NANYANG TECHNOLOGICAL UNIVERSITY

SEMESTER 1 EXAMINATION 2015-2016

CE1006/CZ1006 – COMPUTER ORGANIZATION AND ARCHITECTURE

Nov/Dec 2015 Time Allowed: 2 hours

INSTRUCTIONS

1. This paper contains 4 questions and comprises 8 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 8.

---

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

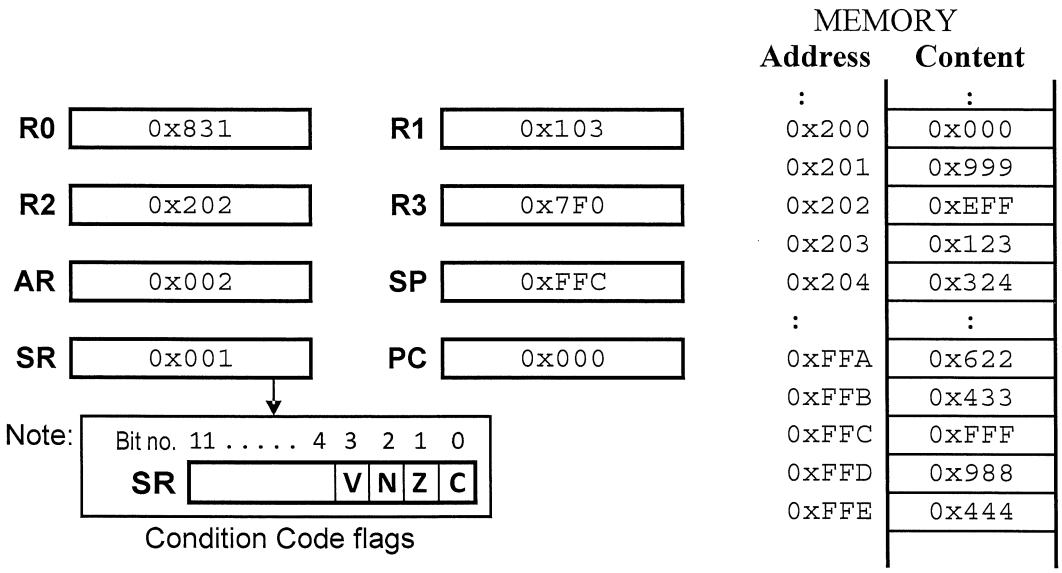

Figure Q1a

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 (vi) 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,[0x201]`

(ii) `MOV R0,[R2+0xFFD]`

(iii) `POP R0`

(iv) `SUB R0,R1`

(v) `RLC R0`

(vi) `EOR R0,R3`

(12 marks)

(b) Write the equivalent C high-level code segment that is represented by VIP assembly code segment shown in Figure Q1b. You may assume that your C integer variables **Var1** to **Var3** are represented by the memory addresses given by labels **Var1** to **Var3** respectively.

(4 marks)

```
Loop    CMP  [Var1],[Var2]
        JLT  Done
        SUB  [Var1],[Var2]
        JMP  Loop
Done    MOV  [Var3],[Var1]
```

Figure Q1b

(c) Using **no more than four** VIP instructions, convert the segment of VIP assembly code in Figure Q1b from a pre-test loop to its post-test loop equivalent. State clearly under what conditions your post-test loop equivalent will fail to work in the same manner as the code segment shown in Figure Q1b.

(6 marks)

(d) Give a single C statement that can replace the C code segment in Q1(b) such that it will always assign the same numeric result into variable **Var3**, given any positive integer values in **Var1** and **Var2**.

(3 marks)

2. (a) An incomplete VIP assembly language program and the contents of several memory variables are given in Figure Q2. Give the mnemonics of (**I1**) to (**I6**) that will complete the missing instructions based on their associated comments.

(8 marks)

```
Main  MOV   SP,#0xFFF      ; Initialize stack pointer
        ?                  ; Push value in memory variable VarX to stack   (I1)
        ?                  ; Push value in memory variable VarY to stack   (I2)
        ?                  ; Push address of memory variable Ans to stack  (I3)
      CALL  SubA           ; Call subroutine SubA to update result in Ans
        ?                  ; Remove parameters on the stack                (I4)
        :
SubA  PSHM  0x006          ; Save away used registers
        :                  ;
        :                  ; to be completed as required in question Q2(b)
        :                  ;
        ?                  ; Restore used registers                        (I5)
        ?                  ; Return to calling program                     (I6)
```

	Address	Contents
VarX	0x200	0x007
VarY	0x201	0x006
Ans	0x202	0x000

Contents in Memory

Figure Q2

(b) The subroutine **SubA** implements the equivalent C language statement given by

```
Ans = (VarX + VarY) * 9;
```

Complete the remaining portion of subroutine **SubA** based on the way the parameters **VarX**, **VarY** and **Ans** have been passed in Figure Q2. Assume that C variables **VarX**, **VarY** and **Ans** are 12-bit integer values in the VIP assembly language implementation. State clearly under what condition(s) the results of your VIP implementation will be incorrect. You will be given more marks for a computationally more optimal implementation.

(11 marks)

(c) Use one or more alternative VIP instructions to implement the equivalent function of **CALL SubA** if you are not allowed to use the **CALL** mnemonic. State clearly the limitation(s) that may arise from your implementation compared to the original **CALL SubA** instruction.

