|  |  |
| --- | --- |
|  | **T.C.**  **GEBZE TEKNİK ÜNİVERSİTESİ**  **MÜHENDİSLİK FAKÜLTESİ** |

**STAJ DEFTERİ**

**Öğrencinin;**

|  |  |
| --- | --- |
| **Adı, Soyadı** | **Batuhan Karaoğlu** |
| **Numarası** | **171024089** |
| **Bölümü** | **Elektronik Mühendisliği** |
| **Staj Yaptığı Yer** | **ASELSAN ELEKTRONİK SANAYİ ve TİCARET A.Ş.** |
| **Staj Tarihleri** | **17.08.2020 – 11.09.2020** |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 17.08.2020** | | **KAPSAMI:** Online eğitimler |
|  | |  |
| **Şirket Hakkında Bilgi:**  ASELSAN elektronik ürünler ve sistemler tasarlayan, geliştiren, üreten ve ürünlerin satış sonrası servis hizmetlerini karşılayan; yüksek teknolojili ve çeşitli ürün yelpazesine sahip bir elektronik sanayi kuruluşudur. Kuruluş amaçlarından bazıları; Türk Silahlı Kuvvetleri’nin her türlü elektronik cihaz ve sistem gereksinimlerini, dışa bağımlılığı en aza indirecek şekilde, en kısa zamanda karşılamaktır.  **Stajın Başlaması:**  Macunköy tesislerine gidilip belgelerin teslim edilmesi ve kayıtlarımızın yapılıp stajyer kartlarımızın alınması.    Pandemi dolayısıyla bizler için online olarak hazırlanan İSG eğitimlerinin ve oryantasyon genel bilgilendirme eğitimlerinin aktifleştirilmesi  **İSG 360 Eğitimine Başlanması:**  Bu eğitim kapsamında bir şirkette çalışmak için gerekli olan İSG eğitimleri aldık ve acil durumda neler yapmamız gerektiği ile ilgili bilgiler edindik.  İSG için alınan bu eğitim bütün detayları ile birlikte 12 saat süren bir eğitimdir. Eğitim içerisinde 24 E-eğitim,1 Video, 8 Sınav bulunmaktadır. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 18.08.2020** | | **KAPSAMI:** Online eğitimler |
|  | |  |
| **Elektrostatik Deşarj Eğitimine Başlanması:**  Bu eğitim kapsamında elektriğin yayılma yollarından statik elektrik incelendi. Sonrasında elektrostatik deşarjın öneminden bahsedildi ve bu durumlar karşısında alınması gereken önlemlere yer verildi.  Bu eğitimle birlikte elektrostatik deşarjın elektronik cihazlara ve insanlara zarar verebileceğini, buna karşı önlemlere sıkıca uymamız gerektiğini öğrenmiş olduk. Bu eğitim sonrasında 1 yıl geçerli olmak üzere elektrostatik deşarj eğitimi sertifikası aldık. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 19.08.2020** | | **KAPSAMI:** Online eğitimler |
|  | |  |
| **Aselsan’a Hoşgeldiniz ve Oryantasyon Genel Bilgilendirme Eğitimlerine Başlanması:**  Bu eğitimler kapsamında 2 E-eğitim aldık ve Aselsan hakkında genel bilgiler edindik.  Aselsan’ın birçok alanda faaliyet gösterdiğini ve birçok çalışma alanını içinde barındırdğını öğrendik.    Oryantasyon eğitimi kapsamında ise Aselsan’da staj yapacağımız süre boyunca hangi kurallara riayet edeceğimizi öğrendik. Ayrıca bu eğitimle staj dönemimizi nasıl ve ne şekilde yürüteceğimiz gösterildi. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 20.08.2020** | | **KAPSAMI:** Staj Bölümüne yönlendirme |
|  | |  |
| Online eğitimlerimizi aldıktan sonra staj yapılacak bölüme yönlendirme maksadıyla Macunköy tesislerine gidildi burada staj sözleşmelerimizi geri aldıktan sonra ilgili bölümlere yönlendirildik. Gölbaşı yerleşkesinde bulunan REHİS(Radar Elektronik Harp İstihbarat Sistemleri) bölümüne yönlendirildim.  Burada Anten Tasarım bölümüne katıldım ve çalışma yaptığım alanlar ve ilgi duyduğum konular üzerine bilgi verdikten sonra ilgi alanıma göre seçilen sayısal tasarım ekibine gelerek ilgili mühendis ile tanıştırıldım.  Bu birimin yaptığı çalışmalar üzerine bilgi aldım, radar sistemlerinin taarruz ve savunma olarak 2 türlü yöntemi olduğunu öğrendim. Daha sonra ilgili departmanda yürütülen bir projeye dahil olarak projede ne yapılması gerektiği, amacı ve projenin kapsamı hakkında bilgi sahibi oldum. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 21.08.2020** | | **KAPSAMI:** Gömülü sistem için sinyal işleme |
