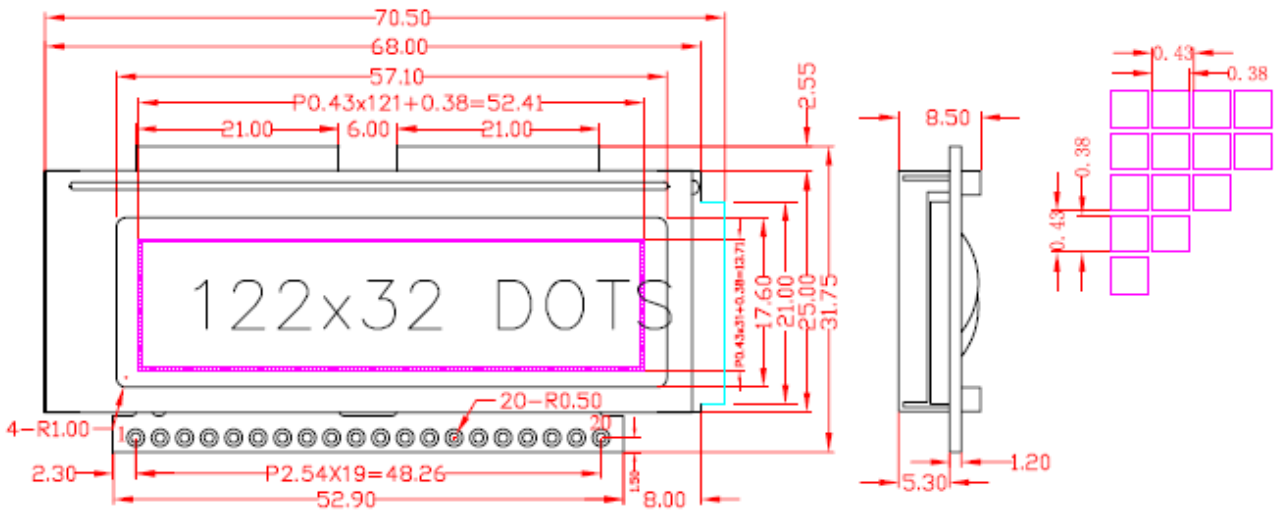
中文液晶显示器使用说明书

ZX12232G-1



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
VSS	VDD	NC	RS	R/W	E	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7	PSB	RES	BLA	BLK	NC	NC

Pin	Symbol	LeveL	Function
1	VSS	0V	GND
2	VDD	+5V	Power supply
3	NC	--	No connection
4	RS(CS)	H/L	H:Data L:Instructioncode (chip enable for senal mode)
5	R/W(SID)	H/L	H:Read L:Write (Serial data for serial mode)
6	E(SCLK)	H.H-L	Enable(clock for serial mode)
7	DB0	H/L	Data bus line
8	DB1	H/L	
9	DB2	H/L	
10	DB3	H/L	
11	DB4	H/L	
12	DB5	H/L	
13	DB6	H/L	
14	DB7	H/L	
15	PSB	H/L	H:Parallel mode L:Serial mode (JP1:left-Vdd,right-Vss)
16	/RST	L	Reset signal active"L"
17	LEDK	0V	Power supply for LED bacdlight
18	LEDA	+5V	
19	NC	NC	NC
20			

功能說明 ST7920

系統介面

ST7920提供三種介面來連接微處理機：8-位元匯流排,4-位元匯流排及串列匯流排介面，經由外部PSB腳來選擇介面的種類，當PSB腳接 “1” 時爲選擇8/4-位元介面模式，而當接 “0” 時爲串列介面模式。

在讀或是寫ST7920的動作中，有兩個8-位元的暫存器將會被使用到，一個是資料暫存器（DR）另一個是指令暫存器（IR）。透過資料暫存器（DR）可以存取DDRAM/CGRAM/GDRAM以及IRAM的值，待存取目標RAM的位址，透過指令命令來選擇，每次的資料暫存器（DR）存取動作都將自動的以上回選擇的目標RAM位址當主體來作寫入或讀取。

配合RS及RW可以選擇決定控制介面的4種讀寫模式，詳見下表：

RS	RW	功 能 說 明
L	L	MPU寫指令到指令暫存器（IR）
L	H	讀出忙碌旗標（BF）及位址計數器（AC）的狀態
H	L	MPU寫入資料到資料暫存器（DR）
H	H	MPU從資料暫存器（DR）中讀出資料

忙碌旗標（BF）

當BF爲 “1” 時，表示內部的操作正在進行中，亦即是內部處於忙碌狀態，此時並不接受新的指令動作，要輸入新的指令前，必須先讀取BF旗標，一直要到BF旗標讀取 “0” 時，才能接受輸入新的指令；一般而言任何的指令輸入後ST7920內部都需要時間處置，在處置完成前並不接受下一個指令，而每一個指令的處置時間並不相同，所以要知道ST7920內部是否已處置完成，可以接受下一指令可以由讀取BF旗標來確認。

位址計數器（AC）

位址計數器（AC）用來儲存DDRAM/CGRAM/IRAM/GDRAM之一的位址，它可藉由設定指令暫存器（IR）來改變，之後只要讀取或是寫入DDRAM/CGRAM/IRAM/GDRAM的值時，位址計數器（AC）的值就會自動加一，當RS爲 “0” 時而RW爲 “1” 時，位址計數器（AC）的值會被讀取到DB6～DB0中。

中文字型產生 ROM (CGROM)及半寬字型ROM（HCGROM）

ST7920 字型產生 ROM 提供 8192 個 16 x 16 點的中文字形圖像以及 126 個 16 x 8 點的數字符號圖像，它使用兩個位元組來提供字型編碼選擇，配合 DDRAM 將要顯示的字型碼寫入到 DDRAM 上，硬體將自動的依照編碼從 CGROM 中將要顯示的字型顯示在螢幕上。

字型產生 RAM (CGRAM)

ST7920 字型產生 RAM 提供使用者圖像定義（造字）功能，可以提供四組 16x16 點的自訂圖像空間，使用者可以將內部字型沒有提供的圖像字型自行定義到 CGRAM 中，便可和 CGRAM 中的定義一般的透過 DDRAM 顯示在螢幕中。

ICON RAM（IRAM）

ST7920 提供 240 點的 ICON 顯示，它分別由 15 組的 IRAM 位址來組成，每一組 IRAM 位址由 16 個位元構成，每次寫入一組 IRAM 時，需先指定 IRAM 的位址，再透過連續寫入兩個位元組的資料來完成，先寫入高位元組（D15～D8）再寫入低位元組（D7～D0）。

顯示資料 RAM（DDRAM）

顯示資料 RAM 提供 64x2 個位元組的空間，最多可以控制 4 行 16 字（64 個字）的中文字型顯示，當寫入顯示資料 RAM 時，可以分別顯示 CGROM、HCGROM 與 CGRAM 的字型；ST7920 可以顯示三種字型，分別是半寬的 HCGROM 字型、CGRAM 字型及中文 CGROM 字型，三種字型的選擇，由在 DDRAM 中寫入的編碼選擇，在 0000H～0006H 的編碼中將選擇 CGRAM 的自定字型，02H～7FH 的編碼中將選擇半寬英數字的字型，至於 A1 以上的編碼將自動的結合下一個位元組，組成兩個位元組的編碼達成中文字型的編碼 BIG5（A140～D75F） GB(A1A0～F7FF)，詳細各種字型編碼如下：

