Specifications of the Address checker for MEGA DRIVEs

SEGA ENTERPRISES, LTD.

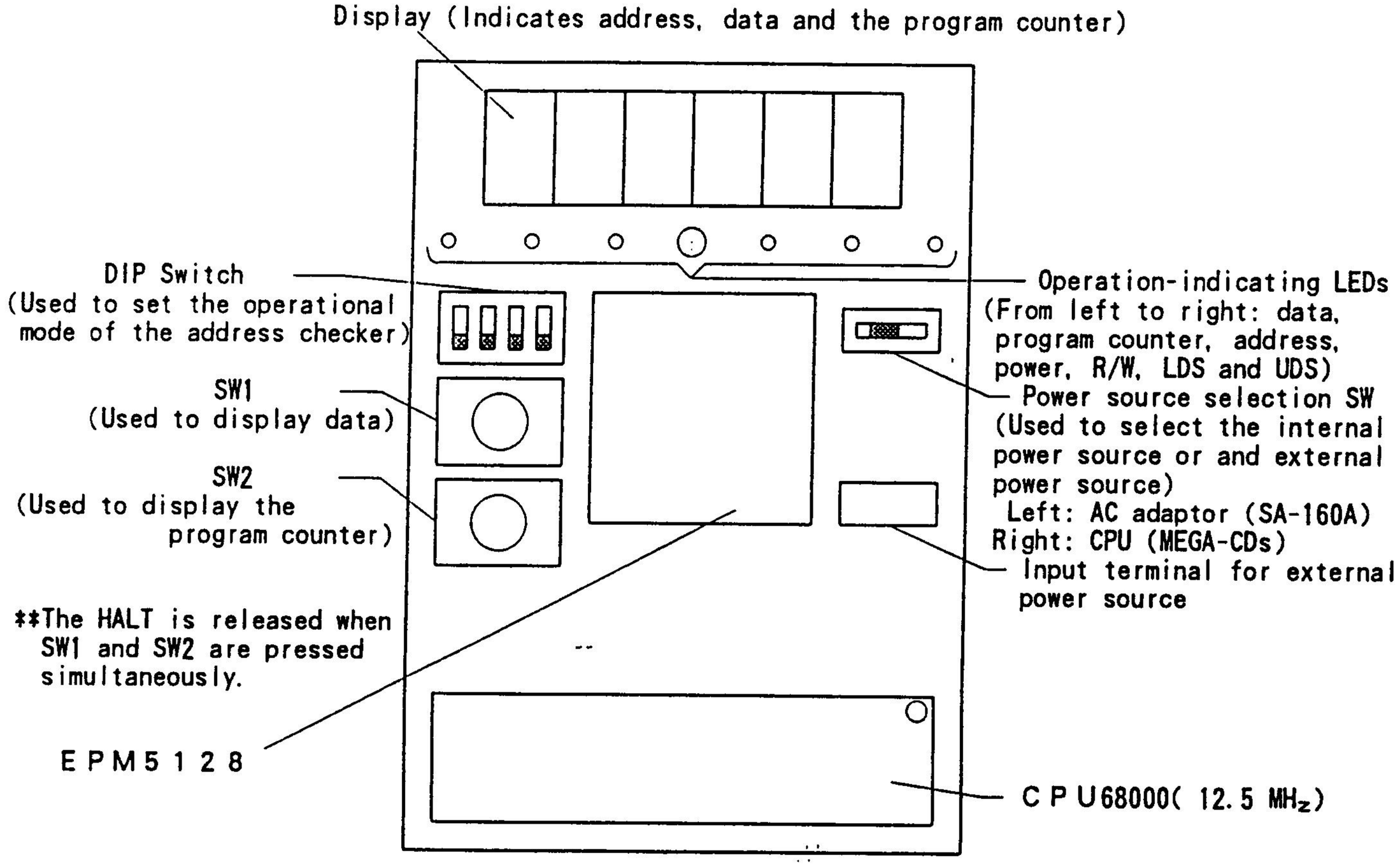
1. Specifications

① To use the MAIN-CPU to access access-prohibit areas, which is necessary during the development of software for the MEGA DRIVE (hereafter referred to as "MD") and MEGA-CDs (hereafter referred to simply as "CD"), and to check whether data written to the various registers is correct.

