

# I2C Master Top Modülünün OpenLane ile GDSII Üretim Süreci Raporu

**Hazırlayanlar:** Emre YAĞCI (360523012), Sabri SEVİNÇLİ (360523033)

## Amaç

Bu çalışmada amaç, “I2C Master Top” modülünü OpenLane kullanarak ASIC akışına sokmak, sentetik, floorplan, placement ve routing adımlarını gerçekleştirip final GDS çıktısı almaktır.

## Kullanılan Ortam

- WSL
- Ubuntu (Ubuntu 22.04 kullanıldı)
- Docker
- Python 3
- OpenLane
- PDK olarak Sky130A
- Klayout (layout görüntüleme için)
- Volare (PDK yönetimi için)

## İçindekiler

WSL Kurulumu.....	2
Openlane Ortamının Hazırlanması.....	2
Docker Desktop Ubuntu için Enable Etmek .....	4
OpenLane Kurulumunun Tamamlanması Volare ile sky 130 pdk Kurulumu.....	5
i2c_master_top RTL Dosyasını İndirmek. ....	6
clean_verilog.py.....	7
Neden Bu temizliği Yaptık.....	11
VS Code WSL Entegrasyonu.....	13
RTL to GDSII ASIC Tasarımının Başlaması.....	16
OpenLane Akışı .....	19
Sonuç ve Değerlendirme .....	20
Kaynakça.....	21
Teşekkürler .....	21

## WSL Kurulumu

İlk olarak WSL'i gösterildiği üzere kurup Ubuntu 24.04 sistemini kuruyoruz. WSL bize linux komutlarını Windows üzerinden erişmemizi sağlayacak.

```
smfse@Sabri: /mnt/c/Users/smfse$ wsl -l
Linux için Windows Alt Sistemi Dağıtımları:
docker-desktop (Varsayılan)
PS C:\Users\smfse> wsl --list --online
Aşağıdakiler, yüklenemedik geçerli dağıtımların listesidir.
'wsl.exe --install <Distro>' kullanarak yükleyin.

NAME                                FRIENDLY NAME
AlmaLinux-8                         AlmaLinux OS 8
AlmaLinux-9                         AlmaLinux OS 9
AlmaLinux-Kitten-10                 AlmaLinux OS Kitten 10
AlmaLinux-10                        AlmaLinux OS 10
Debian                             Debian GNU/Linux
FedoraLinux-42                      Fedora Linux 42
SUSE-Linux-Enterprise-15-SP5        SUSE Linux Enterprise 15 SP5
SUSE-Linux-Enterprise-15-SP6        SUSE Linux Enterprise 15 SP6
Ubuntu                             Ubuntu
Ubuntu-24.04                        Ubuntu 24.04 LTS
archlinux                           Arch Linux
kali-linux                          Kali Linux Rolling
openSUSE-Tumbleweed                 openSUSE Tumbleweed
openSUSE-Leap-15.6                  openSUSE Leap 15.6
Ubuntu-18.04                        Ubuntu 18.04 LTS
Ubuntu-20.04                        Ubuntu 20.04 LTS
Ubuntu-22.04                        Ubuntu 22.04 LTS
OracleLinux_7.9                     Oracle Linux 7.9
OracleLinux_8.7                     Oracle Linux 8.7
OracleLinux_9.1                     Oracle Linux 9.1
PS C:\Users\smfse> wsl --install --distribution Ubuntu-24.04
İndiriliyor: Ubuntu 24.04 LTS
Yükleniyor: Ubuntu 24.04 LTS
Dağıtım başarıyla yüklendi. 'wsl.exe -d Ubuntu-24.04' ile başlatılabilir
Başlatılıyor Ubuntu-24.04...
Provisioning the new WSL instance Ubuntu-24.04
This might take a while...
Create a default Unix user account: smfse
logger: socket /dev/log: No such file or directory
logging to syslog failed: command line logger --id=157 --tag=adduser --priority=user.info -- Adding user 'smfse' ... returned error: 256
logger: socket /dev/log: No such file or directory
logging to syslog failed: command line logger --id=157 --tag=adduser --priority=user.info -- Selecting UID/GID from range 1000 to 59999 ... returned error: 256
logger: socket /dev/log: No such file or directory
logging to syslog failed: command line logger --id=157 --tag=adduser --priority=user.info -- Adding new group 'smfse' (1000) ... returned error: 256
logger: socket /dev/log: No such file or directory
logging to syslog failed: command line logger --id=157 --tag=adduser --priority=user.info -- Adding new user 'smfse' (1000) with group 'smfse (1000)' ... returned error: 256
logger: socket /dev/log: No such file or directory
logging to syslog failed: command line logger --id=157 --tag=adduser --priority=user.info -- Creating home directory '/home/smfse' ... returned error: 256
logger: socket /dev/log: No such file or directory
logging to syslog failed: command line logger --id=157 --tag=adduser --priority=user.info -- Copying files from '/etc/skel' ... returned error: 256
New password:
Retype new password:
passwd: password updated successfully
logger: socket /dev/log: No such file or directory
logging to syslog failed: command line logger --id=157 --tag=adduser --priority=user.info -- Adding new user 'smfse' to supplemental / extra groups 'users' ... returned error: 256
logger: socket /dev/log: No such file or directory
logging to syslog failed: command line logger --id=157 --tag=adduser --priority=user.info -- Adding user 'smfse' to group 'users' ... returned error: 256
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

## Openlane Ortamının Hazırlanması

Ardından Openlane kurmak için gerekli ortamı hazırlamalıyız.

mkdir -p openlanemarmara adında openlane çalıştıracağımız bir dosya oluşturuyoruz.

cd openlanemarmara dosyamızın içine giriyoruz.

```
smfse@Sabri: /mnt/c/Users/smfse$ mkdir openlanemarmara
smfse@Sabri: /mnt/c/Users/smfse$ cd openlanemarmara
```

Ardından openlane kurmak için bu repositoryu klonladık.

```
(venv) smfse@fabri:~/openlanemarmara$ git clone https://github.com/the-OpenROAD-Project/OpenLane.git
Cloning into 'OpenLane'...
remote: Enumerating objects: 18739, done.
remote: Counting objects: 180% (298/298), done.
remote: Compressing objects: 100% (284/284), done.
remote: Total 18739 (delta 199), reused 186 (delta 94), pack-reused 18432 (from 3)
Receiving objects: 100% (18739/18739), 855.78 MiB | 19.65 MiB/s, done.
Resolving deltas: 100% (13488/13488), done.
(venv) smfse@fabri:~/openlanemarmara$
```

Openlanemarmara içine klonladığımız openlane klasörüne giriyoruz.

Make diyerek openlane i çalıştırıyoruz ama önümüze bazı hatalar çıkıyor.

Make, bizim için gerekli her şeyi kuruyor python3 volare dahil, ayrı bir işlemle Python3 ve volare kurulmasına gerek yok.

Volare ve Python, OpenLane için gerekli olan PDK dosyasını oluşturmayı sağlıyor.

```
(venv) smfse@fabri:~/openlanemarmara$ cd OpenLane
(venv) smfse@fabri:~/openlanemarmara/OpenLane$ make
Traceback (most recent call last):
  File "/home/smfse/openlanemarmara/OpenLane/.env.py", line 242, in <module>
    main()
  File "/home/smfse/openlanemarmara/OpenLane/.env.py", line 238, in main
    commands[args[0]]()
  File "/home/smfse/openlanemarmara/OpenLane/.env.py", line 53, in docker_config
    raise Exception("No container engine found.")
Exception: No container engine found.
make[1]: Entering directory '/home/smfse/openlanemarmara/OpenLane'
Traceback (most recent call last):
  File "/home/smfse/openlanemarmara/OpenLane/.env.py", line 242, in <module>
    main()
  File "/home/smfse/openlanemarmara/OpenLane/.env.py", line 238, in main
    commands[args[0]]()
  File "/home/smfse/openlanemarmara/OpenLane/.env.py", line 53, in docker_config
    raise Exception("No container engine found.")
Exception: No container engine found.

The command 'docker' could not be found in this WSL 2 distro.
We recommend to activate the WSL integration in Docker Desktop settings.

For details about using Docker Desktop with WSL 2, visit:
https://docs.docker.com/go/wsl2/