1. 顯示半寬字型：將 8 位元資料寫入 DDRAM 中，範圍爲 02H～7FH 的編碼。
2. 顯示 CGRAM 字型：將 16 位元資料寫入 DDRAM 中，總共有 0000H，0002H，0004H， 0006H 四種編碼。
3. 顯示中文字形：將 16 位元資料寫入 DDRAM 中
   範圍爲 A140H～D75FH 的編碼(BIG5)，範圍爲 A1A0H～F7FFH 的編碼(GB)。

將 16 位元資料寫入 DDRAM 方式爲透過連續寫入兩個位元組的資料來完成，先寫入高位元組（D15～D8）再寫入低位元組（D7～D0）。

參照 Table 5 顯示 CGRAM 的位址、DDRAM 資料以及顯示圖像的關係。

CGRAM 字型與中文字形之編碼只可出現在每一 Address conuter 的啓始位置(參考 Table 4)

80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
H L	H L	H L	H L	H L	H L	H L	H L	H L	H L	H L	H L	H L	H L	H L	H L
S i	t r	o n	i x		S T	7 9	2 0								
矽	創	電	子	.	.	中	文	編	碼		(	正	確	)	
矽	創	電	子	.	.	.	中	文	編	碼					

錯誤填入中文碼位置

Table 4

繪圖 RAM（GDRAM）

繪圖顯示 RAM 提供 64x32 個位元組的記憶空間(由擴充指令設定繪圖 RAM 位址)，最多可以控制 256x64 點的二維繪圖緩衝空間，在更改繪圖 RAM 時，由擴充指令設定 GDRAM 位址先設垂直位址再設水平位址(連續寫入兩個位元組的資料來完成垂直與水平的座標位址)，再寫入兩個 8 位元的資料到繪圖 RAM，而位址計數器（AC）會自動加一,整個寫入繪圖 RAM 的步驟如下：

1. 先將垂直的位元組座標（Y）寫入繪圖 RAM 位址。
2. 再將的水平座標（X）寫入繪圖 RAM 位址。
3. 將 D15～D8 寫入到 RAM 中(寫入第一個 Bytes)。
4. 將 D7～D0 寫入到 RAM 中(寫入第二個 Bytes)。

繪圖顯示的記憶體對應分佈請參考 Table-8。

LCD 驅動電路

LCD 驅動電路提供 33 common 以及 64 segment 訊號線來驅動 LCD 面版，segment 資料從 CGRAM/CGROM 轉換儲存到 64 位元的 segment 串列閂鎖，當 33 個 common 中的一個 common 輸出時，相對應的 segment 資料將從 64 位元的串列閂鎖輸出到 segment 驅動電路。

游標/閃爍控制電路

ST7920 提供硬體游標及閃爍控制電路，由位址計數器(address counter)的值來指定 DDRAM 中的游標或閃爍位置。

DDRAM 資料 (字元代碼)					CGRAM 位址						CGRAM 資料 (高位元組)								CGRAM 資料 (低位元組)							
B15~ B4	B3	B2	B1	B0	B5	B4	B3	B2	B1	B0	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	X	00		X	00		0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0
							0	0	0	1	1	1	1	1	1	1	0	0	0	1	0	0	0	0	0	0
							0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0
							0	0	1	1	0	0	0	1	0	0	0	0	0	1	1	1	1	1	1	0
							0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	1	0	0
							0	1	0	1	0	0	1	1	1	1	0	0	1	0	0	0	0	1	0	0
							0	1	1	0	0	1	1	0	0	1	0	1	0	1	0	0	1	0	0	0
							0	1	1	1	1	0	1	0	0	1	1	0	0	1	0	0	1	0	0	0
							1	0	0	0	0	0	1	0	0	1	0	0	0	1	0	1	0	0	0	0
							1	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0
							1	0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0
							1	0	1	1	0	0	1	1	1	1	0	0	0	0	0	1	0	0	0	0
							1	1	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0
							1	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
							1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
							1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	X	01		X	01		0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	1	0
							0	0	0	1	0	0	0	1	1	0	1	0	0	0	0	0	0	1	0	0
							0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	1	1	0	1	0	0
							0	0	1	1	0	1	0	1	1	1	0	1	1	0	1	0	0	1	0	0
							0	1	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0
							0	1	0	1	0	1	1	1	1	1	1	1	0	0	1	0	0	1	0	0
							0	1	1	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1	0	0
							0	1	1	1	0	1	1	1	1	1	1	1	0	0	1	0	0	1	0	0
							1	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1	0	0
							1	0	0	1	0	1	1	1	1	1	1	1	0	0	1	0	0	1	0	0
							1	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0
							1	0	1	1	0	1	1	1	1	1	1	1	0		0	0	1	0	0	
							1	1	0	0	1	0	1	0	0	0	0	1	0	1	0	0	1	0	0	
							1	1	0	1	1	0	1	1	1	1	1	1	1	0	0	1	1	1	0	0
							1	1	1	0	1	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0
							1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5： DDRAM資料（字元代碼），CGRAM位址以及CGRAM資料（顯示圖像）的相互對照關係圖

附註：

1. DDRAM 資料(字元代碼) 的位元 1 到 2 和 CGRAM 位址的位元 4 到 5 同步吻合(2 位元:4 組圖像).
2. CGRAM 位址的位元 0 到 3 指定字型圖像的列位址，總共指定 16 列（4 位元），第 16 列是游標的顯示區域，游標的顯示和第 16 行的資料採用邏輯 OR 的方式產生顯示結果.
3. 顯示圖像的橫列圖素對應到 CGRAM 資料的位元 0 到 15 (位元 15 在最左邊).
4. 選擇到 CGRAM 的圖像資料，DDRAM 資料的位元 4 到 15 須設爲 0，至於位元 0 及位元 3 則可爲任意值。

