

T. C

MANİSA CELAL BAYAR ÜNİVERSİTESİ

ASSEMBLY ile HARMONİK
ORTALAMA HESAPLAMA

İSMET KIZGIN

BİLGİSAYAR MİMARİSİ

MÜGE EREL
ÖZÇEVİK

MANİSA 2020

BİLGİSAYAR MİMARİSİ

Assembly ile Harmonik Ortalama Hesaplama

ANALİZ	2
PROBLEM TANIMI	2
TASARIM	2
ÇÖZÜM ALGORİTMASI	2
PROGRAMCI KATALOĞU	2
PROJE İÇİN ZAMANLAMA PLANI	2
SPEEDUP	3
Github Hesabında Bulunan b.asm Örneğinin SPEEDUP Hesaplaması	3
Github Hesabında Bulunan c.asm Örneğinin SPEEDUP Hesaplaması	3
PROJE KAYNAK ÇIKTI KODU	3
PROJE BAĞLANTILARI	9
GITHUB BAĞLANTISI	9
KULLANICI KATALOĞU	9

BİLGİSAYAR MİMARİSİ

Assembly ile Harmonik Ortalama Hesaplama

ANALİZ

PROBLEM TANIMI

Kullanıcı tarafından girilen 10 integer değeri bir dizi içinde tutarak harmonik ortalamasının alınması gerekmektedir.

TASARIM

ÇÖZÜM ALGORİTMASI

Kullanıcı tarafından girilen 10 integer değeri için 40 byte yer ayrılır. Loop ile bir döngü oluşturularak kullanıcıya değerleri girmesi gerektiği söylenir. Girilen değerlerin tamamlandığını koşul ile sağladıktan sonra bir loop yönlendirilerek 40 byte yer ayırdığımız dizimizin için de olan değerleri double dönüştürüp hesaplamalar sağlanır. Son olarak hesaplanan ortalama ekrana bastırılır.

PROGRAMCI KATALOĞU

PROJE İÇİN ZAMANLAMA PLANI

PROJE İÇİN AYRILAN TOPLAM SÜRE: 1 Hafta

BİLGİSAYAR MİMARİSİ

Assembly ile Harmonik Ortalama Hesaplama

SPEEDUP

Github Hesabında Bulunan b.asm Örneğinin SPEEDUP Hesaplaması

Komut Sayısı: **476**

İş Hattı Olmadan Hesaplam: **$476 * 8 = 3808$**

İş Hattı ile Hesaplam: **$476 * 2 + 4 * 2 = 960$**

SpeedUp: **$3808 / 960 = 3,966666$**

Github Hesabında Bulunan c.asm Örneğinin SPEEDUP Hesaplaması

Komut Sayısı: **413**

İş Hattı Olmadan Hesaplam: **$413 * 8 = 3304$**

İş Hattı ile Hesaplam: **$413 * 2 + 4 * 2 = 834$**

SpeedUp: **$3304 / 834 = 3,961630$**

PROJE KAYNAK ÇIKTI KODU

```
.data
```

.data altında proje içinde kullanılacak sabitler belirlenir

```
myArray: .space 40
```

40 byte bir alan ayrılarak integer bir dizi oluşturulur

BİLGİSAYAR MİMARİSİ

Assembly ile Harmonik Ortalama Hesaplama

```
zero: .double 0.0  
one: .double 1.0
```

proje içinde sabit değerlerin daha kolay atanması için sabit değerler oluşturulur

```
N: .double 10.0
```

hesaplama sırasında ki toplam değer sabiti için oluşturulur

```
input_number_text: .asciiz ". Lütfen Sayı Giriniz: "  
result_text: .asciiz "\nGirmis Oldugunuz Sayilarin Harmonik Ortalamasi: "  
ln: .asciiz "\n"
```

uygulama içinde ki sözel ifadeler için sabitler oluşturulur

```
main:  
  
ldc1 $f10, zero
```

algoritmanın ilk adımına register değerine sıfır değerini atayarak başlıyoruz

```
addi $t0, $zero, 0  
addi $t1, $zero, 1
```

oluşturulacak loop için artış miktarı değişkeni ve kullanıcıya gösterilecek olan değer numarası değişkeni oluşturulur

