Nama: Willi Susanti NIM: L200180060

Kelas : B

## MODUL 1

1. Kode Standar Amerika untuk Pertukaran Informasi atau American Standard Code for Information Interchange (ASCII) merupakan suatu standar internasional dalam kode huruf dan simbol seperti Hex dan Unicode tetapi ASCII lebih bersifat universal, contohnya 124 adalah untuk karakter "|". Ia selalu digunakan oleh komputer dan alat komunikasi lain untuk menunjukkan teks. Kode ASCII sebenarnya memiliki komposisi bilangan biner sebanyak 7 bit. Namun, ASCII disimpan sebagai sandi 8 bit dengan menambakan satu angka 0 sebagai bit significant paling tinggi. Bit tambahan ini sering digunakan untuk uji paritas. Karakter control pada ASCII dibedakan menjadi 5 kelompok sesuai dengan penggunaan yaitu berturut-turut meliputi logical communication, Device control, Information separator, Code extention, dan physical communication. Code ASCII ini banyak dijumpai pada papan ketik (keyboard) computer atau instrument-instrument digital.

## TABEL ASCII:

Nilai ANSI ASCII			Karakte	
0	00	00000000	NUL	
1	01	0000001	SOH	
2	02	0000010	STX	
3	03	0000011	ETX	
4	04	00000100	EOT	
5	05	00000101	ENQ	
6	06	00000110	ACK	
7	07	00000111	BEL	
8	08	00001000	BS	
9	09	00001001	HT	
10	0A	00001010	LF	
11	OB	00001011	VT	
12	0C	00001100	FF	
13	0D	00001101	CR	
14	0E	00001110	SO	
15	0F	00001111	SI	
16	10	00010000	DLE	
17	11	00010001	DC1	
18	12	00010010	DC2	
19	13	00010011	DC3	
20	14	00010100	DC4	
21	15	00010101	NAK	
22	16	00010110	SYN	
23	17	00010111	ETB	
24	18	00011000	CAN	
25	19	00011001	EM	
26	1A	00011010	SUB	
27	1B	00011011	ESC	
28	1C	00011100	FS	
29	1D	00011101	GS	

20	1.5	00011110	DC
30	1E 1F	00011110	RS
31		00011111	US
32	20	00100000	space
33	21	00100001	!
34	22	00100010	#
35	23	00100011	
36	24	00100100	\$
37	25	00100101	%
38	26	00100110	&
39	27	00100111	
40	28	00101000	
41	29	00101001	*
42	2A	00101010	
43	2B	00101011	+
44	2C	00101100	ı
45	2D	00101101	-
46	2E	00101110	·
47	2F	00101111	/
48	30	00110000	0
49	31	00110001	1
50	32	00110010	2
51	33	00110011	3
52	34	00110100	4
53	35	00110101	5
54	36	00110110	6
55	37	00110111	7
56	38	00111000	8
57	39	00111001	9
58	3A	00111010	:
59	3B	00111011	;
60	3C	00111100	<
61	3D	00111101	=
62	3E	00111110	>
63	3F	00111111	?
64	40	01000000	@
65	41	01000001	Α
66	42	01000010	В
67	43	01000011	С
68	44	01000100	D
69	45	01000101	Е
70	46	01000110	F
71	47	01000111 G	
72	48	01001000 H	
73	49	01001001	ı
74	4A	01001010	J

75	4B	01001011	K
76	4C	01001100	L
77	4D	01001101	М
78	4E	01001110	N
79	4F	01001111	0
80	50	01010000	Р
81	51	01010001	Q
82	52	01010010	R
83	53	01010011	S
84	54	01010100	T
85	55	01010101	U
86	56	01010110	V
87	57	01010111	W
88	58	01011000	Х
89	59	01011001	Υ
90	5A	01011010	Z
91	5B	01011011	[
92	5C	01011100	\
93	5D	01011101	]
94	5E	01011110	٨
95	5F	01011111	_
96	60	01100000	`
97	61	01100001	a
98	62	01100010	b
99	63	01100011	С
100	64	01100100	d
101	65	01100101	е
102	66	01100110	f
103	67	01100111	g
104	68	01101000	h
105	69	01101001	i
106	6A	01101010	j
107	6B	01101011	k
108	6C	01101100	1
109	6D	01101101	m
110	6E	01101110	n
111	6F	01101111	0
112	70	01110000	р
113	71	01110001	q
114	72	01110010	r
115	73	01110011	S
116	74	01110100	t
117	75	01110101	u
118	76	01110110	V
119	77	01110111	W

