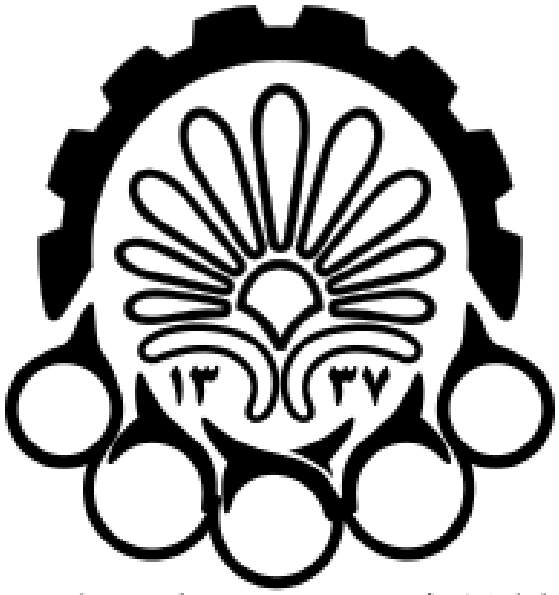


دانشگاه صنعتی امیرکبیر

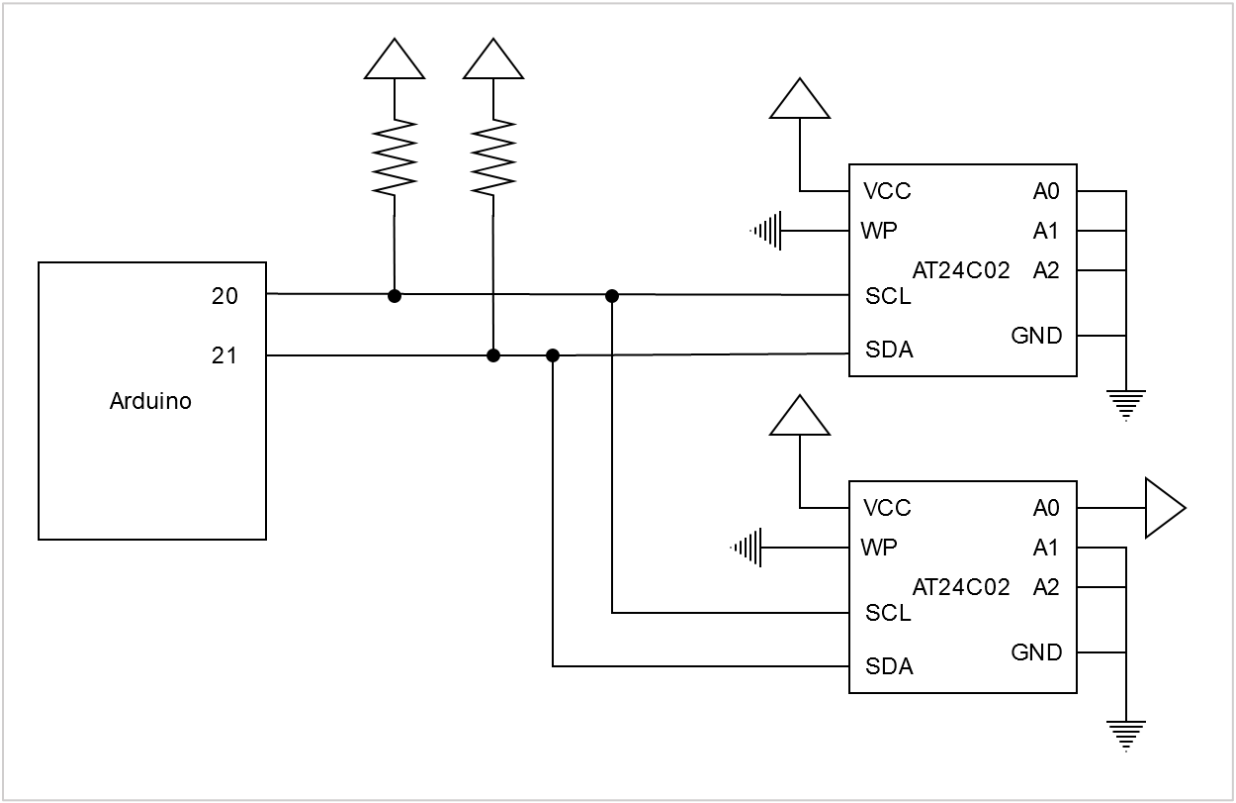
(پلی تکنیک تهران)