|  | |  |
| Projede yapılacak iş için c programlama dili kullanarak yazılan kodları öğrendim ve bunları geliştirmek üzere çalışmalara başladım. Öncelikle DFT’nin bilgisayar üzerinde C programlama dili ile herhangi bir kütüphane kullanılmaksızın nasıl elde edilebileceğini gözlemledim. Ayrıca mikro denetleyiciler floating-point kabiliyetine sahip olmadıkları için ölçekleme yapmam gerektiğini de öğrenmiş oldum.    Buna göre;    Bilgisayar üzerinde işlem yaparken zamanı örneklediğimiz gibi frekansı da örnekleyerek fourier dönüşümünü o şekilde elde etmemiz gerekir. Bu sebeple ayrık zamanlı fourier dönüşümü yerine ayrık fourier dönüşümü kullanarak bilgisayarda bu işlemi gerçekleştirebiliriz.  Ayrık fourier dönüşümü formülü:    İle gösterilmektedir.  N örnek kullanarak ayrık fourier dönüşümünü hesapladığımızda bilgisayar üzerinde oluşacak zaman karmaşıklığı, O(N2) olacaktır.  Bu noktada belirtilen işlem(DFT) ayrık fourier dizisi işlemine denk gelmektedir. Bu nedenle x[n] periyodik olmasa bile biz ayrık fourier dönüşümü alırken işaretimizi periyodik gibi kabul etmekteyiz. Bu aslında bize algoritmadaki zaman karmaşıklığını azaltmak için olanak sağlamaktadır. Bu basitleştirilmiş forma Fast Fourier Transform(FFT) denilmektedir. FFT bizim değişkenlerimizi tek ve çift diye ayırarak gruplandırmaktadır:  Tek:    Çift:  Burada DFT formülündeki n değerini r olarak güncelliyoruz. Sonuç olarak:  Olarak elde edilir. Buna göre çarpma sayısı N2 iken tane çarpmaya indirgenmiştir. Bu işlem devam ettirildiği takdirde FFT sonucunda oluşacak çarpma sayısı Nlog2N olarak belirlenir. Bu da yazılacak koddaki zaman karmaşıklığının azalması demektir. Bu etkenler kapsamında kodu en az zaman karmaşıklığına sebep olan formatta yazacağız. Sonuç olarak bu teknikler mikro denetleyici üzerinde kontrol edilecektir. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 24.08.2020** | | **KAPSAMI:** sinyal işleme (LomontFFT) |
|  | |  |
| C programlama dili kullanarak öğrendiğimiz FFT ve DFT uygulamalarını kod üzerinde test etmek için gerekli algoritmalar oluşturuldu. Bu algoritmalar, gömülü sistemde kullanılacağı için işlemlerin hız durumu ve kütüphane kullanmadan yazılması gereken işlemler için araştırmalar yapıldı. Sonuç olarak cihaz üzerine aktarılacak kod için öncelikle FFT işlemleri gerçekleştirildi. Buna göre;  Öncelikle FFT(fast fourier transform) için hız ve kütüphane kullanılmaması bakımından karşılaştırılarak bulunan en uygun kod yazımı C# üzerinde yazılan LomontFFT adlı çalışmadır.  ışık içeren bir resim  Açıklama otomatik olarak oluşturuldu  Bu grafikte birçok kez çalıştırılan algoritmaların 2048, 4096 ve 8192 noktalı örnekler için çalışma sürelerini görmekteyiz. Grafikte de görüleceği üzere FFTW algoritması en ideal algoritmadır fakat bu algoritma içinde çeşitli kütüphaneler barındırmaktadır. Bu nedenle alternatif olarak Lomont’un real input için olan algoritmasını kullanmaktayız.  **LomontFFT:** LomontFFT algoritması 2010 yılında Chris Lomont tarafından geliştirilmiştir.  N-noktalı girişler x0,x1,….,xN-1 olsun, fourier dönüşümü N-noktalı karmaşık sayı olan y0,y1,….,yN-1 için;  a ve b değerleri başlangıçtaki ölçekleme değerinde ve üstel kısımda kullanılmaktadır. Sinyal işleme için a ve b değerleri (1, -1) olmaktadır. Bu karmaşık sayılar, xk içindeki mevcut frekanslardaki genlik ve faz değerlerini temsil eder. Ölçekleme terimini ile ifade edelim. Daha sonra Fourier transformunu tek ve çift x girdileri için yeniden yazalım: N/2 noktalı çift indexli ve N/2 noktalı tek indexlileri belirtmektedir. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 25.08.2020** | | **KAPSAMI:** sinyal işleme (LomonFFT) |
