FOREWORD

Since the release of the PC-1500 on market, we have had great number of questions from users regarding the machine language of the PC-1500.

To meet with such demand from ardent users, we are now sending this text for study of the machine language of the Sharp's original design LH5801 Microprocessor and LH5811 Peripheral Control LSI in concern with the PC-1500 system. Because the text is edited on the basis of user questions, it may not support quality as a guidebook. In such an event, you are suggested to make reference to microprocessor guidebooks published on market, in addition to this text.

Your opinions and questions are welcome through our products distributor.

NOTE: Machine language program, which controls hardware directly, gives you more various functions than BASIC programs. However, you should check your machine language program enough to make no error before executing it because single wrong key operation may upset the program or occasionally make the machine break down.

Sharp Corporation assumes no liability or responsibility of any kind arising from the use of programs or program materials or any part thereof.

promsoft.com

CONTENTS

1. Machine Language 1

2. LH5801 Microprocessor 5
   2-1. Outline of LH5801 6
   2-2. Internal structure 7
      2-2-1. Block diagram 7
      2-2-2. Internal registers 8
      2-2-3. Status flags 8
      2-2-4. CPU pin description 9
   2-3. Functions 14
      2-3-1. Timer 14
      2-3-2. Interrupts 17
      2-3-3. Reset 20
      2-3-4. CPU system sequence 21
      2-3-5. BF flipflop 22
      2-3-6. WAIT function 23
   2-4. LH5801 instructions 24
      2-4-1. Outline 24
      2-4-2. Add, subtract, and logical instructions 25
      2-4-3. Compare and bit test 31
      2-4-4. Transfer and search instructions 33
      2-4-5. Block transfer and search instructions 38
      2-4-6. Rotate and shift instructions 39
      2-4-7. CPU control instructions 42
      2-4-8. Jump instructions 45
      2-4-9. Subroutine jump instructions 49
      2-4-10. Return instructions 53
   2-5. Command list 54
   2-6. Electrical characteristics and timings 63

3. LH5810/LH5811 I/O port controller 67
   3-1. Outline 68
   3-2. Functions 68
   3-3. Internal structure 70
      3-3-1. Block diagram 70
      3-3-2. Internal registers 70
      3-3-3. Pin description 73
   3-4. Functions 74
      3-4-1. Operation 74
      3-4-2. Wait control 75
      3-4-3. Serial data input 76
      3-4-4. Reset 77
   3-5. Specification 78
      3-5-1. I/O port controller input/output circuits 78
      3-5-2. Pad layout and structure 79
      3-5-3. Electrical characteristics 79

promsoft.com

4. PC-1500 hardware description 85
4-1. PC-1500 system configuration 86
4-1-1. Outline 86
4-1-2. Block diagram 87
4-1-3. Power supplies (PC-1500, CE-150, CE-158, CE-159) 87
4-2. PC-1500 89
4-2-1. Outline 89
4-2-2. Block diagram 90
4-2-3. Chip select circuit 91
4-2-4. PC-1500 system memory map 94
4-3. Connector signals/LSI signals 102
4-3-1. 40-pin connector 102
4-3-2. 60-pin connector 103
4-3-3. LH5801 Microprocessor 104
4-3-4. I/O PC 105
4-4. Key matrix and key code chart 109

5. PC-1500 software 111
5-1. BASIC command related PC-1500 machine language 112
5-1-1. NEW 112
5-1-2. STATUS 112
5-1-3. PEEK 113
5-1-4. POKE 113
5-1-5. CALL 114
5-1-6. CSAVE M 114
5-1-7. CLOAD M 114
5-2. Internal code chart 115
5-3. Expression of variable and program 116
5-3-1. Expression of decimal number 116
5-3-2. Expression of binary number 116
5-3-3. Expression of character string 116
5-3-4. Structure of variable name 117
5-3-5. Structure of program 117
5-3-6. Structure of reserve area 118
5-4. System subroutines 120
5-4-1. Character functions 122
5-4-2. Arithmetic subroutines 127
5-4-3. Comparison 128
5-4-4. Search 129
5-4-5. Display 131
5-4-6. Printer 135
5-4-7. Cassette tape 138
5-4-8. Caution for system subroutine call 141

6. Machine language programming examples 143
6-1. Binary to hexadecimal conversion 144
6-2. Display inversion 145
6-3. Single display dot left shift 146
6-4. Single display dot right shift 147
6-5. Conversion of USING format expressed numerical data into character string 148
6-6. Power off that does not activate the printer upon power on 148

promsoft.com

[REFERENCE]

1. Determining printing character size and direction 150
2. Restoration of array and two-character variable 150
3. Knowing the use of CE-150 150
4. CMT format 151
5. Circuit diagram 153
5-1. PC-1500 154
5-2. CE-150 155
5-3. CE-151 156
5-4. CE-153 156
5-5. CE-155 157
5-6. CE-158 158
5-7. CE-159 159

promsoft.com

1

Machine Language

promsoft.com

1. Machine language

There are many program languages for each purpose. PC-1500, for example, is designed to carry out both BASIC and machine language. BASIC is easy to use, however, execution speed is slow. On the other hand, machine language is difficult to understand but execution speed is fast.

Usually, machine language program would be written with the assemble language, which consists of mnemonic codes, and then the assemble language will be translated into machine language.

[EXAMPLE] DISPLAY REVERSE PROGRAM

1. Prepare the program with assemble language consisting of mnemonic codes.

LDI	UH,78H	prepare for assignment of the first display buffer address.
LDI	UL,4DH	
DEC	UH	advance the address
LDA	U	take data in the accumulator
EAI	FFH	take the complement
STA	U	return data into memory
LOP	06H	make the loop
CPI	UH,77H	
BCS	−0EH	
RTN		return to BASIC

2. Translate the above program into machine language. The assembler translates the assemble language into the machine language automatically according to the list. However, a short program can be translated manually. (hand assemble)
   The above program can be translated as follows:

```
68 78 6A 4D FD 62 25 BD
FF 2E 88 06 6C 77 93 0E
9A
```

promsoft.com

3. After the completion of the machine language program, write it in PC-1500 by using POKE instruction. And execute the program together with BASIC by CALL instruction.

Execute the following program. You can run the BASIC program with [DEF] [A] and the machine language program with [DEF] [B]. Display reverse in machine language program is faster than that of BASIC. You would know how functional the machine language program is.

EX. Write the following program after executing NEW &4100 [ENTER].

```
10  "A" WAIT 0
20  PRINT" sharp pocket computer"
30  FOR A=0 TO 155
40  GCURSOR A
50  GPRINT 255-POINT A
60  NEXT A
70  GOTO 30
80  END
100 "B" WAIT 0
110 PRINT "sharp pocket computer"
120 POKE &40C5,&68,&78,&6A,&4D,
    &FD,&62,&25,&BD,&FF,&2E,&88,&06
130 POKE &40D1,&6C,&77,&93,&0E,&9A
140 CALL &40C5
150 WAIT 20:PRINT:GOTO 140
160 END
```

This manual is divided into three major sections; description of LSI (pp. 5~84: LH5801 Microprocessor & LH5810 LH5811 I/O port controller), PC-1500 hardware description (pp. 85~109), and PC-1500 software description (pp. 111~141).
If you want to know about PC-1500 system first, read from p. 86.

promsoft.com

promsoft.com

2

LH5801 Microprocessor

promsoft.com

2-1. Outline of LH5801

The LH5801 Microprocessor is a CMOS static 8-bit microprocessor that features low power dissipation performance inherent to CMOS LSI and large capacity data processing. Not only that, it enables to to configure a variety of systems with a few additional chips because such as the LCD backplate signal generator, input port, external latch clock, and timer are built in the LH5801.

Features of the LH5801

① 8-bit parallel data processing
② Direct accessing of 128K bytes
③ Use of a 6-byte general purpose register, in addition to the accumulator, allows to comprise three pairs of 2-byte date pointers.
④ 9-bit timer capability
⑤ Three kinds of interrupts
- Non-maskable interrupt
- Maskable interrupt
- Timer interrupt

⑥ 82 instruction set
⑦ WAIT function (memory access control possible)
⑧ Clock Pϕ for input port (8-bit) and external latch
⑨ Memory backup function (BFI, BFO)
⑩ LCD backplate signal control
⑪ External crystal connection for clock generation
⑫ Reducing program steps by means of 28-kind single step vector subroutine jump

promsoft.com

2-2. Internal Structure

2-2-1. Block diagram

Fig. 1-1 Block diagram of LH5801

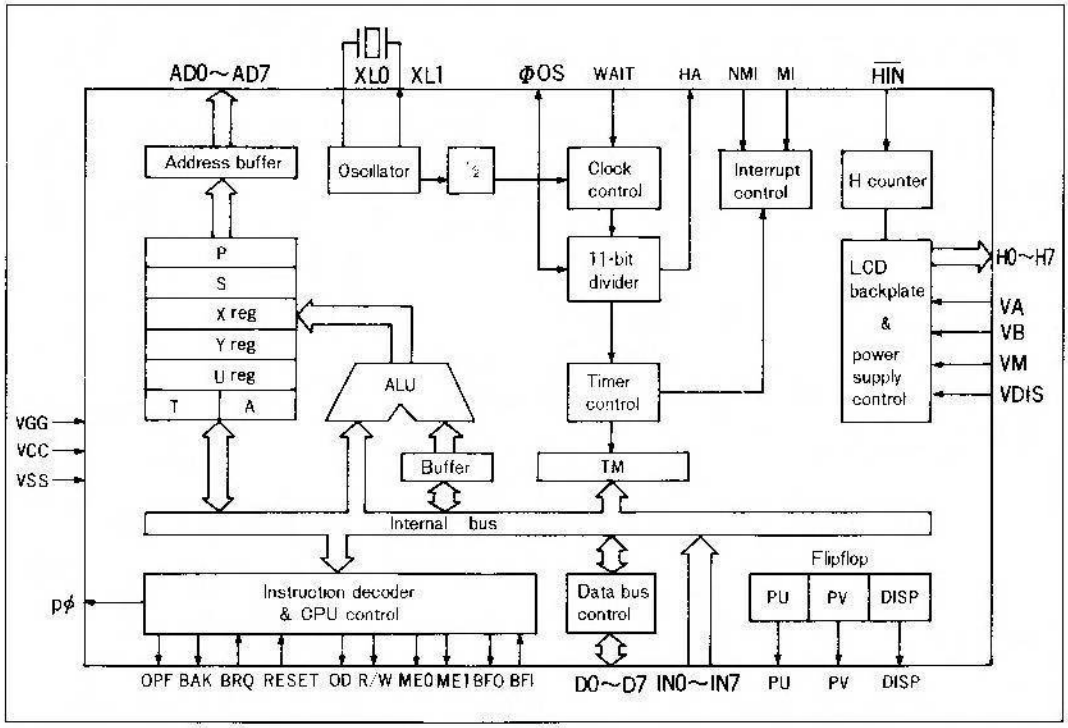
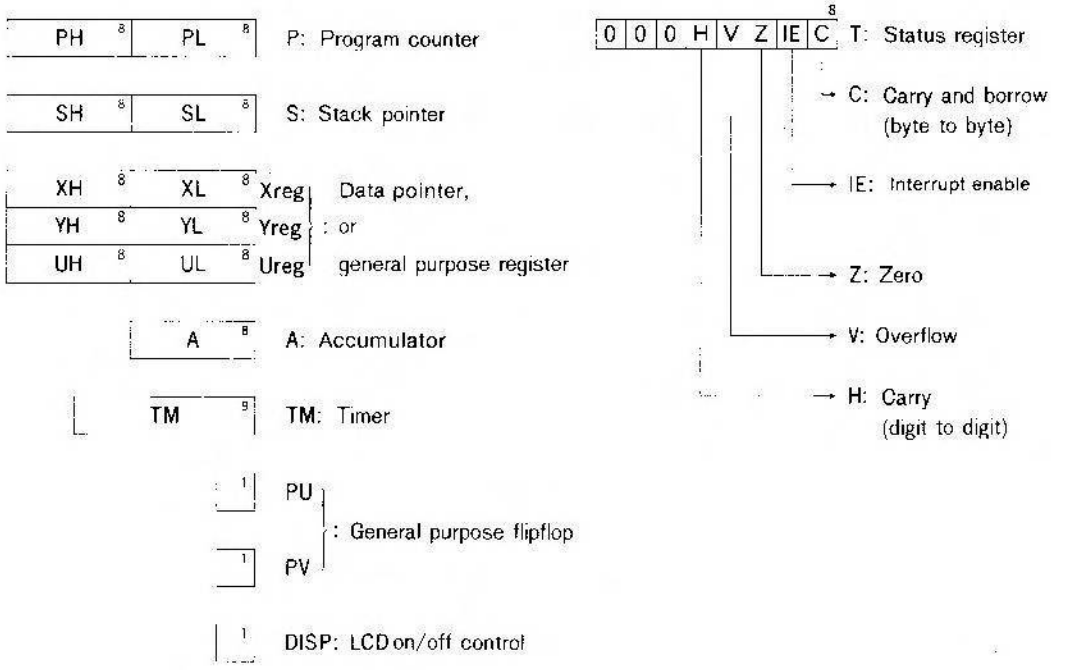


Fig. 1-2 Internal register & flipflops



promsoft.com

2-2-2. Internal registers

Symbol		Name	Bit size	Description
P		Program counter	16	Indicates the address next to the address the CPU is now in execution. It will be incremented by "1" when the next instruction is fetched.
S		Stack pointer	16	Indicates the stack address.
A		Accumulator	8	Used for retention of operational result or for data transfer with the external memory.
Xreg	XL	Data register	8	XL, XH, YL,YH,UL, UH comprise independent 8-bit registers. Also, used as 16-bit data pointers, Xreg, Yreg, and Ureg, when used in a pair.
	XH		8	
Yreg	YL		8	
	YH		8	
Ureg	UL		8	
	UH		8	
TM		Timer counter	9	When "0" is set to the TM, it stops the counter action. When anything other than "0" is set, it puts the counter into action. When the TM turns full of "1" with the interrupt enable signal IE on, CPU executes an interrupt processing.
PU			1	General purpose flip-flop.
PV			1	
DISP			1	LCD on/off control.
T		Status register	8	Low order 5 bits represent one of five status of operational result.

2-2-3. Status flags

Status flags, C, V, H, Z, IE are contained in the 8-bit status register. In general, flags other than IE change their state after the execution of arithmetical instruction.

Status register T = | 0 | 0 | 0 | H | V | Z | IE | C |

① **Carry flag C**

Carry flag C is set when there is a carry from the MSB and reset when there is no carry. For SUBTRACT, the flag is set when there is no borrow or reset when there is.

② **Half carry flag H**

Half carry flag H is set when there is a carry from the bit position "3" (digit-to-digit carry) and reset when there is no carry.

③ **Zero flag Z**

Zero flag Z is set when the operational result is zero and reset when not.

④ **Overflow flag V**

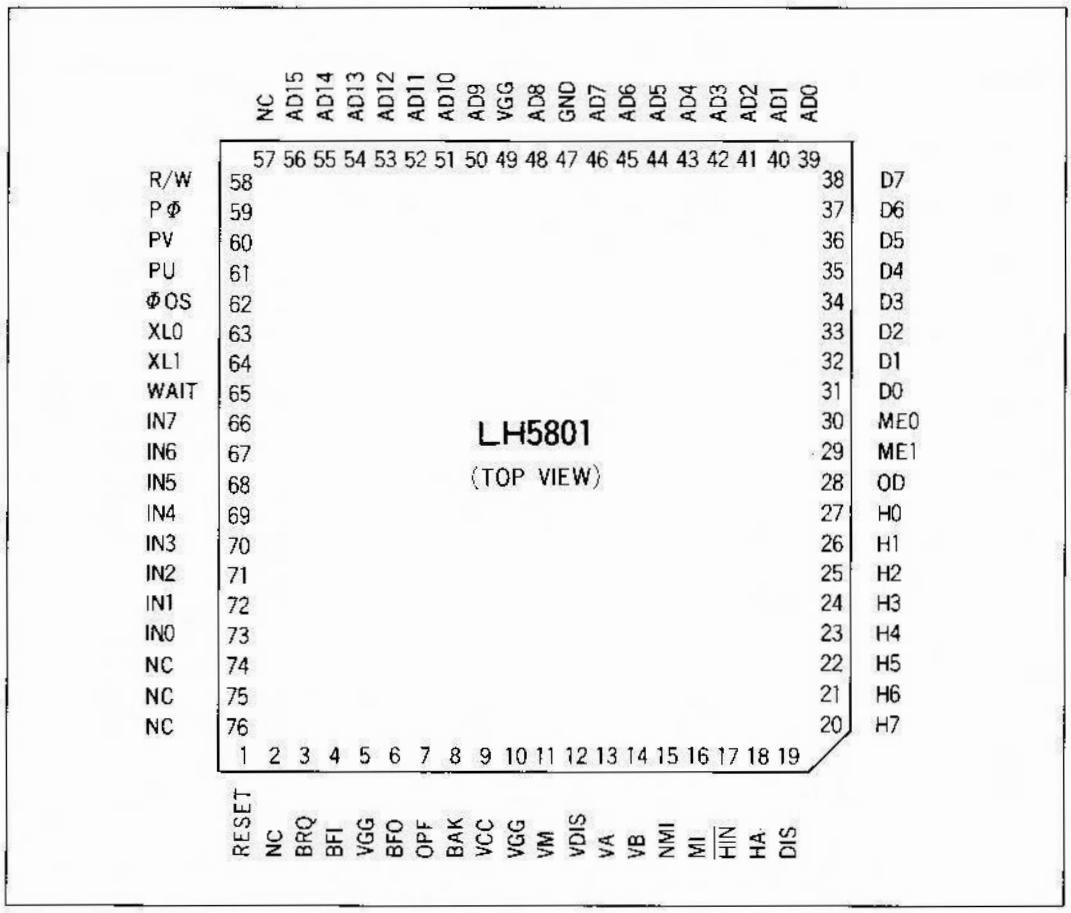
Overflow flag V is set or reset depending on the operational result of "C6⊕C7"; where, the carry from the bit position 6 of a single byte data is assumed to be C6 and the carry from the bit position 7 to be C7.

Single byte data

B7 B6 B5 B4 B3 B2 B1 B0

promsoft.com

2-2-4. CPU pin description



① **XL0, XL1**

These are external crystal connection pins. XL0 is the input pin and XL1 is the output pin. Clock frequency is divided by two inside the CPU. When the 2.6MHz crystal is connected, the CPU operates under 1.3MHz of internal machine cycle.

② **AD0~AD15**

Address bus. Outputs from these pins are 3-state (three output states of high, low and high impedance) and go high impedance with BRQ (Bus ReQuest). Basically, 64K bytes of memory area is supported for direct accessing, but it is made possible to access even 128K area of memory area when ME0 and ME1 are used.

③ **D0~D7**

Bidirectional data bus which is used to write or data to/from the external memory.

④ **ME0, ME1**

Memory enable signals. As memory area of 64K bytes is accessed by ME0 and another 64K bytes by ME1, it permits direct access of memory area of 128K bytes in total. Since ME0 is used for instruction fetch and stack operation, accessing by means of the program counter P and stack pointer S is limited to a maximum of 64K bytes.
As for data accessing, both memory areas covered by ME0 and ME1 can be controlled by a CPU instruction.

promsoft.com

⑤ **ΦOS**

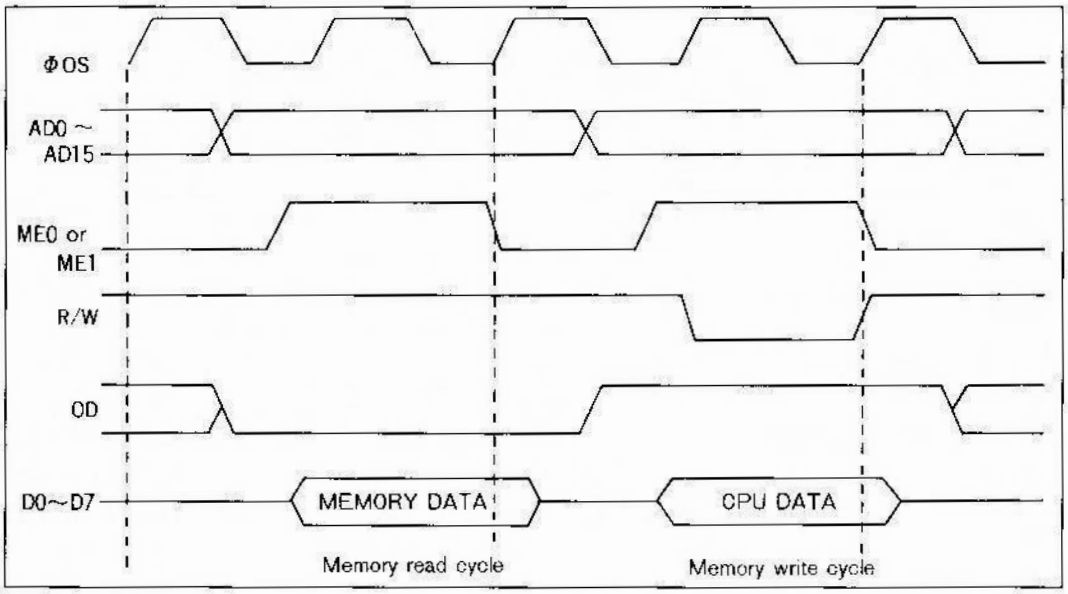
Through this line can be supplied the clock to an external system since the clock in the same phase as the CPU internal basic clock is on this line. Connection of the 2.6MHz crystal oscillator to XL0 and XL1 will supply the clock of 1.3MHz.

⑥ **R/W**

Memory write signal. A low on this line causes the CPU to send data on the data bus.

⑦ **OD**

Output disable signal. A high on this line causes the CPU to prohibit data output to the external device. It is used in writing data to the memory.



⑧ **RESET**

CPU reset input. High state of this signal resets the CPU and the contents of the address FFFEH is set to the register PH and the contents of the address FFFFH to the register PL. When it turns from high to low level, it starts program execution from the address of the program counter.

⑨ **NMI**

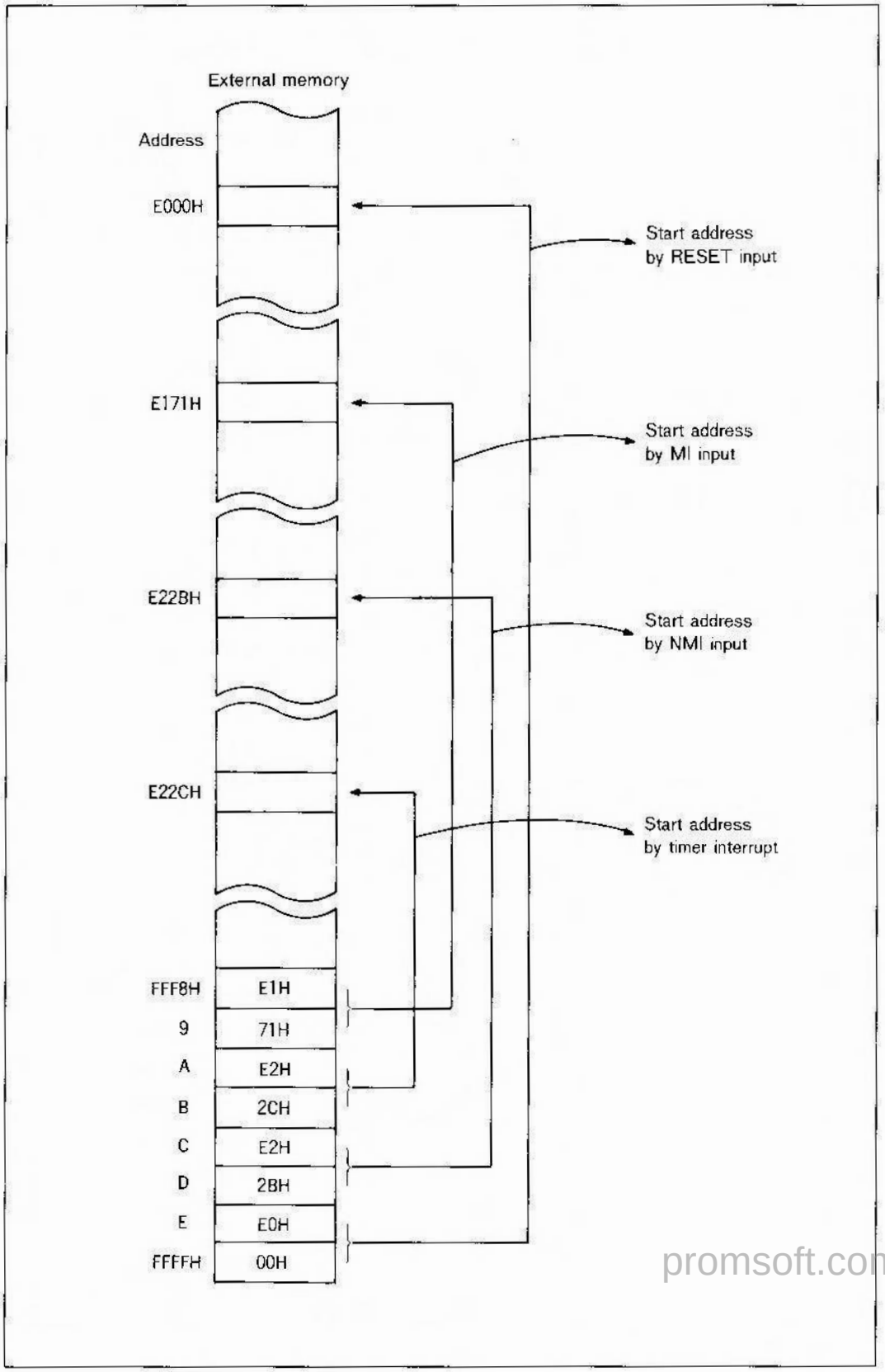
Non-maskable interrupt input. High state of this signal causes interrupt to the CPU, to which the CPU unconditionally responds and starts the interrupt processing routine of which high order byte address is represented by the address FFFCH and low order byte address by FFFDH.

promsoft.com

⑩ **MI**

Maskable interrupt input. When the interrupt enable flag IE is active, a high on the pin MI requests interrupt, to which the CPU starts to execute the interrupt processing routine whose high order byte address is represented by the address FFF8H and low order byte address by FFF9H.

How instruction execution address is determined against the reset and interrupt input



promsoft.com

⑪ **BRQ**

Bus request signal.

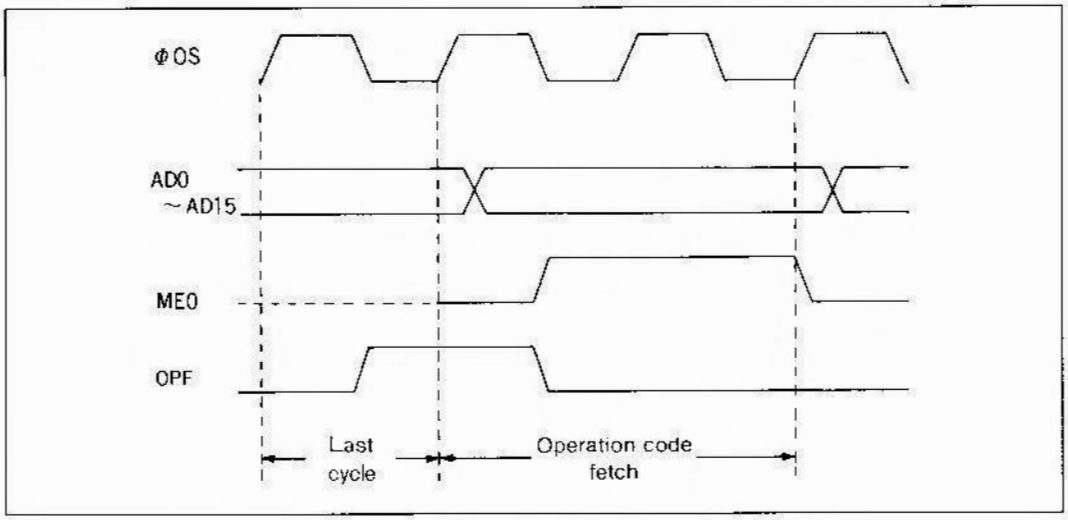
⑫ **BAK**

Bus request ackowledge signal. When BRQ goes high, the CPU issues a high level of signal on BAK in response to it. The CPU keeps address bus (AD0~AD15), data bus (D0~D7), ME0, ME1, R/W, and OD high impedance when BAK is in high level.

⑬ **OPF**

Operation code fetch signal which is sent out when CPU fetches operation code (instruction code).

OPF is issued only when operation code is fetched and will not be issued in fetching address data, immediate data, and second byte of the two-step instruction.



⑭ **IN0~IN7**

Input port through which the CPU receives 8-bit data into the accumulator. As internal pullup resistor is used, the CPU assumes it to be high level when not connected.

⑮ **PΦ**

External latch clock. With high level of this clock, the contents of the accumulator is sent on the data bus. Addition of IC will comprise an output port.

REFERENCE: ATP instruction.

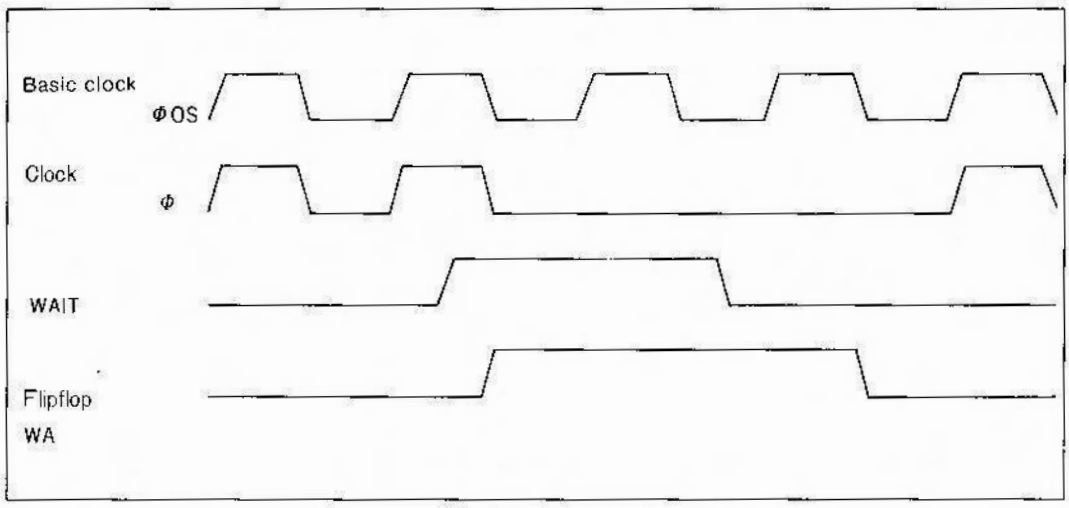
⑯ **PU, PV**

These are CPU internal flipflop outputs. PU and PV are furnished with set and reset instructions.

promsoft.com

⑰ **WAIT**

CPU wait signal. A high on this line stops the clock Φ so that the CPU halts its operation. As soon as WAIT turns low, the CPU resumes the operation.



NOTE: WA is the flipflop dedicated to WAIT, by which the WAIT input is received at the falling edge of the clock ΦOS. Because the CPU operating clock Φ stops when WA is high, it makes the CPU stopped, consequently.

⑱ **H0~H7**

LCD backplate signal output. As the liquid crystal display (LCD) is driven by backplate signals and segment signals, the CPU controls the backplate signals.

⑲ **VA, VB, VM, VDIS**

LCD backplate power supply inputs.

⑳ **$\overline{\text{HIN}}$**

Input signal to the counter from which the LCD backplate signals H0~H7 are generated. Normally, connected to pin HA of the CPU.

㉑ **HA**

CPU internal divider output pin. It is used for the basic clock that drives the LCD that connected with $\overline{\text{HIN}}$ and the segment signal generating LSI.

㉒ **DISP**

LCD on/off control signal output which can be set or reset by means of instruction.

㉓ **BFO, BFI**

BF flipflop output (BFO) and input (BFI). The BF flipflop is reset by the OFF instruction from the CPU, and will be set when the input BFI is turned to high level. BFO is in low level when the BF flipflop is set and in high level when reset.

As VGG is power supply to the BF flipflop, the contents of the flipflop are retained as long as VGG is in supply.

Normally, it is used for memory backup system.

promsoft.com

2-3. Functions

2-3-1. Timer

The timer is a 9-bit polynominal counter. The counter value can be set by the AM0 or AM1 instruction of the CPU. Shown in "POLINOMINAL COUNTER" is the list of hexadecimal data of counter with decimal count number.

The timer operates continuously at all times. When the counter value reaches 1FFH, it issues interrupt request to the CPU. If IE (Interrupt Enable) flag is active at that point, the CPU jumps to the timer interrupt processing routine whose high order byte address is represented by the contents of the address FFFAH and low order byte address by the contents of the address FFFBH. The timer has to be set to 000H when it is not used. Since it counts in synchronization with the clock ΦF, each one cycle of ΦF increments the counter one step.



When the 4MHz crystal oscillator is in connection, the timer is incremented at each 32μsec, because ΦF is 31.25kHz.

promsoft.com

"POLINOMINAL COUNTER"

COUNT NUMBER	DATA OF COUNTER	COUNT NUMBER	DATA OF COUNTER	COUNT NUMBER	DATA OF COUNTER	COUNT NUMBER	DATA OF COUNTER	COUNT NUMBER	DATA OF COUNTER
511	1FF	459	0B7	407	1AB	355	0A7	303	1E5
510	0FF	458	05B	406	1D5	354	153	302	1F2
509	07F	457	02D	405	0EA	353	0A9	301	1F9
508	03F	456	116	404	075	352	154	300	0FC
507	01F	455	18B	403	03A	351	1AA	299	17E
506	00F	454	1C5	402	11D	350	0D5	298	1BF
505	107	453	1E2	401	08E	349	06A	297	0DF
504	183	452	0F1	400	047	348	035	296	06F
503	1C1	451	078	399	123	347	01A	295	137
502	1E0	450	13C	398	191	346	10D	294	09B
501	0F0	449	19E	397	0C8	345	186	293	04D
500	178	448	1CF	396	064	344	0C3	292	126
499	1BC	447	1E7	395	032	343	161	291	093
498	1DE	446	1F3	394	119	342	1B0	290	049
497	1EF	445	0F9	393	08C	341	1D8	289	124
496	1F7	444	07C	392	046	340	1EC	288	092
495	0FB	443	13E	391	023	339	0F6	287	149
494	07D	442	19F	390	111	338	17B	286	1A4
493	03E	441	0CF	389	088	337	0BD	285	0D2
492	11F	440	167	388	044	336	05E	284	169
491	08F	439	1B3	387	022	335	12F	283	1B4
490	147	438	0D9	386	011	334	197	282	1DA
489	1A3	437	06C	385	008	333	0CB	281	1ED
488	1D1	436	036	384	004	332	165	280	1F6
487	0E8	435	11B	383	002	331	1B2	279	1FB
486	074	434	08D	382	001	330	1D9	278	0FD
485	13A	433	146	381	100	329	0EC	277	07E
484	19D	432	0A3	380	080	328	076	276	13F
483	0CE	431	151	379	040	327	13B	275	09F
482	067	430	0A8	378	020	326	09D	274	04F
481	133	429	054	377	010	325	04E	273	127
480	099	428	12A	376	108	324	027	272	193
479	04C	427	095	375	084	323	113	271	0C9
478	026	426	04A	374	042	322	089	270	164
477	013	425	025	373	021	321	144	269	0B2
476	009	424	112	372	110	320	0A2	268	159
475	104	423	189	371	188	319	051	267	0AC
474	082	422	1C4	370	0C4	318	028	266	056
473	041	421	0E2	369	062	317	014	265	12B
472	120	420	071	368	031	316	10A	264	195
471	090	419	038	367	018	315	085	263	0CA
470	148	418	11C	366	10C	314	142	262	065
469	0A4	417	18E	365	086	313	0A1	261	132
468	052	416	0C7	364	043	312	150	260	199
467	129	415	163	363	121	311	1A8	259	0CC
466	194	414	1B1	362	190	310	0D4	258	066
465	1CA	413	0D8	361	1C8	309	16A	257	033
464	0E5	412	16C	360	0E4	308	0B5	256	019
463	172	411	0B6	359	072	307	05A	255	00C
462	1B9	410	153	358	139	306	12D	254	006
461	0DC	409	0AD	357	09C	305	196	253	003
460	16E	408	156	356	14E	304	1CB	252	101

promsoft.com

COUNT NUMBER	DATA OF COUNTER	COUNT NUMBER	DATA OF COUNTER	COUNT NUMBER	DATA OF COUNTER	COUNT NUMBER	DATA OF COUNTER	COUNT NUMBER	DATA OF COUNTER
251	180	199	09A	147	141	95	00E	43	01C
250	0C0	198	14D	146	1A0	94	007	42	10E
249	060	197	1A6	145	0D0	93	103	41	087
248	030	196	0D3	144	168	92	181	40	143
247	118	195	069	143	0B4	91	1C0	39	1A1
246	18C	194	134	142	15A	90	0E0	38	1D0
245	0C6	193	19A	141	1AD	89	070	37	1E8
244	063	192	1CD	140	1D6	88	138	36	0F4
243	131	191	1E6	139	1ER	87	19C	35	17A
242	098	190	0F3	138	1F5	86	1CE	34	1BD
241	14C	189	079	137	0FA	85	0E7	33	0DE
240	0A6	188	03C	136	17D	84	173	32	16F
239	053	187	11E	135	0BE	83	0B9	31	1B7
238	029	186	18F	134	15F	82	05C	30	0DB
237	114	185	1C7	133	0AF	81	12E	29	06D
236	18A	184	1E3	132	157	80	097	28	136
235	0C5	183	1F1	131	0AB	79	04B	27	19B
234	162	182	0F8	130	155	78	125	26	0CD
233	0B1	181	17C	129	0AA	77	192	25	166
232	058	180	1BE	128	055	76	1C9	24	0B3
231	12C	179	1DF	127	02A	75	1E4	23	059
230	096	178	0EF	126	015	74	0F2	22	02C
229	14B	177	177	125	00A	73	179	21	016
228	1A5	176	0BB	124	005	72	0BC	20	10B
227	1D2	175	05D	123	102	71	15E	19	185
226	1E9	174	02E	122	081	70	1AF	18	1C2
225	1F4	173	017	121	140	69	1D7	17	0E1
224	1FA	172	00B	120	0A0	68	0EB	16	170
223	1FD	171	105	119	050	67	175	15	1B8
222	0FE	170	182	118	128	66	0BA	14	1DC
221	17F	169	0C1	117	094	65	15D	13	1EE
220	0BF	168	160	116	14A	64	0AE	12	0F7
219	05F	167	0B0	115	0A5	63	057	11	07B
218	02F	166	158	114	152	62	02B	10	03D
217	117	165	1AC	113	1A9	61	115	9	01E
216	08B	164	0D6	112	1D4	60	08A	8	10F
215	145	163	16B	111	1EA	59	045	7	187
214	1A2	162	185	110	0F5	58	122	6	1C3
213	0D1	161	0DA	109	07A	57	091	5	1E1
212	068	160	16D	108	13D	56	048	4	1F0
211	034	159	1B6	107	09E	55	024	3	1F8
210	11A	158	1DB	106	14F	54	012	2	1FC
209	18D	157	0ED	105	1A7	53	109	1	1FE
208	1C6	156	176	104	1D3	52	184	0	1FF
207	0E3	155	1BB	103	0E9	51	0C2		
206	171	154	0DD	102	174	50	061		
205	0B8	153	06E	101	1BA	49	130		
204	15C	152	037	100	1DD	48	198		
203	1AE	151	01B	99	0EE	47	1CC		
202	0D7	150	00D	98	077	46	0E6		
201	06B	149	106	97	03B	45	073		
200	135	148	083	96	01D	44	039		

promsoft.com

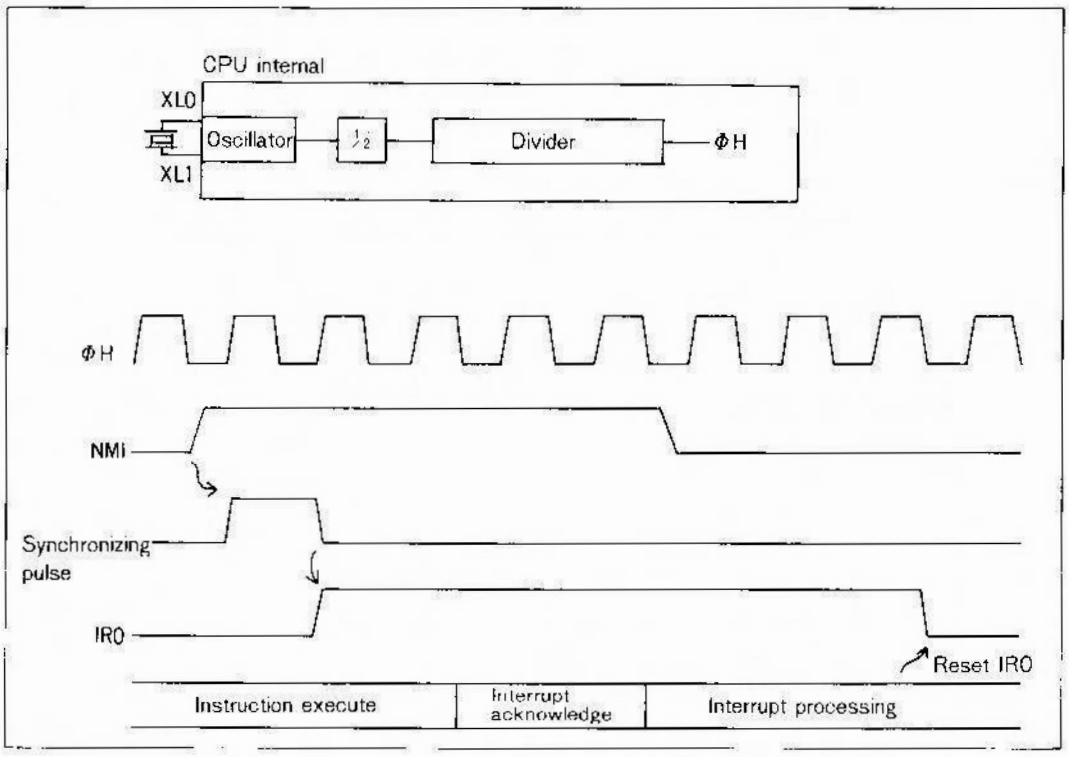
2-3-2. Interrupts

There are following three kinds of interrupt for the LH5801 CPU.

① Non-maskable interrupt

When NMI is turned from low to high level, it sets the flipflop IR0 active, upon which time interrupt is requested to the CPU, so that the CPU starts executing the interrupt processing after completion of current instruction execution.
Since the non-maskable interrupt is given with the highest priority order, the interrupt will be acknowledged at once, regardless of its internal state.
IR0 will be reset in a course of the interrupt process.

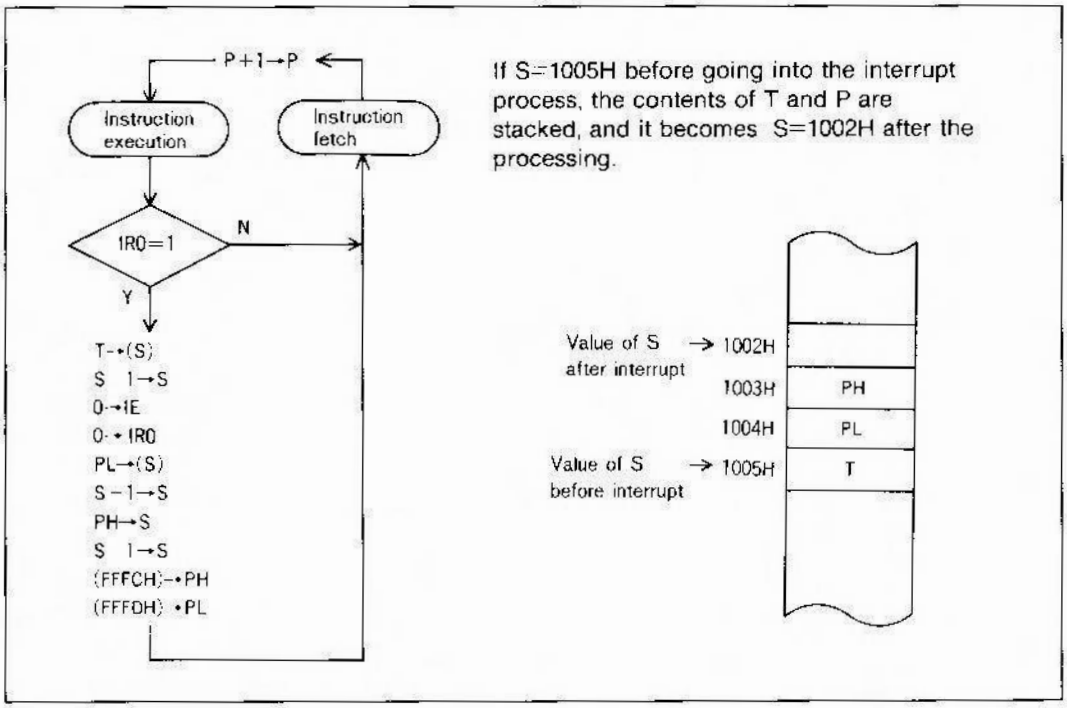
As the NIM input is sampled by the clock ΦH, it will become 250kHz when the 4MHz crystal oscillator is connected.



