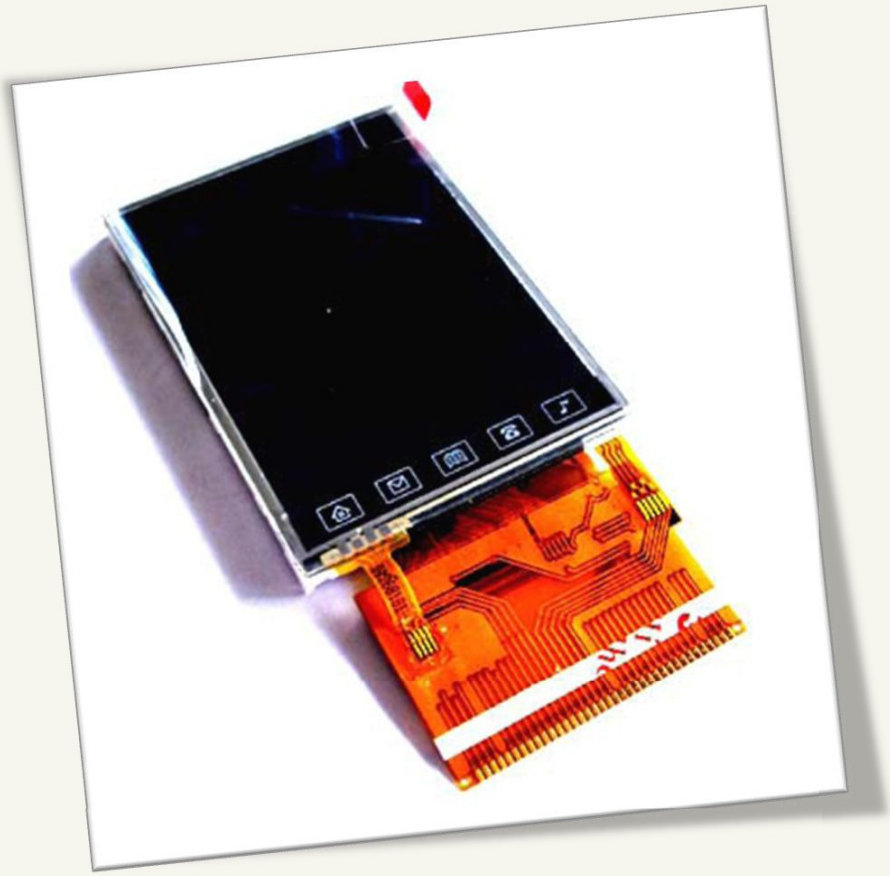
راه اندازی LCDN96 (معروف به N96 چینی)

در این بخش قصد این را داریم که در مورد یکی از ماژول های نمایشگر LCD که به تازگی در بین علاقه مندان الکترونیک رواج پیدا کرده را توضیح بدهیم . این LCD هنوز به طور کامل شناسایی نشده (از صدقه سر این چینی ها) ولی یک سری مشخصاتی هست که می توان آن را شناسایی کرد که در ادامه توضیح خواهیم داد . بعد از آن شروع به تعریف و توضیح پایه های این ماژول کرده و بعد از آن کتابخانه ی این LCD برای شما عزیزان تشریح خواهد شد . قیمت در بازار تهران در این تاریخ حدودا ۱۴۰۰۰/۱۲۰۰۰ تومان به فروش می رسد.



از مهمترین خصوصیات این LCD در زیر به آن اشاره شده است :

- ✔ ابعاد حدود ۳ اینچ
- ✔ تعداد پیکسل ۲۴۰ در ۳۲۰
- ✔ قابلیت تفکیک ۲۶۲ هزار رنگ
- ✔ دارای دو مد دیتا باس ۸ و ۱۶ بیتی
- ✔ توان مصرفی بسیار پایین (بدون بک لایت) و کار کردن در رنج ولتاژ ۳.۳ ولت
- ✔ دارای یک تاچ اسکرین تعبیه شده روی ماژول

اولین مرحله ای که برای شروع به کار آن نیاز دارید ، همانطور که گفته شد ، شناسایی این LCD و انتخاب صحیح آن می باشد . بر اساس آماری که گرفته شده حدود ۲۰۰ الی ۳۰۰ نوع LCD گوشی های چینی در بازار ایران وجود دارد که فقط ۵ تا ۱۰ مورد آن ها ، قابل راه اندازی (تا الان) است .

