3) 42 | 16 = 2.2 10
2 | 16   2

$(42)_{10}$

0100 0010
2

0,100
1 1 0 0
x 0 1 1
    60

2,625

5) a) 96 + 97

1001 0110 1001 0111

  1001 0110
+ 1001 0111
---
(1) 0010 1001
↑
carry - it's wrong

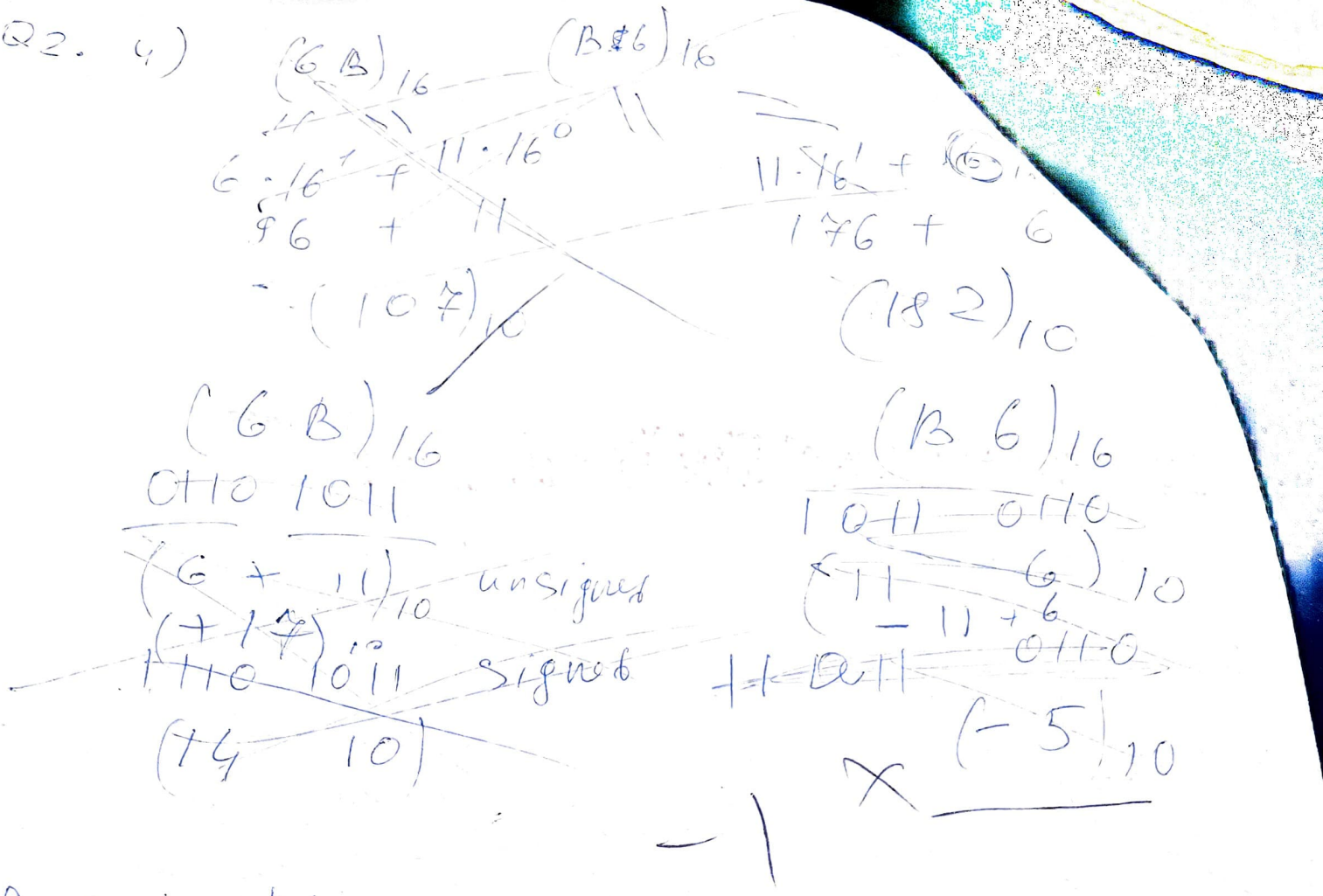
$V = C \oplus P$

$V = 1 \oplus 0 = 1$

~~for unsigned~~
Wrong because
of carry over

~~for signed~~: right
→ 1 (penult mat carry)
V = 1 XOR 1 = 0

Q2. 4)



