بهبود عملکرد الگوریتم ضرب کم توان برای کاربردهای پردازش سیگنال در شبکه‌ی حسگرهای بی‌سیم

عبدالرسول جعفری
گروه مهندسی برق و کامپیوتر، دانشکده‌ی فنی، دانشگاه تهران
ar.jafari@ut.ac.ir
زهرا جعفری
گروه مهندسی برق و کامپیوتر، دانشکده‌ی فنی، دانشگاه پیام نور
jafari.z@pnu.ac.ir

چکیده:
عملیات ضرب یکی از مهمترین عملیات پایه در الگوریتم‌های پردازش سیگنال، به ویژه کاربردهای شبکه‌ای از حسگرهای بی‌سیم می‌باشد. به نحوی که کارایی آن بطور مستقیم کارایی کل سیستم را تحت الشعاع قرار می‌دهد. در بعضی از کاربردهای خاص امکان استفاده از مدارهای سخت‌افزاری ضرب کننده وجود ندارد، یا این واحد به طور کلی موجود نیست و یا به خاطر ملاحظات مصرف توان و انرژی استفاده از آن مقرون به صرفه نیست. در چنین کاربردهایی می‌بایست از الگوریتم‌های نرم‌افزاری بهره گرفت. میزان متوسط مصرف انرژی در چنین سیستم‌هایی بستگی مستقیم به تعداد عملیات پایه‌ای دارد که انجام می‌شوند. الگوریتم **Horner** یکی از معروف‌ترین روش‌هایی است که در این زمینه استفاده شده‌است. در این مقاله براساس الگوریتم **Horner** الگوریتم دیگری پیشنهاد شده‌است که برتری‌هایی نسبت به حالت استاندارد الگوریتم **Horner** دارا می‌باشد.

واژگان کلیدی: الگوریتم ضرب دودویی، طراحی سیستم‌های کم‌توان، الگوریتم **Horner**، شبکه‌ی حسگرهای بی‌سیم.

مقدمه

پردازش سیگنال درشبکه حسگرهای بی‌سیم (WSN)[1] کاربردهای متنوعی دارد. به عنوان مثال می‌توان به فیلترهای FIR[2] و IIR[3] و کالامان[4] اشاره کرد که در کاربردهای ردگیری اشیاء، نظارت محیطی و بسیاری کاربردهای دیگر نقش حیاتی ایفا می‌کنند. انجام این کارها نیاز به محاسبات بسیار پیچیده‌ای دارد و انرژی بسیار زیادی از منابع هر یک از گره‌های درون یک شبکه از حسگرهای بی‌سیم را مصرف می‌کند [۱]. عملیات ضرب، هسته‌ی اصلی بسیاری از این سیستم‌های پرازش سیگنال می‌باشد. بدیهی است که در صورتی که در این حسگرها از ضرب کننده‌هایی استفاده گردد که مصرف انرژی کمتری داشته باشند، در افزایش طول عمر استفاده از آن‌ها تاثیر بسزایی خواهد داشت. در این مقاله الگوریتم ضربی معرفی خواهد شد که برای استفاده در میکروکنترلرهایی که در این شبکه‌ها کاربرد دارند، مناسب می‌باشد. معمولاً در این شبکه‌ها از سخت‌افزار اختصاصی برای عملیات ضرب استفاده نمی‌شود، زیرا یا چنین واحدی به طور کلی وجود ندارد و یا درصورت وجود، به دلیل آن که معمولاً چنین واحدهایی دارای مصرف توان قابل توجهی هستند و باعث کاهش عمر شبکه‌ی حسگرها خواهند شد، استفاده از آن‌ها قابل توجیه نخواهد بود.

در روشی که در این مقاله معرفی خواهد شد، با کاهش تعداد عملیات جمع مورد نیاز برای انجام عملیات ضرب، انرژی مصرف شده کاهش خواهد یافت. ایده‌ی اصلی این الگوریتم، انجام عملیات در مبناهای بزرگ‌تر از ۲ می‌باشد. به کارگیری این روش، بدون اینکه باعث افزایش پیچیدگی برنامه‌ی نوشته شده‌شود، تعداد کل دستورالعمل‌های لازم جهت اجرا را کاهش خواهد داد. حتی کاهشی نیز در دقت محاسبات نخواهیم داشت.

شبیه‌سازی‌های انجام شده، نشان دهنده‌ی کاهش ۵۰ درصدی مصرف انرژی و به همین میزان، افزایش سرعت محاسبات می‌باشد، این درحالی است که کاهشی نیز در دقت محاسبات انجام شده در مقایسه با روش پایه‌ی Horner [۲] نخواهیم داشت. به منظور اثبات کارایی مثبت این روش از آن در پیاده‌سازی یک فیلتر FIR خاص نیز بهره گرفته شده‌است.