When NMI is turned high, the CPU creates synchronizing pulse at the rising edge of the clock ΦH, which sets IR0 active. If IR0 is active when the CPU acknowledged the interrupt after completion of execution, it goes into the interrupt routine and IR0 is reset in a course of the interrupt processing.

promsoft.com

Non-maskable interrupt processing sequence



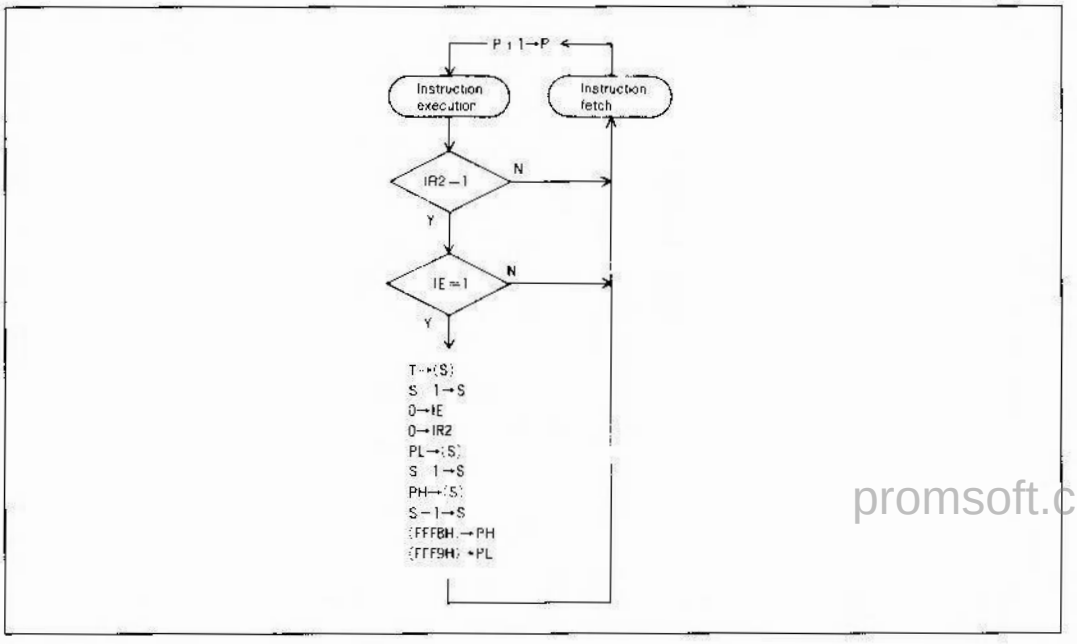
② Maskable interrupt

When MI is turned from low to high level, it sets the flipflop IR2 active. If the interrupt enable IE is active at this point, interrupt is requested to the CPU, so that the CPU starts executing the interrupt processing after completion of current instruction execution. IR2 will be reset in a course of the interrupt process.

When interrupt request is issued while IE is inactive, the interrupt will be ignored even though IR2 is set.

MI input is sampled in the same manner as in the case of NMI

Maskable interrupt processing sequence



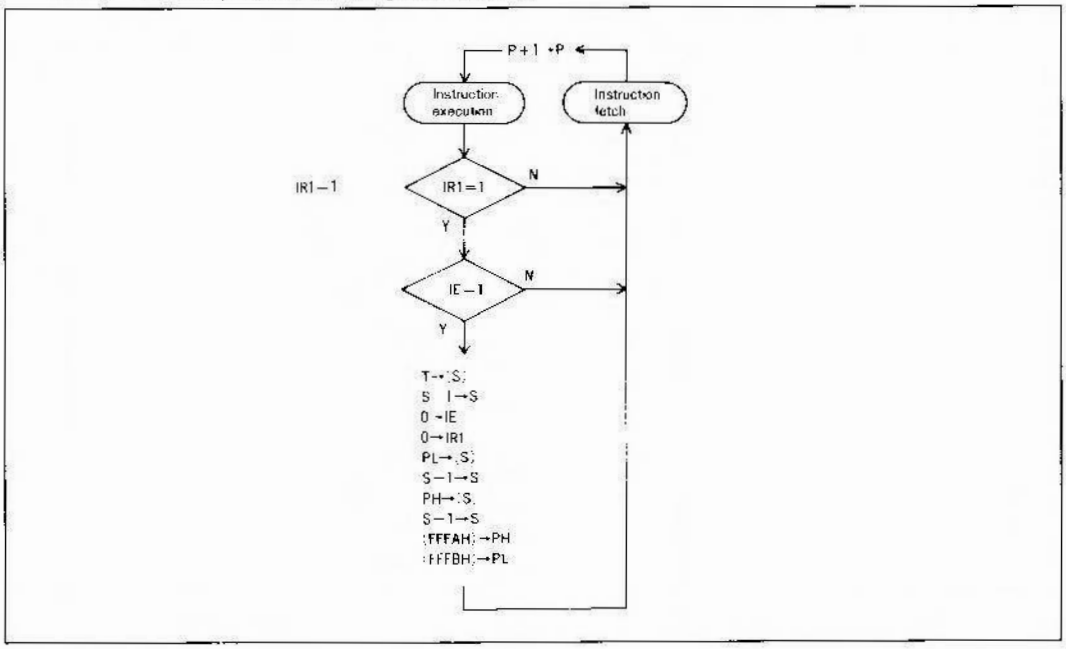
promsoft.com

③ Timer interrupt

When interrupt is requested from the timer, it sets the flipflop IR1 active. If the interrupt enable IE is active at this point, the CPU starts executing the interrupt processing after completion of current instruction execution, and IR1 will be reset in a course of the interrupt process.

When interrupt is requested while IE is inactive, the interrupt will be ignored even though IR1 is set.

Timer interrupt processing sequence



Return to main routine

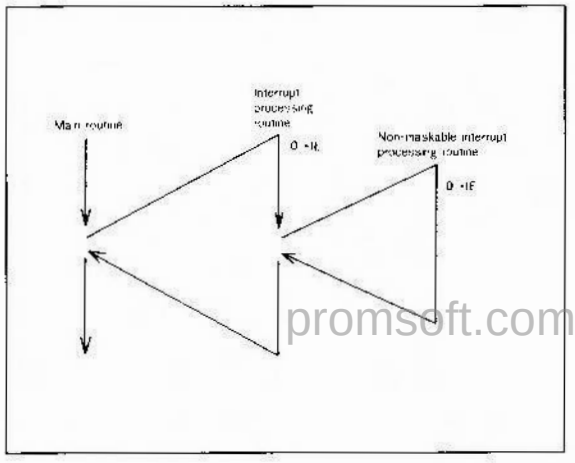
The RTI instruction is used for returning from the interrupt processing routine to the main routine.

Because the contents of the T register and the program counter are stored in the stack at the beginning of the interrupt processing routine, the contents of the T register in the stack returns by the RTI instruction. Since the interrupt enable flag IE is contained in the T register, the contents of IE immediately before the interrupt returns by the RTI instruction.

To disable maskable or timer interrupt by the main routine after returning from the interrupt processing routine, the bit in the stack corresponding to the flag IE must be reset.

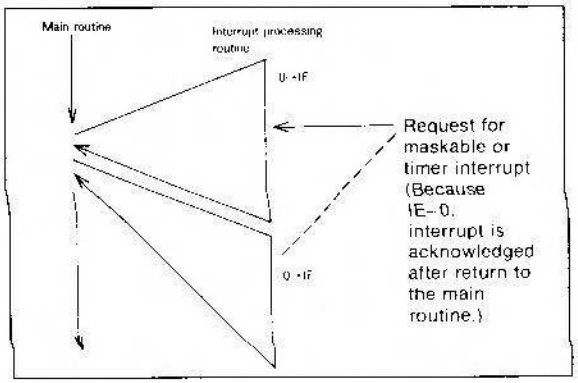
Priority order of interrupts

(1) Non-maskable interrupt responds to the interrupt request whatever the CPU internal state may be. Also, it responds in the first priority even during execution of interrupt routine by other interrupt.

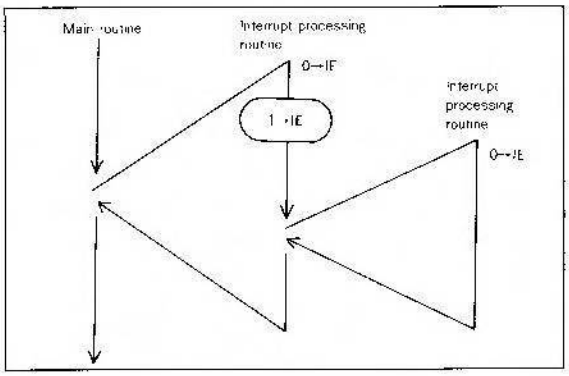


promsoft.com

(2) When either maskable or timer interrupt request is met during execution of the interrupt processing routine, it will be acknowledged upon returning to the main routine because IE was reset during execution of the interrupt processing routine.

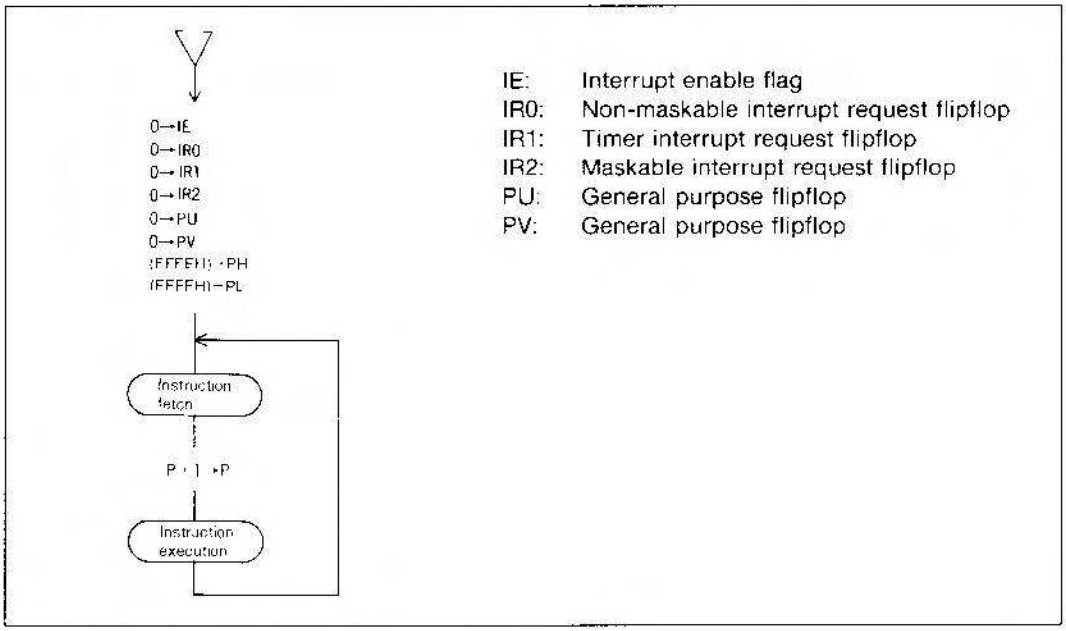


(3) In order to acknowledge maskable or timer interrupt during execution of the interrupt processing routine, the flag IE must be set active at the top of that processing routine.



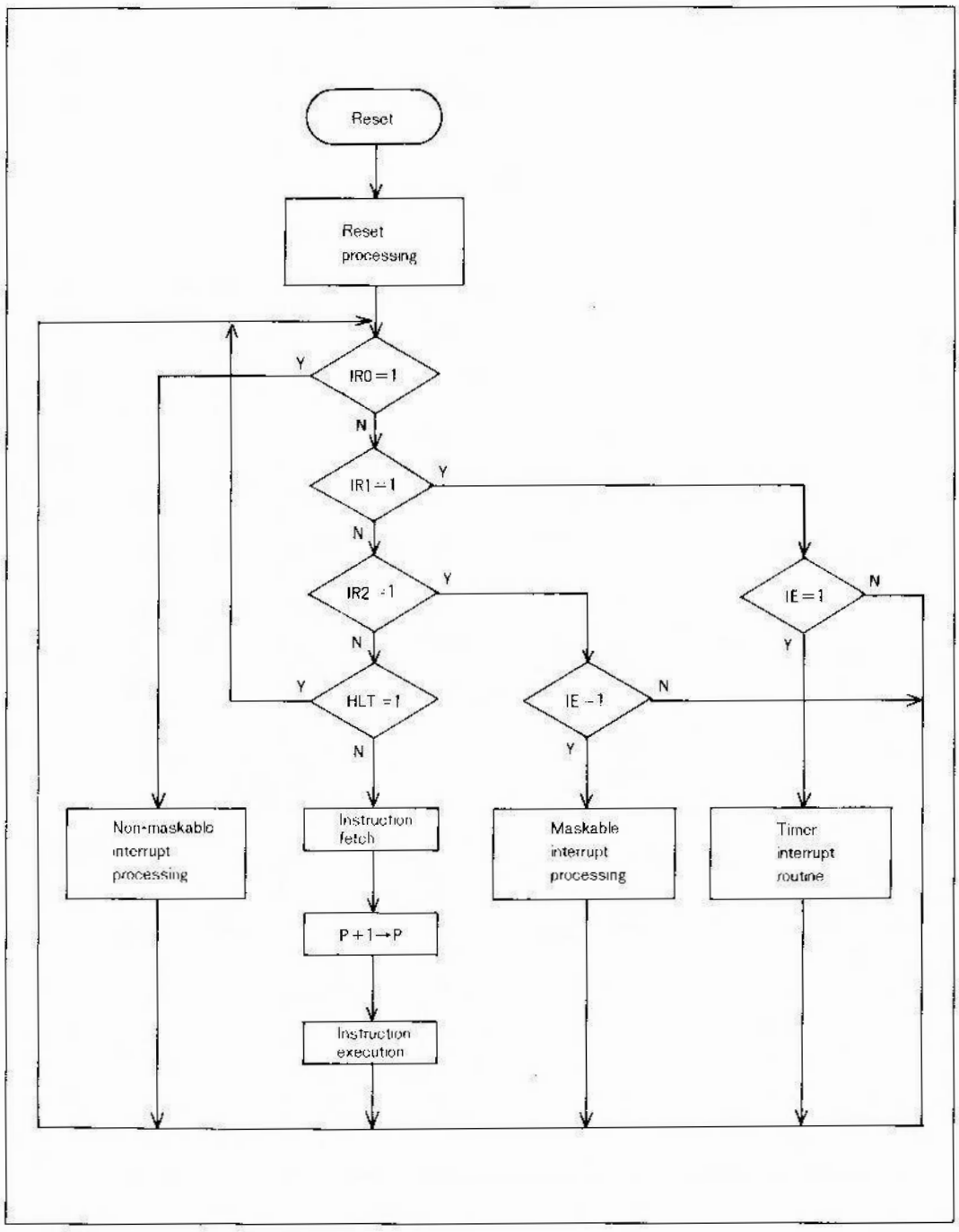
2-3-3. Reset

When RESET is turned from low to high and low again, it resets CPU in the sequence described below, and the contents of the external memory address FFFEH is sent to the register PH and the contents of the address FFFFH to the register PL, then instruction is executed.



promsoft.com

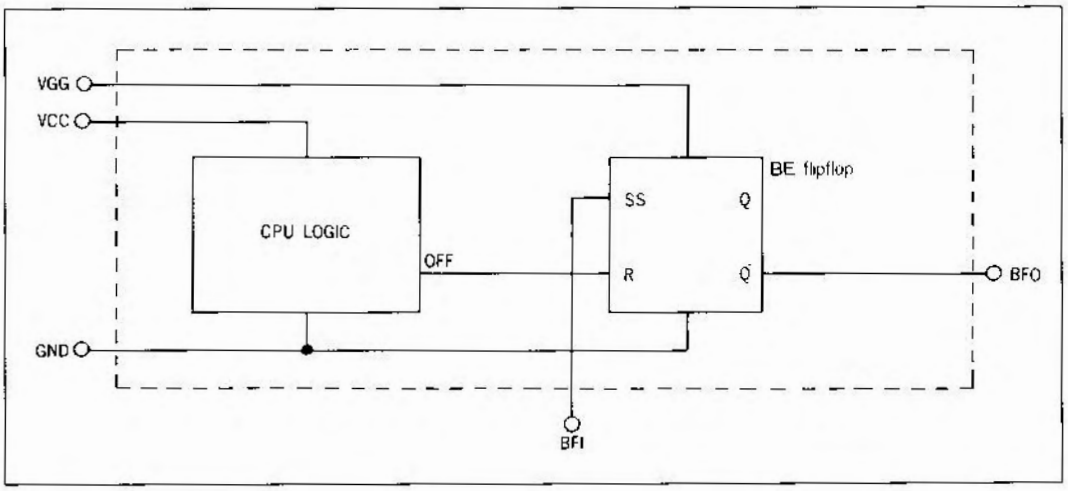
2-3-4. CPU system sequence



promsoft.com

2-3-5. BF flipflop

The CPU has two supplies of power; VCC and VGG. As the BF flipflop is driven by VGG, the state of the BF flipflop is retained as long as VGG is in supply, even if VCC supply is out. The BF flipflop is set when the BFI input is turned from low to high level and reset when the OFF instruction is issued from the CPU.
Low state of signal is on the BF flipflop output BFO when the flipflop is active and high state of signal when inactive.



It is possible to comprise a memory backup system using this BF flipflop.

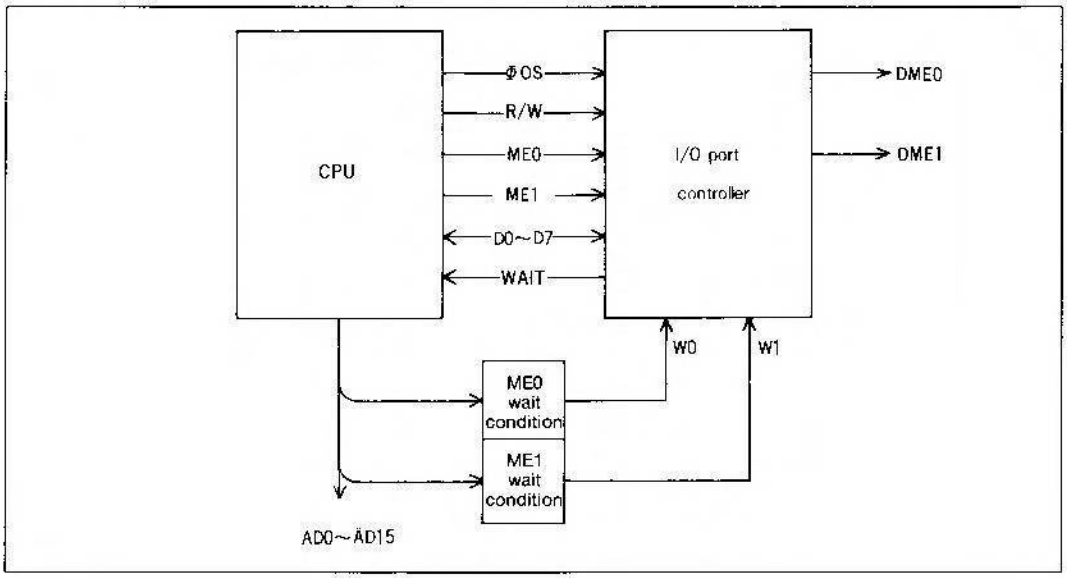
promsoft.com

2-3-6. WAIT function (interfacing with the slow access time memory)

Because the CPU access time is very fast, the CPU must be held with the wait signal in order to access a slow access time memory.

Basically, the wait signal is created by the externally provided counter, but the I/O port controller with the programmable wait time control feature (LH8511) is used for this purpose. For more details of this function, refer to the section discussing the I/O port controller.

Shown in the next page is the connection of the CPU with the I/O port controller.

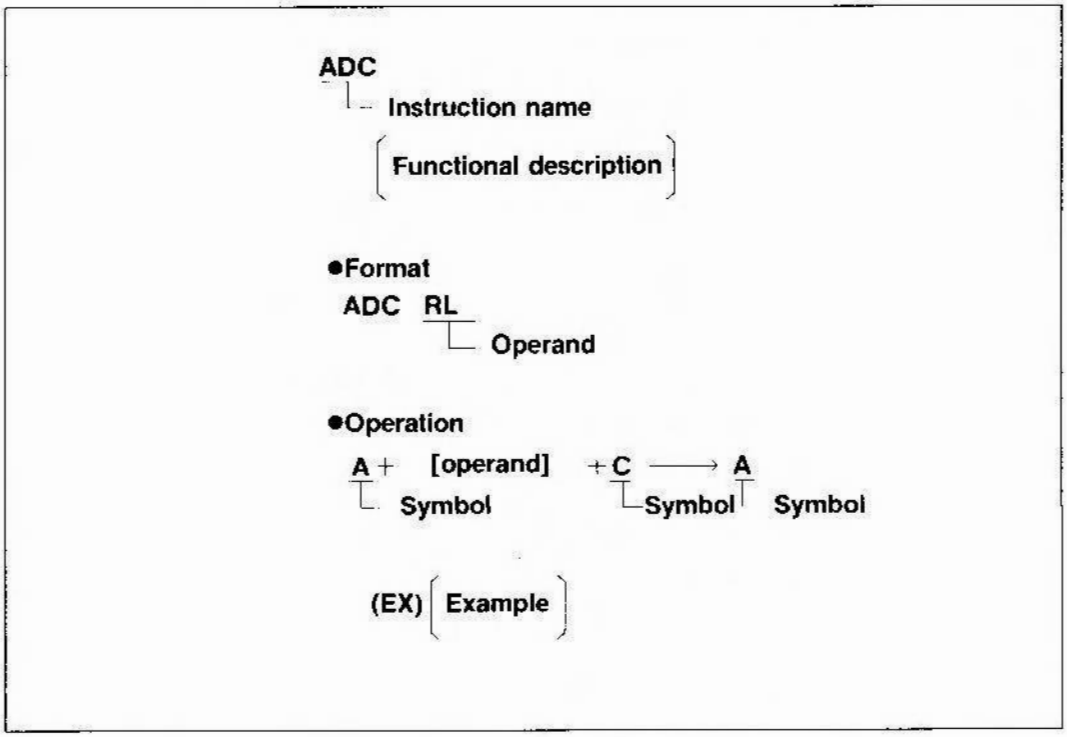


promsoft.com

2-4. LH5801 instructions

2-4-1. Outline

This section deals with function of each instruction.



Symbols used in discussing operand, function, and operation

A: Contents of the accumulator (8 bits)
XL: Contents of the low order 8 bits of Xreg
YL: Contents of the low order 8 bits of Yreg
UL: Contents of the low order 8 bits of Ureg
(RL represents either of XL, YL, or UL.)
XH: Contents of the high order 8 bits of Xreg
YH: Contents of the high order 8 bits of Yreg
UH: Contents of the high order 8 bits of Ureg
(RH represents either of XH, YH, or UH.)
Xreg: Contents of the X register (16 bits)
Yreg: Contents of the Y register (16 bits)
Ureg: Contents of the U register (16 bits)
(Rreg represents either of Xreg, Yreg, or Ureg.)
P: Contents of the program counter (16 bits)
PL: Contents of low order 8 bits of the program counter
PH: Contents of high order 8 bits of the program counter
S: Contents of the stack pointer (16 bits)
SL: Contents of low order 8 bits of the stack pointer
SH: Contents of high order 8 bits of the stack pointer
(Rreg): Contents of the memory represented by Rreg (accessed with ME0)
#(Rreg): Contents of the memory represented by Rreg (accessed with ME1)
(*ab*): Contents of the memory represented by 16 bits of *ab* (accessed by ME0); where *a* represents high order 8 bits of the address and *b* low order 8 bits

promsoft.com

#(*ab*):	Contents of the memory represented by 16 bits of *ab* (accessed by ME1); where *a* represents high order 8 bits of the address and *b* low order 8 bits
i:	8-bit immediate data
i, *j*:	16-bit immediate data of which high order 8 bits are represented by *i* and low order 8 bits by *j*
C:	Carry flag
IE:	Interrupt enable flag
Z:	Zero flag
V:	Overflow flag
H:	Half carry flag
$\rightleftarrows$	Data flow direction
$\wedge$	AND
$\vee$	OR
$\oplus$	Exclusive OR
+	ADD
−	SUBTRACT

2-4-2. Add, subtract, and logical instructions

① ADC (ADd with Carry)

Either the contents of the internal register or external memory are 8-bit added with the accumulator including carry, and its result is stored in the accumulator.
C, H, Z, and V may change.

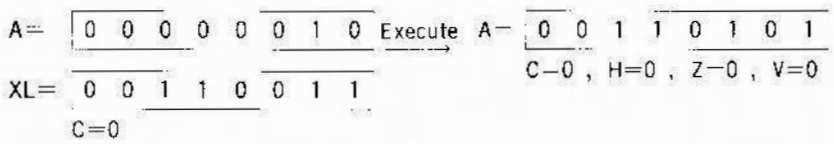
•Format

ADC RL
ADC RH
ADC (Rreg)
ADC #(Rreg)
ADC (*ab*)
ADC #(*ab*)

•Operation

A + [operand] + C → A

(EX) ADC XL
Value of the XL register is added to the accumulator, providing "C=0".



promsoft.com

② ADI (ADd Immediate)

To either the accumulator or the external memory is added the immediate data. In the case of ADI to the accumulator, carry is included in the operation.

C, H, Z, and V may change.

•Format

ADI A,*i*

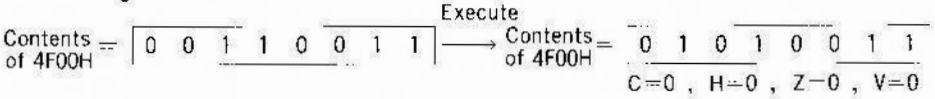
ADI (Rreg),*i*
ADI #(Rreg),*i*
ADI (*ab*),*i*
ADI #(*ab*),*i*

•Operation

A + *i* + C → A

[operand] + *i* → [operand]

(EX) ADI (Xreg), 20H
20H is added to the memory represented by X register.
Xreg=4F00H



③ DCA (DeCimal Add)

Decimal addition is carried out between the external memory and the accumulator including carry, and its result is stored in the accumulator.

C, H, Z, and V may change.

•Format

DCA (Rreg)
DCA #(Rreg)

•Operation

1) A + 66H → A
2) A + [operand] + C → A
(C, H, Z, and V may change.)
3) A + DA → A
Where, DA is used for compensation of decimal number which is dependent on the value of flags C and H in regard to Item 2).

C	H	DA
0	0	9AH
0	1	A0H
1	0	FAH
1	1	00H

(EX) DCA (Yreg)
Decimal addition is carried out between the memory represented by the Y register and the accumulator.

Y reg = 4700H

A	Contents of 4700H	C	After execution of DCA C	H	A	Decimal addition
35H	27H	0	0	1	62H	35+27+0=62
35H	27H	1	0	1	63H	35+27+1=63
35H	67H	0	1	1	02H	35+67+0=102
35H	67H	1	1	1	03H	35+67+1=103

promsoft.com

④ ADR (ADd Rreg)

Contents of the accumulator are added to the R register.
C, H, Z, and V may change.

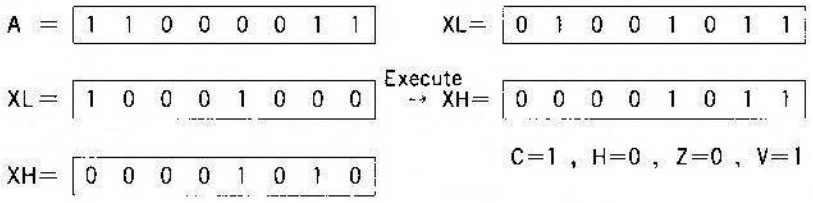
•Format

ADR Rreg

•Operation

1) RL + A → RL
(C, H, Z, and V may change.)
2) RH + C → RH

(EX) ADR Xreg
Contents of the accumulator are added to X register.



⑤ SBC (SuBtract with Carry)

The contents of the accumulator are subtracted by the internal register or external memory including $\overline{C}$, and its result is stored in the accumulator.
This operation may also be expressed in the following manner.
Complement of the internal register or external memory is obtained first, addition is carried out including carry, then its result is stored in the accumulator.
C, H, Z, and V may change.

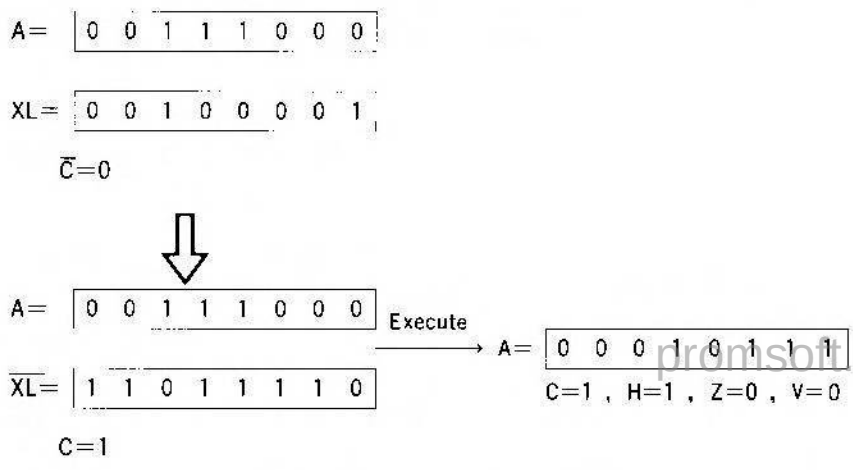
•Format

SBC RL
SBC RH
SBC (Rreg)
SBC #(Rreg)
SBC (*ab*)
SBC #(*ab*)

•Operation

A − [operand] − $\overline{C}$ → A
(C, H, Z, and V may change.)
or,
A + $\overline{[\text{operand}]}$ + C → A
(C, H, Z, and V may change.)

(EX) SBC XL
Contents of XL register are subtracted from the accumulator.



promsoft.com

⑥ SBI (SuBtract Immediate)

The contents of the accumulator are subtracted by the immediate data including $\overline{C}$. C, H, Z, and V may change.

•Format

SBI A,i

•Operation

$A - i - \overline{C} \rightarrow A$

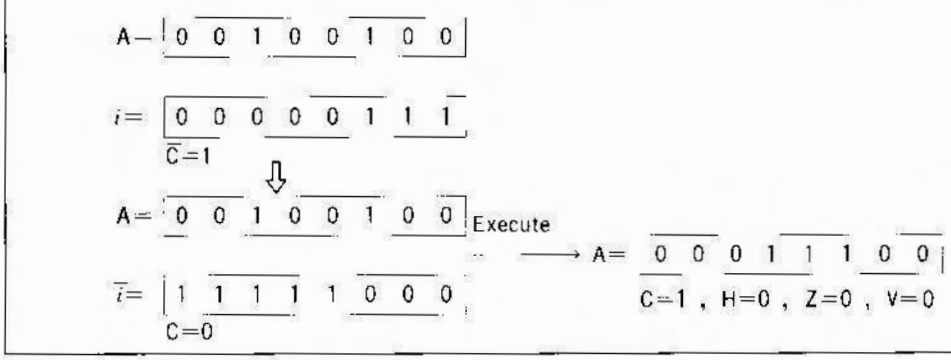
(C, H, Z, and V may change.)

or,

$A + \overline{i} + C \rightarrow A$

(C, H, Z, and V may change.)

(EX) SBI A, 07H



⑦ DCS (DeCimal Subtract)

The contents of the accumulator are decimal subtracted by the external memory including $\overline{C}$, and its result is stored in the accumulator.

C, H, Z, and V may change.

•Format

DCS (Rreg)

DCS #(Rreg)

•Operation

1) $A + \overline{[operand]} + C \rightarrow A$

(C, H, Z, and V may change.)

2) $A + DA \rightarrow A$

Where, DA is used for compensation of decimal number which is dependent on the value of flags C and H in regard to item 1).

C	H	DA
0	0	9AH
0	1	A0H
1	0	FAH
1	1	00H

(EX) DCS (Xreg)

The contents of the memory represented by the X register are decimal subtracted from the accumulator.

Xreg=4700H

A	Contents of 4700H	C	After execution of DCS C	H	A	Decimal subtraction
42H	31H	1	1	1	11H	42−31−0=11
42H	31H	0	1	1	10H	42−31−1−10
23H	54H	1	0	0	69H	23−54−0=69−100
23H	54H	0	0	0	68H	23−54 1−68−100

pronsoft.com

⑧ AND

The contents of the accumulator are ANDed with the value of the external memory, and its result is stored in the accumulator.
Only the flag Z changes.

•Format

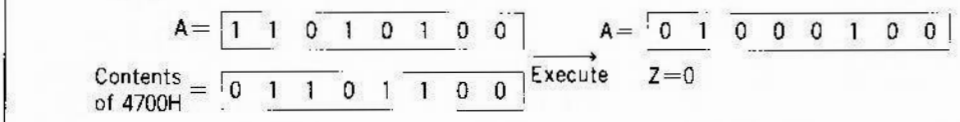
AND (Rreg)
AND #(Rreg)
AND (*ab*)
AND #(*ab*)

•Operation

A ∧ [operand] → A
(Z changes.)

(EX) AND (Ureg)

Ureg=4700H



⑨ ANI (ANd Immediate)

The contents of the accumulator or external memory are ANDed with the immediate data, and its result is stored in the accumulator or the external memory.
Only the flag Z changes.

•Format

ANI A,*i*
ANI (Rreg),*i*
ANI #(Rreg),*i*
ANI (*ab*),*i*
ANI #(*ab*),*i*

•Operation

[operand] ∧ *i* → [operand]
(Flag Z changes.)

⑩ ORA (OR Acc)

The contents of the accumulator are ORed with the value of the external memory, and its result is stored in the accumulator.
Only the flag Z changes.

•Format

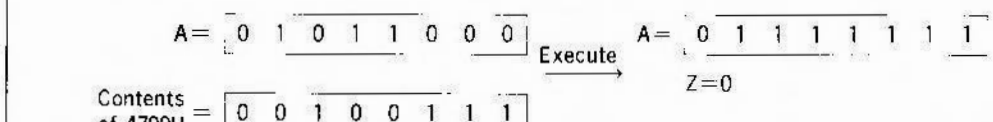
ORA (Rreg)
ORA #(Rreg)
ORA (*ab*)
ORA #(*ab*)

•Operation

A ∨ [operand] → A
(Flag Z changes.)

(EX) ORA (Ureg)

Ureg=4700H



promsoft.com

⑪ ORI (OR Immediate)

The contents of the accumulator or external memory are ORed with the immediate data, and its result is stored in the accumulator or the external memory.
Only flag Z changes.

•Format

ORI A,*i*
ORI (Rreg),*i*
ORI #(Rreg),*i*
ORI (*ab*),*i*
ORI #(*ab*),*i*

•Operation

[Operand] ∨ *i* → [operand]
(Z changes.)

⑫ EOR (Exclusive OR)

The contents of the accumulator are exclusively ORed with the value of the external memory, and its result is stored in the accumultor.
Only the flag Z changes.

•Format

EOR (Rreg)
EOR #(Rreg)
EOR (*ab*)
EOR #(*ab*)

•Operation

A ⊕ [operand] → A
(Z changes.)

(EX) EOR (Xreg)

Xreg=4700H

A	−	0	0	1	1	0	1	1	0	Execute → A = 0 1 0 1 1 0 1 1
Contents of 4700H	−	0	1	1	0	1	1	0	1	Z−0

⑬ EAI (Exclusive or Acc and Immediate)

The contents of the accumulator are exclusively ORed with the immediate data, and its result is stored in the accumulator.
Only the flag Z changes.

•Format

EAI *i*

•Operation

A ⊻ *i* → A
(Z changes.)

⑭ INC (INCrement)

The value of the accumulator or the register is INCremented by one. In the case of the 8-bit register (A, RL, RH), it makes flags C, V, H and Z changed. In the case of the 16-bit register Rreg, no flag change takes place.

•Format

INC A
INC RL
INC RH

INC Rreg

•Operation

[Operand] + 1 → [operand]
(C, V, H, and Z changed.)

[Operand] + 1 → [operand]

promsoft.com

(EX) INC XL

XL= 0 0 1 1 0 1 1 1 —Execute→ XL= 0 0 1 1 1 0 0 0

C=0 , H=0 , Z=0 , V=0

⑮ DEC (DECrement)

The value of the accumulator or register is DECremented by one. In the case of the 8-bit register (A, RL, RH), it makes flags C, V, H and Z changed. In the case of the 16-bit register Rreg, no flag change takes place.

•Format

DEC A
DEC RL
DEC RH

DEC Rreg

•Operation

[Operand] − 1 → [operand]
(C, V, H, and Z changed.)

[Operand] − 1 → [operand]

(EX) DEC Xreg

XL= 0 0 0 0 0 0 0 0 —Execute→ XL= 1 1 1 1 1 1 1 1

XH= 0 0 0 1 0 0 0 1 XH= 0 0 0 1 0 0 0 0

2-4-3. Compare and bit test

⑯ CPA (ComPare Acc)

The contents of the accumulator are compared with register or external memory, and its result is represented by flags C, V, H, and Z.

•Format

CPA RL
CPA RH
CPA (Rreg)
CPA #(Rreg)
CPA (*ab*)
CPA #(*ab*)

•Operation

A − [operand] → change in C, V, H, and Z
or
A + $\overline{[\text{operand}]}$ + 1 → change in C, V, H, and Z

Comparison	C	Z	V	H
A>[operand]	1	0	*	*
A=[operand]	1	1	*	*
A<[operand]	0	0	*	*

* : Flags V and H may change according to the condition mentioned in "Status flag", but it has no meaning with the CPA instruction.

(EX) CPA XL

A= 0 1 0 1 0 1 0 0 —Execute→ C=1 , Z=0

XL= 0 1 0 1 0 0 0 0

promsoft.com

⑰ CPI (ComPare Immediate)

The contents of the accumulator or register are compared with the immediate data, and its result is represented by flags C, V, H, and Z.

•Format

CPI RL,*i*
CPI RH,*i*
CPI A,*i*

•Operation

[Operand] − *i* → change in C, V, H, Z

Comparison	C	Z	V	H
[Operand]>*i*	1	0	*	*
[Operand]=*i*	1	1	*	*
[Operand]<*i*	0	0	*	*

* : Refer to "CPA instruction".

⑱ BIT (test BIT)

The accumulator is ANDed with the external memory, and its result is represented by the flag Z.

•Format

BIT (Rreg)
BIT #(Rreg)
BIT (*ab*)
BIT #(*ab*)

•Operation

A ∧ [operand] → Z flag change

(EX) BIT (Xreg)

Xreg=4F00H

A= 1 0 0 0 0 0 0 0 Execute → Z=1

Contents of 4F00H = 0 0 0 0 1 1 1 1

⑲ BII (test BIt Immediate)

The contents of the accumulator or the external memory are ANDed with the immediate data, and its result is represented by state of the flag Z.

•Format

BII A,*i*
BII (Rreg),*i*
BII #(Rreg),*i*
BII (*ab*),*i*
BII #(*ab*),*i*

•Operation

[Operand] ∧ *i* → Z flag change

promsoft.com

2-4-4. Transfer and search instructions

⑳ LDA (LoaD Acc)

The contents of the RL register, RH register, or the internal memory are transferred to the accumulator.
Only the flag Z changes.

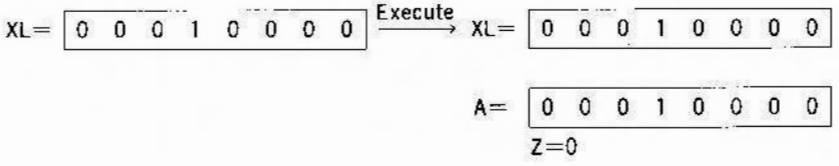
•Format

LDA RL
LDA RH
LDA (Rreg)
LDA #(Rreg)
LDA (*ab*)
LDA #(*ab*)

•Operation

[Operand] → A
Flag Z
1: when [operand] = 00H
0: when [operand] ≠ 00H

(EX) LDA XL



㉑ LDE (Load and DEcrement)

The contents of the external memory (R register) are transferred to the accumulator, then "1" is decremented from R register.
Only the flag Z changes.

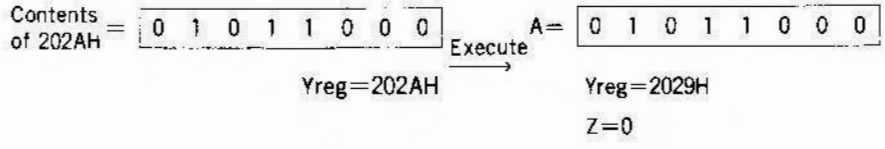
•Format

LDE Rreg

•Operation

(Rreg) → A
Rreg − 1 → Rreg
Flag Z
1: when (Rreg) = 00H
0: when (Rreg) ≠ 00H

(EX) LDE Yreg



promsoft.com

㉒ LIN (Load and INcrement)

The contents of the external memory (R register) are transferred to the accumulator, then "1" is added to the R register.
Only the flag Z changes.

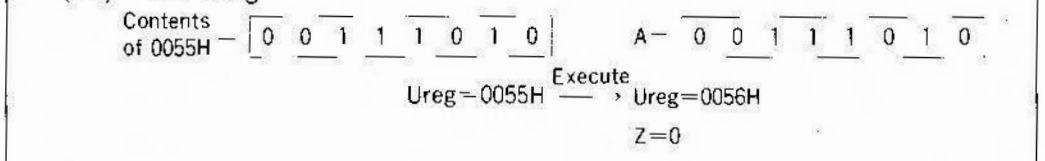
•Format

LIN Rreg

•Operation

(Rreg) → A
Rreg + 1 → Rreg
Flag Z
1: when (Rreg) = 00H
0: when (Rreg) ≠ 00H

(EX) LIN Ureg



㉓ LDI (LoaD Immediate)

The immediate data is loaded in the accumulator, RL register, RH register, or stack pointer S.
The flag Z changes when transfer is done to the accumulator and no flag change occurs in otherwise case.
Two bytes of the immediate value are transferred in the case of the stack pointer.

•Format

① LDI A,*i*

•Operation

i → A
Flag Z
1: when *i* = 00H
0: when *i* ≠ 00H

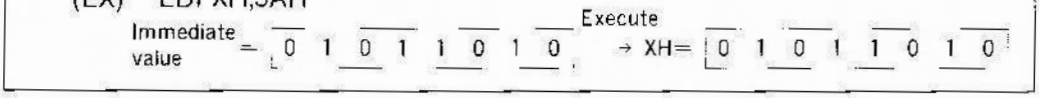
② LDI RL,*i*
LDI RH,*i*

i → RL
i → RH
(No flag changes)

③ LDI S,*i*, *j*

i → SH
j → SL
(No flag changes)

(EX) LDI XH,5AH



promsoft.com

㉔ LDX (LoaD Xreg)

The contents of the R register, stack pointer, or program counter are transferred to the X register.

No change takes place in flags.

•Format

LDX Rreg
LDX S
LDX P

•Operation

[Operand] → Xreg

(EX) LDX S

S=4700H $\xrightarrow{\text{Execute}}$ Xreg=4700H

㉕ STA (STore Acc)

The contents of the accumulator are transferred to the RL register, RH register, or the external memory.

No change takes place in flags.

•Format

STA RL
STA RH
STA (Rreg)
STA #(Rreg)
STA (*ab*)
STA #(*ab*)

•Operation

A → [operand]

(EX) STA XL

A= 0 0 1 1 0 0 1 1 $\xrightarrow{\text{Execute}}$ XL= 0 0 1 1 0 0 1 1

㉖ SDE (Store and DEcrement)

The contents of the accumulator are transferred to the external memory (Rreg), then "1" is decremented from Rreg.

No change takes place in flags.

•Format

SDE Rreg

•Operation

A → (Rreg)
Rreg − 1 → Rreg

(EX) SDE Yreg

A= 0 0 1 1 1 1 1 1 Execute Contents of 4700H − 0 0 1 1 1 1 1 1

Yreg=4700H

Yreg=46FFH

promsoft.com

㉗ SIN (Store and INcrement)

The contents of the accumulator are transferred to the external memory (Rreg), then "1" is incremented to Rreg.

No change takes place in flags.

•Format

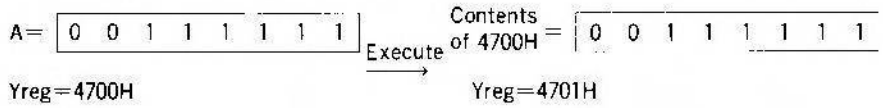
SIN Rreg

•Operation

A → (Rreg)

Rreg + 1 → Rreg

(EX) SIN Yreg



㉘ STX (STore Xreg)

The contents of Xreg are transferred to the R register, stack pointer, or program counter.

No change takes place in flags.

•Format

STX Rreg

STX S

STX P

•Operation

Xreg → [operand]

㉙ PSH (PuSH)

The contents of the accumulator or R register are stacked in the memory specified by the stack pointer.

In the case of the accumulator, the stack pointer is decremented by one. In the case of the R register, the stack pointer is decremented by two.

