پیش‌گزارش دستورکار هفتم آزمایشگاه ریزپردازنده و زبان اسمبلی

نگار موقتیان، مریم موسوی – گروه ۳

۱. کاربرهای EEPROM و تفاوت آن با حافظه‌های RAM و Flash

حافظهٔ EEPROM برخلاف حافظهٔ RAM یک حافظهٔ دائمی می‌باشد، به این معنا که با قطع شدن تغذیهٔ آن و خاموش شدن مدار اطلاعات داخل آن باقی می‌ماند. حافظهٔ Flash نیز یک حافظهٔ دائمی می‌باشد، اما مزیت EPPROM نسبت به آن این است که نوشتن داده‌ها می‌تواند بایت به بایت انجام شود، در حالی که در حافظه‌های Flash تنها می‌توانیم داده‌ها را به صورت بلوکی نوشته و بخوانیم (و به همین دلیل از آن‌ها برای نگهداری برنامهٔ میکروکنترلر استفاده می‌شود که آن را یکباره نوشته و یکباره پاک می‌کنیم). به علاوه تعداد دفعاتی که می‌توانیم بر روی این حافظه‌ها داده بنویسیم از حافظهٔ Flash بیش‌تر است.

در نتیجه زمانی که می‌خواهیم داده‌هایی مانند تنظیمات کاربر را زمان اجرا ذخیره کنیم، اما بتوانیم پس از قطع شدن تغذیهٔ مدار نیز دوباره به آن‌ها دسترسی داشته باشیم از EEPROM استفاده می‌کنیم.

۲. فرآیند نوشتن در حافظه‌های Flash

همانطور که در قسمت قبل نیز اشاره شد نوشتن بر روی حافظه‌های Flash باید بلوک به بلوک انجام شود (در این نوع از حافظه‌ها به طور معمول از بلوک‌های 4KB ای استفاده می‌شود). برای نوشتن یک بایت بر روی این حافظه اگر مقدار روی حافظه یک باشد می‌توانیم آن را به صفر تغییر دهیم اما بالعکس این کار امکان پذیر نیست و باید کل بلوک حاوی داده را پاک کرده و آن را دوباره بنویسیم.

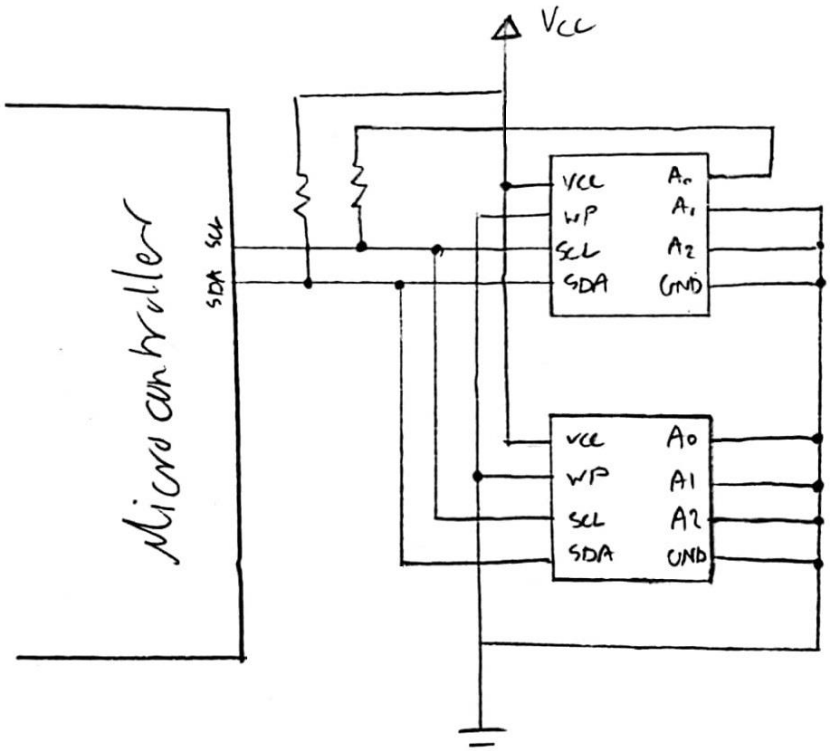
برای اینکه دیگر داده‌های قبلی نوشته شده بر روی حافظه طی این فرآیند از بین نرود می‌توانیم کل بلوک حافظه را خوانده و در یک بافر ذخیره کنیم، سپس دادهٔ دلخواه را بر روی این بافر تغییر داده و دوباره کل محتوای بافر را بر روی بلوک فوق بنویسیم. بدیهیست این فرآیند زمانبر بوده و کارا نیست، بنابراین برای ذخیره‌سازی چنین داده‌هایی بهتر است از حافظه‌های EEPROM استفاده کنیم.

۳. اگر یک حافظهٔ EEPROM بیرونی دارای 4KB حافظه و ۲ پایهٔ آدرس باشد، در این صورت می‌توان حداکثر چند KB حافظهٔ EEPROM بیرونی بر روی یک باس مشترک داشت؟

برای این که این حافظه‌های بیرونی بتوانند بر روی یک باس مشترک کار کنند باید آدرسی یکتا داشته باشند تا میکروکنترلر بتواند در هر لحظه مشخص کند بر روی کدام یک می‌خواهد داده بنویسد و یا از کدام یک داده بخواند. از آن‌جایی که حافظهٔ فوق ۲ پایهٔ آدرس دارد، می‌توان ۴ آدرس یکتا به آن تخصیص داد، بنابراین می‌توانیم ۴ تا از این حافظه‌ها را به میکروکنترلر متصل کنیم. در نتیجه در نهایت 16KB حافظهٔ EEPROM بیرونی خواهیم داشت.

