



ارائه شده توسط:

سایت ترجمه فا

مرجع جدیدترین مقالات ترجمه شده

از نشریات معتبر

الگوریتم اندازه بندی بافر برای شبکه های روی تراشه با استفاده از TDMA و

کنترل جریان انتها به انتها بر اساس اعتبار

چکیده

هنگام طراحی یک سیستم بر روی تراشه (SOC) با استفاده از یک شبکه بر روی تراشه (NOC)، سطح سیلیکون و مصرف توان، دو عنصر کلیدی برای بهینه سازی می باشند. بخش غالب سطح NOC و مصرف توان ناشی از بافرها در واسطه های شبکه (NIS) می باشد که برای جداسازی محاسبات از ارتباطات مورد نیاز هستند. چنین جداسازی، مانع از به تاخیر انداختن بلوک های IP با توجه به اتصال های ارتباطی می شود. اندازه این بافرها به خصوص در سیستم های زمان واقعی مهم است، همانطور که آنها باید به اندازه کافی برای به دست آوردن عملکرد قابل پیش بینی بزرگ باشند. برای اطمینان از اینکه بافرها سرریز نمی کنند، کنترل جریان انتها به انتها مورد نیاز است. یک شکل از کنترل جریان انتها به انتهای مورد استفاده در NOCها، کنترل جریان مبتنی بر اعتبار است. این شکل، الزامات اضافی را برای اندازه بافر قرار می دهد، چرا که تاخیر در کنترل جریان باید در نظر گرفته شود. در این کار، ما یک الگوریتم برای پیدا کردن حداقل اندازه بافر جداسازی را برای NOC با استفاده از TDMA و کنترل جریان انتها به انتها مبتنی بر اعتبار، تحت محدودیت های عملکرد در برنامه های کاربردی در حال اجرا در SOC ارائه می دهیم. آزمایشات ما نشان می دهد که زمانی که با روش های مدرن اندازه بندی بافر مقایسه می شود، روش ما به کاهش 84٪ از کل سطح بافر NOC منجر می شود. علاوه بر این، روش ما دارای پیچیدگی زمان اجرای کم، تولید نتایج در اندازه دقیقه برای آزمایش های ما، فعال نمودن چرخه های طراحی سریع برای طراحی های SOC بزرگ است. در نهایت، روش ما می تواند برای موارد استفاده متعدد در حال اجرا در همان SOC در نظر گرفته شود.

دسته بندی ها و توصیفات موضوع: B.4.3 ورودی / خروجی و ارتباطات داده ها: ارتباطات داخلی

شرایط عمومی: الگوریتم ها، تأیید

واژه های کلیدی: سیستم بر روی تراشه، شبکه بر روی تراشه، سطح، بافرها

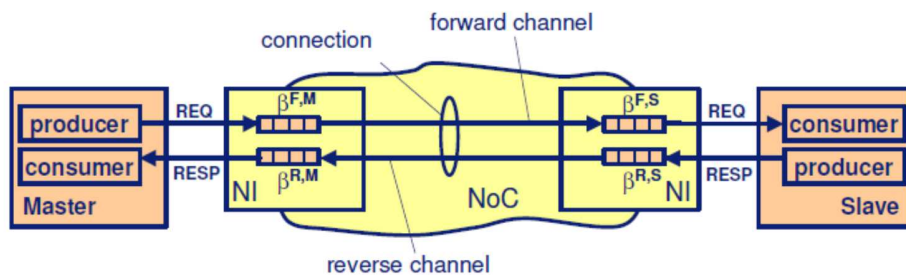
1 مقدمه

برای مقابله موثر با افزایش پیچیدگی طراحی SOCها، معماری محاسبات باید از معماری ارتباطات [16] جدا شود. با چنین جداسازی، معماری محاسبات و ارتباطات را می توان به طور مستقل طراحی نمود که در نتیجه تمامی مراحل طراحی را تسریع می کند و از این رو زمانی برای بازاریابی SOCها را نیز کاهش می دهد. NOCها می توانند چنین جداسازی را با بافرهای جداسازی بین بلوک های محاسباتی و بلوک های ارتباطی ارائه دهند و در نتیجه تفاوت ها بین سرعت عمل و تسریع هسته ها و NOC را مخفی نمایند. این کار به هسته اجازه می دهد تا تبادلات خود را بدون توجه به وجود یا تاثیر یک اتصال داخلی اجرا نماید، به عنوان مثال، در صورتی که NOC مشغول هسته دیگری باشد، آنها نمی خواهند در کار آنها تاخیر ایجاد شود.

روش ها برای یافتن حداقل اندازه بافر جداسازی NOC برای مجموعه ای از برنامه های کاربردی که بر روی SOC اجرا می شوند، به دو دلیل یک مسئله مهم است. نخست، بافر جداسازی مقدار قابل توجهی از سطح NOC و انرژی را می گیرد و در نتیجه یافتن حداقل نیازهای بافرینگ، کلیدی را برای دستیابی به یک پیاده سازی کارآمد NOC است. دوم، برای رفتار سیستم قابل پیش بینی، ما نیاز به محاسبه حداقل بافرینگ داریم که هنوز الزامات این کاربرد را برآورده سازد.

علاوه بر این، برخی از NOCها مکانیزم های کنترل جریان انتها به انتهای مبتنی بر اعتبار را برای ارائه عملکرد تضمین شده سیستم و حذف بن بست های وابسته به پیام در سیستم به کار می گیرند [1]. در این مورد، بافرینگ اضافی برای مخفی کردن زمان تاخیر انتها به انتها برای مکانیزم کنترل جریان و ارائه عملکرد عملیاتی کامل مورد نیاز است. اگر بافرها خیلی کوچک باشند، آنگاه توان عملیاتی و تاخیر تاثیرپذیر می شوند و هیچ تضمینی انتها به انتها را ن می توان ارائه داد.

