automatically (Initial value).

1: In the case of interval mode, this LSI automatically negates INTOUT immediately after the end of interval processing. In the case of sleep mode, this LSI automatically negates INTOUT immediately after a microcontroller wakes up this LSI.

PDCLP

The flag to select the method of dynamic offset calibration for the plus range of measurement data.

0: Dynamic offset calibration for the plus range is performed only when measurement value of all touches are less than or equal to the threshold value. (Initial value)

1: Dynamic offset calibration for the plus range is determined and performed at each individual enabled channel.

Note: Only if measurement data corresponding to a switch doesn't vary at all when another switch is touched (In other words, if each switch is sufficiently far from each other), this PDCLP bit can be set to "1".

TOFFTH

This bit is used to specify the calculation means from peak of the dynamic threshold when moving from ON state to OFF state.

However, when the value of the CinX Threshold Register (Address=0x0A to 0x19) is larger than the selected peak value, the value of the CinX Threshold Register is used as the actual threshold value.

For more information, refer to "2.2.4 Description of Dynamic Threshold Control When Moving from ON State to OFF State".

0: 3/4 of the peak of the measurement data while detecting the touch (Initial value)

1: 1/2 of the peak of the measurement data while detecting the touch

LIVALB

The flag for selecting the base time for long interval

0: The base time for long interval is 100 ms (Typ) (Initial value)

1: The base time for long interval is 0 ms

Rsvd2 to Rsvd0

Set these bits to "0".

The explanation about the INTOUT signal:

In order to clear the INTOUT signal when the INTMD2 bit is set to "0", set the IntOut bit in the Control 2 Register (Address=0x40) to "0" during interval processing (In the case of interval mode) or during sleep period (In the case of sleep mode).

When the INTMD2 bit is set to "1", a microcontroller does not need to clear the INTOUT signal. It can check the timing of the measurement end by detecting rising edge of INTOUT signal.

Continued to the next page.

Continued from the previous page.

INTOUT output mode

INTMD1	INTMD2	INTOUT output mode
0	0	This LSI asserts INTOUT after every measurement. To clear INTOUT signal, set the IntOut bit in the Control 2 Register (Address=0x40) to "0".
0	1	This LSI asserts INTOUT after every measurement. After that, this LSI automatically negates INTOUT signal either at the end of either interval processing (In the case of interval mode) or at the wake-up of this LSI (In the case of sleep mode). -> INTOUT signal is pulse output.
1	0	Only while touch ON of at least one channel is detected, this LSI asserts INTOUT after every measurement. To clear INTOUT signal, set the IntOut bit in the Control 2 Register (Address=0x40) to "0".
1	1	Only while touch ON of at least one channel is detected, this LSI asserts INTOUT after every measurement. After that, the LSI automatically negates INTOUT signal either at the end of either interval processing (In the case of interval mode) or at the wake-up of this LSI (In the case of sleep mode). -> INTOUT signal is pulse output.

When INTOUT signal is asserted, INTOUT output level is "High". When it is negated, INTOUT output level is "Low".

<Conditions that INTOUT signal is asserted>

- When this LSI completes the initialization processing, INTOUT is asserted.
- When the INTMD1 bit in the Measurement Mode 1 Register (Address=0x3A) is set to "0", INTOUT signal is asserted at each timing of the end of the measurement of all the enabled channels. However, when static offset calibration has been completed, INTOUT is not asserted.
- When the INTMD1 bit in the Measurement Mode 1 Register (Address=0x3A) is set to "1", INTOUT is asserted after every measurement while touch ON of at least one channel is detected. Otherwise (That is, touch ON is not detected), INTOUT is not asserted.
- When a microcontroller sets the IntOut bit in the Control 2 Register (Address=0x40) to "1", INTOUT signal is asserted.

<Conditions that INTOUT signal is negated>

- When the internal reset of this LSI was released, INTOUT signal is negated.
- When the LSI completes parameter updating processing after the WriteReq bit and ParaCh bit in the Control 1 Register (Address=0x2F) are set to "1", INTOUT signal is negated.
- When the INTMD2 bit in the Measurement Mode 1 Register (Address=0x3A) is set to "1", this LSI automatically negates INTOUT immediately after the end of interval processing in the case of interval mode, otherwise this LSI automatically negates INTOUT immediately after this LSI wakes up by a microcontroller in the case of sleep mode.
- When a microcontroller sets the IntOut bit in the Control 2 Register (Address=0x40) to "0" during interval processing (In the case of interval mode) or during sleep period (In the case of sleep mode), INTOUT signal is negated.

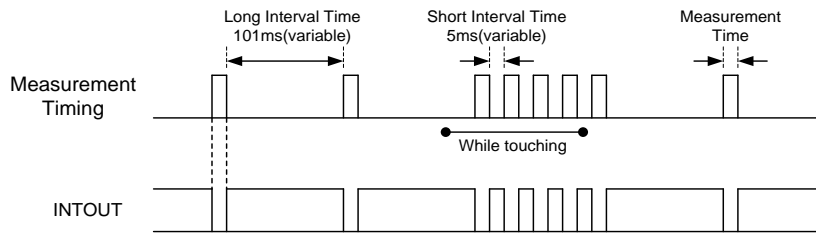


Fig.1-1 When INTMD1=0 and INTMD2=1

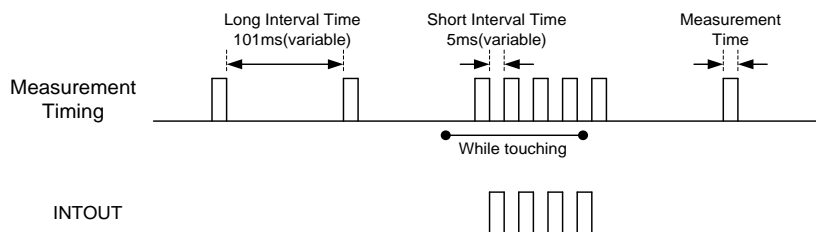


Fig.1-2 When INTMD1=1 and INTMD2=1

AND9594/D

●Measurement Mode 2 Register

Address	Bit	7	6	5	4	3	2	1	0
0x3B	Name	Rsvd7	Rsvd6	Rsvd5	CIN8CINP2	CIN0CINP2	CIN8CINP	CIN0CINP	Rsvd0
	Reset	0	0	0	0	0	0	0	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify measurement mode.

Rsvd7 to Rsvd5

Set these bits to "0".

CIN8CINP2

This bit is used to select what should be connected to the input of 1st amplifier (+) side when Cin8 to Cin15 are measured.

0: When Cin8 to Cin15 are measured, only Cref is connected to the input of 1st amplifier (+) side.
(Initial value)

1: When Cin8 to Cin15 are measured, both Cref and CrefAdd are connected to the input of 1st amplifier (+) side.

CIN0CINP2

This bit is used to select what should be connected to the input of 1st amplifier (+) side when Cin0 to Cin7 are measured.

0: When Cin0 to Cin7 are measured, only Cref is connected to the input of 1st amplifier (+) side.
(Initial value)

1: When Cin0 to Cin7 are measured, both Cref and CrefAdd are connected to the input of 1st amplifier (+) side.

CIN8CINP

This bit is fixed to "0".

CIN0CINP

This bit is fixed to "0".

Rsvd0

Set these bits to "0".

●Long Interval Mode Start Count Register

Address	Bit	7	6	5	4	3	2	1	0
0x3C	Name	LIMSC_7	LIMSC_6	LIMSC_5	LIMSC_4	LIMSC_3	LIMSC_2	LIMSC_1	LIMSC_0
	Reset	0	0	0	0	0	0	0	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify the period (Measurement cycles) for short interval mode before stating long interval mode, after "Touch OFF" of all enabled channels are judged.

The period value is specified in units of 4 measurement points. The initial value is 0 point (0x00).

LIMSC_7	LIMSC_6	LIMSC_5	LIMSC_4	LIMSC_3	LIMSC_2	LIMSC_1	LIMSC_0	Cycle before start of long interval
0	0	0	0	0	0	0	0	0 point (Initial value)
0	0	0	0	0	0	0	1	4 points
0	0	0	0	0	0	1	0	8 points
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	1	1	1	0	1016 points
1	1	1	1	1	1	1	1	1020 points

Calculation formula of time for short interval mode before stating long interval mode after all channels became the "Touch OFF" judgment is as follows:

$$(Measurement\ time + Short\ interval\ time) \times (Setting\ value\ of\ the\ long\ interval\ mode\ start\ count) [ms]$$

AND9594/D

●Cin 1st Gain Adjust Register

Address	Bit	7	6	5	4	3	2	1	0
0x3D	Name	Gain8_F3	Gain8_F2	Gain8_F1	Gain8_F0	Gain0_F3	Gain0_F2	Gain0_F1	Gain0_F0
	Reset	0	0	0	0	0	0	0	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify the 1st-amplifier gain setting of each channel.

The initial values are 1600 (0x00). Normally, specify the gain of the 1st-amplifier at the minimum.

The gain of 2nd-amplifier is specified with the CinO/E 2nd Gain Register (Address=0x02 to 0x09).

Gain0_F0 to 3: This field is used to specify gain of the 1st-amplifier from Cin0 to Cin7 by the lower 4 bits. The gain setting of the 1st-amplifier from Cin0 to Cin7 is set in common. It can't be set individually.

Gain8_F0 to 3: This field is used to specify gain of the 1st-amplifier from Cin8 to Cin15 by the upper 4 bits. The gain setting of the 1st-amplifier from Cin8 to Cin15 is set in common. It can't be set individually.

Gain0_F3	Gain0_F2	Gain0_F1	Gain0_F0	1st-amplifier gain Cf [fF] (LSI design typical value)
Gain8_F3	Gain8_F2	Gain8_F1	Gain8_F0	
0	0	0	0	1600 (Minimum gain setting) (Initial value)
0	0	0	1	1500
0	0	1	0	1400
0	0	1	1	1300
0	1	0	0	1200
0	1	0	1	1100
0	1	1	0	1000
0	1	1	1	900
1	0	0	0	800
1	0	0	1	700
1	0	1	0	600
1	0	1	1	500
1	1	0	0	400
1	1	0	1	300
1	1	1	0	200
1	1	1	1	100 (Maximum gain setting)

<Calculation formula>

Output voltage of a two-stage amplifier

$$\text{Output voltage of the 1st-amplifier: } \Delta V_1 = (\Delta C / C_f) \times V_{CDRV} \quad \Delta V_1 < 0.8 \times V_{DD}$$

$$\text{Output voltage of the 2nd-amplifier: } \Delta V_2 = \Delta V_1 \times ga_2 \quad \Delta V_2 < 0.8 \times V_{DD}$$

ΔC : Change in input capacitance (Capacitance change when touch)

C_f : The 1st-amplifier gain setting (Feedback capacitance in the LSI)

V_{CDRV} : High output voltage of Cdrv (= V_{DD})

ga_2 : The 2nd-amplifier gain setting (Times)

●Control 2 Register

Bit	7	6	5	4	3	2	1	0	
Address	Name	SoftRst	Rsvd6	Hold	Rsvd4	Rsvd3	Rsvd2	IntOut	WakeUp
0x40	Reset	0	0	0	0	0	0	0	0
	R/W	RW	RW	RW	RW	RW	RW	RW	RW

This register is used to specify operation of the LSI processing.

Note: When writing a certain value to this register either to clear INTOUT output or to wake up this LSI that is sleeping, do it during interval processing(In the case of interval mode) or during sleep period(In the case of sleep mode). If it can't be guaranteed that a microcontroller writes during interval processing, select the Interval Mode and select the mode that INTOUT is negated automatically (In other words, set the IntMode bit in Control 1 Register (Address=0x2F) to "1", and set the INTMD2 bit in Measurement Mode 1 Register (Address=0x3A) to "1").

SoftRst

Software-reset request bit. When this bit is set to "1", software-reset is performed in the LSI. After software-reset is performed, it is returned to "0" automatically.

The software-reset starts operation from sequence just after released internal reset in the "3.1 An Example of Operation Sequence After External Reset".

- 0: Normal operation (Initial value)
- 1: Software-reset.

Rsvd6

Set this bit to "0".

Hold

This bit is Fixed to "0".

Rsvd2 to Rsvd4

Set these bits to "0".

IntOut

INTOUT pin output control bit. When clearing INTOUT output (That is, when setting the output from INTOUT pin to "Low"), set this bit to "0".

When the microcontroller clears the INTOUT output to "Low", set this bit to "0" while this LSI is in interval operation period or in sleep operation period.

When INTOUT is asserted, output level of INTOUT is "High". When INTOUT is negated, it is "Low".

- 0: Setting INTOUT pin to "Low" (Initial value)
- 1: Setting INTOUT pin to "High"

WakeUp

Wake up bit. When the microcontroller sets this bit to "1", it can wake up this LSI which is sleeping. Immediately after this LSI has been woken up, this bit returns to "0" automatically.

- 0: Normal operation (Initial value)
- 1: Wake up the LSI which is sleeping.

●CDAC Offset Plus Register

Address	Bit	7	6	5	4	3	2	1	0
0x6C	Name	CdacP7	CdacP6	CdacP5	CdacP4	CdacP3	CdacP2	CdacP1	CdacP0
	Reset	X	X	X	X	X	X	X	X
	R/W	R	R	R	R	R	R	R	R

This register is used to read out the plus side's offset capacitance value (CDACP) coordinated by the offset calibration execution. It is 8bit data from 0x00 to 0xFF.

The measurement is performed in the order from Cin0 to Cin15.

This LSI overwrote this register (Address=0x6C) with a CDACP value decided by the offset calibration execution, therefore only a CDACP value of the last enabled channel is maintained at the time of the offset calibration completion.

$$\text{Plus side's offset capacitance value [fF] (Typ)} = (\text{Value of the CDAC Offset Plus Register}) \times 31.25 \text{ [fF]}$$

●CDAC Offset Minus Register

Address	Bit	7	6	5	4	3	2	1	0
0x6D	Name	CdacM7	CdacM6	CdacM5	CdacM4	CdacM3	CdacM2	CdacM1	CdacM0
	Reset	X	X	X	X	X	X	X	X
	R/W	R	R	R	R	R	R	R	R

This register is used to read out the minus side's offset capacitance value (CDACM) coordinated by the offset calibration execution. It is 8bit data from 0x00 to 0xFF.

The measurement is performed in the order from Cin0 to Cin15.

This LSI overwrote this register (Address=0x6D) with a CDACM value decided by the offset calibration execution, therefore only a CDACM value of the last enabled channel is maintained at the time of the offset calibration completion.

$$\text{Minus side's offset capacitance value [fF] (Typ)} = (\text{Value of the CDAC Offset Minus Register}) \times 31.25 \text{ [fF]}$$

■ Reading sequence of the offset capacitance value of plural channels

The CDAC value is read every one channel by the following sequences.

- (1) Set only one enabled channel by the Use Channel 1 Register (Address=0x00) and the Use Channel 2 Register (Address=0x01).
- (2) Write 0x8F to the Control 1 Register (Address=0x2F) for starting static offset calibration and measurement after having reflected new setting.
- (3) Wait until static offset calibration processing is completed.
- (4) Read the Error Status Register (Address=0x2C), and confirm that a calibration error does not occur.
- (5) Read the CDAC Offset Plus Register (Address=0x6C) and the CDAC Offset Minus Register (Address=0x6D), and confirm an offset capacitance value.
- (6) Repeat from (1) to (5) for all channels which want to acquire an offset capacitance value.

AND9594/D

●SLAVE Address Register

Address 0x7F	Bit	7	6	5	4	3	2	1	0
	Name	Rsvd7	Slave6	Slave5	Slave4	Slave3	Slave2	Slave1	SA0
Reset (SA1=Lo)	0	0	0	1	0	1	1	X	
Reset (SA1=Hi)	0	0	0	1	1	0	0	X	
R/W	R	R	R	R	R	R	R	R	R

I²C bus slave address (7bit) can be read out from this register by a microcontroller.

When the input to the SA0/SO pin is “High”, the SA0 bit is “1”. When the input to the SA0/SO pin is “Low”, the SA0 bit is “0”.

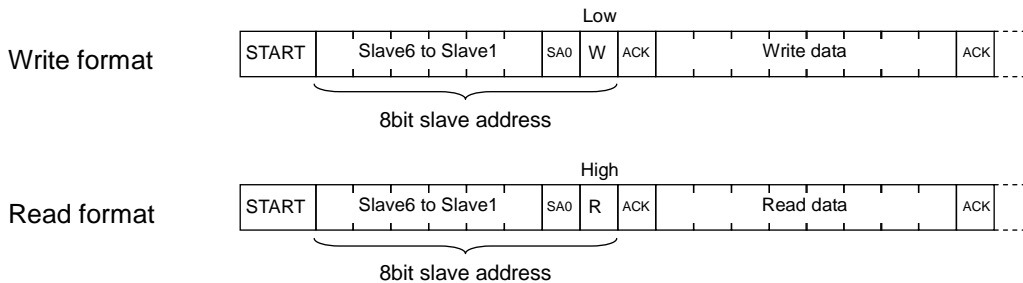
As the input to the SA0/SO pin is reflected to the SA0 bit, the SA0 bit value can be changed at any time. However, basically the bit always has to be fixed to either “0” or “1”.

Rsvd7

Set this bit to “0”.

SA1 pin input	SA0/SO pin input	7bit slave address (Slave6 to Slave1+SA0)	Binary notation	8bit slave address
Low	Low	0x16	00101100b (Write)	0x2C
			00101101b (Read)	0x2D
Low	High	0x17	00101110b (Write)	0x2E
			00101111b (Read)	0x2F
High	Low	0x18	00110000b (Write)	0x30
			00110001b (Read)	0x31
High	High	0x19	00110010b (Write)	0x32
			00110011b (Read)	0x33

I²C bus communication format



AND9594/D

1.4 Recommended Setting

The recommended setting value in a recommendation pattern is shown in the table below.
For this recommendation pattern, refer to an application note AND9197 (Another document).

Register Address	Recommended Value	Register Name	Description
0x00	-	Use Channel 1 Register	*1
0x01	-	Use Channel 2 Register	
0x02	0x55	Cin1/0 2nd Gain Register	- Cin0 2nd Gain = 6 times *2 - Cin1 2nd Gain = 6 times *2
0x03	0x55	Cin3/2 2nd Gain Register	- Cin2 2nd Gain = 6 times *2 - Cin3 2nd Gain = 6 times *2
0x04	0x55	Cin5/4 2nd Gain Register	- Cin4 2nd Gain = 6 times *2 - Cin5 2nd Gain = 6 times *2
0x05	0x55	Cin7/6 2nd Gain Register	- Cin6 2nd Gain = 6 times *2 - Cin7 2nd Gain = 6 times *2
0x06	0x55	Cin9/8 2nd Gain Register	- Cin8 2nd Gain = 6 times *2 - Cin9 2nd Gain = 6 times *2
0x07	0x55	Cin11/10 2nd Gain Register	- Cin10 2nd Gain = 6 times *2 - Cin11 2nd Gain = 6 times *2
0x08	0x55	Cin13/12 2nd Gain Register	- Cin12 2nd Gain = 6 times *2 - Cin13 2nd Gain = 6 times *2
0x09	0x55	Cin15/14 2nd Gain Register	- Cin14 2nd Gain = 6 times *2 - Cin15 2nd Gain = 6 times *2
0x0A	0x0A	Cin0 Threshold Register	Cin0 threshold = 10
0x0B	0x0A	Cin1 Threshold Register	Cin1 threshold = 10
0x0C	0x0A	Cin2 Threshold Register	Cin2 threshold = 10
0x0D	0x0A	Cin3 Threshold Register	Cin3 threshold = 10
0x0E	0x0A	Cin4 Threshold Register	Cin4 threshold = 10
0x0F	0x0A	Cin5 Threshold Register	Cin5 threshold = 10
0x10	0x0A	Cin6 Threshold Register	Cin6 threshold = 10
0x11	0x0A	Cin7 Threshold Register	Cin7 threshold = 10
0x12	0x0A	Cin8 Threshold Register	Cin8 threshold = 10
0x13	0x0A	Cin9 Threshold Register	Cin9 threshold = 10
0x14	0x0A	Cin10 Threshold Register	Cin10 threshold = 10
0x15	0x0A	Cin11 Threshold Register	Cin11 threshold = 10
0x16	0x0A	Cin12 Threshold Register	Cin12 threshold = 10
0x17	0x0A	Cin13 Threshold Register	Cin13 threshold = 10
0x18	0x0A	Cin14 Threshold Register	Cin14 threshold = 10
0x19	0x0A	Cin15 Threshold Register	Cin15 threshold = 10
0x30	0x80	Average Count Register	Average counts = 128 times
0x31	0x0C	Filter Parameter Register	- Filter parameter 1 = 12 - Filter parameter 2 = 0
0x32	0x01	Debounce Count 1 Register	Debounce count (OFF to ON) = 2 counts
0x33	0x01	Debounce Count 2 Register	Debounce count(ON to OFF) = 2 counts
0x34	0x05	Short Interval Time Register	Short interval time = 5 ms
0x35	0x01	Long Interval Time Register	Long interval time = 101 ms
0x36	0x03	Short Interval Dynamic OffCal Cycle Register	The judgment cycle number for dynamic offset calibration execution in short interval mode = once every 3 measurements
0x37	0x03	Dynamic OffCal Count Plus Register	The plus side's dynamic offset calibration execution count number = 24 points
0x38	0x03	Dynamic OffCal Count Minus Register	The minus side's dynamic offset calibration execution count number = 3 points

Continued to the next page.

AND9594/D

Continued from the previous page.

Register Address	Recommended Value	Register Name	Description
0x39	0x80	Static OffCal CDAC Base Register	The reference capacity used at static offset calibration = $4 \text{ pF} \cdot *3$
0x3A	0x10	Measurement Mode 1 Register	<ul style="list-style-type: none"> - INTOUT is always asserted after every measurement - INTOUT is not negated automatically - The plus's side dynamic offset calibration is performed when all enabled channels are OFF - Touch OFF threshold = 1/2 of peak of the value while touch is detected - Long interval base time = 100 ms
0x3B	0x00	Measurement Mode 2 Register	<ul style="list-style-type: none"> - Cin8 to Cin15 measurement = use only Cref - Cin0 to Cin7 measurement = use only Cref
0x3C	0x05	Long Interval Mode Start Count Register	Cycle before start of long interval = 20 points
0x3D	0x00	Cin 1st Gain Adjust Register	<ul style="list-style-type: none"> - Cin8 to Cin15 1st Gain = 1600(Min) - Cin0 to Cin7 1st Gain = 1600(Min)

*1 Set according to the number of the enabled channels.

