1 Nicemleme ve Örnekleme

"Analog işaretleri sayısal devrelerle işlemek için öncelikle onları sayısal işaretlere dönüştürmemiz gerekir." cümlesi aslında içinde birçok gizli ima barındırmaktadır. İçinde geçen kelimelerin anlamlarını bilmeyenler ve yeni öğrenecekler için çok önemli bilgiler veriyor gibi görünse de, bu anlamları bilenler için çok sıradan hatta söylenmesi bile gerekmeyen bir cümledir. O zaman anahtar terimlerin anlamlarını açıklama ile başlayalım.

Analog ve sayısal işaret ne demektir, farkı nedir?: Tüm elektriksel işaretler analogdur. Sadece yorumlanması farklıdır. Analog işaretleri sürekli zamanın *sonsuz* küçük her anında olası *sonsuz* farklı değerden birisi olabilen voltaj değerleri şeklinde düşünürüz. Burada 2 defa *sonsuz* kelimesinin geçtiğine dikkat ediniz. Yani hem zaman eksenindeki ölçüm noktaları hem de voltaj değerleri sonsuz farklı olasılık içermektedir. Bu işaretlerin temsil ettiği fiziksel büyüklükler de aynı şekilde analogdur. Örneğin havadaki basınç dalgalarından oluşan sesi bir algılayıcı/mikrofon ile elektriksel işarete çevirdiğimizde işaretin büyüklüğü basıncın büyüklüğünü temsil eder, her ikisinin de hem zaman hem büyüklük ekseninde sonsuz farklı değer olasılığı vardır. İşareti zaman ekseninin bir anından diğerine bağlayan dalgaşekli de (waveform) sonsuz farklı şekil alabilir.

Sayısal işaret ise olası dalgaşekillerine bir sonluluk getirmeyi hedeflemektedir. Tabi ki dalgaşekillerinden bahsedince, zaman ekseninde de dalgaşekillerinin iki ucu arasının sonlu olması beklenir. Yani, ne kadar küçük/büyük olabilse de, sonlu zaman aralıklarından (aralıklar eşit olmayabilir) bahsediyoruz. Bu tasarımdaki amaç, elektriksel işaretler ile, sonlu bir zaman aralığındaki sonlu sayıda olası farklı dalgaşekli ile yine sonlu sayıda farklı mesajı temsil etmektir.

Anlaşılacağı üzere, birinci durumdaki işaretlere analog diğerine ise sayısal işaret denir. Elektriksel işaretin kendisi ise, anlık hedeflenen dalgaşekilleri sayısı sınırlı olmasına rağmen, sonsuz farklı şekil ve sonsuz farklı değer alabilir. Yani, işaret aslında analogtur ama temsil ettiği mesaj sayısı itibariyle sayısaldır. Herşey tasarımda hedeflenen yorumlamaya bağlıdır.

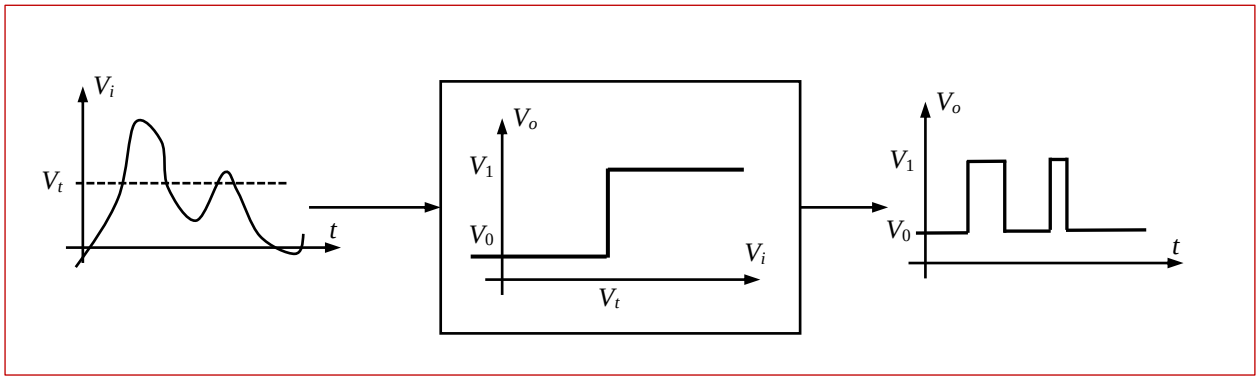
Analog devre nedir?: Kısaca işaretlerin analog olmasını gözeten ve koruyan devrelerdir.

Sayısal devre nedir?: Sayısal devreler ise olası durum (state) sayısının sonluluğu ile tanımlanır. Durum sayısı doğrudan devrenin hafızası ile ilgilidir. Olası durum sayısı istenildiği kadar çok (ama yine sonsuz değil) olabilir. Bu devreye hangi girdiler verilirse verilsin bu devre ancak sonlu adet durumdan birine sokulabilir. Bu durumda, sayısal devrelerin çıkışları da sonlu sayıda ve o anki

durumdan ve girdilerden hesaplanabilir olmaktadır. Buna kısaca *deterministik* yada determinizmi yüksek diyoruz. "Peki analog devreler deterministik değil mi?" sorusunun cevabı belirgin değildir. Çünkü girdiler sonsuz farklı dalgaşekli ve değerden oluşabilmekte, buna bağlı olarak çıktılar da sonsuz farklı dalgaşekli ve değer olabilmektedir. Analog devrelerin durumu pratikte hesaplanabilir olmayan atom-altı olaylara da bağlıdır. Buradan da yorumla "analog devrelerin durum sayısı sonsuzdur" veya "determinizmi düşüktür" diyebiliriz. Daha ilerisi filozofiye gireceğinden burada duralım ve sayısal devrelerin -en azından hedeflenen- durum sayısının sonlu olduğu ile yetinelim. Ancak sonraki bölümlerde tekrar değineceğiz ki, bu sebeple, sayısal devrelerden rastgele sayılar/işaretler üretilemez.

Sayısal devrelerde sonlu durum sayısı, hafıza elemanlarının girişlerinde karşılaştığı işaretleri sonlu sayıda sınıftan birisine sokmaya çalışması ile sağlanır. Günümüzde sayısal devreler bu hafıza elemanlarının girişindeki anlık değeri iki sınıftan birisine sokması şeklinde geliştirilmiş olduğundan herbir elemanın olası durum sayısı da r=2'dir. Sistemde/devrede H adet hafıza varsa olası durum sayısı da 2^H olur. Peki r=3 yada daha fazla olması mümkün değil miydi? İkinin avantajı nedir? Ya da $r=\infty$ (analog) olmasının ne gibi bir avantajı/dezavantajı olabilirdi ve neden 2 tercih edildi? Bu soruların cevapları elektroniğin tarihsel gelişimiyle ve yatırımlarla ilgilidir. Gelecekte analog bilgisayarların yaygın hale gelmeyeceğinin bir garantisi yoktur. Ancak yakın gelecekte olmayacağını söyleyebiliriz, çünkü teknoloji, altyapı, yatırımlar ve aritmetik r=2 yönünde gelişmiştir. Aksi yönde çok önemli bir gelişme olmadığı sürece (örn: kuantum bilgisayarlar), böyle kalacağı kesin gibidir.

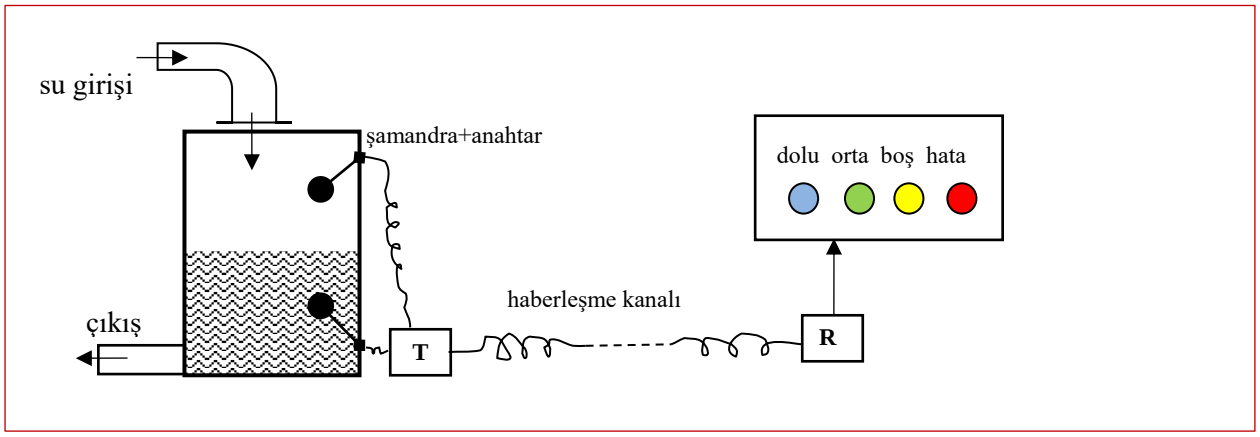
Şimdi bir devre düşünelim ki girişinde anlık olarak olası sonsuz farklı voltaj değerine göre çıkışında sadece 2 voltaj değeri üretsin. Bunun en kolay ve anlaşılır yöntemi girişte beklenen en yüksek ve en yüksek voltaj değerinin tam ortasına bir eşik koymak ve çıkışı ona göre belirlemektir.



Şekil 1.1. Girişindeki voltaja eşik değeri uygulayan devre.

Böylelikle çıkışta girişin bu eşik değerinden düşük yada yüksek olduğunu söyleyen bir tasarım yapmış olduk. Aslında 2 bölmeli bir nicemleme yapmış olduk. Bunlardan birisine 0 diğerine de 1 diyelim. Böyle bir devre, çıkıştaki işaretin bir lambayı yakması yada söndürmesi şeklinde kullanılarak, bir su tankının içindeki su seviyesini şamandıra ile belirleme amacıyla kullanılabilir. Lambanın yanması tankta şamandıradan yüksekte su olduğunu, sönmesi ise seviyenin daha aşağıda olduğunu söyler.

Tasarımı biraz daha ileriye götürelim ve Şekil 1.2'deki gibi 2 adet şamandıra kullanalım.



Şekil 1.2. İki şamandıralı+anahtarlı su tankı projesi.

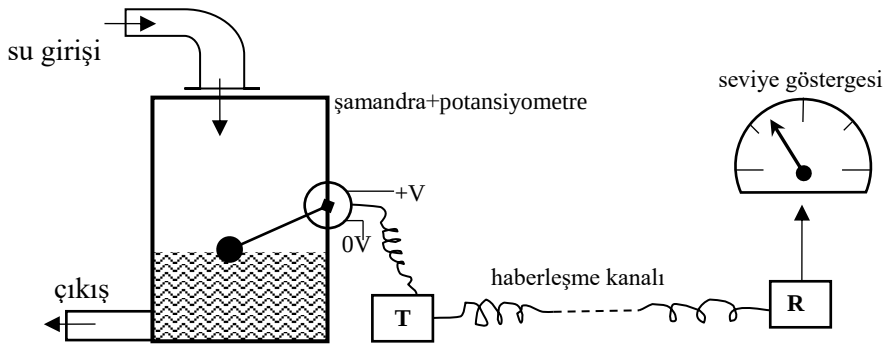
Şamandıraların her birindeki anahtarlar Şekil 1.1'deki gibi sadece 2 farklı (açık-kapalı) bilgi verebileceği için şamandıra sisteminden alabileceğimiz bilgi sadece 4 farklı durum belirtebilir;

1- anahtarlar açık-açık : tank boş ya da boşa yakın
2- kapalı-açık : su orta seviyelerde
3- kapalı-kapalı : tank dolu
4- açık-kapalı : şamandıralardan birisi bozuk

Şimdi Şekil 1.3'teki gibi tek şamandıra kullanalım ama şamandraya eşik değerinin geçildiğini belirten bir anahtar yerine dönen (rotary) bir potansiyometre takalım. Öyle ki, şamandıranın yükselişi potansiyometreyi bir yönde ve düşüşü de diğer yönde döndürsün. Böylelikle potansiyometrenin diğer iki ucuna uygulayacağımız sabit voltaj ile, potansiyometrenin orta ucu 0 ve +V arasında bir voltaj üretsin. Bir analog kadranlı voltmetre de üretilen voltajı göstersin.

