علوم رایانش و فناوری اطلاعات
نشریه علمی انجمن کامپیوتر ایران
مجلد ۱۷، شماره ۱، ۱۳۹۸
صفحات ۱۰۶-۹۴
مقاله عادی

مدیریت اوج توان در سیستم‌های رزرو-آماده‌باش

محسن انصاری [۱]، مصطفی پسندیده [۲]، علیرضا اجلالی *[۳]

*نویسنده مسئول، دریافت: ۹۷/۰۷/۱۵، بازنگری: ۹۷/۱۲/۰۲، پذیرش: ۹۸/۰۳/۰۷

[۱] دانشجوی دکتری، مهندسی کامپیوتر، دانشکده‌ی مهندسی کامپیوتر، دانشگاه صنعتی شریف، تهران، ایران
[۲] دانشجوی کارشناسی ارشد، مهندسی کامپیوتر، دانشکده‌ی مهندسی کامپیوتر، دانشگاه صنعتی شریف، تهران، ایران
[۳] دانشیار، مهندسی کامپیوتر، دانشکده‌ی مهندسی کامپیوتر، دانشگاه صنعتی شریف، تهران، ایران

چکیده

سیستم‌های بی‌درنگ عمدتاً باید دارای قابلیت اطمینان بالا باشند. افزایش تعداد هسته‌های پردازشی در یک تراشه امکان استفاده از روش‌های تحمل‌پذیری اشکال مبتنی بر افزونگی سخت‌افزاری را برای این سیستم‌ها فراهم می‌کند؛ اما این روش‌ها ممکن است باعث افزایش مصرف توان و عبور اوج توان مصرفی از توان حرارتی طراحی (TDP) شوند. توان حرارتی طراحی آن حد از دما است که واحد خنک‌کننده می‌تواند برای آن جوابگو باشد. در صورت عبور دما از این حد، سیستم (بدون کنترل طراح سیستم) راه‌اندازی مجدد می‌شود یا کارایی خود را به‌شدت کاهش می‌دهد تا از آسیب دائمی جلوگیری گردد. ازاین‌رو، مطالعه‌ی تأثیر روش‌های تحمل‌پذیری اشکال بر اوج توان مصرفی و ارائه‌ی روش‌هایی برای دست‌یابی به تحمل‌پذیری اشکال در سیستم‌های بی‌درنگ تحت محدودیت توان حرارتی طراحی حائز اهمیت است. در این پژوهش، یک روش مدیریت اوج توان مصرفیِ آگاه به قابلیت اطمینان برای سیستم‌های چندهسته‌ای بی‌درنگ که از افزونگی سخت‌افزاری رزرو-آماده‌باش استفاده می‌کنند، ارائه می‌شود.

کلمات کلیدی: اوج توان مصرفی، توان حرارتی طراحی، سیستم‌های بی‌درنگ، تحمل‌پذیری اشکال

۱- مقدمه

با پیشرفت فناوری ساخت CMOS، تراشه‌های چندهسته‌ای به‌طور گسترده در سیستم‌های بی‌درنگ به دلیل پتانسیل آن‌ها برای دست‌یابی به کارایی بالا به همراه هزینه‌ی کم، استفاده می‌شوند [۲۶]. زمانی که اندازه‌ی ترانزیستورهای CMOS به‌صورت پیوسته کوچک می‌شود، اکثر هسته‌ها و دستگاه‌های الکترونیکی روی یک تراشه مجتمع می‌شوند. ازآنجایی‌که چگالی توان[۱] برای یک تراشه‌ی چندهسته‌ایِ پیشرفته به‌سرعت در حال رشد است، باید مدیریت توان به‌عنوان یکی از مسائل بحرانی که بر روی کارآمدی خنک‌سازی تراشه و قابلیت اطمینان[۲] تراشه تأثیرات مخرب می‌گذارد، در نظر گرفته شود. برای این منظور، بدون نقض نیازمندی‌های زمانی، روش‌های مدیریت توان متعددی برای کمینه کردن انرژی مصرفی در [۳۲] [۳۵] یا برای کمینه کردن دمای تراشه [۴] [۱۰] [۴۱]، به‌وسیله‌ی کنترل کردن متوسط توان مصرفی ارائه شده است. بنابراین مدیریتِ توان مصرفی یک تراشه‌ی چندهسته‌ای یک مسئله‌ی مهم در طراحی سیستم‌های بی‌درنگ است. لازم به ذکر است که اکثر روش‌های موجود از طریق کاهش انرژی مصرفی[۳] کل و یا کاهش دمای تراشه[۴]، متوسط توان مصرفی[۵] را کاهش می‌دهند. حال آن‌که در این بین توجه اندکی به بدترین رفتار لحظه‌ایِ توان مصرفی[۶] در یک تراشه (اوج توان مصرفی[۷] در سطح تراشه) شده است.

بدترین رفتار لحظه‌ایِ توان مصرفی یک پارامتر مهمِ طراحی در تعیین هزینه، سایز تراشه و منبع تغذیه است. روند افزایشی مجتمع‌سازی در تراشه‌های چندهسته‌ای، باعث افزایش منطق دیجیتال و به‌تبع آن افزایش توان مصرفی کل تراشه خواهد شد. این افزایشِ توان تحت عنوان سیلیکون تاریک[۸] بیان می‌گردد و بدین معناست که در سیستم‌های چندهسته‌ای، درصد مهمی از هسته‌های در دسترس، به دلیل محدودیت دمایی نمی‌توانند به‌طور هم‌زمان فعال باشند [۸] [۱۵] [۴۰]. به‌طورمعمول سازندگان تراشه‌های چندهسته‌ای، عددی تحت عنوان توان حرارتی طراحی[۹] برای هر تراشه ارائه می‌دهند. مقدار TDP، حداکثر توانِ مصرفی قابل قبولی است که تراشه می‌تواند بدون استفاده از مکانیزم‌های تنزل کارایی[۱۰] مصرف کند [۱۷]؛ ازجمله‌ی این مکانیزم‌ها می‌توان به تکنیک مدیریت حرارتی پویا[۱۱] اشاره نمود.

در سیستم‌های بی‌درنگ سخت به دلیل اثر منفیِ وجود واحدِ خنک‌سازِ فعال بر قابلیت اطمینان، معمولاً در بسیاری از موارد خنک‌ساز فعال وجود ندارد و از خنک‌ساز غیرفعال استفاده می‌شود که موجب کاهش قابل‌توجه TDP می‌گردد.

دسته‌ی مهمی از سیستم‌های توزیع‌شده، سیستم‌هایی هستند که در کاربردهایی با محدودیت‌های بی‌درنگی سخت[۱۲] به کار گرفته می‌شوند. این دسته از سیستم‌ها دارای قابلیت اطمینان بالایی هستند که از طریق تحمل‌پذیری اشکال حاصل می‌شود. سیستم‌های بی‌درنگ سخت با قابلیت تحمل‌پذیری اشکال[۱۳] به‌گونه‌ای طراحی و ساخته می‌شوند که دارای سه ویژگی باشند: ۱) کار تخصیص داده شده به این سیستم‌ها، به‌درستی و با صحتِ کامل انجام شوند، ۲) قیود زمانی را برآورده کرده، به‌طوری‌که ضرب‌الاجل[۱۴] مربوط به بی‌درنگی سخت نقض نشود [۲۸] [۳۴] [۱۹] [۲۴] و ۳) محدودیتِ TDP مربوط به تراشه رعایت شود [۴۵] [۲۶]؛ زیرا با نقض هریک از موارد بالا ممکن است این سیستم‌ها با تنزل قابلیت اطمینان روبرو شوند. در این نوع از سیستم‌ها، تحمل‌پذیری اشکال از راه تکرار وظایف مبتنی بر افزونگی سخت‌افزاری به دست می‌آید. از سوی دیگر روش‌های مبتنی بر تکرار وظایف، باعث تحمیل سربارِ توان مصرفی بسیار قابل‌توجهی به سیستم می‌شوند.

معمولاً برای تحمل‌پذیری اشکال در سیستم‌های بی‌درنگ سخت از افزونگی سخت‌افزاری و افزونگی زمانی استفاده می‌شود. افزونگی زمانی از زمان بیکاری[۱۵] برای بازگشت از اشکال[۱۶] استفاده می‌کند که نسبت به افزونگی سخت‌افزاری از سربار سخت‌افزاری کمتری برخوردار است [۲۵] [۳۳]. این افزونگی فقط زمانی کاربرد دارد که به‌قدر کافی زمان برای اجرای مجددِ وظایفی که دچار اشکال شده‌اند، وجود داشته باشد [۶]. در طرف مقابل با استفاده از افزونگی سخت‌افزاری، معمولاً چند نسخه از یک وظیفه به‌طور هم‌زمان یا با همپوشانی در هنگام اجرای وظایف اجرا می‌شوند. در این افزونگی به دلیل استفاده از چند سخت‌افزار به‌طور هم‌زمان، زمان بیکاری مناسبی برای ایجادِ تحمل‌پذیری اشکال فراهم می‌گردد. به‌این‌ترتیب در سیستم‌های بی‌درنگِ سخت، استفاده از افزونگی سخت‌افزاری می‌تواند مفید باشد [۲۵]. درهرصورت، اگر در استفاده از افزونگی سخت‌افزاری دقت کافی نشود، ممکن است اوج توان مصرفی سیستم از محدودیت TDP که برای تراشه تعریف شده است، عبور کند و باعث نقض محدودیت مذکور شود. با نقض این محدودیت، قابلیت اطمینان سیستم به‌طور قابل‌توجهی کاهش پیدا می‌کند؛ زیرا ممکن است برخی از هسته‌ها راه‌اندازی مجدد شده یا به‌طور کامل از کار بیافتند. برای مثال فرض کنید در یک سیستم که دارای دو هسته است از تکنیک مشهور رزرو-آماده‌باش[۱۷] استفاده شود [۶] [۲۵] [۱۳]؛ در این سیستم یکی از هسته‌ها، هسته‌ی اصلی و دیگری هسته‌ی رزرو است که به ترتیب وظایف اصلی[۱۸] و پشتیبان[۱۹] را اجرا می‌کنند. اگر در این سیستم دو وظیفه به ترتیب بر روی هسته‌ی اصلی و هسته‌ی رزرو به‌طور هم‌زمان اجراAl شوند ممکن است که اوج توان مصرفی آن‌ها باعث شود که محدودیت TDP نقض گردد، بنابراین باید برای زمان‌بندی این سیستم یک سیاست مناسبی اندیشیده شود. بنابراین برای حل مشکلِ اوجِ توان مصرفی در سیستم‌های تحمل‌پذیر اشکال، در این پژوهش مدیریتِ اوجِ توانِ مصرفی در سطحِ تراشه در زمان طراحی و بدون نقض محدودیت‌های بی‌درنگی انجام می‌گیرد. برای دست‌یابی به این منظور، زمان‌بندی وظایفِ بی‌درنگ، بدون تکیه بر پیاده‌سازی سخت‌افزارِ اضافی برای مدیریت توان همانند تکنیک تغییر پویای ولتاژ و فرکانس[۲۰] صورت خواهد گرفت.

۲- بررسی کارهای پیشین

به‌طورکلی پژوهش‌های مرتبط در راستای مدیریت اوج توان مصرفی در سیستم‌های تحمل‌پذیر اشکال، به سه دسته تقسیم‌بندی می‌شوند. این سه دسته عبارت‌اند از: ۱) مدیریت اوج توان مصرفی در سیستم‌های بدونِ هیچ روش تحمل‌پذیری اشکال؛ ۲) مدیریت توان مصرفی متوسط در سیستم‌های تحمل‌پذیر اشکال؛ ۳) مدیریت دمای تراشه در سیستم‌های بی‌درنگ. بنابراین به ترتیب پژوهش‌های انجام شده در سه دسته‌ی مذکور را به‌طور مختصر بررسی می‌نماییم. لازم به ذکر است که تاکنون هیچ پژوهشی بر روی مدیریت اوج توان در سیستم‌های تحمل‌پذیر اشکال صورت نگرفته است. ازاین‌رو ابتدا پژوهش‌هایی که اوج توان مصرفی در سیستم‌های بی‌درنگ را در نظر گرفته‌اند، مورد مطالعه قرار می‌دهیم.

تا به امروز، مطالعاتی در زمینه‌ی این‌که چطور می‌توان اوج توان مصرفی را برای سیستم‌هایی با اهداف عمومی[۲۱] مدیریت کرد، صورت گرفته است [۱۸] [۲۹] [۲۳]، اما مشخص نیست که چطور می‌توان آن‌ها را با قیود زمانی بی‌درنگ وفق داد. هدف پژوهش [۲۶] کمینه کردن اوج توان مصرفی سطح تراشه در زمان طراحی بدون نقض محدودیت‌های زمانی است. برای دست‌یابی به این هدف، پژوهش [۲۶] فقط بر روی زمان‌بندی وظایف در سطح نرم‌افزار بدون هیچ سخت‌افزار اضافی همانند تکنیک مدیریت تغییر پویای ولتاژ و فرکانس تکیه می‌کند. تکنیک مدیریت تغییر پویای ولتاژ و فرکانس به‌صورت گسترده برای مدیریت توان در ریزپردازنده‌ها استفاده می‌شود [۱۰] [۴] [۳۲]، اما از سربار کلیدزنی غیر صفر تغییرِ ولتاژ/فرکانس رنج می‌برند. بنابراین، یکی از مزایای اصلی این پژوهش سادگی و کاربردپذیری آن در بسترهای چندهسته‌ای بدون نیاز به پشتیبانی از سخت‌افزار اضافی است.

