GT23L16U2W3-S

标准汉字字库芯片

11X12 点阵

15X16 点阵

横置横排

UNICODE 全拼输入法码表

GT-三维部件输入法

（SPI 接口/PLII 接口）

用户手册

VER 2.7

2007-Q4

版本修订记录

版本号	修改内容	日期	备注
V2.5	生成数据手册	2007-04	
v2.7	增加 GT 输入法，更名为 GT23L16U2W3-S	2007-10	

目　　录

1 概述.....4
2 特点.....5
3 引脚配置.....7
4 引脚描述.....7
4.1 SPI 接口引脚描述.....8
4.2 PLII 接口引脚描述.....8
5 SPI 接口/PLII 接口与主机接口电路示意图.....9
5.1 SPI 接口与主机接口电路示意图.....9
5.2 PLII 接口与主机接口电路示意图.....10
6 操作指令.....11
6.1 SPI 接口模式下操作.....11
6.2 PLII 接口模式下操作.....13
7 电气特性.....15
7.1 绝对最大额定值.....15
7.2 DC 特性.....15
7.3 AC 特性.....15
8 封装尺寸：SOP20.....19
9 功能描述.....20
9.1 汉字点阵排列格式.....20
9.2 汉字点阵字库结构.....30
9.3 汉字点阵在芯片中的地址计算方法.....32
9.4 全拼拼音－UNICODE 全拼输入法（21003 字）.....37
10 附录.....39
10.1 UNICODE3.0（GB13000）字符区.....39
10.2 扩展字符区字符.....44
10.3 5×7 点 ASCII 字符（96 字符）.....45
10.4 7×8 点 ASCII 字符（96 字符）.....45
10.5 Unicode 字符区字符.....46
10.6 8×16 点扩展字符区 ASCII 字符（126 字符）.....52
10.7 8×16 自定义特殊字符区 ASCII 字符（64 字符）.....53
10.8 8×16 自定义国际音标字符区 ASCII 字符（96 字符）.....53
10.9 汉字点阵排列格式转换程序.....54
10.10 UNICODE3.0 字符区对照表.....57

1 概述

GT23L16U2W3是一款支持UNICODE3.0字符集（27484汉字）的11 X 12、15X16点阵字库芯片，另外还含有1套6X12点扩展字符、1套8X16点扩展字符、1套5X7点ASCII字符、1套7X8点ASCII字符。1套12点ASCII方头字符、1套12点ASCII白正字符、1套16点ASCII方头字符、1套16点ASCII白正字符。同时内含Unicode字符区拉丁文系（LATIN）、希腊文系（LATIN）、西里尔文系（CYRILLIC）文系字符基本可以涵盖81个国家及地区。同时还有非常丰富的自定义区的各种图形符号，以及12点阵和16点阵的国际音标字符。

每个汉字在芯片中是以汉字点阵字模的形式存储的，每个点用一个二进制位表示，存 1 的点，当显示时可以在屏幕上显示亮点，存 0 的点，则在屏幕上不显示。点阵排列格式为横置横排：即一个字节的高位表示左面的点，低位表示右面的点，排满一行的点后再排下一行。这样把点阵信息用来直接在显示器上按上述规则显示，则将出现对应的汉字。

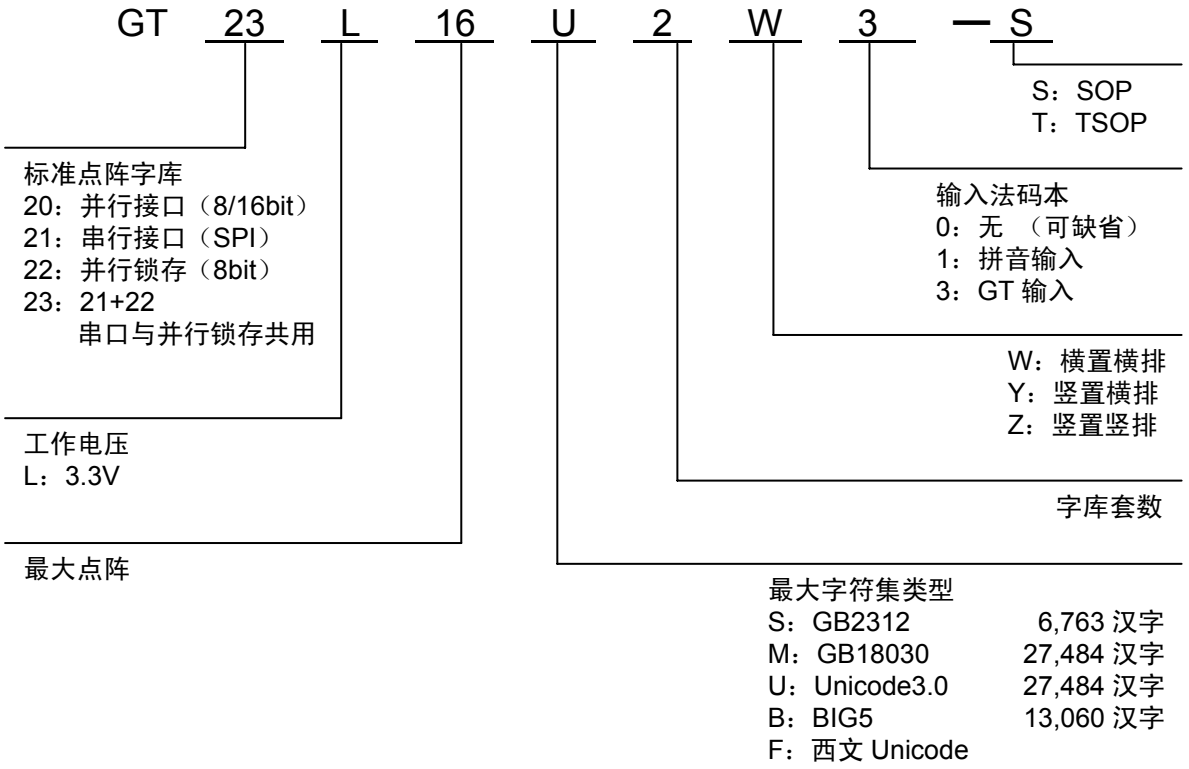
用户只要知道汉字的内码，就可以通过本手册提供的方法计算出该汉字点阵在芯片中的地址，然后就可从该地址连续读出点阵信息。

同时本字库芯片还含有UNICODE（21003汉字）全拼输入法码表，只要将键值串相应的拼音ASCII字符进行索引，即可得到该键值串所对应的汉字内码群。因此本汉字库芯片根据用户的具体需要即可以输入汉字相应的输入编码获取对应该的汉字内码群，也可以输入汉字的内码获取对应该汉字的点阵字形数据。

GT23L16U2W3 支持 2 种不同的总线接口：即 SPI 接口、PLII 接口。

本字库芯片内含 GT 输入法码本，该输入法适用于数字小键盘，如手机、遥控器、电话终端、税控收款机等。

命名规则（标准点阵字库芯片）



2 特点

- 数据总线：SPI 总线兼容的串行接口/PLII 接口
- 点阵排列方式：字节竖置横排
- 访问速度：SPI 时钟频率：40MHz(max.)
  PLII 访问速度：70ns(max.) @3.3V
- 工作电压：2.7V – 3.6V
- 电流：
  工作电流：30mA(max.) @3.3V
  待机电流：15uA(max.) @3.3V
- 封装： SOP20
- 字库内容：

汉 字 字 库	
11×12 点	UNICODE3.0 （27,484 汉字） （ 1,088 字符)
15×16 点	UNICODE3.0 （27,484 汉字） （ 1,088 字符)

ASCII 字符	
6×12 点	扩展字符（126 字符）
12 点阵	ASCII 方头（96 字符）
12 点阵	ASCII 白正（96 字符）
8×16 点	扩展字符（126 字符）
5×7 点	ASCII 字符（96 字符）
7×8 点	ASCII 字符（96 字符）
16 点阵	ASCII 方头（96 字符）
16 点阵	ASCII 白正（96 字符）
8×16 点	Unicode 字符区(786 字符） －拉丁文系（376 字符） －西里尔文系（250 字符） －希腊字字符区（96 字符） 自定义字符区（64 字符）

- 字型样张

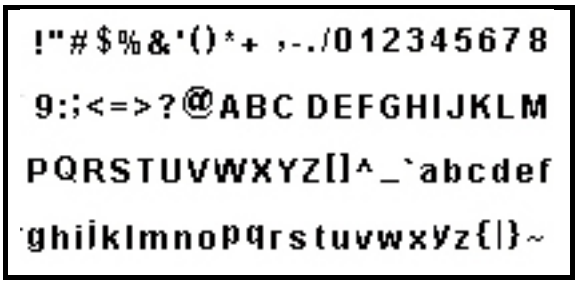
11×12 点 UNICODE3.0 （27,484 汉字）



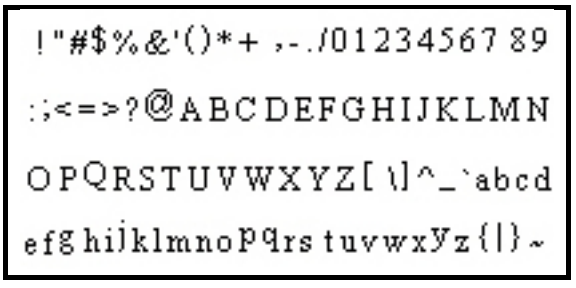
6×12 点扩展字符（126 字符）



12 点阵 ASCII 方头（96 字符）



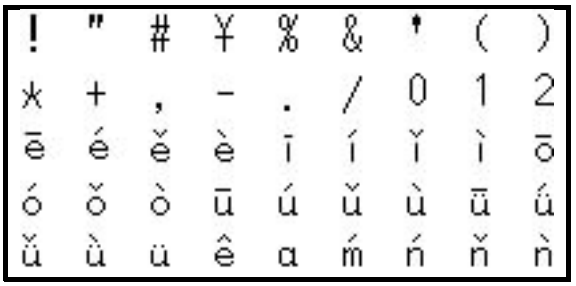
12 点阵 ASCII 白正（96 字符）



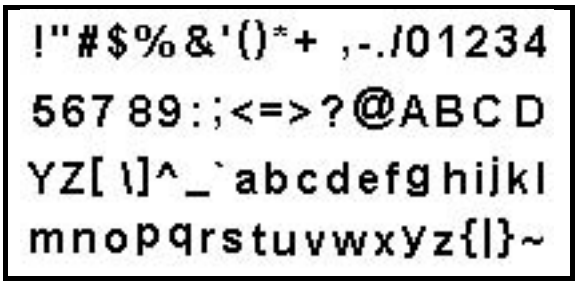
15×16 点 UNICODE3.0 （27,484 汉字）



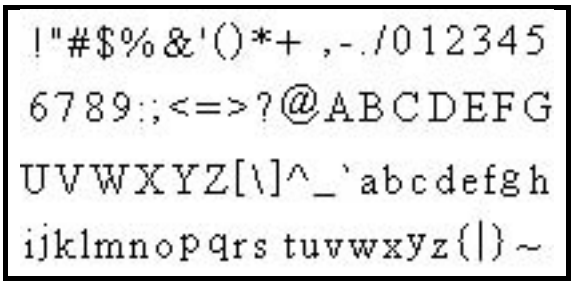
8×16 点扩展字符（126 字符）



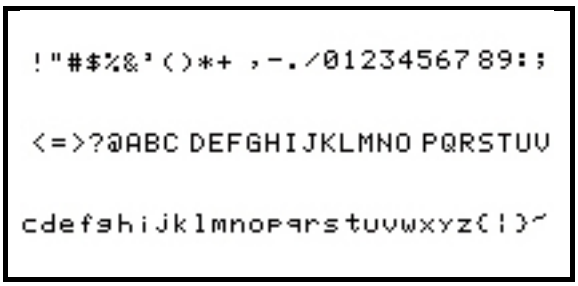
16 点阵 ASCII 方头（96 字符）



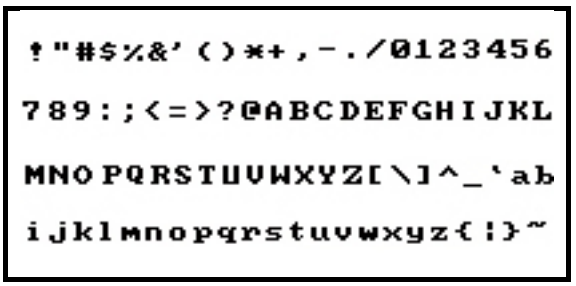
16 点阵 ASCII 白正（96 字符）



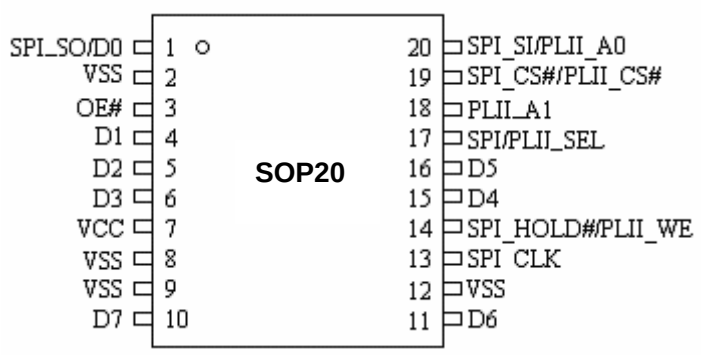
5×7 点 ASCII 字符（96 字符）



7×8 点 ASCII 字符（96 字符）



3 引脚配置



SOP20

4 引脚描述

SOP20	名称	描述	
		GT21L(SPI 接口)	GT22L(PLII 接口)
1	SPI_SO/D0	Serial data output	Data Outputs
2	VSS	Ground	
3	OE#	No Connection	Output Enable Input
4	D1	No Connection	Data Outputs
5	D2	No Connection	Data Outputs
6	D3	No Connection	Data Outputs
7	VCC	Power Supply(3.3V)	
8	VSS	Ground	
9	VSS	Ground	
10	D7	No Connection	Data Outputs
11	D6	No Connection	Data Outputs
12	VSS	Ground	
13	SPI_CLK	Serial clock input	No Connection
14	SPI_HOLD#/PLII_WE	Hold(to pause the device)	Write Enable Input
15	D4	No Connection	Data Outputs
16	D5	No Connection	Data Outputs
17	SPI/PLII_SEL	SPI/PLII SELECT	
		NC: SPI	GND: PLII
18	PLII_A1	No Connection	Address Inputs
19	SPI_CS#/PLII_CS#	Chip enable input	Chip Enable Input
20	SPI_SI/PLII_A0	Serial data input	Address Inputs

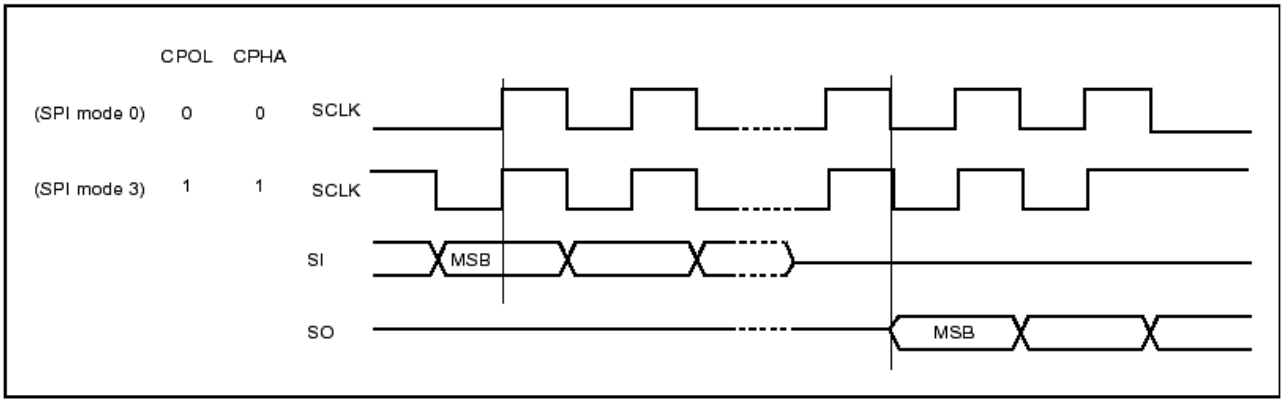
4.1 SPI 接口引脚描述

串行数据输出（SO）：该信号用来把数据从芯片串行输出，数据在时钟的下降沿移出。

串行数据输入（SI）：该信号用来把数据从串行输入芯片，数据在时钟的上升沿移入。

串行时钟输入（SCLK）：数据在时钟上升沿移入，在下降沿移出。

片选输入（CS#）：所有串行数据传输开始于CE#下降沿，CE#在传输期间必须保持为低电平，在两条指令之间保持为高电平。

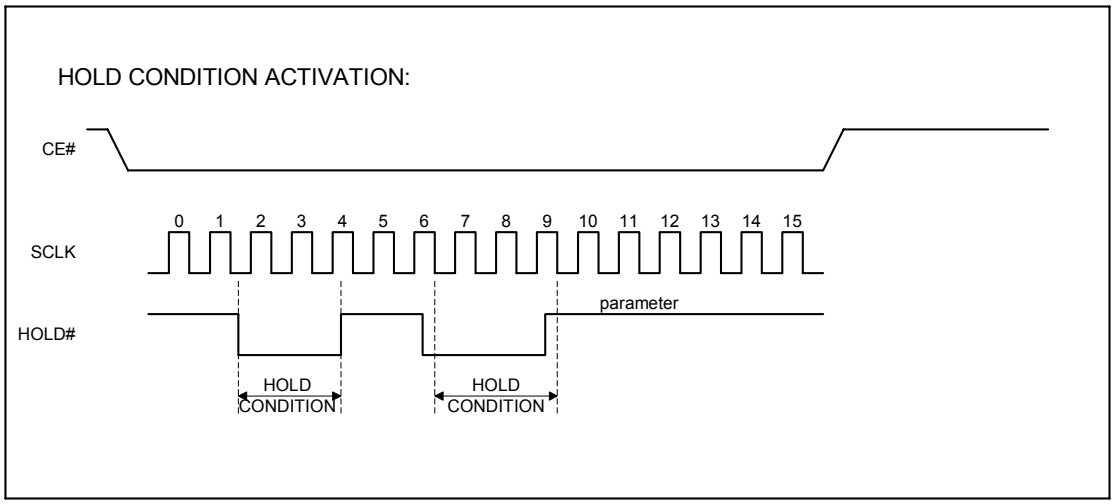


总线挂起输入（HOLD#）：

该信号用于片选信号有效期间暂停数据传输，在总线挂起期间，串行数据输出信号处于高阻态，芯片不对串行数据输入信号和串行时钟信号进行响应。

当HOLD#信号变为低并且串行时钟信号（SCLK）处于低电平时，进入总线挂起状态。

当HOLD#信号变为高并时串行时钟信号（SCLK）处于低电平时，结束总线挂起状态。



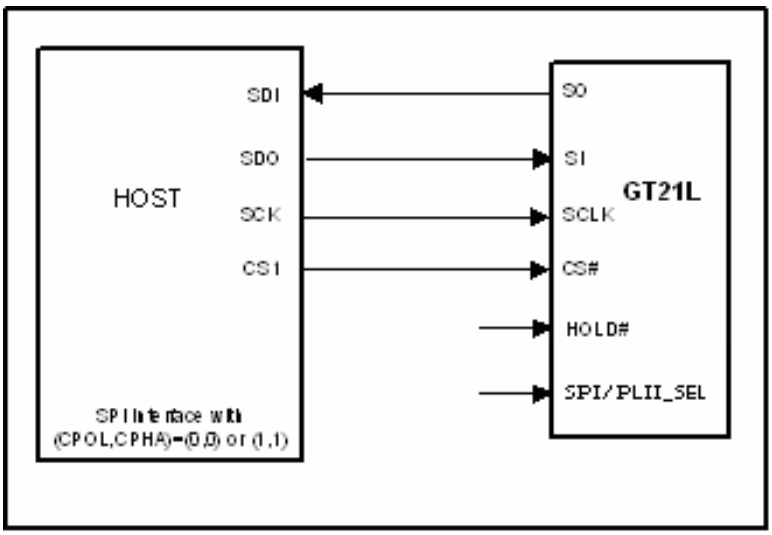
4.2 PLII 接口引脚描述

Pin name	I/O	描述
A[1..0]	I	地址寄存器寻址
D[7..0]	I/O	地址输入/数据输出
CE#	I	片选信号输入，低有效
OE#	I	“输出使能”信号输入，OE# 为低时输出使能
WE#	I	“写使能”信号输入，WE# 为低时写使能