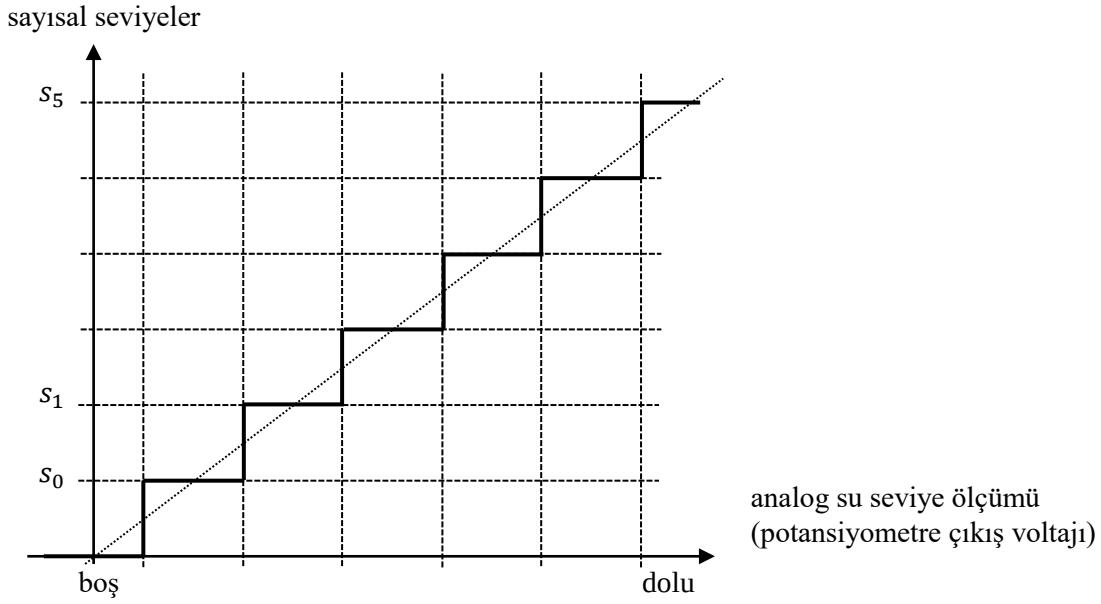
Şimdi de bu iki sistemi karşılaştıralım. Şekil 1.2'deki sistem bir sayısal ölçüm ve sayısal iletişim sistemidir. Çünkü ölçülen ve gönderilen mesaj sayısı sadece 4'tür (sonlu sayıda). Haberleşme kanalı diye belirtilen iletim hattında ne gönderildiğinin önemi yoktur, yeter ki bu 4 durumu birbirinden ayırabilecek bir işaret sistemi ve bir alıcı olsun. Buradan da anlaşılacağı üzere işaret sistemi ve alıcı birbirinden bağımsız şeyler değildir.

Şekil 1.3'teki sistem ise bir analog ölçüm ve iletişim sistemidir (T ile gösterilen iletim devresinin de analog olduğunu varsayalım). Su seviyesi de seviye göstergesinin ibre pozisyonu da sonsuz farklı değer alabilir. Aradaki iletim kanalı büyük ihtimal ölçülen voltaj değerini (belki de biraz kuvvetlendirerek) gösterge tarafına gönderir. Yani iletim hattı aslında bir analog haberleşme hattıdır.



Şekil 1.3. Tek şamandıralı-potansiyometreli su tankı projesi.

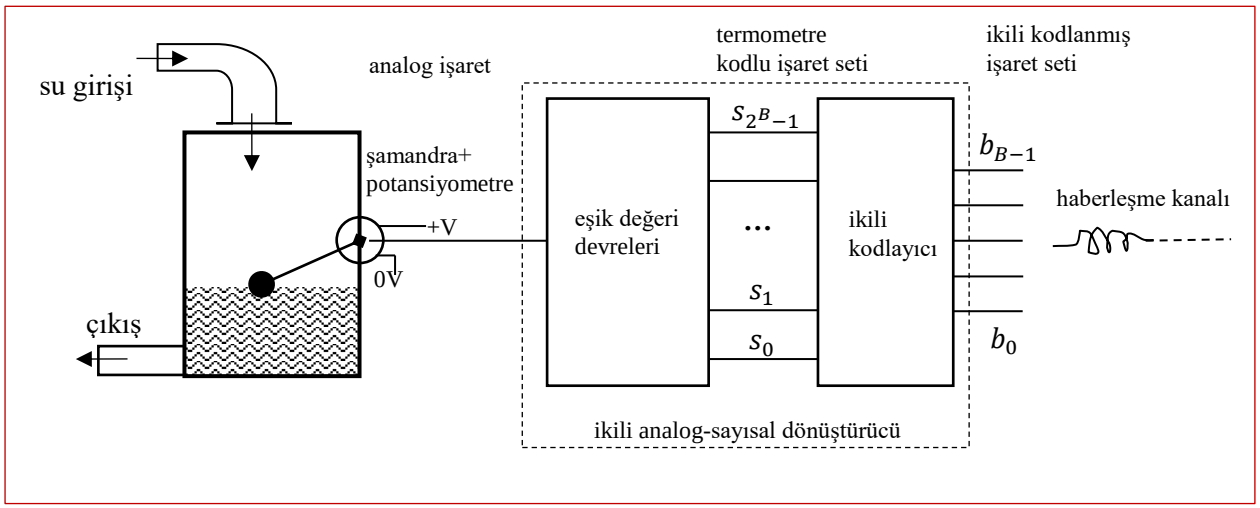
Bu iki örnek ile hem sayısal ve analog ölçüm farkını hem de sayısal ve analog haberleşme sistemlerinin farkını görmüş olduk. Tabi henüz birbirlerine göre avantaj/dezavantajlarını görmedik, ilerleyen bölümlerde göreceğiz. Şimdilik sayısal haberleşme sistemlerinin avantajlarının farkında olduğumuzu ve ölçüm yönteminden bağımsız olarak sayısal iletişim yapmak istediğimizi varsayalım. Örneğimizin özelinde, tank doluluğunu sayısal olarak ama daha hassas bir şekilde ölçmek isteyebiliriz. Hassasiyeti arttırmak için çok sayıda şamandıra kullanılabilir. Örneğin farklı seviyelere yerleştirilmiş 6 şamandıra kullanarak tankın doluluğunu tankın 1/12'si hatayla ölçebiliriz. Ya da daha ekonomik olabilmek için potansiyometre yöntemini kullanıp, analog olarak ölçülen su yüksekliğini sayısala dönüştürebiliriz. Örnek bir analogtan sayısala dönüştürme eğrisi Şekil 1.4'teki gibi olabilir.



Şekil 1.4. Analog seviyelerden 7 bölmeli sayısala dönüştürme eğrisi.

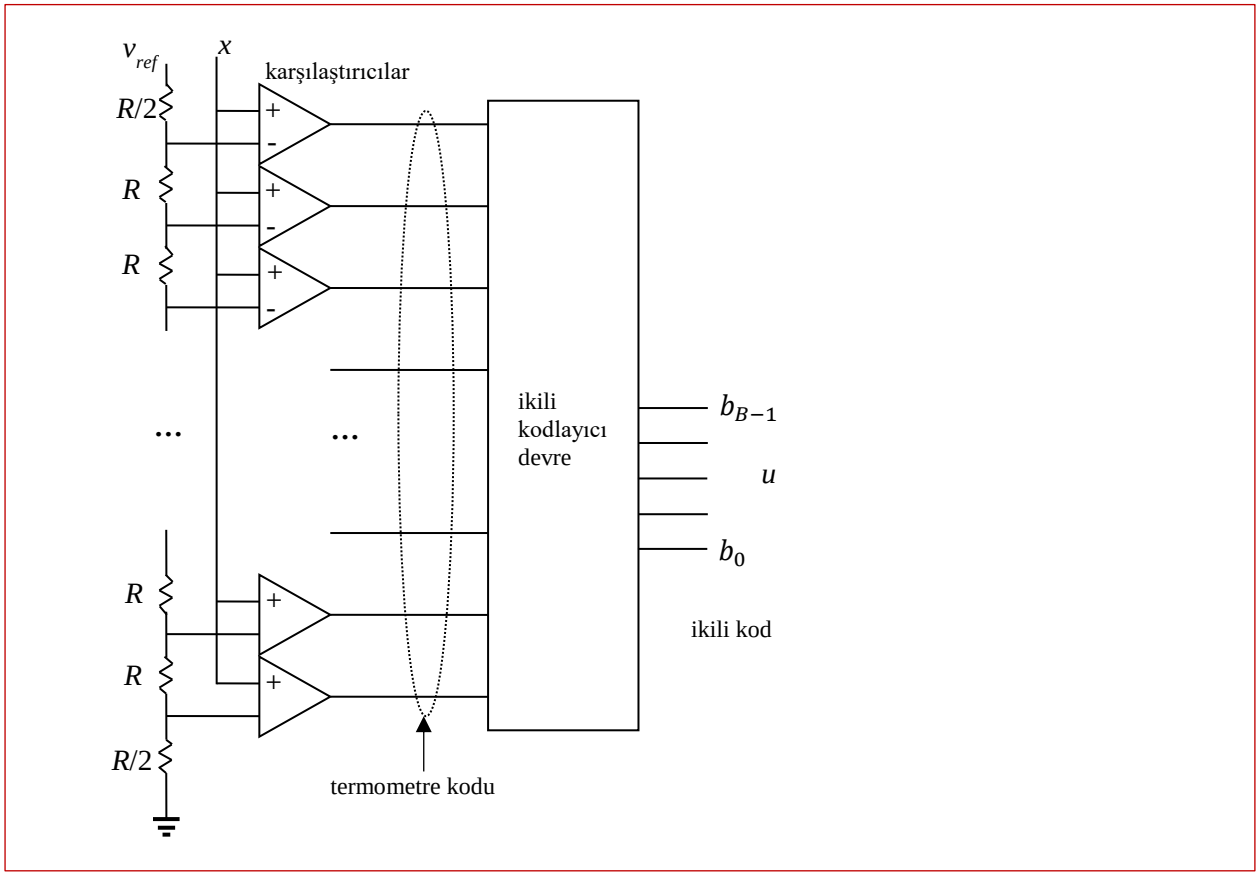
Altı adet şamandıra kullanıldığında ya da analog ölçüm voltajını 6 adet eşik değeri ölçme devresi ile sayısala çevirdiğimizde aldığımız bilgi 6 adet açık/kapalı (eşik değerinin üzerinde/altında)

bilgisidir. Yani bu haliyle gösterge cihazına (belki 6 adet uyarı lambası) 6 adet iletim hattı çekilmesi gerekir. Bu verimli bir sayısal gösterim değildir. Daha hassas (örneğin 32 eşik değeri) ölçüm yapmak istediğimizde 32 adet kablo mu çekeceğiz? (*not: Olası her değer için ayrı bir elektriksel işaret yolu olan bu kodlama şekline termometre kodu diyelim*). Sayısal devreler yardımıyla 32 eşik değeri durumunu belli sürelerle ardışıl olarak tek bir iletim hattından gönderme seçeneğimiz de vardır. Ancak, bu yöntem temsil şeklinin verimli olmaması durumunu düzeltmez. Daha verimli bir sayısal kodlama yapılması gerektiği açıktır. Olası 32 adet seviye aslında ikili temsil sistemiyle 5 adet işaret yolu ile gösterilebilir (2^5=32). O zaman tank seviyesi uyarı sistemi Şekil 1.5'teki hale evrilir.



Şekil 1.5. Analog ölçümlü sayısala çevirmeli su tankı projemizin tank bölümü.

Şekil 1.5'teki analog-sayısal dönüştürücünün analog devre içeren kısmı (eşik değeri devreleri) biraz daha detaylı olarak Şekil 1.6'da verilmiştir. Her karşılaştırıcının (–) girişine gerilim bölücü dirençler kullanılarak farklı bir referans voltajı uygulanmaktadır. Bu voltaj su tankı benzetimindeki şamandıra yüksekliğine benzer. Karşılaştırıcıların (+) girişine uygulanan *x* ile bu bölünmüş referans voltajı karşılaştırılır ve *x* daha yüksek ise çıkış lojik-1'e karşı gelen voltaj, *x* daha düşük ise 0 volt olur. Doğal olarak bir karşılaştırıcın çıkışı yüksek ise (lojik-1) onun altındaki karşılaştırıcıların çıkışları da yüksek olur (bu temsil şekline *termometre* kodu demiştik). Yani, örneğin 31 adet karşılaştırıcı ve 5 bit çıkış var ise ikili kodlayıcının görevi, (0,1,2,...31) seviyelerine karşı gelen 32 adet 31'er bitlik (000...0, 100...0, 110...0, 1110...0, ..., 111...1) kodu 32 adet 5'er bitlik (00000, 00001, 00010, ..., 11111) ikili koda dönüştürmektir. Böylelikle, girişteki 32 adet farklı voltaj aralığının her birini 5 bitlik bir ikili sayı ile temsil etmiş oluruz. Voltaj aralıkları eşit olmak zorunda değildir ama eşit olacak şekilde devreyi üretmek daha kolay ve anlaşılır olmaktadır. Örneğin voltaj aralığımız (0V, 32V) ise bu aralıklar ((-∞, 0.5V), (0.5V, 1.5V), (1.5V,2.5V), ..., (31.5V, +∞)) olur. Giriş voltajı 0.5'in altında ise 00000 ile 31.5'in üstündeyse de 11111 ile temsil edilmiş olur. Tabi bu dönüşümle hem bu voltaj değerleri hem de bu voltaj seviyeleri arasındaki değerleri gösterebilme hassasiyeti kaybolmuş olur. Örneğin girişteki voltaj 1.7V ise çıkışta 1.5V'a karşılık gelen 00001 kodu ile göstermiş oluruz. Aradaki 0.2V farkın temsil ettiği bilgi kaybolmuştur ve tekrar geri konulamaz. Bu tamamen bölütleme şekli ile ilgilidir ve ismi de *nicemleme*dir. (quantization). Daha hassas bir ölçüm isteniyorsa daha çok bölütleme gereklidir ancak hiçbir zaman analog hassasiyette olamaz.

