晶心科技線上技術研討會

AndesCore™便捷的全C嵌入式编程

Driving Innovations™

晶心科技 市場及技術服務部
毛礼杰 軟件經理

大纲

- 系统初始化介绍
- 异常和中断说明
- 全C语法例子说明
- 总结

系统初始化(1)

- CPU相关特性
  - 1：中断向量表
  - 2：系统寄存器
  - 通常需要用assembly（汇编/组合）语言来操作
- AndesCore™全C嵌入式编程
  - C扩展语法
    - 用于Exception/Interrupt处理
  - Intrinsic function
    - 用于系统寄存器设置等
  - 适用于
    - Non-OS系统

系统初始化(2)

系统初始化

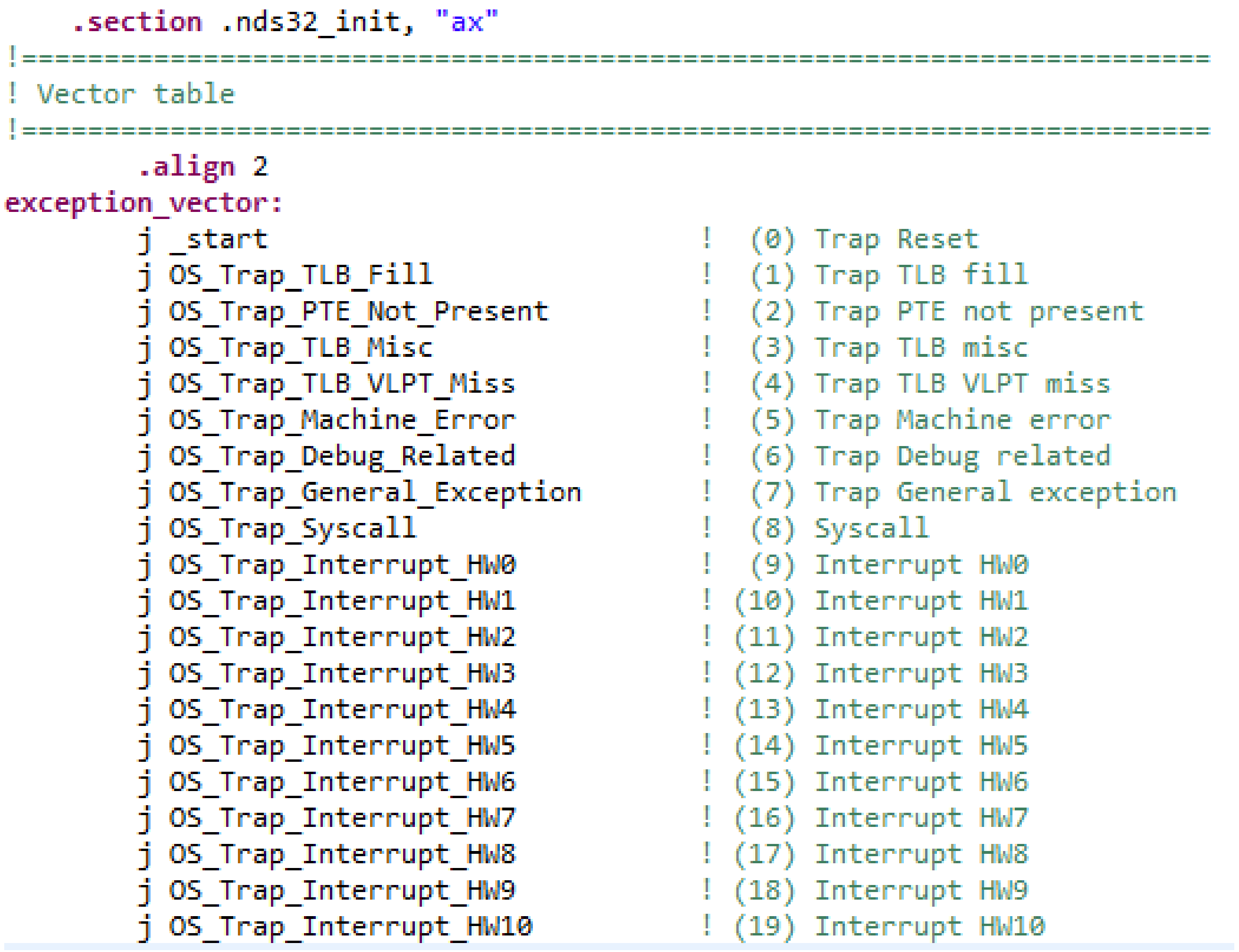
1：中断向量表

2：系统寄存器

3：memory的初始化

汇编初始化(1)

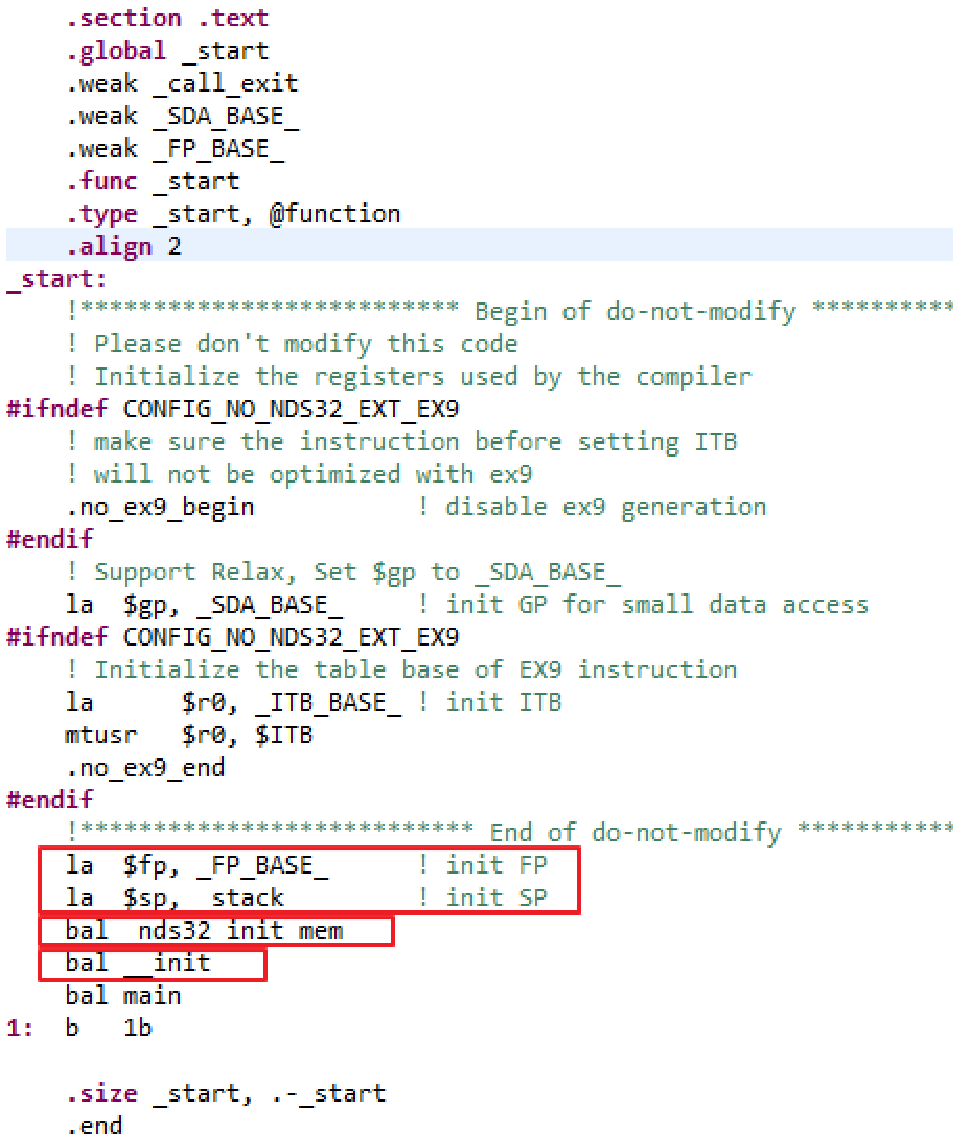
❖ 中断向量表



```
    .section .nds32_init, "ax"
!========================================================================
! Vector table
!========================================================================
        .align 2
exception_vector:
        j _start                        !  (0) Trap Reset
        j OS_Trap_TLB_Fill              !  (1) Trap TLB fill
        j OS_Trap_PTE_Not_Present       !  (2) Trap PTE not present
        j OS_Trap_TLB_Misc              !  (3) Trap TLB misc
        j OS_Trap_TLB_VLPT_Miss         !  (4) Trap TLB VLPT miss
        j OS_Trap_Machine_Error         !  (5) Trap Machine error
        j OS_Trap_Debug_Related         !  (6) Trap Debug related
        j OS_Trap_General_Exception     !  (7) Trap General exception
        j OS_Trap_Syscall               !  (8) Syscall
        j OS_Trap_Interrupt_HW0         !  (9) Interrupt HW0
        j OS_Trap_Interrupt_HW1         ! (10) Interrupt HW1
        j OS_Trap_Interrupt_HW2         ! (11) Interrupt HW2
        j OS_Trap_Interrupt_HW3         ! (12) Interrupt HW3
        j OS_Trap_Interrupt_HW4         ! (13) Interrupt HW4
        j OS_Trap_Interrupt_HW5         ! (14) Interrupt HW5
        j OS_Trap_Interrupt_HW6         ! (15) Interrupt HW6
        j OS_Trap_Interrupt_HW7         ! (16) Interrupt HW7
        j OS_Trap_Interrupt_HW8         ! (17) Interrupt HW8
        j OS_Trap_Interrupt_HW9         ! (18) Interrupt HW9
        j OS_Trap_Interrupt_HW10        ! (19) Interrupt HW10
```

