DUBLIN INSTITUTE OF TECHNOLOGY
KEVIN STREET DUBLIN 8

BSc. (Honours) Degree in Computer Science

Year 1

Semester 2 Examinations 2012/2013

Microprocessor Systems

Mr. F. Duignan
Dr. D. Lillis

Wednesday 22nd May 4.00 p.m. - 6.00 p.m.

Instructions
Attempt ***three*** out of the following four questions.
All questions carry equal marks.

Question 1. (33 Marks)

(a) Which 80386 arithmetic flags are set (=1) by the calculations that follow:
(note: some calculations will set multiple flags). You may ignore the Auxiliary Carry Flag (AF) and Parity Flag (PF)

(1) 1 – 1 [1]
(2) 1 – 2 [2]
(3) 0x7fff ffff + 1 [2]
(4) 0xffff ffff + 1 [2]

(b) The C-code in listing Q1(a) implements a simple 'for' loop. Write the 80x86 assembly language equivalent of this code paying particular attention to how the loop decision is made. **[10]**

(c)

(1) Using the ASCII table provided in Figure Q1a , show how the contents of memory occupied by the null terminated ASCII string "1234" differ from those occupied by the 16 bit value 1234. [4]

(2) Write a C-function that will accept a value in the range 0 to 15 and return the ASCII code for the equivalent hexadecimal digit (e.g. 'B' returned for an input of 11). The function prototype is as follows:

char IntToHexDigit(int val); [5]

(3) Using the function IntToHexDigit described above, write a function that will convert a 16 bit integer into a character string containing the hexadecimal representation of that value. The function prototype is as follows:

void IntToHexString(unsigned short val, char *HexString);

On entry the value to be converted is in the parameter ***val***. On exit, the character array ***HexString*** contains the hexadecimal representation of ***val***. [6]

Listing Q1a

```
int k;
for (k=0;k<10;k++)
{
     // body of loop not shown (not necessary for question)
}
```

COLLEGE EXAMINATIONS

AMENDMENTS TO EXAMINATION QUESTION PAPER

95.

COURSE REF: S228/9999

VENUE: B1

(+4) x sgl.

SUBJECT: MICROPROCESSOR SYSTEMS

DATE: WED 22nd MAY

TIME: 4-6pm

SIGNED:

INSTRUCTIONS:

Q.1.(c)(3)

The score for this part should be 5, not 6.

Figure Q1a (Partial) ASCII Table

' '	*32*	'!'	*33*	'"'	*34*	'#'	*35*	'$'	*36*	'%'	*37*	'&'	*38*	'''	*39*
'('	*40*	')'	*41*	'*'	*42*	'+'	*43*	','	*44*	'-'	*45*	'.'	*46*	'/'	*47*
'0'	*48*	'1'	*49*	'2'	*50*	'3'	*51*	'4'	*52*	'5'	*53*	'6'	*54*	'7'	*55*
'8'	*56*	'9'	*57*	':'	*58*	';'	*59*	'<'	*60*	'='	*61*	'>'	*62*	'?'	*63*
'@'	*64*	'A'	*65*	'B'	*66*	'C'	*67*	'D'	*68*	'E'	*69*	'F'	*70*	'G'	*71*
'H'	*72*	'I'	*73*	'J'	*74*	'K'	*75*	'L'	*76*	'M'	*77*	'N'	*78*	'O'	*79*
'P'	*80*	'Q'	*81*	'R'	*82*	'S'	*83*	'T'	*84*	'U'	*85*	'V'	*86*	'W'	*87*
'X'	*88*	'Y'	*89*	'Z'	*90*	'['	*91*	'\'	*92*	']'	*93*	'^'	*94*	'_'	*95*
'`'	*96*	'a'	*97*	'b'	*98*	'c'	*99*	'd'	*100*	'e'	*101*	'f'	*102*	'g'	*103*
'h'	*104*	'i'	*105*	'j'	*106*	'k'	*107*	'l'	*108*	'm'	*109*	'n'	*110*	'o'	*111*
'p'	*112*	'q'	*113*	'r'	*114*	's'	*115*	't'	*116*	'u'	*117*	'v'	*118*	'w'	*119*
'x'	*120*	'y'	*121*	'z'	*122*	'{'	*123*	'|'	*124*	'}'	*125*	'~'	*126*	'⌂'	*127*

Question 2. (33 Marks)

(a)
(1) Show how a **#define** statement can be used to define a symbol **P1IN** which is equivalent to the 8 bit contents of memory address 0x20 in a C program.