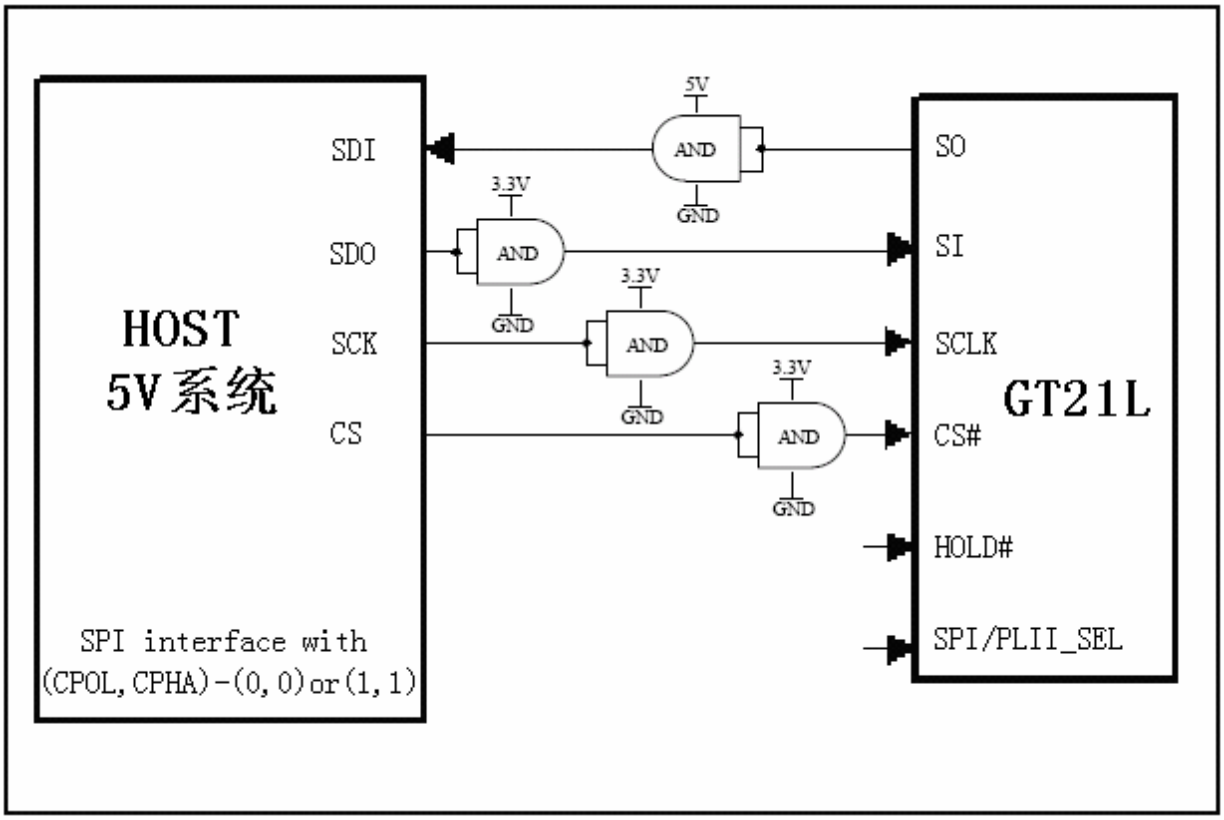
5 SPI 接口/PLII 接口与主机接口电路示意图

5.1 SPI 接口与主机接口电路示意图

SPI/PLII_SEL（管脚内部有 100K 上拉电阻）悬空，字库芯片选择 SPI 接口模式，与主机接口电路连接可以参考下图。

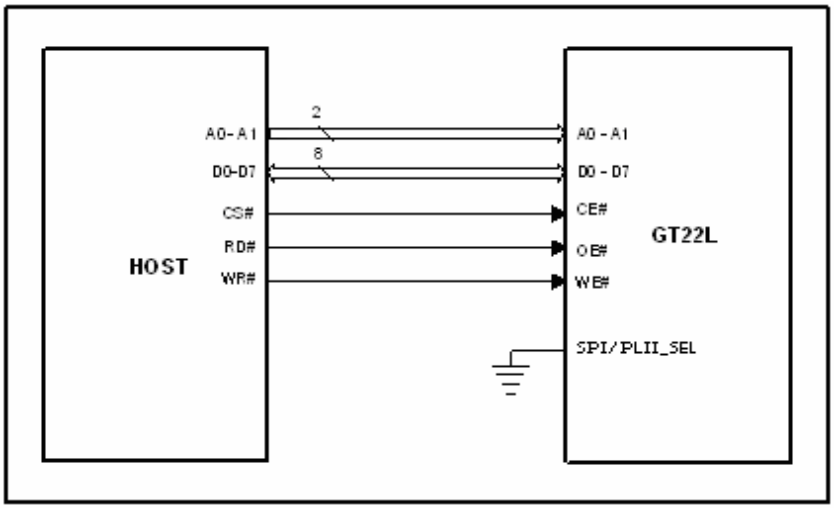


注意：若是采用系统电压为 5V 的，则需要进行电平转换匹配连接 GT21L 库芯片，可以参考下图。

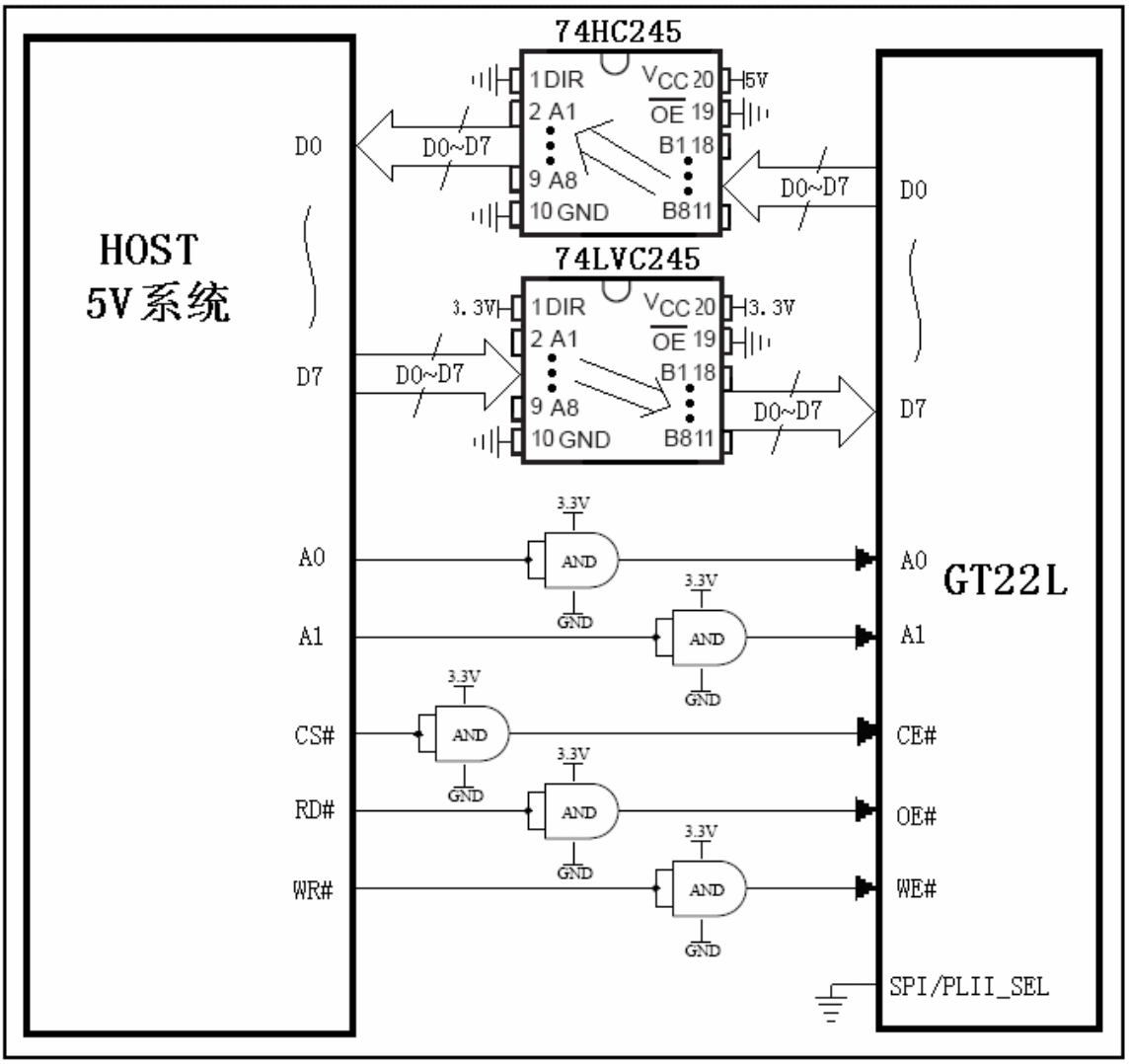


5.2 PLII 接口与主机接口电路示意图

SPI/PLII_SEL（管脚内部有 100K 上拉电阻）接地，字库芯片选择 SPI 接口模式，与主机接口电路连接可以参考下图。



注意：若是采用系统电压为 5V 的，则需要进行电平转换匹配(如:通过 74LVC245 进行电平转换)，连接 GT20L 库芯片，可以参考下图。



6 操作指令

6.1 SPI 接口模式下操作

6.1.1 指令参数

Instruction Set

Instruction	Description	Instruction Code(One-Byte)		Address Bytes	Dummy Bytes	Data Bytes
READ	Read Data Bytes	0000 0011	03 h	3	—	1 to ∞
FAST_READ	Read Data Bytes at Higher Speed	0000 1011	0B h	3	1	1 to ∞

所有对本芯片 SPI 接口的操作只有 2 个，那就是 Read Data Bytes (READ “一般读取”)和 Read Data Bytes at Higher Speed (FAST_READ “快速读取点阵数据”)。.

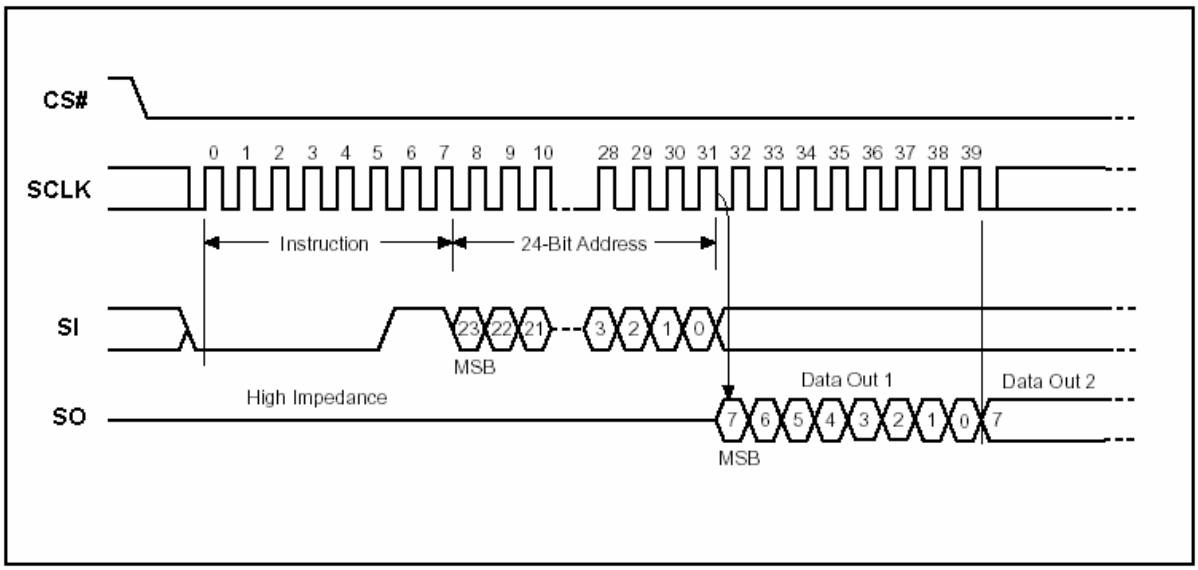
6.1.2 Read Data Bytes（一般读取）

Read Data Bytes 需要用指令码来执行每一次操作。READ 指令的时序如下(图)：

- 首先把片选信号（CS#）变为低，紧跟着的是 1 个字节的命令字（03 h）和 3 个字节的地址和通过串行数据输入引脚（SI）移位输入，每一位在串行时钟（SCLK）上升沿被锁存。
- 然后该地址的字节数据通过串行数据输出引脚（SO）移位输出，每一位在串行时钟（SCLK）下降沿被移出。
- 读取字节数据后，则把片选信号（CS#）变为高，结束本次操作。

如果片选信号（CS#）继续保持为底，则下一个地址的字节数据继续通过串行数据输出引脚（SO）移位输出。

图：Read Data Bytes (READ) Instruction Sequence and Data-out sequence



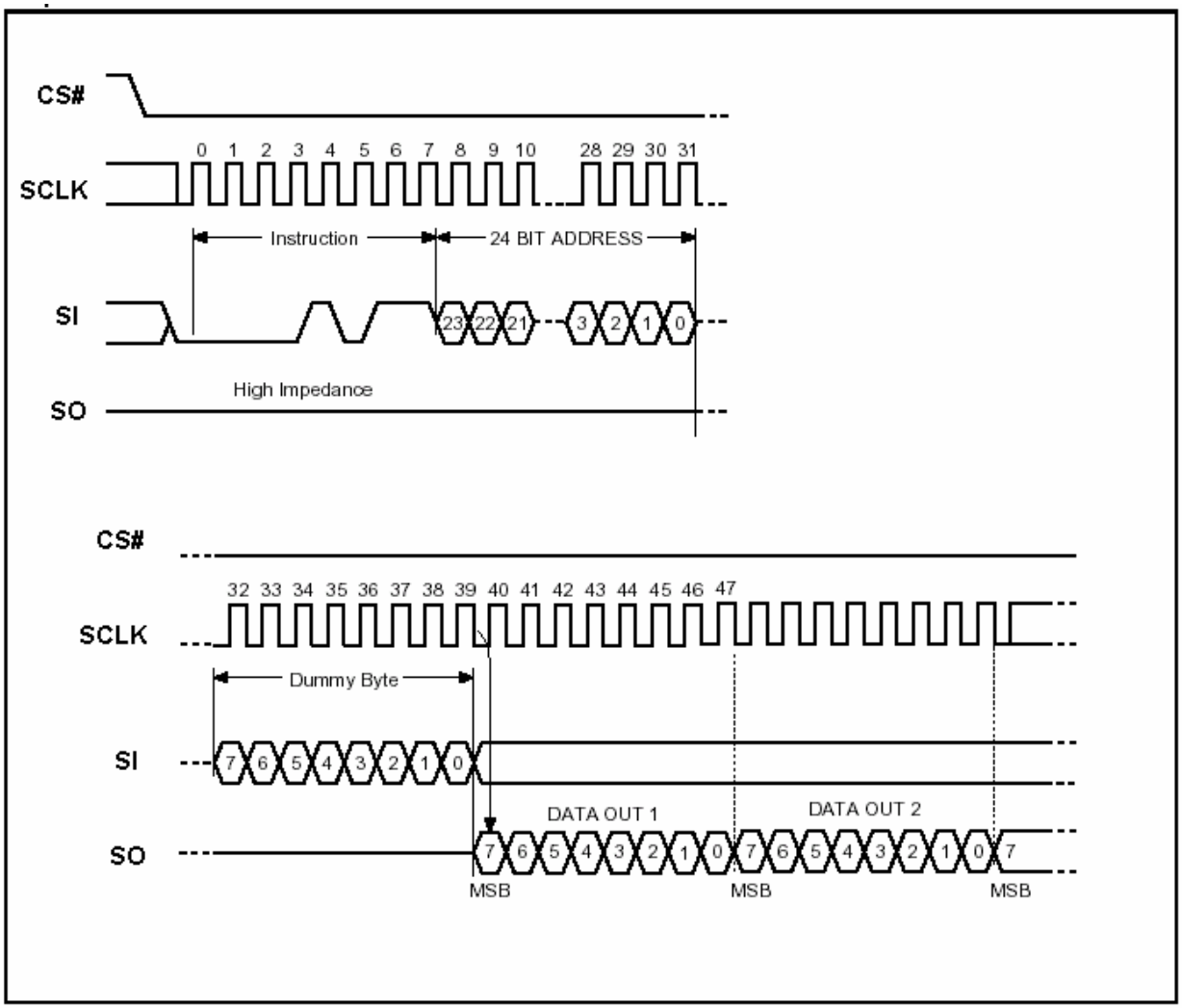
6.1.3 Read Data Bytes at Higher Speed（快速读取点阵数据）

Read Data Bytes at Higher Speed 需要用指令码来执行操作。READ_FAST 指令的时序如下(图)：

- 首先把片选信号（CS#）变为低，紧跟着的是 1 个字节的命令字（0B h）和 3 个字节的地址以及一个字节 Dummy Byte 通过串行数据输入引脚（SI）移位输入，每一位在串行时钟（SCLK）上升沿被锁存。
- 然后该地址的字节数据通过串行数据输出引脚（SO）移位输出，每一位在串行时钟（SCLK）下降沿被移出。
- 如果片选信号（CS#）继续保持为底，则下一个地址的字节数据继续通过串行数据输出引脚（SO）移位输出。例：读取一个 15x16 点阵汉字需要 32Byte，则连续 32 个字节读取后结束一个汉字的点阵数据读取操作。

  如果不需要继续读取数据，则把片选信号（CS#）变为高，结束本次操作。

图：Read Data Bytes at Higher Speed (READ_FAST) Instruction Sequence and Data-out sequence



6.2 PLII 接口模式下操作

在PLII模式下，字库芯片内部有3个地址保持寄存器，HOST 读字库芯片时，字库芯片把地址寄存器对应字库芯片地址内容送给HOST，并且HOST每读取一个字节后，字库芯片内部会把地址寄存器的值增1。当地址寄存器越过最大地址时自动归零。字库芯片可以对地址寄存器进行写操作。字库芯片上电时，内部硬件把地址寄存器清零。

6.2.1 信号描述

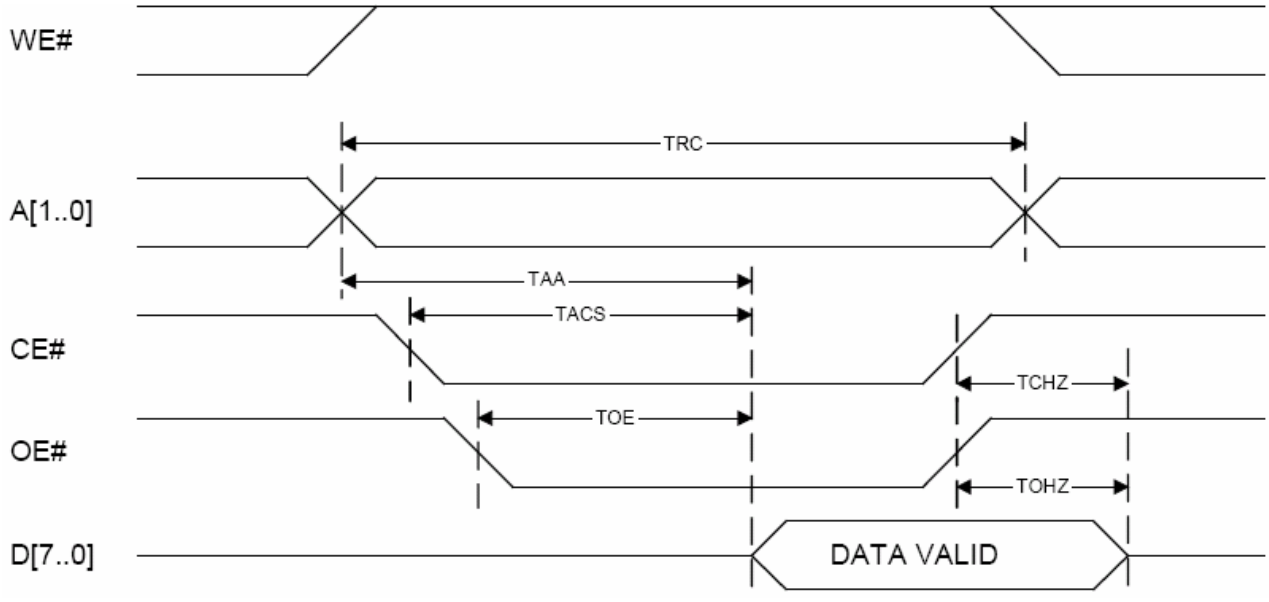
Pin name	I/O	描述
A[1..0]	I	地址寄存器寻址
D[7..0]	I/O	地址输入/数据输出
CE#	I	片选信号输入，低有效
OE#	I	“输出使能”信号输入，OE# 为低时输出使能
WE#	I	“写使能”信号输入，WE# 为低时写使能

真值表

Mode	CE#	OE#	WE#	D[7..0]
Other	H	X	X	High-Z
Read	L	L	H	Data Out
write	L	H	L	Addr In

6.2.2 读操作

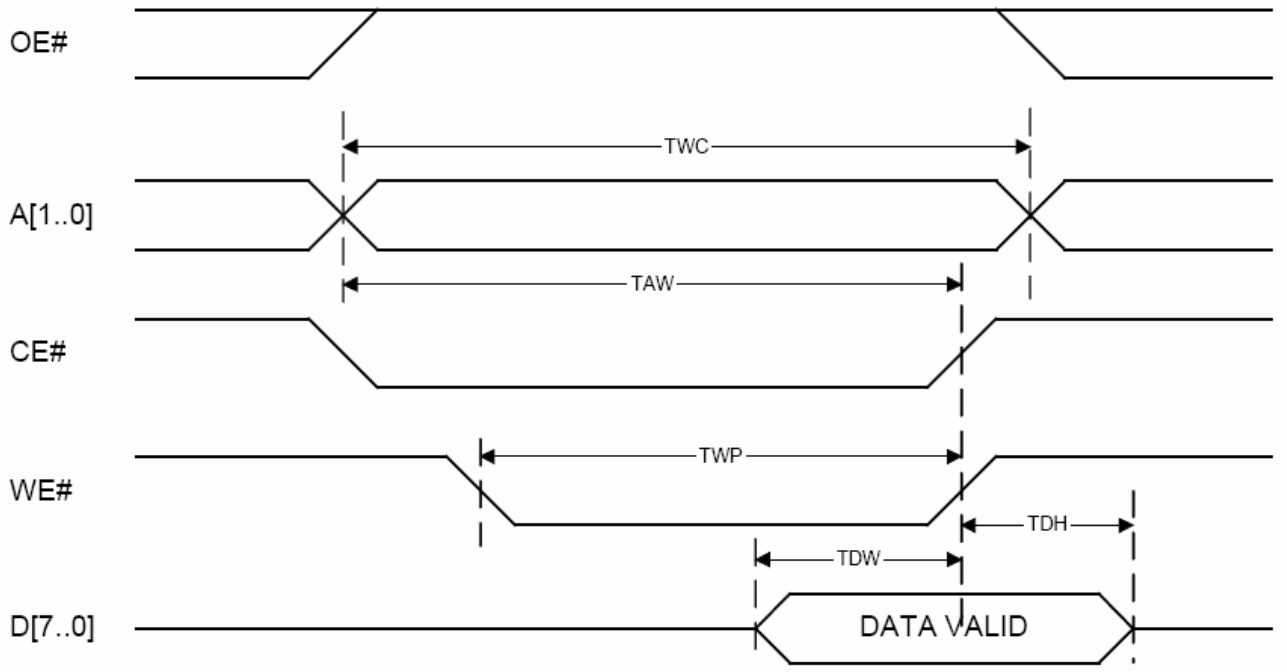
读字库芯片内部的点阵字库数据时，HOST 写字库芯片的 3 个地址寄存器，字库芯片把对应地址的数据送给 HOST。在 OE#和 CE#同时为低电平、WE#为高电平的情况下，HOST 可以从数据总线（D[7..0]）读出字库芯片的个字节的数据。当出现 OE# 或 CE# 中的一个信号变高，则字库芯片内部在此时刻把地址寄存器增 1，并保持地址寄存器的值。



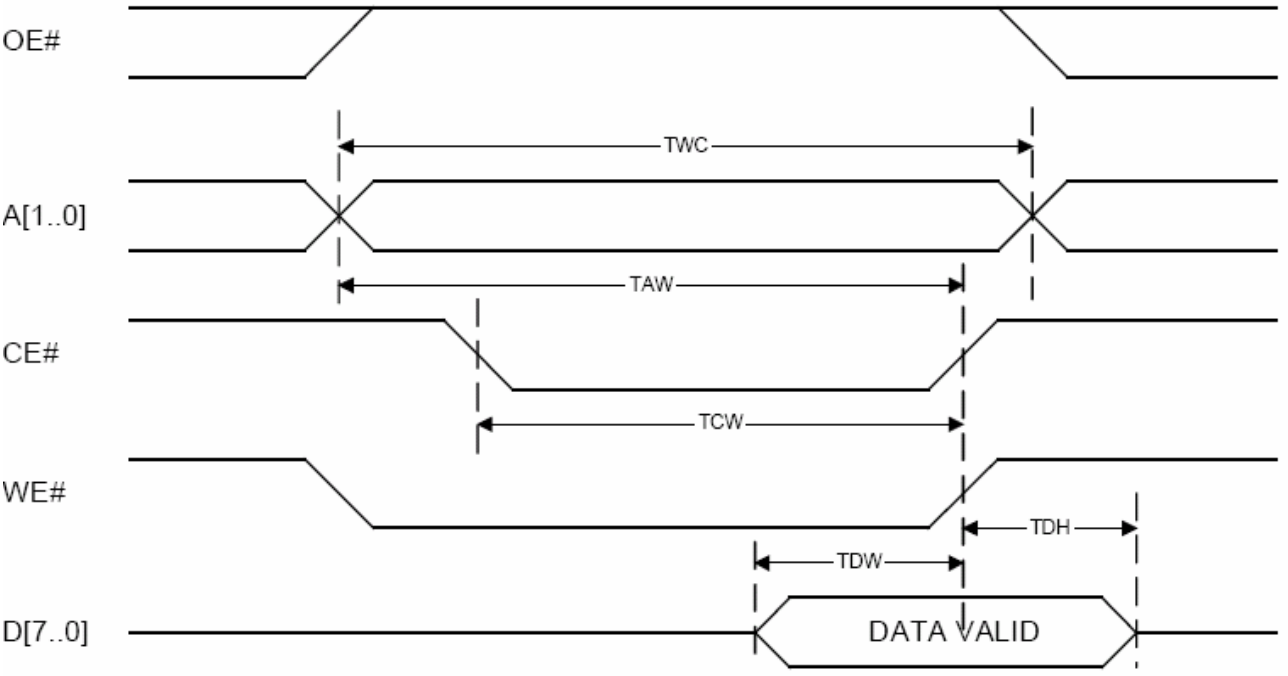
读周期时序波形图

6.2.3 写操作

字库芯片内部有 3 个地址保持寄存器，HOST 读字库芯片时需要把地址写到字库芯片中。在 WE#和 CE#同时为低电平、OE#为高电平的情况下，HOST 可以通过数据总线（D[7..0]）写 1 个字节数据到字库芯片。



写周期时序波形图（WE#控制的时序）



写周期时序波形图（CE#控制的时序）

7 电气特性

7.1 绝对最大额定值

Symbol	Parameter	Min.	Max.	Unit	Condition
T_{OP}	Operating Temperature	0	70	℃	
T_{STG}	Storage Temperature	-65	125	℃	
VCC	Supply Voltage	2.7	3.6	V	
V_{IN}	Input Voltage	-0.5	VCC+0.5	V	
GND	Power Ground	0	0	V	

