گاهنامه تخصصی
مهندسی برق و کامپیوتر

شماره دوازدهم
تابستان 95

رگولار کردن الگوی فضایی مشترک . . . (BCI)

LCD کارکتری و راه اندازی آن

الگوریتم PSO

ساختار لامپ TWT

آموزش اینورتر دلتا



شماره مجوز وزارت فرهنگ و ارشاد اسلامی:۷۷۸۲۱

ISSN : 2322-3723

فهرست مطالب

درباره کهربا ۱

پیشگفتار ۲

آشنایی با مشاهیر ۳

iBoardbot ۵

LCD کارکتری و نکات مربوط به راه اندازی آن..... ۷

رگولار کردن الگوی فضایی مشترک . . . (BCI) ۱۸

الگوریتم PSO ۲۴

ساختار لامپ TWT ۳۷

آموزش اینورتر دلتا ۴۲

درباره کهربا

مجله الکترونیکی کهربا با همت و تلاش جمعی از دانشجویان و مهندسین برق بخصوص اعضای انجمن علمی پژوهشی نواندیشان در سال ۹۰ شروع به کار کرد. کهربا از سال ۹۱ با کسب شماره بین‌المللی استاندارد مجلات(ISSN: ۲۳۲۲-۳۷۲۳) و ثبت در سازمان اسناد و کتابخانه ملی ایران بعنوان فصلنامه‌ای رسمی و تخصصی در حوزه صنعت برق شناخته شده است. این نشریه تخصصی در سال ۹۵ موفق به اخذ پروانه انتشار از وزارت فرهنگ و ارشاد اسلامی با مجوز شماره ۷۷۸۲۱ گردید و در بانک اطلاعات نشریات کشور ثبت شد. امید است که با همت و همیاری دوستان بتوانیم قدمی هرچند کوچک جهت جلب رضایت شما عزیزان و بالابردن سطح دانش دوستان برداریم و روز به روز شاهد موفقیت و نیکبختی هریک از شما باشیم.

ایمیل(kahroba@noandishaan.com)بعنوان پل ارتباطی ما با شما دوستان آماده دریافت انتقادات , پیشنهادات و مقالات شما می باشد.

پیشگفتار

میاسای زآموختن یک زمان ز دانش میفکن دل اندر گمان
زمـانی میاسـای زآموختـن اگر جان همی خواهی افروختن
به آموختن گر ببنـدی میان ز دانـش روی بـر سپهـر روان
توانـا بود هـر که دانـا بود ز دانـش، دل پیـر، برنـا بـود

با عرض سلام و ارادت خدمت خوانندگان محترم کهربا

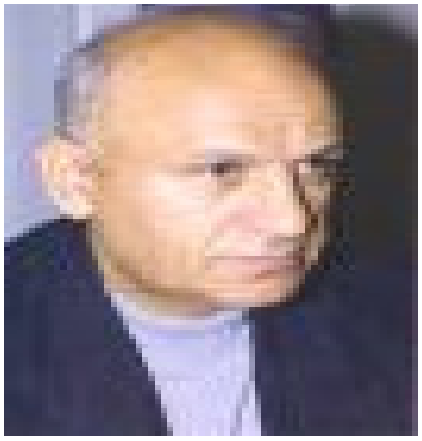
مجله ایی که پیش روی شماست، شماره دوازدهم کهربا و چهارمین شماره ایی است که بصورت ادغام برق و کامپیوتر است و در حالی منتشر می شود که مجله موفق به اخذ پروانه انتشار از وزارت فرهنگ و ارشاد اسلامی گردیده و در بانک اطلاعات نشریات کشور ثبت شده است. امید است مورد توجه تمامی همراهان مجله قرار گیرد.

از تلاش مهندسین برق و کامپیوتر انجمن علمی و پژوهشی نواندیشان که در منتشر شدن مجله همکاری نموده اند، کمال تشکر را داریم. هنوز راه درازی تا رسیدن به موفقیت های بزرگ وجود دارد که امیدواریم با همکاری همه عزیزان، به این امر دست یابیم..

با تشکر
سر دبیر مجله کهربا

آشنایی با مشاهیر برق

دکتر قدرت الله حیدری



دکتر قدرت الله حیدری از نام‌آوران و چهره‌های برجسته صنعت برق ایران و از بنیانگذاران کنفرانس‌های مهندسی برق ایران بوده و در زمینه خطوط انتقال برق و شبکه‌های توزیع نیروی برق دارای تالیفات و مقالات متعددی است.

این استاد برجسته برق قدرت ایران سال‌ها عضو اصلی انجمن مهندسین برق و الکترونیک ایران بوده و در چند دوره نیز به‌عنوان دبیر انجمن فعالیت داشته است. وی سابقه تدریس در دانشگاه‌های علم و صنعت؛ امیرکبیر، تهران، دانشگاه صنعت آب و برق (شهید عباسپور) و... را نیز دارد. کتاب طراحی خطوط انتقال برق ایشان سال‌ها به‌عنوان یکی از مراجع اصلی دانشگاهی در رشته‌های برق قدرت تدریس می‌شد.

دکتر قدرت الله حیدری در اول تیر ۱۳۲۸ هجری شمسی در روستای نفت چال لفور متولدشد. ایشان تحصیلات دبستانی خود را در دبستان گنج افروز شیرگاه و تحصیلات دبیرستانی خود را در دبیرستان‌های سپهر و رازی قایم‌شهر سپری نمود. ایشان در سال ۱۳۴۶ در رشته مهندسی برق دانشگاه صنعتی امیر کبیر (پلی تکنیک تهران) پذیرفته شد.

دکتر حیدری پس از فارغ التحصیلی از دانشگاه امیرکبیر کار خود را در صنعت برق کشور آغاز نمود. دکتر قدرت الله حیدری تحصیلات تکمیلی خود را در دانشگاه سنت پترزبورگ تکمیل و موفق به اخذ درجه دکترای مهندسی برق در گرایش انتقال انرژی شده است. ایشان منشا خدمات ارزشمندی در زمینه‌های علمی و پژوهشی و از برجستگان صنعت برق در داخل و خارج کشور بود.

درابتدای سال ۱۳۶۸ و در جریان برگزاری چهارمین کنفرانس بین‌المللی برق، فکر برگزاری کنفرانس مستقلی در خصوص شبکه‌های توزیع نیروی برق توسط دکتر قدرت اله حیدری مطرح و به مهندس محمد ملاکی مدیرعامل وقت شرکت توانیر پیشنهاد گردید که بعداز موافقت ایشان مقررگردید دبیرخانه کنفرانس بین‌المللی برق که در آن موقع با مسوولیت آقای دکتر قدرت اله حیدری اداره می‌گردید نسبت به اجرایی نمودن آن برنامه‌ریزی لازم را انجام دهند

ابتدای سال ۱۳۶۹این نتیجه حاصل شد که کنفرانسی تحت عنوان»کنفرانس شبکه‌های توزیع نیروی برق» در اردیبهشت ماه سال ۱۳۷۰ در ایران برگزار گردد که پس از مذاکرات انجام شده بین آقایان دکتر قدرت اله حیدری و مهندس حسین لوایی و دکتر فریبرز موجی مقرر، شهرستان بندر انزلی به عنوان محل برگزاری نخستین کنفرانس انتخاب گردید.

کنفرانس دوم تا پنجم نیز با یک برنامه منسجم در استان‌های اصفهان، فارس، هرمزگان و خراسان برگزار و مقرر شد ششمین کنفرانس در اردیبهشت ماه ۱۳۷۵ در شهرستان بابلسر استان مازندران برگزار گردد. اما در شرایطی که کلیه اقدامات لازم جهت اجرای این کنفرانس انجام شده بود، یک هفته قبل از برگزاری آن، اجرای این کنفرانس به آینده نا معلومی موکول شد.

از سال ۱۳۷۵ تا سال ۱۳۸۱ برگزاری این کنفرانس متوقف گردید، اما آقای دکتر قدرت اله حیدری پایه‌گذار این کنفرانس، موضوع برگزاری مجدد آن‌را در بهمن ماه سال ۱۳۷۹ در یکی از جلسات هیات مدیره انجمن مهندسین برق و الکترونیک ایران مطرح نمود و بر مبنای این پیشنهاد، مقرر گردید کنفرانس شبکه‌های توزیع نیرو با همان سبک گذشته و با مدیریت ایشان که دبیر و عضو هیات مدیره انجمن مهندسین برق و الکترونیک ایران نیز بوده‌اند، فعالیت خود را از سر گیرد.

آقای دکتر حیدری پس از انجام مطالعات و بررسی‌های لازم پیشنهاد خودرا در اردیبهشت ماه سال ۱۳۸۰ به هیات مدیره انجمن ارایه داده‌اند. در این پیشنهاد آقایان مهندس احمد شکوری و مهندس احمد فریدون درافشان رییس و دبیر انجمن مهندسین شاخه تهران به عنوان مسوولین کنفرانس انتخاب شده‌اند. خوشبختانه بعد از ۶ سال وقفه، هفتمین کنفرانس با حضور بیش از ۱۰۰۰ نفر در اردیبهشت ماه سال ۱۳۸۱ در سالن کنفرانس پژوهشگاه نیرو برگزار گردید و از آن سال به بعد این کنفرانس با یک برنامه منسجم هرساله در تهران و شهرستان‌های مختلف کشور برگزار می‌گردد.

سرانجام این شخصیت برجسته مهندسی برق کشور، در تاریخ ۱۹ دی ماه سال ۱۳۹۴ پس از یک سال بیماری در سن ۶۶ سالگی دار فانی را وداع گفت. روحش شاد و قرین آرامش باد.

گردآورنده: علی کمالی

کارشناس ارشد مهندسی برق قدرت

iBoardbot

iBoardbot رباتی است که با وصل شدن به اینترنت می‌تواند با ظرافتی عالی بنویسید و نقاشی کند. همچنین می‌تواند خیلی سریع و موثر پاک کند. اطلاعات خود را از هر نقطه ای از جهان که هستید به iBoardbot بفرستید. با توجه به اینکه قابلیت اتصال به چندین کاربر را دارد، می‌توانید با کودکانتان بازی کنید و با آن‌ها رقابت کنید، یا از آن به عنوان برد اعلان گروهی و یا به عنوان تابلوی فروشگاه‌تان استفاده کنید. لذا از آموختن و نصب iBoardbot لذت ببرید!

IBoardbot قابلیت بازتولید و تکثیر آنچه را که شما از راه دور می کشید با استفاده از برنامه وب خود از هر نقطه از جهان و از هر وسیله ای مانند لپ‌تاپ، گوشی هوشمند یا تبلت را دارد. برنامه وب قبل اشتراک‌گذاری است، لذا کاربر می‌تواند برنامه‌های ضروری خود را از طریق کنترل برد بیابد. شما قادر خواهید بود با انتخاب خود نقاشی کنید، بنویسید (حتی از گوشه‌ی دیگری از جهان!) و به معنی وضعیت iBoadbot دست پیدا کنید. این ربات طراحی می‌تواند در آن واحد توسط هر تعداد کاربری که می‌خواهید مورد استفاده قرار گیرد.

اما چگونه کار می‌کند؟

iBoardbot رباتی است که از برد Arduino (atmega۳۲u۴) با ماجول وای‌فای با قابلیت اتصال با سرویس Cloud ساخته شده است. هنگامی‌که شما متن یا طرحی را توسط انگشتانتان پیاده می‌کنید، اطلاعات به سرور (در cloud) فرستاده می‌شود سپس، پردازش شده و به داده‌هایی با سایز کوچک‌تر تبدیل شده و به iBoardbot فرستاده می‌شود تا پیام شما را تقریبا همزمان ترسیم کند.

در سرور برای هر رباتی یک وظیفه مصوف وجود دارد. iBoardbot این صف را اداره می‌کند، لذا نیازی نیست که شما منتظر فرستادن دستورات جدید باشید! این یک ربات طراحی دوبعدی با طراحی ساده و دقتی بالاست. با توجه به اینکه ساختاری مکانیکی دارد از موتور پله‌ای، یاتاقان و تسمه تایم استفاده می‌کند. این «روبات نمایشگر» یک پاک‌کن مجتمع در قسمت نوار نقاشی دارد و شما می‌توانید اتومات برد را پاک کنید و به طور پیوسته هر آنچه که می‌خواهید پیام یا گرافیک ارسال کنید!

حال بیایید نگاهی به دموی آن بیاندازیم:

Text mode: هر متنی که می‌خواهید بفرستید. iBoardbot آن را به سایز مورد نظر تنظیم می‌کند لذا منطبق با اندازه برد بوده و همچنین با فونت خودش آن را می‌نویسد. کافیست بنویسید و send را بزنید.

Drawing mode: آنچه را که شما برای آن تعیین کرده اید می‌تواند دقیق بکشد.

کاربرهای متفاوت با دستگاه های متفاوت از نقاط مختلف می‌توانند همزمان در نقاشی با شما همکاری کنند. مهم نیست نقاش کجاست؛ ظرافت و دقت نقاشی به گونه ای خواهد بود که گویا نقاش مداد را در دستانش دارد.

Graphic mode: یک گالری آنلاین وجود دارد و شما می‌توانید هر طرحی را که تاکنون اپلود شده را انتخاب کنید و یا حتی طرح خود را از طریق WEB APP آپلود کنید.

iBoardbot به صورت IFTTT نیز به خوبی کار می‌کند:

IFTTT سرویسی بر پایه وب است و به کاربر اجازه می‌دهد تا زنجیره ساده ای از عبارات شرطیی را، به آن «دستورالعمل» گفته می‌شود، که براساس تغییرات سایر سرویس‌های وب چون Gmail, Instagram و Twitter راه اندازی می‌شود، ایجاد کنند. IFTTT مخفف «If This Then That» است.

مثالی از «دستورالعمل» می‌تواند شامل فرستادن ایمیل باشد درصورتیکه کاربر IFTTT از برچسبهای مختلفی استفاده کند. یا درصورتیکه، ایمیلی را دریافت کند، موضوعی را به اسپریدشیت درایو Google ارسال کند. اینها صرفا دو مثال بودند. کاری که در اینجا انجام می‌دهیم این است: استفاده از اطلاعات IFTTT، ارسال از منابع متعدد، برای iBoardbot. چرا نباید عنوان و موضوع ایمیلی را که دریافت کرده اید را به iBoardbot ارسال نکنیم؟ یا آخرین پیامکی را که دریافت کرده‌اید؟ یا دمای اتاقتان؟ یا وضعیت آب و هوای فردا؟ یا..... . دستورالعمل‌های زیادی وجود دارند.

توجه: به محضی که دستورالعملی ایجاد شد قابلیت اشتراک‌گذاری با هر شخصی را دارد.

کد آن را از این قسمت می‌توانید بیابید:

https://github.com/jjrobots/iBoardbot

راهنمای نصب

http://jjrobots.com/iboardbot- :

/assembly-instructions

منبع

http://www.instructables.com/id/ :

IBoardbot-the-Internet-Controlled-

/Whiteboard-Robot

مترجم: مهسا سرائی

کارشناس ارشد مخابرات میدان

LCD کارکتری و نکات مربوط به راه اندازی آن

LCDها ابزاری برای نمایش اطلاعاتی هستند که شامل حروف و اعداد و همچنین برخی کاراکترهای گرافیکی می‌شود. بطور معمول در تجربیات اولیه در نمایش اطلاعات دیجیتال از نمایشگرهای هفت قسمتی (seven segment) استفاده می‌شود که این نمایشگرها فقط ارقام (۰ تا ۹) و بعضی حروف مثل A b C را بصورت نه چندان زیبا نمایش می‌دهند. اما با بکارگیری LCD اطلاعات را بصورت زیبا و کاملتر می‌توان نمایش داد. البته استفاده از LCD برای مدارات ساده توصیه نمی‌شود و عموما آنرا همرا با میکروکنترلر یا CPUها بکار می‌برند. LCD دارای یک کنترلر است که با فرستادن اطلاعات به آن این اطلاعات را در صفحه‌ای که عموما به چند سطر و ستون تقسیم شده نمایش می‌دهد.

مثلا برای نمایش حرف «M» کافیست کد اسکی این حرف را طبق یک پروتکل ساده به LCD ارسال کنیم. همچنین می‌توان دستوراتی از قبیل پاک کردن صفحه نمایش، جابجایی مکان نما، خاموش روشن کردن مکان نما و غیره را نیز به LCD ارسال کرد. LCDهای مختلفی در بازار موجودند مانند LCD ۲×۱۶، LCD ۴×۱۶، LCD ۲×۲۰، LCD ۴×۲۰، ال سی دی گرافیکی ۱۲۸ در ۶۴ پیکسل و ....