در این مقاله ما به مشکل محاسبه حداقل اندازه بافرهای جداسازی NoC می پردازیم. ما یک روش طراحی نرم افزار خاص را برای تعیین حداقل اندازه های بافر برای اتصالات توان تضمین شده (GT) از معماری NoC Æthereal ارائه می دهیم [15]. ما رفتار ترافیک نرم افزار و رفتار شبکه را برای تعیین مرزهای دقیق آن بر روی اندازه بندی بافر مدلسازی می نماییم. در روش ما، ما همچنین الزامات بافرینگ را ناشی از استفاده از کنترل جریان انتها به انتها بر اساس اعتبار در نظر می گیریم. ما روش خود را برای چندین طرح SoC اعمال می نماییم که نشان می دهد که



شکل 1: بافرها برای یک اتصال

روش پیشنهادی منجر به کاهش زیادی در سطح کلی بافر NoC (84٪ به طور متوسط) و مصرف توان در مقایسه با روش تحلیلی می شود. روش ما دارای پیچیدگی زمان اجرا کم می شود و بنابراین به طرح SOC پیچیده قابل اجرا است. این روش را می توان برای طرح های با موارد استفاده متعدد، با در نظر گرفتن حداکثر اندازه بافر مورد نیاز بیش از همه مورد استفاده برای هر بافر استفاده شود. نهایتاً، این روش نیز در جریان طراحی کاملاً اتوماتیک یکپارچه می شود که چرخه های طراحی سریع روی یک طراحی SOC فعال می سازد. اگر چه روش الگوریتمی که برای معماری Æthereal ارائه شده است، می تواند برای هر NoC اعمال شود که برای آن رفتار هر دو هسته های IP و شبکه های دوره ای است، مانند [5] aSoc و [6] Nostrum.

به طور سنتی، روش های مبتنی بر شبیه سازی (یا ردیابی) مانند [12] برای محاسبه الزامات بافر در سیستم استفاده می شوند. در حالی که آنها مرز مطلوب را برای ردیابی معین ارائه می دهند، هیچ تضمینی وجود ندارد که اندازه های بافر نتیجه گرفته شده، آثار مختلف را برآورده خواهد ساخت. از این رو، آنها نمی توانند برای ساخت سیستم های

قابل پیش بینی استفاده شوند. روشهای تحلیل برای اندازه گیری بافرها بر اساس رفتار دوره ای محدود، شناخته شده است مانند آنهایی که در [2، 3] ارائه شده است. این روش ها معمولاً بیش از حد بدبینانه هستند و می توانند به بافرهای بزرگتر نسبت به بافرهای مورد نیاز برای طراحی منجر شوند. ما این کمیت را در بخش 5 تعیین می کنیم. رویکردهای تصادفی بر اساس تئوری صف بندی در [7] نشان داده شده اند. چنین مدل های تصادفی می توانند مشخصات ترافیک واقعی کاربرد را تخمین بزنند و از اینرو رفتار سیستم نمی تواند تضمین شود.

یک نظریه ریاضی کلی، برای مدل سازی رفتار شبکه به طور کلی [8]، ایجاد شده است. این نظریه، مرزهای محاسبات در تاخیر و ارتباط پشتیبان مربوط در شبکه را اجازه می دهد. پایه های رویکرد الگوریتمی پیشنهاد شده برای اندازه بندی بافر، بر اساس مدل های حساب دیفرانسیل و انتگرال شبکه هستند.

نمودارهای جریان داده ها همزمان (SDF) برای مدلسازی پردازش سیگنال و برنامه های کاربردی چند رسانه ای توسط چند محقق ارائه شده است [9]. استفاده از مدل های SDF برای به حداقل رساندن الزامات بافرینگ پردازنده ها در [10] ارائه شده است. استفاده از SDFها برای مدلسازی NOCها در [11] ارائه شده است. با این حال مدل های SDF، یک تولید و مصرف داده یکنواخت را برای محاسبه الزامات بافرینگ فرض می کند. در NOCهایی که که ارائه دهنده تضمین توان عملیاتی هستند، شیارهای TDMA اختصاص داده شده به یک جریان ترافیک نباید به طور یکسان در طول زمان گسترش یابند. بنابراین، مدل های SDF نمی تواند شبکه را در جزئیاتی که در اینجا نشان داده شده است مدلسازی نمایند و از این رو نتایج به دست آمده کمتر مطلوب هستند.

2. Ethereal NOC

معماری Ethereal NOC از اندیشه ارتباط برای ارائه جریان های ارتباطی بین IP و هسته ها استفاده می کند. [15]. چنین ارتباطاتی به منظور تخصیص منابع مانند شیارهای TDMA و بافرها برای رفتار زمان واقعی نیاز می شوند. یک ارتباط شامل دو کانال می شود، یک کانال ارسال و یک کانال معکوس (شکل 1 را ببینید). در کانال مستقیم، درخواست ها ممکن است از یک پایه به پیرو فرستاده شوند و در کانال معکوس، یک پاسخ می تواند

فرستاده شود (در صورت یک معامله خواندن به عنوان مثال). در نتیجه در کانال مستقیم، پایه تولیدکننده و پیرو، یک مصرف کننده است؛ در کانال معکوس، این نقش ها عوض می شود.

هر اتصال دارای چهار بافر در واسطه های شبکه (NIها) برای اتصال هسته های IP به شبکه است: $\beta^{F,M}$ و $\beta^{F,S}$ ، که نشان دهنده بافرها در پایه و NIهای پیرو برای کانال مستقیم است و $\beta^{R,S}$ ، $\beta^{R,M}$ برای کانال معکوس. بافرها در NI باید تفاوت ها در سرعت عامل و اندازه های پشت سر هم بین هسته های IP و NOC را جبران کنند. دلیل اینکه هر اتصال دارای صف خود است اینست که اگر ارتباطات یک صف واحد را به اشتراک بگذارند، وابستگی بین این ارتباط بوجود می آیند. اگر در یکی از اتصال ها، داده ها مصرف نشوند، اتصالات دیگر را مسدود می کند که به نوبه خود الزامات زمانی برآورده نمی شود و یا حتی به بن بست منجر می شود [1].