汇编初始化(2)

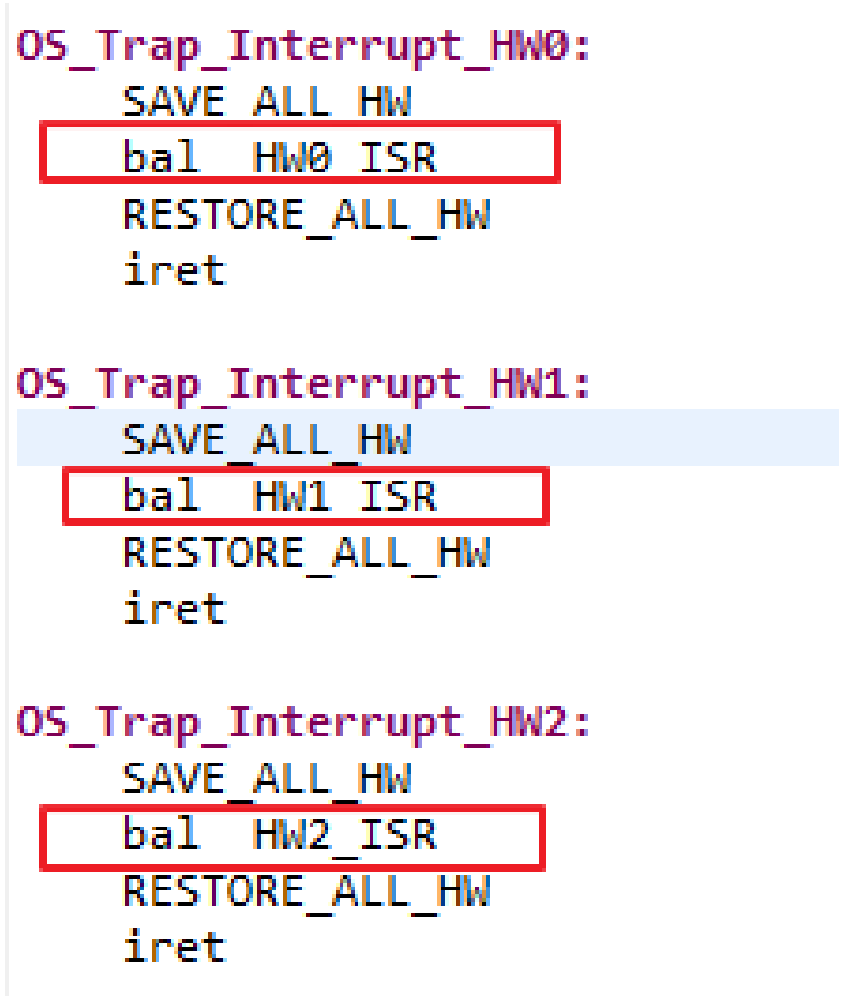
- ❖ 系统寄存器
- ❖ memory初始化
- ❖ 其它初始化



```
        .section .text
        .global _start
        .weak _call_exit
        .weak _SDA_BASE_
        .weak _FP_BASE_
        .func _start
        .type _start, @function
        .align 2
_start:
        !************************** Begin of do-not-modify **********
        ! Please don't modify this code
        ! Initialize the registers used by the compiler
#ifndef CONFIG_NO_NDS32_EXT_EX9
        ! make sure the instruction before setting ITB
        ! will not be optimized with ex9
        .no_ex9_begin               ! disable ex9 generation
#endif
        ! Support Relax, Set $gp to _SDA_BASE_
        la  $gp, _SDA_BASE_         ! init GP for small data access
#ifndef CONFIG_NO_NDS32_EXT_EX9
        ! Initialize the table base of EX9 instruction
        la      $r0, _ITB_BASE_ ! init ITB
        mtusr   $r0, $ITB
        .no_ex9_end
#endif
        !*************************** End of do-not-modify ***********
        la  $fp, _FP_BASE_          ! init FP
        la  $sp, _stack             ! init SP
        bal  _nds32_init_mem
        bal __init
        bal main
1:      b   1b

        .size _start, .-_start
        .end
```

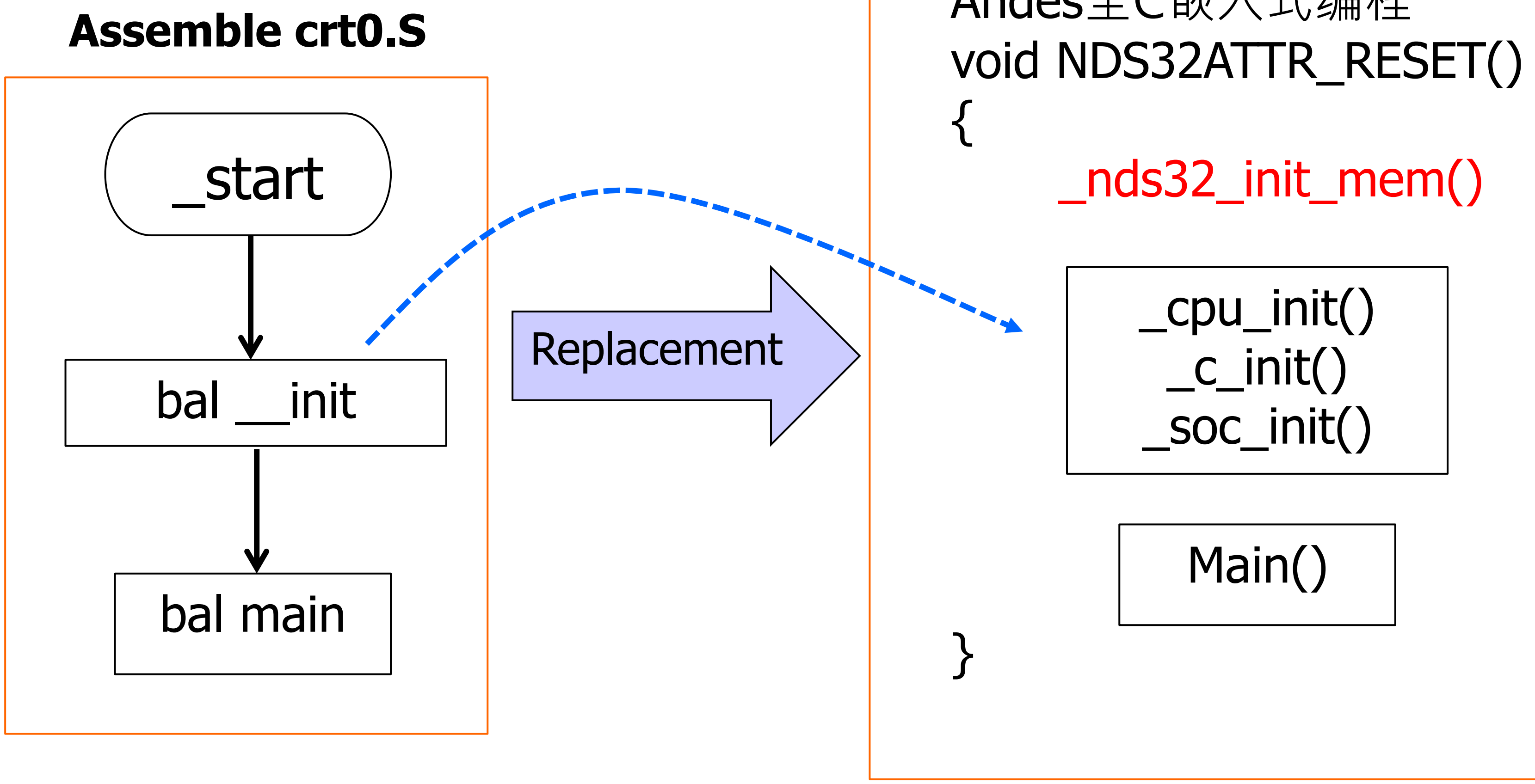
汇编初始化(3)