LCDها عموما دارای ۱۶ پایه هستند که ۸ خط آن مربوط به فرستادن یا خواندن داده ها یا دستورالعمل‌ها می‌باشد. پایه‌های دیگر خطوط کنترل و ولتاژهای تغذیه می‌باشند. لیست کامل خطها به‌قرار زیر است:



۱- Vss زمین

۲- Vcc ولتاژ ۵ ولت برای کنترلر

۳- Vee ولتاژ تنظیم درخشندگی(contrast)

۴- RS انتخابگر ثبات دستور / داده

۵- RW انتخابگر خواندن / نوشتن

۶- Enable فعال کننده

۷-۱۴ Bus ۸ خط گذرگاه داد یا دستور

۱۵- ولتاژ ۵ ولت برای لامپ پشت صفحه

۱۶- زمین برای لامپ پشت صفحه

Vee برای تنظیم درخشندگی کاراکترها بکار می‌رود که باید ولتاژی بین صفر و ۵ ولت به این پایه اعمال نمود. برای بیشترین درخشندگی این پایه را به زمین متصل کنید. وقتی RS=۰ رجیستر دستور انتخاب می‌شود و هنگامی‌که RS=۱ رجیستر داده انتخاب می‌گردد و پین R/W برای انتخاب خواندن یا نوشتن است، به طوری که اگر قصد خواندن رجیستری را داشته باشیم باید R/W=۱ و اگر تصمیم بر نوشتن مقداری روی رجیستر داریم باید R/W=۰ باشد.

مُدهای کاری LCD

مد ۸ بیتی:

در مد ۸ بیتی پین‌های شماره ۷ تا ۱۴ از ماژول LCD به ۸ پین ورودی/خروجی میکروکنترلر متصل می‌شوند. در نتیجه در این مد ما برای تبادل اطلاعات نیاز به ۸ پین داریم. مزیت این مد در این است که برنامه‌نویسی برای این مد آسان بوده و داده ها به سرعت آپدیت می‌شوند.

مد ۴ بیتی:

در مد ۴ بیتی پین‌های شماره ی ۱۱ تا ۱۴ از ماژول LCD به چهار پین ورودی/خروجی میکروکنترلر وصل می‌گردند. از این رو در این مد برای تبادل داده فقط به ۴ پین نیاز داریم. دلیل اصلی استفاده از مد ۴ بیتی استفاده از ۴ پایه‌ی میکروکنترلر به جای استفاده از ۸ پایه است.

برای نمایش حروف و کلمات بر روی LCD راه های مختلفی وجود دارد که استفاده از نرم افزار CodeVision به دلیل در اختیار داشتن توابع کتابخانه ای لازم از بقیه بسیار آسان‌تر می‌باشد. این توابع از بیشترLCDهای موجود در بازار، با تعداد ردیف و ستون‌های مختلف پشتیبانی می‌کند.

توابع سطح پایین:

void _lcd_ready();

زمانی که این تابع فراخوانی شود، میکروکنترلر منتظر آماده شدن LCD برای دریافت اطلاعات می‌شود.

void_lcd_write_data(unsigned char data);

از این تابع برای مقداردهی کردن ثبات درون LCD استفاده می‌شود. این ثبات برای کنترل تنظیمات LCD می‌باشد و به کمک این تابع تنظیمات LCD را تغییر می‌دهیم.

void lcd_write_byte(unsigned char addr, unsigned char data);

این تابع محتویات data را در آدرس مشخص شده در addr درون حافظه DDRAM (Data Display Ram) یا CGRAM (Character Generator RAM) می نویسد

جدول کد های فرمان LCD:

عملکرد فرمان	کدHEX
صفحه نمایش روشن و مکان نما روشن	0X0E
صفحه نمایش روشن و مکان نما در حالت چشمک زن	0X0F
جابجایی مکان نما به چپ	0X10
جابجایی مکان نما به راست	0X14
حرکت کل صفحه نمایش به چپ	0X18
سازمان دهی 8 بیتی و ماتریس 5×7	0X38
سازمان دهی 4 بیتی و ماتریس 5×7	0X28
حرکت کل صفحه نمایش به راست	0X10
مکان نما به آغاز خط دوم انتقال می‌یابد	0XC0

عملکرد فرمان	کدHEX
پاک کردن صفحه نمایش	0X01
بازگشت مکان نما به سطر و ستون HOME 0و0	0X02
انتقال مکان نما به چپ	0X04
جابجایی صفحه نمایش به راست	0X05
انتقال مکان نما به راست	0X06
جابجایی صفحه نمایش به چپ	0X07
صفحه نمایش و مکان نما خاموش	0X08
صفحه نمایش خاموش و مکان نما روشن	0X0A
صفحه نمایش روشن و مکان نما خاموش	0X0C

توابع سطح بالا:

void char lcd_init(unsigned char lcd_columns)

این تابع تعداد ستون های LCD را به عنوان آرگومان ورودی دریافت می‌کند و مکان نمای چاپ را به خانه (۰،۰) می‌برد و همچنین مکان نمای چاپگر را غیر فعال می‌کند.در صورتی که میکروکنترلر ماژول LCD را تشخیص دهد این تابع مقدار ۱ را بر می‌گرداند در غیر این صورت مقدار صفر را بر می‌گرداند!

Void lcd_gotoxy(unsigned char x, unsigned char y);

مکان نمای چاپ را به (X,Y) مورد نظر می‌برد. بعد از فراخوانی این تابع کاراکترها از این نقطه شروع به چاپ شدن می‌کنند! X: ستون و Y: ردیف

void lcd_write_byte(unsigned char addr, unsigned char data);

بوسیله این تابع می‌توانیم یک متغیر Char را طوری مقداردهی کنیم که یک کاراکتر دلخواه شود، سپس بوسیله این تابع می‌توانیم آن را نمایش دهیم!

Unsigned char lcd_read_byte(unsigned char addr);

این تابع مشابه تابع قبل عمل می‌کند با این تفاوت که یک کاراکتر را از LCD می‌خواند و یک متغیر که شکل آن را دربر دارد را برمی‌گرداند.

void lcd_clear();

به وسیله این تابع صفحه نمایش پاک می‌شود!

void lcd_putchar(char c);

کاراکتر c را در موقعیت فعلی نمایش می‌دهد.

*void lcd_puts(char *str);*

رشته str را که آرایه ای از کارکترها می‌باشد، را در موقعیت فعلی نمایش می‌دهد. توجه کنید که وقتی از این تابع استفاده می‌کنیم که رشته str درون SRAM قرار داشته باشد.

*void lcd_putsf(char flash *str);*

رشته str را که آرایه ای از کارکترها می‌باشد، را در موقعیت فعلی نمایش می‌دهد. توجه کنید که وقتی از این تابع استفاده می‌کنیم که رشته str درون Flash قرار داشته باشد.

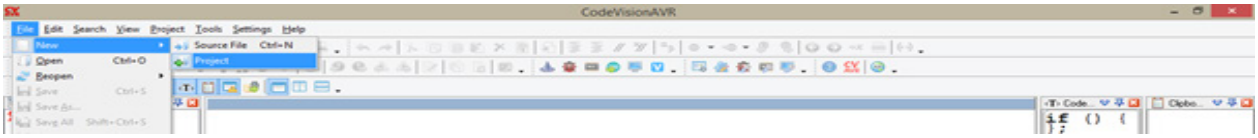
*void lcd_putse(char eeprom *str)*

رشته str که آرایه ای از کارکترهای درون eeprom می‌باشد را در موقعیت فعلی نمایش می‌دهد.

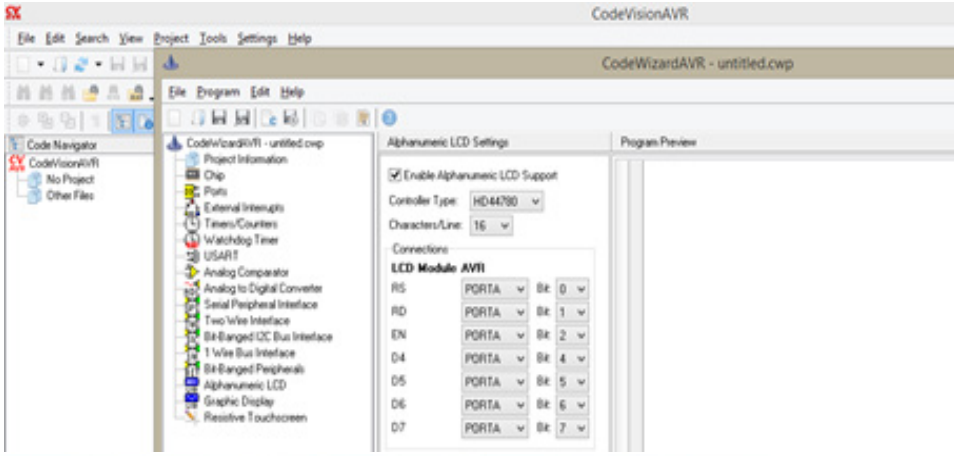
* دستورات بالا در نسخه های قدیمی‌نرم افزار در کتابخانه lcd.h و در نسخه ۲ به بعد در کتابخانه alcd.h قرار دارند .

راه اندازی در محیط کامپایلر کدویژن

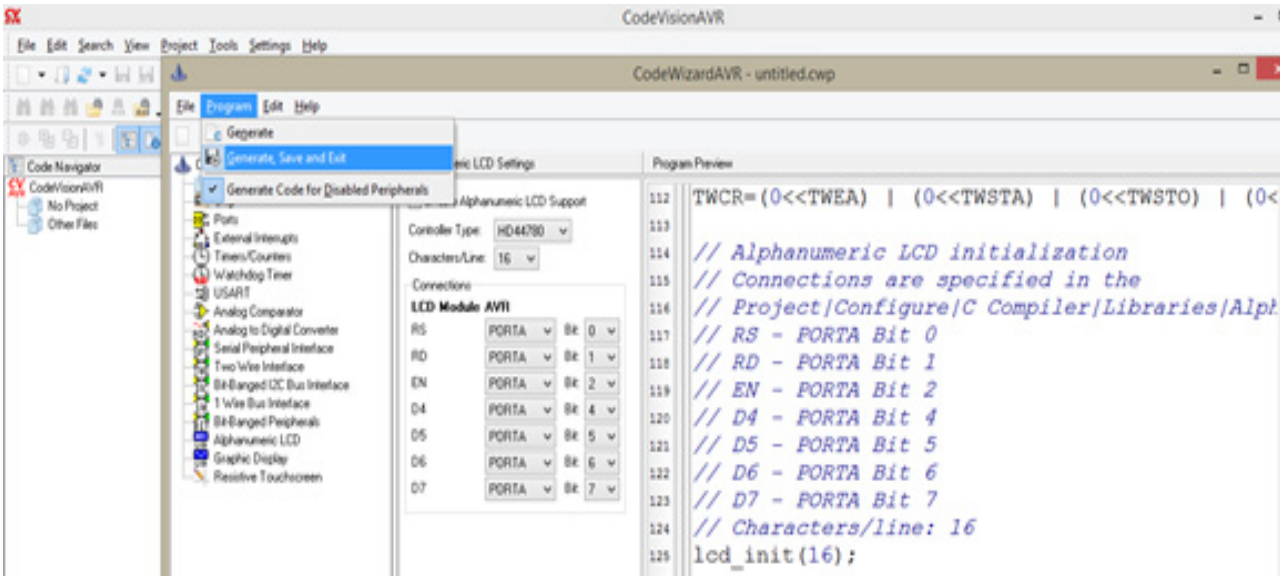
ابتدا یک پروژه ایجاد می‌کنیم و سپس وارد بخش CodeWizard می‌شویم.



پس از انتخاب نوع پردازنده وارد قسمت Alphanumeric LCD شده و تیک آن را می‌زنیم تا فعال گردد. حال باید نوع تراشه کنترلر و تعداد ستون های lcd مورد نظر و پورتی که lcd به ان متصل می‌شود را خود را انتخاب کنیم



در نهایت از منوی **program** گزینه **generate save and exit** را زده نامی‌برای پروژه انتخاب می‌کنیم و **ok** می‌کنیم .



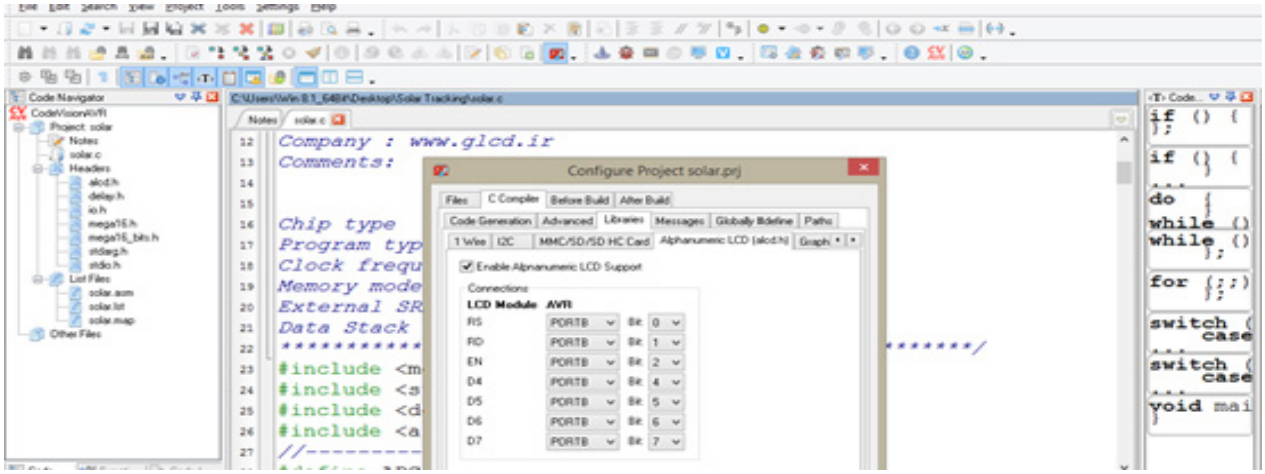
حال کافی ایست بسته به نیاز برنامه مورد نظر را بنویسیم .

اگر نیاز به تغییر پورت ای که **lcd** به آن متصل است بود در هنگام استفاده از کتابخانه **lcd.h** کافیست در بخش کد زیر آدرس پورت را تغییر دهیم .

```
// Alphanumeric LCD Module Functions
#asm
  .equ __lcd _port=0x12 ;//  PORTD
#endasm #include
```

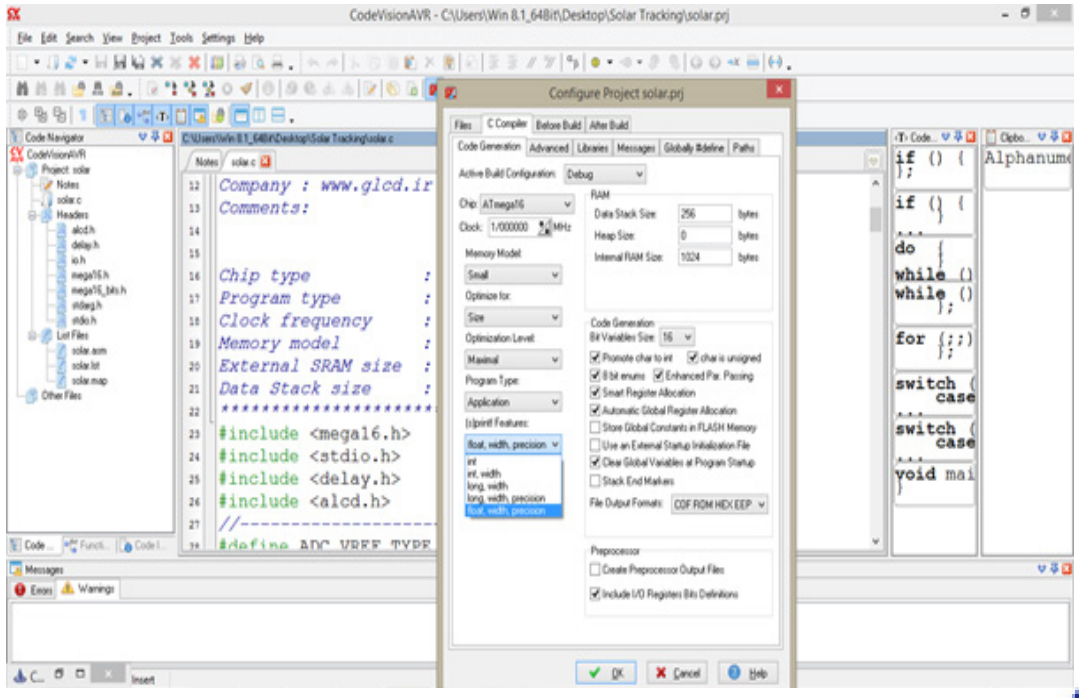
نام پورت	آدرس پورت
PORTA	ox1b
PORTB	ox18
PORTC	ox15
PORTD	ox12

در هنگام استفاده از کتابخانه **alcd.h** نیزکافیست از بخش تنظیمات پروژه وارد بخش کتابخانه‌ها شده و تنظیمات مورد نظر خود را دوباره وارد کنید .



