SPD2010 Protocol Guide V1.05

SOLOMON SYSTECH

SW

2021-04-02

YK.Kim

Content

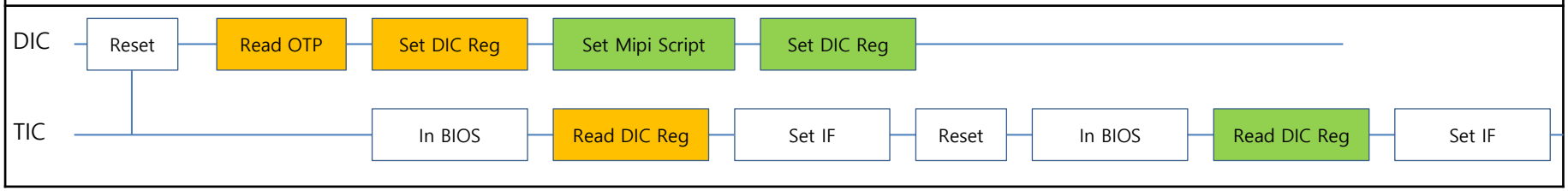
- ❖ Available Interface Setting
- ❖ I2C Protocol
  - working sequence & packet flow
  - Download fw to external flash with I2C
  - FW File Parsing
- ❖ SPI Protocol
  - working sequence & packet flow
  - Download fw to SRAM Protocol with SPI
  - FW File Parsing
- ❖ Read Data Description
- ❖ MP Test Protocol

Available Interface Setting

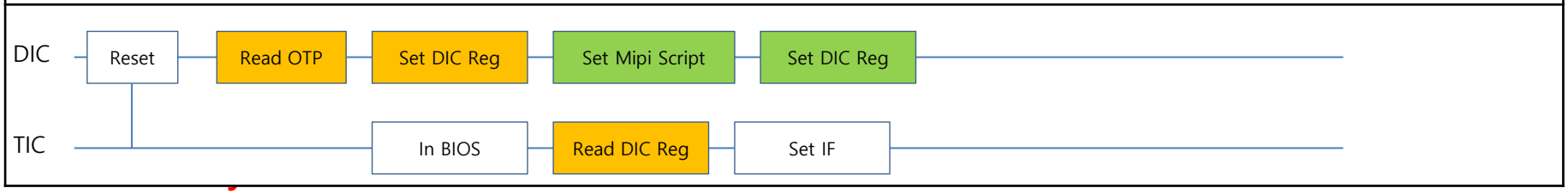
D-RST / T-RST	OTP Setting		Mipi Script Setting	
	AutoRun	IF	AutoRun	IF
Separated	don't care	don't care	Enable	I2C
	don't care	don't care	Disable	I2C/SPI
	Enable	I2C	x	x
	Disable	I2C/SPI	x	x
Tied (Not Use TRST)	Enable	I2C	Enable	I2C
	Disable	I2C	Disable	I2C
	Disable	SPI	Disable	SPI
	Enable	I2C	x	x
	Disable	I2C/SPI	x	x

Host Interface	
AutoRun	IF
Enable	I2C
Disable	I2C/SPI
Enable	I2C
Disable	I2C/SPI
Enable	I2C
Disable	I2C
Disable	SPI
Enable	I2C
Disable	I2C/SPI

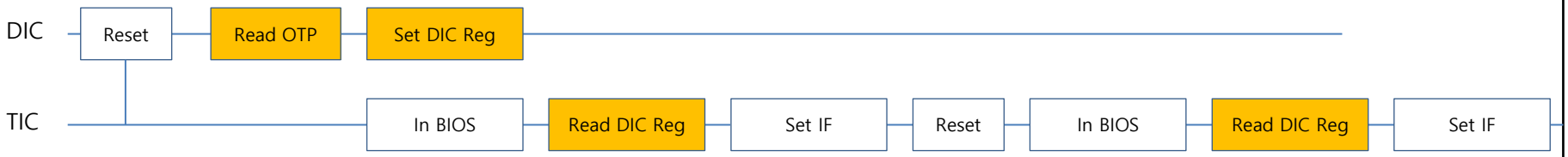
Separated D-SRT and T-RST



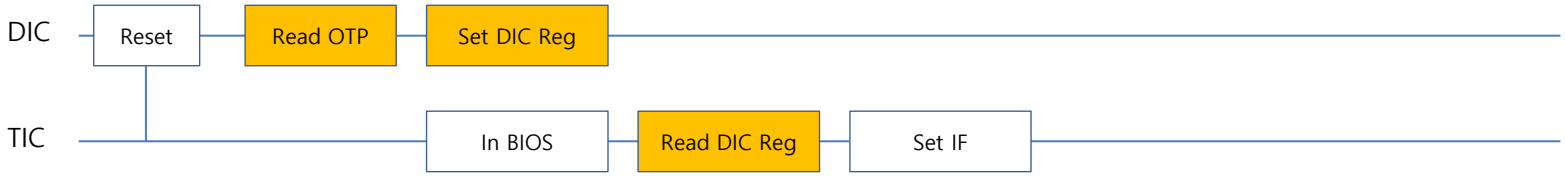
Tied D-SRT and T-RST (or not use T-RST)



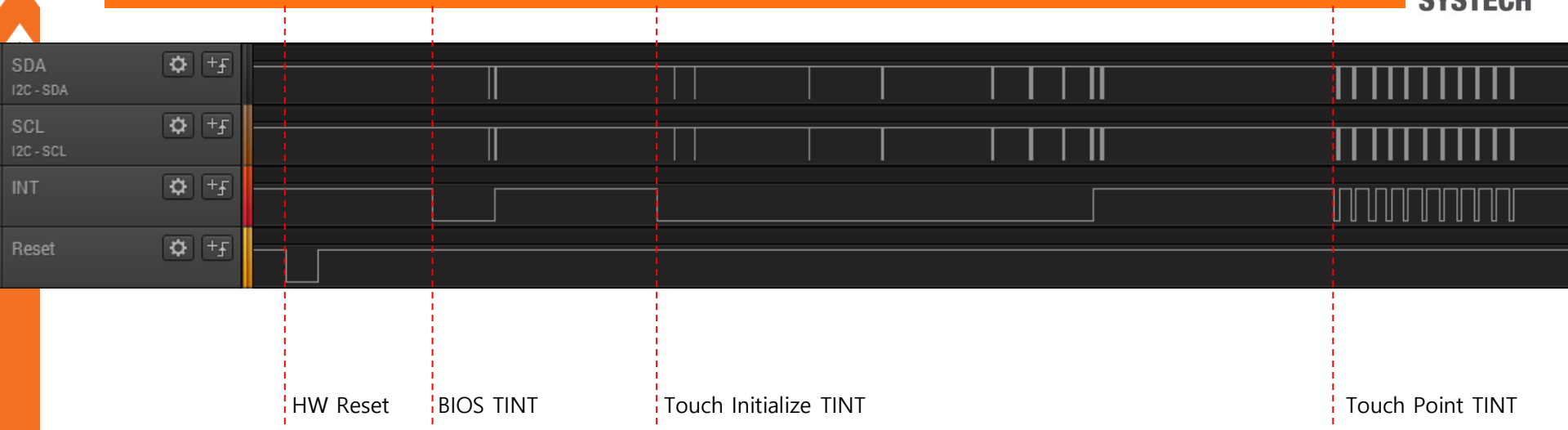
Separated D-SRT and T-RST, not set mipi scritp



Tied D-SRT and T-RST (or not use T-RST) , not set mipi scritp



Start Up Signal Flow



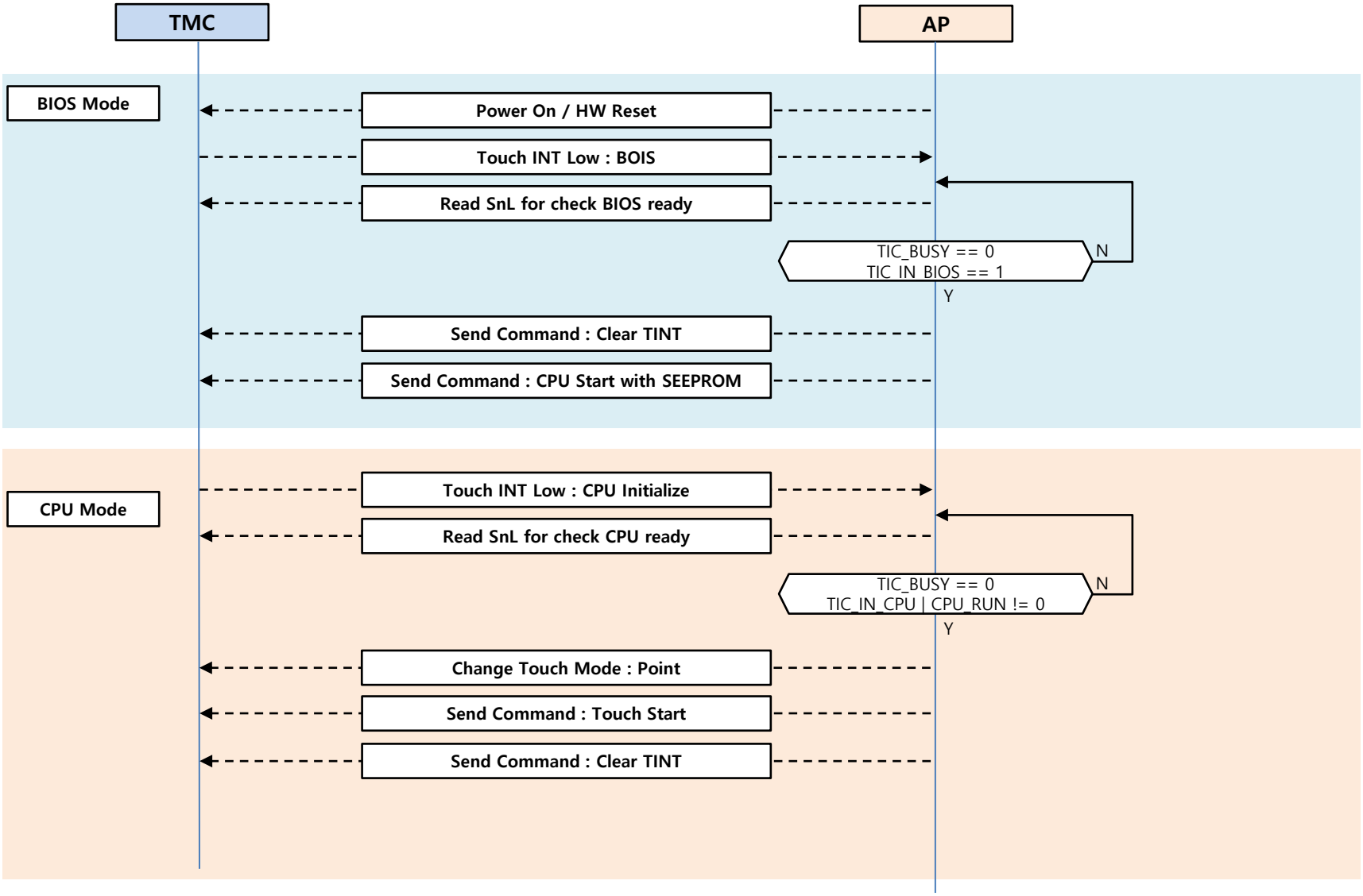
I2C Protocol

Solomon Systech confidential

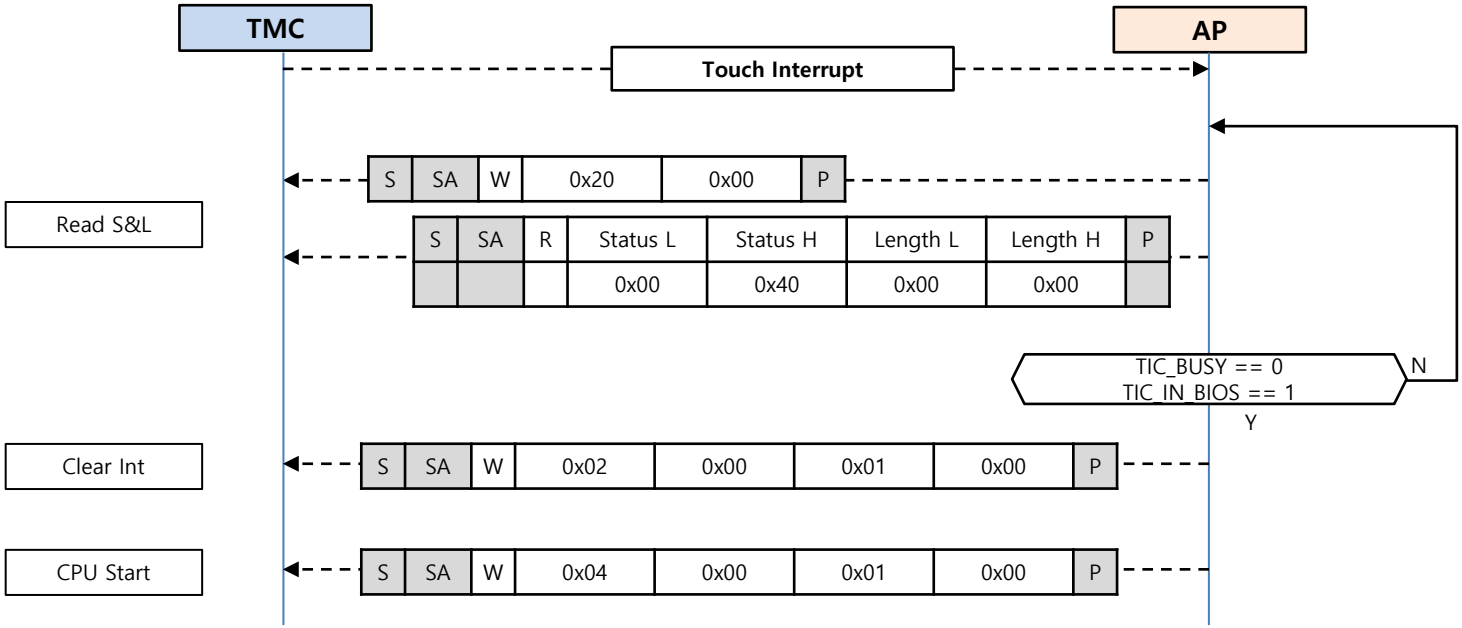
Compare protocol I2C and SPI

Interface	Protocol
I2C Read	S | 0x54 | W | Add L | Add H | P S | SA | W | rd 0 | rd 1 | … | rd n-1 | P
I2C Write	S | SA | W | Add L | Add H wd0 | wd1 | … | wd n-1 | P
SPI Read	CSN_L | 0x07 | Add L | Add H | Dmy rd 0 | rd 1 | … | rd n-1 | CSN_H
SPI Write	CSN_L | 0x06 | Add L | Add H wd0 | wd1 | … | wd n-1 | CSN_H

TIC Start Up flow with SEEPROM(Host IF : I2C)



example) BIOS Mode Start Up

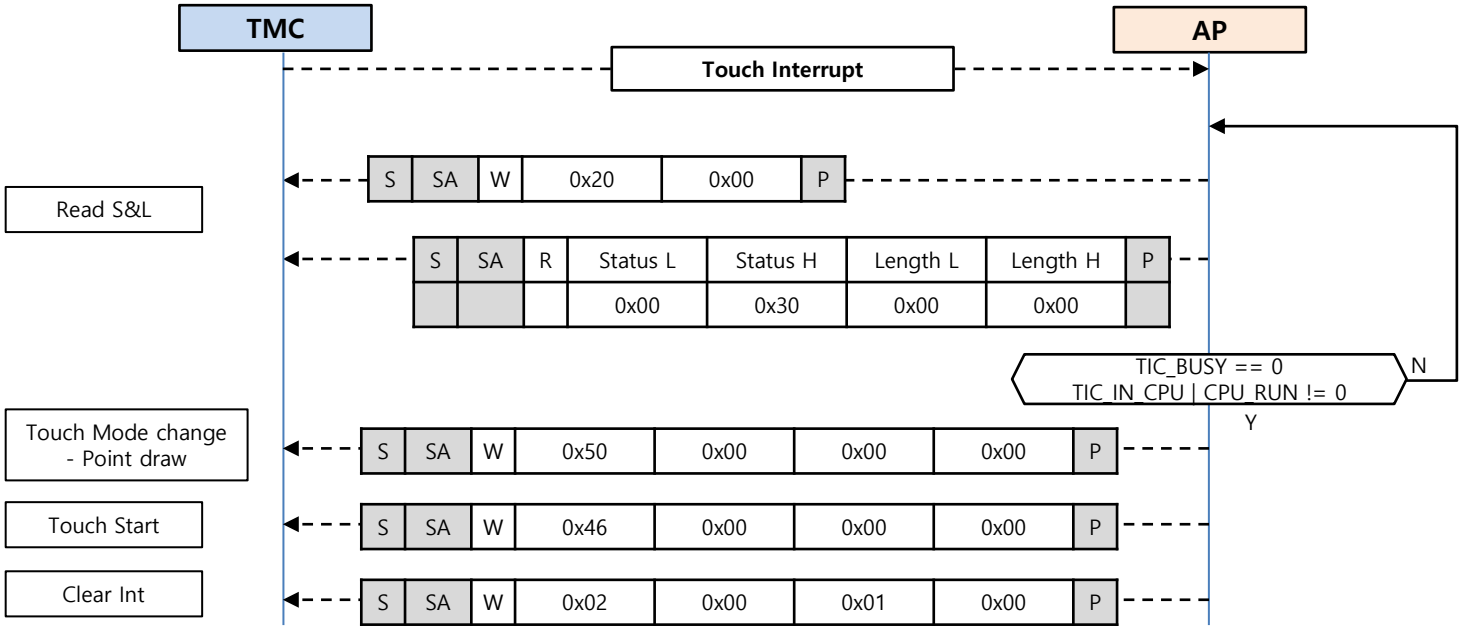


Name	Address	7bit	6bit	5bit	4bit	3bit	2bit	1bit	0bit
STATUS	0x0020	-	-	-	-	-	-	-	-
	0x0021	TIC_BUSY	TIC_IN_BIOS	CPU Init INT	TINT Low	CPU Run	-	-	-
Length	0x0022	Read Length Low byte							
	0x0023	Read Length Hi byte							

expect Status	Description
0xC000	BIOS Mode, Not Ready
0x4000	BIOS Mode, Ready
0xA000	CPU Mode, Not Ready
0x3000	CPU Mode, Touch Initialize Interrupt, Ready
0x08XX	CPU Mode, Touch Running, Ready

If set TIC_BUSY, retry read SnL until clear TIC_BUSY.

