2521/302 2602/302
2601/302 2603/302
MICROCONTROLLER TECHNOLOGY
AND MICROPROCESSOR SYSTEMS
Oct./Nov. 2022
Time: 3 hours

THE KENYA NATIONAL EXAMINATIONS COUNCIL

DIPLOMA IN ELECTRICAL AND ELECTRONIC ENGINEERING
(POWER OPTION)
(TELECOMMUNICATION OPTION)
(INSTRUMENTATION OPTION)

MODULE III

MICROCONTROLLER TECHNOLOGY AND MICROPROCESSOR SYSTEMS

3 hours

INSTRUCTIONS TO CANDIDATES

You should have the following for this examination:
- *Answer booklet;*
- *8080/85 microprocessor instruction set;*
- *8051 microcontroller instruction set;*
- *Non-programmable scientific calculator.*

*This paper consists of **EIGHT** questions in **TWO** sections; **A** and **B**.*
*Answer any **THREE** questions from section **A** and any **TWO** questions from section **B** in the answer booklet provided.*
All questions carry equal marks.
Maximum marks for each part of a question are as indicated.
Candidates should answer the questions in English.

This paper consists of 11 printed pages.

Candidates should check the question paper to ascertain that all the pages are printed as indicated and that no questions are missing.

SECTION A: MICROPROCESSOR SYSTEMS

*Answer **THREE** questions from this section.*

1. (a) State **two** defining features of each of the following input/output (I/O) techniques:

(i) memory mapped;
(ii) I/O mapped. (4 marks)

(b) Table I shows part of a microprocessor instruction set.

Table 1

Instruction	Meaning
ADD A, D	A ⟵ A + D
MUL D	A ⟵ A x D
MUL A	A ⟵ A x A
MOV D,A	D ⟵ A
MOV A, DATA	A ⟵ DATA
END	Halt program

Use the given instructions to write an assembly language program to perform each of the following:

(i) $y = x^4 + x^2$

(ii) $z = 8x$ (8 marks)

(c) Write Intel 8085 microprocessor assembly language program segments to perform each of the following:

(i) $2789 + 3273$

(ii) $y = (11010011)_2 NAND (11101100)_2$

(iii) $(00000110)_2 \times (1100)_2$ (8 marks)

2. (a) Describe each of the following microprocessor addressing modes, using an example in each case:

(i) immediate;
(ii) stack;
(iii) register indirect. (6 marks)

(b) With aid of an example, distinguish between an opcode and an operand with respect to a microprocessor instruction. (3 marks)

(c) Write an assembly language program, using stack operations, to exchange data between register pairs BC and DE. (4 marks)

(d) With aid of a flowchart, write an assembly language program to search for a byte 'BTEY' from a block of consecutive memory locations, which is 256 bytes long and report the number of occurrences of the byte. The starting address of the memory block is 2000H. (7 marks)

3. (a) Figure 1 shows a pin diagram of a microprocessor system device.

