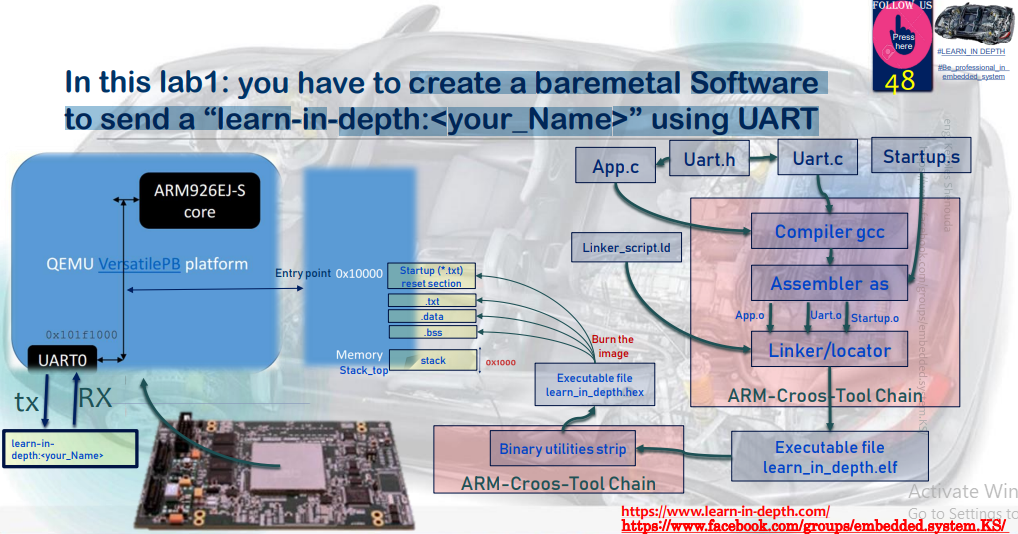
# Lab 1

# Description:

Create a bare-metal Software to send a “learn-in-depth: <<your name>>” using UART of the ARM Versatile PB board.



# Files Created:

# Analysis Files:

* uart.o
* app.o
* startup.o
* Map\_file.map
* uart.h
* uart.c
* app.c
* startup.s
* linker\_script.ld

# Executable Files:

* learn-in-depth.elf
* learn-in-depth.bin

# Git Commands Used In Compilation Process:

## To get the object files:

* $ arm-none-eabi-gcc.exe -c -g -I . -mcpu=arm926ej-s uart.c -o uart.o
* $ arm-none-eabi-gcc.exe -c -g -I . -mcpu=arm926ej-s app.c -o app.o
* $ arm-none-eabi-gcc.exe -c -g -I . -mcpu=arm926ej-s startup.c -o startup.o

## To link the object files together using linker\_script and get the .elf and .map files:

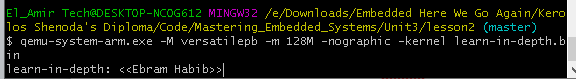
* $ arm-none-eabi-ld.exe -T linker\_script.ld app.o uart.o startup.o -o learn-in-depth.elf -Map=Map\_file.map

## To get the bin file:

* $ arm-none-eabi-objcopy.exe -O binary learn-in-depth.elf learn-in-depth.bin

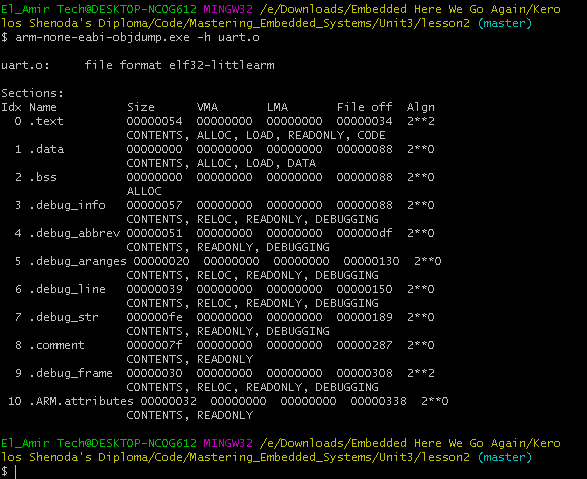
## To run the program in the QEMU Simulator (“VersatilePB physical Board”):