ICON RAM 位址 在擴充指令集將 SR 設爲"0",再利用設定 IRAM 位址指令來設定 AC3…..AC0				ICON RAM 資料															
				高位元組								低位元組							
AC3	AC2	AC1	AC0	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15
0	0	0	1	SEG16	SEG17	SEG18	SEG19	SEG20	SEG21	SEG22	SEG23	SEG24	SEG25	SEG26	SEG27	SEG28	SEG29	SEG30	SEG31
0	0	1	0	SEG32	SEG33	SEG34	SEG35	SEG36	SEG37	SEG38	SEG39	SEG40	SEG41	SEG42	SEG43	SEG44	SEG45	SEG46	SEG47
0	0	1	1	SEG48	SEG49	SEG50	SEG51	SEG52	SEG53	SEG54	SEG55	SEG56	SEG57	SEG58	SEG59	SEG60	SEG61	SEG62	SEG63
0	1	0	0	SEG64	SEG65	SEG66	SEG67	SEG68	SEG69	SEG70	SEG71	SEG72	SEG73	SEG74	SEG75	SEG76	SEG77	SEG78	SEG79
0	1	0	1	SEG80	SEG81	SEG82	SEG83	SEG84	SEG85	SEG86	SEG87	SEG88	SEG89	SEG90	SEG91	SEG92	SEG93	SEG94	SEG95
0	1	1	0	SEG96	SEG97	SEG98	SEG99	SEG100	SEG101	SEG102	SEG103	SEG104	SEG105	SEG106	SEG107	SEG108	SEG109	SEG110	SEG111
0	1	1	1	SEG112	SEG113	SEG114	SEG115	SEG116	SEG117	SEG118	SEG119	SEG120	SEG121	SEG122	SEG123	SEG124	SEG125	SEG126	SEG127
1	0	0	0	SEG128	SEG129	SEG130	SEG131	SEG132	SEG133	SEG134	SEG135	SEG136	SEG137	SEG138	SEG139	SEG140	SEG141	SEG142	SEG143
1	0	0	1	SEG144	SEG145	SEG146	SEG147	SEG148	SEG149	SEG150	SEG151	SEG152	SEG153	SEG154	SEG155	SEG156	SEG157	SEG158	SEG159
1	0	1	0	SEG160	SEG161	SEG162	SEG163	SEG164	SEG165	SEG166	SEG167	SEG168	SEG169	SEG170	SEG171	SEG172	SEG173	SEG174	SEG175
1	0	1	1	SEG176	SEG177	SEG178	SEG179	SEG180	SEG181	SEG182	SEG183	SEG184	SEG185	SEG186	SEG187	SEG188	SEG189	SEG190	SEG191
1	1	0	0	SEG192	SEG193	SEG194	SEG195	SEG196	SEG197	SEG198	SEG199	SEG200	SEG201	SEG202	SEG203	SEG204	SEG205	SEG206	SEG207
1	1	0	1	SEG208	SEG209	SEG210	SEG211	SEG212	SEG213	SEG214	SEG215	SEG216	SEG217	SEG218	SEG219	SEG220	SEG221	SEG222	SEG223
1	1	1	0	SEG224	SEG225	SEG226	SEG227	SEG228	SEG229	SEG230	SEG231	SEG232	SEG233	SEG234	SEG235	SEG236	SEG237	SEG238	SEG239
1	1	1	1	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Table 6 ICON RAM的位址，資料以及Segment接腳的對應表

