CALL <u>INSTRUCTIO</u>NS

Call instruction is used to call subroutine

- Subroutines are often used to perform tasks that need to be performed frequently
- This makes a program more structured in addition to saving memory space

LCALL (long call)

- > 3-byte instruction
 - First byte is the opcode
 - Second and third bytes are used for address of target subroutine
 - Subroutine is located anywhere within 64K byte address space

ACALL (absolute call)

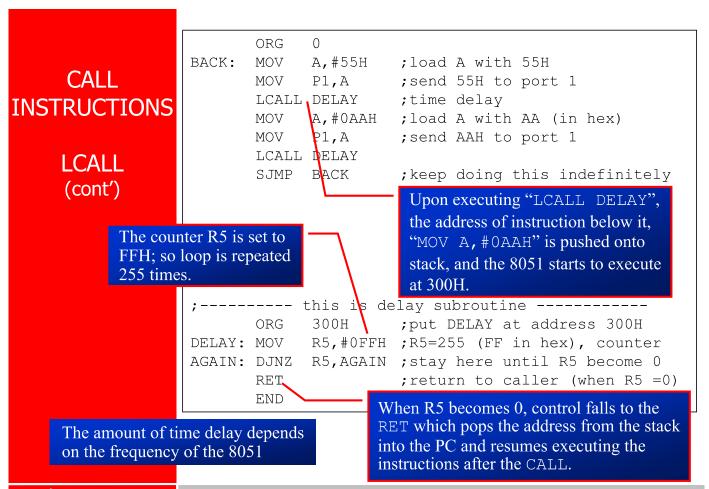
- 2-byte instruction
 - 11 bits are used for address within 2K-byte range



CALL INSTRUCTIONS LCALL

- When a subroutine is called, control is transferred to that subroutine, the processor
 - Saves on the stack the the address of the instruction immediately below the LCALL
 - Begins to fetch instructions form the new location
- After finishing execution of the subroutine
 - The instruction RET transfers control back to the caller
 - Every subroutine needs RET as the last instruction





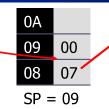


CALL INSTRUCTIONS

CALL Instruction and Stack

```
001 0000
                       ORG
                            0
002 0000 7455
                      MOV A, #55H ; load A with 55H
               BACK:
003 0002 F590
                       MOV P1,A
                                    ;send 55H to p1
004 0004 120300
                       LCALL DELAY
                                    ;time delay
005 (0007) 74AA
                      MOV A, #0AAH ; load A with AAH
006 0009 F590
                      MOV P1,A
                                    ; send AAH to p1
007 000B 120300
                      LCALL DELAY
008 ∮00E 80F0
                       SJMP BACK ; keep doing this
009 0010
010 010 ;-----this is the delay subroutine-----
011 0300
                       ORG
                            300H
012 0300
                DELAY:
013 0300 7DFF
                            R5,#0FFH ;R5=255
                       VOM
               AGAIN: DJNZ R5, AGAIN; stay here
014 (302 DDFE
015 0304 22
                       RET
                                    ;return to caller
016 0305
                       END
                                     ;end of asm file
```

