September 1973
A publication of:
Philips-Electrologica B.V.
Main Marketing Group OEM
P.O. Box 245 Apeldoorn, Netherlands

Printed in The Netherlands

5122 991 16902

PHILIPS

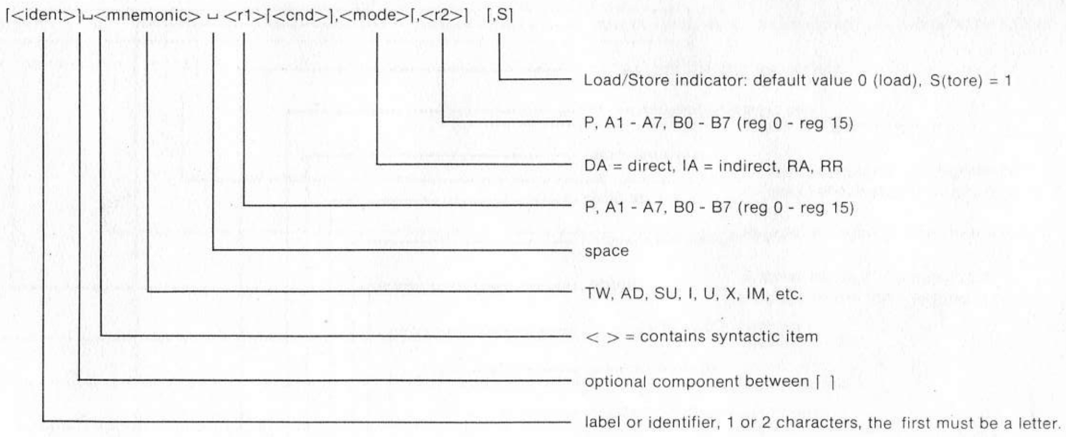
P850M/P855M

Hardware Reference Data

PHILIPS Data Systems

CONTENTS

ASSEMBLY LANGUAGE P850M - GENERAL FORMAT - 1)



[<ident>]␣<mnemonic> ␣ <r1>[<cnd>],<mode>[,<r2>] [,S]

Load/Store indicator: default value 0 (load), S(tore) = 1

P, A1 - A7, B0 - B7 (reg 0 - reg 15)

DA = direct, IA = indirect, RA, RR

P, A1 - A7, B0 - B7 (reg 0 - reg 15)

space

TW, AD, SU, I, U, X, IM, etc.

< > = contains syntactic item

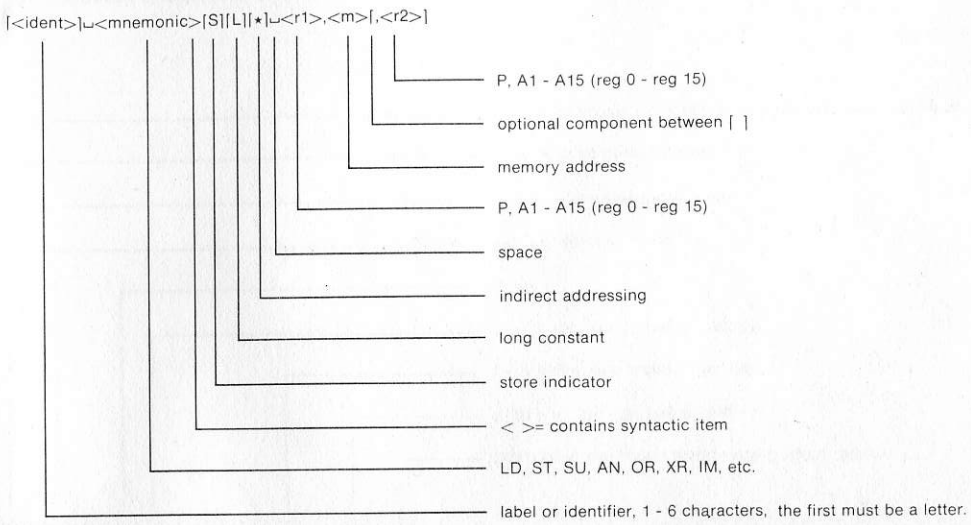
optional component between [ ]

label or identifier, 1 or 2 characters, the first must be a letter.

* in operand field represents current value of location counter

[1] For details, see P850M User Manual (publication number 5122 991 1453X)

ASSEMBLY LANGUAGE P855M - general format - 2)



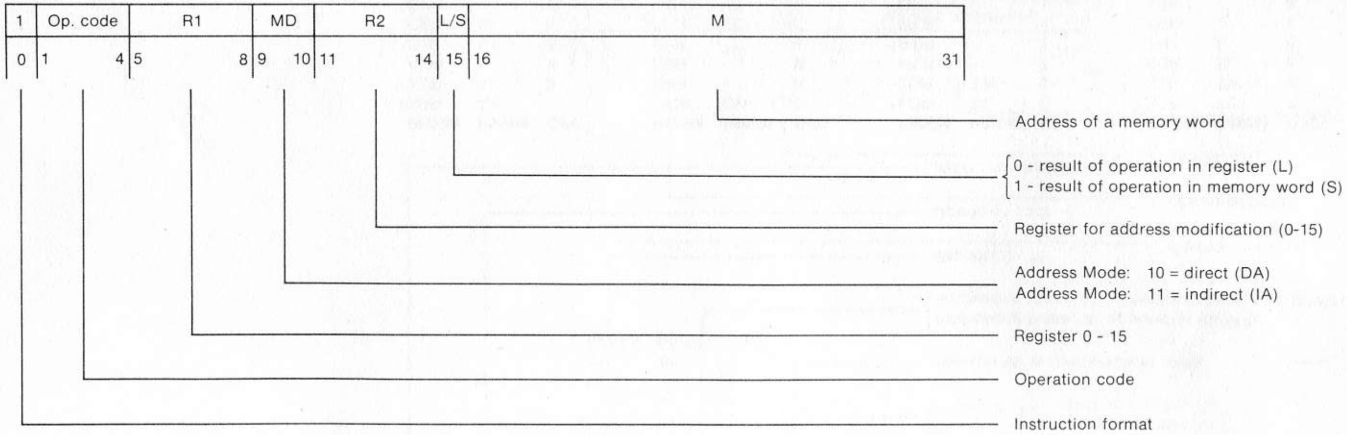
[<ident>]␣<mnemonic>[S][L][∗]␣<r1>,<m>[,<r2>]

- P, A1 - A15 (reg 0 - reg 15)
- optional component between [ ]
- memory address
- P, A1 - A15 (reg 0 - reg 15)
- space
- indirect addressing
- long constant
- store indicator
- < >= contains syntactic item
- LD, ST, SU, AN, OR, XR, IM, etc.
- label or identifier, 1 - 6 characters, the first must be a letter.

∗ in operand field represents current value of location counter (except when immediately after mnemonic)

2)For details, see P855M/P860M System Software Manual Paper Tape or Disc (publication number 5122 991 1155X or 5122 991 1162X)

Memory Reference Instructions



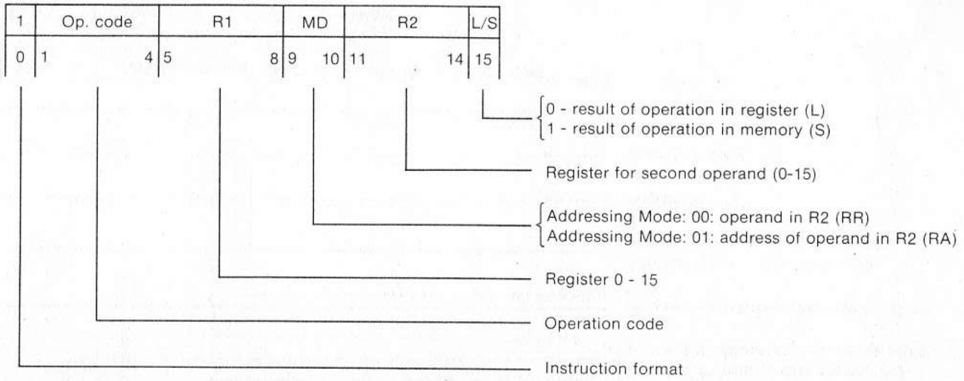
1	Op. code	R1	MD	R2	L/S	M
0	1 4	5 8	9 10	11 14	15	16 31

- Address of a memory word
- 0 - result of operation in register (L)
1 - result of operation in memory word (S)
- Register for address modification (0-15)
- Address Mode: 10 = direct (DA)
Address Mode: 11 = indirect (IA)
- Register 0 - 15
- Operation code
- Instruction format

P855M	P850M	OPC	P855M	P850M	OPC	P855M	P850M	OPC	P855M	P850M	OPC
*ABI	AB	1	*CFI	CF	E	*LC	TC	C	*RB	RB	B
AD	AD	2	CW	CW	D	LD	TW	0	*RF	RF	A
AN	I	4	DA	-	A	ML	-	7	*SC	TC	C
C1	C1	F	DS	-	B	*MS	-	7	*ST	TW	0
C2	C2	3	DV	-	9	MU	-	8	SU	SU	3
CC	-	D	IM	IM	2	OR	U	5	XR	X	6

*No CR setting

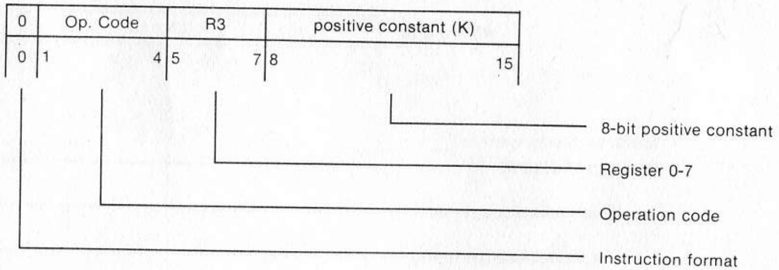
Register/Register Instructions


1
Op. code
R1
MD
R2
L/S
0
1
4
5
8
9
10
11
14
15
0 - result of operation in register (L)
1 - result of operation in memory (S)
Register for second operand (0-15)
Addressing Mode: 00: operand in R2 (RR)
Addressing Mode: 01: address of operand in R2 (RA)
Register 0 - 15
Operation code
Instruction format

P855M	P850M	OPC	P855M	P850M	OPC	P855M	P850M	OPC	P855M	P850M	OPC
*ABR	AB	1	CWR	CW	D	*LCR	TC	C	*SCR	TC	C
ADR	AD	2	DAR	-	A	LDR	TW	0	*STR	TW	0
ANR	I	4	DSR	-	B	MLR	-	7	SUR	SU	3
C1R	C1	F	DVR	-	9	*MSR	-	7	TM	I	4
C2R	C2	3	ECR	EC	C	MUR	-	8	TNM	U	6
CCR	-	D	IMR	IM	2	ORR	U	5	XRR	X	6
*CFR	CF	E				*RTN	RT	E			

*No CR setting

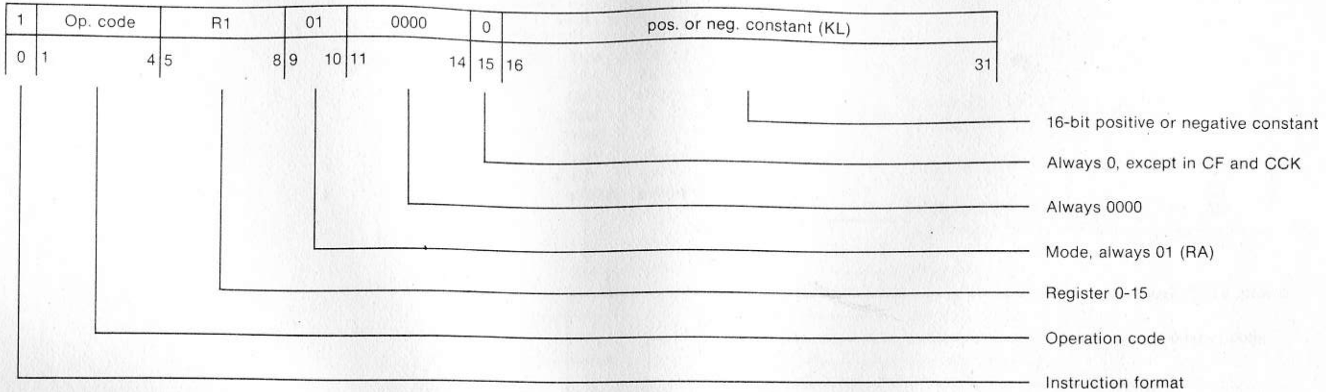
Constant instructions (short format)