اولین نکته اینکه به هیچ وجه به PCB منعطف ماژول و نحوه ی طراحی آن و سیم کشی های روی این LCD توجه نکنید . ماژول های مشابه دارای طراحی های مشابه نمی باشند . تنها و تنها موردی که باید برای پیدا کردن این LCD باید توجه کرد ، اندازه ی آن می باشد . در جدول زیر این اندازه و تعدادی از مشخصات آن را مشاهده می کنید .

Parameter	Value	Unit
LCD Mode	a-Si TFT/transmissive	-
Color	262K	-
Display Resolution	240*RGB*320	pixels
OUTLINE DIMEMSIONS	50.00(W) x69.20(H) x4.05(T)	mm
Active Area(A.A)	43.20 (W) x 57.60(H)	mm
Pixel Arrangement	RGB-stripe	-
Viewing Direction	12 O'clock	
Display Mode	Normally white	
LCD Controller/Driver	ILI9325	-
IC Package Type	COG	-
MPU interface	Standard 8080 system18-/16-bit paraller	-
Power Supply Voltage	2.5~3.3	V
Back-light	White LED*4	pcs

فروشنده های این ماژول ها هم این LCD را با نام LCD گوشی N96 چینی می شناسند . اما فاکتور اصلی انتخاب نمی باشد . بعد از در دست گرفتن ماژول مشخصات زیر باید از روی PCB منعطف مشهود باشد .

۱ – تعداد پایه ها حتما ۳۷ تا باشد .

۲ – پایه های ۱۲ تا ۱۵ به Touch Panel ال سی دی رفته باشد .

۳ – پایه های ۱۶ تا ۲۰ به قسمت بک لایت ماژول رفته باشد .

۴ – پایه ی ۲۱ ، NC (Not Connected) باشد .

۵ – پایه های ۵ و ۳۴ به خطوط زخیم تر از پایه های دیگر متصل شده باشد .

این چند موارد تنها ویژگی ظاهری این ماژول می باشد که ملاک اصلی انتخاب شما برای خرید این LCD است . بعد از خرید این ماژول نوبت به تبدیل پایه های خروجی این ماژ ول به صورتیکه بتوان روی برد بورد از آن استفاده کرد ، است . نکته ی

اول اینکه سوکتی برای این ماژول وجود ندارد . نکته ی دوم هم اینکه بهترین و ارزانترین راه لحیم پایه های این ماژول به سیم های حتی الامکان نازک است .

در مورد کنترلر این LCD هم باید گفت که معروفترین آن ها ILI9325 یا ILI9320 است که برگه ی اطلاعاتی این کنترلر در اینترنت به کثرت وجود دارد . کنترلر های دیگه ای هم وجود دارد که با دقت کردن در آن ها قابل مشاهده است که اینترفیس و کدهای دستورالعمل مشابهی نسبت به دو کنترلر ذکر شده دارند . در این مقاله تنها دو کنترلر گفته شده مورد بررسی قرار خواهند گرفت .

خود ماژول هم یک دیتاشیت مخصوص به خود دارد که به نام ELT240320 مشخص شده است که اطلاعاتی در مورد نام پایه ها ، ابعاد ماژول و .... در آن وجود دارد . در اینجا نیز در مورد پایه های این LCD بیشتر توضیح خواهیم داد . شکل صفحه ی بعد شماره و نام پایه ها را نشان می دهد .

- پایه GND و VCC : این دو پایه که مجموعا ۵ پایه از پایه های ماژول را شامل می شوند ، وظیفه ی تغذیه ی LCD را به عهده دارند که در برگ اطلاعاتی ماژول به ولتاژ حدود ۲.۵ الی ۳.۳ ولت اشاره شده است .
- پایه
- پایه های DB1-DB8 و DB10-DB17 : این ۱۶ پایه دیتا باس (گذرگاه داده) بین کنترلر LCD و میکروکنترلر می باشند . داده ها و کد دستورات از طریق این خطوط به LCD منتقل یا دریافت می شوند .
- پایه RESET : این پایه در صورتیکه صفر شود کنترلر LCD را بازرشانی می کند .
- پایه RS : این پایه وظیفه ی انتخاب رجیستر دستورالعمل و یا رجیستر داده را دارد .
- پایه CS : پایه ی انتخاب تراشه (Chip Select)
- پایه WR : در صورتیکه این پایه صفر شود ، قصد نوشتن داده یا دستور در رجیستر های دستورالعمل یا داده را داریم .

