

HCS08

MANUALE ISTRUZIONI

ASSEMBLY



Document Revision : 3.40 20.Dicembre.2015

Author : Rech Marzio – **IW3FBA**

marzio.rech@negrelliforcellini.gov.it : **IZ3JCN**

ABBREVIAZIONI DI TERMINI USATI

MCU : Microcontrollore

REGISTRO DEL MCU : Memoria utilizzata dal MCU per manipolare i Dati

PC : Program Counter : Contatore dell' indirizzo dell' istruzione corrente del programma da eseguire

REGISTRO DI MEMORIA: Memoria riservata al MCU per configurare i suoi moduli interni da programma

RAM : Memoria di scrittura e lettura Dati. Trattiene i Dati solo fino a che il MCU è alimentato

STACK : Catasta di DATI : Zona di memoria RAM utilizzata dal **MCU** per salvare temporaneamente i DATI.

FLASH : Memoria dove risiede il Programma del MCU. I Dati vengono conservati anche se manca l'alimentazione

A : Accumulatore : Registro del **MCU** usato per le operazioni aritmetiche e per muovere i dati

X : Registro Indice **X** : Registro del **MCU** usato come indice in molte istruzioni e per muovere i dati

H : Registro indice **H** : Registro del **MCU** che assieme a **X** estende l'indice a 16 Bits per muovere i dati

SP: Registro Stack Pointer : Registro a 16 Bits che informa la posizione dell'ultimo dato depositato nello STACK

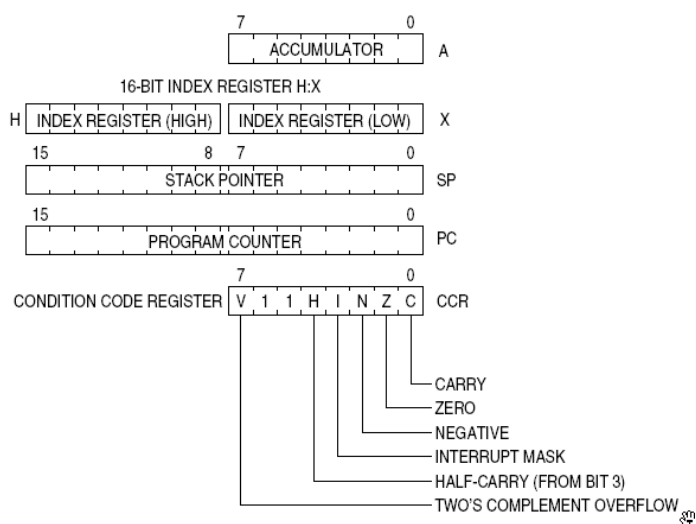
CCR : Registro di Stato : Contiene degli indicatori (FLAG) che informano sullo stato dell'ultima istruzione eseguita.

N : Flag del **CCR** che indica se un'istruzione ha prodotto un risultato negativo (Bit 7 = 1) : %1xxxxxxx

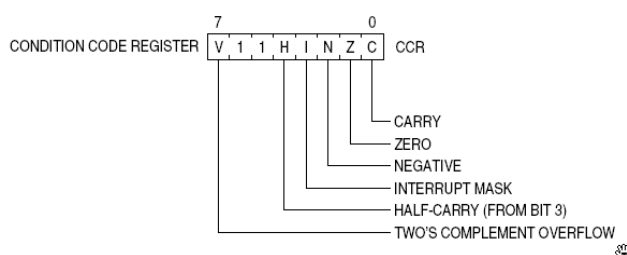
Z : Flag del **CCR** che indica se un'istruzione ha prodotto un risultato nullo : %00000000

C : Flag del **CCR** che indica se un'istruzione ha generato un riporto .

