

## طراحی و پیاده‌سازی کارآمد STAP به روش چندبرداری برای آشکارسازی هدف در رادارهای هوایی

نرجس حسنی خواه<sup>۱</sup>، سیاوش امین‌نژاد<sup>۲\*</sup>، غفار درویش<sup>۳</sup>، محمدرضا منیری<sup>۴</sup>

۱- دانشجوی دکتری، دانشگاه آزاد اسلامی، واحد علوم و تحقیقات، گروه مهندسی برق- الکترونیک، تهران، ایران ۲- استادیار، دانشگاه گیلان،

۳- دانشیار، دانشگاه آزاد اسلامی، واحد علوم و تحقیقات، گروه مهندسی برق- الکترونیک، تهران، ایران

۴- استادیار، دانشگاه آزاد اسلامی، واحد یادگار امام خمینی (ره) شهر ری، دانشکده برق، تهران، ایران

(دریافت: ۹۶/۰۸/۳؛ پذیرش: ۹۷/۰۶/۲۱)

### چکیده

پردازش وقتی زمانی- مکانی (STAP) یک روش اساسی در بهبود عملکرد رادارهایی است که در حضور اختلالات شدید دینامیکی مانند کلاتر و جمینگ عمل می‌کنند. عملکرد STAP بر اساس نمونه‌برداری با نرخ بسیار بالا از سیگنال‌هایی است که به‌طور هم‌زمان از چندین آرایه آنتنی و تعدادی پالس دریافت شده‌اند. لذا، حجم محاسبات بسیار بالا بوده و پیاده‌سازی آن نیز دشوار است. در این مقاله، روش چندبرداری برای کاهش حجم محاسبات STAP ارائه گردیده است. نتایج پیاده‌سازی نشان می‌دهد که روش پیشنهادی علاوه بر کاهش حجم محاسبات منجر به کاهش منابع سخت‌افزاری، کاهش توان مصرفی و افزایش حداکثر فرکانس عملیاتی نیز می‌شود. به‌طور مثال، حجم محاسبات با ۶ آرایه آنتنی، ۱۰ پالس دریافتی و ۲۰۰ نمونه فاصله با اندازه بردار ۱۷، ۲۸٫۱ GFLOPs با حداکثر فرکانس ۲۷۸ MHz و زمان اجرای ۲۶۲  $\mu$ s روی تراشه Arria 10 به‌دست آمده است. بنابراین، روش چندبرداری می‌تواند نیازمندی‌های زمان- آنتنی وزن‌های STAP را برآورده سازد. الگوی وقتی زاویه- داپلر و تست استاتیکی برای آشکارسازی هدف نشان می‌دهند که استفاده از روش مذکور برای محاسبه وزن‌ها بسیار کاربردی است.

### واژگان کلیدی

پردازش وقتی زمانی- مکانی، جمینگ، رادار هوایی، کلاتر

#### ۱- مقدمه

هرچند که اصول STAP در اوایل ۱۹۷۰ بنا نهاده شد [۶]، ولی با توجه به حجم بسیار سنگین محاسبات و لزوم اجرای آن در بازه زمانی بسیار کوتاه، تلاش برای تحقق و پیاده‌سازی زمان-آنتنی<sup>۳</sup> آن به‌طور کامل موفقیت‌آمیز نبوده است. پیچیدگی محاسباتی STAP قابلیت‌های محاسباتی پردازنده‌ها را محدود ساخته است. یک پردازنده STAP مجموعه‌ای از وزن‌های وقتی را از حل یک مدل زمان- آنتنی که شامل معادلات خطی از مرتبه<sup>۳</sup> (NM) است، به‌دست می‌آورد. در یک شیوه کاملاً وقتی، هنگامی که رادار حجمی از فضا را اسکن نموده و در محدوده سرعت معمول هدف به دنبال آن می‌گردد، فاصله زمانی پردازش پیوسته معمولاً کمتر از یک ثانیه خواهد بود. همچنین، یک پردازنده STAP باید بتواند مسأله محاسبه وزن‌های وقتی را چندین بار شکل داده و در این فاصله زمانی کوتاه حل نماید. حاصل چنین شرایطی می‌تواند بازده محاسباتی از مرتبه صدها

پردازش وقتی زمانی- مکانی (STAP)<sup>۱</sup> یک پردازش سیگنال پیشرفته است که برای تشخیص وجود یک شیء در سطح زمین و تعیین بُرد آن به کار می‌رود. STAP که از ترکیب دو پردازش در حوزه‌های زمانی و مکانی حاصل می‌شود، سیگنال‌های دریافتی از چندین آنتن در یک آرایه مکانی و از چندین پالس دریافتی در حوزه زمان را به شکل وقتی با هم ترکیب می‌نماید [۱-۳]. همچنین به شیوه‌های متفاوتی می‌تواند عملکرد رادارهای هوایی را بهبود بخشد. از جمله می‌توان به تشخیص اهدافی با سرعت کم به وسیله حذف بهتر لب<sup>۲</sup> اصلی کلاتر، تشخیص اهدافی با سطح مقطع کوچک که ممکن است توسط لب جانبی کلاتر پوشانده شوند و تشخیص اهداف در محیط‌های حاوی کلاتر و جمینگ، اشاره نمود [۴-۵].

\* رایانامه نویسنده پاسخگو: amin@liv.ac.uk

3- Real-time

1- Space time adaptive processing (STAP)

2- Lobe

کارآمدی آن برای اندازه های ماتریسی بزرگ از جمله معایب تحقیق مذکور به شمار می رود. در مطالعه دیگری، طراحی مبتنی بر FPGA الگوریتم  $D^3$  و یک بسته نرم افزاری جدید برای STAP ارائه شد [۱۶]. نتایج آنها با تست چند اندازه ماتریسی کوچک نشان داد که حداکثر فرکانس عملیاتی ( $F_{max}$ ) بسیار پایین و مصرف منابع سخت افزاری بسیار بالا است. مرجع [۱۷] به پیاده سازی STAP روی FPGA برای آشکارسازی هدف در رادارهای دوپایه و غیرفعال اختصاص یافته است. این تحقیق بر طراحی و پیاده سازی فیلترینگ تطبیق یافته تمرکز داشت و به حجم محاسبات مورد نیاز در پیاده سازی STAP اشاره ای نشد. در مرجع [۱۸] نیز به پیاده سازی STAP روی GPU<sup>۸</sup> های مختلف پرداخته شد؛ ولی در نهایت توان مصرفی به ۷۵ W و زمان محاسبات به ۳ ms کاهش یافت. بنابراین، پیاده سازی STAP بر روی پردازشگرهای قابل برنامه ریزی هنوز به عنوان یک مسأله چالش برانگیز باقی مانده است و در این بخش نیاز به فعالیت و پژوهش بیشتری محسوس است. لذا در این مقاله، به طراحی و پیاده سازی مدلی کارآمد برای STAP پرداخته ایم. ابتدا، بلوک دیگرامی برای محاسبه بردار وزن بدون نیاز به محاسبه صریح ماتریس کواریانس ارائه گردید. سپس با انجام اصلاحاتی در دستورات QRD-MGS توانستیم ضمن خارج نمودن مسیر کامل محاسبه Q، سرعت الگوریتم را به طور نسبی افزایش داده و طراحی QRD را ساده تر نماییم. ما برای برقراری مصالحه بین زمان اجرای محاسبات و سایر پارامترها مانند توان مصرفی، مصرف منابع سخت افزاری و بار محاسباتی، بلوک QRD را به صورت انعطاف پذیر با در نظر گرفتن پارامتری به نام "اندازه بردار" طراحی نمودیم. از آنجا که پردازش هر ماتریس به صورت ستونی (برداری) انجام می شود؛ با انتخاب اندازه بردار دلخواه، هر ستون به قسمت های کوچکتری تقسیم شده و پردازش هر یک از بخش ها به ترتیب انجام می شود. ما روش پردازش چندبرداری را برای پیاده سازی کارآمد ماتریس های با ابعاد بزرگ در نظر گرفتیم که در حوزه STAP تاکنون چنین روشی استفاده نشده است. برای سنجش صحت بردار وزن نیز از الگوی وقفی زاویه- داپلر و تست استاتیکی برای تشخیص هدف استفاده می شود.