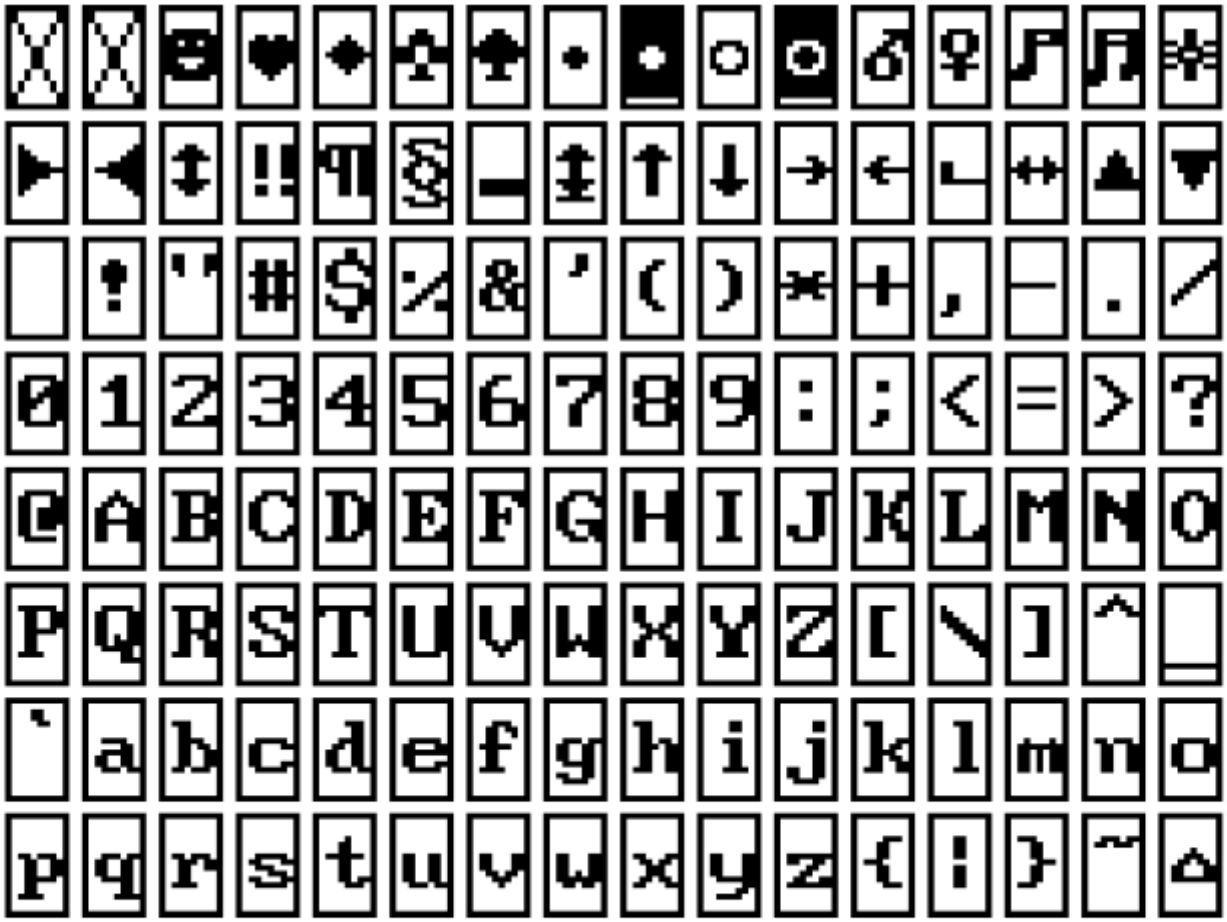


Table 7 16x8 半寬字型符號表

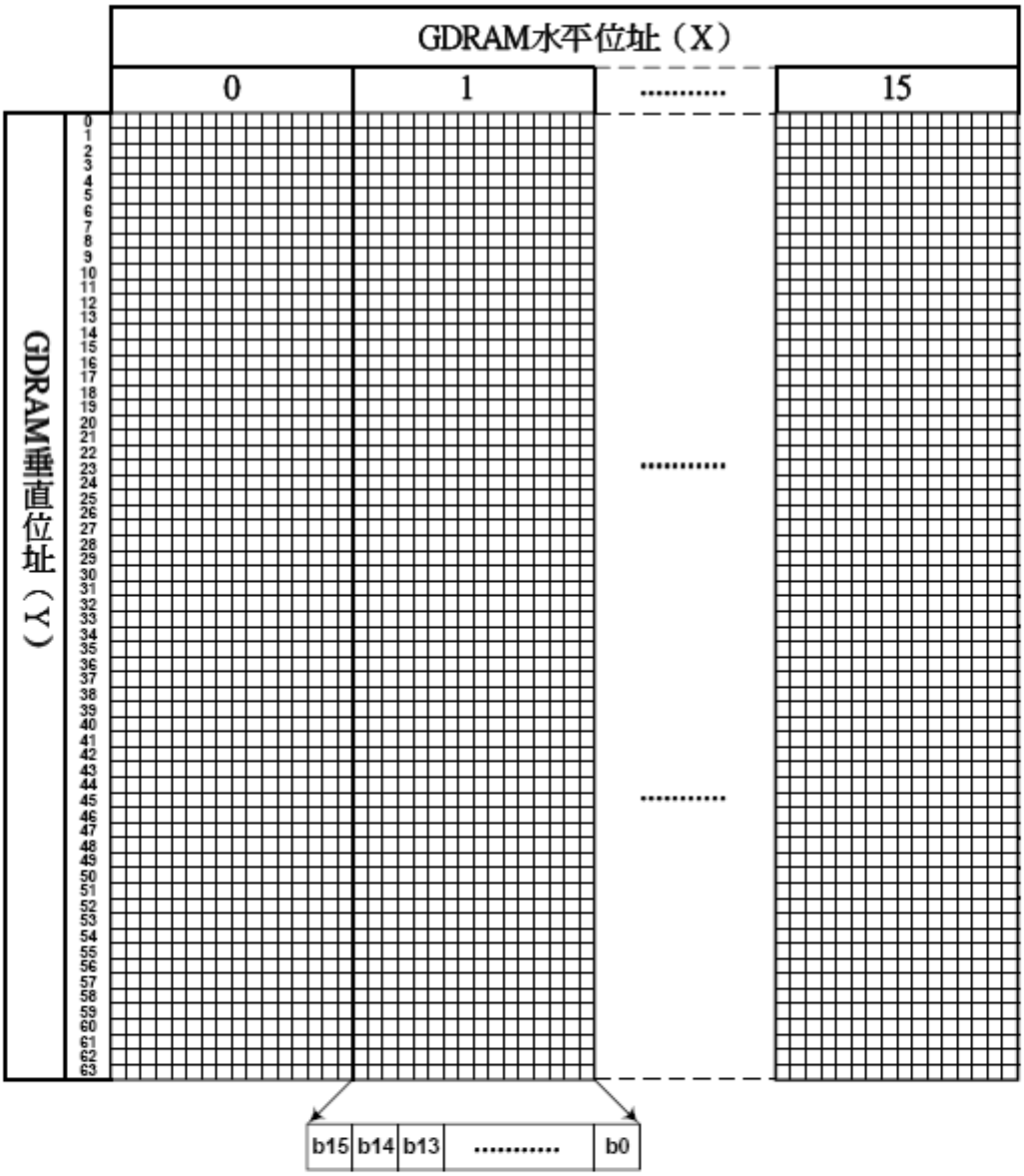


Table 8 GDRAM座標位址與資料排列順序對照表

指令

ST7920 提供兩套控制命令，基本指令和擴充指令如下：

指令表 1: (RE=0: 基本指令集)

指令	指令碼										說明	執行時間 (540KHZ)
	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
清除顯示	0	0	0	0	0	0	0	0	0	1	將 DDRAM 填滿 "20H"，並且設定 DDRAM 的位址計數器（AC）到"00H"	1.6 ms
位址歸位	0	0	0	0	0	0	0	0	1	X	設定 DDRAM 的位址計數器（AC）到"00H"，並且將游標移到開頭原點位置；這個指令並不改變 DDRAM 的內容	72us
進入點設定	0	0	0	0	0	0	0	1	I/D	S	指定在資料的讀取與寫入時，設定游標的移動方向及指定顯示的移位	72us
顯示狀態開/關	0	0	0	0	0	0	1	D	C	B	D=1: 整體顯示 ON C=1: 游標 ON B=1: 游標位置反白 ON	72 us
游標或顯示移位控制	0	0	0	0	0	1	S/C	R/L	X	X	設定游標的移動與顯示的移位控制位元；這個指令並不改變 DDRAM 的內容	72 us
功能設定	0	0	0	0	1	DL	X	0 RE	X	X	DL=1 8-BIT 控制介面 DL=0 4-BIT 控制介面 **RE=1: 擴充指令集動作** **RE=0: 基本指令集動作**	72 us
設定 CGRAM 位址	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	設定 CGRAM 位址到位址計數器（AC） **需確認擴充指令中 SR=0 (捲動位址或 RAM 位址選擇)**	72 us
設定 DDRAM 位址	0	0	1	0 AC6	AC5	AC4	AC3	AC2	AC1	AC0	設定 DDRAM 位址到位址計數器（AC） AC6 固定爲 0	72 us
讀取忙碌旗標（BF）和位址	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	讀取忙碌旗標（BF）可以確認內部動作是否完成，同時可以讀出位址計數器（AC）的值	0 us
寫資料到 RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	寫入資料到內部的 RAM (DDRAM/CGRAM/IRAM/GDRAM)	72 us
讀出 RAM 的值	1	1	D7	D6	D5	D4	D3	D2	D1	D0	從內部 RAM 讀取資料 (DDRAM/CGRAM/IRAM/GDRAM)	72 us

指令表 2: (RE=1: 擴充指令集)

指令	指令碼										說明	執行時間 (540KHZ)
	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
待命模式	0	0	0	0	0	0	0	0	0	1	進入待命模式，，執行任何其他指令都可終止待命模式 (Com1..32 停止動作,只保留 Com33 ICON 顯示)	72 us
捲動位址或 RAM 位址選擇	0	0	0	0	0	0	0	0	1	SR	SR=1: 允許輸入垂直捲動位址 SR=0: 允許輸入 IRAM 位址**(擴充指令)** SR=0: 允許設定 CGRAM 位址**(基本指令)**	72 us
反白選擇	0	0	0	0	0	0	0	1	R1	R0	選擇 4 行中的任一行作反白顯示，並可決定反白與否 **R1,R0 初值爲 00 當第一次設定時爲反白顯示在一次設定時爲正常顯示**	72 us
睡眠模式	0	0	0	0	0	0	1	SL	X	X	SL=1: 脫離睡眠模式 SL=0: 進入睡眠模式	72 us
擴充功能設定	0	0	0	0	1	DL	X	1 RE	G	0	DL=1 8-BIT 控制介面 DL=0 4-BIT 控制介面 **RE=1: 擴充指令集動作** **RE=0: 基本指令集動作** G=1 :繪圖顯示 ON G=0 :繪圖顯示 OFF	72 us
設定 IRAM 位址或捲動位址	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	SR=1: AC5~AC0 爲垂直捲動位址 SR=0: AC3~AC0 爲 ICON RAM 位址	72 us
設定繪圖 RAM 位址	0	0	1	0 AC6	0 AC5	0 AC4	AC3 AC3	AC2 AC2	AC1 AC1	AC0 AC0	設定 GDRAM 位址到位址計數器（AC） 先設垂直位址再設水平位址(連續寫入兩個位元組的資料來完成垂直與水平的座標位址) 垂直位址範圍 AC6...AC0 水平位址範圍 AC3...AC0	72 us

