1. Instruction at 0x300

- **Instruction:** `PUSH 0x800`
- **Registers/Memory Locations Modified:**
  - Stack pointer (SP) moves from 0x118 to 0x114.
  - Value `0x800` is pushed onto the stack at address `0x114`.
- **New Contents:**
  - SP = 0x114
  - Memory location `0x114` = 0x800
- **Program Counter (PC) After Instruction:** 0x304

2. Instruction at 0x304

- **Instruction:** `PUSH *(0x804)`
- **Registers/Memory Locations Modified:**
  - Stack pointer (SP) moves from 0x114 to 0x110.
  - Value `0x200` (from `*0x804`) is pushed onto the stack at address `0x110`.
- **New Contents:**
  - SP = 0x110
  - Memory location `0x110` = 0x200
- **Program Counter (PC) After Instruction:** 0x308

3. Instruction at 0x308

- **Instruction:** `CALL 0x400 // do_switch(sp2, &sp1)`
- **Registers/Memory Locations Modified:**
  - Stack pointer (SP) moves from 0x110 to 0x10C.
  - Return address `0x30C` is pushed onto the stack at address `0x10C`.
  - Program counter (PC) jumps to 0x400.
- **New Contents:**
  - SP = 0x10C
  - Memory location `0x10C` = 0x30C
  - PC jumps to `0x400`

4. Instruction at 0x400

- **Instruction:** `MOV *(SP+8) -> EAX // EAX = oldsp_ptr`
- **Registers/Memory Locations Modified:**
  - EAX is loaded with the value at address `SP+8`, which is `0x804` (the address of `sp2`).
- **New Contents:**
  - EAX = 0x804
- **Program Counter (PC) After Instruction:** 0x404

5. Instruction at 0x404

- **Instruction:** `MOV SP -> *EAX // *oldsp_ptr = SP`
- **Registers/Memory Locations Modified:**
  - The value of SP (`0x10C`) is written to the memory location at `EAX` (which is `0x804`), so `sp2` is updated.
- **New Contents:**
  - Memory location `0x804` = 0x10C (the value of SP)
- **Program Counter (PC) After Instruction:** 0x408

6. Instruction at 0x408

- **Instruction:** `MOV *(SP+4) -> EAX // EAX = newsp`
- **Registers/Memory Locations Modified:**
  - EAX is loaded with the value at `SP+4`, which is `0x800` (the value of `sp1`).
- **New Contents:**
  - EAX = 0x800
- **Program Counter (PC) After Instruction:** 0x40C

7. Instruction at 0x40C

- **Instruction:** `MOV EAX -> SP // SP = newsp`
- **Registers/Memory Locations Modified:**
  - Stack pointer (SP) is updated to the value of `EAX`, which is `0x800` (the value of `sp1`).
- **New Contents:**
  - SP = 0x800
- **Program Counter (PC) After Instruction:** 0x410

8. Instruction at 0x410

- **Instruction:** `RET`
- **Registers/Memory Locations Modified:**
  - Stack pointer (SP) is incremented, and the value at address `SP` (which is `0x30C`) is popped into the program counter (PC).
  - PC jumps back to `0x30C`.
- **New Contents:**
  - SP = 0x804
  - PC = 0x30C

9. Instruction at 0x30C

- **Instruction:** `ADD 8 -> SP`
- **Registers/Memory Locations Modified:**
  - Stack pointer (SP) is incremented by 8.
- **New Contents:**
  - SP = 0x80C
- **Program Counter (PC) After Instruction:** 0x500

10. Instruction at 0x500

- **Instruction:** `POP EAX // EAX = stack.pop()`
- **Registers/Memory Locations Modified:**
  - Stack pointer (SP) is incremented and the value at `SP` (`0x500`) is popped into `EAX`.
- **New Contents:**
  - EAX = 500
  - SP = 0x808
- **Program Counter (PC) After Instruction:** 0x504

11. Instruction at 0x504

- **Instruction:** `POP EBX // EBX = stack.pop()`
- **Registers/Memory Locations Modified:**
  - Stack pointer (SP) is incremented and the value at `SP` (`0x7`) is popped into `EBX`.
- **New Contents:**
  - EBX = 7
  - SP = 0x80C
- **Program Counter (PC) After Instruction:** 0x508