```
SOC- System On Chip
SIP- System In Package
SOM- System On Module
SBC- Single Board Computer
HEX

Toolchain
1. Manual Build
2. Get prebuilt one
    - Linaro (Maintainers)
    - Hardware Vendor
3. Build or Download using BuildSystem


RootFS
SquashFS ReadOnly
/
/bin
/lib
/usr
/etc
/dev
/data
    - /mtd0/mtdblk5  ( Writable Partion mounted on /data)

--------------------
Device Driver


rtc_set(), rtc_get()
I2C RTC Client Driver:      uboot-rba5d2x/drivers/rtc/ds1307.c

i2c_init()  i2c_reg_read() i2c_reg_write()
HAL I2C Sub System:             uboot-rba5d2x/drivers/gpio/at91_gpio.c
Atmel I2C Controler Driver :    uboot-rba5d2x/drivers/i2c/at91_i2c.c

I2C device connected to which BUS channel
I2C device Address

---------------------

printf() on 8051
    1. Parse the argument, process the string
    2. Print on the Console


printf:   (8051, msp430, pic, rl, lpc2148, )
Makefile
print.c
console.c ( uart_init(), uart_putc(),uart_getc() )
console-8051.c
console-msp430.c
console-pic.c
console-rl.c
console-lpc.c

CONFIG_8051
    console-8051.c

CONFI_MSP430
    console-msp430.c


Compiling Bootloader
cd <bootloader_src>
make distclean
make <board_name>_defconfig  =>  make rugged_board_a5d2x_qspiflash_defconfig
#.config file will be created

// If you want to do additional configuration
make menuconfig
```

```
#Modification will be saved in .config

make

ll /boot -h

-----------------
User space Management:
init -SystemV   - busybox
systemd   (latest)


----------------------------
Linux Images:
Vmlinux -> Kernel Image with debug symbols
Image -> Striped Kernel Image
zImage -> Zipped Kernel Image
uImage -> zImage + UBoot Headers

---------------------------------
Before Linux application execution , embedded Linux OS typically takes more time for booting
and Rootfs loading, is there any options or possibilities to reduce this time?
    - Bootchart


Todo List:
    Can we build the Rugged board SDK in WSL?
    How can we create nodes in uboot ?
    Does Rugged board has on-board JTAG debugger?

What Next ???
Embedded Linux Engineering:
https://www.community.ruggedboard.com/how-to-build-embedded-linux-system
```