*2 Adjust the 2nd Gain according to an application note AND9197 (Another document).

*3 Adjust the setting value depending on capacitance between Cin (Cref) and Cdrv according to an application note AND9197 (Another document).

Chapter 2 FUNCTIONS

2.1 Description of Functions

2.1.1 Interval Mode and Sleep Mode

- Interval Mode (Recommended)

This is the mode that the LSI performs interval processing with either short interval or long interval after measurement processing. This mode is chosen when the IntMode bit in the Control 1 Register (Address=0x2F) is set to "1".

- Sleep Mode

This is the mode that the LSI performs sleep processing after measurement processing. (In the LSI which is sleeping, main clock and logic circuit are both suspended). And then, the LSI operation is resumed by wake-up from a microcontroller. This mode is chosen when the IntMode bit in the Control 1 Register (Address=0x2F) is set to "0".

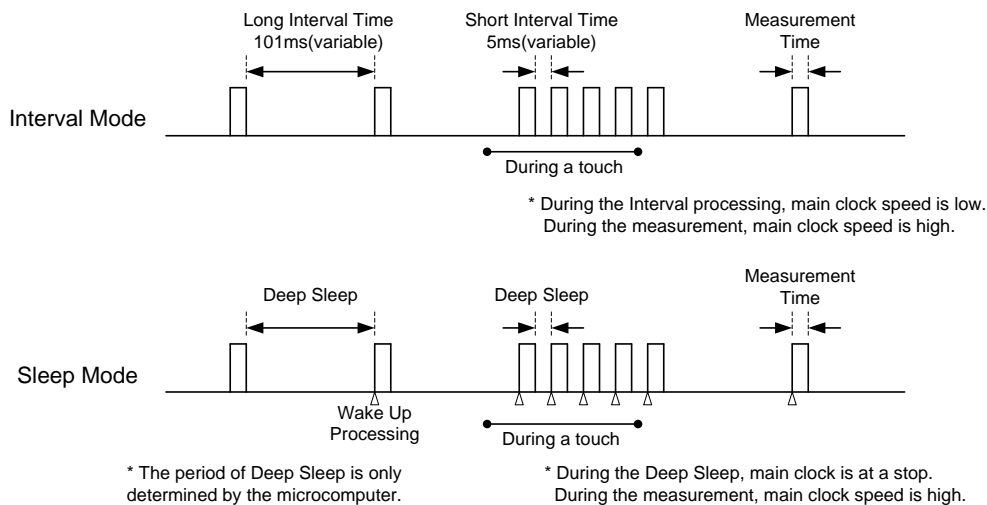


Fig.2-1 Interval Mode and Sleep Mode

2.1.2 Short Interval Mode and Long Interval Mode

- Short Interval Mode

This is the mode that the period of interval processing at Interval Mode is decided by the Short Interval Time Register (Address=0x34) (Short interval time). When the touch ON is detected, the mode changes into Short Interval Mode immediately. Furthermore, when this LSI operates with this mode and the touch ON has not been detected only for a little while, the mode doesn't change.

Condition for change to "Short Interval Mode"

Regardless of whether the previous state is Short Interval Mode or Long Interval Mode, if the measurement data of at least one channel is greater than or equal to the threshold, the mode changes into "Short Interval Mode" immediately.

- Long Interval Mode

This is the mode that the period of interval processing at Interval Mode is decided by the Long Interval Time Register (Address=0x35) and the LVIALB bit in the Measurement Mode 1 Register (Address=0x3A) (Long interval time). When the touch ON is not detected at all, the mode changes into Long Interval Mode.

Condition for change to "Long Interval Mode"

During the "Short Interval Mode" operation, if the number of consecutive measurement points that measurement data is less than the threshold for all the enabled channels becomes equal to the debounce counts that is specified by the Debounce Count 2 Register (Address=0x33), and subsequently the number of consecutive measurement points that data is less than the threshold for all the enabled channels becomes equal to the number of points which is specified by Long Interval Mode Start Count Register

(Address=0x3C), the mode changes into "Long Interval Mode". (In short, after "all the enabled channels have been in no touch state for a while, the mode changes into "Long Interval Mode".)

2.1.3 Calibration

- Static Offset Calibration

A measurement data of each enabled channel while not touching at all is used as a reference value. Static offset calibration is the processing to adjust this reference value to the median of measurement data range (-128 to 127).

When starting the system or when setting both the WriteReq bit and the StaCal bit in the Control 1 Register (Address=0x2F) to "1", static offset calibration is performed in the LC717A10.

If this is normal use environment, the static offset calibration is completed at once processing to adjust to a center level. However, recalibration is performed to up to three times consecutively with a worst case situation by inferior noise environment.

When a static offset calibration error occurs (In other words, when a reference value lies far from the median), the CALERR bit in the Error Status Register (Address=0x2C) is set to "1". And bits corresponding to channels where calibration error has occurred, which are in the Error Channel Status 1 Register (Address=0x2D) and the Error Channel Status 2 Register (Address=0x2E), are set to "1".

- Dynamic Offset Calibration

Dynamic offset calibration is the processing to correct continuously the difference between a reference value and the median of the entire measurement data range, which is caused by external factors such as temperature and humidity and so on.

When the long interval mode, the LSI judges the plus side's dynamic offset calibration execution at every measurement in the following cases and the minus side's dynamic offset calibration execution at every measurement regardless of a touch ON/OFF state.

When the short interval mode, the LSI judges the dynamic offset calibration execution at measurement every execution cycle of the number of times appointed by the Short Interval Dynamic OffCal Cycle Register (Address=0x36) in the following cases and the minus side's dynamic offset calibration execution at every measurement regardless of a touch ON/OFF state.

Case1: When the PDCLP bit is set to "0" and all channels are touch OFF states (OFF detection of all channels), this LSI carries out judgment processing.

Case2: When the PDCLP bit is set to "1", this LSI carries out judgment processing for a channel of touch OFF state because dynamic offset calibration is determined and performed at each individual enabled channel.

When the measurement data of the judgment processing (Judgment points) under the constant number of times specified by the Dynamic OffCal Count Plus Register (Address=0x37) continues in the plus side's dynamic offset calibration operating range (4 to CinX threshold), this LSI executes the dynamic offset calibration.

When the measurement data of the judgment processing (Judgment points) under the constant number of times specified by the Dynamic OffCal Count Minus Register (Address=0x38) continues in the minus side's dynamic offset calibration operating range (-128 to -4), this LSI executes the dynamic offset calibration.

When dynamic offset calibration error occurs (in other words, when a reference value lies far from the median), the CALERR bit in the Error Status Register (Address=0x2C) is set to "1", and the bit in the Error Channel Status 1 Register (Address=0x2D) and the Error Channel Status 2 Register (Address=0x2E) corresponding to the channels of the error occurrence are set to "1".

The LSI can be set without dynamic offset calibration execution by registers (Address=0x36 to 0x38) related to the dynamic offset calibration.

2.1.4 Noise Suppression

To increase the immunity against external noise(In other words, to suppress noise), some filters are used in this LSI. This LSI performs this function automatically.

2.2 Measurement Function

2.2.1 Relation Between Measurement Timing and Long Interval / Short Interval Setting

For example, the following schematic view shows the measurement timing when this LSI operates with the next settings.

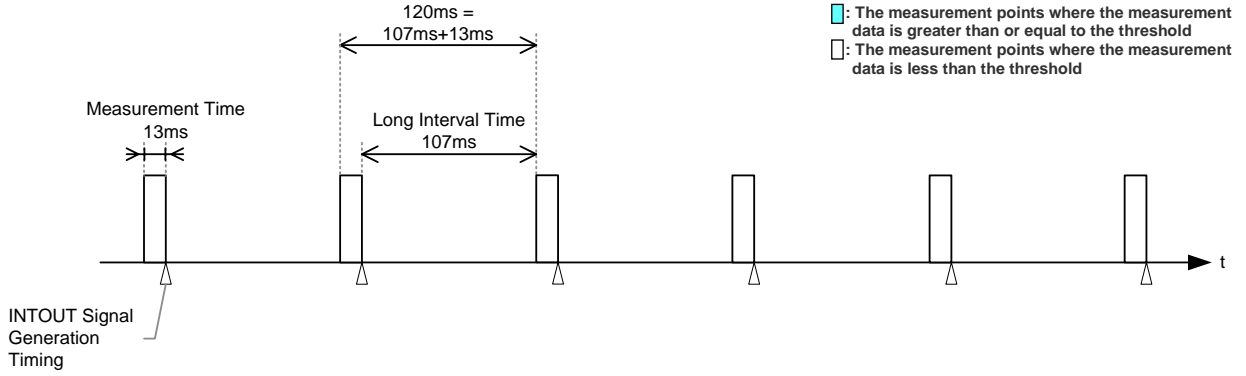
- Setting value of the Use Channel 1 Register (Address=0x00) is 0xFF
(Cin0 to Cin7 are enabled. This LSI performs the measurement of Cin0 to Cin7).
- Setting value of the Use Channel 2 Register (Address=0x01) is 0x00
(Cin8 to Cin15 are disabled. This LSI does not perform the measurement of Cin8 to Cin15).
- Setting value of the Average Count Register (Address=0x30) is 0x40
(One measurement time = 13 ms (Typ)).
- Setting value of the Filter Parameter Register (Address=0x31) is 0x04
(Initial value)
- Setting value of the Debounce Count 1 Register (Address=0x32) is 0x02
(Debounce count (OFF to ON) = 3 counts)
- Setting value of the Debounce Count 2 Register (Address=0x33) is 0x02
(Debounce count (ON to OFF) = 3 counts)
- Setting value of the Short Interval Time Register (Address=0x34) is 0x07
(Short interval time = 7ms (Typ)).
- Setting value of the Long Interval Time Register (Address=0x35) is 0x07
(Long interval time = 107 ms (Typ)).
- Setting value of the LIVALB bit in the Measurement Mode 1 Register (Address=0x3A) is "0"
(Long interval base time = 100 ms (Typ)).
- Setting value of the Long Interval Mode Start Count Register (Address=0x3C) is 0x01
(Cycle before start of long interval = 4 points)

In the following diagrams, the measurement points where the measurement data is greater than or equal to the threshold are shown as a light-blue box, measurement points where the measurement data is less than the threshold are shown as a white box.

The value of each "CinXACT" (Touch "ON"/"OFF" Status, X=0 to 15) in the diagrams is reflected to the corresponding bit in the Result Data 1 Register (Address=0x2A) and the Result Data 2 Register (Address=0x2B) respectively.

Both "AllChNoTch" (All Channels No Touch Status) and "LongIntvl" (Long Interval Mode Status) in the diagrams are the status this LSI has inside itself.

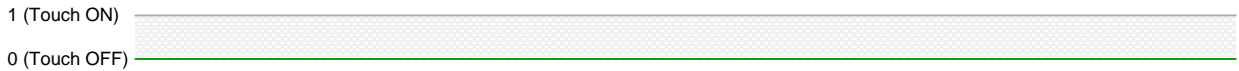
(1) Measurement timing when touch is not detected for all the enabled channels



Cin0 Touch ON/OFF Status : Cin0ACT (LSI Internal Signal)



Cin1 Touch ON/OFF Status : Cin1ACT (LSI Internal Signal)

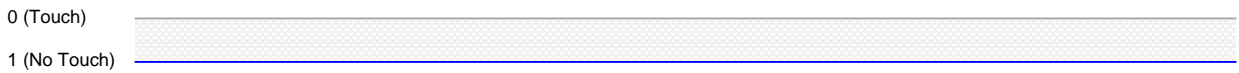


⋮

Cin15 Touch ON/OFF Status : Cin15ACT (LSI Internal Signal)



All Channels No Touch Status : AllChNoTch (LSI Internal Signal)



Long Interval Mode Status : LongIntvl (LSI Internal Signal)



Fig.2-2 Measurement timing when touch is not detected for all the enabled channels

(2) Measurement timing around the time when one or more enabled channels have been touched in the previous state of (1)

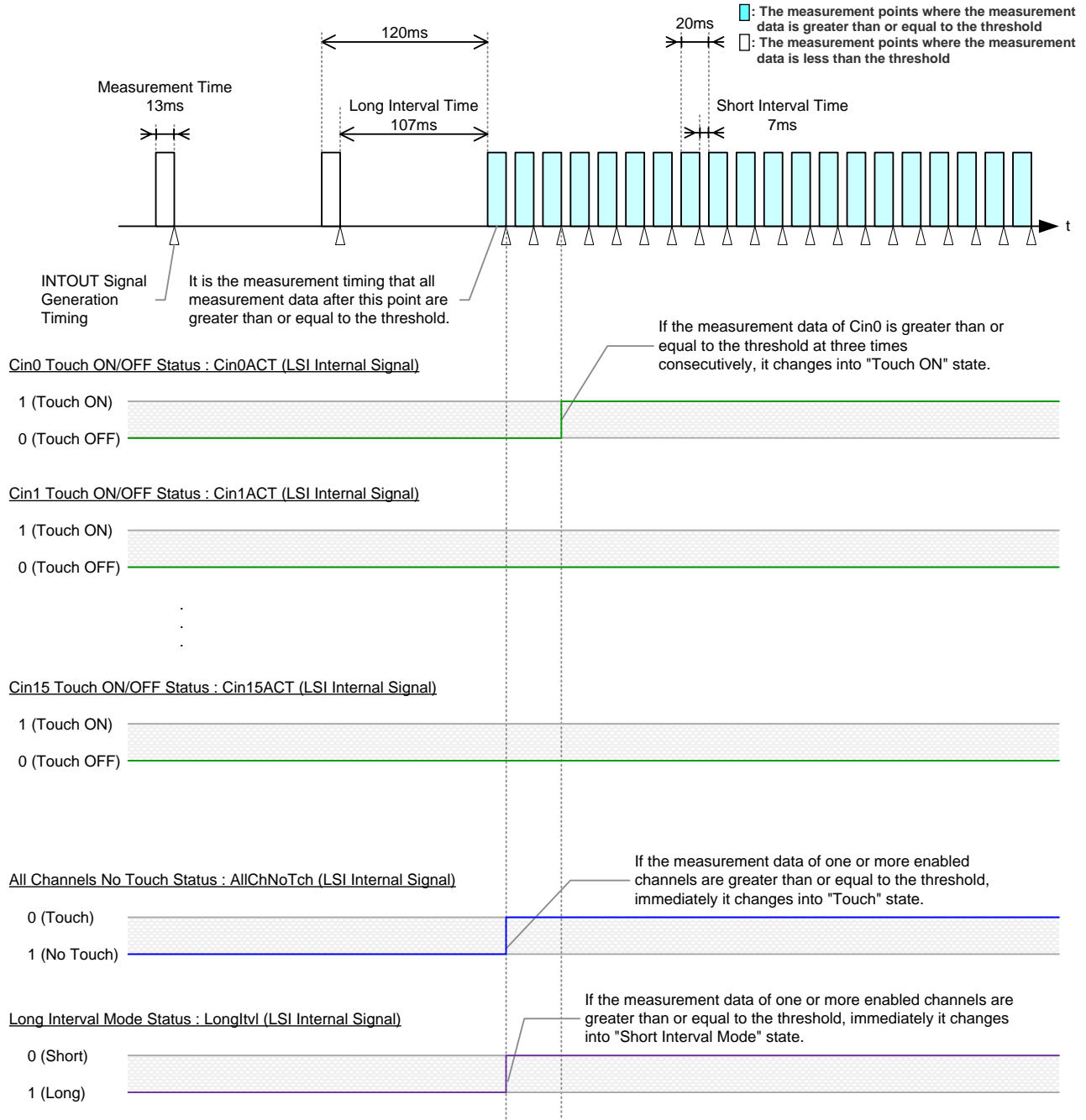
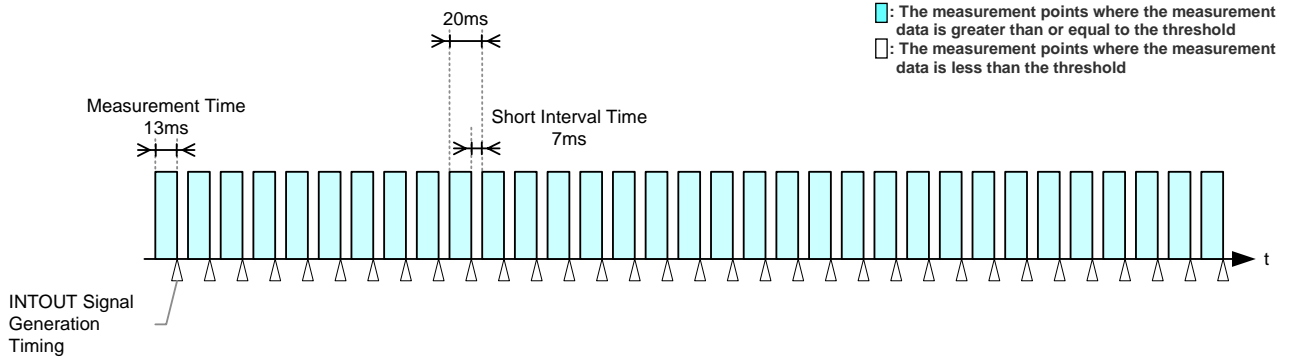
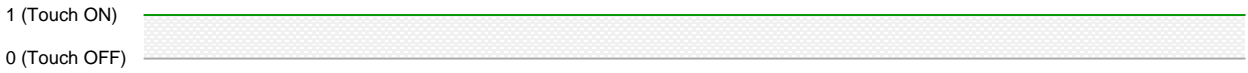


Fig.2-3 Measurement timing around the time when one or more enabled channels have been touched in the previous state of (1)

(3) Measurement timing when one or more enabled channels are being touched for a long time



Cin0 Touch ON/OFF Status : Cin0ACT (LSI Internal Signal)



Cin1 Touch ON/OFF Status : Cin1ACT (LSI Internal Signal)

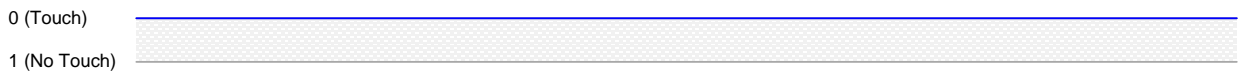


⋮

Cin15 Touch ON/OFF Status : Cin15ACT (LSI Internal Signal)



All Channels No Touch Status : AllChNoTch (LSI Internal Signal)



Long Interval Mode Status : LongIntvl (LSI Internal Signal)



Fig.2-4 Measurement timing when one or more enabled channels are being touched for a long time

(4) Measurement timing around the time when the touch has been released for all the enabled channels in the previous state of (3)

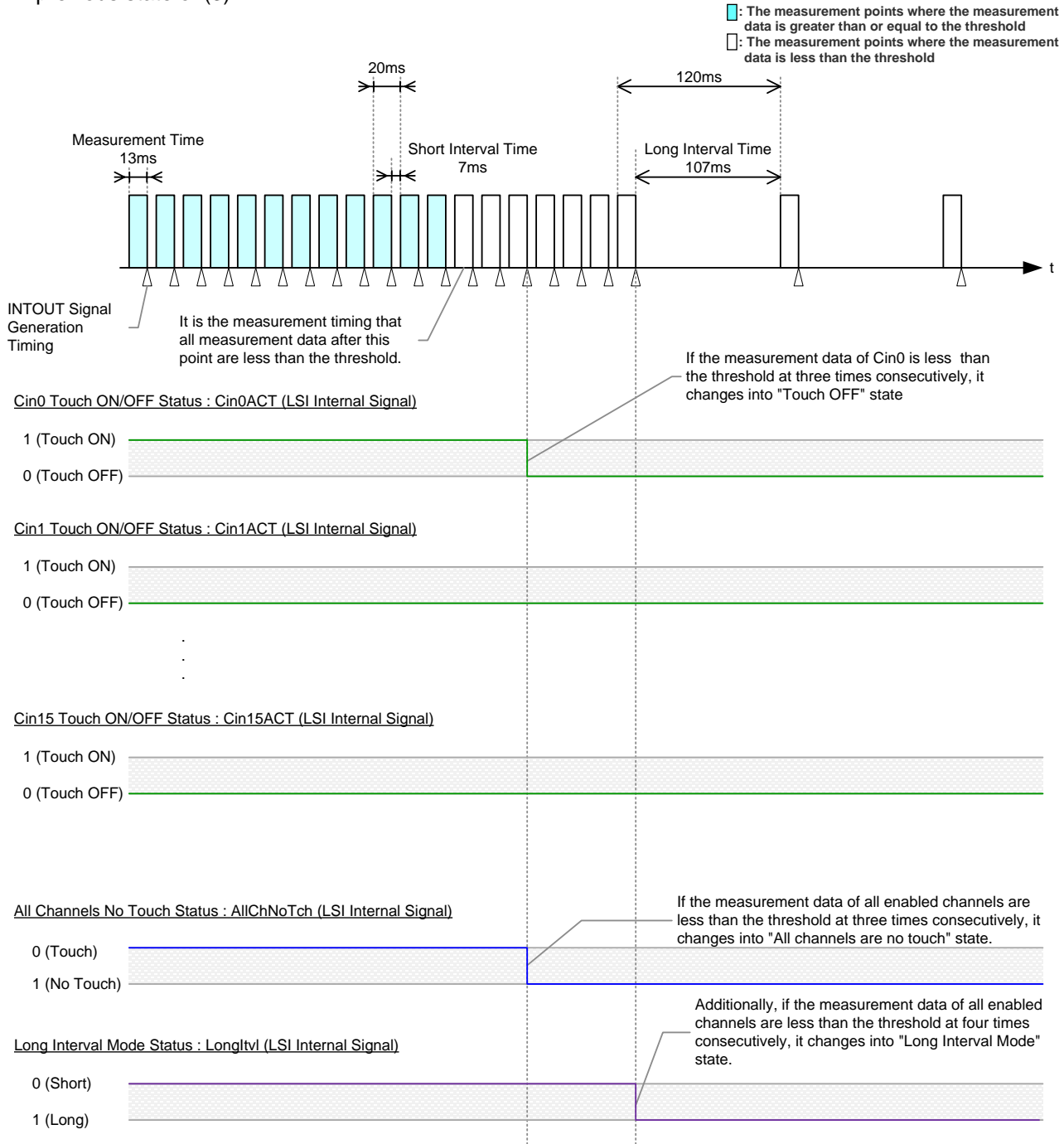


Fig.2-5 Measurement timing around the time when the touch has been released for all the enabled channels in the previous state of (3)

(5) In the previous state of (1), if measurement data is greater than or equal to the threshold only at a certain measurement point (Shown as a light-blue rectangle below), the measurement is performed in short interval mode for at least the time Interval determined depending on a debounce count which is specified by the Debounce Count 2 Register (Address=0x33), and a long interval start count which is specified by the Long Interval Mode Start Count Register (Address=0x3C). Thereafter, if the number of consecutive measurement points that measurement data is less than the threshold equals the number of the sum of the debounce count and the long interval start count, this LSI moves from short interval mode to long interval mode and starts to operate in long interval mode. In this case, the judgment result are all "OFF".