ادامه مقاله به صورت زیر سازمان یافته است. در بخش بعدی مدل سامانه شرح داده می شود. سپس روش ارائه شده برای طراحی و پیاده سازی STAP و ساختارهای پیشنهادی آن معرفی خواهند شد. نتایج شبیه سازی و پیاده سازی به ترتیب در بخش های چهارم و پنجم ارائه می شوند. بخش ششم نیز به نتیجه گیری اختصاص دارد.

میلیارد عملیات ممیز شناور در ثانیه باشد. بنابراین، پیاده سازی کارآمد STAP با وجود جریان داده بزرگ یکی از چالش های عمده در این حوزه بوده و پردازش در واحد اندازه، توان و وزن (SWaP)<sup>۱</sup> نیز اهمیت خاصی دارد [۷-۹]. در سکوی رادارهای هواپرد برای حداقل نمودن SWaP به الگوریتم های پردازش سیگنال با پیچیدگی محاسباتی کم، نیاز است. از آنجا که الگوریتم پیشرفته پردازش سیگنال STAP به شدت محاسباتی است؛ نیازهای آموزشی و بار محاسباتی آن با قید SWaP پایین متناقض می باشد. بنابراین، STAP چالشی را برای طراحان پردازش سیگنال مطرح نموده؛ به طوریکه می توان گفت هدف اصلی بهبود سامانه های STAP به افزایش سرعت مورد نیاز آنها برای عملکرد کارآمدتر در شرایط پویا و مختلط وابسته است. همچنین اکثر پردازنده های دیجیتال از طول کلمه ۱۶ بیتی و فرمت ممیز ثابت بهره می گیرند که برای STAP مناسب نمی باشد. گزینه دیگر استفاده از پردازنده های ممیز شناور است که محدوده دینامیکی وسیعی را فراهم می نمایند. بنابراین، پیاده سازی ممیز شناور STAP از مهمترین حوزه های تحقیقاتی روز به شمار می رود.

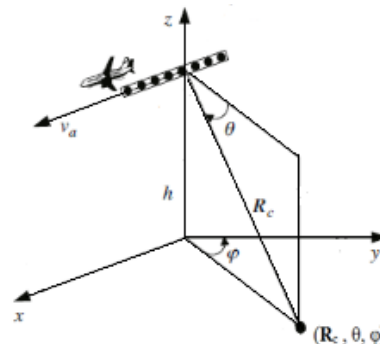
در گذشته، ابتدا از کامپیوترهای با کارایی بالا (HPC)<sup>۲</sup> ها برای پیاده سازی STAP استفاده می شد [۱۱-۱۰]. با عرضه تراشه های DSP<sup>۳</sup>، از آنها برای پیاده سازی زمان-آنی STAP استفاده گردید [۱۳-۱۲]. با توسعه و گسترش کارایی STAP در حوزه های مختلف، انگیزه های بیشتری برای پیاده سازی آن ایجاد شد. در مرجع [۱۴]، چهار شیوه پیاده سازی تجزیه QR (QRD) برای کاربرد STAP شامل الگوریتم گرام-اشمیت کلاسیک (CGS)<sup>۴</sup>، الگوریتم گرام-اشمیت اصلاح شده (MGS)<sup>۵</sup>، دوران گیونز (GR)<sup>۶</sup> و تبدیل هاوس هولدر (HT)<sup>۷</sup> بررسی شدند و نتایج حاصل از پیاده سازی بر روی HPC و DSP مقایسه گردید. نتایج نشان داد که الگوریتم MGS از سرعت بالاتری نسبت به سایر روش ها برخوردار است. همچنین سرعت پیاده سازی DSP ها بالاتر از HPC ها است. در مرجع [۱۵] نیز، طراحی و پیاده سازی زمان-آنی بخش محاسبات QRD با استفاده از الگوریتم MGS روی FPGA انجام گرفت. هر چند که طراحی ارائه شده از مصرف منابع سخت افزاری مناسبی برخوردار بود، ولی استفاده صرف از یک اندازه ماتریسی کوچک و عدم تست انعطاف پذیری و

- 1- Size, weight and power (SWaP)
- 2- High performance computer (HPC)
- 3- Digital signal processing (DSP)
- 4- Classical Gram-schmidt (CGS)
- 5- Modified gram-schmidt (MGS)
- 6- Givens rotation (GR)
- 7- Householder transformation (HT)

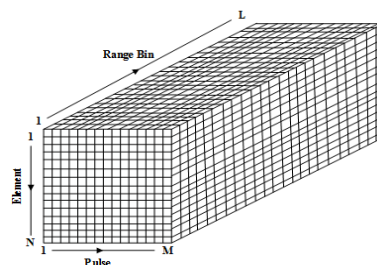
8- Graphics processor unit (GPU)

## ۲- مدل سامانه

مدل سامانه مطابق شکل (۱) در نظر گرفته شده است. در این شکل، یک رادار پهلونگر<sup>۱</sup> پالسی و تک پایه با آنتنی با چند مرکز فاز مانند آنتن آرایه<sup>۲</sup> فازی در نظر گرفته شده که بر روی یک هواپرد قرار دارد. هواپرد در امتداد محور آنتن که متشکل از یک آرایه<sup>۳</sup> خطی یکنواخت است؛ در ارتفاع  $h$  از سطح زمین و با سرعت  $v_a$  در حال حرکت است. مختصات هدف با  $(R_c, \theta, \varphi)$  نشان داده شده است. برد هدف و  $(\theta, \varphi)$  نیز موقعیت مکانی هدف را مشخص می‌کنند. سیگنال دریافتی مطابق شکل (۲) به‌صورت یک مکعب داده است که  $N$  تعداد عناصر آرایه<sup>۴</sup> آنتنی و  $M$  تعداد پالس‌های دریافتی است. در فاصله هر پالس، سیگنال‌های دریافتی با نرخ بالایی نمونه‌برداری می‌شوند که نمونه‌های فاصله نامیده شده و با  $L$  نشان داده شده‌اند. سیگنال دریافتی خالص نبوده و با سیگنال‌های ناخواسته‌ای همچون نویز و تداخل‌هایی مانند کلاتر و جمینگ همراه است. STAP عمل فیلترینگ برداری را بر روی ترکیب داده‌های عناصر آرایه<sup>۵</sup> آنتنی و پالس‌های مجاور برای یک نمونه فاصله خاص انجام می‌دهد. این روند برای تمامی نمونه‌های فاصله صورت می‌گیرد.



شکل (۱): مدل سامانه



شکل (۲): مکعب داده STAP

با محاسبه وزن‌های وقتی می‌توان سیگنال مطلوب را با کیفیت مناسب از مجموع سیگنال‌های دریافتی تفکیک نمود. در STAP وزن‌های بهینه از رابطه (۱) محاسبه می‌شوند.  $t$  بردار

فرمان<sup>۶</sup> مکان-زمان،  $S_I$  ماتریس کواریانس تداخل و  $k$  ضریب نرمالیزه است.

$$w = k S_I^{-1} t^* \quad (۱)$$

$t$  مطابق رابطه (۲) از ضرب کرونیگر<sup>۳</sup> بردارهای فرمان مکانی و زمانی حاصل می‌شود که به ترتیب در روابط (۳) و (۴) نشان داده شده است.

$$t = \alpha(\theta, \varphi) \otimes \alpha(f_s) \quad (۲)$$

$$\alpha(\theta, \varphi) = \left[ 1, \dots, e^{j 2\pi(N-1) \frac{d}{\lambda} \sin\theta \cos\varphi} \right]^T \quad (۳)$$

$$\alpha(f_s) = \left[ 1, \dots, e^{j 2\pi(M-1) f_s / f_r} \right]^T \quad (۴)$$

$\lambda$  طول موج،  $d$  فاصله بین آرایه‌های آنتنی مجاور،  $f_s$  فرکانس داپلر و  $f_r$  فرکانس تکرار پالس است.

