MELPS 740 CPU CORE BASIC FUNCTIONS

Each series of the MELPS 740 Family uses the standard MELPS 740 instruction set. The functions of the MELPS 740 CPU core are explained below. The multiply and divide instructions are not available in every microcomputer, and the clock control instructions differ in each microcomputer. For details, refer to the table of machine instruction or the functional explanation of each microcomputer.

CENTRAL PROCESSING UNIT (CPU) INTERNAL REGISTERS

The central processing unit (CPU) has the six registers.

Accumulator (A)

The accumulator is an 8-bit register. Data operations such as data transfer, etc., are executed mainly through the accumulator.

Index register X (X), Index register Y (Y)

Both index register X and index register Y are 8-bit registers. In the index addressing modes, the value of the OPERAND is added to the contents of register X or register Y and specifies the real address.

These index registers also have increment, decrement, comparison, and data transfer functions to allow these registers to take some of the functions of the accumulator.

When the T flag in the processor status register is set to "1", the value contained in index register X becomes the address for the second OPERAND.

Stack pointer (S)

The stack pointer is an 8-bit register used during subroutine calls and interrupts. The stack is used to store the current address data and processor status when branching to subroutines or interrupt routines.

The lower eight bits of the stack address are determined by the contents of the stack pointer. The upper eight bits of the stack address are determined by the Stack Page Selection Bit. If the Stack Page Selection Bit is "0", then the RAM in the zero page is used as the stack area. If the Stack Page Selection Bit is "1", then RAM in page 1 is used as the stack area.

The Stack Page Selection Bit is located in the SFR area in the zero page. Note that the initial value of the Stack Page Selection Bit varies with each microcomputer type. Also some microcomputer types have no Stack Page Selection Bit and the upper eight bits of the stack address are fixed.

The operations of pushing register contents onto the stack and popping them from the stack are shown in Fig. 2.

Program counter (PC)

The program counter is a 16-bit counter consisting of two 8-bit registers PC_H and PC_L. It is used to indicate the address of the next instruction to be executed.

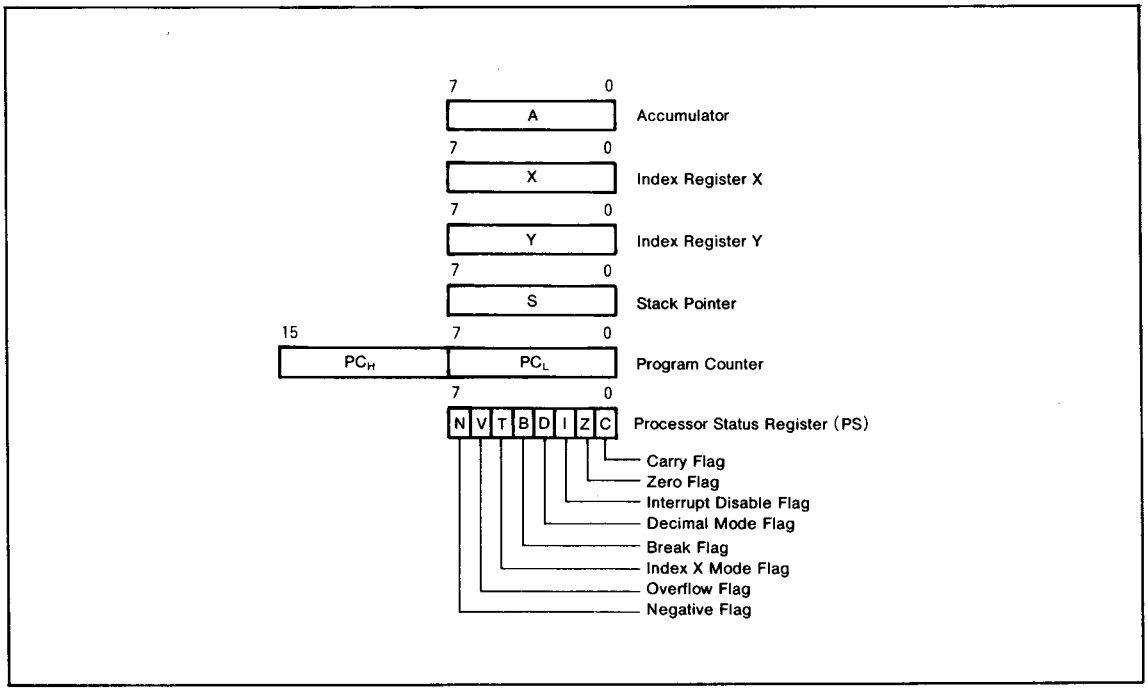
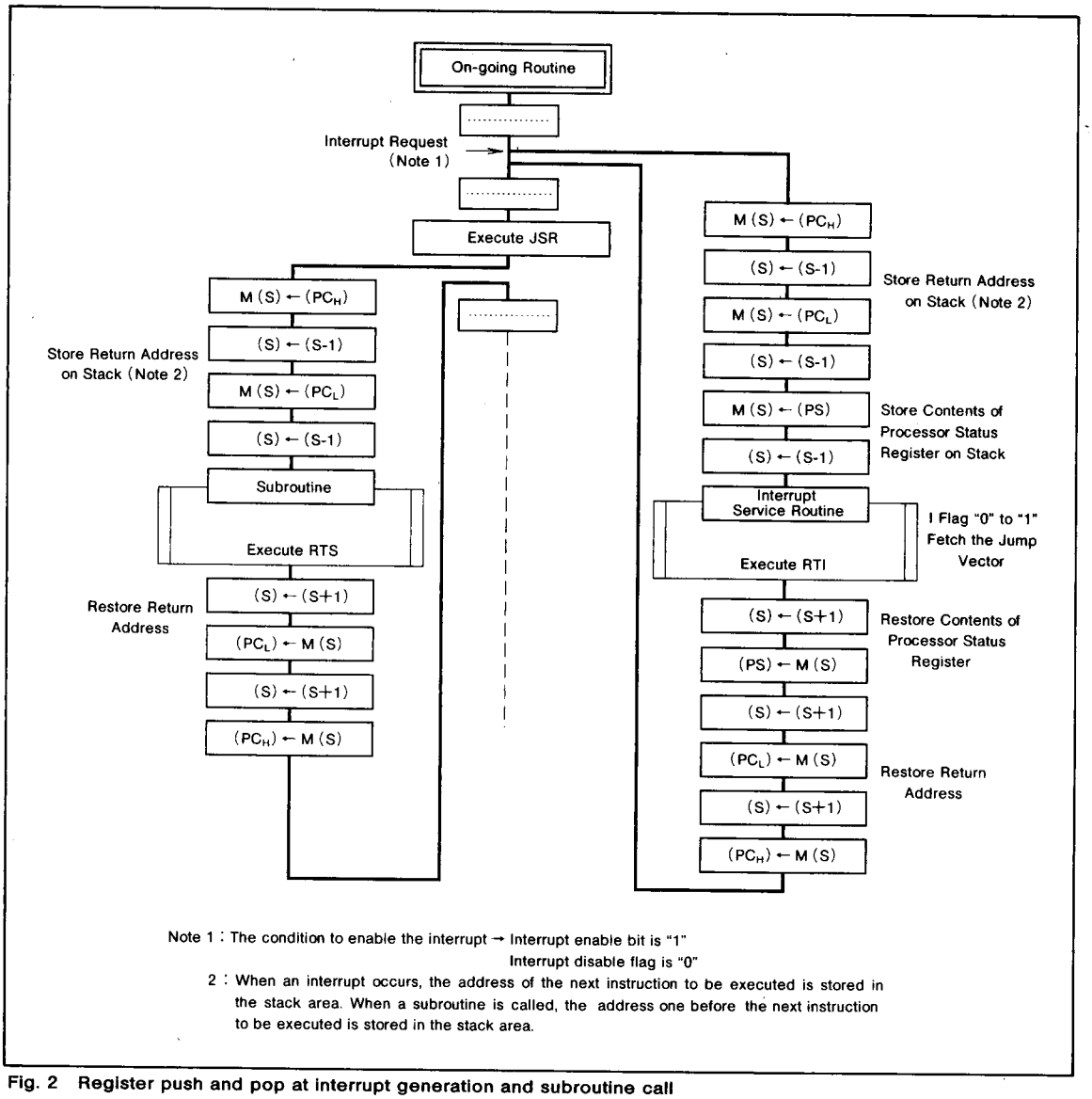


Fig. 1 MELPS 740 CPU register structure



Note 1 : The condition to enable the interrupt → Interrupt enable bit is "1"
Interrupt disable flag is "0"

2 : When an interrupt occurs, the address of the next instruction to be executed is stored in the stack area. When a subroutine is called, the address one before the next instruction to be executed is stored in the stack area.

Fig. 2 Register push and pop at interrupt generation and subroutine call

Table 1. Push and pop instructions of accumulator or processor status register

	Push instruction to stack	Pop instruction from stack
Accumulator	PHA	PLA
Processor status register	PHP	PLP

Processor status register (PS)

The processor status register is an 8-bit register consisting of flags which indicate the status of the processor after an arithmetic operation. Branch operations can be performed by testing the Carry (C) flag, Zero (Z) flag, Overflow (V) flag, or the Negative (N) flag. In decimal mode, the Z, V, N flags are not valid.

After reset, the Interrupt disable (I) flag is set to "1", but all other flags are undefined. Since the Index X mode (T) and Decimal mode (D) flags directly affect arithmetic operations, they should be initialized in the beginning of a program.

