پیش‌گزارش دستورکار اول آزمایشگاه ریزپردازنده و زبان اسمبلی

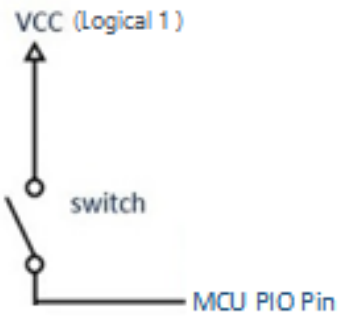
نگار موقتیان، ۹۸۳۱۰۶۲

۱. با توجه به درس سیستم عامل، تفاوت روش‌های سرکشی و وقفه محور را بیان کنید:

روش سرکشی یک روش نرم‌افزاری است که در آن پردازنده دائماً در حال بررسی این است که برای مثال آیا دستگاه IO نیاز به پردازش CPU دارد یا خیر، تا در صورت نیاز به آن پاسخ مناسب را بدهد. در مقابل روش وقفه محور روشی سخت‌افزاری است که در آن دستگاه IO در صورت تغییر وضعیت، خود توسط سیگنالی به نام وقفه و یا interrupt به پردازنده خبر می‌دهد که نیاز به پردازش آن دارد. پردازنده با دریافت وقفه پردازش برنامه‌ای که در حال اجرای آن است را متوقف کرده و قطعه کدی به نام Interrupt Service Routine و یا به اختصار ISR که خاص آن وقفه است را اجرا می‌کند.

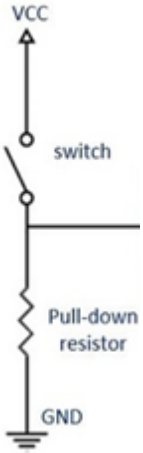
مزیت روش وقفه محور نسبت به روش سرکشی این است که در روش وقفه محور CPU تنها زمانی که interrupt را دریافت می‌کند مشغول به کارهای IO می‌شود، اما در روش سرکشی دائم درگیر بررسی این است که آیا هر یک از دستگاه‌های IO به آن نیاز دارد یا خیر و این کار پردازنده را مشغول نگه می‌دارد.

۲. چرا این روش برای فهمیدن اینکه چه زمانی کلید بسته شده درست نیست؟ در این مدار پایه میکرو در چه حالتی می‌باشد؟

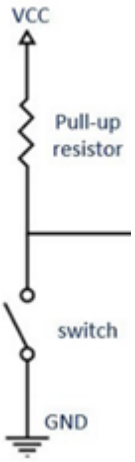


زمانی که کلید بسته باشد پایهٔ میکرو مقدار یک منطقی به خود خواهد گرفت (که البته در این زمان نیز بهتر است یک مقاومت با این پایه سری کنیم تا جریان زیادی ایجاد نشود)، اما زمانی که کلید باز باشد انتهای پایهٔ میکرو به صفر و یا یک منطقی متصل نیست و مقداری نامشخص به خود می‌گیرد. این مقدار نامشخص معادل مقداری که انتظار آن را داشتیم یعنی صفر منطقی نیست، پس این روش درست نیست.

۳. درباره چگونگی کارکرد مدارهای بالا توضیح دهید. به چه دلیل نیاز به مقاومت (Pull-up/Pull-down داریم؟



در این مدار که به صورت Pull-down بسته شده‌است زمانی که کلید بسته باشد ولتاژ پایهٔ میکرو برابر با یک منطقی، و زمانی که کلید باز باشد برابر با صفر منطقی خواهد بود. وجود مقاومت Pull-down در این مدار ضروری است زیرا بدون آن زمانی که کلید بسته باشد یک طرف سیم عمودی در این مدار ولتاژ VCC و یک طرف آن ولتاژ زمین را خواهد داشت. در نتیجه اتصال کوتاه برقرار شده و اجزاء مدار آسیب می‌بینند.