* ممکن است بعد از اتصال LCD به میکروکنترلر، با اینکه همه موارد به درستی رعایت شده است، LCDکار نکند!!! در نظر داشته باشید که بعضی از مدل‌های میکروکنترلرهای AVR دارای قابلیتی به نام JTAG هستند. پایه‌های TMS،TDO،TDI و TCK مربوط به JTAG هستند. مثلا برای مدل‌های ATMEGA۱۶ و ATMEGA۳۲ این پایه‌ها روی پورت C قرار دارند. برای اینکه بتوانید از این پایه‌ها به عنوان ورودی و خروجی استفاده کنید باید قابلیت JTAG را غیرفعال کنید. برای کار باید فیوزبیت JTAGEN را از حالت برنامه‌ریزی شده خارج کرد. برای این کار باید عدد یک را در آن نوشت.

* برای اینکه در شبیه‌سازی برنامه خود در نرم افزار پروتئوس دچار مشکل نشوید، از بخش تنظیمات پروژه قسمت sprint features را در روی حالت float قرار دهید.



نمایش متن ثابت روی LCD

```
lcd_clear() ;
lcd_gotoxy(0,0) ;
lcd_putf(“hello”);
delay_ms(200);
```

نمایش مقدار یک متغیر روی LCD

برای این منظور باید ابتدا متغیر را توسط دستوراتی مانند **sprintf** درون یک رشته قرار داد(**sprintf** در کتابخانه **stdio.h** قرار دارد). به عنوان مثال در برنامه زیر مقدار ولتاژ ورودی پایه ۰**adc** ابتدا خوانده شده و در متغیر **x** قرار گرفته سپس این متغیر به صورت یک رشته درون **lcd_buffer** قرار گفته و نهایتا نمایش داده می‌شود. علت وجود تاخیر این است که متن نوشته شده به صورت ثابت روی **lcd** قرار بگیرد و پرش نداشته باشد .

```
lcd_clear();
lcd_gotoxy(0,0);
x= read_adc(0);
sprintf(lcd_buffer,»Uadc= %4umV»,x);
lcd_puts(lcd_buffer);
delay_ms(100);
```

عدد و حرف بعد از علامت % بیانگر نوع و تعداد ارقام نمایش متغیر در خروجی است .

مثلا در دستور

```
sprintf(lcd_buff,"humidity=
%%% 4.2 f",humidity);
```

عدد اعشاری **humidity** به صورت متن در خواهد آمد و در قالب یک عدد ۴ رقمی‌شامل ۲ رقم صحیح و ۲ رقم اعشار که کنار آن علامت ٪ قرار دارد داخل متغیر **lcd_buff** (آرایه کارکتری) ذخیره خواهد شد.

نوع اطلاعات ارسالی	کاراکتر
یک تک کاراکتر	% c
عدد صحیح علامت دار در مبنای 10	% d
عدد صحیح علامت دار در مبنای 10	% i
نمایش عدد ممیز شناور به صورت علمی	% e
نمایش عدد ممیز شناور به صورت علمی	% E
عدد اعشاری	% f
عبارت رشته ای واقع در حافظهSRAM	% s
عدد صحیح بدون علامت در مبنای 10	% u
به فرم هگزا دسیمال با حروف بزرگ	% X
به فرم هگزا دسیمال با حروف کوچک	% x
عبارت رشته ای واقع در حافظهFLASH	% P
نمایش علامت ٪	%%

حال اگر بخواهیم به جای sprintf از دستور itoa یا ftoa موجود درکتابخانه stdlib.h استفاده کنیم برنامه به صورت زیر می‌شود. دستور itoa برای نمایش اعداد صحیح می‌باشد مثلا

```
Int x;
lcd_putsf("Uadc=");
x=read_adc(0);
itoa(x, lcd_buffer);
lcd_puts(lcd_buffer);
delay_ms(100);
```

اما اگر متغیر اعشاری بود از دستور باید استفاده می‌شد. فرمت این دستور به صوت زیر است

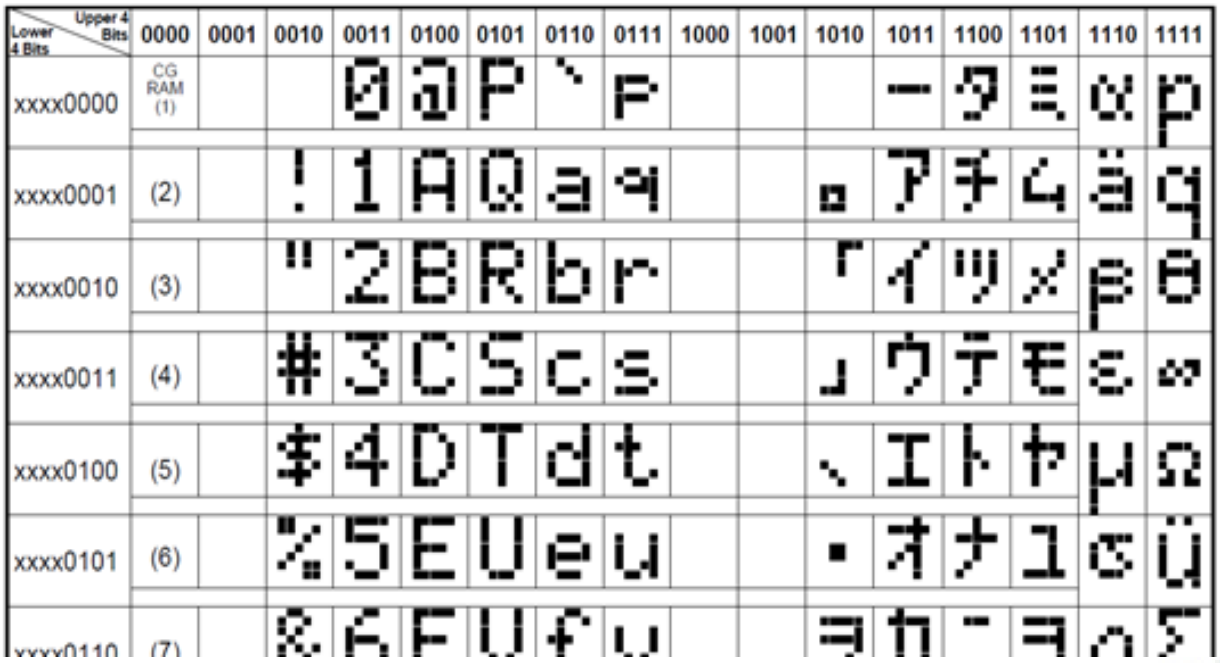
```
void ftoa(float fNumber, unsigned char
(decimals, char *str);
```

این تابع عدد اعشاری fNumber را در رشته str قرار می‌دهد. تعداد ارقامی‌که توسط این عدد اعشاری قرار است، در رشته قرار داده شود توسط متغیر decimals مشخص می‌شود مثلا

```
float x;
lcd_putsf("Uadc=");
x=read_adc(0);
ftoa(x,4, lcd_buffer);
lcd_puts(lcd_buffer);
delay_ms(100);
```

درج کارکترهای ویژه

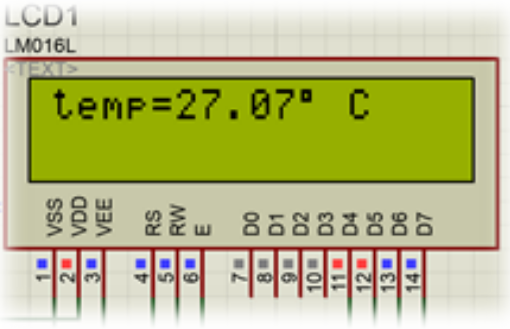
در دیتاشیت ای‌سی‌های کنترلر LCD جدولی از حروف و کارکترهای قابل نمایش توسط آن LCD وجود دارد .



به عنوان مثال فرض کنید بخواهیم مقدار دما را روی LCD نمایش دهیم (مقدار دما در متغیری مانند T محاسبه شده). با توجه به جدول کدباینری علامت °درجه ۱۱۰۱۱۱۱۱ می‌باشد و معادل هگز آن DF می‌باشد .برای نمایش این کارکترکنار دما به صورت زیر عمل می‌کنیم .

```
lcd_clear();
lcd_gotoxy(0,0);
sprintf(str,»temp= %3.2f\xdf C»,t);
lcd_puts(str);
```

خروجی برنامه بالا به صورت زیر می‌شود



استفاده از کدهای دستوری

فرض کنیم بخواهیم متن نوشته شده روی lcd حرکت کند . ساده ترین راه استفاده از برنامه‌ای مشابه برنامه زیر است

```
while (1)
      {
       for (j=0;j<7;j++)
       {
         lcd_clear();
         lcd_gotoxy(j,0);
         lcd_putsf("HELLO");
         delay_ms(500);
       };
       for (j=7;j>=0;j--)
       {
         lcd_clear();
         lcd_gotoxy(j,0);
         lcd_putsf("HELLO");
         delay_ms(500);
       };

      }
}
```

اما اگر بخواهیم حرکت متن را توسط کدهای دستوری lcd انجام دهیم به صورت زیر عمل می‌کنیم .

```
while (1)
      {
        // _lcd_write_data(0x0f);//فعال کردن چراغ چشمک زن
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Hello");
        ////////////text shift on lcd/////////////
         for(i=0;i<=160;i++)
          {
            delay_ms(180);
           _lcd_write_data(0x1c); // کل صفحه نمایش به چپ انتقال داده شود
          }
      }
```

یا مثلا اگر بخواهیم **blink** و **cursor** غیرفعال کنیم از دستور زیر استفاده می‌کنیم .

(در حالت استفاده از **blink** تماتی ۵ x ۸ (۵ ستون) پیکسل با هم روشن و خاموش می‌شوند)

```
_lcd_ready();
_lcd_write_data(0x0c);
```

ساخت کاراکتر و حروف دلخواه و نمایش آنها روی LCD

برای نمایش یک کاراکتر در هر آرایه باید ۷ عدد هشت بیتی را که مقادیر هفت ردیف آن آرایه هستند، به ال‌سی‌دی داد. برای این کار ابتدا نرم‌افزار **LCD Codevision AVR** را نصب می‌کنیم .و درون آن کاراکتر دلخواه خود را طراحی می‌کنیم و کد تولید شده را کپی می‌کنیم. محیط این برنامه به صورت زیر است .



به عنوان مثال کارکترهای خروجی کلمه سلام به صورت زیر می‌شود .

```
flash unsigned char char0[8] = { 0x0, 0x0, 0x15, 0x15, 0x1F, 0x0, 0x0,0x0 }; //ی
flash unsigned char char1[8] = { 0xA, 0xA, 0xA, 0xA, 0x1F, 0x0, 0x0,0x0 };//لا
flash unsigned char char2[8] = { 0x0, 0x0, 0x0, 0x0, 0xF, 0xB, 0x8,0x8 } ;//م
```

حال از تابع زیر برای نمایش این کارکترها استفاده می‌کنیم .

```
unsigned char i,a;
void define_char(unsigned char flash *pc,unsigned char char_code)
}
  a=(char_code<<0|(3x40;
  for(i=0;i<8;i++) lcd_write_byte(a++,*pc++);
};
```

برای فراخوانی کارکترها توسط این تابع به صورت زیر عمل می‌کنیم

```
define_char(char0,0);
define_char(char1,1);
define_char(char2,2);
```

برای نمایش این کارکترها نیز از دستور **lcd_putchar** استفاده می‌کنیم .

```
lcd_clear; ()
lcd_gotoxy(0,0);
lcd_putchar (2);
lcd_gotoxy(1,0);
lcd_putchar (1);
lcd_gotoxy(2,0);
lcd_putchar(0);
delay_ms(50);
```

امین شیخ نجدی

کارشناس ارشد مهندسی برق الکترونیک

رگولار کردن الگوی فضایی مشترک جهت بهبود طراحی سیستم‌های واسط مغز و رایانه(BCI)

خلاصه

یکی از عمده‌ترین روش‌های استخراج ویژگی در سیستم‌های واسط مغز و رایانه (BCI) مبتنی بر ثبت الکتروانسفالوگرام(EEG)، استفاده از الگوی فضایی مشترک (CSP) می‌باشد. با وجود کاربرد گسترده این روش از آن جا که CSP بسیار حساس به نویز است، در این مقاله از روش های رگولار شده و بهبود یافته الگوی فضایی مشترک به عنوان راهکاری جهت استخراج دقیق‌تر ویژگی‌های موردنظر استفاده شده است. در نهایت عملکرد ۴ الگوریتم RCSP را روی مجموعه داده‌های چهارمین دوره مسابقات جهانی BCI مقایسه می‌کنیم. نتایج نشان می‌دهد که الگوی فضایی منظم شده تیخونوف دارای کارائی بهتری نسبت به دیگر روش‌های الگوی فضایی منظم شده هستند.

کلمات کلیدی: استخراج ویژگی، الگوی فضایی مشترک، سیستم واسط مغز و رایانه، الگوی فضایی منظم شده

۱. مقدمه

بیماری‌های مختلفی می‌توانند باعث صدمه دیدن سیستم عصبی عضلانی، که مغز از طریق آن قادر به ایجاد ارتباط و اعمال کنترل به محیط خارج است، شوند. در شرایط حاد بیماری، فرد مبتلا ممکن است تمام حرکات ارادی خود از جمله حرکت چشم و تنفس را نیز از دست بدهد. به این گونه بیماران، اصطلاحا Locked-in گفته می‌شود.

پیشرفت‌های اخیر در سخت افزار و پردازش سیگنال استفاده از سیگنال های EEG را برای ارتباط انسان و رایانه ممکن ساخته است. سیستم واسط مغز و رایانه(BCI)، یک سیستم ارتباطی برای انتقال پیام از یک شخص به دنیای بیرون است که از طریق مسیرهای خروجی معمولی مغز، یعنی عصب‌های محیطی و ماهیچه‌ها صورت نمی‌گیرد. در سیستم های BCI ورودی همان، سیگنال‌های الکتریکی ثبت شده از داخل مغز(روش تهاجمی) و یا از روی پوست سر (روش غیرتهاجمی) می‌باشد. این سیگنال‌ها می‌توانند بر اساس باندهای فرکانسی طبقه‌بندی شوند. ریتم‌های مختلف مغز، دلتا، تتا، آلفا، بتا و گاما هستند. علاوه بر آلفا یک نوع موج دیگر در همان محدوده به نام امواج میو وجود دارد. و در هنگامی که دست‌ها و بازوها بیکار هستند زیاد می‌شوند

ریتم	محدوده فرکانسی (هرتز)	دامنه سیگنال (میلی ولت)	حالت ذهنی
دلتا	کمتر از 4	20-200	خواب عمیق
تتا	4-8	بیشتر از 20	خواب آلودگی در بزرگسالان
آلفا	8-12	30-50	استراحت آگاهانه
بتا	12-30	5-30	تفکر فعال
گاما	بیشتر از 30	کمتر از 5	مکانیستم آگاهی

جدول: ریتم های مختلف مغز

این ریتم ها همراه با محدوده فرکانسی خود، دامنه و حالت مربوطه از ذهن در جدول بالا مایش داده شده است.

دو نوع از نوسانات مهم در رسانه‌های واسط مغز و رایانه (BCI) عبارتند از: ریتم میو و بتا که هر دو در ناحیه حسی-حرکتی مغز ایجاد می‌شوند. تصور حرکت می‌تواند سبب کاهش دامنه (ERD) و یا افزایش دامنه (ERS) در دو باند فرکانسی میو و بتا شود. در حین تصور حرکت ERD در زمان شروع تصور در باندهای فرکانسی میو و بتا در نیمکره مخالف مغز ایجاد می‌شود در حالی‌که ERS در زمان پایان تصور در باند فرکانسی بتا ایجاد می‌شود. تاکنون روش‌های مختلفی برای افزایش کارائی سامانه‌های واسط مغز و رایانه ارائه شده است. یکی از روش‌های معمول استفاده از الگوی فضایی مشترک(CSP) است. این الگوهای فضایی از دو مجموعه آزمون‌های EEG مربوط به دو کلاس به‌دست آمده به طوری که تفاوت این دو مجموعه را زیاد کند.