7.2 DC 特性

Condition：T_{OP} = 0℃ to 70℃，GND=0V

Symbol	Parameter	Min.	Max.	Unit	Condition
I_{DD}	VCC Supply Current(active)		40	mA	
I_{SB}	VCC Standby Current		10	uA	
V_{IL}	Input LOW Voltage	-0.3	0.6	V	VCC=3V
V_{IH}	Input HIGH Voltage	2.2	VCC+0.3	V	
V_{OL}	Output LOW Voltage		0.4 (I_{OL}=1.6mA)	V	
V_{OH}	Output HIGH Voltage	0.8VCC (I_{OH}=-0.4mA)		V	
I_{LI}	Input Leakage Current	0	+10	uA	
I_{LO}	Output Leakage Current	0	+10	uA	

Note：I_{IL}：Input LOW Current，I_{IH}：Input HIGH Current，
I_{OL}：Output LOW Current，I_{OH}：Output HIGH Current，

7.3 AC 特性

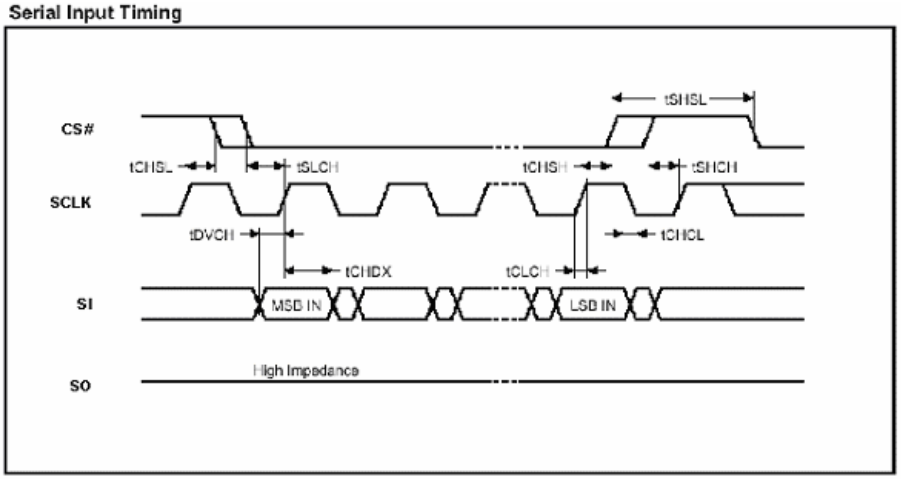
7.3.1 SPI 接口模式下 AC 特性

Condition：T_{OP} = 0℃ to 70℃，VCC= 2.7V to 3.6V

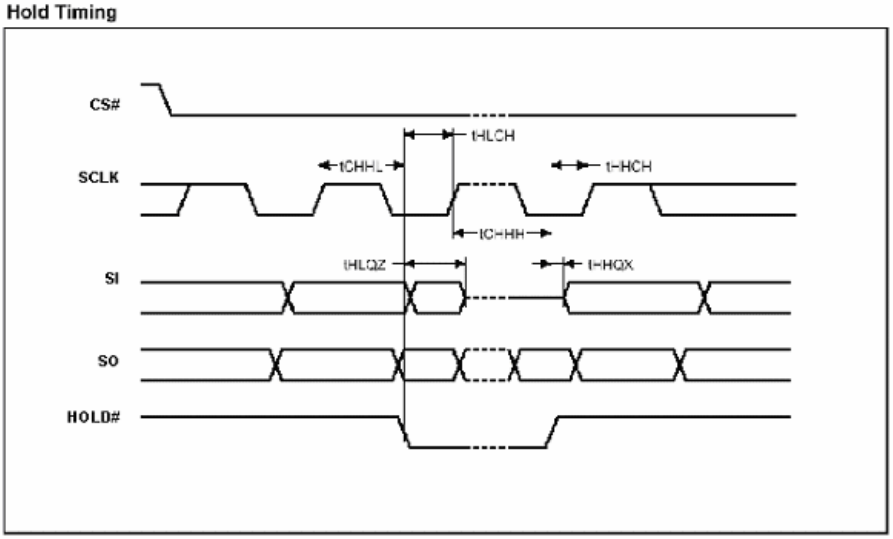
Symbol	Alt.	Parameter	Min.	Max.	Unit
Fc	Fc	Clock Frequency	D.C.	40	MHz
t_{CH}	t_{CLH}	Clock High Time	11		ns
t_{CL}	t_{CLL}	Clock Low Time	11		ns
t_{CLCH}		Clock Rise Time(peak to peak)	0.1		V/ns
t_{CHCL}		Clock Fall Time (peak to peak)	0.1		V/ns
t_{SLCH}	t_{CSS}	CS# Active Setup Time (relative to SCLK)	5		ns
t_{CHSL}		CS# Not Active Hold Time (relative to SCLK)	5		ns
t_{DVCH}	t_{DSU}	Data In Setup Time	2		ns
t_{CHDX}	t_{DH}	Data In Hold Time	5		ns
t_{CHSH}		CS# Active Hold Time (relative to SCLK)	5		ns
t_{SHCH}		CS# Not Active Setup Time (relative to SCLK)	5		ns
t_{SHSL}	t_{CSH}	CS# Deselect Time	100		ns
t_{SHQZ}	t_{DIS}	Output Disable Time		9	ns

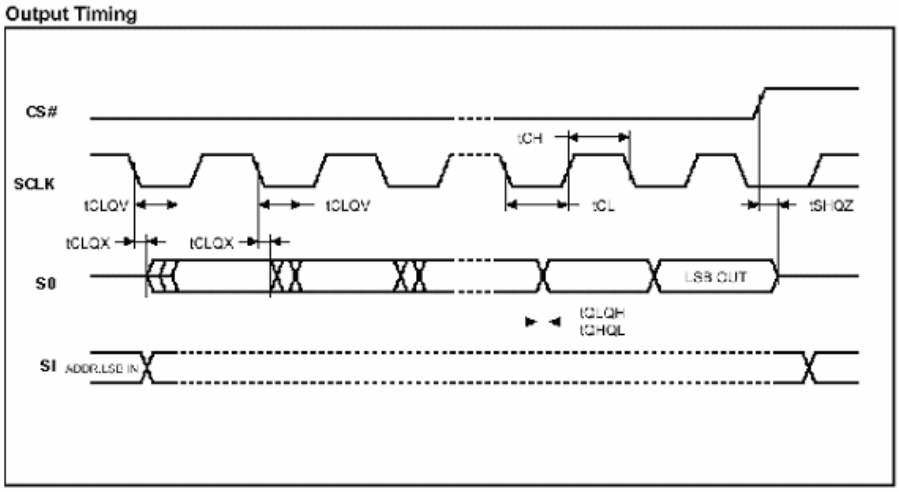
t CLQV	tV	Clock Low to Output Valid		9	ns
t CLQX	tHO	Output Hold Time	0		ns
t HLCH		HOLD# Setup Time (relative to SCLK)	5		ns
t CHHH		HOLD# Hold Time (relative to SCLK)	5		ns
t HHCH		HOLD Setup Time (relative to SCLK)	5		ns
t CHHL		HOLD Hold Time (relative to SCLK)	5		ns
t HHQX	tLZ	HOLD to Output Low-Z		9	ns
t HLQZ	tHZ	HOLD# to Output High-Z		9	ns

Serial Input Timing



Hold Timing

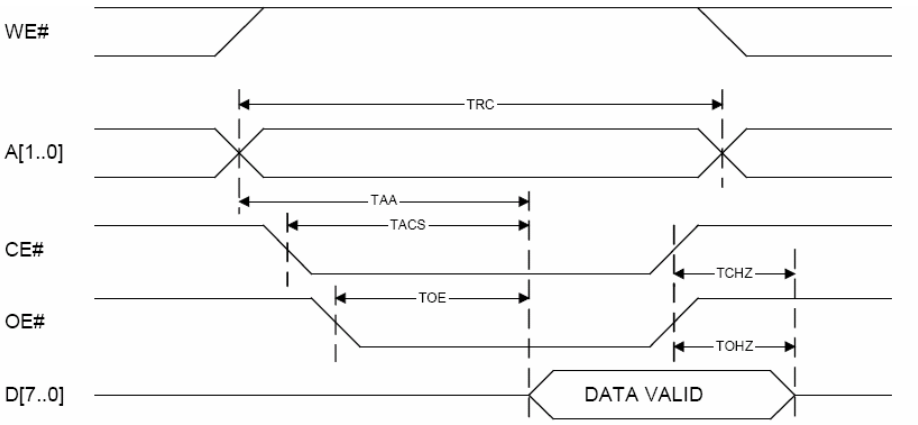




7.3.2 PLII 接口模式下 AC 特性

7.3.2.1 读周期时间特性

Symbol	Parameter	Min.	Max.	Unit
TRC	Read Cycle Time	70	-	ns
TAA	Address Access Time	-	60	ns
TACS	Chip Select Access Time	-	60	ns
TOE	Output Enable to Output Valid	-	30	ns
TCHZ	Chip Deselect to Output in High-Z	-	10	ns
TOHZ	Output Disable to Output in High-Z	-	10	ns

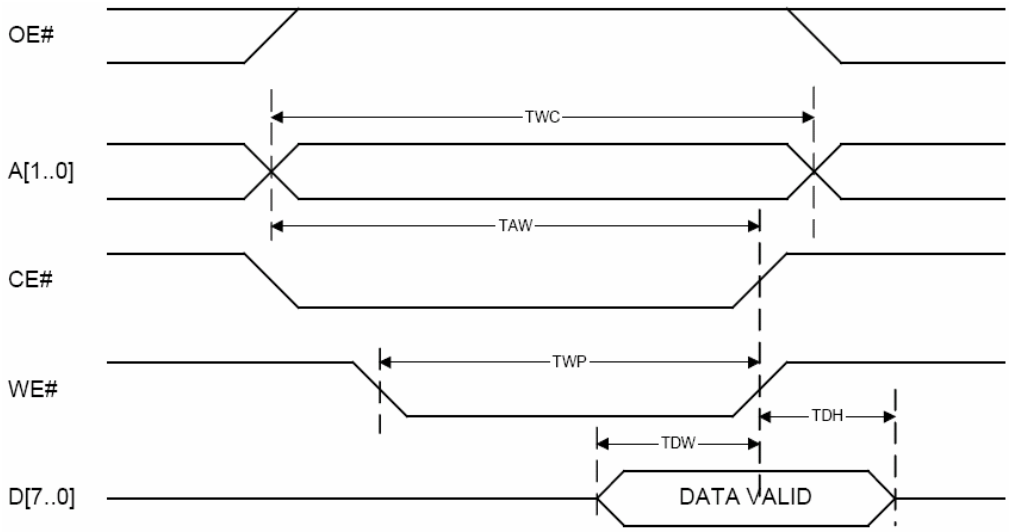


读周期时序波形图

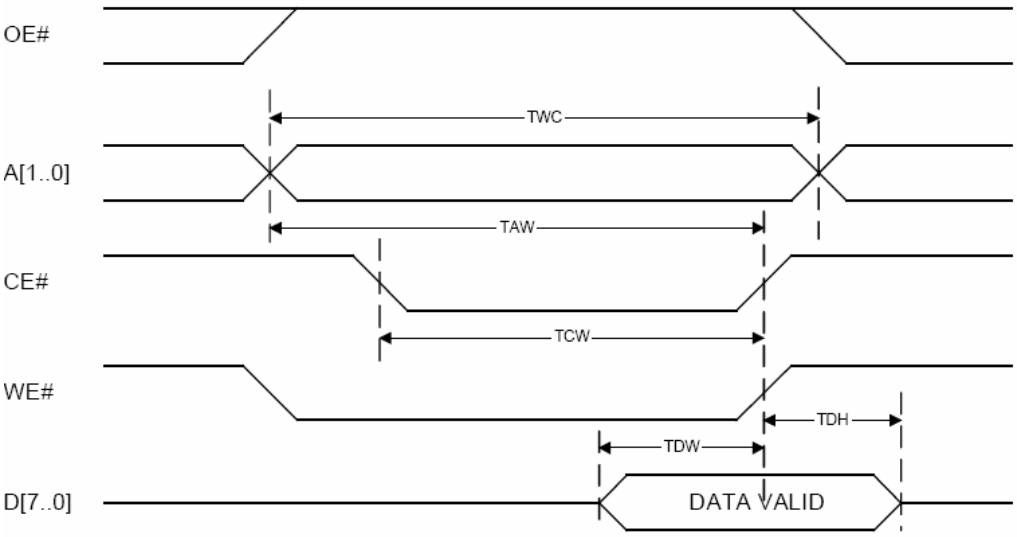
7.3.2.2 写周期时间特性

Symbol	Parameter	Min.	Max.	Unit
TWC	Write Cycle Time	70		ns
TAW	Address Valid to End-of-Write	60		ns

TCW	Chip Select to End-of-Write	35		ns
TWP	Write Pulse Width	35		ns
TDW	Dada to Write Time Overlap	30		ns
TDH	Data Hold from Write Time	5		ns



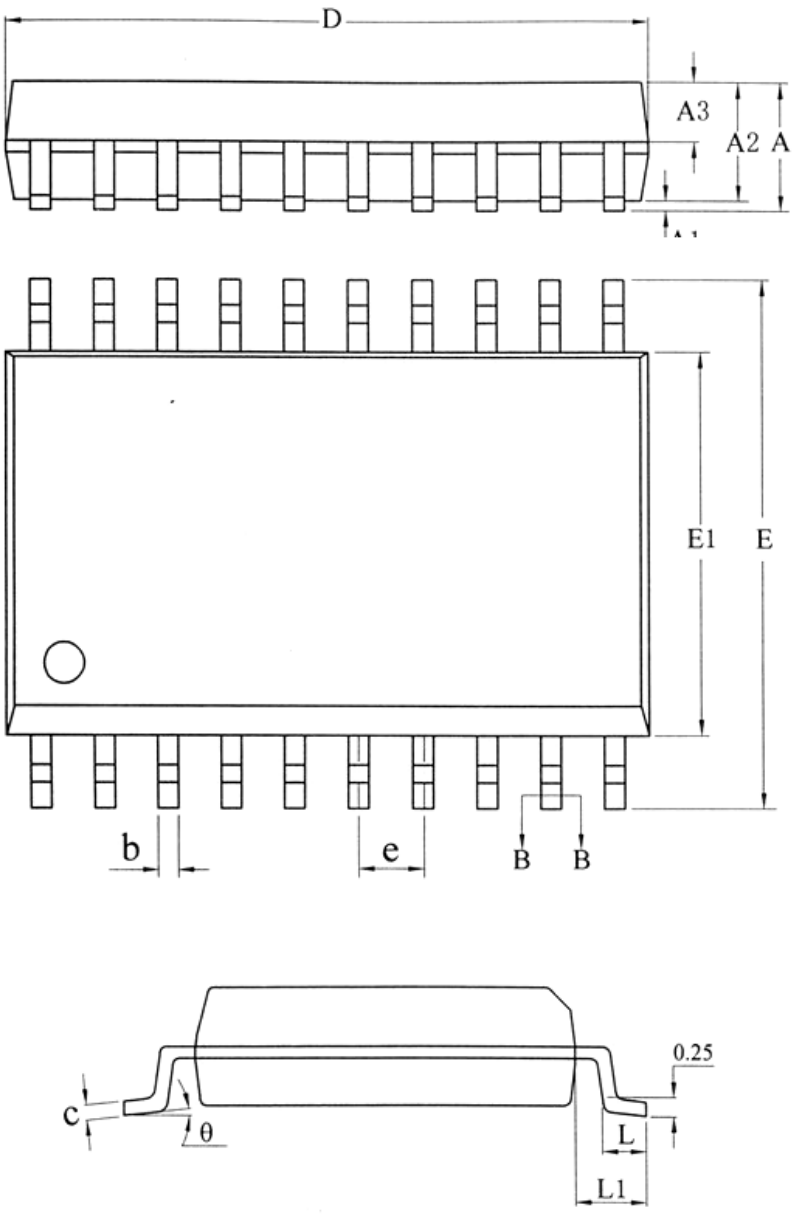
写周期时序波形图（WE#控制的时序）



写周期时序波形图（CE#控制的时序）

8 封装尺寸：SOP20

单位: mm



SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	—	—	2.70
A1	0.10	0.20	0.30
A2	2.10	2.30	2.50
A3	0.92	1.02	1.12
b	0.35	—	0.44
b1	0.34	0.37	0.39
c	0.26	—	0.31
c1	0.24	0.25	0.26
D	12.60	12.80	13.00
E	10.10	10.30	10.50
E1	7.30	7.50	7.70
e	1.27BSC		
L	0.70	0.85	1.00
L1	1.40BSC		
θ	0	—	8°

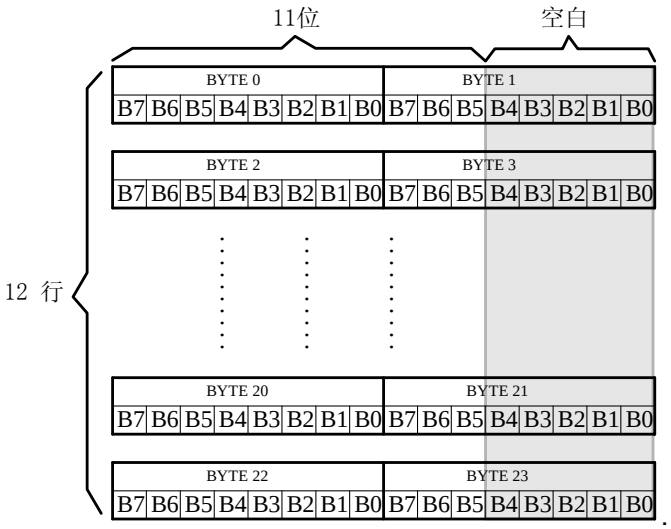
9 功能描述

9.1 汉字点阵排列格式

每个汉字在芯片中是以汉字点阵字模的形式存储的，每个点用一个二进制位表示，存 1 的点，当显示时可以在屏幕上显示亮点，存 0 的点，则在屏幕上不显示。点阵排列格式为横置横排：即一个字节的高位表示左面的点，低位表示右面的点（如果用户按 word mode 读取点阵数据，请注意高低字节的顺序），排满一行的点后再排下一行。这样把点阵信息用来直接在显示器上按上述规则显示，则将出现对应的汉字。

9.1.1 11X12 点汉字排列格式

11X12 点汉字的信息需要 24 个字节（BYTE 0 – BYTE 23）来表示。该 11X12 点汉字的点阵数据是横置横排的，其具体排列结构如下图：.

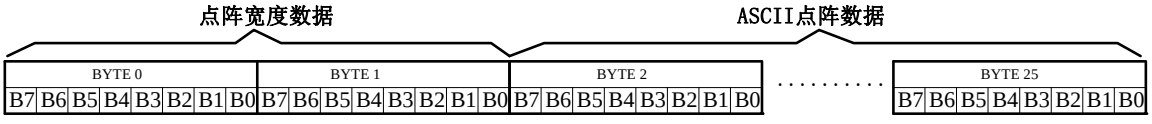


9.1.2 12 点阵 ASCII 方头字符排列格式

12 点阵 ASCII 方头字符的信息需要 26 个字节（BYTE 0 – BYTE25）来表示。

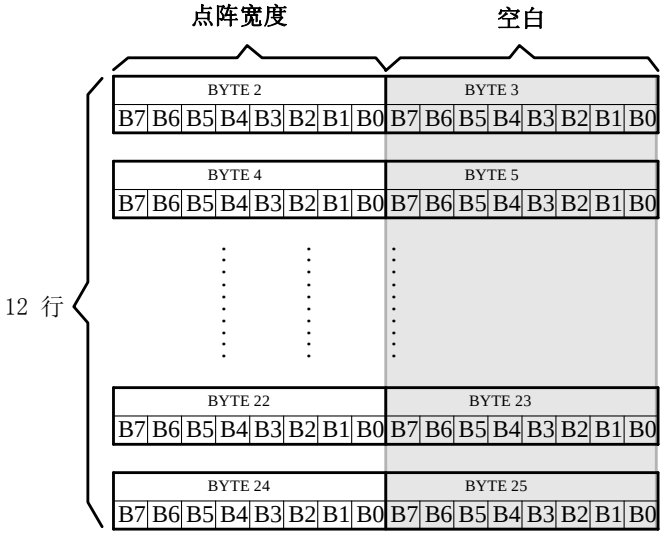
■ 存储格式

由于 ASCII 方头字符是不等宽的，因此在存储格式中 BYTE0~ BYTE1 存放点阵宽度数据，BYTE2-25 存放点阵数据。具体格式见下图：



■ 存储结构

不等宽 ASCII 字符的存储结构是以宽度为 BYTE 取整的，根据不同字符宽度会出现相应的空白区。根据 BYTE0~ BYTE1 所存放点阵的宽度数据，可以对还原下一个字的显示或排版留作参考。



12 点阵 ASCII 方头字符排列格式图

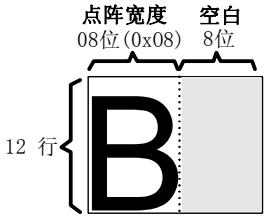
例如：ASCII 方头字符 B

0-25BYTE 的点阵数据是: 00 08 00 00 00 00 00 00 FC 00 C6 00 C6 00 C6 00 FC 00 C6 00 C6 00 C6 00 FC 00

其中：

BYTE0~ BYTE1: 00 08

为 ASCII 方头字符 B 的点阵宽度数据 00 08，即：8 位宽度。字符后面有 8 位空白区，可以在排版下一个字时考虑到这一点，将下一个字的起始位置前移。（见下图）



BYTE2-25: 00 00 00 00 00 00 FC 00 C6 00 C6 00 C6 00 FC 00 C6 00 C6 00 C6 00 FC 00

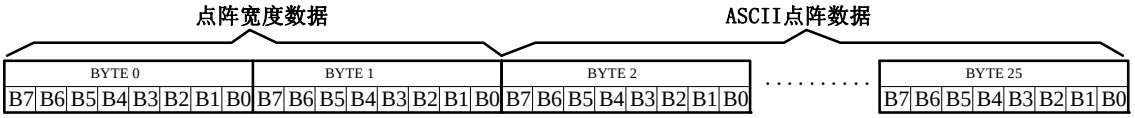
为 ASCII 方头字符 B 的点阵数据。

9.1.3 12 点阵 ASCII 白正字符排列格式

12 点阵 ASCII 白正字符的信息需要 26 个字节（BYTE 0 – BYTE25）来表示。

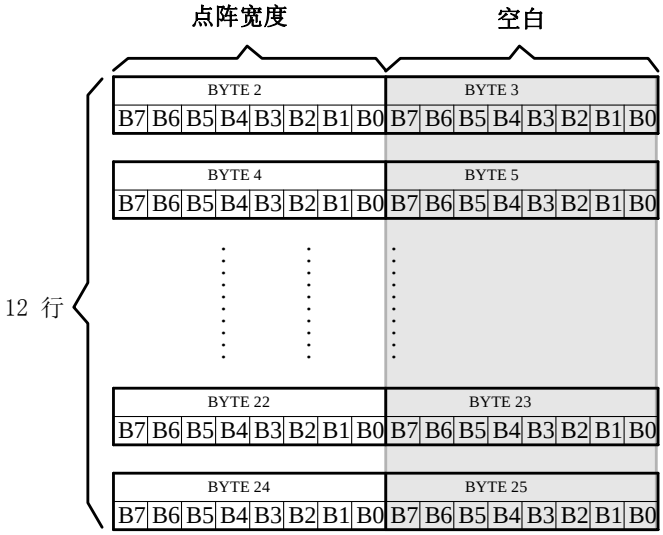
■ 存储格式

由于 ASCII 白正字符是不等宽的，因此在存储格式中 BYTE0~ BYTE1 存放点阵宽度数据，BYTE2-25 存放点阵数据。具体格式见下图：



■ 存储结构

不等宽 ASCII 字符的存储结构是以宽度为 BYTE 取整的，根据不同字符宽度会出现相应的空白区。根据 BYTE0~ BYTE1 所存放点阵的宽度数据，可以对还原下一个字的显示或排版留作参考。



