OLED
Character type

Confidential

Version 3.4
Date: 2015/01/27

WINSTAR_OLED Application Note

WINSTAR_OLED Application Note

1 RECORD OF REVISION

Revision Date	Page	Contents	Editor
2010/1/20	-	New Release	
2010/11/17			
2010/12/10			
2011/3/23	9	Add description for Entry Mode Set command	
2011/4/7	9	Modified the description for Clear Display command	
2011/7/20	7	Modified the SPI circuit	
2011/10/04		Modified 4-bit test code	
2012/10/19	10	Modified Instruction Table	
2012/12/05	31~33	Update MPU Interface Timing	Austin
2013/05/02	10、17	Modified description for Function Set command.	Austin
2013/11/07	45~48	Add Appendix Font table	Austin
2014/08/29	6	Add Interface table	Austin
2014/10/16	6	Modify Interface table	Austin
2015/01/06	6,8	Modify description of Jumper function	Brian
2015/01/27	8	Show CS pin location, when use SPI mode	Brian

2. DESCRIPTION

WINSTAR OLED WEH001602AGPP5N0000*0* utilizing CMOS Technology specially designed to display alphanumeric and Japanese kana characters as well as symbols and graphics. It can interface with either 4-bit , 8-bit(8080 ,6800) or 3-lin SPI Microprocessor and display two 8-character lines.

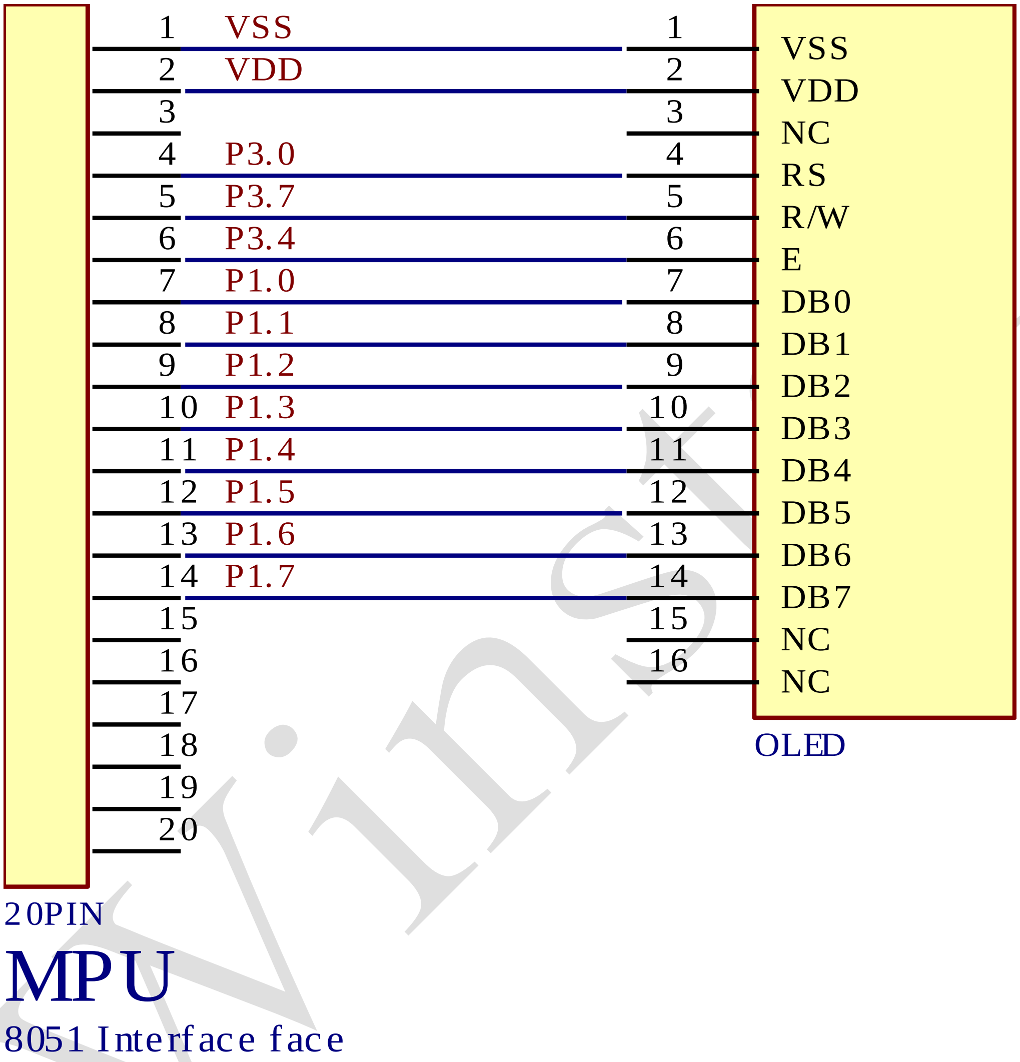
Display RAM, 4 Character Generator, OLED Driver as well as a wide range of instruction functions such as Display Clear, Cursor Home, Display ON/OFF, Cursor ON/OFF, Display Character Blink, Cursor Shift, Display Shift are all incorporated into the module having the highest performance and reliability. Pin assignments and application circuit are optimized for easy PCB layout and cost saving advantages.

2.1 FEATURES

- IC CMOS technology
- Low power consumption
- Microprocessor Interface
  -- High-speed 8-bit parallel bi-directional interface with 6800-series or 8080-series
  -- Serial interface available
- 4-bit or 8-Bit MPU interface
- High speed MPU interface: 2MHz (VDD=5V)
- 128 x 8-bit display RAM (128 characters max.)
- Auto reset function
- 5 x 8 and 5 x 10 dot matrix
- Built-in oscillator with external resistors
- Programmable duty cycle:
  - 1/8 duty: (1 display line, 5 x 8 dots with cursor)
  - 1/11 duty: (1 display line, 5 x 10 dots with cursor)
  - 1/16 duty: (2 display lines, 5 x 8 dots with cursor)
- Build-in selectable three set of character generator ROM (CGROM)
  - English Japanese Character
  - Western European Character-I
  - English Russian Character
  - Western European Character-II
- 64 x 8-bits character generator RAM (CGRAM)
  - Either 8 character fonts (5 x 8 dot matrix)
  - or 4 character fonts (5 x 10 dot matrix)
- 16 common x 100 segment OLED drivers
- Support graphic mode
- Embedded DC-DC voltage converter
- Package：bare die

3 Application Circuit

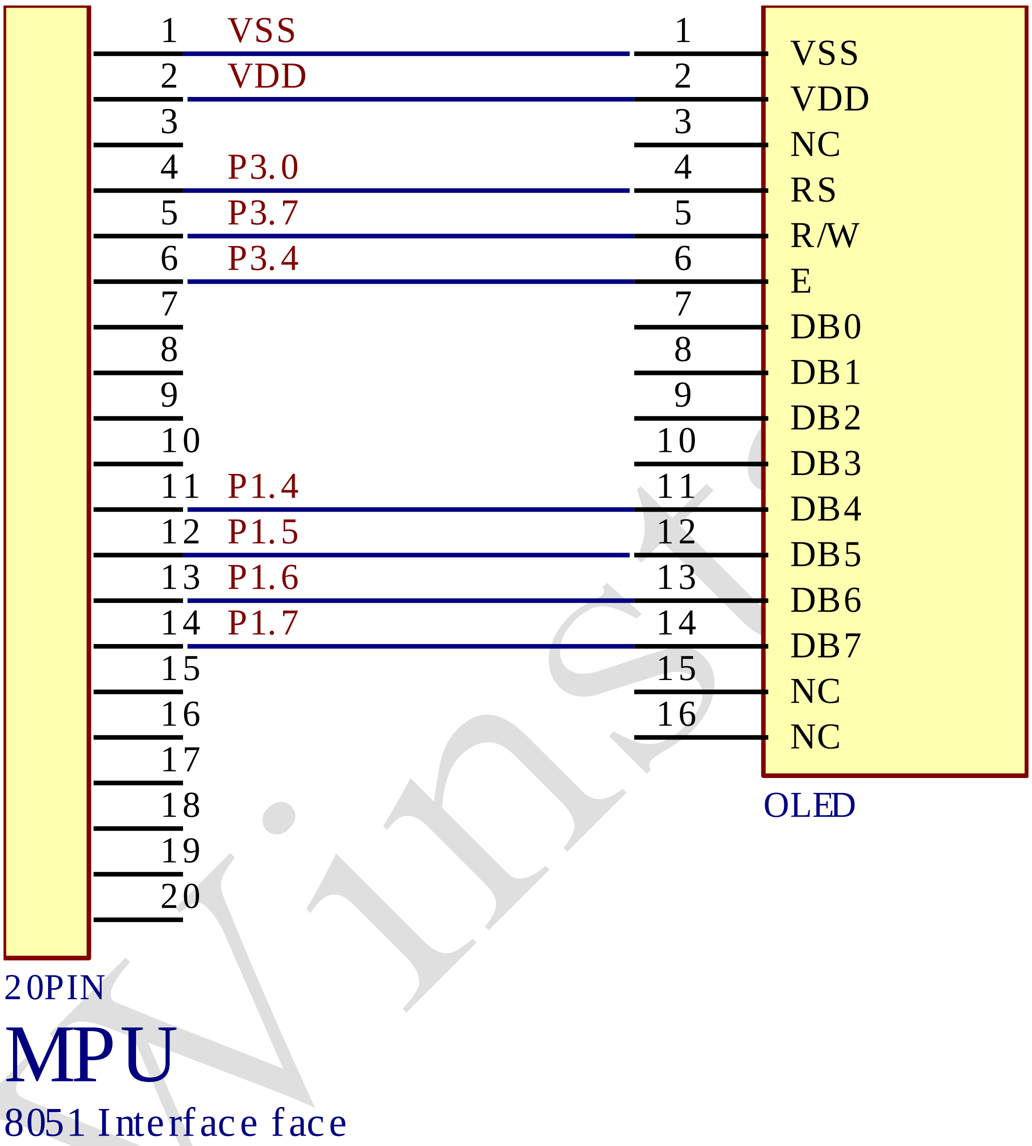
3.1 Parallel 8-bit 68/80 mode



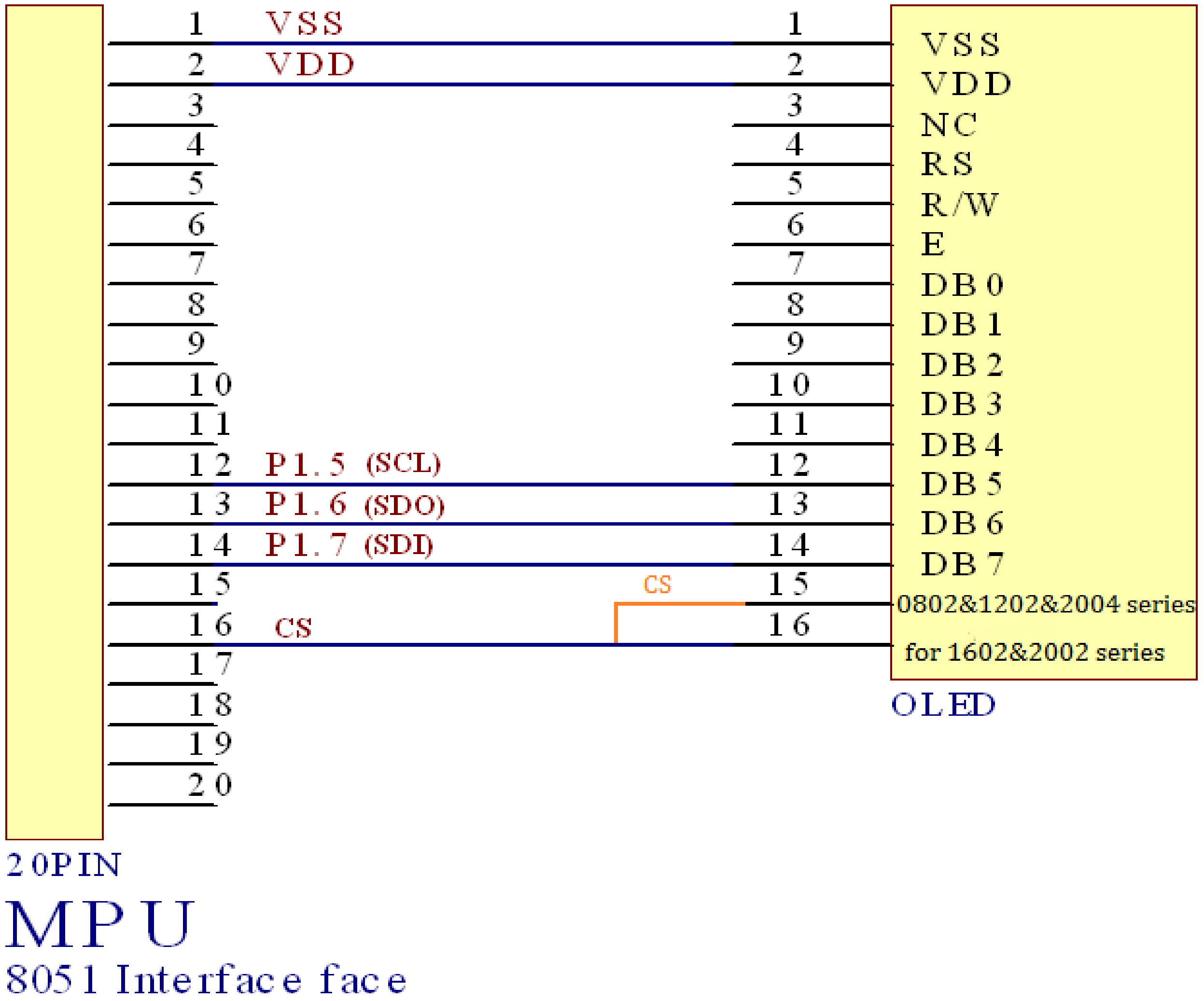
Interface	L_PS_H	L_CS_H	J80_J68	L_SHL_H	JCS	JFG	JHG
6800 Parallel 8-bit (default)	H	L	J68	H	X	short	open
8080 Parallel 8-bit	H	L	J80	H	X		
Serial SPI	L	open	X	H	short		

X : don't care

Parallel 4-bit 68/80 mode

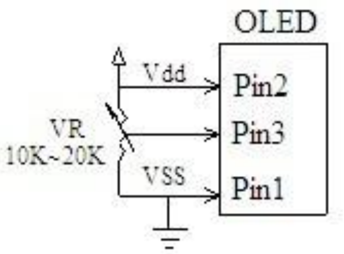


3.2 4-Line series SPI



*CS pin is necessary in SPI mode, Different module has different pin out, please check it before you use.