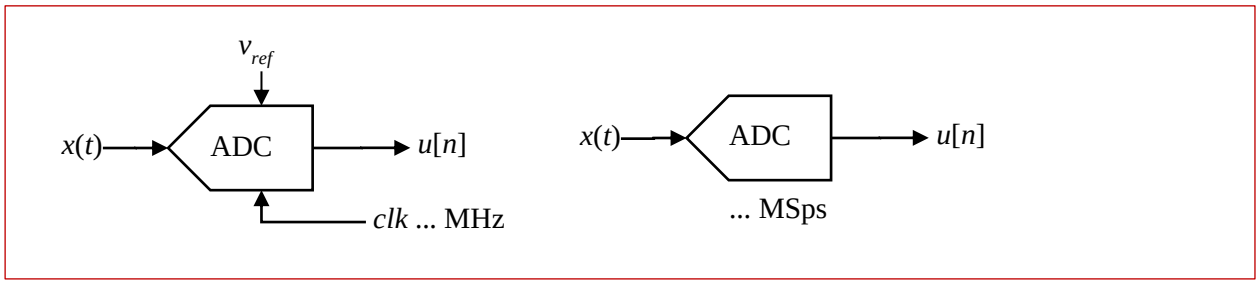


Şekil 1.6. Tam-çözümlü sayısal işaret kullanan analog-sayısal dönüştürücü.

Şekil 1.6'daki kavramsal devre analog voltaj değerlerini sayısal gösterime çevirmenin tek yöntemi değildir ama en kolay anlaşılanıdır. Burada, örneğin B=12 bitlik bir hassasiyet isteniyorsa, 4095 adet karşılaştırıcı devresi gerektiği açıktır. Zorluk sadece sayının çokluğundan değildir, aralıkları eşit yapabilmek te teknik zorluklar içermektedir. Tıpatıp aynı dirençleri üretebilmek te yetmez, karşılaştırıcıların giriş akımlarının (ve olası farklarının) da hesaba katılması gerekir.

Tahmin edileceği üzere, girişteki voltajın değişmesi ile çıkışta yeni voltaj değerini temsil eden ikili sayının değişmesi arasında bir zaman farkı vardır. Bu zaman farkının en büyük olduğu durumdaki süreye çeşitli isimler verilebilmekle birlikte en çok kullanılanına çevrim süresi, ardışıl çevrimlerin tekrarlanma frekansına da örnekleme frekansı (sampling frequency) ismi verilmiştir. Örnekleme frekansının birimi Örnek/saniyedir (Samples/second, Sa/s, Sps) (örn: MSps: megasamples per second). Bir sonraki çevrim süresince de çıkışların değişmemesi için sonuçlar bir kaydedicide tutulur. Bu kaydedicinin saat girişi aynı zamanda yeni bir çevrimi başlatmak için kullanılır. Bu saat işaretinin frekansı da çevrim frekansıyla aynı olduğu için çevrim frekansının Hz cinsinden söylenmesi olağandır. Saat işaretinin her peryodunda (yükselen kenarından başlayarak) analog işaretin o anki değeri bir sayısal değere çevrilir ve sonuçta bir sayı dizisi üretilir. Bu olaya örnekleme denir (not: birçok kaynak analog elektriksel işaretinin anlık değerini bir sonraki ölçüme kadar bir kapasitör üzerinde tutmak (hold) ve ölçüm sırasında ölçüm devresine sabit bir voltaj sunmak için için kapasitörü doldurma işlemine örnekleme demektedir. Ancak, asıl olay sayısal değerin çıkışta görünmesi olduğundan ve biz örnekle-tut kısmı ile ilgilenmediğimizden sayısal değeri çıkışta görme işlemine örnekleme demekte sakınca yoktur).

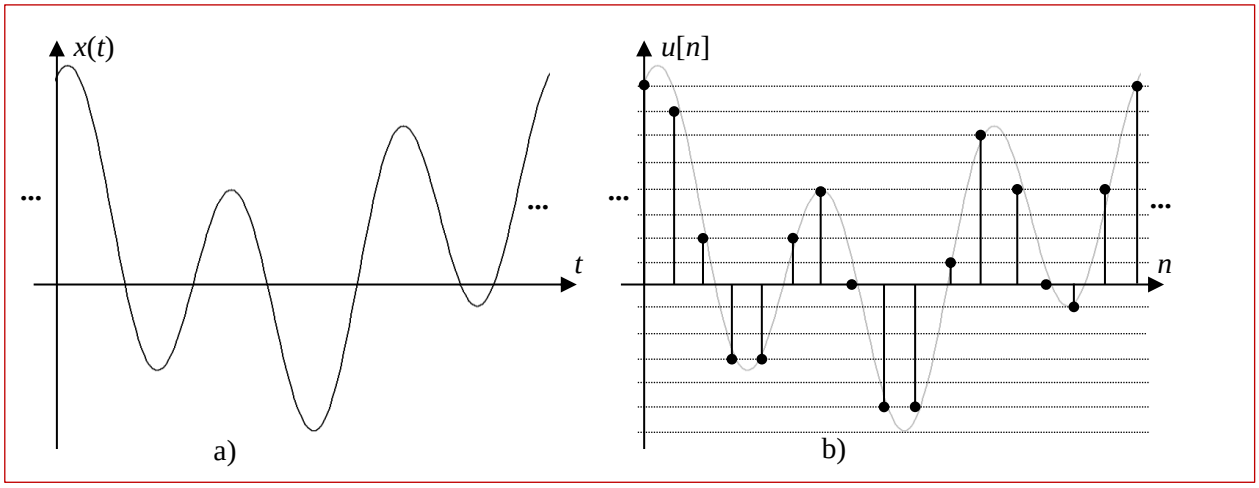
Yukarıda bahsi geçen konuları gözönüne aldığımızda bir analog-sayısal dönüştürücünün (ADC: analog-to-digital converter) blok gösterimi Şekil 1.7'deki gibi olur.



Şekil 1.7. ADC blok şeması. a) saat ve referans işaretleri ile, b) ana işaretler ile gösterim.

Analog işaretleri sayısal örnek dizilerine çevirmemizdeki amaç yapılacak işlemleri (işaret işleme) sayısal devrelerle gerçekleştirmektir. ADC'ler ve bu işlemin tersini yapan sayısal-analog dönüştürücüler (DAC'lar) sayısal devrelerin analog giriş-çıkış bağlantı noktalarıdır. Hem analog hem sayısal devreleri içerdiğinden ve birazdan açıklayacağımız üzere bu devrelerden yüksek başarım beklendiğinden, bu devreler tam analog ya da tam sayısal devrelere göre daha pahalı olurlar.

İkili kod kullanıldığında sayısal örnek dizileri de zaman grafikleri çizildiğinde bir şekilde analog karşılıklarına benzerler ama aynısı değildir. Şekil 1.8 bir analog işaret ve örneklerini göstermekte.



Şekil 1.8. a) Sürekli (analog) işaret. b) örnek dizisi

Örnek dizisi değerlerinin sadece belirli değerleri alabildiğini göstermek üzere yatay ızgara çizgileri de gösterilmiştir. Sadece örnekleme anlarındaki değerlerin bilindiğini göstermek üzere de dikey çubuklar çizilmiş, araları boş bırakılmıştır.

Birçok uygulamada sayısal işaretlerin (örnek dizilerinin) analog işarete çevrilmesi gerekir. Sayısal örnek dizisi analoğa dönüştürüleceği zaman (DAC ile) bu boşlukların da en uygun şekilde doldurulması (sürekli hale getirilmesi) gereklidir. Analog-sayısal dönüştürmedeki olası sorunlar da sayısal-analog dönüşümü için seçenekler düşünülürken anlaşılır. İki noktadan sonsuz sayıda farklı eğri geçebilir. Hangisinin orijinal eğri ile aynı olduğu ya da hatasının en az olduğu garanti edilebilir? Nyquist–Shannon teoreminin bu konuda söyleyeceği bir şey var; "*Eşit aralıklarla örneklenmiş diziden orijinal analog işaretin doğru olarak oluşturulabilmesi için örnekleme frekansının işaretin içindeki en*

yüksek frekanslı bileşenin frekansının iki katından yüksek olması gerekir." Yani, f_m işaretin içindeki en yüksek frekans olmak üzere

$$f_{clk} > 2f_m \tag{1.1}$$

şartı sağlanıyor ise, orijinal işaret teorik olarak yeniden oluşturulabilir demektedir. İşaret bir bantgeçiren (bandpass) işaret ise (yani en düşük frekans 0 değilse), bant ile ilgili bazı parametreleri saklayıp kullanarak, daha düşük frekansta örneklenmiş diziden orijinal analog işaret üretmek mümkündür. Buradaki şart ise

$$f_{clk} > 2BW_x \tag{1.2}$$

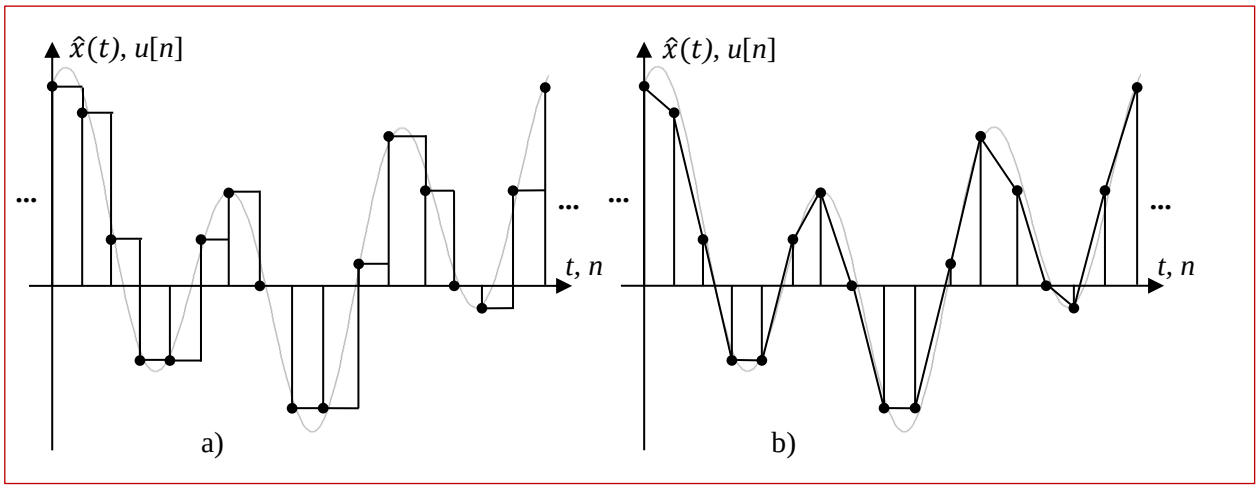
dir. Genel olarak, haberleşmedeki hedef orijinal işareti yeniden oluşturmak değil, işaret ile taşınan verinin doğru olarak elde edilmesidir, ki bu durumda işareti yeniden oluşturmayacağımız için bant ile ilgili parametreleri saklamak çoğu zaman gereksizdir. Buradaki BW_x bantgenişliği işareti oluşturan (Fourier Dönüşümü konusunda gördüğümüz) sinüs işaretlerinin en yüksek frekanslı ile en düşük frekanslı (sıfıra en yakın) olanların frekans farkıdır. 3dB bantgenişliği değil, mutlak bantgenişliğidir. $2BW_x$ değerine Nyquist frekansı $f_{Nyquist}$ denmektedir. Denklemler (1.1) ve (1.2)'de eşitlik olmadığına dikkat ediniz. Bu durumda teorem ile ilgili önceki cümlemizi değiştirelim, ve "*Eşit aralıklarla örneklenmiş diziden orijinal analog işaretin doğru olarak oluşturulabilmesi için örnekleme frekansının işaretin bantgenişliğinin iki katından yüksek olması gerekir*" yapalım. Buradaki mekaniği daha sonra örneklerle anlatacağız.

Analog işaretleri sayısallaştırma aşamasında en önemli beklentimiz bu sayısal dizilerin analog işareti yeterince iyi/yakın temsil edebilmesidir. Yani sayısal işareti hiçbir işlem yapmadan tekrar analog işarete dönüştürdüğümüzde orijinal analog işarete ne kadar yakın olduğu önemlidir. Orijinal analog işaret $x(t)$ ve ADC-DAC çiftinden üretilmiş analog işaret $\hat{x}(t)$ olmak üzere, $|x(t) - \hat{x}(t)|$ hatasının iki kaynağı vardır. Birincisini daha önce görmüştük; nicemleme hatası. Bu hata hesaplanabilir ve temsil bit sayısını arttırarak (daha fazla tasarım maliyeti) istenildiği kadar küçük yapılabilir. Hatta, işaret istatistiklerine göre tasarlanmış doğrusal olmayan bir analog süzgeç yardımıyla, her örnekte olmasa da ortalamada daha düşük hata oranları elde edilebilir. İkinci hata kaynağı ise (1.2)'in sağlanıp sağlanmadığıyla ilgilidir. Sağlanıyorsa, en azından teorik olarak analog işaret tekrar aynen oluşturulabilir. Sağlanmıyor ise, ki birazdan açıklama getireceğiz, işaret tekrar oluşturulamaz, önemli sayılabilecek hatalar oluşabilir, telafisi de yoktur.

