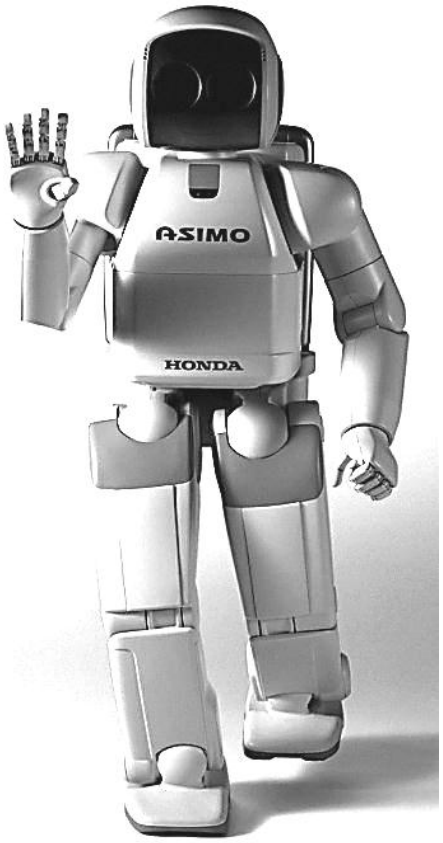
دستور کار

آزمایشگاه روباتیک

بر اساس کتاب «سلامی به روبات‌ها با روبات مسیریاب»



دکتر جواد راستی

مقدمه

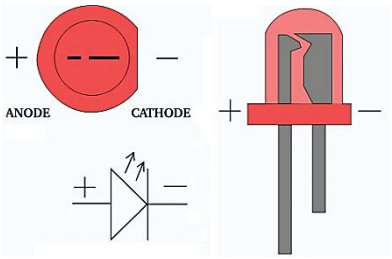
در این دستورکار، تلاش کرده‌ایم مراحل ساخت روبات‌ها را در قالب طراحی یک روبات ساده مسیریاب در یک روند ۵ روزه (روز حسگر، روز مبدل، روز پردازشگر، روز موتور و روز مکانیک که در این نوشتار نیامده است) شرح دهیم. در پایان آزمایش‌ها، باید قادر باشید بلوک‌های مختلف یک روبات را بسازید. در انتهای این دستورکار، نقشه و جزییات طراحی سه روبات مسیریاب (بدون پردازنده، با پردازنده و بدون PWM و با پردازنده و PWM) آمده است که به کمک آنها می‌توانید خودتان یک روبات کامل را بسازید.

قطعات و تجهیزات لازم برای آزمایشگاه

- کامپیوتر – دستگاه برنامه‌ریز (پروگرامر - با قابلیت برنامه‌ریزی میکروکنترلرهای ۸۰۵۱، AVR، PIC)
- چسب برق و تخته سفیدرنگ (پیست روبات مسیریاب)
- بردبورد و ورابورد – سیم استخوانی – هویه - سیم لحیم – قلع‌کش
- منبع تغذیه - رگولاتور 7805
- اسیلوسکوپ - ولت‌متر
- LED به رنگ‌های مختلف - DIP Switch - کلید فشاری - مجموعه کامل مقاومت و خازن و پتانسیومتر
- چند نوع فوتوسل (مقاومت متغیر با نور) - دیود فرستنده و گیرنده مادون قرمز (IR) - ماژول CNY70
- تراشه مقایسه‌کننده (LM311 یا LM324) - تراشه ADC804
- تراشه NOT (7404) - میکروکنترلر AVR مدل ATMEGA32
- آرمیچر - رله و ترانزیستور برای راه‌اندازی آرمیچر - تراشه L298
- موتور پله‌ای - تراشه 74244 - تراشه ULN2003

آزمایش ۱) ساخت مدار حسگر

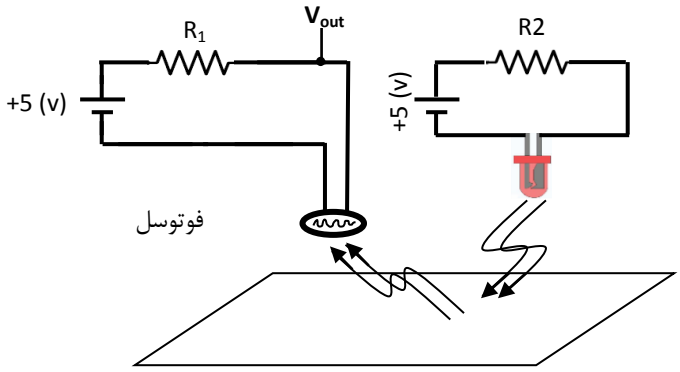
هدف ما در این مرحله ساخت یک مدار حساس به نور با خروجی ولتاژ است که وقتی در دو شرایط نوری تاریک و روشن قرار می‌گیرد، دو ولتاژ با تفاوت قابل ملاحظه (مثلاً دو ولت) تولید کند.



۱) یک عدد LED به رنگ مناسب بردارید و پایه‌های آن را بررسی کنید. پایه بلندتر (آند) را به ولتاژ مثبت و پایه کوتاهتر (کاتد) را به زمین متصل کنید. برای جلوگیری از سوختن LED در اثر عبور جریان زیاد، یک مقاومت چندصد اهمی با آن سری کنید. آیا نور این LED به عنوان فرستنده نوری مناسب است؟

۲) یک عدد فوتوسل بردارید و مقاومت آن را اندازه بگیرید. سپس روی آن را بپوشانید و مجدداً مقاومت آن را اندازه بگیرید. چه نتیجه‌ای می‌گیرید؟

۳) فوتوسل را با یک مقاومت هم‌اندازه آن سری و مطابق شکل زیر به منبع تغذیه متصل کنید. اکنون ولتاژ V_{out} را در حالتی که فوتوسل در معرض نور است و در حالتی که روی آن را پوشانده‌اید یادداشت کنید. این مدار بخش حسگر روبات شماست.



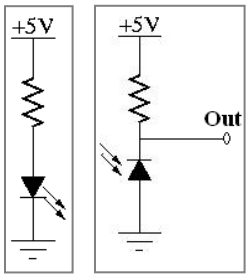
۴) اکنون فرستنده و گیرنده نوری را کنار هم بگذارید و طوری دور آن را با کاغذ یا چسب برق بپوشانید که گیرنده فقط در معرض نور بازگشتی فرستنده از سطح مقابل باشد (بین فرستنده و گیرنده نیز مانعی قرار دهید که گیرنده مستقیماً فرستنده را نبیند). سپس این مجموعه فرستنده و گیرنده را یکبار مقابل سطح سفید و یکبار مقابل سطح سیاه بگیرید و ولتاژ خروجی حسگر را یادداشت کنید. آیا تفاوت ولتاژ در این دو حالت مناسب است؟

۵) مدار گیرنده را طوری تغییر دهید که ولتاژ خروجی آن در محیط سفید بیشتر از محیط مشکی باشد.

۶) این آزمایش را با LEDهایی به رنگهای دیگر نیز انجام دهید و بهترین رنگ LED فرستنده را انتخاب کنید.

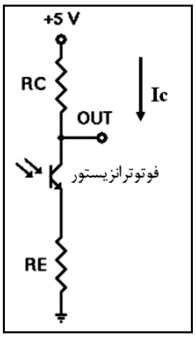
۷) مدار فوق را مقابل چند کاغذ رنگی بگیرید و ولتاژ خروجی حسگر در این حالات را یادداشت کنید. آیا این مدار برای تشخیص رنگ نیز مناسب است؟

۸) مطابق قسمت ۱، یک فرستنده مادون قرمز را راه‌اندازی کنید. برای اینکه از سالم بودن فرستنده مادون قرمز (که نور آن را نمی‌توان با چشم غیرمسلح دید) مطمئن شوید، کافی است به فرستنده در دوربین تلفن همراه خود نگاه کنید تا درخشش نور آن را ببینید.



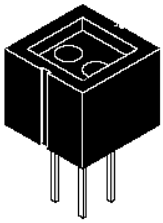
۹) مدار فرستنده/گیرنده مادون قرمز را مطابق شکل زیر ببندید و قسمت‌های ۴ و ۷ آزمایش را در مورد آن تکرار و نتایج را یادداشت کنید. به یاد داشته باشید دیود گیرنده باید در گرایش معکوس بسته شود؛ یعنی پایه بلندتر (آند) را باید به زمین متصل کنید.