❖ ISR入口前期汇编部分



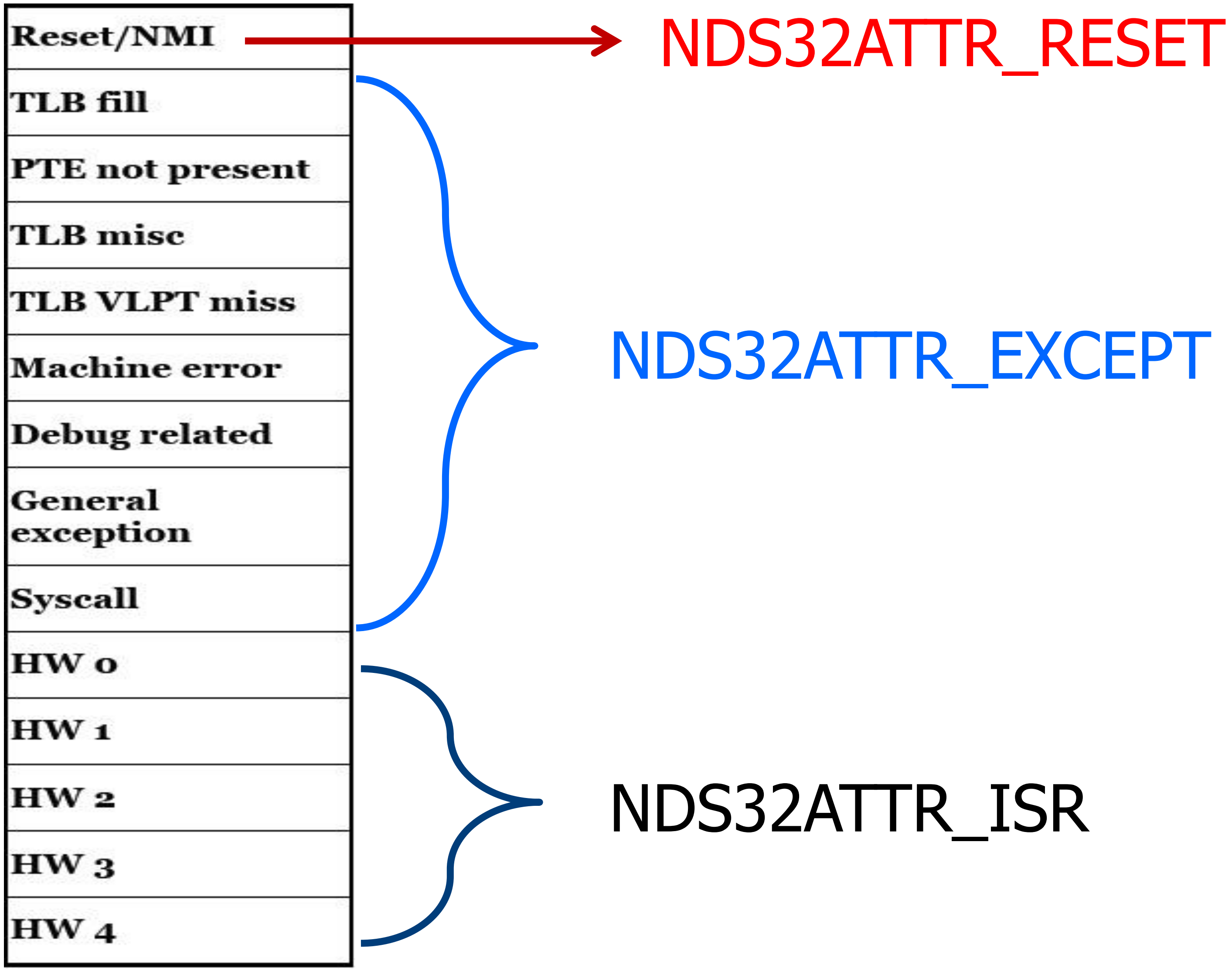
```
OS_Trap_Interrupt_HW0:
    SAVE_ALL_HW
    bal   HW0_ISR
    RESTORE_ALL_HW
    iret

OS_Trap_Interrupt_HW1:
    SAVE_ALL_HW
    bal   HW1_ISR
    RESTORE_ALL_HW
    iret

OS_Trap_Interrupt_HW2:
    SAVE_ALL_HW
    bal   HW2_ISR
    RESTORE_ALL_HW
    iret
```

汇编与全C嵌入式编程系统初始化对比



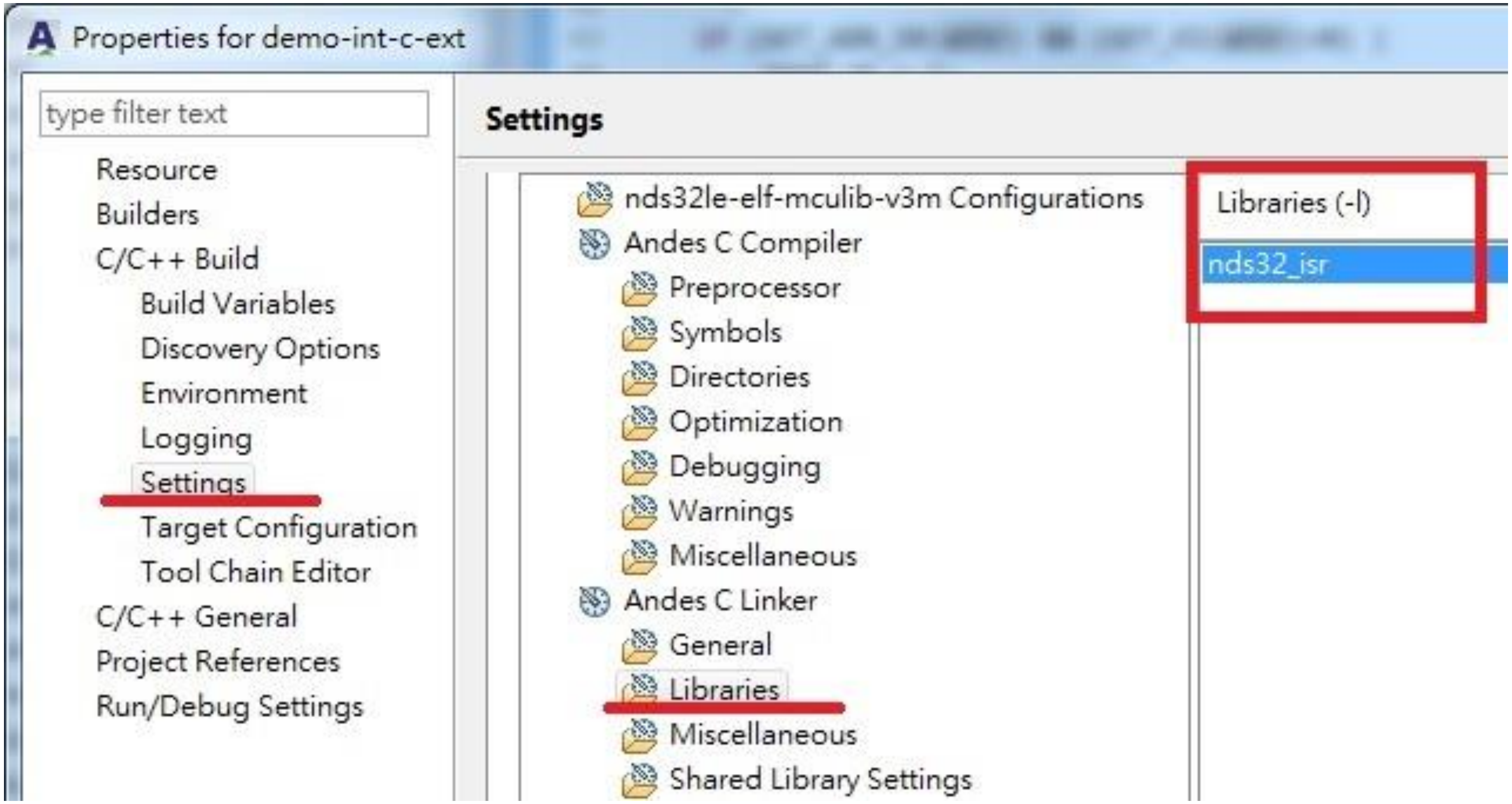
```
Andes全C嵌入式编程
void NDS32ATTR_RESET()
{
        _nds32_init_mem()

        _cpu_init()
        _c_init()
        _soc_init()

        Main()
}
```

