www.microdesigner.ir

پروژه تابلو روان سه رنگ

تعریف پروژه

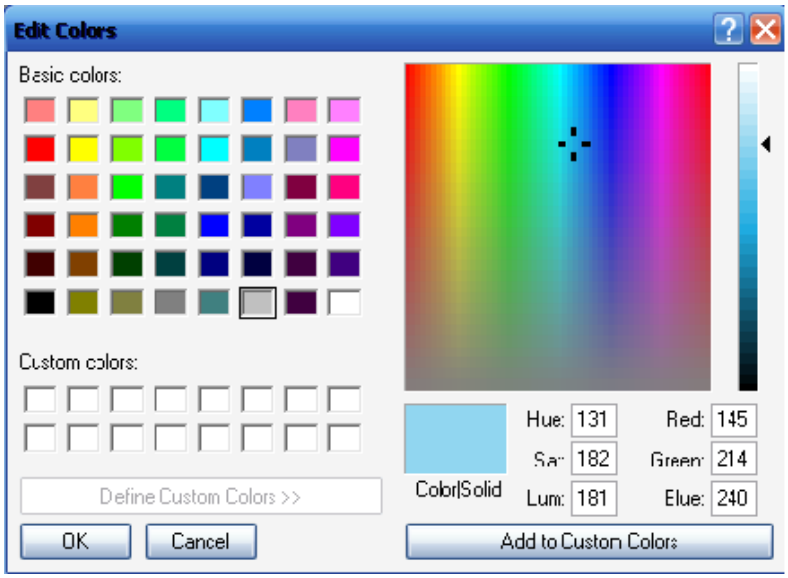
تابلو روان رنگی یک تابلو روان نیست بلکه یک تابلو گرافیکی می باشد و قابلیت نمایش هر تصویری را دارد(در ابعاد بزرگ)تابلو روان رنگی از مجموع led هایی که منظم در کنار هم قرار گرفته اند تشکیل شده است که در این تابلو روان ازسه led در کنار هم استفاده شده است این led ها با خاموش و روشن شدن شان رنگ های مختلفی را تولید می کنند اما اگر بخواهیم تمام یا مقداری از رنگ های طبیعت را درست کنیم باید از سه رنگ مادر استفاده کنیم که این سه رنگ عبارت اند از : سبز و آبی وقرمز،که در این پروژه از این سه رنگ که در کنار یکدیگر قرار گرفته اند استفاده شده است ،که معمولا از این پروژه برای تبلیغات استفاده می شود.

علت استفاده از rgb billboard

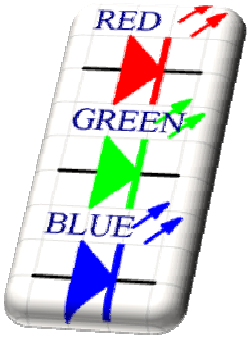
معمولا از این پروژه در فضاهای بیرون استفاده می شود و از lcd ها استفاده نمی شود زیرا این lcd ها در بیرون و یا در زیر نورمستقیم آفتاب هیچ تصویری دیده نمی شود اما این پروژه چون هر کدام از پیکسل ها ی آن led می باشد و led نیز از خود نور می دهد در زیر نور آفتاب نیز تصویر نشان می دهد .

چگونه با ید رنگ تولید کنیم؟

اگر در برنامه ی piant در قسمت رنگ های آن نگاه کنید و با تغییر مقدار هر یک از رنگ های قرمز وآبی و سبز می توانید رنگ های مختلفی درست کنید



ما در این پروژه از led های سبز وآبی و قرمز استفاده کرده ایم که برای کم یا زیاد کردن مقدارنور led ها می توانیم از دو روش استفاده کنیم ۱-تغییر ولتاژ در سر led ها ۲- تغییر زمان روشن بودن led ها می باشد .



۱ - برای تغییر ولتاژ دو سر ledها ما باید از آی سی ها ی DAC استفاده کنیم .

۲ – برای تغییر زمان روشن بودن LED ها ما باید از PWM استفاده کنیم

بعضی از کاربرد های آن

یکی از کاربرد ها ی معروف آن که تقریبا همه دیده اند تبلیغات در دور زمین های فوتبال می باشد که آها قابلیت ها ی خاصی دارند از جمله : آنها زدضربه هستند وزد آب هستند و...

ودر میدان ها ی بزرگ شهر و بعضی از شرکت ها می باشد .

1_ در داخل فروشگاهها و مغازه ها برای تبلیغ و اطلاع رسانی

2– سردر فروشگاهها

3– داخل ویترین

4– نصب در خیابانها ، چهارراهها و اتوبانها

5– نصب در کارخانه و کارگاه برای اطلاع رسانی به کارگران

6– در هر مکان بنا به خاصیت آن مکان : مساجد ، سینماها ، مراکز اداری ، موزه ها ، مراکز نظامی و...

ویژگی عمومی تابلوهای روان

*امکان ارسال اطلاعات با سیم رابط استاندارد تا ۵۰۰ متر(rs۴۸۵)

*تک رنگ و سه رنگ (قرمز ، سبز ، زرد)ورنگی

*در مدلهای سایه دید (مخصوص فضاهای سر پوشیده و مسقف) و روزدید (مخصوص فضاهای روباز با قابلیت رویت در نور مستقیم خورشید و مقاوم در برابر باران و شرایط جوی)

*امکان اضافه کردن دماسنج ، فشار سنج و رطوبت سنج

*ارسال اطلاعات از کامپیوتر به تابلو در کمتر از سه ثانیه

*قابلیت نمایش ساعت و تاریخ

*ارتباط از طریق پورت سریال کامپیوتر

*قابلیت نمایش متن , گرافیک و انیمیشن

*عدم نیاز به اتصال دائم به کامپیوتر

*قابلیت ذخیره حجم بالایی از اطلاعات در تابلو

انوع تابلو روان

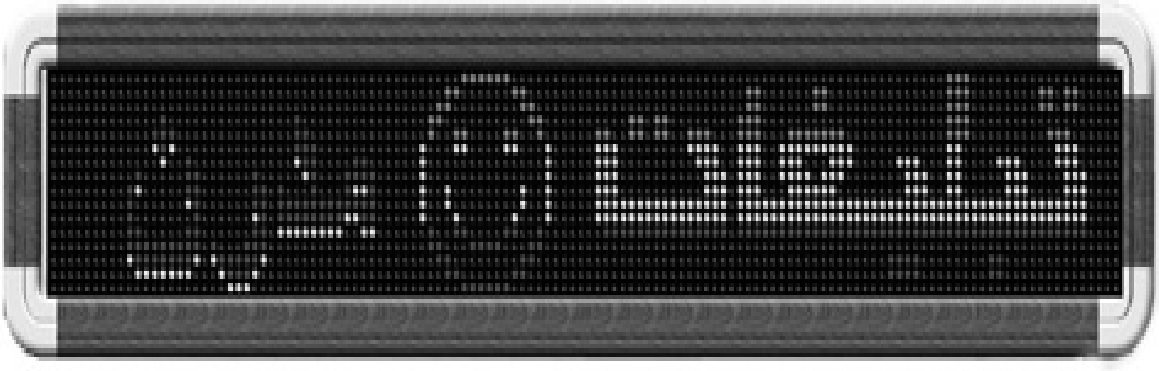
تابلوهای خطی– تک رنگ

€

این دسته از تابلوها که اکثرا از دیودهای قرمز ساخته می شوند قابلیت نصب در سر درب مغازه ها ، فروشگاهها ، رستورانها ، کتابخانه ، ها واحدهای روابط عمومی اداره ها و شرکتهای خصوصی و دولتی را دارا می باشد . و یکی از بهترین ابزارهای اطلاع رسانی و پیام رسانی به مردم، مشتریان و یا مخاطبان می باشند .

تابلوهای خطی – سه رنگ

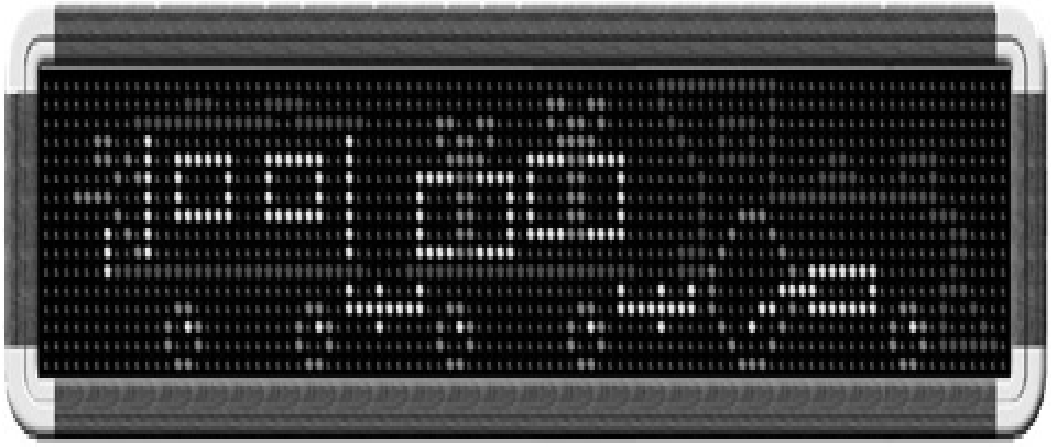
Error!



این دسته از تابلوها معمولا از سه رنگ با رنگهای سبز , قرمز ونارنجی یا زرد طراحی و ساخته می شوند . سر درب نمایشگاهها , سالنهای رستورانها , دانشگاهها و اتاق سمینارها و مواد متعدد دیگری که طیف گسترده ای از صنوف را در بر میگیرد , هر کدام به نحوی میتواند از امکانات بی نظیر تابلوهای فوق استفاده نمایند .

دارا بودن سه رنگ نمایشی نسبت به تابلوهای تک رنگ , امکانات گسترده تری را جهت طراحی افکت های ویژه به تابلو خواهد داد , علاوه بر اینکه خود سه رنگ نیز نوشته ها و حروف را دارای جذابیت بیشتری از جهت دید مشتری می کند .

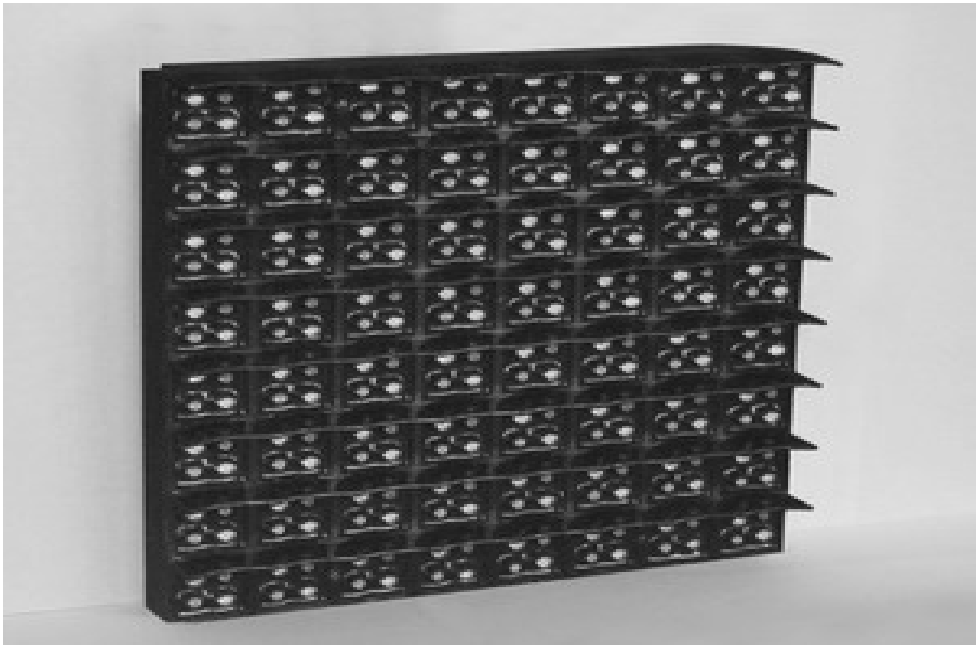
تابلوهای گرافیکی – خطی



در واقع شکل بزرگتر ارائه می گردد ترکیبی از تابلوهای خطی و گرافیکی است که به

علت تعداد سطر بالاتر امکان نمایش گرافیکهای زیباتر و بزرگتری را در خود دارد. این مدل از تابلوها دارای ۳۲ ردیف در عرض و ...در ستون میباشد. ابعاد ، تابلوها در عرض از ۸۰ سانت تا ۳ متر و در طول از ۳ متر تا۲۰مترمی تواند طراحی و ساخته شود.
تابلوهای فوق با قابلیت تک رنگ و سه رنگ, تک پیکسل و یا چند پیکسلی کلاستری توانایی طراحی و ساخت را دارا می باشند .

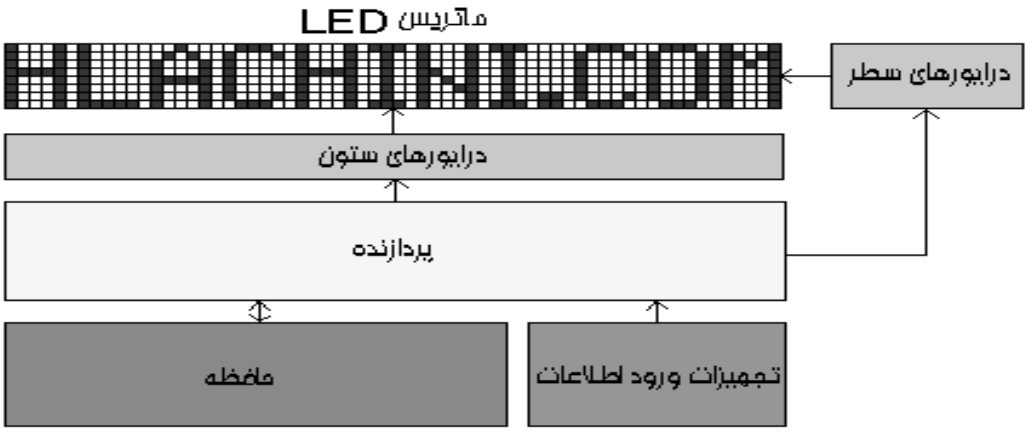
تابلوهای کلاستری



کلاستر نام استوانه مجوف شکلی میباشد که درون آن با ۴ , ۵ , ۷ , ۹, ۱۳,و یا بیشتر LEDیک رنگ و یا ترکیبی از رنگهای مختلف پر میشود. لبه بالای کلاستر جلوی تابش مستقیم نور خورشید را گرفته و باعث می شود که تابلو در روز نور کافی داشته باشد. تابلوهای کلاستری دارای ابعاد بزرگتری میباشند و خاص فضاهای بیرونی

است (out door) امکانات وتوانمندیهای تابلوهای کلاستری با تابلوهای خطی تفاوت خاصی ندارد.

بلوک دیاگرام تابلو روان ساده



همانطور که در تصویر مشاهده می کنید، این تابلوها از بلوکهای :

- **ماتریس LED**
- **درایورهای سطر و ستون**
- **پردازنده**
- **تجهیزات ورود اطلاعات**
- **حافظه**

تشکیل شده‌اند.

در واقع یک تابلوی نمایشگر دیجیتالی، متن مورد نظر خود را از طریق تجهیزات ورودی همچون کیبورد و یا پورت سریال دریافت میکند. و این اطلاعات را در اختیار پردازنده قرار میدهد. سپس پردازنده پس از آنالیز اطلاعات آن را در حافظه تابلو ذخیره نموده. علاوه بر آن حافظه موجود در تابلو میتواند کدهای برنامه را در خود نگهداری نماید. از طرفی پردازنده با توجه به اطلاعات ذخیره شده، سیگنالهای لازم را جهت

نمایش تولید کرده و در اختیار درایورها قرار میدهد. با توجه به اینکه نحوه چیدمان LED ها در نمایشگر بنا به دلایلی که بعدا توضیح داده خواهد شد به صورت ماتریسی می باشد، لذا دو دسته درایور برای راه اندازی ماتریس نیاز است که شامل داریورهای سطر و داریورهای ستون میباشند. این درایورها با توجه به فرامین دریافتی از سوی پردازنده، با روشن و خاموش نگاه داشتن LED های موجود در ماتریس، باعث به نمایش در آمدن مطالب (اعم از متن و یا تصویر) بر روی ماتریس خواهند شد.

اثر فلیکر

اثر نور در چشم انسان برای مدت کوتاهی باقی می‌ماند. این خاصیت را اثر پس ماند نور می‌نامند. برمبنای همین خاصیت است که در سینما و تلویزیون احساس پیوستگی تصویر بوجود می‌آید .

چنانچه تصاویری که از یک حرکت مثلا راه رفتن انسان عکس برداری شود و سپس با سرعت ۱۶ بار در ثانیه به نمایش درآید، چشم انسان منقطع بودن تصاویر را احساس نکرده و تصاویر را بطور پیوسته حس می‌کند. بر مبنای این خاصیت بود که صنعت سینما بوجود آمد .

ترتیب کار به این صورت است که توسط یک دوربین فیلمبرداری مخصوص که قادر است در هر ثانیه ۱۶ تصویر از یک صحنه عکس برداری نماید، تصاویر تهیه شده سپس با همان سرعت به نمایش در می‌آیند. البته به علت اینکه با ۱۶ تصویر در ثانیه حرکات نرم و طبیعی نداریم، فرکانس مزبور بعدا به ۲۴ تصویر در ثانیه افزایش داده شد. در این فرکانس

برای بیش از ۹۰ درصد حرکات، پیوستگی طبیعی بوجود می‌آید. به همین علت به فرکانس مزبور حد پیوستگی گفته میشود. مشکل دیگر مسئله چشمک زدن تصویر است .

در فرکانس ۲۴ تصویر در ثانیه اگر چه مسئله پیوستگی تصاویر حل میشود اما تصاویر چشمک می‌زنند و این بخاطر این است که چشم اگرچه در این فرکانس، تصاویر را پیوسته می‌بیند و حرکات را طبیعی احساس میکند اما خاموش شدن صحنه در حین تعویض یک تصویر به تصویر بعدی بوجود می‌آید را بصورت چشمک زدن تصویر احساس میکند. این پدیده بخصوص برای تصاویری با نور بیشتر محسوس‌تر است .