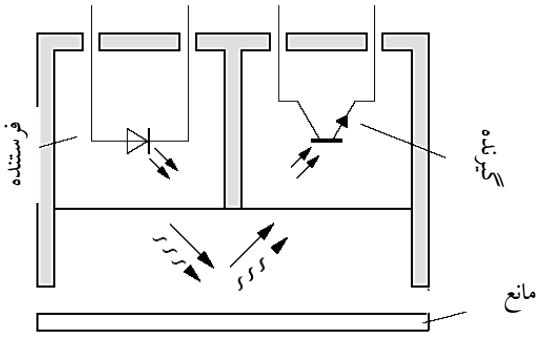
۱۰) آزمایش های فوق را در مورد فوتوترانزیستور تکرار و نتایج را یادداشت کنید. از چه مقاومت‌هایی استفاده می کنید؟



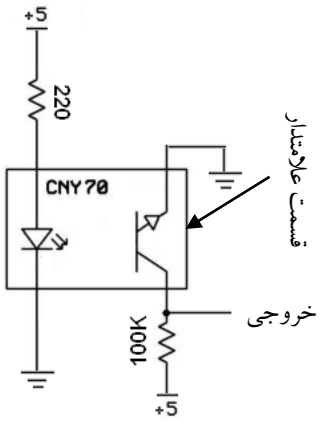
۱۱) چگونه می‌توانید اثر سوء اشباع حسگر را در مورد مدارات خود نشان دهید؟ برای جلوگیری از بروز این مشکل، علاوه بر جداسازی نوری فرستنده/گیرنده از محیط بیرون، باید تنظیم نهایی حسگرها را در محیط مسابقه انجام دهید.

۱۲) قطعه CNY70 یک فرستنده/گیرنده مادون قرمز است که در کنار هم قرار گرفته‌اند و برای تشخیص وجود مانع و نیز فاصله مانع به کار می‌روند. این قطعه از یک دیود و یک فوتوترانزیستور مادون قرمز تشکیل شده است.





مزیت مهم این قطعه، حالت بسته‌بندی آن است که باعث می‌شود زاویه فرستنده همواره نسبت به گیرنده ثابت بماند. توجه کنید که فاصله CNY70 تا مانع مقابل باید کمتر از سه میلی‌متر باشد. به کمک مدار مقابل می‌توانید از حسگر CNY70 استفاده کنید. قطعات GP2S09، GP2S24، GP2S26 و GP2S27 نیز مجموعه بسته‌بندی شده فرستنده/گیرنده مادون قرمز است که مدار آن مانند CNY70 می‌باشد. در سمت فرستنده، یک خوردگی مثلث‌شکل وجود دارد که نشانه فرستنده به شمار می‌رود.

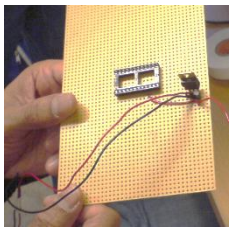


مدار CNY70 را راه‌اندازی کنید و قسمت‌های ۴ و ۷ آزمایش را در مورد آن انجام دهید.

مزیت این مدار نسبت به مدارات قبلی چیست؟

۱۳) چگونه از مدارات بالا می‌توان برای تشخیص وجود مانع استفاده کرد؟

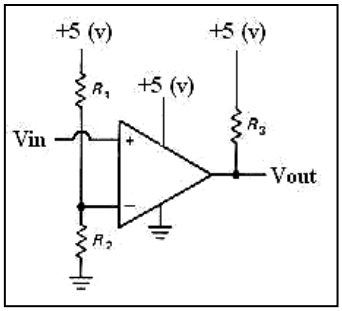
۱۴) آیا این مدارات برای تشخیص فاصله مانع نیز مناسب هستند؟ در مورد مدارات فرستنده/گیرنده ماوراء صوت (Ultrasonic) و لیزری تحقیق کنید و در صورت تمایل آنها را بسازید. این مدارات از اجزاء مهم روباتهایی که باید در محیطهای بسته کار کنند به شمار می‌روند.



۱۵) هرکدام از مدارات را که بیشتر می‌پسندید انتخاب کنید و مجموعه فرستنده/گیرنده را روی ورابورد (veroboard) لحیم کنید و سیم‌های آن را به صورت دسته‌بندی شده و منظم درآورید تا در بخش‌های بعدی بتوانید از آن استفاده کنید.

آزمایش ۲) ساخت مدار سوییچ

در گام اول یک مدار حسگر نوری ساختیم که در سطوح سیاه و سفید ولتاژهای مختلفی را نشان می‌دهد. هدف ما در این مرحله ساخت یک مدار سوییچ است که ولتاژ خروجی حسگر را به یک ولتاژ «صفر» یا «یک» تبدیل کند.



۱) یک تراشه مقایسه‌کننده بردارید و پایه‌های آن را با کاتالوگ تطبیق دهید. به پایه‌های $V_{in(-)}$ و $V_{in(+)}$ آن دو عدد پتانسیومتر متصل کنید تا ولتاژ متغیری را روی این دو پایه تأمین کنند. بقیه مدار را مطابق شکل مقابل تکمیل کنید (مقاومت Pull-up خروجی را 1k انتخاب کنید). به پایه V_{out} مقایسه‌کننده یک LED با مقاومت مناسب متصل کنید. اکنون اگر مقایسه‌کننده به اشباع مثبت برود LED روشن و در غیر این صورت خاموش می‌شود (ممکن است اتصال این LED باعث افت ولتاژ در خروجی شود و آن را از حالت دیجیتال صفر یا ۵ ولت خارج کند؛ در این صورت از یک بافر بین خروجی مقایسه‌کننده و LED استفاده کنید). با چرخاندن پتانسیومتر، ولتاژ ورودی منفی را برابر ۲/۵ ولت تنظیم کنید. اکنون پتانسیومتر متصل به ورودی مثبت را به آرامی بچرخانید و ببینید LED در چه زمانی روشن می‌شود؟ سپس پتانسیومتر را در خلاف جهت بچرخانید و ببینید LED در چه ولتاژی خاموش می‌شود؟

۲) آزمایش قسمت ۱ را با تغییر ولتاژ نقطه آتش مجدداً تکرار کنید. آیا ولتاژ سوییچ دقیقاً برابر ولتاژ نقطه آتش است؟

۳) چگونه می‌توانید مدار بالا را طوری تغییر دهید که LED همیشه روشن باشد و با ورود حسگر به خط، خاموش شود؟

۴) اکنون یک مقاومت 10k بین پایه مثبت مقایسه‌کننده و پتانسیومتر متصل به آن قرار دهید و خروجی (پایه V_{out}) را با یک مقاومت 47k به پایه مثبت مقایسه‌کننده فیدبک کنید تا مدار شما به یک مدار اشمیت‌تریگر یا مقایسه‌کننده غیرمعکوس‌کننده (non-inverting) با هیسترزیس تبدیل شود. آزمایش قسمت ۱ را مجدداً تکرار کنید و ولتاژهای LTP و UTP را اندازه بگیرید. با تغییر مقاومت 47k چه اتفاقی می‌افتد؟

۵) اکنون خروجی حسگر مرحله قبل را به ورودی مثبت مقایسه‌کننده قسمت ۱ متصل کنید. به کمک ولتاژهایی که در آزمایش قبلی به دست آوردید، یک نقطه آتش مناسب یافته و آن را با کمک پتانسیومتر بسازید و به ورودی منفی مقایسه‌کننده متصل کنید. اکنون فرستنده/گیرنده را به آرامی از سطح سفید به سطح مشکی ببرید و ببینید LED خروجی چه زمانی روشن می‌شود؟

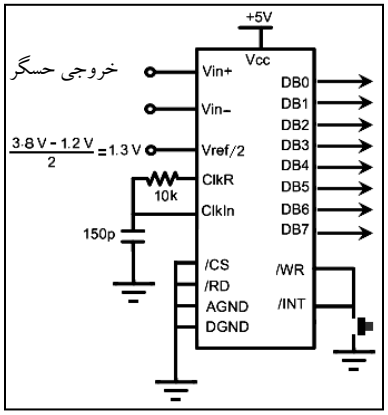
۶) با تغییر ولتاژ نقطه آتش قسمت ۵ آزمایش را تکرار کنید. چه نتیجه‌ای می‌گیرید؟

۷) قسمت‌های ۵ و ۶ را به کمک مدار اشمیت‌تریگر قسمت ۴ مجدداً تکرار کنید. تفاوت نتایج این قسمت با مدار با یک سطح سوییچ در چیست؟

۸) میکروکنترلر AVR مدل ATMEGA32 این قابلیت را دارد که دو سیگنال آنالوگ که به پایه‌های AIN0 و AIN1 (پایه‌های PB2 و PB3) آن متصل می‌شود را با هم مقایسه کند. چنانچه با برنامه‌نویسی میکروکنترلرها آشنا هستید، به کمک این قابلیت میکروکنترلر AVR قسمتهای ۱ و ۵ و ۶ آزمایش را مجدداً تکرار کنید.