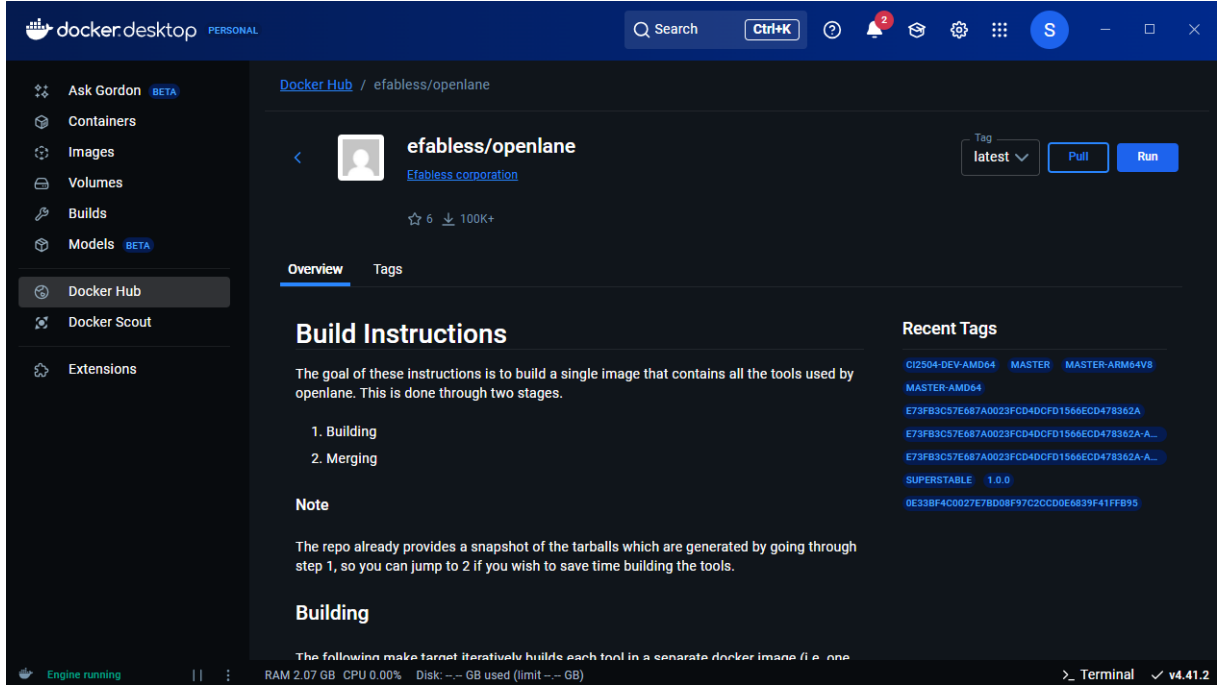
make[1]: *** [Makefile:125: pull-openlane] Error 1
make[1]: Leaving directory '/home/smfse/openlanemarmara/OpenLane'
make[1]: Entering directory '/home/smfse/openlanemarmara/OpenLane'
Traceback (most recent call last):
  File "/home/smfse/openlanemarmara/OpenLane/.env.py", line 242, in <module>
    main()
  File "/home/smfse/openlanemarmara/OpenLane/.env.py", line 238, in main
    commands[args[0]]()
  File "/home/smfse/openlanemarmara/OpenLane/.env.py", line 53, in docker_config
    raise Exception("No container engine found.")
Exception: No container engine found.
make[2]: Entering directory '/home/smfse/openlanemarmara/OpenLane/docker'
NIX_SYSTEM=x86_64-linux BUILD_ARCH=amd64 bash build.sh
+ set -e
** nix build --no-link --print-out-paths --accept-flake-config --option system x86_64-linux --extra-platforms x86_64-linux ../packages.x86_64-linux.openlane1-docker
build.sh: line 19: nix: command not found
+ TARBALL=
make[2]: *** [Makefile:13: openlane] Error 127
make[2]: Leaving directory '/home/smfse/openlanemarmara/OpenLane/docker'
make[1]: *** [Makefile:117: openlane] Error 2
make[1]: Leaving directory '/home/smfse/openlanemarmara/OpenLane'
make: *** [Makefile:128: get-openlane] Error 2
(venv) smfse@fabri:~/openlanemarmara/OpenLane$
```

Bu hatalar genellikle docker, volare ve Python3 kurulumundan kaynaklanan hatalar olmakta. Şu anki hatamız docker dan kaynaklanmaktadır.

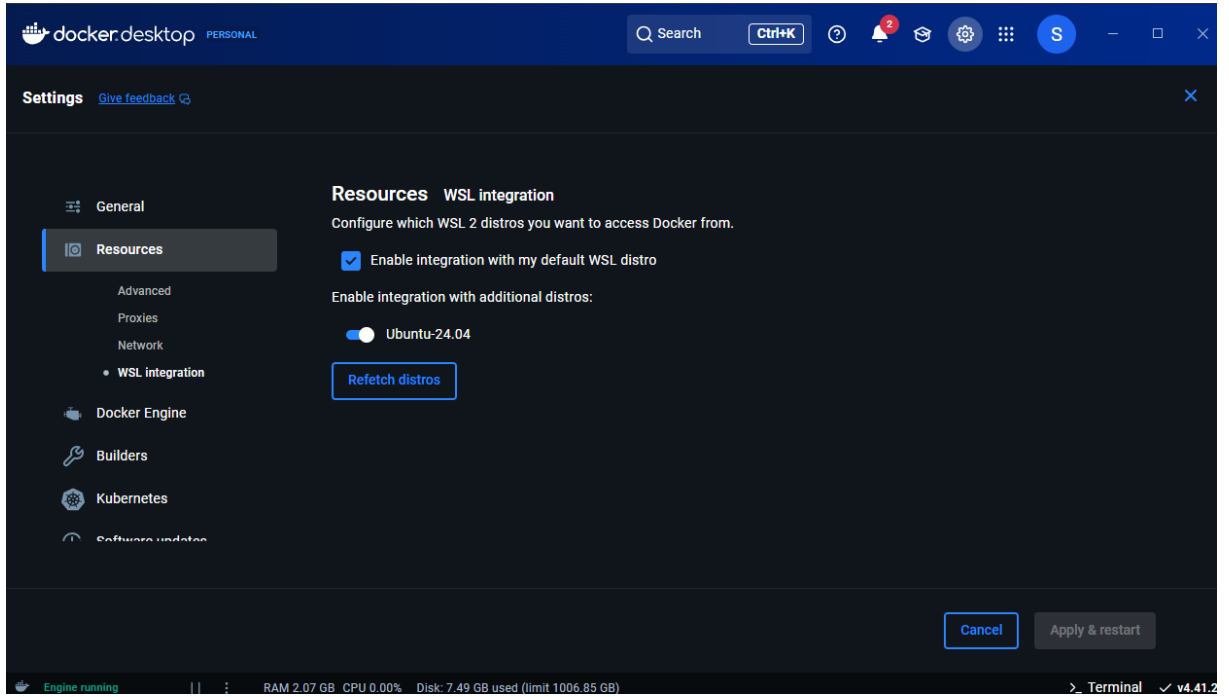
WSL üzerinden OpenLane kurduğumuz için docker ı docker desktop indirerek aktif etmemiz gerekmekte. Docker'ın Windows için olan sürümünü indirdikten sonra docker da bazı ayarlamalar yapmamız gerekti.

## Docker Desktop Ubuntu için Enable Etmek

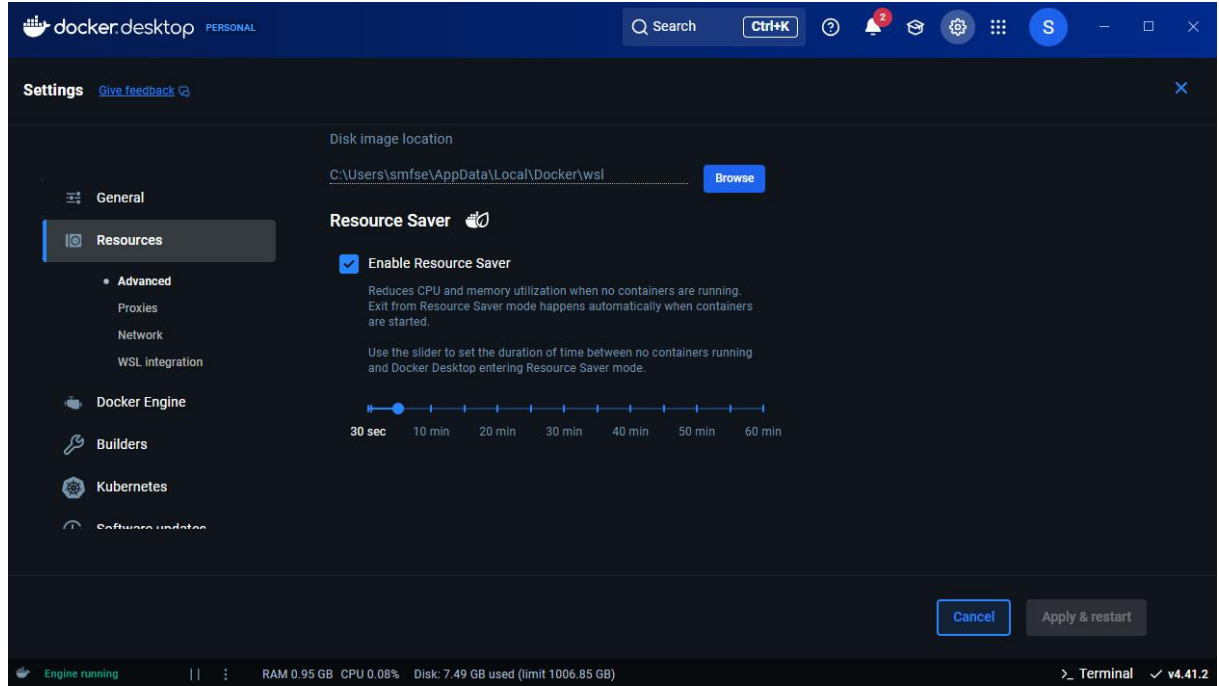
Burada docker hub içerisinde efabless openlane hub ı pull etmeliyiz.



Ardından ayarlar bölümünden resource kısmından WSL entegrasyonu sağlayıp kullandığımız Ubuntu-24.04 dosyasını Windows için entegre etmeliyiz.



Aynı zamanda resource advanced kısmında Resource saver aktif edilmeli.



Openlane kurulumundan GDSII ya kadar her aşamada docker dektop açık olmalıdır.



## OpenLane Kurulumunun Tamamlanması Volare ile sky 130 pdk Kurulumu.

Şimdi make dediğimizde errorlar ortadan kalkmış olacak.

Burada volare ile pdk “sky 130 pdk” olarak belirlendi.

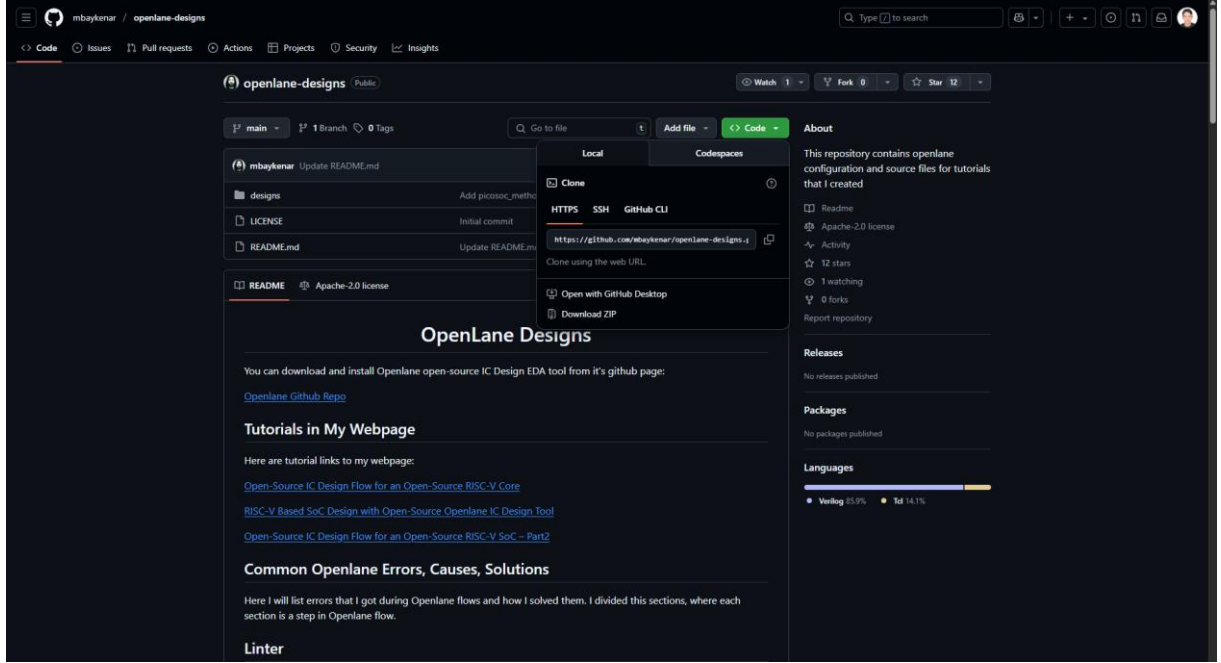
Python3.12 kuruldu

Make bizim için gerekli Python ve volare kurdu.

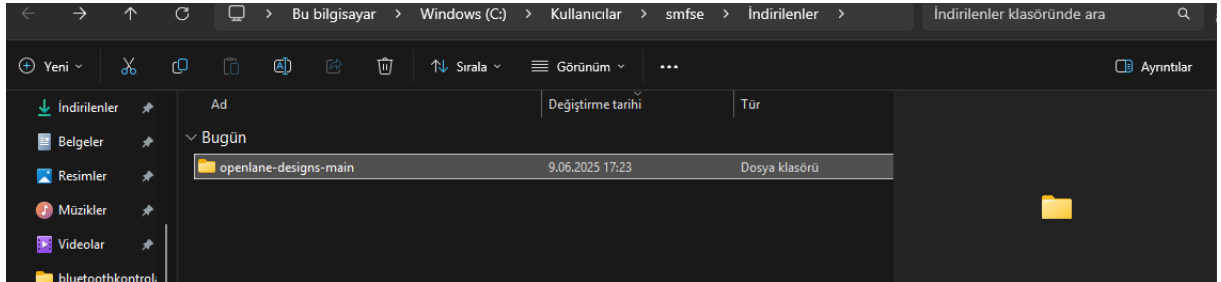
```
msf@Sakari:~/openlanemarmara/OpenLane$ make
make[1]: Entering directory '/home/smfse/openlanemarmara/OpenLane'
e73fb3c57e687a0923fcd9dcfd1566ecd478362a: Pulling from efabless/openlane
Digest: sha256:26719ced98c315b8b4ad7b9dc3e9a176991cea4c3f3282668d8d6d0f0cae229
Status: Image is up to date for efabless/openlane:e73fb3c57e687a0923fcd9dcfd1566ecd478362a
docker.io/efabless/openlane:e73fb3c57e687a0923fcd9dcfd1566ecd478362a
make[1]: Leaving directory '/home/smfse/openlanemarmara/OpenLane'
PYTHONPATH=. ./venv/bin/python3 -m pip install --upgrade --no-cache-dir 'volare==0.12.3'
Requirement already satisfied: volare==0.12.3 in ./venv/lib/python3.12/site-packages (0.20.6)
Requirement already satisfied: click==9.0 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (8.2.1)
Requirement already satisfied: httpx==0.29.0 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (0.28.1)
Requirement already satisfied: pcpp==2.1.2 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (1.30)
Requirement already satisfied: pyyaml<7.0 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (6.0.2)
Requirement already satisfied: rich==14.0 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (13.9.0)
Requirement already satisfied: zstandard==1.0 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (0.23.0)
Requirement already satisfied: anyio in ./venv/lib/python3.12/site-packages (from httpx==0.29.0) (4.9.0)
Requirement already satisfied: certifi in ./venv/lib/python3.12/site-packages (from httpx==0.29.0) (2025.4.26)
Requirement already satisfied: httpcore==1.0 in ./venv/lib/python3.12/site-packages (from httpx==0.29.0) (1.0.9)
Requirement already satisfied: idna in ./venv/lib/python3.12/site-packages (from httpx==0.29.0) (3.10)
Requirement already satisfied: h11==0.16 in ./venv/lib/python3.12/site-packages (from httpcore==1.0) (0.16.0)
Requirement already satisfied: markdown-it-py==2.0 in ./venv/lib/python3.12/site-packages (from rich==14.0) (3.0.0)
Requirement already satisfied: pygments==3.0 in ./venv/lib/python3.12/site-packages (from rich==14.0) (2.19.1)
Requirement already satisfied: mdurl==0.1 in ./venv/lib/python3.12/site-packages (from markdown-it-py==2.0) (0.1.2)
Requirement already satisfied: sniffio==1.1 in ./venv/lib/python3.12/site-packages (from anyio-->httpx==0.29.0) (1.3.1)
Requirement already satisfied: typing_extensions==4.5 in ./venv/lib/python3.12/site-packages (from anyio-->httpx==0.29.0) (4.14.0)
./venv/bin/volare enable --pdk sky130
Version bdc9412b3e468182d01b7cf63378e86c6e9c9a enabled for the sky130 PDK.
```

## i2c\_master\_top RTL Dosyasını İndirmek.

Şimdi de “i2c\_master\_top” rtl dosyasını Burak Aykenar’ın reposundan edindik.



Zip olarak indirip çıkarttık.



LICENSE	9.06.2025 17:23	Dosya	12 KB
README.md	9.06.2025 17:23	Markdown Kayna...	10 KB
designs	9.06.2025 17:23	Dosya klasörü	

picov32	9.06.2025 17:23	Dosya klasörü	
picosoc_mem	9.06.2025 17:23	Dosya klasörü	
picosoc_method1	9.06.2025 17:23	Dosya klasörü	
picosoc_method3	9.06.2025 17:23	Dosya klasörü	
simpleuart	9.06.2025 17:23	Dosya klasörü	
spimemio	9.06.2025 17:23	Dosya klasörü	

Designs içersindeki bu dosyaları

<\\wsl.localhost\Ubuntu-24.04\home\smfse\openlanemarmara\OpenLane\designs>

Adresine kopyaladık.

“make mount ” demekle desing içerisindeki RTL dosyalarını GDS formatına dönüşümünü başlattık. Ancak edindiğimiz bu dosyalarda verilog dosyaları hatalı yazılmış bu hatalı temizleyecek kodu gpt den istedik.