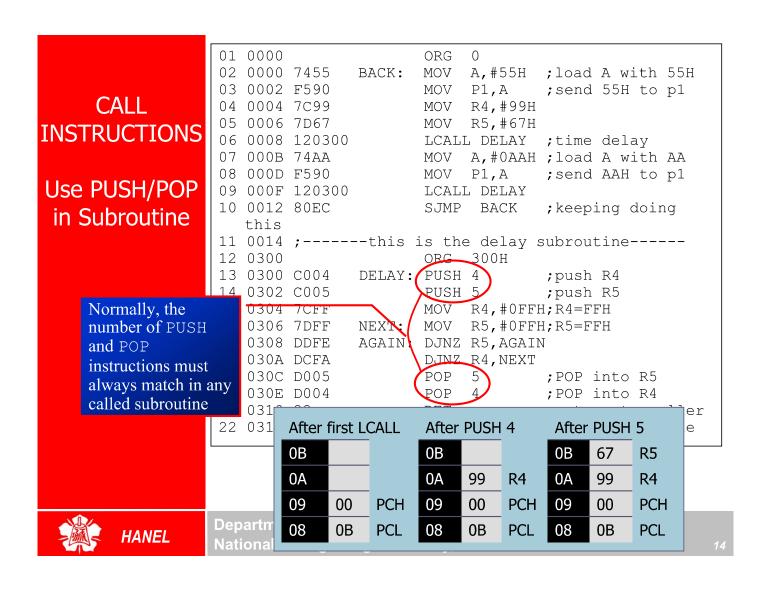
Stack frame after the first LCALL



Low byte goes first and high byte is last



HANEL



CALL INSTRUCTIONS

Calling Subroutines

```
; MAIN program calling subroutines
        ORG 0
                                     It is common to have one
        LCALL
MAIN:
                         SUBR 1
                                     main program and many
        LCALL
                         SUBR 2
                                     subroutines that are called
        LCALL
                         SUBR 3
                                     from the main program
HERE:
                         HERE
        SJMP
;----end of MAIN
SUBR 1: ...
                                     This allows you to make
        . . .
                                     each subroutine into a
        RET
  -----end of subroutine1
                                     separate module
                                     - Each module can be
SUBR 2: ...
                                     tested separately and then
        . . .
                                     brought together with
        RET
                                     main program
  -----end of subroutine2
                                     - In a large program, the
SUBR 3: ...
                                     module can be assigned to
        . . .
                                     different programmers
        RET
        ----end of subroutine3
                         ;end of the asm file
        END
```



CALL INSTRUCTIONS ACALL

- The only difference between ACALL and LCALL is
 - ➤ The target address for LCALL can be anywhere within the 64K byte address
 - ➤ The target address of ACALL must be within a 2K-byte range
- The use of ACALL instead of LCALL can save a number of bytes of program ROM space



CALL INSTRUCTIONS

ACALL (cont')

```
ORG
BACK:
       VOM
             A, #55H
                        ;load A with 55H
       VOM
             P1,A
                        ;send 55H to port 1
                        ;time delay
       LCALL DELAY
             A, #OAAH
                        ; load A with AA (in hex)
       VOM
       VOM
             P1,A
                        ; send AAH to port 1
       LCALL DELAY
       SJMP
             BACK
                        ; keep doing this indefinitely
        . . .
       END
                        ;end of asm file
```

A rewritten program which is more efficiently

```
ORG
              0
                        ;load A with 55H
       VOM
              A, #55H
BACK:
       VOM
              P1,A
                        ;send 55H to port 1
       ACALL DELAY
                        ;time delay
                        ; complement reg A
       CPL
              Α
                        ; keep doing this indefinitely
       SJMP
              BACK
        . . .
       END
                        ;end of asm file
```



- CPU executing an instruction takes a certain number of clock cycles
 - > These are referred as to as *machine cycles*
- The length of machine cycle depends on the frequency of the crystal oscillator connected to 8051
- In original 8051, one machine cycle lasts 12 oscillator periods

Find the period of the machine cycle for 11.0592 MHz crystal frequency

Solution:

```
11.0592/12 = 921.6 kHz;
machine cycle is 1/921.6 kHz = 1.085 \mus
```



TIME DELAY FOR VARIOUS 8051 CHIPS (cont')

For 8051 system of 11.0592 MHz, find how long it takes to execute each instruction.

- (a) MOV R3, #55 (b) DEC R3 (c) DJNZ R2 target
- (d) LJMP (e) SJMP (f) NOP (g) MUL AB

Solution:

	Machine cycles	Time to execute
(a)	1	$1x1.085 \mu s = 1.085 \mu s$
(b)	1	$1x1.085 \mu s = 1.085 \mu s$
(C)	2	$2x1.085 \mus = 2.17 \mus$
(d)	2	$2x1.085 \mus = 2.17 \mus$
(e)	2	$2x1.085 \mu s = 2.17 \mu s$
(f)	1	$1x1.085 \mu s = 1.085 \mu s$
(g)	4	$4 \times 1.085 \mu\mathrm{s} = 4.34 \mu\mathrm{s}$



Delay Calculation Find the size of the delay in following program, if the crystal frequency is 11.0592MHz.

MOV A,#55H

AGAIN: MOV P1, A

ACALL DELAY

CPL A

SJMP AGAIN

;---time delay-----

DELAY: MOV R3,#200

HERE: DJNZ R3, HERE

RET

A simple way to short jump to itself in order to keep the microcontroller busy

HERE: SJMP HERE

We can use the following: SJMP \$

Solution:

Machine cycle

DELAY: MOV R3,#200 1 HERE: DJNZ R3,HERE 2

RET 2

Therefore, $[(200x2)+1+2]x1.085 \mu s = 436.255 \mu s$.



Increasing
Delay Using
NOP

Find the size of the delay in following program, if the crystal frequency is 11.0592MHz.

