Instituto Politécnico Nacional

Escuela Superior de Cómputo

Práctica 9 - Pila Hardware 1

Unidad de aprendizaje: Arquitectura de Computadoras

Grupo: 3CV1

Alumno(a):
Ramos Diaz Enrique

Profesor(a):
Vega García Nayeli

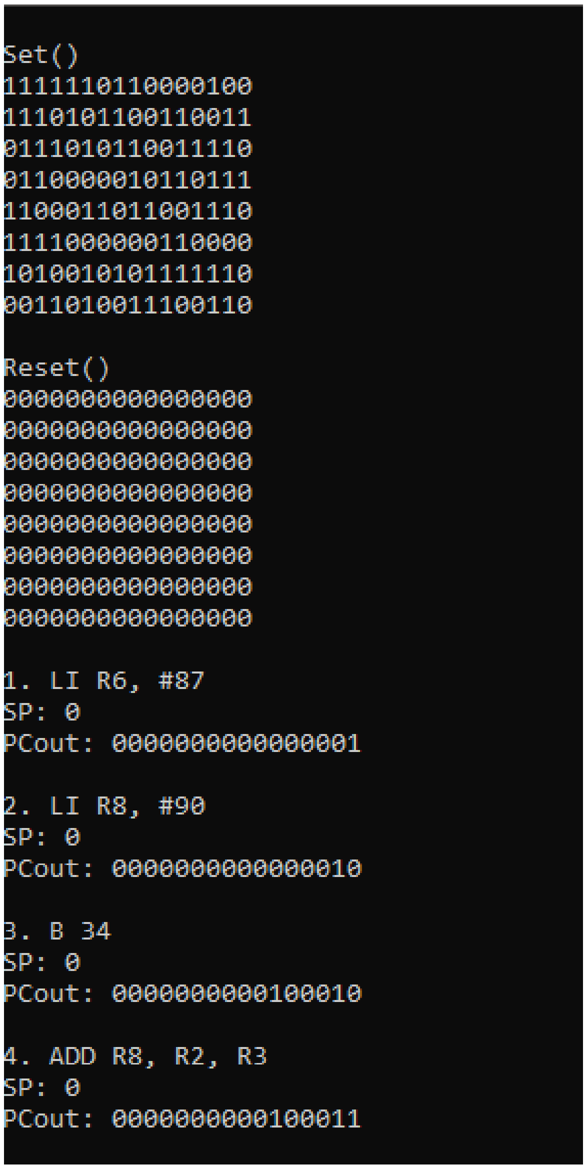
29 de abril 2020

1. Código de implementación

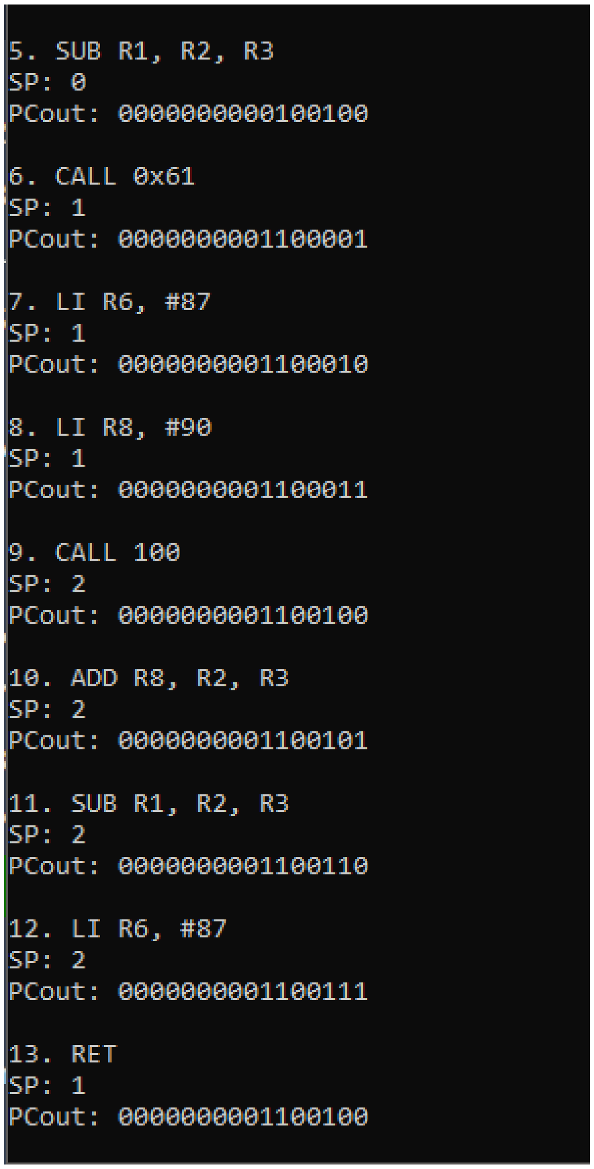
```
#include <time.h>
#include <bitset>
#include <math.h>
#include <iostream>
using namespace std;

class Pila {
   private:
      int PC[8];
      int stackPointer;
      int PCout;
   public:
      Pila();
      void set();
      void get();
      void operacion(int clr, int WPC, int UP, int DW, int PCin);
      void operacion();
};

Pila::Pila() { }

void Pila::set() {
   for(int i = 0; i < 8; i++)
      PC[i] = rand() % 65535;
   stackPointer = 0;
}

void Pila::get() {
    for (int i = 0; i < 8; i++)
       cout << bitset<16>(PC[i]) << endl;
}

void Pila::operacion() {
   PCout = PC[stackPointer];
   cout << "SP: " << stackPointer << endl;
   cout << "PCout: " << bitset<16>(PCout) << endl;
}

void Pila::operacion(int clr, int WPC, int UP, int DW, int PCin) {
   if (clr == 1) {
      // Reset
      stackPointer = 0;
      fill(PC, PC + 8, 0);
   }
   else if (clr == 0 && WPC == 0 && UP == 0 && DW == 0) {
      // Otras instrucciones
      stackPointer = stackPointer;
      PC[stackPointer]++;
   }
   else if (clr == 0 && WPC == 1 && UP == 0 && DW == 0) {
      // Saltos B, BNEI, BEQI, etc
      if (PCin > 65535)
         cout << "PC[SP] desbordado" << endl;
      else {
         stackPointer = stackPointer;
         PC[stackPointer] = PCin;
      }
   }
   else if (clr == 0 && WPC == 1 && UP == 1 && DW == 0) {
      // Llamadas subrutinas CALL
      if (PCin > 65535)
         cout << "PC[SP] desbordado" << endl;
      else {
         stackPointer++;
         if (stackPointer == 8)
            stackPointer = 0;
         PC[stackPointer] = PCin;
      }
   }
   else if (clr == 0 && WPC == 0 && UP == 0 && DW == 1) {
      // Retornos RET
      stackPointer--;
      if (stackPointer == -1)
         stackPointer = 7;
      PC[stackPointer]++;
   }
   else
