بسم الله الرحمن الرحيم

فصل چهارم: دستورات اسمبلی و برخی از تنظیمات اولیه

عناوین مطالب:

- زبان اسمبلی در AVR
- دستورات انتقال داده
- مدهای آدرس دهی
- پورت‌ها و تنظیمات اولیه
- پشته و تنظیمات اولیه
- خواندن و نوشتن در حافظه برنامه
- نام گذاری رجیسترها

* زبان اسمبلی در AVR *

برای برنامه ریزی میکروکنترلرهای AVR، از زبان‌های مختلفی می توان استفاده نمود. زبان‌های برنامه نویسی C، Basic و اسمبلی سه زبان رایج برای برنامه ریزی میکرو کنترلرهای AVR است.

به طور کلی زبان‌های برنامه نویسی به دو دسته سطح بالا و سطح پایین تقسیم می‌شوند. زبان سطح پایین، زبانیست که هر یک از دستورالعمل‌هایش، معادل یک عمل واقعی در پردازنده یا CPU باشد. به عبارت دیگر هر خط از دستورات نوشته شده با یک زبان سطح پایین، معادل یک عملیات سخت افزاری در پردازنده است. اما در زبان های سطح بالا، یک دستورالعمل، می‌تواند معادل چندین عملیات سخت افزاری باشد. هر کدام از این دو سطح، مزایا و معایبی دارند. به عنوان مثال، برنامه نویسی با زبان های سطح بالا ساده تر است چرا که این زبان ها به زبان تکلم ما نزدیک تر هستند اما برنامه نویسی با زبان های سطح پایین قدری دشوارتر است به این دلیل که این زبان ها، از مجموعه‌ای از دستورات قابل فهم برای پردازنده تشکیل شده است و تسلط به آنها نیازمند تأمل بیشتری نسبت به دستورات زبان های سطح بالاست. در مقابل، زبان های سطح پایین به برنامه نویس اجازه می دهند که مستقیماً با پردازنده ارتباط برقرار کند و به آن فرمان دهد. این ویژگی باعث می شود که برنامه نویس از جزئیات عملیات پردازنده آگاه باشد و بتواند برنامه هایی دقیق تر و سریع تر بنویسد و در صورت نیاز به قسمت های مختلف حافظه برنامه و حافظه داده دسترسی داشته باشد.

مشهورترین زبان سطح پایین، زبان اسمبلی است که برای برنامه ریزی پردازنده های گوناگونی مورد استفاده قرار می‌گیرد. توجه داشه باشید که زبان اسمبلی، بسته به پردازنده های مختلف، شکل های مختلفی دارد. مثلاً زبان اسمبلی برای میکروهای AVR با زبان اسمبلی برای برنامه ریزی پردازنده های intel متفاوت است. اما از آنجایی که این تفاوت ها اندک هستند و دستورات، یک قالب کلی واحد دارند، به همه این زبان ها، اسمبلی گفته می شود.

فراگیری زبان اسمبلی، علاوه بر اینکه ما را با فرآیند برنامه نویسی و مقدمات کار عملی با میکرو کنترلرهای AVR آشنا می کند، کمک شایانی به فهم عملکرد CPU و نحوه ارتباط آن با حافظه ها و سایر بخش‌های میکرو می‌کند. به همین دلیل در این فصل به بررسی دستورات اصلی زبان اسمبلی در AVR خواهیم پرداخت.

به صورت کلی دستورات اسمبلی در AVR به ۵ دسته زیر تقسیم می شوند:

- دستورات انتقال داده
- دستورات محاسباتی و منطقی
- دستورات انشعابی
- دستورات بیتی
- دستورات کنترلی

برای مشاهده لیست کامل این دستورات درمیکروکنترلر ATMega32، به پیوست الف مراجعه فرمایید.

* دستورات انتقال داده *

غالب این دستورات، برای انتقال اطلاعات از حافظه داده به رجیسترهای عمومی و یا بالعکس تعبیه شده‌اند. سایر دستورات انتقال داده برای جابجایی اطلاعات میان رجیسترهای عمومی، خواندن از حافظه برنامه و نوشتن در آن مورد استفاده قرار می گیرند. در ادامه، این دستورات را بررسی خواهیم نمود.

دستور LDI

- فرم کلی دستور: LDI Rd,k[1]
- مفهوم دستور: انتقال یک مقدار عددی به یکی از رجیسترهای عمومی (Load Immediate)
- شرح عملیات: Rd<---k
- بررسی عملوندها: R16<Rd<R31 k یک عدد صحیح مثبت است
- نوع آدرس دهی: فوری (شرح مفهوم آدرس دهی، در ادامه همین فصل آمده است)
- توضیحات: به طور کلی دستورات Load، دستوراتی هستند که داده‌ای را به یکی از رجیسترهای عمومی منتقل می‌کنند. در دستور LDI، عدد k به رجیستر مورد نظر منتقل می شود.
- مثال:

LDI R16,0x55[2]

[1] Rd به معنای R destination یا رجیستر مقصد است.

[2] نوشتن عبارت 0x قبل از عدد به این معناست که عدد ذکر شده در مبنای ۱۶ می‌باشد. می‌توان به جای عبارت 0x از علامت $ نیز استفاده نمود. همچنین برای بیان اعداد در مبنای ۲، از عبارت 0b استفاده می کنیم و برای بیان عددی در مبنای ۱۰، خود عدد را بدون نوشتن علامت ذکر می‌نماییم.

پس از نوشتن این دستور، مقدار رجیستر R16 برابر با ۵۵ در مبنای ۱۶ که معادل ۰۱۰۱۰۱۰۱ در مبنای ۲ است، می‌شود.