P855M	P850M	OPC
*AB	BK	1
ADK	AK	2
ANK	IK	4
*LDK	LK	0
ORK	UK	5
SUK	SK	3
XRK	XK	6

*No CR setting

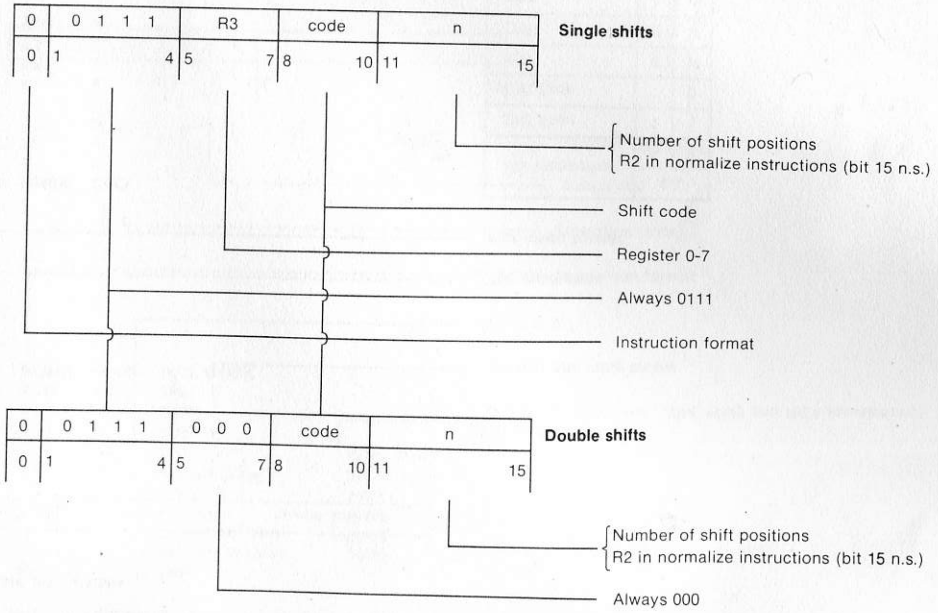
Constant Instructions (long format)



P855M	P850M	OPC	P855M	P850M	OPC
*ABL	AB	1	DVK	-	9
ADKL	AD	2	LDKL	TW	0
ANKL	I	4	*LCK	TC	C
CCK	-	D	MLK	-	7
*CF	CF	E	MUK	-	8
CWK	CW	D	ORKL	U	5
DAK	-	A	SUKL	SU	3
DSK	-	B	XRKL	X	6

*No CR setting

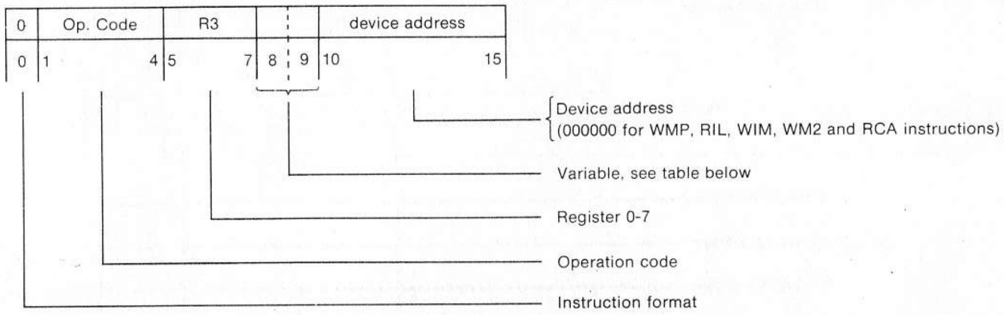
Shift Instructions



P855M	P850M	Shift Code
DLA	-	0
DLC	-	6
DLL	-	2
*DLN	-	4
DRA	-	1
DRC	-	7
DRL	-	3
*DRN	-	5
SLA	-	0
-	LA	0
SLC	-	6
SLL	-	2
-	LL	2
SLN	-	4
SRA	-	1
-	RA	1
SRC	-	7
-	RC	7
SRL	-	3
*SRN	-	5

*No CR setting

Input/Output Instructions

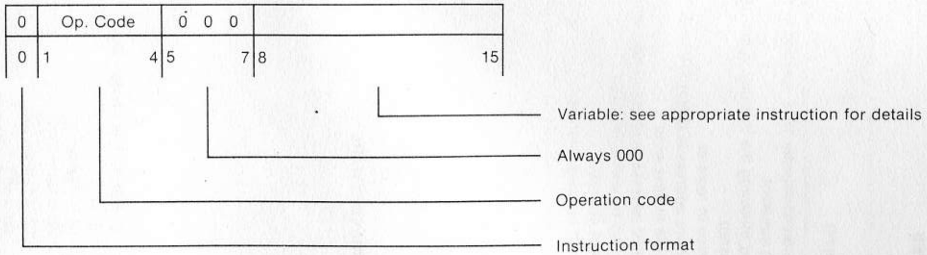


P855M	P850M[1]	OPC
CIO	CT	8
INR	IN	9
OTR	OT	8
RCA	–	9
RIL	RL	9
SST	SS	9
TST	TS	9
WIM	WM	8
WM2	–	8
WMP	–	8

[1] Mnemonics are not recognized by the Assembler

I/O Instruction	bit	
	8	9
CIO Start	1	1
CIO Stop	1	0
INR	0	X
OTR	0	X
TST	1	0
SST	1	1

Miscellaneous instructions



P855M	P850M	HEX. CODE	HEX. CODE
ENB	EN	2840	
HLT	HT	207F	
INH	IH	20BF	
LKM	LM	2804	
RIT	RI	control panel	20DF
		PFAURE	20EF
		RT clock	20FE
		prog. error/LKM	20FB
		memory protect	20FD
SMD	⌐	2801	

[1] Mnemonics are not recognized by the Assembler

ADDRESSING MODES - REFERENCE TYPES

Mode	MD		R2	Effective	
	9	10		Addresses	Meaning
RR	0	0	xxxx	R2	Register-to-register (2nd operand in R2)
RA	0	1	0000	(P)	Long constant. (word following the instruction is a 16-bit operand).
RA	0	1	≠0	(R2)	Address in register (effective address is in R2)
DA	1	0	0000	(q)	Address in next word
DA	1	0	≠0	(q) + (R2)	Indexed address (address in next word is modified by contents of R2).
IA	1	1	0000	[(q)]	Indirect address
IA	1	1	≠0	[(q) + (R2)]	Indirect indexed address

P = instruction counter
q = contents of P (i.e. word after instruction)
R2 = bits 11 - 14 of instruction word
() = contents of
[ ] = indirect addressing.

FILE CODES TO BE USED WITH STANDARD SOFTWARE

a P850M File codes

01 = Source input file
02 = Listing output file
03 = Punch output file
04 = Object code input file
05 = Operator's typewriter

b P855M File codes - Basic

01 = Standard Source Input
02 = Standard Listing Output
03 = Standard Punch Output
04 = Standard Object Input
05 = Standard Operator Keyboard
06 = ASR Reader
07 = ASR Punch
08 = Punched Tape Reader
09 = Tape Punch
0A = Line Printer
0B = Card Reader
0C = Cassette Tape.

c P855M File codes - Disc

01 = Typewriter
02 = Print unit
03 = Punch unit
D4 = /S
D5 = /0 } for DOM
D6 = /L
D7 = System object file
D8 = User object file
E0 = Control command input
E1 = Source input
E2 = Object input
EF = Typewriter (display only)
F0 - FF = Disc units

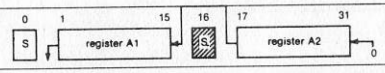
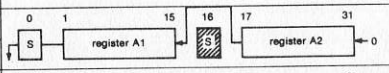
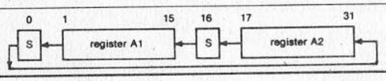
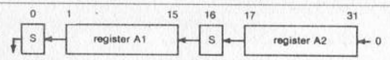
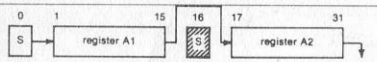
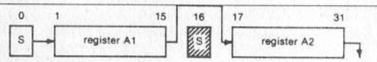
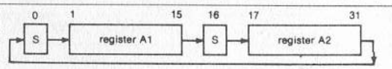
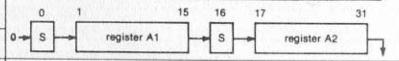
DIRECTIVES

```
[<ident>] ␣DATA␣<data expression>[,<data expression>, ....]
          (up to 16 words generated)
[<ident>] ␣EQU␣<predefined expression>
          ␣IDENT␣<module name>
[<ident>] ␣END␣[<predefined expression>][,<symbol>]
[<ident>] ␣RES␣<predefined absolute expression>
          ␣AORG␣<predefined absolute expression>
          ␣RORG␣[<predefined relocatable expression>]
          ␣ENTRY␣<entry point name>[,<entry point name>, ....,
          <entry point name>]
          ␣EXTRN␣<external name>[,<external name>, ....,
          <external name>]
          ␣STAB␣[<internal symbol>][,<internal symbol>, ....,
          <internal symbol>]
          ␣NLIST␣
          ␣LIST␣
          ␣EJECT␣
          ␣[IFT|IFF]␣<predefined absolute expression>=
          <predefined absolute expression>
          ␣XIF␣
[<ident>] ␣COMN␣<common field definition list>
 <ident>  ␣FORM␣<field definition>[,<field definition>, ....,
          <field definition>][/<field number list>]
 <ident>  ␣XFORM␣<FORM defined pseudo-mnemonic>,<field list>
          ␣GEN␣

          <data expression> ::= <expression>|<character string>
          <field definition> ::= <field length definition>
          [=<field value definition>]
```

name (in alphabetical order)	mnemonic		for-mat	OP-code	mode	L/S (0/1) bit	function	condition register	execution time in cycles 6)			remarks
	P850M	P855M							P850M	P855M		
									m.c.	l.c.	m.c.	
Absolute branch	AB	ABI	1	0001	10	0	(M) → P		7		3	bits 5-7: condition
					10	0	(M + (R2)) → P		7		3	bit 8: n.s.
					11	0	((M)) → P		9		4	
					11	0	((M + (R2))) → P		9		4	
						0	no branch: (P) + 4 → P		3	1	1	
Absolute conditional	BK	AB	0	0001	-	n.s.	K → P }		4	1	1	short format
branch (with constant)							(q) → P }	3)				
							no branch: (P) + 2 → P		3	1	1	
Absolute conditional	AB	ABL	1	0001	01	0	KL → P		5	1	2	long format
branch (with constant)							no branch: (P) + 2 → P		3	1	1	
Absolute conditional	AB	ABR	1	0001	00	n.s.	(R2) → P		4	1	1	bits 5-7: condition
branch to register					01	0	((R2)) → P		5		2	bit 8: n.s.
							no branch: (P) + 2 → P		3	1	1	
Add constant	AK	ADK	0	0010	-	-	(R3) + K → R3		4	1	1	short; R1 = 1111:
	AD	ADKL	1	0010	01	0	(R1) + KL → R1		5		2	long system mode
Addition	AD	AD	1	0010	10	0	(R1) + (M) → R1		7		3	R1 = 1111: system mode
					10	1	(R1) + (M) → M	2)	8		4	when l/s bit = 1,
					10	0	(R1) + (M + (R2)) → R1		7		3	R1 must be ≠ 0
					10	1	(R1) + (M + (R2)) → M		8		4	
					11	0	(R1) + ((M)) → R1		9		4	
					11	1	(R1) + ((M)) → M		10		5	
					11	0	(R1) + ((M + (R2))) → R1		9		4	
					11	1	(R1) + ((M + (R2))) → M	2)	10		5	
Addition/register	AD	ADR	1	0010-	00	n.s.	(R1) + (R2) → R1		4	1	1	when l/s bit = 1,
					01	0	(R1) + ((R2)) → R1		5		2	R1 must be ≠ 0;
					01	1	(R1) + ((R2)) → (R2)		6		3	R1 = 1111: system mode
Call function	CF	CFI	1	1110			(P) → (R1), (R1) - 2 ⟶ R1 } P855/P860					R1 = 1111: system mode
							(PSW) → (R1), (R1) - 2 → R1 }					
							(CR) → (R1), (R1) - 2 ⟶ R1 } P850					
							(P) → (R1), (R1) - 2 ⟶ R1 }					
							then:					
					10	1	(M) → P		13	2	5	if reg. 15 = stack
					10	1	(M + (R2)) → P		13	2	5	pointer,
					11	1	((M)) → P		15	2	6	$>128_{10}$ → overflow
					11	1	((M + (R2))) → P		15	2	6	
Call function/constant	CF	CF	1	1110	01	1	(P) → (R1), (R1) - 2 ⟶ R1 } P855/60	3)	11	2	4	if reg. 15 = stack
							(PSW) → R1, (R1) - 2 → R1, }					pointer
							KL → P }					$>128_{10}$ → overflow;
							(CR) → R1, (R1) - 2 ⟶ R1 }					
							(P) → R1, (R1) - 2 ⟶ R1 } KL → P P850					R1 = 1111: system mode