کارکرد این مدار مانند مدار قبل است، با این تفاوت که مقاومت آن به صورت Pull-up بسته شده‌است و در نتیجه زمانی که کلید بسته باشد ولتاژ پایهٔ میکرو برابر با صفر منطقی، و زمانی که کلید باز باشد برابر با یک منطقی خواهد بود. درست مانند مدار قبل وجود مقاومت Pull-up در این مدار نیز برای جلوگیری از اتصال کوتاه زمان بسته شدن کلید ضروری است.

۴. آیا رخ دادن یک اتفاق در صورت اعلام شدن (Assertion) لزوما منجر به اجرای روال سرویس وقفه متناظر با آن می‌شود؟

خیر؛ در مواقعی با توجه به الویت وقفه‌هایی که به پردازنده می‌روند و نحوهٔ رسیدگی به آن‌ها ممکن است ISR مربوط به آن وقفه اجرا نشود و یا با تاخیر پس از انجام وقفه‌های با الویت بالاتر اجرا شود.

۵. پایه‌های وقفه در برد ATmega 2560 و شیوه پیاده سازی وقفه ورودی را بدست آورید.

در این مدار پایه‌های 2، 3، 18، 19، 20 و 21 می‌توانند برای پیاده‌سازی وقفه‌ها به کار روند (که با توجه به توضیحات مربوطه پایه‌های 20 و 21 زمانی که برای ارتباط I2C به کار می‌روند دیگر قابل استفاده به عنوان پایه‌های وقفه نیستند).

برای پیاده‌سازی وقفه‌ها در این برد می‌توان از دستور زیر استفاده کرد:

```
attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)
```

که در آن pin شماره پایه‌ای است که وقفه به آن مربوط می‌شود، ISR تابعی است که در صورت آمدن وقفه به پایهٔ pin اجرا می‌شود و mode مشخص کنندهٔ این است که در چه حالتی از پایهٔ pin وقفه باید اعلام شود (انواع این mode ها در سوال ۷ بررسی شده‌اند).

۶. اگر بخواهیم در زمان تغییر مقدار پایه، وقفه فعال شود از چه mode ای درون تابع attachInterrupt استفاده می‌شود؟ CHANGE

۷. انواع اتفاق‌های ورودی را که واحد GPIO در برد آردوینو ATmega 2560 می‌تواند رخ دادن آن‌ها را بفهمد و اعلام کند بنویسید.

- LOW: زمانی که مقدار پایهٔ وقفه برابر با صفر منطقی می‌باشد.
- CHANGE: زمانی که مقدار پایهٔ وقفه تغییر می‌کند.
- RISING: زمانی که مقدار پایهٔ وقفه از صفر منطقی به یک منطقی تغییر می‌کند.
- FALLING: زمانی که مقدار پایهٔ وقفه از یک منطقی به صفر منطقی تغییر می‌کند.
- HIGH: زمانی که مقدار پایهٔ وقفه برابر با یک منطقی می‌باشد.

پیاده سازی با استفاده از روش سرکشی

۱. اگر دکمه را در حالت فشرده برای زمان طولانی نگه داریم چه اتفاقی خواهد افتاد؟ آیا با منطق کارکرد خواسته شده سازگار است؟ چه راه حلی برای این مشکل (در صورت وجود) می‌توان پیشنهاد کرد؟

در روش سرکشی اگر دکمه را در حالت فشرده نگه داریم کارکرد مربوط به تابع تا زمانی که دکمه فشرده است تکرار می‌شود. برای مثال با پایین نگه داشتن دکمهٔ ۲ مشاهده می‌شود که LED ها بیش از تعداد حروف اسم چشمک می‌زنند (بسته به مدت زمان فشرده نگه داشتند دکمه به تعداد مضرب صحیحی از طول اسم). این کارکرد مورد نظر مدار نیست، زیرا در دستورکار اشاره شده که با هر بار فشردن دکمه اتفاقی بیفتد اما در مدار موجود با فشرده نگه داشتن دکمه نیز این اتفاق می‌افتد. دلیل آن نیز این است که تنها صفر بودن پایهٔ ورودی بررسی شده و نه صفر شدن آن.