      operacion(); // PCout
}

int main() {
   srand(time(NULL));
   Pila p;

   //Set
   cout << "\nSet()" << endl;
```

```
    p.set();
    p.get();

    //Reset
    cout << "\nReset()" << endl;
    p.operacion(1, 0, 0, 0, 0);
    p.get();

    cout << "\n1. LI R6, #87" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n2. LI R8, #90" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n3. B 34" << endl;
    p.operacion(0, 1, 0, 0, 34);
    p.operacion();

    cout << "\n4. ADD R8, R2, R3" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n5. SUB R1, R2, R3" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n6. CALL 0x61" << endl;
    p.operacion(0, 1, 1, 0, 0x61);
    p.operacion();

    cout << "\n7. LI R6, #87" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n8. LI R8, #90" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n9. CALL 100" << endl;
    p.operacion(0, 1, 1, 0, 100);
    p.operacion();

    cout << "\n10. ADD R8, R2, R3" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n11. SUB R1, R2, R3" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n12. LI R6, #87" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n13. RET" << endl;
    p.operacion(0, 0, 0, 1, 0);
    p.operacion();

    cout << "\n14. SUB R1, R2, R3" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n15. LI R6, #87" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n16. RET" << endl;
    p.operacion(0, 0, 0, 1, 0);
    p.operacion();

    cout << "\n17. B 300" << endl;
    p.operacion(0, 1, 0, 0, 300);
    p.operacion();

    cout << "\n18. CALL 889" << endl;
    p.operacion(0, 1, 1, 0, 889);
    p.operacion();

    cout << "\n19. ADD R8, R2, R3" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n20. SUB R1, R2, R3" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n21. LI R6, #87" << endl;
    p.operacion(0, 0, 0, 0, 0);
    p.operacion();

    cout << "\n22. RET" << endl;
    p.operacion(0, 0, 0, 1, 0);
    p.operacion();

    cout << "\n23. RET" << endl;
    p.operacion(0, 0, 0, 1, 0);
    p.operacion();

    cout << "\nGet()" << endl;
    p.get();

    cout << endl;
    return 0;
}
```

