



**PLEASE CHECK FOR CHANGE INFORMATION
AT THE REAR OF THIS MANUAL.**

This manual supports the following TEKTRONIX products:

Product	Factory Installed Option	Field Installed Option
8550	1A	8300A01

8550
**MICROCOMPUTER
DEVELOPMENT LAB**
8080A/8085A
**ASSEMBLER SPECIFICS
USERS MANUAL**

Tektronix, Inc.
P.O. Box 500
Beaverton, Oregon 97077

Serial Number _____

LIMITED RIGHTS LEGEND

Software License No. _____

Contractor: Tektronix, Inc.

Explanation of Limited Rights Data Identification Method

Used: Entire document subject to limited rights.


Those portions of this technical data indicated as limited rights data shall not, without the written permission of the above Tektronix, be either (a) used, released or disclosed in whole or in part outside the Customer, (b) used in whole or in part by the Customer for manufacture or, in the case of computer software documentation, for preparing the same or similar computer software, or (c) used by a party other than the Customer, except for: (i) emergency repair or overhaul work only, by or for the Customer, where the item or process concerned is not otherwise reasonably available to enable timely performance of the work, provided that the release or disclosure hereof outside the Customer shall be made subject to a prohibition against further use, release or disclosure; or (ii) release to a foreign government, as the interest of the United States may require, only for information or evaluation within such government or for emergency repair or overhaul work by or for such government under the conditions of (i) above. This legend, together with the indications of the portions of this data which are subject to such limitations shall be included on any reproduction hereof which includes any part of the portions subject to such limitations.

RESTRICTED RIGHTS IN SOFTWARE

The software described in this document is licensed software and subject to **restricted rights**. The software may be used with the computer for which or with which it was acquired. The software may be used with a backup computer if the computer for which or with which it was acquired is inoperative. The software may be copied for archive or backup purposes. The software may be modified or combined with other software, subject to the provision that those portions of the derivative software incorporating restricted rights software are subject to the same restricted rights.

Copyright © 1980 Tektronix, Inc. All rights reserved. Contents of this publication may not be reproduced in any form without the written permission of Tektronix, Inc.

Products of Tektronix, Inc. and its subsidiaries are covered by U.S. and foreign patents and/or pending patents.

TEKTRONIX, TEK, SCOPE-MOBILE, and  are registered trademarks of Tektronix, Inc. TELEQUIPMENT is a registered trademark of Tektronix U.K. Limited.

Printed in U.S.A. Specification and price change privileges are reserved.

Section 12A

8080A/8085A ASSEMBLER SPECIFICS

	Page
Demonstration Run	12A-1
Architecture	12A-1
16-Bit Registers.....	12A-1
8-Bit Registers.....	12A-2
Status Register	12A-3
Addressing Modes	12A-4
Notational Conventions	12A-6
Instruction Set	12A-9
Jumps	12A-9
Stack Operations.....	12A-10
Subroutine Handling: Calls	12A-10
Subroutine Handling: Returns	12A-11
Data Transfer	12A-12
Arithmetic Operations	12A-13
Logical Operations	12A-15
Miscellaneous (IN, OUT, HLT, NOP, Interrupt Instructions)	12A-17
Instructions Available On 8085A Only	12A-18
Reserved Words	12A-19
Page Size	12A-19
Error Messages	12A-20
Irregularities	12A-20

TABLES

Table No.		
12A-1	Register Organization.....	12A-1
12A-2	Register Notation	12A-8

FIGURES

Fig. No.		
12A-1	The 8080A/8085A Status Register	12A-3
12A-2	Stack Implementation.....	12A-5

INDEX OF 8080A/8085A MNEMONICS

	Page
ACI—Add immediate byte to accumulator with carry	12A-13
ADC—Add source to accumulator with carry	12A-13
ADD—Add source to accumulator with no carry	12A-13
ADI—Add immediate byte to accumulator with no carry	12A-13
ANA—AND accumulator with source	12A-15
ANI—AND accumulator with immediate byte	12A-15
CALL—Call subroutine unconditionally	12A-10
CC—Call if carry set	12A-10
CM—Call if minus	12A-10
CMA—Complement accumulator	12A-15
CMC—Complement carry flag	12A-16
CMP—Compare accumulator with source	12A-15
CNC—Call if carry not set	12A-10
CNZ—Call if not zero	12A-10
CP—Call if plus	12A-10
CPE—Call if parity even	12A-11
CPI—Compare accumulator with immediate byte	12A-16
CPO—Call if parity odd	12A-11
CZ—Call if zero	12A-10
DAA—Decimal adjust accumulator	12A-14
DAD—Dual addition of register pairs	12A-13
DCR—Decrement 8-bit register	12A-14
DCX—Decrement register pair	12A-14
DI—Disable interrupts	12A-17
EI—Enable interrupts	12A-17
HLT—Halt processor	12A-17
IN—Send in byte from device to accumulator	12A-17
INR—Increment 8-bit register	12A-14
INX—Increment register pair	12A-14
JC—Jump if carry set	12A-9
JM—Jump if minus	12A-9
JMP—Jump unconditionally	12A-9
JNC—Jump if carry not set	12A-9
JNZ—Jump if not zero	12A-9

