10/12/2021

Homework 1

Lec 1-5

MICROPROCESSOR
AND
ASSEMBLY LANGUAGE

Fall 2021

Homework 1

1) به پرسش های زیر در مورد ISA پاسخ دهید:

الف) ISA پردازنده ما باید شامل کدام گروه از فانکشن‌ها باشد تا ISA کاملی به حساب آید؟

جواب:

ISA باید یک سری از حداقل ها را برای ما براورده کند. برای اینکه یک ISA کامل باشد و بتوان یک کامپیوتر با آن ساخت باید سه دسته دستورالعمل داشته باشد.

یک نوع دستور LOAD/STORE برای تعامل با حافظه و خواندن و نوشتن دستورالعمل و داده.

یک نوع دستور CONTROL یا کنترلی که branchهای شرطی و غیر شرطی و if و else ها را شکل میدهند.

یک نوع دستور Arithmetic/logic که محاسبات و عملیات منطقی را انجام میدهند.

ب) ISA چه ویژگی هایی از پردازنده را مشخص می‌کند (حداقل به سه مورد اشاره کنید)؟

مثال: Risc یا Cisc بودن پردازنده

جواب:

ISA مشخص میکند که طول رجیستر های پردازنده چقدر باشد. همچنین تعداد رجیسترهای پردازنده را مشخص میکند. تایپ رجیسترهای پردازنده را نیز مشخص میکند. همچنین با مشخص کردن تعداد بیت آدرس سایر حافظه را مشخص میکند و اینکه چطور به حافظه دسترسی داشته باشیم.

2) به سوالات ریز در رابطه با Microcontrollers پاسخ دهید:

الف) میکرو ای که ما در درس استفاده می کنیم (SAM3X8E) از کدام یک از معماری های Harvardیا Von Neumann استفاده می کند و دلایل آن چیست (دو دلیل)؟

جواب:

از معماری هاروارد استفاده میکند و این معماری بدین صورت است که برای هر کدام از داده و دستورالعمل یک memory جدا در نظر گرفته شده است و واقع دو گذرگاه[1] جدا برای داده و دستورالعمل دارد.

1- امکان همزمان خواندن و نوشتن LOAD/STORE داده و fetch کردن دستورالعمل را به ما میدهد.
2- میتوان از تعداد بیت های متفاوت برای داده و دستورالعمل استفاده کرد و برخلاف معماری فون نیومن نیاز نیست که تعداد بیت های داده و دستور العمل برابر باشند

ب) چند تا از برتری هایی که باعث شده ست در سیستم های نهفته از Microcontroller استفاده شود را نام ببرید (سه مورد کافی است).

1- با توجه به اینکه موری و پردازنده و دستگاه I/O به صورت داخلی[2] به همدیگر متصل شده اند، **حجم کمتری** را نسبت به دستگاه هایی دارند که موری و دستگاه های I/O به صورت خارجی[3] به پردازنده متصل شده اند.
2- با توجه به متصل بودن موری به صورت داخلی، دسترسی به حافظه **سریعتر** میباشد.
3- **آماده سازی راحت تر** بطوریکه میتوان میکروکنترلر با قطعاتی که از قبل آماده شده را خرید و به راحتی آنرا کانفیگ(پروگرم) کرد.
4- **مصرف انرژی کمتری** دارد و برای دستگاه هایی که باتری دارند و مصرف بهینه باتری مهم است، حائز اهمیت میباشد.

به دلیل اینکه دستگاه I/O و موری به صورت داخلی وصل شده اند مصرف انرژی کمتری را دارند.

[1] bus
[2] internally
[3] externally

ج) حلت های مختلف میکرو (SAM3X8E) در Low Power Modes را نام ببرید و برای هر کدام یکی از مواقع استفاده را مثال بزنید.

جواب:

1- Backup Mode: مینیمم مصرف توان انرژی در این حالت میباشد. در این حالت هسته یا پردازنده کاملا خاموش میباشد و در واقع سیستم خاموش میباشد. اما peripheral ها روشن هستند. برای هنگامی که کامپیوتر در حالت sleep میباشد، پردازنده خاموش میباشد و با فشردن کلید یک interrupt به سیستم داده میشود و پردازنده روشن میشود.

3- Wait Mode: در این حالت کلاک پردازنده متوقف میباشد و مانند این میباشد که سیستم فریز شده است. مثلا هنگامی که منتظر دریافت ورودی از کاربر میباشیم و پردازنده منتظر میماند تا کاربر ورودی را وارد نماید و سپس کلاک پردازنده روشن شده و pc[4] شروع به افزایش میکند.

