**1 ) MSP430’da Linker (Bağlayıcı) Nasıl Çalışır?**

| **Adım** | **MSP430 tarafında neler olur?** | **Notlar** |
| --- | --- | --- |
| **Derleme** | msp430-elf-as (GNU) veya TI cl430 --asm tek tek *.s* dosyalarını **nesne** (*.o*) dosyalarına dönüştürür. | Nesne dosyaları hâlâ göreli adresler içerir. |
| **Bağlama** | msp430-elf-ld (GNU) veya TI Linker, **linker betiği** (.ld veya .cmd) ile bellek bölgelerini eşler, sembolleri çözer, röleksiyon (relocation) yapar ve tek bir *.elf* üretir. | MCU seçimi (-mmcu=msp430g2553) otomatik betik çağırır; kendiniz yazarsanız MEMORY{ } & SECTIONS{ } bloklarıyla flash-RAM sınırlarını belirlersiniz. Betik kullanılmadığında –T mymap.ld yoluyla elle verilir. [ti.com](https://www.ti.com/lit/ug/slau646f/slau646f.pdf) |
| **Dönüştürme** | İsteğe bağlı msp430-elf-objcopy -O ihex veya TI hex430 ile \*.hex / *.txt* oluşur; bunlar yükleyicinin okuyacağı formattır. | CRC ekleme veya “upper/lower region” seçenekleri de bağlayıcıda verilir. [e2e.ti.com](https://e2e.ti.com/support/microcontrollers/msp430/f/msp-low-power-microcontroller-forum/848786/compiler-msp430fr5969-msp430-gcc-linker-section-checksum?utm_source=chatgpt.com) |

**Fark / SIC-SIC ⁄ XE**  
SIC loader’ı çoğu adres çözümünü **çalıştırma zamanında** (Pass 1/Pass 2 linking loader) yapar; MSP430’da ise adres çözümü bağlayıcıda biter, loader sadece imajı flash’a kopyalar.

**2 ) MSP430’da Loader (Yükleyici) Nasıl Çalışır?**

* **Donanımdaki ROM BSL** (BootStrap Loader) her MSP430’da küçük bir ROM alanında durur.
  + UART-, I²C-, bazılarında USB tabanlı komut seti vardır.
  + “BSL giriş dizisi” gönderildiğinde MCU, uygulama flash’ını silme/yazma, CRC okuma, cihaz çıkışı gibi komutları kabul eder. [ti.com](https://www.ti.com/lit/pdf/slau319)
  + PC tarafında **MSP430-Flasher**, **UniFlash**, **mspdebug** (“load -e main.hex”) gibi araçlar yaygındır.
* **JTAG / SBW** (Spy-Bi-Wire) yoluyla yükleme: IDE (CCS, IAR, Energia) debug probe üzerinden *.elf* dosyasını doğrudan RAM’e veya flash’a yazar.
* **Uygulama içi (IAP) yükleyici**: Kendi flash sayfalarınızı korumak için kullanıcı yazılımı içinde ikinci bir “mini-boot” kodu da eklenebilir (ör. haricî seri flash’tan güncelleme) – TI E2E’de örnekleri var. [e2e.ti.com](https://e2e.ti.com/support/microcontrollers/msp-low-power-microcontrollers-group/msp430/f/msp-low-power-microcontroller-forum/494522/bsl-bootloader-using-external-serial-flash?utm_source=chatgpt.com)

**Fark / SIC-SIC ⁄ XE**  
SIC’de yükleyici “Header-Text-Modification-End” kayıtlarını okuyup belleğe yazar, gerektiğinde röleksiyon bitlerini işler. MSP430 BSL ise tüm röleksiyon bitmiş, **mutlak** adresli imaj bekler; görevi saf kopyalama ve bütünlük doğrulamadır. [enggnotebook.weebly.com](https://enggnotebook.weebly.com/uploads/2/2/7/1/22718186/chapter_3loaders_and_linkersnew.pdf)

