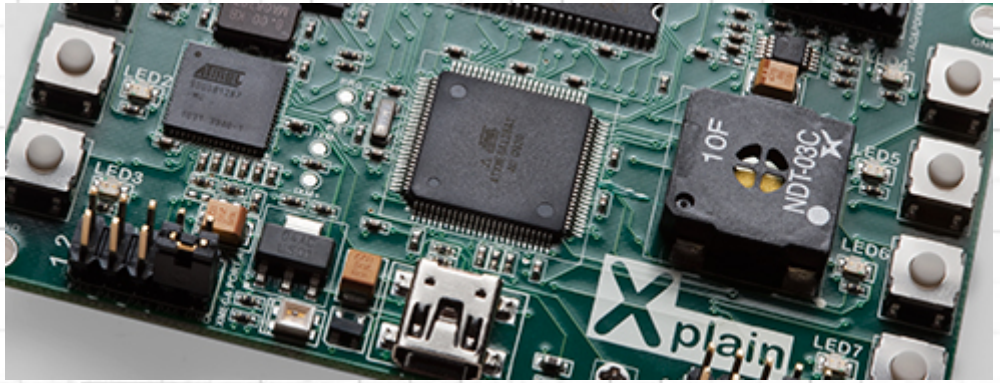
Percorsi abilitativi speciali
Elettronica programmabile

03. L'insieme delle istruzioni AVR

Roberto Roncella

La famiglia AVR (1)

- Scegliamo l’assembly di questa famiglia di microcontrollori ATMEL a 8 b
  - Esempio di dimensioni medie: ATXMEGA32A4U
    - 16384 istruzioni da 16 b nella memoria di programma
      - Cioè 32 kB, organizzati in righe da 2 B
    - 4096 byte (4 kB) nella memoria dati
    - Numerose periferiche
      - USB,
      - Convertitori AD e DA
      - Comparatori, contatori
      - Porte seriali
      - Numerosi pin di I/O



La famiglia AVR (2)

- Motivazioni
  - Basso costo e grandissima diffusione
  - Strumenti di sviluppo facilmente accessibili
    - ***Data sheets*** completi di dispositivi e ***assembly***
    - ***Software development kit*** fornito dal produttore (AVRStudio6)
      - Con ***Integrated Development Environment*** (IDE)
    - ***Application notes*** e ***reference designs***
    - ***Development boards*** e ***demo boards***
    - Programmatori a basso costo

Notazioni flag (1)

- Registro di stato
  - CPU_SREG Registro di stato, nello spazio di I/O
    - I suoi bit, detti flag, danno informazioni sull’esito dell’esecuzione di una istruzione
    - Sono normalmente indicati simbolicamente da una lettera maiuscola, riconosciuta dall'assembler
    - Sono usati dalle istruzioni di controllo del programma per influenzare il flusso di esecuzione delle istruzioni

Notazioni flag (2)

- **C** Carry flag in status register (SREG0)
  - Il valore C = 1 significa che un'istruzione aritmetica su numeri interi senza segno ha generato un risultato non rappresentabile
    - R < 0 oppure R > 255
- **Z** Zero flag in status register (SREG1)
  - Il valore Z = 1 significa che un'istruzione aritmetica o logica ha generato un risultato nullo
    - R = 0

Notazioni flag (3)

- **N** Negative flag in status register (SREG2)
  - Il valore N = 1 significa che il valore memorizzato come risultato, interpretato come intero con segno, ha valore negativo, cioè R7 = 1
- **V** Two's complement overflow indicator (SREG3)
  - Il valore V = 1 significa che un'istruzione aritmetica tra interi con segno ha generato un risultato non rappresentabile
    - R < −128 oppure R > 127
  - In questo caso il segno del risultato calcolato dalla macchina è opposto a quello del risultato corretto

Notazioni flag (4)

- **S** (N ⊕ V) For signed tests (SREG4)
  - L'operazione ⊕ combina due bit dando 1 solo se gli argomenti sono diversi
  - Il valore di S indica il segno effettivo di una operazione aritmetica tra interi con segno
  - La risposta è corretta anche in caso di overflow
    - In assenza di overflow il risultato è negativo (S = 1) quando N è 1; in presenza di overflow avviene il contrario
- **H** Half Carry flag in the status register (SREG5)
  - Il valore di H ha lo stesso significato di C, ma riferito ai valori contenuti nei 4 b (digit) meno significativi degli operandi

Notazioni flag (5)

- T Transfer bit used by BLD and BST (SREG6)
  - Bit di appoggio in cui è possibile memorizzare o riprendere bit di altri registri interni
    - BST (Bit STore) memorizza
    - BLD (Bit LoaD) riprende
- I Global interrupt enable/disable flag (SREG7)
  - Bit di abilitazione della funzione di interruzione del processore
  - Esistono istruzioni apposite per impostare I
    - SEI (SEt Interrupt)
    - CLI (CLear Interrupt)

Notazioni flag (6)

- Effetto di una operazione sui flag
  - ⇔ Flag affected by instruction
    - Il flag sarà modificato dall'esecuzione dell'istruzione in accordo al suo esito
  - 0 Flag cleared by instruction
    - Il flag è azzerato
  - 1 Flag set by instruction
    - Il flag è posto a 1
  - – Flag not affected by instruction
    - Il flag mantiene il valore precedente

Notazioni valori (1)

- Registri e operandi
  - Rd Destination (and source) register in the register file
    - Registro contenente uno degli operandi e destinato a ricevere il risultato
    - Se non specificato diversamente, è $0 \leq d \leq 31$, perché i registri interni sono 32.
      - Alcune istruzioni operano solo con parte dei registri, per ridurre i bit per codificare l'indirizzo dell'operando
  - Rr Source register in the register file
    - Secondo registro sorgente, contenente l’altro termine nelle istruzioni a due operandi

Notazioni valori (2)

- R Result after instruction is executed
  - Valore del risultato dell'operazione, che sarà posto di norma nel registro Rd
  - Per indicare i bit del risultato, oppure di uno dei registri sorgente, viene posto il numero del bit dopo il nome del registro
    - R = R7 (MSB), R6, R5, R4, R3, R2, R1, R0 (LSB)
- K Constant data
  - Valore costante su 8 b, con $0 \leq K \leq 255$
- k Constant address
  - Indirizzo di memoria su 16 b, con $0 \leq k \leq 65535$

Notazioni valori (3)

- b Bit in the register file or I/O register (3 bit)
  - Valore 0 ≤ b ≤ 7 che indica un particolare bit di un registro interno del register file o di I/O
  - Il bit 0 è il meno significativo, il 7 ha peso 2^7
- s Bit in the status register (3 bit)
  - Valore 0 ≤ s ≤ 7 che indica un particolare bit del registro di stato, cioè SREG
  - Di norma si fa riferimento a questi bit (i flag) usando particolari lettere maiuscole
  - L'assembler riconosce queste lettere e assegna a s (in uno dei campi dell'istruzione) il giusto valore

Notazioni valori (4)

- **X, Y, Z** Indirect address register
  - Registri puntatori contenenti indirizzi (16 b), ottenuti combinando coppie di registri interni
    - X = R27:R26, Y = R29:R28 e Z = R31:R30
  - Il registro con indirizzo maggiore contiene i bit più significativi
- **A** I/O location address
  - Indirizzo di un registro di I/O, con $0 \leq A \leq 63$
  - Viene anche chiamato *porta* di I/O
- **q** Displacement for direct addressing (6 bit)
  - Valore costante contenuto nell'istruzione, da sommare al valore di un puntatore per ottenere l'indirizzo effettivo di un operando

Indirizzamento (1)

- Esistono diversi meccanismi con cui le istruzioni si procurano gli operandi
  - L'indirizzamento ha un notevole impatto sulla dimensione dei campi dell’istruzione
  - Influenza il tempo di esecuzione, perché ogni meccanismo richiede un tempo diverso per portare il corretto valore alla ALU
- Nelle istruzioni a due operandi, possono essere usati due meccanismi di indirizzamento diversi

Indirizzamento (1)

- Esistono diversi meccanismi con cui le istruzioni si procurano gli operandi
  - L'indirizzamento ha un notevole impatto sulla dimensione dei campi dell’istruzione
  - Influenza il tempo di esecuzione, perché ogni meccanismo richiede un tempo diverso per portare il corretto valore alla ALU
- Nelle istruzioni a due operandi, possono essere usati due meccanismi di indirizzamento diversi

Indirizzamento (2)

- Generalmente, uno degli operandi è un registro di lavoro
- L'istruzione deve inoltre specificare il destinatario, cioè dove mettere il risultato
  - Di norma, in assembly coincide con il primo degli operandi specificato

Indirizzamento (3)

- I principali meccanismi di indirizzamento si possono classificare in grandi gruppi:
  - Indirizzamento **immediato**
  - Indirizzamento **diretto**
  - Indirizzamento **indiretto**
  - Indirizzamento **indiretto con modifica** del puntatore