(1) Carry flag (C)

The C flag contains a carry or borrow generated by the arithmetic logic unit (ALU) immediately after an arithmetic operation. It can also be changed by a shift or rotate instruction.

(2) Zero flag (Z)

The Z flag is set if the result of an immediate arithmetic operation or a data transfer is "0", and cleared if the result is anything other than "0".

(3) Interrupt disable flag (I)

The I flag disables all interrupts except for the interrupt generated by the BRK instruction.

Interrupts are disabled when the I flag is "1".

When an interrupt occurs, this flag is automatically set to "1" to prevent other interrupts from interfering until the current interrupt is serviced.

(4) Decimal mode flag (D)

The D flag determines whether additions and subtractions are executed in binary or decimal. Binary arithmetic is executed when this flag is "0"; decimal arithmetic is executed when it is "1". Decimal correction is automatic in decimal mode. Only the ADC and SBC instructions can be used for decimal arithmetic.

(5) Break flag (B)

The B flag is used to indicate that the current interrupt was generated by the BRK instruction. The BRK flag in the processor status register is always "0". When the BRK instruction is used to generate an interrupt, the processor status register is pushed onto the stack with the break flag set to "1". The saved processor status is the only place where the break flag is ever set.

(6) Index X mode flag (T)

When the T flag is "0", arithmetic operations are performed between accumulator and memory, e.g. the results of an operation between two memory locations is stored in the accumulator. When the T flag is "1", direct arithmetic operations and direct data transfers are enabled between memory locations, i.e. between memory and memory, memory and I/O, and I/O and I/O. In this case, the result of an arithmetic operation performed on data in memory location 1 and memory location 2 is stored in memory location 1. The address of memory location 1 is specified by index register X, and the address of memory location 2 is specified by normal addressing modes.

(7) Overflow flag (V)

The V flag is used during the addition or subtraction of one byte of signed data. It is set if the result exceeds +127 to −128. When the BIT instruction is executed, bit 6 of the memory location operated on by the BIT instruction is stored in the overflow flag.

(8) Negative flag (N)

The N flag is set if the result of an arithmetic operation or data transfer is negative. When the BIT instruction is executed, bit 7 of the memory location operated on by the BIT instruction is stored in the negative flag.

Table 2. Set and clear instructions of each bit of processor status register

	C flag	Z flag	I flag	D flag	B flag	T flag	V flag	N flag
Set instruction	SEC	—	SEI	SED	—	SET	—	—
Clear instruction	CLC	—	CLI	CLD	—	CLT	CLV	—

ADDRESSING MODE

The MELPS 740 Family has 17 addressing modes and a powerful memory access capability.

When extracting data required for arithmetic and logic operations from memory or when storing the results of such operations in memory, a memory address must be specified. The specification of the memory address is called addressing. The MELPS 740 Family instructions can be classified as 1-byte, 2-byte, and 3-byte instructions. In each case, the first byte is known as the OPCODE which forms the basis of the instruction. A second or third byte is called an OPERAND which affects the addressing. The contents of index registers X and Y can also effect the addressing.

Although there are many addressing modes, there is always a particular memory location specified. What differs is whether the operand, the index register contents, or a combination of both should be used to specify the memory or jump destination. Based on these 3 types of instructions, the range of variation is increased and operation is enhanced by combinations of the bit operation instructions, jump instruction, and arithmetic instructions.

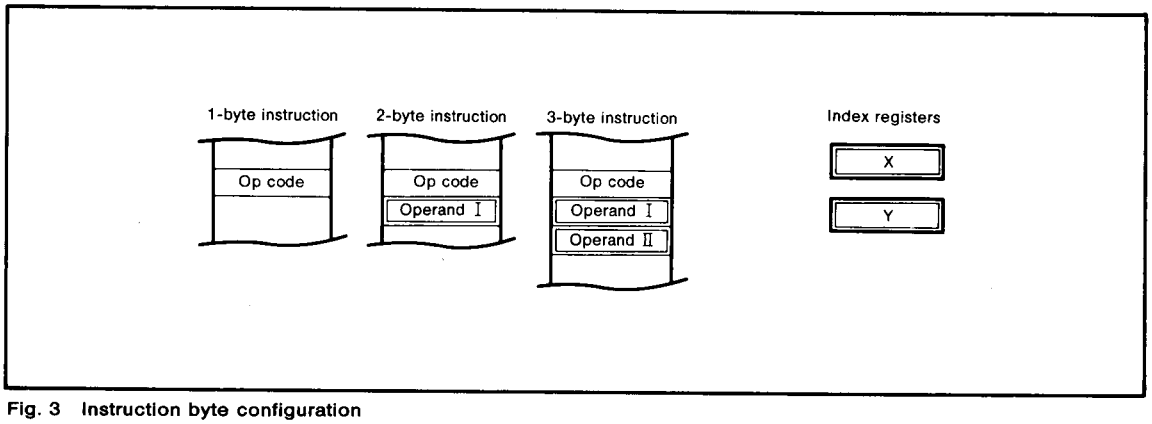
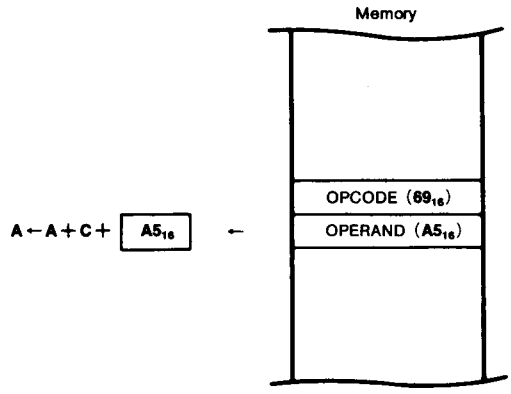
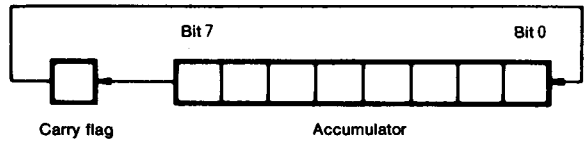


Fig. 3 Instruction byte configuration

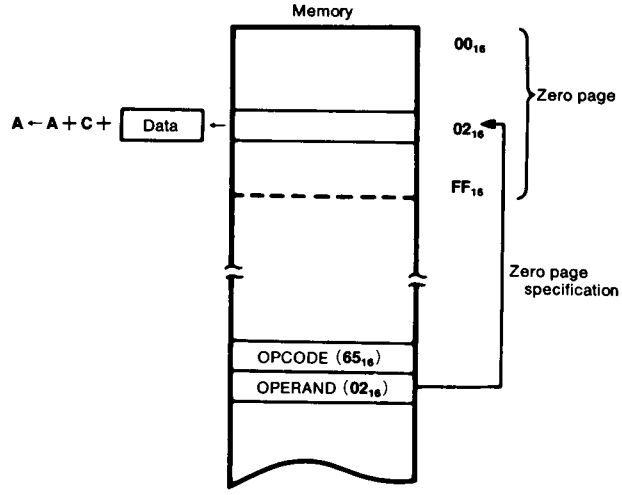
Name : Immediate addressing mode
Function : The OPERAND follows immediately after the OPCODE.
Instructions : **ADC, AND, CMP, CPX, CPY, EOR, LDA, LDX, LDY, ORA, SBC**
Example : Mnemonic Machine code
ADC #$A5 69_{16} $A5_{16}$



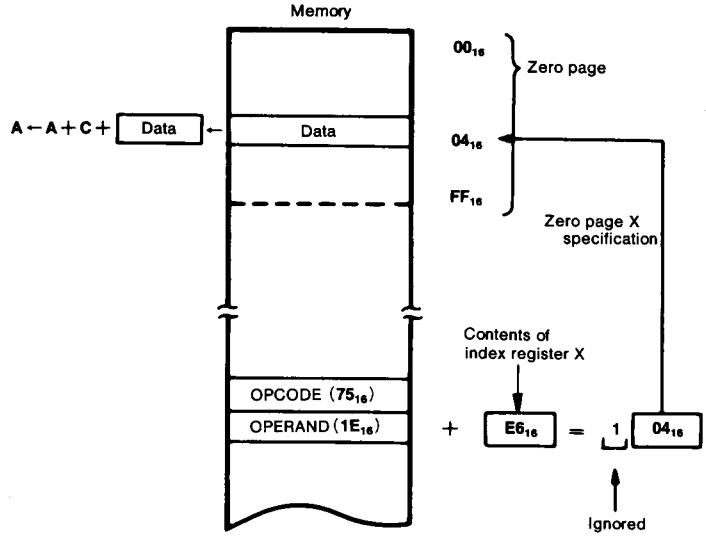
Name : Accumulator addressing mode
Function : The operation is performed on the accumulator.
Instructions : **ASL, DEC, INC, LSR, ROL, ROR**
Example : Mnemonic Machine code
ROL A $2A_{16}$



