4) **LAHF**: This instruction load low order 8 bit of flag register into AH register

5) **SAHF**: Store the content of AH register to flag register

→ **XLAT**/**XUTB**: This instruction read byte from look up table mainly used for code conversion. One code is placed in AL corresponding other code in lookup table

→ **XCHG R|M**:  

```
PUSH R|M: This instruction push the contents of 16 bit register on to the top of stack
```

```
POP R|M: This instruction pops from top of the stack and specify the content of 16 bit register
```

```
IN Port or IN AX or AL, DX: This instruction read 8 or 16 bit data from code whose address in DX register
```

```
OUT Port: 
```

6) **Arithmetic Register**: 

```
ADD RR, ADD MR, ADD M|R : ADD AX[2000H]
```

```
ADC RR, ADC RM, ADC MR
```

```
SUB RR, R|MR, M|R, or SBB RR, R|MR, M|MR
```
Multiplication
flag affected
MUL R1M: This instruction multiply two unsigned
overflow copy no if content of Specified register as memory
flag will location is 8 bit it will be multiplied by a 1 register
have chance and register stored in A 16 bit the A register

IMUL R1M: Signed no multiplication

In case of divide D, V is used
DIV R1M
IDIV R1M

Increment-Decrement: flag affected (overflow, sign flag,
zero, parity)
INC R1M
DEC R1M

DAD: It is used after AND instruction to get correct
result in BCD. This instruction for 8 bit operation Sign flag,
zero flag, carry flag, auxiliary flag & overflow flag

DAS: Decimal Adjust After Subtraction: This instruction
is used to get result in BCD when BCD no is
subtracted from another BCD no and used for 8 bit
operation.
(Conversion into Binary)

ASCII Adjust: for Addition: Numeric data which come from an input device into a microprocessor are usually in ASCII code. 8026 allows addition of ASCII code of two decimal digits. This is used to get correct result in BCD format.

Auxillary flag & carry flag will affect.

AAD: Adjust AX register for division. This instruction converts to BCD digit in AX into a binary format. This is done by dividing two BCD digit in AX register by another BCD byte.

AA M: Adjust result of BCD multiplication. Used for multiplication of two BCD no to get correct binary result & the finally result is placed in AL register.

Sign flag, zero flag, parity flag will get affected.

No flag will affect.

CBW: Convert Byte Word. This instruction copies sign bit of an operand into a AL register to all bit of AH register. This is used before signed operation in AL and also known as convert of signed byte to signed word.

→ CWD: Convert Word Double Word. No flag will after

→ NEG M/R: We can obtain two's complement from this instruction.
- Compare instruction: AU flag will affect (6 flag will be affected)

  CMP R/M
  CMP data
  CMP M,R
  CMP M,data

- Logical instruction:
  1) AND R/M
  2) OR R/M
  3) XOR R/M

  AND data
  OR data
  XOR data
  AND M,R
  OR M,R
  XOR M,R
  AND M,data
  OR M,data
  XOR M,data

  Parity, carry, sign, zero flag

- NOT H/R: This instruction is used for 1's complement operation

- TEST instruction: Related to arithmetic operation, but result will not be stored only required flag affected depending upon the operation

- Rotate instruction

  - Rotate data with carry (RCL R/M, count); Rotate left through carry depending upon count operation. Count may be 1 depending upon the contents may be present in memory.

  RCL AX, count = 1

  RCL AX [2007]
RCR R:
RCR AX, count = 1
RCR AX [2000H]

Rotation without carry (ROL R/M, count): This instruction rotates all bit of operand contain 8 bit or 16 bit register or memory location left by specific no of bits. The MSB of operand moved to LSB position of the operand as well as to the Carry flag.

ROR R:
ROR AX, count = 1

→ Shift operation
1) SA1 | SHL M|R, count: Shift each bit of operand by specified no of bits that is shifting.
2) SAR | SHR M|R, count: Max. 16 bit operation.

Branch instructions:
JM: Jump instruction

JA: Jump if Above: JA 8 bit displacement
JNBE: 8 bit displacement
JAE: Jump if a move or may be equal to 8 bit displacement
JNB: Jump if not Below 8 bit displacement
JNC: Jump if no Carry 8 bit displacement.
TB: Jump if Below when carry flag = 1 - 8 bit displac. 

JNAE: Jump if not above & equal (carry flag = 1) 

JE: Jump if carry 

JBE: Jump if below & equal if zero flag = 1 

JNA: Jump if Not above but but zero flag = 1 

JABE 

JE: Jump if Equal if zero flag = 1 

JZ: Jump if zero when zero flag = 1 

JG: Jump if greater 

JMP 16 bit address or JMP 8 or 16 bit displacement 

JMP M

→ Call Instruction 

CALL MIR1 address (8 or 16 bit displacement) (Unconditional) 

RET MIR1 address / 8 or 16 bit displacement 

IRET: Return for interrupt service subroutine.
If external IC interrupt the processor then this instruction will be valid.

Interrupt instructions:

1) INT n  
   - n Stand from 0 to 256

INTO: This interrupt will occur on integer overflow.

loop instruction

→ loop displacement [JUMP to specified label until CX register is zero]

loop C

loop Z [Decrement CX register and Jump if CX is not equal to 0 and ZF = 1]

loop LOOP C/ LOOP LE

Flag Manipulation instruction:

1) CLC:
   - CLD: Clear direction flag
   - CLL: Clear loop
   - CMC: Complement carry flag
   - SIC: Set carry flag
   - STD: Set directed
   - STI

HLT → Stop
NOP → No operation
ESC → Escape. This instruction makes bus free from external peripheral device.
WAIT: when executed processor enter in idle state until test bar pin is made low

lock # it is related with

string instruction
(related with extra segment)
Mivos / MOV SW / MOVSB D Moveh H to M.

CMPS / CMPSW / CMPSB E Compare H to M.

Sca S / SCASD / SCASB E Scan Memory to Memory
G+ Compare contents of of accumulator with
content of memory string to be scanned by
the extra segment register

LodS , LODSB, LODSW D loading H to M
8 or 16 bit data from memory address by AX
register

Stos , STOSB / STOSW E Store content AX register to
IX register

Rep: Repeat Instruction

Rep: This instruction repeat string until CX=0

Repe: Repeat if equal repeat if CX > 0 D ZF=1

Repl if CX > 0 E ZF=0