برای داشتن بهره<sup>۷</sup> واحد در جهت سیگنال مطلوب، می‌توان از قید  $w^H t = 1$  استفاده نمود که به آن شکل‌دهنده<sup>۸</sup> پرتو بدون اعوجاج اطلاق می‌شود. با اعمال این قید به رابطه (۲)، ضریب  $k$  مطابق رابطه (۵) تعیین می‌شود [۱۹].

$$k = \frac{1}{t^H (S_I^{-1} t^*)} \quad (۵)$$

با قراردادن رابطه (۵) در رابطه (۱)، رابطه (۶) برای بردار وزن به‌دست می‌آید.

$$w = k S_I^{-1} t^* = \frac{S_I^{-1} t^*}{t^H (S_I^{-1} t^*)} = \frac{u}{t^H u^*} \quad (۶)$$

در رابطه فوق،  $u \equiv S_I^{-1} t^*$  است. محاسبه مستقیم  $w$  با به‌دست آوردن معکوس  $S_I$  برای مسائلی همچون شکل‌دهنده‌های پرتو وقتی با تعداد متوسط آنتن‌ها امکان‌پذیر است. اما در STAP به دلیل ابعاد بالای ماتریس  $S_I$ ، محاسبه معکوس آن از لحاظ عددی حساس و محاسباتش بسیار پیچیده می‌باشد. لذا، به‌ندرت این کار صورت می‌گیرد. از آنجا که در رابطه (۶) نتیجه نهایی  $w$  است،  $S_I$  یک فاکتور میانجی محسوب می‌شود. از طرف دیگر، حل معادله (۱) معادل با یافتن پاسخ برای مجموعه معادلات خطی  $w S_I = k t^*$  است. بنابراین، با توجه به حجم بسیار سنگین محاسبات ناشی از محاسبه معکوس  $S_I$  در رابطه (۶)، می‌توان رابطه  $u \equiv S_I^{-1} t^*$  را به  $u \equiv t^*$  تبدیل نمود؛ سپس از روش‌های حل معادلات خطی برای تعیین  $u$  استفاده کرد. ولی معمولاً

2- Steering  
3- Kronecker product

1- Side looking

فکتورگیری LU<sup>۴</sup>، تجزیه چالاسکی<sup>۵</sup> و QRD متداول هستند که همه آنها بعد از تعدادی عملیات مشخص، راه حل دقیقی را برای معادلات ارائه می دهند. ایده اصلی روش های ذکر شده، کاهش تعداد معادلات کلی به یک فرم بالامتثلی و یا پایین مثلثی است. سپس با استفاده از روش های جایگزینی پسر و پیشرو، نتیجه نهایی با دقت مطلوبی به دست می آید. در این میان، روش QRD کاربرد گسترده تری دارد. QRD به دو شیوه می تواند برای یافتن بردار وزن در الگوریتم SMI استفاده شود که عبارت است از: حوزه توان و حوزه ولتاژ [۱۹]. روش حوزه توان با تشکیل ماتریس کواریانس  $\hat{S}_I$  آغاز می شود. در این روش QRD مستقیماً به ماتریس  $\hat{S}_I$  اعمال می شود. از آنجا که رابطه (۷) شامل ضرب مقادیر دامنه ولتاژ است، مقادیر ماتریس کواریانس در دامنه توان می باشند. از معایب روش حوزه توان، اثر مجذور نمودن مقادیر ناشی از ضرب خارجی است. در بسیاری از سامانه های واقعی، نرخ واقعی محاسبات لازم برای پردازش واقعی بسیار بالا است و سخت افزارهای مورد استفاده معمولاً ممیز ثابت می باشند. مجذور کردن یک سیگنال، دامنه دینامیکی آن را بر حسب دسیبل دو برابر نموده و بنابراین تعداد بیت های لازم برای نمایش بدون سرریز آن را نیز دو برابر می نماید. در نتیجه با افزایش نرخ داده در پردازشگر، ابعاد آن و همچنین پیچیدگی طراحی افزایش می یابد. لذا، مطلوب است که سیگنال بی جهت مجذور نشود. در حوزه ولتاژ، QRD مستقیماً به بردارهای داده مکان-زمان اعمال و از محاسبه صریح ماتریس کواریانس  $\hat{S}_I$  اجتناب می گردد. در این روش ابتدا یک ماتریس  $Y$  در ابعاد  $NM \times L_s$  از به هم چسباندن بردارهای داده مکان-زمان مطابق رابطه (۹) تشکیل می شود.

$$Y = [y_{00}, y_{01}, \dots, y_{L_s-1}] \quad (9)$$

از ماتریس  $Y$  برای محاسبه ماتریس  $\hat{S}_I$  نیز می توان بهره گرفت که در رابطه (۱۰) آمده است. ولی از آنجا که مجذور عناصر را شامل می شود، محاسبه آن مطلوب نیست.

$$\hat{S}_I = Y^* Y' \quad (10)$$

سپس، QRD به  $Y'$  اعمال می شود. ابعاد ماتریس های  $Q$  و  $R$  که از تجزیه  $Y'$  به دست می آیند، به ترتیب  $L_s \times L_s$  و  $L_s \times NM$  است.  $Q$  ماتریس متعامد و  $R$  با رابطه (۱۱) بیان می شود.  $\hat{R}$  یک ماتریس بالامتثلی در ابعاد  $NM \times NM$  و  $0$ ، ماتریسی با درایه های صفر در ابعاد  $(L_s - NM) \times NM$  است.

$$R = \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix} \quad (11)$$

با قراردادن ماتریس های  $Q$  و  $R$  در رابطه (۱۰)، رابطه (۱۲)

ماتریس کواریانس از پیش مشخص نیست و با توجه به داده های دریافتی تعیین می شود. یکی از روش های بسیار متداول برای محاسبه  $\hat{S}_I$  الگوریتم شکل دهنده پرتو SMI<sup>۱</sup> است [۱۹]. در این الگوریتم، ماتریس کواریانس با انتخاب تعدادی از نمونه های فاصله که سلول های آموزشی ( $L_s$ ) نامیده می شوند، به دست می آید.  $L_s$  نباید شامل سلول هایی باشد که اکوهای هدف را در بر می گیرد. برای جلوگیری از آلودگی ماتریس کواریانس با اکوهای هدف، تعدادی سلول محافظ قبل و بعد از سلول هدف در نظر گرفته می شوند. به این ترتیب تخمینی از  $\hat{S}_I$  به دست می آید که در رابطه (۷) نشان داده شده است.

$$\hat{S}_I = E \{ Y Y^H \} = \frac{1}{L_s} \sum_{i=1}^{L_s} y(i) y^H(i) \quad (7)$$

در رابطه فوق، ابعاد ماتریس  $Y$ ،  $NM \times L_s$  است که  $NM < L_s < L$  می باشد و از بردار  $y(i)$  تشکیل شده است.  $y(i)$  ها تصاویر لحظه ای<sup>۲</sup> از مکعب داده هستند. برای ایجاد  $y(i)$ ، ابتدا یک برش دو بعدی از مکعب داده در نمونه فاصله مورد نظر ( $l_0$ ) که یک تصویر لحظه ای دو بعدی در ابعاد  $N \times M$  است، صورت می گیرد. سپس، تصویر لحظه ای دو بعدی مطابق رابطه (۸) به یک بردار ستونی  $y_{l_0}$  در ابعاد  $NM \times 1$  تبدیل می شود. ماتریس کواریانس برای هر یک از  $L_s$  نمونه فاصله به طور مجزا محاسبه شده و سپس با متوسط گیری، ماتریس  $\hat{S}_I$  به دست می آید. پس از تعیین  $\hat{S}_I$  از رابطه (۶) برای محاسبه  $w$  استفاده می شود.

$$y_{l_0} = \begin{bmatrix} y(l_0, 0, 0) \\ y(l_0, 0, 1) \\ \vdots \\ y(l_0, 0, N-1) \\ y(l_0, 1, 0) \\ \vdots \\ y(l_0, 1, N-1) \\ \vdots \\ y(l_0, M-1, 0) \\ \vdots \\ y(l_0, M-1, N-1) \end{bmatrix} \quad (8)$$

برای پرهیز از محاسبه معکوس  $\hat{S}_I$  می توان از روش های حل معادلات خطی بهره گرفت. بخش وسیعی از کاربردهای پردازش سیگنال و مخابرات به مسائل حداقل مربعات (LS) مرتبط بوده و به حل آنها نیازمندند. مسأله LS با حل معادلات خطی  $Ax=b$  شناخته می شود. روش های متفاوتی برای حل این معادلات وجود دارد که مهمترین آنها شیوه حل مستقیم است [۲۰، ۲۱]. روش های مستقیم به طرق مختلفی مانند حذف گوسی،

1- Sample Matrix inversion (SMI)

2- Snapshots

3- Least squares (LS)

4- Lower upper (LU) factorization  
5- Cholesky decomposition

به‌دست می‌آید.