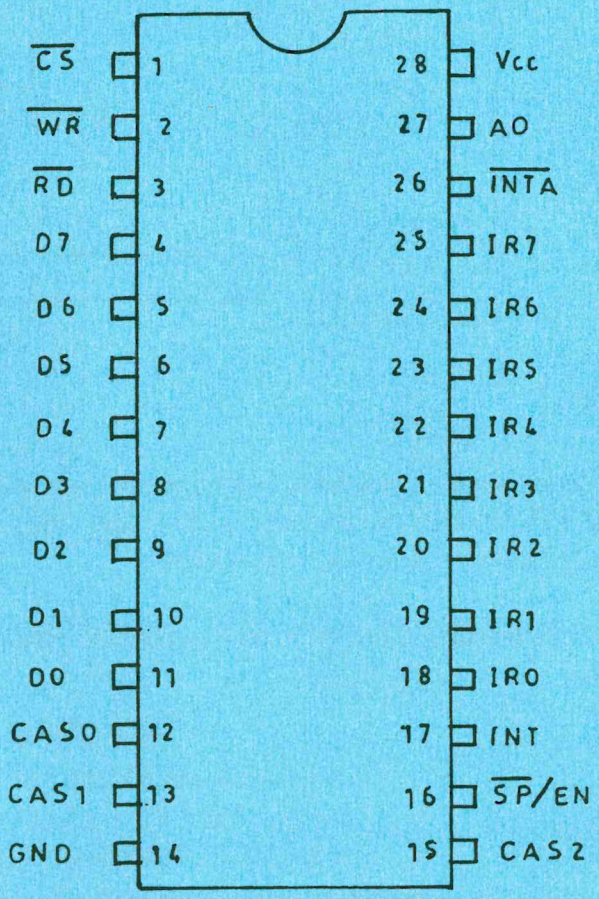

Fig.1

(i) Identify the device.
(ii) Explain the purpose of the device.
(iii) For signals $\overline{INTA}$, INT, IR0-IR7:

I. State the function of each;
II. List a device each signal would typically be connected to in a system.
(9 marks)

(b) Figure 2 shows a block diagram of a printer connected via a parallel interface to a microprocessor using two 8-bit ports. Port A address is 40 H, port B address is 41H and port C address is 42H.

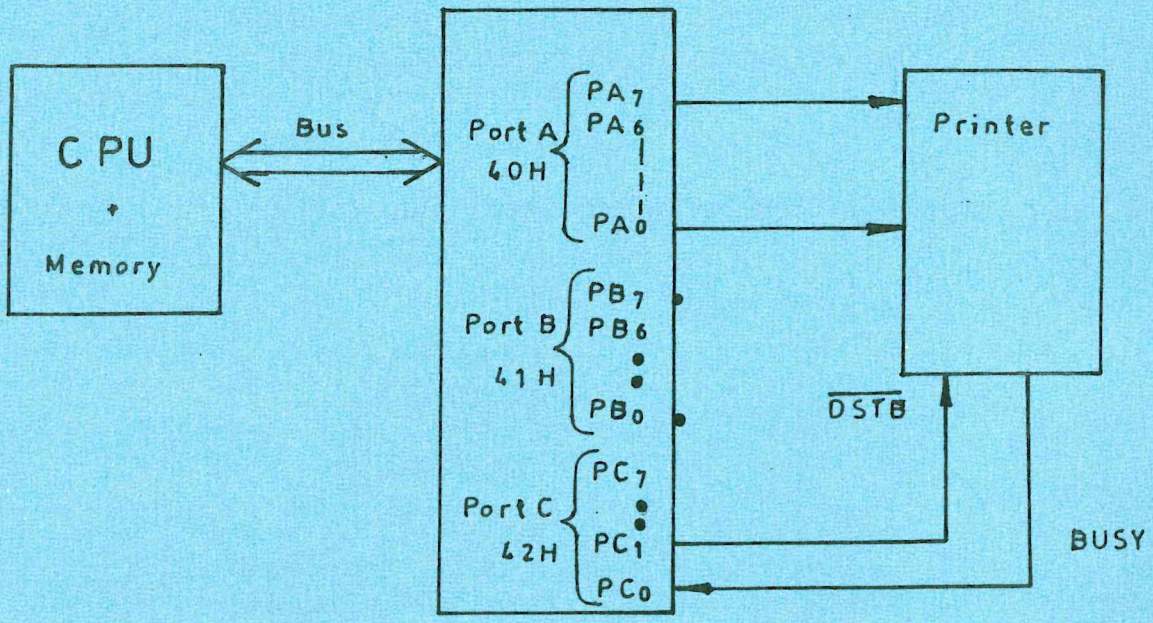

Fig. 2

With aid of a flowchart, write an assembly language program to transfer five ASCII characters from the CPU memory to the printer. The characters are stored in consecutive memory locations, starting from 4000H. (11 marks)

4. (a) State **two** tests that can be carried out on a microprocessor system using each of the following equipment:

(i) multimeter;
(ii) oscilloscope;
(iii) logic analyser. (6 marks)

(b) (i) Outline the checksum method of testing a microcomputer random access memory (ROM).
(ii) Explain the shortcoming of the method in (b)(i). (6 marks)

(c) Table 2 shows an 8085 assembly language program listing. Draw a trace table of the program execution. (8 marks)

Table 2

LX1 SP, F000H
LX1 H, 8040H
MVIA, 35H
ADI 18H
MOV M, A
ADD M
LXI B, 8050H
STAX B
XRA A
INX B
STAX B
ADD B
MOV C, A
HLT

5. (a) With aid of a diagram, explain the use of in-circuit emulator (ICE) in development of a microprocessor-based system. (6 marks)