آزمایشگاه ریزپردازنده و زبان اسمبلی

رادین شایانفر

پاییز ۱۳۹۹

- **کاربردهای EEPROM و دلیل استفاده به جای RAM و FLASH**: در مواقعی که نیاز است تا داده‌ها پس از خاموش شدن دستگاه و با قطع برق از بین نروند لازم است که از EEPROM استفاده کنیم. دلیل عدم استفاده از RAM نیز همان فرار بودن آن است که پس از هر بار قطعی برق اطلاعات آن پاک می‌شود. هم‌چنین دلیل استفاده نکردن از حافظه FLASH کارکرد آن به صورت block-wise است. در حالی که EEPROM به صورت byte-wise کار می‌کند و می‌توان بایت به بایت (مانند نیاز ما در اینجا) بر روی آن اطلاعات را خواند و تغییر داد. در ضمن سرعت EEPROM نیز از FLASH بیشتر است.
- **نحوه‌ی نوشتن روی FLASHها:** برای این کار از آنجا که عمل نوشتن روی FLASHها به شکل بلاک به بلاک است، می‌توان هر داده را روی بلاک‌های مختلف نوشت. اما این روش چندان کارا نیست. روش دیگر این است که پیش از نوشتن روی یک بلاک، ابتدا آن را بخوانیم، مقدارهای مورد نیاز را تغییر دهیم و سپس مجدد روی همان بلاک در FLASH بنویسیم.
- **حداکثر حافظه روی باس مشترک:** برای این کار به کمک دو پایه‌ی A_0 و A_1 می‌توان 2^2 دستگاه مختلف را آدرس‌دهی کرد و در نتیجه $4 \times 4KB = 16KB$ خواهیم داشت.
- **شماتیک اتصال دو AT24C02 روی باس مشترک:**



شکل (۱) – شماتیک اتصال دو AT24C02 روی باس مشترک

- **هم‌خوانی فریم‌های AT24C02 و TWI**:

فریم write:

Memory: start (1 bit) → device address (7 bits) → W mode (1 bit) → ACK
TWI: start (1 bit) → device (7 bits) → W mode (1 bit) → ACK

→ word address (8 bits) → ACK → data (8 bits) → ACK → ... → stop (1 bit)
→ data (8 bits) → ACK → data (8 bits) → ACK → … → stop (1 bit)

فریم read:

Memory: start (1 bit) → device address (7 bits) → W mode (1 bit) → ACK
TWI: start (1 bit) → device (7 bits) → W mode (1 bit) → ACK

→ word address (8 bits) → ACK → stop (1 bit) → start (1 bit)
→ data (8 bits) → ACK → stop (1 bit) → start (1 bit)

→ device address (7 bits) → R mode (1 bit) → ACK → data (8 bits) → ACK → …
→ device (7 bits) → R mode (1 bit) → ACK → data (8 bits) → ACK → …

→ stop (1 bit)
→ stop (1 bit)

- **فرکانس کلاک**: فرکانس کلاک در برد آردوینو (master) پیکربندی می‌شود و هم‌چنین master مسئولیت تولید این کلاک را به عهده دارد. از آنجا که برای هر بار عمل نوشتن تک بایت نیاز به ۲۹ بیت داده داریم، سرعت نوشتن با کلاک 10 KHz برابر است با:

$$\frac{10 \times 10^3}{29} \sim 344 \ {}^{byte}/_{s}$$

- **توابع Wire**:
  - ()begin: ارتباط master و slave را با I^2C آغاز می‌کند. اگر پارامتری نداشته باشد به عنوان master عمل می‌کند و در غیر این صورت پارامتر وارد شده شماره slave را تعیین می‌کند.
  - ()setClock: برای تغییر فرکانس ارتباط استفاده می‌شود.
  - ()beginTransmission: ارتباط را برای شروع ارسال داده به آدرس داده شده را آغاز می‌کند.
  - ()write: داده را روی slave می‌نویسد (این تابع پس از ()beginTransmission معمولا صدا زده می‌شود).

- endTransmission(): ارتباط را 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(): بایت ارسال شده توسط slave را می‌خواند.

- **کد تولید فریم خواندن و نوشتن:**

Read:

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

Write:

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