Jumper JVO (if short) offers contrast adjustment via pin 3:



Normally JVO is open and JV short and contrast cannot be adjusted.

4. INSTRUCTIONS

Instruction Register (IR) and Data Register (DR) are the only registers that can be controlled by the MPU. Prior to the commencement of it internal operation, IST0010 temporarily stores the control information to its Instruction Register (IR) and Data Register (DR) in order to easily facilitate interface with various types of MPU. The internal operations of the IST0010 are determined by the signals (RS, R/WB, DB0 to DB7) that are sent from the MPU. These signals are categorized into 4 instructions types, namely:

1. Function Setting Instructions (i.e. Display, Format, Data Length etc.)
2. Internal RAM Address Setting Instructions
3. Data Transfer with Internal RAM Instructions
4. Miscellaneous Function Instructions

The generally used instructions are those that execute data transfers with the internal RAM. However, when the internal RAM addresses are auto incremented/decremented by 1 after each Data Write, the program load of the MPU is lightened. The Display Shift Instruction can be executed at the same time as the Display Data Write, thereby minimizing system development time with maximum programming efficiency.

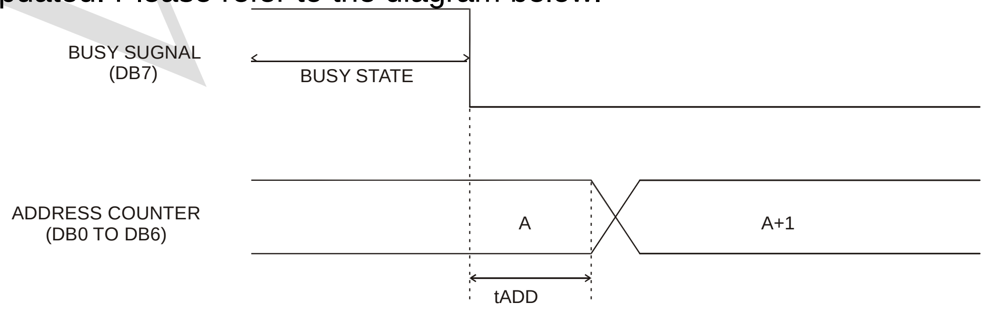
When an instruction is being executed for an internal operation, only the Busy Flag/Address Read Instruction can be performed. The other instructions are not valid. It should be noted that during the execution of an instruction, the Busy Flag is set to "1". The Busy Flag is set to "0" when the instructions are can be accepted and executed. Therefore, the Busy Flag should be checked to make certain that BF = "0" before sending another instruction from the MPU. If not, the time between the first instruction and the next instruction is longer than the time it takes to execute the instruction itself.

4.1 INSTRUCTIONS TABLE

Instruction	Code										Description	Max. Execution Time when fsp or fosc = 250KHz
	RS	R/WB	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Clears entire display.	6.2ms
Return Home	0	0	0	0	0	0	0	0	1	0	Sets DDRAM Address 0 into the Address Counter. Returns shifted display to original position. DDRAM contents remain unchanged. (DB0 is test pin. User should set DB0=0 all the time)	TBD
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Sets cursor move direction and specifies display shift. (These operations are performed during data write and read.) DDRAM address is incremented/decremented by 1. Default value is 0x40	TBD
Display ON/OFF Control	0	0	0	0	0	0	1	D	C	B	Sets entire Display (D) ON/OFF. Sets Cursor (C) ON/OFF. Sets Blinking (B) of Cursor Position Character.	TBD
Cursor/ Display Shift/ Mode/ Pwr	0	0	0	0	0	1	S/C	R/L	0	0	Moves cursor & shifts display without changing DDRAM contents.	TBD
							G/C	PWR	1	1	Sets Graphic/Character Mode Sets internal power on/off	
Function Set	0	0	0	0	1	DL	N	F	FT1	FT0	Sets interface data length (DL). Sets number of display lines (N). Sets Character Font (F). Sets Font Table (FT)	TBD
Set CGRAM Address	0	0	0	1	ACG	ACG	ACG	ACG	ACG	ACG	Sets CGRAM Address. CGRAM data is sent and received after this setting.	TBD
Set DDRAM Address	0	0	1	ADD	ADD	ADD	ADD	ADD	ADD	ADD	Sets DDRAM Address. The DDRAM data Is sent and received after this setting.	TBD
Read Busy Flag & Address	0	1	BF	AC	AC	AC	AC	AC	AC	AC	Reads Busy Flag (BF) indicating that internal operation is being performed. Reads Address Counter contents.	TBD
Write data into the CGRAM or DDRAM	1	0	Write Data								Writes data into the CGRAM or DDRAM	TBD tADD=TBD
Read Data from the CGRAM or DDRAM	1	1	Read Data								Read data from the CGRAM or DDRAM	TBD tADD=TBD

Notes:

1. After the CGRAM/DDRAM Read or Write Instruction has been executed, the RAM Address Counter is incremented or decremented by 1. After the Busy Flag is turned OFF, the RAM Address is updated.
2. I/D=Increment/Decrement Bit
   - I/D="1": Increment
   - I/D="0": Decrement
3. S=Shift Entire Display Control Bit. When S=”0”, shift function disable.
4. BF=Busy Flag
   - BF="1": Internal Operating in Progress
   - BF="0": No Internal Operation is being executed, next instruction can be accepted.
5. R/L=Shift Right/Left
   - R/L="1": Shift to the Right
   - R/L="0": Shift to the Left
6. S/C=Display Shift/Cursor Move
   - S/C="1": Display Shift
   - S/C="0": Cursor Move
7. G/C=Graphic/Character mode selection. G/C=”0”, Character mode is selected. G/C=”1”, Graphic mode is selected.
8. PWR=Internal DCDC on/of control. PWR=”1”, DCDC on. PWR=”0”, DCDC off.
9. DDRAM=Display Data RAM
10. CGRAM=Character Generator RAM
11. ACG=CGRAM Address
12. ADD=Address Counter Address (corresponds to cursor address)
13. AC=Address Counter (used for DDRAM and CGRAM Addresses)
14. F=Character Pattern Mode
    - F="1": 5 x 10 dots
    - F="0": 5 x 8 dots
15. N=Number of Lines Displayed
    - N="1": 2 -Line Display
    - N="0": 1-Line Display
16. tADD is the time period starting when the Busy Flag is turned OFF up to the time the Address Counter is updated. Please refer to the diagram below.



where:

tADD depends on the operation frequency and may be calculated using the following equation
tADD=T.B.D. seconds

4.2 INSTRUCTION DESCRIPTION

4.2.1 CLEAR DISPLAY INSTRUCTION

RS	R/WB	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	0	1

This instruction is used to clear the Display Write Space 20H in all DDRAM Addresses. That is, the character pattern for the Character Code 20H must be a BLANK pattern. It then sets the DDRAM Address 0 into the Address Counter and reverts the display to its original state (if the display has been shifted). The display will be cleared and the cursor or blinking will go to the left edge of the display. If there are 2 lines displayed, the cursor or blinking will go to the first line’s left edge of the display. Under the Entry Mode, this instruction also sets the I/D to 1 (Increment Mode). The S Bit of the Entry Mode does not change.

4.2.2 RETURN HOME INSTRUCTION

RS	R/WB	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	1	*

Note: * = Not Relevant

This instruction is used to set the DDRAM Address 0 into the Address Counter and revert the display to its original status (if the display has been shifted). The DDRAM contents do not change.
The cursor or blinking will go to the left edge of the display. If there are 2 lines displayed, the cursor or blinking will go to the first line's left edge of the display.

4.2.3 ENTRY MODE SET INSTRUCTION

The Entry Mode Set Instruction has two controlling bits: I/D and S. Please refer to the table below.

RS	R/WB	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	1	I/D	S

I/D IS THE INCREMENT/DECREMENT BIT.

When I/D is set to "1", the DDRAM Address is incremented by "1" when a character code is written into or read from the DDRAM. An increment of 1 will move the cursor or blinking one step to the right.

When I/D is set to "0", the DDRAM is decremented by 1 when a character code is written into or read from the DDRAM. A decrement of 1 will move the cursor or blinking one step to the left.

S: SHIFT ENTIRE DISPLAY CONTROL BIT

This bit is used to shift the entire display. When S is set to "1", the entire display is shifted to the right (when I/D ="0") or left (when I/D ="1"). When S is set to "0", the display is not shifted.

Ex1 : I/D=1, S=1

		1	2	3	4	_	Initial display
	1	2	3	4	A	_	Input new character "A"
1	2	3	4	A	B	_	Input new character "B"
2	3	4	A	B	C	_	Input new character "C"
3	4	A	B	C	D	_	Input new character "D"

Ex2 : I/D=0, S=1

1	2	3	4	_			Initial display
	1	2	3	4	A		Input new character "A"
		1	2	3	B	A	Input new character "B"
			1	2	C	B	Input new character "C"
				1	D	C	Input new character "D"

***Note：Default setting is 0 for I/D which means the cursor will move from right to left.

4.2.4 DISPLAY ON/OFF CONTROL INSTRUCTION

The Display On / OFF Instruction is used to turn the display ON or OFF. The controlling bits are D, C and B.

RS	R/WB	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	1	D	C	B

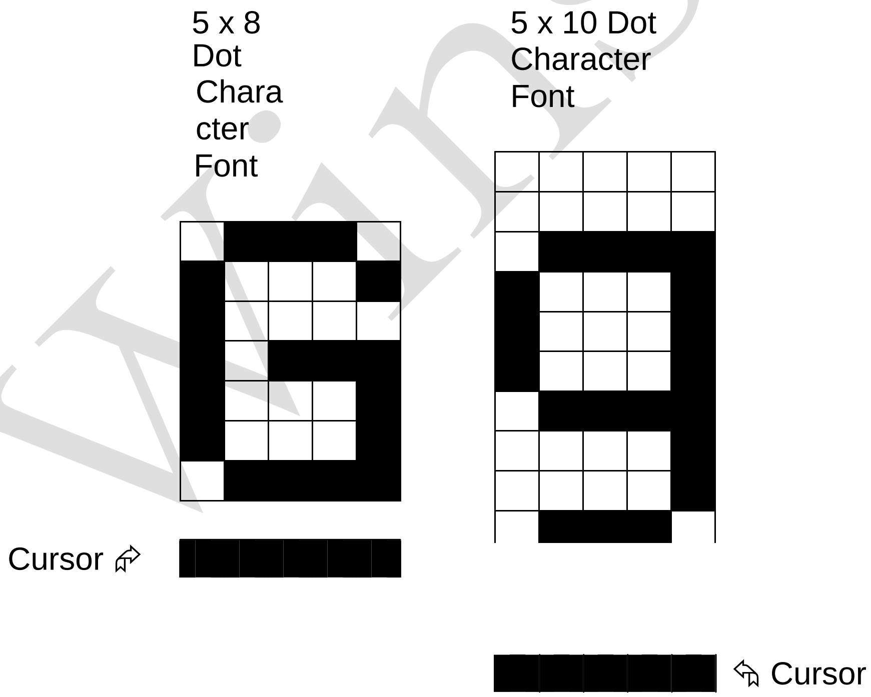
D: DISPLAY ON/OFF BIT

When D is set to "1", the display is turned ON. When D is set to "0", the display is turned OFF and the display data is stored in the DDRAM. The display data can be instantly displayed by setting D to "1".