127	7F	01111111	DEL
126	7E	01111110	~
125	7D	01111101	}
124	7C	01111100	
123	7B	01111011	{
122	7A	01111010	Z
121	79	01111001	у
120	78	01111000	x

## 2. Daftar Perintah Bahasa Assembly:

Perintah bahasa assembly lengkap untuk mesin intel keluarga dan 86. Digunakan sebagai pedoman untuk memahami program boot.asm dan kemel.asm.

## a. Assembly Directive

1. EQU => Detinisian konstanta.

2. DB => Definisian data dengan satuan 1 byte.

3. DW => Definisian data dengan satuan 1 word.

4. DBIT => Pendefinisian data dengan satuan 1 bit.

5. DS => Pemesanan tempat penyimpanan data di PAM.

6. ORG => Insiluasi alamat mulai program.

7. EHD => Penanda akhir program.

8. CSGC => Penempatan penanda dicode segment.

9. XSEG => Penanda penempatan di eksternal data.

10. DSEG => Penanda penempatan internal direct data.

11. 1SEG => Penanda penempatan di internal indirect data.

12. 13SEG => Penanda penempatan di bit data.

13. CODE => Penanda mulai pendefinisian program.

14. XDATA => Pendefinisian external data.

15. 1DATA => Pendefinisian indirect data.