|  | |  |
| **8 noktalı girdi için 5. Terimi ele alalım:**  bıçak içeren bir resim  Açıklama otomatik olarak oluşturuldu  Sondan 2. satırda xj e ve o gösterimleri ile, en sonda ise alt simgeli olarak kullanılmıştır. Bunlar aşağıdaki model ile ilişkilidir:    e = 0 ve o = 1 değerlerini kullanırsak giriş sırasını elde etmiş oluruz bit çevirme işlemi sonucunda çıkışta oluşan değerler “output order” ile ifade edilen kısımdır ve denklemin son satırındaki ifadelere karşılık gelmektedir. Genel boyutu N=2n olan girdiler için 1 adım bit çevirme işlemi O(N) karmaşıklığındadır, ve bit çevirmek için n=log2N adım kullanılmaktadır. Sonuç olarak oluşan zaman karmaşıklığı O(Nlog2N) olmaktadır.  **Gerçek Değerli FFT:** Girdimizin gerçek sayı olmasıdır. Gerçek girdi için bir FFT'nin çıktısı hala karmaşık değerlidir, bu nedenle bu FFT'ye gerçekte karmaşık FFT adı verilir. Gerçekten karmaşığa FFT, aynı giriş verilerinde karmaşıktan karmaşık FFT'ye göre yaklaşık iki kat daha hızlı performans gösterir. FFT’nin eşleniği bize FFT’nin simetri olduğu bilgisini verir.  saat içeren bir resim  Açıklama otomatik olarak oluşturuldu  Bu nedenle yalnızca çıkışın yarısını üretmek bize fourier dönüşümünü verecektir. Fakat *y0* bu simetri listesi içinde değildir onun simetri eşi *yN/2* olmaktadır. Gerçek değerli *y0* ve *yN/2* , *x0* ve *x1* in bulunduğu yerde olacaktır. Bu kısım kodda şu şekildedir:    Ayrıca RealFFT fonksiyonunun içinde TableFFT adlı fonksiyon çağrılmaktadır. Bu fonksiyonun amacı ise fourier transformu esnasında kullanılan kosinüs ve sinüs değerlerinin oluşturulan bir tablodan çekilmesidir. Böylelikle kod ilk başta tabloyu oluşturduktan sonra işlemleri gerçekleştirmek için tablodaki değerleri kullanacaktır ve performans artacaktır.  Bit Reverse işlemi ise kodda aşağıdaki şekilde yapılmaktadır:  ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu    Bu kısımlar C diline çevrildikten sonra C üzerinde bu kod çalıştırıldı. Girilen işarete göre gerçek ve sanal kısımlar olarak çıktısı alındı. Bu çıktılar ile genlik ve faz değerleri bulundu ve ekrana yazdırıldı. Daha sonra fourier transformu, FFT algoritması kullanmadan DFT ile bulundu. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 26.08.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama. |
|  | |  |
| Önceki günlerde C ile yazılan FFT ve DFT algoritmalarını kullanarak C# ve Windows Forms Application kullanarak bir arayüz üzerinde bu verileri kontrol etmem istendi.  Öncelikle daha önceden kullanmadığım bir dil olan C# yazılım dilinin temel yazılış tarzını öğrendim. Daha sonra yapacağım proje kapsamında önceden ilgili mühendis tarafından C# ile yazılmış Windows uygulamasını inceleme fırsatı buldum. Bu kodun içeriğinden yola çıkarak ZedGraph aracını kullanmayı öğrendim daha sonra arayüz uygulamasını oluşturmaya başladım.  VisualStudio kullanarak uygulama geliştirmek için form uygulaması için proje oluşturduk ve öncelikle arayüzde kullanacağımız araçları yerleştirdik. Fonksiyonun üretilmesi, FFT ve DFT başlıkları altında 3 adet sekme oluşturuldu.  Daha sonra ZedGraph’ı kullanabilmek için referanslar kısmına ZedGraph.dll dosyası eklendi. Grafik panelini forma eklemek için ise araçlar kısmından ZedGraphControl aracını kullanmaktayız:  ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu      Bu şekilde fonksiyon oluşturma kısmındaki işaretlerin çizdirileceği grafik oluşturulurmuş oldu. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 27.08.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama. |