example) CPU Mode Start Up

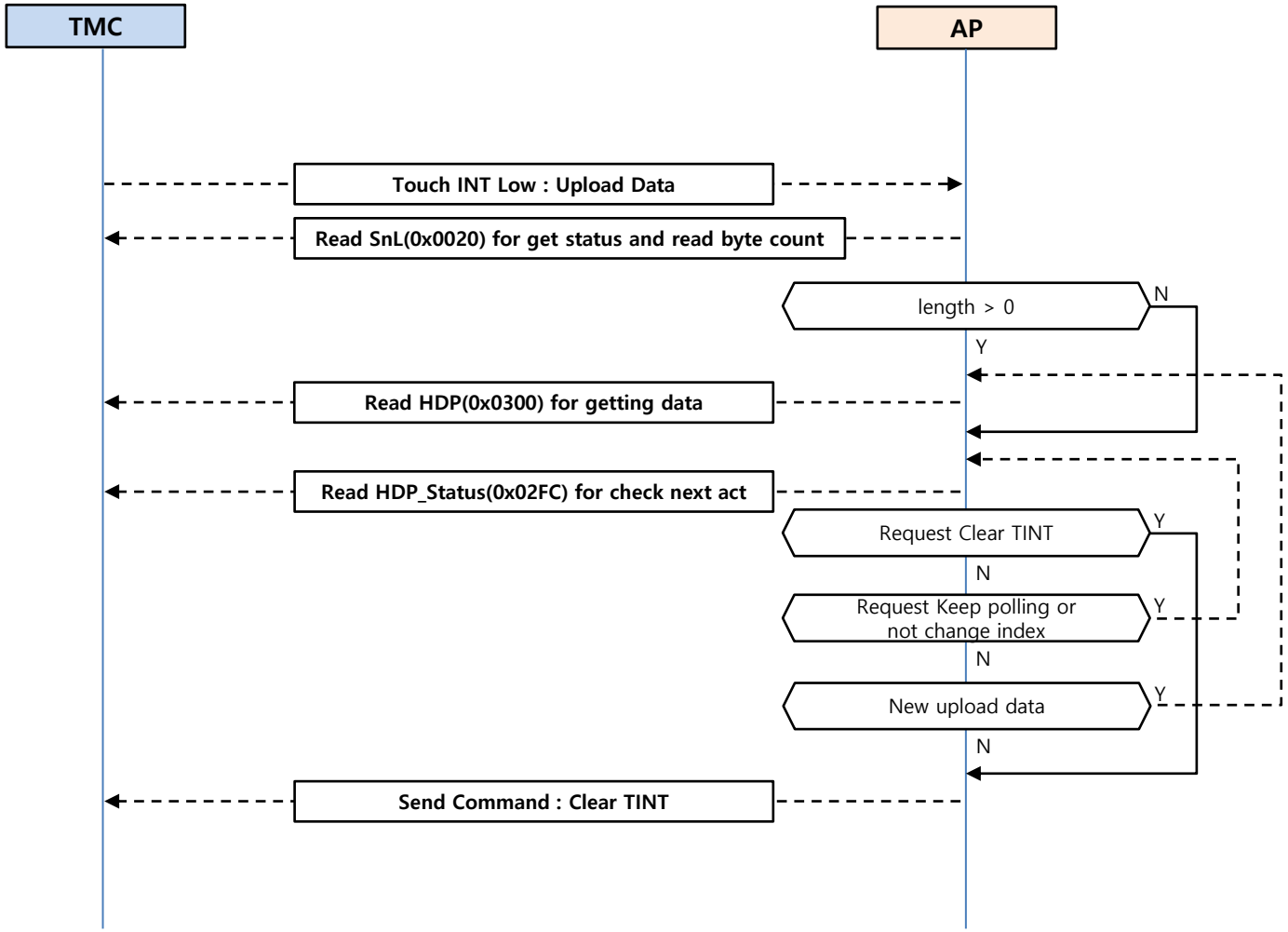


Name	Address	7bit	6bit	5bit	4bit	3bit	2bit	1bit	0bit
STATUS	0x0020	-	-	-	-	-	-	-	-
	0x0021	TIC_BUSY	TIC_IN_BIOS	TIC_IN_CPU	TINT L	CPU_RUN	-	-	-
Length	0x0022	Read Length Low byte							
	0x0023	Read Length Hi byte							

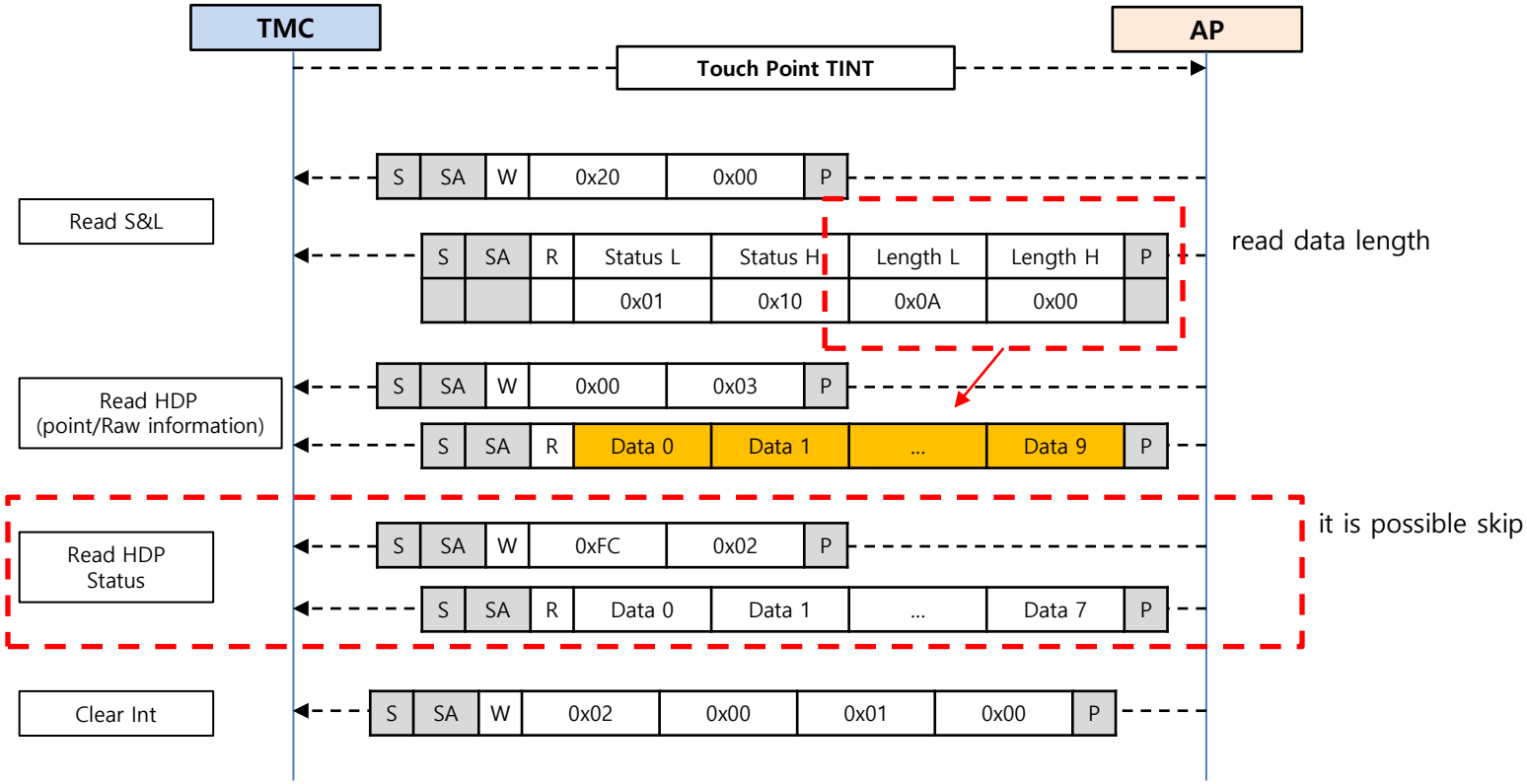
expect Status	Description
0xC000	BIOS Mode, Not Ready
0x4000	BIOS Mode, Ready
0xA000	CPU Mode, Not Ready
0x3000	CPU Mode, Touch Initialize Interrupt, Ready
0x08XX	CPU Mode, Touch Running, Ready

If set TIC_BUSY, retry read SnL until clear TIC_BUSY.

Point/Raw Data Read Flow



example) Point/Raw Data Read Flow



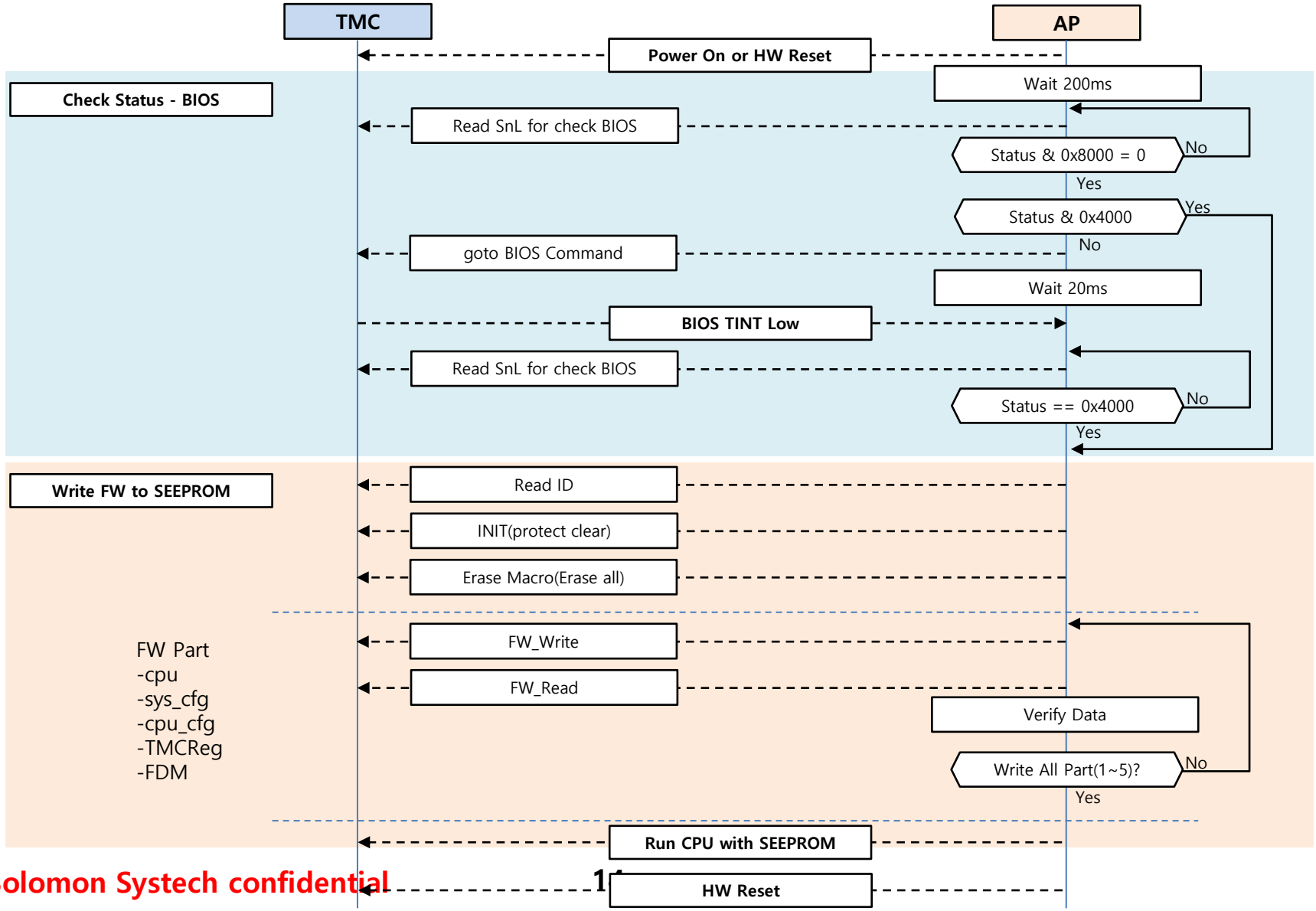
Name	Address	7bit	6bit	5bit	4bit	3bit	2bit	1bit	0bit
STATUS	0x0020	-	-	Raw(1)/PT(0)	Keep	Aux	Key	Gesture	PT EXIST
	0x0021	TIC_BUSY	TIC_IN_BIOS	TIC_IN_CPU	TINT Low	-	-	-	-
Length	0x0022	Read Length Low byte							
	0x0023	Read Length Hi byte							

when receive Touch Interrupt, read SnL. SnL has TMC status and next packet information.

Download FW to External Flash with I2C

Solomon Systech confidential

FW Download Flow



FWDL : Host side, Software-level function

SPI Flash Command		I2C Packet

INIT (protect clear)	**SE_GAMMA_SL_INIT ( ) ;**	void SE_GAMMA_SL_INIT ( ) {
		SE_GAMMA_LL_WEN ( );
		SE_GAMMA_LL_WRSR (0x0002);
		SE_GAMMA_LL_WAIT_RDY_SR ( ) ;
		}

RDID	**SE_GAMMA_SL_RDID (uint32_t * id32) ;**	void SE_GAMMA_SL_RDID (uint32_t * id32) {
		SE_GAMMA_LL_RDID (id32);
		}

Erase MACRO	**SE_GAMMA_SL_ERASE_MACRO ( ) ;**	void SE_GAMMA_SL_ERASE_MACRO ( ) {
		SE_GAMMA_LL_WEN ( ) ;
		SE_GAMMA_LL_ERASE_MACRO ( ) ;
		SE_GAMMA_LL_WAIT_RDY_SR ( ) ;
		}

Erase SECTOR	**SE_GAMMA_SL_ERASE_SECTOR (uint32_t add) ;**	void SE_GAMMA_SL_ERASE_SECTOR (uint32_t add) {
		SE_GAMMA_LL_WEN ( ) ;
		SE_GAMMA_LL_ERASE_SECTOR (add) ;
		SE_GAMMA_LL_WAIT_RDY_SR ( ) ;
		}

PROGRAM_PAGE	**SE_GAMMA_SL_PROGRAM_PAGE (uint32_t add, uint32_t nb, uint8_t * wm8) ;**	void SE_GAMMA_SL_PROGRAM_PAGE (uint32_t add, uint32_t nb, uint8_t * wm8) {
		SE_GAMMA_LL_WEN ( ) ;
		SE_GAMMA_LL_PROGRAM_PAGE (add, nb, wm8) ;
		SE_GAMMA_LL_WAIT_RDY_SR ( ) ;
		}

READ_DATA	**SE_GAMMA_SL_READ_DATA (uint32_t add, uint32_t nb, uint8_t * rm8) ;**	void SE_GAMMA_SL_READ_DATA (uint32_t add, uint32_t nb, uint8_t * rm8) {
		SE_GAMMA_LL_READ_DATA (add, nb, rm8) ;
		}

FWDL : Host side, low-level function

SOLOMON SYSTECH

- LL (low-level) function
  - basic functions to access SPI flash.
  - Functions cover SPI flash's various commands.
- note
  - busy check only menas "SPI flash command is sent"
  - Actual "Program Page Done" should be checked with SE_GAMMA_LL_RDSR bit[0]=0

SPI Flash Command		I2C Packet
		In reference source code, each function uses the function **TIC_HQARG_WHDP16_WHDP32_WHDP_HCMD_CHKBUSY_RHDP16_RHDP32_RHDP**
Wait Ready, with HCMD Clear	**SE_GAMMA_LL_WAIT_HCMD_CLR (uint32_t add) ;**	(Read HCMD) S SA/W add_Byte0 add_Byte1 P S SA/R hcmd_rd_Byte0 hcmd_rd_Byte0 P * Repeat "Read HCMD" until hcmd_rd becomes 0x0000
Read ID	**SE_GAMMA_LL_RDID (uint32_t * id32) ;**	(Write HCMD) S SA/W **08 00 00 02** P (busy check) SE_GAMMA_LL_WAIT_HCMD_CLR (0x0008) (Read HDP) S SA/W 40 00 P + S SA/R ID_Byte2 ID_Byte1 ID_Byte0 P
Write EN	**SE_GAMMA_LL_WEN ( ) ;**	(Write HCMD) S SA/W **08 00 30 02** P (busy check) SE_GAMMA_LL_WAIT_HCMD_CLR (0x0008)
Write Status Register	**SE_GAMMA_LL_WRSR (uint16_t wd16) ;**	(Write HQARG) S SA/W 10 00 wd16_Byte0 wd16_Byte1 0x00 0x00 P (Write HCMD) S SA/W **08 00 20 02** P (busy check) SE_GAMMA_LL_WAIT_HCMD_CLR (0x0008)
Read Status Register	**SE_GAMMA_LL_RDSR (uint16_t* rd16) ;**	(Write HCMD) S SA/W **08 00 10 02** P (busy check) SE_GAMMA_LL_WAIT_HCMD_CLR (0x0008) (Read HDP) S SA/W 40 00 P + S SA/R rd_Byte0 rd_Byte1 P

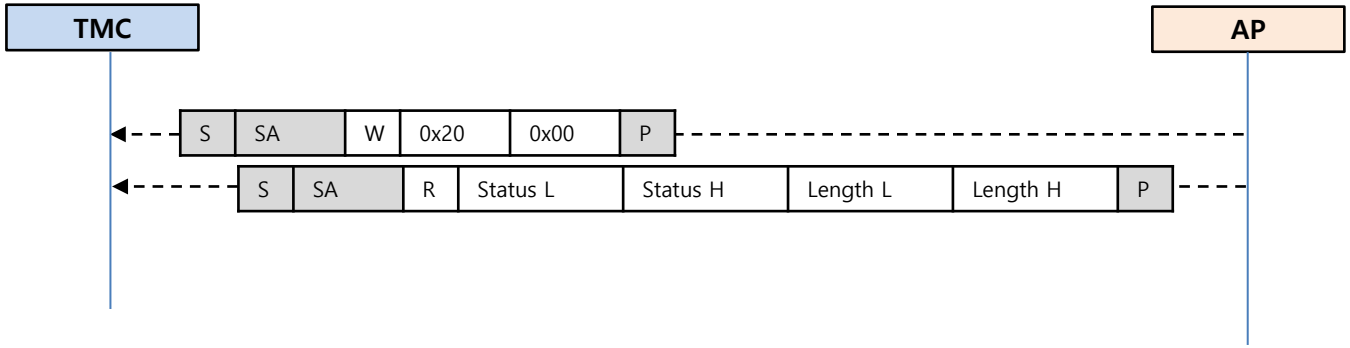
Wait Ready, with Status Register	**SE_GAMMA_LL_WAIT_RDY_SR ( ) ;**	uint32_t SE_GAMMA_LL_WAIT_RDY_SR ( ) {
		uint16_t sr;
		while (1) {
		SE_GAMMA_LL_RDSR (& sr);
		if ((sr&0x01)==0x00) {
		break;
		}
		}
		return 0;

FWDL : Host side, low-level function

SPI Flash Command		I2C Packet
		In reference source code, each function uses the function **TIC_HQARG_WHDP16_WHDP32_WHDP_HCMD_CHKBUSY_RHDP16_RHDP32_RHDP**
Read Data	**SE_GAMMA_LL_READ_DATA (uint32_t add, uint32_t nb, uint8_t * rm8) ;**	(Write HQARG) S SA/W 10 00 add_Byte0 add_Byte1 add_Byte2 add_Byte3 nbyte_Byte0 nbyte_Byte1 nbyte_Byte2 nbyte_Byte3 P (Write HCMD) S SA/W **08 00** **30 03** P (busy check) SE_GAMMA_LL_WAIT_HCMD_CLR (0x0008) (Read HDP) S SA/W 40 00 P + S SA/R D[0] D[1] D[2] D[--] P * note : IC reads SPI flash with DUAL_READ
Program Page	**SE_GAMMA_LL_PROGRAM_PAGE (uint32_t add, uint32_t nb, uint8_t * wm8) ;**	(Write HQARG) S SA/W 10 00 add_Byte0 add_Byte1 add_Byte2 add_Byte3 nbyte_Byte0 nbyte_Byte1 nbyte_Byte2 nbyte_Byte3 P (Write HDP) S SA/W 40 00 D[0] D[1] D[2] D[--] P (Write HCMD) S SA/W **08 00** **20 03** P (busy check) SE_GAMMA_LL_WAIT_HCMD_CLR (0x0008)
Erase Sector	**SE_GAMMA_LL_ERASE_SECTOR (uint32_t add) ;**	(Write HQARG) S SA/W 10 00 add_Byte0 add_Byte1 add_Byte2 add_Byte3 P (Write HCMD) S SA/W **08 00** **10 03** P (busy check) SE_GAMMA_LL_WAIT_HCMD_CLR (0x0008)
Erase Macro	**SE_GAMMA_LL_ERASE_MACRO ( ) ;**	(Write HCMD) S SA/W **08 00** **00 03** P (busy check) SE_GAMMA_LL_WAIT_HCMD_CLR (0x0008)

example) I2C Protocol

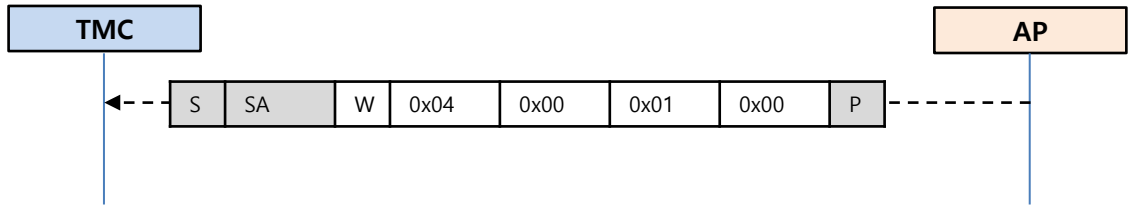
Read SnL



Name	Access Address	Address	7bit	6bit	5bit	4bit	3bit	2bit	1bit	0bit
TIC STATUS	0x0020 (Read Only)	0x0020	-							
		0x0021	TIC_BUSY	IN_BIOS						

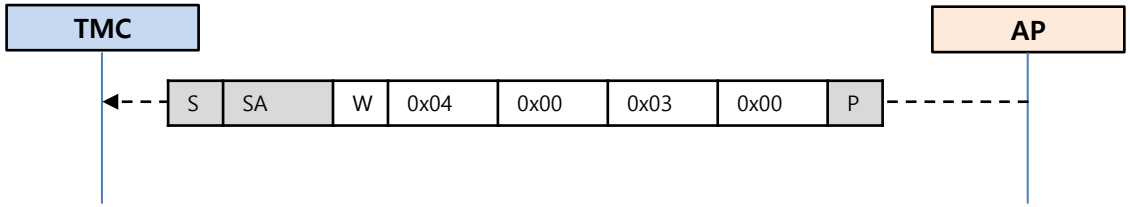
TIC Status : 0x4000 - in BIOS, not Busy
need to check 'IN BIOS' for download fw to SEEPROM.