備註：

1. 當 ST7920 在接受指令前，微處理器必須先確認 ST7920 內部處於非忙碌狀態，即讀取 BF 旗標時 BF 需爲 0，方可接受新的指令；如果在送出一個指令前並不檢查 BF 旗標，那麼在前一個指令和這個指令中間必須延遲一段較長的時間，即是等待前一個指令確實執行完成，指令執行的時間請參考指令表中的個別指令說明。
2. “RE”爲基本指令集與擴充指令集的選擇控制位元，當變更“RE”位元後，往後的指令集將維持在最後的狀態，除非再次變更“RE”位元，否則使用相同的指令集時，不需每次重設“RE”位元。

指令集初始值(Register flag) (RE=0: 基本指令集)

指令	指令碼										說明
	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
進入點設定	0	0	0	0	0	0	0	1	I/D	S	游標右移,DDRAM 位址計數器（AC）加 1
									1	0	
顯示狀態開/關	0	0	0	0	0	0	1	D	C	B	控制整體顯示,游標,游標位置反白 ALL OFF
								0	0	0	
游標或顯示移位控制	0	0	0	0	0	1	S/C	R/L	X	X	無游標與顯示移位動作
							X	X			
功能設定	0	0	0	0	1	DL	X	0 RE	X	X	8 BIT MPU 控制界面，基本指令集動作
						1		0			

指令集初始值(Register flag) (RE=1: 擴充指令集)

指令	指令碼										說明
	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
捲動位址或 RAM 位址選擇	0	0	0	0	0	0	0	0	1	SR	允許輸入 IRAM 位址 or 設定 CGRAM 位址
										0	
反白選擇	0	0	0	0	0	0	0	1	R1	R0	當第一次設定時爲反白顯示再一次設定時爲正常顯示
									0	0	
睡眠模式	0	0	0	0	0	0	1	SL	X	X	未進入待命模式
								0			
擴充功能設定	0	0	0	0	1	DL	X	1 RE	G	0	繪圖顯示 OFF
									0		

基本指令集說明

- **清除顯示**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	0	0	0	0	0	0	1

將 DDRAM 填滿 "20H"(space code)，並且設定 DDRAM 的位址計數器(AC)到"00H",重設進入點設定將 I/D 設爲 "1" 游標右移 AC 加 1

- **位址歸位**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	0	0	0	0	0	1	x

設定 DDRAM 的位址計數器（AC）到"00H"，並且將游標移到開頭原點位置；這個指令並不改變 DDRAM 的內容

- **進入點設定**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	0	0	0	0	1	I/D	S

指定在資料的讀取與寫入時，設定游標的移動方向及指定顯示的移位

I/D :位址計數器遞增遞減選擇

當 I/D = "1", 游標右移,DDRAM 位址計數器（AC）加 1

當 I/D = "0", 游標左移,DDRAM 位址計數器（AC）減 1

S: 顯示畫面整體位移

S	I/D	DESCRIPTION
H	H	畫面整體左移
H	L	畫面整體右移

- **顯示狀態開關**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	0	0	0	1	D	C	B

控制整體顯示,游標,游標位置反白 ON/OFF

D : 整體顯示 ON/OFF 控制位元

當 D = "1",整體顯示 ON

當 D = "0",整體顯示 OFF ,但不改變 DDRAM 的內容

C : 游標 ON/OFF 控制位元

當 C = "1",游標顯示 ON.

當 C = "0",游標顯示 OFF.

B : 游標位置反白 ON/OFF 控制位元

當 B = "1",游標位置顯示反白 ON,將游標所在之位址上的資料反白顯示.

當 B = "0", .游標位置顯示反白 OFF

- **游標或顯示移位控制**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	0	0	1	S/C	R/L	x	x

設定游標的移動與顯示的移位控制位元；這個指令並不改變 DDRAM 的內容

S/C	R/L	Description	AC Value
L	L	游標向左移動	AC=AC-1
L	H	游標向右移動	AC=AC+1
H	L	顯示(display)向左移動,且游標跟這移動	AC=AC
H	H	顯示(display)向右移動,且游標跟這移動	AC=AC

- **功能設定**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	0	1	DL	X	RE	x	x

DL : 4/8 BIT 界面控制位元

當 DL = "1", 爲 8 BIT MPU 控制界面

當 DL = "0", 爲 4 BIT MPU 控制界面

RE : 指令集選擇控制位元

當 RE = "1", 爲擴充指令集動作

當 RE = "0", 爲基本指令集動作

同一指令之動作不可同時改變 RE 及 DL 需先改變 DL 後在改變 RE 才可確保 FLAG 正確設定

- **設定 CGRAM 位址**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0

設定 CGRAM 位址到位址計數器（AC）

AC 範圍爲 00H..3FH

需確認擴充指令中 SR=0 (捲動位址或 RAM 位址選擇)

- **設定 DDRAM 位址**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0

設定 DDRAM 位址到位址計數器（AC）.

第一行 AC 範圍爲 80H..8FH

第二行 AC 範圍爲 90H..9FH

第三行 AC 範圍爲 A0H..AFH

第四行 AC 範圍爲 B0H..BFH

- **讀取忙碌旗標（BF）和位址**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0

讀取忙碌旗標（BF）可以確認內部動作是否完成，同時可以讀出位址計數器（AC）的值

當 BF = "1", 表示內部忙碌中此時不可下指令需等 BF = "0"才可下新指令.

- **寫入資料到 RAM**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	1	0	D7	D6	D5	D4	D3	D2	D1	D0

寫入資料到內部的 RAM 當寫入後會使 (AC) 改變

每個 RAM 位址(CGRAM,DDRAM,IRAM.....)都可連續寫入兩個位元組的資料(2-Bytes)當寫入第二 BYTE 時位址計數器（AC）的值就會自動加一

- **讀取 RAM 的值**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	1	1	D7	D6	D5	D4	D3	D2	D1	D0

從內部的 RAM 讀取資料, 當讀取後會使 (AC) 改變

當下設定位址指令後(CGRAM,DDRAM,IRAM.....)若要讀取資料時需先 DUMMY READ 一次才會讀取到正確資料

第二次讀取時則不需 DUMMY READ 除非又下設定位址指令才需再次 DUMMY READ。

擴充指令集說明

- **待命模式**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	0	0	0	0	0	0	1

進入待命模式，，執行任何其他指令都可終止待命模式；這個指令並不改變 RAM 的內容

- **捲動位址或 RAM 位址選擇**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	0	0	0	0	0	1	SR

當 SR = "1",允許輸入垂直捲動位址

當 SR = "0",允許輸入 IRAM 位址(擴充指令)及允許設定 CGRAM 位址(基本指令)

- **反白選擇**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	0	0	0	0	1	R1	R0

選擇 4 行中的任一行作反白顯示，並可決定反白與否

R1,R0 初值爲 00 當第一次設定時爲反白顯示再一次設定時爲正常顯示

R1	R0	Description
L	L	第一行反白或正常顯示
L	H	第二行反白或正常顯示
H	L	第三行反白或正常顯示
H	H	第四行反白或正常顯示

- **睡眠模式**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	0	0	0	1	SL	0	0

SL=1: 脫離睡眠模式

SL=0: 進入睡眠模式

- **擴充功能設定**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	0	1	DL	x	RE	G	x