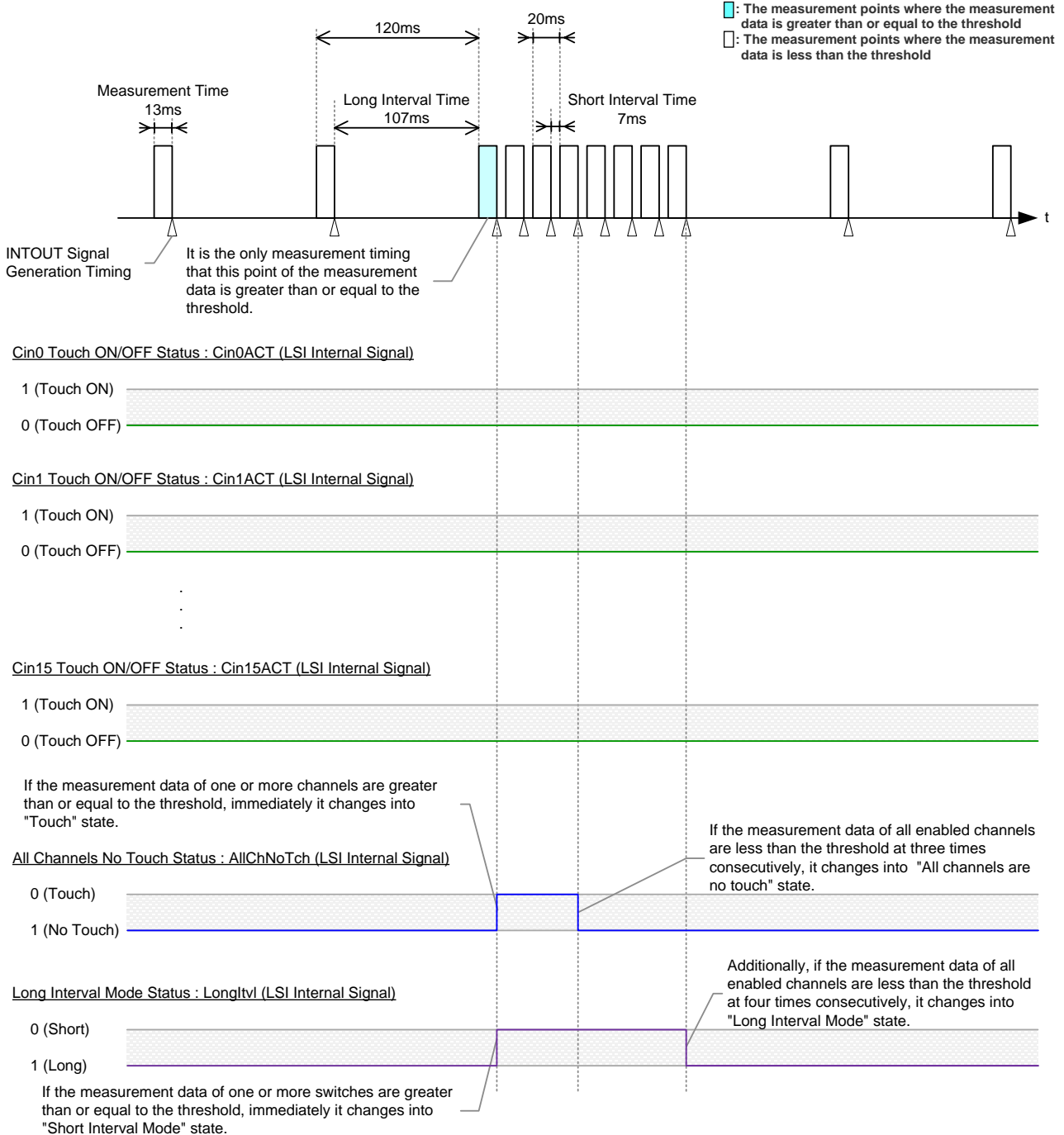


Fig.2-6 Measurement timing in the case that measurement data is greater than or equal to the threshold only at a certain measurement point In the previous state of (1)

(6) Measurement timing when the measurement data at a certain channel is affected by chattering etc.

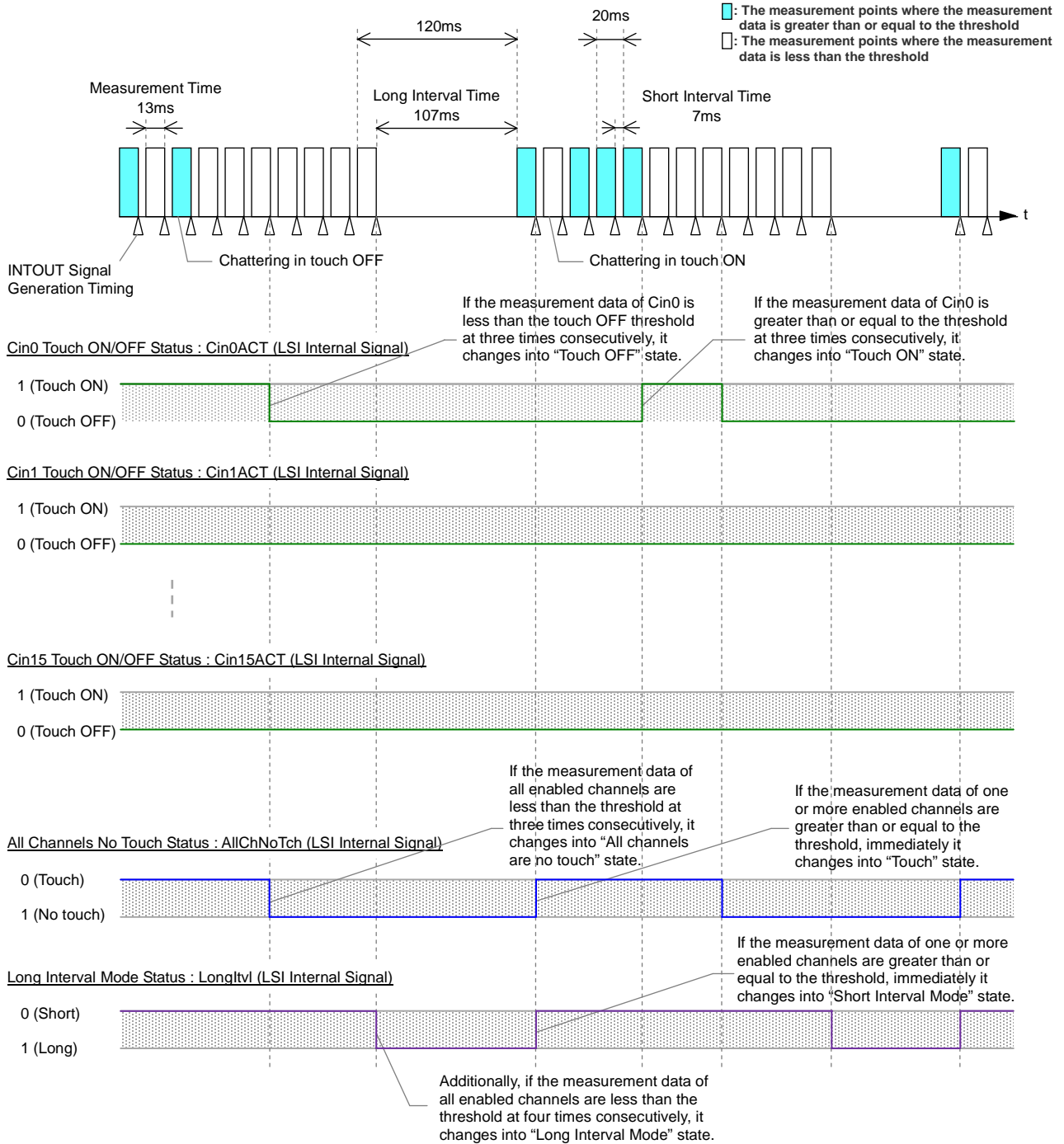


Fig.2-7 Measurement timing when the measurement data at a certain channel is affected by chattering etc.

(7) Measurement timing depending on the setting of the cycle before start of long interval

(a) Long Interval Mode Start Count Register (Address=0x3C) is 0x01(Cycle before start of long interval = 4 points)

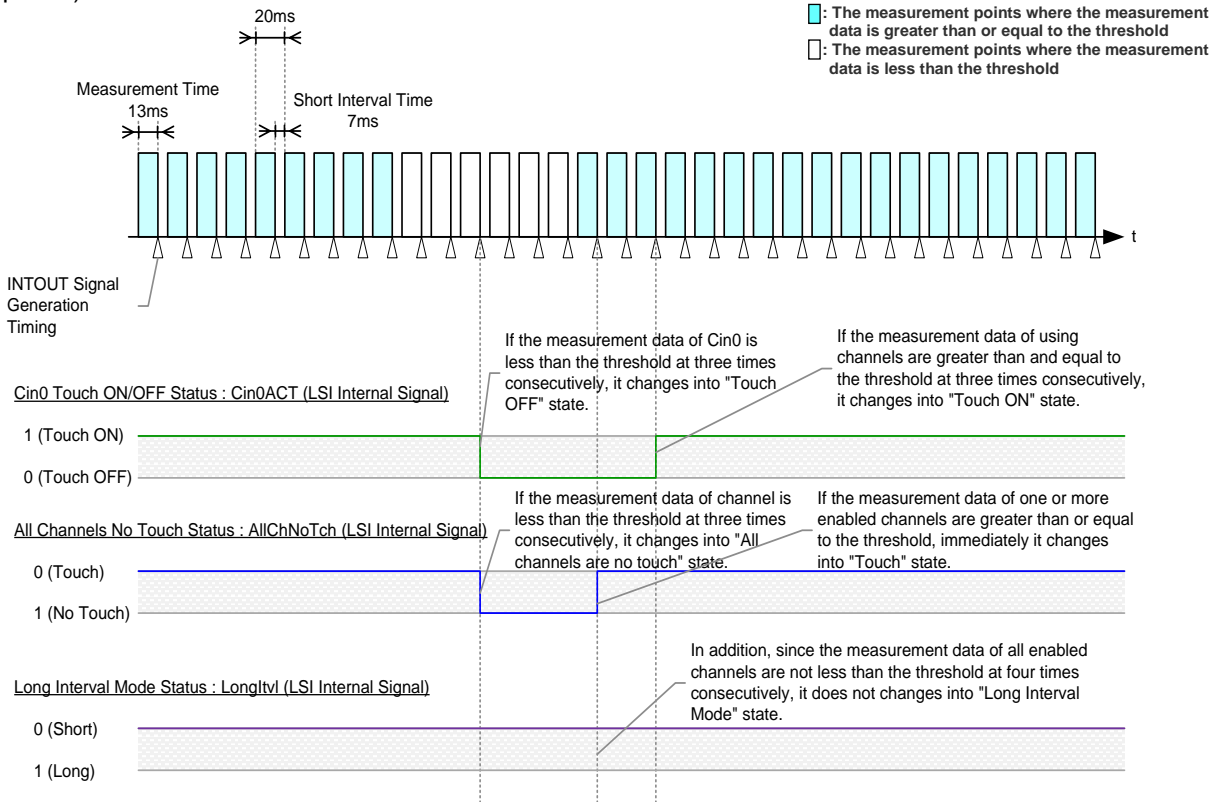


Fig.2-8-1 Measurement timing when the cycle before start of long interval is set to 4

(b) Long Interval Mode Start Count Register (Address=0x3C) is 0x00(Cycle before start of long interval = 0 point)

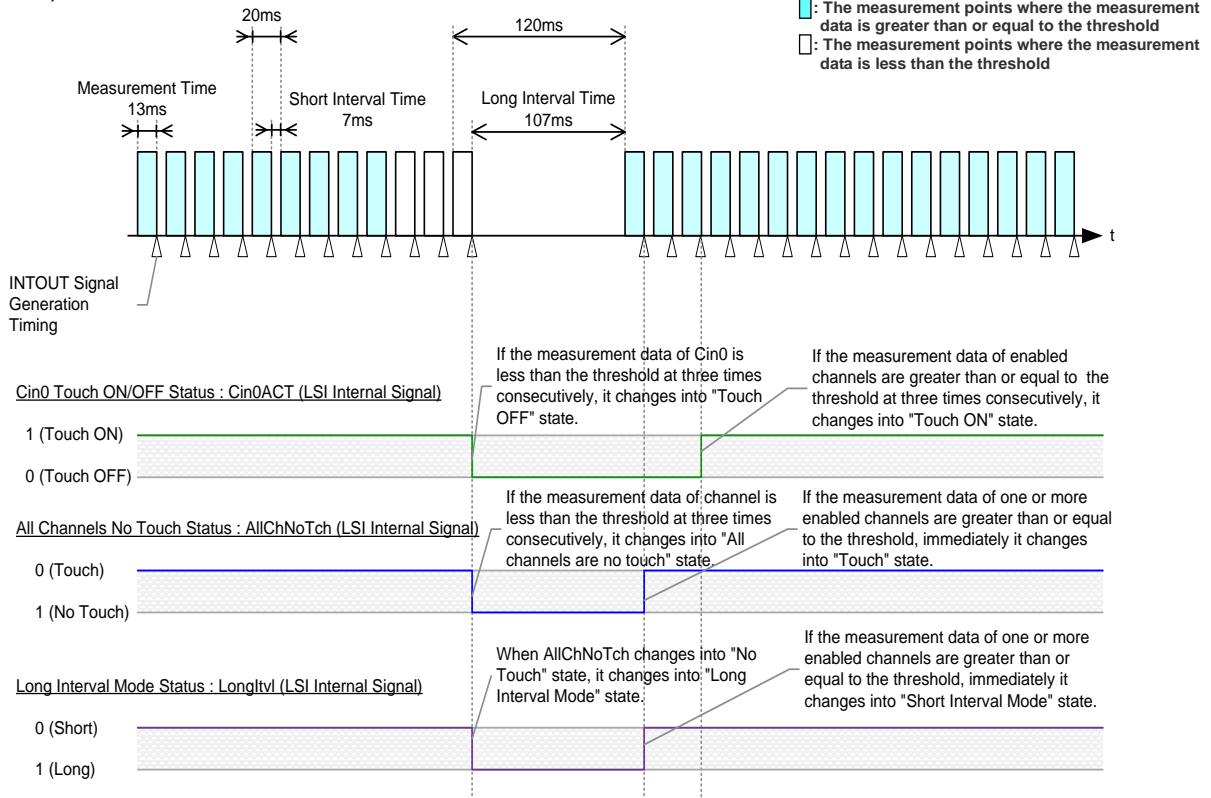


Fig.2-8-2 Measurement timing when the cycle before start of long interval is set to 0

2.2.2 Description of Dynamic Offset Calibration

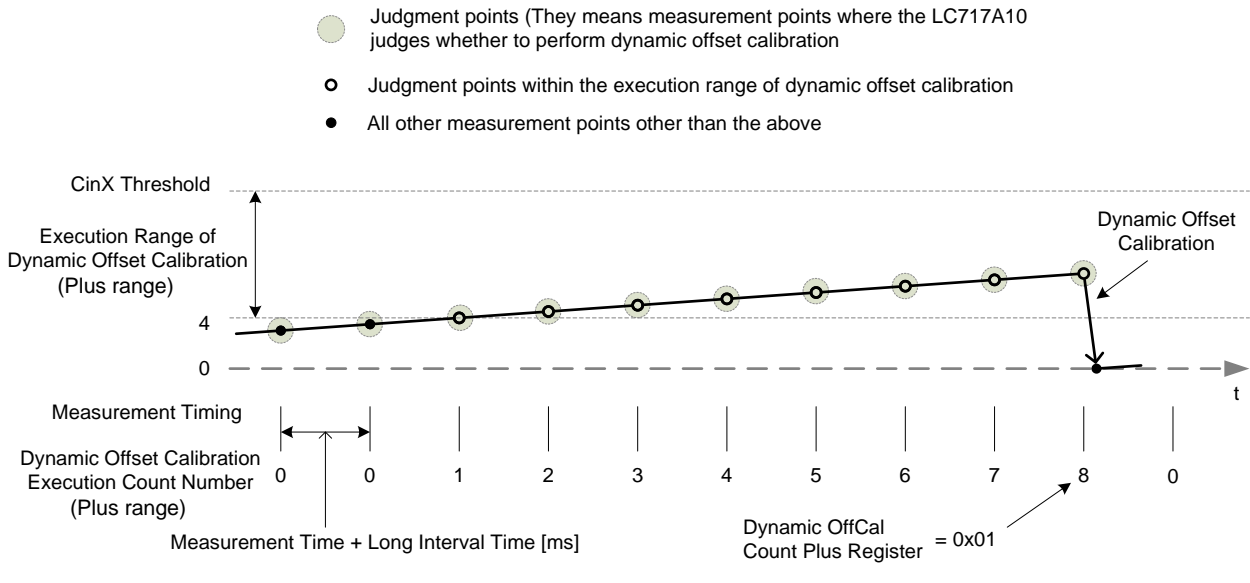
When measurement data at a certain channel are consecutively within the execution range of dynamic offset calibration (4 to CinX threshold, or -128 to -4) for the period of time corresponding to a value which is specified by the Short Interval Dynamic OffCal Cycle Register (Address=0x36), the Dynamic OffCal Count Plus Register (Address=0x37) and the Dynamic OffCal Count Minus Register (Address=0x38), dynamic offset calibration is performed and the reference value at the channel is gradually corrected to zero.

(1) During Long Interval Mode

Assume that touch is not detected for all enabled channels for a long time and this LSI operates in long interval mode. In such a case, the judgment of whether or not dynamic offset calibration is performed is made at each measurement. (In short, all measurement points are all judgment points)

If the number of consecutive measurement points that measurement data is within the plus range (4 to CinX threshold) equals a value which is specified by the Dynamic OffCal Count Plus Register (Address=0x37), dynamic offset calibration is automatically performed.

The following figure shows that measurement data increases gradually in the plus direction, where the Dynamic OffCal Count Plus Register (Address=0x37) = 0x01. (This means that dynamic offset calibration execution count is 8).



- * Either when measurement data is out of the execution range of dynamic calibration or when dynamic offset calibration has been performed, the plus side's dynamic offset calibration execution count number is immediately reset to "0".
- * In this case, all measurement points are judgment points.

Fig.2-9 Processing example of the plus side's dynamic offset calibration in long interval mode

Calculation formula of time until dynamic offset calibration execution in long interval mode is as follows:

$$(Measurement\ time + Long\ interval\ time) \times (Dynamic\ offset\ calibration\ execution\ counts - 1) [ms]$$

(2) During Short Interval Mode

Assume that touch is detected for at least one channel and this LSI operates in short interval mode.

In such a case, as measurement is performed with short interval during short interval mode, measurement interval during short interval mode is shorter than that during long interval mode. For this reason, the judgment of whether or not dynamic offset calibration is performed is made per the number of measurement points specified by the Short Interval Dynamic OffCal Cycle Register (Address=0x36). (In short, judgment points are some of measurement points)

If the number of consecutive judgment points that measurement data is within the minus range (-128 to -4) equals a value which is specified by the Dynamic OffCal Count Minus Register (Address=0x38), dynamic offset calibration is automatically performed.

The following figure shows that measurement data decreases gradually in the minus direction, where the Dynamic OffCal Count Minus Register (Address=0x38) = 0x08. (This means that dynamic offset calibration execution count is 8) and the Short Interval Dynamic OffCal Cycle Register (Address=0x36) = 0x02 (This setting means the judgment of whether or not dynamic offset calibration is performed is made per 2 measurement points).

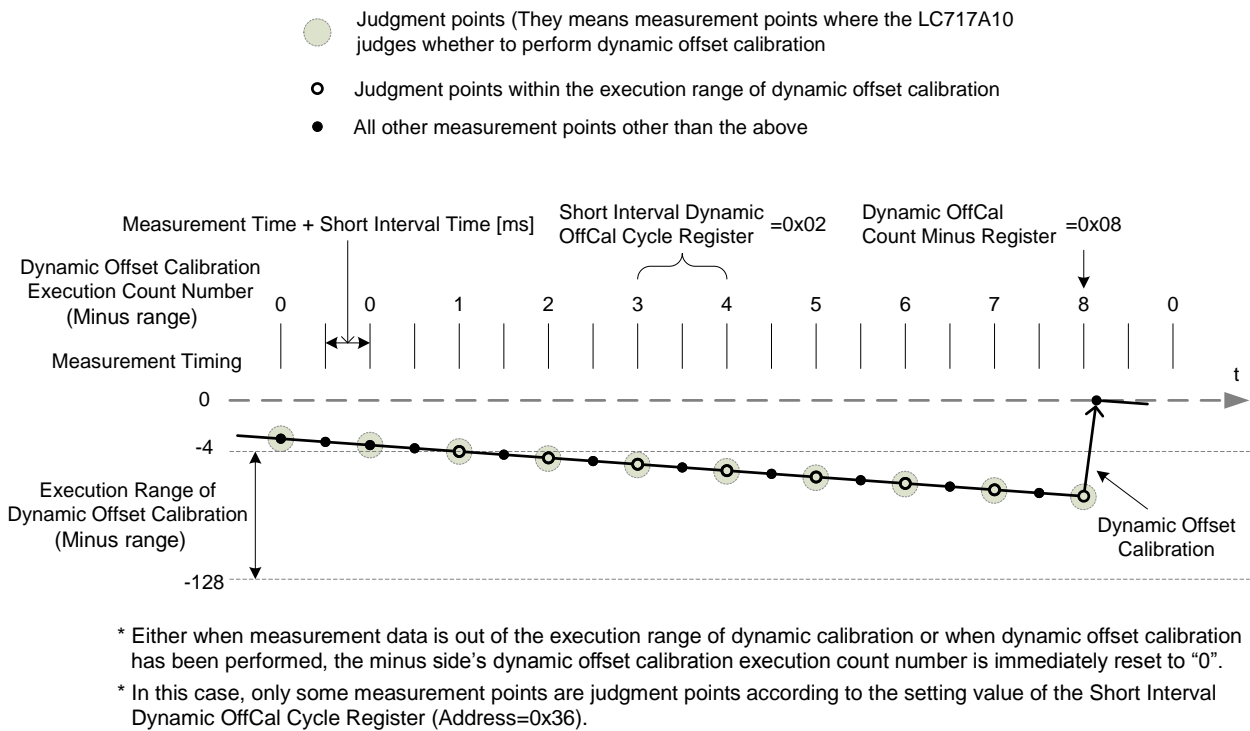


Fig.2-10 Processing example of the minus side's dynamic offset calibration in short interval mode

Calculation formula of time until dynamic offset calibration execution in short interval mode is as follows:

$$(Measurement\ time + Short\ interval\ time) \times (Dynamic\ offset\ calibration\ execution\ counts - 1) \times (The\ number\ of\ measurement\ points\ specified\ by\ Short\ Interval\ Dynamic\ OffCal\ Cycle\ Register) [ms]$$

2.2.3 Approximate Calculation Formula for Measurement Interval

The approximate calculation formulas for measurement interval (The interval from measurement start to next measurement start) for 16 channels are as follows.

