Aim of this project was to test the boot-loader design.

Section	Addess	Size
BootloaderCode	0x0800_0000	24KB
BootloaderCode	0x0800_5FFF	24NB
matadata ima 0	0x0800_6000	4KB
metadata_img_0	0x0800_6FFF	4ND
lma0 codo	0x0800_7000	241/0
Img0_code	0x0800_CFFF	24KB
matadata ima 1	0x0800_D000	4KD
metadata_img_1	0x0800_DFFF	4KB
lma1 codo	0x0800_E000	2440
Img1_code	0x0801_3FFF	24KB
_		

Figure 1: Organisation of Flash

Img0_code and Img1_code represents application code which could be selected by Bootloader for execution.

metadata_img_0 and metadata_img_1 are the meta-data about corresponding images. This would be used by bootloader.

Step1: Img_0 Setup

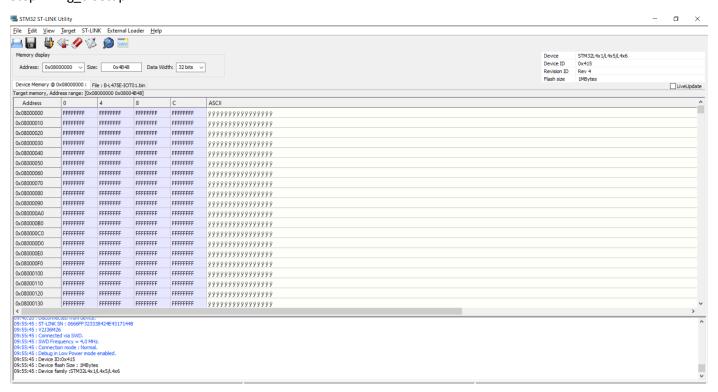


Figure 2:Erase the whole Flash Memory using STM-tool

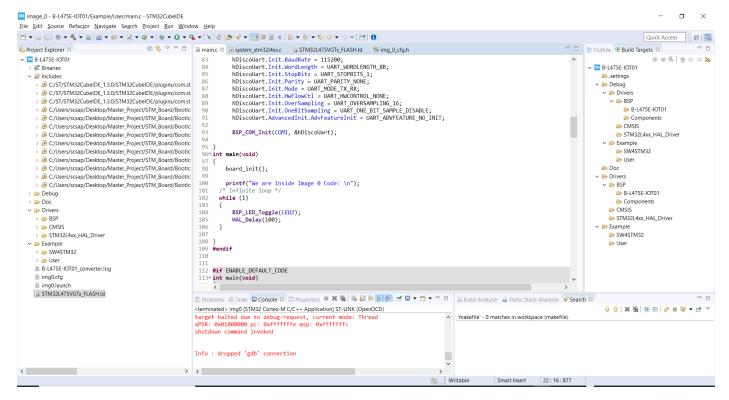


Figure 3: Image_0 Application code.

Figure 4:Img_0 vector Table offset

```
☐ main.c ☐ system_stm32l4xx.c ☐ STM32L475VGTx_FLASH.ld ☒ 🛅 img_0_cfg.h
    35/* Highest address of the user mode stack */
   36_estack = 0x20018000; /* end of RAM */
37/* Generate a link error if heap and stack don't fit into RAM */
38_Min_Heap_Size = 0x200; /* required amount of heap */
39_Min_Stack_Size = 0x400; /* required amount of stack */
   41/* Specify the memory areas */
   42 MEMORY
   43 {
   44 RAM (xrw) : ORIGIN = 0x20000000, LENGTH = 96K

45 RAM2 (xrw) : ORIGIN = 0x10000000, LENGTH = 32K

46 FLASH (rx) : ORIGIN = 0x8007000, LENGTH = 1024K
   48
   49 /* Define output sections */
   50 SECTIONS
   51{
52 /* The startup code goes first into FLASH */
   53 .i
54 {
           .isr_vector :
   54 {
55 . = ALIGN(8);
56 KEEP(*(.isr_vector)) /* Startup code */
57 . = ALIGN(8);
58 } >FLASH
        /* The program code and other data goes into FLASH */
   61 .text:
```

Figure 5: Change in Img_O Load Script.