16. BIT => Pendefinisian data bit.

17. INCLUDE => Mengikut sertakan file program lain.

18. DATA => Pendefinisian internal direct data

ACALL ADD ADD ADD ADD ADD ADD ADD ADD ADD	Instruksi	Keterangan Singkatan	
ADDC Add with Carry  AJMP Absolute Jump  ANL AND Logic  CJNE Compare and Jump if Not Equal  CLR Clear  CPL Complement  DA Decimal Adjust  DEC Decrement  DIV Divide  DJNZ Decrement and Jump if Not Zero  INC Increment  JB Jump if Bit Set  JBC Jump if Bit Set and Clear Bit  JC Jump if Carry Set  JMP Jump to Address  JNB Jump if Not Bit Set  JNC Jump if Not Bit Set  JNC Jump if Accumulator Not Zero  INC Jump if Accumulator Not Zero  JZ Jump if Accumulator Zero  LCALL Long Call  LJMP Long Jump  MOV Move from Memory  MOVC Move from Extended Memory  MOVX Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Interrupt  RL Rotate Left  RCC Rotate Left through Carry  SETB Set Bit  SMAP Swap Nibbles  XCH Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit  SUBH Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit	ACALL	Absolute Call	
AJMP Absolute Jump ANL AND Logic CJNE Compare and Jump if Not Equal CLR Clear CPL Complement DA Decimal Adjust DEC Decrement DIV Divide DJNZ Decrement and Jump if Not Zero INC Increment JB Jump if Bit Set JBC Jump if Bit Set and Clear Bit JC Jump if Carry Set JMP Jump to Address JNB Jump if Not Bit Set JNC Jump if Carry Not Set JNZ Jump if Accumulator Not Zero ICALL Long Call LJMP Long Jump MOV Move from Memory MOVC Move from Code Memory MOVX Move from Extended Memory MUL Multiply NOP No Operation ORL OR Logic POP Pop Value From Stack PUSH Push Value Onto Stack RET Return From Interrupt RL Rotate Left RRC Rotate Right through Carry SETB Set Bit SRT Return From Subrow SWAP Swap Nibbles XCH Exchange Digits XRL Exclusive OR Logic RRC Rotate Right through Carry SETB Set Bit SRT Return From Subrow SETB Set Bit RRC Rotate Right through Carry SETB Set Bit RRC Rotate Right through Carry SETB Set Bit SETB Set Bit RRC Rotate Right through Carry SETB Set Bit RRC Rotate Right through Carry SETB Set Bit RRC Rotate Right through Carry SETB Set Bit	ADD	Add	
ANL AND Logic CJNE Compare and Jump if Not Equal CLR Clear CPL Complement DA Decimal Adjust DEC Decrement DIV Divide DJNZ Decrement and Jump if Not Zero INC Increment JB Jump if Bit Set JBC Jump if Bit Set and Clear Bit JC Jump if Carry Set JMP Jump to Address JNB Jump if Not Bit Set JNC Jump if Carry Not Set JNZ Jump if Accumulator Not Zero ICALL Long Call LJMP Long Jump MOV Move from Memory MOVC Move from Code Memory MOVX Move from Extended Memory MUL Multiply NOP No Operation ORL OR Logic POP Pop Value From Stack PUSH Push Value Onto Stack RET Return From Interrupt RL Rotate Left RLC Rotate Left through Carry SETB Set Bit XRL Exclusive OR Logic RR Rotate Right through Carry SETB Set Bit SRTS	ADDC	Add with Carry	
CJNE Compare and Jump if Not Equal CLR Clear CPL Complement DA Decimal Adjust DEC Decrement DIV Divide DJNZ Decrement and Jump if Not Zero INC Increment JB Jump if Bit Set JBC Jump if Bit Set JBC Jump if Carry Set JMP Jump in Over Increment JNB Jump if Not Bit Set JNB Jump if Not Bit Set JNB Jump if Not Bit Set JNC Jump if Accumulator Not Zero JNZ Jump if Accumulator Not Zero JZ Jump if Accumulator Zero LCALL Long Call LJMP Long Jump MOV Move from Memory MOVC Move from Code Memory MOVX Move from Extended Memory MUL Multiply NOP No Operation ORL OR Logic POP Pop Value From Stack PUSH Push Value Onto Stack RET Return From Subroutine RETI Return From Interrupt RL Rotate Left RLC Rotate Left RCC Rotate Right RRC Rotate Right SWAP Swap Nibbles XCH Exchange Bytes XCH Exchange Bytes XCHD Exchange Digits XRL Exclusive OR Logic RRR Rotate Right RRC Rotate Right through Carry SETB Set Bit	AJMP	Absolute Jump	