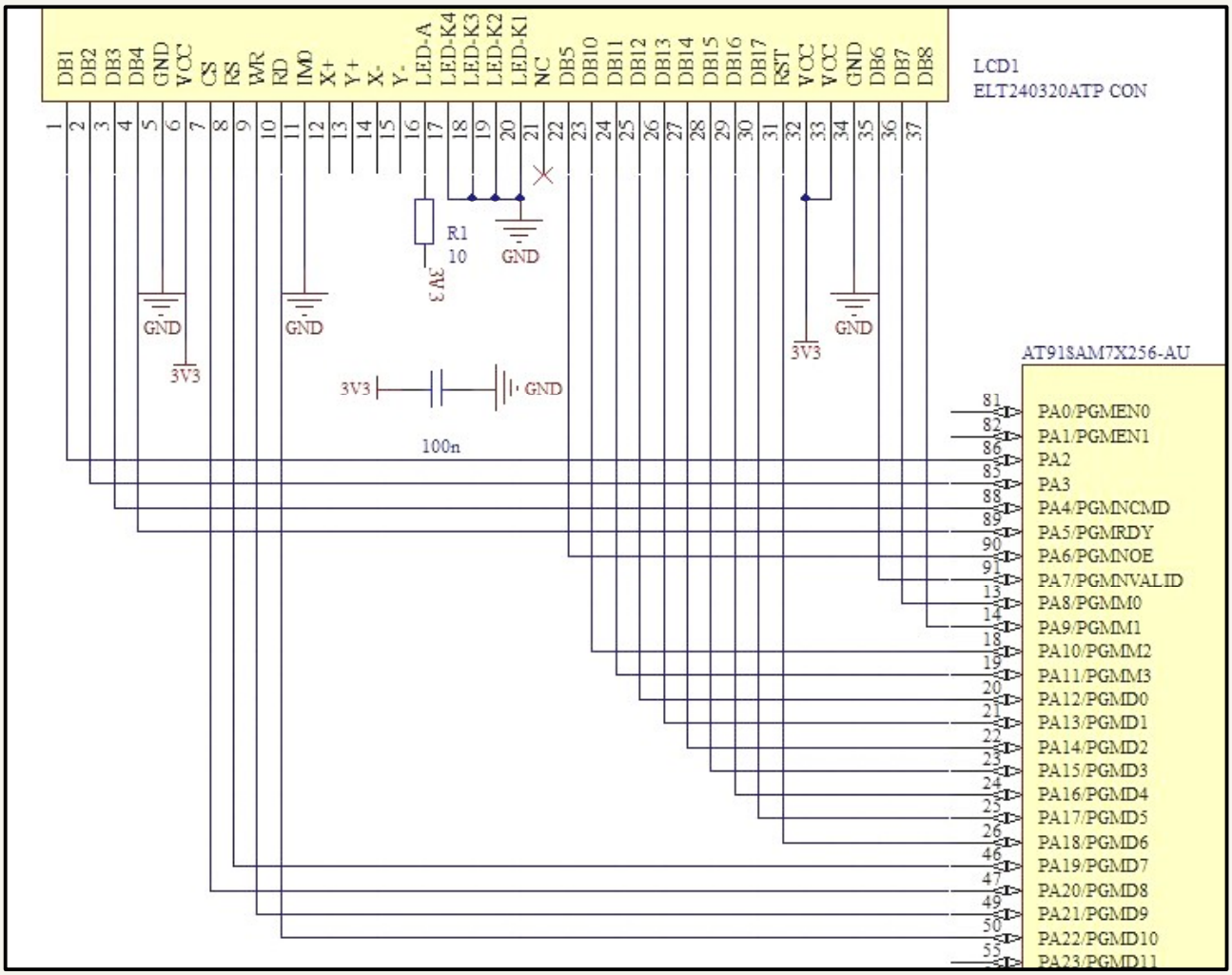
No.	Symbol	Functional	Remark
1	DB1	Data bus	
2	DB2	Data bus	
3	DB3	Data bus	
4	DB4	Data bus	
5	GND	Ground	
6	VCC	Power	
7	CS	Chip select pin of serial inter face	
8	RS	Data or command	
9	WR	Write signal	
10	RD	Read signal	
11	IMO	Interface mode select	
12	X+	Touch panel X+	
13	Y+	Touch panel Y+	
14	X-	Touch panel X-	
15	Y-	Touch panel Y-	
16	LED-A	LED A	
17	LED-K4	LED K4	
18	LED-K3	LED K3	
19	LED-K2	LED K2	
20	LED-K1	LED K1	
21	NC	No connection	
22	DB5	Data bus	
23	DB10	Data bus	
24	DB11	Data bus	
25	DB12	Data bus	
26	DB13	Data bus	
27	DB14	Data bus	
28	DB15	Data bus	
29	DB16	Data bus	
30	DB17	Data bus	
31	REST	Reset din	
32	VCC	Power	
33	VCC	Power	
34	GND	Ground	
35	DB6	Data bus	
36	DB7	Data bus	
37	DB8	Data bus	