|  | |  |
| İlk kısmın arayüzünün arka planını oluşturmak için ise C# dili kullanıldı. Değişkenler global olarak form sınıfı içinde tanımlandı böylece değişkenlerimize class içindeki tüm fonksiyonların erişmesine olanak sağlandı.  Daha sonra oluşturulan GenerateFunction() fonksiyonu ile arayüzde kullanıcıdan alınan sinyal sayısı, sinyal boyutu ve sinyal tipi bilgilerine göre rastgele frekans, genlik ve faz bilgisine sahip sinyaller üretildi. Daha sonra bu fonksiyon plotting() adında bir fonksiyon içinde çağrıldı. Burada ise arayüzde oluşturulan panel üzerinde üretilen sinyallerin rastgele atanan genlik, faz ve frekans bilgileri gösterilmektedir.  GenerateFunction() fonksiyonu içinde global olarak tanımlanan x ve y değerlerinden x’e sinyalin oluşturulduğu noktalar x = 0,1,2,3….N-1 atanırken y değişkenine ise sinyalin x noktasında aldığı değeri atandı. Bu fonksiyon içinde global olarak tanımlanan PointPairList tipindeki değişkene sinyalin bilgileri(x ve y) eklendi.  Plotting() fonksiyonu çağrıldığında, x ve y değişkenleri kullanılarak oluşturulan PoinPairList kullanılarak grafik üzerine bu noktaların çizdirilmesi sağlandı.  Arayüz üzerine eklenen buton ile beraber oluşturulan fonksiyonun içinde plotting() fonksiyonu çağrıldı ve aynı zamanda butona basıldığında grafiğin önce temizlenmesi ve ölçek ayarlarının sıfırlanması için ZedGraphControl aracının fonksiyonları kullanıldı.  Bu işlemleri gösteren örnek bir çıktı ve örnek kod aşağıdadır:  Çıktı:  ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu  Kod:  ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 28.08.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama. |
|  | |  |
| 2. kısımda bu sinyallerin fourier dönüşümünü FFT kullanarak almak üzerine çalıştım. Bu noktada LomontFFT algoritmasını kullandım.  Plotting() fonksiyonu ile değeri atanan y değişkeni, RealFFT(y , true) şeklinde 2.sekmedeki FFT butonunun fonksiyonu içinde çağrıldı. RealFFT fonksiyonun kodu şu şekildedir:  ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 31.08.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama(DFT) |
|  | |  |
| Kod üzerinde FFT kısmı tamamlandıktan sonra 3. Sekme olan DFT tabının kodu yazılmaya başlandı.  **DFT:**    Formülü ile ifade edilir. Bu formül kod üzerinde aşağıdaki şekilde DFT tabına entegre edilmiştir:  ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 01.09.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama(Goertzel) |
|  | |  |
| DFT tabının içinde radiobutton vasıtası ile yöntemi seçmemiz sağlandı. DFT sekmesine bu şekilde “Goertzel Algoritması” da eklenmiş oldu.  **Goertzel Algoritması:** Öncelikle DFT işlemini konvolüsyon şeklinde ifade edelim:          x[n] sinyalini dürtü cevabı olan bir LTI filtreden geçirelim, sonucu gözlemlersek yk[N], n=N noktasında karşılık gelen DFT katsayısını verir. .  Filtre, z-dönüşümü gösterimi ile şu şekildedir:    nesne, saat içeren bir resim  Açıklama otomatik olarak oluşturuldu  Buradan, olarak elde edilir. Filtrenin z-dönüşümüne çeşitli  işlemler uygulayalım:  nesne, saat içeren bir resim  Açıklama otomatik olarak oluşturuldunesne içeren bir resim  Açıklama otomatik olarak oluşturuldu  En son elde edilen sonuçla birlikte başta bahsettiğimiz filtreleme işlemi aşağıdaki gibidir:  nesne, saat içeren bir resim  Açıklama otomatik olarak oluşturuldu  Bu işlem “Goertzel Algoritması” olarak isimlendirilmektedir. Bu filtrenin çıktısını yalnızca yk[N]’de değerlendirdiğimiz için, çarpan olan sadece n=N anında kullanılmaktadır. Sonuç olarak, algoritma belirli bir k değeri için x[k] değerinin dönüşümünü hesaplamak için N tane gerçek sayının çarpımı ve 1 adet karmaşık sayı çarpımı içermektedir. FFT ile kıyaslandığında belirli bir k için hesap yapmak bu algoritma ile daha hızlı olmaktadır.  