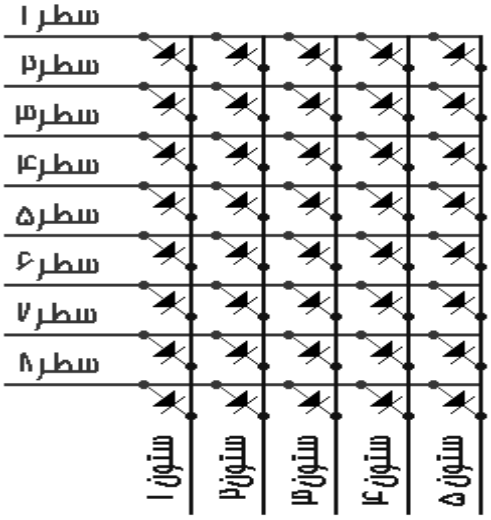
برای رفع این مشکل باید حداقل ۴۸ تصویر در ثانیه به نمایش درآید تا اثر چشمک زدن از بین رود. در سینما چون نمایش ۴۸ تصویر در ثانیه، اشکالات عملی بوجود می‌آورد، مسئله را به طریق دیگری حل نموده‌اند. به این ترتیب که سرعت حرکت نوار فیلم از مقابل لامپ پروژکتور همان ۲۴ تصویر در ثانیه است منتهی به کمک یک دیافراگم گردان به هنگام تعویض یک فریم به فریم بعدی و همچنین در زمان نمایش

فریم و درست در وسط زمان مزبور نور لامپ پروژکتور به فیلم قطع میشود. با اینکار هر فریم دو بار روشن و خاموش میشود .

با این تدبیر که هر تصویر دو بار روشن میشود و سرعت حرکت نوار ۲۴ تصویر در هر ثانیه است، از نظر چشم ۴۸ تصویر در ثانیه احساس میشود و مشکل چشمک زدن از بین میرود. در تابلوهای روان هم مسائل پیوستگی تصاویر و همچنین چشمک زدن، عوامل تعیین کننده سیستم جاروب و زمانهای مربوطه هستند .

جاروب ساده

جهت تشکیل تصویر بر روی پانل تابلو، نیاز به روشن و خاموش نگه داشتن LED های موجود بر روی تابلو متناسب با تصویر مورد نظر است. بنابراین نیاز به کنترل تک تک LED های موجود در تابلو میباشد. از طرفی هر LED دارای دو پایه است (با فرض تک رنگ بودن) و در صورتی که ما یک پانل LED با ماتریس ۱۶*۱۶داشته باشیم،حدودا دویست پایه و یا دویست سیم جهت کنترل داریم. مسلما استفاده از این تعداد سیم مقرون به صرفه نخواهد و باعث پیچیدگی مدار خواهد شد. جهت بر طرف کردن مشکل فوق می‌توان پایه های یکسان درLED ها را به صورت سطری و ستونی به یکدیگر متصل نمود. به تصویر زیر دقت کنید :



همانطور که در تصویر مشاهده نمودید، در این آرایش آند تمامی LED های موجود در یک سطر یکسان به هم متصل شدند، همچنین کاتد LED های موجود در یک ستون نیز به هم اتصال داده شده‌اند.

حال ببینیم نحوه عملکرد این روش چگونه است. شما در این حالت جهت روشن کردن هر LED کافی است که سطری که آن LED در آنجا قرار دارد را به سطح ولتاژ مثبت اتصال داده و سپس ستون مربوط به همان LED را به زمین مدار وصل کنید.

با این روش ما توانستیم از تعداد سیمهای مورد نیاز جهت کنترل LED ها بکاهیم ولی در مقابل امکان کنترل همزمان تمامی سطرها را از دست دادیم و در هر لحظه فقط و فقط میتوان LED های موجود در یک سطر و یا یک ستون را کنترل نمود.

در ادامه همین بحث خواهید دید که جهت نمایش نیازی هم به تمامی LED ها نبوده و میتوان توسط جاروب نمودن سطرها و یا ستون‌ها نیز به نمایش تصویر در تابلو روان پرداخت.

به هر حال در صورت عدم استفاده از روش فوق شما مدار پیچیده‌ای خواهید داشت، مثلا برای کنترل LED ها موجود در تصویر روبرو شما حداقل باید از طریق ۵۱۲

سیم مدار را کنترل میکردید. در حالی که با استفاده از روش ماتریسی شما فقط به ۳۲ سیم نیاز دارید. فقط در این حالت برنامه شما کمی پیچیده خواهد شد. که البته به نظر من شما یک بار برنامه می نویسید و همیشه از آن استفاده می کنید ولی سخت افزار را باید تا همیشه مونتاژ نموده و هزینه آن را پرداخت کنید .

جاروب یک در میان

حال فرض کنید شما یک تابلو با ۱۶ ستون می‌خواهید طراحی کنید و از جاروب ستونی هم استفاده می‌کنید حال زمان نمایش هر فریم برابر با ۱۶ میلی ثانیه خواهد بود و در هر فریم ۱۶ ستون جهت جاروب داریم پس زمان روشن بودن هر سستون برابر با ۱ میلی ثانیه خواهد بود.

خوب شما مدار را طراحی کرده و می سازید اما در پایان متوجه می شوید که نور LED ها بسیار کم تر از حالت معمولی است و حسابی متعجب خواهید شد که چرا با وجود استفاده از LED های مرغوب نور تابلو روان تا این حد کم است

نکته اینجاست که شما هر LED را فقط به مدت ۱ میلی ثانیه روشن نگاه می دارید و سپس به مدت ۱۵ برابر این مدت خاموش نگاه می دارید(به خاطر جاروب ۱۵ ستون دیگر) یعنی ۱ میلی ثانیه روشن و ۱۵ میلی ثانیه خاموش است. ودر واقع اثر نور LED در چشم به میزان قابل توجه‌ای کاهش می‌یابد.

جهت کم کردن این اثر و افزایش نور تابلو روان چند کار را می‌توان انجام داد :

۱- افزایش ولتاژ اعمالی به LED ها که معمولا این کار خطر سوختن LED ها در اثر هنگ کردن تابلو افزایش داده و همچنین از عمر مفید آن نیز خواهد کاست.

۲- تقسیم تابلو به سگمنت های جداگانه مثلا برای مثال فوق تقسیم تابلو به ۲ سگمنت ۸ ستونی. این روش مناسبی است ولی به پیچدگی نرم‌افزار و سخت‌افزار خواهد افزود و در تابلو های کوچک توصیه نمی‌شود.

۳- جاروب یک در میان، این روش راه حل مناسبی برای بر طرف نمودن مشکل فوق است. در عین حال که به پیچیدگی مدار منجر نخواهد شد و همچنین با تغییر ساده‌ای در الگوریتم برنامه میتوان از آن بهره برد.

ترتیب کار به این صورت است که ما هر فریم کامل را که در مثال فوق برابر با ۱۶ سطر است به دو نیم فریم تقسیم میکنیم. نیم فریم اول شامل ستون های فرد (۱،۳،۵،۷،۹،۱۱،۱۳،۱۵) و نیم فریم دوم شامل ستون های زوج (۲،۴،۶،۸،۱۰،۱۲،۱۴،۱۶) است. حال در هر جاروب فقط یکی از دو نیم فریم زوج و یا فرد را جاروب می کنیم .

بدین ترتیب که یکبار نیم فریم فرد و بار بعد نیم فریم زوج و دوباره نیم فریم فرد و ... در این حالت چون در هر بار فقط ۸ سطر جاروب میشوند لذا زمان روشن بودن هر LED بیشتر خواهد شد .

طراحی مدار slave

برای طراحی یک تابلو روان (یا گرافیکی) باید اول بدانیم یک تابلو روان از چه قسمت های تشکیل شده است که در این گزارش کار مختصرا درباره ی این قسمت ها توضیح داده ایم وبعد این قسمتها از چه قطعاتی تشکیل شده اند را توضیح خواهیم داد.که در نهایت با شناخت قسمت های مختلف و قطعات می توان یک شماتیک برای تابلو روان طراحی کرد.

معمولا تابلو روان ها وتابلو های رنگی از سه قسمت اصلی تشکیل می شوند که عبارت اند از:

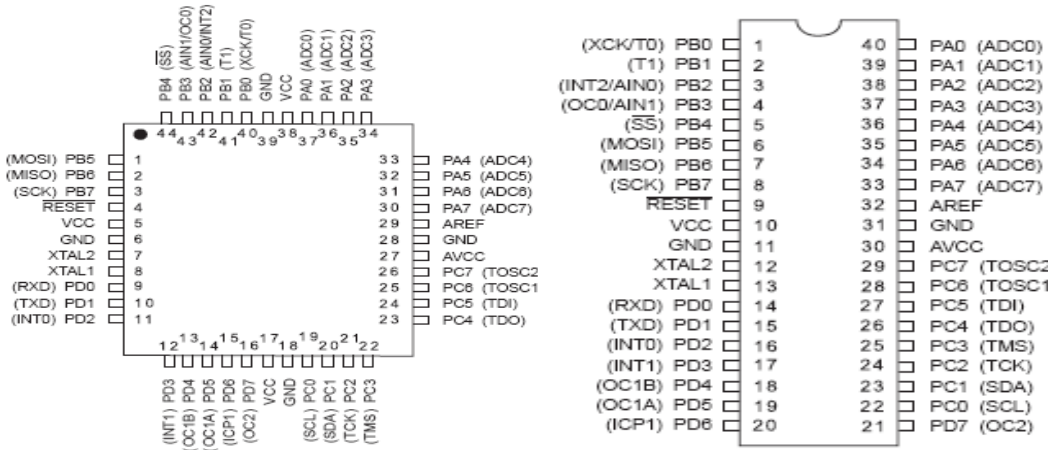
۱. پردازشگر

۲. درایو

۳.صفحه نمایش

۱.پردازشگر:

به قسمتی از مدار گفته می شود که اطلاعات را از کامپیوتر یا ... می گیرد ودر خود اطلاعات را نسبت به مدار خروجی پردازش می کند وتغییرات لازم برای نمایش روی صفحه نمایش می دهد که ما در این پروژه از آی سی میکرو کنترولر ATmega۳۲ استفاده کرده ایمکه در بازارایران هر دو نوع معمولی و SMD موجود می باشد که بعداً توضیحات کامل آن را شرح می کنیم .

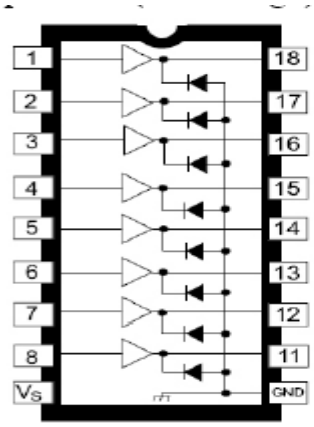


ATmega۳۲

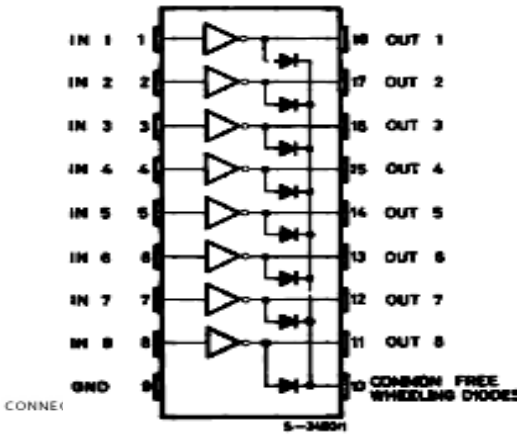
۲.درایور ها:

به قسمتی از مدار تابلو روان گفته می شود که به سطر و ستون های مدار متصل شده است که وظیفه اصلی این قسمت تامین جریان قسمت صفحه نمایش می باشد . اگر ما از درایور ها استفاده نکنیم صفحه نمایش نه نور کافی را دارد ونه نور یکسان در کل صفحه نمایش که ما در این پروژه از آی سی های UDN۲۹۸۱ که آی سی های جریان دهنده هستند که باید به قسمت مثبت led ها متصل شود و دوم از آی سی ULN۲۸۰۱ که آی سی های جریان گیرنده می باشند که باید به قسمت منفی ledها متصل شوند استفاده کرده ایم که بعداً توضیحات کامل آن را شرح می کنیم .

نمای ظاهری



UDN۲۹۸۱

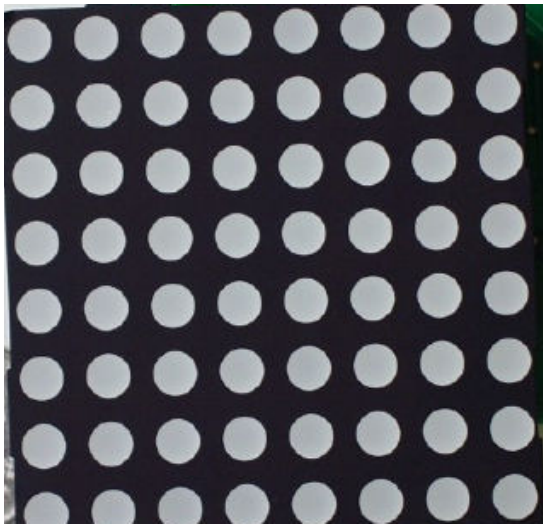


ULN۲۸۰۱

۳.صفحه نمایش :

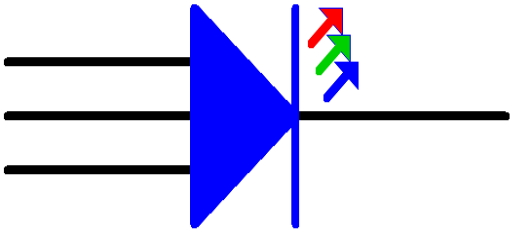
یکی از مهمترین قسمت مدار که در مقابل دید مردم می با شد که باید با نور وکیفیت تصویر جلب توجه کند که باید در این قسمت در انتخاب led های مناسب (پر نور و نور یکدست) دقت کنیم که در بازار انواع مختلفی موجود می باشد .که در شکل زیر عکس یک ماتریس آماده ۸*۸ که led های آن rgb نیز می

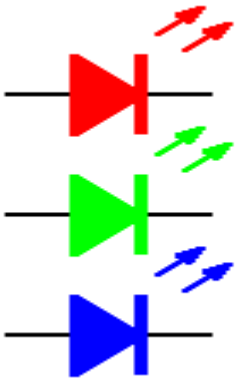
باشند نشان داده ایم.



ماتریس۸*۸ rgb

در این پروژه ما اول می خواستیم از led های rgb استفاده کنیم که بعد به دلیل این که این ledها هم نور کمتری نسبت به ۳عدد led قرمز و سبز و آبی می دهند وهمچنین قیمت ۱عدد rgb led از ۳عدد led قرمز و سبز و آبی گران تر می باشد ما تصمیم گرفتیم که از ۳ عدد led قرمز سبز آبی استفاده کنیم .





نمای شماتیک led rgb و led های قرمز سبز آبی که به دلیل پرینت سیاه وسفید رنگ آنها معلوم نمی باشد

توضیحات کلی قطعات برد master و slave

برد MASTER

۱- ATMEGA ۳۲

۲- MAX ۲۳۲

۳- LM۲۵۷۶ (منبع تغذیه سوئیچینگ)

برد SLAVE

۱- ATMEGA ۱۶

۲- ۴۰۲۸

۳- UDN۲۹۸۱

۴- ULN۲۸۰۳A

۵- مقاومت های کنترل جریان

قطعات برد MASTER

۱- ATMEGA ۳۲

میکروهای AVR دارای انعطاف پذیری غیر قابل مقایسه و بی همتایی هستند. آن ها قادر به ترکیب هر نوع کدی با یک معماری کارآمد از طریق زبان های C و

ASSEMBLY هستند و قادرند از طریق این برنامه ها تمام پارامتر های ممکن در یک سیکل یا چرخه ی ماشین را با دقت بسیار بالا هماهنگ کنند. میکرو AVR دارای معماری است که می تواند در تمام جهات استفاده شود. این میکرو دارای معماریی است که به ما کارایی ۱۶ بیتی ارائه می دهد که البته قیمتش به اندازه یک ۸ بیتی تمام می شود.

بهره های کلیدی AVR

دارای بهترین MCU برای حافظه ی فلش (MCU: Master Control Unit)

دارای سیستمی با بهترین هماهنگی

دارای بالاترین کارایی و اجرا در CPU (یک دستورالعمل در هر سیکل کلاک)

دارای کد هایی با کوچکترین سایز

دارای حافظه ی خود برنامه ریز

دارای سخت افزار ضرب کننده روی خود

دارای بهترین ابزار ها برای پیشرفت و ترقی

دارای حالات زیادی برای ترفیع دادن یا UPGRADE

میکروکنترلر AVR برمبنای معماری RISC (کاهش مجموعه دستورالعمل های کامپیوتر) پایه گذاری شده و مجموعه ای از دستورالعمل ها را که با ۳۲ ثبات کار می کنند ترکیب می کند.به کار گرفتن حافظه از نوع فلش که AVR ها به طور یکسان از آن بهره می برند از جمله مزایای آن ها است. یک میکرو AVR می تواند با استفاده از یک منبع تغذیه ی ۲.۷ تا ۵.۵ ولتی از طریق شش پین ساده در عرض چند ثانیه برنامه ریزی شود. میکروهای AVR در هر جا که باشند با ۱.۸ ولت تا ۵.۵ ولت تغذیه می شوند. البته با انواع توان پایین که موجودند.

خانواده های محصولات AVR

Tiny AVR

میکروکنترلری با اهداف کلی و با بیش از ۴ کیلو بایت حافظه فلش و ۱۲۸ بایت حافظه استاتیک و قابل برنامه ریزی است.(منظور از حافظه استاتیک SRAM و حافظه ی قابل برنامه ریزی EEPROM است.)