Run CPU with SEEPROM



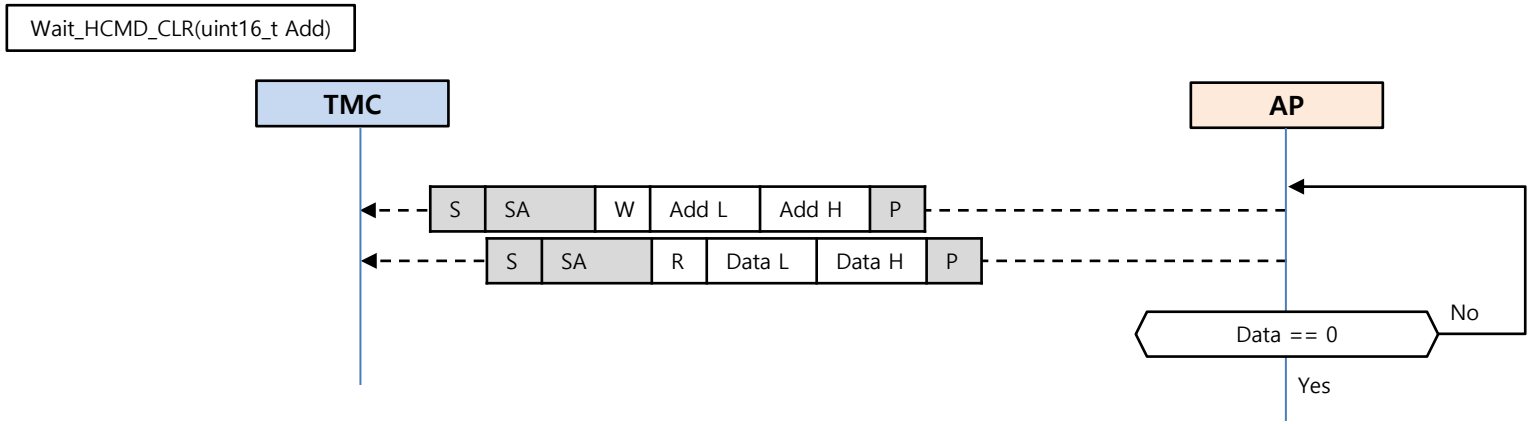
In BIOS mode, when receive command, loading FW from SEEPROM to SRAM and then jump to CPU mode

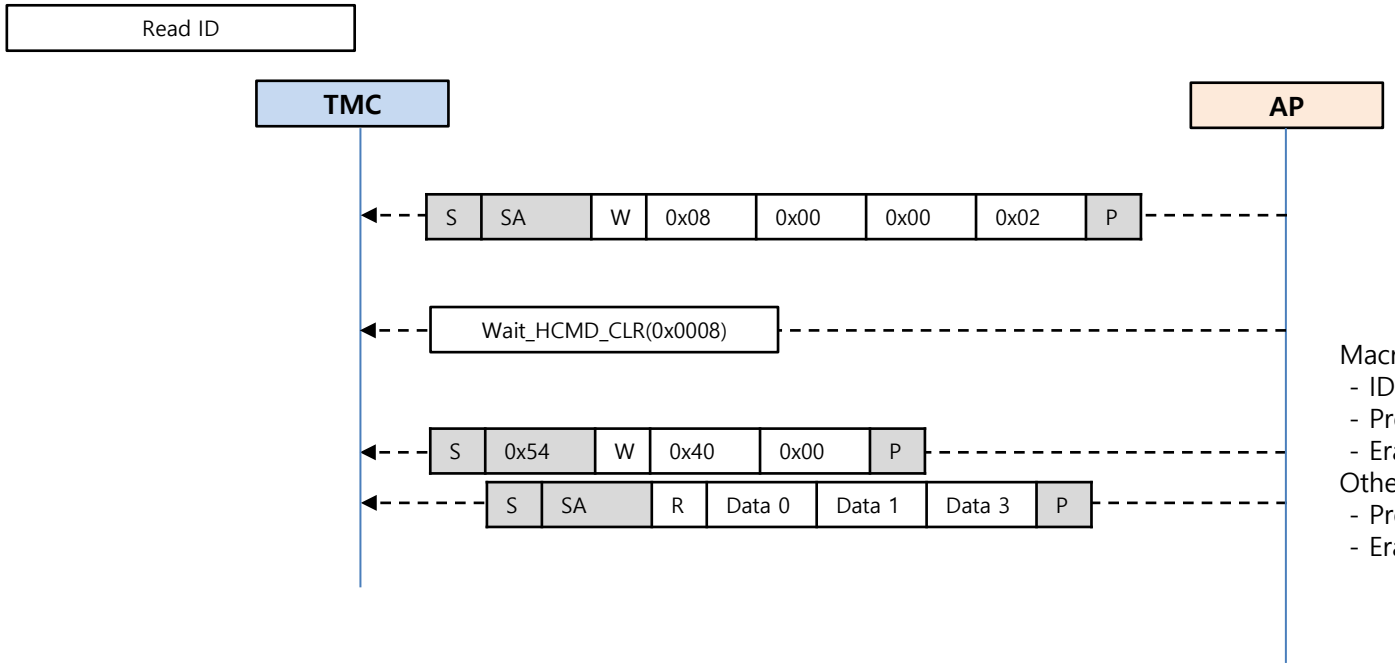
goto BIOS Command



In CPU mode, when receive 'goto BIOS' command, TMC jump to BIOS

example) I2C Protocol

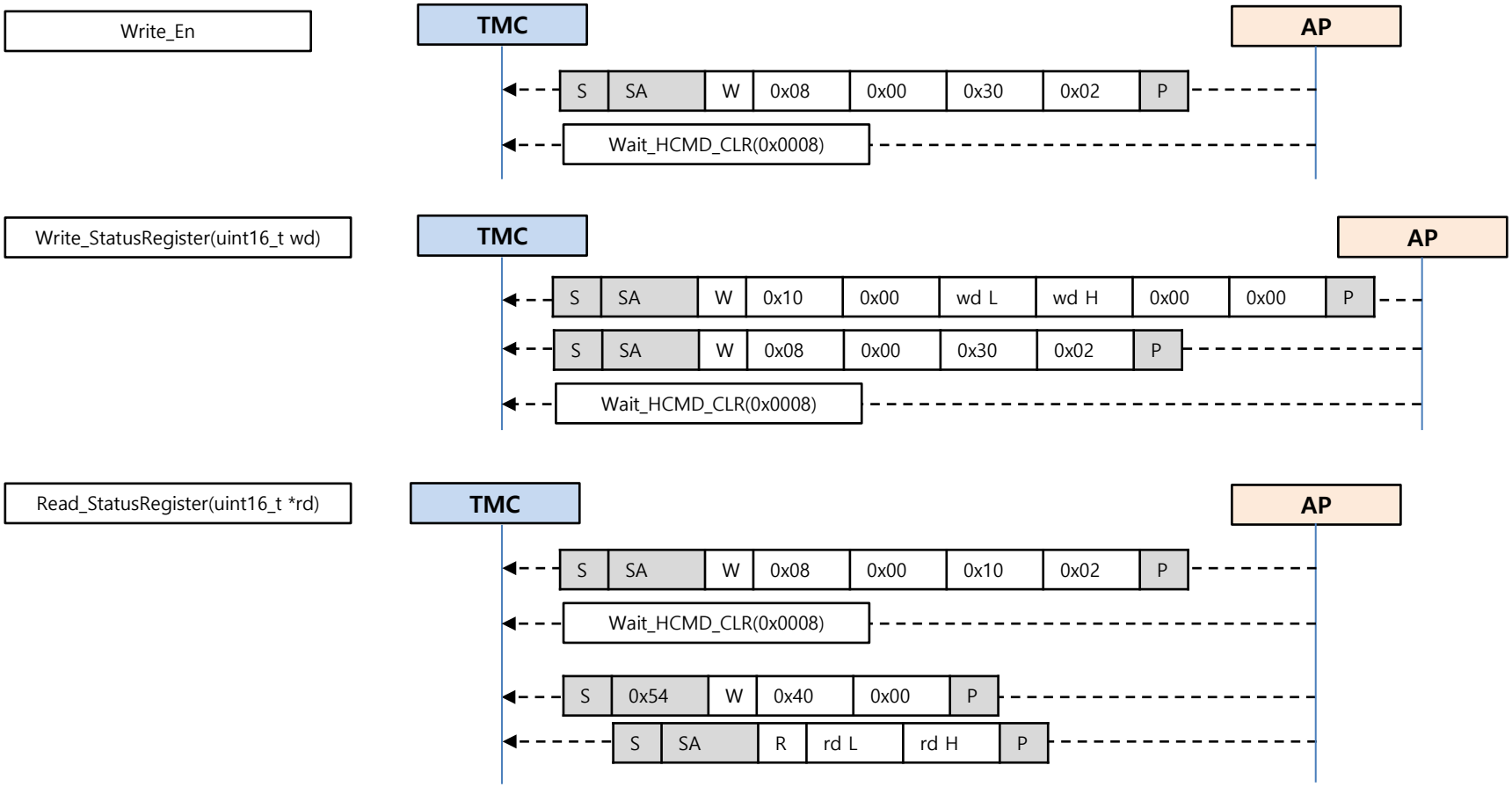




Macronix MX25U1001E
- ID : 0x3125C2
- Program page size : 32 Byte
- Erase Sector size : 4 K Byte

Other
- Program page size : 256 Byte
- Erase Sector size : 4 K Byte

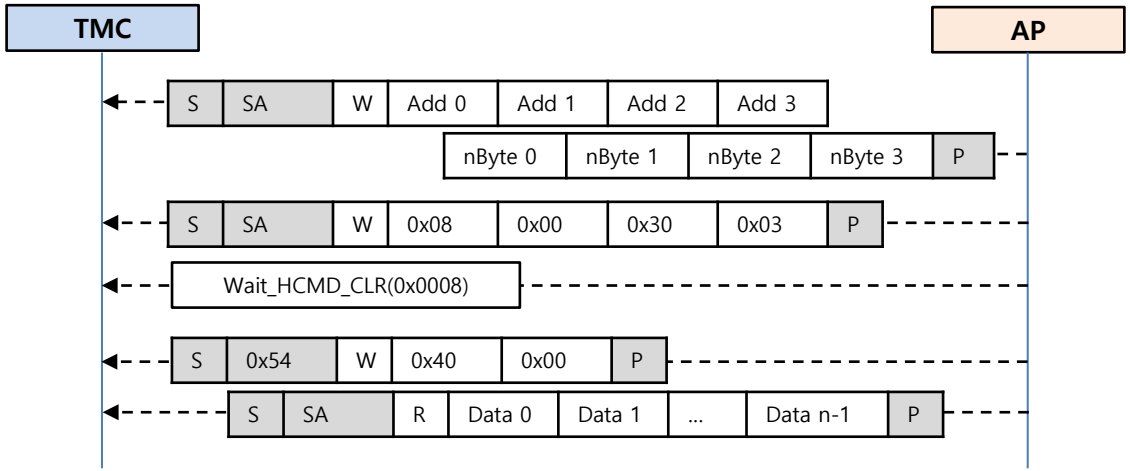
example) I2C Protocol



example) I2C Protocol

Read_Data(uint32_t Add, uint32 nByte, uint8_t* Data)

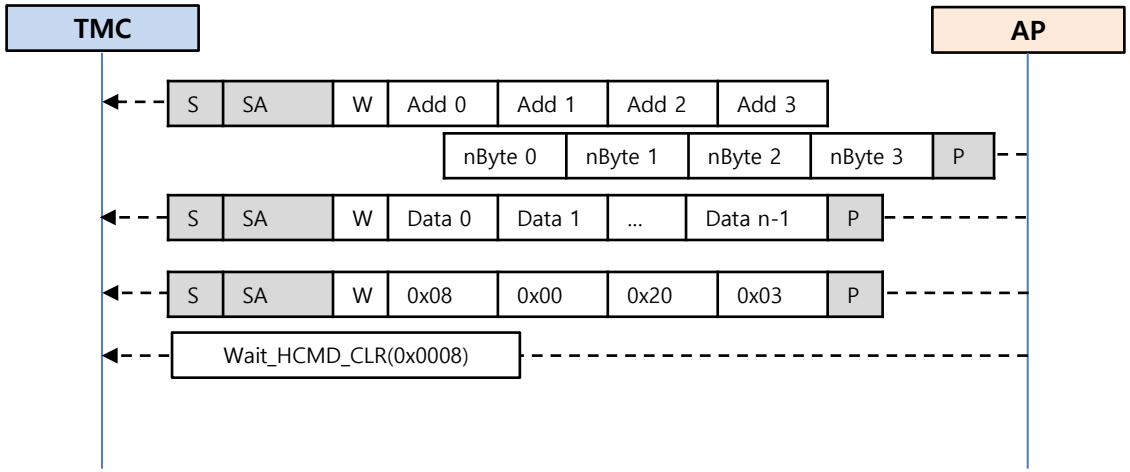
Read Size Max : 1024 – 64 byte



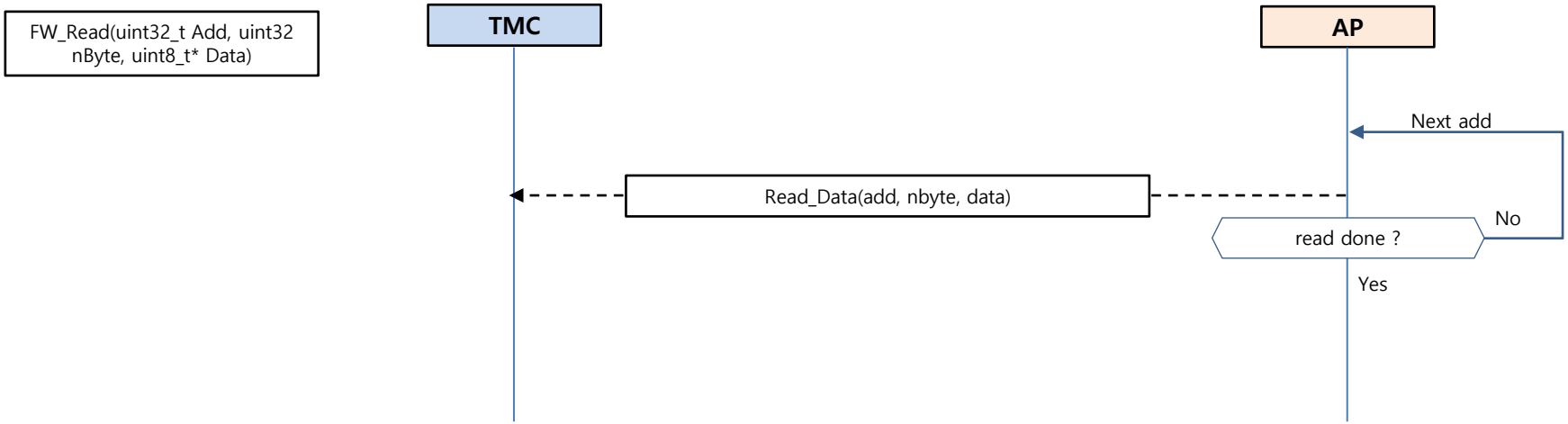
Program_Page(uint32_t Add, uint32 nByte, uint8_t* Data)

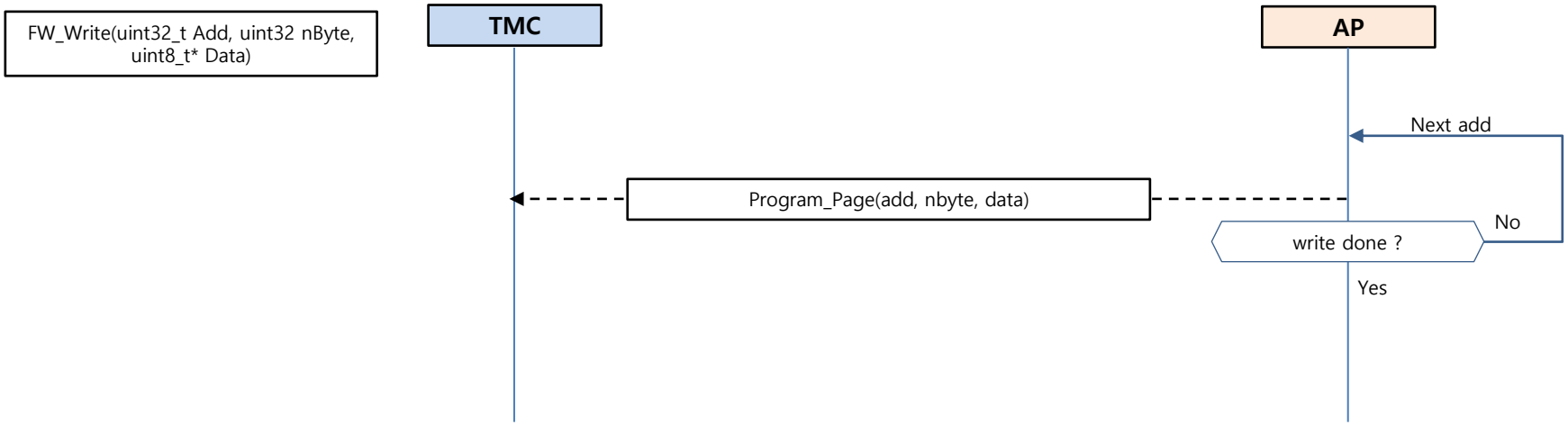
Macronix MX25U1001E
- ID : 0x3125C2
- Program page size : 32 Byte
- Erase Sector size : 4 K Byte

Other
- Program page size : 256 Byte
- Erase Sector size : 4 K Byte



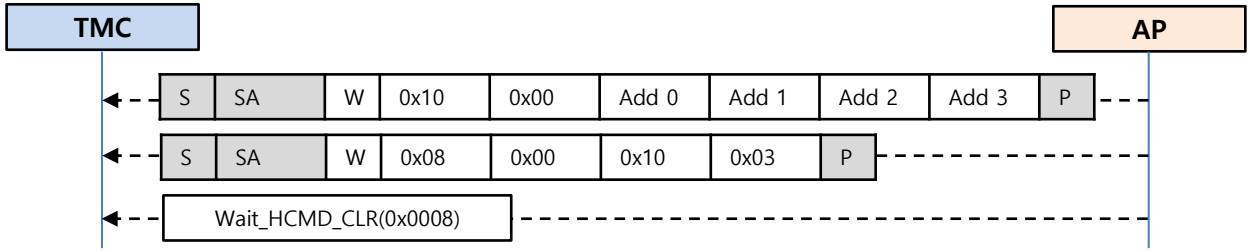
example) I2C Protocol



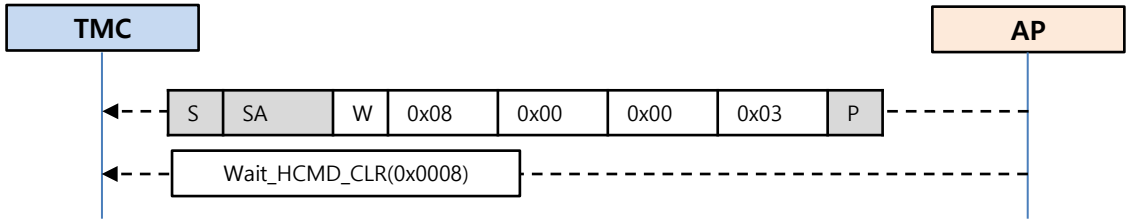


example) I2C Protocol

Erase_Sector(uint32_t Add)



Erase_Macro



NOTE

SOLOMON SYSTECH

❖ SPI Flash

- Macronix MX25U1001E
  - ID : 0x3125C2
  - Program page size : 32 Byte
  - Erase Sector size : 4 K Byte
- Other
  - Program page size : 256 Byte
  - Erase Sector size : 4 K Byte

❖ Read Data

- IC has 1KB DMAMEM (MB)
- The first 64B is being used for HCMD (host command)
- So, SPI Flash read size is max. 1024-64.
- For an alignment, read 512 Bytes from SPI flash data at a time.

