CHAPTER 6 INSTRUCTION SET

The instructions for the uPD7502 and uPD7503 are a subset of the uPD7500 SET A instructions.

6.1 Operand Format and Description

Operands are coded in the operand field of an instruction according to the format prescribed. For details of the operand format, refer to the assembler specifications.

addr	11-bit immediate data or label (uPD7502) 12-bit immediate data or label (uPD7503)
caddr	11-bit immediate data or label
addr1	3, 4, 5, 6, BH, CH, EH, FH immediate data or label
addr2	3-bit immediate data or label (uPD7502) 4-bit immediate data or label (uPD7503)
addr3	0, 1, 4, 5, 6 immediate data or label
taddr1	0C0H to 0CFH immediate data or label
taddr2	0D0H to 0FFH immediate data or label
mem	7-bit immediate data or label (uPD7502) 8-bit immediate data or label (uPD7503)
byte	8-bit immediate data or label
n4	4-bit immediate data or label
n3	3-bit immediate data or label
bit	2-bit immediate data or label
pr	DL, DE, HL-, HL+, HL

6.2 Legend

A:	Accumulator
D:	D register
E:	E register
H:	H register
L:	L register
DE:	Register pair (DE)
DL:	Register pair (DL)
HL:	Register pair (HL)
pr:	Register pair (DE, HL-, HL+, HL, DL)
SP:	Stack pointer
PC:	Program counter
PCn:	Bit n of the program counter
C:	Carry flag
PSW:	Program status word
SIO:	Shift register
MOD:	Modulo register
CT:	Count register
In:	Immediate data for byte n4 or n3
Pn:	Immediate data for addr, caddr, addr2, taddr1, or taddr2
Bn:	Immediate data for bit
Dn:	Immediate data for mem, addr1, addr3, or addr4
Rn:	Immediate data for pr
(xx):	Data addressed by xx
xxH:	Hexadecimal data

6.3 Instruction Execution Time

The instructions provided for the uPD7502 and uPD7503 are one or two byte long. One-byte instructions are executed in one machine cycle, and 2-byte instructions are executed in two machine cycles, with some exceptions. One machine cycle is equal to one cycle for system clock ϕ. When the RC oscillation frequency or CL1 input frequency is 200 kHz (when the operating voltage is 5 V), one machine cycle is 10 us.

When the skip condition for an instruction having the skip function is established, a skip takes place during one machine cycle, irrespective of whether the instruction to be skipped is a 1-byte or 2-byte instruction. In this case, the instruction execution time is one machine cycle longer than that required when no skip is performed.

6.4 Explanation of Instructions

The instructions are explained in terms of:

① Instruction code (binary)
② Byte count
③ Machine cycle count
④ Function
⑤ Example
⑥ Note

6.4.1 Load/store instructions

LAI n4 (Load A with Immediate)

① Instruction code:

0	0	0	1	I_3	I_2	I_1	I_0

② Byte count: 1

③ Machine cycle count: 1

④ Function: A ← n4 $n4 = I_{3-0}$:0-FH
Four-bit immediate data n4 is loaded into the accumulator. The instruction has a string effect. When the same instruction is coded more than once successively, the second and subsequent instructions are processed as NOPs.

⑤ Example: Subroutines that add 1, 2, and 3 to the data at address Z in binary notation

```
ZAD1:LAI   1;(1)
ZAD2:LAI   2;(2)
ZAD3:LAI   3;(3)
     LHLI  Z
     ASC
     NOP
     XAM HL
     RT
```

(1) When ZAD1 is called

The value 1 is loaded into the accumulator by LAI 1, and the subsequent instructions (2) and (3) change to NOP, and 1 is added to the data at data memory address Z in binary.

(2) When ZAD2 is called

The value 2 is loaded into the accumulator by LAI 2, and instruction (3) changes to NOP, and 2 is added to the data at data memory address Z in binary.

(3) When ZAD3 is called

The value 3 is loaded into the accumulator by LAI 3, and 3 is added to the data at data memory address Z in binary.

LDI n4 (Load D with Immediate)

① Instruction code:

0	0	1	1	1	1	1	0

0	0	1	0	I_3	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: D ← n4 n4=I_{3-0}:0-FH
Four-bit immediate data n4 is loaded to the D register.

LEI n4 (Load E with Immediate)

① Instruction code:

0	0	1	1	1	1	1	0

0	0	0	0	I_3	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: E ← n4 n4=I_{3-0}:0-FH
Four-bit immediate data n4 is loaded to the E register.

LHI n4 (Load H with Immediate)

① Instruction code:

0	0	1	1	1	1	1	0

0	0	1	1	I_3	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: H ← n4 $n4 = I_{3-0}$:0-FH
Four-bit immediate data n4 is loaded in the H register.

LLI n4 (Load L with Immediate)

① Instruction code:

0	0	1	1	1	1	1	0

0	0	0	1	I_3	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: L ← n4 $n4 = I_{3-0}$:0-FH
Four-bit immediate data n4 is loaded to the L register.

LAM pr (Load A with Memory)

① Instruction code:

0	1	0	R_4	0	0	R_1	R_0

② Byte count: 1

③ Machine cycle count: 1 or 2 (when a skip is made)

④ Function: A ← (pr)

When pr=HL+: skip if L=0

When pr=HL-: skip if L=FH

pr	R_4	R_1	R_0
DL	0	0	0
DE	0	0	1
HL-	1	0	0
HL+	1	0	1
HL	1	1	0

The contents of the data memory location addressed by a register pair are loaded into the accumulator. If HL+ is specified for pr, the L register is incremented after data loading. If the result is 0, the following instruction is skipped. If HL- is specified for pr, the the L register is decremented after data loading. If the result is FH, the following instruction is skipped.

⑥ Note: LAM HL-, LAM HL+, and LAM HL can be coded as LDS, LIS, and L, respectively (each having no operand).

LADR mem (Load A Direct): uPD7502

① Instruction code:

0	0	1	1	1	0	0	0	0	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← (mem) mem=D_{7-0}:00H-7FH
The contents of the data memory location addressed by 7-bit immediate data mem are loaded into the accumulator.

⑤ Example: The data at memory address 6FH is loaded into the accumulator.
LADR 6FH;A ← (6FH)

LADR mem (Load A Direct): uPD7503

① Instruction code:

0	0	1	1	1	0	0	0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← (mem) mem=D_{7-0}:00H-DFH
The contents of the data memory location addressed by 8-bit immediate data mem are loaded into the accumulator.

LDEI byte (Load DE with Immediate)

① Instruction code:

0	1	0	0	1	1	1	1	I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: DE ← byte byte=I_{7-0}:00H-FFH
Eight-bit immediate data byte is loaded to register pair DE.

LHLI byte (Load HL with Immediate)

① Instruction code:

0	1	0	0	1	1	1	0	I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: HL ← byte byte= I_{7-0} :00H-FFH

Eight-bit immediate data byte is loaded to register pair HL. This instruction has a string effect. When more than one LHLI or LHLT instruction is coded successively, the second and subsequent instructions are processed as NOP instructions.

⑤ Example:

```
LDHLI:   LHLI    12H    ; *
         LHLI    34H    ;SKIP *
         LHLI    56H    ;SKIP SKIP *
         LHLI    78H    ;SKIP SKIP SKIP *
         LHLI    9AH    ;SKIP SKIP SKIP SKIP *
       ; RESULT      HL=12H,34H,56H,78H,9AH
       ;
LHLTB:   LHLT    TABL1 ; *
         LHLT    TABL2 ;SKIP *
         LHLT    TABL3 ;SKIP SKIP *
         LHLT    TABL4 ;SKIP SKIP SKIP *
         LHLI      00  ;SKIP SKIP SKIP SKIP
       ; RESULT      HL=8CH,4BH,2AH,19H
         ORG    0C0H
TABL1:   DB      8CH
TABL2:   DB      4BH
TABL3:   DB      2AH
TABL4:   DB      19H
       ;
```

Caution: * is an entry address. The value to be loaded in register pair HL varies depending on the entry address.