Mega AVR

این نوع میکرو ها قابلیت خود برنامه ریزی دارند و می توان آن ها را بدون استفاده از مدارات اضافه برنامه ریزی کرد. همچنین بیش از K ۲۵۶ بایت حافظه ی فلش و ۴K بایت حافظه ی استاتیک و قابل برنامه ریزی دارند.

LCD AVR

این نوع میکرو دارای درایور برای نمایشگر LCD با قابلیت کنترل اتوماتیک تباین و مقایسه ی تصویر می باشد. باعث تمدید عمر باتری می شود و در حالت فعال دارای توان مصرفی پایینی است.

توان مصرفی پایین:

توان مصرفی پایین آن ها برای استفاده بهینه از باتری و همچنین کاربرد میکرو در وسایل سیار و سفری طراحی شده که میکرو های جدید AVR با توان مصرفی کم از شش روش اضافی در مقدار توان مصرفی برای انجام عملیات بهره می برند. این میکرو ها تا مقدار ۱.۸ ولت قابل تغذیه هستند که این امر باعث طولانی تر شدن عمر باتری می شود. در میکرو هایی با توان پایین عملیات شبیه حالت Standby هستند. یعنی میکرو می تواند تمام اعمال داخلی و جنبی را متوقف کند و کریستال خارجی را به همان وضعیت شش کلاک در هر چرخه رها کند.

مدل های tiny AVR:

میکروهای مدل tiny توانـایی هـای عظیمـی دارنـد به خاطر کوچک بودن و داشتن MCU بسیار پر قدرت به اینگونه میکروها نیـاز فراوانـی ەسـت آنهـا بـه هـیچ منطق خارجی نیاز نداشته و به ەمراه یک مجتمع مبدل آنـالوگ بـه دیجیتـال و یـک حافظـه قابـل برنامـه ریزی EEPROM قابلیتهای خود را ثابت می کنند.

نکات کلیدی و سودمند مدلTiny آنهـا به منظـور انجـام یـک عملیات سـاده بهینـه سـازی شـده و در سـاخت وسـایلی کـه به میکرو های کوچک احتیاج است کاربرد فراوان دارند. کارایی عظیم آنها برای ارزش و بهای وسایل موثر است.

مدل های Mega AVR

اگر شما به میکرویی احتیاج دارید که دارای سرعت و کارایی بالا باشد و توانایی اجـرای حجـم زیـادی از کد برنامه را داشته و بتواند داده ەای زیادی را سروسامان دەد باید از AVRهـای مـدل Mega اسـتفاده کنید آنها به ازای ەر یک مگا ەرتز سرعت ، توانایی اجرای یک میلیـون دسـتورالعمل در هـر یـک ثانیـه رادارند ەمچنین قابل برنامه ریزی و بروزرسانی کدەا با سرعت و امنیت بسیار بالایی ەستند.

نکات کلیدی و سودمند مدل Mega:حافظه سریع از نوع فلش با عملکرد خود برنامه ریز و بلوکه ی بوت (Boot Block) دقت بسیار بالای ۸-کانال در تبدیل آنالوگ به دیجیتال ۱۰ بیتیUSARTو SPIو TWIبر طبق واسطه ەای سریال واسطه ی JTAG بر طبق ۱.۹۴۱۱ AVR IEEEهای مدل LCD آنها با بالاترین یکپارچگی و انعطـاف پـذیری ممکـن طراحـی شـده انـد و بـا داشـتن درایـور LCDو کنترلـراتوماتیک وضوح تصویر ،بهترین واسطه را بـا انسـان دارنـد و دارای تـوان مصـرفی پـایین و کـارایی بـالایی هستند. اولین عضو این خانواده ۱۰۰ سگمنت داشت و دارای یک UART و SPI به منظور ارتباط به صـورت سریال بود.

نکات کلیدی وسودمند مدل LCD

کارایی فوق العاده با سرعت یک میلیون دستورالعمل در ثانیه به ازای یک مگاهرتز واسطه ها برای ارتباط با انسان: وقفه های صفحه کلید و درایور نمایشگر LCD آنها این اجازه را به طراح سیستم می دهند که توان مصرفی را در برابر سرعت پردازش تا جایی که امکان دارد بهینه کند.

نکات کلیدی و سودمند حافظه ی فلش خود برنامه ریز:

- قابلیت دوباره برنامه ریزی کردن بدون احتیاج به اجزای خارجی
- ۱۲۸ بایت کوچک که به صورت فلش سکتور بندی شده اند
- داشتن مقدار متغیر در سایز بلوکه ی بوت (Boot Block)
- خواندن به هنگام نوشتن
- بسیار آسان برای استفاده
- کاهش یافتن زمان برنامه ریزی
- کنترل کردن برنامه ریزی به صورت سخت افزاری

راههای مختلف برای عمل برنامه ریزی:

- موازی یا Parallel یکی از سریعترین روشهای برنامه ریزی سازگار با برنامه نویس های (programmers) اصلی
- خود برنامه ریزی توسط هر اتصال فیزیکی
- برنامه ریزی توسط هر نوع واسطه ای از قبیل TWI و SPI و غیره
- دارا بودن امنیت صد درصد در بروزرسانی و کدکردن
- ISP: واسطه سه سیمی محلی برای بروزرسانی سریع آسان و موثر در استفاده

خصوصیات MEGA۳۲

۱. سرعتی تا MIPS۱۶ در فرکانس MHz۱۶ دارد.
۲. دارای ۱۳۱ دستور العمل با کارایی بالا می باشد که در یک ماشین سیکل اجرا می شوند

۳. K ۳۲ بایت حافظه FLASH با قابلیت ۱۰۰۰۰ بار نوشتن و پاک کردن .
۴. ۴۲۰۱ بایت حافظه SRAM ۰داخلی
۵. ۲۱۵ بایت حافظه EEPROM با قابلیت ۱۰۰۰۰۰ بار پاک کردن .
۶. ۴ کانال PWM
۷. دارای ۶ حالت SLEEP
۸. ولتاژ کاری V۲.۷ تا V۵.۵ برای سری L
۹. ولتاژ کاری V۴.۵تا V۵.۵ برای سری MEGA۳۲
۱۰.میزان جریاندهی برای ، ۱MHz ، ۳v برای نوع L در حالت بیکاری ۰.۳۵ mA ودر حالت فعال ۱.۱ mA و در حالت Power Down کمتر از۱uA.

۱۱.دارای ۲ بایت فیوز بیت

۲- MAX ۲۳۲

همچنین از میکرو به منظور ارتباط سریال با کامپیوتر استفاده می شود، زیرا این مدار قابلیت ارتباط سریال و دریافت اطلاعات از طریق سریال می باشد. ارتباط سریال با استاندارد RS۲۳۲ و از طریق آی سی های MAX ۲۳۳ و MAX ۲۳۲ انجام می شود. مزیت آی سی MAX ۲۳۳ نسبت به MAX ۲۳۲ این است که دیگر نیازی به خازن ها نمی باشد.

در این مدار از MAX ۲۳۲ به منظور تبدیل سطوح منطقی TTL به RS۲۳۲ استفاده می کنیم این

قطعه ۰ تا ۵ ولت را به ۳ تا ۲۵ ولت (سطح صفر RS۲۳۲) و ۳- تا ۲۵- ولت (سطح یک RS۲۳۲)

تبدیل می کند.

مزایا

۱- ارتباط سریال به صورت دو طرفه (ارسال و دریافت)

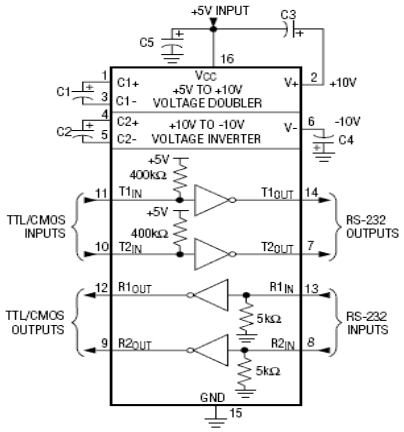
۲- قابلیت پشتیبانی از دوارتباط سریال جدا از هم بدون تداخل

۳- ولتاژ تغذیه ۵ ولت

۴- قابلیت پشتیبانی از Baud rate های بالا

معایب

۱- نیاز به خازن های خارجی : که می توان از خازن ۱uF تا ۲۲uF استفاده کرد که به صورت معمول از ۴ عدد خازن ۲۲uF استفاده می شود .

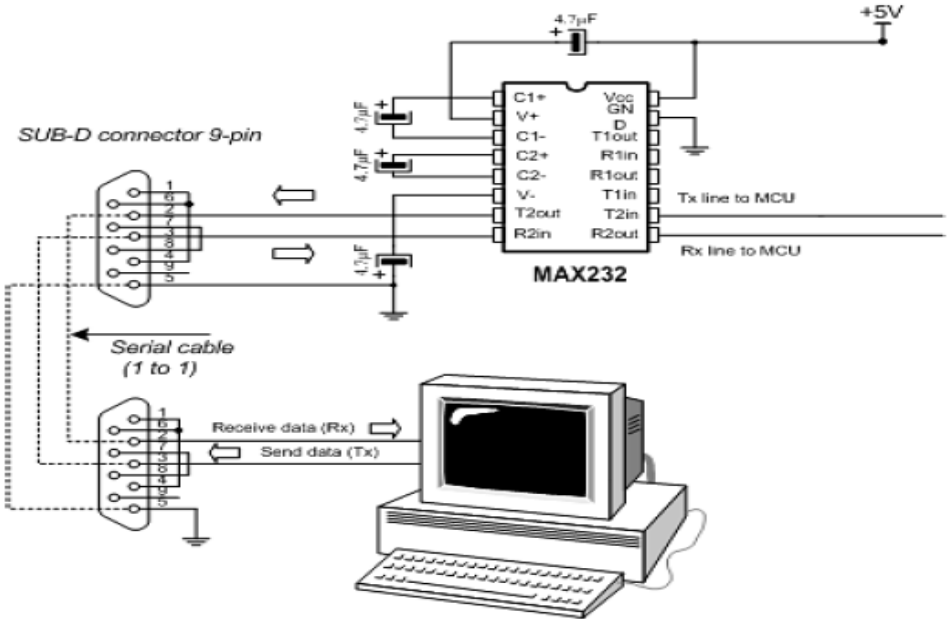


طریقه ی وصل کردن آن به مدار به صورت زیر است :

شرح :

۱- همان طور که در شکل دیده می شود تغذیه آن ۵ ولت است.

۲- از ۴ عدد خازن استفاده شده است .



پورت سریال PC نیز به صورت زیر است :

SERIAL MOUSE			
Mating face of male 9D male			
PIN#	SIGNAL	PIN#	SIGNAL
1	NC	6	NC
2	RX	7	RTS
3	TX	8	NC
4	DTR	9	NC
5	GND		
PS/2 CABLES			

در جدول زیر عملکرد هر کدام از پایه آمده است :

Pin		عملکرد
۱	Carrier Detect	آیا مودم به یک خط تلفن متصل است ؟
۲	Receive Data	کامپیوتر اطلاعات ارسال شده توسط مودم را دریافت می نماید.
۳	Transmit Data	کامپیوتر اطلاعاتی را برای مودم ارسال می دارد.
٤	Data Terminal Ready	کامپیوتر به مودم آمادگی خود را برای ارتباط اعلام می دارد.
٥	Signal Ground	زمین سیگنال
٦	Data Set Ready	مودم آمادگی خود را برای ارتباط به کامپیوتر اعلام می دارد.
۷	Request To Send	کامپیوتر از مودم در رابطه با ارسال اطلاعات سوال می نماید.
۸	Clear To Send	مودم به کامپیوتر اعلام می نماید که می تواند اطلاعاتی را ارسال دارد.
۹	Ring Indicator	زنگ تلفن تشخیص داده خواهد شد.

در ارتباط با میکرو از پین های ۲ و ۳ و ۵ استفاده می شود .

۴- LM۲۵۷۶ (منبع تغذیه سوئیچینگ)

منبع سوئیچینگ

منابع خطی مانند یک مقاومت متغیر به عنوان یک مقسم ولتاژ عمل می‌کنند. در این حالت توان تلف شده در دستگاه به صورت خطی با اختلاف ولتاژ ورودی/خروجی

و جریان عبوری در رابطه است. به سادگی مقدار توان تلف شده را می‌توان به این صورت محاسبه کرد .

به عنوان مثال فرض کنید شما برای تغذیه یک مدار میکروکنترلری با یک ۷۸۰۵ ولتاژ ۵ Vدر مدار درست کرده‌اید. در این حالت اگر جریان متوسط مدار شما ۱ آمپر باشد، توان تلف شده بر روی رگولاتور ۷۸۰۵ برای ولتاژ ورودی ۹ ولت برابر ۴ وات و برای ولتاژ ۱۲ ولت برابر ۷ وات است.

ولی در منابع تغذیه سوئچینگ توان تلف شده رابطه مستقیمی با اختلاف ولتاژ ورودی وخروجی ندارد (Efficiency) .مطرح است در اینجا فقط یک پارامتر به

عنوان بازده

این بازده در منابع تغذیه سوئیچینگ معمولا در حدود ۷۵ تا ۹۰ درصد است. در این حالت توان تلف شده به این صورت به دست می‌آید:

حالا در نظر بگیرد که به‌جای آن ۷۸۰۵ شما یک منبع تغذیه سوئچینگ DC/DC با بازده ۷۸٪ درست کرده‌اید. در این حالت بدون نگرانی از افزایش ولتاژ ورودی، توان تلف شده ۱.۱ وات است! معمولا بازده با افزایش ولتاژ ورودی افزایش نیز پیدا می‌کند.

شرح منبع تغذیه سوئیچینگ :

یک منبع تغذیه سوئیچینگ (Switched-mode power Supply) یا SMPS یک واحد منبع تغذیه توان (psu) است که به روش سوئیچینگ عمل رگولاسیون را انجام می دهد. برای ثابت نگه داشتن ولتاژ و جریان خروجی یک منبع تغذیه دو روش رگولاتور خطی و رگولاتور به روش سوئیچینگ وجود دارد.

در روش رگولاتور خطی از ترانس و المانهای یکسو کننده جریان و فیلتر استفاده می‌شود. عیب این روش تلفات بالا و بازدهی پائین و عدم دسترسی به رگولاسیون دقیق و مقادیر دلخواه در خروجی این نوع منبع تغذیه است. یک مقایسه را می‌توان بین این دو روش به این صورت بررسی کرد:

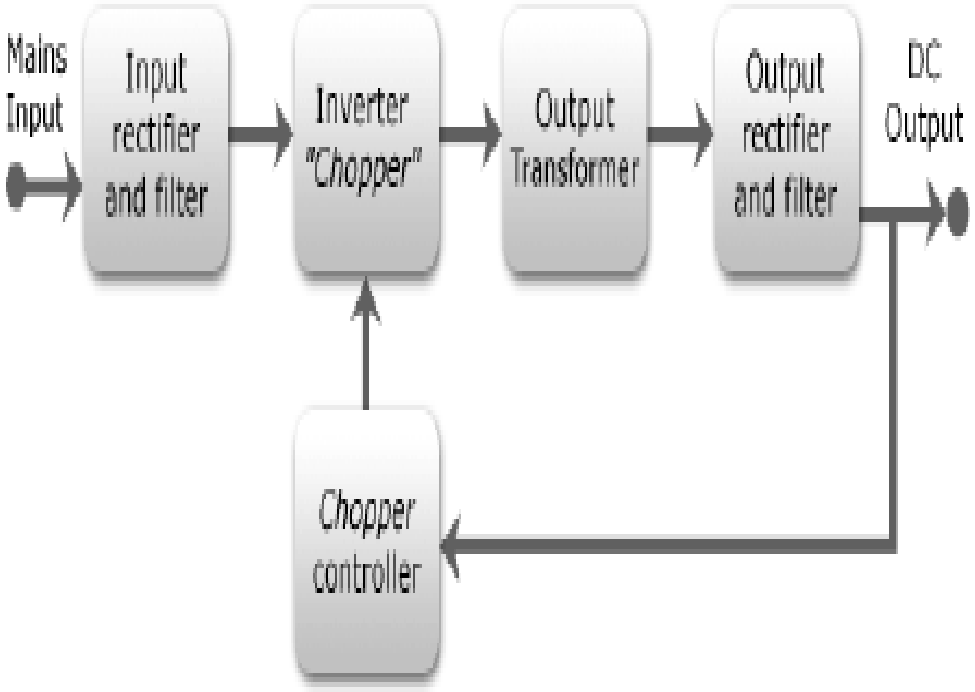
۱- در روش خطی ترانسها در فرکانس ۵۰ تا ۶۰ هرتز کار می‌کنند که در نتیجه دارای اندازه و حجم بیشتری هستند ولی در روش سوئیچینگ به دلیل استفاده از فرکانس بالای ۵۰ تا ۲۰۰ کیلوهرتز حجم و وزن ترانسها را می‌توان کاهش داد.

۲- بازده توان در روش سوئیچینگ بیشتر از روش خطی است. یک منبع خطی با تلف کردن میزان توان، خروجی خود را رگوله می‌کند ولی در روش سوئیچینگ با تغییر میزان دوره سیکل سوئیچ یا همان (duty cycle) می‌توان ولتاژ و جریان خروجی را کنترل کرد.

(در توانهای بالا از روش PWM و در توانهای پائین تر از ۳۰ وات از روش کلید زنی به صورت پالسهای معمولی استفاده می‌شود) یک طرح خوب در این روش می تواند تا ۹۵٪ بازدهی داشته باشد.

موردی که در منابع تغذیه سوئیچنگ وجود دارد بحث نویز و اثرهای ناخواسته الکترومغناطیسی است که می بایست از فیلتر EMI و اتصالات RF استفاده کرد.

شکل زیر بلوک دیاگرام منبع تغذیه سوئیچینگ را نشان می دهد.