16 点阵 ASCII 白正字符排列格式图

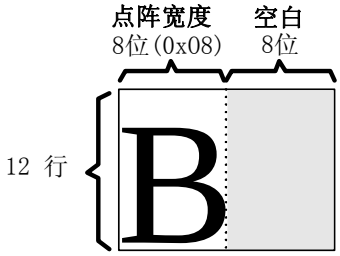
例如：ASCII 白正字符 B

0-25BYTE 的点阵数据是: 00 08 00 00 00 00 00 00 F8 00 44 00 44 00 44 00 7C 00 42 00 42 00 42 00 FC 00

其中：

BYTE0~ BYTE1: 00 08

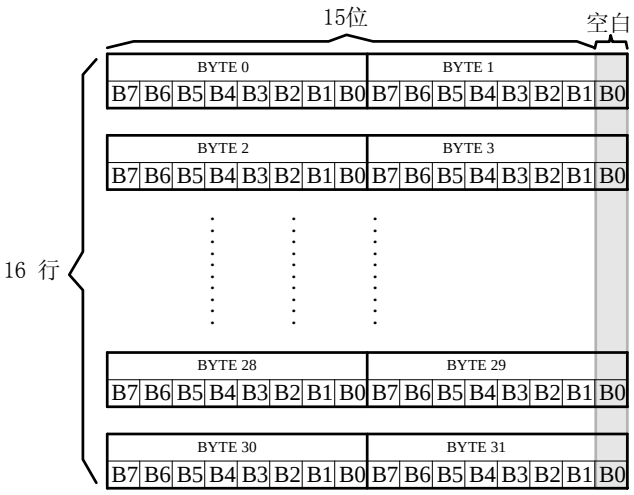
为 ASCII 白正字符 B 的点阵宽度数据，即：8 位宽度。字符后面有 8 位空白区，可以在排版下一个字时考虑到这一点，将下一个字的起始位置前移。（见下图）



BYTE2-25: 00 00 00 00 00 00 F8 00 44 00 44 00 44 00 7C 00 42 00 42 00 42 00 FC 00
为 ASCII 白正字符 B 的点阵数据。

9.1.4 15X16 点汉字排列格式

15X16 点汉字的信息需要 32 个字节（BYTE 0 – BYTE 31）来表示。该 15X16 点汉字的点阵数据是横置横排的，其具体排列结构如下图：



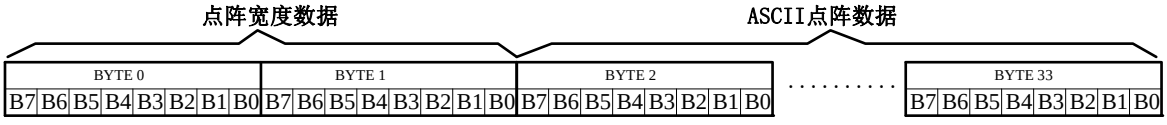
15X16 点汉字排列格式图

9.1.5 16 点阵 ASCII 方头字符排列格式

16 点阵 ASCII 方头字符的信息需要 34 个字节（BYTE 0 – BYTE33）来表示。

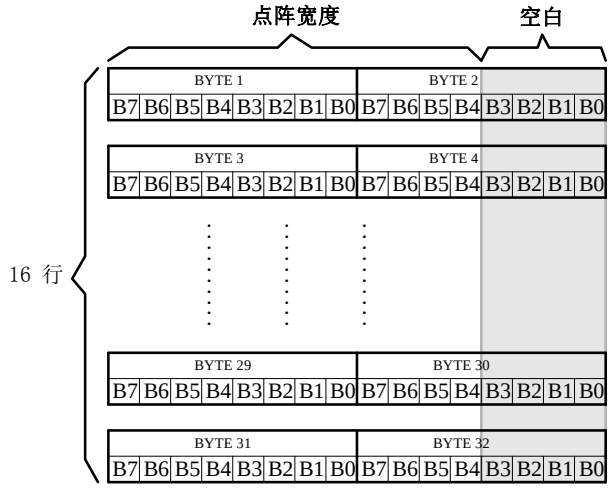
■ 存储格式

由于 ASCII 方头字符是不等宽的，因此在存储格式中 BYTE0~ BYTE1 存放点阵宽度数据，BYTE2-33 存放点阵数据。具体格式见下图：



■ 存储结构

不等宽 ASCII 字符的存储结构是以宽度为 BYTE 取整的，根据不同字符宽度会出现相应的空白区。根据 BYTE0~ BYTE1 所存放点阵的宽度数据，可以对还原下一个字的显示或排版留作参考。



16 点阵 ASCII 方头字符排列格式图

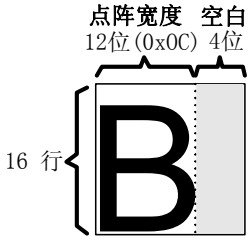
例如：ASCII 方头字符 B

0-33BYTE 的点阵数据是: 00 0C 00 00 00 00 00 00 00 00 7F 80 7F C0 60 C0 60 C0 60 C0 7F 80 7F C0 60 E0 60 60 60 60 7FC0 7F 80

其中：

BYTE0~ BYTE1: 00 0C

为 ASCII 方头字符 B 的点阵宽度数据，即：12 位宽度。字符后面有 4 位空白区，可以在排版下一个字时考虑到这一点，将下一个字的起始位置前移。（见下图）



BYTE2-33: 00 00 00 00 00 00 00 00 7F 80 7F C0 60 C0 60 C0 60 C0 7F 80 7F C0 60 E0 60 60 60 60 7FC0 7F 80

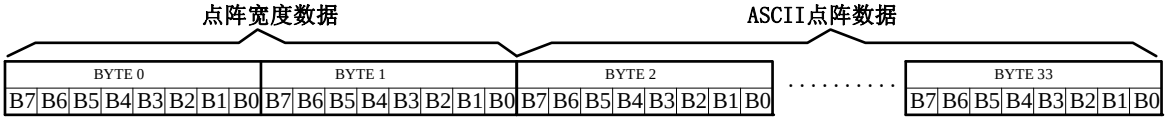
为 ASCII 方头字符 B 的点阵数据。

9.1.6 16 点阵 ASCII 白正字符排列格式

16 点阵 ASCII 白正字符的信息需要 34 个字节（BYTE 0 – BYTE33）来表示。

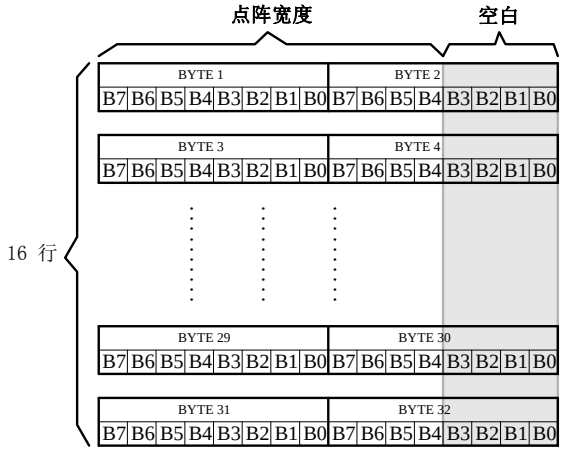
■ 存储格式

由于 ASCII 白正字符是不等宽的，因此在存储格式中 BYTE0~ BYTE1 存放点阵宽度数据，BYTE2-33 存放点阵数据。具体格式见下图：



■ 存储结构

不等宽 ASCII 字符的存储结构是以宽度为 BYTE 取整的，根据不同字符宽度会出现相应的空白区。根据 BYTE0~ BYTE1 所存放点阵的宽度数据，可以对还原下一个字的显示或排版留作参考。



16 点阵 ASCII 白正字符排列格式图

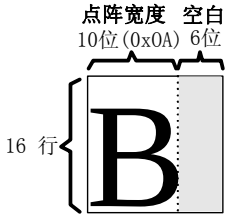
例如：ASCII 白正字符 B

0-33BYTE 的点阵数据是: 00 0A 00 00 00 00 00 00 00 00 FC 00 22 00 21 00 21 00 22 00 3E 00 21 00 20 80 20 80 21 00 FE 00 00 00

其中：

BYTE0~ BYTE1: 00 0A

为 ASCII 白正字符 B 的点阵宽度数据，即：10 位宽度。字符后面有 6 位空白区，可以在排版下一个字时考虑到这一点，将下一个字的起始位置前移。（见下图）

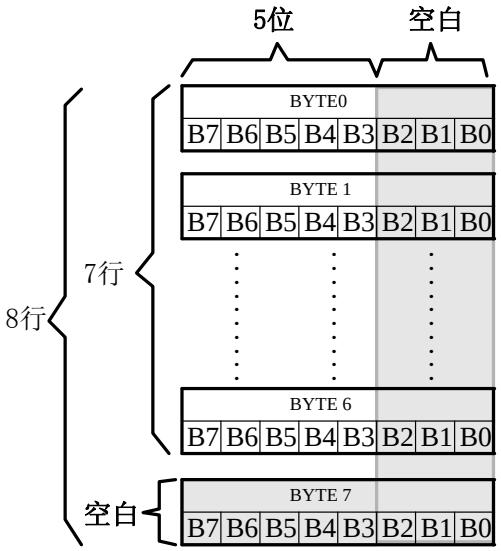


BYTE2-33: 00 00 00 00 00 00 00 00 FC 00 22 00 21 00 21 00 22 00 3E 00 21 00 20 80 20 80 21 00 FE 00 00 00

为 ASCII 白正字符 B 的点阵数据。

9.1.7 5X7 点 ASCII 字符排列格式

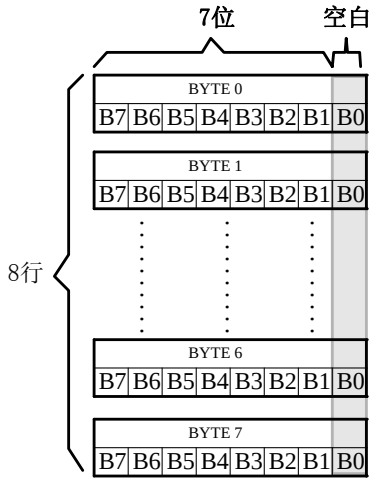
5X7 点 ASCII 的信息需要 8 个字节（BYTE 0 – BYTE7）来表示。该 ASCII 点阵数据是横置横排的，其具体排列结构如下图：



5X7 点 ASCII 字符排列格式图

9.1.8 7X8 点 ASCII 字符排列格式

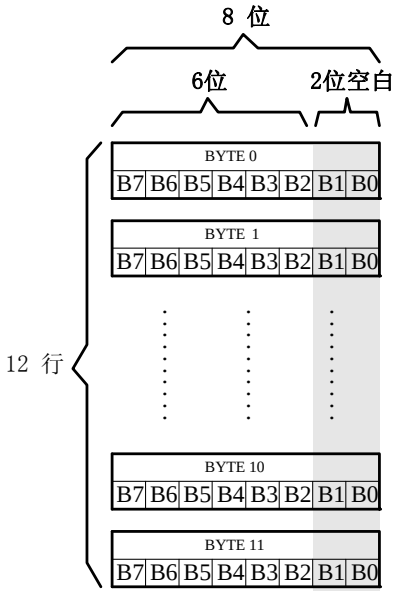
7X8 点 ASCII 的信息需要 8 个字节（BYTE 0 – BYTE7）来表示。该 ASCII 点阵数据是横置横排的，其具体排列结构如下图：



7X8 点 ASCII 字符排列格式图

9.1.9 6X12 点 ASCII 扩展字符排列格式

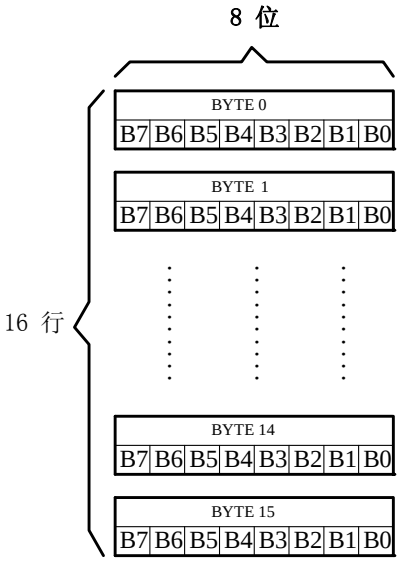
6X12 点 ASCII 的信息需要 12 个字节（BYTE 0 – BYTE12）来表示。



6X12 点 ASCII 字符排列格式图

9.1.10 8X16 点 ASCII 扩展字符排列格式

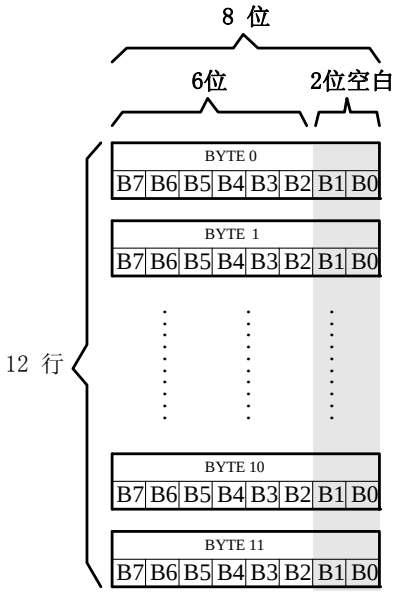
8X16 点 ASCII 的信息需要 16 个字节（BYTE 0 – BYTE15）来表示。



8X16 点 ASCII 字符排列格式图

9.1.11 6X12 点国际音标字符排列格式

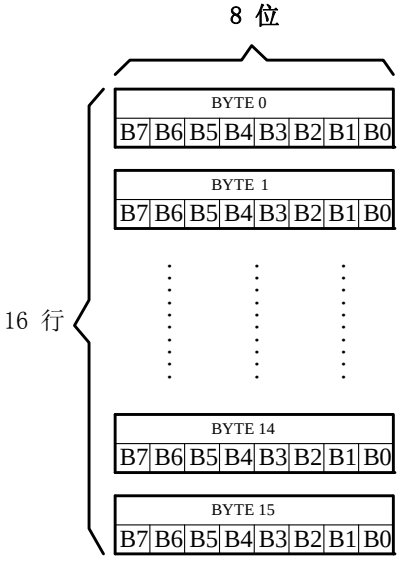
6X12 点 ASCII 的信息需要 12 个字节（BYTE 0 – BYTE12）来表示。



8X16 点 ASCII 字符排列格式图

9.1.12 8X16 点国际音标字符排列格式

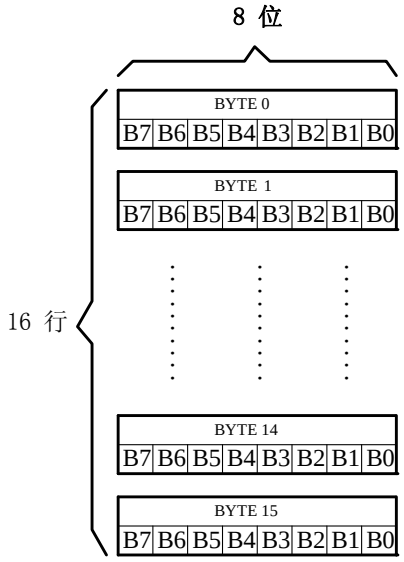
8X16 点 ASCII 的信息需要 16 个字节（BYTE 0 – BYTE15）来表示。



8X16 点 ASCII 字符排列格式图

9.1.13 8X16 点 Unicode 字符（拉丁语、希腊语、西里尔语）及自定义字符区排列格式

8X16 点 ASCII 的信息需要 16 个字节（BYTE 0 – BYTE15）来表示。

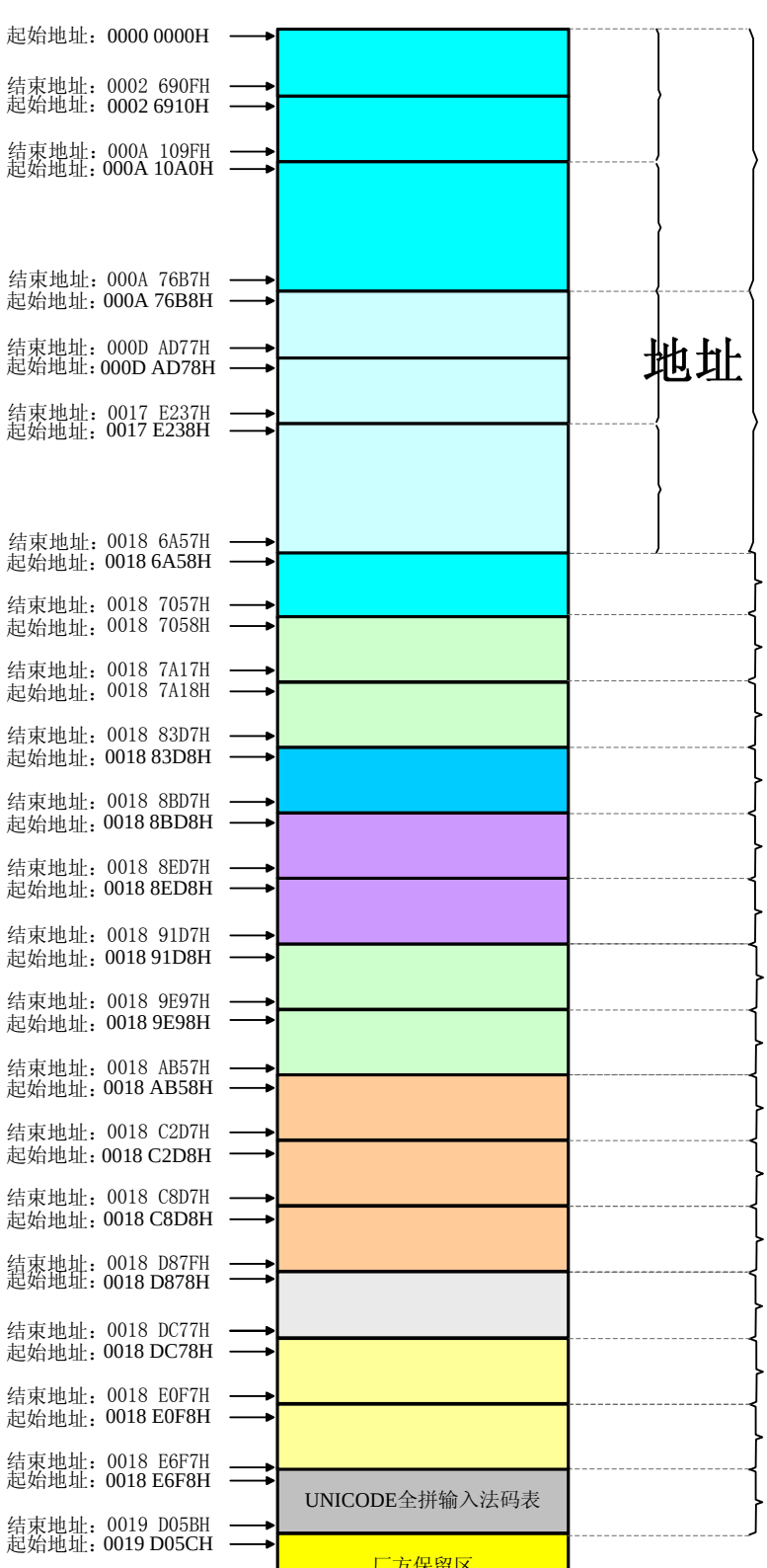


8X16 点 ASCII 字符排列格式图

9.2 汉字点阵字库结构

- UNICODE3.0 标准点阵字库
  点阵字库中的 1 区对应 UNICODE3.0 扩展集码位的 3400~4DB5 共计 6582 个汉字；
  点阵字库中的 2 区对应 UNICODE3.0 基本集码位的 4E00~9FA5 共计 20902 个汉字；
  点阵字库中的 3 区对应 UNICODE3.0 符号码位的 00A1~33D5、E76C~FFE0 共计 985 个字符。
- 扩展字符对应 GB2312 码位的 AAA1~ABC0 共计 126 个字符。
- ASCII 方头为不等宽 ASCII，所对应码位 20~7E 共计 96 个字符。
- ASCII 白正为不等宽 ASCII，所对应码位 20~7E 共计 96 个字符。
- 5X7 点 ASCII 字符对应 ASCII 码位的 20~7E 共计 96 个字符。
- 7X8 点 ASCII 字符对应 ASCII 码位的 20~7E 共计 96 个字符。
- 1 套 6X12 点扩展字符对应 GB18030 码位的 AAA1~ABC0 共计 126 个字符。
- 1 套 8X16 点 ASCII 扩展字符集合 对应内码 AAA1~ABC0 共计 126 个字符 (或者对应 ASCII 码位 20~7E 共计 96 个字符,其余忽略) 。
- 8X16 点自定义特殊字符区对应码位的 ACA1~ACDF 共计 64 个字符。
- 8X16 Unicode 字符-拉丁文系:对应拉丁文系 Unicode 码 00A0H~04E9H 共计 376 个字符
- 8X16 Unicode 字符-希腊文系: 对应希腊文系 Unicode 码 0370H~03CFH 共计 96 个字符。
- 8X16 Unicode 字符-西里尔文系:对应西里尔文系 Unicode 码 0400H~04F9H 共计 250 个字符。

GT23L16U2W3　字库内容说明



9.3 汉字点阵在芯片中的地址计算方法

用户只要知道汉字的内码，就可以计算出该汉字点阵在芯片中的地址，然后就可从该地址连续读出点阵信息用于显示。

11X12 点阵字库地址分配(字节地址)：0000 0000 H ~ 0000 C9CF H ;

15X16 点阵字库地址分配(字节地址)：0002 C9D0 H ~ 0006 818F H ;

9.3.1 标准点阵字库汉字与字符地址的计算

GBCode 表示 汉字内码。

MSB 表示汉字内码 GBCode 的高 8bits。

LSB 表示汉字内码 GBCode 的低 8bits。

Address 表示汉字或 ASCII 字符点阵在芯片中的字节地址。

BaseAdd：说明点阵数据在字库芯片中的起始地址。

(1) 11x12 汉字点阵在芯片中的字节地址的计算方法如下：

Ucode 表示 汉字内码

MSB 表示汉字内码 Ucode 的高 8bits。

LSB 表示汉字内码 Ucode 的低 8bits。

Address 表示汉字或 ASCII 字符点阵在芯片中的字节地址。

```
BaseAdd=0x0000 ;
if(Ucode >=0x3400 && Ucode <= 0x4DB5)          //UNICODE3.0 扩展汉字区 6582 个汉字
    Address =(unicode-0x3400)*24;
else if(Ucode >=0x4E00 && Ucode <= 0x9FA5)    //UNICODE3.0 字符区 20902 个汉字
    Address =(unicode-0x4E00+6582)*24;
```

(2) 15x16 汉字点阵在芯片中的字节地址的计算方法如下：

Ucode 表示 汉字内码

MSB 表示汉字内码 Ucode 的高 8bits。

LSB 表示汉字内码 Ucode 的低 8bits。

Address 表示汉字或 ASCII 字符点阵在芯片中的字节地址。

```
BaseAdd=0x0A76B8 ;
if(Ucode >=0x3400 && Ucode <= 0x4DB5)          //UNICODE3.0 扩展汉字区 6582 个汉字
    Address =(unicode-0x3400)*32+ BaseAdd;
else if(Ucode >=0x4E00 && Ucode <= 0x9FA5)    //UNICODE3.0 字符区 20902 个汉字
    Address =(unicode-0x4E00+6582)*32+ BaseAdd;
```

9.3.2 字符点阵在芯片中的地址计算方法

(1) 11x12 汉字点阵 ASCII 基本字符在 UNICODE3.0 字符区中的地址计算

```
BaseAdd=0x0000 ;
if(Ucode >=0xFF00 && Ucode <= 0xFF5E || Ucode >=0x20 && Ucode <= 0x7E )
{    if(Ucode ==0xFF00 || Ucode == 0x20)              //空格
         Address =（27484+538）*24+ BaseAdd;
     else if(Ucode >0xFF00 && Ucode < 0xFF5E)
         Address = (Ucode -0xFF00+27484+987)*24+ BaseAdd;
     else if(Ucode >0x20 && Ucode < 0x7E )
         Address = (Ucode -0x20+27484+987)*24+ BaseAdd;
}
```

