Stack & Subroutine in AVR

Stack

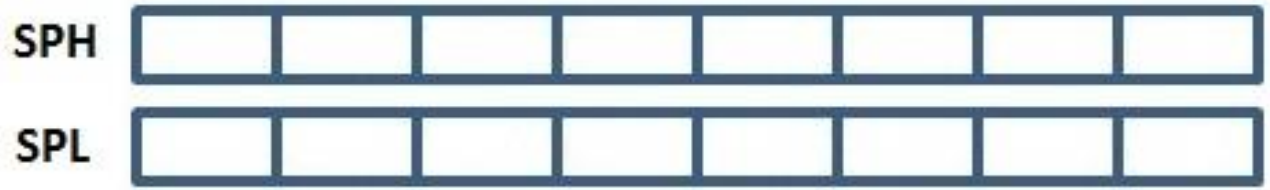
- Stack adalah blok berurutan yang terletak pada Data Memory yang disediakan oleh programmer.
- Operasi stack menggunakan mekanisme **Last In First Out (LIFO)**.
- Digunakan sebagai media penyimpanan oleh *microcontroller internal control* ataupun media penyimpanan sementara.

Stack (cont.)

- Sebelum digunakan, harus ditetapkan dulu nilai Stack pointer-nya.
- Stack pointer adalah suatu register I/O pada AVR yang berisi posisi pointer stack pada Data Memory.
- Stack pointer biasanya diisi dengan alamat terakhir dari Data Memory.

Stack Pointer

- Stack Pointer memiliki panjang 16 bit, dibagi menjadi 8 bit SPL dan 8 bit SPH.



Stack Pointer (cont.)

Program Counter	0x000000
Stack Pointer	0x0000
X pointer	0x0000
Y pointer	0x0000
Z pointer	0x0000
Cycle Counter	0
Frequency	4.0000 MHz
Stop Watch	0.00 us
SREG	I T H S V N Z C

Awalnya bernilai nol

Inisialisasi nilai SPL dan SPH

```
ldi  temp,low(RAMEND)
out  SPL,temp      ;init Stack Pointer
ldi  temp,high(RAMEND)
out  SPH,temp
```

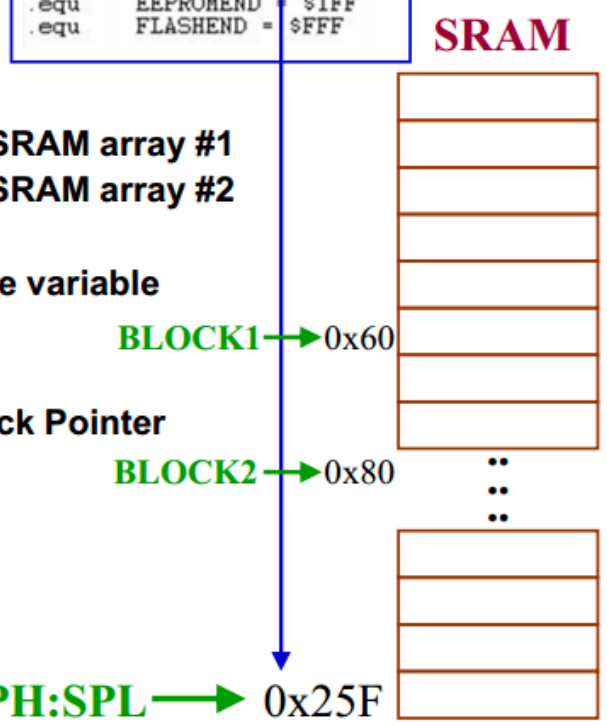
Program Counter	0x000010
Stack Pointer	0x025F
X pointer	0x0000
Y pointer	0x0000
Z pointer	0x0000
Cycle Counter	6

Berisi alamat terakhir dari Data Memory

Init Stack Pointer

```
.include "m8515def.inc"

  rjmp   RESET        ;reset handle
  ..........

.equ BLOCK1 =$60 ;start address of SRAM array #1
.equ BLOCK2 =$80 ;start address of SRAM array #2

.def temp = r16      ;temporary storage variable

RESET:
  ldi     temp,low(RAMEND)  ;init Stack Pointer
  out     SPL,temp
  ldi     temp,high(RAMEND)
  out     SPH,temp
```

```
.def    XL  =r26
.def    XH  =r27
.def    YL  =r28
.def    YH  =r29
.def    ZL  =r30
.def    ZH  =r31

.equ    RAMEND =$25F
.equ    EEPROMEND = $1FF
.equ    FLASHEND = $FFF
```