Bu algoritmanın teorik kısmını öğrenildikten sonra C# dilinde kullanılmak üzere, yazılmış kodlar incelendi. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 02.09.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama(Goertzel) |
|  | |  |
| FFT ile Goertzel algoritmasının karşılaştırılması:   * N’nin 2’nin tamsayı üssü olmasına gerek yoktur. * Rezonans frekansı sıfır ile fs Hz arasında herhangi bir değer olabilir. * İşleme başlamadan önce FFT’de olduğu gibi bir giriş verisi bloğunun depolanmasına gerek yoktur. İşleme, ilk giriş zamanı örneği ile başlayabilir. * Goertzel algoritması için veri bitlerinin ters çevrilmesine gerek yoktur. * M tane farklı frekans tespit edilmek istendiğinde goertzel algoritması M kez uygulanacaktır. Eğer M < log2N ise FFT işleminden daha etkilidir.   Goertzel algoritmasını DFT tabına eklemek için gerekli kod yazıldı. Bu algoritmayı koda uyarladıktan sonra genliği doğru elde edilmesine rağmen faz kısmı hatalı olarak elde ediliyordu.  Bu sorunu çözmek için internette çeşitli araştırmalar yaptıktan sonra sorunu çözmek için dönüşümün gerçek ve sanal kısımları hesaplanmadan önceki satıra şu kısım eklendi:    Bu işlem sonucunda genlik ve faz doğru şekilde elde edildi. Bu algoritma için C# dilinde yazılan kod aşağıdadır:  ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 03.09.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama(Goertzel) |
|  | |  |
| metin içeren bir resim  Açıklama otomatik olarak oluşturuldu  Yukarıda DFT ve Goertzel algoritması ile hesaplanan sonuçlar görülmektedir. Burada girişe sinüzoidal bir işaret verilmiştir. Frekansı ise k = 4 noktasında olacak şekildedir. FFT ile uyumlu olması açısından bu değerler 2’nin tamsayı üssü olacak şekilde seçilmiştir. Burada görüleceği üzere k = 4 noktasında iki algoritma da doğru sonucu vermiştir. Faz kısmına geldiğimizde işarete başlangıçta verdiğimiz 60olik fazı her iki çıkışta da görmekteyiz. Faz kısmının diğer frekans değerleri için bu kadar gürültülü olmasının sebebi ise genlik kısmında bilgisayar üzerinde hesap yapıldığı için sayıların tam olarak sıfıra eşit olmamasından kaynaklıdır. Diğer frekanslar üzerinde genlikler sıfıra çok çok yakın olsalar da aralarında faz farkı oluşmaktadır. Bu nedenle diğer frekanslardaki fazlar gürültü olarak nitelendirilebilir.  **Pencereleme Fonksiyonları:**  Arayüz üzerinde 3.sekme olan DFT sekmesi de tamamlandıktan sonra, 2. sekme olan FFT sekmesi için pencereleme(windowing) fonksiyonları arayüze eklenecektir. Bu nedenle pencereleme işleminin neden uygulandığı ve nasıl uygulandığı konusunda araştırmalar yapıldı.  Buna göre;  Tek bir frekansta salınan sinüzoidal bir sinyalimiz olduğunu varsayalım, bu sinyali frekans uzayında incelemek istediğimizde karşılaşmayı beklediğimiz işaret, pozitif kısımda bir adet dürtü olacaktır. Bununla birlikte, daha yüksek bir frekansta daha küçük bir dürtü ile karşılaşmamız da mümkündür, bu bize sinüs dalgasının optimum olarak üretilmediğini söylemektedir. Frekans uzayı, zaman uzayındaki temiz bir sinyale gerçekte karışma olup olmadığı, gürültü eklenip eklenmediği veya titreşim içerip içermediğini anlamamız adına gayet başarılıdır.  