DL : 4/8 BIT 界面控制位元

當 DL = "1", 爲 8 BIT MPU 控制界面

當 DL = "0", 爲 4 BIT MPU 控制界面

RE : 指令集選擇控制位元

當 RE = "1", 爲擴充指令集動作

當 RE = "0", 爲基本指令集動作

G : 繪圖顯示控制位元

當 G = "1", 繪圖顯示 ON

當 G = "0", 繪圖顯示 OFF

同一指令之動作不可同時改變 RE 及 DL, G 需先改變 DL 或 G 後在改變 RE 才可確保 FLAG 正確設定

- **設定 IRAM 位址或捲動位址**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0

SR=1: AC5~AC0 爲垂直捲動位址

SR=0: AC3~AC0 爲 ICON RAM 位址

- **設定繪圖 RAM 位址**

	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Code	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0

設定 GDRAM 位址到位址計數器（AC）

先設垂直位址再設水平位址(連續寫入兩個位元組的資料來完成垂直與水平的座標位址)

垂直位址範圍 AC6...AC0

水平位址範圍 AC3…AC0

繪圖 RAM 之位址計數器（AC）只會對水平位址(X 軸)自動加一,當水平位址=0FH 時會重新設爲 00H 但並不會對垂直位址做進位自動加一故當連續寫入多筆資料時程式需自行判斷垂直位址是否需重新設定

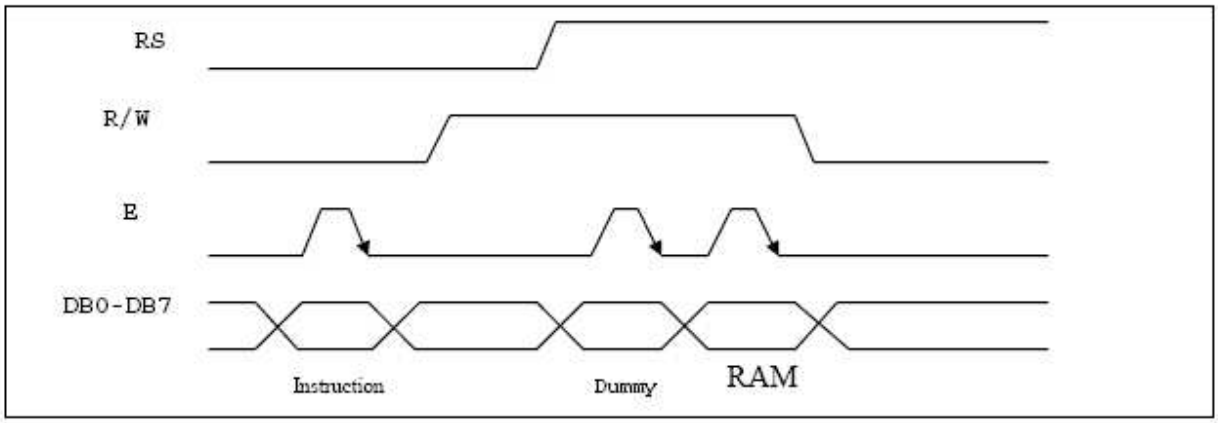
並列介面資料傳輸訊號

當PSB腳接高電位時，ST7920將進入並列模式，在並列模式下可由指令 **DL FLAG** 來選擇8-位元或4-位元介面，主控制系統將配合(RS , RW , E , DB0..DB7)來達成傳輸動作。

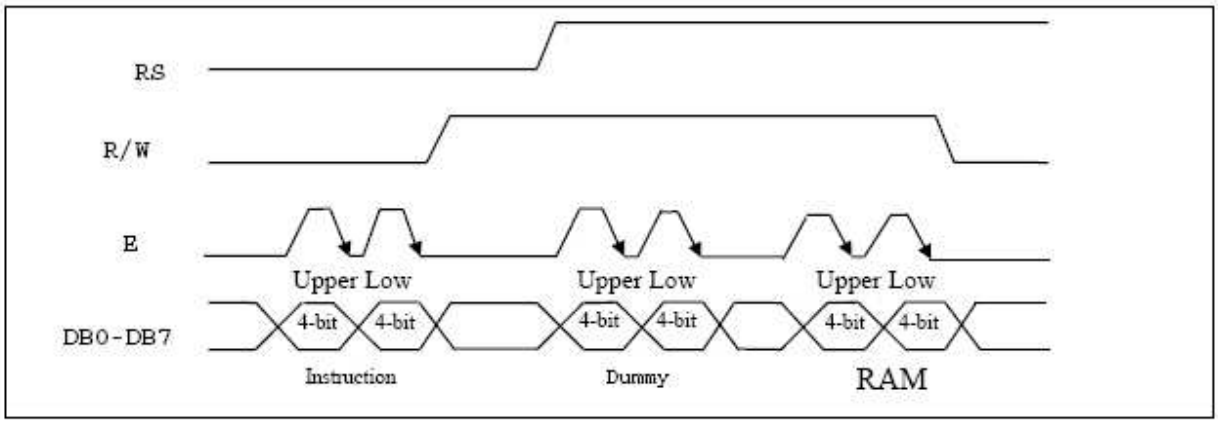
從一個完整的流程來看，當下設定位址指令後(CGRAM,DDRAM,IRAM.....)若要讀取資料時需先 DUMMY READ 一次才會讀取到正確資料第二次讀取時則不需 DUMMY READ 除非又下設定位址指令才需再次 DUMMY READ。

在4-位元傳輸模式中，每一個八位元的指令或資料都將被分爲兩個位元組動作：較高4位元（DB7~DB4）的資料將會被放在第一個位元組的（DB7~DB4）部分，而較低4位元（DB3~DB0）的資料則會被放在第二個位元組的（DB7~DB4）部分，至於相關的另四位元則在4-位元傳輸模式中DB3~DB0介面未使用。