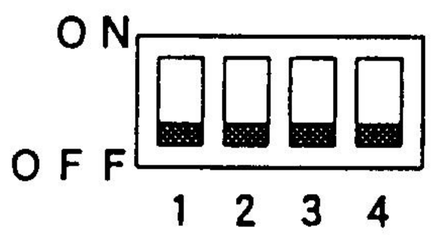
② When an error has occurred, the address, data and program counter in question can be indicated with the LEDs or displayed on the display along with the R/W, UDS and LDS statuses. (CPU operations are stopped during displaying.)

2. The functions of various parts and how to use them

2 – 1 Names of parts



2 – 2 DIP Switches : initial settings and functions



No. 1 ON—— Switches the unit to normal operations from address checker operations without having to plug a different connector in the CPU socket.
OFF— The unit functions as an address checker.
No. 2 ON—— Checks addresses in the MD mode.
OFF— Checks addresses in the CD mode.
No. 3 Not used
No. 4 Not used

2 – 3 How to operate

① Plug the address checker into the MD's CPU socket.

② Run the software. (Confirm that the unit's power LED is on.)

③ If operations are not halted even after all routines in the program have been run, this can be interpreted as indicating that there are no access-inhibits, etc., in the program.

④ If operations are halted, look at the display and LEDs to determine the status of the program.

⑤ Then press SW1 and SW2 simultaneously to resume execution of the program beginning at the address after the one where the error occurred.

Note: To deliberately stop the CPU and display the current address, data and program counter when no error has occurred, move DIP switch 1 to "ON" start the unit and press SW2 after the desired location has been reached. To release this stop, press SW1 and SW2 simultanously and release SW2 first, then SW1.

2 – 4 Interpreting the display and LEDs during operations

① When the main unit and MD(with CD) are turned on, all LEDs and the entire display come on during resetting of the CPU. Once resetting is completed, all LEDs except for the power LED go out.

② When an error occurs, first the error address appears in the display (six digits).
When SW1 is pressed, the error data is displayed as long as the button is held down (lower four digits).
When SW2 is pressed, the value of the program counter when the error occurred is displayed as long as the button is down (six digits).

③ LEDs(These indicate the I/O status of the CPU when an error has occurred.)

