4.8 新增指令

3.8.1 EICALL - 延长间接调用子程序

说明:间接调用由寄存器文件中的Z(16位) 指针和I/O端口中的EIND寄存器指向的子程序。该指令允许调用整个程序存储器空间内的子程序。该指令在双字节PC的设备中是无效的，见ICALL。在EICALL指令执行期间堆栈指针使用一种后进先出的设置。

操作:

(i) PC(15:0)←Z(15:0)
PC(21:16)←EIND

语法:	操作码:	程序计数器:	堆栈:
(i) EICALL	None	See Operation	STACK←PC + 1 SP←SP - 3 (3 bytes, 22 bits)

16位操作码:

1001	0101	0001	1001

状态寄存器(SREG) 和布尔格式:

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

例子: (实践操作程序481.ASM)

```
ldi r16,$05          ; 设置EIND和Z指针
out EIND,r16
ldi r30,$00
ldi r31,$10
eicall               ; 调用$051000
```

Words: 1 (2 bytes)
Cycles: 4 (仅在带22位PC的器件上执行)

4.8.2 EIJMP - 扩展间接跳转

说明:间接跳转到由寄存器文件中的 Z (16 位) 指针和 I/O 端口中的 EIND 寄存器指向的地址。该指令允许间接跳转到整个程序存储器空间。

操作:

(i) PC(15:0) ←Z(15:0)
PC(21:16)←EIND

语法:	操作码:	程序计数器:	堆栈:
(i) EIJMP	None	See Operation	不影响

16位操作码:

1001	0100	0001	1001

状态寄存器(SREG) 和布尔格式:

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

例子: (**实践操作程序**482.ASM)

```
ldi r16,$05          ; 设置EIND和Z指针
out EIND,r16
ldi r30,$00
ldi r31,$10
eijmp                ; 跳转到$051000
```

Words: 1 (2 bytes)
Cycles: 2

4.8.3 ELPM -扩展装载程序存储器

说明:装入由Z寄存器和I/O端口中的RAMPZ寄存器指向的一个字节，将这一字节装入目的寄存器Rd。该指令描述一个百分之百空间有效的常量初始化或常量数据引出。程序存储器是按16位字组织起来，Z寄存器最重要的位选择低字节（0）或高字节（1）。该指令能寻址整个程序存储器空间。该操作不改变Z指针寄存器，或使之增加。增加适用于RAMPZ和Z指针寄存器的整个24位串联。这些合并的结果是不确定的:

```
ELPM r30, Z+
ELPM r31, Z+
```

	操作:		**注释:**
(i)	R0 ← (RAMPZ:Z)		RAMPZ:Z: Unchanged, R0 implied destination register
(ii)	Rd ← (RAMPZ:Z)		RAMPZ:Z: Unchanged
(iii)	Rd ← (RAMPZ:Z)	(RAMPZ:Z) ← (RAMPZ:Z) + 1	RAMPZ:Z: Post incremented

	语法:	**操作码:**	**程序计数器:**
(i)	ELPM	None, R0 implied	PC ← PC + 1
(ii)	ELPM Rd, Z	0 ≤ d ≤ 31	PC ← PC + 1
(iii)	ELPM Rd, Z+	0 ≤ d ≤ 31	PC ← PC + 1

16位操作码:

(i)	1001	0101	1101	1000
(ii)	1001	000d	dddd	0110
(iii)	1001	000d	dddd	0111

状态寄存器(SREG) 和布尔格式:

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

例子: (**实践操作程序**483.ASM)

```
clr r16              ; RAMPZ寄存器清零
out RAMPZ, r16
```

```
clr r31              ; 清除Z指针寄存器的高字节
ldi r30,$F0          ; Z指针寄存器的低字节置1
elpm r16, Z+         ; 从程序装入常数
                     ; RAMPZ:Z (r31:r30)指向的存储器
```