DAC ardışıl gelen sayısal diziden analog işareti üretmek için bir aradeğerleme yapar, yani işaretin değerinin bilinmediği anlardaki işaret değerini komşu örneklerden hesaplar. Şekil 1.9 iki adet aradeğerleme yöntemi göstermektedir. Birincisi, aradaki değerlerin son örnek değeri ile aynı olduğu varsayımını yapar, yani örnek değerini tutar (zero order interpolation/hold). İkincisi ise son ve sıradaki örneklerin uzaklığa göre doğrusal ağırlıklı ortalamasını üretir, sonuçta iki örnek değerini gösteren noktaları düz çizgi ile birleştiren bir işaret ortaya çıkar. İkinci yöntemdeki hataların daha az olduğunu görebiliyoruz. Tabi daha fazla sayıda önceki ve sonraki örnek değerlerini de kullanarak polinom aradeğerlemesi yapılırsa orijinale daha yakın sonuçların elde edilebileceği yorumu üretilebilir. En doğrusu ise her aradeğer için tüm örneklerin kullanılarak *sinc* ağırlıkları ile aradeğerleme yapmaktır. Şekil 1.10'da açıklanan sinc aradeğerlemesi

$$\hat{x}(t) = \sum_{n=-\infty}^{\infty} u[n] sinc((t - nT_s)/T_s) \qquad (1.3)$$

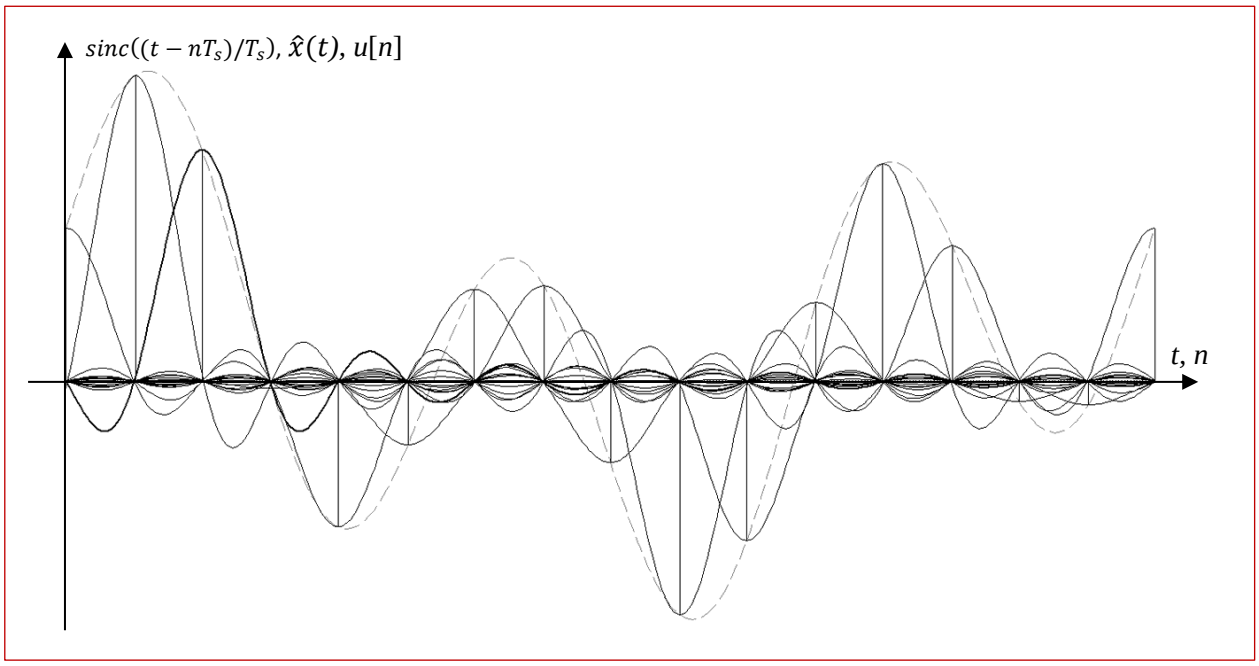
ile ifade edilir.



Şekil 1.9. Aradeğerleme yöntemi örnekleri a) sıfır dereceli aradeğerleme (tutma) (zero order interpolation (hold)) b) doğrusal aradeğerleme (bir dereceli: first order). Anlaşılması için asıl örnek değerleri de gösterilmiştir.

Şekil 1.10 şöyle açıklanabilir; Her noktadan sonsuz adet *sinc* fonksiyonu geçmektedir. Bunlar frekans karakteristiği aynı ama zamanda kaydırılmış ve genliği değiştirilmiştir. Her örnek noktasına yerleştirilen *sinc*'lerin merkez tepe noktaları örnek değerindedir. *Sinc*'lerin zaman eksenindeki sıfır geçişleri diğer örnek noktalarındadır. Bu durumda tüm *sinc*'ler (sonsuz adet) toplandığında orijinal işaret elde edilmektedir.

Bu yaklaşımda üstesinden gelinemeyecek birkaç zorluk bulunmaktadır. Dizinin tüm örnek değerleri (sonsuz adet) bilinmek zorundadır. Bu ise sayısal devreler için sonsuz hafıza ve gerçek zamanlı olmamak demektir. Sayısal devrelerde hiçbir şey sonsuz olamaz. *Sinc*'lerin merkez noktasından uzaklaştıkça değerlerinin ihmal edilebileceği (ve oluşan hatanın kabul edildiği) bir yaklaşımda bile analog çıkış işareti belirli bir gecikmeyle hesaplanabilir. Ayrıca Şekil 1.10'da çizdiğimiz *sinc*'ler zaten analog işaretlerdir, yani analog kısımda üretilmesi, toplamanın da analog olarak yapılması gerekmektedir (!?). Açıklanan nedenlerle, bu yaklaşım pratikte hiç kullanıl(a)maz. Onun yerine, Şekil 1.9a'daki sıfır dereceli tutma yöntemi ve ardından, farkında olarak üretilmiş istenmeyen frekans bileşenlerini atmak için, bir IIR (infinite impulse response) süzgeç kullanılır. Zaten DAC'lar için sıfır dereceli tutma yöntemi en kolayıdır.



Şekil 1.10. *Sinc* aradeğerlemesi. Karmaşıklığı önlemek için sadece görünen zaman aralığındaki 17 örneği merkez alan *sinc*'ler, 1 tanesi koyu olmak üzere, çizilmiştir. Toplam işaret kesikli çizgi ile gösterilmiştir.

Bir ara özet yapalım (bizim bakış açımızla);

Analog-sayısal dönüştürme : her aralığında sonsuz voltaj olasılığı olan sonlu sayıda voltaj aralıklarını sonlu sayıdaki kod ile temsil etmektir. Kod genellikle ikili sayı sisteminden olur.

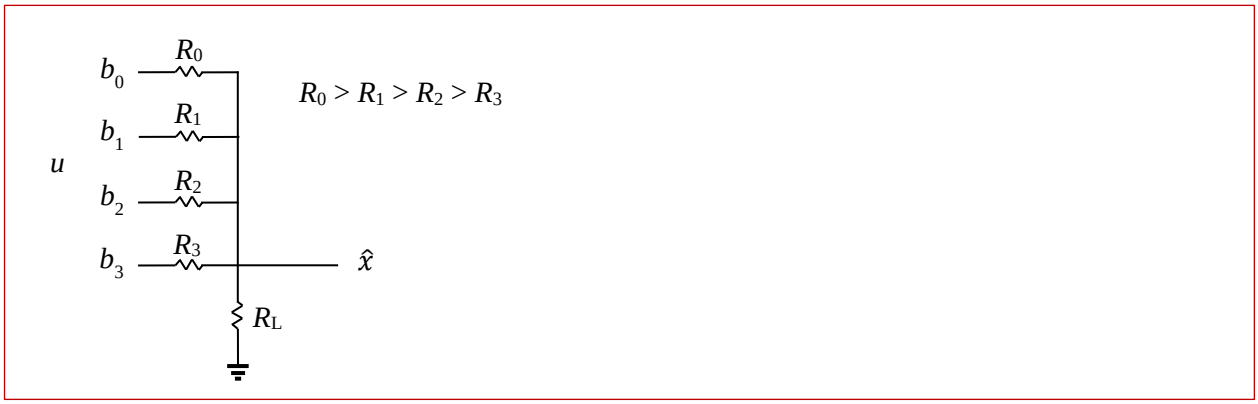
Sayısal-analog dönüştürme : Sonlu sayıdaki kodları voltaj değerlerine dönüştürmektir.

Voltaj'dan koda ve koddan voltaja dönüşüm için elektronik devreleri kullandığımıza göre dönüştürme için bir süre gereklidir. Bu süreye dönüşüm süresi (conversion-time) denir. Gerekli olmasa da her dönüşüm aynı sürede gerçekleşecek şekilde tasarım yapılır, saniyede kaç dönüşüm yapıldığı (Sa/s) verilir.

Bazı dönüşüm sistemlerinde Sa/s belirgin değildir. Örneğin, analog değerin frekans ile gösterildiği dönüşümlerde veya daha sonra göreceğimiz *delta-sigma modülasyonu* olarak adlandırılan dönüşüm yönteminde gerçekten bağımsız bir kod üretilmiyor sürekli bir ikili akış veriliyor olabilir. Bunlar daha çok düşük hızlı yöntemler olduğundan haberleşme sistemlerinde kullanımı azdır, o nedenle çok detaya girmeyeceğiz.

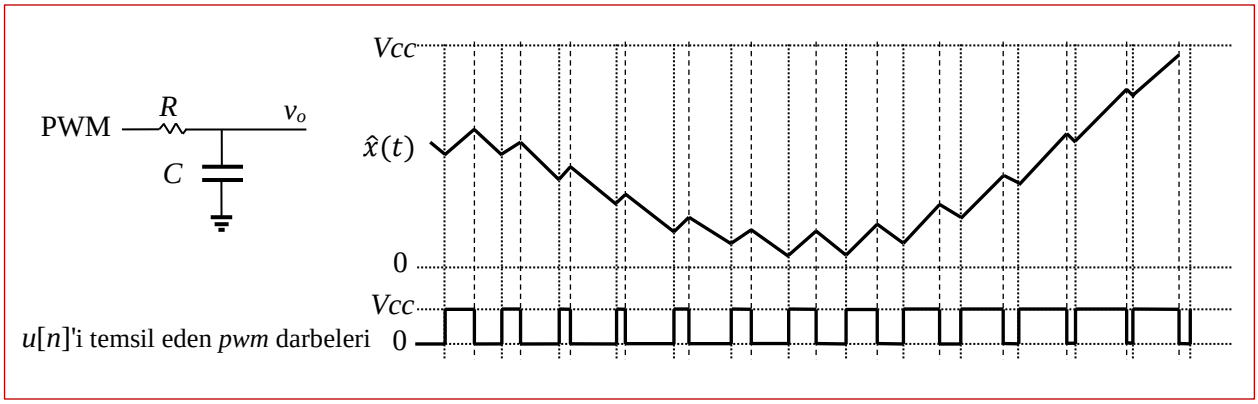
Şekil 1.11 en anlaşılır ve kolay tasarlanabilir sayısal-analog çeviriciyi özetlemektedir. Örneğimizde sayısal giriş 4-bittir. Her bit girişi b_i'nin sayısal 0 ve 1 değerlerini temsil etmek üzere 0 V veya Vcc Volt değerlerini aldığını varsayalım. Direnç değerlerinin $R_0 > R_1 > R_2 > R_3$ şartını sağlayacak şekilde seçilmesi durumunda b_0 değerinin vo çıkışında en az, b_3 değerinin de en fazla etki yapacağını görebiliriz. Yani girişteki 4 bitin 16 farklı değerine karşılık çıkışta (0-Vcc) aralığında16 farklı voltaj değeri göreceğiz. Bu voltaj değerleri aralıklarının düzgün (uniform) olabilmesi (yani dönüşümün düzgün olabilmesi) için direnç değerlerinin her durum için geçecek akımlar da hesaba katılarak hesaplanması gerekir. Ama yöntem anlaşılmıştır, bu hesapları tasarımı yapacak kişiye bırakalım. Çıkış

voltajının farklı bir aralığa getirilebilmesi için (örneğin (-5V, +5V)) ayrıca bir kazanç katı koymak gerekebilir.



Şekil 1.11. Ağırlıklı toplayıcı yöntemi ile sayısal-analog dönüştürücü.

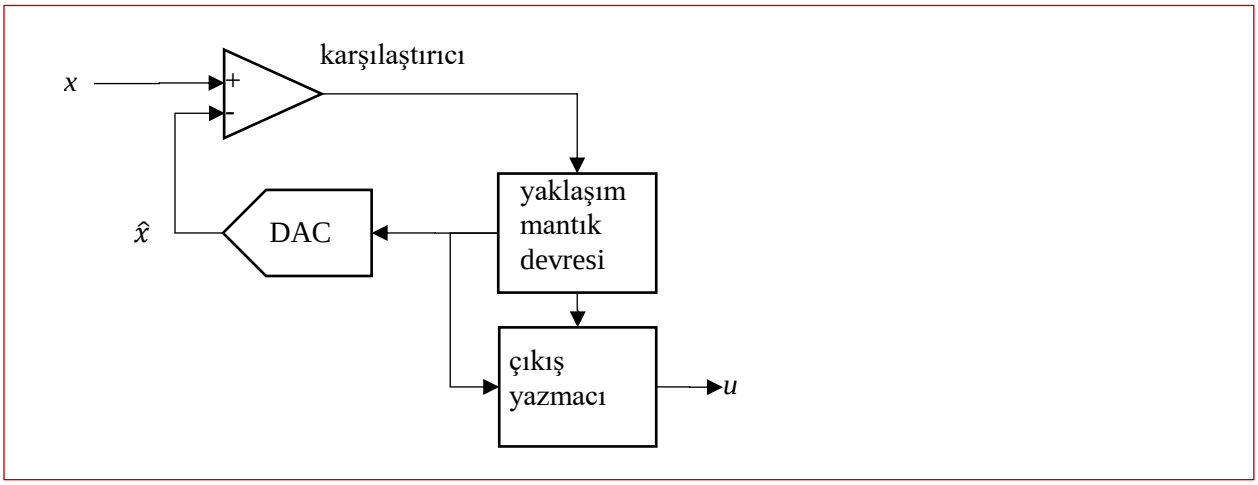
Bir başka sayısal-analog çevrim yöntemi ise sayısal değerlerin önce aynı frekanslı ama farklı görev döngülü dikdörtgen dalga akışına dönüştürmek ve analog bir devre ile bu işaretin daha geniş zamanlı ortalamasını üretmek olabilir. Bu şekilde dikdörtgen dalga üretimine *darbe genişliği modülasyonu* (PWM : pulse width modulation) ismi verilir. *B* bitlik bir DAC tasarlanacaksa dikdörtgen dalganın peryodu en az 2^B eşit parçaya ayrılır. Analoga çevrilecek sayısal değere karşı gelen daha önce bahsettiğimiz *termometre kodu* sayısal çıkışa seri olarak verilir. Örneğin en küçük değeri temsil etmek üzere 000...0, en büyük değeri temsil etmek üzere 111...1 dizileri tek bitlik çıkışa verilir. 0 ve 1 sayısal değerlere karşı gelen işaret voltajları da sırasıyla 0 V ve Vcc ise, ortalama hesaplayıcı devrenin çıkışından da (0, Vcc) aralığında bir voltaj görülür. En basit ortalama değer alıcı devre ise RC (alçakgeçiren süzgeç) devresidir. Tabi sadece bir değer üretmek için bile en az *B* saat darbesi gerekir. Yani, bu DAC yavaştır. Ama, PWM üretebilecek kadar karmaşık sayısal sistemlerde ses frekanslarında (yani düşük frekans) işaretler üretebilmek için kolaylıkla ve ucuz şekilde kullanılabilir. Böyle ihtiyaçlarda yüksek performanslı bir DAC için para harcamaya gerek yoktur.