در این مقاله ما از یک فیلتر اصلاح شده (RCSP) برای بهبود استخراج ویژگی از سیگنال‌های مغزی در حین تصور حرکت استفاده می‌کنیم. سپس از فیلتر TRCSP برای بهبود بیشتر استفاده کرده و خواهیم دید الگوی فضایی مشترک منظم شده تیخونوف کارآیی بهتری دارد.

۲. آزمایش و پیش پردازش

در این تحقیق، مجموعه داده 2a از چهارمین دوره مسابقات BCI استفاده شده است. این مجموعه شامل ۲۲ کانال از سیگنال‌های EEG مربوط به ۹ فرد با نرخ نمونه‌برداری Hz ۲۵۰ می‌باشد. سیگنال‌های ثبت شده از فیلتر میان گذر در محدوده Hz ۱۰۰-۰٫۵ با فرکانس قطع Hz ۵۰ عبور داده شده‌اند. در مقابل فرد مورد مطالعه صفحه نمایشی وجود داشته است. در شروع آزمایش در t=۰ یک علامت + بر روی صفحه مشکی ظاهر می‌شود و صدای هشدار کوتاهی پخش می‌گردد.

بعد از ۲ ثانیه یکی از علامت های چپ، راست، بالا و پایین مطابق با یکی از چهار حالت تصور حرکت (تصور حرکت دست راست، دست چپ، پا یا زبان) بر روی صفحه نمایش ظاهر می‌شود. این علامت به مدت ۱,۲۵ ثانیه بر روی صفحه نمایش باقی می‌ماند. فرد مورد مطالعه تا زمان t =۶ ثانیه فعالیت خواسته شده را تصور می کند. بعد از یک زمان استراحت کوتاه همین روند کار تکرار می‌شود. در اینجا تنها از داده های مربوط به تصور دست راست و چپ استفاده شده است. داده‌های EEG برای هر فرد شامل ۷۲ داده آموزش و ۷۲ داده تست در هر کلاس بوده و داده های تست در روز دیگری ثبت شده‌اند. در این تحقیق از محدوده زمانی ۰,۵ تا ۲,۵ ثانیه بعد از ظاهر شدن علامت بر روی صفحه نمایش برای استخراج ویژگی استفاده شده است(مانند برنده چهارمین مسابقه BCI در همین مجموعه داده).

سپس برای حذف نویز از یک فیلتر میان‌گذر با باند عبور ۸ تا ۳۰ هرتز روی سیگنال استفاده کرده‌ایم. این بازه دو باند فرکانسی میو و بتا را که جز باندهای مهم در سیستم های BCI به شمار میروند را در بر داشته و نسبت به بازه‌های با پهنای باند باریک‌تر عملکرد بهتری در جداسازی الگوهای تصورات حرکتی از خود نشان داده و دقت طبقه‌بندی را افزایش می‌دهند.

روش‌ها

الگوی فضایی مشترک (CSP)

این فیلترها که بیشتر برای بازشناسی الگو طراحی شده‌اند در سال ۱۹۹۵ توسط Koles مطرح شد و نخستین بار برای تجزیه مؤلفه های نورولوژیک در کاربردی کلینیکی از الکتروانسفالوگرافی استفاده شد. روش الگوهای فضایی مشترک بر اساس قطری سازی همزمان دو ماتریس متقارن حقیقی (که توسط Fukunaga ارائه شدند) کار می‌کند. قطری سازی همزمان اجازه تجزیه سیگنال EEG خام به دو الگوی مجزا از دو کلاس را می‌دهد که در این دو کلاس به صورت همزمان، واریانس یکی از کلاس‌ها بیشینه و واریانس دیگری کمینه است. در این صورت اگر تنها از فیلترهای با قابلیت تفکیک بیشتر استفاده شود، ابعاد داده هم کاهش می‌یابد و عملکرد طبقه بندی کننده نیز بهتر می‌شود[۵].

یکی از روش‌های استخراج ویژگی در این مقاله، روش الگوی فضایی مشترک است که در اینجا به بیان مفهوم آن می‌پردازیم:

فرض کنید X و Y آزمایش های مربوط به دو کلاس با ابعاد N*M باشند که N تعداد کانال و M تعداد نمونه ها در هر آزمایش است.

واریانس نرمالیزه شده برای این دو آزمایش به صورت زیر محاسبه می‌شود:

$$R_x = \frac{X * X^T}{trace(X * X^T)} \qquad (۱)$$

$$R_y = \frac{Y * Y^T}{trace(Y * Y^T)}$$

www.kahrobamag.ir
کهربا

که در آن trace مجموع درایه های روی قطر اصلی ماتریس و T ترانهاده ماتریس است.

سپس واریانس ترکیبی برای تمامی آزمایشات یک کلاس محاسبه می‌شود:

$$R = <Rx> + <Ry> \qquad (۲)$$

ماتریس R می تواند به بردارهای ویژه خود تجزیه شود:

$$R = P_0 \ \lambda_0 \ P_0^T \qquad (۳)$$

که در آن P۰ ماتریس N*N از بردارهای ویژه R بوده و λ ماتریس قطری مقادیر ویژه آن می‌باشد. توجه داشته باشید که مقادیر ویژه به صورت نزولی مرتب شده‌اند.

حال تبدیل سفیدکننده زیر را در نظر بگیرید که تحت این تبدیل، بردارها به فضایی می روند که بردارهای پایه آن، بردارهای ویژگی P۰ هستند.

$$Q = \sqrt{\lambda_0} \ P_0^T \qquad (۴)$$

حال ماتریس واریانس منحصر بفرد <Rx> و <Ry> به صورت زیر تغییر شکل یافته‌اند:

$$Ux = Q < Rx > Q$$
$$Uy = Q < Ry > Q \qquad (۵)$$

Ux و Uy دارای بردارهای ویژه یکسان بوده و رابطه زیر همواره بین آن‌ها برقرار خواهد بود:

$$Ux = U \ \lambda_x \ U^T$$
$$Uy = U \ \lambda_y \ U^T \qquad (۶)$$
$$\lambda_x + \lambda_y = I$$

که در آن I ماتریس واحد است. یعنی جمع مقادیر ویژه هر دو کلاس یک خواهد شد.

به عبارتی بزرگترین مقادیر ویژه یک کلاس متناظر با کوچکترین مقادیر ویژه کلاس دیگر خواهد بود و برعکس. این ویژگی باعث می شود که از بردارهای ویژه U برای کلاسه بندی دو کلاس استفاده شود. سیگنال سفید شده بردارهای EEG در اولین و آخرین بردار ویژگی U (مربوط به بزرگترین مقادیر ویژه دو کلاس) بردارهای ویژگی را می‌دهد که برای جداسازی مناسب هستند.

با ماتریس جداسازی W=(UT Q)T ، می توان سیگنال های EEG را به فضای دیگری برای جداسازی برد. یعنی

$$Z=WE$$

واریانس اولین ستونZ برای یک گروه ماکزیمم و برای گروه دیگر مینیمم می‌باشد و برای سطر آخر برعکس . بنابراین واریانس سطر اول و آخر Z می تواند اطلاعات مرتبطی برای مجزاسازی داشته باشد و همچنین می توان نوشت $E=W^{-1}Z$ یعنی ماتریس E به دو ماتریس تجزیه شده است که ستونهای W-۱ همان(Common spatial pattern (CSP ها می‌باشند و می‌توان گفت ، واریانس سطرهای Z وقتی بالاست که توزیع زمانی این سطرها نیز بالا باشد . چون CSP ها خود از EEG دست آمده‌اند، دارای اطلاعاتی راجع به توزیع دامنه سیگنالها می‌باشند. پس می توان CSP ها را به عنوان بردارهای توزیع منبع EEG

در نظر گرفت که با زمان تغییر نمی‌کنند .

در این راستا، فیلتر فضایی W طوری به دست می‌آید که تابع ارزشی ذیل را حداکثر کند:

$$J(W)=\frac{W^T R_x W}{W^T R_y W} \qquad (۷)$$

با استفاده از روش ضرب کننده لاگرانژ، می توان مساله بهینه‌سازی مقید فوق را با ماکزیمم کردن تابع ذیل به دست آورد:

$$L(\lambda,W)=W^T R_x W-\lambda(W^T R_y W-1)$$

می توان نشان داد فیلتری که رابطه (۸) را حداکثر می کند بردارهای ویژه متناظر با کوچکترین و بزرگترین مقادیر ویژه ماتریس Rx ۱-M=Ry هستند.

رگولار کردن الگوی فضایی مشترک

با وجود کارائی، فیلتر CSP نسبت به نویز و داده های کم آموزشی بسیار حساس است[۶]. یک روش برای حل این مساله اضافه کردن اطلاعات قبلی به CSP و بدین ترتیب رگولار کردن آن می‌باشد. نشان داده شده است که رگولار شده CSP نسبت به CSP بهتر عمل می کند[۴]. روش های مختلفی برای رگولار کردن پیشنهاد شده است. یک روش اصلاح تابع ارزشی الگوریتم CSP به صورت ذیل است:

$$J_p(w)=\frac{W^T R_x W}{W^T R_y W+\alpha P(W)} \qquad (۸)$$

در حالی که P(W) تابع جریمه بوده و بیان کننده این است که چقدر فیلتر W حاوی اطلاعات قبلی است. برای ماکزیمم کردن JP(W) باید P(W) می نیمم شود. پارامتر α مثبت بوده و میزان اهمیت تابع جریمه را مشخص می‌کند و بوسیله کاربر تعیین می شود. تابع جریمه به صورت P(W)=WT K W تعریف شده است. این تابع موجب می شود تا فیلترهای به دست آمده دارای نُرم کوچک باشند و اثر تداخل و نمونه های غیر معمول را کاهش دهد. فیلترهای فضایی که رابطه (۹) را حداکثر می کنند بردارهای ویژه متناظر با بزرگترین مقدار ویژه ماتریس دارای ضرایبی به صورت زیر هستند:

$$M1=(Ry+\alpha K)\,1-Rx$$
$$M2=(Rx+\alpha \mathbf{K})۱-Ry \qquad (۹)$$

به این روش اصلاح شده الگوی CSP، الگوی فضایی مشترک منظم شده یا RCSP می گویند. جدول ۲ به طور خلاصه روش CSP و RCSP را نشان می دهد. الگوری RCSP خود دارای انواع مختلفی است، که یکی از آن ها روش تیخونوف می باشد. در این روش ماتریس K در تابع جریمه به صورت K=I تعریف می شود. نشان داده شده است که الگوری فضایی مشترک منظم شده تیخونوف دارای کارائی بهتری نسبت به دیگر روش های الگوی فضایی مشترک منظم شده دارند.

	CSP	RCSP
تابع ارزشی	$J(W)=\frac{W^T R_x W}{W^T R_y W}$	$J_p(w)=\frac{W^T R_x W}{W^T R_y W+\alpha P(W)}$
روش حل تابع ارزشی	بردارهای ویژه متناظر با کوچکترین و بزرگترین مقادیر ویژه ماتریس M=Ry-1 Rx	بردارهای ویژه متناظر با بزرگترین مقدار ویژه ماتریس M1=(Ry + α K) -1 Rx و M2=(Rx + α K)-1 Ry

روی داده های مربوط به ۹ نفر روش های شرح داده شده را پیاده سازی کرده و نتایج را به صورت جدول ۳ در آوردیم.

subject	C1	C2	C3	C4	C5	C6	C7	C8	C9
CSP	88.89	51.39	96.53	70.14	54.86	71.53	81.25	93.75	93.75
RCSP	88.89	63.19	96.53	66.67	63.19	63.89	78.47	95.83	92.36
TRCSP	88.89	54.17	96.53	70.83	62.5	67.36	81.25	95.87	91.67

نتیجه گیری

در این مقاله تئوری و الگوریتم ساده ای از روش CSP و RCSP ارائه نمودیم. سپس با استفاده از روش تیخونوف به الگوریتم TRCSP دست یافتیم. الگوریتم های معرفی شده را روی داده های مربوط به ۹ فرد پیاده سازی کردیم و نتایج این پیاده سازی را با هم مقایسه کرده مشاهده کردیم در این روش TRCSP به نتایج مطلوب تری دست یافتیم. بنابراین پیشنهاد می دهیم که طراحان BCI از این روش به منظور دست یابی به سیستم های مؤثر و کارآمد واسط مغز و رایانه استفاده نمایند.

در آینده می توان الگوریتم RCSP را به گونه ای بهبود ببخشید که قابل استفاده برای افرادی که می خواهند با سیستم های انتقال موضوع به موضوع کار کنند، در مجموعه های آموزش کوچک به طوری که زمان کالیبراسیون BCI کاهش یابد و برای انطباق با CSP چند کلاسه مورد استفاده قرار گیرد.

مریم فریور

کارشناس ارشد مهندسی برق الکترونیک

الگوریتم PSO

در این بخش موارد زیر را بررسی میکنیم:

- تاریخچه PSO
- مزایا و مشکلات الگوریتم PSO
- تجزیه و تحلیل PSO (پیوسته)
- تجزیه و تحلیل PSO (گسسته)
- بهبود بخشیدن الگوریتم PSO
- نحوه عملکرد PSO گسسته در MATLAB

الگوریتم بهینه‌سازی ازدحام ذرات PSO به عنوان یک تکنیک جستجو توسط ابرهارت و کندی در سال ۱۹۹۵ معرفی گردید. در تدوین این روش از حرکات جمعی و گروهی ذرات الگو برداری شده است و در برخی اوقات این الگوریتم را به اشتباه تحت عنوان الگورتیم پرندگان و یا ماهی ها نیز می‌شناسند. این روش بهینه‌سازی جزء روش‌های هوش ازدحامی یا هوش گروهی به حساب آمده و اساس کار آن بر این اصل استوار است که در هر لحظه هر ذره مکان خود را در فضای جستجو با توجه به بهترین مکانی که تاکنون خود در آن قرار گرفته و بهترین مکانی که در کل یافت شده است، تنظیم می‌کند. در ابتدا این الگوریتم به منظور کشف الگوهای حاکم بر پرواز همزمان پرندگان و تغییر ناگهانی مسیر آنها و تغییر شکل بهینه‌ی دسته جمعی به کار گرفته شد. در PSO، ذرات در فضای جستجو جاری می‌شوند. تغییر مکان ذرات در فضای جستجو تحت تأثیر تجربه و دانش خودشان و همسایگانشان می‌باشد. بنابراین موقعیت دیگر ازدحام ذرات روی چگونگی جستجوی یک ذره اثر می‌گذارد. نتیجه‌ی مدل سازی این رفتار اجتماعی فرایند جستجویی است که ذرات به سمت نواحی موفق میل می‌کنند. در این راستا، ذرات از یکدیگر می‌آموزند و بر مبنای دانش به دست آمده به سمت بهترین همسایگان خود می‌روند. در این تکنیک از مفهوم «جانمایی همسایگی جی‌بست » که توسط کندی و دیگران معرفی شده، استفاده می‌شود. مفهوم جانمایی یعنی هر ذره بهترین موقعیت قبلی خود و هر یک از ذرات موجود در جمعیت را به خاطر می‌آورد و به عبارت دیگر در هر لحظه هر ذره مکان خود را در فضای جستجو با توجه به بهترین مکانی که تا کنون در آن قرار گرفته و بهترین مکانی که در کل همسایگی اش وجود دارد، تنظیم می‌کند. هر راه حل که به آن یک ذره گفته می‌شود، در الگوریتم معادل یک پرنده در الگوی حرکت جمعی پرندگان می‌باشد. هر ذره یک مقدار شایستگی دارد که توسط یک تابع شایستگی محاسبه می‌شود. طبق تحقیقات انجام شده الگوریتم بهینه‌سازی ازدحام ذرات در بسیاری از مسائل مهندسی از قبیل:

- مسئله تخمین پارامتر
- طراحی شبکه های تامین

• مسئله انتخاب سبد بهینه سهام

• مسئله حرکت خودروها

• مسئله برنامه ریزی تولید

• مسئله خوشه بندی

• طبقه بندی

و غیره عملکرد مناسبی را دارا می‌باشد.



حرکت دسته‌جمعی پرندگان

مزایا و مشکلات الگوریتم PSO

مزایا :

• PSO براساس هوش ذرات است.می تواند در تحقیقات علمی و مصارف مهندسی مورد استفاده قرار گیرد.

• PSO محاسبات همپوشانی و جهش ندارد. جستجو با استفاده از سرعت ذره انجام می‌شود. در طی نمو تعدادی از نسل‌ها، تنها مطلوب‌ترین ذره می‌تواند اطلاعات به سایر ذرات فرستد و سرعت جستجو بسیار بالاست.

• PSO کد مقادیر واقعی را می‌پذیرد و این مستقیما توسط پاسخ تصمیم‌گیری می‌شود. تعداد ابعاد برابر ثابت پاسخ است.