LHLT taddr1 (Load HL with Table data): uPD7502

① Instruction code:

1	1	0	0	P_3	P_2	P_1	P_0

② Byte count: 1

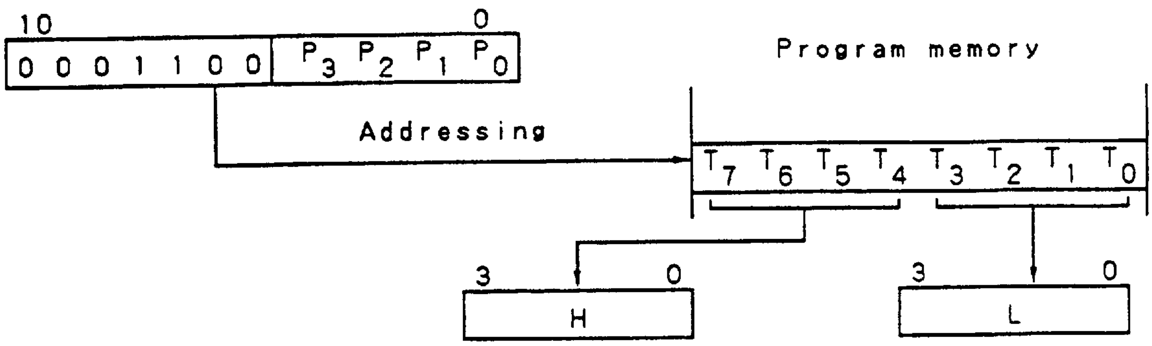
③ Machine cycle count: 2

④ Function: HL ← ROM (0001100P_3 P_2 P_1 P_0)
taddr1=0,CH,P_{3-0}:0C0H-0CFH

The high-order four bits of the table data (T_{7-0}) in the program memory location addressed by 4-bit immediate data (P_{3-0}) (the value of the high-order four bits of the address is fixed to 0CH) are loaded to the H register, and the low-order four bits of the table data are loaded in the L register. This instruction has a string effect. When more than one LHLI or LHLT instruction is coded successively, the second and subsequent instructions are processed as NOP instructions.

The program counter is not affected by the execution of this instruction. An area at addresses 0C0H to 0CFH is allocated to the table area for the LHLT instruction. Before the LHLT instruction is executed, table data must be programmed with an assembler pseudo instruction in the table area.

Only values ranging from OCOH to OCFH can be coded as taddr1.



⑤ Example: Subroutines that load table data to register pair HL

```
TCAL1:    LHLT    TABL1        ; (1)
TCAL2:    LHLT    TABL2        ; (2)
TCAL3:    LHLT    TABL3        ; (3)
          LHLI    00           ; (4)
          RT
           :
          ORG     OCOH

TABL1:    DB        8CH        ;LHLT TABLE AREA
TABL2:    DB        4BH
TABL3:    DB        2AH
```

(1) When TCAL1 is called

The H register is loaded with eight, the L register is loaded with CH, and the following instructions (2), (3), and (4) change to NOPs, then TCAL1 returns to the calling program.

(2) When TCAL2 is called

The H register is loaded with four, the L register is loaded with BH, and the following instructions (3) and (4) change to NOPs, then TCAL2 returns to the calling program.

(3) When TCAL3 is called

The H register is loaded with two, the L register is loaded with AH, and the following instruction (4) changes to NOP, then TCAL3 returns to the calling program.

LHLT taddr1 (Load HL with Table data): uPD7503

① Instruction code:

1	1	0	0	P_3	P_2	P_1	P_0

② Byte count: 1

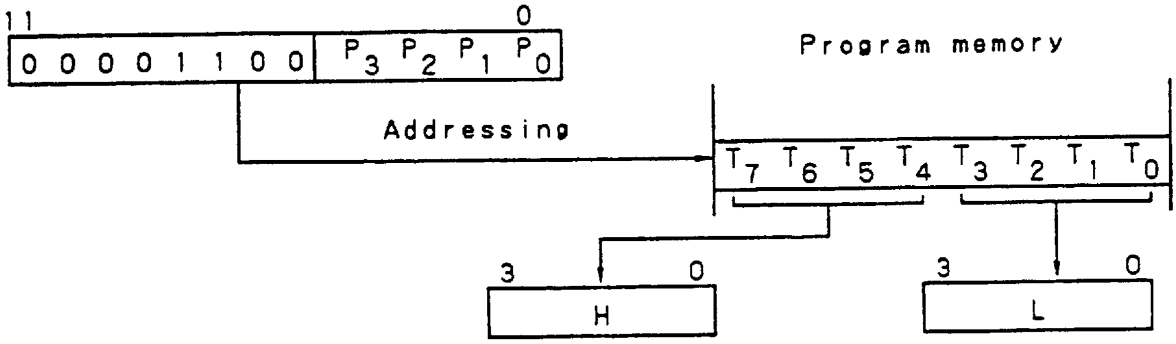
③ Machine cycle count: 2

④ Function: HL ← ROM (00001100P_3 P_2 P_1 P_0)
taddr1=0,CH,P_{3-0}:0C0H-0CFH

The high-order four bits of table data (T_{7-0}) at the program memory location addressed by 4-bit immediate data (P_{3-0}) (the higher bits of the address are fixed to 0CH) are loaded to the H register, and the low-order four bits of the table data are loaded to the L register. This instruction has a string effect. When an LHLI or LHLT instruction is coded more than once successively, the second and subsequent LHLI or LHLT instructions are processed as NOPs.

The program counter is not affected by this instruction. An area at addresses 0C0H to 0CFH is allocated to the table area for the LHLT instruction. Before an LHLT instruction is executed, table data must be programmed in this area with an assembler pseudo instruction.

Only values ranging from 0C0H to 0CFH can be coded as taddr1.



LAMT (Load A and Memory with Table data): uPD7502 only

① Instruction code:

0	1	0	1	1	1	1	0

② Byte count: 1

③ Machine cycle count: 2

④ Function: $A \leftarrow ROM(PC_{10-6}, 0, C, A)_H$

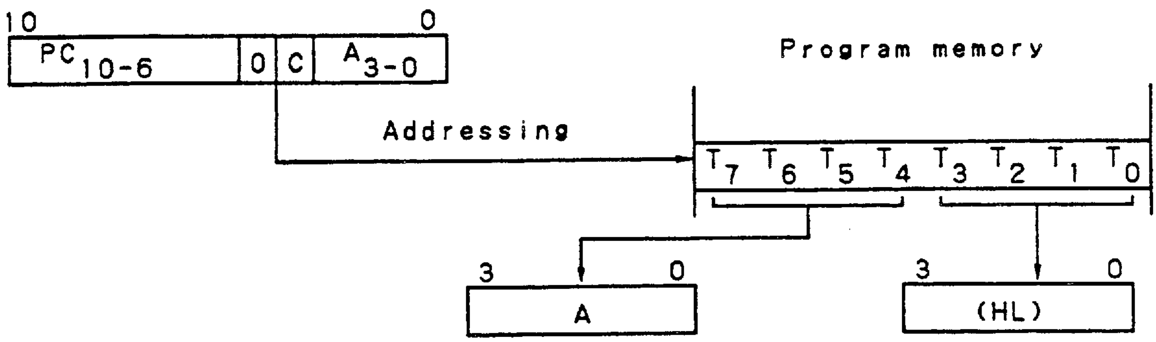
$(HL) \leftarrow ROM(PC_{10-6}, 0, C, A)_L$

The high-order four bits of the table data (T_{7-0}) at a program memory location are loaded to the accumulator, and the low-order four bits of the table data are loaded in the data memory location addressed by register pair HL. The program memory location of the table data is addressed by the high-order five bits (PC_{10-6}) of the program counter, the carry flag (C), and the contents of the accumulator (A) (bit 5 of the ROM address is always 0).

Before LAMT is executed, necessary table data must be programmed with an assembler pseudo instruction.

The program counter is not affected by the LAMT instruction.

The table area for this instruction can be placed in any current page (indicated by PC_{10-6} when the LAMP instruction is executed). Note that when the LAMT instruction is used at the last address of a page (PC_{5-0}=3FH), the high-order bits (P_{10-6}) of an address for table reference indicates the next page.



LAMTL (Load A and Memory with Table data): uPD7503 only

① Instruction code:

0	0	1	1	1	1	1	1	0	0	1	1	0	1	0	0

② Byte count: 2

③ Machine cycle count: 3

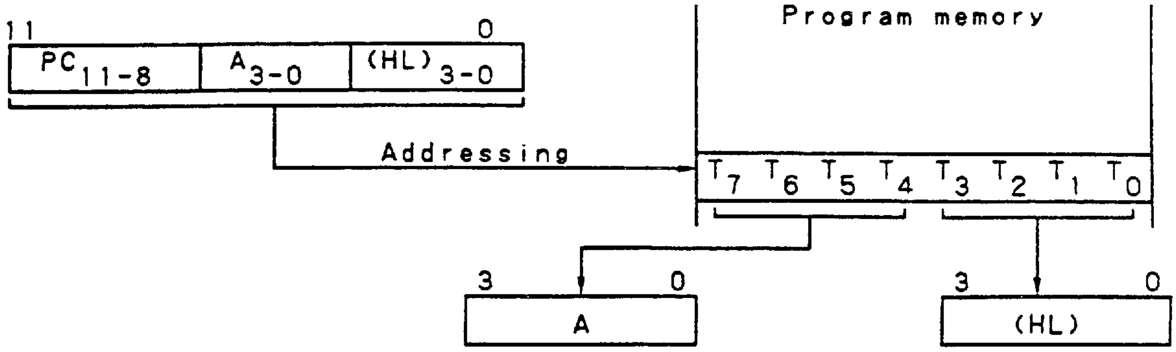
④ Function: $A \leftarrow ROM(PC_{11-8}, A_{3-0}, (HL)_{3-0})H$

$(HL) \leftarrow ROM(PC_{11-8}, A_{3-0}, (HL)_{3-0})L$

The high-order four bits of the table data (T_{7-0}) at a program memory location is loaded to the accumulator, and the low-order 4 bits are loaded to a data memory location addressed by register pair HL. The location of the table data is addressed by the contents of the data memory location addressed by the high-order four bits (PC_{11-8}) of the program counter, accumulator contents (A_{3-0}), and register pair HL.

