بسمه تعالی

درس سیستم‌های نهفته مبتنی بر هسته

تکلیف کامپیوتری ۳: یادگیری استفاده از ابزار TCE برای پیاده‌سازی روی پردازنده TTA

پردیس دانشکده‌های فنی دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر
دکتر احمد شعبانی
نیم‌سال دوم سال تحصیلی 1401-1400
نگارش: فرید حسینی (f.hosseiny96@gmail.com)

مقدمه: هدف از این تمرین کامپیوتری، پیاده‌سازی یک نمونه پردازنده ASIP مبتنی بر ساختار TTA برای یک کاربرد خاص است. کاربرد خاص مورد نظر در این تمرین، پیاده‌سازی الگوریتم تبدیل گسسته کسینوسی DCT 8×8 است. تبدیل کسینوسی گسسته دنباله‌ای از اعداد را به صورت مجموع توابع کسینوسی با فرکانس‌های متفاوت نمایش می‌دهد. این تبدیل در پردازش تصویر و ویدئو و فشرده‌سازی داده کاربردهای فراوانی دارد. از آنجا که پیاده‌سازی سخت افزاری مستقیم این تبدیل به دلیل بالا بودن تعداد ضرب کننده‌های ممیز شناور، پیچیده و مستلزم زمان اجرای زیاد است، بررسی روشهای بهینه‌سازی و کاهش زمان اجرای این تبدیل برای یک کاربرد خاص منظوره مورد توجه است. در حالت کلی، میتوان تبدیل گسسته کسینوسی تک بعدی را برای یک بلوک با ابعاد N*N به شکل زیر بیان نمود که در این رابطه x_n نمونه‌های داده ورودی و X_K خروجی ضرایب تبدیل گسسته کسینوسی می‌باشد.

$$X_k = \sum_{n=0}^{N-1} x_n cos\left[\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right] \qquad for \;\; k = 0, 1, 2, \dots, N-1$$

ابزار TCE که در این تمرین با آن آشنا می‌شویم به ما کمک می‌کند که الگوریتم موردنظر را روی پردازنده‌ی همه منظوره TTA پیاده‌سازی کنیم و در جهت بهبود پارامترهای طراحی و شخصی‌سازی پردازنده، تغییراتی را برای یک کاربرد خاص در این پردازنده انجام دهیم و نتایج مربوطه را استخراج نماییم. این ابزار علاوه بر فراهم ساختن امکاناتی برای مشاهده نحوه اجراء بر روی پردازنده مذکور، امکان اعمال تغییرات در سطح معماری را فراهم می‌سازد. ضمن اینکه، امکان پروفایل‌گیری بمنظور مشخص کردن نقاط هات اسپات در الگوریتم و اضافه کردن یکسری شتاب‌دهنده سخت افزاری وجود دارد.

گام اول:

۱-مطابق دستورات فایل install_tutorial نرم‌افزار را نصب کنید و خروجی نهایی که نشان‌دهنده‌ی نصب کامل نرم‌افزار است را در گزارش خود بیاورید. (۵)

گام دوم: حالا از پوشه‌ی tce وارد پوشه‌ی data و سپس mach شوید و فایل minimal.adf را کپی کنید. آن را در پوشه‌ی DCT که از پیوست ذخیره کرده‌اید پیست کنید و در همان پوشه ترمینال جدیدی باز کنید. با اجرای خط کد زیر می‌توانید سیستم اولیه‌ی مینیمالی که در اختیار داریم را مشاهده کنید.

```
prode minimal.adf
```

حال می‌خواهیم الگوریتم موجود در کدهایی که به صورت آماده در فایل dct_8x8_16_bit.c در اختیار شما قرار داده شده‌است را بر روی این سیستم مینیمال پیاده کنیم.

```
tcecc -O2 -a minimal.adf -o dct.tpef dct_8x8_16_bit.c
proxim minimal.adf dct.tpef
```

