**پیش­گزارش دستورکار اول آزمایشگاه ریزپردازنده و زبان اسمبلی**

نگار موقتیان، 9831062

1. **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 دائماٌ 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.

1. **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 پایۀ 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 پایۀ 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.

1. **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 پایۀ 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 پایۀ 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.

1. **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 نحوۀ 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.

1. **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 پایۀ pin اجرا می­شود و mode مشخص کنندۀ این است که در چه حالتی از پایۀ pin وقفه باید اعلام شود (انواع این mode ها در سوال 7 بررسی شده­اند).

1. **اگر بخواهیم در زمان تغییر مقدار پایه، وقفه فعال شود از چه mode ای درون تابع attachInterrupt استفاده می­شود؟** CHANGE
2. **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 پایۀ وقفه برابر با صفر منطقی می­باشد.
* CHANGE: زمانی که مقدار پایۀ وقفه تغییر می­کند.
* RISING: زمانی که مقدار پایۀ وقفه از صفر منطقی به یک منطقی تغییر می­کند.
* FALLING: زمانی که مقدار پایۀ وقفه از یک منطقی به صفر منطقی تغییر می­کند.
* HIGH: زمانی که مقدار پایۀ وقفه برابر با یک منطقی می­باشد.

**پیاده سازی با استفاده از روش سرکشی**

1. **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 دکمۀ 2 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 پایۀ ورودی بررسی شده و نه صفر شدن آن.

برای حل این مشکل می­توان برای هر پایۀ ورودی یک متغیر flag قرار داد. زمانی که دکمه فشرده نشده این پرچم مقدار false دارد. زمانی که مقدار صفر را از پایۀ 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 مشاهدۀ 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.

1. **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 5 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.

1. **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 نسبتاٌ بالایی دارد.

1. **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. وظیفۀ بررسی وضعیت پایه­های ورودی در هر لحظه به عهدۀ 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**

1. **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 پایۀ 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.

1. **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 5 ثانیه یک وقفه با الویت بالا ایجاد کرد تا وضعیت LED را تغییر دهد.

1. **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 پایۀ 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.

1. **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.