در ادامه ابتدا کمی در مورد ماهیت شبکه‌ی حسگرهای بی‌سیم صحبت خواهد شد و سپس در بخش (۳) مروری بر الگوریتم‌های ضرب موجود خواهیم داشت. در بخش (۴) به نحوه‌ی سنجش میزان انرژی مصرفی پرداخته‌ایم. بخش‌های (۵) و (۶) به ترتیب به معرفی الگوریتم پیشنهادی و

1 Wireless Sensor Network (WSN)

2 Finite Impulse Response

3 Infinite Impulse Response

4 Kalaman

نتایج مقایسه‌ای حاصل از اجرای آن اختصاص یافته‌است و در نهایت بخش (۷) به خلاصه و نتیجه‌گیری اختصاص یافته‌است.

شبکه‌ی حسگرهای بی‌سیم

پیشرفت‌های اخیر در زمینه الکترونیک و مخابرات بی‌سیم توانایی طراحی و ساخت حسگرهایی را با توان مصرفی پایین، اندازه کوچک، قیمت مناسب و کاربری‌های گوناگون داده است. این حسگرهای کوچک که توانایی انجام اعمالی چون دریافت اطلاعات مختلف محیطی (براساس نوع حسگر، پردازش و ارسال آن اطلاعات را دارند، موجب پیدایش ایده‌ای برای ایجاد و گسترش شبکه‌های موسوم به شبکه‌های بی‌سیم حسگر شده‌اند [۳].

یک شبکه حسگر متشکل از تعداد زیادی گره‌های حسگری است که در یک محیط به طور گسترده پخش شده و به جمع‌آوری اطلاعات از محیط می‌پردازند. لزوماً مکان قرار گرفتن گره‌های حسگری، از قبل تعیین‌شده و مشخص نیست. چنین خصوصیتی این امکان را فراهم می‌آورد که بتوانیم آنها را در مکان‌های خطرناک و یا غیرقابل دسترس رها کنیم. از طرف دیگر این بدان معنی است که پروتکل‌ها و الگوریتم‌های شبکه‌های حسگری باید دارای توانایی‌های خودساماندهی باشند [۴].

دیگر خصوصیت‌های منحصر به فرد شبکه‌های حسگری، توانایی همکاری و هماهنگی بین گره‌های حسگری است. هر گره حسگر روی برد خود دارای یک پردازشگر است و به جای فرستادن تمامی اطلاعات خام به مرکز یا به گره‌ای که مسئول پردازش و نتیجه‌گیری اطلاعات است، ابتدا خود یک سری پردازش‌های اولیه و ساده را روی اطلاعاتی که به دست آورده است، انجام می‌دهد و سپس داده‌های نیمه پردازش شده را ارسال می‌کند. با اینکه هر حسگر به تنهایی توانایی ناچیزی دارد، ترکیب صدها حسگر کوچک امکانات جدیدی را عرضه می‌کند. در واقع قدرت شبکه‌های بی‌سیم حسگر در توانایی به‌کارگیری تعداد زیادی گره کوچک است که خود قادرند سرهم و سازماندهی شوند و در موارد متعددی چون مسیریابی هم‌زمان، نظارت بر شرایط محیطی، نظارت بر سلامت ساختارها یا تجهیزات یک سیستم به کار گرفته شوند. گستره ی کاربری شبکه‌های بی‌سیم حسگر بسیار وسیع بوده و از کاربردهای کشاورزی، پزشکی و صنعتی تا کاربردهای نظامی را شامل می‌شود. به عنوان مثال یکی از متداول‌ترین کاربردهای این تکنولوژی، نظارت بر یک محیط دور از دسترس است. مثلاً نشتی یک کارخانه ی شیمیایی در محیط وسیع کارخانه می‌تواند توسط صدها حسگر که به طور خودکار یک شبکه ی بی‌سیم را تشکیل می‌دهند، نظارت شده و در هنگام بروز نشت شیمیایی به سرعت به مرکز اطلاع داده شود [۵].

در این سیستم‌ها بر خلاف سیستم‌های سیمی قدیمی، از یک سو هزینه‌های پیکربندی و آرایش شبکه کاسته می‌شود. از سوی دیگر به جای نصب هزاران متر سیم فقط باید دستگاه‌های کوچکی را

که تقریباً به اندازه یک سکه هستند را در محل مناسبی مورد استفاده قرار داد. شبکه حسگر بی‌سیم به یک شبکه بی‌سیم از حسگرهای خودراهبر گفته می‌شود که با فاصله از هم پخش شده‌اند و برای اندازه‌گیری گروهی برخی از کمیت‌های فیزیکی یا شرایط محیطی مانند دما، صدا، لرزش، فشار، حرکت یا آلاینده‌ها، در مکان‌های مختلف یک محدوده جغرافیایی کاربرد دارند. شبکه‌های حسگر با انگیزه استفاده در کاربردهای نظامی مانند نظارت بر میدان جنگ، توسعه پیدا کرد. اما امروزه شبکه‌های حسگر بی‌سیم در صنعت و بسیاری از مقاصد غیر نظامی استفاده می‌شوند، از جمله: نظارت و کنترل فرآیندهای صنعتی، نظارت بر سلامت دستگاه‌ها، نظارت بر محیط و یا خانه، کاربردهای مراقبت از سلامتی، خانه‌های هوشمند و کنترل ترافیک [۶].

