MSP430 Assembler & Linker GUI

Bu proje, MSP430 mimarisi için bir Assembly derleyicisi ve bağlayıcısı (assembler & linker) ile entegre edilmiş bir grafik kullanıcı arayüzü (GUI) sunar. Python ve Tkinter kütüphanesi kullanılarak geliştirilmiştir ve aşağıdaki ana bileşenlerden oluşur:

- **MSP430Assembler**: Assembly kodunu analiz eder, semboller, sectionlar, export/import ve relocation işlemlerini yönetir; makine kodu üretir.
- **LinkEditor**: Birden fazla .obj dosyasını birleştirir, relocation (bağlama) işlemlerini yapar ve çalıştırılabilir dosya oluşturur.
- **MSP430AssemblerUI**: Kullanıcıya kod yazma, makine koduna çevirme, obj dosyası üretme ve modülleri bağlama işlevlerini sunan ana Tkinter arayüzüdür.
- **LineNumberedText**: Satır numaralı ve sözdizimi renklendirmeli metin editörü.

İçindekiler

Kurulum

Sistem Gereksinimleri

- **Python 3.7+**
- **Tkinter** (çoğu Python dağıtımında varsayılan olarak gelir)
- **OS**: Windows, Linux, MacOS (Tkinter destekliyor olmalı)

Gerekli Kütüphaneler

Sadece standart kütüphaneler (`tkinter`, `os`, `re`, vb.) kullanılmıştır. Ekstra bir modül yüklemeniz gerekmez.

Dosya Listesi

- `main.py` : Tüm kodun yer aldığı ana Python dosyası.
- `README.md` : Bu dosya, detaylı kullanım ve kurulum yönergeleri içerir.
- (Oluşturulacak) `temp/` : Obj ve final obj dosyaları burada oluşur.

Çalıştırma

1. **Python'un kurulu olduğundan emin olun.**

   ```
   python --version
   ```

2. **Projeyi bir klasöre çıkarın ve terminal/cmd ile o klasöre gidin.**
3. **Programı başlatmak için:**

   ```
   python main.py
   ```

   veya Windows'ta:
   - `main.py` dosyasına çift tıklayın.

Kullanım

Arayüz açıldığında örnek bir MSP430 Assembly kodu otomatik olarak yüklenecektir. İsterseniz yeni kod yazabilir, bir dosyadan kod yükleyebilir veya mevcut kodu kaydedebilirsiniz.

Arayüz Bileşenleri

- **Sol Panel:** Assembly kodunuzu yazabileceğiniz, satır numaralı ve sözdizimi renklendirmeli alan.

- **Sağ Panel:** Makine koduna dönüştürülmüş çıktının satır satır gösterildiği alan.
- **Semboller/Sectionlar Paneli (Alt):** Kodunuzdaki semboller, export/import edilen semboller, section bilgileri ve detayları.
- **Butonlar:**
  - **Dosya Aç:** `.asm` dosyası yükler.
  - **Kaydet:** Assembly + makine kodunu bir dosyaya kaydeder.
  - **Temizle:** Tüm arayüzü sıfırlar.
  - **Kodu Çevir:** Kodunuzu analiz eder ve makine koduna çevirir.
  - **Modülleri Link Et:** Birden çok obj dosyasını birleştirip final çalıştırılabilir dosya üretir.

Kod Dönüştürme

- Kodunuzu yazın veya yükleyin.
- `Kodu Çevir` butonuna tıkladığınızda:
  - Semboller ve sectionlar analiz edilir (PASS1).
  - Makine kodu üretilir (PASS2).
  - Semboller, sectionlar ve referanslar ilgili tablolarda gösterilir.
  - `temp/` klasöründe otomatik olarak bir obj dosyası oluşturulur.

Obj Dosyası Üretimi

- Kodunuzu başarıyla çevirdiğinizde, program otomatik olarak `temp/` klasöründe bir `.obj` dosyası oluşturur.
- Bu obj dosyası, COFF (Common Object File Format) benzeri bir düzende, bölümler, semboller ve relocation verileri içerir.
- Farklı asm dosyalarını ayrı ayrı makine koduna çevirip her biri için ayrı obj dosyaları oluşturabilirsiniz.