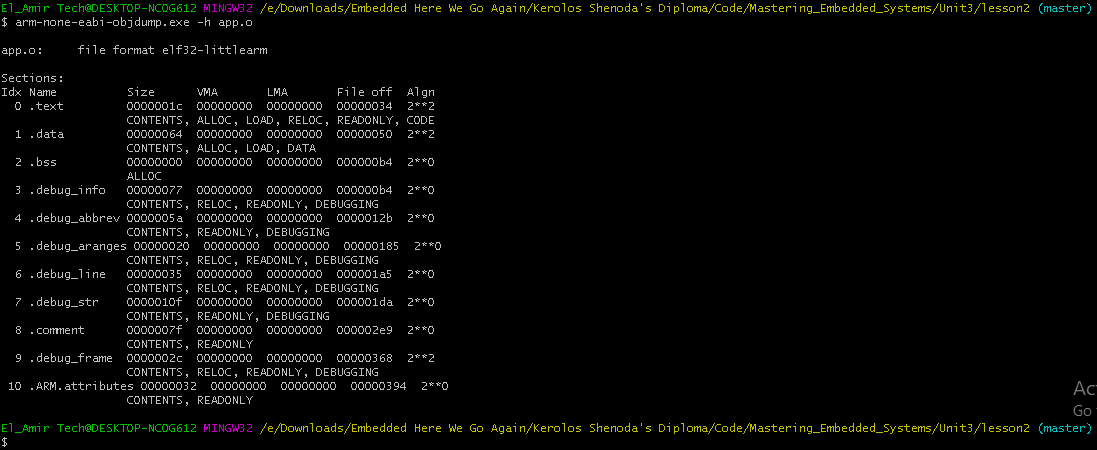
* $ qemu-system-arm.exe -M versatilepb -m 128M -nographic -kernel learn-in-depth.bin



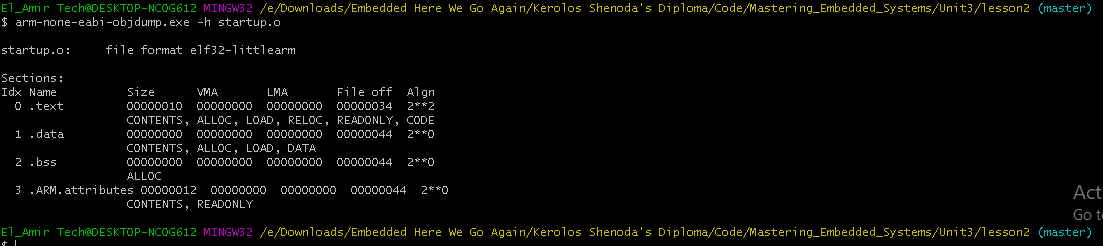
# Git Commands Used In Analysis Process:

## To display the content of the section headers:

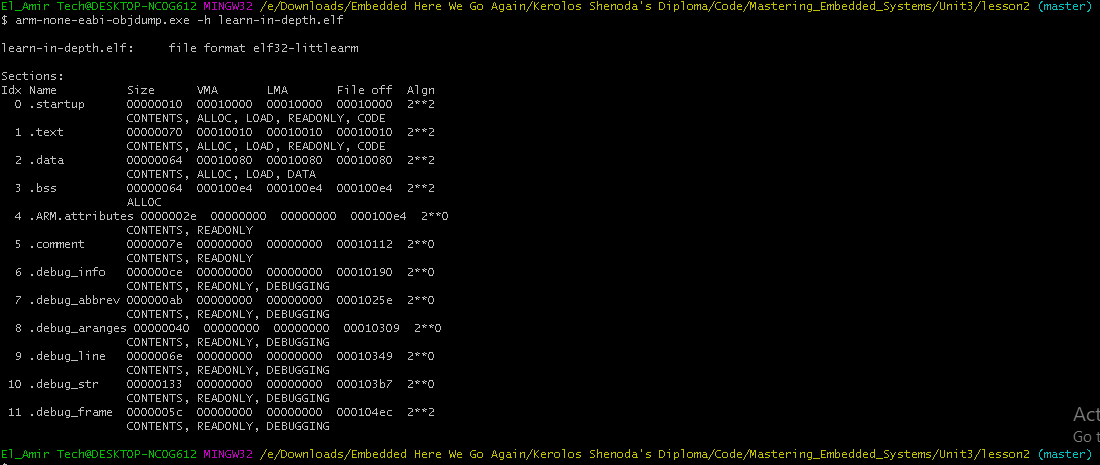
* $ arm-none-eabi-objdump.exe -h uart.o
* $ arm-none-eabi-objdump.exe -h app.o