Before the LAMTL instruction is executed, necessary table data must be programmed with an assembler pseudo instruction.

The program counter is not affected by the LAMTL instruction. Note that if the LAMTL instruction is executed with PC_{7-0} = address FFH, the value of PC_{11-8} when the LAMTL instruction is executed plus one is used to address the table area.



ST (Store A to Memory)

① Instruction code:

0	1	0	1	0	1	1	1

② Byte count: 1

③ Machine cycle count: 1

④ Function: (HL) ← A
The contents of the accumulator are stored in the data memory location addressed by register pair HL.

TAD (Transfer A to D)

① Instruction code:

0	0	1	1	1	1	1	0

1	0	1	0	1	0	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: D ← A
The contents of the accumulator are transferred to the D register.

TAE (Transfer A to E)

① Instruction code:

0	0	1	1	1	1	1	0	1	0	0	0	1	0	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: E ← A
The contents of the accumulator are transferred to the E register.

TAH (Transfer A to H)

① Instruction code:

0	0	1	1	1	1	1	0	1	0	1	1	1	0	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: H ← A
The contents of the accumulator are transferred to the H register.

TAL (Transfer A to L)

① Instruction code:

0	0	1	1	1	1	1	0	1	0	0	1	1	0	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: L ← A
The contents of the accumulator are transferred to the L register.

TDA (Transfer D to A)

① Instruction code:

0 0 1 1 1 1 1 0	1 0 1 0 1 0 1 1

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← D
The contents of the D register are transferred to the accumulator.

TEA (Transfer E to A)

① Instruction code:

0 0 1 1 1 1 1 0	1 0 0 0 1 0 1 1

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← E
The contents of the E register are transferred to the accumulator.

THA (Transfer H to A)

① Instruction code:

0	0	1	1	1	1	1	0	1	0	1	1	1	0	1	1

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← H
The contents of the H register are transferred to the accumulator.

TLA (Transfer L to A)

① Instruction code:

0	0	1	1	1	1	1	0	1	0	0	1	1	0	1	1

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← L
The contents of the L register are transferred to the accumulator.

XAD (Exchange A with D)

① Instruction code:

0	1	0	0	1	0	1	0

② Byte count: 1

③ Machine cycle count: 1

④ Function: A ⟷ D

The contents of the accumulator and D register are exchanged.

XAE (Exchange A with E)

① Instruction code:

0	1	0	0	1	0	1	1

② Byte count: 1

③ Machine cycle count: 1

④ Function: A ⟷ E

The contents of the accumulator and E register are exchanged.

XAH (Exchange A with H)

① Instruction code:

0	1	1	1	1	0	1	0

② Byte count: 1

③ Machine cycle count: 1

④ Function: A ⟷ H

The contents of the accumulator and H register are exchanged.

XAL (Exchange A with L)

① Instruction code:

0	1	1	1	1	0	1	1

② Byte count: 1

③ Machine cycle count: 1

④ Function: A ↔ L

The contents of the accumulator and L register are exchanged.

XAM pr (Exchange A with Memory)

① Instruction code:

0	1	0	R_4	0	1	R_1	R_0

② Byte count: 1

③ Machine cycle count: 1/2 (when a skip is made)

④ Function: A ↔ (pr) pr=DL,DE,HL-,HL+,HL

When pr=HL+: Skip if L=0

When pr=HL-: Skip if L=FH

pr	R_4	R_1	R_0
DL	0	0	0
DE	0	0	1
HL-	1	0	0
HL+	1	0	1
HL	1	1	0

The contents of the accumulator and the contents of the data memory location addressed by a register pair are exchanged. When HL+ is specified for pr, the data is exchanged, then the L register is incremented by one. When the L register content becomes 0, the succeeding instruction is skipped. When HL- is specified for pr, the data is exchanged, then the L register is decremented by one. When the L register content becomes FH, the succeeding instruction is skipped.

⑥ Note: XAM HL−, XAM HL+, and XAM HL can be written as XDS, XIS, and X, respectively (each having no operands).

XADR mem (Exchange A with Memory Direct): uPD7502

① Instruction code:

0	0	1	1	1	0	0	1	0	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ⟷ (mem) mem=D_{6-0}:00H-7FH
The contents of the accumulator and the contents of the data memory location addressed by 7-bit immediate data mem are exchanged.

⑤ Example: The contents of a data memory location at address 2FH are transferred to address 10H.
LADR 2FH ;A ← (2FH)
XADR 10H ;A ⟷ (10H)

XADR mem (Exchange A with Memory Direct): uPD7503

① Instruction code:

0	0	1	1	1	0	0	1	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ⟷ (mem) mem=D_{7-0}:00H-DFH
The contents of the accumulator and the contents of the data memory location addressed by 8-bit immediate data mem are exchanged.

XHDR mem (Exchange H with Memory Direct): uPD7502

① Instruction code:

0	0	1	1	1	0	1	0		0	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: H ⟷ (mem) mem=D_{6-0}:00H-7FH
The contents of the H register and the contents of the data memory location addressed by 7-bit immediate data mem are exchanged.

XHDR mem (Exchange H with Memory Direct): uPD7503

① Instruction code:

0	0	1	1	1	0	1	0		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: H ⟷ (mem) mem=D_{7-0}:00H-DFH
The contents of the H register and the contents of the data memory location addressed by 8-bit immediate data mem are exchanged.

XLDR mem (Exchange L with Memory Direct): uPD7502

① Instruction code:

0	0	1	1	1	0	1	1	0	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: L ←→ (mem) mem=D_{6-0}:00H-7FH
The contents of the L register and the contents of the data memory location addressed by 7-bit immediate data mem are exchanged.

XLDR mem (Exchange L with Memory Direct): uPD7503

① Instruction code:

0	0	1	1	1	0	1	1	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: L ←→ (mem) mem=D_{7-0}:00H-DFH
The contents of the L register and the contents of the data memory location addressed by 8-bit immediate data mem are exchanged.

6.4.2 Operation instructions

AISC n4 (Add Immediate to A, Skip if Carry)

① Instruction code:

0	0	0	0	I_3	I_2	I_1	I_0

② Byte count: 1

③ Machine cycle: 1/2 (when a skip is made)

④ Function: A ←→ A+n4:Skip if carry. $n4=I_{3-0}$:0-FH
Four-bit immediate data n4 is added to the accumulator in binary. When a carry is produced, a skip is performed. The carry flag is not affected.

⑥ Note: When n4 = 0, an NOP instruction is executed.

ASC (Add Memory to A, Skip if Carry)

① Instruction code:

0	1	1	1	1	1	0	1

② Byte count: 1

③ Machine cycle count: 1/2 (when a skip is made)

④ Function: A ← A+(HL):Skip if carry.
The contents of the data memory location addressed by register pair HL are added to the accumulator in binary. When a carry is produced as a result of the addition, a skip is performed. The carry flag is not affected.

ACSC (Add Memory to A with Carry, Skip if Carry)

① Instruction code:

0	1	1	1	1	1	0	0

② Byte count: 1

③ Machine cycle: 1/2 (when a skip is made)

④ Function: A,C ← A+(HL)+C;Skip if carry.

The contents of the data memory location addressed by register pair HL are added to the accumulator in binary (including carry flag). When a carry is produced as a result of the addition, the carry flag is set and a skip is made. When no carry is produced, the carry flag is reset, and if the following instruction is an AISC instruction, the skip condition for the AISC instruction is inhibited. (A skip is performed when a carry is produced.) Only the AISC instruction is affected.

EXL (Exclusive-Or Logic A and Memory)

① Instruction code:

0	1	1	1	1	1	1	0

② Byte count: 1

③ Machine cycle count: 1

④ Function: A ← A $\veebar$ (HL)

The contents of the accumulator are exclusive-ORed with the contents of the data memory location addressed by register pair HL, then the result is stored in the accumulator.

ANL (And Logic A and Memory)

① Instruction code:

0	0	1	1	1	1	1	1	1	0	1	1	0	0	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← A ∧ (HL)

The contents of the accumulator are ANDed with the contents of the data memory location addressed by the register pair HL, then the result is stored in the accumulator.

ORL (Or Logic A and Memory)

① Instruction code:

0	0	1	1	1	1	1	1	1	0	1	1	0	1	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← A ∨ (HL)

The contents of the accumulator are ORed with the contents of the data memory location addressed by register pair HL, then the result is stored in the accumulator.