REGISTRI DEL CORE HCS08



REGISTRO DI STATO : CCR Condition Code Register



ISTRUZIONI DI LETTURA SCRITTURA

ISTRUZIONE	SIGNIFICA	COSA FA	FLAG CCR	ESEMPI
LDA	LoaD A	Carica l'accumulatore (A) con un numero (DATO)	N – Z	LDA #45 : $A \leftarrow 45$
				LDA \$9A : $A \leftarrow$ Contenuto della loc. di memoria \$9A
				LDA N_lampi : $A \leftarrow$ Contenuto della loc. di mem lampi
				LDA ,X : $A \leftarrow$ Contenuto della locazione corrispondente a X. Se $X = \$85$ A sarà caricato con il contenuto della loc. \$85
				LDA \$45,X : $A \leftarrow$ Contenuto della locazione corrispondente a $\$45+X$ Se $X = 3$ A sarà caricato con il contenuto della loc. $\$45+3 = \48
STA	STorage A	Deposita il contenuto dell'Accumulatore (A) nella memoria	N – Z	STA \$22 : Deposita A nella loc. di memoria \$22 STA counter : Deposita A nella loc. di mem. 'counter'
LDX	LoaD X	Carica il Registro indice (X) con un numero (DATO)	N – Z	LDX #45 : $X \leftarrow 45$
				LDX \$9A : $X \leftarrow$ Contenuto della loc. di memoria \$9A
				LDX N_lampi : $X \leftarrow$ Contenuto della loc. di mem lampi
LDX	LoaD X	Carica il Registro indice (X) con un numero (DATO)	N – Z	LDX ,X : $X \leftarrow$ Contenuto della locazione corrispondente a X. Se $X = \$85$ X sarà caricato con il contenuto della loc. \$85
				LDX \$45,X : $X \leftarrow$ Contenuto della locazione corrispondente a $\$45+X$
STX	STorage X	Deposita il contenuto del registro indice X nella memoria	N – Z	STX \$22 : Deposita X nella loc. di memoria \$22 STX counter : Deposita X nella loc. di mem. 'counter'
LDHX	LoaD H:X	Carica i Registri H e X con un valore a 16 Bits. H contiene la parte alta (B15..B8) X la parte bassa (B7..B0)	N – Z V = 0	LDHX # \$9B38 H:X $\leftarrow \$9B38$ LDHX \$82 Il contenuto delle Locazioni \$82 e \$83 Sono caricate rispettivamente in H e X
STHX	STorage H:X	Deposita il Contenuto a 16 Bits dei Registri H e X in memoria	N – Z V = 0	STHX \$82 Il Contenuto di H:X sono scritti nelle Locazioni di memoria \$82 \$83

ISTRUZIONI DI MOVIMENTO / COPIA DATO				
ISTRUZIONE	SIGNIFICA	COSA FA	FLAG CCR	ESEMPI
MOV	MOVE	Muove / Copia un valore (Dato) nella memoria	N - Z	MOV #65,\$05 : \leftarrow Mette il numero 65 nella loc. di memoria o registro \$05.
				MOV \$65,\$90 : \leftarrow Copia il contenuto della loc. di memoria \$65 nella loc. \$90
				MOV origine,destin \leftarrow muove origine in destin
TAX	Transfer A to X	Copia il Valore di A in X	-	TAX : $X \leftarrow A$
TXA	Transfer X to A	Copia il Valore di X in A	-	TXA : $A \leftarrow X$
TAP	Transfer A to CCR	Copia i Bits di A nel Registro CCR	N-Z-C V-H-I	TAP : $CCR \leftarrow A$
TPA	Transfer CCR to A	Copia i Bits del CCR in A	-	TPA : $A \leftarrow CCR$
TSX	Transfer SP to X	Copia il Valore del Registro Stack Pointer in X	-	TSX : $X \leftarrow SP$
TXS	Transfer X to SP	Copia il Valore di X nello Stack Pointer	-	TXA : $SP \leftarrow X$
NSA	Nibble Swap Accumulator	Scambia i 4 Bits di destra dell' Accumulatore con i 4 Bits di sinistra	-	NSA : Se $A = \%00101101$ $A \leftarrow \%11010010$