Measurement interval of the Short Interval Mode [ms] (Typ)

$$= \frac{\{FP1k \times (\text{Setting value of the Average Count Register} + 4) + 0.1\}}{\text{Measurement time for one channel}} \times \frac{16}{\text{Number of channels measured}} + (\text{Short interval time})$$

Processing time in this LSI (Except measurement time)

Measurement interval of the Long Interval Mode [ms] (Typ)

$$= \{FP1k \times (\text{Setting value of the Average Count Register} + 4) + 0.1\} \times 16 + (\text{Long interval time})$$

In the case of FP2_[3:0] = 0, FP1k can be calculated by the following formula from FP1_[3:0] in the Filter Parameter Register (Address=0x31).

$$FP1k = 0.021 + (FP1_{[3:0]} \times 0.000375)$$

The measurement interval in the case of initial setting (Specifically, FP1_[3:0] = 4, the value of average count = 64 times, the value of the short interval time = 5 ms, the value of the long interval time = 101 ms and the number of enabled channels are 16) is as follows.

Measurement interval of the short interval mode = 31.08 ms (Typ)

Measurement interval of the long interval mode = 127.08 ms (Typ)

2.2.4 Description of Dynamic Threshold Control When Moving from ON State to OFF State

The LC717A10 does the dynamic threshold control processing only when the ON/OFF judgment result moves from ON state to OFF state.

In the case that the TOFFTH bit in the Measurement Mode 1 Register (Address=0x3A) is set to “0”, the threshold value for judging “Touch OFF” is the larger of the value of DYNTHR1 and the threshold value specified by CinX Threshold Register (Address=0x0A to 0x19), where $DYNTHR1 = MAXDATA - (MAXDATA / 4) = (3/4)MAXDATA$ and MAXDATA is the maximum value of all the measurement data after the ON/OFF judgment result has changed into ON state.

On the other hand, in the case that the TOFFTH bit is set to “1”, the threshold value for judging “Touch OFF” is the larger of the value of DYNTHR2 and the threshold value specified by CinX Threshold Register (Address=0x0A to 0x19), where $DYNTHR2 = MAXDATA - (MAXDATA / 2) = (1/2)MAXDATA$ and MAXDATA is the maximum value of all the measurement data after the ON/OFF judgment result has changed into ON state.

Note: Normally, set the TOFFTH bit to “1”.

Attention:

The LC717A10 does the dynamic threshold control processing when the ON/OFF judgment result moves from ON state to OFF state. For this reason, the phenomenon where the ON/OFF judgment result changes into unexpected OFF state and subsequently returns to ON state may occur as shown in Fig.2-12. When the TOFFTH bit is set to “0”, this phenomenon may not occur by setting the TOFFTH bit to “1”.

In the ON/OFF judgment processing in the LC717A10, it is not assumed that the measurement data that measurement data value is less than the threshold for judging “Touch ON” and is greater than or equal to the threshold specified by the CinX Threshold Register (Address=0x0A to 0x19) continues for a while after the ON/OFF judgment result has changed into ON state.

When the above-described phenomenon may occur, the way of solving a problem is as follows: a microcontroller reads out the measurement value and then it carries out ON/OFF judgment without using the result data read from the LC717A10.

(1) In the case that the TOFFTH bit is set to "1"

For example, the following schematic view (Fig.2-11) describes dynamic threshold control when the LC717A10 operates with the next settings. However, LC717A10's all channels assume that touch ON has not been detected for a long time at first.

- Interval mode
- The TOFFTH bit in the Measurement Mode 1 Register (Address=0x3A) = "1"
- The sum of the measurement time and the short interval time is 20 ms
- The sum of the measurement time and the long interval time is 120 ms
- Debounce count 1 = 3 counts
- Debounce count 2 = 3 counts
- CinX Threshold value = 15
- Cycle before start of long interval = 40 points

In Fig.2-11, the threshold value for judging "Touch ON" is the one specified by CinX Threshold Register (Address=0x0A to 0x19). On the other hand, the threshold value for judging "Touch OFF" is the larger of the threshold value specified by CinX Threshold Register (Address=0x0A to 0x19) and the half of the maximum value of all the measurement data after the ON/OFF judgment result has changed into ON state.

In the case that the TOFFTH bit in the Measurement Mode 1 Register (Address=0x3A) is set to "1", in order that the ON/OFF judgment result at a certain channel which is in ON state moves from ON state to OFF state, it is necessary that the number of consecutive measurement points that measurement data is less than the threshold value for judging "Touch OFF" gets to the value of the debounce count specified by the Debounce Count 2 Register (Address=0x33).

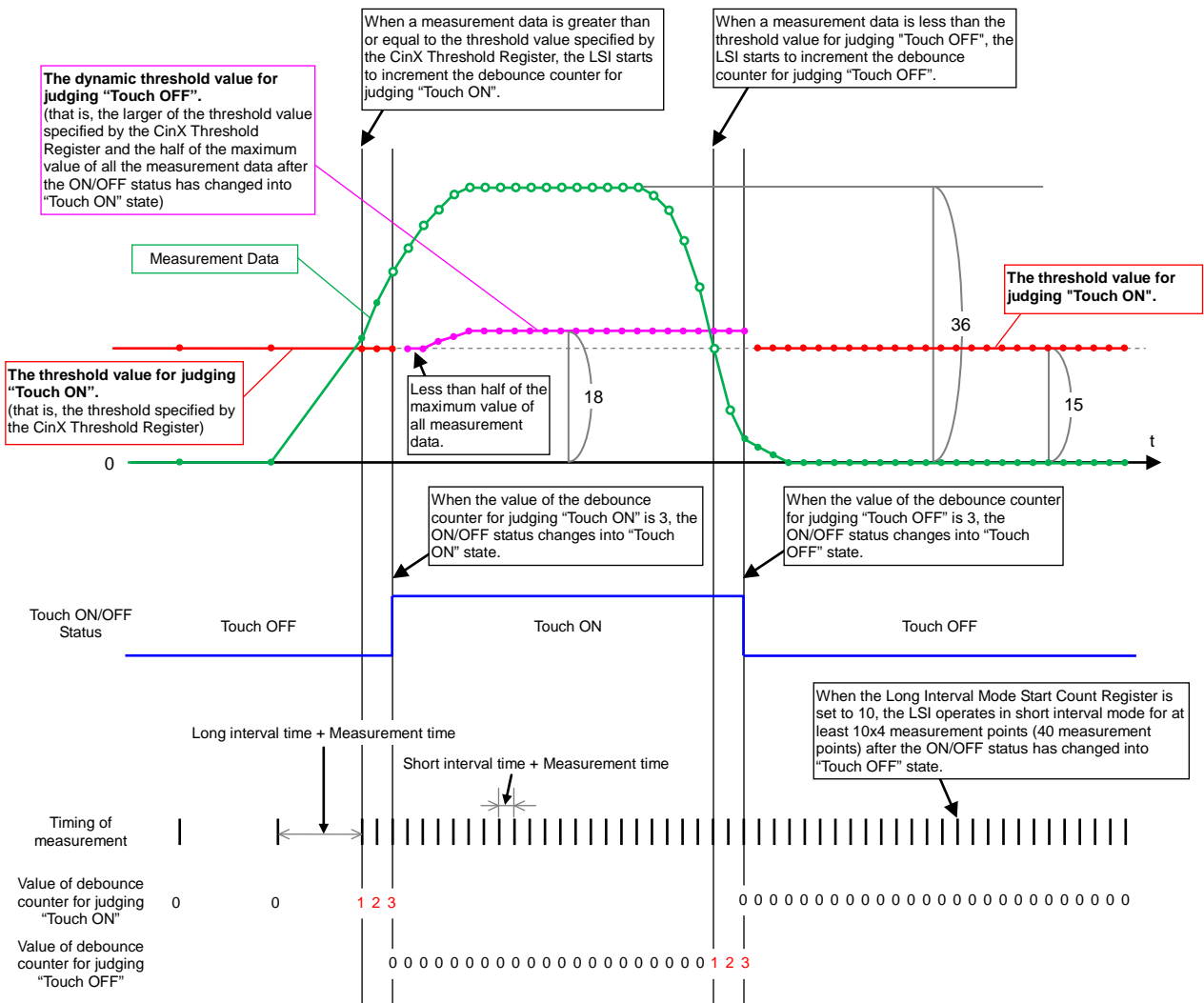


Fig.2-11 In the case that the TOFFTH bit is set to "1"

(2) In the case that the TOFFTH bit is set to "0"

For example, the following schematic view (Fig.2-12) describes dynamic threshold control when the LC717A10 operates with the next settings. However, LC717A10's all channels assume that touch ON has not been detected for a long time at first.

- Interval mode
- The TOFFTH bit in the Measurement Mode 1 Register (Address=0x3A) = "0"
- The sum of the measurement time and the short interval time is 20 ms
- The sum of the measurement time and the long interval time is 120 ms
- Debounce count 1 = 3 counts
- Debounce count 2 = 3 counts
- CinX Threshold value = 15
- Cycle before start of long interval = 40 points

In Fig.2-12, the threshold value for judging "Touch ON" is the one specified by CinX Threshold Register (Address=0x0A to 0x19). On the other hand, the threshold value for judging "Touch OFF" is the larger of the threshold value specified by the CinX Threshold Register (Address=0x0A to 0x19) and the three-fourths of the maximum value of all the measurement data after the ON/OFF judgment result has changed into ON state.

In the case that the TOFFTH bit in the Measurement Mode 1 Register (Address=0x3A) is set to "0", in order that the ON/OFF judgment result at a certain channel which is in ON state moves from ON state to OFF state, it is necessary that the number of consecutive measurement points that measurement data is less than the threshold value for judging "Touch OFF" gets to the value of the debounce count specified by the Debounce Count 2 Register (Address=0x33).

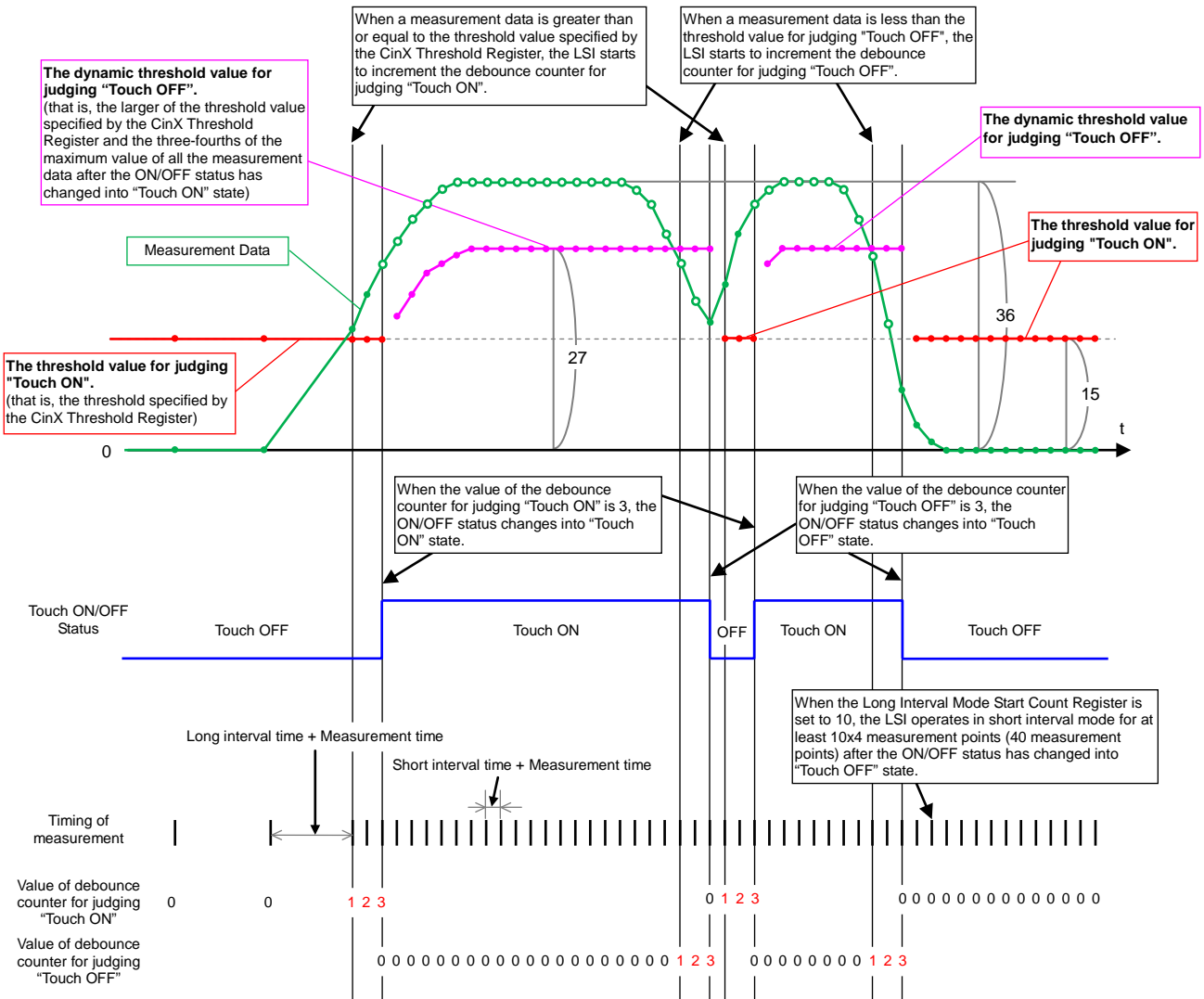


Fig.2-12 In the case that the TOFFTH bit is set to "0"

Chapter 3 OPERATION SEQUENCE

In chapter 3, examples of operation sequence of the LC717A10 are described.

Note: *In this chapter, the term “Measurement” means that an LC717A10 measures the amount of capacitance variation for each channel which is specified by the Use Channel 1 Register (Address=0x00) and the Use Channel 2 Register (Address=0x01), and performs ON/OFF judgment for each channel, and reflects measurement results to registers such as the Result Data 1 Register (Address=0x2A) and the Result Data 2 Register (Address=0x2B).*

Note: *When at least one bit of the WriteReq bit, the ParaCh bit and the StaCal bit is not “0”, the microcontroller must not write to the Control 1 Register. (In other word, only when all of the WriteReq bit, the ParaCh bit and the StaCal bit are “0”, the microcontroller is able to write to the Control 1 Register.)
For more information, refer to explanation of the Control 1 Register (Address=0x2F) in this document.*

3.1 An Example of Operation Sequence After “External Reset”

Fig.3.1 Shows the sequence of operations in the case that a microcontroller does not update configuration of the LC717A10 after the external reset (nRST) of the LC717A10.

The details are described below.

- (1) The microcontroller performs the external reset of the LC717A10.
The reset period: *1 μs or more*
- (2) After a definite period of time (*Maximum 20 ms*) since the external reset has been released, the internal reset of the LC717A10 is released automatically.
When the internal reset has been released, the LC717A10 negates the INTOOUT signal.
- (3) The LC717A10 performs initialization processing.
In the initialization processing, the LC717A10 sets each register to predetermined default value.
The time required for internal initialization: *about 1 ms (Typ)*
- (4) After the initialization processing has completed, the LC717A10 asserts the INTOOUT signal in order to notify the completion of internal initialization to the microcontroller.
- (5) After (4), Either I²C interface or SPI interface is available in the LC717A10. The microcontroller is able to read / write to the registers of the LC717A10.
- (6) The LC717A10 enters “Waiting-for-host-access state”. (In other words, the LC717A10 starts waiting for the access from the host.) As long as the microcontroller doesn't access to the registers of the LC717A10 at all during the “Waiting-for-host-access state”, the LC717A10 automatically exits the state *10 ms (Typ)* later. (In other words, the LC717A10 automatically stops waiting for the access from the host)
Note: *The moment the microcontroller sets at least the WriteReq bit in the Control 1 Register (Address=0x2F) to “1” during the “host-connection-waiting state”, the LC717A10 exits the “Host-connection-waiting state”. In such a case as this, the contents of the following (7) and (8) need to be changed.*
- (7) The LC717A10 performs static offset calibration based on its own initial settings.
When the static offset calibration has completed, the LC717A10 automatically sets the StaCal bit in the Control 1 Register (Address=0x2F) to “0”.
The time required for static offset calibration (Normal): *about 206 ms (Typ)*
The time required for static offset calibration (Maximum recalibration): *about 618 ms (Typ)*
- (8) The LC717A10 repeats processing of measurement of Cin0 to Cin15 and subsequent interval processing.

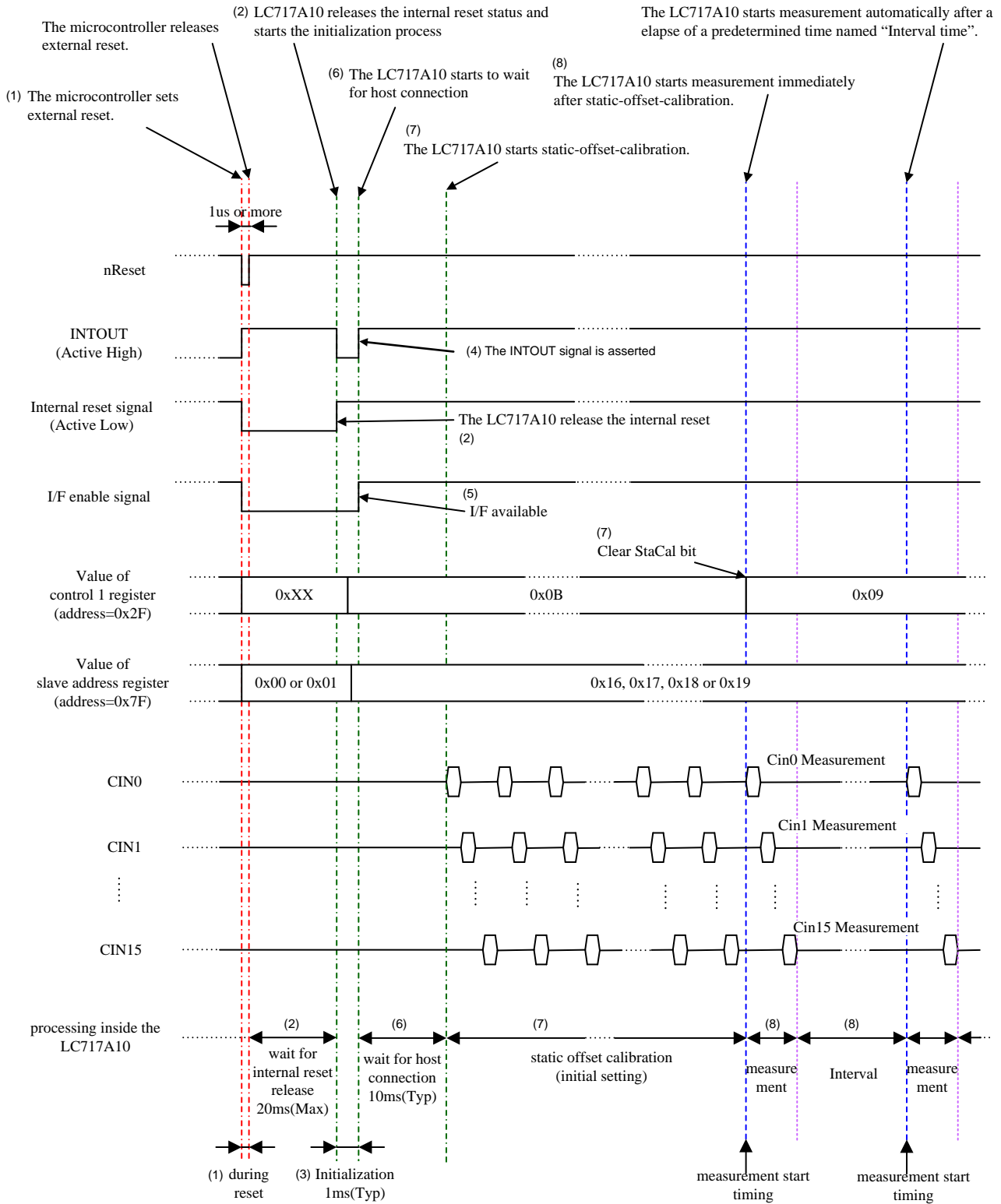


Fig.3-1 An example of operation sequence after "External Reset"

3.2 An Example of Operation Sequence in the “Interval Mode”

Note: Generally, it is recommended for the LC717A10 to operate in “Interval” mode rather than in “Sleep” mode.

An example of operation sequence in the “Interval mode” is shown in Fig.3-2 and Fig.3-3. In this example, both the INTMD1 bit and the INTMD2 bit in the Measurement 1 Register (Address=0x3A) are always set to “0”, so an LC717A10 always asserts the INTOUT immediately after the measurement has completed, and it doesn't negate the INTOUT signal automatically.

The details about Fig.3-2 and Fig.3-3 are described below.

- (1) Assume that an LC717A10 operates in the “Interval mode” with all 16 channels enabled.
(In concrete terms, the LC717A10 repeats measurement, and the microcontroller negates the INTOUT signal and reads measurement results over the I²C (or SPI) bus every time it detects the assertion of the INTOUT signal.)
- (2) In order to change the settings of the LC717A10, the microcontroller writes an appropriate parameter value to each register of the LC717A10, the address range of which is 0x00 to 0x19 or 0x30 to 0x3D, over the I²C (or SPI) bus. (*Note: At this time, the LC717A10 doesn't still operate according to the new settings.*)
- (3) The microcontroller writes 0x8F to the Control 1 Register (Address=0x2F) in order to issue a request to the LC717A10 for setting parameters and performing static offset calibration.
- (4) LC717A10 pends this request until the end of the period of “Interval”.
- (5) As soon as the period of “Interval” ends, the LC717A10 accepts the request described in the above (3). As a result, the WriteReq bit in the Control 1 Register (Address=0x2F) changes from “1” to “0”. Subsequently, the new settings are reflected to the LC717A10's operation. (That is to say, the LC717A10 operates according to the new settings.)
- (6) After (5), the LC717A10 performs the static offset calibration based on the new settings.
- (7) The LC717A10 starts measurement. Every time the measurement of all the enabled CinXs has completed. The LC717A10 asserts the INTOUT signal.
- (8) Every time the microcontroller detects the assertion of the INTOUT signal, the microcontroller negates the INTOUT signal and reads measurement results over the I²C (or SPI) bus.