Indirizzamento immediato

- L’operando è immediatamente disponibile come costante numerica contenuta in un campo dell’istruzione
- Nell’istruzione assembly appare un valore costante
  - ANDI R16,122
    - Esegue un’operazione logica tra un registro interno e la costante 122, che è fornita in modo immediato
  - LDI R23,0xAA
    - Carica in un registro interno la costante esadecimale 0xAA (170), fornita in modo immediato

Indirizzamento diretto

- L’operando è specificato dichiarando direttamente la memoria che lo contiene
- La memoria può essere un registro, una porta o una cella di memoria dati
  - In questo ultimo caso, l’istruzione sarà lenta e ingombrante
    - MOV R1, R15
      - Copia in R1 il contenuto del registro R15
    - STS 31456, R3
      - Memorizza (STorize) R3 nella memoria all’indirizzo specificato direttamente (31456)

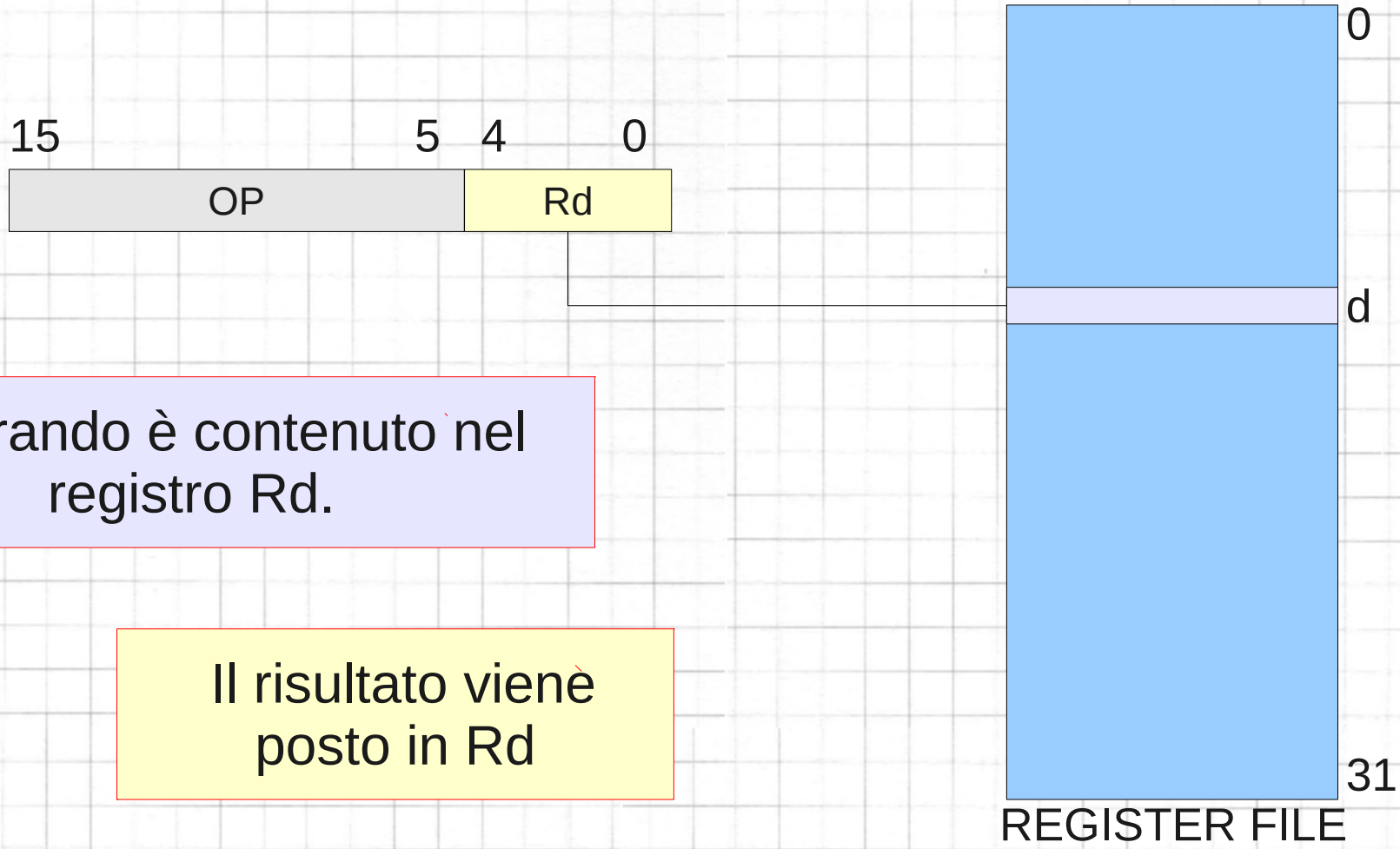
Indirizzamento indiretto

- L’operando, nella memoria dati, è specificato indicando un puntatore che contiene il suo indirizzo
  - Nel nostro esempio il puntatore è un registro da 16 b (cioè una coppia di registri da 8 b)
    - X = R27:R26; Y = R29:R28; Z = R31:R30
  - L’indirizzamento indiretto è efficiente, perché l'indirizzo è già nel register file
    - LD R10, X
      - Copia in R10 il contenuto puntato dalla coppia R27:R26

Indiretto con automodifica

- Istruzioni con indirizzamento indiretto che alterano automaticamente il puntatore
  - Se ripetute, operano su dati diversi!
  - La modifica consiste nella variazione del puntatore di ± 1, prima o dopo l'operazione
    - Si parla di pre-decremento o post-incremento
    - ST Y+, R10
      - Copia R10 dove punta X e poi incrementa X
    - LD R10, -X
      - Decrementa X e poi copia in R10 il contenuto della memoria puntato da X stesso (la coppia R27:R26)

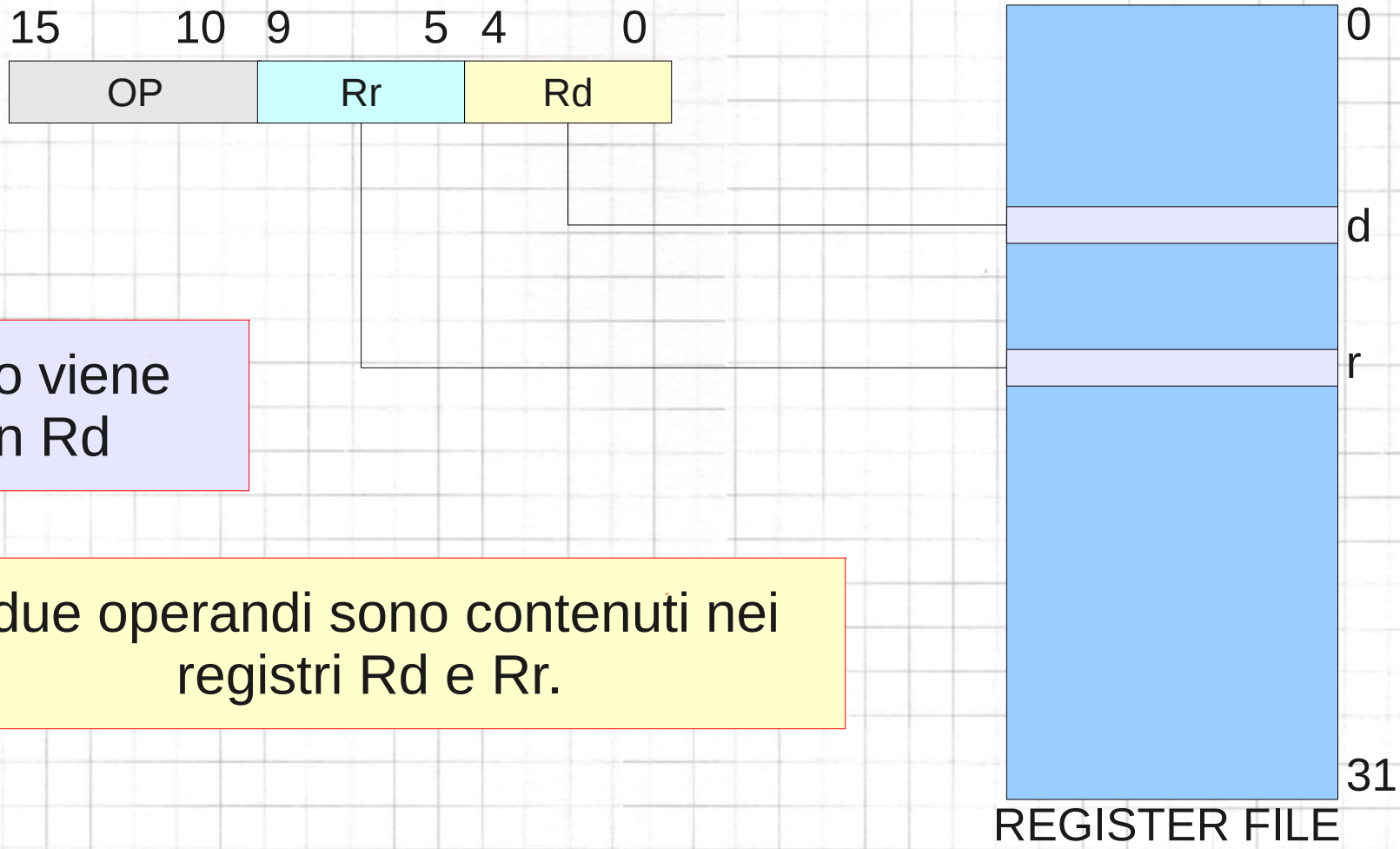
Diretto, singolo registro



L'operando è contenuto nel registro Rd.

