

M.I.T. LAB Assignment – 08

U19CS012

1. Write a program to convert a given number of *binary* data bytes into their *BCD* equivalents, and store them as unpacked BCDs in the output buffer.

The number of data bytes is specified in register D in the main program.

The converted numbers should be stored in groups of three consecutive memory locations. If the number is not large enough to occupy all three locations, Zeros should be loaded in those locations.

Notepad Code:

```
2 ; Q-(1) WAP to convert a given number of binary data bytes into their BCD equivalents
3 ; & Store them as unpacked BCDs in the output buffer [3000H Onwards]
4 ; The number of data bytes is specified in register D in the main program.
5
6 MVI D, 05H ; COUNTER = Number of Data Bytes [Elements to Convert]
7 LXI H, 2000H ; Starting Address of First Data Byte
8 LXI B, 3000H ; Starting Address of Destination Location
9
10 MAIN: MOV A, M ; Get the Data in Accumulator
11 CALL BIN_TO_BCD
12 DCR D ; Decrement the Counter 'D'
13 INX H ; [HL] Points to Next Location
14 JNZ MAIN ; Until Counter 'D' Reaches Zero, Keep on Reading Data
15 HLT
```

```


16
17 BIN_TO_BCD: PUSH H      ; Store HL Location Safely
18             MVI H, 00H  ; Hundreds Place Counter
19             MOV L, H    ; Tens Place Counter
20
21 HUNDRED:    CPI 64H      ; Compare with 100
22             JC TENS     ; If Less than 100, Then CY = 1, So Goto Tens Place
23             SUI 64H     ; Subtract 100
24             INR H       ; Increment Hundreds Counter
25             JMP HUNDRED
26
27 TENS:       CPI 0AH      ; Compare with 10
28             JC UNITS    ; If Less the 10, Then CY = 1, So Goto Units Place
29             SUI 0AH     ; Subtract 10
30             INR L       ; Increment Tens Counter
31             JMP TENS
32
33 UNITS:      STAX B       ; Push the Units Digit = A at Location Pointer by BC Pair
34             INX B
35             MOV A, L     ; Get Tens Counter
36             STAX B
37             INX B
38             MOV A, H     ; Get Hundred Counter
39             STAX B
40             INX B
41             POP H       ; Get the Original [HL] Location Back
42 RET

```

Input: Counter D = 5 Numbers, {124,47,6,96,189}


Start

Address (Hex)	Address	Data
2000	8192	124
2001	8193	47
2002	8194	6
2003	8195	96
2004	8196	189
2005	8197	0
2006	8198	0
2007	8199	0
2008	8200	0

 **5 Numbers**

Output:

Address (Hex)	Address	Data
3000	12288	4
3001	12289	2
3002	12290	1
3003	12291	7
3004	12292	4
3005	12293	0
3006	12294	6
3007	12295	0
3008	12296	0
3009	12297	6
300A	12298	9
300B	12299	0
300C	12300	9
300D	12301	8
300E	12302	1



2. A set of ten BCD readings is stored in the Input Buffer. Convert the numbers into binary and add the numbers. Store the sum in the Output Buffer, the sum can be larger than FFH.

