

## AN0026

应用笔记

### Extending SRAM in User's Program

## 前言

这篇应用指南描述了怎么在AT32芯片上执行程序来扩展SRAM。

支持型号列表:

	AT32F403xx				
	AT32F403Axx				
古体刑导	AT32F407xx				
又苻至亏	AT32F413xx AT32F435xx				

# **Y5=**751;

# 目录

ית ש	र को दीन		
<b>び</b> 直	[例程		
2.1	例程分	↑析	
	2.1.1	函数说明	
	2.1.2	例程展示	



# 表目录

表 1. AT32F403A EOPB0	设定值7
表 2. 文档版本历史	



# 图目录

图 1. extend_sram 函数	7
图 2. KEIL 工程启动文件修改	8
图 3. IAR 工程启动文件修改	8
图 4. KEIL 程序中 SRAM 大小选择	9
图 5.KEIL 工程 C/C++/Preprocessor Symbols 配置	9
图 6. IAR 程序中 SRAM 大小选择	10
图 7. IAR 工程 C/C++ Compiler-Preprocessor 配置	10



### 1 概述

AT32 MCU某些型号片上SRAM,有提供一个特别的SRAM扩展模式,可让用户通过设定用户系统数据区的EOPB0来调整SRAM的大小。

一般此扩展模式的设置都建议使用雅特力的ICP或ISP工具,在产品批量生成时跟程序的烧录一起完成,但在使用者无法使用ICP/ISP工具的情境下,也可以通过执行程序来完成。 本篇指南将说明如何在程序中正确的设置EOPB0来完成SRAM的扩展。



### 2 设置例程

本文列举的支持扩展SRAM的型号MCU都是通过修改EOPB0的值来完成SRAM的扩展,但各型号能够调整的SRAM容量大小并不相同,所以都有提供一个例程供使用者参考,具体可调整SRAM大小以RM或者DS描述为准。

本文档以AT32F403A为例,基于AT-START-F403A开发板,例程放在固件库的如下路径: \project\at\_start\_f403a\examples\sram\extend\_sram目录。



### 2.1 例程分析

#### 2.1.1 函数说明

AT32F403A出厂预设的SRAM大小为96K字节,修改EOPB0后可扩展至224K字节,EOPB0的设定值如下:

表	1. AT32F403A EOPB0	设定值
---	--------------------	-----

EOPB0: 扩充	的系统选项
位7:0	224KB_MODE
	<b>0xFE</b> :片上内存为 <b>224K</b> 字节
	<b>0xFF</b> :片上内存为 <b>96K</b> 字节
	其他设置保留

extend\_sram()函数操作修改EOPB0,通过EXTEND\_SRAM宏定义可将SRAM从默认的96K字节扩展 到224K字节,或从224K字节改回96K字节。其中EXTEND\_SRAM的定义值在工程项目配置选项中完成。须注意函数内,不可使用全局变量。

修改EOPB0之后,必须执行系统复位,新的EOPB0数值才会生效并真正的设定到所选的SRAM大小,函数如下图

图 1. extend\_sram 函数



通过修改 startup\_at32f403a\_407.s 的启动汇编代码,使extend\_sram函数在程序初始化之前执行, 范例中的 startup\_at32f403a\_407\_ext\_ram.s就是修改后的启动文件。下面的图分别是在KEIL和IAR 工程中如何修改启动代码。



: Reset handler Reset_Handler PROC EXPORT IMPORT IMPORT	Reset_Handler main SystemInit	[WEAK]
, and for extend sram IMPORT MOV32 MOV LDR BLX MOV32 LDR	extend_sram R0, #0x20001000 SP, R0 R0, =extend_sram R0, #0x08000000 SP, [R0]	
LDR BLX LDR BX ENDP	RO, =SystemInit RO RO, =main RO	-

#### 图 2. KEIL 工程启动文件修改

#### 图 3. IAR 工程启动文件修改

Reset_H	PUBWEAK SECTION Handler	Res .te	et_Handler xt:CODE:REORDE	R:NOROOT (2)
; add i	for exten	d sr	am	
	MOV32	R0,	#0x20001000	
	MOV	SP,	R0	
	LDR	R0,	=extend_sram	
	BLX	R0		
	MOV32	R0,	#0x08000000	
	LDR	SP,	[R0]	
			_	
	LDR	R0,	=SystemInit	
	BLX	R0		
	LDR	R0,	=iar_progra	m_start
	BX	R0		

需注意的两个重点:

- 必须在Reset\_Handler的一开头就去做EOPB0的修改,不要在SystemInit()函数里头设置,因为 用户一开始在Keil/IAR等开发环境设定的SRAM范围,就可能是以扩充后的224K字节作设定,且 实际用到的SRAM可能超过了默认的96K,此时堆栈(STACK)的指针初始值会被设定到96K之后 的地址,执行SystemInit()时就会出错,甚至发生HardFault而造成死机。
- 2) 在调用extend\_sram()函数前,要将堆栈(STACK)的指针先改到 96K字节内(例程中是修改到 (0x20001000),避免因STACK的指针初始值被设定到96K之后的地址,而造成extend\_SRAM函 数执行时发生错误。

# ,**17[**7]

#### 2.1.2 例程展示

#### 2.1.2.1 KEIL 工程

在例程内,可以通过"Select Target"窗口选择96K 字节或224K 字节的范例程序。如下图:

	图 4. KEIL 程序中 SRAM 大小选择
Image: Set stars	
	18 vold extend_sram(vold)

当选择set\_sram\_96k或者set\_sram\_224k时,EXTEND\_SRAM的宏定义值对应设置在 C/C++ → Preprocessor Symbols 的定义框里面,编译时extend\_sram()函数就会选择对应的配置。