Il risultato viene posto in Rd

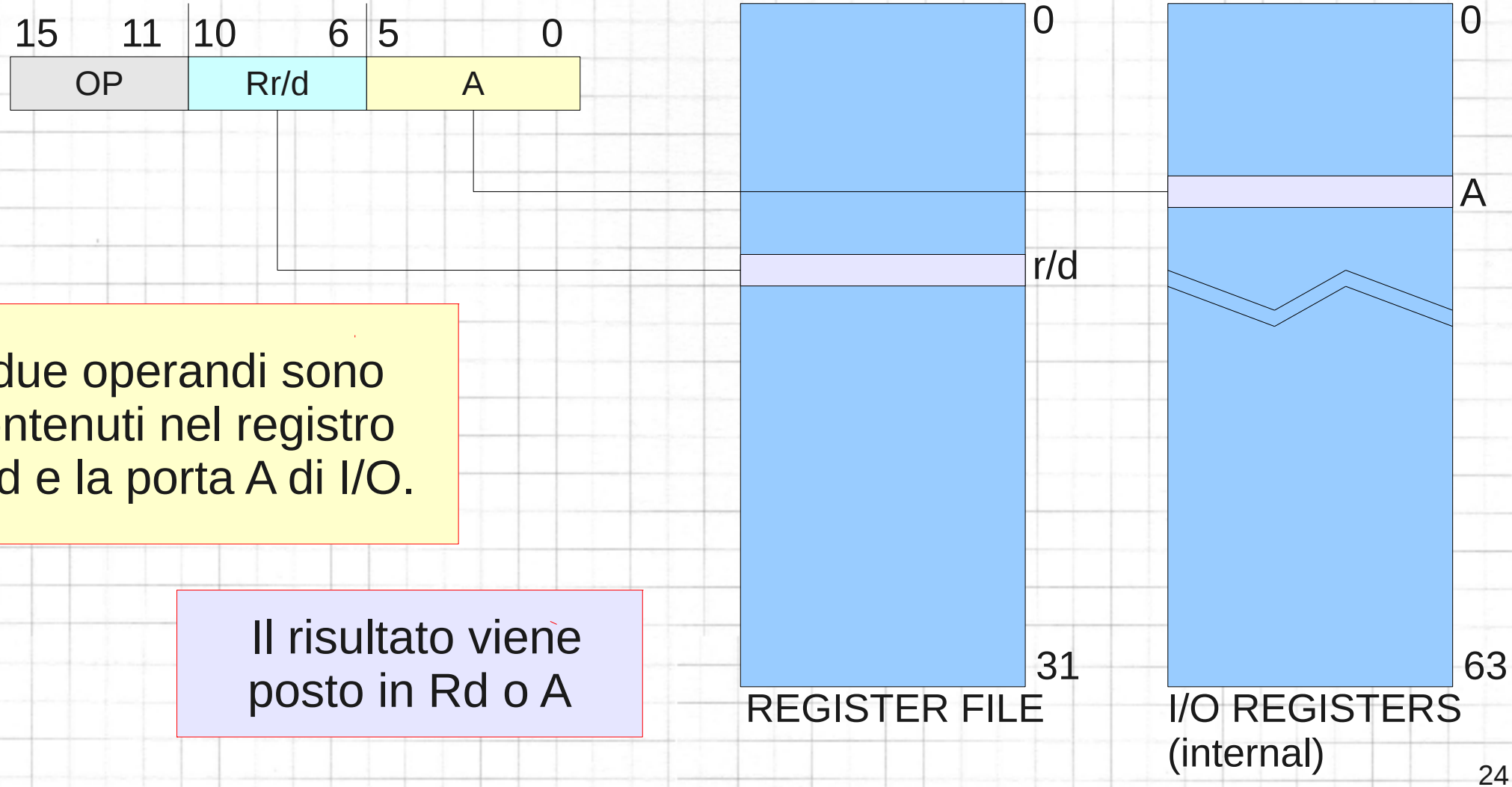
Diretto-diretto,
due registri Rd e Rr



Il risultato viene posto in Rd

I due operandi sono contenuti nei registri Rd e Rr.

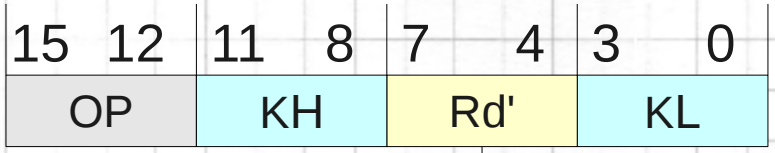
Diretto-diretto, registro Rd/r e porta A

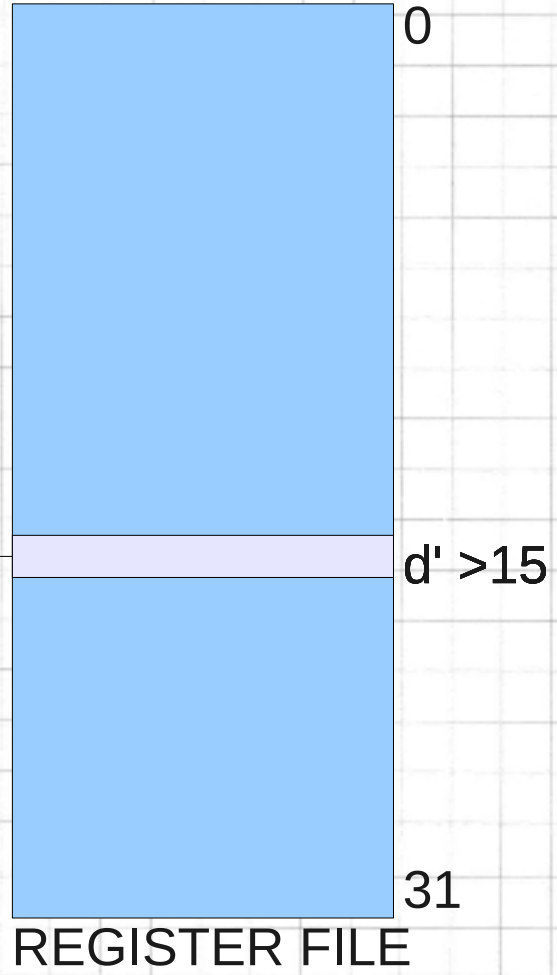


I due operandi sono contenuti nel registro Rr/d e la porta A di I/O.