```
smfse@Sabri:~/openlanemarmara/OpenLane$ make
make[1]: Entering directory '/home/smfse/openlanemarmara/OpenLane'
e73fb3c57e687a8023fcd4dcfd1566ecd478362a: Pulling from #fabless/openlane
Digest: sha256:26719ced98c315b8b4ad7b9dc3e9a176991cea4c3f3282668d6d8d0f0cae229
Status: Image is up to date for #fabless/openlane:e73fb3c57e687a8023fcd4dcfd1566ecd478362a
docker: io/fabless/openlane:e73fb3c57e687a8023fcd4dcfd1566ecd478362a
make[1]: Leaving directory '/home/smfse/openlanemarmara/OpenLane'
PYTHONPATH= ./venv/bin/python3 -m pip install --upgrade --no-cache-dir 'volare==0.12.3'
Requirement already satisfied: volare==0.12.3 in ./venv/lib/python3.12/site-packages (0.28.6)
Requirement already satisfied: click==9.0.0 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (8.2.1)
Requirement already satisfied: httpx==0.29.0 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (0.28.1)
Requirement already satisfied: pcppe2==1.2 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (1.30)
Requirement already satisfied: pyyaml<7,>=5 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (6.0.2)
Requirement already satisfied: rich==14.0.0 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (13.9.0)
Requirement already satisfied: zstandard==1.0 in ./venv/lib/python3.12/site-packages (from volare==0.12.3) (0.23.0)
Requirement already satisfied: anyio in ./venv/lib/python3.12/site-packages (from httpx==0.29.0) (4.9.0)
Requirement already satisfied: certifi in ./venv/lib/python3.12/site-packages (from httpx==0.29.0) (2025.4.26)
Requirement already satisfied: httpcore==1.0 in ./venv/lib/python3.12/site-packages (from httpx==0.29.0) (1.0.9)
Requirement already satisfied: idna in ./venv/lib/python3.12/site-packages (from httpx==0.29.0) (3.10)
Requirement already satisfied: h11==0.16 in ./venv/lib/python3.12/site-packages (from httpcore==1.0) (0.16.0)
Requirement already satisfied: markdown-it-py==2.0 in ./venv/lib/python3.12/site-packages (from rich==14.0.0) (3.0.0)
Requirement already satisfied: pygments==3.0.0 in ./venv/lib/python3.12/site-packages (from rich==14.0.0) (2.19.1)
Requirement already satisfied: mdurl==0.1 in ./venv/lib/python3.12/site-packages (from markdown-it-py==2.0) (0.1.2)
Requirement already satisfied: sniffio==1.1 in ./venv/lib/python3.12/site-packages (from anyio->httpx==0.29.0) (1.3.1)
Requirement already satisfied: typing-extensions==4.5 in ./venv/lib/python3.12/site-packages (from anyio->httpx==0.29.0) (4.14.0)
./venv/bin/volare enable --pdk sky130
Version bdc9412b3e468c182d01b7cf6377be06ced9c9a enabled for the sky130 PDK.
smfse@Sabri:~/openlanemarmara/OpenLane$ make mount
cd /home/smfse/openlanemarmara/OpenLane && \
docker run --rm -v /home/smfse/home/smfse:/home/smfse -v /home/smfse/openlanemarmara/OpenLane:/openlane -v /home/smfse/openlanemarmara/OpenLane/empty:/openlane/install -v /home/smfse/volare:/home/smfse/volare -e PDK_ROOTS=/home/smfse/volare -e PDK=sky130A --user 1000:1000 -e DISPLAY=:0 -v /tmp/.X11-unix:/tmp/.X11-unix -v /home/smfse/.Xauthority:/.Xauthority --network host --security-opt seccomp=unconfined -ti #fabless/openlane:e73fb3c57e687a8023fcd4dcfd1566ecd478362a-amd64
OpenLane Container (1.1.1):/openlane% ./flow.tcl -design i2c_master_top
OpenLane v2.1.1 (e73fb3c57e687a8023fcd4dcfd1566ecd478362a)
All rights reserved. (c) 2020-2024 Efabless Corporation and contributors.
Available under the Apache License, version 2.0. See the LICENSE file for more details.
[ERROR]: Design i2c_master_top not found
OpenLane Container (1.1.1):/openlane% exit
make: *** [Makefile:137: mount] Error 255
smfse@Sabri:~/openlanemarmara/OpenLane$
```

clean\_verilog.py

nano ile clean\_verilog.py uzantılı dosya oluşturduk.