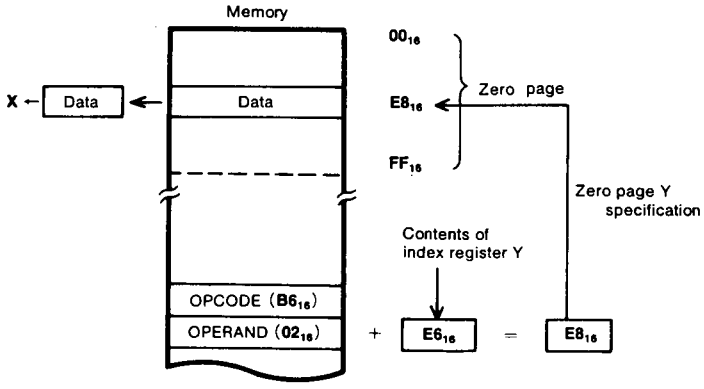
Name : Zero page addressing mode
Function : The operation is performed in zero page memory (00_{16} to FF_{16})
Instructions : **ADC, AND, ASL, BIT, CMP, COM, CPX, CPY, DEC, EOR, INC, LDA, LDM, LDX, LDY, LSR, ORA, ROL, ROR, RRF, SBC, STA, STX, STY, TST**
Example : Mnemonic — Machine code
ADC $02 — 65_{16} 02_{16}



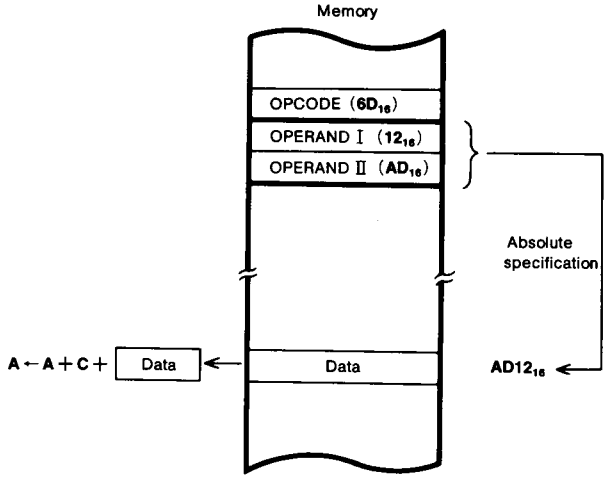
Name : Zero page X addressing mode
Function : The operation is performed on the zero page memory location whose address is specified by adding the OPERAND to the contents of index register X.
Instructions : **ADC, AND, ASL, CMP, DEC, DIV, EOR, INC, LDA, LDY, LSR, MUL, ORA, ROL, ROR, SBC, STA, STY**
Example : Mnemonic — Machine code
ADC $1E,X — 75_{16} $1E_{16}$



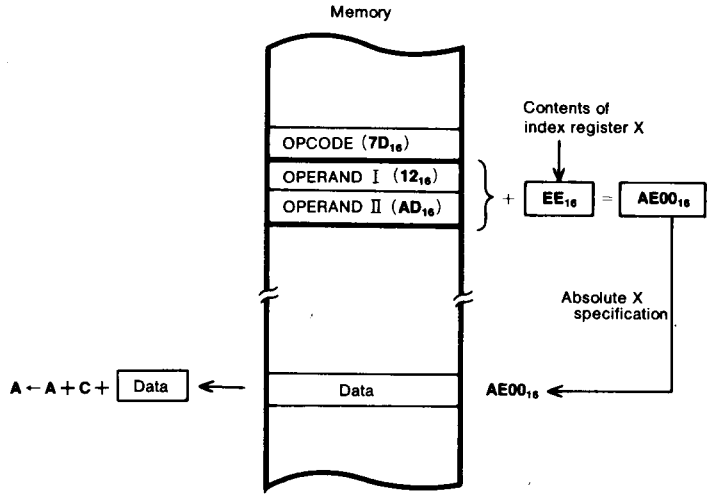
Name : Zero page Y addressing mode
Function : The operation is performed on the zero page memory location whose address is specified by adding the OPERAND to the contents of index register X.
Instructions : **LDX, STX**
Example : Mnemonic Machine code
LDX $02,Y **$B6_{16}$ 02_{16}**



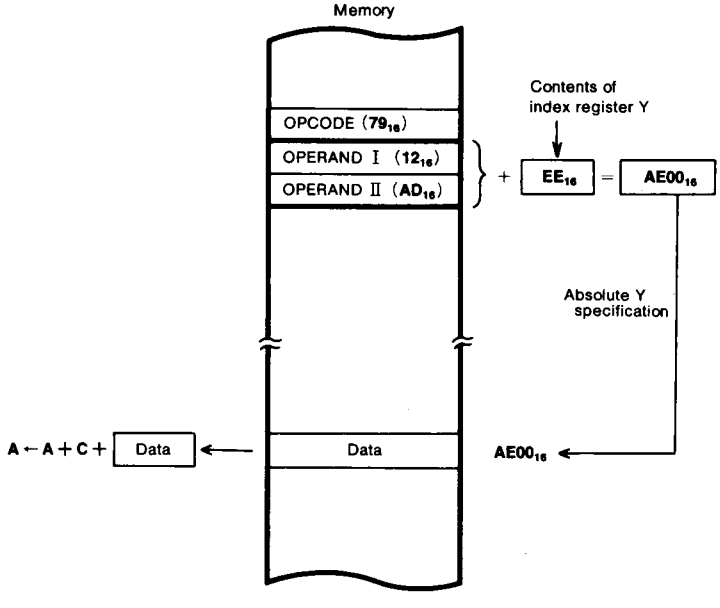
Name : Absolute addressing mode
Function : The operation is performed on the memory whose address is specified by first and second OPERAND.
Instructions : **ADC, AND, ASL, BIT, CMP, CPX, CPY, DEC, EOR, INC, JMP, JSR, LDA, LDX, LDY, LSR, ORA, ROL, ROR, SBC, STA, STX, STY**
Example : Mnemonic Machine code
ADC $AD12 **$6D_{16}$ 12_{16} AD_{16}**



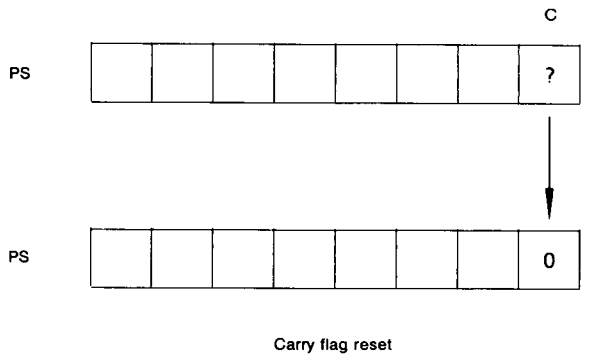
Name : Absolute X addressing mode
Function : The operation is performed on the memory location whose address is specified by adding the contents of index register X to the value indicated by the first and second OPERAND.
Instructions : **ADC, AND, ASL, CMP, DEC, EOR, INC, LDA, LDY, LSR, ORA, ROL, ROR, SBC, STA**
Example : Mnemonic Machine code
ADC $AD12,X $7D_{16}$ 12_{16} AD_{16}



Name : Absolute Y addressing mode
Function : The operation is performed on the memory location whose address is specified by adding the contents of index register Y to the value indicated by the first and second OPERAND.
Instructions : **ADC, AND, CMP, EOR, LDA, LDX, ORA, SBC, STA**
Example : Mnemonic Machine code
ADC $AD12,Y 79_{16} 12_{16} AD_{16}



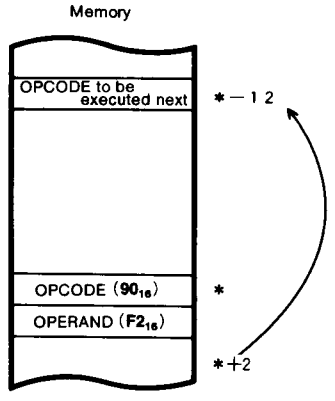
Name : Implied addressing mode
Function : Implied addressing mode operations need no OPERAND.
Instructions : **BRK, CLC, CLD, CLI, CLT, CLV, DEX, DEY, FST, INX, INY, NOP, PHA, PHP, PLA, PLP, RTI, RTS, SEC, SED, SEI, SET, SLW, STP, TAX, TAY, TSX, TXA, TXS, TYA, WIT**
Example : Mnemonic Machine code
CLC **18_{16}**



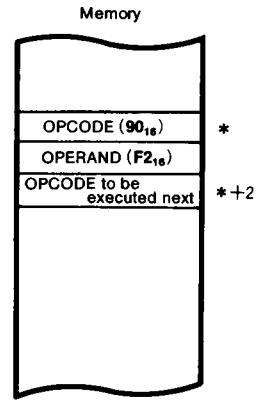
Carry flag reset

Name : **Relative addressing mode**
Function : Conditionally jumps to the address produced by adding the Program Counter to the OPERAND.
Instructions : **BCC, BCS, BEQ, BMI, BNE, BPL, BRA, BVC, BVS**
Example : Mnemonic Machine code
BCC ＊−12 **90_{16} $F2_{16}$**

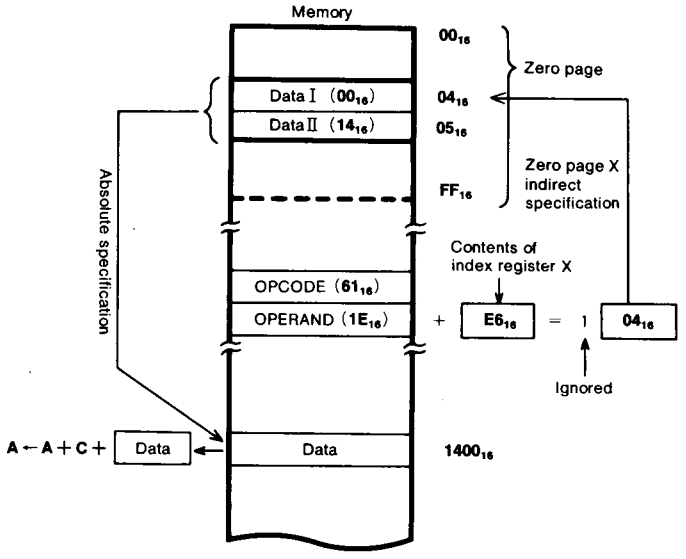
Jumps to ＊−12 address when carry flag(C) is cleared.