علاوه بر یک یا چند سنسور، هر گره از شبکه معمولاً مجهز به یک فرستنده و گیرنده‌ی رادیویی (یا هر وسیله‌ی مخابراتی بی‌سیم دیگر)، یک میکروکنترلر کوچک، و یک منبع انرژی (معمولاً یک باتری)، می‌باشد. اندازه یک گره سنسوری بسته به اندازه‌ی بسته‌بندی آن تغییر کرده و تا یک دانه شن قابل کوچک‌سازی است. به طور مشابه قیمت هر گره حسگر می‌تواند بین بین چند میلیون تا چند صد ریال، بسته به اندازه و پیچیدگی مورد نیاز یک گره متفاوت باشد. محدودیت‌های قیمت و اندازه در گره‌های حسگر منجر به محدودیت در منابعی مانند انرژی، حافظه، سرعت پردازش و پهنای باند در آن‌ها می‌شود [۳].

در حال حاضر شبکه‌های بی‌سیم حسگر یکی از موضوعات فعال تحقیقاتی در علوم کامپیوتر و ارتباطات است که هر ساله تعداد بی‌شماری کارگاه و کنفرانس در این زمینه انجام می‌شود.

پردازش سیگنال درشبکه حسگرهای بی‌سیم کاربردهای متنوعی دارد. به عنوان مثال می‌توان به فیلترهای FIR، IIR و کالامان اشاره کرد که در کاربردهای ردگیری اشیاء، نظارت محیطی و بسیاری کاربردهای دیگر نقش حیاتی ایفا می‌کنند. انجام این کارها نیاز به محاسبات بسیار پیچیده‌ای دارد و انرژی بسیار زیادی از منابع هر یک از گره‌های درون یک شبکه از حسگرهای بی‌سیم را مصرف می‌کند [۱].

بنابراین معمولاً در این گره‌ها به طور کلی از تعبیه‌ی واحدهای سخت‌افزاری خاص انجام اینگونه عملیات پردازش سیگنال اجتناب می‌گردد و از راه‌حل‌های جایگزین استفاده می‌گردد. یکی از این روش‌های جایگزین انجام و شبیه‌سازی این محاسبات به صورت نرم‌افزاری می‌باشد. از طرفی هسته‌ی اصلی بیشتر این عملیات پردازش سیگنال، عملیات ضرب می‌باشد. بنابراین انجام عملیات ضرب به لحاظ مصرف انرژی تأثیر بسیار زیادی بر روی کل انرژی مصرفی هریک از گره‌ها خواهد داشت. بنابراین انجام عملیات ضرب به صورت بهینه عمر یک شبکه‌ی حسگر بی‌سیم (WSN) را افزایش خواهد داد. لازم به ذکر است که گره‌های شبکه‌های حسگر بی‌سیم، معمولاً با باتری کار می‌کنند و نگهداری از آن‌ها و تعویض باتری‌ها بسیار هزینه‌بر و گاهی غیر ممکن خواهد بود. زیرا از

این شبکه‌ها معمولاً در طبیعت و شرایط محیطی بسیار سخت استفاده می‌گردد و خیلی مواقع با اتمام باتری، عمر آن‌ها نیز به پایان خواهد رسید. به همین دلیل کاهش مصرف انرژی، به طور مستقیم باعث افزایش عمر و کاهش هزینه‌ی نگهداری استفاده از چنین شبکه‌هایی خواهد شد[۱].
گره‌های حسگری که امروزه در بازار موجود هستند مانند Micaz [۷] که مبتنی بر یک میکروکنترلر ۸ بیتی و Telosb [۸] که دارای یک میکروکنترلر ۱۶ بیتی هستند، هیچ یک ضرب کننده‌ی ممیز شناور (Floating Point) ندارند و حتی عملیات ضرب صحیح درون آن‌ها نیز در صورت فعال شدن به خاطر مصرف توان بالا کلیه‌ی منابع انرژی را به یک‌باره مصرف می‌کند. برای غلبه بر چنین مشکلاتی الگوریتم‌های ضرب متعددی پیشنهاد شده‌است که تقریباً همگی مبتنی بر جمع‌های متوالی هستند و می‌بایست دستورالعمل‌های زیادی درون پردازنده اجرا گردند تا عملیات ضرب انجام گردد. علاوه براین معمولاً دقت محاسبات انجام شده نیز ممکن است کافی نباشد.

