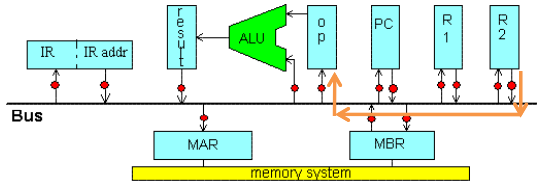
Last of Microcode?

COMP375

Computer Architecture and Organization

Register Indirect with Offset

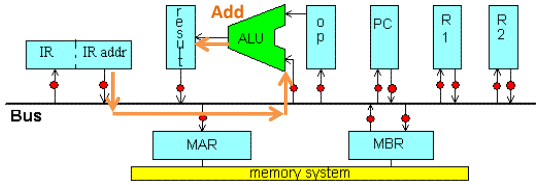
Add to R1 using Register Indirect with Offset with R2 as the index



bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
				X						X					
	X		X											add	
		X									X				read
				X				X							wait
			X										X	add	
		X					X								

Register Indirect with Offset

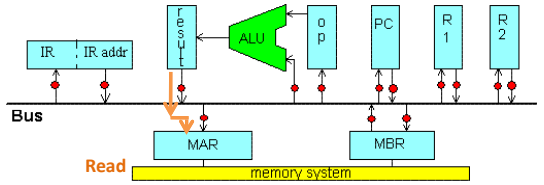
Add to R1 using Register Indirect with Offset with R2 as the index



bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
				X						X					
	X		X											add	
		X									X				read
				X				X							wait
			X										X	add	
		X					X								

Register Indirect with Offset

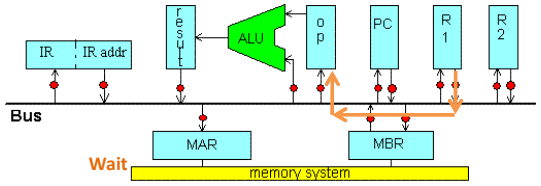
Add to R1 using Register Indirect with Offset with R2 as the index



bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
				X						X					
	X		X											add	
		X									X				read
				X				X							wait
			X										X	add	
		X					X								

Register Indirect with Offset

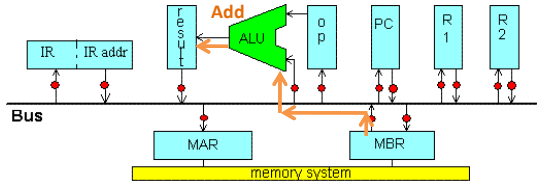
Add to R1 using Register Indirect with Offset with R2 as the index



bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
				X						X					
	X		X											add	
		X									X				read
				X				X							wait
			X										X	add	
		X					X								

Register Indirect with Offset

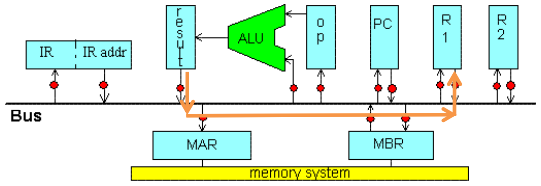
Add to R1 using Register Indirect with Offset with R2 as the index



bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
				X						X					
	X		X											add	
		X									X				read
				X				X							wait
			X										X	add	
		X					X								

Register Indirect with Offset

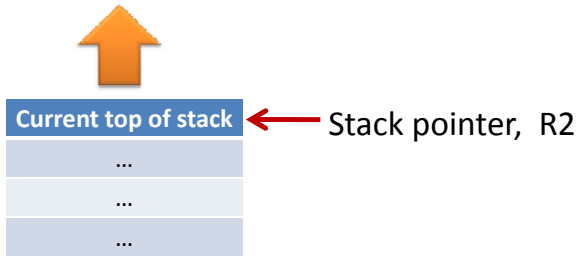
Add to R1 using Register Indirect with Offset with R2 as the index



bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
				X						X					
	X		X											add	
		X									X				read
				X				X							wait
			X											add	
		X					X								

Pop Instruction

- Read the memory location whose address is in the stack pointer



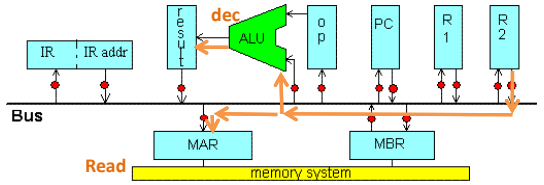
Pop Instruction

- Decrement the stack pointer

Stack pointer, R2

Pop R1

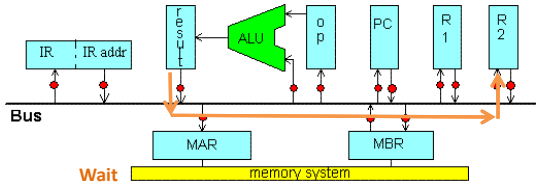
Read the address in R2 and decremented R2.



bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
			X							X	X			dec	read
		X							X						wait
							X						X		

Pop R1

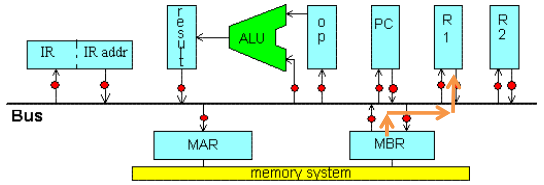
Save result in R2. Wait for read.



bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
			X							X	X			dec	read
		X							X						wait
							X						X		

Pop R1

Copy top of stack data to R1.



