201 - QIJ3OYAZHI5NGYG

0406186522

FÖRSÄTTSBLAD TENTAMEN/ EXAMINATION COVER

Jag intygar att mobiltelefon och annan otillåten elektronisk utrustning är avstängd och förvaras på anvisad plats. / I hereby confirm that mobile phones and other unauthorized electronic equipment is shut off and placed according to instructions

MARKERA MED "X"/
MARK WITH "X" ☒

IFYLLES AV STUDENT OCH TENTAMENSVAKT/
TO BE FILLED IN BY THE STUDENT AND THE INVIGILATOR:

KURSKOD / COURSE CODE: IS1200	EFTERNAMN / FAMILY NAME: Hedén Malm
KURSNAMN / COURSE NAME: Datorteknik, grundkurs	FÖRNAMN / FIRST NAME: Jacob
PROVKOD / TEST CODE: TEN1	NAMNTECKNING / YOUR SIGNATURE: Jacob Hedén
TENTAMENSDATUM / EXAMINATION DATE Y/Y/Y/Y M/M D/D: 2019-03-15	PERSONNUMMER / PERSONAL NUMBER Y/Y/M/M/D/D: 980405-1499

PROGRAMKOD / PROGRAM CODE:	INLÄMNINGSTID / TIME SUBMITTED:	SIGNATUR TENTAMENSVAKT / SIGNATURE INVIGILATOR:	ANTAL BLAD / NO OF SHEETS:
CINTE	11:27		09

MARKERA BEHANDLADE UPPGIFTER MED "X "OCH EJ BEHANDLADE UPPGIFTER MED "-" /
MARK WITH "X" PROBLEMS SOLVED. MARK WITH "-" PROBLEMS NOT ATTEMPTED

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
X	X	9	X	X	X	X		X											

IFYLLES AV INSTITUTIONEN / TO BE FILLED IN BY THE DEPARTMENT:

BEDÖMNING / ASSESSMENT																			
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

BONUSPOÄNG/ BONUS POINTS: ,

SLUTSUMMA / FINAL POINTS: ,

BETYG/ GRADE:

Godkänns av examinator /
approved by Examiner.....

Family name, first name	Personal Registration Number	Programme	Sheet no.	Problem no.
Hedén Malm, Jacob	980405-1499	CINTE	01	01

a)

$t1 = -32

sra : r, 0, 3, rd = rt >>> shamt

$t0 $t1 2

Oplode = 000000

rs = 00000

rt = 9 = 01001

rd = 8 = 01000

shamt = 00010 = 2

funct = 3 = 000011

sra $t0, $t1, 2 :

000000 00000 01001 01000 00010 000011

0x 0 0 0 9 4 0 8 3

0x00094083

i)

32 = 01000000

-32 = 10111111 +1

= 11000000 >>> 2

= 11111111 000

= -8

$t0 = -8

Family name, first name	Personal Registration Number	Programme	Sheet no.	Problem no.
Hesén Malm, Jacob	1998 0405-1499	CINTE	02	01

b)

l1: *P += 2;

l2: P++;

lSt = [9, 10, 12, 0]

Family name, first name	Personal Registration Number	Programme	Sheet no.	Problem no.
Helén Malm, Jacob	1998-0405-1499	CINTE	03	02

a)

i. True

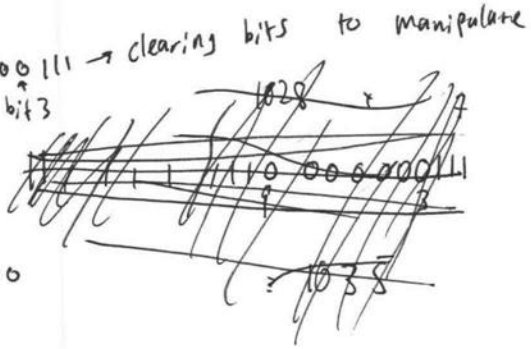
ii. False, volatile is a keyword used to tell the compiler not to do certain efficiency compilations that assume that a variable will not change if it isn't changed in Sofware. Useful when IO might change the value.

iii. True

iv. False; after the interrupt is handled, the program resumes as normal.

b)

```
lui   $t0, 0x9500      // create pointer to correct mem address
ori   $t0, $t0, 0x3f00
lw    $t1, 0($t0)      // access data at address
~~$t1, $t1, $t2~~      // -505 = 1111 0000 0111 → clearing bits to manipulate
                       //        bit4  bit3
andi  $t1, $t1, -505
ori   $t1, $t1, 968    // 968: 0111 1001 000
                       //      9   5   3
sw    0($t0), $t1      // update IO device
```



Family name, first name	Personal Registration Number	Programme	Sheet no.	Problem no.
Heden Malm, Jacob	1998 0405 - 1499	CINTE	04	~~03~~ 04

a)

A = 25:21 = rs

rs = $a0 = 4

A = 0x4

B = Value of register rt

rt = $t8 = 0x2f

C = Sum of ALU operation

which is 0x3500 - 8 = 0x34f8

D = Next instruction address.

we do not know current instruction address, so D = unknown

E = 0x1, Write is enabled because we want to write the value of address 0x34f8 to $t8.

b) The first hazard occurs between instruction 1 and instruction 2. This is because we use register $t0 before we have written the sum of the previous operation in it. We solve this by Stalling 1 cycle and forwarding.