ISTRUZIONI DI AZZERAMENTO/SETTAGGIO REGISTRI E MEMORIA				
ISTRUZIONE	SIGNIFICA	COSA FA	FLAG CCR	ESEMPI
CLRA	CleAR A	Azzera l'Accumulatore	Z = 1 N = 0	CLRA $A : \leftarrow 0$
CLR	CleAR	Azzera una locazione di memoria In pagina '0' (00..\$FF)	Z = 1 N = 0	clr contatore $contatore \leftarrow 0$
CLR X	CleAR X	Azzera il registro indice X	Z = 1 N = 0	CLR X $X : \leftarrow 0$
CLR H	CleAR H	Azzera il registro indice H	Z = 1 N = 0	CLR H $H : \leftarrow 0$
SEC	Setta C	Setta (porta a 1) il Bit di CARRY Nel CCR	C=1	SEC $C \leftarrow 1$
CLC	Clear C	Azzera (porta a 0) il Bit di CARRY Nel CCR	C=0	CLC $C \leftarrow 0$

ISTRUZIONI DI INCREMENTO - DECREMENTO				
ISTRUZIONE	SIGNIFICA	COSA FA	FLAG CCR	ESEMPI
INCA	INC rement A	Incrementa di 1 il valore di A	V-N-Z	INCA : $A \leftarrow A+1$
INCX	INC rement X	Incrementa di 1 il valore di X	V-N-Z	INCX : $X \leftarrow X+1$
INC	INC rement	Incrementa di 1 il valore della memoria RAM di pagina 0 \$00..\$FF	V-N-Z	INC \$65 : contenuto di \$65 \leftarrow contenuto di \$65 +1 INC conteggio : conteggio \leftarrow conteggio + 1
DECA	DEC rement A	Decrementa di 1 il valore di A	V-N-Z	DECA : $A \leftarrow A-1$
DECX	DEC rement X	Decrementa di 1 il valore di X	V-N-Z	DECX : $X \leftarrow X-1$
DEC	DEC rement	Decrementa di 1 il valore della memoria RAM di pagina 0 \$00..\$FF	V-N-Z	DEC \$65 : contenuto di \$65 \leftarrow contenuto di \$65 -1 DEC conteggio : conteggio \leftarrow conteggio - 1
AIX	Add Immediate X	Addiziona in Complemento a 2 un valore al registro a 16 Bits H:X nel campo -128 ...+127	-	AIX #105 : Addiziona ad H:X il numero 105 AIX #-68 : Addiziona ad H:X il numero -68
AIS	Add Immediate SP	Addiziona in Complemento a 2 un valore al registro a 16 Bits SP nel campo -128 ...+127	-	AIX #-15 : Sposta il Puntatore SP dello STACK di 15 posizioni più in basso per il salvataggio temporaneo di dati AIS #15 : Sposta il Puntatore SP dello STACK di 15 posizioni più in alto per liberare 15 locazioni usate per il salvataggio di dati e non più usate.

ISTRUZIONI DI COMPARAZIONE				
ISTRUZIONE	SIGNIFICA	COSA FA	FLAG CCR	ESEMPI
CMP	CoMPare	Compara A con un valore dato o con un valore contenuto in una locazione di memoria	N-Z-C	CMP #37 : Compara A con il numero decimale 37 CMP #\$8B : Compara A con il numero esadecimale 8B CMP #%01010011 : Compara A con il numero binario CMP \$8B : Compara A con il contenuto della loc \$8B CMP conteggio : Compara A con il contenuto della loc conteggio
CPX	CoMpare X	Compara X con un valore dato o con un valore contenuto in una locazione di memoria	N-Z-C	CPX #37 : Compara X con il numero decimale 37 CPX #\$8B : Compara X con il numero esadecimale 8B CPX #%01010011 : Compara X con il numero binario CPX \$8B : Compara X con il contenuto della loc \$8B CPX conteggio : Compara X con il contenuto della loc conteggio
CPHX	CoMpare H:X	Compara il numero a 16 Bits contenuto H:X con un valore dato A 16 Bits o con un valore contenuto in due locazione di memoria adiacenti	N-Z-C V	CPHX #47832 : Compara H:X con 47832 CPHX \$B8 : Compara H:X con il contenuto A 16 Bits delle locazioni di Memoria \$B8:\$B9