در پژوهش [۴۵]، یک روش برای کاهش اوج توان مصرفی برای کاربردهای بی‌درنگ متناوب و مبتنی بر قاب[۲۲] بر روی سیستم‌های چندهسته‌ای به‌وسیله‌ی زمان‌بندی زمان‌های بیکاری هسته‌های فعال ارائه شده است. همچنین برای چهار حالت مختلفِ وظایف و هسته‌ها، الگوریتم‌های زمان‌بندی مختلفی مطرح نموده است. این چهار حالت به شرح زیر هستند: ۱) هسته‌ها و وظایف همگن باشند؛ یعنی تمام وظایف بر روی تمام هسته‌ها دارای اوج توان مصرفی یکسانی می‌باشند؛ در این حالت با توجه به میزانِ بهره‌وری[۲۳] هر هسته، تمام وظایف را در یک زمان‌بند ساده، زمان‌بندی می‌کند و اثبات می‌کند که اگر ضرب مقدار عددی توان (برای تمام هسته‌ها و وظایف ثابت است) در میزان بهره‌وری کل سیستم کمتر از مقدار عددی TDP باشد، زمان‌بندی ارائه شده دو شرط زمانی و محدودیت TDP را رعایت می‌کند. ۲) هسته‌ها ناهمگن ولی وظایف همگن باشند؛ در این حالت به ازای یک وظیفه‌ی مشخص، تمام هسته‌ها اوج توان مختلفی دارند ولی دو وظیفه‌ی مختلف بر روی یک هسته‌ی خاص اوج توان مصرفی یکسانی دارند. راه‌حل ارائه شده برای این مورد بدین‌صورت است که ابتدا هسته‌ای که بیشترین اوج توان را مصرف می‌کند انتخاب کرده و وظایف تخصیص داده شده به آن هسته، در فضاهای زمانی که کمترین میزان اوج توان مصرفی را دارند زمان‌بندی می‌کند، پس از هسته‌ی مذکور، هسته‌ی بعدی که از نظر اوج توان مصرفی، بیشترین مقدار عددی را دارد انتخاب می‌نماید و این روند آن‌قدر ادامه پیدا می‌کند تا تمام وظایف تخصیص داده‌شده به هسته‌ها زمان‌بندی شوند. ۳) هسته‌ها همگن و وظایف ناهمگن باشند؛ یعنی به ازای یک وظیفه‌ی مشخص، تمام هسته‌ها اوج توان یکسانی دارند ولی دو وظیفه‌ی مختلف اوج توان مختلفی دارند. ۴) هسته‌ها ناهمگن و وظایف ناهمگن باشند؛ یعنی به ازای یک وظیفه‌ی مشخص، تمام هسته‌ها اوج توان مصرفی متفاوت و دو وظیفه‌ی مختلف نیز اوج توان مختلفی داشته باشند. برای مورد ۳ و ۴ یک الگوریتم مطرح شده است. در این الگوریتم ابتدا وظایف بر اساس میزان اوج توان مصرفی مرتب شده و وظیفه‌ای که بیشترین اوج توان را دارد انتخاب شده و سپس زمان‌بندی می‌شود. این روند آن‌قدر ادامه پیدا می‌کند تا تمام وظایف زمان‌بندی شوند.

در پژوهش [۲۷] یک زمان‌بندی پویا با در نظر گرفتن وابستگی داده‌ای[۲۴] برای جلوگیری از رخداد اوج توان مصرفی در سیستم‌های چندهسته‌ای ارائه شده است. درواقع مدل وظایفِ این پژوهش، از نوع گراف وظیفه[۲۵] است و با استفاده از سه مرحله، از رخداد اوج توان جلوگیری می‌کند. ابتدا با توجه به نمودار توان مصرفی هر وظیفه، اوج توان را در بازه‌های زمانی که بر اساس زمان پایه‌ای خود به دست می‌آورد، با سه مقدار ۱-، ۰ و ۱+ بر اساس شیب مربوط به نمودار توان مصرفی مقداردهی می‌کند. سپس از رخداد وظایفی که با شیب ۱+ یا ۰ می‌خواهند هم‌زمان اجرا شوند، جلوگیری می‌کند.

تابه‌حال پژوهش‌های صورت گرفته بر روی سیستم‌های تحمل‌پذیر اشکال به‌منظور کاهش متوسط توان مصرفی بوده است که پژوهش‌های [۶] [۱۳] [۱۴] با رویکرد کاهش متوسط توان مصرفیِ سیستم رزرو-آماده‌باش، از این دسته‌اند. در پژوهش [۶] یک روش مناسب برای مدیریت انرژی و قابلیت اطمینان در قالبِ یک سیستم رزرو-آماده‌باش ارائه شده است. ایده‌ی این روش ترکیب افزونگی سخت‌افزار با تکنیک تغییر پویای ولتاژ و فرکانس برای ذخیره‌ی انرژی است درحالی‌که قابلیت اطمینان اولیه‌ی سیستم حفظ شود. در این روش از دو پردازنده استفاده شده است که به ترتیب اصلی و رزرو نامیده می‌شوند. وظایفِ هر کاربرد بر روی پردازنده‌ی اصلی با استفاده از تکنیک تغییر پویای ولتاژ و فرکانس اجرا خواهد شد. پردازنده‌ی رزرو از تکنیک تغییر پویای ولتاژ و فرکانس استفاده نمی‌کند و در فرکانس و ولتاژ حداکثر، وظایف پشتیبان را اجرا خواهد کرد. به‌محض اجرای کامل و موفقیت‌آمیز یک وظیفه-ی اصلی، وظیفه‌ی پشتیبانِ مرتبط اجرا نخواهد شد و از مصرف انرژی اجتناب می-شود. اگر وظیفه‌ی اصلی با موفقیت اجرا نشد، وظیفه‌ی پشتیبان به‌طور کامل در حداکثر فرکانس اجرا خواهد شد.

در پژوهش [۱۳] نیز سیستم رزرو-آماده‌باش در نظر گرفته شده است. راه‌حل این پژوهش بدین‌صورت است که به‌صورت ایستا یک ساختار زمان‌بندی بر روی پردازنده‌ی رزرو که وظایف پشتیبان را به تأخیر می‌اندازد به‌قدری که همپوشانی میان وظیفه‌ی اصلی و وظیفه‌ی پشتیبان حداقل شود، ایجاد می‌کند. این روش مبتنی بر مشاهداتی است که چندین کار بی‌درنگ، زودتر از زمان اجرای واقعی کامل شده‌اند و ازاین‌رو پشتیبان‌ها اغلب بدون حتی شروع برای اجرا، لغو خواهند شد. به‌علاوه، کلِ زمانِ بیکاری پویا و ایستا برای کاهش انرژی بر روی پردازنده‌ی اصلی که هیچ زمانی را برای کارهای پشتیبان رزرو نکرده است، در دسترس است. در پژوهش [۱۴] نیز از الگوریتم زمان‌بندی RMS[۲۶] به همراه تکنیک تغییر پویای ولتاژ و فرکانس و تکنیک مدیریت پویای توان برای کاربردهای بی‌درنگی سخت که اولویت ثابت[۲۷] دارند، استفاده شده است.

[۴۶] یک روش مدیریت انرژی برای سیستم رزرو-آماده‌باش ارائه داده است که در آن از بازخورد برای به دست آوردن زمان اجرای واقعی وظایف استفاده می‌کند. در این روش از یک کنترلر PID برای به دست آوردن رابطه‌ی بین بارکاری در گذشته و آینده و تخمین بارکاری آینده استفاده می‌کند. با این کار مقدار زمان بیکاری که وظایف (در آینده) در اختیار خواهند داشت را با دقت بالایی می‌تواند به دست آورده و این زمان بیکاری را به‌صورت مناسب بین تمام وظایف پخش کند. این سیستم شامل یک پخش‌کننده‌ی زمان بیکاری است که با استفاده از یک کنترلر PID که تغییرات بارکاری را دنبال می‌کند، زمان اجرای واقعی وظایف را با خطایی محاسبه می‌کند. بعد از تخمین زمان اجرای مورد انتظار یک وظیفه، بخش دیگری که انتخاب-کننده‌ی ولتاژ نام دارد، سطح ولتاژ مناسب برای اجرای آن کار را مشخص می‌کند. ازآنجایی‌که روش ارائه شده در این مقاله به‌صورت برخط است، نمی‌توان از محاسبات سنگین بهینه‌سازی برای اختصاص دادن زمان بیکاری موجود به وظایف استفاده کرد. بنابراین از دو روش مکاشفه‌ای برای تخصیص زمان بیکاری استفاده شده است. [۴۷] یک روش مدیریت انرژی آگاه به قابلیت اطمینان برای اجرای وظایف وابسته روی یک سیستم چندپردازنده‌ای با حافظه‌ی مشترک را ارائه داده است. این روش مشابه روشی است که در [۴۸] برای وظایف مستقل ارائه شده بود. ازآنجایی‌که در بسیاری از کاربردها، وظایف بی‌درنگ دارای وابستگی هستند، این وابستگی‌ها ارتباط ورودی-خروجی بین وظایف را نشان می‌دهد. در این مقاله نشان داده شده است که مدیریت انرژی آگاه از قابلیت اطمینان برای وظایف وابسته در سیستم‌های چندپردازنده‌ای یک مسئله‌ی NP-hard است. با در نظر گرفتن این‌که وظایفِ بازیابی چگونه زمان‌بندی می‌شوند، دو روش مکاشفه‌ای ارائه کرده‌اند که عبارت‌اند از: ۱) مدیریت انرژی به همراه بازیابی مجزا برای وظایف، ۲) مدیریت انرژی به همراه بازیابی اشتراکی برای وظایف.

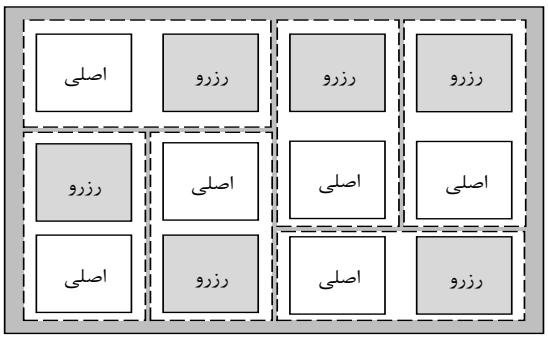
حوزه‌ی سومی که به این پروژه مرتبط است، مدیریت دمای سیستم‌های چندهسته‌ای است. [۲۰] با در نظر گرفتن محدودیت‌های بی‌درنگی و با استفاده از به تعویق انداختن بهینه‌ی هر وظیفه، انرژی مصرفی و به‌تبع آن دمای تراشه را با کنترل کردن انرژی ناشی از جریان‌های نشتی کاهش می‌دهد. در [۱۰] یک زمان‌بند آگاه از دما برای وظایف بی‌درنگ از نوع گاه‌وبیگاه[۲۸]، به جهت کاهش دمای تراشه در سیستم‌های چندهسته‌ای همگن ارائه شده است. در این زمان‌بند ممکن است دمای تعدادی از هسته‌ها به برخی دیگر از هسته‌ها منتقل شود. الگوریتم ارائه شده در این پژوهش باعث می‌شود تا دمای تراشه به میزان ۳۰ تا ۷۰ درجه‌ی سانتی‌گراد نسبت به الگوریتم‌هایی که از روش‌های متعادل کردن بارکاری[۲۹] استفاده می‌کنند، کاهش یابد.

در [۱۶]، یک الگوریتم زمان‌بندی و تخصیص وظیفه‌ی آگاه از دمای تراشه برای سیستم‌های بی‌درنگ ارائه شده است. زمان‌بندِ این پژوهش هم آگاه از توان مصرفی و هم آگاه از دمای سیستم بی‌درنگ است. این زمان‌بند ابتدا به بسترهایی که آگاه از توان و سپس به بسترهایی که آگاه از دما هستند، اعمال می‌شود. سپس بسته به کاربرد کاربر از بستر مناسب استفاده می‌شود. پژوهش [۳] یک روش زمان‌بندی و تخصیص وظایف تحتِ محدودیت‌های زمانی بی‌درنگ سخت را ارائه می‌دهد که دمای تراشه را تا حد قابل قبولی (به‌طور متوسط ۱۰ درجه‌ی سانتی‌گراد و حداکثر حدود ۳۰ درجه‌ی سانتی‌گراد، در مقایسه با روش‌های مبتنی بر کاهش متوسط توان مصرفی و به‌طور متوسط ۹ درجه‌ی سانتی‌گراد و حداکثر حدود ۲۳ درجه‌ی سانتی-گراد، در مقایسه با روش‌های مبتنی بر کاهش اوج توان) کاهش می‌دهد. همچنین [۳۱] یک سیستم بی‌درنگ محکم[۳۰] با تک پردازنده (با دو حالت کاری: ۱. حالت بیکار، ۲. حالت اجرای وظیفه) را در نظر گرفته و یک روش ارزیابی تحلیلی با استفاده از مدل مارکوف را ارائه می‌کند. در این مدل کارایی، قابلیت اطمینان و انرژی مصرفی سیستم به‌صورت تحلیلی با تمرکز بر روی رفتار دمای پردازنده ارزیابی می‌شود. در پژوهش [۴۲] یک زمان‌بند آگاه از دما در زمان اجرا که از مهاجرت وظایف بین هسته‌ها و تکنیک قطع منبع تغذیه[۳۱] استفاده می‌کند، ارائه کرده است. زمان‌بندِ این پژوهش وظایفِ بی‌درنگِ متناوب را به‌گونه‌ای به هسته‌ها تخصیص می‌دهد که میزان بهره‌وری تمام هسته‌ها متعادل گردد.

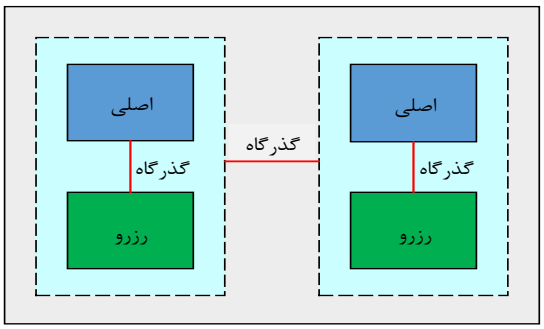
۳- ارائه‌ی روش پیشنهادی

روش پیشنهادی را با بیان یک مثال انگیزشی شرح می‌دهیم. فرض کنید یک تراشه‌ی چندهسته‌ای با چهار هسته‌ی همگن (یعنی دارای دو زیرسیستم رزرو-آماده‌باش) مطابق شکل ۱ قسمت ب وجود دارد که مقدار اوج توان مصرفی روی تمام هسته‌ها به ازای اجرای انواع وظایف بر روی هریک از آن‌ها، یکسان و برابر با ۲۵ وات باشد. به‌عبارت‌دیگر وظایف و هسته‌ها همگن در نظر گرفته شده است. برای این تراشه مقدار TDP برابر با ۷۵ وات در نظر گرفته شده است. همچنین هنگامی‌که هسته‌ها در حالت خواب[۳۲] قرار می‌گیرند، مقدار توانی که مصرف می‌کنند ناچیز و برابر صفر وات در نظر گرفته شده است. این بدین معنی است که از تکنیک مدیریت پویای توان[۳۳] استفاده شده است [۱۴] [۵۲] [۵۰] [۳۸] [۱]. در نظر بگیرید که دو وظیفه‌ی بی‌درنگ τ_1 و τ_2 با زمان ورود ۰ و ضرب‌الاجل ۱ در این سیستم وجود دارد؛ یعنی وظایفِ موجود، مبتنی بر قاب هستند. به‌منظور تحمل‌پذیری اشکال، هر وظیفه یک پشتیبان دارد که روی هسته‌های پشتیبان در هر زیرسیستمِ رزرو-آماده-باش اجرا می‌شود. تمامی وظایف اعم از وظایف اصلی و وظایف پشتیبان نیاز به ۰/۷۵ واحد زمانی برای اجرا دارند. ما زمان‌بندی افراز شده را در نظر می‌گیریم که هر وظیفه به یک زیرسیستمِ مشخص تخصیص داده می‌شود و مهاجرت وظایف بین زیرسیستم‌ها امکان‌پذیر نیست؛ در این سیستم زمانی که وظیفه‌ای به یک زیرسیستم تخصیص داده می‌شود، وظیفه‌ی اصلی و پشتیبان به ترتیب روی هسته‌ی اصلی و رزرو اجرا می‌شوند [۲۲]. در این سیستم، τ_1 و τ_2 به ترتیب به زیرسیستم ۱ و ۲

تخصیص داده شده است. بر این اساس حاصل‌جمع بارکاری تمام وظایف در این سیستم برابر با ۳ می‌باشد.