•Format

① PSH A

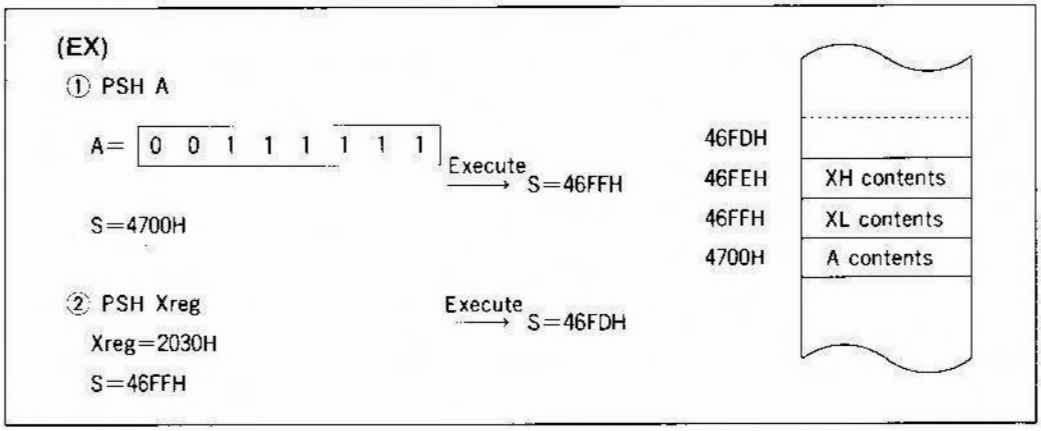
② PSH Rreg

•Operation

A → (S)
S − 1 → S

RL → (S)
S − 1 → S
RH → (S)
S − 1 → S

promsoft.com



㉚ POP (POP)

The contents of the stack pointer transferred by the PSH instruction are returned to the accumulator or the R register.

In the case of the accumulator, the stack pointer is added by one. In the case of the R register, the stack pointer is added by two.

•Format

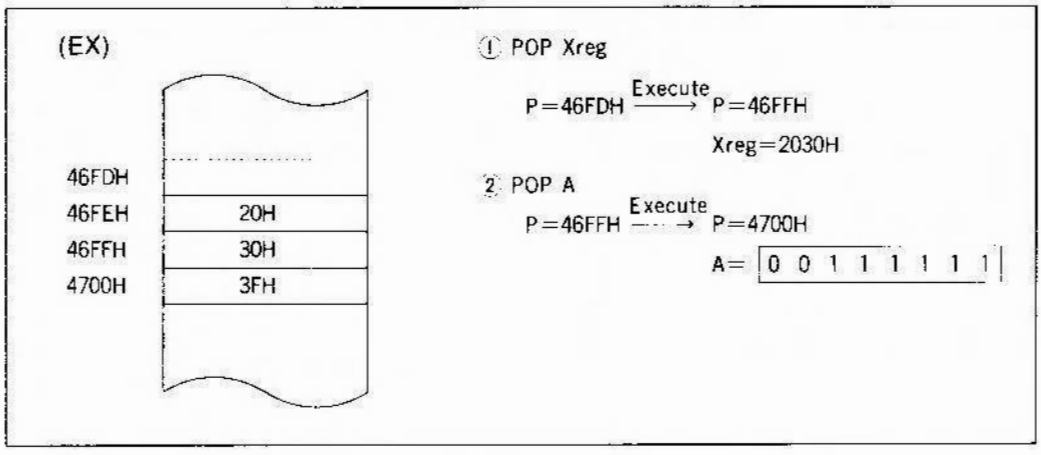
① POP A

② POP Rreg

•Operation

S + 1 → S
(S) → A
Flag Z
1: when (S) = 00H
0: when (S) ≠ 00H

S + 1 → S
(S) → RH
S + 1 → S
(S) → RL
(No flag change)



㉛ ATT (Acc To T)

The contents of the accumulator are transferred to the T register.

promsoft.com

•Format

ATT

•Operation

A → T

㉜ TTA (T To Acc)

The contents of the T register are transferred to the accumulator.
The flag Z changes.

•Format

TTA

•Operation

T → A

Flag Z

1: when T = 00H

0: when T ≠ 00H

2-4-5. Block transfer and search instructions

㉝ TIN (Transfer and INcrement)

The contents of the external memory (Xreg) are transferred to the external memory (Yreg), then both Xreg and Yreg are added by one.
No change takes place in flags.

•Format

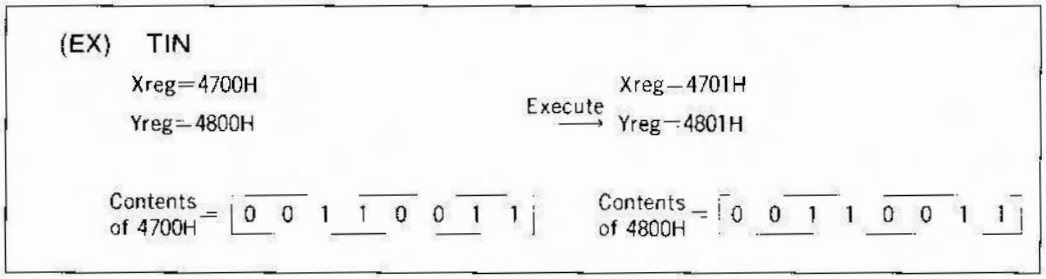
TIN

•Operation

(Xreg) → (Yreg)

Xreg + 1 → Xreg

Yreg + 1 → Yreg



㉞ CIN (Compare and INcrement)

The contents of the accumulator are compared with that of the external memory (Xreg), its result is represented by flag states, then Xreg is incremented by one.

•Format

CIN

•Operation

A − (Xreg) → change in C, H, Z, and V

Xreg + 1 → Xreg

Relation between the comparison result (A with Xreg) and flags (C, H, Z, V) is the same as in ⑯ CPA.

promsoft.com

2-4-6. Rotate and shift instructions

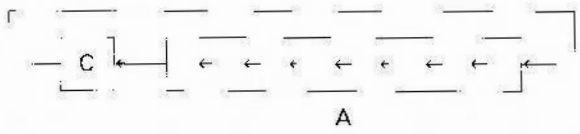
㉟ ROL (ROtate Left)

The accumulator contents are rotated left through the flag C.

• Format

ROL

• Operation



(EX) ROL



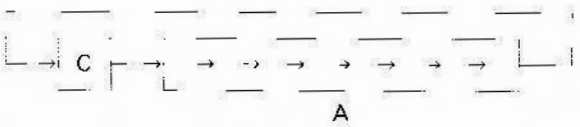
㊱ ROR (ROtate Right)

The accumulator contents are rotated right through the flag C.

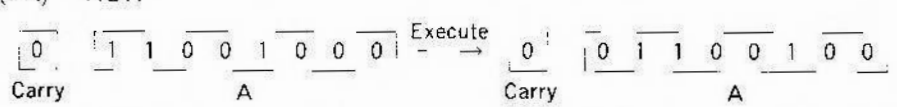
• Format

ROR

• Operation



(EX) ROR



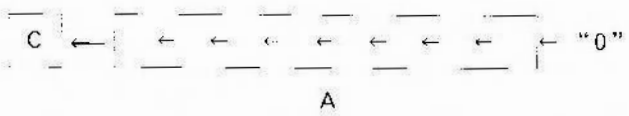
㊲ SHL (SHift Left)

The contents of the accumulator are shifted left.

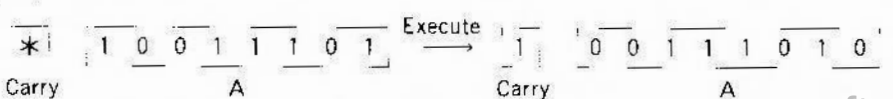
• Format

SHL

• Operation



(EX) SHL



promsoft.com

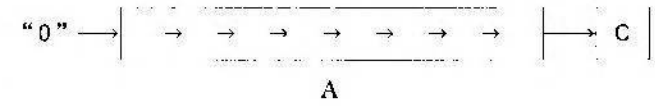
㊳ SHR (SHift Right)

The contents of the accumulator are shifted right.

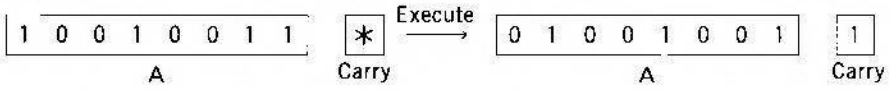
•Format

SHR

•Operation



(EX) SHR



㊴ DRL (Digit Rotate Left)

Left rotation takes between the accumulator and the external memory (Xreg) or #(Xreg) in unit of digit (4 bits).

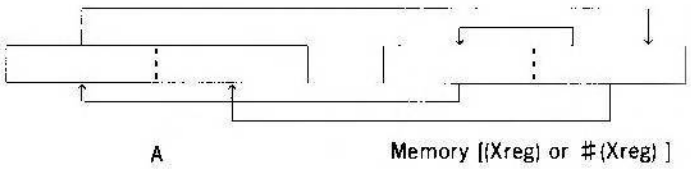
In other words, the low order 4 bits of the external memory are moved to the high order 4 bits of the external memory, the high order 4 bits of the external memory are moved to the high order 4 bits of the accumulator, the high order 4 bits of the accumulator are moved to the low order 4 bits of the external memory, and the low order 4 bits of the accumulator are moved to the low order 4 bits of the external memory at all times. No change takes place in flags.

•Format

DRL (Xreg)

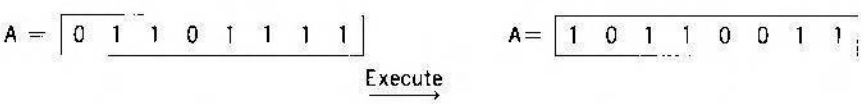
DRL #(Xreg)

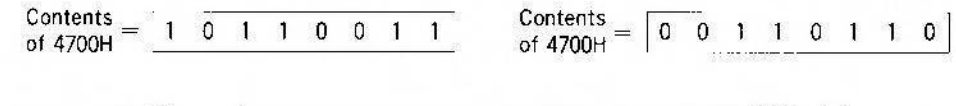
•Operation



(EX) DRL (Xreg)

Xreg=4700H





promsoft.com

㊵ DRR (Digit Rotate Right)

Right rotation takes between the accumulator and the external memory (Xreg) or #(Xreg) in unit of digit (4 bits).

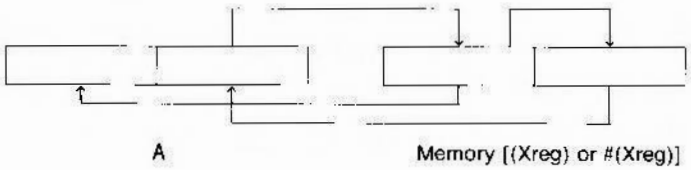
In other words, the low order 4 bits of the external memory are moved to the low order 4 bits of the accumulator, the low order 4 bits of the accumulator are moved to the high order 4 bits of the external memory, the high order 4 bits of the external memory are moved to the low order 4 bits of the external memory, and the high order 4 bits of the external memory are moved to the high order 4 bits of the accumulator memory at all times.

No change takes place in flags.

•Format

DRR (Xreg)
DRR #(Xreg)

•Operation



(EX) DRR (Xreg)

Xreg=4700H

A= 1 1 0 0 0 0 0 1 → Execute → A = 0 1 0 0 0 0 1 0

Contents of 4700H = 0 1 0 0 0 0 1 0 → Contents of 4700H = 0 0 0 1 0 1 0 0

㊶ AEX (EXchange Acc)

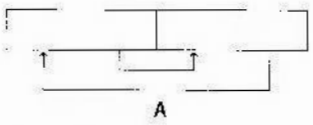
The high order 4 bits of the accumulator are swapped with the low order 4 bits.

No change takes place in flags.

•Format

AEX

•Operation



(EX) AEX

A= 1 1 0 0 1 0 1 0 → Execute → A= 1 0 1 0 1 1 0 0

promsoft.com

2-4-7. CPU control instructions

㊷ SEC (SEt Carry)

Sets the carry flag.

No change takes place in flags.

• Format	• Operation
SEC	1 → C

㊸ REC (REset Carry)

Resets the carry flag.

No change takes place in flags.

• Format	• Operation
REC	0 → C

㊹ CDV (Clear DiVider)

Clears the internal divider. In other words, since the CPU clock is supplied through the divider, it makes clock reset by the CDV instruction.

• Format	• Operation
CDV	0 → divider

㊺ ATP (Acc To Port)

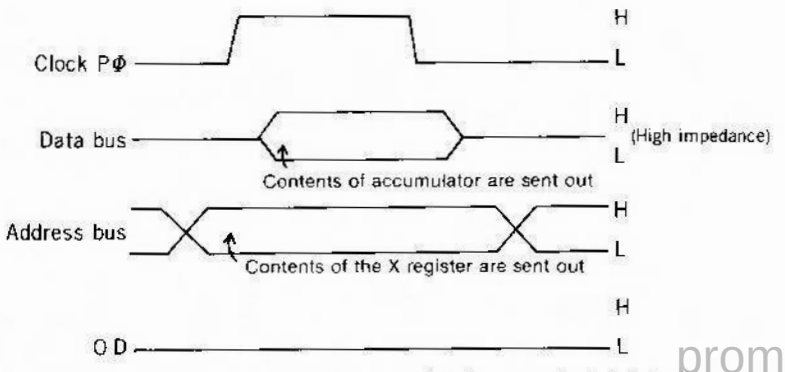
The contents of the accumulator are sent on the data bus. As the clock PΦ is sent out from the CPU at this moment, it may be used for the clock of the latch IC to comprise an output port.

No change takes place in flags.

• Format

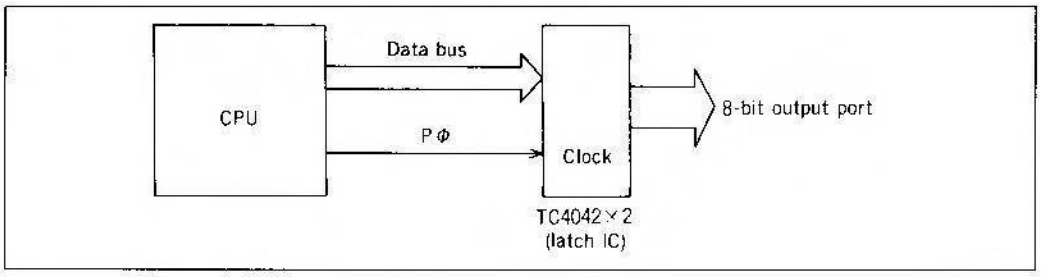
ATP

• Operation



NOTE: Though data is output with high state of OD (output disable) during memory write, OD is low state in the case of the ATP instruction.

promsoft.com



㊻ SPU (Set PU)

Sets the general purpose flipflop PU.
No change takes place in flags.

•Format
SPU

•Operation
1 → PU

㊼ RPU (Reset PU)

Resets the general purpose flipflop PU.
No change takes place in flags.

•Format
RPU

•Operation
0 → PU

㊽ SPV (Set PV)

Sets the general purpose flipflop PV.
No change takes place in flags.

•Format
SPV

•Operation
1 → PV

㊾ RPV (Reset PV)

Resets the general purpose flipflop PV.
No change takes place in flags.

•Format
RPV

•Operation
0 → PV

㊿ SDP (Set DisP)

Sets the LCD on off control flipflop DISP.

•Format
SDP

•Operation
1 → DISP
On pattern signal is generated from the CPU internal LCD backplate signal lines (H0~H7).

promsoft.com

⑤① RDP (Reset DisP)

Resets the LCD on/off control flipflop DISP.

•Format	•Operation
RDP	0 → DISP Off pattern signal is generated from the CPU internal LCD backplate signal lines (H0~H7).

⑤② ITA (In To Acc)

The contents of the input port (IN0~IN7) are transferred to the accumulator.
Only the flag Z changes.

•Format	•Operation
ITA	IN0~7 → Accumulator

⑤③ SIE (Set IE)

Sets the interrupt enable flag IE. After this, it becomes ready for maskable interrupt and timer interrupt acknowledge.
No change takes place in other flags.

•Format	•Operation
SIE	1 → IE

⑤④ RIE (Reset IE)

Resets the interrupt enable flag IE. After this, maskable interrupt and timer interrupt are disabled.
No change takes place in other flags.

•Format	•Operation
RIE	0 → IE

⑤⑤ AM0 (Acc to Tm and 0)

The contents of the accumulator are transferred to the timer register (TM). Since the timer register consists of 9 bits, the accumulator contents are transferred to the low order 8 bits of the register and "0" is entered in the highest order bit.
No change takes place in other flags.

•Format	•Operation
AM0	A → TM (TM0~TM7) 0 → TM8

promsoft.com

⑯ AM1 (Acc to Tm and 1)

Same as AM0, but "1" is entered in the highest order bit.
No change takes place in other flags.

•Format	•Operation
AM1	A → TM (TM0~TM7) 1 → TM8

⑰ NOP (No Operation)

⑱ HLT (HaLT)

Stops CPU operation. (Only the divider is in operation.)
Released from stop by interrupt.
No change takes place in flags.

⑲ OFF

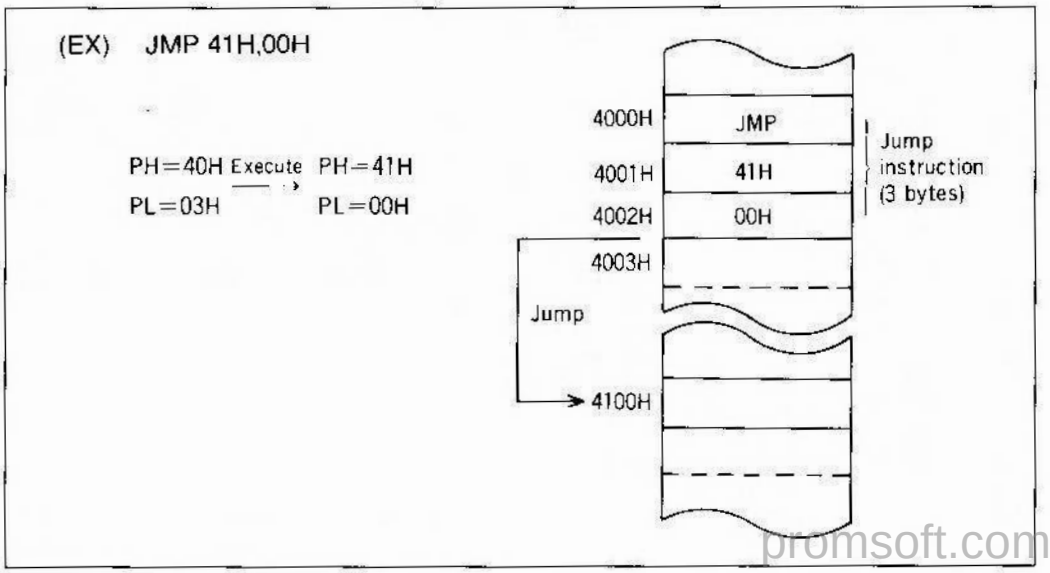
BF flipflop reset instruction.
No change takes place in flags.

2-4-8. Jump instructions

⑳ JMP (JuMP)

Jumps to a new program step represented by the second and third bytes of the immediate data *i*, *j*.
No change takes place in flags.

•Format	•Operation
JMP *i*, *j*	*i* → PH *j* → PL



promsoft.com

⑥① BCH (BranCH)

Jumps to a new program step which is indicated by the program counter of which value is added/subtracted by the value of the immediate data i, j. It will be possible to jump within a range of $-255<i<255$.

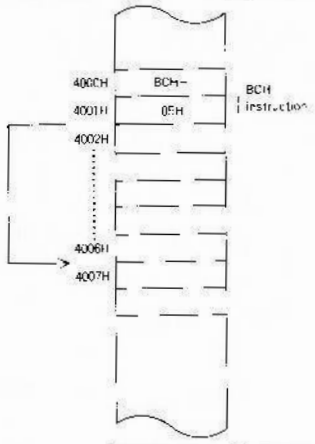
No change takes place in flags.

• Format	• Operation
BCH+i	P + i → P
BCH−i	P − i → P

(EX) BCH+5

P=4002H $\xrightarrow{\text{Execute}}$ P=4007H

As soon as the BCH instruction is read, the program counter becomes "P=4002H", and after execution of the BCH instruction, the next program execution starts from the address 4007H.



⑥② BCS (Branch if C Set)

Conditional relative address jump.

When C=1, it jumps to a program step represented by the program counter of which value is added/subtracted with the value of the immediate data.

If C=0, the control proceeds directly to a next program step without causing jump.

No change takes place in flags.

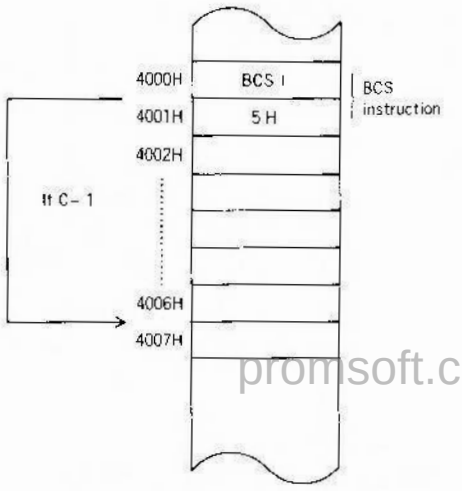
• Format	• Operation
BCS+i	C = {1: P + i → P; 0: P not changed.}
BCS−i	C = {1: P − i → P; 0: P not changed.}

(EX) BCS+5

P=4002H $\xrightarrow{\text{Execute}}$ P=4007H

C=1

As soon as the BCS instruction is read, the program counter becomes "P=4002H". If C=1 at that point, next program execution starts from the address 4007H. If C=0, it starts from 4002H.



promsoft.com

⑥③ BCR (Branch if C Reset)

If C=0, it jumps by relative address.
If C=1, it executes the next program step.
No change takes place in flags.

- **Format**

BCR+*i*
BCR−*i*

⑥④ BHS (Branch if H Set)

If H=1, it jumps by relative address.
If H=0, it executes the next program step.
No change takes place in flags.

- **Format**

BHS+*i*
BHS−*i*

⑥⑤ BHR (Branch if H Reset)

If H=0, it jumps by relative address.
If H=1, it executes the next program step.
No change takes place in flags.

- **Format**

BHR+*i*
BHR−*i*

⑥⑥ BZS (Branch if Z Set)

If Z=1, it jumps by relative address.
If Z=0, it executes the next program step.
No change takes place in flags.

- **Format**

BZS+*i*
BZS−*i*

⑥⑦ BZR (Branch if Z Reset)

If Z=0, it jumps by relative address.
If Z=1, it executes the next program step.
No change takes place in flags.

- **Format**

BZR+*i*
BZR−*i*

⑥⑧ BVS (Branch if V Set)

If V=1, it jumps by relative address.
If V=0, it executes the next program step.
No change takes place in flags.

- **Format**

BVS+*i*
BVS−*i*

promsoft.com

⑥⑨ BVR (Branch if V Reset)

If V=0, it jumps by relative address.
If V=1, it executes the next program step.
No change takes place in flags.

•Format

BVR+*i*
BVR−*i*

⑦⓪ LOP (LOoP)

If borrow is not produced after subtracting "1" from the UL register, the program counter is subtracted by the immediate data, then it jumps to relative address for next program execution.
If there is borrow, (UL<0), it proceeds to the succeeding program step.
No change takes place in flags.

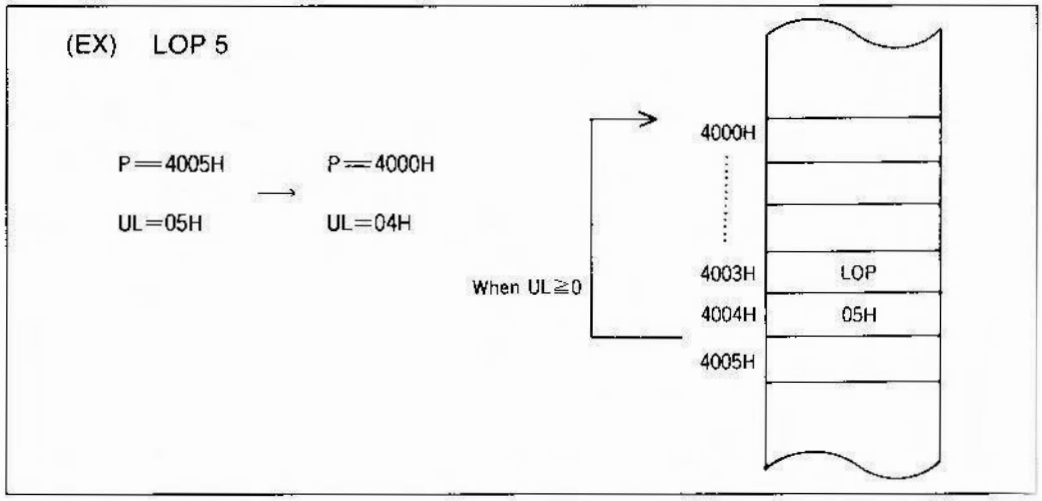
•Format

LOP *i*

•Operation

UL − 1 → UL

$$\text{Borrow} = \begin{cases} 0: \text{P} - i \rightarrow \text{P} \\ 1: \text{To succeeding step} \end{cases}$$



promsoft.com

2-4-9. Subroutine jump instructions

⑦① SJP (Subroutine JumP)

The contents of the program counter, which show the next program executing address, are stored in the stack pointer, then the control jumps to the subroutine address represented by *i* and *j* of the 16-bit immediate data.
No change takes place in flags.

•Format

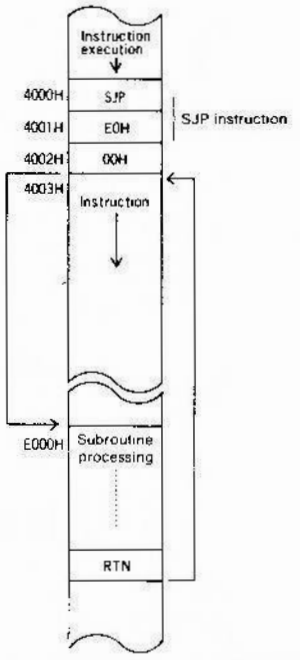
SJP *i*, *j*

•Operation

PL → (S)
S − 1
PH → (S)
S − 1
i → PH
j → PL

(EX) SJP E0H, 00H

4003H is stored in the stack pointer by the execution of STP instruction and jumps to the subroutine of E000H. With the RTN instruction of the subroutine, the control returns to 4003H to resume previous program execution.



promsoft.com

⑫ VEJ (VEctor subroutine Jump)

One step subroutine jump instruction that jumps to the address indicated by the two-byte vector, whose high order address byte is represented by FFH and low order address byte by the operand of the instruction.

The flag Z is reset.

There are 28 kinds of VEJ operand within two bytes range of 11000000 (C0H) to 11110110 (F6H). Therefore, the vector address table contains the address area of FFC0H to FFF6H.

• Format

VEJ *i*

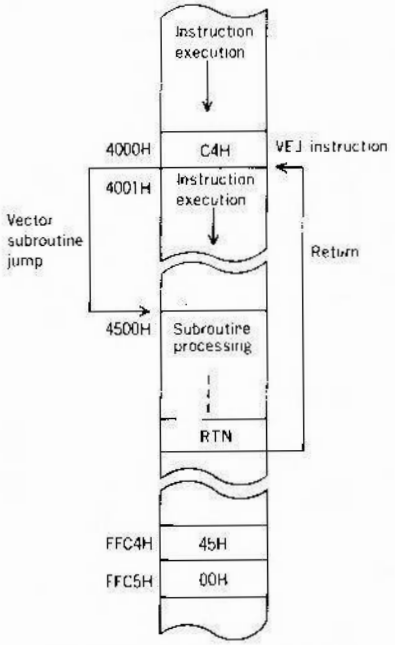
• Operation

PL → (S)
S − 1 → S
PH → (S)
S − 1 → S
(FF00H+*i*) → PH
(FF00H+*i*+1) → PL

NOTE: *i* has 28 kinds of VEJ operands.

(EX) VEJ C4H

The contents of FFC4H and FFC5H are interrogated at the address 4000H and jump to the subroutine of 4500H.
With the RTN instruction of the subroutine, the execution resumes from the address 4001H again.
Because it permits subroutine jump by a single byte of instruction, it helps to reduce program steps, when applied to too often appearing subroutine.



promsoft.com

⑬ VMJ (Vector 2 byte Subroutine Jump)

Jumps to the address indicated by the two-byte vector address, whose high order address byte is represented by FFH and low order address byte by the immediate data. The flag Z is reset.

Vector address table contains FF00H thru FFF6H. Immediate value *i* may take even number of 00H thru F6H.

•Format

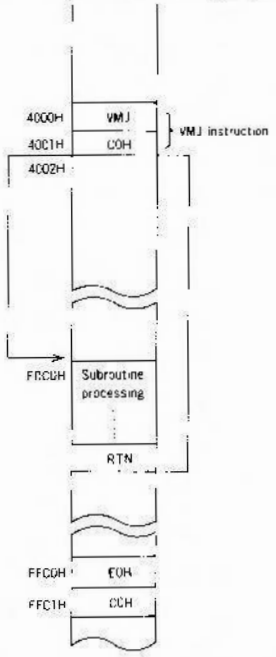
VMJ *i*

•Operation

PL → (S)
S − 1 → S
PH → (S)
S − 1 → S
(FF00H+*i*) → PH
(FF00H+*i*+1) → PL

(EX) VMJ C0H

The contents of FFC0H are interrogated by the VMJ instruction and jump to the subroutine of E0C0H.
With the RTN instruction of the subroutine, the execution resumes from the address 4002H again.
Because it permits subroutine jump by two bytes of instruction, it helps to reduce program steps, as compared with the SJP instruction.



promsoft.com

⑭ VCS (Vector subroutine jump if C Set)

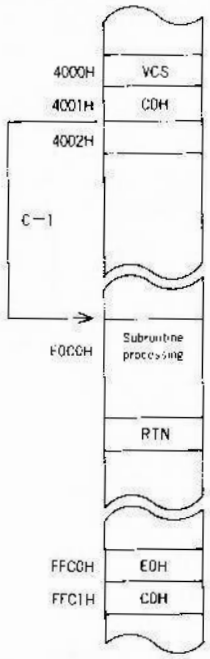
Conditional vector subroutine jump instruction.
If C=1, it performs the vector subroutine jump, the same as in the VMJ instruction.
If C=0, the control proceeds to the succeeding program step.

• Format

VCS *i*

(EX) VCS C0H

The contents of C are interrogated by the VCS instruction and perform the vector subroutine jump, if C=1.
If C=0, it continues to execute the program in the address 4002H.



⑮ VCR (Vector subroutine jump if C Reset)

If C=0, it performs the vector subroutine jump, the same as in the VMJ instruction.
If C=1, the control proceeds to the succeeding program step.

• Format

VCR *i*

⑯ VHS (Vector subroutine jump if H Set)

If H=1, it performs the vector subroutine jump, the same as in the VMJ instruction.
If H=0, the control proceeds to the succeeding program step.

• Format

VHS *i*

⑰ VHR (Vector subroutine jump if H Reset)

If H=0, it performs the vector subroutine jump, the same as in the VMJ instruction.
If H=1, the control proceeds to the succeeding program step.

• Format

VHR *i*

⑱ VZS (Vector subroutine jump if Z Set)

If Z=1, it performs the vector subroutine jump, the same as in the VMJ instruction.
If Z=0, the control proceeds to the succeeding program step.

promsoft.com

• Format

VZS *i*

⑦⑨ VZR (Vector subroutine jump if Z Reset)

If Z=0, it performs the vector subroutine jump, the same as in the VMJ instruction.
If Z=1, the control proceeds to the succeeding program step.

•Format

VZR *i*

⑧⓪ VVS (Vector subroutine jump if V Set)

If V=1, it performs the vector subroutine jump, the same as in the VMJ instruction.
If V=0, the control proceeds to the succeeding program step.

•Format

VVS *i*

2-4-10. Return instructions

⑧① RTN (ReTurN from subroutine)

The instruction used to return from the subroutine to the main routine.
No change takes place in flags.
The previous program address is gotten from the external memory stack to be transferred to the program counter.
The next instruction will be fetched from the address indicated by the program counter.

•Format

RTN

•Operation

S + 1 → S
(S) → PH
S + 1 → S
(S) → PL

⑧② RTI (ReTurn from Interrupt)

The instruction used to return from the interrupt service routine to the main routine. After executing the same procedure as in the RTN instruction, then the contents of the T register at the time of interrupt are gotten from the external memory stack to be transferred to the T register. Flags are also set to their previous states.

•Format

RTI

•Operation

S + 1 → S
(S) → PH
S + 1 → S
(S) → PL
S + 1 → S
(S) → T

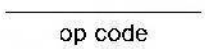
promsoft.com

2-5. Command list

List of LH5801 Microprocessor will be shown in pages to follow. There are following nine types of commands.

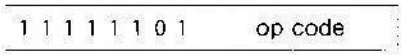
Single byte command

(1)



Two-byte command

(2)

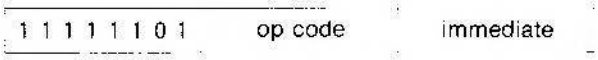


(3)

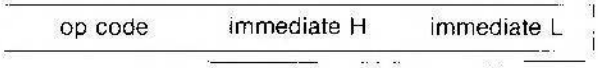


Three-byte command

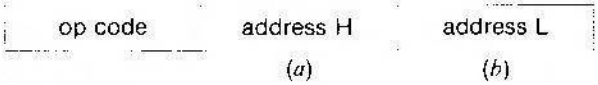
(4)



(5)

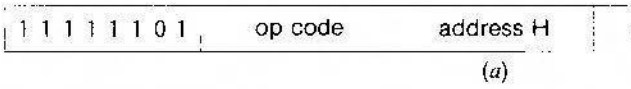


(6)



Four-byte command

(7)

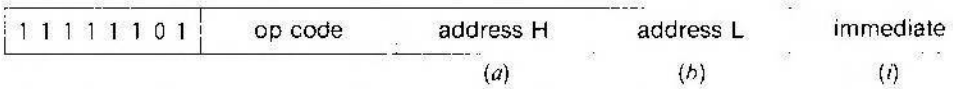


(8)



Five-byte command

(9)



promsoft.com

8-bit CPU command list (1)

Arithmetic/logical

MNEMONIC		SYMBOLIC OPERATION	STATUS C V H Z IE	MACHINE LANGUAGE 7 6 5 4 3 2 1 0	BYTE	CYCLE
ADC	R_L	A+R_L+C→A	○○○○−	0 0 R_L 0 0 1 0	1	6
	R_H	A+R_H+C→A		1 0 R_H 0 0 1 0	1	6
	(R)	A+(R)+C→A		0 0 R 0 0 1 1	1	7
	(a,b)	A←(a,b)+C→A		1 0 1 0 0 0 1 1	3	13
	‡(R)	A←‡(R)+C→A		FD 0 0 R 0 0 1 1	2	11
	‡(a,b)	A←‡(a,b)+C→A		FD 1 0 1 0 0 0 1 1	4	17
ADI	A,*i*	A←*i*+C→A		1 0 1 1 0 0 1 1	2	7
	(R),*i*	(R)+*i*→(R)		0 1 R 1 1 1 1	2	13
	(a,b),*i*	(a,b)+*i*→(a,b)		1 1 1 0 1 1 1 1	4	19
	‡(R),*i*	‡(R)+*i*→‡(R)		FD 0 1 R 1 1 1 1	3	17
	‡(a,b),*i*	‡(a,b)+*i*→‡(a,b)		FD 1 1 1 0 1 1 1 1	5	23
DCA	(R)	A+(R)+C→A(BCD)		1 0 R 1 1 0 0	1	15
	‡(R)	A+‡(R)+C→A(BCD)		FD 1 0 R 1 1 0 0	2	19
ADR		R_L+A→R_L (16-bit register operation) RH+1→RH, *if* C7		FD 1 1 R 1 0 1 0	2	11
SBC	R_L	A−R_L−$\overline{C}$→A	○○○○−	0 0 R_L 0 0 0 0	1	6
	R_H	A−R_H−$\overline{C}$→A		1 0 R_H 0 0 0 0	1	6
	(R)	A←(R)−$\overline{C}$→A		0 0 R 0 0 0 1	1	7
	(a,b)	A←(a,b)−$\overline{C}$→A		1 0 1 0 0 0 0 1	3	13
	‡(R)	A←‡(R)−C→A		FD 0 0 R 0 0 0 1	2	11
	‡(a,b)	A←‡(a,b)−C→A		FD 1 0 1 0 0 0 0 1	4	17
SBI	A,*i*	A←*i*−C→A		1 0 1 1 0 0 0 1	2	7
DCS	(R)	A←(R)−$\overline{C}$→A(BCD)		0 0 R 1 1 0 0	1	13
	‡(R)	A←‡(R)−$\overline{C}$→A(BCD)		FD 0 0 R 1 1 0 0	2	17
AND	(R)	A∧(R)→A	−−−○−	0 0 R 1 0 0 1	1	7
	(a,b)	A∧(a,b)→A		1 0 1 0 1 0 0 1	3	13
	‡(R)	A∧‡(R)→A		FD 0 0 R 1 0 0 1	2	11
	‡(a,b)	A∧‡(a,b)→A		FD 1 0 1 0 1 0 0 1	4	17
ANI	A,*i*	A∧*i*→A		1 0 1 1 1 0 0 1	2	7
	(R),*i*	(R)∧*i*→(R)		0 1 R 1 0 0 1	2	13
	(a,b),*i*	(a,b)∧*i*→(a,b)		1 1 1 0 1 0 0 1	4	19
	‡(R),*i*	‡(R)∧*i*→‡(R)		FD 0 1 R 1 0 0 1	3	17
	‡(a,b),*i*	‡(a,b)∧*i*→‡(a,b)		FD 1 1 1 0 1 0 0 1	5	23

COMMENT

- B5 B4 | R_L | R_H | R
  - 0 0 | X_L | X_H | X
  - 0 1 | Y_L | Y_H | Y
  - 1 0 | U_L | U_H | U
  - 1 1 | * | * | *

- Address of (a,b)

15 ····· 8	7 ····· 0
a	*b*
(High order)	(Low order)

- (R)···ME0 accessed
  ‡(R)···ME1 accessed

promsoft.com

8-bit CPU command list (2)

MNEMONIC		SYMBOLIC OPERATION	STATUS C V H Z IE	MACHINE LANGUAGE 7 6 5 4 3 2 1 0	BYTE	CYCLE	COMMENT
ORA	(R)	A∨((R))→A	---○-	00 R 1011	1	7	
	((a,b))	A∨((a,b))→A		10101011 FD	3	13	
	#(R)	A∨#((R))→A		00 R 1011 FD	2	11	
	#((a,b))	A∨#((a,b))→A		10101011	4	17	
ORI	A,i	A∨i→A		10111011	2	7	
	(R),i	((R))∨i→((R))		01 R 1011	2	13	
	((a,b)),i	((a,b))∨i→((a,b))		11101011 FD	4	19	
	#(R),i	#((R))∨i→#(R)		01 R 1011 FD	3	17	
	#((a,b)),i	#((a,b))∨i→#((a,b))		11101011	5	23	
EOR	(R)	A⊕((R))→A	---○-	00 R 1101	1	7	
	((a,b))	A⊕((a,b))→A		10101101 FD	3	13	
	#(R)	A⊕#((R))→A		00 R 1101 FD	2	11	
	#((a,b))	A⊕#((a,b))→A		10101101	4	17	
EAI	i	A⊕i→A		10111101	2	7	
INC	A	A+1→A	○○○○	11011101	1	5	
	RL	RL+1→RL		01 RL 0000 FD	1	5	
	RH	RH+1→RH		01 RH 0000	2	9	
	R	R+1→R	---	01 R 0100	1	5	
DEC	A	A−1→A	○○○○	11011111	1	5	
	RL	RL−1→RL		01 RL 0010 FD	1	5	
	RH	RH−1→RH		01 RH 0010	2	9	
	R	R−1→R	---	01 R 0110	1	5	

Compare and bit test

CPA	RL	A−RL	○○○○	00 RL 0110	1	6	
	RH	A−RH		10 RH 0110	1	6	
	(R)	A−(R)		00 R 0111	1	7	
	((a,b))	A−((a,b))		10100111 FD	3	13	
	#(R)	A−#(R)		00 R 0111 FD	2	11	
	#((a,b))	A−#((a,b))		10100111	4	17	
CPI	RL,i	RL−i		01 RL 1110	2	7	
	RH,i	RH−i		01 RH 1100	2	7	
	A,i	A−i		10110111	2	7	
BIT	(R)	A∧((R))→Z	---○-	00 R 1111	1	7	
	((a,b))	A∧((a,b))→Z		10101111 FD	3	13	
	#(R)	A∧#((R))→Z		00 R 1111 FD	2	11	
	#((a,b))	A∧#((a,b))→Z		10101111	4	17	
BII	A,i	A∧i→Z		10111111	2	7	
	(R),i	((R))∧i→Z		01 R 1101	2	10	
	((a,b)),i	((a,b))∧i→Z		11101101 FD	4	16	
	#(R),i	#((R))∧i→Z		01 R 1101 FD	3	14	
	#((a,b)),i	#((a,b))∧i→Z		11101101	5	20	

promsoft.com

8-bit CPU command list (3)

Load and store

MNEMONIC		SYMBOLIC OPERATION	STATUS C V H Z IE	MACHINE LANGUAGE 7 6 5 4 3 2 1 0	BYTE	CYCLE	COMMENT
LDA	R_L	$R_L \rightarrow A$	---○-	00 R_L 0100	1	5	
	R_H	$R_H \rightarrow A$		10 R_H 0100	1	5	
	(R)	$(R) \rightarrow A$		00 R 0101	1	6	
	(a,b)	$(a,b) \rightarrow A$		10100101	3	12	
	#(R)	$\#(R) \rightarrow A$		FD 00 R 0101	2	10	
	#(a,b)	$\#(a,b) \rightarrow A$		FD 10100101	4	16	
LDE	R	$(R) \rightarrow A, R-1 \rightarrow R$		01 R 0111	1	6	
LIN	R	$(R) \rightarrow A, R+1 \rightarrow R$		01 R 0101	1	6	
LDI	R_L,i	$i \rightarrow R_L$	----	01 R_L 1010	2	6	
	R_H,i	$i \rightarrow R_H$		01 R_H 1000	2	6	
	A,i	$i \rightarrow A$	--○-	10110101	2	6	
	S,i,j	$ij \rightarrow S$	----	10101010	3	12	
LDX	R	$R \rightarrow X$		FD 00 R 1000	2	11	
	S	$S \rightarrow X$		FD 01001000	2	11	
	P	$P \rightarrow X$		FD 01011000	2	11	
STA	R_L	$A \rightarrow R_L$	----	00 R_L 1010	1	5	
	R_H	$A \rightarrow R_H$		00 R_H 1000	1	5	
	(R)	$A \rightarrow (R)$		00 R 1110	1	6	
	(a,b)	$A \rightarrow (a,b)$		10101110	3	12	
	#(R)	$A \rightarrow \#(R)$		FD 00 R 1110	2	10	
	#(a,b)	$A \rightarrow \#(a,b)$		FD 10101110	4	16	
SDE	R	$A \rightarrow (R), R-1 \rightarrow R$		01 R 0011	1	6	
SIN	R	$A \rightarrow (R), R+1 \rightarrow R$		01 R 0001	1	6	
STX	R	$X \rightarrow R$		FD 01 R 1010	2	11	
	S	$X \rightarrow S$		FD 01001110	2	11	
	P	$X \rightarrow P$		FD 01011110	2	11	
PSH	A	$A \rightarrow (S), S-1 \rightarrow S$		FD 11001000	2	11	
	R	$R_L \rightarrow (S), RH \rightarrow (S-1),$ $S-2 \rightarrow S$		FD 10 R 1000	2	14	
POP	A	$(S+1) \rightarrow A, S+1 \rightarrow S$	---○-	FD 10001010	2	12	
	R	$(S+1) \rightarrow RH, (S+2) \rightarrow RL,$ $S+2 \rightarrow S$	-- --	FD 00 R 1010	2	15	
ATT		$A \rightarrow T(STATUS)$	○○○○○	FD 11101100	2	9	
TTA		$T(STATUS) \rightarrow A$	--○-	FD 10101010	2	9	

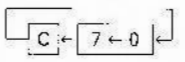
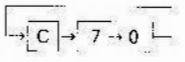
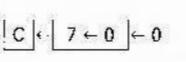
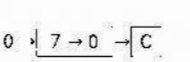
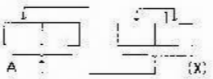
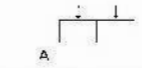
Block transfer and search

TIN		$(X) \rightarrow (Y)$, $X+1 \rightarrow X$, $Y+1 \rightarrow Y$	-- --	11110101	1	7	
CIN		$A \rightarrow (X), X+1 \rightarrow X$	○○○○-	11110111	1	7	

promsoft.com

8-bit CPU command list (4)

Rotate and shift

MNEMONIC	SYMBOLIC OPERATION	STATUS C V H Z IE	MACHINE LANGUAGE 7 6 5 4 3 2 1 0	BYTE	CYCLE	COMMENT
ROL		○○○○–	1 1 0 1 1 0 1 1	1	8	
ROR			1 1 0 1 0 0 0 1	1	9	
SHL			1 1 0 1 1 0 0 1	1	6	
SHR			1 1 0 1 0 1 0 1	1	9	
DRL			1 1 0 1 0 1 1 1	1	12	
DRL =	ME1 Area		FD 1 1 0 1 0 1 1 1	2	16	
DRR			1 1 0 1 0 0 1 1	1	12	
DRR =	ME1 Area		FD 1 1 0 1 0 0 1 1	2	16	
AEX			1 1 1 1 0 0 0 1	1	6	

CPU control

AM0	A →TIMER(T0~T7),0 →T8	– – – –	FD 1 1 0 0 1 1 1 0	2	9	
AM1	〃 1 →T8		FD 1 1 0 1 1 1 1 0	2	9	
CDV	divider clear		FD 1 0 0 0 1 1 1 0	2	8	
ATP	A →Output port (Clock output)		FD 1 1 0 0 1 1 0 0	2	9	
SDP	1 →Disp		FD 1 1 0 0 0 0 0 1	2	8	
RDP	0 →Disp		FD 1 1 0 0 0 0 0 0	2	8	
SPU	1→PU		1 1 1 0 0 0 0 1	1	4	
RPU	0→PU		1 1 1 0 0 0 1 1	1	4	
SPV	1→PV		1 0 1 0 1 0 0 0	1	4	
RPV	0→PV		1 0 1 1 1 0 0 0	1	4	
ITA	IN →A	○	FD 1 0 1 1 1 0 1 0	2	9	
RIE	0→IE	– – – – ○	FD 1 0 1 1 1 1 1 0	2	8	
SIE	1→IE	– – – – ○	FD 1 0 0 0 0 0 0 1	2	8	
HLT			FD 1 0 1 1 0 0 0 1	2	9	
OFF			FD 0 1 0 0 1 1 0 0	2	8	
NOP			0 0 1 1 1 0 0 0	1	5	
SEC	1→C	○ – – – –	1 1 1 1 1 0 1 1	1	4	
REC	0 →C	○	1 1 1 1 1 0 0 1	1	4	

promsoft.com

8-bit CPU command list (5)

Jump

MNEMONIC	SYMBOLIC OPERATION	STATUS C V H Z I E	MACHINE LANGUAGE 7 6 5 4 3 2 1 0	BYTE	CYCLE	COMMENT
JMP	$ij \rightarrow P$	- - - - - -	1 0 1 1 1 0 1 0	3	12	
						$s=0$: $+i$
BCH	$s=0$: $P+i \rightarrow P$		1 0 0 s 1 1 1 0	2	8	$s=1$: $-i$
	$s=1$: $P-i \rightarrow P$				9	(Includes one more cycle)
BCS	*if* C=1, $P \pm i \rightarrow P$		1 0 0 s 0 0 1 1	2	8/10/11	
	if C=0, continue					
BCR	*if* C=0, $P \pm i \rightarrow P$		1 0 0 s 0 0 0 1	2	8/10/11	B3 B2 B1 | Condition 0 0 0 | NC: non carry 0 0 1 | C: carry 0 1 0 | NH: non harf 0 1 1 | H: half 1 0 0 | NZ: non zero 1 0 1 | NV: zero 1 1 0 | NV: non overflow 1 1 1 | V: overflow
	if C=1, continue					
BVS	*if* V=1, $P \pm i \rightarrow P$		1 0 0 s 1 1 1 1	2	8/10/11	
	if V=0, continue					
BVR	*if* V=0, $P \pm i \rightarrow P$		1 0 0 s 1 1 0 1	2	8/10/11	
	if V=1, continue					
BHS	*if* H=1, $P \pm i \rightarrow P$		1 0 0 s 0 1 1 1	2	8/10/11	
	if H=0, continue					
BHR	*if* H=0, $P+i \rightarrow P$		1 0 0 s 0 1 0 1	2	8/10/11	
	if H=1, continue					
BZS	*if* Z=1, $P \pm i \rightarrow P$		1 0 0 s 1 0 1 1	2	8/10/11	
	if Z=0, continue					
BZR	*if* Z=0, $P \pm i \rightarrow P$		1 0 0 s 1 0 0 1	2	8/10/11	
	if Z=1, continue					
LOP UL,*i*	UL−1 →UL		1 0 0 0 1 0 0 0	2	8/11	
	if Borrow=0, $P-i \rightarrow P$					
	if Borrow=1, continue					

Call

SJP	$P_L \rightarrow (S)$, $P_H \rightarrow (S-1)$, $S-2 \rightarrow S$, $ij \rightarrow P$	- - - - - -	1 0 1 1 1 1 1 0	3	19	• Vector address (Q) VEJ: FF→q_H 1 1 i 0 →q_L VMJ: FF→q_H etc. $i \rightarrow q_L$
VEJ	$P_L \rightarrow (S)$, $P_H \rightarrow (S-1)$, $S-2 \rightarrow S$, $(q) \rightarrow P_H$, $(q+1) \rightarrow P_L$	- - - C - -	1 1 ← i → 0	1	17	
VCS	*if* C=1, $(q) \rightarrow P_H \rightarrow (S-1)$, $(q+1) \rightarrow P_L \rightarrow (S)$, $S-2 \rightarrow S$		1 1 0 0 0 0 1 1	2	8/21	
VCR	*if* C=0, 〃		1 1 0 0 0 0 0 1	2	8/21	
VHS	*if* H=1, 〃		1 1 0 0 0 1 1 1	2	8/21	
VHR	*if* H=0, 〃		1 1 0 0 0 1 0 1	2	8/21	
VZS	*if* Z=1, 〃		1 1 0 0 1 0 1 1	2	8/21	
VZR	*if* Z=0, 〃		1 1 0 0 1 0 0 1	2	8/21	
VVS	*if* V=1, 〃		1 1 0 0 1 1 1 1	2	8/21	
VMJ	$(q) \rightarrow P_H \rightarrow (S-1)$, $S-2 \rightarrow S$, $(q+1) \rightarrow P_L \rightarrow (S)$		1 1 0 0 1 1 0 1	2	20	

Return

RTN	$(S+1) \rightarrow P_H$, $(S+2) \rightarrow P_L$, $S+2 \rightarrow S$	- - - - - -	1 0 0 1 1 0 1 0	1	11	
RTI	$(S+1) \rightarrow P_H$, $(S+2) \rightarrow P_L$, $(S+3) \rightarrow T$, $S+3 \rightarrow S$	O O O O O	1 0 0 0 1 0 1 0	1	14	

promsoft.com

NOTE: P in above list indicates a succeeding byte. For a command accompanying the immediate value, it indicates the byte that follows to the immediate value.