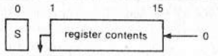
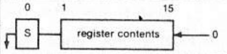
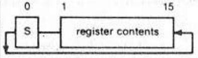
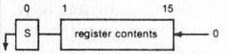
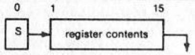
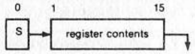
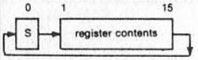
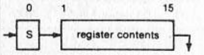
name (in alphabetical order)	mnemonic		for-mat	OP-code	mode	L/S (0/1) bit	function	condition register	execution time in cycles 6)			remarks
									P850M	P855M		
	P850M	P855M							m.c.	l.c.	m.c.	
Call function/register	CF	CFR	1	1110			(P) → (R1),·(R1) - 2 → R1 P855/60					if reg. 15 = stack
							(PSW) → (R1), (R1) - 2 → R1					pointer,
							(CR) → (R1), (R1) - 2 → R1 P850					$>128_{10}$: overflow
							(P) → (R1), (R1) - 2 → R1	3)				
							then:					R1 = 1111: system mode
					00	1	(R2) → P		10	3	3	
					01	1	((R2)) → P		11	2	4	
Compare characters	-	CC	1	1101	10	1	(R1)r ÷ (M) l/r → CR				4	R1 = 1111: system mode
					10	1	(R1)r ÷ (M + (R2)) l/r → CR				4	
					11	1	(R1)r ÷ ((M)) l/r → CR				5	
					11	1	(R1)r ÷ ((M + (R2))) l/r → CR				5	
Compare characters	-	CCR	1	1101	01	1	(R1)r ÷ ((R2)) l/r → CR				3	R1 = 1111: system mode
register/register								4)				
Compare character	-	CCK	1	1101	01	1	(R1)r ÷ KLl → CR		-		3	R1 = 1111: system mode
with constant												
Compare words	CW	CW	1	1101	10	0	(R1) ÷ (M) → CR		7		3	R1 = 1111: system mode
					10	0	(R1) ÷ (M + (R2)) → CR		7		3	
					11	0	(R1) ÷ ((M)) → CR		9		4	
					11	0	(R1) ÷ ((M + (R2))) → CR		9		4	
Compare words	CW	CWR	1	1101	00	n.s.	(R1) ÷ (R2) → CR		4	1	1	R1 = 1111: system mode
register/register					01	0	(R1) ÷ ((R2)) → CR		5		2	
Compare word	CW	CWK	1	1101	01	0	(R1) ÷ KL → CR		5		2	R1 = 1111: system mode
with constant												
Control Input/Output	CT	CIO	0	1000	-	-	Start (bit 9=1) or stop (bit 9=0)	5)	4		3	bit 8 = 1
							any I/O operation					system mode
							quotient remainder					
Divide	-	DV	1	1001	10	0	(A1, A2) / (M) → A2 A1		-	10.5	3	
					10	0	(A1, A2) / (M + (R2)) → A2 A1		-	10.5	3	
					11	0	(A1, A2) / ((M)) → A2 A1		-	10.5	4	
					11	0	(A1, A2) / ((M + (R2)))→ A2 A1		-	10.5	4	
							quotient remainder					
Divide by constant	-	DVK	1	1001	01	0	(A1, A2) / KL → A2 A1		-	10.5	2	
							quotient remainder	2)				
Divide registers/registers	-	DVR	1	1001	00	0	(A1, A2) / (R2) → A2 A1		-	11.5	1	
					01	0	(A1, A2) / ((R2)) → A2 A1		-	10.5	2	
Double Add	-	DA	1	1010	10	0	(M, M + 1) + (A1, A2) → A1, A2		-	1	4	
					10	0	(M + (R2), M + (R2) + 1) + (A1, A2) → A1, A2		-	1	4	
					11	0	((M), (M) + 1) + (A1, A2) → A1, A2		-	1	5	
					11	0	((M + (R2)), ((M + (R2) + 1) + (A1, A2) → A1, A2		-	1	5	

name	mnemonic		for-mat	OP-code	mode	L/S (0/1) bit	function	condition register	execution time in cycles 6)			remarks
(in alphabetical order)	P850M	P855M							P850M	P855M		
									m.c.	l.c.	m.c.	
Double add registers/	-	DAR	1	1010	00	0	(R2, R2 + 1) + (A1, A2) ⟶ A1, A2	2)	–	3	1	
registers					01	0	((R2), (R2 + 1)) + (A1, A2) → A1, A2		–	1	3	
Double add with constant	-	DAK	1	1010	01	0	KL + (A1, A2) → A1, A2		–	1	3	
Double subtract	-	DS	1	1011	10	0	(A1, A2) – (M, M + 1) ⟶ A1, A2		–	1	4	
					10	0	(A1, A2) – (M + (R2), M + (R2) + 1) → A1, A2		–	1	4	
					11	0	(A1, A2) – ((M), (M) + 1) ⟶ A1, A2		–	1	5	
					11	0	(A1, A2) – ((M + (R2)), ((M + (R2)) + 1) → A1, A2		–	1	5	
Double subtract	-	DSR	1	1011	00	0	(A1, A2) – (R2, R2 + 1) ⟶ A1, A2		–	3	1	
registers/registers					01	0	(A1, A2) – ((R2), (R2 + 1)) → A1, A2		–	1	3	
Double subtract	-	DSK	1	1011	01	0	(A1, A2) – KL → A1, A2		–	1	3	
with constant												
Double left and	-	DLN	1	0111	-	n.s.		3)	–	$(1.5+\frac{n}{2})$	1	bits 8-10: 100
normalize shift												bits 11-14: R2;
												bit 15: n.s.
Double left arithmetic shift	-	DLA	0	0111	-	-		2)	–	$(1.5+\frac{n}{2})$		bits 8-10: 000
Double left circular shift	-	DLC	0	0111	-	-		1)	–	$(1.5+\frac{n}{2})$	1	bits 8-10: 110
Double left logical shift	-	DLL	0	0111	-	-		3)	–	$(1.5+\frac{n}{2})$	1	bits 8-10: 010
Double right and	-	DRN	0	0111	-	-			–	$(1.5+\frac{n}{2})$	1	bits 8-10: 101
normalize shift												bits 11-14: R2;
												bit 15: n.s.
Double right arithmetic	-	DRA	0	0111	-	-		1)		$(1.5+\frac{n}{2})$	1	bits 8-10: 001
shift												
Double right circular shift	-	DRC	0	0111	-	-			–	$(1.5+\frac{n}{2})$	1	bits 8-10: 111
Double right logical shift	-	DRL	0	0111	-	-			–	$(1.5+\frac{n}{2})$	1	bits 8-10: 011
Enable interrupt	EN7)	ENB	0	0101	-	-	machine status = 'permit interrupt'	3)	4	1	1	bits 8-15: 01000000
Exchange characters	EC	ECR	1	1100	00	n.s.	(R2) → (R1)r; (R2)r → (R1)		5	1	1	R1 = 1111: system mode
register/register												
Exclusive OR	X	XR		0110	10	0	(R1) ⊽ (M) → R1	1)	7		3	R1 = 1111: system mode
					10	1	(R1) ⊽ (M) → M		8		4	
					10	0	(R1) ⊽ (M + (R2)) → R1		7		3	
					10	1	(R1) ⊽ (M + (R2)) → M + (R2)		8		4	
					11	0	(R1) ⊽ ((M))→R1		9		4	
					11	1	(R1) ⊽ ((M))→(M)		10		5	
					11	0	(R1) ⊽ ((M + (R2)))→R1		9		4	
					11	1	(R1) ⊽ ((M + (R2)))→(M + (R2))		10		5	

name (in alphabetical order)	mnemonic		for-mat	OP-code	mode	L/S (0/1) bit	function	condition register	execution time in cycles 6)			remarks
	P850M	P855M							P850M	P855M		
									m.c.	l.c.	m.c.	
Exclusive OR	X	XRR	1	0110	00	0	(R1) ∀ (R2) → R1	1)	4	1	2	R1 = 1111: system mode
register/register					01	0	(R1) ∀ ((R2)) → R1		5		2	
					01	1	(R1) ∀ ((R2)) → R2		6		3	
Exclusive OR	XK	XRK	0	0110	-	-	$(R3)_{8-15}$ ∀ K → $R3_{8-15}$		4	1	1	short; R1 = 1111: system
with constant												mode
Exclusive OR	X	XRKL	1	0110	01	0	(R1) ∀ KL → R1		5		2	long; R1 = 1111: system
with constant												mode
Halt	HT 7)	HLT	0	0100	-	-	machine → 'halt' mode	3)	4	1	1	bits 8-15: 01111111;
												system mode
Increment Memory	IM	IM	1	0010	10	1	(M) + 1 → M	2)	8		4	bits 5-8: 0000
					10	1	(M + (R2)) + 1 → M		8		4	
					11	1	((M +)) + 1 → M		10		5	
					11	1	((M) + (R2))) + 1 → (M + (R2))		10		5	
Increment memory/	IM	IMR	1	0010	01	1	((R2)) + 1 → R2		6		3	bits 5-8: 0000
register												
Inhibit interrupt	IH7)	INH	0	0100	-	-	machine status = 'prohibit all interrupts'	3)	4	1	1	bits 8-15: 10111111
												system mode
Input to register	IN7)	INR	0	1001	-	-	word/character from device → R3	5)	4		3	bit 8 = 0; system mode
Link to monitor	LM7)	LKM	0	0101	-	-	user mode → system mode	3)	4	1	1	bits 8-15: 00000100
Load character	TC	LC	1	1100	10	0	(M) l/r → R1r		7		3	R1 = 1111: system mode
					10	0	(M + (R2)) l/r → R1r		7		3	R1 must be =/0
					11	0	((M)) l/r → R1		9		4	
					11	0	((M + (R2))) l/r → R1r		9		4	
Load character/constant	TC	LCK	1	1100	01	0	KL l → R1r	3)	5		2	R1 = 1111: system mode
Load character/register	TC	LCR	1	1100	01	0	((R2)) l/r → R1r		5		2	R1 = 1111: system mode
Load constant	LK	LDK	0	0000	-	-	K → $R3_{8-15}$, 0 → $R3_{0-7}$		4	1	1	short; R1 = 1111: system
												mode
	TW	LDKL	1	0000	01	0	KL → R1		5		2	long; R1 = 1111: system
												mode
Load register	TW	LD	1	0000	10	0	(M) → R1	1)	7		3	R1 = 111: system mode
					10	0	(M + (R2)) → R1		7		3	
					11	0	((M)) → R1		9		4	
					11	0	((M + (R2))) → R1		9		4	
Load register/ register	TW	LDR	1	0000	00	n.s.	(R2) → R1		4	1	1	R1 = 1111: system mode
					01	0	((R2)) → R1		5		2	
					01	0	(A15) + 2 → A15, ((A15)) → R1		6	1	2	system mode
Logical AND	I	AN	1	0100	10	0	(R1) ∧ (M) → R1		7		3	R1 = 1111: system mode
					10	1	(R1) ∧ (M) → M		8		4	
					10	0	(R1) ∧ (M + (R2)) → R1		7		3	
					10	1	(R1) ∧ (M + R2)) → M + (R2)		8		4	

