UNIVERSITY OF VICTORIA
EXAMINATIONS APRIL 2004

C SC 230 Computer Architecture and Assembly Language

NAME (print) _____ REG NO. _____

SIGNATURE _____ DURATION: 3 hours

INSTRUCTOR D. MICHAEL MILLER

TO BE ANSWERED ON THIS EXAMINATION PAPER.

STUDENTS MUST COUNT THE NUMBER OF PAGES IN THIS EXAMINATION PAPER BEFORE BEGINNING TO WRITE, AND REPORT ANY DISCREPANCY IMMEDIATELY TO THE INVIGILATOR.

THIS EXAMINATION HAS **NINE** PAGES PLUS THIS COVER PAGE. THERE IS A **THREE** PAGE HANDOUT.

ATTEMPT EVERY QUESTION. ANSWER IN THE SPACES PROVIDED (YOU DO NOT NECESSARILY HAVE TO USE ALL LINES PROVIDED AND MAY USE OTHER AREAS ON THE **FRONTS** OF THE PAGES IF NECESSARY). USE THE BACKS OF THE PAGES FOR ROUGH WORK.

THIS IS A CLOSED BOOK EXAMINATION. **NO COURSE NOTES, BOOKS OR CALCULATORS, ARE PERMITTED.**

YOU **ARE** PERMITTED TO USE THE INFORMATION SHEETS DISTRIBUTED WITH THE EXAM.

QUESTION	MAX. MARK	STUDENT'S MARK
1	20	
2	5	
3	10	
4	5	
5	5	
6	10	
7	25	
8	20	
TOTAL	100	

1. (20 marks) Indicate whether each of the following statements is true or false by circling the appropriate response. **MARKING:** +1 for a correct answer , -0.5 for an incorrect answer, 0 if neither True of False is circled.

The 8-bit two's complement representation of -13_{10} is 11110010_2.	True	False
Signed magnitude representation has different representations for +0 and –0.	True	False
In two's complement addition, overflow can only occur when adding two numbers with the same sign.	True	False
The stored exponent for the IEEE single precision floating point representations for 37.5 is 130 (in decimal).	True	False
Single bit parity allows for the detection and correction of only single bit errors.	True	False
An arithmetic shift right always preserves the sign of the 2's complement value being shifted.	True	False
On the 6811, the user must enable an interrupt beforee it can be used.	True	False
On the 6811, the external address and data buses are asynchronous.	True	False
A processor need not have a stack pointer register in order to support a jump to subroutine instruction.	True	False
Subroutine parameters can not be used if the program instruction code is in ROM.	True	False
Direct addressing on the 6811 means the address is within the first 256 locations of the address space.	True	False
The data bus to main memory must have at least as many data lines as there are bits in the machine word size.	True	False
On the 6811, on-processor memory can share addresses with off processor memory.	True	False
The 6811 is a CISC design.	True	False
The PENTIUM is a RISC design.	True	False
The PowerPC uses a link register to implement subroutine calls.	True	False
The PowerPC design mandates separate code and data caches.	True	False
Later model PENTIUMs include some SIMD instructions	True	False
Multiprocessor systems guarantee at least linear execution speedup.	True	False
A multiprocessor system must have separate memories for each processor resulting in the problem of cache coherency.	True	False

2. (5 marks) Assuming ACCB is available for use, write the shortest sequence of 6811 instructions you can to multiply the contents of ACCA by 6 without using the MUL instruction. All numbers are to be treated as unsigned integers and you can assume the answer will fit in 8 bits.

3. Answer each of the following in the space provided:

   (a) (3 marks) Explain the fundamental difference between polling and interrupts for recognizing external events and when it is necessary to use interrupts.

   (b) (5 marks) Number the following steps from 1 to 5 in the order they are performed in processing an interrupt on the 6811 using the interrupt jump table technique

   ___ execute the first instruction of the interrupt handling routine
   ___ load the PC with the value from the appropriate interrupt vector
   ___ push the processor registers onto the stack
   ___ recognize the interrupt event and set the event flag
   ___ execute the appropriate jump instruction in the jump table

   (c) (2 marks) What must you do to allow nested interrupts on the 6811, i.e. to allow an interrupt to interrupt while a previous interrupt is being handled?

