NANYANG TECHNOLOGICAL UNIVERSITY

SEMESTER 2 EXAMINATION 2016-2017

CE1006/CZ1006 – COMPUTER ORGANIZATION AND ARCHITECTURE

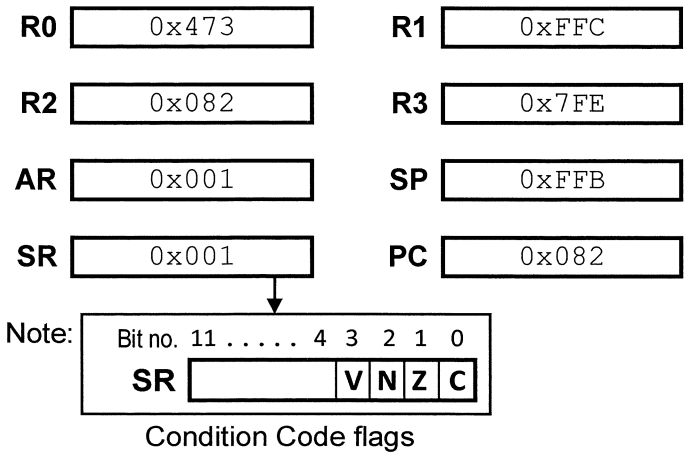
Apr/May 2017 Time Allowed: 2 hours

INSTRUCTIONS

1. This paper contains 4 questions and comprises 7 pages.
2. Answer **ALL** questions.
3. This is a closed-book examination.
4. All questions carry equal marks.
5. The VIP Instruction Set Summary Chart is provided in Appendix 1 on page 7.

---

1. Figure Q1a shows the hexadecimal contents of several registers in the VIP processor and a section of its memory.



MEMORY

Address	Content
:	:
0x080	0x808
0x081	0x432
0x082	0x40D
0x083	0xFFD
0x084	0xBFC
:	:
0xFF9	0x121
0xFFA	0x572
0xFFB	0xFFF
0xFFC	0x000
0xFFD	0x004
0xFFE	0x123

Figure Q1a

Note: Question No. 1 continues on Page 2

(a) Give (*in hexadecimal*) the 12-bit contents in the two registers **R0** and **SR**, immediately after the execution of each instruction given below.

Note: Instructions (i) to (v) are **not consecutive instructions**. You must use the initial conditions shown in Figure Q1a to derive your answer for each of the instructions given below.

(i) `MOV R0,[0x083]`

(ii) `MOV R0,[R1]`

(iii) `OR R0,[R2+0xFFE]`

(iv) `ADDC R0,R3`

(v) `RRC R0`

(10 marks)

(b) A VIP assembly language code segment is given in Figure Q1b. Rewrite this code segment to optimize its execution speed. Your optimized code segment must produce the same result in the memory variable at address **0x100**. Also ensure that the contents in registers **R0** to **R3** prior to the label **START** remain unaltered by your code execution. More marks will be given for a more optimized code.

(8 marks)

```
START   PSHM 15          ; save used registers
        MOV  R1,#0       ; initialize R1 and R2
        MOV  R2,#16
LOOP    ADD  R1,R2       ; cumulate results
        SUB  R2,#1       ; manage count loop
        JNE  LOOP
        ADD  R1,R2       ; add R2 to R1 one last time
        MOV  R3,#0x100   ; update memory var at 0x100
        MOV  [R3+0],R1
        POPM 15          ; restore used registers
```

Figure Q1b

(c) Describe clearly what you expect to observe if code execution proceeds with the initial VIP processor states shown in Figure Q1a. You must describe clearly which registers are expected to change and how they will be changing during execution. Answer this question using the initial conditions shown in Figure Q1a and relevant information in Appendix 1.

(7 marks)

2. A VIP assembly language program and the contents of several memory variables are given in Figure Q2.

(a) Give the instructions needed to achieve the requirements stated in the comments at (**a1**) based on the manner the two parameters are passed at the labels **Pas1** and **Pas2**.