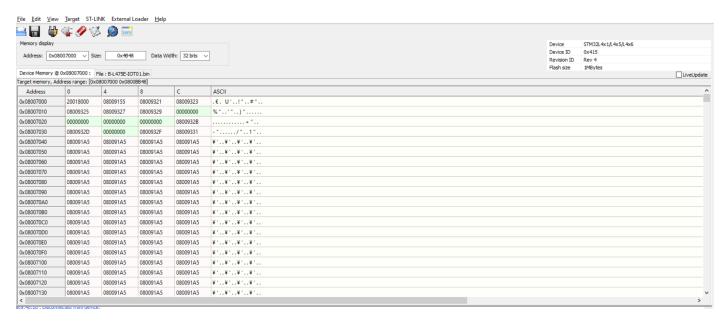


Figure 6:Verify Img_0 code is programmed into Flash.

Step 2: Img_1 Setup

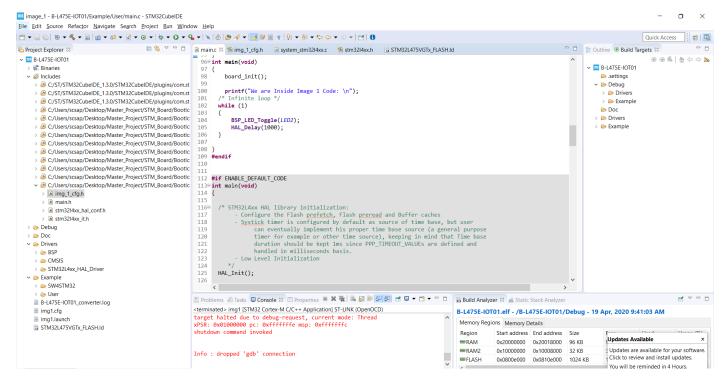


Figure 7: Image_1 Application Code.

```
111
    112
    113
    1149/**
                         * @}
    115
    116
    117
     118⊖/** @addtogroup STM32L4xx_System_Private_Defines
    119
    120
    121
     123⊕/*!< Uncomment the following line if you need to relocate your vector Table in
                                        Internal SRAM.
    124
     125 /* #define VECT_TAB_SRAM */
    1269 /*
    127 #define VECT_TAB_OFFSET 0x00 //< Vector Table base offset field.
                                                                                                                                                        This value must be a multiple of 0x200. */
128 THIS VOICE MOST TO THE PROPERTY TO THE PROPERTY OF THE PRO
     128
    1319/**
                          * @}
    132
    133
```

Figure 8: Img_1 Vector Offset.

Figure 9:Img_1 Load script changes.

STM32 ST-LINK	Utility				-	- o >
ile <u>E</u> dit <u>V</u> iew	<u>T</u> arget ST-L	INK External	oader <u>H</u> elp			
<u> </u>	₩ 🏈 🦞	\$ 🙆 🔜				
Memory display				Device STM32L4x1/L4x5/L4x6		
Address: 0x0800E000 V Size: 0x4848 Data Width: 32 bits V				ith. 22 hite	Device TD OLATE	
				iui: 32 bits <	Revision ID Rev 4	
Device Memory @	0v0800E000 :	Ela : 0.1 4755-10	TO1 bin		Flash size 1MBytes	LiveUpdat
Target memory, Ad						LiveUpdat
Address	0	4	8	С	ASCII	
0x0800E000	20018000	08010155	08010321	08010323	.6. U!#	
0x0800E010	08010325	08010327	08010329	00000000	%')	
0x0800E020	00000000	00000000	00000000	0801032B	+	
0x0800E030	0801032D	00000000	0801032F	08010331	/1	
0x0800E040	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E050	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E060	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E070	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E080	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E090	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E0A0	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E0B0	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E0C0	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E0D0	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E0E0	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E0F0	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E100	080101A5	080101A5	080101A5	080101A5	¥¥¥	
0x0800E110	080101A5	080101A5	080101A5	080101A5	VVV	
0x0800E120	080101A5	080101A5	080101A5	080101A5	VVV	
0x0800E130	080101A5	080101A5	080101A5	080101A5	¥¥¥	

