

# Computer Organization & Architecture (Instruction Set Architecture)

Dr. Sonu Lamba



Department of Computer Science and Engineering  
The LNM Institute of Information Technology Jaipur

August 28, 2020

# Parameter Passing

- What is parameter passing?
- Ways of parameter passing.

# Parameter Passing through Registers

## Calling program

Load	R2, N	Parameter 1 is list size.
Move	R4, #NUM1	Parameter 2 is list location.
Call	LISTADD	Call subroutine.
Store	R3, SUM	Save result.
:		

## Subroutine

LISTADD:	Subtract	SP, SP, #4	Save the contents of
	Store	R5, (SP)	R5 on the stack.
	Clear	R3	Initialize sum to 0.
LOOP:	Load	R5, (R4)	Get the next number.
	Add	R3, R3, R5	Add this number to sum.
	Add	R4, R4, #4	Increment the pointer by 4.
	Subtract	R2, R2, #1	Decrement the counter.
	Branch_if_[R2]>0	LOOP	
	Load	R5, (SP)	Restore the contents of R5.
	Add	SP, SP, #4	
	Return		Return to calling program.

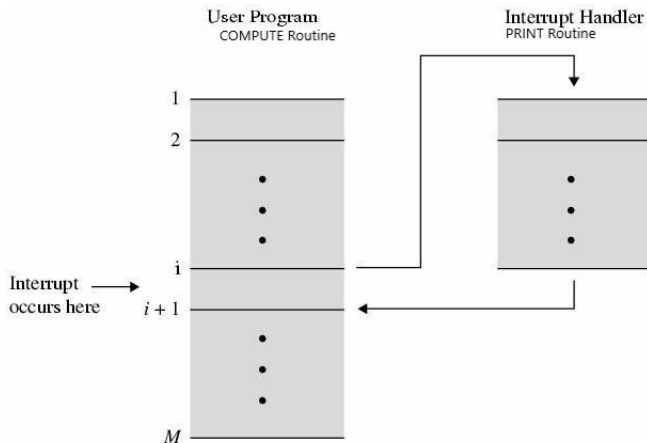
# Parameter Passing through Stack

	Move	R2, #NUM1	Push parameters onto stack.
	Subtract	SP, SP, #4	
	Store	R2, (SP)	
	Load	R2, N	
	Subtract	SP, SP, #4	
	Store	R2, (SP)	
	Call	LISTADD	Call subroutine (top of stack is at level 2).
	Load	R2, 4(SP)	Get the result from the stack
	Store	R2, SUM	and save it in SUM.
	Add	SP, SP, #8	Restore top of stack (top of stack is at level 1).
	:		
LISTADD:	Subtract	SP, SP, #16	Save registers
	Store	R2, 12(SP)	
	Store	R3, 8(SP)	
	Store	R4, 4(SP)	
	Store	R5, (SP)	(top of stack is at level 3).
	Load	R2, 16(SP)	Initialize counter to n.
	Load	R4, 20(SP)	Initialize pointer to the list.
	Clear	R3	Initialize sum to 0.
LOOP:	Load	R5, (R4)	Get the next number.
	Add	R3, R3, R5	Add this number to sum.
	Add	R4, R4, #4	Increment the pointer by 4.
	Subtract	R2, R2, #1	Decrement the counter.
	Branch_if_[R2]>0	LOOP	
	Store	R3, 20(SP)	Put result in the stack.
	Load	R5, (SP)	Restore registers.
	Load	R4, 4(SP)	
	Load	R3, 8(SP)	
	Load	R2, 12(SP)	
	Add	SP, SP, #16	(top of stack is at level 2).
	Return		Return to calling program.

# Interrupt and Traps

- What is Interrupt?
- Types of Interrupts
- Need of Interrupts?

# Interrupt Service Routine



# Example

- Consider the following sequence of micro-operations.

$MBR \leftarrow PC$

$MAR \leftarrow X$

$PC \leftarrow Y$

$Memory \leftarrow MBR$

Which one of the following is a possible operation performed by this sequence?

- (A) Instruction fetch
- (B) Operand fetch
- (C) Conditional branch
- (D) Initiation of interrupt service