Elektronik İstihbarat Algoritma Yazılımlarının OpenMP Kullanılarak Çok Çekirdekli İşlemciler Üzerinde Paralelleştirilmesi

Murat Sever, Alparslan Fişne

REHİS, ASELSAN, Ankara, Türkiye
{msever,afisne}@aselsan.com.tr

Özet. Sayısal Sinyal İşlemciler (DSP) gömülü sistemlerde geniş kullanım alanına sahiptirler. Aşırı güç tüketimi ve ısınma problemleri işlemcilerde çok çekirdekli mimariyi ortaya çıkarmış ve günümüzde bu eğilim de hız kesmeden devam etmektedir. Bu değişime paralel olarak aslen tek çekirdekli işlemcilerde çalışmak üzere yazılmış uygulamaların çok çekirdekli mimariye uygun bir biçimde taşınması yani paralelleştirilmesi, günümüz yazılım geliştirme dünyasının en önemli gündem maddelerinden biri olmuştur. Bu çalışmada elektronik istihbarat amaçlı mevcut sinyal işleme algoritmaları paralelleştirme imkanları açısından incelenerek, OpenMP çatısı kullanılarak Texas Instruments'a (TI) ait 8-çekirdekli işlemci üzerinde paralelleştirilecektir. Yine TI tarafından sağlanan UIA altyapısı ile toplanacak ölçüm sonuçları ile paralelleştirmenin ne kadar ölçeklenebilir olduğu raporlanacaktır.

Anahtar Kelimeler: DSP·OpenMP·Sinyal İşleme·Paralelleştirme·Gömülü Sistemler·Comint

Abstract. Digital Signal Processors have an extensive use in embedded systems. There is a growing popularity in integration of many cores into one chip due to power consumption and heating considerations. In parallel with the multicore trend, an important aspect of software development today is proper migration to multicore environment. In this paper, we investigate parallelization in existing communication intelligence algorithms. Parallelized versions of selected algorithms are implemented on 8-core Texas Instruments (TI) DSP using OpenMP framework. We will present performance results, obtained by Unified Instrumentation Architecture (UIA) and report scalability of OpenMP parallelization.

Keywords: DSP· OpenMP·Signal Processing·Paralellization· Embedded Systems·Comint

1 Giriş

Moore yasası, işlemci hesap güçlerinin her 18 ayda bir ikiye katlanacağını öngörmüş ve bu kural uzun bir süre geçerliliğini korumuştur. İşlemcilerin saat hızlarının arttırılması, yakalanan performans başarısı yanında ciddi problemleri de doğurmuştur: Aşırı güç tüketimi ve ısınma. Tek bir işlem birimi ile mümkün olabilecek performans sınırına yaklaşılması çip üreticilerini işi aralarında paylaşıp aynı anda çalışabilen daha verimli çok çekirdekli işlemcilere yöneltmiş ve birden fazla işlem birimi içeren yongalar piyasaya sunulmuştur. Çok çekirdekli mimari sayesinde veri eş-zamanlı olarak işlenerek daha yüksek performans elde edilmektedir. Ancak tek çekirdekli mimaride çalışan kodun çok çekirdekli mimariye aktarılması kolay bir iş olmayıp ayrıyeten bir takım kütüphane ve derleyici desteği de gerektirmektedir. Texas Insruments (TI) firması kendi geliştirme ortamları olan Code Composer Studio'ya (CCS) OpenMP desteği sunarak bu yolda önemli bir adım atmıştır.