	Page
JP—Jump if plus	12A-9
JPE—Jump if parity even	12A-9
JPO—Jump if parity odd	12A-9
JZ—Jump if zero	12A-9
LDA—Load accumulator with 8-bit register	12A-12
LDAX—Load accumulator with contents of register pair.....	12A-12
LHLD—Load H register pair with consecutive memory bytes	12A-12
LXI—Load register pair with immediate address	12A-12
MOV—Move source to destination.....	12A-12
MVI—Move immediate source to destination.....	12A-12
NOP—No operation	12A-17
ORA—OR accumulator with source.....	12A-15
ORI—OR accumulator with immediate byte	12A-15
OUT—Send byte from accumulator to device	12A-17
PCHL—Load program counter with H register pair.....	12A-9
POP—Send top of stack to destination	12A-10
PUSH—Send source to top of stack	12A-10
RAL—Rotate accumulator left through carry	12A-16
RAR—Rotate accumulator right through carry.....	12A-16
RC—Return if carry set	12A-11
RET—Return unconditionally.....	12A-11
RIM—Read interrupt mask	12A-18
RLC—Rotate accumulator left with carry affected	12A-17
RM—Return if minus	12A-11
RNC—Return if carry not set.....	12A-11
RNZ—Return if not zero	12A-11
RP—Return if plus.....	12A-11
RPE—Return if parity even.....	12A-11
RPO—Return if parity odd.....	12A-11
RRC—Rotate accumulator right with carry affected	12A-16
RST—Restart at vectored subroutine.....	12A-17
RZ—Return if zero	12A-11
SBB—Subtract source from accumulator with borrow	12A-14
SBI—Subtract immediate byte from accumulator with borrow.....	12A-14
SHLD—Store H register pair in consecutive memory bytes	12A-12

	Page
SIM—Set interrupt mask	12A-18
SPHL—Load stack pointer with H register pair	12A-10
STA—Store accumulator at destination	12A-12
STAX—Store accumulator at register pair destination	12A-13
STC—Set carry flag to one	12A-16
SUB—Subtract source from accumulator without borrow	12A-13
SUI—Subtract immediate byte from accumulator with no borrow.....	12A-14
XCHG—Exchange register pairs	12A-13
XRA—Exclusive OR accumulator with source	12A-15
XRI—Exclusive OR accumulator with immediate byte	12A-15
XTHL—Exchange top of stack with H register pair.....	12A-10

Section 12A

8080A/8085A ASSEMBLER SPECIFICS

DEMONSTRATION RUN

This Assembler Specifics section is intended as a supplement to the 8300AXX Assembler Users Manual. The Demonstration Run for the 8080A/8085A can be found in the Learning Guide of that manual.

ARCHITECTURE

The 8080A/8085A is an 8-bit microprocessor with a 64K addressable memory space. Its architecture includes a control unit, an 8-bit bidirectional data bus, an 8-bit accumulator served by an Arithmetic Logic Unit (ALU), and a versatile set of internal 8-bit and 16-bit registers. The following paragraphs discuss those registers accessible to the user.

16-Bit Registers

There are three types of 16-bit registers: simple 16-bit register, register pair, and pseudo-register pair. Note that the generality of the 16-bit registers is limited; there are no instructions that apply to all six 16-bit registers. Table 12A-1 shows the organization of the 16-bit registers and their constituent 8-bit registers.

Table 12A-1
8080A/8085A Register Organization

Name	16-Bit Register Type	High-Order Byte	Low-Order Byte
PSW	Pseudo-Register Pair	a	status
B	Register Pair	b	c
D	Register Pair	c	d
H	Register Pair	h	l
SP	Simple 16-Bit		
PC	Simple 16-Bit		

In Table 12A-1, each of the first four 16-bit registers (**PSW**, **B**, **D**, and **H**) contains two separately addressable 8-bit registers. These 8-bit registers are labeled with lowercase letters. This 8-bit/16-bit duality provides for simple and efficient processing of addresses and their contents.

Simple 16-bit Registers: PC, SP

The program counter (**PC**) and the stack pointer (**SP**) are simple 16-bit registers and have no directly addressable single bytes. Only in a limited number of instructions may either of these registers be treated as a register pair.

Register Pairs: B, D, and H

The three register pairs are designated with the uppercase letters **B**, **D**, and **H**. They are **not** to be confused with the corresponding 8-bit registers **b**, **d**, and **h**. This uppercase/lowercase convention is used here for readability. In actual assembly language programming uppercase letters are used to designate both 8-bit and 16-bit registers. The instruction itself determines whether the uppercase letter represents an 8-bit or a 16-bit register.

Among the register pairs, the **H** register pair has additional features: it is uniquely supported by the **PCHL**, **SPHL**, **XTHL**, **XCHG**, **LHLD**, and **SHLD** instructions. The **H** register pair also holds the address of the memory byte **M**.

Pseudo-Register Pairs: PSW

The Program Status Word (**PSW**) is classified as a pseudo-register pair because the accumulator and status register are distinct and separate 8-bit registers and have no logical connection. However, the **POP** and **PUSH** instructions treat the accumulator and status registers as if they are a register pair.