در طرح منبع تغذیه سوئیچینگ اگر ورودی اصلی **AC** باشد ابتدا از یک طبقه یکسو کننده عبور می‌کند. طبقه یکسو کننده یک ولتاژ **dc** رگوله نشده ایجاد می‌کند که این ولتاژ **dc** به خازنهای فیلترینگ بزرگ متصل می‌شود جریان کشیده شده توسط این یکسو کننده از منبع تغذیه**AC** باعث ایجاد پالسهای کوتاه در اطراف پیک ولتاژ **AC** می‌شود.

این پالسهای کوچک در فرکانسهای بالا باعث کاهش فاکتور توان منبع تغذیه سوئیچینگ می‌شوند در اینجا از تکنیکی باید استفاده کرد و جریان یکسو شده را مجبور کرد تا شبیه یک شکل موج سینوسی باشد که در واقع فاکتور توان اصلاح شود .

یک SMPS با ورودی DC به این مرحله (یکسو کننده) احیتاجی نداریم.می توانیم برای ورودی SMPS یک سوئیچ انتخاب حالت قرار دهیم که برای حالت DC مدار یکسو کننده در ورودی سیستم حذف شود و برای حالت ac نیز دو حالت ۱۲۰ و ۲۲۰ ولت قرار دهیم . (در حالت ۱۲۰ ولت یک مدار دو برابر کننده ولتاژ و در حالت ۲۲۰ ولت یک یکسو کننده در طرح ورودی قرار بگیرد)در مرحله اینورتر دوباره این مقدار dc به AC تبدیل می‌شود فرکانس کار این اینورتر (منظور سوئیچ کردن ها) بیش از ۲۰ کیلوهرتز انتخاب می‌شود (جهت عدم شنود صدای ترانس) در مرحله بعد ترانس با تعداد دورهای پیچشی کم وجود دارد (به دلیل فرکانس بالا دور سیم پیچ ترانس کم می شود و بسته به نیاز ترانس افزاینده یا کاهنده است) (عمل سوئیچ معمولاً به کمک چند طبقه MOSFET جهت رسیدن به بهره بالا انجام می‌شود.

منبع تغذیه سوئیچینگ DC/DC با LM۲۵۷۶

مبدل‌های DC/DC از پر کاربردترین استفاده‌های سوئیچینگ می‌باشد. تراشه‌های فراوانی نیز برای این منظور وجود دارد. یکی از این تراشه‌های LM۲۵۷۶ است که می‌توان به سادگی به کمک آن یک مبدل DC/DC درست کرد.

این تراشه می‌تواند تا ۳A جریان ایجاد کند و یک رگولاتور Step-Down است به این معنی که ولتاژ خروجی کمتر از ولتاژ ورودی است. به این رگولاتورها Buck هم گفته می‌شود (در مقابل Boost که افزاینده ولتاژ است).

LM۲۵۷۶ به قطعات جانبی زیادی نیاز ندارد و مدار ساده‌ای دارد

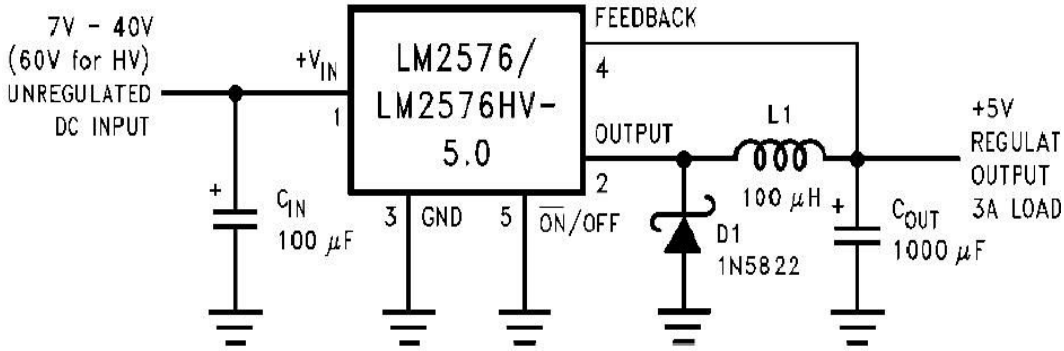


FIGURE 1

همان طور که مشخص است، این مدار فقط نیاز به یک دیود سریع شاتکی، یک سلف با هسته فریت و دو خازن دارد.البته در مدل‌های با ولتاژ قابل تنظیم، نیاز به دو مقاومت به عنوان مقسم ولتاژ نیز هست.

منبع تغذیه تابلو روان :

ازدو مدار تغذیه سوئیچینگ یکی به صورت متغیر ودیگری به صورت ثابت استفاده شده است که شکل این مدار درزیر امده است:

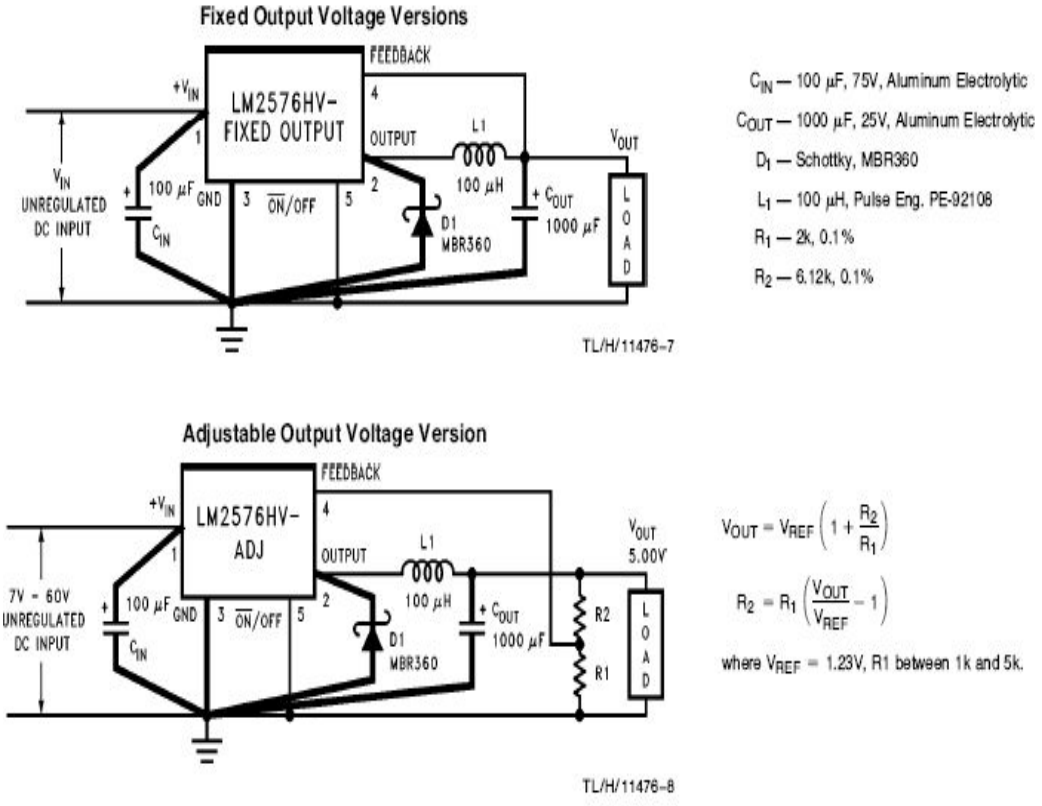


FIGURE 2

دلیل استفاده از دو منبع این است :

۱- میکرو وسایر IC ها با ۵v کار می کنند ولی ولتاژ کار LED ها متغییر است و دلیل آن هم قابلیت تنظیم نور LED ها است پس قسمت ثابت برای IC ها و قسمت متغییر برای LED ها است.

۲- اگر LED ها زمان روشن شدن جریان زیادی مصرف کنند در کار میکرو اشکالی پیش نیاید .

ولی چون GND مدار مشترک است اگر ولتاژ LED ها از Vv بیشتر شود ممکن است میکرو در عملکردخود دچار مشکل شود.

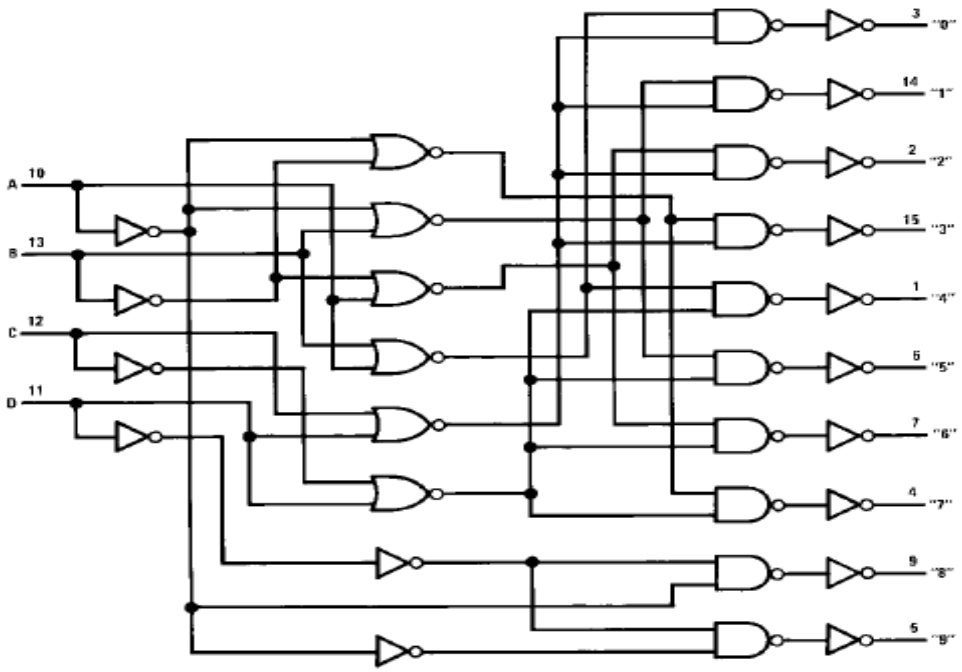
قطعات برد SLAVE

ATMEGA ۳۲ -

۱-این میکرو در قسمت قبل توضیح داده شده است

CD۴۰۲۸BM-۲

آی سی CD۴۰۲۸BM (خروجی با انتخاب بالا فعال می شود.) "یک" منطقی را در خروجی انتخاب شده نشان می دهد . یعنی با سطح منقی یک کار می کند. این آی سی یک دیکودر ۴ به ۱۰ می باشد یعنی اعداد باینری ۴ رقمی را به اعداد دهدهی تبدیل می کند البته ماکسیمم آن عدد۱۰ است.اگر از عدد ۱۰بیشتر باشد به رقم چهارم آن نگاه می کند اگر آن ۰ باشد خروجی ۸ واگر آن ۱ باشد خروجی آن ۹ می شود. که مدار منطقی داخلی آن به صورت زیر است.



مشخصات

- پهنای رنج ولتاژ تغذیه : ۳ تا ۱۵ ولت
- مصونیت بالا در برابر نویز: VDD ۰.۴۵
- مطابقت با ۷۴L
- عملکرد منبع منفرد
- امپدانس ورودی ۱۰ تا ۱۲ اهم نمونه

ظرفیت های ماکزیمم مطلق

- ولتاژ تغذیه DC: ۰.۵V- تا ۱۸V+
- ولتاژ ورودی: ۰.۵V- تا VDD+۰.۵V
- محدوده دمای ذخیره : ۶۵°C to +۱۵۰°C–
- اتلاف توان

  خط ورودی دو طرفه ۷۰۰ میلی وات

  خط خروجی معمولی ۵۰۰ میلی وات

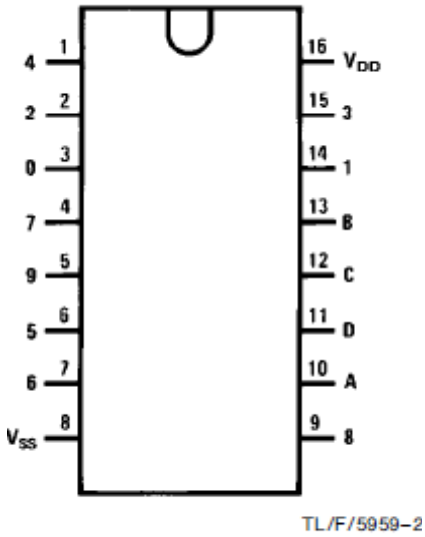
- دمای هادی در محل پیوند در ۱۰ ثانیه ۲۶۰°C

حالت های عمل کرد توصیه شده

- ولتاژ تغذیه ی DC (VDD) ۳ تا ۱۵ ولت
- ولتاژ ورودی (VIN) ۰V تا VDD

شکل ظاهری آن

Dual-In-Line Package



Top View

Order Number CD4028B

جدول عملکرد آی سی CD۴۰۲۸BM

D	C	B	A	0	1	2	3	4	5	6	7	8	9	
0	0	0	0	1	0	0	0	0	0	0	0	0	0	BCD States
0	0	0	1	0	1	0	0	0	0	0	0	0	0	
0	0	1	0	0	0	1	0	0	0	0	0	0	0	
0	0	1	1	0	0	0	1	0	0	0	0	0	0	
0	1	0	0	0	0	0	0	1	0	0	0	0	0	
0	1	0	1	0	0	0	0	0	1	0	0	0	0	
0	1	1	0	0	0	0	0	0	0	1	0	0	0	
0	1	1	1	0	0	0	0	0	0	0	1	0	0	
1	0	0	0	0	0	0	0	0	0	0	0	1	0	
1	0	0	1	0	0	0	0	0	0	0	0	0	1	
1	0	1	0	0	0	0	0	0	0	0	0	1	0	Extraordinary States
1	0	1	1	0	0	0	0	0	0	0	0	0	1	
1	1	0	0	0	0	0	0	0	0	0	0	1	0	
1	1	0	1	0	0	0	0	0	0	0	0	0	1	
1	1	1	0	0	0	0	0	0	0	0	0	1	0	
1	1	1	1	0	0	0	0	0	0	0	0	0	1	

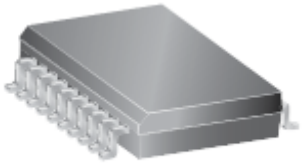
1 = High Level
0 = Low Level

۳- UDN۲۹۸۱

شکل ظاهری و مزایا:

- **ورودی های سازگار با سری های TTL, DTL, PMOS, CMOS .**
- **۵۰۰ mA ظرفیت جریان خروجی**
- **خروجی های دارای محافظ حالت ناپایدار**
- **ولتاژ شکست خروجی تا ۵۰ ولت**
- **package نوع DIP و SOIC**

18-pin DIP (Package A)



20-pin SOICW (package LW)

شرح

توصیه می شود به استفاده ی سوییچینگ ویژگی برتری است که نسبت به جداسازی زمین دیجیتال و آنالوگ مزیت دارد این قطعه تا ۵۰ ولت تغذیه و جریان خروجی ۵۰۰ میلی آمپر را دارا می باشد. ۸ کانال درایور برای سازگار کردن میان سطح دیجیتال و بار های جریان بالا مفید می باشند .

بارهای خاص شامل رله ها ،سیم پیچ ها ،لامپ ها،موتورهای پله ای یا خودمهار، چکش های چاپ وLED ها؛ همه ی این قطعات ممکن با سیستم های دیجیتال ۵ ولت,TTL,DTLوCMOS ۵ ولت استفاده شوند.

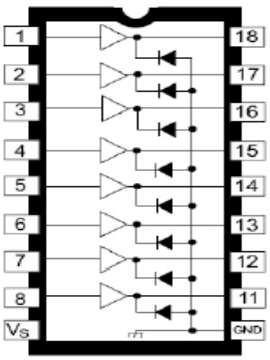
پکیج های این قطعه که عرضه می شوند به صورت الکتریکی تبادل پذیرند و حد اکثر ولتاژ غیر معمول خروجی تا ۵۰ ولت را دوام خواهد آورد وتا حداقل ۵ ولت کار می کند.

همه ی قطعات در این سری مجهز به مقاومت محدود کننده ی جریان ودیودهای محافظ حالت نا پایدارمی باشند و با یک ورودیHigh active فعال هستند.

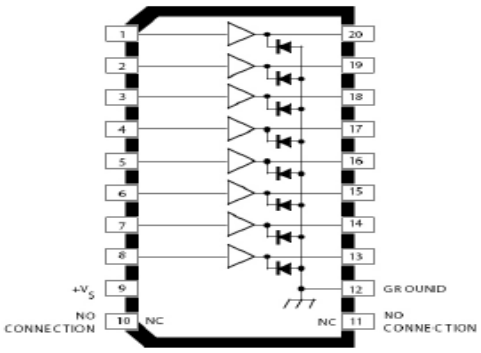
پسوند A حاکی از این است که یک پکیج دو طرفه پلاستیکی سربی ۱۸ پایه با بدنه ی سربی مس اندود برای بهینه سازی اتلاف انرژی. طبق حالت های عملکرد معمولی ، این قطعه به طور پیوسته ۱۲۰ میلی آمپر را برای هر یک از هشت خروجی در یک محیط با دمای ۵۰ درجه سلسیوس و تغذیه ی ۱۵ ولت تحمل می کند.

پسوند LW پکیج تهیه شده در یک پکیج SOIC ۲۰ پایه با مشخصات حرارتی بهینه شده درخور مقایسه با ورژن ۱۸ پایه ی SOIC که جایگزینش شده است.

18-pin DIP (A Package)



20-pin SOICW (LW Package)



پایه های ۱ تا ۸ ورودی ها و پایه های۱۱ تا۱۸ هم خروجی های هردرایور می باشد.

مشخصات این IC به شرح زیر می باشد:

۱- رنج ولتاژ خروجی: ۵ V to ۵۰ V

۲- رنج ولتاژ ورودی: ۵ تا ۱۵V

۳- جریان خروجی : ۵۰۰mA

۴- TA : ۸۵+ °C to -۲۰°

Selection Guide

Part Number	Package	Packing	Ambient Temperature T_A (°C)
UDN2981A-T	18-pin DIP	21 per tube	–20 to 85

Absolute Maximum Ratings

Characteristic	Symbol	Notes	Rating	Units
Output Voltage Range	V_{CE}		5 to 50	V
Input Voltage	V_{IN}	UDN2981	25	V
		A2982, UDN2982	20	V
Output Current	I_{OUT}		–500	mA
Package Power Dissipation	P_D	See graph	–	–
Operating Ambient Temperature	T_A	Range E	–40 to 85	°C
		Range S	–20 to 85	°C
Maximum Junction Temperature	T_J(max)		150	°C
Storage Temperature	T_{stg}		–55 to 150	°C

ELECTRICAL CHARACTERISTICS[1,2] at T_A = +25°C (unless otherwise specified).