ISTRUZIONI DI SALTO INCONDIZIONATO E CHIAMATA SUBROUTINE					
ISTRUZIONE	SIGNIFICA	COSA FA	FLAG	ESEMPI	
JMP	JuMP	Salta all'indirizzo o etichetta specificata. E' un salto incondizionato	-	jmp \$b7fa jmp task	Salta alla locazione di memoria \$b7fa Salta alla locazione 'task'
BRA	BRANCH ALWAYS	Dirama sempre all' indirizzo o etichetta specificata. E' una diramazione incondizionata. Viene usata al posto di JMP se la posizione in cui dirama ha un offset fra -128 e + 127. Impiega solo 2 bytes	-	bra main	Va all'indirizzo 'main'
BSR	Branch to SubRoutine	Va ad eseguire il codice specificato dall'indirizzo o etichetta specificato. Al ritorno, determinato dall'istruzione RTS, prosegue per l'istruzione successiva. E' come 'JSR' ma impiega solo 2 Bytes. Viene usata al posto di JSR se la posizione in cui dirama ha un offset fra -128 e + 127.	-	bsr \$2A3B bsr delay	Esegue la Subroutine Locata all'indirizzo \$2ABE Esegue la subroutine 'delay'
JSR	Jump to SubRoutine	Va ad eseguire il codice specificato dall'indirizzo o etichetta specificato. Al ritorno, determinato dall'istruzione RTS, prosegue per l'istruzione successiva	-	jsr \$2A3B Jsr delay	Esegue la Subroutine Locata all'indirizzo \$2ABE Esegue la subroutine 'delay'
RTS	ReTurn From Subroutine	Ultima istruzione da inserire in uno Spezzone di programma se questa viene definita una SubRoutine	-	nop rts	Ritorna all'istruzione successiva alla linea JSR del chiamante.

ISTRUZIONI PERDITEMPO					
ISTRUZIONE	SIGNIFICA	COSA FA	FLAG	ESEMPI	
BRN	BRANCH NEVER	Non Dirama mai, va sempre dritto (Serve per perdere tempo 2 cicli Macchina ...) E' come un NOP ma con 2 Bytes	-	brn loop here nop	Non va mai a 'loop' ma procede sempre per 'here' ... non fa nulla !
NOP	No OPerate	Non fa niente ! Fa perdere solo il tempo di un ciclo macchina con 1 Byte	- nop non fa nulla !