آزمایش ۳) آشنایی با ADC

در گام اول یک مدار حسگر نوری ساختیم که در سطوح سیاه و سفید ولتاژهای مختلفی را نشان می‌دهد. هدف ما در این مرحله پیاده‌سازی یک مدار A/D است که ولتاژ خروجی حسگر را به صورت یک عدد ۸ بیتی نمایش دهد.



۱) یک تراشه ADC804 بردارید و پایه‌های آن را با کاتالوگ مطابقت دهید. مدار صفحه ۴۵ را ببندید و ۸ عدد LED با مقاومت مناسب به پایه‌های خروجی آن متصل کنید. پایه $V_{in(-)}$ آن را به زمین متصل کنید تا ADC تنها ورودی $V_{in(+)}$ را به عدد دیجیتال تبدیل کند (چگونه؟). اکنون به کمک دو پتانسیومتر، ولتاژهای متغیری روی پایه‌های $V_{in(+)}$ و $V_{ref/2}$ بسازید. پتانسیومتر متصل به پایه $V_{in(+)}$ را بچرخانید تا ولتاژ خروجی آن صفر شود و همه LEDها خاموش شوند. ولتاژ $V_{ref/2}$ را روی ۲/۵ ولت تنظیم کنید. اکنون کلید فشاری را یک بار فشار دهید و پتانسیومتر متصل به پایه $V_{in(+)}$ را به آرامی بچرخانید و روشن شدن LEDها را ببینید. وقتی پتانسیومتر تا انتها چرخانده شود باید تمام LEDها روشن باشند. اکنون دوباره ولتاژ $V_{in(+)}$ را به صفر برسانید و شروع به چرخاندن پتانسیومتر کنید. در ۵ نقطه دست نگهدارید و ولتاژ $V_{in(+)}$ و عدد خروجی ADC را یادداشت کنید.

۲) اکنون ولتاژ $V_{ref/2}$ را روی ۱ و ۱/۵ و ۲ ولت تنظیم کنید و آزمایش قسمت ۱ را تکرار کنید. نتیجه چیست؟

۳) پتانسیومتر متصل به $V_{in(+)}$ را باز کنید و آن را به $V_{in(-)}$ ببندید و ولتاژ آن را برابر مقدار ولتاژ خروجی حسگرتان در سطح سفید تنظیم کنید (این ولتاژ در متن برابر ۱/۲ ولت فرض شده است). خروجی حسگر را به پایه $V_{in(+)}$ متصل و ولتاژ

$V_{ref/2}$ را با توجه به مطالبی که گفته شد و ولتاژهای خروجی حسگرتان، تنظیم کنید. اکنون کلید فشاری را یک بار فشار دهید و فرستنده/گیرنده را به آرامی از سطح سفید به سطح سیاه منتقل کنید و روشن شدن تدریجی LEDها را ببینید.

۴) به LED متصل به پایه D7 دقت کنید. وقتی این LED روشن می‌شود، ولتاژ $V_{in(+)}$ چقدر است؟ با توجه به این نکته، چگونه می‌توان به کمک ADC یک مدار سوییچ ساخت؟ چگونه با استفاده از مقادیر پایه‌های D6 و D7 می‌توان یک مدار سوییچ ۴ سطحی ساخت؟

۵) با تغییر ولتاژ $V_{ref/2}$، قسمت‌های ۳ و ۴ این آزمایش را تکرار کنید و نتیجه را گزارش کنید.

۶) چنانچه با یک میکروکنترلر دارای ADC داخلی آشنایی دارید (مثلاً AVR مدل ATmega32)، ۵ قسمت قبلی این آزمایش را با آن تکرار کنید و نتایج را گزارش کنید. توجه کنید ADC داخلی میکروکنترلر ATmega32 ده بیتی است و ولتاژ V_{ref} را باید به آن متصل کنید.

پیش‌آزمایش) آشنایی با میکروکنترلر *AVR*

بهترین گزینه برای طراحی بخش پردازشگر روبات، استفاده از میکروکنترلرهاست. انتظار می‌رود با یک میکروکنترلر آشنا باشید تا بتوانید مدارات کنترلی روبات‌های خود را به خوبی طراحی کنید. اگر به یک میکروکنترلر مسلط هستید، می‌توانید از این آزمایش بگذرید. اما اگر تسلط عملی به میکروکنترلر پیدا نکرده‌اید، بهتر است این آزمایش را انجام دهید تا بتوانید در بخش‌های بعد به راحتی بخش پردازشگر روبات خود را طراحی کنید.

۱) به کمک میکروکنترلر AVR سیستمی شامل سخت‌افزار و نرم‌افزار طراحی کنید که اعداد صفر تا ۲۵۵ را مرتباً روی ۸ عدد LED بشمارد.

۲) به کمک میکروکنترلر AVR سیستمی شامل سخت‌افزار و نرم‌افزار طراحی کنید که محتویات یک DIP Switch هشت‌تایی را بخواند و منفی آن را روی ۸ عدد LED نمایش دهد.

۳) به کمک میکروکنترلر AVR سیستمی شامل سخت‌افزار و نرم‌افزار طراحی کنید که با هربار فشار دادن یک کلید فشاری، یک واحد به عددی که روی ۸ عدد LED نمایش می‌دهد، اضافه کند.

۴) قسمت ۱ را طوری اصلاح کنید که با فشار یک کلید فشاری، جهت شمارش عدد (افزایشی یا کاهشی) برعکس شود.

آزمایش ۴) طراحی بخش پردازشگر

در گامهای قبلی مدار حسگر و مدار تبدیل خروجی حسگر به مقادیر قابل فهم پردازشگر را طراحی کردیم. اکنون زمان طراحی بخش پردازشگر است. در ادامه اطلاعات مجموعه حسگر و ADC را به کمک یک میکروکنترلر دریافت و پردازش می‌کنیم. انتخاب نوع میکروکنترلر با شماست؛ اگر از میکروکنترلری ساده مانند ۸۰۵۱ استفاده می‌کنید، باید تراشه خارجی ADC را به آن متصل کنید. میکروکنترلرهای پیشرفته‌تر مانند PIC، AVR و ARM دارای ADC داخلی هستند. در برنامه میکروکنترلر اطلاعات دو حسگر را پردازش می‌کنیم؛ پس دو مجموعه حسگر و ADC آماده کنید. از دو LED برای نشان دادن خاموش و روشن شدن موتورها استفاده می‌کنیم.

۱) برنامه‌ای برای میکروکنترلر مورد استفاده‌تان بنویسید که اگر نیمی از حسگر سمت راست (چپ) وارد خط شد، به راست (چپ) بپیچد. میکروکنترلر را برنامه‌ریزی کنید و حسگر و ADC را به آن متصل کنید و سپس با عبور دادن حسگر از مقابل خط مشکی، صحت کار سیستم خود را کنترل کنید.