(2) 11x12 汉字点阵 UNICODE3.0 字符区的地址计算

由于该字符区的 UNICODE 内码是离散的。因此需要查表后，根据该码在字符区中的序号计算其地址。（该内码表见：**附件 9.7**）

Ucode 表示 汉字内码

Address 表示 汉字或 ASCII 字符点阵在芯片中的字节地址。

ZFindex 表示：根据附件 9.1 得到的查表函数为 WORD ZFindex(WORD Ucode)，返回在表中的序号;

芯片中的字节地址的计算方法如下：

```
if (Ucode>=00A1&& Ucode <=33D5 || Ucode>= E76C && Ucode <= FFE5)   //字符区范围
     Address = ZFindex(Ucode)*32+27484*32+ BaseAdd;
```

(3) 15x16 汉字点阵 ASCII 基本字符在 UNICODE3.0 字符区中的地址计算

```
BaseAdd=0x0A76B8 ;
if(Ucode >=0xFF00 && Ucode <= 0xFF5E || Ucode >=0x20 && Ucode <= 0x7E )
{    if(Ucode ==0xFF00 || Ucode == 0x20)              //空格
         Address =（27484+538）*32+ BaseAdd;
     else if(Ucode >0xFF00 && Ucode < 0xFF5E)
         Address = (Ucode -0xFF00+27484+987)*32+ BaseAdd;
     else if(Ucode >0x20 && Ucode < 0x7E )
         Address = (Ucode -0x20+27484+987)*32+ BaseAdd;
}
```

(4) 15x16 汉字点阵 UNICODE3.0 字符区的地址计算

由于该字符区的 UNICODE 内码是离散的。因此需要查表后，根据该码在字符区中的序号计算其地址。（该内码表见：**附件**）

Ucode 表示 汉字内码

Address 表示 汉字或 ASCII 字符点阵在芯片中的字节地址。

ZFindex 表示：根据附件 9.1 得到的查表函数为 WORD ZFindex(WORD Ucode)，返回在表中的序号;

芯片中的字节地址的计算方法如下：

```
BaseAdd=0x0A76B8 ;
if (Ucode>=00A1&& Ucode <=33D5 || Ucode>= E76C && Ucode <= FFE5)   //字符区范围
     Address = ZFindex(Ucode)*32+27484*32+ BaseAdd;
```

9.3.3 扩展字符地址的计算

GBCode 表示 汉字内码。
MSB 表示汉字内码 GBCode 的高 8bits。
LSB 表示汉字内码 GBCode 的低 8bits。
Address 表示汉字或 ASCII 字符点阵在芯片中的字节地址。
BaseAdd：说明点阵数据在字库芯片中的起始地址。

(1) 6X12 点扩展字符在芯片中的字节地址的计算方法如下：

BaseAdd=0x 186A58;
if (MSB >= 0x AAH&& LSB>=0x A1 && LSB<=0x FE) then //内码为 AAA1~AAFE 的 94 个扩展字符
Address = (LSB–0x A1) * 24+ BaseAdd
if (MSB >= ABH && LSB>=A1 && LSB<=C0) then //内码为 ABA1~ABC0 的 32 个扩展字符
Address = (LSB–0x A1 +94) * 24+ BaseAdd

(2) 8X16 点扩展字符在芯片中的字节地址的计算方法如下：

BaseAdd=0x 1883D8 ;
if (MSB >= 0x AA && LSB>=0x A1 && LSB<=0x FE) then //内码为 AAA1~AAFE 的 94 个扩展字符
Address = (LSB–0x A1) * 32+ BaseAdd
if (MSB >= ABH && LSB>=A1 && LSB<=C0) then //内码为 ABA1~ABC0 的 32 个扩展字符
Address = (LSB–0x A1 +94) * 32+ BaseAdd

9.3.4 ASCII 地址的计算

ASCIICode 表示 ASCII 码（8bits）
MSB 表示汉字内码 GBCode 的高 8bits。
LSB 表示汉字内码 GBCode 的低 8bits。
Address 表示汉字或 ASCII 字符点阵在芯片中的字节地址。
BaseAdd：说明点阵数据在字库芯片中的起始地址。

(1) 5X7 ASCII 字符点阵在芯片中的字节地址的计算方法如下：

BaseAdd=0x 188BD8
if (ASCIICode >= 0x 20) and (ASCIICode <= 0x 7E) then
Address = (ASCIICode –0x 20) * 8+ BaseAdd

(2) 7X8 ASCII 字符点阵在芯片中的字节地址的计算方法如下：

BaseAdd=0x 188ED8
if (ASCIICode >= 0x 20) and (ASCIICode <= 0x 7E) then
Address = (ASCIICode –0x 20) * 8+ BaseAdd

(3) 12 点阵 ASCII 方头字符在芯片中的字节地址的计算方法如下：

BaseAdd=0X187058
if (ASCIICode >= 0x 20) and (ASCIICode <= 0x 7E) then
Address = (ASCIICode –0x 20) * 26+ BaseAdd

(4) 12 点阵 ASCII 白正字符在芯片中的字节地址的计算方法如下：

BaseAdd=0x187A18

if (ASCIICode >= 0x 20) and (ASCIICode <= 0x 7E) then

Address = (ASCIICode –0x 20) * 26+ BaseAdd

(5) 16 点阵 ASCII 方头字符在芯片中的字节地址的计算方法如下：

BaseAdd=0x1891D8

if (ASCIICode >= 0x 20) and (ASCIICode <= 0x 7E) then

Address = (ASCIICode –0x 20) * 34+ BaseAdd

(6) 16 点阵 ASCII 白正字符在芯片中的字节地址的计算方法如下：

BaseAdd=0x 189E98

if (ASCIICode >= 0x 20) and (ASCIICode <= 0x 7E) then

Address = (ASCIICode –0x 20) * 34+ BaseAdd

9.3.5 8X16 自定义特殊字符地址的计算

8X16 自定义特殊字符计算方法如下：

GBCODE 表示 GB 内码（16bits）

Address 表示字符点阵在芯片中的字节地址。

BaseAdd: 说明点阵数据在字库芯片中的起始地址。

BaseAdd=0x 18D878

if (GBCODE > 0x ACA0) and (GBCODE <=0x ACDF) then

Address = (GBCODE –0x ACA0) * 16+BaseAdd

9.3.6 国际音标字符地址的计算

(1) 6X12 国际音标字符字符计算方法如下：

CODE 表示 国际音标内码（16bits）

Address 表示字符点阵在芯片中的字节地址。

BaseAdd: 说明点阵数据在字库芯片中的起始地址。

BaseAdd=0x 18DC78

if (CODE >= 0x20) and (GBCODE <=0x 7E) then

Address = (CODE –0x 20) * 12+BaseAdd

(2) 8X16 国际音标字符字符计算方法如下：

CODE 表示 国际音标内码（16bits）

Address 表示字符点阵在芯片中的字节地址。

BaseAdd: 说明点阵数据在字库芯片中的起始地址。

BaseAdd=0x 18E0F8
if (CODE >= 0x20) and (GBCODE <=0x 7E) then
Address = (CODE –0x 20) * 32+BaseAdd

9.3.7 8X16　拉丁语、西里尔语 Unicode 字符在芯片中的字节地址的计算方法

计算方法如下：
unicode表示unicode内码。
Address 表示unicode字符点阵在芯片中的字节地址。
BaseAdd：说明点阵数据在字库芯片中的起始地址。

1.字节地址的计算函数调用方法－拉丁文系

BaseAdd= 0x 18AB58
if (Unicode > = 0x 00A0) and (unicode <=0x 0217) then
Address = (unicode –0x 00A0) * 16+BaseAdd

2.字节地址的计算函数调用方法－希腊文系

BaseAdd= 0x 18C2D8
if (Unicode > = 0x 0370) and (unicode <=0x 03CF) then
Address = (unicode –0x 0370) * 16+BaseAdd

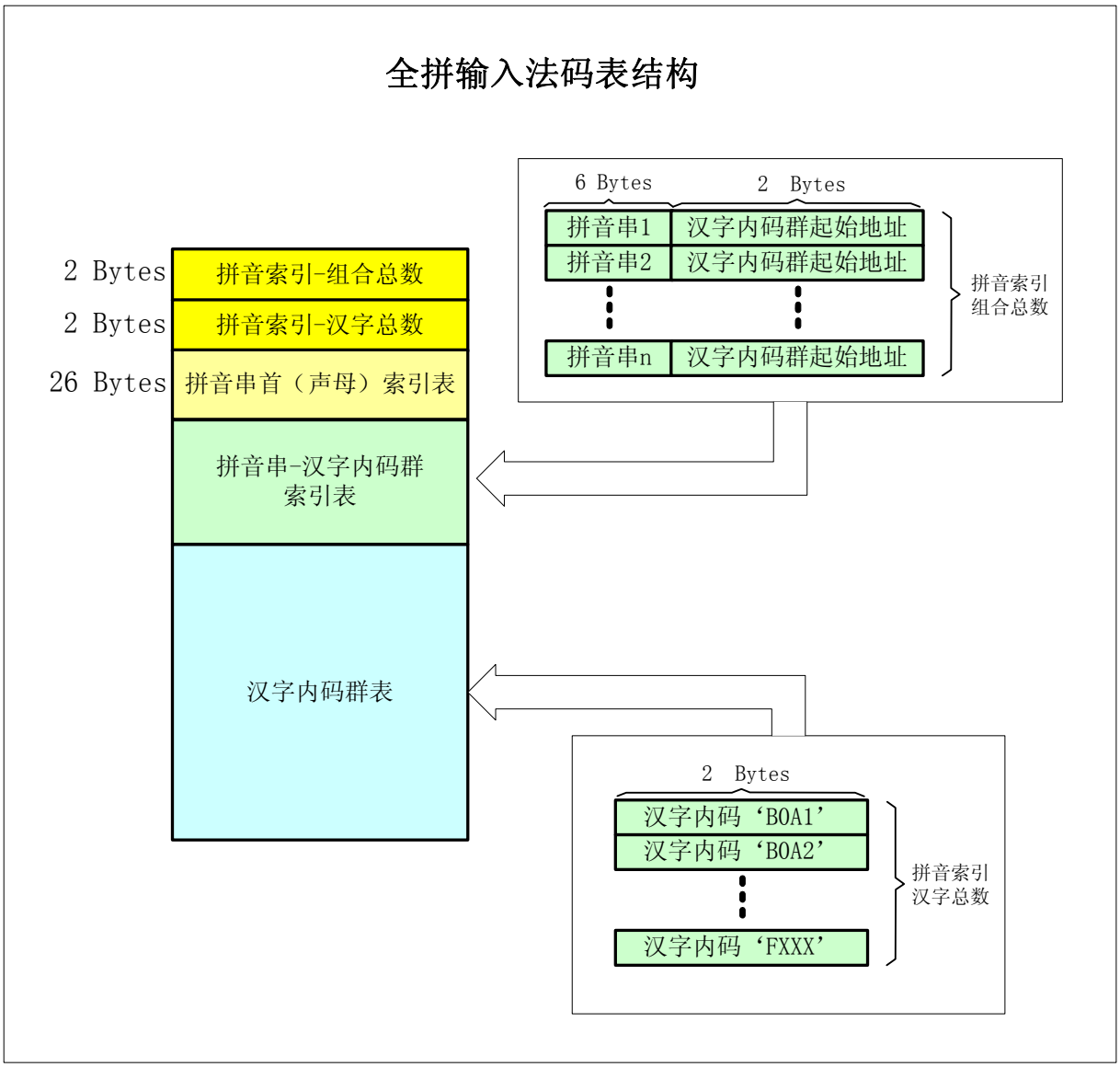
3.字节地址的计算函数调用方法－西里尔文系

BaseAdd= 0x 18C8D8
if (Unicode > = 0x 0400) and (unicode <=0x 04F9) then
Address = (unicode –0x 0400) * 16+BaseAdd

9.4 全拼拼音－UNICODE 全拼输入法（21003 字）

本字库芯片含有 UNICODE 全拼输入法码表（21003 字），只要将键值串相应的拼音 ASCII 字符进行索引，即可得到该键值串所对应的汉字内码群，产品开发时可以根据用户的需要进行汉字输入的操作。

9.4.1 全拼输入法码表结构



全拼输入法码表结构说明：

1. 拼音索引－组合总数：是全拼输入法码表的所有拼音组合的总数，即声母＋韵母的所有汉字组合。
2. 拼音索引－汉字总数：是本输入法能够支持拼音拼写汉字的总数。
3. 拼音串首（声母）索引表：根据输入拼音串的字首声母来定位该声母在“拼音串－汉字内码群”中该声母的起始地址，主要目的是为了加快查找表速度。
4. 拼音串－汉字内码群：就是根据输入的拼音串进行查找，并找到相应的“汉字内码群“起始地址。
5. 汉字内码群表：该拼音串所对应的汉字内码群。

9.4.2 算法举例：

1. 拼音输入法码表的存储格式

为了方便结构的理解结合 8.4.1 图例和以下 ANSI C 语言对结构进行描述。

//存储格式:

short int pycount；//拼音索引－组合总数 2byte

short int hzcount；//拼音索引－汉字总数 2byte

short int pyidx[26]; //拼音串首（声母）索引表，以 26 个英文字母开头的拼音串在拼音表中的起始地址

```
struct {
        char py[6] ;//拼音串
        short int   //汉字码表的起始地址
}
.....................
short int hzcode;       //汉字 1 内码
short int hzcode;       //汉字 2 内码
.....................
```

注: 起始地址已转成绝对地址，不用再计算

2. 拼音输入法的索引

假设拼音串都为小写字母. 输入拼音串 **bang**

1. 用拼音串第一个字母 b-'a'得出拼音串索引下标 1。(检索拼音串首索引表并得到拼音串－汉字内码群地址->idx1)
2. 读出 pyidx[1]的值->变量 idx1。(若 idx1 为 0 说明无此字母开头的拼音串,如 i,u,v 开头，返回)
3. 由 idx1 开始寻址，读出拼音码表,直到找到拼音串 bang。(检索"拼音串－汉字内码群"，并得到汉字内码群表地址。当寻址地址>=pycount*8+56 时无此拼音组合)
   同时读出下一个拼音串。
4. 由找到的拼音串结构,确定汉字码表的地址->idx2，下一个拼音串结构所指汉字码表地址->idx3（下一个拼音串之前的都是该拼音对应的汉字内码区间）
5. 由 idx2 寻址到"汉字内码群表"，查找该拼音串所对应的汉字内码群,同时地址<idx3 的所有汉字均为满足 bang 拼音的汉字 。
6. 当地址>=pycount*8+56+hzcount*2 时，已寻址到码表尾。说明该拼音串不在本输入法码表的范围之内。

10 附录

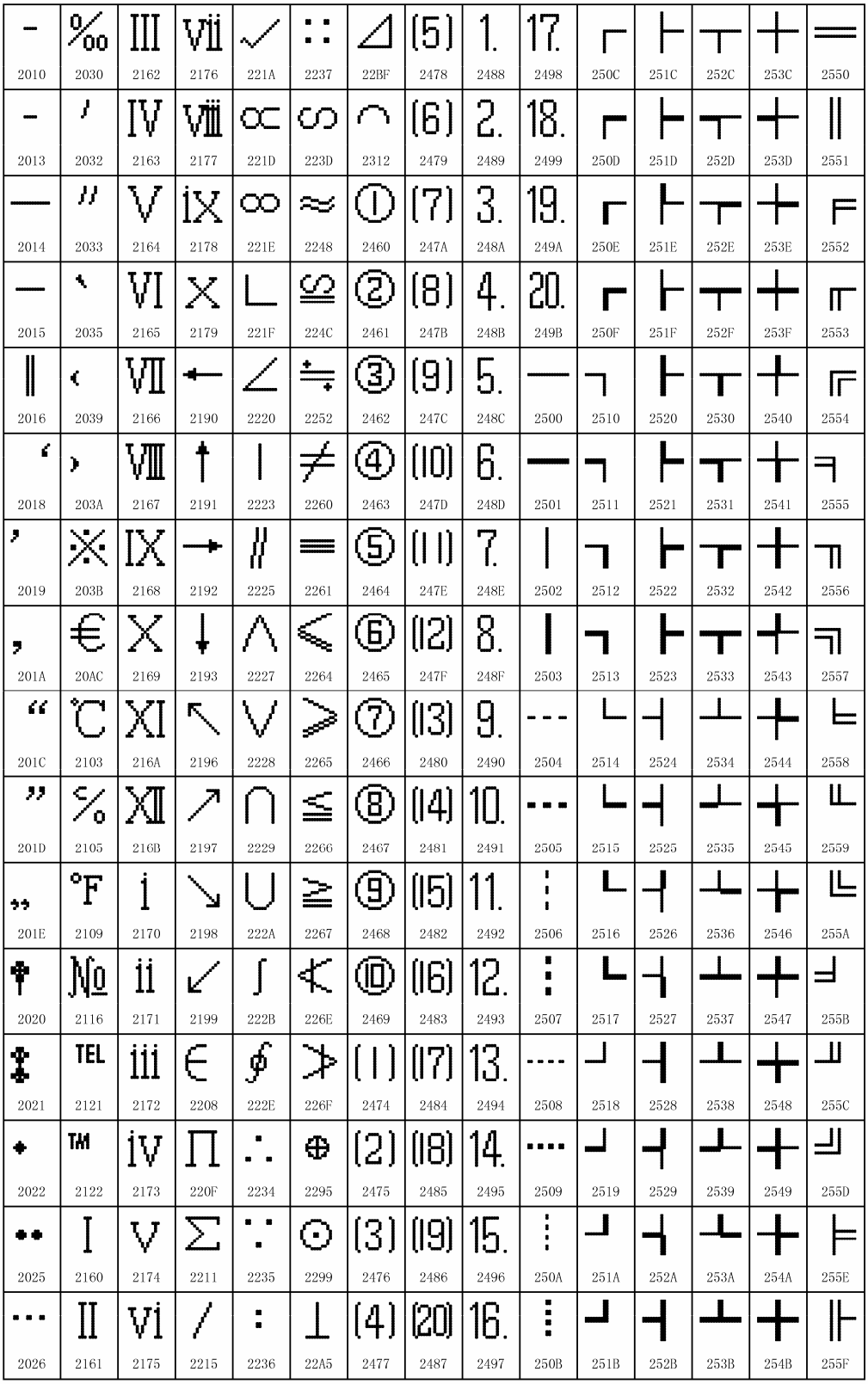
10.1 UNICODE3.0（GB13000）字符区

UNICODE 字符区码位由 00A1~33D5、E76C~FFE5 离散分布。共计 1088 个字符；

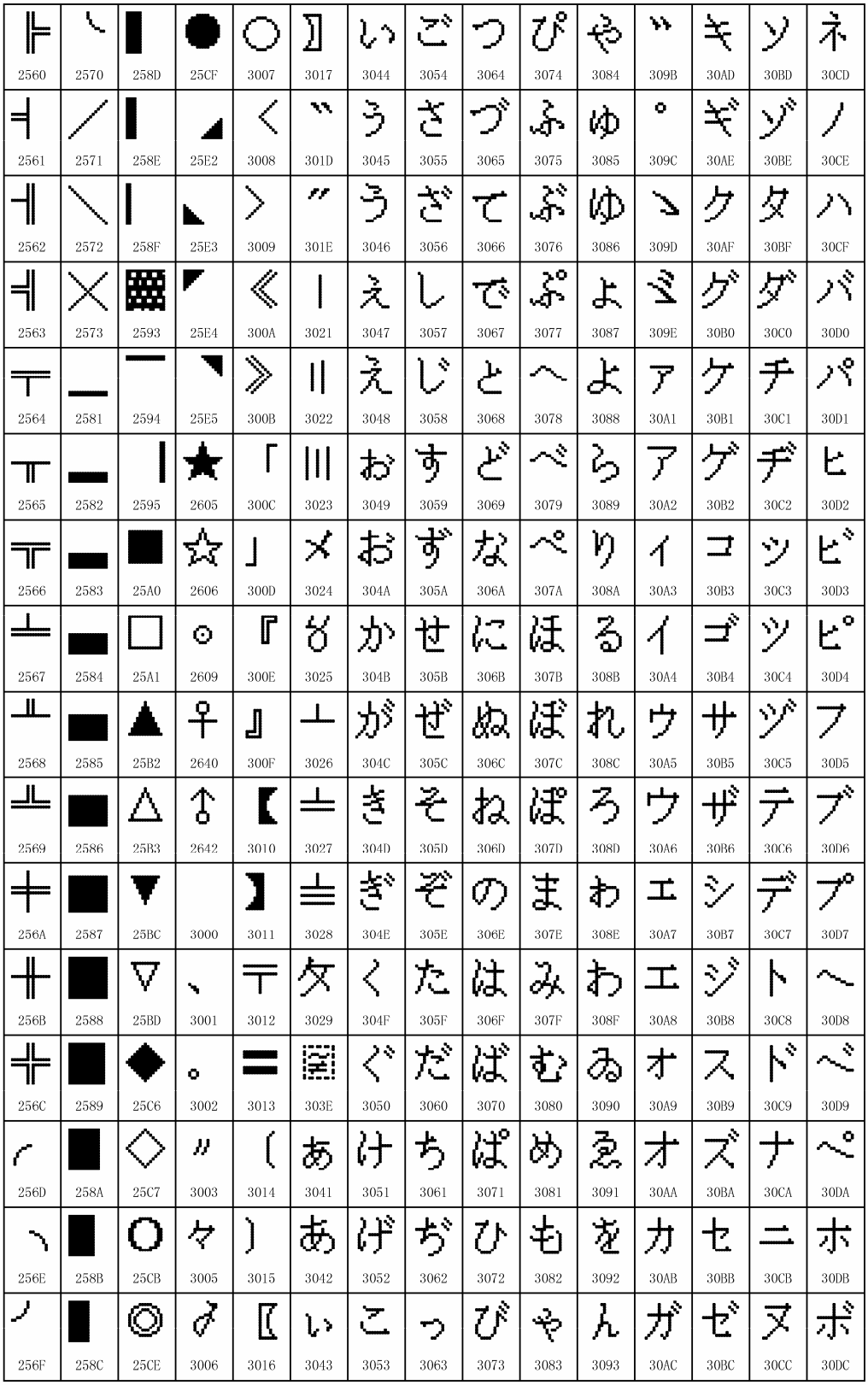
UNICODE3.0 字符区

¡ A1	± B1	Á C1	Ñ D1	á E1	ñ F1	ē 113	ǔ 1D4	Γ 393	Τ 3A4	λ 3BB	Б 411	С 421	б 431	с 441
¢ A2	² B2	Â C2	Ò D2	â E2	ò F2	ě 11B	ǫ 1D6	Δ 394	Υ 3A5	μ 3BC	В 412	Т 422	в 432	т 442
£ A3	³ B3	Ã C3	Ó D3	ã E3	ó F3	ī 12B	ǫ 1D8	Ε 395	Φ 3A6	ν 3BD	Г 413	У 423	г 433	у 443
¤ A4	´ B4	Ä C4	Ô D4	ä E4	ô F4	ń 144	ǔ 1DA	Ζ 396	Χ 3A7	ξ 3BE	Д 414	Ф 424	д 434	ф 444
¥ A5	µ B5	Å C5	Õ D5	å E5	õ F5	ň 148	ǜ 1DC	Η 397	Ψ 3A8	ο 3BF	Е 415	Х 425	е 435	х 445
¦ A6	¶ B6	Æ C6	Ö D6	æ E6	ö F6	ō 14D	ɑ 251	Θ 398	Ω 3A9	π 3C0	Ж 416	Ц 426	ж 436	ц 446
§ A7	• B7	Ç C7	× D7	ç E7	÷ F7	Œ 152	ɡ 261	Ι 399	α 3B1	ρ 3C1	З 417	Ч 427	з 437	ч 447
¨ A8	¸ B8	È C8	Ø D8	è E8	ø F8	œ 153	ˆ 2C6	Κ 39A	β 3B2	σ 3C3	И 418	Ш 428	и 438	ш 448
© A9	¹ B9	É C9	Ù D9	é E9	ù F9	Š 160	ˇ 2C7	Λ 39B	γ 3B3	τ 3C4	Й 419	Щ 429	й 439	щ 449
ª AA	º BA	Ê CA	Ú DA	ê EA	ú FA	š 161	— 2C9	Μ 39C	δ 3B4	υ 3C5	К 41A	Ъ 42A	к 43A	ъ 44A
« AB	» BB	Ë CB	Û DB	ë EB	û FB	ū 16B	ˊ 2CA	Ν 39D	ε 3B5	φ 3C6	Л 41B	Ы 42B	л 43B	ы 44B
¬ AC	¼ BC	Ì CC	Ü DC	ì EC	ü FC	Ÿ 178	ˋ 2CB	Ξ 39E	ζ 3B6	χ 3C7	М 41C	Ь 42C	м 43C	ь 44C
AD	½ BD	Í CD	Ý DD	í ED	ý FD	ƒ 192	• 2D9	Ο 39F	η 3B7	ψ 3C8	Н 41D	Э 42D	н 43D	э 44D
® AE	¾ BE	Î CE	Þ DE	î EE	þ FE	ǎ 1CE	˜ 2DC	Π 3A0	θ 3B8	ω 3C9	О 41E	Ю 42E	о 43E	ю 44E
¯ AF	¿ BF	Ï CF	ß DF	ï EF	ÿ FF	ǐ 1D0	Α 391	Ρ 3A1	ι 3B9	Ё 401	П 41F	Я 42F	п 43F	я 44F
° B0	À C0	Ð D0	à E0	ð F0	ā 101	ǒ 1D2	Β 392	Σ 3A3	κ 3BA	А 410	Р 420	а 430	р 440	ё 451