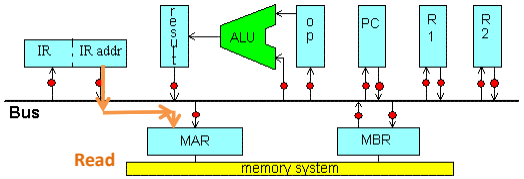
bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
			X							X	X			dec	read
		X							X						wait
							X						X		

Memory Operand Instruction

`inc dog`

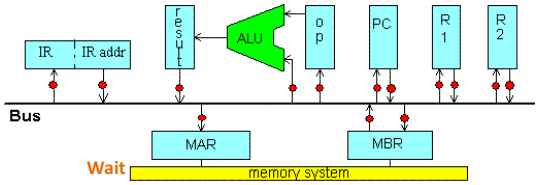
- The value at the memory location (dog) is incremented by one.
- Direct memory addressing
- Calculations occur in the CPU
  - Read memory value
  - Increment number
  - Store value back to memory

Read memory location dog



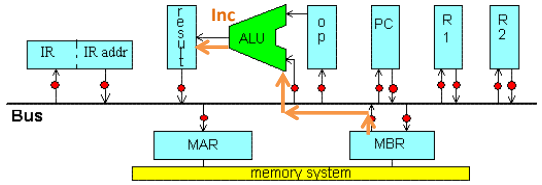
bus → IR	IR adr → bus	result → bus	bus → ALU	bus → oprnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
	X										X				read
															wait
			X										X	inc	
		X										X			write
															wait

Wait for the read



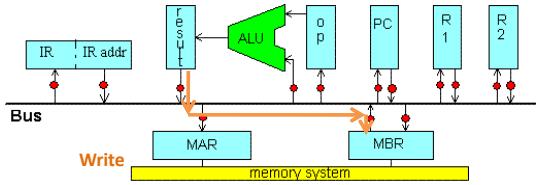
bus → IR	IR adr → bus	result → bus	bus → ALU	bus → oprnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
	X										X				read
															wait
			X										X	inc	
		X										X			write
															wait

Copy dog to the ALU and inc



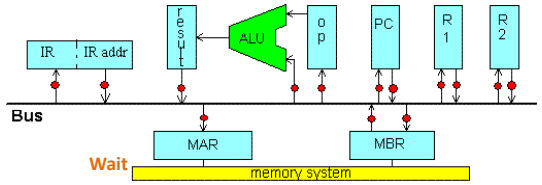
bus → IR	IR adr → bus	result → bus	bus → ALU	bus → oprnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
	X										X				read
															wait
			X										X	inc	
		X										X			write
															wait

Write the results back to dog



bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
	X										X				read
															wait
			X										X	inc	
		X										X			write
															wait

inc dog

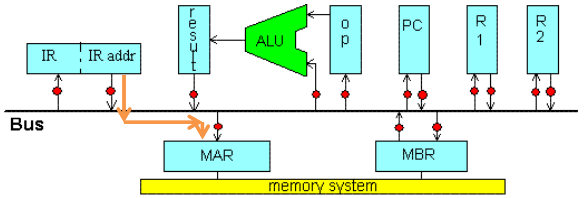


bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
	X										X				read
															wait
			X										X	inc	
		X										X			write
															wait

store R2,camel

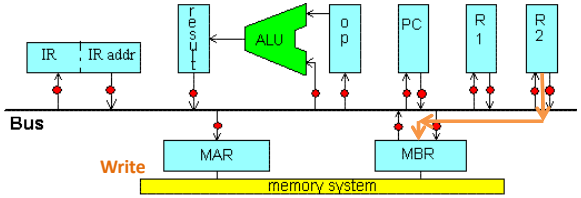
- A store instruction write the value of the register to address
- With direct memory addressing, the address is in the instruction

Put the address in the MAR



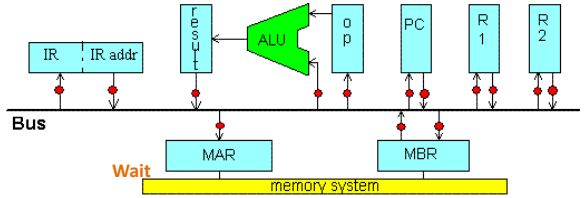
bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
	X										X				
										X		X			write
															wait

Write the data, R2



bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
	X										X				
										X		X			write
															wait

Wait for the Write to complete



bus → IR	IR adr → bus	result → bus	bus → ALU	bus → opnd	bus → PC	PC → bus	bus → R1	R1 → bus	bus → R2	R2 → bus	bus → MAR	bus → MBR	MBR → bus	ALU fun	Mem func
	X										X				
										X		X			write
															wait