**STM32Flash.id :**

\*\*\*\*\*\*

\*\*

\*\* @file : LinkerScript.ld

\*\*

\*\* @author : STM32CubeIDE

\*\*

\*\* Abstract : Linker script for STM32F407G-DISC1 Board embedding STM32F407VGTx Device from stm32f4 series

\*\* 1024Kbytes FLASH

\*\* 64Kbytes CCMRAM

\*\* 128Kbytes RAM

\*\*

\*\* Set heap size, stack size and stack location according

\*\* to application requirements.

\*\*

\*\* Set memory bank area and size if external memory is used

\*\*

\*\* Target : STMicroelectronics STM32

\*\*

\*\* Distribution: The file is distributed as is, without any warranty

\*\* of any kind.

\*\*

\*\*\*\*\*\*

\*\* @attention

\*\*

\*\* Copyright (c) 2022 STMicroelectronics.

\*\* All rights reserved.

\*\*

\*\* This software is licensed under terms that can be found in the LICENSE file

\*\* in the root directory of this software component.

\*\* If no LICENSE file comes with this software, it is provided AS-IS.

\*\*

\*\*\*\*\*\*

\*/

/\* Entry Point \*/

ENTRY(Reset\_Handler)

/\* Highest address of the user mode stack \*/

\_estack = ORIGIN(RAM) + LENGTH(RAM); /\* end of "RAM" Ram type memory \*/

\_Min\_Heap\_Size = 0x200; /\* required amount of heap \*/

\_Min\_Stack\_Size = 0x400; /\* required amount of stack \*/

/\* Memories definition \*/

MEMORY

{

CCMRAM (xrw) : ORIGIN = 0x10000000, LENGTH = 64K

RAM (xrw) : ORIGIN = 0x20000000, LENGTH = 128K

FLASH (rx) : ORIGIN = 0x8000000, LENGTH = 1024K

}

/\* Sections \*/

SECTIONS

{

/\* The startup code into "FLASH" Rom type memory \*/

.isr\_vector :

{

. = ALIGN(4);

KEEP((.isr\_vector)) / Startup code \*/

. = ALIGN(4);

} >FLASH

/\* The program code and other data into "FLASH" Rom type memory \*/

.text :

{

. = ALIGN(4);

(.text) / .text sections (code) \*/

(.text) /\* .text\* sections (code) \*/

(.glue\_7) / glue arm to thumb code \*/

(.glue\_7t) / glue thumb to arm code \*/

\*(.eh\_frame)

KEEP (\*(.init))

KEEP (\*(.fini))

. = ALIGN(4);

\_etext = .; /\* define a global symbols at end of code \*/

} >FLASH

/\* Constant data into "FLASH" Rom type memory \*/

.rodata :

{

. = ALIGN(4);

(.rodata) / .rodata sections (constants, strings, etc.) \*/

(.rodata) /\* .rodata\* sections (constants, strings, etc.) \*/

. = ALIGN(4);

} >FLASH

.ARM.extab : {

. = ALIGN(4);

(.ARM.extab .gnu.linkonce.armextab.\*)

. = ALIGN(4);

} >FLASH

.ARM : {

. = ALIGN(4);

\_\_exidx\_start = .;

(.ARM.exidx)

\_\_exidx\_end = .;

. = ALIGN(4);

} >FLASH

.preinit\_array :

{

. = ALIGN(4);

PROVIDE\_HIDDEN (\_\_preinit\_array\_start = .);

KEEP ((.preinit\_array))

PROVIDE\_HIDDEN (\_\_preinit\_array\_end = .);

. = ALIGN(4);

} >FLASH

.init\_array :

{

. = ALIGN(4);

PROVIDE\_HIDDEN (\_\_init\_array\_start = .);

KEEP ((SORT(.init\_array.)))

KEEP ((.init\_array))

PROVIDE\_HIDDEN (\_\_init\_array\_end = .);

. = ALIGN(4);

} >FLASH

.fini\_array :

{

. = ALIGN(4);

PROVIDE\_HIDDEN (\_\_fini\_array\_start = .);

KEEP ((SORT(.fini\_array.)))

KEEP ((.fini\_array))

PROVIDE\_HIDDEN (\_\_fini\_array\_end = .);

. = ALIGN(4);

} >FLASH

/\* Used by the startup to initialize data \*/

\_sidata = LOADADDR(.data);

/\* Initialized data sections into "RAM" Ram type memory \*/

.data :

{

. = ALIGN(4);

\_sdata = .; /\* create a global symbol at data start \*/

(.data) / .data sections \*/

(.data) /\* .data\* sections \*/

(.RamFunc) / .RamFunc sections \*/

(.RamFunc) /\* .RamFunc\* sections \*/

. = ALIGN(4);

\_edata = .; /\* define a global symbol at data end \*/

} >RAM AT> FLASH

\_siccmram = LOADADDR(.ccmram);