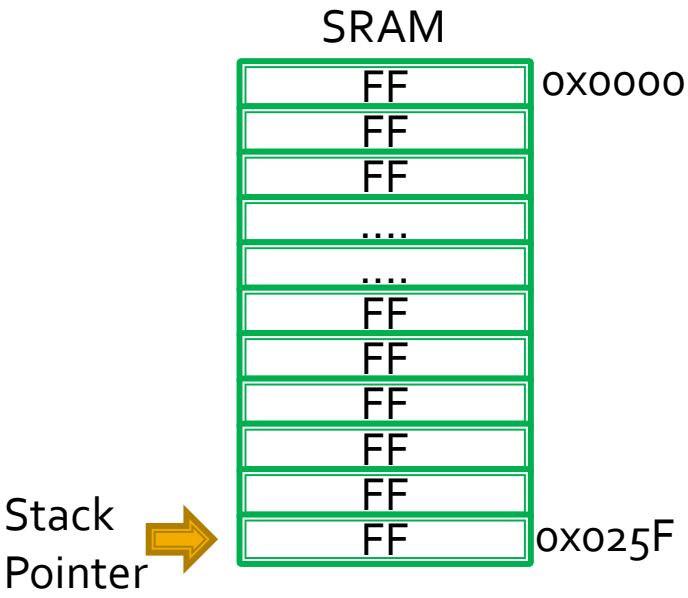


Operasi Stack

- Menggunakan perintah PUSH [Register] untuk memasukkan data.
  - Contoh: PUSH R1
  - Artinya: Memasukan nilai R1 pada stack dan mengubah SP menjadi SP -1
- Menggunakan perintah POP [Register] untuk mengambil data.
  - Contoh: POP R1
  - Artinya: Mengubah SP menjadi SP +1, mengambil nilai pada stack, lalu nilainya dimasukkan ke R1.

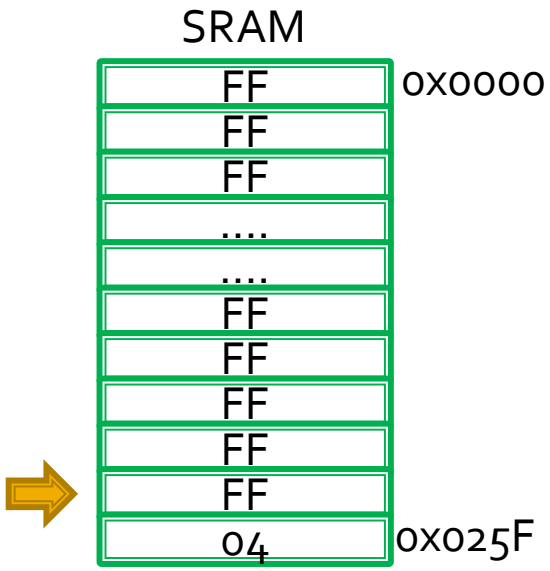
Operasi Stack (cont.)

```
LDI   R16, 04
MOV   R1,R16
PUSH  R1
....
```



Operasi Stack (cont.)

```
LDI   R16, 04
MOV   R1,R16
PUSH  R1
....
```

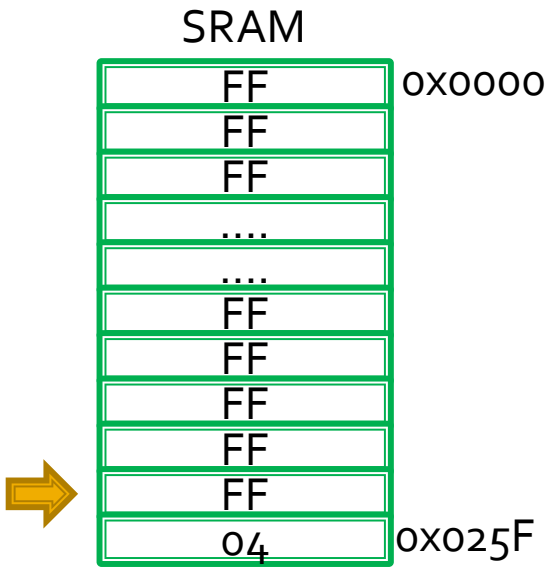


Operasi Stack (cont.)

