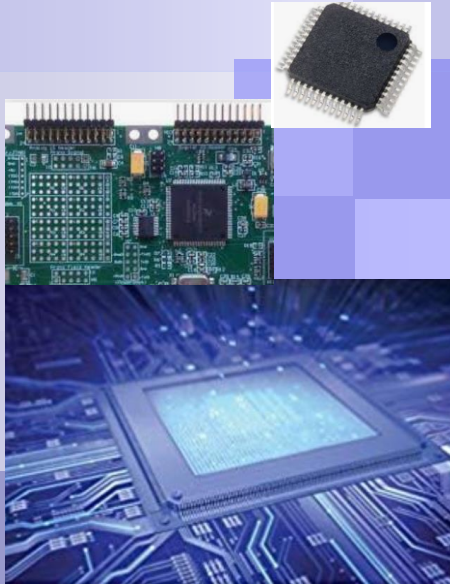


CE320 : Microcomputers I



Lab4: Code conversion

Jung Me Park, Ph.D

ECE Department, Kettering University

jpark@kettering.edu



Binary Coded Decimal (BCD)

- Although all computers work internally with binary numbers,
 - the input and output equipment generally uses decimal numbers.
 - the decimal numbers must be coded in terms of binary signals.

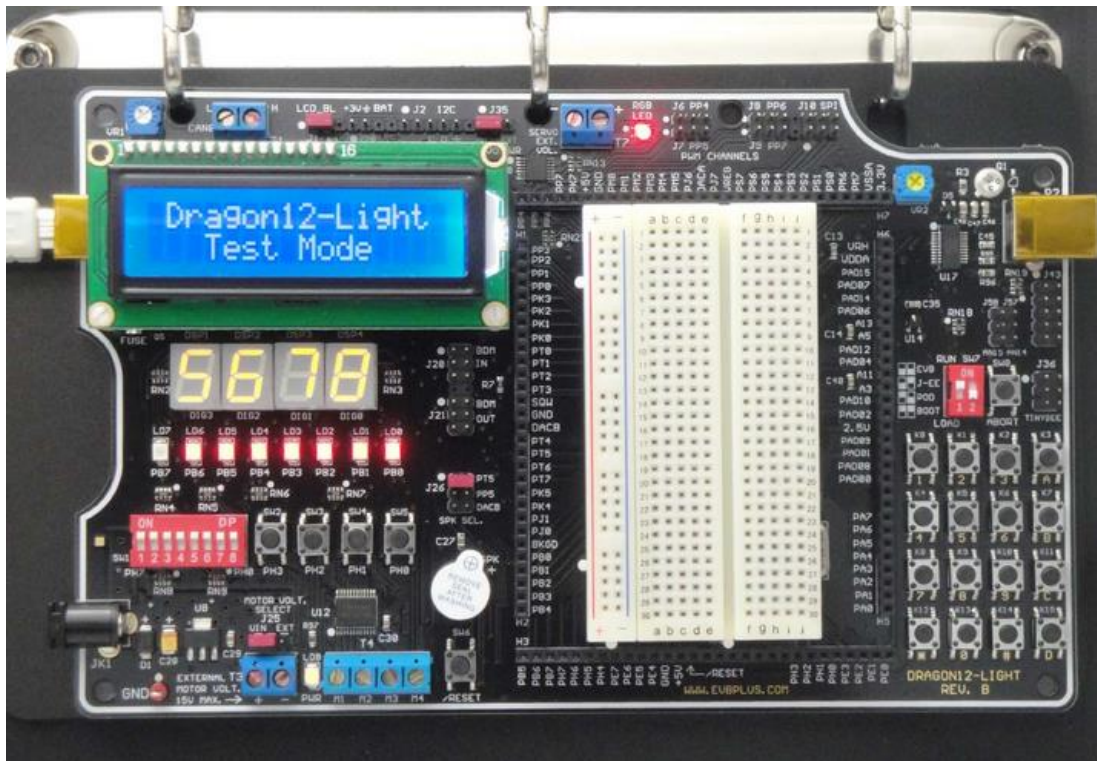


- A binary coded decimal (BCD):
 - In the simplest form of binary code, each decimal digit is represented by its binary equivalent.
 - For example, **5,678** is represented by

5	6	7	8
0101	0110	0111	1000
 - This representation is called **a Binary Coded Decimal (BCD)**

Binary Coded Decimal (BCD)

- ❑ Binary Coded Decimal (BCD) is a very convenient way of representing numbers that will be sent to external 7-segment displays.





Lab4: Data Allocation in Lab4

HexData

\$1010

\$98

\$1011

\$2D

\$1012

\$1013

\$1014

...

