WEEK -2 TASK

EFR32ZG28:

The EFR32ZG28 is a wireless SoC (System on Chip) developed by Silicon Labs as part of their EFR32 "Wireless Gecko" series. It is optimized for low-power and secure wireless communication, primarily used in Sub-GHz, Wi-SUN, and Proprietary RF applications.

Memory:

Flash : 512 KB
RAM : 64 KB

I/O and Peripherals

- GPIOs
- USART / I2C / SPI
- ADC / DAC
- Timers / PWM
- Crypto engine for AES-128/256, ECC, SHA-1, SHA-256

Core:

Family: **ARM Cortex-M33** in EFR32ZG28

he EFR32ZG28 SoC (System on Chip) is built around the ARM Cortex-M33 core, which is highly efficient, secure, and low-power, making it perfect for IoT and wireless communication tasks.

Clock Speed : Up to **76.8 Mhz**

Bit width : 32 bit

TrustZone® Support

ARM TrustZone splits the system into two parts:

- Secure World – for sensitive tasks (e.g., secure boot, crypto, device keys)
- Non-Secure World – for general tasks (e.g., networking, sensors)

This enables secure OTA updates, crypto processing, and safe separation between app code and system-level security.

DSP and FPU Support

- DSP (Digital Signal Processing): Allows efficient processing of signals, such as filtering, FFT, or analyzing sensor data in real-time.
    - Example: Processing environmental sensor data like gas levels, temperature, etc.
- FPU (Floating Point Unit): Hardware accelerator for floating-point math (e.g., `float`, `double`)
    - Example: Smooth BLE RSSI averaging, signal filtering, sensor fusion algorithms

CONTIKI -OS:

Contiki OS is a lightweight, open-source operating system designed for:

- IoT (Internet of Things) and wireless sensor networks
- Low-power embedded devices
- IPv6, 6LoWPAN, CoAP, RPL support (for networking in tiny devices)
- It runs directly on hardware (no other OS below it)
- Mostly written in **C language**.

Contiki-NG does **not officially support** EFR32ZG28

contiki-ng/

├── arch/ <-- Platform architecture

│ └── cpu/

│ └── arm/

│ └── cortex-m/ <-- Your core is here (Cortex-M33)

├── platform/ <-- Your EFR32ZG28 board files go here

│ └── efr32zg28/ <-- YOU create this

├── os/ <-- OS core

└── examples/ <-- For testing

CONTIKI OS INSTALLATION:

sudo apt update

sudo apt upgrade

sudo apt install git build-essential python3-pip srecord ccache

sudo apt install gcc-arm-none-eabi

openocd screen

git clone https://github.com/contiki-ng/contiki-ng.git

cd contiki-ng/

git submodule update --init –recursive

Goal

Run a basic **LED blink process** on the **EFR32ZG28** board using Contiki-NG, which does **not natively support** this platform.

```
contiki-ng/
├── arch/
│   └── platform/
│       └── efr32zg28/
│           ├── Makefile.include
│           ├── contiki-conf.h
│           ├── platform.h
│           └── dev/
│               ├── leds-arch.c
│               └── leds-arch.h
├── examples/
│   └── efr32zg28/
│       ├── blink.c
│       ├── Makefile
│       └── project-conf.h
```

Create Platform Folder

mkdir -p ~/contiki-ng/arch/platform/efr32zg28

contiki-conf.h
device_config.h
Makefile.efr32zg28
Makefile.include
platform.c
platform.h
rtimer-arch.c
rtimer-arch.h

Fix Makefile.include

Board Configuration

BOARD = efr32zg28

BOARDS = efr32zg28

Gecko SDK Path (verify this matches your installation)

```
GECKO_SDK = /home/abarnikag/SimplicityStudio/SDKs/gecko_sdk

### Device Configuration - MUST COME FIRST
CFLAGS := -DPART_NUMBER=EFR32ZG28A122F1024GM48 \
        -DDEVICE_FAMILY=efr32zg28 \
        -D__PROJECT__=\"EFR32ZG28\" \
        $(CFLAGS)

### Include Paths
CONTIKI_TARGET_DIRS += . ../../cpu/arm/cortex-m33
CFLAGS += -I$(CONTIKI)/arch/platform/$(TARGET) \
        -I$(CONTIKI)/arch/cpu/arm/cortex-m33 \
        -I$(GECKO_SDK)/platform/emlib/inc \
        -I$(GECKO_SDK)/platform/Device/SiliconLabs/EFR32ZG28/Include \
        -I$(GECKO_SDK)/hardware/kit/common/drivers \
        -I$(GECKO_SDK)/platform/CMSIS/Core/Include

### Source Files
CONTIKI_TARGET_SOURCEFILES += platform.c rtimer-arch.c

### Module Configuration
MODULES += $(CONTIKI_NG_ARCH_DIR)/platform/$(TARGET)

### Include ARM Cortex-M common makefile
include $(CONTIKI)/arch/cpu/arm/Makefile.arm
```

`blink.c` – Creating the Process

Path: `~/contiki-ng/examples/efr32zg28/blink.c`

`Makefile` – Building the Example

Path: `~/contiki-ng/examples/efr32zg28/Makefile`

Build the Blink file:

make TARGET=efr32zg28 clean

make TARGET=efr32zg28 BOARD=efr32zg28 V=1

Here I am Facing Error: