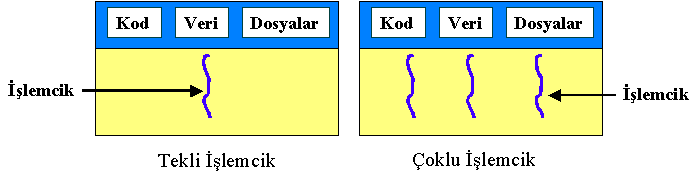
**5.3 İşlemcik (Thread)**

Çoklu program işleme tekniğinde, MİB'de, bir işlemden diğer bir işleme gerçerken, periyodik olarak konteks anahtarlama gerçekleşir. Gerçekleşen bu konteks anahtarlamadan dolayı belirli bir zaman kaybı olur. Konteks anahtarlama sırasında işlemler sürekli olarak  ana bellek üzerine yazılır ya da ana bellek üzerinden okunurlar. Bu nedenle ana bellek üzerinde sürekli olarak okunup yazılma işlemi gerçekleştirilir. Bu iki durumda istenmeyen  durumlardır. İyi bir işletim sisteminde konteks anahtarlama sayısının düşük olması istenir.   
Modern işletim sistemlerinde işlemler geleneksel(yüksek ağırlıklı) işlemler ve hafif ağırlıklı (işlemcik -thread) işlemler olmak üzere ikiye ayrılır. Modern ve karmaşık yazılımlar tek bir işlem yerine birlikte çalışabilen işlemlerden oluşur. Bu işlemler arasında yukarıda anlatılan konteks anahtarlama sorununun yaşanmaması için, bazı işlemler için gerekli olan alt işlemler gruplandırılarak o işlemin bir parçası haline getirilmiştir. Bu tür işlemlere hafif ağırlıklı  işlemler denir. Hafif ağırlıklı işlemler içerisinde birden fazla işlemciğe (çoklu işlemcik-multithread -) sahiptir. Ana işlemlere ise yüksek ağırlıklı işlemler denir. Yüksek ağırlıklı işlemlerin içerisinde bir tek işlemcik (tek işlemcik) bulunur. Çoklu işlemcik ile tek işlemcik arasındaki fark Şekil 5.7 'de görülmektedir. Örneğin, bir kelime işlemci programında, yazma işlemi gerçekleştirilirken, klavyeden bilginin alınması, ekrana basılması ayrı ayrı gerçekleşen iki işlemciktir. Bunun yanında bir diğer işlemcikte yazım hatalarını kontrol ediyor olabilir.   
  



Şekil 5.7 Tek işlemcikli ve çok işlemcikli işlemler.

İşlemcik(thread) için çeşitli tanımlamalar yapılabilir.

* İcra edilmekte olan bir işlemin, aynı kodunu ve adres boşluğunu kullanarak yeniden çalıştırılmasıdır. Modern işletim sistemlerin de bu işleme işlemcik(thread) adı verilir.
* Diğer bir ifadeyle işlemcik, bir işlemin adres boşluğu içinde yer alan yeni bir işlemdir. Bir işlem  birden fazla işlemciğe sahip olabilir.
* Bir işlemin adres boşluğu içinde işlemlerini gerçekleştiren birbirinde bağımsız işlemlere "işlemcik" adı verilir.

Farklı tanımlamalar olmasına rağmen ortak noktalardan ortaya çıkan sonuç : İşlemcik, 

* + Belli bir işleme bağlıdır.
  + Bağlı  bulundukları işlemin adres boşluğunu kullanırlar.
  + Kendi  program sayacına sahiptirler.
  + Gerçekleştirilen prosedürler için kendi adres ve yerel değişkenlere sahiptirler.
  + İşletim  sistemi kaynaklarını paylaşırlar.

İşlemcikler, işlemlere göre iki temel avantaja sahiptir. 