Memory Map - Flash

- M_MAP_FW = 0x0000
- M_MAP_CPU_CFG = 0x1F400
- M_MAP_SYS_CFG = 0x1F000
- M_MAP_TMCREG = 0x1C000
- M_MAP_FDM = 0x1D000

FW File

mgd_*.hex / cpufw_*.hex / dset_*.hex

SOLOMON SYSTECH

- Format(WinTAG, MP Device, AnTAG)
  - $Display Version
  - $Hidden Version
  - $Product ID1(ASCII)
  - $Product ID2(ASCII)
  - $Chip Name1(ASCII)
  - $Chip Name2(ASCII)
  - $Reserved * 2ea
  - $Erase option & File Count
    - 0xAABBCCDD
    - 0xAABB : erase option, 1 all erase, 0 page erase
    - 0xCCDD : File count
  - $data start address offset(32bit, ASCII) * 9ea
  - #eFlash Address(32bit, ASCII)
  - *Byte Count(32bit, ASCII)
  - *Erase Page Count(32bit, ASCII)
  - *Version(32bit, ASCII)
  - *Check Sum(32bit, ASCII)
  - *Reserved_01(32bit, ASCII)
  - *Reserved_02(32bit, ASCII)
  - *Reserved_03(32bit, ASCII)
  - Content(32bit, bin)

```
$00000001
$00000000
$546f7669
$73353200
$53534400
$32303938
$ffffffff
$ffffffff
$00010005
$000000c6
$000001e0
$000002ba
$0000f4c4
$0000f782
$FFFFFFFF
$FFFFFFFF
$FFFFFFFF
$FFFFFFFF
#00000000
*000000c0
*00000001
*ffffffff
*fea8de00
*00000000
*00000000
*00000000
?]|┌◄‼
*00000080
*00000000
*ffffffff
*f646383e
*00000000
*00000000
*00000000
??┌   ┼  ┼ D?  D?   ┼ I
#00001000
*0000f1b0
*00000010
*ffffffff
```

mgd file detail

mgd_SPD2010_7x6_210218_test_1.hex

Header	Part 1	Part 2	Part 3	Part 4	Part 5
$FFFFFFFF $FFFFFFFF $FFFFFFFF $FFFFFFFF $FFFFFFFF $FFFFFFFF $FFFFFFFF $FFFFFFFF $00010005 $000000c6 $0000fd08 $0000ffc6 $0001082c $00010986 $FFFFFFFF $FFFFFFFF $FFFFFFFF $FFFFFFFF	#00000000 *0000fbe8 *00000010 *ffffffff *f75650bc *fffffcb5 *00000000 *00000000 `? <趙┐??<? ?	#0001c000 *00000264 *00000001 *00000001 *d65523d1 *ffffffff *ffffffff *ffffffff r X┐ ?U? ※	#0001d000 *0000080c *00000001 *00000003 *8bf8fde6 *ffffffff *ffffffff *ffffffff └ _ ◘ 艶?AB	#0001f000 *00000100 *00000001 *00000000 *bf724dee *ffffd4c1 *00000000 *00000000 璃滋 ?	#0001f400 *00000100 *00000000 *ffffffff *f1a947d7 *ffff6975 *00000000 *00000000 ? ¶? 졺J

#0001c000 --------→ #eFlash Address(32bit, ASCII)
*00000264 --------→ *Byte Count(32bit, ASCII)
*00000001 --------→ *Erase Page Count(32bit, ASCII)
*00000001 --------→ *Version(32bit, ASCII)
*d65523d1 --------→ *Check Sum(32bit, ASCII)
*ffffffff --------→ *Reserved_01(32bit, ASCII)
*ffffffff --------→ *Reserved_02(32bit, ASCII)
*ffffffff --------→ *Reserved_03(32bit, ASCII)
r X┐ ?U? ※ --→ Content(32bit, bin)

Program_Page(uint32_t Add, uint32 nByte, uint8_t* Data)

SPI Protocol

Solomon Systech confidential

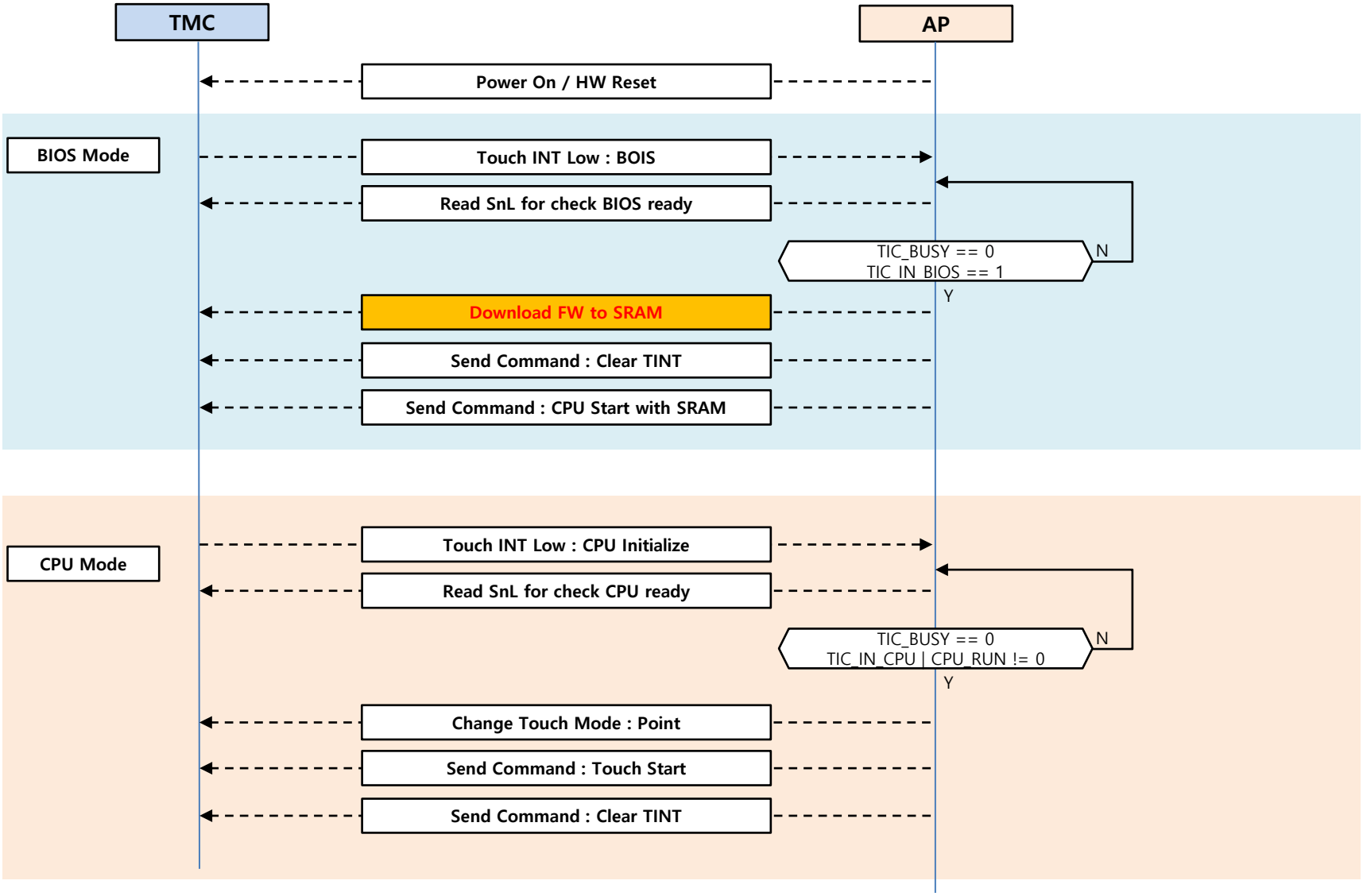
Compare protocol I2C and SPI

Interface	Protocol
I2C Read	S | 0x54 | W | Add L | Add H | P S | SA | W | rd 0 | rd 1 | … | rd n-1 | P
I2C Write	S | SA | W | Add L | Add H wd0 | wd1 | … | wd n-1 | P
SPI Read	CSN_L | 0x07 | Add L | Add H | Dmy rd 0 | rd 1 | … | rd n-1 | CSN_H
SPI Write	CSN_L | 0x06 | Add L | Add H wd0 | wd1 | … | wd n-1 | CSN_H

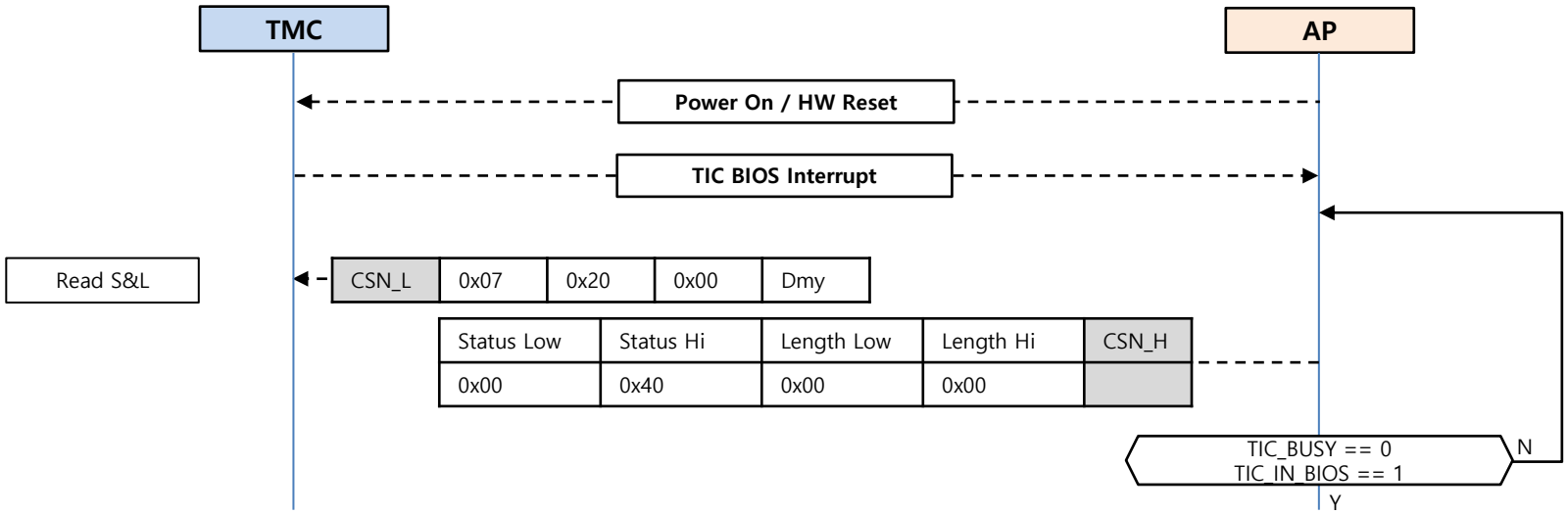
Comment

- SPI Interface
  - SPI Interface limitation
    - can not use SEEPROM
    - can not use 'Auto Run'
  - when start up(Power On or HW Reset), need to download fw to SRAM

TIC Start Up flow with SRAM(Host IF : SPI)



example) BIOS Mode Start Up flow

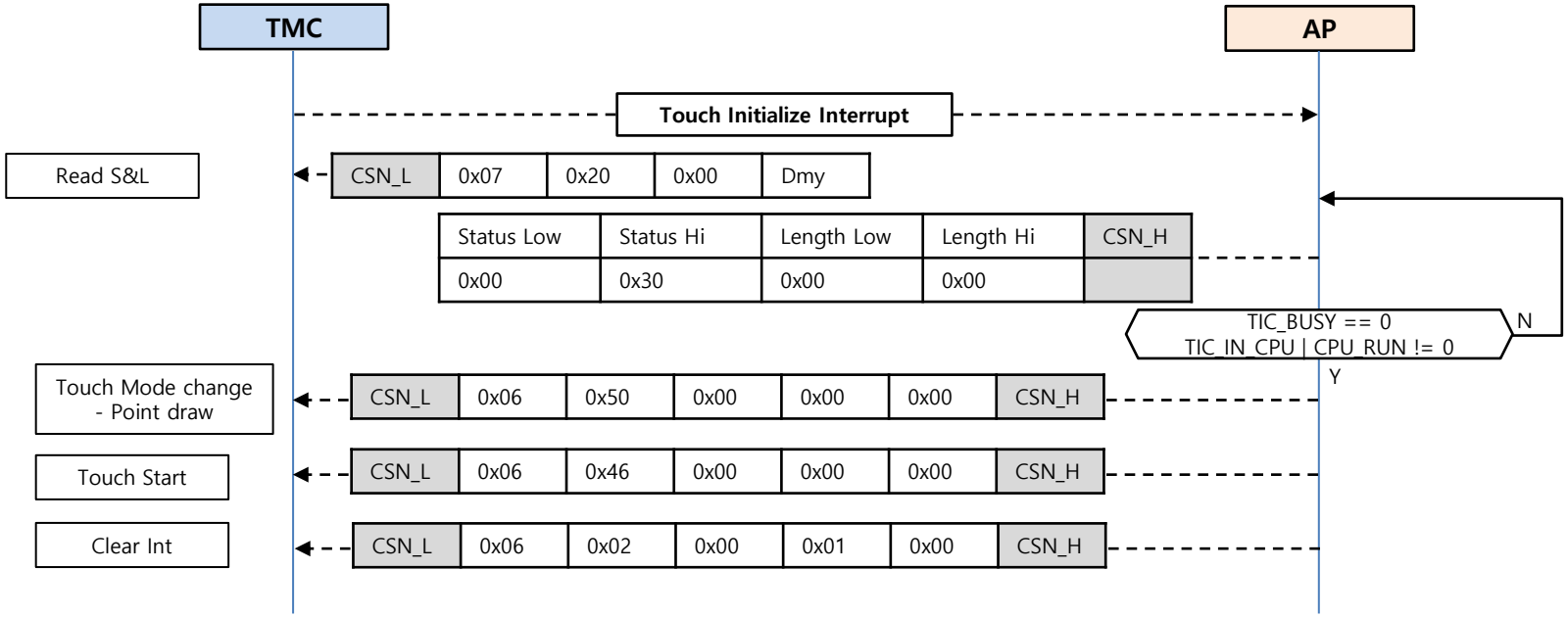


Name	Address	7bit	6bit	5bit	4bit	3bit	2bit	1bit	0bit
STATUS	0x0020	-	-	-	-	-	-	-	-
	0x0021	TIC_BUSY	TIC_IN_BIOS	TIC_IN_CPU	TINT L	CPU Run	-	-	-
Length	0x0022	Read Length Low byte							
	0x0023	Read Length Hi byte							

expect Status	Description
0xC000	BIOS Mode, Not Ready
0x4000	BIOS Mode, Ready
0xA000	CPU Mode, Not Ready
0x3000	CPU Mode, Touch Initialize Interrupt, Ready
0x08XX	CPU Mode, Touch Running, Ready

If set TIC_BUSY, retry read SnL until clear TIC_BUSY.

example) CPU Mode Start Up flow

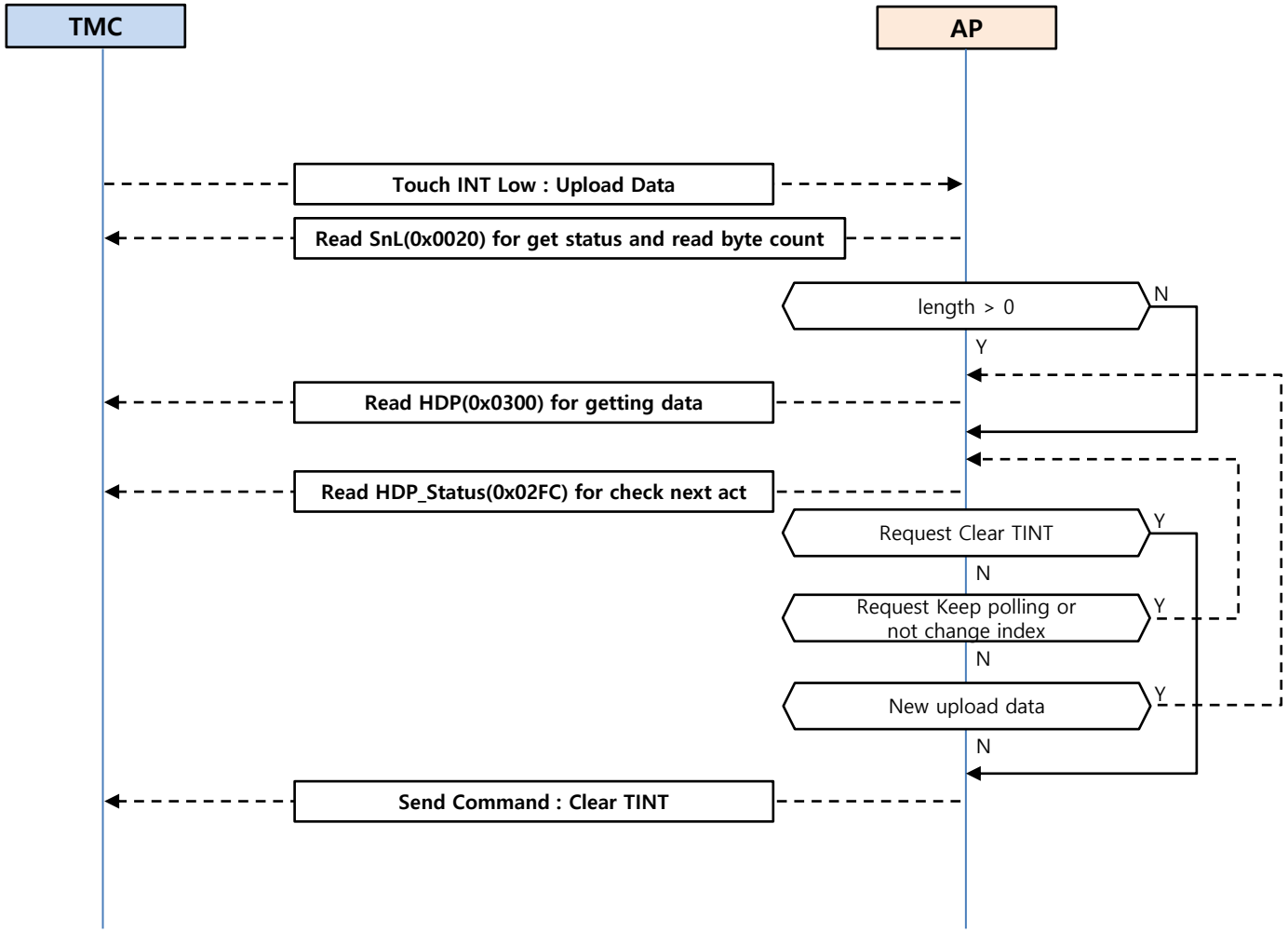


Name	Address	7bit	6bit	5bit	4bit	3bit	2bit	1bit	0bit
STATUS	0x0020	-	-	-	-	-	-	-	-
	0x0021	TIC_BUSY	TIC_IN_BIOS	TIC_IN_CPU	TINT L	CPU Run	-	-	-
Length	0x0022	Read Length Low byte							
	0x0023	Read Length Hi byte							