C: CURSOR DISPLAY CONTROL BIT

When C is set to "1", the cursor is displayed. In a 5 x 8 dot character font, the cursor is displayed via the 5 dots in the 8th line. In a 5 x 10 dot character font, it is displayed via 5 dots in the 11th line.

When C is set to "0", the cursor display is disabled.
During a Display Data Write, the function of the I/D and others will not be altered even if the cursor is not present. Please refer to the figure below.



B: BLINKING CONTROL BIT

When B is set to '1", the character specified by the cursor blinks. The blinking feature is displayed by switching between the blank dots and the displayed character at a speed of 409.6ms intervals when the fcp or fosc is 250kHz. Please refer to the figure below.

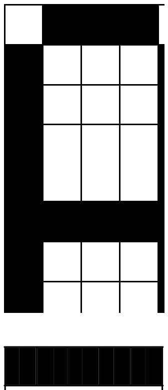


Figure 1

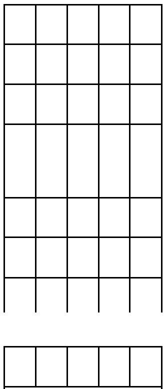


Figure 2

Note: Figures 1 and 2 are alternately displayed

The cursor and the blinking can be set to display at the same time. The blinking frequency depends on the fosc or the reciprocal of fcp.

To illustrate, when fosc=TBD Hz, then, the blinking frequency=409.6 x 250/270=379.2ms

4.2.5 CURSOR/DISPLAY SHIFT INSTRUCTION

This instruction is used to shift the cursor or display position to the left or right without writing or reading the Display Data. This function is used to correct or search the display. Please refer to the table below.

RS	R/WB	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	1	S/C	R/L	0	0
0	0	0	0	0	1	G/C	PWR	1	1

S/C	R/L	Shift Function
0	0	Shifts the cursor position to the left. (AC is decremented by 1).
0	1	Shifts cursor position to the right. (AC incremented by 1).
1	0	Shifts entire display to the left. The cursor follows the display shift.
1	1	Shifts the entire display to the right. The cursor follows the display shift.

In a 2-line Display, the cursor moves to the second line when it passes the 40th digit of the first line.
The first and second line displays will shift at the same time.

When the displayed data is shifted repeatedly, each line moves only horizontally. The second line display does not shift into the first line position.

The Address Counter (AC) contents will not change if the only action performed is a Display Shift.

G/C: GRAPHIC MODE / CHARACTER MODE SELECTION

This bit is used to select the display mode for further process.
When G/C = 1, the *GRAPHIC MODE* will be selected.
When G/C = 0, the *CHARACTER MODE* will be selected.

PWR: ENABLE/DISABLE INTERNAL POWER

This bit is used to turn ON or turn OFF the internal power.
When PWR = 1, the internal power is turned ON.
When PWR = 0, the internal power is turned OFF.

4.2.6 FUNCTION SET INSTRUCTION

The Function Set Instruction has three controlling 3 bits, namely: DL, N and F. Please refer to the table below.

RS	R/WB	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	DL	N	F	FT1	FT0

DL: INTERFACE DATA LENGTH CONTROL BIT

This is used to set the interface data length. When DL is set to "1", the data is sent or received in 8-bit length via the DB0 to DB7 (for an 8-Bit Data Transfer). When DL is set to "0", the data is sent or received in 4-bit length via DB4 to DB7 (for a 4-Bit Data Transfer). When the 4-bit data length is selected, the data must be sent or received twice.

N: NUMBER OF **DISPLAY LINE**

This is used to set the number of display lines. When N="1", the 2-line display is selected. When N is set to "0", the 1-line display is selected.

F: CHARACTER FONT SET

This is used to set the character font set. When F is set to "0", the 5 x 8 dot character font is selected. When F is set to "1", the 5 x 10 dot character font is selected.

It must be noted that the character font setting must be performed at the head of the program before executing any instructions other than the Busy Flag and Address Instruction. Otherwise, the Function Set Instruction cannot be executed unless the interface data length is changed.

FT1, FT0: FONT TABLE SELECTION

These two bits are used to select one font table out of the three for further process.
When (FT1, FT0) = (0, 0), the *ENGLISH_JAPANESE CHARACTER FONT TABLE* will be selected.
(FT1, FT0) = (0, 1), the *WESTERN EUROPEAN CHARACTER FONT TABLE-I* will be selected.
(FT1, FT0) = (1, 0), the *ENGLISH_RUSSIAN CHARACTER FONT TABLE* will be selected.
(FT1, FT0) = (1, 1), the *WESTERN EUROPEAN CHARACTER FONT TABLE-II* will be selected.

4.2.7 SET CGRAM ADDRESS INSTRUCTION

This instruction is used to set the CGRAM Address binary AAAAAA into the Address Counter. Data is then written to or read from the MPU for CGRAM.

RS	R/WB	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	ACG	ACG	ACG	ACG	ACG	ACG

Note: ACG is the CGRAM Address

4.2.8 SET DDRAM ADDRESS INSTRUCTION

This instruction is used to set the DDRAM Address binary AAAAAAA into the Address Counter. The data is written to or read from the MPU for the DDRAM. If 1-line display is selected (N="0"), then AAAAAAA can be 00H to 4FH. When the 2-line display is selected, then AAAAAAA can be 00H to 27H for the first line and 40H to 67H for the second line.

RS	R/WB	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	ADD	ADD	ADD	ADD	ADD	ADD	ADD

Note: ADD = DDRAM Address

CHARACTER MODE ADDRESSING

WIN0010 provides two kind of character mode. User can fill in 128 characters data (N=0, one line) or 64 characters data per line (N=1, two line) in embedded RAM to display graphic. Character mode address can be controlled by DDRAM address instruction.

Address Format	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
CA (Character Address)	1	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0

(1)1-Line condition (N=0)

1	2	3	4	……	……	125	126	127	128
CA=10000000	CA=10000001	CA=10000010	CA=10000011			CA=11111100	CA=11111101	CA=11111110	CA=11111111

(2)2-Line condition (N=1)

1	2	3	4	……	……	61	62	63	64
CA=10000000	CA=10000001	CA=10000010	CA=10000011			CA=10111100	CA=10111101	CA=10111110	CA=10111111
CA=11000000	CA=11000001	CA=11000010	CA=11000011			CA=11111100	CA=11111101	CA=11111110	CA=11111111

GRAPHIC MODE ADDRESSING

WIN0010 provides not only character mode but also graphic mode. User can fill in 100x16 data in embedded RAM to display graphic. Graphic mode addressing is different from character mode.

Use DDRAM address instruction to set X-axis address of Graphic mode and CGRAM address instruction to set Y-axis of Graphic mode.

Address Format	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
GXA (Graphic X-axis Address)	1	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0
GYA (Graphic Y-axis Address)	0	1	0	0	0	0	0	CGA0

	1	2	3	4	……	……	97	98	99	100
CGA0=0	GXA=10000000 GYA=01000000	GXA=10000001 GYA=01000000	GXA=10000010 GYA=01000000	GXA=10000011 GYA=01000000	D0		GXA=11100000 GYA=01000000	GXA=11100001 GYA=01000000	GXA=11100010 GYA=01000000	GXA=11100011 GYA=01000000
					D1					
					D2					
					D3					
					D4					
					D5					
					D6					
					D7					
CGA0=1	GXA=10000000 GYA=01000001	GXA=10000001 GYA=01000001	GXA=10000010 GYA=01000001	GXA=10000011 GYA=01000001	D0		GXA=11100000 GYA=01000001	GXA=11100001 GYA=01000001	GXA=11100010 GYA=01000001	GXA=11100011 GYA=01000001
					D1					
					D2					
					D3					
					D4					
					D5					
					D6					
					D7					

4.2.9 READ BUSY FLAG AND ADDRESS INSTRUCTION

This instruction is used to read the Busy Flag (BF) to indicate if IST0010 is internally operating on a previously received instruction. If BF is set to "1", then the internal operation is in progress and the next instruction will not be accepted. If the BF is set to "0", then the previously received instruction has been executed and the next instruction can be accepted and processed. It is important to check the BF status before proceeding to the next write operation. The value of the Address Counter in binary AAAAAAA is simultaneously read out. This Address Counter is used by both the CGRAM and the DDRAM and its value is determined by the previous instruction. The contents of the address are the same as for the instructions -- Set CGRAM Address and Set DDRAM Address.

RS	**R/WB**	**DB7**	**DB6**	**DB5**	**DB4**	**DB3**	**DB2**	**DB1**	**DB0**
0	1	BF	AC	AC	AC	AC	AC	AC	AC

Notes:
1. BF=Busy Flag
2. AC=Address Counter

4.2.10 WRITE DATA TO CGRAM / DDRAM INSTRUCTION

This instruction writes 8-bit binary data -- DDDDDDDD to the CGRAM or the DDRAM. The previous CGRAM or DDRAM Address setting determines whether a data is to be written into the CGRAM or the DDRAM. After the write process is completed, the address is automatically incremented or decremented by 1 in accordance with the Entry Mode instruction. It must be noted that the Entry Mode instruction also determines the Display Shift.

RS	R/WB	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	0	D	D	D	D	D	D	D	D

4.2.11 READ DATA FROM THE CGRAM OR DDRAM INSTRUCTION

This instruction reads the 8-bit binary data -- DDDDDDDD from the CGRAM or the DDRAM. The Set CGRAM Address or Set DDRAM Address Set Instruction must be executed before this instruction can be performed, otherwise, the first Read Data will not be valid.

RS	R/WB	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	1	D	D	D	D	D	D	D	D

When the Read Instruction is executed in series, the next address data is normally read from the Second Read. There is no need for the Address Set Instruction to be performed before this Read instruction when using the Cursor Shift Instruction to shift the cursor (Reading the DDRAM). The Cursor Shift Instruction has the same operation as that of the Set the DDRAM Address Instruction.

After a Read instruction has been executed, the Entry Mode is automatically incremented or decremented by 1. It must be noted that regardless of the Entry Mode, the Display Shift is not executed.

After the Write instruction to either the CGRAM or DDRAM has been performed, the Address Counter is automatically increased or decreased by 1. The RAM data selected by the Address Counter cannot be read out at this time even if the Read Instructions are executed. Therefore, in order to correctly read the data, the following procedure has suggested:

1. Execute the Address Set or Cursor Shift (only with DDRAM) Instruction
2. Just before reading the desired data, execute the Read Instruction from the second time the Read Instruction has been sent.

5. CHARACTER GENERATOR RAM (CGRAM)

The Character Generator RAM (CGRAM) is used to generate either 5 x 8 dot or 5 x 10 dot character patterns. It can generate eight 5 x 8 dot character patterns or four 5 x 10 dot character patterns. The character patterns generated by the CGRAM can be rewritten. User-defined character patterns for the CGRAM are supported.

RELATIONSHIP BETWEEN CGRAM ADDRESS, DDRAM CHARACTER CODE AND CGRAM CHARACTER PATTERNS (FOR 5 X 8 DOT CHARACTER PATTERN)

Character Codes (DDRAM Data)									CGRAM Address							Character Patterns (CGRAM Data)								
7	6	5	4	3	2	1	0		5	4	3	2	1	0		7	6	5	4	3	2	1	0	
High						Low			High				Low			High						Low		
0	0	0	0	*	0	0	0		0	0	0	0	0	0		*	*	*	1	1	1	1	0	Character pattern 1
												0	0	1		*	*	*	1	0	0	0	1	
												0	1	0		*	*	*	1	0	0	0	1	
												0	1	1		*	*	*	1	1	1	1	0	
												1	0	0		*	*	*	1	0	1	0	0	
												1	0	1		*	*	*	1	0	0	1	0	
												1	1	0		*	*	*	1	0	0	0	1	
												1	1	1		*	*	*	0	0	0	0	0	Cursor Position
0	0	0	0	*	0	0	1		0	0	1	0	0	0		*	*	*	1	0	0	0	1	Character pattern 2
												0	0	1		*	*	*	0	1	0	1	0	
												0	1	0		*	*	*	1	1	1	1	1	
												0	1	1		*	*	*	0	0	1	0	0	
												1	0	0		*	*	*	1	1	1	1	1	
												1	0	1		*	*	*	0	0	1	0	0	
												1	1	0		*	*	*	0	0	1	0	0	
												1	1	1		*	*	*	0	0	0	0	0	Cursor position
0	0	0	0	*	. . . .	. . . .	. . . .		. . . .	. . . .	. . . .	. . . .	. . . .	. . . .		*	*	*	. . . .	. . . .	. . . .	. . . .	. . . .	Character pattern 3~7
0	0	0	0	*	1	1	1		1	1	1	0	0	0		*	*	*	0	0	0	0	0	Character pattern 8
												0	0	1		*	*	*	0	1	0	1	0	
												0	1	0		*	*	*	0	0	0	0	0	
												0	1	1		*	*	*	0	0	0	0	0	
												1	0	0		*	*	*	1	0	0	0	1	
												1	0	1		*	*	*	0	1	1	1	0	
												1	1	0		*	*	*	0	0	1	0	0	
												1	1	1		*	*	*	0	0	0	0	0	Cursor position

