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;
```

```
43          fill(PC, PC + 8, 0);
44       }
45       else if (clr == 0 && WPC == 0 && UP == 0 && DW == 0) {
46          // Otras instrucciones
47          stackPointer = stackPointer;
48          PC[stackPointer]++;
49       }
50       else if (clr == 0 && WPC == 1 && UP == 0 && DW == 0) {
51          // Saltos B, BNEI, BEQI, etc
52          if (PCin > 65535)
53             cout << "PC[SP] desbordado" << endl;
54          else {
55             stackPointer = stackPointer;
56             PC[stackPointer] = PCin;
57          }
58       }
59       else if (clr == 0 && WPC == 1 && UP == 1 && DW == 0) {
60          // Llamadas subrutinas CALL
61          if (PCin > 65535)
62             cout << "PC[SP] desbordado" << endl;
63          else {
64             stackPointer++;
65             if (stackPointer == 8)
66                stackPointer = 0;
67             PC[stackPointer] = PCin;
68          }
69       }
70       else if (clr == 0 && WPC == 0 && UP == 0 && DW == 1) {
71          // Retornos RET
72          stackPointer--;
73          if (stackPointer == -1)
74             stackPointer = 7;
75          PC[stackPointer]++;
76       }
77       else
78          operacion(); // PCout
79    }
80
81    int main() {
82       srand(time(NULL));
83       Pila p;
84
85       //Set
86       cout << "\nSet()" << endl;
```

```
87      p.set();
88      p.get();
89
90      //Reset
91      cout << "\nReset()" << endl;
92      p.operacion(1, 0, 0, 0, 0);
93      p.get();
94
95      cout << "\n1. LI R6, #87" << endl;
96      p.operacion(0, 0, 0, 0, 0);
97      p.operacion();
98
99      cout << "\n2. LI R8, #90" << endl;
100     p.operacion(0, 0, 0, 0, 0);
101     p.operacion();
102
103     cout << "\n3. B 34" << endl;
104     p.operacion(0, 1, 0, 0, 34);
105     p.operacion();
106
107     cout << "\n4. ADD R8, R2, R3" << endl;
108     p.operacion(0, 0, 0, 0, 0);
109     p.operacion();
110
111     cout << "\n5. SUB R1, R2, R3" << endl;
112     p.operacion(0, 0, 0, 0, 0);
113     p.operacion();
114
115     cout << "\n6. CALL 0x61" << endl;
116     p.operacion(0, 1, 1, 0, 0x61);
117     p.operacion();
118
119     cout << "\n7. LI R6, #87" << endl;
120     p.operacion(0, 0, 0, 0, 0);
121     p.operacion();
122
123     cout << "\n8. LI R8, #90" << endl;
124     p.operacion(0, 0, 0, 0, 0);
125     p.operacion();
126
127     cout << "\n9. CALL 100" << endl;
128     p.operacion(0, 1, 1, 0, 100);
129     p.operacion();
130
```

```
131     cout << "\n10. ADD R8, R2, R3" << endl;
132     p.operacion(0, 0, 0, 0, 0);
133     p.operacion();
134 
135     cout << "\n11. SUB R1, R2, R3" << endl;
136     p.operacion(0, 0, 0, 0, 0);
137     p.operacion();
138 
139     cout << "\n12. LI R6, #87" << endl;
140     p.operacion(0, 0, 0, 0, 0);
141     p.operacion();
142 
143     cout << "\n13. RET" << endl;
144     p.operacion(0, 0, 0, 1, 0);
145     p.operacion();
146 
147     cout << "\n14. SUB R1, R2, R3" << endl;
148     p.operacion(0, 0, 0, 0, 0);
149     p.operacion();
150 
151     cout << "\n15. LI R6, #87" << endl;
152     p.operacion(0, 0, 0, 0, 0);
153     p.operacion();
154 
155     cout << "\n16. RET" << endl;
156     p.operacion(0, 0, 0, 1, 0);
157     p.operacion();
158 
159     cout << "\n17. B 300" << endl;
160     p.operacion(0, 1, 0, 0, 300);
161     p.operacion();
162 
163     cout << "\n18. CALL 889" << endl;
164     p.operacion(0, 1, 1, 0, 889);
165     p.operacion();
166 
167     cout << "\n19. ADD R8, R2, R3" << endl;
168     p.operacion(0, 0, 0, 0, 0);
169     p.operacion();
170 
171     cout << "\n20. SUB R1, R2, R3" << endl;
172     p.operacion(0, 0, 0, 0, 0);
173     p.operacion();
174 
```

```
175     cout << "\n21. LI R6, #87" << endl;
176     p.operacion(0, 0, 0, 0, 0);
177     p.operacion();
178 
179     cout << "\n22. RET" << endl;
180     p.operacion(0, 0, 0, 1, 0);
181     p.operacion();
182 
183     cout << "\n23. RET" << endl;
184     p.operacion(0, 0, 0, 1, 0);
185     p.operacion();
186 
187     cout << "\nGet()" << endl;
188     p.get();
189 
190     cout << endl;
191     return 0;
192 }
```

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
```