Şekil 1.12. *pwm* girişine karşılık RC devresindeki kapasitörün dolma ve boşalması.

Şekil 1.12'deki sistemin anlaşılmasını kolaylaştırmak için pwm üretim oranı analog işarete göre biraz düşük seçilmiştir. Böylelikle çıkış işaretinin inişli çıkışlı yapısı (ripple) görülebilmektedir.

Şimdi de ADC için daha ucuz ve dolayısıyla daha düşük hızlarda çevrim yapan bir yönteme değinelim. Yöntem bir DAC kullanarak geri besleme yapıyor (Şekil 1.13). Analog karşılığı üretilecek sayısal örneğin en önemliden en önemsize doğru her biti sırasıyla 1 yapılıp DAC ile elde edilen analog işaret girdi analog işaretle karşılaştırılıyor. Eğer üretilen işaret girdiden büyük ise ilgili bit tekrar 0 yapılıp sonraki (bir düşük değerdeki) bit için aynı şey tekrarlanıyor. Böylece üretilen analog işaret girdi işaretine giderek yaklaşıyor. Bu nedenle yöntemin adı *ardışıl yaklaşım* (succesive approximation).



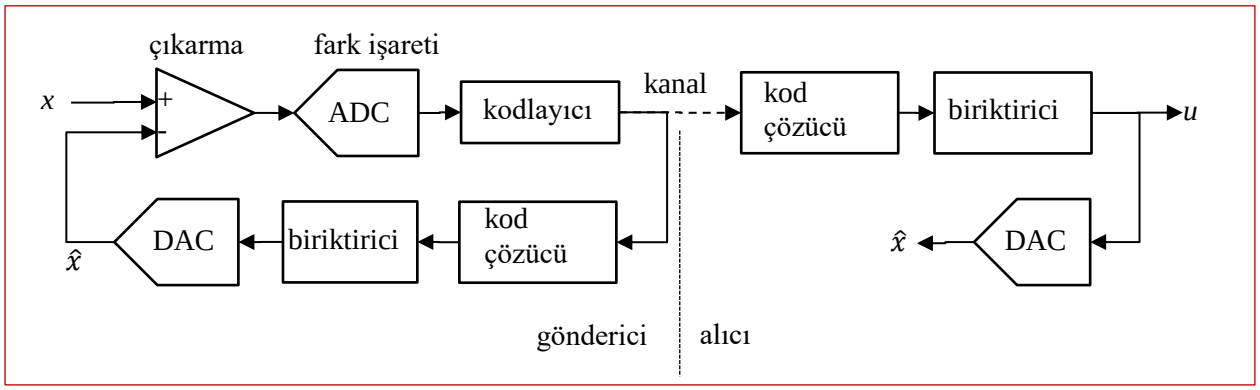
Şekil 1.13. Ardışıl yaklaşım ADC prensip şeması.

Ardışıl yaklaşım ADC'leri en az çıktı bit sayısı kadar saat darbesi gerektirdiği için, örneğin 12 bitlik bir çevrim için 12+ saat darbesi gerektirdiğinden, daha önce bahsedilen termometre kodlu paralel çevrim ADC'lerinden (bazen flash-ADC diye anılır) daha yavaş olması beklenir. Bu yönteme benzer daha basit bir yaklaşım ise, Şekil 1.13'teki *yaklaşım mantık devresi* yerine 0'dan 2^B-1'e kadar sayan bir sayıcı kullanmaktır. DAC çıkışı giriş voltajını geçtiği anda sayıcı değeri çıkış yazmacına aktarılır, böylece çevrim gerçekleştirilmiş olur. Ancak bu yöntemde çevrim süresi sabit değildir, 1 saat darbesinden 2^B saat darbesine kadar değişebilir. Ortalamada 2^{B-1} saat darbesi ile bu yöntem bahsedilen diğer yöntemlere göre oldukça yavaştır.

Şekil 1.12'de verdiğimiz pwm-DAC örneği dışında bahsettiğimiz ADC ve DAC yöntemleri örnek tabanlı yaklaşımlar olup daha önceki girdilerden bağımsız (hafızasız) çalışmaları hedeflenir. İşaretin (analog yada sayısal) sadece son değeri değil de önceki değerlerinin de hasaba katıldığı hafızalı yöntemler de basitliklerinden ve kolayca genişletilebilir olduklarından dolayı yaygındırlar. Örneğin şimdi bahsedeceğimiz delta-modülasyonu yöntemi farksal kodlamanın (differential coding) bir türü olup önceki değer(ler) ile yeni/son değerin farkının kodlanmasına dayanır. Buradan beklenen avantaj, girdi işaretinin örnekler arasında çok fazla değişmemesi, dolayısıyla farkın küçük olması beklendiğinden daha az bit sayısı ile daha yüksek çözünürlük elde edilebilme ihtimalidir. Cümle çok fazla ardışıl ihtimal/beklenti içeriyor görünebilir, ama ADC/DAC'ın üzerinde çalışacağı işaret karakteristiklerinin önceden bilinmesi ile bu yöntemin kullanışlılığı belirgin olur ve oldukça avantajlı da olabilir.

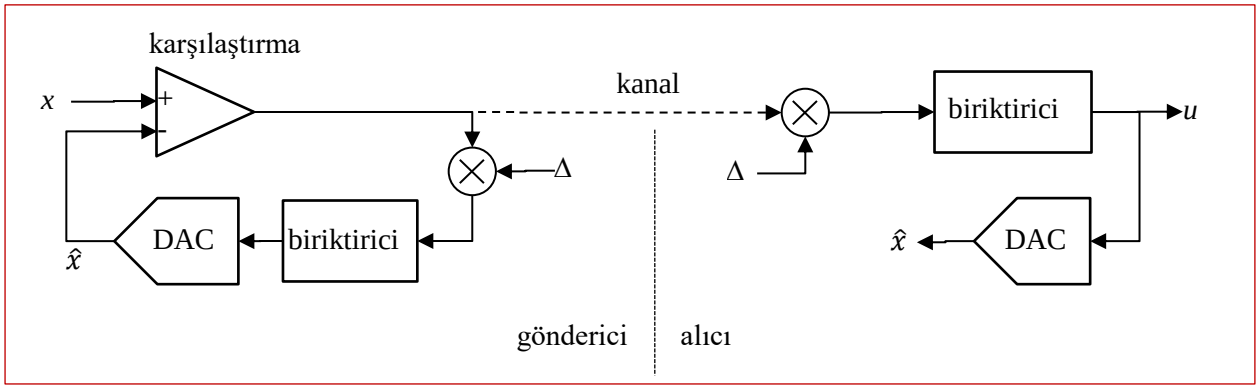
Şekil 1.14 fark kodlaması kullanan bir iletişim sistemini özetliyor. Burada analog işaretin kendisi değil bir önceki örnekten farkı sayısala çevriliyor. Bu sayısal fark alıcı tarafında önceki sayısal örneğin üzerine ekleniyor ve gerekiyorsa analoga çevriliyor. Yöntem sadece farkları sayısala çevirdiği için

hem ADC bit sayısı hem de kanaldan iletilen bit sayısı her örneğin bağımsız (hafıza olmadan) gönderildiği yöntemlerden düşük oluyor.



Şekil 1.14. Fark kodlaması (differential coding).

Gönderici tarafındaki işaret zaten sayısal ise ADC ve DAC gerekmeden bu sayısal veri fark gönderme yöntemiyle alıcıya iletilebilir. Bu durumdaki ismi de farksal kodlama oluyor. Farksal kodlamanın bir avantajının olabilmesi için ardışıl sayısal değerlerin arasındaki farkın küçük olması gerekir. İşaret analog ise bu farkın küçük olması için ardışıl alınan ölçümler arasının yeterince küçük olması gerekir (yani saat frekansının yüksek olması). Hatta saat frekansı yeterince yüksek olabilirse (ya da analog işaretin değişimi yeterince yavaş ise) gönderilen fark 1 bitle temsil edilebilir. Bu durumda ADC+kodlayıcı yerine sadece 1 bit kodlayıcı yeterlidir. Böyle bir yaklaşım Şekil 1.15'te gösterilmiştir.

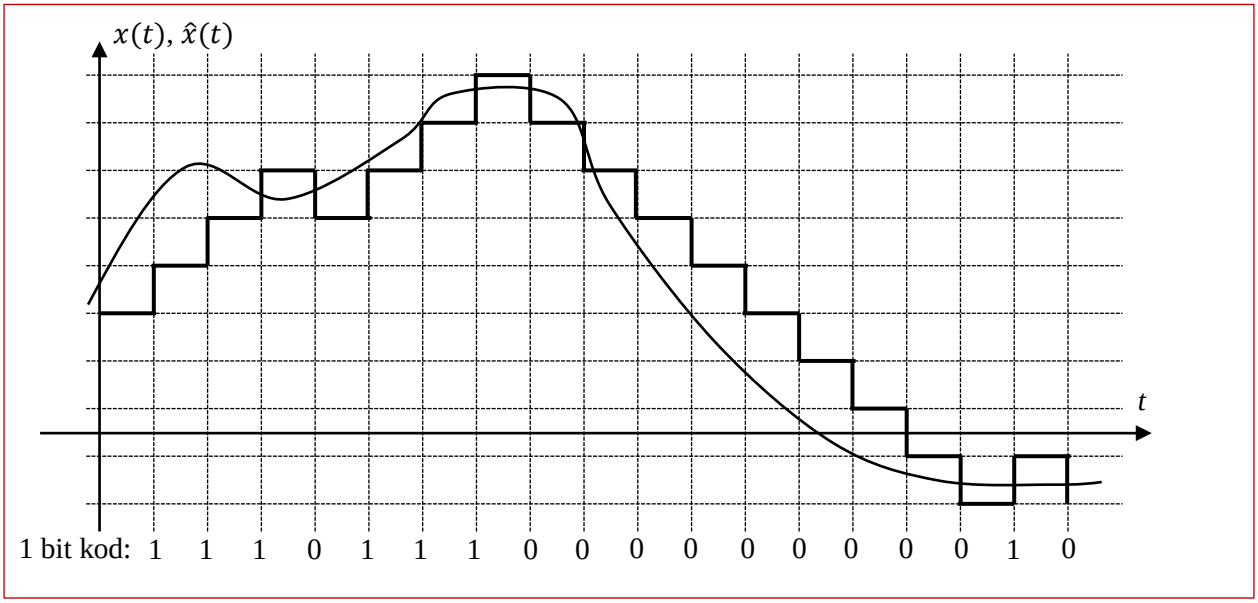


Şekil 1.15. Delta-modülatör.

Şekil 1.15'teki karşılaştırıcının çıkışı x ve $\hat{x}$ karşılaştırma sonucuna göre +1 yada -1'dir. Δ ise x ve $\hat{x}$ arasındaki farkı kapatıcı yönde atılacak adımın büyüklüğüdür. Fark 1 Δ'dan büyükse ardışıl birden çok adım gerekebilir. Ancak biraz mantık yürüterek adım sayısı azaltılabilir. Örneğin + yönde atılacak bir adımdan sonra yine + yönde bir adım atılması gerektiğinde adım büyüklüğü (Δ) de büyütülebilir. Böylece hedefe daha hızlı varılır (yada hedef geçilir). Yani Δ dinamik olarak belirlenebilir. Tabi ki aynı yaklaşımın alıcıda da gerçekleştirilmesi gerekir.

Δ-modülatör yönteminde kanaldan iletilen her bir bit iletilmek istenen sayıya doğru atılan bir adımı temsil eder, hiçbir zaman sayının kendisini temsil etmez. Ancak yeterince hızlı adımlar atılırsa

(analog işaretin değişim hızına göre yüksek bir saat hızı) $\hat{x}$ her zaman x'in yakınlarında bir yerlerdedir. Böylece hiç ADC kullanmadan ADC işlevini elde ederiz. Biriktirici de (accumulator) girişteki analog işaretin anlık değerini temsil eden asıl sayısal değerin yakınlarında bir değere sahiptir.

