STARTUP CODE

By Mustafa Onur Parlak

- **Start-up Code Nedir?**
  - Main fonksiyonundan önce çağrılan bir koddur.
  - Uygulama için temeli oluşturur.
  - Assembly'de yazılan küçük koddur.

- **Kullanım Amacı**
  - Donanıma özel başlatma yapabilir.
    - Örneğin, Donanımın **frekans saatini** yükseltmek için **PLL** kullanılıyorsa, başlangıç kodunda **öncelikli olarak PLL** ayarlanabilir. PLL önceliği sonrası geri kalan Startup kodlar azami hızda çalıştırılır.
    - Kartın adres/veri yolunda bulunan harici aygıtlar varsa, **başlatma kodunda yapılandırılır.** Yine aynı şekilde **donanımda bulunan watchdog,** startup code modunda devre dışı bırakılır. Sistem yapılandırılmadan sıfırlanmaz.
    - **Zararları konusunda,** GPIO, Timer ve serial bağlantıların başlatılma önceliğinin kullanıldığı uygulamalarda **özel donanım başlatma** durumunu yapmamakta fayda var.
    - **Donanım ve çalışma ortamında** söz konusu özelleştirmeler varsa Startup code'u özelleştirmeniz gerekebilir.
      Ayrıca taslak olarak kaydetmeniz durumunda tekrar tekrar aynı taslak üzerinden kısmi değişimler ile işleminizi halledebilirsiniz.

- **Kodun Bölümleri:**
  - Stack alanının tanımlanması
  - Heap alanının tanımlanması
  - Vektör tablo
  - Reset handler code
  - Diğer exception handler code

- **Start-up Code adımları:**
  - Tüm Interrupt'lar disable edilecek
  - İlk değerleri verilmiş data'yı ROM'dan RAM'e kopyala
  - İlk değerleri verilmemiş data bölümünü sıfırla
  - Stack için yer ayır ve initialize et
  - İşlemcinin stack pointer'ını initialize et
  - Heap oluştur ve initialize et (heap = stack = yığın)
  - Interruptları enable et
  - Main fonksiyonunu çağır
  - .bss bölümünü sıfırlayın

```
// Program Bölümleri
.vectors    // Hariç tutulan vektör tablosu.
.init       // Uygulamanın main fonksiyonuna çağrı yapılmadan önce çalışan başlangıçkodudur
.ctors      // Statik yapıcı fonksiyon tablosu.
.dtors      // Statik yıkıcı fonksiyon tablosu.
.text       // Program kodu.
.fast       // Hızlı yürütme için flash'tan RAM'e kopyalama kodu.
.data       // Başlatılan statik veriler.
.bss        // Sıfırlanmış statik veriler.
.rodata     // Programın salt okunur sabitleri ve değişmezleri.
.ARM.exidx  // C ++ istisna tablosu.
.tbss       // Yerel depolamada sıfırlanan verileri iş parçacığı ve ardından
.tdata      // Yerel depolamaya başlatılmış verileri işleyin.


// Cortex-M mimarisi için tasarlanan stack ve linker symbol stack'leri (Bağlayıcı sembol yığını)
.stack  __STACKSIZE__                   // Main stack.
.stack_process  __STACKSIZE_PROCESS__   // Process stack.
```