Proceed to next address when carry flag(C) is set.

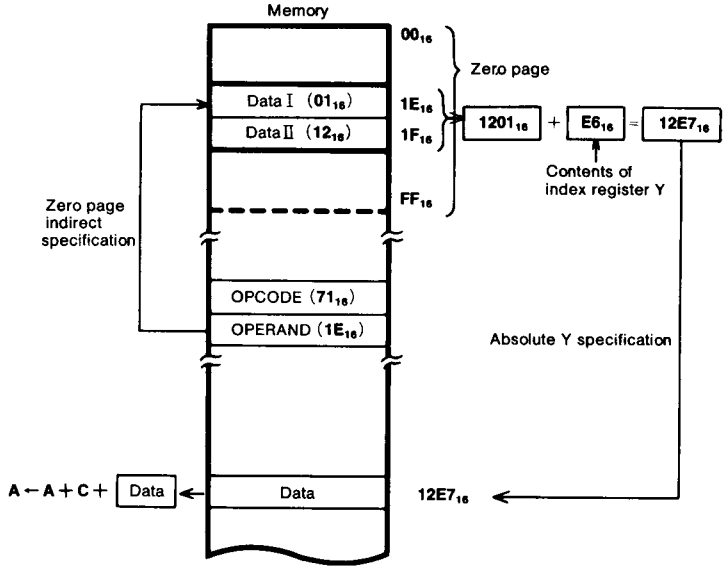


Name : Indirect X addressing mode
Function : The operation is performed on the memory location indicated by the contents of two consecutive bytes in zero page memory whose first address is specified by adding the OPERAND and the contents of index register X.
Instructions : **ADC, AND, CMP, EOR, LDA, ORA, SBC, STA**
Example : Mnemonic Machine code
ADC ($1E,X) 61_{16} $1E_{16}$



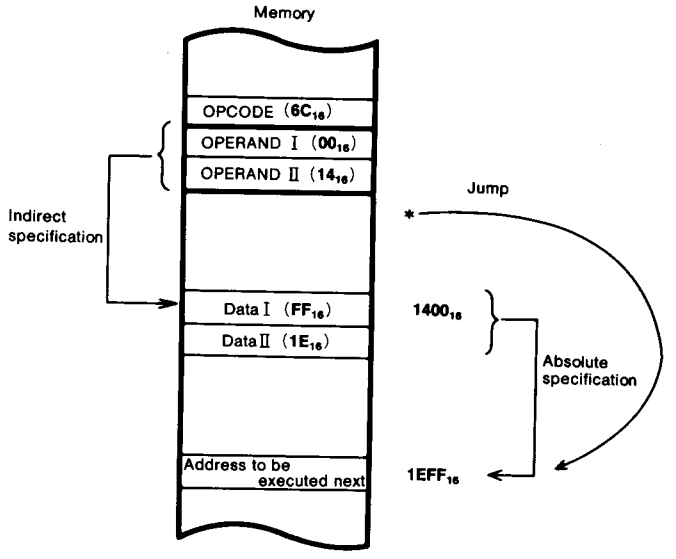
In this example, data I (00_{16}) and data II (14_{16}) have been stored beforehand.

Name : Indirect Y addressing mode
Function : The operation is performed on the memory location indicated by adding the contents of index register Y to the contents of two consecutive bytes in zero page memory whose first address is specified by the OPERAND.
Instructions : **ADC, AND, CMP, EOR, LDA, ORA, SBC, STA**
Example : Mnemonic Machine code
ADC ($1E),Y 71_{16} $1E_{16}$



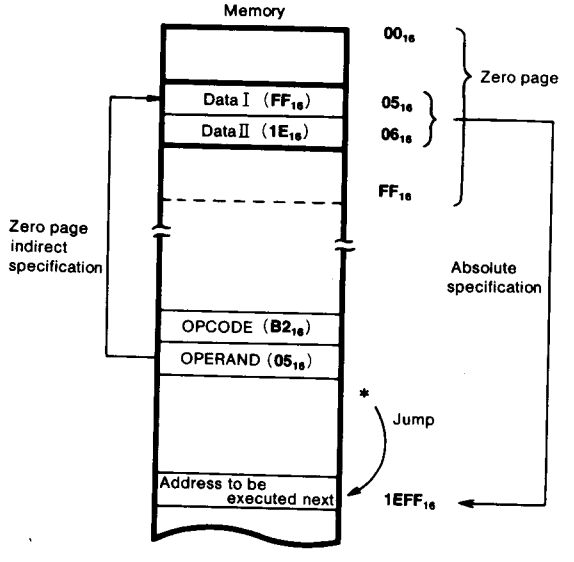
In this example, data I (01_{16}) and Data II (12_{16}) have been stored beforehand.

Name : Indirect absolute addressing mode
Function : Jumps to the location specified by the contents of two consecutive bytes whose first address is specified by the first and second OPERAND.
Instructions : **JMP**
Example : Mnemonic — Machine code
JMP ($1400) — **$6C_{16}$ 00_{16} 14_{16}**



In this example, FF_{16} as data I and $1E_{16}$ as data II have been stored beforehand.

Name : Zero page indirect absolute addressing mode
Function : Jumps to the location specified by the contents of two consecutive bytes in zero page memory whose first address is specified by the OPERAND.
Instructions : **JMP, JSR**
Example : Mnemonic — Machine code
JMP ($05) — **$B2_{16}$ 05_{16}**



In this example, FF_{16} as data I and $1E_{16}$ as data II have been stored beforehand.

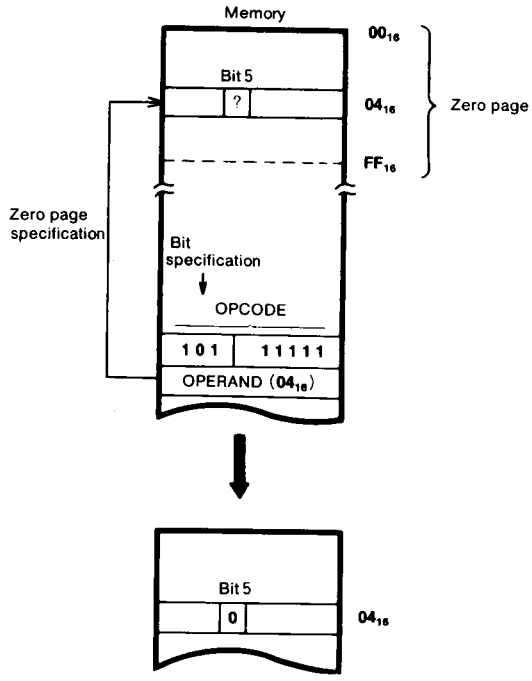
Name : Zero page bit addressing mode

Function : The operation is performed on the bit (specified by the three high order bits of the OPCODE), on the zero page memory location specified by the OPERAND.

Instructions : **CLB, SEB**

Example : Mnemonic — Machine code

CLB 5,$04 — **$BF_{16}$ 04_{16}**



Name : **Zero page bit relative addressing mode**

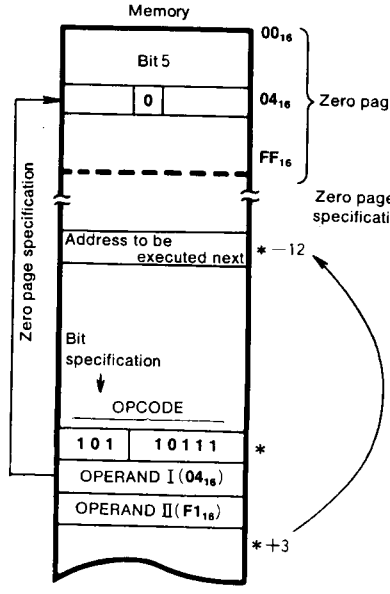
Function : Conditionally jumps to the address specified by adding the second OPERAND to the program counter, depending on the bit (specified by the three higher order bits of the OPCODE) in the zero page memory location specified by the first OPERAND.

Instructions : **BBC, BBS**

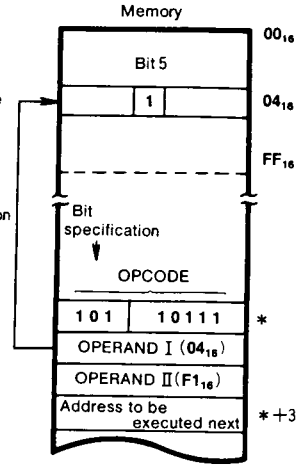
Example : Mnemonic — Machine code

BBC 5,$04,*−12 — **$B7_{16}$ 04_{16} $F1_{16}$**

Jump to * − 12 address when 04_{16} address bit 5 is cleared.