Il risultato viene posto in Rd o A

Diretto-immediato, registro Rd e costante K





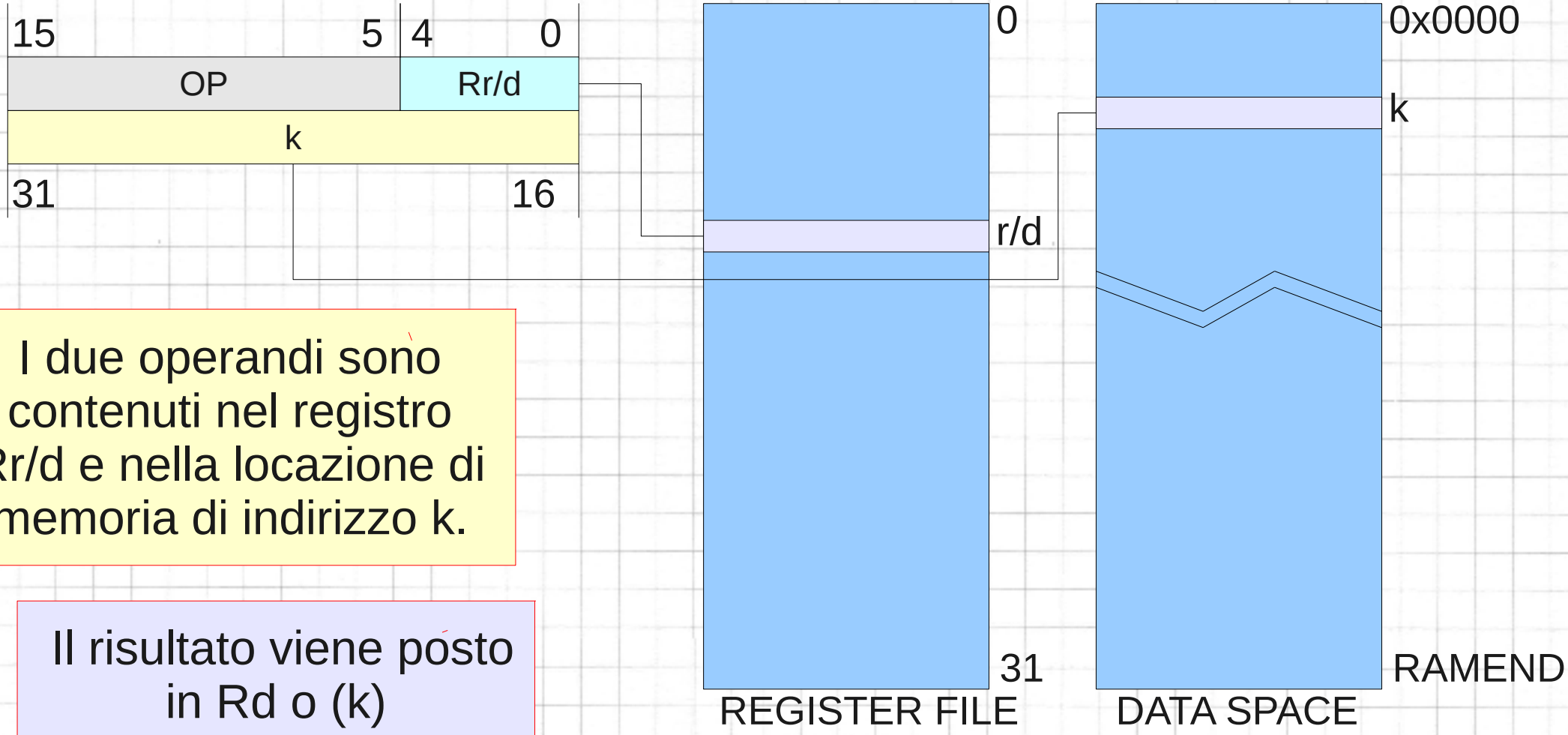
Un operando è costante, l'altro è contenuto in Rd

La costante è spezzata nelle 2 cifre esadecimali

Deve essere d' >15

Il risultato viene posto in Rd

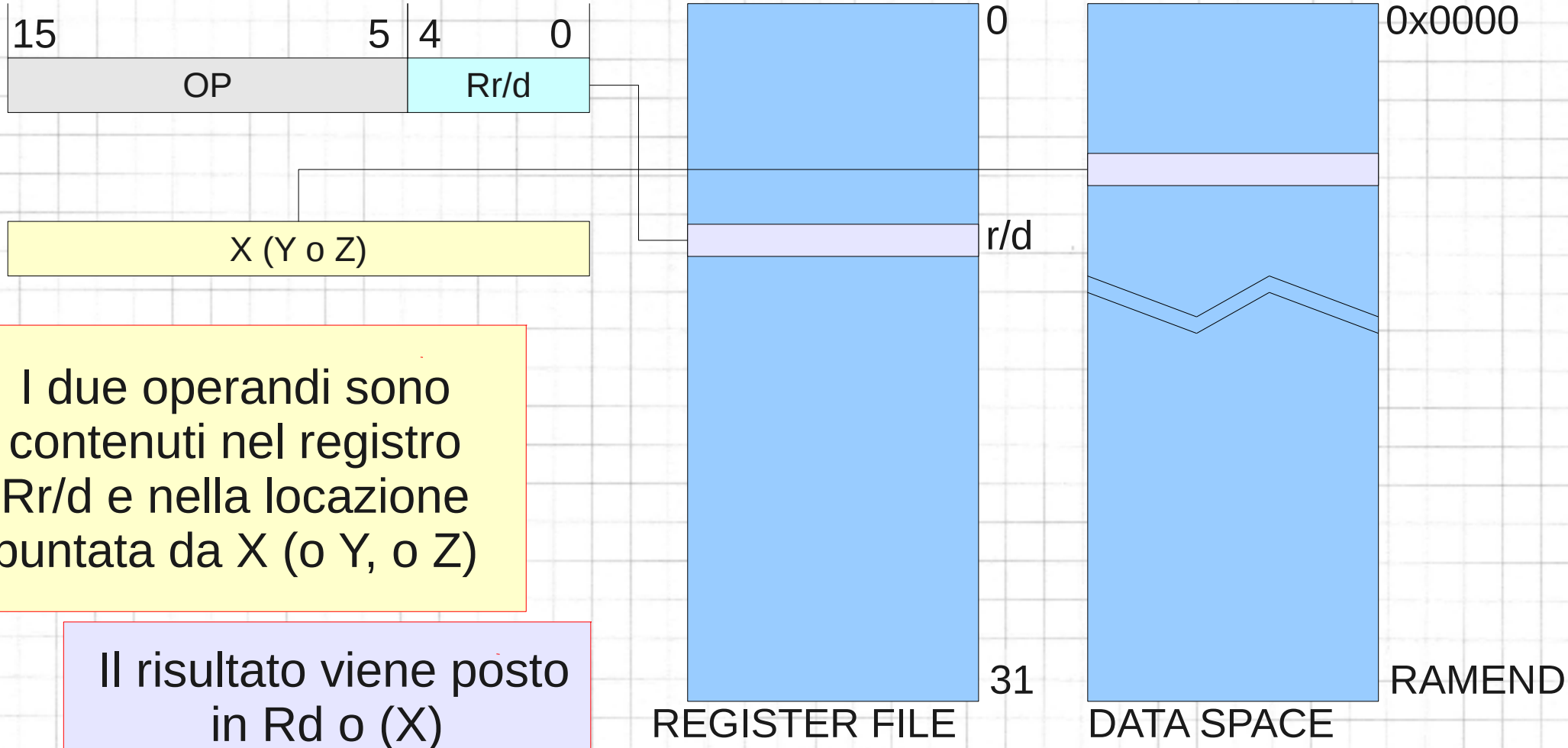
Diretto-diretto, registro Rd/r e memoria



I due operandi sono contenuti nel registro Rr/d e nella locazione di memoria di indirizzo k.

Il risultato viene posto in Rd o (k)

Indiretto-diretto, puntatore e registro Rr/d

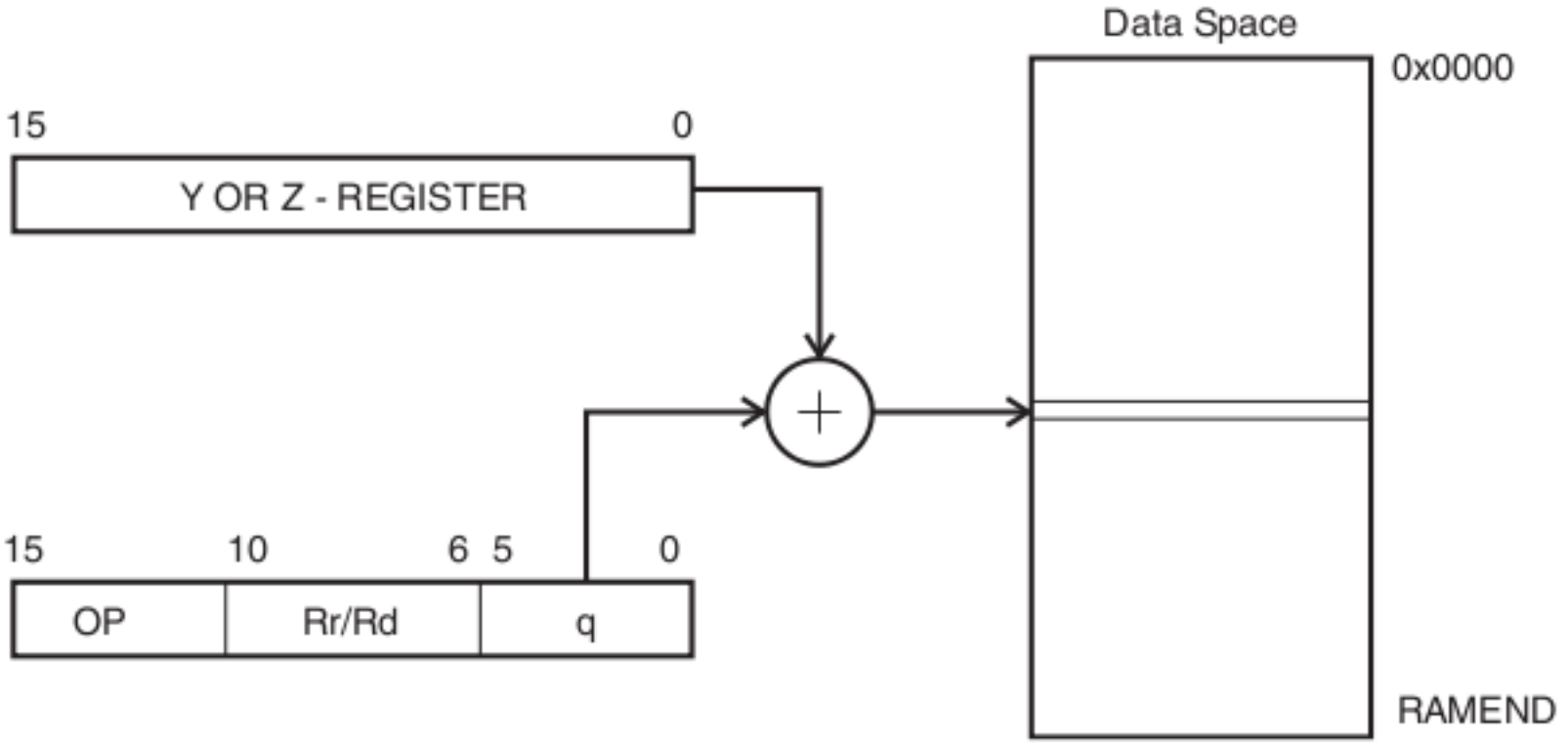


I due operandi sono contenuti nel registro Rr/d e nella locazione puntata da X (o Y, o Z)