Sinyali oluştururken periyot sayısı bir tam sayı olmadığında, uç noktalar süreksizdir. Bu yapay süreksizlikler, orijinal sinyalde bulunmayan yüksek frekanslı bileşenler olarak frekans uzayında ortaya çıkar. Bunun sonucunda sanki bir frekanstaki enerji diğer frekanslara sızıyormuş gibi görünür. Buna, ince spektral çizgilerin daha geniş sinyallere yayılmasına neden olan spektral sızıntı denmektedir. Ortaya çıkan bu ekstra bileşenleri pencereleme adı verilen yöntemle en aza indirmek mümkündür. Pencereleme, işaretin kenarlarda yumuşak ve kademeli olarak sıfıra doğru değişen bir genliğe sahip, sonlu uzunluklu bir pencere ile çarpılması işlemi olarak adlandırılır. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 04.09.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama (Windowing) |
|  | |  |
| Arayüz üzerinde kullanılan pencereleme yöntemleri şunlardır:   * Hamming:   Pencere, en yakın yan lobu en aza indirecek şekilde optimize edilmiştir ve bu, daha basit katsayılara sahip yükseltilmiş bir kosinüs olan hanning penceresinin yaklaşık beşte biri kadar bir yükseklik verir. Bu pencere aşağıdaki şekilde formüle edilebilir.     * Hanning:   Gürültü ve titreşim ölçümleri yapılırken hanning penceresi yaygın olarak kullanılır. Kademeli geçişe sahiptir. Kademeli geçiş, ölçülen sinyali pencere ile çarparken genliklerde yumuşak bir değişiklik sağlar, bu da spektral sızıntıyı azaltmaya yardımcı olur. Hanning penceresinin formülize edilmiş şekli aşağıdaki gibidir. Pencereye ait grafikler de verilmiştir. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 07.09.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama (Windowing) |
|  | |  |
| * BlackmanHarris:   Blackman-Harris penceresi Hamming ve Hanning pencerelerine benzerdir. Pencereleme sonucu oluşan spektrum, geniş bir tepe noktasına sahip olmakla beraber, iyi bir yan lob bastırılmasına sahiptir. Bu pencerenin iki ana türü vardır, 4 terimli Blackman-Harris 90dB’lerin üstünde yan lob bastırılmasına ve orta derecede geniş bir main loba sahip, genel amaçlı iyi bir penceredir. 7 terimli Blackman-Harris penceresi, ihtiyaç duyulan tüm dinamik aralığa sahiptir, ancak geniş bir main lobu içinde barındırır.     * Flat Top:   Flat Top penceresi de olabildiğince sinüzoidaldir, ancak aşağıda görüldüğü gibi sıfır çizgisini geçmektedir. Bu da, frekans uzayında, sinyalin gerçek genliğine diğer pencerelerden daha yakın olan, çok daha geniş bir tepe noktasına neden olur. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 08.09.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama (Windowing) |
|  | |  |
| * Nuttall:   Bu pencere aslında Blackman-Harris pencere ailesinin bir üyesidir. Katsayıları daha özelleştirilmiş işlemler yapmak için aşağıda verilen şekilde seçilmiştir.  metin içeren bir resim  Açıklama otomatik olarak oluşturuldu  Pencereleme işleminde seçilmesi gereken pencere tipi oluşturacağımız giriş sinyaline göre farklılık göstermektedir. Bu nedenle üretilen sinyale göre en uygun pencere tipini seçip o şekilde pencereleme işlemini uygulamalıyız.  Bu bilgiler doğrultusunda arayüz üzerinde oluşturulan comboBox vasıtası ile pencere tiplerinin seçilmesi sağlandı ve kod kısmında da seçime göre pencere fonksiyonlarının üretilen sinyal ile çarpılma işlemi gerçekleştirildi. Daha sonra bu sinyalin fourier transformu FFT ile alınarak grafik üzerinde gösterildi. Ek olarak zaman uzayında sinyalin aldığı şekil de 1.Sekmede  gösterildi. Örneğin Blackman-Harris penceresi kullanılarak gerçekleştirilen bir pencereleme işlemi sonucunda, tek frekanslı sinüzoidal bir giriş sinyali aşağıdaki gibi revize edilecektir:  metin içeren bir resim  Açıklama otomatik olarak oluşturuldu | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 09.09.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama (Veri Tipleri) |