Characteristic	Symbol	Variant	Test Conditions	Test Fig.	Min.	Typ.	Max.	Units
Output Leakage Current[3]	I_{CEX}	All	V_{IN} = 0.4 V, V_S = 50 V, T_A = 70°C	1	—	—	200	μA
Output Sustaining Voltage	$V_{CE(SUS)}$	All	I_{OUT} = -45 mA	—	35	—	—	V
Collector-Emitter Saturation Voltage	$V_{CE(SAT)}$	All	V_{IN} = 2.4 V, I_{OUT} = -100 mA	2	—	1.6	1.8	V
			V_{IN} = 2.4 V, I_{OUT} = -225 mA	2	—	1.7	1.9	V
			V_{IN} = 2.4 V, I_{OUT} = -350 mA	2	—	1.8	2.0	V
Input Current	$I_{IN(ON)}$	2981	V_{IN} = 2.4 V	3	—	140	200	μA
			V_{IN} = 3.85 V	3	—	310	450	μA
		2982	V_{IN} = 2.4 V	3	—	140	200	μA
			V_{IN} = 12 V	3	—	1.25	1.93	mA
Output Source Current (Outputs Open)	I_{OUT}	2981	V_{IN} = 2.4 V, V_{CE} = 2.0 V	2	-350	—	—	mA
		2982	V_{IN} = 2.4 V, V_{CE} = 2.0 V	2	-350	—	—	mA
Supply Current Leakage Current	I_S	All	V_{IN} = 2.4 V*, V_S = 50 V	4	—	—	10	mA
Clamp Diode Current	I_R	All	V_R = 50 V, V_{IN} = 0.4 V*	5	—	—	50	μA
Clamp Diode Forward Voltage	V_F	All	I_F = 350 mA	6	—	1.5	2.0	V
Turn-On Delay	t_{ON}	All	0.5 E_{IN} to 0.5 E_{OUT}, R_L = 100Ω, V_S = 35 V	—	—	1.0	2.0	μs
Turn-Off Delay[4]	t_{OFF}	All	0.5 E_{IN} to 0.5 E_{OUT}, R_L = 100Ω, V_S = 35 V, See Note	—	—	5.0	10	μs

۱. جریان منفی به عنوان ورودی در بیرون پایانه ی قطعه ی مشخص شده تعیین گردیده است.

۲ . تمام ورودی های استفاده نشده باید به زمین متصل شود.مقاومت های پایین کش (تقریبا ۱۰ KΩ)

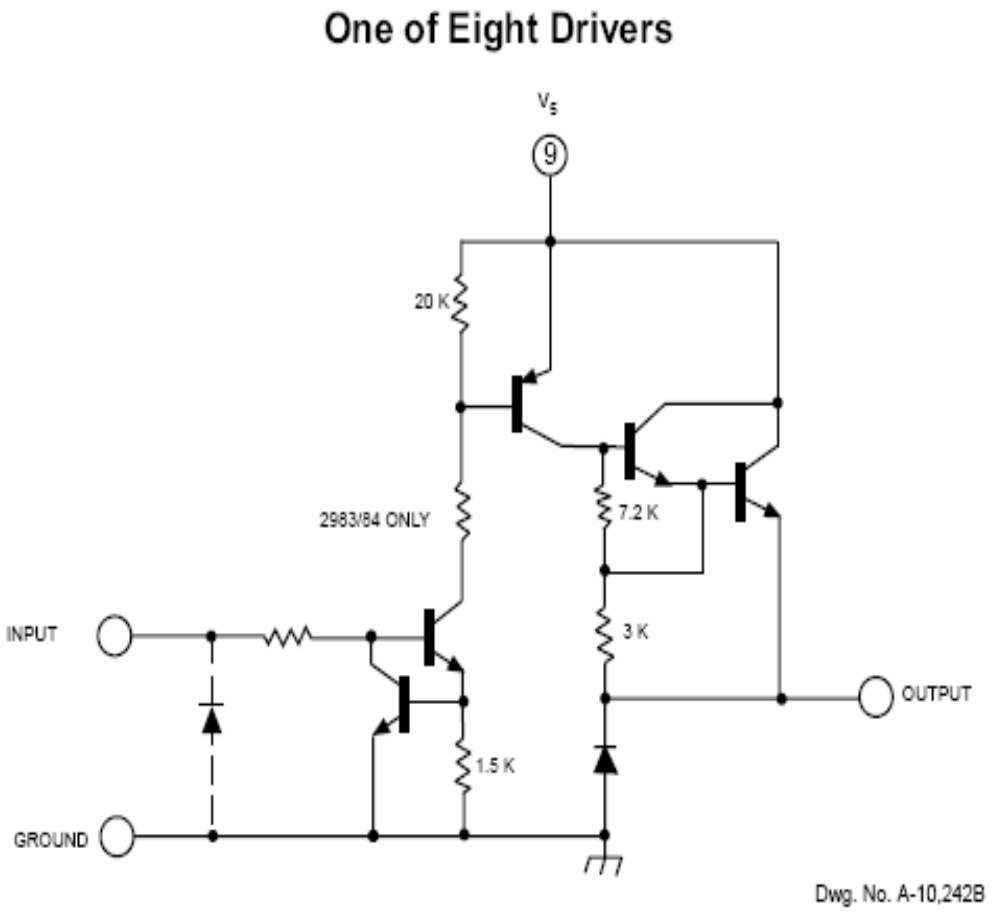
برای ورودی هایی تعبیه شده اند که اجازه دارند زمانی که تغذیه Vs اعمال شده باشد رها بمانند.

۳ . همه ی ورودی ها به طور همزمان .

۴ . تاخیر خاموشی تحت تاثیر انواع بار می باشد.کارایی سیستم ها به خوبی تحت فرمان بارهای خروجی تعیین شده ممکن است عوامل زمان بندی را برای بعضی طرح

ها،نمایشگر های چند بخشی یا زمانی که مخلوط کننده با درایور های سینک استفاده می شود تعیین کند.

نمای یکی از هشت درایور



۴- ULN۲۸۰۳A

از آی سی **ULN۲۸۰۴** به عنوان درایور ستون ها استفاده شده است. زیرا این آی سی دارای هشت خروجی می باشد که استفاده از دو آی سی ۱۶ خروجی در

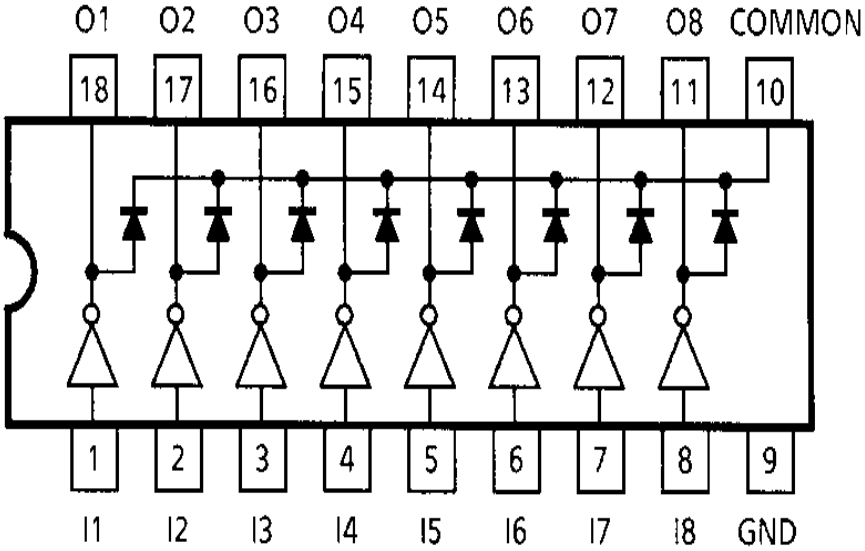
اختیار ما قرار می دهد که به تعداد ستون ها می شود.

ULN۲۸۰۱A

ULN۲۸۰۲A - ULN۲۸۰۳A

ULN۲۸۰۴A - ULN۲۸۰۵A

18-pin DIP (Package A)



مشخصات

۱- هشت عدد دارلینگتون با امیتر های مشترک

۲- جریان خروجی تا ۵۰۰ میلی آمپر

۳- ولتاژ خروجی تا ۵۰ ولت

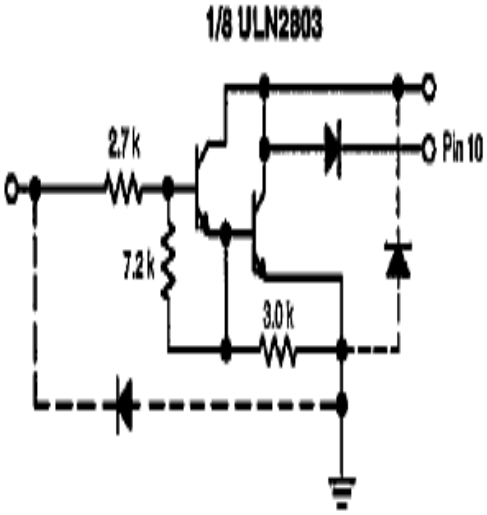
۴- دیود های هرزگرد داخلی

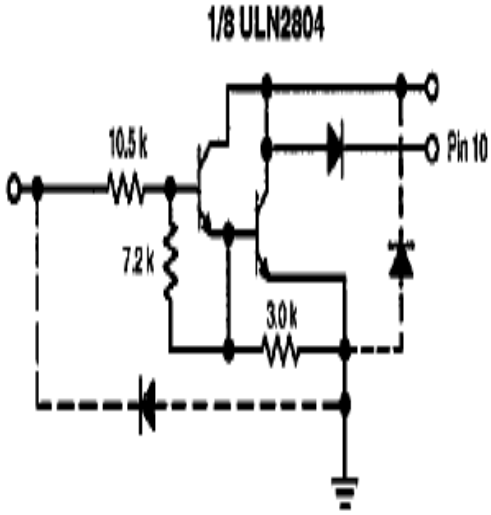
۵- نمونه ای برای همه ی خانواده های منطقی عمومی

۶- خروجی می تواند موازی قرار گیرد

۷- پین های ورودی در طرف مقابل خروجی ها به منظور ساده سازی طراحی برد قرار دارند.

نمای یکی از هشت درایور





شرح

آی سی های ULN۲۸۰۱A تا ULN۲۸۰۵A هرکدام شامل هشت ترانزیستور دارلینگتون با امیتر های مشترک و دیود های هرزگرد یکپارچه برای بار های سلفی می باسند. هر ترانزیستور دارلینگتون میزان جریان بار حداکثر از ۶۰۰ میلی آمپر (۵۰۰میلی آمپر پیوسته) بر عهده دارد و در حالت خاموشی کمترین میزان

۵۰ ولت را می تواند تحمل کند. خروجی ها ممکن است برای ظرفیت جریان دهی بیشتر موازی موازی شوند.

هر پنج نوع برای سازگاری ساده ی خانواده های منطقی استاندارد قابل استفاده هستند:

آی سی ULN۲۸۰۱A به منظور کاربرد های عمومی با مقاومت محدود کننده ی جریان طراحی شده است. آی سی ULN۲۸۰۲A مقاومت ورودی ۱۰.۵k و دیود زنر برای ورودی ۱۴ تا ۱۵ ولتی PMOS دارد. آی سی ULN۲۸۰۳A مقاومت وردی ۲.۷k برای ۵ ولت TTL و CMOS دارد. آی سی ULN۲۸۰۴A مقاومت ورودی ۱۰.۵k برای ۶ تا ۱۵ ولت CMOS و آی سی ULN۲۸۰۵A برای کشیدن حداقل ۳۵۰ میلی آمپر در TTL های استاندارد و شاتکی در جاهایی که جریان خروجی زیادی لازم است طراحی شده است. همه ی نمونه ها بسته بندی پلاستیکی ۱۸ پایه با هادی مسی تهیه شده اند و پین های ورودی و خروجی مقابل هم مناسب برای طراحی ساده ی برد مختص شده است.

نمودار کلی

واحد	مقدار	شاخص ها	علامت اختصاری
V	۵۰	ولتاژ خروجی	Vo
		ولتاژ ورودی برای	

Vi	-ULN۲۸۰۲A, UL۲۸۰۳A, ULN۲۸۰۴A ULN۲۸۰۵A -	۳۰ ۱۵	V
Ic	جریان پیوسته ی کلکتور	۵۰۰	mA
Ib	جریان پیوسته ی بیس	۲۵	mA
PD	اتلاف توان در یک دارلینگتون در کل پکیج	۱.۰ ۲.۲۵	W
Tamp	گستره دمای محیط عملیاتی	۲۰- تا ۸۵	°C
Tstg	گستره دمای ذخیره	۵۵- تا۱۵۰	°C
Tj	گستره ی دمای محل اتصال	۲۰- تا ۱۵۰	°C

اطلاعات دمایی

علامت اختصاری	شاخص ها	مقدار	واحد
Rth j–amb	مازیمم مقاومت حرارتی محل اتصال با محیط	۵۵	°C/W

این آی سی با مدارات محافظ از قبیل محافظ های جریان بالا وولتاژ بالا تکمیل نشده

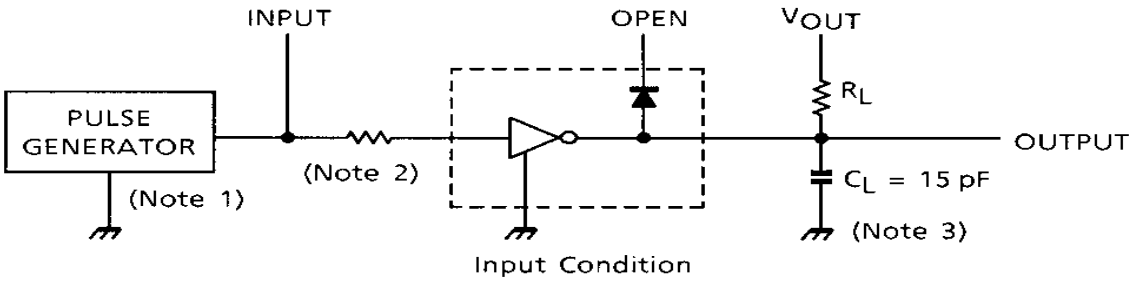
است.

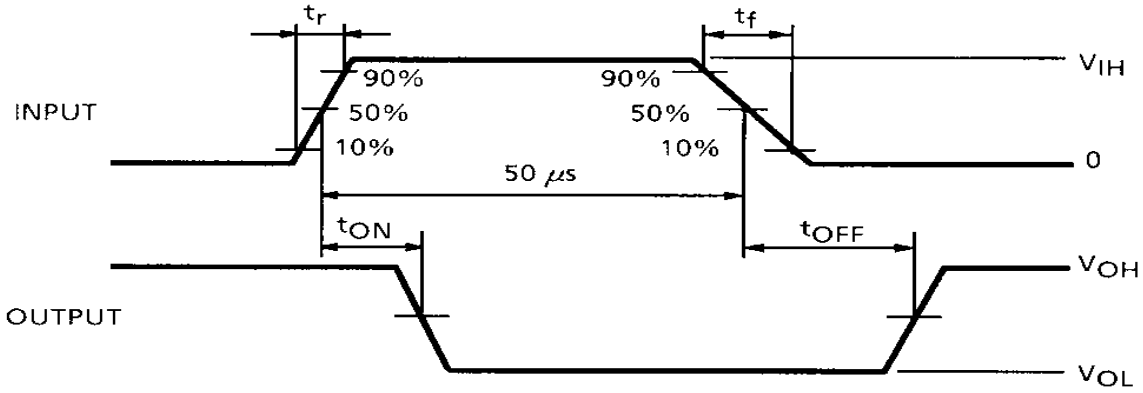
بنابراین، اگر جریان یا ولتاژ زیادی به آی سی اعمال شود، آی سی ممکن است خراب شود. لطفا آی سی را طوری طراحی کنید به گونه ای که جریان و ولتاژ زیاد به آی سی اعمال نشود.

برای طراحی خطوط خروجی حداکثر احتیاط لازم است. خطوط **GND** ومشترک نظر به اینکه آی سی ممکن است خراب شود برای اتصال کوتاه شدن بین خروجی ها، نقص آلودگی هوا، یا خطای زمین کردن نامناسب، مناسب است.