The 8-Bit Registers

The individual 8-bit registers all exist as the high-order and low-order bytes of 16-bit registers. A discussion of the 8-bit registers follows.

The Accumulator

The accumulator (**a**) is uniquely supported by logical and arithmetic instructions because of its direct connection to the Arithmetic Logic Unit (**ALU**). The accumulator is also the first byte of the Program Status Word (**PSW**).

The Simple 8-Bit Registers

The simple 8-bit registers (**b**, **c**, **d**, **e**, **h**, and **l**) behave alike when addressed as 8-bit registers. The accumulator may be considered a member of this group although it is supported by additional arithmetic and logical instructions. The status register does not belong to this group and is discussed separately.

The Status Register

The 8-bit status register contains a sequence of flags which may be set by conditions generated from instruction operations. Figure 12A-1 shows the position of these flags within the status register.

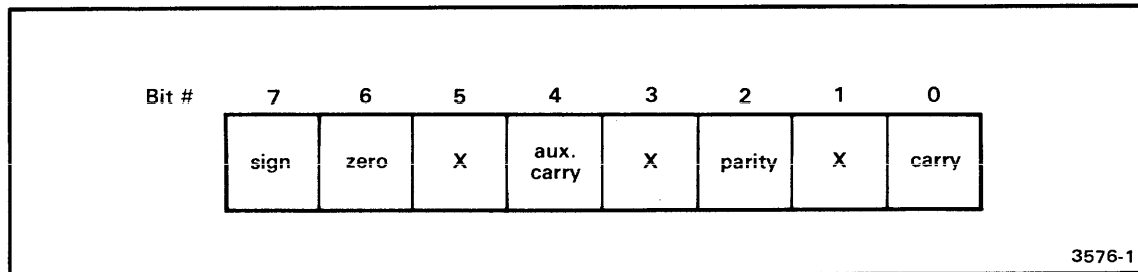


Fig. 12A-1. The 8080A/8085A Status Register and its flags.

- Bit 0 The **carry flag** is set to 1 if either a carry **or borrow** occurs out of the accumulator during instruction execution.
- Bits 1,3,5 The values of bits 1, 3, and 5 vary depending on chip manufacturer and on whether the chip is an 8080A or 8085A. These bits are **not** written to during a POP PSW instruction.
- Bit 2 The **parity flag** is set to 1 if parity is even, and cleared to 0 if parity is odd. Parity is calculated from the sum of 1-valued bits in the accumulator. For instance, if the binary value in the accumulator is 00000011, then the sum of 1-valued bits is 2 and parity even.
- Bit 4 The **auxiliary carry flag** is used with Binary Coded Decimal (BCD) arithmetic. A BCD number requires only a 4-bit representation; thus two BCD digits may be stored in a byte. The auxiliary carry flag is set to 1 if either a carry **or** a borrow occurs between BCD numbers within that byte.
- Bit 6 The **zero flag** is set to 1 when an operation results in 0.
- Bit 7 The **sign flag** is set to 1 when an operation results in a two's-complement negative number with the seventh bit a 1.

The carry and auxiliary carry flags are inconsistently implemented for several instructions. For instance, the INR and DCR instructions affect all status flags except the carry flag, while the INX and DCX instructions affect no flags whatsoever. Flag settings differ for the 8080A and 8085A as well. For example, the auxiliary carry settings for the ANI and ANA are processor dependent. The description of the instruction set in this section describes these differences in detail.

ADDRESSING MODES

The following addressing modes are used by the 8080A/8085A.

Immediate Addressing

In immediate addressing, the explicit value of the instruction operand is the value on which the instruction operates.

Direct Addressing

In direct addressing, the instruction operand is a register or address containing the value on which the instruction operates.

Indirect Addressing

The 8080A/8085A supports a form of indirect addressing in which the instruction operand **M**, designates a memory byte on which the instruction operates. Note that **M** is indirectly addressed by **H** in this addressing mode.

Implied Addressing

In implied addressing, the operand is implicitly specified by the instruction; there is no explicit instruction operand. For example, the RAL instruction has no operands, since the accumulator is the implied register operand.

Register Addressing

In register addressing, the instruction operand is a name or expression specifying an 8-bit register. The designated register contains the value on which the instruction operates. Any 8-bit register may be specified, except for the status register.

This is an efficient addressing mode: only three bits are needed to specify any given register. Another advantage of register addressing is that the speed of program execution is increased: registers reside within the processor itself, and thus instructions do not require external access of data.

Register Pair Addressing

Register pair addressing allows simple and direct manipulation of 16-bit quantities. In this addressing mode the instruction operand is a 16-bit register name or expression that evaluates to a legal register pair. The value operated on is the 16-bit quantity contained in the specified register pair.

Stack Addressing

In stack addressing, a LIFO (last in-first out) stack growing downward in memory is referenced by any stack instruction, most commonly a POP or PUSH. The stack is normally initialized with a SPHL instruction which sets the value of the initial stack pointer (SP).