Machine Cycle

DELAY:	VOM	R3,#250	1
HERE:	NOP		1
	DJNZ	R3,HERE	2
	RET		2

Solution:

The time delay inside HERE loop is $[250\,(1+1+1+1+2)\,]\,\text{x1.085}\,\mu\,\text{s} = 1627.5\,\mu\,\text{s}.$ Adding the two instructions outside loop we have $1627.5\,\mu\,\text{s} + 3\,$ x $1.085\,\mu\,\text{s} = 1630.755\,\mu\,\text{s}$



Large Delay
Using Nested
Loop

Find the size of the delay in following program, if the crystal frequency is 11.0592MHz.

Machine Cycle

DELAY: MOV R2,#200 1
AGAIN: MOV R3,#250 1
HERE: NOP 1
NOP 1
DJNZ R3,HERE 2
DJNZ R2,AGAIN 2
RET 2

Notice in nested loop, as in all other time delay loops, the time is approximate since we have ignored the first and last instructions in the subroutine.

Solution:

For HERE loop, we have $(4x250) \times 1.085 \, \mu \, \mathrm{s} = 1085 \, \mu \, \mathrm{s}$. For AGAIN loop repeats HERE loop 200 times, so we have $200 \times 1085 \, \mu \, \mathrm{s} = 217000 \, \mu \, \mathrm{s}$. But "MOV R3,#250" and "DJNZ R2,AGAIN" at the start and end of the AGAIN loop add $(3x200 \times 1.805) = 651 \, \mu \, \mathrm{s}$. As a result we have $217000 + 651 = 217651 \, \mu \, \mathrm{s}$.



Delay Calculation for Other 8051

- Two factors can affect the accuracy of the delay
 - Crystal frequency
 - The duration of the clock period of the machine cycle is a function of this crystal frequency
 - > 8051 design
 - The original machine cycle duration was set at 12 clocks
 - Advances in both IC technology and CPU design in recent years have made the 1-clock machine cycle a common feature

Clocks per machine cycle for various 8051 versions

Chip/Maker	Clocks per Machine Cycle
AT89C51 Atmel	12
P89C54X2 Philips	6
DS5000 Dallas Semi	4
DS89C420/30/40/50 Dallas Semi	1



Delay Calculation for Other 8051 (cont') Find the period of the machine cycle (MC) for various versions of 8051, if XTAL=11.0592 MHz.

(a) AT89C51 (b) P89C54X2 (c) DS5000 (d) DS89C4x0

Solution:

- (a) 11.0592 MHz / 12 = 921.6 kHz; MC is $1/921.6 \text{kHz} = 1.085 \, \mu \, \text{s} = 1085 \, \text{ns}$
- (b) 11.0592MHz/6 = 1.8432MHz; MC is 1/1.8432MHz = 0.5425 μ s = 542ns
- (c) 11.0592 MHz / 4 = 2.7648 MHz; MC is $1/2.7648 \text{MHz} = 0.36 \, \mu \, \text{s} = 360 \, \text{ns}$
- (d) 11.0592 MHz / 1 = 11.0592 MHz; MC is $1/11.0592 \text{MHz} = 0.0904 \, \mu \, \text{s} = 90 \, \text{ns}$



Delay Calculation for Other 8051 (cont')

Instruction	8051	DSC89C4x0
MOV R3, #55	1	2
DEC R3	1	1
DJNZ R2 target	2	4
LJMP	2	3
SJMP	2	3
NOP	1	1
MUL AB	4	9

For an AT8051 and DSC89C4x0 system of 11.0592 MHz, find how long it takes to execute each instruction.

- (a) MOV R3, #55 (b) DEC R3 (c) DJNZ R2 target
- (d) LJMP (e) SJMP (f) NOP (g) MUL AB

Solution:

DOTACTOII.						
AT8051	DS89C4x0					
(a) $1 \times 1085 \text{ns} =$	$1085ns$ $2 \times 90ns = 180ns$					
(b) $1 \times 1085 \text{ns} =$	$1085ns 1 \times 90ns = 90ns$					
(c) $2 \times 1085 \text{ns} =$	$2170ns$ $4 \times 90ns = 360ns$					
(d) $2 \times 1085 \text{ns} =$	$2170ns$ $3 \times 90ns = 270ns$					
(e) $2 \times 1085 \text{ns} =$	$2170ns$ $3 \times 90ns = 270ns$					
(f) $1 \times 1085 \text{ns} =$	$1085ns 1 \times 90ns = 90ns$					
(g) $4 \times 1085 \text{ns} =$	$4340ns$ $9 \times 90ns = 810ns$					