• محاسبه PSO بسیار ساده است. در مقایسه با سایر محاسبات پیشرفته، توانایی بهینه‌سازی بیشتری را داشته و به سادگی کامل می‌شود.

مشکلات :

• روش به راحتی تحت تاثیر مقادیر مطلوب جزیی قرا می‌گیرد، که منجر به تنظیم کم دقت سرعت و مسیر می‌شود.

• روش نمی‌تواند روی مشکلات پراکندگی و بهینه‌سازی کار کند.

• روش نمی‌تواند روی مشکلات سیستم ناهماهنگ مانند پاسخ به میدان انرژی و قوانین حرکتی ذرات در میدان انرژی کار کند.

تجزیه و تحلیل PSO (پیوسته)

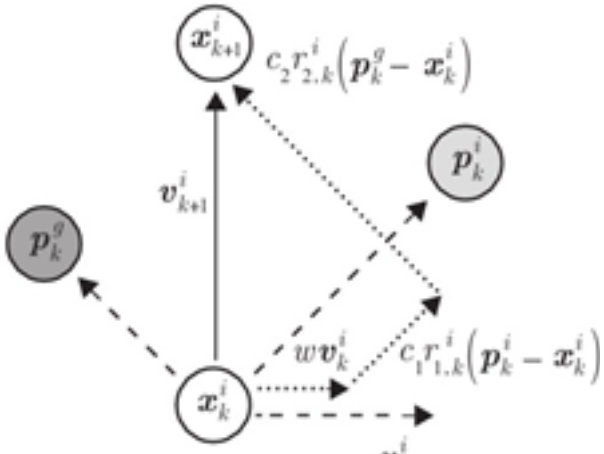
هر ذره دارای یک بردار سرعت است که هدایت حرکت ذره را بر عهده دارد. هر ذره با دنبال کردن ذرات بهینه در حالت فعلی، به حرکت خود در فضای مسأله ادامه می‌دهد. به این شکل است که گروهی از ذرات در آغاز به کار به صورت تصادفی به وجود می‌آیند و با به روز کردن نسلها سعی در یافتن راه حل بهینه می نمایند.

فرض کنید، فضای حل مسئله یک فضای D بعدی می‌باشد در این راستا i امین ذره از جمعیت می‌تواند توسط یک بردار سرعت و یک بردار موقعیت نمایش داده شود. تغییر موقعیت (مکان) هر ذره ، با تغییر در ساختار موقعیت و سرعت قبلی امکان‌پذیر است.

هر ذره امکاناتی شامل بهترین مقداری که تا کنون به آن رسیده است (pbest) و موقعیت xi را داراست. این اطلاعات از تلاش‌هایی که هر ذره برای یافتن بهترین جواب انجام می‌دهد یا به عبارتی از بین موقعیت‌هایی که تا کنون یک ذره در آنها قرار گرفته است، به دست می‌آید. از طرف دیگر، بهترین جوابی را که تا کنون توسط کلیه ذرات موجود در گروه (از مقدار pbest ها) تاکنون تجربه شده است را توسط (gbest) نشان داده می‌شود. در این راستا هر ذره همواره در تلاش است تا موقعیت فعلی خود را در راستای حرکت کنونی، بهترین موقعیت کسب شده توسط خود و بهترین موقعیت کسب شده توسط کل ذرات اصلاح نماید و به موقعیت جدیدی انتقال یابد.

$$v_{ij}^{k+1} = v_{ij}^{k} + c_1 r_1^{k}\left(pbest_{ij}^{k} - x_{ij}^{k}\right) + c_2 r_2^{k}\left(gbest_{j}^{k} - x_{ij}^{k}\right)$$

$$x_{ij}^{k+1} = x_{ij}^{k} + v_{ij}^{k+1}$$



دیاگرام به روز رسانی مکان و سرعت ذره

لذا موقعیت جدید براساس موقعیت کنونی ذره xi، سرعت کنونی ذره vi و فاصله‌ی بین موقعیت کنونی و بهترین پاسخ تجربه شده توسط ذره pbest و همچنین فاصله‌ی بین موقعیت کنونی و بهترین موقعیت کسب شده در کل gbest به دست می‌آید. بدین ترتیب سرعت هر ذره، طبق رابطه‌ی (۱) تغییر می کند، موقعیت جدید هر ذره، بوسیله‌ی مجموع موقعیت گذشته و سرعت جدید، با استفاده از رابطه‌ی (۲) تعیین می‌شود که در آن vij (t)، بعد j ام هر ذره در تکرار t ام است، ω، وزن اینرسی است ، c۱ و c۲ نشان دهنده مقدار سرعت دهنده، تنظیم کننده طول به هنگام پرواز به سمت بهترین ذره گروه و به سمت بهترین ذره به صورت جداگانه است. اگر مقدار خیلی کوچک باشد، ذره احتمالا بسیار دور از مکان مورد نظر است، اگر مقدار خیلی بزرگ باشد،ذره ممکن است سریعا به مکان مورد نظر پرواز کند یا آنسوی مقصد فرود آید. مقادیر مناسب برای c۱ و c۲ میتواند سرعت پرواز ذره را کنترل کند و پاسخ بهتر خواهد بود. معمولا، c۱ با c۲ مساوی بوده و برابر با ۲ هستند؛r۱ و r۲ مقادیر تصادفی بوده که بین ۱-۰ تغییر می‌کند، xij (t) موقعیت بعد j ام هر ذره در تکرار t ام، pbestij برابر pbest بعد j ام هر ذره و Gbestj نیز gbest در گروه است. همچنین به منظور اجتناب از اینکه ذره از مکان مورد نظر خیلی دور شود،

۱. ارزیابی عملکرد هر ذره، توسط مکانی که در حال حاضر ذره در آن است.

۲. مقایسه عملکرد هر ذره با بهترین عملکردی که تا به حالا داشته است:

$$\text{if } x_{ij}(t) < \text{pbest}_{ij} : \ \text{pbest}_{ij} = x_{ij}(t)$$

۳. مقایسه عملکرد هر ذره با بهترین ذره عمومی:

$$\text{if } x_{ij}(t) < \text{gbest}_{j} : \ \text{gbest}_{j} = x_{ij}(t)$$

۴. تغییر سرعت با استفاده از رابطه (۱).

۵. انتقال هر ذره به مکانی جدید با استفاده از رابطه (۲).

۶. بازگشت به مرحله ۲ تا زمانیکه همگرایی رخ دهد.

فلوچارت کلی این الگوریتم در زیر نمایش داده شده است:



تجزیه و تحلیل PSO (گسسته)

کندی و ابرهارت نسخه‌ای باینری گسسته از PSO را برای مسایل باینری پیشنهاد دادند. در این مدل ذرات یا «بله» هستند و یا «خیر»، یا «صحیح» هستند و یا «غلط»، یا «شامل می‌شوند» و یا «شامل نمی‌شوند» وغیره. همچنین این مقادیر باینری می‌توانند مثالی از مقادیر واقعی در فضای جستجوی باینری باشند.

در PSO باینری، بهترین خاطره شخصی ذره و بهترین خاطره عمومی همانند نسخه پیوسته به روز رسانی می‌شوند. تفاوت اصلی بین PSO گسسته با نسخه پیوسته آن در این است که سرعت ذرات بیشتر از نظر احتمال و اینکه یک بیت به یک تغییر می‌کند، تعریف می‌شود.با توجه به این تعریف سرعت باید در محدوده (۱و۰) قرار گیرد.لذا الگویی تولید می‌شود تا تمامی مقادیر عددی سرعت را در بازه (۱و۰) ترسیم کند. تابع نرمالیزه شده‌ای که در اینجا از آن استفاده می‌کنیم تابع سیگموید است:

$$v'_{ij}(t) = \mathrm{sig}\left(v_{ij}(t)\right) = \frac{1}{1+e^{-v_{ij}(t)}}$$

همچنین معادله (۱) برای به روز رسانی بردار سرعت ذره استفاده میشود. و مکان جدید ذره از رابطه زیر بدست می آید:

$$x_{ij}(t+1) = \begin{cases} 1 & \text{if } r_{ij} < \mathrm{sig}\left(v_{ij}(t+1)\right) \\ 0 & \text{otherwise} \end{cases}$$

r_ij یک عدد تصادفی یکنواخت در رنج (۱و۰)

مشکلات اساسی PSO گسسته

در اینجا دو تا از اصلی ترین مشکلات این نسخه از PSO بیان شده است. اولین مشکل مربوط به پارامترها و دومین مربوط به حافظه PSO است.

پارامترهای PSO

صرفا بیان سرعت و مسیر ذرات روی PSO باینری تاثیر نمی گذارد. وزن اینرسی نیز به طور اساسی روی مقادیر واقعی PSO تاثیر می گذارد. درواقع اثرات این پارامترها برخلاف مقادیر واقعی PSO است. در نسخه پیوسته مقادیر بزرگ برای سرعت بیشینه ذرات اکتشاف را به همراه دارد. اما در PSO باینری مقادیر کوچک برای سرعت بیشینه باعث اکتشاف می شود، حتی اگر راه حل های خوبی ارائه داده شده باشد. اگر V_max=۰ لذا جستجو به جستجویی کاملا تصادفی تبدیل می شود. مقادیر بزرگ برای V_max اکتشاف را محدود می سازد. برای مثال اگر V_max=۴ آنگاه sig(V_max)=۰,۹۸۲ لذا احتمال دارد بیت x_ij یک شود.

مشکلاتی نیز برای انتخاب مقدار مناسب برای وزن اینرسی وجود دارد. در PSO باینری، اگر w > ۱ باشد از همگرایی جلوگیری می شود. اگر ۱- > w > ۱ باشد v_ij صفر می شود.

www.kahrobamag.ir
کهربا

برای اینکه $sig(0) = 0.5$ شود باید w>۱ بوده لذا داریم $\lim_{t\to\infty}(v_{ij}(t)) = 0.5$. اگر w>1 سرعت در طی زمان افزایش یافته و $\lim_{t\to\infty}(v_{ij}(t)) = 1$ لذا تمامی بیتها ۱ می شوند. اگر W<-۱ لذا $\lim_{t\to\infty}(v_{ij}(t)) = 0$ و احتمال صفر شدن بیتها افزایش میابد. لذا برای حل این مشکل سه راه وجود دارد:

راه اول: حذف این جمله است. با توجه به اینکه تغییر مکان ذره توسط r_ij به طور تصادفی تحت تاثیر قرار می گیرد، لذا به این جمله احتمالا نیازی نیست.

راه دوم: مقداری تصادفی برای w در رنج (۱و۱-) استفاده کنیم.

راه سوم: از یک عاملی به نام wdamp در رنج (۱و۰) استفاده کنیم تا در هر تکرار مقداری از w کم کند. لذا در انتهای تکرار چنین عبارتی را می نویسیم:

w=w×wdamp

که در الگوریتم استفاده شده در این پروژه از راه سوم استفاده شده است.

حافظه PSO باینری

براساس معادله (۴) مقدار بعدی بیت کاملا مستقل از مقدار حال بیت است و این مقدار با استفاده از بردار سرعت به تنهایی به روزرسانی می شود. در نسخه پیوسته PSO دستور به روزرسانی از مکان حال ذرات و بردار سرعت که صرفا حرکت ذرات در فضا را نشان می دهد استفاده می کند.

بهبود بخشیدن الگوریتم PSO

وزن اینرسی

وزن اینرسی توسط شی وسایرین پیشنهاد شده است. وزن اینرسی ω یک عامل نسبی است که با آخرین سرعت نسبت دارد، و فرمول برای تغییرات سرعت به صورت زیر است:

$$v_{ij}^{k+1} = \omega v_{ij}^{k} + c_1 r_1^{k}(pbest_{ij}^{k} - x_{ij}^{k}) + c_2 r_2^{k}(gbest_{j}^{k} - x_{ij}^{k})$$

اثری که آخرین سرعت بر روی سرعت حال دارد توسط وزن اینرسی قابل کنترل است. هر چه ω بزرگتر باشد، توانایی PSO برای جستجو برای کل بیشتر خواهد شد، و هر چقدر کوچکتر شود، توانایی جسنجوی بخشی PSO بیشتر می‌شود. عموما، ω برابر با ۱ است، لذا در آخرین زمان نسلهای متعدد، کمبودی برای توانایی جستجوی بخشی احساس می‌شود. نتایج تجربی نشان می‌دهد که PSO سرعت بیشتری در همگرایی دارد اگر ω بین ۰/۸ و ۱/۲ باشد. اگر طی آزمایشات، ω بین ۰/۹ تا ۰/۴ تعریف شده باشد براساس این نزول خطی، PSO می‌تواند از ابتدا فضای بزرگتری را جستجو کند و در جایی که بهترین پاسخ وجود دارد سریعا مستقر شود. با کاهش ω، سرعت ذره برای جستجوی بخش مورد نظر(حساس) آهسته خواهد شد.

این روش سرعت همگرایی را زیاد کرده، و تابع PSO ارتقا می‌یابد. زمانی که مشکلی که حل شده خیلی پیچیده باشد، این روش که توانایی جستجوی PSO را برای کل در دوره‌ی بعدی پس از نسلهای متعدد ایجاد می‌کند کافی نیست، مطلوب‌ترین پاسخ یافت نمی‌شود، لذا وزن اینرسی برای حل این مشکل به کار می‌آید.

عامل افزایش همگرایی

الگوریتم PSO با عوامل همگرایی توسط کلرک ام،۱۹۹۹ ،معرفی شده و فرمول زیر تغییرات سرعت

$$v_{ij}^{k+1} = \omega v_{ij}^{k} + c_1 r_1^{k}\left(pbest_{ij}^{k} - x_{ij}^{k}\right) + c_2 r_2^{k}\left(gbest_{j}^{k} - x_{ij}^{k}\right)$$

عامل همگرایی $\varphi > 4$ $=c2+c1$ است. عموما φ برابر با ۴/۱ لذا χ برابر با ۰/۷۲۹ است. نتایج تجربی نشان می‌دهد که در مقایسه با وزن اینرسی، سرعت همگرایی در PSO با عامل همگرایی بسیار بیشتر است. در واقع، هنگامیکه نسبت ω،c۱ و c۲ تصمیم‌گیری می‌شود، دو روش محاسبه مشابه هم هستند. بنابراین، الگوریتم PSO با عامل همگرایی مثال ویژه‌ای از الگوریتم PSO با وزن اینرسی است. در این بین، پارامترهایی که بدرستی انتخاب شده اند می‌توانند عملکرد روش را بهبود بخشند.

اثر انعکاسی

یکی از تکنیک‌های بهبودبخشیدن PSO تغییر مکان دادن ذرات غیرعملی و قرار دادن آن در یک فضای امکان‌پذیر است. در این روش، درصورتیکه ذره از مرز فراتر رود، ذره روی مرز قرار می‌گیرد؛ در هر بعد i :

$$\text{If} \quad X_{i,t+1} > X_{i,max}, \text{then } X'_{i,t+1} = X_{i,max}$$

$$\text{If} \quad X_{i,t+1} < X_{i,min}, \text{then } X'_{i,t+1} = X_{i,min}$$

$X_{i,min}$ و $X_{i,max}$ = حد پایین و بالای بعد i هستند.

در هر شرایطی اگر تنها مکان تغییر کند، لذا به خاطر حرکت آنی براثر سرعت جریان، بسیار محتمل است که ذره در تکرار بعدی ناحیه قابل قبول خود را ترک نماید. شکل ۵۰ دو نظریه برای تغییر مقادیر سرعت را نشان می دهد.

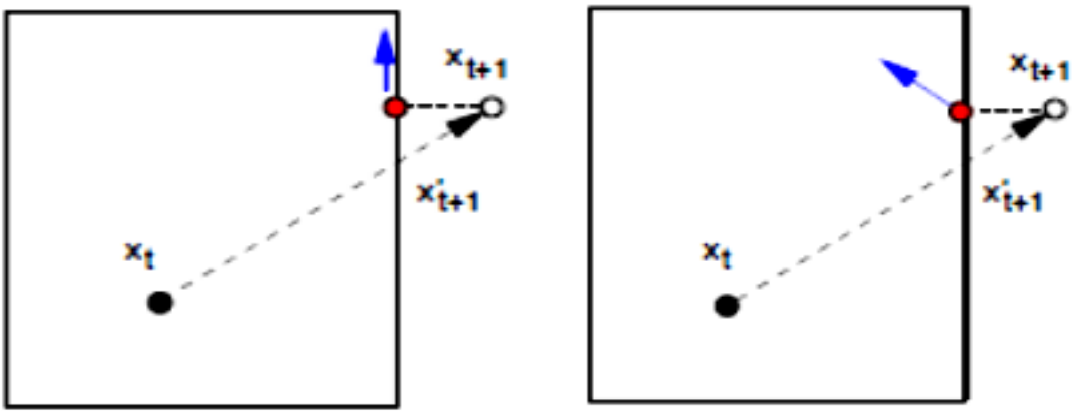
(a)جذب: سرعت صفر می‌شود: $V_{i,t+1} = 0$

(b) بازگشت قطعی: سرعت برای مجبور ساختن ذره جهت بازگشت به فضای قابل قبول در تکرارهای بعدی، معکوس می شود.

$$V_{i,t+1} = -\lambda V_{i,t+1}$$

در این معادله، λ بین ۰،۵ تا ۱ در نظر گرفته می شود.