Family name, first name	Personal Registration Number	Programme	Sheet no.	Problem no.
Hedén Malm, Jacob	1998 0405 - 1499	CINTE	05	05

a)

~~8 bytes = 64 bits~~

~~log2 64 = 6~~

i. byte Offset = $\log_2 8 = 3$ bits

Set Offset = 16 - 3 - 6

= 7

$2^7 = 32 * 2 * 2 =$ **128 sets**

ii.

128 * 8 = 512 + 2 = 1024

Capacity = **1024 bytes**

iii.

Address = 100101 0011111 000

tag | set = 31 | byte Offset, arbitrary.

31 = 16 + 8 + 4 + 2 + 1

iii. cont.

~~1001 0100 0111 1000~~

1001 01 00 1111 1 000

9 4 f 8

address = 0x 94f8

h/m

b)

byte Offset = 4 bits = $\log_2 16$

Set field = $\log_2 4096$ = 12 bits

tag = 32 - 16 = 16 bits

in total 3 misses, 4 hits.

hit rate = 4/7.

instr #	tag	set	byte
1	4000	301	C x
2	4000	302	0 x
3	4000	302	4 ✓
4	4000	302	8 ✓
5	4000	302	C ✓
6	4000	303	0 x
7	4000	303	4 ✓

Family name, first name	Personal Registration Number	Programme	Sheet no.	Problem no.
Hedén Malm, Jacob	1998 0405-1499	CINTE	06	06

a)

max speedup = 5.

$$5 = \frac{10}{T_{after}}$$

$T_{after} = 2$

∴ Sequential part takes 2 seconds.

$T_{original} = T_{affected} + T_{unaffected}$.

$10 = 8 + 2$

$T_{After} = \frac{8}{N} + 2$

$T_{After} = \frac{8}{4} + 2$

$T_{After} = 2 + 2$

$T_{After} = 4$ seconds

$$Speedup = \frac{T_{Before}}{T_{After}}$$

$$Speedup = \frac{10}{4}$$

$$Speedup = \frac{5}{2}$$

b)

i. True

ii. Wrong, this is provided by Segmented virtual memory. A mux chooses 1 input stream from many.

iii. True

iv. True

v. True

Family name, first name	Personal Registration Number	Programme	Sheet no.	Problem no.
Hedén Malm, Jacob	1998 0405-1499	CINTE	07	07

7)

a) Data level parallelism means that you can operate one instruction on multiple sets of data in parallel. For example, a VLIW add instraction can take 4 operands and perform 2 separate additions at the same time. Multiple sets of hardware make this possible. MIMD means multiple instruction multiple data. This means that we have multiple processes (instruction sets) running concurrently. Each one of these processes can use data level parallelism. As such, the difference is that MIMD has more than one instruction set, but the similarity is that data-level parallelism can be present in both.

b) If we poll a timer we are constantly checking to see if the period overflow flag is set on the timer. This is something we constantly check in ~~a Soft~~ our sofware somewhere. If we use interrupts, we tell the timer to trigger an interrupt when the timer overflow flag is set. This will cause a break from normal activity in the program where we deal with the interrupt, after which the program seamlessly continues. The similarity is that both methods allow us to see when the timer reaches it's max value, the difference is that with interrupts we do not constantly have to check this, the hardware will notify us when it happens.

Family name, first name	Personal Registration Number	Programme	Sheet no.	Problem no.
Heden Malm, Jacob	19980405-1499	CINTE	08	07

c) Hardware multithreading means that we find ways to concurrent processes on the same core. We get one set of hardware to ~~[illegible]~~ operate more than 1 set of instructions. This allows us to have the performance of ~~multiple~~ cores more cores than we physically have, in the form of threads.

Family name, first name	Personal Registration Number	Programme	Sheet no.	Problem no.
Hedén Malm, Jacob	1998 0405- 1499	CINTE	09	09

a)

```
int foo(int *p) {
    if(*p == 0) {
        return 0; }

    else {
        int counter = 0; }
        char *t1 = *(*p);
    while (*t1 != 0) {
        t1 = *(*p);
        counter ++;
        *p ++; }

    p++;

    int c;
    c += foo(p);

    return c;
```

b) The program is adding up all of the characters within the array and returning the total amount of characters. As such, it would print the number 9.

THisisfun → 9 total chars.
1 2 3 4 5 6 7 8 9