Il risultato viene posto in Rd o (X)

Altri indirizzamenti (1)

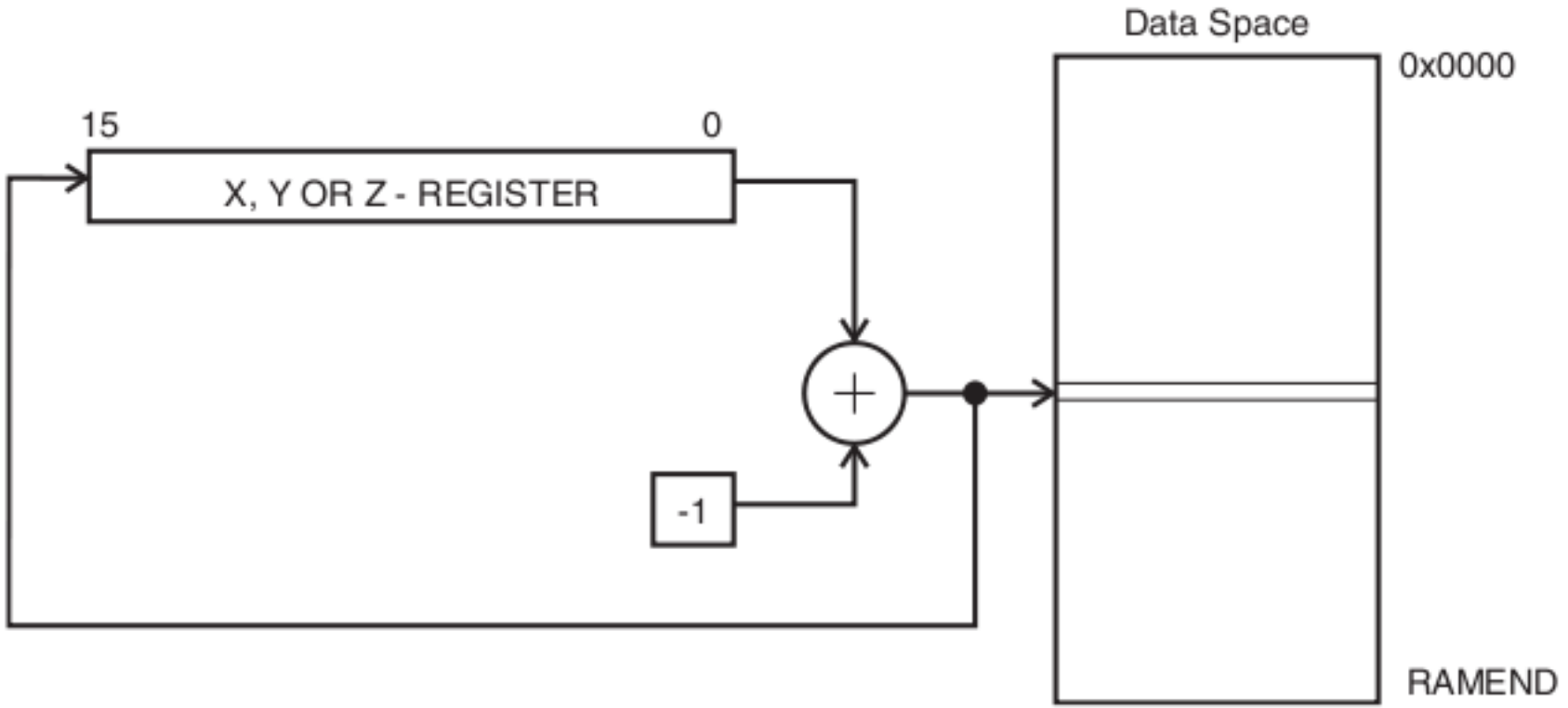
Figure 5. Data Indirect with Displacement



Operand address is the result of the Y- or Z-register contents added to the address contained in 6 bits of the instruction word. Rd/Rr specify the destination or source register.

Altri indirizzamenti (2)

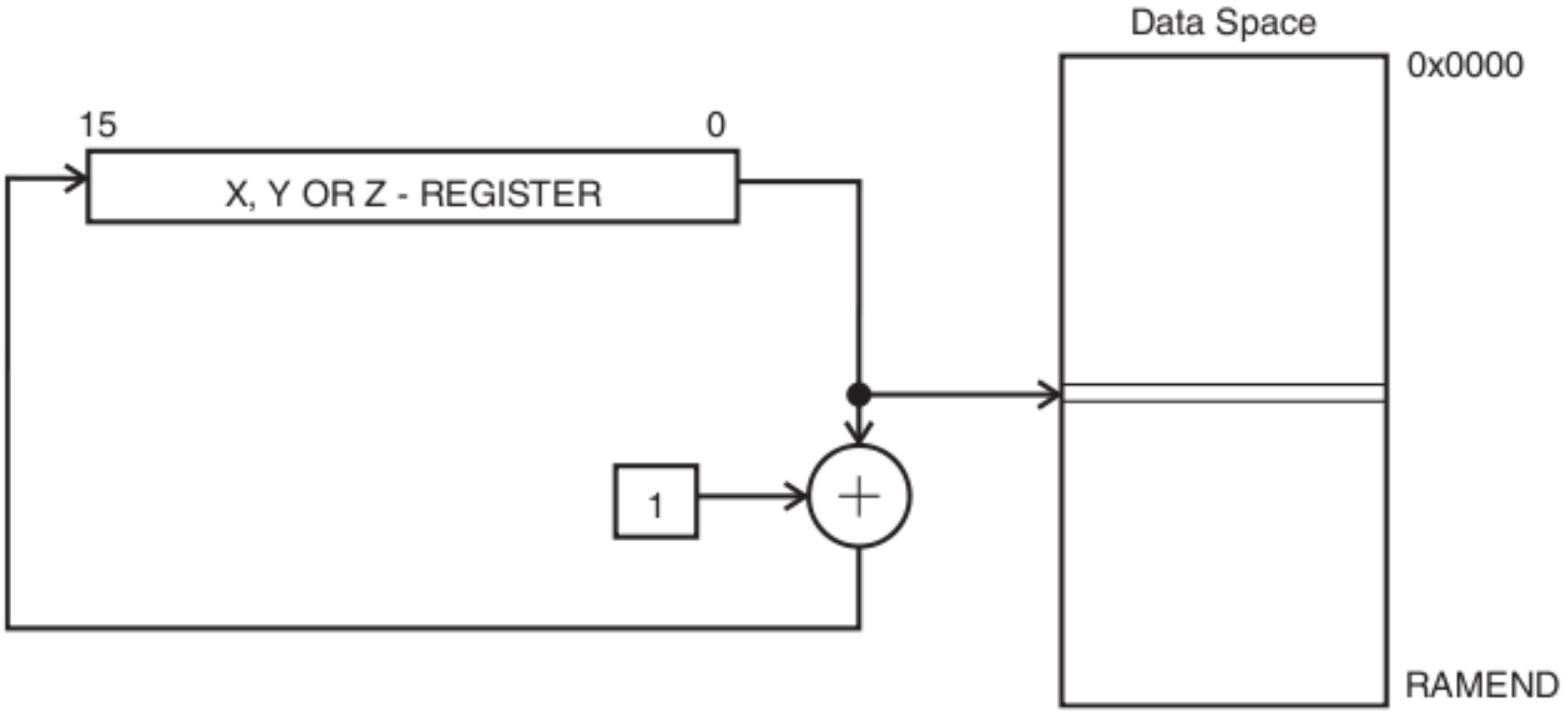
Figure 7. Data Indirect Addressing with Pre-decrement



The X,- Y-, or the Z-register is decremented before the operation.
Operand address is the decremented contents of the X-, Y-, or the Z-register.