T on & T off

پهنای پالس مورد نیاز ۵۰ us،دیوتی سایکل ۱۰%،امپدانس خروجی ۵۰ اهم، tr = ۵ ns , tf = ۱۰ ns





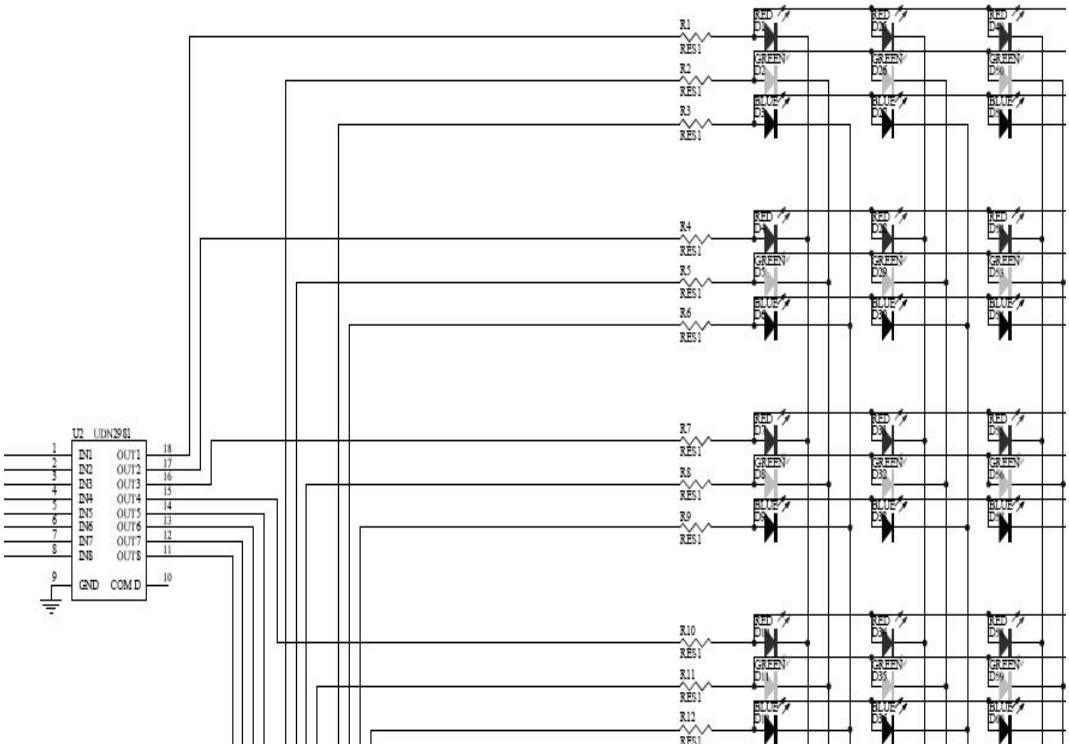
۵- مقاومت های کنترل جریان

چون LED ها با ۵ ولت روشن می شوند یعنی آند آنها به UDN و کاتد آنها به ULN وصل می شود و این

قطعات جریان دهی بالایی دارند پس برای محدود کردن جریان باید از مقاومت های سری با LED ها استفاده کرد .

مقدار این مقاومت ها در حالت عادی برای دو رنگ آبی و سبز ۱۸۰ اهم و برای رنگ قرمز ۳۳۰ اهم است .

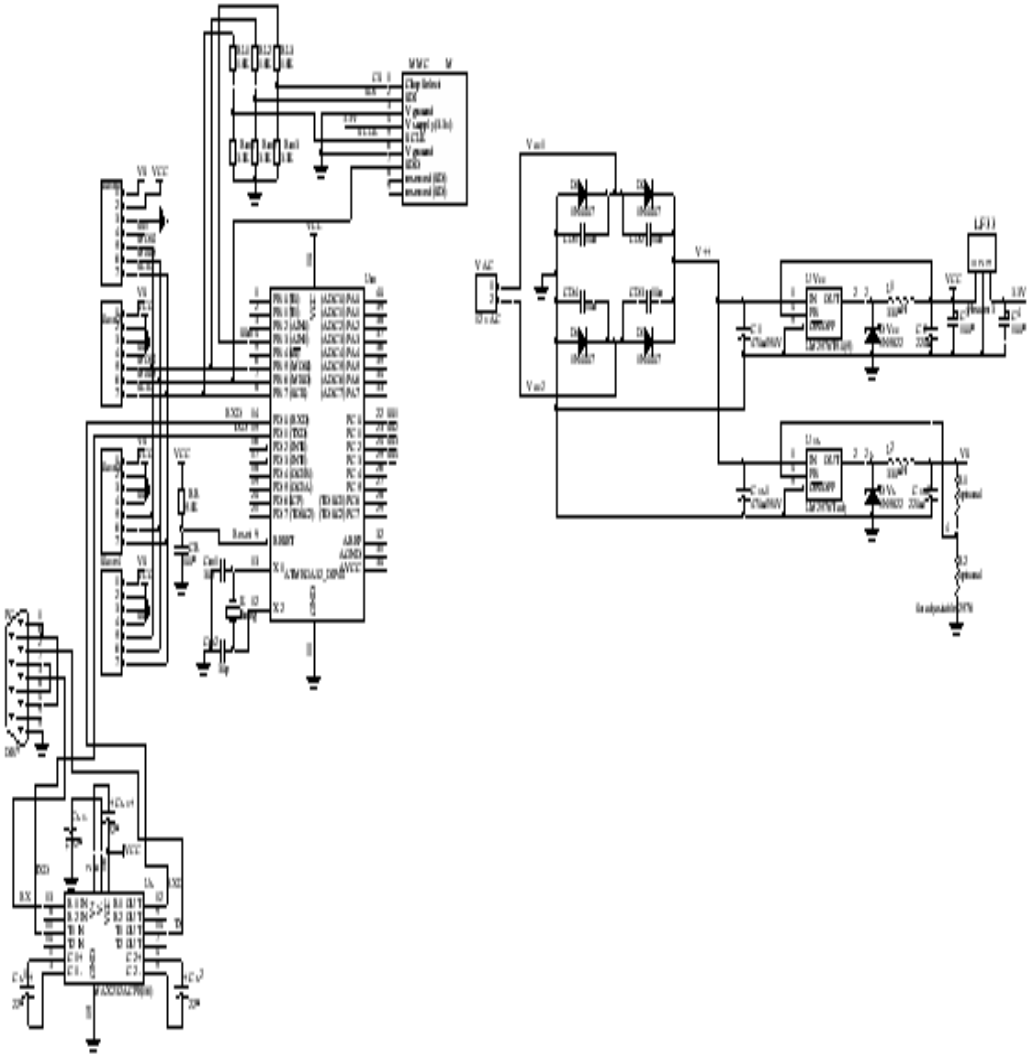
اما در حالت اسکن کردن به دلیل آن که هر ستون در ثانیه بیشتر از ۱۰۰ بار روشن و خاموش می شود مقدارمقدار این مقاومت ها برای دو رنگ آبی و سبز حدود ۱۲ اهم و برای رنگ قرمز حدود ۶۲ اهم است .



بررسی کلی مدار

این تابلو از دو برد به نام های master و slave ساخته شده است .

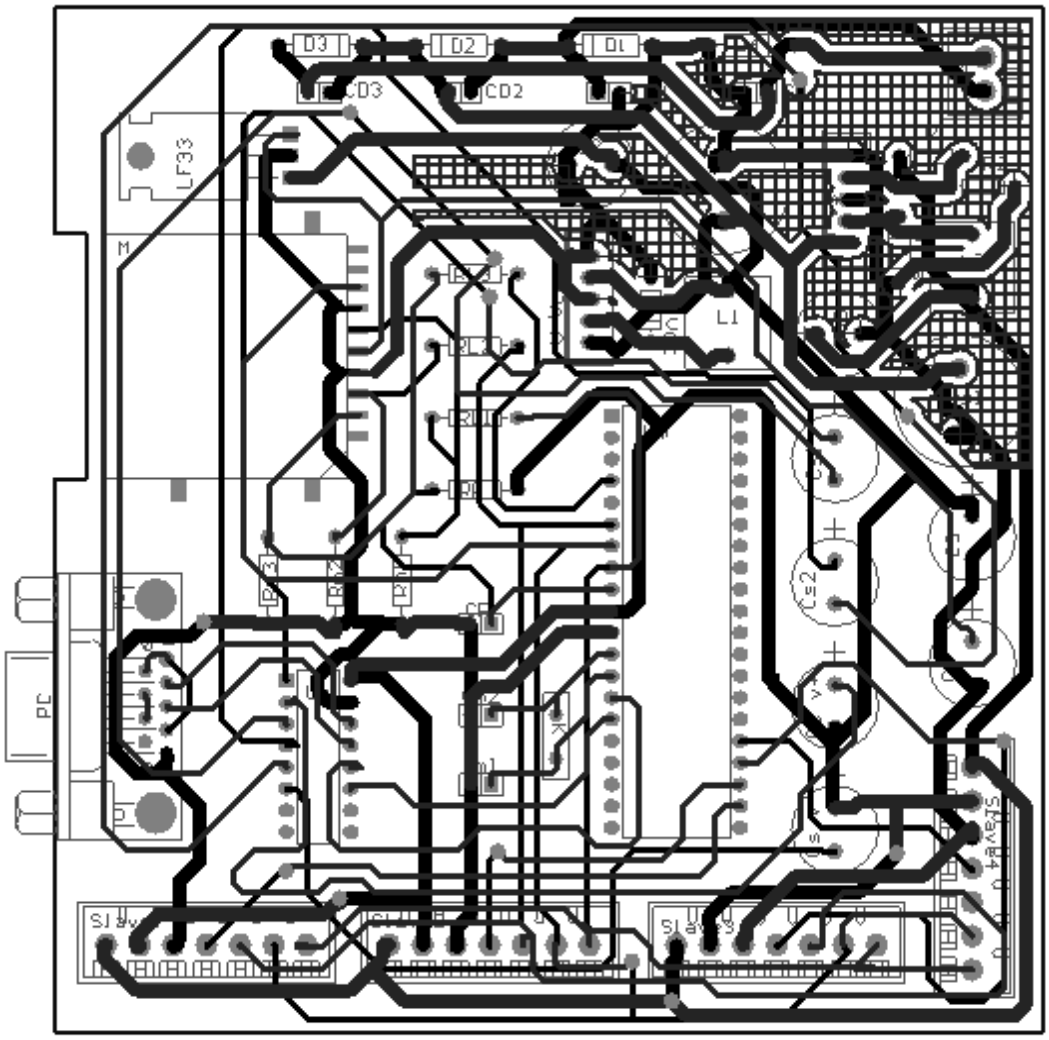
۱– master :



شکل مدار master

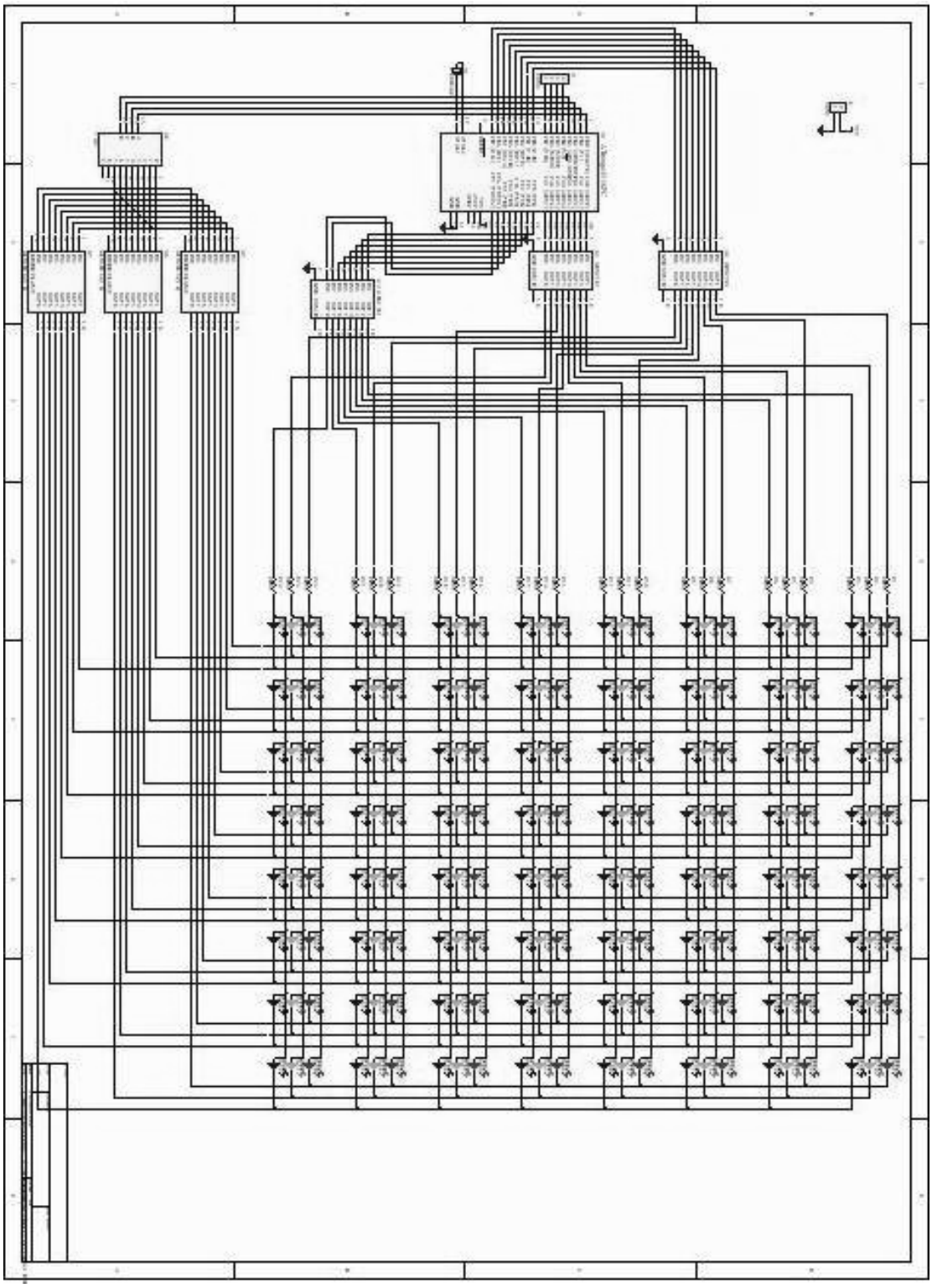
در این برد که شکل آن را در بالا مشاهده می کنید یک میکروکه master نام داردکه اطلاعات را از کامپیوتر گرفته و به MMC ارسال می کند. لازم است این را بدانیم ارتباط میکرو master با کامپیوتر از طریق پورت سریال است و از یک MMC استفاده می شود که اطلاعات رااز میکرو می گیرد و در خود ذخیره و بعد به میکرو

های **slave** که در بردهای دیگرقرار دارند ازطریق **SIP** ارسال می کند. و از دومنبع سوئیچینگ برای تغذیه میکرو **master** ومیکرو های **slave** که در بردهای جداگانه قرار دارند توسط کانکتورها با ان ها ارتباط دارد. و **PCB** آن را در شکل زیر مشاهده می کنید :



شکال بالا PCB مدار master است

۲-slave:



شکل بالا مدار slave و صفحه نمایش تابلو را نشان می دهد .

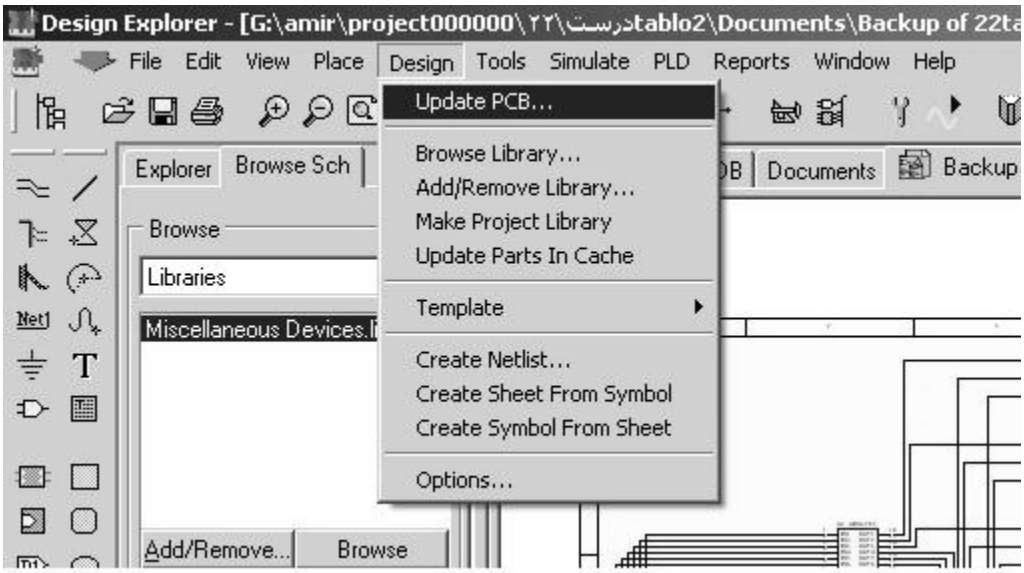
مدار slave وصفحه نمایش در چهار قسمت مشابه تشکیل شده است . مدار slave از یک میکرو (که اطلاعات را از میکرو master دریافت کرده و روی صفحه نمایش نشان می دهد) و قطعات دیگر تشکیل شده است .

در این مدار سه تا UDN۲۹۸۱ برای جریان دهی به LEDها استفاده شد که برای هر هشت سطر از یک UDN استفاده می کند. و از سه ULN ۲۸۰۳ استفاده می شود که این سه به یک ۴۰۲۸ وصل شده است که برای هر هشت ستون یک ULN استفاده می شود . صفحه نمایش آن از ۱۹۲ تا LED استفاده شده است که به صورت ماتریس ۸*۲۴ به هم وصل شده است.که از LED های سبز وقرمز و آبی استفاده شده است .