CLR Clear CPL Complement DA Decimal Adjust DEC Decrement DIV Divide DJNZ Decrement and Jump if Not Zero INC Increment JB Jump if Bit Set JBC Jump if Bit Set and Clear Bit JC Jump if Carry Set JMP Jump if Not Bit Set JNB Jump if Not Bit Set JNC Jump if Accumulator Not Zero INC Jump if Accumulator Not Zero JZ Jump if Accumulator Zero LCALL Long Call LJMP Long Jump MOV Move from Memory MOVC Move from Code Memory MOVX Move from Extended Memory MUL Multiply NOP No Operation ORL OR Logic POP Pop Value From Stack PUSH Push Value Onto Stack RET Return From Interrupt RL Rotate Left RCC Rotate Left RRC Rotate Right Subract With Borrow SWAP Swap Nibbles XCH Exchange Bytes XCH Exchange Digits XRL Exclusive OR Logic RRR Rotate Right RRC Rotate Right through Carry SETB Set Bit	ANL	AND Logic	
CPL Complement DA Decimal Adjust DEC Decrement DIV Divide DJNZ Decrement and Jump if Not Zero INC Increment JB Jump if Bit Set JBC Jump if Bit Set and Clear Bit JC Jump if Carry Set JMP Jump to Address JNB Jump if Not Bit Set JNC Jump if Carry Not Set JNZ Jump if Accumulator Not Zero JZ Jump if Accumulator Not Zero JZ Jump if Accumulator Recomplement LJMP Long Jump MOV Move from Memory MOVC Move from Code Memory MOVX Move from Extended Memory MUL Multiply NOP No Operation ORL OR Logic POP Pop Value From Stack PUSH Push Value Onto Stack RET Return From Interrupt RL Rotate Left RCC Rotate Right RRC Rotate Right through Carry SETB Set Bit SJMP Swap Nibbles XCH Exchange Bytes XCH Exchange Bytes XCH Exchange Digits XRL Exclusive OR Logic RRR Rotate Right through Carry SETB Set Bit RRC Rotate Right through Carry SETB Set Bit	CJNE	Compare and Jump if Not Equal	
DA Decimal Adjust  DEC Decrement  DIV Divide  DJNZ Decrement and Jump if Not Zero  INC Increment  JB Jump if Bit Set  JBC Jump if Bit Set and Clear Bit  JC Jump if Carry Set  JMP Jump to Address  JNB Jump if Not Bit Set  JNC Jump if Carry Not Set  JNZ Jump if Accumulator Not Zero  JZ Jump if Accumulator Zero  LCALL Long Call  LJMP Long Jump  MOV Move from Memory  MOVC Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit  SMAP Swap Nibbles  XCH Exchange Bytes  XCH Exchange Bytes  XCH Exchange Digits  XRL Exclusive OR Logic  RRC Rotate Right through Carry  SETB Set Bit  RRC Rotate Right  RRC Rotate Right through Carry  SETB Set Bit	CLR	Clear	
DEC Decrement DIV Divide DJNZ Decrement and Jump if Not Zero INC Increment JB Jump if Bit Set JBC Jump if Bit Set and Clear Bit JC Jump if Carry Set JMP Jump to Address JNB Jump if Not Bit Set JNC Jump if Carry Not Set JNZ Jump if Accumulator Not Zero JZ Jump if Accumulator Zero LCALL Long Call LJMP Long Jump MOV Move from Memory MOVC Move from Code Memory MOVX Move from Extended Memory MUL Multiply NOP No Operation ORL OR Logic POP Pop Value From Stack PUSH Push Value Onto Stack RET Return From Interrupt RL Rotate Left RLC Rotate Left through Carry RR Rotate Right RRC Rotate Right through Carry SETB Set Bit SJMP Short Jump SUBB Subtract With Borrow SWAP Swap Nibbles XCH Exchange Bytes XCHD Exchange Digits XRL Exclusive OR Logic RRR Rotate Right through Carry SETB Set Bit RRC Rotate Right through Carry SETB Set Bit	CPL	Complement	
DIV Divide  DJNZ Decrement and Jump if Not Zero  INC Increment  JB Jump if Bit Set  JBC Jump if Bit Set and Clear Bit  JC Jump if Carry Set  JMP Jump to Address  JNB Jump if Not Bit Set  JNC Jump if Carry Not Set  JNZ Jump if Accumulator Not Zero  JZ Jump if Accumulator Zero  LCALL Long Call  LJMP Long Jump  MOV Move from Memory  MOVC Move from Code Memory  MOVX Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Interrupt  RL Rotate Left  RCC Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCH Exchange Digits  XRL Exclusive OR Logic  RRC Rotate Right through Carry  SETB Set Bit  RRC Rotate Right  XRL Exclusive OR Logic  RR Rotate Right  RRC Rotate Right  RRC Rotate Right through Carry  SETB Set Bit	DA	Decimal Adjust	