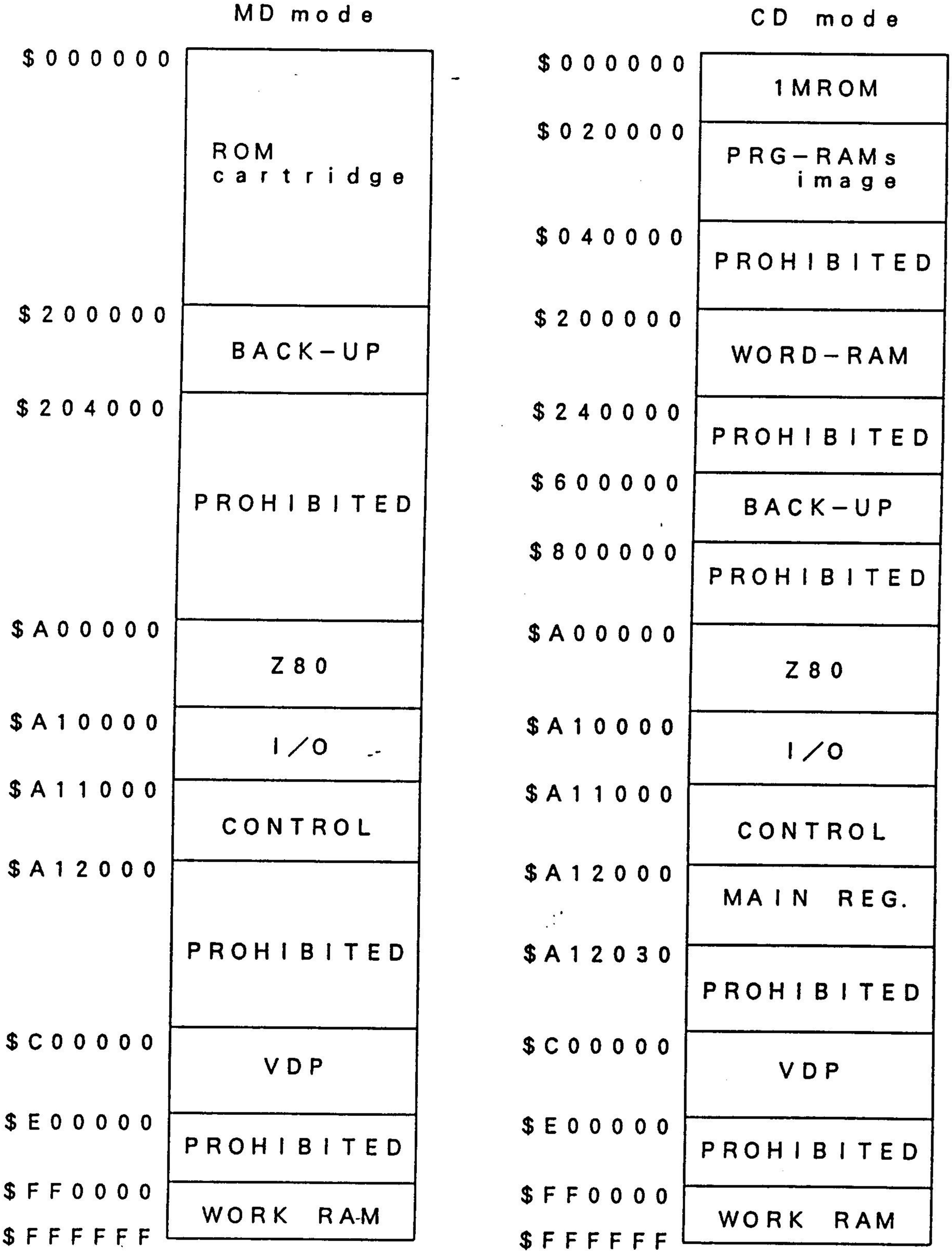
*R/W
On : Indicates that the unit was in the write state.
Off: Indicates that the unit was in the read state.
*LDS
On : Indicates that the lower-half of the data bytes have been input/output.
Off: Indicates that the lower half of the data bytes have not been input/output.
*UDS
On : Indicates that the upper half of the data bytes have been input/output.
Off: Indicates that the upper half of the data bytes have not been input/output.

Note:
* Both LDS and UDS come on during word access.
* As the 68000's running program counter register contains the value of the next operation, the cause of the error may actually be indicated by a value prior to the PC(Program counter) down.
* The address checker only checks mapping and the value entered in the register; it is not a function for finding bugs in the program.

3. Items checked with this device

3 – 1 Mapping

When the MAIN-CPU accesses one of the disabled areas shown below, an error is displayed and the MAIN-CPU is halted.



3 – 2 Register settings

When data other than an accepted value is written to a register, an error is displayed and the MAIN-CPU is halted. Please refer to Attachment 5, "Table of Register Settings," for the settings of the VDP and MAIN-CPU registers.

4. An example of analysis using an ICE

Caution:

As the system is halted when, during analysis with and ICE, and error occurs and the address checker is activated, when reading the history, etc., carry out analysis after first stopping the program and pressing SW1 and SW2 simultaneously to release the stop.

4 – 1 An example of analysis: (1) address error

①Access address (before pressing SW2) : 000004
②PC address (when pressing SW2) : 0018FC
③LED(s) that come on : UDS, LDS, R/W

1 : H D (History dump)

Point	T	Address	St	Data	Mem	Opcode	Operand
0021		0018EC		41F80000	SP	LEA. L	$000000. W, A0
0019		0018F0		43F8FE12	SP	LEA. L	$FFFE12. W, A1
0017		0018F4		217C000980000004	SP	MOVE. L	#$00098000, $0004(A0)
0012	☆1	000004	MW	0009	SD		
0011		000006	MW	8000	SD		
0013	(PC1)	0018FC		237C000980000000	SP	MOVE. L	#$00098000, $0000(A1)

2 : H M (History map)

☆1, 2 : The part where the address checker error occurred.
(PC1, 2): The part of the address checker displayed by the PC.

