دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

طراحی کامپیوتری سیستم‌های دیجیتال

پاییز 1401

دستیاران آموزشی: سارا رضائی منش، شایان شبیهی

مقدمه

در این تمرین همانند تمرین قبل از شما خواسته می‌شود که یک کنترلر و مسیر داده برای مدار خواسته شده، روی کاغذ طراحی کنید و سپس طراحی خود را با استفاده از زبان توصیف سخت‌افزاری وریلاگ، پیاده سازی نمایید. مهلت انجام این تمرین تا ۱۰ بهمن در نظر گرفته شده است.

دقت کنید که در هنگام تحویل، طراحی شما با کدتان تطبیق داده می‌شود. پس در هنگام موازی سازی چند عملیات، به تعداد کامپوننت هایی که برای اینکار احتیاج دارید توجه کرده و آن را در طراحی خود ذکر کنید.

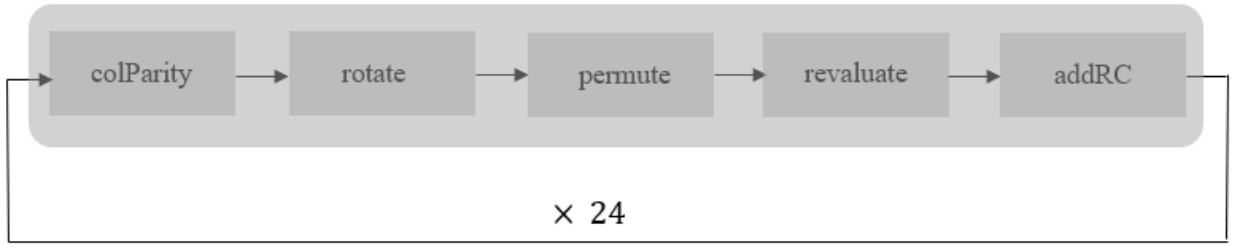
توجه: انجام این تمرین به صورت گروه های دو نفره خواهد بود.

توضیحات پروژه

هدف از این تمرین طراحی تابعی به نام encoder است. برای طراحی تابع encoder باید چند ماژول جدید به ماژول هایی که در پروژه اول و امتحان میانترم طراحی کردید، اضافه کنید. تابع encoder یک ورودی ۱۶۰۰ بیتی دریافت می کند و در نهایت یک خروجی رمزگذاری شده بازمی‌گرداند. عملیات encode در این تابع خود متشکل از چند تابع است به طوریکه خروجی هر تابع، ورودی تابع بعدی خواهد بود. جزئیات این توابع در ادامه توضیح داده شده است.

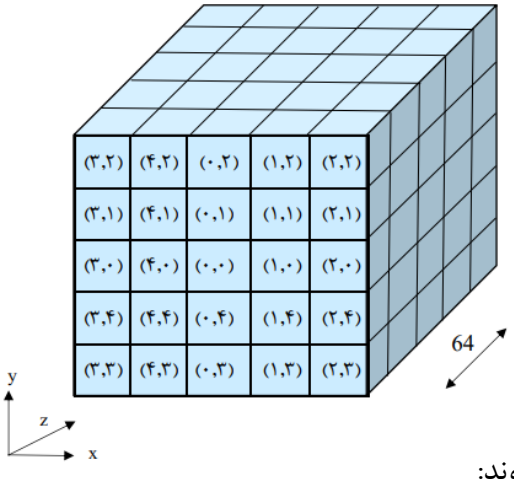
تابع encoder

تابع encoder عملیات رمزگذاری را با استفاده از پنج تابع انجام می‌دهد که تابع اول و سوم را در پروژه قبل و امتحان میانترم پیاده سازی کرده‌اید. این تابع یک ورودی ۱۶۰۰ بیتی دریافت کرده و ۵ تابع نمایش داده شده در شکل زیر را به ورودی اعمال می‌کند این عملیات ۲۴ بار تکرار خواهد شد و در نهایت خروجی تکرار ۲۴ام به عنوان متن رمز نگاری شده در نظر گرفته می‌شود. نمای کلی توابع به صورت زیر است:

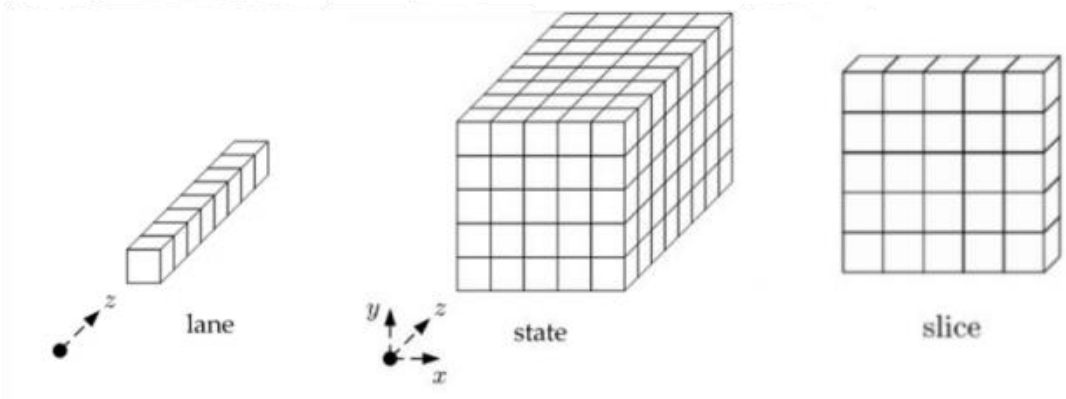


ماتریس ورودی

تمام توابع توضیح داده شده در ادامه، مانند پروژه قبل بر روی یک ماتریس ۵ در ۵ در ۶۴ به صورت زیر اعمال می‌شوند:



در این ماتریس مفاهیم lane و slice به صورت زیر تعریف می‌شوند:



توضیحات توابع

۱. تابع colParity

توضیحات این تابع مطابق توضیحات داده شده در امتحان میانترم می‌باشد.

۲. تابع rotate

این تابع برای هر بیت در ماتریس A به صورت زیر تعریف می‌شود:

$$A[x,y,z] = A[x,y,z] \ if \ x = y = 0$$

$$A[x,y,z] = A[x,y,(z-(t+1)(t+2)/2) \ mod \ 64] \ otherwise$$

$$0 < t < 24 \text{ and } \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix}^t \begin{pmatrix} 1 \\ 0 \end{pmatrix} mod \ 5 = \begin{pmatrix} x \\ y \end{pmatrix}$$

عملکرد تابع به شرح زیر است:

۱. Lane ای که در موقعیت (x, y) = 0 قرار دارد بی تغییر باقی می‌ماند و در lane های دیگر بیت ها با اعمال فرمول داده شده به موقعیت های دیگر در راستای محور z منتقل می‌شوند. مشخص است که بیت های متعلق به یک lane به یک میزان جا به جا می‌شوند.

۲. متغیر t برای تعیین میزان شیفت هر بیت و مقدار شیفتی که به هر lane اختصاص داده می‌شود مورد استفاده قرار می‌گیرد.

۳. ۲۴ شیفت انجام شده در این تابع توسط فرمول زیر محاسبه می‌شوند:

$$\frac{(t+1)(t+2)}{2} mod \ 64$$

۴. مقدار t بین ۰ تا ۲۴ قرار دارد و برای هر مقدار t، موقعیت متناظر آن در ماتریس به صورت زیر تعریف می‌شود:

$$\begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix}^t \begin{pmatrix} 1 \\ 0 \end{pmatrix} mod \ 5 = \begin{pmatrix} x \\ y \end{pmatrix}$$

به عنوان مثال برای t = 3 داریم:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix}^3 \begin{pmatrix} 1 \\ 0 \end{pmatrix} mod\ 5$$

$$= \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} mod\ 5$$

$$= \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} 0 \\ 2 \end{pmatrix} mod\ 5$$

$$= \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} 2 \\ 6 \end{pmatrix} mod\ 5 = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} mod\ 5$$

$$= \begin{pmatrix} 1 \\ 7 \end{pmatrix} mod\ 5 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

مقادیر جابه‌جایی بیت ها(مقداری که از z کم می‌شود) برای هر lane در جدول زیر آمده است و نیازی به محاسبه مجدد آن نیست. می‌توانید از مقادیر زیر به صورت مستقیم در طراحی خود استفاده کنید:

y/x	x = 0	x = 1	x = 2	x = 3	x = 4
y = 4	18	2	61	56	14
y = 3	41	45	15	21	8
y = 2	3	10	43	25	39
y = 1	36	44	6	55	20
y = 0	0	1	62	28	27

۳. تابع permute

توضیحات این تابع مطابق توضیحات داده شده در پروژه اول می‌باشد.