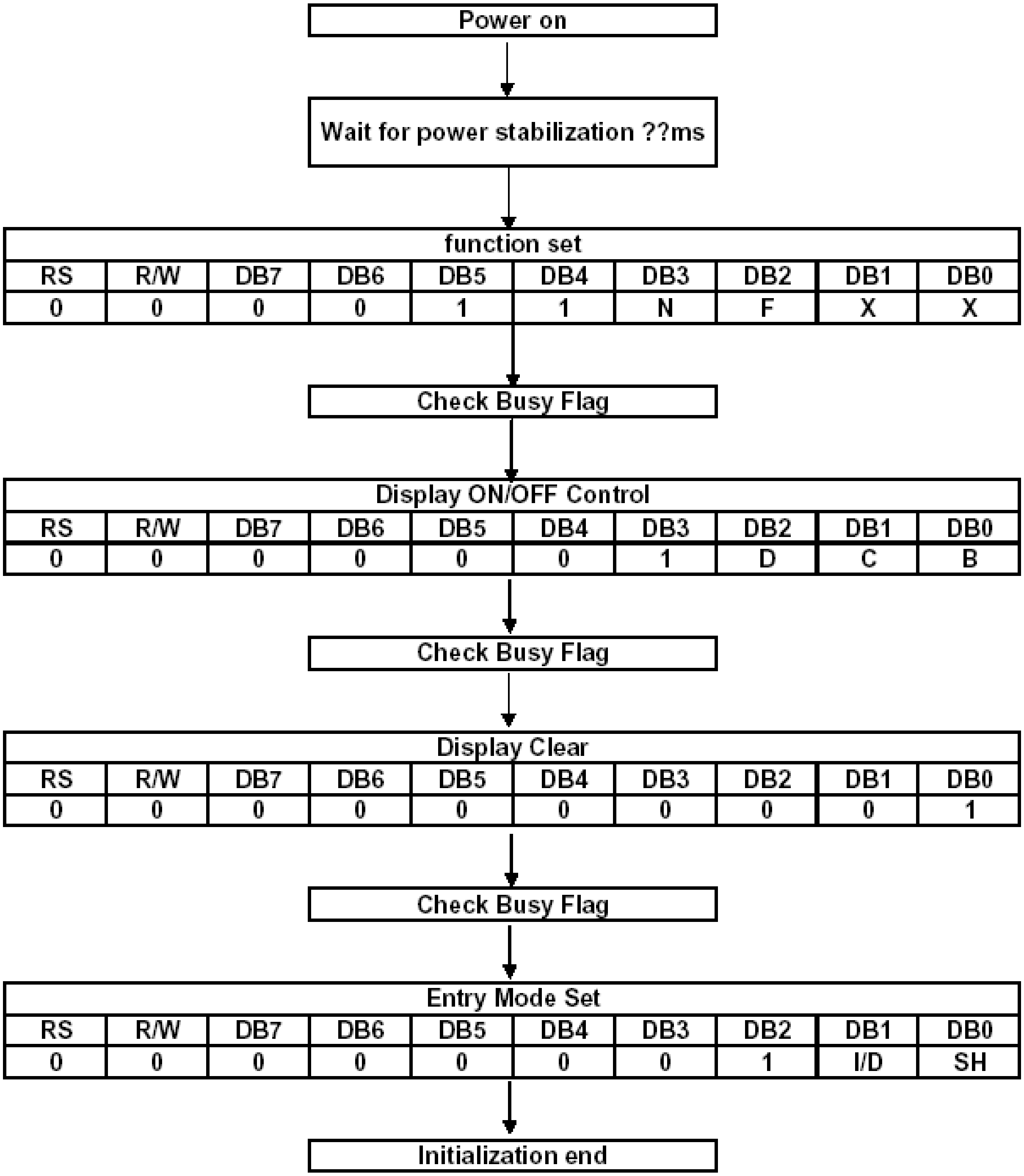
Notes:

1. * = Not Relevant
2. The character pattern row positions correspond to the CGRAM data bits -- 0 to 4, where bit 4 is in the left position.
3. Character Code Bits 0 to 2 correspond to the CGRAM Address Bits 3 to 5 (3 bits: 8 types)
4. If the CGRAM Data is set to "1", then the selection is displayed. If the CGRAM is set to "0", there no selection is made.
5. The CGRAM Address Bits 0 to 2 are used to define the character pattern line position. The 8th line is the cursor position and its display is formed by the logical OR with the cursor. The 8th line CGRAM data bits 0 to 4 must be set to "0". If any of the 8th line CGRAM data bits 0 to 4 is set to "1", the corresponding display location will light up regardless of the cursor position.
6. When the Character Code Bits 4 to 7 are set to "0", then the CGRAM Character Pattern is selected.

It must be noted that Character Code Bit 3 is not relevant and will not have any effect on the character display. Because of this, the first Character Pattern shown above (R) can be displayed when the Character Code is 00H or 08H.

6 Initialization by Instruction

6.1 8-bit mode



Power on

Wait for power stabilization ??ms

function set

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	1	N	F	X	X

Check Busy Flag

Display ON/OFF Control

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	1	D	C	B

Check Busy Flag

Display Clear

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	0	1

Check Busy Flag

Entry Mode Set

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	1	I/D	SH

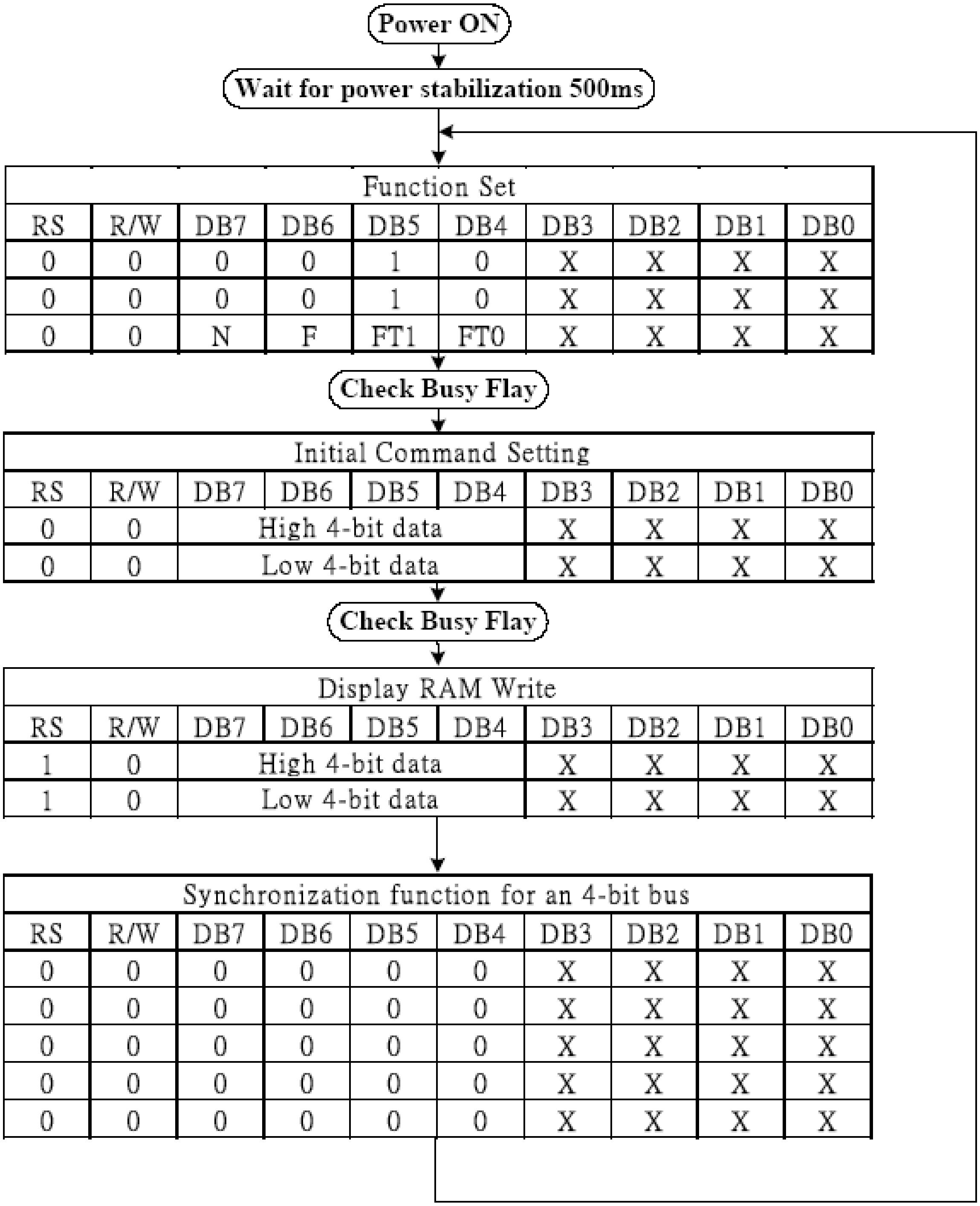
Initialization end

6.2 4-bit mode

Note: Repeated procedures for an 4-bit bus interface

Noise causing transfer mismatch between the four upper and lower bits can be corrected by a reset triggered by consecutively writing a “0000” instruction five times. The next transfer starts from the lower four bits and then first instruction “Function set” can be executed normally.

Please insert the synchronization function in the end of procedures. The repeated procedures are show as follows：



Power ON

Wait for power stabilization 500ms

Function Set

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	0	X	X	X	X
0	0	0	0	1	0	X	X	X	X
0	0	N	F	FT1	FT0	X	X	X	X

Check Busy Flay

Initial Command Setting

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	High 4-bit data				X	X	X	X
0	0	Low 4-bit data				X	X	X	X

Check Busy Flay

Display RAM Write

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	0	High 4-bit data				X	X	X	X
1	0	Low 4-bit data				X	X	X	X

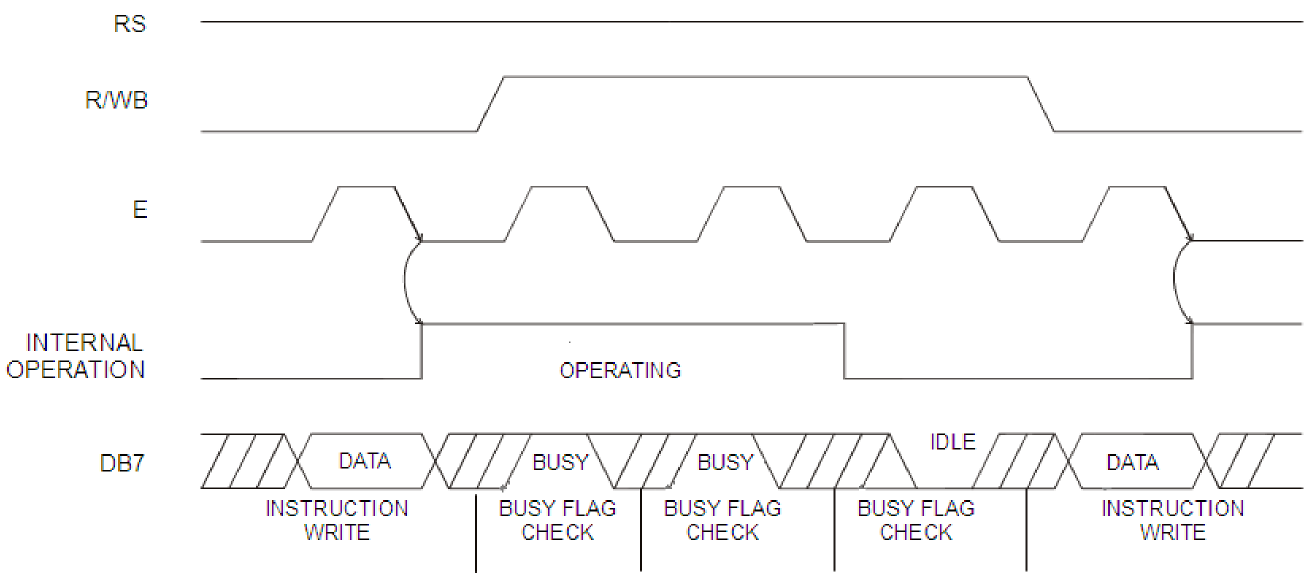
Synchronization function for an 4-bit bus

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	X	X	X	X
0	0	0	0	0	0	X	X	X	X
0	0	0	0	0	0	X	X	X	X
0	0	0	0	0	0	X	X	X	X
0	0	0	0	0	0	X	X	X	X