Point	T	Address	St	Data	Mem		
0021		0018EC	MR	41F8	SP	LEA. L	$000000. W, A0
0020		0018EE	MR	0000	SP		
0019		0018F0	MR	43F8	SP	LEA. L	$FFFE12. W, A1
0018		0018F2	MR	FE12	SP		
0017		0018F4	MR	217C	SP		
0016		0018F6	MR	0009	SP	MOVE. L	#$00098000, $0004(A0)
0015		0018F8	MR	8000	SP		
0014		0018FA	MR	0004	SP		
0013	(PC2)	0018FC	MR	237C	SP	MOVE. L	
0012	☆2	000004	MW	0009	SD		

Cause: Data was written to the ROM area.

In locations ☆1 and 2, "0009" has been written to address $000004 (point 12).

In the MD address map, $000000 to $0FFFFF is ROM area and therefore read-only. Therefore, the address checker caused an error.

When looking at the H M, we see that as for the program, before the error occurred there was a read cycle for reading the instruction in address 18F4.

The H D shows us that while the instruction in $0018FC was the source of the error, in the H M the instruction in $0018FC was read before this instruction was executed (i.e., prefetch). The checker therefore shows this value. This is because until an error occurs the address checker displays as the PC value only the last program address it stored.

3 : Reverse assembly/source program

== Supervisor Program Memory ==

ADDRESS	CODE	Mnemonic		
0018EC	41F80000	LEA. L	$000000. W, A0	;A0←$000000
0018F0	43F8FE12	LEA. L	$FFFE12. W, A1	
0018F4	217C000980000004	MOVE. L	#$00098000, $0004(A0)	;ROM AREA WRITE
0018FC	237C000980000000	MOVE. L	#$00098000, $0000(A1)	

4 – 2 An example of analysis: (2) register setting error

①Access address : C00004
②Data (when SW1 has been pressed) : 8008
③PC (when SW2 has been pressed) : 0601A8
④LEDs that come on : UDS, LDS and R/W

1 : H D (History dump)

Point	T	Address	St	Data	Mem	Opcode	Operand
0014		06019E		207C00C00004	SP	MOVEA.L	#$00C00004,A0
0011		0601A4		30BC8008	SP	MOVE.W	#$8008,(A0)
0008	☆1	C00004	MW	8008	SD		
0009	(PC1)	0601A8		30BC8124	SP	MOVE.W	#$8124,(A0)

☆1,2 : The part where the address checker error occurred.
(PC1,2): The part of the address checker displayed by the PC.

2 : H M (History map)

Point	T	Address	St	Data	Mem	
0014		06019E	MR	207C	SP	MOVEA.L #$00C00004,A0
0013		0601A0	MR	00C0	SP	
0012		0601A2	MR	0004	SP	
0011		0601A4	MR	30BC	SP	MOVE.W #$8008,(A0)
0010		0601A6	MR	8008	SP	
0009	(PC2)	0601A8	MR	30BC	SP	MOVE.W
0008	☆2	C00004	MW	8008	SD	

Cause: Data was written to a disabled register.

In ☆1 and 2, "8008H" has been written in address $C00004 (Point 8).

In the table of MD register settings, the data 8008H listed as not authorized. The address checker therefore caused the error.

3 : Reverse assembly/source program

== Supervisor Program Memory ==

ADDRESS	CODE	Mnemonic	
06019E	207C00C00004	MOVEA.L #$00C00004,A0	;A0←$C00004
0601A4	30BC8008	MOVE.W #$8008,(A0)	;VDP RGSTR SET

4 – 3 : An example of analysis: (3) address setting error

①Access address : C00004
②Address setting data (before pressing SW2) : 4010
③PC (when SW2 is pressed) : 0614A8
④LEDs that come on : UDS, LDS, R/W and 2nd

1 : H D (History dump)

Point		T Address	St	Data	Mem	Opcode	Operand
0016		06149E		23FC40004010 00C00004	SP	MOVE.L	#$40004010, $C00004.L
0010		C00004	MW	4000	SD		
0009	☆1	C00006	MW	4010	SD		
0011	(PC1)	0614A8		2038B3B8	SP	MOVE.L	$FFB3B8.W, D0

☆1, 2 : The part where the address checker error occurred.
(PC1, 2): The part of the address checker displayed by the PC.