Altri indirizzamenti (3)

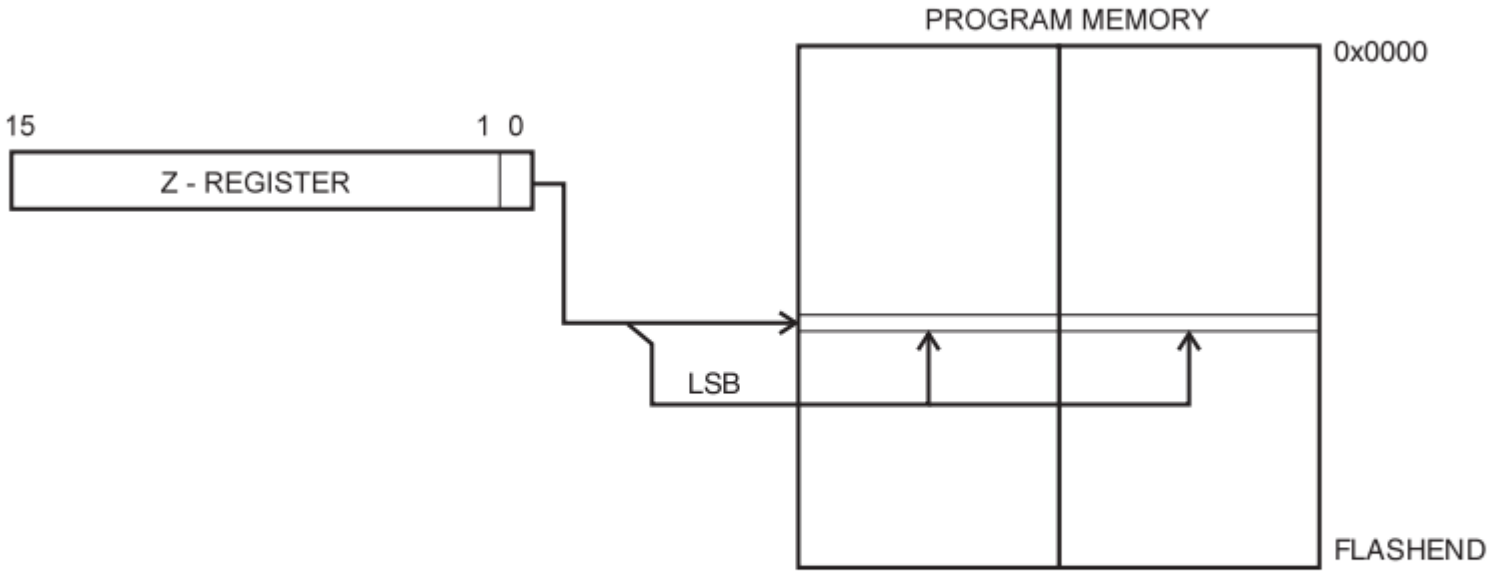
Figure 8. Data Indirect Addressing with Post-increment



The X-, Y-, or the Z-register is incremented after the operation. Operand address is the content of the X-, Y-, or the Z-register prior to incrementing.

Altri indirizzamenti (4)

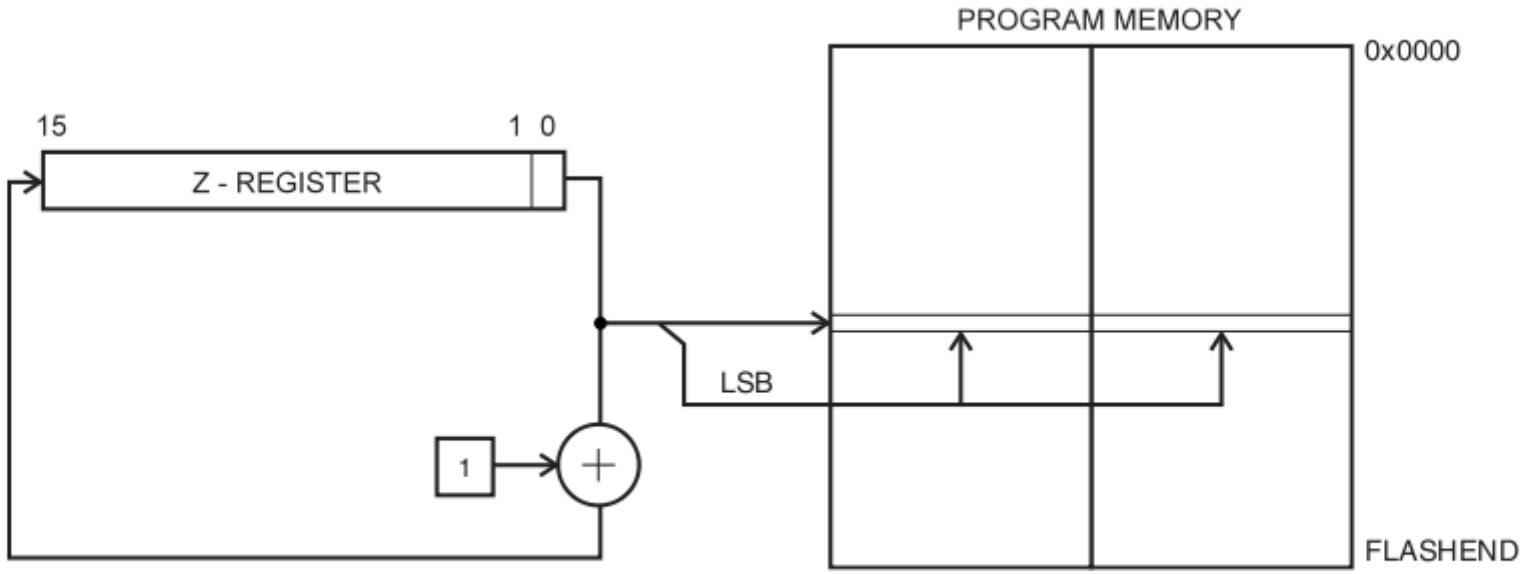
Figure 9. Program Memory Constant Addressing



Constant byte address is specified by the Z-register contents. The 15 MSBs select word address. For LPM, the LSB selects low byte if cleared (LSB = 0) or high byte if set (LSB = 1). For SPM, the LSB should be cleared. If ELPM is used, the RAMPZ Register is used to extend the Z-register.

Altri indirizzamenti (5)

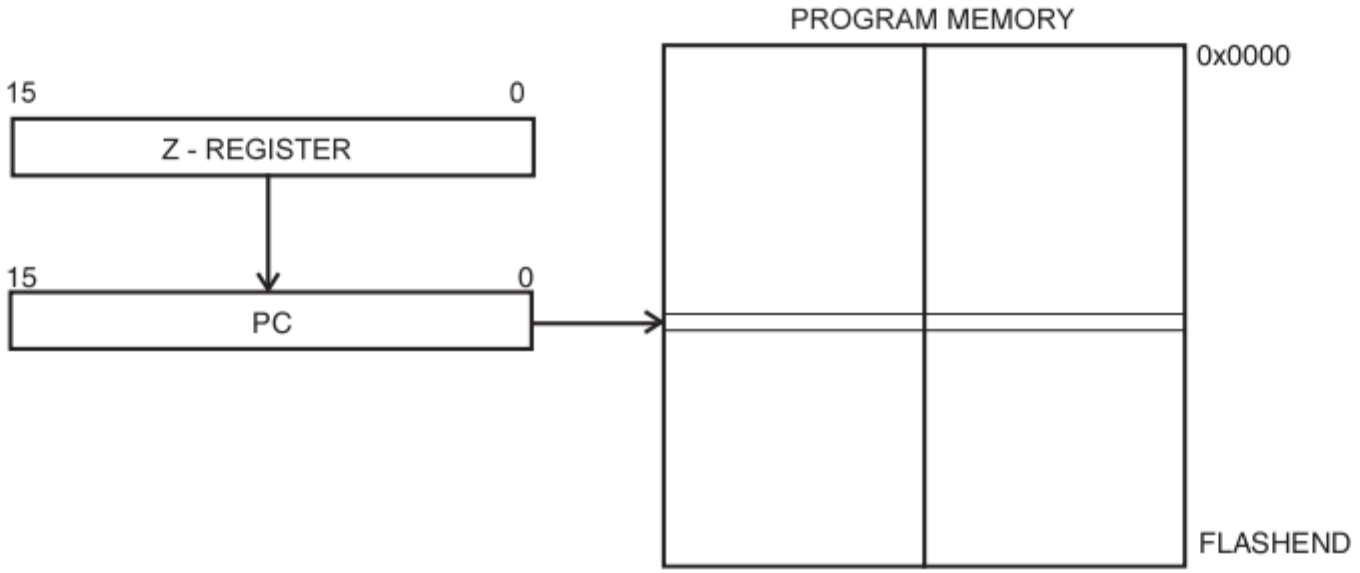
Figure 10. Program Memory Addressing with Post-increment



Constant byte address is specified by the Z-register contents. The 15 MSBs select word address. The LSB selects low b
if cleared (LSB = 0) or high byte if set (LSB = 1). If ELPM Z+ is used, the RAMPZ Register is used to extend the Z-regis

Altri indirizzamenti (6)