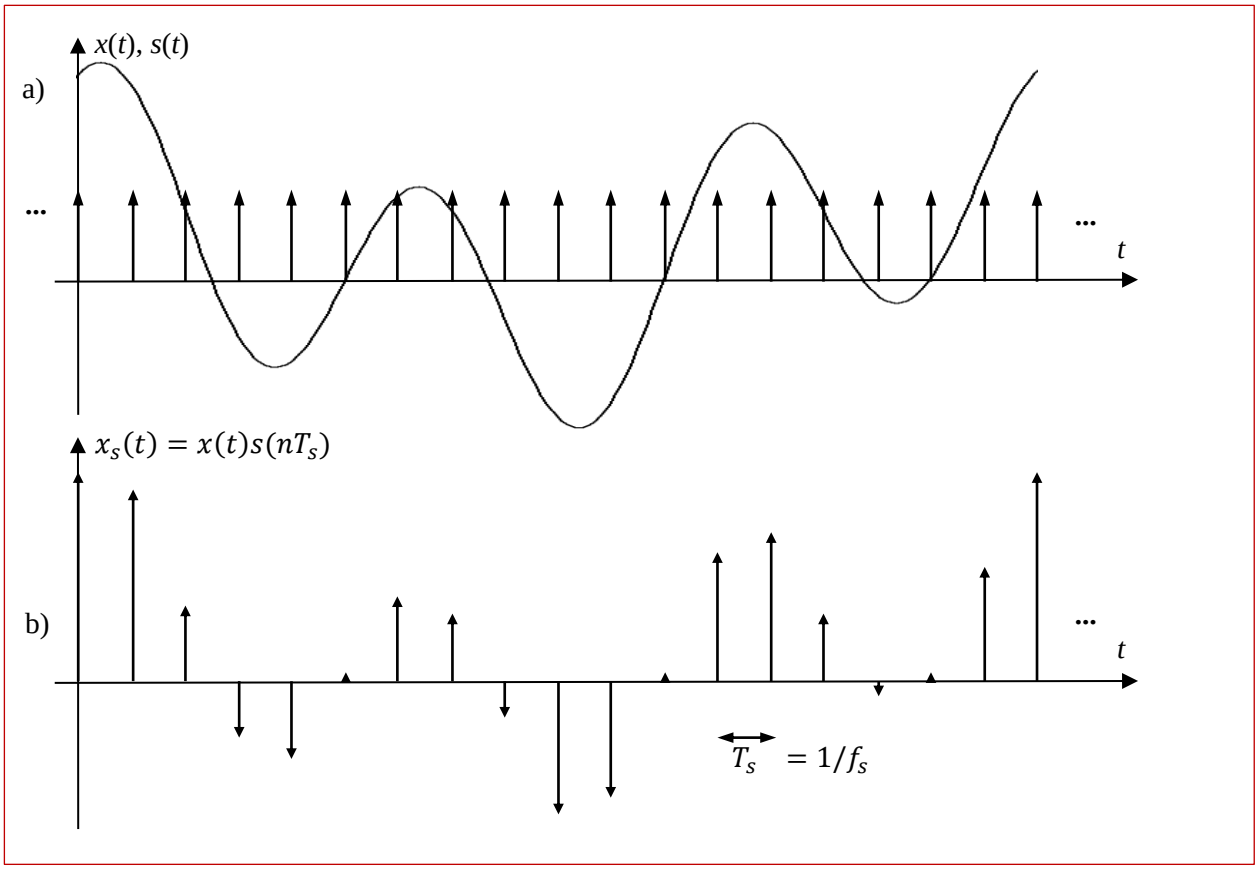


Şekil 1.16. Δ-*modulasyon* prensibini anlatan örnek.

Şekil 1.16'daki sürekli işaret $x(t)$ analog girdi işaretidir. Basamaklı olarak görülen ise $x(t)$'yi takip etmek isteyen ve $x(t)$ hızlı değiştiğinde pek başaramayan (farkın büyüdüğü) biriktirici değeridir ki analog karşılığı $\hat{x}(t)$'dir. En altta görülen 1 bitlik kodlar ise kanaldan gönderilen sayısal değerlerdir. Burada, işaret alıcı tarafında tekrar analoga çevrilecekse, 1 değeri önceki analog değere göre Δ kadarlık artımı, 0 ise Δ kadarlık azaltımı temsil etmektedir. $x(t)$'deki en büyük değişimler B bit ile ifade edilebiliyor ise doğru bir temsil için en az B bitin üretilip gönderilmesi gerekir. Yani Δ-modülatör hem ADC hem de serileştirme işlevini yerine getirmektedir, bedeli ise saat frekansının aynı $x(t)$ işaretinde B kat yüksek olmasının gerekmesidir.

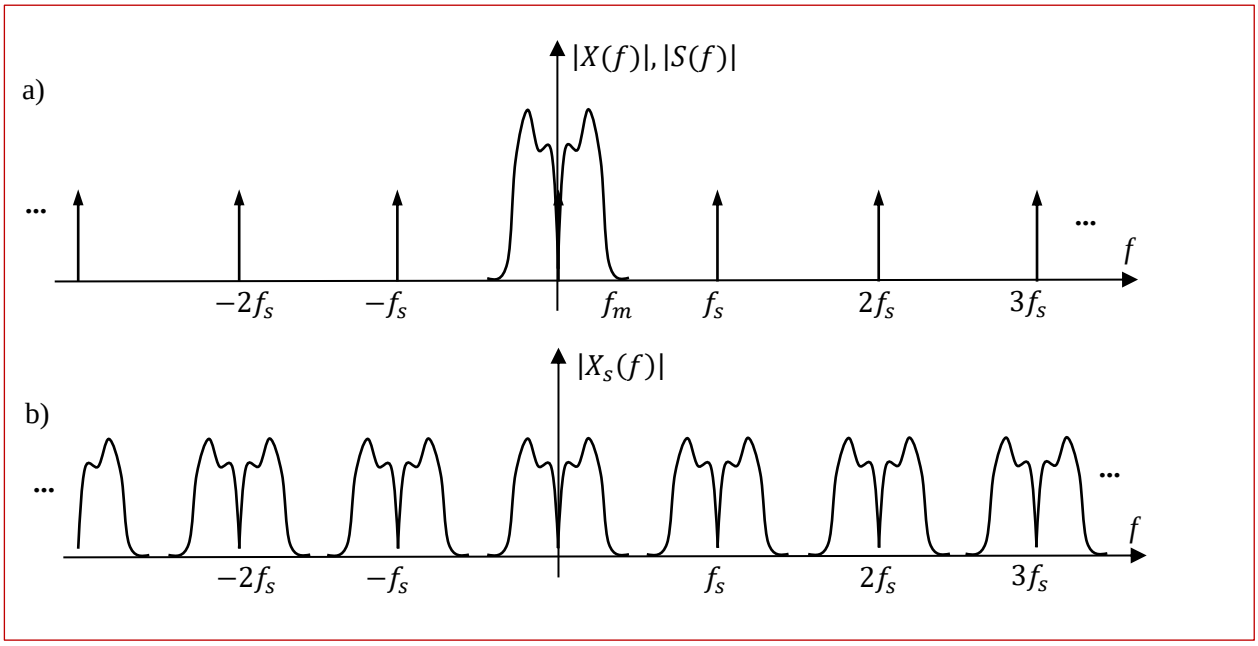
1.1 Örneklenmiş İşaretin Frekans Karakteristiği

Analogtan sayısala çevirme işleminin analog işaretin değerini belirli aralıklarla ölçüp, ölçüm sonucunun nicemlenerek 2'li kodla temsil edilmesi olarak gördük. Ölçüm aralıklarının T_s olduğunu kabul edelim. Bu durumda, ideal olarak, nicemlenip ölçülecek değerler girdi $x(t)$ analog işaretinin nT_s (n=0, ±1, ±2, ...) anlarındaki değerleridir, yani $x(nT_s)$'dir. Bunu, $s(nT_s) = \sum_{n=-\infty}^{\infty} \delta(nT_s)$ birim dürtü treni (**Error! Reference source not found.**) olmak üzere $x_s(t) = x(t)s(nT_s)$ ile özetleyebiliriz (Şekil 1.17).



Şekil 1.17. a) Analog işaret ve birim dürtü treni. b) örneklenmiş işaret treni

Peki Şekil 1.17b'deki örneklenmiş işaret treninin tayfı nasıldır? Kolayca hesaplayamayacağımızı düşünebiliriz. Ama önceki bölümlerde gördüğümüz Fourier Dönüşümü özeliklerini kullanarak ve incelediğimiz örneklerden faydalanarak neredeyse hiç hesap yapmadan çizebiliriz. Varsayımlarımız $x(t)$ işaretinin bantsınırlı olduğu ve örnekleme frekansının $x(t)$'nin içindeki en yüksek frekansın (f_m diyelim) iki katından yüksek olduğudur (Nyquist şartı $f_s > 2f_m$).



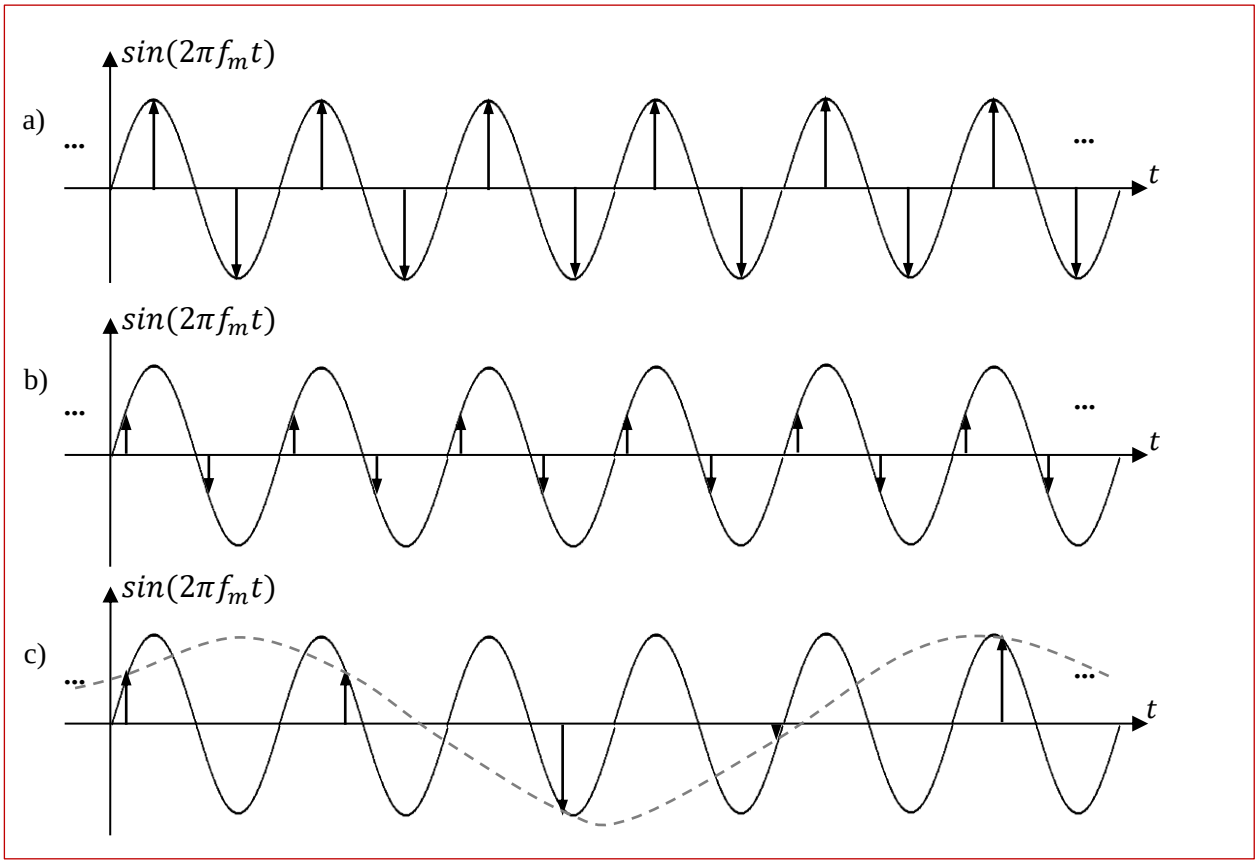
Şekil 1.18. a) Bantsınırlı analog işaretin tayfı ve birim dürtü treni tayfı. b) örneklenmiş işaret treninin tayfı.