}
HEX
input
data

ORG \$1010

HexData:

DC.W

\$982D

; 2-byte input data

BCDNum

\$1020

\$1021

\$1022

\$1023

\$1024

}
BCD
output

ORG \$1020

BCDNum:

DS.B

5

; reserve 5-byte for the BCD output



Binary to BCD Conversion

- A binary number can be converted to BCD format **using repeated division by 10.**

- The number is divided by 10, the remainder is recorded
- The quotient is then divided by 10, the remainder is recorded
- **The process is repeated until the quotient is zero**

□ Ex:

	Quotient	Remainder	
12345 / 10 =	1234	5	;Least significant digit
1234 / 10	123	4	
123 / 10	12	3	
12 / 10	1	2	
1 / 10	0	1	;Most significant digit



Division Instructions

Assembly Instruction	Meaning	Operation :Important!
IDIV	Integer DIV ision 16 bit by 16 bit unsigned Integer Division	$[D] \div [X] \Rightarrow \begin{matrix} X=\text{Quotient} \\ D=\text{Remainder} \end{matrix}$
IDIVS	Integer DIV ision 16 bit by 16 bit Signed integer Division	$[D] \div [X] \Rightarrow \begin{matrix} X=\text{Quotient} \\ D=\text{Remainder} \end{matrix}$
EDIV	Extended DIV ision unsigned 32 bit by 16 bit division	$[Y:D] \div [X] \Rightarrow \begin{matrix} Y=\text{Quotient} \\ D=\text{Remainder} \end{matrix}$
EDIVS	Extended DIV ision Signed 32 bit by 16 bit division	$[Y:D] \div [X] \Rightarrow \begin{matrix} Y=\text{Quotient} \\ D=\text{Remainder} \end{matrix}$

Binary to BCD Conversion

❑ Pseudocode: How to convert Binary to BCD

1. Load the address of BCD for **the least significant digit**.

LDY #BCDNum +4

2. Load the 16 bit number into register D

3. Compare register Y with **#BCDNum**

CPY #BCDNum

4. If register **Y** < **#BCDNum**, then exit the loop

BLO ExitLoop ; ExitLoop is a label

5. Load #10 into register **X** ; register X is a divider

6. **IDIV** ; register **D** / register **X**

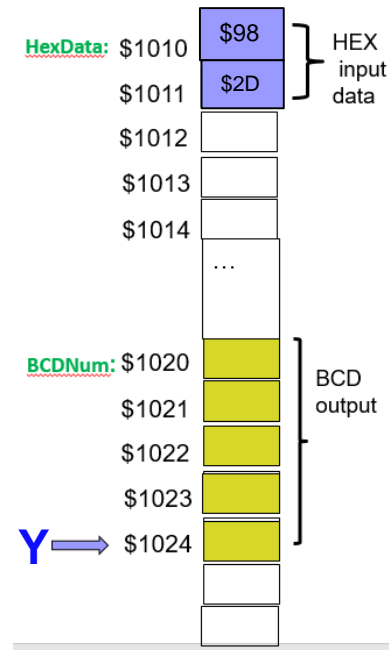
i) Quotient is saved in register X, ii) Remainder is saved in register D (B register inside D)

7. Save the remainder (register B) at the memory location in **Y**

8. **XGDX** :Exchange D with X ; → so D can hold the Quotient for the next division

9. Decrement register Y by 1

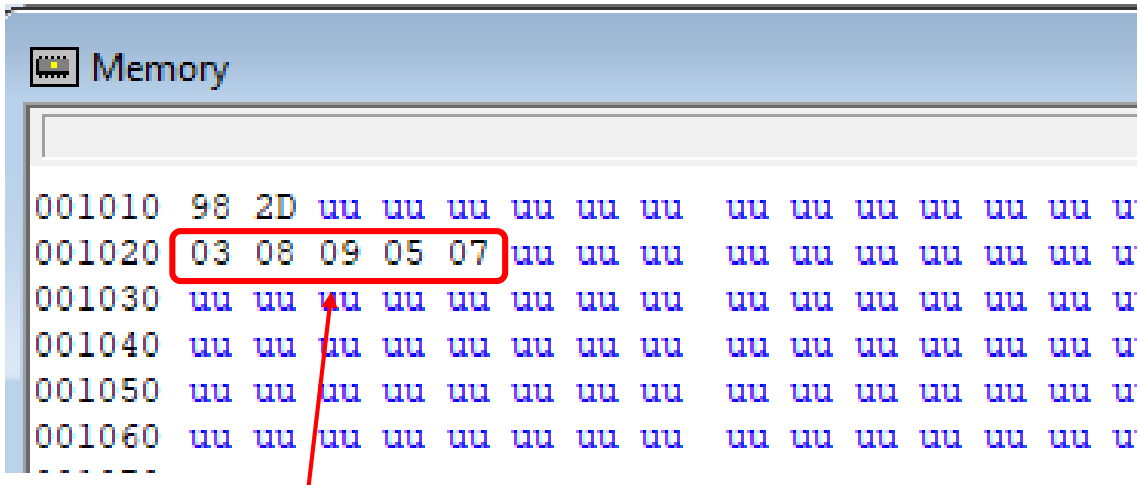
10. BRA Step3.