6.4.3 Accumulator manipulation instruction

CMA (Complement A)

① Instruction code:

0	1	1	1	1	1	1	1

② Byte count: 1

③ Machine cycle count: 1

④ Function: A ← $\overline{A}$

One's complement of the accumulator contents is obtained. (Each bit is inverted.)

⑤ Example: When A=5

		3	2	1	0	
⋮	Before execution	0	1	0	1	A
CMA						
⋮	After execution	1	0	1	0	A

RAR (Rotate A Right)

① Instruction code:

0	0	1	1	1	1	1	1	1	0	1	1	0	0	1	1

② Byte count: 2

③ Machine cycle count: 2

④ Function: $A_{n-1} \leftarrow A_n$ (n=1-3)

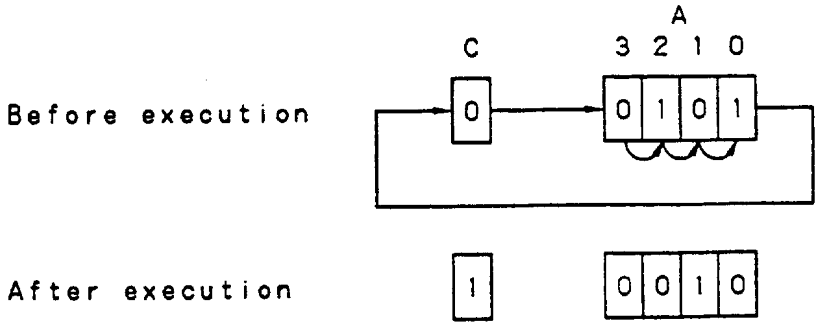
$C \leftarrow A_0$

$A_3 \leftarrow C$

The contents of the accumulator, including the carry flag, are rotated one bit to the right.

⑤ Example:

RAR



6.4.4 Carry flag manipulation instructions

RC (Reset Carry)

① Instruction code: 0 1 1 1 1 0 0 0

② Byte count: 1

③ Machine cycle count: 1

④ Function: $C \leftarrow 0$

The carry flag is reset.

SC (Set Carry)

① Instruction code: 0 1 1 1 1 0 0 1

② Byte count: 1

③ Machine cycle count: 1

④ Function: C ← 1
The carry flag is set.

6.4.5 Increment/decrement instructions

IES (Increment E, Skip if E=0)

① Instruction code:

0	1	0	0	1	0	0	1

② Byte count: 1

③ Machine cycle count: 1/2 (when a skip is made)

④ Function: E ← E+1;Skip if E=0
The E register is incremented. When the result is E=0, a skip is made.

ILS (Increment L, Skip if L=0)

① Instruction code:

0	1	0	1	1	0	0	1

② Byte count: 1

③ Machine cycle count: 1/2 (when a skip is made)

④ Function: L ← L+1;Skip if L=0
The L register is incremented. When the result is L=0, a skip is made.

IDRS mem (Increment memory Direct, Skip if memory=0):
uPD7502

① Instruction code:

0	0	1	1	1	1	0	1	0	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2/3 (when a skip is made)

④ Function: (mem) ← (mem)+1;Skip if(mem)=0

mem=D_{6-0}:00H-7FH

The contents of the data memory location addressed by 7-bit immediate data mem are incremented. When the result is 0, a skip is made.

IDRS mem (Increment memory Direct, Skip if memory=0):
uPD7503

① Instruction code:

0	0	1	1	1	1	0	1	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2/3 (when a skip is made)

④ Function: (mem) ← (mem)+1;Skip if(mem)=0

mem=D_{7-0}:00H-DFH

The contents of the data memory location addressed by 8-bit immediate data mem are incremented. When the result is 0, a skip is made.

DES (Decrement E, Skip if E=FH)

① Instruction code:

0	1	0	0	1	0	0	0

② Byte count: 1

③ Machine cycle count: 1/2 (when a skip is made)

④ Function: E ← E-1;Skip if E=FH
The E register is decremented, and when the decrement results in E=FH, a skip is made.

DLS (Decrement L, Skip if L=FH)

① Instruction code:

0	1	0	1	1	0	0	0

② Byte count: 1

③ Machine cycle count: 1/2 (when a skip is made)

④ Function: L ← L-1;Skip if L=FH
The L register is decremented. When the decrement results in L=FH, a skip is made.

DDRS mem (Decrement memory Direct, SKip if Memory=FH):
uPD7502

① Instruction code:

0	0	1	1	1	1	0	0		0	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2/3 (when a skip is made)

④ Function: (mem) ← (mem)-1;Skip if(mem)=FH

mem=D_{6-0}:00H-7FH

The contents of the data memory location addressed by 7-bit immediate data mem are decremented. When the decrement results in FH, a skip is made.

DDRS mem (Decrement memory Direct, Skip if memory=FH):
uPD7503

① Instruction code:

0	0	1	1	1	1	0	0		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2/3 (when a skip is made)

④ Function: (mem) ← (mem)-1;Skip if (mem)=FH

mem=D_{7-0}:00H-DFH

The contents of the data memory location addressed by 8-bit immediate data mem are decremented. When the decrement results in FH, a skip is made.

6.4.6 Memory bit manipulation instructions

RMB bit (Reset Memory Bit)

① Instruction code: | 0 | 1 | 1 | 0 | 1 | 0 | B_1 | B_0 |

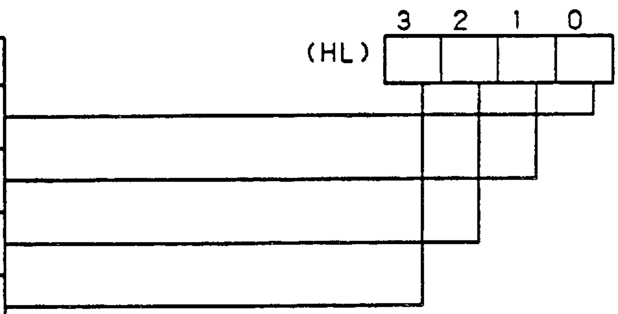
② Byte count: 1

③ Machine cycle count: 1

④ Function: (HL)bit ← 0   bit=B_{1-0}:0-3

A particular bit at the data memory location addressed by register pair HL is reset. The bit to be reset is specified by 2-bit immediate data bit.

bit	B_1	B_0
0	0	0
1	0	1
2	1	0
3	1	1



The bits other than the specified bit are left unchanged.

⑤ Example: Bit 2 at data memory address 10H is reset.

```
LHLI  10H
RMB   2              ;(10H)2 ← 0
```

	3	2	1	0	
Before execution	0	1	1	1	(10H)
After execution	0	0	1	1	(10H)

SMB bit (Set Memory Bit)

① Instruction code:

0	1	1	0	1	1	B_1	B_0

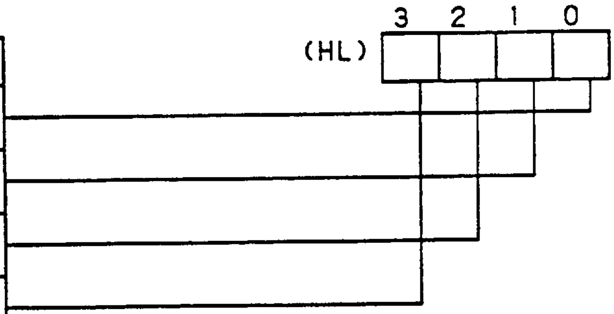
② Byte count: 1

③ Machine cycle count: 1

④ Function: (HL)bit ← 1 bit=B_{1-0}:0-3

A particular bit at the data memory location addressed by register pair HL is set. The bit to be set is specified by 2-bit immediate data bit.

bit	B_1	B_0
0	0	0
1	0	1
2	1	0
3	1	1



The bits other than the specified bit are left unchanged.

⑤ Example: Bit 1 at data memory address 11H is set.

```
LHLI   11H
SMB    1                 ;(11H)1 ← 1
```

	3	2	1	0	
Before execution	0	0	0	0	(11H)
After execution	0	0	1	0	(11H)

6.4.7 Jump instructions

JMP addr (Jump): uPD7502

① Instruction code:

0	0	1	0	0	P_{10}	P_9	P_8	P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: $PC_{10-0} \leftarrow P_{10-0}$ addr=P_{10-0}:000H-7FFH
A jump is made to the address indicated by 11-bit immediate data addr.

JMP addr (Jump): uPD7503

① Instruction code:

0	0	1	0	P_{11}	P_{10}	P_9	P_8	P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: $PC_{11-0} \leftarrow P_{11-0}$ addr=P_{11-0}:000H-FFFH
A jump is made to the address indicated by 12-bit immediate data addr.

JCP addr (Jump in the Current Page): uPD7502

① Instruction code:

1	0	P_5	P_4	P_3	P_2	P_1	P_0

② Byte count: 1

③ Machine cycle count: 1

④ Function: $PC_{5-0} \leftarrow P_{5-0}$ addr=PC_{10-6},P_{5-0}:000H-7FFH

A jump is made. The destination address is obtained by replacing the low-order six bits of the program counter (PC_{5-0}) with 6-bit immediate data P_{5-0}. The high-order five bits of the program counter (PC_{10-6}) are not affected.

