***Instituição:*** Instituto Federal de Educação, Ciência e Tecnologia - Paraíba (IFPB).

***Disciplina:*** Microprocessadores e microcontroladores**.**

**Curso:** Engenharia de Computação – 4º período

***Professor:*** Fagner de Araujo Pereira.

Aluno (a): Jardel Brandon de Araujo Regis

***Tabela de programação – Processador didático***

**Set de instruções do processador**

\*\*\*Instruções aritméticas e lógicas Campos de bits do código de máquina correspondente

**00000 ADD ra, rb, rc** – [rc = ra + rb] Opcode(5)+destino(4)+operandoA(4)+operandoB(4)

**00001 ADI ra, #cte** – [ra = ra + #cte] Opcode(5)+destino(4)+constante(8)

**00010 SUB ra, rb, rc** – [rc = ra – rb] Opcode(5)+destino(4)+operandoA(4)+operandoB(4)

**00011 SUBI ra, #cte** – [ra = ra – cte] Opcode(5)+destino(4)+constante(8)

**00100 AND ra, rb, rc** – [rc = ra . rb] Opcode(5)+destino(4)+operandoA(4)+operandoB(4)

**00101 ANDI ra, #cte** – [ra = ra . #cte] Opcode(5)+destino(4)+constante(8)

**00110 OR ra, rb, rc** – [rc = ra | rb] Opcode(5)+destino(4)+operandoA(4)+operandoB(4)

**00111 ORI ra, #cte** – [ra = ra | #cte] Opcode(5)+destino(4)+constante(8)

**01000 EOR ra, rb, rc** – [rc = ra XOR rb] Opcode(5)+destino(4)+operandoA(4)+operandoB(4)

**01001 EORI ra, #cte** – [ra = ra XOR #cte] Opcode(5)+destino(4)+constante(8)

**01010 NOT ra, rb** – [rb = ~ra] Opcode(5)+destino(4)+operando(4)

**01011 INC ra, rb** – [rb = ra + 1] Opcode(5)+destino(4)+operando(4)

**01100 DEC ra, rb** – [rb = ra – 1] Opcode(5)+destino(4)+operando(4)

**01101 MUL ra,rb** – [rx = ra \* rb] Opcode(5)+operandoA(4)+operandoB(4)

**01110 MULI ra, #cte** – [rx = ra \* #cte] Opcode(5)+operando(4)+constante(8)

**01111 SHL ra, rb** – [rb = ra << 1] Opcode(5)+destino(4)+operando(4)

**10000 SHR ra, rb** – [rb = ra >> 1] Opcode(5)+destino(4)+operando(4)

\*\*\*Instruções de transferência de dados

**10001 MOV ra, rb** – [rb = ra] Opcode(5)+destino(4)+operando(4)

**10010 LDI ra, #cte** – [ra = #cte] Opcode(5)+destino(4)+constante(8)

**10011 IN ra** – [ra = porta de entrada] Opcode(5)+destino(4)

**10100 OUT ra** – [porta de saída = ra] Opcode(5)+operando(4)

\*\*\*Instruções de desvio

**10101 JMP k** – [PC = k] Opcode(5)+constante(8)

**10110 RJMP k** – [PC = PC + k] Opcode(5)+constante(8)

**10111 BRL ra, rb, y4** – [PC = PC+y se ra<rb] Opcode(5)+operandoA(4)+operandoB(4)+constante(4)

**11000 BRLE ra, rb, y4** – [PC = PC+y se ra<=rb] Opcode(5)+operandoA(4)+operandoB(4)+constante(4)

**11001 BRE ra, rb, y4** – [PC = PC+y se ra=rb] Opcode(5)+operandoA(4)+operandoB(4)+constante(4)

**11010 BRNE ra, rb, y4** – [PC = PC+y se ra!=rb] Opcode(5)+operandoA(4)+operandoB(4)+constante(4)

**11011 BRGE ra, rb, y4** – [PC = PC+y se ra>=rb] Opcode(5)+operandoA(4)+operandoB(4)+constante(4)

**11100 BRG ra, rb, y4** – [PC = PC+y se ra>rb] Opcode(5)+operandoA(4)+operandoB(4)+constante(4)

**11101 BRZ ra, k** – [PC = PC+k se ra=0] Opcode(5)+operando(4)+constante(8)

\*\*\*Instruções de controle do processador

**11110 NOP** – [nenhuma operação] Opcode(5)

**11111 RESET** – [PC=0] Opcode(5)

Nas instruções acima, os seguintes termos são utilizados:

ra, rb e rc – representa um dos registradores do banco;

rx – é o registrador de 16 bits formado pela concatenação XH:XL;

#cte – é uma constante de 8 bits;

k – é uma constante de 8 bits;

y4 – é uma constante de 4 bits.

**2. Programando o processador didático**

**Considerando o set de instruções apresentado, escreva programas utilizando a linguagem Assembly do processador didático apresentado que execute as seguintes tarefas:**

a) Leia o estado do bit 0 da porta de entrada e reproduza esse estado no bit 1 da porta de saída.