ISTRUZIONI DI SALTO CONDIZIONATO - DIRAMAZIONE					
ISTRUZIONE	SIGNIFICA	COSA FA	FLAG	ESEMPI	
BEQ	BRANCH If EQUAL	Dirama ad un indirizzo o etichetta di programma se il FLAG Z del CCR = 1 Se eseguita dopo CMP- CPX - CPHX Dirama se il dato è uguale	-	loop dec conteggio beq there there nop	Va a 'loop' se il contenuto della loc. 'conteggio' è uguale a 0 altrimenti prosegue per 'there'
BNE	BRANCH If NOT EQUAL	Dirama ad un indirizzo o etichetta di programma se il FLAG Z del CCR = 0 Se eseguita dopo CMP- CPX - CPHX Dirama se il dato NON è uguale	-	loop dec conteggio bne there there nop	Va a 'loop' se il contenuto della loc. 'conteggio' è diverso da 0. altrimenti prosegue per 'there'
				lda delay cmp conteggio bne there here nop there nop	Va a 'there' se il contenuto della loc. 'delay' è ≠ dal contenuto della loc. 'conteggio' altrimenti prosegue per 'here'
BPL	BRANCH If PLUS	Dirama ad un indirizzo o etichetta di programma se il FLAG N del CCR = 0	-	loop dec conteggio bpl there there nop	Va a 'loop' se il contenuto della loc. 'conteggio' è positivo (B7 = 0) altrimenti prosegue per 'there'
BMI	BRANCH If MINUS	Dirama ad un indirizzo o etichetta di programma se il FLAG N del CCR = 1	-	loop dec conteggio bmi there there nop	Va a 'loop' se il contenuto della loc. 'conteggio' è negativo (B7 = 1) altrimenti prosegue per 'there'
BHI	BRANCH If HIGHER	Se eseguita dopo CMP- CPX - CPHX Dirama ad un indirizzo o etichetta di programma se il valore in A o X o HX È maggiore del valore indicato o del contenuto della loc. indicata	-	lda delay cmp #149 bhi there here nop there nop	Va a 'there' se il contenuto della loc. 'delay' è > 149 altrimenti prosegue per 'here'
BHS	BRANCH If HIGHER OR SAME	Se eseguita dopo CMP- CPX - CPHX Dirama ad un indirizzo o etichetta di programma se il valore in A o X o HX È maggiore o uguale del valore indicato o del contenuto della loc. indicata	-	lda delay cmp conteggio bhs there here nop there nop	Va a 'there' se il contenuto della loc. 'delay' è ≥ del contenuto della loc. 'conteggio' altrimenti prosegue per 'here'
BLO	BRANCH If LOWER	Se eseguita dopo CMP- CPX - CPHX Dirama ad un indirizzo o etichetta di programma se il valore in A o X o HX È minore del valore indicato o del contenuto della loc. indicata	-	lda delay cmp conteggio blo there here nop there nop	Va a 'there' se il contenuto della loc. 'delay' è < del contenuto della loc. 'conteggio' altrimenti prosegue per 'here'
BLS	BRANCH If LOWER OR SAME	Se eseguita dopo CMP- CPX - CPHX Dirama ad un indirizzo o etichetta di programma se il valore in A o X o HX È minore o uguale del valore indicato o del contenuto della loc. indicata	-	lda delay cmp conteggio bls there here nop there nop	Va a 'there' se il contenuto della loc. 'delay' è ≤ del contenuto della loc. 'conteggio' altrimenti prosegue per 'here'

ISTRUZIONI DI SALTO CONDIZIONATO – DIRAMAZIONE ‘continuazione’					
ISTRUZIONE	SIGNIFICA	COSA FA	FLAG	ESEMPI	
BGE	BRANCH If GREATER than or Equal to	Comparazione con SEGNO Se BGE è eseguita immediatamente dopo l’esecuzione di una istruzione di comparazione o sottrazione , la diramazione avviene se e solo se il complemento a 2 del numero rappresentato dall’ appropriato registro interno (A,X o H:X) era più grande di o uguale al complemento a 2 del numero rappresentato in memoria.	-	CMP #-17 BGE star CPHX #-5860 BGE star CPX #78 BGE there	Compara A con -17 Va a ‘star’ se A è uguale o maggiore di -17 Compara H:X con -5860 Dirama a ‘star’ se H:X è uguale o maggiore di -5860 Compara X con 78 Dirama a there se X >= 78
BGT	BRANCH If GREATER than	Comparazione con SEGNO Se BGT è eseguita immediatamente dopo l’esecuzione di una istruzione di comparazione o sottrazione , la diramazione avviene se e solo se il complemento a 2 del numero rappresentato dall’ appropriato registro interno (A,X o H:X) era più grande del complemento a 2 del numero rappresentato in memoria.	-	CMP #-17 BGT star CPHX #-5860 BGT star CPX #78 BGT there	Compara A con -17 Va a ‘star’ se A è maggiore di -17 Compara H:X con -5860 Dirama a ‘star’ se H:X è maggiore di -5860 Compara X con 78 Dirama a there se X > 78
BLE	BRANCH If LESS than or Equal to	Comparazione con SEGNO Se BLE è eseguita immediatamente dopo l’esecuzione di una istruzione di comparazione o sottrazione , la diramazione avviene se e solo se il complemento a 2 del numero rappresentato dall’ appropriato registro interno (A,X o H:X) era più piccola di o uguale al complemento a 2 del numero rappresentato in memoria.	-	CMP #-17 BLE star CPHX #-5860 BLE star CPX #78 BLE there	Compara A con -17 Va a ‘star’ se A è uguale o minore di -17 Compara H:X con -5860 Dirama a ‘star’ se H:X è uguale o minore di -5860 Compara X con 78 Dirama a there se X <= 78
BLT	BRANCH If LESS than	Comparazione con SEGNO Se BLT è eseguita immediatamente dopo l’esecuzione di una istruzione di comparazione o sottrazione , la diramazione avviene se e solo se il complemento a 2 del numero rappresentato dall’ appropriato registro interno (A,X o H:X) era più piccolo del complemento a 2 del numero rappresentato in memoria.	-	CMP #-17 BLT star CPHX #-5860 BLT star CPX #78 BLT there	Compara A con -17 Va a ‘star’ se A è minore di -17 Compara H:X con -5860 Dirama a ‘star’ se H:X è minore di -5860 Compara X con 78 Dirama a there se X < 78

