Student: **Collegekaartnummer:**

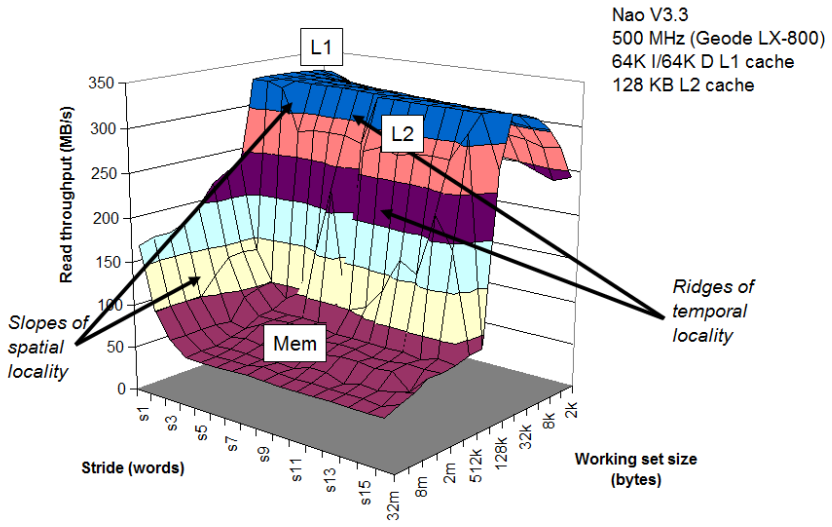
Tentamen Computersystemen voor AI programmeurs

baiCSAI3 2e jaar bachelor AI, 2e semester 23 maart 2011

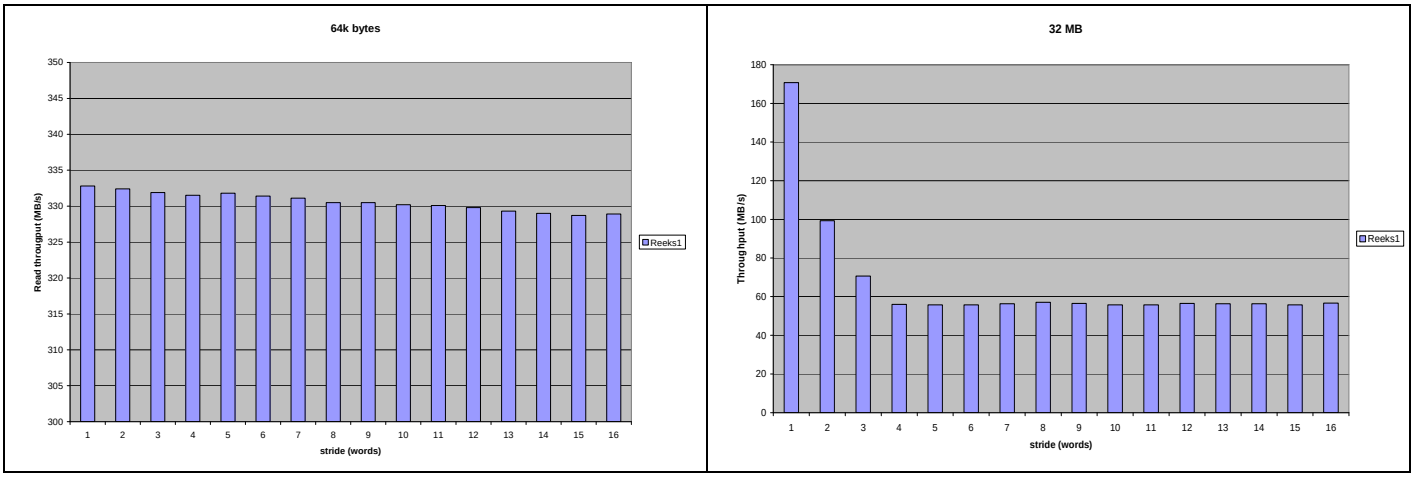
vraag 1

U heeft een Aldebaran Nao onder uw hoede gekregen. Deze humanoid robot bevat een AMD Geode LX-800 processor.

U meet het geheugen als functie van de *spatial* en *temporal locality*. U krijgt het volgende resultaat:



In de volgende figuur zijn twee doorsneden van deze figuur te zien, voor respectievelijk een kleine en grote *working set*.



a) Met welke snelheid kan data uit het RAM geheugen gelezen worden?
b) Met welke snelheid kan data uit de *cache* gelezen worden?
c) Reken dit voor een *4-byte word* om in nanoseconden en tikken.

Student: **Collegekaartnummer:**

De Geode processor heeft een klassieke *in-order pipeline* architectuur. De processor kan één *integer* operatie per *cycle* starten. Voor *floating point* operaties is een aparte *queue* van vier diep.
Als de *floating point queue* vol zit wordt de *integer queue* opgehouden (*stalled*).

d) Zal de functie `combine6` (bladzijde 549) bij een Geode processor een betere *performance* geven dan de functie `combine5` (bladzijde 544)? Motiveer uw antwoord.

vraag 2

Bekijk het volgende stukje code, waar `R`, `S` en `T` constanten zijn die elders zijn gedefinieerd:

```
int A[R][S][T];

int store_ele(int i, int j, int k, int *dest)
{
   *dest = A[i][j][k];
   return sizeof (A);
}
```