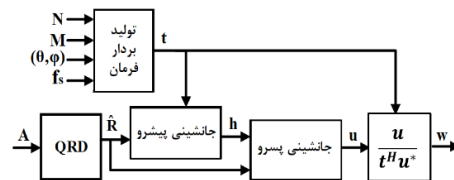
$$\hat{S}_I = Y^* Y' = R^H Q^H Q R = R^H R = \hat{R}^H \hat{R} \quad (12)$$

اگر رابطه فوق را در  $\hat{S}_I u = t^*$  جایگذاری نماییم، رابطه (۱۳) حاصل می‌شود که با فرض  $h \equiv \hat{R} u$  به رابطه (۱۴) می‌رسیم.

$$\hat{R}^H \hat{R} u = t^* \quad (13)$$

$$\hat{R}^H h = t^* \quad (14)$$

در رابطه (۱۴)،  $\hat{R}^H$  یک ماتریس پایین مثلثی است. بنابراین، بردار  $h$  به طول  $NM$  با استفاده از جانشینی پیشرو به‌دست می‌آید. سپس از رابطه  $h \equiv \hat{R} u$  بردار  $u$  با استفاده از جانشینی پسرو محاسبه می‌شود. نهایتاً بردار وزن  $w$  از رابطه (۶) به‌دست خواهد آمد. در شکل (۳)، بلوک دیاگرام روش فوق‌الذکر برای پیاده‌سازی STAP به‌منظور تعیین بردار وزن نشان داده شده است.  $A$  همان ماتریس  $Y'$  است.



شکل (۳): بلوک دیاگرام مدل STAP برای محاسبه بردار وزن

### ۳- طراحی مدل STAP

اولین مرحله بسیار مهم پس از محاسبه ماتریس داده  $Y$ ، تجزیه QR است. تحقیقات و پژوهش‌ها نشان می‌دهند؛ از میان روش‌های ذکر شده برای QRD در حوزه STAP، GR و HT از کارایی کمتری برخوردارند [۱۴]. MGS تغییریافته CGS است که از لحاظ هزینه پیاده‌سازی تقریباً مشابه آن می‌باشد، ولی پایداری عددی بهتری را نشان می‌دهد [۲۱]. دستورات استاندارد الگوریتم MGS همراه با تعداد عملیات موردنیاز در جدول (۱) آمده است. درایه‌های ماتریس  $\hat{R}$ ، سطر به سطر محاسبه می‌شوند و در هر سطر ابتدا عنصر قطری و سپس سایر عناصر محاسبه خواهند شد. محاسبه عناصر ماتریس  $Q$  نیز به‌صورت ستونی انجام می‌شود. به‌عبارت دیگر، پس از تشکیل سطر اول ماتریس  $\hat{R}$ ، ستون اول ماتریس  $Q$  تشکیل می‌شود. سپس ستون‌های باقیمانده ماتریس  $A$  که در مرحله بعد مورد نیازند، محاسبه خواهند شد. بنابراین، در هر چرخه ابتدا یک سطر از  $\hat{R}$ ، سپس یک ستون از  $Q$  و نهایتاً ستون‌های باقیمانده‌ای از  $A$  که در چرخه‌های بعد موردنیاز هستند، محاسبه می‌شوند. این روند به تعداد ستون‌های  $A$  ادامه می‌یابد تا در نهایت ماتریس‌های خروجی  $\hat{R}$  و  $Q$  تشکیل شوند. همان‌طور که در دستورات استاندارد MGS آمده است، برای محاسبه عناصر قطری  $\hat{R}$  از تابع نرم استفاده می‌شود. از آنجا

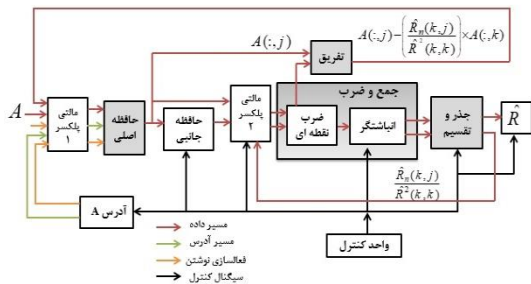
که مقصود اصلی در این مقاله، سازگاری طراحی از لحاظ پیاده‌سازی سخت‌افزاری می‌باشد؛ سطر دوم را تغییر می‌دهیم تا با اجزای سخت‌افزاری موجود، آسان‌تر پیاده‌سازی شود. همچنین به‌محاسبه  $\hat{R}(k,k)$  قبل از  $\hat{R}(k,j)$ ، و محاسبه  $\hat{R}(k,j)$  و  $Q(1:L_s,k)$  پیش از  $A(1:L_s,j)$  نیاز است. این روند تعدادی زیادی سیکل غیرفعال ایجاد می‌نماید که مانع بهره‌برداری مناسب از منابع سخت‌افزاری می‌شود و پیاده‌سازی ممیز شناور آن را نیز دشوار می‌سازد. بنابراین، تغییراتی مطابق جدول (۲) در دستورات استاندارد MGS ایجاد می‌نماییم.

دستورات جدول (۲) از دو حلقه اصلی تشکیل شده است. در حلقه اول، ملزومات محاسباتی اولیه برای عناصر قطری ماتریس  $\hat{R}$  و ستون‌های باقیمانده از ماتریس  $A$  که در چرخه بعد موردنیاز هستند، انجام می‌شوند. بنابراین مطابق بلوک دیاگرام شکل (۴) که بر اساس جدول (۲) رسم شده است، عملگر جذر از مسیر بحرانی محاسبه  $A$  خارج شده است. در حلقه دوم نیز، محاسبه عناصر قطری و سایر عناصر غیرصفر برای هر سطر از ماتریس  $\hat{R}$  به پایان رسیده و ستون  $Q(1:L_s,k)$  محاسبه می‌شود. ترتیب مراحل مذکور از مزایایی نیز برخوردار است. از جمله این‌که؛ محاسبه ستون‌های باقیمانده ماتریس  $A$  به عملیات کمتری نیاز دارد. با توجه به این‌که ملزومات محاسباتی برای عناصر هر سطر از ماتریس  $\hat{R}$  در حلقه اول لحاظ شده است؛ حلقه دوم می‌تواند تا در دسترس قرار گرفتن داده‌های موردنیاز، زمان‌بندی شود. همچنین در شکل (۳) مشاهده می‌شود که نیازی به محاسبه  $Q$  برای تعیین بردار وزن وجود ندارد. بنابراین، همان‌طور که در شکل (۴) نشان داده شده، محاسبه آن به‌طور کامل از مسیر بحرانی خارج شده است. به این ترتیب، می‌توانیم به طراحی QRD ساده‌تری دست یابیم. از جدول‌های (۱ و ۲) مشهود است که مزایای ذکر شده منجر به افزایش تعداد عملگرها نشده است.

