2019-10-29, Microprocessor Midterm Exam

* Please solve the following problems in the answer sheet. (답안은 답안지에 작성해 주시기 바랍니다.)
* You must show your work unless the answer is obvious. (풀이 과정이 없는 답은 인정하지 않습니다.)

Student ID	Name	Signature

----------Lecture Problems----------

1. (10) (a) (5) Is "data transfer using a 64 bit-wide data bus" always faster than "data transfer using a 1 bit-wide data bus"? Please explain the reason for your answer.
   (b) (5) Explain the meaning of "Universal gates", and prove that 2-input NOR gates are universal gates.
2. (a) (5) Explain why HEX file format is used, and how to use the checksum data inside the HEX files.
   (b) (5) When we upgrade a PC (personal computer) from a 32-bit system to a 64-bit system, which bus is the most critical to improve performance among data, address, and control buses? Explain the reason. (This is not just for 8051 systems, but for typical PCs.)
3. (10) (a) (5) Which of the following is (are) illegal?
   (1) MOV R1, #300 (2) MOV R2, #70 (3) MOV A, #120H (4) MOV A, #FFH (5) MOV R17, #50H
   (b) (5) When register bank 3 (from 0, 1, 2, 3) is used and SP is set to 07H, what is the maximum number of consecutive PUSH instructions after reset, without bank conflicts?
4. (10) (a) (5) When both DB (defined byte) and EQU (equate) directives can be used for a constant value, which one is better to use? Explain the reason. (Hint: memory)
   (b) (5) How much does PC (program counter) increase when executing an instruction?
   (Is this a fixed amount or not?)
5. (10) Show the contents of R0~R7 and SP after finishing the program execution. (default setting)

1 (instruction #)	ORG 0	5	MOV R5, #4H	9	POP 3
2	MOV R0, #1H	6	PUSH 0	10	PUSH 4
3	MOV R1, #2H	7	PUSH 1	11	POP 6
4	MOV R4, #3H	8	POP 2	12	PUSH 5

6. (10) (a) (5) Find the value of R0 and R1 after execution of the following program.
   (b) (5) Rewrite this program using only two DJNZ instructions.
   (Hint: "DJNZ R3, LOOP2 and DJNZ R4, LOOP3" can be changed to one DJNZ instruction)

```
LOC   OBJ            LINE    SOURCE
0000  7900              1    MOV R1, #0H
0002  7800              2    MOV R0, #0H
0004  7400              3    MOV A, #0H
0006  7C02              4    MOV R4, #2H
0008  7B02              5    LOOP3: MOV R3, #2H
000A  7A02              6    LOOP2: MOV R2, #2H
000C  F4                7    LOOP1: CPL A
000D  09                8    INC R1
000E  70(7.(a))         9    JNZ TARGET
0010  08               10    INC R0
0011  DA(7.(b))        11    TARGET: DJNZ R2, LOOP1
0013  DB(7.(c))        12    DJNZ R3, LOOP2
0015  DC(7.(d))        13    DJNZ R4, LOOP3
```

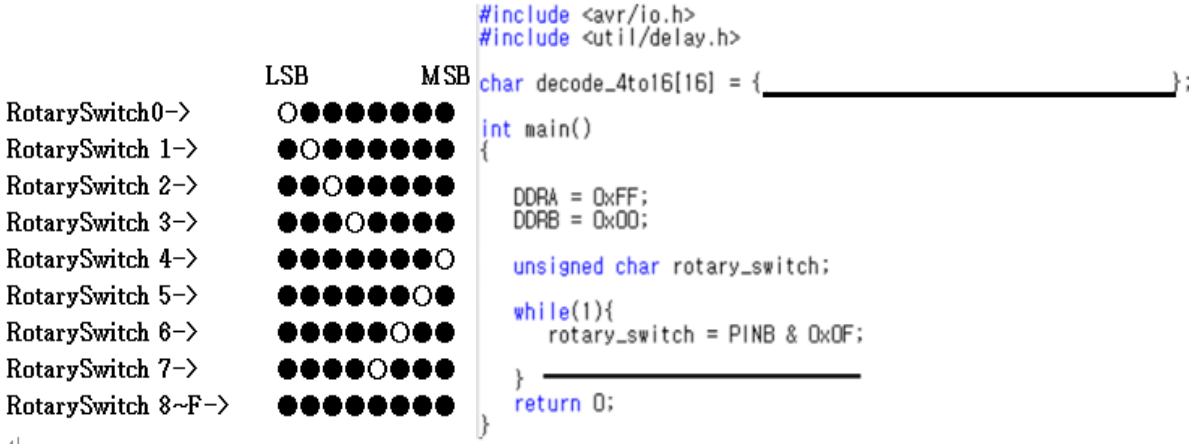
7. (10) For the program of Problem 6, what are the OBJs for four branch instructions (Line 9, 11, 12, 13)? (Hint: 70(7.(a)), DA(7.(b)), DB(7.(c)), DC(7.(d)))
8. (10) (a) (5) Show the Stack storage status and the value of SP after executing “TEST1: LCALL TEST2”.
   (b) (5) Please show the execution order of the four instructions in line (3, 6, 7, 8).

```
LOC  OBJ        LINE    SOURCE
0000            1           ORG 0H
0000 7400       2           MOV A, #0H
0002 120300     3           LCALL TEST1
0005 7403       4           MOV A, #3H
0300            5           ORG 300H
0300 120304     6     TEST1: LCALL TEST2
0303 22         7           RET
0304 22         8     TEST2: RET
```

9. (10) (a) (5) How many push operations are performed when executing an ACALL instruction?
   (b) (5) For the program of Problem 8, can we add a “PUSH 2, PUSH 3, POP 2” sequence between line 6 and 7 without errors? Please explain the reason about your answer.