4- Sleep Mode: در این حالت تنها کلاک پردازنده خاموش میباشد و کلاک موری و peripheralها روشن میباشد. هنگامی که میخواهیم حجم انبوهی از داده انتقال داده شود، از DMA استفاده میکنیم. در این حالت CPU فرماندهی گذرگاه را به DMA میسپارد و خودش کاری را انجام نمیدهد. بدون استفاده از DMA باید هر سری عملیات Instruction fetch انجام شود که این کار پردازنده را مشغول و سرعت انتقال را کاهش میدهد.

[4] Program counter

3) به سوالات زیر در مورد اجزای ریزپردازنده (SAM3X8E) پاسخ دهید:

الف) سه مدل مختلف تایمر در این ریزپردازنده را نام ببرید و موارد استفاده از هر کدام را شرح دهید.

Real time timer: این تایمر یک شمارنده ی 32 بیتی میباشد و ثانیه های سپری شده را میشمارد و هر ثانیه یکی به مقدار آن اضافه میشود. از این تایمر میشود برای اندازه گیری زمان و interrupt دادن و آلارم دادن استفاده نمود.

Real time clock: این تایمر برای استفاده به عنوان ساعت مناسب میباشد و به طوری است که مثلا هر ثانیه یکی به رجیستر ثانیه اضافه و هر 60 ثانیه یکی به رجیستر دقیقه اضافه میکند و ...
تایم این ساعت قابل ست کردن میباشد و هر بار با خاموش شدن میکرو تایم ریست شده و به مقدار دیفالت خود در می آید.
همچنین برای وقفه های دوره ای[5] مناسب میباشد. مثلا میخواهیم که هر روز ساعت 7صبح وقفه بدهد.

Watchdog timer یا تایمر مراقب: سیستم را از حالت قفل شدگی نجات میدهد بدین صورت که از قبل یک مقداری را ست میکنیم مثلا 10 ثانیه. و سپس پردازنده شروع به انجام آن عملیات میکند. اگر که پس از تایم مشخص شده یعنی آن 10(حداکثر آن 16 ثانیه میباشد در این میکرو خاص) ثانیه آن عملیات انجام نشد، این تایمر یک وقفه میدهد و موجب ریست سیستم میشود.
باید توجه کنیم که در صورت ست کردن تایمر و پس از انجام عملیات، تایمر را غیرفعال کنیم چرا که در غیر اینصورت سیستم را ریست میکند

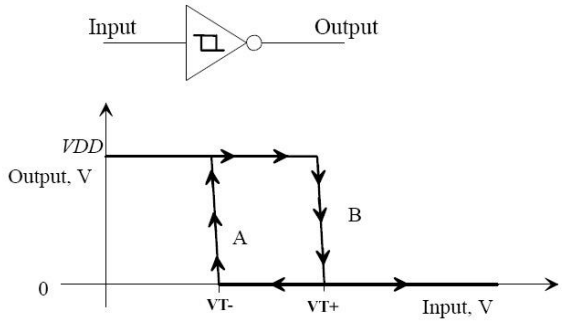
ب) شکل زیر نشان دهنده کدام GPIO میکرو ماست و نمودار آن را توضیح دهید.
مدار Schmitt trigger میباشد بدین صورت که یک threshold بالا و پایین در نظر گرفته میشود و یک ناحیه ای بین این دو threshold به وجود می آید که بسته به حالتی که در آن میباشد به 0 یا vdd تعبیر میشود.
مثلا threshold بالا و پایین را به ترتیب 2 و 3.5 ولت در نظر میگیریم و از 0 تا 2.5 به صفر تعبیر و از 2.5 تا 5 به یک تعبیر میشود. اما چون مدار Schmitt trigger میباشد یک مقدار threshold بالا و پایین وجود دارد که مزیت آن این است که اگر مثلا یک نویز ایجاد شود و امواج ولتاژ بین 2.5 نوسان کند، در خروجی صفر و یک بسیار زیادی مشاهده نشود.

[5] Periodic interrupt

بدین صورت که اگر ولتاژ به صفر تعبیر شود و در حال افزایش باشد، دقیقا هنگام رسیدن به 2.5 یک نمیشود و صبر میکند تا ولتاز به 3.5 برسد(طبق مثالی که زدم) و از آنطرف اگر مقدار ولتاژ به یک تعبیر شود و در حال کاهش باشد صبر میکند تا کمتر از 2 شود و آنگاه خروجی به صفر تعبیر میشود.

در شکل زیر نیز بدین گونه میباشد و مثلا هنگامی که ولتاژ به یک تعبیر میشود و درحال کاهش میباشد، همان لحظه به صفر تعبیر نمیشود و باید از مقدار threshold پایینتر بیاید.