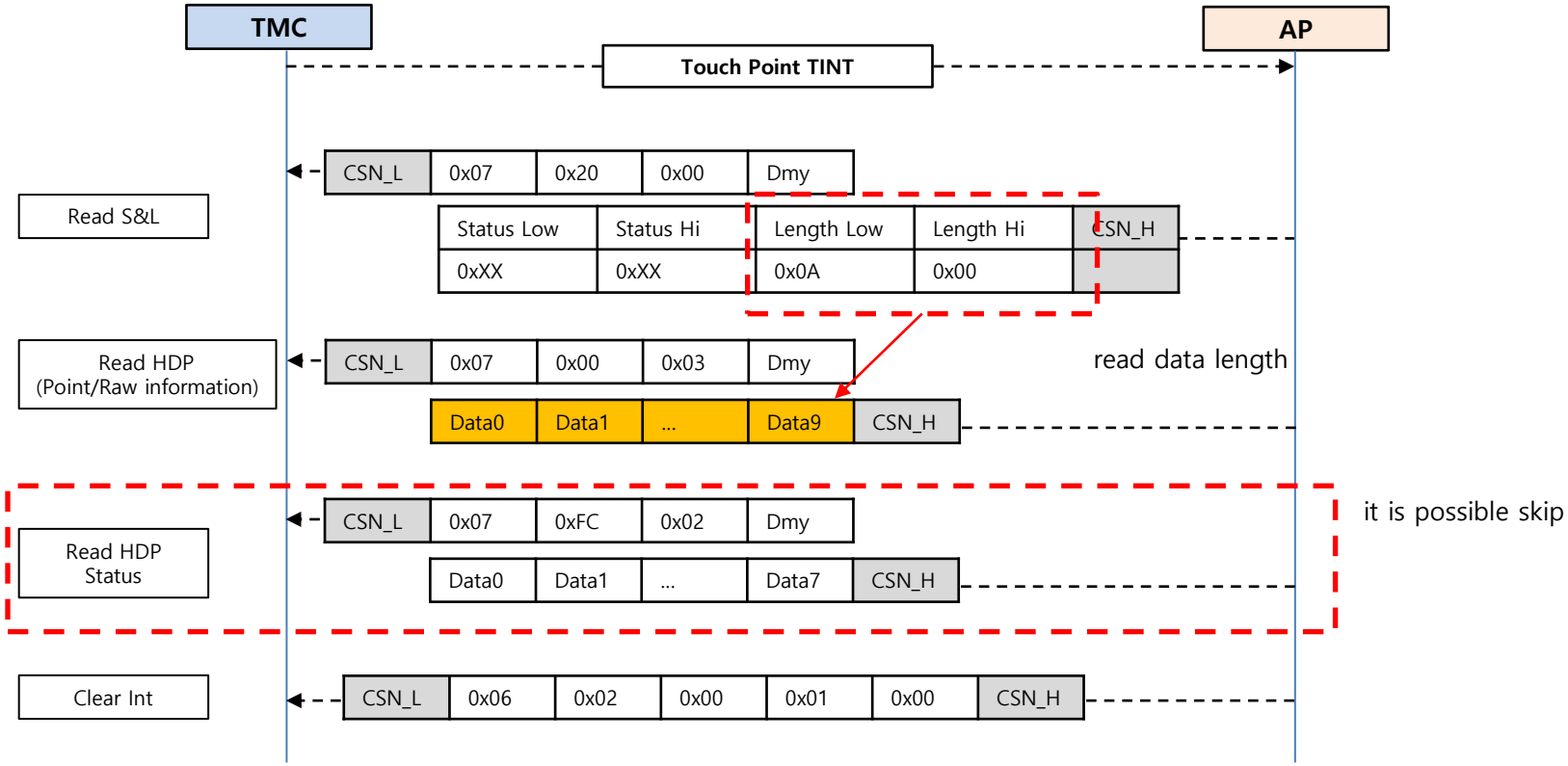
expect Status	Description
0xC000	BIOS Mode, Not Ready
0x4000	BIOS Mode, Ready
0xA000	CPU Mode, Not Ready
0x3000	CPU Mode, Touch Initialize Interrupt, Ready
0x08XX	CPU Mode, Touch Running, Ready

If set TIC_BUSY, retry read SnL until clear TIC_BUSY.

Point/Raw Data Read Flow



example) Point/Raw Data Read Flow - SPI



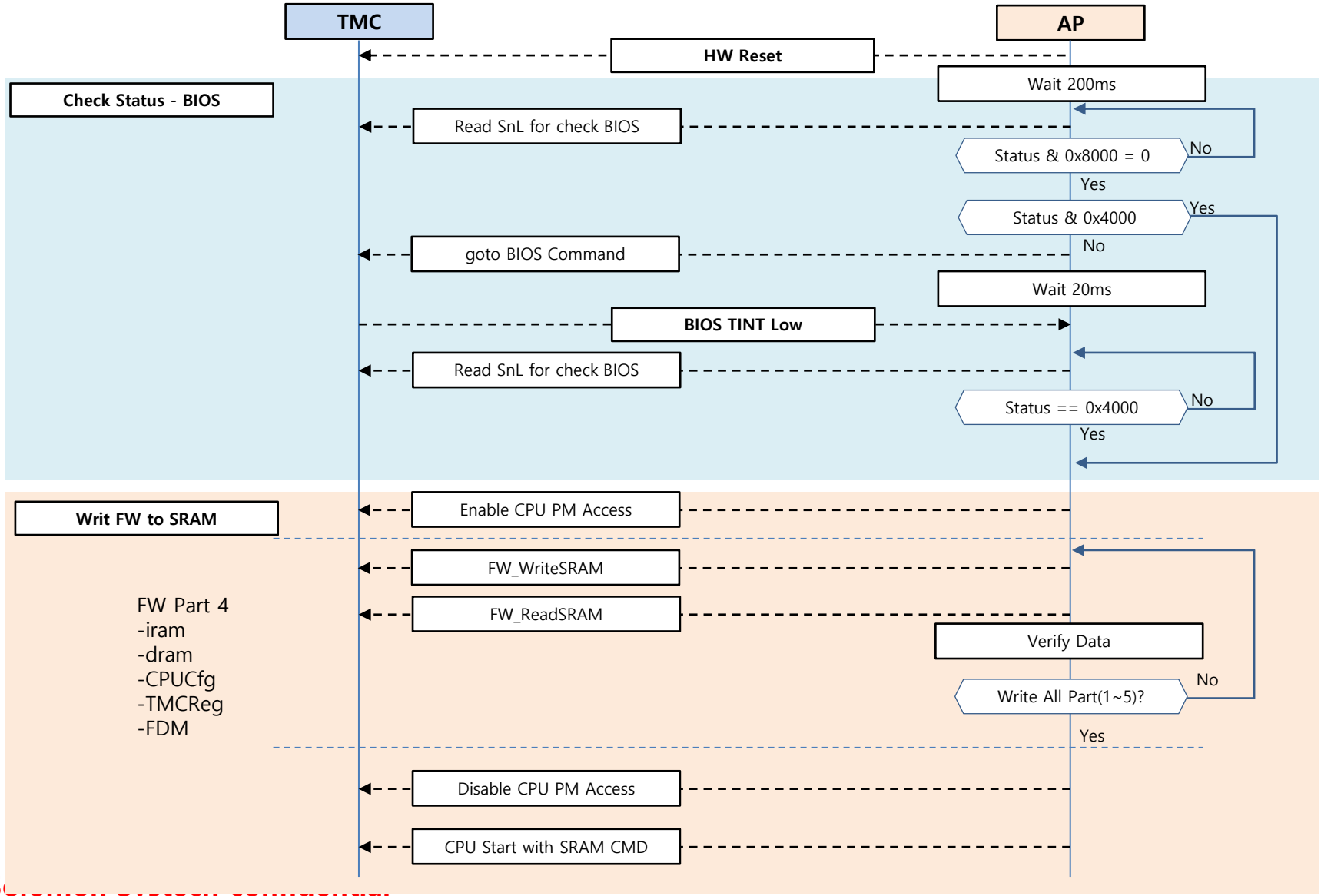
Name	Address	7bit	6bit	5bit	4bit	3bit	2bit	1bit	0bit
STATUS	0x0020	-	-	Raw(1)/PT(0)	Keep	Aux	Key	Gesture	PT EXIST
	0x0021	TIC_BUSY	TIC_IN_BIOS	TIC_IN_CPU	TINT Low	-	-	-	-
Length	0x0022	Read Length Low byte							
	0x0023	Read Length Hi byte							

when receive Touch Interrupt, read SnL. SnL has TMC status and next packet information.

Download FW to SRAM with SPI

Solomon Systech confidential

FW Download and Start up Flow with SRAM

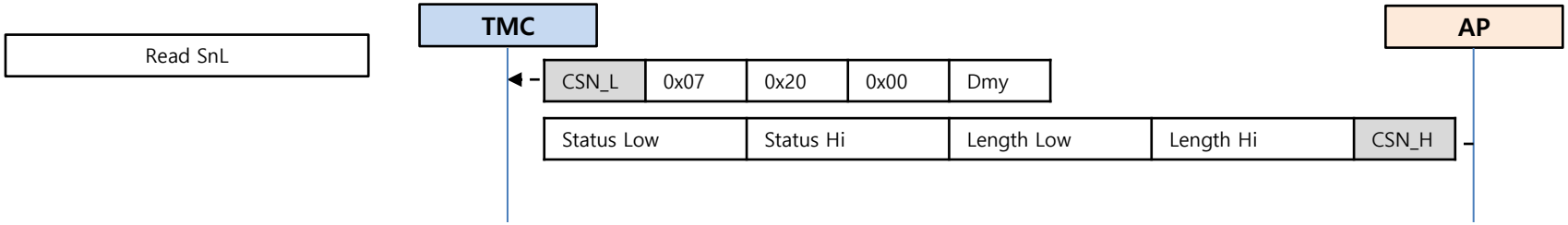


Solomon Systech confidential

FWDL : Host side, low-level function

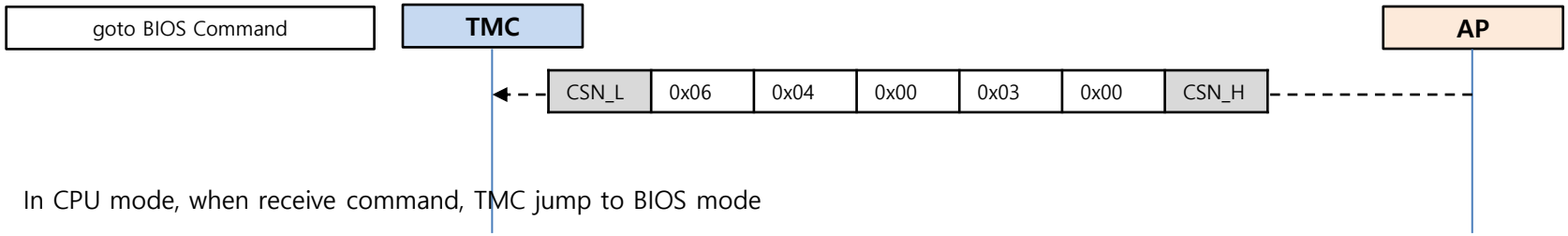
SPI Command		Packet
Write Reg BIOS	**SE_GAMMA_LL_WRITE_IC_REG_BIOS (uint32_t add, uint32 wd) ;**	(Write HDP) CSN_L 06 40 00 add0 add1 add2 add3 wd0 wd1 wd2 wd3 CSN_H (Write BIOS CMD) CSN_L 06 06 00 10 00 CSN_H
Read Reg BIOS	**SE_GAMMA_LL_READ_IC_REG_BIOS (uint32_t add, uint32* rd) ;**	(Write HDP) CSN_L 06 40 00 add0 add1 add2 add3 CSN_H (Write BIOS CMD) CSN_L 06 06 00 11 00 CSN_H (Read HDP) CSN_L 07 40 00 Dummy(1byte) rd0 rd1 rd2 rd3 CSN_H
Write BURST	**SE_GAMMA_LL_WRITE_IC_BURST_BIOS (uint32_t add, uint8_t* data, uint32_t nByte) ;**	(Write HDP) CSN_L 06 40 00 nByte0 nByte1 nByte2 nByte3 add0 add1 add2 add3 data0 data1 … data n-1 CSN_H (Write BIOS CMD) CSN_L 06 06 00 30 00 CSN_H
Read BURST	**SE_GAMMA_LL_READ_IC_BURST_BIOS (uint32_t add, uint8_t* rdata, uint32_t rnByte) ;**	(Write HDP) CSN_L 06 40 00 nByte0 nByte1 nByte2 nByte3 add0 add1 add2 add3 CSN_H (Write BIOS CMD) CSN_L 06 06 00 31 00 CSN_H SE_GAMMA_LL_BIOS_CMD_DONE( ); (Read HDP) CSN_L 07 40 00 Dummy(1byte) rd_data0 rd_data1 … rd_data n-1 CSN_H

CHECK CMD DONE	**SE_GAMMA_LL_BIOS_CMD_DONE () ;**	uint32_t SE_GAMMA_LL_BIOS_CMD_DONE ( ) {
		uint16_t rd;
		while (1) {
		(Read BIOS CMD) CSN_L 07 06 00 Dummy(1byte) rd0 rd1 CSN_H
		
		if ((rd == 0x00) break;
		}
		return 0;
		}

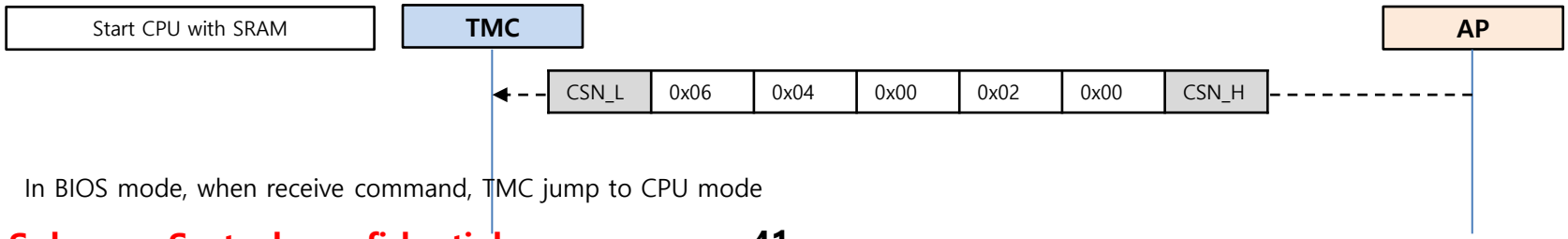


Name	Access Address	Address	7bit	6bit	5bit	4bit	3bit	2bit	1bit	0bit
TIC STATUS	0x0020 (Read Only)	0x0020	-							
		0x0021	TIC_BUSY	IN_BIOS						

TIC Status : 0x4000 - in BIOS, not Busy
need to check IN BIOS for download FW to SRAM

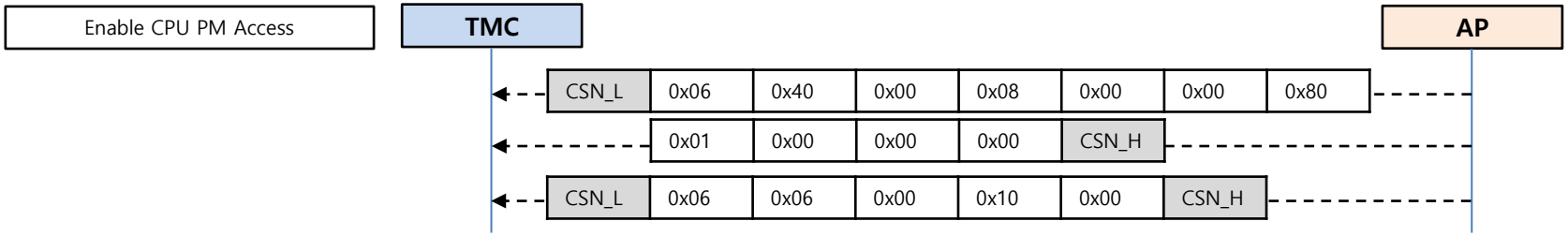


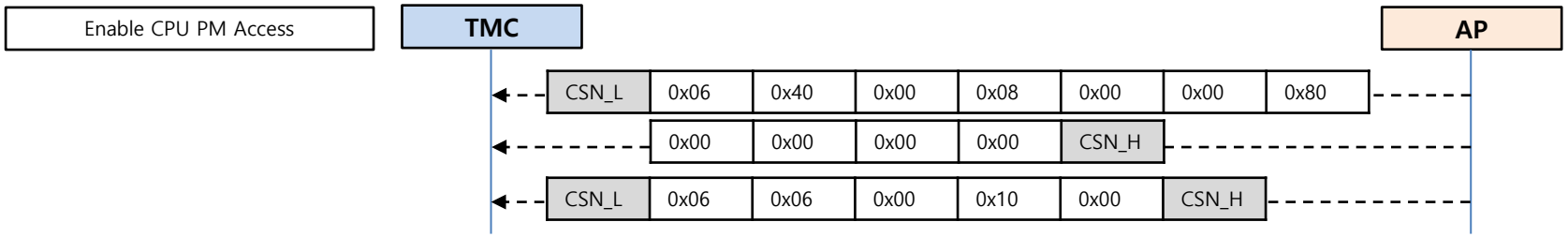
In CPU mode, when receive command, TMC jump to BIOS mode