- پایه RD : در صورتیکه این پایه صفر شود ، قصد خواندن داده از روی LCD را داریم .
- IM0 : این پایه وظیفه ی انتخاب مد دیتاباس را دارد . در صورتی که یک شود مد ۸ بیت و در غیر اینصورت مد ۱۶ بیت انتخاب خواهد شد .
- پایه های X+ ، X- ، Y+ و Y- : این چهار پایه خطوط متصل به صفحه ی لمسی LCD می باشند .
- پایه های LED-A ، LED-K1 ، LED-K2 ، LED-K3 ، LED-K4 : پایه LED-A پایه ی آنود چهار دیود نورانی پشت صفحه (Back Light) و چهار پایه ی LED-K1 تا LED-K4 ، پایه های کاتد هر کدام از دیود های نورانی پشت صفحه هستند .
- پایه NC : پایه ی بدون اتصال

بعد از توضیحات تقریبا اجمالی در مورد ماژول LCD سراغ بررسی توابع نوشته شده در فایل سرآمد tftlcd_functions.h که جهت راه اندازی LCD مورد استفاده قرار می گیرد ، می پردازیم . نحوه ی اجرا و کارکرد توابع به عهده ی خواننده گذاشته می شود و در اینجا تنها به توضیح توابع و آرگومان های ارسالی به LCD پرداخته می شود .اولین مرحله تعیین و تعریف نحوه ی اتصال ماژول به میکروکنترلر می باشد . همانطور گفته شد ، ماژول LCD دارای ۱۶ پایه ی دیتا باس و ۵ پایه ی کنترل می باشد . برای تعریف نحوه ی اتصال از تعاریف پیش پردازنده های #define زیر کمک می گیریم .

```
/* LCD Pin Configuration */
#define TFTLCD_DATAPORT_x
#define TFTLCD_DATAPORT_OFFSET
#define TFTLCD_CONTROLPORT_x
#define TFTLCD_RST
#define TFTLCD_RS
#define TFTLCD_CS
#define TFTLCD_WR
#define TFTLCD_RD
```

تکه برنامه ی بالا پورت های اتصالی از میکروکنترلر را به دیتا پورت و کنترل پورت ماژول را مشخص می کند و حتما باید قبل از اتصال فایل سرآمد توابع LCD در برنامه قرار داده شود . به این صورت که خط اول (define #TFTLCD_DATAPORT_x) پورت اتصالی به دیتا باس LCD را تعریف می کند و باید به جای x ، دو گزینه ی A (پورت A میکروکنترلر) یا B (پورت B میکروکنترلر) قرار داده شود . خط دوم (define #TFTLCD_DATAPORT_OFFSET) فاصله ی شروع دیتا باس LCD به میکروکنترلر را نشان می دهد . برای مثال در صورتی که قصد دارید دیتا باس LCD از پایه ی PORTx.5 شروع به اتصال به میکرو کنترلر کند این تعریف را مساوی ۵ قرار دهید . خط دوم پورت اتصالی میکروکنترلر به کنترل پورت LCD را مشخص می کند که در این صورت باید به جای x گزینه ی A یا B انتخاب شود . خطوط ۴ تا ۸ نیز شماره ی پایه ی اتصالی به هر کدام از خطوط کنترلی LCD (مانند RST ، CS و ...) را مشخص می کند . برای مثال ، شکل زیر که یک نمونه نحوه ی اتصال این LCD به میکروکنترلر را مشخص می کند را مشاهده می فرمایید .



