Birçok bilgisayar programında kayan noktalı sayılar ile yapılan hesaplama işlemlerinde yuvarlama hatası problemleri ile karşılaşılmaktadır. Kaya noktalı sayılar ile yapılan döngü kontrollerinde noktadan sonraki sayıların hesaplanması ile döngüde tutarsızlıklar meydana gelir. Aksine bu işlemler tamsayı olarak çarpma bölme gibi matematiksel ifadelerin sonucunun yine tamsayı olması sebebiyle döngülerde kesin doğru sonuç elde edilir.

Eğer tamsayı olmayan değerler için kullanılması gerekli ise kesirli kısmın 2 nin katları olan ifadeler kullanmak, 10 un katları olan değerleri kullanmaktan daha doğru sonuç verebilir.

**Yürütme Hızı: Program Dili**

Basit bir şekilde Asembly,derleme ve uygulama arasındaki farkı anlamak için IBM de kullanılan 8088 mikroişlemcisi, gruplar halinde flip-floplar barındırır. Bunlar 1 ve 0 bit serilerini tutarlar. Her biri 16 bit olan 4 kaydedicisi vardır.Ayrıca 9 adet de özel amaçlı kaydedicisi bulunur.

Örneğin 1234 ve 4321 sayılarını toplayan bir program yazılsın. Bunun için birkaç satır kod çalışmalıdır. Hafızaya alınan komutlar yorumlanır ve AX kaydedicisindeki veriye BX kaydedici içeriği eklenir ve AX kaydedicisi geriye döndürülür. Bunu yapan da makine kodudur.

1 ve 0 lar ile yapılacakları belirlemek Assembly ile mümkündür. Asembly, makine kodu gibi çalışır ve anlaşılması da kolaydır. Asembly dilini 2 lik sisteme Asembler programı dönüştürür. Örneğin ADD AX, BX ifadesi aslında 00000011 11000011dır.

MOV CX,1234 CX kaydedicisine 1234 değerini atar.

MOV DS:[0], CX CX içeriğini DS[0] adresine tutar.

MOV CX,4321 CX CXkaydedicisine 4321 değerini atar.

MOV DS:[2], CX CX içeriğini DS[2] adresine tutar.

MOV AX, DS:[0] DS:[0] adres içeriğini AX kaydedicisine alır.

MOV BX, DS:[2] DS:[2] adres içeriğini BX kaydedicisine alır.

ADD AX, BX AX ve BX kaydedicilerinin içeriğini toplayarak AX kaydedicisine yazar.

MOV DS:[4], AX AX kaydedicisinin içeriğini DS[4] adresinde tutar.

Böylece toplama işlemi Asembly dilinde yazılmış olur. Ayriyeten Asembly dışında, soyut değişkenleri referans donanımı olmadan da işleyebilen yüksek seviyeli diller de vardır. Örneğin C, BASIC,FORTRAN gibi. Bu dillerde yazılımcı, donanıma girmeden sadece değişkenleri bilse yeterlidir.

Bir compiler, yüksek seviyeli dil komutunu, makine komutuna çevirir. Örneğin A değişkeni tutalmak istendiğinde compiler bellekte A verisini tutacak kadar yer ayırır. Ayrıca compiler birçok işlemin nasıl yapıldığını bilir (LOG, \*, / vs).

Üst düzey diller, programcıyı donanımdan ayırır. Programcı kişi, yazdığı kodun kodların diğer işlemcilerde nasıl koştuğu ile ilgilenmez. Çoğu derleyici programı çalıştırmadan kodları makine diline çevirir. Interpreter BASIC olarak bilinen compiler tipi, tüm kodu derlemek yerine satır satır derleme yapar ve bu işlem diğerlerine göre çok vakit alır.

Yüksek seviyeli dillerde DSP paketleri uygulamalarında pek çok karmaşıklık bulunmakta. Örneğin yeni bir DSP ye sahip işlemci aldığınızda birçok yerleşik giriş çıkışlar bunun içinde bulunur. Bir C compiler için bu işlemcinin nasıl çalıştığını bilmenize gerek yoktur.

En iyi durumda üretici, bizlere programlarımızda yardımcı olacak paketlere sahip bir yazılım sağlar. Bu paketlerdeki kütüphanelerin içinde algoritmalar, hata ayıklama gibi pek çok özellik bulunur.