4. (a) (4 marks) Identify four significant differences usually seen between RISC and CISC designs:

_____

_____

_____

_____

(b) (1 mark) What does it mean to say a processor is *superscalar*?

_____

_____

5. (5 marks) Consider the following segment of a 6811 program

```
0001 d000                          SBASE     EQU   $D000
0002 d000                                    ORG   $D000
0003 d000 48 41 52 52 59 20  SOURCE    FCC   "HARRY POTTER"
          50 4f 54 54 45 52
0004 d00c 00                               FCB   $00
0005 d00d                          DEST    RMB   80
0006 c000                                  ORG   $C000
0007 c000 8e df ff                         LDS   #SBASE
0008 c003 ce d0 0d                         LDX   #DEST
0009 c006 3c                               PSHX
0010 c007 ce d0 00                         LDX   #SOURCE
0011 c00a 3c                               PSHX
0012 c00b bd c0 13                         JSR   STRCPY
0013 c00e 31                               INS
0014 c00f 31                               INS
0015 c010 31                               INS
0016 c011 31                               INS
0017 c012 cf                               STOP
0018
0019 c013 36                       STRCPY  PSHA
0020 c014 37                               PSHB
0021 c015 3c                               PSHX
0022 c016 34                               DES
0023 c017 30                               TSX
```

Show the stack contents after the TSX is executed showing each BYTE on the stack and its location in hexadecimal. ***You may not need all the spaces provided.***

Address (both in hexadecimal)	*Contents* (both in hexadecimal)
_____	____
_____	____
_____	____
_____	____
_____	____
_____	____
_____	____
_____	____
_____	____
_____	____
_____	____
_____	____

6. (10 marks) You are to write a 6811 assembly language function called ISQRT that accepts a one byte unsigned integer as a parameter on the stack. Your function should return with ACCA equal to the smallest integer greater than or equal to the square root of the parameter value. Any registers except ACCA that you use should be protected. ***Include comments to explain what your code is doing.***

   For example, the function call

```
LDAA #9
PSHA
JSR  ISQRT
```

   would return with 3 in ACCA.

   The call

```
LDAA #120
PSHA
JSR  ISQRT
```

   would return with 11 in ACCA.

   The call

```
LDAA #0
PSHA
JSR  ISQRT
```

   would return with 0 in ACCA. ***You may not need all the space provided.***

ISQRT _____

7. (25 marks) You may answer this question using C or 6811 assembly language. If you use C, you may include the "hc11.h" file listed on the handout. ***Assume the processor has a 2 MHZ clock.***

   You are to write a **complete** program that reads a stream of characters each of which is to be coded with even parity. Each character is to be checked for correct parity and if the parity is set correctly, is to forward it to another device (see below). If a parity error is found a signal is set as described below. After checking a character, and either forwarding it or sending an error signal, the program repeats to process the next character. The program repeats this process forever.

   Each character is read a follows:

   - the program sets bit 5 of PORT A to 1 to indicate it is ready for a character
   - the external device providing the characters will then set bit 0 of PORT A to 1 to indicate a character is ready - your program must **poll** for this
   - the program reads the character as an 8 bit number on PORT E – bit 7 is the parity bit
   - after reading the character the program sets bit 5 of PORT A to 0 to signal it received the character
   - shortly after that the device will drop bit 0 of PORT A to 0

   After loading a character from PORT E, the program should check that it is correctly coded for even parity. If it is incorrect, your program is to signal the error by raising bit 6 of PORT A to 1 for **approximately** 1 msec. Except when sending an error signal, bit 6 of PORT A should be 0.

   Each character which has a correct parity bit setting is to be 'forwarded' to another device simply by loading it into PORT B. Characters with incorrect parity setting are not to be forwarded.