Advance to * +3 address when 04_{16} address bit 5 is set.



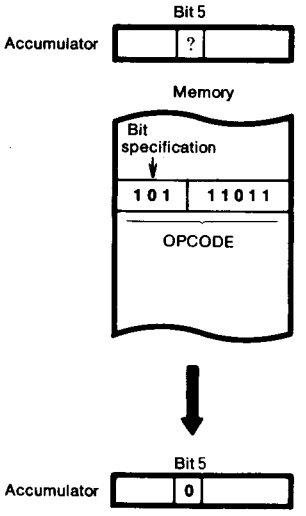
Name : Accumulator bit addressing mode

Function : The operation is performed on the bit in the accumulator which is specified by the three high order bits of the OPCODE. There is no OPERAND.

Instructions : **CLB, SEB**

Example : Mnemonic — Machine code

CLB 5,A — **BB_{16}**



Name : Accumulator bit relative addressing mode

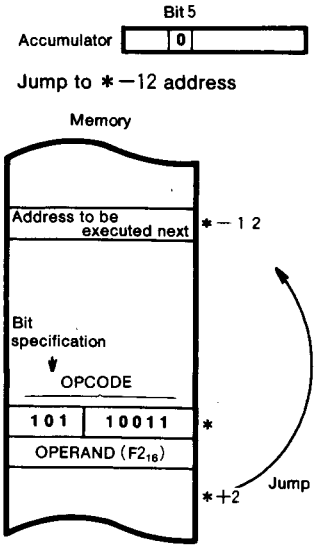
Function : Conditionally jumps to the address produced by adding the OPERAND to the program counter, depending on the bit in accumulator (specified by the high order three bits of the OPCODE).

Instructions : **BBC, BBS**

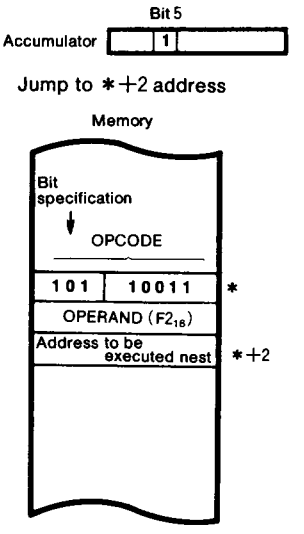
Example : Mnemonic — Machine code

BBC 5,A,*−12 — **$B3_{16}$ $F2_{16}$**

When accumulator bit 5 is cleared

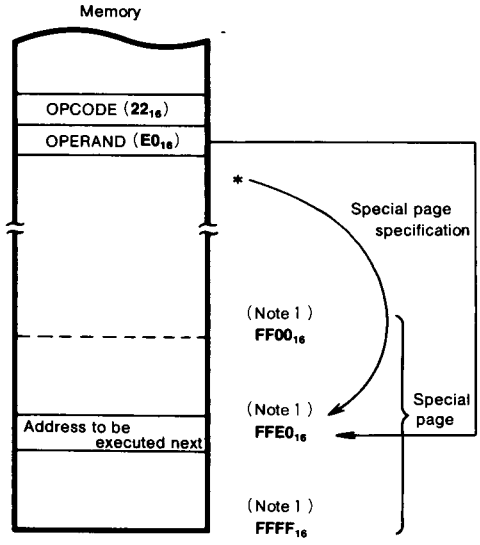


When accumulator bit 5 is set



Name : **Special page addressing mode**
Function : Jumps to the specified address in the special page area. The lower eight bits are specified by the OPERAND and the upper eight bits are defined by the special page (see Note 1).
Instructions : **JSR**
Example : Mnemonic Machine code
JSR ＼$FFE0 22_{16} $E0_{16}$

Note 1 : Note that the special page is defined as the highest addressable 256 bytes of any given microcomputer and may be "FF_{16}", "$1F_{16}$", "$2F_{16}$", etc..



LIST OF INSTRUCTION CODES

$D_3 \sim D_0$ / $D_7 \sim D_4$ / Hexadecimal notation	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	BRK	ORA IND, X	JSR ZP, IND	BBS 0, A	—	ORA ZP	ASL ZP	BBS 0, ZP	PHP	ORA IMM	ASL A	SEB 0, A	—	ORA ABS	ASL ABS	SEB 0, ZP
0001	BPL	ORA IND, Y	CLT	BBC 0, A	—	ORA ZP, X	ASL ZP, X	BBC 0, ZP	CLC	ORA ABS, Y	DEC A	CLB 0, A	—	ORA ABS, X	ASL ABS, X	CLB 0, ZP
0010	JSR ABS	AND IND, X	JSR SP	BBS 1, A	BIT ZP	AND ZP	ROL ZP	BBS 1, ZP	PLP	AND IMM	ROL A	SEB 1, A	BIT ABS	AND ABS	ROL ABS	SEB 1, ZP
0011	BMI	AND IND, Y	SET	BBC 1, A	—	AND ZP, X	ROL ZP, X	BBC 1, ZP	SEC	AND ABS, Y	INC A	CLB 1, A	LDM ZP	AND ABS, X	ROL ABS, X	CLB 1, ZP
0100	RTI	EOR IND, X	STP Note	BBS 2, A	COM ZP	EOR ZP	LSR ZP	BBS 2, ZP	PHA	EOR IMM	LSR A	SEB 2, A	JMP ABS	EOR ABS	LSR ABS	SEB 2, ZP
0101	BVC	EOR IND, Y	—	BBC 2, A	—	EOR ZP, X	LSR ZP, X	BBC 2, ZP	CLI	EOR ABS, Y	—	CLB 2, A	—	EOR ABS, X	LSR ABS, X	CLB 2 ZP
0110	RTS	ADC IND, X	MUL (Note)	BBS 3, A	TST ZP	ADC ZP	ROR ZP	BBS 3, ZP	PLA	ADC IMM	ROR A	SEB 3, A	JMP IND	ADC ABS	ROR ABS	SEB 3, ZP
0111	BVS	ADC IND, Y	—	BBC 3, A	—	ADC ZP, X	ROR ZP, X	BBC 3, ZP	SEI	ADC ABS, Y	—	CLB 3, A	—	ADC ABS, X	ROR ABS, X	CLB 3, ZP
1000	BRA	STA IND, X	RRF ZP	BBS 4, A	STY ZP	STA ZP	STX ZP	BBS 4, ZP	DEY	—	TXA	SEB 4, A	STY ABS	STA ABS	STX ABS	SEB 4, ZP
1001	BCC	STA IND, Y	—	BBC 4, A	STY ZP, X	STA ZP, X	STX ZP, Y	BBC 4, ZP	TYA	STA ABS, Y	TXS	CLB 4, A	—	STA ABS, X	—	CLB 4, ZP
1010	LDY IMM	LDA IND, X	LDX IMM	BBS 5, A	LDY ZP	LDA ZP	LDX ZP	BBS 5, ZP	TAY	LDA IMM	TAX	SEB 5, A	LDY ABS	LDA ABS	LDX ABS	SEB 5, ZP
1011	BCS	LDA IND, Y	JMP ZP, IND	BBC 5, A	LDY ZP, X	LDA ZP, X	LDX ZP, Y	BBC 5, ZP	CLV	LDA ABS, Y	TSX	CLB 5, A	LDY ABS, X	LDA ABS, X	LDX ABS, Y	CLB 5, ZP
1100	CPY IMM	CMP IND, X	SLW / Note WIT	BBS 6, A	CPY ZP	CMP ZP	DEC ZP	BBS 6, ZP	INY	CMP IMM	DEX	SEB 6, A	CPY ABS	CMP ABS	DEC ABS	SEB 6, ZP
1101	BNE	CMP IND, Y	—	BBC 6, A	—	CMP ZP, X	DEC ZP, X	BBC 6, ZP	CLD	CMP ABS, Y	—	CLB 6, A	—	CMP ABS, X	DEC ABS, X	CLB 6, ZP
1110	CPX IMM	SBC IND, X	FST / Note DIV	BBS 7, A	CPX ZP	SBC ZP	INC ZP	BBS 7, ZP	INX	SBC IMM	NOP	SEB 7, A	CPX ABS	SBC ABS	INC ABS	SEB 7, ZP
1111	BEQ	SBC IND, Y	—	BBC 7, A	—	SBC ZP, X	INC ZP, X	BBC 7, ZP	SED	SBC ABS, Y	—	CLB 7, A	—	SBC ABS, X	INC ABS, X	CLB 7, ZP

Note. Support of these instructions depends on the microcomputer type

Instruction	Supported in the following microcomputer types
FST SLW	M50740A-XXXSP, M50740ASP, M50741-XXXSP, M50752-XXXSP, M50757-XXXSP, M50758-XXXSP
MUL DIV	Series 7450, Series 38000, M37424M8-XXXSP, M37524M4-XXXSP

Instruction	Not supported in the following microcomputer types
WIT	M50740A-XXXSP, M50740ASP, M50741-XXXSP, M50752-XXXSP, M50757-XXXSP, M50758-XXXSP
STP	M50752-XXXSP, M50757-XXXSP, M50758-XXXSP, M37424M8-XXXSP, M37524M4-XXXSP