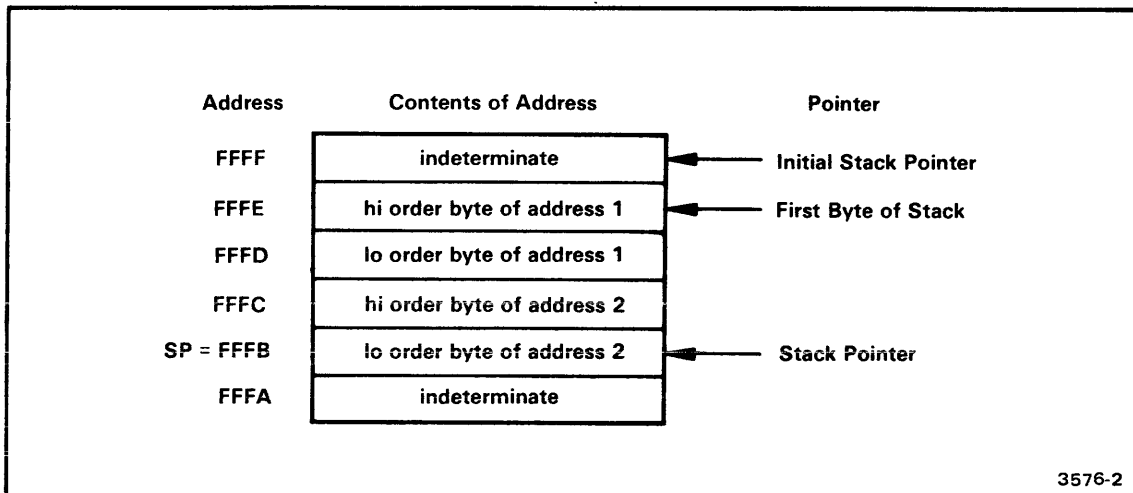


Fig. 12A-2. Stack Implementation.

The stack shown above has been initialized with the address FFFF as the value of the initial stack pointer. Two subsequent PUSH instructions have loaded addresses 1 and 2 onto the stack. Note that because the stack holds 16-bit quantities, the basic stack unit is two bytes stored in consecutive memory addresses. The high-order byte of this basic stack unit is stored at a higher address than the low-order byte. Note also that the stack pointer (SP) always points to a low-order byte.

NOTATIONAL CONVENTIONS

The following list contains short descriptions of the symbols and abbreviations used in the specification of the instruction set.

(address)	Contents of specified address.
((address))	Contents of address pointed to by specified address.
/	Separates number of cycles if condition not met, on left, from number if condition met, on right.
= 0	The designated flag is cleared.
= 1	The designated flag is set.
-	Assignment of right argument to left argument.
A	The 8-bit accumulator (a). Also the 16-bit PSW .
AC	The Auxiliary Carry flag.
B	The 16-bit B register pair. Also the 8-bit b register.
C	The 8-bit c register.
clear	To assign 0 to a flag value.
CY	The Carry flag.
cycle	A clock cycle.
D	The 16-bit D register pair. Also the 8-bit d register.
destination	The destination operand in a data transfer instruction.
devicenum	The number (0-255) of an external device.
E	The 8-bit e register.
H	The 16-bit H register pair. Also the 8-bit h register.
immed16	A 16-bit immediate address, expression, or symbol.
immed8	An 8-bit immediate data byte, expression, or symbol.
L	The 8-bit l register.
label	A symbol or expression representing a 16-bit address.

M	The memory byte indirectly addressed by the H register pair.
operand	The general type of data object required by the instruction.
operand type	A particular type of data object required by the instruction.
P	The Parity flag.
PC	The 16-bit Program Counter.
PSW	The Program Status Word: the accumulator (a) and status register considered as a 16-bit pseudo-register pair.
register	An 8-bit register: a , b , c , d , e , h , or l .
regpair	A 16-bit register pair: B , D , or H .
regpairhi	The high-order byte of a register pair.
regpairlo	The low-order byte of a register pair.
S	The Sign flag.
set	To assign 1 to a flag value.
source	The source operand in a data transfer instruction.
SP	The 16-bit Stack Pointer.
status	The 8-bit status register containing condition flags.
target	An expression or label reducible to a 16-bit address.
vectornum	Number (0-7) of restart vector.
Z	The Zero flag.

Register Notation

Wherever a register is specified in an operand field, a numeric argument may be given as well. In fact, all register names are treated as symbols and will appear in the symbol table of the assembler listing. Each register name is assigned the numeric value given in Table 12A-2. Note that this numeric argument may never exceed 7.