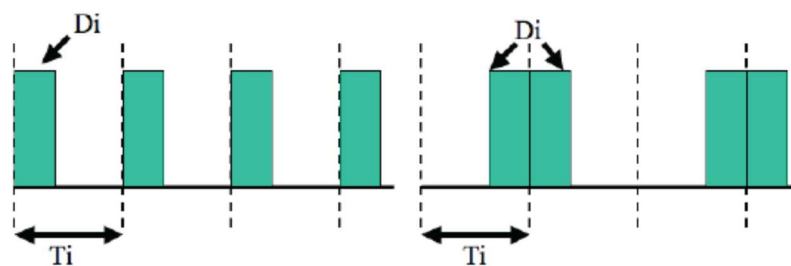
NOC توان عملیاتی فراهم می کند و زمان تاخیر های با استفاده از [14] TDMA تضمین می شود. این کار با استفاده از جداول حافظه در NIها پیاده سازی می شود که در آن هر حافظه نشان دهنده مقدار مساوی از زمان است. سپس هر اتصال، تعدادی از حافظه ها را برای مطابقت با پهنای باند آن و زمان تاخیر مورد نیاز اختصاص می دهد [4]. در هر حافظه، تعداد ثابتی از کلمات می تواند به شبکه فرستاده شود. کلمه اول همیشه یک هدر بسته است، مگر اینکه حافظه قبلی ارسال شده توسط اتصال همان اشغال شده باشد.

هنگامی که داده ها، NI را ترک می کنند، NOC، یک مسیر عاری از مباحثه را به هدف [14] NI تضمین می کند. این مورد توسط رزرو حافظه های زمانی همتراز شده برای هر لینک مسیریاب به دست می آید. محاسبه حافظه ها و مسیرهای عاری از بحث مربوطه از طریق شبکه مسیریاب در حال حاضر در زمان طراحی انجام می شود [4]. داشتن یک راه بدون مشاجره منجر به حداقل بافرها در مسیریاب ها می شود، چرا که هیچ بسته ای همیشه نباید منتظر بماند. هر چند لازم است که داده ها از مسیریاب ها توسط NIها پذیرفته شوند، در غیر این صورت، اگر مصرف کننده کند باشد است و یا هرگز پاسخ ندهد، بافرهای NI پر می شوند و در نهایت به شبکه ریخته می شوند. این کار مسیریابی عاری از بحث را می شکد و تضمین می شود.

به منظور جلوگیری از این مورد، Aethereal NoC کنترل جریان انتها به انتها را با استفاده از یک مکانیزم مبتنی بر اعتبار به کارگیری می کند [17]. شمارنده های محلی در NI، مقدار فضا را در بافرهای NI های از راه دور برای هر اتصال پیگیری می نماید. هر گاه یک کلمه NI را ترک می کند، شمارنده یک واحد کم می کند. اگر شمارنده به صفر برسد، NI مجاز به ارسال داده ها به شبکه نمی باشد. هر گاه یک کلمه در NI مصرف کننده مصرف شود، اعتبار تولید می شود و بر روی شبکه ارسال می شود وقتی که یک حافظه زمانی برای اتصال در دسترس است.

از آنجا که اعتبارها بلافاصله نمی رسند، یک تولید کننده بلافاصله نمی داند که چه زمان داده های آن مصرف می شوند. برای جلوگیری از به تاخیر انداختن تولیدکننده اگر اعتبار NI تمام شود، ما باید برای کنترل جریان انتها به انتها را در محاسبه بافر خود نیز به حساب آوریم. این روی محاسبات بافرینگ کانال مستقیم و معکوس تاثیر می گذارد. در کانال مستقیم، اعتبارات کنترل جریان برای کانال های معکوس فرستاده می شوند. در کانال معکوس، اعتبارات کنترل جریان برای کانال مستقیم فرستاده می شوند.

اعتبار کنترل جریان در هدر بسته اطلاعاتی [14] ارسال می شود. در نتیجه، پهنای باند موجود مستقل از پهنای باند اعتبار است. در عرض یک اتصال، کانال مستقیم و معکوس نیز می تواند به طور مستقل از یکدیگر محاسبه شوند. در نهایت، هر اتصال دارای بافر خود در NIS است؛ اتصالات دیگر نمی توانند دخالت کند؛ و ما می توانیم به هر اتصال به صورت جدا با توجه به اندازه بافر در NI نگاه کنیم. نتیجه این، استقلال در ترکیبات است؛



شکل 2: دوره ای (چپ) و غیردوره ای (راست)

اتصالات می توانند بدون تاثیر بر دیگری حذف و افزوده شوند و باعث تایید آسانتر می شوند و تضمین های زمان واقعی می تواند به آسانی حفظ شود. این کار الگوریتم های اضافه نمودن و ساده تر مانند الگوریتم بافرینگ توصیف شده در این مقاله را میسر می سازد که می تواند الزامات بافرینگ را برای هر اتصال به صورت مجزا توصیف نماید. ما الگوریتم اندازه بندی بافر را برای کانال های مستقیم و معکوس توصیف می نماییم. قبل از این، ما باید رفتار برنامه را مشخص نماییم.

3. رفتار برنامه

به منظور محاسبه اندازه بافرهای جداسازی در NI، ما باید رفتار برنامه هسته های IP را مشخص نماییم. ما سه نوع الگوی تولید را برای هسته های IP در نظر می گیریم. اولاً، الگوی تولید دوره ای که در آن تولیدکننده، سلسله ای با اندازه ثابت در همان زمان در هر دوره تولید می شود. مثالی از الگوی دوره ای تولیدکننده با اندازه سلسله D_i و دوره T_i در سمت چپ در شکل 2 نشان داده شده است.

ثانیاً، الگوی تولید غیردوره ای که در آن تولیدکننده، سلسله ای با اندازه ثابت را تولید می کند، اما این سلسله ها می توانند در هر جایی در این دوره ظاهر شوند. چنین مدلی، مشخصه کاربردها با عدم قطعیت حدودی در زمانی است که در آن سلسله ها تولید می شوند. بیشتر الگوهای ترافیک از برنامه های پردازش ویدئو دوره ای و سلسله وار هستند [13] و می توانند توسط الگوهای تولید غیردوره ای یا دوره ای مدلسازی شوند. مثالی از الگوی تولید در سمت راست در شکل 2 نشان داده شده است.

نهایتاً، الگوی تولید چند دوره ای که در آن یک تولیدکننده، چندین سلسله را تولید می کند، با اندازه های سلسله متفاوت و زمان بین سلسله ها. به عنوان مثال، در سیستم های نمایش ویدئو که سلسله های داده ها برای هر خط اسکن افقی دارای اندازه سلسله ثابت هستند، با یک دوره خالی ثابت بین دو خط اسکن. هرچند، بعد از مجموعه کامل از اسکن های افقی، یک دوره خالی بزرگتر برای اسکن عمودی وجود دارد.