Words: 1 (2 bytes)
Cycles: 3

4.8.4 ESPM - 扩展存储程序存储器

说明:ESPM 指令用来擦除程序存储器中的一页，写程序存储器中的一页（那是已经被擦除的），设置引导装载器锁位。在一些器件中，程序存储器是一次写一字节，另一些器件中在先填充一个暂时页缓冲器后能同时编程一整页。就一切情况而论，程序存储器必须一次被擦除一页。擦除程序存储器时，RAMPZ和Z寄存器被用来页寻址。写程序存储器时，RAMPZ和Z寄存器被用作页或字寻址，R1:R0 寄存器组被当作数据。设置引导装载锁位时，R1:R0 寄存器组被当作数据。参考器件文件以获得ESPM 用法的详细说明。该指令可寻址整个程序存储器。

	操作:	注释:
(i)	(RAMPZ:Z)←$ffff	擦除程序存储器页
(ii)	(RAMPZ:Z)←R1:R0	写程序存储器字
(iii)	(RAMPZ:Z)←R1:R0	写暂时页缓冲器
(iv)	(RAMPZ:Z)←TEMP	写暂时页缓冲器到程序存储器
(v)	BLBITS ←R1:R0	设置引导装载器锁位

	语法:	操作码:	程序计数器:
(i)-(v)	ESPM	None	PC ←PC + 1

16位操作码:

1001	0101	1111	1000

状态寄存器(SREG) 和布尔格式:

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

Example: (实践操作程序484.ASM)

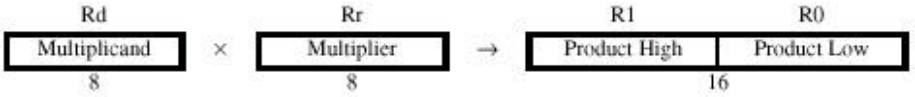
```
                     ; 此例说明了对于带页写的器件的字ESPM写
clr r31              ;Z寄存器的高字节清零
clr r30              ; Z寄存器的低字节清零
ldi r16,$F0          ; 装入RAMPZ寄存器
out RAMPZ, r16 ;
ldi r16, $CF         ;存入数据
mov r1, r16
ldi r16, $FF
mov r0, r16
ldi r16,$03          ; ESPM使能,擦除页
out SPMCR, r16 ;
espm                 ; 从$F00000开始擦除页
ldi r16,$01          ; ESPM使能,将R1:R0存入暂时缓冲器
out SPMCR, r16 ;
espm                 ; 执行ESPM,将R1:R0存入暂时缓冲器的$F00000处
```

```
ldi r16,$05     ; ESPM使能,写页
out SPMCR, r16 ;
espm            ; 执行ESPM,将暂时缓冲器存入从程序存储器地址$F00000开始的页
```

Words: 1 (2 bytes)
Cycles: 依操作而定

4.8.5 FMUL - 小数乘法

说明: 该指令完成8位×8位→16位的无符号数乘法操作并把结果左移一位。



(N.Q) 表示一个小数点左边有N个二进制数位，小数点右边有Q个二进制数位的小数。以(N1.Q1)和(N2.Q2)为格式的两个数相乘产生格式为((N1+N2).(Q1+Q2))的结果。为处理符号，以(1.7) 格式作输入，产生(2.14) 格式的结果。结果的高字节要左移一位以使结果的格式与输入的一致。FMUL指令在与MUL指令相同的周期内合并了左移操作。

被乘数Rd和乘数Rr是两个包含无符号小数的寄存器，固定的小数位在第6位和第7位之间。16位无符号小数结果的固定小数位在第14位和第15位之间，即存放在R1(高字节)和R0(低字节)。

操作:

(i) R1:R0 ← Rd × Rr (unsigned (1.15) ← unsigned (1.7) × unsigned (1.7))

	语法:	操作码:	程序计数器:
(i)	FMUL Rd,Rr	16 ≤ d ≤ 23, 16≤ r ≤ 23	PC ← PC + 1

16位操作码:

0000	0011	0ddd	1rrr

状态寄存器(SREG) 和布尔格式:

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	⇔	⇔

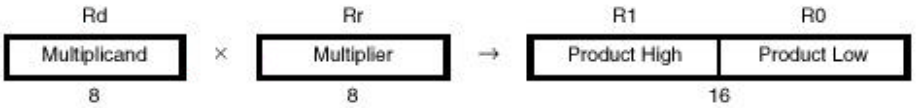
C: R16
如果结果的第15 位在左移前被置1则置1，否则清除。
Z: R15 •R14 •R13 •R12 •R11 •R10 •R9 •R8 •R7 R6 R5 R4 R3 R2 •R1 R0
如果结果是$0000则置1，否则清除。
R (结果)操作后等于R1,R0。

例子: (实践操作程序485.ASM)

```
fmul r23,r22        ; 无符号数r23和r22以(1.7)格式相乘,产生(1.15)格式的结果
movw r22,r0         ; 复制结果回r23:r22
```

Words: 1 (2 bytes)
Cycles: 2

4.8.6 FMULS -有符号数乘法



说明: 该指令完成8位×8位→16位的有符号数乘法操作并把结果左移一位。
(N.Q) **表示一个小数点左边有N个二进制数位，小数点右边有Q个二进制数位的小数。以**(N1.Q1)**和**(N2.Q2)**为格式的两个数相乘产生格式为**((N1+N2).(Q1+Q2))**的结果。为处理符号，以**(1.7) **格式作输入，产生**(2.14) **格式的结果。结果的高字节要左移一位以使结果的格式与输入的一致。**FMULS **指令在与**MULS **指令相同的周期内合并了左移操作。**
被乘数Rd**和乘数**Rr**是两个包含有符号小数的寄存器，固定的小数位在第**6**位和第**7**位之间。**16**位有符号小数结果的固定小数位在第**14**位和第**15**位之间，即存放在**R1(**高字节**)**和**R0(**低字节**)。**

操作:

(i) R1:R0 ← Rd × Rr (signed (1.15) ← signed (1.7) × signed (1.7))

	语法:	**操作码:**	**程序计数器:**
(i)	FMUL Rd,Rr	16 ≤ d ≤ 23, 16≤ r ≤ 23	PC ← PC + 1

16位操作码:

0000	0011	1ddd	0rrr

状态寄存器(SREG) 和布尔格式:

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	⇔	⇔

C: R16
如果结果的第15 **位在左移前被置**1**则置**1**，否则清除。**
Z: R15 •R14 •R13 •R12 •R11 •R10 •R9 •R8 •R7 R6 R5 R4 R3 R2 •R1 R0
如果结果是$0000**则置**1**，否则清除。**
R**(结果)操作后等于**R1,R0**。**
例子: **(实践操作程序**486.ASM)

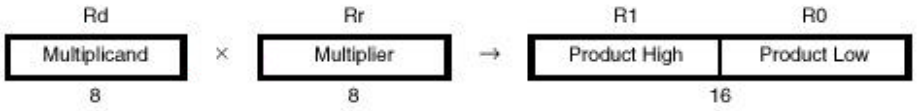
```
fmuls r23,r22 ; 有符号数r23和r22以(1.7)格式相乘,产生(1.15)格式的结果

movw r22,r0碍  ; 复制结果回r23:r22
```

Words: 1 (2 bytes)
Cycles: 2

4.8.7 FMULSU - 有符号小数和无符号小数乘法

说明: 该指令完成8位×8位→16位的有符号数乘法操作并把结果左移一位。



(N.Q) **表示一个小数点左边有**N**个二进制数位，小数点右边有**Q**个二进制数位的小数。以**(N1.Q1)**和**(N2.Q2)**为格式的两个数相乘产生格式为**((N1+N2).(Q1+Q2))**的结果。为处理符号，以**(1.7) **格式作输入，产生**(2.14) **格式的结果。结果的高字节要左移一位以使结果的格式与输入的一致。**FMULSU**指令在与**MULSU**指令相同的周期内合并了左移操作。被乘数**Rd**和乘数**Rr**是两个包含小数的寄存器，暗含的小数位在第**6**位和第**7**位之间。被乘数**Rd**是一个有符号小数，乘数**Rr**是一个无符号小数。**16**位有符号小数结果暗含的小数位在第**14**位和第**15**位之间，即存放在**R1(高字节)和R0(低字节)。