۲) سیستم قسمت ۱ را طوری اصلاح کنید که به ورود ⅓ حسگر به خط واکنش نشان دهد و آن را عملاً امتحان کنید. تفاوت آن با سیستم قبلی در چیست؟

۳) سیستم قسمت ۱ را طوری تغییر دهید که دارای دو سطح سوییچ (هیسترزیس) باشد که روی قرار داشتن ⅓ حسگر (سطح LTP) و ⅔ حسگر (سطح UTP) در خط تنظیم شده باشند و آن را عملاً امتحان کنید. تفاوت آن با سیستم قبلی در چیست؟

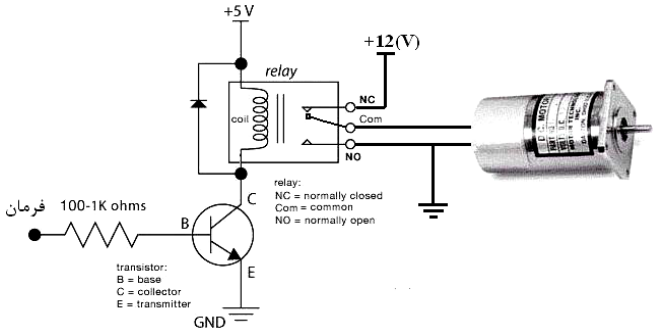
آزمایش ۵) راه‌اندازی موتور DC

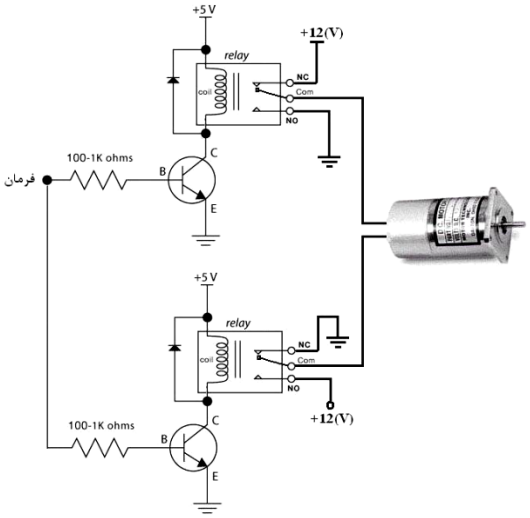
هدف این آزمایش آشنایی با نحوه راه‌اندازی یک موتور DC به کمک رله/ترانزیستور و نیز تراشه L298 است.

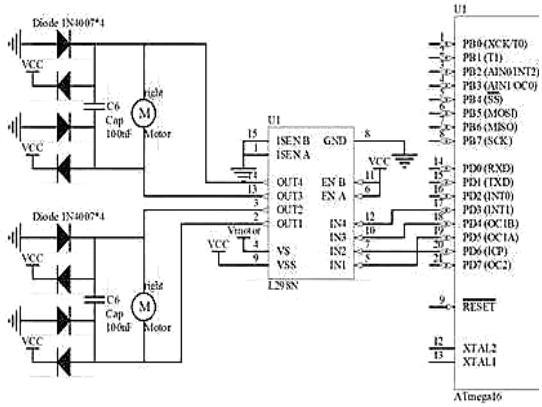
۱) یک موتور DC بردارید و مشخصات آن (ولتاژ کاری، RPM و ...) را بررسی کنید. آن را به منبع تغذیه متصل کنید و با کم و زیاد کردن ولتاژ منبع تغذیه، تغییر سرعت آن را ببینید.

۲) یک رله پنج اتصالی (5-contacts) بردارید و پایه‌های آن را بررسی کنید. به کمک منبع تغذیه، پایه‌های فرمان آن را امتحان کنید.

۳) مدارات زیر را ببندید و مشاهدات خود را گزارش کنید. برای قطع و وصل فرمان از یک کلید فشاری استفاده کنید.







۴) مدار مقابل را ببندید و برنامه‌ای بنویسید که ابتدا هر دو موتور را (به کمک مقداردهی مناسب پایه‌های PD3 تا PD6) به یک سمت بچرخاند. سه کلید فشاری در سیستم تعبیه کنید که با فشردن اولی جهت چرخش موتور بالا برعکس شود، با فشردن دومی جهت چرخش موتور پایین برعکس شود و با فشردن سومی هر دو موتور اگر روشن هستند، خاموش شوند و اگر خاموش هستند، روشن شوند. میکروکنترلر را برنامه‌ریزی کنید و اثر اجرای برنامه را ببینید. در هر مرحله مقدار پایه‌های IN و OUT تراشه L298 را اندازه‌گیری کنید.

۵) اکنون خروجی مرحله قبل (مدار سوییچ یا ADC) را به میکروکنترلر متصل کنید. فرض کنید روی موتور راست روبات خود کار می‌کنید. برنامه‌ای بنویسید که موتور را رو به جلو بچرخاند و اگر نیمی از حسگر وارد خط شد، موتور را خاموش کند. مدار خود را عملاً تست کنید؛ حسگر را از سطح سفید به آرامی به خط مشکی منتقل کنید و اگر در زمان مناسب موتور خاموش شد، دوباره حسگر را به آرامی به سطح سفید منتقل کنید. اگر باز هم در زمان مناسب موتور روشن شد، تبریک ما را پذیرا باشید! مغز کنترلی روبات شما ساخته شده و تنها به یک بستر مکانیکی مناسب نیاز دارد.

۶) برنامه مرحله قبل را طوری اصلاح کنید که سر پیچ‌ها (هنگامی که نیمی از حسگر وارد خط شد)، موتور سمت موافق را معکوس بچرخاند. این مدار را هم عملاً تست کنید.

۷) اکنون می‌خواهیم سرعت موتورها را به کمک PWM کنترل کنیم. برنامه زیر را در محیط CodeVision بنویسید و روی میکروکنترلر ATMEGA32 اجرا کنید:

```
#include <mega32.h>
void main(void)
{
    TCCR1A=0xA1;
    TCCR1B=0x0D;
```

```
    OCR1A = 255;
    OCR1B = 100;

    DDRD = 0xFF;
    PORTD = 0;

    while (1);
}
```

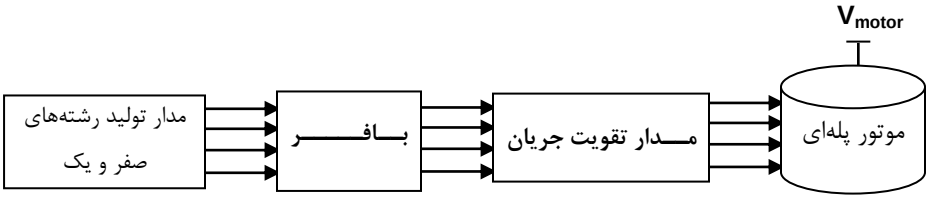
سرعت چرخش موتورها چگونه است؟ مقادیر پایه‌های IN تراشه LM298 را با ولت‌متر و اسیلوسکوپ ببینید. چه نتیجه‌ای می‌گیرید؟

۸) اکنون در برنامه بالا پایه PD6 را یک کنید (PORTD.6 = 1;) و ثبات OCR1A را برابر صفر قرار دهید. نتیجه چیست؟ با زیاد کردن مقدار ثبات OCR1A چه اتفاقی می‌افتد؟ مقادیر پایه‌های IN تراشه LM298 را با ولت‌متر و اسیلوسکوپ ببینید.

۹) برنامه بالا را به نحوی اصلاح کنید که یک موتور را با حداکثر سرعت به سمت راست و دیگری را با نصف حداکثر سرعت به سمت چپ بچرخاند.

آزمایش ۶) راه‌اندازی موتور پله‌ای

هدف ما در این آزمایش آشنایی با نحوه راه‌اندازی یک موتور پله‌ای از دید نرم‌افزار و سخت‌افزار است.