```
smfse@Sabri: ~/openlanemar X + v
smfse@Sabri:~$ cd openlanemarmara
smfse@Sabri:~/openlanemarmara$ cd OpenLane
smfse@Sabri:~/openlanemarmara/OpenLane$ nano clean_verilog.py
```



*İçerisine aşağıdaki kodu yapıştırıp çıktık.*

```
import os
import re

# Hangi dizini temizleyecek
INPUT_DIR = "designs/i2c_master_top"
OUTPUT_DIR = "designs/i2c_master_top_clean"
# Çıktı dizinini oluştur
os.makedirs(OUTPUT_DIR, exist_ok=True)

# Verilog dosyalarını bul
verilog_files = []
for root, dirs, files in os.walk(INPUT_DIR):
    for file in files:
        if file.endswith(".v"):
            verilog_files.append(os.path.join(root, file))

# Kalıp: delay pattern
delay_pattern = re.compile(r'#\s*[0-9]+')

# Temizle ve yeni dosyaya yaz
for filepath in verilog_files:
    filename = os.path.basename(filepath)
    output_path = os.path.join(OUTPUT_DIR, filename)
    with open(filepath, "r", encoding="utf-8", errors="ignore") as f:
        lines = f.readlines()
    cleaned_lines = []
    skip_initial_block = False
    initial_block_level = 0
    for line in lines:
        stripped = line.strip()
```



```
# Initial block başlarsa → skip başlat
if re.match(r'initial\s+begin', stripped):
    skip_initial_block = True
    initial_block_level = 1
    continue

# Eğer initial block içindeyse → end arayana kadar atla
if skip_initial_block:
    if "begin" in stripped:
        initial_block_level += stripped.count("begin")
    if "end" in stripped:
        initial_block_level -= stripped.count("end")
        if initial_block_level <= 0:
            skip_initial_block = False
    continue # skip this line

# Wait satırlarını atla
if re.search(r'\bwait\b', stripped):
    continue

# Delay (#) kaldır
line_no_delay = delay_pattern.sub("", line)
cleaned_lines.append(line_no_delay)

# Temiz dosyayı yaz
with open(output_path, "w", encoding="utf-8") as f:
    f.writelines(cleaned_lines)

print(f"\nTemizleme tamamlandı! → Çıktı dizini: {OUTPUT_DIR}")
```

Yapıştırırken importu düzgün kopyalamıyor onu düzenlemeyi unutmayalım.



```
GNU nano 7.2 clean_verilog.py
import os
import re

# Hangi dizini temizleyecek
INPUT_DIR = "designs/i2c_master_top"
OUTPUT_DIR = "designs/i2c_master_top_clean"

# Çıktı dizini oluştur
os.makedirs(OUTPUT_DIR, exist_ok=True)

# Verilog dosyalarını bul
verilog_files = []
for root, dirs, files in os.walk(INPUT_DIR):
    for file in files:
        if file.endswith(".v"):
            verilog_files.append(os.path.join(root, file))

# Kalıp: delay pattern
delay_pattern = re.compile(r'#\s*[0-9]+\s+')

# Temizle ve yeni dosyaya yaz
```

```
GNU nano 7.2 clean_verilog.py
import os
import re
```

Bu kod i2c\_master\_top içerisindeki verilog dosyalarını istediğimiz doğrultusunda temizleyip i2c\_master\_top\_clean olarak kaydediyor.

Aslında “i2c\_master\_top” içerisinde bazı yerler farklı yazılmış 1’ler ‘#1’ olarak verilmişti bu .py dosyası bize tek tek bunları temizlemekle uğraştırmadan bizim için temizlemiş oldu.

Çıktısından örnek bir kısım önceki ve sonraki hali:

```
start_b:
begin
    c_state <= #1 start_c;
    scl_oen <= #1 1'b1; // set SCL high
    sda_oen <= #1 1'b1; // keep SDA high
    sda_chk <= #1 1'b0; // don't check SDA output
end
```

```
start_b:
begin
    c_state <= start_c;
    scl_oen <= 1'b1; // set SCL high
    sda_oen <= 1'b1; // keep SDA high
    sda_chk <= 1'b0; // don't check SDA output
end
```

```
smfse@Sabri:~/openlanemarmara/OpenLane$ nano clean_verilog.py
smfse@Sabri:~/openlanemarmara/OpenLane$ python3 clean_verilog.py