   You may not need all the space provided. ***Include comments.***

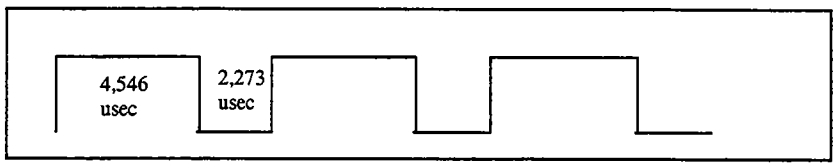
8. (20 marks) You may answer this question using **C or 6811 assembly language**. If you use C, you may include the file hc11.h. ***Assume the processor has a 8MHZ clock***.

   Write a complete C or 6811 assembly language program that behaves as follows:

   (a) After appropriate initialization, a **RISING** edge on input capture pin 1 (IC1), **detected by polling**, causes the program to produce the wave depicted below on bit 6 of Port A.

   (b) A **RISING** edge on IC2, **detected by an interrupt** stops the production of the wave after completing the current cycle, i.e. when the output signal next rises from 0 to 1. Bit 6 of Port A is to be set to 1 when the wave production stops.

   The wave to be produced is a repeating signal for which each cycle consists of high for 4,546 usec. and then low for 2,273 usec. as shown below: Use **OC3 and polling** for the required timing



You may not need all the space provided. ***Include comments.***

***** End of Examination *****

UNIVERSITY OF VICTORIA

EXAMINATIONS APRIL 2004

BIOLOGY 455 (SO1)

TO BE ANSWERED IN BOOKLETS DURATION: 3 hours

TOTAL MARKS: 80 INSTRUCTOR: Dr. G. A. Allen

STUDENTS MUST COUNT THE NUMBER OF PAGES IN THIS EXAMINATION PAPER BEFORE BEGINNING TO WRITE, AND REPORT ANY DISCREPANCY IMMEDIATELY TO THE INVIGILATOR.

THIS QUESTION PAPER HAS 3 PAGES.

I. Answer any **four (4)** of the following questions (5 marks each).

1. The morphological features of organisms are generally the result of diverse and often opposing selection pressures. Choose one of the following examples and describe some of the selection pressures that may be acting to give the trait in question:
   (a) The height of a mature Douglas fir tree (typically over 20 metres); or
   (b) The antlers of a bull elk, which are grown and shed each year.

2. What is gene flow? Describe two ways to measure it.

3. What is phenotypic plasticity? Give an example of a physical trait in humans that exhibits phenotypic plasticity, and give supporting evidence for your example.

4. In 1866, the German biologist Ernst Haeckel made the famous statement "Ontogeny recapitulates phylogeny". What does this statement mean, and how correct does it seem in light of present-day evolutionary knowledge?

5. Mutation rates can vary from one population or species to another, and from one gene region to the next in the same genome, but are typically in the range of 1 in 10^4 to 1 in 10^{10}. What evolutionary processes act on mutation rates, and with what effect? Discuss.

6. What is horizontal gene transfer, how common is it, and what is its evolutionary importance?.

IV Answer any **two (2)** of the following questions (10 marks each).

1. The table below gives a series of 15-base DNA sequences for 5 species and their hypothetical ancestor. (Each nucleotide base position can be viewed as a character.) Draw a cladogram showing the relationships among these taxa that are best supported by these data. On the cladogram you have drawn, indicate <u>three</u> monophyletic groupings, and indicate which characters support each grouping.

Taxon	DNA sequence (base position)														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Ancestor	A	T	G	T	G	T	T	A	A	C	C	A	T	G	A
Species 1	A	C	G	C	G	A	A	T	A	G	C	T	T	T	T
Species 2	A	C	G	C	G	A	A	T	A	G	C	T	T	T	A
Species 3	T	C	T	C	G	T	A	T	C	G	G	A	T	G	A
Species 4	T	C	G	C	G	T	A	T	C	G	G	A	T	G	A
Species 5	A	T	G	C	G	T	T	T	A	C	C	A	G	T	A

2. Compare and contrast the biological species, morphological species, and phylogenetic species concepts, describing the advantages and disaddvantages of each.

3. Many bird species exhibit cooperative behaviour. Two examples are:
(a) mobbing (in which a group of individuals band together to attack and chase away a predator), and
(b) helping at the nest (in which adult offspring of a breeding pair help their parents raise younger siblings, rather than starting their own nest).
Describe the selective pressures that could lead to each of these behaviours.