DJNZ Decrement and Jump if Not Zero INC Increment JB Jump if Bit Set JBC Jump if Bit Set and Clear Bit JC Jump if Carry Set JMP Jump to Address JNB Jump if Not Bit Set JNC Jump if Carry Not Set JNZ Jump if Accumulator Not Zero JZ Jump if Accumulator Zero LCALL Long Call LJMP Long Jump MOV Move from Memory MOVC Move from Code Memory MOVX Move from Extended Memory MUL Multiply NOP No Operation ORL OR Logic POP Pop Value From Stack PUSH Push Value Onto Stack RET Return From Interrupt RL Rotate Left RLC Rotate Left through Carry RR Rotate Right RRC Rotate Right through Carry SETB Set Bit SUMP Swap Nibbles XCH Exchange Bytes XCH Exchange Digits XRL Exclusive OR Logic RRC Rotate Right through Carry SETB Set Bit RRC Rotate Right through Carry SETB Set Bit	DEC	Decrement	
INC Increment JB Jump if Bit Set JBC Jump if Bit Set and Clear Bit JC Jump if Carry Set JMP Jump to Address JNB Jump if Not Bit Set JNC Jump if Carry Not Set JNC Jump if Carry Not Set JNZ Jump if Accumulator Not Zero JZ Jump if Accumulator Zero LCALL Long Call LJMP Long Jump MOV Move from Memory MOVC Move from Code Memory MOVX Move from Extended Memory MUL Multiply NOP No Operation ORL OR Logic POP Pop Value From Stack PUSH Push Value Onto Stack RET Return From Subroutine RETI Return From Interrupt RL Rotate Left RLC Rotate Left through Carry RR Rotate Right RRC Rotate Right through Carry SETB Set Bit SMP Short Jump SUBB Subtract With Borrow SWAP Swap Nibbles XCH Exchange Bytes XCH Exchange Digits XRL Exclusive OR Logic RRC Rotate Right through Carry SETB Set Bit	DIV	Divide	
JBB Jump if Bit Set JBC Jump if Bit Set and Clear Bit JC Jump if Carry Set JMP Jump to Address JNB Jump if Not Bit Set JNC Jump if Carry Not Set JNC Jump if Carry Not Set JNZ Jump if Accumulator Not Zero JZ Jump if Accumulator Zero LCALL Long Call LJMP Long Jump MOV Move from Memory MOVC Move from Code Memory MOVX Move from Extended Memory MUL Multiply NOP No Operation ORL OR Logic POP Pop Value From Stack PUSH Push Value Onto Stack RET Return From Interrupt RL Rotate Left RLC Rotate Left through Carry RR Rotate Right RRC Rotate Right through Carry SETB Set Bit SUBB Subtract With Borrow SWAP Swap Nibbles XCH Exchange Digits XRL Exclusive OR Logic RRC Rotate Right through Carry SETB Set Bit RRC Rotate Right through Carry SETB Set Bit	DJNZ	Decrement and Jump if Not Zero	
JBC Jump if Bit Set and Clear Bit  JC Jump if Carry Set  JMP Jump to Address  JNB Jump if Not Bit Set  JNC Jump if Carry Not Set  JNZ Jump if Accumulator Not Zero  JZ Jump if Accumulator Zero  LCALL Long Call  LJMP Long Jump  MOV Move from Memory  MOVC Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Interrupt  RL Rotate Left  RLC Rotate Right through Carry  SETB Set Bit  SUBB Subtract With Borrow  SVAP Swap Nibbles  XCH Exchange Bytes  RCC Rotate Right through Carry  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit  Set Bit	INC	Increment	
JC Jump if Carry Set  JMP Jump to Address  JNB Jump if Not Bit Set  JNC Jump if Carry Not Set  JNZ Jump if Accumulator Not Zero  JZ Jump if Accumulator Zero  LCALL Long Call  LJMP Long Jump  MOV Move from Memory  MOVC Move from Code Memory  MOVX Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SWAP Swap Nibbles  XCH Exchange Bytes  XCH Exchange Digits  RRC Rotate Right through Carry  SETB Set Bit  SCH Bit  RRC Rotate Right  RRC Rotate Right Push  SWAP Swap Nibbles  XCH Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit  RRC Rotate Right through Carry  SKAL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit	JB	Jump if Bit Set	
JMP Jump to Address  JNB Jump if Not Bit Set  JNC Jump if Carry Not Set  JNZ Jump if Accumulator Not Zero  JZ Jump if Accumulator Zero  LCALL Long Call  LJMP Long Jump  MOV Move from Memory  MOVC Move from Code Memory  MOVX Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Subroutine  RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCH Exchange Bytes  XRL Exclusive OR Logic  RRC Rotate Right through Carry  SETB Set Bit  RRC Rotate Right  RRC Rotate Right through Carry	JBC	Jump if Bit Set and Clear Bit	