برای حل این مشکل می‌توان برای هر پایهٔ ورودی یک متغیر flag قرار داد. زمانی که دکمه فشرده نشده این پرچم مقدار false دارد. زمانی که مقدار صفر را از پایهٔ مورد نظر دریافت می‌کنیم در صورتی که مقدار پرچم مربوط به آن پایه مقدار false داشت بدین معناست که دکمه برای اولین بار مقدار صفر به خود گرفته و روال مربوط به آن دکمه انجام می‌شود. به علاوه مقدار پرچم مربوطه را true می‌کنیم تا دیگر با مشاهدهٔ صفر بودن پایه روال آن انجام نشود. سپس با دریافت مقدار یک از دکمه دوباره مقدار پرچم را false می‌کنیم تا در صورتی که دوباره دکمه فشرده شد آن را تشخیص دهیم.

۲. فرض کنید می‌خواهیم برد مورد نظر علاوه بر فراهم کردن کارکرد خواسته شده در بالا، عمل دیگری را نیز به صورت زمان دار انجام دهد. روشی برای افزودن این کارکرد تازه به برنامه پیشنهاد دهید.

یک راه ساده استفاده از دو کلاک مجزا برای این کار است (در صورت امکان). راه دیگر این است که به جای استفاده از delay از شرط‌ها استفاده کنیم. برای این کار زمان را در هر لحظه دریافت می‌کنیم (برای مثال با استفاده از تابع millis() می‌توان مدت زمانی که از شروع به کار برد گذشته را دریافت کرد). سپس به ازای هر یک از کارهایی که باید انجام شود بررسی می‌کنیم که از بار آخری که کار مذکور انجام شده مقدار مورد نظر گذشته است یا خیر (در این مثال ۵ ثانیه). اگر زمان مورد نظر سپری شده بود روال مربوطه را انجام می‌دهیم (در این مثال وضعیت LED را تغییر می‌دهیم) و آخرین زمانی که کار در آن انجام شده را آپدیت می‌کنیم. از آن جایی

که در این روش از delay استفاده نمی‌کنیم و تنها نیاز به بررسی یک شرط داریم این کار مداخله‌ای در روند کارهای دیگری که باید انجام شوند ایجاد نمی‌کند.

۳. **فرض کنید میخواهیم کارکرد دیگری را به دستگاه اضافه کنیم به این صورت که در صورت یک شدن یک پایه عملیات مشخصی را به عنوان پاسخ انجام دهد (محدودیت زمانی برای پاسخ دادن وجود دارد). هیچ یک از اتفاق‌های یک شدن پایه نباید از دست برود (بی پاسخ بماند). و یک شدن پایه نیز در هر زمانی ممکن است رخ دهد. آیا برنامه شما -که به روش سرکشی واحدهای جانبی را بررسی می‌کند- می‌تواند در هر شرایطی (مثلا هنگام فشرده شدن کلید) این کارکرد را فراهم کند؟**

خیر؛ به دلیل استفاده از delay در برنامه اگر در حال اجرای روال مربوط به یکی از پایه‌ها باشیم یک شدن پایه از دست می‌رود و متوجه آن نمی‌شویم زیرا در آن لحظه شرطی برای بررسی آن وجود ندارد. البته می‌توان با استفاده از روشی که در قسمت قبل ذکر شد به گونه‌ای این مشکل را برطرف کرد اما پیاده‌سازی آن پیچیدگی نسبتاً بالایی دارد.

۴. **فرض کنید به دلیل محدودیت در توان مصرفی می‌خواهیم پردازنده در هنگام بیکاری به خواب برود. در زمان خواب پردازنده هیچ دستوری را اجرا نمی‌کند. روش سرکشی چه قدر با این نیازمندی سازگاری دارد؟ آیا می‌توان با این روش هم به خواب رفت و هم کارکرد درست آزمایش را فراهم کرد؟**

خیر؛ در روش سرکشی اگر پردازنده به خواب برود فشرده شدن دکمه‌ها را از دست می‌دهد. وظیفهٔ بررسی وضعیت پایه‌های ورودی در هر لحظه به عهدهٔ پردازنده است، پس اگر پردازنده به خواب برود در صورت تغییر وضعیت ورودی‌ها متوجه آن نشده و مدار هیچ عملی انجام نمی‌دهد. بنابراین نمی‌توان کارکرد مورد انتظار از مدار را فراهم کرد.