همچنین در حالتی که به صفر تعبیر میشود نیز به هنگام افزایش، سریعا به یک تعبیر نمیشود و باید از مقدار threshold بالاتر برود.



4) به پرسش های زیر در مورد وقفه های تودرتو پاسخ دهید:

الف) NVIC چگونه وقفه‌های تودرتو را مدیریت می‌کند (از دیدگاه رجیسترهای NVIC شرح دهید)؟

جواب:

مکانیزم های مختلفی وجود دارند که برای هندل کردن وقفه ها در کنار هم کار میکنند. هنگامی که ISR یک وقفه در حال اجرا میباشـــد و یک وقفه ی جدید می آید، اســـتیت وقفه ی جدید برابر با PENDING قرار میگیرد و منتظر میماند تا وقفه مورد نظر تمام شـود. یکی از مکانیزم هایی که گفته شـــد tail-chaining میباشـــد؛ بدینگونه که پس از اجرای ISR مربوط به یک وقفه اگر وقفه ای در حالت PENDING بود و منتظر اجرا بود، استیت قبلی cpu از استک pop نمیشود و وقفه ی جدید اجرا میشود و پس از اتمام وقفه ها استیت قبلی cpu بازیابی میشود و از پشته pop میشوند.

یکی دیگر از این مکانیزم ها late arriving میباشـد که هنگامی که پس از آمدن وقفه، cpu در حال ذخیره کردن اسـتیت میباشـد، یک وقفه ی جدید با اولویت بالاتر از قبلی بیاید، وقفه ی جدید اجرا میشود.

ب) چهارتا از دسـتوراتی که تعداد کلاک بالایی برای اجرا نیاز دارند را نام ببرید و اگر درحین پردازش این دستورات وقفه ای رخ دهد چگونه با آنها برخورد خواهد شد؟

جواب:

دستور تقسیم / PUSH and POP / Store multiple(STM / Load multiple(LDM)

در صـــورتی که اینگونه دســـتور ها در حال اجرا باشـــد و وقفه ای رخ دهد، پردازنده اجرای اینگونه دسـتور[6] ها را رها میکند و به وقفه سـرویس میدهد. پس از اجرای وقفه دوباره دسـتوری را که رها کرده از اول اجرا میکند.

[6] Instruciton

5) به سوالات زیر در مورد NVIC پاسخ دهید:

الف) دلیل وجود دو حالت مختلف Active و A&P برای وقفه‌ها در NVIC را شرح دهید.

جواب:

حالت Active به این معنا میباشد که ISR وقفه ی مربوط به دیوایس در حال اجرا میباشد. اما A&P[7] به این معنی میباشد که علاوه بر در حال اجرا بودن ISR مربوط به دیوایس، آن دیوایس یک وقفه ی دیگر نیز داده است که در حالت Pending میباشد؛ چرا که در لحظه تنها یک وقفه را میتوان پردازش کرد.

ب) فرق بین دو ویژگی Tail-chaining و Late-arriving را توضیح دهید.

جواب:

هنگامی که وقفه میرسد و پردازنده میخواهد به وقفه سرویس بدهد، ابتدا 8تا رجیستر مربوط به برنامه در stack پوش میشود و ISR مربوط به وقفه اجرا میشود. حال پس از اجرا شدن ISR وقفه، استیت برنامه ی متوقف شده بازیابی میشود و آن 8 رجیستر از stack، pop میشوند.

حال هنگامی که یک ISR در حال اجرا باشد و همچنین یک وقفه ی دیگر در حالت pending داشته باشیم، پس از اجرای ISR اولی، رجیستر ها از پشته خارج میشوند. سپس پردازنده میخواهد که ISR دومی را اجرا کند و دوباره آن 8 رجیستر را وارد پشته میکند. و پس از اجرا آن 8 رجیستر را از پشته خارج میکند تا استیت برنامه ی متوقف شده بازیابی شود. در این حالت که چندین وقفه داریم و پس از اجرای یک وقفه، وقفه ی دیگر قرار است که اجرا شود، یک کار بیهوده و وقت گیر انجام شد و آن خارج و وارد کردن آن 8 رجیستر از پشته بود؛ در صورتی که نیازی به خارج کردن آن 8 رجیستر و وارد کردن مجدد آن نبود.

سیاست tail-chaining بدین صورت میباشد که اگر علاوه بر ISR فعلی که در حال اجرا میباشد، وقفه ی دیگری در حالت Pending داشته باشیم، آن 8 رجیستر را از پشته خارج نمیکند و وقفه ی دیگر را اجرا میکند و پس از تمام شدن اجرای ISR وقفه ها، آن رجیستر را از پشته خارج میکند.