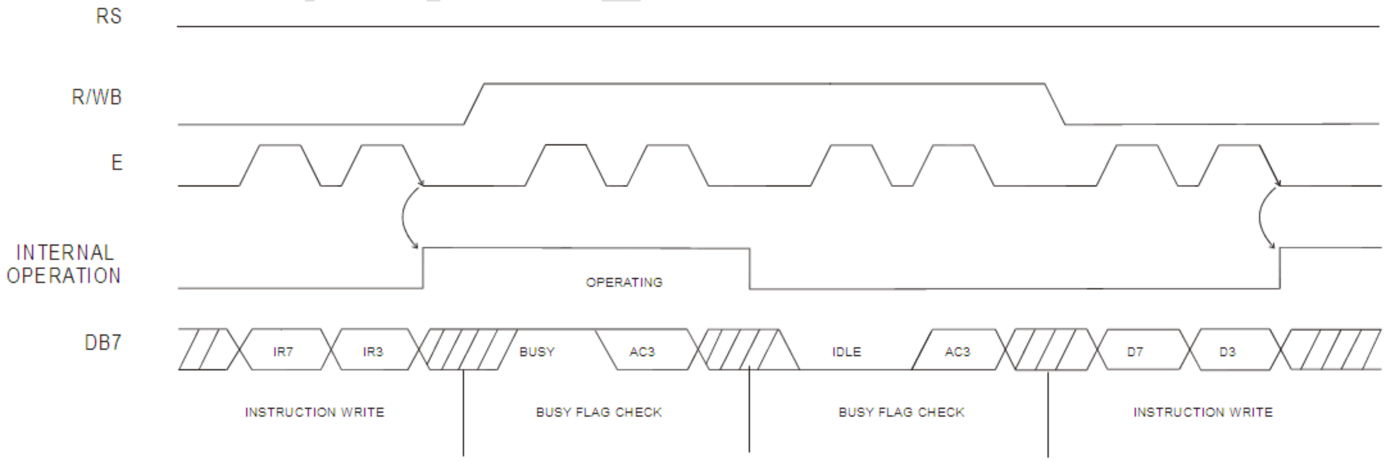
7 MCU Interface

7.1 6800 series

(a)8-bit

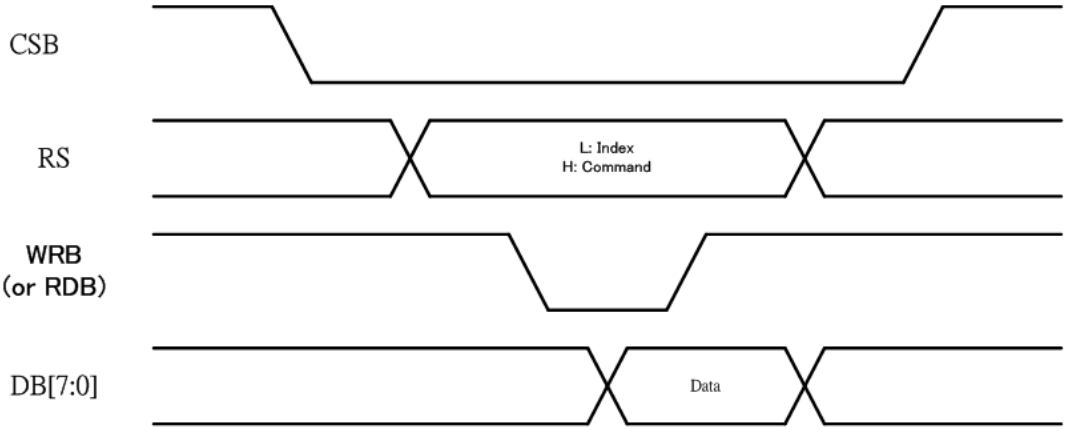


(b)4-bit

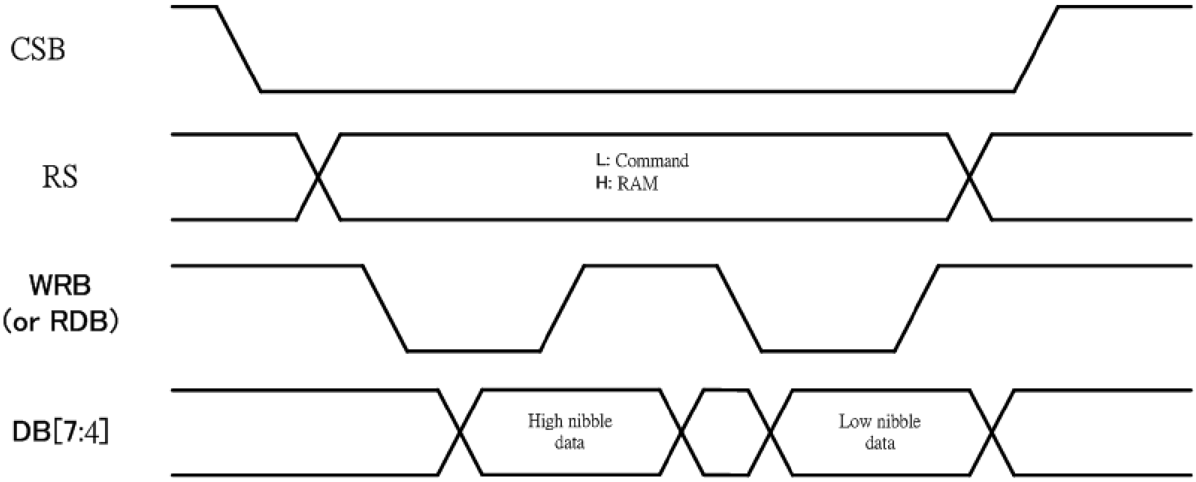


7.2 8080 series

(a)8-bit

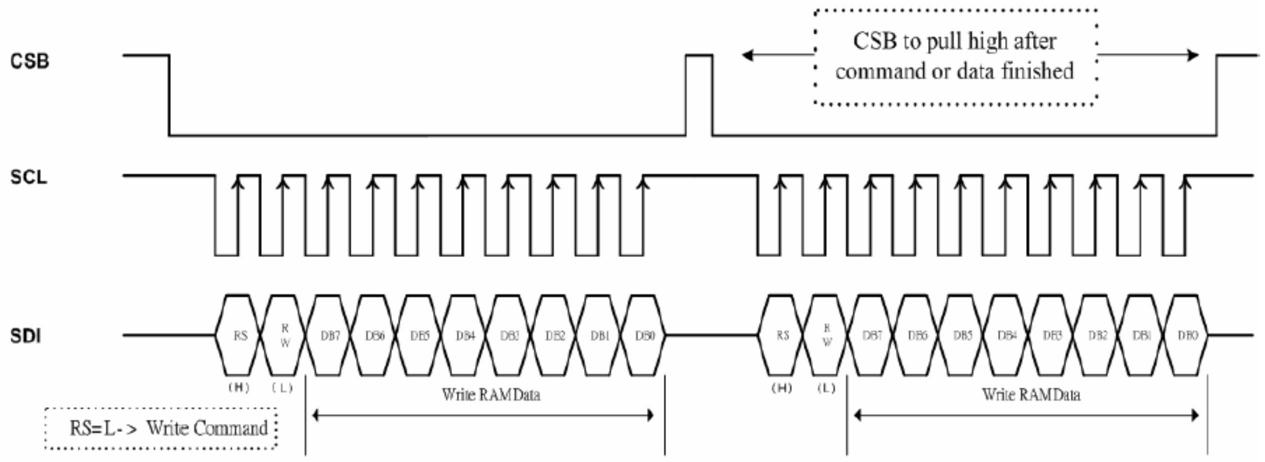


(b)4-bit

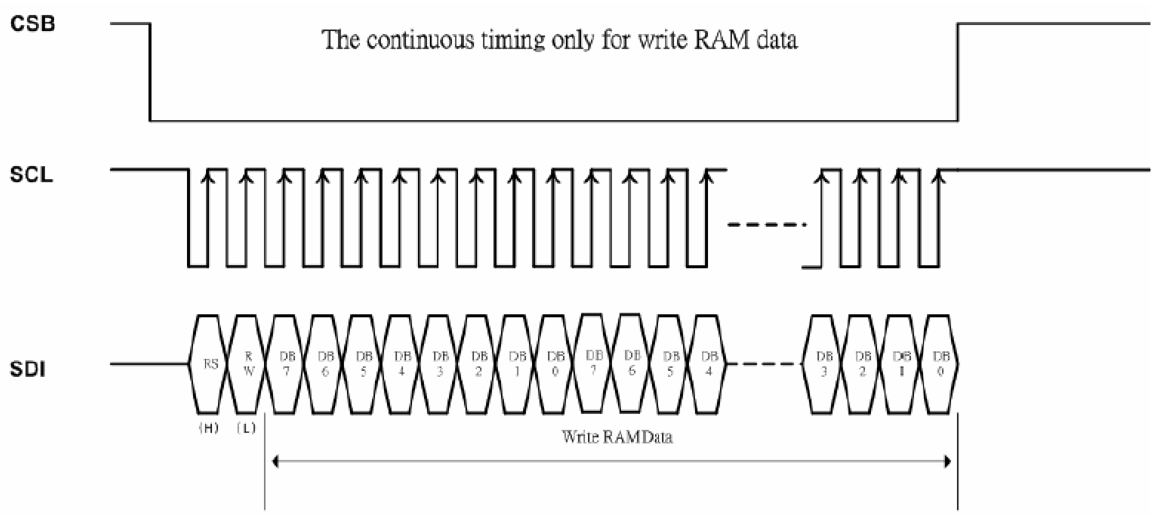


7.3 SPI-4

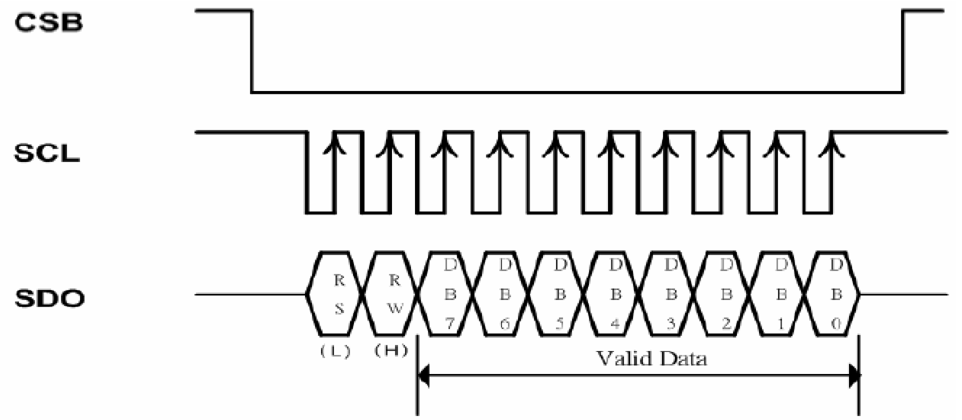
(a) Command/RAM Data Write(Single)