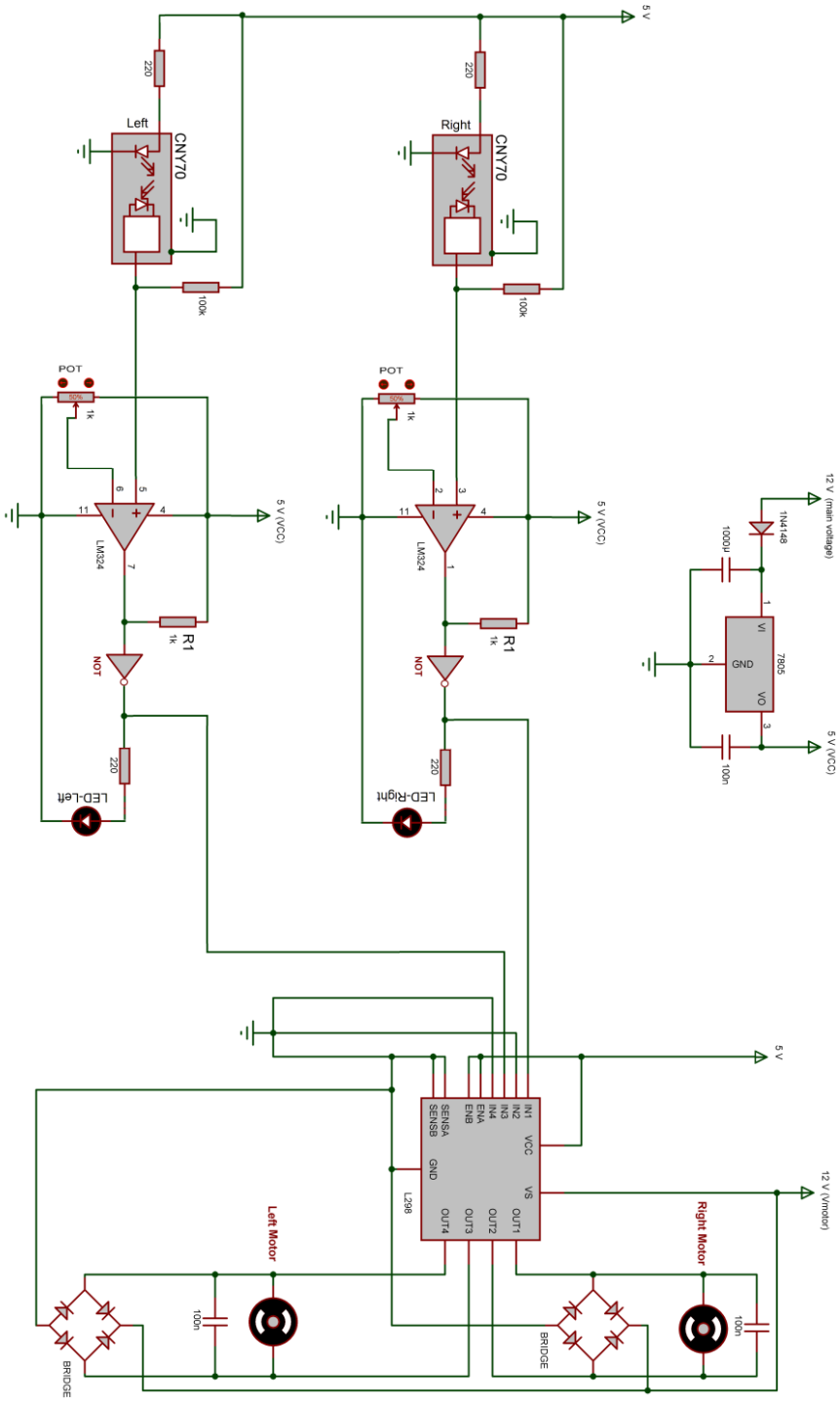
۱) یک موتور پله‌ای در دست بگیرید و شافت آن را بچرخانید. تفاوت شافت این موتور با یک موتور DC در چیست؟

۲) مداری طراحی و پیاده‌سازی کنید یک موتور پله‌ای را بچرخاند:

- این مدار باید یک کلید فشاری داشته باشد که با فشردن آن جهت چرخش موتور تغییر کند.
- سرعت چرخش موتور باید به کمک یک کلید فشاری یا یک DIP Switch قابل تنظیم باشد.

مدار کامل روبات مسیریاب بدون میکروکنترلر

این مدار از دو فرستنده/حسگر مادون قرمز، تراشه‌های مقایسه‌کننده LM324، موتورهای ۱۲ ولت و تراشه راه‌انداز موتور L298 استفاده می‌کند که شماتیک آن را در شکل زیر می‌بینید.



پتانسیومترهای متصل به تراشه مقایسه‌کننده، برای تنظیم نقطه سوییچ (نقطه آتش – Trigger Point) به کار می‌روند. ولتاژ خروجی حسگر در سطح سفید و همین ولتاژ در خط سیاه را اندازه بگیرید و ولتاژ نقطه آنش را حدود میانه آنها تنظیم کنید (مثلاً اگر خروجی حسگر در سطح سفید ۱/۲ ولت و در سطح مشکی ۳/۸ ولت باشد، ولتاژ نقطه آتش را حدود ۲/۵ ولت تنظیم کنید). هرچه این ولتاژ کمتر باشد، روبات هنگام ورود به خط زودتر متوجه می‌شود. مقدار مناسب این ولتاژ به مکانیک و اینرسی روبات شما بستگی دارد. اگر روبات شما کند است و هنگامی که در سر پیچ‌ها به آن فرمان چرخش می‌دهید به فرمان شما به کندی پاسخ می‌دهد، ولتاژ نقطه آتش را کم کنید تا به محض ورود روبات به خط، فرمان پیچیدن صادر شود.

وقتی حسگر در سطح سفید باشد، ولتاژ خروجی آن پایین و کمتر از ولتاژ سطح آتش است. پس خروجی مقایسه‌کننده «صفر منطقی» و پس از عبور از گیت NOT «یک منطقی» می‌شود و با ورود به L298 باعث به حرکت درآمدن موتور می‌شود.

با ورود حسگر به خط، ولتاژ خروجی آن افزایش می‌یابد. پس از عبور ولتاژ خروجی حسگر از ولتاژ آتش، تراشه مقایسه‌کننده سوییچ کرده و خروجی آن «یک منطقی» و با عبور از گیت NOT «صفر منطقی» می‌شود با ورود به L298 باعث توقف موتور می‌شود.

چند نکته در مورد این مدار شایان توجه است:

- از LEDهایی که به خروجی تراشه‌های مقایسه‌کننده LM324 متصل شده‌اند، برای نظارت بر عملکرد حسگرها استفاده شده است؛ هرکدام از این LEDها مربوط به یک حسگر است و با ورود هر حسگر به خط، LED مربوط به آن خاموش می‌شود.
- از تراشه L298 برای راه‌اندازی موتور استفاده شده است. پایه‌های IN1 و IN2 (متصل به OUT1 و OUT2) برای راه‌اندازی موتور سمت چپ و پایه‌های IN3 و IN4 (متصل به OUT3 و OUT4) برای راه‌اندازی موتور سمت راست مورد استفاده قرار گرفته‌اند. پایه‌های IN2 و IN4 به زمین متصل شده‌اند. پایه‌های IN1 و IN3 از خروجی گیت‌های NOT تغذیه می‌شوند و در صورت «یک» شدن، باعث حرکت موتور می‌شوند.
- این مدار بدون استفاده از دیودهای هرزگرد بخش راه‌انداز موتور (پل دیود) نیز به درستی کار می‌کند؛ اما ممکن است به دلیل وجود جریانات بازگشتی باعث سوختن تراشه L298 شود.
- ورودی مدار تغذیه، یک باطری یا منبع تغذیه با ولتاژی برابر ولتاژ موتور است که به کمک رگولاتور 7805 ولتاژ ۵ ولت برای مدارات کنترلی را ایجاد می‌کند. برای نصب رگولاتور، آن را طوری به سمت خود بگیرید که نوشته‌های روی آن رو به شما باشد؛ در این حالت پایه سمت چپ ورودی، پایه وسط زمین و پایه سمت راست خروجی می‌باشد. استفاده از یک دیود محافظ در ورودی تغذیه، مدار را از آسیب‌های ناشی از برعکس وصل کردن قطب‌های تغذیه ایمن می‌سازد.

- اگر از مجموعه دیودهای فرستنده/گیرنده مادون قرمز آماده استفاده می‌کنید، به جهت نصب صحیح پایه‌های آن توجه کنید. اگر از دیودهای مجزا استفاده می‌کنید، دقت کنید که دیود فرستنده به صورت معمولی (پایه کوتاهتر (کاتد) به زمین) و دیود گیرنده به صورت معکوس (پایه بلندتر (آند) به زمین) متصل می‌شوند. اگر فرستنده‌ها را به صورت صحیح بسته باشید و از طریق دوربین تلفن همراه به آنها نگاه کنید، نور ملایمی‌را خواهید دید.
- یکسان بودن تغذیه موتورها و مدار کنترل روبات، ممکن است باعث ایجاد نویز روی تراشه‌های کنترلی شود. استفاده از خازنهای 100nf از نوع پلی‌استر باعث خنثی شدن تاثیرات سلفی موتورها روی مدار کنترل می‌شود.
- در بستن مدار، سعی کنید نظم را رعایت کنید. این کار باعث می‌شود بتوانید با صرف زمان کمی مشکل روبات خود را پیدا کنید یا آن را به سرعت برای تطابق با شرایط جدید، تغییر دهید. به عنوان مثال زمانی که لازم است دو قسمت بورد با چند سیم به هم متصل شوند، بهتر است به جای استفاده از سیمهای مجزا که مدار را شلوغ کرده و امکان خرابی را در آن بالاتر می‌برند، از کابلهای IDC یا IDE به همراه رابط‌های مناسب (شکل مقابل) استفاده کنید.