لذا تعریف پایه های ارتباطی ماژول LCD و میکروکنترلر در ابتدای برنامه به صورت زیر می باشد .

```
/* LCD Pin Configuration */
#define TFTLCD_DATAPORT_A
#define TFTLCD_DATAPORT_OFFSET      2
#define TFTLCD_CONTROLPORT_A
#define TFTLCD_RST                  18
#define TFTLCD_RS                   19
#define TFTLCD_CS                   20
#define TFTLCD_WR                   21
#define TFTLCD_RD                   22
```

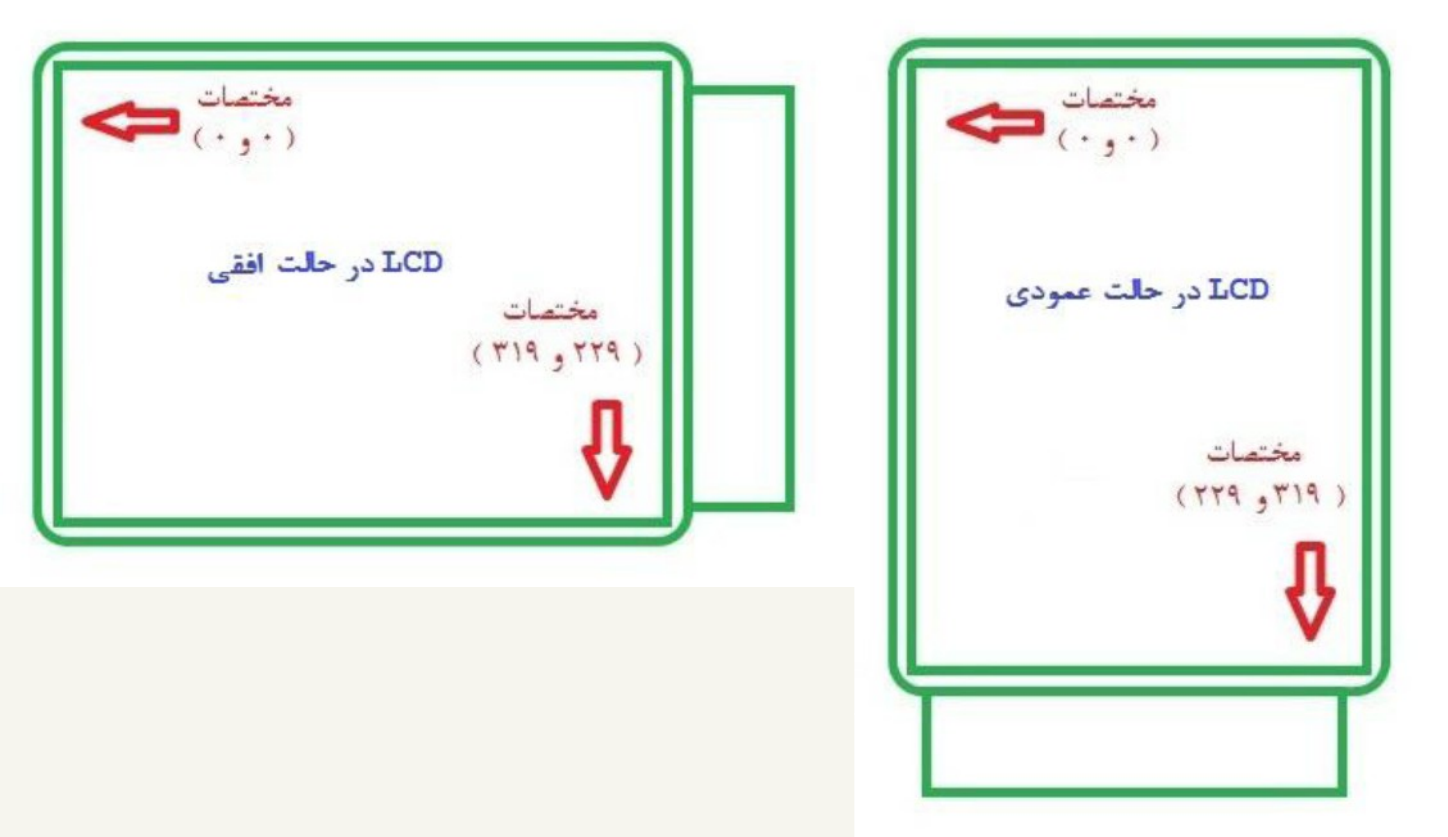
بعد از این مرحله باید نحوه ی قرار گیری و مختصات بند ی صفحه ی LCD را مشخص کنید . در صورتیکه قصد دارید LCD را به طور عمودی استفاده کنید ، تعریف زیر را در برنامه ی خود و قبل از اضافه کردن فایل سرآمد ، قرار دهید .

```
/* Declare LCD Rotation */
#define PORTRAIT
```

در غیر اینصورت دستور زیر را قرار دهید . دستور زیر نحوه ی مختصات بندی صفحه نمایش را به صورت افقی قرار می دهد .

```
/* Declare LCD Rotation */
#define LANDSCAPE
```

شکل زیر مختصات بندی صفحه نمایش در حالت های افقی و عمودی را نشان می دهد .