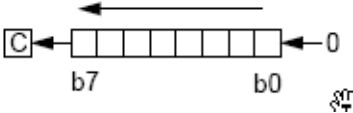
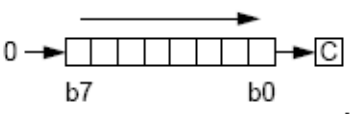
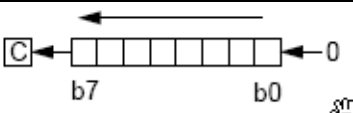
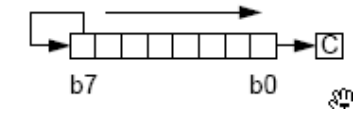
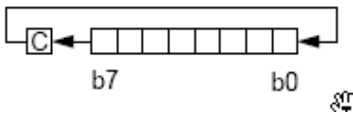
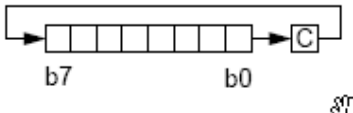
ISTRUZIONI DI SALTO CONDIZIONATO – DIRAMAZIONE ‘continuazione’

<i>ISTRUZIONE</i>	SIGNIFICA	COSA FA	FLAG	ESEMPI	
BCS	BRANCH If CARRY SET	Effettua la diramazione se il Flag C (CARRY) del CCR è settato (1) Viene di solito usata per verificare il riporto/prestito nelle operazioni aritmetiche e nelle istruzioni di rotazione e shift (ROR/ROL/LSL/LSR)	-	bcs there here nop there nop	Va a ‘there’ se il CARRY = 1 altrimenti prosegue per ‘here’
BCC	BRANCH If CARRY CLEAR	Effettua la diramazione se il Flag C (CARRY) del CCR è a 0 Viene di solito usata per verificare il riporto/prestito nelle operazioni aritmetiche e nelle istruzioni di rotazione e shift (ROR/ROL/LSL/LSR)	-	bcc there here nop there nop	Va a ‘there’ se il CARRY = 0 altrimenti prosegue per ‘here’

ISTRUZIONI PER LA MODIFICA DEI SINGOLI BIT					
ISTRUZIONE	SIGNIFICA	COSA FA	FLAG	ESEMPI	
BSET	BitSET	Setta il Bit (0..7) specificato della locazione di memoria specificata	-	bset 4,\$2A bset 2,config	Setta il Bit 4 della Loc di memoria \$2A Setta il Bit 2 della Loc di memoria 'config'
BCLR	BitCLR	Azzer il Bit (0..7) specificato della locazione di memoria specificata	-	bclr 4,\$2A bclr 2,config	Azzer il Bit 4 della Loc di memoria \$2A Azzer il Bit 2 della Loc di memoria 'config'

ISTRUZIONI PER IL TEST DEI SINGOLI BIT					
ISTRUZIONE	SIGNIFICA	COSA FA	FLAG	ESEMPI	
BRSET	BRANCH If Bit SET	Dirama ad un indirizzo o etichetta di programma se il valore del BIT specificato (0..7) è Settato (1)	C	brset 3,option,there here nop there nop	Va a 'there' se il BIT 3 della loc. option = 1 Altrimenti prosegue per 'here'
BRCLR	BRANCH If Bit CLEAR	Dirama ad un indirizzo o etichetta di programma se il valore del BIT specificato (0..7) è azzerato (0)	C	brclr 3,option,there here nop there nop	Va a 'there' se il BIT 3 della loc. option = 0 Altrimenti prosegue per 'here'