LH5801

MNEMONIC		MACHINE LANGUAGE	MNEMONIC		MACHINE LANGUAGE	MNEMONIC		MACHINE LANGUAGE
ADC	XL	02	ANI	(*ab*)	E9 *a b i*	BVR	−	9D *i*
	YL	12		#(X)	FD 49 *i*	BZS	+	8B *i*
	UL	22		#(Y)	FD 59 *i*		−	9B *i*
	XH	82		#(U)	FD 69 *i*	BZR	+	89 *i*
	YH	92		#(*ab*)	FD E9 *a b i*		−	99 *i*
	UH	A2	AM0		FD CE	CDV		FD 8E
	(X)	03	AM1		FD DE	CIN		F7
	(Y)	13	ATP		FD CC	CPA	XL	06
	(U)	23	ATT		FD EC		YL	16
	(*ab*)	A3 *a b*	BCH	+	8E *i*		UL	26
	#(X)	FD 03		−	9E *i*		XH	86
	#(Y)	FD 13	BCS	+	83 *i*		YH	96
	#(U)	FD 23		−	93 *i*		UH	A6
	#(*ab*)	FD A3 *a b*	BCR	+	81 *i*		(X)	07
ADI	A	B3 *i*		−	91 *i*		(Y)	17
	(X)	4F *i*	BHS	+	87 *i*		(U)	27
	(Y)	5F *i*		−	97 *i*		(*ab*)	A7 *a b*
	(U)	6F *i*	BHR	+	85 *i*		#(X)	FD 07
	(*ab*)	EF *a b i*		−	95 *i*		#(Y)	FD 17
	#(X)	FD 4F *i*	BII	A	BF *i*		#(U)	FD 27
	#(Y)	FD 5F *i*		(X)	4D *i*		#(*ab*)	FD A7 *a b*
	#(U)	FD 6F *i*		(Y)	5D *i*	CPI	A	B7 *i*
	#(*ab*)	FD EF *a b i*		(U)	6D *i*		XL	4E *i*
ADR	X	FD CA		(*ab*)	ED *a b i*		YL	5E *i*
	Y	FD DA		#(X)	FD 4D *i*		UL	6E *i*
	U	FD EA		#(Y)	FD 5D *i*		XH	4C *i*
AEX		F1		#(U)	FD 6D *i*		YH	5C *i*
AND	(X)	09		#(*ab*)	FD ED *a b i*		UH	6C *i*
	(Y)	19	BIT	(X)	0F	DCA	(X)	8C
	(U)	29		(Y)	1F		(Y)	9C
	(*ab*)	A9 *a b*		(U)	2F		(U)	AC
	#(X)	FD 09		(*ab*)	AF *a b*		#(X)	FD 8C
	#(Y)	FD 19		#(X)	FD 0F		#(Y)	FD 9C
	#(U)	FD 29		#(Y)	FD 1F		#(U)	FD AC
	#(*ab*)	FD A9 *a b*		#(U)	FD 2F	DCS	(X)	0C
ANI	A	B9 *i*		#(*ab*)	FD AF *a b*		(Y)	1C
	(X)	49 *i*	BVS	+	8F *i*		(U)	2C
	(Y)	59 *i*		−	9F *i*		#(X)	FD 0C
	(U)	69 *i*	BVR	+	8D *i*		#(Y)	FD 1C

promsoft.com

MNEMONIC		MACHINE LANGUAGE
DCS	#(U)	FD 2C
DEC	A	DF
	XL	42
	YL	52
	UL	62
	XH	FD 42
	YH	FD 52
	UH	FD 62
	X	46
	Y	56
	U	66
DRL	(X)	D7
	#(X)	FD D7
DRR	(X)	D3
	=(X)	FD D3
EAI		BD *i*
EOR	(X)	0D
	(Y)	1D
	(U)	2D
	(*ab*)	AD *a b*
	=(X)	FD 0D
	#(Y)	FD 1D
	#(U)	FD 2D
	=(*ab*)	FD AD *a b*
HLT		FD B1
INC	A	DD
	XL	40
	YL	50
	UL	60
	XH	FD 40
	YH	FD 50
	UH	FD 60
	X	44
	Y	54
	U	64
ITA		FD BA
JMP		BA *i j*
LDA	XL	04
	YL	14

MNEMONIC		MACHINE LANGUAGE
LDA	UL	24
	XH	84
	YH	94
	UH	A4
	(X)	05
	(Y)	15
	(U)	25
	(*ab*)	A5 *a b*
	#(X)	FD 05
	=(Y)	FD 15
	#(U)	FD 25
	#(*ab*)	FD A5 *a b*
LDI	A	B5 *i*
	XL	4A *i*
	YL	5A *i*
	UL	6A *i*
	XH	48 *i*
	YH	58 *i*
	UH	68 *i*
	S	AA *i j*
LDE	X	47
	Y	57
	U	67
LDX	X	FD 08
	Y	FD 18
	U	FD 28
	S	FD 48
	P	FD 58
LIN	X	45
	Y	55
	U	65
LOP	UL	88 *i*
NOP		38
OFF		FD 4C
ORA	(X)	0B
	(Y)	1B
	(U)	2B
	(*ab*)	AB *a b*
	=(X)	FD 0B

MNEMONIC		MACHINE LANGUAGE
ORA	#(Y)	FD 1B
	#(U)	FD 2B
	=(*ab*)	FD AB *a b*
ORI	A	BB *i*
	(X)	4B *i*
	(Y)	5B *i*
	(U)	6B *i*
	(*ab*)	EB *a b i*
	=(X)	FD 4B
	#(Y)	FD 5B
	#(U)	FD 6B
	=(*ab*)	FD EB *a b i*
POP	A	FD 8A
	X	FD 0A
	Y	FD 1A
	U	FD 2A
PSH	A	FD C8
	X	FD 88
	Y	FD 98
	U	FD A8
RDP		FD C0
REC		F9
RIE		FD BE
ROL		DB
ROR		D1
RPU		E3
RPV		B8
RTI		8A
RTN		9A
SBC	XL	00
	YL	10
	UL	20
	XH	80
	YH	90
	UH	A0
	(X)	01
	(Y)	11
	(U)	21
	(*ab*)	A1 *a b*

promsoft.com

MNEMONIC		MACHINE LANGUAGE
SBC	#(X)	FD 01
	#(Y)	FD 11
	#(U)	FD 21
	#(*ab*)	FD A1 *a b*
SBI		B1 *i*
SDE	X	43
	Y	53
	U	63
SDP		FD C1
SEC		FB
SHL		D9
SHR		D5
SIE		FD 81
SIN	X	41
	Y	51
	U	61
SJP		BE *i j*
SPU		E1
SPV		A8
STA	XL	0A
	YL	1A
	UL	2A
	XH	08
	YH	18
	UH	28
	(X)	0E
	(Y)	1E
	(U)	2E
	(*ab*)	AE *a b*
	#(X)	FD 0E
	#(Y)	FD 1E
	#(U)	FD 2E
	#(*ab*)	FD AE *a b*
STX	X	FD 4A
	Y	FD 5A
	U	FD 6A
	S	FD 4E
	P	FD 5E
TIN		F5
TTA		FD AA
VCS		C3 *i*
VCR		C1 *i*
VEJ	C0	C0
	C2	C2
	C4	C4
	C6	C6
	C8	C8
	CA	CA
	CC	CC
	CE	CE
	D0	D0
	D2	D2
	D4	D4
	D6	D6
	D8	D8
	DA	DA
	DC	DC
	DE	DE
	E0	E0
	E2	E2
	E4	E4
	E6	E6
	E8	E8
	EA	EA
	EC	EC
	EE	EE
	F0	F0
	F2	F2
	F4	F4
	F6	F6
VMJ		CD *i*
VVS		CF *i*
VZS		CB *i*
VZR		C9 *i*
VHR		C5 *i*
VHS		C7 *i*

MNEMONIC	MACHINE LANGUAGE

promsoft.com

2-6. Electrical characteristics and timings

Absolute maximum ratings

Parameter	Symbol	Limits	Unit
Supply voltage	V_{CC}	−0.3 to +7	V
Input voltage	V_{IN}	0.3 to +7	V
Output voltage	V_{OUT}	−0.3 to +7	V
Operating temperature	Topr	0 to +40	°C
Storage temperature	Tstg	−55 to +150	°C

Electrical characteristics

DC characteristics (Ta=0 to 40°C, $V_{CC}-V_{GG}$=4.5V ±0.5V)

Parameter	Symbol	Min.	Typ.	Max.	Unit	Test conditions	Applicable pins
Supply current during operation	I_{CC}		7	15	mA	3.80MHz crystal in connection	
Supply current during halt	I_{HLT}		4		mA		
Input voltage	V_{IH}	V_{CC}-1.0			V		D0~7, BFI, RESET, HIN, WAIT, NMI, MI, IN0~7
	V_{IL}			0.4	V		
Output voltage	V_{OH}	2.4			V	I_{OH}=400μA	AD0~15, OPF, BFO, R/W, OD, ME0~1, PΦ, PU, PV, ΦOS, HA, DISP, D0~7
	V_{OL}			0.4	V	I_{OL}=1.6mA	
Input current	$\|I_{LI}\|$			1.0	μA	$V_{IH}=V_{CC}$	Input pins other than RESET, BFI
				5.0	μA		RESET, BFI
				1.0	μA	V_{IL}=0	Input pins other than IN0~7, RESET, BFI
			30	60	μA		IN0~7
Power switch ON resistance	R_{VA}			300	Ω		VA
	R_{VB}			300	Ω		VB
LCD drive ON resistance	R_H			3.5	kΩ	H_i~V_{CC}	H0~7
	R_M			3.5	kΩ	H_i~V_M	
	R_I			5.0	kΩ	H_i~V_{DIS}	
Supply current during standby	I_{ST}			5	μA	V_{CC}=0V, V_{GG}=5.5V	
3-state output leakage current	$\|I_{LO}\|$			1.0	μA		AD0~15, D0~7, R/W, ME0~1, OD

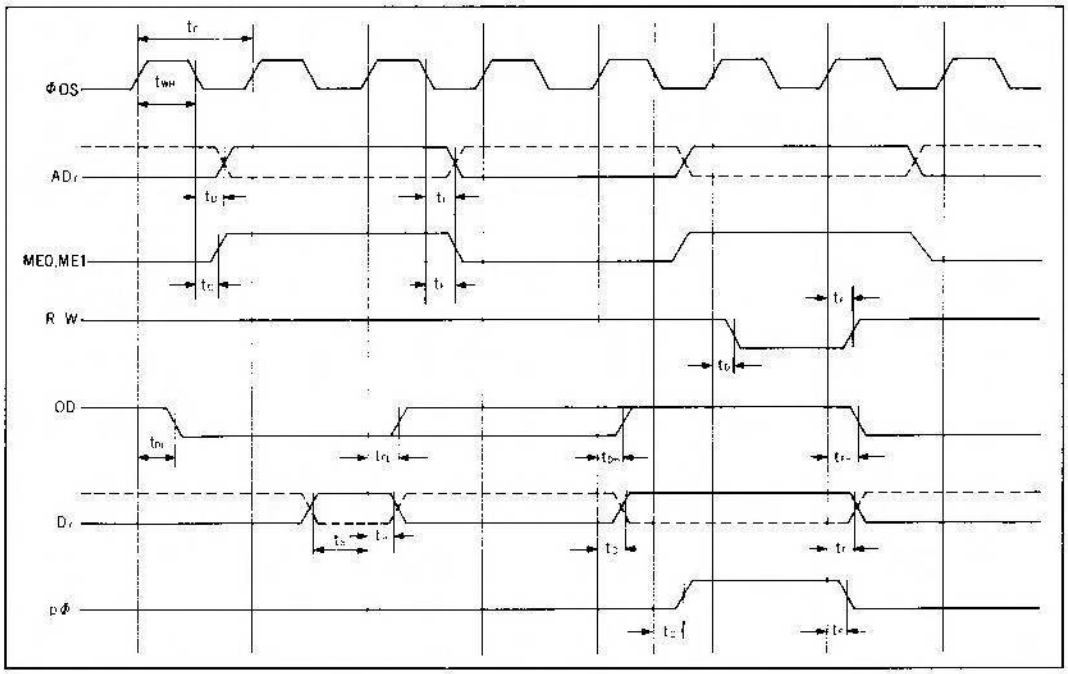
promsoft.com

AC characteristics

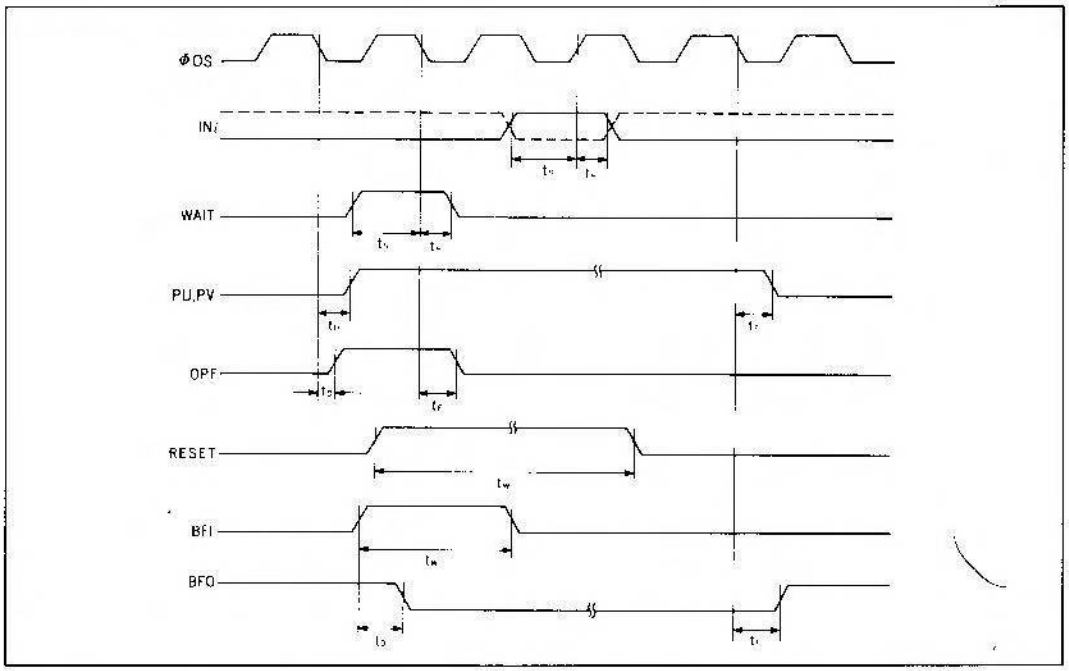
Parameter	Symbol	Min.	Max.	Unit	Test conditions
ΦOS	T_C	667			C_L : 20pF
	tr		150	nS	tr : (V_{CC}×0.1)V→(V_{CC}×0.9)V
	tf		40		tf : (V_{CC}×0.9)V→(V_{CC}×0.1)V
	t_{WH}	220			
AD0~15	t_D		250	nS	C_L : 20pF
	t_F	80			V_{OH} : (V_{CC}×0.5)V
					V_{OL} : 0.4V
ME0,1	t_D		260	nS	C_L : 20pF
	t_F	80	175		V_{OH} : (V_{CC}×0.5)V
					V_{OL} : 0.4V
OD	t_{DL}		250	nS	C_L : 20pF
	t_{FL}	170			V_{OH} : (V_{CC}×0.8)V
	t_{DH}		370		V_{OL} : 0.4V
	t_{FH}	100			
R/W	t_D	−30	50	nS	C_L : 20pF
	t_F		200		V_{OH} : (V_{CC}×0.8)V
					V_{OL} : 0.4V
D0~7	t_D		600	nS	C_L : 20pF
	t_F	220			V_{OH} : (V_{CC}×0.8)V
	t_S	170			V_{OL} : 0.4V
	t_H	100			
PΦ	t_D		350	nS	C_L : 20pF
	t_F		100		V_{OH} : (V_{CC}×0.8)V
					V_{OL} : 0.4V
IN0~7	t_S	190		nS	C_L : 20pF
	t_H	30			V_{IH} : (V_{CC}−1.0)V
					V_{IL} : 0.4V
WAIT	t_S	130		nS	C_L : 20pF
	t_H	20			V_{IH} : (V_{CC}−1.0)V
					V_{IL} : 0.4V
PU, PV	t_D		340	nS	C_L : 20pF
	t_F	50			V_{OH} : (V_{CC}×0.8)V
					V_{OL} : 0.4V
OPF	t_D		310	nS	C_L : 20pF
	t_F		190		V_{OH} : (V_{CC}×0.8)V
					V_{IH} : 0.4V
RESET	t_W	2		mS	C_L : 20pF
BFI	t_W	250		nS	C_L : 20pF
BFO	t_D		150	nS	C_L : 20pF
	t_F		360		V_{OH} : (V_{CC}×0.8)V
					V_{OL} : 0.4V

promsoft.com

Timing (1)



Timing (2)



promsoft.com

promsoft.com

3

LH5810/LH5811 I/O PORT CONTROLLER

promsoft.com

3-1. Outline

The LH5810/LH5811 is a single chip CMOS static LSI that features the following functions:

(1) two pairs of 8-bit bidirectional port
(2) one pair of 8-bit output port
(3) two interrupt request inputs (one of them port input)
(4) one interrupt request output
(5) CPU wait control
(6) serial data transfer control

3-2. Functions

① Ports PA0~7 and PB0~7 can be programmed of their data flow direction in bit unit. Also, it can be accessed as one location of the memory, as seen from the CPU.

② Latch clock PΦ can be directly given from the external source through output ports PC0~7. Also, it can be accessed as one location of the memory, as seen from the CPU.

③ As there are two interrupt request inputs of IRQ and PB7, interrupt request can be issued to the CPU at the rising edge of the input when the corresponding bit of the MSK register is "1". PB7 must be in the input mode before using PB7 for the interrupt input.

④ Since there is the CPU wait control circuit, two memory enable signals can be output to the memory of slow access time. Besides, it has two inputs of wait conditions. Up to 8 varieties of access time can be programmed.

⑤ It has the following functions to handle serial data transfer.

A. Serial data transmit

Serial data transfer takes place in a format of a start bit, 8 bits of data, and two stop bits. Transmission clock is selectable by means of internal and external clock select program, as well as the clock rate (1/1, 1/2, 1/128, 1/256, 1/512, 1/1024, 1/2048, and 1/4096 of the basic clock).

B. Serial data receive

When the start bit is received in an idle state, the serial data following to it is received. After receiving the data comprised of 8 bits, it is then stored in the internal register and the interrupt request flag is set active. Receive clock is furnished from the external source which becomes the receiving clock by itself. It must be in synchronization with the serial data input.

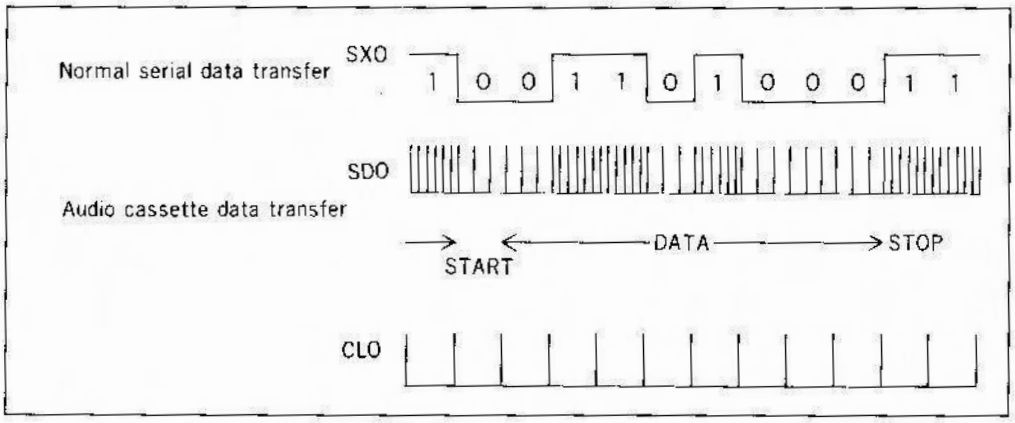
C. Pulse waveform

It is possible to have continuous output of pulse waveform. Frequency is programmable to eight kinds of 1/1, 1/2, 1/128, 1/256, 1/512, 1/1024, 1/2048, and 1/4096 of the basic clock.

promsoft.com

D. Data transfer to the audio cassette tape

Format of data transferred to the audio cassette tape consists of a start bit, 8 bits of data, and two stop bits, with the modulation signal generated from the SDO output.



Assume now that normal serial data output is to be SXO, the modulation clock to the data 1 to be FX, the modulation clock to the data 0 to be FY, and audio cassette tape data output to be SDO, then the following equation comprises:

$$SDO = SXO \cdot FX + \overline{SXO} \cdot FY$$

Whereas, FX and FY can be set independently to 1/64, 1/128, 1/256, 1/512 or 1/1024 of the basic clock by means of programming.

The serial transmit clock CLO can be programmed as discussed in Item A.

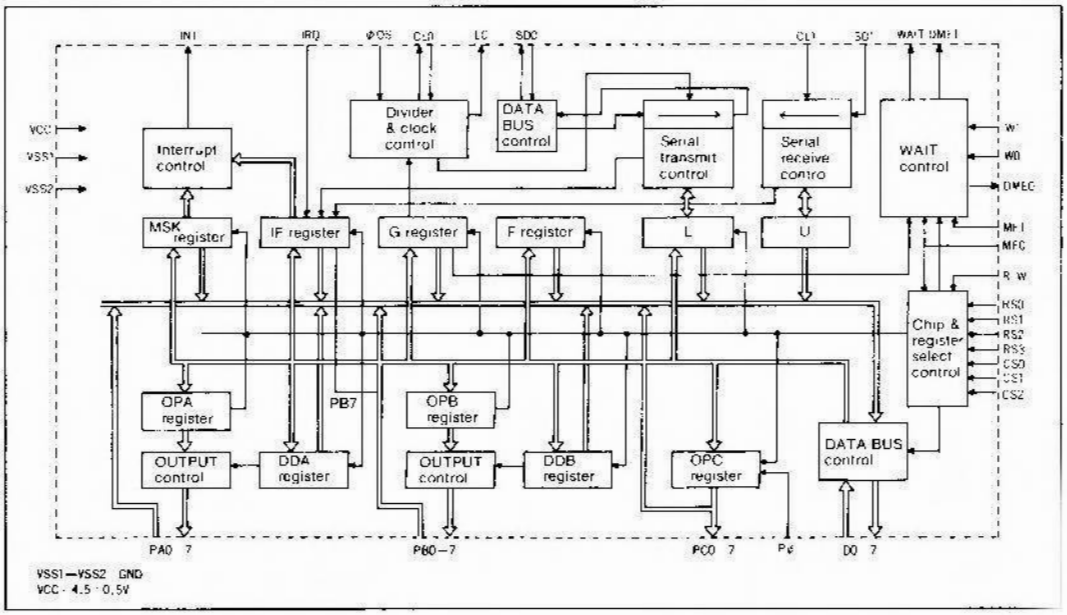
promsoft.com

3-3. Internal structure

3-3-1. Block diagram

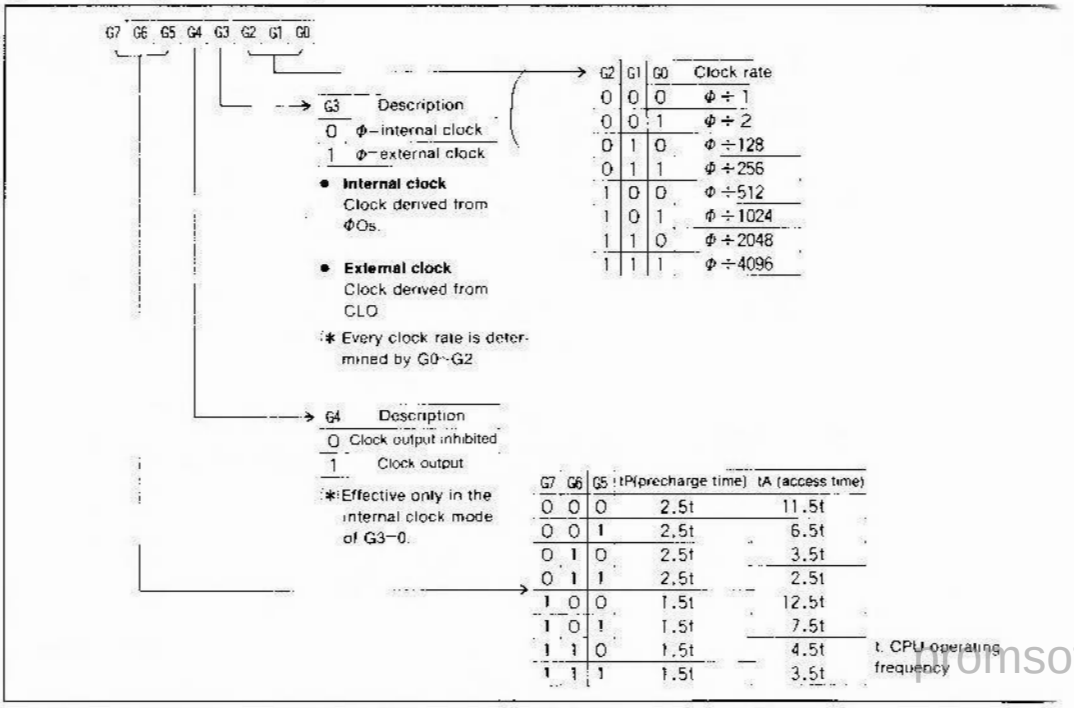
The I/O port controller consists of nine internal registers, wait controller, serial controller, and interrupt controller, and each of internal registers can be accessed as one location of the memory as seen from the CPU.

I/O port controller system block diagram



3-3-2. Internal registers

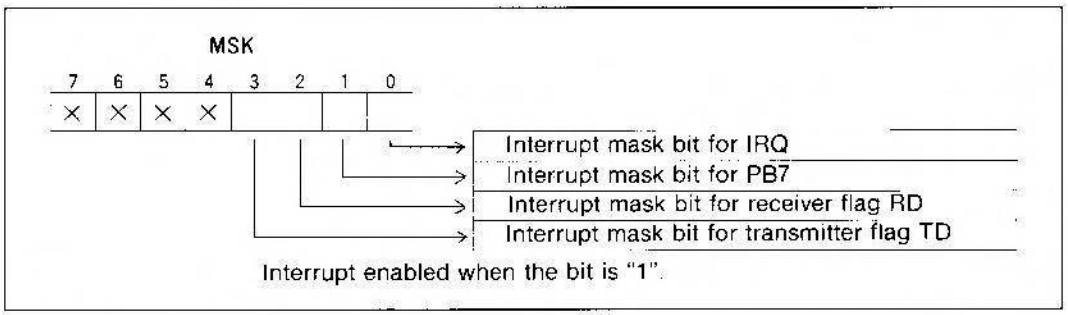
① **G register (RS3~0=1001)**



promsoft.com

The G register can read/write data when register is selected (1001).

② **MSK register**



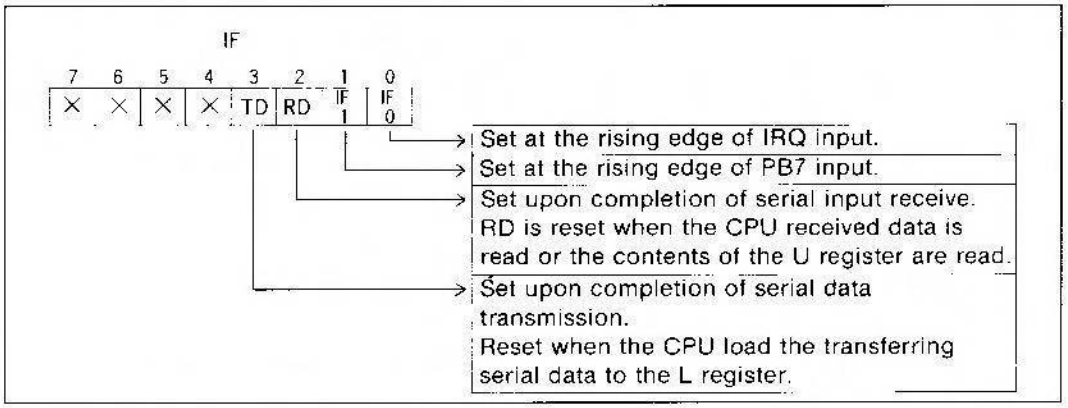
The MSK register can read/write data when register is selected (1010).

NOTE: When the contents of the MSK register are read, the contents of CL1, SD1, PB7, and IRQ are stored in high order digit positions.

**	CL1	SD1	PB7	IRQ	MSK 3	MSK 2	MSK 1	MSK 0

Contents of MSK read

③ **IF register**

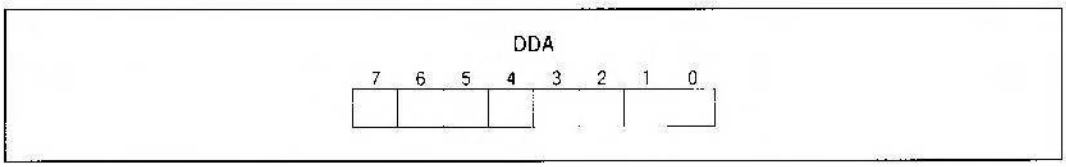


IF0 and IF1 can read/write data when register is selected (1011).

RD and TD are dedicated to read only.

NOTE: During receive of serial data, RD is reset. Term "serial data receive" means the period during which an 8-bit data is in reception, with the start bit excluded.

④ **DDA register**



The register used to determine the direction of the port PA.

***i*-th bit of the DDA register**

When 0	PA*i* is in the input mode.
When 1	PA*i* is in the output mode and outputs the contents of OPA*i*.

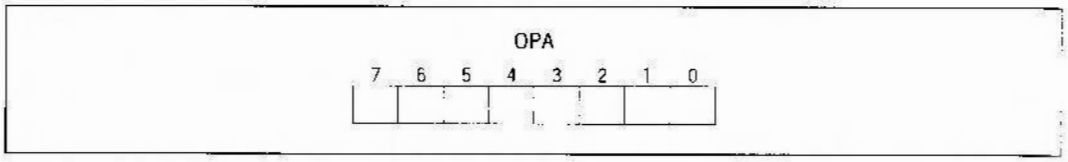
DDA can read/write when the register is selected (1100).

promsoft.com

⑤ DDB register

The register used to determine the direction of the port PB.
Selection of the input/output port is the same as in the DDA register.
DDB can read/write data when the register is selected (1101).

⑥ OPA register



The OPA register is the buffer for input and output of data to the port PA.
In the case of output, DDA*i* must be set to "1" (output mode) and the register must be selected (1110), then the data on the bus line is loaded intoOPA*i*, to be output on PA*i*.
In the case of input, DDA*i* must be set to "0" (input mode) which prohibits output from OPA*i*, then the contents of PA*i* are loaded into OPA*i* and the data is sent on the bus line.

⑦ OPB register

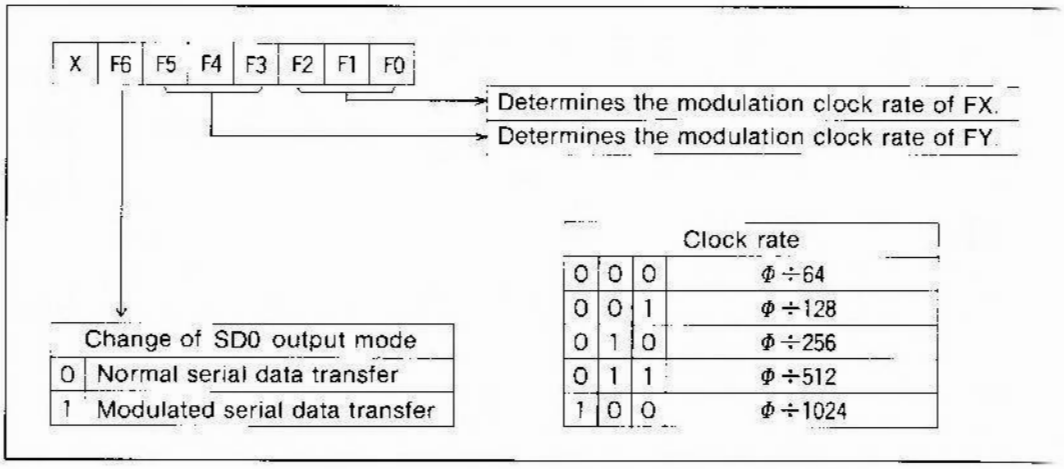
The OPB register is the buffer for input and output of data to the port PB.
It has the same function as the OPA register.
OPB can read/write data when the register is selected (1111).

⑧ OPC register

The OPC register can read/write data when the register is selected (1000).
Also, the contents of the data bus can be latched to the OPC register at the falling edge of the external input clock PΦ.

⑨ F register

The F register can read/write data when the register is selected (0111).
F0-2 determines the clock rate of the modulation clock FX and F3-5 the modulation clock of FY.



Change of SD0 output mode	
0	Normal serial data transfer
1	Modulated serial data transfer

			Clock rate
0	0	0	Φ÷64
0	0	1	Φ÷128
0	1	0	Φ÷256
0	1	1	Φ÷512
1	0	0	Φ÷1024

promsoft.com

3-3-3. Pin description

(*): See Pad layout and structure in 3-5-2.

No.	Signal name	Function	No.	Signal name	Function
1	PA1	Port input/output	31	RS2	Register select input
2	PA2	Port input/output	32	RS3	Register select input
3	PA3	Port input/output	33	R/W	Read/write input
4	PA4	Port input/output	34	ME0	Memory enable input
5	PA5	Port input/output	35	ME1	Memory enable input
6	PA6	Port input/output	36	W0	Wait condition input
7	PA7	Port input/output	37	W1	Wait condition input
8	GND	Power source	38	GND	Power source
9	PB0	Port input/output	39	VCC	Power source
10	PB1	Port input/output	40	DME0	Memory enable output
11	PB2	Port input/output	41	DME1	Memory enable output
12	PB3	Port input/output	42	WAIT	Wait output
13	PB4	Port input/output	43	INT	Interrupt output
14	PB5	Port input/output	44	RESET	Initialize output
15	PB6	Port input/output	45	IRQ	Interrupt input
16	PB7	Port input/output. Interrupt input	46	ΦOS	Basic clock input
17	PΦ	Port PC latch clock	47	CL1	Serial receive clock input
18	PC0	Port output	48	SD1	Serial receive input
19	PC1	Port output	49	LC	Not used
20	PC2	Port output	50	CL0	Serial rec/trn clock input/output
21	PC3	Port output	51	SD0	Serial rec/trn input/output
22	PC4	Port output	52	D0	Data bus input/output
23	PC5	Port output	53	D1	Data bus input/output
24	PC6	Port output	54	D2	Data bus input/output
25	PC7	Port output	55	D3	Data bus input/output
26	CS0	Chip select input	56	D4	Data bus input/output
27	CS1	Chip select input	57	D5	Data bus input/output
28	$\overline{CS2}$	Chip select input	58	D6	Data bus input/output
29	RS0	Register select input	59	D7	Data bus input/output
30	RS1	Register select input	60	PA0	Port input/output

promsoft.com

3-4. Functions

3-4-1. Operation

I/O port controller operation

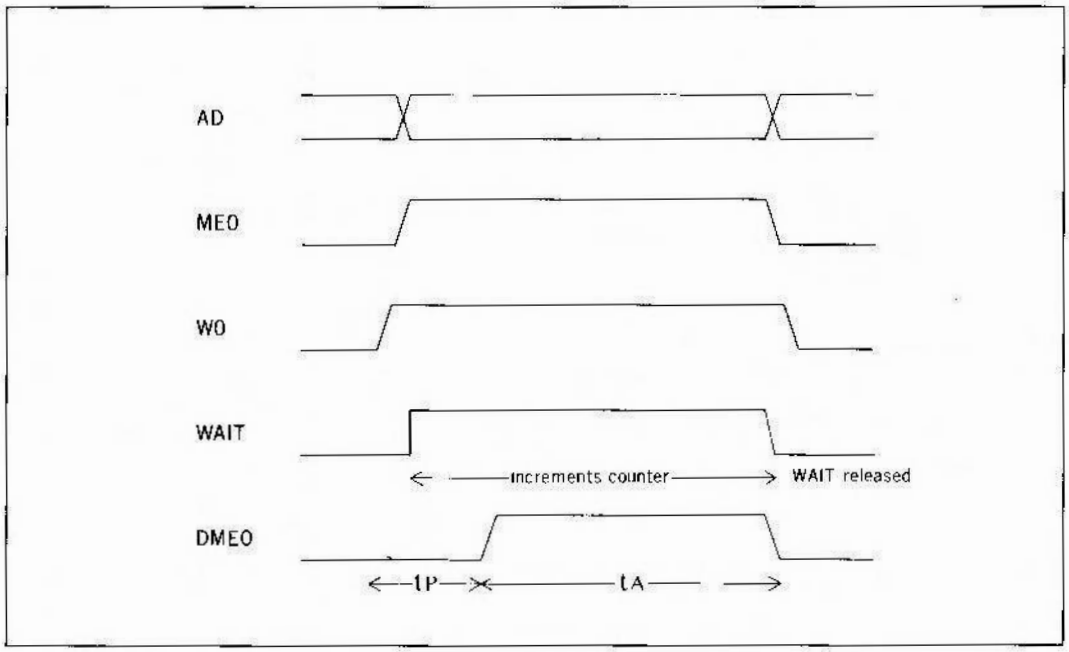
$\overline{CS2}$	CS1	CS0	RS3	RS2	RS1	RS0	R/W	Operation
0	1	1	0	1	0	0	0	Resets the divider internally contained in the I/O port chip.
			0	1	0	1	1	Reads the contents of the U register and sends them on the data bus. At the same time, the receive flag RD is reset.
			0	1	1	0	0	The contents of the data bus are converted into a serial data signal of start/data/stop structure. At the same time, the transmit flag TD is reset.
			0	1	1	1	0	The contents of the data bus are stored in the F register.
			1	0	0	0	0	The contents of the data bus are stored in the OPC register.
							1	The contents of the OPC register are sent on the data bus.
			1	0	0	1	0	The contents of the data bus are stored in the G register.
							1	The contents of the G register are sent on the data bus.
			1	0	1	0	0	The contents of the data bus are stored in the MSK register.
							1	The contents of the MSK register are sent on the data bus.
			1	0	1	1	0	The contents of the data bus are stored in the IF register.
							1	The contents of the IF register are sent on the data bus.
			1	1	0	0	0	The contents of the data bus are stored in the DDA register.
							1	The contents of the DDA register are sent on the data bus.
			1	1	0	1	0	The contents of the data bus are stored in the DDB register.
							1	The contents of the DDB register are sent on the data bus.
			1	1	1	0	0	The contents of the data bus are stored in the OPA register.
							1	The contents of the PA*i* are sent on the data bus.
			1	1	1	1	0	The contents of the data bus are stored in the OPB register.
							1	The contents of the PB*i* are sent on the data bus.

promsoft.com

3-4-2. Wait control

① Function

Wait is a function applied in accessing a slow action memory. It makes the CPU operation temporarily halted until a complete access is done to the slow action memory. When the CPU makes access to the address where slow action memory is assigned, the condition "1" is entered to the wait condition inputs W0 and W1. W0 is the wait condition input which is applied to place wait to the memory area controlled by ME0, while W1 is the wait condition for the memory area controlled by ME1. When the wait condition is established, that is, when "W0·ME0+W1·ME1=1" is met, the WAIT signal is issued to the CPU. As the wait control counter is provided internally, it releases WAIT to the CPU when it is counted to the value set by the program, then the CPU proceeds to execute a next machine cycle.