Na compilatie, heeft GCC de volgende assembly code gegenereerd:

```
        movl 8(%ebp),%ecx
        movl 12(%ebp),%eax
        leal (%eax,%eax,8),%eax
        movl %ecx,%edx
        sall $6,%edx
        subl %ecx,%edx
        addl %edx,%eax
        addl 16(%ebp),%eax
        movl A(,%eax,4),%edx
        movl 20(%ebp),%eax
        movl %edx,(%eax)
        movl $2772,%eax
```

a) Op pagina 270 staat formule 3.1 voor een tweedimensionaal *array element*.
Geef een equivalente formule voor een driedimensionaal *array element*.

b) Interpreteer de assembly code om er achter te komen welke waarden voor de constanten `R`, `S` en `T` zijn gebruikt.

vraag 3

De MIPS processor van een Aibo heeft 32 generieke registers, terwijl men het in de IA32 met acht *integer* en acht *floating-point* registers moet doen. Vier van de generieke MIPS-registers worden door de compiler gebruikt om functieargumenten te bewaren, terwijl men in de IA32 architectuur hiervoor de *stack* gebruikt. Bekijk de volgende code:

```
int swap_add(int *xp, int *yp)
{
    int x = *xp;
    int y = *yp;

    *xp = y;
    *yp = x;
    return x + y;
}
```

Student: **Collegekaartnummer:**

```
int caller()
{
    int arg1 = 534;
    int arg2 = 1057;
    int sum = swap_add(&arg1, &arg2);
    int diff = arg1 - arg2;

    return sum * diff;
}
```

Als we deze code compileren voor de MIPS architectuur van de Aibo, krijgen we de volgende *assembly code*:

```
swap_add:
      the set up of a new frame for the function
      .frame $sp,0,$31  # stackpointer and returnadress (reg31)
      lw    $6,0($4)    # load x with value of arg1 (reg4)
      lw    $3,0($5)    # load y with value of arg2 (reg5)
      addu  $2,$6,$3    # return value (reg2) is addition of x and y
      sw    $3,0($4)    # store y at adress of arg1
      sw    $6,0($5)    # store x at adress of arg2
      j     $31         # return
```

Als we dezelfde code compileren voor IA32, ziet de *assembly code* er als volgt uit:

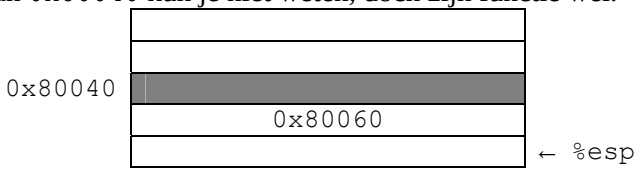
```
swap_add:
1     the set up of a new frame for the function
2     pushl %ebp        save old framepointer
3     movl %esp,%ebp    set new %ebp just after the stackpointer %esp
4     pushl %ebx        callee save
5     the body of the function
6     movl 8(%ebp),%edx
7     movl 12(%ebp),%ecx
8     movl (%edx),%ebx
9     movl (%ecx),%eax
10    movl %eax,(%edx)
11    movl %ebx,(%ecx)
12    addl %ebx,%eax
13    restoring the frame of the calling function
14    popl %ebx              callee restore
15    movl %ebp,%esp         recalcute old stackpointer
16    popl %ebp              restoring old framepointer
17    ret
```

a. Welk register wordt er bij de IA32 gebruikt om het eindresultaat te bewaren?
b. Waarom worden `%eax`, `%edx` en `%ecx` niet gesaved op de *stack*?
c. Is er bij de IA32 ook een register voor het *returnadress*?
d. Wordt de *stackpointer* verhoogd of verlaagd met 4 voor iedere `pushl`?
e. Als bij het aanroepen van de functie `swap_add` de stackpointer de waarde `0x80040had`, welke waarde krijgt `%ebp` dan op regel 3?
f. Welke waarde krijgt `%esp` op regel 4?
g. Vlak voor de aanroep naar `swap_add` staat de volgende code van de `caller`:

```
movl $534,-8(%ebp)
movl $1057,-4(%ebp)
addl $-8,%esp
leal -4(%ebp),%eax
pushl %eax
leal -8(%ebp),%eax
pushl %eax
call swap_add
```

Student: **Collegekaartnummer:**

Teken de *stack* op regel 5 van de code `swap_add` van geheugenadress `0x80048` tot `0x80038`. De inhoud van `0x80040` kun je niet weten, doch zijn functie wel.



Vraag 4

Bestudeer het volgende C programma, inclusief regelnummers:

```
1 int main() {
2     int counter = 0;
3     int pid;
4
5     while (counter < 4 && !(pid = fork())) {
6           counter += 2;
7           printf("%d", counter);
8     }
9
10    if (counter > 0) {
11          printf("%d", counter);
12    }
13
14    if (pid) {
15          waitpid(pid, NULL, 0);
16          counter += 3;
17          printf("%d", counter);
18    }
19 }
```

U mag voor het beantwoorden van de vraag uitgaan van de volgende aannamen:
• Alle processen kunnen ongestoord hun code afmaken; de processen worden niet onderbroken door *traps* in system calls.
• `printf`() wordt niet tussentijds onderbroken en voert een `fflush`(`stdout`) uit na het printen van de invoer, voordat de functie terugkomt in de *user code*.
• Logische operatoren zoals && evalueren hun argumenten van links naar rechts. Hierbij evalueren ze het minimale aantal argumenten om tot een beslissing te komen.

a) Geef een lijst van alle mogelijke uitvoer van dit programma.

b) Als in regel de *operator* > wordt vervangen door de *operator* >=, dan verschijnen er nullen in de uitvoer. Hoeveel verschillende regels kan het programma nu uitvoeren na deze verandering. Geef het aantal, niet een lijst.

Succes!