[4]

(2) Write C-code that can be used to set BIT 3 of an 8-bit memory location identified by the symbol **P1OUT** without affecting other bits.

[3]

(3) Write C-code that can be used to clear BIT 5 of an 8-bit memory location identified by the symbol **P1OUT** without affecting other bits.

[3]

(4) What is the purpose of a **Data Direction Register** in MSP430 parallel port I/O.

[4]

(b)
(1) A TIMSP430 has a free-running 16 bit counter that counts at a rate of 1000Hz. The contents of the counter can be read/written using the symbol **TAR**. Write C-code which uses **TAR** to measure the number of milliseconds between a low to high transition on PORT 1, bit 1 and a low to high transition on PORT 1, bit 2.

[10]

(2) How would you deal with an overflow of TAR and still produce an accurate measurement of time?

[6]

(3) How do microprocessors typically produce a stable, known, clock signal (square wave)?

[3]

Question 3. (33 Marks)

(a) Show how the storage of the 32 bit number 0x12345678 differs between little-endian and big-endian computer systems. [4]

(b) Listing Q3a contains assembly language for a function that adds two 128 numbers on a 32 bit 80x86 system. Listing Q3b shows its invocation from C.

(1) What are the contents of the stack after execution of Line A. [6]
(2) What number is moved into eax when Line B is executed? Justify your answer [9]
(3) What is the difference between the ***add*** and ***adc*** instructions? [4]
(4) Why are the registers popped in reverse order to their being pushed? [4]
(5) What will be displayed by the program? [6]

Listing Q3a

```
	.text
	.global Add128
# C-protoype:  void Add128(int *operand1, int * operand2, int *result);
Add128:
# This function adds two 128 bit integers
	push %ebp
	mov %esp,%ebp ................................A
	push %eax
	push %ebx
	push %ecx
	push %edx
# Copy the addresses to registers for use as pointers
	mov 8(%ebp),%ebx
	mov 12(%ebp),%ecx
	mov 16(%ebp),%edx
# Process bytes 0-3
	mov (%ebx),%eax
	add (%ecx),%eax
	mov %eax,(%edx)
# Process bytes 4-7
	mov 4(%ebx),%eax ................................B
	adc 4(%ecx),%eax
	mov %eax,4(%edx)
# Process bytes 8-11
	mov 8(%ebx),%eax
	adc 8(%ecx),%eax
	mov %eax,8(%edx)
# Process bytes 11-15
	mov 12(%ebx),%eax
	adc 12(%ecx),%eax
	mov %eax,12(%edx)
# restore changed registers
	pop %edx
	pop %ecx
	pop %ebx
	pop %eax
	pop %ebp
# return to main
	ret
	.end
```

Listing Q3B

```
void Add128(int *op1, int *op2, int *result) asm("Add128");
int op1[]={0xccddeeff,0x8899aabb,0x44556677,0x00112233};
int op2[]={0xccddeeff,0x8899aabb,0x44556677,0x00112233};
int res[]={0,0,0,0};
int main(int argc, char *argv[])
{
	Add128(op1,op2,res);
	printf("%08x %08x %08x %08x \n",op1[3],op1[2],op1[1],op1[0]);
	printf("+\n");
	printf("%08x %08x %08x %08x \n",op2[3],op2[2],op2[1],op2[0]);
	printf("-----------------------------------\n");
	printf("%08x %08x %08x %08x \n",res[3],res[2],res[1],res[0]);
}
```

Question 4. (33 Marks)

(a)

Using suitable examples, explain what is meant by the following terms in the context of microprocessors systems

(1) Interrupt [4]

(2) Interrupt service routine [4]

(3) Interrupt vector table [4]

(4) Why is Interrupt driven I/O preferred over polled I/O in portable, battery powered applications? [4]

(b)

Explain each of the following terms as applied to serial data transmission

(1) Baud rate [2]

(2) Asynchronous [2]

(3) Handshaking [3]

(4) Simple parity based error checking can be quite unreliable. Using an example, illustrate why this is true. Again, using a suitable example, demonstrate why CRC error checking can be a far more reliable form of error checking. [7]

(5) A 100GByte file is to be transferred over a network 100Mbit/s connection. Assuming a transmission overhead of 2%, calculate how long it will take to transmit this file. [3]