Computer Science CSCI 261

Computer Architecture and Assembly Language

Dr. Peter Walsh
Department of Computer Science
Vancouver Island University
peter.walsh@viu.ca

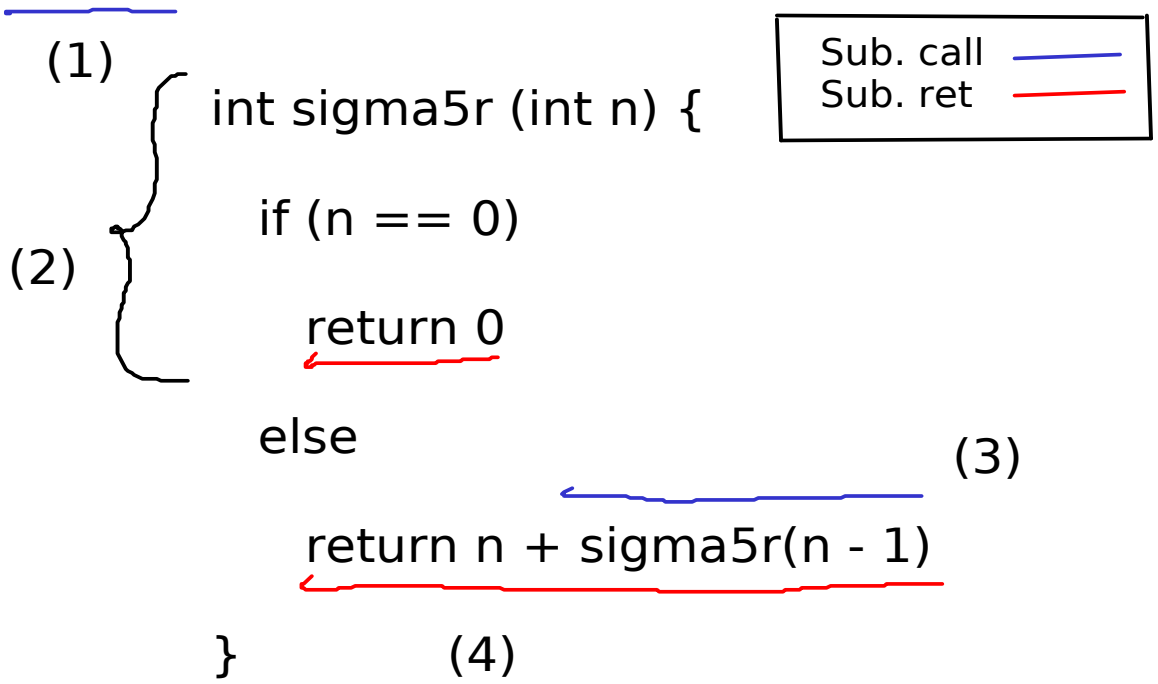
SSBC Recursion

◯ Σ 5 Recursive Components

- (1) main: subroutine call (calculate $\Sigma\ n$))
- (2) subroutine: base case (return 0)
- (3) subroutine: recursive call (calculate $\Sigma(n-1)$)
- (4) subroutine: return (return $n + \Sigma\ (n-1)$)

SSBC Recursion cont.

- Σ 5 Recursive Pseudocode



SSBC Recursion Component (1)

- Σ 5 Recursive (Assembly Language)

```
        pushimm 0          -- clear Z and N
        popext PSW
        pushimm L(YY)      -- save L(YY) on stack
        pushimm H(YY)      -- save H(YY) on stack
        pushimm 5          -- save parameter n
                           -- on the stack
        jnz sigma5r        -- jump to subroutine,
                           -- return address on
                           -- the stack
YY:     popext A           -- write returned value
                           -- of sigma(n) to port A
        halt
```

SSBC Recursion Component (1) cont.