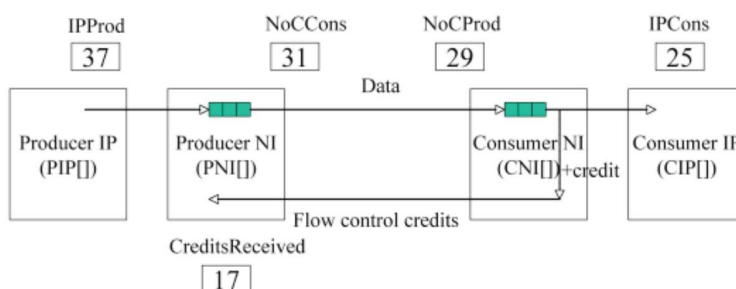
حتی اگر یک رفتار تولید کننده خاص با یکی از این سه نوع تولید متناسب نباشد، چندین روش را می توان استفاده کرد تا آن را به یکی از آنها تبدیل کند. مشخصات بدترین حالت می تواند برای محاسبه اندازه دوره و سلسله مورد استفاده قرار می گیرد که به عنوان مثال از تخمین های تحلیلی و یا از چند اجرای تجربی از برنامه به دست می آید.

4. محاسبه اندازه های بافر NI

به منظور محاسبه اندازه های بافر در NIها، ما می خواهیم حداکثر اختلاف بین تعداد کلمات تولید شده و تعداد کلمات مصرف شده را در هر نقطه ای از زمان محاسبه نماییم. ما ابتدا مشکل اندازه بافر را برای تولید کننده و مصرف کننده به طور کلی توضیح می دهیم.

ما دو ورودی و خروجی آرایه را برای ارائه الگوهای تولید یک تولیدکننده و مصرف کننده معرفی می نمایم که در آن:

تعریف 1. $Input[t]$ دارای مقدار 1 است اگر تولیدکننده یک مقدار را در زمان t تولید نماید، در غیر اینصورت دارای مقدار 0 است



شکل 3: جریان داده ها و متغیرهای الگوریتم

تعریف 2. $Output[t]$ دارای مقدار 1 است اگر مصرف کننده حاضر به مصرف دقیق یک مقدار در زمان t باشد، در غیر اینصورت دارای مقدار 0 است.

توجه کنید که خروجی، موجودیت مصرف کننده برای حذف داده ها از بافر توصیف می کند. بافر شامل داده ها است یا می تواند خالی باشد. ما $input.t$ را به عنوان تعداد آیتم های داده تولید شده در $[0..t]$ تعریف می نماییم:

$$input.t = \sum_{0 \leq i < t} input[i] \quad (1)$$

به طور مشابه، ما $output.t$ را برای نشان دادن تعداد آیتم های داده مصرف شده در بازه $[0..t]$ تعریف می نماییم. به یاد داشته باشید که آرایه خروجی تنها نشان دهنده اینست که آیا مصرف کننده برای مصرف یک مقدار آماده است، نه مقداری که واقعاً مصرف می شود، زیرا وابسته به موجودیت داده ها در بافر است. برای تطبیق تعریف $output.t$ با خروجی آرایه، ما باید شرط اضافی را برای کنترل این مورد بگنجانیم که آیا داده هایی در حال حاضر در بافر وجود دارد. داده ها در نقطه زمانی j در دسترس است، زمانی که تعداد کلمات تولید شده در بازه $[0..j]$ از

کلمات مصرف شده در بازه $[0..j]$ بیشتر می شود، $input.(j + 1) > output.j$

$$output.t = \sum_{0 \leq j < t \wedge input.(j+1) > output.j} output[j] \quad (2)$$

با استفاده از معادله (1) و معادله (2)، مشخصات اندازه بافر مینیمم مورد نیاز، ماکزیمم تفاوت بین تعداد کلمات تولید شده و مصرف شده در بازه با طول T است:

$$maxbuffer = \max_{0 \leq t < T} \{input.t - output.t\} \quad (3)$$

برای محاسبه این، آثار تولید کننده و مصرف کننده برای یک بازه زمانی می تواند برای محاسبه اندازه بافر مورد نیاز در آن بازه مورد استفاده قرار گیرد. البته هیچ تضمینی وجود ندارد که یک مقدار به دست آمده از یک اثر، اثری از هر گونه دیگر را برآورده سازد و اینکه بافر به اندازه کافی بزرگ برای هر گونه اطلاعات تولید شده در خارج از بازه بزرگ باشد. در بخش بعدی ما نشان می دهیم که چگونه داشتن تولیدکننده دوره ای و رفتار مصرف کننده به اجازه می دهد تا الزامات بافر را برای یک مقدار بینهایت زمان محاسبه نماییم.

در بخش بعدی، ما برای اولین بار محاسبه برای بافرهای NI متصل به IPهای تولیدکننده را در نظر می گیریم. در اینجا IP ها، تولید کننده و NOC، مصرف کننده است. سپس، ما به بافرهای NI متصل به IP های مصرف کننده نگاه خواهیم کرد، که در آن NOC تولید کننده و IP ها مصرف کننده هستند. در هر دو مورد ما تنها الگوی تولید دوره ای را در نظر می گیریم. در بخش 4.3، ما نشان می دهیم که چگونه گسترش الگوریتم به الگوهای دیگر صورت می گیرد.

4.1 محاسبه بافر NI تولید کننده

زمانی در نظر گرفت بافرینگ NI تولیدکننده برای یک ارتباط، ما نیاز به محاسبه اندازه بافر اصلی مستقیم $\beta^{F,M}$ برای پایه ای داریم که درخواست ها روی کانال مستقیم تولید می کند و اندازه بافر پیرو معکوس $\beta^{R,S}$ برای پیرو که داده های پاسخ را روی کانال معکوس تولید می کند. در زیر، ما تنها کانال مستقیم را بررسی می کنیم، زیرا همین الگوریتم برای کانال معکوس استفاده می شود.