**Table 12A-2
Register Numbering**

Register Name	Numeric Value
A	7
B	0
C	1
D	2
E	3
H	4
L	5
M	6
PSW	6
SP	6

Mnemonic	Operands	(Description)	Cycles		Flags Affected	
			Bytes	8080A 8085A		
Operand Types					Examples	
INSTRUCTION SET						
Jumps						
JMP	target	(Load the PC with the target address)			none	
	label		3	10	10	JMP LOOP+5
PCHL	none	(Load the PC with the contents of the H register pair)			none	
	—		1	5	6	PCHL
JC	target	(Jump if carry set—CY=1)			none	
	label		3	10	7/10	JC LOOP+0FFH
JNC	target	(Jump if carry not set—CY=0)			none	
	label		3	10	7/10	JNC 0FFFFH
JZ	target	(Jump if zero set—Z=1)			none	
	label		3	10	7/10	JZ (10+34)
JNZ	target	(Jump if zero not set—Z=0)			none	
	label		3	10	7/10	JNZ SUM
JP	target	(Jump if plus—S=0)			none	
	label		3	10	7/10	JP POSRES
JM	target	(Jump if minus—S=1)			none	
	label		3	10	7/10	JM NEG
JPE	target	(Jump if parity even—P=1)			none	
	label		3	10	7/10	JPE EVEN
JPO	target	(Jump if parity odd—P=0)			none	
	label		3	10	7/10	JPO ODD+5

Mnemonic	Operands	(Description)	Cycles		Flags Affected
			Operand Types	Bytes	
Stack Operations					
POP	destination	(regpairlo←(SP), SP←SP+1, regpairhi←(SP), SP←SP+1)			none ^a
	regpair PSW	1 1	10 10	10 10	POP B POP PSW
PUSH	source	(SP←SP-1, (SP)←regpairhi, SP←SP-1, (SP)←regpairlo)			none
	regpair PSW	1 1	11 11	12 12	PUSH H PUSH PSW
SPHL	none	(Load SP with H register pair)			none
	—	1	5	6	SPHL
XTHL	none	(Exchange 16-bit value at top of stack with H register pair)			none
	—	1	18	16	XTHL
Subroutine Handling: Calls					
CALL	target	(Call subroutine at target address, unconditionally)			none
	label	3	17	18	CALL WRITENUM
CC	target	(Call if carry set—CY=1)			none
	label	3	11/17	9/18	CC SUB1
CNC	target	(Call if carry not set—CY=0)			none
	label	3	11/17	9/18	CNC SUB2
CZ	target	(Call if zero set—Z=1)			none
	label	3	11/17	9/18	CZ SUBZ+5
CNZ	target	(Call if zero not set—Z=0)			none
	label	3	11/17	9/18	CNZ NOZEE
CM	target	(Call if minus S=1)			none
	label	3	11/17	9/18	CM NEG-4
CP	target	(Call if plus—S=0)			none
	label	3	11/17	9/18	CP SUBP

Mnemonic	Operands	(Description)	Cycles		Flags Affected
			Operand Types	Bytes	
Subroutine Handling: Calls (cont.)					
CPE	target	(Call if parity even—P=1)			none
	label		3	11/17	9/18
CPO	target	(Call if parity odd—P=0)			none
	label		3	11/17	9/18
Subroutine Handling: Returns					
RET	none	(Return from last called subroutine, unconditionally)			none
	—		1	10	10
RC	none	(Return if carry set—CY=1)			none
	—		1	5/11	6/12
RNC	none	(Return if carry not set—CY=0)			none
	—		1	5/11	6/12
RZ	none	(Return if zero set—Z=1)			none
	—		1	5/11	6/12
RNZ	none	(Return if zero not set—Z=0)			none
	—		1	5/11	6/12
RM	none	(Return if minus—S=1)			none
	—		1	5/11	6/12
RP	none	(Return if plus—S=0)			none
	—		1	5/11	6/12
RPE	none	(Return if parity even—P=1)			none
	—		1	5/11	6/12
RPO	none	(Return if parity odd—P=0)			none
	—		1	5/11	6/12

Mnemonic	Operands	(Description)	Cycles		Flags Affected
			Operand Types	Bytes	
Data Transfer					
MOV	destination, source	(Copy source to destination)			none
	register,register	1	5	4	MOV B,A
	register,memory	1	7	7	MOV E,M
	memory,register	1	7	7	MOV M,D
MVI	destination, source	(Copy immediate byte to destination)			none
	register,immed8	2	7	7	MVI D,255
	memory,immed8	2	10	10	MVI M,0FFH
LDA	source	(Load accumulator with contents of source address)			none
	label	3	13	13	LDA 444Q
LHLD	source	(Load l with byte at source, Load h with byte at source+1)			none
	label	3	16	16	LHLD VAL
LDAX	source	(Load accumulator with contents addressed by source register pair)			none
	regpair ^b	1	7	7	LDAX D
LXI	destination, source	(Load destination with 16-bit immediate data)			none
	regpair,immed16	3	10	10	LXI B,0FA14H
	SP,immed16	3	10	10	LXI SP,NEWSP
STA	destination	(Store contents of accumulator at destination)			none
	label	3	13	13	STA BUFFER
SHLD	destination	(Store l at destination, Store h at destination+1)			none
	label	3	16	16	SHLD FROM+1