Modül Linkleme (Bağlayıcı)

- Birden fazla `.obj` dosyası varsa, **Modülleri Link Et** butonuna basarak bunları birleştirebilirsiniz.
- Bağlayıcı (link editor), export edilen sembolleri ve relocation girişlerini kullanarak, farklı modüllerdeki kod ve veri segmentlerini bir araya getirir, adresleri günceller.
- Sonuçta `temp/final.obj` adında, çalıştırılabilir formatta bir dosya oluşur.

Semboller ve Sectionlar

- Kodunuzu çevirdiğinizde, alt panelde otomatik olarak şu tablolar güncellenir:
  - **Semboller Tablosu:** Kodunuzda tanımlanan tüm label'lar ve adresleri.
  - **Exports (.def):** Dışa aktarılan (başka modüller tarafından kullanılabilecek) semboller.
  - **Imports (.ref):** Dışarıdan kullanılacak (başka bir obj'de tanımlı) semboller.
  - **Section Tablosu:** Her section (ör. `.text`, `.data`, `.bss`) için başlangıç adresi ve toplam boyut.
  - **Detaylar:** Bir section seçildiğinde, o section'daki semboller ve referanslar detaylı olarak gösterilir.

Assembly Dili Özellikleri

Sözdizimi

- Semboller (etiketler): `label:`
- Komutlar: `INSTRUCTION OPERAND1, OPERAND2`
- Direktifler: `.data`, `.bss`, `.text`, `.word`, `.byte`, `.space`, `.def`, `.ref`, `.org`, `.end`
- Yorumlar: `;` işaretiyle başlar ve satır sonuna kadar devam eder.
- Export/Import: `.def` ve `.ref` ile sembol tanımlama
- Desteklenen adresleme türleri: Register, immediate (`#değer`), label referansları.

Desteklenen Direktifler ve Komutlar

Direktifler

- `.text`, `.data`, `.bss` : Section başlangıcı
- `.word`, `.byte` : Veri tanımlama (16-bit veya 8-bit)
- `.space` : BSS segmentinde alan ayırma
- `.org` : Adres başlangıcı
- `.def` : Export edilen sembol(ler)
- `.ref` : Import edilen sembol(ler)
- `.end` : Kod sonu

Komutlar (Bazıları)

- **Transfer/Veri**: `MOV`, `MOV.W`
- **Aritmetik**: `ADD`, `ADD.W`, `SUB`, `SUB.W`
- **Karşılaştırma**: `CMP`
- **Jump/Branch**: `JMP`, `JEQ`, `JNE`, `JC`, `JN`, `JNC`, `JGE`, `JL`
- **Çağrı/Çıkış**: `CALL`, `RET`
- **NOP**: `NOP`

Not: Komutlar ve operandlar büyük/küçük harf duyarlı değildir. Semboller (label) harf veya alt çizgi ile başlar.

Dosya Formatları

Assembly Dosyası (`.asm`)

- MSP430 assembly kodu içerir.
- Örnek:

```
 .def start
.text
start: MOV.W #0x1234, R4
       NOP
```

Obj Dosyası (`.obj`)

- Otomatik olarak oluşturulur, COFF benzeri sade bir yapıya sahiptir.
- Bölümler:
  - `SECTION .text`
  - `SECTION .data`
  - `EXPORTS`
  - `RELOCATIONS`
- Her bölüm, ilgili verileri içerir; relocations, import edilen sembolleri ve konumlarını belirtir.
- `EOF` ile biter.

Final Obj Dosyası (`final.obj`)

- Çoklu modül birleştirme sonrası elde edilir.
- Bağlama ve relocation işlemleri tamamlanmış, doğrudan çalıştırılabilir (simülasyon için) formattadır.

Tipik Kullanım ve Örnekler

1. **Tek modül derleme:**
   - Kodunuzu arayüze yazın veya dosyadan yükleyin.
   - `Kodu Çevir` ile makine kodunu ve obj dosyasını oluşturun.
   - Simülasyon veya yükleme için `temp/` klasöründeki obj dosyasını kullanın.
2. **Çoklu modül derleme ve linkleme:**
   - Farklı assembly dosyaları için ayrı ayrı `Kodu Çevir` yapın, her biri için obj dosyası oluşur.
   - `Modülleri Link Et` ile tüm obj dosyalarını birleştirip final obj dosyası oluşturun.
3. **Section ve sembol inceleme:**
   - Kodunuzu çevirdikten sonra semboller ve sectionlar tablolarında oluşan adreslemeleri inceleyin.
   - Referanslar ve export/import ilişkilerini kolayca görebilirsiniz.