A value different from the current contents of PC_{10-6} cannot be coded for addr.

⑤ Example:

```
ROM  Address
108
109       JCP    13BH   :   NON ERROR
10A
 :
13B
13C       JCP    150H   :   ERROR
 :
1FF       JCP    230H   :   NON ERROR
200
```

⑥ Note: The PC contents indicates the address next to the JCP instruction while a JCP instruction is being executed. For example, when a JCP instruction is executed at 1FFH, the PC indicates 200H, and so a jump to address 230H is allowed.

JCP addr (Jump in the Current Page): uPD7503

① Instruction code:

1	0	P_5	P_4	P_3	P_2	P_1	P_0

② Byte count: 1

③ Machine cycle count: 1

④ Function: $PC_{5-0} \leftarrow P_{5-0}$ addr=PC_{11-6} P_{5-0}:000H-FFFH

A jump is made to an address obtained by replacing the low-order six bits of the program counter (PC_{5-0}) with 6-bit immediate data P_{5-0}. The high-order six bits of the program counter (PC_{11-6}) are not affected.

A value different from the current contents of PC_{11-6} cannot be coded for addr.

JAM addr2 (Jump with A and Memory): uPD7502

① Instruction code:

0	0	1	1	1	1	1	1

0	0	0	1	0	P_2	P_1	P_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: $PC_{10-8} \leftarrow P_{2-0}$, $PC_{7-4} \leftarrow A_{3-0}$,
$PC_{3-0} \leftarrow$ (HL) addr2=P_{2-0}:0-7

The high-order three bit (PC_{10-8}) of the program counter are replaced by 3-bit immediate data addr2, the intermediate four bits (PC_{7-4}) are replaced by the accumulator contents, and the low-order four bits (PC_{3-0}) are replaced by the contents of the data memory location addressed by register pair HL. A jump to the resultant address is then made. This means that the jump destination varies within the 256-address space depending on the accumulator and data memory contents.

JAM addr2 (Jump with A and Memory): uPD7503

① Instruction code:

0	0	1	1	1	1	1	1	0	0	0	1	P_3	P_2	P_1	P_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: $PC_{11-8} \leftarrow P_{3-0}$, $PC_{7-4} \leftarrow A_{3-0}$, $PC_{3-0} \leftarrow (HL)$ addr2=P_{3-0}:0-FH

The high-order four bits of the program counter (PC_{11-8}) are replaced with 4-bit immediate data addr2, the intermediate four bits of the program counter (PC_{7-4}) are replaced with the accumulator contents, and the low-order four bits (PC_{3-0}) are replaced with the contents of the data memory location addressed by register pair HL. A jump to the resultant address is then made. This means that the jump destination varies within the 256-address space depending on the accumulator and data memory contents.

6.4.8 Subroutine stack control instructions

CALL caddr (Call Subroutine): uPD7502

① Instruction code:

0	0	1	1	0	P_{10}	P_9	P_8	P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: $(SP-1) \leftarrow PC_{7-4}$, $(SP-2) \leftarrow PC_{3-0}$,
$(SP-3) \leftarrow PSW$, $(SP-4) \leftarrow 0, PC_{10-8}$
$PC_{10-0} \leftarrow P_{10-0}$, $SP \leftarrow SP-4$
$caddr = P_{10-0}$: 000H-7FFH

The contents of the program counter (return address) and PSW are saved in the data memory location (stack) addressed by the stack pointer (SP), then a jump is made to the address indicated by 11-bit immediate data caddr.

CALL caddr (Call Subroutine): uPD7503

① Instruction code:

0	0	1	1	0	P_{10}	P_9	P_8	P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: (SP-1) ← PC_{7-4}, (SP-2) ← PC_{3-0},
(SP-3) ← PSW, (SP-4) ← PC_{11-8},
PC_{11-0} ← 0, P_{10-0}, SP ← SP-4
caddr=P_{10-0}:000H-7FFH

The contents of the program counter (return address) and PSW are saved in the data memory location (stack) addressed by the stack pointer (SP), then a jump is made to the address indicated by 11-bit immediate data caddr.

CALT taddr2 (Call Table): uPD7502

① Instruction code:

1	1	P_5	P_4	P_3	P_2	P_1	P_0

② Byte count: 1

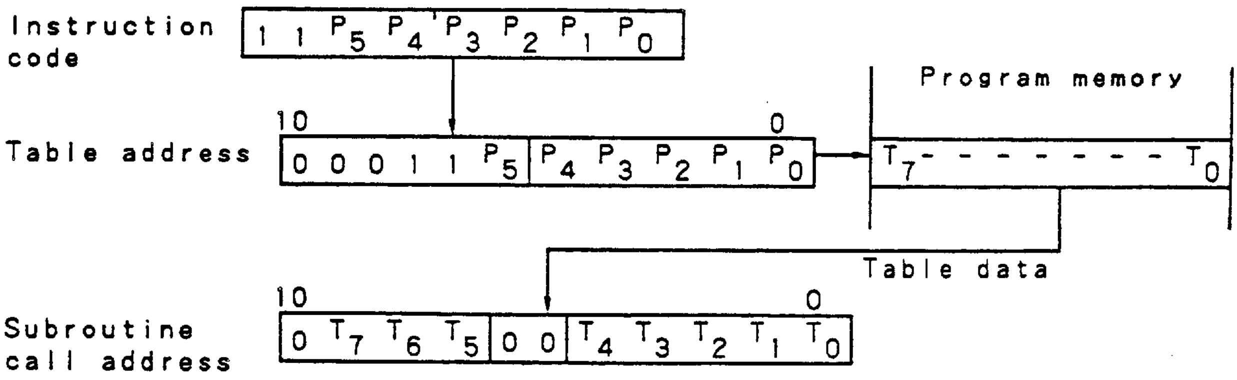
③ Machine cycle count: 2

④ Function: (SP-1) ← PC_{7-4}, (SP-2) ← PC_{3-0},
(SP-3) ← PSW, (SP-4) ← 0, PC_{10-8}
$PC_{9-7,4-0}$ ← ROM(00011$P_5P_4P_3P_2P_1P_0$)
$PC_{11,10,6,5}$ ← 0
SP ← SP-4
taddr2=00011$P_5P_4P_3P_2P_1P_0$:0D0H-0FFH

The contents of the program counter (return address) and PSW are saved in the data memory location (stack) addressed by the stack pointer (SP), 8-bit table data (T_{7-0}) at the table address (00011$P_5P_4P_3P_2$ P_1P_0) specified in the instruction is loaded into the program counter in form of 0$T_7T_6T_5$00$T_4T_3T_2T_1T_0$, then a jump is made.

Be sure to code a table address in the operand of the CALT instruction.

The table area for the CALT instruction is assigned to addresses 0D0H to 0FFH. Before the CALT instruction is executed, data for the subroutine call address must be programmed in this address with an assembler pseudo instruction (DET).



⑤ Example:

```
;
            ORG         0D0H
XZSOFF:     DET         ZSOFF
XZNAAD:     DET         ZNAAD
XZNDAD:     DET         ZNDAD
XZTMAW:     DET         ZTMAW
XZNOAM:     DET         ZNOAM
                :
;
MAIN0:      LAI         0FH
            CALT        XZSOFF      ;CALL TABLE
                :
MAIN1:      LHLT        TABLE
            CALT        XZNOAM      ;CALL TABLE
                :
```

CALT taddr2 (Call Table): uPD7503

① Instruction code: | 1 1 P_5 P_4 P_3 P_2 P_1 P_0 |

② Byte count: 1

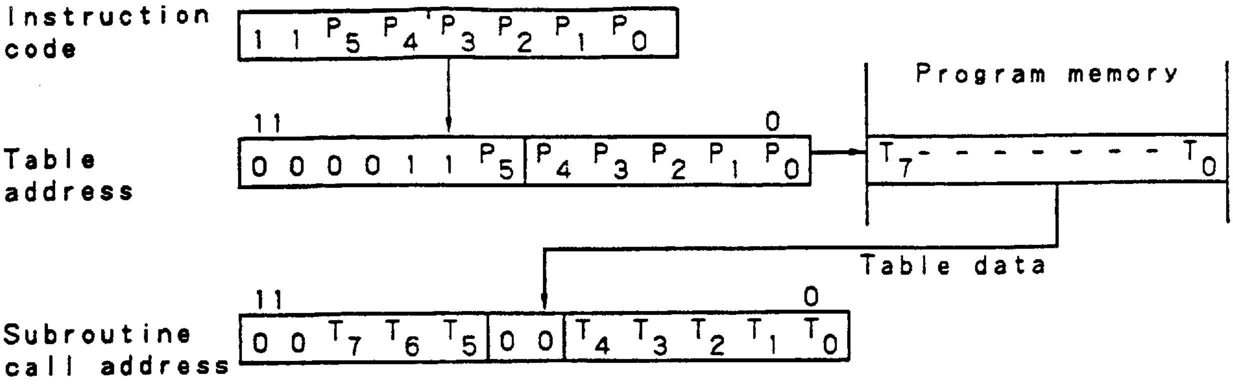
③ Machine cycle count: 2

④ Function: $(SP-1) \leftarrow PC_{7-4}$, $(SP-2) \leftarrow PC_{3-0}$,
$(SP-3) \leftarrow PSW$, $(SP-4) \leftarrow PC_{11-8}$
$PC_{9-7,4-0} \leftarrow ROM(000011P_5P_4P_3P_2P_1P_0)$
$PC_{11,10,6,5} \leftarrow 0$
$SP \leftarrow SP-4$
$taddr2=000011P_5P_4P_3P_2P_1P_0$:0D0H-0FFH

The contents of the program counter (return address) and PSW are saved in the data memory location (stack) addressed by the stack pointer (SP), 8-bit table data (T_{7-0}) at the table address ($000011P_5P_4P_3P_2P_1P_0$) specified in the instruction is loaded into the program counter in form of $00T_7T_6T_500T_4T_3T_2T_1T_0$, then a jump is made.