(الف)



(ب)

شکل ۱- الف) مثالی از تراشه‌ی چندهسته‌ای، ب) تراشه‌ای با چهار هسته یا دو زیرسیستم رزرو-آماده‌باش

زمان‌بندی برای این سیستم در شکل ۲ نشان داده شده است. تمامی حالات موجود در این شکل، قیود زمان‌بندی را رعایت کرده‌اند. در شکل ۲ (الف) تمام وظایف به‌طور هم‌زمان از زمان ۰ تا ۰/۷۵ اجرا می‌شوند و پس از زمان ۰/۷۵ تمامی هسته‌ها به حالت خواب می‌روند. در این مورد اوج توان مصرفی در بازه‌ی ۰ تا ۰/۷۵ برابر با ۱۰۰ وات و در بازه‌ی ۰/۷۵ تا ۱ تقریباً برابر با صفر وات می‌باشد؛ بنابراین محدودیت TDP در این مورد نقض شده است. در شکل ۲ (ب) زمان‌بندی دیگری مشاهده می‌شود که از سیاست‌های زمان‌بندی "نزدیک‌ترین ضرب‌الاجل نخست[۳۴]" و " نزدیک‌ترین ضرب‌الاجل دیرتر[۳۵]" به ترتیب بر روی هسته‌های اصلی و رزرو استفاده شده است [۱۳]. در این مورد محدودیت TDP در اکثر زمان‌ها نقض شده است. از طرف دیگر، در شکل ۲ قسمت (پ) تمامی محدودیت‌های زمانی و محدودیت TDP رعایت شده است. در این مورد اوج توان مصرفی در تمامی زمان‌ها برابر عناصر ارتباطی با ۷۵ وات است؛ زیرا که در تمام زمان‌ها فقط سه هسته فعال هستند. این مثال نشان می‌دهد که کاهش اوج توان مصرفی یک مسئله‌ی مهم است؛ زیرا زمانی که مقدار اوج توان مصرفی تجمعیِ تمام هسته‌ها از محدودیت TDP تجاوز کند، سیستم به حالت ناپایدار می‌رود.

بنابراین هدف این پژوهش کاهش اوج توان مصرفی با در نظر گرفتن روش‌های تحمل‌پذیری اشکال در سیستم‌های بی‌درنگ سخت است. برای این منظور باید تمامی محدودیت‌های زمانی و محدودیت TDP برای سیستم‌های چندهسته‌ای رعایت شود. این محدودیت‌ها عبارت‌اند از: ۱) اوج توان مصرفی نباید از حدِ TDP تعریف شده برای تراشه‌ی چندهسته‌ای تجاوز کند. ۲) تمامی محدودیت‌های زمانی باید رعایت شوند. ۳) قابلیت اطمینان سیستم باید در سطح قابل قبولی حفظ گردد. بنابراین ایده‌ی اصلی این پژوهش مدیریت اوج توان مصرفی در سیستم‌های بی‌درنگِ سختی است که از روش‌های تحمل‌پذیری اشکال به جهت افزایش قابلیت اطمینان استفاده می‌کنند. بدین ترتیب، کارایی روش پیشنهادی را در مقایسه با روش ارائه شده در [۱۳] مقایسه می‌نماییم. در ادامه مدل سیستم، الگوریتم ارائه شده در دو مرحله‌ی برون‌خط و برخط شرح داده می‌شود و درنهایت نتایج شبیه‌سازی‌های انجام شده را نشان خواهیم داد.

۳-۱- مدل سیستم

یک مجموعه از وظایف متناوب با n وظیفه $\Psi = \{\tau_1, \tau_2, \tau_3, ..., \tau_n\}$ را در نظر می‌گیریم که زمان آزادسازی و ضرب‌الاجل یکسانی دارند؛ به‌عبارت‌دیگر وظایفی مبتنی بر قاب هستند. هر وظیفه‌ی τ_i دارای بدترین زمان اجرای[۳۶] WC_i تحت ولتاژ و فرکانس حداکثر است. تمام وظایف نیز دارای دوره‌ی تناوب T_i هستند. پارامتری تحت عنوان فاصله‌ی زمانی پایه[۳۷] یا BTI مبتنی بر بزرگ‌ترین مقسوم‌علیه مشترک[۳۸] بدترین زمان اجرای تمام وظایف تعریف می‌شود. بر اساس این مقدار، تمام وظایف را به یکسری زیروظیفه[۳۹] تقسیم می‌کنیم که هر زیروظیفه را به‌صورت τ_{ij} نشان می‌دهیم. τ_{ij} به معنی زیروظیفه‌ی jام از وظیفه‌ی iام می‌باشد. همچنین میزان بهره‌وری هر وظیفه برابر با WC_i/T و میزان بهره‌وری کل سیستم برابر با حاصل‌جمع تک‌تک بهره‌وری وظایف می‌باشد.

همچنین ما یک تراشه‌ی چندهسته‌ای که شامل m هسته‌ی همگن $C = \{C_j\}_{j=1}^{m}$ و از $k=m/2$ زیرسیستمِ رزرو-آماده‌باش $SS = \{ss_l\}_{l=1}^{k}$ تشکیل شده است را در نظر می‌گیریم. در این سیستم، هر زیرسیستمِ رزرو-آماده‌باش دارای یک هسته‌ی اصلی است که تمام وظایف اصلیِ تخصیص داده شده به این زیرسیستم را اجرا می‌کند؛ به‌علاوه دارای هسته‌ای رزرو است که وظایف پشتیبان را اجرا می‌کند. در این مدل هر وظیفه‌ی τ_i که به یک زیرسیستم تخصیص داده شده است فقط بر روی هسته‌ی اصلی زیرسیستمِ تخصیص داده شده اجرا می‌شود و Ψ_l یک مجموعه از وظایف تخصیص داده شده به ss_l است. Ψ یک مجموعه از تمام وظایف است که بر روی تراشه‌ی چندهسته‌ای (یعنی $\Psi = \bigcup_{l=1}^{k} \Psi_l$) اجرا می‌شود. تمامی وظایف اعم از اصلی و پشتیبان در فرکانس و ولتاژ حداکثر اجرا می‌شوند؛ به معنای دیگر، از تکنیک تغییر پویای ولتاژ و فرکانس استفاده نمی‌شود. استفاده نکردن از تکنیک تغییر پویای ولتاژ و فرکانس به این دلیل است که اگر تعداد هسته‌های موجود بر روی تراشه زیاد باشد و به ازای هر هسته یک کنترلر برای تنظیم این تکنیک به کار برده شود، سربار هزینه، مساحت و توان مصرفی زیادی را به سیستم تحمیل می‌کند [۲۶] [۴۵]. همچنین، توان مصرفی کلِ سیستم شامل توان ایستا[۴۰] و پویا[۴۱] است. قسمت غالب توان ایستا P_s، توان ناشی از جریان نشتی[۴۲] است. توان پویا نیز شامل یک جزء توان مصرفی وابسته به فرکانس[۴۳] P_{dep} و یک جزء توان مصرفی غیر وابسته به فرکانس[۴۴] P_{ind} است که به‌وسیله‌ی ماژول‌های سیستم همچون زیرسیستم‌های حافظه و I/O فعال، مصرف می‌شوند.

$$P = P_s + P_d \quad (۱)$$

$$P_d = P_{ind} + P_{dep} \quad (۲)$$

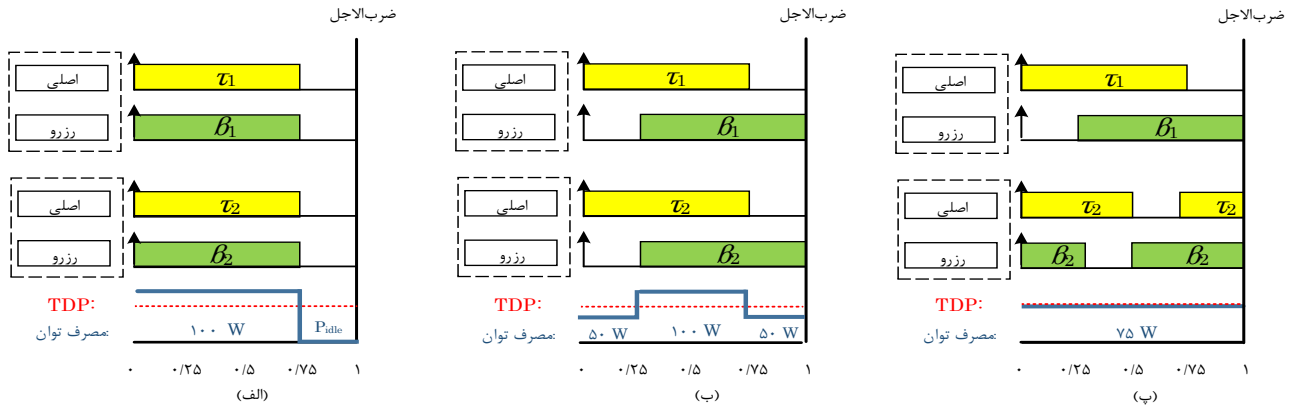
جزء وابسته به فرکانس از رابطه‌ی (۳) به دست می‌آید:

$$P_{dep} = C_{eff} . V_{dd}^2 . f \quad (۳)$$

در این رابطه C_{eff} خازن کلیدزنی مؤثر سیستم است. ولتاژ تغذیه‌ی پردازنده V_{dd} یک رابطه‌ی خطی با فرکانس f دارد. بنابراین رابطه‌ی (۳) می‌تواند به‌صورت رابطه‌ی (۴) بازنویسی شود:

$$P_{dep} = C_{eff} . V_{dd}^3 \quad (۴)$$

بنابراین روش پیشنهادی بر روی توان کل تمرکز کرده است. تحقیقات موجود نشان می‌دهد که به خاطر وجود توان پویای وابسته به فرکانس P_{dep}، کاهش سرعت



شکل ۲- مثال انگیزشی با سه مدل زمان‌بندی

یک وظیفه به‌صورت دلخواه، همیشه کارآمدی توان را در بر ندارد [۹] [۴۹] [۵۰] [۵۱] [۵۲] [۵۳].

همانند اکثر مطالعات صورت گرفته روی زمان‌بندی سیستم‌های بی‌درنگ، ما نیز بر روی مجموعه وظایفی که همه‌ی وظایف مستقل از یکدیگر هستند، تمرکز می‌نماییم. این مدلِ وظیفه آن‌چنان‌که به نظر می‌رسد، محدودکننده نیست؛ زیرا روش‌هایی برای انتقال یک مجموعه از وظایف وابسته به وظایف مستقل وجود دارد [۲۲]. علاوه بر این هر وظیفه مقدار متفاوتی از توان الکتریکی در هنگام اجرای خود مصرف می‌کند که وابسته به بارِ محاسباتی و ویژگی‌های آن وظیفه دارد. برای کاهش اوجِ توانِ مصرفی در زمان طراحی، به یک حد بالا از اتلاف توان لحظه‌ای هر وظیفه نیاز داریم. برای این منظور، از یک پارامتر مرتبط با هر وظیفه‌ی τ_i استفاده می‌کنیم که تحت عنوان بدترین توان مصرفی لحظه‌ای[۴۵] PP_i معرفی می‌شود. به‌عبارت‌دیگر، هر زیروظیفه از وظیفه‌ی τ_i، اوج توان لحظه‌ای بیش از PP_{ij} در هنگام اجرای خود مصرف نمی‌کند. همچنین PP_{max} اوجِ توانِ لحظه‌ایِ مصرف‌شده توسط وظایف در مجموعه‌ی Ψ است. زمانی که مجموعه‌ی وظایف و مدل تراشه در زمان طراحی داده می‌شود، اوج توانِ هر زیروظیفه (PP_{ij}) از هر وظیفه می‌تواند به‌صورت برون‌خط از خواندن سنسورِ توانِ نهفته شده در تراشه به دست آید (هر وظیفه به‌طور پیوسته برای یک مجموعه‌ی مشخصی از ورودی‌های آن وظیفه اجرا شده است که شبیه به اندازه‌گیری توان در [۳۰] می‌باشد). بدین منظور، ورودی هر وظیفه باید شناخته شده باشد، زیرا مقدار اتلاف توان می‌تواند بسته به مسیر اجرای تعیین‌شده توسط ورودی‌ها، متفاوت باشد. اگرچه داده‌ی ورودی وظیفه می‌تواند به‌صورت تجربی جمع-آوری شود؛ همان‌طور که برای تخمین بدترین زمان اجرای وظایف انجام گرفته است (به‌طور مثال آمار سرعت وسایل نقلیه می‌تواند برای تعیین مجموعه‌ای از ورودی‌ها برای وظیفه‌ی کنترل ABS در یک سیستم کنترل خودرو استفاده شود).