Geliştirici Notları

Klasör ve Dosya Yapısı

- **Ana dosya:** `main.py` (tüm mantık bir dosyada toplanmıştır)
- **Geçici dosyalar:** Kodunuzu çevirdiğinizde ve link işlemi yaptığınızda oluşan tüm `.obj` ve `final.obj` dosyaları `temp/` klasörüne yazılır.
- **Ekstra dosya/dizin gereksinimi yoktur.** Programı başka bir dizinde veya farklı bir bilgisayarda çalıştırmak için yalnızca `main.py` ve bu `README.md` yeterlidir.

Kodda Dikkat Edilmesi Gerekenler

- **Yorumlar:** Kod, hem ana sınıflarda hem de fonksiyonlar içinde anlamayacak kişilerin dahi rahatlıkla takip edebilmesi için bolca yorum içerir.

- **GUI:** Tkinter kullanıldığı için arayüzde platforma göre (ör. Windows/Linux) ufak farklılıklar olabilir.
- **Dil:** Kodda değişken isimleri, arayüz metinleri ve hata mesajları Türkçedir.

Kendi Assembly Komutlarınızı Eklemek

- `MSP430Assembler` sınıfı içinde `self.instructions` ve gerekiyorsa operand işleme fonksiyonlarında yeni komutlar kolayca eklenebilir.
- Register listesine yeni kayıt ekleyebilirsiniz.

Genişletme Önerileri

- Daha fazla MSP430 komut desteği eklenebilir.
- .obj ve .coff dosya formatları daha gelişmiş hale getirilebilir.
- Dışa aktarım için Intel HEX veya başka binary format desteği eklenebilir.
- Komut satırı arayüzü (CLI) eklenebilir.
- Kod analizi ve hata gösterimi geliştirilebilir.

Sorun Giderme

- **Tkinter Hatası:** Eğer program açılmıyorsa ve `ModuleNotFoundError: No module named 'tkinter'` gibi bir hata alıyorsanız, Python kurulumunuzda Tkinter eksik olabilir. Ubuntu/Linux'ta şu şekilde yükleyebilirsiniz:

  ```
  sudo apt-get install python3-tk
  ```

- **Dosya İzinleri:** `temp/` klasörüne yazma izniniz olmalıdır.
- **Klavye/Türkçe karakter sorunları:** Kodunuzda Türkçe karakter kullanıyorsanız, kaydettiğiniz dosyanın encoding'i UTF-8 olmalıdır.
- **Beklenmeyen Hata:** Hata penceresinde görülen hatayı geliştiriciye bildirin veya ilgili kod satırını kontrol edin.

Lisans

Bu yazılım eğitim/akademik amaçlı geliştirilmiştir. Dilediğiniz gibi kullanabilir, değiştirebilir ve paylaşabilirsiniz. Tüm telif hakları ve sorumluluklar kullanıcıya aittir.

Sıkça Sorulan Sorular (SSS)

S: Programı başka bir bilgisayara nasıl taşıyabilirim?

C: Sadece `main.py` ve `README.md` dosyalarını kopyalamanız yeterlidir. Python 3 ve Tkinter yüklü olmalı.

S: Kodumu yazdım, ama hatalı satırda neden kırmızı işaret çıkıyor?

C: Sözdiziminde veya desteklenmeyen bir komutta hata yapmış olabilirsiniz. Kodunuzu ve örneklere uygunluğunu kontrol edin.

S: Birden fazla .obj dosyasını nasıl birleştiririm?

C: Her bir kod parçası için ayrı ayrı `Kodu Çevir` işlemi yapın, ardından `Modülleri Link Et` ile birleştirin.

S: Hangi dosyalar kaydediliyor?

C: Yalnızca siz `Kaydet` butonuna bastığınızda, hem assembly kodunuz hem de makine kodu kaydedilir. Otomatik olarak ise obj dosyaları `temp/` altına yazılır.

Katkıda Bulunma

Pull request ve issue açarak projeye katkı sağlayabilirsiniz. Yeni özellik talepleriniz ve hata bildirimleriniz memnuniyetle karşılanır.