Bu çalışmada, Aselsan bünyesinde tek çekirdekli işlemcilerde çalıştırılmak üzere elektronik istihbarat amaçlı geliştirilen kod parçacıkları paralelleştirme imkanları açısından incelenecek, paralelleştirme kabiliyeti barındıran kodlar OpenMP programlama modeli kullanarak paralelleştirilip CCS ortamına aktarılacak ardından hedef DSP platformu olan TMDSEVM6678LE üzerinde çalıştırılarak zaman ölçümleri alınacaktır. Zaman ölçümlerinin alımında System Analyzer bileşeni kullanılarak gerçek zamanlı çalışan koda en az biçimde müdahale edileecektir. Farklı alanlardan birçok uygulamanın C6678 hedef DSP platformu üzerinde paralelleştirme örneğini görmek mümkündür. Yakın zamanda yer alan çalışmalardan birinde, HEVC video çözümleme işlemi OpenMP kullanılarak paralelleştirilmiş ve tek çekirdeğe kıyasla %70 civarında hızlanma kaydedilmiştir [1]. Yine TI tarafından yapılan çalışmada Sentetik Açıklıklı Radar (SAR) algoritmaları C6678 platformu üzerinde OpenMP yardımıyla çok çekirdekli mimariye aktarılmış ve SAR algoritmalarının 2, 4 ve 8 çekirdek üzerinde başarılı bir biçimde ölçeklendiği teyit edilmiştir [2].

Bu çalışma şu şekilde biçimlendirilmiştir: Elektronik istihbarat amaçlı kullanılan algoritmalara 2. bölümde yer verilmiştir. 3. Bölümde paralelleştirme ve OpenMP ile alakalı genel bilgiler sunulacaktır. Çalışmada kullanılan gömülü platformlar 4. bölümde tanıtılacaktır. Gerçeklemeye ilişkin detaylar ve sonuçlar 5. bölümde ele alınmıştır. Son bölümde ise çalışmadan elde edilen neticeler değerlendirilecektir.

2 Elektronik İstihbarat Algoritmaları

Elektronik istihbarat yeteneklerinin ani tepkilere karşı duyarlı olabilmesi için ilgili algoritmaların hızlı bir şekilde çalıştırabilir olması gerekmektedir. Gelişen işlemci mimarisine bağlı olarak algoritmaların çalışma süreleri giderek azalmakta, paralel işlemeye uygun algoritmaların kullanılması bu süreleri daha da kısaltmaktadır. Elektronik istihbarat algoritmalarının çok sayıda alt branşı vardır. Bunlardan en önemlileri, telsiz yayınların kip-çözme işlemleri ve ortamdaki yayın tespitidir. Bu çalışmada algoritma paralelleştirme çalışmalarına örnek olarak Kip-Çözme (Demodülasyon)

Algoritması ve Yayın Tespit Algoritması ele alınmaktadır. Bu iki algoritma paralelleştirmeye uygun olduğu için çalışmamızın odak noktasını oluşturmuştur.

2.1 Kip-Çözme Algoritması

Kip-Çözme Algoritması'nda ele alınan işlem genlik kiplenimine sahip yayınların dinlenmesidir. Almaçlardan alınan taban banda indirgenmiş kompleks verinin genliği alınarak sayısal ses verisi oluşturulmaktadır [3]. Kip-Çözme senaryosu Fig. 1'de sunulmuştur.

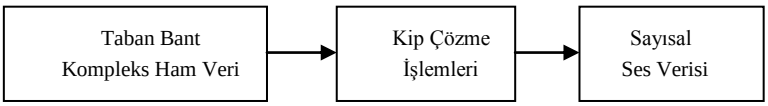


Fig. 1. Kip-Çözme senaryosu

2.2 Yayın Tespit Algoritması

Almaçlardan alınan frekans uzamındaki veri üzerinde öncelikle spektrum oluşturma işlemi yapılmaktadır. Kullanıcı tarafından girilen eşik değerine bağlı olarak spektrumdaki eşik noktasını geçen noktalar yayın olarak tespit edilmektedir. Yayın tespitine ilişkin senaryo Fig. 2'de gösterilmektedir.