name (in alphabetical order)	mnemonic		for- mat	OP- code	mode	L/S (0/1) bit	function	condition register	execution time in cycles[6)]			remarks
	P850M	P855M							P850M	P855M		
									m.c.	l.c.	m.c.	
					11	0	(R1) ∧ ((M)) → R1		9		4	
					11	1	(R1) ∧ ((M)) → (M)		10		5	
					11	0	(R1) ∧ ((M + (R2))) → R1		9		4	
					11	1	(R1) ∧ ((M + (R2))) → (M + (R2))		10		5	
Logical AND	I	ANR	1	0100	00	0	(R1) ∧ (R2) → R1		4	1	1	R1 = 1111: system mode
register/register					01	0	(R1) ∧ ((R2)) → R1		5		2	
					01	1	(R1) ∧ ((R2)) → (R2)		6		3	
Logical AND	IK	ANK	0	0100	-	-	$(R3)_{8-15} \wedge K \rightarrow R3_{8-15}$		4	1	1	short; R1 = 1111: system
with constant	I	ANKL	1	0100	01	0	(R1) ∧ KL → R1	1)	5		2	long; mode
Logical OR	U	OR	1	0101	10	0	(R1) ∨ (M) → R1		7		3	R1 = 1111: system mode
					10	1	(R1) ∨ (M) → M		8		4	R1 = 1111: system mode
					10	0	(R1) ∨ (M + (R2)) → R1		7		3	
					10	1	(R1) ∨ (M + (R2)) → M + (R2)		8		4	
					11	0	(R1) ∨ ((M)) → R1		9		4	
					11	1	(R1) ∨ ((M)) → (M)		10		5	
					11	0	(R1) ∨ ((M + (R2))) → R1		9		4	
					11	1	(R1) ∨ ((M + (R2))) → (M + (R2))		10		5	
Logical OR	U	ORR	1	0101	00	0	(R1) ∨ (R2) → R1		4	1	1	R1 = 1111: system mode
register/register					01	0	(R1) ∨ ((R2)) → R1		5		2	
					01	1	(R1) ∨ ((R2)) → (R2)		6		3	
Logical OR	UK	ORK	0	0101	-	-	$(R3)_{8-15} \vee K \rightarrow R3_{8-15}$		4	1	1	short; R1 = 1111: system
with constant												mode
Logical OR	U	ORKL	1	0101	01	0	(R1) ∨ KL → R1		5		2	long; R1 = 1111: system
with constant												mode
Multiple load	-	ML	1	0111	10	0	(M) ... (M + n) → A1 ... An		-		*	*3+(n-1)1 bits 5-8: n
					10	0	(M + (R2)) ... (M + (R2) + n) → A1 ... An		-			
					11	0	((M)) ... ((M) + n) → A1 ... An		-		**	**4+(n-1)1
					11	0	((M + (R2))) ... ((M + (R2)) + n) → A1 .. An		-			n = 1111: system mode
Multiple load/constant	-	MLK	1	0111	01	0	KL1, KL2, ... KLn → A1, A2, ... An		-		n+1	n = 1111: system mode
Multiple load/register	-	MLR	1	0111	01	0	((R2)) → A1; ((R2) + 2) → A2; ..... ;		-		n+1	n = 1111: system mode
							((R2) + 2n-2) → An	1)				bits 5-8; n.
						-	(A15) + 2n → A15; ((A15)) → A1;		-	1	n+1	system mode;
							((A15) - 2) → A2; .... ;					bits 5-8: n
							((A15) - 2n + 2) → An					
Multiple store	-	MS	1	0111	10	1	A1 ... An → M ... M + n		-	n	2+n	n = 1111: system mode;
					10	1	A1 ... An → M + (R2) ... M + (R2) + n	3)	-	n	2+n	bits 5-8: n
					11	1	A1 ... An → (M) ... (M) + n		-	n	2+n	
					11	1	A1 ... An → (M + (R2)) ... (M + (R2)) + n		-	n	2+n	

name (in alphabetical order)	mnemonic		for-mat	OP-code	mode	L/S (0/1) bit	function	condition register	execution time in cycles 6)			remarks
	P850M	P855M							P850M	P855M		
									m.c.	l.c.	m.c.	
Multiple store/register	-	MSR	1	0111	01	1	(A1) → (R2) ; (A2) → (R2) + 2 ;		-	n	n+1	n = 1111: system mode;
							....; (An) → (R2) + 2n - 2					bits 5-8: n
							(A1) → (A15) ; (A2) → (A15) - 2n + 2		-	n	n+1	pointer 128$_{10}$: stack overfl.
							.... ; (An, → (A15) - 2n + 2 ;					bits 5-8: n;
							(A15) - n → (A15)					system mode
Multiply	-	MU	1	1000	10	0	(A2) × (M) → A1, A2		-	8.5	3	
					10	0	(A2) × (M + (R2)) → A1, A2		-	8.5	3	
					11	0	(A2) × ((M)) → A1, A2		-	8.5	4	
					11	0	(A2) × ((M + (R2))) → A1, A2	2)	-	8.5	4	
Multiply registers/registers	-	MUR	1	1000	00	0	(A2) × (R2) → A1, A2		-	9.5	1	
					01	0	(A2) × ((R2)) → A1, A2		-	8.5	2	
Multiply with constant	-	MUK	1	1000	01	0	(A2) × KL → A1, A2		-	8.5	2	
One's complement	C1	C1	1	1111	10	0	($\overline{M}$) → R1		7		3	R1 = 1111: system mode
					10	1	($\overline{M}$) → M		8		4	
					10	0	$\overline{(M + (R2))}$ → R1	1)	7		3	
					10	1	$\overline{(M + (R2))}$ → M + (R2)		8		4	
					11	0	$\overline{((M))}$ → R1		9		4	
					11	1	$\overline{((M))}$ → (M)		10		5	
					11	0	$\overline{((M + (R2)))}$ → R1		9		4	
					11	1	$\overline{((M + (R2)))}$ → (M + (R2))		10		5	
One's Complement	C1	C1R	1	1111	00	n.s.	$\overline{(R2)}$ → R1		4	1	1	when l/s bit = 0,
register/register					01	0	$\overline{((R2))}$ → R1		5		2	R1 must be ≠ 0;
					01	1	$\overline{((R2))}$ → (R2)		6		3	R1 = 1111: system mode
Output from register	OT	OTR	0	1000	-	-	word/character from R3 → device	5)	4		3	bit 8 = 0; system mode
Read channel address	-	RCA	0	1001	-	-	channel number causing MIDB interrupt		-		3	bits 8-15: 01000000;
							request → R3					system mode
Read interrupt lines	RL	RIL	0	1001	-	-	State interrupt lines → R3		4	2	1	bits 8-15: 00000000;
												system mode
Relative backwards	RB	RB	0	1011	-	0	(P) → displ. → P (branch effective)		4	1	1	bits 5-7: condition
conditional branch							(P) → P (no branch)		3	1	1	bits 8-14: displ.;
								3)				bit 15: n.s.
Relative forward	RF	RF	0	1010	-	0	(P) + displ. → P (branch effective)		4	1	1	bits 5-7: condition
conditional branch						-	(P) → P (no branch)		3	1	1	bits 8-14: displ.;
												bit 15: n.s.
Reset internal	RI 7)	RIT	0	0100	-	1	to clear internal interrupt bits		4	1	1	bits 8, 9, 15: 1,
interrupt												bits 10-14: D.A.; system
												mode
Return from Function	RT	RTN	1	1110	01	0	(R2) + 4 → R2 ; ((R2)) → P ;	*	9	1	3	*reloaded from stack;
							((R2) -2) → PSW					R2 = 1111: system mode
Send status	SS 7)	SST	0	1001	-	-	status character/word from device → R3	5)	4		3	bits 8-9:11; system mode

name (in alphabetical order)	mnemonic		for- mat	OP- code	mode	L/S (0/1) bit	function	condition register	execution time in cycles 6)			remarks
	P850M	P855M							P850M	P855M		
									m.c.	l.c.	m.c.	
Set mode	-	SMD	0	0101	-	-	system mode → user mode	3)	-	1	1	bits 8-15: 00000001
Single left and	-	SLN	0	0111	-	-			-	$(1.5+\frac{n}{2})$	1	bits 8-10: 100;
normalize shift												bits 11-14: R2; bit 15: n.s.
Single left arithmetic	-	SLA	0	0111	-	-		2)	-	$(1.5+\frac{n}{2})$	1	bits 8-10: 000
shift												
Single left	-	SLC	0	0111	-	-			-		*	bits 8-10: 110
circular shift												*(2+½n+½)
Single left	-	SLL	0	0111	-	-		1)	-	$(1.5+\frac{n}{2})$	1	bits 8-10: 010
logical shift												
Single right and	-	SRN	0	0111	-	-		3)	-	$(1.5+\frac{n}{2})$	1	bits 8-10: 101
normalize shift												bits 11-14: R2; bit 15: n.s.
Single right arithmetic	-	SRA	0	0111	-	-			-	$(1.5+\frac{n}{2})$	1	bits 8-10: 001
shift												
Single right circular	-	SRC	0	0111	-	-		1)	-	$(1.5+\frac{n}{2})$	1	bits 8-10: 111
shift												
Single right logical	-	SRL	0	0111	-	-			-	$(1.5+\frac{n}{2})$	1	bits 8-10: 011
shift												
Store character	TC	SC	1	1100	10	1	(R1)r → (M) r/l		8		4	R1 = 1111: system mode
					10	1	(R1)r → (M + (R2)) r/l		8		4	R1 must be ≠ 0
					11	1	(R1)r → ((M)) r/l		10		5	
					11	1	(R1)r → ((M + (R2))) r/l		10		5	
Store character/register	TC	SCR	1	1100	01	1	(R1)r → (R2) r/l		6		3	R1 = 1111: system mode
Store register	TW	ST	1	0000	10	1	(R1) → M	3)	8	1	3	R1 = 1111: system mode
					10	1	(R1) → M + (R2)		8	1	3	
					11	1	(R1) → ((M))		10	1	4	
					11	1	(R1) → ((M + (R2)))		10	1	4	
Store register/register	TW	STR	1	0000	01	1	(R1) → (R2)		6	1	2	R1 = 1111: system mode
					01	1	(R1) → (A15) ; (A15) - 2 → A15		6	1	2	if pointer 128_{10}: stack
												overflow
Subtract constant	SK	SUK	0	0011	-	-	(R3) - K → R3		4	1	1	short; R1 = 1111: system
												mode
Subtract constant	SU	SUKL	1	0011	01	0	(R1) - KL → R1		5		2	long; R1 = 1111: system
												mode
Subtract register/register	SU	SUR	1	0011	00	n.s.	(R1) - (R2) → R1	2)	4	1	1	when l/s bit = 1, R1 must
					01	0	(R1) - ((R2)) → R1		5		2	be ≠ 0; R1 = 1111:
					01	1	(R1) - ((R2)) → (R2)		6		3	system mode
Subtract word	SU	SU	1	0011	10	0	(R1) - (M) → R1		7		3	R1 = 1111: system mode
					10	1	(R1) - (M) → M		8		4	when l/s bit = 1,
					10	0	(R1) - (M + (R2)) → R1		7		3	R3 must be ≠ 0

name (in alphabetical order)	mnemonic		for- mat	OP- code	mode	L/S (0/1) bit	function	condition register	execution time in cycles 6)			remarks
	P850M	P855M							P850M	P855M		
									m.c.	l.c.	m.c.	
					10	1	(R1) - (M + (R2)) ⟶ M + (R2)		8		4	
					11	0	(R1)-((M)) ⟶ R1		9		4	
					11	1	(R1) - ((M)) ⟶ (M)		10		5	
					11	0	(R1) - ((M + (R2))) ⟶ R1		9		4	
					11	1	(R1 - ((M + (R2))) ⟶ (M + (R2))		10		5	
Test mask	I	TM	1	0100	00	1	[(R1) ∧ (R2)] ÷ 0 ⟶ CR	1)	4	1	1	R1 = 1111: system mode
Test not mask	U	TNM	1	0110	00	1	[(R1) ∀ (R2)] ÷ 0 ⟶ CR		4	1	1	R1 = 1111: system mode
Test status	TS 7)	TST	0	1001	-	-	test DCU 'ready state'	5)	4		3	bits 8-9: 10; system mode
Two's complement	C2	C2	1	0011	10	1	0 - (M) ⟶ M	2)	8		4	bits 5-8: 0000
					10	1	0 - (M + (R2)) ⟶ M + (R2)		8		4	
					11	1	0 - ((M)) ⟶ (M)		10		5	
					11	1	0 - ((M + (R2))) ⟶ (M + (R2))		10		5	
Two's complement/register	CR	C2R	1	0011	01	1	0 - ((R2)) ⟶ (R2)	3)	6		3	bits 5-8: 0000
Write interrupt mask	W 7)	WIM	0	1000	-	-	(R3) ⟶ mask register		4	2	1	bits 8-15: 00000000
												system mode
Write mask protection	-	WMP	0	1000	-	-	(R3) ⟶ M.P. key register		-	2	1	bits 8-15; 01000000;
												system mode
Write mask protection no. 2		WM2	0	1000	-	-	(R3) ⟶ M.P. key register		-	2	1	bits 8-15: 11000000; s.m.