Be sure to code a table address in the operand of the CALT instruction.

The table area for the CALT instruction is assigned to addresses 0D0H to 0FFH. Before the CALT instruction is executed, the subroutine call address data must be programmed in this area with an assembler pseudo instruction (DET).



RT (Return from subroutine): uPD7502

① Instruction code:

0	1	0	1	0	0	1	1

② Byte count: 1

③ Machine cycle count: 2

④ Function: $PC_{10-8} \leftarrow (SP)$, $PC_{3-0} \leftarrow (SP+2)$, $PC_{7-4} \leftarrow (SP+3)$, $SP \leftarrow SP+4$

The contents of the data memory location (stack) addressed by the stack pointer (SP) are restored in the program counter (PC), then SP is incremented. This instruction is used to return a subroutine when the PSW need not be restored.

RT (Return from subroutine): uPD7503

① Instruction code:

0	1	0	1	0	0	1	1

② Byte count: 1

③ Machine cycle count: 2

④ Function: $PC_{11-8} \leftarrow (SP)$, $PC_{3-0} \leftarrow (SP+2)$,
$PC_{7-4} \leftarrow (SP+3)$, $SP \leftarrow SP+4$

The contents of the data memory location (stack) addressed by the stack pointer (SP) are restored in the program counter (PC), then SP is incremented.

⑥ Note: PSW is not restored.

RTS (Return from Subroutine, then Skip): uPD7502

① Instruction code:

0	1	0	1	1	0	1	1

② Byte count: 1

③ Machine cycle count: 3

④ Function: $PC_{10-8} \leftarrow (SP)_{2-0}$, $PC_{3-0} \leftarrow (SP+2)$,
$PC_{7-4} \leftarrow (SP+3)$, $SP \leftarrow SP+4$
Then skip unconditionally

The contents of the data memory location (stack) addressed by the stack pointer (SP) are restored in the program counter (PC), SP is incremented, then skip is unconditionally made.

⑤ Example: Return processing is performed in one of the following two ways, which depends on the subroutine processing result:

```
;MAIN ROUTINE
        CALL  SUBRT

    →  [INSTA]

    →  [INSTB]
          :
;SUBROUTINE
 SUBRT  :  .
           .
           .
        SKC         ;SKIP IF CARRY
        RT          ;RETURN
        RTS         ;RETURN & SKIP
```

If the subroutine processing result carry flag is set, control is returned to INSTB. If the flag is not set, control is returned to INSTA.

⑥ Note: PSW is not returned.

RTS (Return from subroutine, then Skip): uPD7503

① Instruction code:

0	1	0	1	1	0	1	1

② Byte count: 1

③ Machine cycle count: 3

④ Function: $PC_{11-8} \leftarrow (SP)_{3-0}$, $PC_{3-0} \leftarrow (SP+2)$,
$PC_{7-4} \leftarrow (SP+3)$,
$SP \leftarrow SP+4$
Then skip unconditionally

The contents of the data memory location (stack) addressed by the stack pointer (SP) are restored in the program counter (PC), SP is incremented, then skip is unconditionally made.

⑥ Note: PSW is not restored.

RTPSW (Return from subroutine and restore PSW): uPD7502

① Instruction code:

0	1	0	0	0	0	1	1

② Byte count: 1

③ Machine cycle count: 2

④ Function: $PC_{10-8} \leftarrow (SP)_{2-0}$, $PSW \leftarrow (SP+1)$,
$PC_{3-0} \leftarrow (SP+2)$, $PC_{7-4} \leftarrow (SP+3)$
$SP \leftarrow SP+4$
The contents of the data memory location (stack) addressed by the stack pointer (SP) are restored in the program counter (PC) and program status word (PSW), then SP is incremented.

RTPSW (Return from subroutine and restore PSW): uPD7503

① Instruction code:

0	1	0	0	0	0	1	1

② Byte count: 1

③ Machine cycle count: 2

④ Function: $PC_{11-8} \leftarrow (SP)_{3-0}$, $PSW \leftarrow (SP+1)$,
$PC_{3-0} \leftarrow (SP+2)$, $PC_{7-4} \leftarrow (SP+3)$
$SP \leftarrow SP+4$
The contents of the data memory location (stack) addressed by the stack pointer (SP) are restored in the program counter (PC) and program status word (PSW), then SP is incremented.

PSHDE (Push DE on stack)

① Instruction code:

0	0	1	1	1	1	1	0	1	0	0	0	1	1	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: (SP-1) ← D, (SP-2) ← E

SP ← SP-2

The contents of the D and E registers are saved in the data memory location (stack) addressed by the stack pointer (SP).

PSHHL (Push HL on stack)

① Instruction code:

0	0	1	1	1	1	1	0	1	0	0	1	1	1	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: (SP-1) ← H, (SP-2) ← L

SP ← SP-2

The contents of the H and L registers are saved in the data memory location (stack) addressed by the stack pointer (SP).

POPDE (Pop DE off stack)

① Instruction code:

0	0	1	1	1	1	1	0	1	0	0	0	1	1	1	1

② Byte count: 2

③ Machine cycle count: 2

④ Function: E ← (SP), D ← (SP+1)
SP ← SP+2
The contents of the data memory location (stack) addressed by the stack pointer (SP) are restored in the D and E registers, then SP is incremented.

POPHL (Pop HL off stack)

① Instruction code:

0	0	1	1	1	1	1	0	1	0	0	1	1	1	1	1

② Byte count: 2

③ Machine cycle count: 2

④ Function: L ← (SP), H ← (SP+1)
SP ← SP+2
The contents of the data memory location (stack) addressed by the stack pointer (SP) are restored in the H and L registers, then SP is incremented.

TAMSP (Transfer A and Memory to SP)

① Instruction code:

0	0	1	1	1	1	1	1	0	0	1	1	0	0	0	1

② Byte count: 2

③ Machine cycle count: 2

④ Function: $SP_{7-4} \leftarrow A$, $SP_{3-1} \leftarrow (HL)_{3-1}$, $SP_0 \leftarrow 0$

The contents of the accumulator are transferred to the high-order four bits of the stack pointer, and the high-order three bits in the data memory location addressed by register pair HL are transferred to the low-order three bits (SP_{3-1}) of the stack pointer.

SP_0 is automatically loaded with 0.

TSPAM (Transfer SP to A and Memory)

① Instruction code:

0	0	1	1	1	1	1	1	0	0	1	1	0	1	0	1

② Byte count: 2

③ Machine cycle count: 2

④ Function: $A \leftarrow SP_{7-4}$, $(HL)_{3-1} \leftarrow SP_{3-1}$, $(HL)_0 \leftarrow 0$

The high-order four bits of the stack pointer (SP_{7-4}) are transferred to the accumulator, and the low-order three bits of the stack pointer (SP_{3-1}) are transferred to the high-order three bits of the data memory location addressed by register pair HL. The least significant bit in the data memory location is automatically set to 0.

6.4.9 Skip instructions

SKC (Skip if Carry)

① Instruction code:

0	1	0	1	1	0	1	0

② Byte count: 1

③ Machine cycle count: 1/2 (when a skip is made)

④ Function: Skip if C=1

If the carry flag is 1, a skip is made.