2 : H M (History map)

Point		T Address	St	Data	Mem	
0016		06149E	MR	23FC	SP	┐
0015		0614A0	MR	4000	SP	│
0014		0614A2	MR	4010	SP	├ MOVE.L #$40004010, $C00004.L
0013		0614A4	MR	00C0	SP	│
0012		0614A6	MR	0004	SP	┘
0011	(PC2)	0614A8	MR	2038	SP	─ MOVE.L
0010		C00004	MW	4000	SD	
0009	☆2	C00006	MW	4010	SD	

Cause : Setting data for a disabled address has been written.

In ☆1 and 2, "$4010H" has been written to address $C00004. (Point 9)

In the table of MD address register settings, the data therefore caused the error.

3 : DI (reverse assembly/source program)

== Supervisor Program Memory ==

ADDRESS	CODE	Mneumonic
06149E	23FC40004010 00C00004	MOVE.L #$40004010, $C00004.L ;ADRS RGSTR SET

5. Appendix

5 – 1 Table of VDP register settings

5 – 2 Table of VDP address settings

5 – 3 Table of MAIN-CPU register settings

Note:

*5-1. and 5-2 are tables of settings used in both the MD and CD modes; 5-3 shows settings used only in the CD mode. (In the MD mode, the settings in 5-3 are access-prohibit areas.)

*In the MD and CD modes, use of VDP register#23 causes partial change in the DMA's source address.

*In 5-3, "Table of MAIN-CPU register settings," bit F in $A12000(*rest, halt) and bit 2 in $A12002 (*mmode, write protect) were originally zero but changed to "don't care" as the checker would otherwise stop at their location in the CD's BOOT-ROM.

5 — 1　Table of VDP register settings

VDP reg. #0　($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	0	IE0	0	M4	M3	0
BIT	1	0	0	0	0	0	0	0	0	0	0	X	0	1	X	0
HEX	8				0				0, 1				4, 6			

VDP reg. #1　($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	BLNK	IE0	M1	M2	M5	0	0
BIT	1	0	0	0	0	0	0	1	0	X	X	X	X	1	0	0
HEX	8				1				0 — 7				4, C			

VDP reg. #2　($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	SA15	SA14	SA13	0	0	0
BIT	1	0	0	0	0	0	1	0	0	0	X	X	X	0	0	0
HEX	8				2				0 — 3				0, 8			

VDP reg. #3　($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	WD15	WD14	WD13	WD12	WD11	0
BIT	1	0	0	0	0	0	1	1	0	0	X	X	X	X	X	0
HEX	8				3				0 — 3				0, 2, 4, 6, 8, A, C, E			

VDP reg. #4　($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	0	0	0	SB15	SB14	SB13
BIT	1	0	0	0	0	1	0	0	0	0	0	0	0	X	X	X
HEX	8				4				0				0 — 7			

VDP reg. #5　($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	AT15	AT14	AT13	AT12	AT11	AT10	AT9
BIT	1	0	0	0	0	1	0	1	0	X	X	X	X	X	X	X
HEX	8				5				0 — 7				0 — F			

VDP reg. #6 ($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	0	0	0	0	0	0
BIT	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
HEX	8				6				0				0			

VDP reg. #7 ($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	CPT1	CPT0	COL3	COL2	COL1	COL0
BIT	1	0	0	0	0	1	1	1	0	0	X	X	X	X	X	X
HEX	8				7				0 − 3				0 − F			

VDP reg. #8 ($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	0	0	0	0	0	0
BIT	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
HEX	8				8				0				0			

VDP reg. #9 ($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	0	0	0	0	0	0
BIT	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
HEX	8				9				0				0			

VDP reg. #10 ($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	HIT7	HIT6	HIT5	HIT4	HIT3	HIT2	HIT1	HIT0
BIT	1	0	0	0	1	0	1	0	X	X	X	X	X	X	X	X
HEX	8				A				0 − F				0 − F			

VDP reg. #11 ($C00004)