جدول (۱): دستورات استاندارد الگوریتم QRD-MGS

شماره	دستور	جذر	تقسیم	ضرب
۱	for k=1:NM			
۲	$\hat{R}(k,k) = \text{norm}(A(1:L_s, k));$	k		$k \times L_s$
۳	for j = k+1:NM			
۴	$\hat{R}(k,j) = \text{dot}(A(1:L_s, k), A(1:L_s, j)) / \hat{R}(k,k);$	$k^2/2$	$k^2/2$	$L_s \times k^2/2$
۵	end			
۶	$Q(1:L_s, k) = A(1:L_s, k) / \hat{R}(k,k);$		k	
۷	for j = k+1:NM			
۸	$A(1:L_s, j) = A(1:L_s, j) - \hat{R}(k,j) \times Q(1:L_s, k);$			$L_s \times k^2/2$
۹	end			
۱۰	end			
مجموع		k	$k+k^2/2$	$(k+k^2)L_s$

است که از چهار حالت تشکیل شده است. حالت‌ها عبارتند از: محاسبه اندازه مقادیر، محاسبه ضرب‌های نقطه‌ای، موجود بودن اندازه‌ها، محاسبه مقادیر جدید ماتریس  $A$  مالتی پلکسر ۱، منبع داده‌ای را که به طبقه بعد هدایت می‌شود را انتخاب می‌کند. داده‌ها می‌توانند از خارج از محیط طراحی به‌عنوان ورودی اولیه و یا از داخل آن باشند. در این بلوک بردارهایی به‌تعداد ستون‌های ماتریس ورودی، برای ذخیره داده‌ها ایجاد شده است. بنابراین، تمامی عملیات کاملاً برداری صورت می‌گیرد. در این طراحی تعداد ضرب‌کننده‌ها با پارامتری به‌نام اندازه بردار مشخص شده‌اند که بیانگر تعداد ضرب‌کننده‌های مختلط و ممیز شناوری است که می‌توانند به‌طور موازی عمل نمایند. حافظه اصلی از یک حافظه دومنظوره و یک مولد آدرس تشکیل شده است. در ابتدا حافظه دومنظوره با ماتریس ورودی  $A$  مقداردهی می‌شود. سپس در طی اجرای الگوریتم با داده‌های میانی که در گذرهای آتی استفاده می‌شوند، مقداردهی خواهد شد.

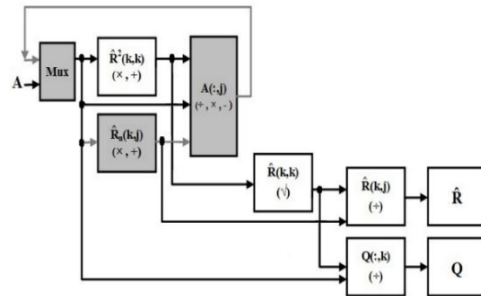


شکل (۵): طراحی الگوریتم اصلاح شده QRD-MGS

یک پیاده‌سازی کارآمد به استفاده مؤثر از حافظه بستگی دارد. در QRD نیاز است به‌طور مستمر از یک حافظه بزرگ خوانده شده و در آن نوشته شود. به‌طوری‌که بتواند کل ماتریس را ذخیره نماید. بنابراین، از آنجا که مقادیر جدید ماتریس  $A$  و مقادیر  $\hat{R}_n$  به بردار  $A(1:m,k)$  و سایر بردارهای ماتریس  $A$  نیاز دارند، مطلوب است که دو پورت مجزای خواندن برای محاسبه آنها در نظر گرفته شود. به جای استفاده از پورت مجزای خواندن برای حافظه بزرگ  $A$ ، یک حافظه جانبی که تنها اولین بردار را ذخیره می‌کند، به‌کار گرفته شده است. حافظه جانبی از یک حافظه دومنظوره و شمارنده‌هایی برای تأمین پورت آدرس تشکیل شده است. این حافظه، هنگام محاسبه  $\hat{R}(k,k)$  فراخوانده می‌شود. سپس  $A(1:m,k)$  برای استفاده مجدد در سایر بخش‌ها، ذخیره خواهد شد. مالتی پلکسر ۲، مسیر داده را به ضرب‌کننده هدایت می‌کند. واحد ضرب و جمع از ضرب نقطه‌ای و انباشتگر تشکیل شده است. محاسبه  $\hat{R}^2(k,k)$  و به‌هنگام نمودن‌های مکرر ستون‌های باقی‌مانده ماتریس  $A$  در این بخش انجام می‌شود. از آنجا که این واحد روی بردارها عمل می‌کند، پیاده‌سازی تمام موازی آن نیز امکان‌پذیر است. همان‌طور که مطرح شد، در این

جدول (۲): دستورات اصلاح شده الگوریتم QRD-MGS

شماره	دستور	جذر	تقسیم	ضرب
۱	for k=1:NM			
۲	$\hat{R}^2(k,k) = \text{dot}(A(1:L_s, k), A(1:L_s, k));$			$k \times L_s$
۳	$\hat{R}(k,k) = \text{sqrt}(\hat{R}^2(k,k));$	k		
۴	for j = k+1:NM			
۵	$\hat{R}_n(k,j) = \text{dot}(A(1:L_s, k), A(1:L_s, j));$			$L_s \times k^2/2$
۶	$A(1:L_s, j) = A(1:L_s, j) - (\hat{R}_n(k,j)/\hat{R}^2(k,k)) \times A(1:L_s, k);$		$k^2/2$	$L_s \times k^2/2$
۷	$\hat{R}(k,j) = \hat{R}_n(k,j)/\hat{R}(k,k);$		$k^2/2$	
۸	end			
۹	$Q(1:L_s, k) = A(1:L_s, k) / \hat{R}(k,k);$			k
۱۰	end			
مجموع		k	$k+k^2$	$(k+k^2)L_s$



شکل (۴): بلوک دیگرام دستورات اصلاح شده الگوریتم QRD-MGS

پیاده‌سازی سامانه‌ها به‌صورت دیجیتال با روش‌های مختلفی قابل اجرا است. از روش‌های جدیدی که با هدف پیاده‌سازی روی FPGA استفاده می‌شود، ابزار DSP Builder است [۲۲]. این بسته نرم‌افزاری به برنامه‌متلب الحاق گردیده و کتابخانه‌هایی را به قسمت سیمولینک اضافه می‌نماید. با برخورداری از این ابزار می‌توان سامانه‌های بزرگ و پیچیده را طوری طراحی نمود که پس از تست نتایج اولیه در متلب با لینک‌شدن به نرم‌افزار کوارتس<sup>۱</sup>، مستقیماً پروژه HDL ایجاد شده را مورد ارزیابی قرار داد. همچنین با لینک شدن به نرم‌افزار مدل‌سیم<sup>۲</sup>، پروژه HDL مورد شبیه‌سازی سخت‌افزاری قرار می‌گیرد. این روند، فرصت‌های جدیدی را برای طراحی در اختیار طراحان قرار می‌دهد و آن را از پیاده‌سازی‌های مبتنی بر قواعد ملالت‌آور رها می‌سازد. این‌گونه طراحی‌ها، به سازگاری یک طراحی بحرانی ناشی از آزمایشی بودن بُرد نیز کمک می‌نماید. در این مقاله، از ابزارهای مذکور برای طراحی و پیاده‌سازی استفاده شده است.

طراحی بلوک QRD در شکل (۵) نشان داده شده است. طراحی QRD از پنج بخش اصلی مشتمل بر واحد کنترل، حافظه اصلی، حافظه جانبی، واحد جمع و ضرب، و واحد جذر و تقسیم تشکیل شده است. واحد کنترل متشکل از یک ماشین حالت

3- Accumulator

1- Quartus  
2- Modelsim

#### ۴- نتایج شبیه‌سازی

همانطور که در قسمت دوم ذکر شد، با طراحی یک رادار پهلونگر پالسی تک‌پایه و استفاده از پردازش STAP برای تعیین موقعیت هدف و تضعیف تداخل‌ها، مکعب داده‌ای مطابق شکل (۲) به‌دست آمد. مشخصات کامل داده‌ها، در جدول (۳) آمده است. با خارج نمودن سلول هدف و سلول‌های محافظ اطراف آن، ۱۸۹ سلول بُرد باقی می‌ماند که از این تعداد، ۱۷۰ سلول برای آموزش در نظر گرفته شد. بنابراین، ماتریس A در ابعاد  $170 \times 60$  به‌دست آمد.