(1) The LC717A10 starts measurement automatically after an elapse of a predetermined time named "Interval time".

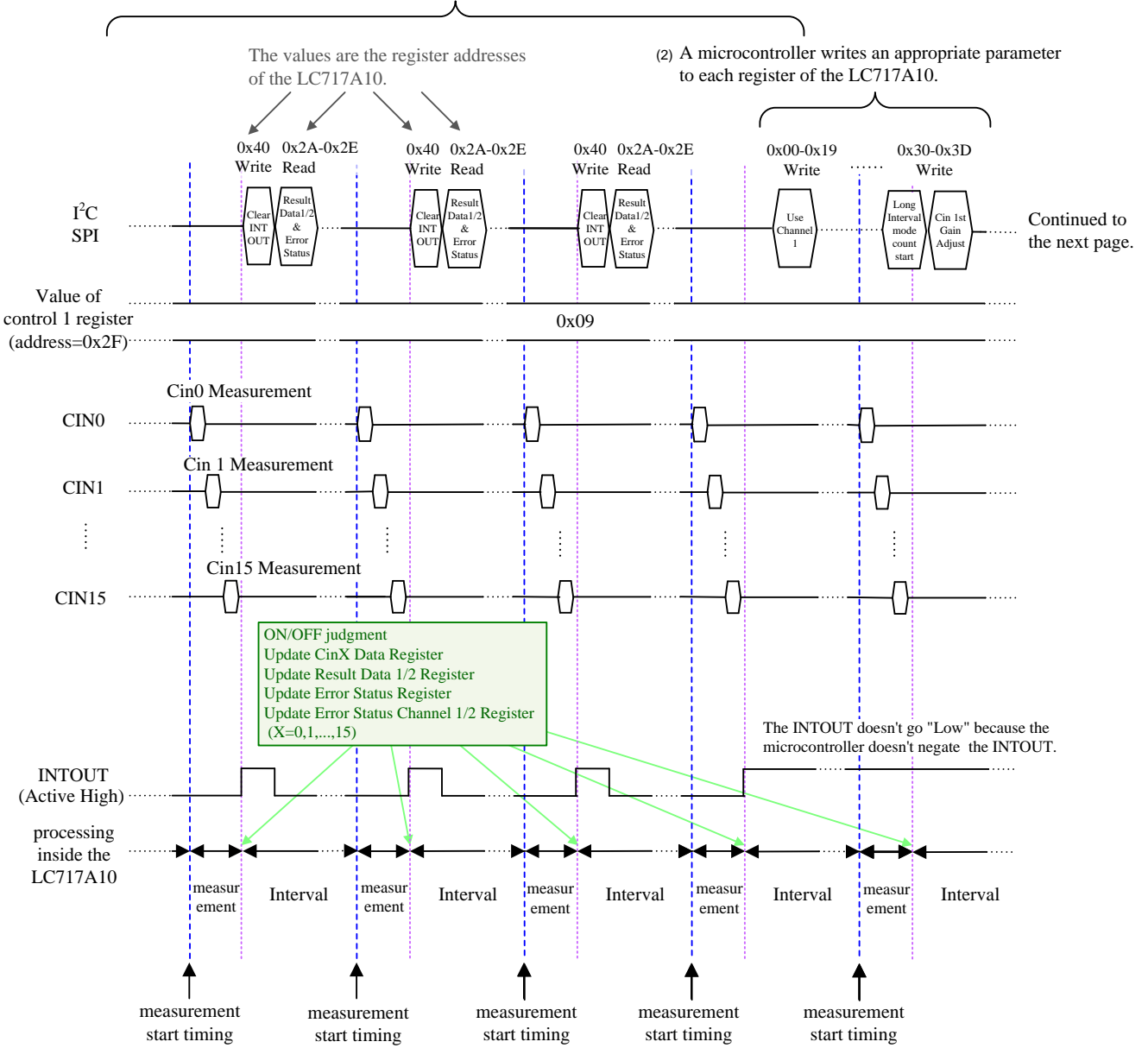


Fig.3-2 An example of operation sequence in the "Interval Mode" (1/2)

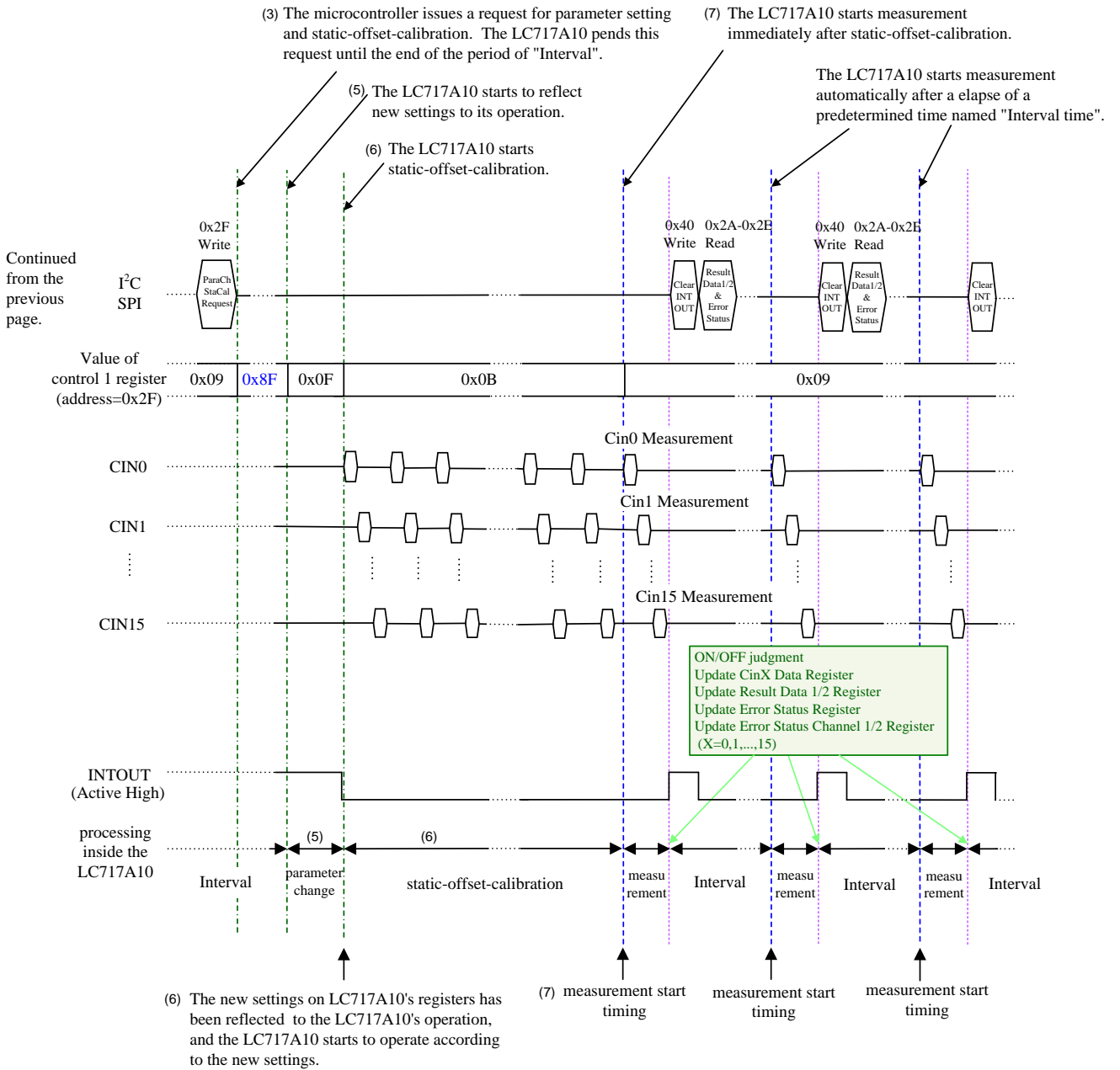


Fig.3-3 An example of operation sequence in the "Interval Mode" (2/2)

3.3 An Example of Operation Sequence in the “Sleep Mode”

An example of operation sequence in the “Sleep” mode is shown in Fig.3-4 and Fig.3-5. In this example, both the INTMD1 bit and the INTMD2 bit in the Measurement 1 Register (Address=0x3A) are always set to “0”, so an LC717A10 always asserts the INTOUT immediately after the measurement has completed.

The details about Fig.3-4 and Fig.3-5 are described below.

- (1) Assume that an LC717A10 operates in the “Sleep mode” with all 16 channels enabled.
(In concrete terms, the LC717A10 and the microcontroller operate as follows: Every time the microcontroller detects the assertion of the INTOUT signal (That is, every time the LC717A10 has gone to sleep), the microcontroller reads measurement results over the I²C (or SPI) bus and it wakes up the LC717A10 at a predetermined time.
- (2) In order to change the settings of the LC717A10, the microcontroller writes an appropriate parameter value to each register of the LC717A10, the address range of which is 0x00 to 0x19 or 0x30 to 0x3D, over the I²C (or SPI) bus. (*Note: At this time, the LC717A10 doesn't still operate according to the new settings.*)
- (3) The microcontroller writes 0x87 to the Control 1 Register (Address=0x2F) in order to issue a request to the LC717A10 for setting parameters and performing static offset calibration.
- (4) After (3), the microcontroller wakes up the LC717A10.
- (5) After the LC717A10 has been woken up, the LC717A10 accepts the request described in the above (3). As a result, the WriteReq bit in the Control 1 Register (Address=0x2F) changes from “1” to “0”. Subsequently, the new settings are reflected to the LC717A10's operation. (That is to say, the LC717A10 operates according to the new settings.)
- (6) After the reflection of the new settings, the LC717A10 performs the static offset calibration based on the new settings.
- (7) The LC717A10 starts measurement. Every time the measurement of all the enabled CinXs has completed, the LC717A10 asserts the INTOUT signal and goes to sleep.
- (8) Every time the microcontroller detects the assertion of the INTOUT signal(that is, the LC717A10 has gone to sleep), the microcontroller reads measurement results over the I²C (or SPI) bus and it wakes up the LC717A10 at a predetermined time.
- (9) The microcontroller repeats the above (7) to (8).

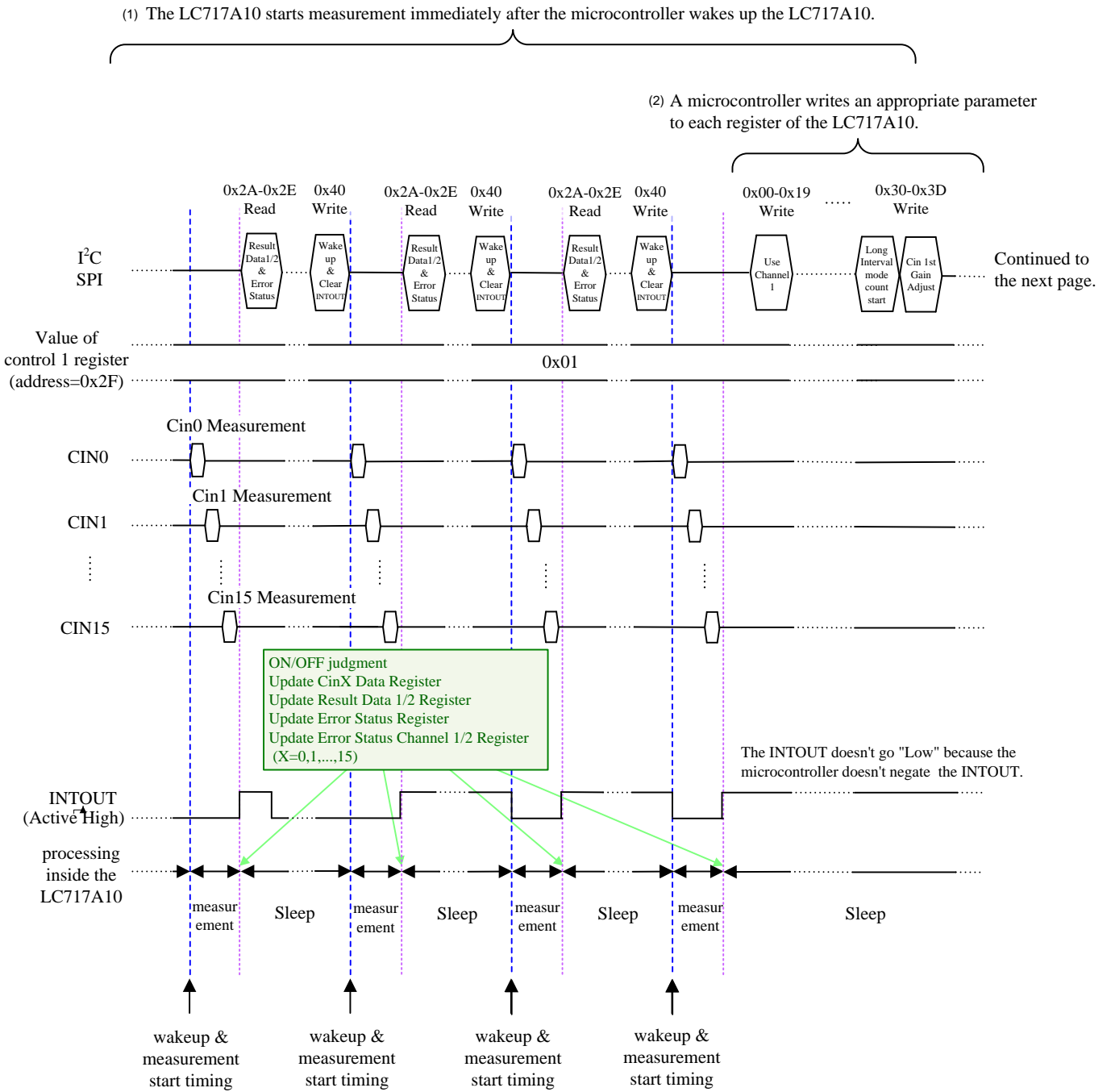


Fig.3-4 An example of operation sequence in the "Sleep Mode" (1/2)

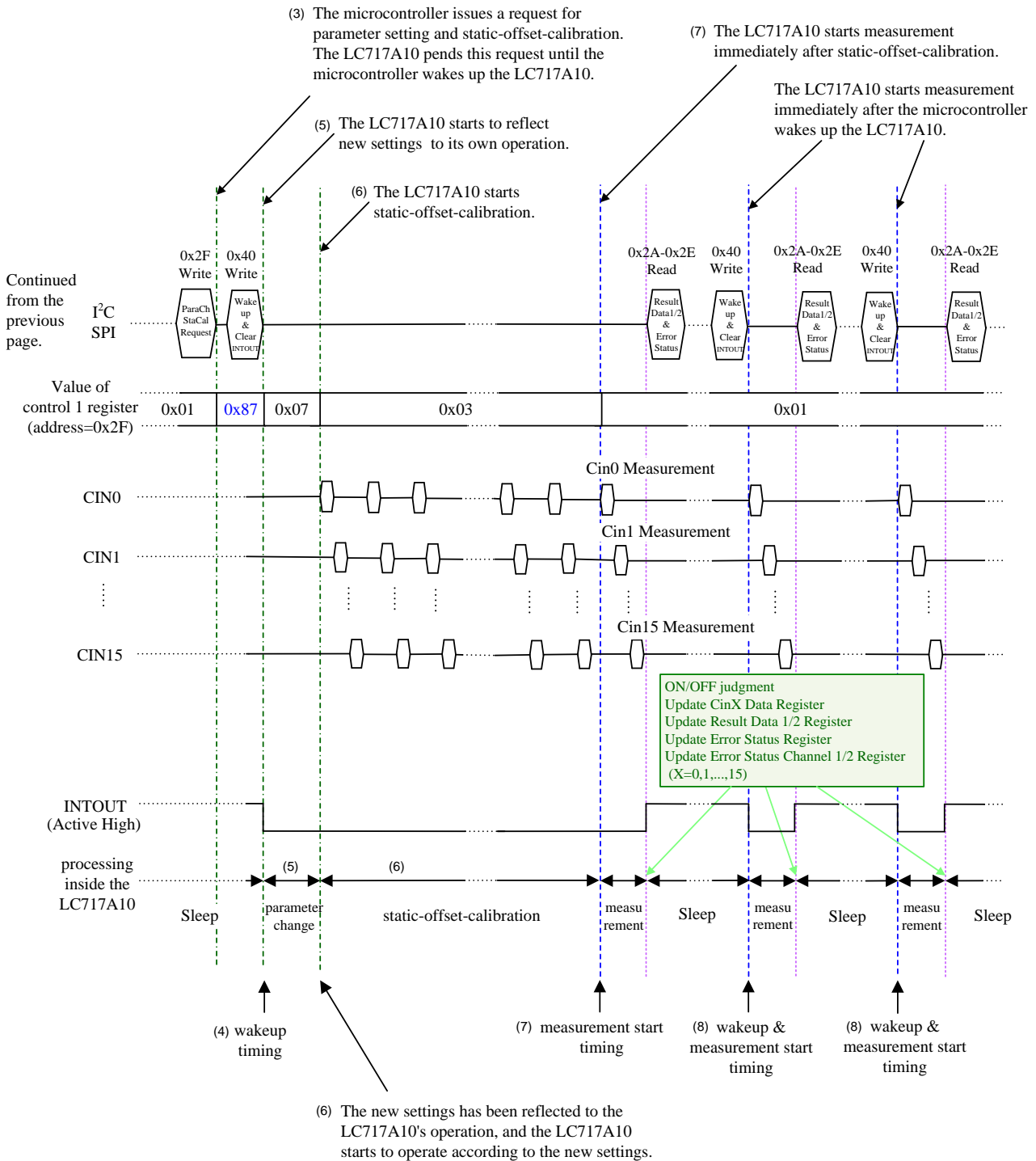


Fig.3-5 An example of operation sequence in the "Sleep Mode" (2/2)

Chapter 4 CONTROL EXAMPLES FOR A MICROCONTROLLER

In chapter 4, three control flow examples for a microcontroller to control an LC717A10 are shown and described.

- The microcontroller reads the measurement result with the INTOUT signal in the “Interval Mode”
- The microcontroller reads the measurement result without the INTOUT signal in the “Interval Mode”
- The microcontroller reads the measurement result with the INTOUT signal in the “Sleep Mode”

Note: In this chapter, the term “Measurement” means that an LC717A10 measures the amount of capacitance variation for each channel which is specified by the Use Channel 1 Register (Address=0x00) and the Use Channel 2 Register (Address=0x01), and performs ON/OFF judgment for each channel, and reflects measurement results to registers such as the Result Data 1 Register (Address=0x2A) and the Result Data 2 Register (Address=0x2B).

Note: When at least one bit of the WriteReq bit, the ParaCh bit and the StaCal bit is not “0”, the microcontroller must not write to the Control 1 Register. (In other word, only when all of the WriteReq bit, the ParaCh bit and the StaCal bit are “0”, the microcontroller is able to write to the Control 1 Register.)

For more information, refer to explanation of the Control 1 Register (Address=0x2F) in this document.

4.1 Control Flow Examples

4.1.1 The microcontroller reads the measurement result with the INTOUT signal in the “Interval Mode”

One example of control sequence is described here.

A typical system configuration corresponding to this example is as follows. The microcontroller is connected to the LC717A10 over I²C (or SPI) bus, and a GPIO port of the microcontroller is connected to the INTOUT pin of the LC717A10, and another GPIO port is connected to the nRST pin.

The microcontroller reads the status of the INTOUT signal via a GPIO port in order to check the completion of the measurement the LC717A10 takes. After detecting the assertion of the INTOUT signal, the microcontroller reads measurement results from the Result Data 1 Register (Address=0x2A), the Result Data 2 Register (Address=0x2B) and so on. The details of the control sequence are as follows.

The microcontroller always sets the LC717A10’s operation mode to “Interval Mode”. (In concrete terms, the microcontroller always writes “1” to the IntMode bit in the Control 1 Register (Address=0x2F))

In addition, both the INTMD1 bit and the INTMD2 bit in the Measurement 1 Register (Address=0x3A) are always set to “0”, so the LC717A10 always asserts the INTOUT immediately after the measurement has completed.