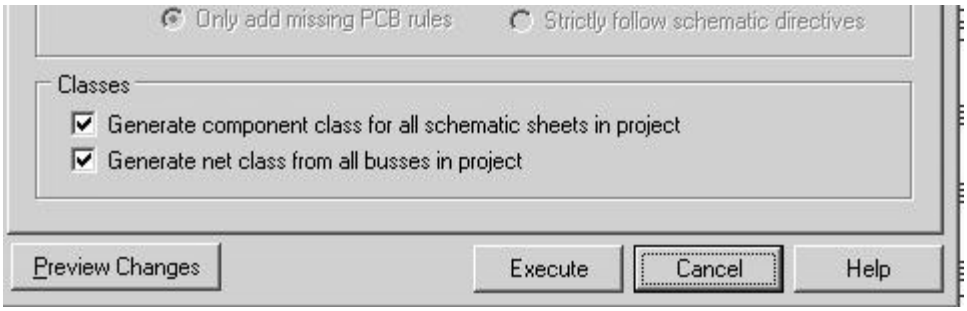
Pcb مدار:

در اولین مرحله بعد از طراحی کامل شماتیک باید آن را saveکرده وبعد از دادن تمام فوت پرینت ها ودر ضمن باید اسم تمام قطعات مثل هم با هم فرق داشته باشد

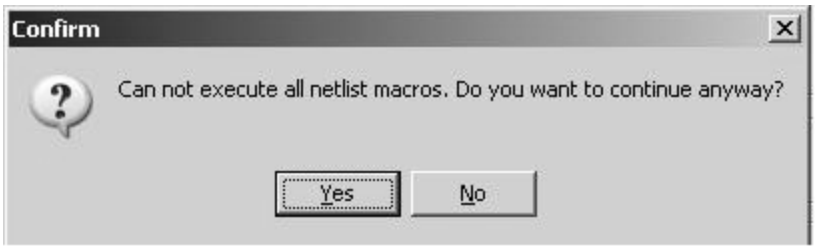
اول باید ازمنوی design گزینه ی update pcb را متابق شکل زده



در مرحله ی بعد باید **excute** را باید بزنیم



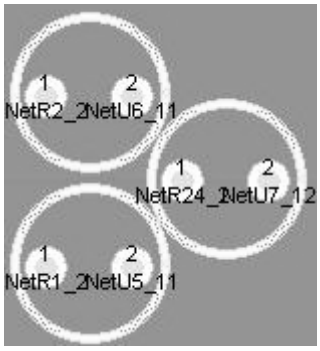
بعد اگر در دادن **footprint**ها یا نام قعات اشتباهی رخ داده شود این پنجره باز می شود



بعد به محیط **pcb** میرویم اما در اول همه ی قطعات جای خود نیستند وما باید آنها را جای خود بگذاریم

در اینجا نحوه ی چیدن **led** ها می توان دو نوع است که به صورت زیر است

شکل۲



شکل۱

که ما در مدار خد از شکل ۱ استفاده کردیم چون که در این مدل چیدن نور led ها باهم مخلوط شده وکسی که از فاصله ی دور نگاه می کند فکر میکند که ۱عدد led می باشد

ما در این مدا رتمام led ها رابه صورت شکل ۱ بر اساس شماره led ها می چینیم

اول کادر سبز را پاک می کنیم.

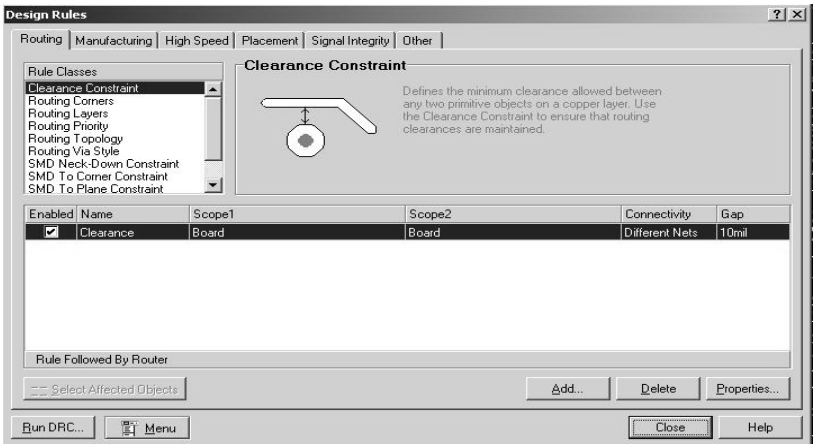
بعد با انتخاب سیم بنفش(keep out layer) اندازه ای که می خواهیم باشد را می کشیم

بعد گزینه ی tools گزینه ی auto placement گزینه ی auto placer را زده بعد گزینه cluster placer تیک اش را فعال کرده بعد گزینه ی Quick component plecment را میزنیم

بعد خود مدار بطور اتوماتیک قطعات را می چیند.اما چیدن این برنامه را ما استفاده نمی کنیم

بعد از چیدن قطعات به طور صحیح از منوی desing گزینه یrules را انتخاب کرده

بعد برای مثال از زبانه ی ruting قسمت Clearance برای فاصله ی pad ها از سیم ها می باشد

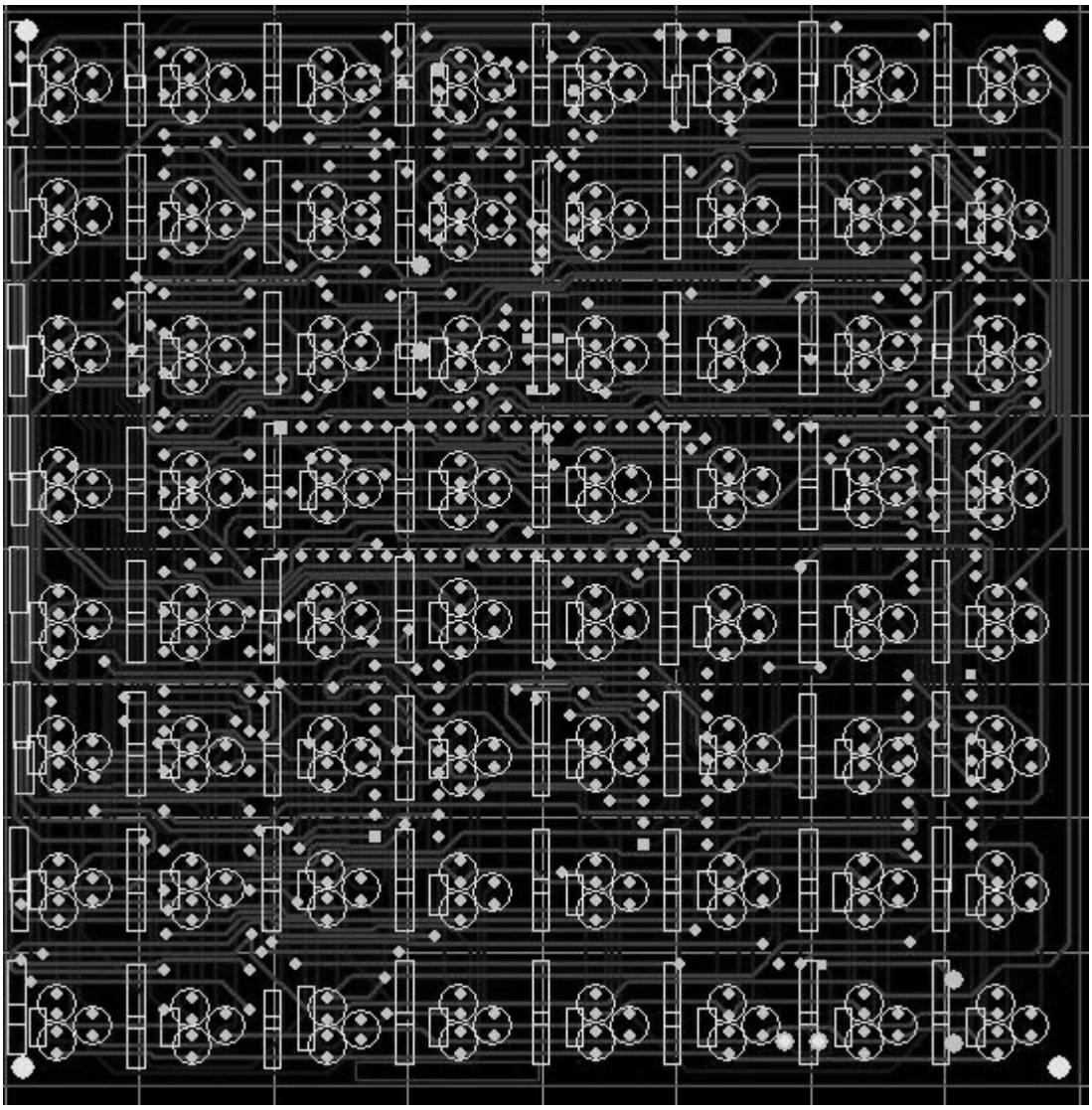


وزبانه ی **Width constraint** برای بیشترین و کمترین مقدار پهنای سیم کشی می باشد ما باید مقدار **pad** ها ومقدار سوراخ **pad** ها رانیز بدهیم.

بعد برای سیم کشی نهایی باید از منوی **auto route** گزینه ی **ALL** را میزنیم

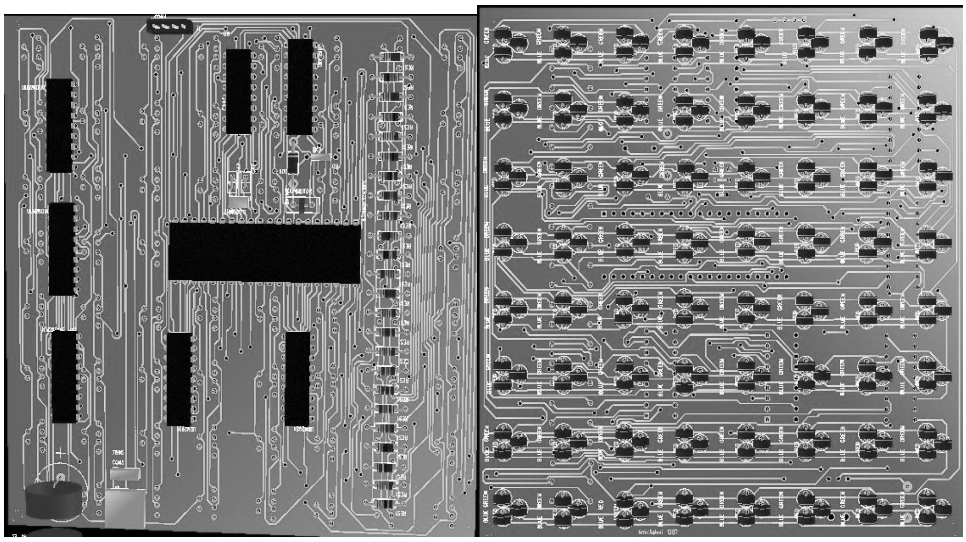
برای آن سیم کشی روی برد با زیر برد تشکیل یک خازن را ندهند یک بار با انتخاب سیم قرمز و یک بار با انتخاب سیم آبی این کار را انجام می دهیم .

مادر تابلو نمی توانیم تمام قطعات را در یک طرف برد بگزاریم بلکه باید تمام **led** ها در یک طرف وبقیه قطعات در طرف دیگرکه در این صورت مدار به شکل زیر در می آید



که شکل سه بعدی آن به صورت زیر می باشد

که چون قطعه ی led را خودم ساختم در اینجا به این صورت نشان داده است



برنامه ی slave وتوضیحات آن

```
*****************************************************/

This program was produced by the

CodeWizardAVR V۴.۲۲.۱ Professional

Automatic Program Generator

©Copyright ۱۹۹۸-۲۰۰۸ Pavel Haiduc, HP InfoTech s.r.l.

http://www.eittc.ac.ir.com

Project : rgb billboard

Version : ۱.۰۰

Date    : ۲۰۰۹/۰۱/۲۵

Author  : amir aslani

Company :amiry
```

Comments :

Chip type : ATmega۳۲

Program type : Application

Clock frequency : ۱۶.۰۰۰۰۰۰ MHz

Memory model : Small

External SRAM size : ۰

Data Stack size : ۵۱۲

```
/*****************************************************

#include <mega۳۲.h>

#include <spi.h>

#include <delay.h>

unsigned char R[۶۴]={۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,

۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,

۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,

۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,

۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,

۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,

۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰;}
```

اطلاعات رنگ قرمز

```
unsigned char G[۶۴]={۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,    // PORT   A

۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,

۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,
```

```
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰};
```

اطلاعات رنگ سبز

```
unsigned char B[۶۴]={۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,    // PORT C
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,
۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰,۰xff,۰x۰۰}
```

اطلاعات رنگ آبی

```
unsigned int I۰=۰,I۱,I۲,I۳,I۴,I۵,I۶,I۷ ;
 unsigned int s,counter;
 void SETRGB (void)
   {
    #asm ("WDR");

PORTA=۰X۰۰;
 PORTC=۰X۰۰;
```

```
PORTD=۰X۰۰;
}
```

زمانی که این زیر برنامه اجرا می شود تمام پورت ها صفر می شوند .

```
void SOOTOON (void)
{
   #asm ("WDR");

I۱=I۰+۰x۰۸;
I۲=I۰+۰x۱۰;
I۳=I۰+۰x۱۸;
I۴=I۰+۰x۲۰;
I۵=I۰+۰x۲۸;
I۶=I۰+۰x۳۰;
I۷=I۰+۰x۳۸;
}
```

زیر برنامه ای که اطلاعات ستون ها را درست می کند.

```
//Timer ۰ overflow interrupt service routine
interrupt [TIM۰_OVF] void timer۰_ovf_isr(void(
{
//Place your code here
    #asm ("WDR");

s++;
if(s>=۴)
}
```

```
  s=٠;
  SETRGB();
I٠++:
    #asm ("WDR");

SOOTOON;()
  SETRGB;()
    #asm ("WDR");

  PORTB=I٠;
  if(PORTB>٠X٠٧(
   {
    PORTB=٠X٠٠;
    I٠=٠;
    #asm ("WDR");

  }
  if(TCNT٠>=B[I٠])PORTC.٠=١;
  if(TCNT٠>=B[I١])PORTC.١=١;
    #asm ("WDR");
  if(TCNT٠>=B[I٢])PORTC.٢=١;
  if(TCNT٠>=B[I٣])PORTC.٣=١;
  if(TCNT٠>=B[I۴])PORTC.۴=١;
    #asm ("WDR");
  if(TCNT٠>=B[I۵])PORTC.۵=١;
  if(TCNT٠>=B[I۶])PORTC.۶=١;
  if(TCNT٠>=B[I٧])PORTC.٧=١;
```

```
  #asm ("WDR");

if(TCNT۰>=G[I۰])PORTA.۰=۱;
if(TCNT۰>=G[I۱])PORTA.۱=۱;
if(TCNT۰>=G[I۲])PORTA.۲=۱;
  #asm ("WDR");
if(TCNT۰>=G[I۳])PORTA.۳=۱;
if(TCNT۰>=G[I۴])PORTA.۴=۱;
if(TCNT۰>=G[I۵])PORTA.۵=۱;
  #asm ("WDR");
if(TCNT۰>=G[I۶])PORTA.۶=۱;
if(TCNT۰>=G[I۷])PORTA.۷=۱;
  #asm ("WDR");

if(TCNT۰>=R[I۰])PORTD.۰=۱;
if(TCNT۰>=R[I۱])PORTD.۱=۱;
if(TCNT۰>=R[I۲])PORTD.۲=۱;
  #asm ("WDR");
if(TCNT۰>=R[I۳])PORTD.۳=۱;
if(TCNT۰>=R[I۴])PORTD.۴=۱;
if(TCNT۰>=R[I۵])PORTD.۵=۱;
  #asm ("WDR");
if(TCNT۰>=R[I۶])PORTD.۶=۱;
if(TCNT۰>=R[I۷])PORTD.۷=۱;
 }
```

در زیز برنامه ی وقفه ی تایمر ۱ که هر ۱.۰۲ میلی ثانیه اتفاق می افتد ستون یک بار اضافه شده وسپس اطلاعات مطابق با آن ستون جا به جا شده وبعد از آن تایمر با آنها

مقایسه شده وخروجی ها را یک می کند در نتیجه هر led مطابق با اطلاعات خود یک pwm برایش ساخته می شود.

```
//SPI interrupt service routine

interrupt [SPI_STC] void spi_isr(void)

{

   #asm ("WDR");
unsigned char data;

data=SPDR;

//Place your code here



        if(counter<۶۴)    R[counter]=data;



    else   if(counter<۱۲۸)   G[counter-۶۴]=data ;

       #asm ("WDR");


    else   if(counter<۱۹۲)   B[counter-۱۲۸]=data;

        #asm ("WDR");
    counter;++

    if(data=='R')   counter=۰;


}
```

این زیر برنامه ی (اس پی آی) می باشد که زمانی اسلیو سلکت آن صفر شود منتظر ورود اطلاعات می شود و زمانی که اطلاعات آن برابر R شود شمارنده را صفر کرده و در اطلاعات بعدی(وقفه ی بعدی) اطلاعات در R[۰] ریخته شده وتا زمانی که شمارنده از ۶۴ کوچکتر باشد در متغییر R[] می ریزد واگر بین ۶۴ و ۱۲۸ باشد در

متغییر G[] می ریزد وبه همین صورت متغییر B[] نیز پر می شود تا زمانی که کد اسکی R بیاید و شمارنده را صفر کند .

```
//Declare your global variables here


void main(void(
{
//Declare your local variables here


//Input/Output Ports initialization
//Port A initialization
//Func۷=Out Func۶=Out Func۵=Out Func۴=Out Func۳=Out Func۲=Out Func۱=Out Func۰=Out
//State۷=۰ State۶=۰ State۵=۰ State۴=۰ State۳=۰ State۲=۰ State۱=۰ State۰=۰
PORTA=۰x۰۰;
DDRA=۰xFF;
   #asm ("WDR");

//Port B initialization
//Func۷=In Func۶=Out Func۵=In Func۴=In Func۳=Out Func۲=Out Func۱=Out Func۰=Out
//State۷=T State۶=۰ State۵=T State۴=T State۳=۰ State۲=۰ State۱=۰ State۰=۰
PORTB=۰x۰f;
DDRB=۰x۴F;
```

//Port C initialization

//Func۷=Out Func۶=Out Func۵=Out Func۴=Out Func۳=Out Func۲=Out Func۱=Out Func۰=Out

//State۷=۰ State۶=۰ State۵=۰ State۴=۰ State۳=۰ State۲=۰ State۱=۰ State۰=۰

PORTC=۰x۰۰;

DDRC=۰xFF;

//Port D initialization

//Func۷=Out Func۶=Out Func۵=Out Func۴=Out Func۳=Out Func۲=Out Func۱=Out Func۰=Out

//State۷=۰ State۶=۰ State۵=۰ State۴=۰ State۳=۰ State۲=۰ State۱=۰ State۰=۰

PORTD=۰x۰۰;

DDRD=۰xFF;

```
   #asm ("WDR");
```

//Timer/Counter ۰ initialization

//Clock source: System Clock

//Clock value: ۲۰۰۰.۰۰۰ kHz

//Mode: Normal top=FFh

//OC۰ output: Disconnected

TCCR۰=۰x۰۲;

TCNT۰=۰x۰۰;

