Logo, company name

Description automatically generated

Documentație procesor AC

Petruta Darius-Ispas

Grupa 30231

Problema abordată de mine pe acest procesor este găsirea celui mai mare factor prim din descompunerea unui număr și verificare dacă acesta este la putere pară/impară.

Am transpus codul c în asamblare și cod mașina și la final pentru afișarea rezultatelor am făcut adunarea pe rând a registrelor 3 și 4 cu 0 pentru că acolo se vor afla rezultatele.

Descriere cod in C:

#include <iostream>

using namespace std;

int main() {

int n, p = 0, d = 2, i;

cin >> n;

while (n != 1) {

if (n % d == 0) {

p++;

}

else {

p = 1;

while (n % d != 0) {

d++;

}

}

n = n / d;

}

if (p % 2 == 0) {

i = 1;

}

else {

i = 0;

}

cout << d << " " << i;

}

În general am urmat descrierile din laborator și m-am ajutat și de curs. Am reușit să implementez fără erori.

Am implementat pentru instrucțiunile de tip I bneq și bg. Bneq va face un salt peste numărul de instrucțiuni arătat de imediat dacă cele 2 registre nu sunt egale. Bg va face un salt peste numărul de instrucțiuni arătat dacă primul registru este mai mare decât al 2-lea.

Descriere cod masina:

b"010\_000\_001\_0000000", --0 lw $1, 0($0)

b"001\_000\_010\_0000010", --1 addi $2, $0, 2

b"000\_000\_000\_011\_0\_000", --2 add $3, $0, $0

b"000\_000\_000\_100\_0\_000", --3 add $4, $0, $0

b"001\_000\_111\_0000001", --4 addi $7, $0, 1

b"000\_000\_001\_101\_0\_000", --5 add $5, $0, $1

b"000\_101\_010\_101\_0\_001", --6 sub $5, $5, $2

b"110\_101\_000\_1111110", --7 bg $5, $0, -2

b"100\_101\_000\_0010001", --8 beq $5, $0, 17

b"001\_000\_011\_0000001", --9 addi $3, $0, 1

b"000\_000\_001\_101\_0\_000", --10 add $5, $0, $1

b"001\_010\_010\_0000001", --11 addi $2, $2, 1

b"000\_101\_010\_101\_0\_001", --12 sub $5, $5, $2

b"110\_101\_000\_1111110", --13 bg $5, $0, -2

b"101\_000\_101\_1111011", --14 bneq $5, $0, -5

b"000\_001\_010\_001\_0\_001", --15 sub $1, $1, $2

b"001\_000\_110\_0000001", --16 addi $6, $0, 1

b"110\_001\_000\_1111101", --17 bg $1, $0, -3

b"000\_000\_110\_001\_0\_000", --18 add $1, $0, $6

b"101\_001\_111\_1110001", --19 bneq $1, $7, -15

b"000\_000\_011\_100\_0\_000", --20 add $5, $0, $3

b"001\_000\_001\_0000010", --21 addi $1, $0, 2

b"000\_101\_001\_101\_0\_001", --22 sub $5, $5, $1

b"110\_101\_000\_1111110", --23 bg $5, $0, -2

b"100\_101\_000\_0000011", --24 beq $5, $0, 3

b"111\_0000000010111", --25 jmp 23

b"001\_011\_011\_0000001", --26 addi $3, $3, 1

b"111\_0000000001111", --27 jmp 15

b"001\_000\_100\_0000001", --28 addi $4, $0, 1

b"001\_000\_011\_011\_0\_000", --29 add $3, $0, $3

b"001\_000\_100\_100\_0\_000", --30 add $4, $0, $4