	MSB															LSB
NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	0	0	IE2	VPSC	HPSC	LSCR
BIT	1	0	0	0	1	0	1	1	0	0	0	0	0	X	X	X
HEX	8				B				0				0 − 7			

VDP reg. #12 ($C00004)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	PCH2	0	0	0	S/TE	LSM1	LSM0	RS40
BIT	1	0	0	0	1	1	0	0	A	0	0	0	X	X	X	A
HEX	8				C				A=0 : 0 A=1 : 8				A=0 : even A=1 : odd			

Note: Set PCH2=RS40 (bit pattern)

VDP reg. #13 ($C00004)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	HS15	HS14	HS13	HS12	HS11	HS10
BIT	1	0	0	0	1	1	0	1	0	0	X	X	X	X	X	X
HEX	8				D				0 – 3				0 – F			

VDP reg. #14 ($C00004)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	0	0	0	0	0	0
BIT	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
HEX	8				E				0				0			

VDP reg. #15 ($C00004)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	INC7	INC6	INC5	INC4	INC3	INC2	INC1	INC0
BIT	1	0	0	0	1	1	1	1	X	X	X	X	X	X	X	X
HEX	8				F				0 – F				0 – F			

VDP reg. #16 ($C00004)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	0	0	VSZ1	VSZ0	0	0	HSZ1	HSZ0
BIT	1	0	0	1	0	0	0	0	0	0	X	X	0	0	X	X
HEX	9				0				0 – 3				0 – 3			

VDP reg. #17 ($C00004)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	RIGT	0	0	WHP4	WHP3	WHP2	WHP1	WHP0
BIT	1	0	0	1	0	0	0	1	X	0	0	X	X	X	X	X
HEX	9				1				0, 1, 8, 9				0 – F			

VDP reg. #18 ($C00004)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	DOWN	0	0	WVP4	WVP3	WVP2	WVP1	WVP0
BIT	1	0	0	1	0	0	1	0	X	0	0	X	X	X	X	X
HEX	9				2				0, 1, 8, 9			0 – F				

VDP reg. #19 ($C00004)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	LG7	LG6	LG5	LG4	LG3	LG2	LG1	LG0
BIT	1	0	0	1	0	0	1	1	X	X	X	X	X	X	X	X
HEX	9				3				0 – F				0 – F			

VDP reg. #20 ($C00004)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	LG15	LG14	LG13	LG12	LG11	LG10	LG9	LG8
BIT	1	0	0	1	0	1	0	0	X	X	X	X	X	X	X	X
HEX	9				4				0 – F				0 – F			

VDP reg. #21 ($C00004)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1
BIT	1	0	0	1	0	1	0	1	X	X	X	X	X	X	X	X
HEX	9				5				0 – F				0 – F			

VDP reg. #22 ($C00004)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	SA16	SA15	SA14	SA13	SA12	SA11	SA10	SA9
BIT	1	0	0	1	0	1	1	0	X	X	X	X	X	X	X	X
HEX	9				6				0 – F				0 – F			

VDP reg. #23 ($C00004) DMA *ROM TO VRAM

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	DMD1	DMD0	BA22	BA21	BA20	BA19	BA18	BA17	mode
BIT	1	0	0	1	0	1	1	1	0	0	0	0	X	X	X	X	MD mode
HEX	9				7				0				0 – F				
BIT	1	0	0	1	0	1	1	1	0	0	0	0	0	0	0	X	CD mode
HEX	9				7				0				0, 1				

VDP reg. #23 ($C00004) DMA *BACK-UP TO VRAM

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	DMD1	DMD0	BA22	BA21	BA20	BA19	BA18	BA17	mode
BIT	1	0	0	1	0	1	1	1	0	0	0	1	0	0	0	1	MD mode
HEX	9				7				1				1				
BIT	1	0	0	1	0	1	1	1	0	1	0	1	0	0	0	1	CD mode
HEX	9				7				5				1				

VDP reg. #23 ($C00004) DMA *WORK-RAM TO VRAM

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	DMD1	DMD0	BA22	BA21	BA20	BA19	BA18	BA17
BIT	1	0	0	1	0	1	1	1	0	1	1	1	1	1	1	1
HEX	9				7				7				F			