۴. تابع revaluate

این تابع تنها تابع غیر خطی encoder است. تابع revaluate برای هر بیت در ماتریس A به صورت زیر تعریف می‌شود:

$$A[x,y,z] = A[x,y,z] \oplus (NOT(A[x+1,y,z])\ AND\ A[x+2,y,z])$$

توجه: کد datapath و controller مربوط به این تابع داخل صفحه درس آپلود شده است و نیازی به پیاده سازی آن نیست.

۵. تابع addRc

این تابع بر روی ماتریس A به صورت زیر تعریف می‌شود:

$$A[0,0] = A[0,0] \oplus RC[i_r]$$

مقدار $RC[i_r]$ به ازای هر بار اجرای پنج مرحله بالا متفاوت و در هر مرحله ثابت است. وظیفه تابع addRC این است که هر lane در موقعیت [0, 0] را با یک ثابت ترکیب کند.(bitwise xor)

همانطور که گفته شد، در این تمرین کامپیوتری، توابع ذکر شده ۲۴ بار روی ماتریس وروی اعمال می‌شوند پس ۲۴ مقدار متفاوت RC خواهیم داشت. این مقادیر به صورت hexadecimal در جدول زیر آورده شده اند:

RC[0] = 0x0000000000000001	RC[12] = 0x000000008000808B
RC[1] = 0x0000000000008082	RC[13] = 0x800000000000008B
RC[2] = 0x800000000000808A	RC[14] = 0x8000000000008089
RC[3] = 0x8000000080008000	RC[15] = 0x8000000000008003
RC[4] = 0x000000000000808B	RC[16] = 0x8000000000008002
RC[5] = 0x0000000080000001	RC[17] = 0x8000000000000080
RC[6] = 0x8000000080008081	RC[18] = 0x000000000000800A
RC[7] = 0x8000000000008009	RC[19] = 0x800000008000000A
RC[8] = 0x000000000000008A	RC[20] = 0x8000000080008081
RC[9] = 0x0000000000000088	RC[21] = 0x8000000000008080
RC[10] = 0x0000000080008009	RC[22] = 0x0000000080000001
RC[11] = 0x000000008000000A	RC[23] = 0x8000000080008008

همانطور که در بالا مشخص است، هر ثابت ۶۴ بیت دارد و هر lane هم شامل ۶۴ بیت است.

نکات مهم:

- همانند پروژه های قبل، داده‌ها به صورت اسلایس خوانده شده و پردازش می‌شوند.
- در این ماتریس ایندکس ها در جهت سه محور به صورت **چرخشی** هستند.
- حین اجرای هر تابع باید برای آپدیت کردن مقدار یک خانه، از **مقادیر اولیه ای** که در ماتریس ورودی هستند استفاده کنید و نه مقادیر آپدیت شده.
- برای خواندن ورودی تنها مجاز به استفاده از یک **رجیستر ۲۵ بیتی** هستید.

فایل های ورودی و خروجی

نکات مربوط به فایل های ورودی و خروجی همانند پروژه قبل و امتحان میانترم هستند.

مواردی که باید در حین پیاده سازی در نظر بگیرید

- بخشی زیادی از نمره نهایی شما، مربوط به اجرای درست برنامه می‌شود. بدین منظور با بررسی تست کیس ها مختلف و متنوع از اجرای درست برنامه مطمئن شوید.
- این پروژه تحویل حضوری دارد و برنامه شما با تست کیس های جدید بررسی خواهد شد.

مواردی که باید تحویل دهید

- گزارش شامل طراحی کنترلر (FSM) و مسیر داده بر روی کاغذ
- خروجی های تست کیس ها مطابق روشی که در پروژه اول ذکر شد.
- تمامی فایل های لازم برای اجرای پروژه (فایل های hdl، تست بنچ و... به فرمت trunk).