向量表：异常和中断


Reset/NMI
TLB fill
PTE not present
TLB misc
TLB VLPT miss
Machine error
Debug related
General exception
Syscall
HW 0
HW 1
HW 2
HW 3
HW 4
NDS32ATTR_RESET
NDS32ATTR_EXCEPT
NDS32ATTR_ISR

AndeSight Project設定

- Include head file : nds32_isr.h
- Project Linker Setting
  - Link libnds32_isr.a



Properties for demo-int-c-ext

type filter text

- Resource
- Builders
- C/C++ Build
  - Build Variables
  - Discovery Options
  - Environment
  - Logging
  - Settings
  - Target Configuration
  - Tool Chain Editor
- C/C++ General
- Project References
- Run/Debug Settings

Settings

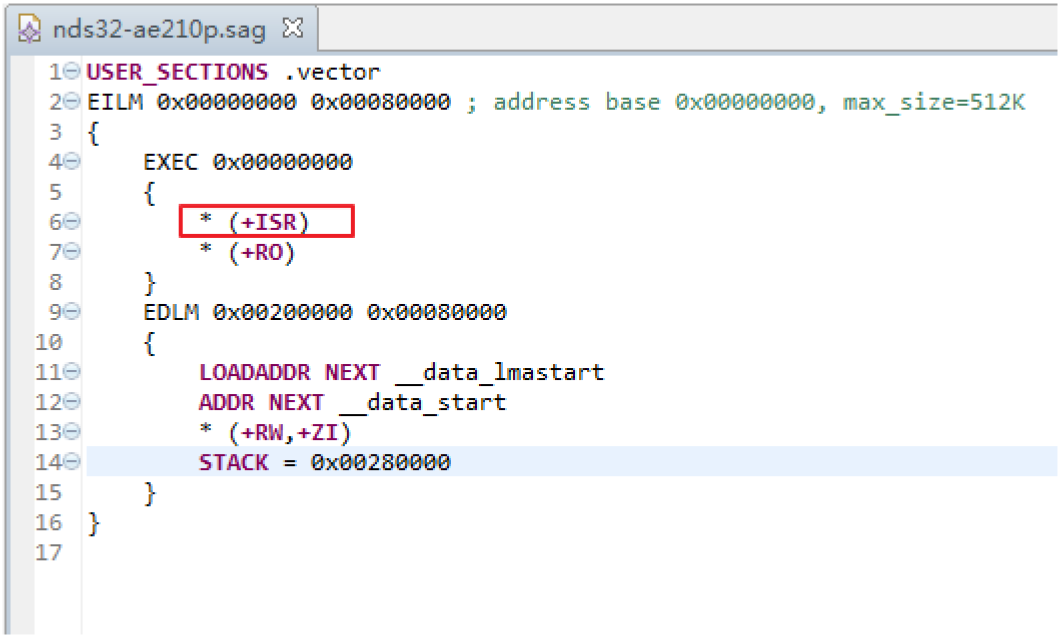
- nds32le-elf-mculib-v3m Configurations
- Andes C Compiler
  - Preprocessor
  - Symbols
  - Directories
  - Optimization
  - Debugging
  - Warnings
  - Miscellaneous
- Andes C Linker
  - General
  - Libraries
  - Miscellaneous
  - Shared Library Settings

Libraries (-l)

nds32_isr

中断向量表的设置(1)

❖ "+ISR" at SaG



nds32-ae210p.sag

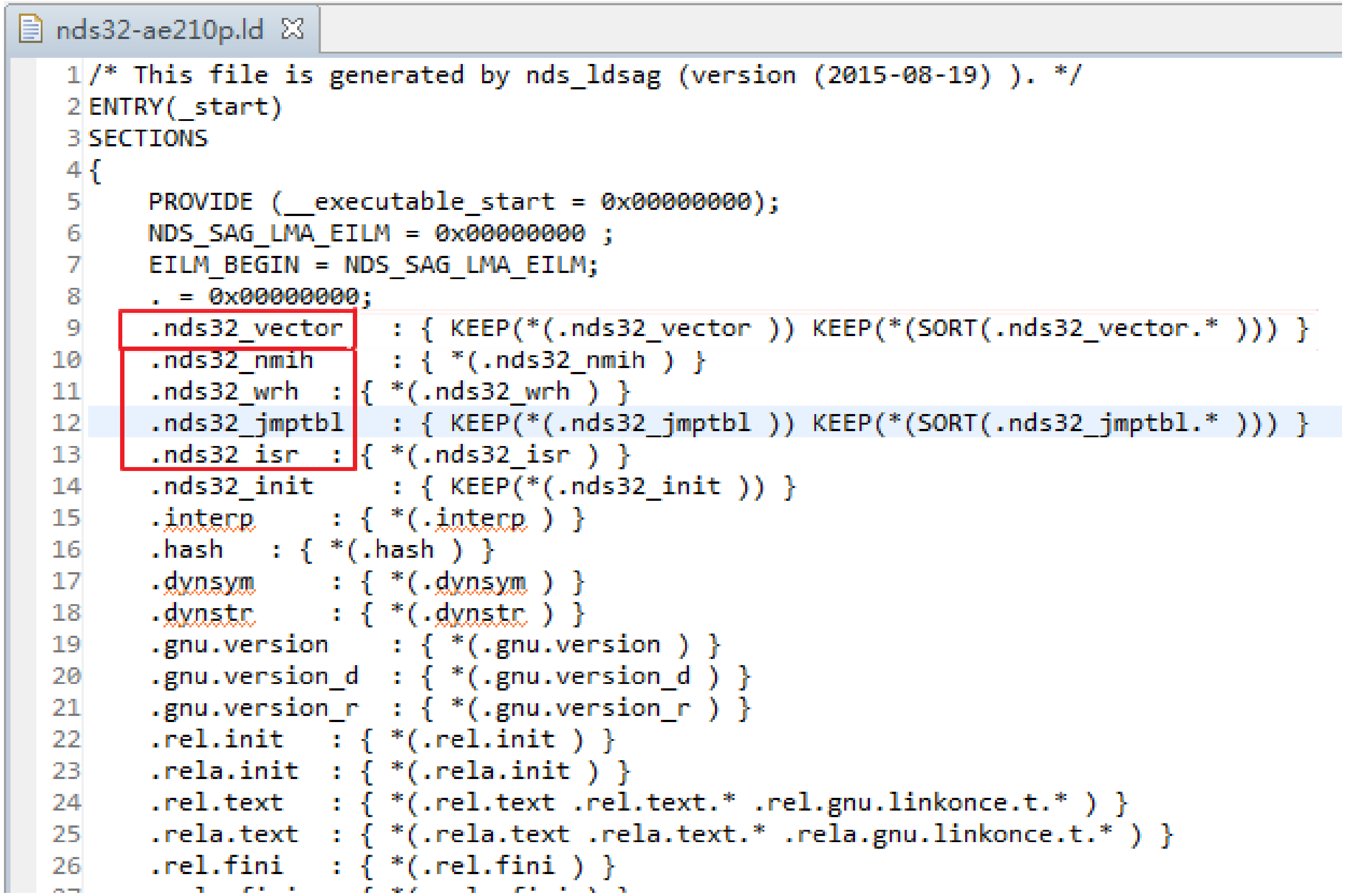
```
USER_SECTIONS .vector
EILM 0x00000000 0x00080000 ; address base 0x00000000, max_size=512K
{
    EXEC 0x00000000
    {
        * (+ISR)
        * (+RO)
    }
    EDLM 0x00200000 0x00080000
    {
        LOADADDR NEXT __data_lmastart
        ADDR NEXT __data_start
        * (+RW,+ZI)
        STACK = 0x00280000
    }
}
```