و در مرحله ی سوم توسط دستور زیر فایل سرآمد توابع راه اندازی LCD را به برنامه ملحق کنید .

```
/* LCD Functions */
#include "tftlcd_functions.h"
```

بعد از تعاریف اولیه ی برنامه باید در ابتدای تابع main و قبل از همه ی توابع دیگر LCD ، ماژول LCD مقدار دهی اولیه بشود . تابع زیر این کار را انجام می دهد و LCD را آماده ی پذیرفتن توابع دیگر می کند .

```
/* LCD Initialization */
tftlcd_init();
```

بعد از اجرای صحیح و کامل مرحله ی آخر شما می توانید توابعی که در زیر به آن ها پرداخته شده است را در برنامه استفاده کنید . به طور کلی توابع موجود به سه دسته تقسیم می شوند . دسته ی اول توابع پایه هستند که جزو ضروری ترین توابع فایل سرآمد می باشد و می توان توسط این سه تابع بقیه ی توابع را بوجود آورد و کار های مختلف پایه ای روی LCD انجام داد . دسته ی دوم توابعی هستند که معمولا برای رسم اشکال هندسی مختلف و اعمال مختلف روی پیکسل ها و صفحه ی نمایش مورد استفاده قرار می گیرند . و دسته ی سوم توابع مربوط به چاپ کاراکتر و ... می باشد . ابتدا به دسته ی اول توابع می پردازیم .

❖ **تابع tftlcd_write_index_register() :**

این تابع درون رجیستر دستورالعمل LCD ، عملیات نوشتن را انجام می دهد (کد دستورالعمل را به LCD ارسال می کند) و الگوی آن به صورت زیر است که آرگومان ارسالی آن همان کد دستوری است که قرار است به LCD ارسال شود .

```
void kslcd_write_index_register(int command)
```

❖ **تابع tftlcd_write_wdr() :**

این تابع همانطور که از نام آن پیداست ، درون رجیستر wdr ماژول LCD می نویسد . داده هایی که درون این رجیستر نوشته می شوند ، به طور معمول به حافظه ی RAM صفحه ی نمایش ارسال می شوند . این تابع نیز یک پارامتر ارسالی دارد که همان داده ای است که قرار است در رجیستر wdr نوشته شود .

```
void kslcd_write_wdr(int data)
```

❖ **تابع tftlcd_read_rdr() :**

این تابع عمل عکس تابع بالا را انجام می دهد . به این صورت که این رجیستر حاوی اطلاعات موجود روی صفحه ی نمایش LCD است که توسط این تابع از LCD خوانده می شود . پارامتر بازگشتی این تابع همان محتویات موجود در ثبات rdr ماژول LCD است .

```
void kslcd_write_wdr(int data)
```

دسته ی دوم توابع که در زیر به آن ها اشاره شده است .

❖ **تابع tftlcd_clear() :**

این تابع کل صفحه نمایش LCD را پاک کرده و مکان نمای مجازی را به مختصات (۰ و ۰) می برد . این تابع هیچ مقدار ارسالی و بازگشتی ندارد .

```
void kslcd_clear(void)
```

❖ **تابع tftlcd_write_pixel()** : الگوی این تابع به صورت زیر است .

```
void kslcd_write_pixel(int x,int y,int color)
```

پارامتر x و y ، مختصات پیکسلی است که قرار است با رنگ color پر شود .

❖ **تابع tftlcd_read_pixel()** : این تابع رنگ پیکسلی به مختصات x و y را بر می گرداند .

```
int kslcd_read_pixel(int x,int y)
```

❖ **تابع tftlcd_draw_line()** : این تابع یک خط راست از مختصات شروع x0 و y0 تا مختصات انتهای x1 و y1 با رنگ مشخص شده توسط پارامتر color رسم می کند . فرم کلی این تابع به صورت زیر است .

```
void kslcd_draw_line(int x0,int y0,int x1,int y1,int color)
```

❖ **تابع tftlcd_draw_rectangle()** : فرم کلی این تابع به صورت زیر است .

```
void kslcd_draw_rectangle(int x0,int y0,int x1,int y1,char fill,int color)
```

این تابع وظیفه ی رسم یک چهارگوشه را دارد که می تواند توسط پارامتر fill ، تو خالی یا توپر باشد . در صورتیکه fill = 0 باشد ، چهار گوشه توخالی و اگر fill = 1 باشد چهار گوشه توپر خواهد شد . پارامتر های x0 و y0 ، مختصات شروع رسم چهار گوشه و x1 و y1 ، مختصات انتهای این چهارگوشه را تعیین می کند . پارامتر color نیز تعیین کننده ی رنگ این چهارگوشه است .