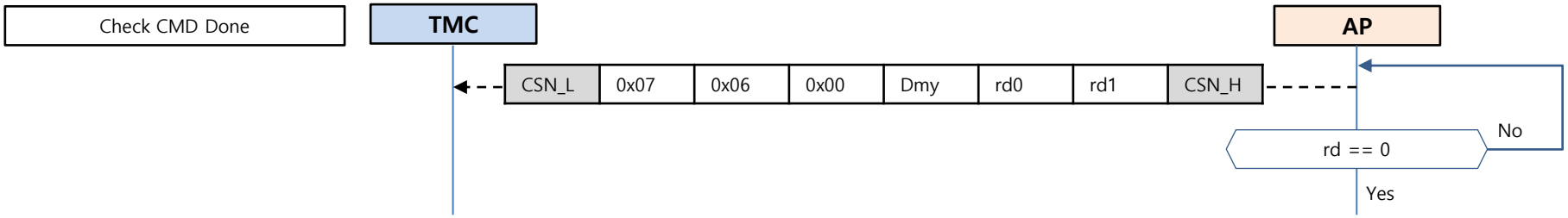


In BIOS mode, when receive command, TMC jump to CPU mode

example) I2C Protocol



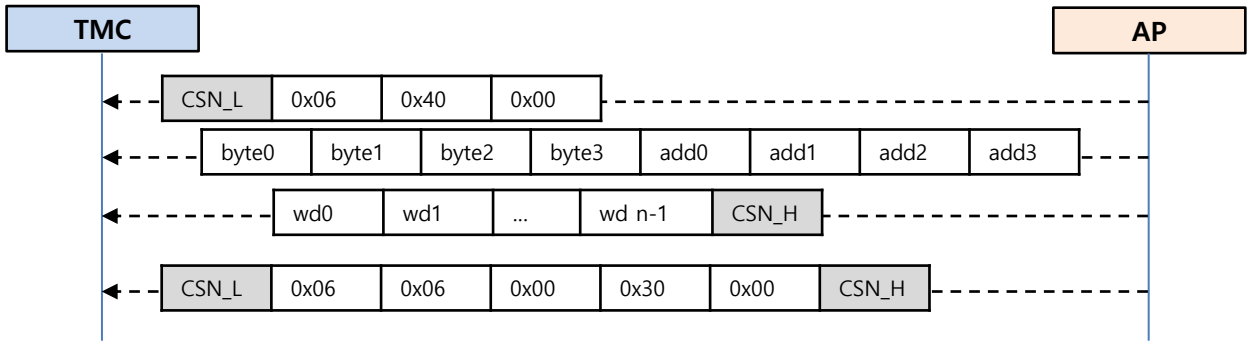




example) SPI Protocol

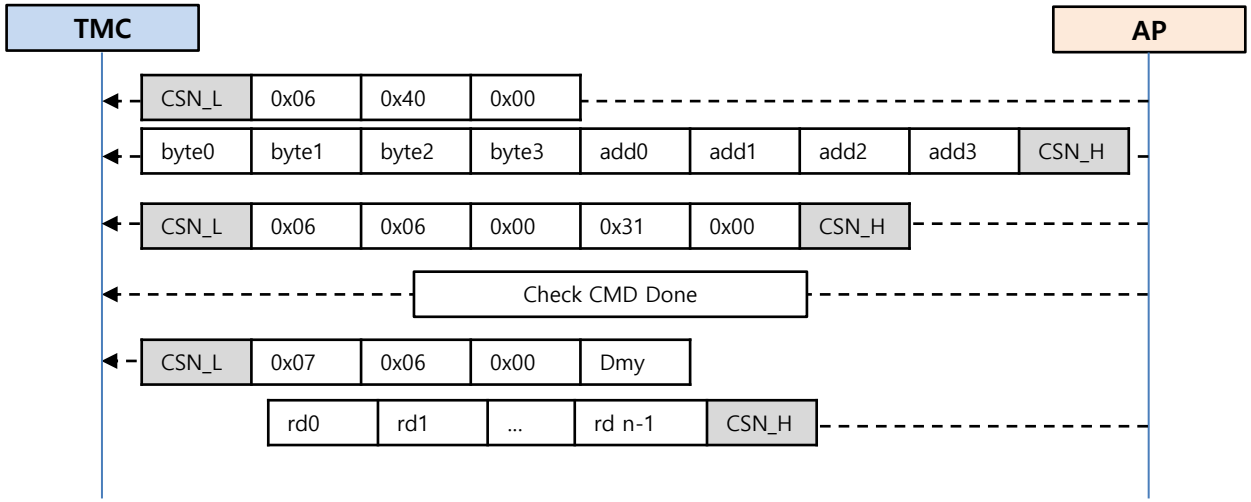
Write Burst(uint32_t add, uint32_t byte, uint8_t* wd)

Read Size Max : 1024 – 64 byte

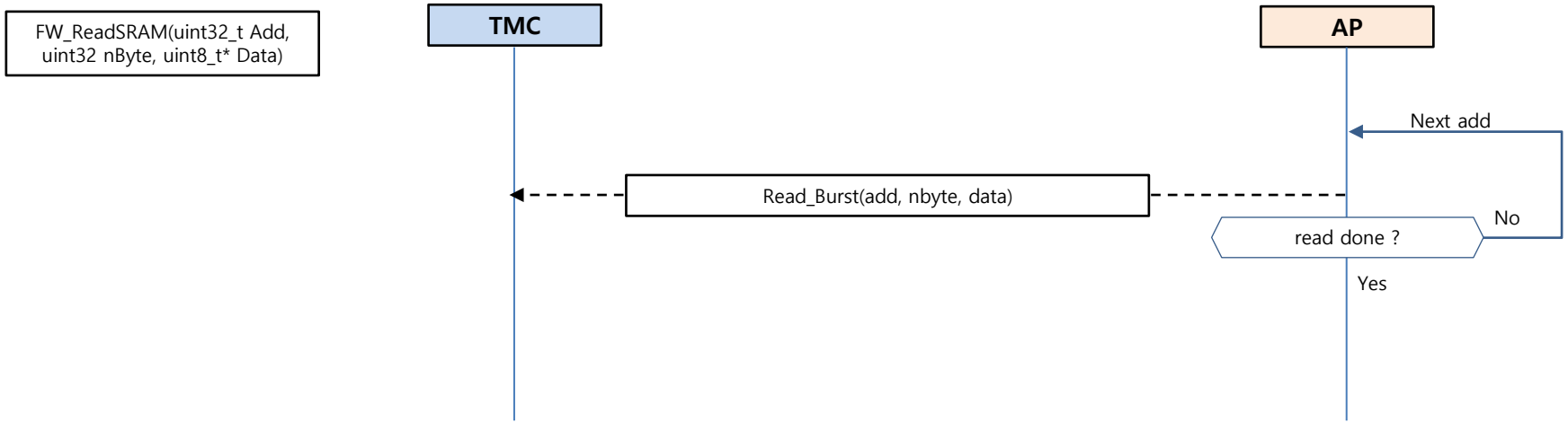


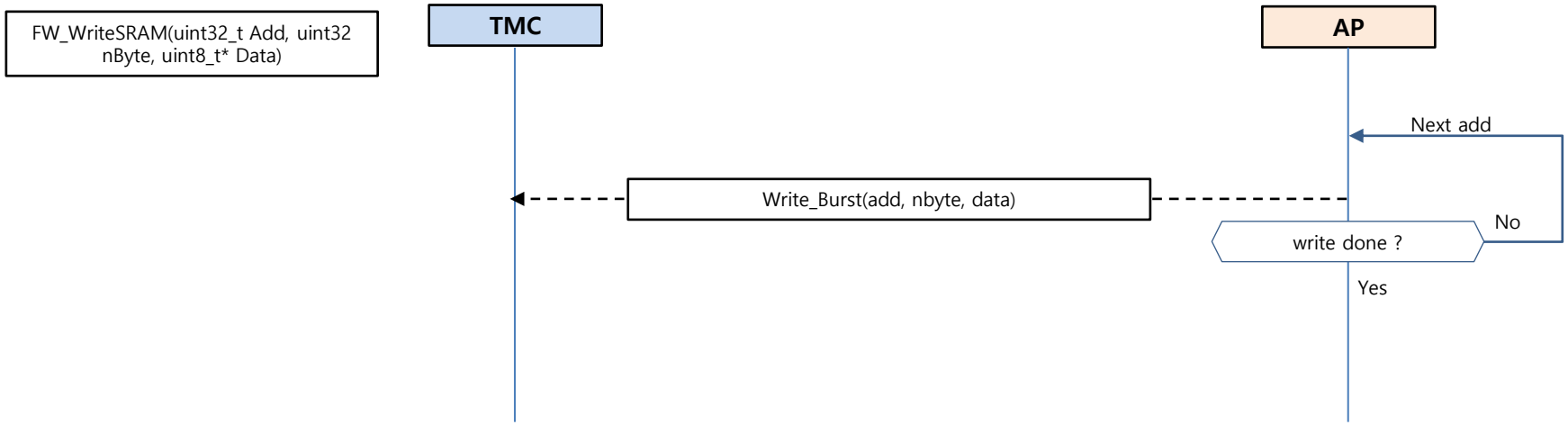
Read Burst(uint32_t add, uint32_t byte, uint8_t* rd)

Write Size Max : 1024 – 128 byte



example) SPI Protocol





FW File

Download File Description

SOLOMON SYSTECH

- ❖ mgd file parsing by each content
  - CPU
  - CPU_Cfg
  - Sys_Cfg
  - TMCReg
  - FDM

- ❖ Change write address for SRAM
  - CPU -> iram, dram
  - CPU_Cfg
  - Sys_Cfg -> remove
  - TMCReg
  - FDM

- ❖ CPU Content has iram and dram data
  - separate iram and dram data
  - CPU_Cfg has information which address and nbyte for Iram/dram

mgd file convert for SRAM

SOLOMON SYSTECH

mgd_SPD2010_7x6_210218_test_1.hex

CPU	TMCReg	FDM	Sys_Cfg	CPU_Cfg
#00000000 *0000fbe8 *00000010 *ffffffff *f75650bc *fffffcb5 *00000000 *00000000 `? <趙┐??<? ?	#0001c000 *00000264 *00000001 *00000001 *d65523d1 *ffffffff *ffffffff *ffffffff ┌ X┐ ?U? ※	#0001d000 *0000080c *00000001 *00000003 *8bf8fde6 *ffffffff *ffffffff *ffffffff └ _ ■ 艶?AB	#0001f000 *00000100 *00000001 *00000000 *bf724dee *ffffd4c1 *00000000 *00000000 ┌◀‼ ` (? 機滋 ?	#0001f400 *00000100 *00000000 *ffffffff *f1a947d7 *ffff6975 *00000000 *00000000 ┌◀‼ ? ¶? 절J

Change write address for SRAM

iram	dram	TMCReg	FDM	CPU_Cfg
#70000000 *iram data nbyte *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ uint8_t *content	#40000000 *dram data nbyte *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ uint8_t *content	#40003c00 *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ uint8_t *content	#40003F00 *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ uint8_t *content	#40003800 *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ *~~~~~~~ uint8_t *content

```
0x20131101    // [ 0x00 ] ( 0 ) Signature (20131101)
0x000000F0    // [ 0x01 ] ( 0 ) { nW                 }
0xFFFFFFFF    // [ 0x02 ] ( 0 ) { 16'hFFFF,   CPUCFG_CRC }
0xFFFFFFFF    // [ 0x03 ] ( 0 ) { CPUCFG_XOR, CPUCFG_SUM
0xFFFFFFFF    // [ 0x04 ] ( 0 ) iram data start add
0xFFFFFFFF    // [ 0x05 ] ( 0 ) iram data nbyte
0xFFFFFFFF    // [ 0x06 ] ( 0 ) { 16'hFFFF,   text_crc   }
0xFFFFFFFF    // [ 0x07 ] ( 0 ) { text_xor,   text_sum   }
0xFFFFFFFF    // [ 0x08 ] ( 0 ) dram data start add
0xFFFFFFFF    // [ 0x09 ] ( 0 ) dram data nbyte
0xFFFFFFFF    // [ 0x0A ] ( 0 ) { 16'hFFFF,   rodata_crc }
0xFFFFFFFF    // [ 0x0B ] ( 0 ) { rodata_xor, rodata_sum }
0xFFFFFFFF    // [ 0x0C ] ( 0 ) bss_dst
0xFFFFFFFF    // [ 0x0D ] ( 0 ) bss_len
0xFFFFFFFF    // [ 0x0E ] ( 0 ) - ( 0xFFFFFFFF)
0xFFFFFFFF    // [ 0x0F ] ( 0 ) - ( 0xFFFFFFFF)
```

```
#70000000
*iram data nbyte
*~~~~~~~
*~~~~~~~
*~~~~~~~
*~~~~~~~
*~~~~~~~
*~~~~~~~
uint8_t *content
```

uint32_t CPU_Cfg.Content[4] : Iram data address in cpu content
uint32_t CPU_Cfg.Content[5] : Iram data nbyte
uint32_t CPU_Cfg.Content[8] : Dram data address in cpu content
uint32_t CPU_Cfg.Content[9] : Dram data nbyte

FW_WriteSRAM(uint32_t Add, uint32 nByte, uint8_t* Data)

RAM Memory Map

- //..FW RAM Download case
- M_MAP_IRAM = 0x70000000
- M_MAP_DRAM = 0x40000000
- M_MAP_RAM_CPUCFG = 0x40003800
- M_MAP_RAM_TMCREG = 0x40003C00
- M_MAP_RAM_FDM = 0x40003F00

Flash Memory Map

- M_MAP_FW = 0x0000
- M_MAP_CPU_CFG = 0x1F400
- M_MAP_SYS_CFG = 0x1F000
- M_MAP_TMCREG = 0x1C000
- M_MAP_FDM = 0x1D000

Read Data Description

Solomon Systech confidential

Description of Read Data from HDP(0x0300)

SOLOMON SYSTECH

I2C	SPI
S | 0x54 | W | 0x00 | 0x03 | P	CSN_L | 0x07 | 0x00 | 0x03 | Dmy
S | SA | W | rd 0 | rd 1 | ... | rd n-1 | P	rd 0 | rd 1 | ... | rd n-1 | CSN_H

rd 0 | rd 1 | ... | rd n-1

Packet		Header	Data
Point Info	IC->H	Packet Code | DataInfo | Index | CS	Point Info | | |
RawData	IC->H	Packet Code | DataInfo | Index | CS	Add L | Add H | dummy | dummy | Data 0 | Data 1 | Data ... | Data N

Packet Code	Type -AFE : 0x20, Delta : 0x30, Baseline : 0x40, MDelta : 0x50 Mode - Direction -HostToIC : 0x00, ICToHost : 0x02 Packet Code = Type | Mode	
Index	Packet Index when refresh hdp data, increase value 1 for distinguish old data.	
Data Info	Bit 7: 0(fix)	
Xor CS(CheckSum)	Data0 ~ Data n-1. XOR CheckSum	

Packet Code				
Type[7:4]	Mode[3:0]			
			Dir	
1	0	0	1	0
2~5	0	0	1/0	0

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DataInfo	0(fix)	Last Packet	-	-		-	-	

Packet - Point Information format

0x12	0x00	Index L	Index H

Pint ID	X L	Y L	XH/YH	W	Reserved
0x00	0x0B	0x04	0x11	0x86	0x00

Pint ID	X L	Y L	XH/YH	W	Reserved
0x01	0x0B	0x04	0x11	0x86	0x00

…

Pint ID	X L	Y L	XH/YH	W	Reserved
0x09	0x0B	0x04	0x11	0x86	0x00

8bit 0bit

Point ID 0	
X Position LSB	
Y Position LSB	
X Position MSB(4bit)	Y Position MSB(4bit)
Weight	
-	

Point ID : 0
X Position : 0x10B(267)
Y Position : 0x104(260)
weight : 134

…

~ Point ID 9	
X Position LSB	
Y Position LSB	
X Position MSB(4bit)	Y Position MSB(4bit)
Weight	
-	

Packet – extension Point Information Data

0x12	0x00	Index	Xor CS	Pint ID	X L	Y L	XH/YH	W	Reserved	...	if exist point Info
				Key ID	Key L	Key H	Dummy	Dummy	Dummy		if exist Key Info
				Gesture ID	Gesture L	Gesture H	Dummy	Dummy	Dummy		if exist Gesture Info
				Aux ID	Aux L	Aux H	Dummy	Dummy	Dummy		if exist Aux Info

- Data Part - each data part has 6 bytes
  - Point Info(each point has 6 bytes)
  - Key Info
  - Gesture Info
  - Aux Info
- Packet structure
  - add each part, when generate upload data
  - ex)
    - only Point Info
    - only Key Info
    - only Gesture Info
    - only Aux Info
    - Point Info + Gesture + Key …

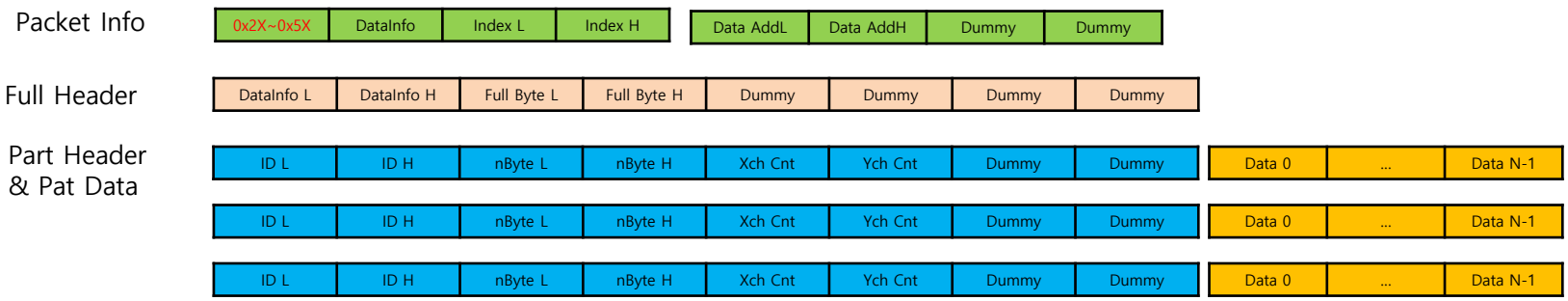
8bit							0bit
Point ID (0x00 ~ 0x0A)							
X Position LSB							
Y Position LSB							
X Position MSB(3bit)				Y Position MSB(3bit)			
Weight							
-							
Gesture ID (0xF6)							
Gesture ASCII Code							
-	-	-	-		DTap	Palm Reject	Large
-							
-							
-							
Key ID (0xF5)							
...	...	....	...	...	Key 03 Down	Key 02 Down	Key 01 Down
Key 16 Down	...	...	...	...	...	...	...
...	...	....	...	...	Key 03 Up	Key 02 Up	Key 01 Up
Key 16 Up	...	...	...	...	...	...	...
-							

Gesture ASCII Value

Gesture ID (0xF6)							
Gesture ASCII Code							
-	-	-	-		DTap	Palm Reject	Large
-							
-							
-							

Gesture	ASCII	Gesture	ASCII
Slide Left('L')	0x4C	'>'	0x3e
Slide Right('R')	0x52	'v'	0x76
Slide Top('T')	0x54	'^'	0x5e
Slide Bottom('B')	0x42	'w'	0x77
'o'	0x6f	'm'	0x6d
'c'	0x63	'z'	0x7a
		's'	0x73

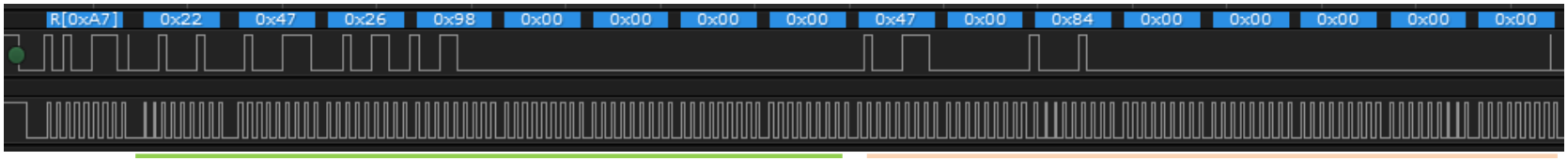
Packet - RawData format



	Name	Description
Full Header	Data Info	Data Information, include data type
	Full Byte	Full byte count of Data, except Full header byte count
Part Header	ID	indicate data part, 0x01 : Normal Cell, 0x02 : Large Cell, 0x04 : Key Data if set data type, add rawdata packet. normally upload only Normal Cell Data
	nByte	data byte count not include part header byte count (same as Xch*Ych*2)
	Xch	Xch count
	Ych	Ych count
Part Data	Data	each part Raw Data, (2 byte / node)

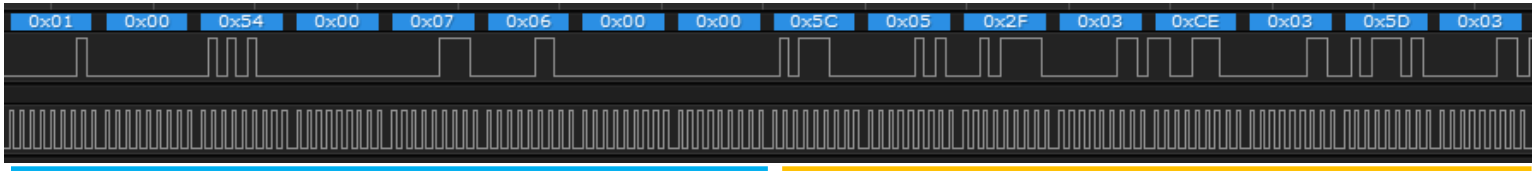
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DataInfo	0(fix)	Last Packet	-	-	NMS	KEY	LC	NC

example) RawData Packet



PacketCode	DataInfo	Index	Xor CS	Data AddL	Data AddH	Dummy	Dummy
0x22	0x47	0x00	0x84	0x00	0x00	0x00	0x00

DataInfo L	DataInfo H	Full Byte L	Full Byte H	Dummy	Dummy	Dummy	Dummy
0x47	0x00	0x84	0x00	0x00	0x00	0x00	0x00



...