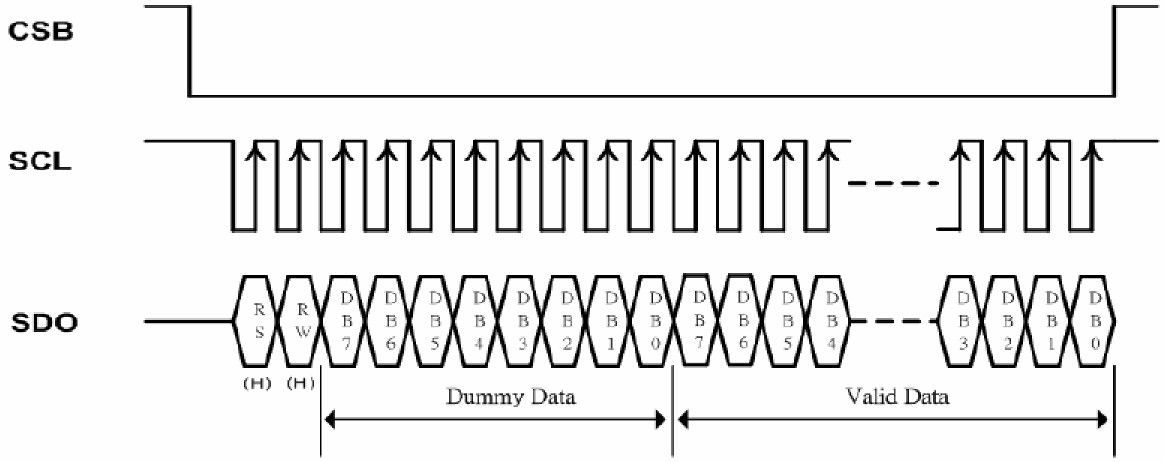
(b) RAM Data Continuous Write



(c) Command Read



(d) RAM Data Read



8 MCU Interface Timing

8.1 Read / Write Characteristics (6800-series Microprocessor)

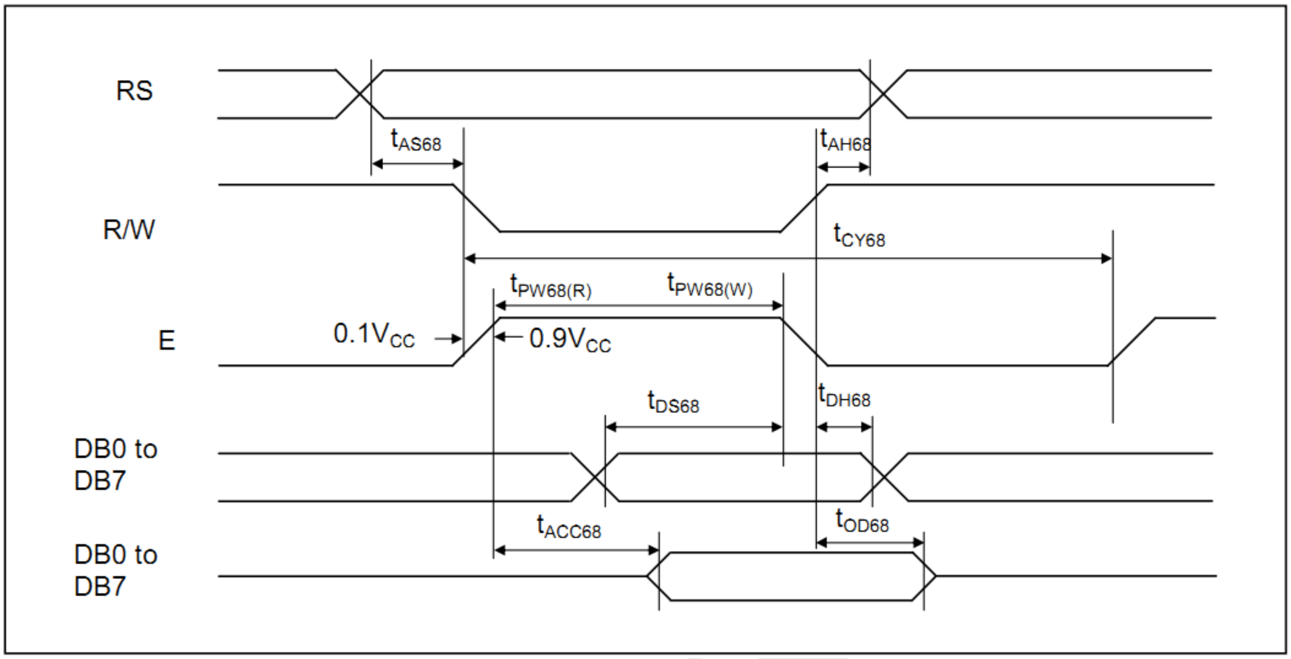


Figure 1. Read / Write Characteristics (6800-series MPU)

* Our standard set is 6800 interface

(Vcc = 4.5 to 5.5V, Ta = -40 to +85℃)

Item	Signal	Symbol	Min.	Typ.	Max.	Unit	Remark
Address setup time Address hold time	RS	**tAS68** **tAH68**	20 0	-	-	ns	
System cycle time		**tCY68**	500	-	-	ns	
Pulse width (E)	E_RDB	**tPW68**(W)	250	-	-	ns	
Pulse width (E)	E_RDB	**tPW68**(R)	250	-	-	ns	
Data setup time Data hold time	DB7 to DB0	**tDS68** **tDH68**	40 20	-	-	ns	
Read access time Output disable time		**tACC68** **tOD68**	- 10	-	180	ns	CL = 100pF

Note:

The name deferent between 8080 and 6800

Pin number	8080 series	6800 series
Pin 4	RS (A0)	RS (A0)
Pin 5	Write	RW
Pin 6	Read	E

8.2 Read / Write Characteristics (8080-series Microprocessor)

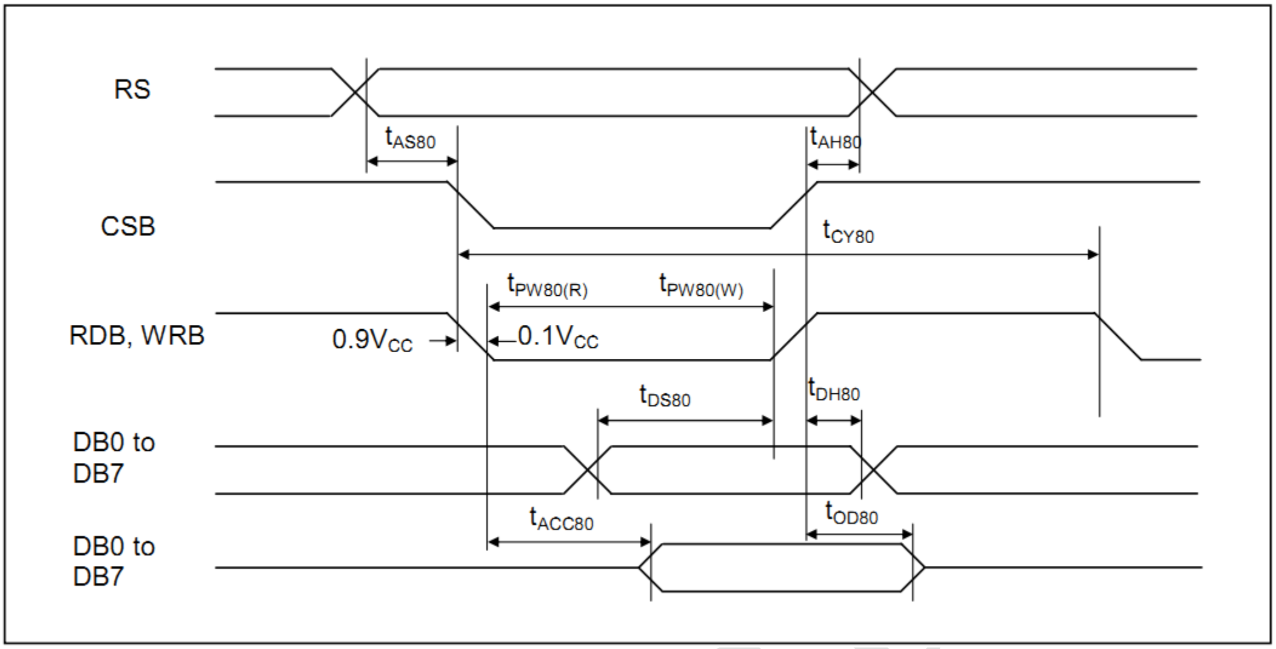


Figure 2. Read / Write Characteristics (8080-series MPU)

VCC = 4.5 to 5.5V, Ta = -40 to +85℃)

Item	Signal	Symbol	Min.	Typ.	Max.	Unit	Remark
Address setup time Address hold time	RS	**tAS80** **tAH80**	20 0	-	-	ns	
System cycle time		**tCY80**	500	-	-	ns	
Pulse width (WRB)	RW_WRB	**tPW80(W)**	250	-	-	ns	
Pulse width (RDB)	E_RDB	**tPW80(R)**	250	-	-	ns	
Data setup time Data hold time	DB7 to DB0	**tDS80** **tDH80**	40 20	-	-	ns	
Read access time Output disable time		**tACC80** **tOD80**	- 10	-	180	ns	CL = 100pF

8.3 Serial Interface Characteristics

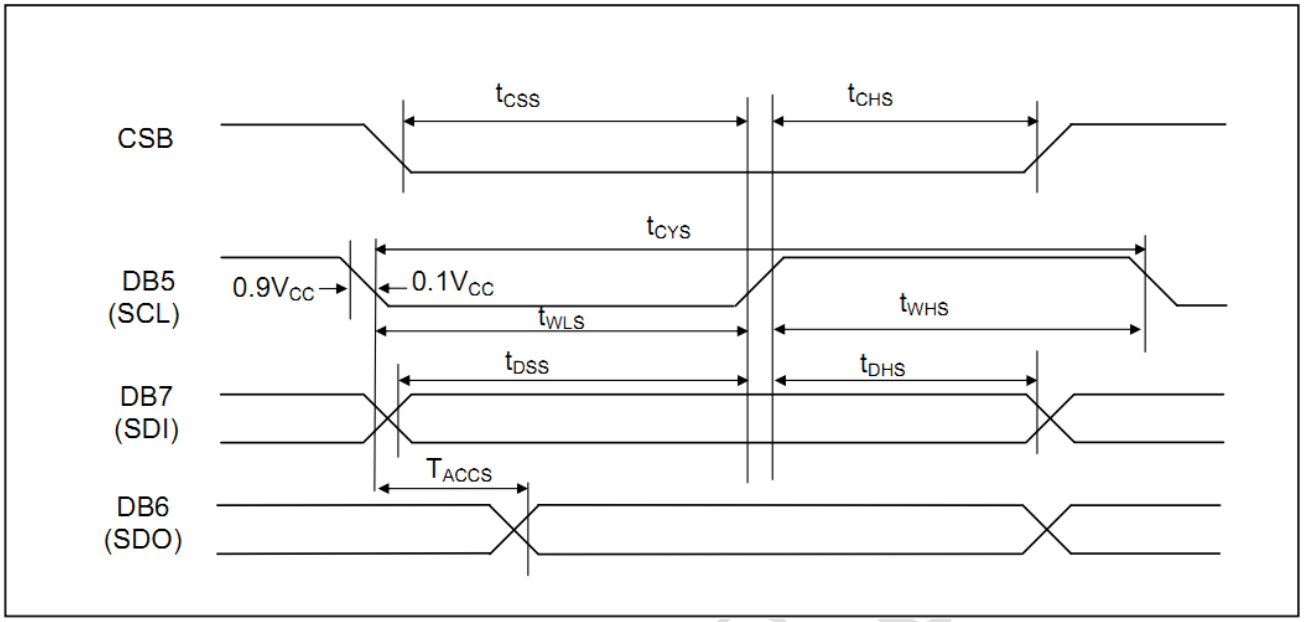


Figure 3. Serial Interface Characteristics

(Vcc = 4.5 to 5.5V, Ta = -40 to +85℃)

Item	Signal	Symbol	Min.	Typ.	Max.	Unit	Remark
Serial clock cycle SCL high pulse width SCL low pulse width	DB5 (SCL)	t_{CYS} t_{WHS} t_{WLS}	300 100 100	- - -	- - -	ns	
CS1B setup time CS1B hold time	CSB	t_{CSS} t_{CHS}	150 150	- -	- -	ns	
Data setup time Data hold time	DB7 (SDI)	t_{DSS} t_{DHS}	100 100	- -	- -	ns	
Read access time	DB6 (SDO)	t_{ACCS}	-	-	80	ns	

9 Reference Initial code :

9.1 8 bit-68 interface mode

```
#define         one             0x80
#define         two             0xc0
#define         Data_BUS P1
sbit            busy     =P1^7;
sbit            RS        =P3^0;
sbit            RW         =P3^7;
sbit            Enable   =P3^4;
//sbit          CS      =P3^3;       //only for SPI

unsigned char code CGRAM1[8] ={0x04,0x0E,0x15,0x04,0x04,0x04,0x04,0x04,};   //  ↑

main()
{
   unsigned char i,mode,CHAR_ADD,font;

   Initial_IC();
   CGRAM();
   CHAR_ADD=0X41;
   while(1)
   {
         //show char Max
            WriteIns(one);                     //the first word in top line address is 0x80
         for(i = 0; i<20;i++)
            WriteData(CHAR_ADD+i);
            WriteIns(two);                     //the first word in second line address is 0xC0
         for(i = 0; i<20;i++)
            WriteData(CHAR_ADD+i);
   }
}

void Initial_IC()
{
   WriteIns(0x38);        //function set,8-bit transfer,2-lines display & 5*8 dot characteristic, font 00
   delay(1);
   WriteIns(0x38);        //function set,8-bit transfer,2-lines display & 5*8 dot characteristic, font 00
   delay(1);
   WriteIns(0x38);        //function set,8-bit transfer,2-lines display & 5*8 dot characteristic, font 00
   delay(1);
   WriteIns(0x38);        //function set,8-bit transfer,2-lines display & 5*8 dot characteristic, font 00
   delay(1);
   WriteIns(0x0C);        //Display ON/OFF control,Display ON,Cursor&Blink OFF
   delay(1);
   WriteIns(0x06);        //Entry Mode Set, address increment & Shift off
   delay(1);
   WriteIns(0x02);        //Return Home
   delay(1);
   WriteIns(0x01);        //Clear Display
   delay(1);
}
```

```
void WriteIns(char instruction)
{
    CheckBusy();
    RS = 0;
    RW = 0;
    Enable = 1;
    Data_BUS = instruction;
    Enable = 0;                //1us
  }


void WriteData(char data1)
{
    CheckBusy();
    RS = 1;
    RW = 0;
    Enable = 1;
    Data_BUS = data1;
    Enable = 0;
  }


void CheckBusy()
{
    Data_BUS = 0xff;
    RS = 0;
    RW = 1;
    do
    {

       Enable = 1;
       busy_f = busy;
       Enable = 0;
    }while(busy_f);
}


void CGRAM()
{
    unsigned char i;
    WriteIns(0x40);             //SET CG_RAM ADDRESS 000000
    for(i = 0;i<=24;i++)
    {
        WriteData(CGRAM1[i]);
    }
}
```