UNICODE3.0 字符区



UNICODE3.0 字符区



UNICODE3.0 字符区



UNICODE3.0 字符区

FE4D	FE5F	FF05	FF15	FF25	FF35	FF45	FF55							
FE4E	FE60	FF06	FF16	FF26	FF36	FF46	FF56							
FE4F	FE61	FF07	FF17	FF27	FF37	FF47	FF57							
FE50	FE62	FF08	FF18	FF28	FF38	FF48	FF58							
FE51	FE63	FF09	FF19	FF29	FF39	FF49	FF59							
FE52	FE64	FF0A	FF1A	FF2A	FF3A	FF4A	FF5A							
FE54	FE65	FF0B	FF1B	FF2B	FF3B	FF4B	FF5B							
FE55	FE66	FF0C	FF1C	FF2C	FF3C	FF4C	FF5C							
FE56	FE68	FF0D	FF1D	FF2D	FF3D	FF4D	FF5D							
FE57	FE69	FF0E	FF1E	FF2E	FF3E	FF4E	FF5E							
FE59	FE6A	FF0F	FF1F	FF2F	FF3F	FF4F	FFE0							
FE5A	FE6B	FF10	FF20	FF30	FF40	FF50	FFE1							
FE5B	FF01	FF11	FF21	FF31	FF41	FF51	FFE2							
FE5C	FF02	FF12	FF22	FF32	FF42	FF52	FFE3							
FE5D	FF03	FF13	FF23	FF33	FF43	FF53	FFE4							
FE5E	FF04	FF14	FF24	FF34	FF44	FF54	FFE5							

10.2 扩展字符区字符

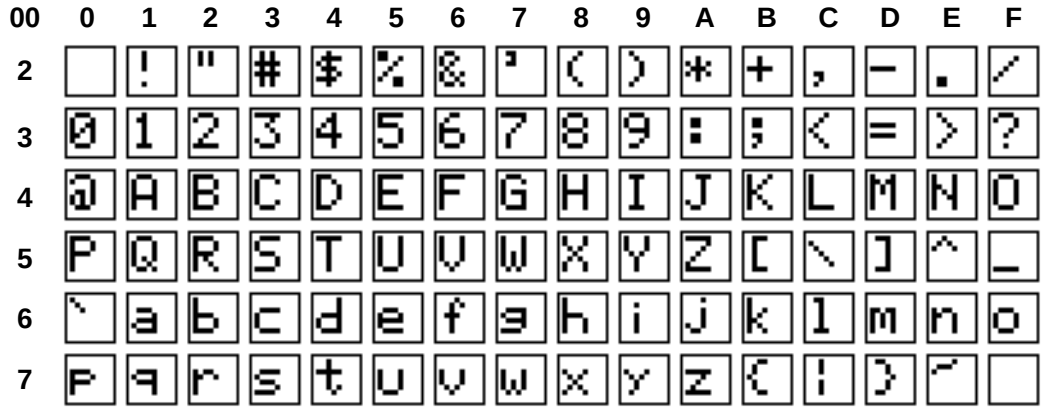
扩展字符区的内码组成为 AAA1~ABC0 共计 126 个字符。

扩展字符区

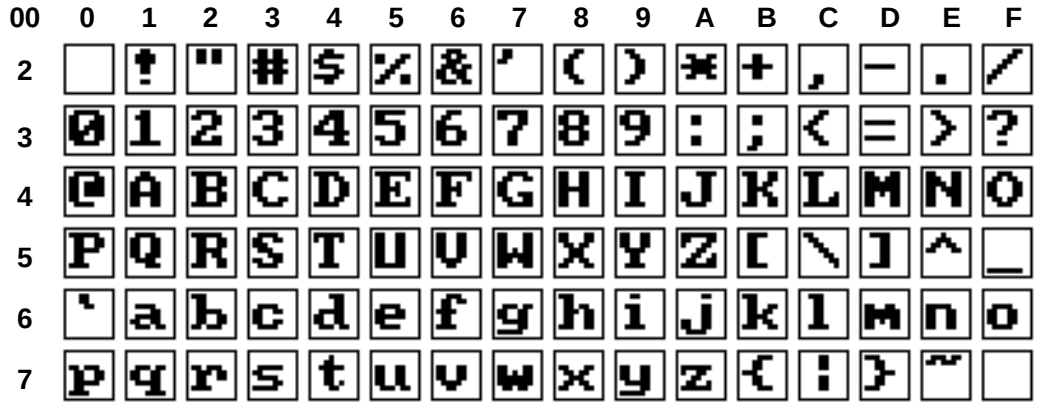
AA	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		!	"	#	¥	%	&	'	(	)	*	+	,	-	.	/
B	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
C	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
D	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
E	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
F	p	q	r	s	t	u	v	w	x	y	z	{	|	}	¯	

AB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A		ā	á	ǎ	à	ē	é	ě	è	ī	í	ǐ	ì	ō	ó	ǒ
B	ò	ū	ú	ǔ	ù	ǖ	ǘ	ǚ	ǜ	ü	ê	ɑ	ḿ	ń	ň	ɲ
C	g															
D																
E																
F																

10.3 5×7 点 ASCII 字符（96 字符）



10.4 7×8 点 ASCII 字符（96 字符）



10.5 Unicode 字符区字符

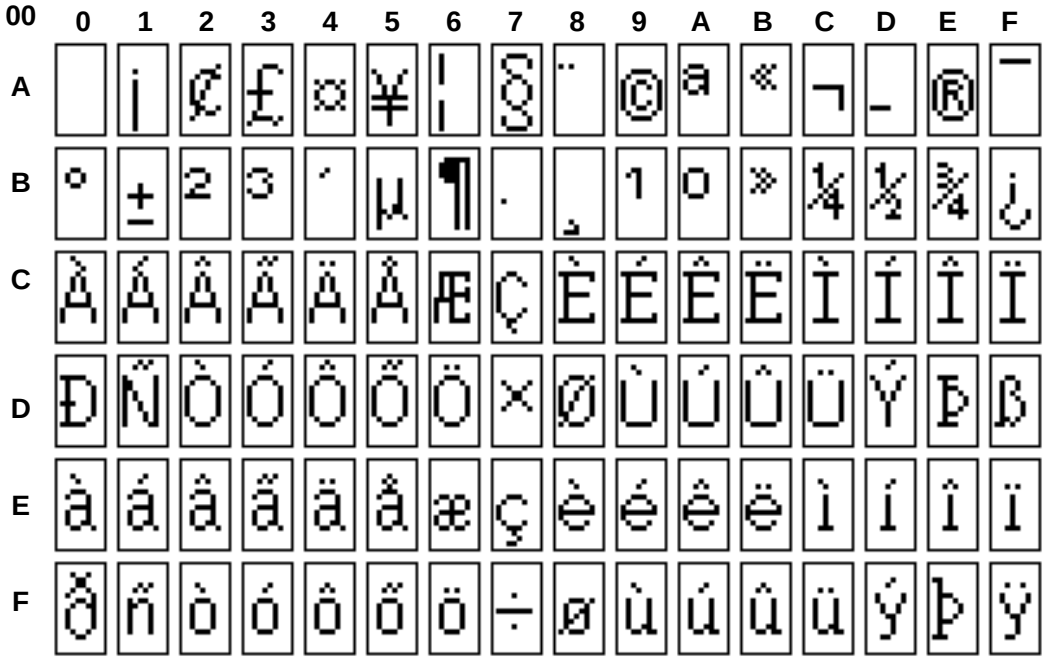
Unicode 字符区共收录　拉丁文系（LATIN）、希腊文系（GREEK）、西里尔文系（CYRILLIC）3 大文系，共计收录字符 456 个

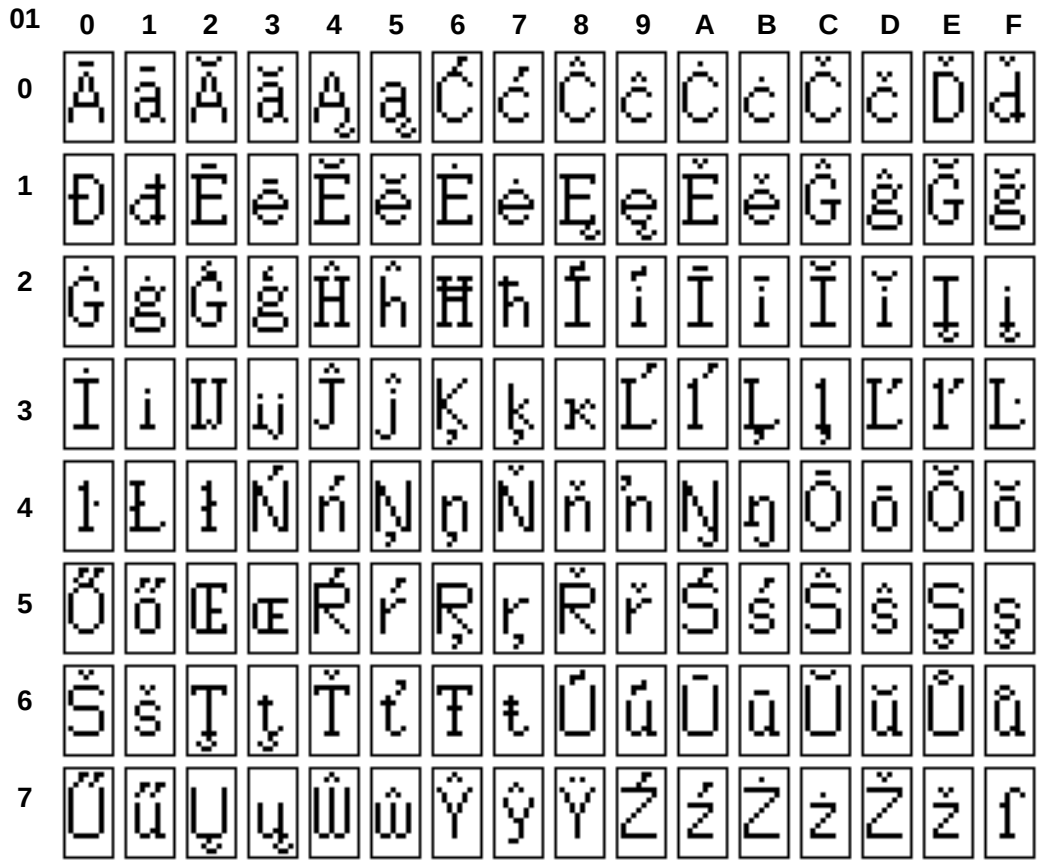
1）拉丁文系（8×16 点 LATIN）

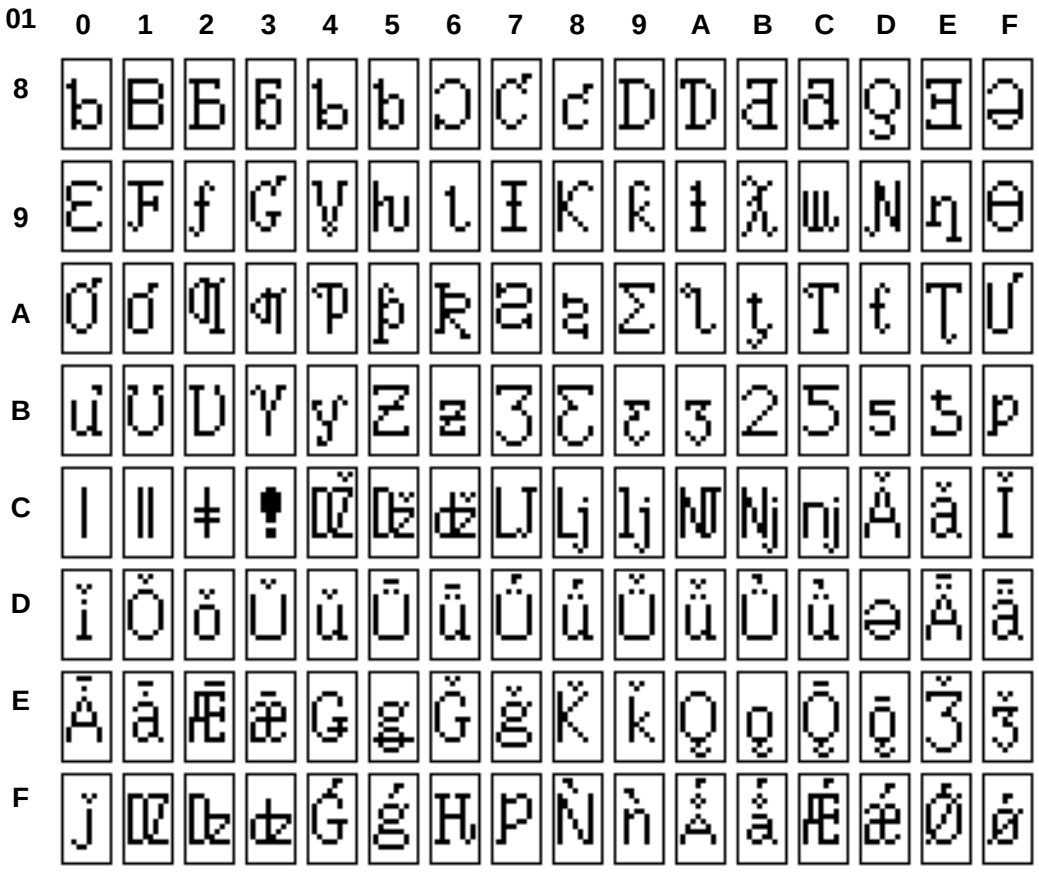
Unicode 字符区-拉丁文系的内码范围为 00A0~0217 共计收录 376 个字符。

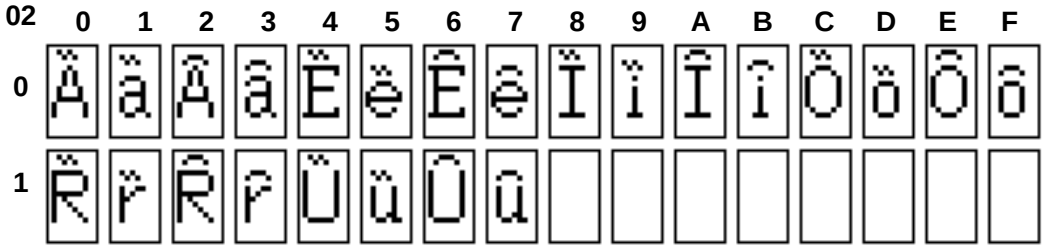
***说明**：由于拉丁文系（BASIC LATIN）的内码范围 0020~007E 与基本 ASCII 编码一致（可以共用），因此未列入 Unicode 字符区-拉丁文系中（见附件：10.2）。

Unicode 字符区-拉丁文系




01
0 1 2 3 4 5 6 7 8 9 A B C D E F
0
1
2
3
4
5
6
7

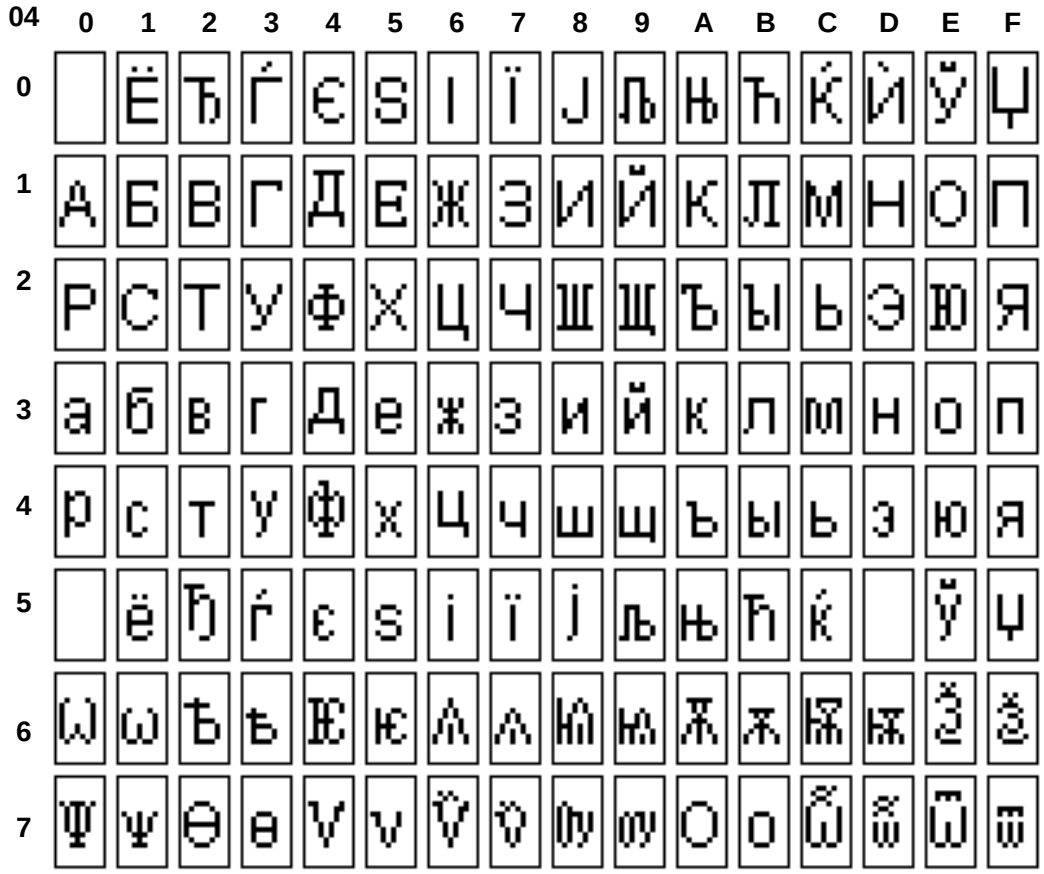

01
0 1 2 3 4 5 6 7 8 9 A B C D E F
8
9
A
B
C
D
E
F

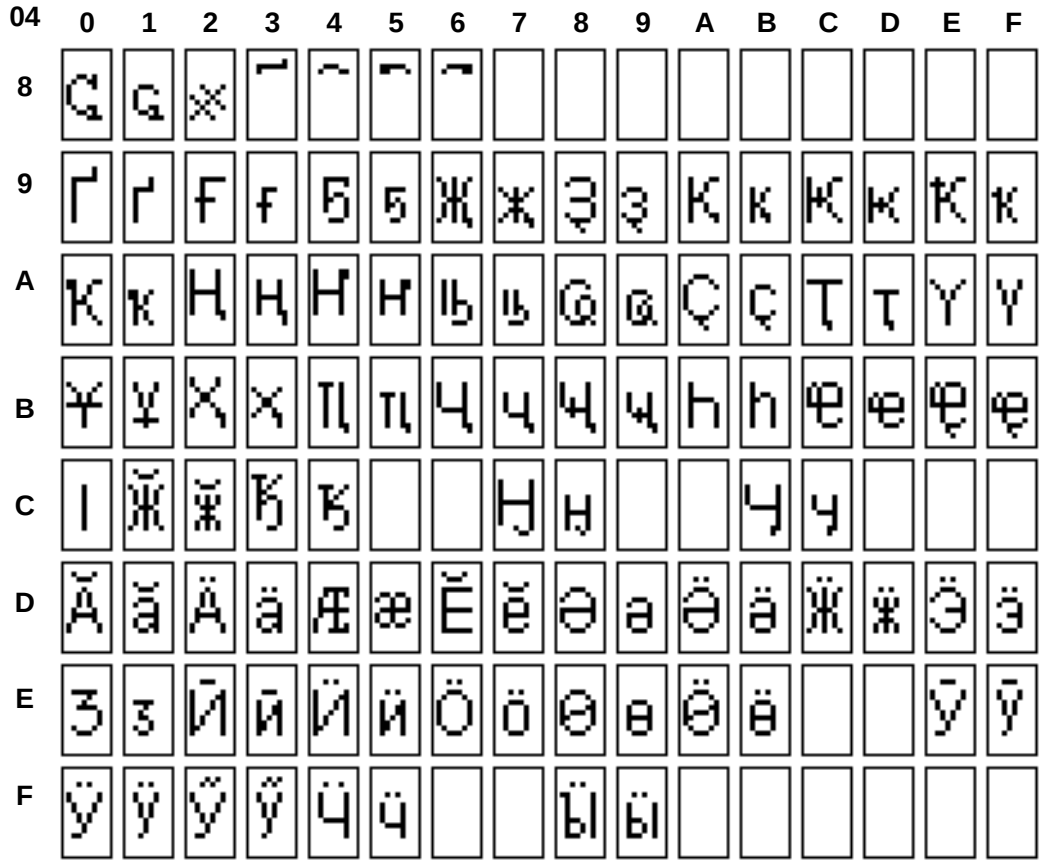

02
0 1 2 3 4 5 6 7 8 9 A B C D E F
0
1

2） 西里尔文系（8×16 点 CYRILLIC）

Unicode 字符区-希腊文系的内码范围为 0400~04F9 共计收录 250 个字符。

Unicode 字符区-西里尔文系




04
0 1 2 3 4 5 6 7 8 9 A B C D E F
8
9
A
B
C
D
E
F

2） 希腊文系（8×16 点 BASIC GREEK）

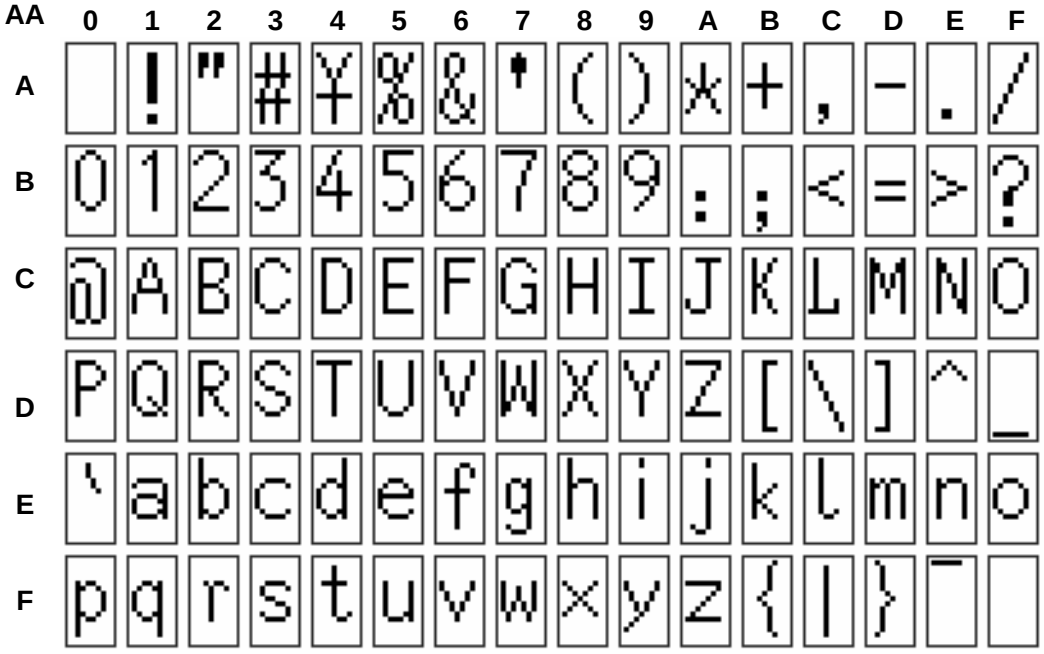
Unicode 字符区-希腊文系的内码范围为 0370~03CF 共计收录 96 个字符。