操作:

(i) R1:R0 ← Rd × Rr (signed (1.15) ← signed (1.7) × unsigned (1.7))

	语法:	操作码:	程序计数器:
(i)	FMULSU Rd,Rr	16 ≤ d ≤ 23, 16≤ r ≤ 23	PC ← PC + 1

16位操作码:

0000	0011	1ddd	1rrr

状态寄存器(SREG) 和布尔格式:

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	⇔	⇔

C: R16
如果结果的第15 **位在左移前被置**1**则置**1**，否则清除。**
Z: R15 •R14 •R13 •R12 •R11 •R10 •R9 •R8 •R7 R6 R5 R4 R3 R2 •R1 R0
如果结果是$0000**则置**1**，否则清除。**
R **(结果)操作后等于**R1,R0**。**

例子: **(实践操作程序**487.ASM)

```
fmulSU r23,r22  ; 有符号数r23和无符号数r22以(1.7)格式相乘,产生(1.15)格式的结果
movw r22,r0     ; 复制结果回r23:r22
```

Words: 1 (2 bytes)
Cycles: 2

4.8.8 MOVW -拷贝寄存器字

说明：该指令完成将一个寄存器组拷贝到另一个寄存器组的操作。源寄存器组 Rr+1:Rr **不改变，目的寄存器组**Rd+1:Rd**则是** Rr + 1:Rr**所含内容的拷贝。**

操作:

(i) Rd+1:Rd ← Rr+1:Rr

语法:	**操作码:**	**程序计数器:**
(i) MOVW Rd,Rr	d ∈ {0,2,...,30}, r ∈ {0,2,...,30}	PC ← PC + 1

16位操作码:

0000	0001	dddd	rrrr

状态寄存器(SREG) 和布尔格式：

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

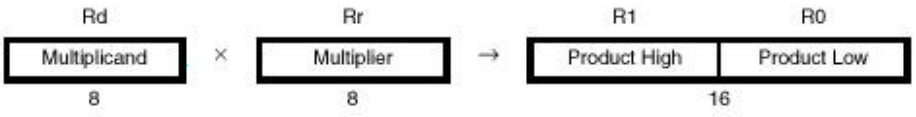
例子： (**实践操作程序**488.ASM)

```
            movw r16,r0          ; 拷贝 r1:r0 到 r17:r16
            call check           ; 调用子程序
            ...
check:      cpi r16,$11          ; r16 - $11
            ...
            cpi r17,$32          ; r17 - $32
            ...
            ret                  ; 子程序返回
```

Words: 1 (2 bytes)
Cycles: 1

4.8.9 MULS -有符号数乘法

说明：该指令完成8**位×**8**位→**16**位有符号数乘法的操作。**



被乘数Rd**和乘数**Rr**是两个包含有符号数的寄存器。**16**位有符号结果存放在**R1 (**高字节**) **和** R0 (**低字节**)**中。**

操作:

(i) R1:R0 ← Rd × Rr (signed ← signed × signed)

语法:	**操作码:**	**程序计数器:**

(i)	MULS Rd,Rr	16 ≤ d ≤ 31, 16 ≤ r ≤ 31	PC ← PC + 1

16位操作码:

0000	0010	dddd	rrrr

状态寄存器(SREG) 和布尔格式:

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	⇔	⇔

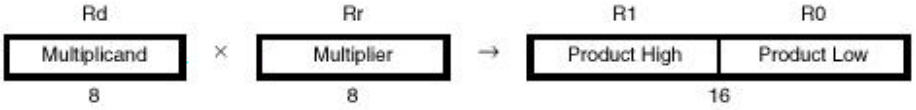
C: R15
如果结果的第15 位被置1则置1，否则清除。
Z: R15 •R14 •R13 •R12 •R11 •R10 •R9 •R8 •R7 R6 R5 R4 R3 R2 •R1 R0
如果结果是$0000则置1，否则清除。
R (结果)操作后等于R1,R0。
例子: (实践操作程序489.ASM)

```
muls r21,r20      ; 有符号数r21和r20相乘
movw r20,r0       ; 拷贝结果回 r21:r20
```

Words: 1 (2 bytes)
Cycles: 2

4.8.10 MULSU - 有符号数与无符号数乘法

说明: 该指令完成8位×8位→16位的一个有符号数和一个无符号数乘法的操作。



被乘数Rd和乘数Rr是两个寄存器。被乘数Rd是有符号数，乘数Rr是无符号数。16位有符号结果存放在R1 (高字节) 和 R0 (低字节)中。

操作:

(i)	R1:R0 ← Rd × Rr	(signed ← signed × signed)

	语法:	操作码:	程序计数器:
(i)	MULSU Rd,Rr	16 ≤ d ≤ 23, 16 ≤ r ≤ 23	PC ← PC + 1

16位操作码:

0000	0011	0ddd	0rrr

状态寄存器(SREG) 和布尔格式:

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	⇔	⇔

C: R15
如果结果的第15 位被置1则置1，否则清除。
Z: R15 •R14 •R13 •R12 •R11 •R10 •R9 •R8 •R7 R6 R5 R4 R3 R2 •R1 R0
如果结果是$0000则置1，否则清除。
R (结果)操作后等于R1,R0。

例子: (实践操作程序4810.ASM)

```
mulsu r21,r20     ; 有符号数r21和无符号数r20相乘得有符号数结果。
movw r20,r0       ; 拷贝结果回 r21:r20
```

Words: 1 (2 bytes)
Cycles: 2

4.8.11 SPM -存储程序存储器

说明:SPM 指令可用来擦除程序存储器的一页，写程序存储器中的一页（那是已经被擦除的），设置引导装载器锁位。在一些器件中，程序存储器是一次写一字节，另一些器件中在先填充一个暂时页缓冲器后能同时编程一整页。就一切情况而论，程序存储器必须一次被擦除一页。擦除程序存储器时，Z寄存器被用来页寻址。写程序存储器时，Z寄存器被用作页或字寻址，R1:R0 寄存器组被当作数据。设置引导装载器锁位时，R1:R0 寄存器组被当作数据。参考器件文件以获得SPM用法的详细说明。该指令可寻址程序存储器的前64K字节（32K字）。

	操作:	注释:
(i)	(Z)←$ffff	擦除程序存储器页
(ii)	(Z)←R1:R0	写程序存储器字
(iii)	(Z)←R1:R0	写暂时页缓冲器
(iv)	(Z)←TEMP	写暂时页缓冲器到程序存储器
(v)	BLBITS←R1:R0	设置引导装载器锁位

	语法:	操作码:	程序计数器:
(i)-(v)	SPM	None	PC ← PC + 1

16位操作码:

1001	0101	1110	1000

状态寄存器(SREG) 和布尔格式:

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

例子: (**实践操作程序**4811.ASM)

```
                        ; 此例显示对带字写的器件的SPM字写
    ldi r31, $F0        ; 装入 Z 的高字节
    clr r30             ; 清除 Z 的低字节
    ldi r16, $CF        ; 装入数据以便存储
    mov r1, r16
    ldi r16, $FF
    mov r0, r16
    ldi r16,$03         ; SPM使能, 擦除页
    out SPMCR, r16      ;
    spm                 ; 从$F000开始擦除页
    ldi r16,$01         ; SPM使能,存储到存储器
    out SPMCR, r16      ;
    spm                 ;执行SPM操作, 存储R1:R0 到存储器的$F000单元。
```

Words: 1 (2 bytes)
Cycles: 依据操作而定