Lecture 12 :
Boolean Logic Instructions

Today’s Goals

- Learn how to use Boolean instructions in assembly code

Logical Instructions

- One of the main purposes of the logical instructions is to affect individual bits of a byte without affecting the others.
- Target and Mask byte
  - Target byte with the data
  - Mask byte which determines which bits are affected.
- Format
  - [logical instruction][register or memory] [mask byte]
  - Ex. ANDA #%00001111

Function	0 Mask Bit	1 Mask Bit
AND	Clear to 0	No affect
OR	No affect	Set to 1
XOR	No affect	Toggle

AND

ANDA and ANDB

- ANDA and ANDB
  - affect N and Z
  - clear V
  - no affect on C

- Example
  - Determine if the number in location $1000 is evenly divisible by 8.

```
LDAA    $1000
ANDA    #%00000111 ; or #$07
; If the Branch is taken, the number is divisible
BEQ     xxx
```

OR, XOR, and NOT

- ORAA, ORAB
  - affect N and Z
  - clear V
  - no affect on C
- EORA, EORB (meaning XOR)
  - affect N and Z
  - clear V
  - no affect on C
- COMA, COMB, COM (meaning NOT)
  - All eight bits are complemented.
  - A mask byte is not used. (right?)
  - affect N, Z
  - clear V
  - set C to 1

Example

- Consider a two-door sports car with a trunk and a glove box.
  - Assume that contact switches are used to
    - monitor each door and
    - send signals to the processor indicating
      - whether the door is open (TRUE) or closed (FALSE)
    - Four bits are need to monitor two side doors, a trunk, and a glove box.
    - The four bits will be 7, 6, 5, and 4 of memory $0000.
  - Microprocessor can read the contents of this location at any time to read the status of the doors.
  - Also the microprocessor maintains a bit for the cabin light, the trunk light, and the glove box light.
    - Storing a 0 in the bit causes the light to be OFF
    - Storing a 1 makes the light ON.
    - These four bits will be 2, 1, and 0 of the location $1000 respectively.

	7	6	5	4	3	2	1	0
$0000	GBOXD	LEFTD	RGHTD	TRNKD	-	GBOXL	CBNL	TRNKL

Example

Turn off the glove box light without affecting the other bits.

- Turn OFF → Use AND with a proper mask byte

```
LDAA   $00
ANDA   #%11111011
STAA   $00
```

Example

Turn on the trunk light without affecting the other bits.

- Turn ON → Use OR with a proper mask byte

```
LDAA   $00
ORA    #%00000001
STAA   $00
```


Example

Turn on <u>the glove box light</u> and <u>the cabin light</u> without affecting the other bits.

- Turn ON → Use OR with a proper mask byte

```
LDAA   $00
ORA    #%00000101
STAA   $00
```

Example

Toggle the cabin light without affecting the other bits.

- Toggle → Use XOR with a proper mask byte

```
LDAA   $00
EORA   #%00000010
STAA   $00
```

Example

Negate accumulator D

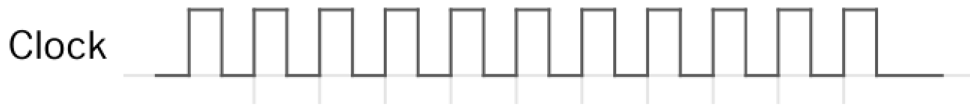
- Negate accumulator D

```
COMA
COMB
ADD     #1
```

- Negate D without using the logical complement functions

```
EORA   #%11111111    ; #$FF
EORB   #%11111111    ; #$FF
ADDD   #1
```

Example



<u>Toggle</u> the cabin lights at exactly 1000 Hz

```
flip:   LDAA    $00     ; 3
        EORA    #CBNL   ; 2
        STAA    $00     ; 3
        LDX     #N      ; 2
loop:   DEX             ; N
        BNE     loop    ; 3(N-1)+1
        BRA     flip    ; 3
```

- 1KHz → 1000 times / sec

- Clock speed of Dragon12+:
  - 24 MHz (24,000,000 Hz) means 24 million clock cycles / sec

- When the sum of all cycles of the lines become 24,000, we can say the module runs 1,000 times per second.

- 3 + 2 + 3 + 2 + N + 3(N–1) + 1 +3 = 24,000
  - 11 + 4N = 24,000 then, 4N = 23989. Therefore, N = 5997.25
  - N should be an integer, so 4N + 11 + ? = 24,000
  - If 5 is used for ?, then N = 5996

Example – continued

<u>Toggle</u> the cabin lights at exactly 1,000 Hz

```
flip:   LDAA    $00     ; 3
        EORA    #CBNL   ; 2
        NOP             ; 1
        NOP             ; 1 (to add 5 extra clock cycles)
        BRA     0       ; 3 (use 3 clock cycles while do nothing)
        STAA    $00     ; 3
        LDX     #5996   ; 2
loop:   DEX             ; 5996
        BNE     loop    ; 3(5996-1)+1
        BRA     flip    ; 3
```

A Short Story about K and M in bytes

- In general,
  - K means 1,000
  - M means 1,000,000
- When you count bytes,
  - K means 1,024
  - M means 1,024 x 1,024
- 1,024 comes from
  - $2^{10} = 1,024$
  - Remember 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, …

Questions?

Wrap-up

What we've learned

- Boolean logical instructions
- ANDx, ORAx, EORx, and COMx

What to Come

- Bit instructions
- Stack
- Subroutines