Figure 10: Verify the programmed Img_1 in Flash

Step 3: Bootloader Setup

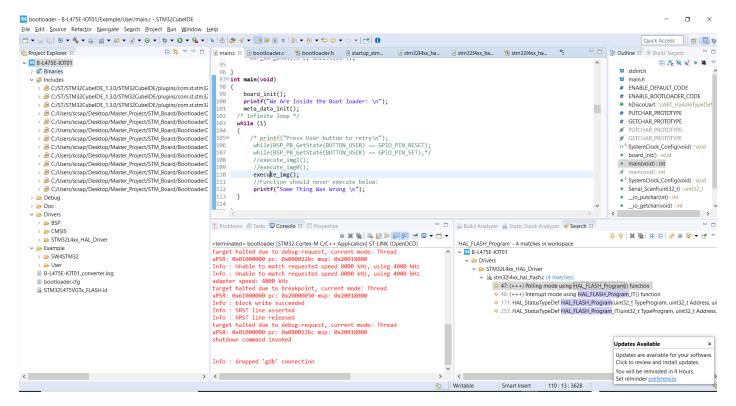


Figure 11: Bootloader Code

```
- -
STM32L475VGT... 🛭 🚨 main.c
                                                                                        🛅 stm32l4xx_ha...
                           le bootloader.c
                                         🛅 bootloader.h
                                                         startup_stm...
                                                                        stm32I4xx_ha...
  35 /* Highest address of the user mode stack */
  36_estack = 0x20018000; /* end of RAM */
  37/* Generate a link error if heap and stack don't fit into RAM */
  38_Min_Heap_Size = 0x200; /* required amount of heap */
  39 Min_Stack_Size = 0x400; /* required amount of stack */
  40
  41/* Specify the memory areas */
  42 MEMORY
  43 {
  44 RAM (xrw)
                  : ORIGIN = 0x20000000, LENGTH = 96K
  45 RAM2 (xrw)
                   : ORIGIN = 0x10000000, LENGTH = 32K
                 : ORIGIN = 0x8000000, LENGTH = 1024K
  46 FLASH (rx)
  47 }
  49 /* Define output sections */
  50 SECTIONS
  51 {
     /* The startup code goes first into FLASH */
STM32L475VGTx FLASH.Id
🗟 Build Analyzer 🚊 Static Stack Analyzer 🔗 Search 🛭
```

Figure 12: Bootloader Loader Scrip Changes.

```
i system_stm3... 

□ system_stm3... 
□ 34
STM32L475VGT...
                @ main.c
                         h bootloader.h
                                     c stm32l4xx_ha...
                                                     stm32I4xx_ha...
                                                                    stm32l4xx_hal.c
 117
 118⊖/** @addtogroup STM32L4xx_System_Private_Defines
119
120
 121
 123⊕/*!< Uncomment the following line if you need to relocate your vector Table in
         Internal SRAM. */
 125 /* #define VECT_TAB_SRAM */
1260 #define VECT_TAB_OFFSET 0x00 /*!< Vector Table base offset field.
                                    This value must be a multiple of 0x200. */
 127
 128 /**************
1299 /**
 130
 131
133⊖/** @addtogroup STM32L4xx_System_Private_Macros
134
 135
136
 1379 /**
```

Figure 13: Bootloader code Vect_Table Offset.

```
1 stm32l4xx_ha...
                                                               stm32l4xx_hal.c
                                                                                system_stm3...
 46
 47
 480 void meta_data_init(){
     //Creating a circular LinkedList
  49
         uint64_t data_strt_next=0;
  51
         uint32_t* addr=(uint32_t*)IMG_0_META_DATA_START_ADDR;
  52
  53
         HAL_FLASH_Unlock();
  54
         //initialise metadata for img_0:
  55
         data_strt_next=IMG_1_META_DATA_START_ADDR;
  56
         data_strt_next=data_strt_next<<32 | IMG_0_START_ADDR;</pre>
  57
         HAL_FLASH_Program(FLASH_TYPEPROGRAM_DOUBLEWORD, addr, data_strt_next);
  58
  59
         addr=addr+2;
  60
         data_strt_next=0;
         data_strt_next= FALSE;
  61
         data_strt_next=data_strt_next << 32 | FALSE;</pre>
  62
  63
         HAL_FLASH_Program(FLASH_TYPEPROGRAM_DOUBLEWORD, addr, data_strt_next);
  64
  65
         //initialise metadata for img_1:
  66
         addr=(uint32_t*)IMG_1_META_DATA_START_ADDR;
         data_strt_next=IMG_0_META_DATA_START_ADDR;
  67
  68
         data_strt_next=data_strt_next<<32 | IMG_1_START_ADDR;</pre>
  69
  70
         HAL_FLASH_Program(FLASH TYPEPROGRAM DOUBLEWORD, addr, data strt next);
  71
         addr=addr+2;
  72
         data_strt_next=0;
  73
         data_strt_next= TRUE;
         data_strt_next=data_strt_next << 32 | TRUE;</pre>
  74
  75
         HAL_FLASH_Program(FLASH_TYPEPROGRAM_DOUBLEWORD, addr, data_strt_next);
  76
  77
         HAL_FLASH_Lock();
  78
  79 }
  20
    //returns the address of image to Everute
```

Figure 14: Bootloader Function to initialise meta-data for both images

File Edit View	Jarget ST-LII	NK External L	oader <u>H</u> elp					
🔙 🔚 🏺 (IF 🏈 🖞	, 👰 🔜						
Memory display				Device	STM32L4x1/L4x5/L4x6			
Address: 0x0800	Address: 0x08000000 V Size: 0x4948 Data Width: 32 bits V			Device ID	0x415			
						Revision ID Flash size	Rev 4 1MBytes	
Device Memory @ 0x	08000000 : F	le : B-L475E-10T	01.bin			riash size	D*IDYtes	LiveUpdate
Target memory, Addre	ess range: [0x0	8000000 0x0800	4848]					
Address	0	4	8	C	ASCII			^
0x08000000	20018000	080022BD	080025A1	080025A3	.€. ¼";%£%			
0x08000010	080025A5	080025A7	080025A9	00000000	¥%§%©%			
0x08000020	00000000	00000000	00000000	080025AB				
0x08000030	080025AD	00000000	080025AF	080025B1	- % ~ % ± %			
0x08000040	0800230D	0800230D	0800230D	0800230D	.***			
0x08000050	0800230D	0800230D	0800230D	0800230D	.***			
0x08000060	0800230D	0800230D	0800230D	0800230D	.***			
0x08000070	0800230D	0800230D	0800230D	0800230D	.***			
0x08000080	0800230D	0800230D	0800230D	0800230D	.***			
0x08000090	0800230D	0800230D	0800230D	0800230D	.***			
0x080000A0	0800230D	0800230D	0800230D	0800230D	.***			
0x080000B0	0800230D	0800230D	0800230D	0800230D	.***			
0x080000C0	0800230D	0800230D	0800230D	0800230D	.***			
0x080000D0	0800230D	0800230D	0800230D	0800230D	.***			
0x080000E0	0800230D	0800230D	0800230D	0800230D	.***			
0x080000F0	0800230D	0800230D	0800230D	0800230D	.***			
0x08000100	0800230D	0800230D	0800230D	0800230D	.***			
0x08000110	0800230D	0800230D	0800230D	0800230D	.***			
0x08000120	0800230D	0800230D	0800230D	0800230D	.***			
0x08000130	0800230D	0800230D	0800230D	0800230D				~
<								>

Figure 15: Verify Bootloader Code in Flash

Address: 0x08006000 V Size: 0x4B48 Data Width: 32 bits V								
Device Memory @ 0x08006000 : File : B-L475E-IOT0 1.bin Target memory, Address range: [0x08006000 0x0800AB48]								
Address	0	4	8	С	ASCII			
0x08006000	08007000	0800D000	00000000	00000000	.pĐ			
0x08006010	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	yyyyyyyyyyy y			

Figure 16: Verify Metadata for Img-0 (* Img_0 is set to be invalid for Demo Purpose)

Address: 0x0800D000 V Size: 0x4848 Data Width: 32 bits V								
Device Memory @ 0x0800D000: File: B-L475E-IOT01.bin								
arget memory, Ad	Farget memory, Address range: [0x0800D000 0x08011B48]							
Address	0	4	8	С	ASCII			
0x0800D000	0800E000	08006000	00000001	00000001	.à`			
0x0800D010	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ			

Figure 17: Verify Meta-data for Img-1 (* Img_1 is set to be valid and updated for Demo Purpose)

```
//Every image will have a meta-data denoting below information.

typedef struct meta_data{

    //this is the address of the Image in FLASH
    uint32_t img_strt_add;

    //this is pointer to metadata of next image
    struct meta_data *next;

    //This field indicates if the image pointed by this metadata is valid or not
    uint32_t isValid;

    //This field indicates if the image is updated or not
    uint32_t isUpdated;

}meta_data_t;
```

Figure 18: Meta-Data Structure in Bootloader Code

Step 4: Verify the Working of Bootloader

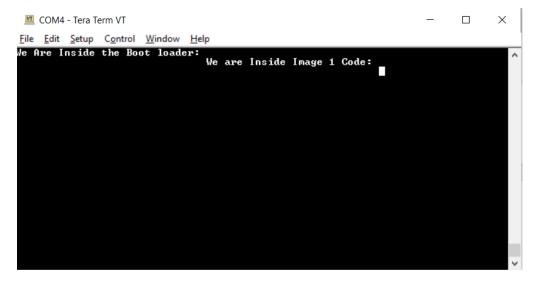


Figure 19: Boot loader Executed Img_1 , since Img_0 was set to be invalid.