BCD Output

- The BCD output will be displayed at the memory location, \$1020:1024. Take the screen shots of the BCD output in the memory pane, Assembly pane and Register pane.



The screenshot shows a memory pane with the title 'Memory'. The address 001020 is highlighted, and the value 03 08 09 05 07 is displayed in a red box. A red arrow points from the text 'BCD result at \$1020:1024' to the red box. The memory pane shows the following data:

Address	Value
001010	98 2D uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu
001020	03 08 09 05 07 uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu
001030	uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu
001040	uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu
001050	uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu
001060	uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu uu

BCD result at \$1020:1024



ASCII CODE CONVERSION

Binary to BCD Conversion: Example Code(I)

ASCII code =
\$BCD digit + \$30

HexData: \$1010

\$FE

\$1011

\$FE

\$1012

\$1013

\$1014

...

AsciiNum : \$1030

\$36

\$1031

\$35

\$1032

\$32

\$1033

\$37

\$1034

\$38

\$1035

\$00

}
HEX
input
data

}
ASCII output

Binary to Ascii Conversion

❑ Pseudocode: How to convert Binary to BCD

1. Load the address of Ascii for **the least significant digit**.

LDY # AsciiNum +4

2. Load the 16 bit number into register D

3. Compare register Y with **# AsciiNum**

CPY # AsciiNum

4. If register **Y** < **#AsciiNum**, then exit the loop

BLO ExitLoop ; ExitLoop is a label where the brach is taken

5. Load **#10** into register **X** ; register X is a divider

6. **IDIV** ; register **D / register X**

i) Quotient is saved in register X, ii) Remainder is saved in register D (B register inside D)

7. **Add \$30 to register B**

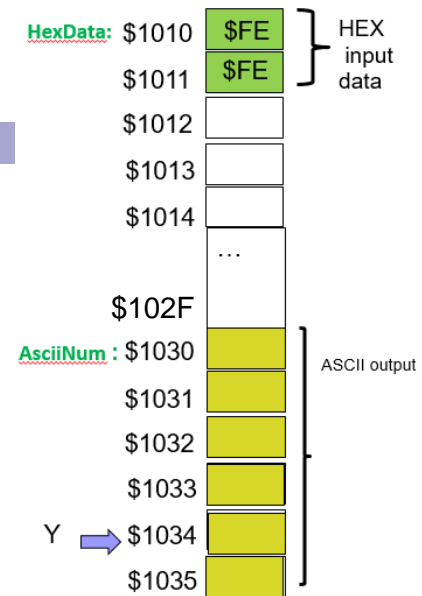
8. Save register B at the memory location in register **Y**

9. **XGDX** ;Exchange register D with register X, so D can hold the Quotient
; for the next division

7. Decrement register Y register by 1

11. BRA Step3.

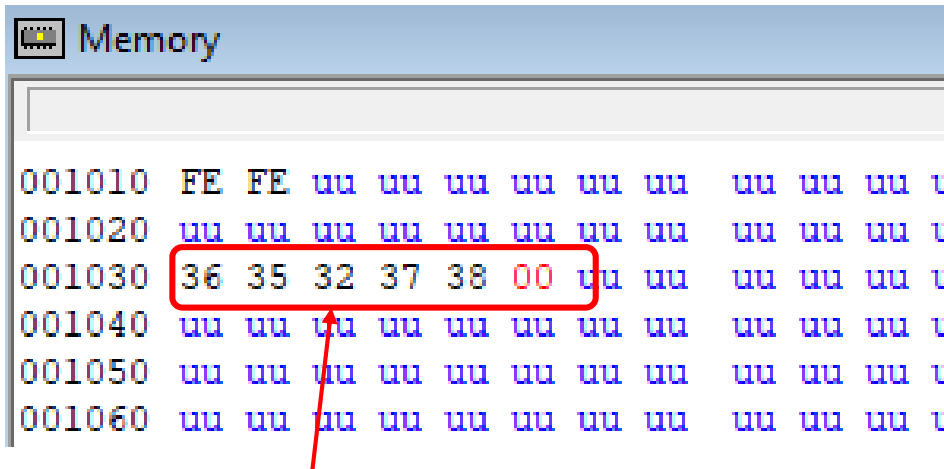
12. Move a byte \$00 at the memory location **AsciiNum +5 (=\$1035)**





ASCII Code Output

- ❑ The BCD output will be displayed at the memory location, \$1020:1024. Take the screen shots of the BCD output in the memory pane, Assembly pane and Register pane.



```
Memory
001010 FE FE uu uu uu uu uu uu uu uu uu uu t
001020 uu uu uu uu uu uu uu uu uu uu uu uu t
001030 36 35 32 37 38 00 uu uu uu uu uu t
001040 uu uu uu uu uu uu uu uu uu uu uu uu t
001050 uu uu uu uu uu uu uu uu uu uu uu uu t
001060 uu uu uu uu uu uu uu uu uu uu uu uu t
```

ASCII code result at \$1030:1034 and ending with a special character, **\$00** at \$1035