VDP reg. #23 ($C00004) DMA *FILL VRAM/VRAM COPY

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	DMD1	DMD0	BA22	BA21	BA20	BA19	BA18	BA17
BIT	1	0	0	1	0	1	1	1	1	X	0	0	0	0	0	0
HEX	9				7				8, C				0			

VDP reg. #23 ($C00004) DMA *WORD-RAM TO VRAM (Only CD mode)

MSB LSB

NAME	1	0	0	RG4	RG3	RG2	RG1	RG0	DMD1	DMD0	BA22	BA21	BA20	BA19	BA18	BA17
BIT	1	0	0	1	0	1	1	1	0	0	0	1	0	0	0	X
HEX	9				7				1				0 – 1			

5 — 2 Table of VDP address settings

CRAM READ ($C00004)

MSB upperWORD LSB

NAME	CD1	CD0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
BIT	0	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X
HEX	0				0				0 — 7				0 — F			

MSB lowerWORD LSB

NAME	0	0	0	0	0	0	0	0	CD5	CD4	CD3	CD2	0	A16	A15	A14
BIT	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
HEX	0				0				2				0			

CRAM WRITE ($C00004)

MSB upperWORD LSB

NAME	CD1	CD0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
BIT	1	1	0	0	0	0	0	0	0	X	X	X	X	X	X	X
HEX	C				0				0 — 7				0 — F			

MSB lowerWORD LSB

NAME	0	0	0	0	0	0	0	0	CD5	CD4	CD3	CD2	0	A16	A15	A14
BIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
HEX	0				0				0				0			

CRAM DMA ADDRESS SET ($C00004)

MSB upperWORD LSB

NAME	CD1	CD0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
BIT	1	1	0	0	0	0	0	0	0	X	X	X	X	X	X	X
HEX	C				0				0 — 7				0 — F			

MSB lowerWORD LSB

NAME	0	0	0	0	0	0	0	0	CD5	CD4	CD3	CD2	0	A16	A15	A14
BIT	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
HEX	0				0				8				0			

VRAM READ ($C00004)

MSB — upperWORD — LSB

NAME	CD1	CD0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
BIT	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X	X
HEX	0 − 3				0 − F				0 − F				0 − F			

MSB — lowerWORD — LSB

NAME	0	0	0	0	0	0	0	0	CD5	CD4	CD3	CD2	0	A16	A15	A14
BIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	X
HEX	0				0				0				0 − 3			

VRAM WRITE ($C00004)

MSB — upperWORD — LSB

NAME	CD1	CD0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
BIT	0	1	X	X	X	X	X	X	X	X	X	X	X	X	X	X
HEX	4, 5, 6, 7				0 − F				0 − F				0 − F			

MSB — lowerWORD — LSB

NAME	0	0	0	0	0	0	0	0	CD5	CD4	CD3	CD2	0	A16	A15	A14
BIT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	X
HEX	0				0				0				0 − 3			

VRAM DMA ADDRESS SET ($C00004)

MSB — upperWORD — LSB

NAME	CD1	CD0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
BIT	0	1	X	X	X	X	X	X	X	X	X	X	X	X	X	X
HEX	4, 5, 6, 7				0 − F				0 − F				0 − F			

MSB — lowerWORD — LSB

NAME	0	0	0	0	0	0	0	0	CD5	CD4	CD3	CD2	0	A16	A15	A14
BIT	0	0	0	0	0	0	0	0	1	0	0	0	0	0	X	X
HEX	0				0				8				0 − 3			

V S R A M　R E A D　($C00004)

MSB　　upperWORD　　LSB

NAME	CD1	CD0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
BIT	0	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X
HEX	0				0				0 – 7				0 – F			

MSB　　lowerWORD　　LSB

NAME	0	0	0	0	0	0	0	0	CD5	CD4	CD3	CD2	0	A16	A15	A14
BIT	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
HEX	0				0				1				0			

V S R A M　W R I T E　($C00004)

MSB　　upperWORD　　LSB

NAME	CD1	CD0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
BIT	0	1	0	0	0	0	0	0	0	X	X	X	X	X	X	X
HEX	4				0				0 – 7				0 – F			