Temizleme tamamlandı! → Çıktı dizini: designs/i2c_master_top_clean
smfse@Sabri:~/openlanemarmara/OpenLane$ |
```

Ardından temizlenmiş dosyalar için belirttiği klasör yoluna gidiyoruz.

[\\wsl.localhost\Ubuntu-24.04\home\smfse\openlanemarmara\OpenLane\designs\i2c\\_master\\_top\\_clean](#) adresinde temizlenmiş verilog dosyaları mevcut olacak.

### Neden Bu temizliği Yaptık

OpenLane içinde Verilator linter çalışıyor, bu aşama Verilog kodunda sentaks ve yapısal sorunları buluyor. Verilator, donanımsal olmayan kodları (yani simülasyon için olan kodları) hata olarak algılıyor.

Bizim amacımız: GDSII üretmek, bu yüzden sadece sentaks ve donanımsal anlamda geçerli olan kodlar bırakmamız gerekiyordu.

ci	9.06.2025 20:21	Dosya klasörü	
i2c_master_top	9.06.2025 20:56	Dosya klasörü	
i2c_master_top_clean	9.06.2025 20:58	Dosya klasörü	
mem_decode	9.06.2025 20:56	Dosya klasörü	
picorv32	9.06.2025 20:56	Dosya klasörü	
i2c_master_bit_ctrl.v	9.06.2025 21:12	V Dosyası	21 KB
i2c_master_byte_ctrl.v	9.06.2025 21:12	V Dosyası	11 KB
i2c_master_defines.v	9.06.2025 21:12	V Dosyası	3 KB
i2c_master_top.v	9.06.2025 21:12	V Dosyası	10 KB
sky130_fd_sc_hd_blackbox.v	9.06.2025 21:12	V Dosyası	67 KB
timescale.v	9.06.2025 21:12	V Dosyası	1 KB

i2c\_master\_top\_clean içerisindeki temiz dosyaları i2c\_master\_top içerisine eski dosyaların üzerine yapııştırıyoruz.

[\\wsl.localhost\Ubuntu-24.04\home\smfse\openlanemarmara\OpenLane\designs\i2c\\_master\\_top\src](#) bu adresteki klasördeki dosyaları değiştiriyoruz.

i2c_master_top	9.06.2025 20:56	Dosya klasörü	
src	9.06.2025 20:56	Dosya klasörü	

Burada Identifier olanları silmemiz gerekiyor.

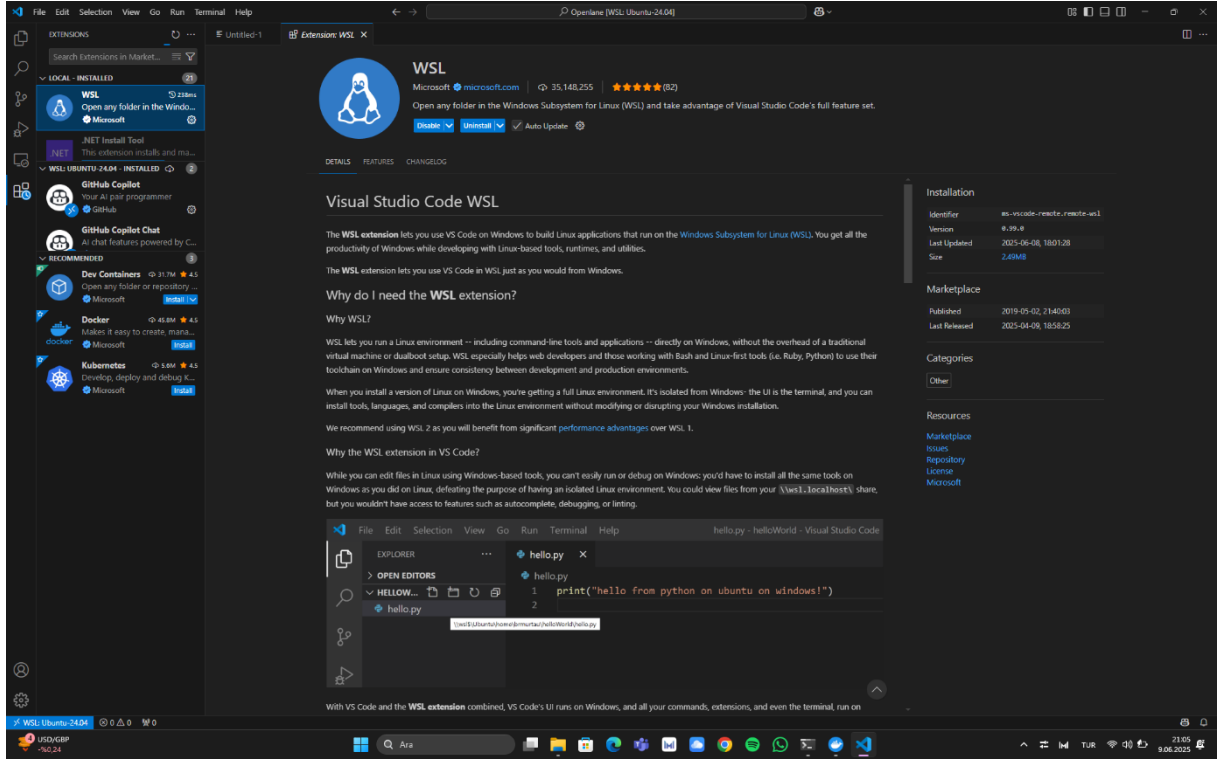
i2c_master_bit_ctrl.v	9.06.2025 17:23	V Dosyası	21 KB
i2c_master_bit_ctrl.v Zone.Identifier	9.06.2025 17:23	IDENTIFIER Dosyası	1 KB
i2c_master_byte_ctrl.v	9.06.2025 17:23	V Dosyası	11 KB
i2c_master_byte_ctrl.v Zone.Identifier	9.06.2025 17:23	IDENTIFIER Dosyası	1 KB
i2c_master_defines.v	9.06.2025 17:23	V Dosyası	3 KB
i2c_master_defines.v Zone.Identifier	9.06.2025 17:23	IDENTIFIER Dosyası	1 KB
i2c_master_top.sdc	9.06.2025 17:23	SDC Dosyası	1 KB
i2c_master_top.sdc Zone.Identifier	9.06.2025 17:23	IDENTIFIER Dosyası	1 KB
i2c_master_top.v	9.06.2025 17:23	V Dosyası	11 KB
i2c_master_top.v Zone.Identifier	9.06.2025 17:23	IDENTIFIER Dosyası	1 KB
timescale.v	9.06.2025 17:23	V Dosyası	1 KB
timescale.v Zone.Identifier	9.06.2025 17:23	IDENTIFIER Dosyası	1 KB

Tekrar make mount dediğimizde gene error aldık.

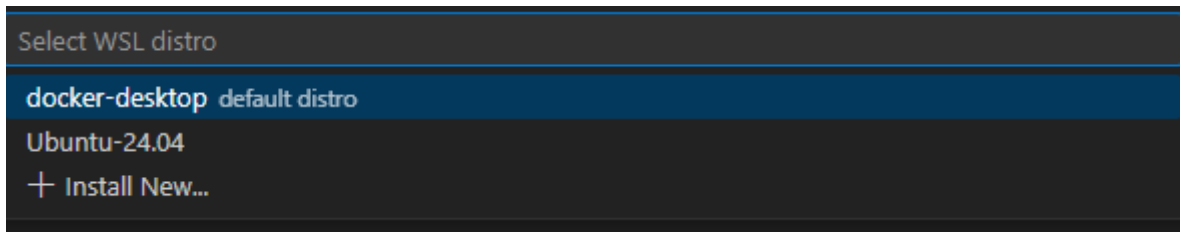
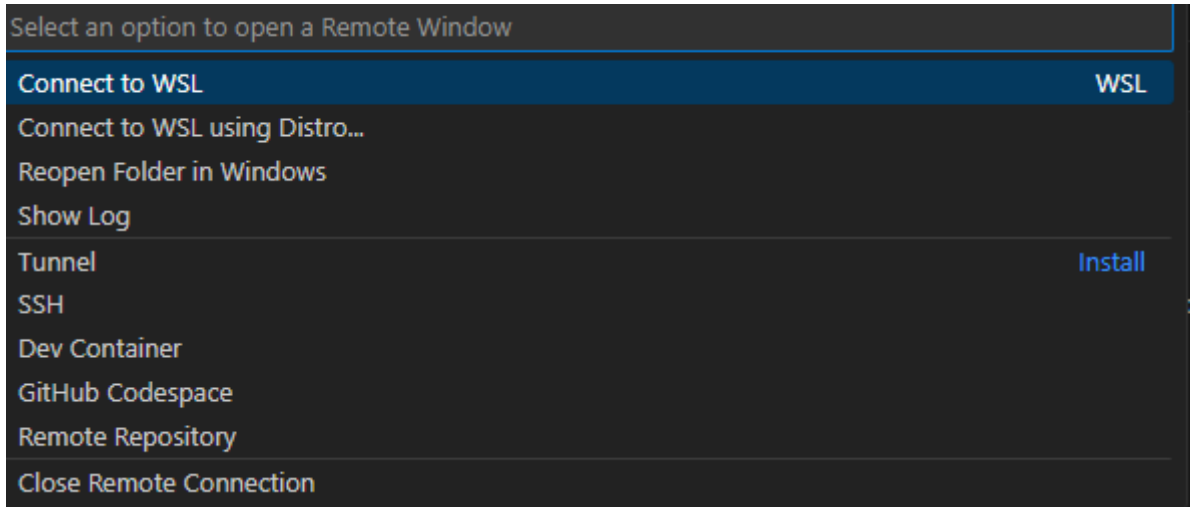
```
[INFO]: Using configuration in 'designs/i2c_master_top/config.tcl'...
[INFO]: Process Design Kit: sky130A
[INFO]: PDK Root: /home/smfse/volare
[INFO]: Standard Cell Library: sky130_fd_sc_hd
[INFO]: Optimization Standard Cell Library: sky130_fd_sc_hd
[INFO]: Run Directory: /openlane/designs/i2c_master_top/runs/RUN_2025.06.09.18.04.51
[INFO]: Saving runtime environment...
[INFO]: Preparing LEF files for the nom corner...
[INFO]: Preparing LEF files for the min corner...
[INFO]: Preparing LEF files for the max corner...
[INFO]: Running linter (Verilator) (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.04.51/logs/synthesis/linter.log)...
[ERROR]: 2 errors found by linter
[ERROR]: Step 0 (verilator_lint_check) failed with error:
-code 1 -level 0 -errorstack [INNER {invokeStkl throw_error} CALL run_verilator CALL run_verilator_step CALL {run_non_interactive_mode -design i2c_master_top}] -errorcode 1
while executing
"throw_error"
    (procedure "run_verilator" line 86)
    invoked from within
    "run_verilator"
    (procedure "run_verilator_step" line 3)
    invoked from within
    "run_verilator_step" -errorline 1
[INFO]: Saving current set of views in 'designs/i2c_master_top/runs/RUN_2025.06.09.18.04.51/results/final'...
[INFO]: Generating final set of reports...
[ERROR]: Failed to create manufacturability and metric reports:
[ERROR]: Traceback (most recent call last):
  File "/openlane/scripts/generate_reports.py", line 184, in <module>
    cli()
  File "/nix/store/8ywrzlg8yyqrf2sfvbgg8vsxhj80ry7-python3-3.11.9-env/lib/python3.11/site-packages/click/core.py", line 1157, in __call__
    return self.main(*args, **kwargs)
    ~~~~~^~~~~~
  File "/nix/store/8ywrzlg8yyqrf2sfvbgg8vsxhj80ry7-python3-3.11.9-env/lib/python3.11/site-packages/click/core.py", line 1078, in main
    rv = self.invoke(ctx)
    ~~~^~~~~~
  File "/nix/store/8ywrzlg8yyqrf2sfvbgg8vsxhj80ry7-python3-3.11.9-env/lib/python3.11/site-packages/click/core.py", line 1434, in invoke
    return ctx.invoke(self.callback, **ctx.params)
    ~~~~~^~~~~~
  File "/nix/store/8ywrzlg8yyqrf2sfvbgg8vsxhj80ry7-python3-3.11.9-env/lib/python3.11/site-packages/click/core.py", line 783, in invoke
    return __callback(*args, **kwargs)
    ~~~~~^~~~~~
```

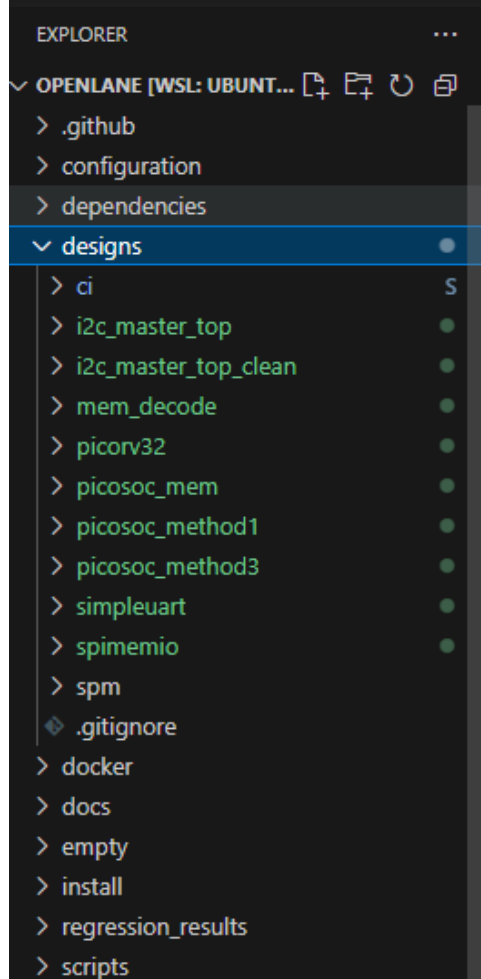
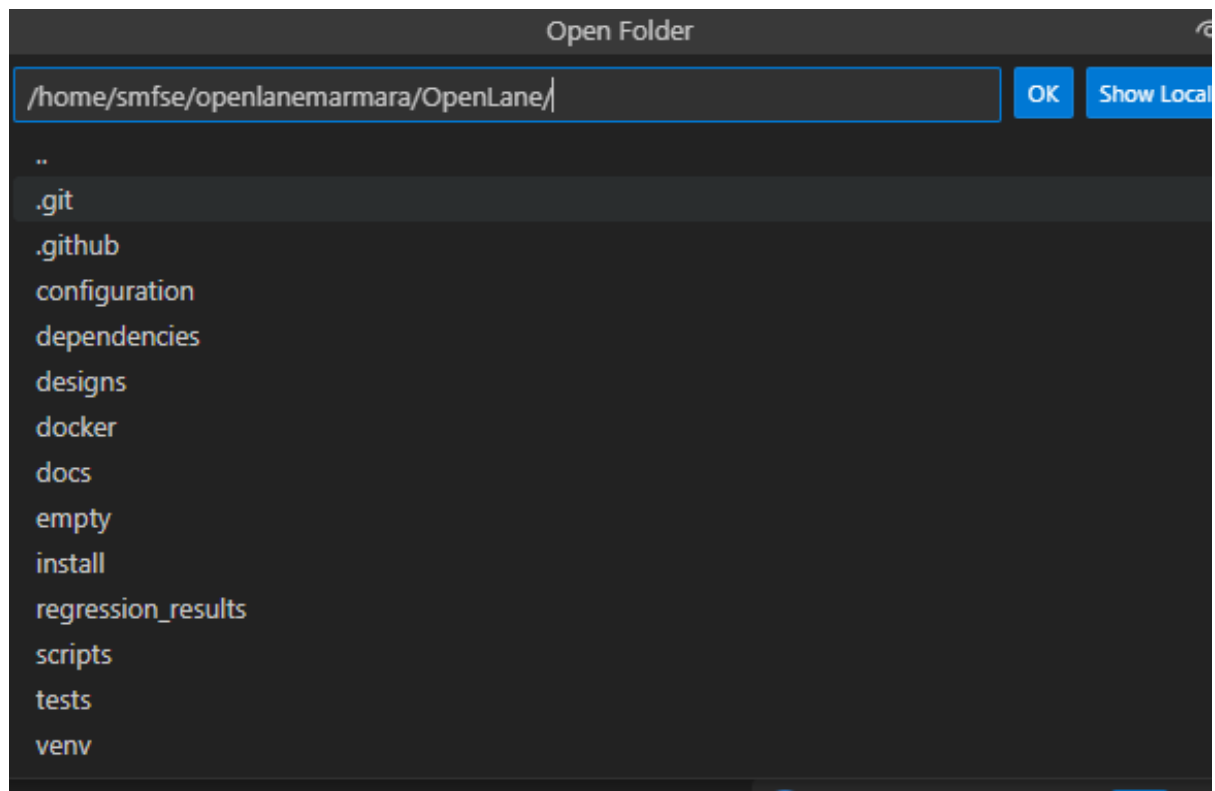
Bu hata verilog dosyasında bir kısmın eksik olmasından kaynaklanıyordu bunu verilog dosyasını elle düzenleyerek çözdük bunun için verilog dosyasının içerisine erişmemiz gerektiğini bunu da visual studio ile yaptık. Ancak WSL kullandığımız için vs kodun dosyalara erişimi VS code da WSL eklentisini kurarak yaptık.

# VS Code WSL Entegrasyonu



Gerekli konum resimlerdeki gösterildiği üzere gittik.





Burada designs içerisindeki src dosyasının altında “i2c\_master\_bit\_ctrl.v” verilog koduna bir ekleme yaptık.