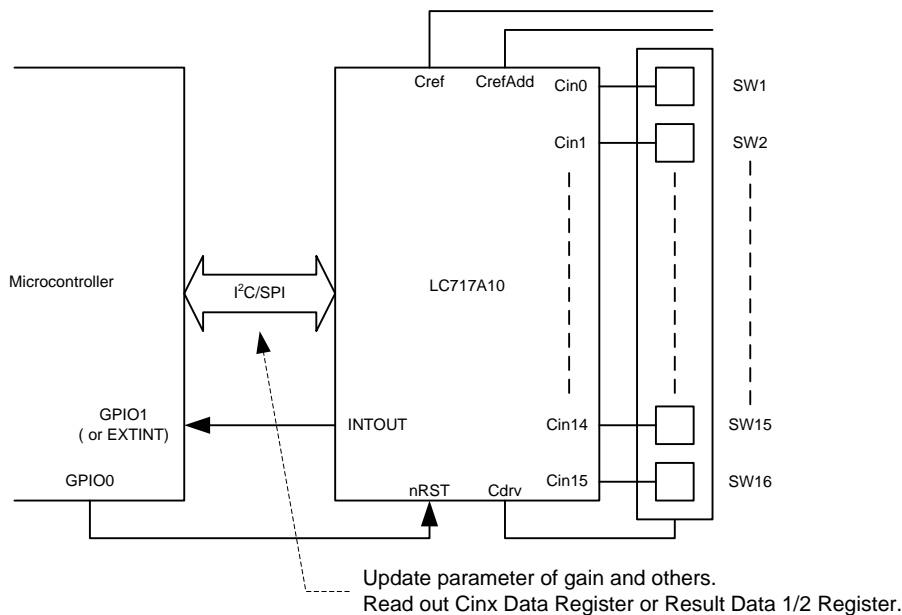


Fig.4-1 Connection between a microcontroller and the LC717A10

- (1) The microcontroller resets the LC717A10 by controlling the nRST signal line via a GPIO port.
Note: In concrete terms, the microcontroller outputs “Low” to nRST signal line, and then it outputs “High” to nRST signal line. The output from the GPIO port of the microcontroller, which is connected to the nRST pin, has to be kept “Low” for at least 1 μ s.
- (2) After controlling the nRST signal line, the microcontroller checks the status of the INTOUT signal via another GPIO port. As soon as the microcontroller detects the rising edge of the INTOUT signal, it sets 0x88 to the Control 1 Register (Address=0x2F) over the I²C (or SPI) bus. To put it concretely, the microcontroller writes 0x88 to the Control 1 Register (Address=0x2F) within 5 ms after the INTOUT signal goes “High”.
Note: By doing (2), the execution of static offset calibration and measurement, which is performed automatically after initialization in the LC717A10, can be cancelled. (Conversely, if you allow the microcontroller to perform the static offset calibration and the subsequent measurement which are performed automatically by the LC717A10 according to the default setting of the LC717A10, the microcontroller need not do the processing described above, but the microcontroller usually has to wait at least until the static offset calibration has completed)
- (3) A microcontroller reads the Control 1 Register (Address=0x2F), and waits until all of the WriteReq bit, the ParaCh bit and the StaCal bit are “0”. By doing (3), the microcontroller can confirm that static offset calibration was completed.
- (4) The microcontroller writes an appropriate parameter value to each register of the LC717A10 over the I²C (or SPI) bus. (For more information about registers the microcontroller may write a parameter to, refer to the inside of the blue dashed line box(a) on the control flow diagram shown in Fig.4-2)
 Especially, in this example, the microcontroller writes “0” to both the INTMD1 bit and the INTMD2 bit in the Measurement 1 Register (Address=0x3A) so that LC717A10 may not negate the INTOUT signal automatically.
Note: At this time, new settings on all the registers are not still reflected to the LC717A10’s operation. (In other words, the LC717A10 still operates not according to the new settings but according to the previous settings)
- (5) To reflect the new settings to the LC717A10’s operation and to make the LC717A10 perform static offset calibration and measurement, the microcontroller writes 0x8F to the Control 1 Register (Address=0x2F) over the I²C (or SPI) bus. This processing request for the LC717A10 is pending until the internal trigger signal which occurred after progress at appointed long interval time.
- (6) As soon as trigger occurred, the new settings on all the registers are reflected to the LC717A10’s operation. (That is to say, the LC717A10 operates according to the new settings)
Note: At this time, the INTOUT is negated (In other words, the LC717A10 outputs “Low” to the INTOUT pin). Furthermore, the Error Status Register (Address=0x2C), the Error Channel Status 1 Register (Address=0x2D) and the Error Channel Status 2 Register (Address=0x2E) are all initialized into 0x00. And counters inside this LSI which are used for the processing of dynamic offset calibration and the processing of debounce are all cleared.
- (7) The LC717A10 performs the static offset calibration based on the new settings. After completing the static offset calibration, the LC717A10 starts measurement.
- (8) In order to check the completion of the measurement in the LC717A10, the microcontroller checks the assertion of the INTOUT signal. After this, the microcontroller writes 0x00 to the Control 2 Register (Address=0x40) to negate the INTOUT signal, and it reads measurement results from the Result Data 1 Register (Address=0x2A) , the Result Data 2 Register (Address=0x2B) and so on.
- (9) After the elapse of a predetermined time (Either short interval time or long interval time) from the completion of the measurement, the LC717A10 starts the next measurement automatically.
- (10) The microcontroller repeats the above (8) through (9).

The control flow diagram corresponding to the control sequence described above is shown on the next page.

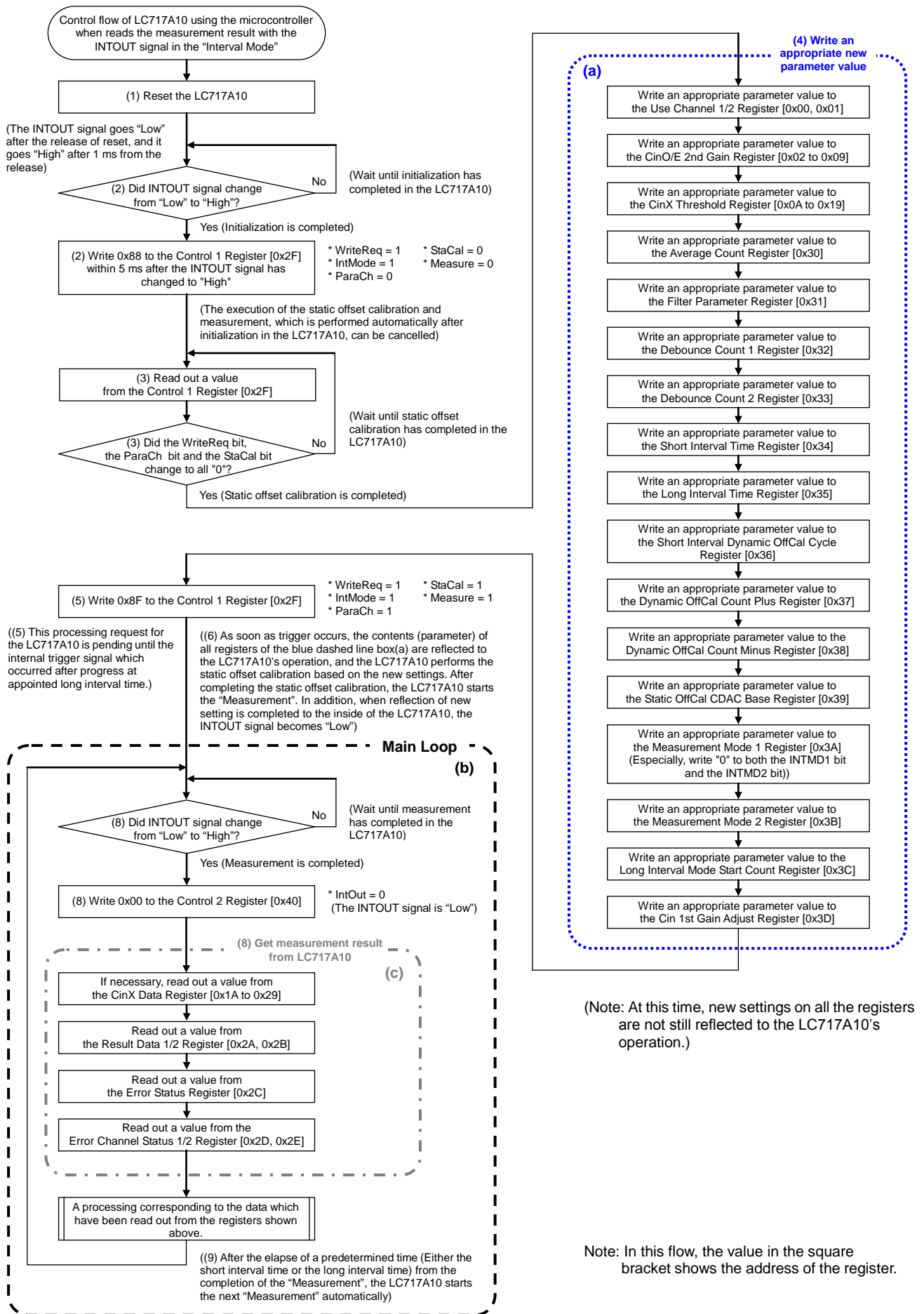


Fig.4-2 Control flow example when the microcontroller reads the measurement result with the INTOUT signal in the "Interval Mode"

4.1.2 The microcontroller reads the measurement result without the INTOUT signal in the “Interval Mode”

Another example of control sequence is described here.

A typical system configuration corresponding to this example is as follows. The microcontroller is connected to the LC717A10 over I²C (or SPI) bus, and a GPIO port of the microcontroller is connected to the nRST pin of the LC717A10. The microcontroller doesn't check the INTOUT signal of the LC717A10.

The microcontroller always sets the LC717A10's operation mode to “Interval Mode”. (In concrete terms, the microcontroller always writes “1” to the IntMode bit in the Control 1 Register (Address=0x2F))

The microcontroller doesn't check the status of INTOUT signal. Without detecting the assertion of the INTOUT signal, the microcontroller reads measurement results periodically (at 50 ms intervals) from the Result Data 1 Register (Address=0x2A), the Result Data 2 Register (Address=0x2B) and so on. The details of the control sequence are as follows.

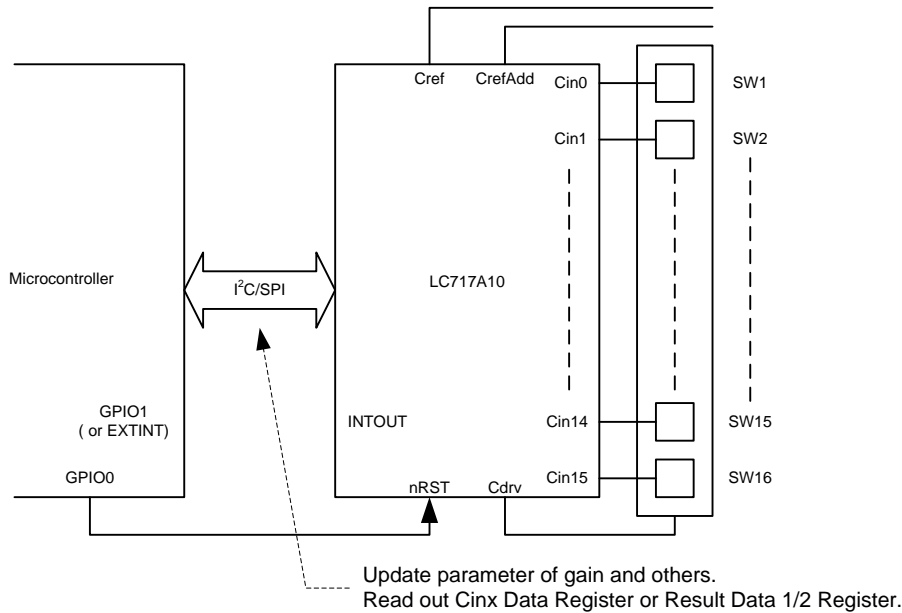


Fig.4-3 Connection between a microcontroller and the LC717A10

- (1) The microcontroller resets the LC717A10 by controlling the nRST signal line via a GPIO port.
Note: In concrete terms, the microcontroller outputs “Low” to nRST signal line, and then it outputs “High” to nRST signal line. The output from the GPIO port of the microcontroller, which is connected to the nRST pin, has to be kept “Low” for at least 1 μs.
- (2) After controlling the nRST signal line, the microcontroller waits until a reasonable amount of time passes since the LC717A10 has gone into static offset calibration. For example, the microcontroller waits for 100 ms from the release of reset. Afterwards, the microcontroller reads the Control 1 Register (Address=0x2F), and waits until all of the WriteReq bit, the ParaCh bit and the StaCal bit are “0”.
- (3) The microcontroller writes an appropriate parameter value to each register of the LC717A10 over the I²C (or SPI) bus. (For more information about registers the microcontroller may write a parameter to, refer to the inside of the blue dashed line box(a) on the control flow diagram shown in Fig.4-4)
 Especially, In this example, the microcontroller writes “0” to both the INTMD1 bit and the INTMD2 bit in the Measurement 1 Register (Address=0x3A) so that LC717A10 may not negate the INTOUT signal automatically.
Note: At this time, new settings on all the registers are not still reflected to the LC717A10's operation. (In other words, the LC717A10 still operates not according to the new settings but according to the previous settings)
- (4) To reflect the new settings to the LC717A10's operation and to make the LC717A10 perform static offset calibration and measurement, the microcontroller writes 0x8F to the Control 1 Register (Address=0x2F) over the I²C (or SPI) bus. This processing request for the LC717A10 is pending until the internal trigger signal which occurred after progress at appointed long interval time.

- (5) As soon as trigger occurred, the new settings on all the registers are reflected to the LC717A10's operation. (That is to say, the LC717A10 operates according to the new settings)
Note: At this time, the Error Status Register (Address=0x2C), the Error Channel Status 1 Register (Address=0x2D) and the Error Channel Status 2 Register (Address=0x2E) are all initialized into 0x00. And counters inside this LSI which are used for the processing of dynamic offset calibration and the processing of debounce are all cleared.
- (6) The LC717A10 performs the static offset calibration based on the new settings. The microcontroller checks that all of the WriteReq bit, the ParaCh bit and the StaCal bit are zero in order to wait for the completion of the static offset calibration in the LC717A10.
- (7) After completing the static offset calibration, the LC717A10 starts measurement.
- (8) The LC717A10 completes the measurement. After the time which is named "Interval time" passed since the completion of the measurement, the LC717A10 restarts next measurement automatically.
- (9) Without detecting the assertion of the INTOUT signal, the microcontroller reads measurement results periodically (For example, at 50 ms intervals) from from the Result Data 1 Register (Address=0x2A), the Result Data 2 Register (Address=0x2B) and so on.
- (10) The LC717A10 repeats the above (8), and the microcontroller repeats the above (9).

The control flow diagram corresponding to the control sequence described above is shown on the next page.

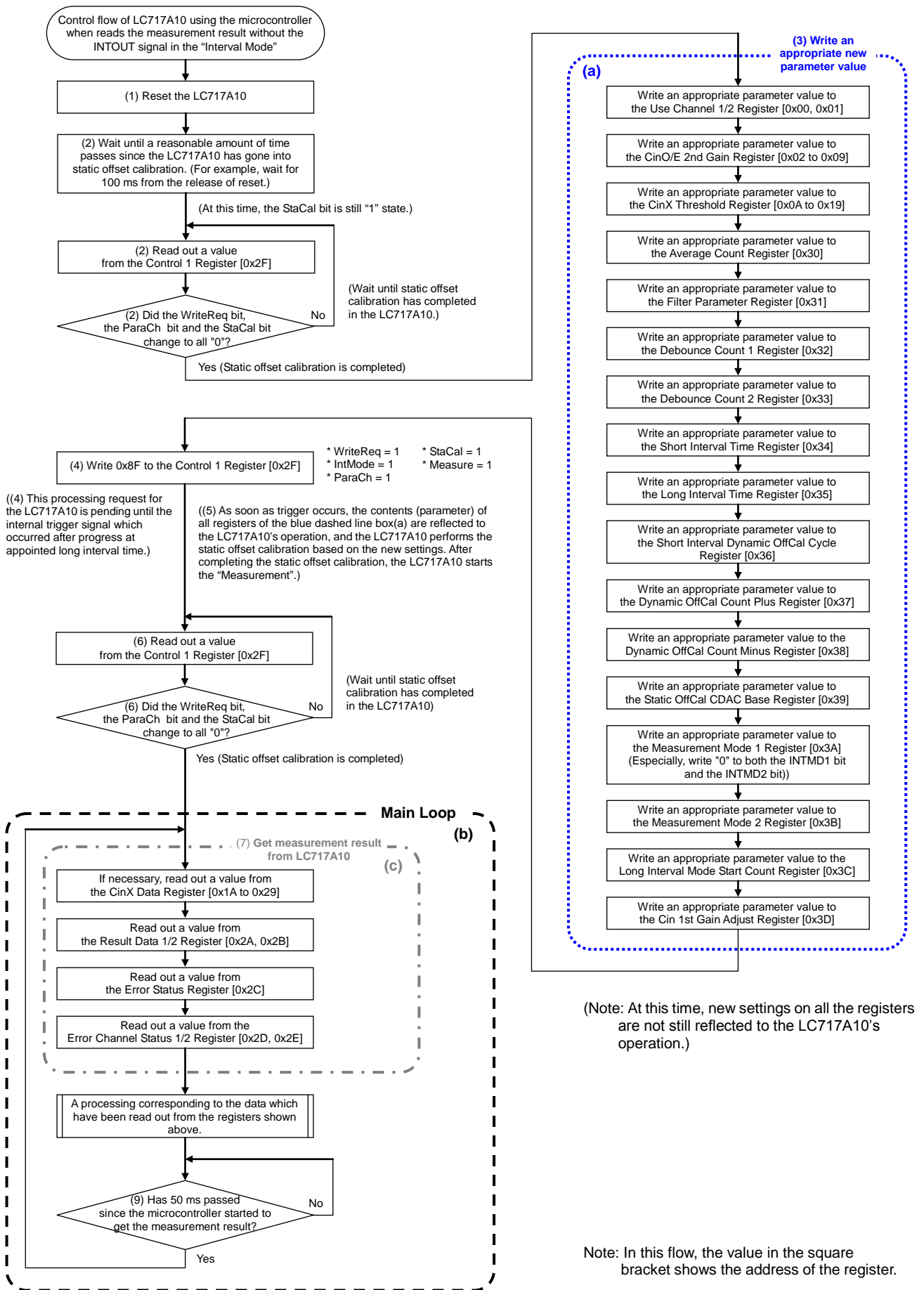


Fig.4-4 Control flow example when the microcontroller reads the measurement result without the INTOUT signal in the "Interval Mode"

4.1.3 The microcontroller reads the measurement result with the INTOUT signal in the “Sleep Mode”

Another example of control sequence is described here.

A typical system configuration corresponding to this example is as follows. The microcontroller is connected to the LC717A10 over I²C (or SPI) bus, and a GPIO port of the microcontroller is connected to the INTOUT pin of the LC717A10, and another GPIO port is connected to the nRST pin.

The microcontroller always sets the LC717A10’s operation mode to “Sleep Mode”. (In concrete terms, the microcontroller always writes “0” to the IntMode bit in the Control 1 Register (Address=0x2F))

In addition, both the INTMD1 bit and the INTMD2 bit in the Measurement 1 Register (Address=0x3A) are always set to “0”, so the LC717A10 always asserts the INTOUT immediately after the measurement has completed.

In order to check that the LC717A10 has completed the measurement and it has gone to sleep, the microcontroller checks the status of INTOUT signal via a GPIO port. After detecting the assertion of the INTOUT signal, the microcontroller reads measurement results from the Result Data 1 Register (Address=0x2A), the Result Data 2 Register (Address=0x2B) and so on, and then it wakes up the LC717A10. The details of the control sequence are as follows.

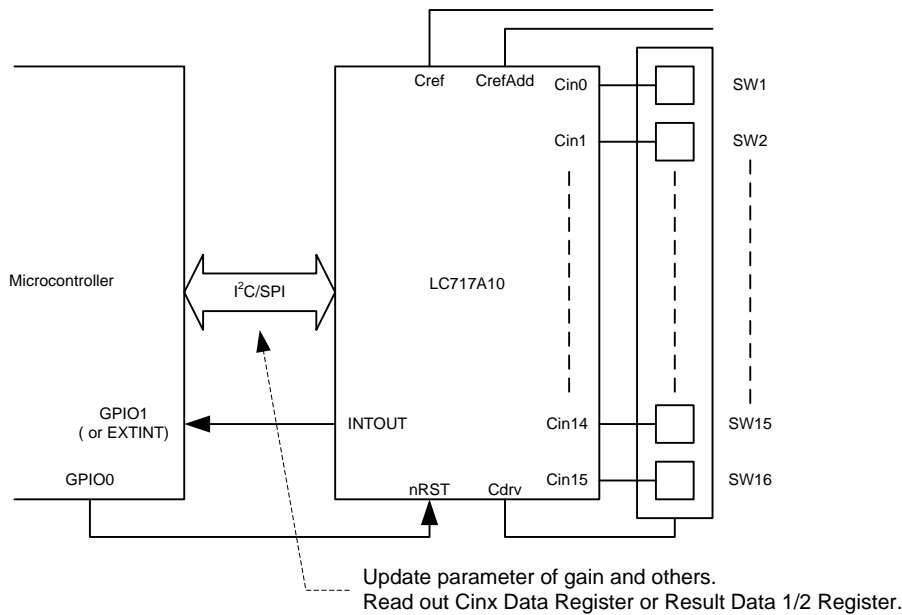


Fig.4-5 Connection between a microcontroller and the LC717A10

- (1) The microcontroller resets the LC717A10 by controlling the nRST signal line via a GPIO port.
Note: In concrete terms, the microcontroller outputs “Low” to nRST signal line, and then it outputs “High” to nRST signal line. The output from the GPIO port of the microcontroller, which is connected to the nRST pin, has to be kept “Low” for at least 1 μs.
- (2) After controlling the nRST signal line, the microcontroller checks the status of the INTOUT signal via another GPIO port. As soon as the microcontroller detects the rising edge of the INTOUT signal, it writes 0x80 to the Control 1 Register (Address=0x2F) over the I²C (or SPI) bus. To put it concretely, the microcontroller writes 0x80 to the Control 1 Register (Address=0x2F) within **5 ms** after the INTOUT signal goes “High”. By doing (2), the execution of static offset calibration and measurement, which is performed automatically after initialization in the LC717A10, can be cancelled.
- (3) A microcontroller reads the Control 1 Register (Address=0x2F), and waits until all of the WriteReq bit, the ParaCh bit and the StaCal bit are “0”. By doing (3), the microcontroller can confirm that static offset calibration was completed.

- (4) The LC717A10 goes to sleep shortly.
 The microcontroller writes an appropriate parameter value to each register of the LC717A10 over the I²C (or SPI) bus. (For more information about registers the microcontroller may write a parameter to, refer to the inside of the blue dashed line box(a) on the control flow diagram shown in Fig.4-6)
 Especially, In this example, the microcontroller writes “0” to both the INTMD1 bit and the INTMD2 bit in the Measurement 1 Register (Address=0x3A) so that LC717A10 may not negate the INTOUT signal automatically.
Note: At this time, new settings on all the registers are not still reflected to the LC717A10’s operation. (In other words, the LC717A10 still operates not according to the new settings but according to the previous settings.)
- (5) The microcontroller writes 0x87 to the Control 1 Register (Address=0x2F) over the I²C (or SPI) bus. And then, in order to negate the INTOUT signal and wake up the LC717A10, the microcontroller writes 0x01 to the Control 2 Register (Address=0x40).
- (6) After (5), the new settings on all the registers are reflected to the LC717A10’s operation. (That is to say, the LC717A10 operates according to the new settings)
Note: At this time, the INTOUT is negated (In other words, the LC717A10 outputs “Low” to the INTOUT pin). Furthermore, the Error Status Register (Address=0x2C), the Error Channel Status 1 Register (Address=0x2D) and the Error Channel Status 2 Register (Address=0x2E) are all initialized into 0x00. And counters inside this LSI which are used for the processing of dynamic offset calibration and the processing of debounce are all cleared.
- (7) The LC717A10 performs the static offset calibration based on the new settings. After completing the static offset calibration, the LC717A10 starts measurement.
- (8) After the completion of measurement, the LC717A10 asserts the INTOUT signal. And then it goes to sleep. The microcontroller checks the assertion of the INTOUT signal in order to check that the LC717A10 has gone to sleep. After detecting the assertion of the INTOUT signal, the microcontroller reads measurement results from the Result Data 1 Register (Address=0x2A), the Result Data 2 Register (Address=0x2B) and so on.
Note: At the time of the completion of the measurement, the LC717A10 outputs “High” to the INTOUT signal line.
- (9) The microcontroller waits until it is time to wake up the LC717A10.
- (10) In order to wake up the LC717A10, the microcontroller writes 0x01 to the Control 2 Register (Address=0x40).
- (11) After (10), the INTOUT signal is negated, and the LC717A10 is waked up by the microcontroller. And then, the LC717A10 start the next measurement.
Note: The WakeUp bit in the Control 2 Register (Address=0x40) is set to “0” automatically at the time when the LC717A10 has been waked up by the microcontroller.
- (12) The microcontroller repeats the above (8) through (11).