1. First to binary
   if most left (significant) is 0 positive unsign
   1 signed

2. not group them like i did if we only do to hex but we just to $(6B)_{16} = (0110\ 1011)_2 = (107)_{10}$

$(B6)_{16} = 1011\ 0110 = -128 + 32 + 16 + 4 + 2$

$= (-74)_{10}$

Spring 2022 CSC 220 Midterm Exam
Section code: E001

Instructions: This is a closed-book exam, no calculators or electronic devices can be used. Answers to Q1, Q3, and Q5 can be placed directly on this exam paper, answers to all other questions should be put in the provided blue booklet.

Q1. [20 pts] Short answer, fill-in-the-blank, and multiple-choice questions

1) What are purposes of program counter (PC) and instruction register (IR)? What are stored in these two special registers?

Program counter points to memory address of instruction and it store to instruction register. After execution is complete it removed from Instruction Register.

2) Explain why the PC doesn't always increment instruction number by one?

-1 If it is a big instruction CPU can not execute, so it repeat the cycle and every time PC increment by 1.

multiple instructions of the same time

Size of instructions

3) What model does ARM follow for memory access? How does that model work?

Aside from exclusive and explicitly ordered loads and stores addresses may have arbitrary alignment unless strict alignment checking is enabled. SP is used as the base register the value of the current stack pointer prior to adding the offset must be quadword aligned.

4) Which architecture provides separate buses for program and data memory?
   A. Von Neumann architecture
   (B.) Harvard architecture
   C. None of the above
   D. Both Harvard and von Neumann architectures

5) The instruction "add w0, w1, #5" does what?
   A. Add the value of 5 to the address of w1 and places 5 in that address
   B. Finds the memory location 5 and adds that content to that of w1
   (C.) Add 5 to the value of w1 and places it in w0
   D. None of the above

pair Вызывать first

6) What are the results after the instruction "stp x29, x30, [sp, -48]!" is executed?
   A. Store the content of registers x29 and x30 at sp and sp + 4, respectively; then decrease sp by 48, i.e., sp = sp – 48.
   B. Store the content of registers x29 and x30 at sp and sp + 8, respectively; then decrease sp by 48, i.e., sp = sp – 48.
   C. Decrease sp by 48, i.e., sp = sp – 48, then store the content of registers x29 and x30 at sp and sp + 4, respectively.
   (D.) Decrease sp by 48, i.e., sp = sp – 48, then store the content of registers x29 and x30 at sp and sp + 8, respectively.

book 1:9.2 stp. ldp

7) Which method of signed binary encoding representation of signed integer has two representations of '0'?

A. Sign-magnitude
B. 1's complement
C. 2's complement
D. None of the above

2

complex = big instruction

8) In computer architecture, ISA stands for instruction set archite

and RISC for reduced instruction set computer.

= smaller in size

9) The von Neumann architecture consists of five main components / units, they are Main Memory, ALU CPU, Input, Output, and Hard Drive.

10) The four stages of instruction execution cycle are fetch, decode, execut and store.

Q2. [30 pts] Number systems and arithmetic. You must explain how you get your answe in details, procedural steps are necessary.

1) What are the binary, octal, and hexadecimal representations for the decimal value of 327?

2) What are the decimal representations for the values of 0b100011 and 0x3AF?

3) For the decimal values +42 and -25, what are their two's complement representations in 8-bit hexadecimal forms? = last thing to convert into hex

4) The 8-bit hexadecimal values 0x6B and 0xB6 are stored in two's complement code. What are the equivalent signed decimal numbers?

5) Add the following two pairs of 8-bit numbers, which are already in their 2's complement hexadecimal forms if they are treated as signed integers. Then indicate whether your result is "right" or "wrong" in terms of overflow. First treat them as unsigned values, then as signed values.

a. 0x96 + 0x97

b. 0x62 + 0x45

Q3. [10 pts] Trace this C program and their outputs.

```
int main(void) {
  int x;
  int leftShift, rightShift;
```

```
    printf("Enter an integer in hexadecimal format: ");
    scanf("%x", &x);

    printf("Left shift number of bits: ");
    scanf("%d", &leftShift);
    printf("%x becomes %x\n", x, x << leftShift);

    printf("Right shift number of bits: ");
    scanf("%d", &rightShift);
    printf("%x becomes %x\n", x, x >> rightShift);

    return 0;
}
```

1) Fill in the blanks by the outputs of printf statements for this sample run with provided inputs **7890abcd** for x, **4** for leftShift, and **8** for rightShift. Also explain your answers.