Since a slow access memory usually consists of dynamic logic, it needs a precharge time (t_P). Thus, the I/O port controller issues the memory enable signals DME0 and DME1 to the dynamic memory, including the precharge time. DME0 is for ME0 and DME1 is for ME1.

When the CPU accesses the I/O port controller, the signal WAIT is issued without the wait condition in W0 and W1.

② Wait time

Wait time can be programmed by means of bit positions of G5, 6 and 7 of the G register. Relation of G5, 6,7 with t_A and t_P refer to page 70.

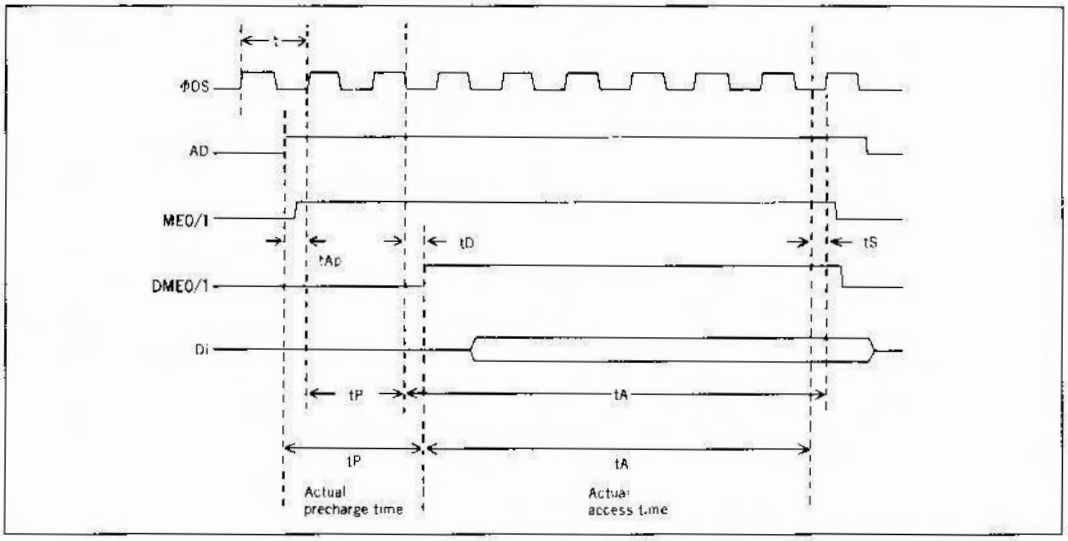
promsoft.com

③ Wait time of I/O port controller itself

When the CPU accesses the I/O port controller, wait of a single CPU machine cycle is automatically applied.

If the W0 and W1 is in the wait state when the CPU accessed the I/O port controller, wait ends in one cycle.

NOTE: When there is no wait condition in W0 and W1, the same waveform as ME0 and ME1 are sent on DME0 and DME1.



t_P' and t_A' for actual ROM is as follows:

$$t_P' = t_{AP} + t_P + t_D$$

$$t_A' = t_A - t_D - t_S$$

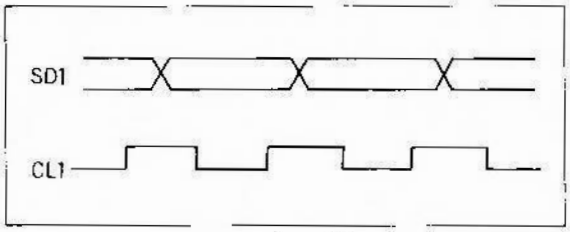
$$t_S = 200nS$$

Since t_{AP} and t_D differ depending on the peripheral circuit configuration, they should be computed on the basis of load capacitance.

3-4-3. Serial data input

SDI is a serial data transfer input and CLI serial data transfer clock.

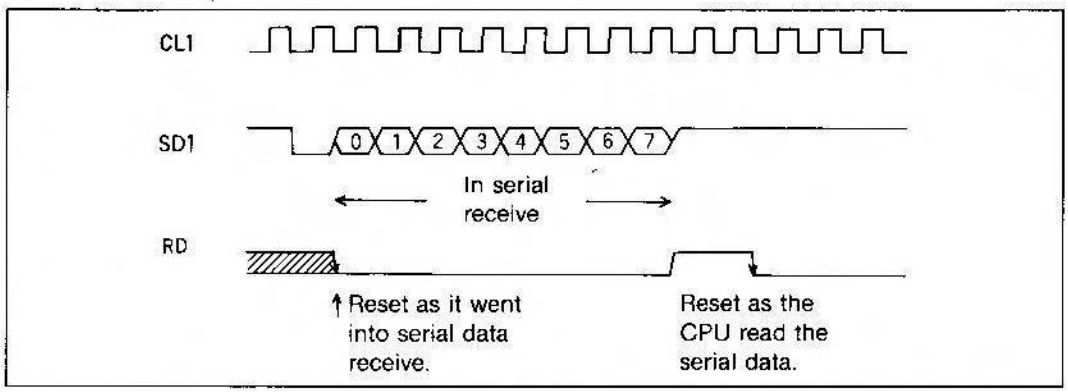
The I/O port controller reads the input data at the rising edge of CLI. Serial data goes into the receiving mode when it changes from an idle state (SDI=1) to low state and reads data from a next clock.



When the 8-bit data is received, it sets the receive and flag RD active. If the mask bit is on at this point, interrupt request is issued to the CPU. RD will be reset upon reading the 8-bit data.

RD will be reset in a course of serial data receiving, which is the period that the 8-bit data is being received, without including the start bit.

promsoft.com



3-4-4. Reset

When the RESET line is kept in "1" at least for three ΦOS clock cycles, it causes internal reset, by which all internal registers are cleared to "0" and ports PA and PB go into the input mode. The divider, however, will not be reset.
Terminal states after the reset are as shown in Table below.

Pin name	In/Out	State after reset
D0~D7	In/Out	High impedance
PA0~PA7	"	"
PB0~PB7	"	"
PC0~PC7	Out	Low level
WAIT	"	(NOTE)
DME0	"	"
DME1	"	"
INT	"	Low level
SD0	In/Out	High level
CL0	"	Low level

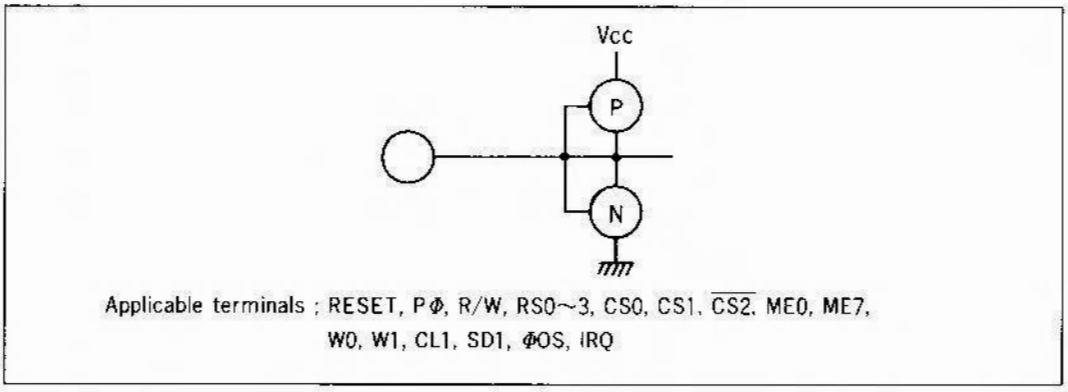
NOTE: WAIT, DME0, and DME1 are output dependent of W0, ME0, W1, and ME1 and do not have any connection with reset.

promsoft.com

3-5. Specification

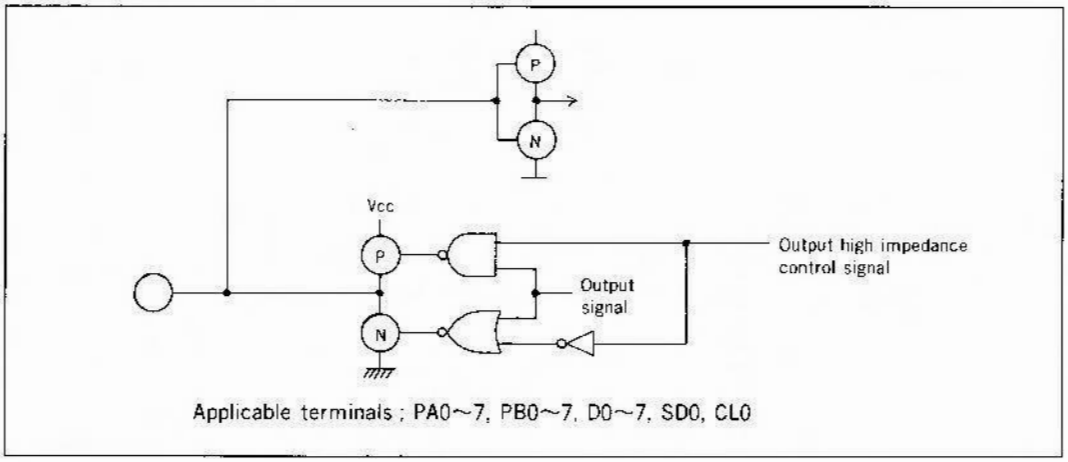
3-5-1. I/O port controller input/output circuits

① Input



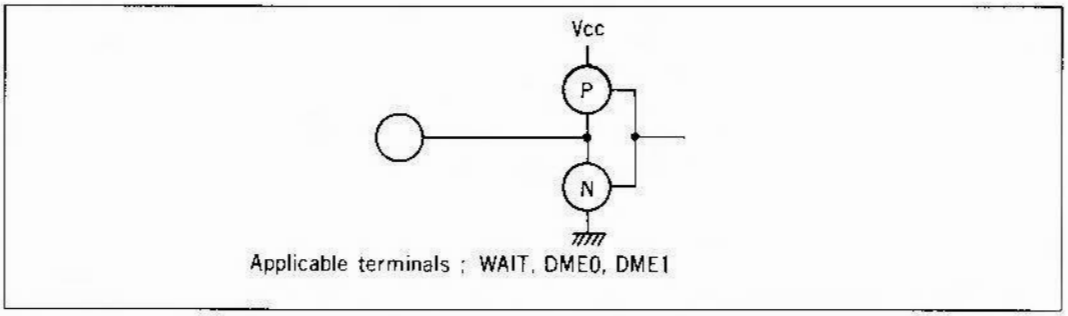
Applicable terminals ; RESET, PΦ, R/W, RS0~3, CS0, CS1, $\overline{CS2}$, ME0, ME7, W0, W1, CL1, SD1, ΦOS, IRQ

② Input/output



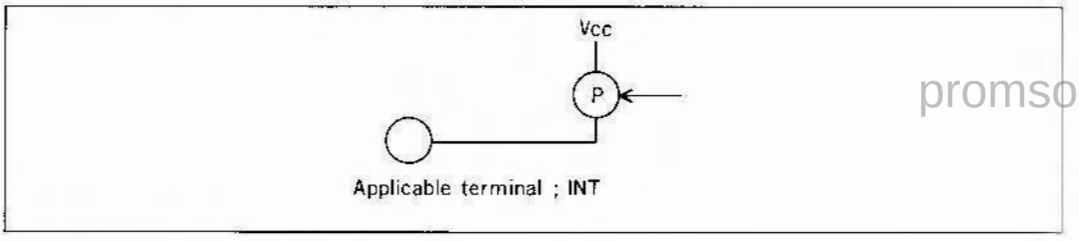
Applicable terminals ; PA0~7, PB0~7, D0~7, SD0, CL0

③ Output



Applicable terminals ; WAIT, DME0, DME1

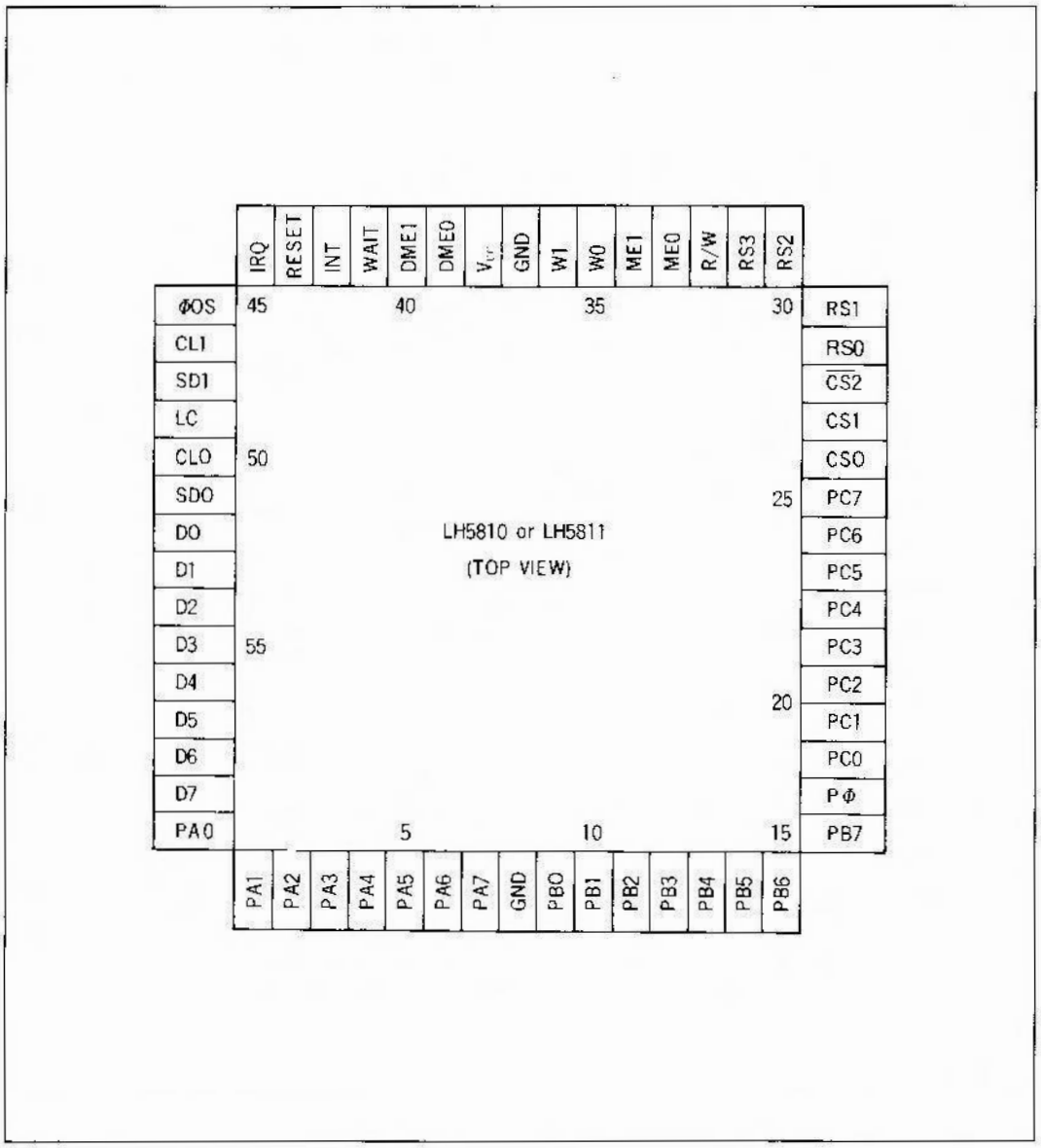
④ Output



Applicable terminal ; INT

promsoft.com

3-5-2. Pad layout and structure



3-5-3. Electrical characteristics

① Absolute maximum ratings

Parameter	Symbol	Limits	Unit
Input apply voltage (*1)	V_{IN1}	−0.3 to 6.5	V
Input apply voltage (*2)	V_{IN2}	−0.3 to Vcc +0.3	V
Operating temperature	Topr	−5 to +55	°C
Storage temperature	Tstg	−55 to +150	°C

*1: Applicable to Vcc and with respect to GND.
*2: Applicable to other than Vcc, GND, and with respect to GND.

② Operating condition

Parameter	Symbol	Limits	Unit
Supply voltage	Vcc	4.0 to 5.0	V

promsoft.com

③ Electrical characteristics

③ - 1 - DC characteristics

Ta=−5~55℃
Vcc=4.0~5.0V

Parameter	Symbol	Limits			Unit	Test condition	Note
		Min	Typ	Max			
Input voltage	V_{IL}	0		0.8	V		1
	V_{IH}	2.4		Vcc	V		
Output voltage 1	V_{OL1}			0.4	V	I_{OL}=1.6mA	2
	V_{OH1}	2.4			V	I_{OH}=0.4mA	
Output voltage 2	V_{OH2}	1.5			V	I_{OH}=1.5mA	3
Output leakage current	$\|I_{LO}\|$			1.0	μA		4
Input current	$\|I_{LI}\|$			1.0	μA	V_{IN}=0V or Vcc	5
Normal power dissipation	Iccа		0.4	0.9	mA	ΦOS=2MHz	6
	Iccb			5	μA	ΦOS=0V	7

NOTES: For Note number, refer to the list below.

Note No.	Applicable terminals
1	PA0~7, PB0~7, D0~7, R/W, RS0~3, CS0, CS1, $\overline{CS2}$, ME0, ME1, W0, W1, SD1, CL1, ΦOS, SD0, CL0, IRQ, RESET.
2	PA0~7, PB0~7, PC0~7, D0~7, DME0, DME1, WAIT, CL0, INT
3	PA0~7, CL0, SD0.
4	All input terminals.
5	All output terminals.
6	The term "normal" applies to the reset state that the I/O port controller is not selected and the 2MHz clock should be supplied through the ΦOS pin. The following input terminals should be connected to 0V except $\overline{CS2}$. PA0~7, PB0~7, D0~7, CS0, CS1, $\overline{CS2}$=Vcc, R/W, RS0~3, PΦ, ME0, ME1, W0, W1, CL1, SD1, IRQ.
7	Applied to the state that no clock is supplied to the ΦOS terminal in the normal condition.

promsoft.com

③ - 2 - AC characteristics

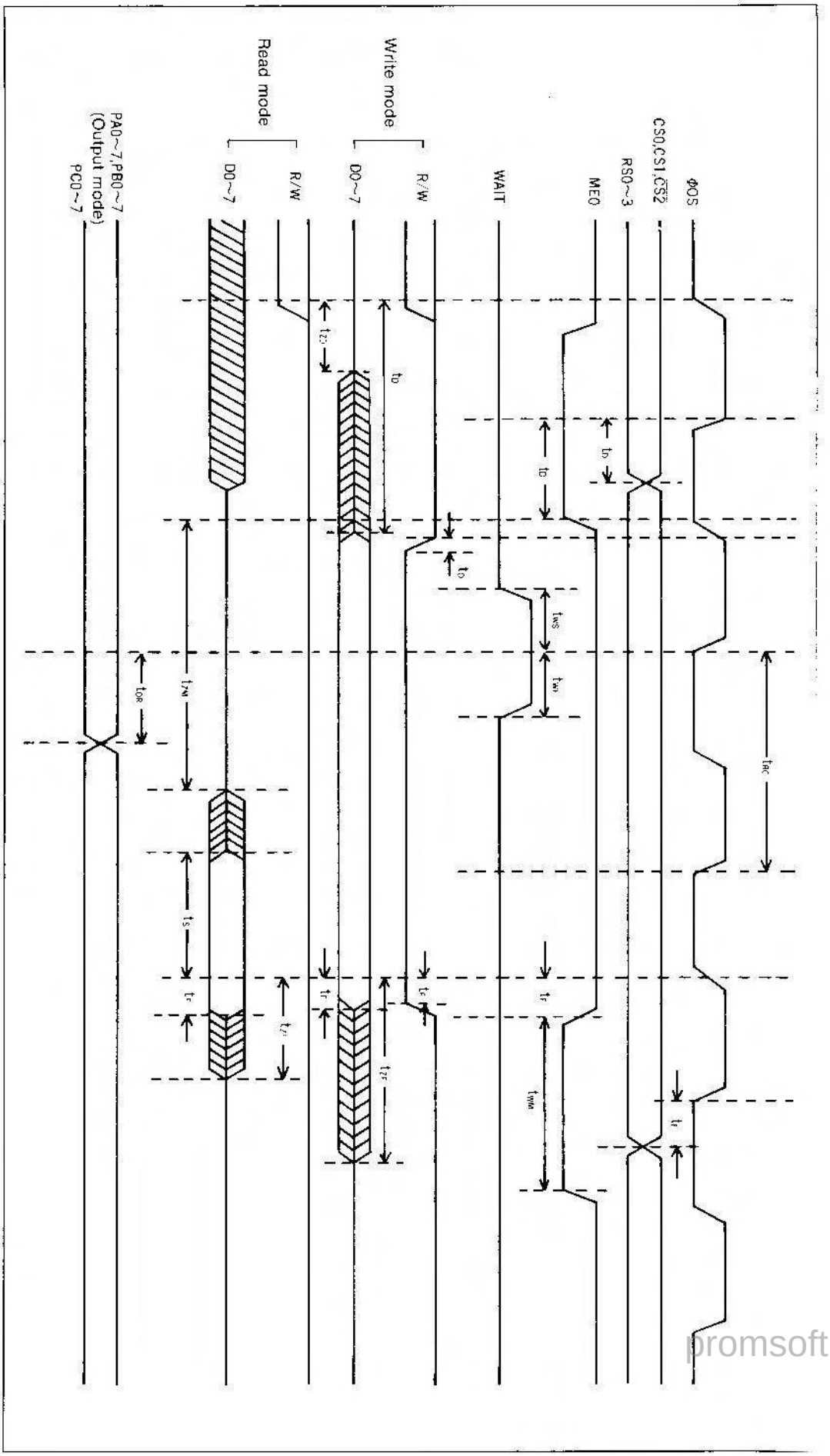
(Ta=−5 to 55°C, V_{CC}=4.0 to 5.0V)

Parameter	Symbol	Limits			Unit	Test condition
		Min	Typ	Max		
CS0, CS1, $\overline{CS2}$, RS0~3	t_D			230	ns	
	t_F	80			ns	
ME0, ME1	t_D			240	ns	
	t_F	80		175	ns	
	t_{WM}	250			ns	
WAIT	t_{WS}	150			ns	C_L=50pF
	t_{WF}	50		200	ns	
D0~7 (input mode)	t_D			500	ns	
	t_F	130			ns	
	t_{ZO}	100			ns	V_{OH}=0.8Vcc
	t_{ZF}			300	ns	
D0~7 (output mode)	t_S	200			ns	
	t_F	50			ns	
	t_{ZM}	150			ns	
	t_{ZF}			100	ns	
R/W	t_D	−30		50	ns	V_{OH}=0.8Vcc
	t_F			150	ns	
ΦOS	t_{RC}	500			ns	
	t_r			150	ns	0.1→0.9Vcc
	t_F			40	ns	0.9→0.1Vcc
PA0~7, PB0~7 (output mode)	t_{DR}			1	μs	C_L=100pF
PC0~7 (register select)	t_{DR}			1	μs	C_L=100pF
PΦ	t_D			350	ns	V_{OH}=0.8Vcc
	t_F			100	ns	
PC0~7 (PΦ latch)	t_{DP}			1	μs	C_L=100pF
SD0 (input mode) SD1	t_S	100			ns	
	t_H	50			ns	
SD0 (output mode)	t_D			100	ns	C_L=100pF
DME0	t_D			120	ns	C_L=50pF
DME1	t_F			120	ns	
SD0	t_Z			200	ns	C_L=100pF
CL0	t_r			100	ns	0.1→0.9Vcc C_L=50pF
	t_f			100	ns	0.9→0.1Vcc C_L=50pF

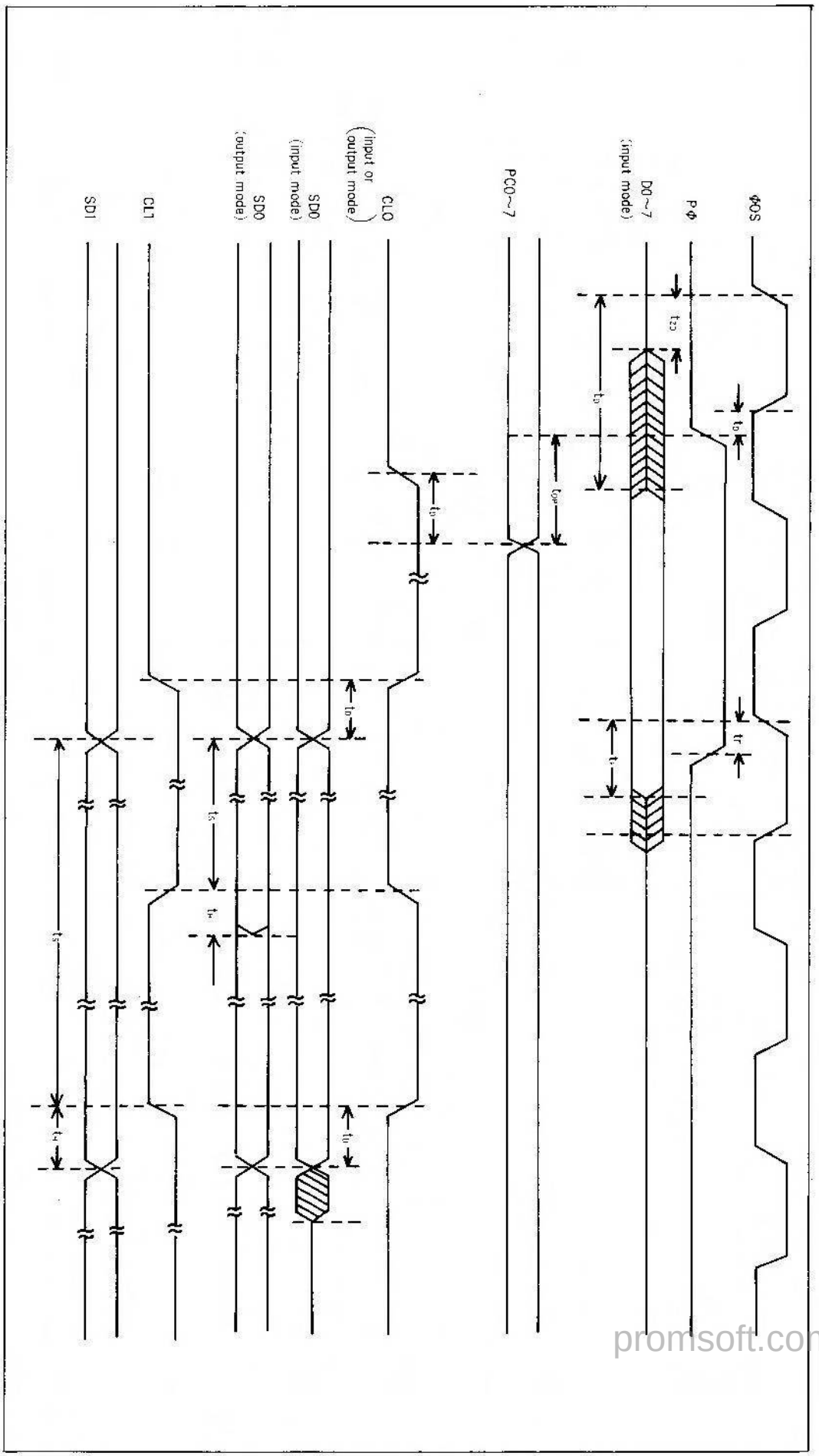
(*): Unless otherwise specified, the criterion shall be V_{OH}=2.0V, V_{OL}=0.4V

promsoft.com

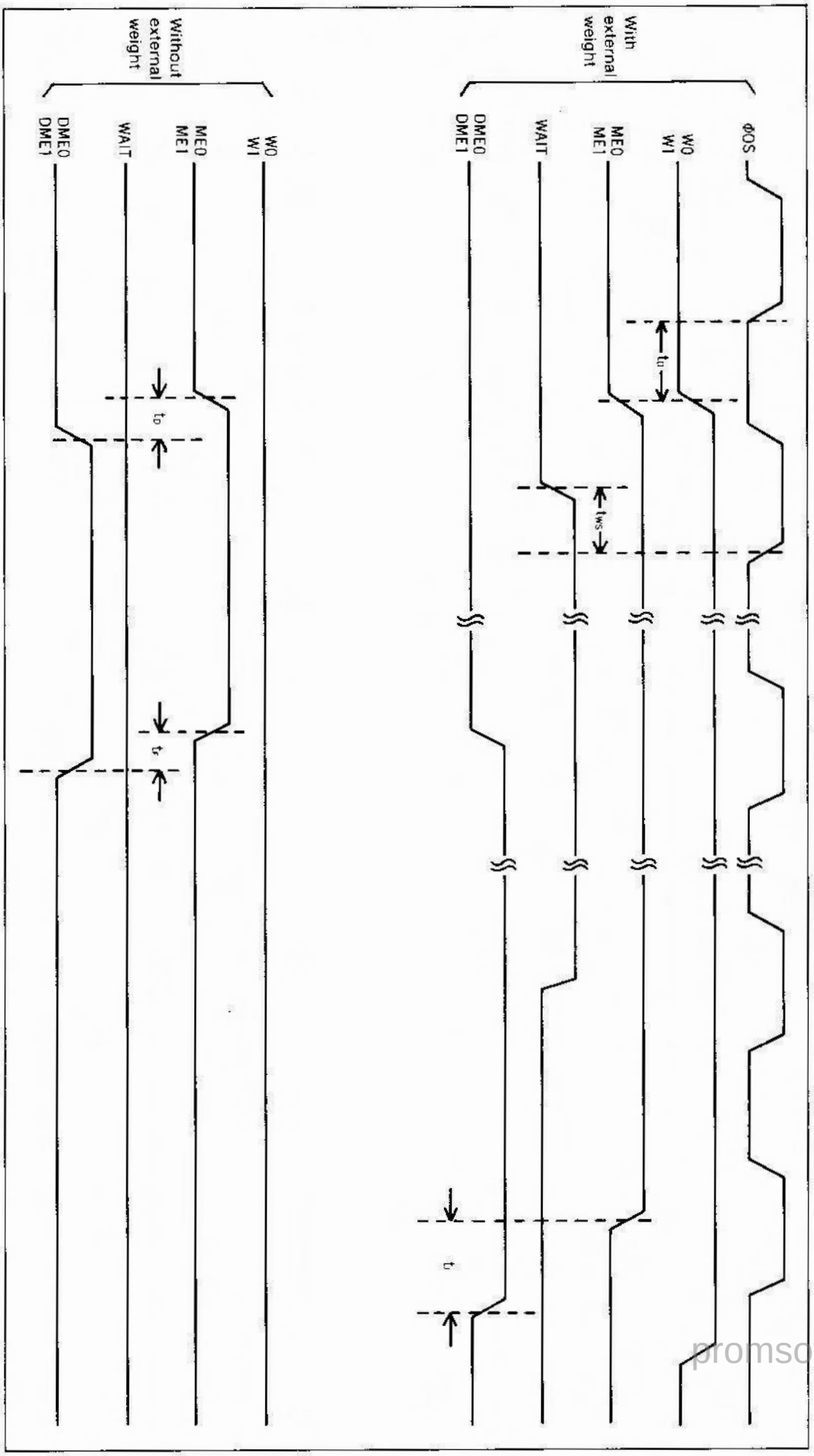
③ - 3 - Timings



promsoft.com



promsoft.com



promsoft.com

4

PC-1500 hardware description

promsoft.com

4-1. PC-1500 system configuration

4-1-1. Outline

Around the PC-1500 Pocket Computer, the PC-1500 system can be configured with a variety of peripherals, which include the CE-150 Color Graphic Printer, CE-151 4KB Memory Module, CE-152 Cassette Tape Recorder, CE-153 Software Board, CE-154 System Carrying Case, CE-155 8KB Memory Module, CE-158 Serial and Parallel Interface, and CE-159 Program Module.

① **PC-1500 Pocket Computer**

It is a pocket computer that features high speed processing with the CMOS 8-bit microprocessor in use, and it has memories of 16KB ROM and 3.5KB RAM on standard. It also permits expansion by the use of options.

Four pieces of Type AA dry batteries are used for its power source and the 7×156 dot LCD is used for the display.

② **CE-150 Color Graphic Printer**

It is a ball-point pen type four color printing X-Y plotter. It has 8KB ROM for the memory and is driven by five pieces of Type AA Ni-Cd batteries. It also has the cassette interface built in the unit.

③ **CE-151 4KB Memory Module**

It is a 4KB option memory unit in which two chips of 2KB RAM are contained.

④ **CE-152 Cassette Tape Recorder**

It is a cassette tape recorder dedicated for use with the PC-1500 system. It is driven by four pieces of Type AA dry batteries.

⑤ **CE-153 Software Board**

It has 140 keys arranged on the keyboard and each key assignment is defined by means of programming. As it does not have power supply by itself, it is supplied from the PC-1500.

⑥ **CE-154 System Carrying Case**

The carrying case exclusively designed for carrying of the PC-1500, CE-150, CE-152, CE-153.

⑦ **CE-155 8KB Memory Module**

It is an 8KB option memory unit in which four chips of 2KB RAM are contained.

⑧ **CE-158 Serial and Parallel Interface**

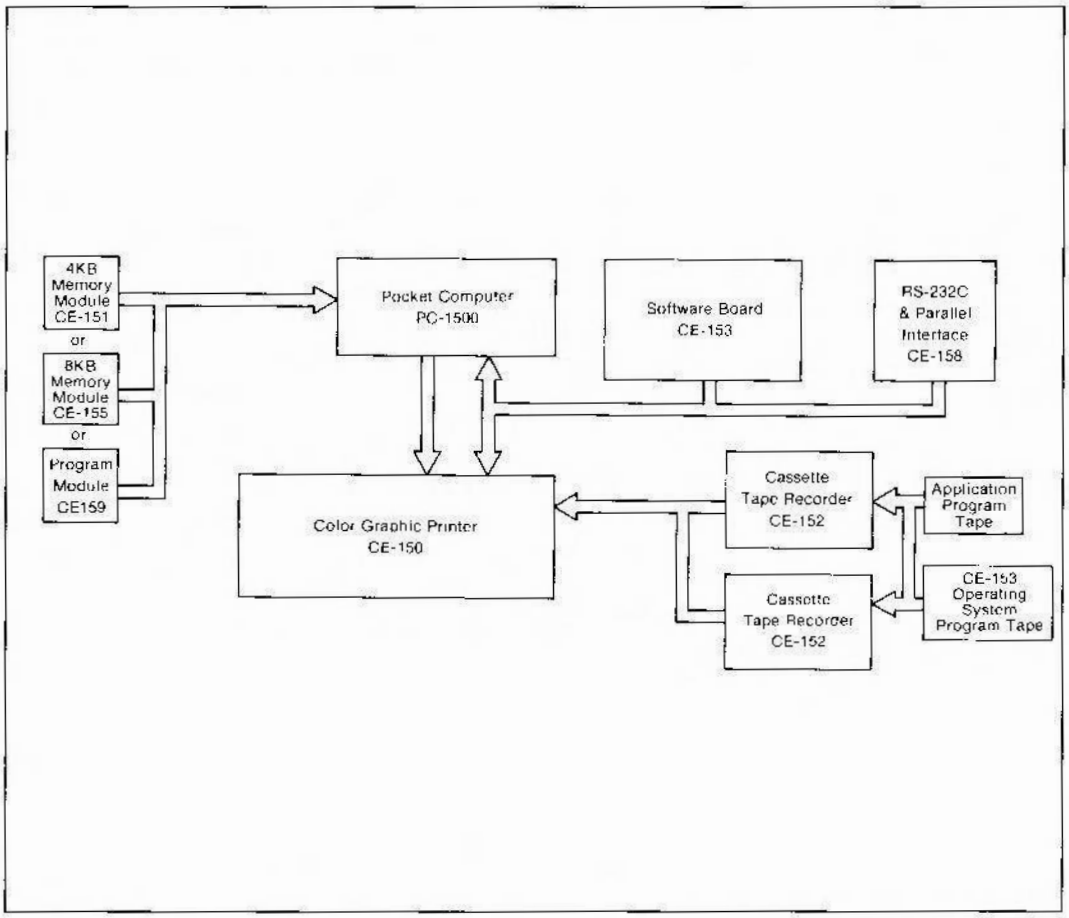
It has the RS-232C serial interface and the Centronics parallel interface with the built-in 16KB ROM. It is driven by four pieces of Type AA Ni-Cd batteries. It will be possible to interface with a normal type printer, computer, instrument, etc.

⑨ **CE-159 Program Module**

It is a detachable memory module that consists of four chips of 2KB RAM, of which area can be partially or entirely used for the read only area. The contents of the memory can be retained by means of the internal lithium battery even if it was detached from the PC-1500.

promsoft.com

4-1-2. Block diagram



4-1-3. Power supplies (PC-1500, CE-150, CE-158, CE-159)

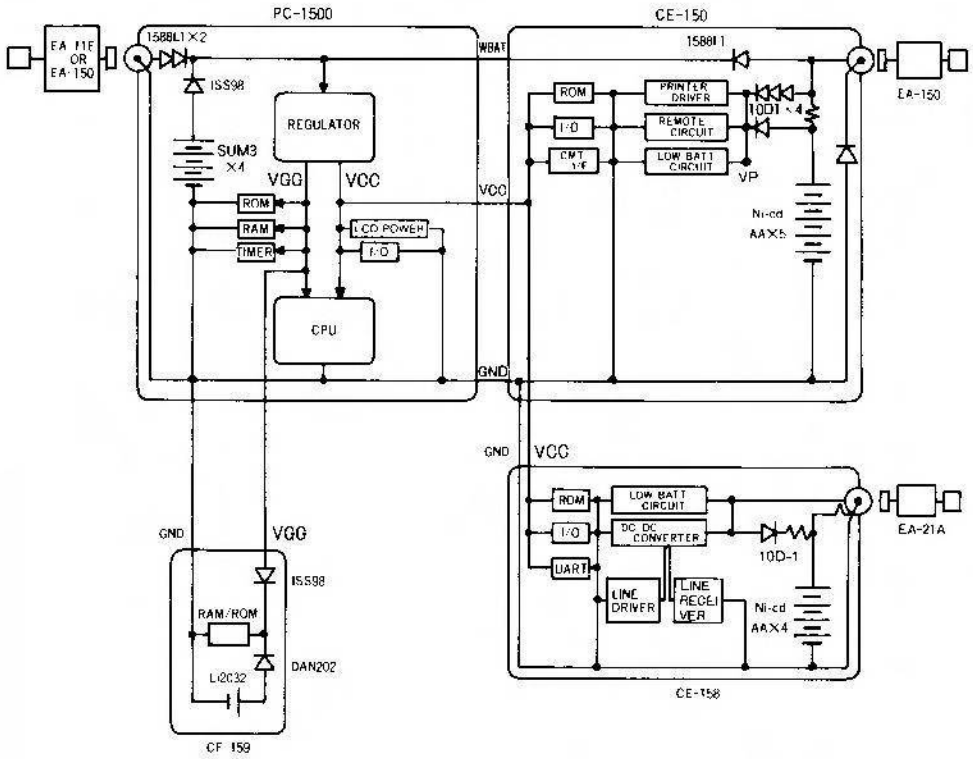
① PC-1500 and its peripheral units (CE-150, CE-158, CE-159) are driven independently by dry battery, Ni-Cd battery, or lithium battery.
When the PC-1500 is connected with peripheral, RAM, ROM, I/O PC, CMT I/F, and UART are driven by VCC or VGG supplied from the PC-1500.

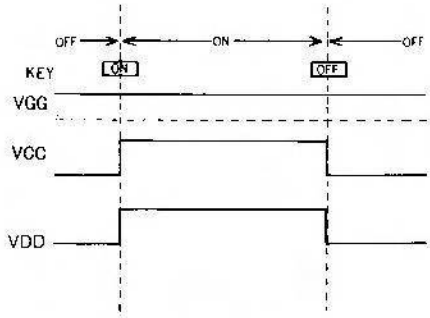
② When the Ni-Cd battery of the CE-150 is charged capable to drive the PC-1500, power is supplied from the CE-150 to the PC-1500 and therefore the battery of the PC-1500 does not consume. In addition, it drives the driver of the printer and remote control relays.

③ The Ni-Cd battery of the CE-158 drives the driver and receiver of the RS-232C serial interface and the Centronics parallel interface.

④ When the CE-159 is in connection with the PC-1500, it is operated by VGG of the PC-1500, and its memory contents can be retained by the internal lithium battery when disconnected from the PC-1500.

promsoft.com





NOTE: VGG is always output.

promsoft.com

4-2. PC-1500

4-2-1. Outline

① CPU

The LH5801 CMOS 8-bit Microprocessor is used. As it is operated by the clock frequency of 2.6MHz from the crystal oscillator, its internal machine cycle is 1.3MHz.

BFI ..... Depression of the ON key turns it to high level so that VCC is supplied from BFO.

IN0~IN7 ..... Input port for other than the ON key.

D0~D7 ..... Bidirectional data bus which is used to write and read data to/from the external memory.

D0: LSB

D7: MSB

AD0~AD15 .... Address bus.

AD0: LSB

AD15: MSB

XL0, XL1 ..... External crystal connection terminals, through which the 2.6MHz crystal is connected.

XL0: Input

XL1: Output

② I/O PC

Either LH5810 or LH5811 is used. The same chip is used for I/O of CE-150, CE-153, and CE-158.

PA0~PA7 ..... Key strobe output

PB7 ..... ON key input

AD0~3, AD12~13, DME1

..... Address of I/O port is set within F000H to F00FH of the ME1 area.

③ Timer IC

μPD1990AC is used and connected with the 32.768kHz crystal.

④ Chip select decoder

It consists of two chips of TC40H139F and TC40H138F and it is used to select chip by means of S0~4, S6, S7, $\overline{2Y2}$ and $\overline{2Y3}$. For more details, refer to "Chip select circuit".

⑤ Display chip

Because 4-bit SC882G is used, chip 1 is used in pair with chip 3 and chip 2 with chip 4. Select signal is commonly used by chip 1 and chip 3 or chip 2 and chip 4. Data bus is therefore divided into D0~D3 and D4~D7 in order to handle compatible with the 8-bit RAM.

Address is within 7600H to 77FFH of the ME0 area and is used for the fixed variable area, in addition to the display buffer.

⑥ System ROM

It is the PC-1500 system program residing ROM for which the 8-bit×16K SC61328F is used. Address is within C000H to FFFFH of the ME0 area.

⑦ System RAM

Because the 4-bit×1K bytes TC5514 is used in a pair, the data bus is divided into two of D0~D3 and D4~D7 and the select signal S7 is commonly shared so as to be compatible with the 8-bit RAM.

Address is within 7800H to 7BFFH of the ME0 area which is used for the system memory area and for the fixed variable area.

promsoft.com

⑧ User RAM

It is the user RAM for which 8-bit×2KB HM6116 is used. Address selected by S0 is within 4000H to 47FFH.

⑨ 40-pin connector

It is the connector used for the connection of one of optional memory modules and program module, on which provided signal terminals for address bus, data bus, and chip select.

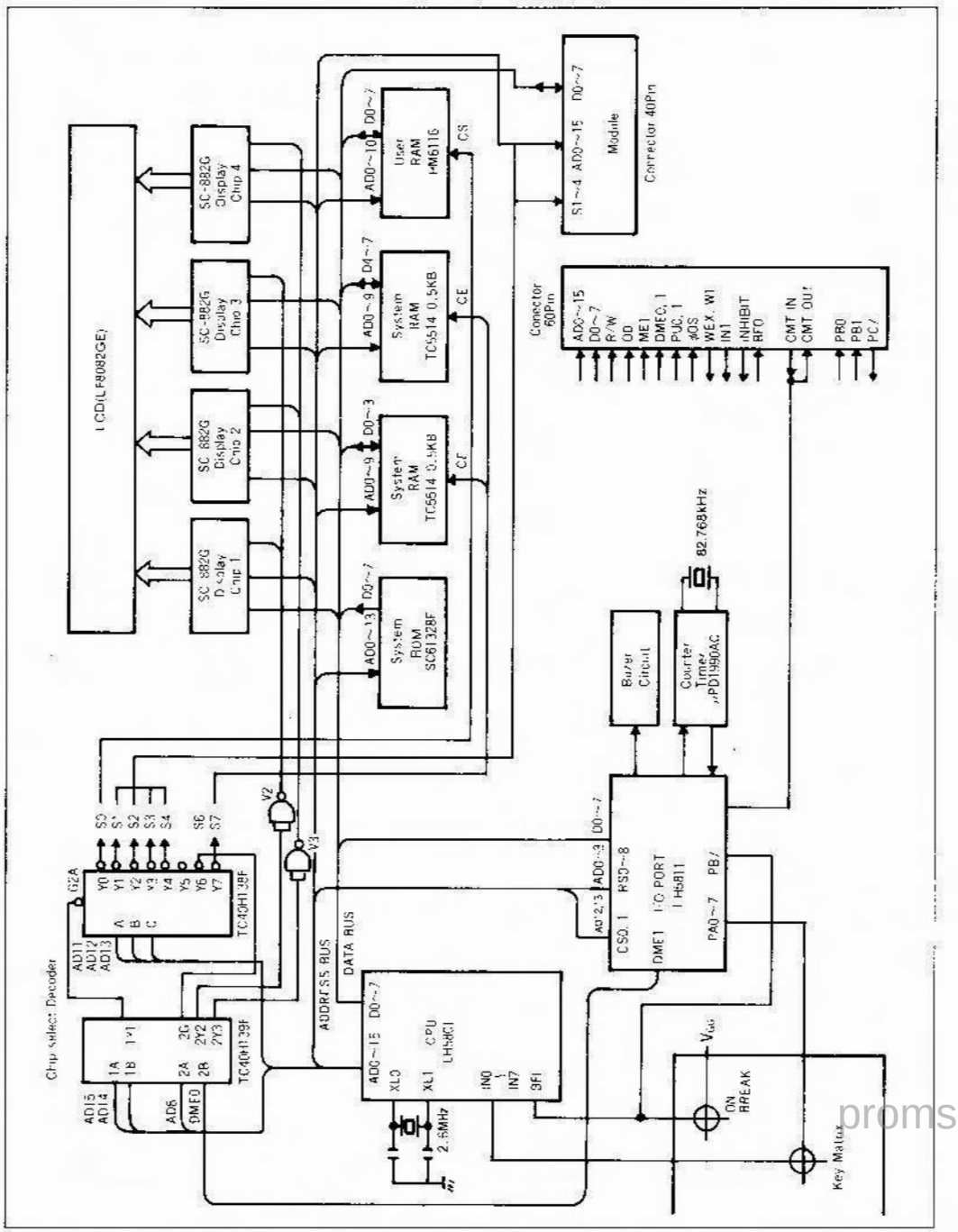
⑩ 60-pin connector

It is the connector used for the connection of optional printer and interface, on which provided signal terminals of address bus and data bus for the control of peripherals.