در این سیستم، زمان‌بندی افراز شده در نظر گرفته شده است، به‌طوری‌که هر وظیفه به یک زیرسیستم تخصیص داده می‌شود و مهاجرت وظایف بین زیرسیستم‌ها اتفاق نمی‌افتد. از طرف دیگر فرض شده است که وظایف مستقل از هم هستند. بنابراین هیچ وابستگی داده‌ای یا اشتراک منابع بین آن‌ها رخ نمی‌دهد. درهرصورت ما بر روی اوج توان مصرفی هر زیروظیفه‌ی τ_{ij} که با PP_{ij} نشان داده می‌شود تمرکز می‌کنیم. همچنین برای کاهش توان مصرفی در هنگام بیکاری هسته‌ها از تکنیک مدیریت پویای توان برای تمامی هسته‌ها استفاده می‌نماییم. برای این منظور یک حد زمان بیکاریِ هسته تحت عنوان $\Delta_{critical}$ در نظر گرفته می‌شود. هنگامی‌که زمان بیکاری هسته از این مقدار بزرگ‌تر باشد، می‌توان هسته را به حالت خواب بُرد تا توانِ مصرفی بسیار ناچیزی داشته باشد. مدل اشکال[۴۶] نیز همانند پژوهش‌های پیشین [۱۳] [۱۴] [۳۹] در نظر گرفته شده است. نرخ متوسطِ اشکالِ سیستم به فرکانس هسته وابسته است و طبق فرمول (۵) به دست می‌آید. در این فرمول 10^{-7} = λ_0 نرخ اشکال در فرکانس حداکثر و d حساسیت سیستم به تغییر ولتاژ است.

چون در سیستم پیشنهادی از تکنیک تغییر پویای ولتاژ و فرکانس استفاده نمی‌شود f=۱ است؛ بنابراین نرخ متوسط اشکال در این سیستم برابر با $\lambda(f)=\lambda_0$ خواهد شد.

$$\lambda(f)=\lambda_0\times 10^{\frac{d(1-f)}{1-f_{\min}}} \quad (۵)$$

بنابراین قابلیت اطمینان هر وظیفه طبق پژوهش‌های [۷] [۱۱] مطابق با فرمول (۶) است که در آن t_i زمان واقعی اجرای وظیفه‌ی τ_i است. بنابراین قابلیت اطمینان یک زیرسیستم و کل سیستم، به ترتیب از فرمول (۷) و (۸) به دست می‌آید [۷] [۱۱].

$$R_i(\tau_i)=e^{\lambda(f).t_i} \quad (۶)$$

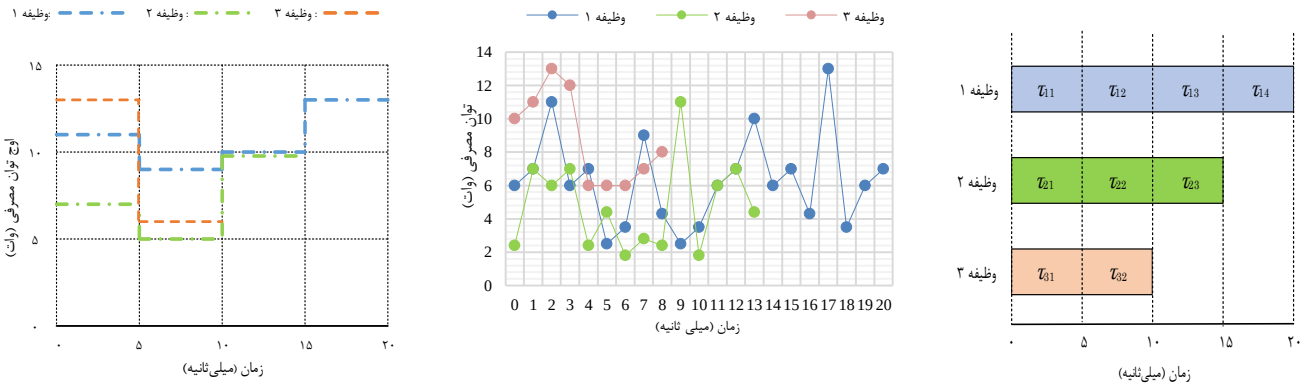
$$R_{Sub-sys}(k)=\prod_{i=1}^{i=n}(1-(1-R(\tau_i))^2) \quad (۷)$$

$$R_{tot-sys}=\prod_{l=1}^{l=k}(R_{sub-sys}(k)) \quad (۸)$$

در زمان‌های اتمام وظایف (هم بر روی هسته‌های اصلی و هم بر روی هسته‌های رزرو)، یک تست پذیرش[۴۷] (یا سازگار پذیری[۴۸]) برای تعیین رخداد اشکالات گذرا انجام می‌شود [۳۶]. اگر یک اشکال کشف شود، وظیفه‌ی پشتیبان (یا اصلی) می‌تواند در هسته‌ی رزرو (اصلی) اجرا گردد. در غیر این صورت از اجرای وظیفه‌ی پشتیبان (یا اصلی) صرف‌نظر می‌شود. همچنین فرض شده است که هزینه‌ی اجرای تست پذیرش در بدترین حالت، شامل زمان اجرای کارها باشد [۴۴].

۳-۲- راه‌حل پیشنهادی

برای دست‌یابی به هدف این پژوهش، یک روش مدیریت اوج توان مصرفی در سطح سیستم ارائه می‌دهیم. درواقع یک الگوریتم زمان‌بندی آگاه از اوج توان مصرفی در سیستم‌های چندهسته‌ای بی‌درنگ سخت که از زیرسیستم‌های رزرو-آماده‌باش عبارت‌اند از: ۱) در مرحله‌ی اول، ما بدترین نوع اوج توان مصرفی هر وظیفه را با استفاده از ابزار ردیابی توان به دست می‌آوریم. ۲) در مرحله‌ی دوم تمام وظایف به زیرسیستم‌های مربوط به خود تخصیص داده می‌شوند. ۳) در مرحله‌ی سوم نیز جدولی تحت عنوان جدول همپوشانی[۴۹] که تمام جایگشت‌های بین زیروظایف در یک زیرسیستم به‌طوری‌که حاصل‌جمع اوج توان مصرفی بین دو وظیفه، کمتر از تشکیل شده است، ارائه می‌کنیم. راه‌حل این پیشنهاد شامل چهار مرحله است، که مقدار TDP تخصیص داده شده به هر زیرسیستم باشد، تولید می‌شود تا از روی آن در مرحله‌ی چهارم زمان‌بندی صورت بگیرد. ۴) یک الگوریتم آگاه از اوج توان مصرفی ارائه می‌شود که البته تمام قیود زمانی را نیز رعایت می‌کند.



(الف) (ب) (پ)

شکل ۳- الف) افراز وظایف به تعدادی زیروظیفه، ب) نمودار توان مصرفی وظایف، پ) نگاشت نمودار توان مصرفی به بدترین حالت اوج توان مصرفی

۳-۲-۱- مرحله‌ی اول: تنظیمات اولیه

در این قسمت، اوج توان مصرفی هر زیروظیفه را مبتنی بر مراحل زیر به دست می‌آوریم. ابتدا نمودار توان مصرفی هر وظیفه را در نظر می‌گیریم. سپس، پارامتر فاصله‌ی زمانی پایه را بر اساس بزرگ‌ترین مقسوم‌علیه مشترکِ بدترین زمانِ اجرای وظایف محاسبه می‌کنیم. بر مبنای این پارامتر، تمام وظایف را به چند زیروظیفه تقسیم می‌کنیم تا به‌طور دقیق‌تری اوج توان مصرفی را به دست آوریم. بر این اساس هر زیروظیفه یک اوج توان مصرفی خاصی دارد که از نمودار توان مصرفی وظیفه حاصل شده است. درنهایت تمامی این اطلاعات را برای استفاده در مراحل بعدی مستندسازی می‌نماییم. در پایان این مرحله نیز یک روش تحمل‌پذیری اشکال همانند تکنیک رزرو-آماده‌باش انتخاب می‌شود. تکنیک رزرو-آماده‌باش یکی از متداول‌ترین روش‌های تحمل‌پذیری اشکال است که برای افزایش قابلیت اطمینان سیستم به کار برده می‌شود. در این تکنیک از دو ماژول (هسته) جهت کشف اشکال، مکان‌یابی اشکال و بازگشت از اشکال برای دست‌یابی به تحمل‌پذیری اشکال استفاده می‌شود [۲۱]. روش افزونگی آماده‌باش خود به دو گونه است: ۱) در این حالت هسته‌ی اصلی فعال است و وظایف اصلی را اجرا می‌کند. هسته‌ی رزرو نیز همیشه فعال است و پشتیبان وظایف اصلی را اجرا می‌نماید. به این‌گونه از افزونگی آماده‌باش، آماده‌باش گرم[۵۰] گفته می‌شود. ۲) در این حالت هسته‌ی اصلی همیشه فعال است و وظایف اصلی را اجرا می‌کند؛ ولی هسته‌ی رزرو غیرفعال است و زمانی فعال می‌شود که یک اشکالی در وظیفه‌ی فعلی که روی هسته‌ی اصلی اجرا می‌شود، تشخیص داده شود و آن وظیفه‌ی مورد نظر را دوباره اجرا می‌نماید. به این‌گونه از افزونگی آماده‌باش، آماده‌باش سرد[۵۱] گفته می‌شود. در این پژوهش از افزونگی آماده‌باش گرم استفاده شده است.

۳-۲-۲- مرحله‌ی دوم: تخصیص وظایف به زیرسیستم‌ها

در این مرحله، زمان‌بندی افراز شده را برای نگاشت وظایف به زیرسیستم‌ها در نظر می‌گیریم. یافتن زمان‌بندی افراز شده‌ای که تمامی قیود زمانی را رعایت کند یک مسئله‌ی اساسی در این پژوهش و سایر پژوهش‌ها است. در این زمینه، [۵] مطالعه‌ی فراگیری ارائه کرده است. درواقع پیدا کردن تخصیص‌دهنده‌ی امکان‌پذیر برای مجموعه‌ای از وظایف با محدودیت‌های زمانی بی‌درنگ به تعدادی هسته، مسئله‌ای NP-hard است. بنابراین طبق این مسئله، تخصیص مجموعه‌ای از وظایف با قیود زمانی بی‌درنگ به تعدادی زیرسیستم رزرو-آماده‌باش مسئله‌ای NP-hard است. خروجی الگوریتم افراز کننده‌ی وظایف، ماتریس افرازی است که با $V = [v_{il}]_{n*k}$ نشان داده می‌شود. در این ماتریس عنصر v_{il} زمانی مقدار ۱ به خود می‌گیرد که τ_i به زیرسیستم SS_j تخصیص داده شده باشد؛ در غیر این صورت مقدار v_{il} صفر خواهد شد. درهرصورت الگوریتم تخصیص وظایف به زیرسیستم‌ها باید تضمین نماید که بهره‌وری هر زیرسیستمِ رزرو-آماده‌باش کمتر یا مساوی ۱۰۰ درصد باشد تا وظایف بتوانند ضرب‌الاجل خود را رعایت کنند. اگر پس از تخصیص وظایف، زیرسیستمی بیش از ۱۰۰ درصد بهره‌وری داشت، یا باید وظایف خود را به سایر زیرسیستم‌ها برای اجرا بسپارد یا این‌که به‌طورکلی تراشه‌ی چندهسته‌ای باید تغییر کند تا تعداد زیرسیستم‌های رزرو-آماده‌باش بیشتری داشته باشد. همچنین ممکن است با تغییر الگوریتمِ تخصیصِ وظایف، بتوان تمام وظایف را به‌گونه‌ای به زیرسیستم‌ها سپرد که همه‌ی آن‌ها میزان بهره‌وری کمتر از ۱۰۰ درصد داشته باشند.

۳-۲-۳- مرحله‌ی سوم: تولید جدول همپوشانی

در این قسمت، با در نظر گرفتن محدودیت TDP، الگوریتم ارائه شده، جدولی تحت عنوان جدول همپوشانی تولید می‌کند. این جدول برای هر زیرسیستمِ رزرو-آماده‌باش تولید می‌شود و آن زیرسیستم با توجه به این جدول زمان‌بندی می‌شود. این جدول حاوی سطرهایی است که جایگشت‌های مختلف تمام زیروظایف با این شرط که از مقدار TDP تخصیص یافته به آن زیرسیستم تجاوز نکند را در خود جای داده است. برای مثال فرض کنید مقدار TDP تخصیص داده شده به یک زیرسیستمی ۲۰ وات باشد، در جدول همپوشانی مربوط به این زیرسیستم، جایگشت‌های تمام زیروظایف به‌طوری‌که اوج توان مصرفی حاصل‌جمع آن‌ها (حاصل‌جمع اوج توان مصرفی بر روی هسته‌ی اصلی و هسته‌ی رزرو) از ۲۰ وات تجاوز نکند، قرار گرفته است. مثالی از جدول همپوشانی را در جدول ۱ مشاهده می‌نمایید.

۳-۲-۴- مرحله‌ی چهارم: تصمیم‌گیری زمان‌بندی

بعد از مرحله‌ی سوم، نوبت به زمان‌بندی تمام هسته‌ها می‌رسد. این زمان‌بندی باید سه شرط را رعایت کند: ۱) تمام قیود بی‌درنگی سخت رعایت شود. ۲) محدودیت TDP مربوط به تراشه نقض نشود. ۳) قابلیت اطمینان سیستم از حد مورد قبول خود تنزل پیدا ننماید. الگوریتم زمان‌بندی ارائه شده درصدد است تا هر سه شرط مذکور را در نظر بگیرد. برای این منظور دو سیاست زمان‌بندی ارائه نمودیم؛ اولین سیاست "حداکثر اوج توان نخست[۵۲]" است که بر روی هسته‌های اصلی و دومین سیاست "حداکثر اوج توان آخر[۵۳]" بر روی هسته‌های رزرو استفاده می‌شود. منظور از "حداکثر اوج توان نخست" این است که هر وظیفه‌ای که اوج توان بیشتری دارد زودتر از سایر وظایف زمان‌بندی می‌شود. همچنین "حداکثر اوج توان آخر" بدین مفهوم است که هر وظیفه‌ای که اوج توان بیشتری دارد دیرتر زمان‌بندی می‌شود. این دو سیاست در هنگام زمان‌بندی در یک زیرسیستم سبب می‌شوند که تداخل وظایف اصلی و پشتیبان به کمترین حد خود برسد و اوج توان مصرفی کمتری را به سیستم تحمیل کنند. درنهایت باید یک تست قابلیت زمان‌بندی فراهم شود تا سه شرط مذکور را بررسی کرده و در صورت صحت عملکرد الگوریتم، زمان‌بندی

ارائه شده را تأیید نماید. به‌عبارت‌دیگر یکی از خروجی‌های این الگوریتمِ زمان‌بندی، نتیجه‌ی بررسی سه شرط بیان‌شده در ابتدای این قسمت است.