JNB Jump if Not Bit Set  JNC Jump if Carry Not Set  JNZ Jump if Accumulator Not Zero  JZ Jump if Accumulator Zero  LCALL Long Call  LJMP Long Jump  MOV Move from Memory  MOVC Move from Code Memory  MOVX Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  RET Return From Subroutine  RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right through Carry  SETB Set Bit  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCH Exchange Digits  XRL Exclusive OR Logic  RRC Rotate Right through Carry  SETB Set Bit  SATE Return From Subroutine  RRC Rotate Right Horough Carry  SETB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCH Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit	JC	Jump if Carry Set	
JNC Jump if Carry Not Set  JNZ Jump if Accumulator Not Zero  JZ Jump if Accumulator Zero  LCALL Long Call  LJMP Long Jump  MOV Move from Memory  MOVC Move from Code Memory  MOVX Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Subroutine  RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SWAP Swap Nibbles  XCH Exchange Bytes  XCH Exchange Digits  XRL Exclusive OR Logic  RRC Rotate Right through Carry  SETB Set Bit  SXRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit  SKET SCH Sit	JMP	Jump to Address	
JNZ Jump if Accumulator Not Zero  JZ Jump if Accumulator Zero  LCALL Long Call  LJMP Long Jump  MOV Move from Memory  MOVC Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Subroutine  RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XRL Exclusive OR Logic  RRC Rotate Right through Carry  SETB Set Bit  SXRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit  SALL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit  SYAL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit	JNB	Jump if Not Bit Set	
JZ Jump if Accumulator Zero  LCALL Long Call  LJMP Long Jump  MOV Move from Memory  MOVC Move from Code Memory  MOVX Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Subroutine  RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCH Exchange Digits  XRL Exclusive OR Logic  RRC Rotate Right through Carry  SETB Set Bit  SCH Rotate Right  RRC Rotate Right  RRC Rotate Right  RRC Rotate Right  SWAP Swap Nibbles  XCH Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit	JNC	Jump if Carry Not Set	
LCALL  LJMP  Long Jump  MOV  Move from Memory  MOVC  Move from Code Memory  MOVX  Move from Extended Memory  MUL  Multiply  NOP  No Operation  ORL  OR Logic  POP  Pop Value From Stack  PUSH  Push Value Onto Stack  RET  Return From Subroutine  RETI  Return From Interrupt  RL  Rotate Left  RLC  Rotate Left through Carry  RR  Rotate Right  RRC  Set Bit  SJMP  Subtract With Borrow  SWAP  Swap Nibbles  XCH  Exchange Bytes  XRL  Exclusive OR Logic  RRC  Rotate Right through Carry  SETB  Set Bit  SCH  Exchange Digits  XRL  Exclusive OR Logic  RR  ROTATE Right through Carry  SETB  Set Bit  SALL  Exclusive OR Logic  RR  ROTATE Right through Carry  SETB  Set Bit  SALL  Exclusive OR Logic  RR  ROTATE Right through Carry  SETB  Set Bit	JNZ	Jump if Accumulator Not Zero	
LJMP Long Jump  MOV Move from Memory  MOVC Move from Code Memory  MOVX Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Subroutine  RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCH Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit  SALE Rotate Right Through Carry  SETB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit	JZ	Jump if Accumulator Zero	
MOV Move from Memory  MOVC Move from Code Memory  MOVX Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Subroutine  RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit  SALE Rotate Right  RRC Rotate Right through Carry  SETB Set Bit	LCALL	Long Call	
MOVC Move from Code Memory  MOVX Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Subroutine  RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit  SAL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit	LJMP	Long Jump	
MOVX Move from Extended Memory  MUL Multiply  NOP No Operation  ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Subroutine  RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit  SAL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit	MOV	Move from Memory	