❖ 关于SaG语法可以参考技术文档：Andes 的分散聚合（SAG）机制

- http://www.andestech.com/cn/news-events/technical-article/2014/Andes20141008.pdf

中断向量表的设置(2)

■ Link Script File


nds32-ae210p.ld

```
/* This file is generated by nds_ldsag (version (2015-08-19) ). */
ENTRY(_start)
SECTIONS
{
    PROVIDE (__executable_start = 0x00000000);
    NDS_SAG_LMA_EILM = 0x00000000 ;
    EILM_BEGIN = NDS_SAG_LMA_EILM;
    . = 0x00000000;
    .nds32_vector    : { KEEP(*(.nds32_vector )) KEEP(*(SORT(.nds32_vector.* ))) }
    .nds32_nmih      : { *(.nds32_nmih ) }
    .nds32_wrh  : { *(.nds32_wrh ) }
    .nds32_jmptbl    : { KEEP(*(.nds32_jmptbl )) KEEP(*(SORT(.nds32_jmptbl.* ))) }
    .nds32_isr  : { *(.nds32_isr ) }
    .nds32_init      : { KEEP(*(.nds32_init )) }
    .interp     : { *(.interp ) }
    .hash   : { *(.hash ) }
    .dynsym     : { *(.dynsym ) }
    .dynstr     : { *(.dynstr ) }
    .gnu.version     : { *(.gnu.version ) }
    .gnu.version_d   : { *(.gnu.version_d ) }
    .gnu.version_r   : { *(.gnu.version_r ) }
    .rel.init   : { *(.rel.init ) }
    .rela.init  : { *(.rela.init ) }
    .rel.text   : { *(.rel.text .rel.text.* .rel.gnu.linkonce.t.* ) }
    .rela.text  : { *(.rela.text .rela.text.* .rela.gnu.linkonce.t.* ) }
    .rel.fini   : { *(.rel.fini ) }
```

NDSATTR_RESET(1)

- void **NDS32ATTR_RESET**(“<option_list>”) reset_handler(void)
  - <option_list> contains zero or more of the following separated by “;”
    - vectors=XXX
    - nmi_func=YYY
    - warm_func=ZZZ

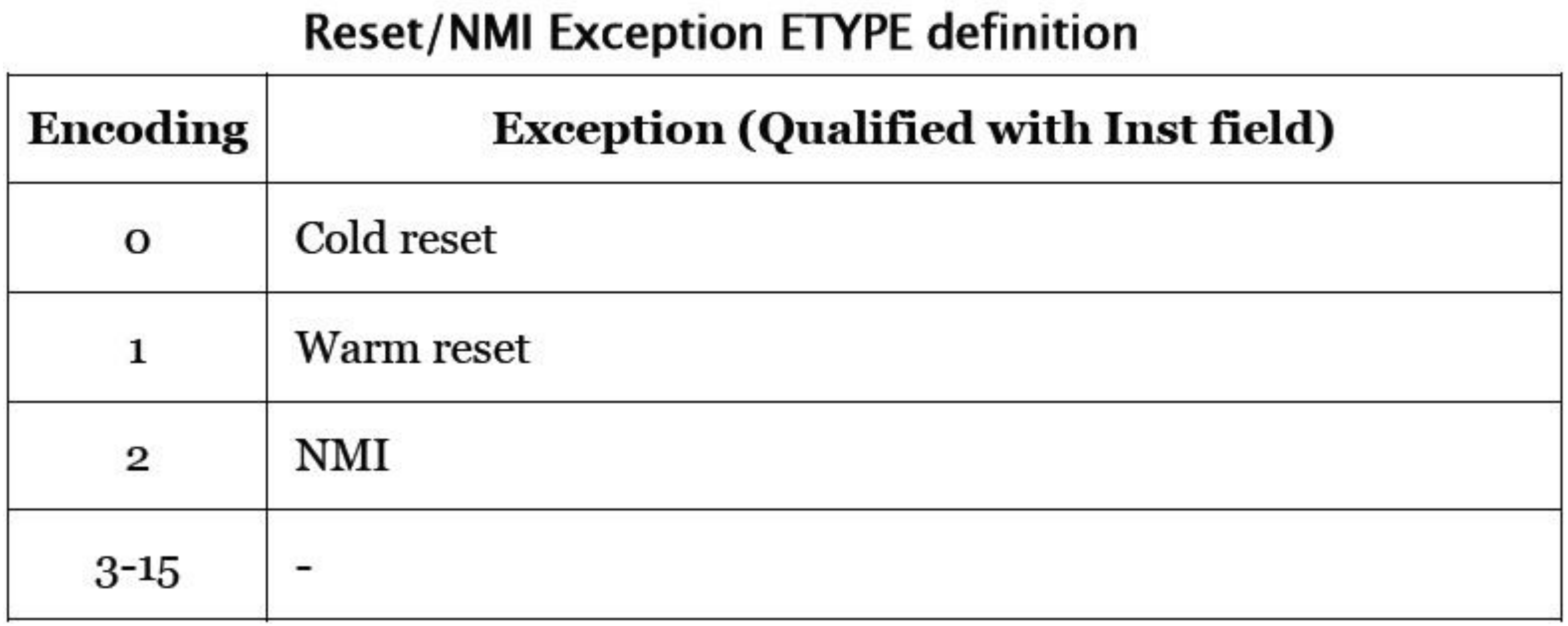
NDSATTR_RESET(2)

❖ Example

```
void
NDS32ATTR_RESET("vectors=32;nmi_func=nmi_handler;warm_func=warm_handler") reset_handler(void)
{
        _nds32_init_mem();
        __cpu_init();
        __c_init();
        __soc_init();
        main();
}
```

区分NMI 和 Warm_Reset

- Cold Reset, NMI 和 Warm Reset共用vector 0
  - 用ir6→ETYPE 区分
- 系统根据ir6→ETYPE自动跳转到nmi_fun或者warm_fun



Reset/NMI Exception ETYPE definition

Encoding	Exception (Qualified with Inst field)
0	Cold reset
1	Warm reset
2	NMI
3-15	-

_nds32_init_mem()

- 内存初始化函数
  - 系统一上电或者是复位，通常memory是不可用的，在函数使用到memory之前准备好memory空间
- 系统reset后最先被呼叫
  - 必须是个leaf function，因为leaf function可以做到不push stack, 即不使用还未准备好的memory

Intrinsic Function(1)

- AndesCore™ Intrinsic Function
  - 和CPU紧密相关
    - 用于设置系统寄存器，cache操作等
  - 通常对应于一条或几条机器指令，以函数的形式来使用
  - 类似于inline assembly，但避免了inline assembly使用上的较复杂语法

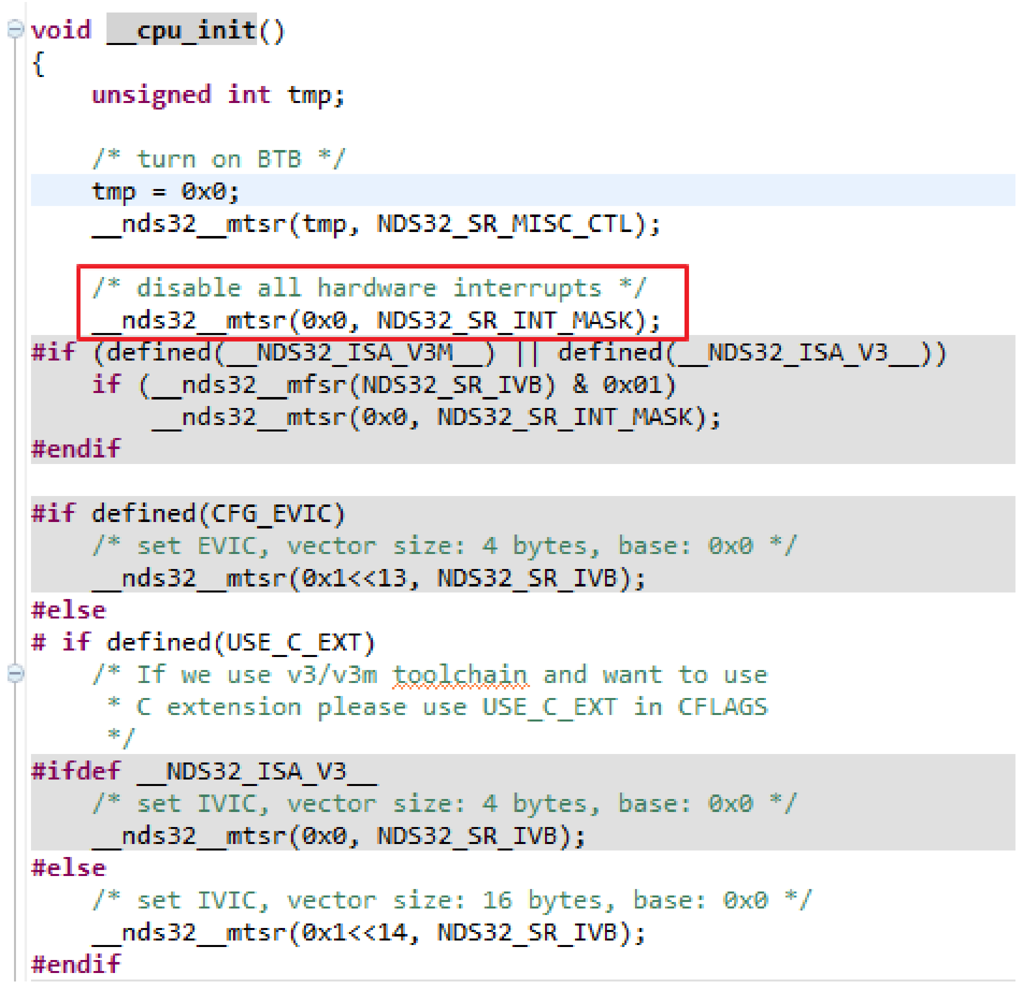
Intrinsic Function(2)