۲- دو خط بالا را اجرا کرده و در پنجره‌ی باز شده run را بزنید و در پایین صفحه دستور info proc cycles و سپس info proc stats را وارد کنید و تعداد سیکلی که طول می‌کشد تا کد اجرا شود و خلاصه‌ای از آمار داده‌شده را در گزارش خود بیاورید. از منوی source می‌توانید profile data و سپس highlight top execution counts را انتخاب کنید تا بیشترین تعداد انجام شدن دستورات را مشاهده کنید. در تمام قسمت‌ها نتایج را در گزارش آورده و فایل‌ها را با نام مناسب ضمیمه کنید. (۸)

حال دوباره دستور prode minimal.adf را اجرا کنید و روی واحد RF:RF دبل کلیک کرده و مقدار size را از ۵ به ۱۰ تغییر دهید و در قسمت ports گزینه‌ی add را انتخاب کرده و دو پورت جدید اضافه کنید و save as را انتخاب کرده و با نام added_registers.adf ذخیره کنید.

۳- کدهای زیر را در ترمینال وارد کرده و مراحل سوال ۲ را برای سیستم جدید پیاده کرده و نتایج را مقایسه کنید. (۸)

```
tcecc -O2 -a added_registers.adf -o dct.tpef dct_8x8_16_bit.c
proxim added_registers.adf dct.tpef
```

سیستم مینیمالی که در اختیار داریم فقط یک باس برای اطلاعات دارد. prode added_registers.adf را اجرا کرده و از منوی edit گزینه‌ی add و سپس transport bus انتخاب کنید. این کار را تکرار کرده تا زمانی که ۸ باس داشته باشیم. می‌توانید هر سوکت را به هر باس متصل یا جدا کنید اما در این تمرین کافی است از منوی tools گزینه‌ی fully connect IC را انتخاب کنید.

۴- سیستم جدید را با اسم multiple_bus.adf ذخیره کرده و مراحل قبل را تکرار و نتایج را مقایسه کنید. (۸)

۵- با توجه به نتایج و این که از هر باس چند درصد مواقع استفاده شده بگویید آیا به ۸ باس احتیاج داشتیم یا با تعداد باس کمتر نیز به نتایج نسبتا مشابه می‌رسیدیم؟ با ذکر دلیل توضیح دهید چند باس برای بهبود کار ما کافی بود؟ تعداد باس را به مقدار پیشنهادی تغییر دهید و تعداد سیکل را مقایسه کنید. (۱۰)

۶- برای این که مقایسه کنیم کدام دستورات پرتکرارترین هستند ابتدا دستوری که در ترمینال وارد می‌کنیم را به شکل زیر تغییر می‌دهیم و سپس highlight top execution counts را انتخاب می‌کنیم. (۸)

```
tcecc -O2 -a multiple_bus.adf -o dct.tpef dct_8x8_16_bit.c --disable-inlining
proxim multiple_bus.adf dct.tpef
```

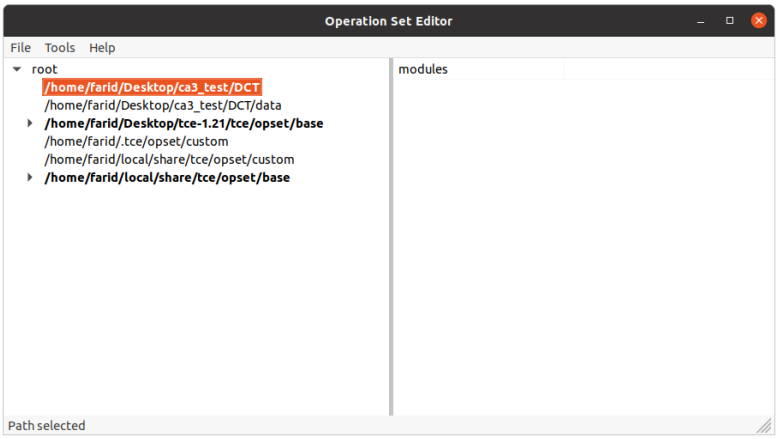
گام سوم: حال می‌خواهیم واحد محاسباتی جدیدی معرفی کنیم که به صورت سخت افزار اختصاصی بخش‌های پرتکرار الگوریتم را انجام دهد. در کد dct داده شده سه تابع مشخص شده‌اند که ضرب و جمع و کسینوس را محاسبه می‌کنند.