- برای جلوگیری از قطع شدن اتصالات مدار در طول حرکت روبات، بهتر است آن را روی وِرابورد (veroboard) لحیم کنید یا بورد چاپی (PCB) آن را طراحی کنید. دقت داشته باشید که اجزایی را که نیاز به درآوردن و جاسازی مکرر روی بورد دارند، نباید لحیم کرد؛ چون جابجایی آنها بسیار مشکل است و ممکن است آسیب ببینند. به عنوان مثال ممکن است به دلیل سوختن تراشه راه‌انداز موتور یا تراشه مقایسه‌کننده یا تراشه NOT ناچار به تعویض آنها شویم؛ بنابراین برای اتصال این تراشه‌ها به بورد حتماً از سوکتهای مناسب استفاده کنید.

- حرارت تراشه راه‌انداز موتور (L298) در هنگام کار روبات زیاد می‌شود. برای جلوگیری از سوختن آن، روی آن Heat-Sink نصب کنید.
- این مدار تنها یک سطح سوییچ دارد که با پتانسیومتر تنظیم می‌شود. چنانچه می‌خواهید مدار شما دارای دو سطح سوییچ بالا و پایین باشد (دارای حافظه یا هیسترزیس شود)، یک مقاومت 10k بین پایه مثبت مقایسه‌کننده و خروجی حسگر متصل به آن قرار دهید و خروجی (پایه V_{out}) را با یک مقاومت 47k به پایه مثبت مقایسه‌کننده فیدبک کنید تا مدار شما به یک مدار اشمیت‌تریگر یا مقایسه‌کننده غیرمعکوس‌کننده (non-inverting) با هیسترزیس تبدیل شود. با تغییر مقاومت 47k می‌توانید اختلاف بین سطوح آتش بالا و پایین را تنظیم کنید.

قطعات لازم برای ساخت این روبات

- ۲ فرستنده/گیرنده مادون قرمز (یا قطعه آماده CNY70)

- ۴ مقاومت ۲۲۰ اهم – ۲ مقاومت ۱۰۰ کیلواهم – ۲ مقاومت ۱ کیلواهم – ۲ پتانسیومتر ۱ کیلواهم
- ۲ عدد LED معمولی
- ۱ تراشه LM324 (مقایسه‌کننده)
- ۱ تراشه 7404 (گیت NOT)
- ۱ عدد خازن ۱۰۰۰ میکروفاراد پلی‌استر – ۳ عدد خازن ۱۰۰ نانوفاراد الکترولیت
- ۱ عدد دیود 1N4148
- ۱ عدد پل دیودی (یا ۸ عدد دیود 1N4148 که به حالت پل بسته شوند)
- یک رگولاتور 7805
- دو موتور DC با چرخش ۵۰۰ دور در دقیقه (ترجیحاً گیربکس‌دار)
- تراشه L298 با سوکت مناسب

مدار کامل روبات مسیریاب با میکروکنترلر (بدون PWM)

اکنون مدار یک روبات مسیریاب که از میکروکنترلر استفاده می‌کند را بررسی می‌کنیم. این مدار از سه فرستنده/حسگر مادون قرمز، میکروکنترلر AVR مدل ATMEGA8 با مبدل‌های آنالوگ به دیجیتال داخلی، موتورهای ۱۲ ولت و تراشه راه‌انداز موتور L298 استفاده می‌کند که شماتیک آن را در شکل صفحه بعد می‌بینید.