(b) Explain the activities in each of the following microprocessor-system development stages:

(i) schematic capture;
(ii) simulation and timing verification;
(iii) components libraries and packages;
(iv) printed circuit board (PCB) layout. (8 marks)

(c) Describe typical features of each of the following types of operating systems:

(i) real time;
(ii) network;
(iii) batch. (6 marks)

SECTION B: MICROCONTROLLER TECHNOLOGY

*Answer **TWO** questions from this section.*

6. (a) State **three** advantages of programmable logic controllers (PLCs) over relays in industrial control. (3 marks)

(b) Draw circuit diagrams for each of the following PLC interfaces:

(i) opto-isolator input;
(ii) darlington output;
(iii) TRIAC output. (6 marks)

(c) Figure 3 shows a ladder diagram for controlling a drilling machine using a PLC.

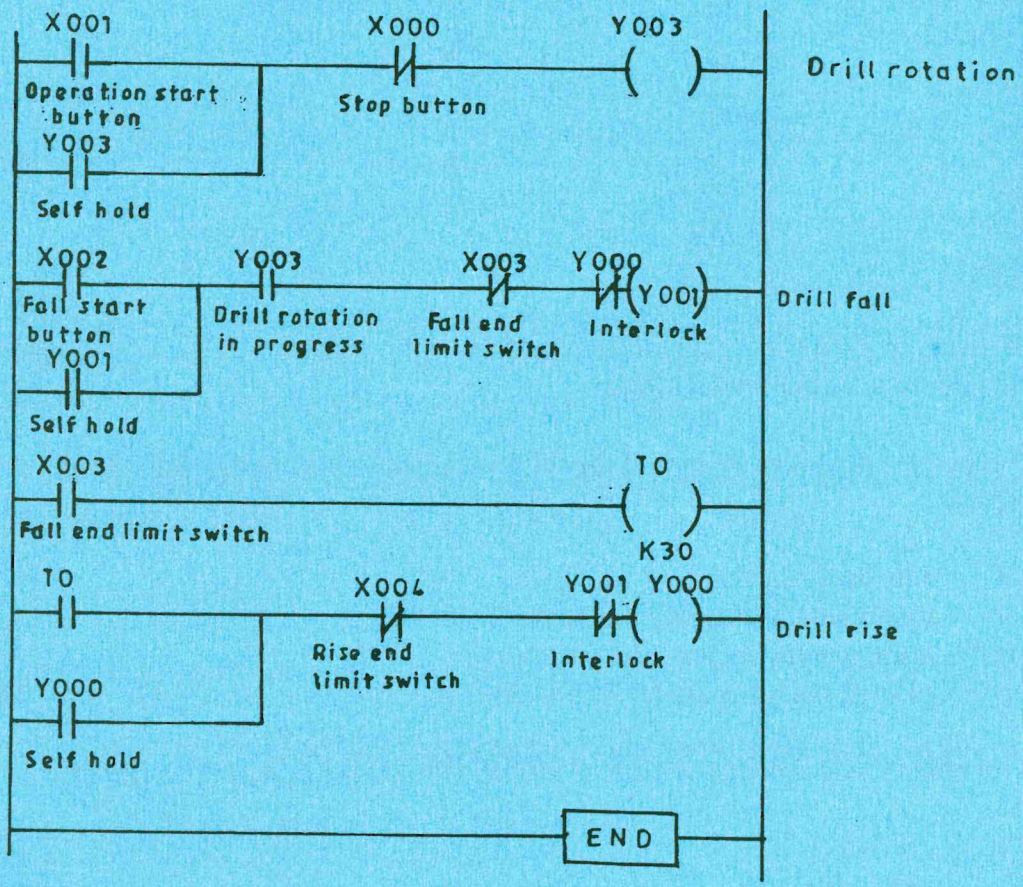

Fig.3

(c) (i) Write down the program listing for the control.

