```
            section .data

L1:         db      'BLOCK1      :',0x0a    ;Labels
L2:         db      'BLOCK2      :',0x0a
L3:         db      'SETUP BLOCKS ',0x0a
L4:         db      'SWAP WITH LONG REG ADDRESSING           ',0x0a
L5:         db      'SWAP WITH LONG REG+OFF ADDRESSING       ',0x0a
L6:         db      'SWAP WITH 32 BIT REG ADDRESSING         ',0x0a
L7:         db      'SWAP WITH 32 BIT REG+OFF ADDRESSING     ',0x0a

OUTPUT:     db      '            '          ;
HEX:        db      '0123456789ABCDEF'      ;hex table
LF:         db      0x0a                    ;line feeds
LF2:        db      0x0a,0x0a

            section .bss
BLOCK1:     resb 32                         ;Data Block tables
BLOCK2:     resb 32


            section .text
            global  main                    ;Tell linker about main
            extern  write, exit
main:
            mov     rbp, rsp                ; for correct debugging
            push    rbp
            mov     rbp, rsp

            lea     rsi,[L3]
            call    MYWRITE
            call    SETUP
            call    DISPLAY

            call    COPY1
            call    DISPLAY

            call    COPY2
            call    DISPLAY

            call    COPY3
            call    DISPLAY

            call    COPY4
            call    DISPLAY


MX:         xor     edi, edi                ; 0 return = success
            call    exit



; Copy with Long Register Addressing...
COPY1:      lea rsi,[L4]                    ;Write Label
            call MYWRITE2
            lea rsi,[BLOCK1]                ;point to blocks
            lea rdi,[BLOCK2]
            mov rcx,32                      ;setup loop counter
CP1:
            mov al,[rsi]                    ;get bytes
            mov ah,[rdi]
            mov [rsi],ah                    ;swap bytes
            mov [rdi],al
            inc rsi                         ;inc pointers
            inc rdi
            loop CP1                        ;loop till done
            ret
```

```
; Copy with Long Register + Offset Addressing...
COPY2:      lea rsi,[L5]                    ;Write Label
            call MYWRITE2
            lea rdx,[BLOCK1]                ;point to blocks
            lea rbx,[BLOCK2]
            mov rcx,31                      ;Load counter
CP2:
            mov al,[rdx+rcx]                ;get bytes
            mov ah,[rbx+rcx]
            mov [rdx+rcx],ah                ;swap bytes
            mov [rbx+rcx],al
            loop CP2                        ;loop till done
            mov al,[rdx+rcx]                ;swap last byte
            mov ah,[rbx+rcx]
            mov [rdx+rcx],ah
            mov [rbx+rcx],al
            ret



; Copy with 32bit Register Addressing...
COPY3:      lea rsi,[L6]                    ;Write Label
            call MYWRITE2
            lea rsi,[BLOCK1]                ;point to blocks
            lea rdi,[BLOCK2]
            mov rcx,32                      ;load counter
CP3:
            mov al,[rsi]                    ;get bytes
            mov bl,[rdi]
            mov [rsi],bl                    ;swap bytes
            mov [rdi],al
            inc rsi                         ;inc pointers
            inc rdi
            loop CP3                        ;loop till done
            ret


; Copy with 32bit Register + Offset Addressing...
COPY4:      lea rsi,[L7]                    ;Write label
            call MYWRITE2
            lea edx,[BLOCK1]                ;point to blocks
            lea ebx,[BLOCK2]
            mov ecx,31                      ;load counter
CP4:
            mov al,[edx+ecx]                ;get bytes
            mov ah,[ebx+ecx]
            mov [edx+ecx],ah                ;swap bytes
            mov [ebx+ecx],al
            loop CP4                        ;loop till done
            mov al,[edx+ecx]                ;swap last byte
            mov ah,[ebx+ecx]
            mov [edx+ecx],ah
            mov [ebx+ecx],al
            ret




SETUP:      lea rsi,[BLOCK1]                ;point to first block
            mov rcx,32                      ;setup counter
s1:
            mov rax,rsi                     ;get address in rax
            and RAX,0xFF                    ;only want low byte
            xor rax,0xFF                    ; ones compliment
            mov [rsi],al                    ;store number
            inc rsi                         ;next number
            loop s1                         ;loop till done

            lea rsi,[BLOCK2]                ;point to first block
            mov rcx,32                      ;setup counter
s2:
            mov rax,0                       ;set to zero
            mov [rsi],al                    ;store number
            inc rsi                         ;next number
            loop s2                         ;loop till done

            ret



DISPLAY:    lea rsi,[L1]
            call MYWRITE
            lea rdi,[BLOCK1]                ;point to block1
            mov rbx,32
D1:         mov rax,[rdi]                   ; get value
            and rax,0xFF                    ; keep clean
            call  TOHEX
            push qword rdi
            lea rsi,[OUTPUT]
            mov     edx, 3                  ; write hex value
            mov     edi, 1
            call    write
            pop  qword rdi
            inc rdi
            dec rbx
            jnz D1

            lea rsi,[LF]
            mov     edx, 1                  ; write hex value
            mov     edi, 1
            call    write

            lea rsi,[L2]
            call MYWRITE
            lea rdi,[BLOCK2]                ;point to block1
            mov rbx,32
D2:         mov rax,[rdi]                   ; get value
            and rax,0xFF                    ; keep clean
            call  TOHEX
            push qword rdi
            lea rsi,[OUTPUT]
            mov     edx, 3                  ; write hex value
            mov     edi, 1
            call    write
            pop  qword rdi
            inc rdi
            dec rbx
            jnz D2
            lea rsi,[LF2]
            mov     edx, 2                  ; write hex value
            mov     edi, 1
            call    write
            ret




; Usage: Load RSI with label
MYWRITE:
            mov     edx, 14                 ; write label
            mov     edi, 1
            call    write
            ret


; Usage: Load RSI with label
MYWRITE2:
            mov     edx, 40                 ; write label
            mov     edi, 1
            call    write
            ret


; Usage: Load rax with value
TOHEX:
            push qword rbx
            mov rbx,rax
            lea edx,[OUTPUT+1]              ;point to end of output string needed
            mov rcx,2

TH1:        mov rax,rbx                     ;loop start
            and rax,0xF                     ;and to get lowest byte value...
            mov al,[HEX+eax]
            mov [edx],al                    ;store number...
            shr rbx,4                       ;shift working value right for next byte
            dec edx
            loop TH1
            mov eax,0x20
            mov [OUTPUT+3],al
            pop qword rbx
            ret
```