⑪ LCD

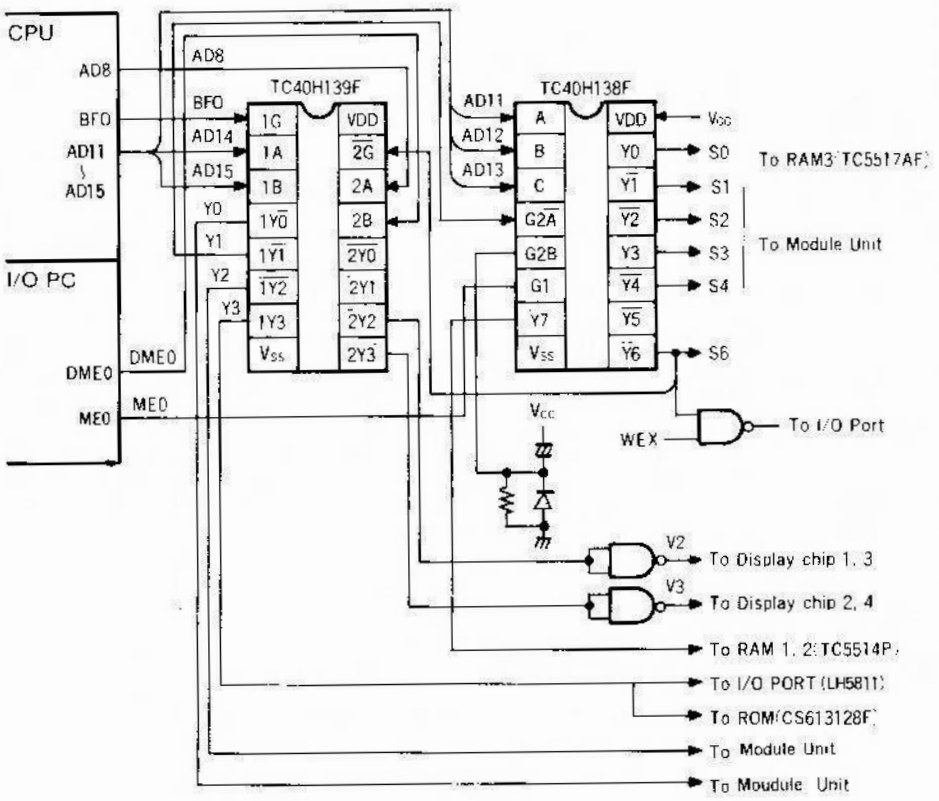
The 7×156 dot LF8082GE multi-display is used.

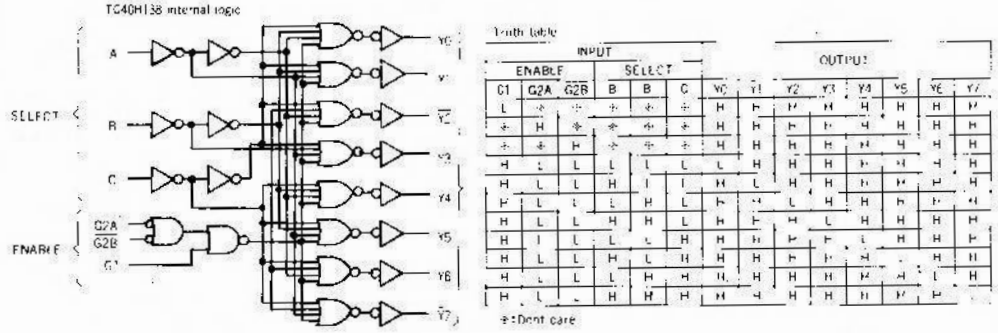
4-2-2. Block diagram

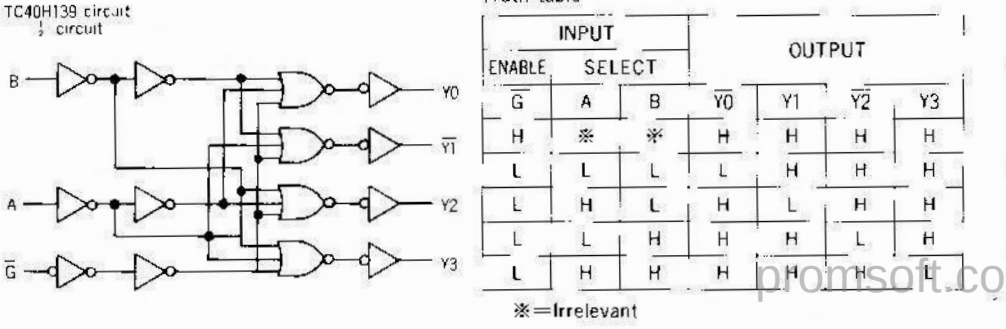


promsoft.com

4-2-3. Chip select circuit







promsoft.com

- **$\overline{1Y0}$~$\overline{1Y3}$ is selected by the decoder IC (TC40H139F) when the gate signal input BFO ($\overline{1G}$) is low.**

  Y0 ($\overline{1Y0}$) ..... With low state of AD14 and AD15, the Y0 output goes low so as to select the user memory area of the module unit. (Address assignment of 0000H~3FFFH)

  Y1 ($\overline{1Y1}$) ..... With high state of AD14 and low state of AD15, the Y1 output goes low so as to select the gate ($\overline{G2A}$) of the IC (TC40H138F). (Address assignment of 4000H~7FFFH)

  Y2 ($\overline{1Y2}$) ..... With low state of AD14 and high state of AD15, the Y2 output goes low so as to select the optional ROM area of the module unit. (Address assignment of 8000H~BFFFH)

  Y3 ($\overline{1Y3}$) ..... With high state of AD14 and AD15, the Y3 output goes low so as to select the system program ROM (SC61328F) and the I/O port (LH5811 or LH5810). (Address assignment of C000H~FFFFH)

- **S0~S7 is selected by the decoder IC (TC40H138F) when the gate signal input ME0 (G1) is high, Y1 ($\overline{G2A}$) low, and $\overline{G2B}$ is low (normally low).**

  S0 ($\overline{Y0}$) ..... With all of AD11, AD12 and AD13 in low state, the S0 output goes low so as to select the user RAM (HM6116). (Address assignment of 4000H~47FFH)

  S1 ($\overline{Y1}$) ..... With high state of AD11 and low state of AD12 and AD13, the S1 output goes low so as to select the optional user RAM area. (Address assignment of 4800H~4FFFH)

  S2 ($\overline{Y2}$) ..... With low state of AD11 and AD13 and high state of AD12, the S2 output goes low so as to select the optional user RAM area. (Address assignment of 5000H~57FFH)

  S3 ($\overline{Y3}$) ..... With low state of AD11 and AD12 and high state of AD13, the S3 output goes low so as to select the user RAM area. (Address assignment of 5800H~5FFFH)

  S4 ($\overline{Y4}$) ..... With high state of AD11 and AD13 and low state of AD12, the S4 output goes low so as to select the user RAM area. (Address assignment of 6000H~67FFH)

  S5 ..... Do not use.

  S6 ($\overline{Y6}$) ..... With low state of AD11 and high state of AD12 and AD13, the S6 output goes low so as to receive on the I/O port the data from the display chip RAM or interrupt input from the option. (Address assignment of 7000H~77FFH)

  S7 ($\overline{Y7}$) ..... With all of AD11, AD12, and AD13 in high state, the S7 output goes low so as to select the system RAM (TC5514) (Address assignment of 7800H~7FFFH)

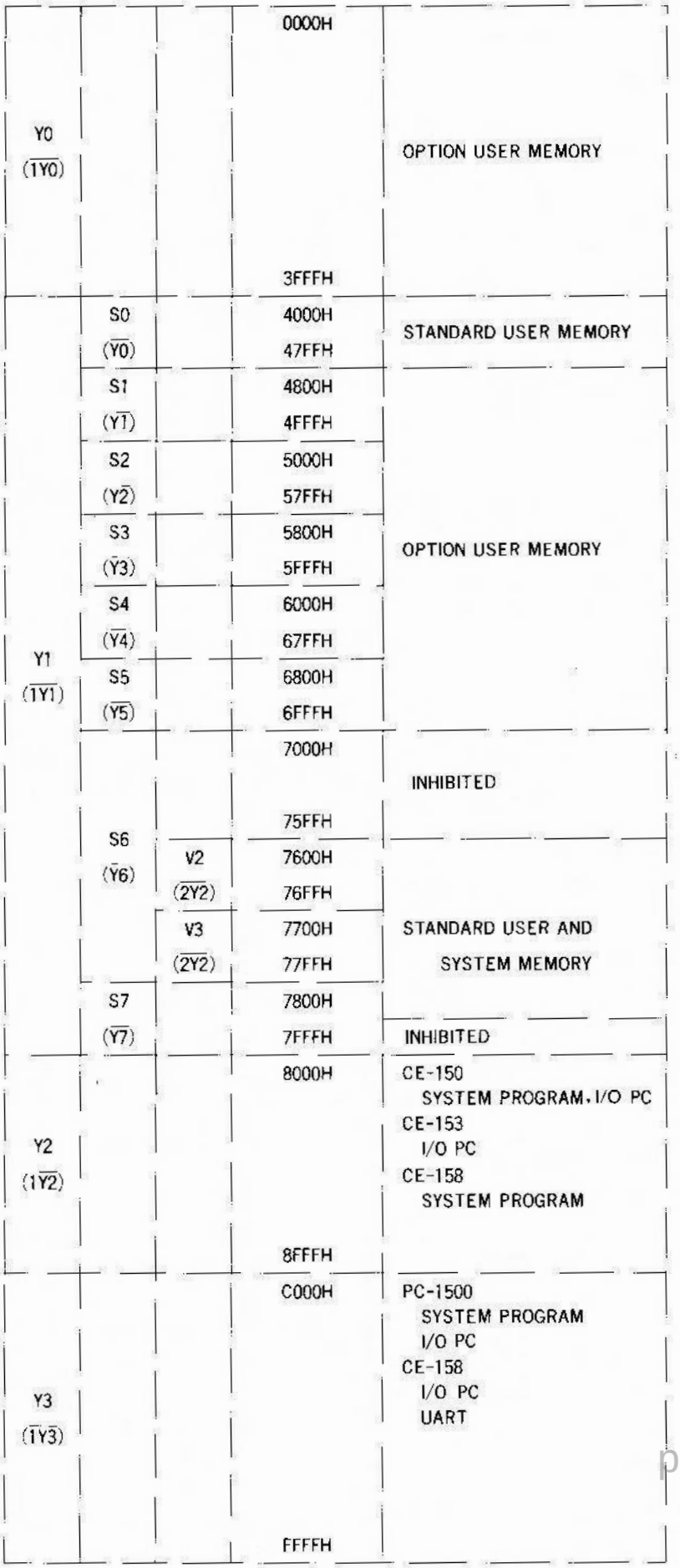
- **$\overline{2Y2}$, $\overline{2Y3}$ is selected by the decoder IC (TC40H139) when the $\overline{2G}$ gate becomes effective after the selection (low state) of the TC40H138F output S6 ($\overline{Y6}$).**

  V2 ($\overline{2Y2}$) ..... With low state of AD8 and high state of DME0, the $\overline{2Y2}$ output goes low and makes the NAND gate output V2 high so as to select display chip 1 and 3.(Address assignment of 7600H~76FFH)

  V3 ($\overline{2Y3}$) ..... With high state of AD8 and DME0, the $\overline{2Y3}$ output goes low and makes the NAND gate output V3 high so as to select display chip 2 and 4. (Address assignment of 7700H~77FFH)

promsoft.com

Chip select signal



NOTE: S0~S7, V2, and V3 are applicable only for ME0 area.

promsoft.com

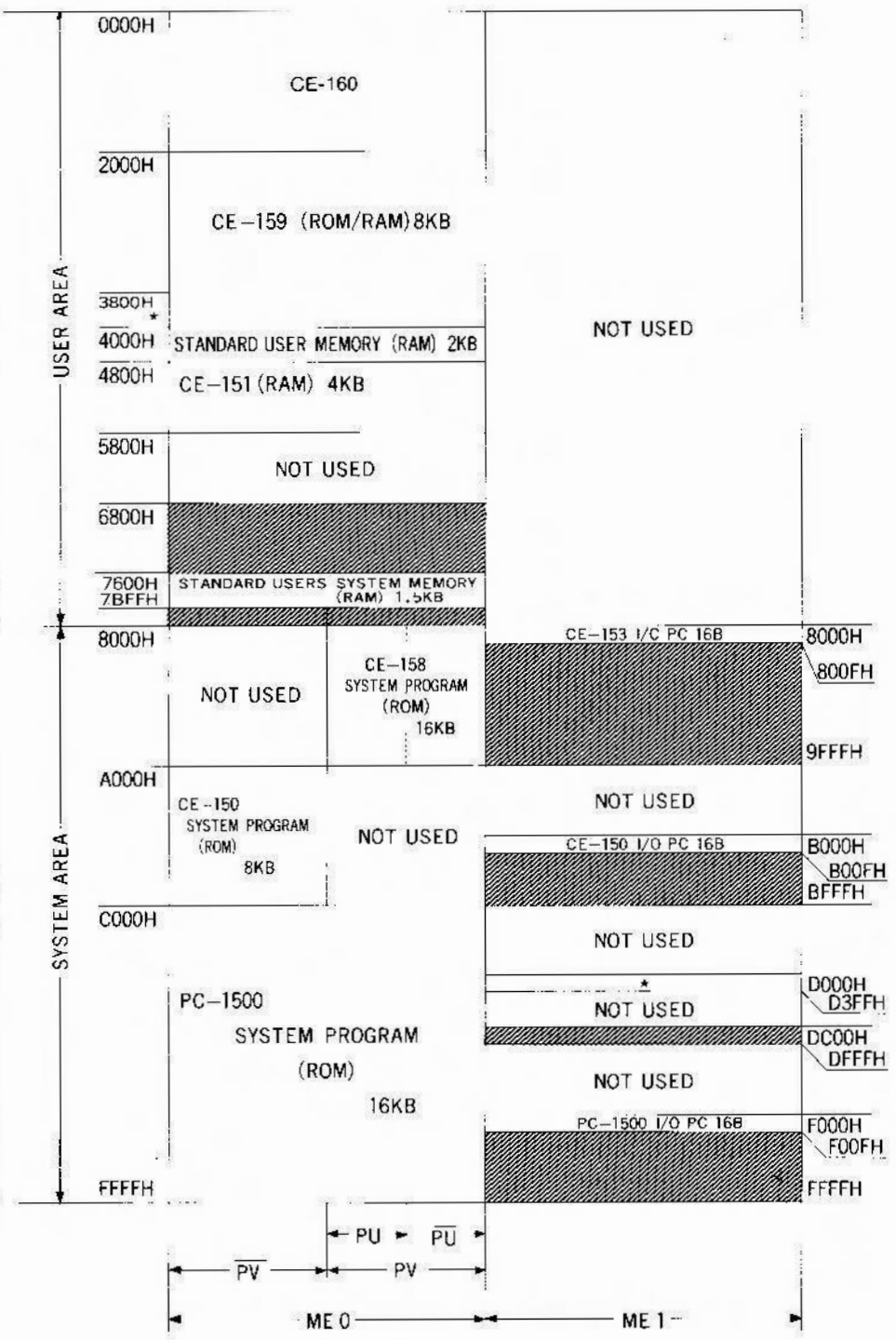
4-2-4. PC-1500 system memory map

① In the ME0 area is contained such as the system program and user RAM area and the ME1 is used for such as I/O port.
Memory bank is assigned by PV and $\overline{PV}$ for the ME0 area of 8000H thru BFFFH, and PU and $\overline{PU}$ are used to assign the PV area of 8000H thru 9FFFH. Address 0000H thru 7FFFH is used for the user area and 8000H thru FFFFH for the system area.
The system area is used by the mnemonic programming system by which the CPU, I/O PC, etc. are controlled, and the user area is used to store the program written in BASIC, mnemonic programming language, and data.
With the PC-1500, ME0 memory area 4000H thru 47FFH and 7600H thru 7BFFH are used for the user memory and C000H thru FFFFH for the system program.
When the CE-150 or CE-158 is connected, the system program, I/O port, and RAM are arranged as shown in the next page.

② Upon power on to the PC-1500, initialization starts for peripheral units according to the system program residing in the system area, and the user area ROM and RAM are checked, then it becomes ready for a key entry.

promsoft.com

MEMORY MAP I

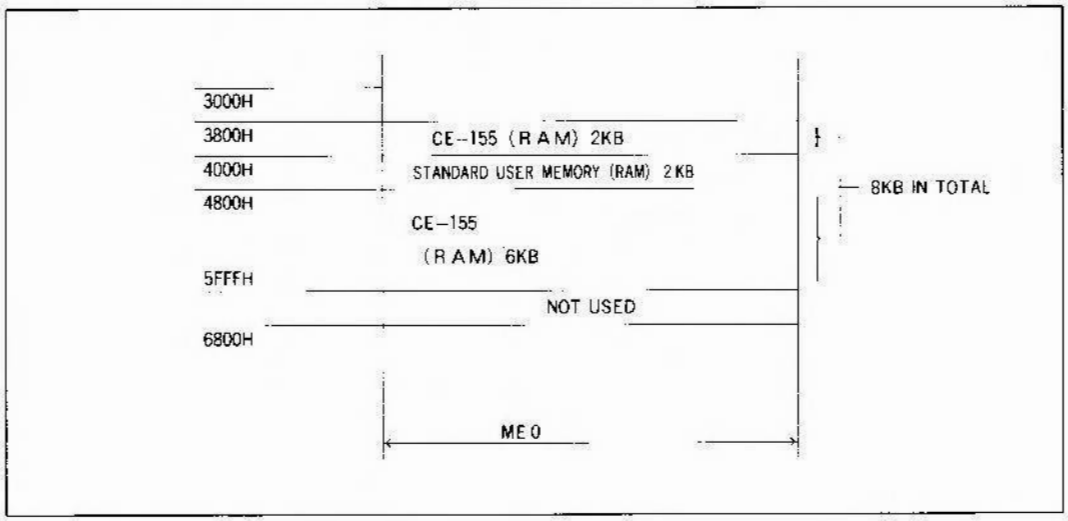


- Area reserved for option will become unused, in case the respective option is not connected.
- Refer to "Memory map II and III" for *
- PV and PU are bank select signals.
- Shadowed area can not be used.

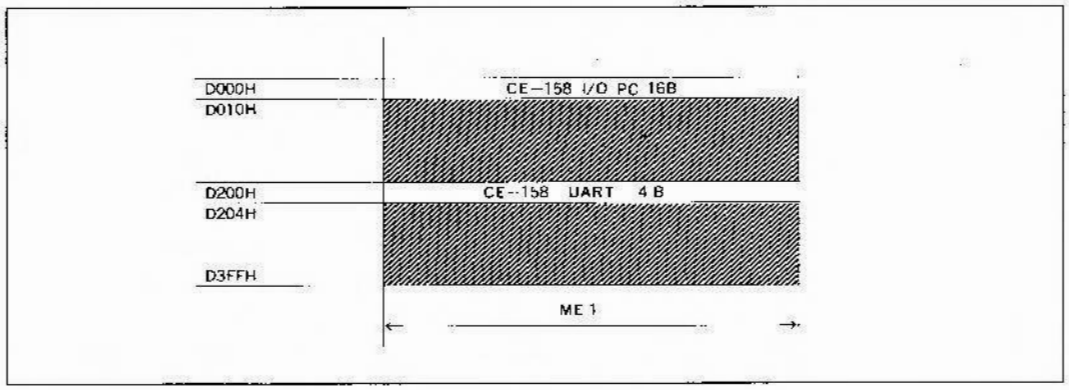
promsoft.com

MEMORY MAP Ⅱ

MEMORY MAP when the CE-155 is used.



MEMORY MAP Ⅲ



MEMORY MAP Ⅳ STANDARD USER & SYSTEM MEMORY (RAM) 1.5KB

7600H~76FFH

HIGH / LOW	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0						E	F	G	H	I	J	K	L	M	N	O
2						$	$	$	$	$	$	$	$	$	$	$
4																
6	DISPLAY															
8	BUFFER															
A																
C																
E																

7700H~77FFH

HIGH / LOW	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
2																
4																
6	DISPLAY					P	Q	R	S	T	U	V	W	X	Y	Z
8	BUFFER					$	$	$	$	$	$	$	$	$	$	$
A																
C																
E																

7800H~78FFH

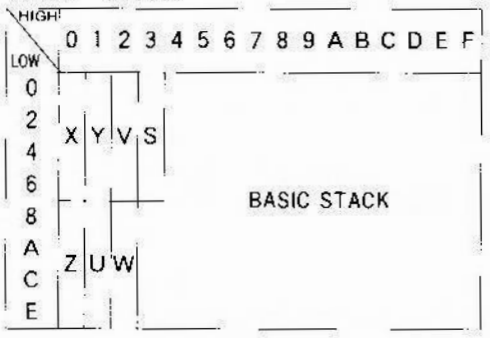
HIGH / LOW	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
2																
4																
6	SYSTEM					SYSTEM					A	B	C	D		
8	STACK					MEMORY					$	$	$	$		
A																
C																
E																

7900H~79FFH

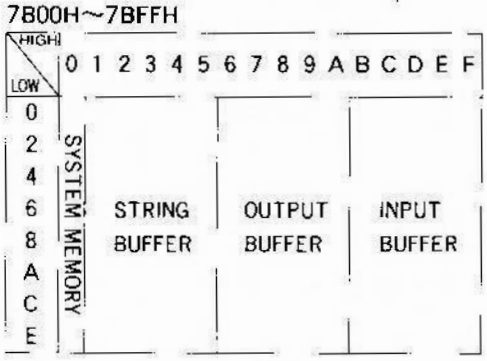
HIGH / LOW	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
2		A	C	E	G	I	K	M	O	Q	S	U	W	Y		
4																
6															SYSTEM	
8															MEMORY	
A		B	D	F	H	J	L	N	P	R	T	V	X	Z		
C																
E																

promsoft.com

7A00H~7AFFH



7B00H~7BFFH



- A~Z (7900H~79CFH) are fixed numerical variables.
- A$~Z$ are fixed character variables.
- The arithmetic registers X, Y, V, S, Z, U and W (7A00H~7A37H) are different from fixed numerical variables.
- BASIC stack includes stacks for FOR—NEXT, GOSUB, DATA, FUNCTION.
- See the next page for details of the display buffer.
- For system memory, refer to MEMORY MAP VII.

MEMORY MAP V

I/O port

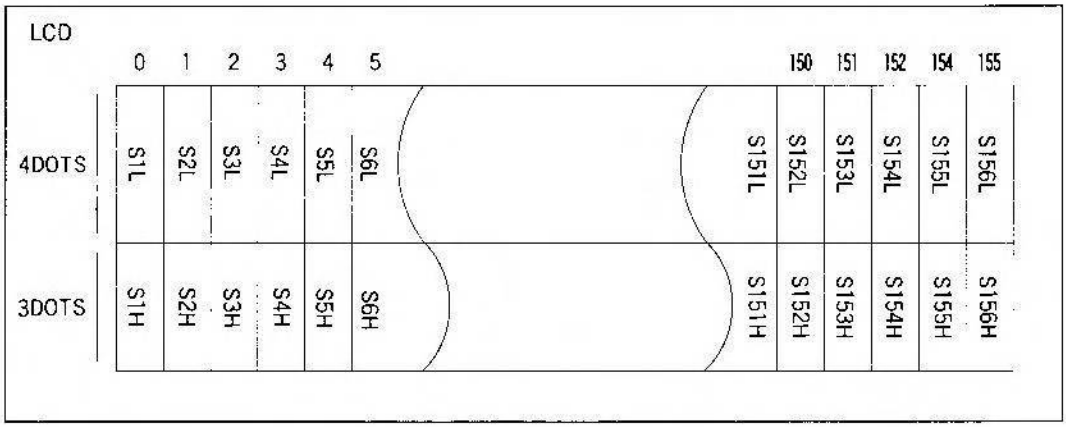
	PC–1500	CE–150	CE–153	CE–158
Do not use	F000H	B000H	8000H	D000H
Do not use	F001H	B001H	8001H	D001H
Do not use	F002H	B002H	8002H	D002H
Do not use	F003H	B003H	8003H	D003H
Divider reset	F004H	B004H	8004H	D004H
U register output	F005H	B005H	8005H	D005H
Serial transfer	F006H	B006H	8006H	D006H
Load divider to F register	F007H	B007H	8007H	D007H
Port C input/output	F008H	B008H	8008H	D008H
G register input/output	F009H	B009H	8009H	D009H
MSK register input/output	F00AH	B00AH	800AH	D00AH
IF register input/output	F00BH	B00BH	800BH	D00BH
Specify port A I/O direction	F00CH	B00CH	800CH	D00CH
Specify port B I/O direction	F00DH	B00DH	800DH	D00DH
Port A input/output	F00EH	B00EH	800EH	D00EH
Port B input/output	F00FH	B00FH	800FH	D00FH

pcmsoft.com

Memory map VI

Display buffer

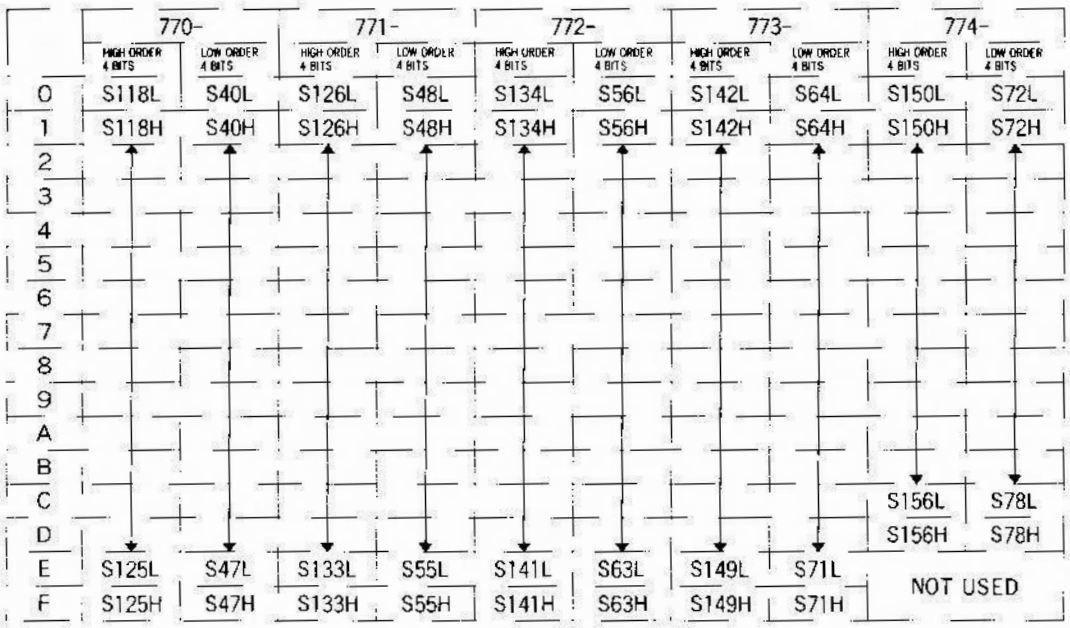
Since there are 7 × 156 dots with 14 additional dots of symbols (DEF, I, PRO, etc.), each dot is allocated with a corresponding bit of the memory in order to control activation of dots. Address is represented by the hexadecimal notation and "H" is used to represent high state and "L" to represent low state of segment drive signals.



	760–		761–		762–		763–		764–	
	HIGH ORDER 4 BITS	LOW ORDER 4 BITS	HIGH ORDER 4 BITS	LOW ORDER 4 BITS	HIGH ORDER 4 BITS	LOW ORDER 4 BITS	HIGH ORDER 4 BITS	LOW ORDER 4 BITS	HIGH ORDER 4 BITS	LOW ORDER 4 BITS
0	S79L	S1L	S87L	S9L	S95L	S17L	S103L	S25L	S111L	S31L
1	S79H	S1H	S87H	S9H	S95H	S17H	S103H	S25H	S111H	S31H
2	S80L	S2L								
3	S80H	S2H								
4	81	3								
5	81	3								
6	82	4								
7	82	4								
8	83	5								
9	83	5								
A	84	6								
B	84	6								
C	85	7							S117L	S39L
D	85	7							S117H	S39H
E	S86L	S8L	S94L	S16L	S102L	S24L	S110L	S30L		
F	S86H	S8H	S94H	S16H	S102H	S24H	S110H	S30H		

ADDRESS	HIGH ORDER 4 BITS				LOW ORDER 4 BITS			
764EH	DEF	I	II	III	SMALL	SML	SHIFT	BUSY
764FH	NOT USED	RUN	PRO	RESERVE	NOT USED	RAD	G	DE

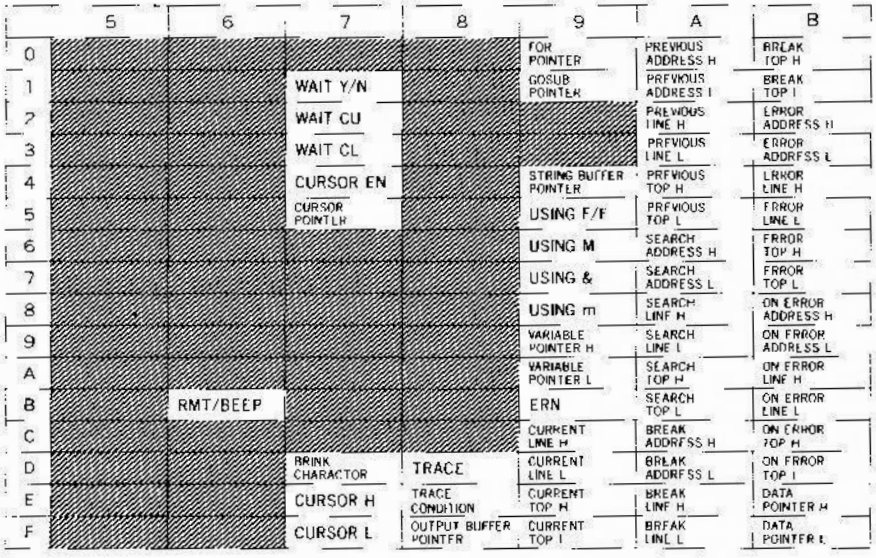
promsoft.com



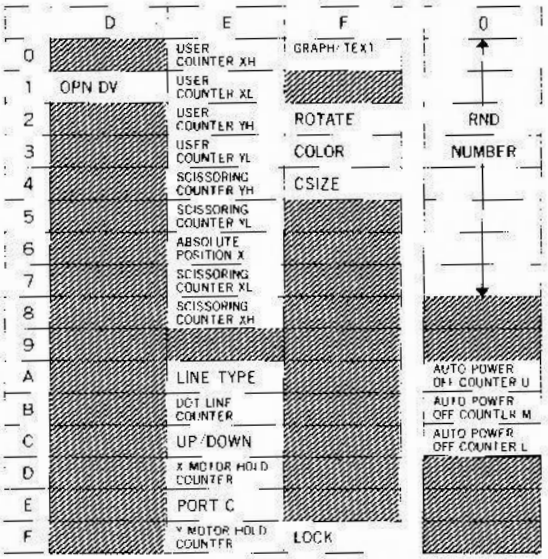
Memory Map VII

MEMORY SYSTEM DETAIL

78XX



79XX



7BXX

promsoft.com

Address	Name	Function
786BH	RMT/BEEP	Remote and beep on/off pointer
7871H	WAIT Y/N	WAIT(0), WAIT0(3), WAIT1(2)
7872H	WAIT COUNTER H	WAIT counter
7873H	WAIT COUNTER L	
7874H	CURSOR ENABLE	(01H) if used. (00H) if not.
7875H	CURSOR POINTER	Cursor pointer (0~155)
787DH	BLINK CHARACTER	Character code to be blinked.
787EH	BLINK CURSOR H	Blinking cursor position (address of the display buffer)
787FH	BLINK CURSOR L	
788DH	TRACE	Trace on/off pointer
788EH	TRACE CONDITION	Status when trace on.
788FH	OUTPUT BUFFER POINTER	Output buffer pointer
7890H	FOR POINTER	FOR-NEXT stack pointer
7891H	GOSUB POINTER	GOSUB pointer
7894H	STRING BUFFER POINTER	String buffer pointer
7895H	USING F/F	Using format (presence of decimal point, comma, etc.)
7896H	USING M	Integer part of Using
7897H	USING &	Using of character string
7898H	USING m	Decimal part of Using
7899H	VARIABLE POINTER H	Variable pointer
789AH	VARIABLE POINTER L	
789BH	ERL	Error number when occurred.
789CH	CURRENT LINE H	Current line number
789DH	CURRENT LINE L	
789EH	CURRENT TOP H	Leading address of program of the current line
789FH	CURRENT TOP L	
78A0H	PREVIOUS ADDRESS H	Address of immediately preceding line
78A1H	PREVIOUS ADDRESS L	
78A2H	PREVIOUS LINE H	Line number immediately preceding
78A3H	PREVIOUS LINE L	
78A4H	PREVIOUS TOP H	Leading address of program of the line immediately preceding
78A5H	PREVIOUS TOP L	
78A6H	SEARCH ADDRESS H	Address of the line found during search
78A7H	SEARCH ADDRESS L	
78A8H	SEARCH LINE H	Line number found after search
78A9H	SEARCH LINE L	
78AAH	SEARCH TOP H	Leading address of the searched program block
78ABH	SEARCH TOP L	
78ACH	BREAK ADDRESS H	Address of breakpoint
78ADH	BREAK ADDRESS L	
78AEH	BREAK LINE H	Address of breakpoint line number
78AFH	BREAK LINE L	
78B0H	BREAK TOP H	Top address of the program block to which break is applied.
78B1H	BREAK TOP L	
78B2H	ERROR ADDRESS H	Address where error is met.
78B3H	ERROR ADDRESS L	
78B4H	ERROR LINE H	Line number where error is met.
78B5H	ERROR LINE L	

promsoft.com

Address	Name	Function
78B6H	ERROR TOP H	Leading address of the program block in which error is met.
78B7H	ERROR TOP L	
78B8H	ON ERROR ADDRESS H	Address to which program jumps when an error is met.
78B9H	ON ERROR ADDRESS L	
78BAH	ON ERROR LINE H	Line number to which program jumps when an error is met.
78BBH	ON ERROR LINE L	
78BCH	ON ERROR TOP H	Leading address of program block in which an error is met.
78BDH	ON ERROR TOP L	
78BEH	DATA POINTER H	Pointer for data statement
78BFH	DATA POINTER L	
79D1H	OPN DV	Peripheral device select
79E0H	USER COUNTER XH	Counter by which X-coordinates of the pen are indicated.
79E1H	USER COUNTER XL	
79E2H	USER COUNTER YH	Counter by which Y-coordinates of the pen are indicated.
79E3H	USER COUNTER YL	
79E4H	SCISSORING COUNTER YH	Y-direction scissoring counter
79E5H	SCISSORING COUNTER YL	
79E6H	ABSOLUTE POSITION X	X-direction absolute point counter
78E7H	SCISSORING COUNTER XL	X-direction scissoring counter
79E8H	SCISSORING COUNTER XH	
79EAH	LINE TYPE	Kind of line
79EBH	DOT LINE COUNTER	Dot line counter
79ECH	UP/DOWN	Pen up/down position select
79EDH	X MOTOR HOLD COUNTER	X motor hold counter
79EEH	PORT C	Indicates current motor phase.
79EFH	Y MOTOR HOLD COUNTER	Y motor hold counter
79F0H	GRAPH/TEXT	Printer mode select (graph=255, text=0)
79F2H	ROTATE	Printing direction select
79F3H	COLOR	Color select
79F4H	CSIZE	Printing character size select
79FFH	LOCK	Lock/unlock select
7B00H	RND NUMBER	Random number
7B01H	RND NUMBER	
7B02H	RND NUMBER	
7B03H	RND NUMBER	
7B04H	RND NUMBER	
7B05H	RND NUMBER	
7B06H	RND NUMBER	
7B07H	RND NUMBER	
7B0AH	AUTO P-OFF COUNTER U	Auto power off counter
7B0BH	AUTO P-OFF COUNTER M	
7B0CH	AUTO P-OFF COUNTER L	

promsoft.com

4-3. Connector signals/LSI signals

Note that there may be a different kind of connector used for the 40 and 60 pin connector of the PC-1500 on account of product revision. The 64-pin connector on the back of the CE-150 is compatible with the 60-pin connector of the PC-1500. Either the LH5811 or LH5810 is used for the I/O PC. The LH5811 is an upgraded version of the LH5810.

4-3-1. 40-pin connector

Pin no.	Signal name	Description
1	VCC	VCC
2	PV	Chip select
3	PU	Chip select
4	Y0	Address designation of 0000H~3FFFH
5	S4	Address designation of 6000H~6700H
6	DME0	Chip select with WAIT condition in consideration (ME0 area assignment)
7	D7	Data bus
8	D6	Data bus
9	D5	Data bus
10	D4	Data bus
11	D3	Data bus
12	D2	Data bus
13	D1	Data bus
14	D0	Data bus
15	INHIBIT	Prohibits ROM select of the PC-1500, when connected to GND.
16	S1	Address designation of 4800H~4FFFH
17	S2	Address designation of 5000H~57FFH
18	S3	Address designation of 5800H~5FFFH
19	Y2	Address designation of 8000H~BFFFH
20	VGG	VGG
21	GND	GND
22	AD15	Address bus
23	AD14	Address bus
24	AD13	Address bus
25	AD12	Address bus
26	AD11	Address bus
27	AD10	Address bus
28	AD9	Address bus
29	AD8	Address bus
30	AD7	Address bus
31	AD6	Address bus
32	AD5	Address bus
33	AD4	Address bus
34	AD3	Address bus
35	AD2	Address bus
36	AD1	Address bus
37	AD0	Address bus
38	OD	Output disable
39	R/W	Memory read/write
40	GND	GND

NOTE: S4 of No. 5 may be NC. depending on production month.

promsoft.com

4-3-2. 60-pin connector

Pin no.	Signal name	Description
1	AD7	Address bus
2	AD6	Address bus
3	AD5	Address bus
4	AD4	Address bus
5	AD3	Address bus
6	AD2	Address bus
7	AD1	Address bus
8	AD0	Address bus
9	PB0	Not used
10	PC7	Not used
11	VCC	VCC
12	VCC	VCC
13	NC	NC
14	NC	NC
15	PV	Chip select
16	PU	Chip select
17	D7	Data bus
18	D6	Data bus
19	D5	Data bus
20	D4	Data bus
21	D3	Data bus
22	D2	Data bus
23	D1	Data bus
24	D0	Data bus
25	$\overline{\text{INHIBIT}}$	Prohibits ROM select of the PC-1500, when connected to GND.
26	$\overline{\text{WEX}}$	External WAIT signal
27	$\overline{\text{CMTIN}}$	Cassette data input
28	W1	WAIT condition input
29	CMTOUT	Cassette data output
30	$\overline{\text{INT}}$	Interrupt request to CPU
31	AD8	Address bus
32	AD9	Address bus
33	AD10	Address bus
34	AD11	Address bus
35	AD12	Address bus
36	AD13	Address bus
37	AD14	Address bus
38	AD15	Address bus
39	PB1	Not used
40	NC	NC
41	VCC	VCC
42	VCC	VCC
43	F·GND	Frame GND
44	VBAT	VBAT
45	VBAT	VBAT
46	VBAT	VBAT
47	VBAT	VBAT
48	VBAT	VBAT
49	NC	NC
50	BFO	VCC output
51	ϕOS	Clock in the same phase as the LSI internal clock
52	GND	GND
53	GND	GND
54	GND	GND
55	GND	GND
56	DME0	Chip select taking consideration of WAIT condition (ME0 area designation)
57	R/W	Memory read/write signal
58	DME1	Chip select taking consideration of WAIT condition (ME1 area designation)
59	ME1	ME1 area designation
60	OD	Output disable

NOTE: PB0 of No. 9 may be NC depending on production month.
PB1 of No. 39 may be NC depending on production month.

promsoft.com

4-3-3. LH5801 Microprocessor

Pin no.	Signal name	Description
1	RESET	Reset input
2	NC	NC
3	BRO	Fixed to GND level
4	BFI	ON key input signal
5	VGG	VGG
6	BFO	VCC output
7	OFF	Not used
8	BAK	Not used
9	VCC	VCC (4.0~4.7V)
10	VGG	VGG (4.0~4.7V)
11	VM	LCD power supply
12	VDIS	LCD power supply
13	VA	LCD power supply
14	VB	LCD power supply
15	NVI	GND
16	MI	Maskable interrupt input
17	HIN	LCD backplate signal
18	HA	LCD backplate signal
19	DISP	LCD on/off control
20	H7	Not used
21	H6	Backplate control
22	H5	Backplate control
23	H4	Backplate control
24	H3	Backplate control
25	H2	Backplate control
26	H1	Backplate control
27	H0	Backplate control
28	OD	Output disable
29	ME0	ME0 area designation
30	ME1	ME1 area designation
31	D0	Data bus
32	D1	Data bus
33	D2	Data bus
34	D3	Data bus
35	D4	Data bus
36	D5	Data bus
37	D6	Data bus
38	D7	Data bus
39	AD0	Address bus
40	AD1	Address bus
41	AD2	Address bus
42	AD3	Address bus
43	AD4	Address bus
44	AD5	Address bus
45	AD6	Address bus
46	AD7	Address bus
47	GND	GND
48	AD8	Address bus
49	VGG	VGG
50	AD9	Address bus
51	AD10	Address bus
52	AD11	Address bus
53	AD12	Address bus
54	AD13	Address bus
55	AD14	Address bus
56	AD15	Address bus
57	NC	NC
58	R/W	Memory read/write
59	P0	Not used
60	PV	Chip select
61	PU	Chip select
62	φOS	Clock which is in the same phase as the LSI internal clock
63	XI0	2.6MHz oscillator input
64	XI1	2.6MHz oscillator output
65	WAIT	CPU WAIT signal
66	IN7	Key input port
67	IN6	Key input port
68	IN5	Key input port
69	IN4	Key input port
70	IN3	Key input port
71	IN2	Key input port
72	IN1	Key input port
73	IN0	Key input port
74	N2	NC
75	N1	NC
76	N0	NC

promsoft.com

4-3-4. I/O PC

When the pin No. 34 (ME0) of the I/O PC is connected with ME1 of the CPU, it will be accessed as the ME1 area.