- AndesCore™ intrinsic function例子

Intrinsic Function Syntax	Mapped Andes Instruction
unsigned int __nds32__mfsr (const enum nds32_sr srname)	MFSR
unsigned int __nds32__mfusr (const enum nds32_usr usrname)	MFUSR
void __nds32__mtsr (unsigned int val, const enum nds32_sr srname)	MTSR
void __nds32__mtsr_isb(unsigned int val, const enum nds32_sr srname)	MTSR ISB
void __nds32__mtsr_dsb(unsigned int val, const enum nds32_sr srname)	MTSR DSB
void __nds32__mtusr (unsigned int val, const enum nds32_usr usrname)	MTUSR

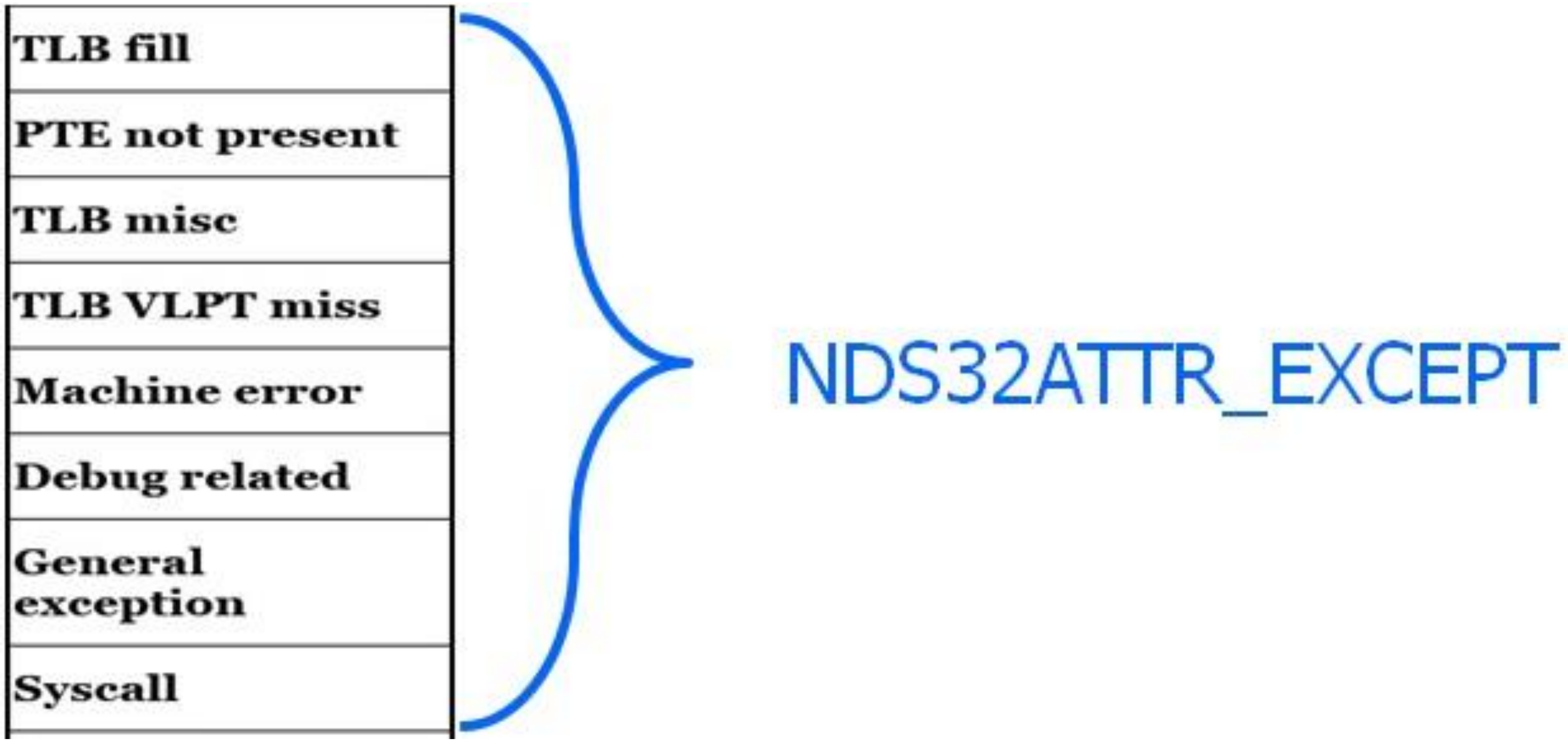
Intrinsic Function(3)

❖ _cpu_init()函数使用intrinsic function



```
void __cpu_init()
{
    unsigned int tmp;

    /* turn on BTB */
    tmp = 0x0;
    __nds32__mtsr(tmp, NDS32_SR_MISC_CTL);

    /* disable all hardware interrupts */
    __nds32__mtsr(0x0, NDS32_SR_INT_MASK);
#if (defined(__NDS32_ISA_V3M__) || defined(__NDS32_ISA_V3__))
    if (__nds32__mfsr(NDS32_SR_IVB) & 0x01)
        __nds32__mtsr(0x0, NDS32_SR_INT_MASK);
#endif

#if defined(CFG_EVIC)
    /* set EVIC, vector size: 4 bytes, base: 0x0 */
    __nds32__mtsr(0x1<<13, NDS32_SR_IVB);
#else
# if defined(USE_C_EXT)
    /* If we use v3/v3m toolchain and want to use
     * C extension please use USE_C_EXT in CFLAGS
     */
#ifdef __NDS32_ISA_V3__
    /* set IVIC, vector size: 4 bytes, base: 0x0 */
    __nds32__mtsr(0x0, NDS32_SR_IVB);
#else
    /* set IVIC, vector size: 16 bytes, base: 0x0 */
    __nds32__mtsr(0x1<<14, NDS32_SR_IVB);
#endif
```

NDS32ATTR_EXCEPT(1)



ID=1	TLB fill
ID=2	PTE not present
	TLB misc
	TLB VLPT miss
	Machine error
	Debug related
	General exception
	Syscall

NDS32ATTR_EXCEPT(2)

- void **NDS32ATTR_EXCEPT** (“id=xxx[;save_caller_regs;<is_nested>]”) excpt_hdlr(int vid)
  - id=xxx, id should be 1 to 8

```
void NDS32ATTR_EXCEPT("not_nested;id=8")
syscall_ISR()
{
    printf(“syscall except ”);
    return;
}
```