الگوریتم برای محاسبه الزامات بافر NI تولیدکننده، یک پیاده سازی ساده از معادله (3)، که در آن دو متغیر آرایه PIP (IP تولید کننده) و PNI (NI تولیدکننده) برای ضبط الگوهای ورودی / خروجی IP و NI استفاده می شوند و دو متغیر در حال اجرای IPprod و NoCCons برای ذخیره تعداد کل کلمات تولید شده توسط IP و مصرف شده توسط NOC به ترتیب استفاده می شوند که منعکس کننده سمت چپ معادلات (1) و (2) به ترتیب می باشد. ما بافر مصرف کننده NI را در همان الگوریتم محاسبه می نماییم که جزئیات آن در بخش بعدی توضیح داده شده

است. ما متغیرها را در شکل 3 تصویری نموده ایم که در آن IPها و NIها نشان داده اند. متغیرهای شمارنده مانند IPProd، NoCCons، توسط کادرهای مربعی نشان داده می شوند، در حالی که متغیرهای آرایه ها در IPها و NIها با براکت در پشت سر آنها نشان داده شده اند.

Ti را دوره بلوک IP تولید کننده داده ها در نظر بگیرید. مصرف کننده در این مورد، NI متصل به IP تولیدکننده، " NI تولید کننده " است. به علت جدول حافظه، NI تولیدکننده نیز دوره ای با یک دوره از To، برابر با تعداد حافظه ها ضرب در مدت زمان یک حافظه است. علاوه بر این در هر دوره، IP تولید کننده و NI تولیدکننده، تعداد ثابتی از کلمات، Di و Do را به ترتیب مصرف/ تولید می کنند. Di مربوط به سلسله تولید کننده، و Do برابر با تعداد کل کلمات اختصاص داده شده برای اتصال در یک چرخش جدول حافظه است.

با استفاده از این اطلاعات، ما می توانیم آرایه های 'PIP' (IP تولید کننده) و 'PNI' (NI تولید کننده) را پر کنیم. به علت تناوب، آرایه ها باید تنها تا زمان دوره های مربوطه خود وجود دارند. آرایه PIP به شرح زیر برای تولید کننده های دوره ای ساخته می شود:

$$PIP[t] = \begin{cases} 1 & \text{if } 0 \leq t < D_i; \\ 0 & \text{if } D_i \leq t < T_i. \end{cases}$$

آرایه PNI، مطابق با تخصیص جدول حافظه شناخته شده برای NI متناظر ساخته می شود:

$$PNI[t] = \begin{cases} 1 \\ 0 \end{cases}$$

اگر مصرف کننده حاضر به مصرف در زمان t باشد؛

در غیراینصورت.

همانطور که در بخش 4 ذکر شد، لازم نیست ما نباید شبیه سازی را برای یک دوره زمانی نامحدود برای محاسبه حداکثر انجام دهیم، به دلیل اینکه رفتار تولید کننده و مصرف کننده تناوبی است. مشاهده بسیار مهم اینست که

رفتار دو فواصل دوره ای به خودی خود پس از مضرب مشترک (LCM) دو دوره تکرار می شود. این ما را به محاسبه حداقل مضرب مشترک T_i و T_o ، T_{lcm} هدایت می کند. بعد از T_{lcm} ، تولید کننده و مصرف کننده دوباره همتراز می شود و این الگو به خودی خود تکرار می شود.

ما در حال حاضر الگوریتم 1 را برای محاسبه الزامات بافر تولیدکننده ارائه می دهیم. خطوط 1 تا 7 مقداردهی اولیه است. توجه داشته باشید که در خط 1 ما باید دوره مصرف کننده را در محاسبه LCM، در زمان در نظر گرفتن بافرینگ NI مصرف کننده در بخش بعدی اتخاذ نماییم. خط 8 نشان دهنده حلقه زمانی است که دارای lcm تکرار است تا ماکزیمم تفاوت بین تولیدکننده و مصرف کننده در بازه زمانی $[0..lsm]$ محاسبه شود. در خط 9، تعداد کلمات تولید شده تا زمان n توسط اضافه نمودن $PIP[n\%T_i]$ به IPProd به روزسازی می شود.

خط 10 کنترل می کند که در حال حاضر کلماتی در بافر NI تولیدکننده وجود دارد یا خیر. اگر وجود داشته باشد، در خط 11، متغیر NoCCons توسط اضافه نمودن $PNI[n\%T_o]$ به روزسازی می شود (که در صورتی 1 است که NI بتواند داده ها را در آن نقطه ارسال کند و در غیراینصورت صفر است). در خط 12، متغیر NocArriving برای نشان دادن این مورد که یک کلمه داده به زمان مصرف کننده T_{fwd} NI (زمانی که برای عبور از شبکه طول می کشد) از حال حاضر خواهد رسید، یا خیر، به روزسازی می شود. این متغیر برای ضبط الگوی تولید در NI مصرف کننده استفاده می شود و با جزئیات بیشتر در بخش بعدی توضیح داده خواهد شد. نهایتاً در خط 14، ما می بینیم که اگر تفاوت کنونی بین تعداد کلمات تولید شده و تعداد کلمات مصرف شده بزرگتر از ماکزیمم کنونی باشد، و اگر ما آن را در خط 15 جایگزین نماییم. خطوط 17 تا 29 برای محاسبه بافر NI مصرف کننده استفاده می شوند و در بخش بعدی توضیح داده خواهند شد.

الگوریتم 1، الزامات بافر را در NIهای تولیدکننده و مصرف کننده برای یک تولیدکننده و مصرف کننده دوره ای