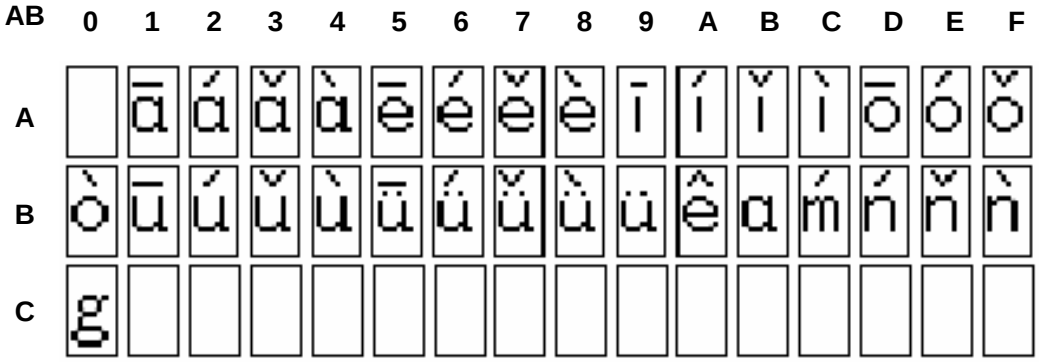
Unicode 字符区-西里尔文系



10.6 8×16 点扩展字符区 ASCII 字符（126 字符）

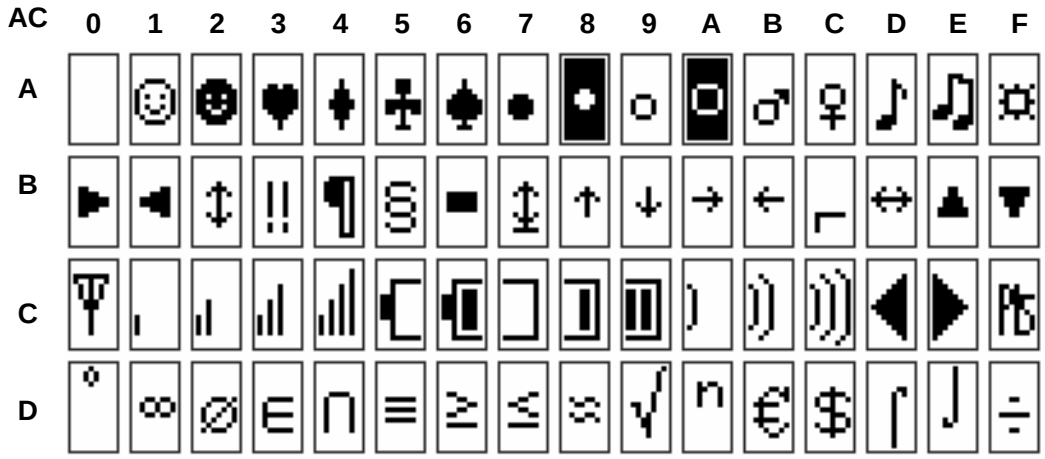
内码组成为 AAA1~ABC0 共计 126 个字符





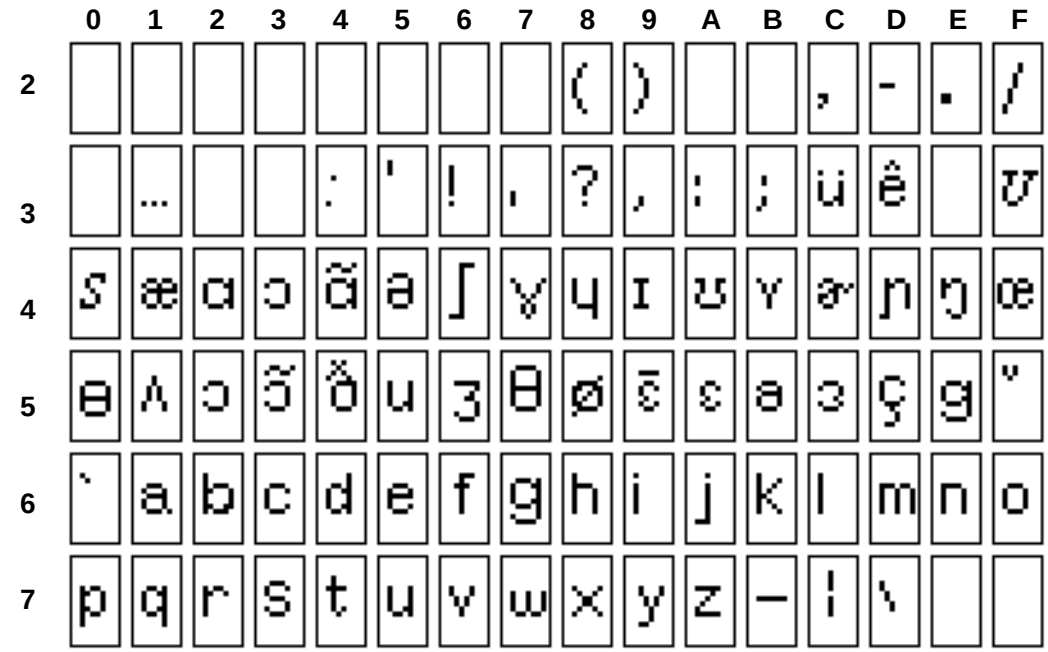
10.7 8×16 自定义特殊字符区 ASCII 字符（64 字符）

内码组成为 AAA1~ABC0 共计 64 个字符



10.8 8×16 自定义国际音标字符区 ASCII 字符（96 字符）

内码组成为 20~7F 共计 96 个字符



10.9 汉字点阵排列格式转换程序

汉字点阵排列格式转换程序，可以把任意一种排列格式转换成其他格式

```
//rotate.c

void Rotate(unsigned char *src,unsigned char *des,unsigned short w,short h,unsigned char srcstyle,unsigned char desstyle);

//extern int loadfile(const char *filename, void *buffer, int num);
//extern int savefile(const char *filename, void *buffer, int num);
//extern int appendfile(const char *filename, const void *buffer, int num);

void main()
{
    unsigned char srcfont[32]=
{0x00,0x00,0x0E,0xFC,0xEA,0x08,0xAA,0x08,0xAA,0xE8,0xAA,0xA8,0xAC,0xA8,0xAA,0xA8,
0xAA,0xA8,0xAA,0xA8,0xEA,0xE8,0xAA,0xA8,0x0C,0x08,0x08,0x08,0x08,0x28,0x08,0x10};
    unsigned char desfont[32];
    Rotate(srcfont,desfont,16,16,0,2); //把 16 点阵横置横排点阵转换成竖置横排点阵
//  savefile("srcb0.b16",srcfont,32);
//  savefile("desb2.b16",desfont,32);
}

void Rotate(unsigned char *src,unsigned char *des,unsigned short w,short h,unsigned char srcstyle,unsigned char desstyle)
// 功能：排列格式的变换，可以把任意一种排列格式转换成其他格式
// 参数：src - 待转换的汉字点阵数据地址
//       des - 用于保存转换后的汉字点阵数据的 memory 地址
///      w - 汉字点阵宽度
//       h - 汉字点阵高度
//       srcstyle - 待转换的汉字点阵数据排列方式
//       desstyle - 转换后的汉字点阵排列方式
//                  汉字点阵排列方式：0-横置横排，（b7 在左面，b0 在右面）
//                                    1-横置竖排，（b7 在左面，b0 在右面）
//                                    2-竖置横排，（b0 在上面，b7 在下面）
//                                    3-竖置竖排，（b0 在上面，b7 在下面）
{

    unsigned short x,y;
    unsigned char nbit;
    unsigned short posbyte;
    unsigned char posbit;

    //w=(w+1)/2*2;
    if(desstyle<2)
```

```
        y=(w+7)/8*h;
    else
        y=(h+7)/8*w;
    for(x=0; x<y; x++)
        des[x]=0;

    /*行列扫描*/
    for(y=0;y<h;y++)                                    /*扫描每一行*/
    {
        for(x=0;x<w;x++)                                /*扫描一行中的每一列*/
        {

            switch(srcstyle)
            {
            case 0:                                      //0-横置横排
                posbyte=(w+7)/8*y + x/8;
                posbit=7-x%8;
                break;
            case 1:                                      //1-横置竖排
                posbyte=x/8*h + y;
                posbit=7-x%8;
                break;
            case 2:                                      //2-竖置横排
                posbyte=y/8*w + x;
                posbit=y%8;
                break;
            default:                                     //3-竖置竖排
                posbyte=(h+7)/8*x + y/8;
                posbit=y%8;
                break;
            }

            nbit=src[posbyte] & (0x01<<posbit);

            if(nbit!=0)
            {
                switch(desstyle)
                {
                case 0:                                 //0-横置横排
                    posbyte=(w+7)/8*y + x/8;
                    posbit=7-x%8;
                    break;
                case 1:                                 //1-横置竖排
                    posbyte=x/8*h + y;
                    posbit=7-x%8;
                    break;
```

```
                case 2:                         //2-竖置横排
                    posbyte=y/8*w + x;
                    posbit=y%8;
                    break;
                default:                        //3-竖置竖排
                    posbyte=(h+7)/8*x + y/8;
                    posbit=y%8;
                    break;
                }
                des[posbyte] |= (0x01<<posbit);
            }
        }
    }
}
```

10.10 UNICODE3.0 字符区对照表

该表描述了字符区离散字符所对应的码位，根据该表进行查询，可以得到该码位在字符区中的字序位置，从而计算出相应地址。

按照 WORD 方式进行排列的字符对应表：

unsigned int ZFTABLE[1087]={

0xa1,0xa2,0xa3,0xa4,0xa5,0xa6,0xa7,0xa8,
0xa9,0xaa,0xab,0xac,0xad,0xae,0xaf,0xb0,
0xb1,0xb2,0xb3,0xb4,0xb5,0xb6,0xb7,0xb8,
0xb9,0xba,0xbb,0xbc,0xbd,0xbe,0xbf,0xc0,
0xc1,0xc2,0xc3,0xc4,0xc5,0xc6,0xc7,0xc8,
0xc9,0xca,0xcb,0xcc,0xcd,0xce,0xcf,0xd0,
0xd1,0xd2,0xd3,0xd4,0xd5,0xd6,0xd7,0xd8,
0xd9,0xda,0xdb,0xdc,0xdd,0xde,0xdf,0xe0,
0xe1,0xe2,0xe3,0xe4,0xe5,0xe6,0xe7,0xe8,
0xe9,0xea,0xeb,0xec,0xed,0xee,0xef,0xf0,
0xf1,0xf2,0xf3,0xf4,0xf5,0xf6,0xf7,0xf8,
0xf9,0xfa,0xfb,0xfc,0xfd,0xfe,0xff,0x101,
0x113,0x11b,0x12b,0x144,0x148,0x14d,0x152,0x153,
0x160,0x161,0x16b,0x178,0x192,0x1ce,0x1d0,0x1d2,
0x1d4,0x1d6,0x1d8,0x1da,0x1dc,0x251,0x261,0x2c6,
0x2c7,0x2c9,0x2ca,0x2cb,0x2d9,0x2dc,0x391,0x392,
0x393,0x394,0x395,0x396,0x397,0x398,0x399,0x39a,
0x39b,0x39c,0x39d,0x39e,0x39f,0x3a0,0x3a1,0x3a3,
0x3a4,0x3a5,0x3a6,0x3a7,0x3a8,0x3a9,0x3b1,0x3b2,
0x3b3,0x3b4,0x3b5,0x3b6,0x3b7,0x3b8,0x3b9,0x3ba,
0x3bb,0x3bc,0x3bd,0x3be,0x3bf,0x3c0,0x3c1,0x3c3,
0x3c4,0x3c5,0x3c6,0x3c7,0x3c8,0x3c9,0x401,0x410,
0x411,0x412,0x413,0x414,0x415,0x416,0x417,0x418,
0x419,0x41a,0x41b,0x41c,0x41d,0x41e,0x41f,0x420,
0x421,0x422,0x423,0x424,0x425,0x426,0x427,0x428,
0x429,0x42a,0x42b,0x42c,0x42d,0x42e,0x42f,0x430,
0x431,0x432,0x433,0x434,0x435,0x436,0x437,0x438,
0x439,0x43a,0x43b,0x43c,0x43d,0x43e,0x43f,0x440,
0x441,0x442,0x443,0x444,0x445,0x446,0x447,0x448,
0x449,0x44a,0x44b,0x44c,0x44d,0x44e,0x44f,0x451,
0x2010,0x2013,0x2014,0x2015,0x2016,0x2018,0x2019,0x201a,
0x201c,0x201d,0x201e,0x2020,0x2021,0x2022,0x2025,0x2026,
0x2030,0x2032,0x2033,0x2035,0x2039,0x203a,0x203b,0x20ac,
0x2103,0x2105,0x2109,0x2116,0x2121,0x2122,0x2160,0x2161,
0x2162,0x2163,0x2164,0x2165,0x2166,0x2167,0x2168,0x2169,
0x216a,0x216b,0x2170,0x2171,0x2172,0x2173,0x2174,0x2175,
0x2176,0x2177,0x2178,0x2179,0x2190,0x2191,0x2192,0x2193,
0x2196,0x2197,0x2198,0x2199,0x2208,0x220f,0x2211,0x2215,

0x221a,0x221d,0x221e,0x221f,0x2220,0x2223,0x2225,0x2227,
0x2228,0x2229,0x222a,0x222b,0x222e,0x2234,0x2235,0x2236,

0x2237,0x223d,0x2248,0x224c,0x2252,0x2260,0x2261,0x2264,
0x2265,0x2266,0x2267,0x226e,0x226f,0x2295,0x2299,0x22a5,
0x22bf,0x2312,0x2460,0x2461,0x2462,0x2463,0x2464,0x2465,
0x2466,0x2467,0x2468,0x2469,0x2474,0x2475,0x2476,0x2477,
0x2478,0x2479,0x247a,0x247b,0x247c,0x247d,0x247e,0x247f,
0x2480,0x2481,0x2482,0x2483,0x2484,0x2485,0x2486,0x2487,
0x2488,0x2489,0x248a,0x248b,0x248c,0x248d,0x248e,0x248f,
0x2490,0x2491,0x2492,0x2493,0x2494,0x2495,0x2496,0x2497,
0x2498,0x2499,0x249a,0x249b,0x2500,0x2501,0x2502,0x2503,
0x2504,0x2505,0x2506,0x2507,0x2508,0x2509,0x250a,0x250b,
0x250c,0x250d,0x250e,0x250f,0x2510,0x2511,0x2512,0x2513,
0x2514,0x2515,0x2516,0x2517,0x2518,0x2519,0x251a,0x251b,
0x251c,0x251d,0x251e,0x251f,0x2520,0x2521,0x2522,0x2523,
0x2524,0x2525,0x2526,0x2527,0x2528,0x2529,0x252a,0x252b,
0x252c,0x252d,0x252e,0x252f,0x2530,0x2531,0x2532,0x2533,
0x2534,0x2535,0x2536,0x2537,0x2538,0x2539,0x253a,0x253b,
0x253c,0x253d,0x253e,0x253f,0x2540,0x2541,0x2542,0x2543,
0x2544,0x2545,0x2546,0x2547,0x2548,0x2549,0x254a,0x254b,
0x2550,0x2551,0x2552,0x2553,0x2554,0x2555,0x2556,0x2557,
0x2558,0x2559,0x255a,0x255b,0x255c,0x255d,0x255e,0x255f,
0x2560,0x2561,0x2562,0x2563,0x2564,0x2565,0x2566,0x2567,
0x2568,0x2569,0x256a,0x256b,0x256c,0x256d,0x256e,0x256f,
0x2570,0x2571,0x2572,0x2573,0x2581,0x2582,0x2583,0x2584,
0x2585,0x2586,0x2587,0x2588,0x2589,0x258a,0x258b,0x258c,
0x258d,0x258e,0x258f,0x2593,0x2594,0x2595,0x25a0,0x25a1,
0x25b2,0x25b3,0x25bc,0x25bd,0x25c6,0x25c7,0x25cb,0x25ce,
0x25cf,0x25e2,0x25e3,0x25e4,0x25e5,0x2605,0x2606,0x2609,
0x2640,0x2642,0x3000,0x3001,0x3002,0x3003,0x3005,0x3006,
0x3007,0x3008,0x3009,0x300a,0x300b,0x300c,0x300d,0x300e,
0x300f,0x3010,0x3011,0x3012,0x3013,0x3014,0x3015,0x3016,
0x3017,0x301d,0x301e,0x3021,0x3022,0x3023,0x3024,0x3025,
0x3026,0x3027,0x3028,0x3029,0x303e,0x3041,0x3042,0x3043,
0x3044,0x3045,0x3046,0x3047,0x3048,0x3049,0x304a,0x304b,
0x304c,0x304d,0x304e,0x304f,0x3050,0x3051,0x3052,0x3053,
0x3054,0x3055,0x3056,0x3057,0x3058,0x3059,0x305a,0x305b,
0x305c,0x305d,0x305e,0x305f,0x3060,0x3061,0x3062,0x3063,
0x3064,0x3065,0x3066,0x3067,0x3068,0x3069,0x306a,0x306b,
0x306c,0x306d,0x306e,0x306f,0x3070,0x3071,0x3072,0x3073,
0x3074,0x3075,0x3076,0x3077,0x3078,0x3079,0x307a,0x307b,
0x307c,0x307d,0x307e,0x307f,0x3080,0x3081,0x3082,0x3083,
0x3084,0x3085,0x3086,0x3087,0x3088,0x3089,0x308a,0x308b,
0x308c,0x308d,0x308e,0x308f,0x3090,0x3091,0x3092,0x3093,

0x309b,0x309c,0x309d,0x309e,0x30a1,0x30a2,0x30a3,0x30a4,
0x30a5,0x30a6,0x30a7,0x30a8,0x30a9,0x30aa,0x30ab,0x30ac,

0x30ad,0x30ae,0x30af,0x30b0,0x30b1,0x30b2,0x30b3,0x30b4,
0x30b5,0x30b6,0x30b7,0x30b8,0x30b9,0x30ba,0x30bb,0x30bc,
0x30bd,0x30be,0x30bf,0x30c0,0x30c1,0x30c2,0x30c3,0x30c4,
0x30c5,0x30c6,0x30c7,0x30c8,0x30c9,0x30ca,0x30cb,0x30cc,
0x30cd,0x30ce,0x30cf,0x30d0,0x30d1,0x30d2,0x30d3,0x30d4,
0x30d5,0x30d6,0x30d7,0x30d8,0x30d9,0x30da,0x30db,0x30dc,
0x30dd,0x30de,0x30df,0x30e0,0x30e1,0x30e2,0x30e3,0x30e4,
0x30e5,0x30e6,0x30e7,0x30e8,0x30e9,0x30ea,0x30eb,0x30ec,
0x30ed,0x30ee,0x30ef,0x30f0,0x30f1,0x30f2,0x30f3,0x30f4,
0x30f5,0x30f6,0x30fc,0x30fd,0x30fe,0x3105,0x3106,0x3107,
0x3108,0x3109,0x310a,0x310b,0x310c,0x310d,0x310e,0x310f,
0x3110,0x3111,0x3112,0x3113,0x3114,0x3115,0x3116,0x3117,
0x3118,0x3119,0x311a,0x311b,0x311c,0x311d,0x311e,0x311f,
0x3120,0x3121,0x3122,0x3123,0x3124,0x3125,0x3126,0x3127,
0x3128,0x3129,0x3220,0x3221,0x3222,0x3223,0x3224,0x3225,
0x3226,0x3227,0x3228,0x3229,0x3231,0x32a3,0x338e,0x338f,
0x339c,0x339d,0x339e,0x33a1,0x33c4,0x33ce,0x33d1,0x33d2,
0x33d5,0xe76c,0xe78d,0xe78e,0xe78f,0xe790,0xe791,0xe792,
0xe793,0xe794,0xe795,0xe796,0xe7c7,0xe7c8,0xe7e7,0xe7e8,
0xe7e9,0xe7ea,0xe7eb,0xe7ec,0xe7ed,0xe7ee,0xe7ef,0xe7f0,
0xe7f1,0xe7f2,0xe7f3,0xe815,0xe816,0xe817,0xe818,0xe819,
0xe81a,0xe81b,0xe81c,0xe81d,0xe81e,0xe81f,0xe820,0xe821,
0xe822,0xe823,0xe824,0xe825,0xe826,0xe827,0xe828,0xe829,
0xe82a,0xe82b,0xe82c,0xe82d,0xe82e,0xe82f,0xe830,0xe831,
0xe832,0xe833,0xe834,0xe835,0xe836,0xe837,0xe838,0xe839,
0xe83a,0xe83b,0xe83c,0xe83d,0xe83e,0xe83f,0xe840,0xe841,
0xe842,0xe843,0xe844,0xe845,0xe846,0xe847,0xe848,0xe849,
0xe84a,0xe84b,0xe84c,0xe84d,0xe84e,0xe84f,0xe850,0xe851,
0xe852,0xe853,0xe854,0xe855,0xe856,0xe857,0xe858,0xe859,
0xe85a,0xe85b,0xe85c,0xe85d,0xe85e,0xe85f,0xe860,0xe861,
0xe862,0xe863,0xe864,0xf92c,0xf979,0xf995,0xf9e7,0xf9f1,
0xfa0c,0xfa0d,0xfa0e,0xfa0f,0xfa11,0xfa13,0xfa14,0xfa18,
0xfa1f,0xfa20,0xfa21,0xfa23,0xfa24,0xfa27,0xfa28,0xfa29,
0xfe30,0xfe31,0xfe33,0xfe34,0xfe35,0xfe36,0xfe37,0xfe38,
0xfe39,0xfe3a,0xfe3b,0xfe3c,0xfe3d,0xfe3e,0xfe3f,0xfe40,
0xfe41,0xfe42,0xfe43,0xfe44,0xfe49,0xfe4a,0xfe4b,0xfe4c,
0xfe4d,0xfe4e,0xfe4f,0xfe50,0xfe51,0xfe52,0xfe54,0xfe55,
0xfe56,0xfe57,0xfe59,0xfe5a,0xfe5b,0xfe5c,0xfe5d,0xfe5e,
0xfe5f,0xfe60,0xfe61,0xfe62,0xfe63,0xfe64,0xfe65,0xfe66,
0xfe68,0xfe69,0xfe6a,0xfe6b,0xff01,0xff02,0xff03,0xff04,
0xff05,0xff06,0xff07,0xff08,0xff09,0xff0a,0xff0b,0xff0c,
0xff0d,0xff0e,0xff0f,0xff10,0xff11,0xff12,0xff13,0xff14,

0xff15,0xff16,0xff17,0xff18,0xff19,0xff1a,0xff1b,0xff1c,
0xff1d,0xff1e,0xff1f,0xff20,0xff21,0xff22,0xff23,0xff24,