(ii) Identify **two** physical contacts for each for the following:

I. inputs;
II. outputs. (11 marks)

7. (a) Define each of the following with respect to process control:

(i) transient response;
(ii) offset;
(iii) measured variable. (3 marks)

(b) (i) State **two** merits of hydraulic actuators.
(ii) Figure 4 shows a diagram of a hydraulic gear motor. Describe its operation. (5 marks)

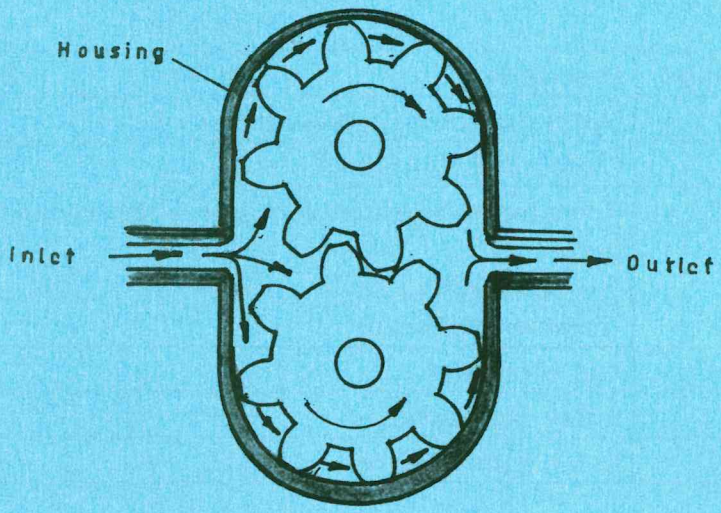

Fig. 4

(c) A spring-loaded spring-acting cylinder has a diameter of 4 cm and a stroke of 5 cm. The return spring has a spring constant of 6 kg/cm. The available pressure is $2 \times 10^5 N/m^2$. Determine the:

(i) piston surface area;
(ii) force on the piston;
(iii) force from the cylinder. (8 marks)

(d) Explain the Ziegler-Nichols closed-loop method of tuning a PID controller. (4 marks)

8. (a) With aid of a diagram, describe each of the following robot-arm motions:

(i) yaw;
(ii) roll;
(iii) pitch. (6 marks)

(b) Describe each of the following elements of a robotic system:

(i) actuator;
(ii) sensor;
(iii) drive. (6 marks)

(c) Table 3(a) shows the commands of a robot programming language while table 3(b) is a program to control the movement of the robot. Sketch a trace of the robot path under the program control, indicating the distances. (8 marks)

Table 3(a)

No.	Command	Description
1.	Open	Opens the Robot Gripper Jaws
2.	Close	Closes the Robot Gripper Jaws
3.	Move (x, y)	Robot moves x-metres eastwards, y-meters northwards Note: -x= move westwards -y=move southwards
4.	End	End of program
5.	Stop	Robot stops moving
6.	Start	Robot starts to move

Table 3(b)

- Open
- Close
- Start
- Move (30,0)
- Stop
- Start
- Move (0, -15
- Stop
- Start
- Move (-25, -10)
- Stop
- Start
- Move (8, -6)
- Stop
- Open
- End

