

Pop15_0

Using the indirect addressing mode instruction `ld` and `st`, pop registers `r15` to `r0` from a stack buffer starting at SRAM address `0x4FF`. To test your subroutine, begin by adding and then calling subroutines `PreReg0_20`, `Push0_15`, `Clr0_15`, `Pop15_0` in the setup section of your code.

Name	Value
Program Counter	0x00004
Stack Pointer	0x08FF
X pointer	0x0010
Y pointer	0x04F0
Z pointer	0x0000
Cycle Counter	237
Frequency	1.0000 MHz
Stop Watch	237.00 us
SREG	00000000

```

* write a subroutine to restore registers
* stack buffer located at SRAM 0x04FF
* Hint: SRAM address 0x0000 to 0x0001F
*/

.INCLUDE <m328pdef.inc>

Setup:
    ldi YH,0x05 ; YH = r29, initialize
    ldi YL,0x00 ; YL = r28, pre-decrement

    rcall PreReg0_20
    rcall SaveReg0_15
    rcall Clr0_15
    rcall RestoreReg15_0

done:
    rjmp done

RestoreReg15_0:
    clr XH ; point X at register r
    ldi XL, 16 ; after pre-decrement (
    ldi r24, 16 ; loop 16 times (r15...
pop_loop:
    ld r25, Y+
    st -X, r25
    dec r24
    brne pop_loop
    ret
  
```

Address	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	00	00	00
0004F0	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	00	00	00
000506	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Figure 1 Registers `r0` to `r20` saved to stack buffer starting at location `0x04FF`.

Name	Value
Program Counter	0x00005
Stack Pointer	0x08FF
X pointer	0x0010
Y pointer	0x04F0
Z pointer	0x0000
Cycle Counter	327
Frequency	1.0000 MHz
Stop Watch	327.00 us
SREG	00000000

```

* write a subroutine to restore registers
* stack buffer located at SRAM 0x04FF
* Hint: SRAM address 0x0000 to 0x0001F
*/

.INCLUDE <m328pdef.inc>

Setup:
    ldi YH,0x05 ; YH = r29, initialize
    ldi YL,0x00 ; YL = r28, pre-decrement

    rcall PreReg0_20
    rcall SaveReg0_15
    rcall Clr0_15
    rcall RestoreReg15_0

done:
    rjmp done

RestoreReg15_0:
    clr XH ; point X at register r
    ldi XL, 16 ; after pre-decrement (
    ldi r24, 16 ; loop 16 times (r15...
pop_loop:
    ld r25, Y+
    st -X, r25
    dec r24
    brne pop_loop
    ret
  
```

Address	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	00	00	00
0004F0	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	00	00	00
000506	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Figure 2 Registers `r0` to `r20` cleared.

Name	Value
Program Counter	0x00006
Stack Pointer	0x08FF
X pointer	0x0000
Y pointer	0x0500
Z pointer	0x0000
Cycle Counter	448
Frequency	1.0000 MHz
Stop Watch	448.00 us
SREG	00000000
Registers	
R00	0x15
R01	0x14
R02	0x13
R03	0x12
R04	0x11
R05	0x10
R06	0x0F
R07	0x0E
R08	0x0D
R09	0x0C
R10	0x0B
R11	0x0A
R12	0x09
R13	0x08
R14	0x07
R15	0x06
R16	0x05

```

* write a subroutine to restore registers
* stack buffer located at SRAM 0x04FF
* Hint: SRAM address 0x0000 to 0x0001F
*/

#include <m328pdef.inc>

Setup:
    ldi YH,0x05 ; YH = r29, initialize
    ldi YL,0x00 ; YL = r28, pre-decrement

    rcall PreReg0_20
    rcall SaveReg0_15
    rcall Clr0_15
    rcall RestoreReg15_0

done:
    rjmp done

RestoreReg15_0:
    clr XH ; point X at register r24
    ldi XL,16 ; after pre-decrement (16)
    ldi r24,16 ; loop 16 times (r15...r0)
pop_loop:
    ld r25,Y+
    st -X,r25
    dec r24
    brne pop_loop
    ret
    
```

Memory																				
Data	8/16			abc	Address: 0x4F0	Cols: Auto														
0004F0	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	00	00	00	00
000506	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Figure 3 Registers r0 to r20 restored to saved values.