Note: The important thing about “Sleep Mode”.

- If the LC717A10 has gone to sleep, the LC717A10 doesn’t resume its operation as long as the microcontroller doesn’t wake up the LC717A10.
- The time interval between the current measurement and the next measurement depends only on the wake-up timing of the LC717A10 by the microcontroller. (In a word, the microcontroller determines the time interval)
- In the case of “Sleep Mode”, the LC717A10 checks neither the Short Interval Time Register (Address=0x34) setting nor the Long Interval Time Register (Address=0x35) setting nor the Long Interval Mode Start Count Register (Address=0x3C) setting. (In other words, the LC717A10 ignores the setting value of those registers)
- When sleep mode is selected, be sure to set Short Interval Dynamic OffCal Cycle Register (Address=0x36) to 0x01. At this time, the cycle of judgments whether or not dynamic offset calibration execution is every measurement point.

The control flow diagram corresponding to the control sequence described above is shown on the next page.

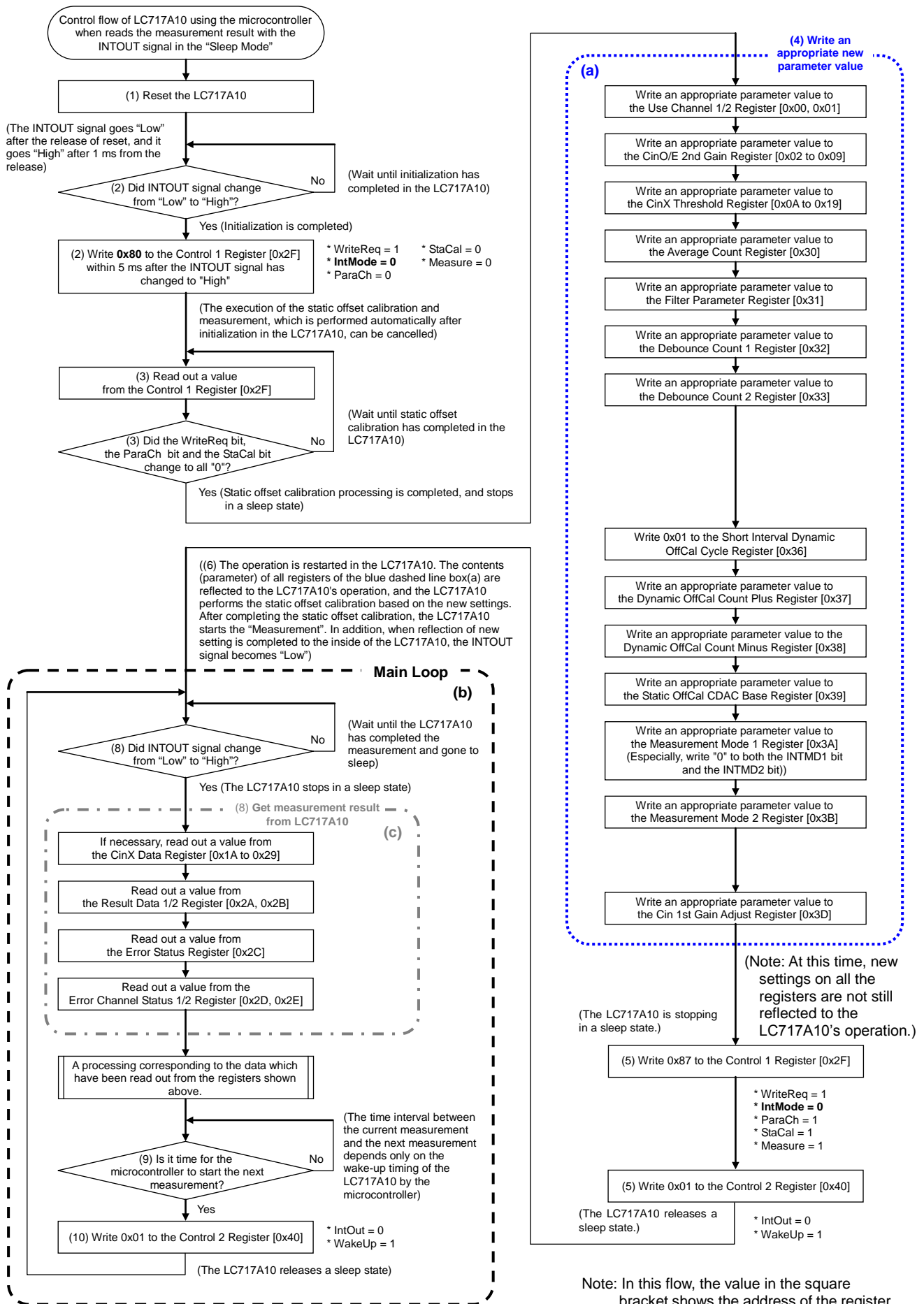


Fig.4-6 Control flow example when the microcontroller reads the measurement result with the INTOUT signal in the "Sleep Mode"

4.2 Pseudo Code Examples for a Microcontroller to Control an LC717A10

DISCLAIMER

Pseudo code examples shown in this document are made only for the help of the software development of customers. You will need to add appropriate code which you think is required for your system.

Even if the use of the program created by referring to the pseudo code sample causes you any loss or damage, our company does not accept any responsibility for the loss or damage at all.

In chapter 4.2, three pseudo code examples are shown here, which could also be implemented similarly in any other programming language.

```
void I2c1byteWrite(unsigned char i2c_adr, unsigned char reg_adr, unsigned char data_w);
```

The job of this function is to write 1 byte data to over the I²C bus.

The first argument "i2c_adr" contains the I²C bus slave address of the slave device targeted.

The second argument "reg_adr" contains the subaddress of the register being accessed within the target device.

The third argument "data_w" contains a 1 byte value to be written.

```
unsigned char I2c1byteRead(unsigned char i2c_adr, unsigned char reg_adr);
```

The job of this function is to read 1 byte data over the I²C bus.

The first argument "i2c_adr" contains the I²C bus slave address of the slave device targeted.

The second argument "reg_adr" contains the sub-address of the register being accessed within the target device.

The function returns the value read as a 1 byte integer.

```
void Wait_until_INTOUT_is_Low( void );
```

The job of this function is to wait until the INTOUT signal level goes "Low".

```
void Wait_until_INTOUT_is_High( void );
```

The job of this function is to wait until the INTOUT signal level goes "High".

```
void nRstCtrl_for_Reset( void );
```

The job of this function is to output "Low" to nRST signal line, and then to output "High" to nRST signal line in order to reset the LC717A10. The output from the GPIO port of the microcontroller, which is connected to the nRST pin, has to be kept "Low" for at least 1 microsecond to reset the LC717A10.

```
unsigned short GetNowTime_msec( void );
```

The job of this function is to get the current time in milliseconds.

The function returns the value read as 2 bytes integer.

```
void Wait_for_100msec( void );
```

The job of this function is to wait for 100 milliseconds.

```
void Wait_for_50msec( void );
```

The job of this function is to wait for 50 milliseconds.

Note: *These functions and these related hardware drivers need be provided by the user.*

AND9594/D

The contents of the header file that is used in common in three pseudo code examples (Which are shown in chapter 4.2) as follows. (File name: "LC717A10.h").

```
//-----  
//          This is an example include file for LC717A10.  
//          (LC717A10.h)  
//-----  
  
#define SLAVE_ADDRESS          (0x16)    // I2C slave address of LC717A10  
//                               // (0x16, 0x17, 0x18 or 0x19)  
  
#define USECHANNEL1_REG        (0x00)    // Address of the Use Channel 1 Register  
#define USECHANNEL2_REG        (0x01)    // Address of the Use Channel 2 Register  
#define CIN1CIN02NDGAIN_REG    (0x02)    // Address of the Cin1/0 2nd Gain Register  
#define CIN3CIN22NDGAIN_REG    (0x03)    // Address of the Cin3/2 2nd Gain Register  
#define CIN5CIN42NDGAIN_REG    (0x04)    // Address of the Cin5/4 2nd Gain Register  
#define CIN7CIN62NDGAIN_REG    (0x05)    // Address of the Cin7/6 2nd Gain Register  
#define CIN9CIN82NDGAIN_REG    (0x06)    // Address of the Cin9/8 2nd Gain Register  
#define CIN11CIN102NDGAIN_REG  (0x07)    // Address of the Cin11/10 2nd Gain Register  
#define CIN13CIN122NDGAIN_REG  (0x08)    // Address of the Cin13/12 2nd Gain Register  
#define CIN15CIN142NDGAIN_REG  (0x09)    // Address of the Cin15/14 2nd Gain Register  
#define CIN0THRESHOLD_REG      (0x0A)    // Address of the Cin0 Threshold Register  
#define CIN1THRESHOLD_REG      (0x0B)    // Address of the Cin1 Threshold Register  
#define CIN2THRESHOLD_REG      (0x0C)    // Address of the Cin2 Threshold Register  
#define CIN3THRESHOLD_REG      (0x0D)    // Address of the Cin3 Threshold Register  
#define CIN4THRESHOLD_REG      (0x0E)    // Address of the Cin4 Threshold Register  
#define CIN5THRESHOLD_REG      (0x0F)    // Address of the Cin5 Threshold Register  
#define CIN6THRESHOLD_REG      (0x10)    // Address of the Cin6 Threshold Register  
#define CIN7THRESHOLD_REG      (0x11)    // Address of the Cin7 Threshold Register  
#define CIN8THRESHOLD_REG      (0x12)    // Address of the Cin8 Threshold Register  
#define CIN9THRESHOLD_REG      (0x13)    // Address of the Cin9 Threshold Register  
#define CIN10THRESHOLD_REG     (0x14)    // Address of the Cin10 Threshold Register  
#define CIN11THRESHOLD_REG     (0x15)    // Address of the Cin11 Threshold Register  
#define CIN12THRESHOLD_REG     (0x16)    // Address of the Cin12 Threshold Register  
#define CIN13THRESHOLD_REG     (0x17)    // Address of the Cin13 Threshold Register  
#define CIN14THRESHOLD_REG     (0x18)    // Address of the Cin14 Threshold Register  
#define CIN15THRESHOLD_REG     (0x19)    // Address of the Cin15 Threshold Register  
#define CIN0DATA_REG           (0x1A)    // Address of the Cin0 Data Register  
#define CIN1DATA_REG           (0x1B)    // Address of the Cin1 Data Register  
#define CIN2DATA_REG           (0x1C)    // Address of the Cin2 Data Register  
#define CIN3DATA_REG           (0x1D)    // Address of the Cin3 Data Register  
#define CIN4DATA_REG           (0x1E)    // Address of the Cin4 Data Register  
#define CIN5DATA_REG           (0x1F)    // Address of the Cin5 Data Register  
#define CIN6DATA_REG           (0x20)    // Address of the Cin6 Data Register  
#define CIN7DATA_REG           (0x21)    // Address of the Cin7 Data Register  
#define CIN8DATA_REG           (0x22)    // Address of the Cin8 Data Register  
#define CIN9DATA_REG           (0x23)    // Address of the Cin9 Data Register  
#define CIN10DATA_REG          (0x24)    // Address of the Cin10 Data Register  
#define CIN11DATA_REG          (0x25)    // Address of the Cin11 Data Register  
#define CIN12DATA_REG          (0x26)    // Address of the Cin12 Data Register  
#define CIN13DATA_REG          (0x27)    // Address of the Cin13 Data Register  
#define CIN14DATA_REG          (0x28)    // Address of the Cin14 Data Register  
#define CIN15DATA_REG          (0x29)    // Address of the Cin15 Data Register  
#define RSLTDATA1_REG          (0x2A)    // Address of the Result Data 1 Register  
#define RSLTDATA2_REG          (0x2B)    // Address of the Result Data 2 Register  
#define ERRSTS_REG             (0x2C)    // Address of the Error Status Register  
#define ERRCHSTS1_REG          (0x2D)    // Address of the Error Channel Status 1 Register  
#define ERRCHSTS2_REG          (0x2E)    // Address of the Error Channel Status 2 Register  
#define CONTROL1_REG           (0x2F)    // Address of the Control 1 Register  
#define AVECNT_REG             (0x30)    // Address of the Average Count Register  
#define FILPRM_REG             (0x31)    // Address of the Filter Parameter Register  
#define DEBCNT1_REG            (0x32)    // Address of the Debounce Count 1 Register  
#define DEBCNT2_REG            (0x33)    // Address of the Debounce Count 2 Register  
#define SHORTITVL_REG          (0x34)    // Address of the Short Interval Time Register  
#define LONGITVL_REG           (0x35)    // Address of the Long Interval Time Register  
#define SIDCALCYCLE_REG        (0x36)    // Address of the Short Interval  
//                               // Dynamic OffCal Cycle Register  
#define DCALCNTP_REG           (0x37)    // Address of the Dynamic OffCal Count Plus Register  
#define DCALCNTM_REG           (0x38)    // Address of the Dynamic OffCal Count Minus Register  
#define SCALCDACBASE_REG       (0x39)    // Address of the Static OffCal Base Register  
#define MEASMODE1_REG          (0x3A)    // Address of the Measurement 1 Mode Register  
#define MEASMODE2_REG          (0x3B)    // Address of the Measurement 2 Mode Register  
#define LITVLMODESTACNT_REG    (0x3C)    // Address of the Long Interval Mode Start Count Register  
#define CIN1STGAINADJ_REG      (0x3D)    // Address of the Cin 1st Gain Adjust Register  
#define CONTROL2_REG           (0x40)    // Address of the Control 2 Register
```

AND9594/D

```
//For the Control 1 Register
#define WRITE_REQ_BIT      (0x80)    // The WriteReq bit
#define INT_MODE_BIT      (0x08)    // The IntMode bit
#define PARA_CH_BIT      (0x04)    // The ParaCh bit
#define STA_CAL_BIT      (0x02)    // The StaCal bit
#define MEASURE_BIT      (0x01)    // The Measure bit

//For the Control 2 Register
#define WAKEUP_BIT      (0x01)    // The WakeUp bit
```


4.2.1 Pseudo Code Example When a Microcontroller Reads the Measurement Result With the INTOUT Signal in the "Interval Mode"

Note: Generally, it is recommended for the LC717A10 to operate in "Interval Mode" rather than in "Sleep Mode".

Here is one pseudo code example for a microcontroller when a microcontroller is connected to an LC717A10 over I²C (or SPI) bus, and a GPIO port of the microcontroller is connected to the INTOUT pin of the LC717A10, and another GPIO port is connected to the nRST pin.

In this pseudo code example, the microcontroller always sets the LC717A10's operation mode to "Interval Mode". In addition, both the INTMD1 bit and the INTMD2 bit in the Measurement 1 Register (Address=0x3A) are always set to "0", so the LC717A10 always asserts the INTOUT immediately after the measurement has completed.

The microcontroller checks the status of the INTOUT signal via a GPIO port in order to check the completion of the measurement the LC717A10 takes. After detecting the assertion of the INTOUT signal, the microcontroller reads measurement results from the Result Data 1 Register (Address=0x2A), the Result Data 2 Register (Address=0x2B) and so on.

```
//-----
//--
//-- This is a pseudo-code example.
//-- in the case that a microcontroller is connected to an LC717A10
//-- over an I2C bus and it checks the status of the INTOUT signal.
//--
//-----

// Include file
#define "LC717A10.h"

//Declaration of function prototypes
void nRstCtrl_for_Reset( void );
void Wait_until_INTOUT_is_Low( void );
void Wait_until_INTOUT_is_High( void );
void I2c1byteWrite( unsigned char, unsigned char, unsigned char );
unsigned char I2c1byteRead( unsigned char, unsigned char );

void main( void )
{
    unsigned char w_data, cnt, rslt_data;
    unsigned char err_sts, errch_sts;

    /*-----
    /** The microcontroller resets the LC717A10
    /** and the completion of initialization in the inside ofLC717A10.
    /**-----
    //The microcontroller resets the LC717A10 by controlling
    //the nRST signal line via a GPIO port.
    //(In concrete terms, the microcontroller outputs "Low" to nRST signal line,
    // and then it outputs "High" to nRST signal line.
    // The output from the GPIO port of the microcontroller, which is connected
    // to the nRST pin, has to be kept "Low" for at least 1 microsecond.)
    nRstCtrl_for_Reset();

    /*-----
    /** The microcontroller checks both the release of reset of the LC717A10
    /** and the completion of initialization in the inside ofLC717A10.
    /**-----
    //The microcontroller checks the release of reset of the LC717A10.
    //(In concrete terms, the microcontroller checks that the INTOUT signal is "Low"
    // with a GPIO port which is connected to INTOUT pin of the LC717A10.)
    Wait_until_INTOUT_is_Low(); //The microcontroller waits until the INTOUT signal is "Low".

    //-----
    //-- Note: Notes in the case that you select "SPI" as a interface
    //-- If you select not "I2C" but "SPI" as a interface, please output "Low"
    //-- to the nCS pin for at least 1 microsecond at this timing(that is,
    //-- at the timing when the microcontroller has detected that the INTOUT signal
    //-- goes "Low").
    //--
```

AND9594/D

```
//--
//-- (In short, please output "High" to the nCS pin, and then output "Low",
//-- and then output "High" again.)
//-----
// nCS = 1;
// nCS = 0;
// nCS = 1;
//

//The microcontroller checks the completion of initialization in the inside of LC717A10.
//(In concrete terms, the microcontroller checks that the INTOUT signal is "High"
// with a GPIO port which is connected to the INTOUT pin of the LC717A10.)
Wait_until_INTOUT_is_High(); //The microcontroller waits until the INTOUT signal is "High".

//-----
//-- Note: In short, the microcontroller only has to detect
//-- the rising edge of the INTOUT signal.
//-----

/*****
/** The microcontroller cancels the execution of both static offset calibration
/** and measurement, which is performed in the LC717A10 automatically
/** after initialization of the inside of LC717A10.
*****/
//Within 5 milliseconds after the microcontroller detects the rising edge of the
//INTOUT signal, it sets 0x88 to the Control 1 Register in order to cancel the execution
//of both static offset calibration and measurement.
w_data = (WRITE_REQ_BIT | INT_MODE_BIT);
I2c1byteWrite( SLAVE_ADDRESS, CONTROL1_REG, w_data );

//Before setting parameters, the microcontroller checks that
//the static-offset-calibration has completed in LC717A10.
for(;;)
{
    r_data = I2c1byteRead( SLAVE_ADDRESS, CONTROL1_REG );

    //The microcontroller waits until all of the WriteReq bit, the ParaCh bit
    //and the StaCal bit are zero.
    if( ( r_data & ( WRITE_REQ_BIT | PARA_CH_BIT | STA_CAL_BIT ) ) == 0x00 )
    {
        break;
    }
}

/*****
/** The microcontroller sets parameters.
/** (This is an example for setting parameters.)
*****/
//1.The microcontroller sets 'the channels to be used' to Cin0-Cin5.
//2.The microcontroller sets the CinX/X 2nd Gain Registers associated with Cin0-Cin5
// to 0x11. (2nd stage AMP gain = X 2)
//3.The microcontroller sets threshold level of Cin0-Cin5 to 10.
//4.The microcontroller sets the Average Count Register to 0x80.
// (Average count = 128)
//5.The microcontroller sets the Filter Parameter Register to 0x0C.
//6.The microcontroller sets the Measurement 1 Register to 0x00
// so that the LC717A10 may assert the INTOUT signal after every measurement.
//7.The microcontroller sets the Cin 1st Gain Adjust Register to 0x00.
// (1st stage AMP gain at all channels = 1600[FF]).

//The microcontroller sets 'the channels to be used' to Cin0-Cin5.
I2c1byteWrite( SLAVE_ADDRESS, USECHANNEL1_REG, 0x3F );
I2c1byteWrite( SLAVE_ADDRESS, USECHANNEL2_REG, 0x00 );

//The microcontroller sets the CinX/X 2nd Gain Registers associated with Cin0-Cin5 to 0x11.
// (2nd stage AMP gain = X 2).
w_data = 0x11;
for( cnt=0; cnt<=2; cnt++ )
{
    I2c1byteWrite( SLAVE_ADDRESS, (CIN1CIN02NDGAIN_REG+cnt), w_data );
}

//The microcontroller sets threshold level of Cin0-Cin5 to 10.
w_data = 0x0A;
for( cnt=0; cnt<=5; cnt++ )
{
```

AND9594/D

```
I2c1byteWrite( SLAVE_ADDRESS, (CIN0THRESHOLD_REG+cnt), w_data );
}

//The microcontroller sets the Average Count Register to 0x80.
// (Average count = 128)
I2c1byteWrite( SLAVE_ADDRESS, AVECNT_REG, 0x80 );

//The microcontroller sets the Filter Parameter Register to 0x0C.
I2c1byteWrite( SLAVE_ADDRESS, FILPRM_REG, 0x0C );

//The microcontroller sets the Measurement 1 Register to 0x00
// so that the LC717A10 may assert the INTOUT signal after every measurement.
I2c1byteWrite( SLAVE_ADDRESS, MEASMODE1_REG, 0x00 );

//The microcontroller sets the Cin 1st Gain Adjust Register to 0x00.
// (1st stage AMP gain at all channels = 1600[ff]).
I2c1byteWrite( SLAVE_ADDRESS, CIN1STGAINADJ_REG, 0x00 );

//-----
//-- Note: The microcontroller usually sets          --
//-- other registers in the same way as described above. --
//-----

// If necessary, after doing the parameter-setting processing,
// please verify all the register settings that you have set by reading back them.

/*****
/* The microcontroller issues a request to the LC717A10 for setting of parameters, */
/* static-offset-calibration, measurement and selection of "interval mode".      */
*****/
w_data = (WRITE_REQ_BIT | INT_MODE_BIT | PARA_CH_BIT | STA_CAL_BIT | MEASURE_BIT);
I2c1byteWrite( SLAVE_ADDRESS, CONTROL1_REG, w_data );

/*****
** The microcontroller checks the status of the INTOUT signal          **
** in order to check the completion of the current measurement.        **
** After the microcontroller has detected the assertion of the INTOUT signal, **
** the microcontroller reads measurement results over the I2C bus.      **
** (Note: measurement results include the judgment result of ON/OFF status, **
** error status and so on.)                                           **
*****/
for(;;)
{
    //The microcontroller waits until the INTOUT signal is "High".
    //(In other words, the microcontroller waits until the measurement has completed
    // in the LC717A10.)
    Wait_until_INTOUT_is_High();

    //The microcontroller writes 0x00 to the Control 2 Register.
    I2c1byteWrite( SLAVE_ADDRESS, CONTROL2_REG, 0x00 );

    /*****
    ** The microcontroller reads measurement results. **
    *****/
    //If necessary, the microcontroller reads the CinX Data Register.

    //The microcontroller reads the Result Data 1 Register.
    rslt_data = I2c1byteRead( SLAVE_ADDRESS, RSLTDATA1_REG );

    //The microcontroller reads the Error Status Register.
    err_sts = I2c1byteRead( SLAVE_ADDRESS, ERRSTS_REG );

    //The microcontroller reads the Error Channel Status 1 Register.
    errch_sts = I2c1byteRead( SLAVE_ADDRESS, ERRCHSTS1_REG );

    //The code for processing the measurement results should be described here.
    // :
    // :
    // :
    // :
    // :
}
}
```

4.2.2 Pseudo Code Example When a Microcontroller Reads the Measurement Result Without the INTOUT Signal in the “Interval Mode”

Note: Generally, it is recommended for the LC717A10 to operate in “Interval Mode” rather than in “Sleep Mode”.