Appendix A: Instruction Set of 8051 Microcontroller

Mnemonics, Arranged Alphabetically

MNEMONIC	DESCRIPTION	BYTES	CYCLES	FLAGS
ACALL addr11	PC + 2 → (SP); addr11 → PC	2	2	
ADD A, direct	A + (direct) → A	2	1	C OV AC
ADD A, @Ri	A + (Ri) → A	1	1	C OV AC
ADD A, #data	A + #data → A	2	1	C OV AC
ADD A, Rn	A + Rn → A	1	1	C OV AC
ADDC A, direct	A + (direct) + C → A	2	1	C OV AC
ADDC A, @Ri	A + (Ri) + C → A	1	1	C OV AC
ADDC A, #data	A + #data + C → A	2	1	C OV AC
ADDC A, Rn	A + Rn + C → A	1	1	C OV AC
AJMP addr11	Addr11 → PC	2	2	
ANL A, direct	A AND (direct) → A	2	1	
ANL A, @Ri	A AND (Ri) → A	1	1	
ANL A, #data	A AND #data → A	2	1	
ANL A, Rn	A AND Rn → A	1	1	
ANL direct, A	(direct) AND A → (direct)	2	1	
ANL direct, #data	(direct) AND #data → (direct)	3	2	
ANL C, bit	C AND bit → C	2	2	C
ANL C, $\overline{bit}$	C AND $\overline{bit}$ → C	2	2	C
CJNE A, direct, rel	[A <> (direct)]: PC + 3 + rel → PC	3	2	C
CJNE A, #data, rel	[A <> data]: PC + 3 + rel → PC	3	2	C
CJNE @Ri, #data, rel	[(Ri) <> data]: PC + 3 + rel → PC	3	2	C
CJNE Rn, #data, rel	[Rn <> data]: PC + 3 + rel → PC	3	2	C
CLR A	0 → A	1	1	
CLR bit	0 → bit	2	1	
CLR C	0 → C	1	1	0
CPL A	$\overline{A}$ → A	1	1	
CPL bit	$\overline{bit}$ → bit	2	1	
CPL C	$\overline{C}$ → C	1	1	C
DA A	Abin → Adec	1	1	C
DEC A	A – 1 → A	1	1	
DEC direct	(direct) – 1 → (direct)	2	1	
DEC @Ri	(Ri) – 1 → (Ri)	1	1	
DEC Rn	Rn – 1 → Rn	1	1	
DIV AB	A/B → AB	1	4	0 OV
DJNZ direct, rel	[(direct) – 1 <> 00]: PC + 3 + rel → PC	3	2	
DJNZ Rn, rel	[Rn – 1 <> 00]: PC + 2 + rel → PC	2	2	
INC A	A + 1 → A	1	1	
INC direct	(direct) + 1 → (direct)	2	1	
INC DPTR	DPTR + 1 → DPTR	1	2	
INC @Ri	(Ri) + 1 → (Ri)	1	1	
INC Rn	Rn + 1 → Rn	1	1	
JB bit, rel	[b=1]: PC + 3 + rel → PC	3	2	
JBC bit, rel	[b=1]: PC + 3 + rel → PC; 0 → bit	3	2	
JC rel	[C=1]: PC + 2 + rel → PC	2	2	
JMP @A + DPTR	DPTR + A → PC	1	2	
JNB bit, rel	[b=0]: PC + 3 + rel → PC	3	2	
JNC rel	[C=0]: PC + 2 + rel → PC	2	2	
JNZ rel	[A>00]: PC + 2 + rel → PC	2	2	
JZ rel	[A=00]: PC + 2 + rel → PC	2	2	
LCALL addr16	PC + 3 → (SP); addr16 → PC	3	2	
LJMP addr16	Addr16 → PC	3	2	
MOV A, direct	(direct) → A	2	1	
MOV A, @Ri	(Ri) → A	1	1	
MOV A, #data	#data → A	2	1	
MOV A, Rn	Rn → A	1	1	
MOV direct, A	A → (direct)	2	1	
MOV direct, direct	(direct) → (direct)	3	2	
MOV direct, @Ri	(Ri) → (direct)	2	2	
MOV direct, #data	#data → (direct)	3	2	
MOV direct, Rn	Rn → (direct)	2	2	
MOV bit, C	C → bit	2	2	C
MOV C, bit	bit → C	2	1	
MOV @Ri, A	A → (Ri)	1	1	
MOV @Ri, direct	(direct) → (Ri)	2	2	