相關介面傳輸訊號請參考下圖說明：



Timing Diagram of 8-bit Parallel Bus Mode Data Transfer



Timing Diagram of 4-bit Parallel Bus Mode Data Transfer

串列介面與串列傳輸資料

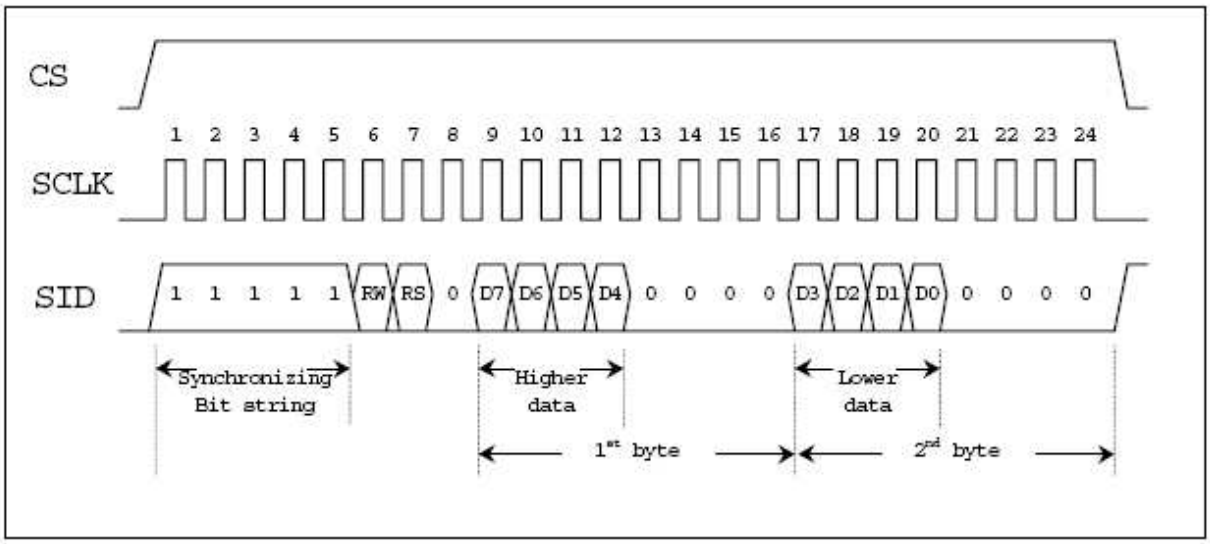
當PSB腳接低電位時，ST7920將進入串列模式，在串列模式下將使用兩條資料傳輸線作串列資料的傳送，主控制系統將配合傳輸同步時脈線（SCLK）與接收串列資料線（SID），來達成串列傳輸的動作。

當需要同時連接數顆ST7920晶片時，晶片選擇腳（CS）將要被配合使用，在晶片選擇腳（CS）設爲高電位時，同步時脈線（SCLK）輸入的訊號才會被接收，另一方面，當晶片選擇腳（CS）設爲低電位時，ST7920的內部串列傳輸計數與串列資料將會被重置，也就是說在此狀態下，傳輸中的資料將被終止清除，並且將待傳輸的串列資料計數重設回第一位元；在一個最小的系統架構下，由一個微處理器連接控制單一個ST7920晶片時，相關的連接介面只需要使用同步時脈線（SCLK）與接收串列資料線（SID）兩隻腳，在這個模式下晶片選擇腳（CS）將被固定接到高電位。

ST7920的同步時脈線（SCLK）具有獨立的操作時脈，但是當有連續多個指令需要被傳送時，指令執行的時間將需要被考慮，必須確實等到前一個指令完全執行完成才能傳送下一筆資料，因爲ST7920內部並沒有傳送/接收緩衝區。

從一個完整的串列傳輸流程來看，一開始先傳輸啓始位元組，它需先接收到五個連續的"1"（同步位元字串）在啓始位元組，此時傳輸計數將被重置並且串列傳輸將被同步，再跟隨的兩個位元字串分別指定傳輸方向位元（RW）及暫存器選擇位元（RS），最後第八的位元則爲"0"。

在接收到同步位元及RW和RS資料的啓始位元組後，每一個八位元的指令將被分爲兩個位元組接收到：較高4位元（DB7~DB4）的指令資料將會被放在第一個位元組的LSB部分，而較低4位元（DB3~DB0）的指令資料則會被放在第二個位元組的LSB部分，至於相關的另四位元則都爲0。
串列傳輸訊號請參考下圖說明：