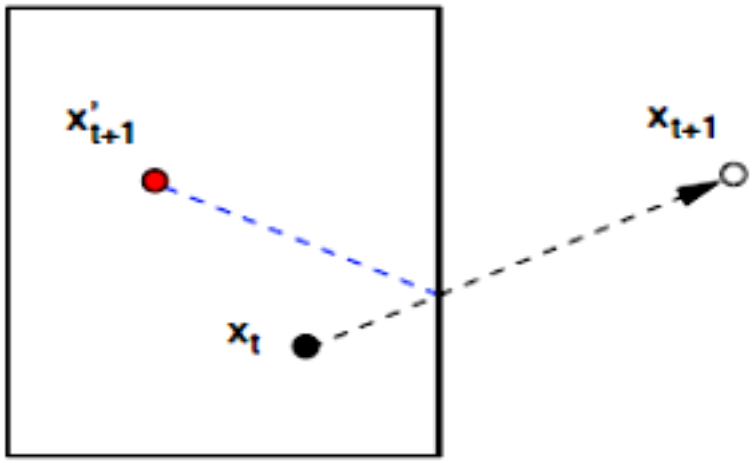
www.kahrobamag.ir
کهربا



الف) جذب ب) بازگشت قطعی

روش‌های مختلف تغییر سرعت برای جابه‌جایی ذره برروی مرز

اثر انعکاسی مکان را در فضای جستجو زمانیکه ذره از مرز فراتر می‌رود معکوس می‌کند. در شکل زیر سرعت ذره توسط یکی از دو روش a یا b به روز رسانی می‌شود (که ما در این مقاله از روش دوم استفاده کرده‌ایم).



اثر انعکاسی. ذره از مرز افقی تجاوز کرده و مکان جدید در فضای جستجو نسبت به محور عمود منعکس شده است – حتی اگر دورتر از مکان قبلی باشد .

نحوه عملکرد PSO گسسته در MATLAB

در این بخش به نحوه نوشتن الگوریتم PSO گسسته در مطلب می‌پردازیم. در PSO ما نیازمند تابع هزینه و الگوریتم هستیم. در اینجا برای نمونه تابع هزینه اسفیر را در نظر میگیریم. این تابع به صورت زیر است:

$$y = \sum x^2 + 3 \times x_1 + 4 \times x_2$$

نحوه نوشتن تابع در MATLAB به شکل زیر است:

function (نام) = (نام تابع هزینه) (نام متغیر خروجی)

(متغیر ورودی

(محل قرار گرفتن تابع مورد نظر)

end

حال به نوشتن الگوریتم PSO می پردازیم. این الگوریتم دارای چند بخش اساسی است که شامل تعریف مسئله، پارامترهای الگوریتم، تنظیمات اولیه، حلقه اصلی و نتایج است. فاز اول شامل:

- تابع هزینه
- تعداد متغیرها
- رنج تغییر متغیرها
- سایز متغیرها

تعریف تابع هزینه به صورت زیر خواهد بود(که ما در اینجا به طور مثال از تابع اسفیر استفاده می کنیم):

CostFunction=@(x) Sphere(x)

از @ جهت تعریف تابع در یک خط استفاده می‌شود. درصورتیکه تعداد ورودی‌های costfunction و sphere با هم برابرباشند می‌توان تابع را به صورت زیر نیز نوشت:

CostFunction=@Sphere

که البته برای اینکه این تابع در تمامی شرایط صادق باشد و برای نوشتن یک تابع جدیدی که ممکن است این تساوی در آن برقرار نباشد شکل کلی را ترجیح می‌دهیم.

در فاز دوم به تنظیمات الگوریتم می‌پردازیم:

- اندازه جمعیت(اندازه ازدحام) swarm size
- شرط توقف: پیشینه تعداد تکرار max iteration
- ضرایب یادگیری c1 و c2
- وزن اینرسی w
- حدود سرعت: velmax و velmin
- حدود سرعت

به این معنی است که قدرمطلق v_ij باید در هر لحظه از v_max کوچکتر باشد.

$$|v_{ij}(t)| < V_{max} \quad => \quad -V_{max} < v_{ij}(t) < V_{max}$$

و V_max ضریبی از عرض فضای جستجو است که این ضریب معمولا ۰,۱ در نظر گرفته میشود؛یعنی:

$$V_{max} = \propto (x_{max} - x_{min}) \qquad (۱۲)$$

و درنهایت حد پایین سرعت را قرینه حد بالای آن در نظر می‌گیریم.

در فاز سوم به تنظیمات اولیه الگوریتم می‌پردازیم. بدین صورت ابتدا یک ساختار خام ایجاد می‌کنیم:

empty_particle = [];

سپس با تکثیر جمعیت، ساختارها را پر می‌کنیم. تابعی که برای تکثیر بکار می‌رود به شکل زیر است:

repmat (m بار ستونی , n نام ماتریس ,بار سطری)

www.kahrobamag.ir
کهربا

بدین معنی که ماتریس مورد نظر را n بار زیر هم و m بار کنار هم می‌نویسد.

در تنظیمات اولیه با توجه به اینکه هنوز ذره‌ای تولید نشده است نمی‌توان رکوردی هم برایش درنظر گفت. در اینصورت برای GlobalBest. Cost چه مقداری را باید انتخاب کنیم؟ بهترین راه حل این است که بدترین حالت ممکن را در نظر بگیریم و چون در PSO هدف ما کمینه سازی است لذا این مقدار را برابر inf یا بی‌نهایت قرار می‌دهیم.برای ثبت رکوردهای جدید کافیست به شکل زیر عمل کنیم:

```
if    Particle(ii).Best.Cost<GlobalBest.
Cost
      GlobalBest.Position = Particle(ii).
Best.Position;
    GlobalBest.Cost = Particle(ii).Best.
Cost;
  End
```

این بدین معنی است که اگر بهترین Cost ذره از GlobalBest کمتر باشد این بهترین ذره اعم از مکان و Cost اش را در GlobalBest قرار می‌دهد. با توجه به این که دو عبارت فوق از نظر تعداد و ماهیت هم جنس هستند می‌توان عبارت فوق را به صورت زیر بنویسیم:

```
GlobalBest = Particle(ii).Best;
```

با توجه با اینکه این اولین ذره بوده و GlobalBest مقدارش بی‌نهایت است این شرط حتما صحیح بوده در دور اول حتما پر می‌شود. در دورهای بعدی این روند ادامه داشته که البته این شرط ممکن است برقرار باشد یا نباشد.

در ابتدا به ارزیابی تابع هزینه پرداخته و مقادیر را در موقعیت جدید بررسی می‌کنیم.

```
Particle(ii).Cost=
feval(CostFuncName,Particle(ii).
Position);
```

سپس در قسمت تنظیمات اولیه مکان و سرعت، سرعت اولیه ذرات را صفر درنظر می‌گیریم و با توجه به اینکه هر مولفه باید صفر باشد از دستور zeros استفاده می‌کنیم. جهت مقدار دهی ماکن اولیه ذرات از دستور unifrnd استفاده می‌کنیم که عددی تصادفی بین VarMin و VarMax و به اندازه VarSize انتخاب می‌کند:

```
Particle(ii)Position=unifrnd(VarMin,Var
Max,VarSize);
Particle(ii).Velocity=zeros(VarSize);
```

بهترین خاطره کجاست؟ با توجه به اینکه تمامی پارامترها در شرایط اولیه قرار دارند لذا خاطره فعلی بهترین خاطره خواهد بود

```
Particle(ii).Best.Position = Particle(ii).
Position;
Particle(ii).Best.Cost = Particle(ii).Cost;
```

در فاز بعدی که حلقه اصلی را تشکیل می‌دهد مراحل زیر را انجام می‌دهیم:

- به روزرسانی سرعت

```
Particle(ii).Vel = w*Particle(ii).Vel
+C1*rand(VarSize).*(Particle(ii).Best.
Position - Particle(ii).Position)
+ C2*rand(VarSize).*(GlobalBest.
Position - Particle(ii).Position);
```

- اعمال حدود برای سرعت به منظور جلوگیری از اینکه سرعت ذره از حد معینی فراتر رود.

```
Particle(ii).Vel = max(Particle(ii).
Vel,VelMin);
Particle(ii).Vel = min(Particle(ii).
Vel,VelMax);
```

- به روزرسانی مکان

```
SigV = 1./(1+exp(-Particle(ii).Vel));
Particle(ii).Position = (SigV > Particle(ii)
.Vel);
```

- اثر انعکاس سرعت

```
IsOutside = (Particle(ii).Position<VarMin
| Particle(ii).Position>VarMax);
Particle(ii).Vel(IsOutside) = -Particle(ii).
Vel(IsOutside);
```

- اعمال حدود مکان

```
Particle(ii).Position = max(Particle(ii).
Position,VarMin);
Particle(ii).Position = min(Particle(ii).
Position,VarMax);
```

- ارزیابی تابع هزینه به منظور بررسی مقدار جدید آن در موقعیت جدید
- بررس شکستن رکورد شخصی و سپس رکورد جمعی. زیرا درصورتیکه ذره نتواند رکورد شخصی خود را بشکند رکورد جمعی نیز شکسته نخواهد شد.

```
If Particle(ii).Cost < Particle(ii).Best.Cost
  Particle(ii).Best.Position = Particle(ii).
Position;
  Particle(ii).Best.Cost = Particle(ii).Cost;
  if Particle(ii).Best.Cost < GlobalBest.
Cost;
    GlobalBest = Particle(ii).Best;
  end
end
```

- ذخیره بهترین جواب بدست آمده از حلقه تکرار **GlobalBest**

```
BestCost(Iter) = GlobalBest.Cost;
```

و در آخر نتایج بهینه‌سازی را مشاهده می‌کنیم.

مهسا سرائی

کارشناس ارشد مخابرات میدان

```
MyCost
function [z sol] = MyCost (X)
global NFE;
   if isempty(NFE)
      NFE=0;
   end
   NFE=NFE1+;
z = sum(X.^3+(2*X(4 + (1*X(2);
sol.z=z;
end
1.2.4. PSO
clc;
clear;
close all;
%% Problem Definition
global NFE;
NFE=0;
CostFunction=@(x) MyCost (x);        % Cost
Function
nVar=5;      % Number of Decision Variables
VarSize=[1  nVar];      % Size of Decision
Variables Matrix
VarMin=0;       % Lower Bound of Variables
VarMax=1;       % Upper Bound of Variables
%% PSO Parameters
MaxIt=200;          % Maximum Number of
Iterations
nPop=100;          % Population Size (Swarm
Size)
w=1;          % Inertia Weight
wdamp=0.99;      % Inertia Weight Damping
Ratio
c2=1;          % Personal Learning Coefficient
c2=2;          % Global Learning Coefficient
% Velocity Limits
VelMax=0.1*(VarMax-VarMin);
VelMin=-VelMax;
 %% Initialization
 empty_particle.Position=[];
empty_particle.Cost=[];
empty_particle.Sol=[];
empty_particle.Velocity=[];
empty_particle.Best.Position=[];
empty_particle.Best.Cost=[];
empty_particle.Best.Sol=[];
 particle=repmat(empty_particle,nPop,1);
 GlobalBest.Cost=inf;
 for i=1:nPop

   % Initialize Position
    particle(i).Position=unifrnd(VarMin,VarM
ax,VarSize);

   % Initialize Velocity
   particle(i).Velocity=zeros(VarSize);

   % Evaluation
              [particle(i).Cost     particle(i).
Sol]=CostFunction(particle(i).Position);

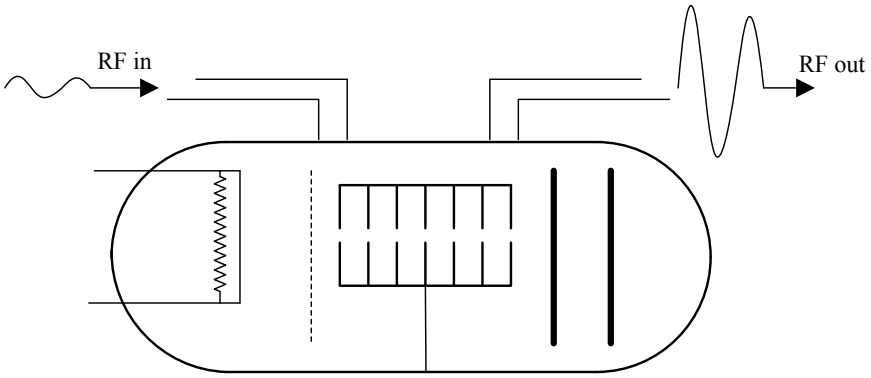
   % Update Personal Best
         particle(i).Best.Position=particle(i).
Position;
   particle(i).Best.Cost=particle(i).Cost;
   particle(i).Best.Sol=particle(i).Sol;
```

```
% Update Global Best
   if particle(i).Best.Cost<GlobalBest.Cost
      GlobalBest=particle(i).Best;
   end
end
BestCost=zeros(MaxIt,1);
nfe=zeros(MaxIt,1);
%% PSO Main Loop
for it=1:MaxIt
   for i=1:nPop
      % Update Velocity
      particle(i).Velocity = w*particle(i).Velocity
...
         +c1*rand(VarSize).*(particle(i).Best.
Position-particle(i).Position) ...
         +c2*rand(VarSize).*(GlobalBest.
Position-particle(i).Position);
      % Apply Velocity Limits
      particle(i).Velocity = max(particle(i).
Velocity,VelMin);
      particle(i).Velocity = min(particle(i).
Velocity,VelMax);
      % Update Position
      SigV = +1)/.1exp(-particle(i).Velocity));
      particle(i).Position = (SigV > particle(i).
Velocity);
      % Velocity Mirror Effect
      IsOutside=(particle(i).Position<VarMin |
particle(i).Position>VarMax);
         % Update Position
      SigV = +1)/.1exp(-particle(i).Velocity));
      particle(i).Position = (SigV > particle(i).
Velocity);
      % Velocity Mirror Effect
      IsOutside=(particle(i).Position<VarMin |
particle(i).Position>VarMax);
       particle(i).Velocity(IsOutside)=-particle(i).
Velocity(IsOutside);
      % Apply Position Limits
      particle(i).Position = max(particle(i).
Position,VarMin);
      particle(i).Position = min(particle(i).
Position,VarMax);
      % Evaluation
      [particle(i).Cost particle(i).Sol] =
CostFunction(particle(i).Position);
      % Update Personal Best
      if particle(i).Cost<particle(i).Best.Cost
         particle(i).Best.Position=particle(i).
Position;
         particle(i).Best.Cost=particle(i).Cost;
         particle(i).Best.Sol=particle(i).Sol;
         % Update Global Best
         if particle(i).Best.Cost<GlobalBest.Cost
            GlobalBest=particle(i).Best;
         end
      end
   end
   BestCost(it)=GlobalBest.Cost;
   nfe(it)=NFE;
   disp([‹Iteration ‹ num2str(it) ‹: NFE
= ‹ num2str(nfe(it)) ‹, Best Cost = ‹
num2str(BestCost(it))]);
   w=w*wdamp;
end
%% Results
figure;
plot(nfe,BestCost,›LineWidth›,2);
xlabel(‹NFE›);
ylabel(‹Best Cost›);
```

ساختار لامپ TWT

لامپ TWT یکی از انواع لامپ‌های ماکروویو است که ساختار آن در حالت کلی دارای اجزایی مشابه با بقیه این لامپ‌ها می‌باشد. انواع متداول کلایسترون تقویت‌کننده‌های RF کلاس A می‌باشند. به این معنی که در صورت عدم اعمال RF، لامپ، توان دریافتی از منبع تغذیه را تلف می‌کند و در صورت وجود RF، قسمتی از این توان دریافتی از منبع تغذیه، صرف تقویت RF می‌شود. در این لامپ با استفاده از یک باریکه الکترونی که توسط یک تفنگ الکترونی ایجاد می‌شود به امواج RF انرژی داده می‌شود و عمل تقویت RF صورت می‌گیرد. در TWT سرعت حرکت طولی امواج کاهش می‌یابد و در حد سرعت جریان الکترونی می‌شود که توسط تفنگ الکترونی در لامپ برقرار شده است. در این حالت امکان واکنش بین الکترون‌ها و امواج فراهم شده و الکترون‌ها قسمتی از انرژی جنبشی خود را به صورت امواج RF آزاد می‌کنند و به این ترتیب RF تقویت می‌شود. با این دید این لامپ یک تقویت‌کننده می‌باشد

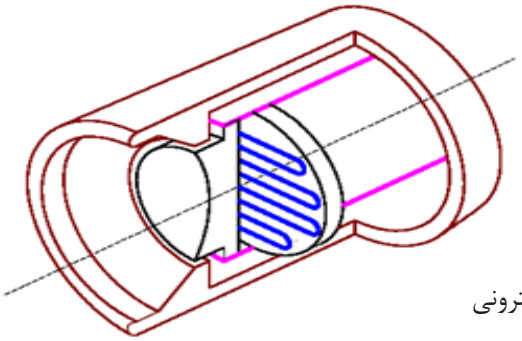
اجزای تشکیل دهنده کلایسترون به شرح زیر می‌باشند.