MNEMONIC	DESCRIPTION	BYTES	CYCLES	FLAGS
MOV Rn, #data	#data → Rn	2	1	
MOVC A, @A+DPTR	(A+DPTR) → A	1	2	
MOVC A, @A+PC	(A+PC) → A	1	2	
MOVX A, @DPTR	(DPTR)^ → A	1	2	
MOVX A, @Ri	(Ri)^ → A	1	2	
MOVX @DPTR, A	A → (DPTR)^	1	2	
MOVX @Ri, A	A →(Ri)^	1	2	
NOP	PC + 1 → PC	1	1	
MUL AB	A x B → AB	1	4	0 OV
ORL A, direct	A OR (direct) → A	2	1	
ORL A, @Ri	A OR (Ri) → A	1	1	
ORL A, #data	A OR #data → A	2	1	
ORL A, Rn	A OR Rn → A	1	1	
ORL direct, A	(direct) OR A → (direct)	2	1	
ORL direct, #data	(direct) OR #data → (direct)	3	2	
ORL C, bit	C OR bit → C	2	2	C
ORL C, /bit	C OR /bit → C	2	2	C
POP direct	(SP) → (direct)	2	2	
PUSH direct	(direct) → (SP)	2	2	
RET	(SP) → PC	1	2	
RETI	(SP) → PC; EI	1	2	
RL A	A0←A7←A6..←A1←A0	1	1	
RLC A	C←A7←A6..←A0←C	1	1	C
RR A	A0→A7→A5..→A1→A7	1	1	
RRC A	C→A7→A6..→A0→C	1	1	C
SETB bit	1 → bit	2	1	
SETB C	1 → C	1	1	1
SJMP rel	PC + 2 + rel→ PC	2	2	
SUBB A, direct	A – (direct)–C → A	2	1	C OV AC
SUBB A, @Ri	A – (Ri)–C → A	1	1	C OV AC
SUBB A, #data	A – #data–C → A	2	1	C OV AC
SUBB A, Rn	A – Rn–C → A	1	1	C OV AC
SWAP A	Alsn ↔ Amsn	1	1	
XCH A, direct	A ↔ (direct)	2	1	
XCH A, @Ri	A ↔ (Ri)	1	1	
XCH A, Rn	A ↔ Rn	1	1	
XCHD A, @Ri	Alsn ↔ (Ri)lsn	1	1	
XRL A, direct	A XOR (direct) → A	2	1	
XRL A, @Ri	A XOR (Ri) → A	1	1	
XRL A,#data	A XOR #data → A	2	1	
XRL A, Rn	A XOR Rn → A	1	1	
XRL direct, A	(direct) XOR A → (direct)	2	1	
XRL direct, #data	(direct) XOR #data → (direct)	3	2	

ACRONYMS

addr11	Page address of 11 bits, which is in the same 2K page as the address of the following instruction.
addr16	Address for any location in the 64K memory space.
bit	The address of a bit in the internal RAM bit address area or a bit in an SFR.
C	The carry flag.
#data	An 8-bit binary number from 00 to FFh.
#data16	A 16-bit binary number from 0000 to FFFFh.
direct	An internal RAM address or an SFR byte address.
lsn	Least significant nibble.
msn	Most significant nibble.
rel	Number that is added to the address of the next instruction to form an address +127d to –128d from the address of the next instruction.
Rn	Any of registers R0 to R7 of the current register bank.
@Ri	Indirect address using the contents of R0 or R1.
[ ]:	IF the condition inside the brackets is *true*, THEN the action listed will occur; ELSE go to the next instruction.
^	EXTERNAL memory location.
( )	Contents of the location inside the parentheses.

Note that flags affected each instruction are shown where appropriate; any operations which affect the PSW address may also affect the flags.