SKMBT bit (Skip if Memory Bit True)

① Instruction code:

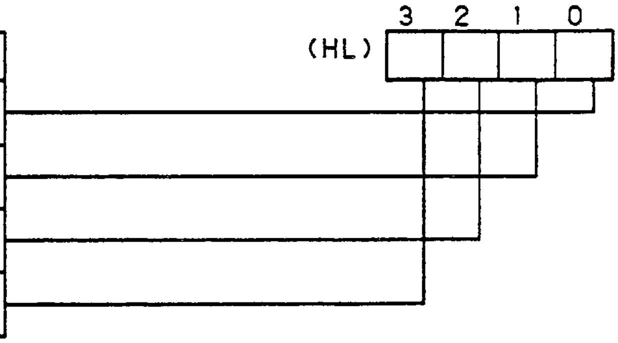
0	1	1	0	0	1	B_1	B_0

② Byte count: 1

③ Machine cycle count: 1/2 (when a skip is made)

④ Function: Skip if (HL)bit=1 bit=B_{1-0}:0-3
A skip is made if a particular bit in the data memory location addressed by register pair HL is 1. The bit is specified by 2-bit immediate data bit.

bit	B_1	B_0
0	0	0
1	0	1
2	1	0
3	1	1



SKABT bit (Skip if A Bit True)

① Instruction code:

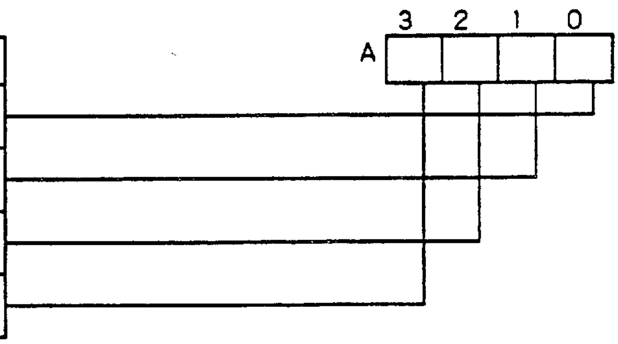
0	1	1	1	0	1	B_1	B_0

② Byte count: 1

③ Machine cycle count: 1/2 (when a skip is made)

④ Function: Skip if Abit=1 bit=B_{1-0}:0-3
A skip is made if a particular bit in the accumulator is 1. The bit is specified by 2-bit immediate data bit.

bit	B_1	B_0
0	0	0
1	0	1
2	1	0
3	1	1



SKMBF bit (Skip if Memory Bit False)

① Instruction code: | 0 | 1 | 1 | 0 | 0 | 0 | B_1 | B_0 |

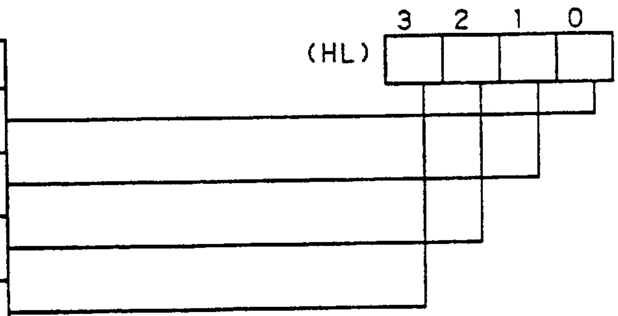
② Byte count: 1

③ Machine cycle count: 1/2 (when a skip is made)

④ Function: Skip if (HL)bit=0 bit=B_{1-0}:0-3

A skip is made if a particular bit in the data memory location addressed by register pair HL is 0. The bit is specified by 2-bit immediate data bit.

bit	B_1	B_0
0	0	0
1	0	1
2	1	0
3	1	1



⑤ Example: Whether bit 3 of data memory (10H) is 0 is tested.

```
    LHLI   10H  ;HL=10H
    SKMBF  3    ;SKIP IF (10H)3=0
    JMP    ERR  ;JUMP TO ERR
OK:
```

SKAEM (Skip if A Equals Memory)

① Instruction code: | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |

② Byte count: 1

③ Machine cycle count: 1/2 (when a skip is made)

④ Function: Skip if A=(HL)
A skip is made if the contents of the accumulator match the contents of the data memory location addressed by register pair HL.

⑤ Example: The number of data 5 existing in data memory (10H-1FH) is recorded in data memory (20H).

```
COMP  :LAI   0
       XADR 20H; CLEAR COUNTER
       LHLI 1FH; HL=1F
       LAI   5
LOOP  :SKAEM     ; SKIP IF (HL)=5
       JCP   BR
       IDRS 20H; INCREMENT COUNTER
  BR  :DLS       ; DECREMENT DATA POINTER
       JCP   LOOP
END   :
```

The match counter (20H) is cleared, 5 is set in the accumulator, then the contents of data memory and the accumulator are compared by an SKAEM instruction while the data memory address is being updated. When they match, the match counter is incremented.

SKAEI n4 (Skip if A Equals Immediate)

① Instruction code:

0	0	1	1	1	1	1	1	0	1	1	0	I_3	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2/3 (when a skip is made)

④ Function: Skip if A=n4 $n4=I_{3-0}$:0-FH
A skip is made if the contents of the accumulator match 4-bit immediate data n4.

SKDEI n4 (Skip if D Equals Immediate)

① Instruction code:

0	0	1	1	1	1	1	0	0	1	1	0	I_3	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2/3 (when a skip is made)

④ Function: Skip if D=n4 $n4=I_{3-0}$:0-FH
A skip is made if the contents of the D register match 4-bit immediate data n4.

SKEEI n4 (Skip if E Equals Immediate)

① Instruction code:

0	0	1	1	1	1	1	0	0	1	0	0	I_3	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2/3 (when a skip is made)

④ Function: Skip if E=n4 $n4=I_{3-0}$:0-FH
A skip is made if the contents of the E register match 4-bit immediate data n4.

SKHEI n4 (Skip if H Equals Immediate)

① Instruction code:

0	0	1	1	1	1	1	0	0	1	1	1	I_3	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2/3 (when a skip is made)

④ Function: Skip if H=n4 n4=I_{3-0}:0-FH
A skip is made if the contents of the H register match 4-bit immediate data n4.

SKLEI n4 (Skip if L Equals Immediate)

① Instruction code:

0	0	1	1	1	1	1	0	0	1	0	1	I_3	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2/3 (when a skip is made)

④ Function: Skip if L=n4 n4=I_{3-0}:0-FH
A skip is made if the contents of the L register match 4-bit immediate data n4.

6.4.10 SIO control instructions

TAMSIO (Transfer A and Memory to SIO)

① Instruction code:

0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: $SIO_{7-4} \leftarrow A$, $SIO_{3-0} \leftarrow (HL)$
The contents of the accumulator are stored in the high-order four bits of the shift register (SIO_{7-4}), and the contents of the data memory location addressed by register pair HL are stored in the low-order four bits (SIO_{3-0}) of the shift register.

TSIOAM (Transfer SIO to A and Memory)

① Instruction code:

0	0	1	1	1	1	1	1	0	0	1	1	1	0	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← SIO_{7-4}, (HL) ← SIO_{3-0}

The high-order four bits of the shift register (SIO_{7-4}) are transferred to the accumulator, and the low-order four bits (SIO_{3-0}) are transferred to the data memory location addressed by register pair HL.

SIO (Start SIO)

① Instruction code:

0	0	1	1	1	1	1	1	0	0	1	1	0	0	1	1

② Byte count: 2

③ Machine cycle count: 2

④ Function: 3BIT CNT ← 0, INT0/S RQF ← 0

The 3-bit counter of the serial interface is cleared, INT0/S request flag is reset (when INTS is selected by the shift mode register), then shift operation is started.

6.4.11 Timer control instructions

TAMMOD (Transfer A and Memory to timer Modulo register)

① Instruction code:

0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	1

② Byte count: 2

③ Machine cycle count: 2

④ Function: $MOD_{7-4} \leftarrow A$, $MOD_{3-0} \leftarrow (HL)$

The contents of the accumulator are transferred to the high-order four bits of the modulo register (MOD_{7-4}), and the contents of the data memory location addressed by register pair HL are transferred to the low-order four bits (MOD_{3-0}).

TIMER (Start Timer)

① Instruction code:

0	0	1	1	1	1	1	1	0	0	1	1	0	0	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: $CT \leftarrow 0$, $INTT\ RQF \leftarrow 0$

The count register (CT) is cleared, the INTT request flag is reset, then the timer operation is started.

TCNTAM (Transfer timer Count register to A and Memory)

① Instruction code:

0	0	1	1	1	1	1	1	0	0	1	1	1	0	1	1

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← CT_{7-4}, (HL) ← CT_{3-0}

The high-order four bits (CT_{7-4}) of the count register are transferred to the accumulator. The low-order four bits (CT_{3-0}) of the count register are transferred to the data memory location addressed by register pair HL.

The count operation is suspended during execution of this instruction.

6.4.12 Interrupt control instructions

EI n3 (Enable Interrupt)

① Instruction code:

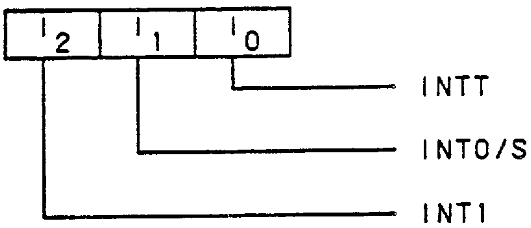
0	0	1	1	1	1	1	1

1	0	0	1	0	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: Interrupt enable register ← interrupt enable register ∨ n3 (n3≠0)
Interrupt master enable F/F ← 1 (n3=0)
$n3=I_{2-0}$:0-7



The contents of the interrupt enable register are ORed with 3-bit immediate data n3, and the result is stored in the interrupt enable register. This means that by setting the bit specified for a desired interrupt to 1, the associated one or more interrupts are enabled.