❖ **تابع tftlcd_draw_circle() :** این تابع نیز به مختصات مرکز x و y و شعاع r و با رنگ color ، دایره ای رسم می کند . در صورتیکه آرگومان fill برابر صفر باشد دایره توخالی و اگر fill = 1 باشد دایره توپر خواهد بود .

```
void kslcd_draw_circle(int x0,int y0,int radius,char fill,int color)
```

❖ **تابع tftlcd_write_pic() :** این تابع یک عکسی که در حافظه ی flash میکروکنترلر ذخیره شده است را در مختصات شروع x و y نمایش می دهد . الگوی این تابع به صورت زیر است و

```
void kslcd_write_pic(int x,int y,const short *pointer)
```

آرگومان سوم این تابع نام ثابتی است که به صورت آرایه در حافظه ی flash میکروکنترلر ذخیره شده است . باید به این نکته توجه داشته باشید که آرایه ی اول و دوم این ثابت را به ترتیب به طول و عرض همان عکس اضافه کنید . به صورت زیر

```
const short picture[] =
{
    length,width,
    0xBA62,0x9200,0xA200,0x9A00,0xA220,0xB282,0xC241,0xC240,
    0xA220,0xA220,0x9A20,0xA220,0xB261,0x9220,0x9200,0x9A20,
    0xA220,0x9A20,0xAA40,0x9A20,0xAA60,0xA220,0xAA40,0xA240,
    .
    .
    .
    .
    .
    0x9A20,0x9A20,0x9A00,0xAA40,0xAA20,0xB240,0xAA60,0xAA00,
    0x9A00,0x9A00,0xAA42,0xAA62,0xAA01,0x89E0,0x99E0,0x9A20,
    0xAA21,0xA222,0x9A01,0x9A21,0x89E0,0x89C1,0x81C1,0x79C1
    };
```

که شما باید به جای length و width ، به ترتیب طول و عرض عکس را قرار دهید . دستور زیر نیز این عکس را در مختصات (۰ و ۰) قرار می دهد .

```
kslcd_write_pic(0,0,picture);
```

و در اینجا هم آخرین دسته از توابع را بررسی خواهیم کرد . قابل توجه اینکه یک فونت ۸ در ۱۶ نیز برای کار با توابع چاپ رشته در برنامه تعریف شده است .

❖ **تابع tftlcd_gotoxy() :** این تابع مکان نمای مجازی متن را به مختصات x و y می برد . باید توجه داشته باشید که حداکثر مقدار x ، ۴۰ و y برابر ۱۵ در حالت افقی و در حالت عمودی به ترتیب ۱۵ برای x و ۴۰ برای y می باشد . الگوی این تابع به صورت زیر است .

```
void kslcd_gotoxy(int x,int y)
```

❖ **تابع tftlcd_putchar() :** فرم کلی این تابع به صورت زیر است .

```
void kslcd_putchar(char character,int forground_color,int background_color,int transparent_mode)
```

آرگومان اول کاراکتری است که در مختصات فعلی مکان نما چاپ می شود . آرگومان دوم رنگ کاراکتر و آرگومان سوم رنگ پس زمینه ی کاراکتر را تعیین می کند . آرگومان چهارم نیز یک امکان جدید را برای کاربر قرار می دهد . در صورتیکه این پارامتر مقدار صفر داشته باشد این ویژگی غیر فعال می شود . اما در صورتیکه مقدار ۱ داشته باشد ، رنگ پس زمینه غیر فعال می شود و به جای آن رنگ فعلی پیکسل در نظرگرفته می شود . شکل زیر این ویژگی را بهتر نمایش می دهد .