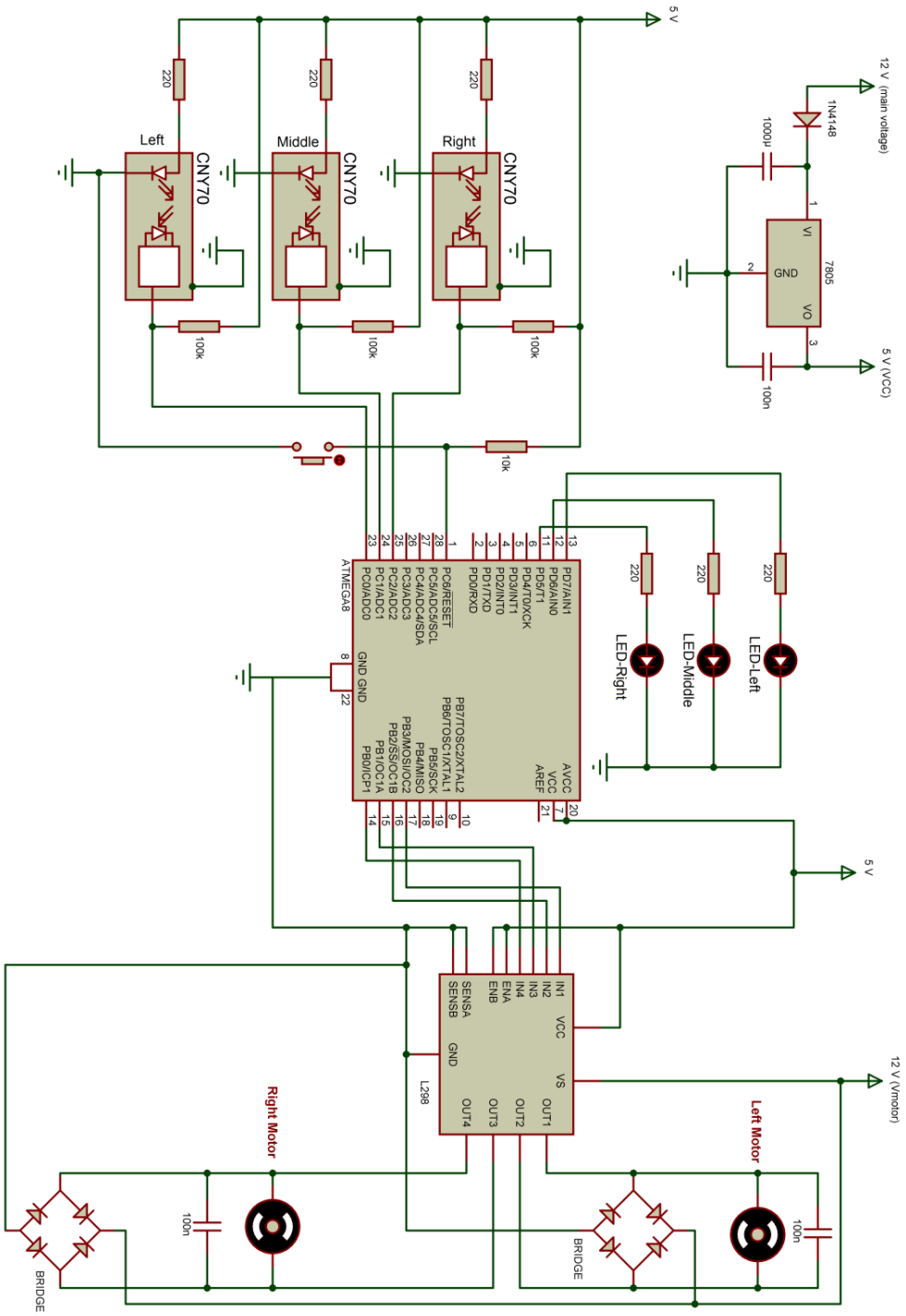
نرم‌افزار میکروکنترلر

برنامه میکروکنترلر AVR برای کنترل این روبات که در CodeVision قابل کامپایل است، به صورت زیر می‌باشد:

```
#include <mega8.h>
#include <delay.h>


#define   ADC_VREF_TYPE   0x00

#define   go_ahead   0b00001010
#define   turn_left     0b00001000
#define   turn_right    0b00000010

#define   m_LTP    650   // LTP for the ADC of the middle sensor
#define   m_UTP   710   // UTP for the ADC of the middle sensor
#define   side_TP  200   // TP for the ADC of the left & right sensors

#define   m_LTP    650
#define   m_UTP   710
#define   side_TP  200
```

```
unsigned int read_adc(unsigned char adc_input)
// input: adc_input is the number of the used ADC
// output: the 10-bit value of the ADC
{
        ADMUX = adc_input | (ADC_VREF_TYPE & 0xFF);

        delay_us(10);

        ADCSRA |= 0x40;

        while ((ADCSRA & 0x10) == 0);

        ADCSRA |= 0x10;

        return ADCW;
}
void main(void)
{
  DDRB=0xFF;
  PORTB=0x00;

  DDRC=0x00;

  DDRD=0xFF;

  ADMUX=ADC_VREF_TYPE & 0xFF;

  ADCSRA=0x83;

  unsigned int left, middle, right;


  while (1) {

            left   = read_adc(0);
            middle = read_adc(1);
            right  = read_adc(2);

            if (middle > m_LTP && right < side_TP && left < side_TP){
               PORTB  = go_ahead;
               PORTD = 0b01000000;
            }

           else if(middle < m_UTP&&right > side_TP&&left < side_TP){
               PORTB  = turn_right;
               PORTD = 0b00100000;
            }
```

```
        else if(middle < m_UTP&&right < side_TP&&left > side_TP){
            PORTB  = turn_left;
            PORTD = 0b10000000;
          }


    }
}
```



علاوه بر نکاتی که در مورد مدار قبلی گفته شد، چند نکته در مورد این مدار شایان توجه است:

- این روبات دارای سه حسگر مادون قرمز است؛ دو حسگر در دو طرف خط و حسگر وسطی روی خط قرار می‌گیرد. همانطور که مشاهده می‌کنید، خروجی حسگرهای نوری به ADCهای داخلی میکروکنترلر AVR متصل شده است؛ به صورتی که مقدار ADC0 خروجی حسگر چپ، مقدار ADC1 خروجی حسگر وسط و مقدار ADC2 خروجی حسگر راست را مشخص می‌کند. به بیان دیگر عدد خروجی ADCها معیار تصمیم‌گیری است. مدار ADC داخلی این مدل از میکروکنترلر AVR دارای دقت ۱۰ بیتی است و ولتاژ آنالوگ ورودی را به صورت عددی بین صفر تا ۱۰۲۳ نشان می‌دهد. در برنامه میکروکنترلر AVR، مقادیر این ADCها خوانده شده و براساس آنها تصمیم گرفته می‌شود. متغیرهای left، middle و right به ترتیب مقادیر ADCهای متصل به حسگر چپ، وسط و راست روبات را در خود نگهداری می‌کند. اگر حسگر وسط روی خط باشد (middle > m_LTP) و دو حسگر چپ و راست خارج خط باشند (مقدار ADC نظیر آنها از side_TP کمتر باشد)، روبات به مسیر مستقیم ادامه می‌دهد. اگر یکی از حسگرها وارد خط شود (مقدار ADC نظیر آن از side_TP بیشتر باشد)، روبات به سمت آن حسگر خواهد پیچید. تنظیمات حسگرها (زاویه، پوشش، مقادیر مقاومتها و ...) را در محیط مسابقه یا جایی که روبات باید عمل کند، انجام دهید.
- از LEDهایی که به پایه‌های PD5 تا PD7 متصل شده‌اند، برای نظارت بر عملکرد حسگرها استفاده شده است؛ هرکدام از این LEDها مربوط به یک حسگر است و میکروکنترلر طوری برنامه‌ریزی شده است که با ورود هر حسگر به خط، LED مربوط به آن روشن می‌شود. با مشاهده حالت این LEDها می‌توانید از صحت عملکرد حسگرها مطمئن شوید.
- گسترش این مدار بسیار ساده است. می‌توانید تعداد حسگرها را تا شش عدد افزایش دهید. کافی است مدارات حسگر را مطابق شکل ببندید و خروجی آنها را به ADCهای دیگر میکروکنترلر AVR متصل کنید و در برنامه اصلاحات لازم را انجام دهید. در صورتی که تعداد کانال‌های ADC میکروکنترلر برای پوشش دادن همه حسگرها کافی نبود، می‌توانید حسگرها را به کمک یک مالتی‌پلکسر آنالوگ (مثلاً تراشه ۴۰۵۱) به یک کانال ADC وصل کنید و با مقداردهی خطوط انتخاب مالتی‌پلکسر و برنامه‌ریزی مناسب، همه حسگرها را با یک کانال ADC مدیریت کنید.
- جهت برنامه ریزی میکروکنترلر باید از دستگاه برنامه‌ریز (پروگرامر) استفاده کنید. اگر این دستگاه را ندارید می‌توانید به راحتی آن را به کمک یک کانکتور پورت موازی (DB-25) و پنج رشته سیم به طول ۱/۵ متر بسازید. نحوه اتصال سیمها به پایه‌های کانکتور DB-25 در بخش پیوست مشخص شده است. طرف دیگر این ۵ رشته را به ترتیبی که در آن بخش مشخص شده (STK200/300) به یک سوکت پنج‌تایی متصل کنید. با اتصال این ۵ رشته به پینهای برنامه‌ریزی میکروکنترلر (پایه‌های GND، Reset، SCK، MOSI و MISO)، روبات آماده برنامه ریزی است.

قطعات لازم برای ساخت این روبات

- ۳ فرستنده/گیرنده مادون قرمز (یا قطعه آماده CNY70)
- ۶ مقاومت ۲۲۰ اهم
- ۳ مقاومت ۱۰۰ کیلواهم
- ۱ مقاومت ۱۰ کیلواهم
- یک کلید فشاری
- ۳ عدد LED معمولی
- یک میکروکنترلر ATMEGA8 با سوکت مناسب
- ۱ عدد خازن ۱۰۰۰ میکروفاراد پلی‌استر
- ۳ عدد خازن ۱۰۰ نانوفاراد الکترولیت
- ۱ عدد دیود 1N4148
- ۱ عدد پل دیودی (یا ۸ عدد دیود 1N4148 که به حالت پل بسته شوند)
- یک رگولاتور 7805
- دو موتور DC با چرخش ۵۰۰ دور در دقیقه (ترجیحاً گیربکس‌دار)
- تراشه L298 با سوکت مناسب

توجه کنید فرض ما بر استفاده از کریستال داخلی میکروکنترلر است که باید در CodeVision تنظیمات آن را انجام دهید. اگر مدار را در CodeVision طوری تنظیم کنید که از کریستال خارجی استفاده کند به یک کریستال ۱۲ مگاهرتز و دو خازن عدسی ۲۲ پیکوفاراد نیز نیاز دارید.