اما سیاست late-arriving بدینگونه میباشد که اگر یک وقفه بیاید، پردازنده قبل از اجرای ISR مربوط به وقفه، استیت فعلی اش را در استک ذخیره میکند که در یک نوع میکرو کنترلر 16کلاک به طول می انجامد. حال در این هنگام که در حال ذخیره ی استیت میباشد، اگر یک وقفه ای با اولویت بیشتر از آن وقفه که قرار بوده اجرا شود، برسد، پس از ذخیره ی استیت، ISR آن وقفه ی جدید تر که دارای اولویت بیشتر میباشد را اجرا میکند. در واقع پردازنده بخاطر اجرا کردن ISR وقفه

[7] Active and pending

ی قدیمی استیتش را ذخیره کرده، اما به دلیل رسیدن یک وقفه ای با اولویت باشد، به آن وقفه ی جدیدتر سرویس میدهد.

ج) دلایل وجود قابلیت Masking را نام ببرید و انواع حالاتی که می‌توانیم با استفاده از رجیسترهای CPU جمعی از وقفه‌ها را باهم Mask کنیم را شرح دهید.

جواب:

میتوان به صورت نرم افزاری بعضی از وقفه ها یا همه ی آنها را mask کرد به این منظور که میتوان آنها را نادیده گرفت. هنگامی که ISR مربوط به یک وقفه ی مهم در حال اجرا میباشد و نمیخواهیم که هیچ وقفه ای بتواند این وقفه ی در حال اجرا را متوقف کند، قبل از اجرای وقفه ی موردنظر، به آن اولویت 0 را میدهیم و مقدار 0 دارای بیشترین اولویت میباشد؛ در نتیجه هیچ وقفه ای نمیتواند این وقفه ی در حال اجرا را متوقف کند.(به جز وقفه ها با اولویت منفی)

- برای این منظور یک رجیستر PRI_MASK وجود داره که در صورت اینکه مقدار آن یک شود اولویت برنامه در حال اجرا را برابر با صفر میکند.
- همچنین یک رجیستر FAULT_MASK داریم که در صورتی که یک شود، اولویت برنامه ی در حال اجرا منفی یک میشود و به جز وقفه های -2و -3، هیچ وقفه ی دیگری نمیتواند آنرا متوقف نماید.
- یک رجیستر دیگر به نام FAULT_MASK نیز وجود دارد که رجیستر هشت بیتی میباشد و مقدار آن از صفر تا ماکزیمم تعداد وقفه ها قابل تنظیم میباشد. بدین صورت که وقفه های با شماره ی اولویت پایینتر از مقدار این رجیستر میتوانند این وقفه را متوقف نمایند و وقفه های دارای مقدار اولویت بیشتر از مقدار این رجیستر نمیتوانند وقفه ی در حال اجرا را متوقف نمایند. در صورتی که مقدار رجیستر برابر با صفر قرار بگیرد، MASKکردن توسط این رجیستر غیر فعال میشود.

6) به پرسش‌های زیر در مورد NVIC Register پاسخ دهید:

الف) Vector table چیست و محتوی آن چگونه است؟

جواب:

جدولی اســـت که محتوی vector میباشـــد. واژه ی vector در embedded programming برای آدرس حافظه به کار برده میشـــود و در اینجا منظور جدولی اســـت که محتوای آدرس های حافظه میباشد. در واقع vector table فضای حافظه ای است که آدرس های مربوط به هر کدام از ISR[8] ها را نگه میدارد. و هنگامی که وقفه اتفاق میوفتد، NVIC که وظیفه ی کنترل وقفه ها را دارد، از vector table استفاده میکند و آدرس مربوط به ISR وقفه را پیدا میکند.

ب) چرا وقفه‌های NVIC در سری ریزپردازنده‌های ARM از شماره 1 شروع می‌شوند نه 0؟

جواب:

به دلیل اینکه ردیف شـماره ی صـفرم در vector table مربوط به آدرس stack pointer میباشـد و آدرس مربوط به هر کدام از ISR وقفه ها در ردیف یکم به بعد قرار دارند.

ج) با توجه به اینکه رجیســـتر NVIC-IPR اعداد Unsigned را در خود ذخیره می‌کند چگونه وقفه‌هایی با اولویت منفی داریم؟

جواب:

این نوع وقفه ها به صورت سخت افزاری پیاده سازی شده اند و اصلا رجیستری برا آنها وجود ندارد که بخواهیم اولویت آنها را تنظیم کنیم و دارای اولویت های بالاتری نسبت به تمام وقفه های دیگر میباشند.

[8] Interrupt service routine