4. In a number of species (both plants and animals), expression of maleness or femaleness is environmentally determined. Give <u>two</u> examples of species that show environmental sex determination, explaining in each case the pattern of sex expression and the tradeoffs involved.

END

M6811 INSTRUCTION SET

Registers:
A and B are 8-bit accumulators, D is the concatenation of A and B with A the most significant byte. IX and IY are unsigned 16 bit index registers. S is an unsigned 16-bit stack pointer, PC is an unsigned 16-bit program counter. CCR is the condition code register with flags SXHINZVC with C the least significant bit.

Addressing Modes :

Immediate:	IMM	data value is included in the instruction
Direct	DIR	operand is the page 0 address of the data value
Extended	EXT	operand is the 16-bit address of the data value
Indexed	IND	operand is an 8-bit unsigned offset which is added to the value from IX or IY to determine the address of the data value
Inherent	INH	opcode specifies registers
Relative	REL	destination of the branch is specified relative to the address in the PC

Instructions :
The instructions are shown in groups with each group preceded by the allowed addressing modes and the flags affected for the group. The addressing modes are in italics. The flags affected are underlined. none means no flags are affected..

Load: *IMM,DIR,EXT,IND* NZV=0 LDAA, LDAB, LDD, LDS, LDX, LDY

Store: *DIR,EXT,IND* NZV=0 STAA, STAB, STD, STS, STX, STY

Transfer: *INH* NZ TAB, TBA
INH none TSX, TXS, TSY, TYS

Exchange: *INH* none XGDX, XGDY

Stack: *INH* none DES, INS, PSHA, PSHB, PSHX, PSHY, PULA, PULB, PULX, PULY

Arithmetic: *INH* HNZVC ABA
IMM,DIR,EXT,IND HNZVC ADDA, ADDB, ADCA, ADCB
INH none ABX, ABY, INS, DES
INH NZVC SBA *note:* A←(A)-(B)
IMM,DIR,EXT,IND NZVC ADDD, SUBA, SUBB, SUBD, SBCA, SBCB
INH NZV=?C DAA
EXT,IND NZV DEC, INC
INH NZV DECA, DECB, INCA, INCB
INH Z DEX, DEY, INX, INY, *INH* NZVC NEGA, NEGB
EXT,IND NZVC NEG
INH ZVC FDIV *fractional*: (D)/(IX) IX←quotient D←remainder
INH ZV=0C IDIV *integer*: (D)/(IX) IX←quotient D←remainder
INH C MUL *multiplication:* D←(A) * (B)

Logical: *IMM,DIR,EXT,IND* NZV=0 ANDA, ANDB, ORAA, ORAB, EORA, EORB
INH NZV=0C=1 COMA, COMB
EXT,IND NZV=0C=1 COM

Shift & Rotate: *INH* NZVC LSLA, LSLB, LSLD=ASLD, ASLA, ASLB, ASLD=LSLD, ASRA, ASRB, RORA, RORB, ROLA, ROLB
INH N=0ZVC LSRA, LSRB, LSRD
EXT,IND NZVC LSL, ASL,ASR,ROR,ROL, *EXT,IND* N=0ZVC LSR
note: operation of C flag is dictated by the type of shift or rotate.

Decision Making:

No Flag	Carry Flag	Zero Flag	Sign(N) Flag
BRA	BCC, BCS	BEQ, BNE	BMI, BPL
JMP			
Overflow Flag	**2's Complement**	**Unsigned**	**Bit Test**
BVS, BVC	BGE, BGT	BHI, BHS	BRCLR
	BLE, BLT	BLO, BLS	BRSET

B** instructions use *REL*, BRCLR & BRSET use *DIR* or *IND*, JMP uses *EXT* or *IND*
None affect any flags .