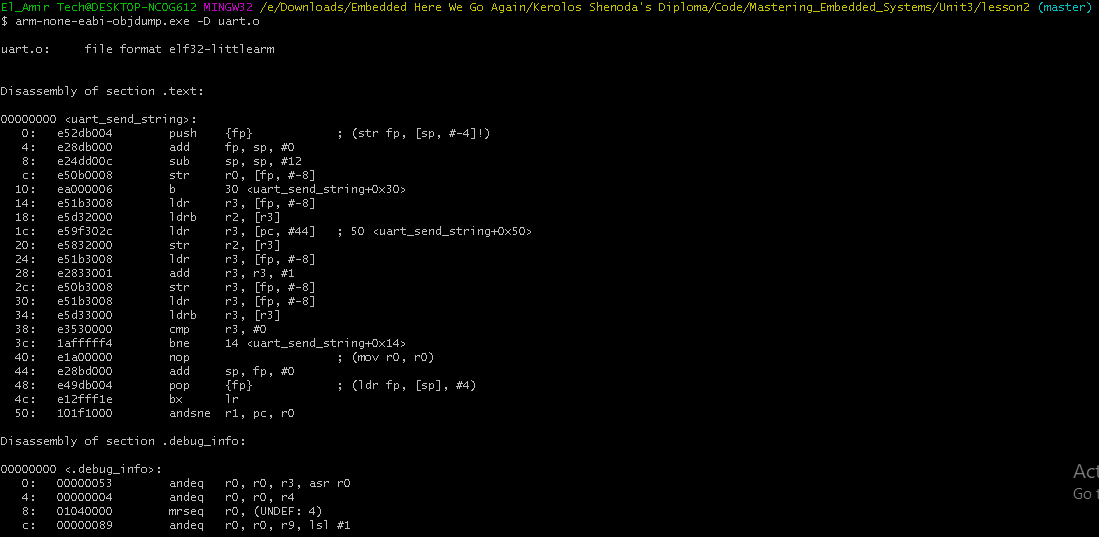
* $ arm-none-eabi-objdump.exe -h startup.o