مروری بر الگوریتم‌های ضرب موجود

عملیات ضرب اغلب برای کارایی بهتر سخت‌افزار به صورت شیفت و جمع پیاده‌سازی می‌گردد [۹].
در این روش با ضرب هریک از رقم‌های مضروب در مضروبٌ فیه تعدادی حاصل‌ضرب جزیی به دست می‌آید. هریک از این حاصل‌ضرب‌های جزیی به اندازه یک رقم شیفت یافته و سپس حاصل‌ضرب‌های جزیی با هم جمع می‌گردند تا حاصل‌ضرب نهایی به دست آید. حاصل‌ضرب دودویی نیز به همین صورت به دست می‌آید. البته از آنجایی که ارقام دودویی تنها صفر و یک هستند هریک از حاصل‌ضرب‌های جزیی یا تنها صفر می‌باشد و یا دقیقاً یک کپی از مضروب می‌باشد. به جای ضرب در رقم صفر حاصل‌ضرب جزیی صفر و به جای ضرب در یک همان عدد را کپی می‌کنیم. به عنوان مثال فرض کنید که قرار است حاصل ضرب دو عدد A و B را محاسبه کنیم:

$A = 0.14325 = 0.001001001010$
$B = 0.12345 = 0.000111111001$

در روش سنتی این حاصل‌ضرب به صورت زیر به دست می‌آید:

$$0.14325 * 0.12345 =$$
$$0.001001001010_b * (2^{-4}+2^{-5}+2^{-6}+ 2^{-7}+2^{-8}+2^{-9}+2^{-12}) =$$
$$0.000000100100_b +$$
$$0.000000010010_b +$$
$$0.000000001001_b +$$
$$0.000000000100_b +$$
$$0.000000000010_b +$$

0.000000000001_b +
0.000000000000_b +
0.000001000110_b = 0.01708984375

حاصل دقیق این حاصل‌ضرب برابر 0.0176842125 است. بنابراین این روش سنتی باعث 0.00059436875 خطا خواهد شد که معادل تقریباً 2.5LSB می‌باشد. خطای حاصل به علت اثر طول محدود اعداد[1] [۱۰] که ناشی از طول محدود رجیسترها می‌باشد رخ می‌دهد. برای کاهش این خطا می‌بایست تعداد بیت‌های نمایش دهنده‌ی هر عدد را افزایش داد. ولی با استفاده از روش هورنر (Horner) در حالی‌که طول کلمات ثابت می‌باشد می‌توان این خطا را به نحو قابل ملاحظه‌ای کم کرد.

از روش هورنر عمدتاً برای انجام عملیات ضرب در وسایلی که سخت‌افزار ضرب اختصاصی ندارند استفاده می‌گردد [۱۱]. در این روش تعدادی معادله وجود دارد که به ازای هر مضروبی منحصر بفرد می‌باشند. این معادلات در واقع بیان کننده‌ی دنباله‌ی شیفت و جمع‌هایی هستند که باید بر روی مضروبٌ فیه اعمال گردند. الگوریتم هورنر مبتنی بر موقعیت یک‌ها در مضروب و فاصله‌ی آن‌ها از یکِ دیگری که بلافاصله در سمت چپ بعد از آن قرار گرفته‌است، می‌باشد. در این الگوریتم از سمت راست‌ترین بیت شروع می‌کنیم و با حرکت به سمت چپ و رسیدن به آخرین یک قبل از علامت ممیز ادامه می‌دهیم. در عدد 0.14325 که معادل باینری آن 0.001001001010_b می‌باشد، از سمت راست شروع می‌کنیم. اولین یک در موقعیت 2^{-11} می‌باشد و فاصله‌ی آن تا یک بعد از آن ۲ می‌باشد و به همین صورت فاصله تا یک بعدی ۳ و .... اگر عددی که قرار است در این عدد ضرب گردد را A فرض کنیم، معادلات مربوط به این حاصل ضرب به صورت زیر تبدیل خواهد شد:

$A_1=A*2^{-3}+A$
$A_2=A_1*2^{-1}+A$
$A_3=A_2*2^{-1}+A$
$A_4=A_3*2^{-1}+A$
$A_5=A_4*2^{-1}+A$
$A_6=A_5*2^{-1}+A$

نتیجه نهایی برابر است با $A_6*2^{-4}=0.017578125$ یا به عبارتی دیگر تنها 0.43LSB خطا خواهیم داشت.