جمع و ضرب به شکلی پیاده شده‌اند که با وجود این که نوع داده‌ها بدون علامت است محاسبات علامت‌دار باشند. عملیات ضرب، خروجی را ۱۵ بار به راست شیفت می‌دهد که دلیل آن مرتبط با این است که هرجا عملیات ضرب داریم یکی از اپرندهای آن کسینوس است، بنابراین؛ می‌توان به جای کسینوس 2^{15} برابر آن را محاسبه کرد و بعد از ضرب تقسیم را انجام داد. بدین صورت مقادیر کسینوس به اندازه کافی بزرگ می‌شوند تا بتوانیم از نوع داده‌ی بدون علامت استفاده کنیم.

با توجه به رابطه‌ی dct می‌توان مشاهده کرد که به تمام مقادیر کسینوس نیازی نداریم و فقط ضرایب فرد $\frac{\pi}{16}$ مورد نیاز هستند. پس کد را می‌توان به گونه‌ای نوشت که خروجی کسینوس برای این مقادیر را به صورت بدون علامت بدهد.

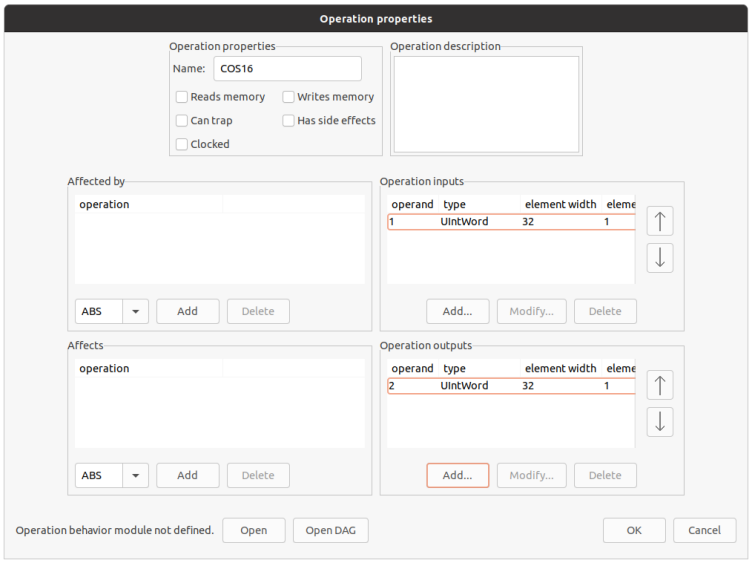
۷- با مطالعه‌ی کد dct_8x8_16_bit.c و سه تابع معرفی شده در انتهای آن توضیح دهید این سه تابع به چه شکل کار می‌کنند. (نیازی به جزئیات کامل نیست، کافی است توضیح دهیدعملکرد کلی این سه تابع با توجه به ورودی‌های مختلفی که می‌گیرند به چه صورت است) (۱۵)

برای اضافه کردن توابع مورد نظر در ترمینال osed را وارد کنید. در صفحه‌ی باز شده root را باز کرده و مانند شکل زیر اولین خط آدرس محلی است که در آن قرار دارید. روی آن کلیک راست کنید و یک ماژول جدید به نام custom درست کنید.