محاسبه می کند

Require: Arrays $PIP[0..T_i]$, $PNI[0..T_o]$, $CNI[0..T_o]$ and $CIP[0..T_c]$

- 1: $lcm \leftarrow$ the least common multiple of T_i , T_o and T_c
- 2: $IPProd \leftarrow 0$
- 3: $NoCCons \leftarrow 0$
- 4: $NoCProd \leftarrow 0$
- 5: $IPCcons \leftarrow 0$
- 6: $maxprodbuffer \leftarrow 0$
- 7: $maxconsbuffer \leftarrow 0$
- 8: **for** $n = 0$ **to** $lcm - 1$ **do**
- 9: $IPProd \leftarrow IPProd + PIP[n \% T_i]$
- 10: **if** $IPProd - NoCCons > 0$ **then**
- 11: $NoCCons \leftarrow NoCCons + PNI[n \% T_o]$
- 12: $NoCArriving[n + T_{Fwd}] \leftarrow PNI[n \% T_o]$
- 13: **end if**
- 14: **if** $IPProd - NoCCons > maxprodbuffer$ **then**
- 15: $maxprodbuffer \leftarrow IPProd - NoCCons$
- 16: **end if**
- 17: $NoCProd \leftarrow NoCProd + NoCArriving[n]$
- 18: **if** $NoCProd - IPCcons > 0$ **then**
- 19: $IPCcons \leftarrow IPCcons + CIP[n \% T_c]$
- 20: $credit \leftarrow credit + CIP[n \% T_c]$
- 21: **end if**
- 22: **if** $CNI[n \% T_o] \wedge credit > 0$ **then**
- 23: $creditArriving[n + T_{Rev}] \leftarrow credit$
- 24: $credit \leftarrow 0$
- 25: **end if**
- 26: $CreditsReceived \leftarrow CreditsReceived$
 $+ creditArriving[n]$
- 27: **if** $NoCProd - CreditsReceived > maxconsbuffer$
then
- 28: $maxconsbuffer \leftarrow NoCProd - CreditsReceived$
- 29: **end if**
- 30: **end for**

توجه کنید که محاسبه $IPProd$ تنها وابسته به خود تولیدکننده است، اما تعداد کلمات واقعاً مصرف شده ($NoCCon$ ها) نیز وابسته به موجودیت اعتبارهای کنترل جریان است. ما با کنترل جریان در بافرینگ NI مصرف کننده سروکار داریم و مطمئن می شویم که بافر NI مصرف کننده (و از اینرو اعتبارهای کافی برای NI تولیدکننده در دسترس هستند) برای حفظ توان عملیاتی مورد نیاز به اندازه کافی بزرگ است.

4.2 بافرینگ NI مصرف کننده

زمانی که داده ها از NI تولیدکننده فرستاده می شود، بعد از زمان T_{fwd} به NI مصرف کننده می رسد. ما فرض می کنیم که می توانیم رفتار مصرف کننده را مشابه با رفتار تولیدکننده مشخص نماییم: مصرف کننده با دوره T_c متناوب است و سلسله هایی با اندازه DC را در آغاز هر دوره T_c مصرف می کند. ما این اطلاعات را در CIP آرایه ذخیره می نماییم. تعداد کل کلمات تولید شده و مصرف شده توسط NI و IP به ترتیب در NoCProd و IPCons ذخیره می شوند. هنگامی که مصرف کننده داده ها را مصرف می کند، یک اعتبار را به صورت نشان داده شده در شکل 3 تولید می کند. اعتبارات به اولین حافظه ارسال می شوند که برای اتصال در NI مصرف کننده در دسترس است (جدول حافظه مصرف کننده NI در آرایه CNI). هر زمان که اعتبار فرستاده می شود، آنها هنوز ثابت T_{rev} تاخیر شبکه را قبل از اینکه NI تولیدکننده آنها را دریافت کند، اتخاذ می کنند و تعداد کل اعتبار دریافت شده در یک متغیر CreditsReceived ذخیره می شود. از این رو NI تولیدکننده، اعتبارات را به محض اینکه داده ها مصرف می شوند دریافت نمی کند بلکه تنها پس از تاخیر، و در عین حال یک سلسله جدیدی از داده ها از تولید می تواند دریافت شود. از آنجا که در بافرینگ NI تولیدکننده ما فرض کردیم که اعتبارات به اندازه کافی در دسترس (به منظور اجتناب از اصطبل) هستند، باید مطمئن شویم که بافر NI مصرف کننده برای دریافت کلمات اضافی از داده ها به اندازه کافی بزرگ است در حالی که اعتبارها در راه هستند. این ما را به محاسبه حداکثر اختلاف بین تعداد کلمات تولید شده توسط شبکه و تعداد اعتبارهای دریافت شده توسط تولید کننده NI (به جای استفاده از تعداد کلمات مصرف شده توسط مصرف کننده) هدایت می کند.

خطوط 17-29 از الگوریتم 1 برای محاسبه اندازه مورد نیاز بافر مصرف کننده NI استفاده می شوند. خط 17 برای به روز رسانی متغیر NoCProd برای منعکس کردن تعداد کلماتی است که شبکه تا زمان t، با اضافه کردن متغیر NoCArriving [N] به آن تولید کرده است. این متغیر زمانی برابر 1 تنظیم می شود که تولیدکننده NI، یک زمان کلمه T_{fwd} را زودتر در خط 12 تولید می کند. در خط 18، پر کردن کنونی بافر مصرف کننده NI تعیین می شود. اگر این تفاوت بزرگتر از صفر باشد، CIP [n%Tc] به IPCons در خط 19 اضافه می شود. (نشان می دهد که

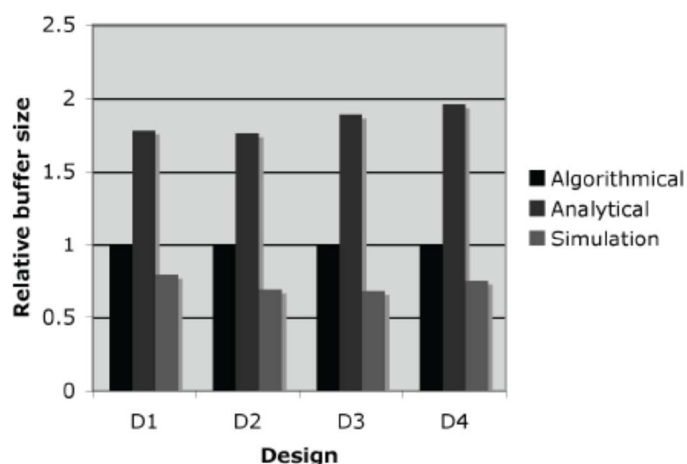
آیا مصرف کننده IP در زمان n مصرف می کند یا خیر). به منظور مدلسازی مکانیزم اعتبار، ما $CIP [n\%Tc]$ را به متغیر اعتبار اضافه می کنیم که به طور موثر یک اعتبار را می افزاید هر زمان که مصرف کننده یک کلمه را مصرف می کند. اعتبارات به NI تولیدکننده فرستاده می شوند، هر زمان که یک حافظه در دسترس وجود دارد که در خط 22 کنترل می شود. اگر یک حافظه در دسترس باشد، در خط 23، آرایه `creditArriving` برای منعکس نمودن رسیدن یک اعتبار در NI تولیدکننده در زمان $n + T_{Rev}$ به روزسازی می شود. خط 24، شمارنده اعتبار را بعد از اینکه اعتبارها فرستاده می شوند، ریست می کند. در خط 26، تعداد اعتباراتی که ناشی از رسیدن به NI تولیدکننده هستند، به متغیر `CreditRecieved` اضافه می شوند. نهایتاً، تفاوت بین تعداد کلمات تولید شده توسط شبکه و تعداد اعتبارات دریافت شده محاسبه می شود و ماکزیمم ذخیره می شود.