Zaman alanındaki $x_s(t) = x(t)s(t)$ çarpımı frekans alanındaki

$$X_s(f) = X(f) * S(f) = \int X(\Omega)S(\Omega - f)d\Omega$$

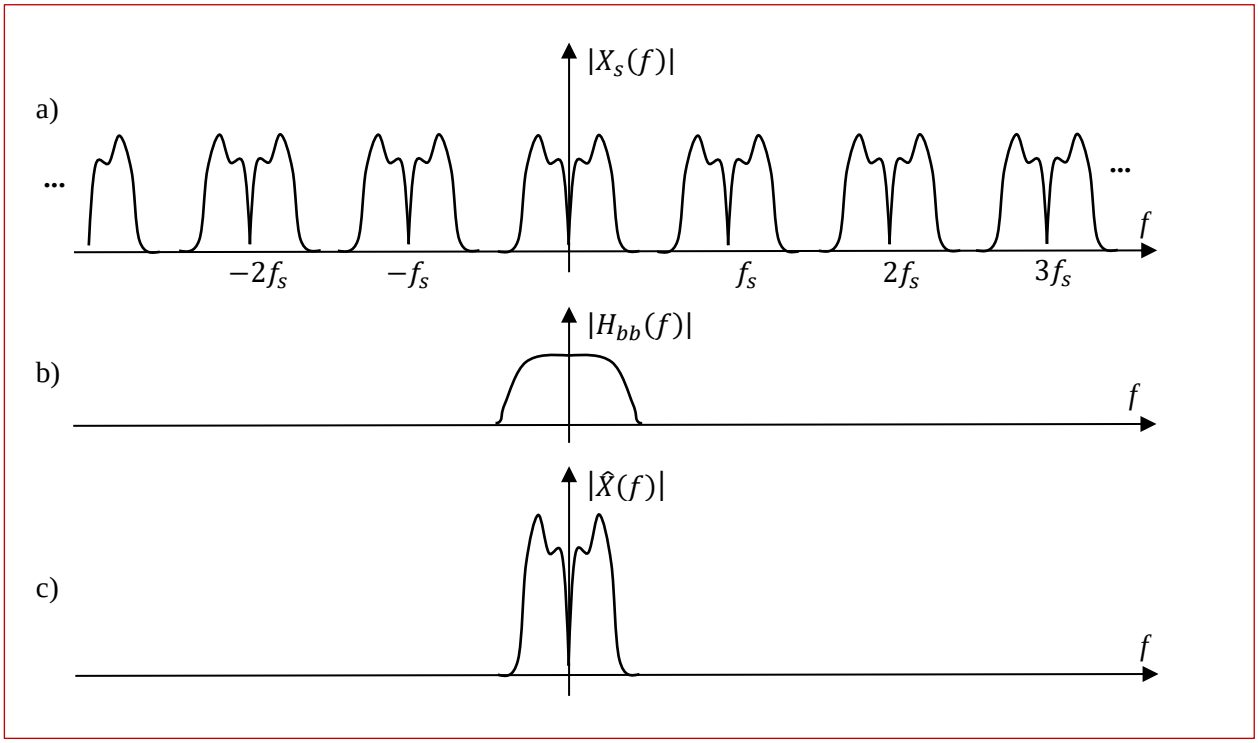
evrişimine denktir. Frekans alanındaki birim dürtü işaretinin eleme (sifting) özeliğinden dolayı $X_s(f)$ tayfının kopyaları Şekil 1.18b'deki gibi nf_s frekans merkezlerine yerleşirler. Bu durumu anlamak için aslında Fourier Dönüşümünün kipleme özeliğinden de faydalanabiliriz. Sıfır frekansının her iki tarafındaki birim dürtüler aslında birer sinüs (kosinüs) işaretini oluştururlar ($-kf_s$ ile $+kf_s$). **Error! Reference source not found.**'da gösterdiğimiz üzere kipleme özeliği temelbanttaki bir işaretin tayfını $-kf_s$ ve $+kf_s$ frekanslarına kopyalıyor idi. Burada da tüm birim dürtüler için işlemi yaptığımızda $|X_s(f)|$ tayfının oluştuğunu görebiliriz.

Şekil 1.18b'yi inceleyerek Nyquist $f_s > 2f_m$ şartının nereden geldiğini anlayabiliriz. Eğer $f_s < 2f_m$ olsaydı bu alt-tayfların bazı kısımları birbirinin üzerine eklenirdi, ki buna *örtüşme* diyoruz. Bu durum da örtüşen bileşenleri geri kurtarılamayacak şekilde bozacağı için bozulmuş bileşenlerden de asıl (sürekli analog) işaret geri elde edilemezdi. $f_s = 2f_m$ eşitlik durumunun da kabul edilemeyeceğini Şekil 1.19'daki çizimlerden anlayabiliriz. Burada f_m frekanslı bir sinüs işareti aynı $f_s = 2f_m$ örnekleme frekansıyla örnekleniyor ancak örneklemeye farklı zamanlarda başlanmış. Örneklerin aynı büyüklükte olmayacağı açıkça görülüyor. Hatta tüm örnekler sıfır dahi olabilir. $f_s < 2f_m$ ile örneklenmiş bir işaretin aslında farklı bir sinüs işaretini temsil edeceği de Şekil 1.19c şeklinden görülüyor.



Şekil 1.19. f_m frekanslı bir sinüs işaretinin $f_s > 2f_m$ örnekleme frekansı ile örneklenmesi gerektiğini gösteren çizimler. a) $f_s = 2f_m$ ile örneklenmesi b) $f_s = 2f_m$ ile farklı fazda örneklemesi c) $f_s < 2f_m$ frekansıyla örneklenmesi.

Tabanbant bir işaret uygun şekilde örneklendiği zaman işaret kopyalarının Şekil 1.18b'deki gibi nf_s frekanslarını merkez alarak sonsuz kez tekrar ettiğini gördük. Diyelim ki bu işareti ADC ile sayısala çevirdik ve sayısal işaret işleme yöntemleriyle bunun üzerinde işlemler yaptık (ya da hiçbirşey yapmadık). Analog işareti tekrar geri elde etmek için ne yapılabilir. Kuramsal olarak, frekans alanındaki bu sonsuz kopyadan tabanbantta olan kopyayı bir tabanbant süzgeci ile çıkarabiliriz (Şekil 1.20), böylece diğer bütün kopyalar kaybolur. Kuramsal olarak diyoruz, çünkü ne örneklenmiş işaret dediğimiz $X_s(f)$'i ($x_s(t)$'yi) ideal olarak elde edebiliriz (çünkü dirac-delta fonksiyonu gerçek değildir) ne de Şekil 1.20b'dekinin yerine olması gereken ideal süzgeci (ideal süzgeç gerçek değildir) gerçekleyebiliriz. Ayrıca, gerçekten de bantsınırlı bir işaret yoktur. Bunun sonucunda da $x(t)$'yi tam olarak elde edemeyiz, o nedenle $\hat{x}(t)$ diyoruz.



Şekil 1.20. Kuramsal olarak örneklenmiş tabanbant işaretten orijinal işaretin geri elde edilişi a) Örneklenmiş işaretin tayfı. b) tabanbant süzgeç. c) Süzgeç çıkışı

Şekil 1.20'yi biraz daha incelediğimizde aslında tabanbant bir süzgeç yerine bantgeçiren bir süzgeç kullandığımızda kf_s (k=0, 1, ..., ∞) merkezli bir kopyayı elde edebileceğimizi görürüz. k=0 durumu Şekil 1.20'de gösterilendir. Bu işlemle yaptığımız, Fourier Dönüşümünün kipleme özeliği konusunda gördüğümüz ve **Error! Reference source not found.**'da anlatılana benzer bir sonuç doğurmakta. Yani aslında tabanbant olan bir işaret bir bantgeçiren işarete dönüşmekte. Benzerliğinden dolayı bu işleme frekans yukarı taşıma (frequency upconversion) ismi verilir. Peki frekans aşağı taşıma (frequency downconversion) diye birşey var mıdır? Elbette. f_x merkezli bir bantgeçiren işareti $f_y < f_x$ olmak üzere f_y frekansına taşımaya frekans aşağı taşıma ismi verilir. Genel olarak f_x merkezli bir bantgeçiren işaret frekans tayfındaki herhangi başka bir f_y merkezine taşınabilir. Özellikle kf_s (k≠0) merkezli bir işaretin tabanbanta (k=0) taşınması oldukça kullanışlıdır. Bu özelik sayesinde ve Nyquist kriterini anlatırken eklediğimiz $f_s > 2BW_x$ şartına da uyarak, yüksek frekanslardaki bir bantgeçiren işareti ADC ile sayısala çevirirken aynı zamanda tabanbanta indiriyoruz. Ki bu yaklaşım pratikte çokça kullanılmaktadır. Yani, bantgeçiren bir işaretin kipleme yöntemi ile tabanbanta indirilmesi ve ardından ADC kullanılarak sayısallaştırılması ile uygun bir f_s seçilerek işaretin doğrudan ADC ile sayısallaştırılması eşdeğerdir. Her iki yöntemde de istenmeyen frekansların uygun süzgeçler (analog ya da sayısal) ile bastırılması gereklidir.

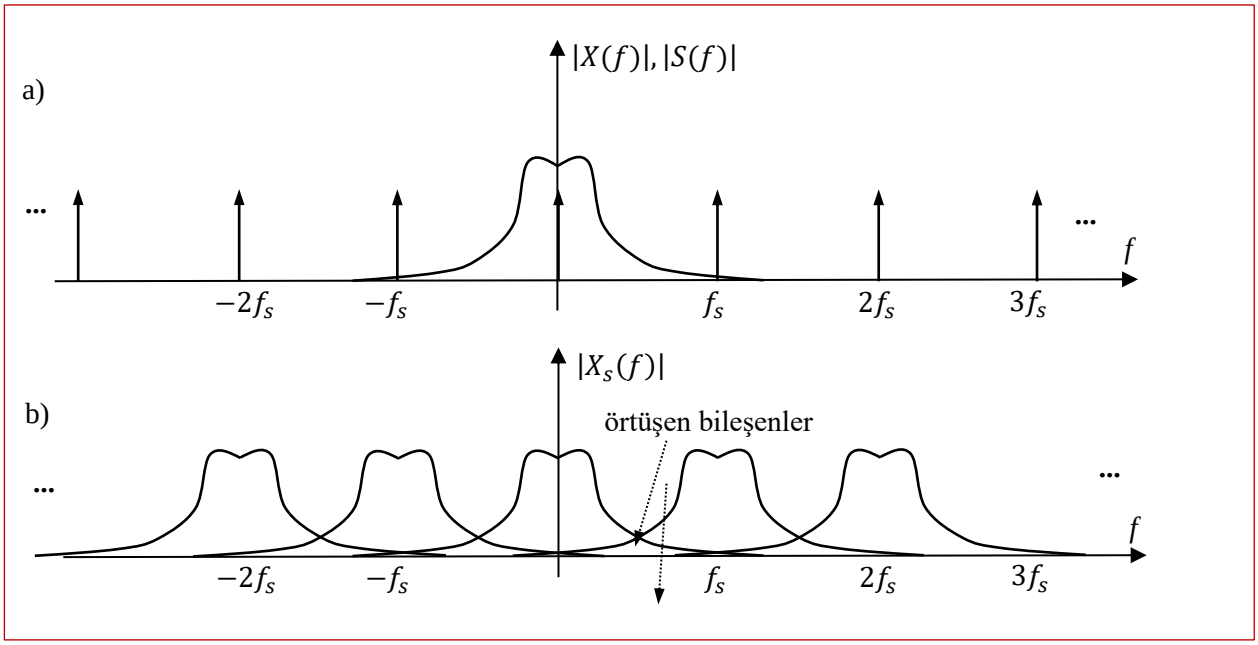
Örnekleme konusuna kipleme bölümlerine geldiğimizde uygun yerlerde yine değineceğiz. Bu bölümde örnekleme işlemini anlatırken atlanmaması gereken konulara değinmeye devam edelim.

Biraz önce gerçekten bantsınırlı bir işaret olamayacağını söylemiştik. Sebep belli, Fourier konusunda bahsetmiştik. Biz işaretleri t=-∞'dan +∞'a var olduğunu varsayarak fonksiyon olarak inceleme imkanına kavuşuyoruz. Pratikte ise bunun mümkün olmadığını biliyoruz. Yani pratikteki tüm işaretlerin en azından bir başlama zamanı var ve bu -∞ değil. Karşılaştığımız tüm işaretler zaman sınırlı. Bu da kuramsal olarak bu işaretlerin frekans tayfının sınırsız olması sonucunu doğuruyor.

Frekans tayfının sınırlı olması için ise işaret zaman alanında t=-∞'dan +∞'a kadar devam ediyor olması gerekiyor. İşaretimizin tayfının sınırlı olmasını engelleyen sebep ne olursa olsun, aynı zamanda örnekleme için geliştirilen kuram ve uygulamaları da engelliyor/sınırlıyor. Ama birçok uygulama için milisaniyeler bile oldukça uzun sürelerdir ve bahsettiğimiz kuramsal etkileri görmezden gelebiliriz. Buna rağmen, işaretimizin bantgenişliği sonludur diyemiyoruz.

Örnek olarak ses işaretini ele alalım. Ses havadaki basınç dalgalarıdır. Ses dalgalarını elektronik yöntemlerle işlemek (kuvvetlendirmek, iletmek vb.) için dönüştürücü/algılayıcılar (mikrofon) kullanırız. Her ne kadar insan kulağı çok yüksek frekanslı sesleri duymasa da dönüştürücüler bunları algılar, hatta elektriksel işarete dönüştükten sonra da üzerine daha da yüksek frekanslı, gürültü diyebileceğimiz işaretler de çeşitli yollarla eklenir. Özet olarak, istenmeyen yüksek frekanslı işaretlerden kurtulmak isteriz. Bu çoğu zaman istek meselesi değil, zorunluluktur.

Şekil 1.21'deki tabanbant tayfına sahip işareti örnekleyeceğimizi ve sayısal olarak işleyeceğimizi varsayalım. Ancak ADC'mizin en yüksek örnekleme hızının işaretimizin en yüksek frekansının iki katından daha düşük olduğunu düşünelim. Yani $f_s > 2BW_x$ şartını sağlayamıyoruz. Bu durumda örneklenmiş işaretin tayfındaki frekans bileşenleri Şekil 1.21'deki gibi örtüşür.



Şekil 1.21.Örneklemede örtüşme.