MSB　　lowerWORD　　LSB

NAME	0	0	0	0	0	0	0	0	CD5	CD4	CD3	CD2	0	A16	A15	A14
BIT	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
HEX	0				0				1				0			

V S R A M　D M A　A D D R E S S　S E T　($C00004)

MSB　　upperWORD　　LSB

NAME	CD1	CD0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
BIT	0	1	0	0	0	0	0	0	0	X	X	X	X	X	X	X
HEX	4				0				0 – 7				0 – F			

MSB　　lowerWORD　　LSB

NAME	0	0	0	0	0	0	0	0	CD5	CD4	CD3	CD2	0	A16	A15	A14
BIT	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
HEX	0				0				9				0			

VRAM COPY ($C00004)

MSB — upperWORD — LSB

	CD1	CD0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
NAME																
BIT	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X	X
HEX	0 – 3				0 – F				0 – F				0 – F			

MSB — lowerWORD — LSB

	0	0	0	0	0	0	0	0	CD5	CD4	CD3	CD2	0	A16	A15	A14
NAME																
BIT	0	0	0	0	0	0	0	0	1	1	0	0	0	0	X	X
HEX	0				0				C				0 – 3			

5 — 3 Table fo MAIN-CPU register settings

$A12000 *RESET, HALT

MSB LSB

NAME	IEN2	0	0	0	0	0	0	IFL2	0	0	0	0	0	0	SBRQ	SRES
BIT	X	0	0	0	0	0	0	X	0	0	0	0	0	0	X	X
HEX	0, 8				0, 1				0				0 — 3			

$A12002 *MMODE, WRITE PROTECT

MSB LSB

NAME	WP7	WP6	WP5	WP4	WP3	WP2	WP1	WP0	BK1	BK0	0	0	0	MODE	DMNA	RET
BIT	X	X	X	X	X	X	X	X	X	X	0	0	0	X	X	X
HEX	0 — F				0 — F				0, 4, 8, C				0 — 7			

$A12004 *CDC MODE (READ ONLY)

(Error:byte data was read in address $A12005)

MSB LSB

NAME	EDT	DSR	0	0	0	DD2	DD1	DD0	0	0	0	0	0	0	0	0
BIT	X	X	0	0	0	X	X	X	0	0	0	0	0	0	0	0
HEX	0, 4, 8, C				0 — 7				0				0			

$A12006 *H-INT VECTOR (WORD ACCESS)

MSB LSB

NAME	HIBF	HIBE	HIBD	HIBC	HIBB	HIBA	HIB9	HIB8	HIB7	HIB6	HIB5	HIB4	HIB3	HIB2	HIB1	HIB0
BIT	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
HEX	0 — F				0 — F				0 — F				0 — F			

$A12008 *CDC HOST DATA (READ ONLY , WORD ACCESS)

MSB LSB

NAME	HD15	HD14	HD13	HD12	HD11	HD10	HD9	HD8	HD7	HD6	HD5	HD4	HD3	HD2	HD1	HD0
BIT	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
HEX	0 — F				0 — F				0 — F				0 — F			

$A1200C *STOP WATCH (READ ONLY , WORD ACCESS)

MSB LSB

NAME	0	0	0	0	TP11	TP10	TP9	TP8	TP7	TP6	TP5	TP4	TP3	TP2	TP1	TP0
BIT	0	0	0	0	X	X	X	X	X	X	X	X	X	X	X	X
HEX	0				0	0 — F			0 — F				0 — F			

$A1200E *COMMUNICATION FLAG (READ ONLY , WORD ACCESS)

(Error:byte data was written in address $A1200F)

	MSB															LSB
NAME	CFM7	CFM6	CFM5	CFM4	CFM3	CFM2	CFM1	CFM0	CFS7	CFS6	CFS5	CFS4	CFS3	CFS2	CFS1	CFS0
RD BIT	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
RD HEX	0 – F				0 – F				0 – F				0 – F			
WR BIT	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0
WR HEX	0 – F				0 – F				0				0			

$A12010–$A1201E *COMMUNICATION COMMAND (WORD ACCESS)

$A12020–$A1202E *COMMUNICATION STATUS (READ ONLY , WORD ACCESS)