① PC-1500 I/O PC

Pin no.	Signal name	Description
1	PA1	Key strobe
2	PA2	Key strobe
3	PA3	Key strobe
4	PA4	Key strobe
5	PA5	Key strobe
6	PA6	Key strobe
7	PA7	Key strobe
8	GND	GND
9	PB0	Not used
10	PB1	Not used
11	PB2	Serial data input from the cassette tape
12	PB3	VCC (export model), GND (domestic model)
13	PB4	GND
14	PB5	Timer control
15	PB6	Timer control
16	PB7	ON key input
17	Pϕ	GND
18	PC0	Timer control
19	PC1	Timer control
20	PC2	Timer control
21	PC3	Timer control
22	PC4	Timer control
23	PC5	Timer control
24	PC6	Buzzer on/off control
25	PC7	Not used
26	CS0	Chip select (AD12)
27	CS1	Chip select (AD13)
28	$\overline{CS2}$	Chip select
29	RS0	Chip select (AD0)
30	RS1	Chip select (AD1)
31	RS2	Chip select (AD2)
32	RS3	Chip select (AD3)
33	R/W	Memory read/write
34	ME0	ME0 area designation
35	ME1	ME1 area designation
36	W0	WAIT condition input
37	W1	WAIT condition input
38	GND	GND
39	VCC	VCC
40	DME0	Chip select taking consideration of WAIT condition (assignment of ME0 area).
41	DME1	Chip select taking consideration of WAIT condition (assignment of ME1 area).
42	WAIT	WAIT to CPU
43	INT	Interrupt to CPU
44	RESET	I/O port reset input
45	IRQ	Interrupt input
46	ϕOS	Basic clock input
47	CL1	CMT demodulation clock input
48	SD1	VCC level
49	LC	Not used
50	CL0	CMT signal
51	SD0	Serial data output to cassette tape
52	D0	Data bus
53	D1	Data bus
54	D2	Data bus
55	D3	Data bus
56	D4	Data bus
57	D5	Data bus
58	D6	Data bus
59	D7	Data bus
60	PA0	Key strobe

promsoft.com

② CE-150 I/O PC

Pin no.	Signal name	Description
1	PA1	Remote 0 control
2	PA2	Remote 0 control
3	PA3	Remote 1 control
4	PA4	Remote 1 control
5	PA5	Manual print mode assignment
6	PA6	Not used
7	PA7	Not used
8	GND	GND
9	PB0	Pen ascending signal
10	PB1	Pen descending signal
11	PB2	Color detection
12	PB3	Not used
13	PB4	Not used
14	PB5	Not used
15	PB6	Low battery
16	PB7	Paper feed key input
17	Pϕ	Not used
18	PC0	X motor control
19	PC1	X motor control
20	PC2	X motor control
21	PC3	X motor control
22	PC4	Y motor control
23	PC5	Y motor control
24	PC6	Y motor control
25	PC7	Y motor control
26	CS0	AD13 input/output
27	CS1	AD12 input/output
28	$\overline{CS2}$	I/O port address designation
29	RS0	AD0
30	RS1	AD1
31	RS2	AD2
32	RS3	AD3
33	R/W	Memory read/write
34	ME0	ME0 area designation
35	ME1	GND
36	W0	GND
37	W1	WAIT condition output
38	GND	GND
39	VCC	VCC
40	DME0	Not used
41	DME1	Not used
42	WAIT	WAIT condition output
43	INT	Interrupt request to CPU
44	RESET	LH5811 reset
45	IRQ	Interrupt request from external source
46	ϕOS	Internal clock having the same phase as LSI
47	CL1	GND
48	SD1	GND
49	LC	Not used
50	CL0	Not used
51	SD0	Not used
52	D0	Data bus
53	D1	Data bus
54	D2	Data bus
55	D3	Data bus
56	D4	Data bus
57	D5	Data bus
58	D6	Data bus
59	D7	Data bus
60	PA0	For stable input level when CMT is not in use.

promsoft.com

③ CE-153 I/O PC

Pin no.	Signal name	Description
1	PA1	Key input port
2	PA2	Key input port
3	PA3	Key input port
4	PA4	Key input port
5	PA5	Key input port
6	PA6	Key input port
7	PA7	Key input port
8	GND	GND
9	PB0	Key input port
10	PB1	Key input port
11	PB2	Key strobe
12	PB3	Key strobe
13	PB4	Key strobe
14	PB5	Key strobe
15	PB6	Key strobe
16	PB7	Key strobe
17	Pϕ	GND
18	PC0	Key strobe
19	PC1	Key strobe
20	PC2	Key strobe
21	PC3	Key strobe
22	PC4	Key strobe
23	PC5	Key strobe
24	PC6	Key strobe
25	PC7	Key strobe
26	CS0	AD15
27	CS1	AD14
28	$\overline{CS2}$	AD13
29	RS0	AD0
30	RS1	AD1
31	RS2	AD2
32	RS3	AD3
33	R/W	Memory read/write
34	ME0	ME0 area designation
35	ME1	GND
36	W0	GND
37	W1	GND
38	GND	GND
39	VCC	VCC (4.70±0.02V)
40	DME0	Not used
41	DME1	Not used
42	WAIT	WAIT condition
43	$\overline{INT}$	Not used
44	RESET	Reset
45	$\overline{IRQ}$	GND
46	ϕOS	Internal clock having the same phase as LSI.
47	CL1	GND
48	SD1	GND
49	LC	Not used
50	CL0	Not used
51	SD0	Not used
52	D0	Data bus
53	D1	Data bus
54	D2	Data bus
55	D3	Data bus
56	D4	Data bus
57	D5	Data bus
58	D6	Data bus
59	D7	Data bus
60	PA0	Key input port

promsoft.com

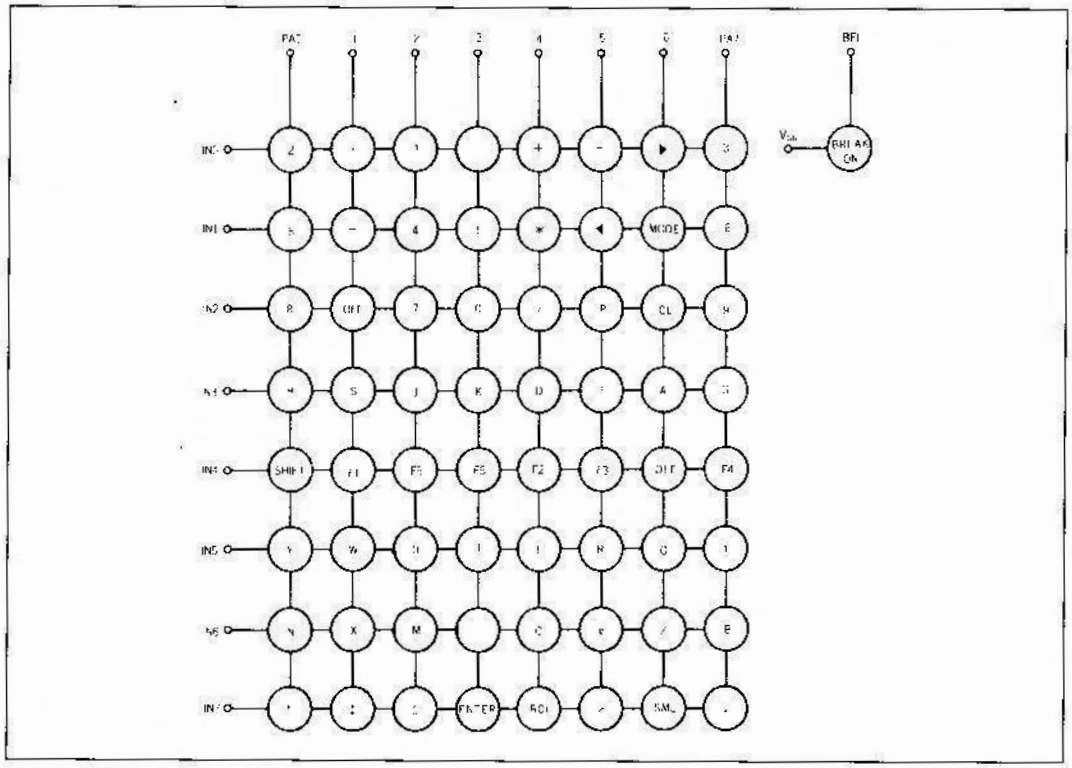
④ CE-158 I/O PC

Pin no.	Signal name	Description
1	PA1	RS232C I/F send request
2	PA2	RS232C I/F ready to receive
3	PA3	RS232C I/F carrier detect
4	PA4	RS232C I/F data set ready
5	PA5	Low battery
6	PA6	Baud rate select
7	PA7	Baud rate select
8	GND	GND
9	PB0	Centronics parallel I/F DATA 2
10	PB1	Centronics parallel I/F DATA 3
11	PB2	Centronics parallel I/F DATA 4
12	PB3	Centronics parallel I/F DATA 5
13	PB4	Centronics parallel I/F DATA 6
14	PB5	Centronics parallel I/F DATA 7
15	PB6	Centronics parallel I/F DATA 8
16	PB7	Centronics parallel I/F BUSY input
17	Pϕ	GND
18	PC0	Baud rate select
19	PC1	Baud rate select
20	PC2	Baud rate select
21	PC3	Baud rate select
22	PC4	Baud rate select
23	PC5	Centronics parallel I/F DATA 1
24	PC6	Centronics parallel I/F $\overline{\text{STROBE}}$
25	PC7	Centronics parallel I/F $\overline{\text{INIT}}$
26	CS0	VCC (4.70±0.02V)
27	CS1	VCC (4.70±0.02V)
28	$\overline{\text{CS2}}$	GND
29	RS0	AD0
30	RS1	AD1
31	RS2	AD2
32	RS3	AD3
33	R/W	Memory read/write
34	ME0	ME0 area designation
35	ME1	GND
36	W0	GND
37	W1	GND
38	GND	GND
39	VCC	VCC (4.70±0.02V)
40	DME0	Not used
41	DME1	Not used
42	WAIT	Not used
43	INT	Interrupt request to CPU
44	RESET	LH5811 reset
45	IRQ	Interrupt request
46	ΦOS	Internal clock having the same phase as LSI.
47	CL1	GND
48	SD1	GND
49	LC	Not used
50	CL0	Not used
51	SD0	Not used
52	D0	Data bus
53	D1	Data bus
54	D2	Data bus
55	D3	Data bus
56	D4	Data bus
57	D5	Data bus
58	D6	Data bus
59	D7	Data bus
60	PA0	RS232C I/F terminal ready

promsoft.com

4-4. Key matrix and key code chart

Key matrix



Key code chart

HIGH / LOW	0	1	2	3	4	5
0			SPACE	0		P
1	SHIFT	F1		1	A	Q
2	SML	F2		2	B	R
3		F3		3	C	S
4		F4		4	D	T
5		F5		5	E	U
6		F6		6	F	V
7				7	G	W
8	◀	CL	(	8	H	X
9	▲▼	RCL	)	9	I	Y
A	↑		*		J	Z
B	↓	DEF	+		K	
C	▶				L	
D	ENTER		−	=	M	
E			.		N	
F	OFF	MODE	/		O	

NOTE: For ON key, refer to "KEY SCAN" of "SYSTEM SUBROUTINE".

promsoft.com

promsoft.com

5

PC-1500 software

promsoft.com

5-1. BASIC command related PC-1500 machine language

In this section we will discuss the BASIC command related PC-1500 machine language.

5-1-1. NEW

Format (PRO mode)

NEW ┌ expression ┐

① NEW expression (=0)

Clears program and all data areas and secures the BASIC program area following to the reserve program area.

	Top of the BASIC program
Operation of the PC-1500 only	40C5H
Operation in conjuction with the CE-155	38C5H

See (System memory)

② NEW expression (≠0)

Effective when there is no user ROM area.

Clears BASIC program and all data areas and sets the BASIC program area from the address represented in the expression, which is valid within an address range of RAM area except the reserve area.

③ NEW

Clears BASIC program and all data areas without affecting the BASIC program area. When executed in the reserve mode, it clears the reserve area.

5-1-2. STATUS

Format

Status → expression →

① STATUS 0

Similar to MEM, the sum of unoccupied memory size and data memory size is given in terms of bytes.

② STATUS 1

Size of the memory used for the BASIC program is given in terms of bytes.

③ STATUS 2

The last BASIC program address plus one address (next to FFH) is given.

④ STATUS 3

The smallest address where data stored is given.

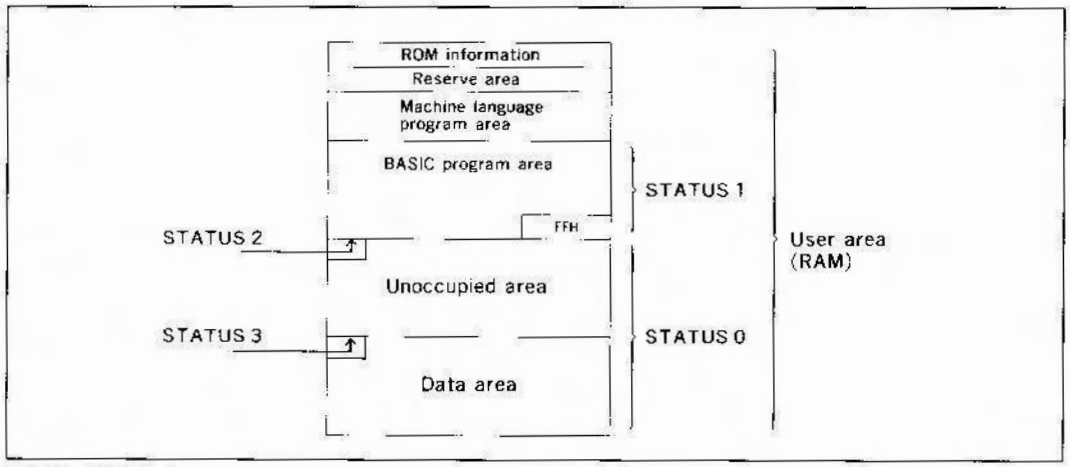
⑤ STATUS 4

When BASIC program is in execution, the line number executed immediately before.

When program execution is at halt, the current line number.

In otherwise condition, "0". Same for STATUS 5 thru 255.

promsoft.com



NOTE: ROM information is 8 bytes and the reserve area is 189 bytes.

5-1-3. PEEK

Format

PEEK [#] **expression** →

① **PEEK expression**

Recalls the data in ME0 area whose address is represented by the expression.

② **PEEK # expression**

Recalls the data in the ME1 area whose address is represented by the expression.

5-1-4. POKE

Format

POKE [#] **expression-1** [, → **expression-*i***]

① **POKE expression-1, expression-2, expression-3, ...**

Starting from the ME0 area address represented by the expression-1, it begins to store successive data in order of the expression-2, expression-3, ...

② **POKE # expression-1, expression-2, expression3, ...**

Starting from the ME1 area address represented by the expression-1, it begins to store successive data in order of the expression-2, expression-3, ...

CAUTION: Since the I/O port controller is in the ME1 area for the PC-1500, care must be taken not to have a wrong use of this command, as it may possibly result in destruction.

(EX): POKE &4700, &01, &02, &03

Address	Data
4700H	01H
4701H	02H
4703H	03H

promsoft.com

5-1-5. CALL

Format

CALL → expression ┌ , → variable ┐

① CALL expression

Machine language program starts to execute starting from address specified by the expression.

Progam returns from the machine language routine by the command.

② CALL expression, variable

1) When the variable is a numerical variable (within a range of −32768 thru 32767)
   1. The value of the variable is transferred to the Xreg.
   2. Machine language program is executed from the starting address represented by the expression.
   3. If there is a carry when returns, the value of the Xreg is transferred to the variable.
2) When the variable is a nonnumeric variable
   1. The leading address of the nonnumeric variable is transferred to the Xreg and the size information of the nonnumeric variable is transferred to the accumulator.
   2. Machine language program is executed from the starting address represented by the expression.
   3. If there is a carry when returns, the character string whose size is indicated by the accumulator is transferred to the variable from the address represented by the Xreg.

NOTE: It will result in ERROR 7 for a two-nonnumeric variable whose variable has not been defined.

5-1-6. CSAVE M

Format

CSAVE M ┌ −1 ┐ ┌ "file name"; ┐ expression-1, expression-2 ┌ , expression 3 ┐

① CSAVE M expression-1, expression-2, expression-3

Data residing from the address represented by the expression-1 to the address represented by the expression-2 is recorded on the tape as the machine language program.

When the expression-3 is given, execution will automatically take place from the address represented by the expression-3 upon loading of the program from the tape.

File name will also be recorded on the tape, when there is a file name.

② CSAVE M-1

Tape control will be set to the REMOTE-1 side.

5-1-7. CLOAD M

Format

CLOAD M ┌ −1 ┐ ┌ "file name"; ┐ expression ┐

① CLOAD M

The machine language program recorded on the tape is loaded into the same memory area as when recorded.

When the expression is given, program load will take place from the address represented by the expression.

If there have been the expression-3 during data save, program execution will automatically take place from the address represented by the expression-3 upon completion of program load. However, automatic execution will not start if the expression is given.

② CLOAD M-1

Tape control will be set to the REMOTE-1 side.

promsoft.com

5-2. Internal code chart

With the PC-1500 system, BASIC command is converted into internal code of two bytes to be processed by the PC-1500 system.
Shown next is the list of commands with corresponding internal codes.

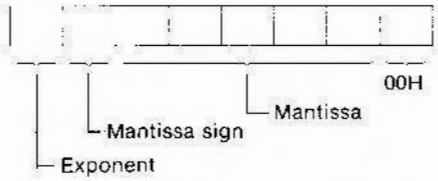
COMMAND	INTERNAL CODE
ABS	F170H
ACS	F174H
AND	F150H
AREAD	F180H
ARUN	F181H
ASC	F160H
ASN	F173H
ATN	F175H
BEEP	F182H
BREAK	F0B3H
CALL	F18AH
CHAIN	F0B2H
CHR$	F163H
CLEAR	F187H
CLOAD	F089H
CLS	F088H
COM$	E858H
CONSOLE	F0B1H
CONT	F183H
COLOR	F0B5H
COS	F17EH
CSAVE	F095H
CSIZE	E680H
CURSOR	F084H
DATA	F18DH
DEG	F165H
DEGREE	F18CH
DEV$	E857H
DIM	F18BH
DMS	F166H
DTE	E884H
END	F18EH
ERL	F053H
ERN	F052H
ERROR	F1B4H
EXP	F178H
FEED	F0B0H
FOR	F1A5H
GCURSOR	F093H
GLCURSOR	E682H
GOSUB	F194H
GOTO	F192H
GPRINT	F09FH
GRAD	F186H
GRAPH	E681H
IF	F196H
INKEY$	F15CH
INPUT	F091H
INSTAT	E859H
INT	F171H
LCURSOR	E683H
LEFT$	F17AH
LEN	F164H
LET	F198H
LF	F0B6H
LINE	F0B7H
LIST	F090H
LLIST	F0B8H
LN	F176H
LOCK	F1B5H
LOG	F177H
LPRINT	F0B9H
MEM	F158H
MERGE	F08FH
MID$	F17BH
NEW	F19BH
NEXT	F19AH
NOT	F16DH
OFF	F19EH
ON	F19CH
OPN	F19DH
OR	F151H
OUTSTAT	E880H
PAUSE	F1A2H
PEEK	F16FH
PEEK#	F16EH
PI	F15DH
POINT	F168H
POKE	F1A1H
POKE#	F1A0H
PRINT	F097H
RADIAN	F1AAH
RANDOM	F1A8H
READ	F1A6H
REM	F1ABH
RESTORE	F1A7H
RETURN	F199H
RIGHT$	F172H
RINKEY$	E85AH
RLINE	F0BAH
RMT	E7A9H
RND	F17CH
ROTATE	E685H
RUN	F1A4H
SETCOM	E882H
SETDEV	E886H
SGN	F179H
SIN	F17DH
SORGN	E684H
SPACE$	F061H
SQR	F16BH
STATUS	F167H
STEP	F1ADH
STOP	F1ACH
STR$	F161H
TAB	F0BBH
TAN	F17FH
TERMINAL	E883H
TEST	F0BCH
TEXT	E686H
THEN	F1AEH
TIME	F15BH
TO	F1B1H
TRANSMIT	E885H
TROFF	F1B0H
TRON	F1AFH
UNLOCK	F1B6H
USING	F085H
VAL	F162H
WAIT	F1B3H
ZONE	F0B4H

promsoft.com

5-3. Expression of variable and program

5-3-1. Expression of decimal number

Decimal number consists of eight bytes which are used to represent number within a range of $-9.999999999 \times 10^{99}$ to $+9.999999999 \times 10^{99}$ and are composed of the exponent part, mantissa sign and mantissa part.



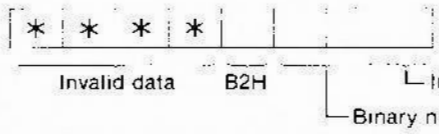
Exponent part: expressed in a binary number. Negative number is represented by complement.
Mantissa sign: 00H (Positive) 80H (Negative)

(Example when numeric is stored from 7A00H to 7A07H)

7A00H							7A07H	
03H	00H	15H	00H	00H	00H	00H	00H	1500
00H	00H	12H	34H	56H	00H	00H	00H	1.23456
FDH	00H	12H	34H	56H	78H	90H	00H	0.00123456789012
08H	80H	12H	34H	00H	00H	00H	00H	-1.234×10^{6}

5-3-2. Expression of binary number

Binary number consists of eight bytes, but five bytes are not used. It expresses a binary number within an integer range of −32768 thru +32767.



Negative number is expessed by a complement.

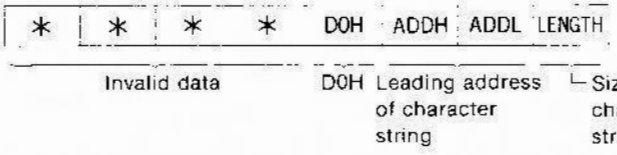
(Example when numeric is stored from 7A00H to 7A07H)

7A00H							7A07H	
*	*	*	*	B2H	05H	DCH	*	1500
*	*	*	*	B2H	FFH	FBH	*	−5
*	*	*	*	B2H	7FH	FFH	*	32767
*	*	*	*	B2H	80H	00H	*	−32768

NOTE: Noted with an asterisk (*) indicates invalid data.

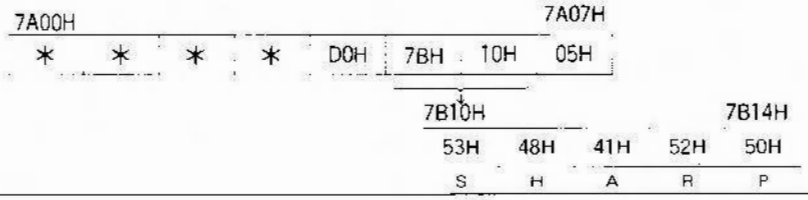
5-3-3. Expression of character string

Character string information is composed of eight bytes (with four bytes of valid data) and it resides in the address contained in the character string information.



Size of character string: 01H thru 50H
Leading address of character string: 0000H thru FFFFH

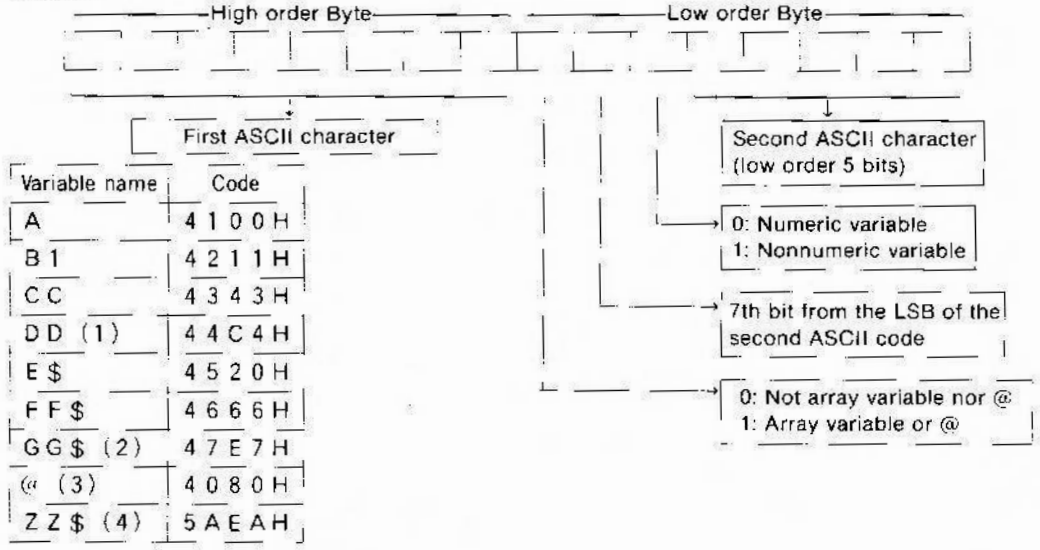
Example that the character string information resides in the arithmetic register and the character string "SHARP" resides in the string buffer.



promsoft.com

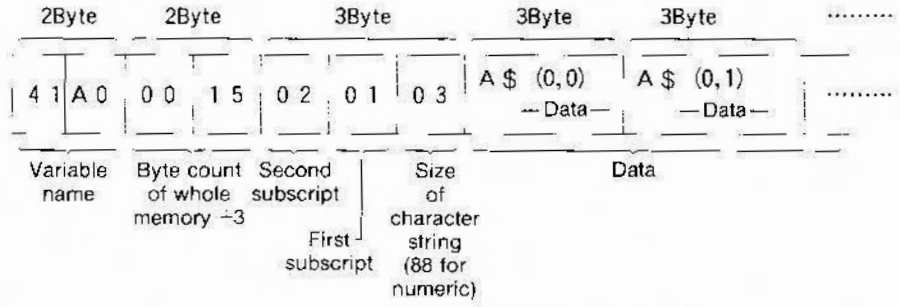
5-3-4. Structure of variable name

Variable name field is made of two bytes which consists of variable name composed of ASCII character, distinction of numeric and nonnumeric variable, and presence of array assignment.



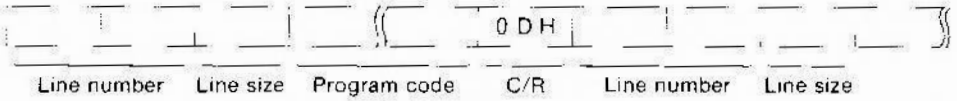
Variable name	Code
A	4100H
B1	4211H
CC	4343H
DD (1)	44C4H
E$	4520H
FF$	4666H
GG$ (2)	47E7H
@ (3)	4080H
ZZ$ (4)	5AEAH

(EX) When array declared to "A$(1,2) * 3"



5-3-5. Structure of program

Each of program lines is composed of the line number, line size, program code, and end code.




```
10 PRINT A
20 END
```

The above programmed lines will be as follows (without (module):

Address	Data	
40C5H	00H	10
40C6H	0AH	
40C7H	04H	……Line size
40C8H	F0H	PRINT
40C9H	97H	
40CAH	41H	……A
40CBH	0DH	……C/R
40CCH	00H	20
40CDH	14H	
40CEH	03H	……Line size
40CFH	F1H	END
40D0H	8AH	
40D1H	0DH	……C/R
40D2H	FFH	……Code to indicate end of BASIC program.

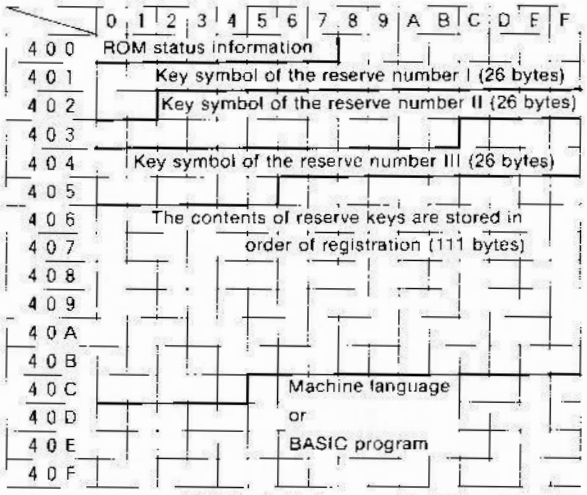
promsoft.com

5-3-6. Structure of reserve area

① Leading address of the reserve area

System configuration	Leading address
PC–1500 ONLY	4000H
PC–1500+CE–151	4000H
PC–1500+ CE–155	3800H
PC–1500–CE–159	2000H

② Reserve memory configuration (when only the PC-1500 is used)



NOTE: When NEW [ENTER] is executed in the RESERVE mode, address area from 4008H to 40C4H is filled up with "00H".

③ Reserve key code

Key \ RESERVE NO.	I	II	III
F1	01H	11H	09H
F2	02H	12H	0AH
F3	03H	13H	0BH
F4	04H	14H	0CH
F5	05H	15H	0DH
F6	06H	16H	0EH

- Reserve key contents are stored following to the key code.
- Reserve programs are stored in order of registration. In the case of re-registration, the previous program is deleted and the new program is added following to it.

④ ROM status information

ADDRESS	DESCRIPTION
4000H	55H
4001H	High order one byte of the ROM top address
4002H	High order one byte of the top address of the BASIC program, assuming the ROM top address to be "0000H".
4003H	Low order one byte of the top address of the BASIC program, assuming the ROM top address to be "0000H"
4004H	Write the following code according to the ROM size 1KB: "04H" 2KB: "08H" 4KB: "10H" 8KB: "20H" 16KB: "40H"
4005H	Undefined
4006H	Undefined
4007H	To make the program confidential, write "00H" To make the program not confidential, write "FFH"

promsoft.com

(An example of registration)

Key symbol

Reserve No. I	P	R	T		I	N	P		G	T	O		G	S	B		H	E	T
Reserve No. II	S	I	N		C	O	S		T	A	N								
Reserve No. III	R	U	N		G	T	O												

Reserve contents

Registration order		Key	Registered contents
1	I	F 1	PRINT
2	I	F 3	GOTO
3	I	F 2	INPUT
4	I	F 4	GOSUB
5	I	F 5	RETURN
6	II	F 1	SIN
7	II	F 3	TAN
8	III	F 2	GOTO(α
9	III	F 1	RUN(α
10	II	F 2	COS

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4 0 0										20	50	52	54	20	49	4E	50
4 0 1	20	47	4F	54	20	47	53	42	20	52	45	54	20	00	00	00
4 0 2	00	00	20	53	49	4E	20	43	4F	53	20	54	41	4E	20	00
4 0 3	00	00	00	00	00	00	00	00	00	00	00	20	52	55	4E	
4 0 4	20	47	54	4F	20	00	00	00	00	00	00	00	00	00	00	
4 0 5	00	00	00	00	00	00	01	F0	97	03	F1	92	02	F0	91	04
4 0 6	F1	94	05	F1	99	11	F1	70	13	F1	7F	0A	F1	92	40	09
4 0 7	F1	A4	40	12	F1	7E	00	00	00	00	00	00	00	00	00	00
4 0 8	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
4 0 9	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
4 0 A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
4 0 B	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
4 0 C	00	00	00	00	00											
4 0 D																
4 0 E																
4 0 F																

Characters are stored in form of character code and commands in form of internal code. "00H" is for the end code of data.

promsoft.com

5-4. System subroutines

System subroutines used by the machine language program will be introduced next. However, care must be taken for printer related entry address for it differs depending on the ROM version of the CE-150. The version number will be indicated in the address A800H.

Version 0	Address A800H is 44H.
Version 1	Address A800H is BEH.

① Character function

Combination of character	D925H	VAL		D9D7H
CHR$	D9B1H	LEN, ASC		D9DDH
STR$	D9CFH	RIGHT$, MID$, LEFT$		D9F3H

② Arithmetic operations

Substract	EFB6H	10^n	F1D4H	DEG	F531H
Add	EFBAH	COS	F391H	DMS	F564H
Multiply	F01AH	TAN	F39EH	ABS	F597H
Divide	F084H	SIN	F3A2H	SGN	F59DH
LN	F161H	ACS	F492H	INT	F5BEH
LOG	F165H	ATN	F496H	Power raise	F89CH
EXP	F1CBH	ASN	F49AH		

③ Compare

Numerical comparison	D0D2H	Character string comparison	D0F9H

④ Search

Line number search	D2EAH	Variable search	D461H
KEY scan (I)	E42CH	KEY scan (II)	E243H

⑤ Display

Auto-power-off	E33FH	One character display	ED57H
Program display	E8CAH	"n" character display	ED3BH
Graphic display	EDEFH	Cursor move after one character display	ED4DH
Hexadecimal (2 bytes → 1 byte)	ED95H	Cursor move after "n" character display	ED00H

promsoft.com

⑥ Printer related

Color designation	A519H(A4F7H)	Pen up/down	AAE3H(AABDH)
Print	A781H(A75BH)	Motor drive	A8DDH(A8B7H)
Linefeed	A9F1H(A9CBH)	Motor off	A769H(A747H)
Paper feed	AA04H(A9DEH)	Get GRAPHIC mode ready	ABEFH(ABC6H)
Get TEXT mode ready	ACBBH(AC8FH)		

(Figures in parentheses indicate version 0.)

⑦ Cassette tape

Remote on	BF11H	Header input/output	BCE8H
Remote off	BF43H	CMT I/O control	.BBF5H
One character save	BDCCH	Create header	BBD6H
One character load	BDF0H	Transfer file	BD3CH

promsoft.com

5-4-1. Character functions

■ Combination of character string

Combination of character string-1 and character string-2

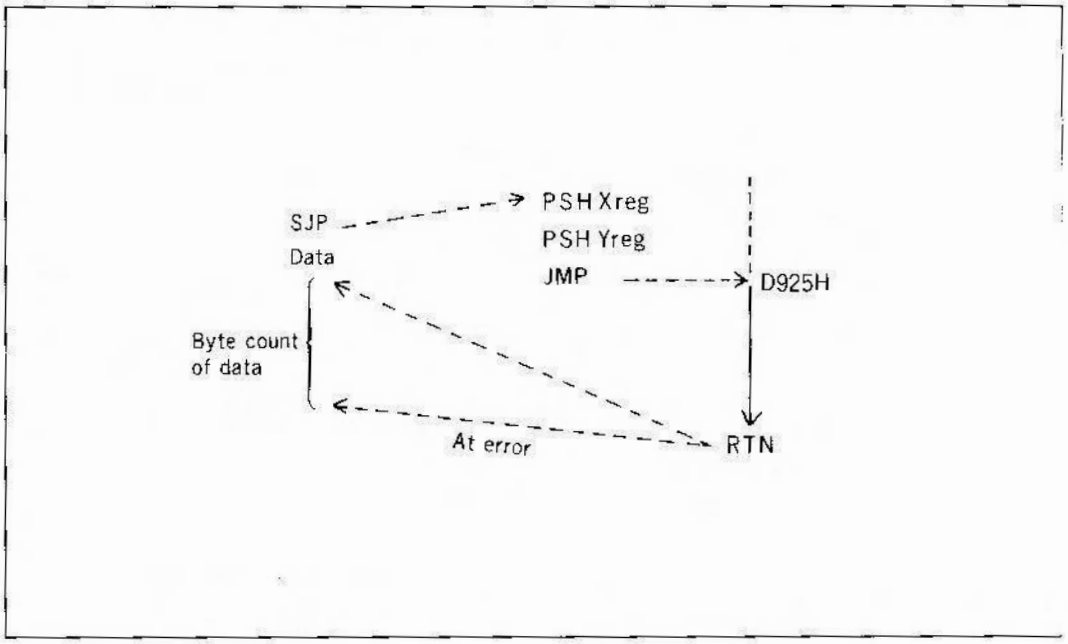
① ENTRY PREPARATION

- Preparation of character strings
  Character string information of the character string-1 is stored in the arithmetic register (7A00H~7A07H).
  Character string information of the character string-2 is stored in the arithmetic register (7A10H~7A17H).
- 10H is stored in the string buffer pointer (7894H).

② ENTRY ADDRESS

D925H

- Subroutine is called in the following format:
  SJP (Address where PSH Xreg is written)
  DB DATA (Error return address)
  PSH Xreg
  PSH Yreg } Write in other addresses.
  JMP D925



③ EXIT STATE

- When no error
  Returns to the address next to the data. Resultant character string information will then be stored in the arithmetic register X (7A00H~7A07H) and the combined character strings will be stored in the string buffer (7B10H~).
- When error
  After storing the error code in the UH register, it returns to the data stored address plus one.

promsoft.com

NOTE: For details of character string information, refer to the paragraph discussing the structure of variable and program.

■ CHR$

① ENTRY PREPARATION

- Preparation of numeric value
  An integer of 0 thru 255 is stored in the arithmetic register (7A00H~7A07H) in decimal or binary figure.
- 10H is stored in the string buffer pointer (7894H).

② ENTRY ADDRESS

D9B1H

③ EXIT STATE

- When no error
  UH = 00H

Address	Contents
7A04H	C1H
7A05H	7BH
7A06H	10H
7A07H	00H or 01H
7B10H	ASCII code

When ASCII code is 00H, the contents of 7A07H become 00H or 01H when other than 00H.

- When error
  UH ≠ 00H (Error code is stored in UH)

■ STR$

① ENTRY PREPARATION

- Number to be converted is stored in the arithmetic register (7A00H~7A07H) in decimal or binary figure.
- 10H is stored in the string buffer pointer (7894H).

② ENTRY ADDRESS

D9CFH

promsoft.com

③ EXIT STATE

- When no error

  UH = 00H

Address	Data
7A04H	D0H
7A05H	Leading address of character string (high order one byte)
7A06H	Leading address of character string (low order one byte)
7A07H	Size of character string

For an actual character string, the string buffer (7B10H~7B5FH) can be used.

- When error

  UH ≠ 00H (Error code will be stored in UH.)

■ VAL

① ENTRY PREPARATION

- Character string information to be converted is stored in the arithmetic register X (7A00H~7A07H) in character string format.

② ENTRY ADDRESS

D9D7H

③ EXIT STATE

- When no error

  UH = 00H

  Result is stored in arithmetic register X (7A00H~7A07H) in decimal figure.

- When error

  UH ≠ 00H (Error code will be stored in UH.)

■ ASC, LEN

① ENTRY PREPARATION

- Character string information to be converted is stored in the arithmetic register X (7A00H~7A07H).

Address	Data
7A04H	D0H
7A05H	Leading address of character string (high order one byte)
7A06H	Leading address of character string (low order one byte)
7A07H	Size of character string

promsoft.com

- Setup of function

Function	YL
ASC	60H
LEN	64H

② ENTRY ADDRESS

D9DDH

③ EXIT STATE

- When no error

  UH= 00H

  Result is stored in arithmetic reister X (7A00H ~ 7A07H) in decimal figure.

- When error

  UH≠ 00H (UH=error code)

■ RIGHT$ (Character string, numeric value-1)

LEFT$ (Character string, numeric value-1)

MID$ (Character string, numeric value-2, numeric value-1)

① ENTRY PREPARATION

- Setup of function

Function	YL
RIGHTS	02H
LEFTS	7AH
MIDS	7BH

- Existence of availability for 8 bytes in the BASIC stack (7A38H~7AFFH) is checked.
  - In the case of RIGHT$, LEFT$

    (Data of 7890H) < (data of 7891H) − 8
  - In the case of MID$

    (Data of 7890H) < (data of 7891H) − 16

  Since re-write of 7890H and 7891H is not permitted, it must be avoided to call this subroutine, unless the above condition is satisfied.

promsoft.com

- Change of the data pointer (7892H)

Function	Data
RIGHT$ LEFT$	(data of 7890H) + 8
MID$	(data of 7890H) + 16

- Preparation of character string

Address	Data
(Data of 7890H) + 4	D0H
(Data of 7890H) + 5	Leading address of character string (high order one byte)
(Data of 7890H) + 6	Leading address of character string (low order one byte)
(Data of 7890H) + 7	Size of character string

NOTE: High order byte of address is 7AH.

For actual character string, the string, the string buffer (7B10H~7B5FH) can be used.

- Preparation of numerical data

Numeric-1 is stored in 7A00H~7A07H in decimal or binary figure.

(In the case of MID$, the numeric-2 is stored from the address of "data of 7890H" plus eight to the address "data of 7890H" plus fifteen.)

Stored in the above address in a decimal or binary format.

② ENTRY ADDRESS

D9F3H

③ EXIT STATE

- When no error

UH = 00H

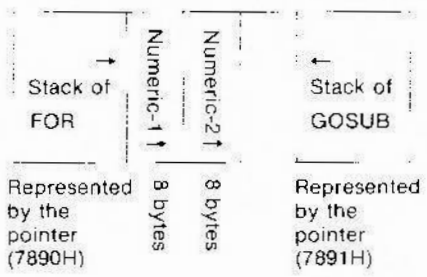
Address	Data
7A04H	D0H
7A05H	Leading address of character string (high order one byte)
7A06H	Leading address of character string (low order one byte)
7A07H	Size of character string

For an actual character string, the string buffer (7B10H~7B5FH) can be used.

- When error

UH ≠ 00H (UH is substituted with error code.)

NOTE:



promsoft.com

5-4-2. Arithmetic subroutines

① ENTRY PREPARATION

- Preparation of numeric value
  Numeric value should be prepared in the arithmetic register X (7A00H~7A07H) and the arithmetic register Y (7A10H~7A17H).
  (In the case of a single variable, only the arithmetic register needs to be prepared.

② ENTRY ADDRESS

		Operation	Address
Two-variable function	Add	X + Y → X	EFBAH
	Subtract	X - Y → X	EFB6H
	Multiphy	X*Y → X	F01AH
	Divide	X / Y → X	F084H
	Power raise	X^Y → X	F89CH
Single-variable function	Square root	SQR X→ X	F0E9H
	Logarithm	LN X → X	F161H
		LOG X → X	F165H
		EXP X→ X	F1CBH
		10^X → X	F1D4H
	Trigonometric function	SIN X → X	F3A2H
		COS X→ X	F391H
		TAN X→ X	F39EH
	Inverse trigonometric function	ASN X→ X	F49AH
		ACS X→ X	F492H
		ATN X→ X	F496H
	Degree to minute and second conversion	DEG X→ X	F531H
		DMS X→ X	F564H
	Absolute value	ABS → X	F597H
	Sign	SGN X→ X	F59DH
	Conversion into an integer number	INT X→ X	F5BEH

③ EXIT STATE

- Result will be stored in the arithmetic register X (7A00H~7A07H).

promsoft.com

5-4-3. Comparison

■ Comparison of numeric

(Numeric 1) ◯ (numeric 2)

◯ Indicates the operand.

① ENTRY PREPARATION

- Setup of operand

Operand	Accumulator data
<>	00H
<	01H
>	02H
–	04H
<–	05H
>–	06H

- The numeric is stored in the arithmetic register in a format of decimal figure.

Numeric 1	Arithmetic register X (7A00H~7A07H)
Numeric 2	Arithmetic register X (7A10H~7107H)

② ENTRY ADDRESS

D0D2H

③ EXIT STATE

- When the operand is established

  The flag Z is reset to "0" and the arithmetic register turns to "1".

7A00H							7A07H
00H	00H	10H	00H	00H	00H	00H	00H

- When the operand is not established

  The flag Z is set to "1" and the arithmetic register turns to "0".

7A00H							7A07H
00H	00H	00H	00H	00H	00H	00H	00H

■ Comparison of character string

(Character string-1) ◯ (character string-2)

◯ Indicates the operand.

① ENTRY PREPARATION

- Designation of the operand

Operand	Accumulator data
>	00H
<	01H
>	02H
–	04H

promsoft.com

• Preparation of character string

Contents \ Address		Character string-1	Character string-2
D0H		7A04H	7A14H
Leading address of the character string	(high order)	7A05H	7A15H
	(low order)	7A06H	7A16H
Size of the character string		7A07H	7A17H

For the address where the character string is stored, the string buffer (7B10H~7B5FH) can be used.

• 10H is stored in the string buffer pointer (7894H).

② ENTRY ADDRESS

D0F9H

③ EXIT STATE

• When the condition for the operand is established (Z=0)

7A00H							7A07H
00H	00H	10H	00H	00H	00H	00H	00H

• When the condition for the operand is not established (Z=1)

7A00H							7A07H
00H	00H	00H	00H	00H	00H	00H	00H

5-4-4. Search

■ Variable address search

① ENTRY PREPARATION

• Designation of variable name
  The variable name is stored in the Ureg.

• Whether array is one dimension or two dimension is stored in parameter F F (788CH).
  one dimension: 01H
  two dimension: 02H

• Subscript is stored in the arithmetic register X (7A00H~7A07H).
  When one dimension array: the first subscript
  When two dimension array: the second subscript

• Subscript is stored in the arithmetic register Y (7A10H~7A17H).
  When one dimension array: No need
  When two dimension array: the first subscript

promsoft.com

② **ENTRY ADDRESS**

D461H

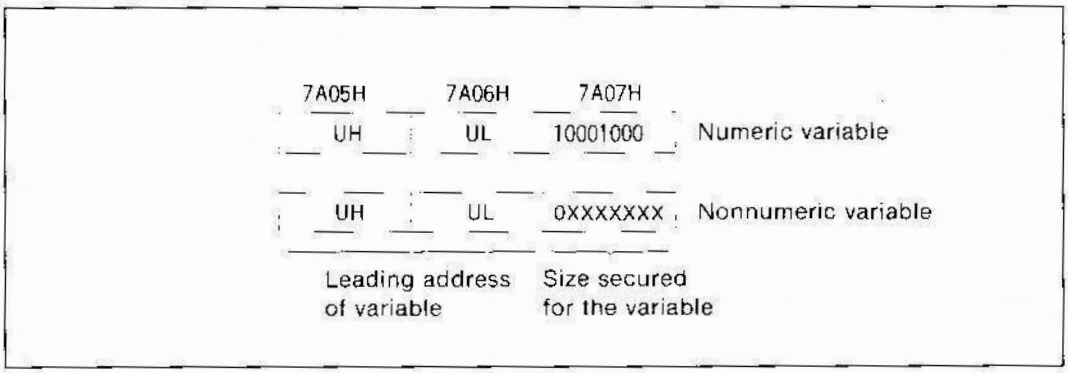
- Subroutine must be called in the following format:

```
SJP   D461H
DB    FAH
DB    DATA ......(Decides the address to return when an error is met.)
```

③ **EXIT STATE**

- When no error

  The leading address of the variable is stored in the Ureg, and the variable name and the data size is stored in the arithmetic register X. Then, it returns to the address that follows DB DATA.



- When error

  Returns to the address that data plus one is added to the "data written address" after storing the error code is stored in UH.

■ **Key scan (I)**

① **ENTRY PREPARATION**

- None

② **ENTRY ADDRESS**

E42CH

③ **EXIT STATE**

- The key code of the key that depressed at that time is stored in the accumulator. If there is no key depression, "00H" will be stored in the accumulator.