Test / Compare:
IMM,DIR,EXT,IND NZV=0 BITA, BITB
INH NZVC CBA *note*: flags set based on (A) - (B)
IMM,DIR,EXT,IND NZVC CMPA, CMPB, CPD, CPX, CPY
INH NZV=0C=0 TSTA, TSTB
EXT,IND NZV=0C=0 TST

Subroutine Linkage:
REL none BSR, *DIR,EXT,IND* none JSR, *INH* none RTS

Interrupt Handling: *INH* I CLI, SEI
INH I=1 SWI
INH none WAI
INH SHINZVC RTI

Setting / Clearing:
DIR,IND NZV=0 BCLR, BSET,
INH Z=1 N=V=C=0 CLRA, CLRB
EXT,IND Z=1 N=V=C=0 CLR

Condition Code
INH SHINZVC TAP
INH none TPA,
INH C CLC, SEC
INH V CLV, SEV

Miscellaneous: *INH* None BRN, NOP, STOP

M6811 Timer Section

Input Captures The 6811 timer section has three input capture pins which can be used to capture an external event. The event can be programmed to be a rising edge, a falling edge or either edge. When an input event occurs the value of the free-running timer TCNT is copied into a time of capture(TIC) register.

Input capture specifics:

Input Capture	Port Location	Int. Vector	Jump Table Location	Time of Input Capture (TIC)
1	Port A, Bit 2	$FFEE-$FFEF	$00E8	$1010-$1011
2	Port A, Bit 1	$FFEC-$FFED	$00E5	$1012-$1013
3	Port A, Bit 0	$FFEA-$FFEB	$00E2	$1014-$1015

The type of event, the input capture flag and the interrupt enable are contained in three registers

Name	Addr.	B7	B6	B5	B4	B3	B2	B1	B0
TCTL2	$1021	0	0	E1B	E1A	E2B	E2A	E3B	E3A
TMSK1	$1022						IC1I	IC2I	IC3I
TFLG1	$1023						IC1F	IC2F	IC3F

ICnI is the interrupt enable flag for input capture n and ICnF is the event flag for input capture n.

The edge sensitivity is programmed as follows:

EnB	EnA		EnB	EnA	
0	0	Capture disabled	1	0	Capture on falling edge
0	1	Capture on rising edge	1	1	Capture on either edge

Output Compares The 6811 has 5 output compares (OC), but you should avoid using OC1 as it is a multi-function pin with several unique features. The program loads TOC register with a value and when TCNT reaches the specified value, the output compare event happens. The particular event is programmable and an interrupt can be generated.

Output compare specifics:

Output Compare	Port Location	Int. Vector	Jump Table Location	Time of Output Compare (TOC)
1	Port A, Bit 7	$FFE8-$FFE9	$00DF	$1016-$1017
2	Port A, Pin 6	$FFE6-$FFE7	$00DC	$1018-$1019
3	Port A, Pin 5	$FFE4-$FFE5	$00D9	$101A-$101B
4	Port A, Pin 4	$FFE2-$FFE3	$00D6	$101C-$101D
5	Port A, Pin 3	$FFE0-$FFE1	$00D3	$101E-$101F

The type of event, the input capture flag and the interrupt enable are contained in three registers

Name	Addr.	B7	B6	B5	B4	B3	B2	B1	B0
TCTL1	$1020	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5
TMSK1	$1022	OC1I	OC2I	OC3I	OC4I	OC5I			
TFLG1	$1023	OC1F	OC2F	OC3F	OC4F	OC5F			

The output event is programmed as follows:

OMn	OLn		OMn	OLn	
0	0	Pin is not affected (OC1 may be)	1	0	Clear OCn to 0
0	1	Toggle OCn	1	1	Set OCn to 1