Mnemonic	Operands	(Description)	Cycles		Flags Affected	
			Bytes	8080A		8085A
Operand Types					Examples	
Data Transfer (cont.)						
STAX	destination	(Store contents of accumulator at destination address contained in register pair)			none	
	regpair ^b		1	7	7	STAX B
XCHG	none	(Exchange contents of H and D register pairs)			none	
	—		1	4	4	XCHG
Arithmetic Operations						
ADD	source	(Add source to accumulator with no carry)			CY,AC,S,Z,P	
	register		1	4	4	ADD B
	memory		1	7	7	ADD M
ADC	source	(Add source to accumulator with carry)			CY,AC,S,Z,P	
	register		1	4	4	ADC E
	memory		1	7	7	ADC M
ADI	source	(Add immediate byte to accumulator with no carry)			CY,AC,S,Z,P	
	immed8		2	7	7	ADI 34Q
ACI	source	(Add immediate byte to accumulator with carry)			CY,AC,S,Z,P	
	immed8		2	7	7	ACI 34H
DAD	source	(Add contents of source register pair to H register pair)			CY	
	regpair		1	10	10	DAD H
	SP		1	10	10	DAD SP
SUB	source	(Subtract source byte from accumulator with no borrow)			CY,AC,S,Z,P	
	register		1	4	4	SUB D
	memory		1	7	7	SUB M

Mnemonic	Operands	(Description)	Cycles		Flags Affected
			Bytes	8080A	
Operand Types					Examples
Arithmetic Operations (cont.)					
SBB	source	(Subtract source byte from accumulator with borrow)			CY^c,AC,S,Z,P
	register	1	4	4	SBB C
	memory	1	7	7	SBB M
SUI	source	(Subtract immediate source byte from accumulator with no borrow)			CY^c,AC,S,Z,P
	immed8	2	7	7	SUI 255
SBI	source	(Subtract immediate source byte from accumulator with borrow)			CY^c,AC,S,Z,P
	immed8	2	7	7	SBI 251
DAA	none	(Convert contents of accumulator to BCD. Use immediately after accumulator addition or increment)			CY,AC,S,Z,P
	—	1	4	4	DAA
INR	destination	(Increment destination by 1, carry flag unaffected)			AC,S,Z,P
	register	1	5	4	INR B
	memory	1	10	10	INR M
INX	destination	(Increment destination register pair by 1)			none
	regpair	1	5	6	INX H
	SP	1	5	6	INX SP
DCR	destination	(Decrement destination by 1, carry flag unaffected)			AC,S,Z,P
	register	1	5	4	DCR E
	memory	1	10	10	DCR M
DCX	destination	(Decrement destination register pair by 1)			none
	regpair	1	5	6	DCX H
	SP	1	5	6	DCX SP

Mnemonic	Operands	(Description)	Cycles		Flags Affected
			Bytes	8080A	
Operand Types					Examples
Logical Operations					
ANA	source	(AND source byte with contents of accumulator)			CY=0, AC^d,S,Z
	register memory	1 1	4 7	4 7	ANA C ANA M
ANI	source	(AND immediate byte with contents of accumulator)			CY=0, AC^d,S,Z
	immed8	2	7	7	ANI 123
ORA	source	(OR source byte with contents of accumulator)			CY=0, AC=0, S,Z,P
	register memory	1 1	4 7	4 7	ORA C ORA M
ORI	source	(OR immediate byte with contents of accumulator)			CY=0, AC=0, S,Z,P
	immed8	2	7	7	ORI 10101010B
XRA	source	(XOR source byte with contents of accumulator)			CY=0,AC=0, S,Z,P
	register memory	1 1	4 7	4 7	XRA B XRA M
XRI	source	(XOR immediate byte with contents of accumulator)			CY=0,AC=0, S,Z,P
	immed8	2	7	7	XRI 11010101B
CMA	none	(One's-complement contents of accumulator)			none
	—	1	4	4	CMA
CMP	source	(Subtract source from accumulator, result not stored. If source greater, then CY=1, else CY=0)			CY,AC,S,Z,P
	register memory	1 1	4 7	4 7	CMP B CMP M

Mnemonic	Operands	(Description)	Cycles		Flags Affected
			Bytes	8080A 8085A	
Operand Types					Examples

Logical Operations (cont.)

CPI **source** **(Subtract immediate byte from accumulator. Result not stored. If source greater, then CY=1, else CY=0)** **CY,AC,S,Z,P**

immed8 2 7 7 CPI MASK—4

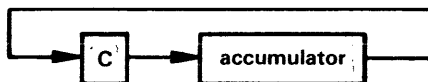
STC **none** **(Set carry flag to 1)** **CY=1**

