KHULNA UNIVERSITY OF ENGINEERING & TECHNOLOGY

KUET

SESSIONAL REPORT

Course No. CSE-2204

Department Of CSE

Experiment No. 01

Name of the Experiment 8086 Instruction descriptions

Remarks

Name MD. Atique Faisal

Roll No 1907038

Group No. A2

Year 2nd

Semester 2nd

Date of Performance 15/09/22

Date of Submission 19/09/22

Instruction:

AAA → ASCII Adjust For Addition

This instruction is used to make sure the result is the correct unpacked BCD.

```
ADD  AL,BL;
AAA ;
```

The AAA instruction works only on the AL register.

AAD → BCD to binary convert before Division

It converts two unpacked BCD digits in AH and AL to the equivalent binary number in AL

```
AAD;
DIV CH;
```

ADD → Addition (ADD destination, source)

It adds two numbers.

```
ADD Dx, Bx
```

ADC → Add with carry

It adds the status of the carry flag into the result.

```
ADD AL, 74H;
ADC CL, BL
```

AND → AND corresponding bits of two operands

This instruction ANDs each bit in a source byte or word with the same number bit in a destination byte or word.

```
AND BH, CL;
```

CALL → Call a procedure

It is used to transfer execution to sub-program or procedure.

```
CALL MULTO
```

CBW → convert signed byte to signed word

It copies the sign of a byte in AL to all the bits in AH.

CBW ;

CLC → Clear the carry flag

It resets the carry flag to 0.

CLC;

CLD → Clear direction flag-

It resets the direction flag to 0.

CLD;

CLI → clear interrupt flag

It resets the interrupt flag to 0.

CLI;

CMC → Complement the carry flag

If ~~carry~~ It resets carry flag 0 to 1 or 1 to 0.

CMC;

CMPS/CMPSB/CMPSW → Compare string bytes

It can be used to compare a byte in one string with a byte in another string.

```
REPE CMPS;
```

CWD → Convert signed word to signed doubleword

It copies the sign bit of a word in AX to all the bits of the DX register.

```
CWD;
```

DAA → Decimal adjust AL after BCD addition

It is used to make sure the result of adding two packed BCD numbers is adjusted to be a legal BCD number.

```
ADD AL, BL;
DAA;
```

DIV → Division

Divides the contents of GP register concatenated with MQ register by contents of GP register and stores the result in GP register.

```
DIV RT, RA, RB
```

ESC → Escape

It is used to pass instructions to a coprocesso
such as

HLT → Hall Processing

It will cause the 8086 to stop fetching and executing instructions.

IN → Copy data from a port

It will copy data from a port to the AL or AX register.

```
IN AL, OC8H;
```

INC → Increment

It adds 1 to a specified register or to a memory location specified in any one of the 24 ways. ~~shown~~

```
INC BL;
INC CX;
```

INT → Interrupt program execution

The term 'type' in the instruction formate refers to a number between 0 and 255 which identifies the interrupt.

```
INT 35;
```

JA/JNBE → Jump if above/jump if not below or equal

The number 0111 is above the number 0010. If after a compare or some other instruction which affects flage, the zero flag and carry flag are both 0, this instruction will cause execution to jump to a label given in the instruction.

```
CMP AX, 4371H;
JA  RUN-PRESS;
```

JNS → Jump if not signed

It will cause execution to jump a specified destination if the sign flag is 0.

```
DEC AL;
JNS REDO;
```

JNO → Jump if $\overset{no}{\wedge}$ Overflow

The overflow flag will be set if the result of some signed arithmetic operation is too large to fit in the destination register.

```
ADD AL, BL;
JNO DONE;
```

JO → Jump if overflow

It will cause the 8086 to jump to a destination given in the instruction if the overflow flag is set.

```
ADD AL,BL;
JO  ERROR;
```

LEA → Load effective address

It determines the offset of the variable mamed as source and puts this offset in the indicated 16-bit Register.

```
LEA BX, PRICES;
```

LODS/LODSB/LODSW → Load String byte into AL or Load string word into AX

```
LODS SOURCE_STRING;
```

MOV → copy a word or byte

It copies a word or byte of data from a specified source to a specified destination.

```
MOV AX, BX;
```

MUL → Multiply unsigned bytes or words

It multiplies an unsigned byte from some source times an unsigned byte in the AL register.

```
MUL BH;
MUL CX;
```

NEG → Form 2's complement

NEG AL;
NEG BX;

NOT → Invert each bit

NOT Bx;

OR → Logically OR corresponding bits of 2 operands

OR AH, CL;

OUT → Output a byte or word to a port

OUT 3BH, AL;

POP → POP destination

It copies a word from stack location pointed to by the stack pointer to destination.

POP Dx;

PUSH → PUSH source

It decrements the stack pointer by 2 and copies word from source to the location in the sack segment where the stack pointer then points

PUSH BX;

RCL → Rotate operand around to the left through CF

RCL DX, 1;

SAL/SHL → shift operand bits left,
→ Put zero in LSB

SAL AL, CL;

SAR → shift operand bits right
→ New MSB = old MSB

SAR AL, 1;

STC → set carry flag to a 1

STD → set direction flag to a 1

STI → set Interrupt flag

WAIT → wait for test or interrupt signal

XOR → Exclusive OR operation