Programa:

* **IN r0**
* **SHR r0, r0**
* **OUT r0**
* **JMP 0**

Na memória de programa, cada instrução é representada por uma palavra binária de até 17 bits. A palavra na memória de programa pode ser representada da seguinte forma:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instrução | B16 | B15 | B14 | B13 | B12 | B11 | B10 | B9 | B8 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
| IN r0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SHR r0, r0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| JMP 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

b) Faça com que 8 LEDs conectados à porta de saída acendam sequencialmente, um de cada vez, desde o bit 0 até o bit 7, em um loop infinito.

Programa:

* **LDI r0, #0x01**
* **OUT r0**
* **SHR r0, r0**
* **BRZ r0, 0x02**
* **JMP 1**
* **JMP 0**

Na memória de programa, cada instrução é representada por uma palavra binária de até 17 bits. A palavra na memória de programa pode ser representada da seguinte forma:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instrução | B16 | B15 | B14 | B13 | B12 | B11 | B10 | B9 | B8 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
| LDI r0, #0x01 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| OUT r0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SHR r0, r0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| BRZ r0, #0x02 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| JMP 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| JMP 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

c) Faça com que os 4 bits menos significativos da porta de saída permaneçam setados, enquanto os 4 bits mais significativos permaneçam resetados enquanto o bit 3 da porta de entrada estiver setado. A saída deve funcionar de forma complementar enquanto o bit 3 da porta de entrada estiver desabilitado.

Programa:

* **LDI r3, #0x08**
* **LDI r2, #0xF0**
* **LDI r1, #0x0F**
* **IN r0**
* **AND r0, r3, r0**
* **BRNE r0, r3, 0x3**
* **OUT r1**
* **JMP 3**
* **OUT r2**
* **JMP 3**

Na memória de programa, cada instrução é representada por uma palavra binária de até 17 bits. A palavra na memória de programa pode ser representada da seguinte forma:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instrução | B16 | B15 | B14 | B13 | B12 | B11 | B10 | B9 | B8 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
| LDI r3, #0x08 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| LDI r2, #0xF0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| LDI r1, #0x0F | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| IN r0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| AND r0, r3, r0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| BRNE r0,r3, 0x3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| OUT r1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| JMP 3 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| OUT r2 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| JMP 3 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |

d) Considerando que os bits 0-3 da porta de entrada formem um dado X de 4 bits e os bits 4-7 formem um dado Y

também de 4 bits, fazer com que a porta de saída exiba, simultaneamente, a operação lógica bit a bit X+Y, nos 4

bits menos significativos e X.Y, nos 4 bits mais significativos.

Programa:

* **LDI r2, #0xF0**
* **LDI r1, #0x0F**
* **IN r0**
* **AND r3, r2, r0**
* **AND r4, r1, r0**
* **OR r5, r3, r4**
* **OUT r5**
* **AND r6, r3, r4**
* **OUT r6**
* **JMP 3**

Na memória de programa, cada instrução é representada por uma palavra binária de até 17 bits. A palavra na memória de programa pode ser representada da seguinte forma:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instrução | B16 | B15 | B14 | B13 | B12 | B11 | B10 | B9 | B8 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
| LDI r2, #0xF0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| LDI r1, #0x0F | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| IN r0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| AND r3, r2, r0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| AND r4, r1, r0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| OR r5, r3, r4 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| OUT r5 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| AND r6, r3, r4 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| OUT r6 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| JMP 3 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

e) Exiba permanentemente na porta de saída uma contagem decrescente de 150 a 0.

Programa:

* **LDI r0, #0x96**
* **BRZ r0, #0x03**
* **DEC r0, r0**
* **JMP 1**
* **JMP 0**

Na memória de programa, cada instrução é representada por uma palavra binária de até 17 bits. A palavra na memória de programa pode ser representada da seguinte forma:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instrução | B16 | B15 | B14 | B13 | B12 | B11 | B10 | B9 | B8 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
| LDI r0, #096 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| BRZ r0, #0x03 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| DEC r0, r0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| JMP 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| JMP 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

f) Exiba na porta de saída o triplo do valor da porta de entrada.

Programa:

* **IN r0**
* **MULI r0, #0x03**
* **OUT rx**
* **RESET**

Na memória de programa, cada instrução é representada por uma palavra binária de até 17 bits. A palavra na memória de programa pode ser representada da seguinte forma:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instrução | B16 | B15 | B14 | B13 | B12 | B11 | B10 | B9 | B8 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
| IN r0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| MULI r0, #0x03 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| OUT rx | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RESET | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

g) Habilite o bit 7 da porta de saída se o valor da porta de entrada for ímpar e desabilite o mesmo bit, caso seja par.

