**Understanding the reset sequence of microcontroller**

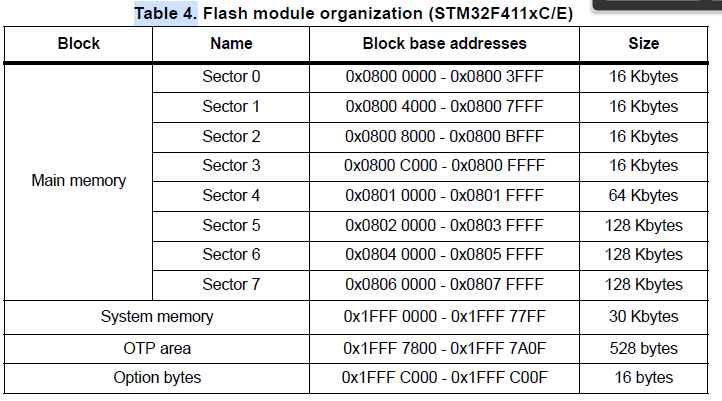
When microcontroller resets, the PC of the microcontroller is loaded with the values as 0x00000000. Then in the next sequence the processor reads the value at the memory location of 0x00000000 from the flash and put it to the MSP (Main Stack Pointer). We can say that the processor first initializes the main stack pointer. After this the processor reads memory location as 0x00000004 transfer the content into the PC. This contains the value at which address of the reset handler lies. The PC now jumps into the reset handler. A reset handler is just the exact C codes which was dumped into the flash with required initializations. After the completion of reset handler the main() will be executed which is the main application.

Figure Memory organization

The user flash starts at 0x8000 0000(sect 0) for stm32f411ve by default. This can be verified by using some debugging like SWD/JTAG.

The answer is MEMORY ALIASING.


MEMORY ALIASING

Figure :Memory Aliasing

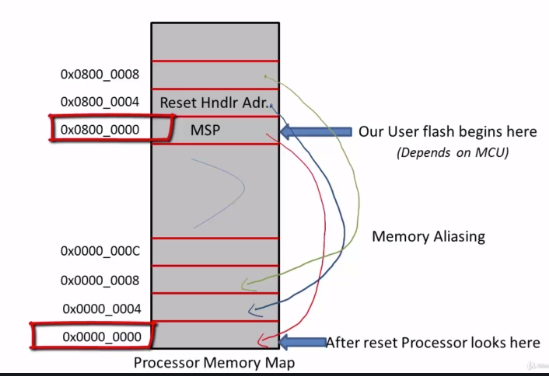


Figure : Processor Memory Map

As from the above image its clear that how 0x0000 0000 is mapped with 0x0800 0000 address so when the CPU generate 0x0000 0000 then its automatically mapped to the flash memory address.

**Understanding the Boot configuration of microcontroller**

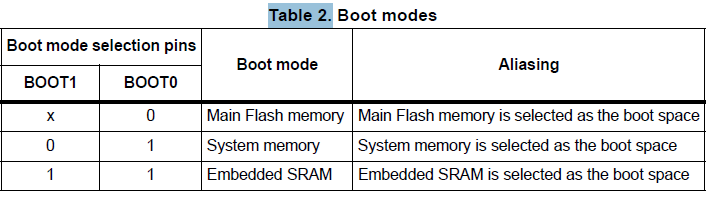


Figure : Boot mode configuration

By default the main Flash memory is selected as boot area. So if BOOT[1:0] is 01 the the boot area is selected as system memory starts at 0x1FFF 0000.

**Jumping to user application and bootloader based on the switch button status**



**Understanding the use of Vector table offset register**

We know that the by default the memory address of 0x0000 0000 is alias as 0x0800 0000 memory location so here the mcirocontoller starts its execution. But we can relocate this address so that it change its execution to user defined vector table where user application resides by modifying in the VTOR(Vector Table Offset Register) register. So with this we can jump into any location in the vector table. Here in our case we have selected sector 0 and sector 1 as flash memory for bootloader and from sector 2 user application will be residing. This can be modifed as startup file of project find reset handler. From reset handler look for SystemInit() you need to modify the VECT\_TAB\_OFFSET macro defination to the desied sector. In our case we have modified to sector 2 of of the flash memory.



**Placing User Application on sector 2**

Linker has to modified for placing the application. So, this can be done by modifying the Linker script. In Keil it’s easy as it has GUI. Following things must be done on option for target inside Keil

1. Set target IROM1 Inside Keil to sector address as 0x0800 8000
2. Linker:
   1. Uncheck “Use Memory Layout from target Dialog”
   2. R/O Base: 0x08008000
   3. Check “Use Memory Layout from target Dialog”
3. Now on the target tab click of default settings (NOTE: Keil linker script needs)
4. Rebuild the code

**Signature Structure**

This structure is stored at the end of flash memory of the flash so that it does not get overwrite by application codes. When ARM microcontroller is booting, it read this structure and load the application accordingly. This structure will be modified only when new firmware is available for flash.