NOTES FOR INSTRUCTION SET

Condition register

1) CR=0 if result = 0
   1 if result > 0
   2 if result < 0

2) CR=0 if result = 0
   1 if result > 0
   2 if result < 0
   3 if overflow

3) CR unchar.ged

4) CR=0 if a = b 4 ≠ 0
   1 if a > b 5 ≤
   2 if a < b 6 ≥

5) CR=0 if command accepted
   1 if command not accepted
   3 if device address unknown

6) Execution times in microseconds

	P850M	P855M
mc (memory cycle)	1.6	0.84
lc (logic cycle)	-	0.72

Mnemonics are not recognized by the Assembler.

P850M MEMORY MODULES

0000-03FE: MM0
0400-07FE: MM1
0800-0BFE: MM2
0C00-0FFE: MM3

P855M MEMORY PROTECTION

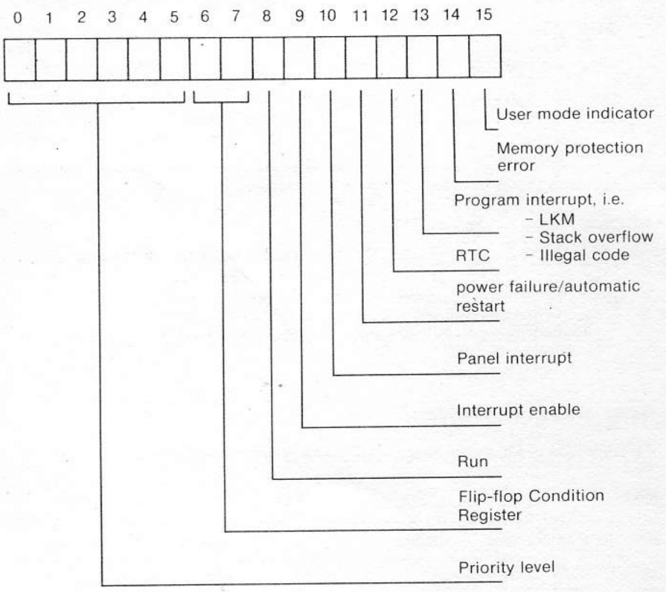
If a certain bit is set, the corresponding memory area is protected.

MK1			MK2		
bit	memory area	module	bit	memory area	module
15	0000 - 07FE	MM0	15	8000 - 87FE	MM4
14	0800 - 0FFE		14	8800 - 8FFE	
13	1000 - 17FE		13	9000 - 97FE	
12	1800 - 1FFE		12	9800 - 9FFE	
11	2000 - 27FE	MM1	11	A000 - A7FE	MM5
10	2800 - 2FFE		10	A800 - AFFE	
9	3000 - 37FE		9	B000 - B7FE	
8	3800 - 3FFE		8	B800 - BFFE	
7	4000 - 47FE	MM2	7	C000 - C7FE	MM6
6	4800 - 4FFE		6	C800 - CFFE	
5	5000 - 57FE		5	D000 - D7FE	
4	5800 - 5FFE		4	D800 - DFFE	
3	6000 - 67FE	MM3	3	E000 - E7FE	MM7
2	6800 - 6FFE		2	E800 - EFFE	
1	7000 - 77FE		1	F000 - F7FE	
0	7800 - 7FFE		0	F800 - FFFE	

DEDICATED MEMORY LOCATIONS

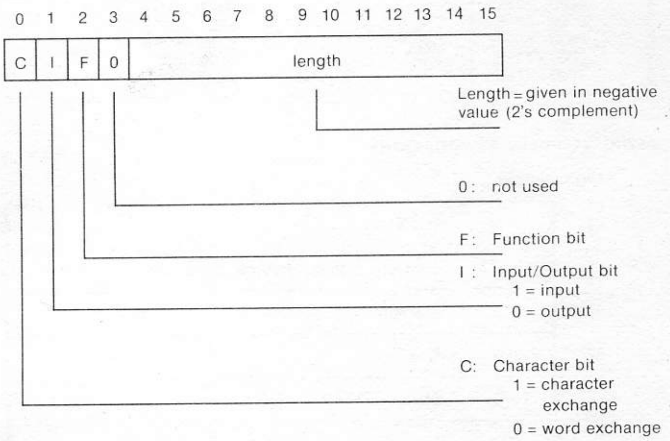
Hexa address	
0 3E	Bootstrap
40 7E	Interrupt routine addresses (levels 0-31)
80	Common interrupt line
82	CVT
84 BE	Multiplex control words
C0 DE	Interrupt routine addresses (levels 32-47)
E0 FE	Stack overflow
100	Overflow ↑ Interrupt location stack area

PROGRAM STATUS WORD P850M/P855M

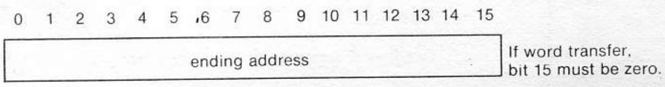


MULTIPLEX CONTROL WORDS

1st word



2nd word



MEMORY CLEAR PROGRAM

This program puts the contents of A2 in the memory location X'40' upto the address given in A3.

address	code	mnemonic P850M	mnemonic P855M
0	822D	TW A2,RA,A3,S	STR A2, A3
2	1B02	SK A3,2	SUK A3,2
4	5C06	RB 4,6	RB(4) *-4
6	207F	HT	HLT

This program puts the contents of A1 in the memory locations /4C upto the address given in A3.

```
P850M                                          P855M
40  024C   LK     A2,/4C                                  LDK      A2,/4C
42  8129   TW     A1,RA,A2,S                   LAB        STR      A1,A2
44  1202   AK     A2,2                                    ADK      A2,2
46  EA0C   CW     A2,RR,A3                                CWR      A2,A3
48  5A08   RB     2,8                                     RB(2)    LAB
4A  207F   DATA   /207F                                   HLT
4C
```

Powers of 16

16^n	n
1	0
16	1
256	2
4 096	3
65 536	4
1 048 576	5
16 777 216	6
268 435 456	7
4 294 967 296	8
68 719 476 736	9
1 099 511 627 776	10
17 592 186 044 416	11
281 474 976 710 656	12
4 503 599 627 370 496	13
72 057 594 037 927 936	14
1 152 921 504 606 846 976	15
18 446 744 073 709 551 616	16

Powers of 2

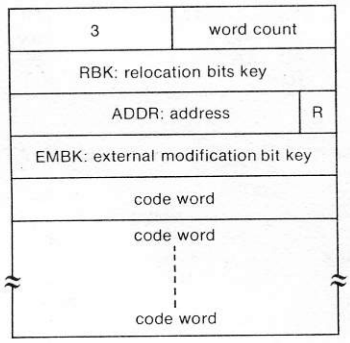
2^n	n
256	8
512	9
1 024	10
2 048	11
4 096	12
8 192	13
16 384	14
32 768	15
65 536	16
131 072	17
262 144	18
524 288	19
1 048 576	20
2 097 152	21
4 194 304	22
8 388 608	23
16 777 216	24

P850M/P855M OBJECT CODE FORMAT

Program identification

IDENT – program name – line feed – X off – carriage return.

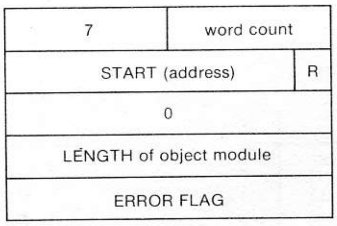
Code cluster



= 0: address is absolute
= 1: address is relative

P855M only

END/START cluster



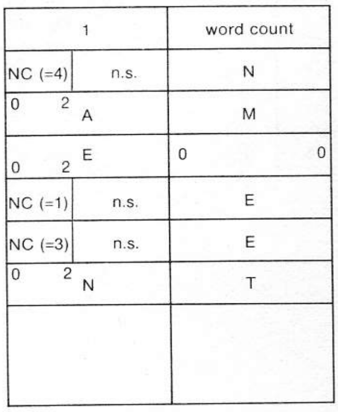
= 0: start address absolute
= 1: start address relative

even number of characters

indicates number of errors in binary

P855M OBJECT CODE FORMAT

Entry point names cluster

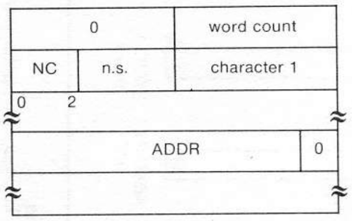


NC = number of characters of entry point name
n.s. = not significant

External reference names cluster

Same format as 'Entry point names cluster'.

BLOCK data cluster

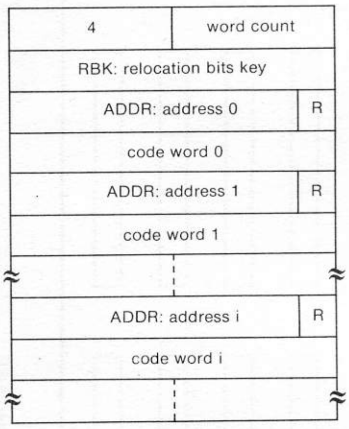


name of labelled COMMON block (up to 6 characters)

absolute data words to be loaded, starting at ADDR in named COMMON block

NC = number of characters of the symbol (3 bits)
n.s. = not significant.

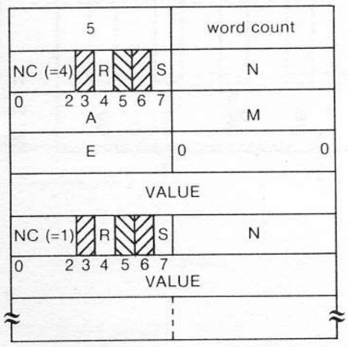
Internal modification cluster



= 0: address absolute
= 1: address relative

maximum number of code words: 16

Entry point definition cluster

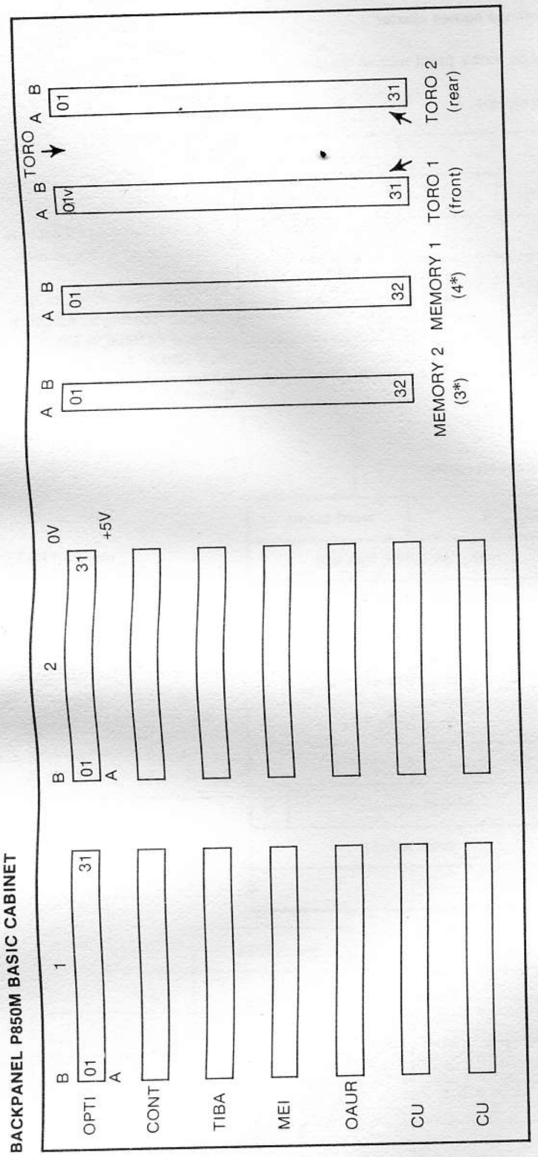


NC = number of characters of entry point name
R = 0: absolute entry point
= 1: relative entry point
S = 1 if entry point name = internal symbol table name.