■ **Key scan (II)**

① **ENTRY PREPARATION**

- None

② **ENTRY ADDRESS**

E243H

③ **EXIT STATE**

- The key code of newly depressed key is stored in the accumulator. Although it does not return until a next key is depressed, it may end in auto-power-off unless a key is depressed within seven minutes. (However, the previous state resumes with depression of the [BREAK ON] key.

  When the [BREAK ON] key is pushed, PB7 of the PC-1500 IF register (#F00BH) will be set.

  NOTE: So long as PB7 of the address #F00BH is set, "0EH" will be stored in the accumulator.
  PB7 of the address #F00BH will be reset when "ANI #F00BH, FDH" is executed.

promsoft.com

■ **Search of program line**

① **ENTRY PREPARATION**

- Line number is stored in the Ureg (0001H ≤ Ureg ≤ FFFFH).

② **ENTRY ADDRESS**

D2EAH

- Subroutine is called in the following format:

```
SJP    D2EAH
DB     DATA
```

③ **EXIT STATE**

- When the specified line is found

  Returns to the address that follows DB DATA after storing the data in SEARCH (78A6H~78ABH).

78A6H	Leading address of the line
78A7H	Leading address of the line
78A8H	Line number
78A9H	Line number
78AAH	Leading address of the line plus 3
78ABH	Leading address of the line plus 3

- When the specified line is not found

  Returns to the "data written address" plus one after storing the error code in UH.

Carry	Condition
0	No specified line found, and search is made to the last of the program.
1	Found the line larger than the specified line.

5-4-5. Display

■ **One character display**

① **ENTRY PREPARATION**

- Display start position is stored in the cursor pointer (7875H). Cursor will be effective within a range of 00H to 98H.
- Code of display character is stored in the accumulator.

② **ENTRY ADDRESS**

ED57H

promsoft.com

③ EXIT STATE

- No change takes place in the cursor pointer.
- Change will be met in carry depending on the cursor called.

Cursor	Carry
00H~95H	0
96H~9BH	1

- One character will be displayed on LCD.

■ Moving cursor after displaying one character

① ENTRY PREPARATION

- Display start position is stored in the cursor pointer (7875H). Cursor will be effective within a range of 00H to 98H.
- Code of display character is stored in the accumulator.

② ENTRY ADDRESS

ED4DH

③ Change will be met in the cursor pointer.

Cursor position when called	Cursor position after return
00H~95H	Previous cursor position −6
96H~9BH	00H

- One character will be displayed on LCD.

■ Auto-power-off

① ENTRY PREPARATION

- None

② ENTRY ADDRESS

E33FH

③ EXIT STATE

- When power is turned on once after auto-power-off, no printer initialization takes place.
- When power is turned on by means of the [ON] key once after auto-power-off, it needs to push any key, except [SHIFT], [SML], and [OFF] key, as subroutine is in execution.

Note: An example of manual operation

Example of manual operation

Key operation	Display
CALL & E33F	CALL & E33F
[ENTER]	(OFF state)
[ON]	BUSY CALL & E33F
Any key except [SHIFT], [SML], and [OFF]	>

promsoft.com

■ **"n" character display**

① **ENTRY PREPARATION**

- Display start position is stored in the cursor pointer (7875H).
- Size of the character string is stored in the accumulator (01H ≤ ACC ≤ 1AH).
- Leading address of the character string is stored in the Ureg (0000H ≤ Ureg ≤ FFFFH).

② **ENTRY ADDRESS**

ED00H

③ **EXIT STATE**

- Change will be met in carry.

Carry	Cursor position
0	Next to the rightmost position of the character string on display.
1	Indicates the last character on the display, in case display should end at 26th digit or exceed 26th digit.

NOTE: When display exceeds 156 dots, the contents after this dot position will be ignored.

■ **Output of "n" characters from the top of LCD**

"n" characters will be displayed unconditionally from the left side of LCD.

① **ENTRY PREPARATION**

- The leading address of the character string is stored in the Ureg (0000H ≤ Ureg ≤ FFFFH).
- Size of the character string is stored in XL (01H ≤ Xreg ≤ 1AH).

② **ENTRY ADDRESS**

ED3BH

③ **EXIT STATE**

- Change will be met in carry.

Carry	Contents
0	Character string within 25 characters.
1	Character string more than 26 characters.

■ **Hexadecimal (2 byte → 1 byte)**

ASCII code of two bytes is assumed to be a hexidecimal figure and is changed into numeric of one byte.

① **ENTRY PREPARATION**

- Leading address of the ASCII code is stored in the Xreg.

promsoft.com

② **ENTRY ADDRESS**

ED95H

③ **EXIT STATE**

- In Xreg is stored the value of previous Xreg added with 02H.
- The one byte of the derived data is stored in the accumulator.

■ Graphic display

① **ENTRY PREPARATION**

- Output pattern is stored in the accumulator.

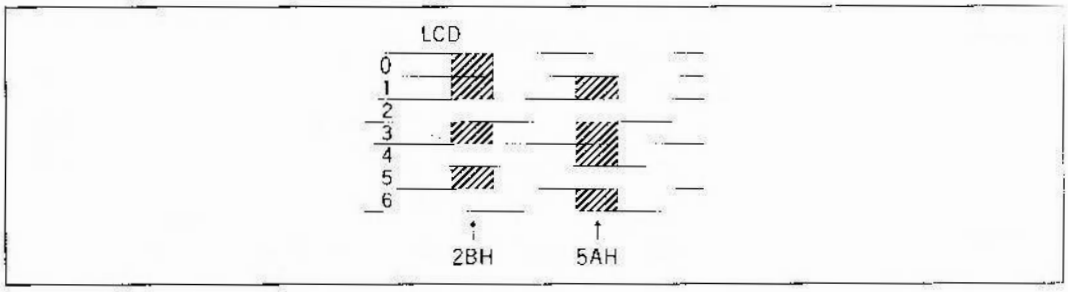
② **ENTRY ADDRESS**

EDEFH

③ **EXIT STATE**

- Contents of Xreg, Ureg, and accumulator become irrelevant.
- No change takes place in the cursor position.

(Reference)



■ Program display

① **ENTRY PREPARATION**

A. In the case of numeric

Data is stored in the arithmetic register (7A00H~7A07H).

B. In the case of character string or program

Data is stored in the input buffer (7BB0H~7BFFH).

Cursor position is stored in Yreg. (In the case of program, 7BH is stored in YH.)

C. Parameter F F (7880H) is set.

Data of 7880H	Display contents
40H	Character string is displayed according to Yreg.
00H	Character string is displayed from the top, regardless of Yreg contents.
20H	Numerical value of the arithmetic register X is displayed.
10H	Line number, space, and program are displayed from the top.
14H	Line number, colon, and program are displayed from the top.
50H	Line number and space are displayed in a middle of program according to Yreg.
54H	Line number and colon are displayed in a middle of program according to Yreg.

promsoft.com

② **ENTRY ADDRESS**
E8CAH

③ **EXIT STATE**
- Displayed on LCD according to direction.

NOTES: 1. In the case of an internal code, the cursor must show the code position on the lower side of 2 bytes.
2. Numeric is displayed in right justified manner and character string in left justified manner.
3. Numeric has no concern with USING.
4. Only 26 characters will be handled for a character string exceeding 26 characters.
5. 0DH is required at the end of character string for the input buffer.

5-4-6. Printer

■Color designation.

① **ENTRY PREPARATION**
- The specified color code (0~3) is stored in the UL.

② **ENTRY ADDRESS**
A519H (A4F7H for the version 0.)

③ **EXIT STATE**
- Return with the motor on.
(Motor off routine must be called after EXIT.)

```
(EX)  LDI  UL, color code
      SJP  A519H
      SJP  A769H
      RTN
```

■ Motor drive

① **ENTRY PREPARATION**
- Set the address pointer in the Xreg to indicate the value of relative movement, then store the value of relative movement after that address pointer.

Address shown by Xreg	ΔXH: High order 8 bits of the relative movement value towards the X direction.
Address shown by Xreg+1	ΔYH: High order 8 bits of the relative movement value towards the Y direction.
Address shown by Xreg+2	ΔXL: Low order 8 bits of the relative movement value towards the X direction.
Address shown by Xreg+2	ΔYL: Low order 8 bits of the relative movement value towards the Y direction.

② **ENTRY ADDRESS**
A8DDH (A8B7H for the version 0.)

promsoft.com

③ EXIT STATE

- The motor is on. (Motor off routine must be called after EXIT.)

NOTE: Negative movement must be indicated by a complement.
Relative movement must be within a range of −32768 thru 32767.

■ Motor off

① ENTRY PREPARATION

- None

② ENTRY ADDRESS

A769H (A747H for the version 0.)

③ EXIT STATE

- Motor off

■ Pen up/down

① ENTRY PREPARATION

- Either 00H or FFH is stored in the PEN UP/DOWN F. F (79E9H).

UP	00H
DOWN	FFH

② ENTRY ADDRESS

AAE3H (AABDH for the version 0.)

③ EXIT STATE

- Pen ascends or descends then solenoid turn inactive.

■ GRAPHIC mode preapration

① ENTRY PREPARATION

- None

② ENTRY ADDRESS

ABEFH (ABC6H for the version 0.)

③ EXIT STATE

Line (79EAH)	00H
Rotate (79F2H)	00H
User counter (79E0H~79E3H)	00H

NOTES:
1. Because G/T (79F0H) is not changed, it needs to store FFH in G/T (79F0H) after return.
2. CSIZE does not change.
3. Scissoring counter does not change.

promsoft.com

■ TEXT mode preparation

① ENTRY PREPARATION

- None

② ENTRY ADDRESS

ACBBH (ACBFH for the version 0.)

③ EXIT STATE

Line (79EAH)	00H
G/T (79F0H)	00H
Rotate (79F2H)	00H

NOTES:

1. When the scissoring counter YH and YL (79E4H~79E5H) exceeds 0200H, it makes 79E4H turned to 01H and 79E5H to FFH. When (79E4H~79E5H) is below 01FFH, it causes no change in 79E4H and 79E5H.
2. CSIZE does not change.

■ Paper feed

① ENTRY PREPARATION

- Paper feed count is stored in the address represented by the Xreg. (Negative number is indicated by a complement.)

② ENTRY ADDRESS

AA04H (A9DEH for the version 0.)

③ EXIT STATE

- The motor stays on. (Motor off routine must be called after EXIT.)

NOTE: Paper feed count may change after setting the CSIZE.

■ Linefeed

① ENTRY PREPARATION

- Line kind (79EAH) must be reset to 0.
- Address area that can be destructed should be stored in Xreg. (X−10 ~ X+1 will be destructed.)

② ENTRY ADDRESS

A9F1H (A9CBH for the version 0.)

③ EXIT STATE

- The motor stays on. (Motor off routine must be called after EXIT.)

NOTES:

1. When the contents of Xreg is 7A22H, data in 7A18H thru 7A23H will be destructed.
2. Paper feed count changes after CSIZE.

promsoft.com

■ **Print**

① **ENTRY PREPARATION**

- 00H is stored in LINE TYPE (79EAH).
- Print code storing address must be stored in the Xreg.
  (When 7A20H is stored in Xreg, it affects the data in 7A08H thru 7A37H. So, care must be exerted not to destruct address area of the Xreg represented address minus ten and plus one.
- Print code is stored in the address represented by the Xreg.

② **ENTRY ADDRESS**

A781H (A75BH for the version 0.)

③ **EXIT STATE**

- Stays on.
- The value of 6×CSIZE will be added to the contents of CURSOR (79E6H). (Motor off routine must be called after EXIT.)

5-4-7. Cassette tape

■ **REMOTE ON**

① **ENTRY PREPARATION**

- Sets the PARAMETER F/F (7879H).

Contents of 7879H	Remote	CMT input port
0XX0XXXX	0	Close (CMT output)
0XX1XXXX	1	
1XX0XXXX	0	Open (CMT input)
1XX1XXXX	1	

② **ENTRY ADDRESS**

BF11H

③ **EXIT STATE**

- REMOTE 0 or 1 turns active according to the contents of the PARAMETER F/F (7879H).

NOTE: This system subroutine drives the relay in the CE-150, regardless of the REMOTE switch position.

■ **REMOTE OFF**

① **ENTRY PREPARATION**

- RMT/BEEP (786BH) must be set to control REMOTE 1.
  (Preparation is not required in the case of REMOTE 0.

Contents of 786BH	Remote
0XXXXXXX	OFF
1XXXXXXX	ON

promsoft.com

② **ENTRY ADDRESS**

BF43H

③ **EXIT STATE**

- REMOTE 0 is off.
- REMOTE 1 will be in accordance with RMT/BEEP (786BH).

NOTE: This system subroutine drives the relay in the CE-150, regardless of the REMOTE switch position.

■ **Save of one character**

① **ENTRY PREPARATION**

- Data is stored in the accumulator.

② **ENTRY ADDRESS**

BDCCH

③ **EXIT STATE**

- None.

NOTE: This system call must be executed after saving of the synchronizing header.

■ **Load of one character**

① **ENTRY PREPARATION**

- None

② **ENTRY ADDRESS**

BDF0H

③ **EXIT STATE**

- Data has been sent in the accumulator.
- Change takes place in carry.

Carry	Condition
0	End of data read.
1	Depression of the [BREAK] key.

■ **Creation of header**

① **ENTRY PREPARATION**

- File mode is set in the accumulator.

File mode	Contents
00H	Machine language
01H	BASIC program
02H	Reserve
04H	Data

Use another code to avoid confusion, as the file mode for other than PC-1500 is used.

- File name is stored in 7B69H~7B78H of the output buffer.

② **ENTRY ADDRESS**

BBD6H

promsoft.com

③ EXIT STATE

- Header is established in 7B60H~7B87H of the output buffer.

Address	Contents
7B60H~7B67H	Synchronizing header
7B68H	File mode
7B69H~7B78H	File name
7B79H~7B87H	All 00H

■ Header input/output

① ENTRY PREPARATION

- Parameter F_F (7879H) is set.

Contents of 7879H	In/Out	Remote
0XX0XXXX	Out	0 side
0XX1XXXX		1 side
1XX0XXXX	In	0 side
1XX1XXXX		1 side

- RMT_BEEP (786BH) is set.

Contents of 786BH	BEEP
XXXXXXX0	OFF
XXXXXXX1	ON

- Header is created in the case of output.

② ENTRY ADDRESS

BCE8H

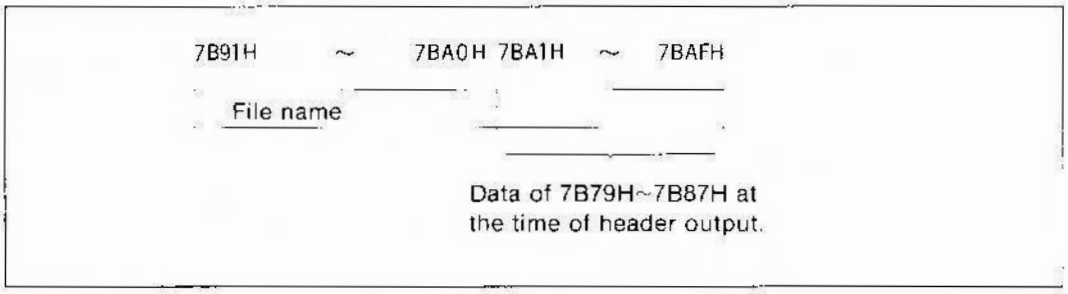
③ EXIT STATE

- Carry changes.

Carry	Condition
0	Broken in a middle.
1	Input is complete.

- In the case of input

  The read data is stored in the output buffer of 7B91H~7BAFH.



- In the case of output

  Stop bit modulation signal stays on.

NOTES: 1. BEEP will be in accordance with 786BH.
2. Paper feed operation stays prohibited for the CE-150.
3. In the case of input, the file name coincident of the header and file mode is displayed on the LCD. In case the file name is specified, it is searched until found.

pronsoft.com

■ **File transfer**

① **ENTRY PREPARATION**

1) Parameter F_F (7879H) is set.

7879H	Contents
00XXXXXX	In the case of load
01XXXXXX	In the case of verification
0XXXXXXX	In the case of save

2) Leading address of input/output data is stored in the Xreg.

3) Byte count of the data minus one is stored in the Ureg.

② **ENTRY ADDRESS**

BD3CH

③ **EXIT STATE**

- In the case of save

(Carry changes)

Carry	Condition
1	Broken in a middle.
0	Input/output is complete.

■ **Termination of CMT I/O control**

① **ENTRY PREPARATION**

- Parameter F_F (7879H) is set.

Contents of 7879H	Data input/output
1XXXXXXX	Input
0XXXXXXX	Output

② **ENTRY ADDRESS**

BBF5H

③ **EXIT STATE**

- Serial port is reset.
- PAPER FEED key of the CE-150 becomes operative.
- Motor drive turns off.
- In the case of load

Change is met in carry, H, and V.

Carry	H	V	State
0	-	-	Load and verification are complete.
1	1	-	In break state.
1	0	1	Error occurrence during verification.
1	0	0	Occurrence of check sum error.

NOTE: The last transfer "address plus one" is stored in the Xreg.

5-4-8. Caution for system subroutine call

1. Printer related system call

Although the error code is put on the display when an error such as low battery occurred during execution of printer related system subroutine, it is not possible to check the error line using the ↑ key.

promsoft.com

promsoft.com

6

Machine language programming examples

*Machine language program discussed is assumed to start from the address 40C5H.

promsoft.com

6-1. Binary to hexadecimal conversion

The binary number stored in the Xreg is converted into hexadecimal equivalent and stored in the fixed nonnumeric variable YS. Binary number within a range of $-32769 < a < 32768$ is applicable.

ADDRESS	MACHINE LANGUAGE		MNEMONIC
40C5	68 77		LDI UH 77H
7	6A E0		LDI UL E0H
9	84		LDA XH
A	BE 40 E0		SJP ①
D	61		SIN U
E	84		LDA XH
F	BE 40 E1		SJP ②
D2	61		SIN U
3	04		LDA XL
4	BE 40 E0		SJP ①
7	61		SIN U
8	04		LDA XL
9	BE 40 E1		SJP ②
C	61		SIN U
D	69 00		ANI U, 00H
F	9A		RTN
E0	F1	①	AEX
1	B9 0F	②	ANI A, 0FH
3	B7 0A		CPI A, 0AH
5	83 03		BCS ③
7	B3 30		ADI A, 30H
9	9A		RTN
A	B3 36	③	ADI A, 30H
C	9A		RTN

promsoft.com

6-2. Display inversion

The current display contents are inverted.

ADDRESS	MACHINE LANGUAGE	MNEMONIC
40C5	68 78	LDI UH, 78H
7	6A 4D	LDI UL 4DH
9	FD 62	DEC UH
B	25	LDA U
C	BD FF	EAI FFH
E	2E	STA U
F	88 06	LOP 06H
D1	6C 77	CPI UH, 77H
3	93 0E	BCS, −0EH
5	9A	RTN

promsoft.com

6-3. Single display dot left shift

The current display contents are shifted to the left by one dot position.

ADDRESS	MACHINE LANGUAGE		MNEMONIC
40C5	FD 88		PSH X
7	FD 98		PSH Y
9	FD A8		PSH U
B	A5 76 00		LDA 7600H
E	F1		AEX
F	B9 0F		ANI A, 0FH
D1	0A		STA XL
2	A5 76 01		LDA 7601H
5	F1		AEX
6	B9 0F		ANI A, 0FH
8	08		STA XH
9	68 78		LDI UH, 78H
B	FD 62	①	DEC UH
D	6A 4D		LDI UL, 4DH
F	66	②	DEC U
E0	65		LIN U
1	1A		STA YL
2	25		LDA U
3	18		STA YH
4	84		LDA XH
5	63		SDE U
6	04		LDA XL
7	2E		STA U
8	FD 18		LDX Y
A	88 0D		LOP ②
C	6C 77		CPI UH, 77H
E	93 15		BCS ①
F0	04		LDA XL
1	F1		AEX
2	AE 77 4E		STA 774EH
5	84		LDA XH
6	F1		AEX
7	AE 77 4F		STA 774FH
A	FD 2A		POP U
C	FD 1A		POP Y
E	FD 0A		POP X
4100	F9		REC
1	9A		RTN

promsoft.com

6-4. Single display dot right shift

The current display contents are shifted to the right by one dot position.

ADDRESS	MACHINE LANGUAGE	MNEMONIC
40C5	FD 88	PSH X
7	FD 98	PSH Y
9	FD A8	PSH U
B	A5 77 4C	LDA 774CH
E	F1	AEX
F	B9 F0	ANI A, F0H
D1	0A	STA XL
2	A5 77 4D	LDA 774DH
5	F1	AEX
6	B9 F0	ANI A F0H
8	08	STA XH
9	68 75	LDI UH, 75H
B	6A FF	LDI UL, FFH
D	64	INC U
E	65	LIN U
F	1A	STA YL
E0	25	LDA U
1	18	STA YH
2	84	LDA XH
3	63	SDE U
4	04	LDA XL
5	61	SIN U
6	FD 18	LDX Y
8	6E 4D	CPI UL, 4DH
A	91 0F	BCR, -0FH
C	6C 77	CPI UH, 77H
E	91 15	BCR, -15H
F0	64	INC U
1	04	LDA XL
2	F1	AEX
3	61	SIN U
4	84	LDA XH
5	F1	AEX
6	2E	STA U
7	FD 2A	POP U
9	FD 1A	POP Y
B	FD 0A	POP X
D	F9	REC
E	9A	RTN

promsoft.com

6-5. Conversion of USING format expressed numerical data into character string

① **ENTRY PREPARATION**

- Numeric data is stored in the fixed numeric variable A in decimal figure and the format is specified by means of the USING statement.
  (Format is within 16 characters and no error is detected during conversion.)

② **ENTRY ADDRESS** 40C5H

③ **EXIT STATE**

- Character string is stored in the fixed nonnumeric variable A$. However, A$ can be anything when in error.

④ **PROGRAM**

ADDRESS	MACHINE LANGUAGE	MNEMONIC
40C5	48 79	LDI XH, 79H
7	4A 00	LDI, 00H
9	58 7A	LDI YH, 7AH
B	BE F7 3F	SJP F73FH
E	B5 01	LDI A, 01H
D0	CD 96	VMJ 96H
2	DF	DEC A
3	2A	STA UL
4	58 78	LDI YH
6	5A C0	LDI YL
8	F5	TIN
9	88 03	LOP 03H
B	14	LDA YL
C	B9 0F	ANI A, 0FH
E	8B 02	BZS, -02H
E0	59 00	ANI (Y), 00H
2	9A	RTN

6-6. Power off that does not activate the printer upon power on

With the following program, the printer will not be activated when power is turned on after power was turned off, the same manner as in the case of OFF to ON.

ADDRESS	MACHINE LANGUAGE	MNEMONIC
40C5	AA 78 4F	LDI S, 784FH
8	BE CF CC	SJP CFCCH
B	BE D0 2B	SJP D02BH
E	B5 3E	LDI A, 3EH
D0	1E	STA Y
1	E9 78 8A EF	ANI 788AH, EFH
5	E9 76 4E FE	ANI 764EH, FEH
9	B5 00	LDI A, 00H
B	AE 78 80	STA 7880H
E	AE 78 9C	STA 789CH
E1	AE 78 9D	STA 789DH
4	BE E8 CA	SJP E8CAH
7	48 CA	LDI XH, CAH
9	4A 92	LDI XL, 92H
B	FD 88	PSH X
D	BA E3 3F	JMP E33FH

bromsoft.com

REFERENCE

promsoft.com

1. Determining printing character size and direction

- Specifying printing character size
  Print character size (1~9) must be stored in the character size memory (79F4H).
- Specifying printing direction
  Print direction (0~3) must be stored in the print direction memory (79F2H).

2. Restoration of array and two-character variable

- Array variable and two-character variable that cleared by means of RUN operation or CLEAR command can be restored by operating the variable pointer (7899H, 789AH). Number of bytes dominated by the array variable and two-character variable should be deducted from the last address of the free area, then store it in the variable pointer.
- How to store
  Assume now "x" is the number that the byte count of array and two-character variable deducted from the last address of the free area.
  $a = x / 256$
  $b = x - 256 * a$
  Where,
  a: High order two digits when x is displayed in hexadecimal figure.
  b: Low order two digits when x is displayed in hexadecimal figure.
  Whereas, store a in the variable pointer 7899H and b in 789AH.
- Last address of the free area

System configuration	Address
PC-1500 only	4800H
PC-1500 + CE-151	5800H
PC-1500 + CE-155	6000H
PC-1500 + CE-159	4800H

- Number of bytes used for the array and two-character variable
  Numeric array variable:
  7 bytes + 8 bytes * size of array
  Nonnumeric array variable:
  7 bytes + character length * size of array
  (Character size is normally 16 characters.)
  Numeric two-character variable:
  7 bytes + 8 bytes
  Character two-character variable:
  7 bytes + 16 bytes

3. Knowing the use of CE-150

Because the system program ROM of the CE-150 resides in the CE-150, FFH will be read when a ME0 address range of A000H thru BFFFH is accessed with the CE-150 not connected to the PC-1500. If connected, the contents of A000H will fetch C0H.
NOTE: PV must be reset.

promsoft.com

ADDENDUM

SPECIAL NOTICE TO PC1500A CUSTOMERS

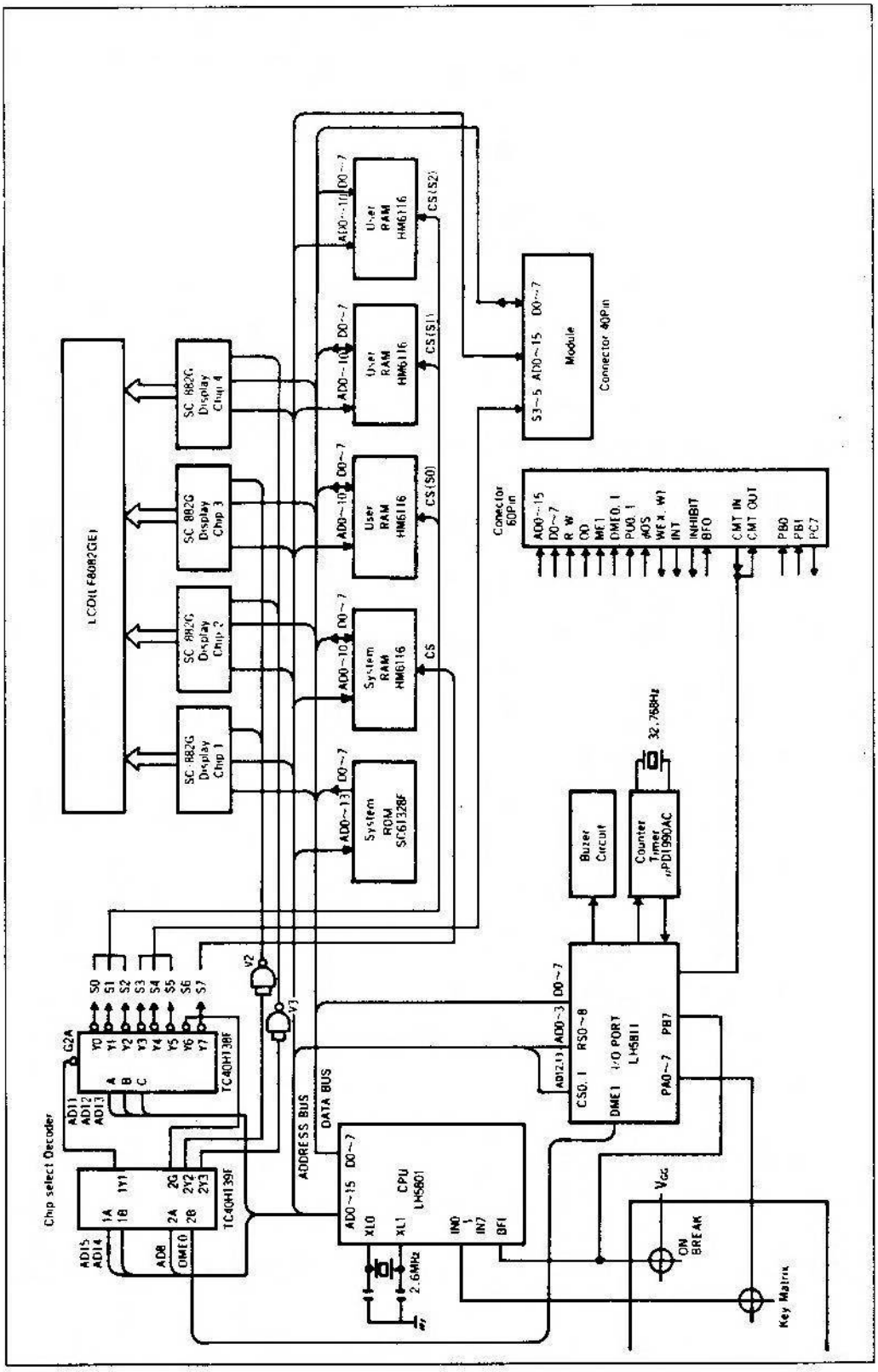
Because the PC-1500A provides more RAM than the PC-1500, some of the descriptions of the CHIP SELECT SIGNAL, MEMORY MAP, and 40-PIN CONNECTOR (for memory modules) require modifications to suit the PC-1500A.

A summary of the differences between the PC-1500A and the PC-1500 is given in this addendum. Take notice of these differences when using machine language.

Page	PC-1500	PC-1500A
89	④ **Chip select decoder** ...and it is used to select chip by means of S0~4, S6, S7, $\overline{2Y2}$ and $\overline{2Y3}$. ...	④ **Chip select decoder** ...and it is used to select chip by means of S0~7, $\overline{2Y2}$ and $\overline{2Y3}$. ...
	⑦ **System RAM** Because the 4-bit × 1K bytes TC5514 is used in a pair, the data bus is divided into two of D0~D3 and D4~D7 and the select signal S7 is commonly shared so as to be compatible with the 8-bit RAM. Address is within 7800H to 7BFFH of the ME0 area which is used for the system memory area and for the fixed variable area.	⑦ **System RAM** Address is within 7800H to 7FFFH of the ME0 area which is used for the system memory area, machine language area (7C01H~7FFFH) and for the fixed variable area.
90	⑧ **User RAM** It is the user RAM for which 8-bit × 2KB HM6116 is used. Address selected by S0 is within 4000H to 47FFH.	⑧ **User RAM** It is the user RAM for which 8-bit × 2KB HM6116 is used. Address selected by S0~S2 is within 4000H to 57FFH.
	4-2-2. Block diagram	**4-2-2. Block diagram for the PC-1500A** Refer to 4-2-2 diagram for the PC-1500A.
91	**4-2-3. Chip select circuit**	**4-2-3. Chip select circuit for the PC-1500A** Refer to 4-2-3 circuit for the PC-1500A.

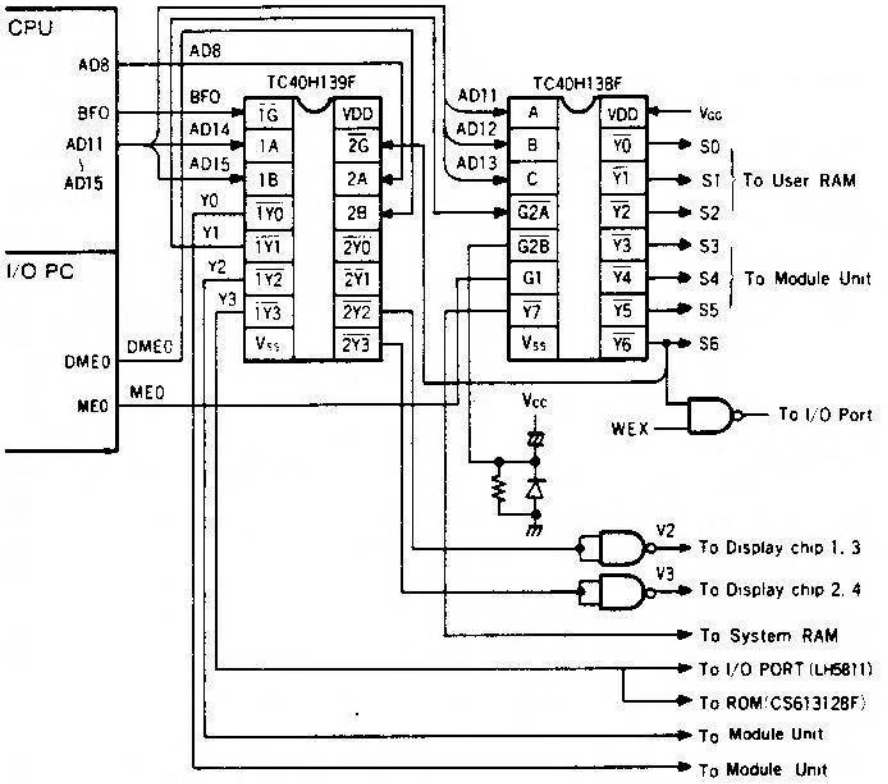
promsoft.com

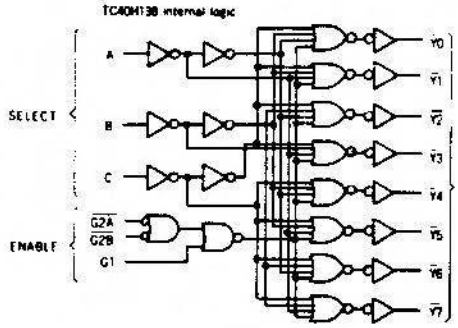
4-2-2. Block diagram for the PC-1500A



promsoft.com

4-2-3. Chip select circuit for the PC-1500A



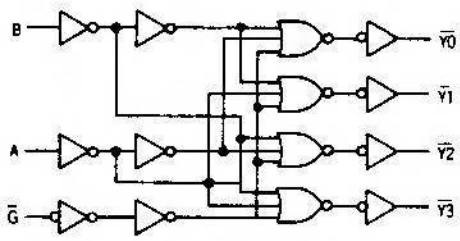


Truth table

INPUT						OUTPUT							
ENABLE			SELECT										
G1	G2A	G2B	B	B	C	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
L	*	*	*	*	*	H	H	H	H	H	H	H	H
*	H	*	*	*	*	H	H	H	H	H	H	H	H
*	*	H	*	*	*	H	H	H	H	H	H	H	H
H	L	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	H	L	L	H	L	H	H	H	H	H	H
H	L	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	L	H	H	H	L	H	H	H	H
H	L	L	L	L	H	H	H	H	H	L	H	H	H
H	L	L	H	L	H	H	H	H	H	H	L	H	H
H	L	L	L	H	H	H	H	H	H	H	H	L	H
H	L	L	H	H	H	H	H	H	H	H	H	H	L

*: Dont care

TC40H139 circuit
½ circuit



Truth table

INPUT			OUTPUT			
ENABLE	SELECT					
G	A	B	Y0	Y1	Y2	Y3
H	※	※	H	H	H	H
L	L	L	L	H	H	H
L	H	L	H	L	H	H
L	L	H	H	H	L	H
L	H	H	H	H	H	L

※ = Irrelevant

promsoft.com

Page	PC-1500	PC-1500A
92	**● S0~S7 is selected by the decoder IC (TC40H138F) when the gate signal input ME0 (G1) is high, Y1 ($\overline{G2A}$) low, and $\overline{G2B}$ is low (normally low).** S1 ($\overline{Y1}$) .....With high ..., low so as to select the optional user RAM area. ... S2 ($\overline{Y2}$) .....With low ..., the S2 output goes low so as to select the optional user RAM area. ... S3 ($\overline{Y3}$) .....With low ..., the S3 output goes low so as to select the user RAM area. ... S4 ($\overline{Y4}$) .....With high ..., the S4 output goes low so as to select the user RAM area. ... S5 .....Do not use S7 ($\overline{Y7}$) .....so as to select the system RAM (TC5514) ...	**● S0~S7 is selected by the decoder IC (TC40H138F) when the gate signal input ME0 (G1) is high, Y1 ($\overline{G2A}$) low, and $\overline{G2B}$ is low (normally low).** S1 ($\overline{Y1}$) .....With high ..., low so as to select the user RAM area. ... S2 ($\overline{Y2}$) .....With low ..., the S2 output goes low so as to select the user RAM area. ... S3 ($\overline{Y3}$) .....With low ..., the S3 output goes low so as to select the optional user RAM area. ... S4 ($\overline{Y4}$) .....With high ..., the S4 output goes low so as to select the optional user RAM area. ... S5 ($\overline{Y5}$) .....Optional user RAM area (Address assignment of 6800H~6FFFH) S7 ($\overline{Y7}$) .....so as to select the system RAM (HM6116) ...
93	**Chip select signal**	**Chip select signal for the PC-1500A** Refer to following chart for the PC-1500A.

promsoft.com

Chip select signal for the PC-1500A

Y0 ($\overline{1Y0}$)			0000H 3FFFH	OPTIONAL USER MEMORY
Y1 ($\overline{1Y1}$)	S0 ($\overline{Y0}$)		4000H 47FFH	STANDARD USER MEMORY
	S1 ($\overline{Y1}$)		4800H 4FFFH	
	S2 ($\overline{Y2}$)		5000H 57FFH	
	S3 ($\overline{Y3}$)		5800H 5FFFH	OPTIONAL USER MEMORY
	S4 ($\overline{Y4}$)		6000H 67FFH	
	S5 ($\overline{Y5}$)		6800H 6FFFH	
	S6 ($\overline{Y6}$)		7000H 75FFH	INHIBITED
		V2 ($\overline{2Y2}$)	7600H 76FFH	STANDARD USER AND SYSTEM MEMORY
		V3 ($\overline{2Y2}$)	7700H 77FFH	
	S7 ($\overline{Y7}$)		7800H 7FFFH	MACHINE LANGUAGE AREA (7C01H~7FFFH)
Y2 ($\overline{1Y2}$)			8000H BFFFH	CE-150 SYSTEM PROGRAM,I/O PORT CE-153 I/O PORT CE-158 SYSTEM PROGRAM
Y3 ($\overline{1Y3}$)			C000H FFFFH	PC-1500A SYSTEM PROGRAM I/O PORT CE-158 I/O PORT UART

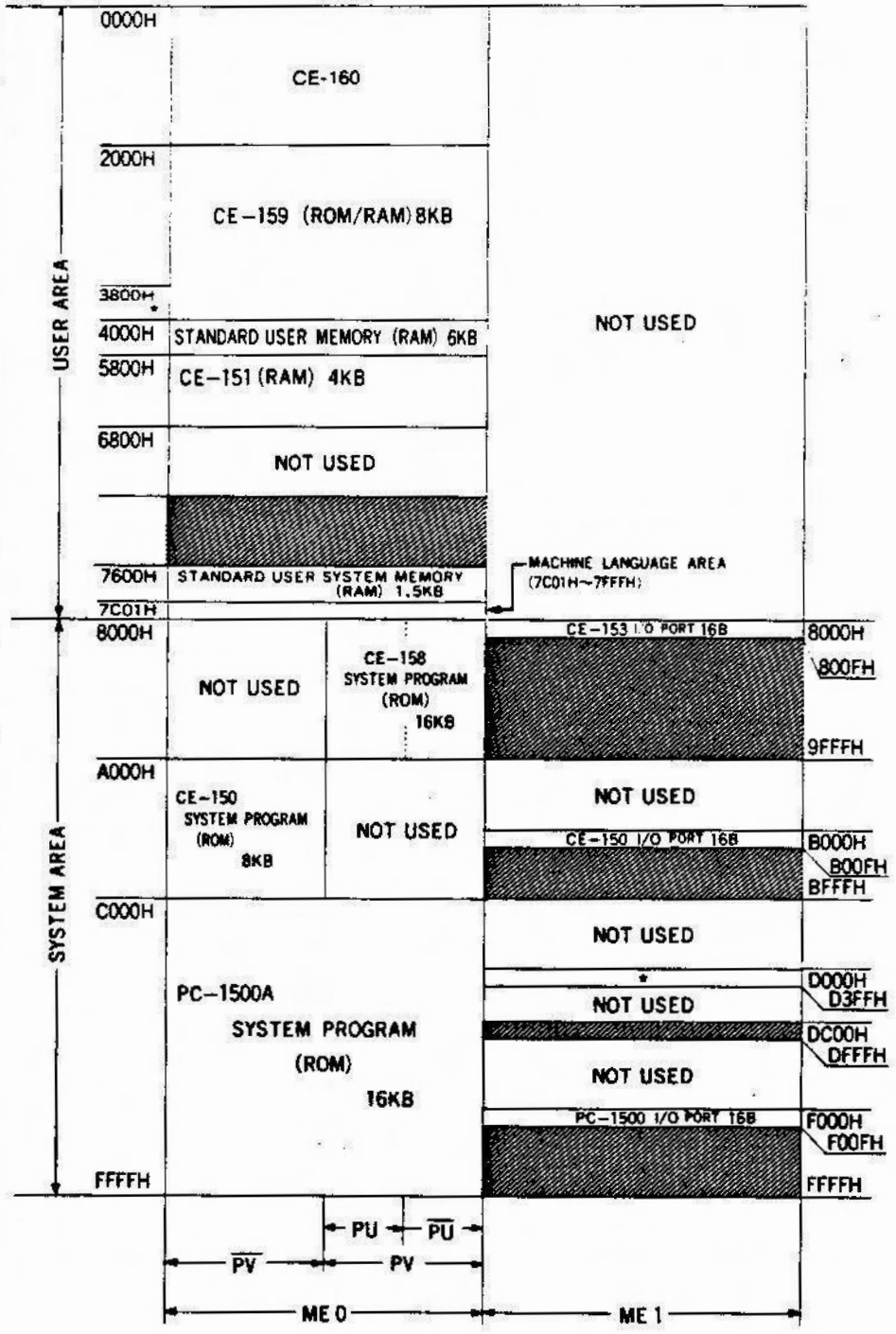
NOTE: S0~S7, V2, and V3 are applicable only for ME0 area.

promsoft.com

Page	PC-1500	PC-1500A
94	**4-2-4. PC-1500 system memory map** ① . . . With the PC-1500, ME0 memory area 4000H thru 47FFH and 7600H thru 7BFFH are used for the user memory and C000H thru FFFFH for the system program. . . .	**4-2-4. PC-1500A system memory map** ① . . . With the PC-1500A, ME0 memory area 4000H thru 57FFH and 7600H thru 7FFFH are used for the user memory and C000H thru FFFFH for the system program. . . .
95	MEMORY MAP I	MEMORY MAP I Refer to following MAP I for the PC-1500A.
96	MEMORY MAP II	MEMORY MAP II Refer to following MAP II for the PC-1500A.

promsoft.com

MEMORY MAP I for the PC-1500A

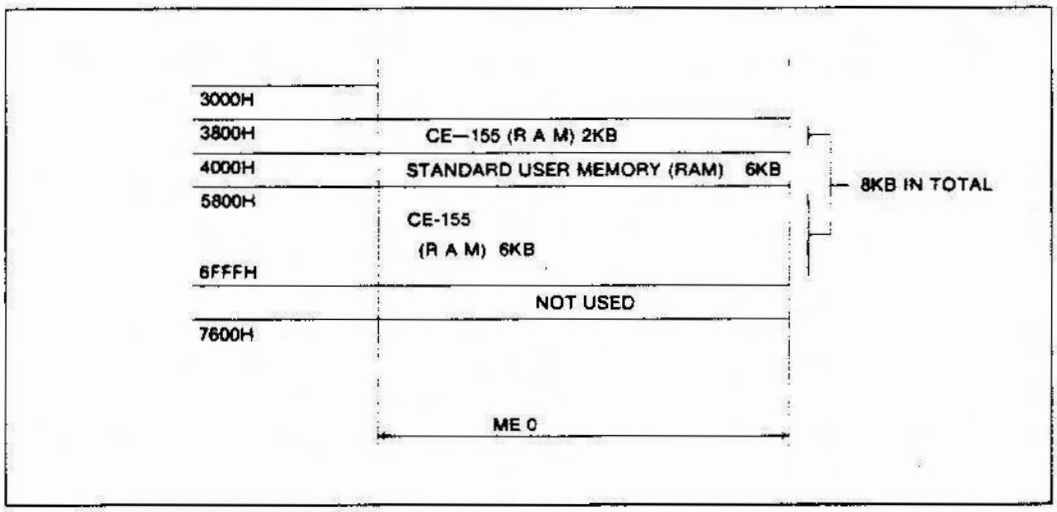


- Area reserved for option will become unused, in case the respective option is not connected.
- Refer to "Memory map II and III" for *
- PV and PU are bank select signals.
- Shadowed area can not be used.

promsoft.com

MEMORY MAP II for the PC-1500A

MEMORY MAP when the CE-155 is used.



Page	PC-1500	PC-1500A
102	**4-3-1. 40-pin connector** Pin no. / Signal name / Description 5 / S4 / Address designation of 0000H~ 3FFFH 16 / S1 / Address designation of 4800H~4FFFH 17 / S2 / Address designation of 5000H~57FFH 18 / S3 / Address designation of 5800H~5FFFH	**4-3-1. 40-pin connector** Pin no. / Signal name / Description 5 / NC / NC 16 / S3 / Address designation of 5800H~5FFFH 17 / S4 / Address designation of 6000H~67FFH 18 / S5 / Address designation of 6800H~6FFFH The above portion should be changed.
154	**5-1. PC-1500 Circuit diagram**	**5-1. PC-1500A Circuit Diagram** For the PC-1500A, refer to the circuit diagram on the next page.

promsoft.com