PORT Addresses: A $1000; B $1004; C $1003; D $1008; E $100A

Other Addresses: DDRC $1007; DDRD $1009; TCNT $100E,$100F

```
/*************************************************/
/*                                               */
/* C SC 230 Course Software                      */
/* 68HC11 standard register section definitions */
/*                                               */
/*************************************************/

#define _IO_BASE  0x1000
#define PORTA     *(unsigned char volatile *)(_IO_BASE + 0x00)
#define PIOC      *(unsigned char volatile *)(_IO_BASE + 0x02)
#define PORTC     *(unsigned char volatile *)(_IO_BASE + 0x03)
#define PORTB     *(unsigned char volatile *)(_IO_BASE + 0x04)
#define PORTCL    *(unsigned char volatile *)(_IO_BASE + 0x05)
#define DDRC      *(unsigned char volatile *)(_IO_BASE + 0x07)
#define PORTD     *(unsigned char volatile *)(_IO_BASE + 0x08)
#define DDRD      *(unsigned char volatile *)(_IO_BASE + 0x09)
#define PORTE     *(unsigned char volatile *)(_IO_BASE + 0x0A)
#define CFORC     *(unsigned char volatile *)(_IO_BASE + 0x0B)
#define OC1M      *(unsigned char volatile *)(_IO_BASE + 0x0C)
#define OC1D      *(unsigned char volatile *)(_IO_BASE + 0x0D)
#define TCNT      *(unsigned short volatile *)(_IO_BASE + 0x0E)
#define TIC1      *(unsigned short volatile *)(_IO_BASE + 0x10)
#define TIC2      *(unsigned short volatile *)(_IO_BASE + 0x12)
#define TIC3      *(unsigned short volatile *)(_IO_BASE + 0x14)
#define TOC1      *(unsigned short volatile *)(_IO_BASE + 0x16)
#define TOC2      *(unsigned short volatile *)(_IO_BASE + 0x18)
#define TOC3      *(unsigned short volatile *)(_IO_BASE + 0x1A)
#define TOC4      *(unsigned short volatile *)(_IO_BASE + 0x1C)
#define TOC5      *(unsigned short volatile *)(_IO_BASE + 0x1E)
#define TCTL1     *(unsigned char volatile *)(_IO_BASE + 0x20)
#define TCTL2     *(unsigned char volatile *)(_IO_BASE + 0x21)
#define TMSK1     *(unsigned char volatile *)(_IO_BASE + 0x22)
#define TFLG1     *(unsigned char volatile *)(_IO_BASE + 0x23)
#define TMSK2     *(unsigned char volatile *)(_IO_BASE + 0x24)
#define TFLG2     *(unsigned char volatile *)(_IO_BASE + 0x25)
#define PACTL     *(unsigned char volatile *)(_IO_BASE + 0x26)
#define PACNT     *(unsigned char volatile *)(_IO_BASE + 0x27)
#define SPCR      *(unsigned char volatile *)(_IO_BASE + 0x28)
#define SPSR      *(unsigned char volatile *)(_IO_BASE + 0x29)
#define SPDR      *(unsigned char volatile *)(_IO_BASE + 0x2A)
#define BAUD      *(unsigned char volatile *)(_IO_BASE + 0x2B)
#define SCCR1     *(unsigned char volatile *)(_IO_BASE + 0x2C)
#define SCCR2     *(unsigned char volatile *)(_IO_BASE + 0x2D)
#define SCSR      *(unsigned char volatile *)(_IO_BASE + 0x2E)
#define SCDR      *(unsigned char volatile *)(_IO_BASE + 0x2F)
#define ADCTL     *(unsigned char volatile *)(_IO_BASE + 0x30)
#define ADR1      *(unsigned char volatile *)(_IO_BASE + 0x31)
#define ADR2      *(unsigned char volatile *)(_IO_BASE + 0x32)
#define ADR3      *(unsigned char volatile *)(_IO_BASE + 0x33)
#define ADR4      *(unsigned char volatile *)(_IO_BASE + 0x34)
#define OPTION    *(unsigned char volatile *)(_IO_BASE + 0x39)
#define COPRST    *(unsigned char volatile *)(_IO_BASE + 0x3A)
#define PPROG     *(unsigned char volatile *)(_IO_BASE + 0x3B)
#define HPRIO     *(unsigned char volatile *)(_IO_BASE + 0x3C)
#define INIT      *(unsigned char volatile *)(_IO_BASE + 0x3D)
#define TEST1     *(unsigned char volatile *)(_IO_BASE + 0x3E)
#define CONFIG    *(unsigned char volatile *)(_IO_BASE + 0x3F)
```