ISTRUZIONI DI SCORRIMENTO (SHIFT) e ROTAZIONE

ISTRUZIONE	SIGNIFICA	COSA FA	FLAG	ESEMPI
LSL LSLA LSLX	Logical Shif Left		N-Z-C V	LSL \$85 Muove a sinistra di una posizione tutti i bits della locazione di memoria \$85 Il Bit 7 casca fuori e va a finire nel CARRY a destra nel Bit 0 entra uno Zero.
				LSLA Senza nessun altro argomento effettua lo scorrimento come nell'esempio sopra però riferito all'Accumulatore
				LSLX Stessa cosa per X
LSR LSRA LSRX	Logical Shif Right		Z-C V N=0	LSR \$85 Muove a destra di una posizione tutti i bits della locazione di memoria \$85 Il Bit 0 casca fuori e va a finire nel CARRY a sinistra nel Bit 7 entra uno Zero.
				LSRA Senza nessun altro argomento effettua lo scorrimento come nell'esempio sopra però riferito all'Accumulatore
				LSRX Stessa cosa per X
ASL ASLA ASLX	Aritmetic Shif Left		N-Z-C V	Esattamente la stessa cosa di LSL – LSLA – LSLX La dicitura ASL (Aritmetich Shift Left) vuole ricordare che uno scorrimento a Sinistra di un Bit , equivale A moltiplicare X2 il numero binario.
ASR ASRA ASRX	Aritmetic Shif Right		N-Z-C V	ASR \$85 Muove a destra di una posizione tutti i bits della locazione di memoria \$85 Il Bit 0 casca fuori e va a finire nel CARRY a sinistra il Bit 7 resta inalterato
				Come ASL anche ASR ha un significato aritmetico. Spostando a destra di una posizione i Bits di una locazione, e come effettuare la divisione X 2 del numero. Il Bit 7 rimanendo inalterato, conserva il segno del numero. Se il segno non interessa, usare LSR
ROL ROLA ROLX	ROtate Left		N-Z-C V	ROL \$85 Ruota in senso circolare e verso sinistra i bits della locazione \$85. Il Bit 7 va a finire nel CARRY Il CARRY entra nel Bit 0. ROLA : Stessa cosa per A ROLX : Stessa cosa per X
ROR RORA RORX	ROtate Right			ROR \$85 Ruota in senso circolare e verso destra i bits della locazione \$85. Il Bit 0 va a finire nel CARRY Il CARRY entra nel Bit 7. RORA : Stessa cosa per A RORX : Stessa cosa per X