جدول(۳): مشخصات مکعب داده

تعداد آرایه‌های آنتنی	۶
تعداد پالس‌ها	۱۰
تعداد سلول‌های بُرد	۲۰۰
تعداد سلول‌های آموزشی	۱۷۰
تعداد سلول‌های محافظ	۱۰
مختصات مکانی هدف	( $45^\circ, -25^\circ$ )
بُرد هدف	۱۷۳۲ m
فرکانس حامل	۱۰ GHz
طول موج	۳ cm
فرکانس تکرار پالس	۳۰ KHz
فرکانس داپلر	۶۳۴۳ Hz
سرعت هواپرد	۲۲۵ m/s
ارتفاع هواپرد	۱۰۰۰ m
نسبت کلانر به نویز	۶۵ dB
نسبت سیگنال به نویز در گیرنده	۵ dB
زاویهٔ جمر ۱	$40^\circ$
زاویهٔ جمر ۲	$60^\circ$
توان جمرها	۱۰۰ W
حداکثر بُرد	۵۰۰۰ m

از آنجا که، یکی از مهمترین وظایف بخش پردازش یک گیرندهٔ راداری، تضعیف سیگنال‌های مزاحمی همچون کلانر و جمر است [۲۳، ۲۴]؛ از الگوی وقتی زاویه- داپلر استفاده می‌نماییم. شکل (۷) الگوی وقتی زاویه- داپلر را نشان می‌دهد که بر اساس وزن‌های به‌دست آمده، ترسیم شده است. مطابق جدول (۳)، دو سیگنال جمینگ با توان یکسان در زوایای مختلف وجود دارد. در شکل (۷)، توانی حدود  $130 \text{ dB}$ - در زوایای  $40^\circ$  و  $60^\circ$  مشهود است که بیانگر تضعیف جمرها است. خط انحنادار که در طول محور داپلر گسترده شده، نشان‌دهندهٔ کلانر است که با قدرتی حدود  $80 \text{ dB}$  تا  $100 \text{ dB}$ - تضعیف شده است. طبق جدول (۳)، زاویهٔ هدف  $45^\circ$  آزیموت<sup>۱</sup> و فرکانس داپلر نرمالیزه

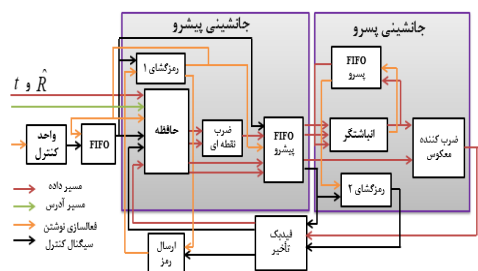
طراحی با در نظر گرفتن پارامتری به‌نام اندازهٔ بردار، پردازش در چندین مرحله انجام می‌شود. لذا، بردارها در بلوک انباشتگر جمع‌آوری می‌شوند. با تغییر اندازهٔ بردار و یا تعداد ستون‌های ماتریس ورودی، تعداد مراحل پردازشی نیز تغییر می‌کند. اگر بخواهیم طراحی را به‌طور تمام برداری با تنها یک بردار پردازش نماییم، می‌توان اندازهٔ بردار را برابر با تعداد ستون‌های ماتریس ورودی در نظر گرفت. در این صورت، پردازش کاملاً برداری و در یک مرحله انجام می‌شود و بلوک انباشتگر نیز حذف خواهد شد. بنابراین، QRD به‌طور کاملاً انعطاف‌پذیر طراحی شده و برای هر ماتریس با ابعاد دلخواه و اندازه‌های برداری مختلف قابل استفاده است.

بلوک  $t$  نیز مطابق روابط (۲) تا (۴) طراحی شده است. با به‌دست آوردن  $\hat{R}$  و  $t$ ، در مراحل بعدی به جانشینی‌های پیشرو و پسرو نیاز است. مطابق روابط زیر، ابتدا از جانشینی پیشرو بردار  $h$  به‌دست می‌آید؛ سپس بردار  $u$  از جانشینی پسرو حاصل می‌شود. در تمامی روابط  $k=1, \dots, NM$  است. روابط (۱۵) و (۱۶) به ترتیب جانشینی‌های پیشرو و پسرو را نشان می‌دهند.

$$h_k = \frac{t_k^* - \sum_{j=1}^{k-1} \hat{R}_{k,j}^H h_j}{\hat{R}_{k,k}^H} \quad (15)$$

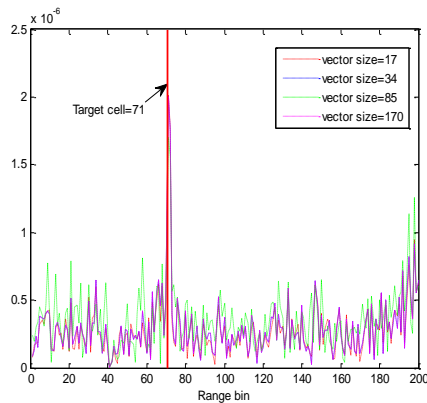
$$u_k = \frac{h_k - \sum_{j=1}^{k-1} \hat{R}_{k,j} u_j}{\hat{R}_{k,k}} \quad (16)$$

در شکل (۶) طراحی‌های جانشینی‌های پیشرو و پسرو نشان داده شده که از سه بخش اصلی واحد کنترل، واحد حافظه و مسیر داده تشکیل شده است. واحد حافظه، ورودی‌های  $\hat{R}$  و  $t$  را دریافت می‌کند و از چندین حافظهٔ دومنظوره تشکیل شده است. واحد کنترل شامل پنج حالت است. حالت اول به اولین درایه از هر سطر و حالت دوم به آخرین درایه از همان سطر اشاره دارد. حالت سوم به تکمیل سطر قبلی پیش از شروع سطر جدید می‌پردازد. حالت چهارم به هر محاسبهٔ جدید در بردار خروجی و حالت پنجم به مزدوج مختلط اختصاص دارد. از آنجا که داده‌های ورودی مختلط هستند، برای ترانهاده نمودن  $\hat{R}$  به حالت پنجم واحد کنترل نیاز است.



شکل (۶): طراحی جانشینی‌های پیشرو و پسرو

1- Azimuth



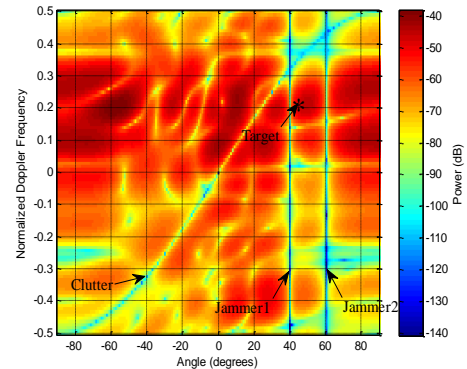
شکل (۸): تست استاتیکی برای تعیین موقعیت هدف

### ۵- نتایج پیاده‌سازی

در این قسمت به نتایج پیاده‌سازی ساختار طراحی شده روی تراشه Arria 10 FPGA پرداخته می‌شود. مصرف منابع سخت‌افزاری مدل STAP در فاکتورهای مختلف برای هر یک از اندازه‌های برداری در جدول (۴) نشان داده شده است. ماژول منطقی وفقی (ALM)<sup>۱</sup>، پایه‌ترین بلوک سازنده این نوع FPGAها است که از دو ALUT<sup>۲</sup> تشکیل شده است. استفاده از یک FPGA با بلوک‌های DSP ممیز شناور مانند Arria 10 برای پیاده‌سازی مدل STAP، نشان داد که محدودیت در پیاده‌سازی یک‌بردار با ابعاد ماتریسی بزرگ، تعداد بلوک‌های DSP است. به این ترتیب، تعداد ALMها حفظ می‌شوند. مقادیر نشان می‌دهند که با افزایش اندازه بردار، مصرف منابع نیز افزایش می‌یابد. در جدول (۴) برای روش یک برداری (۱۶۰)، حدود ۳۶٪ از ALMها و ۷۳٪ از DSPهای موجود، مصرف شده‌اند. همچنین در این حالت، ۹۶٪ از بلوک‌های DSP مصرف شده، ممیز شناور هستند. از سویی دیگر، حدود ۷۰٪ صرفه‌جویی در مصرف منابع سخت‌افزاری برای کوچکترین اندازه برداری (۱۷) نیز به دست آمده است. بنابراین، روش چندبردار منجر به یک پیاده‌سازی کارآمد شده است.