* + Aynı işlem içindeki işlemcikler arasında gerçekleşen konteks anahtarlama zamanı, işlemler arasında gerçekleşen konteks anahtarlama zamanından oldukça küçüktür. Aynı zamanda, işlemler ana bellek üzerine yazılıp-okunurken, işlemcikler kaydediciler (register) üzerine yazılıp-okunurlar.
  + İşlemler arasında haberleşme mesaj iletimi ile sağlanırken, işlemcikler aynı adres boşluğunu kullandıkları için aralarında mesajlaşmaya gerek yoktur. Eğer arzu edilirse işlemcikler kendi aralarında mesajlaşa bildikleri gibi dışarıdan diğer bir işlem ile de mesajlaşabilirler.

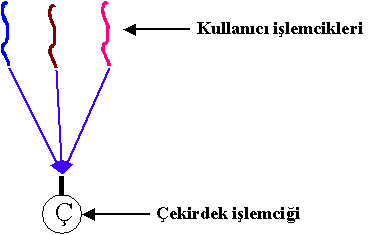
İşlemcikler kullanıcı işlemcikleri ve çekirdek işlemcikleri olmak üzere ikiye ayrılır. Kullanıcı işlemcikleri çekirdek katmanının üstünde olup kullanıcı işlemleri tarafından yaratılırlar. Bu işlemciklerin yaratılması, çizelgelemesi ve yönetilmesi çekirdekten bağımsız olarak gerçekleştirilir. Kullanıcı işlemciklerinin tamamı kullanıcı adres boşluğu içerisinde koşturulur.   
Çekirdek işlemcikleri ise doğrudan doğruya işletim sistemi tarafında yaratılır. Çekirdek işlemciklerinin yaratılması, çizelgelemesi ve yönetimi işletim sistemi tarafında gerçekleştirilir.

**5.3.1 Çoklu İşlemcik Modelleri**

Modern işletim sistemlerinden çoğu hem kullanıcı hem de çekirdek işlemciklerini destekler. Bu nedenle modern işletim sistemlerinin kullandıkları çeşitli çok işlemcikli modeller vardır.

**5.3.1.1 Çoktan-Bire Modeli**

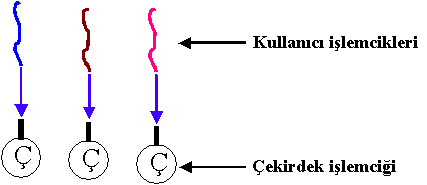
Şekil 5.8'de bu modelin şekli görülmektedir. Bu modelde birden fazla kullanıcı işlemciği ve tek bir  çekirdek işlemciği bulunur. Bu modelde, aynı anda çekirdek katmanında sadece bir işlemcik bulunabildiği için işlemci üzerinde sadece bir işlemcik koşturulabilir.   
  



Şekil 5.8 Çoktan-bire modeli.

**5.3.1.2 Bire Bir Modeli**

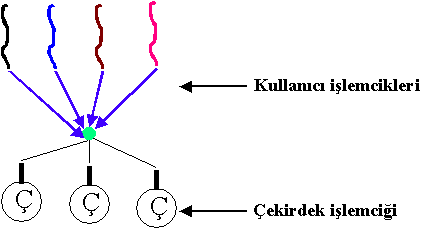
Şekil 5.9'da bu modelin şekli görülmektedir. Bu modelde her kullanıcı işlemciği için çekirdek katmanında bir çekirdek işlemciği koşturulur. Bu nedenle işlemci üzerinde birden fazla işlemcik aynı anda koşturulabilir.   
  



Şekil 5.9 Bir-bir modeli.

**5.3.1.3 Çoktan Çoğa Modeli**

Şekil 5.10'da bu modelin şekli görülmektedir. Bu modelde bir çok kullanıcı işlemciğine karşıklık ya aynı sayıda ya da daha az sayıda çekirdek işlemciği bulunur. Bu modelde de  işlemci üzerinde birden fazla işlemcik aynı anda koşturulabilir.



Şekil 5.10 Çoktan-çoğa modeli.