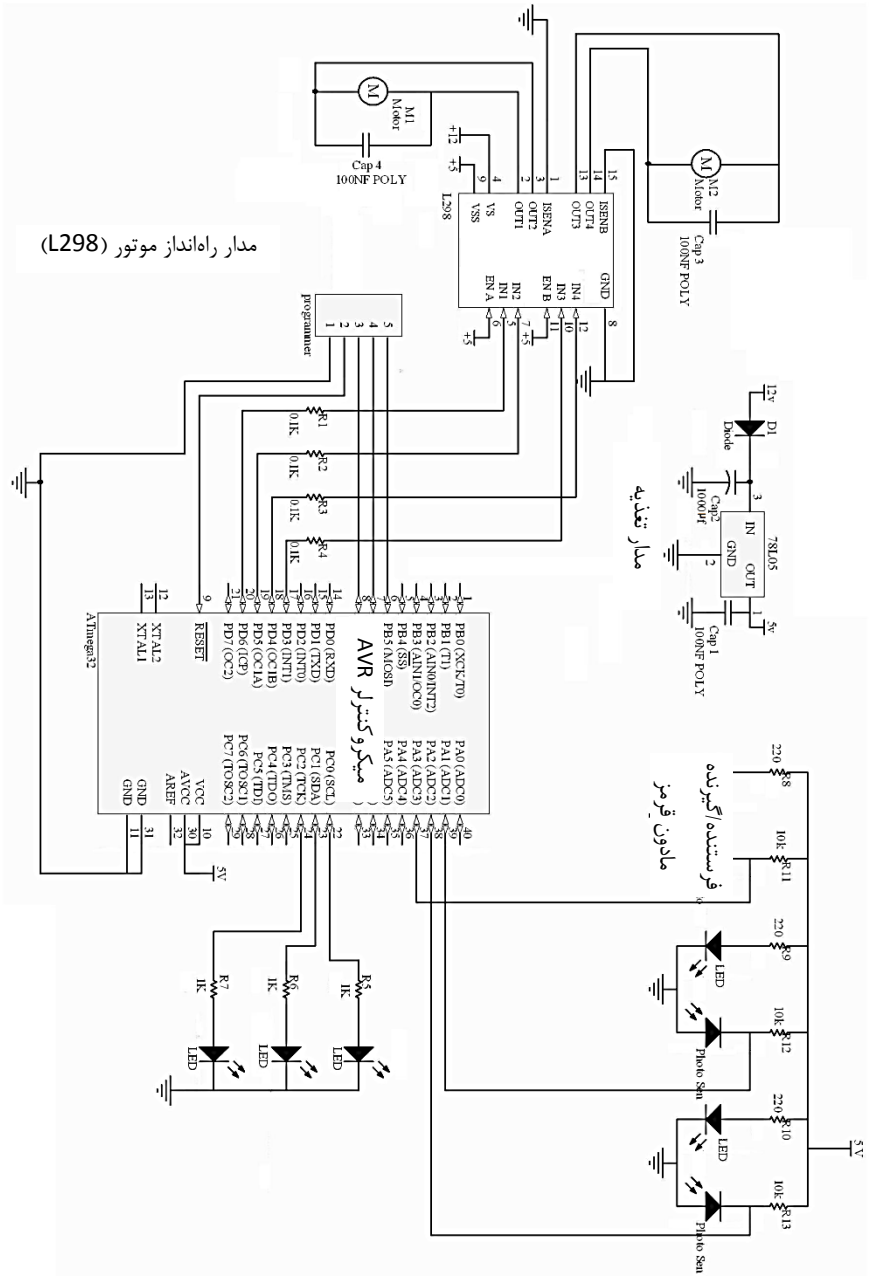
پرسش) با مشاهده رشته‌های turn_right، turn_left و go_ahead مشخص است که این روبات برای چرخیدن به یک سمت، موتور آن سمت را خاموش می‌کند و موتور سمت مقابل همچنان به کار خودش ادامه می‌دهد. نشان دهید با تغییر رشته‌های فوق به صورت زیر، هنگام چرخش به یک سمت موتور آن سمت به شکل معکوس می‌چرخد:

```
#define   go_ahead   0b00001010
#define   turn_left     0b00001001
#define   turn_right   0b00000110
```

پرسش) با تغییر مقادیر m_LTP و m_UTP و side_TP چه اتفاقی می‌افتد؟ تحت چه شرایطی ممکن است به این کار نیاز داشته باشیم؟

مدار کامل روبات مسیریاب با میکروکنترلر (با PWM)

در مدار بالا، موتورها تنها دارای دو حالت حرکت یا توقف هستند؛ این در حالی است که با توجه به بازه وسیع خروجی ADC می‌توان سرعت موتور را متناسب با آن تنظیم کرد؛ مثلاً وقتی خروجی ADC به صورت ناگهانی زیاد شود، می‌توان نتیجه گرفت روبات وارد یک پیچ تند شده و باید به آهستگی بپیچد تا منحرف نشود.



به بهانه توضیح PWM یک مدار کامل روبات مسیریاب دیگر را معرفی می‌کنیم که از نظر اصول طراحی بسیار شبیه مدار قبلی است؛ فقط می‌تواند از قابلیت PWM استفاده کند. شکل این مدار در صفحه قبل رسم شده است.

نرم‌افزار میکروکنترلر

برنامه میکروکنترلر AVR برای کنترل این روبات که در CodeVision قابل کامپایل است، به صورت زیر می‌باشد. در ابتدا برنامه ساده (که تنها دو حالت خاموش یا روشن را برای موتور در نظر می‌گیرد) را بررسی می‌کنیم.

```
#include      <mega16.h>
#include      <delay.h>

#define       MotorLeft      PORTD.6
#define       MotorRight     PORTD.3

#define       SpeedLeft      OCR1A
#define       SpeedRight     OCR1B

#define       led1    PORTC.0
#define       led2    PORTC.1
#define       led3    PORTC.2

#define       ADC_VREF_TYPE      0x40

int   center, left, right, TriggerPoint  = 700;

// Read the ADC conversion result
unsigned   int   read_adc(unsigned char   adc_input){
        ADMUX = adc_input | (ADC_VREF_TYPE & 0xff);
        // Delay needed for the stabilization of the ADC input voltage
        delay_us(10);
        // Start the AD conversion
        ADCSRA |= 0x40;
        // Wait for the AD conversion to complete
        while ((ADCSRA & 0x10) == 0);
        ADCSRA |= 0x10;
        return  ADCW;
}

void main(void) {

//*************** Input/Output Ports initialization*************************
// Port A initialization
PORTA=0x00;
DDRA=0x00;

// Port B initialization
PORTB=0x00;
DDRB=0x00;
```

```
// Port C initialization
PORTC=0x00;
DDRC=0x07;

// Port D initialization
PORTD=0x00;
DDRD=0x78;

//***************** Timer/Counter 1(PWM) initialization**********************
// Clock source: System Clock
// Clock value: 7.813 kHz
// Mode: Fast PWM top=00FFh
// OC1A output: Non-Inv.
// OC1B output: Non-Inv.
// 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=0xA1;
TCCR1B=0x0D;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

//**********************hardware initialization************
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

MCUCR=0x00;
MCUCSR=0x00;
TIMSK=0x00;
ACSR=0x80;
SFIOR=0x00;

// **********************ADC initialization******************

// ADC Clock frequency: 62.500 kHz
// ADC Voltage Reference: AVCC pin
// ADC Auto Trigger Source: None
ADMUX = ADC_VREF_TYPE & 0xff;
ADCSRA = 0x87;
```

```
 MotorRight  =  0;  // Not Reverse
 MotorLeft    = 0;   // Not Reverse

while (1) {
    //********read sensor*****************
        left = read_adc(1);
        delay_ms(10);
        center = read_adc(2);
        delay_ms(10);
        right = read_adc(3);
        delay_ms(10);

    //************center sensor*************
    if (center > TriggerPoint)
    {
                SpeedLeft = 255;
                SpeedRight = 255;
                led2 = 1;
    }
    else
                led2 = 0;

    //************left sensor*************
    if (left > TriggerPoint)
    {
                SpeedLeft = 0;
                SpeedRight = 255;
                led1 = 1;
    }
    else
                led1 = 0;

    //************right sensor*************
    if (right > TriggerPoint)
    {
                SpeedLeft = 255;
                SpeedRight = 0;
                led3 = 1;
    }
    else
                led3 = 0;
   }
}
```

در این برنامه از عدد ۷۰۰ به عنوان سطح سوییچ استفاده شده است؛ اگر خروجی ADC بیش از ۷۰۰ باشد به معنای ورود حسگر متناظر به خط مشکی است.

در برنامه میکروکنترلر، متغیرهای SpeedRight و SpeedLeft (که متناظر با ثباتهای OCR1A و OCR1B هستند) برای مقداردهی PWM به کار می‌روند. در برنامه دیده می‌شود که این متغیرها فقط با مقادیر صفر و ۲۵۵ مقداردهی شده‌اند؛ به

همین جهت فقط می‌توان موتور را متوقف کرد یا با حداکثر سرعت به چرخش درآورد. اگر برای مقداردهی این متغیرها از عددی بین صفر تا ۲۵۵ استفاده کنید، موتور با سرعتی متناسب با این عدد خواهد چرخید. با استفاده از حسگرهای بیشتر و برنامه‌ریزی PWM میکروکنترلر، می‌توانید کنترل دقیقتری روی رفتار روبات داشته باشید.

اگر متغیرهای MotorRight و MotorLeft را به جای «صفر» با «یک» مقداردهی کنید، می‌توانید موتور را درجهت معکوس به چرخش درآورید. اگر در این حالت به ثباتهای PWM میکروکنترلر AVR (OCR1A و OCR1B که متناظر با متغیرهای SpeedRight و SpeedLeft در برنامه ما هستند) عدد n – 255 را ارسال کنید، هرچه n بیشتر باشد سرعت چرخش موتور در جهت معکوس بیشتر می‌شود.