BİLGİSAYAR MİMARİSİ

Assembly ile Harmonik Ortalama Hesaplama

```
while_input:  
    beq $t0, 40, harmonic_average
```

while_input adında loop oluşturulur ve beq ile koşulu sağlanır

```
li $v0, 1  
  
addi $a0, $t1, 0  
  
syscall
```

"1. Değeri Girin: " şeklinde bir ifadenin başında girilen değerin numarası ekrana basılır

```
li $v0, 4  
  
la $a0, input_number_text  
  
syscall
```

sabit değer içinde bulunan değer ekrana basılır

```
li $v0, 5  
  
syscall
```

kullanıcıdan integer değer girişi yapılması sağlanır

```
sw $v0, myArray($t0)
```

girilen değer oluşturulmuş olan dizi içine aktarılır

BİLGİSAYAR MİMARİSİ

Assembly ile Harmonik Ortalama Hesaplama

```
addi $t0, $t0, 4  
  
addi $t1, $t1, 1
```

oluşturulmuş sayac değişkenleri artırılır işlevine göre, t0 değişkeni integer değer 4 byte yer kaplamasından kaynaklı 4 artırılır

```
j while_input
```

loop başına geri döner

```
harmonic_average:
```

harmonik hesaplama için gerekli loop oluşturulur

```
addi $t0, $zero, 0  
  
ldc1 $f8, one
```

loop için belirlenen artış miktarı sıfırlanır ve işlemde kullanılmak üzere iki adet sabit oluşturulur

```
while_harmonic:  
  
beq $t0, 40, calculation
```

loop içinde bir loop daha oluşturularak koşulu belirtilmiştir

BİLGİSAYAR MİMARİSİ

Assembly ile Harmonik Ortalama Hesaplama

```
lw $t2,myArray($t0)
```

belirtilen koşul doğrultusunda dizi içine atılanlar loop yardımı ile tek tek okunur

```
mtc1.d $t2, $f2
```

```
cvt.d.w $f2, $f2
```

dizi içinden alınan integer değerler double çevrilir

```
div.d $f2, $f8, $f2
```

```
add.d $f4, $f4, $f2
```

harmonik hesaplama için girilen sayılar ile bölme ve toplama işlemi gerçekleşir

```
addi $t0, $t0, 4
```

loop için belirlenen sayaç 4 artırılır (1 integer deger 4 byte eşittir)

```
j while_harmonic
```

loop başına geri dönülür

```
calculation:
```

girilen değerler sabit ile bölünüp daha sonra toplandıktan sonra hesaplama label gelir

BİLGİSAYAR MİMARİSİ

Assembly ile Harmonik Ortalama Hesaplama

```
ldc1 $f8, N  
  
div.d $f12, $f8, $f4
```

N (Toplam integer sayısı) sabiti ile değişken içine tanımlanır ve toplam değere bölünür

```
exit:  
  
li $v0, 4  
  
la $a0, result_text  
  
syscall  
  
  
li $v0, 3  
  
syscall
```

exit label'ı yardımı ile result_text sabit metnini ekrana yazıp daha sonra bulunan sonuç ekrana yazılır