از آنجا که داده‌های STAP مختلط هستند، پیاده‌سازی ارائه شده در این مقاله نیز ممیز شناور تک دقتی<sup>۳</sup> است. ولی به‌طور معمول، حجم محاسبات به‌صورت ممیز شناور حقیقی (RFLOP)<sup>۴</sup> مطرح می‌شود. بر این اساس، RFLOPهای مورد نیاز برای هر بخش از مدل STAP بر حسب ابعاد مکعب داده در جدول (۵) آورده شده‌اند [۱۶]. مطلوب‌تر است که برای ارزیابی پیاده‌سازی، RFLOP در هر ثانیه (RFLOPs)<sup>۵</sup> محاسبه شود که با تقسیم RFLOP بر زمان اجرای محاسبات به دست می‌آید. در

حدود ۰/۲۱ است. بنابراین، مختصات (۰/۲۱ و ۴۵°) در شکل (۷)، بیانگر موقعیت هدف است که توانی حدود ۴۰dB- آن مشهود است.



شکل (۷): الگوی وفقی زاویه-داپلر

همان‌طور که پیشتر مطرح شد، در این مقاله برای کاهش حجم محاسبات STAP روش چندبردار پیاده‌سازی شده است. اندازه‌های برداری به‌گونه‌ای انتخاب شدند که مضری از سلول‌های آموزشی باشند. بنابراین، چهار اندازه برداری ۱۷، ۳۴، ۸۵ و ۱۷۰ در نظر گرفته شد که به ترتیب پردازش در ۱۰، ۵، ۲ و ۱ مرحله انجام می‌شود. از اینجا به بعد، تأثیر اندازه‌های برداری نیز در عملکرد مدل مورد بررسی قرار می‌گیرد.

پردازش وفقی زمانی- مکانی عمل فیلترینگ برداری را بر روی ترکیب داده‌های مکانی و زمانی برای هر نمونه فاصله خاص به‌طور مجزا انجام داده و از وزن‌های فیلتر وفقی برای آشکارسازی هدف استفاده می‌کند [۲۶-۲۵]. در اینجا از تست استاتیکی برای تعیین حضور یا غیاب هدف استفاده می‌نماییم که در رابطه (۱۷) به‌صورت یک مسأله دو فرضیه‌ای آمده است. فرضیه  $H_0$  بیانگر غیاب هدف و فرضیه  $H_1$  نشان‌دهنده حضور هدف می‌باشد.

$$\begin{cases} H_0: y = y_j + y_c + y_n \\ H_1: y = \alpha y_t + y_j + y_c + y_n \end{cases} \quad (17)$$

در شکل (۸) سیگنال‌های جمع‌آوری شده توسط آرایه آنتنی در هر یک از پردازش‌های ۱۰، ۵، ۲ و ۱ برداری بعد از انجام پردازش STAP و اعمال فیلترینگ آن، نشان داده شده است. مطابق جدول (۳)، بُرد هدف ۱۷۳۲ m است که متناظر با سلول شماره ۷۱ می‌باشد. در شکل (۸)، موقعیت سلول هدف با یک خط عمودی قرمز نشان داده شده است. همان‌طور که ملاحظه می‌شود، تمامی خروجی‌ها حول خط مذکور دارای یک پیک با قدرت بالا هستند. بنابراین، مدل طراحی شده در هر یک از حالت‌ها به‌درستی موقعیت هدف را تشخیص داده است.

1- Adaptive Logic Module (Alm)  
2- Adaptive Look-Up Tables (Alut)  
3- Single Precision  
4- Real Floating Point  
5- Rflop Per Second (Rflops)



### ۶- نتیجه‌گیری

در این مقاله، به طراحی و پیاده‌سازی پردازش وقتی زمانی-مکانی در رادارهای هوایی پرداخته شد و روشی مؤثر برای پیاده‌سازی کارآمد آن پیشنهاد گردید. یک هواگرد با رادار پهلونگر پالسی تک‌پایه که از پردازش وقتی زمانی- مکانی برای آشکارسازی هدف و تضعیف تداخل‌هایی همچون کلاتر و جمینگ استفاده می‌کند، در محیط متلب شبیه‌سازی شد و داده‌های موردنیاز برای تست و بررسی سامانه به‌دست آمدند. مدل ارائه‌شده برای محاسبه بردار وزن STAP با چندین اندازه برداری مورد شبیه‌سازی و پیاده‌سازی قرار گرفت. الگوی وقتی زاویه- داپلر و تست استاتیکی برای آشکارسازی هدف نشان داد که بردار وزن در هر یک از اندازه‌های برداری، به‌درستی محاسبه شده است. نتایج پیاده‌سازی روی تراشه Arria 10 حاکی است که تقسیم پردازش یک مرحله‌ای به چندین مرحله پردازش متوالی، منجر به کاهش حجم محاسبات، کاهش مصرف منابع سخت‌افزاری، توان مصرفی پایین‌تر و فرکانس عملکرد بالاتر می‌گردد. هرچند اندازه برداری بزرگتر به کمترین زمان اجرا منجر می‌شود. ولی حجم محاسبات نیز بسیار سنگین خواهد شد و این امر مانع بهره‌برداری مناسب از تراشه‌های با ظرفیت محدود می‌شود. لذا، با توجه به مزیت‌های فراوان روش چندبرداری نسبت به تک‌برداری، می‌توان از آن در پیاده‌سازی زمان- آنی سایر پردازش‌های چندبُعدی نیز بهره گرفت.

### ۷- مراجع

- [1] W. L. Melvin, "A stap Overview: IEEE Aerospace and Electronic Systems Magazine," vol. 19, no. 1, pp. 19-35, 2004.
- [2] R. Klemm, "Principles of Space-Time Adaptive Processing," IEE Radar, Sonar, Navigation and Avionics, London: IEE Press, 2002.
- [3] J. Ward, "Space-Time Adaptive Processing for Airborne Radar," (Technical Report no. 1015) Lexington, MA: Lincoln Laboratory, MIT, 1994.
- [4] W. L. Melvin, "Space-Time Adaptive Radar Performance in Heterogeneous Clutter," IEEE Transactions on Aerospace and Electronic Systems, vol. 36, no. 2, pp. 621-633, 2000.
- [5] Z. Hao and M. Luo, "Adaptive Mainlobe Jamming Suppression Method for STAP Airborne Radar," IET International Radar conf., 2013.
- [6] L. E. Brennan and L. S. Reed, "Theory of Adaptive Radar," IEEE Transactions on Aerospace and Electronic Systems, vol. 9, no. 2, pp. 237-252, 1973.
- [7] A. S. Paulus, W. L. Melvin, and B. Himed, "Performance and Computational Trades for RD-STAP Algorithms in Challenging Detection Environments," IEEE Radar conf., 2016.
- [8] N. A. Gawande, J. B. Manzano, A. Tumeo, N. R. Tallent, D. J. Kerbyson, and A. Hoisie, "Power and Performance trade-offs for Space Time Adaptive Processing," IEEE 26th International conf. on Application-Specific Systems, Architectures and Processors, 2015.