ساختار کلی لامپ TWT

الف – بخش تفنگ الکترونی

تفنگ الکترونی وظیفه تولید جریان الکترونی را بر عهده دارد که از چند بخش اساسی تشکیل شده است که در ادامه توضیح داده می‌شوند.



ساختار تفنگ الکترونی

هیتر

وظیفه هیتر گرم کردن کاتد و رساندن آن به دمایی است که پدیده ترمویونیک و اثر ادیسون رخ می‌دهد. جنس هیتر سیم تنگستن است که با لایه‌ای از اکسید آلومینیوم از کاتد جدا می‌شود.

کاتد

کاتد از ماده ای با قابلیت وقوع پدیده ترمویونیک در آن، ساخته شده است. در اثر گرم‌شدن، کاتد الکترون آزاد می‌کند و جریان الکترون درون لامپ را بوجود می‌آورد. نسبت قطر کاتد به باریکه الکترونی حدود ۱:۴۰ الی ۱:۱۲۰ است.

آند

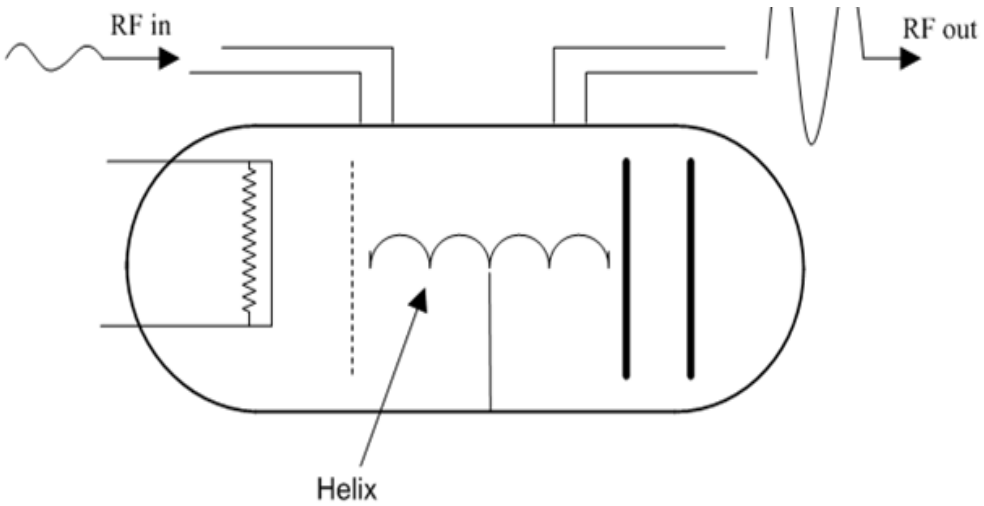
اختلاف پتانسیل بین آند وکاتد باعث شتاب دادن به الکترون‌ها می‌شود و جریان الکترونی در لامپ جاری می‌شود. از آند می‌توان برای متمرکز کردن اشعه نیز استفاده کرد.

ب - بخش کند کننده امواج

همانطور که ذکر شد، هدف از این بخش، کند کردن حرکت طولی امواج RF است طوری‌که سرعت طولی آنها در امتداد لامپ با سرعت حرکت الکترون‌ها برابر شود و فرآیند تقویت امواج صورت گیرد. از این نقطه نظر دو نوع مختلف از لامپ TWT وجود دارد که به شرح زیر می‌باشند.

۱- لامپ TWT از نوع هلیکس

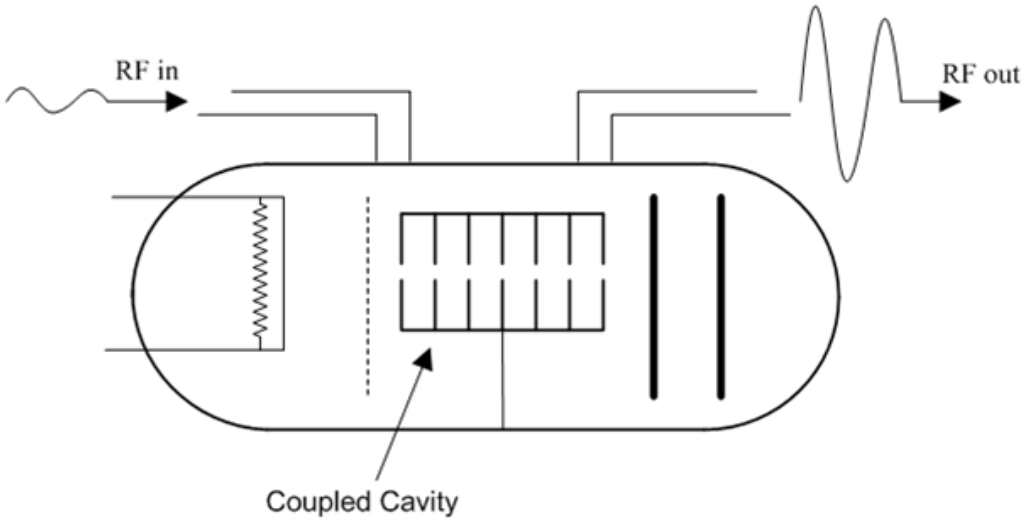
در این نوع که نمودار کلی آن در شکل زیر نشان داده شده است فرایند کاهش سرعت RF توسط شبکه هلیکس انجام می‌شود. هلیکس معمولا بصورت سیم فنری شکلی می‌باشد.



ساختار لامپ TWT با هلیکس

۲- لامپ TWT از نوع Coupled Cavity

در این نوع لامپ فرایند تقویت در **cavity** انجام می‌شود. این نوع نسبت به قبلی توان بالاتری تولید می‌کند.



ساختار لامپ TWT با Couypled cavity

TWT می‌گذرد، الکترون‌های مجاور یکدیگر را دفع می‌کنند و در نتیجه باریکه الکترونی میل به واگرا شدن دارد. برای جلوگیری از این موضوع در داخل **TWT** تعدادی آهن‌ربای دائمی‌قرار می‌گیرد که از واگراشدن الکترون‌ها جلوگیری می‌کنند. به این آرایش قرارگیری آهن‌رباها **PPM** گفته می‌شود.

ج - کلکتور

وقتی جریان الکترونی به انتهای لامپ می‌رسد در آنجا توسط کلکتور جمع‌آوری می‌شود. از آنجا که الکترون‌ها هنوز دارای انرژی جنبشی می‌باشند، فرآیند جمع‌آوری الکترون‌ها با ایجاد گرما همراه است.

بنابراین گرم‌ترین بخش لامپ، کلکتور آن است که باید به کمک هیت سینک مناسب آن را خنک کرد. جنس کلکتور باید از ماده‌ای باشد که درآن پدیده ترمویونیک رخ ندهد تا از حرکت معکوس الکترون‌ها در لامپ جلوگیری شود.

بعلت شکل خطی باریکه الکترونی لامپ **TWT** یکی از خانواده لامپ‌های تقویت‌کننده خطی می‌باشد در **TWT** فرایند پالسی کردن **RF** بکمک شبکه گرید انجام می‌شود بنابراین مدولاتور **TWT** ساختار ساده‌تری از انواع دیگر لامپ‌های ماکروویو دارد زیرا در **TWT** ولتاژ پالس گرید دامنه کوچکی دارد و توان آن نیز بسیار کم می‌باشد اما در لامپ‌های دیگر ولتاژ پالسی کاتد در عین‌حال

که دارای ولتاژ بالایی می‌باشد تمامی‌توان پالس را نیز تحویل می‌دهد و ساخت آن مشکل است. البته با تکنولوژی موجود لامپ از نوع TWT نمی‌تواند توان بسیار بالایی داشته باشد زیرا شبکه گرید بسیار ظریف است و در توان‌های بالا جریان‌های نشتی این شبکه آنقدر بالا می‌رود که باعث از بین رفتن گرید می‌شود. به همین دلیل تنها گزینه لامپ تقویت‌کننده در توان‌های خیلی بالا لامپ کلایسترون است.

تفاوت دیگر انواع مختلف لامپ‌های ماکروویو و TWT در وجود Depressed Collector در TWT و عدم وجود آن در دیگر لامپ‌ها می‌باشد. در لامپ TWT کلکتور در ولتاژی که نسبت به کاتد مثبت است قرار می‌گیرد و در نتیجه بخشی از انرژی باریکه الکترونی که به کلکتور می‌رسد جذب میدان کلکتور می‌شود. در حالیکه در دیگر لامپ‌ها این انرژی بصورت گرما تلف می‌شود. بنابراین کلکتور دیگر انواع لامپ عمدتا توان بالایی را تلف می‌کند و با آب خنک می‌شود.

میدان الکتریکی که در لامپ توسط امواج RF تولید شده است یک میدان متغیر با زمان است و الکترون‌ها را مدوله می‌کند. در این حالت باریکه الکترونی بصورت خوشه‌ای (Bunched) در می‌آید و به بعضی الکترون‌ها شتاب مثبت وارد شده و به بعضی دیگر شتاب منفی اعمال می‌شود. الکترون‌هایی که به آنها شتاب منفی وارد می‌شود، بخشی از انرژی جنبشی خود را به امواج RF می‌دهند و باعث تقویت RF می‌شوند. این فرآیند در طول لامپ ادامه می‌یابد و چگالی بار مجتمع در یک خوشه با مربع فاصله افزایش می‌یابد. به بیان ساده‌تر در صورتی‌که شرایط برای برخورد الاستیک کامل بین الکترون و فوتون درون لامپ فراهم شود، لامپ در حالت تقویت‌کنندگی قرار می‌گیرد. در این حالت، انرژی الکترون‌ها در برخورد الاستیک مذکور به فوتون‌ها منتقل می‌شود و RF تقویت می‌شود. از نظر توان خروجی، وقتی که توان RF ورودی کوچک است، توان خروجی به صورت خطی افزایش می‌یابد. اما پس از مدتی با بالا بردن توان RF ورودی ، خروجی به اشباع می‌رسد.

برای توان خروجی RF در حالت اشباع ، بازدهی لامپ بصورت زیر تعریف می‌شود:

$$\eta_e = \frac{P_{rf}}{IV}$$

که rf P توان RF خروجی و V و I ولتاژ و جریان کاتد می‌باشند. جریان باریکه الکترونی بصورت زیر محاسبه می‌شود.

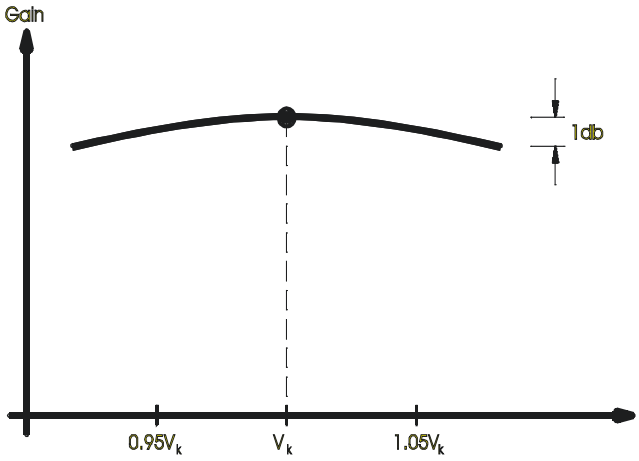
$$I = Per \cdot V^{3/2}$$

که در این رابطه Per، ثابت لانگمیر می‌باشد و بستگی به هندسه الکترودهای درون لامپ دارد. به این ترتیب رابطه توان RF بصورت زیر بدست می‌آید.

$$P_{rf} = \eta_e \cdot Per \cdot V^{5/2}$$

که با معلوم بودن بازده و ثابت Per و توان RF ، ولتاژ کاتد محاسبه می‌شود. البته این محاسبات بیشتر در مرحله ساخت لامپ به کار می‌روند و در مرحله طراحی مدولاتور اطلاعات سازنده لامپ که به عنوان شناسنامه لامپ ارائه می‌شوند، استفاده می‌شوند.

از نظر توان خروجی، وقتی که در TWT توان RF ورودی کوچک است، توان خروجی به صورت خطی افزایش می‌یابد. اما پس از مدتی با بالا بردن توان RF ورودی ، خروجی به اشباع می‌رسد. تغییرات گین برحسب ولتاژ کاتد نیز رفتاری به صورت شکل زیر دارد.



وابستگی گین به ولتاژ کاتد

مهدی عارف خیابانی

کارشناسی مهندسی برق

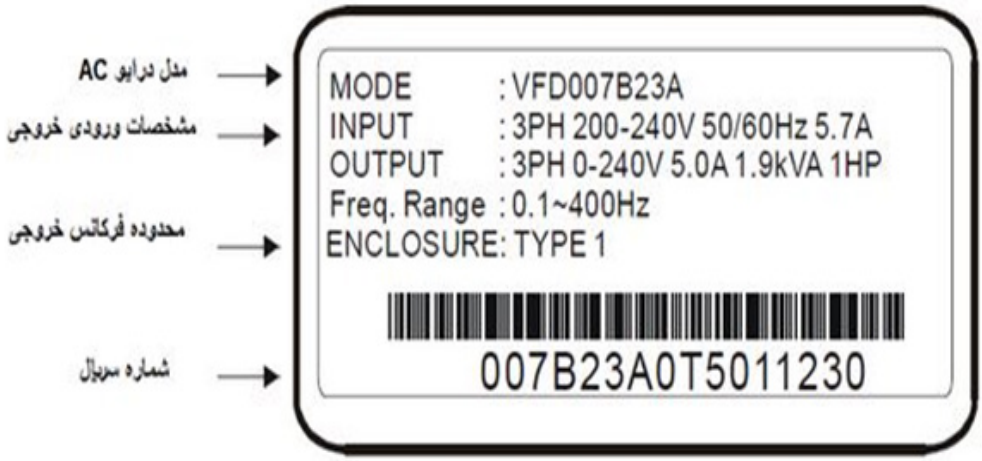
آموزش اینورتر دلتا

ترکیب سیستم های الکترونیک قدرت (مانند کنترل کننده های ولتاژ و فرکانس) و موتورهای الکتریکی همراه با مکانیسم کنترل آنها را محرک‌های تنظیم‌پذیر سرعت می‌نامند که عموما به اختصار VFD (Variable Frequency Drive) یا درایوهای فرکانس متغییر می‌نامند. در حقیقت این محرک ها قابل تنظیم بوده و برای کنترل سرعت یا کنترل گشتاور موتورهای الکتریکی مورد استفاده قرار می‌گیرند .