----------Practice Problems----------

10. (a) (5) In Atmega128, explain about a Data Direction Register (DDR), a PORT register, and a PIN register, including explanation of the functionality when each bit is set to 0 and 1.
    (b) (5) Fill the blanks of a program that decodes a Rotary Switch value into PORT A.



PORTA -> LED								
MCU	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
LED	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0
PORTB -> Rotary Switch								
MCU	-	-	-	-	PB3	PB2	PB1	PB0
Rotary	-	-	-	-	RS3	RS2	RS1	RS0

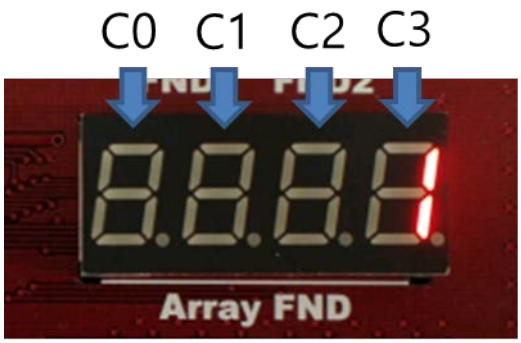
11. (10) (a) (5) Explain the difference between a Static FND and a Dynamic FND (an array of 4 Static FNDs (7-segments)). In addition to this, explain how to display "1234" in a Dynamic FND when using 2 8-bit GPIO ports in our practice board environment.
    (b) (5) The following code is for displaying 1 in the array FND. Fill the following table to complete the following code. (pin a~p: active High, pin C0~C3: active Low)

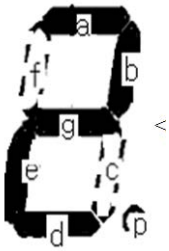
```c
#include<avr/io.h>

int main(void)
{
    DDRB=0xFF;
    DDRG=0xFF;

    PORTB=0x__;
    PORTG=0x__;

    return 0;
}
```





FND	p	g	f	e	d	c	b	a	C3	C2	C1	C0
MCU	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0	PG3	PG2	PG1	PG0
Bit												

Arithmetic operations	
ADD	A,Rn
ADD	A,direct
ADD	A,@RI
ADD	A,#data
ADDC	A,Rn
ADDC	A,direct
ADDC	A,@RI
ADDC	A,#data
SUBB	A,Rn
SUBB	A,direct
SUBB	A,@RI
SUBB	A,#data
INC	A
INC	Rn
INC	direct
INC	@RI
DEC	A
DEC	Rn
DEC	direct
DEC	@RI
INC	DPTR
MUL	AB
DIV	AB
DA	A
CLR	A
CPL	A
RL	A
RLC	A
RR	A
RRC	A
SWAP	A

Program and machine c	
ACALL	addr11
LCALL	addr16
RET	
RETI	
AJMP	addr11
LJMP	addr16
SJMP	rel
JMP	@A+DPTR
JZ	rel
JNZ	rel
JC	rel
JNC	rel
JB	bit,rel
JNB	bit,rel
JBC	bit,rel
CJNE	A,direct,rel

jne A,#data,@
(jump if A != data)

je A, #data,@
(jump if A == data)

ja, jnbe A,#data,@
(jump if A > data)

jae, jnb A,#data,@
(jump if A >= data)

jb, jnae A,#data,@
(jump if A < data)

jbe, jna A,#data,@
(jump if A <= data)

switch A <,==,> #data
(no A modification)

Data transfer	
MOV	A,Rn
MOV	A,direct")
MOV	A,@RI
MOV	A,#data
MOV	Rn,A
MOV	Rn,direct
MOV	Rn,#data
MOV	direct,A
MOV	direct,Rn
MOV	direct,direct
MOV	direct,@RI
MOV	direct,#data
MOV	@RI,A
MOV	@RI,direct
MOV	@RI,#data
MOV	DPTR,#data16
MOVC	A,@A+DPTR
MOVC	A,@A+PC
MOVX	A,@RI
MOVX	A,@DPTR
MOVX	@RI,A
MOVX	@DPTR,A
PUSH	direct
POP	direct
XCH	A,Rn
XCH	A,direct
XCH	A,@RI
XCHD	A,@RI
CJNE	A,#data,rel
CJNE	Rn,#data,rel
CJNE	@Rn,#data,rel
DJNZ	Rn,rel
DJNZ	direct,rel
NOP	

Logic operations	
ANL	A,Rn
ANL	A,direct
ANL	A,@RI
ANL	A,#data
ANL	direct,A
ANL	direct,#data
ORL	A,Rn
ORL	A,direct
ORL	A,@RI
ORL	A,#data
ORL	direct,A
ORL	direct,#data
XRL	A,Rn
XRL	A,direct
XRL	A,@RI
XRL	A,#data
XRL	direct,A
XRL	direct,#data

Boolean variable manip	
CLR	C
CLR	bit
SETB	C
SETB	bit
CPL	C
CPL	bit
ANL	C,bit
ANL	C,/bit
ORL	C,bit
ORL	C,/bit
MOV	C,bit
MOV	bit,C