**3 ) Macro Direktifleri**

| **Özellik** | **SIC ⁄ SIC-XE** | **MSP430 Assembler** |
| --- | --- | --- |
| **Başlama / bitiş** | MACRO … MEND | .macro … .endm |
| **Parametre işareti** | &COUNT, &BUF | \count (GNU) veya çıplak ada karşılık gelen parametre (TI) |
| **Yerleşik fonksiyonlar** | Sınırlı (string değişimi yok) | .asg, .eval, .loop, koşullu derleme vb. geniş makro-dili [ti.com](https://www.ti.com/sc/docs/products/micro/msp430/userguid/as_6.pdf) |
| **Kütüphane** | Ders kitabı genelde tek dosya kullanır | .mlib ile arşivlenmiş makro kütüphanesi çağrılabilir |

**SIC’de Makro Örneği**

asm

KopyalaDüzenle

RDBUFF MACRO &INDEV,&BUFADR

CLEAR X

LOOP TD &INDEV

JEQ LOOP

RD &INDEV

STCH &BUFADR,X

TIX &BUFADR

JLT LOOP

MEND

MACRO / MEND arası gövde, her çağrıda parametre yerine gerçek argümanları koyarak genişletilir. [geeksforgeeks.org](https://www.geeksforgeeks.org/macro-processor/?utm_source=chatgpt.com)

**MSP430’da Eşdeğeri**

asm

KopyalaDüzenle

; add3.m43

.macro add3 a,b,c

mov \a, R4

add \b, R4

add \c, R4

.endm

add3 val1, val2, val3 ; derleyici burada üç MOV+ADD dizisi üretir

Makrolar .include ile eklenebilir; asm listesine genişlemiş hâli isteğe bağlı yazdırılır. [ti.com](https://www.ti.com/sc/docs/products/micro/msp430/userguid/as_6.pdf)

**MSP430’da “SICDE Macro Direktifi Var mı?”**  
Evet—adı farklı (.macro/.endm) ve çok daha yetenekli; SIC kitaplarındaki kısıtlı MACRO dili yerine tam TMS-msp430 makro-dili, koşullu derleme ve döngü direktifleri sunar.

**4 ) MSP430 ⇄ SIC / SIC-XE Karşılaştırmasının Özü**

* **Linkleme aşaması**
  + **SIC-XE**: Parçalar **loader** tarafından bağlanır; “Modification record” veya “bit-mask” ile röleksiyon.
  + **MSP430**: Röleksiyon bağlayıcıda tamam; loader imajı sadece taşır.
* **Loader tasarımı**
  + **SIC**: Yazılım loader’ı çalıştırma zamanı RAM’e kopyalama + adres düzeltme yapar.
  + **MSP430**: ROM BSL + debug arayüzü; taşınabilir güncelleme kolaylığı, fakat adres sabit.
* **Makro işleyici**
  + **SIC**: Derleyiciden önce tek-geçişli makro işlemcisi; yalnızca metin değiş-koy.
  + **MSP430**: Assembler içine gömülü güçlü makro dili; koşullu, yinelemeli, kütüphane destekli.
* **Mimari etki**
  + **SIC** 24-bit tek bellek, örnekleyici amaçlı.
  + **MSP430** Harvard-vari 16-/20-bit mimari, gerçek donanım; flash-RAM haritaları linker betiğinde.

**5 ) Hızlı Başlangıç Önerileri**

1. **Linker betiğini inceleyin:** Projenizde otomatik gelen msp430xxxx.ld dosyasını açın; hangi MEMORY bloğunun .text, .data, .bss gibi SECTIONS’a eşlendiğini görün.
2. **Özel bölge ekleyecekseniz:**

ld

KopyalaDüzenle

.crc16 : { KEEP(\*(.crc16)) } > INFOB /\* info flash’ta sakla \*/

1. **BSL ile yükleme testi:** msp430-flasher -w app.txt -v -g komutu (UART veya USB seçerek) temel akışı gösterir.
2. **Makroları modüler tutun:** TI’nin .mlib arşiv biçimini kullanırsanız makroları proje içinde derleyici otomatik bulur; SIC’teki gibi tüm makroları başa kopyalama ihtiyacı kalkar.