....

POP R17

....



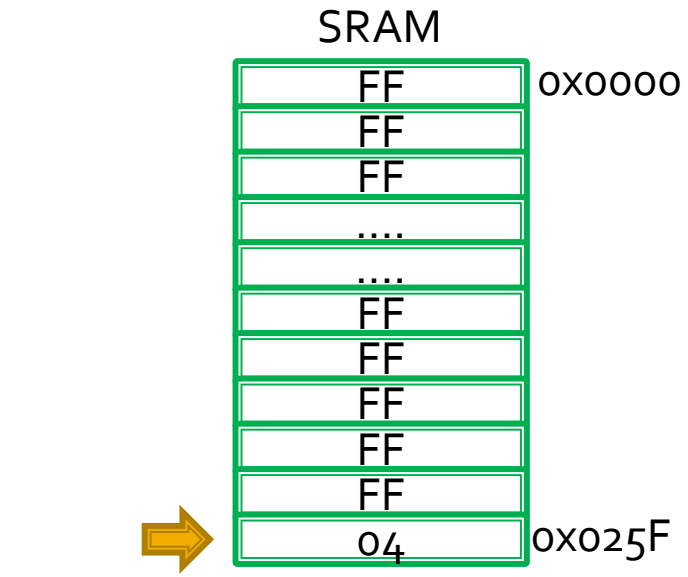
Operasi Stack (cont.)

....

POP R17

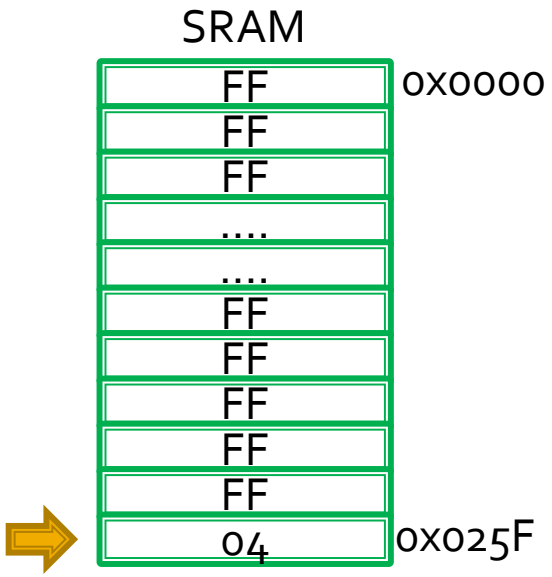
....