Here is another pseudo code example for a microcontroller when a microcontroller is connected to an LC717A10 over I²C (or SPI) bus, and a GPIO port of the microcontroller is connected to the nRST pin of the LC717A10.

In this pseudo code example, the microcontroller always sets the LC717A10’s operation mode to “Interval Mode”.

The microcontroller doesn’t check the status of INTOUT signal. Without detecting the assertion of the INTOUT signal, the microcontroller reads measurement results from the Result Data 1 Register (Address=0x2A), the Result Data 2 Register (Address=0x2B) and so on at 50 ms intervals.

```
//-----
//--
//-- This is a pseudo-code example. --
//-- in the case that a microcontroller is connected to an LC717A10 --
//-- over an I2C bus and it does not check the status of the INTOUT signal. --
//-- --
//-----

// Include file
#define "LC717A10.h"

//Declaration of function prototypes
void nRstCtrl_for_Reset( void );
void Wait_for_100msec( void );
void I2cbyteWrite( unsigned char, unsigned char, unsigned char );
unsigned char I2cbyteRead( unsigned char, unsigned char );
unsigned short GetNowTime_msec( void );

void main( void )
{
    unsigned char r_data, w_data, cnt, rslt_data;
    unsigned char err_sts, errch_sts;
    unsigned short sta_time_msec, now_time_msec;

    /*-----
    /** The microcontroller resets the LC717A10 */
    /*-----
    //The microcontroller resets the LC717A10 by controlling
    //the nRST signal line via a GPIO port.
    //(In concrete terms, the microcontroller outputs "Low" to nRST signal line,
    // and then it outputs "High" to nRST signal line.
    // The output from the GPIO port of the microcontroller, which is connected
    // to the nRST pin, has to be kept "Low" for at least 1 microsecond.)
    nRstCtrl_for_Reset();

    /*-----
    /** The microcontroller waits until static-offset-
    /** calibration has completed in LC717A10 */
    /*-----
    //The microcontroller waits until a reasonable amount of time passes
    //since LC717A10 has gone into static-offset-calibration
    //(For example, the microcontroller waits for 100 milliseconds.)
    Wait_for_100msec();

    //-----
    //-- Note: Notes in the case that you select "SPI" as a interface --
    //-- If you select not "I2C" but "SPI" as a interface, please output "Low" --
    //-- to the nCS pin for at least 1 microsecond at any time after the internal --
    //-- initialization in the LC717A10 has completed. --
    //-- --
    //-- (For example, the microcontroller waits for 50 milliseconds after the --
    //-- release of the reset, outputs "High" to the nCS pin, and then outputs --
    //-- "Low", and then outputs "High" again, and then waits for 50 milliseconds. --
    //-- --
    //-----
```

AND9594/D

```
//-----  
// Wait_for_50msec();  
// nCS = 1;  
// nCS = 0;  
// nCS = 1;  
// Wait_for_50msec();  
//  
  
//Before setting parameters, the microcontroller checks that  
//the static-offset-calibration has completed in LC717A10.  
for(;;)  
{  
    r_data = I2c1byteRead( SLAVE_ADDRESS, CONTROL1_REG );  
  
    //The microcontroller waits until all of the WriteReq bit, the ParaCh bit  
    //and the StaCal bit are zero.  
    if( ( r_data & ( WRITE_REQ_BIT | PARA_CH_BIT | STA_CAL_BIT ) ) == 0x00 )  
    {  
        break;  
    }  
}  
  
/*****  
**    The microcontroller sets parameters.    **  
**    (This is an example for setting parameters.)    **  
*****/  
//1.The microcontroller sets 'the channels to be used' to Cin0-Cin5.  
//2.The microcontroller sets the CinX/X 2nd Gain Registers associated with Cin0-Cin5  
//   to 0x11. (2nd stage AMP gain = X 2)  
//3.The microcontroller sets threshold level of Cin0-Cin5 to 10.  
//4.The microcontroller sets the Average Count Register to 0x80.  
//   (Average count = 128)  
//5.The microcontroller sets the Filter Parameter Register to 0x0C.  
//6.The microcontroller sets the Cin 1st Gain Adjust Register to 0x00.  
//   (1st stage AMP gain at all channels = 1600[ff]).  
  
//The microcontroller sets 'the channels to be used' to Cin0-Cin5.  
I2c1byteWrite( SLAVE_ADDRESS, USECHANNEL1_REG, 0x3F );  
I2c1byteWrite( SLAVE_ADDRESS, USECHANNEL2_REG, 0x00 );  
  
//The microcontroller sets the CinX/X 2nd Gain Registers associated with Cin0-Cin5 to 0x11.  
//   (2nd stage AMP gain = X 2).  
w_data = 0x11;  
for( cnt=0; cnt<=2; cnt++ )  
{  
    I2c1byteWrite( SLAVE_ADDRESS, (CIN1CIN02NDGAIN_REG+cnt), w_data );  
}  
  
//The microcontroller sets threshold level of Cin0-Cin5 to 10.  
w_data = 0x0A;  
for( cnt=0; cnt<=5; cnt++ )  
{  
    I2c1byteWrite( SLAVE_ADDRESS, (CIN0THRESHOLD_REG+cnt), w_data );  
}  
  
//The microcontroller sets the Average Count Register to 0x80.  
//   (Average count = 128)  
I2c1byteWrite( SLAVE_ADDRESS, AVECNT_REG, 0x80 );  
  
//The microcontroller sets the Filter Parameter Register to 0x0C.  
I2c1byteWrite( SLAVE_ADDRESS, FILPRM_REG, 0x0C );  
  
//The microcontroller sets the Measurement 1 Register to 0x00  
// so that the LC717A10 may assert the INTOUT signal after every measurement.  
I2c1byteWrite( SLAVE_ADDRESS, MEASMODEL_REG, 0x00 );  
  
//The microcontroller sets the Cin 1st Gain Adjust Register to 0x00.  
//   (1st stage AMP gain at all channels = 1600[ff]).  
I2c1byteWrite( SLAVE_ADDRESS, CIN1STGAINADJ_REG, 0x00 );  
  
//-----  
//-- Note: The microcontroller usually sets    --  
//-- other registers in the same way as described above. --  
//-----  
  
// If necessary, after doing the parameter-setting processing,  
// please verify all the register settings that you have set by reading back them.
```

AND9594/D

```

/*****
/* The microcontroller issues a request to the LC717A10 for setting of parameters, */
/* static-offset-calibration, measurement and selection of "interval mode". */
/*****
w_data = (WRITE_REQ_BIT | INT_MODE_BIT | PARA_CH_BIT | STA_CAL_BIT | MEASURE_BIT);
I2c1byteWrite( SLAVE_ADDRESS, CONTROL1_REG, w_data );

/*****
/** The microcontroller waits until the static offset calibration */
/** has completed in LC717A10. */
/*****
for(;;)
{
    r_data = I2c1byteRead( SLAVE_ADDRESS, CONTROL1_REG );

    //The microcontroller checks that all of the WriteReq, the ParaCh
    //and the StaCal bit are zero.
    if( ( r_data & ( WRITE_REQ_BIT | PARA_CH_BIT |STA_CAL_BIT) ) == 0x00 )
    {
        break;
    }
}

/*****
/** Without checking the status of the INTOUT signal, the microcontroller */
/** reads measurement results periodically over the I2C bus */
/** (Note: measurement results include the judgment result of ON/OFF status, */
/** error status and so on.) */
/*****
//The microcontroller gets the time at this point.(in milliseconds)
sta_time_msec = GetNowTime_msec();
for(;;)
{
    /*****
    /** The microcontroller reads measurement results. */
    /*****
    //If necessary, the microcontroller reads the CinX Data Register.

    //The microcontroller reads the Result Data 1 Register.
    rslt_data = I2c1byteRead( SLAVE_ADDRESS, RSLTDATA1_REG );

    //The microcontroller reads the Error Status Register.
    err_sts = I2c1byteRead( SLAVE_ADDRESS, ERRSTS_REG );

    //The microcontroller reads the Error Channel Status 1 Register.
    errrch_sts = I2c1byteRead( SLAVE_ADDRESS, ERRCHSTS1_REG );

    //The code for processing the measurement results should be described here.
    //
    //
    //
    //
    //

    /*****
    /** The microcontroller waits until a predetermined period of time */
    /** (for example, 50 milliseconds) has passed since the microcontroller */
    /** started to get the measurement results. */
    /*****
    for(;;)
    {
        //The microcontroller gets the time at this point.(in milliseconds)
        now_time_msec = GetNowTime_msec();

        if( (now_time_msec - sta_time_msec)>=50 )
        {
            break;
        }
    }

    //The microcontroller updates the variable named "sta_time_sec".
    sta_time_msec = now_time_msec;
}
}

```

4.2.3 Pseudo Code Example When a Microcontroller Reads the Measurement Result With the INTOUT Signal in the "Sleep Mode"

Note: Generally, it is recommended for the LC717A10 to operate in "Interval Mode" rather than in "Sleep Mode".

Here is one pseudo code example for a microcontroller when a microcontroller is connected to an LC717A10 over I²C (or SPI) bus, and a GPIO port of the microcontroller is connected to the INTOUT pin of the LC717A10, and another GPIO port is connected to the nRST pin.

In this pseudo code example, the microcontroller always sets the LC717A10's operation mode to "Sleep Mode". In addition, both the INTMD1 bit and the INTMD2 bit in the measurement 1 Register (Address=0x3A) are always set to "0", so the LC717A10 always asserts the INTOUT immediately after the measurement has completed.

The microcontroller checks the status of the INTOUT signal via a GPIO port in order to check the completion of the measurement the LC717A10 takes (That is, in order to check that the LC717A10 has gone to sleep). After detecting the assertion of the INTOUT signal, the microcontroller reads measurement results from the Result Data 1 Register (Address=0x2A), the Result Data 2 Register (Address=0x2B) and so on.

```
//-----
//--
//-- This is a pseudo-code example in which the microcontroller --
//-- always sets the LC717A10's operation mode to "sleep" mode. --
//--
//-----

// Include file
#define "LC717A10.h"

//Declaration of function prototypes
void nRstCtrl_for_Reset( void );
void Wait_until_INTOUT_is_Low( void );
void Wait_until_INTOUT_is_High( void );
void I2cbyteWrite( unsigned char, unsigned char, unsigned char );
unsigned char I2cbyteRead( unsigned char, unsigned char );

void main( void )
{
    unsigned char w_data, cnt, rslt_data;
    unsigned char err_sts, errch_sts;

    /*-----
    /** The microcontroller resets the LC717A10          **/
    /*-----
    //The microcontroller resets the LC717A10 by controlling
    //the nRST signal line via a GPIO port.
    //(In concrete terms, the microcontroller outputs "Low" to nRST signal line,
    // and then it outputs "High" to nRST signal line.
    // The output from the GPIO port of the microcontroller, which is connected
    // to the nRST pin, has to be kept "Low" for at least 1 microsecond.)
    nRstCtrl_for_Reset();

    /*-----
    /** The microcontroller checks both the release of reset of the LC717A10          **/
    /** and the completion of initialization in the inside of LC717A10.          **/
    /*-----
    //The microcontroller checks the release of reset of the LC717A10.
    //(In concrete terms, the microcontroller checks that the INTOUT signal is "Low"
    // with a GPIO port which is connected to INTOUT pin of the LC717A10.)
    Wait_until_INTOUT_is_Low(); //The microcontroller waits until the INTOUT signal is "Low".

    //-----
    //-- Note: Notes in the case that you select "SPI" as a interface --
    //-- If you select not "I2C" but "SPI" as a interface, please output "Low" --
    //-- to the nCS pin for at least 1 microsecond at this timing(that is, --
    //-- at the timing when the microcontroller has detected that the INTOUT signal --
    //-- goes "Low"). --
    //-----
```

AND9594/D

```
//--
//-- (In short, please output "High" to the nCS pin, and then output "Low",
//-- and then output "High" again.)
//-----
// nCS = 1;
// nCS = 0;
// nCS = 1;
//

//The microcontroller checks the completion of initialization in the inside of LC717A10.
//(In concrete terms, the microcontroller checks that the INTOUT signal is "High"
// with a GPIO port which is connected to the INTOUT pin of the LC717A10.)
Wait_until_INTOUT_is_High(); //The microcontroller waits until the INTOUT signal is "High".

//-----
//-- Note: In short, the microcontroller only has to detect
//-- the rising edge of the INTOUT signal.
//-----

// If necessary, after doing the parameter-setting processing,
// please verify all the register settings that you have set by reading back them.

/*****
/** The microcontroller cancels the execution of both static offset calibration
/** and measurement, which is performed in the LC717A10 automatically
/** after initialization of the inside of LC717A10,
/** Furthermore, the microcontroller makes the LC717A10 go to sleep.
*****/
//Within 5 milliseconds after the microcontroller detects the rising edge of the
//INTOUT signal, it sets 0x80 to the Control 1 Register in order to cancel the execution
//of both static offset calibration and measurement.
w_data = WRITE_REQ_BIT;
I2c1byteWrite( SLAVE_ADDRESS, CONTROL1_REG, w_data );

//Before setting parameters, the microcontroller checks that
//the static-offset-calibration has completed in LC717A10.
for(;;)
{
    r_data = I2c1byteRead( SLAVE_ADDRESS, CONTROL1_REG );

    //The microcontroller waits until all of the WriteReq bit, the ParaCh bit
    //and the StaCal bit are zero.
    if( ( r_data & ( WRITE_REQ_BIT | PARA_CH_BIT | STA_CAL_BIT ) ) == 0x00 )
    {
        break;
    }
}

/*****
/** The microcontroller sets parameters.
/** (This is an example for setting parameters.)
*****/
//1.The microcontroller sets 'the channels to be used' to Cin0-Cin5.
//2.The microcontroller sets the CinX/X 2nd Gain Registers associated with Cin0-Cin5
// to 0x11. (2nd stage AMP gain = X 2)
//3.The microcontroller sets threshold level of Cin0-Cin5 to 10.
//4.The microcontroller sets the Average Count Register to 0x80.
// (Average count = 128)
//5.The microcontroller sets the Filter Parameter Register to 0x0C.
//6.The microcontroller sets the measurement 1 Register to 0x00
// so that the LC717A10 may assert the INTOUT signal after every measurement.
//7.The microcontroller sets the CinX 1st Gain Adjust Register to 0x00.
// (1st stage AMP gain at all channels = 1600[ff]).

//The microcontroller sets 'the channels to be used' to Cin0-Cin5.
I2c1byteWrite( SLAVE_ADDRESS, USECHANNEL1_REG, 0x3F );
I2c1byteWrite( SLAVE_ADDRESS, USECHANNEL2_REG, 0x00 );

//The microcontroller sets the CinX/X 2nd Gain Registers associated with Cin0-Cin5 to 0x11.
// (2nd stage AMP gain = X 2).
w_data = 0x11;
for( cnt=0; cnt<=2; cnt++ )
{
    I2c1byteWrite( SLAVE_ADDRESS, (CIN1CIN02NDGAIN_REG+cnt), w_data );
}

```


AND9594/D

```
//The microcontroller sets threshold level of Cin0-Cin5 to 10.
w_data = 0x0A;
for( cnt=0; cnt<=5; cnt++ )
{
    I2c1byteWrite( SLAVE_ADDRESS, (CIN0THRESHOLD_REG+cnt), w_data );
}

//The microcontroller sets the Average Count Register to 0x80.
// (Average count = 128)
I2c1byteWrite( SLAVE_ADDRESS, AVECNT_REG, 0x80 );

//The microcontroller sets the Filter Parameter Register to 0x0C.
I2c1byteWrite( SLAVE_ADDRESS, FILPRM_REG, 0x0C );

//The microcontroller sets the Measurement 1 Register to 0x00
// so that the LC717A10 may assert the INTOUT signal after every measurement.
I2c1byteWrite( SLAVE_ADDRESS, MEASMODE1_REG, 0x00 );

//The microcontroller sets the Cin 1st Gain Adjust Register to 0x00.
// (1st stage AMP gain at all channels = 1600[ff]).
I2c1byteWrite( SLAVE_ADDRESS, CIN1STGAINADJ_REG, 0x00 );

//-----
//-- Note: The microcontroller usually sets --
//-- other registers in the same way as described above. --
//-----

// If necessary, after doing the parameter-setting processing,
// please verify all the register settings that you have set by reading back them.

/*****
/* The microcontroller issues a request to the LC717A10 for setting of parameters, */
/* static-offset-calibration, measurement and selection of "sleep mode". */
/*****
w_data = (WRITE_REQ_BIT | PARA_CH_BIT | STA_CAL_BIT | MEASURE_BIT);
I2c1byteWrite( SLAVE_ADDRESS, CONTROL1_REG, w_data );

/*****
/** The microcontroller wakes up the LC717A10. */
/*****
I2c1byteWrite( SLAVE_ADDRESS, CONTROL2_REG, WAKEUP_BIT );

//The microcontroller stores a dummy initial value to the variable sta_time_msec.
sta_time_msec = GetNowTime_msec();

/*****
/** The microcontroller wakes up the LC717A10 at about 50-millisecond intervals. */
/**
/** In the code listed below, the microcontroller checks the status of the */
/** INTOUT signal in order to check that the LC717A10 has gone to sleep. */
/** After the microcontroller has detected the assertion of the INTOUT signal, */
/** the microcontroller reads measurement results over the I2C bus. */
/** (Note: measurement results include the judgment result of ON/OFF status, */
/** error status and so on.) */
/*****
for(;;)
{
    //The microcontroller waits until the INTOUT signal is "High".
    //(In other words, the microcontroller waits until the measurement has completed
    // in the LC717A10 and LC717A10 has gone to sleep.)
    Wait_until_INTOUT_is_High();

    //The microcontroller writes 0x00 to the Control 2 Register.
    I2c1byteWrite( SLAVE_ADDRESS, CONTROL2_REG, 0x00 );

    /*****
    /** The microcontroller reads measurement results. */
    /** (At this time, the LC717A10 is in "sleep" state.) */
    /*****
    //If necessary, the microcontroller reads the CinX Data Register.

    //The microcontroller reads the Result Data 1 Register.
    rslt_data = I2c1byteRead( SLAVE_ADDRESS, RSLTDATA1_REG );

    //The microcontroller reads the Error Status Register.
```

AND9594/D

```
err_sts = I2c1byteRead( SLAVE_ADDRESS, ERRSTS_REG );

//The microcontroller reads the Error Channel Status 1 Register.
errch_sts = I2c1byteRead( SLAVE_ADDRESS, ERRCHSTS1_REG );

//The code for processing the measurement results should be described here.
//      :
//      :
//      :
//      :
//      :

/*****/
/** The microcontroller waits until a predetermined period of time      **/
/** (for example, 50 milliseconds) has passed since the microcontroller **/
/** started to get the measurement results.                             **/
/*****/
for(;;)
{
    //The microcontroller gets the time at this point.(in milliseconds)
    now_time_msec = GetNowTime_msec();
    if( (now_time_msec - sta_time_msec)>=50 )
    {
        break;
    }
}
sta_time_msec = now_time_msec;

/*****/
/** The microcontroller negates the INTOUT      **/
/** and wakes up the LC717A10.                **/
/*****/
I2c1byteWrite( SLAVE_ADDRESS, CONTROL2_REG, WAKEUP_BIT );
}
}
```

*: I²C Bus is a Trademark of Philips

ON Semiconductor and the ON Semiconductor logo are trademarks of Semiconductor Components Industries, LLC dba ON Semiconductor or its subsidiaries in the United States and/or other countries. ON Semiconductor owns the rights to a number of patents, trademarks, copyrights, trade secrets, and other intellectual property. A listing of ON Semiconductor's product/patent coverage may be accessed at www.onsemi.com/site/pdf/Patent-Marking.pdf. ON Semiconductor reserves the right to make changes without further notice to any products herein. ON Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does ON Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation special, consequential or incidental damages. Buyer is responsible for its products and applications using ON Semiconductor products, including compliance with all laws, regulations and safety requirements or standards, regardless of any support or applications information provided by ON Semiconductor. "Typical" parameters which may be provided in ON Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. ON Semiconductor does not convey any license under its patent rights nor the rights of others. ON Semiconductor products are not designed, intended, or authorized for use as a critical component in life support systems or any FDA Class 3 medical devices or medical devices with a same or similar classification in a foreign jurisdiction or any devices intended for implantation in the human body. Should Buyer purchase or use ON Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold ON Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that ON Semiconductor was negligent regarding the design or manufacture of the part. ON Semiconductor is an Equal Opportunity/Affirmative Action Employer. This literature is subject to all applicable copyright laws and is not for resale in any manner.