0xff25,0xff26,0xff27,0xff28,0xff29,0xff2a,0xff2b,0xff2c,
0xff2d,0xff2e,0xff2f,0xff30,0xff31,0xff32,0xff33,0xff34,
0xff35,0xff36,0xff37,0xff38,0xff39,0xff3a,0xff3b,0xff3c,
0xff3d,0xff3e,0xff3f,0xff40,0xff41,0xff42,0xff43,0xff44,
0xff45,0xff46,0xff47,0xff48,0xff49,0xff4a,0xff4b,0xff4c,
0xff4d,0xff4e,0xff4f,0xff50,0xff51,0xff52,0xff53,0xff54,
0xff55,0xff56,0xff57,0xff58,0xff59,0xff5a,0xff5b,0xff5c,
0xff5d,0xff5e,0xffe0,0xffe1,0xffe2,0xffe3,0xffe4,0xffe5

};

按照 BYTE 方式进行排列的字符对应表：

unsigned char ZFTABLE[2174]={

0x00,0xa1,0x00,0xa2,0x00,0xa3,0x00,0xa4,0x00,0xa5,0x00,0xa6,0x00,0xa7,0x00,0xa8,
0x00,0xa9,0x00,0xaa,0x00,0xab,0x00,0xac,0x00,0xad,0x00,0xae,0x00,0xaf,0x00,0xb0,
0x00,0xb1,0x00,0xb2,0x00,0xb3,0x00,0xb4,0x00,0xb5,0x00,0xb6,0x00,0xb7,0x00,0xb8,
0x00,0xb9,0x00,0xba,0x00,0xbb,0x00,0xbc,0x00,0xbd,0x00,0xbe,0x00,0xbf,0x00,0xc0,
0x00,0xc1,0x00,0xc2,0x00,0xc3,0x00,0xc4,0x00,0xc5,0x00,0xc6,0x00,0xc7,0x00,0xc8,
0x00,0xc9,0x00,0xca,0x00,0xcb,0x00,0xcc,0x00,0xcd,0x00,0xce,0x00,0xcf,0x00,0xd0,
0x00,0xd1,0x00,0xd2,0x00,0xd3,0x00,0xd4,0x00,0xd5,0x00,0xd6,0x00,0xd7,0x00,0xd8,
0x00,0xd9,0x00,0xda,0x00,0xdb,0x00,0xdc,0x00,0xdd,0x00,0xde,0x00,0xdf,0x00,0xe0,
0x00,0xe1,0x00,0xe2,0x00,0xe3,0x00,0xe4,0x00,0xe5,0x00,0xe6,0x00,0xe7,0x00,0xe8,
0x00,0xe9,0x00,0xea,0x00,0xeb,0x00,0xec,0x00,0xed,0x00,0xee,0x00,0xef,0x00,0xf0,
0x00,0xf1,0x00,0xf2,0x00,0xf3,0x00,0xf4,0x00,0xf5,0x00,0xf6,0x00,0xf7,0x00,0xf8,
0x00,0xf9,0x00,0xfa,0x00,0xfb,0x00,0xfc,0x00,0xfd,0x00,0xfe,0x00,0xff,0x01,0x01,
0x01,0x13,0x01,0x1b,0x01,0x2b,0x01,0x44,0x01,0x48,0x01,0x4d,0x01,0x52,0x01,0x53,
0x01,0x60,0x01,0x61,0x01,0x6b,0x01,0x78,0x01,0x92,0x01,0xce,0x01,0xd0,0x01,0xd2,
0x01,0xd4,0x01,0xd6,0x01,0xd8,0x01,0xda,0x01,0xdc,0x02,0x51,0x02,0x61,0x02,0xc6,
0x02,0xc7,0x02,0xc9,0x02,0xca,0x02,0xcb,0x02,0xd9,0x02,0xdc,0x03,0x91,0x03,0x92,
0x03,0x93,0x03,0x94,0x03,0x95,0x03,0x96,0x03,0x97,0x03,0x98,0x03,0x99,0x03,0x9a,
0x03,0x9b,0x03,0x9c,0x03,0x9d,0x03,0x9e,0x03,0x9f,0x03,0xa0,0x03,0xa1,0x03,0xa3,
0x03,0xa4,0x03,0xa5,0x03,0xa6,0x03,0xa7,0x03,0xa8,0x03,0xa9,0x03,0xb1,0x03,0xb2,
0x03,0xb3,0x03,0xb4,0x03,0xb5,0x03,0xb6,0x03,0xb7,0x03,0xb8,0x03,0xb9,0x03,0xba,
0x03,0xbb,0x03,0xbc,0x03,0xbd,0x03,0xbe,0x03,0xbf,0x03,0xc0,0x03,0xc1,0x03,0xc3,
0x03,0xc4,0x03,0xc5,0x03,0xc6,0x03,0xc7,0x03,0xc8,0x03,0xc9,0x04,0x01,0x04,0x10,
0x04,0x11,0x04,0x12,0x04,0x13,0x04,0x14,0x04,0x15,0x04,0x16,0x04,0x17,0x04,0x18,
0x04,0x19,0x04,0x1a,0x04,0x1b,0x04,0x1c,0x04,0x1d,0x04,0x1e,0x04,0x1f,0x04,0x20,
0x04,0x21,0x04,0x22,0x04,0x23,0x04,0x24,0x04,0x25,0x04,0x26,0x04,0x27,0x04,0x28,
0x04,0x29,0x04,0x2a,0x04,0x2b,0x04,0x2c,0x04,0x2d,0x04,0x2e,0x04,0x2f,0x04,0x30,

0x04,0x31,0x04,0x32,0x04,0x33,0x04,0x34,0x04,0x35,0x04,0x36,0x04,0x37,0x04,0x38,
0x04,0x39,0x04,0x3a,0x04,0x3b,0x04,0x3c,0x04,0x3d,0x04,0x3e,0x04,0x3f,0x04,0x40,
0x04,0x41,0x04,0x42,0x04,0x43,0x04,0x44,0x04,0x45,0x04,0x46,0x04,0x47,0x04,0x48,
0x04,0x49,0x04,0x4a,0x04,0x4b,0x04,0x4c,0x04,0x4d,0x04,0x4e,0x04,0x4f,0x04,0x51,
0x20,0x10,0x20,0x13,0x20,0x14,0x20,0x15,0x20,0x16,0x20,0x18,0x20,0x19,0x20,0x1a,
0x20,0x1c,0x20,0x1d,0x20,0x1e,0x20,0x20,0x20,0x21,0x20,0x22,0x20,0x25,0x20,0x26,
0x20,0x30,0x20,0x32,0x20,0x33,0x20,0x35,0x20,0x39,0x20,0x3a,0x20,0x3b,0x20,0xac,
0x21,0x03,0x21,0x05,0x21,0x09,0x21,0x16,0x21,0x21,0x21,0x22,0x21,0x60,0x21,0x61,
0x21,0x62,0x21,0x63,0x21,0x64,0x21,0x65,0x21,0x66,0x21,0x67,0x21,0x68,0x21,0x69,
0x21,0x6a,0x21,0x6b,0x21,0x70,0x21,0x71,0x21,0x72,0x21,0x73,0x21,0x74,0x21,0x75,
0x21,0x76,0x21,0x77,0x21,0x78,0x21,0x79,0x21,0x90,0x21,0x91,0x21,0x92,0x21,0x93,
0x21,0x96,0x21,0x97,0x21,0x98,0x21,0x99,0x22,0x08,0x22,0x0f,0x22,0x11,0x22,0x15,
0x22,0x1a,0x22,0x1d,0x22,0x1e,0x22,0x1f,0x22,0x20,0x22,0x23,0x22,0x25,0x22,0x27,
0x22,0x28,0x22,0x29,0x22,0x2a,0x22,0x2b,0x22,0x2e,0x22,0x34,0x22,0x35,0x22,0x36,
0x22,0x37,0x22,0x3d,0x22,0x48,0x22,0x4c,0x22,0x52,0x22,0x60,0x22,0x61,0x22,0x64,
0x22,0x65,0x22,0x66,0x22,0x67,0x22,0x6e,0x22,0x6f,0x22,0x95,0x22,0x99,0x22,0xa5,
0x22,0xbf,0x23,0x12,0x24,0x60,0x24,0x61,0x24,0x62,0x24,0x63,0x24,0x64,0x24,0x65,
0x24,0x66,0x24,0x67,0x24,0x68,0x24,0x69,0x24,0x74,0x24,0x75,0x24,0x76,0x24,0x77,
0x24,0x78,0x24,0x79,0x24,0x7a,0x24,0x7b,0x24,0x7c,0x24,0x7d,0x24,0x7e,0x24,0x7f,
0x24,0x80,0x24,0x81,0x24,0x82,0x24,0x83,0x24,0x84,0x24,0x85,0x24,0x86,0x24,0x87,
0x24,0x88,0x24,0x89,0x24,0x8a,0x24,0x8b,0x24,0x8c,0x24,0x8d,0x24,0x8e,0x24,0x8f,
0x24,0x90,0x24,0x91,0x24,0x92,0x24,0x93,0x24,0x94,0x24,0x95,0x24,0x96,0x24,0x97,
0x24,0x98,0x24,0x99,0x24,0x9a,0x24,0x9b,0x25,0x00,0x25,0x01,0x25,0x02,0x25,0x03,
0x25,0x04,0x25,0x05,0x25,0x06,0x25,0x07,0x25,0x08,0x25,0x09,0x25,0x0a,0x25,0x0b,
0x25,0x0c,0x25,0x0d,0x25,0x0e,0x25,0x0f,0x25,0x10,0x25,0x11,0x25,0x12,0x25,0x13,
0x25,0x14,0x25,0x15,0x25,0x16,0x25,0x17,0x25,0x18,0x25,0x19,0x25,0x1a,0x25,0x1b,
0x25,0x1c,0x25,0x1d,0x25,0x1e,0x25,0x1f,0x25,0x20,0x25,0x21,0x25,0x22,0x25,0x23,
0x25,0x24,0x25,0x25,0x25,0x26,0x25,0x27,0x25,0x28,0x25,0x29,0x25,0x2a,0x25,0x2b,
0x25,0x2c,0x25,0x2d,0x25,0x2e,0x25,0x2f,0x25,0x30,0x25,0x31,0x25,0x32,0x25,0x33,
0x25,0x34,0x25,0x35,0x25,0x36,0x25,0x37,0x25,0x38,0x25,0x39,0x25,0x3a,0x25,0x3b,
0x25,0x3c,0x25,0x3d,0x25,0x3e,0x25,0x3f,0x25,0x40,0x25,0x41,0x25,0x42,0x25,0x43,
0x25,0x44,0x25,0x45,0x25,0x46,0x25,0x47,0x25,0x48,0x25,0x49,0x25,0x4a,0x25,0x4b,
0x25,0x50,0x25,0x51,0x25,0x52,0x25,0x53,0x25,0x54,0x25,0x55,0x25,0x56,0x25,0x57,
0x25,0x58,0x25,0x59,0x25,0x5a,0x25,0x5b,0x25,0x5c,0x25,0x5d,0x25,0x5e,0x25,0x5f,
0x25,0x60,0x25,0x61,0x25,0x62,0x25,0x63,0x25,0x64,0x25,0x65,0x25,0x66,0x25,0x67,
0x25,0x68,0x25,0x69,0x25,0x6a,0x25,0x6b,0x25,0x6c,0x25,0x6d,0x25,0x6e,0x25,0x6f,
0x25,0x70,0x25,0x71,0x25,0x72,0x25,0x73,0x25,0x81,0x25,0x82,0x25,0x83,0x25,0x84,
0x25,0x85,0x25,0x86,0x25,0x87,0x25,0x88,0x25,0x89,0x25,0x8a,0x25,0x8b,0x25,0x8c,
0x25,0x8d,0x25,0x8e,0x25,0x8f,0x25,0x93,0x25,0x94,0x25,0x95,0x25,0xa0,0x25,0xa1,
0x25,0xb2,0x25,0xb3,0x25,0xbc,0x25,0xbd,0x25,0xc6,0x25,0xc7,0x25,0xcb,0x25,0xce,
0x25,0xcf,0x25,0xe2,0x25,0xe3,0x25,0xe4,0x25,0xe5,0x26,0x05,0x26,0x06,0x26,0x09,
0x26,0x40,0x26,0x42,0x30,0x00,0x30,0x01,0x30,0x02,0x30,0x03,0x30,0x05,0x30,0x06,
0x30,0x07,0x30,0x08,0x30,0x09,0x30,0x0a,0x30,0x0b,0x30,0x0c,0x30,0x0d,0x30,0x0e,
0x30,0x0f,0x30,0x10,0x30,0x11,0x30,0x12,0x30,0x13,0x30,0x14,0x30,0x15,0x30,0x16,
0x30,0x17,0x30,0x1d,0x30,0x1e,0x30,0x21,0x30,0x22,0x30,0x23,0x30,0x24,0x30,0x25,
0x30,0x26,0x30,0x27,0x30,0x28,0x30,0x29,0x30,0x3e,0x30,0x41,0x30,0x42,0x30,0x43,

0x30,0x44,0x30,0x45,0x30,0x46,0x30,0x47,0x30,0x48,0x30,0x49,0x30,0x4a,0x30,0x4b,
0x30,0x4c,0x30,0x4d,0x30,0x4e,0x30,0x4f,0x30,0x50,0x30,0x51,0x30,0x52,0x30,0x53,
0x30,0x54,0x30,0x55,0x30,0x56,0x30,0x57,0x30,0x58,0x30,0x59,0x30,0x5a,0x30,0x5b,
0x30,0x5c,0x30,0x5d,0x30,0x5e,0x30,0x5f,0x30,0x60,0x30,0x61,0x30,0x62,0x30,0x63,
0x30,0x64,0x30,0x65,0x30,0x66,0x30,0x67,0x30,0x68,0x30,0x69,0x30,0x6a,0x30,0x6b,
0x30,0x6c,0x30,0x6d,0x30,0x6e,0x30,0x6f,0x30,0x70,0x30,0x71,0x30,0x72,0x30,0x73,
0x30,0x74,0x30,0x75,0x30,0x76,0x30,0x77,0x30,0x78,0x30,0x79,0x30,0x7a,0x30,0x7b,
0x30,0x7c,0x30,0x7d,0x30,0x7e,0x30,0x7f,0x30,0x80,0x30,0x81,0x30,0x82,0x30,0x83,
0x30,0x84,0x30,0x85,0x30,0x86,0x30,0x87,0x30,0x88,0x30,0x89,0x30,0x8a,0x30,0x8b,
0x30,0x8c,0x30,0x8d,0x30,0x8e,0x30,0x8f,0x30,0x90,0x30,0x91,0x30,0x92,0x30,0x93,
0x30,0x9b,0x30,0x9c,0x30,0x9d,0x30,0x9e,0x30,0xa1,0x30,0xa2,0x30,0xa3,0x30,0xa4,
0x30,0xa5,0x30,0xa6,0x30,0xa7,0x30,0xa8,0x30,0xa9,0x30,0xaa,0x30,0xab,0x30,0xac,
0x30,0xad,0x30,0xae,0x30,0xaf,0x30,0xb0,0x30,0xb1,0x30,0xb2,0x30,0xb3,0x30,0xb4,
0x30,0xb5,0x30,0xb6,0x30,0xb7,0x30,0xb8,0x30,0xb9,0x30,0xba,0x30,0xbb,0x30,0xbc,
0x30,0xbd,0x30,0xbe,0x30,0xbf,0x30,0xc0,0x30,0xc1,0x30,0xc2,0x30,0xc3,0x30,0xc4,
0x30,0xc5,0x30,0xc6,0x30,0xc7,0x30,0xc8,0x30,0xc9,0x30,0xca,0x30,0xcb,0x30,0xcc,
0x30,0xcd,0x30,0xce,0x30,0xcf,0x30,0xd0,0x30,0xd1,0x30,0xd2,0x30,0xd3,0x30,0xd4,
0x30,0xd5,0x30,0xd6,0x30,0xd7,0x30,0xd8,0x30,0xd9,0x30,0xda,0x30,0xdb,0x30,0xdc,
0x30,0xdd,0x30,0xde,0x30,0xdf,0x30,0xe0,0x30,0xe1,0x30,0xe2,0x30,0xe3,0x30,0xe4,
0x30,0xe5,0x30,0xe6,0x30,0xe7,0x30,0xe8,0x30,0xe9,0x30,0xea,0x30,0xeb,0x30,0xec,
0x30,0xed,0x30,0xee,0x30,0xef,0x30,0xf0,0x30,0xf1,0x30,0xf2,0x30,0xf3,0x30,0xf4,
0x30,0xf5,0x30,0xf6,0x30,0xfc,0x30,0xfd,0x30,0xfe,0x31,0x05,0x31,0x06,0x31,0x07,
0x31,0x08,0x31,0x09,0x31,0x0a,0x31,0x0b,0x31,0x0c,0x31,0x0d,0x31,0x0e,0x31,0x0f,
0x31,0x10,0x31,0x11,0x31,0x12,0x31,0x13,0x31,0x14,0x31,0x15,0x31,0x16,0x31,0x17,
0x31,0x18,0x31,0x19,0x31,0x1a,0x31,0x1b,0x31,0x1c,0x31,0x1d,0x31,0x1e,0x31,0x1f,
0x31,0x20,0x31,0x21,0x31,0x22,0x31,0x23,0x31,0x24,0x31,0x25,0x31,0x26,0x31,0x27,
0x31,0x28,0x31,0x29,0x32,0x20,0x32,0x21,0x32,0x22,0x32,0x23,0x32,0x24,0x32,0x25,
0x32,0x26,0x32,0x27,0x32,0x28,0x32,0x29,0x32,0x31,0x32,0xa3,0x33,0x8e,0x33,0x8f,
0x33,0x9c,0x33,0x9d,0x33,0x9e,0x33,0xa1,0x33,0xc4,0x33,0xce,0x33,0xd1,0x33,0xd2,
0x33,0xd5,0xe7,0x6c,0xe7,0x8d,0xe7,0x8e,0xe7,0x8f,0xe7,0x90,0xe7,0x91,0xe7,0x92,
0xe7,0x93,0xe7,0x94,0xe7,0x95,0xe7,0x96,0xe7,0xc7,0xe7,0xc8,0xe7,0xe7,0xe7,0xe8,
0xe7,0xe9,0xe7,0xea,0xe7,0xeb,0xe7,0xec,0xe7,0xed,0xe7,0xee,0xe7,0xef,0xe7,0xf0,
0xe7,0xf1,0xe7,0xf2,0xe7,0xf3,0xe8,0x15,0xe8,0x16,0xe8,0x17,0xe8,0x18,0xe8,0x19,
0xe8,0x1a,0xe8,0x1b,0xe8,0x1c,0xe8,0x1d,0xe8,0x1e,0xe8,0x1f,0xe8,0x20,0xe8,0x21,
0xe8,0x22,0xe8,0x23,0xe8,0x24,0xe8,0x25,0xe8,0x26,0xe8,0x27,0xe8,0x28,0xe8,0x29,
0xe8,0x2a,0xe8,0x2b,0xe8,0x2c,0xe8,0x2d,0xe8,0x2e,0xe8,0x2f,0xe8,0x30,0xe8,0x31,
0xe8,0x32,0xe8,0x33,0xe8,0x34,0xe8,0x35,0xe8,0x36,0xe8,0x37,0xe8,0x38,0xe8,0x39,
0xe8,0x3a,0xe8,0x3b,0xe8,0x3c,0xe8,0x3d,0xe8,0x3e,0xe8,0x3f,0xe8,0x40,0xe8,0x41,
0xe8,0x42,0xe8,0x43,0xe8,0x44,0xe8,0x45,0xe8,0x46,0xe8,0x47,0xe8,0x48,0xe8,0x49,
0xe8,0x4a,0xe8,0x4b,0xe8,0x4c,0xe8,0x4d,0xe8,0x4e,0xe8,0x4f,0xe8,0x50,0xe8,0x51,
0xe8,0x52,0xe8,0x53,0xe8,0x54,0xe8,0x55,0xe8,0x56,0xe8,0x57,0xe8,0x58,0xe8,0x59,
0xe8,0x5a,0xe8,0x5b,0xe8,0x5c,0xe8,0x5d,0xe8,0x5e,0xe8,0x5f,0xe8,0x60,0xe8,0x61,
0xe8,0x62,0xe8,0x63,0xe8,0x64,0xf9,0x2c,0xf9,0x79,0xf9,0x95,0xf9,0xe7,0xf9,0xf1,
0xfa,0x0c,0xfa,0x0d,0xfa,0x0e,0xfa,0x0f,0xfa,0x11,0xfa,0x13,0xfa,0x14,0xfa,0x18,
0xfa,0x1f,0xfa,0x20,0xfa,0x21,0xfa,0x23,0xfa,0x24,0xfa,0x27,0xfa,0x28,0xfa,0x29,
0xfe,0x30,0xfe,0x31,0xfe,0x33,0xfe,0x34,0xfe,0x35,0xfe,0x36,0xfe,0x37,0xfe,0x38,

0xfe,0x39,0xfe,0x3a,0xfe,0x3b,0xfe,0x3c,0xfe,0x3d,0xfe,0x3e,0xfe,0x3f,0xfe,0x40,
0xfe,0x41,0xfe,0x42,0xfe,0x43,0xfe,0x44,0xfe,0x49,0xfe,0x4a,0xfe,0x4b,0xfe,0x4c,
0xfe,0x4d,0xfe,0x4e,0xfe,0x4f,0xfe,0x50,0xfe,0x51,0xfe,0x52,0xfe,0x54,0xfe,0x55,
0xfe,0x56,0xfe,0x57,0xfe,0x59,0xfe,0x5a,0xfe,0x5b,0xfe,0x5c,0xfe,0x5d,0xfe,0x5e,
0xfe,0x5f,0xfe,0x60,0xfe,0x61,0xfe,0x62,0xfe,0x63,0xfe,0x64,0xfe,0x65,0xfe,0x66,
0xfe,0x68,0xfe,0x69,0xfe,0x6a,0xfe,0x6b,0xff,0x01,0xff,0x02,0xff,0x03,0xff,0x04,
0xff,0x05,0xff,0x06,0xff,0x07,0xff,0x08,0xff,0x09,0xff,0x0a,0xff,0x0b,0xff,0x0c,
0xff,0x0d,0xff,0x0e,0xff,0x0f,0xff,0x10,0xff,0x11,0xff,0x12,0xff,0x13,0xff,0x14,
0xff,0x15,0xff,0x16,0xff,0x17,0xff,0x18,0xff,0x19,0xff,0x1a,0xff,0x1b,0xff,0x1c,
0xff,0x1d,0xff,0x1e,0xff,0x1f,0xff,0x20,0xff,0x21,0xff,0x22,0xff,0x23,0xff,0x24,
0xff,0x25,0xff,0x26,0xff,0x27,0xff,0x28,0xff,0x29,0xff,0x2a,0xff,0x2b,0xff,0x2c,
0xff,0x2d,0xff,0x2e,0xff,0x2f,0xff,0x30,0xff,0x31,0xff,0x32,0xff,0x33,0xff,0x34,
0xff,0x35,0xff,0x36,0xff,0x37,0xff,0x38,0xff,0x39,0xff,0x3a,0xff,0x3b,0xff,0x3c,
0xff,0x3d,0xff,0x3e,0xff,0x3f,0xff,0x40,0xff,0x41,0xff,0x42,0xff,0x43,0xff,0x44,
0xff,0x45,0xff,0x46,0xff,0x47,0xff,0x48,0xff,0x49,0xff,0x4a,0xff,0x4b,0xff,0x4c,
0xff,0x4d,0xff,0x4e,0xff,0x4f,0xff,0x50,0xff,0x51,0xff,0x52,0xff,0x53,0xff,0x54,
0xff,0x55,0xff,0x56,0xff,0x57,0xff,0x58,0xff,0x59,0xff,0x5a,0xff,0x5b,0xff,0x5c,
0xff,0x5d,0xff,0x5e,0xff,0xe0,0xff,0xe1,0xff,0xe2,0xff,0xe3,0xff,0xe4,0xff,0xe5 };