COMMON length definition cluster

Same format as 'Entry point definition cluster'.

BACKPANEL P850M BASIC CABINET

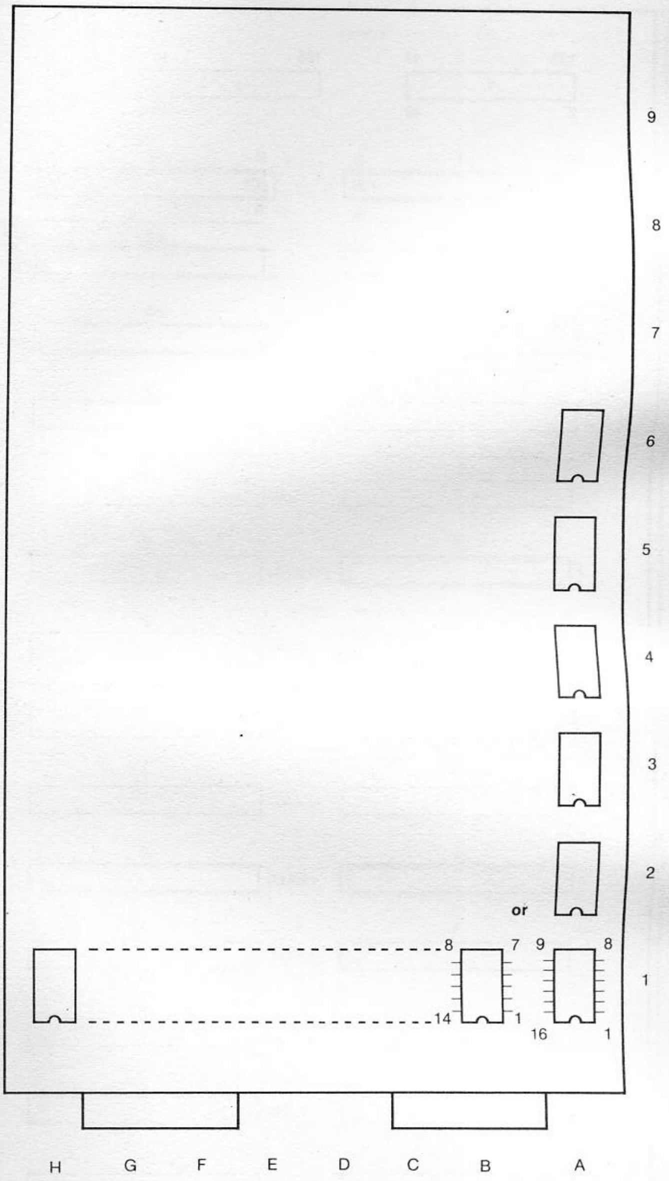


MEI 1 A 25
- pin number (1-31)
- row indicator: A = up – B = down
- 1 = left – 2 = right
- card type

MEM 2 1 A 10
- pin number (1-31)
- connector 1A or 1B
- memory number

* In Extended cabinet

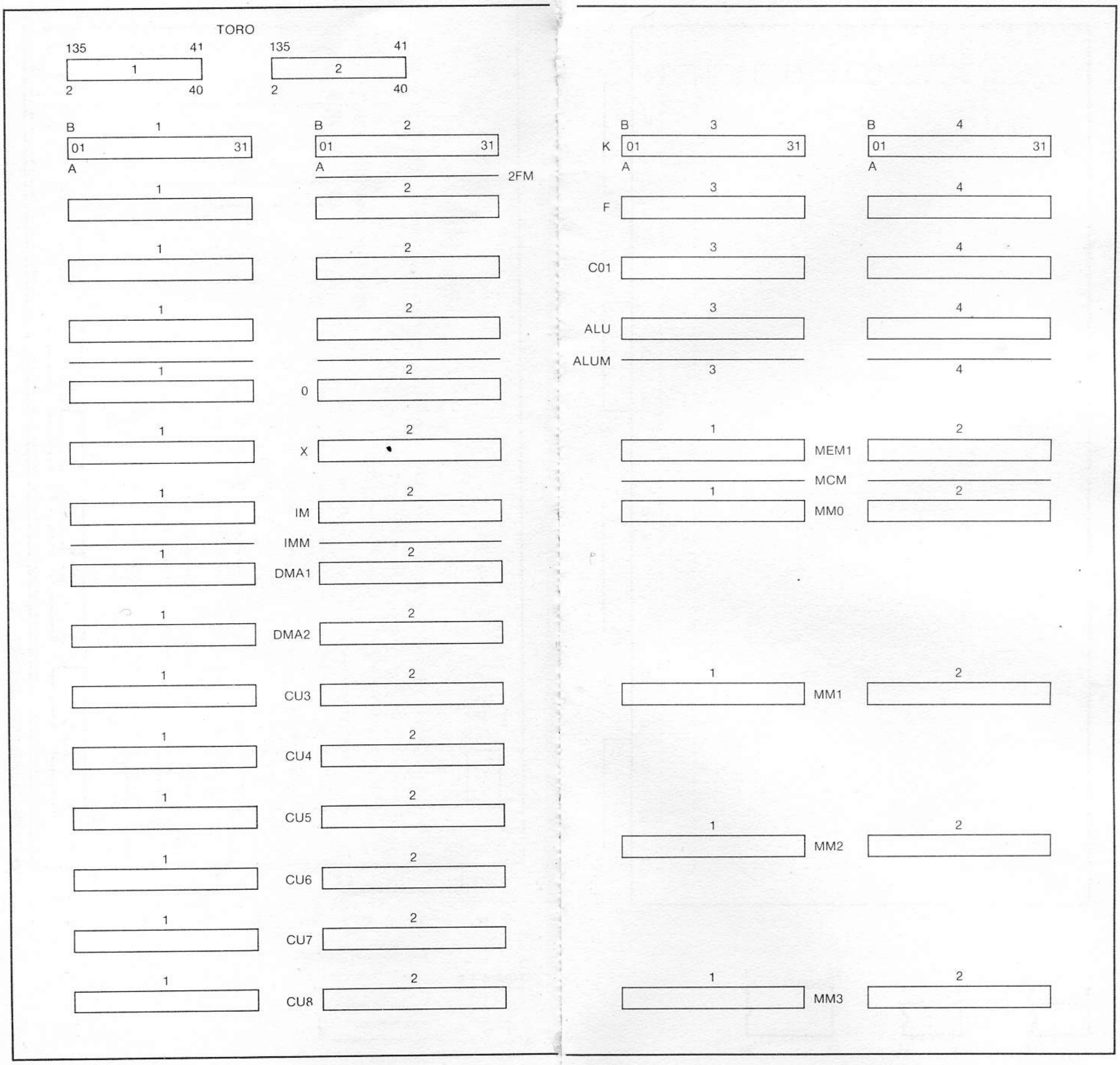
P850M CPU CARD AND P850M/P855M CU CARD



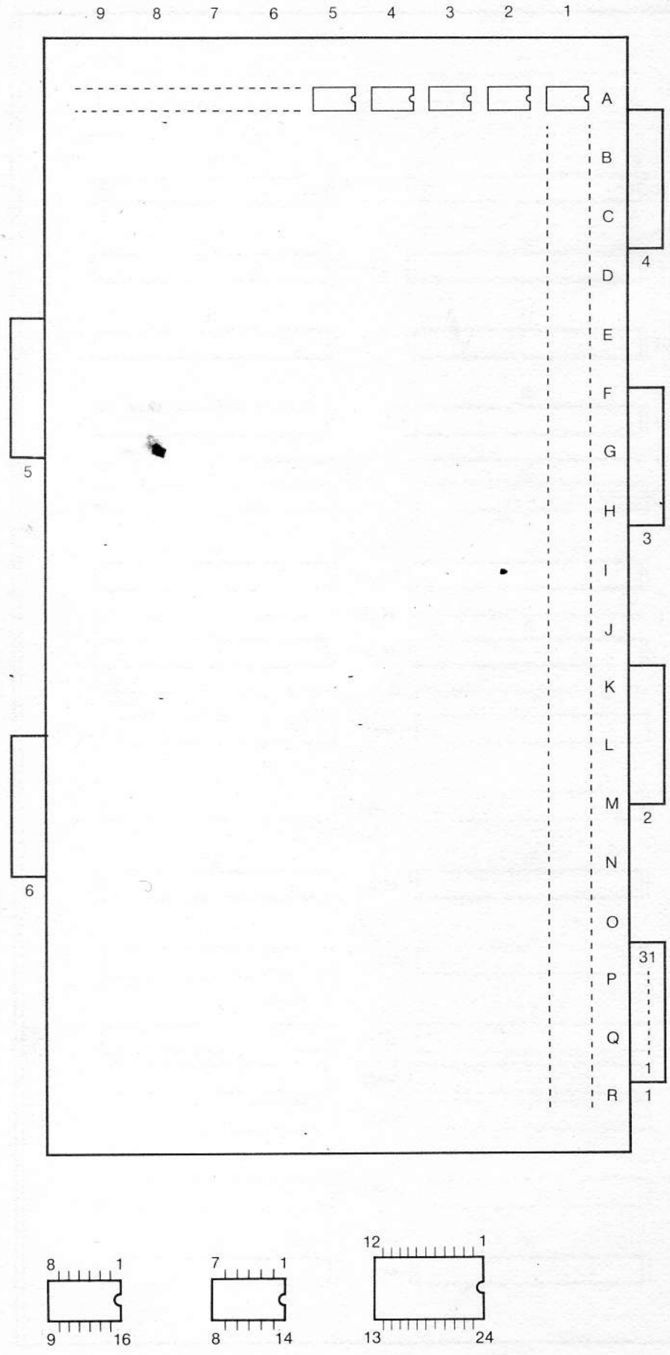
indication of location:

TIBA A 1 8
- i.c. pin number
- row number
- column number
- card type

BACKPANEL P855M BASIC CABINET (8 UNITS)



DOUBLE CARDS (P855M CPU)



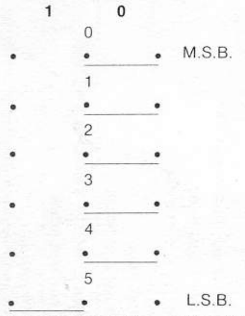
CONTROL UNITS

Status flip-flops

	F0	F1
INCT	0	0
EX	0	1
EXCH	1	1
WST	1	0

CHANGING THE DEVICE ADDRESS

The device address on a C.U. card may be altered by means of jumpers.



Example for device address 0001.