Enter an integer in hexadecimal format: 7890abcd

Left shift number of bits: 4

7890abcd becomes 890abcd0

Right shift number of bits: 8

7890abcd becomes 7890ab

2) Fill in the blanks by the outputs of printf statements for this sample run with provided inputs **98abcdef28** for x, **4** for leftShift, and **8** for rightShift. Also explain your answers.

Enter an integer in hexadecimal format: 98abcdef28

Left shift number of bits: 4

98abcdef28 becomes 8abcdef280

Right shift number of bits: 8

98abcdef28 becomes 8abcdef

Q4. [10] Convert the C code into ARM64/ARMv8 assembly code.

1) For the following C statement, write the corresponding ARMv8 assembly code. Assume that the C integer variables a, b, and c, have already been placed in registers w0, w1, and w2, respectively. Use a minimal number of ARMv8 assembly instructions.

a = (b + c) − 15;

2) Branch and If statement

```
int a = 7, b;
if (a >= 7)
   b = a - 3;
else
   b = a + 3;
```

Q5. [10 pts] Execute ARM64 code segments and get their results.

1) Follow the instructions below CONSECUTIVELY, and after each instruction write the value of destination register in decimal into the space provided.

ORR X0, X31, #4 // X0 = 31
ADD X1, X0, X31 // X1 = X0 + X31
LSL X2, X1, 4 //X2 = shift to the left 4
SUB X3, X2, X1 //X3 = X2 - X1
AND X4, X3, 12 //X4= X3 · 12

2) For the following ARM64 assembly code, assume that registers x0 and x6 contain the values 4 and 0xef70 (an address), respectively. Also, assume that memory contains the values shown in the table below. After execution of these instructions, which memory address's content is changed and what is the value in w2?

Adress	Contents
0xef68	5
0xef6c	10
0xef70	15
0xef74	25
0xef78	35
0xef7c	45

```
ldr w1, [x6, x0]
sub w1, w1, #3
str w1, [x6, #8]
ldr w2, [x6, #8]   // w2 = 45
```

Q6. [10 pts] Miscellaneous questions.

1) We use fixed<w, b> to denote the fixed-point representation of a real number in binary form. w is width of the number representation and b is binary point position within the number. For example, fixed<8, 3> denotes an 8-bit fixed-point number, of which 3 right most bits are fractional.

Given a binary bit pattern 0b00110110, what decimal value does it represent if it's a fixed<8, 3> fixed-point number? What decimal value does it represent if it is a fixed<8, 2> fixed-point number? For the same bit pattern, what is the relationship between the fixed<8, 3> and fixed<8, 2> numbers? [Hint: shifting left one bit, i.e., shifting binary point right one bit, the fixed<8, 3> becomes fixed<8, 2> number.]

2) Given an integer variable and its value as "**int tmp = 0xAABBCCDD;**" it is stored in a little-endian format in the memory at starting address X. Fill in the byte values in the table below.

Mem. Addr.:	X	X+1	X+2	X+3
Byte Value:	DD	CC	BB	AA

Q7. [10 pts] Given the following ARM64 assembly code for a function, add comments to each line, use its stack frame with corresponding variables, and trace the code execution to figure out what it does. Notice that local variables are stored in the function's stack frame. Hint: this is a void function that takes two input parameters.

```
funct:
    sub   sp, sp, #32
    str   x0, [sp, 8]
    str   x1, [sp]
    ldr   x0, [sp, 8]
    ldr   w0, [x0]
    str   w0, [sp, 28]
    ldr   x0, [sp]
    ldr   w1, [x0]
    ldr   x0, [sp, 8]
    str   w1, [x0]
    ldr   x0, [sp]
    ldr   w1, [sp, 28]
    str   w1, [x0]
    nop
    add   sp, sp, 32
    ret
```

Address	Contents
Sp	stack pointer locate 32 memory load to memory &
Sp + 4	store to the register
Sp + 8	stack pointer value to register
Sp + 12	add value to register
Sp + 16	resign ~~[illegible]~~ store
Sp + 20	other place the
Sp + 24	return value
Sp + 28	

Code tracing is missing

& ~~commenting~~

~~each line~~

what the function does -4