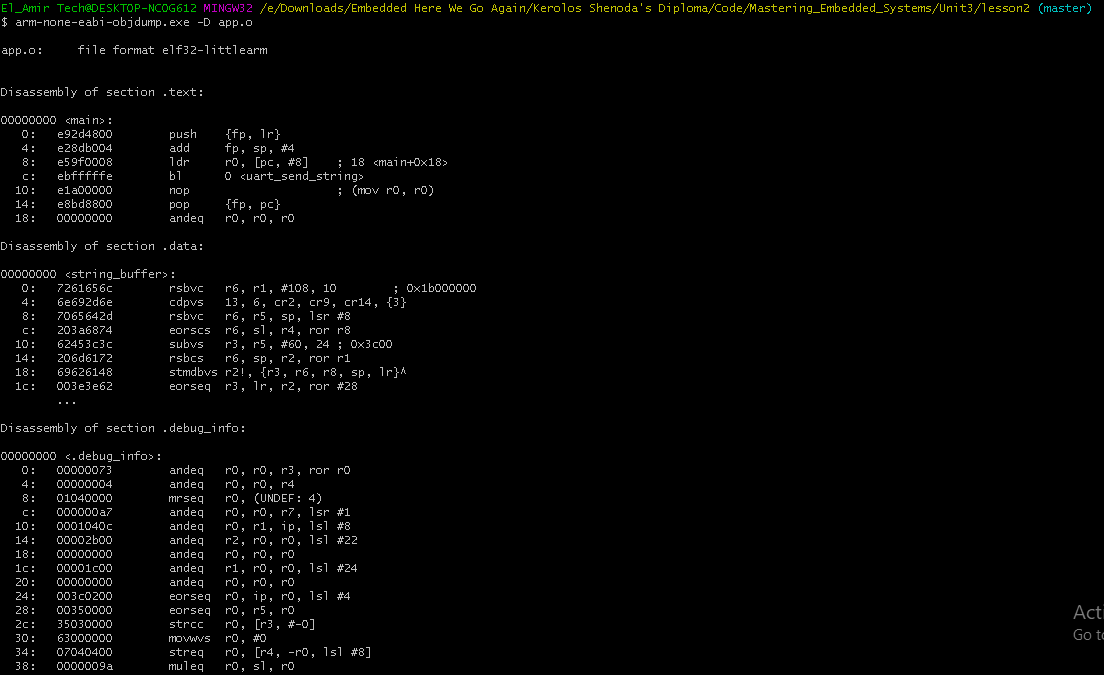


* $ arm-none-eabi-objdump.exe -h learn-in-depth.elf

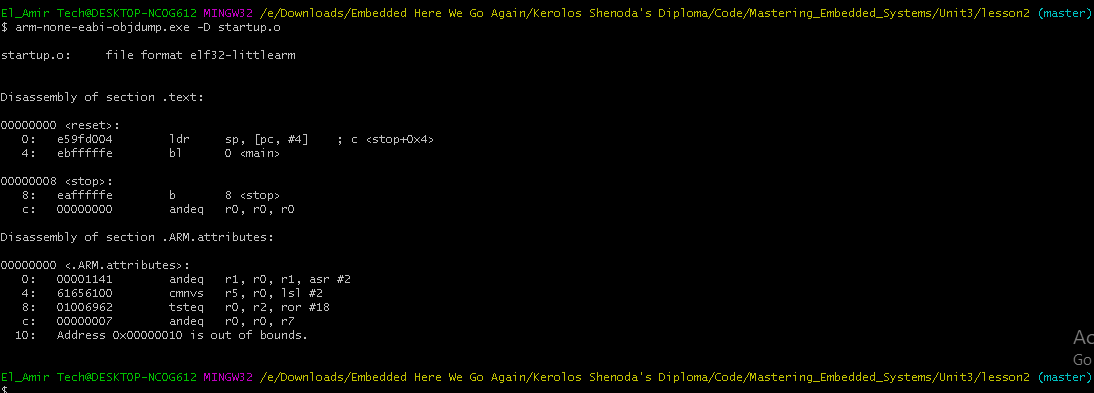


## To display the assembler content of all the sections:

* $ arm-none-eabi-objdump.exe -D uart.o
* $ arm-none-eabi-objdump.exe -D app.o

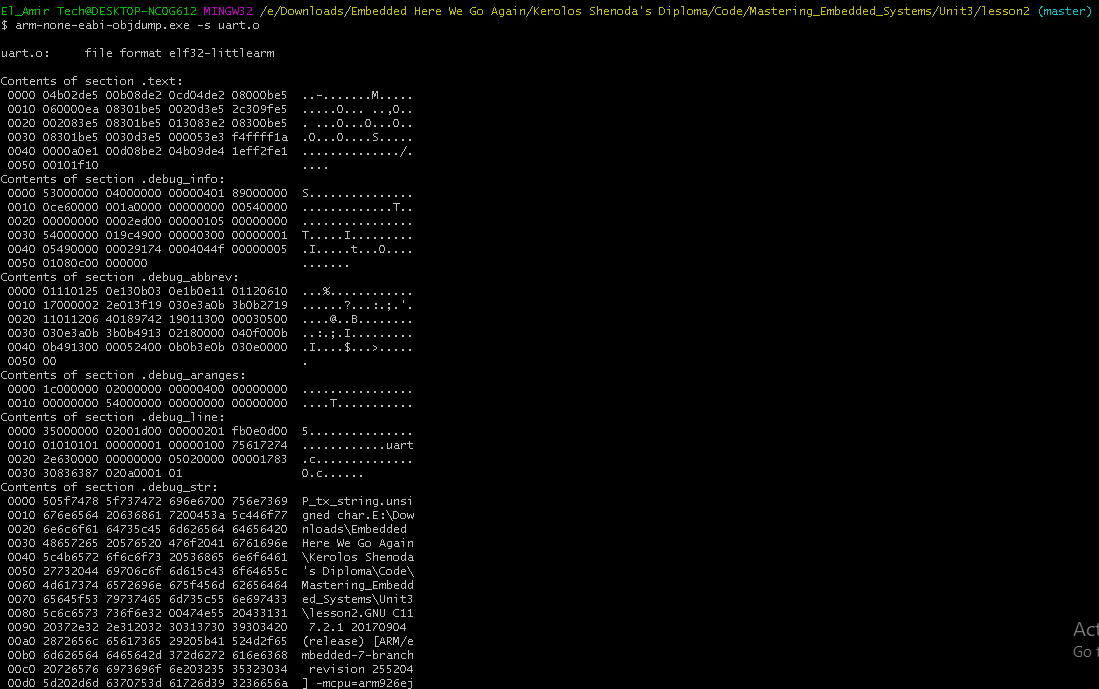


* $ arm-none-eabi-objdump.exe -D startup.o

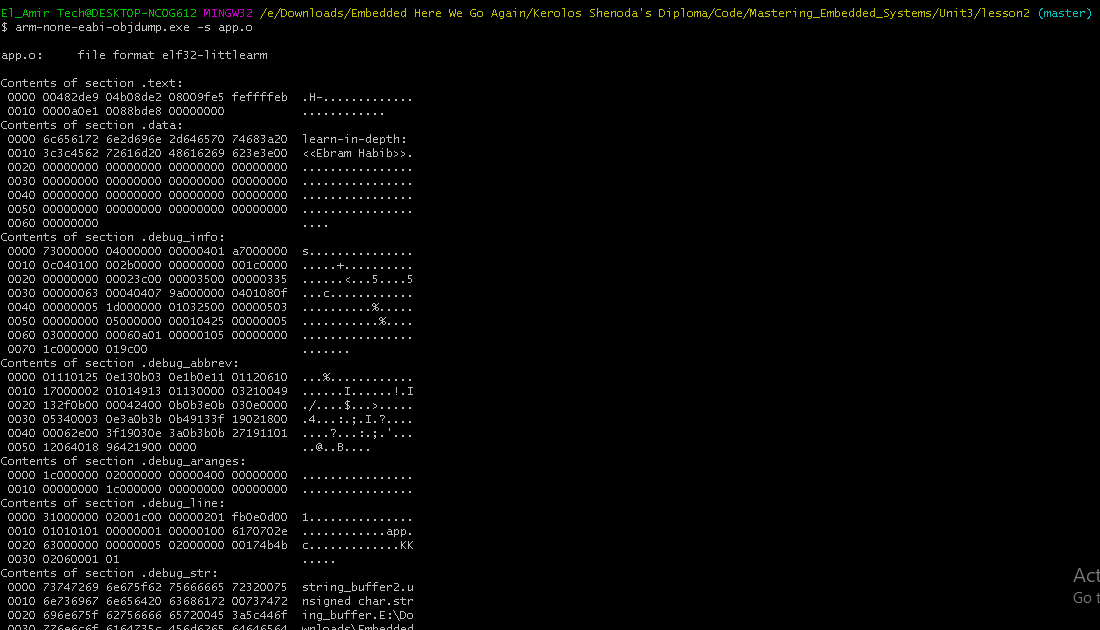


## To display the full content of all sections requested:

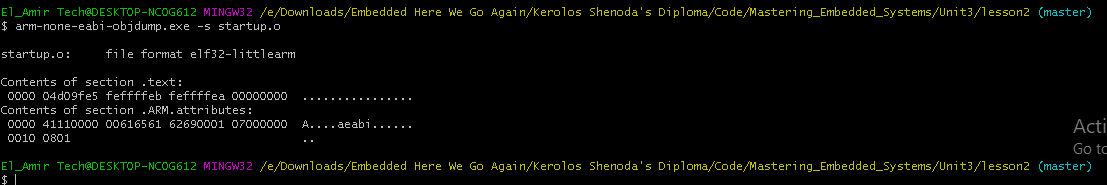