图 5.KEIL 工程 C/C++/Preprocessor	<sup>·</sup> Symbols 配置
--------------------------------	-------------------------

Device   Target   Output   Listing	User C/C++ Asm Linker Debug	Utilities
Preprocessor Symbols <u>D</u> efine: AT32F403AVGT7,USE_S	TDPERIPH_DRIVER,AT_START_F403A_VI.	EXTEND_SRAM=0xFE
Language / Code Generation Execute-only Code Optimization: Level 0 (-00)	Strict ANSIC Enum Container always int	<u>W</u> amings: All Wamings  ▼
Optimize for Time     Split Load and Store Multiple     One <u>E</u> LF Section per Function	Plain Char is Signed     Read-Only Position Independent <u>R</u> ead-Write Position Independent	Thumb Mode No Auto Includes C99 Mode
Include Paths Misc Controls	ers\inc;\\\\\ibraries\cmsis\cm4\core_	support;\\\.Nibran
Compiler control string	)0 -apcs=interwork -split_sections -I .\.\.\. s\cm4\core_support -I .\.\.\.\.\.\libraries\cn	\.\Ibraries\drivers\inc -1 nsis\cm4\device_support -1
01	Cancel Defaults	: Help

程序运行完SRAM扩展并进入main()函数时,会检查EOPB0 的数值以确认是否有正确的去配置成所选择的SRAM大小,并且通过点亮LED4显示结果。

#### 2.1.2.2 IAR 工程

在例程内,可以通过"Select Target"窗口选择96K 字节或224K 字节的范例程序。如下图:

图 6. IAR 程序中 SRAM 大小选择				
Norkspace		•	џх	
extend_sram_224k			•	
extend_sram_224k				
_extend_sram_96k				JI
🗆 💭 extend_sram - e	🗆 🛡 extend_sram - exten 🗸			
–-⊞ 🛋 bsp	– ⊞ 🖬 bsp 🔹 🔹			
🖵 🛋 cmsis				
—⊞ ∎ startup_at32f   —⊞ ⊚ system_at32				

当选择set\_sram\_96k或者set\_sram\_224k时,EXTEND\_SRAM的宏定义值对应设置在 C/C++ Compiler→Preprocessor的定义框里面,编译时extend\_sram()函数就会选择对应的配置。



Category: General Options Static Analysis Runtime Checking	Multi-file Com	<b>bilation</b> Inused Public	:s			Factory	Settings
C/C++ Compiler	MISRA-C:1	MISRA-C:1998 Encodings Extra Options					ons
Assembler	Language 1 I	anguage 2	Code	Optimiza	ations	Output	List
Output Converter	Preproces	sor	Diagnostics		M	MISRA-C:2004	
Debugger Simulator CADI CMSIS DAP GDB Server I-jet/JTAGjet J-Link/J-Trace TI Stellaris Nu-Link PE micro ST-LINK Third-Party Driver TI MSP-FET TI XDS	\$PROJ_DIR\$\ \$PROJ_DIR\$\ \$PROJ_DIR\$\ \$PROJ_DIR\$\ Preinclude <u>Defined symt</u> AT32F403AVG USE_STDPERI AT_START_F4 EXTEND_SRAM	Additional include directories: (one per line)  \$PROJ_DIR\$\\\\\\libraries\drivers\inc \$PROJ_DIR\$\\\\\\libraries\cmsis\cm4\de \$PROJ_DIR\$\\\\\\libraries\cmsis\cm4\de \$PROJ_DIR\$\\\\\\libraries\cmsis\cm4\de \$PROJ_DIR\$\\\\\\libraries\cmsis\cm4\de \$PROJ_DIR\$\\\\\\\\\\\\\.					to : rective

程序运行完SRAM扩展并进入main()函数时,会检查EOPB0 的数值以确认是否有正确的去配置成所选择的SRAM大小,并且通过点亮LED4显示结果。



## 3 版本历史

表 2.	文档版本历史
------	--------

日期	版本	变更
2021.05.24	2.0.0	新版初版



#### 重要通知 - 请仔细阅读

买方自行负责对本文所述雅特力产品和服务的选择和使用,雅特力概不承担与选择或使用本文所述雅特力产品和服务相关的任何责任。

无论之前是否有过任何形式的表示,本文档不以任何方式对任何知识产权进行任何明示或默示的授权或许可。如果本文档任何部分涉及任何 第三方产品或服务,不应被视为雅特力授权使用此类第三方产品或服务,或许可其中的任何知识产权,或者被视为涉及以任何方式使用任何 此类第三方产品或服务或其中任何知识产权的保证。

除非在雅特力的销售条款中另有说明,否则,雅特力对雅特力产品的使用和/或销售不做任何明示或默示的保证,包括但不限于有关适销性、适合特定用途(及其依据任何司法管辖区的法律的对应情况),或侵犯任何专利、版权或其他知识产权的默示保证。

雅特力产品并非设计或专门用于下列用途的产品: (A) 对安全性有特别要求的应用,如:生命支持、主动植入设备或对产品功能安全有要求 的系统; (B) 航空应用; (C) 汽车应用或汽车环境; (D) 航天应用或航天环境,且/或(E) 武器。因雅特力产品不是为前述应用设计的,而采 购商擅自将其用于前述应用,即使采购商向雅特力发出了书面通知,风险由购买者单独承担,并且独力负责在此类相关使用中满足所有法律 和法规要求。

经销的雅特力产品如有不同于本文档中提出的声明和/或技术特点的规定,将立即导致雅特力针对本文所述雅特力产品或服务授予的任何保证 失效,并且不应以任何形式造成或扩大雅特力的任何责任。

© 2021 雅特力科技 (重庆) 有限公司 保留所有权利