(6 marks)

```
Pas1  PSH   #0x100      ;
Pas2  PSH   #0x101      ;
      CALL  SubZ        ;
        :
SubZ  PSHM  15          ;
        ?               ; get the content in memory variable Var_A into...
        ?               ; ... R0 using the address of Var_A on the stack.   } (a1)
        ?               ; get the address of Var_B on the stack into R1.
Loop  MOV   R2,[R1]     ;
      MOV   R3,#3       ; (b1)
Nxt1  CMP   R2,#0       ; (b2)
      JPL   Nxt2        ; (b3)
      NEG   R2          ; (b4)
      JMP   Nxt3        ; (b5)
Nxt2  ADD   R2,R2       ; (b6)
Nxt3  SUB   R3,#1       ; (b7)
      JNE   Nxt1        ; (b8)
      MOV   [R1],R2     ;
      ADD   R1,#1       ;
      SUB   R0,#1       ;
      JNE   Loop        ;
        ?               ;
        ?               ; } (c1)
```

Address	Contents
Var_A 0x100	0x003
Var_B 0x101	0x111
Var_C 0x102	0xFFF
Var_D 0x103	0x999
Var_E 0x104	0x111

Figure Q2

(b) Give the instructions required at (**c1**) to return from subroutine **SubZ** correctly.

(4 marks)

(c) Based on the memory variable values shown in Figure Q2, give the 12-bit hexadecimal values of **Var_A**, **Var_B**, **Var_C**, **Var_D** and **Var_E** immediately after returning from the subroutine **SubZ**. You may assume that the program execution begins at the label **Pas1**.

(8 marks)

(d) Write the equivalent C high-level language constructs that represents the code segment given by instructions (**b1**) to (**b8**). Assume the registers **R2** and **R3** used in Figure Q2 are represented by the C integer variables **Var2** and **Var3** respectively.

(7 marks)

3. (a) Determine the most suitable memory to be used for each requirement listed in Table Q3 below. The memory types available are: SRAM, DRAM, EEPROM, NAND Flash, NOR Flash and Magnetic HDD. Note that each memory type may be used more than once if deemed most suitable.

(6 marks)

Table Q3

Requirements	Memory Type
Memory to store voice recordings in a Voice Recorder Pen. Recordings needs to be preserved even after the Pen is powered off.	
Memory in Video Recorder used to record TV shows. The recorder is able to store up to 7 days of high quality video recordings.	
System memory of a High Performance Graphics Card in a Gaming PC.	

(b) Explain and illustrate with a diagram of a Floating Gate Transistor, why Flash memory is non-volatile in nature.

(4 marks)

(c) Figure Q3 shows some printed circuit board (PCB) traces between two high speed interfaces. The wavy traces are known as serpentine traces. Explain the purpose of using serpentine traces in PCB designs.

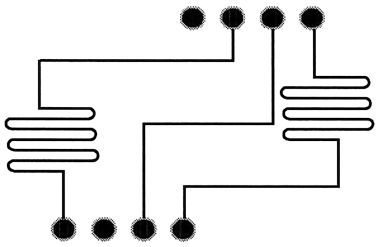


Figure Q3

(5 marks)

Note: Question No. 3 continues on Page 5

(d) Consider a computer system with the following specification.

- UART interface is used between computer and keyboard
- One ASCII character is transmitted from the keyboard to the computer every time a button is pressed
- UART configuration used
  - 1 Start, 8 Data, 1 Stop Bit
  - Even Parity Scheme
  - Baud rate = 9600

Answer the following questions using the information given above,

(i) Can the system support a typist with a typing speed of 200 characters per second (cps)? Justify your answer with detailed calculations.

(4 marks)

(ii) Explain whether the maximum data transfer rate (cps) of this system can be increased without changing the baud rate. Is there any trade off to this new scheme (if any)?

(4 marks)

(iii) What does Even Parity Scheme mean with respect to the UART connection?

(2 marks)

4. (a) Describe two techniques used in solid state drives (SSD) to mitigate the finite erasure cycles of flash memories used to build SSD.

(4 marks)

(b) Consider a computer system with the following characteristics.

- Main memory access time = 100 ns
- Translation Lookaside Buffer (TLB)
  - TLB access time = 5 ns
  - Average TLB hit rate = 90%
- TLB and Page Table access do not overlap

What is the average time needed to retrieve the physical memory address information? Assume no Page Table miss occurs.