■ **توابع tftlcd_puts() و tftlcd_putsf() :** این دو تابع هر دو وظیفه ی نمایش یک رشته را روی صفحه ی نمایش دارند با این تفاوت که تابع tftlcd_puts() رشته ی ارسالی ذخیره شده در SRAM میکروکنترلر را می پذیرد و تابع tftlcd_putsf() رشته ای که در حافظه ی flash ذخیره شده است را نمایش می دهد . آرگومان های دوم تا چهارم نیز وظیفه ی مشابه تابع توضیح داده شده در قسمت قبل را دارند .

```
void kslcd_putsf(const char *string,int forground_color,int background_color,int transparent_mode)
```

```
void kslcd_puts(char *string,int forground_color,int background_color,int transparent_mode)
```

مثال :

برنامه ی زیر هم به صورت نمایشی نحوه ی استفاده از توابع توضیح داده شده را نشان می دهد

نمونه مثال راه اندازی LcdN96

```
/* AT91SAMT7X256 Register definitions */
#include < AT91SAM7X256.H>
/* Delay Functions */
#define F_CPU 72000000
#include <delay.h>
/* Declare LCD Rotation */
#define PORTRAIT
/* LCD Pin Configuration */
#define TFTLCD_DATAPORT_A
#define TFTLCD_DATAPORT_OFFSET 0
#define TFTLCD_CONTROLPORT_A
#define TFTLCD_RST 16
#define TFTLCD_RS 17
#define TFTLCD_CS 18
#define TFTLCD_WR 19
#define TFTLCD_RD 20
/* LCD Functions */
#include "picture.h"
#include "tftlcd_functions.c"
int main(void)
{
delay_ms(100);
/* LCD Initialization */
tftlcd_init();
```

```
#ifdef LANDSCAPE
tftlcd_write_pic(0,0,picture);
delay_ms(5000);
tftlcd_draw_line(5,5,180,120,RED);
tftlcd_draw_line(10,5,185,120,GREEN);
tftlcd_draw_line(15,5,190,120,BLUE);
tftlcd_draw_rectangle(160,20,250,120,0,YELLOW);
tftlcd_draw_rectangle(165,25,245,115,0,PURPLE);
tftlcd_draw_rectangle(170,30,240,110,1,BLUE);
tftlcd_draw_circle(70,150,50,0,GREEN);
tftlcd_draw_circle(90,150,50,1,PURPLE);
tftlcd_gotoxy(20,12);
tftlcd_putsf( "...:::HELLO:::...",RED,WHITE,0);
tftlcd_gotoxy(2,9);
tftlcd_putsf( "without Transparent",GREEN,WHITE,0);
tftlcd_gotoxy(2,10);
tftlcd_putsf( "Transparent Mode",BLUE,WHITE,1);
delay_ms(10000);
while(1)
{
tftlcd_draw_rectangle(0,0,319,239,1,RED);
delay_ms(1000);
tftlcd_draw_rectangle(0,0,319,239,1,GREEN);
delay_ms(1000);
tftlcd_draw_rectangle(0,0,319,239,1,BLUE);
delay_ms(1000);
};
#endif

#ifdef PORTRAIT
tftlcd_write_pic(0,0,picture);
delay_ms(50000);
tftlcd_draw_line(5,5,180,120,RED);
tftlcd_draw_line(10,5,185,120,GREEN);
tftlcd_draw_line(15,5,190,120,BLUE);
tftlcd_draw_rectangle(160,20,200,120,0,YELLOW);
tftlcd_draw_rectangle(165,25,205,115,0,PURPLE);
tftlcd_draw_rectangle(170,30,205,110,1,BLUE);
tftlcd_draw_circle(70,150,50,0,GREEN);
tftlcd_draw_circle(90,150,50,1,PURPLE);
tftlcd_gotoxy(3,3);
tftlcd_putsf( "...:::HELLO:::...",RED,WHITE,0);
tftlcd_gotoxy(3,4);
tftlcd_putsf( "without Transparent",GREEN,WHITE,0);
tftlcd_gotoxy(3,6);
tftlcd_putsf( "Transparent Mode",BLUE,WHITE,1);
delay_ms(10000);
while(1)
{
tftlcd_draw_rectangle(0,0,239,319,1,RED);
delay_ms(1000);
tftlcd_draw_rectangle(0,0,239,319,1,GREEN);
delay_ms(1000);
tftlcd_draw_rectangle(0,0,239,319,1,BLUE);
delay_ms(1000);
};
#endif
}
```

This document was created with Win2PDF available at http://www.win2pdf.com.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.
This page will not be added after purchasing Win2PDF.