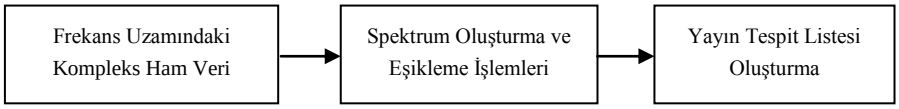


Fig. 2. Yayın tespit senaryosu

3 Paralelleştirme ve OpenMP

Donanımdaki çok çekirdekli mimariye geçiş eğilimi ile birlikte, yazılım dünyasında da tek çekirdek üzerinde çalışan kodların çok çekirdekli mimaride çalışmasına yönelik birçok paralel programlama modelleri oluşturulmuştur. Message-Passing Interface (MPI), Pthreads, Open Computing Language (OpenCL) ve Open Multi-processing (OpenMP) bu çabanın bir sonucu olarak ortaya çıkmıştır. MPI genelde dağıtık bellek mimarili platformlarda, OpenCL ise heterojen sistemlerde yoğun olarak kullanılmaktadır. OpenMP ise paylaşımlı bellek mimarili çok çekirdekli platformlar için bir programlama çatısı sunmaktadır.

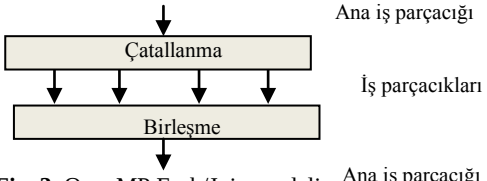


Fig. 3. OpenMP Fork/Join modeli

OpenMP yeni bir programlama dili olmayıp, mevcut sıralı bir şekilde işleyen kodların çok çekirdekli mimarilere kolay bir şekilde taşınabilmesine yönelik derleyici direktifleri içermektedir. Direktifler kodun hangi parçasının paralel bir şekilde çekirdeklere dağıtılarak çalıştırılacağını belirtir. Uygulama OpenMP desteği ile çalıştırıldığında paralel çalışırken, OpenMP desteği belirtilmediğinde kaynak kodda herhangi bir değişiklik gerektirmeksizin tek çekirdekte çalıştığı gibi çalışabilmektedir. Böylece paralel versiyon ile sıralı versiyon tek bir kaynak kodda tutulabilmektedir. Uygulama ilk çalıştığında tek bir iş parçacığı ile çalışmaya başlar. Paralel bölgeye ulaşıldığında ise iş parçacıkları oluşturulur ve bu iş parçacıkları farklı çekirdekler üzerine atanarak çalıştırılır. Paralel bölgenin sonunda ise senkronizasyon işlemi gerçekleştirilir ve kod ana iş parçacığı üzerinden çalışmaya devam eder. Paralel bölgenin başlangıcı *#pragma omp parallel* direktifi ile belirtilir. Bu durum Fig. 3'de sunulmuştur [4].

OpenMP yapıları sayesinde mevcut iş yükünün bütün çekirdekler üzerinde paylaşılmasını sağlayan "veri paralelleştirmesi" kolayca tanımlanabilir. Veri paralelleştirmesinin en kolay uygulanacağı bölgeler ise *for* döngüsü içeren kod parçalarıdır. Yalnız kodda yer alan her for döngüsü iş yükü paylaştırması açısından uygun olmayabilir. OpenMP yalnızca paralelleştirilecek bölgelerin tanımlanmasına ve kullanılacak çekirdek sayısına kontrol imkanı sunar. Kodun tek çekirdek üzerinde çalışma süresini T_1; aynı kod parçasının P adet çekirdek üzerinde paralel çalıştığında geçen zamanı T_P olarak kabul edersek, hızlanma (speedup) (1) numaralı formüldeki gibi tanımlanır. Hızlanma çekirdek sayısıyla doğru orantılı bir biçimde artarsa sistemin ölçeklenebilir olduğu söylenebilir.