— 1 4 4 STC

CMC **none** **(Complement carry flag)** **CY**

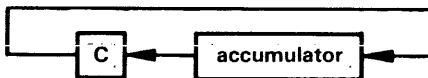
— 1 4 4 CMC

RAR **none** **(Rotate accumulator right through carry)** **CY**



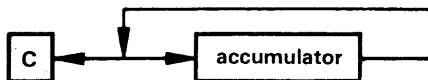
— 1 4 4 RAR

RAL **none** **(Rotate accumulator left through carry)** **CY**



— 1 4 4 RAL

RRC **none** **(Rotate accumulator right, carry affected)** **CY**



— 1 4 4 RRC

Mnemonic	Operands	(Description)	Cycles		Flags Affected
			Bytes	8080A 8085A	
Operand Types					Examples
Logical Operations (cont.)					
RLC	none	(Rotate accumulator left, carry affected)			CY
	—		1	4 4	RLC
Miscellaneous (IN, OUT, HLT, NOP, RST, Interrupt Instructions)					
IN	devicenum	(Send byte to accumulator from external device [0—255])			none
	immed8		1	10 10	IN 255
OUT	devicenum	(Send contents of accumulator to external device [0—255])			none
	immed8		1	10 10	OUT PRINTER
HLT	none	(Stop execution of processor)			none
	—		1	7 5	HLT
NOP	none	(No operation)			none
	—		1	4 4	NOP
RST	vectornum	(PUSH PC on stack, load high-order byte of PC with [00], load low-order byte with [8 X vectornum])			none
	immed8		1	11 12	RST 7
EI	none	(Enable interrupts after next instruction)			none
	—		1	4 4	EI
DI	none	(Disable interrupts after next instruction)			none
	—		1	4 4	DI

Mnemonic	Operands	(Description)	Cycles		Flags Affected
			8080A	8085A	
	Operand Types	Bytes			Examples
Instructions Available on 8085A only					
RIM	none	(Copy interrupt mask to accumulator)			none
	—	1	—	4	RIM
SIM	none	(Copy contents of accumulator to interrupt mask)			none
	—	1	—	4	SIM

^aPOP PSW affects all flags since the low-order byte of the top stack unit is loaded into the status register.

^bLDAX and STAX apply to B and D register pairs only; the H register pair is illegal.

^cCarry set to 1 if borrow required.

^dAC=1 for 8085A.

RESERVED WORDS

The following names may **not** be used to represent an address, data item, or variable. They are words reserved for specifying instruction mnemonics, register names, and assembler directives, options, and operators.

8080A/8085A Mnemonics

ACI	CMC	DAD	JM	LHLD	PUSH	RP	SPHL
ADC	CMP	DCR	JMP	LXI	RAL	RPE	STA
ADD	CNC	DCX	JNC	MOV	RAR	RPO	STAX
ADI	CNZ	DI	JNZ	MVI	RC	RRC	STC
ANA	CP	EI	JP	NOP	RET	RST	SUB
ANI	CPE	HLT	JPE	ORA	RIM	RZ	SUI
CALL	CPI	IN	JPO	ORI	RLC	SBB	XCHG
CC	CPO	INR	JZ	OUT	RM	SBI	XRA
CM	CZ	INX	LDA	PCHL	RNC	SHLD	XRI
CMA	DAA	JC	LDAX	POP	RNZ	SIM	XTHL

8080A/8085A Register Names

A	B	C	D	E
H	L	M	PSW	SP

Tektronix Assembler Directives, Options, and Operators

ABSOLUTE	END	INPAGE	PAGE	STITLE
ASCII	ENDIF	LIST	REPEAT	STRING
BASE	ENDM	LO	RESERVE	SYM
BLOCK	ENDOF	MACRO	RESUME	TITLE
BYTE	ENDR	ME	SCALAR	TRM
CND	EQU	MEG	SECTION	WARNING
COMMON	EXITM	MOD	SEG	WORD
CON	GLOBAL	NAME	SET	
DBG	HI	NCHR	SHL	
DEF	IF	NOLIST	SHR	
ELSE	INCLUDE	ORG	SPACE	

PAGE SIZE

The page size of the 8080A/8085A Assembler is 256 bytes.

ERROR MESSAGES

The following error messages are specific to the 8080A and 8085A microprocessors:

******* ERROR: 251 Missing or invalid operand.** Either a syntax error has occurred in an operand or the operand has been omitted.

******* ERROR: 252 Register expression greater than 7.** A register designation expression exceeds the number 7 and thus does not specify a legal register. Refer back to the Notational Conventions in this section for a discussion of register notation.

******* ERROR: 253 Invalid register pair.** A register designation expression is either odd, greater than 6, or illegal for the particular instruction. Note that all register pairs must be specified by an even numeric argument. Refer back to the Notational Conventions in this section for a discussion of register notation.

******* ERROR: 254 Register expression is not scalar.** An address expression is used where a register expression is required. Refer back to the Notational Conventions in this section for a discussion of register notation.

IRREGULARITIES

Differences Between The 8080A and the 8085A

The 8085A supports features unavailable on the 8080A: the RIM and SIM instructions, and an interrupt mask with three additional restart vectors. The 8085A and 8080A also implement the ANA and ANI instructions differently. The 8085A sets the auxiliary carry flag to one for both instructions. On the 8080A, the auxiliary carry flag is assigned the value resulting from a logical OR of bit 3 of the accumulator and bit 3 of the source byte.

Timing differences between the 8080A and 8085A are documented with the description of each instruction.

MANUAL CHANGE INFORMATION

At Tektronix, we continually strive to keep up with latest electronic developments by adding circuit and component improvements to our instruments as soon as they are developed and tested.

Sometimes, due to printing and shipping requirements, we can't get these changes immediately into printed manuals. Hence, your manual may contain new change information on following pages.

A single change may affect several sections. Since the change information sheets are carried in the manual until all changes are permanently entered, some duplication may occur. If no such change pages appear following this page, your manual is correct as printed.