```
src
├── i2c_master_bit_ctrl.v
├── i2c_master_bit_ctrl.vZone.Identifier
├── i2c_master_byte_ctrl.v
├── i2c_master_byte_ctrl.vZone.Identifier
├── i2c_master_defines.v
├── i2c_master_defines.vZone.Identifier
├── i2c_master_top.sdc
├── i2c_master_top.v
├── i2c_master_top.vZone.Identifier
├── timescale.v
├── timescale.vZone.Identifier
├── config.tcl
├── config.tclZone.Identifier
├── pin_order.cfg
├── pin_order.cfgZone.Identifier
└── i2c_master_top_clean

172 reg [ 1:0] cSCL, cSDA; // capture SCL and SDA
173 reg [ 2:0] fSCL, fSDA; // SCL and SDA filter inputs
174 reg sSCL, sSDA; // filtered and synchronized SCL and SDA inputs
175 reg dSCL, dSDA; // delayed versions of sSCL and sSDA
176 reg dsc1_oen; // delayed scl_oen
177 reg sda_chk; // check SDA output (Multi-master arbitration)
178 reg clk_en; // clock generation signals
179 reg [15:0] cnt; // clock divider counter (synthesis)
180 reg [13:0] filter_cnt; // clock divider for filter
181
```

Koda “reg slave wait ” eklemesini yaptık ve kaydettik.

```
i2c_master_bit_ctrl.v U x
designs > i2c_master_top > src > i2c_master_bit_ctrl.v
143 module i2c_master_bit_ctrl (
144     input sda_i, // i2c data line input
145     output sda_o, // i2c data line output
146     output reg sda_oen // i2c data line output enable (active low)
147 );
148
149 //
150 // variable declarations
151 //
152
153 reg [ 1:0] cSCL, cSDA; // capture SCL and SDA
154 reg [ 2:0] fSCL, fSDA; // SCL and SDA filter inputs
155 reg sSCL, sSDA; // filtered and synchronized SCL and SDA inputs
156 reg dSCL, dSDA; // delayed versions of sSCL and sSDA
157 reg dsc1_oen; // delayed scl_oen
158 reg sda_chk; // check SDA output (Multi-master arbitration)
159 reg clk_en; // clock generation signals
160 reg slave_wait; // slave inserts wait states
161 reg [15:0] cnt; // clock divider counter (synthesis)
162 reg [13:0] filter_cnt; // clock divider for filter
163
```

Otemizliklerde delay (#) kaldırırken veya initial blokları silerken bu slave\_wait tanımını ya da kullanımını yanlışlıkla kaldırabiliyor veya eksik bırakabiliyor.

Mesela slave\_wait sinyali clock enable (clk\_en) üretirken kullanılıyor ve bu i2c zamanlamasında çok kritik:

## RTL to GDSII ASIC Tasarımının Başlaması

Tekrar make mount dediğimizde asicc tasarım aşamalarının hepsini yapmış olduk.

```
[INFO]: Saving runtime environment...
[ERROR]: Flow failed
OpenLane Container (1.1.1): /openlane% /flow tcl -design i2c_master_top
OpenLane v1.1.1 (c737b3c57ad6f400223fcd4dcfd1d6dc0478262a)
All rights reserved. (c) 2020-2024 Efabless Corporation and contributors.
Available under the Apache License, version 2.0. See the LICENSE file for more details.

