CSC 230
Summer 2022 (A01: CRN 30207; A02: CRN 30208)
Midterm #1: Thursday, 2 June 2022

Marking Key

Students must check the number of pages in this examination paper before beginning to write, and report any discrepancy immediately.

- **All answers are to be written on this exam paper.**
- The exam is closed book. Other than the AVR reference booklet provided to you, no books or notes are permitted.
- When answering questions, please do not detach any exam pages!
- ***A basic calculator is permitted***. Cellphones must be turned off.
- Partial marks are available for the questions in sections B and C.
- The total marks for this exam is 76.
- There are ten (10) printed pages in this document, including this cover page.
- We strongly recommend you read the entire exam through from beginning to end before starting on your answers.
- **Please have your UVic ID card available for inspection by an exam invigilator.**

Section A (36 marks): For each question in this section, place an X beside all answers that apply. Each question is worth three (3) marks. *Partial marks are not given for incomplete answers*.

Remember: Hexadecimal numbers begin with `0x`; binary numbers begin with `0b`; octal numbers begin with `0`; and all other numbers are decimal. *All numbers are unsigned unless the question specifies otherwise.*

Question 1: The unsigned integer `0x345` can be represented as:

____ `0b010101000011`

____ `0b011100101`

__X__ `0b001101000101`

____ `0575`

____ None of the above.

Question 2: The unsigned integer `0b10100011` is equivalent to:

__X__ `163`

____ `0xA1`

____ `147`

____ `0xA5`

____ None of the above.

Question 3: The unsigned integer `0755` is equivalent to:

____ `0b1011101101`

____ `0x1DF`

____ `0x1CD`

____ `0b011101010101`

__X__ None of the above.

Question 4: The decimal number `-27` represented as a 10-bit two's complement number is equivalent to:

____ `0b1000011011`

____ `0b1111100100`

__X__ `0b1111100101`

____ `0b1000000101`

____ None of the above.

Question 5: The six-bit two's complement number `0b101111` is equivalent to:

____ `0b00101111` as an eight-bit two's complement number.

____ `-31`

____ `-15`

____ `-29`

__X__ None of the above.

Question 6: If we consider I/O port B, then:

____ we set the data-direction for each bit (input or output) by an appropriate value stored in the PINB and PORTB registers.

__X__ we set the data-direction for each bit (input or output) by an appropriate value stored in the DDRB register.

__X__ we read values from the port by reading from the PINB register.

____ we read values form the port by reading from the PORTB register.

____ None of the above.

Question 7: The largest positive value that can be represented using a five-bit two's complement number is:

____ `0b10000`

__X__ `0b01111`

____ `64`

____ `31`

____ None of the above.

Question 8: The most negative value that can be represented using a six-bit two's complement number is:

__X__ `0b100000`

____ `0b111111`

____ `-31`

__X__ `-32`

____ None of the above.

Question 9: The *fetch-decode-execute cycle*:

____ Describes the steps taken when the assembler generates a sequence of `fet`, `dec`, and `exe` instructions that appear in a loop.

__X__ Permits the AVR mega2560 to fetch and decode an instruction in one cycle, and to execute it in the next cycle(s).

____ Permits the AVR mega2560 to fetch, decode, and execute an instruction in one cycle.

____ Is another name for the meaning of the `brne` instruction .

____ None of the above.

Question 10: A *little-endian* computer system:

_____ means that the system always addresses memory as *quadwords*.

_____ stores the more-significant bytes of a 32-bit integer at lower addresses than the less-significant bytes.

_____ is another way of saying that a system uses the von Neumann (i.e., Princeton) machine architecture.

__X__ stores bytes for a 32-bit integer in an order reversed to that of a *big-endian* system.

_____ None of the above.

Question 11: The AVR architecture's *pseudo-registers* X, Y, and Z:

__X__ actually refer to specific pairs of general-purpose registers.

_____ actually refer to memory-addressed I/O ports.

__X__ can be referred to as X, Y and Z in certain machine instructions.

__X__ may be used to hold unsigned 16-bit integers, signed 16-bit integers, or 16-bit memory addresses.

_____ None of the above.

Question 12: The *hex file* produced by the AVR assembler:

__X__ is transferred to a AVR mega2560 board by using `avrdude`.

_____ contains the source code for an AVR assembly-language program.

_____ must be further edited in order to resolve branch addresses.

__X__ is not meant to be easily human-readable.

_____ None of the above.

Section B (20 marks): One question

Question 13: Consider the following program written in AVR assembler:

```
.cseg
.org 0

start:      ldi r16, 0b00001010
            ldi r17, 0b00000101
pointA:     add r16, r17
            dec r18
            dec r17
            cpi r17, 0x02
            brne pointA
stop:       rjmp stop
```

a) How many executions occur for the `dec r17` instruction? *Explain your answer.*

It executes three times, such that the value in `r17` goes from 5 to 4, then from 4 to 3, and from 3 to 2. At this third decrement, the value in `r17` is equal to 2, which means the `cpi` instruction will results in the Z flag being set, and `brne` will not track to `pointA`, such that the program proceeds to the `rjmp` loop.

(8 marks)

b) What is the value in `r16` when the program reaches the instruction `rjmp stop`? *Explain your answer.*

This code fragment repeatedly adds the value of `r17` to `r16`, although `r17` is decremented by 1 each time through the loop:

1. First add: 10 + 5 = 15
2. Second add: 15 + 4 = 19
3. Third add: 19 + 3 = 22

Therefore the final value stored in `r16` is 22 (`0x16`)

(6 marks)

c) How would you modify the end of this program such that the value in `r16` would be stored at the highest address in SRAM? Answer this by providing the AVR assembler lines needed before or after (or both!) the `rjmp stop` line. *Explain your answer.*

There are two ways – one a bit more explicit than the other. Either is fine, but must be explained.

The highest address in `SRAM` is `0x21ff` (for the processor we are using in the course) therefore the following can be added at the start of the program:

```
.dseg
.org 0x21ff
ANSWER: .byte 1
```

and this added just before the `rjmp`

```
sts ANSWER, r16
```

Equivalently, the address could just be hardcoded into the instruction:

```
sts 0x21ff, r16
```

and other variants of this exist (i.e. including the "m2560def.inc" file, and using `RAMEND`), or using a pseudo-register with `HIGH(0x21ff)` and `LOW(0x21ff)` along with the `st` instruction, etc.)

(6 marks)

Section C (20 marks): Two questions

Question 14: 10 marks

Consider the following instruction:

```
andi r21, 125
```

Using the information provided to you in the "AVR Architecture Reference Booklet" that comes with this exam, what is the encoding of this instruction?

Your answer must not only show all work but also explain how you arrived at your answer.

Refer to page 20 of the AVR Instruction Set document for the 16-bit Opcode details. This was provided to you during the test in the eight-page AVR booklet.

Note that the instruction only permits registers 16 to 31 to be used. Although 21 is `0b10101` in binary, only the right-most four bits are used (i.e., `dddd` = `0101`).

The constant 125 is easily converted to an 8-bit binary number `0b01111101`.

Inserting the `dddd` first gives us:

```
0111 KKKK 0101 KKKK
```

and then inserted the `KKKKKKKK` gives us:

```
0111 0111 0101 1101
```

Therefore the encoding is `0x775D`

- Good answer: 10 marks
- Answer is incorrect, but because of a minor mistake: 7 marks
- Answer is incorrect, but because of a significant mistake: 4 or 5 marks
- Failing answer (various): < 4 marks

Question 15: 10 marks

Consider the following encoding of an instruction:

```
0xF7A4
```

Using the information provided to you in the "AVR Architecture Reference Booklet" that comes with this exam, what is the actual AVR instruction (i.e. mnemonic plus argument)?

Your answer must not only show all work but also explain how you arrived at your answer.

The binary of the instruction (separated into groups of four bits) is as follows:

```
1111 0111 1010 0100
```

Examining the 16-bit opcodes provided in the AVR Architecture Reference Booklet provided with the test, there is a single instruction that begins with `1111`: the `BRGE` instruction.

This instruction has the following encoding for the value of k (which is a 7-bit two's complement number, representing values from -64 to +63):

```
1111 01KK KKKK K100
```

Therefore the value of `KK KKKK K` here is `11 1010 0`.

That is, `1110100` is a negative number (which you can immediately tell because of the left-most set bit). Negating all of this in the conversion technique shown in the course produces `0001100` – which is 12. Therefore the value of k is -12.

Putting this all together, the instruction is: `BRGE -12`

- Good answer: 10 marks
- Answer is incorrect, but due to minor mistake: 7 marks
- Answer is incorrect, but due to significant mistake: 4 or 5 marks
- Failing answer (various): < 4 marks

(This page intentionally left blank.)