جدول ۱- جدول همپوشانی

ردیف	(وظیفه‌ی رزرو، وظیفه‌ی اصلی)	اوج توان مصرفی (وات)
۱	(τ_{11}, β_{12}) , (τ_{12}, β_{11})	۲۰
۲	(τ_{11}, β_{21}) , (τ_{21}, β_{11})	۱۸
۳	(τ_{11}, β_{22}) , (τ_{22}, β_{11})	۱۶
۴	(τ_{11}, β_{32}) , (τ_{32}, β_{11})	۱۹
۵	(τ_{12}, β_{12})	۱۸
۶	(τ_{12}, β_{13}) , (τ_{13}, β_{12})	۱۹
۷	(τ_{12}, β_{21}) , (τ_{21}, β_{12})	۱۶
۸	(τ_{12}, β_{22}) , (τ_{22}, β_{12})	۱۴
۹	(τ_{12}, β_{23}) , (τ_{23}, β_{12})	۱۹
۱۰	(τ_{12}, β_{32}) , (τ_{32}, β_{12})	۱۷
۱۱	(τ_{13}, β_{13})	۲۰
۱۲	(τ_{13}, β_{21}) , (τ_{21}, β_{13})	۱۷
۱۳	(τ_{13}, β_{22}) , (τ_{22}, β_{13})	۱۵
۱۴	(τ_{13}, β_{23}) , (τ_{23}, β_{13})	۲۰
۱۵	(τ_{13}, β_{32}) , (τ_{32}, β_{13})	۱۸
۱۶	(τ_{14}, β_{21}) , (τ_{21}, β_{14})	۲۰
۱۷	(τ_{14}, β_{22}) , (τ_{22}, β_{14})	۱۸
۱۸	(τ_{21}, β_{21})	۱۴
۱۹	(τ_{21}, β_{22}) , (τ_{22}, β_{21})	۱۲
۲۰	(τ_{21}, β_{23}) , (τ_{23}, β_{21})	۱۷
۲۱	(τ_{21}, β_{31}) , (τ_{31}, β_{21})	۲۰
۲۲	(τ_{21}, β_{32}) , (τ_{32}, β_{21})	۱۵
۲۳	(τ_{22}, β_{22})	۱۰
۲۴	(τ_{22}, β_{23}) , (τ_{23}, β_{22})	۱۵
۲۵	(τ_{22}, β_{31}) , (τ_{31}, β_{22})	۱۸
۲۶	(τ_{22}, β_{32}) , (τ_{32}, β_{22})	۱۳
۲۷	(τ_{23}, β_{23})	۲۰
۲۸	(τ_{23}, β_{32}) , (τ_{32}, β_{23})	۱۸

۳-۳- الگوریتم زمان‌بندی

برای توضیح الگوریتمِ زمان‌بندی ارائه شده، ابتدا با یک مثال بحث را آغاز می‌نماییم. فرض کنید سه وظیفه‌ی τ_1، τ_2 و τ_3 با بدترین زمان اجرای به ترتیب ۲۰، ۱۵ و ۲۰ (میلی‌ثانیه) و ضرب‌الاجل یکسان ۱۰۰ (میلی‌ثانیه) وجود دارند. برای این مجموعه از وظایف مقدار پارامتر فاصله‌ی زمانی پایه که مبتنی بر مقسوم‌علیه مشترکِ بدترین زمانِ اجرای تمام وظایف به دست می‌آید برابر با ۵ است. همچنین فرض کنید یک تراشه‌ی دو هسته‌ای با مقدار TDP برابر با ۲۰ وات وجود دارد؛ به‌عبارت‌دیگر این تراشه دارای یک زیرسیستمِ رزرو-آماده‌باش است.

در مرحله‌ی اول، وظایف را بر اساس مقدار فاصله‌ی زمانی پایه به تعدادی زیروظایف تقسیم می‌کنیم. بر این اساس وظیفه‌های ۱، ۲ و ۳ به ترتیب به ۴، ۳ و ۲ زیروظیفه افراز می‌شوند که در شکل ۳- (الف) نشان داده شده است. سپس، نمودار توان مصرفی هر وظیفه را به دست می‌آوریم (شکل ۳- (ب)). در مرحله‌ی دوم باید وظایف را به زیرسیستم‌های رزرو-آماده‌باش تخصیص دهیم. چون در این سیستم یک زیرسیستم رزرو-آماده‌باش وجود دارد، بنابراین تمام وظایف را به این زیرسیستم تخصیص می‌دهیم. هر یک از وظایف نیز دارای یک وظیفه‌ی پشتیبان هستند که بر روی هسته‌ی رزرو از این زیرسیستم اجرا می‌شوند. در مرحله‌ی سوم نیز باید جدول همپوشانی مطابق جدول ۱ تولید گردد. در این جدول تمام جایگشت‌های مربوط به تمام زیروظایف از سه وظیفه‌ی موجود بر روی دو هسته‌ی اصلی و رزرو که محدودیت TDP را رعایت می‌کنند نشان داده شده است. در این جدول منظور از τ_{ij}، زیروظیفه‌ی jام از وظیفه‌ی اصلی iام و منظور از β_{xy}، زیروظیفه‌ی yام از وظیفه‌ی پشتیبان xام است. در این جدول هر سطر دارای دو ترکیب مختلف است (سطرهایی که یک ترکیب دارند دارای دو ترکیب بوده‌اند ولی به خاطر یکسان بودن ترکیبشان، یکی از ترکیب‌هایشان حذف شده است)؛ چون اوج توان مصرفی که یک وظیفه‌ی اصلی روی هسته‌ی اصلی مصرف می‌کند با اوج توان مصرفی که پشتیبان همان وظیفه مصرف می‌کند (یعنی $PP(\tau_{ij})=PP(\beta_{ij})$ و $PP(\tau_{xy})=PP(\beta_{xy})$)، یکسان است؛ بنابراین ترکیب (τ_{ij} ,β_{xy}) با (τ_{xy} ,β_{ij}) به یک میزان اوجِ توان مصرف می‌کنند. البته این یکسان بودن از این مفهوم ناشی شده است که هسته‌های موجود در این زیرسیستم همگن هستند.

در مرحله‌ی چهارم نیز باید الگوریتم زمان‌بندی را به سیستم اعمال نماییم. در فاز برون‌خط، بر روی هسته‌ی اصلی از سیاست "حداکثر اوج توان نخست" استفاده می‌شود؛ این سیاست بدین مفهوم است که سه وظیفه را باید ابتدا بر اساس مقدار اوج توان آن‌ها به‌صورت نزولی مرتب نماییم. سپس وظیفه‌ای که بیشترین اوج توان را دارد بر روی هسته‌ی اصلی به‌طوری‌که محدودیت TDP مربوط به زیرسیستم رعایت شود، زمان‌بندی نماییم (در این مثال به ترتیب وظایف ۳، ۱ و ۲ بر روی هسته‌ی اصلی زمان‌بندی می‌شوند). بر روی هسته‌ی رزرو نیز از سیاست "حداکثر اوج توان آخر" استفاده می‌شود؛ این سیاست ابتدا وظیفه‌ای را که بیشترین اوج توان را دارد، در نظر گرفته و از ضرب‌الاجل خود رو به عقب (در این مثال از ۶۰ میلی‌ثانیه به سمت ۰ میلی‌ثانیه حرکت می‌کند) به شرطی که محدودیت TDP زیرسیستم رعایت شود، زمان‌بندی می‌کند (در این مثال به ترتیب وظایف ۳، ۱ و ۲ بر روی هسته‌ی رزرو از ثانیه‌ی ۶۰ به سمت ۰ زمان‌بندی می‌شوند). این زمان‌بندی برای مثال مورد نظر، در شکل ۴- (الف) نشان داده شده است.

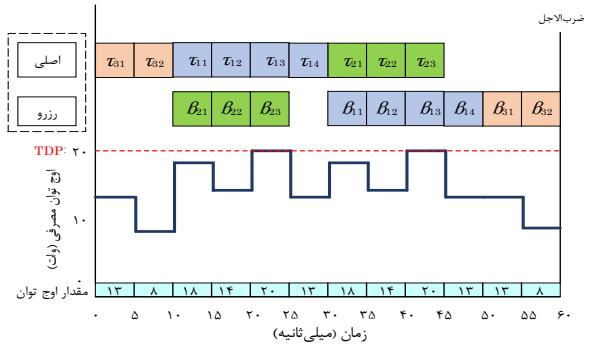
در فاز برخط نیز هنگامی‌که یک وظیفه بر روی هسته‌ی اصلی یا هسته‌ی رزرو در یک زیرسیستمِ رزرو-آماده‌باش اجرایش به پایان می‌رسد، باید از اجرای ادامه‌ی وظیفه‌ی متناظرش (اگر وظیفه‌ی اصلی به اتمام رسید از ادامه‌ی اجرای وظیفه‌ی پشتیبان و اگر وظیفه‌ی پشتیبان اجرایش تکمیل شود از ادامه‌ی اجرای وظیفه‌ی اصلی صرف‌نظر می‌شود) جلوگیری به عمل آید تا هم اوج توان مصرفی و هم انرژی سیستم را کاهش دهیم. در این مثال، هنگامی‌که یک سناریوی بدون اشکال را دنبال نماییم، متوجه می‌شویم که می‌توانیم با استفاده از الگوریتم ارائه شده، اوج توان مصرفی سیستم را به‌خوبی کاهش دهیم. در این مثال هنگامی‌که وظیفه‌ی اصلی τ_3 در زمان ۱۰ میلی‌ثانیه بدون اشکال اجرا می‌شود از اجرای وظیفه‌ی پشتیبانِ متناظر با این وظیفه بر روی هسته‌ی رزرو در بازه‌ی زمانی ۵۰ تا ۶۰ میلی‌ثانیه جلوگیری به عمل می‌آید. پس‌ازآن وقتی وظیفه‌ی پشتیبان β_2 در زمان ۲۵ میلی‌ثانیه به‌درستی اجرا می‌شود از اجرای وظیفه‌ی متناظر با آن یعنی τ_2 در بازه‌ی زمانی ۳۰ تا ۴۵ میلی‌ثانیه بر روی هسته‌ی اصلی منصرف می‌شویم. همچنین زمانی که وظیفه‌ی اصلی τ_1 بر روی هسته‌ی اصلی در زمان ۳۰ میلی‌ثانیه به اتمام می‌رسد و هیچ اشکالی کشف نمی‌شود، از اجرای وظیفه‌ی پشتیبان β_1 بر روی هسته‌ی رزرو جلوگیری به عمل می‌آید. با این کار هم اوج توان مصرفی را به‌خوبی در بازه‌ی زمانی ۳۰ تا ۶۰ میلی‌ثانیه کاهش دادیم و هم این‌که انرژی مصرفی را تا حد قابل قبولی ذخیره نمودیم. پس از مثالی که در مورد الگوریتم ارائه شده بیان نمودیم، حال به شرح الگوریتم جامع با تعداد زیرسیستم‌های متعدد می‌پردازیم؛ به‌عبارت‌دیگر کارایی الگوریتم خود را در دو فاز برون‌خط و برخط برای سیستم‌های چندهسته‌ای تحمل‌پذیر اشکال با قیود زمانی بی‌درنگ سخت ارائه می‌دهیم. بنابراین برای فاز برون‌خط از الگوریتم شماره‌ی یک و برای فاز برخط از الگوریتم شماره‌ی دو استفاده می‌شود که در ادامه به توضیح هریک پرداخته می‌شود

الگوریتم ۱- (فاز برون‌خط) مدیریت اوج توان مصرفی در سیستم‌های چندهسته‌ای تحمل‌پذیر اشکال با قیود زمانی سخت

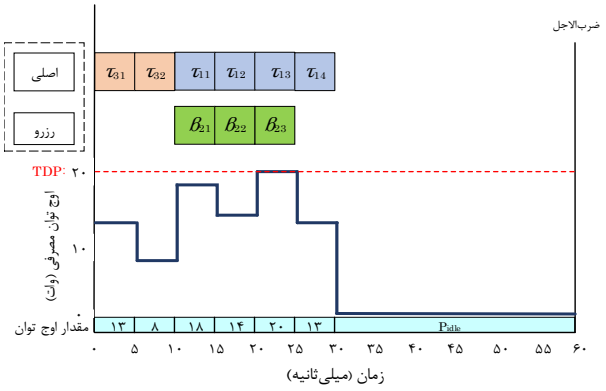
Inputs: Set of application tasks $\Psi = \{\tau_1, \tau_2, \tau_3, ..., \tau_n\}$, Set of cores $C = \{C_1, C_2, C_3, ..., C_m\}$, Set of standby-sparing sub-systems $SS = \{SS_1, SS_2, ..., SS_k\}$, Chip-level Thermal Design power (TDP) value, Peak Power Consumption for all sub-tasks, Partition Matrix: $K = [k_{i,j}]_{n\times k}$, Utilization Matrix: $U_{per\text{-}pair} = [u_i]_k$, Least common multiple of task's periods H, Greatest Common Divisor of all task's worst-cases BTI, Total number of time interval: $V = H/BTI$.

Output:
Feasibility schedule (Yes/No);
Tasks scheduling.