9.2 4 bit-68 interface mode

```
main()
{
   The same as 8 bit-68 interface mode
}

void Initial_IC()
{
  /*need to set five "0x00" cmds. Could put in the start of program or in the end of program*/
  WriteIns(0x00);
  WriteIns(0x00);
  WriteIns(0x00);
  WriteIns(0x00);
  WriteIns(0x00);

   WriteIns(0x20);//function set //do it only once
   WriteCmd(0x28);//function set
   WriteCmd(0x0c);//display on
   WriteCmd(0x06);//entry mode set
   WriteCmd(0x02); //Return Home
   WriteCmd(0x01);//clear display
}


void CheckBusy()
{
  Data_BUS = 0xff;
  RS = 0;
  RW = 1;
  do
  {
        Enable = 1;
        busy_f = busy;
        Enable = 0;
        Enable = 1; //dummy read
        Enable = 0;
        _nop_();
  }while(busy_f);
}


void WriteIns(unsigned char instruction)
{
  RS = 0;
  Enable = 0;
  RW = 0;
  Data_BUS = instruction&0xf0;
  Enable = 1;                  //1us
  _nop_();                  //1us
  Enable = 0;               //1us
}


void WriteCmd(unsigned char cmd)
{
  unsigned char hIns=cmd,lIns=cmd;

  RS = 0;
  RW = 0;
  Data_BUS = hIns&0xf0;
```

```
   Enable = 1;                //1us
   _nop_();                 //1us
   Enable = 0;              //1us
   Data_BUS = lIns<<4;
   Enable = 1;                //1us
   _nop_();                 //1us
   Enable = 0;              //1us
   CheckBusy();
}

void WriteData(unsigned char dat)
{
   unsigned char hDat=dat,lDat=dat;

   RS = 1;
   RW = 0;
   Data_BUS = hDat&0xf0;
   Enable = 1;
   _nop_();
   Enable = 0;

   Data_BUS = lDat<<4;
   Enable = 1;
   _nop_();
   Enable = 0;
   CheckBusy();
}


void WriteString(unsigned char count,unsigned char * MSG)
{
   unsigned char    i;

   for(i = 0; i<count;i++)
   {
          WriteData(MSG[i]);
   }
}


void CGRAM()
{
    unsigned char i;

    WriteIns(0x40);                //SET CG_RAM ADDRESS 000000
    for(i = 0;i<=24;i++)
    {
        WriteData(CGRAM1[i]);
    }
}
```

9.3 8 bit-80 interface mode

```
main()
{
   The same as 8 bit-68 interface mode
}

void Initial_IC()
{
   The same as 8 bit-68 interface mode
}

void WriteIns(char instruction)
{
   CheckBusy();
   RDB = 1;
   RS = 0;
   WRB = 0;
   Data_BUS = instruction;
   WRB = 1;                              //1us
}

void WriteData(char data1)
{
    CheckBusy();
    RDB = 1;
    RS = 1;
    WRB = 0;
    Data_BUS = data1;
    WRB = 1;
}

void CheckBusy()
{
   Data_BUS = 0xff;
   RS = 0;
   WRB = 1;
   do
   {

      RDB = 0;
      busy_f = busy;
      RDB = 1;
   }while(busy_f);
}

void CGRAM()
{
   unsigned char i;
   WriteIns(0x40);      //SET CG_RAM ADDRESS 000000
   for(i = 0;i<=24;i++)
   {
       WriteData(CGRAM1[i]);
   }
}
```

9.4 4 bit-80 interface mode

```
main()
{
   The same as 8 bit-68 interface mode
}

void Initial_IC()
{
   The same as 4 bit-68 interface mode
}

void WriteIns(unsigned char instruction)
{
   RS = 0;
   WRB = 1;
   RDB = 1;
   Data_BUS = instruction&0xf0;
   WRB = 0;                    //1us
   _nop_();                //1us
   WRB = 1;                //1us
}

void WriteCmd(unsigned char cmd)
{
   unsigned char hIns=cmd,lIns=cmd;

   RS = 0;
   RDB = 1;
   Data_BUS = hIns&0xf0;
   WRB = 0;                    //1us
   _nop_();                //1us
   WRB = 1;                //1us
   Data_BUS = lIns<<4;
   WRB = 0;                    //1us
   _nop_();                //1us
   WRB = 1;                //1us
   CheckBusy();
}
```

```
void WriteData(unsigned char dat)
{
   unsigned char hDat=dat,lDat=dat;

   RS = 1;
   RDB = 1;
   Data_BUS = hDat&0xf0;
   WRB = 0;
   _nop_();
   WRB = 1;

   Data_BUS = lDat<<4;
   WRB = 0;
   _nop_();
   WRB = 1;
   CheckBusy();
}

void WriteString(unsigned char count,unsigned char * MSG)
{
   unsigned char    i;

   for(i = 0; i<count;i++)
   {
          WriteData(MSG[i]);
   }
}

void CheckBusy()
{
   Data_BUS = 0xff; //訊號由 high 變為 low 比較容易,所以全部設為 high.
   RS = 0;
   WRB = 1;
   do
   {
          RDB = 0;
          busy_f = busy;
          RDB = 1;
          RDB = 0; //dummy read
          RDB = 1;
          _nop_();
   }while(busy_f);
}
```

9.5 4-Lines serial interface mode

```
main()
{
   The same as 8 bit-68 interface mode
}

void Initial_IC()
{
   WriteIns(0x38);//function set
   WriteIns(0x06);//entry mode set
   WriteIns(0x02);
   WriteIns(0x01);//clear display
   delay(30);
   WriteIns(0x0c);//display on

}


void WriteIns(unsigned char ins)
{
  unsigned char i;

  CS=0;
  SDI = 0; //RS = 0
  _nop_();
  SCL = 1;
  _nop_();
  SCL = 0;
  _nop_();
  SDI = 0; //RW = 0
  SCL = 1;
  _nop_();
  SCL = 0;

  for (i=0x80;i;i>>=1)
  {
        SDI = ins & i;

        SCL = 1;
        _nop_();
        SCL = 0;
  }
  CS=1;
}


void WriteData(unsigned char dat)
{
  unsigned char i;

  for (i=0x80;i;i>>=1)
  {
        SDI = dat & i;

        SCL = 1;
        _nop_();
        SCL = 0;
  }
}
```

```
void WriteOneData(unsigned char dat)
{
   CS=0;
   SDI = 1; //RS = 1
   _nop_();
   SCL = 1;
   _nop_();
   SCL = 0;
   SDI = 0; //RW = 0
   _nop_();
   SCL = 1;
   _nop_();
   SCL = 0;

   WriteData(dat);
   CS=1;
}


void WriteSerialData(unsigned char count,unsigned char * MSG)
{
   unsigned char    i;

   CS=0;
   SDI = 1; //RS = 1
   _nop_();
   SCL = 1;
   _nop_();
   SCL = 0;
   SDI = 0; //RW = 0
   _nop_();
   SCL = 1;
   _nop_();
   SCL = 0;

   for(i = 0; i<count;i++)
   {
           WriteData(MSG[i]);
   }
   CS=1;
}


void CGRAM(void)
{
   WriteIns(0x40);
   WriteSerialData(8,CGRAM1);
}
```

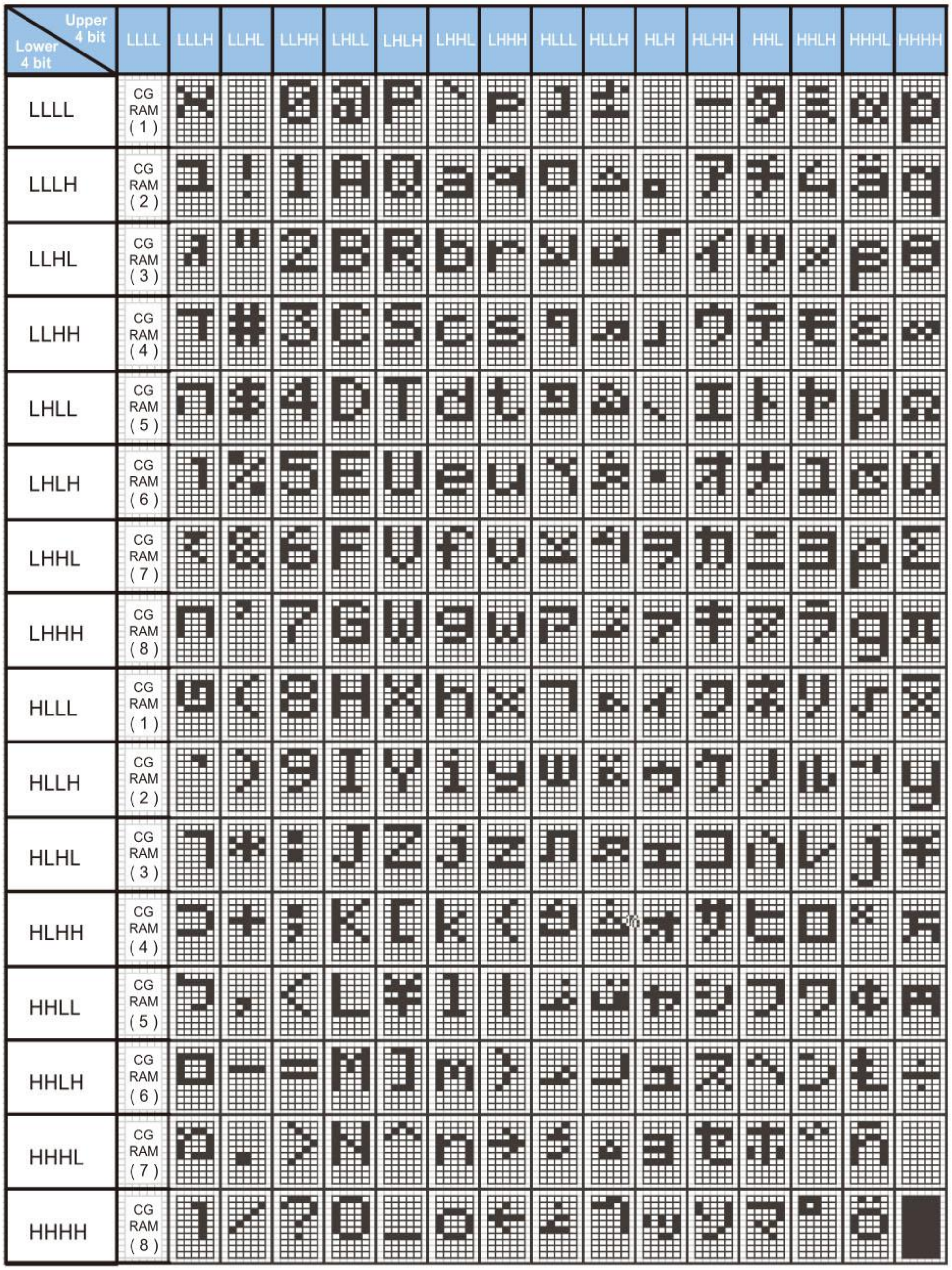
9.6 Graphic mode

```
main()
{
   The same as 8 bit-68 interface mode
}

void Initial_IC()
{
   WriteIns(0x38);        //function set,8-bit transfer,2-lines display & 5*8 dot characteristic, font 00
   delay(1);
   WriteIns(0x38);        //function set,8-bit transfer,2-lines display & 5*8 dot characteristic, font 00
   delay(1);
   WriteIns(0x38);        //function set,8-bit transfer,2-lines display & 5*8 dot characteristic, font 00
   delay(1);
   WriteIns(0x38);        //function set,8-bit transfer,2-lines display & 5*8 dot characteristic, font 00
   delay(1);
   WriteCmd(0x0C);          //Display ON/OFF control,Display ON,Cursor&Blink OFF
   delay(1);
   WriteCmd(0x06);          //Entry Mode Set, address increment & Shift off
   delay(1);
   WriteCmd(0x1F);          //Graphic mode
   delay(1);
   WriteCmd(0x02);          //Return Home
   delay(1);
   WriteCmd(0x01);          //Clear Display
   delay(1);
}


void WriteIns(unsigned char instruction)
{
  RS = 0;
  RW = 0;
  Data_BUS = instruction;
  Enable = 1;                     //1us
  _nop_();                    //1us
  Enable = 0;                 //1us
}


void WriteCmd(char instruction)
{
    CheckBusy();
    RS = 0;
    RW = 0;
    Enable = 1;
    Data_BUS = instruction;
    Enable = 0;             //1us
 }
```