همانگونه که ملاحظه می‌گردد تعداد عملیات جمع که در واقع معیاری از انرژی مصرفی می‌باشد در روش پایه‌ی هورنر برابر تعداد یک‌های مضروب می‌باشد. بنابراین شاید اولین روشی که برای کاهش

1 Finite Word Length Effect

مصرف انرژی به ذهن می‌رسد کاستن از تعداد یک‌ها می‌باشد. در [۱۲] برای انجام این کار از یک روش تقریبی استفاده شده‌است که هرچند باعث کاهش دقت محاسبات شده‌است ولی کاهش تعداد عملیات جمع و به طبع آن کاهش مصرف انرژی را نیز در پی داشته است. در [۲] چند روش Heuristic برای انجام این کار معرفی شده‌است. هرچند که در بعضی موارد کارایی خوبی را موجب می‌گردد، اما در حالت کلی قابل بسط نیست. در این مقاله با انجام عمل ضرب در مبناهای بالاتر عدد نویسی (مشابه روش Booth [۱۳]) تعداد حاصل‌ضرب‌های جزیی کمتری برای جمع کردن ایجاد شده و در نتیجه مصرف توان کمتری نیز موجب می‌گردد.

معیار سنجش میزان انرژی مصرفی

مبنای اندازه‌گیری انرژی مصرف شده در این مقاله، سنجش میزان فعالیت مدار می‌باشد. زیرا انرژی کل مصرفی یک مدار الکترونیکی از رابطه‌ی $P_{total} = P_{dyn} + P_{static}$ به دست می‌آید [۱۴]. در این رابطه P_{static} نشان دهنده‌ی توان مصرفی استاتیکی مدار می‌باشد. این پارامتر همواره مقداری ثابت دارد و مقدار آن قابل تغییر نیست. اما P_{dyn} نشان دهنده‌ی انرژی دینامیکی مصرف شده در مدار می‌باشد که بستگی مستقیم به سطح فعالیت آن دارد. انرژی دینامیکی مصرف شده در مدار نیز از رابطه‌ی $P_{dyn} = \alpha C V_{DD}^2$ به دست می‌آید [۱۴]. که در آن V_{DD} ولتاژ کاری میکروکنترلر، C مجموع خازن‌های کل گیت‌های درون میکروکنترلر و α ضریب فعالیت مدار و یا به عبارت دیگر نرخ شارژ و دشارژ شدن خازن‌های درون میکروکنترلر می‌باشد.

از میان این پارامترها تنها فعالیت مدار، α، توسط برنامه‌نویسان یک میکروکنترلر قابل تنظیم می‌باشد. به همین دلیل میزان مصرف انرژی در این الگوریتم‌ها نیز بر مبنای همین پارامتر گزارش شده‌است.

الگوریتم پیشنهادی

در این مقاله، یک الگوریتم ضرب برای محاسبه‌ی حاصلضرب دو عدد ممیز ثابت بین صفر تا یک ارایه شده‌است به گونه‌ای که طول حاصلضرب ایجاد شده نیز با عملگرهای آن برابر است. بدیهی است که در چنین مسائلی همواره محاسبات دارای مقداری خطا نیز خواهد بود. می‌توان نشان داد که روش Horner بهترین جواب و با کمترین خطای ممکن را تولید می‌کند [۱۵]. در [۱۲] با در نظر گرفتن کاربرد چنین ضرب کننده‌هایی که در طراحی فیلترهای دیجیتال می‌باشد و معمولاً یک عملوند همواره ثابت می‌باشد، روشی پیشنهاد شده‌است که هرچند باعث کاستن از دقت محاسبات می‌شود، اما تعداد حاصل‌ضرب‌های جزئی کمتری ایجاد شده و در نتیجه تعداد عملیات‌جمع کمتری نیز انجام خواهد شد. بنابراین هم سرعت محاسبات افزایش خواهد یافت و هم توان دینامیکی

کمتری مصرف خواهد شد. ایده‌ی اصلی مورد استفاده در [۱۲]، گرد کردن دنباله‌ی یک‌های موجود در عملوند ثابت می‌باشد. بنابراین در برنامه‌ی اسمبلی نوشته شده در میکروکنترلری که عملیات ضرب برروی آن انجام می‌شود، تنها تعدادی عمل جمع، متناسب با مکان‌هایی که ضریب ثابت دارای مقادیر غیر صفر می‌باشد، انجام می‌گردد. بدیهی است که اساس کار این الگوریتم بر گرد کردن عدد است که خود به‌خود باعث کاهش دقت محاسبات خواهد شد. روشی که در این مقاله پیشنهاد شده‌است، مبتنی بر انجام محاسبات در مبناهای بزرگتر می‌باشد. خواهیم دید که هرچقدر مبنای مورد استفاده بزرگتر باشد، سرعت بیشتر، مصرف توان کمتر و در عوض حافظه‌ی بیشتری برای انجام محاسبات مورد نیاز خواهد بود.