$$S = \frac{T_1}{T_P} \tag{1}$$

4 Gömülü Platform

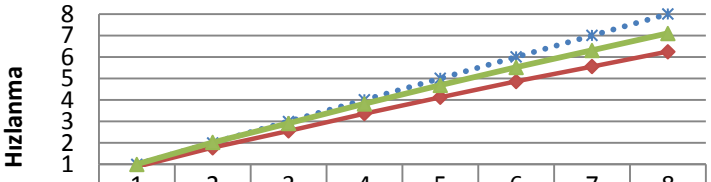
Çalışmada bahsedilen algoritmalar iki farklı gömülü platform üzerinde çalıştırılarak ölçümler alınmıştır. Bunlardan ilki algoritmaların asıl üzerinde koştuğu PowerPC platformudur. Platform, 1.0 Ghz işlemci hızına, 32kB L1, 1MB L2 belleğe ve toplamda 35W'lık güç tüketimine sahiptir. OpenMP paralelleştirmenin gerçekleneceği platform ise çok çekirdekli TI6678 platformudur. TMDSEVM6678LE modülü üzerinde 8 adet birbiri ile eşdeğer C66x çekirdeği barındıran, hem sabit hem kayar nokta işlem gücüne sahip bir DSP platformudur. 1GHz saat hızında sadece 10W gibi düşük güç tüketimine sahip oluşu platformu gömülü sistemlerde popüler hale getirmiştir. Sistem 32kBL1, 512 kBL2 belleğe sahiptir. C6678 aynı zamanda 4096kB'lık paylaşılır bir belleğe (MSMCRAM) sahiptir [5]. Bu çalışmada L1 bellekler önbellek olarak L2 bellek ise iş parçacıklarına özel alan olarak tanımlanmıştır. TI OpenMP gerşekleşiminde ise MSMCRAM alanı hem önbelleklenebilir hem de önbelleklenmez olarak kullanıcının erişimine sunulmuştur [6]. Kod ve sabitler önbelleklenebilir alana, çekirdekler arası erişilmesi gereken değişkenler ise önbelleklenemez alana yerleştirilmiştir.

5 Çok Çekirdekli Gerçekleştirim

Bu bölümde seçili, tek çekirdek için yazılmış mevcut elektronik istihbarat algoritmalarının nasıl CCS ortamına aktarıldığı anlatılacaktır. Algoritmalar içindeki veri paralelleştirmesi kullanılarak değişen sayıda çekirdek kullanarak (1,2,...,8) zaman ölçümleri UIA aracılığı ile alınır. OpenMP kütüphanesinde yer alan `omp_set_num_threads` metodu ile iş paylaşımı yapacak çekirdek sayısı çalışma anında belirlenebilir. `#pragma omp parallel for` derleyici direktifi ile paralel bölgenin başlangıcı belirtilir. Zaman ölçümü ise UIA aracı vasıtasıyla `Log_write2` metodu ile başlama ve bitiş olayları yayınlayarak yapılır.

5.1 Test Sonuçları

Seçilen algoritmalar ilk olarak tek çekirdekli PowerPC8640D (PPC) platformu üzerinde çalıştırılmış ve ölçümler alınmıştır. Ardından C6678 hedef DSP platformuna taşınmış ve burada çok çekirdekli mimaride çalıştırılarak ölçümler tekrarlanmıştır. AM Kip-Çözme için OpenMP paralelleştirmesi neticesinde, 8 çekirdek kullanım durumunda, PPC tek çekirdeğe göre 6,24 kat, TI tek çekirdeğe göre ise yaklaşık 7 kat hızlanma sağlanmıştır. Hızlanmaya ait grafik Fig. 4'te gösterilmiştir.



	1	2	3	4	5	6	7	8
İdeal	1	2	3	4	5	6	7	8
Hızlanma PPC (Tek çekirdek)	0,88	1,78	2,56	3,36	4,12	4,86	5,55	6,24
Hızlanma TI (Tek çekirdek)	1,00	2,02	2,91	3,82	4,68	5,52	6,31	7,09