C.U. STATUS WORD CONFIGURATION

Bit	Description	CU									
		ASR	CR	MHD	LP	TP	PTR	CASS Tape	PLOT	FHD	MT
0	–										
1	ready			x				x			x
2	rewind										x
3	tape mark has been read										x
4	EOT							x			
5	on cylinder load point			x							x
6	seek error			x							
	write unable							x			x
7	A or B side							x			
8	Device Address							x		x	
9	Device Address			x				x		x	
10	EOT						x	x			
	tape low					x					
11	Program error			x				x	x	x	x
12	Incorrect length		x	x				x		x	x
	Y limit overpass								x		
13	Parity error							x			
	Data fault		x	x						x	x
14	throughput error	x	x	x			x	x		x	x
15	not operable	x	x	x	x	x	x	x	x	x	x

ASR	I/O typewriter	TP	tape punch	PLOT	plotter
CR	card reader	PTR	punched tape reader	FHD	fixed head disc
MHD	moving head disc	CASS	cassette tape	MT	magnetic tape
LP	line printer				

DEVICES AND RECOGNIZED COMMANDS

PTR	CR	PLOT PTP	ASR	LP	Recognised Command	Format 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Meaning
x			x	x	TST	0 1 0 0 1 R3 1 0 D A	Test status DA = Device Address R3 = register into which status is loaded.
x	x	x	x	x	CIO start (read a card for CR)	0 1 0 0 0 R3 1 1 D A 15 Register indicated in R3	R3 only significant for ASR bit 15 = 1 start input bit 15 = 0 start output
x	x	x	x	x	CIO stop	0 1 0 0 0 R3 1 0 D A	Stop input R3 not significant
x	x		x		INR	0 1 0 0 1 R3 0 X D A	Input to register indicated in R3 field. X: not significant
x	x	x	x	x	SST	0 1 0 0 1 R3 1 1 D A	Send status The status word of the control unit is sent to the register indicated in R3 field.
		x	x	x	OTR	0 1 0 0 0 R3 0 X D A R3 Formats 8 9 10 11 12 13 14 15	Output from the register indicated in R3 field to CU
				x		1 1 X 0 Chan. Nr. 8 9 10 11 12 13 14 15	Advance till given channel
				x		1 1 X 1 Nr. of lines	Skip the given number of lines
				x		0 0 0 0 1 0 1 0	Paper feed and carriage return
				x		0 0 0 0 1 1 0 0	Form feed and carriage return
				x		0 0 0 0 1 1 0 1	Carriage return

DIOS

Recognised Commands	Format 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Meaning
OTR	0 1 5 8 10 11 12 15 0 1 0 0 0 R3 0 0 W A CUA	Output data word R3 = register from which data is read CUA = CU address WA = Word address 00 = word A 01 = word B 10 = word C 11 = word D

INR	0 1 5 8 10 11 12 15
	0 1 0 0 1 R3 0 0 W A CUA
		Input data word
		R3 = register into which data is loaded
		CUA = CU address
		WA = word address (see above)

INR	0 1 5 8 10 11 12 15
	0 1 0 0 1 R3 0 1 W A CUA
		Input word address
		R3 = register into which the address is loaded
		CUA = CU address
		WA = not significant

	Format of R3 0 12 15 A B C D	A one-bit in field 12-15 indicates that the corresponding word is active.

MOVING HEAD DISC CU

Recognised Commands	Format 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	

CIO start general	0 5 8 10 12 15
	0 1 0 0 0 | R3 | 1 1 | D N | CUA
		Start an Operation
		DN = Device number 00 or 01
		CUA = Control Unit address
		R3 = specifies the command

Seek command	**R3 Formats** 0 4 5 13 14 15 n.s. | D | S | 1 0	move the head to another cylinder D = difference between cylinder number S = direction of motion 1 = forward 0 = reverse
Seek to zero command	0 14 15 | 1 1	move heads to cylinder 00
Write a sector	0 9 14 15 | S A | 0 1	Start a write operation on the sector address on the BOU lines on the cylinder of the moment. SA = Sector Address 0-31
Read a sector	0 9 14 15 | S A | 0 0	Start a read operation on the sector which is addressed on the BOU lines on the cylinder of the moment SA = Sector Address 0-31

MOVING HEAD DISC CU (CONTINUED)

Recognised Commands	Format 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Meaning

SST	0 1 5 8 12 15
	0 1 0 0 1 | R3 | 1 1 | 0 | X | CUA
		Exchange status between CU and CPU
		CUA = CU address
		X = not significant
		R3 = specifies register into which status is loaded (see status word).

CIO Stop	0 1 5 8 10 12 15
	0 1 0 0 0 | R3 | 1 0 | DN | CUA
		Stop data transfer
		R3 = not significant
		DN = Device number
		CUA = CU address

TST	0 1 5 8 10 12 15 0 1 0 0 1 | R3 | 1 0 | 0 | X | CUA	Test status of CU CUA = CU Address R3 = register into which status is loaded.

CASSETTE TAPE CU

Recognised Commands	Format 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Meaning

TST	0 1 5 8 10 12 15
	| | | R3 | 1 0 | D N | CUA |
		Test status
		CUA = CU address
		DN = Device number (00-11)
		R3 = register into which status is loaded

CIO Start	0 1 5 8 10 12 15
	| 0 | 1 0 0 0 | R3 | 1 1 | D N | CUA |
		Start command
		CUA = CU address
		DN = Device number
		R3 = register specifying the command.

R3 contents

bit number 12	13	14	15	Command
0	0	0	0	Lock/unlock
0	0	0	1	Erase forward
0	0	1	0	Backwards space block
0	0	1	1	Forward space block
0	1	0	1	Write a block forward
0	1	1	1	Read a block forward
1	0	0	0	Rewind at fast speed
1	0	1	0	Search tape mark backwards
1	0	1	1	Search tape mark forward

CASSETTE TAPE CU (CONTINUED)

Recognised Commands	Format 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Meaning

CIO Stop	0 1 5 8 10 12 15
	| 0 | 1 0 0 0 | R3 | 1 0 | X X | CUA |
		Stop transfer
		CUA = CU address
		DN = not significant
		R3 = register specifying the command
		X = not significant

INR	0 1 5 8 10 12 15
	| 0 | 1 0 0 1 | R3 | 0 X | 0 X | CUA |
		Input to register
		DN = not significant
		R3 = register into which data is loaded
		X = not significant

OTR	0 1 5 8 10 12 15
	| 0 | 1 0 0 0 | R3 | 0 X | 0 X | CUA |
		Output from register
		DN = not significant
		X = not significant
		R3 = register from which the data is read.

SST	0 1 5 8 10 12 15 | 0 | 1 0 0 1 | R3 | 1 1 | 0 X | CUA |	Send Status See status word configuration on page 37 For explanation of field, see above

FIXED HEAD DISC CU

Recognised Commands	Format 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Meaning

TST	0 1 5 8 10 12 15
	0 1 0 0 1 R3 1 0 DN CUA
		Test status of CU
		DN = not significant
		CUA = CU address
		R3 = register into which status is loaded.

Write one sector	0 1 5 8 10 12 15
	0 1 0 0 0 R3 1 1 DN CUA
		Start a write operation on the sector on the BOU-lines
		DN = Device number 00, 01, 10 or 11
		CUA = CU address
		R3 = register specifying the command

	0 3 13 14 15 X X X sector address 0-1023 X 0	R3 format

Read one sector	0 1 5 8 10 12 15
	0 1 0 0 0 R3 1 1 DN CUA
		Start a write operation on the sector on the BOU-lines.
		DN = Device number 00, 01, 10 or 11
		CUA = CU address
		R3 = register specifying the command.

	0 3 13 14 15 X X X sector address 0-1023 X 1	R3 format

FIXED HEAD DISC CU (CONTINUED)

Recognised Commands	Format 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Meaning

INR	0 1 5 8 10 12 15
	0 1 0 0 1 R1 0 X X X CUA
		Input data word
		X = not significant
		CUA = CU address
		R1 = register into which the word is loaded.

OTR	0 1 5 8 10 12 15
	0 1 0 0 0 R1 0 X X X CUA
		Output data word
		X = not significant
		CUA =CU address
		R1 = register from which the word is read.

SST	0 1 5 8 10 12 15
	0 1 0 0 1 R3 1 1 X X CUA
		Exchange status between CU and CPU
		X = not significant
		CUA = CU address
		R3 = register into which status word is loaded (see status word).

CIO Stop	0 1 5 8 10 12 15
	0 1 0 0 0 X X 1 0 X X CUA
		Stop data transfer
		X =not significant
		CUA = CU address

MAGNETIC TAPE CU

Recognised Commands	Format 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Meaning

TST	0 1 5 8 10 12 15
	0 | 1 0 0 1 | R3 | 1 0 | DN | CUA
		Test status
		CUA = CU address
		DN = Device number
		R3 = register into which status word is loaded

CIO Start	0 1 5 8 10 12 15
	0 | 1 0 0 0 | R3 | 1 1 | DN | CUA
		Start command
		CUA = CU address
		DN = Device number 00, 01, 10 or 11
		R3 = register specifying the command.

R3 contents

bit number 10	11	12	13	14	15	Command
1	X	X	0	X	X	Off-line
1	X	X	1	X	X	Rewind
0	0	0	0	1	1	Write
0	0	1	0	1	1	Write-edit
0	1	0	0	0	1	Write file mark
0	1	1	0	0	1	Erase gap
0	0	0	0	1	0	Read
0	1	0	0	0	0	Search tape mark forward
0	1	0	1	0	0	Search tape mark backwards
0	0	0	0	0	0	Forward space block
0	0	0	1	0	0	Backwards space block
0	0	1	1	0	0	Backwards space block edit

MAGNETIC TAPE CU (CONTINUED)

Recognised Commands	Format 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Meaning

CIO stop	0 1 5 8 10 12 15
	0 | 1 0 0 0 | X X X | 1 0 | DN | CUA
		Stop transfer
		CUA = CU address
		DN = Device number 00, 01, 10 or 11
		X = not significant

SST	0 1 5 8 10 12 15
	0 | 1 0 0 1 | R3 | 1 1 | X X | CUA
		Send status
		CUA = CU address
		R3 = register into which status word is loaded
		X = not significant

DMA-channel

Recognised Commands	Format 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Meaning
CIO Start	0 1 5 8 10 15 0 1 0 0 0 X X X 1 1 DA 1)	Initialize a DMAC exchange DA= DMAC address
OTR write length	0 1 5 8 10 15 0 1 0 0 0 R1 0 0 DA	Load DMAC length register DA= DMAC address R1= register from which length and functions are read.

	0 1 2 3 length format 15
	X I F 2's complement of length
	1)
		I = input bit
		F = function bit
		Length, expressed in characters, must be an even number.

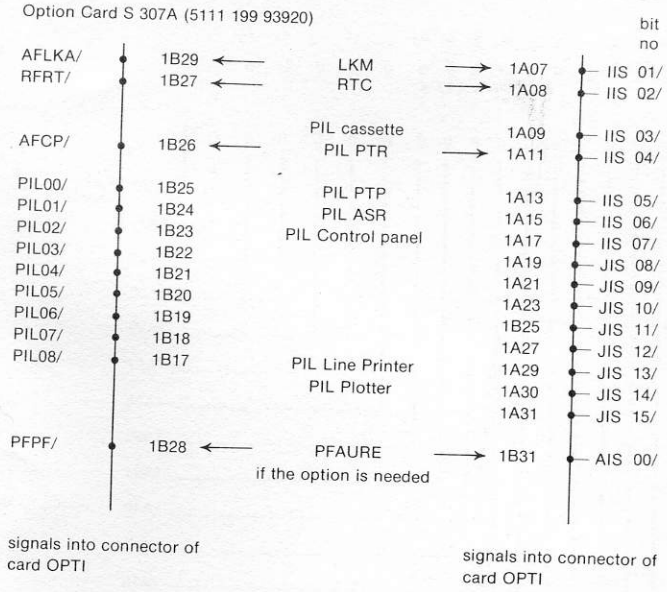
OTR write address	0 1 5 8 10 15 0 1 0 0 0 R1 0 1 DA	DA= DMAC address R1 = register from which address is read
	0 14 15 starting address X1)	address format
CIO Stop	0 1 5 8 15 0 1 0 0 0 X X X 1 0 DA 1)	De-activate DMAC DA= DMAC address

INR read length	0 1 5 8 10 15
	0 1 0 0 1 R1 0 X DA
	1)
		Transfer DMAC length into R1
		DA= DMAC address
		R1 = register into which DMAC length is loaded

1) X = not significant

P850M OPTION CARD (OPTI) STANDARO INTERRUPT CABLING

Option Card S 307A (5111 199 93920)