همانگونه که می‌دانیم حاصلضرب دو عدد X و C در مبنای r در روش Horner از رابطه‌ی زیر به دست می‌آید [۱۵]:

$$XC = C\sum_{i=0}^{n-1} x_i r^i = Cx_0 + r(Cx_1 + r(Cx_2 + r(\dots)))$$

از طرفی می‌دانیم که C ضریب ثابت می‌باشد، به همین دلیل مقدار x_iC تنها مقدار مختلف و ثابت به خود می‌گیرد. زیرا C ثابت بوده و x_i تنها r حالت مختلف دارد. برای محاسبه‌ی این حاصلضرب به سادگی می‌توان آنها را در یک جدول ذخیره کرد و با توجه به مقدار x_i نتیجه را جستجو کرد. اگر r بفرم 2^n باشد، حاصلضرب $r(\dots)$ را نیز با عمل شیفت می‌توان پیاده‌سازی کرد. بنابراین با فرض $r = 16$ از الگوریتم زیر می‌توان برای محاسبه‌ی حاصلضرب مورد نظر بهره گرفت:

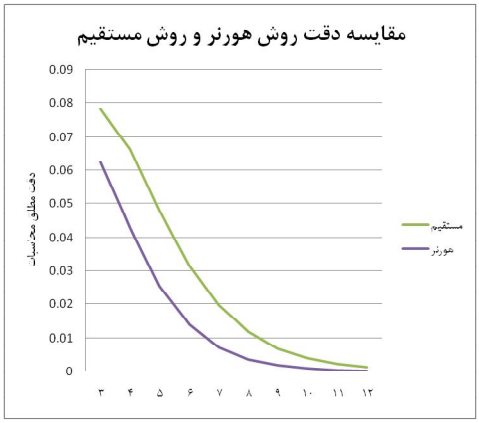
```
const int C = 0x1234; //Const. Coefficient
const int CLUT[16] = {
    0*C,  1*C,  2*C,  3*C,
    4*C,  5*C,  6*C,  7*C,
    8*C,  9*C, 10*C, 11*C,
   12*C, 13*C, 14*C, 15*C
};

int mul(int op, int size) {
   int result = 0;
   for(int i=size/4; i>0; i--) {
      result = result >> 4;
      result += CLUT[ op & 0xF ];
      op = op >> 4;
   }
   return result;
}
```

شکل (۴): الگوریتم پیشنهادی برای ضرب ضریب ثابت و یک عدد

نتایج به دست آمده از شبیه‌سازی‌ها

در این قسمت نتایج به دست آمده از آزمایش‌های انجام شده و مقایسه‌ی آن‌ها با سایر روش‌ها ارایه شده‌است. در شکل (۱) مقایسه‌ای بین دقت الگوریتم Horner و روش مستقیم صورت گرفته‌است:

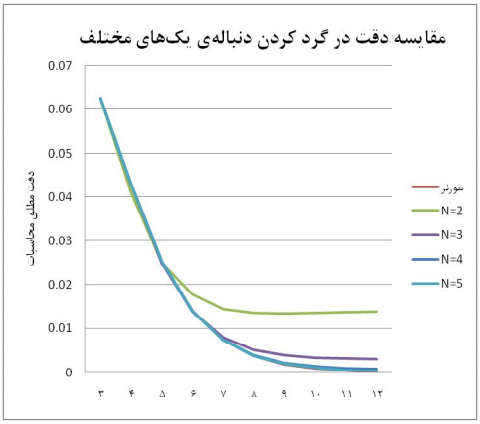


شکل (۱): مقایسه دقت روش هورنر و مستقیم

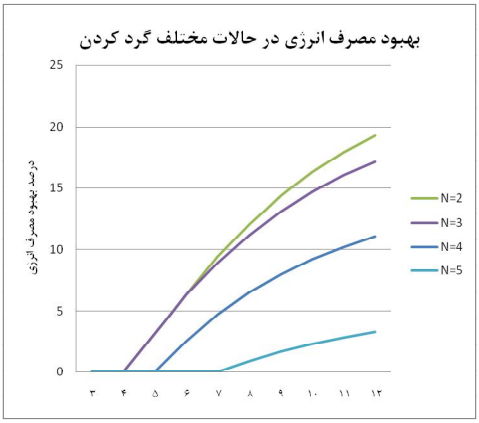
در روش پیشنهاد شده در [۱۲] می‌بایست دنباله‌ی یک‌ها را تقریب زد تا تعداد حاصل‌جمع‌های جزئی ایجاد شده کمتر شود و در نتیجه تعداد عملیات جمع کمتری صورت پذیرد و به تبع آن انرژی کمتری نیز مصرف شود. به عبارت دیگر می‌بایست هر دنباله از یک‌ها که به صورت $011\ldots110$ می‌باشد را با عبارت $100\ldots000$ تقریب زد. بدیهی است که چنین روشی مقداری خطا را به محاسبات اضافه خواهد کرد، اما در عوض به علت کمتر شدن تعداد اعدادی که در حاصل جمع‌های جزئی می‌بایست با هم جمع گردند، انرژی کمتری مصرف خواهد شد. در [۱۲] ادعا شده‌است که از چنین تقریبی برای کلیه‌ی دنباله‌های یک‌ها می‌توان استفاده کرد، اما شبیه‌سازی‌های انجام شده نشان می‌دهد که خطای به وجود آمده بسیار فراتر از مقدار پیش‌بینی شده خواهد شد. به عبارت دیگر پیش‌بینی شده‌است که حدود یک درصد خطا به طور متوسط ایجاد خواهد شد ولی این مقدار خطا تنها زمانی به دست خواهد آمد که، دنباله‌هایی را تقریب زد که حداقل طول آن‌ها برابر ۵ باشد. در نمودار شکل (۲) خطاهایی که با درنظر گرفتن گرد کردن با تعداد بیت‌های مختلف به دست آمده‌است را نمایش می‌دهد.

بخش دیگری که در [۱۲] در مورد آن ادعا شده‌بود، صرف انرژی کمتر در انجام محاسبات می‌باشد. مبنای اندازه‌گیری انرژی مصرف شده در این مقاله، سنجش میزان فعالیت مدار می‌باشد (بخش ۴). در شکل (۲) درصد کاهش میزان فعالیت مدار به ازای گرد کردن طول دنباله‌های مختلف یک‌ها نمایش داده‌شده‌است.

در [۱۲] ادعا شده‌است که حدود ۱۷ درصد در مصرف انرژی بهبود خواهیم داشت. نکته‌ی جالب در این اعداد این است که عدد ۱۷ درصد بهبود مصرف انرژی تنها درصورتی محقق خواهد شد که کلیه‌ی دنباله‌های یک بزرگتر یا مساوی با دو را گرد کنیم که در اینصورت دقت انجام محاسبات کاهش بسیاری خواهد داشت. اما در روش پیشنهادی علاوه بر اینکه سرعت محاسبات به نحو قابل ملاحظه‌ای افزایش خواهد یافت، مصرف توان نیز به نحو قابل توجهی کم خواهد شد. همچنین کاهشی نیز در دقت انجام محاسبات نخواهیم داشت.



شکل (۲): خطاهای مختلف ایجاد شده در حالت‌های گرد کردن دنباله‌ی یک‌های به طول‌های مختلف



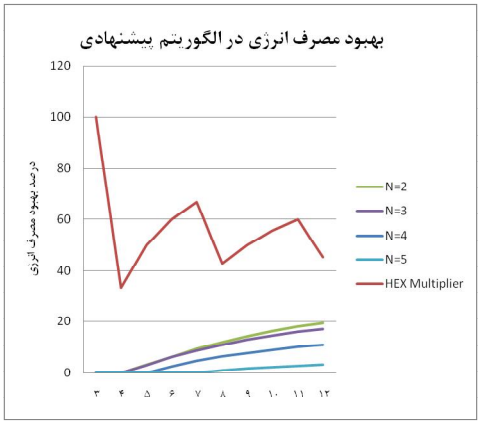
شکل (۳): بهبود مصرف انرژی در حالات مختلف

نتایج حاصل از الگوریتم پیشنهادی

از آنجائی که الگوریتم پیشنهادی، عمل ضرب را در مبناهای بزرگ محاسبه می‌کند، بنابراین تعداد حاصل‌جمع‌های جزئی ایجاد شده بسیار کمتر از حالت قبل می‌باشد. البته ایجاد جدول محاسبه‌ی

$x_i c_i$ در زمان نوشتن برنامه ایجاد می‌گردد و تنها یکبار صورت می‌پذیرد بنابراین محاسبه‌ی آن در زمان اجرا تأثیری نخواهد داشت، ولی خواندن آن از حافظه، سربارهایی را خواهد داشت.

در این روش جدید می‌بایست در حالت $r = 16$ به طور تئوری حدود چهار برابر افزایش سرعت نسبت به حالت استاندارد الگوریتم Horner داشته باشیم، اما با توجه به اینکه در حالت استاندارد نیز به طور متوسط تنها در نصف حالات حاصل‌ضرب جزئی ایجاد می‌گردد و همچنین در این روش نیز برای هر حاصل‌ضرب جزئی یکبار دسترسی به حافظه احتیاج داریم، افزایش سرعت کمتر از مقدار اسمی آن خواهد بود. در شکل (۵) مقایسه‌ای بین بهبود سرعت حاصل از روش پیشنهادی در مقایسه با [۱۲] نمایش داده شده‌است. حدود ۵۰ درصد بهبود در مصرف انرژی نسبت به حالت استاندارد و حدود ۳۳ درصد بهبود نسبت به [۱۲] قابل مشاهده می‌باشد.