Fig. 4. AM Kip-Çözme için sağlanan hızlanma

Yayın tespitine ait algoritma da benzer şekilde ilk olarak PPC üzerinde ardından paralelleştirilerek hedef DSP platformunda çalıştırılmıştır. Yayın tespiti için hesaplamada kullanılan nokta sayısı 6400 adet tutulduğunda ölçeklenebilir bir hızlanma sağlanamamıştır. Bu durum, iş yükünün çekirdeklere dağıtılması esnasında mecburen harcanan zamanın (overhead) fazla olmasından kaynaklanmaktadır. Yayın tespitindeki nokta sayısı 25600'e çıkarıldığında ise orijinal tek çekirdekli PPC platformuna göre 3 kata yakın hızlanma sağlanmıştır. Burada önemli bir diğer husus da güç tüketimidir. C6678 platformu 10W'lık, PPC platformu ise 35W'lık bir güç tüketimine sahiptir. Bu durumda yaklaşık üçte birlik bir güç tüketimi ile üç kata varan

bir performans artışı kaydedildiği unutulmamalıdır. Yayın tespitine ilişkin hızlanmaya ait grafik Fig. 5'te verilmiştir.

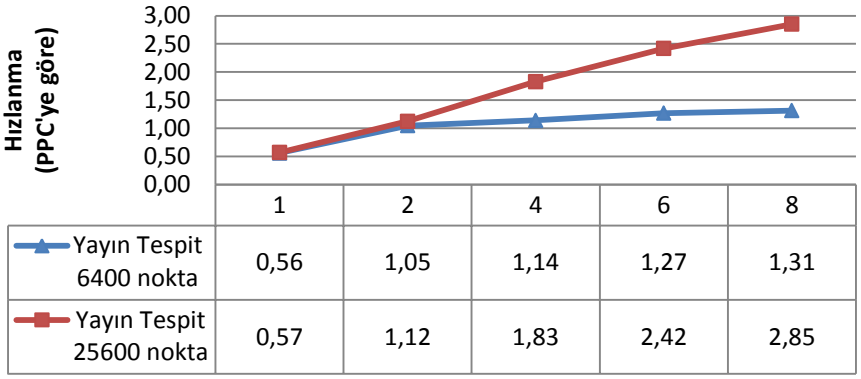


Fig. 5. Yayın tespiti algoritmasının paralelleştirmesi neticesinde sağlanan performans artışı

6 Sonuç

Bu çalışmada, Aselsan içinde elektronik istihbarat amaçlı tek çekirdekli mimarilerde çalışması için yazılmış sinyal işleme algoritmalarının, OpenMP kullanılarak paralelleştirilmesi ve TI tarafından geliştirilen C6678 çok çekirdekli DSP platformu üzerinde gerçeklenmesi ele alınmıştır. OpenMP, bellek paylaşımlı mimarilerde algoritmaların hızlı ve kolay bir biçimde paralelleştrimesine olanak sağlayan bir programlama modelidir. Seçilen algoritmalar, hedef platform bellek mimarisi de gözetilerek OpenMP'nin sunduğu iş-paylaşımlı yapılar sayesinde kolayca çok çekirdekli mimaride çalışacak şekilde modifiye edilmiş, yapılan ölçümler neticesinde paralelleştirmeden ölçeklenebilir bir fayda sağlandığı gözlenmiştir. İş yükünün çekirdeklere paylaştırılması sayesinde, seçili algoritmalarında tek çekirdekli orijinal platformdakinin sadece üçte biri kadarlık güç tüketiminde 3 kattan 6 kata varan hız artışı saptanmıştır.

Kaynaklar

1. M. Chavarrías, F. Pescador, M. J. Garrido, E. Juárez and C. Sanz, "A Multicore DSP HEVC Decoder Using an Actor-based Dataflow Model," in IEEE International Conference on Consumer Electronics (ICCE), 2015.
2. D. Wang, M. Ali and E. Blinka, "Synthetic Aperture Radar (SAR) Implementation on a TMS320C6678 Multicore DSP," Texas Instruments, 2015.
3. [Çevrimiçi].http://www.radio-electronics.com/info/rf-technology-design/am-reception/amplitude-modulation-detection-demodulation.php. [12 May 2015].
4. C. Chapman, G. Jost and R. Van Der Pas, Using OpenMP, The MIT Press, 2008.
5. C. Hu and D. Bell, "KeyStone Memory Architecture," Texas Instruments, 2010.
6. OMP (OpenMP Runtime for SYS/BIOS) Users Guide, Texas Instruments.