|  | |  |
| Data tipi gömülü sistemler için önem arz etmektedir. Bunun sebebi bu cihazların floating-point kabiliyetine sahip olmamasıdır. Double tipinde bir veriyi integer(int32), short(int16) veya long (int64) veri tiplerinden birine çevirirsek verimizde kayıp oluşması muhtemeldir. Bunun önüne geçmek için double olan veri ölçeklendirilerek veri kaybı büyük oranda azaltılabilir.  Ölçekleme (bit-shifting):  Bu işlem aslında bit-shifting olarak adlandırılmaktadır. Bir sayıyı 2’nin tam sayı üssü olan bir sayı ile çarparsak ilk sayıyı hem genişletmiş oluruz hem de aynı zamanda, sayının binary formdaki halini, çarptığımız sayının log2() fonksiyonundan çıkışı kadar kaydırmış oluruz. Bu işlemi gerçekleştiren C dilinde yazılmış bir kod ve çıktısı aşağıdaki gibidir:  ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu  Output:      Yukarıda kodun çıktısında da görüleceği gibi 30 olan giriş sayımız 512 ile çarpılarak 15360 sayısı elde edilmiştir. Burada 30 ve 15360 sayılarının binary formlarını da incelersek, 29 = 512 sayısı ile çarptığımız için 30’un binary formu aslında 9 birim sola kaydırıldı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 10.09.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama (Veri Tipleri) |
|  | |  |
| Arayüz üzerinde ilk sekme olan fonksiyon üretici sekmesine, data tipinin seçilmesine olanak sağlayan radioButton araçları eklendi. Double, Short, Integer veya Long veri tiplerinden birini seçtiğimizde bu seçim 3 sekmedeki işlemlerin de seçilen veri tipiyle yapılmasını sağlamaktadır.  Değişkenler:  Bilgisayarın geçici belleğinde bilginin saklandığı gözlere verilen sembolik adlardır. Bir C programında, bir değişken tanımlandığında bu değişken için bellekte bir yer ayrılır. Her değişkenin tuttuğu değerin nasıl bir veri olduğunu gösteren bir veri tipi vardır.  Aşağıda, veri tiplerinin kaç bit olduğu ve maksimim ve minimum alabileceği değerler gibi özellikleri gösteren tablo verilmiştir:  ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |

|  |  |  |
| --- | --- | --- |
| **YAPILAN İŞİN;** | | |
| **TARİHİ: 11.09.2020** | | **KAPSAMI:** C# ile DSP arayüzü tasarlama (Veri Tipleri) |
|  | |  |
| Double ve Float tipleri için C# dilindeki veri tipi özelliklerini gösteren tablo aşağıdaki gibidir:    ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu    Değişken tipleri arasında kesirli sayı veri tipi olarak float yerine double kullanmamın sebebi; “float” tipinin boyutu 4 bayttır. “double” tipinin boyutu ise 8 bayttır. “float” 7 basamaklı hassasiyete sahipken, “double” 14 - 15 basamaklı hassasiyete sahiptir.    Aselsan’da yapmış olduğum bu staj sayesinde C# dilinde arayüz geliştirmeyi öğrendim ve bu sırada daha önceden bilgim olmayan C# dilini de temel seviyede öğrenmiş oldum. Üniversite’de aldığım dijital sinyal işlemenin temelleri dersinde öğrendiğim teorik bilgilerin burada kod üzerinde gerçeklemesini yapma fırsatı buldum.  Stajımı sayısal tasarım ekibinde yapmam münasebetiyle, FPGA ve kullanım alanları üzerinde bilgi sahibi oldum. FPGA ile daha önce ilgilenmemiş olduğum için, staj süresinde ilgimi çeken FPGA üzerine stajdan sonra da bireysel olarak çalışmalar yapmak düşüncesindeyim.  Aselsan’da öğrendiğim temel seviyedeki bilgiler de bana bu noktada yol gösterici olacaktır. Staj süresince sayısal tasarım ekibinin göstermiş olduğu samimiyet ve ekip çalışması da sosyal alandaki becerilerimin gelişmesine katkı sağladı. | | |
| **Stajımı bu firmada yaptım.**  **Staj Yapanın İmzası** | **Staj Yeri Yetkilisinin**  **Adı, Soyadı, İmzası, Firma Kaşesi** | |
|  |  | |