Timing Diagram of Serial Mode Data Transfer

ZX12232G (ST7920) 并口演示程序 Keil C51

```
//                    DV12232G 测试程序(并口)
//*************************************************************************
//连线表:  CPU=89C52                                                    *
//RS=P2.0          RW=P2.1          E=/(WR*RD)                          *
//FOSC=12MHz       D0-D7=P0.0-P0.7  /RSET=/(CPU RSET)                   *
//*************************************************************************
#include <reg52.h>
#include <stdlib.h>
#include <intrins.h>
#include <stdio.h>

char xdata LcmWriteCom   _at_ 0x0000;      //写指令
char xdata LcmWriteData _at_ 0x0100;      //写数据
char xdata LcmReadBF     _at_ 0x0200;      //读 BF&AC
char xdata LcmReadData   _at_ 0x0300;      //读数据
sbit Key=P3^4;

unsigned char code AC_TABLE[]={
0x80,0x81,0x82,0x83,0x84,0x85,0x86,0x87,       //第一行汉字位置
0x90,0x91,0x92,0x93,0x94,0x95,0x96,0x97,       //第二行汉字位置
};
unsigned char code str2[]="欢迎光临 dvlcd!!";
unsigned char code str1[]="迪威液晶显示技术欢迎您！内含八千汉字库。";
unsigned char code bmp1[];

void CheckBusy( void )
{
    while(LcmReadBF&0x80);       //BF=1 Busy
}

void WriteCommand( unsigned char Cbyte )
{
    CheckBusy();
    LcmWriteCom = Cbyte;
}

void WriteData( unsigned char Dbyte )
{
    CheckBusy();
    LcmWriteData = Dbyte;
}

unsigned char ReadData( void )
{
    CheckBusy();
    return LcmReadData;
```

```
}

void Delay(unsigned int MS)
{
    unsigned char us,usn;
    while(MS!=0)                //for 12M
        { usn = 2;
            while(usn!=0)
                {
                    us=0xf5;
                    while (us!=0){us--;};
                    usn--;
                }
            MS--;
        }
}

//迪威液晶测试架专用延时函数
void DelayKey(unsigned int Second , unsigned int MS100)
{                                   //输入精确到 0.1S,是用","
    unsigned int i;
    for(i=0;i<Second*100+MS100*10;i++)
    {
        if(Key==0)
        {
            Delay(20);
            while(Key==0) {Delay(20);}
            break;
        }
        else Delay(10);
    }
}

void LcmInit( void )
{
    WriteCommand(0x30);       //8BitMCU,基本指令集合
    WriteCommand(0x03);       //AC 归 0,不改变 DDRAM 内容
    WriteCommand(0x0C);       //显示 ON,游标 OFF,游标位反白 OFF
    WriteCommand(0x01);       //清屏,AC 归 0
    WriteCommand(0x06);       //写入时,游标右移动
}

//文本区清 RAM 函数
void LcmClearTXT( void )
{
    unsigned char i;
    WriteCommand(0x30);       //8BitMCU,基本指令集合
    WriteCommand(0x80);       //AC 归起始位
    for(i=0;i<64;i++)
        WriteData(0x20);
```

```
}

//图形区和文本区显示在两个不同的 RAM 区
//图形区清 RAM 函数
void LcmClearBMP( void )
{
    unsigned char i,j;
    WriteCommand(0x34);        //8Bit 扩充指令集,即使是 36H 也要写两次
    WriteCommand(0x36);        //绘图 ON,基本指令集里面 36H 不能开绘图
    for(i=0;i<32;i++)          //12864 实际为 256x32
    {
        WriteCommand(0x80|i);      //行位置
        WriteCommand(0x80);        //列位置
        for(j=0;j<32;j++)          //256/8=32 byte
            WriteData(0);
    }
}


void PutStr(unsigned char row,unsigned char col,unsigned char *puts)
{
    WriteCommand(0x30);        //8BitMCU,基本指令集合
    WriteCommand(AC_TABLE[8*row+col]);        //起始位置
    while(*puts != '\0')       //判断字符串是否显示完毕
    {
        if(col==8)             //判断换行
        {                      //若不判断,则自动从第一行到第三行
            col=0;
            row++;
        }
        if(row==2) row=0;      //一屏显示完,回到屏左上角
        WriteCommand(AC_TABLE[8*row+col]);
        WriteData(*puts);      //一个汉字要写两次
        puts++;
        WriteData(*puts);
        puts++;
        col++;
    }
}

void ReadDemo( void )
{
    unsigned char i;
    unsigned char x,y;
    PutStr(0,0,str2);          //显示一行文字
    for(i=0;i<8;i++)           //只操作第一行
    {
        WriteCommand(0x80|i);
        x = ReadData();        //假读一次,无作用
        x = ReadData();        //第二个读取才能正确
```

```
            y = ReadData();       //每个循环读和写要两次
                                  //每个 AC 地址有高低两个字节
            WriteCommand(0x90|i);       //将读取的内容写到第二行
            WriteData(x);
            WriteData(y);
        }
    }

    void PutBMP(unsigned char *puts)
    {
        unsigned int x=0;
        unsigned char i,j;
        WriteCommand(0x34);       //8Bit 扩充指令集,即使是 36H 也要写两次
        WriteCommand(0x36);       //绘图 ON,基本指令集里面 36H 不能开绘图
        for(i=0;i<32;i++)         //12864 实际为 256x32
        {
            WriteCommand(0x80|i);       //行位置
            WriteCommand(0x80);       //列位置
            for(j=0;j<16;j++)       //122/8=15.25=16(只能为整数)
            {                         //列位置每行自动增加
                WriteData(puts[x]);
                x++;
            }
        }
    }

    //迪威液晶测试用点阵显示
    void DisplayDots(unsigned char DotByte)
    {
        unsigned char i,j;
        WriteCommand(0x34);       //8Bit 扩充指令集,即使是 36H 也要写两次
        WriteCommand(0x36);       //绘图 ON,基本指令集里面 36H 不能开绘图
        for(i=0;i<32;i++)         //12864 实际为 256x32
        {
            WriteCommand(0x80|i);       //行位置
            WriteCommand(0x80);       //列位置
            for(j=0;j<16;j++)       //122/8=15.25=16(只能为整数)
            {                         //列位置每行自动增加
                WriteData(DotByte);
            }
            DotByte=~DotByte;
        }
    }

    void main( void )
    {
        Delay(100);       //等待复位
        LcmInit();
        LcmClearTXT();
        LcmClearBMP();
```

```
    while(1)
    {
        LcmClearTXT();
        PutBMP(bmp1);
        DelayKey(1,5);

        DisplayDots(0x55);
        DelayKey(1,5);

        LcmClearBMP();
        PutStr(0,0,str1);
        DelayKey(1,5);

        LcmClearTXT();
        ReadDemo();
        DelayKey(1,5);

        LcmClearTXT();
        DisplayDots(0xaa);
        DelayKey(1,5);
    }
}

unsigned char code bmp1[]={
//*************************************************************//
//*****  转换的文件：E:\!Program\!BmpSample\12832_12232.bmp
//*****  源图形宽度 * 高度：128 * 32
//*****  调整后宽度 * 高度：128 * 32
//*****  字模格式：横向取模,冗余模式,字节正序,正色取模,
//*****  点阵转换时间：2005/03/15    13:37:14
//*****  位图点阵占用 512 字节
//*************************************************************//

0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,
0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,
0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,
0x80,0x00,0x00,0x30,0x00,0x00,0x00,0x00,0x00,0x18,0x00,0x00,0x0C,0x00,0x00,0x41,
0x80,0x00,0x00,0x30,0x00,0xE0,0xF0,0x00,0x18,0x30,0x00,0x07,0x18,0x00,0x00,0x41,
0x80,0x00,0x00,0x7F,0x80,0x63,0xCF,0xF0,0x18,0x30,0x30,0x06,0x18,0x00,0x00,0x41,
0x80,0x03,0xFF,0x7F,0xE0,0x76,0x0C,0x30,0x1C,0x30,0x60,0x66,0x3F,0xFE,0x00,0x41,
0x80,0x00,0x07,0xC0,0x60,0x3E,0x18,0x60,0x1C,0x60,0xC0,0x6C,0x30,0x00,0x00,0x41,
0x80,0x00,0x06,0xCC,0xE0,0x0C,0x18,0x60,0x0C,0x61,0x80,0xCC,0x60,0x00,0x00,0x41,
0x80,0x03,0x07,0x98,0xC0,0x0C,0x38,0xE0,0x0C,0xE3,0x00,0xCC,0xCC,0x00,0x00,0x41,
0x80,0x03,0x0D,0x99,0xC0,0x1C,0x30,0xC0,0x0C,0xC6,0x00,0xD8,0xCF,0x00,0x00,0x41,
0x80,0x03,0x1F,0x19,0x87,0xD8,0x30,0xC0,0x00,0xC0,0x01,0x99,0x83,0x80,0x00,0x41,
0x80,0x03,0x9F,0x33,0x00,0xD8,0x71,0xC3,0xFF,0xFF,0xF1,0x9B,0x01,0xC0,0x00,0x41,
0x80,0x01,0xB0,0x37,0x01,0xF8,0x61,0x80,0x0C,0x30,0x01,0xB6,0x00,0xE0,0x00,0x41,
0x80,0x01,0xE0,0x70,0x01,0xB0,0x61,0x80,0x0C,0x70,0x03,0x30,0x00,0x00,0x00,0x41,
0x80,0x01,0xE0,0x70,0x01,0xB0,0xE3,0x80,0x1C,0x60,0x03,0x33,0xFF,0xC0,0x00,0x41,
```

0x80,0x01,0xC0,0xF0,0x03,0x60,0xC3,0x00,0x18,0x60,0x03,0x67,0x18,0xC0,0x00,0x41,
0x80,0x03,0xC1,0xF0,0x03,0x67,0xC3,0x00,0x30,0xE0,0x06,0x66,0x30,0xC0,0x00,0x41,
0x80,0x07,0xC3,0xB0,0x03,0x7F,0xC7,0x00,0x30,0xC0,0x06,0x6E,0x30,0xC0,0x00,0x41,
0x80,0x0E,0xE3,0x30,0x06,0x71,0xF6,0x00,0x60,0xC0,0xC6,0xCC,0x31,0x80,0x00,0x41,
0x80,0x1C,0xE6,0x38,0x06,0x01,0xBC,0x00,0xC1,0xC1,0x8C,0xCC,0x61,0x80,0x00,0x41,
0x80,0x38,0xEC,0x38,0x0F,0x03,0x80,0x01,0x81,0x81,0x8C,0xD8,0x61,0x80,0x00,0x41,
0x80,0x60,0x78,0x18,0x1B,0x83,0x00,0x07,0x01,0x83,0x0D,0x98,0x63,0x00,0x00,0x41,
0x80,0xC0,0x70,0x1C,0x71,0xC6,0x00,0x1E,0x03,0x87,0x01,0x98,0xC3,0x00,0x00,0x41,
0x80,0x00,0xE0,0x0C,0xE0,0xFF,0xFE,0x38,0x01,0xFE,0x01,0x9F,0xFE,0x00,0x00,0x41,
0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,
0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,
0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,
0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,
0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,
0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,