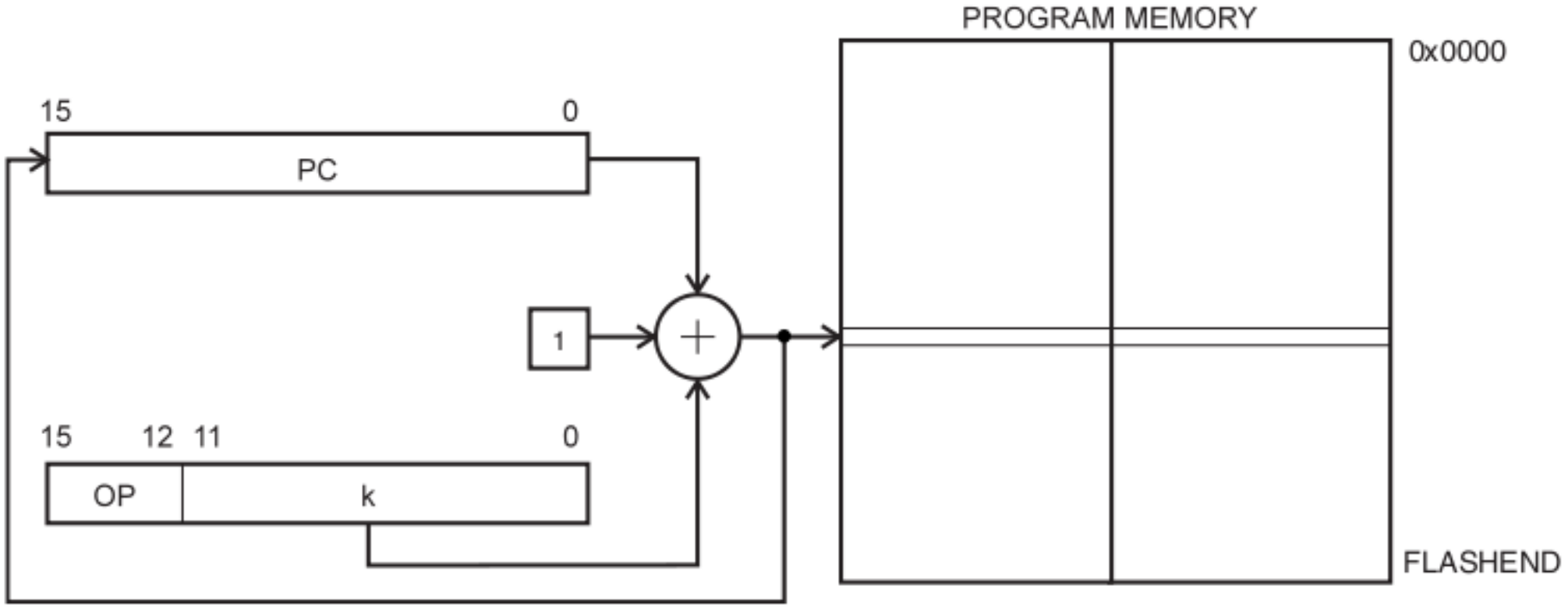
Figure 12. Indirect Program Memory Addressing



Program execution continues at address contained by the Z-register (i.e., the PC is loaded with the contents of the Z-register).

Altri indirizzamenti (7)

Figure 13. Relative Program Memory Addressing



Program execution continues at address PC + k + 1.
The relative address k is from -2048 to 2047.

Categorie di istruzioni (1)

- Operazioni di trasferimento di dati
  - Copiano il valore contenuto in una memoria in memorie dello stesso o di altro tipo
  - Uno dei due operandi è sempre un registro
- Operazioni aritmetiche
  - Nei più semplici microcontrollori sono presenti solo somme, differenze e poco altro
  - Le operazioni più complesse non presenti devono essere eseguite da sottoprogrammi

Categorie di istruzioni (2)

- Operazioni logiche (not, and, or, xor)
  - Operano su singole variabili logiche (1 b)
  - Il registro equivale a 8 variabili indipendenti
    - Le istruzioni agiscono su bit corrispondenti
    - Il risultato viene messo nel bit corrispondente
- Operazioni di manipolazione dei bit
  - Esistono istruzioni che spostano o ruotano tutti i bit di un registro
  - Altre istruzioni intervengono su un singolo bit, lasciando inalterati gli altri

Categorie di istruzioni (3)

- Operazioni di controllo
  - Rientrano in questa categoria tutte le istruzioni che hanno un effetto diretto sul flusso di esecuzione del programma
    - Salti incondizionati o condizionati
    - Chiamata di procedure
  - Oppure un effetto indiretto sul processore
    - Sospensione o blocco del processore
    - Gestione del *watchdog*
    - NOP (No Operation)!

Trasferimento dati (1)

Codice	Operandi	Descrizione	Effetto	Flag	T
MOV	Rd, Rr	Copy Register	Rd←Rr	None	1
MOVW	Rd, Rr	Copy Register Pair	[Rd+1:Rd]←[Rr+1:Rr]	None	1
LDI	Rd', K	Load Immediate	Rd←K	None	1

Codice	Operandi	Descrizione	Effetto	Flag	T
LDS	Rd, k	Load Direct from data space	Rd←(k)	None	2
LD	Rd, X|Y|Z	Load Indirect	Rd←(X)	None	1
LD	Rd, X+|Y+|Z+	Load Indirect and Post-Increment	Rd←(X), X←X+1	None	1
LD	Rd, -X|-Y|-Z	Load Indirect and Pre-Decrement	X←X-1, Rd←(X)	None	2
LDD	Rd,Y+q|Z+q	Load Indirect with Displacement	Rd←(Y+q)	None	2

Trasferimento dati (2)

Codice	Operandi	Descrizione	Effetto	Flag	T
STS	k, Rr	Store Direct to data space	(k)←Rr	None	2
ST	X|Y|Z, Rr	Store Indirect	(X)←Rr	None	1
ST	X+|Y+|Z+, Rr	Store Indirect and Post-Increment	(X)←Rr, X←X+1	None	1
ST	-X|-Y|-Z, Rr	Store Indirect and Pre-Decrement	X←X-1, (X)←Rr	None	2
STD	Y+q|Z+q, Rr	Store Indirect with Displacement	(Y+q)←Rr	None	2

Codice	Operandi	Descrizione	Effetto	Flag	T
IN	Rd, A	In From I/O Location	Rd←I/O(A)	None	1
OUT	A, Rr	Out To I/O Location	I/O(A)←Rr	None	1

Trasferimento dati (3)

Codice	Operandi	Descrizione	Effetto	Flag	T
PUSH	Rr	Push Register on Stack	STACK←Rr, SP←SP-1	None	1
POP	Rd	Pop Register from Stack	SP←SP+1, Rd←STACK	None	2
Codice	Operandi	Descrizione	Effetto	Flag	T
LPM		Load Program Memory	R0←(Z)	None	3
LPM	Rd, Z	Load Program Memory	Rd←(Z)	None	3
LPM	Rd, Z+	Load Program Memory and Post-Increment	Rd←(Z), Z←Z+1	None	3
SPM		Store Program Memory	(Z)←[R1:R0]	None	-
SPM	Z+	Store Program Memory	(Z)←[R1:R0], Z←Z+2	None	-

Trasferimento dati (4)

Codice	Operandi	Descrizione	Effetto	Flag	T
XCH	Z,Rd	Exchange RAM location	Temp←Rd, Rd←(Z), •(Z)←Temp	None	2
LAS	Z,Rd	Load and set RAM location	Temp←Rd, Rd←(Z), •(Z)←Temp∨Rd	None	2
LAC	Z,Rd	Load and clear RAM location	Temp←Rd, Rd←(Z), •(Z)←Temp∧$\overline{Rd}$	None	2
LAT	Z,Rd	Load and toggle RAM location	Temp←Rd, Rd←(Z), •(Z)←Temp⊕Rd	None	2

Aritmetiche (1)

Codice	Operandi	Descrizione	Effetto	Flag	T
ADD	Rd, Rr	Add without Carry	Rd←Rd+Rr	Z,C,N,V,S,H	1
ADC	Rd, Rr	Add with Carry	Rd←Rd+Rr+C	Z,C,N,V,S,H	1
ADIW	Rd[1], K	Add Immediate to Word	[Rd+1:Rd]← [Rd+1:Rd]+K	Z,C,N,V,S	2

[1] Funziona solo con i valori di Rd pari a R24, R26, R28 e R30.

Aritmetiche (2)

Codice	Operandi	Descrizione	Effetto	Flag	T
SUB	Rd, Rr	Subtract without Carry	Rd←Rd-Rr	Z,C,N,V,S,H	1
SBC	Rd, Rr	Subtract with Carry	Rd←Rd-Rr-C	Z,C,N,V,S,H	1
SBIW	Rd[1], K	Subtract Immediate from Word	[Rd+1:Rd]← [Rd+1:Rd]-K	Z,C,N,V,S	2
SUBI	Rd_b, K	Subtract Immediate	Rd←Rd-K	Z,C,N,V,S,H	1
SBCI	Rd_b, K	Subtract Immediate with Carry	Rd←Rd-K-C	Z,C,N,V,S,H	1

[1] Funziona solo con i valori di Rd pari a R24, R26, R28 e R30.