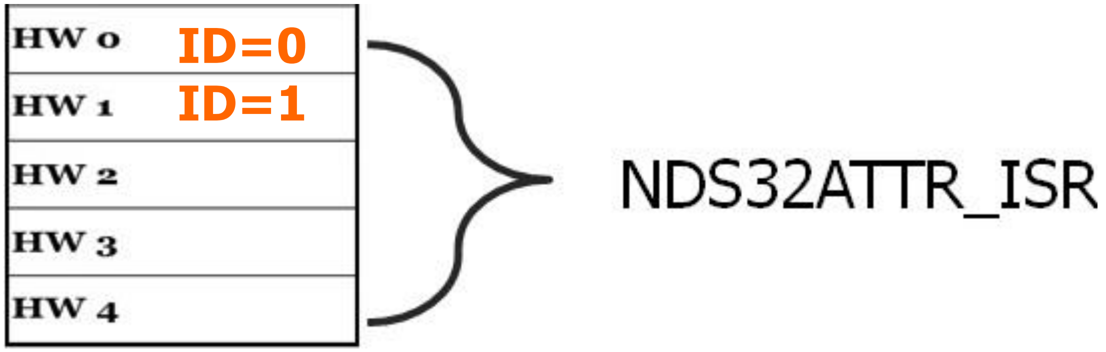
NDS32ATTR_ISR(1)

- void **NDS32ATTR_ISR** (“id=xxx[;save_caller_regs;<is_nested>]”) intr_hdlr(int vid)
  - id=xxx,id should be 0 to 63
  - <save_reg>
  - <is_nested>

NDS32ATTR_ISR(2)

- Example

```
void NDS32ATTR_ISR("not_nested;id=0,1")
HW01_ISR(int vid)
{
    printf(“hw0,1 interrupt isr”);
    return;
}
```



<save_reg> (1)

- 两种：save_caller_regs 或者 save_all_regs.
  - (Default: save_caller_regs)
  - save_caller_regs
    - 保存caller寄存器，通常ISR都是采用这种

```
int foo()
{
    bar();
}
```

→

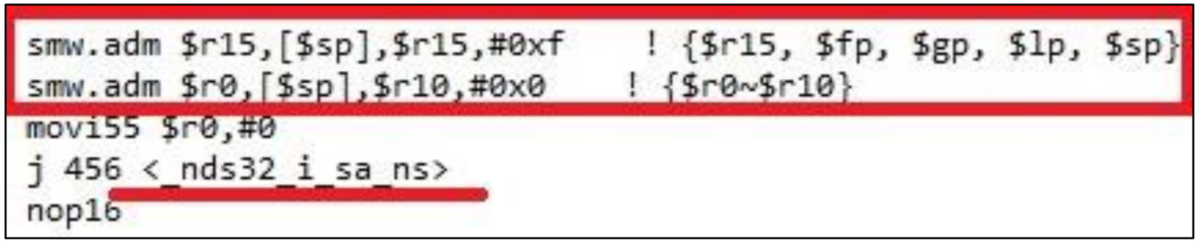
foo(): *caller*
bar(): *callee*

  - save_all_regs
    - 保存所有寄存器，这种方式用于上下文切换（context switch）

ANDES TECHNOLOGY

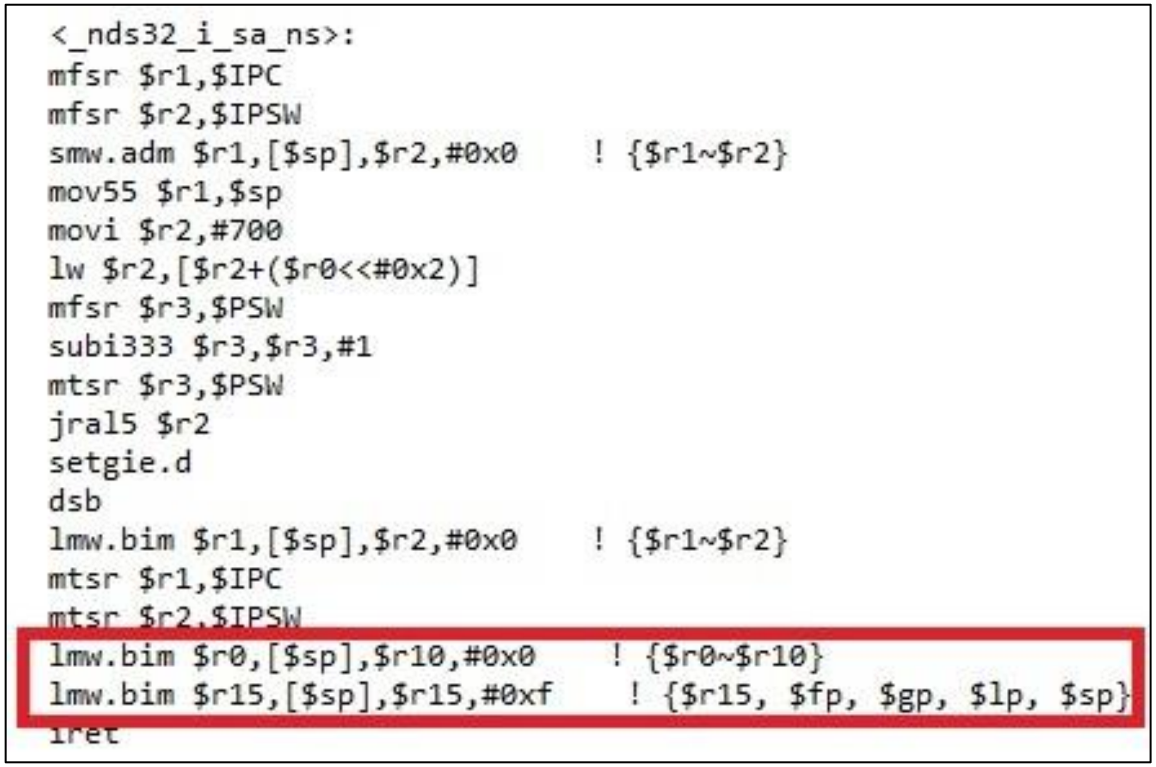
<save_reg> (2)

- save_all_regs(v3m toolchain)



```
smw.adm $r15,[$sp],$r15,#0xf     ! {$r15, $fp, $gp, $lp, $sp}
smw.adm $r0,[$sp],$r10,#0x0     ! {$r0~$r10}
movi55 $r0,#0
j 456 <_nds32_i_sa_ns>
nop16
```

Entry the interrupt

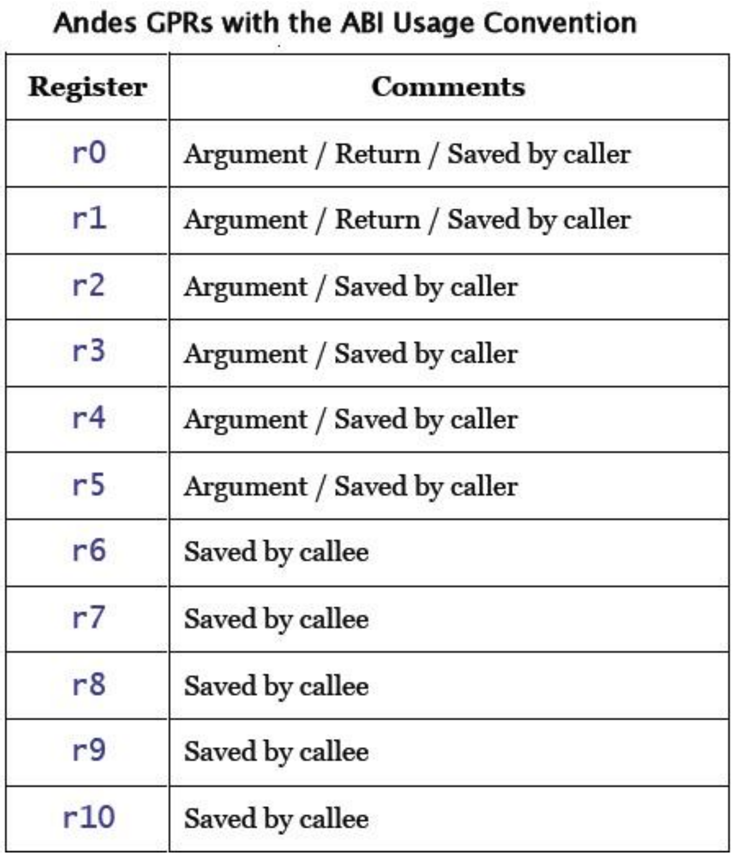


```
<_nds32_i_sa_ns>:
mfsr $r1,$IPC
mfsr $r2,$IPSW
smw.adm $r1,[$sp],$r2,#0x0     ! {$r1~$r2}
mov55 $r1,$sp
movi $r2,#700
lw $r2,[$r2+($r0<<#0x2)]
mfsr $r3,$PSW
subi333 $r3,$r3,#1
mtsr $r3,$PSW
jral5 $r2
setgie.d
dsb
lmw.bim $r1,[$sp],$r2,#0x0     ! {$r1~$r2}
mtsr $r1,$IPC
mtsr $r2,$IPSW
lmw.bim $r0,[$sp],$r10,#0x0     ! {$r0~$r10}
lmw.bim $r15,[$sp],$r15,#0xf     ! {$r15, $fp, $gp, $lp, $sp}
iret
```