Instruction set of

8080/8085

OP CODE	MNEMONIC	OP CODE	MNEMONIC	OP CODE	MNEMONIC	OP CODE	MNEMONIC	OP CODE	MNEMONIC	OP CODE	MNEMONIC
00	NOP	28	DCX H	56	MOV D,M	81	ADD C	AC	XRA H	D7	RST 2
01	LX1 B,D16	2C	INR L	57	MOV D,A	82	ADD D	AD	XRA L	D8	RC
02	STAX B	2D	DCR L	58	MOV E,B	83	ADD E	AE	XRA M	D9	–
03	INX B	2E	MVI L,D8	59	MOV E,C	84	ADD H	AF	XRA A	DA	JC Adr
04	INR B	2F	CMA	5A	MOV E,D	85	ADD L	B0	ORA B	DB	IN D8
05	DCR B	30	SIM	5B	MOV E,E	86	ADD M	B1	ORA C	DC	CC Adr
06	MVI B,D8	31	LXI SPD16	5C	MOV E,H	87	ADD A	B2	ORA D	DD	–
07	RLC	32	STA Adr	5D	MOV E,L	88	ADC B	B3	ORA E	DE	SBI D8
08	–	33	INX SP	5E	MOV E,M	89	ADC C	B4	ORA H	DF	RST 3
09	DAD B	34	INR M	5F	MOV E,A	8A	ADC D	B5	ORA L	E0	RPO
0A	LDAX B	35	DCR M	60	MOV H,B	8B	ADC E	B6	ORA M	E1	POP H
0B	DCX B	36	MVI M,D8	61	MOV H,C	8C	ADC H	B7	ORA A	E2	JPO Adr
0C	INR C	37	STC	62	MOV H,D	8D	ADC L	B8	CMP B	E3	XTHL
0D	DCR C	38	– –	63	MOV H,E	8E	ADC M	B9	CMP C	E4	CPO Adr
0E	MVI C,D8	39	DAD SP	64	MOV H,H	8F	ADC A	BA	CMP D	E5	PUSH H
0F	RRC	3A	LDA Adr	65	MOV H,L	90	SUB B	BB	CMP E	E6	ANI D8
10	– –	3B	DCX SP	66	MOV H,M	91	SUB C	BC	CMP H	E7	RST 4
11	LXI D,D16	3C	INR A	67	MOV H,A	92	SUB D	BD	CMP L	E8	RPE
12	STAX D	3D	DCR A	68	MOV L,B	93	SUB E	BE	CMP M	E9	PCHL
13	INX D	3E	MVI A,D8	69	MOV L,C	94	SUB H	BF	CMP A	EA	JPE Adr
14	INR D	3F	CMC	6A	MOV L,D	95	SUB L	C0	RNZ	EB	XCHG
15	DCR D	40	MOV B,B	6B	MOV L,E	96	SUB M	C1	POP B	EC	CPE Adr
16	MVI D,D8	41	MOV B,C	6C	MOV L,H	97	SUB A	C2	JNZ Adr	ED	
17	RAL	42	MOV B,D	6D	MOV L,L	98	SBB B	C3	JMP Adr	EE	ERI D8
18	– –	43	MOV B,E	6E	MOV L,M	99	SBB C	C4	CNZ Adr	EF	RST 5
19	DAD D	44	MOV B,H	6F	MOV L,A	9A	SBB D	C5	PUSH B	F0	RP
1A	LDAX D	45	MOV B,L	70	MOV M,B	9B	SBB E	C6	ADI D8	F1	POP PSW
1B	DCX D	46	MOV B,M	71	MOV M,C	9C	SBB H	C7	RST 0	F2	JP Adr
1C	INR E	47	MOV B,A	72	MOV M,D	9D	SBB L	C8	RZ	F3	DI
1D	DRC E	48	MOV C,B	73	MOV M,E	9E	SBB M	C9	RET Adr	F4	CP Adr
1E	MVI E,D8	49	MOV C,C	74	MOV M,H	9F	SBB A	CA	JZ	F5	PUSH PSW
1F	RAR	4A	MOV C,D	75	MOV M,L	A0	ANA B	CB	– –	F6	ORI D8
20	RIM	4B	MOV C,E	76	HLT	A1	ANA C	CC	CZ Adr	F7	RST 6
21	LXI H,D16	4C	MOV C,H	77	MOV M,A	A2	ANA D	CD	CALL Adr	F8	RM
22	SHLD Adr	4D	MOV C,L	78	MOV A,B	A3	ANA E	CE	ACI D8	F9	SPHL
23	INX H	4E	MOV C,M	79	MOV A,C	A4	ANA H	CF	RST 1	FA	JM Adr
24	INR H	4F	MOV C,A	7A	MOV A,D	A5	ANA L	D0	RNC	FB	EI
25	DCR H	50	MOV D,B	7B	MOV A,E	A6	ANA M	D1	POP D	FC	CM Adr
26	MVI H,D8	51	MOV D,C	7C	MOV A,H	A7	ANA A	D2	JNC Adr	FD	– –
27	DAA	52	MOV D,D	7D	MOV A,L	A8	XRA B	D3	OUT D8	FE	CPI D8
28	– –	53	MOV D,E	7E	MOV A,M	A9	XRA C	D4	CNC Adr	FF	RST 7
29	DAD H	54	MOV D,H	7F	MOV A,A	AA	XRA D	D5	PUSH D		
2A	LHLD Adr	55	MOV D,L	80	ADD B	AB	XRA E	D6	SUI D8		

D8 = constant, or logical/arithmetic expression that evaluates to an 8-bit data quantity. D16 = constant, or logical/arithmetic expression that evaluates to a 16-bit data quantity. Adr = 16-bit address.

THIS IS THE LAST PRINTED PAGE.

2521/302 **2602/302**
2601/302 **2603/302**
Oct./Nov. 2022