PROJE BAĞLANTILARI

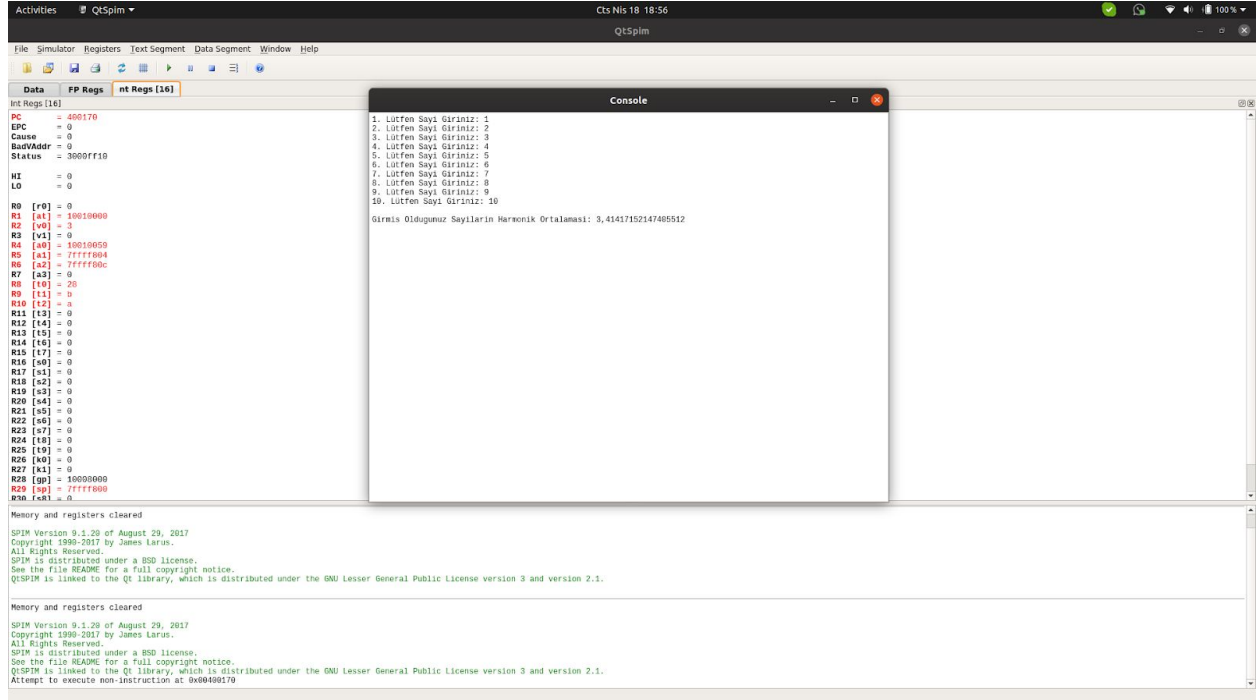
GITHUB BAĞLANTISI

<https://github.com/ismetkizgin/harmonic-average-calculation-Assembly>

BİLGİSAYAR MİMARİSİ

Assembly ile Harmonik Ortalama Hesaplama

KULLANICI KATALOĞU



The screenshot displays the QSPIM (QtSPIM) simulator interface. The top menu bar includes 'File', 'Simulator', 'Registers', 'Text Segment', 'Data Segment', 'Window', and 'Help'. The 'Registers' window is open, showing the 'nt Regs [16]' tab. It lists various registers and their values, including PC (000170), EPC (0), Cause (0), BadVAddr (0), Status (3000ff10), HI (0), LO (0), R0 (0), R1 (0), R2 (0), R3 (0), R4 (0), R5 (0), R6 (0), R7 (0), R8 (0), R9 (0), R10 (0), R11 (0), R12 (0), R13 (0), R14 (0), R15 (0), R16 (0), R17 (0), R18 (0), R19 (0), R20 (0), R21 (0), R22 (0), R23 (0), R24 (0), R25 (0), R26 (0), R27 (0), R28 (0), R29 (0), R30 (0), and R31 (0). The console window is also open, displaying the following output:

```
1. Lütfen Sayı Giriniz: 1
2. Lütfen Sayı Giriniz: 2
3. Lütfen Sayı Giriniz: 3
4. Lütfen Sayı Giriniz: 4
5. Lütfen Sayı Giriniz: 5
6. Lütfen Sayı Giriniz: 6
7. Lütfen Sayı Giriniz: 7
8. Lütfen Sayı Giriniz: 8
9. Lütfen Sayı Giriniz: 9
10. Lütfen Sayı Giriniz: 10

Girilen Olduğunuz Sayıların Harmonik Ortalaması: 3,41417152147485512
```

The bottom of the simulator window shows the memory and registers cleared, and the SPIM version 9.1.20 of August 29, 2017. It also includes copyright information for James Larus and a disclaimer stating that the software is distributed under a BSD license and is linked to the Qt library, which is distributed under the GNU Lesser General Public License version 3 and version 2.1.