NC ID L	NC ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy
0x01	0x00	0x54	0x00	0x07	0x06	0x00	0x00

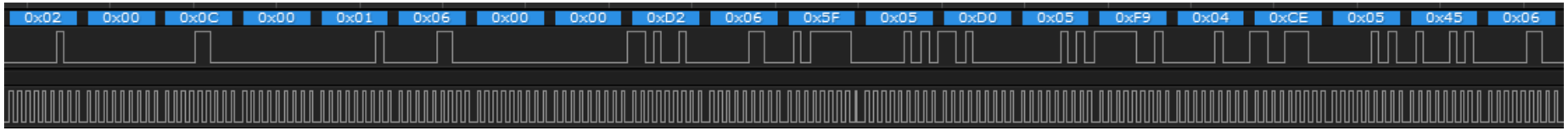
NC Data 0	...	NC Data N-1
0x5C	....	

NC Data
-nByte : 0x0054(84)
-Xch Cnt : 7
-Ych Cnt : 6
-Node Cnt : 42

1node – 2byte

	Ych0	Ych1	Ych2	Ych3	Ych4	Ych5
Xch0	1747	1102	1286	1178	1294	1349
Xch1	1269	1149	1068	1233	1077	1229
Xch2	1418	1028	1221	1103	1218	1108
Xch3	1289	1159	1081	1241	1084	1243
Xch4	1412	1030	1223	1105	1213	1109
Xch5	1277	1163	1083	1233	1079	1232
Xch6	1679	1039	1227	1100	1211	1307

example) RawData Packet



LC ID L	NC ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy
0x02	0x00	0x0C	0x00	0x01	0x06	0x00	0x00

LC Data 0	...	LC Data N-1
0xD2	....	

LC Data
-nByte : 0x000C(12)
-Xch Cnt : 1
-Ych Cnt : 6
-Node Cnt : 6

1node – 2byte

	Ych0	Ych1	Ych2	Ych3	Ych4	Ych5
Xch0	1460	1170	1275	1172	1288	1375

Key ID L	Key ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy
0x04	0x00	0x0C	0x00	0x01	0x06	0x00	0x00

Key Data 0	...	LC Data N-1
0x1A	....	

P

Key Data
-nByte : 0x000C(12)
-Xch Cnt : 1
-Ych Cnt : 6
-Node Cnt : 6

1node – 2byte

	Ych0	Ych1	Ych2	Ych3	Ych4	Ych5
Xch0	1460	1170	1275	1172	1288	1375

Description - HDP Status(0x2FC)

I2C	SPI
S | 0x54 | W | 0xFC | 0x02 | P	CSN_L | 0x07 | 0xFC | 0x02 | Dmy
S | SA | W | rd 0 | rd 1 | ... | rd n-1 | P	rd 0 | rd 1 | ... | rd n-1 | CSN_H

HDP Status : Host Data Port Staus
- Host read HDP Status for next action.
- if HDP Status's 7bit is 0 and different 'Index' with previous read packet, TIC set new upload data. need to read packet
- if HDP Status's 7bit is set and Keep Poll is set, retry read status until change HDP Status.
- if HDP Status's 7bit is set and Req INT Clear is set, send 'Clear TINT' command and finish read.

Dummy	Dummy	nByte L	nByte H	PacketCode	HDP Status	Index	CS

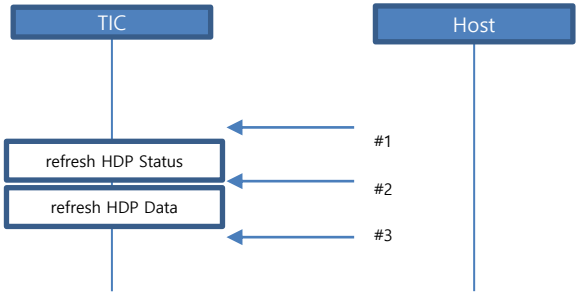
Next Packet Length

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
1	-	-	-	-	Request Data	Req INT Clear	Keep Poll

Upload Data Ready
- difference N
- HDP Status Data Bit7 is 0

Timing of Read HDP Status

refresh HDP Data	0xX2	0x00	Index	#1 Previous Data
refresh HDP Status	0xX2	0x81	x	#2 not ready next action
refresh HDP Next Data	0xX2	0x00	Index + 1	#3 ready upload data
refresh HDP Status	0xX2	0x82	X	#2 request clear TINT command
refresh HDP Status	0xX2	0x84	Next Data Pointer	#2 request Data (TBOD)



Description – HDP Status(0x2FC)

SOLOMON SYSTECH

❖ HDP Status

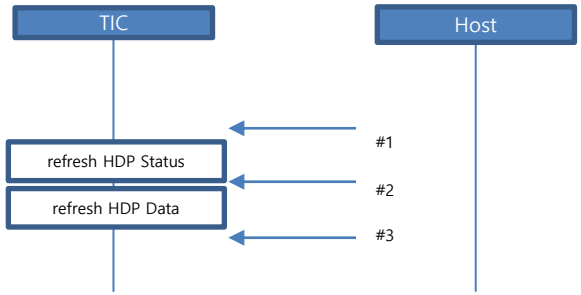
- read 8byte from address 0x02FC for check HDP Status

0x02FC	0x02FD	0x02FE	0x02FF	0x0300	0x0301	0x0302	0x0303
-	-	nByte L	nByte H	Packet Code	HDP Status	Index	CS

- HDP Status and HDP have common memory area
  - address 0x0301~0x0303 has new data information
  - case of new data, HDP Status 7bit is 0 and Index increase
    - new data length : nByte H/L
- when upload data,
  - refresh HDP Status and then refresh HDP

Timing of Read HDP Status

refresh HDP Data	0xX2	0x00	Index	#1 Previous Data
refresh HDP Status	0xX2	0x81	x	#2 not ready next action
refresh HDP Status	0xX2	0x82	X	#2 request clear TINT command
refresh HDP Next Data	0xX2	0x00	Index + 1	#3 ready upload data



❖ HDP Status

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
1	-	-	-	-	Request Data	Req TINT Clear	Keep poll

- bit 7
  - Fix – 1 – HDP Status indicator
- Request Data
  - HDP Area Free. request write next data(use TBOD)
- Req TINT Clear
  - Request TINT Clear Command
  - empty upload data buffer
- Keep poll
  - not ready data. polling HDP Status

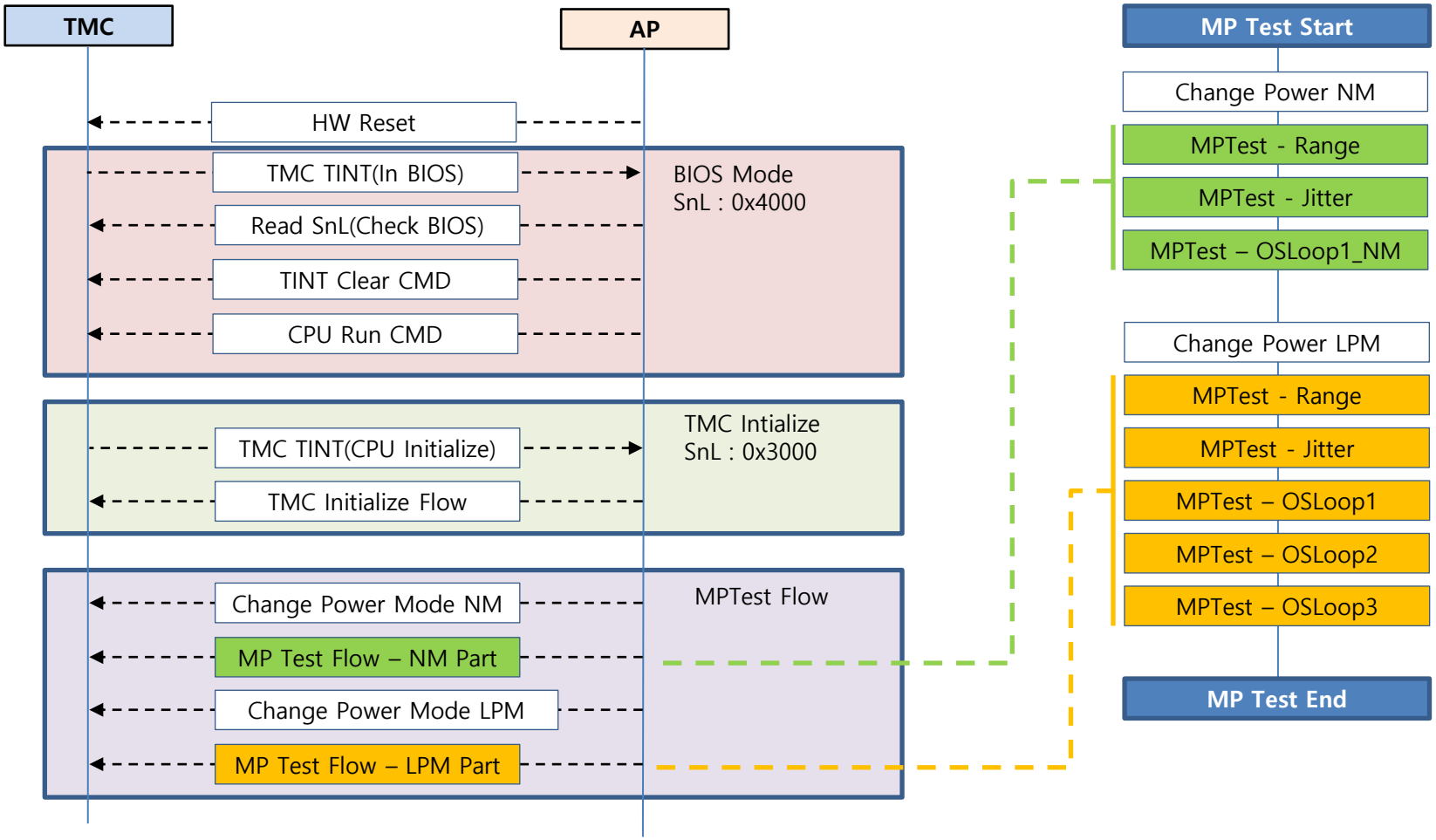
MPTest Protocol

Solomon Systech confidential

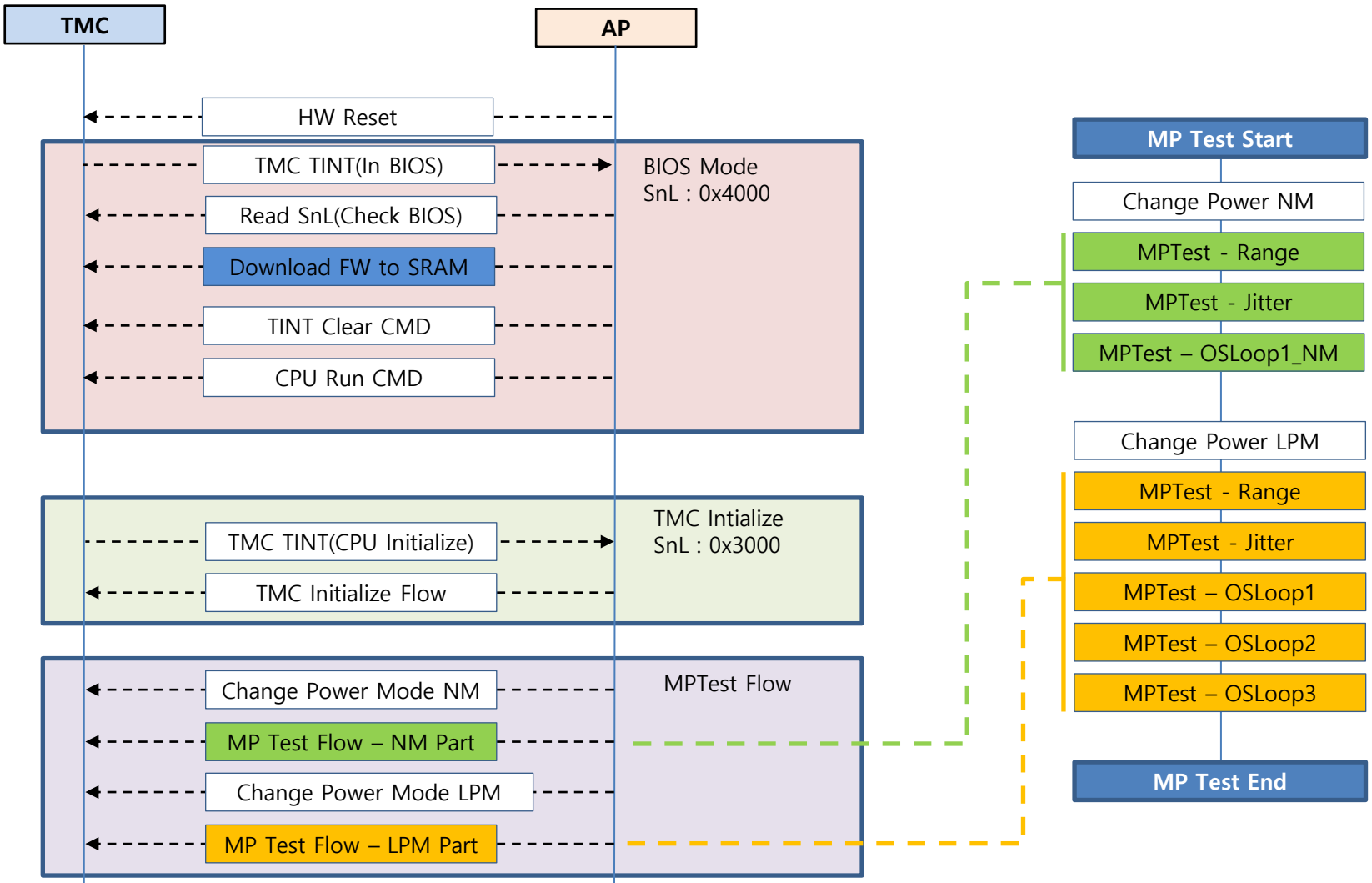
Compare protocol I2C and SPI

Interface	Protocol
I2C Read	S | 0x54 | W | Add L | Add H | P S | SA | W | rd 0 | rd 1 | … | rd n-1 | P
I2C Write	S | SA | W | 0x02 | 0x00 0x04 | 0x00 | 0x01 | 0x00 | P
SPI Read	CSN_L | 0x07 | Add L | Add H | Dmy rd 0 | rd 1 | … | rd n-1 | CSN_H
SPI Write	CSN_L | 0x06 | Add L | Add H wd0 | wd1 | … | wd n-1 | CSN_H

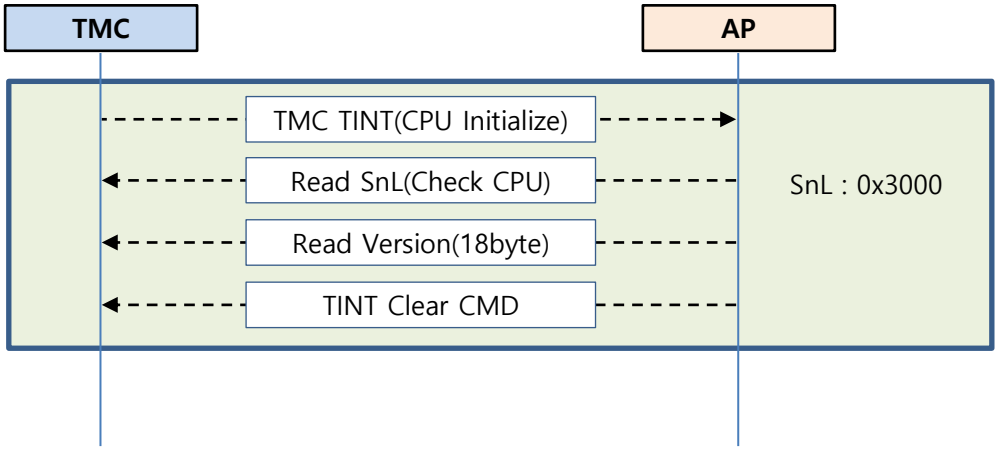
MPTest Full Flow with SEEPROM(Host IF : I2C)



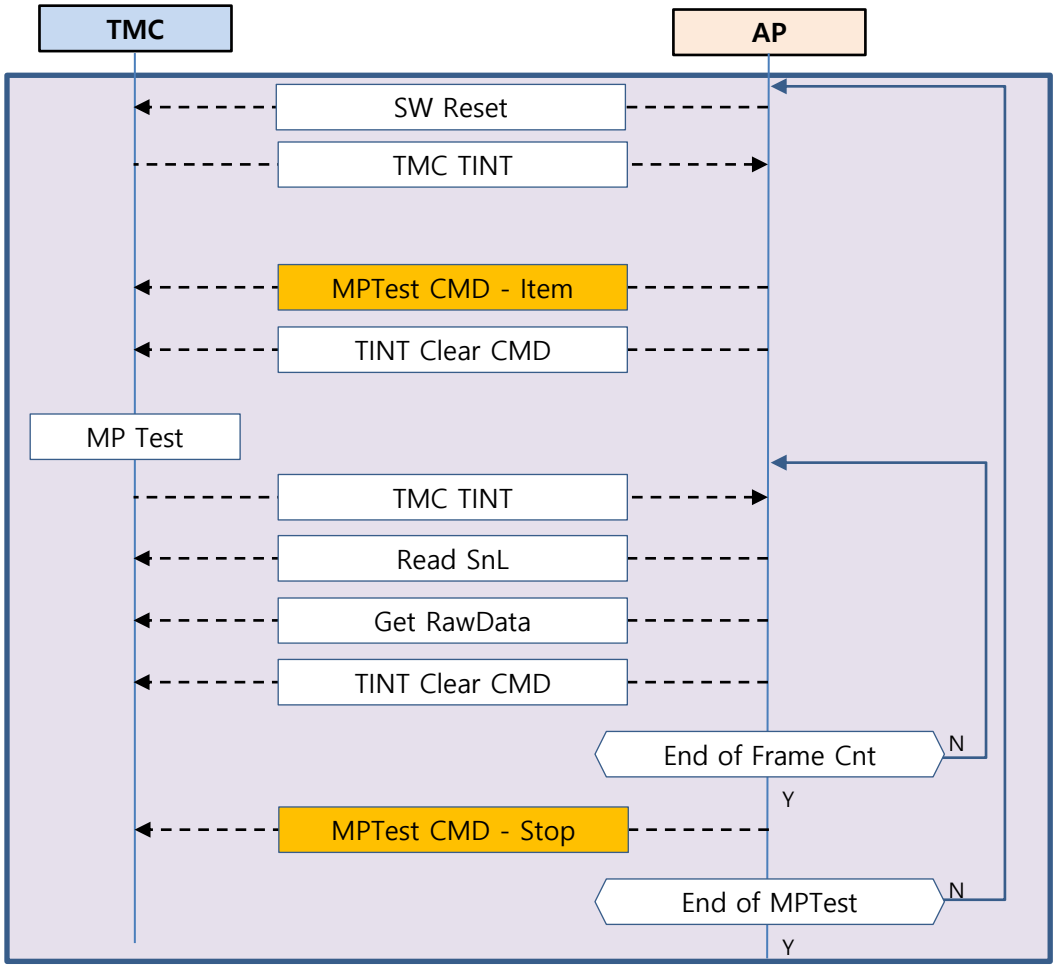
MPTest Full Flow with SRAM(Host IF : SPI)



TMC Initialize Flow



MPTest Flow



MP CMD - Item	Value(2byte)
Stop	0x0000
Range	0x0001
Jitter	0x0002
OSLoop1_NM	0x0101
OSLoop1	0x0201
OSLoop2	0x0302
OSLoop3	0x0402