R17 ← 04



Operasi Stack (cont.)

```
    LDI     R16, 07
    PUSH    R16
    ....
```

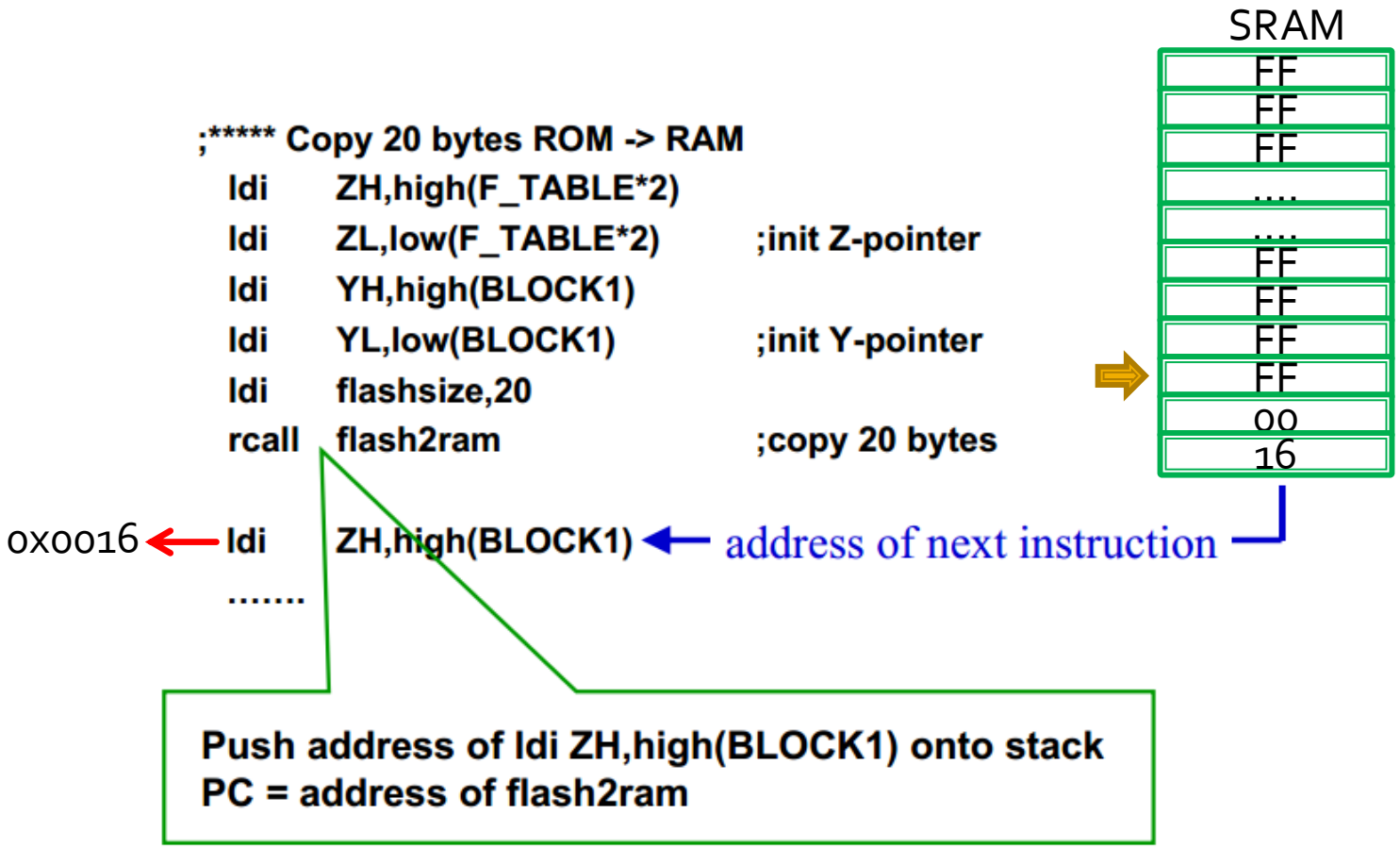


Apa yang terjadi?

Subroutine

- Mirip dengan *subroutine* di MIPS
- Pemanggilan menggunakan perintah RCALL (Relative Call).
  - RCALL [Label]
  - Ketika pemanggilan, alamat untuk kembali akan disimpan di Stack.

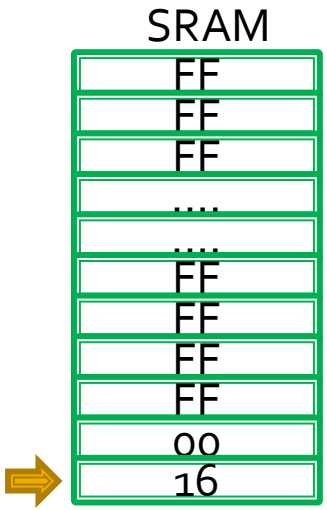
Pemanggilan Subroutine



Pemanggilan Subroutine (cont.)

```
;***** Subroutine Register variables
.def    flashsize=r16   ;size of block to be copied

flash2ram:
  lpm                   ;get constant
  st    Y+,r0           ;store in SRAM and increment Y-pointer
  adiw  ZL,1            ;increment Z-pointer
  dec   flashsize
  brne  flash2ram       ;if not end of table, loop more
  ret
```

PC = Pop(stack)
- Copy the value pointed by TOS to PC
- Increment TOS



PC ← 0x0016