İşaretin yüksek frekanslı bileşenleri daha düşük frekanslılar üzerine örtüşüyor ve onların bozulmasına sebep oluyor. Örtüşme sonucu elde edilen işaret bileşenleri (sayısal ya da analog) bir süzgeç kullanarak eski haline getirilemez. Bir örnek verelim; f_1 frekanslı sinüs işaretini temsil eden kompleks sayı ile $f_s - f_1$ frekanslı sinüs işaretini temsil eden kompleks sayı toplanıyor. Böylece her iki kompleks sayı ve onların temsil ettiği sinüsoidaller geri elde edilemeyecek şekilde bozuluyor. Bunun olmaması için her iki kompleks sayının sıfır olması, yani o frekanslarda bileşen olmaması, yani $f_s > 2BW_x$ olması gerekir.

Peki, iki bileşenden en azından birini bu duruma düşmekten kurtarabilirmiydik? Cevap ise tahmin edileceği gibi bir tanesini (mantıklı olarak küçük olanı) örneklemeden önce sıfırlamaktır. Böylelikle

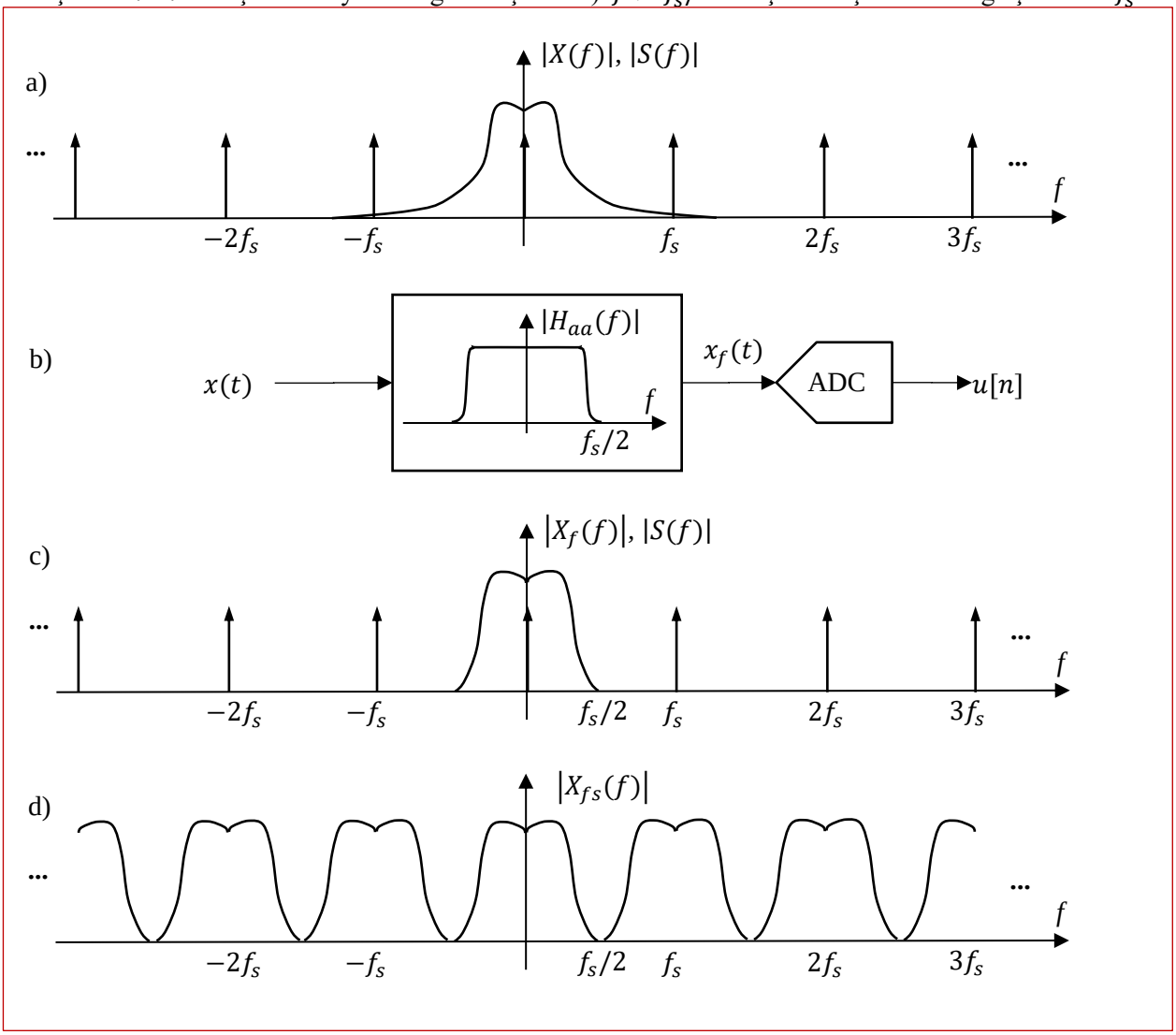
toplamın sonucu sadece büyük olanı temsil eder. Burada, pratikte karşılaşılan çoğu işaretin frekans karakteristiklerinde görülen bir durumdan faydalanıyoruz. Band merkezinden uzak frekanslarda bileşen genlikleri merkezden uzaklaştıkça, çoğunlukla monoton olarak, azalıyor. Bu da merkezden uzak bileşenleri sıfırlamamıza (ya da neredeyse sıfırlamamıza) olanak tanıyor. Böylelikle, zaten genliği düşük olan bileşenleri en baştan çıkararak genliği yüksek olanları örtüşmeden koruyabiliyoruz. Tabi ki bu işlem, işaretin bazı bileşenlerini çöpe attığımız için, asıl işareti bozuyor. Yani, zaten orijinalini geri elde etme imkanımız olmayan işareti daha da bozmuş oluyoruz. Karşılığında da, önemli saydığımız bileşenleri korumuş oluyoruz.

Örtüşmeyi önlemek için yaptığımız bu işlem doğal olarak işaretin önemsiz saydığımız ve frekans merkezinden uzak bileşenlerini iyice bastıran bir analog süzgeç aracılığı ile yapılmaktadır. İsmi de, sürpriz olarak, örtüşme önleyici süzgeç (antialiasing filter) olmaktadır. Şekil 1.22 bu süzgecin sistemdeki yerini ve işlevini açıklamaktadır. Tabi hiçbir gerçek süzgeç istenmeyen/atılacak bileşenleri tam olarak sıfırlayamayacağı için yine de bir miktar örtüşme etkisi görülecektir. Süzgeç kalitesi (mertebesi, keskinliği) yükseldikçe örtüşme etkisi ihmal edilecek seviyelere düşer.

Örtüşme önleyici süzgeç sadece işaretin içerdiği ama daha düşük frekanslı bileşenleri bozacağı için atmak istediğimiz bileşenleri azaltmaz, aynı zamanda yüksek frekanslı gürültü bileşenlerini de atacağı için işaretin işaret/gürültü oranını da (SNR) yükseltmek için pratikte oldukça sık kullanılır. Kullanılmadığı yerler ise zaten ön katlardaki analog devrelerin bir alçak geçiren süzgeç işlevi görecek kadar yavaş olduğu durumlardır. Örneğin mekanik bir algılayıcı (örn: basınç sensörü, mikrofon) zaten kütlesel ataletinden dolayı $f \geq 2f_s$ olan yüksek frekansları algılamayabilir. Bu durumda, ekonomik gerekçelerle örtüşme engelleyici süzgeç kullanılmayabilir.

Örtüşme önleme süzgeci işaret henüz örtüşmeden kullanılınca faydası olacak bir şeydir. Örnekleme işlemi yapıldıktan sonra işareti sayısal bir süzgeçten geçirmenin zaten bozulmuş bileşenlere faydası olmaz. Ancak, örtüşmeyle bozulmuş yüksek frekanslı bileşenleri sayısal süzgeçle atarak bozulmamış bileşenleri temsil eden sayısal örnekler elde edilebilir.

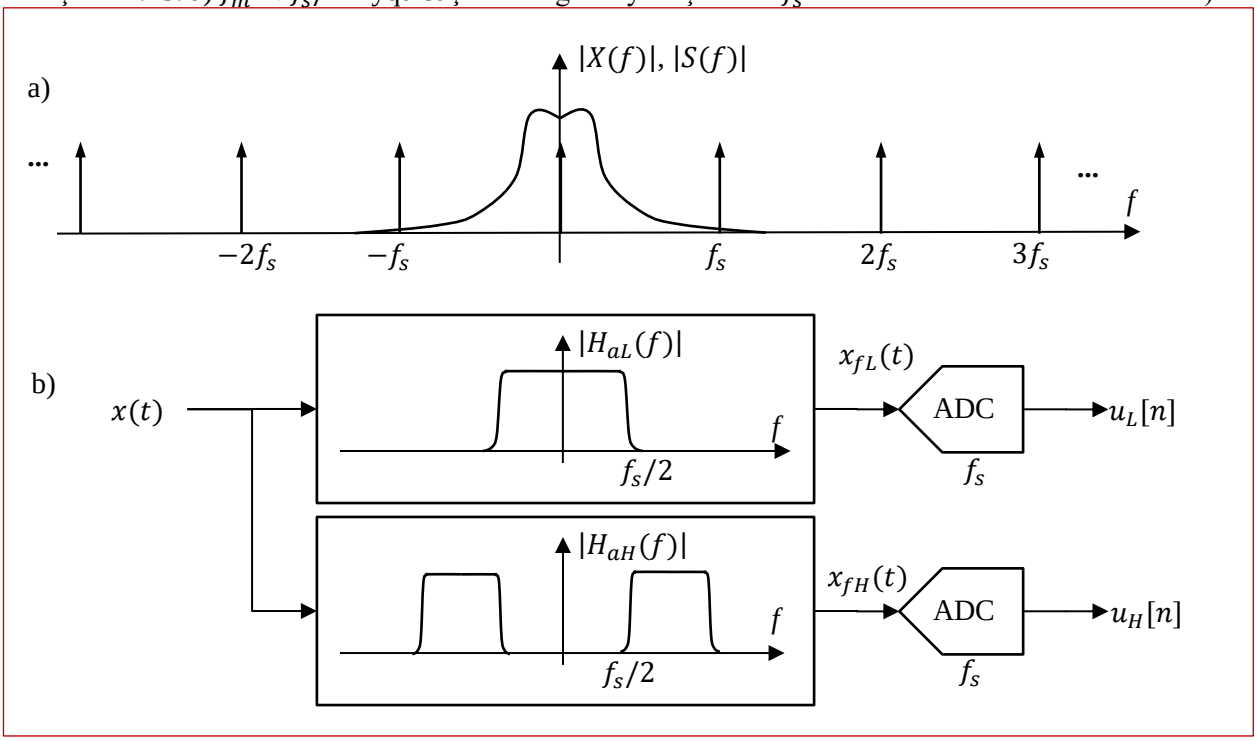
Şekil 1.22. Örtüşme önleyici süzgecin işlevi. a) $f > f_s/2$ bileşenler içeren analog işaret ve f_s



frekanslı dürtü treni. Bu durumda örtüşme oluşur. b) $f_s/2$ mutlak kesim frekanslı alçak geçiren örtüşme önleyici süzgecin ADC'den öncesine yerleştirilmesi. c) Süzgeçten geçmiş analog işaret ve f_s frekanslı dürtü treni. d) Süzgeçten geçmiş işaret ile dürtü treninin evrişimi. Örtüşme oluşmamış ya da oldukça azaltılmış.

Şimdi de Şekil 1.22a'da gösterilen ve $f_m > f_s/2$ bileşenler içeren işareti tekrar ele alalım. f_s dediğimiz şey çoğunlukla elimizde bulunan (temin edilebilen) ADC'nin en yüksek örnekleme oranıdır. $f_s > 2f_m$ şartını sağlamadığı halde $x(t)$ işaretinin $f_s/2$'den yüksek frekanslı bileşenlerini de, bunları örtüşme engelleyici süzgeç ile atmadan, örneklemek ve sayısal alana geçirmek istediğimizi varsayalım. Bu durum için de bir çözümümüz vardır, ancak çözüm hassas analog tasarım içermektedir. Şekil 1.23 böyle bir yaklaşımı göstermektedir. İşaret analog süzgeçlerle 2 yada daha fazla alt-banta ayrılmaktadır. Her alt-bantın bantgenişliği $f_s/2$'den küçük olmalıdır. Böylelikle alt-bant işaretleri ayrı ayrı ADC'leri ile f_s frekansında örneklenip sayısala çevrilebilir. Böylece yüksek frekanslı bileşenlerle ilgili bilgi kaybedilmemiş olur.

Şekil 1.23. a) $f_m < f_s/2$ Nyquist şartını sağlamayan işaret ve f_s frekanslı örnekleme dürtü treni. b)



İşaretin analog süzgeçlerle iki banda ayrılması ve ayrı ayrı ADC'ler ile örneklenmesi.

Pratikte bu böl-yönet yaklaşımı da sıklıkla kullanılmaktadır. Bantgeçiren alt-bantlar analog kipleme yöntemi ile tabanbanta indirilip örneklenebilir ya da Şekil 1.20'yi anlatan açıklamalarda belirtildiği gibi frekans aşağı taşıma (undersampling, frequency down conversion) yöntemiyle de örneklenebilir. Tahmin edileceği gibi burada süzgeç tasarımı oldukça önem kazanmaktadır. Örneklenmiş sayısal işaretler $u_L[n]$ ve $u_H[n]$'nin nasıl işleneceği veya nasıl birleştirileceğine burada değinmeyeceğiz.