شکل (۵): بهبود مصرف انرژی در الگوریتم پیشنهادی

البته از آنجائیکه بهبود سرعت و کاهش مصرف انرژی در این روش هر دو به یک پارامتر بستگی دارند، سرعت انجام محاسبات نیز تقریباً با همین نرخ بهبود خواهد یافت. این در حالی است که نسبت به روش Horner هیچ کاهش دقتی نیز وجود نخواهد داشت، ولی در [۱۲] کاهش دقت نیز وجود داشته‌است. علاوه بر این، در [۱۲] سرعت محاسبات و مصرف انرژی بستگی به ورودی‌های مسئله دارد و ممکن است به ازای ورودی‌های مختلف کارایی متفاوتی مشاهده شود، اما در روش پیشنهادی سرعت اجرا و مصرف انرژی مستقل از داده‌های ورودی و همواره ثابت می‌باشند.

خلاصه

در این مقاله یک الگوریتم ضرب نرم‌افزاری کم مصرف برای استفاده در میکروکنترلرهایی که در شبکه‌های WSN مورد استفاده قرار می‌گیرند، معرفی شده‌است. مزیت اصلی این روش ارایه کارایی ثابت به ازای کلیه‌ی ورودی‌های مختلف و ارایه بهترین دقت ممکن در انجام محاسبات

می‌باشد. میزان انرژی مصرفی دینامیکی نسبت به حالت پایه‌ی الگوریتم Horner حدود ۵۰ درصد بهبود داشته است. سرعت اجرای الگوریتم نیز تقریباً با همین نسبت بهتر شده‌است، زیرا توان مصرفی دینامیکی براساس فعالیت مدار و یا به عبارت بهتر تعداد عملیات جمع لازم برای انجام ضرب سنجیده شده‌است. تنها نکته‌ی منفی در این الگوریتم استفاده از حافظه‌ی بیشتر و تعداد دسترسی‌های مورد نیاز بیشتر به حافظه می‌باشد.

منابع

[1] F. Nakamura, A. F. Loureiro, and C. Frery "*Information Fusion for Wireless Sensor Networks: Methods, Models, and Classifications*" ACM Computing Surveys, Vol. 39, No. 3, Aug. 2007.

[2] B. Parhami, "*Computer Arithmetic: Algorithms and Hardware Designs*", 2nd edition, Oxford University Press, New York, 2010.

[3] Wireless Sensing Interest Group: *http://www.wisig.org/*

[4] WSN WIKI: OverSigma: *http://wsn.oversigma.com/wiki/index.php/main-page*

[5] Dain Tree Networks: *http://www.daintree.net/resowcos/index.php*

[6] Zig bee links: *http://www.zigbeelinks.com/*

[7] Micaz datasheet *http://www.xbow.com/Products/Product _ pdf _files/*

[8] J.Polastre, R. Szewczyk, and D. Culler, "*Telos: enabling ultra-low power wireless research," Information Processing in Sensor Networks*", 2005. IPSN 2005. Fourth International Symposium on , vol., no., pp. 364-369, 15 Apr. 2005.

[9] T. Nguyen and A. Chatterjee, "*Number-Splitting with Shift-and-Add Decomposition for Power and Hardware Optimization in Linear DSP Synthesis*", IEEE Transactions on very large scale integration (VLSI) systems, vol. 8, NO. 4, Aug. 2000.

[10] Lars Wanhammar, "*DSP Integrated Circuits"*, San Diego, CA, Academic Press, 1999.

[11] K. Venkat and M. Raju, "*Efficient Signal Conditioning for Microcontroller Based Medical Solutions*", The International Symposium on Consumer Electronics (ISCE 2007), pp. 20-23 Jun. 2007.

[12] Ahmed Abdelgawad, Sherine Abdelhak, Soumik Ghosh, Magdy Bayoumi, *A Low-Power Multiplication Algorithm for Signal Processing in Wireless Sensor Networks*, MWSCAS '09, 52nd IEEE International Midwest Symposium on Circuits and Systems, 2009.

[13] Andrew D. Booth. "*A signed binary multiplication technique*". The Quarterly Journal of Mechanics and Applied Mathematics, Volume IV, Pt. 2, 1951.

[14] Jan Rabaey, "*Low Power Design Essentials (Integrated Circuits and Systems)*", Springer, 2009.

[15] Donald Knuth, *The Art of Computer Programming*, Volume II, 3rd Ed., Addison-Wesley, pp. 486–488, sec. 4.6.4.