Notepad Code:

```
2 ; (2) - A set of ten BCD readings is stored in the Input Buffer.
3 ;Convert the numbers into binary and add the numbers.
4 ;Store the sum in the Output Buffer, the sum can be larger than FFH.
5
6 ; INPUT : Enter 10 Numbers from Location 2000H Onwards
7 ; OUTPUT: Get Sum at Address 3001 & 3000
8
9 MVI B, 0AH ; Count of Numbers to Sum
10 LXI D, 2000H ; Location of First Data Byte
11 LXI H, 0000H ; [HL] Pair Stores , Sum = 0 [Initially]
12
13 MAIN: CALL BCD_TO_BIN
14 ; The Binary Number is Stored in A
15 MOV C, A ; Store the Binary Number in C
16 MOV A, B ; Store the Counter 'B'
17 MVI B, 00H ; [BC] = [0000 Binary_Answer]
18 DAD B ; [HL] <- [HL] + [BC]
19 INX D ; Point to Next Location
20 MOV B, A ; Restore Back the Counter Value in B
21 DCR B
22 JNZ MAIN
23 SHLD 3000H ; Answer of Sum is Stored at Location 3000H
24 HLT
```

```

25
26 BCD_TO_BIN:  PUSH B    ; Store Value of BC Register in Stack
27              PUSH H    ; Store Value of HL Register in Stack [To Free Registers]
28              LDAX D     ; Get Data from Location Pointed by [DE]
29              ANI 0FH    ; Mask Upper 4 Bits : [0000 abcd] => Get Lower 4 Bits
30              MOV B, A   ; B <= Lower 4 Bits
31              LDAX D     ; Get Data from Location Pointed by [DE]
32              ANI 0F0H   ; Mask Lower 4 Bits : [abcd 0000] => Get Upper 4 Bits
33              RRC        ; [0abc d000]
34              RRC        ; [00ab cd00]
35              RRC        ; [000a bcd0]
36              RRC        ; [0000 abcd]
37              MOV H, A   ; H <= Upper 4 Bits
38              MVI C, 09H
39 MULTIPLY:    ADD H      ; (Upper 4 Bits)x10 + (Lower 4 Bits)
40              DCR C
41              JNZ MULTIPLY
42              ADD B      ; Add Lower 4 Bits
43              POP H      ; Restore Back the [HL] Values
44              POP B      ; Restore Back the [BC] Values
45              RET

```

Input:

Address (Hex)	Address	Data	BINARY	B.C.D. CODE
2000	8192	18	0001 0010	12
2001	8193	48	0011 0000	30
2002	8194	119	0111 0111	77
2003	8195	50	0011 0010	32
2004	8196	37	0010 0101	25
2005	8197	1	0000 0001	01
2006	8198	16	0001 0000	10
2007	8199	82	0101 0010	52
2008	8200	53	0011 0101	35
2009	8201	21	0001 0101	15
SUM =				289
				[01 21] H = (289)

Output:

Address (Hex)	Address	Data	Hexadecimal
3000	12288	33	21 H
3001	12289	1	01 H
= 0121 H = (289)			

3. A set of ASCII Hex digits is stored in the Input Buffer memory. Write a program to convert these numbers into binary. Add these numbers in binary, and store the result in the Output-Buffer memory.

Notepad Code:

```
2 ; (Q)-3 A set of ASCII Hex digits is stored in the Input Buffer memory.
3 ; Write a program to convert these numbers into Binary.
4 ; Add these numbers in Binary, and store the result in the Output-Buffer memory.
5
6 MVI B, 0AH ; Counter
7 MVI C, 00H ; Register to Store Sum
8 LXI D, 2000H ; Source Pointer for ASCII [INPUT]
9 LXI H, 3000H ; Destination Pointer for Equivalent Binary [OUTPUT]
10
11 MAIN: LDAX D ; Get the Input in Accumulator
12 CALL ASC_TO_BIN
13 MOV M, A ; Storing the Binary Equiv to Location by [HL]
14 MOV A, C ; Get the Sum
15 ADD M ; Add the Binary Equiv to Sum
16 MOV C, A ; Store C <- C + Binary_Equiv
17 INX H ; Point to Next Location [INPUT]
18 INX D ; Point to Next Location [OUTPUT]
19 DCR B ; Decrement the Counter
20 JNZ MAIN
21 MOV M, C ; Final Sum Stored
22 HLT
23
24 ; INPUT (A) : ASCII Code
25 ; OUTPUT (A) : BINARY Equivalent
26
27 ASC_TO_BIN: SUI 30H ; Subtract 30H to Get Numeric Value
28 CPI 0AH ; If Less than 10, No Need to Convert [Return]
29 RC
30 SUI 07H ; For A - F
31 RET
```

Input:

Address (Hex)	Address	Data
2000	8192	48
2001	8193	49
2002	8194	50
2003	8195	52
2004	8196	55
2005	8197	58
2006	8198	61
2007	8199	63
2008	8200	68
2009	8201	69

Output:

Address (Hex)	Address	Data
3000	12288	0
3001	12289	1
3002	12290	2
3003	12291	4
3004	12292	7
3005	12293	3
3006	12294	6
3007	12295	8
3008	12296	13
3009	12297	14
300A	12298	58 SUM

SUBMITTED BY:
BHAGYA VINOD RANA
[U19CS012]