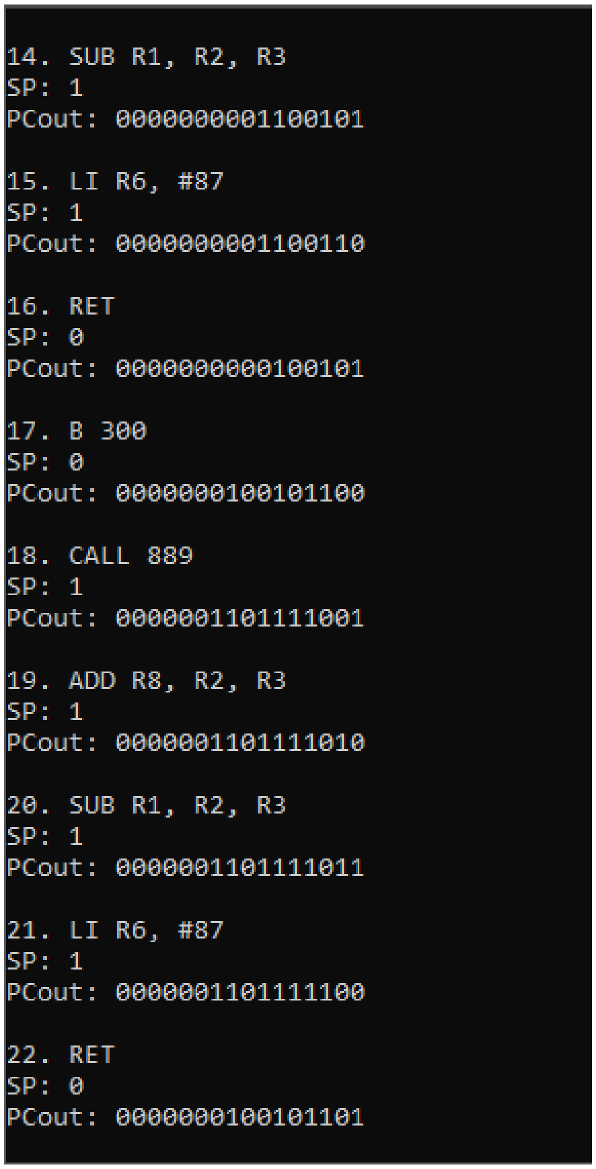
2. Pruebas



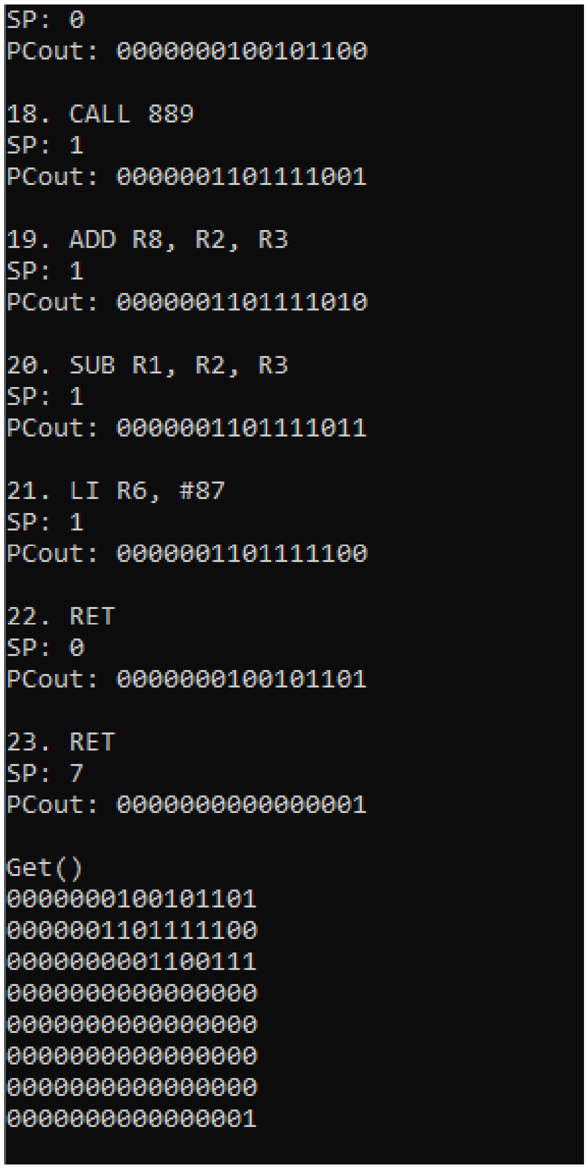
```
Set()
1111110110000100
1110101100110011
0111010110011110
0110000010110111
1100011011001110
1111000000110000
1010010101111110
0011010011100110

Reset()
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000

1. LI R6, #87
SP: 0
PCout: 0000000000000001

2. LI R8, #90
SP: 0
PCout: 0000000000000010

3. B 34
SP: 0
PCout: 0000000000100010

4. ADD R8, R2, R3
SP: 0
PCout: 0000000000100011
```



```
5. SUB R1, R2, R3
SP: 0
PCout: 0000000000100100

6. CALL 0x61
SP: 1
PCout: 0000000001100001

7. LI R6, #87
SP: 1
PCout: 0000000001100010

8. LI R8, #90
SP: 1
PCout: 0000000001100011

9. CALL 100
SP: 2
PCout: 0000000001100100

10. ADD R8, R2, R3
SP: 2
PCout: 0000000001100101

11. SUB R1, R2, R3
SP: 2
PCout: 0000000001100110

12. LI R6, #87
SP: 2
PCout: 0000000001100111

13. RET
SP: 1
PCout: 0000000001100100
```



```
14. SUB R1, R2, R3
SP: 1
PCout: 0000000001100101

15. LI R6, #87
SP: 1
PCout: 0000000001100110

16. RET
SP: 0
PCout: 0000000000100101

17. B 300
SP: 0
PCout: 0000000100101100

18. CALL 889
SP: 1
PCout: 0000001101111001

19. ADD R8, R2, R3
SP: 1
PCout: 0000001101111010

20. SUB R1, R2, R3
SP: 1
PCout: 0000001101111011

21. LI R6, #87
SP: 1
PCout: 0000001101111100

22. RET
SP: 0
PCout: 0000000100101101
```



```
SP: 0
PCout: 0000000100101100

18. CALL 889
SP: 1
PCout: 0000001101111001

19. ADD R8, R2, R3
SP: 1
PCout: 0000001101111010

20. SUB R1, R2, R3
SP: 1
PCout: 0000001101111011

21. LI R6, #87
SP: 1
PCout: 0000001101111100

22. RET
SP: 0
PCout: 0000000100101101

23. RET
SP: 7
PCout: 0000000000000001

Get()
0000000100101101
0000001101111100
0000000001100111
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000001
```