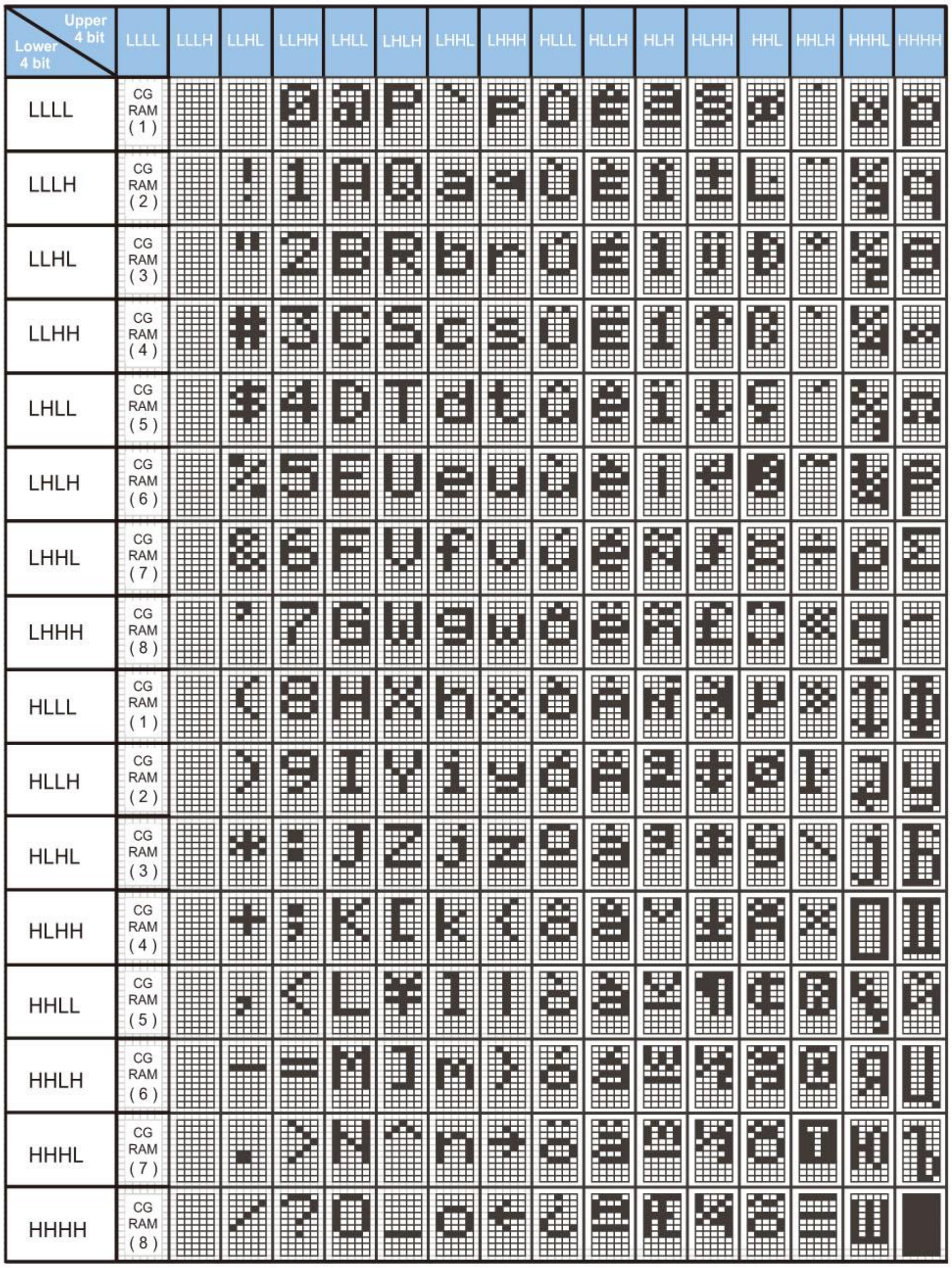
```
void WriteData(char data1)
{
     CheckBusy();
     RS = 1;
     RW = 0;
     Enable = 1;
     Data_BUS = data1;
     Enable = 0;
 }


void CheckBusy()
{
   Data_BUS = 0xff;
   RS = 0;
   RW = 1;
   do
   {
      Enable = 1;
      busy_f = busy;
      Enable = 0;
   }while(busy_f);
```

Appendix Font table

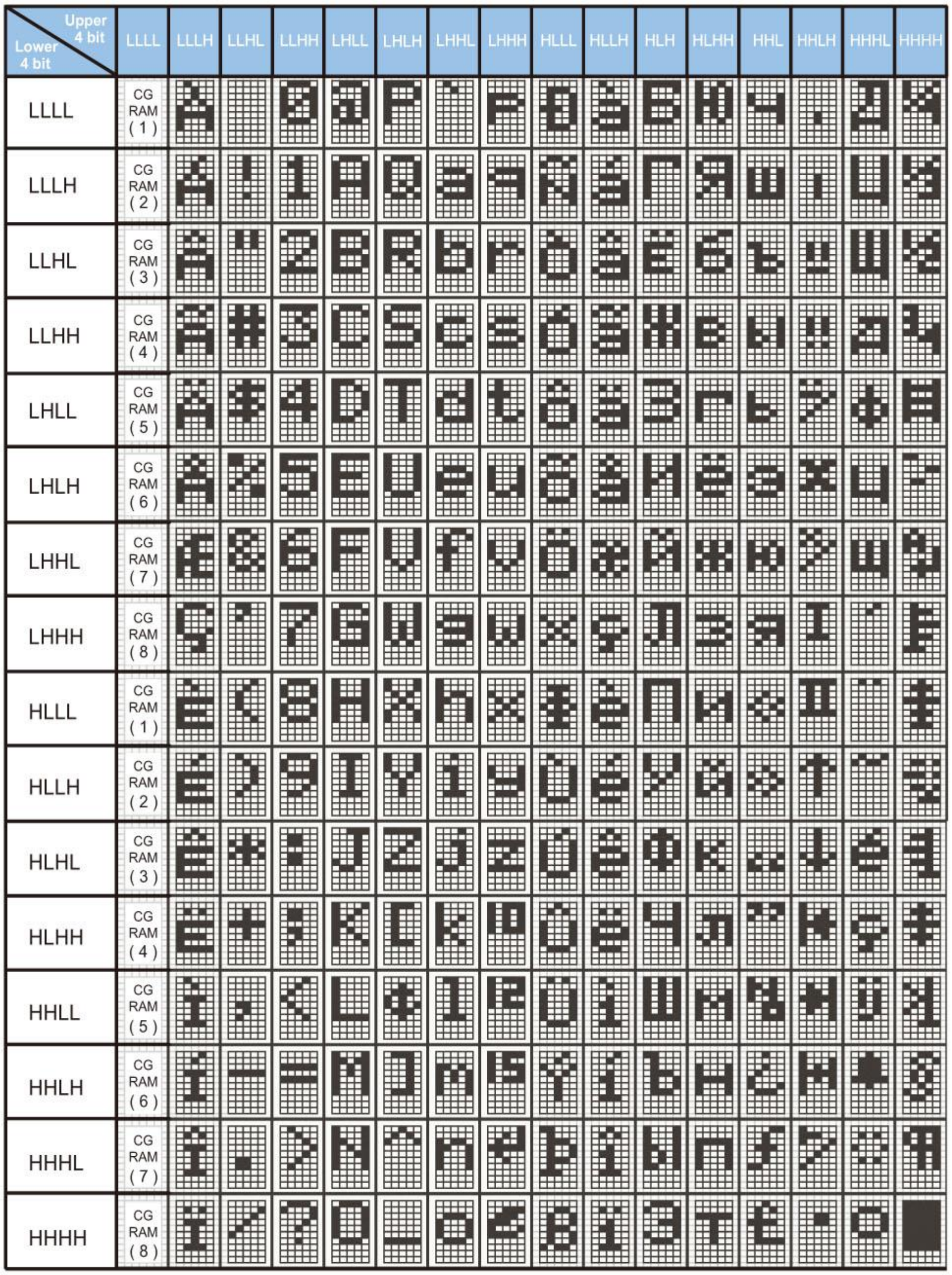
English - Japanese Character Font



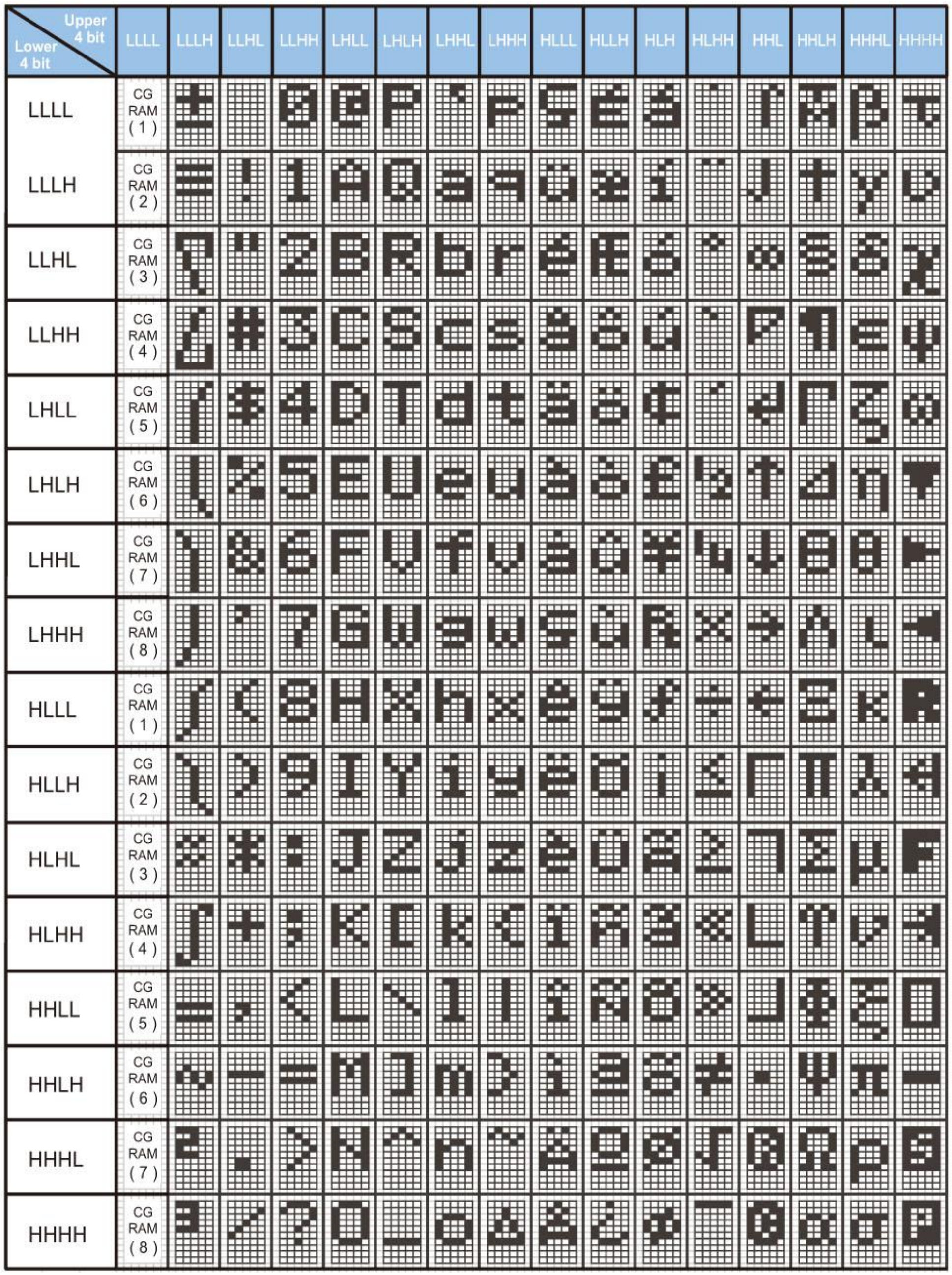
Western European Character Font I


Upper 4 bit
Lower 4 bit
LLLL
LLLH
LLHL
LLHH
LHLL
LHLH
LHHL
LHHH
HLLL
HLLH
HLHL
HLHH
HHLL
HHLH
HHHL
HHHH
LLLL
LLLH
LLHL
LLHH
LHLL
LHLH
LHHL
LHHH
HLLL
HLLH
HLHL
HLHH
HHLL
HHLH
HHHL
HHHH
CG RAM (1)
CG RAM (2)
CG RAM (3)
CG RAM (4)
CG RAM (5)
CG RAM (6)
CG RAM (7)
CG RAM (8)
CG RAM (1)
CG RAM (2)
CG RAM (3)
CG RAM (4)
CG RAM (5)
CG RAM (6)
CG RAM (7)
CG RAM (8)

English - Russian Character Font


Upper 4 bit
Lower 4 bit
LLLL
LLLH
LLHL
LLHH
LHLL
LHLH
LHHL
LHHH
HLLL
HLLH
HLH
HLHH
HHL
HHLH
HHHL
HHHH
LLLL
LLLH
LLHL
LLHH
LHLL
LHLH
LHHL
LHHH
HLLL
HLLH
HLHL
HLHH
HHLL
HHLH
HHHL
HHHH
CG RAM (1)
CG RAM (2)
CG RAM (3)
CG RAM (4)
CG RAM (5)
CG RAM (6)
CG RAM (7)
CG RAM (8)
CG RAM (1)
CG RAM (2)
CG RAM (3)
CG RAM (4)
CG RAM (5)
CG RAM (6)
CG RAM (7)
CG RAM (8)

Western European Character Font II


Upper 4 bit
Lower 4 bit
LLLL
LLLH
LLHL
LLHH
LHLL
LHLH
LHHL
LHHH
HLLL
HLLH
HLHL
HLHH
HHLL
HHLH
HHHL
HHHH
CG RAM (1)
CG RAM (2)
CG RAM (3)
CG RAM (4)
CG RAM (5)
CG RAM (6)
CG RAM (7)
CG RAM (8)