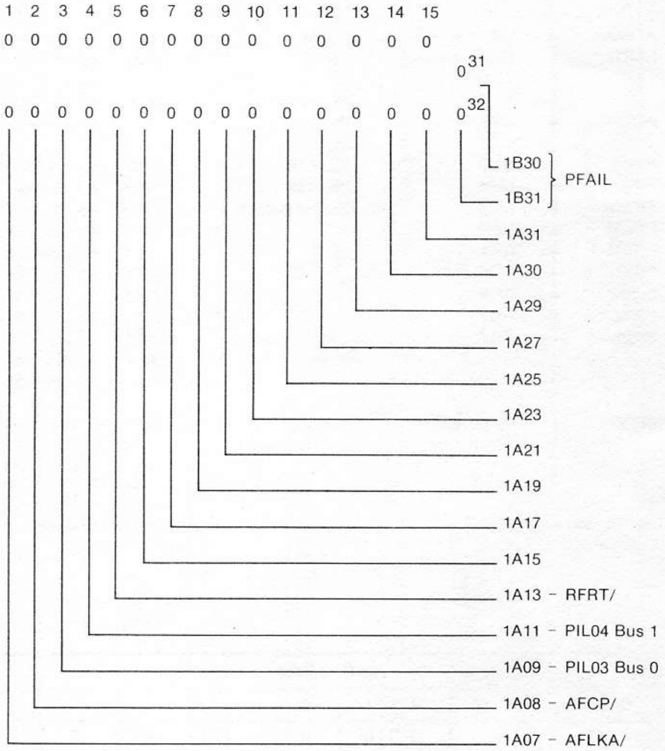


BAD - LINES

2	3	4	5	0 1	0 1	0 1	0 1
				0 0	0 1	1 0	1 1
0	0	0	0	CPU	ASR1	PTR1	PTP1
0	0	0	1	FHD	FHD	FHD	FHD
0	0	1	0	MHD	MHD	PTR2	PTP2
0	0	1	1	MHD	MHD	MHD	MHD
0	1	0	0	MT	MT	MT	MT
0	1	0	1	CR	ASR2	*	*
0	1	1	0	CT	CT	CT	CT
0	1	1	1	*	*	*	*
1	0	0	0	*	*	*	*
1	0	0	1	*	*	*	*
1	0	1	0	DIOC	*	*	*
1	0	1	1	*	*	*	*
1	1	0	0	*	*	*	*
1	1	0	1	LP	*	*	*
1	1	1	0	PLT	*	*	*
1	1	1	1	ADAP	DMA	MIDB	RTC

* Not allocated

P850M OPTION CARD (OPTI) STANDARD INTERRUPT CABLING (up to P850M-26)



P855M PIN BOARD CABLING

Interrupt mask bits	Pin location	Dedicated memory address	Standard connection
BMS KIL 00/	F 1A11		reserved
BMS KIL 01/	F 1A14		plotter
BMS KIL 02/	F 1A16		line printer
BMS KIL 03/	F 1B16		MHD2
BMS KIL 04/	F 1B25		MHD1
BMS KIL 05/	F 1B17		FHD
BMS KIL 06/	F 1B18		PEC tape
BMS KIL 07/	F 1B22		ASR2
BMS KIL 08/	F 1A23		ASR1
BMS KIL 09/	F 1B28		PTP2
BMS KIL 10/	F 1A19		PTP1
BMS KIL 11/	F 1A20		Cass. tape
BMS KIL 12/	F 2B10		PTR2
BMS KIL 13/	F 1A28		PTR1
BMS KIL 14/	F 1A31		CR
BMS KIL 15/	F 2B09		reserved

P855M

Standard interrupt signals	Pin location	Dedicated memory address	Standard connection
CFPF/	IM 1A16		
CFEM/	IM 1A17		
CFRT/	IM 2A27		
CFPI/	F 3B31		
CFCP/	F 4B04		
CFMIDB/	X 1A06		
PIL	CU 1A15		
CISMSK/	F 2A02		

Standard interrupt levels	Pin location	Dedicated memory address	Standard connection
BIS 00/	F 2B12	/0040	PFAURE
BIS 01/	F 2B11	/0042	LKM
BIS 02/	F 2A12	/0044	RTC
BIS 03/	F 2A11	/0046	MIDB
BIS 04/	F 2A10	/0048	reserved
BIS 05/	F 2A09	/004A	common int. line
BIS 06/	F 2A08	/004C	memory protect
BIS 07/	F 2A07	/004E	CP

Optional interrupt levels	Pin location	Address on header connector	Dedicated memory address	Standard connection
JIS 08/	IM 1A02	6A07	/0050	ASR
JIS 09/	IM 1A03	6A08	/0052	disc
JIS 10/	IM 1A04	6A09	/0054	line printer
JIS 11/	IM 1A05	6A10	/0056	card reader
JIS 12/	IM 1A06	6A11	/0058	magtape
JIS 13/	IM 1A07	6A12	/005A	
JIS 14/	IM 1A08	6A13	/005C	
JIS 15/	IM 1A09	6A14	/005E	
JIS 16/	IM 1A10	6A15	/0060	
JIS 17/	IM 1A11	6A16	/0062	

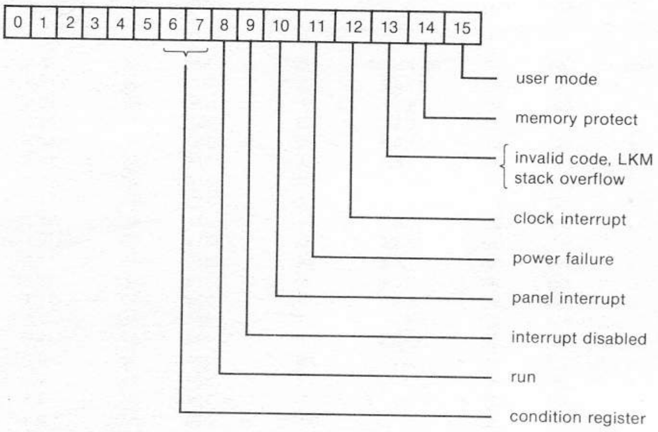
Optional interrupt levels	Pin location	Dedicated memory address	Connected to
KIS 18/	IM 2A05	/0064	
KIS 19/	IM 2A04	/0066	
KIS 20/	IM 2A03	/0068	
KIS 21/	IM 2A02	/006A	
KIS 22/	IM 2A01	/006C	
KIS 23/	IM 2B01	/006E	
KIS 24/	IM 2B02	/0070	
KIS 25/	IM 2B03	/0072	
KIS 26/	IM 2B04	/0074	
KIS 27/	IM 2B05	/0076	
KIS 28C/	IM 5A23	/0078	
KIS 29C/	IM 5A22	/007A	
KIS 30C/	IM 5A21	/007C	
KIS 31C/	IM 5A20	/007E	
LIS 32C/	IM 5A19	/00C0	
LIS 33C/	IM 5A18	/00C2	
LIS 34C/	IM 5A17	/00C4	
LIS 35C/	IM 5A16	/00C6	
LIS 36C/	IM 5A15	/00C8	
LIS 37C/	IM 5A14	/00CA	
LIS 38C/	IM 5A13	/00CC	
LIS 39C/	IM 5A12	/00CE	
LIS 40C/	IM 5A11	/00D0	
LIS 41C/	IM 5A10	/00D2	
LIS 42C/	IM 5A09	/00D4	
LIS 43C/	IM 5A08	/00D6	
LIS 44C/	IM 5A07	/00D8	
LIS 45C/	IM 5A06	/00DA	
LIS 46C/	IM 5A05	/00DC	
LIS 47C/	IM 5A04	/00DE	

P855M BREAK LEVELS

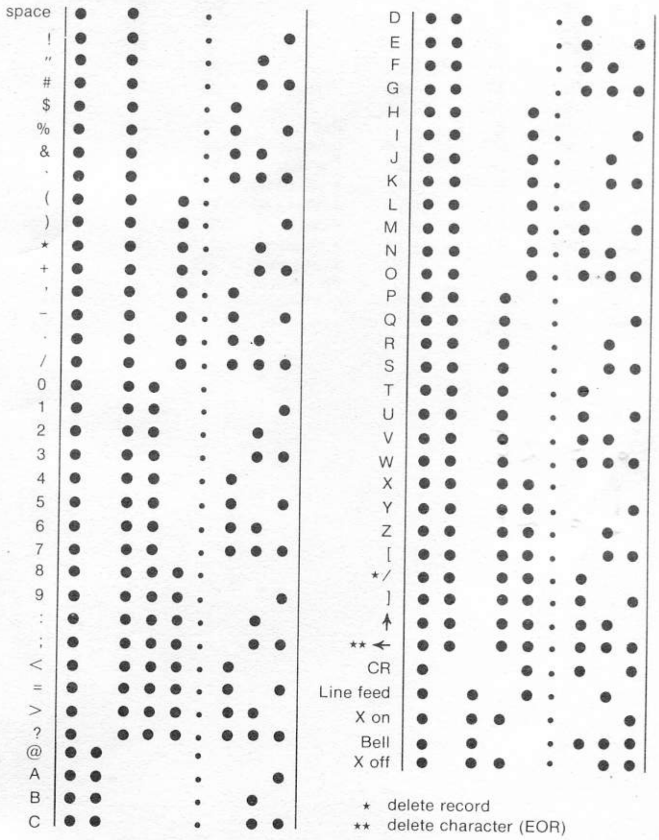
Level number	Pin location	Address on header connector E	Dedicated Multiplex address	Connected to
XBS 01	X 1A02	X 5A11	/0084 - /0086	
XBS 02	X 1B02	X 5A12	/0088 - /008A	
XBS 03	X 1B03	X 5A13	/008C - /008E	
XBS 04	X 1B04	X 5A15	/0090 - /0092	
WBS 05	X 1B05	X 5A17	/0094 - /0096	
WBS 06	X 1B06	X 5A14	/0098 - /009A	
WBS 07	X 1B07	X 5A16	/009C - /009E	
WBS 08	X 2B26	X 5A03	/00A0 - /00A2	
WBS 09	X 2B27	X 5A04	/00A4 - /00A6	
WBS 10	X 2B28	X 5A05	/00A8 - /00AA	
WBS 11	X 2B29	X 5A06	/00AC - /00AE	
WBS 12	X 2B30	X 5A07	/00B0 - /00B2	
WBS 13	X 1A05	X 5A08	/00B4 - /00B6	
WBS 14	X 1A04	X 5A09	/00B8 - /00BA	
WBS 15	X 1A03	X 5A10	/00BC - /00BE	

Note
Break level = device address.
Break is generated on c.u. 1A16.

FORMAT OF CPU STATUS WORD



ASCII CODE



* delete record
** delete character (EOR)

char.	*ASCII octal*	*Intern Hexa*	*char. set punch comb.*	*char.*	*ASCII octal*	*Intern Hexa*	*char. set punch comb.*
space	240	20	on punch	D	304	44	12,4
!	241	21	11,8,2	E	305	45	12,5
"	242	22	8,7	F	306	46	12,6
#	243	23	8,3	G	307	47	12,7
$	244	24	11,8,3	H	310	48	12,8
%	245	25	0,8,4	I	311	49	12,9
&	246	26	12	J	312	4A	11,1
'	247	27	8,5	K	313	4B	11,2
(	250	28	12,8,5	L	314	4C	11,3
)	251	29	11,8,5	M	315	4D	11,4
*	252	2A	11,8,4	N	316	4E	11,5
+	253	2B	12,8,6	O	317	4F	11,6
,	254	2C	0,8,3	P	320	50	11,7
−	255	2D	11	Q	321	51	11,8
.	256	2E	12,8,3	R	322	52	11,9
/	257	2F	0,1	S	323	53	0,2
0	260	30	0	T	324	54	0,3
1	261	31	1	U	325	55	0,4
2	262	32	2	V	326	56	0,5
3	263	33	3	W	327	57	0,6
4	264	34	4	X	330	58	0,7
5	265	35	5	Y	331	59	0,8
6	266	36	6	Z	332	5A	0,9
7	267	37	7	[	333	5B	
8	270	38	8	\	334	5C	
9	271	39	9	]	335	5D	
:	272	3A	8,2	↑	336	5E	
;	273	3B	11,8,6	←	337	5F	
<	274	3C	12,8,4				
=	275	3D	8,6	Bell	207	07	
>	276	3E	0,8,6	Linefeed	212	0A	
?	277	3F	0,8,7	Car.Ret.	215	0D	
@	300	40	8,4	X on reader	221	11	
A	301	41	12,1	X off reader	223	13	
B	302	42	12,2	Rubout	377	7F	
C	303	43	12,3	X on punch	222	12	
				X off punch	224	14	
				FF		0C	

• = 2E

! = 21