DESCRIPTION

TEXT CORRECTION

Page 12A-1

In Table 12A-1, the high-order byte of the D register pair should be "d", and the low-order byte should be "e". Change the third line of the table to read as follows:

D	Register Pair	d	e
---	---------------	---	---

Date: 3-23-81 Change Reference: C10/381
Product: GENERAL LIMITED RIGHTS LEGEND Manual Part No.: GENERAL

DESCRIPTION

LIMITED RIGHTS LEGEND

Software License No. _____

Contractor: Tektronix, Inc.

Explanation of Limited Rights Data Identification Method

Used: Entire document subject to limited rights.


Those portions of this technical data indicated as limited rights data shall not, without the written permission of the above Tektronix, be either (a) used, released or disclosed in whole or in part outside the Customer, (b) used in whole or in part by the Customer for manufacture or, in the case of computer software documentation, for preparing the same or similar computer software, or (c) used by a party other than the Customer, except for: (i) emergency repair or overhaul work only, by or for the Customer, where the item or process concerned is not otherwise reasonably available to enable timely performance of the work, provided that the release or disclosure hereof outside the Customer shall be made subject to a prohibition against further use, release or disclosure; or (ii) release to a foreign government, as the interest of the United States may require, only for information or evaluation within such government or for emergency repair or overhaul work by or for such government under the conditions of (i) above. This legend, together with the indications of the portions of this data which are subject to such limitations shall be included on any reproduction hereof which includes any part of the portions subject to such limitations.

RESTRICTED RIGHTS IN SOFTWARE

The software described in this document is licensed software and subject to **restricted rights**. The software may be used with the computer for which or with which it was acquired. The software may be used with a backup computer if the computer for which or with which it was acquired is inoperative. The software may be copied for archive or backup purposes. The software may be modified or combined with other software, subject to the provision that those portions of the derivative software incorporating restricted rights software are subject to the same restricted rights.

Copyright © 1980 Tektronix, Inc. All rights reserved. Contents of this publication may not be reproduced in any form without the written permission of Tektronix, Inc.

Products of Tektronix, Inc. and its subsidiaries are covered by U.S. and foreign patents and/or pending patents.

TEKTRONIX, TEK, SCOPE-MOBILE, and  are registered trademarks of Tektronix, Inc. TELEQUIPMENT is a registered trademark of Tektronix U.K. Limited.

Printed in U.S.A. Specification and price change privileges are reserved.



COMMITTED TO EXCELLENCE

MANUAL CHANGE INFORMATION

Date: 5/31/83

Change Reference: C2/683

Product: 8550: 8080A/8085A Assembler Specifics

Manual Part No.: 070-3576-00

DESCRIPTION

Product Group 61

CHANGE INSTRUCTIONS

THIS IS A PAGE REPLACEMENT CHANGE.

To implement this change:

- a. Remove old title page.
- b. Insert new title page.

**8550
Microcomputer
Development
Lab**

Assembler Specifics

8080A/8085A

This manual supports the following TEKTRONIX products:

ASM8085 Option 1Z

This manual supports a software module that is compatible with:

DOS 50 Version 1 (8550)
DOS 50 Version 2 (8550)

*Please check for change information
at the rear of this manual*

LIMITED RIGHTS LEGEND

Software License No. _____

Contractor: Tektronix, Inc.

Explanation of Limited Rights Data Identification Method

Used: Entire document subject to limited rights.


Those portions of this technical data indicated as limited rights data shall not, without the written permission of the above Tektronix, be either (a) used, released or disclosed in whole or in part outside the Customer, (b) used in whole or in part by the Customer for manufacture or, in the case of computer software documentation, for preparing the same or similar computer software, or (c) used by a party other than the Customer, except for: (i) emergency repair or overhaul work only, by or for the Customer, where the item or process concerned is not otherwise reasonably available to enable timely performance of the work, provided that the release or disclosure hereof outside the Customer shall be made subject to a prohibition against further use, release or disclosure; or (ii) release to a foreign government, as the interest of the United States may require, only for information or evaluation within such government or for emergency repair or overhaul work by or for such government under the conditions of (i) above. This legend, together with the indications of the portions of this data which are subject to such limitations shall be included on any reproduction hereof which includes any part of the portions subject to such limitations.

RESTRICTED RIGHTS IN SOFTWARE

The software described in this document is licensed software and subject to **restricted rights**. The software may be used with the computer for which or with which it was acquired. The software may be used with a backup computer if the computer for which or with which it was acquired is inoperative. The software may be copied for archive or backup purposes. The software may be modified or combined with other software, subject to the provision that those portions of the derivative software incorporating restricted rights software are subject to the same restricted rights.

Copyright © 1980 Tektronix, Inc. All rights reserved. Contents of this publication may not be reproduced in any form without the written permission of Tektronix, Inc.

Products of Tektronix, Inc. and its subsidiaries are covered by U.S. and foreign patents and/or pending patents.

TEKTRONIX, TEK, SCOPE-MOBILE, and  are registered trademarks of Tektronix, Inc. TELEQUIPMENT is a registered trademark of Tektronix U.K. Limited.

Printed in U.S.A. Specification and price change privileges are reserved.