- Σ 5 Recursive (Machine Language)

```
00000010 pushimm 0          Sigma 5 Recursive Example (see tinpSrc/sigma5r)
00000000
00000101 popext PSW
11111111
11111011
00000010 pushimm L(YY)
00001110                    save L(YY)
00000010 pushimm H(YY)
00000000                    save H(YY)
00000010 pushimm 5
00000101                    save parameter
00000110 jnz sigmar5
00000000
00010011                    call sigma5r
00000101 YY: popext A
11111111
11111100                    write ret. parameter to A
00000001 halt
00000000 ------------------subroutine --------------------------------
00000010 sigma5r: pushimm 0
```

SSBC Recursion Component (2)

- Σ 5 Recursive (Assembly Language)

```
sigma5r:pushimm 0
        add
        jnz reccall
        popext PSW                  -- clear Z and N
        -- self modifying code to take care or
        -- subroutine return
        popext sigma5r + 12h  -- save high byte
                                    -- of return address
        popext sigma5r + 13h  -- save low byte
                                    -- of return address
        pushimm 0                    -- place return
                                     -- value on
                                     -- the stack
        jnz                         -- return
```

SSBC Recursion Component (2) cont.

- Σ 5 Recursive (Machine Language)

```
00000010 sigma5r: pushimm 0
00000000
00001000 add                    activate Z flag
00000110 jnz reccall             recursive call if not base case
00000000
00100111
00000101 popext PSW
11111111
11111011                        clear Z
00000101 popext sigma5r + 12h
00000000
00100101
00000101 popext sigma5r + 13h
00000000
00100110                        load return address
00000010 pushimm 0
00000000                        load return parameter
00000110 jnz
00000000                        return address filled
00000000                        in at run time
00000101 reccall: popext hold
```

SSBC Recursion Component (3)

- Σ 5 Recursive (Assembly Language)

```
reccall:popext hold      -- save parameter
                         -- n in hold
        pushext hold      -- restore n to the stack
        pushimm L(recret) -- save L(recret) on stack
        pushimm H(recret) -- save H(recret) on stack
        pushimm 1         -- place n-1 on the stack
        pushext hold
        sub
        pushimm 0         -- clear Z and N
        popext PSW
        jnz sigma5r       -- recursive call to
                          -- sigma5r
```

SSBC Recursion Component (3) cont.

- Σ 5 Recursive (Machine Language)

```
00000101 reccall: popext hold
00000001
00000000                        save parameter in hold
00000011 pushext hold
00000001
00000000
00000010 pushimm L(recret)
00111111
00000010 pushimm H(recret)
00000000                        load return address
00000010 pushimm 1
00000001
00000011 pushext hold
00000001
00000000
00001001 sub                     sub 1 from parameter
00000010 pushimm 0
00000000
00000101 popext PSW
11111111
11111011
00000110 jnz sigma5r              recursive call
00000000
00010011
00001000 recret: add              add sigma n and sigma n-1
```

27 oX

3F oX

SSBC Recursion Component (4)

- Σ 5 Recursive (Assembly Language)

```
recret:add              -- add n and sigm(n-1)
     popext hold
     pushimm 0
     popext PSW                 -- clear Z and N
     -- self modifying code to take care
     -- of subroutine return
     popext sigma5r + 3fh
     -- save high byte of return address
     popext sigma5r + 40h
     -- save low byte of return address
     pushext hold    -- place sigma(n)
                     -- on the stack
     jnz             -- return
```

SSBC Recursion Component (4) cont.

- Σ 5 Recursive (Machine Language)

```
00001000 recret: add                add  n and sigma n-1
00000101 popext hold
00000001
00000000
00000010 pushimm 0
00000000
00000101 popext PSW
11111111
11111011
00000101 popext sigma5r + 3fh
00000000
01010010
00000101 popext sigma5r + 40h
00000000
01010011
00000011 pushext hold               load return parameter
00000001
00000000
00000110 jnz
00000000                            return address
00000000                            in at runtime
```