**A**

activity 1-3 : mainly use SBIS,SBIC and SBI,CBI to move data from one I/O register to another, also you could use ASR if the bit is /2

Q 3.2 R16 = $48 because it use carry flag in the register to rotate

Q 3.3 R16 = $29

Q3.4 It works perfectly fine without SWAP( swap nibble of register) , if i have to use SWAP maybe we could SWAP then Shift right but I don’t think it necessary

Q3.5 SWAP works on general purpose register R0-R31

Q3.6 R2 = 0xFF because 0000 0000 XOR 1111 1111 = 1111 1111

Q3.7 R10 = 0x55 = 0101 0101 because 1010 1010 XOR 1111 1111 = 0101 0101

**B**

**activity 1**

copy from program memory $200 -> when access indirect using LPM($400), I use .db to define string at $200 however the data after .db need to be even byte or else you will need to padding 0 to make both page of the address (SRAM) full.

technique to check the end string we need to check for the 0x00 which is NULL that we pad in the .db I use the technique that if I load data to save in the R16 -> R16+0 if the zero flag set then it is end of the string(use BRNE after to continue the copying loop)

**activity 2**

extra technique I use is instead of MOV 2 times to move addr from Z->X I use MOVW XL,ZL instead and it automatically move the entire 16bits

**activity 3**

2. different between LPM and LD

LPM : load from program memory(flash ROM)

LD : load from data space(SRAM+register)

3. a. LDS R20, 60

b. LD R30, Z

c. LD R25, Z+

d. LPM R25, Z+4

d. is invalid it only support Z or Z+ not Z+4

4. Explain the difference between the following two instructions:

a. LDS R20, $40

b. LDI R20, $40

LDS: load direct from dataspace -> copy content in SRAM : address $40 to r20

LDI : load immediate -> set R20 to $40

**C**

1. show 7 7 7 7 on the 7-segment because PORTD has 0b00000111
2. change 0b00000111 to 0b01100111
3. activity 3 and 4 are the same thing I, write the LED 7-SEGMENT driver -> there are 2 methods using 1 PORT(blinking 1 4bits 7 segment) and using 2 PORTS (using 2 1 bits 7-segment)