برای تبدیل اعداد مبنای ۱۶ به مبنای ۲، کافیست که برای هر یک از ارقام عدد مورد نظر، چهار بیت در نظر بگیریم و سپس معادل دودویی (مبنای دو) هر رقم را به جای آن قرار دهیم. به عنوان مثال عدد D7 در مبنای ۱۶ برابر است با 11010111 در مبنای ۲. همان طور که مشاهده می کنید چهار رقم سمت چپ این عدد یعنی 1101، معادل مبنای دو رقم D و چهار رقم سمت راستی عدد، معادل دودویی رقم 7 می باشد.

دستور LDS

- فرم کلی دستور: LDS Rd,k
- مفهوم دستور: انتقال محتویات یکی از خانه های حافظه داده به یکی از رجیسترهای عمومی (Load Direct from Data Space)
- شرح عملیات: Rd<---(k)[۳]
- بررسی عملوندها:

R0<Rd<R31

k آدرس یکی از خانه های حافظه داده است که بسته به حجم حافظه داده می تواند مقادیر مختلفی اختیار کند. مثلاً اگر حجم حافظه داده، معادل ۲ کیلو بایت باشد، k می تواند مقادیر بین 0 تا 800H را اختیار نماید.

- نوع آدرس دهی: مستقیم

*** مدهای آدرس دهی ***

قبل از اینکه به ادامه دستورات بپردازیم، مفهوم آدرس دهی و انواع آن را بررسی می نماییم.
آدرس دهی، نحوه دسترسی CPU به عملوند مورد نظر را مشخص می‌کند.

انواع آدرس دهی عبارتند از:

- آدرس دهی فوری یا Immediate Addressing
- آدرس دهی مستقیم یا Direct Addressing
- آدرس دهی غیر مستقیم یا Indirect Addressing

– در آدرس دهی فوری، عملوند نوشته شده در دستور، مستقیماً به CPU منتقل می شود. به عنوان نمونه، دستور LDI R16,0x20، عدد ۲۰ در مبنای ۱۶ که معادل ۳۲ در مبنای ۱۰ است را به رجیستر R16 منتقل می کند.

[۳] Rd <---(k) یعنی محتویات خانه‌ای از حافظه با آدرس k به رجیستر Rd منتقل شود.

– در دستوراتی که از آدرس دهی مستقیم استفاده می کنند، عملوند، معادل آدرس یکی از خانه های حافظه داده است. CPU با دسترسی به حافظه داده، محتویات آدرس مذکور را به رجیسترهای عمومی یا واحد محاسبه و منطق منتقل می کند و یا محتویات یکی از رجیسترهای عمومی به در آدرس مذکور ذخیره می نماید. برای مثال، دستور LDS یکی از دستوراتیست که از آدرس دهی مستقیم استفاده می کند. فرض کنیم عدد A5 در مبنای ۱۶ در آدرس 250H[۴] از حافظه داده ذخیره شده باشد، پس از نوشتن دستور LDS R10,0x250، عدد A5H به رجیستر R10 منتقل می شود.

– اما در آدرس دهی غیر مستقیم، عملوند یکی از رجیسترهای اشاره گر است. رجیسترهای اشاره گر عبارتند از رجیستر X، رجیستر Y و رجیستر Z. هر کدام از این رجیسترها شامل دو رجیستر هشت بیتی هستند که برای نگه داری آدرس‌های ۱۶ بیتی حافظه داده، مورد استفاده قرار می گیرند. مثلاً رجیستر اشاره گر X، از دو رجیستر هشت بیتی به نام‌های XL و XH تشکیل شده است.[۵] اگر بخواهیم یک آدرس ۱۶ بیتی را در این رجیستر ذخیره کنیم، ۸ بیت پرارزش آدرس را در XH و ۸ بیت کم ارزش را در XL قرار می‌دهیم.
بنابراین رجیسترهای اشاره گر در مجموع از شش رجیستر ۸ بیتی تشکیل شده‌اند. این رجیسترها به ترتیب، شش بایت انتهایی از رجیسترهای عمومی، یعنی R26 تا R31 هستند.[۶]
هنگام نوشتن دستوراتی که از این مد آدرس دهی استفاده می‌کنند، ابتداً آدرس داده مورد نظر را در یکی از رجیسترهای اشاره گر قرار می دهیم. سپس به جای اینکه خود آدرس را مستقیماً در دل دستور بنویسیم، از اشاره گر مقدار دهی شده استفاده می کنیم. به عبارت دیگر زمانی که از دستورات با آدرس دهی غیر مستقیم استفاده می کنیم، توسط این رجیسترها به یکی از خانه های حافظه داده اشاره می کنیم.
دستور LD، یکی از دستوراتیست که با استفاده از آدرس دهی غیر مستقیم، محتویات حافظه داده را به رجیسترهای عمومی منتقل می کند. فرض کنیم در رجیستر R26، عدد 50H و در رجیستر R27، عدد 02H ذخیره شده باشد. با نوشتن دستور LD R16,X، محتویات خانه ۲۵۰ حافظه داده به R16 منتقل می‌شود.

---

[۴] 250H به معنای ۲۵۰ در مبنای هگزا دسیمال (Hex Decimal) یا همان مبنای ۱۶ است.
[۵] H و L، حروف اول کلمات High و Low هستند.
[۶] توجه داشته باشید که این ۶ رجیستر، علاوه بر اینکه می‌توانند مانند سایر رجیسترهای عمومی برای ذخیره سازی اطلاعات مورد استفاده قرار گیرند، می توانند در نقش اشاره گر برای دستوراتی که از آدرس دهی غیر مستقیم استفاده می‌کنند، ظاهر شوند.