پیاده سازی با استفاده از روش وقفه محور

۱. **اگر دکمه را در حالت فشرده برای زمان طولانی نگه داریم چه اتفاقی خواهد افتاد؟ آیا با منطق کارکرد خواسته شده سازگار است؟ چه راه حلی برای این مشکل (در صورت وجود) می‌توان پیشنهاد کرد؟**

روال مورد نظر برای دکمه تنها یکبار اجرا می‌شود. به راحتی می‌توان زمان مشخص کردن پایهٔ وقفه از حالت FALLING استفاده کرد، در این صورت تنها زمانی که دکمه فشرده می‌شود ISR مربوطه اجرا می‌شود و با نگه داشتن دکمه اتفاقی نمی‌افتد. این کارکرد با کارکرد مورد انتظار از مدار سازگار است.

۲. **فرض کنید می‌خواهیم برد مورد نظر علاوه بر فراهم کردن کارکرد خواسته شده در بالا، عمل دیگری را نیز به صورت زمان دار انجام دهد. روشی برای افزودن این کارکرد تازه به برنامه پیشنهاد دهید.**

در روش وقفه محور نیز نیاز به delay داریم، لذا می‌توان از همان روشی که برای مدار پیاده‌سازی شده به روش سرکشی بیان شد استفاده کرد. یا می‌توان هر ۵ ثانیه یک وقفه با الویت بالا ایجاد کرد تا وضعیت LED را تغییر دهد.

۳. **فرض کنید میخواهیم کارکرد دیگری را به دستگاه اضافه کنیم به این صورت که در صورت یک شدن یک پایه عملیات مشخصی را به عنوان پاسخ انجام دهد (محدودیت زمانی برای پاسخ دادن وجود دارد). هیچ یک از اتفاق‌های یک شدن پایه نباید از دست برود (بی پاسخ بماند). و یک شدن پایه نیز در هر زمانی ممکن است رخ دهد. آیا برنامه شما -که به روش سرکشی واحدهای جانبی را بررسی می‌کند- می‌تواند در هر شرایطی (مثلا هنگام فشرده شدن کلید) این کارکرد را فراهم کند؟**

بله؛ اینبار این کار به سادگی انجام می‌شود. در صورتی که پردازنده چندین وقفه دریافت کند به تمام آن‌ها بر اساس زمان ایجاد وقفه و یا الویت آن‌ها رسیدگی می‌کند و ISR مربوطه را اجرا می‌کند. بنابراین کافیست پایهٔ مورد نظر را به یکی از پایه‌های وقفه متصل کنیم و به آن الویت بالایی بدهیم. با این کار همواره مطمئن هستیم هر یک از یک شدن‌های پایه به پردازنده اعلام می‌شود و به دلیل الویت بالای وقفه پردازنده می‌تواند در زمان مناسب به آن پاسخ مناسب دهد.

۴. **فرض کنید به دلیل محدودیت در توان مصرفی می‌خواهیم پردازنده در هنگام بیکاری به خواب برود. در زمان خواب پردازنده هیچ دستوری را اجرا نمی‌کند. روش سرکشی چه قدر با این نیازمندی سازگاری دارد؟ آیا می‌توان با این روش هم به خواب رفت و هم کارکرد درست آزمایش را فراهم کرد؟**

بله؛ از آن‌جایی که در روش وقفه محور اعلام نیاز به پردازنده توسط خود دستگاه IO -و نه پردازنده- انجام می‌شود پردازنده می‌تواند تا زمان دریافت وقفه به خواب رود و پس از دریافت آن به کارهای خود ادامه دهد. در این صورت مدار کارکرد درستی خواهد داشت و در مصرف توان پردازنده صرفه جویی می‌شود.