End of the interrupt

<save_reg> (3)

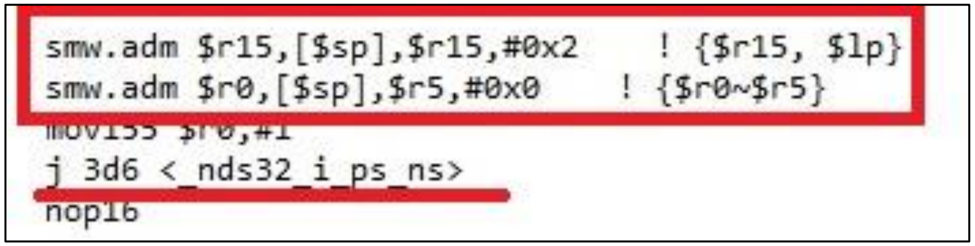
- save_caller_regs
  - What's caller?



Andes GPRs with the ABI Usage Convention

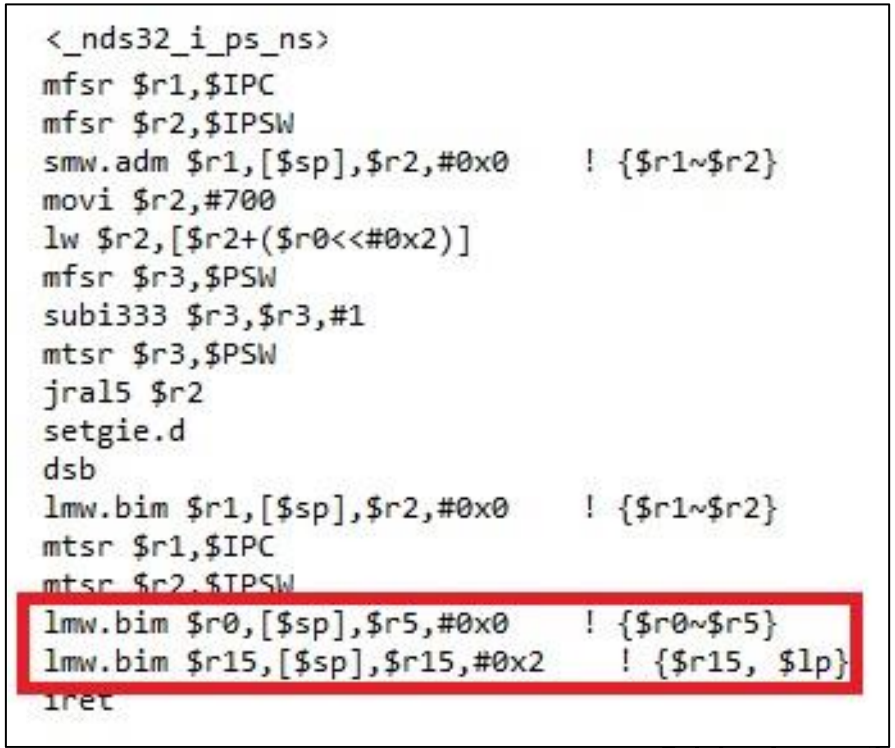
Register	Comments
r0	Argument / Return / Saved by caller
r1	Argument / Return / Saved by caller
r2	Argument / Saved by caller
r3	Argument / Saved by caller
r4	Argument / Saved by caller
r5	Argument / Saved by caller
r6	Saved by callee
r7	Saved by callee
r8	Saved by callee
r9	Saved by callee
r10	Saved by callee

Entry the interrupt



```
smw.adm $r15,[$sp],$r15,#0x2    ! {$r15, $lp}
smw.adm $r0,[$sp],$r5,#0x0     ! {$r0~$r5}
movi55 $r0,#1
j 3d6 <_nds32_i_ps_ns>
nop16
```

End of the interrupt



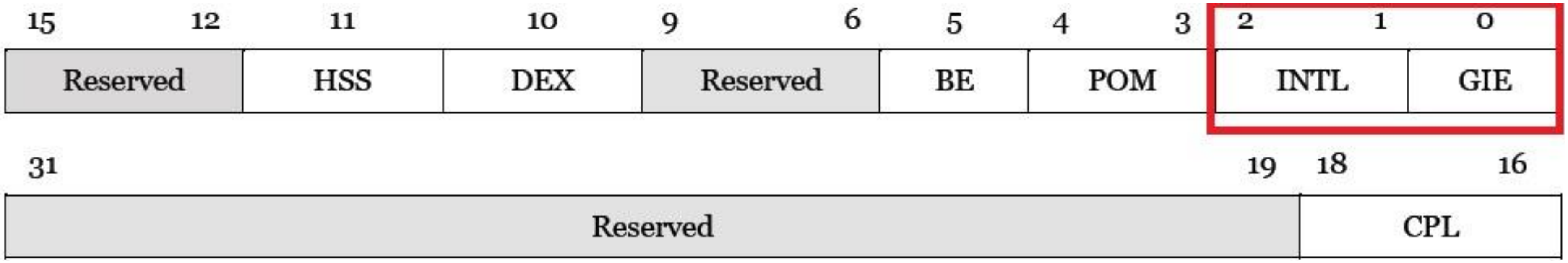
```
<_nds32_i_ps_ns>
mfsr $r1,$IPC
mfsr $r2,$IPSW
smw.adm $r1,[$sp],$r2,#0x0     ! {$r1~$r2}
movi $r2,#700
lw $r2,[$r2+($r0<<#0x2)]
mfsr $r3,$PSW
subi333 $r3,$r3,#1
mtsr $r3,$PSW
jral5 $r2
setgie.d
dsb
lmw.bim $r1,[$sp],$r2,#0x0     ! {$r1~$r2}
mtsr $r1,$IPC
mtsr $r2,$IPSW
lmw.bim $r0,[$sp],$r5,#0x0     ! {$r0~$r5}
lmw.bim $r15,[$sp],$r15,#0x2    ! {$r15, $lp}
iret
```

<is_nested> (1)

- 四种模式：
  - nested ：可被中断嵌套.
  - not_nested ：不可被中断嵌套.
  - ready_nested ：进入中断后手动打开GIE后可被中断嵌套，通常用于完成一小段紧急处理后再打开GIE启用中断嵌套
  - critical ：用于紧急处理，通常很简短，不可被嵌套，而且ISR须要是leaf函数。

<is_nested> (2)

- ir0(PSW) Processor Status Word Register



15–12	11	10	9–6	5	4–3	2–1	0
Reserved	HSS	DEX	Reserved	BE	POM	INTL	GIE

31–19	18–16
Reserved	CPL

- GIE(Global Interrupt Enable)
  - 0:interrupt disable
  - 1:interrupt enable
- INTL(Interruption Stack Level)
  - 0:No interruption
  - 1:Interruption stack level 1
  - 2:Interruption stack level 2
  - …

<is_nested> (3)

- Nested(GIE 启用, INTL 减 1)
  - mfsr $r3,$PSW
  - subi333 $r3,$r3,#1
  - mtsr $r3,$PSW

- Ready_neseted(INTL 减 1)
  - mfsr $r3,$PSW
  - subi333 $r3,$r3,#2
  - mtsr $r3,$PSW

<is_nested> (4)

- No_nested(PSW保持不变)
  - 做保存寄存器相关操作
- Critical
  - 直接跳转到ISR
  - 寄存器也不保存

<is_nested> (5)

<is_nested>	GIE Enable	INTL-1	SAVE_REG
Nested	YES	YES	YES
Ready_Nested	NO	YES	YES
No_Nested	NO	NO	YES
Critical	NO	NO	NO

参考示例代码

- Installation AndeSight folder\Demo\startup\demo-int-c-ext.tgz

总结

- AndesCore™全C嵌入式编程
  - 减轻学习新指令的开销
  - 减少开发时间
  - 便于系统调试和维护

Thank You!

WWW.ANDESTECH.COM