Algorithm:
$Internal\_TDP = TDP_{chip}/k$; //Each pair have an internal TDP value
For all $SS_k \in SS$
1. ST_i , $S\beta_i$: Sort the assigned tasks on the ss_k descending based on their amount of peak power;
2. Use **MPPF** policy on primary core:
3. BEGIN
4. while (ST_i is not empty)
5. Select and remove first T_i from sorted list ST_i;
6. Start at release time to its deadline;
7. Assign [WC_i/BTI] from global list and free time interval in primary core of pair k to task i, while internal TDP constraint should be met;
8. Update globally power consumption list;
9. END
10. Use **MPPL** policy on spare core:
11. BEGIN
12. while ($S\beta_i$ is not empty)
13. Select and remove first β_i from sorted list $S\beta_i$;
14. Start at deadline to its release time;
15. Assign [WC_i/BTI] from global list and free time interval in spare core of pair k to task i, while internal TDP constraint should be met;
16. Update globally power consumption list;
17. END
End For
End Algorithm



(الف)



(ب)

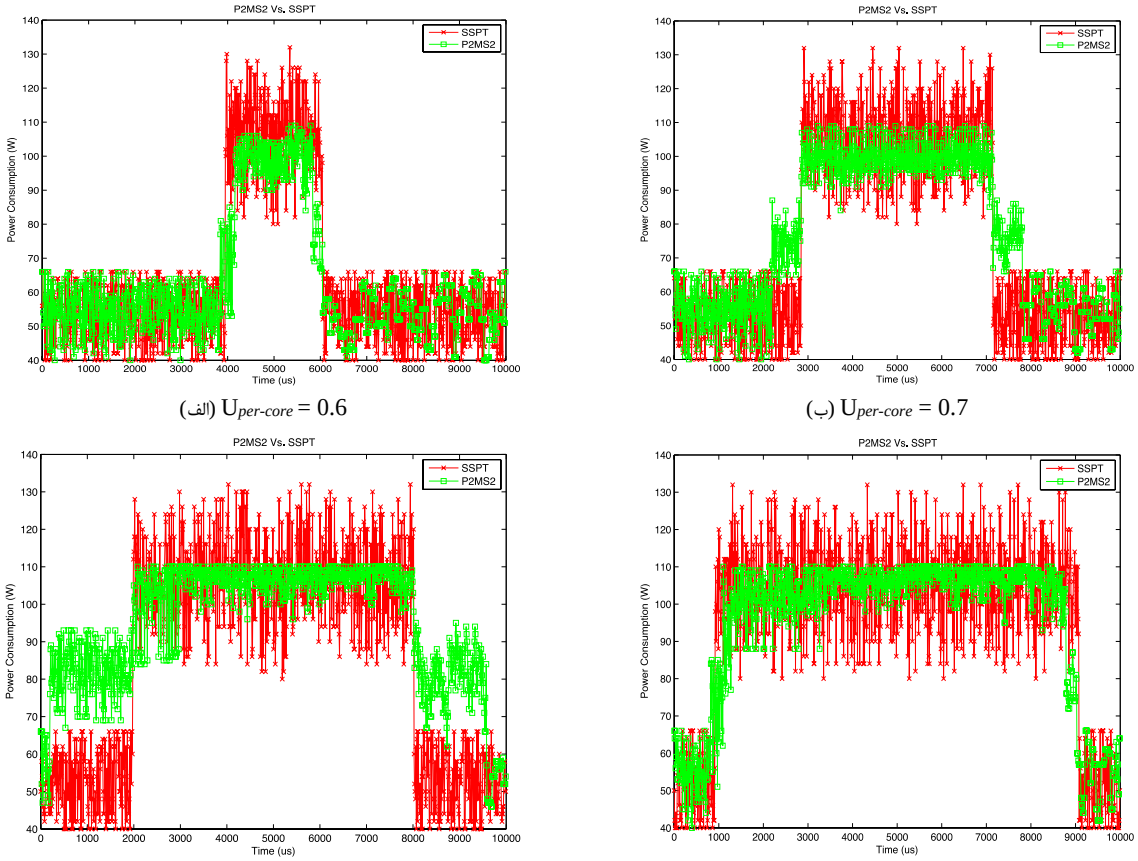
شکل ۴- الف) اجرای وظایف در فاز برون‌خط؛ ب) اجرای وظایف در فاز برخط بدون اشکال.

الگوریتم ۲- (فاز برخط) مدیریت اوج توان مصرفی در سیستم‌های چندهسته‌ای تحمل‌پذیر اشکال با قیود زمانی سخت

1. **Event –** τ_i (or β_i) completes at time t:
2. Run the acceptance test
3. **if** no error is detected **then**
4. Cancel the backup β_i (or τ_i) task on the spare core
5. **end if**
7. **Event –** Cancellation of backup task β_i (or main task τ_i):
6. **if** β_i (τ_i) is the current active task **then**
8. /*Check if the spare (or primary) can 'sleep' in the slack of β_i (or τ_i)
9. $\Delta_{ep} \leftarrow$ time_to_earliest_release_event
10. **if** $\Delta_{ep} \geq \Delta_{critical}$ **then**
11. Set sleep_exit event at t = time_now + Δ_{ep}
12. Put spare (primary) core into sleep mode
13. **end if**
14. **end if**
15. **Event –** Sleep_exit:
16. **if** the backup list (or primary list) is empty **then**
20. $\Delta_{ep} \leftarrow$ time_to_earliest_release_event
21. **if** $\Delta_{ep} \geq \Delta_{critical}$ **then**
22. Set wake-up event at t = time_now + Δ_{er}
23. Put spare (or primary) to sleep mode
24. **end if**
25. **end if**

۳-۳-۱- مرحله‌ی برون‌خط

به علت NP-hard بودن مسئله (در بخش ۳-۲-۲) از یک روش مکاشفه‌ای برای حل مسئله استفاده شده است. در فاز برون‌خط از الگوریتم ۱ استفاده می‌شود.ورودی‌های الگوریتم ۱ عبارت‌اند از: مجموعه‌ای از وظایف Ψ، مجموعه‌ای از هسته‌ها C، مجموعه‌ای از زیرسیستم‌های رزرو-آماده‌باش SS، مقدار TDP تراشه، اوج توان مصرفی تمام زیروظایف، ماتریس افراز شده (حاوی تخصیص وظایف به زیرسیستم‌ها)، ماتریس میزان بهره‌وری هر زیرسیستم، بزرگ‌ترین مقسوم‌علیه مشترک بدترین زمان اجراهای تمام وظایف BTI و تعداد فواصل زمانی پایه که برابر است با $V = H/BTI$. در ابتدای الگوریتم ۱ مقدار محدودیت TDP هر زیرسیستم را تخصیص می‌دهیم. این مقدار از حاصل تقسیم TDP کل تراشه به تعداد زیرسیستم‌ها به دست می‌آید. بنابراین محدودیت TDP هر زیرسیستم یکسان و برابر با TDP_{chip}/k است. سپس به تعداد کل زیرسیستم‌های رزرو-آماده‌باش، مراحل زیر را انجام می‌دهیم: یک زیرسیستم را انتخاب می‌کنیم. در خط شماره‌ی ۱ الگوریتم، تمام وظایف تخصیص داده شده به زیرسیستم انتخاب شده را بر اساس مقدار اوج توان مصرفی به‌صورت نزولی مرتب می‌نماییم. سپس از هسته‌ی اصلیِ آن زیرسیستم شروع می‌نماییم. در هسته‌ی اصلی از سیاست "حداکثر اوج توان مصرفی نخست" استفاده می‌شود. این سیاست بیان می‌کند که ابتدا وظیفه‌ای روی هسته‌ی اصلی زمان‌بندی می‌شود که اوج توان بیشتری نسبت به سایر وظایف دارد. این وظیفه به شرطی زمان‌بندی می‌شود که TDP مربوط به زیرسیستم خود را نقض نکند. سپس سایر وظایف به ترتیبِ مقدار اوج توان مصرفی‌شان انتخاب شده و بر روی هسته‌ی اصلی زمان‌بندی می‌شوند. این کار تا زمانی که تمام وظایف اصلی تخصیص داده‌شده به این زیرسیستم بر روی هسته‌ی اصلی زمان‌بندی شوند، ادامه پیدا می‌کند (خطوط ۲ تا ۹ الگوریتم ۱). سپس بر روی هسته‌ی رزرو از سیاست "حداکثر اوج توان مصرفی آخر" استفاده می‌شود. این سیاست بیان می‌دارد که ابتدا وظیفه‌ای که دارای اوج توان مصرفی بیشتری است از سمت ضرب‌الاجل به سمت زمان آزادسازی خود، به شرطی که محدودیت TDP زیرسیستم خود را رعایت کند، زمان‌بندی می‌شود. این کار تا زمانی که تمام وظایف پشتیبان بر روی هسته‌ی رزرو زمان‌بندی



(الف) $U_{per\text{-}core}$ = 0.6 (ب) $U_{per\text{-}core}$ = 0.7

(پ) $U_{per\text{-}core}$ = 0.8 (ث) $U_{per\text{-}core}$ = 0.9

شکل ۵- نمودار توان مصرفی برای مجموعه وظیفه‌ای در یک قاب (توزیع یکنواخت، دارای چهار هسته، دو زیرسیستم رزرو-آماده‌باش)

نشوند، ادامه پیدا می‌کند (خطوط ۱۰ تا ۱۷ الگوریتم ۱). استفاده از این دو سیاست زمان‌بندی باعث می‌شود تا به‌اندازه‌ی ممکن همپوشانی بین وظایف کاهش یابد و اوج توان مصرفی سیستم تا حد قابل قبولی کنترل گردد. درنهایت، این الگوریتم دو خروجی را تولید می‌کند که یکی از آن‌ها زمان‌بندی سیستم است و دیگری قابلیت زمان‌بندی تحت سه محدودیت زمان، TDP و قابلیت اطمینان است. اگر فاکتور قابلیت زمان‌بندی "بلی" باشد به این معناست که زمان‌بندی خروجی تمامی سه شرط مورد نظر ما را رعایت کرده است. در غیر این صورت شروط مدنظر سیستم رعایت نشده است و باید تعداد زیرسیستم‌ها یا تراشه‌ی مورد استفاده تغییر کند یا این‌که شروط ما کمی نرم‌تر شوند.

۳-۳-۲- مرحله‌ی برخط

الگوریتم ۲، فاز برخط روش ارائه شده را نشان می‌دهد. در فاز برخط، هنگامی‌که اجرای وظیفه‌ای اعم از اصلی و پشتیبان به اتمام می‌رسد، ابتدا یک تست پذیرش صورت می‌گیرد تا وجود اشکال در اجرای وظیفه را چک نماید. اگر هیچ اشکالی کشف نشد، آنگاه از اجرای وظیفه‌ی متناظر با آن وظیفه جلوگیری می‌شود. زمانی که یک وظیفه لغو می‌شود، زمان بیکاری ایجاد شده با حداقل زمانی که می‌توان از تکنیک مدیریت پویای توان استفاده کرد، مقایسه می‌شود. اگر این زمان ایجادشده بیشتر از آن حداقل زمان باشد، بر روی هسته‌ی مورد نظر تکنیک مدیریت پویای توان به کار برده می‌شود. همچنین اگر بر روی هسته‌ها فضای خالی وجود داشته باشد و این فضای زمانی از حداقل زمانِ در نظر گرفته شده بیشتر باشد، می‌توان تکنیک مدیریت پویای توان را برای کاهش اوج توان مصرفی استفاده کرد و درواقع به‌طور مناسبی اوج توان مصرفی را کاهش داد.

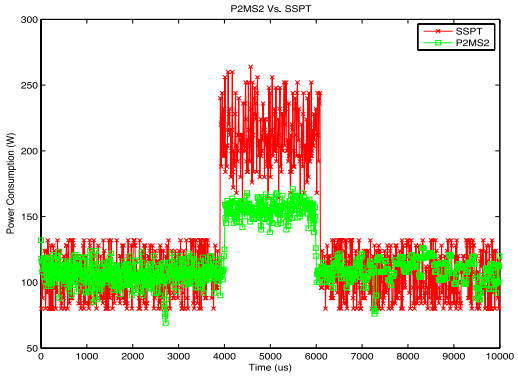
۴- نتایج شبیه‌سازی

در این بخش برای این‌که نشان دهیم روش ارائه شده به‌طور مناسبی کارا است، نتایج شبیه‌سازی را نمایش می‌دهیم. روش پیشنهادی خود را با روشی که از تکنیک رزرو-آماده‌باش به جهت تحمل‌پذیری اشکال و از سیاست‌های زمان‌بندی "نزدیک‌ترین ضرب‌الاجل نخست" و "نزدیک‌ترین ضرب‌الاجل دیرتر" به ترتیب بر روی هسته‌ی اصلی و هسته‌ی رزرو برای کاهش میانگین توان مصرفی استفاده می‌کند [۱۳]، مقایسه می‌نماییم. برای روشن شدن تفاوت میان کاهش اوج توان مصرفی و کمینه‌سازی میانگین توان مصرفی، ما الگوریتم ارائه شده را با یک روش شناخته‌شده در کمینه ساختن میانگین توان مصرفی در سیستم‌های تحمل‌پذیر اشکال مقایسه می‌کنیم که تحت عنوان SSPT شناخته می‌شود [۱۳]. در ابتدا توضیح می‌دهیم که چگونه مجموعه‌ی وظایف با مقدار اوج توان مختلف تولید شده و سپس برتری روش‌مان را نشان خواهیم داد.