☐ 3-byte instruction
☐ 2-byte instruction
☐ 1-byte instruction

MACHINE INSTRUCTIONS

Symbol	Function	Details	Addressing mode																
			IMP			IMM			A			BIT,A			ZP			BIT,ZP		
			OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#
ADC (Note 1) (Note 6)	When T=0 A←A+M+C When T=1 M(X)←M(X)+M+C	Adds the carry, accumulator and memory contents. The results are entered into the accumulator. Adds the contents of the memory in the address indicated by index register X, the contents of the memory specified by the addressing mode and the carry. The results are entered into the memory at the address indicated by index register X.				69	2	2							65	3	2			
AND (Note 1)	When T=0 A←A∧M When T=1 M(X)←M(X)∧M	"AND's" the accumulator and memory contents. The results are entered into the accumulator. "AND's" the contents of the memory of the address indicated by index register X and the contents of the memory specified by the addressing mode. The results are entered into the memory at the address indicated by index register X.				29	2	2							25	3	2			
ASL	7 0 C ← [ ] ←0	Shifts the contents of accumulator or contents of memory one bit to the left. The low order bit of the accumulator or memory is cleared and the high order bit is shifted into the carry flag.							0A	2	1				06	5	2			
BBC (Note 4)	A_b or M_b=0?	Branches when the contents of the bit specified in the accumulator or memory is "0".										13 + 2i	4	2				17 + 2i	5	3
BBS (Note 4)	A_b or M_b=1?	Branches when the contents of the bit specified in the accumulator or memory is "1".										03 + 2i	4	2				07 + 2i	5	3
BCC (Note 4)	C=0?	Branches when the contents of carry flag is "0".																		
BCS (Note 4)	C=1?	Branches when the contents of carry flag is "1".																		
BEQ (Note 4)	Z=1?	Branches when the contents of zero flag is "1".																		
BIT	A∧M	"AND's" the contents of accumulator and memory. The results are not entered anywhere.													24	3	2			
BMI (Note 4)	N=1?	Branches when the contents of negative flag is "1".																		
BNE (Note 4)	Z=0?	Branches when the contents of zero flag is "0".																		
BPL (Note 4)	N=0?	Branches when the contents of negative flag is "0".																		
BRA	PC←PC±offset	Jumps to address specified by adding offset to the program counter.																		
BRK	B←1 M(S)←PC_H S←S−1 M(S)←PC_L S←S−1 M(S)←PS S←S−1 PC_L←AD_L PC_H←AD_H	Executes a software interrupt.	00	7	1															

MITSUBISHI ELECTRIC

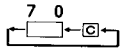
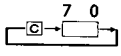
Addressing mode																													Processor status register							
ZP,X			ZP,Y			ABS			ABS,X			ABS,Y			IND			ZP,IND			IND,X			IND,Y			REL			SP			7	6	5	4	3	2	1	0
OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	N	V	T	B	D	I	Z	C
75	4	2				6D	4	3	7D	5	3	79	5	3							61	6	2	71	6	2							N	V	•	•	•	•	Z	C
35	4	2				2D	4	3	3D	5	3	39	5	3							21	6	2	31	6	2							N	•	•	•	•	•	Z	•
16	6	2				0E	6	3	1E	7	3																						N	•	•	•	•	•	Z	C
																																	•	•	•	•	•	•	•	•
																																	•	•	•	•	•	•	•	•
																											90	2	2				•	•	•	•	•	•	•	•
																											B0	2	2				•	•	•	•	•	•	•	•
																											F0	2	2				•	•	•	•	•	•	•	•
						2C	4	3																									M_7	M_6	•	•	•	Z	•
																											30	2	2				•	•	•	•	•	•	•	•
																											D0	2	2				•	•	•	•	•	•	•	•
																											10	2	2				•	•	•	•	•	•	•	•
																											80	4	2				•	•	•	•	•	•	•	•
																																	•	•	•	1	•	1	•	•

Symbol	Function	Details	Addressing mode																	
			IMP			IMM			A			BIT,A			ZP			BIT,ZP		
			0P	n	#	0P	n	#	0P	n	#	0P	n	#	0P	n	#	0P	n	#
BVC (Note 4)	V=0?	Branches when the contents of overflow flag is "0."																		
BVS (Note 4)	V=1?	Branches when the contents of overflow flag is "1."																		
CLB	A_b or M_b←0	Clears the contents of the bit specified in the accumulator or memory to "0."										1B+2i	2	1				1F+2i	5	2
CLC	C←0	Clears the contents of the carry flag to "0."	18	2	1															
CLD	D←0	Clears the contents of decimal mode flag to "0."	D8	2	1															
CLI	I←0	Clears the contents of interrupt disable flag to "0."	58	2	1															
CLT	T←0	Clears the contents of index X mode flag to "0."	12	2	1															
CLV	V←0	Clears the contents overflow flag to "0."	B8	2	1															
CMP (Note 3)	When T=0 A−M When T=1 M(X)−M	Compares the contents of accumulator and memory. Compares the contents of the memory specified by the addressing mode with the contents of the address indicated by index register X.				C9	2	2							C5	3	2			
COM	M←$\overline{M}$	Forms a one's complement of the contents of memory, and stores it into memory.													44	5	2			
CPX	X−M	Compares the contents of index register X and memory.				E0	2	2							E4	3	2			
CPY	Y−M	Compares the contents of index register Y and memory.				C0	2	2							C4	3	2			
DEC	A←A−1 or M←M−1	Decrements the contents of the accumulator or memory by 1.							1A	2	1				C6	5	2			
DEX	X←X−1	Decrements the contents of index register X by 1.	CA	2	1															
DEY	Y←Y−1	Decrements the contents of index register Y by 1.	88	2	1															
DIV (Note 5)	A←(M(zz+X+1), M(zz+X))/A M (S) ← 1's complement of Remainder S←S−1	Divides the 16-bit data that is the contents of M(zz+x+1) for high byte and the contents of M(zz+x) for low byte by the accumulator. Stores the quotient in the accumulator and the 1's complement of the remainder on the stack.																		
EOR (Note 1)	When T=0 A←A∀M When T=1 M(X)←M(X)∀M	"Exclusive-ORs" the contents of accumulator and memory. The results are stored in the accumulator. "Exclusive-ORs" the contents of the memory specified by the addressing mode and the contents of the memory at the address indicated by index register X. The results are stored into the memory at the address indicated by index register X.				49	2	2							45	3	2			
FST (Note 5)		Connects oscillator output to the X_{OUTF} pin.	E2	2	1															
INC	A←A+1 or M←M+1	Increments the contents of accumulator or memory by 1.							3A	2	1				E6	5	2			
INX	X←X+1	Increments the contents of index register X by 1.	E8	2	1															
INY	Y←Y+1	Increments the contents of index register Y by 1.	C8	2	1															

Addressing mode																														Processor status register							
ZP,X			ZP,Y			ABS			ABS,X			ABS,Y			IND			ZP,IND			IND,X			IND,Y			REL			SP			7	6	5	4	3	2	1	0
OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	N	V	T	B	D	I	Z	C
																											50	2	2				•	•	•	•	•	•	•	•
																											70	2	2				•	•	•	•	•	•	•	•
																																	•	•	•	•	•	•	•	•
																																	•	•	•	•	•	•	•	0
																																	•	•	•	•	0	•	•	•
																																	•	•	•	•	•	0	•	•
																																	•	•	0	•	•	•	•	•
																																	•	0	•	•	•	•	•	•
D5	4	2				CD	4	3	DD	5	3	D9	5	3							C1	6	2	D1	6	2							N	•	•	•	•	•	Z	C
																																	N	•	•	•	•	•	Z	•
						EC	4	3																									N	•	•	•	•	•	Z	C
						CC	4	3																									N	•	•	•	•	•	Z	C
D6	6	2				CE	6	3	DE	7	3																						N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•
E2	16	2																															•	•	•	•	•	•	•	•
55	4	2				4D	4	3	5D	5	3	59	5	3							41	6	2	51	6	2							N	•	•	•	•	•	Z	•
																																	•	•	•	•	•	•	•	•
F6	6	2				EE	6	3	FE	7	3																						N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•

Symbol	Function	Details	Addressing mode																	
			IMP			IMM			A			BIT,A			ZP			BIT,ZP		
			OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#
JMP	If addressing mode is ABS $PC_L \leftarrow AD_L$ $PC_H \leftarrow AD_H$ If addressing mode is IND $PC_L \leftarrow M(AD_H, AD_L)$ $PC_H \leftarrow M(AD_H, AD_L+1)$ If addressing mode is ZP, IND $PC_L \leftarrow M(00, AD_L)$ $PC_H \leftarrow M(00, AD_L+1)$	Jumps to the specified address.																		
JSR	$M(S) \leftarrow PC_H$ $S \leftarrow S-1$ $M(S) \leftarrow PC_L$ $S \leftarrow S-1$ After executing the above, if addressing mode is ABS, $PC_L \leftarrow AD_L$ $PC_H \leftarrow AD_H$ If addressing mode is SP, $PC_L \leftarrow AD_L$ $PC_H \leftarrow FF$ If addressing mode is ZP, IND, $PC_L \leftarrow M(00, AD_L)$ $PC_H \leftarrow M(00, AD_L+1)$	After storing contents of program counter in stack, and jumps to the specified address.																		
LDA (Note 2)	When T=0 $A \leftarrow M$ When T=1 $M(X) \leftarrow M$	Load accumulator with contents of memory. Load memory indicated by index register X with contents of memory specified by the addressing mode.				A9	2	2							A5	3	2			
LDM	$M \leftarrow nn$	Load memory with immediate value.													3C	4	3			
LDX	$X \leftarrow M$	Load index register X with contents of memory.				A2	2	2							A6	3	2			
LDY	$Y \leftarrow M$	Load index register Y with contents of memory.				A0	2	2							A4	3	2			
LSR	7 0 $0 \rightarrow \square \rightarrow C$	Shift the contents of accumulator or memory to the right by one bit. The low order bit of accumulator or memory is stored in carry, 7th bit is cleared.							4A	2	1				46	5	2			
MUL (Note 5)	$M(S) \cdot A \leftarrow A \times M(zz+X)$ $S \leftarrow S-1$	Multiplies the accumulator with the contents of memory specified by the zero page X addressing mode and stores the high byte of the result on the stack and the low byte in the accumulator.																		
NOP	$PC \leftarrow PC+1$	No operation.	EA	2	1															
ORA (Note 1)	When T=0 $A \leftarrow A \vee M$ When T=1 $M(X) \leftarrow M(X) \vee M$	"Logical OR's" the contents of memory and accumulator. The result is stored in the accumulator. "Logical OR's" the contents of memory indicated by index register X and contents of memory specified by the addressing mode. The result is stored in the memory specified by index register X.				09	2	2							05	3	2			