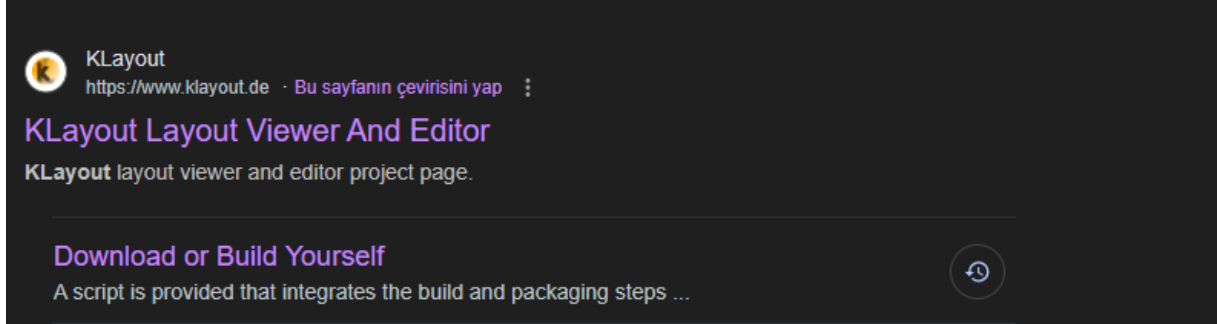
[INFO]: Using configuration in 'designs/i2c_master_top/config.tcl'...
[INFO]: Process Design Kit: sky130A
[INFO]: PDK Root: /home/sarfer/volare
[INFO]: Standard Cell Library: sky130_fd_sc_hd
[INFO]: Optimization Standard Cell Library: sky130_fd_sc_hd
[INFO]: Run Directory: /openlane/designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34
[INFO]: Saving runtime environment...
[INFO]: Preparing LEF files for the nom corner...
[INFO]: Preparing LEF files for the min corner...
[INFO]: Preparing LEF files for the max corner...
[INFO]: Running Linter (Verilator) (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/synthesis/linter.log)...
[INFO]: 0 errors found by Linter
[WARNING]: 876 warnings found by Linter
[STEP 1]
[INFO]: Running Synthesis (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/synthesis/1-synthesis.log)...
[STEP 2]
[INFO]: Running Single-Corner Static Timing Analysis (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/synthesis/2-sta.log)...
[INFO]: Creating a netlist with power/ground pins.
[STEP 3]
[INFO]: Running Initial Floorplanning (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/floorplan/3-initial_fp.log)...
[INFO]: Floorplanned with width 288.00 and height 277.00.
[STEP 4]
[INFO]: Running IO Placement (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/floorplan/4-place_io.log)...
[STEP 5]
[INFO]: Running Tap/Decap Insertion (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/floorplan/5-tap.log)...
[INFO]: Power planning with power [vccd1] and ground [vssd1]...
[STEP 6]
[INFO]: Generating PDN (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/floorplan/6-pdn.log)...
[STEP 7]
[INFO]: Running Global Placement (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/placement/6-global.log)...
[STEP 8]
[INFO]: Running Single-Corner Static Timing Analysis (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/placement/8-gpl_sta.log)...
[STEP 9]
[INFO]: Running Placement Resizer Design Optimizations (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/placement/9-resizer.log)...
[STEP 10]
[INFO]: Running Detailed Placement (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/placement/10-detailed.log)...
[STEP 11]
[INFO]: Running Single-Corner Static Timing Analysis (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/placement/11-dpl_sta.log)...
[STEP 12]
[INFO]: Running Clock Tree Synthesis (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/cts/12-cts.log)...
[STEP 13]
[INFO]: Running Single-Corner Static Timing Analysis (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/cts/13-cts_sta.log)...
[STEP 14]
[INFO]: Running Placement Resizer Timing Optimizations (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/cts/14-resizer.log)...
[STEP 15]
[INFO]: Running Global Routing Resizer Design Optimizations (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/routing/15-resizer_design.log)...
[STEP 16]
[INFO]: Running Single-Corner Static Timing Analysis (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/routing/16-rsz_design_sta.log)...
[STEP 17]
[INFO]: Running Global Routing Resizer Timing Optimizations (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/routing/17-resizer_timing.log)...
[STEP 18]
[STEP 23]
[INFO]: Running Detailed Routing (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/routing/23-detailed.log)...
[INFO]: No Magic DRC violations after detailed routing.
[STEP 24]
[INFO]: Checking Wire Lengths (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/routing/24-wire_lengths.log)...
[STEP 25]
[INFO]: Running SPFE Extraction at the min process corner (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/25-parasitics_extraction.min.log)...
[STEP 26]
[INFO]: Running Multi-Corner Static Timing Analysis at the min process corner (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/26-rcx_msta.min.log)...
[STEP 27]
[INFO]: Running SPFE Extraction at the max process corner (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/27-parasitics_extraction.max.log)...
[STEP 28]
[INFO]: Running Multi-Corner Static Timing Analysis at the max process corner (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/28-rcx_msta.max.log)...
[STEP 29]
[INFO]: Running SPFE Extraction at the nom process corner (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/29-parasitics_extraction.nom.log)...
[STEP 30]
[INFO]: Running Multi-Corner Static Timing Analysis at the nom process corner (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/30-rcx_msta.nom.log)...
[STEP 31]
[INFO]: Running Magic to generate various views...
[INFO]: Streaming out GDSII with Magic (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/31-gdsii.log)...
[INFO]: Generating MAGLEF views...
[INFO]: Generating lef with Magic (/openlane/designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/31-lef.log)...
[STEP 32]
[INFO]: Streaming out GDSII with kLayout (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/32-gdsii-klayout.log)...
[STEP 33]
[INFO]: Running XOR on the layouts using kLayout (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/33-xor.log)...
[INFO]: No XOR differences between kLayout and Magic gds.
[STEP 34]
[INFO]: Running Magic Spice Export from LEF (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/34-spice.log)...
[STEP 35]
[INFO]: Writing Powered Verilog (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/35-write_powered_def.log, designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/35-write_powered_verilog.log)...
[STEP 36]
[INFO]: Writing Verilog (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/35-write_powered_verilog.log)...
[STEP 37]
[INFO]: Running LVS (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/37-lvs.lef.log)...
[STEP 38]
[INFO]: Running Magic DRC (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/38-drc.log)...
[INFO]: Converting Magic DRC database to various tool-readable formats...
[INFO]: No Magic DRC violations after GDS streaming out.
[STEP 39]
[INFO]: Running kLayout DRC (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/39-drc-klayout.log)...
[INFO]: No kLayout DRC violations after GDS streaming out.
[STEP 40]
[INFO]: Running OpenROAD Antenna Rule Checker (log: designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/logs/signoff/40-arc.log)...
[INFO]: Saving current set of views in 'designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/results/final'...
[INFO]: Saving runtime environment...
[INFO]: Generating final set of reports...