(6 marks)

Note: Question No. 4 continues on Page 6

(c) Accumulators are internal memory that store intermediate results of arithmetic computations done in the processor. Typically, an accumulator has a larger bit width compared to usual data registers. For example, if the data registers are n-bit in width, the accumulator may have a width of 2n+1 bit. Explain the reason behind having a '2n' and '+1' bit width design for the accumulator.

(5 marks)

(d) Consider a processor with 4 pipeline stages: Fetch Instruction (F), Decode (D), Execute (E) and Store (S). Assume that

- Branch target address is calculated at the execute stage
- Instruction length for every instruction is one word long
- Each pipeline stage takes 1 cycle to complete
- This processor is not the VIP processor

(i) How many cycles does the code in Figure Q4 take? Assume delay branching is not enabled.

(5 marks)

(ii) Consider the situation where the same code in Figure Q4 is loaded into a non-pipeline processor, i.e. each instruction is fully executed before the next instruction is fetched. Given that each instruction takes 1 cycle to complete, how many cycles will the entire code take?

(2 marks)

(iii) Comment on the results obtained in Q4(d)(i) and Q4(d)(ii) above with respect to the efficiency of pipeline and non-pipeline processors.

(3 marks)

```
      MOV   AR, #5      ; I1
      MOV   R0, #0x800  ; I2
      MOV   R1, #0x300  ; I3
Loop  SUB   [R0],[R1]   ; I4
      INC   R1          ; I5
      JDAR  Loop        ; I6
      ADD   R3, [R0]    ; I7
      MOV   [R1], R3    ; I8
```

Figure Q4

Appendix 1

VIP Instruction Set Summary Chart

VIP Instruction Encoding – Opcode Formats

11	10	9	8	7	6	5	4	3	2	1	0
0-7	Dual operand			d				s			
8	Short Move			d				n			
9-A	Unary/Control			op-code				operand = s, d or n			
B-F	JMP			2's complement -128 to +127 *relative*							

Group 1 – Dual-Operand Instructions (Opcode: 000 to 8FF)

Bits 8-11	Name	Bits 4-7	Bits 0-3	Operation	Flags
0	MOV	d	s	d ← s	NZ
1	AND	d	s	d ← d .AND. s	NZ
2	OR	d	s	d ← d .OR. s	NZ
3	EOR	d	s	d ← d .EOR. s	NZ
4	ADD	d	s	d ← d + s	VNZC
5	ADDC	d	s	d ← d + s + carry	VNZC
6	SUB	d	s	d ← d + (.NOT. s) + 1	VNZC
7	CMP	d	s	d + (.NOT. s) + 1	VNZC
8	MOVS	d	n	d ← n	

Group 2 – Unary and Control Instructions (Opcode: 900 to 9FF)

Bits 4-7	Name	Bit 0-3	Operation	Flags
0	INC	d	d ← d + 1	C
1	DEC	d	d ← d + 0xFFF	NZC
2	ROR	d	Rotate d right : msb ← lsb; and C ← lsb	NZC
3	ROL	d	Rotate d left : lsb ← msb; and C ← msb	NZC
4	RRC	d	Rotate d right including carry	NZC
5	RLC	d	Rotate d left including carry	NZC
6	RAR	d	Rotate d 'arithmetic' right preserving msb	NZC
7	PRSG	d	Left shift lsb from EOR (bits 11,5,3,0)	NZC
8	INV	d	d ← .NOT. d	NZ
9	NEG	d	d ← (.NOT. d) + 1	NZC
A	DADD	s	AR ← AR + s + carry (as 3 BCD digits)	ZC
B	UMUL	s	R1:R0 ← unsigned R0 times unsigned s	Z
C	TST	s	s + 0	NZ
D	EXEC	s	Execute s as an instruction	implied
E	BCSR	n	SR (bits 3-0) ← SR .AND. (.NOT. n)	explicit
F	BSSR	n	SR (bits 3-0) ← SR .OR. n	explicit

Group 3 – Unary and Control Instructions (Opcode: A00 to AFF)