شکل ۱ انتخاب مسیر مناسب در osed

سپس روی نام ماژول کلیک کرده و add operation را انتخاب کنید. نام عملگر را COS16 بگذارید و مانند شکل زیر یک ورودی و یک خروجی از نوع Uint word و با اندازه ۳۲ بیت برای آن اضافه کنید.



شکل ۲ اضافه کردن تابع جدید در osed

دو عملگر دیگر با نام‌های MUL_16_FIX و ADD_16_FIX اضافه کنید که هر کدام دو ورودی و یک خروجی مانند قبل داشته باشند. روی یکی از این عملگرها کلیک راست کرده و modify behavior را انتخاب کنید و محتویات فایل custom.c را در فایل custom.cc که باز شده کپی کنید.

۸- روی ماژول custom کلیک راست کرده و build را بزنید و سپس روی هریک از عملگرها کلیک راست کنید و simulate بزنید و درستی آن‌ها را بررسی کنید. (برای این کار باید توجه کنید هر عملگر چه ورودی‌هایی می‌تواند داشته باشد و خروجی آن به چه صورت تعبیر می‌شود.) (۱۳)

بعد از اطمینان از درستی عملگرها osed را ببندید و با دستور prode آخرین سیستمی که ساختید را باز کنید. از منوی edit گزینه‌ی add و سپس function unit را انتخاب کنید. نام آن را CUSTOM گذاشته و در قسمت سمت راست (پورت‌ها) گزینه‌ی add را انتخاب کنید و گزینه‌ی trigger را انتخاب کنید. دو پورت دیگر نیز اضافه کنید بدون این که تریگر آن‌ها را فعال کنید.

در قسمت operations گزینه‌ی add from Opset را انتخاب کنید و از لیست ADD_16_FIX را با latency 4 انتخاب کنید. فرض می‌کنیم پس از ۴ سیکل خروجی جمع‌کننده آماده می‌شود که در واقعیت احتمالا مقدار کمتری خواهد بود. همین کار را برای MUL_16_FIX و COS16 نیز تکرار می‌کنیم. کسینوس تنها یک ورودی دارد بنابراین از دو پورت استفاده می‌کند. هنگام اضافه کردن این عملگر اپرند ۲ را به پورت ۳ وصل می‌کنیم تا همه‌ی خروجی‌ها به پورت یکسان وصل باشند. سیستم جدید را با نام custom.adf ذخیره کنید.

حال سراغ کد dct_8x8_16_bit.c می‌رویم. باید توابع قبلی با توابع ساخته‌شده جایگزین شوند. در این مرحله به شکل زیر کدها را جایگزین کنید. در حالت اولیه داریم:

```
a = cos16(b)
a = fix_add_16(b,c)
a = fix_mul_16(b,c)
```

در حالت جدید این توابع به شکل زیر در می‌آیند. توجه کنید خروجی‌ها آخر می‌آیند.

```
_TCE_COS16(b,a)
_TCE_ADD_16_FIX(b,c,a)
_TCE_MUL_16_FIX(b,c,a)
```

با توجه به این که نوع آرگومان گرفتن توابع متفاوت است و خروجی در آرگومان‌ها می‌آید باید به دقت کد را اصلاح کنید و در صورت نیاز متغیرهای میانی جدید تعریف کنید.

۹- پس از ساختن واحد محاسباتی جدید و اضافه کردن آن کد را اصلاح کنید. یک بار فقط cos16 و بار دیگر فقط ضرب‌کننده و یک بار فقط جمع‌کننده را جایگزین کنید. یک بار نیز هرسه را با شتاب‌دهنده سخت‌افزاری ساخته‌شده جایگزین کنید. برای هر حالت مراحل قبل را تکرار کنید و نتایج را مقایسه کنید و در گزارش بیاورید. کدام تابع تاثیر بیشتری در تعداد سیکل داشت؟ (۲۵)

تاریخ تحویل: ساعت 23:59 روز 30/2/1401

موفق و سلامت باشید