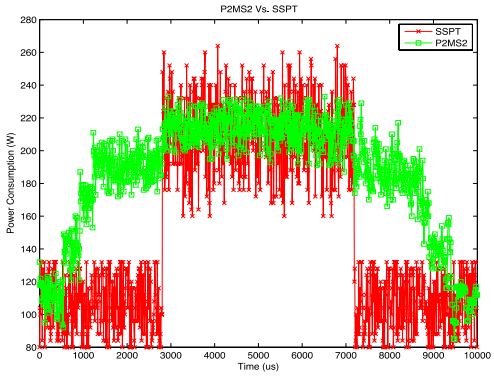
جدول ۲- بدترین اوج توان مصرفی ده وظیفه به ازای ۱۰ بار اجرا

qsort	susan	sha	jpeg	dijkstra	rijndael
۳۳/۰۹	۳۲/۴۳	۳۰/۷۷	۲۹/۱۹	۲۸/۹۹	۲۸/۸۱

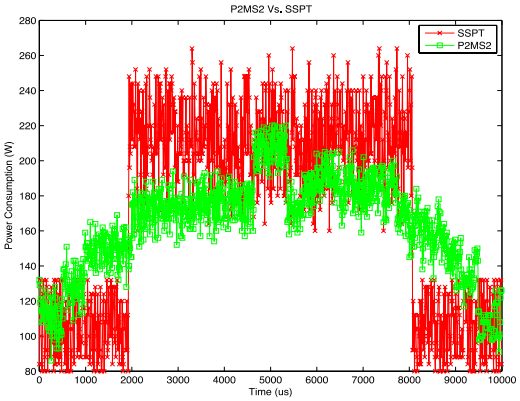
gsm	patricia	fft	bitcnts	basicmath	adpcm
۲۷/۱۳	۲۳/۸۴	۲۳/۶۱	۲۳/۳۵	۲۲/۴۷	۲۰/۷۴



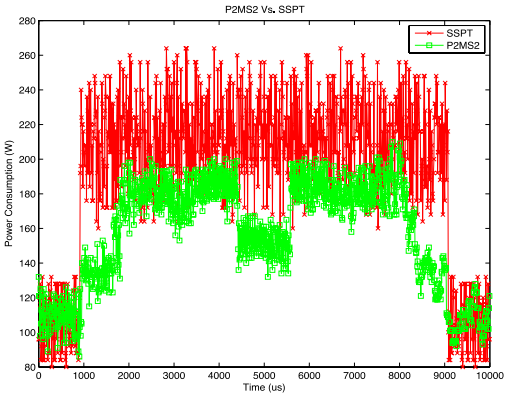
(الف) Uper-core = 0.6



(ب) Uper-core = 0.7

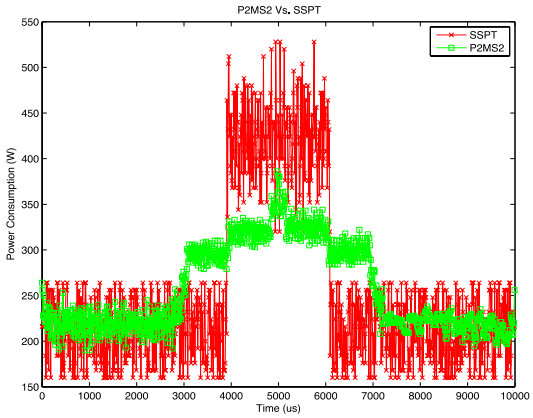


(پ) $U_{per\text{-}core}$ = 0.8

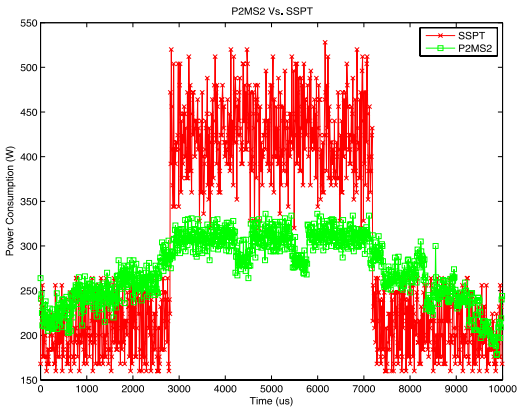


(ث) Uper-core = 0.9

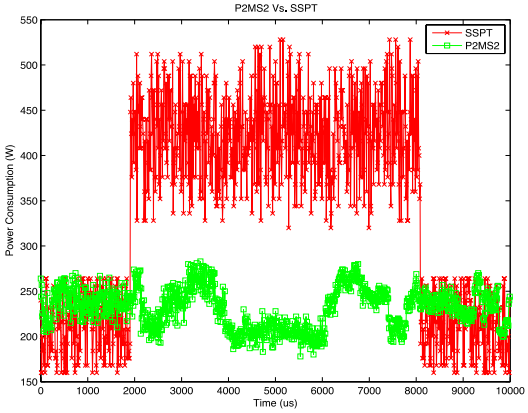
شکل ۶- نمودار توان مصرفی برای مجموعه وظیفه‌ای در یک قاب (توزیع یکنواخت، دارای هشت هسته، چهار زیرسیستم رزرو-آماده‌باش)



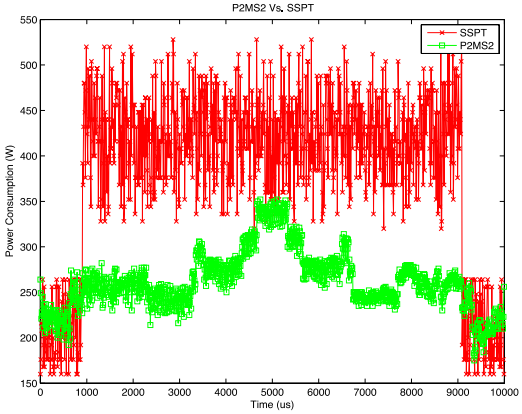
(الف) Uper-core = 0.6



(ب) Uper-core = 0.7



(پ) Uper-core = 0.8



(ث) Uper-core = 0.9

شکل ۷- نمودار توان مصرفی برای مجموعه وظیفه‌ای در یک قاب (توزیع یکنواخت، دارای شانزده هسته، هشت زیرسیستم رزرو-آماده‌باش)

در این مقاله ما توان را به‌صورت آفلاین مانیتور می‌کنیم. در شروع، نمودار توان مصرفی هر وظیفه را در نظر می‌گیریم. سپس، پارامتر فاصله‌ی زمانی پایه را بر اساس بزرگ‌ترین مقسوم‌علیه مشترکِ بدترین زمانِ اجرای وظایف محاسبه می‌کنیم. بر مبنای این پارامتر، تمام وظایف را به چند زیروظیفه تقسیم می‌کنیم تا به‌طور دقیق‌تری اوج توان مصرفی را به دست آوریم. بر این اساس هر زیروظیفه یک اوج توان مصرفی خاصی دارد که از نمودار توان مصرفی وظیفه حاصل شده است. درنهایت تمامی این اطلاعات را برای استفاده در مراحل بعدی مستندسازی می‌نماییم. در پایان این مرحله نیز یک روش تحمل‌پذیری اشکال همانند تکنیک رزرو-آماده‌باش انتخاب می‌شود.

در ادامه مجموعه‌ی وظایف را به‌صورت مصنوعی در نرم‌افزار Matlab به‌منظور شبیه‌سازی تولید می‌نماییم. در این شبیه‌سازی هر وظیفه دارای ۵ پارامتر است: ۱) بدترین زمان اجرا؛ ۲) تعدادی زیروظیفه؛ ۳) اوج توان مصرفی برای تمام زیروظایفِ آن وظیفه؛ ۴) ضرب‌الاجل؛ ۵) میزان بهره‌وری. در هر قاب، با توجه به میزان بهره‌وری کل سیستم که مورد نظر است، برای هر زیرسیستم به تعداد [۱۵۰، ۱۰۰] وظیفه تولید می‌کنیم. همچنین بدترین زمان اجرای وظایف را هم در بازه‌ی زمانی [۱۰۰، ۱۰] میکروثانیه و با نرخ تصادفی در نظر گرفته و تولید می‌کنیم. برای تعیین مقدار اوج توان مصرفی هر زیروظیفه، یعنی P_{ij}، از اطلاعات پژوهش [۲۶] استفاده می‌نماییم. اگرچه پژوهش [۲۶] از مدل ارائه شده در [۳۷] استفاده کرده است. در این مدل هر هسته فرض شده است که هسته‌ای از پردازنده Xeon شرکت اینتل[۵۴] در فناوری ۶۵ نانومتر تکنولوژی CMOS است و در حداکثر فرکانس کاری ۳/۴ گیگاهرتز کار می‌کند. برای به دست آوردن اوج توان مصرفی هر وظیفه روی این مدل، پژوهش [۲۶] چندین برنامه‌ی محکِ نهفته[۵۵] را با ورودی‌های مختلف از مجموعه‌ای از برنامه‌های محک MiBench [۱۲] روی شبیه‌ساز Wattch که در پژوهش [۲] ارائه شده است، اجرا کرده است. سپس با توجه به اطلاعات به‌دست‌آمده اوج توان مصرفی هریک از این برنامه‌ها در جدول ۲ نشان داده شده است. بنابراین مبتنی بر این مقادیرِ گزارش‌شده در پژوهش [۲۶] که در جدول ۲ آمده است، برای مجموعه‌ی وظایفی که تولید می‌کنیم، مقدار اوج توان مصرفی هر زیروظیفه را در بازه‌ی حداکثر و حداقل این جدول یعنی بازه‌ی [۳۳/۰۹، ۲۰/۷۴] در نظر می‌گیریم. نتایج شبیه‌سازی صورت گرفته در شکل‌های ۵، ۶ و ۷ به ترتیب برای سیستم‌هایی با ۴، ۸ و ۱۶ هسته نشان داده شده است. این نتایج، نمودار توان مصرفی برای یک قاب از اجرای وظایف در فاز برون‌خط است که نشان‌دهنده‌ی کارایی روش پیشنهادی نسبت به روش پژوهش [۱۳] در کاهش اوج توان مصرفی است. به‌طور متوسط ۱۸/۵ درصد (حداکثر ۳۸ درصد) بهبود در کاهش اوج توان مصرفی داشته‌ایم که در این شبیه‌سازی فرض نموده بودیم که مقدار TDP سیستم نسبت به حداکثر اوج توان مصرفی در پژوهش [۱۳] ۱۰ درصد پایین‌تر است. در شکل‌های ۵، ۶ و ۷ هرکدام دارای چهار شکل هستند که به ترتیب از چپ به راست میزان بهره‌وری هر هسته را نشان می‌دهد؛ به‌عبارت‌دیگر میزان بهره‌وری در الف، ب، پ و ث به ترتیب ۰/۶، ۰/۷، ۰/۸ و ۰/۹ برای هر هسته است.

۵- نتیجه‌گیری

افزایش نیازهای پردازشی، تعداد هسته‌های پردازشی، امکان مجتمع‌سازی و همچنین استفاده از روش‌های تحمل‌پذیری اشکال باعث می‌شوند توان مصرفی و به‌تبع آن اوج توان مصرفی افزایش یابند که می‌تواند موجب عبور از TDP گردد. از طرف دیگر روش‌های موجود تحمل‌پذیری اشکال در سیستم‌های بی‌درنگ از منظر تأثیر بر اوج توان مصرفی مورد مطالعه قرار نگرفته بودند و روش یا روش‌هایی برای دست‌یابی به تحمل‌پذیری اشکال در سیستم‌های بی‌درنگ با در نظر گرفتن محدودیت اوج توان ارائه نشده بود و تاکنون عمده‌ی تحقیقات انجام شده در این زمینه با هدف کاهش توان مصرفی متوسط و یا کاهش متوسط مصرف انرژی بوده است. روش ارائه شده سبب شد تا بدترین رفتار لحظه‌ایِ توان مصرفی روی یک تراشه، در سیستم‌های چندهسته‌ای تحمل‌پذیر اشکال کاهش یابد و به‌طور متوسط حدود ۱۸/۵ درصد و حداکثر ۳۸ درصد کاهش اوج توان را در سطح تراشه مشاهده بنماییم.

۶- مراجع

[1] L. Benini, A. Bogliolo, and G. De Micheli, "A Survey of Design Techniques for System-Level Dynamic Power Management," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 8, pp. 299 – 316, 2000.

[2] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A Framework for Architectural-Level Power Analysis and Optimizations," *in Proc. 27th annual Int'l symp. Computer architecture (ISCA)*, pp. 83-94, 2000.

[3] T. Chantem, X. Hu, and R. Dick, "Temperature-Aware Scheduling and Assignment for Hard Real-Time Applications on MPSoCs," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 10, pp. 1884-1897, 2011.

[4] J.-J. Chen, S. Wang, and L. Thiele, "Proactive Speed Scheduling for Real-Time Tasks under Thermal Constraints," *in Proc. IEEE 15th Real-Time Technology and Applications Symp. (RTAS)*, pp. 141-150, 2009.

[5] R. I. Davis and A. Burns, "A Survey of Hard Real-Time Scheduling for Multiprocessor Systems," *ACM Computing Surveys (CSUR)*, Vol. 43, no. 4, pp.1-44, 2011.

[6] A. Ejlali, B. M. Al-Hashimi, and P. Eles, "A Standby-Sparing Technique with Low Energy-Overhead for Fault-Tolerant Hard Real-Time Systems," *in Proc. IEEE/ACM 7th Int'l conf. Hardware/software codesign and system synthesis (CODES+ISSS)*, pp. 193-202, 2009.

[7] X. Wu and Z. Yan, "CAC Codec Designs Based On Numeral Systems," *in IEEE Workshop on Signal Processing Systems*, 2009.

[8] X. Wu and Z. Yan, "Efficient CODEC Designs for Crosstalk Avoidance Codes Based on Numeral Systems," *IEEE Transactions On Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 4, pp. 548-558, 2011.

[9] S. R. Sridhara, A. Ahmed, and N. R. Shanbhag, "Area and Energy-Efficient Crosstalk Avoidance Codes for On-Chip Buses," *in Computer Design: VLSI in Computers and Processors*, pp. 12-17, 2004 .

[10] W.-W. Hsieh, P.-Y. Chen, and C.-Y. Wang, "A Bus-Encoding Scheme for Crosstalk Elimination," *in IEEE Transactions On Computer-Aided Design Of Integrated Circuits And Systems*, vol. 26, no. 12, pp. 2222-2227, 2007 .

[11] K. Ramesh and E. Srinivas, "FPGA Implementation of Codec Design for Optimal," *in International Journal of Science and Research (IJSR)*, vol. 2, no. 2, pp. 614-622, 2013.

[12] C. Duan, K. Gulatit, and S. P. Khatrit, "memory-based Cross-talk Canceling CODECs for On-chip Buses," *in Proceeding of IEEE International Symposium on Circuits and Systems,* 2006 .

[13] C. Duan, C. Zhu, and S. P. Khatri, "Forbidden Transition Free Crosstalk," *in ACM/IEEE Design Automation Conference*, 2008 .

[14] M.A. Haque, H. Aydin, and D. Zhu, "Energy Management of Standby-Sparing Systems for Fixed-Priority Real-Time Workloads," *in Proc. Int'l Green Computing Conf. (IGCC)*, pp. 1-10, 2013.

[15] N. Hardavellas, M. Ferdman, B. Falsafi, and A. Ailamaki, *Toward Dark Silicon in Servers,* Micro-44, 2011.

[16] W.-L. Hung, Y. Xie, N. Vijaykrishnan, M. T. Kandemir, and M. J. Irwin, "Thermal-Aware Task Allocation and Scheduling for Embedded Systems," *in Proc. ACM/IEEE Conf. Design, Automation, and Test in Europe (DATE)*, vol. 2, pp. 898-899, 2005.

[17] Intel Corporation, "Desktop 3rd Generation Intel Core Processor Family Thermal Mechanical Specifications and Design Guidelines," 2013.