MUL Multiply NOP No Operation ORL OR Logic POP Pop Value From Stack PUSH Push Value Onto Stack RET Return From Subroutine RETI Return From Interrupt RL Rotate Left RLC Rotate Left through Carry RR Rotate Right RRC Rotate Right through Carry SETB Set Bit SJMP Short Jump SUBB Subtract With Borrow SWAP Swap Nibbles XCH Exchange Bytes XCHD Exchange Digits XRL Exclusive OR Logic RR Rotate Right through Carry SETB Set Bit SJMP Short Jump	MOVC	Move from Code Memory	
NOP  No Operation  ORL  OR Logic  POP  Pop Value From Stack  PUSH  Return From Subroutine  RETI  Return From Interrupt  RL  Rotate Left  RLC  Rotate Left through Carry  RR  Rotate Right  RRC  Rotate Right through Carry  SETB  Set Bit  SJMP  Short Jump  SUBB  Subtract With Borrow  SWAP  Swap Nibbles  XCH  Exchange Bytes  XCHD  Exchange Digits  XRL  RRC  Rotate Right through Carry  SETB  Set Bit  SUBH  SWAP  Swap Nibbles  XCH  Exchange Digits  XRL  Exclusive OR Logic  RR  Rotate Right through Carry  SETB  Set Bit	MOVX	Move from Extended Memory	
ORL OR Logic  POP Pop Value From Stack  PUSH Push Value Onto Stack  RET Return From Subroutine  RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit	MUL	Multiply	
POP Pop Value From Stack PUSH Push Value Onto Stack RET Return From Subroutine RETI Return From Interrupt RL Rotate Left RLC Rotate Left through Carry RR Rotate Right RRC Rotate Right through Carry SETB Set Bit SJMP Short Jump SUBB Subtract With Borrow SWAP Swap Nibbles XCH Exchange Bytes XCHD Exchange Digits XRL Exclusive OR Logic RR Rotate Right through Carry SETB Set Bit	NOP	No Operation	
PUSH Push Value Onto Stack  RET Return From Subroutine  RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit	ORL	OR Logic	
RET Return From Subroutine RETI Return From Interrupt RL Rotate Left RLC Rotate Left through Carry RR Rotate Right RRC Rotate Right through Carry SETB Set Bit SJMP Short Jump SUBB Subtract With Borrow SWAP Swap Nibbles XCH Exchange Bytes XCHD Exchange Digits XRL Exclusive OR Logic RR Rotate Right through Carry SETB Set Bit	POP	Pop Value From Stack	
RETI Return From Interrupt  RL Rotate Left  RLC Rotate Left through Carry  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right through Carry  SETB Set Bit	PUSH	Push Value Onto Stack	
RL Rotate Left RLC Rotate Left through Carry RR Rotate Right RRC Rotate Right through Carry SETB Set Bit SJMP Short Jump SUBB Subtract With Borrow SWAP Swap Nibbles XCH Exchange Bytes XCHD Exchange Digits XRL Exclusive OR Logic RR Rotate Right RRC Rotate Right through Carry SETB Set Bit	RET	Return From Subroutine	
RLC Rotate Left through Carry  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit	RETI	Return From Interrupt	
RR Rotate Right RRC Rotate Right through Carry SETB Set Bit SJMP Short Jump SUBB Subtract With Borrow SWAP Swap Nibbles XCH Exchange Bytes XCHD Exchange Digits XRL Exclusive OR Logic RR Rotate Right RRC Rotate Right through Carry SETB Set Bit	RL	Rotate Left	
RRC Rotate Right through Carry  SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit	RLC	Rotate Left through Carry	
SETB Set Bit  SJMP Short Jump  SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit		-	
SJMP Short Jump SUBB Subtract With Borrow SWAP Swap Nibbles XCH Exchange Bytes XCHD Exchange Digits XRL Exclusive OR Logic RR Rotate Right RRC Rotate Right through Carry SETB Set Bit	RRC	Rotate Right through Carry	
SUBB Subtract With Borrow  SWAP Swap Nibbles  XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit	SETB	Set Bit	
SWAP Swap Nibbles  XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit		Short Jump	
XCH Exchange Bytes  XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit		Subtract With Borrow	
XCHD Exchange Digits  XRL Exclusive OR Logic  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit		'	
XRL Exclusive OR Logic  RR Rotate Right  RRC Rotate Right through Carry  SETB Set Bit			
RR Rotate Right RRC Rotate Right through Carry SETB Set Bit			
RRC Rotate Right through Carry SETB Set Bit	XRL	Exclusive OR Logic	
SETB Set Bit		Rotate Right	
		Rotate Right through Carry	
		Set Bit	