(6 marks)

3. (a) List three differences between Static RAM (SRAM) and Dynamic RAM (DRAM).

(6 marks)

(b) List two differences between NOR and NAND flash.

(4 marks)

(c) Which type of flash memory would you choose as the system memory for a processor system? Give one reason for your choice.

(2 marks)

(d) Figure Q3 shows two processor systems connected via a UART link.

- System B transmits sensor data continuously over the UART to System A. One UART packet is transmitted with each call to the UART transmit routine in CPU-B.
- Each time System A's UART peripheral receives a packet of UART data, it will send an interrupt to CPU-A, which will then proceed to read the data in an interrupt service routine (UART_RX_ ISR).

Table Q3 shows the various timing and UART parameters of the two systems. Instruction cycle time refers to the time needed to execute one instruction.

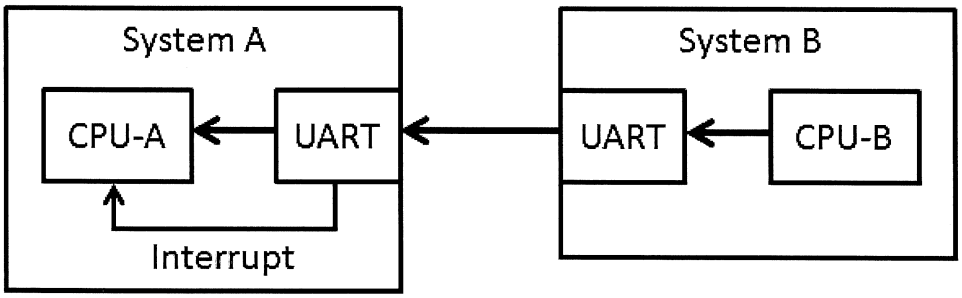

Figure Q3

Note: Question No. 3 continues on Page 5

Table Q3

System A	System B
CPU Interrupt Latency = 10 instruction cycles	CPU Interrupt Latency = 5 instruction cycles
Number of instructions in UART_RX_ISR = 30	Number of instructions in UART transmit routine = 10
Instruction cycle time = 10µs	Instruction cycle time = 20µs
UART Configuration: 1 Start Bit, 7 data bits, Even Parity, 1 STOP bit.	
Allowable UART baud rate: 4800, 9600, 14400, 19200, 28800, 38400, 57600, 115200.	

With reference to Figure Q3 and Table Q3, answer the following questions:

(i) What is the maximum number of UART_RX_ISR that can be serviced by CPU-A in one second?

(2 marks)

(ii) Ignoring the restriction imposed by the allowable UART baud rates shown in Table Q3, what is the maximum number of UART packets that can be transmitted by System B in one second?

(2 marks)

(iii) Given the UART configuration and the allowable UART baud rates stated in Table Q3, find the maximum data transfer rate in which a data file can be transferred from System B to System A. Give your answer in bytes per second. You should state any assumption you have made in deriving your answer.

(5 marks)

(iv) The data transfer rate requirement for this system is 1800 bytes per second. Based on the calculation in part Q3(d)(iii), state whether the current system can meet the requirement. If not, what needs to be done to enable the system to meet the data transfer rate requirement? Note that no hardware or CPU clock rate changes are allowed and software routines are already speed optimized.

(4 marks)

4. (a) Cache design makes use of two locality principles to achieve high cache hit rate.

(i) What are these two principles? Explain the concept behind these two principles.

(4 marks)

(ii) For each of the principles listed in Q4(a)(i), illustrate how software should be designed in order to leverage on the principle to ensure high cache hit rates.

(4 marks)

(b) Consider a system with a virtual address space of 64 Kbyte, a virtual page size of 1 KByte and a physical address space of 4 KByte. With reference to the paging table (partial) shown in Table Q4, what is the virtual address that is mapped to a physical address of 0x0E00? Note that each memory location contains one byte of data and 1 KByte = 1024 Bytes.

(4 marks)

Table Q4

Virtual Page Number	Page Frame Number	Valid Bit
0	2	1
1	-	0
2	3	1
3	1	1
4	-	0
5	0	1
•	-	0
•	-	0

(c) Given two decimal numbers, X = -11 and Y = 11, describe your choice of multiplicand and multiplier if the Booth algorithm is used to compute their product. Assume 5-bit two's complement notation for multiplicand and multiplier. Illustrate clearly how the multiplication is achieved with the Booth algorithm.

(7 marks)

Note: Question No. 4 continues on Page 7

(d) Consider a processor with 4 pipeline stages: Fetch Instruction (F), Decode (D), Execute (E) and Store (S). The processor supports branch delay and the branch target is only known after the Execute Stage. Figure Q4 shows a code segment which performs accumulation of 100 sets of data and all these instructions are one-word instruction. For the instructions within the loop (**I3** to **I6**), answer the following:

(i) Which instructions in Figure Q4 can be inserted in the branch delay slots? Give one reason to support your choices.

(3 marks)

(ii) Which instructions cannot be inserted in the branch delay slots? Give one reason to support your choices.

(3 marks)

```
      MOV   R3, #100    ; I1
      MOV   R0, #200    ; I2
Loop  ADD   R1, [R0]    ; I3
      INC   R0          ; I4
      DEC   R3          ; I5
      CMP   R3, #0      ; I6
      JNZ   Loop        ; I7
```

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.