Addressing mode																														Processor status register							
ZP,X			ZP,Y			ABS			ABS,X			ABS,Y			IND			ZP,IND			IND,X			IND,Y			REL			SP			7	6	5	4	3	2	1	0
OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	N	V	T	B	D	I	Z	C
						4C	3	3							6C	5	3	B2	4	2													•	•	•	•	•	•	•	•
						20	6	3										02	7	2										22	5	2	•	•	•	•	•	•	•	•
B5	4	2				AD	4	3	BD	5	3	B9	5	3							A1	6	2	B1	6	2							N	•	•	•	•	•	Z	•
																																	•	•	•	•	•	•	•	•
			B6	4	2	AE	4	3				BE	5	3																			N	•	•	•	•	•	Z	•
B4	4	2				AC	4	3	BC	5	3																						N	•	•	•	•	•	Z	•
56	6	2				4E	6	3	5E	7	3																						0	•	•	•	•	•	Z	C
62	15	2																															•	•	•	•	•	•	•	•
																																	•	•	•	•	•	•	•	•
15	4	2				0D	4	3	1D	5	3	19	5	3							01	6	2	11	6	2							N	•	•	•	•	•	Z	•

Symbol	Function	Details	Addressing mode																	
			IMP			IMM			A			BIT,A			ZP			BIT,ZP		
			OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#
PHA	M(S)←A S←S−1	Saves the contents of the accumulator in memory at the address indicated by the stack pointer and decrements the contents of stack pointer by 1.	48	3	1															
PHP	M(S)←PS S←S−1	Saves the contents of the processor status register in memory at the address indicated by the stack pointer and decrements the contents of the stack pointer by 1.	08	3	1															
PLA	S←S+1 A←M(S)	Increments the contents of the stack pointer by 1 and restores the accumulator from the memory at the address indicated by the stack pointer.	68	4	1															
PLP	S←S+1 PS←M(S)	Increments the contents of stack pointer by 1 and restores the processor status register from the memory at the address indicated by the stack pointer.	28	4	1															
ROL		Shifts the contents of the memory or accumulator to the left by one bit. The high order bit is shifted into the carry flag and the carry flag is shifted into the low order bit.							2A	2	1				26	5	2			
ROR		Shifts the contents of the memory or accumulator to the right by one bit. The low order bit is shifted into the carry flag and the carry flag is shifted into the high order bit.							6A	2	1				66	5	2			
RRF		Rotates the contents of memory to the right by 4 bits.													82	8	2			
RTI	S←S+1 PS←M(S) S←S+1 PC_L←M(S) S←S+1 PC_H←M(S)	Returns from an interrupt routine to the main routine.	40	6	1															
RTS	S←S+1 PC_L←M(S) S←S+1 PC_H←M(S)	Returns from a subroutine to the main routine.	60	6	1															
SBC (Note 1) (Note 6)	When T=0 A←A−M−$\overline{C}$ When T=1 M(X)←M(X)−M−$\overline{C}$	Subtracts the contents of memory and complement of carry flag from the contents of accumulator. The results are stored into the accumulator. Subtracts contents of complement of carry flag and contents of the memory indicated by the addressing mode from the memory at the address indicated by index register X. The results are stored into the memory of the address indicated by index register X.				E9	2	2							E5	3	2			
SEB	A_b or M_b←1	Sets the specified bit in the accumulator or memory to "1."										0B+2i	2	1				0F+2i	5	2
SEC	C←1	Sets the contents of the carry flag to "1."	38	2	1															
SED	D←1	Sets the contents of the decimal mode flag to "1."	F8	2	1															
SEI	I←1	Sets the contents of the interrupt disable flag to "1."	78	2	1															
SET	T←1	Sets the contents of the index X mode flag to "1."	32	2	1															
SLW (Note 5)		Disconnects the oscillator output from the X_{OUTF} pin.	C2	2	1															

Addressing mode																																Processor status register							
ZP,X			ZP,Y			ABS			ABS,X			ABS,Y			IND			ZP,IND			IND,X			IND,Y			REL			SP			7	6	5	4	3	2	1	0
OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	N	V	T	B	D	I	Z	C
																																•	•	•	•	•	•	•	•
																																•	•	•	•	•	•	•	•
																																N	•	•	•	•	•	Z	•
																																(Value saved in stack)							
36	6	2				2E	6	3	3E	7	3																					N	•	•	•	•	•	Z	C
76	6	2				6E	6	3	7E	7	3																					N	•	•	•	•	•	Z	C
																																•	•	•	•	•	•	•	•
																																(Value saved in stack)							
																																•	•	•	•	•	•	•	•
F5	4	2				ED	4	3	FD	5	3	F9	5	3							E1	6	2	F1	6	2							N	V	•	•	•	•	Z	C
																																•	•	•	•	•	•	•	•
																																•	•	•	•	•	•	•	1
																																•	•	•	•	1	•	•	•
																																•	•	•	•	•	1	•	•
																																•	•	1	•	•	•	•	•
																																•	•	•	•	•	•	•	•

Symbol	Function	Details	Addressing mode																	
			IMP			IMM			A			BIT,A			ZP			BIT,ZP		
			OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#
STA	M←A	Stores the contents of accumulator in memory.													85	4	2			
STP (Note 5)		Stops the oscillator.	42	2	1															
STX	M←X	Stores the contents of index register X in memory.													86	4	2			
STY	M←Y	Stores the contents of index register Y in memory.													84	4	2			
TAX	X←A	Transfers the contents of the accumulator to index register X.	AA	2	1															
TAY	Y←A	Transfers the contents of the accumulator to index register Y.	A8	2	1															
TST	M=0?	Tests whether the contents of memory are "0" or not.													64	3	2			
TSX	X←S	Transfers the contents of the stack pointer to index register X.	BA	2	1															
TXA	A←X	Transfers the contents of index register X to the accumulator.	8A	2	1															
TXS	S←X	Transfers the contents of index register X to the stack pointer.	9A	2	1															
TYA	A←Y	Transfers the contents of index register Y to the accumulator.	98	2	1															
WIT (Note 5)		Stops the internal clock.	C2	2	1															

Note 1 : The number of cycles "n" is increased by 3 when T is 1.
2 : The number of cycles "n" is increased by 2 when T is 1.
3 : The number of cycles "n" is increased by 1 when T is 1.
4 : The number of cycles "n" is increased by 2 when branching has occurred.
5 : Support of these instructions depends on the microcomputer type.

Instruction	Supported in the following microcomputer types
FST SLW	M50740A-XXXSP, M50740ASP, M50741-XXXSP, M50752-XXXSP, M50757-XXXSP, M50758-XXXSP
MUL DIV	Series 7450, Series 38000, M37424M8-XXXSP, M37524M4-XXXSP

Instruction	Not supported in the following microcomputer types
WIT	M50740A-XXXSP, M50740ASP, M50741-XXXSP, M50752-XXXSP, M50757-XXXSP, M50758-XXXSP
STP	M50752-XXXSP, M50757-XXXSP, M50758-XXXSP, M37424M8-XXXSP, M37524M4-XXXSP

6 : N, V, and Z flags are invalid in decimal operation mode.