در بخش 2، ما ذکر کردیم که در هر حافظه، یک هدر بسته باید قبل از هر داده فرستاده شود، مگر اینکه یک اتصال، حافظه قبلی را اعطا کند. ما همچنین ذکر نمودیم که اعتبارات کنترل جریان در بخش رزرو شده هدر بسته با اندازه 5 بیت ذخیره می شوند [14] که تنها ماکزیمم 32 اعتبار را برای فرستاده شدن در هر حافظه مجاز می سازد. در حالیکه در الگوریتم 1 برای سادگی نشان داده نشده است، این موضوعات در پیاده سازی ما در نظر گرفته می شوند.

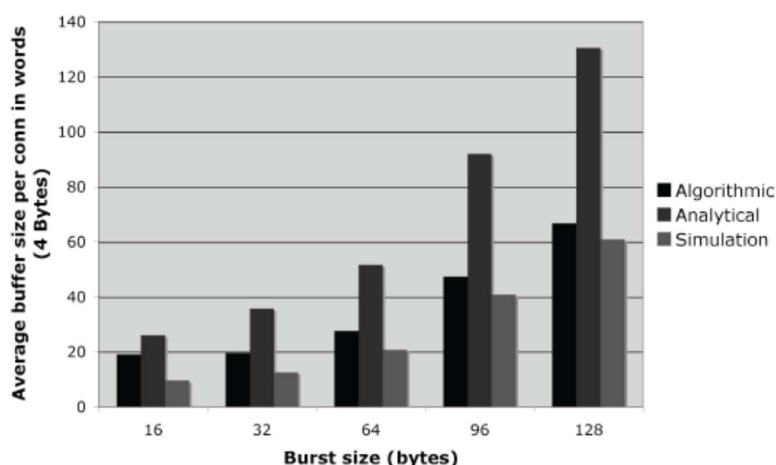
تولیدکننده، NOC و رفتار دوره ای مصرف کننده نباید همراستا باشند. بنابراین ما دو تا از سه آرایه ورودی را جابجا می نماییم و دوباره الگوریتم را برای هر ترکیب اجرا می کنیم. بنابراین اندازه مینیمم بافر را برای تمام همراستایی های ممکن بازه های دوره ای محاسبه می نماییم. بنابراین پیچیدگی زمانی این الگوریتم، $O(T_i \times T_o \times (lcm(T_i, T_o, T_c)))$ و بنابراین چندجمله ای است.

4.3 دیگر الگوهای تولید

در بخش 3، ما دو الگوی تولید دیگر را ذکر نمودیم،



شکل 4: اندازه های نسبی بافر برای طرح های مختلف



شکل 5: اثر اندازه سلسله روی اندازه بندی بافر

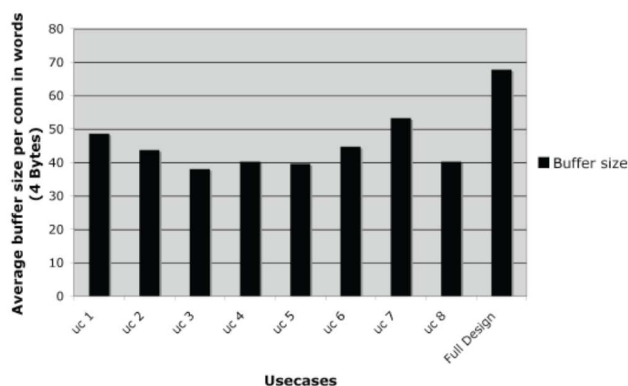
الگوی تولید غیردوره ای و الگوی تولید چند دوره ای. هر دوی آنها، به آسانی می توانند در الگوریتم ارائه شده گنجانده شوند. برای الگوی غیردوره ای، ما بدترین موردی که می تواند رخ دهد را در نظر می گیریم: در هر بازه زمانی با طول 2 دوره (لزوماً یک پنجره لغزشی)، ما کزیمم سه سلسله که می تواند برسد. سپس ما الگوی غیردوره ای را به صورت دوره ای با دو برابر نمودن دوره و سلسله را به اندازه سه برابر مدلسازی می نماییم. این مورد می تواند در آرایه تولیدکننده به آسانی منعکس شود. در حالیکه این مانند سربارهای زیاد به نظر می رسد، در عمل، دوره NI

کوچکتر از دوره IPها است. در مدت یک دوره از بلوک IP، NI می تواند دارای 10 دوره باشد که در آن نصف داده های زمانی مصرف می شوند، و در نتیجه نیاز به بافر بسیار کوچکتر دارد.

برای الگوی تولید چند دوره ای، ما می توانیم بالاترین سطح را در نظر بگیریم که الگوی تولید، دوره ای است. برای سیستم نمایش ویدئوی ذکر شده در بخش 3، این دوره چارچوب خواهد بود. سپس می توانیم آرایه تولیدکننده را با طول این دوره خلق نماییم و سپس در هر سلسله با تنظیم مقادیر برابر با 1، هر زمانی که 1 کلمه از داده ها در این دوره منتقل می شود، پر نماییم.

5. نتایج

ما روش توصیف شده در این مقاله را با روش تحلیلی در [2] و با شبیه سازی های برنامه ها مقایسه نمودیم. ما روش های اندازه بندی بافر را با استفاده از دو طرح مختلف SoC در محل، یک کادر SoC تنظیم شده در بالا (D1) و SoC چندرسانه ای برای تلفن ها (D2) محک زدیم. هر یک از این طرح ها شامل تعداد زیادی از هسته های IP (+10) و تعداد زیادی از اتصالات (+20) می شود. علاوه بر این دو طرح، ما مقدار زیادی از طرح های ساختگی تقسیم شده در دو رده را تولید نمودیم: طرح هایی با هسته IP تنگنا مانند حافظه (D3) و طرح هایی با ارتباطات گسترده (D4). ما اندازه های بافر را برای هر یک از این طرح ها با استفاده از روش های تحلیلی و الگوریتمیک محاسبه نمودیم و آنها را با ماکزیمم پر کردن بافر مشاهده شده در مدت یک شبیه سازی مقایسه نمودیم.