Protocol – I2C

Read SnL	S | 0x54 | W | 0x20 | 0x00 | P S | SA | R | Status L | Status H | Length L | Length H | P
TINT Clear CMD	S | SA | W | 0x02 | 0x00 | 0x01 | 0x00 | P
RUN CPU with SEEPROM	S | SA | W | 0x04 | 0x00 | 0x01 | 0x00 | P
Read Version	S | 0x54 | W | 0x26 | 0x00 | P S | SA | R | Data 0 | Data 1 | … | Data 17 | P
SW Reset	S | SA | W | 0x46 | 0x00 | 0x01 | 0x08 | P
MPTest CMD	S | SA | W | 0x4A | 0x00 | CMD L | CMD H | P
Get RawData	S | 0x54 | W | 0x00 | 0x03 | P S | SA | R | Data 0 | Data 1 | … | Data n-1 | P n = Length in SnL
Power Mode NM	S | SA | W | 0x46 | 0x00 | 0x00 | 0x90 | P + Delay 100ms
Power Mode LPM	S | SA | W | 0x46 | 0x00 | 0x01 | 0x90 | P + Delay 100ms

Protocol - SPI

Read SnL	CSN_L | 0x07 | 0x20 | 0x00 | Dmy Status Low | Status Hi | Length Low | Length Hi | CSN_H
TINT Clear CMD	CSN_L | 0x06 | 0x02 | 0x00 | 0x01 | 0x00 | CSN_H
RUN CPU with SRAM	CSN_L | 0x06 | 0x04 | 0x00 | 0x02 | 0x00 | CSN_H
Read Version	CSN_L | 0x07 | 0x26 | 0x00 | Dmy Data 0 | Data 1 | … | Data 17 | CSN_H
SW Reset	CSN_L | 0x06 | 0x46 | 0x00 | 0x01 | 0x08 | CSN_H
MPTest CMD	CSN_L | 0x06 | 0x4A | 0x00 | CMD L | CMD H | CSN_H
Get RawData	CSN_L | 0x07 | 0x00 | 0x03 | Dmy Data 0 | Data 1 | … | Data n-1 | CSN_H n = Length in SnL
Power Mode NM	CSN_L | 0x06 | 0x46 | 0x00 | 0x00 | 0x90 | CSN_H + Delay 100ms
Power Mode LPM	CSN_L | 0x06 | 0x46 | 0x00 | 0x01 | 0x90 | CSN_H + Delay 100ms

Protocol – SnL(Status and Length) Detail

	Read SnL
I2C	S | 0x54 | W | 0x20 | 0x00 | P S | SA | R | Status L | Status H | Length L | Length H | P
SPI	CSN_L | 0x07 | 0x20 | 0x00 | Dmy Status L | Status H | Length L | Length H | CSN_H

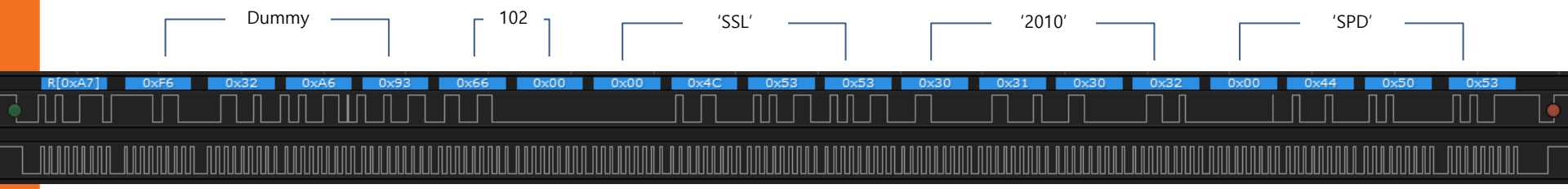
Name	Access Address	Address	7bit	6bit	5bit	4bit	3bit	2bit	1bit	0bit
TIC STATUS	0x0020 (Read Only)	0x0020	-		Raw(1)/PT(0)	Keep	Aux	Key	Gesture	PT EXIST
		0x0021	TIC_BUSY	TIC_BIOS	CPU_BOOT	TINT(1: Low)	-	-	-	NM/LPM
		PT EXIST : Detect Point Gesture : Detect Gesture Key : Detect Gesture Aux : reserved Keep : - NM/LPM : Power mode indicator, 0 : Normal Mode, 1 : Low power Mode TINT : Touch Interrupt indicator, 0 : TINT Hi, 1 : TINT Low CPU_BOOT : CPU initialize – touch initialize interrupt indicator TIC_BIOS : working in BIOS TIC_BUSU : IC busy, if set this bit, retry after delay.								
Length	0x0022 (Read Only)	0x0022								
		0x0023								
		when Host receive Interrupt, read data length(byte count), Full Length.								

Protocol – Read Version Detail

	Read Version(18bytes)
I2C	S | 0x54 | W | 0x26 | 0x00 | P S | SA | R | Data 0 | Data 1 | … | Data 17 | P
SPI	CSN_L | 0x07 | 0x26 | 0x00 | Dmy Data 0 | Data 1 | … | Data 17 | CSN_H

Version Data Part	ByteCnt
Dummy	4byte
D-Version(Dec)	2byte
PID(ASCII)	4byte
ICName(ASCII)	8byte

ex)Version 18 bytes – 0x00 skip in PID and ICName



Protocol - MPTest Command

	MP Test Command
I2C	S | SA | W | 0x4A | 0x00 | CMD L | CMD H | P
SPI	CSN_L | 0x06 | 0x4A | 0x00 | CMD L | CMD H | CSN_H

MP CMD	Value(2byte)
Stop	0x0000
Range	0x0001
Jitter	0x0002
OSLoop1_NM	0x0101
OSLoop1	0x0201
OSLoop2	0x0302
OSLoop3	0x0402

Protocol – Get Data(Raw Data or Point Info)

	Get Data
I2C	S | 0x54 | W | 0x00 | 0x03 | P S | SA | R | Data 0 | Data 1 | … | Data n-1 | P
SPI	CSN_L | 0x07 | 0x00 | 0x03 | Dmy Data 0 | Data 1 | … | Data n-1 | CSN_H

RawData Format

S	SlaveAdd W	HDP L	HDP H	P

S	SlaveAdd R	PacketCode	DataInfo	Index	Xor CS	Data AddL	Data AddH	Dummy	Dummy

Full Header	DataInfo L	DataInfo H	Full Byte L	Full Byte H	Dummy	Dummy	Dummy	Dummy			
Part Header	NC ID L	NC ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy	NC Data 0	...	NC Data N-1
	LC ID L	LC ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy	LC Data 0	...	LC Data N-1
	Key ID L	Key ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy	Key Data 0	...	Key Data N-1

P

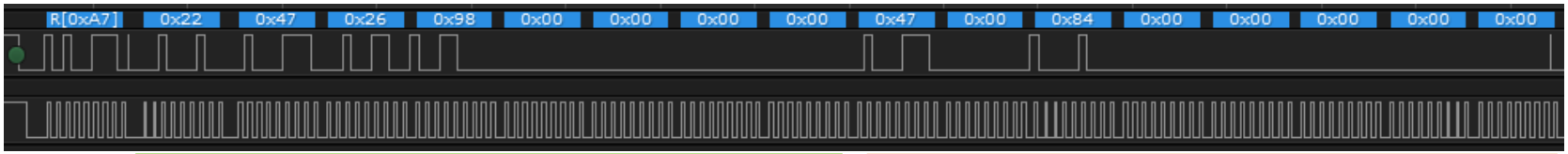
❖ Raw Data extention

- Full Header
  - DataInfo :

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DataInfo	0(fix)	x	-	-	NMS	KEY	LC	NC

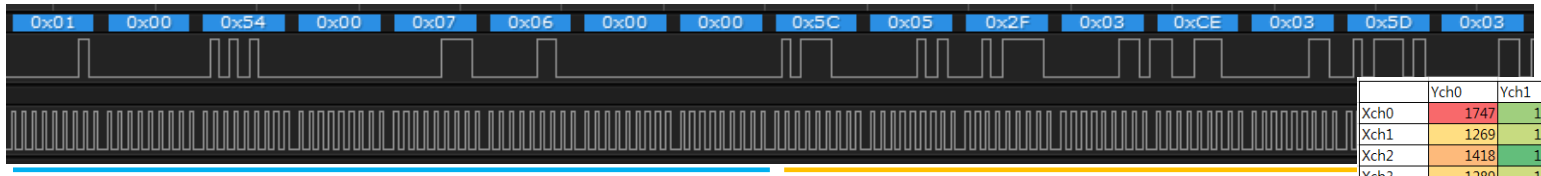
  - Full Byte : data byte count except Full Header(8byte)
- Part Header : each Data Header
  - ID : Data type (0x01 : NC, 0x02 : LC, 0x04 : Key)
  - nByte : Part Data byte count – except part header(8byte)
  - Xch / Ych : Channel count of part data
- Data : part data

example) RawData Format



PacketCode	DataInfo	Index	Xor CS	Data AddL	Data AddH	Dummy	Dummy
0x22	0x47	0x00	0x84	0x00	0x00	0x00	0x00

DataInfo L	DataInfo H	Full Byte L	Full Byte H	Dummy	Dummy	Dummy
0x47	0x00	0x84	0x00	0x00	0x00	0x00



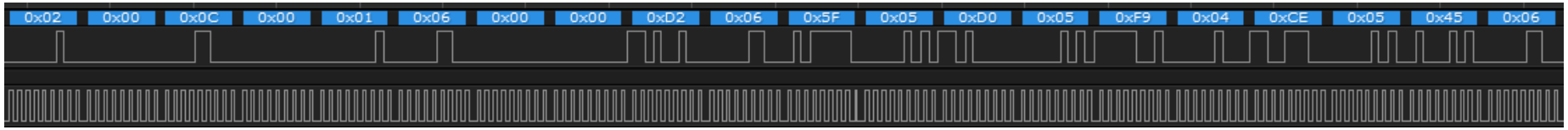
...

	Ych0	Ych1	Ych2	Ych3	Ych4	Ych5
Xch0	1747	1102	1286	1178	1294	1349
Xch1	1269	1149	1068	1233	1077	1229
Xch2	1418	1028	1221	1103	1218	1108
Xch3	1289	1159	1081	1241	1084	1243
Xch4	1412	1030	1223	1105	1213	1109
Xch5	1277	1163	1083	1233	1079	1232
Xch6	1679	1039	1227	1100	1211	1307

node : 2 byte

NC ID L	NC ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy
0x01	0x00	0x54	0x00	0x07	0x06	0x00	0x00

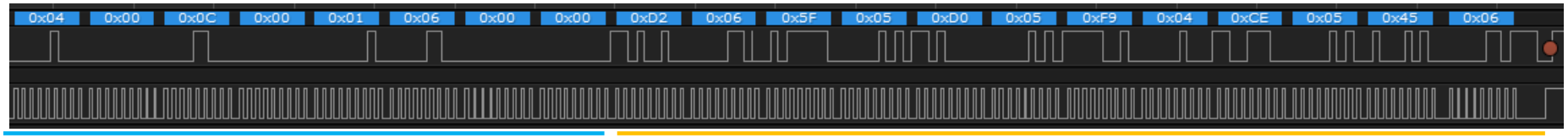
NC Data 0	...	NC
0x5C	....	



LC ID L	NC ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy
0x02	0x00	0x0C	0x00	0x01	0x06	0x00	0x00

LC Data 0	...	LC Data N-1
0xD2	....	

	Ych0	Ych1	Ych2	Ych3	Ych4	Ych5
Xch0	1460	1170	1275	1172	1288	1375



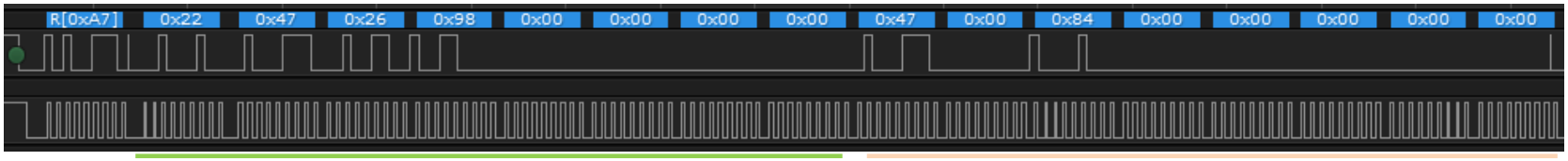
Key ID L	Key ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy
0x04	0x00	0x0C	0x00	0x01	0x06	0x00	0x00

Key Data 0	...	LC Data N-1
0x1A	....	

P

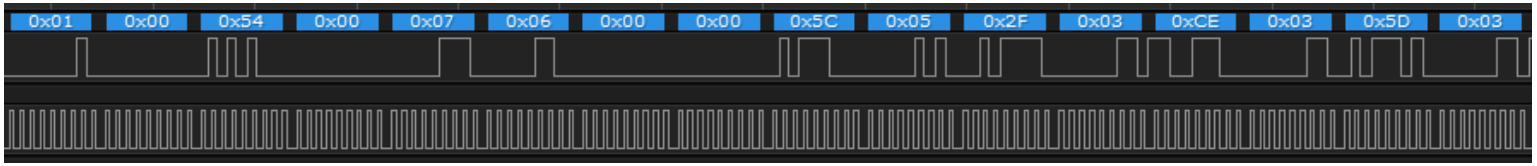
	Ych0	Ych1	Ych2	Ych3	Ych4	Ych5
Xch0	1460	1170	1275	1172	1288	1375

example) RawData Format



PacketCode	DataInfo	Index	Xor CS	Data AddL	Data AddH	Dummy	Dummy
0x22	0x47	0x00	0x84	0x00	0x00	0x00	0x00

DataInfo L	DataInfo H	Full Byte L	Full Byte H	Dummy	Dummy	Dummy	Dummy
0x47	0x00	0x84	0x00	0x00	0x00	0x00	0x00



...

NC ID L	NC ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy
0x01	0x00	0x54	0x00	0x07	0x06	0x00	0x00

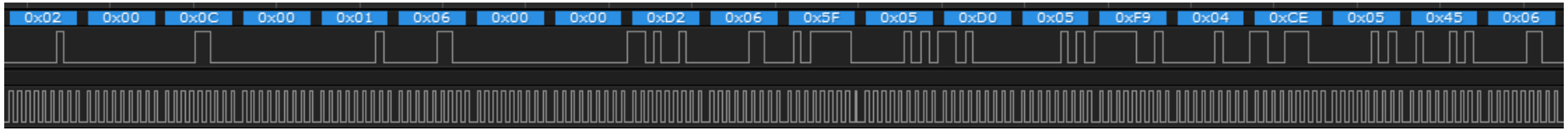
NC Data 0	...	NC Data N-1
0x5C	....	

NC Data
-nByte : 0x0054(84)
-Xch Cnt : 7
-Ych Cnt : 6
-Node Cnt : 42

1node – 2byte

	Ych0	Ych1	Ych2	Ych3	Ych4	Ych5
Xch0	1747	1102	1286	1178	1294	1349
Xch1	1269	1149	1068	1233	1077	1229
Xch2	1418	1028	1221	1103	1218	1108
Xch3	1289	1159	1081	1241	1084	1243
Xch4	1412	1030	1223	1105	1213	1109
Xch5	1277	1163	1083	1233	1079	1232
Xch6	1679	1039	1227	1100	1211	1307

example) RawData Format



LC ID L	NC ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy
0x02	0x00	0x0C	0x00	0x01	0x06	0x00	0x00

LC Data 0	...	LC Data N-1
0xD2	....	

LC Data
-nByte : 0x000C(12)
-Xch Cnt : 1
-Ych Cnt : 6
-Node Cnt : 6

1node – 2byte

	Ych0	Ych1	Ych2	Ych3	Ych4	Ych5
Xch0	1460	1170	1275	1172	1288	1375

Key ID L	Key ID H	nByte L	nByte H	Xch Cnt	Ych Cnt	Dummy	Dummy
0x04	0x00	0x0C	0x00	0x01	0x06	0x00	0x00

Key Data 0	...	LC Data N-1
0x1A	....	

P

Key Data
-nByte : 0x000C(12)
-Xch Cnt : 1
-Ych Cnt : 6
-Node Cnt : 6

1node – 2byte

	Ych0	Ych1	Ych2	Ych3	Ych4	Ych5
Xch0	1460	1170	1275	1172	1288	1375

TMC and Host Action by Command

SOLOMON SYSTECH

MP CMD	Description
Stop	TMC Action : MP Test Stop
Range	TMC Action : Upload AFE Data Host Action : first 2 frame skip for garbage data remove. Getting 10 Frame Data -> make average 10 Frame -> check Max/Min value for range
Jitter	TMC Action : generate Jitter data for 1~2s. Host Action : - first 2 frame skip for garbage data remove. - Getting 1 Frame Data -> check Max/Min value
OSLoop1_NM	TMC Action : generate OSLoop1_NM data Host Action : - first 2 frame skip for garbage data remove. - Getting 1 Frame Data -> generate Vertical Diff value -> check Max/Min value
OSLoop1	TMC Action : generate OSLoop1 data Host Action : - first 2 frame skip for garbage data remove. - Getting 1 Frame Data -> generate Vertical Diff value -> check Max/Min value
OSLoop2	TMC Action : generate OSLoop2 data Host Action : - first 2 frame skip for garbage data remove. - Getting 1 Frame Data -> check Max/Min value
OSLoop3	TMC Action : generate OSLoop3 data Host Action : - first 2 frame skip for garbage data remove. - Getting 1 Frame Data -> check Max/Min value

Criteria

SOLOMON SYSTECH

❖ MP Test Criteria

- file : MPRange_SPD2010_*.ini
- Data Type
  - NC
  - LC
  - Key – reserved
  - NC_VDIFF – Host Generate
- How to check Data
  - out of range – Test Fail

```
==========================
[NM_Range]
==========================
NC_MAX_TH = 3000
NC_MIN_TH = 200

//LC_MAX_TH = 3000
//LC_MIN_TH = 200


==========================
[NM_Jitter]
==========================
NC_MAX_TH = 250
NC_MIN_TH = 0

//LC_MAX_TH = 250
//LC_MIN_TH = 0


==========================
[NM_OSLoop1]
==========================
NC_VDIFF_MAX_TH = 300
NC_VDIFF_MIN_TH = -300


==========================
[LPM_Range]
==========================
NC_MAX_TH = 2300
NC_MIN_TH = 200

LC_MAX_TH = 2300
LC_MIN_TH = 200
```

```
==========================
[LPM_Jitter]
==========================
NC_MAX_TH = 250
NC_MIN_TH = 0

LC_MAX_TH = 250
LC_MIN_TH = 0


==========================
[OSLoop1]
==========================
NC_VDIFF_MAX_TH = 300
NC_VDIFF_MIN_TH = -300


==========================
[OSLoop2]
==========================
NC_MAX_TH = 300
NC_MIN_TH = -300

==========================
[OSLoop3]
==========================
NC_MAX_TH = 300
NC_MIN_TH = -300


==========================
[OSLoop2Log]
==========================
NC_MAX_TH = 300
NC_MIN_TH = -300
```

Vertical Diff Value

	Ych0	Ych1	Ych2	Ych3	Ych4	Ych5
Xch0	3388	3370	3382	3073	3374	3385
Xch1	3389	3385	3383	2939	3372	3364
Xch2	3389	3381	3388	2822	3373	3353
Xch3	3389	3386	3384	2982	3368	3362
Xch4	3389	3381	3388	2814	3369	3352
Xch5	3389	3381	3383	2926	3364	3360
Xch6	3389	3367	3388	2908	3361	3371

< NC Data >

Xch0 – Xch1

	Ych0	Ych1	Ych2	Ych3	Ych4	Ych5
Xch0	-1	-15	-1	134	2	21
Xch1	0	4	-5	117	-1	11
Xch2	0	-5	4	-160	5	-9
Xch3	0	5	-4	168	-1	10
Xch4	0	0	5	-112	5	-8
Xch5	0	14	-5	18	3	-11

< Vertical Diff Value >

SOLOMON SYSTECH

Thank You

http://www.solomon-systech.com

Email : sales@solomon-systech.com
Phone : 852-2207 1111 Fax : 852-2267 0800