* $ arm-none-eabi-objdump.exe -s uart.o



* $ arm-none-eabi-objdump.exe -s app.o



* $ arm-none-eabi-objdump.exe -s startup.o

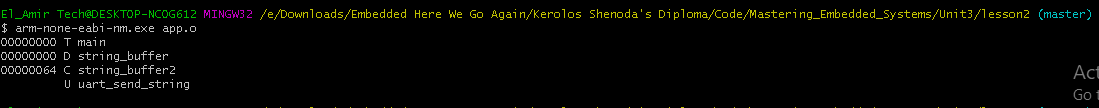


## To read the symbols and check the Entry Point Address:

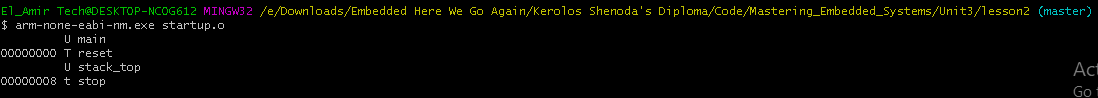
* $ arm-none-eabi-nm.exe uart.o



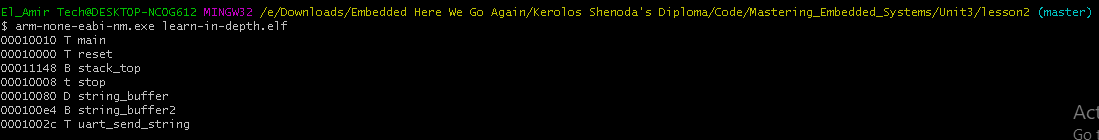
* $ arm-none-eabi-nm.exe app.o



* $ arm-none-eabi-nm.exe startup.o



* $ arm-none-eabi-nm.exe learn-in-depth.elf



* $ arm-none-eabi-readelf.exe -a learn-in-depth.elf