شکل 6: اندازه بافر برای چند مورد استفاده

نتیجه در شکل 4 نشان داده شده است. همانطور که دیده می شود، روش الگوریتمی بهتر از روش تحلیلی عمل می کند، به طور متوسط بافر 84 درصد کوچکتر می شود. دلیل اصلی این تفاوت بزرگ این است که روش تحلیلی این دوره های تولید کننده و مصرف کننده و هم ترازوی خود را به حساب نمی آورد، بلکه از یک الزام بافر بدترین حالت معادل با مجموع اندازه های سلسله تولیدکننده و مصرف کننده استفاده می کند $(D_i + D_o)$ [2].

ماکسیمم به دست آمده در طول شبیه سازی به طور متوسط 38٪ کوچکتر می باشد. دلیل آن این است که شبیه سازی فقط یک اثر است، و این همترازی بدترین حالت که الگوریتم محاسبه می کند همیشه رخ نمی دهد. تکیه بر یک اثر / شبیه سازی برای بافر اندازه ممکن است به بافرهایی منجر شود که خیلی کوچک هستند. همچنین، بافر مصرف کننده اضافه شده برای مخفی کردن تاخیر کنترل جریان رفت و برگشت انتها به انتها همواره مورد استفاده قرار نمی گیرد، زیرا داده ها در بافر مصرف کننده ممکن است قبل از اینکه داده های جدید برسند، مصرف شوند.

شکل 5 نشان می دهد که اندازه سلسله روی اندازه بافر تاثیر می گذارد. ما 100 موارد استفاده مصنوعی را برای هر اندازه سلسله ایجاد نمودیم، و اندازه بافر را برای هر یک از آنها محاسبه نمودیم. شکل نشان دهنده متوسط اندازه بافر در هر اتصال برای هر دو روش تحلیلی و الگوریتم در مقایسه با حداکثر مشاهده شده در طول شبیه سازی است. هنگامی که اندازه های سلسله بسیار بزرگ (128 بایت و بیشتر) باشند، روش الگوریتمی فقط 10٪ به بافر مشاهده شده در طول شبیه سازی اضافه می کند. دلیل آن این است که افزایش حجم سلسله در حالی که پهنای باند حفظ می شود، دوره را نیز افزایش می دهد. نوعاً در این دوره، تنها یک سلسله تک باید بافر شود و اعتبارات کنترل جریان می توانند قبل از آغاز دوره بعدی پشتیبانی شوند. زمانی که اندازه سلسله کوچک است، بافرینگ اضافی در بافر مصرف کننده برای کنترل جریان نوعاً در مقایسه با اندازه سلسله بزرگ است.

شکل 6 اندازه بافر میانگین را در هر اتصال برای هر یک از 8 مورد استفاده از SOC واحد نشان می دهد. اندازه بافر NI برای کل SOC، حداکثر اندازه بافر برای همه موارد استفاده برای هر اتصال است. ستون "طراحی کامل" نتیجه را

نشان میدهد. چون بافرهای اتصال ممکن در موارد استفاده مختلف، بزرگترین باشد، طراحی کامل (به اندازه 27٪) بزرگتر از بزرگترین مورد استفاده (uc7) است.

زمان اجرا به طور معمول در مرتبه چند دقیقه است، به جز زمانی که یک طرح دارای اتصالات با الزامات پهنای باند کم (کمتر از یک مگابایت / ثانیه) است. دوره های بزرگ به یک LCM بزرگ منجر می شود. بهینه سازی الگوریتم برای عمل بهتر برای چنین قابلیت اتصال، بخشی از کار در آینده است.

6. نتیجه گیری ها و کار آینده

NOCهایی که خدمات تضمین شده را ارائه می دهند، برای طراحی SOC آینده با الزامات زمان واقعی ضروری هستند. به منظور ارائه این خدمات برای هسته های IP، NoC باید شامل بافرهای جداکننده تجزیه برای مخفی کردن تفاوت در سرعت عامل بین هسته IP و NoC باشد. این بافرها عامل غالب در سطح و توان NoC می باشند. به حداقل رساندن بافرها در حالی که هنوز مطابق با رفتار برنامه مورد نیاز باشند، یک مسئله مهم است.

در این مقاله ما روش طراحی جدید برای اندازه بافر جداسازی را در NIها از *Ethereal NoC* معرفی نمودیم. این روش از دانش مربوط به رفتار هسته های IP و NoC استفاده می کند و می توانیم سطح بافر را در طرح ها به طور متوسط تا 84٪ کاهش دهیم، هنگامی که با روش بدترین حالت تحلیلی مقایسه می شود. این روش همچنین اثرات پیچیده از کنترل جریان مبتنی بر اعتبار آنها به آنها را در اندازه های بافر مورد نیاز اتخاذ می کند. این روش برای همه برنامه ها، سریع در نظر گرفته می شود، و از طیف گسترده ای از رفتارهای برنامه پشتیبانی می نماید. در نهایت، این روش می تواند موارد استفاده متعدد را برای طراحی SOC تک در نظر بگیرد.

کار آینده شامل پژوهش در زمینه اثر این همراستایی خوب بین هسته های IP و NoC و در نتیجه کاهش الزامات بافر می شود.



این مقاله، از سری مقالات ترجمه شده رایگان سایت ترجمه فا میباشد که با فرمت PDF در اختیار شما عزیزان قرار گرفته است. در صورت تمایل میتوانید با کلیک بر روی دکمه های زیر از سایر مقالات نیز استفاده نمایید:

لیست مقالات ترجمه شده ✓

لیست مقالات ترجمه شده رایگان ✓

لیست جدیدترین مقالات انگلیسی ISI ✓

سایت ترجمه فا ؛ مرجع جدیدترین مقالات ترجمه شده از نشریات معتبر خارجی