جدول (۶) پارامترهای مختلف برای بررسی کارایی مدل، ارائه شده‌اند. حجم محاسبات ممیز شناور حقیقی در واحد زمان با GFLOPs نشان داده شده است. همان‌طور که مشخص است، حجم محاسبات در اندازه برداری ۱۷۰ تقریباً پنج برابر اندازه برداری ۱۷ است. بنابراین، با کاهش اندازه بردار، حجم محاسبات و همچنین توان مصرفی کاهش و  $F_{max}$  افزایش یافته است. هر چند که با بزرگ‌شدن اندازه بردار، زمان اجرا نیز کاهش می‌یابد. ولی با افزایش مصرف منابع سخت‌افزاری، توان مصرفی و حجم محاسبات و همچنین کاهش  $F_{max}$  همراه است. این امر مانع بهره‌برداری مناسب از منابع تراشه‌های با امکانات محدود می‌گردد. بنابراین، نتایج نشان می‌دهد که روش پردازش چندبرداری به‌ویژه برای سامانه‌هایی مانند STAP که از حجم سنگین محاسبات برخوردارند و از ابعاد ماتریسی بزرگ استفاده می‌کنند، بسیار مؤثر و کارآمد است.

جدول (۴): مصرف منابع سخت‌افزاری مدل STAP برای چندین اندازه برداری

اندازه بردار	ALMها	ثبات‌ها	بیت‌های حافظه MLAB	بلوک‌های DSP ممیز شناور	بلوک‌های DSP
۱۷	(/۸)	(/۳)	(/۰/۵)	۱۳۹	(/۱۲)
۳۴	(/۱۱)	(/۵)	(/۱)	۲۴۱	(/۱۹)
۸۵	(/۲۱)	(/۹)	(/۲)	۵۴۹	(/۳۹)
۱۷۰	(/۳۶)	(/۱۸)	(/۳)	۱۰۵۴	(/۷۳)

جدول (۵): RFLOPهای موردنیاز برای محاسبه بردار وزن

تجزیه QR	$AL_s(NM)^T - 2/67(NM)^T$
بردار فرمان	$6(NM)$
جانشینی پیشرو	$4(NM)^T$
جانشینی پسرو	$4(NM)^T$
مجموع	$\{8(L_s+1)-2/67(NM)\}+6(NM)$

جدول (۶): پارامترهای مختلف برای بررسی کارایی مدل STAP

اندازه بردار	زمان اجرا (μs)	توان (mW)	$F_{max}$ (MHz)	GFLOPs	GFLOPs/Watt
۱۷	۲۶۲	۶۳۶۹	۲۷۸	۲۸/۱	۴/۴۱
۳۴	۱۸۹	۷۳۳۰	۲۷۴	۵۱/۷	۷/۰۵
۸۵	۱۵۷	۱۰۴۱۶	۲۶۱	۸۶/۶	۸/۳۱
۱۷۰	۱۴۴	۱۵۴۴۱	۲۱۰	۱۴۰	۹/۰۶

- [17] Z. U. Mahmood, M. Alam, K. Jamil, and Z. O. Al-Hekail, "FPGA Implementation of Space-Time Adaptive Processing (stap) Algorithm for Target Detection in Passive Radars," *Progress in Electromagnetics Research C*, vol. 35, pp. 35-48, 2013.
- [18] T. M. Benson, R. K. Hersey, and E. Culpepper, "GPU-based Space-Time Adaptive Processing (STAP) for Radar," *IEEE High Performance Extreme Computing conf. (HPEC)*, 2013.
- [19] M. A. Richards, "Fundamentals of Radar Signal Processing," Tata McGraw-Hill Education, 2005.
- [20] R. Woods, J. McAllister, G. Lightbody, and Y. Yi, "FPGA-based Implementation of Signal Processing Systems," John Wiley & Sons Ltd. U.K, 2008.
- [21] G. H. Golub and C. F. Van Loan, "Matrix Computations," Johns Hopkins University Press, Baltimore, Third edition, 1996.
- [22] M. Jervis, "Advances in DSP design tool flows for FPGAs," *Military Communications conf.*, pp. 2041-2046, 2010.
- [23] E. Rezagholizadeh, M. A. Sebt, "Clutter Mitigation in Airborne Radar Using DeterministicSubspace Clutter and Direct Data Domain", *Journal of Radar*, vol. 4, no. 2, pp. 1-10, 2016 (In Persian).
- [24] S. Sadeghi, A. Sheikhi, "Jamming Noise Suppression Using Discrete Polynomial-Phase Transform in Pulsed Radars", *Journal of Radar*, vol. 5, no. 1, pp. 53-66, 2017 (In Persian).
- [25] M. R. Taban, A. Sheikh Mozaffari, "Adaptive Radar Signal Detection in Gaussian Clutter with Autoregressive Model Using Kalman Filter", *Journal of Radar*, vol. 2, no. 3, pp. 1-12, 2014 (In Persian).
- [9] L. B. Fertig, "Analytical Expressions for Space-time Adaptive Processing (STAP) Performance," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 1, pp. 42-53, 2015.
- [10] J. M. Lebak and A.W. Bojanczyk, "Design and Performance Evaluation of A Portable Parallel Library for Space-Time Adaptive Processing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 3, pp. 287-298, 2000.
- [11] W. K. Liao, A. Choudhary, D. Weiner, and P. Varshney, "Performance evaluation of a parallel pipeline computational model for space-time adaptive processing," *The Journal of Supercomputing*, vol. 31, no. 2, pp. 137-160, 2005.
- [12] K. Rajan and L. M. Patnik, "Implementation of STAP Algorithms on IBM SP2 and on ADSP 21062 Dual Digital Signal Processor Systems," *Microprocessors and Microsystems*, vol. 27, no. 4, pp. 221-227, 2003.
- [13] F. Xikun and W. Yongliang, "Real-time implementation of Airborne Radar Space-Time Adaptive Processing on Multi-DSP System," *International conf. on Radar (CIE '06)*, 2006.
- [14] S. Bin, L. Shaohong, R. Yi, and L. Jingsheng, "Realization and Comparison of QRD Algorithms for STAP," *2nd Asian-pacific conf. on Synthetic Aperture Radar (APSAR)*, Proc. of IEEE, pp. 306-309, 2009.
- [15] Q. Li, J. S. Cao, and X. Pei, "Real-time and extensible calculation of STAP weights on FPGA," *12th International conf. on Signal Processing*, pp. 425-428, 2014.
- [16] A. Jarrah and M. Jamali, "Software tool for Efficient FPGA Design of Direct Data Domain Approach for Space-Time Adaptive Processing" *Electronics Letters*, vol. 49, no. 13, pp. 789-791, 2013.

---

## **Design & Efficient Implementation of STAP with the Multi-Vector Method for the Target Detection in Airborne Radars**

**N. Hasanikhah, S. Amin-Nejad<sup>\*</sup>, Gh. Darvish, M. R. Moniri**

University of Guilan, Rasht

(Received: 25/10/2017, Accepted: 12/09/2018)

### **Abstract**

Space-Time Adaptive Processing (STAP) is a fundamental technique in improving the performance of radars which are acted in the presence of severe dynamic disturbances such as clutter and jamming. STAP operation is based on very high sampling rates from signals received simultaneously from several antenna arrays and a number of pulses. Therefore, the volume of computation is very high and its implementation is difficult. In this paper, multi-vector method is proposed to reduce the amount of STAP calculations. Implementation results show that the proposed method, in addition to reducing the computational volume, leads to reduced hardware resources, reduced power consumption and increased maximum operating frequency. For example, the calculation volume for 6 antenna arrays, 10 receiving pulses and 200 range samples with a vector size of 17 is obtained 28.1 GFLOPS with maximum frequency of 278 MHz, and the computation time of 262  $\mu$ s on the Arria 10 chip. Therefore, the multi-vectoring method can meet the real-time requirements of STAP weights. Angle-doppler adaptive pattern and the static test for the target detection indicate that using the above method is very practical for calculating weights.

**Keywords:** Airborne Radar, Clutter, Jamming, Space-Time Adaptive Processing

---

<sup>\*</sup> Corresponding author E-mail: amin@liv.ac.uk