OCR۰=۰x۰۰;

```
//Timer/Counter 1 initialization
//Clock source: System Clock
//Clock value: Timer 1 Stopped
//Mode: Normal top=FFFFh
//OC1A output: Discon.
//OC1B output: Discon.
//Noise Canceler: Off
//Input Capture on Falling Edge
//Timer 1 Overflow Interrupt: Off
//Input Capture Interrupt: Off
//Compare A Match Interrupt: Off
//Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
   #asm ("WDR");
```

```
//Timer/Counter ٢ initialization

//Clock source: System Clock

//Clock value: Timer ٢ Stopped

//Mode: Normal top=FFh

//OC٢ output: Disconnected

ASSR=٠x٠٠;

TCCR٢=٠x٠٠;

TCNT٢=٠x٠٠;

OCR٢=٠x٠٠;

    #asm ("WDR");



//External Interrupt(s) initialization

//INT٠: Off

//INT١: Off

//INT٢: Off

MCUCR=٠x٠٠;

MCUCSR=٠x٠٠;


    #asm ("WDR");


//Timer(s)/Counter(s) Interrupt(s) initialization

TIMSK=٠x٠١;
```

```
//Analog Comparator initialization

//Analog Comparator: Off

//Analog Comparator Input Capture by Timer/Counter ١: Off

ACSR=٠x٨٠;

SFIOR=٠x٠٠;


   #asm ("WDR");


//SPI initialization

//SPI Type: Slave

//SPI Clock Rate: ١٠٠٠.٠٠٠ kHz

//SPI Clock Phase: Cycle Half

//SPI Clock Polarity: Low

//SPI Data Order: MSB First

SPCR=٠xC١;

SPSR=٠x٠٠;

//Clear the SPI interrupt flag

#asm

  in   r٣٠,spsr

  in   r٣٠,spdr

#endasm

//Global enable interrupts
```

```
#asm("sei("

while (۱(


{

    #asm ("WDR");

 //    Place your code here

///     SOOTOON;()

     if(TCNT۰>=۰xfa)TCNT۰=۰xfe;
```

این دستور اگر تایمر بیشتر از fa باشد آن را fe می کند .

```
     delay_us(۵;(
```

تاخیر ۵میکرو ثانیه

```
    #asm ("WDR");

     if(TCNT۰>=B[I۰])PORTC.۰=۱;
     if(TCNT۰>=B[I۱])PORTC.۱=۱;
     if(TCNT۰>=B[I۲])PORTC.۲=۱;
     if(TCNT۰>=B[I۳])PORTC.۳=۱;
       #asm ("WDR");

     if(TCNT۰>=B[I۴])PORTC.۴=۱;
     if(TCNT۰>=B[I۵])PORTC.۵=۱;
     if(TCNT۰>=B[I۶])PORTC.۶=۱;
     if(TCNT۰>=B[I۷])PORTC.۷=۱;
        #asm ("WDR");
```

```
    if(TCNT۰>=G[I۰])PORTA.۰=۱;
    if(TCNT۰>=G[I۱])PORTA.۱=۱;
    if(TCNT۰>=G[I۲])PORTA.۲=۱;
    if(TCNT۰>=G[I۳])PORTA.۳=۱;
     #asm ("WDR");

    if(TCNT۰>=G[I۴])PORTA.۴=۱;
    if(TCNT۰>=G[I۵])PORTA.۵=۱;
    if(TCNT۰>=G[I۶])PORTA.۶=۱;
    if(TCNT۰>=G[I۷])PORTA.۷=۱;
       #asm ("WDR");

    if(TCNT۰>=R[I۰])PORTD.۰=۱;
    if(TCNT۰>=R[I۱])PORTD.۱=۱;
    if(TCNT۰>=R[I۲])PORTD.۲=۱;
    if(TCNT۰>=R[I۳])PORTD.۳=۱;
      #asm ("WDR");

    if(TCNT۰>=R[I۴])PORTD.۴=۱;
    if(TCNT۰>=R[I۵])PORTD.۵=۱;
    if(TCNT۰>=R[I۶])PORTD.۶=۱;
    if(TCNT۰>=R[I۷])PORTD.۷=۱;
   }
}
```

در این جا نیز برای ساخت pwm مورد نیاز تایمر با متغییر های R[] و G[] و B[] مقایسه شده و در نهایت در همه ی پین های آن حصئ ساخته می شود.

توضیح برنامه ی master

```
/*****************************************************
This program was produced by the
CodeWizardAVR V۴.۲۲.۱ Professional
Automatic Program Generator
© Copyright ۱۹۹۸-۲۰۰۸ Pavel Haiduc, HP InfoTech s.r.l.
http://www.eittc.ac.ir
Project :
Version :
Date    : ۲۰۰۸/۱۱/۱۱
Author  : amir aslani
Company : amiry
Comments:
Chip type           : ATmega۳۲
Program type        : Application
Clock frequency     : ۱۶.۰۰۰۰۰۰ MHz
Memory model        : Small
External SRAM size  : ۰
Data Stack size     : ۵۱۲
*****************************************************/
#include <mega۳۲.h>
// Standard Input/Output functions
#include <stdio.h>
// SPI functions
#include <spi.h>
```

```
unsigned char r[۲۵۶],g[۲۵۶],b[۲۵۶];
```

تعریف سه متغییر اطلاعات رنگ ها.

```
unsigned int a,bl,c,d,v;

   #asm ("WDR");
#define RXB۸ ۱

#define TXB۸ ۰

#define UPE ۲

#define OVR ۳

#define FE ۴

#define UDRE ۵

#define RXC ۷

#define FRAMING_ERROR (۱<<FE)

#define PARITY_ERROR (۱<<UPE)

#define DATA_OVERRUN (۱<<OVR)

#define DATA_REGISTER_EMPTY (۱<<UDRE)

#define RX_COMPLETE (۱<<RXC)

// USART Receiver buffer

#define RX_BUFFER_SIZE ۲۵۶

char rx_buffer[RX_BUFFER_SIZE];

#if RX_BUFFER_SIZE<۲۵۶

unsigned char rx_wr_index,rx_rd_index,rx_counter;

#else

unsigned int rx_wr_index,rx_rd_index,rx_counter;

#endif
```

```
// This flag is set on USART Receiver buffer overflow
bit rx_buffer_overflow;
    #asm ("WDR");
```

// USART Receiver interrupt service routine

زیر برنامه وقفه ی سریال که اگر به سریال یک کاراکتر بیاید در این وفقه وارد می شود.

```
interrupt [USART_RXC] void usart_rx_isr(void)
{
char status,data;
status=UCSRA;
data=UDR;
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==۰)
  {
    #asm ("WDR");
```

تا این قسمت برنامه صحت کاراکتر وارد شده را به ما می گوید ودر زیر آن نیز تایمر واچ داگ را نیز ریست کرده ایم.

```
   if(data=='R')    a =۱;
   if(data=='G')    a =۲;
   if(data=='B')    a =۳;
```

بر اساس ورودی اگر سه کد اسکی R,G,B باشد به a یک مقدار داده می شود.

```
#asm ("WDR");
     if(a==۱)
     {
      rx_buffer[rx_wr_index]=data;
      if (++rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=۰;
```

```
if (++rx_counter == RX_BUFFER_SIZE)
 {
 #asm ("WDR");
  rx_counter=۰;
  rx_buffer_overflow=۱;
  for(bl=۰;bl<۲۵۶;bl++)    r[bl]= rx_buffer[bl];
  a=۰;
 };
}
```

این if زمانی اجرا می شود که مقدار a برابر یک باشد و تا زمانی که بافر سریال پر نشود این if ادامه دارد ولی زمانی که بافر سریال پر می شود بافر سریال توسط یک for در متغییر r ریخته می شودو مقدار a نیز برابر صفر می شود در ضمن در این بین تایمر واچ داگ نیز ریست می شود.

```
if(a==۲)
{
#asm ("WDR");
 rx_buffer[rx_wr_index]=data;
 if (++rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=۰;
 if (++rx_counter == RX_BUFFER_SIZE)
  {
   #asm ("WDR");
  rx_counter=۰;
  rx_buffer_overflow=۱;
  for(c=۰;c<۲۵۶;c++)    g[c]= rx_buffer[c];
  a=۰;
 };
}
```

این if زمانی اجرا می شود که مقدار a برابر یک باشد و تا زمانی که بافر سریال پر نشود این if ادامه دارد ولی زمانی که بافر سریال پر می شود بافر سریال توسط یک for در متغییر g ریخته می شودو مقدار a نیز برابر صفر می شود در ضمن در این بین تایمر واچ داگ نیز ریست می شود.

```
if(a==۳)
{
 #asm ("WDR");
 rx_buffer[rx_wr_index]=data;
 if (++rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=۰;
 if (++rx_counter == RX_BUFFER_SIZE)
  {
   rx_counter=۰;
   rx_buffer_overflow=۱;
   for(d=۰;d<۲۵۶;d++)    b[d]= rx_buffer[d];
  #asm ("WDR");
   a=۰;
  };
}
```

این if زمانی اجرا می شود که مقدار a برابر یک باشد و تا زمانی که بافر سریال پر نشود این if ادامه دارد ولی زمانی که بافر سریال پر می شود بافر سریال توسط یک for در متغییر b ریخته می شودو مقدار a نیز برابر صفر می شود در ضمن در این بین تایمر واچ داگ نیز ریست می شود.

```
 #asm ("WDR");
if(data=='R')
{
rx_wr_index=۰;
rx_counter=۰;
PORTC=۰x۰۰;
```

```
spi('R');
PORTC=۰x۰f;
}
```

زمانی که در ورودی سریال کد اسکی R بیاید همین کد را برا ی همه ی slaveها ارسال می شود تا آنها اول اطلاعات را بفهمند.

```
#asm ("WDR");
 if(data=='G')
{
rx_wr_index=۰;
rx_counter=۰;
PORTC=۰x۰۰;
spi('G');
PORTC=۰x۰f;
}
```

زمانی که در ورودی سریال کد اسکی G بیاید همین کد را برا ی همه ی slaveها ارسال می شود تا آنها اول اطلاعات را بفهمند.

```
#asm ("WDR");
 if(data=='B')
{
#asm ("WDR");
rx_wr_index=۰;
rx_counter=۰;
PORTC=۰x۰۰;
spi('B');
PORTC=۰x۰f;
}
```

زمانی که در ورودی سریال کد اسکی B بیاید همین کد را برا ی همه ی slaveها ارسال می شود تا آنها اول اطلاعات را بفهمند.

```
  };
}
#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter==۰);
data=rx_buffer[rx_rd_index];
if (++rx_rd_index == RX_BUFFER_SIZE) rx_rd_index=۰;
#asm("cli")
#asm ("WDR");
--rx_counter;
#asm("sei")
return data;
}
#pragma used-
#endif
void network (unsigned char n,unsigned int na)
{
 if(na>=۱۱۲)
  {
```

```
  #asm ("WDR");
   na%=۱۶;
   if(na>=۸)   PORTC =۰x۰d;
   else       PORTC =۰x۰e;
   spi(n);
  }
 else
  {
  #asm ("WDR");
   na%=۱۶;
   if(na>=۸)   PORTC =۰x۰۷;
   else       PORTC =۰x۰b;
   spi(n);
  }
```

در زیر برنامه ی بالا برای شبکه سازی بکار می رود به طوری که این زیر برنامه دو ورودی دارد که یکی شماره متغییر (که بر اساس آن Slave انتخاب می شود)ودیگری نیز اطلاعات آن می باشد

```
}

// Declare your global variables here
void main(void)
{
// Declare your local variables here
// Input/Output Ports initialization
// Port A initialization
// Func۷=In Func۶=In Func۵=In Func۴=In Func۳=In Func۲=In Func۱=In
Func۰=In
```

// State۷=T State۶=T State۵=T State۴=T State۳=T State۲=T State۱=T State۰=T

PORTA=۰x۰۰;

DDRA=۰x۰۰;

// Port B initialization

// Func۷=Out Func۶=In Func۵=Out Func۴=Out Func۳=In Func۲=In Func۱=In Func۰=In

// State۷=۰ State۶=T State۵=۰ State۴=۰ State۳=T State۲=T State۱=T State۰=T

PORTB=۰x۰۰;

DDRB=۰xB۰;

// Port C initialization

// Func۷=In Func۶=In Func۵=In Func۴=In Func۳=Out Func۲=Out Func۱=Out Func۰=Out

// State۷=T State۶=T State۵=T State۴=T State۳=۱ State۲=۱ State۱=۱ State۰=۱

PORTC=۰x۰F;

DDRC=۰x۰F;

// Port D initialization

// Func۷=In Func۶=In Func۵=In Func۴=In Func۳=In Func۲=In Func۱=In Func۰=In

// State۷=T State۶=T State۵=T State۴=T State۳=T State۲=T State۱=T State۰=T

PORTD=۰x۰۰;

DDRD=۰x۰۰;

// Timer/Counter ۰ initialization

// Clock source: System Clock

// Clock value: Timer ۰ Stopped

// Mode: Normal top=FFh

```
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
```

```
// Timer/Counter ۲ initialization
// Clock source: System Clock
// Clock value: Timer ۲ Stopped
// Mode: Normal top=FFh
// OC۲ output: Disconnected
ASSR=۰x۰۰;
TCCR۲=۰x۰۰;
TCNT۲=۰x۰۰;
OCR۲=۰x۰۰;

// External Interrupt(s) initialization
// INT۰: Off
// INT۱: Off
// INT۲: Off
MCUCR=۰x۰۰;
MCUCSR=۰x۰۰;
// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=۰x۰۰;

// USART initialization
// Communication Parameters: ۸ Data, ۱ Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
```

```
// USART Baud Rate: ۱۹۲۰۰
UCSRA=۰x۰۰;
UCSRB=۰x۹۸;
UCSRC=۰x۸۶;
UBRRH=۰x۰۰;
UBRRL=۰x۳۳;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter ۱: Off
ACSR=۰x۸۰;
SFIOR=۰x۰۰;
// SPI initialization
// SPI Type: Master
// SPI Clock Rate: ۴۰۰۰.۰۰۰ kHz
// SPI Clock Phase: Cycle Half
// SPI Clock Polarity: Low
// SPI Data Order: MSB First
SPCR=۰x۵۰;
SPSR=۰x۰۰;
// Watchdog Timer initialization
// Watchdog Timer Prescaler: OSC/۱۶k
WDTCR=۰x۰۸;
// Global enable interrupts

#asm("sei")
```

```
while (۱)
    {
    for(v=۰;v<۲۵۶;v++)
    {
     #asm ("WDR");
    network(r[v],v);
    network(g[v],v);
    network(b[v],v);
```

در اینجا یک حلقه **for** می باشد که سه متغییر **r[],g[],b[]** توسط زیر برنامه ی **network** که در این زیر برنامه نیز از پروتکل **spi** استفاده شده است .متناسب با شماره‌ی آنها به **slave**های خود ارسال می شود.

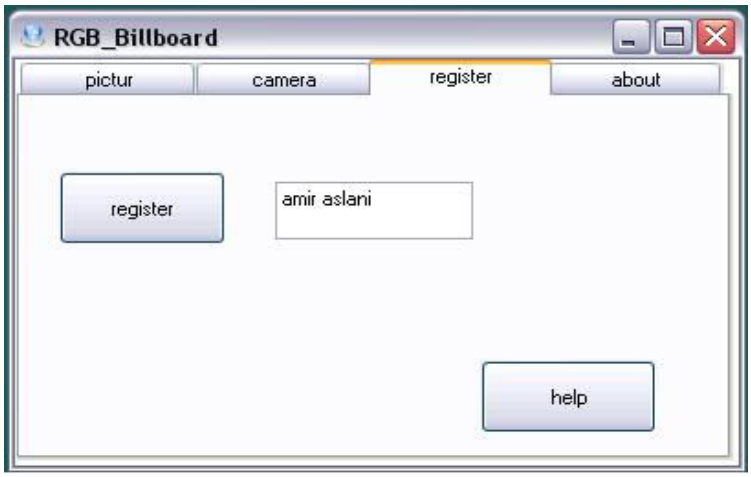
```
    }

    };
}
```

نرم افزار تابلو

این نرم افزار به گو نه ای طراحی شده است در یک تب آن که باید اول آن را رجیستر کنیم تا برنامه فعال شود تا بتوانیم از آن استفاده کنیم

اول باید در تب **rejister** و در آن پسورود **amir aslani** را وارد نموده تا برنامه مطابق شکل فعال شود.



بعد دکمه ی **register** بزنید.

بعد از این مرحله شما می توانید از تمام امکانات این برنامه استفاده کنید.

امکانات برنامه

در تب اول آن که **pictur** می باشد شما می توانید عکس های کوچک تر ۱۶*۱۶ را انتخاب کنید از طریق پرت سریال به تابلو ازسال کنید.مطابق شکل:

بعد با زدن دکمه **show** اطلاعات عکس که سه رنگ آبی و سبز وقرمز می باشد را از طریق پرت سریال ارسال می کند.

تب camera

در این تب به گونه ای است که بر نامه از صفحه ی کامپیوتر عکس می گیرد که اما فقط از وسط صفحه عکس می گیرد که در ورژن های بعدی آن را در اختیار مصرف کننده قرار خواهیم داد.

در این برنامه با زدن دکمه ی **camera** یک عکس از صفحه ی کامپیوتر گرفته و بعد از آن از طریق پورت سریال به تابلو ارسال می شود.مطابق شکل:

اما با زدن دکمه ی show film این برنامه از وسط صفحه نماشی فیلم می گیرد وآن را همزمان از طریق پورت سریال به تابلو می فرستد.

تب about

در این قسمت نیز نیزمطابق شکل نام تهیه کننده که خودم این برنامه را با یاری استاد انصار نوشتم آمده است.

درضمن یک نکته ی مهم که برای اجرا شدن این برنامه باید اول برنامه

Microsoft .NET Framework ۳.۵.exe را نصب کنید که به نظرم در ویندوز ویستا و ویندوز هفت به این بر نامه احتیاجی ندارد.در صورت اجرا نشدن این برنامه را اول نصب کنید.