۴. اتصال دو AT24C02 به میکروکنترلر توسط یک باس مشترک

به یکی از حافظه‌های AT24C02 آدرس 000 و به دیگری آدرس 001 می‌دهیم، از این طریق میکروکنترلر می‌تواند مشخص کند که بر روی کدام یک از حافظه‌ها می‌خواهد داده بنویسد و یا از کدام یک می‌خواهد داده بخواند.



۵. همخوانی دنبالهٔ فریم‌های خواندن از حافظه و پروتکل TWI

در پروتکل TWI برای شروع ارتباط یک سیگنال START توسط master فرستاده شده و برای پایان آن نیز یک سیگنال STOP فرستاده می‌شود. سپس به صورت سریال آدرس slave ای که master می‌خواهد با آن ارتباط برقرار کند بر روی باس داده قرار می‌گیرد. پس از آن یک تک بیت فرستاده می‌شود که مشخص می‌کند master می‌خواهد داده‌ای را بخواند یا بنویسد (مقدار صفر برای نوشتن و یک برای خواندن). حال slave یک بیت ACK می‌فرستد. سپس داده‌ها انتقال می‌یابند. زمان نوشتن master به ازای هر بایت یک ACK دریافت می‌کند تا زمانی که بخواهد به ارتباط خاتمه دهد، زمان خواندن نیز slave داده‌ها را به master ارسال کرده و master به ازای هر بایت داده یک ACK می‌فرستد، جز برای بایت آخر که نفرستادن ACK آن به معنای این است که master دیگر نمی‌خواهد داده‌ای بخواند.

برای نوشتن و خواندن داده از روی حافظهٔ EEPROM فوق نیز روندی مشابه طی می‌شود.

زمان نوشتن ابتدا master ارتباط را شروع کرده، آدرس حافظهٔ مورد نظر را مشخص کرده، و با فرستادن صفر مشخص می‌کند می‌خواهد بر روی slave داده بنویسد. سپس یک ACK از slave دریافت کرده و آدرس دادهٔ مورد نظر را بر روی باس قرار می‌دهد. پس از دریافت ACK آدرس داده را بر روی باس گذاشته و با دریافت ACK بعدی ارتباط را خاتمه می‌دهد.

زمان خواندن نیز ابتدا master ارتباط را شروع کرده، آدرس حافظهٔ مورد نظر را مشخص کرده، و با فرستادن صفر مشخص می‌کند می‌خواهد بر روی slave داده بنویسد. سپس یک ACK از slave دریافت کرده و آدرس دادهٔ مورد نظر را بر روی باس قرار می‌دهد. پس از دریافت ACK ارتباط دیگری را شروع کرده و اینبار پس از مشخص کردن آدرس حافظهٔ مورد نظر مشخص می‌کند که می‌خواهد از روی آن داده‌ای را بخواند. سپس یک ACK ارسال کرده و داده را از slave دریافت می‌کند. این بار اما ACK ارسال نمی‌کند و از این طریق اعلام می‌کند که داده‌ای دیگری را نمی‌خواهد بخواند. سپس با فرستادن سیگنال STOP ارتباط را خاتمه می‌دهد.

۶. اطلاعات مربوط به فرکانس کلاک

فرکانس کلاک همواره توسط master (در این‌جا میکروکنترلری که از آن استفاده می‌کنیم) مشخص شده و توسط master تولید می‌شود.

طبق پروتکل توضیح داده شده نوشتن هر بایت بر حافظه ۲۹ کلاک زمان می‌برد. بنابراین در هر ثانیه می‌توان تقریباً ۳۴۴ بایت بر روی این حافظه نوشت.

البته این مقدار در واقعیت کمتر است، زیرا نوشتن داده‌ها بر روی حافظه نیز زمان می‌برد. با توجه به آن‌چه در دستورکار اشاره شده‌است نوشتن بر حافظه در حداکثر ۵ میلی‌ثانیه انجام می‌شود (و این زمان نسبت به ۲۹ کلاک صرف شده برای برقراری ارتباط، ارسال داده و خاتمهٔ آن بیش‌تر است)، لذا نوشتن هر بایت از داده بر روی حافظه ۵۰ کلاک زمان می‌برد.

۷. توابع کتابخانهٔ Wire

- ()begin: ارتباط master و slave را از طریق پروتکل I2C آغاز می‌کند. اگر پارامتری نداشته باشد دستگاه فوق به عنوان master عمل می‌کند و در غیر این صورت پارامتر وارد شده آدرس slave را تعیین می‌کند.
- ()setClock: برای تعیین فرکانس ارتباط استفاده می‌شود.
- ()beginTransmission: ارتباط را برای شروع ارسال داده به آدرس داده شده آغاز می‌کند.
- ()write: داده را بر روی باس می‌نویسد (با فرض اینکه ارتباط پیش از این برقرار شده‌است).
- ()endTransmission: ارتباط را طبق پروتکل I2C پایان می‌دهد.
- ()requestFrom: برای درخواست خواندن داده از slave توسط master صدا زده می‌شود.
- ()available: تعداد بایت‌هایی که آماده دریافت توسط ()read هستند را می‌دهد.
- ()read: بایت‌های ارسال شده بر روی باس را می‌خواند.

با توجه به این دستورات می‌توان عملیات خواندن و نوشتن از حافظه را مانند زیر انجام داد.

Write:

```
Wire.beginTransmission(DEVICE_ADDRESS);
Wire.write(WORD_ADDRESS);
Wire.write(DATA);
Wire.endTransmission();
```

Read:

```
Wire.beginTransmission(DEVICE_ADDRESS);
Wire.write(WORD_ADDRESS);
Wire.endTransmission();
Wire.requestFrom(DEVICE_ADDRESS, SIZE);
Wire.read();
```