ISTRUZIONI PER OPERAZIONI ARITMETICHE

ISTRUZIONE	SIGNIFICA	COSA FA	FLAG CCR	ESEMPI
ADD	ADDer	Addiziona il valore indicato all'accumulatore	N-Z-C H-V	<p>lda # 187 $A \leftarrow 187$ add # 15 Aggiunge 15 al valore di 'A'</p> <p>lda #17 $A \leftarrow 17$ add residuo Aggiunge il valore contenuto nella loc di memoria 'residuo' ad 'A'</p> <p>lda residuo $A \leftarrow \text{residuo}$ add #17 Aggiunge il valore 17 ad A.</p>
ADC	ADer with Carry	<p>Addiziona il valore indicato all'accumulatore ed aggiunge il Valore del Bit di CARRY</p> <p>Questa Istruzione serve per tener conto nell'addizione di un eventuale riporto prodotto da una addizione parziale precedente.</p>	N-Z-C H-V	<p>lda # 187 $A \leftarrow 187$; C = 0 adc # 15 Aggiunge 15 al valore di 'A' A <- 202</p> <p>lda # 187 $A \leftarrow 187$; C = 1 adc # 15 Aggiunge 15 al valore di 'A' A <- 203 poiché C=1</p> <p>lda #17 $A \leftarrow 17$ adc residuo Aggiunge il valore contenuto nella loc di memoria 'residuo' ad 'A' + il valore del CARRY</p> <p>lda residuo $A \leftarrow \text{residuo}$ adc #17 Aggiunge il valore 17 ad A. + il valore del CARRY</p>
SUB	SUBtract	Sottrae da A il valore indicato	N-Z-C H-V	<p>lda # 187 $A \leftarrow 187$ sub # 15 Sottrae 15 dal valore di 'A'</p> <p>lda #17 $A \leftarrow 17$ sub residuo Sottrae da A il valore contenuto nella loc di memoria 'residuo'</p> <p>lda residuo $A \leftarrow \text{residuo}$ sub #17 Sottrae il valore 17 da A.</p>
SBC	SUBtract with Carry	<p>Sottrae da A il valore indicato ed il valore del BIT di CARRY.</p> <p>Questa Istruzione serve per tener conto nella sottrazione di un eventuale prestito prodotto da una sottrazione parziale precedente.</p>	N-Z-C H-V	<p>lda # 187 $A \leftarrow 187$ C=0 sbc # 15 Sottrae 15 dal valore di 'A' A <- 172</p> <p>lda # 187 $A \leftarrow 187$ C=1 sbc # 15 Sottrae 15 dal valore di 'A' A <- 171 poiché C=1</p> <p>lda #17 $A \leftarrow 17$ sbc residuo Sottrae da A il valore contenuto nella loc di memoria 'residuo' - il valore del CARRY</p> <p>lda residuo $A \leftarrow \text{residuo}$ sbc #17 Sottrae il valore 17 da A - il CARRY</p>

NEGA	NEG ate A	Effettua il Complemento a '2' del dato contenuto nell'accumulatore Al complemento ad '1' viene sommato '1' Viene usato per trasformare un numero positivo in negativo.	N-Z-C-V	lda #35 A ← 35 nega A ← -35
NEG	NEG ate	Effettua il Complemento a '2' del dato contenuto nella loc. di memoria specificata	N-Z-C-V	neg valore Se valore = 35 valore ← -35
MUL	MUL tiplication	Effettua la moltiplicazione a 8 Bit fra il Registro X e l'Accumulatore. Il risultato a 16 Bits è contenuto in X:A (X Bits 15..8 A: Bits 7...0)	C = 0	Se X = 58 A = 250 MUL X:A ← 14500
DIV	DIV ide	Effettua la divisione fra un numero a 16 Bits contenuto in H:A ed un numero a 8 Bits contenuto in X Il Quoziente è piazzato in A ed il resto è piazzato in H	Z-C	Se H = 18 che significa (18*256) in quanto H contiene i Bits da 15...8 cioè la parte alta del numero a 16 Bits, ed A = 135. Quindi H:X = 18*256+135 = 4743 Se X = 87 : DIV produce 4743:87 = 54 con resto di 45 A = 54 H = 45

ISTRUZIONI SALVATAGGIO/RECUPERO/TRASFERIMENTO DATI NELLO STACK				
ISTRUZIONE	SIGNIFICA	COSA FA	FLAG	ESEMPI
PSHA	PuSHA	Salva l'accumulatore A nello STACK	-	A <- 10 psha lda #\$A7 sta work
			-	pula A <- \$A7
PULA	PULIA	Recupera il valore di A dallo STACK	-	A<- 10
PSHX	PuSHX	Salva il registro indice X nello STACK	-	X <- 10 pshx ldx #\$A7 stx work
			-	pulx X <- \$A7
PULX	PULIX	Recupera il valore di X dallo STACK	-	X<- 1
PSHH	PuSHH	Salva il registro indice H nello STACK	-	H <- 10 pshh ldh #\$A7 sth work
			-	pulh H <- \$A7
PULH	PULIH	Recupera il valore di H dallo STACK	-	H<- 10 pshh pulx Salva H nello STACK Il valore di H viene trasferito in X
NOTA !! : I dati salvati nello STACK devono essere recuperati con la modalit� LIFO LAST IN FIRST OUT (Ultimo dentro -Primo fuori) Es. PSHA - PSHH - PSXH → PULX – PULH - PULA				