Aritmetiche (3)

Codice	Operandi	Descrizione	Effetto	Flag	T
NEG	Rd	Two's Complement	Rd←0-Rd	Z,C,N,V,S,H	1
INC	Rd	Increment	Rd←Rd+1	Z,N,V,S	1
DEC	Rd	Decrement	Rd←Rd-1	Z,N,V,S	1

Codice	Operandi	Descrizione	Effetto	Flag	T
CLR	Rd	Clear Register	Rd←0	Z,N,V,S	1
SER	Rd	Set Register	Rd←0xFF	None	1
TST	Rd	Test for Zero or Minus	Rd←Rd&Rd	Z,N,V,S	1
CP	Rd, Rr	Compare	Rd-Rr	Z,C,N,V,S,H	1
CPC	Rd, Rr	Compare with Carry	Rd-Rr-C	Z,C,N,V,S,H	1
CPI	Rd_b, K	Compare with Immediate	Rd-K	Z,C,N,V,S,H	1

Aritmetiche (4)

Codice	Operandi	Descrizione	Effetto	Flag	T
MUL	Rd, Rr	Multiply unsigned	R1:R0←Rd*Rr (UU)	Z,C	2
MULS	Rd, Rr	Multiply signed	R1:R0←Rd*Rr (SS)	Z,C	2
MULSU	Rd, Rr	Multiply signed with unsigned	R1:R0←Rd*Rr (SU)	Z,C	2
FMUL	Rd, Rr	Fractional multiply unsigned	R1:R0←Rd*Rr<< 1 (UU)	Z,C	2
FMULS	Rd, Rr	Fractional multiply signed	R1:R0←Rd*Rr <<1 (SS)	Z,C	2
FMULSU	Rd, Rr	Fractional multiply signed with unsigned	R1:R0←Rd*Rr <<1 (SU)	Z,C	2

Funziona solo con i valori di Rd e Rr da R16 a R23.

Logiche

Codice	Operandi	Descrizione	Effetto	Flag	T
AND	Rd, Rr	Logical AND	Rd←Rd∧Rr	Z,N,V,S	1
ANDI	Rd', K d' > 15	Logical AND with Immediate	Rd←Rd∧K	Z,N,V,S	1
CBR	Rd', K d' > 15	Clear Bit(s) in Register	Rd← Rd∧(0xFF-K)	Z,N,V,S	1
OR	Rd, Rr	Logical OR	Rd←Rd∨Rr	Z,N,V,S	1
ORI	Rd', K d' > 15	Logical OR with Immediate	Rd←Rd∨K	Z,N,V,S	1
SBR	Rd', K d' > 15	Set Bit(s) in Register	Rd←Rd∨K	Z,N,V,S	1
EOR	Rd,Rr	Exclusive OR	Rd←Rd⊕Rr	Z,N,V,S	1
COM	Rd	One's Complement	Rd←0xFF-Rd	Z,C,N,V,S	1

Manipolazione di bit (1)

Codice	Operandi	Descrizione	Effetto	Flag	T
LSL	Rd	Logical Shift Left	C←Rd(7), Rd(n+1)←Rd(n) n=6..0, Rd(0)←0	Z,C,N,V,H	1
LSR	Rd	Logical Shift Right	C←Rd(0), Rd(n)←Rd(n+1) n=0..6, Rd(7)←0	Z,C,N,V	1
ASR	Rd	Arithmetic Shift Right	C←Rd(0), Rd(n)←Rd(n+1) n=0..6, Rd(7)←Rd(7)	Z,C,N,V	1

Manipolazione di bit (2)

Codice	Operandi	Descrizione	Effetto	Flag	T
ROL	Rd	Rotate Left Through Carry	$C \leftarrow Rd(7)$, $n=6..0$ $Rd(n+1) \leftarrow Rd(n)$, $Rd(0) \leftarrow Cold^{old}$	Z,C,N,V,H	1
ROR	Rd	Rotate Right Through Carry	$C \leftarrow Rd(0)$, $n=0..6$ $Rd(n) \leftarrow Rd(n+1)$, $Rd(7) \leftarrow C^{old}$	Z,C,N,V	1
SWAP	Rd	Swap Nibbles	$Rd(3..0) \leftrightarrow (7..4)$	None	1

Manipolazione di bit (3)

Codice	Operandi	Descrizione	Effetto	Flag	T
BSET	s	Flag Set	SREG(s)←1	SREG(s)	1
SE✯[1]		Set flag ✯	✯←1	✯	1
BCLR	s	Flag Clear	SREG(s)←0	SREG(s)	1
CL✯		Clear flag ✯	✯←0	✯	1
SBI	A_a, b	Set Bit in I/O Register	I/O(A, b)←1	None	1
CBI	A_a, b	Clear Bit in I/O Register	I/O(A, b)←0	None	1
BST	Rr, b	Bit Store from Register to T	T←Rr(b)	T	1
BLD	Rd, b	Bit load from T to Register	Rd(b)←T	None	1

[1] Il simbolo ✯ è una delle lettere con cui si designano i flag di SREG, cioè C, Z, N, V, S, H, T, I.

Controllo del programma (1)

Codice	Operandi	Descrizione	Effetto	Flag	T
RJMP	k	Relative Jump	PC←PC+k+1	None	2
IJMP		Indirect Jump	PC←Z	None	2
EIJMP		Indirect Jump	PC←Z	None	2
JMP	k	Absolute Jump	PC←k	None	3

Controllo del programma (2)

Codice	Operandi	Descrizione	Effetto	Flag	T
RCALL	k	Relative Call Subroutine	PUSH PC_H PUSH PC_L PC←PC+k+1	None	3
ICALL		Indirect Call	PUSH PC_H, PUSH PC_L, PC←Z	None	3
RET		Subroutine Return	POP PC_H, PC_L	None	4
RETI		Interrupt Return	POP PC_H, PC_L	I[1]	4

[1] Il flag I viene ripristinato al valore precedente l'interruzione.

Controllo del programma (3)

Codice	Operandi	Descrizione	Effetto	Flag	T
CPSE	Rd, Rr	Compare, Skip if Equal	if (Rd=Rr) PC←PC+2 or 3[1]	None	1[2]
SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b)=0) PC←PC+2 or 3[1]	None	1[2]
SBRS	Rr, b	Skip if Bit in Register Set	if (Rr(b)=1) PC←PC+2 or 3[1]	None	1[2]
SBIC	A, b	Skip if Bit in I/O Register Cleared	if (I/O(A,b)=0) PC←PC+2 or 3[1]	None	2[2]
SBIS	A, b	Skip if Bit in I/O Register Set	If (I/O(A,b)=1) PC←PC+2 or 3[1]	None	2[2]

[1] Il valore del salto è 3 se l'istruzione da saltare è lunga 2 parole.

[2] Tempo di esecuzione in assenza di salto. Se c'è salto occorrono +1T se l'istruzione da saltare è lunga 1 parola oppure +2T se è lunga 2 parole.

Controllo del programma (4)

Codice	Operandi	Descrizione	Effetto	Flag	T
BRBS	s, k	Branch if Status Flag Set	if (SREG(s)=1) PC←PC+k+1	None	1[1]
BRBC	s, k	Branch if Status Flag Cleared	if (SREG(s)=0) PC←PC+k+1	None	1[1]
BRXX[2]	k	Branch if Condition XX is True	if (XX) PC←PC+k+1	None	1[1]

[1] Tempo di esecuzione in assenza di salto. Se c'è salto occorrono 2 T.

[2] Tal posto di XX sostituire le lettere specificate nel seguito.

Condizioni di salto

Flag e Valore	C=1	C=0	Z=1	Z=0	N=1	N=0
Condizione XX	CS, LO	CC, SH	EQ	NE	MI	PL
Significato	Carry Set, Lower	Carry Cleared, Same or Higher	Equal	Not Equal	Minus	Plus

Flag e Valore	S=1	S=0	H=1	H=0	T=1	T=0
Condizione XX	LT	GE	HS	HC	TS	TC
Significato	Less Than	Greater or Equal	Half carry Set	Half carry cleared	T flag Set	T flag Cleared

Flag e Valore	V=1	V=0	I=1	I=0
Condizione XX	VS	VC	IE	ID
Significato	Overflow Set	Overflow Cleared	Interrupt enabled	Interrupt disabled

Controllo del programma (5)

Codice	Operandi	Descrizione	Effetto	Flag	T
BREAK		Break	(see specific description)	None	1
NOP		No operation	(see specific description)	None	1
SLEEP		Sleep	(see specific description)	None	1
WDR		Watchdog Reset	(see specific description)	None	1