CSC111 Assembly Language Programming
Spring 2026 Midterm Exam (B Version)

Student Name: ID: _s

Total Points	

Note: This exam paper should be kept confidential and any dissemination violates copyright.

Q1	Q2	Q3	Q4	Q5	Q6
/10	/10	/20	/20	/20	/20

Q1 (10 points) Multiple-choice questions: enter your answer keys here:

1	2	3	4	5	6	7	8	9	10

For the following multiple-choice questions, each question has exactly one correct answer key. If multiple choices are correct, choose the option "All of the above". **Fill in the answer keys in the table above. (Answer keys written in the question area will not be counted.)**

1. In a 5-bit system, adding unsigned ints 28 and 6 sets which condition?
   A. No flags set
   B. Carry flag set
   C. Overflow flag set
   D. Zero flag set

2. In a 5-bit system, subtracting unsigned ints 3 - 5 results in which carry/borrow status?
   A. Carry=1 (Borrow=0)
   B. Carry=0 (Borrow=1)
   C. Carry=1 (Borrow=1)
   D. Carry=0 (Borrow=0)

3. On ARM Cortex-M3, the borrow and carry flags relation is:
   A. Carry = Borrow
   B. Carry = NOT Borrow
   C. Borrow always 0
   D. Carry always 0

4. In two's complement, TC(x) can be obtained by:
   A. Invert bits
   B. Invert bits and subtract one
   C. Invert bits and add one
   D. Add one then invert bits

5. What is the bit width of each register in ARM Cortex-M processors?
   A) 16 bits

B) 24 bits
C) 32 bits
D) 64 bits

6. Which registers are considered "Low Registers" in ARM Cortex-M and can be accessed by any instruction?
   A) R0-R7
   B) R8-R12
   C) R13-R15
   D) R0-R12

7. In ARM subtraction, what does the carry flag C indicate when a borrow occurs in SUBS?
   A. C = 1 when there is a borrow
   B. C = 0 when there is a borrow
   C. C toggles regardless of borrow
   D. C is always preserved from the previous instruction

8. Which instruction performs reverse subtraction in ARM?
   A. SBC
   B. RSB
   C. SUB
   D. ADC

9. What happens when you execute LDRSB r1, [r0] and the byte at the memory location has value 0xEF?
   A) r1 = 0x000000EF
   B) r1 = 0xFFFFFFEF
   C) r1 = 0xEF000000
   D) r1 = 0x0000FFEF

10. In LDR r1, [r0, r2, LSL #2], what is the effective address used?
    A) r0 + r2
    B) r0 + (r2 × 2)
    C) r0 + (r2 × 4)
    D) r0 + (r2 × 8)

Q2. (10 points) Binary numbers

Assuming a 4-bit system. Show the equivalent decimal values when the data is interpreted as unsigned binary or signed binary.

Binary Value	Signed Decimal Value	Unsigned Decimal Value
1001		
0110		
1100		
0001		
1111		

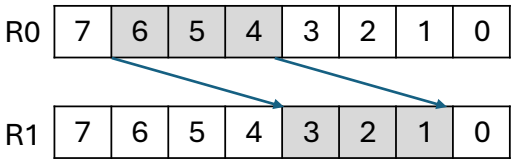
Q3. (20 points) Unsigned and signed arithmetic

Assume a 4-bit system. For each of the following operations, give the equivalent arithmetic calculation in decimal, and the NZCV flags, based on the first row that has been given to you. GT denotes Ground Truth value when there is a carry or overflow.

Operation	Result in binary	Equivalent unsigned arithmetic in decimal	Equivalent signed arithmetic in decimal	NZCV
0011 - 1101	0110	**3 - 13 = 6 (GT=-10)**	**3 - -3 = 6**	**0000**
0011 + 0111				
1011 + 1101				
1011 + 1001				
0011 - 0101				
1011 - 0101				

Q4. (20 points) Bit manipulations

(a) (5 points) Assume an 8-bit system. Copy the top three bits 6..4 in R0 to bits 3..1 in R1, and keep all other bits in R1 unchanged.



(b) (5 points) Assume an 8-bit system. Copy bits 6..3 in R0 to bits 5..2 in R1, and keep R1[7:6] and R1[1:0] unchanged.

(c) (10 points) Assume a 32-bit system. R0 and R1 are initialized with the values below. Write a sequence of assembly instructions that would produce the values in registers R2–R5, from R0 and R1. Do not use MOVW + MOVT). Do not use pseudo-instructions LDR to load large constants into a register.

Given:

R0	0xCAFEBABE
R1	0x0000FFFF

Produce:

R2	0xCAFE4541
R3	0x0000CAFE
R4	0xBABEFFFF

Q5. (20 points) Program execution tracing

Assume a 32-bit system with **big-endian** memory addressing. The following assembly program operates on memory and registers. Fill in the values of registers R1 through R7 and the final memory contents after the entire program has finished executing. Write your answers in hexadecimal format. Assume initially NZCV = 0000.

Assembly program:

```
LDR   R1, =0x20000000
LDR   R2, [R1]
LDR   R3, [R1, #4]!
LSR   R4, R2, #12
MOVW  R5, #7
LSL   R6, R5, #8
BIC   R7, R2, R6
STR   R7, [R1, #8]
```

Initial memory contents:

Addr	Content															
0x20000000	AB	CD	EF	01	23	45	67	89	FE	DC	BA	98	76	54	32	10

Final register values:

R1	R2	R3	R4
R5	R6	R7	

Final memory contents:

Addr	Content															
0x20000000																

Q6. (20 points) Assume a 32-bit system. Consider an array[] of 20 integers (each integer is 4 bytes). Assume register R0 holds the base memory address of array, i.e., array[0]. A compiler associates variables z and y with registers R1 and R2, respectively. Translate this C program into ARM assembly language based on the provided comments. Use R3 as a temporary register. (Note: Assume z is an unsigned integer. You may choose to use either pre-index, or pre-index with update, or post-index addressing.)

C program:

```
uint32_t z = array[3] - y;
array[^4] = z * 16;
array[^5] = z / 4;
array[^6] = z + 20;
array[^7] = z | 0x1F;
```

Assembly program:

_____; load array[3] into temporary register R3

_____; z = array[3] - y

_____; R3 = z * 16

_____; store into array[4]

_____; R3 = z / 4

_____; store into array[5]

_____; R3 = z + 20

_____; store into array[6]

_____; R3 = z | 0x1F

_____; store into array[7]