If n3 is 0, the contents of the interrupt enable register remain unchanged, but the interrupt master enable F/F is set, enabling an interrupt.

DI n3 (Disable Interrupt)

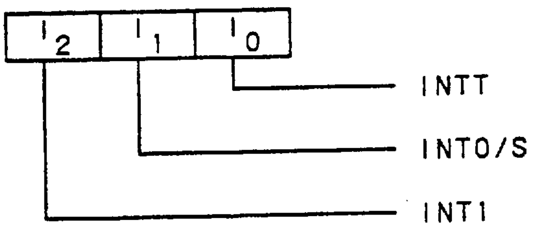
① Instruction code:

0	0	1	1	1	1	1	1	1	0	0	0	0	I_2	I_1	I_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: Interrupt enable register ← interrupt enable register $\wedge$ $\overline{n3}$ (n3≠0)
Interrupt master enable F/F ← 0 (n3=0)
$n3 = I_{2-0}$:0-7



The contents of the interrupt enable register are ANDed with the complement of 3-bit immediate data n3, and the result is stored in the interrupt enable register. This means that by setting the bit associated with a desired interrupt, one or more interrupts are disabled.

If n3 is 0, the contents of the interrupt enable register remain unchanged, but the interrupt master enable F/F is reset, which disables the acceptance of all interrupts.

SKI n3 (Skip if Interrupt request flag is true and reset interrupt request flag)

① Instruction code:

0	0	1	1	1	1	1	1	0	1	0	0	0	I_2	I_1	I_0

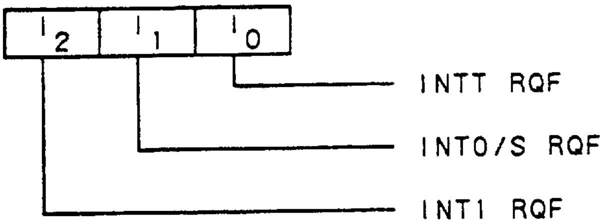
② Byte count: 2

③ Machine cycle count: 2/3 (when a skip is made)

④ Function: Skip if INT RQF $\wedge$ n3≠0

INT RQF $\leftarrow$ INT RQF $\wedge$ $\overline{n3}$ n3=I_{2-0}:0-7

The interrupt request flag associated with the bit set to 1 is tested with 3-bit immediate data n3. If the test shows that the interrupt request flag contains 1, the following instruction is skipped, and the tested interrupt request flag is reset.



6.4.13 I/O instructions

IPL (Input Port specified by L)

① Instruction code:

0	1	1	1	0	0	0	0

② Byte count: 1

③ Machine cycle count: 1

④ Function: A ← Port(L)

Contents of L register	0	1	4	5	6
Port name	Port 0	Port 1	Port 4	Port 5	Port 6

The contents of the sport specified by the L register are loaded to the accumulator.

IP addr3 (Input Port specified by immediate)

① Instruction code:

0	0	1	1	1	1	1	1

1	1	0	0	0	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← Port(addr3) addr3=D_{2-0}:0,1,4-6

addr3	0	1	4	5	6
Port name	Port 0	Port 1	Port 4	Port 5	Port 6

The contents of the port specified by 3-bit immediate data addr3 are loaded to the accumulator.

IP1 (Input Port1)

① Instruction code: `0 1 1 1 0 0 0 1`

② Byte count: 1

③ Machine cycle count: 1

④ Function: A ← Port1

The contents of port 1 are loaded to the accumulator.

IP54 (Input Port 5 and 4)

① Instruction code:

`0 0 1 1 1 1 1 1` `0 0 1 1 1 0 0 0`

② Byte count: 2

③ Machine cycle count: 2

④ Function: A ← Port5, (HL) ← Port4

The contents of port 5 are loaded to the accumulator, and the contents of port 4 are stored in the data memory location addressed by register pair HL.

OPL (Output Port specified by L)

① Instruction code:

0	1	1	1	0	0	1	0

② Byte count: 1

③ Machine cycle count: 1

④ Function: Port/Mode register(L) ← A

Contents of L register	3	4	5	6	BH	CH	EH	FH
Port/mode register	Port 3	Port 4	Port 5	Port 6	Display mode register	Clock mode register	Port 6 mode register	Shift mode register

The contents of the accumulator are output to the port or stored in the mode register specified by the L register.

OP addr1 (Output Port specified by immediate)

① Instruction code:

0	0	1	1	1	1	1	1

1	1	1	0	D_3	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: Port/Mode register(addr1) ← A
addr1=D_{3-0}:3-6,BH,CH,EH,FH

addr1	3	4	5	6	BH	CH	EH	FH
Port/mode register	Port 3	Port 4	Port 5	Port 6	Display mode register	Clock mode register	Port 6 mode register	Shift mode register

The contents of the accumulator are output to the port or stored in the mode register specified by 4-bit immediate data addr1.

OP3 (Output Port3)

① Instruction code: | 0 1 1 1 | 0 0 1 1 |

② Byte count: 1

③ Machine cycle count: 1

④ Function: Port3 ← A
The contents of the accumulator are output to port 3.

OP54 (Output Port 5 and 4)

① Instruction code:

| 0 0 1 1 | 1 1 1 1 | | 0 0 1 1 | 1 1 0 0 |

② Byte count: 2

③ Machine cycle count: 2

④ Function: Port5 ← A, Port4 ← (HL)

The contents of the accumulator are output to port 5, and the contents of the data memory location addressed by register pair HL are output to port 4.

ANP addr4,n4 (And Port with immediate)

① Instruction code:

0	1	0	0	1	1	0	0

I_3	I_2	I_1	I_0	0	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: Port(addr4) ← Port(addr4) $\wedge$ n4

addr4=D_{2-0}:3-6

n4=I_{3-0}:0-FH

addr4	3	4	5	6
Port name	Port 3	Port 4	Port 5	Port 6

The contents of the port specified by 3-bit immediate data addr4 are ANDed with 4-bit immediate data n4, and the result is output to the port specified by addr4.

ORP addr4,n4 (Or Port with immediate)

① Instruction code:

0	1	0	0	1	1	0	1

I_3	I_2	I_1	I_0	0	D_2	D_1	D_0

② Byte count: 2

③ Machine cycle count: 2

④ Function: Port(addr4) ← Port(addr4) ∨ n4

addr4=D_{2-0}:3-6

n4=I_{3-0}:0-FH

addr4	3	4	5	6
Port name	Port 3	Port 4	Port 5	Port 6

The contents of the port specified by 3-bit immediate data addr4 are ORed with 4-bit immediate data n4, and the result is output to the port specified by addr4.

6.4.14 CPU control instructions

HALT (Halt)

① Instruction code:

0	0	1	1	1	1	1	1

0	0	1	1	0	1	1	0

② Byte count: 2

③ Machine cycle count: 2

④ Function: The halt mode is set.

STOP (Stop)

① Instruction code:

0	0	1	1	1	1	1	1		0	0	1	1	0	1	1	1

② Byte count: 2

③ Machine cycle count: 2

④ Function: The stop mode is set.

NOP (No Operation)

① Instruction code:

0	0	0	0	0	0	0	0

② Byte count: 1

③ Machine cycle count: 1

④ Function: One machine cycle is used for doing nothing.

6.5 List of Port and Mode Register Manipulation Instructions

	Port 0	Port 1	Port 3	Port 4	Port 5	Port 6	Port 6 mode register	Shift mode register	Display mode register	Clock mode register
OP addr1			o	o	o	o	o	o	o	o
OPL			o	o	o	o	o	o	o	o
IP addr3	o	o		o	o	o				
IPL	o	o		o	o	o				
ANP addr4,n4			o	o	o	o				
ORP addr4,n4			o	o	o	o				
IP54				o						
OP54				o						
IP1		o								
OP3			o							

o: Available

6.6 Machine Cycles Required for Skip Operation

If the condition of a skip instruction is met, the next instruction is skipped in one machine cycle regardless of the number of machine cycles required for the execution of this instruction.

This is equivalent to replacing the next instruction with one NOP instruction.

6.7 Using Register Pair (DL, DE, HL) as a Data Pointer

Before an instruction which uses a register pair (DL, DE or HL) as a data pointer to access data memory is executed, memory address information is set in the register pair. In this case, the address to be set must be located within the memory area (00H to 7FH for the uPD7502; 00H to DFH for the uPD7503).

Correct:

```
LHLI   50H ;HL=50H
LAM    HL  ;A ← (50H)
```

Wrong: For the uPD7502

```
LDEI   80H ;DE=80H
LAM    DE  ;A is undefined.
```