Addressing mode																														Processor status register							
ZP,X			ZP,Y			ABS			ABS,X			ABS,Y			IND			ZP,IND			IND,X			IND,Y			REL			SP			7	6	5	4	3	2	1	0
OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	N	V	T	B	D	I	Z	C
95	5	2				8D	5	3	9D	6	3	99	6	3							81	7	2	91	7	2							•	•	•	•	•	•	•	•
																																	•	•	•	•	•	•	•	•
			96	5	2	8E	5	3																									•	•	•	•	•	•	•	•
94	5	2				8C	5	3																									•	•	•	•	•	•	•	•
																																	N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•
																																	•	•	•	•	•	•	•	•
																																	N	•	•	•	•	•	Z	•
																																	•	•	•	•	•	•	•	•

Symbol	Contents
IMP	Implied addressing mode
IMM	Immediate addressing mode
A	Accumulator or Accumulator addressing mode
BIT, A	Accumulator bit relative addressing mode
ZP	Zero page addressing mode
BIT, ZP	Zero page bit relative addressing mode
ZP, X	Zero page X addressing mode
ZP, Y	Zero page Y addressing mode
ABS	Absolute addressing mode
ABS, X	Absolute X addressing mode
ABS, Y	Absolute Y addressing mode
IND	Indirect absolute addressing mode
ZP, IND	Zero page indirect absolute addressing mode
IND, X	Indirect X addressing mode
IND, Y	Indirect Y addressing mode
REL	Relative addressing mode
SP	Special page addressing mode
C	Carry flag
Z	Zero flag
I	Interrupt disable flag
D	Decimal mode flag
B	Break flag
T	X-modified arithmetic mode flag
V	Overflow flag
N	Negative flag

Symbol	Contents
+	Addition
−	Subtraction
∧	Logical OR
∨	Logical AND
∀	Logical exclusive OR
—	Negation
←	Shows direction of data flow
X	Index register X
Y	Index register Y
S	Stack pointer
PC	Program counter
PS	Processor status register
PC_H	8 high-order bits of program counter
PC_L	8 low-order bits of program counter
AD_H	8 high-order bits of address
AD_L	8 low-order bits of address
FF	FF in Hexadecimal notation
nn	Immediate value
M	Memory specified by address designation of any addressing mode
M (X)	Memory of address indicated by contents of index register X
M (S)	Memory of address indicated by contents of stack pointer
$M(AD_H, AD_L)$	Contents of memory at address indicated by AD_H and AD_L, in AD_H is 8 high-order bits and AD_L is 8 low-order bits.
$M(00, AD_L)$	Contents of address indicated by zero page AD_L
A_b	1 bit of accumulator
M_b	1 bit of memory
OP	Opcode
n	Number of cycles
#	Number of bytes

NOTES on USE

Keep the following points in mind while programming:

Processor status register

(1) Initialization of processor status register

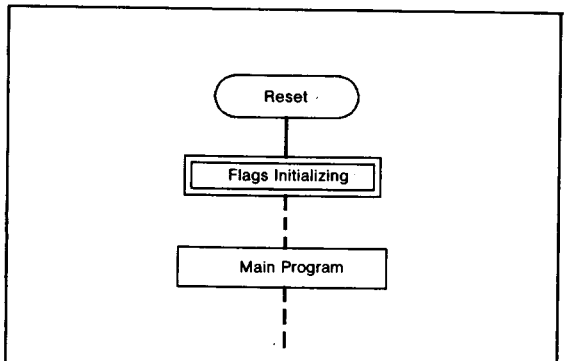
After a reset, the contents of the processor status register (PS) are undefined except for the I flag which is "1". Therefore, flags which affect program execution must be initialized after a reset.

In particular, it is essential to initialize the T and D flags because they have an important effect on calculations.

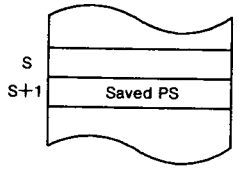
(2) How to reference the processor status register

To reference the contents of the processor status register (PS), execute the PHP instruction once then read the contents of (S+1). If necessary, execute the PLP instruction to return the PS to its original status.

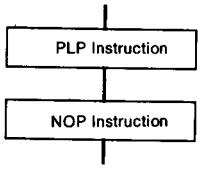
A NOP instruction should be executed after every PLP instruction. (The NOP in unnecessary when using a series 38000 microcomputer).



Initialization of Flags in PS



Stack Memory Contents after PHP Instruction Execution

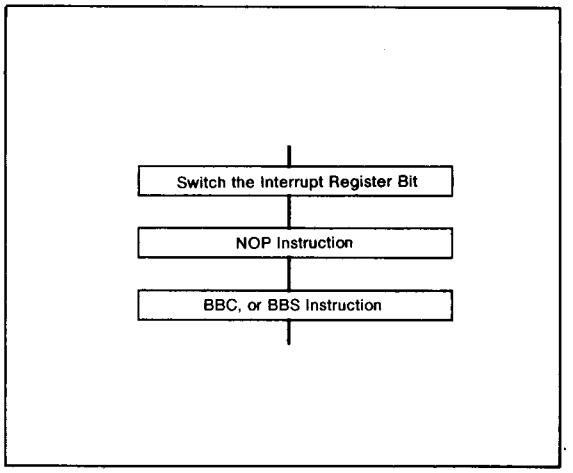


Interrupts

The contents of the interrupt request bits can be changed by software, but the values will not change immediately after being overwritten. Therefore, note the following points:

(1) After changing the value of the interrupt request bits, execute at least one instruction before executing a BBC, BBS, or any other read instruction.

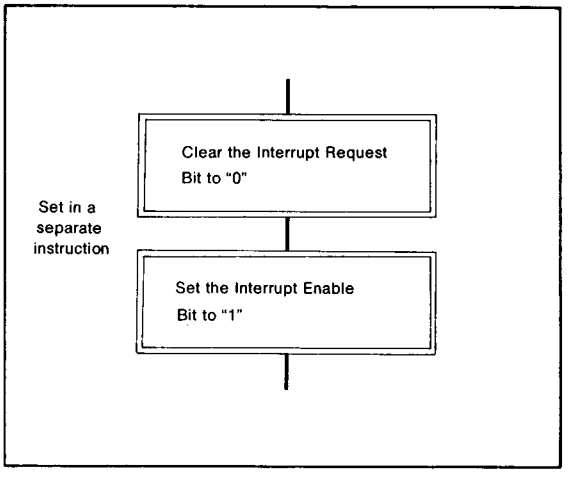
(2) When clearing an interrupt request bit to "0" and setting an interrupt enable bit to "1" (=setting in an interrupt enable state), it needs to be cleared or set these bits in a separate instruction. The interrupt is accepted because it becomes in the interrupt enable state before clearing the interrupt request bit, if clearing the interrupt request bit and setting the interrupt enable bit are performed in an instruction.

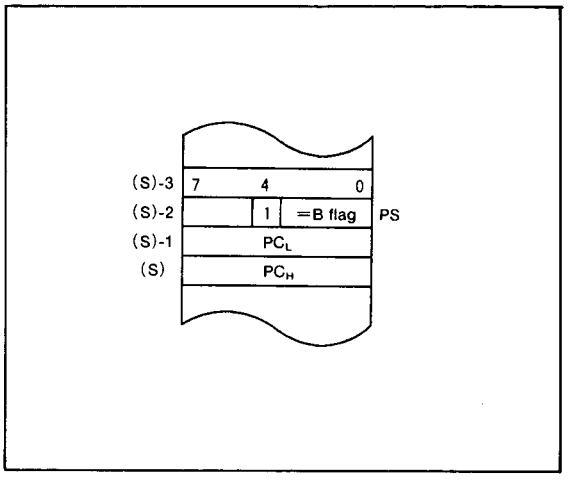


BRK instruction

(1) It can be detected that the BRK instruction interrupt event or the least priority interrupt event by referring the stored B flag state. Refer the stored B flag state in the interrupt routine, in this case.
However, the microcomputer that has an independent BRK instruction interrupt vector (cf. the 7450 series, the 7470 series, and the 38000 series) are not necessary this detection.

(2) The CPU of all 8-bit microcomputers except the 38000 series have the following bug about the BRK instruction execution.
At the following status,
① the interrupt request bit has set to "1".
② the interrupt enable bit has set to "1".
③ the interrupt disable flag (I) has set to "1".
if the BRK instruction is executed, the interrupt disable state is cancelled and it becomes in the interrupt enable state. So that the requested interrupts (the interrupts that corresponding to their request bits have set to "1") are accepted.





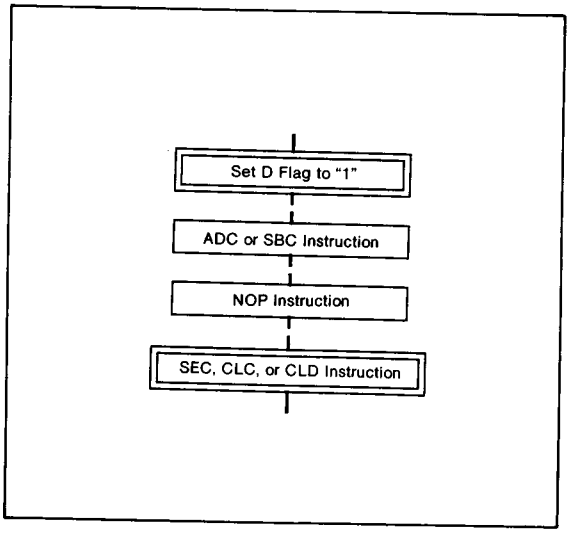
Decimal calculations

(1) Execution of decimal calculations

The ADC and SBC are the only instructions which will yield proper decimal results in decimal mode. To calculate in decimal notation, set the decimal mode flag (D) to "1" with the SED instruction. After executing the ADC or SBC instruction, execute another instruction before executing the SEC, CLC, or CLD instruction.

(2) Note on flags in decimal mode

When decimal mode is selected, the values of three of the flags in the status register (the N, V, and Z flags) are invalid after a ADC or SBC instruction is executed. The Carry flag (C) is set to "1" if a carry is generated as a result of the calculation, or is cleared to "0" if a borrow is generated. To determine whether a calculation has generated a carry, the C flag must be initialized to "0" before each calculation. To check for a borrow, the C flag must be initialized to "1" before each calculation.



JMP instruction

When using the JMP instruction in indirect addressing mode, do not specify the last address on a page as an indirect address.