[18] C. Isci and M. Martonosi, "Runtime Power Monitoring in High-End Processors: Methodology and Empirical Data," *in Proc. IEEE/ACM 36th Int'l Symp. Microarchitecture (MICRO)*, pp. 93-104, 2003.

[19] V. Izosimov, I. Polian, P. Pop, P. Eles, and Z. Peng, "Analysis and Optimization of Fault-tolerant Embedded Systems with Hardened Processors," *in Proc. Design, Automation and Test in Europe Conf. and Exhibition (DATE)*, pp. 682-687, 2009.

[20] R. Jejurikar, C. Pereira, and R. Gupta, "Leakage Aware Dynamic Voltage Scaling for Real-Time Embedded Systems," *in Proc. IEEE 41st annual Design Automation Conf. (DAC)*, pp. 275-280, 2004.
[21] B. Johnson, *Design, and Analysis of Fault-Tolerant Digital Systems*, Reading, Mass.: Addison-Wesley Pub. Co., 1989.
[22] S. Kodase, S. Wang, Z. Gu, and K.G. Shin, "Improving Scalability of Task Allocation and Scheduling in Large Distributed Real-Time Systems Using Shared Buffers," *in Proc. IEEE 9th Real-Time Technology and Applications Symp. (RTAS)*, pp. 181-188, 2003.
[23] V. Kontorinis, A. Shayan, R. Kumar, and D. Tullsen, "Reducing Peak Power with a Table-Driven Adaptive Processor Core," *in Proc. 42nd Ann. IEEE/ACM Int'l Symp. Microarchitecture (MICRO)*, pp. 189-200, 2009.
[24] H. Kopetz, Real-Time Systems: *Design Principles for Distributed Embedded Applications*, KLUWER Academic Publishers, 2002.
[25] I. Koren, and C.M. Krishna, *Fault-Tolerant Systems*, Morgan Kaufmann, Elsevier, San Francisco, CA, 2007.
[26] J. Lee, B. Yun, and K. Shin, "Reducing Peak Power Consumption in Multi-Core Systems without Violating Real-Time Constraints," *IEEE Trans. Parall. Distr. Syst.,* vol. 25, No. 4, pp. 1024 – 1033, 2014 .
[27] B. Lee, J. Kim, Y. Jeung, and J. Chong, "Peak Power Reduction Methodology for Multi-Core Systems," *in Int'l SoC Design Conf. (ISOCC)*, pp.233-235, 2010.
[28] P. Marwedel, *Embedded System Design*. Dortmund (Germany): Springer, 2006.
[29] K. Meng, R. Joseph, R.P. Dick, and L. Shang, "Multi-Optimization Power Management for Chip Multiprocessors*," in Proc. 17th Int'l Conf. Parallel Architectures and Compilation Techniques (PACT)*, pp. 177-186, 2008.
[30] R. McGowen, C. Poirier, C. Bostak, J. Ignowski, M. Millican, W. Parks, and S. Naffziger, "Power and Temperature Control on a 90-nm Itanium Family Processor," *IEEE J. Solid-State Circuits*, vol. 41, no. 1, pp. 229-237, Jan. 2006.
[31] M. Mohaqeqi, M. Kargahi, and A. Movaghar, "Analytical Leakage-Aware Thermal Modeling of a Real-Time System," *IEEE Trans. Computers*, vol. 63, no. 6, pp. 1378-1392, 2014.
[32] P. Pillai, and K.G. Shin, "Real-Time Dynamic Voltage Scaling for Low-Power Embedded Operating Systems," *in Proc. 18th ACM Symp. Operating Systems Principles (SOSP)*, pp. 89-102, 2001.
[33] S. Polena, Fault-Tolerant Real-Time Systems: *The Problem of Replica Determinism,* KLUWER Academic Publishers, 1996.
[34] P. Pop, V. Izosimov, P. Eles, and Z. Peng, "Design Optimization of Time and Cost-Constrained Fault-Tolerant Embedded Systems with Checkpointing and Replication," *IEEE Trans. Very Large Integrated (VLSI) Systems*, vol. 17, pp. 389-402, 2009.
[35] F. R. Poursafaei, S. Safari, M. Ansari, M. Salehi, and A. Ejlali, "Offline Replication and Online Energy Management for Hard Real-Time Multicore Systems," *in Proc. 1th Int'l CSI Symp. Real-Time and Embedded Systems and Technologies (RTEST)*, pp. 1-7, 2015.
[36] D. Pradhan, *Fault Tolerant Computer System Design,* Prentice Hall, 1996.
[37] S. Rusu, S. Tam, H. Muljono, D. Ayers, and J. Chang, "A Dual- Core Multi-Threaded Xeon Processor with 16MB L3 Cache," *in Proc. IEEE Int'l Solid State Circuits Conf. (ISSCC)*, pp. 315-324, 2006.
[38] M.Salehi and A. Ejlali, "A Hardware Platform for Evaluating Low-Energy Multiprocessor Embedded Systems Based on COTS Devices," *IEEE Trans. Industrial Electronics*, vol. 62, no. 3, pp. 1262-1269, 2015.
[39] M. Salehi, A. Ejlali, and B.M. Al-Hashimi, "Two-Phase Low-Energy N-Modular Redundancy for Hard Real-Time Multi-Core Systems," *IEEE Trans. Parallel and Distributed Systems (TPDS)*, no. 99, 2015.
[40] M. Shafique, S. Garg, J. Henkel, and D. Marculescu, "The EDA Challenges in the Dark Silicon Era," *in Proc. IEEE 51st Int'l Design Automation Conf. (DAC)*, pp. 1-6, 2014.
[41] S. Wang, J.-J. Chen, Z. Shi, and L. Thiele, "Energy-Efficient Speed Scheduling for Real-Time Tasks under Thermal Constraints," *in Proc. IEEE 15th Int'l Conf. Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pp. 201-209, 2009.
[42] B. Yun, K. Shin, and S. Wang, "Thermal-Aware Scheduling of Critical Applications Using Job Migration and Power-Gating on Multi-core Chips," *in Proc. 10th Int'l Conf. Trust, Security and Privacy in Computing and Communications (TSPCC)*, pp. 1083-1090, 2011.
[43] J. Zhuo and C. Chakrabarti, "System-Level Energy-Efficient Dynamic Task Scheduling," *in Proc. 42nd Design Automation Conf. (DAC)*, pp. 628-631, 2005.
[44] D. Zhu and H. Aydin, "Reliability-Aware Energy Management for Periodic Real-Time Tasks," *IEEE Trans. Computers*, vol. 58, no. 10, pp. 1382 – 1397, 2009.
[45] W. Munawar, H. Khdr, S. Pagani, M. Shafique, J.-J. Chen, and J. Henkel, "Peak Power Management for Scheduling Real-Time Tasks on Heterogeneous Many-Core Systems," *in IEEE 20th Int'l Conf. Parallel and Distributed Systems (ICPADS)*, pp. 208-209, 2014.
[46] M. Khavari Tavana, M. Salehi, and A. Ejlali, "Feedback-Based Energy Management in a Standby-Sparing Scheme for Hard Real-Time Systems," *in Proc. IEEE Real-Time Systems Symp. (RTSS)*, pp. 349-356, 2011.
[47] Y. Guo, D. Zhu, and H. Aydin, "Reliability-Aware Power Management for Parallel Real-Time Applications with Precedence Constraints," *in Proc. Int'l Green Computing Conf. and Workshops (IGCC)*, pp.1-8, 2011.
[48] X. Qi, D. Zhu, and H. Aydin, "Global Scheduling Based Reliability-Aware Power Management for Multiprocessor Real-Time Systems," *J. Real-Time Syst.*, vol. 47, no. 2, pp. 109-142, 2011.
[49] S. Pagani, H. Khdr, J. Chen, M. Shafique, M. Li and J. Henkel, "Thermal Safe Power (TSP): Efficient Power Budgeting for Heterogeneous Manycore Systems in Dark Silicon," *IEEE Transactions on Computers*, vol. 66, no. 1, pp. 147-162, 1 Jan. 2017.
[50] M. Ansari, S. Safari, A. Y. Khaksar, M. Salehi, and A. Ejlali, "Peak Power Management to Meet Thermal Design Power in Fault-Tolerant Embedded Systems," *IEEE Trans. on Par. and Dis. Sys.*, vol. 30, no. 1, 2019.
[51] H. Khdr, S. Pagani, E. Sousa, V. Lari, A. Pathania, F.Hanning, M. Shafique, J. Teich, and J. Henkel "Power Density-Aware Resource Management for Heterogeneous Tiled Multicores," *in IEEE Transactions on Computers*, vol. 66, no. 3, pp. 488-501, 2017.
[52] S. Safari, M. Ansari, G. Ershadi and S. Hessabi, "On the Scheduling of Energy-Aware Fault-Tolerant Mixed-Criticality Multicore Systems with Service Guarantee Exploration," *IEEE Transactions on Parallel and Distributed Systems*, 2019.
[53] M. Shafique, S. Garg, "Computing in the Dark Silicon Era: Current Trends and Research Challenges," *IEEE Design & Test (DnT)*, vol. 34, no. 2, pp. 5-7, 2017.

محسن انصاری، مدرک کارشناسی ارشد خود را در سال ۱۳۹۵ از دانشگاه صنعتی شریف اخذ کرده است. از سال ۱۳۹۵ تاکنون محسن انصاری دانشجوی دکتری در رشته مهندسی کامپیوتر در دانشگاه شریف است. در حال حاضر ایشان عضو آزمایشگاه طراحی سیستم‌های نهفته دانشکده مهندسی کامپیوتر می‌باشند. زمینه‌های تحقیقاتی ایشان را موارد گسترده‌ای همچون طراحی سیستم‌های کم‌توان، مدیریت اوج توان مصرفی در سیستم‌های نهفته و سیستم‌های چندهسته‌ای با تمرکز بر قابلیت اطمینان و اتکاپذیری در بر می‌گیرد.
آدرس پست الکترونیکی ایشان عبارت است از:
mansari@ce.sharif.edu

مصطفی پسندیده دانشجوی کارشناسی ارشد ورودی ۱۳۹۶ در رشته مهندسی کامپیوتر گرایش معماری کامپیوتر در دانشگاه شریف است. ایشان مدرک کارشناسی خود را از دانشگاه شاهد در رشته مهندسی کامپیوتر اخذ کرده‌اند. از علایق تحقیقاتی وی می‌توان به طراحی سیستم‌های کم‌توانِ نهفته و مدیریت قابلیت اطمینان در سیستم‌های چندهسته‌ای نهفته اشاره کرد.
آدرس پست الکترونیکی ایشان عبارت است از:
smpasandideh@ce.sharif.edu

علیرضا اجلالی، مدرک دکتری خود را در سال ۱۳۸۴ در رشته مهندسی کامپیوتر از دانشگاه صنعتی شریف دریافت کرده‌اند. ایشان در حال حاضر دانشیار دپارتمان مهندسی کامپیوتر هستند. در خلال سال‌های ۱۳۸۳ و ۱۳۸۴ ایشان استادیار در گروه

طراحی سیستم‌های الکترونیکی واقع در دپارتمان کامپیوتر دانشگاه ساوتهمپتون واقع در انگلستان بوده‌اند. ایشان در سال ۱۳۸۵ به‌عنوان استاد به دانشگاه شریف ملحق شده‌اند و در خلال سال‌های ۱۳۹۰ تا ۱۳۹۴ به‌عنوان سرپرست گروه معماری کامپیوتر مشغول بوده‌اند. از زمینه‌های تحقیقاتی ایشان می‌توان به مواردی همچون طراحی سیستم‌های کم‌توان، سیستم‌های بی‌درنگ نهفته و سیستم‌های نهفته تحمل‌پذیر خطا اشاره کرد.

آدرس پست الکترونیکی ایشان عبارت است از:

ejlali@sharif.edu

[1] Power Densities
[2] Reliability
[3] Energy Consumption
[4] Chip Temperature
[5] Average Power Consumption
[6] Worst-case Behavior of Instantaneous Power Consumption
[7] Peak Power Consumption
[8] Dark Silicon
[9] Thermal Design Power
[10] Performance Throttling Mechanisms
[11] Dynamic Thermal Management
[12] Hard Real-Time
[13] Fault-Tolerant
[14] Deadline
[15] Slack Time
[16] Fault Recovery
[17] Standby-Sparing
[18] Primary Tasks
[19] Backup Tasks
[20] Dynamic Voltage and Frequency Scaling
[21] General Purpose
[22] Frame-based
[23] Utilization
[24] Data Dependency
[25] Task Graph
[26] Rate Monotonic Scheduling
[27] Fixed Priority
[28] Sporadic
[29] Load Balancing
[30] Firm Real-Time
[31] Power Gating
[32] Sleeping Mode
[33] Dynamic Power Management
[34] Earliest-Deadline-First
[35] Earliest-Deadline-Late
[36] Worst-Case Execution Time
[37] Basis-Time-Interval
[38] Greastest Common Divisor
[39] Sub-Task
[40] Static Power
[41] Dynamic Power
[42] Leakage
[43] Frequency-dependent Power Consumption
[44] Frequency-independent Power Consumption
[45] Worst-case Instantaneous Power Consumption
[46] Fault Model
[47] Acceptance Test
[48] Consistency
[49] Overlap Table
[50] Hot Standby-Sparing
[51] Cold Standby-Sparing
[52] Maximum Peak Power First (MPPF)
[53] Maximum Peak Power Late (MPPL)
[54] Intel Xeon Processor
[55] Embedded Benchmark Programs

The CSI Journal on Computing Science and Information Technology
Vol. 17, No.1, 2019

Peak Power Management in Standby-Sparing Systems

Mohsen Ansari, Mostafa Pasandideh, and Alireza Ejlali

Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

Abstract

Due to the Thermal Design Power (TDP) constraint, it is not possible to simultaneously power-on all cores in a multicore chip. Violating TDP triggers a performance throttling scenario to avoid possible overheating problems. This can remarkably affect the timeliness of the system. That means only a few tasks can be afforded to run in a fully reliable mode under TDP constraint. Then, to come along with peak power problem, we propose a novel scheme for scheduling frame-based real-time tasks in multicore systems. The proposed scheme aims at removing overlaps of peak power of concurrently executing tasks to keep the power consumption below the chip-level TDP constraint. To do this, considering the tasks' power profiles, we propose a task partitioning method policies to schedule original and redundant copies of tasks, respectively.

Keywords: Peak Power Consumption, Standby-Sparing Systems, Multicore Platforms.