[INFO]: Created manufacturability report at 'designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/reports/manufacturability.rpt'.
[INFO]: Created metrics report at 'designs/i2c_master_top/runs/RUN_2025.06.09.18.16.34/reports/metrics.csv'.
[INFO]: There are no max slew, max fanout or max capacitance violations in the design at the Typical corner.
[INFO]: There are no hold violations in the design at the Typical corner.
[INFO]: There are no setup violations in the design at the Typical corner.
[SUCCESS]: Flow complete
[INFO]: Note that the following warnings have been generated:
[WARNING]: 876 warnings found by Linter
OpenLane Container (1.1.1): /openlane%
```

Burada flow complete diyerek gds dosyasını oluşturdu demek burada 876 tane linter uyarısı aldık. İstersek bu uyarıları tek tek elle düzeltebiliriz. Bunu config dosyasında değerleri değiştirerek yapabiliriz.



## KLayout Programında Tasarımı Görme

.gds dosyalarını görüntülemek için Klayout programını kurmamız gerekli.

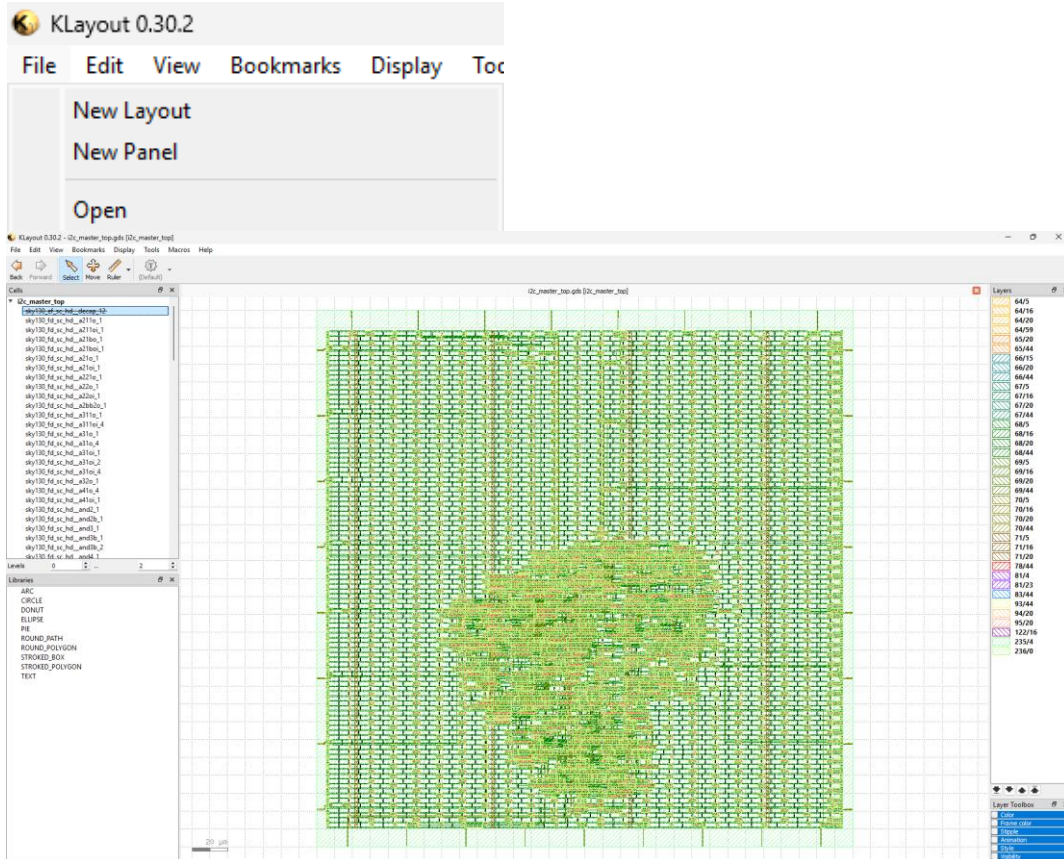


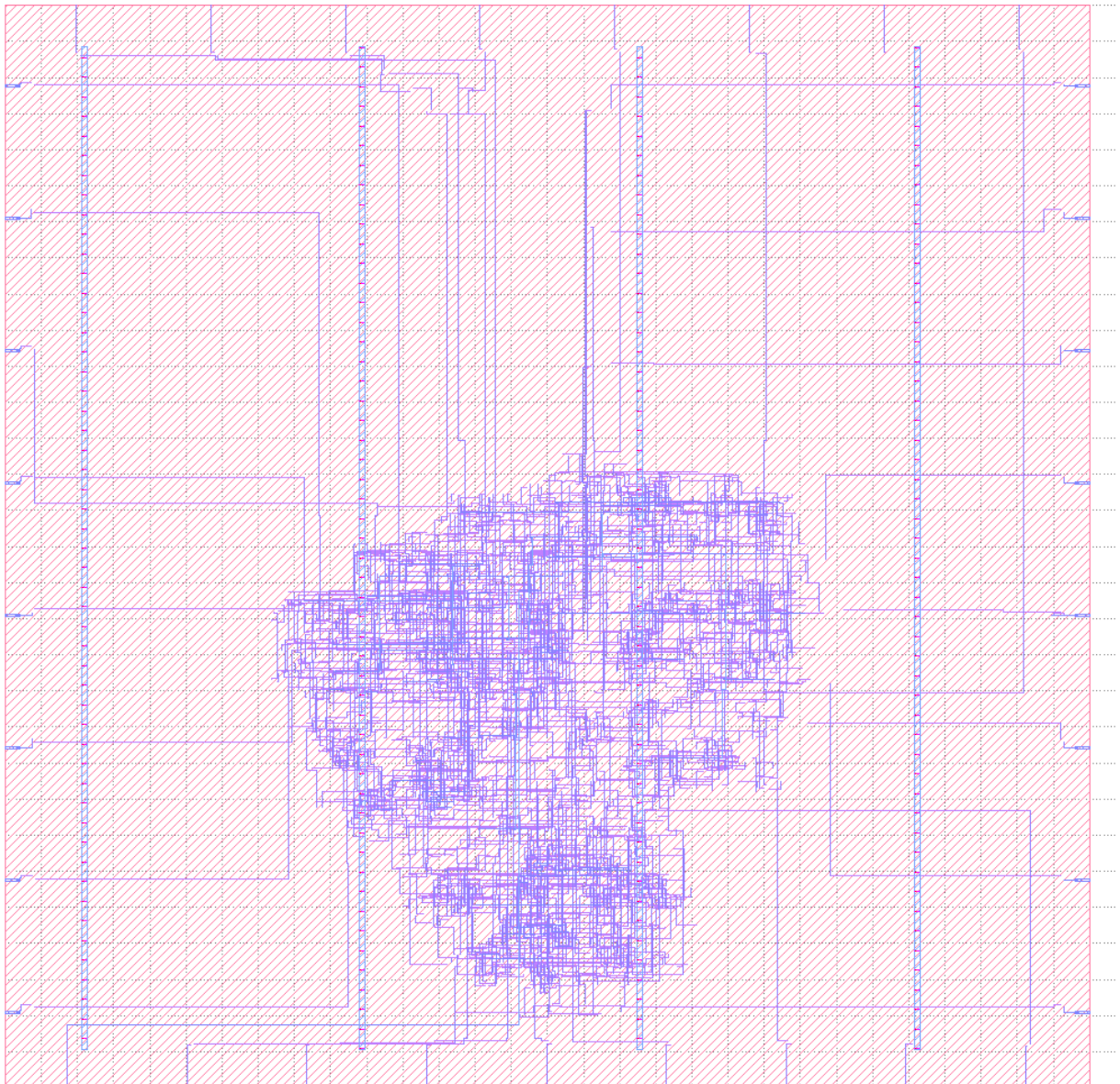
Kurulum sonrasında .gds dosyalarını görüntülememiz gerekli. Klayout programını başlattan açıp içerisinde gds dosyalarını open diyerek belirli adreslerdeki gds dosyalarını açıyoruz.

İstediğimiz gds dosyasını file open tıklayarak görebiliriz.

\\wsl.localhost\Ubuntu-

24.04\home\smfse\openlanemarmara\OpenLane\designs\i2c\_master\_top\runs\RUN\_2025.06.09\_18.16.34\results\final\gds\ i2c\_master\_top.gds





[SUCCESS]: Flow complete.

## **OpenLane Akışı**

### **- Synthesis (Sentetik)**

Bu aşamada i2c\_master\_top tasarımımızdaki Verilog kodunu (yani yazdığımız mantıksal devre tanımını) donanıma çevirecek adımı yapıyoruz. Yazdığımız Verilog kodu doğrudan fiziksel devreye çevrilemez. O yüzden önce mantıksal kapı (gate) seviyesine dönüştürülüyor. Burada Yosys aracı kullanılıyor. Sonuç olarak: Mantıksal kapı seviyesinde netlist dediğimiz bağlantı listesi çıkıyor. Böylece fiziksel olarak neyin nereye bağlanacağı belirlenmiş oluyor. Linter kontrolü burada yapılıyor. Linter, Senteze başlamadan önce Verilog kodunun hatasız olduğundan emin olmak için sentezin başında yapılır.

### **- Floorplan**

Bu adımda artık mantıksal bağlantılar belli olduğuna göre: Çipin üzerinde bu devreyi nereye yerleştireceğimizi belirliyoruz. Çipin boyutunu ayarlıyoruz. Giriş çıkış pinlerinin kenarlarda nereye konulacağını belirliyoruz. Güç ve toprak hatlarını yerleştiriyoruz.

### **- Placement**

Mantıksal hücreleri (AND kapısı, Flip-Flop gibi) fiziksel olarak çip üzerinde uygun yerlere yerleştiriyoruz. Hücrelerin sıkışmaması ve doğru bağlantı yapabilmesi için boşluklar ve düzen sağlıyoruz. Placement sonrası devre artık çip üzerinde nerede duracak tamamen belli olmuş oluyor. Burada STA yapılır.

### **Static Timing Analysis (STA)**

STA çip tasarımında zamanlama (timing) hatalarını bulmak için yapılan bir kontroldür. Saat (Clock) sinyaliyle uyumlu çalışması gereken yolların geç kalmaması gerekir. Buna setup time ve hold time diyoruz. STA aşamasında araç, her yolun geçiş süresini (delay) hesaplıyor. Hiçbir yolun geç kalıp kalmadığını geçersiz bir veri üretmediğini veya bozulma yaşanmadığını kontrol ediyor. Eğer zamanlama problemi (timing violation) çıkarsa, placement veya routing tekrar ayarlanıyor. Kısacası STA aşaması sayesinde çipin tüm sinyallerinin doğru zamanda çalıştığından emin oluyoruz.

### **- CTS (Clock Tree Synthesis)**

Çip içinde clock sinyali kullanılıyor ki tüm işlemler anda gerçekleşsin. Clock sinyali her hücreye eşit sürede ulaşacak şekilde özel yollar (clock tree) oluşturuluyor. Böylece tüm hücreler aynı anda çalışabiliyor. Clock skew dediğimiz zaman farkları minimize ediliyor. Burada da STA yapılır.

## - Routing

Artık hücrelerin yeri belli oldu. Şimdi hücreleri metal yollarla bağlanmalı. Çipin hücreleri metal yollarla bağlanıyor. Hem global routing hem de detaylı routing yapılıyor. Routing bitince çipin tüm bağlantıları tamamlanmış oluyor. Burada da STA yapılır.

## - DRC ve LVS Kontrolleri

Bu aşamada tasarımımızı kontrol ediyoruz:

DRC (Design Rule Check): Çipin üretim kurallarına uyup uymadığını kontrol ediyoruz (minimum çizgi kalınlığı, mesafe vs.).

LVS (Layout vs Schematic): Tasarımın mantıksal olarak doğru olup olmadığını kontrol ediyoruz. Yani Verilog'taki mantıkla fiziksel çizim aynı mı diye bakıyoruz.

## - GDSII Üretimi

Son olarak tasarımımızı GDSII formatına çeviriyoruz. GDSII, fabrikaya gönderilecek dosyadır. Bu dosya doğrudan çipin üretiminde kullanılır. GDSII dosyasını ayrıca Klayout aracı ile açıp görsel olarak da inceledik. Böylece GDSII dosyamız hazır hale geldi.

## Sonuç ve Değerlendirme

Bu projede openlane ile asic design yapmayı öğrendik. Bunu yaparken karşılaştığımız “docker, volare, python3.12” sorunlarını çözmeyi öğrendik. I2c\_master\_top RTL dosyasında Verilog kollarındaki hataları temizlemeyi öğrendik. OpenLane ile tasarımında ne aşamaların yapıldığını öğrendik. WSL ile Ubuntu kurup linux kullanmayı öğrendik. Klayout ile GDSII dosya formatını görmeyi öğrendik. Yani kısaca: I2C Master Top modülünü başarıyla GDSII seviyesine kadar getirdik. Flow sırasında hem tasarım hem de yazılım tarafında pek çok hata ve problem ile karşılaşmış ve çözdük. Clean Verilog işlemi sayesinde daha stabil bir OpenLane flow elde ettik. Klayout kullanarak GDSII dosyasının görsel kontrolünü de yaptık.

## Kaynakça

*Docker dektop download.* (2025, Haziran 1). Docker dektop:

[https://desktop.docker.com/win/main/amd64/Docker%20Desktop%20Installer.exe?utm\\_source=docker&utm\\_medium=webreferral&utm\\_campaign=docs-driven-download-win-amd64](https://desktop.docker.com/win/main/amd64/Docker%20Desktop%20Installer.exe?utm_source=docker&utm_medium=webreferral&utm_campaign=docs-driven-download-win-amd64) adresinden alındı

*efabless github.* (2025, Haziran 1). Github: <https://github.com/efabless/caravel> adresinden alındı

*Kaylayout download.* (2025, Haziran 1). Klayout: <https://www.klayout.org/downloads/Windows/klayout-0.30.2-win64-install.exe> adresinden alındı

*mbaykenar github.* (2025, Haziran 1). github: <https://github.com/mbaykenar/openlane-designs/tree/main/designs> adresinden alındı

*Mehmet burak aykenar chip tasarımı dersleri.* (2025, Haziran 1). Youtube:

<https://youtu.be/haTgQTjgXj8?si=y1Dap7YCeUvpbkBf> adresinden alındı

*The-OpenROAD-Project.* (2025, Haziran 1). GitHub: <https://github.com/The-OpenROAD-Project/OpenLane> adresinden alındı

## Teşekkürler

Dr. Öğr. Üyesi Sezen BAL

Öğr. Emre YAĞCI 360523012

Öğr. Sabri SEVİNÇLİ 360523033