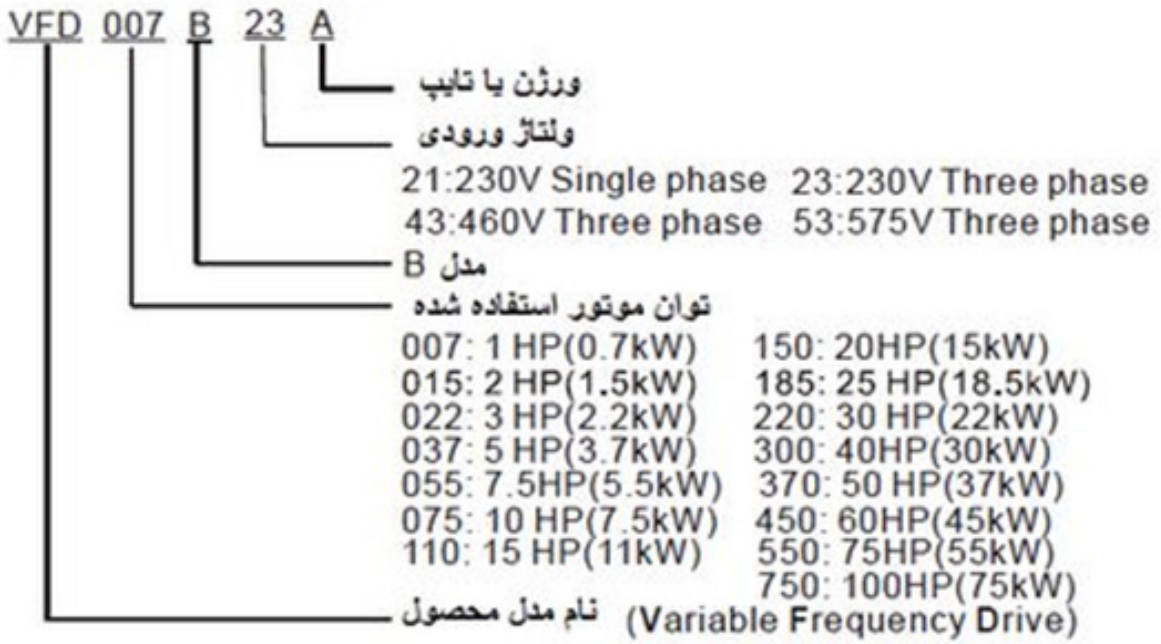
امروزه درایو موتورهای AC چه در حوزه برندها وشرکت‌های تولیدکننده چه در حوزه عملکرد وقابلیت‌ها گسترش چشمگیری داشته است. شرکت‌های تولیدکننده این درایوها دیگر محدود به چند شرکت معتبر اروپایی، آمریکایی و یا ژاپنی نمی‌شود بلکه شرکت‌های کره‌ای، چینی، هندی و اروپایی بسیار زیادی محصولات اتوماسیون صنعتی را در گستره وسیعی تولید می‌کنند. در حوزه کارایی و عملکرد نیز شاهد گسترش بسیار زیادی هستیم به گونهای که امروزه غالبا تمام تولیدکنندگان درایو محصولات خود رابر اساس حوزه عملکرد خود (مثلا فن وپمپ، جرثقیل، پله برقی، بالابر و آسانسور، مصارف گشتاور ثابت و ...)تقسیم‌بندی کرده‌اند. برخی از درایوها امروزه از قابلیت‌هایی چون PLC داخلی، نصب کارت انکودر، ساپورت شبکه‌های مختلف صنعتی، پشتیبانی از مدهای مختلف کاری و ... برخوردار می‌باشد که بسته به احتیاجات لازم جهت کنترل یک پروسه یا ماشین و مدیریت هزینه مهندس طراح باید درایو مناسب را انتخاب نماید.در ادامه ما درایو (اینورتر) دلتا را بعنوان نمونه انتخاب کرده و مراحل کد خوانی، وایرینگ و تنظیم پارامتر را آموزش می‌دهیم.

اطلاعات پلاک مشخصات آموزش اینورتر دلتا

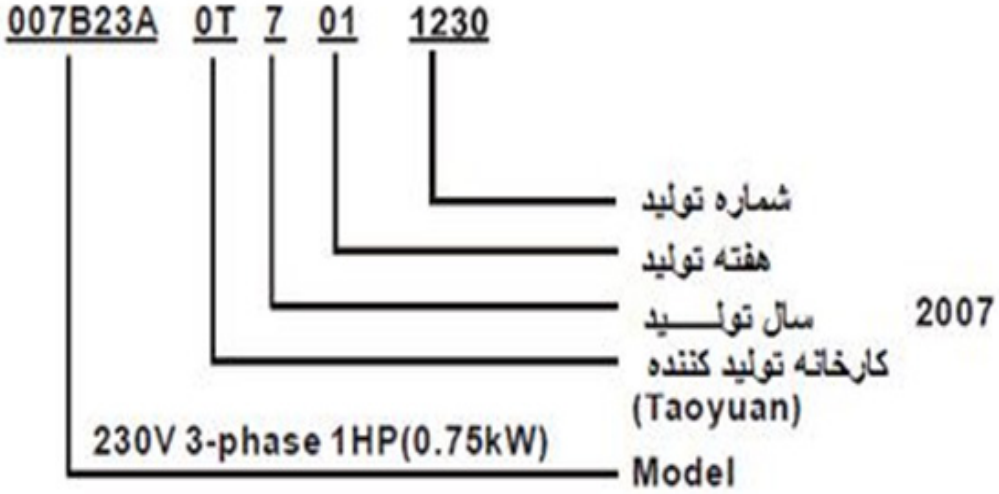
بعنوان مثال برای درایو AC مدل ۱ HP /0.75KW 3Phase



توضیح مدل درایو



توضیح شماره سریال درایو



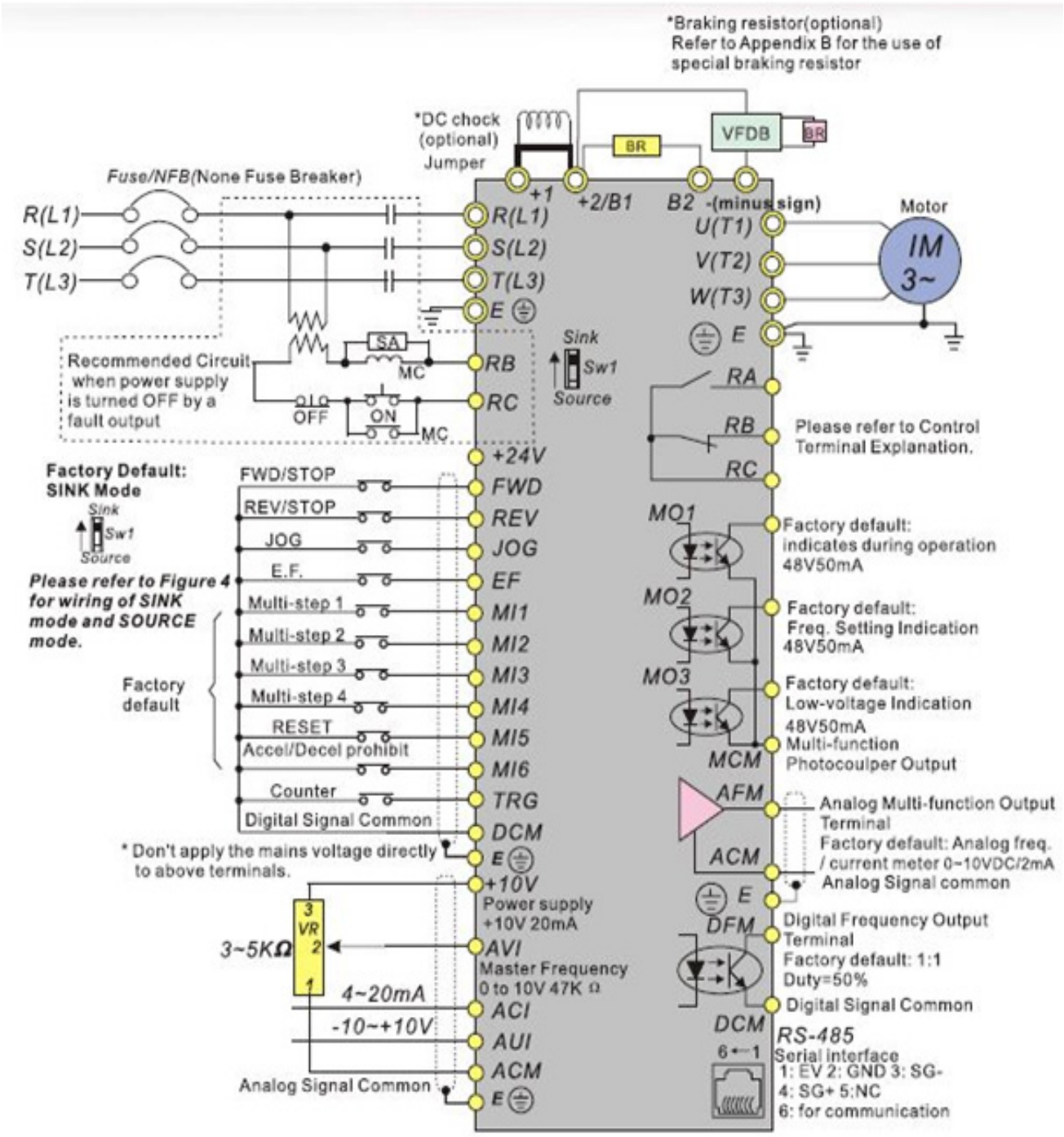
شرایط محیطی محل نصب درایو دلتا

درایو را در محیطی با شرایط مندرج در ذیل باید نصب کرد:

Operation	Air Temperature:	-10 ~ +40°C (14 ~ 104°F)
	Relative Humidity:	<90%, no condensation allowed
	Atmosphere pressure:	86 ~ 106 kPa
	Installation Site Altitude:	<1000m
	Vibration:	<20Hz: 9.80 m/s^2 (1G) max 20 ~ 50Hz: 5.88 m/s^2 (0.6G) max
Storage Transportation	Temperature:	-20°C ~ +60°C (-4°F ~ 140°F)
	Relative Humidity:	<90%, no condensation allowed
	Atmosphere pressure:	86 ~ 106 kPa
	Vibration:	<20Hz: 9.80 m/s^2 (1G) max 20 ~ 50Hz: 5.88 m/s^2 (0.6G) max
Pollution Degree	2: good for a factory type environment.	

دیاگرام کلی سیم‌کشی درایو دلتا

یوزر یا مهندس نصب باید اینورتر را بر اساس دیاگرام آمده در ادامه و مطابق نیازهای کنترلی و عملکردی سیستم، سیم‌کشی کند.

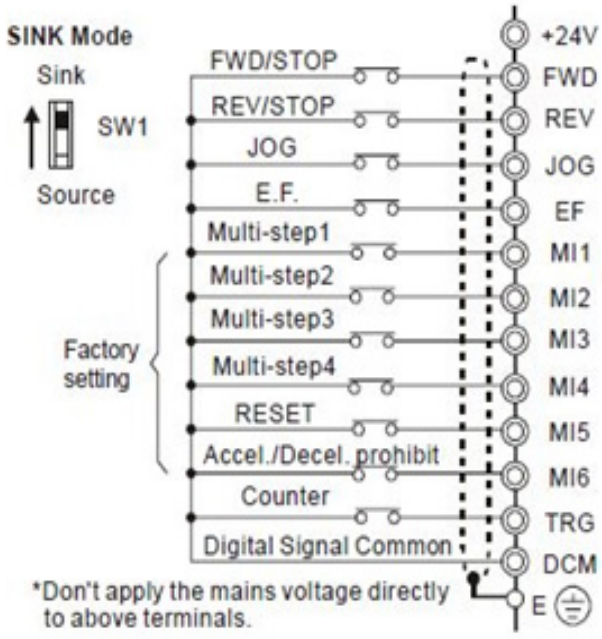


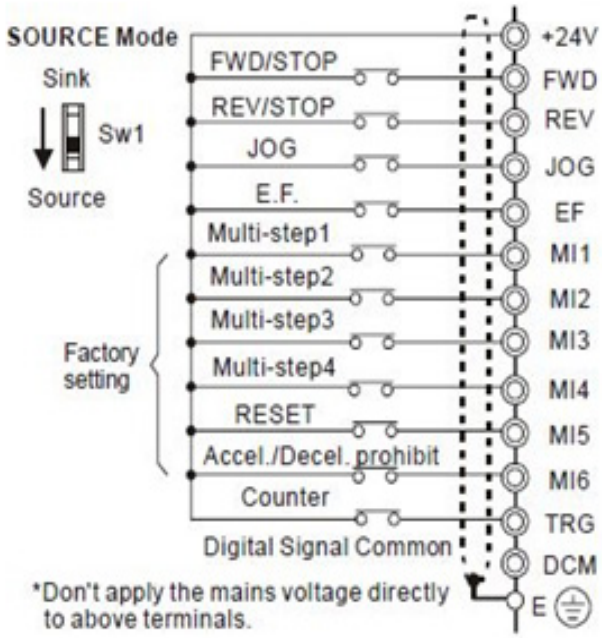
www.kahrobamag.ir
کهربا

توضیح ترمینال‌ها و پورت‌های درایو

ترمینال‌های قدرت

همان‌گونه که در شکل بخش قبلی ملاحظه کردید ترمینال‌های برق ورودی با R ,S ,T یا L۱ , L۲ , L۳ نام‌گذاری شده اند که البته این یک اصل کلی بوده و تقریبا تمام تولید تولیدکنندگان درایو ترمینال‌های برق ورودی را بصورت فوق نام‌گذاری می‌کنند. ترمینال‌های خروجی (کنترلی) موتور که با U , V , W یا T۱ , T۲ , T۳ نام‌گذاری می‌شوند. ترمینال‌های B۱ , B۲ جهت اتصال مقاومت ترمز (Brake Resistor) بوده که توان و مقامت آن متناسب با توان موتور و درایو انتخاب می‌شود. ترمینال‌های مثبت (+) و منفی (-) جهت اتصال یونیت ترمز (درایوهای توان بالا) می‌باشند.





WWW.CONTROLTECH.IR

ترمینال‌های ورودی خروجی دیجیتال

ترمینال های ورودی دیجیتال در اغلب درایوها مربوط به STOP/RUN و FWD/REV (چپگرد و راستگر) و یک‌سری توابع چند منظوره مانند انتخاب چند سرعت، انتخاب زمان شتابگیری و استپ، دکمه (باتن) افزایش پیوسته سرعت، دکمه (باتن) کاهش پیوسته سرعت و ... می‌باشد که بوسیله پارامترهای درایو قابل تنظیم می‌باشد.

برای نمونه اینورتر سری VFD-B دلتا را توضیح می‌دهیم.

ورودی‌های دیجیتال اینورتر مدل VFD-B دلتا قابلیت این را دارد که به دو صورت PNP و NPN مورد استفاده قرار گیرد.برای این منظور جهت استفاده از ورودی‌ها بصورت NPN دیپ سوئیچ تعبیه شده روی برد را در حالت SINK MODE قرار می‌دهیم و برای استفاده بصورت PNP دیپ سوئیچ را در حالت SOURCE MODE قرار می‌دهیم(شکل زیر).

در ادامه ورودی‌های اینورتر سری B دلتا را مرور می‌کنیم:

۲۴+ : تغذیه ۲۴ ولت داخلی درایو

FWD : فرمان حرکت ساعتگرد / استوپ درایو

REV : فرمان بحرکت پاد ساعتگرد / استوپ درایو

JOG : فرمان جاگ یا حرکت دستی موتور

EF : ورودی فالت خارجی برای درایو

MI۶<-MI۱ : ورودی توابع چند منظوره و قابل تنظیم با پارامترهای

۰۴-۰۴.Pr تا ۰۹-۰۴.Pr

TRG : ورودی پالس شمارنده

DCM : پورت مشترک در مد SINK

RA : خروجی رله چند منظوره N.O.

RB : خروجی رله چند منظوره N.C.

RC : مشترک خروجی رله چند منظوره

M۰۳<-MO۱ : خروجی توابع چند منظوره

MCM : مشترک خروجی توابع چند منظوره

۳-۲-۲- ترمینال‌های ورودی خروجی آنالوگ

+۱۰V : تغذیه ورودی آنالوگ ۰-۱۰V , ۴-۲۰MA

AVI : ورودی ولتاژ آنالوگ

ACI : ورودی جریان آنالوگ

AUI : ورودی ولتاژ آنالوگ کمکی

ACM : ترمینال مشترک آنالوگ ورودی

سید مهدی حسینی

کارشناسی مهندسی برق قدرت

سخن پایانی

به پایان آمد این دفتر حکایت همچنان باقیست

این شماره از مجله نیز با حول و قوه الهی به پایان رسید، امید است مورد توجه و عنایت شما واقع گردد. قطعا نظر شما عزیزان در بهبود و ارتقاء و رفع کاستی های مجله کمک شایانی به ما خواهد نمود.

در پایان از زحمات تمامی دوستان عزیزی که مطلب ارسال نمودند و همچنین گروه گرافیک مجله که زحمت ویرایش این نسخه کاملا به عهده داشتند، تشکر و قدردانی داریم.

شما هم می توانید عضو افتخاری مجله ما باشید و با ارایه مطلب ما را درتکمیل شماره بعدی یاری نمایید. شما می توانید مطالب خود را به نشانی ایمیل مجله ارسال کنید:

kahroba@noandishaan.com

منتظر مقالات و مطالب شما هستیم.

صاحب امتیاز

نواندیشان

(www.noandishaan.com)

مدیر مسئول

مهدی عارف خیابانی

سردبیر

زهرا شفیعی

دیگر همکاران

ایمان انصاری
علی کمالی
امین شیخ نجدی
سیدمهدی حسینی
مهسا سرائی
مریم فریور

ویراستار و صفحه آرا

زهرا شفیعی