Bits 4-7	Name	Bits 0-3	Operation	Flags
0	PSH	s	SP ← SP-1; (SP) ← s	
1	POP	d	d ← (SP); SP ← SP+1	explicit if d=SR
2	PSHM	3:2:1:0	Push R3:2:1:0 to stack, R3 first	
3	POPM	3:2:1:0	Pop R3:2:1:0 from stack, R3 last	
4	CALL	s	SP ← SP-1; (SP) ← Return Address PC ← Effective address	
5	RET	n	PC ← (SP) + n; SP ← SP+1	
6			See subgroup 3a	
7	RCN	n	Count for next rotate instruction. if n=0 use bits 3:2:1:0 of AR	
8	JDAR	±n	AR ← AR-1, if AR != 0, PC ← PC ± n	
9	JPE	±n	If parity of AR is even, PC ← PC ± n	
A	JPL	±n	If N = 0, PC ← PC ± n	
B	JVC	±n	If V = 0, PC ← PC ± n	
C	JGE	±n	If N = V, PC ← PC ± n	
D	JLT	±n	If N != V, PC ← PC ± n	
E	JGT	±n	If Z = 0 and N = V, PC ← PC ± n	
F	JLE	±n	If Z = 1 or N != V, PC ← PC ± n	

Group 1 – Jump Instructions (8-bit Range) (Opcode: B00 to FFF)

Bits 8-11	Name	n = Bits 0 to 7	Operation
B	JMP = BRA	-128 to +127	PC ← PC ± n
C	JEQ = JZ	-128 to +127	If Z=1, PC ← PC ± n
D	JNE = JNZ	-128 to +127	If Z=0, PC ← PC ± n
E	JHS = JC	-128 to +127	If C=1, PC ← PC ± n
F	JLO = JNC	-128 to +127	If C=0, PC ← PC ± n

Group 3a – Control Instructions (Opcode: A60 to A6F)

Bits 4-7	Name	Bits 0-3	Operation
6	RETI	0	SR ← (SP); SP ← SP+1; PC ← (SP); SP ← SP+1
6	SWI	1	SP ← SP-1; (SP) ← PC; SP ← SP-1; (SP) ← SR; PC ← (0x009)
6	WAIT	2	IE ← 1; Execution resumes after interrupt signal
6	HALT	3	Stop execution. Non-maskable interrupt or hardware reset to exit.
6	STOP	4	Stop execution. Reset to exit.
6	SYNC	8	Pulse SYNC output pin high for 1 clock cycle
6	NOP	9	No operation
6	LOCK	A	Block interrupts and bus sharing
6	UNLK	B	Allow interrupts and bus sharing
6	MSS	C to F	Memory Space Select override

Addressing Modes

Hex	Symbol	Location of Data	Availability
0	R0	Register R0	Both d and s
1	R1	Register R1	Both d and s
2	R2	Register R2	Both d and s
3	R3	Register R3	Both d and s
4	[R0]	Register R0 indirect	Both d and s
5	[R1]	Register R1 indirect	Both d and s
6	[R2+n]	Register R2 with offset indirect	Both d and s
7	[R3+n]	Register R3 with offset indirect	Both d and s
8	AR	Data is in Auxiliary Register	Both d and s
9	SR	Status Register	Both d and s
A	SP	Stack Pointer	Both d and s
B	PC	Program Counter	Both d and s
C	#n	Immediate, (or just n for CALL)	s only
D	[n]	Absolute (code space for CALL)	Both d and s
E	[SP+n]	SP with offset indirect	Both d and s
F	[PC+n]	PC with offset indirect (CALL is relative with PC+n)	Both d and s

Notation: d = destination; s = source

Description of bits in Status Register

SR	F	R	Description
11-8	*	*	Reserved
7-4	*	*	Defined but not described here
3	V	0	Set if 2's complement sign is incorrect
2	N	0	Is most significant bit of result
1	Z	0	1 if result is zero, otherwise 0
0	C	0	1 if carry out, otherwise 0

Notation: SR=Bits in register; F=Name of flag; R=Value after reset

END OF PAPER

CE1006 COMPUTER ORGANISATION AND ARCHITECTURE
CZ1006 COMPUTER ORGANISATION AND ARCHITECTURE

Please read the following instructions carefully:

1. **Please do not turn over the question paper until you are told to do so. Disciplinary action may be taken against you if you do so.**

2. You are not allowed to leave the examination hall unless accompanied by an invigilator. You may raise your hand if you need to communicate with the invigilator.

3. Please write your Matriculation Number on the front of the answer book.

4. Please indicate clearly in the answer book (at the appropriate place) if you are continuing the answer to a question elsewhere in the book.