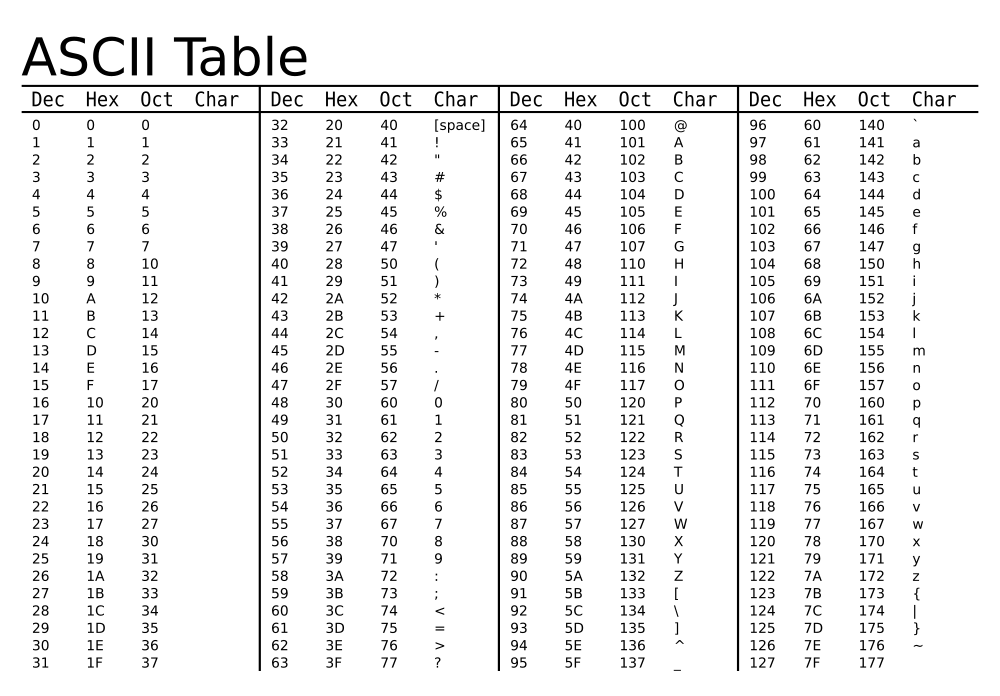
Programa:

* **LDI r1, #0x80**
* **IN r0**
* **ANDI r0, #0x01**
* **BRZ r0, #0x03**
* **OUT r1**
* **JMP 1**
* **OUT r0**
* **JMP 1**

Na memória de programa, cada instrução é representada por uma palavra binária de até 17 bits. A palavra na memória de programa pode ser representada da seguinte forma:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instrução | B16 | B15 | B14 | B13 | B12 | B11 | B10 | B9 | B8 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
| LDI r1, #0x80 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| IN r0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ANDI r0, #0x01 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| BRZ r0, #0x03 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| OUT r1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| JMP 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| OUT r0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| JMP 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

h) Considerando que a porta de saída seja uma porta de comunicação paralela, envie através desta o texto “Hello World!”, sem as aspas.



Programa:

* **LDI r0, #0x48**
* **LDI r1, #0x45**
* **LDI r2, #0x4C**
* **LDI r3, #0x4C**
* **LDI r4, #0x4F**
* **LDI r5, #0x20**
* **LDI r6, #0x57**
* **LDI r7, #0x4F**
* **LDI r8, #0x52**
* **LDI r9, #0x4C**
* **LDI r10, #0x44**
* **LDI r11, #0x21**
* **OUT r0**
* **OUT r1**
* **OUT r2**
* **OUT r3**
* **OUT r4**
* **OUT r5**
* **OUT r6**
* **OUT r7**
* **OUT r8**
* **OUT r9**
* **OUT r10**
* **OUT r11**
* **RESET**

Na memória de programa, cada instrução é representada por uma palavra binária de até 17 bits. A palavra na memória de programa pode ser representada da seguinte forma:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instrução | B16 | B15 | B14 | B13 | B12 | B11 | B10 | B9 | B8 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
| LDI r0, #0x48 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| LDI r1, #0x45 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| LDI r2, #0x4C | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| LDI r3, #0x4C | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| LDI r4, #0x4D | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| LDI r5, #0x20 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| LDI r6, #0x57 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| LDI r7, #0x4F | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| LDI r8, #0x52 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| LDI r9, #0x4C | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| LDI r10, #0x44 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| LDI r11, #0x21 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| OUT r0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r2 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r3 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r4 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r5 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r6 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r7 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r8 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r9 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r10 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUT r11 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RESET | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

i) Exiba na porta de saída a média aritmética dos 4 últimos valores lidos da porta de entrada.

Programa:

* **IN r0**
* **IN r1**
* **IN r2**
* **IN r3**
* **ADD r4, r0, r1**
* **ADD r5, r2, r3**
* **ADD r6, r5, r4**
* **SHL r6, r6**
* **SHL r6, r6**
* **OUT r6**
* **RESET**

Na memória de programa, cada instrução é representada por uma palavra binária de até 17 bits. A palavra na memória de programa pode ser representada da seguinte forma:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instrução | B16 | B15 | B14 | B13 | B12 | B11 | B10 | B9 | B8 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |